Euclides Pinheiro de Melo

Desenvolvimento de um software para jogadores de loterias

Florianópolis – SC
2007

Euclides Pinheiro de Melo

Desenvolvimento de um software para jogadores de loterias

Monografia apresentada ao programa de
Bacharelado em Sistemas de Informação
da Universidade Federal de Santa Catarina
como requisito parcial para obtenção do grau
de Bacharel em Sistemas de Informação.

Orientador: Dalton Francisco de Andrade

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO

Florianópolis – SC
2007

Monografia de graduação sob o título Desenvolvimento de um software para jogadores de loteria, defendida por Euclides Pinheiro de Melo e aprovada em 15 de junho de 2007, em Florianópolis, Santa Catarina, pela banca examinadora constituída pelos professores:

"In lottery, you only have to win once."
Autor desconhecido

Sumário

Resumo

Vencer na loteria tem sido o sonho de muitas pessoas ao longo da história da humanidade. As pessoas vêm constantemente criando e inventando meios para fazer o sonho se tornar o mais próximo e real, e em geral, estes meios exigem "trabalho braçal", como a realização de longas análises sobre resultados passados, o desdobramento de apostas ou o preenchimento manual de um grande número de cartões.

O objetivo deste trabalho consiste em fornecer um software que auxilie o jogador a ter um meio de pôr em prática suas idéias, sem precisar dispender um longo período de tempo com tarefas manuais, deixando o apostador concentrar-se na parte que demanda mais inteligência.

Inicialmente, o trabalho apresenta informações sobre loterias em geral e sobre sistemas para jogadores de loterias; na seqüência, passa a detalhar os requisitos desse software específico, resultando no projeto e implementação do mesmo, concluindo com a disponibilização da versão 1.0 e seu respectivo código fonte gratuitamente para todos aqueles interessados no assunto.

Palavras chave: loterias; análise estatística; métodos de enumeração; software.

Abstract

Winning in a lottery has always been the dream of many people since the early days of lottery drawings, which started to take place many centuries ago.

At the same time, people have always been seeking ways to improve their chances of becoming a winner, usually trying to discover patterns that might be present in the past results, and putting their bets on the possible results that seemed to them the most probable to occur. However, both the statistical analysis of previous data and the choice of a combination of bets satisfying some established criteria often implies in hard and repetitive mechanical work, which turns out to be even worse when it becomes necessary to fill a large number of bet cards manually. With this scenario in mind, the objective of this work is to supply a software that can help a wagerer to formulate and lay his bets without wasting a long time with dull tasks, as mentioned.

The first part of this work provides some general information about lotteries and lottery systems (software); in the sequence, a specific software developed by the author is presented and implemented, which aims at the fulfillment of the pertaining requisites regarding the analysis and automation of bets. Finally, the complete source code of that particular software, (Version 1.0), is offered to be freely distributed to all those interested in the subject.

Keywords: lottery, statistical analysis, enumeration methods, software.

1 Introdução

Vencer. Seja por fortuna, poder, ou simplesmente pelo gostinho de chegar lá, o homem sempre tenta planejar algum meio de conseguir seus objetivos. Ao invés de se deixar levar pelo puro acaso, muitos tentam bolar estratégias, simular possibilidades, para tentar chegar a um plano vencedor.

Esta característica humana está presente em diversos acontecimentos da vida; a profissão que irá exercer, uma partida de *poker*, a guerra, o clima, o caminho para casa, entre muitos outros exemplos. Cada problema tem suas particularidades, grau de certeza e dificuldade de análise. Por exemplo: escolher o caminho de casa envolve algumas poucas variáveis, como distância, grau de periculosidade do lugar; é um problema razoavelmente simples de chegar a uma solução boa. Porém, existem problemas mais complicados, onde não se tem muitas certezas: numa partida de *poker* (justa), você nunca poderá saber as cartas do seu adversário. Mesmo não sabendo-se as cartas do oponente, isto não impediu o homem, ao longo do tempo, de tentar imaginar meios de vencer o oponente sem ter que contar apenas com a sorte.

Entre os muitos problemas difíceis de resolver, um bastante popular é o problema das loterias. A grosso modo, o problema das loterias consiste em descobrir que dezenas serão sorteadas no próximo concurso, ou que aposta, ou conjunto de apostas, terá acertado o maior número de dezenas possível no próximo concurso.

Um fato aceito é que não é possível prever exatamente o próximo resultado, assim como não há meios de saber de certeza as cartas na mão do adversário numa partida de *poker*. Mas isso não impede a mente humana de imaginar meios de chegar perto, tentando analisar as variáveis que estão presentes e imaginar planos, cenários possíveis para tentar tomar uma decisão melhor.

É neste cenário que entra um software para jogadores de loterias. Um software para jogadores de loterias não é uma caixa mágica, onde aperta-se um botão e consegue-se as dezenas que irão sair no próximo concurso. Tal recurso, apesar de desejável, não é possível de ser concebido, devido a natureza do problema. O que um bom software para loterias pode fazer, para auxiliar o jogador, é reunir as informações disponíveis e permitir que o usuário possa visualizar elas de maneira agradável (utilizando-se de interface gráfica, dados sob a forma de gráficos), auxiliando a sua tomada de decisão para o próximo concurso. Outra característica é de automatizar tarefas repetitivas, que podem ser feitas rapidamente utilizando-se um computador, que se fossem feitas pelo homem, seriam tediosas e passíveis de erros.

O tipo de loteria abordado é a loteria clássica, onde o apostador escolhe suas dezenas de um conjunto definido, e o(s) ganhador(es) é(são) quem acertar todas ou parte das dezenas sorteadas. Exemplos deste tipo de loteria: Quina, Lotofácil, Mega sena, Dupla sena, etc (da Caixa Econômica Federal); *pick-6, pick-5, pick-4* (loterias americanas), entre muitas outras.

A concepção de um software para loterias é o foco deste trabalho. Ao longo do texto, serão detalhadas as etapas de análise e projeto do software, levando em conta os anseios dos jogadores de loterias, o que eles desejam que seja automatizado, que espécie de análises irão querer, ou seja, que estratégias eles irão querer que o software os ajude para tentar resolver o problema. Ao final, há a implementação do que foi analisado e projetado. Devido o problema das loterias não ser exato, há inúmeras maneiras de se encarar e abordar, sendo campo fértil para novas estratégias que a mente humana vai imaginando.

1.1 Objetivos

O objetivo deste trabalho é disponibilizar aos apostadores de loteria um software (livre) que facilite aos jogadores implementar, analisar e jogar suas estratégias, sem julgar a validade das idéias inseridas pelos jogadores, tão pouco garantir que as estratégias tentadas pelos jogadores sejam vencedoras. O software irá facilitar o trabalho do apostador, e não garantir vitórias (ganhos). Isso depende apenas da estratégia do apostador, ficando ele com total responsabilidade pelos resultados (bons ou ruins) de sua estratégia.

1.2 Objetivos específicos

Desenvolver um software para jogadores de loterias que contenha as seguintes características / funcionalidades:

1) Carga dos resultados de loterias via internet;
2) permitir análises sobre os resultados passados que foram carregados, e apresentar estas análises de forma visual (tipicamente gráficos);
3) gerar combinações de jogos (conhecidos tipicamente como bolões);
4) permitir restrições sobre jogos combinados (conhecidos tipicamente como filtros);
5) realizar conferência dos jogos gerados;
6) imprimir os jogos gerados em volantes de loterias;
7) software deverá ser extensível para outras loterias similares;
8) software deverá ser fácil de adicionar novas funcionalidades;
9) software deverá rodar pelo menos em Linux e Windows;
10) versões em português do Brasil e inglês americano, não sendo difícil adaptar para outros idiomas;
11) página web para divulgação do software / projeto.

1.3 Justificativa

É comum as pessoas terem idéias (as vezes malucas) para tentar gerar um jogo ou bolão de loterias. Independente se a idéia for boa ou ruim, muitas vezes o que estas pessoas precisam, para pôr em prática, é de um suporte para automatizar tarefas tediosas, repetitivas, ou auxílio para realizar alguma análise, para comprovar alguma idéia que esta pessoa teve. Por isso temos os objetivos citados acima, pois apesar de muitas vezes o usuário ter o desejo de pôr em prática um jogo grande, um bolão para a empresa, ninguém gosta de ter que preencher manualmente vários cartões, ou combinar cada cartão manualmente. Assim como ninguém gosta de ter que contar, resultado a resultado, para saber se deu mais pares ou ímpares até hoje.

Um software que cumpra os objetivos acima, irá facilitar em muito a vida dos jogadores de loterias, facilitar os famosos bolões que acontecem dentro dos órgãos públicos e empresas privadas. Isso promove maior sinergia entre os colaboradores e acentua o espírito de equipe.

Outro resultado importante de tal software, é que auxiliando os jogadores, irá aumentar o volume de jogos realizados, consequentemente aumentado o volume de dinheiro apostado, aumentando assim a arrecadação que as loterias promovem para projetos sociais. Mais informações podem ser vistas na referência [CAIXA01].

As questões de multi-plataforma, multi-idioma e página web para divulgação vêm com o desejo de aumentar o escopo de atuação do software, considerando que diversos países ao redor do globo tem seus sistemas de loterias.

Utilizar software livre, e dispor o projeto sob uma licença livre vêm para, novamente, ampliar o escopo de atuação do programa, pois sob tal licença, qualquer desenvolvedor poderá reaproveitar / personalizar o programa para suas necessidades.

1.4 Limitações

Como dito, há inúmeras estratégias para se encarar o problema das loterias. Neste documento, análise e implementação, serão ouvidos os jogadores e consultada literatura a respeito para se obter algumas estratégias das inúmeras possíveis, logo apenas algumas das inúmeras análises serão implementadas. Mas como o sistema será facilmente extensível (objetivo e requisito), não será difícil terceiros adicionarem suas técnicas.

O projeto do software permite extensibilidade para qualquer tipo de loteria similar à pretendida. A implementação, para validar o projeto, conceberá dois tipos de loterias: Quina e Lotofácil, da Caixa Econômica Federal.

1.5 Organização do trabalho

Este trabalho está dividido da seguinte maneira: o capítulo 2 descreve loterias e discute probabilidades. O capítulo 3 dá uma visão geral sobre sistemas para jogadores de loterias. Os capítulos 4, 5 e 6 tratam da solução para o problema (análise de requisitos, projeto e implementação, respectivamente), enquanto que o capítulo 7 exibirá o sistema em sua versão 1.0, meta definida para este trabalho.

Durante a fase de projeto e implementação, haverá a confecção e publicação do site do projeto, de acordo com um dos objetivos estipulados. Assim que o desenvolvimento do software progrida, novas informações e versões do software serão publicadas.

2 Loterias

Este capítulo tem o propósito de dar uma visão geral do jogo.

A seção 2.1 fala do histórico do jogo. A seção 2.2 mostra as loterias brasileiras. A seção 2.3 mostra a dificuldade de se ganhar na loteria (probabilidades). Por fim, a seção 2.4 mostra que loterias não são apenas probabilidades, envolve a crença das pessoas.

2.1 Histórico

Loterias existem praticamente desde os tempos mais remotos da humanidade. Há inúmeras passagens da história que contém referências do uso de loterias para dividir bens, financiar grandes obras ou a guerra. Algumas passagens interessantes [NASPL]:

- 1200AC: Moisés realizou um tipo de loteria para distribuir as terras à oeste do Rio Jordão (Num 26, 52-56);
- 100DC: Dinastia Hun criou na China, o jogo Keno (um tipo de loteria). Os fundos arrecadados através desta loteria foram utilizados para defesa, principalmente para financiar a construção da muralha da china;
- 1465: Loterias foram utilizadas na Bélgica afim de arrecadar fundos para a construção de capelas, asilos e canais;
- 1567: Rainha Elizabeth I estabelece a primeira loteria estadual inglesa. Prêmios incluem dinheiro, prataria e tapeçaria. Eram utilizados 400.000 bilhetes.
- 1790 até a Guerra civil americana: 50 centros de universidades, 300 escolas e 200 igrejas foram erguidas com dinheiro de loterias. Universidades famosas como Harvard, Yale, Princeton e Columbia foram fundadas com dinheiro de loterias.
- 1999: Criada a “The World Lottery Association” (associação mundial das loterias) que contém 63 países membro (inclusive o Brasil).

Apesar das loterias e dos sistemas de loterias terem evoluído drasticamente através dos anos, seus propósitos continuam os mesmos: arrecadar dinheiro para financiar projetos do governo. Devido às loterias arrecadarem grandes somas em dinheiro, a maior parte dos governos do mundo (incluindo o Brasil) não permitem o setor privado explorar este mercado.

2.2 Loterias no Brasil

A loteria nacional é administrada pela Caixa Econômica Federal, desde 1962, quando realizou o primeiro sorteio da loteria federal. Desde então, a oferta de jogos de loteria tem crescendo cada vez mais.

A partir de 1980, a Caixa adotou um sistema inovador para a realização dos sorteios: criou o chamado caminhão da sorte. Neste caminhão, que percorre o Brasil, é realizado o sorteio das loterias, e o povo da cidade em que o caminhão está pode assistir à apuração. Além de um auditor designado, a população pode desta maneira auditar o resultado da loteria e do processo. A idéia do caminhão, segundo a instituição, é a de transparência nos sorteios, para que os apostadores continuem confiantes na idoneidade do sistema. Enquanto um caminhão fica na cidade do sorteio, outro caminhão vai viajando para a outra e se instalando, para o sorteio seguinte. Este sistema tem se mostrado eficaz, considerando que o volume de jogos lotéricos só vem crescendo através dos anos.

Até o presente momento, os jogos disponíveis pela Caixa pertinentes a este trabalho são: [CAIXA03]

- Mega Sena

É a loteria mais difícil de acertar (e consequentemente, com prêmios mais altos).

Marcar: 6 de 60 números.

Premia com 4, 5, 6.

	sena	quina	quadra
Probabilidade de acerto, 1 em	50.063.860	154.518	2.332

- Dupla Sena

Têm como diferencial a existência de dois sorteios por aposta, logo o apostador tem duas chances de ganhar por aposta.

Marcar: 6 de 50 números.

Premia com 4, 5, 6.

	sena	quina	quadra
Probabilidade de acerto, 1 em	15.890.700	60.192	1.119

- Quina

Uma das mais antigas. Tem este nome por ser sorteados apenas cinco números. Tem como característica a possibilidade de jogar com mais números por volante, com preço reduzido.

Marcar: 5, 6 ou 7 de 80 números.

Premia com 3, 4, 5.

	quina	quadra	terno
(5) Probabilidade de acerto, 1 em	24.040.016	64.106	866
(6) Probabilidade de acerto, 1 em	4.006.669	21.657	445
(7) Probabilidade de acerto, 1 em	1.144.762	9.409	261

- Lotomania

Uma loteria recente. Tem como característica a quantidade de números elevada que o apostador precisa preencher por cartão.

Marcar: 50 em 100 números.

Premia com 0, 16, 17, 18, 19, 20.

Faixas	Probabilidade
20 números	1/11.372.635
19 números	1/352.551
18 números	1/24.235
17 números	1/2.776
16 números	1/472
00 números	1/11.372.635

- Lotofácil

Com o sucesso da Lotomania, veio depois a Lotofácil. Como o nome já diz, é uma loteria que oferece maiores chances ao apostador (probabilidade de acerto maior por aposta).

Marcar: 15 em 25.

Premia com 11, 12, 13, 14, 15.

Faixas	Probabilidade
11 números	1/11
12 números	1/59
13 números	1/691
14 números	1/21.791
15 números	1/3.268.760

Estas referidas loterias são pertinentes a este trabalho porque são loterias onde o usuário escolhe uma certa quantidade de números de um conjunto definido, e ganha se acertar parte ou todas as dezenas sorteadas.

A Caixa Econômica tem ainda outros tipos de loterias, que são a Loteca, Loteria Federal, Lotogol e Loteria Instantânea, totalizando com as citadas anteriormente, 9 tipos de jogos à escolha do apostador. O preço da aposta, para todas as citadas, varia de 50 centavos a 2 reais.

2.3 Probabilidades

Como pôde ser visto no tópico acima, as chances de ganho em uma loteria são muito pequenas. Quando falamos em chance de ganhar, estamos nos referindo a probabilidade de, em um sorteio (um evento), de aparecer a combinação escolhida dentre as possíveis (os números sorteados coincidirem com os escolhidos).

Por exemplo: um jogador realiza uma aposta no jogo "Lotofácil". A probabilidade da aposta que ele realizou ser a vencedora (acertar todos os 15 números) é de 1 para 3.268.760. O motivo de a chance ser ruim é o fato de haver 3.268.760 combinações diferentes de 15 números em 25 dezenas possíveis. Cada jogo tem probabilidades de ganho diferentes devido a cada jogo ter um número de dezenas a escolher diferentes dos outros, além do conjunto possível de números ter tamanho diferente.

Se o mesmo jogador realizasse duas apostas diferentes no mesmo concurso, sua chance, para o jogo citado acima, seria de 2 para 3.268.760, ou 1 para 1.634.380. Isso quer dizer que, quanto maior a quantidade de apostas diferentes realizadas no mesmo concurso, maior a probabilidade de ganho do apostador. Observando os extremos, se o apostador pudesse jogar as 3.268.760 combinações possíveis, este teria probabilidade 1 de ganhar; se não realizasse aposta alguma teria probabilidade 0. Apesar de desejável ter probabilidade 1 de ganho, é inviável devido ao custo das apostas e a impossibilidade de gerar e apostar os 3.268.760 volantes de aposta.

É comum vários apostadores se unirem para fazer um jogo maior, ou seja, cada apostador ajuda com uma quantia em dinheiro para realizar um jogo unificado. São os famosos "bolões". Imagine um cenário onde 10 apostadores se unem para realizar uma aposta unificada na "Lotofácil". Cada apostador contribui com 10 volantes, totalizando assim 100 volantes (diferentes). A chance deste grupo acertar o premio máximo é de 1 em 32.687. Ainda é bastante difícil de se ganhar, mas é uma chance melhor do que cada jogador realizar um jogo individual de 10 volantes (chance de 1 em 326.876). As chances aumentam quanto maior a quantidade de apostas, e o custo fica menor quando é dividindo entre mais apostadores. Um grupo muito grande de apostadores reduz bastante o custo, porém também o prêmio terá de ser dividido entre estes muitos. Como o prêmio de um concurso de loterias ultrapassa a casa dos milhões de reais, geralmente é um melhor negócio participar de um bolão em vez de jogar sozinho.

2.4 A mente do jogador: indo contra as probabilidades

Ao tomar decisões, o ser humano considera os fatos, os riscos, e coloca também um fator não muito preciso: a intuição. Decisões baseadas com forte influência de intuição podem acarretar em coisas boas, mas também más (costumam ser imprevisíveis as conseqüências). Esta seção visa mostrar como as decisões tomadas, quando envolvendo probabilidade e intuição, podem ser diferentes do que se esperaria que o decisor tomaria.

2.4.1 "Prospect Theory"

Em 1979, Kahneman e Tversky publicaram no periódico Econometrica o trabalho "Prospect Theory: An analysis of decisions under risk". Este artigo provou ser um dos trabalhos de maior influência na pesquisa de decisões sob risco, apresentando uma crítica, fundamentada em experimentos, da teoria clássica baseada na maximização da utilidade esperada e oferecendo um modelo alternativo.

A teoria clássica, em resumo, diz que o indivíduo sempre irá agir de acordo com as melhores probabilidades (aversão ao risco). Por exemplo, o indivíduo não apostaria numa loteria porque as chances são muito pequenas, seria melhor negócio guardar o valor da aposta e não jogar. Sabemos que na prática isso não acontece (as pessoas jogam na loteria), e é um dos pontos do modelo de Kahneman e Tversky.

As razões identificadas por Kahneman e Tversky da desobediência do princípio da maximização da utilidade esperada são chamadas de efeitos, discutidos a seguir [PAULO 01].

a) Efeito certeza:

Quando um ganho positivo tende a 100% de certeza, é geralmente preferido a outras opções com risco, mesmo se essas últimas puderem resultar em um prêmio mais alto, dentro de certos limites variáveis.

Foi feito um estudo com uma amostra de 95 indivíduos, e propostos dois problemas: no primeiro, o indivíduo pode escolher entre jogar uma loteria com 80% de chance de ganhar $4000 (ou seja, 20% de chance de sair com nada) e a outra opção era não jogar a loteria e ganhar $3000 (chance 100%). O segundo problema consistia em o indivíduo jogar uma loteria com 20% de chance de ganhar $4000, ou jogar outra loteria com chance de 25% de ganhar 3000.

Pela teoria da maximização da utilidade, o indivíduo deveria jogar sempre naquela em que, proporcionalmente às probabilidades, oferece maior ganho. Porém, o efeito certeza mexe com a cabeça dos indivíduos, que preferiram pegar os $3000 porque era um prêmio que ganhariam com certeza. Quando a probabilidade de ganhar $4000 era pequena e $3000 também, foram no prêmio mais alto (os $4000). Houve uma inversão da preferência de jogo, devido ao efeito certeza.

b) Efeito reflexão:

Quando as loterias envolvem perdas ao invés de ganhos, ou colocando-se sinais negativos nos prêmios, as preferências se invertem. Isto significa dizer que a aversão ao risco que verifica-se em loterias compostas apenas de prêmios positivos transforma-se em afeição ao risco para o caso de perdas.

As hipóteses de que os indivíduos evitam em suas decisões alternativas incertas (preferência por ganhos certos) ou optam pela mínima variabilidade (a variância de (3000, 1) é zero) também não são confirmadas pelos resultados obtidos experimentalmente.

Na pesquisa, a loteria (- 4000, 0.80) é preferida (92% das respostas) à loteria (-3000, 1), apesar de esta última ter um valor esperado maior e uma variância menor.

c) Efeito isolamento:

Entre alternativas que contêm componentes comuns, a escolha é feita com base nas partes que as distinguem. Esse critério de decisão pode ocasionar preferências inconsistentes. Considere-se, por exemplo, uma loteria composta de dois estágios.

No primeiro estágio existe uma probabilidade de 75% de encerrar-se o jogo sem ganhos nem perdas (L11), e uma chance de 25% de passar-se para o segundo estágio (L12), onde as possibilidades são L21: (4000, 0.80; 0, 0.20) e L22: (3000, 1). Investigado o comportamento decisório de 141 indivíduos nessa loteria composta, 110 pessoas (78%) optaram por L22. Este resultado opõe-se à seleção modal observada no problema 2 citado na discussão do Efeito Certeza, cujas alternativas eram idênticas às do presente problema após combinar-se as probabilidades dos primeiro e segundo estágios. Por quê essa discrepância? Aparentemente, os decisores ignoraram o primeiro estágio do jogo, fazendo suas opções como no problema 1. O efeito isolamento implica no aumento da atratividade da certeza contingente, relativamente a uma opção de risco com as mesmas probabilidades combinadas e ganhos.

Em resumo, o estudo de Kahneman e Tversky mostra que, quando as probabilidades estão nos extremos (perto de 100%, perto de 0%) as decisões dos indivíduos se tornam contraditórias ao modelo da máxima utilidade, e algumas vezes imprevistas. Um dos exemplos mostrou isso, o caso das loterias, onde os jogadores apostam, mesmo com ínfima chance de ganho, devido ao enorme prêmio. A loteria é um caso onde chance é perto de 0%.

Um exemplo que envolve perda seria quando as pessoas resolvem praticar esportes radicais. A probabilidade de acontecer uma tragédia (morte) é pequena, porém a perda é muito grande (a morte, uma perda “infinita”). Porém, as pessoas mesmo assim praticam esportes radicais, porque neste momento estão olhando para a probabilidade da perda que é “muito pequena”, e não da perda muito grande.

A prospect theory é citada neste trabalho apenas como uma justificativa de porque as pessoas jogam na loteria, não sendo usada no desenvolvimento do software.

2.4.2 Intuição e sistemas para loterias

Outro fator que afeta a mente das pessoas ao tomar suas decisões de jogar ou não numa loteria, são as crenças em determinados sistemas para loterias, ou crenças em fatores místicos, onde o individuo acredita que a probabilidade de ganho aumentou, quando na realidade, matematicamente, é a mesma.

Imaginemos um caso, em que um jogador em particular, sonhou com um conjunto de dezenas específicas. Este jogador em particular costuma jogar num jogo com 6 dezenas, e coincidentemente sonhou com 6 números. Este jogador vai se sentir com sorte, vai achar que suas chances de ganho aumentaram drasticamente, porque afinal, sonhou com 6 números! Sua decisão de jogar ou não teve uma influência em crenças, e não apenas na probabilidade.

Um exemplo que envolve sistemas para jogadores de loterias são as análises realizadas em resultados passados. Quando um jogador observa que, por exemplo, estão sendo sorteadas muitas dezenas altas, ele acredita que se jogar em dezenas altas, irá aumentar sua probabilidade de ganho, quando novamente, a probabilidade de o jogador ganhar com um cartão (independente de dezenas altas ou baixas) é a mesma (já que sorteios de loterias não são afetados por resultados passados, já que os sorteios não tem memória). Um efeito similar é o jogador acreditar que deveria jogar mais em dezenas baixas, porque estão atrasadas.

Estes fatores pesam nas decisões de jogo dos apostadores, e estes gostam de usar estes sistemas porque faz eles sentirem mais confiantes. É devido a mais este fator, que explica o motivo das pessoas jogarem em loterias, apesar da probabilidade ser muito pequena de ganhar.

3 Sistemas para jogadores de loterias

Este capítulo tem o propósito de descrever características gerais de sistemas para jogadores de loterias, sendo útil também para a análise de requisitos do software (capítulo 4).

A seção 3.1 diz a que serve um sistema para jogadores de loterias. A seção 3.2 mostra as funcionalidades comuns. A seção 3.3 mostra usos comuns (cenários comuns) de uso. A seção 3.4 aponta os principais benefícios de se utilizar um software deste tipo, em detrimento de não usar software algum para jogar na loteria. Por fim, a seção 3.5 aponta as limitações destes softwares.

3.1 Propósito

Um software para jogadores de loterias tem como propósito principal aliviar o usuário de tarefas que não demandam inteligência, ou seja, tarefas mecânicas e repetitivas; e não apenas aliviar o usuário, como também dar celeridade e conveniência ao processo de se jogar na loteria.

O processo de jogar na loteria, consiste, basicamente, nos seguintes passos representados na figura abaixo:

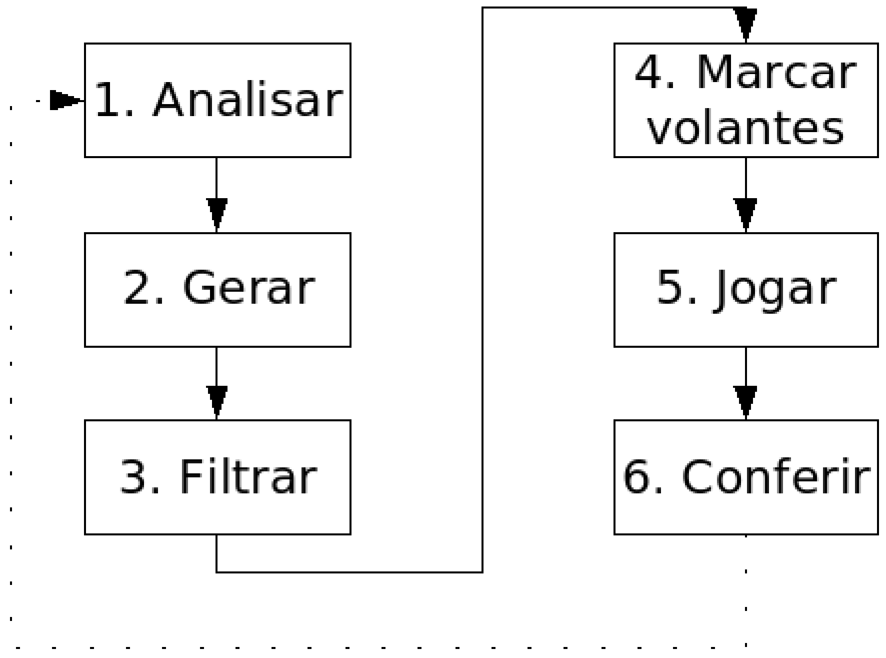


Figura 3.1.1: Processo básico para jogar na loteria

1. Analisar: consiste do passo do usuário pensar, pesquisar, coletar dados para realizar alguma análise em resultados passados ou outra fonte que julgue adequada para escolher dezenas a jogar ou gerar um bolão. Este passo é opcional.
2. Gerar: consiste em o usuário inventar um meio de combinar suas dezenas, gerando

tipicamente um bolão. Este passo é opcional.

3. Filtrar: restringir, usando alguma regra, os jogos gerados, a fim de reduzir os custos do jogo, ou fazer caber a quantidade de cartões na verba disponível. Este passo é opcional.
4. Marcar volantes: consiste no preenchimento físico dos volantes de loteria, a fim de se poder realizar a aposta em uma casa lotérica credenciada. Passo requerido.
5. Jogar: consiste no ato de submeter os volantes ao agente lotérico e realizar o pagamento. Passo requerido.
6. Conferir: averiguar se houve sucesso ou fracasso no jogo que apostou. Passo requerido.

Observando-se cada passo descrito, nota-se que existe uma série de atividades manuais que o usuário precisaria executar se não usar um software para jogador de loterias. Vamos a algumas:

- No caso da análise, em particular de resultados passados, teria que manualmente colher todos os resultados de loteria, e aplicar meios de visualizar estes dados de forma mais agradável. Ex: visualizar a distribuição de pares e ímpares nos resultados passados: colher cada resultado, contar em cada resultado quantos pares e ímpares deram, montar uma tabela ou gráfico exibindo o resultado da análise. Manualmente, é mais de um dia de trabalho;
- No caso de gerar, em particular em bolões, onde se costumam jogar, por exemplo, centenas ou milhares de volantes. Enumerar todos os números de cada volante é um trabalho extremamente repetitivo e demorado;
- No caso de filtrar, aplicar regras de rejeite ou aceite em cada volante, novamente é um trabalho demorado;
- No caso de marcar volantes, pintar cada célula de cada volante de loteria usando caneta é algo que pode de dias a semanas, dependendo do tamanho do jogo;
- No caso de conferir, examinar cada cartão para determinar a quantidade de erros e acertos, no caso de haver muitos volantes, demanda muito tempo e esforço;

Em suma, o propósito básico de um software para jogadores de loterias consiste em automatizar as tarefas descritas acima (podendo ainda oferecer outras facilidades mais) e deixar com o jogador a estratégia que ele julgue adequada para vencer. Deste modo, um software para jogadores de loteria é visto como uma ferramenta, algo que promove a eficiência, minimizar esforço para aquele que tem de executar alguma tarefa. Uma analogia: imagine que você precisa pregar um prego na parede. Você tem o prego, você tem a parede. É possível ser feito só com as mãos, mas o esforço e tempo serão muito grandes. A tarefa se torna mais fácil quando você tem uma **ferramenta** adequada para o problema: no caso, um martelo. Encare o software para jogadores de loteria como o martelo: ter a ferramenta não garante que o prego estará pregado, mas facilita a tarefa. Depende do utilizador da ferramenta para alcançar o objetivo.

3.2 Características gerais

Um software para jogadores de loteria possui, geralmente, as seguintes características:

- Realizar análises sobre os resultados passados;
- Gerar bolões;
- Filtrar bolões gerados;
- Imprimir os jogos em volantes de loteria;
- Conferir os jogos.

O que motiva os softwares para este ramo, terem estas características, é descrito na seção 3.1 acima, pois estas características cobrem o grande trabalho braçal de se jogar na loteria.

Tais características puderam ser encontradas na referência [GAIL01] e também nos seguintes softwares para jogadores de loteria existentes (maiores detalhes sobre eles, acesse o site):

- SpLoto (http://www.spolti.com.br/sploto/);
- Gerasorte (http://www.gerasorte.com.br/);
- Papasorte (http://www.loteria.com.br/);
- Sorte certa (http://www.sortecerta.com.br/);
- HtLoto (http://www.hypertech.com.br/);

3.3 Benefícios

O maior benefício, como já discutido, é centrado na redução de tempo e esforço para se ter o jogo pronto para ser jogado e sua posterior conferência. Mas além destes benefícios, pode-se apontar também:

- Redução de erros, uma vez que quem executa as tarefas repetitivas é o computador;
- Padronização e melhor controle dos jogos;
- Acesso fácil a todos os jogos já gerados;
- Exibição dos dados de forma visual;
- Facilidade de trocar jogos e análises entre outros jogadores.

3.4 Limitações

É comum, quando as pessoas ouvem sobre softwares de loteria, pensar que "inventou-se" um jeito garantido de ganhar, ou então pensar que "o software vai ganhar para mim". Como já discutido ao longo do trabalho, o software é uma ferramenta que você usa para facilitar a vida ao jogar na loteria, e depende da estratégia, de entrada do usuário para fornecer saída.

Ao acessar *sites* de produtos comerciais de loteria, é comum anunciarem que o software vai te fazer ganhar, mas esteja ciente que todo software deste tipo sofre desta limitação, que é a impossibilidade de garantir ganhos (claro, é possível garantir ganho se especificar alguns cenários, mas pense de modo geral).

Outra limitação está no fato de que o software nunca vai satisfazer todos os desejos de todos os jogadores: cada jogador tem sua estratégia única e particular que acha adequado para vencer na loteria. Os softwares disponíveis reúnem algumas estratégias comuns que boa parte dos jogadores gostam / costumam jogar e disponibilizam no software. No capítulo 5, projeto do software, será descrito as decisões de projeto que tentam contornar esta limitação encontrada em praticamente todos os softwares de loteria existentes: falta de **extensibilidade.**

4 Análise de requisitos do software

Através da análise de sistemas existentes e características gerais de softwares para jogadores de loteria (capítulo 3), leitura em livros especializados (HOWARD e JONES) e conversa informal com jogadores, este capítulo contém os requisitos funcionais e não funcionais do software proposto neste trabalho.

4.1 Glossário dos requisitos

- análise – aplicação de algum algoritmo específico sobre a base de resultados, resultando numa forma diferente e mais agradável de visualizar a referida base. Um exemplo: imagine uma base com dados pessoais de 100 usuários. Uma análise seria uma visualização mais específica ou agradável desta base, como por exemplo, exibir sob forma de gráfico de pizza a distribuição destes usuários por faixa etária. Note que é possível realizar "n" análises sobre a base, cada uma focando em alguma informação diferente ou tentando mostrar algum dado de forma diferente (gráfico, tabela..).
- análise altas x baixas – análise cujo objetivo é mostrar ao usuário a distribuição de dezenas abaixo e acima da dezena central por sorteio. Consulte a referência [GAIL01] na página 36 para mais detalhes.
- análise colunas – análise cujo objetivo é mostrar ao usuário a quantidade de dezenas em cada coluna, por resultado. Consulte a referência [GAIL01] na página 48 para mais detalhes.
- análise freqüência das dezenas – análise cujo objetivo é mostrar ao usuário , para cada dezena e cada resultado, quantas vezes ela têm sido sorteada. Consulte a referência [GAIL01] na página 53 para mais detalhes.
- análise linhas - análise cujo objetivo é mostrar ao usuário a quantidade de dezenas em cada linha, por resultado. Consulte a referência [GAIL01] na página 39 para mais detalhes.
- análise par x ímpar - análise cujo objetivo é mostrar ao usuário a distribuição de dezenas pares e ímpares, por sorteio. Consulte a referência [GAIL01] na página 35 para mais detalhes.
- base de resultados – entende-se por base de resultados o conjunto dos sorteios de uma loteria específica. Para cada tipo de loteria disponibilizado no sistema, haverá uma base de resultados. O formato da base é igual para todos os tipos de loteria, o que mudam são os dados e a existência física de um arquivo separado de base para cada tipo de loteria.
- filtro – aplicação de algum algoritmo específico sobre um jogo gerado, a fim de eliminar volantes de loteria. O critério de eliminação ou aceitação será passado como parâmetro ao algoritmo.
- filtro altas x baixas – tipo de filtro específico, destinado a eliminar volantes que contenham uma determinada quantidade de dezenas consideradas baixas e altas. Uma dezena é considerada alta se for maior que a dezena central, logo uma dezena é

considerada baixa se for menor ou igual que a dezena central. A dezena central é obtida dividindo-se a maior dezena por 2. Em caso de o resultado não ser uma dezena pertencente ao conjunto dos naturais, despreza-se as casas decimais.

- filtro colunas – tipo de filtro específico, destinado a eliminar volantes que contenham uma determinada quantidade de dezenas em cada coluna.
- filtro dezena – filtro destinado a eliminar volantes que contenham determinadas dezenas específicas.
- filtro linhas – filtro destinado a eliminar volantes que contenham determinada quantidade de dezenas por linha.
- filtro par x ímpar – filtro destinado a eliminar volantes que contenham determinada quantidade de dezenas pares e determinada quantidade de dezenas ímpares.
- forma de gráfico – um tipo de visualização de análise. Consiste em resumir os dados da análise, exibindo por exemplo, sob a forma de um gráfico de pizza, ou um gráfico de barras.
- forma de tabela – um tipo de visualização de análise. Consiste em exibir, de forma detalhada, a análise sobre cada resultado, usando uma estrutura tabular.
- gerador – um algoritmo específico destinado a gerar um conjunto de volantes de loteria.
- jogo gerado – um conjunto de volantes gerados.
- plugin – forma de extender um programa original, sem que haja necessidade de modificar o código ou recompilar o programa original. Consulte http://en.wikipedia.org/wiki/Plugin para maiores detalhes.
- URL – formato de endereço eletrônico. Consulte http://www.faqs.org/rfcs/rfc1738.html para maiores detalhes.
- verificador – algoritmo destinado a contar quantos acertos houve sobre cada volante de um jogo gerado. É necessário o fornecimento do resultado de loteria como entrada.
- volante – uma combinação de números possível de ser jogada em um determinado tipo de loteria.
- web – entende-se por web, no contexto deste trabalho, como um servidor que disponibiliza um arquivo usando o protocolo HTTP, acessível através da internet.

4.2 Requisitos em alto nível

O capítulo 1, seção 1.2 (objetivos específicos), dá em alto nível, os requisitos do sistema; ou seja, aquilo que o sistema deve fazer. Relembrando:

1. Carga dos resultados de loterias via internet.
2. Permitir análises sobre os resultados passados que foram carregados, e apresentar estas análises de forma visual (tipicamente gráficos).
3. Gerar combinações de jogos (conhecidos tipicamente como bolões).

4. Permitir restrições sobre jogos combinados (conhecidos tipicamente como filtros).
5. Realizar conferência dos jogos gerados.
6. Imprimir os jogos gerados em volantes de loterias.
7. Software deverá ser extensível para outras loterias similares.
8. Software deverá ser fácil de adicionar novas funcionalidades.
9. Software deverá rodar pelo menos em Linux e Windows.
10. Versões em português do Brasil e inglês americano, não sendo difícil adaptar para outros idiomas.
11. Página web para divulgação do software / projeto.

Nem todos os requisitos indicam algo que o sistema deve fazer, e estão presentes na listagem acima. São os chamados requisitos não funcionais. Tanto os funcionais e não funcionais serão detalhados nos tópicos abaixo. Termos em itálico na seção 4.2 e 4.3 serão explicados na seção 4.4, glossário dos requisitos.

4.3 Requisitos funcionais

Para cada requisito alto nível funcional, será detalhado abaixo, na seqüência O número corresponde ao requisito alto nível. São eles:

1. **Carga de resultados** - O sistema deve, na ausência de uma *base de resultados*, perguntar ao usuário se deseja realizar o download dos resultados via *web*. O sistema deve permitir, quando for do desejo do usuário, fazer o download dos resultados, e também quando for de desejo do usuário, exibir os resultados carregados. Para tal, será fornecida uma *URL* para download da base.
2. **Análises** - Realizar *análises* sobre a *base de resultados*. Para a versão 1.0 do sistema, deverão ser implementadas cinco análises: *análise par x ímpar, análise altas x baixas, análise linhas, análise colunas e análise freqüência das dezenas.* Estas análises serão detalhadas por *forma de tabela* e por *forma de gráfico*.
3. **Geradores** - O sistema deve, a partir das dezenas do usuário, ou então todas as dezenas possíveis, enumerar *volante* a *volante*, cada possível combinação que é formada a partir das dezenas escolhidas. Algoritmo na referência [MICHAEL01].
4. **Filtros** - O sistema deve, a partir de um *jogo gerado*, permitir que o usuário especifique regras para eliminar ou manter volantes com determinadas características. Para a versão 1.0 do sistema, deverá ser implementado cinco *filtros*: *filtro par x ímpar, filtro altas x baixas, filtro linhas, filtro colunas e filtro dezena.* Note que neste caso, os filtros estão casando com as análises que deverão ser implementadas, mas não necessariamente uma análise precisará ter um filtro equivalente, apesar de ser comum (uma vez que determinada análise pode gerar o desejo de manter ou retirar cartões com determinada característica observada na análise).
5. **Verificadores** - O sistema deve, para um *jogo gerado* qualquer, permitir que o usuário

entre com um resultado, e então o sistema exibe um relatório mostrando os acertos de cada *volante* que pertence ao jogo gerado. Ao final do relatório, o sistema exibe um sumário dizendo quantos cartões fizeram tantos pontos, como mostra o modelo a seguir:

0.001 : 05 acertos

0.002 : 04 acertos

n : xx acertos

= Resumo =

0 acertos = x cartões (soma dos cartões que fizeram 0 acertos)

1 acertos = y cartões (soma dos cartões que fizeram 1 acertos)

n acertos = z cartões (soma dos cartões que fizeram n acertos)

6. **Imprimir** - O sistema deve, para um *jogo gerado* qualquer, permitir que o usuário imprima cada *volante* diretamente sobre o papel utilizado nas lotéricas. Para a versão 1.0, é suportado volantes da Caixa Econômica para Quina e Lotofácil, porém o formato do cartão poderá ser alterado via configuração. Serão suportadas impressoras que permitem puxar o papel do tamanho do volante, e no centro da área de impressão.

4.4 Requisitos não funcionais

Para cada requisito alto nível não funcional, será detalhado abaixo, na seqüência. O número corresponde ao requisito alto nível. São eles:

7. **Suporte a diversas loterias** - Sistema deverá, através de configuração, aceitar tipos diferentes de loteria. Para a versão 1.0, para demonstrar essa característica, o sistema deverá suportar dois tipos de loterias diferentes: Quina e Lotofácil. As peculiaridades de cada tipo de jogo deverá estar abstraída através de configuração, não podendo ser usadas regras específicas de alguma delas codificada no núcleo do sistema, o que afetaria negativamente este requisito de ser flexível.
8. **Extensibilidade** - Entendendo por novas funcionalidades: adição de novos(as) *análises, filtros*, tipos de loterias, *geradores* e *verificadores*. O sistema deverá se comportar, para estas novas funcionalidades, como se comporta um sistema baseado em *plugins:* na presença de um novo artefato, deixar este novo artefato disponível para uso, sem necessidade de configuração extra ou modificação de código.
9. **Plataforma** - Sistema deverá rodar, no mínimo, em Windows 98 SE, Windows 2000, Windows XP e Linux (distribuição Debian Sarge).
10. **Internacionalização** - Externalizar o texto traduzível em arquivos para posterior tradução, não necessitando recompilar código para se ter o sistema em outro idioma. O idioma padrão será o inglês americano, sendo que deverá haver uma tradução para português do Brasil. A detecção do idioma se dará através da configuração de idioma do ambiente do usuário.

11. **Distribuição** - Disponibilizar o projeto em um repositório de projetos de código aberto, estando lá disponível no controle de versão do repositório tanto os fontes quanto o trabalho de conclusão de curso. No caso, o repositório será o SourceForge (http://sourceforge.net/).

5 Projeto do software

Com base nos requisitos descritos no capítulo anterior, este capítulo contém o projeto do software que cumpre os requisitos citados. Neste capítulo são mostrados detalhes da arquitetura adotada, diagramas de classes e outros artefatos de projeto de software.

5.1 Arquitetura geral

Foi dividido o problema, em seis grandes módulos:

- Módulo núcleo – Classes e abstrações comuns usadas por todos os módulos;
- Módulo análise – Classes e abstrações relacionadas com requisitos de análise;
- Módulo filtro – Classes e abstrações relacionadas com os requisitos de filtro;
- Módulo gerador – Classes e abstrações relacionadas com os requisitos de geração;
- Módulo conferência - Classes e abstrações relacionadas com os requisitos de conferência;
- Módulo impressão – Classes e abstrações relacionadas comos requisitos de impressão.

A representação gráfica pode ser encontrada na figura abaixo:

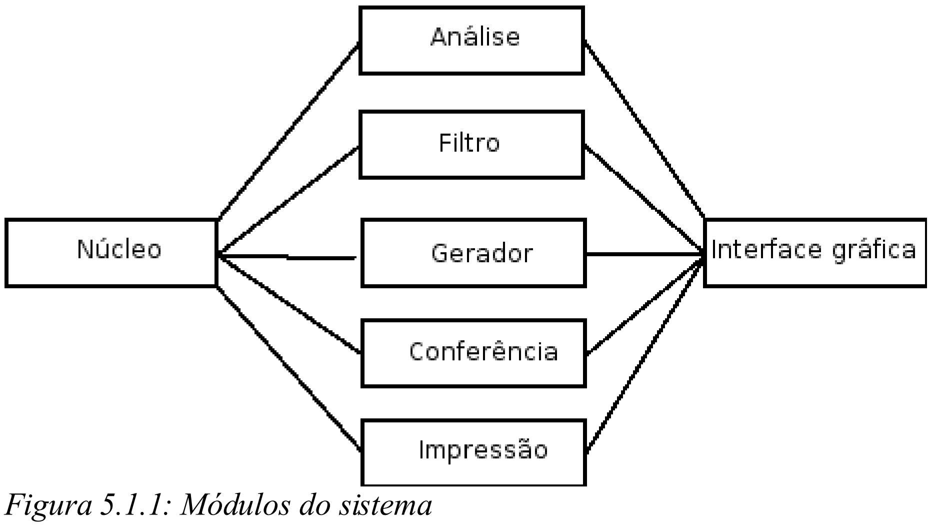


Figura 5.1.1: Módulos do sistema

Este foi o primeiro esforço de divisão do problema, o de tentar quebrar o projeto de software em unidades menores de software, para facilitar a codificação, validação dos requisitos e testes. Esta divisão também facilita a criação de abstrações pertinentes a cada requisito, o que facilita o cumprimento do requisito de extensibilidade. Será detalhado, um para cada módulo, um tópico explicando as suas funcionalidades, e como a necessidade de extensibilidade foi cumprida.

5.2 Diagrama de casos de uso

Para facilitar a visualização das funcionalidades do sistema, segue abaixo o diagrama de casos de uso:

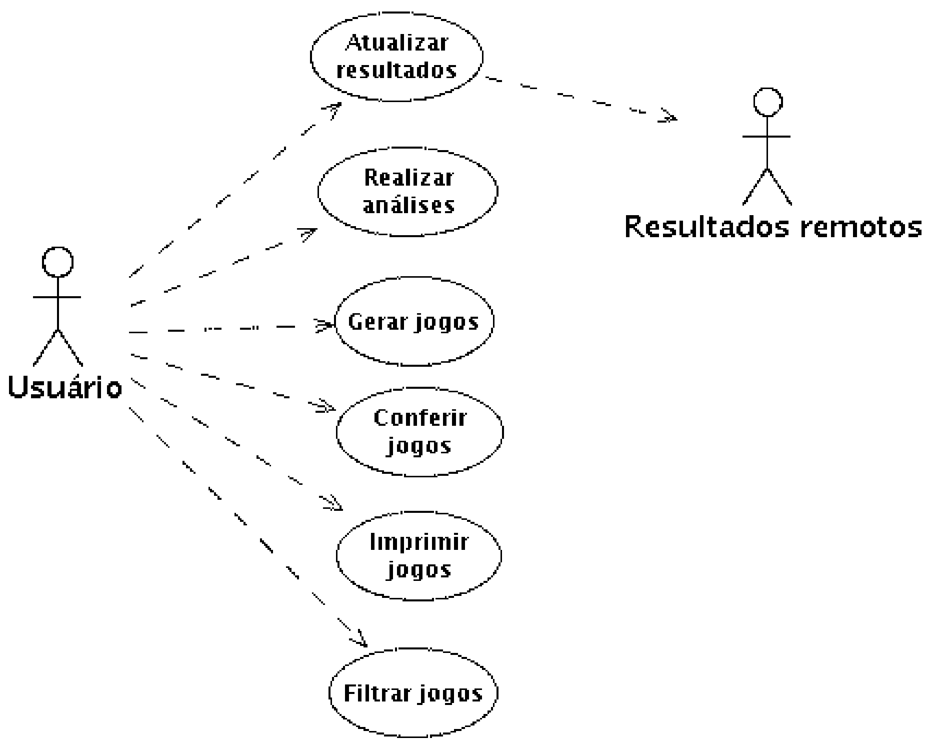


Figura 5.2.1: Diagrama de casos de uso

5.3 Soluções utilizadas em todos os módulos

5.3.1 Princípio aberto-fechado (Open-closed principle)

O princípio aberto-fechado diz que “classes devem ser abertas para extensão, porém fechadas para modificação” [FREEMAN01].

Este princípio vem de referências sobre padrões de projeto de software, e é utilizado em vários módulos para suportar a extensibilidade de funcionalidades. Por exemplo, para adicionar uma nova análise disponível, o programador não precisa **alterar** uma única linha de código sequer do sistema inteiro, basta **criar** uma nova subclasse de uma abstração de análise já existente, ou seja, **extender** de algo que já existe, já foi bem testado e funciona. Mais informações podem ser obtidas na referência da citação acima.

5.3.2 Arquitetura plugin

O sistema usa, em vários módulos, a noção de uma arquitetura plugin, no sentido que código novo introduzido não requer mudança em código existente para que o código existente perceba e possa fazer uso das novas funcionalidades adicionadas. Alguns exemplos clássicos, para facilitar o entendimento:

- Navegadores de internet suportam plugins para adicionar novos tratadores de tipo de mídia: por exemplo, o usuário instala o **plugin** do Macromedia Flash para suportar este tipo de mídia no navegador, e o código do navegador não é alterado ou recompilado;
- Editores de imagem, como o Photoshop, Gimp, Corel suportam plugins para aplicar efeitos nas imagens em edição;
- Ambientes de desenvolvimento, como o Eclipse e o Netbeans, permitem que novas funcionalidades sejam agregadas ao ambiente. O Eclipse levou o conceito tão longe que o próprio software é apenas um grande gerenciador de plugins, não fornecendo ele próprio funcionalidade de uso.

No caso deste software de loterias, foi utilizado este conceito para que outros programadores possam adicionar novas análises, filtros, tipos de jogos, verificadores e geradores sem precisar tocar em uma única linha de código sequer de todo o sistema. Isso facilita muito a adição destas novas funcionalidades, pois quem assim o quiser fazer, apenas precisa escrever seu novo código de interesse, sem se preocupar em como tornar isso disponível para a estrutura existente. A estrutura existente é capaz de detectar que algo novo está presente e tratar de encaixá-la no contexto. Deste modo, o requisito de extensibilidade facilitada fica garantido.

Ficará evidenciado, em cada seção a seguir, as abstrações que suportam essa arquitetura plugin. No capítulo de implementação (capítulo 6), será mostrado como foi implementada a capacidade do sistema reconhecer código novo sem necessidade de alteração de código existente ou recompilação.

5.4 Módulo núcleo

Este módulo contém classes e abstrações comuns úteis para os outros módulos. O diagrama de classes, com as classes mais importantes:

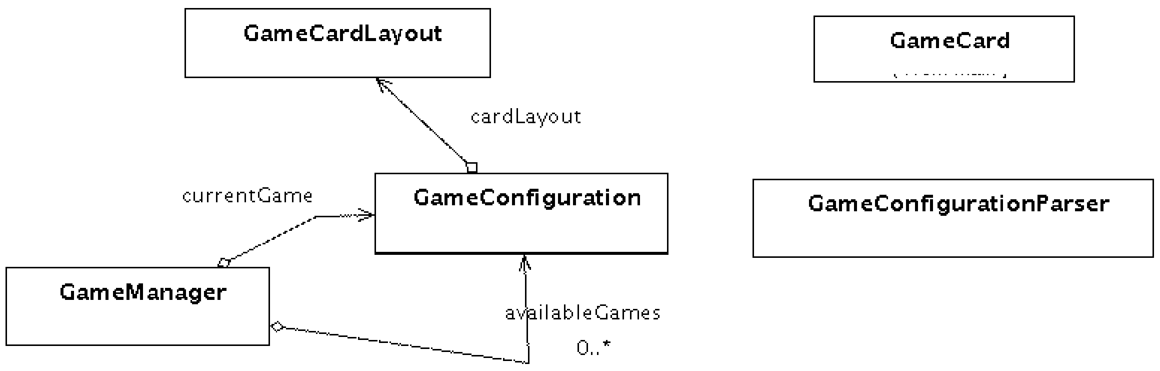


Figura 5.4.1: Diagrama de classes do módulo núcleo

Vamos a descrição de cada uma delas:

- GameCard - Abstração de um jogo ou resultado de loteria. É uma classe simples, porém essencial: ela guarda os números de um jogo ou resultado de loteria.
- GameConfigurationParser - Como definido nos requisitos, as peculiaridades de cada jogo de loteria serão definidas por configuração. A função desta classe é ter o papel de leitor destas configurações. O resultado da leitura de configuração dá vida a uma classe muito importante no sistema: GameConfiguration.
- GameConfiguration – Classe que contém as definições específicas de cada jogo. Descreve, por exemplo, quantas dezenas tem um tipo de jogo, quantas linhas, quantidade de dezenas que pode escolher, quantidade de dezenas sorteadas, entre outras informações definidas em configuração. Faz uso da classe GameCardLayout.
- GameCardLayout – Contém as informações sobre o layout físico de um cartão de loteria. É útil principalmente para o módulo de impressão.
- GameManager – É através do GameManager que os outros módulos tem acesso a configuração de jogo atual (GameConfiguration). Através da dupla GameManager + GameConfiguration, os módulos não precisam saber se estão lidando com um jogo de loteria da Quina, da Lotofácil ou de qualquer outra loteria que possa ser configurada no sistema. Isso garante que o sistema como um todo fica independente de tipo de jogo de loteria. A figura a seguir deve deixar mais claro a responsabilidade principal deste módulo:

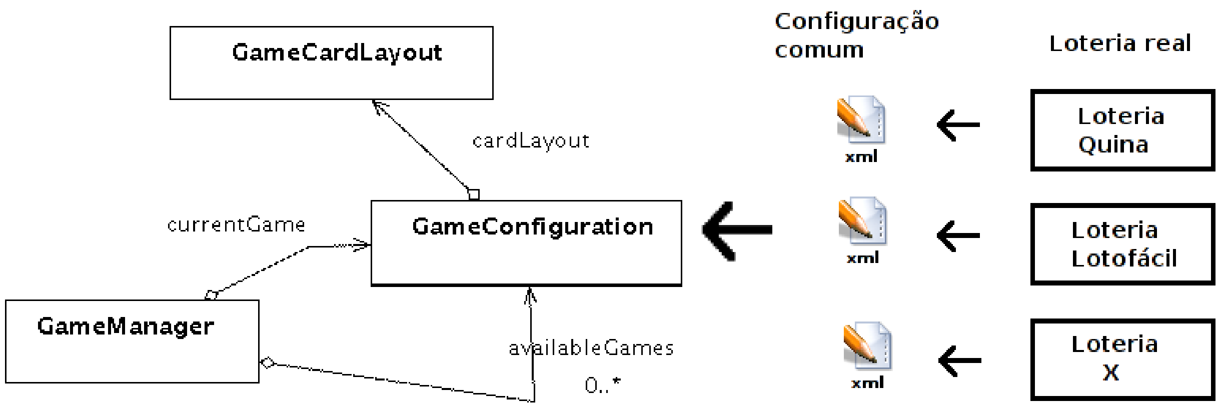


Figura 5.4.2: Detalhando a utilidade de GameManager e GameConfiguration

GameManager, na inicialização, obtém todos os jogos configurados. Após este passo, GameManager terá uma coleção de GameConfiguration disponíveis. Há a escolha de um GameConfiguration para uso corrente no sistema, e este é disponibilizado aos outros módulos. Mais detalhes sobre estes passos, consultar o capítulo de implementação(capítulo 6).

5.5 Módulo análise

O módulo análise define, para o sistema, uma abstração mínima do que seria uma análise de loteria, e vai especializando esta abstração para promover facilidades ao desenvolvedor. Quanto mais próximo da folha, da hierarquia de especialização, mais fácil é a implementação desta análise, pois mais código já está pronto e disponível para o implementador, porém mais restrito este fica quanto a natureza de sua análise. O contrário também é verdadeiro: quanto mais próximo da raiz a classe estiver, mais genérica ela é, fornece mais liberdade, ao custo de que menos funcionalidades prontas já estarão disponíveis, exigindo mais esforço do implementador. Vamos a figura para tornar mais claro a idéia:

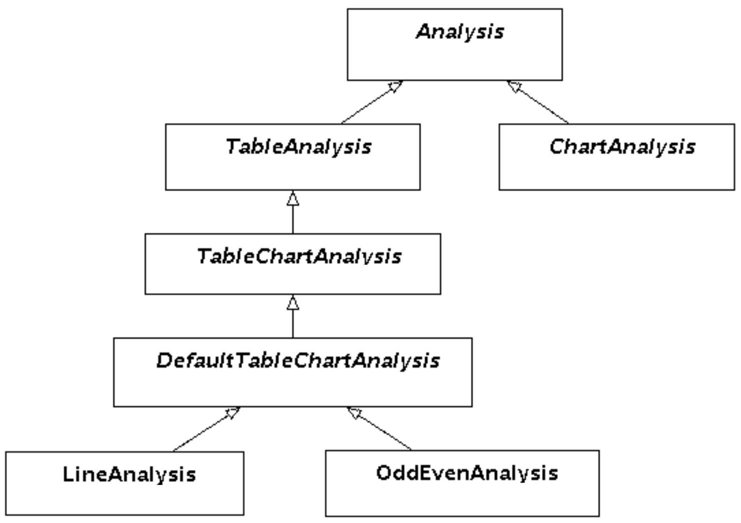


Figura 5.5.1: Hierarquia de análise

A classe análise, no topo da hierarquia, especifica a classe mínima que o desenvolvedor precisa extender para poder inserir no sistema uma nova análise. Porém, através de pesquisa na referência [GAIL01] e de observação de outros softwares existentes, pode-se concluir que uma enorme quantidade de análises se encaixam no formato de análise em tabela e análise em gráfico, ou uma combinação dos dois. Deste modo, criou-se subclasses de análise, específicas para este tipo de análise tão predominante, dando muito “chão já percorrido” para quem quiser adicionar uma nova análise deste tipo, sem contar a melhor organização do código e menos repetição. Todas as cinco análises disponíveis na versão 1.0 herdam de DefaultTableChartAnalysis, classe que deixa disponível facilidade para criar análise que contenha uma tabela e um gráfico (mais detalhes nos capítulos 6 e 7).

Uma classe Manager, assim como no módulo núcleo, é responsável pela carga e disponibilização das análises encontradas.

5.6 Módulo filtro

Muito similar à estrutura do módulo análise, o módulo filtro provê uma abstração básica do que é um filtro, e especializa para facilitar a quem for implementar um novo filtro com algum código comum. No topo da hierarquia do módulo temos a Interface Filterable, que basicamente define um método para aceitar ou rejeitar um GameCard. Desta forma, vários filtros podem ser implementados, onde cada regra específica do filtro é uma classe que implementa esta interface, sobrescrevendo este método que aceita ou rejeita um cartão, especificando a regra específica no corpo do método. Uma classe Manager é responsável pela carga e disponibilização dos filtros para o sistema. O diagrama de classes, na figura abaixo:

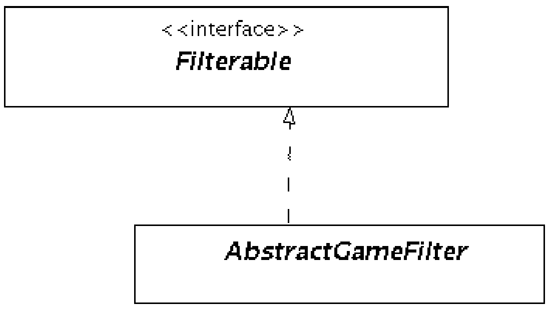


Figura 5.6.1: Hierarquia do módulo filtro

5.7 Módulo gerador

Define o que é necessário para obter como saída um jogo gerado. Também é encontrado neste módulo a abstração do que é o jogo gerado, bem como facilidades para ler e escrever um arquivo de jogo gerado.

Pode parecer mistura de implementação e projeto quando se fala de um **arquivo** de jogo gerado, uma vez que está se especificando a persistência em arquivo, mas de todo modo, se faz necessário levar isso em conta no projeto, uma vez que um jogo gerado pode conter milhões de volantes de loteria, logo se faz necessário definir adequadamente este arquivo de antemão para tentar otimizar o uso de espaço e projetar abstrações para tornar agradável a leitura e escrita deste arquivo. As classes principais estão na figura abaixo:

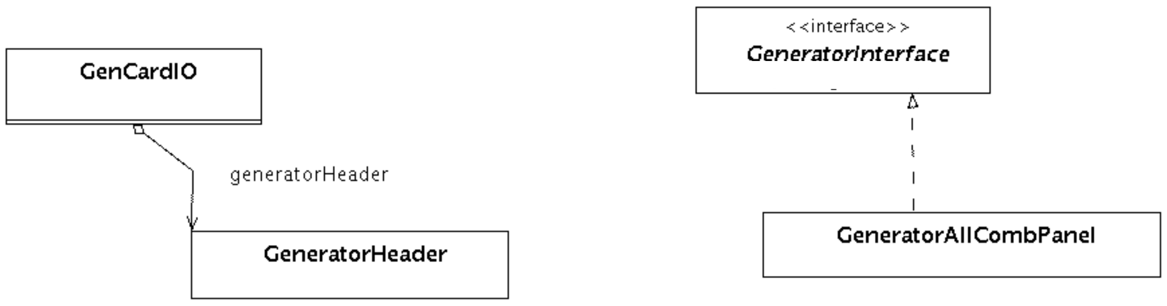


Figura 5.7.1: Classes do módulo gerador

Novamente, há uma classe ou interface (GeneratorInterface, no caso) que especifica, o que é para o sistema, um gerador. Há uma classe concreta, GeneratorAllCombPanel, que implementa a dita interface e fornece ao sistema um tipo de gerador que pode ser usado pelo usuário. Há a presença de duas classes facilitadoras (no jargão de desenvolvimento, *utils)* que facilitam a leitura e escrita de um jogo de loteria gerado: GenCardIO e GeneratorHeader. GenCardIO fornece facilidades para leitura e escrita, enquanto que GeneratorHeader fornece facilidades para leitura de metadados de um jogo gerado (detalhes no capítulo 6).

5.8 Módulo conferência

O módulo conferência define classes e abstrações que permitem ao sistema oferecer ao usuário um meio de conferir seus jogos gerados. Como entrada, temos o resultado e o jogo gerado. Cabe a uma implementação específica de conferência conferir / demonstrar o resultado da conferência ao usuário. Vamos às classes principais, na figura abaixo:

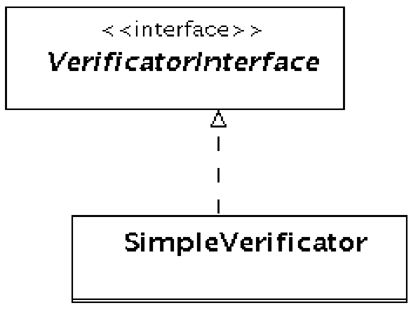


Figura 5.8.1: Módulo conferência

Conferir um jogo gerado levou a uma simples Interface, devido a baixa complexidade de especificar uma abstração para este problema em particular. A classe concreta SimpleVerificator é o verificador padrão do sistema, que segundo os requisitos, deve gerar um relatório simples de averiguação.

5.9 Módulo impressão

Devido o fato de a impressão de um jogo gerado estar intimamente ligado a um tipo de loteria (uma vez que cada tipo de jogo utiliza um layout físico de cartão de loteria diferente), imaginou-se a necessidade de cada tipo de jogo ter uma classe responsável pela impressão deste tipo de jogo. Mas pensando em facilitar a vida do desenvolvedor e evitar repetição de código, criou-se uma hierarquia no estilo do módulo análise, onde há uma especificação alto nível do que é necessário para imprimir e criou-se especializações para cada tipo de jogo de loteria, onde as especializações precisam implementar e escrever menos código novo (aproveitar código comum da superclasse). Vamos ao diagrama para facilitar o entendimento:

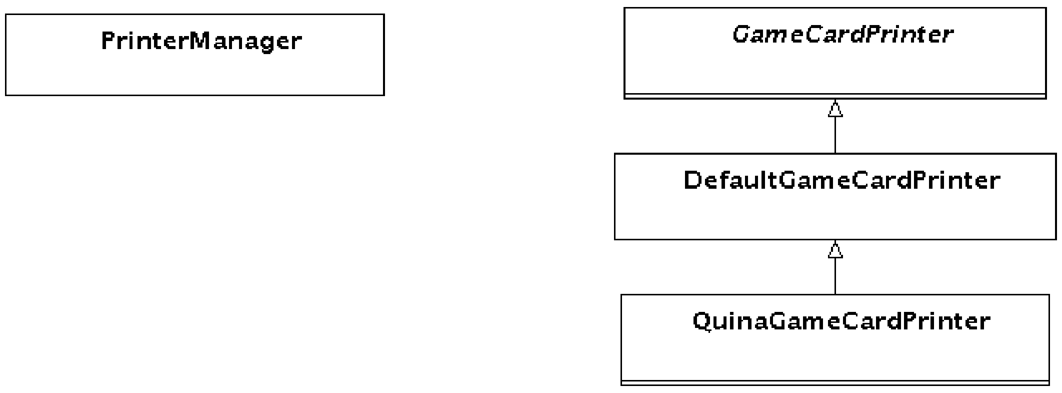


Figura 5.9.1: Módulo impressão

A classe PrinterManager é responsável por iniciar o processo de impressão, utilizando uma classe pertencente a hierarquia de GameCardPrinter. GameCardPrinter é uma classe abstrata que define o que é necessário para que a impressão possa ser realizada, e já disponibiliza alguns métodos comuns concretos. DefaultGameCardPrinter disponibiliza ao desenvolvedor uma implementação que imprime corretamente os números na matriz de números especificada no arquivo de configuração do tipo de jogo. A classe QuinaGameCardPrinter se aproveita do que DefaultGameCardPrinter fornece e adiciona a capacidade de pintar uma célula do cartão de loteria específica da Quina. Note que o tipo de jogo Lotofácil, suportado na versão 1.0 do programa não necessitou de uma implementação específica, a implementação "default" já fornecia o suficiente.

6 Implementação

Neste capítulo é descrita a implementação do software. As tecnologias utilizadas, principais dificuldades e aspectos não documentados no projeto (capítulo 6) que forem pertinentes.

6.1 Tecnologias utilizadas

Como este trabalho é de código aberto, a escolha das tecnologias tem de levar este aspecto em conta, uma vez que se há o desejo da sociedade em poder acessar e editar este código, há a preferência que este o seja feito usando tecnologias livres, ou no mínimo gratuitas, para que, quem for manusear o fonte, possa instalar as mesmas ferramentas que o desenvolvedor principal utilizou. Seria ruim, imaginando que eu fosse um desenvolvedor novo, querendo conhecer e editar o código do projeto, tivesse que comprar uma ferramenta cara para poder abrir a estrutura de projeto de código e editar os fontes.

6.1.1 Linguagem de programação

Escolher a linguagem de programação de um projeto nem sempre é tarefa fácil, mas neste projeto foi uma decisão praticamente direta: a linguagem Java versão 5 (maiores detalhes na referência [SUN01]).

Considerou-se também a linguagem de programação C++ ([STROUSTRUP01]), porém a linguagem Java ganhou pelos seguintes méritos:

- Facilita a distribuição em sistemas operacionais diferentes (existem diversas implementações da máquina virtual java para sistemas operacionais distintos em arquiteturas distintas)
- Possui suporte integrado a reflexão de código, tecnologia que permite inspecionar código dentro do código fonte (importante para implementar a arquitetura plugin)
- Facilita a programação do sistema, por oferecer uma API de programação maior do que a do C++ e por fornecer gerenciamento de memória automático.

6.1.2 Ferramentas de desenvolvimento

Além da necessidade de escolher a linguagem de programação adequada ao problema, se faz necessário escolher cuidadosamente as ferramentas de desenvolvimento, pois o mau casamento entre linguagem e ferramenta pode atravancar o processo de desenvolvimento e perder precioso tempo. Foram consideradas, para este projeto, dois ambientes de desenvolvimento bastante populares: Eclipse [ECLIPSE01] e Netbeans [NETBEANS01].

Ambas as ferramentas possuem funcionalidades muito similares, mas acabou-se decidindo pelo Netbeans por um único importante motivo: o Netbeans fornece um poderoso editor de janelas, o que facilitou em muito o desenvolvimento deste trabalho.

6.1.3 Hospedagem do projeto

Um dos objetivos deste trabalho consistia na disponibilização em massa do código fonte do programa, para que pessoas de todo o mundo pudessem baixar e editar o fonte livremente. Um dos sites que hospedam projetos de código aberto mais famoso é o SourceForge [SOURCEFORGE01]. Devido a fama mundial e os bons serviços que presta, foi escolhido para acomodar este projeto. Tem ainda a vantagem de não cobrar taxa alguma, e fornece espaço para página web, gerenciamento de versão e outras facilidades. Se você, leitor, tiver interesse em baixar este projeto e começar a editar o software, acesse http://sourceforge.net/projects/lagar-lottery .

6.2 Principais dificuldades encontradas

6.2.1 Identificação de subclasses em tempo de execução

Há um requisito que diz que o sistema deve descobrir novas implementações de análises, filtros, etc, sem precisar haver recompilação ou o usuário tiver que explicitamente dizer, através de configuração, que há algo novo. Para poder cumprir este requisito, com a arquitetura de projeto definida, havia a necessidade de descobrir, em tempo de execução, onde estão todas as subclasses de uma classe em particular. Por exemplo, é desejável que o sistema possa descobrir todas as subclasses da superclasse "Análise", pois de posse dessa informação, ele pode instanciar todas as análises disponíveis no sistema e deixar disponível para o usuário final. Como a linguagem escolhida foi Java, que já possui suporte nativo a reflexão de código, bastou pesquisar um pouco na internet para descobrir que alguém já teve um problema parecido e disponibilizou uma implementação de código que faz exatamente o que se desejava: dado o nome de uma superclasse, é retornado uma lista de subclasses que extende aquela superclasse. Mais detalhes sobre o código e a técnica podem ser encontrados na referência [JAVAWORLD01].

É por este motivo que, se você leitor for examinar o código fonte, perceberá que, para cada módulo, há uma classe gerenciadora (Manager) que usa a técnica para descobrir as subclasses que implementam ou extendem da classe de abstração principal do módulo, para disponibilizar a coleção de subclasses para código cliente que faz uso delas.

6.2.2 Interface gráfica

Como o projeto do sistema prevê grande flexibilidade em adição de novas funcionalidades (através da já discutida arquitetura plugin, identificação de subclasses em tempo de execução), a interface gráfica também precisaria ter uma flexibilidade, no sentido que o que a interface mostra precisa refletir as funcionalidades que foram descobertas pelos gerenciadores (Managers). Por exemplo: se o Manager de Análise descobriu 5 análises disponíveis no sistema para execução, a interface precisa mostrar uma lista com as cinco análises para o usuário poder escolher qual quer executar. Você encontrará essa filosofia de "mostrar o que está disponível" e deixar o usuário escolher para "executar" em praticamente toda a interface do usuário. Isso ficará mais claro no capítulo 7, que mostra o software na versão 1.0.

É também por este motivo que as classes que determinam a abstração do módulo tem métodos para "obter nome" ou "obter descricao", pois a interface gráfica não conhece de antemão as funcionalidades que está fornecendo. A interface obtém, em tempo de execução, as informações para mostrar ao usuário. Isso deixou o sistema extremamente flexível e fácil de adicionar novas habilidades.

7 LagarLottery 1.0

LagarLottery foi o nome dado ao software, por este motivo este capítulo tem este nome. Por se tratar de um código que está disponível em um repositório americano (SourceForge) e estar disponível para o mundo, escolheu-se um nome em inglês para o produto (e é por este mesmo motivo que o código fonte está em inglês).

Este capítulo visa mostrar algumas telas do programa em sua versão 1.0, demonstrando as funcionalidades disponíveis. É um capítulo mais visual (imagens) do que textual.

7.1 Tela principal

Primeira tela do sistema. É questionado ao usuário que tipo de loteria este gostaria de trabalhar. Note que a lista de loterias disponíveis é populada dinamicamente de acordo com os arquivos de configuração de loteria disponíveis. Perceba também a barra lateral com botões: para cada módulo do sistema, criou-se um botão que aciona a tela que gerencia o módulo em questão. No menu superior esquerdo, há a opção de atualizar a base de resultados ou listar os resultados da base atual.

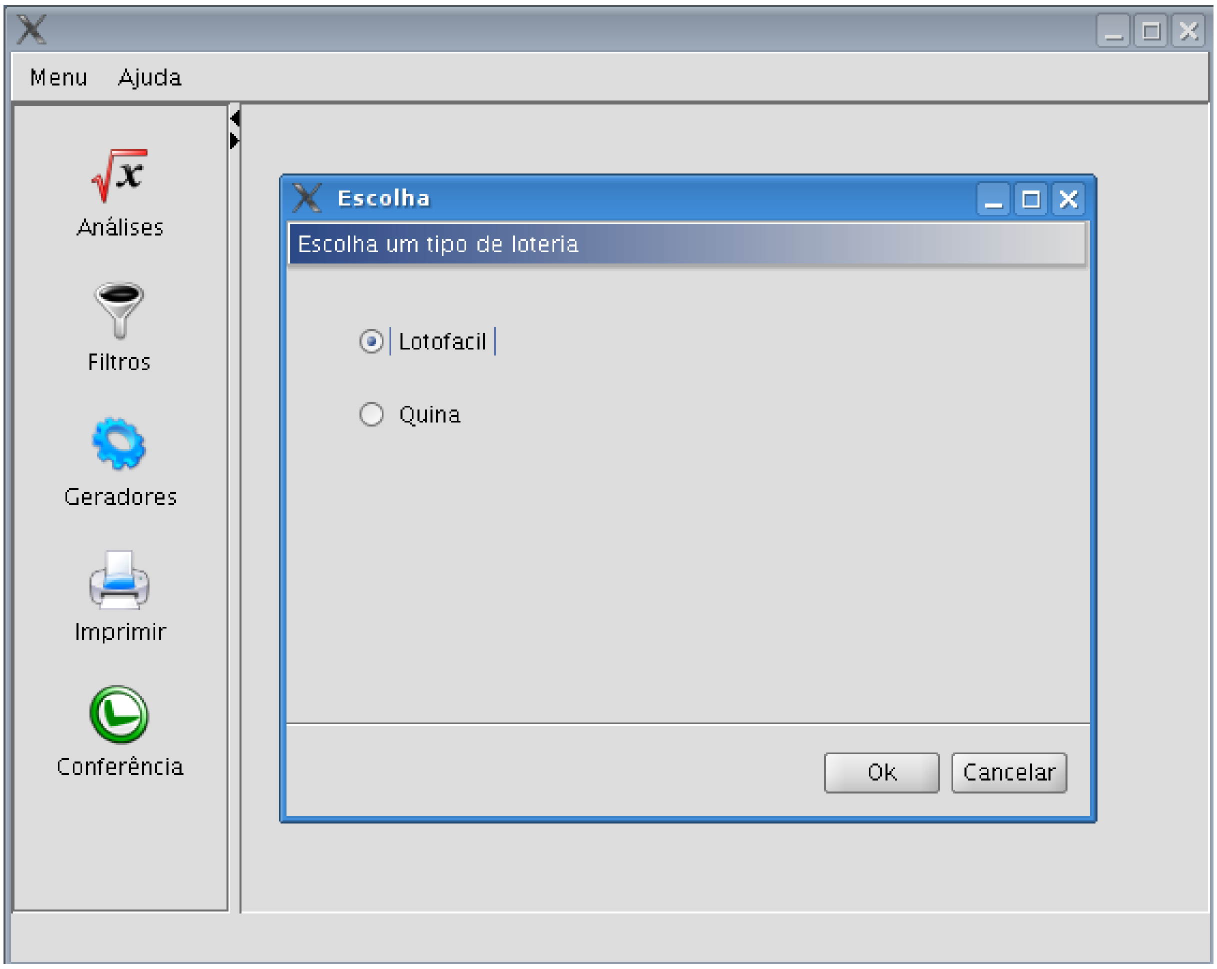
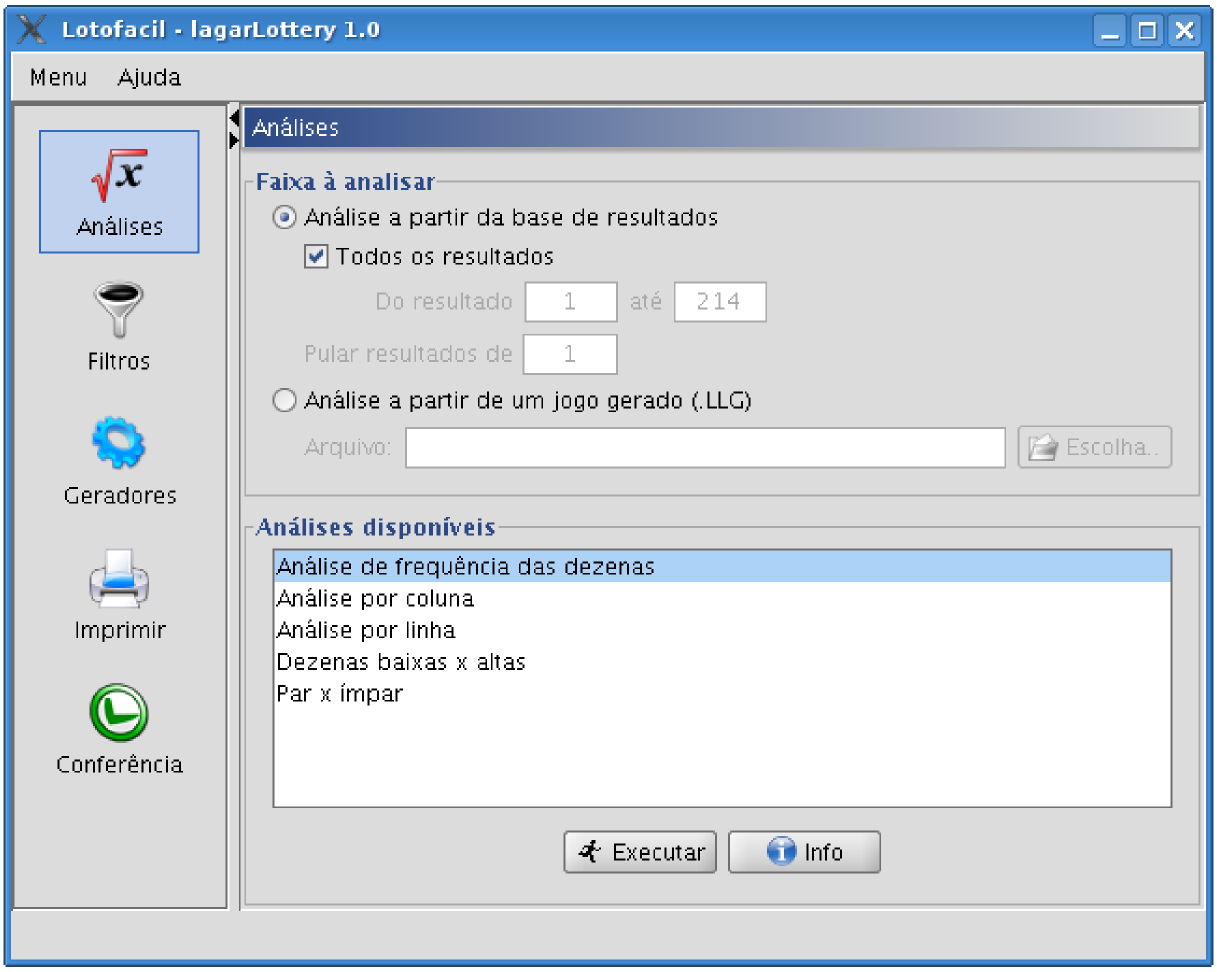


Figura 7.1.1: Tela inicial

7.2 Tela módulo análise

Nesta tela, perceba a lista de análises disponíveis. Ela é populada dinamicamente, em tempo de execução, com as análises que o sistema encontra (subclasses da classe "Análise"). O usuário pode realizar análises sobre a base de resultados ou sobre um jogo gerado. Para classes do módulo "Análise", não há distinção da fonte de "números" graças ao padrão de projeto Iterator (sobre o padrão, ver referência [FREEMAN01]).



Análises

Faixa à analisar

(● Análise a partir da base de resultados)

[✓] Todos os resultados

Do resultado 1 até 214

Pular resultados de 1

○ Análise a partir de um jogo gerado (.LLG)

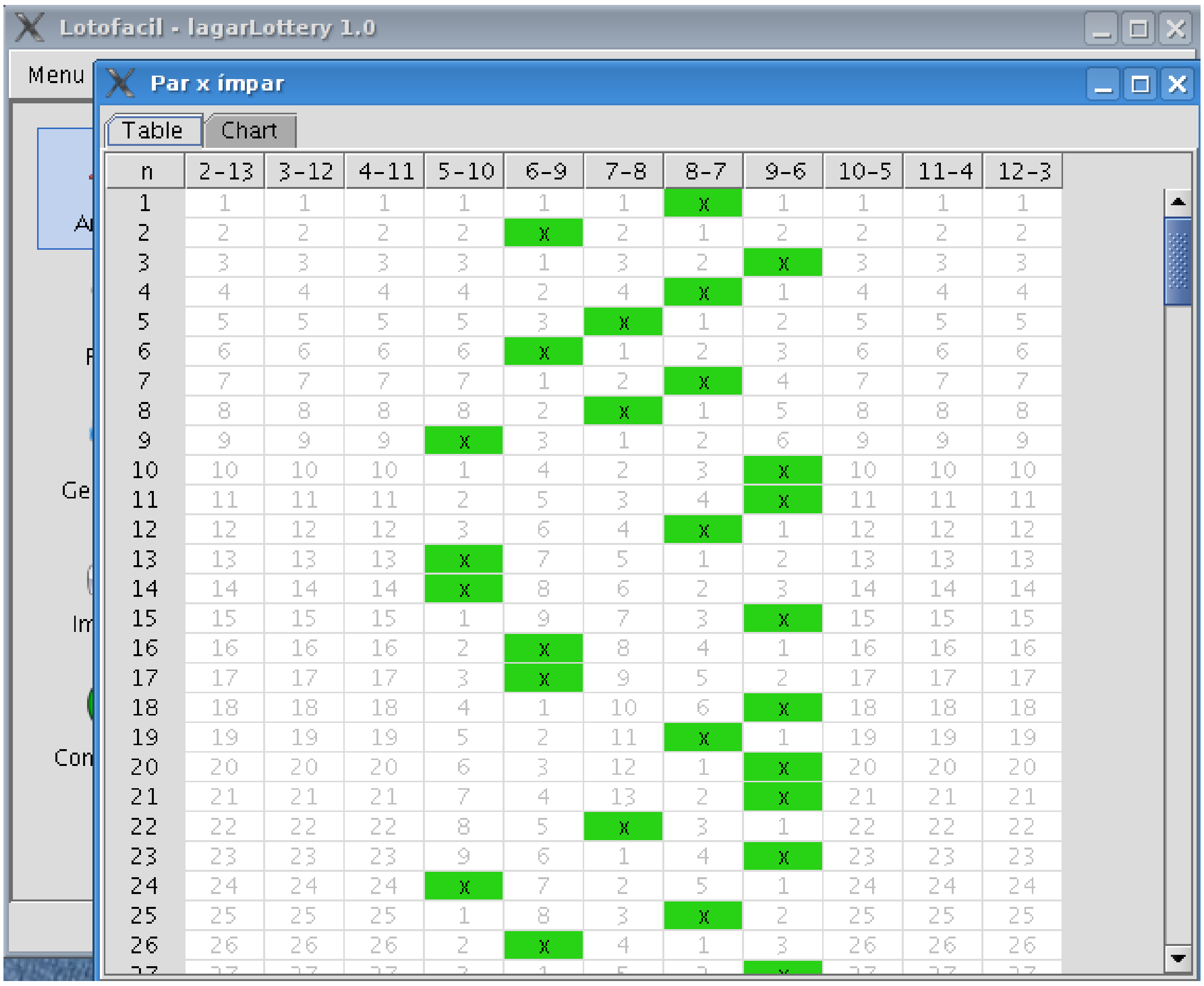
Arquivo:

Análises disponíveis

- Análise de frequência das dezenas
- Análise por coluna
- Análise por linha
- Dezenas baixas x altas
- Par x ímpar

Figura 7.2.1: Tela principal do módulo análise

Na figura abaixo, temos a execução da análise "Par x ímpar". É aberto a análise em uma nova janela, para que o usuário possa executar várias simultaneamente. A figura 7.2.2 mostra a visão da tabela, onde para cada resultado (primeira coluna) temos a categoria que este resultado se encaixa (marcado com um X verde). As demais colunas mostram as categorias possíveis, sendo o primeiro número antes do hífen o número de pares, logo o outro número após o hífen é o número de ímpares). A figura 7.2.3 resume a tabela em um gráfico de pizza.



n	2-13	3-12	4-11	5-10	6-9	7-8	8-7	9-6	10-5	11-4	12-3
1	1	1	1	1	1	1	x	1	1	1	1
2	2	2	2	2	x	2	1	2	2	2	2
3	3	3	3	3	1	3	2	x	3	3	3
4	4	4	4	4	2	4	x	1	4	4	4
5	5	5	5	5	3	x	1	2	5	5	5
6	6	6	6	6	x	1	2	3	6	6	6
7	7	7	7	7	1	2	x	4	7	7	7
8	8	8	8	8	2	x	1	5	8	8	8
9	9	9	9	x	3	1	2	6	9	9	9
10	10	10	10	1	4	2	3	x	10	10	10
11	11	11	11	2	5	3	4	x	11	11	11
12	12	12	12	3	6	4	x	1	12	12	12
13	13	13	13	x	7	5	1	2	13	13	13
14	14	14	14	x	8	6	2	3	14	14	14
15	15	15	15	1	9	7	3	x	15	15	15
16	16	16	16	2	x	8	4	1	16	16	16
17	17	17	17	3	x	9	5	2	17	17	17
18	18	18	18	4	1	10	6	x	18	18	18
19	19	19	19	5	2	11	x	1	19	19	19
20	20	20	20	6	3	12	1	x	20	20	20
21	21	21	21	7	4	13	2	x	21	21	21
22	22	22	22	8	5	x	3	1	22	22	22
23	23	23	23	9	6	1	4	x	23	23	23
24	24	24	24	x	7	2	5	1	24	24	24
25	25	25	25	1	8	3	x	2	25	25	25
26	26	26	26	2	x	4	1	3	26	26	26
27	27	27	27	3	1	5	2	x	27	27	27

Figura 7.2.2: Análise par x ímpar - visão da tabela

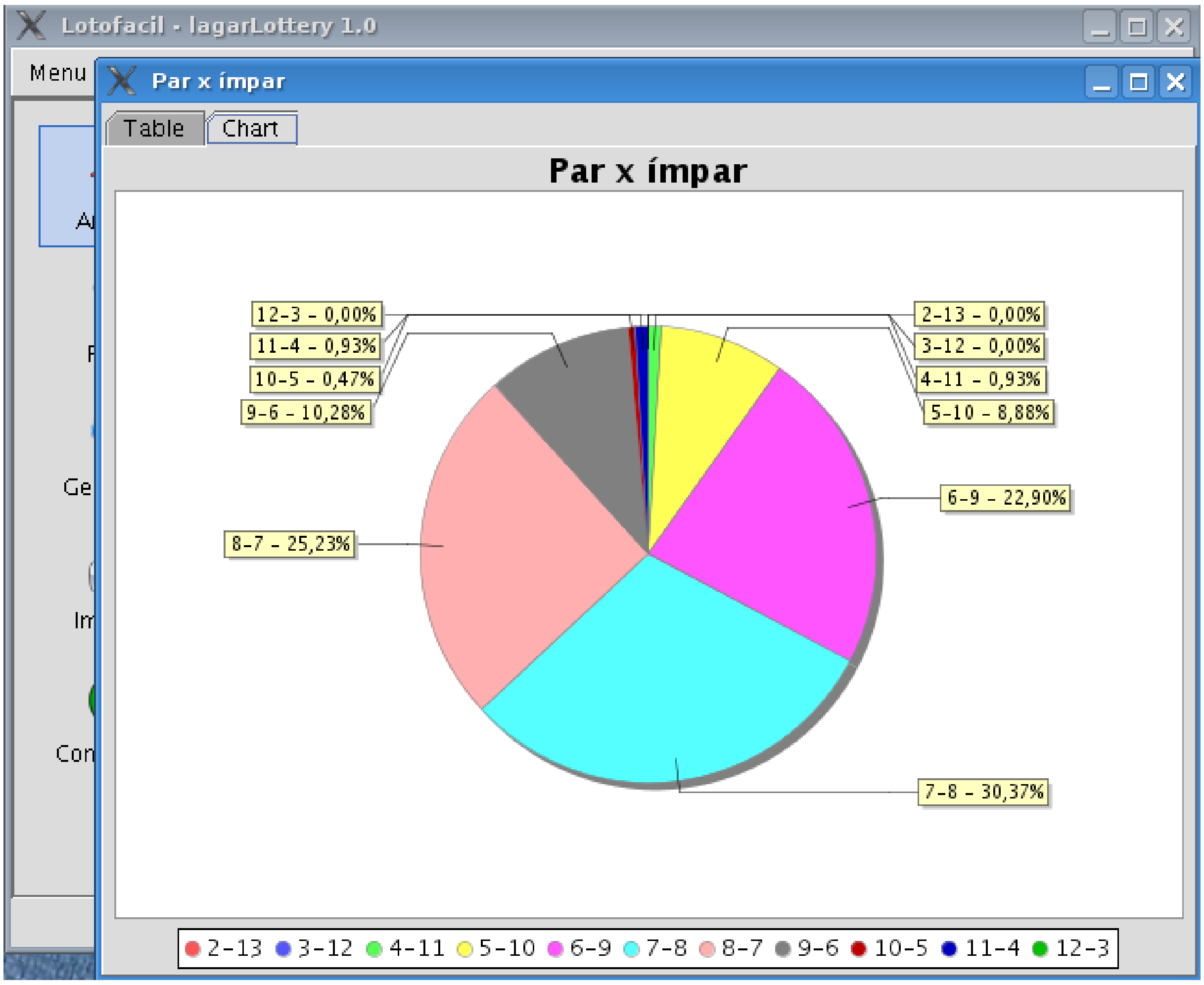
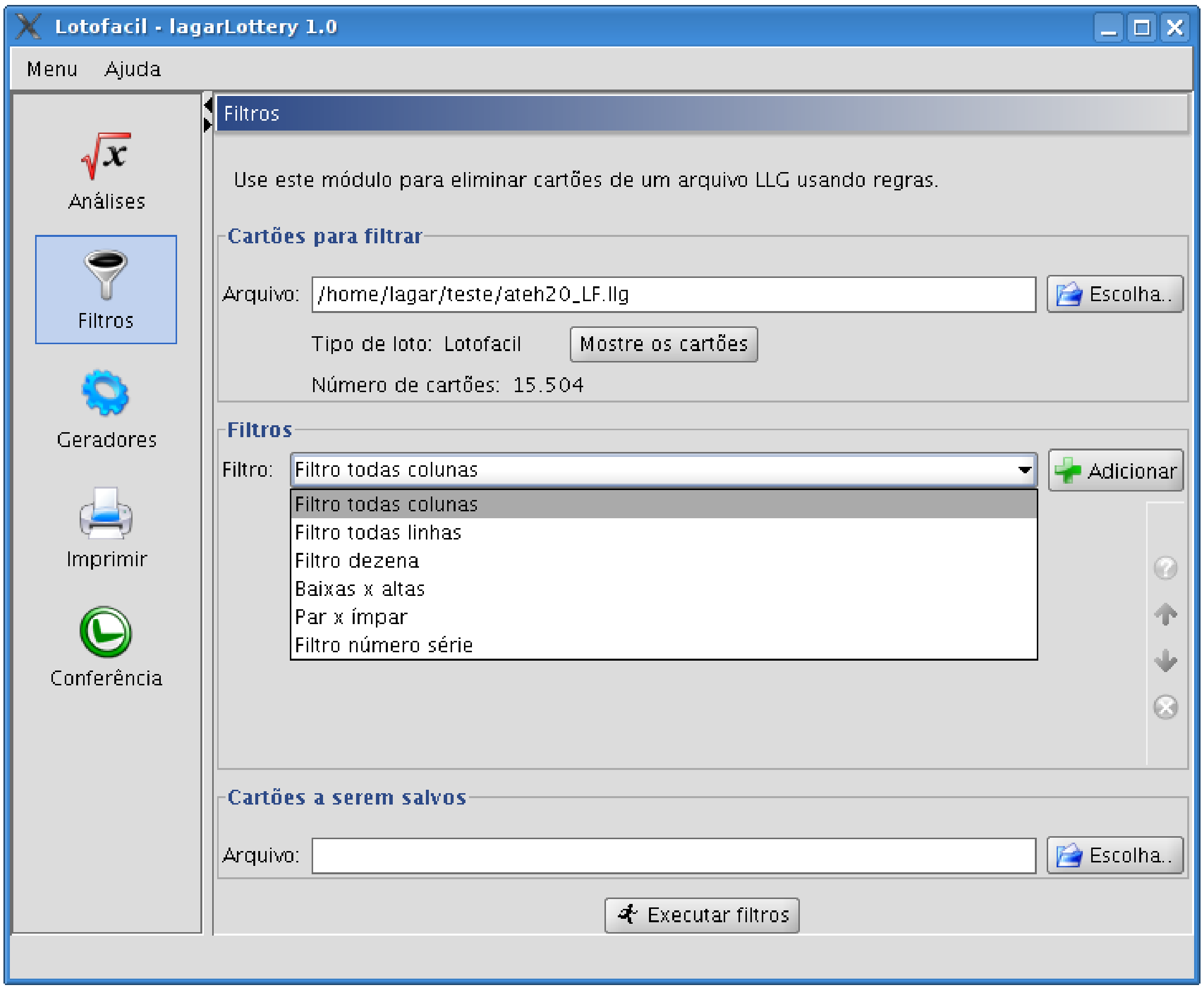


Figura 7.2.3: Análise par x ímpar - visão do gráfico

7.3 Tela módulo filtro

A tela módulo filtro também possui o recurso de obter filtros em tempo de execução, e apresentar uma lista para o usuário adicionar os filtros. Vários filtros podem ser encadeados um sobre o outro. É possível mover a posição dos filtros ou remover da lista. Veja nas figuras abaixo:



Use este módulo para eliminar cartões de um arquivo LLG usando regras.

Cartões para filtrar

Arquivo: /home/lagar/teste/ateh20_LF.llg

Tipo de loto: Lotofacil

Número de cartões: 15.504

Filtros

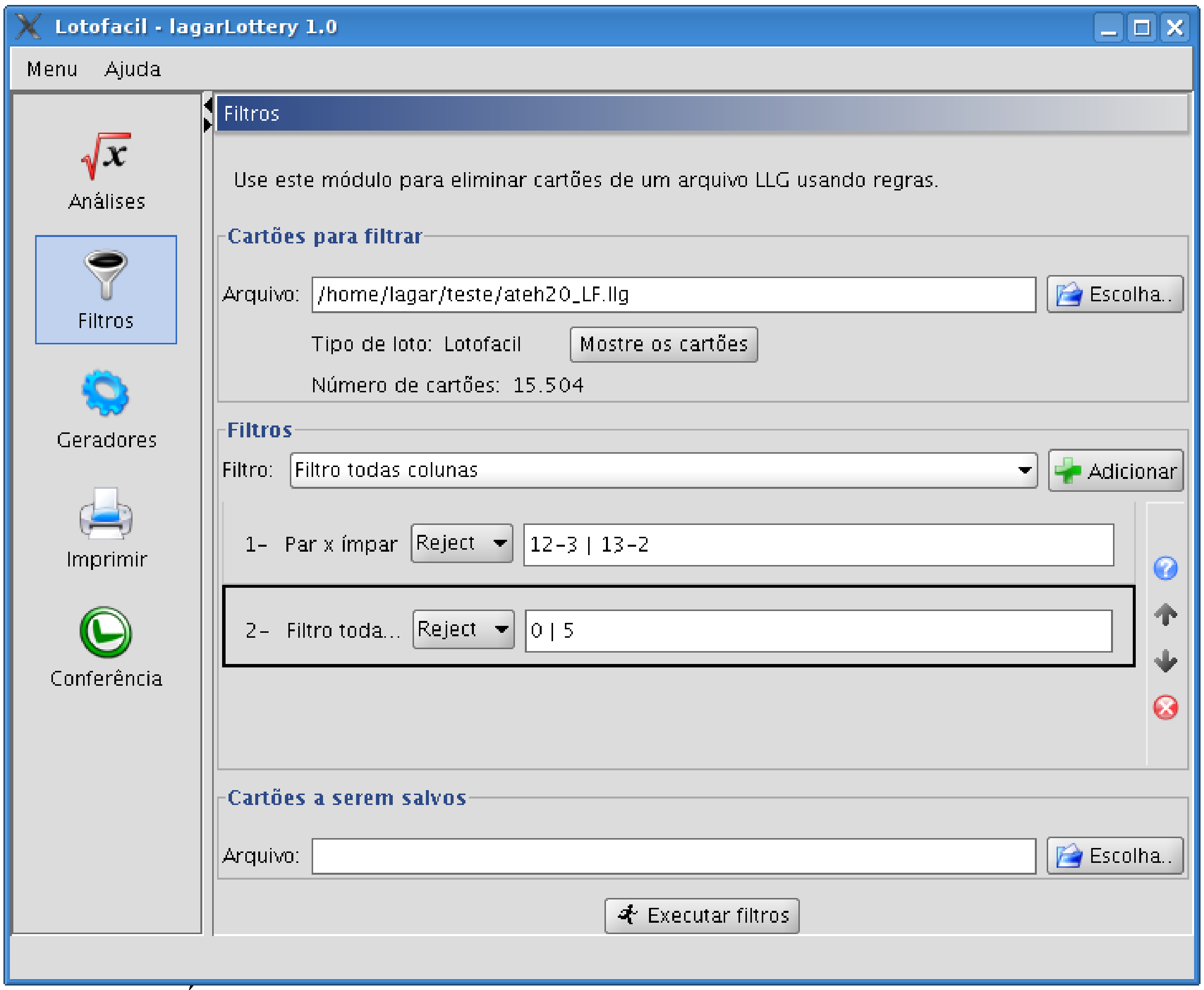
Filtro: Filtro todas colunas

Filtro todas colunas
Filtro todas linhas
Filtro dezena
Baixas x altas
Par x ímpar
Filtro número série

Cartões a serem salvos

Arquivo:

Figura 7.3.1: Tela módulo filtro



Use este módulo para eliminar cartões de um arquivo LLG usando regras.

Cartões para filtrar

Arquivo: /home/lagar/teste/ateh20_LF.llg [Escolha..]

Tipo de loto: Lotofacil [Mostre os cartões]

Número de cartões: 15.504

Filtros

Filtro: Filtro todas colunas [Adicionar]

1- Par x ímpar Reject 12-3 | 13-2

2- Filtro toda... Reject 0 | 5

Cartões a serem salvos

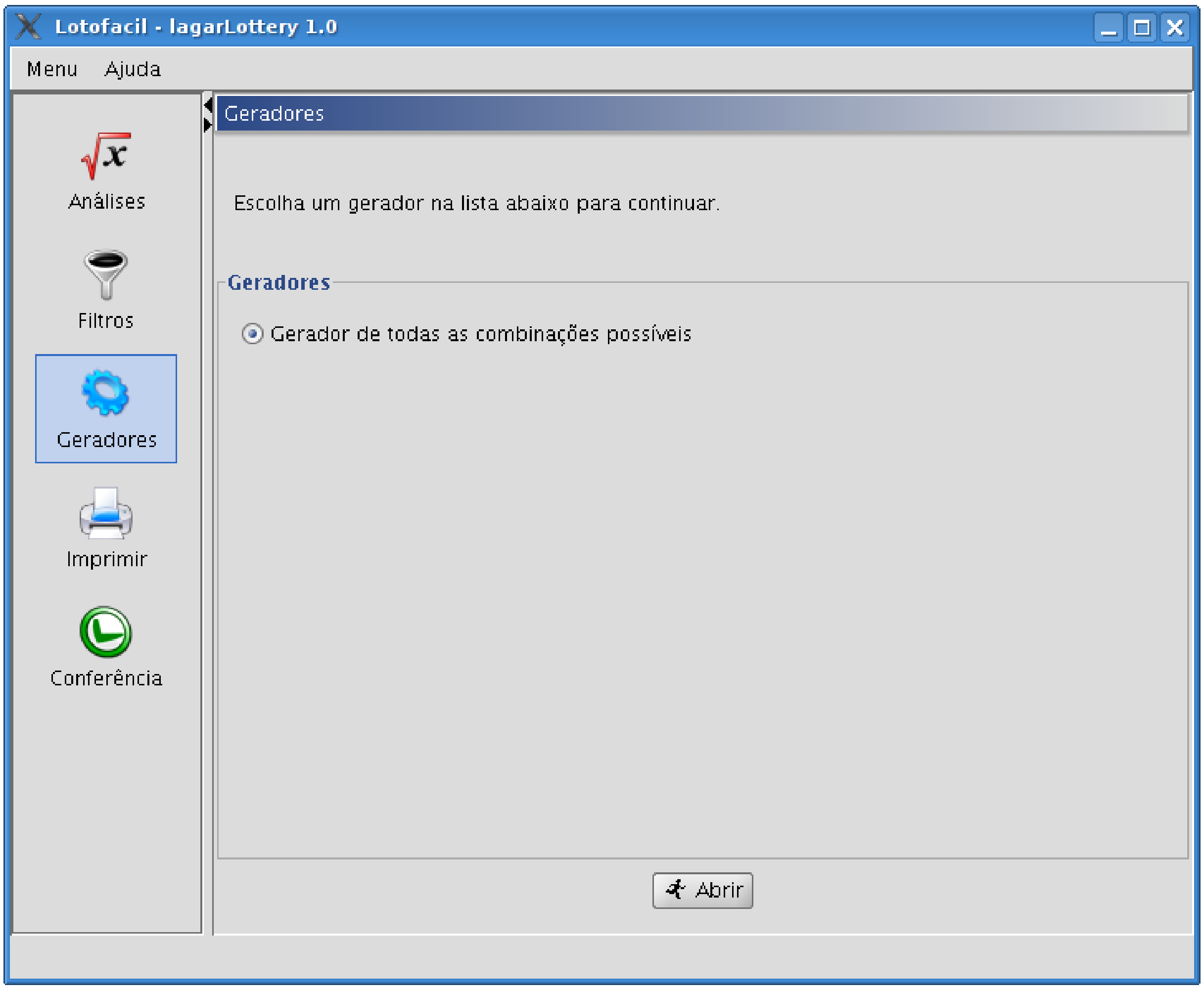
Arquivo: [Escolha..]

[Executar filtros]

Figura 7.3.2: É permitido empilhar quantos filtros quiser

7.4 Tela módulo gerador

Ao ir para a tela principal do módulo gerador, primeiro é perguntado ao usuário que gerador ele quer manusear. A lista de geradores é montada dinamicamente. Após este tomar a decisão por qual gerador, é apresentada a tela fornecida pela implementação de gerador. Qualquer desenvolvedor pode anexar uma tela de gerador, que este aparecerá na lista para escolha. Na figura abaixo, temos o gerador fornecido com o sistema, o gerador de “todas as combinações possíveis”.

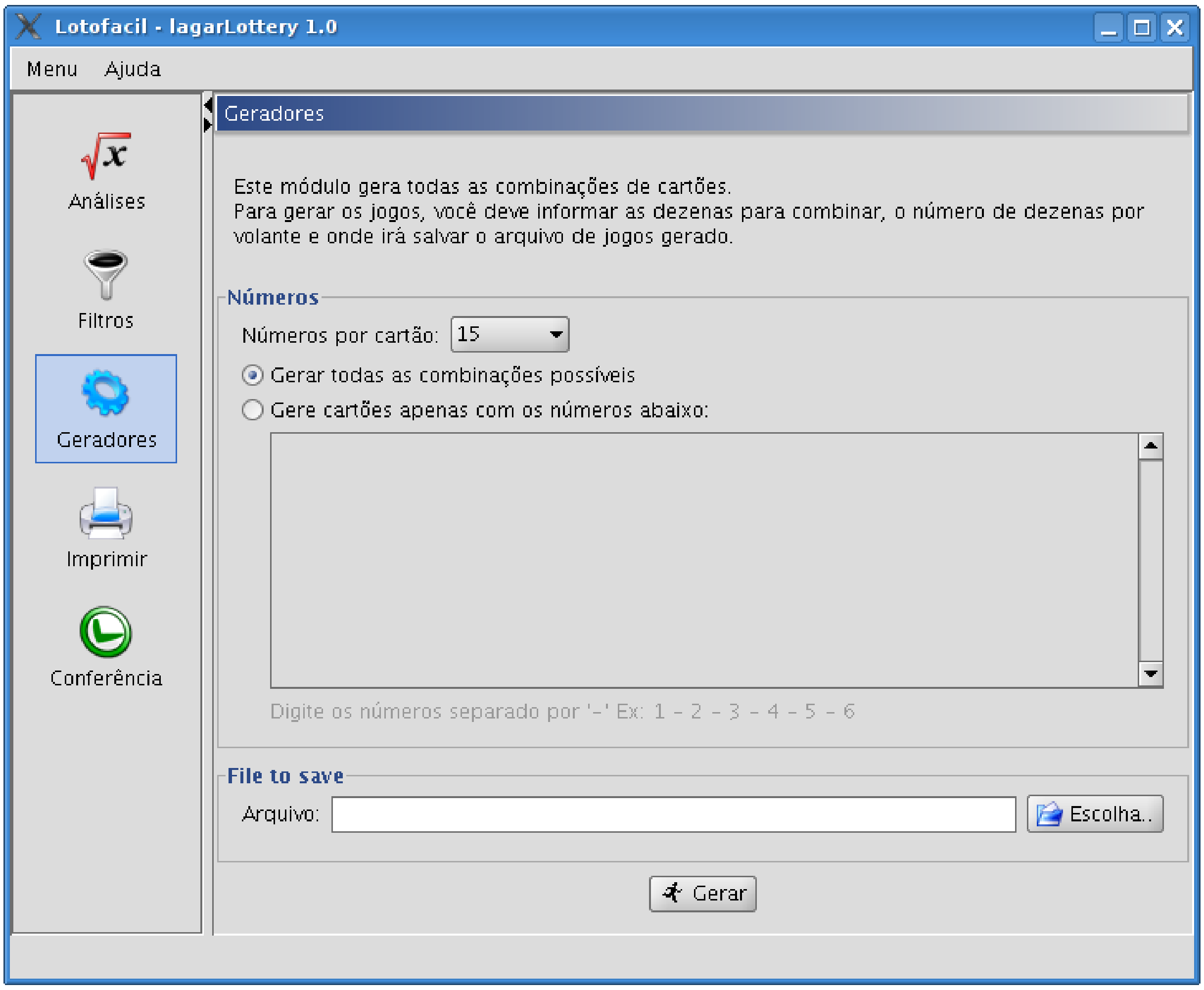


Escolha um gerador na lista abaixo para continuar.

Geradores

[✓ Gerador de todas as combinações possíveis]

Figura 7.4.1: Tela inicial do módulo gerador - lista de geradores disponíveis


Lotofacil - lagarLottery 1.0

Geradores

Este módulo gera todas as combinações de cartões.
Para gerar os jogos, você deve informar as dezenas para combinar, o número de dezenas por volante e onde irá salvar o arquivo de jogos gerado.

Números

Números por cartão: 15

[✓ Gerar todas as combinações possíveis]

Gere cartões apenas com os números abaixo:

Digite os números separado por '-' Ex: 1 - 2 - 3 - 4 - 5 - 6

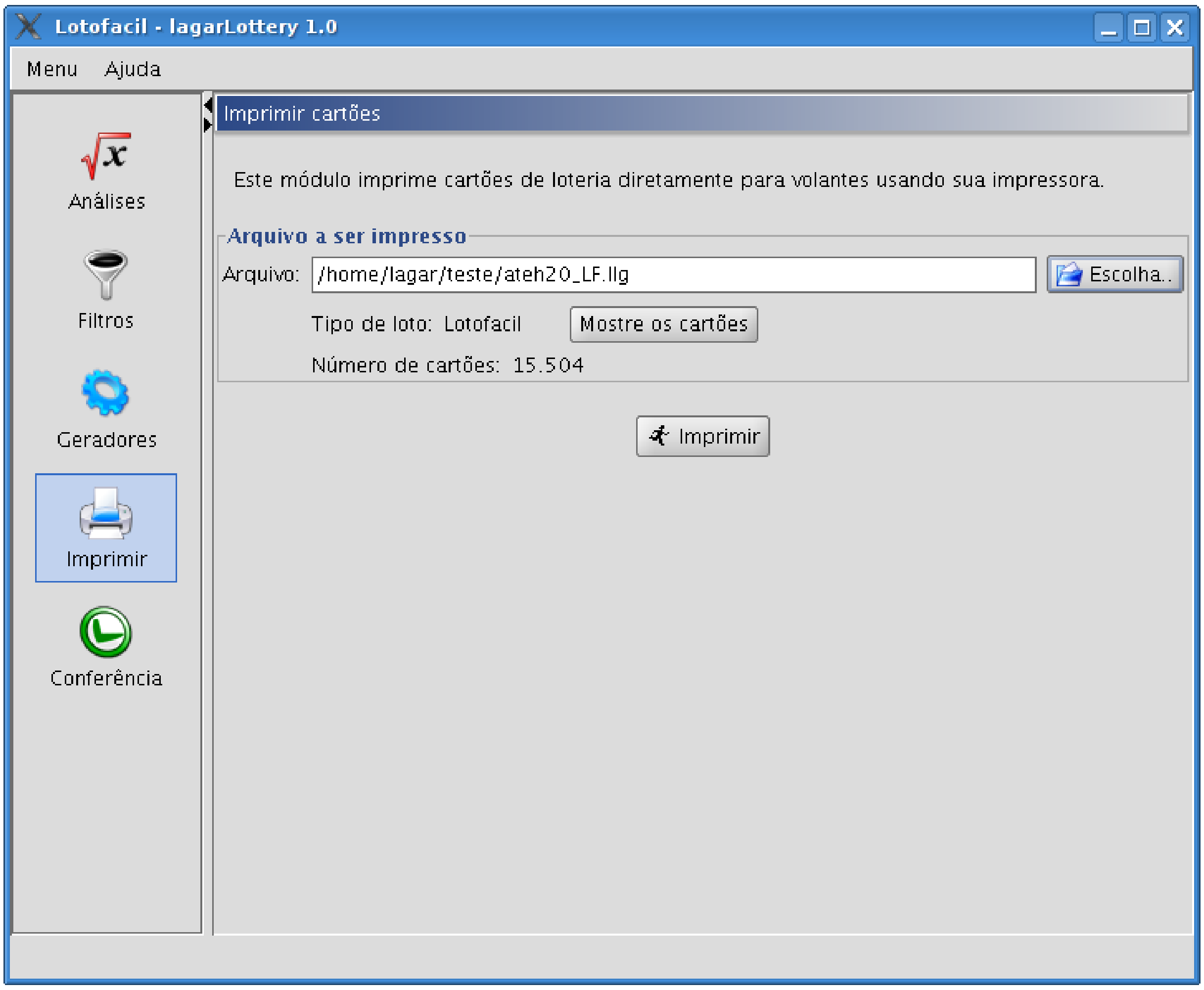
File to save

Arquivo:

Figura 7.4.2: Tela de uso do gerador de todas as combinações possíveis

7.5 Tela módulo impressão

O módulo impressão possui uma interface bastante simples, uma vez que uma implementação de classe responsável por imprimir um tipo de jogo específico já está especificada no arquivo de configuração; logo basta apenas o usuário informar que jogo gerado quer imprimir, que o sistema toma o trabalho de instanciar a classe concreta responsável por imprimir aquele tipo de jogo. Ver figura abaixo:



Este módulo imprime cartões de loteria diretamente para volantes usando sua impressora.

Arquivo a ser impresso

Arquivo: /home/lagar/teste/ateh20_LF.llg

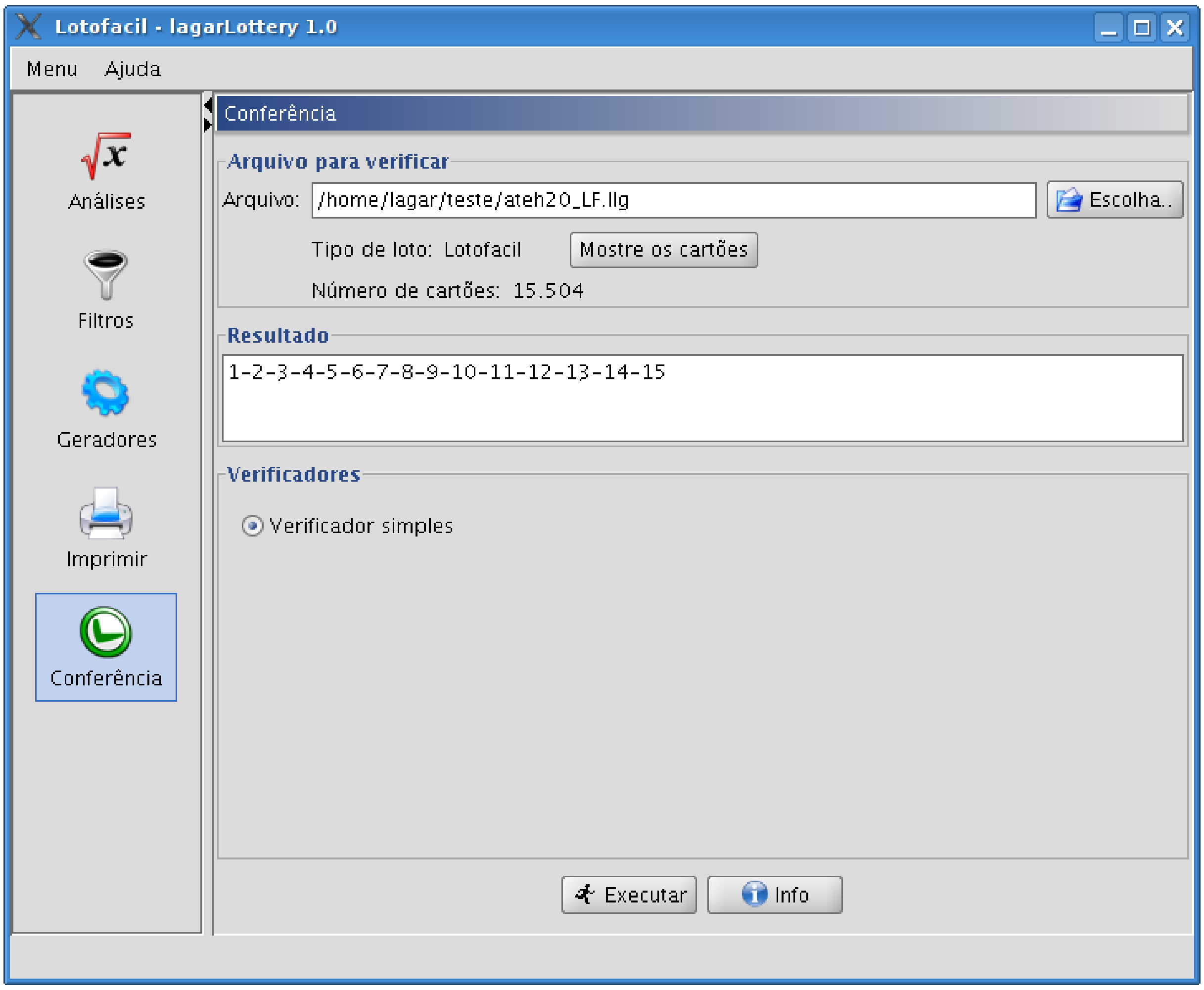
Tipo de loto: Lotofacil

Número de cartões: 15.504

Figura 7.5.1: Tela do módulo de impressão

7.6 Tela módulo verificador

Como entrada para um verificador, temos um jogo gerado e um resultado em particular. O usuário fornecendo estes parâmetros (ver figura abaixo) e escolhendo um verificador em particular (lista montada dinamicamente), há a execução de uma implementação específica de verificador (na figura, temos o verificador fornecido com o sistema, verificador "simples"). O resultado da execução do verificador simples é exibido na figura 7.6.2, com a rolagem posicionada na parte de Sumário.



Arquivo: /home/lagar/teste/ateh20_LF.llg

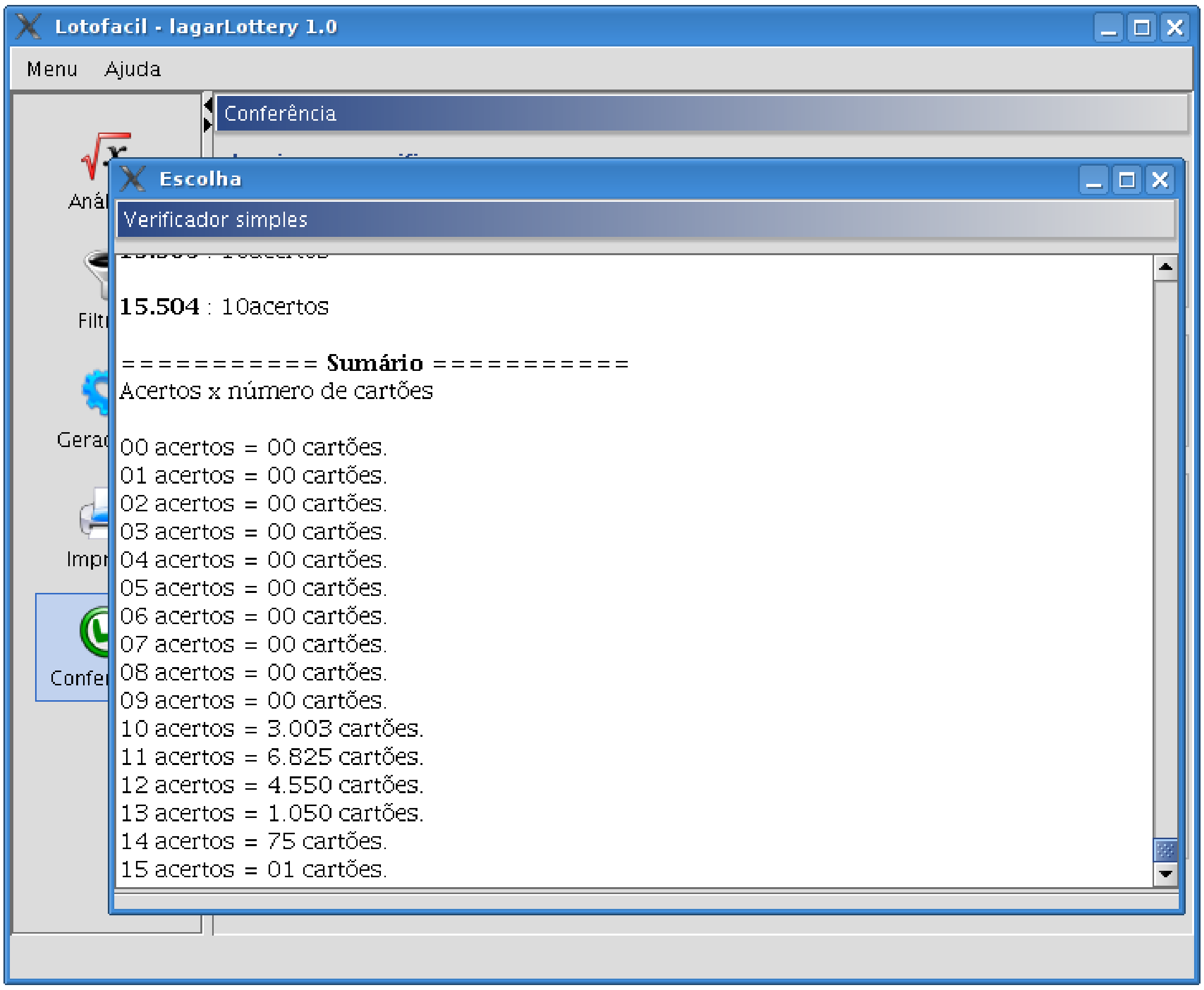
Tipo de loto: Lotofacil

Número de cartões: 15.504

1-2-3-4-5-6-7-8-9-10-11-12-13-14-15

(•) Verificador simples

Figura 7.6.1: Tela inicial do módulo verificador

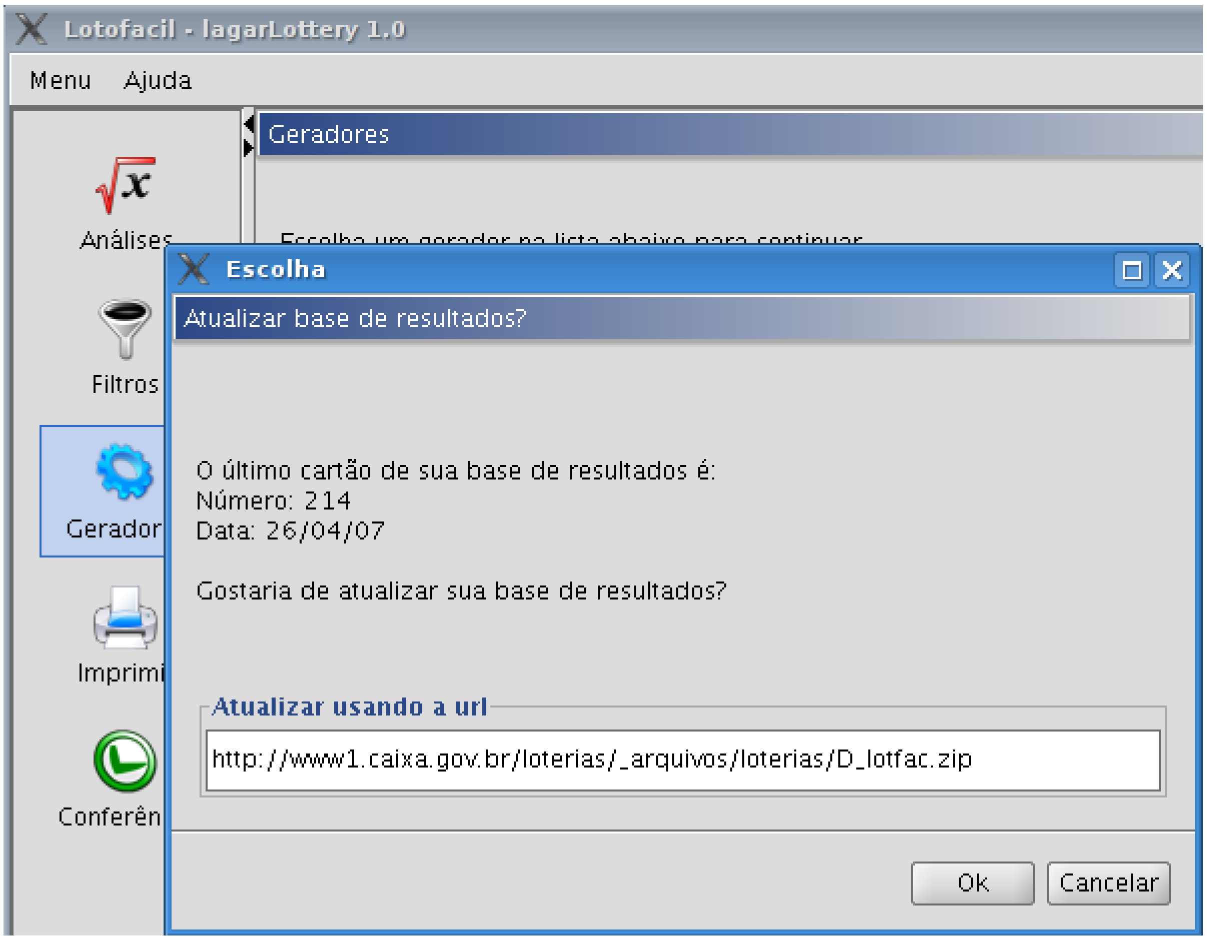


```
15.504 : 10acertos

=========== Sumário ===========
Acertos x número de cartões

00 acertos = 00 cartões.
01 acertos = 00 cartões.
02 acertos = 00 cartões.
03 acertos = 00 cartões.
04 acertos = 00 cartões.
05 acertos = 00 cartões.
06 acertos = 00 cartões.
07 acertos = 00 cartões.
08 acertos = 00 cartões.
09 acertos = 00 cartões.
10 acertos = 3.003 cartões.
11 acertos = 6.825 cartões.
12 acertos = 4.550 cartões.
13 acertos = 1.050 cartões.
14 acertos = 75 cartões.
15 acertos = 01 cartões.
```

Figura 7.6.2: Resultado do verificador simples

7.7 Tela atualização de resultados

Por fim, temos a tela de atualização da base de resultados. Ela exibe ao usuário o último cartão da base, e exibe a URL padrão que estava configurada no arquivo de configuração, podendo o usuário alterar neste momento. Se decidir por atualizar a base, o sistema conectará ao site especificado, irá fazer o download do arquivo apontado e construir a nova base de resultados. Ver figura abaixo para a tela.


Lotofacil - lagarLottery 1.0
Menu
Ajuda
Análises
Filtros
Gerador
Imprimi
Conferên
Geradores
Escolha um gerador na lista abaixo para continuar
Escolha
Atualizar base de resultados?
Atualizar usando a url
Ok
Cancelar

O último cartão de sua base de resultados é:
Número: 214
Data: 26/04/07

Gostaria de atualizar sua base de resultados?

http://www1.caixa.gov.br/loterias/_arquivos/loterias/D_lotfac.zip

Figura 7.7.1: Tela atualização da base de resultados

8 Considerações finais e trabalhos futuros

Com este trabalho, ao longo dos capítulos, mostrei que um software para jogadores de loteria é uma ferramenta útil, e que uma ferramenta desta natureza pode ser muito interessante, até para quem não costuma jogar muito na loto.

Um grande desafio deste trabalho foi tentar contornar o problema da extensibilidade: cada jogador tem sua idéia única e particular de gerar um jogo, realizar uma análise, filtrar cartões, em fim, cada jogador de loteria tem a sua grande idéia, e não é possível prever isso de antemão. A solução adotada neste trabalho foi deixar o software com vários pontos de extensibilidade (um em cada módulo) para que o jogador pudesse ele mesmo (se tiver conhecimento de programação) ou pedir para que outro o faça (o código é livre) para adicionar a sua tão particular funcionalidade. Se o software não tivesse sido projetado para ter os pontos de extensibilidade e usar a "arquitetura plugin", essa tarefa de adicionar código novo seria muito mais demorada e penosa. Mas com essa solução, uma nova análise ou filtro pode ser adicionado em questão de minutos.

Neste ponto temos uma possibilidade de trabalho futuro (versão 2.0) do software: permitir a inserção de código novo (análise, filtro, etc) através de uma linguagem de script, no estilo que ferramentas de escritório como o Microsoft Excel permite adicionar através de macros: o usuário poder, através de uma linguagem simples, especificar uma nova funcionalidade e executar esta nova funcionalidade no próprio ambiente.

Outra possibilidade de trabalho futuro está em capturar regras de análise do usuário, para que o software possa, através destas regras fornecidas, processar os dados da análise automaticamente. Por exemplo: imagine que você sempre olha a característica X da análise de pares x ímpares, procurando sempre por um padrão Y. Se você pudesse dizer isso para o software, este poderia colorir a tabela resultante com cores diferentes indicando um padrão fornecido.

Durante o desenvolvimento deste trabalho, colocou-se em prática muito conteúdo aprendido durante as disciplinas. Em especial, usou-se muito engenharia de software para criar o projeto deste trabalho: tentou-se aplicar boas práticas de desenvolvimento orientado a objeto, com forte influência do livro da referência [FREEMAN01], o qual indico com grande positividade.

Por fim, deixo o convite a realizar o download do projeto em http://sourceforge.net/ projects/lagar-lottery/ , experimentar o software e contribuir com suas idéias e código para que este seja um dos melhores entre os softwares para jogadores de loterias existentes.

9 Bibliografia

[CAIXA01] Repasses sociais das loterias. Disponível em: <http://www.caixa.gov.br/Loterias/Repasses_Sociais/>. Acesso em: abril de 2006.

[NASPL] North American Association of State and Provincial Lotteries. Disponível em: http://www.naspl.org/history.html. Acesso em: junho de 2006.

[CAIXA03] Tipos de jogos de loteria. Disponível em: http://www.caixa.gov.br/loterias/index.asp. Acesso em: junho de 2006.

[PAULO 01] BORGES, Paulo Sergio da Silva, Dr. Probabilidade, Inferência e Decisão. Universidade Federal de Santa Catarina. Tese titular. Ano: 1998. Capítulo 4

[GAIL01] HOWARD, Gail. Lottery master guide. 2003. Smart luck publishers

[MICHAEL01] Combination Generator. Disponível em: http://www.merriampark.com/comb.htm. Acesso em: abril de 2006.

[FREEMAN01] Eric Freeman e outros. Head First Design Patterns. 2004. Oreilly

[SUN01] Linguagem java. Disponível em: http://java.sun.com. Acesso em: abril de 2006.

[STROUSTRUP01] The C++ programming language. Disponível em: http://www.research.att.com/~bs/C++.html. Acesso em: abril de .

[ECLIPSE01] Eclipse IDE. Disponível em: http://www.eclipse.org. Acesso em: abril de 2006.

[NETBEANS01] Netbeans IDE.Disponível em: . Acesso em: abril de 2006.

[SOURCEFORGE01] SourceForge. Disponível em: http://sourceforge.net/. Acesso em: abril de 2006.

[JAVAWORLD01] Identify subclasses at runtime. Disponível em: www.javaworld.com/javatips/jw-javatip113_p.html. Acesso em: abril de 2006.

- Não Referenciada:

JONES, Prof. **Winning Lotto/Lottery for everyday players.** Estados Unidos: Cardoza publishing, 2002.

Wheels / Covering Designs. Disponível em: <http://thelotteryinstitute.tripod.com/wheels.htm>. Acesso em: fevereiro de 2006.

La Jolla Covering Repository. Disponível em: <http://www.ccrwest.org/cover.html>. Acesso em: fevereiro de 2006.

GNU GPL, licença de software. Disponível em: <http://www.gnu.org/copyleft/gpl.html>. Acesso em: março de 2006.

10 Apêndice A – Código fonte

```
/*
 * AboutDialog.java
 *
 * Created on 16 de Maio de 2007, 22:18
 */

package br.ufsc.inf.lagar.main;

import java.awt.Color;

/**
 *
 * @author  lagar
 */
public class AboutDialog extends javax.swing.JDialog {

    /** Creates new form AboutDialog */
    public AboutDialog(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
        this.setLocationRelativeTo(null);
        this.setBackground(Color.WHITE);
        this.pngLabel.setBackground(Color.WHITE);
        this.pngLabel.setForeground(Color.WHITE);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        pngLabel = new javax.swing.JLabel();
        jSeparator1 = new javax.swing.JSeparator();
        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jButton1 = new javax.swing.JButton();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("About");
        setBackground(java.awt.Color.white);
        setResizable(false);
        pngLabel.setBackground(new java.awt.Color(255, 255, 255));
        pngLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        pngLabel.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/LagarLottery1.png")));
        pngLabel.setToolTipText("imagem por Amaweks");

        jLabel1.setFont(new java.awt.Font("Dialog", 1, 18));
        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel1.setText("LagarLottery 1.0");

        jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel2.setText("Euclides Pinheiro de Melo");
        jLabel3.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel3.setText("lagar_lottery@yahoo.com.br");

        org.jdesktop.layout.GroupLayout jPanel1Layout = new
org.jdesktop.layout.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .add(jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayou
t.LEADING)
                    .add(jLabel2, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 264,
Short.MAX_VALUE)
                    .add(jLabel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 264,
Short.MAX_VALUE)
                    .add(jLabel3, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 264,
Short.MAX_VALUE))
                .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .add(jLabel1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 34,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jLabel2)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jLabel3)
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        jButton1.setText("Terms");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jSeparator1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 288,
Short.MAX_VALUE)
            .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, pngLabel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 288, Short.MAX_VALUE)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .add(jLabel4, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 96,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jButton1)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jLabel5, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 96,
Short.MAX_VALUE))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(pngLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 179,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jSeparator1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 10,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jPanel1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
                    .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.B
ASELINE)
                        .add(jButton1)
                        .add(jLabel4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
24, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(jLabel5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 24,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(12, Short.MAX_VALUE))
        );

        layout.linkSize(new java.awt.Component[] {jLabel4, jLabel5},
org.jdesktop.layout.GroupLayout.VERTICAL);

        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_jButton1ActionPerformed

        new TermsDialog(null, false).setVisible(true);

    }//GEN-LAST:event_jButton1ActionPerformed



    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JLabel pngLabel;
    // End of variables declaration//GEN-END:variables

}
/*
 * AbstractGameFilter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter;

import br.ufsc.inf.lagar.main.GameCard;

public abstract class AbstractGameFilter implements Filterable {

    private boolean mode;
    private String input;

    /**
     * Creates a new instance of AbstractGameFilter
     */
    public AbstractGameFilter() {
    }

    public void setAcceptReject(boolean mode) {
        this.mode = mode;
    }

    public void setTextInput(String input) throws TextInputFilterException {
        this.validateInput(input);
        this.input = input;
    }

    public abstract void validateInput(String input) throws
TextInputFilterException;


    public boolean isMode() {
        return mode;
    }

    public String getInput() {
        return input;
    }

    public boolean acceptGame(String gameName){
        return true;
    }

    public String toString(){

        return this.getFullName();
    }

}
/*
 * AllColumnFilter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter.gamefilter;

import br.ufsc.inf.lagar.filter.AbstractGameFilter;
import br.ufsc.inf.lagar.filter.Filterable;
import br.ufsc.inf.lagar.filter.TextInputFilterException;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameManager;
import java.util.Arrays;

public class AllColumnFilter extends AbstractGameFilter{

    private int[] inputBalls;
    private int lineSize;
    private int[] catsCount;
    private int[][] colsData;
    private int numCols;

    /**
     * Creates a new instance of AllColumnFilter
     */
    public AllColumnFilter() {
        this.lineSize = GameManager.getCurrentGame().getNumbersPerLine();
    }

    public void validateInput(String input) throws TextInputFilterException {

        String[] params = input.split(Filterable.OR_SEPARATOR);
        this.inputBalls = new int[params.length];

        for(int i = 0; i < params.length; ++i){

           String s = params[i];

           try{
                this.inputBalls[i] = Integer.parseInt(s.trim());
            }
            catch(NumberFormatException nfex){
                throw new TextInputFilterException(nfex);
            }

        }

        Arrays.sort(this.inputBalls);
        this.catsCount = new int[this.lineSize+1];
        this.numCols = GameManager.getCurrentGame().getNumbersPerCard() /
this.lineSize;

        for(int i = 1; i <= this.lineSize; ++i){
            this.colsData = new int[this.lineSize+1][numCols];
        }

        for(int i = 1; i <= this.lineSize; ++i){

            int currNum = i;
            for(int j = 0; j < numCols; ++j){

                this.colsData[i][j] = currNum;
                currNum += this.lineSize;
            }

        }

    }

    public String getShortName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("All_column_filter");
    }

    public String getFullName() {
        return getShortName();
    }

    public String getDescription() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("All_column_filter_description");
    }

    public String getFilterHelp() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("All_column_filter_help");
    }

    public boolean filter(GameCard card) {

        Arrays.fill(this.catsCount, 0);

        byte[] balls = card.getBalls();

        for(int i = 0; i < balls.length; ++i){
            int ball = balls[i];

            for(int k = 1; k <= numCols; ++k){
                int[] col = this.colsData[k];
                if(Arrays.binarySearch(col, ball) >= 0){
                    ++this.catsCount[k];
                    break;
                }
            }
        }

        if(this.isMode()){
            //accept mode
            boolean ok = true;
            for(int i = 1; i < this.catsCount.length; ++i){

                int b = this.catsCount[i];
                if(Arrays.binarySearch(this.inputBalls, b) < 0){
                    return false;
                }
            }
            return this.isMode();

        }
        else{
            //reject mode
            for(int i = 1; i < this.catsCount.length; ++i){

                int b = this.catsCount[i];
                if(Arrays.binarySearch(this.inputBalls, b) >= 0){
                    return this.isMode();
                }
            }
            return !this.isMode();
        }
    }

}
/*
 * AllCombinationGeneratorService.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.generator;

import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.main.LagarLotteryGlobals;
import br.ufsc.inf.lagar.ui.ProgressInterface;
import java.io.BufferedOutputStream;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.text.NumberFormat;
import java.util.Date;
import java.util.zip.Deflater;
import java.util.zip.GZIPOutputStream;
import java.util.zip.ZipEntry;
import java.util.zip.ZipOutputStream;

public class AllCombinationGeneratorService implements ProgressInterface{

    private CombinationGenerator cg;
    private File path;
    private int[] numbers;
    private int numbersPerCard;
    private boolean cancel = false;
    private int genCount = 0;
    private int total = 0;

    /** Creates a new instance of AllCombinationGeneratorService */
    public AllCombinationGeneratorService(String savePath, int[] numbers, int
numbersPerCard) throws IOException {

        this.path = new File(savePath);
        if(this.path.exists()) this.path.delete();

        this.path.createNewFile();

        this.numbers = numbers;

        this.numbersPerCard = numbersPerCard;

        this.cg = new CombinationGenerator(this.numbers.length, this.numbersPerCard);
        this.total = cg.getTotal().intValue();
    }

    public void generateAllCombinations() throws IOException{


        int[] indices;
        Date today = new Date();

        FileOutputStream fos = new FileOutputStream(this.path);
        ZipOutputStream zos = new ZipOutputStream(fos);
        zos.setLevel(Deflater.BEST_SPEED);

        DataOutputStream dos = new DataOutputStream(zos);

        GeneratorHeader header = new GeneratorHeader();
        header.setGameName(GameManager.getCurrentGame().getName());
        header.setBallsPerCard(this.numbersPerCard);
        header.setNumbers(this.numbers);
        header.setNumberOfCards(this.total);

        zos.putNextEntry(new ZipEntry(LagarLotteryGlobals.GENERATED_HEADER_NAME));
        header.saveToStream(dos);
        dos.flush();
        zos.closeEntry();
        zos.putNextEntry(new ZipEntry(LagarLotteryGlobals.GENERATED_CARDS_NAME));

        byte[] balls = new byte[this.numbersPerCard];
        while(this.cg.hasMore())
        {
            indices = this.cg.getNext();
            for(int i = 0; i < indices.length; i++)
            {
                balls[i] = (byte) this.numbers[indices[i]];
            }
            dos.write(balls);

            ++(this.genCount);

            if(this.cancel){
                dos.close();
                this.path.delete();
                throw new IOException("Cancelled");
            }
        }
        dos.flush();
        zos.closeEntry();


        dos.close();

    }

    public void cancel() {
        this.cancel = true;
    }

    public String getProgressText() {

       NumberFormat nf = NumberFormat.getInstance();

       return nf.format(genCount) + " of " + nf.format(total) + " generated.";
    }

    public int getProgressValue() {
        return this.genCount;
    }

    public int getMaxProgressValue() {
        return this.total;
    }

}
/*
 * AllCombinationGeneratorServiceTest.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.core.generator;

import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.generator.AllCombinationGeneratorService;
import java.util.List;
import junit.framework.*;
import br.ufsc.inf.lagar.main.GameCard;
import java.beans.XMLEncoder;
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileNotFoundException;
```

```java
import java.io.FileOutputStream;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Date;
import org.omg.SendingContext.RunTime;

/**
 *
 * @author lagar
 */
public class AllCombinationGeneratorServiceTest extends TestCase {

    public AllCombinationGeneratorServiceTest(String testName) {
        super(testName);
    }

    /**
     * Test of generateAllCombinations method, of class
br.ufsc.inf.lagar.core.generator.AllCombinationGeneratorService.
     */
    public void testGenerateAllCombinations() throws IOException {
        System.out.println("generateAllCombinations");

        List<GameConfiguration> availableGames = GameManager.getAvailableGames();
        GameConfiguration gc = availableGames.get(1);

        GameManager.setCurrentGame(gc);

        AllCombinationGeneratorService instance = new AllCombinationGeneratorService(
                "/home/lagar/cards.llg", new int[]{1,2,3,4,5,6,7,8,9,10,
                11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,
                31,32,33,34,35}, 6);

        instance.generateAllCombinations();


    }

}
/*
 * AllLineFilter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter.gamefilter;

import br.ufsc.inf.lagar.filter.AbstractGameFilter;
import br.ufsc.inf.lagar.filter.Filterable;
import br.ufsc.inf.lagar.filter.TextInputFilterException;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameManager;
import java.util.Arrays;

public class AllLineFilter extends AbstractGameFilter {

    private int[] inputBalls;
    private int lineSize;
    private int numLines;
    private int[] catsCount;

    /**
     * Creates a new instance of AllLineFilter
     */
    public AllLineFilter() {
        this.lineSize = GameManager.getCurrentGame().getNumbersPerLine();
    }

    public void validateInput(String input) throws TextInputFilterException {

        String[] params = input.split(Filterable.OR_SEPARATOR);
        this.inputBalls = new int[params.length];

        for(int i = 0; i < params.length; ++i){

           String s = params[i];

           if(s.trim().startsWith("s=")){
               try{
                    String[] ls = s.split("=");
                    this.lineSize = Integer.parseInt(ls[1].trim());
                }
                catch(Exception ex){
                    throw new TextInputFilterException(ex);
                }
           }
           else{
               try{
                    this.inputBalls[i] = Integer.parseInt(s.trim());
                }
                catch(NumberFormatException nfex){
                    throw new TextInputFilterException(nfex);
                }
           }

        }

        Arrays.sort(this.inputBalls);
        this.numLines = GameManager.getCurrentGame().getNumbersPerCard() /
this.lineSize;
        this.catsCount = new int[this.numLines];

    }

    public String getShortName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("All_line_filter");
    }

    public String getFullName() {
        return this.getShortName();
    }

    public String getDescription() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("All_line_filter_description");
    }

    public String getFilterHelp() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("All_line_filter_help");
    }

    public boolean filter(GameCard card) {

        Arrays.fill(this.catsCount, 0);

        byte[] balls = card.getBalls();

        for(int i = 0; i < balls.length; ++i){
            int ball = balls[i];

            int cat = (ball-1) / this.lineSize;
            ++catsCount[cat]; //increment sum on the line
        }


        if(this.isMode()){
            //accept mode
            boolean ok = true;
            for(int b: this.catsCount){
                if(Arrays.binarySearch(this.inputBalls, b) < 0){
                    return false;
                }
            }
            return this.isMode();

        }
        else{
            //reject mode
            for(int b: this.catsCount){
                if(Arrays.binarySearch(this.inputBalls, b) >= 0){
                    return this.isMode();
                }
            }
            return !this.isMode();
        }

    }

}
/*
 * AnalysisControlPanel.java
 *
 * Created on 7 de Janeiro de 2007, 19:04
 */

package br.ufsc.inf.lagar.ui.panels;

import br.ufsc.inf.lagar.analysis.Analysis;
import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.ui.dialog.DecisionDialog;
import br.ufsc.inf.lagar.ui.frame.AnalysisFrame;
import br.ufsc.inf.lagar.utils.RTSI;
import java.awt.Dimension;
import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import java.util.ResourceBundle;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.JTextPane;
import javax.swing.ListModel;
import javax.swing.event.ListDataListener;

/**
 *
 * @author  lagar
 */
public class AnalysisControlPanel extends javax.swing.JPanel {

    public static enum SourceMode{
        FROM_ANALYSIS_DATABASE,
        FROM_LLG_FILE
    }

    private SourceMode sourceMode = SourceMode.FROM_ANALYSIS_DATABASE;

    /**
     * Creates new form AnalysisControlPanel
     */
    private GameConfiguration currentGame = null;

    public AnalysisControlPanel() {
        initComponents();
        this.currentGame = GameManager.getCurrentGame();
        this.updateMaxCardsResultsDb(currentGame.getGameResults().getLoadedResults().
size());
        this.fileChooserLLG1.setMode(FileChooserLLG.ChooserMode.LOAD_MODE);
        this.statusLabel.setVisible(false);
    }

    public void updateMaxCardsResultsDb(int cards){
        this.toField.setText(String.valueOf(cards));
    }

    private void processInfoButton() {
        Analysis analysis = (Analysis) this.analysisList.getSelectedValue();
        if(analysis != null){

            String info = analysis.getHelpInfo();
            DecisionDialog dd = new DecisionDialog(null, false);
            JTextPane textPane = new JTextPane();
            textPane.setText(info);
            JScrollPane pane = new JScrollPane(textPane,
                    JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                    JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);

            dd.getMainPanel().add(pane, pane.getClass().getName());
            dd.getButtonsPanel().setVisible(false);
            dd.getTitlePanel().setTitle(analysis.getName());
            dd.setTitle(ResourceBundle.getBundle("resources/bundles/lagarLottery").ge
tString("Help"));
            dd.setVisible(true);
        }
    }

    private void processRunButton() {

        int fromRes = -1;
        int toRes = -1;
        int jumpRes = -1;
        GenCardIO genCardIO = null;

        if(this.sourceMode == SourceMode.FROM_ANALYSIS_DATABASE){
            try{
            fromRes = Integer.parseInt(this.fromField.getText());
            toRes = Integer.parseInt(this.toField.getText());
            jumpRes = Integer.parseInt(this.jumpField.getText());
            }
            catch(Exception ex){
                JOptionPane.showMessageDialog(this,
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("Results_range_contains_invalid_data"),
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("Error"),
                        JOptionPane.ERROR_MESSAGE);
                return;
            }
            if(fromRes < 0 || toRes < 0 || jumpRes < 0){
                JOptionPane.showMessageDialog(this,
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("Results_range_contains_invalid_data"),
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("Error"),
                        JOptionPane.ERROR_MESSAGE);
                return;
            }

            if(this.currentGame.getGameResults().getLoadedResults().size() == 0){
                JOptionPane.showMessageDialog(this,
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("You_don't_have_any_loaded_results."),
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("Error"),
                        JOptionPane.ERROR_MESSAGE);
                return;
            }

        }
        else{
            try {
                genCardIO = new GenCardIO(new
File(this.fileChooserLLG1.getFilePath()));
            } catch (IOException ex) {
                JOptionPane.showMessageDialog(this,
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("Invalid_LLG_file."),
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("Error"),
                        JOptionPane.ERROR_MESSAGE);
                return;
            }
        }
        final Analysis analysis = (Analysis) this.analysisList.getSelectedValue();
        if(analysis != null){
            analysis.setBegin(fromRes);
            analysis.setEnd(toRes);
            analysis.setJump(jumpRes);
            analysis.setSourceMode(this.sourceMode);
            analysis.setGenCardIO(genCardIO);
            analysis.setCurrentGameConfiguration(this.currentGame);

            this.statusLabel.setText(java.util.ResourceBundle.getBundle("resources/bu
ndles/lagarLottery").getString("Loading_")+analysis.getName()+"
"+java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("_in
_background.."));

            Thread t = new Thread(){
                public void run(){
                    final AnalysisFrame frame = new AnalysisFrame(analysis);
                    frame.setLocationRelativeTo(null);
                    //frame.setExtendedState(JFrame.MAXIMIZED_BOTH);
                    frame.setPreferredSize(new Dimension(800,600));
                    frame.setSize(new Dimension(800,600));
                    frame.showAnalysis();
                }
            };
            t.start();
        }
    }

    public void fillAvailableAnalysis() throws Exception{

        final List<Analysis> all = new ArrayList<Analysis>();
        String packageName = "br.ufsc.inf.lagar.analysis.gameanalysis";

        ArrayList<String> classes = RTSI.find(packageName, Analysis.class);
        for(String clazz: classes){

            clazz = packageName+"."+clazz;
            Analysis analysis = (Analysis) Class.forName(clazz).newInstance();
            all.add(analysis);
        }

        this.analysisList.setModel(new ListModel() {
            public void addListDataListener(ListDataListener l) {
            }
            public Object getElementAt(int index) {
                return all.get(index);
            }
            public int getSize() {
                return all.size();
            }
            public void removeListDataListener(ListDataListener l) {
            }
        });

        this.analysisList.setSelectedIndex(0);

    }

    private void processAllCheckBox() {

        if(this.allResultsCheckBox.isSelected()){
            this.toField.setText(String.valueOf(currentGame.getGameResults().getLoade
dResults().size()));
            this.fromField.setText("1");
            this.jumpField.setText("1");
            this.fromLabel.setEnabled(false);
            this.fromField.setEnabled(false);
            this.toLabel.setEnabled(false);
            this.toField.setEnabled(false);
            this.jumpLabel.setEnabled(false);
            this.jumpField.setEnabled(false);
        }
        else{
            this.fromLabel.setEnabled(true);
            this.fromField.setEnabled(true);
            this.toLabel.setEnabled(true);
            this.toField.setEnabled(true);
            this.jumpLabel.setEnabled(true);
            this.jumpField.setEnabled(true);
        }
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        buttonGroup1 = new javax.swing.ButtonGroup();
        simpleInternalFrame1 = new br.inf.ufsc.lagar.swingutil.SimpleInternalFrame();
        mainPanel = new javax.swing.JPanel();
        rangePanel = new javax.swing.JPanel();
        fromLabel = new javax.swing.JLabel();
        fromField = new javax.swing.JTextField();
        toLabel = new javax.swing.JLabel();
        toField = new javax.swing.JTextField();
        allResultsCheckBox = new javax.swing.JCheckBox();
        jumpLabel = new javax.swing.JLabel();
        jumpField = new javax.swing.JTextField();
        resultsRadio = new javax.swing.JRadioButton();
        llgRadio = new javax.swing.JRadioButton();
        fileChooserLLG1 = new br.ufsc.inf.lagar.ui.panels.FileChooserLLG();
        availablePanel = new javax.swing.JPanel();
        scrollPane = new javax.swing.JScrollPane();
        analysisList = new javax.swing.JList();
        runButton = new javax.swing.JButton();
        infoButton = new javax.swing.JButton();
        labelSepLeft = new javax.swing.JLabel();
        labelSepRight = new javax.swing.JLabel();
        statusPanel = new javax.swing.JPanel();
        statusLabel = new javax.swing.JLabel();

        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        simpleInternalFrame1.setTitle(bundle.getString("Analysis")); // NOI18N

        rangePanel.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.getS
tring("Range"))); // NOI18N
        rangePanel.setToolTipText(bundle.getString("Only_affects_new_analysis")); //
NOI18N
        fromLabel.setText(bundle.getString("From_result")); // NOI18N
        fromLabel.setEnabled(false);

        fromField.setHorizontalAlignment(javax.swing.JTextField.CENTER);
        fromField.setText("1");
        fromField.setEnabled(false);

        toLabel.setText(bundle.getString("to")); // NOI18N
        toLabel.setEnabled(false);

        toField.setHorizontalAlignment(javax.swing.JTextField.CENTER);
        toField.setText("10000");
        toField.setEnabled(false);

        allResultsCheckBox.setSelected(true);
        allResultsCheckBox.setText(bundle.getString("All_results")); // NOI18N
        allResultsCheckBox.setBorder(javax.swing.BorderFactory.createEmptyBorder(0,
0, 0, 0));
        allResultsCheckBox.setMargin(new java.awt.Insets(0, 0, 0, 0));
        allResultsCheckBox.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                allResultsCheckBoxActionPerformed(evt);
            }
        });

        jumpLabel.setText(bundle.getString("Jump_results_by")); // NOI18N
        jumpLabel.setEnabled(false);

        jumpField.setColumns(4);
        jumpField.setHorizontalAlignment(javax.swing.JTextField.CENTER);
        jumpField.setText("1");
        jumpField.setEnabled(false);

        buttonGroup1.add(resultsRadio);
        resultsRadio.setSelected(true);
        resultsRadio.setText(bundle.getString("Analysis_from_results_database")); //
NOI18N
        resultsRadio.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0,
0));
        resultsRadio.setMargin(new java.awt.Insets(0, 0, 0, 0));
        resultsRadio.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                resultsRadioActionPerformed(evt);
            }
        });

        buttonGroup1.add(llgRadio);
        llgRadio.setText(bundle.getString("Analysis_from_generated_game_(.LLG)")); //
NOI18N
        llgRadio.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0, 0));
        llgRadio.setMargin(new java.awt.Insets(0, 0, 0, 0));
        llgRadio.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                llgRadioActionPerformed(evt);
            }
        });

        fileChooserLLG1.setEnabled(false);

        org.jdesktop.layout.GroupLayout rangePanelLayout = new
org.jdesktop.layout.GroupLayout(rangePanel);
        rangePanel.setLayout(rangePanelLayout);
        rangePanelLayout.setHorizontalGroup(
            rangePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEAD
ING)
```

```
            .add(rangePanelLayout.createSequentialGroup()
                .addContainerGap()
                .add(rangePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLa
yout.LEADING)
                    .add(org.jdesktop.layout.GroupLayout.TRAILING,
rangePanelLayout.createSequentialGroup()
                        .add(17, 17, 17)
                        .add(fileChooserLLG1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 481, Short.MAX_VALUE))
                    .add(rangePanelLayout.createSequentialGroup()
                        .add(17, 17, 17)
                        .add(rangePanelLayout.createParallelGroup(org.jdesktop.layout
.GroupLayout.LEADING)
                            .add(allResultsCheckBox)
                            .add(rangePanelLayout.createSequentialGroup()
                                .add(rangePanelLayout.createParallelGroup(org.jdeskto
p.layout.GroupLayout.TRAILING)
                                    .add(fromLabel)
                                    .add(jumpLabel))
                                .add(rangePanelLayout.createParallelGroup(org.jdeskto
p.layout.GroupLayout.LEADING, false)
                                    .add(rangePanelLayout.createSequentialGroup()
                                        .addPreferredGap(org.jdesktop.layout.LayoutSt
yle.RELATED)
                                        .add(fromField,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 50,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                                    .add(rangePanelLayout.createSequentialGroup()
                                        .add(5, 5, 5)
                                        .add(jumpField)))
                                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELA
TED)
                                .add(toLabel)
                                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELA
TED)
                                .add(toField,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 45,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))))
                    .add(resultsRadio)
                    .add(llgRadio))
                .addContainerGap())
        );

        rangePanelLayout.linkSize(new java.awt.Component[] {fromField, toField},
org.jdesktop.layout.GroupLayout.HORIZONTAL);

        rangePanelLayout.setVerticalGroup(
            rangePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEAD
ING)
            .add(rangePanelLayout.createSequentialGroup()
                .add(resultsRadio)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(allResultsCheckBox)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(rangePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLa
yout.BASELINE)
                    .add(fromLabel)
                    .add(fromField, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(toLabel)
                    .add(toField, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
```

```
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(rangePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLa
yout.BASELINE)
                    .add(jumpLabel)
                    .add(jumpField, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(llgRadio)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(fileChooserLLG1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        rangePanelLayout.linkSize(new java.awt.Component[] {fromField, toField},
org.jdesktop.layout.GroupLayout.VERTICAL);

        availablePanel.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.
getString("Available_analysis"))); // NOI18N
        scrollPane.setHorizontalScrollBarPolicy(javax.swing.ScrollPaneConstants.HORIZ
ONTAL_SCROLLBAR_NEVER);
        analysisList.setSelectionMode(javax.swing.ListSelectionModel.SINGLE_SELECTION
);
        analysisList.addListSelectionListener(new
javax.swing.event.ListSelectionListener() {
            public void valueChanged(javax.swing.event.ListSelectionEvent evt) {
                analysisListValueChanged(evt);
            }
        });

        scrollPane.setViewportView(analysisList);

        runButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/runprog.png")));
        runButton.setText(bundle.getString("Run")); // NOI18N
        runButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                runButtonActionPerformed(evt);
            }
        });

        infoButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/info.png")));
        infoButton.setText(bundle.getString("Info")); // NOI18N
        infoButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                infoButtonActionPerformed(evt);
            }
        });

        org.jdesktop.layout.GroupLayout statusPanelLayout = new
org.jdesktop.layout.GroupLayout(statusPanel);
        statusPanel.setLayout(statusPanelLayout);
        statusPanelLayout.setHorizontalGroup(
            statusPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEA
DING)
            .add(statusLabel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 498,
Short.MAX_VALUE)
        );
        statusPanelLayout.setVerticalGroup(
            statusPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEA
DING)
            .add(statusLabel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 44,
Short.MAX_VALUE)
        );

        org.jdesktop.layout.GroupLayout availablePanelLayout = new
org.jdesktop.layout.GroupLayout(availablePanel);
        availablePanel.setLayout(availablePanelLayout);
        availablePanelLayout.setHorizontalGroup(
            availablePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.
LEADING)
            .add(availablePanelLayout.createSequentialGroup()
                .addContainerGap()
                .add(availablePanelLayout.createParallelGroup(org.jdesktop.layout.Gro
upLayout.LEADING)
                    .add(org.jdesktop.layout.GroupLayout.TRAILING,
availablePanelLayout.createSequentialGroup()
                        .add(labelSepLeft,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 155, Short.MAX_VALUE)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(runButton)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(infoButton)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(labelSepRight,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 155, Short.MAX_VALUE))
                    .add(scrollPane, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
498, Short.MAX_VALUE)
                    .add(statusPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
                .addContainerGap())
        );

        availablePanelLayout.linkSize(new java.awt.Component[] {infoButton,
runButton}, org.jdesktop.layout.GroupLayout.HORIZONTAL);

        availablePanelLayout.setVerticalGroup(
            availablePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.
LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
availablePanelLayout.createSequentialGroup()
                .add(scrollPane, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 120,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(statusPanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(availablePanelLayout.createParallelGroup(org.jdesktop.layout.Gro
upLayout.LEADING)
                    .add(availablePanelLayout.createParallelGroup(org.jdesktop.layout
.GroupLayout.BASELINE)
                        .add(runButton)
                        .add(infoButton))
                    .add(labelSepLeft,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(labelSepRight,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );

        availablePanelLayout.linkSize(new java.awt.Component[] {infoButton,
runButton}, org.jdesktop.layout.GroupLayout.VERTICAL);

        availablePanelLayout.linkSize(new java.awt.Component[] {labelSepLeft,
labelSepRight}, org.jdesktop.layout.GroupLayout.VERTICAL);

        org.jdesktop.layout.GroupLayout mainPanelLayout = new
org.jdesktop.layout.GroupLayout(mainPanel);
        mainPanel.setLayout(mainPanelLayout);
        mainPanelLayout.setHorizontalGroup(
            mainPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(availablePanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(rangePanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        mainPanelLayout.setVerticalGroup(
            mainPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
mainPanelLayout.createSequentialGroup()
                .add(rangePanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(availablePanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, mainPanel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, simpleInternalFrame1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 532, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(simpleInternalFrame1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
    }// </editor-fold>//GEN-END:initComponents

    private void llgRadioActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_llgRadioActionPerformed

        this.sourceMode = SourceMode.FROM_LLG_FILE;
        this.allResultsCheckBox.setEnabled(false);
        this.fromLabel.setEnabled(false);
        this.fromField.setEnabled(false);
        this.toLabel.setEnabled(false);
        this.toField.setEnabled(false);
        this.jumpLabel.setEnabled(false);
        this.jumpField.setEnabled(false);
        this.fileChooserLLG1.setEnabled(true);
    }//GEN-LAST:event_llgRadioActionPerformed

    private void resultsRadioActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_resultsRadioActionPerformed

        this.sourceMode = SourceMode.FROM_ANALYSIS_DATABASE;
        this.allResultsCheckBox.setEnabled(true);
        if(!this.allResultsCheckBox.isSelected()){
            this.fromLabel.setEnabled(true);
            this.fromField.setEnabled(true);
            this.toLabel.setEnabled(true);
            this.toField.setEnabled(true);
            this.jumpLabel.setEnabled(true);
            this.jumpField.setEnabled(true);
        }
        this.fileChooserLLG1.setEnabled(false);
    }//GEN-LAST:event_resultsRadioActionPerformed

    private void analysisListValueChanged(javax.swing.event.ListSelectionEvent evt)
{//GEN-FIRST:event_analysisListValueChanged

    }//GEN-LAST:event_analysisListValueChanged

    private void infoButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_infoButtonActionPerformed

        this.processInfoButton();
    }//GEN-LAST:event_infoButtonActionPerformed

    private void runButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_runButtonActionPerformed

        this.processRunButton();
    }//GEN-LAST:event_runButtonActionPerformed

    private void allResultsCheckBoxActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_allResultsCheckBoxActionPerformed

        this.processAllCheckBox();
    }//GEN-LAST:event_allResultsCheckBoxActionPerformed



    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JCheckBox allResultsCheckBox;
    private javax.swing.JList analysisList;
    private javax.swing.JPanel availablePanel;
    private javax.swing.ButtonGroup buttonGroup1;
    private br.ufsc.inf.lagar.ui.panels.FileChooserLLG fileChooserLLG1;
    private javax.swing.JTextField fromField;
    private javax.swing.JLabel fromLabel;
    private javax.swing.JButton infoButton;
    private javax.swing.JTextField jumpField;
    private javax.swing.JLabel jumpLabel;
    private javax.swing.JLabel labelSepLeft;
    private javax.swing.JLabel labelSepRight;
    private javax.swing.JRadioButton llgRadio;
    private javax.swing.JPanel mainPanel;
    private javax.swing.JPanel rangePanel;
    private javax.swing.JRadioButton resultsRadio;
    private javax.swing.JButton runButton;
    private javax.swing.JScrollPane scrollPane;
    private br.inf.ufsc.lagar.swingutil.SimpleInternalFrame simpleInternalFrame1;
    private javax.swing.JLabel statusLabel;
    private javax.swing.JPanel statusPanel;
    private javax.swing.JTextField toField;
    private javax.swing.JLabel toLabel;
    // End of variables declaration//GEN-END:variables

}
/*
 * AnalysisFrame.java
 *
 * Created on 15 de Janeiro de 2007, 23:22
 */

package br.ufsc.inf.lagar.ui.frame;

import br.ufsc.inf.lagar.analysis.Analysis;
import java.util.List;
import javax.swing.JPanel;

/**
 *
 * @author  lagar
 */
public class AnalysisFrame extends javax.swing.JFrame {

    private Analysis analysis;

    /** Creates new form AnalysisFrame */
    public AnalysisFrame(Analysis analysis) {
        this.analysis = analysis;
        initComponents();
        this.setTitle(analysis.getName());
    }

    public void showAnalysis(){
        this.analysis.calculateAnalysis();
        List<JPanel> panels = this.analysis.getPanels();

        for(JPanel panel: panels){
            if(panel != null){
                this.tabsPane.addTab(panel.getName(), panel);
            }
        }
        this.setVisible(true);
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        mainPanel = new javax.swing.JPanel();
        tabsPane = new javax.swing.JTabbedPane();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);

        org.jdesktop.layout.GroupLayout mainPanelLayout = new
org.jdesktop.layout.GroupLayout(mainPanel);
        mainPanel.setLayout(mainPanelLayout);
        mainPanelLayout.setHorizontalGroup(
            mainPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(tabsPane, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 527,
Short.MAX_VALUE)
        );
        mainPanelLayout.setVerticalGroup(
            mainPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, tabsPane,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 346, Short.MAX_VALUE)
        );

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        pack();
    }// </editor-fold>//GEN-END:initComponents



    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JPanel mainPanel;
    private javax.swing.JTabbedPane tabsPane;
    // End of variables declaration//GEN-END:variables

}
/*
 * Analysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis;

import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.main.GameCardIterator;
import br.ufsc.inf.lagar.main.GameResultIterator;
```

```java
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.ui.panels.AnalysisControlPanel;
import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import javax.swing.JPanel;

public abstract class Analysis {

    private GameConfiguration currentGameConfiguration;
    private int begin;
    private int end;
    private int jump;
    private AnalysisControlPanel.SourceMode sourceMode;

    private GenCardIO genCardIO;

    public GameConfiguration getCurrentGameConfiguration() {
        return currentGameConfiguration;
    }

    public void setCurrentGameConfiguration(GameConfiguration
currentGameConfiguration) {
        this.currentGameConfiguration = currentGameConfiguration;
    }

    public GameCardIterator getGameCardIterator(){
        if(this.sourceMode == sourceMode.FROM_ANALYSIS_DATABASE)
            return this.getCurrentGameConfiguration().getGameCardIterator(begin, end,
jump);

        if(this.sourceMode == sourceMode.FROM_LLG_FILE){
            return this.genCardIO;
        }

        throw new RuntimeException("Error in getGameCardIterator");
    }

    public abstract void calculateAnalysis();

    public abstract List<JPanel> getPanels();

    public abstract String getName();

    public abstract String getHelpInfo();

    public boolean acceptGame(String gameName){
        return true;
    }

    @Override
    public String toString(){
        return this.getName();
    }

    public final void setBegin(int begin) {
        this.begin = begin;
    }

    public final void setEnd(int end) {
        this.end = end;
    }
    public final void setJump(int jump) {
        this.jump = jump;
    }

    public final void setSourceMode(AnalysisControlPanel.SourceMode mode){

        this.sourceMode = mode;
    }

    public final void setGenCardIO(GenCardIO genCardIO) {
        this.genCardIO = genCardIO;
    }


}
/*
 * Animated1Frame.java
 *
 * Created on 20 de Janeiro de 2005, 04:09
 */

/**
 *
 * @author  lagar
 */
package br.ufsc.inf.lagar.verify.animated1;

import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.verify.VerificatorInterface;
import java.io.*;
import java.net.*;
import java.util.List;

public class Animated1Frame extends javax.swing.JFrame {

    private Ball[] matrix;
    private int lastNumber;
    private int currentCard;
    private int hits;
    private int currentDrw;

    private GenCardIO betcards;
    private GameCard result;
    private GameCard buffer;

    /**
     * Creates new form Animated1Frame
     */
    public Animated1Frame(File llgFile) throws IOException {
        this.initComponents();

        this.setLastNumber(1);
        this.currentCard = 0;
        this.hits = 0;
        this.currentDrw = 0;

        this.betcards = new GenCardIO(llgFile);
        this.limitLabel.setText("/ "+(this.getBetcards().size()));
        this.buffer = new GameCard(0,null,this.betcards.getBallsPerCard());
        this.inicializaMatriz();
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        numbersLabel = new javax.swing.JLabel();
        nextButton = new javax.swing.JButton();
        enterButton = new javax.swing.JButton();
        warnLabel = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        currentDrwLabel = new javax.swing.JLabel();
        summaryButton = new javax.swing.JButton();
        jLabel3 = new javax.swing.JLabel();
        hitsLabel = new javax.swing.JLabel();
        actualLabel = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        limitLabel = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Lotomania Cartoes");
        addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosing(java.awt.event.WindowEvent evt) {
                exitForm(evt);
            }
        });

        jPanel1.setLayout(new java.awt.GridLayout(10, 10, 4, 3));

        jPanel1.setAlignmentX(0.0F);
        jPanel1.setAlignmentY(0.0F);
        jPanel1.addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyPressed(java.awt.event.KeyEvent evt) {
                jPanel1KeyPressed(evt);
            }
        });

        jLabel1.setFont(new java.awt.Font("Dialog", 0, 12));
        jLabel1.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
        jLabel1.setText("Marked numbers");

        numbersLabel.setFont(new java.awt.Font("Dialog", 1, 14));
        numbersLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        numbersLabel.setText("0");

        nextButton.setText("Next");
        nextButton.setEnabled(false);
        nextButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                nextButtonActionPerformed(evt);
            }
        });

        enterButton.setText("Enter res");
        enterButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                enterButtonActionPerformed(evt);
            }
        });

        warnLabel.setFont(new java.awt.Font("Dialog", 1, 14));
        warnLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);

        jLabel2.setFont(new java.awt.Font("Dialog", 0, 12));
        jLabel2.setHorizontalAlignment(javax.swing.SwingConstants.LEFT);
        jLabel2.setText("Current =");

        currentDrwLabel.setFont(new java.awt.Font("Dialog", 1, 14));
        currentDrwLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        currentDrwLabel.setText("0");

        summaryButton.setText("Summary");
        summaryButton.setEnabled(false);
        summaryButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                summaryButtonActionPerformed(evt);
            }
        });

        jLabel3.setFont(new java.awt.Font("Dialog", 1, 14));
        jLabel3.setText("HITS =");

        hitsLabel.setFont(new java.awt.Font("Dialog", 0, 18));
        hitsLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        hitsLabel.setText("0");

        actualLabel.setFont(new java.awt.Font("Dialog", 1, 14));
        actualLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        actualLabel.setText("0");

        jLabel4.setFont(new java.awt.Font("Dialog", 0, 12));
        jLabel4.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        jLabel4.setText("curr");

        limitLabel.setFont(new java.awt.Font("Dialog", 1, 14));
        limitLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        limitLabel.setText("/  x");

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(300, 300, 300)
                .add(warnLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 210,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
            .add(layout.createSequentialGroup()
                .add(100, 100, 100)
                .add(jLabel3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 90,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .add(hitsLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 50,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
            .add(layout.createSequentialGroup()
                .add(10, 10, 10)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
                    .add(layout.createSequentialGroup()
                        .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayo
ut.LEADING)
```

```
                            .add(layout.createSequentialGroup()
                                .add(130, 130, 130)
                                .add(numbersLabel,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                            .add(jLabel1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 140,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                        .add(30, 30, 30)
                        .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayo
ut.LEADING)
                            .add(layout.createSequentialGroup()
                                .add(70, 70, 70)
                                .add(currentDrwLabel,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                            .add(jLabel2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 80,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))
                    .add(jPanel1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
370, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .add(20, 20, 20)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
                    .add(enterButton, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
100, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(nextButton, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
100, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(summaryButton,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 100,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(layout.createSequentialGroup()
                        .add(20, 20, 20)
                        .add(jLabel4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
50, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(layout.createSequentialGroup()
                        .add(30, 30, 30)
                        .add(limitLabel,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 50,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(actualLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
50, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(warnLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
                    .add(jLabel3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(hitsLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
40, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
                    .add(layout.createSequentialGroup()
                        .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayo
ut.LEADING)
                            .add(layout.createParallelGroup(org.jdesktop.layout.Group
Layout.LEADING)
                                .add(numbersLabel,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
```

```
                                .add(jLabel1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                            .add(layout.createParallelGroup(org.jdesktop.layout.Group
Layout.LEADING)
                                .add(currentDrwLabel,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                                .add(jLabel2,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))
                        .add(jPanel1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
300, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(layout.createSequentialGroup()
                        .add(20, 20, 20)
                        .add(enterButton)
                        .add(45, 45, 45)
                        .add(nextButton)
                        .add(15, 15, 15)
                        .add(summaryButton)
                        .add(5, 5, 5)
                        .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayo
ut.LEADING)
                            .add(layout.createSequentialGroup()
                                .add(30, 30, 30)
                                .add(jLabel4,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 30,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                            .add(limitLabel,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                            .add(actualLabel,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 40,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)))))
        );
        java.awt.Dimension screenSize =
java.awt.Toolkit.getDefaultToolkit().getScreenSize();
        setBounds((screenSize.width-521)/2, (screenSize.height-455)/2, 521, 455);
    }// </editor-fold>//GEN-END:initComponents

    private void summaryButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_summaryButtonActionPerformed

//        this.ma1.mostraSumario();
//        System.out.println(ma1.buffer);

    }//GEN-LAST:event_summaryButtonActionPerformed

    private void enterButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_enterButtonActionPerformed

      GameConfiguration gc = GameManager.getCurrentGame();

      try{

        if(Ball.countNumbers == gc.getNumbersPerDrawing()){

            this.getWarnLabel().setText("OK!");

            int[] res = new int[gc.getNumbersPerDrawing()];
            int index = 0;

            for(int i = 1; i < this.getMatrix().length; i++){
                if(getMatrix()[i].isState()){

                    res[index] = getMatrix()[i].getNumber();
                    index++;

                }

            }

            this.result = new GameCard(0,null,gc.getNumbersPerDrawing());
            for(int i = 0; i < res.length; ++i){
                this.result.setBall((byte)res[i], i);
            }

            this.getNextButton().setEnabled(true);
            this.summaryButton.setEnabled(true);

            //ma1 = new MostraAcertos(resultado,gerado1);

        }
        else{

         this.getWarnLabel().setText("Less than "+gc.getNumbersPerDrawing());

        }

      }
      catch(Exception ex){

          this.getWarnLabel().setText("Exception!");

      }



    }//GEN-LAST:event_enterButtonActionPerformed

    private void nextButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_nextButtonActionPerformed

        GameConfiguration gc = GameManager.getCurrentGame();

        this.getNextButton().setEnabled(false);
        Ball.countNumbers = 0;

        for(int i = 1; i < this.getMatrix().length; i++){

            this.getMatrix()[i].restart();

        }

        for(int i = 0; i < this.result.getBalls().length; i++){


            this.getMatrix()[this.result.getBalls()[i]].btClick();

        }

        this.setLastNumber(1);
        this.currentDrwLabel.setText("0");
        this.currentDrw = 0;
        this.hits = 0;
        this.getHitsLabel().setText("0");
        Ball.countNumbers = this.getBetcards().getBallsPerCard();
        this.getNumbersLabel().setText(String.valueOf(this.getBetcards().getBallsPerC
ard()));
        this.incrementCurrentCard();
        this.getWarnLabel().setText("");

        if(this.currentCard < this.getBetcards().size()){

           AnimateRes anima = new AnimateRes(this);
           anima.start();

        }
        else{

            this.getWarnLabel().setText("Game over!");

        }

    }//GEN-LAST:event_nextButtonActionPerformed

    private void jPanel1KeyPressed(java.awt.event.KeyEvent evt) {//GEN-
FIRST:event_jPanel1KeyPressed

        String Texto = evt.getKeyText(evt.getKeyCode());

        if ( Texto.equals("Left"))
            {

                try{
                this.getMatrix()[this.getLastNumber()-1].requestFocus();
                this.setLastNumber(this.getLastNumber() - 1);
                }
                catch(Exception ex){

                    this.setLastNumber(this.getLastNumber() + 1);

                }

            }
            else
            {
                if ( Texto.equals("Right"))
                {

                    try{
                    this.getMatrix()[this.getLastNumber()+1].requestFocus();
                    this.setLastNumber(this.getLastNumber() + 1);
                    }
                    catch(Exception ex){

                        this.setLastNumber(this.getLastNumber() - 1);

                    }

                }
                else
                {
                    if ( Texto.equals("Up"))
                    {
                             try{
                            this.getMatrix()[this.getLastNumber()-10].requestFocus();
                            this.setLastNumber(this.getLastNumber() - 10);
                            }
                            catch(Exception ex){
```

```
                                        this.setLastNumber(this.getLastNumber() + 10);

                                    }
                            }
                            else
                            {
                                if ( Texto.equals("Down"))
                                {
                                        try{
                                        this.getMatrix()[this.getLastNumber()+10].requestFocu
s();
                                        this.setLastNumber(this.getLastNumber() + 10);
                                        }
                                        catch(Exception ex){

                                            this.setLastNumber(this.getLastNumber() - 10);

                                        }
                                }
                            }
                        }
                    }
            //}//End if_Condição_GETTEXT


        }//GEN-LAST:event_jPanel1KeyPressed

        public void incrementHits(){

            this.hits++;
            this.getHitsLabel().setText(String.valueOf(this.getHits()));

        }

        public void incrementDrw(){

            this.currentDrw++;
            this.currentDrwLabel.setText(String.valueOf(this.getCurrentDrw()));

        }

        private void incrementCurrentCard(){

            this.currentCard++;
            this.actualLabel.setText(String.valueOf(this.currentCard));

        }

        public byte[] getCurrentBalls(){
            try {

                this.betcards.skipToCard(this.buffer, this.currentCard);
            } catch (IOException ex) {
                ex.printStackTrace();
                throw new RuntimeException(ex);
            }
            return this.buffer.getBalls();
        }


        private void inicializaMatriz(){

            int bs = GameManager.getCurrentGame().getNumbersPerCard();
            this.matrix = new Ball[bs + 1];

            for(int i = 1; i < this.getMatrix().length; i++){

                this.getMatrix()[i] = new Ball(i, this);
                this.getJPanel1().add(this.getMatrix()[i]);

            }

        }

        public Ball[] getMatrix() {
            return matrix;
        }

        public GenCardIO getBetcards() {
            return betcards;
        }

        public javax.swing.JLabel getHitsLabel() {
            return hitsLabel;
        }

        public javax.swing.JButton getNextButton() {
            return nextButton;
        }

        public javax.swing.JLabel getWarnLabel() {
            return warnLabel;
        }

        public int getCurrentDrw() {
            return currentDrw;
        }

        public javax.swing.JPanel getJPanel1() {
            return jPanel1;
        }

        public javax.swing.JLabel getNumbersLabel() {
            return numbersLabel;
        }

        public int getLastNumber() {
            return lastNumber;
        }

        public void setLastNumber(int lastNumber) {
            this.lastNumber = lastNumber;
        }

        /** Exit the Application */
        private void exitForm(java.awt.event.WindowEvent evt) {//GEN-FIRST:event_exitForm
            //this.dispose();
        }//GEN-LAST:event_exitForm


    public static void main(String[] args) throws IOException{

        List<GameConfiguration> availableGames = GameManager.getAvailableGames();
            GameConfiguration gc = availableGames.get(0);
            gc.getGameResults().loadResults();
            GameManager.setCurrentGame(gc);

        File f = new File("/home/lagar/teste/1ateh10.llg");
        new Animated1Frame(f).setVisible(true);
    }

        public int getHits() {
            return hits;
        }


        // Variables declaration - do not modify//GEN-BEGIN:variables
        private javax.swing.JLabel actualLabel;
        private javax.swing.JLabel currentDrwLabel;
        private javax.swing.JButton enterButton;
        private javax.swing.JLabel hitsLabel;
        private javax.swing.JLabel jLabel1;
        private javax.swing.JLabel jLabel2;
        private javax.swing.JLabel jLabel3;
        private javax.swing.JLabel jLabel4;
        private javax.swing.JPanel jPanel1;
        private javax.swing.JLabel limitLabel;
        private javax.swing.JButton nextButton;
        private javax.swing.JLabel numbersLabel;
        private javax.swing.JButton summaryButton;
        private javax.swing.JLabel warnLabel;
        // End of variables declaration//GEN-END:variables

    }
    /*
     * AnimateRes.java
     *
     * Created on 3 de Novembro de 2004, 03:19
     */

    /**
     *
     * @author  lagar
     */
    package br.ufsc.inf.lagar.verify.animated1;

    import br.ufsc.inf.lagar.main.GameManager;
    import java.awt.*;
    import java.io.IOException;
    import javax.swing.*;
    import java.util.*;


    public class AnimateRes extends Thread{

        Animated1Frame animatedFrame;
        private int delay = 0;
        int emotion;
        private Beeps beep;

        /**
         * Creates a new instance of AnimateRes
         */
        public AnimateRes(Animated1Frame af) {

           this.animatedFrame = af;
           this.emotion = GameManager.getCurrentGame().getNumbersPerDrawing() - 3;
            try {
                this.beep = new Beeps();
            } catch (IOException ex) {
                ex.printStackTrace();
            }
        }

        public void run(){

            Ball[] matrix = this.animatedFrame.getMatrix();
            byte[] oRes = animatedFrame.getCurrentBalls();

            Integer[] resres = new Integer[oRes.length];

            for(int i = 0; i < resres.length; i++)
                resres[i] = Integer.valueOf(oRes[i]);

            Collections.shuffle(Arrays.asList(resres));

            for(int j = 0; j < resres.length; j++){

                   int num = resres[j].intValue();

                   if(matrix[num].isState()){

                       this.beep.beep();
                       matrix[num].blue();
                       this.sleepUtil(50);
                       matrix[num].red();
                       this.beep.goodBeep();

                       this.animatedFrame.incrementHits();
                                       //this.sleepUtil(150);

                   }
                   else{

                       this.beep.beep();
                       matrix[num].blue();

                   }

                   this.animatedFrame.incrementDrw();

                   if(animatedFrame.getHits() >= emotion && (j != resres.length-1)){

                       animatedFrame.getWarnLabel().setText("Almost there!!");
                       this.sleepUtil(this.getDelay()+2400);


                   }
                   else{

                       this.sleepUtil(this.getDelay());
                       animatedFrame.getWarnLabel().setText("");

                   }


               }


            this.animatedFrame.getNextButton().setEnabled(true);
        }


        private void sleepUtil(int time){
        if(time == 0) return;
          try{
            Thread.sleep(time);
          }
          catch(Exception ioex){System.out.println(ioex.getMessage());}

        }

        public int getDelay() {
            return delay;
        }

        public void setDelay(int delay) {
            this.delay = delay;
        }

    }

    package br.ufsc.inf.lagar.verify.animated1.osc;

    /*
     *      AudioCommon.java
     *
     *      This file is part of jsresources.org
     */

    /*
     * Copyright (c) 1999 - 2001 by Matthias Pfisterer
     * All rights reserved.
     *
     * Redistribution and use in source and binary forms, with or without
     * modification, are permitted provided that the following conditions
     * are met:
     *
     * - Redistributions of source code must retain the above copyright notice,
     *   this list of conditions and the following disclaimer.
     * - Redistributions in binary form must reproduce the above copyright
     *   notice, this list of conditions and the following disclaimer in the
     *   documentation and/or other materials provided with the distribution.
     *
     * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
     * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
     * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
     * FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
     * COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
     * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
     * (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
     * SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
     * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
     * STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
     * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
     * OF THE POSSIBILITY OF SUCH DAMAGE.
     */

    /*
    |<---            this code is formatted to fit into 80 columns             --->|
    */

    import javax.sound.sampled.AudioFileFormat;
    import javax.sound.sampled.AudioFormat;
    import javax.sound.sampled.AudioSystem;
    import javax.sound.sampled.DataLine;
    import javax.sound.sampled.Line;
    import javax.sound.sampled.LineUnavailableException;
    import javax.sound.sampled.Mixer;
    import javax.sound.sampled.SourceDataLine;
    import javax.sound.sampled.TargetDataLine;



    /** Common methods for audio examples.
     */
    public class AudioCommon
    {
    	private static boolean		DEBUG = false;



    	public static void setDebug(boolean bDebug)
    	{
    		DEBUG = bDebug;
    	}



    	/** TODO:
    	 */
    	public static void listSupportedTargetTypes()
    	{
    		String	strMessage = "Supported target types:";
    		AudioFileFormat.Type[]	aTypes = AudioSystem.getAudioFileTypes();
    		for (int i = 0; i < aTypes.length; i++)
    		{
    			strMessage += " " + aTypes[i].getExtension();
    		}
    		out(strMessage);
    	}



    	/**	Trying to get an audio file type for the passed extension.
    		This works by examining all available file types. For each
    		type, if the extension this type promisses to handle matches
    		the extension we are trying to find a type for, this type is
    		returned.
    		If no appropriate type is found, null is returned.
    	*/
    	public static AudioFileFormat.Type findTargetType(String strExtension)
    	{
    		AudioFileFormat.Type[]	aTypes = AudioSystem.getAudioFileTypes();
    		for (int i = 0; i < aTypes.length; i++)
    		{
    			if (aTypes[i].getExtension().equals(strExtension))
    			{
    				return aTypes[i];
    			}
    		}
    		return null;
    	}
    	/** TODO:
    	 */
    	public static void listMixersAndExit()
    	{
    		out("Available Mixers:");
    		Mixer.Info[]	aInfos = AudioSystem.getMixerInfo();
    		for (int i = 0; i < aInfos.length; i++)
    		{
    			out(aInfos[i].getName());
    		}
    		if (aInfos.length == 0)
    		{
    			out("[No mixers available]");
    		}
    		System.exit(0);
    	}



    	/** List Mixers.
    		Only Mixers that support either TargetDataLines or SourceDataLines
    		are listed, depending on the value of bPlayback.
    	 */
    	public static void listMixersAndExit(boolean bPlayback)
    	{
    		out("Available Mixers:");
    		Mixer.Info[]	aInfos = AudioSystem.getMixerInfo();
    		for (int i = 0; i < aInfos.length; i++)
    		{
    			Mixer mixer = AudioSystem.getMixer(aInfos[i]);
    			Line.Info lineInfo = new Line.Info(bPlayback ?

    SourceDataLine.class :

    TargetDataLine.class);
    			if (mixer.isLineSupported(lineInfo))
    			{
    				out(aInfos[i].getName());
    			}
    		}
    		if (aInfos.length == 0)
    		{
    			out("[No mixers available]");
    		}
    		System.exit(0);
    	}



    	/**	TODO:
    		This method tries to return a Mixer.Info whose name
    		matches the passed name. If no matching Mixer.Info is
    		found, null is returned.
    	*/
    	public static Mixer.Info getMixerInfo(String strMixerName)
    	{
    		Mixer.Info[]	aInfos = AudioSystem.getMixerInfo();
    		for (int i = 0; i < aInfos.length; i++)
    		{
    			if (aInfos[i].getName().equals(strMixerName))
    			{
    				return aInfos[i];
    			}
```

```
                }
                return null;
        }



        /** TODO:
         */
        public static TargetDataLine getTargetDataLine(String strMixerName,
                                                        AudioFormat audioFormat,
                                                        int nBufferSize)
        {
                /*
                        Asking for a line is a rather tricky thing.
                        We have to construct an Info object that specifies
                        the desired properties for the line.
                        First, we have to say which kind of line we want. The
                        possibilities are: SourceDataLine (for playback), Clip
                        (for repeated playback) and TargetDataLine (for
                         recording).
                        Here, we want to do normal capture, so we ask for
                        a TargetDataLine.
                        Then, we have to pass an AudioFormat object, so that
                        the Line knows which format the data passed to it
                        will have.
                        Furthermore, we can give Java Sound a hint about how
                        big the internal buffer for the line should be. This
                        isn't used here, signaling that we
                        don't care about the exact size. Java Sound will use
                        some default value for the buffer size.
                */
                TargetDataLine  targetDataLine = null;
                DataLine.Info   info = new DataLine.Info(TargetDataLine.class,
                                                         audioFormat, nBufferSize);
                try
                {
                        if (strMixerName != null)
                        {
                                Mixer.Info      mixerInfo =
getMixerInfo(strMixerName);
                                if (mixerInfo == null)
                                {
                                        out("AudioCommon.getTargetDataLine(): mixer
not found: " + strMixerName);
                                        return null;
                                }
                                Mixer   mixer = AudioSystem.getMixer(mixerInfo);
                                targetDataLine = (TargetDataLine)
mixer.getLine(info);
                        }
                        else
                        {
                                if (DEBUG) { out("AudioCommon.getTargetDataLine():
using default mixer"); }
                                targetDataLine = (TargetDataLine)
AudioSystem.getLine(info);
                        }

                        /*
                         *      The line is there, but it is not yet ready to
                         *      receive audio data. We have to open the line.
                         */
                        if (DEBUG) { out("AudioCommon.getTargetDataLine(): opening
line..."); }
```

```
                        targetDataLine.open(audioFormat, nBufferSize);
                        if (DEBUG) { out("AudioCommon.getTargetDataLine(): opened
line"); }
                }
                catch (LineUnavailableException e)
                {
                        if (DEBUG) { e.printStackTrace(); }
                }
                catch (Exception e)
                {
                        if (DEBUG) { e.printStackTrace(); }
                }
                        if (DEBUG) { out("AudioCommon.getTargetDataLine(): returning
line: " + targetDataLine); }
                return targetDataLine;
        }


        /** Checks if the encoding is PCM.
         */
        public static boolean isPcm(AudioFormat.Encoding encoding)
        {
                return encoding.equals(AudioFormat.Encoding.PCM_SIGNED)
                        || encoding.equals(AudioFormat.Encoding.PCM_UNSIGNED);
        }


        /** TODO:
         */
        private static void out(String strMessage)
        {
                System.out.println(strMessage);
        }



}



/*** AudioCommon.java ***/

package br.ufsc.inf.lagar.verify.animated1.osc;

/*
 *      AudioPlayer.java
 *
 *      This file is part of jsresources.org
 */

/*
 * Copyright (c) 1999, 2000 by Matthias Pfisterer
 * All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *
 * - Redistributions of source code must retain the above copyright notice,
 *   this list of conditions and the following disclaimer.
 * - Redistributions in binary form must reproduce the above copyright
 *   notice, this list of conditions and the following disclaimer in the
 *   documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
 * FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
 * COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
 * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
 * (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
 * SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
 * STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
 * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
 * OF THE POSSIBILITY OF SUCH DAMAGE.
 */

/*
|<---            this code is formatted to fit into 80 columns             --->|
*/

import java.io.BufferedInputStream;
import java.io.File;
import java.io.InputStream;
import java.io.IOException;

import java.net.URL;

import javax.sound.sampled.AudioFormat;
import javax.sound.sampled.AudioInputStream;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.DataLine;
import javax.sound.sampled.LineUnavailableException;
import javax.sound.sampled.Mixer;
import javax.sound.sampled.SourceDataLine;

/*      If the compilation fails because this class is not available,
        get gnu.getopt from the URL given in the comment below.
*/
//import gnu.getopt.Getopt;



/**     <titleabbrev>AudioPlayer</titleabbrev>
        <title>Playing an audio file (advanced)</title>

        <formalpara><title>Purpose</title>
        <para>
        Plays a single audio file. Capable of playing some
        compressed audio formats (A-law, &mu;-law, maybe ogg vorbis, mp3,
        GSM06.10).
        Allows control over buffering
        and which mixer to use.
        </para></formalpara>

        <formalpara><title>Usage</title>
        <para>
        <cmdsynopsis>
        <command>java AudioPlayer</command>
        <arg choice="plain"><option>-l</option></arg>
        </cmdsynopsis>
        <cmdsynopsis>
        <command>java AudioPlayer</command>
        <arg><option>-M <replaceable>mixername</replaceable></option></arg>
        <arg><option>-e <replaceable>buffersize</replaceable></option></arg>
        <arg><option>-i <replaceable>buffersize</replaceable></option></arg>
        <arg choice="plain"><replaceable>audiofile</replaceable></arg>
        </cmdsynopsis>
        </para></formalpara>

        <formalpara><title>Parameters</title>
        <variablelist>
        <varlistentry>
        <term><option>-h</option></term>
        <listitem><para>print usage message</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option>-l</option></term>
        <listitem><para>lists the available mixers</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option>-M <replaceable>mixername</replaceable></option></term>
        <listitem><para>selects a mixer to play on</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option>-e <replaceable>buffersize</replaceable></option></term>
        <listitem><para>the buffer size to use in the application
("extern")</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option>-i <replaceable>buffersize</replaceable></option></term>
        <listitem><para>the buffer size to use in Java Sound
("intern")</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option>-E <replaceable>endianess</replaceable></option></term>
        <listitem><para>the endianess ("big" or "little") to use in conversions. The
default is little. Specifying this option forces a conversion, even if the audio
format is supported by SourceDataLines directly.</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option>-S <replaceable>sample size</replaceable></option></term>
        <listitem><para>the sample size in bits to use in conversions. The default
is 16. Specifying this option forces a conversion, even if the audio format is
supported by SourceDataLines directly.</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option>-D</option></term>
        <listitem><para>enable debugging output</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option>-f</option></term>
        <listitem><para>interpret filename arguments as filenames. This is the
default. This option is exclusive to <option>-u</option>.</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option>-u</option></term>
        <listitem><para>interpret filename arguments as URLs. The default is to
interpret them as filenames. This option is exclusive to <option>-
f</option>.</para></listitem>
        </varlistentry>
        <varlistentry>
        <term><option><replaceable>audiofile</replaceable></option></term>
        <listitem><para>the file name of the audio file to play</para></listitem>
        </varlistentry>
        </variablelist>
        </formalpara>

        <formalpara><title>Bugs, limitations</title>
        <para>
        Compressed formats can be handled depending on the
        capabilities of the Java Sound implementation.
        A-law and &mu;-law can be handled in any known Java Sound implementation.
        Ogg vorbis, mp3 and GSM 06.10 can be handled by Tritonus. If you want to play
these
        formats with the Sun jdk1.3/1.4, you have to install the respective plug-ins
        from <ulink url
        ="http://www.tritonus.org/plugins.html">Tritonus Plug-ins</ulink>.
        </para>
        </formalpara>

        <formalpara><title>Source code</title>
        <para>
        <ulink url="AudioPlayer.java.html">AudioPlayer.java</ulink>,
        <ulink url="AudioLoop.java.html">AudioLoop.java</ulink>,
        <olink targetdocent="getopt">gnu.getopt.Getopt</olink>
        </para>
        </formalpara>

*/
public class AudioPlayer
{
        /**     Flag for debugging messages.
         *      If true, some messages are dumped to the console
         *      during operation.
         */
        private static boolean  DEBUG = false;

        private static int      DEFAULT_EXTERNAL_BUFFER_SIZE = 128000;



        public static void main(InputStream inputStream)
                throws Exception
        {
                /** Determines if command line arguments are intereted as URL.
                    If true, filename arguments on the command line are
                    interpreted as URL. If false, they are interpreted as
                    filenames. This flag is set by the command line
                    option "-u". It is reset by the command line option
                    "-f".
                */
                boolean bInterpretFilenameAsUrl = false;

                /** Flag for forcing a conversion.
                    If set to true, a conversion of the AudioInputStream
                    (AudioSystem.getAudioInputStream(..., AudioInputStream))
                    is done even if the format of the original AudioInputStream
                    would be supported for SourceDataLines directly. This
                    flag is set by the command line options "-E" and "-S".
                */
                boolean bForceConversion = false;

                /** Endianess value to use in conversion.
                    If a conversion of the AudioInputStream is done,
                    this values is used as endianess in the target AudioFormat.
                    The default value can be altered by the command line
                    option "-B".
                */
                boolean bBigEndian = false;

                /** Sample size value to use in conversion.
```

```
                        If a conversion of the AudioInputStream is done,
                        this values is used as sample size in the target
                        AudioFormat.
                        The default value can be altered by the command line
                        option "-S".
                    */
                    int     nSampleSizeInBits = 16;


                    String  strMixerName = null;

                    int     nExternalBufferSize = DEFAULT_EXTERNAL_BUFFER_SIZE;

                    int     nInternalBufferSize = AudioSystem.NOT_SPECIFIED;



                    /*
                     *      Parsing of command-line options takes place...
                     */
//                  Getopt  g = new Getopt("AudioPlayer", args, "hlufM:e:i:E:S:D");
//                  int     c;
//                  while ((c = g.getopt()) != -1)
//                  {
//                          switch (c)
//                          {
//                          case 'h':
//                                  printUsageAndExit();
//
//                          case 'l':
//                                  AudioCommon.listMixersAndExit(true);
//
//                          case 'u':
//                                  bInterpretFilenameAsUrl = true;
//                                  break;
//
//                          case 'f':
//                                  bInterpretFilenameAsUrl = false;
//                                  break;
//
//                          case 'M':
//                                  strMixerName = g.getOptarg();
//                                  if (DEBUG) out("AudioPlayer.main(): mixer name: " +
strMixerName);
//                                  break;
//
//                          case 'e':
//                                  nExternalBufferSize =
Integer.parseInt(g.getOptarg());
//                                  break;
//
//                          case 'i':
//                                  nInternalBufferSize =
Integer.parseInt(g.getOptarg());
//                                  break;
//
//                          case 'E':
//                                  String strEndianess = g.getOptarg();
//                                  strEndianess = strEndianess.toLowerCase();
//                                  if (strEndianess.equals("big"))
//                                  {
//                                          bBigEndian = true;
//                                  }
//                                  else if (strEndianess.equals("little"))
//                              {
//                                      bBigEndian = false;
//                              }
//                              else
//                              {
//                                      printUsageAndExit();
//                              }
//                              bForceConversion = true;
//                              break;
//
//                      case 'S':
//                              nSampleSizeInBits = Integer.parseInt(g.getOptarg());
//                              bForceConversion = true;
//                              break;
//
//                      case 'D':
//                              DEBUG = true;
//                              break;
//
//                      case '?':
//                              printUsageAndExit();
//
//                      default:
//                              out("getopt() returned " + c);
//                              break;
//                      }
//              }

                /* We make shure that there is only one more argument, which
                   we take as the filename of the soundfile we want to play. */
//              String  strFilenameOrUrl = null;
//              for (int i = g.getOptind(); i < args.length; i++)
//              {
//                      if (strFilenameOrUrl == null)
//                      {
//                              strFilenameOrUrl = args[i];
//                      }
//                      else
//                      {
//                              printUsageAndExit();
//                      }
//              }
//              if (strFilenameOrUrl == null)
//              {
//                      printUsageAndExit();
//              }

                AudioInputStream audioInputStream = null;
                if (true || bInterpretFilenameAsUrl)
                {
//                      URL url = new URL(strFilenameOrUrl);
                        audioInputStream = 
AudioSystem.getAudioInputStream(inputStream);
                }
                else
                {
//                      // Are we requested to use standard input?
//                      if (strFilenameOrUrl.equals("-"))
//                      {
//                              InputStream inputStream = new 
BufferedInputStream(System.in);
//                              audioInputStream = 
AudioSystem.getAudioInputStream(inputStream);
//                      }
//                      else
//                      {
//                              File file = new File(strFilenameOrUrl);
//                              audioInputStream =
AudioSystem.getAudioInputStream(file);
//                      }
                }

                if (DEBUG) out("AudioPlayer.main(): primary AIS: " +
audioInputStream);

                /*
                 *      From the AudioInputStream, i.e. from the sound file,
                 *      we fetch information about the format of the
                 *      audio data.
                 *      These information include the sampling frequency,
                 *      the number of
                 *      channels and the size of the samples.
                 *      These information
                 *      are needed to ask Java Sound for a suitable output line
                 *      for this audio stream.
                 */
                AudioFormat     audioFormat = audioInputStream.getFormat();
                if (DEBUG) out("AudioPlayer.main(): primary format: " + audioFormat);
                DataLine.Info   info = new DataLine.Info(SourceDataLine.class,
                                                         audioFormat,
nInternalBufferSize);
                boolean bIsSupportedDirectly = AudioSystem.isLineSupported(info);
                if (!bIsSupportedDirectly || bForceConversion)
                {
                        AudioFormat     sourceFormat = audioFormat;
                        AudioFormat     targetFormat = new AudioFormat(
                                AudioFormat.Encoding.PCM_SIGNED,
                                sourceFormat.getSampleRate(),
                                nSampleSizeInBits,
                                sourceFormat.getChannels(),
                                sourceFormat.getChannels() * (nSampleSizeInBits / 8),
                                sourceFormat.getSampleRate(),
                                bBigEndian);
                        if (DEBUG)
                        {
                                out("AudioPlayer.main(): source format: " +
sourceFormat);
                                out("AudioPlayer.main(): target format: " +
targetFormat);
                        }
                        audioInputStream =
AudioSystem.getAudioInputStream(targetFormat, audioInputStream);
                        audioFormat = audioInputStream.getFormat();
                        if (DEBUG) out("AudioPlayer.main(): converted AIS: " +
audioInputStream);
                        if (DEBUG) out("AudioPlayer.main(): converted format: " +
audioFormat);
                }

                SourceDataLine  line = getSourceDataLine(strMixerName, audioFormat,
nInternalBufferSize);
                if (line == null)
                {
                        out("AudioPlayer: cannot get SourceDataLine for format " +
audioFormat);
                        System.exit(1);
                }
                if (DEBUG) out("AudioPlayer.main(): line: " + line);
                if (DEBUG) out("AudioPlayer.main(): line format: " +
line.getFormat());


                /*
                 *      Still not enough. The line now can receive data,
                 *      but will not pass them on to the audio output device
                 *      (which means to your sound card). This has to be
                 *      activated.
                 */
                line.start();

                /*
                 *      Ok, finally the line is prepared. Now comes the real
                 *      job: we have to write data to the line. We do this
                 *      in a loop. First, we read data from the
                 *      AudioInputStream to a buffer. Then, we write from
                 *      this buffer to the Line. This is done until the end
                 *      of the file is reached, which is detected by a
                 *      return value of -1 from the read method of the
                 *      AudioInputStream.
                 */
                int     nBytesRead = 0;
                byte[]  abData = new byte[nExternalBufferSize];
                if (DEBUG) out("AudioPlayer.main(): starting main loop");
                while (nBytesRead != -1)
                {
                        try
                        {
                                nBytesRead = audioInputStream.read(abData, 0,
abData.length);
                        }
                        catch (IOException e)
                        {
                                e.printStackTrace();
                        }
                        if (DEBUG) out("AudioPlayer.main(): read from
AudioInputStream (bytes): " + nBytesRead);
                        if (nBytesRead >= 0)
                        {
                                int     nBytesWritten = line.write(abData, 0,
nBytesRead);
                                if (DEBUG) out("AudioPlayer.main(): written to
SourceDataLine (bytes): " + nBytesWritten);
                        }
                }

                if (DEBUG) out("AudioPlayer.main(): finished main loop");

                /*
                 *      Wait until all data is played.
                 *      This is only necessary because of the bug noted below.
                 *      (If we do not wait, we would interrupt the playback by
                 *      prematurely closing the line and exiting the VM.)
                 *
                 *      Thanks to Margie Fitch for bringing me on the right
                 *      path to this solution.
                 */
                if (DEBUG) out("AudioPlayer.main(): before drain");
                line.drain();

                /*
                 *      All data are played. We can close the shop.
                 */
                if (DEBUG) out("AudioPlayer.main(): before close");
                line.close();

//              /*
//               *      There is a bug in the Sun jdk1.3/1.4.
//               *      It prevents correct termination of the VM.
//               *      So we have to exit ourselves.
//               *      This bug has been fixed for the Sun JDK1.5.0.
//               */
//              if (DEBUG)
//              {
//                      out("AudioPlayer.main(): before exit");
//              }
//              System.exit(0);
        }


        // TODO: maybe can used by others. AudioLoop?
        // In this case, move to AudioCommon.
        private static SourceDataLine getSourceDataLine(String strMixerName,
                                                        AudioFormat audioFormat,
                                                        int nBufferSize)
        {
                /*
                 *      Asking for a line is a rather tricky thing.
                 *      We have to construct an Info object that specifies
                 *      the desired properties for the line.
                 *      First, we have to say which kind of line we want. The
                 *      possibilities are: SourceDataLine (for playback), Clip
                 *      (for repeated playback) and TargetDataLine (for
                 *       recording).
                 *      Here, we want to do normal playback, so we ask for
                 *      a SourceDataLine.
                 *      Then, we have to pass an AudioFormat object, so that
                 *      the Line knows which format the data passed to it
                 *      will have.
                 *      Furthermore, we can give Java Sound a hint about how
                 *      big the internal buffer for the line should be. This
                 *      isn't used here, signaling that we
                 *      don't care about the exact size. Java Sound will use
                 *      some default value for the buffer size.
                 */
                SourceDataLine  line = null;
                DataLine.Info   info = new DataLine.Info(SourceDataLine.class,
                                                         audioFormat, nBufferSize);
                try
                {
                        if (strMixerName != null)
                        {
                                Mixer.Info      mixerInfo =
AudioCommon.getMixerInfo(strMixerName);
                                if (mixerInfo == null)
                                {
                                        out("AudioPlayer: mixer not found: " +
strMixerName);

                                        System.exit(1);
                                }
                                Mixer   mixer = AudioSystem.getMixer(mixerInfo);
                                line = (SourceDataLine) mixer.getLine(info);
                        }
                        else
                        {
                                line = (SourceDataLine) AudioSystem.getLine(info);
                        }
                        /*
                         *      The line is there, but it is not yet ready to
                         *      receive audio data. We have to open the line.
                         */
                        line.open(audioFormat, nBufferSize);
                }
                catch (LineUnavailableException e)
                {
                        if (DEBUG) e.printStackTrace();
                }
                catch (Exception e)
                {
                        if (DEBUG) e.printStackTrace();
                }
                return line;
        }



        private static void printUsageAndExit()
        {
                out("AudioPlayer: usage:");
                out("\tjava AudioPlayer -h");
                out("\tjava AudioPlayer -l");
                out("\tjava AudioPlayer");
                out("\t\t[-M <mixername>]");
                out("\t\t[-e <externalBuffersize>]");
                out("\t\t[-i <internalBuffersize>]");
                out("\t\t[-S <SampleSizeInBits>]");
                out("\t\t[-B (big | little)]");
                out("\t\t[-D]");
                out("\t\t[-u | -f]");
                out("\t\t<soundfileOrUrl>");
                System.exit(1);
        }



        private static void out(String strMessage)
        {
                System.out.println(strMessage);
        }
}



/*** AudioPlayer.java ***/
/*
 * BallFilter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter.gamefilter;

import br.ufsc.inf.lagar.filter.AbstractGameFilter;
import br.ufsc.inf.lagar.filter.Filterable;
import br.ufsc.inf.lagar.filter.TextInputFilterException;
import br.ufsc.inf.lagar.main.GameCard;
import java.util.Arrays;

public class BallFilter extends AbstractGameFilter{

    int[] inputBalls;
    /** Creates a new instance of BallFilter */
    public BallFilter() {
    }

    public void validateInput(String input) throws TextInputFilterException {

        String[] params = input.split(Filterable.OR_SEPARATOR);
        this.inputBalls = new int[params.length];

        for(int i = 0; i < params.length; ++i){

           String s = params[i];

           try{
                this.inputBalls[i] = Integer.parseInt(s.trim());
            }
            catch(NumberFormatException nfex){
                throw new TextInputFilterException(nfex);
            }

        }

        Arrays.sort(this.inputBalls);

    }

    public String getShortName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Ball_filter");
    }

    public String getFullName() {
        return getShortName();
    }

    public String getDescription() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Ball_filter_description");
    }

    public String getFilterHelp() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Ball_filter_help");
```

```
    }

    public boolean filter(GameCard card) {

        byte[] balls = card.getBalls();
        for(int b: balls){
            if(Arrays.binarySearch(this.inputBalls, b) >= 0){
                return this.isMode();
            }
        }

        return !this.isMode();
    }

}
/*
 * BallFrequencyAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis.gameanalysis;

import br.ufsc.inf.lagar.analysis.CellData;
import br.ufsc.inf.lagar.analysis.DefaultTableChartAnalysis;
import br.ufsc.inf.lagar.main.GameCard;
import java.awt.Color;
import java.util.ArrayList;
import java.util.List;

public class BallFrequencyAnalysis extends DefaultTableChartAnalysis{

    /** Creates a new instance of BallFrequencyAnalysis */
    public BallFrequencyAnalysis() {
    }

    protected void fillTableChart() {

        GameCard cardBuffer = new GameCard();

        for(int index = 0; this.gci.hasNext(); ++index){
            this.gci.next(cardBuffer);
            byte[] balls = cardBuffer.getBalls();

            for(int i = 0; i < balls.length; ++i){
                int ball = balls[i];

                String category = String.valueOf(ball);
```

```
                    this.tcols.get(category)[index] = this.hit;
                    Integer dc = this.dataChart.get(category);
                    this.dataChart.put(category, ++dc);
                }

                this.tcols.get(this.columnOne)[index] = new
CellData(String.valueOf(cardBuffer.getSerial()), null, Color.BLACK, false);

            }
        }

    protected List<String> buildCategories() {

        List<String> result = new ArrayList<String>();

        int maxNum = getCurrentGameConfiguration().getNumbersPerCard();
        for(int i = 1; i <= maxNum; ++i){
            result.add(String.valueOf(i));
        }
        return result;
    }

    public String getName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Ball_frequency_analysis");
    }

    public String getHelpInfo() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Ball_frequency_analysis_help_info");
    }

}
/*
 * Ball.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.utils;

import br.ufsc.inf.lagar.main.GameManager;
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
```

```
public class Ball extends JButton implements ActionListener{


  private int number;
  private boolean state;

  public static int countNumbers = 0;

  private int maxBalls;

  public Ball(int number, KeyListener k){

    this.setPreferredSize(new Dimension(15,15));
    this.addKeyListener(k);
    //this.resetKeyboardActions();
    //this.setFocusPainted(false);

    this.setBorder(null);
    this.setBackground(Color.white);
    this.addActionListener(this);

    this.number = number;
    this.state = false;

    this.setText(String.valueOf(number));

    this.maxBalls = GameManager.getCurrentGame().getNumbersPerCard();

  }

  /**
     * Getter for property state.
     *
     * @return Value of property state.
     */
  public boolean isState() {
      return state;
  }

  public void restart(){

      this.state = false;
      this.setBackground(Color.white);

  }

  public void blue(){

      this.setBackground(Color.blue);

  }

  public void red(){

      this.setBackground(Color.red);

  }

  /**
     * Setter for property state.
     *
     * @param state New value of property state.
     */
  public void btClick() {

    if(Ball.countNumbers < this.maxBalls || this.state){

      if(this.state){
          this.setBackground(Color.white);
          this.state = false;
          Ball.countNumbers--;
      }
      else{
          this.setBackground(Color.green);
          this.state = true;
          Ball.countNumbers++;
      }


    }


  }

  /**
     * Getter for property number.
     *
     * @return Value of property number.
     */
  public int getNumber() {
      return number;
  }

  public void actionPerformed(java.awt.event.ActionEvent actionEvent) {

      this.btClick();

  }

}//Button/*
 * BallMatrixPanel.java
 *
 * Created on 18 de Abril de 2007, 21:33
 */

package br.ufsc.inf.lagar.utils;

import br.ufsc.inf.lagar.main.GameManager;
import java.awt.GridLayout;

/**
 *
 * @author  lagar
 */
public class BallMatrixPanel extends javax.swing.JPanel {

    private Ball[] matrix;

    /** Creates new form BallMatrixPanel */
    public BallMatrixPanel() {
        initComponents();
    }

    public void initMatrix(int nums, int rows, int cols){
        GridLayout gl = (GridLayout) this.getLayout();
        gl.setColumns(cols);
        gl.setRows(rows);
        gl.setHgap(4);
        gl.setVgap(3);

        this.inicializaMatriz();
    }

    private void inicializaMatriz(){

        int bs = GameManager.getCurrentGame().getNumbersPerCard();

        this.matrix = new Ball[bs + 1];

        for(int i = 1; i < this.getMatrix().length; i++){

            this.getMatrix()[i] = new Ball(i, this.getKeyListeners()[0]);
            this.add(this.getMatrix()[i]);

        }

    }

    public Ball[] getMatrix() {
        return matrix;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {

        setLayout(new java.awt.GridLayout());

        addKeyListener(new java.awt.event.KeyAdapter() {
            public void keyPressed(java.awt.event.KeyEvent evt) {
                formKeyPressed(evt);
            }
        });

    }// </editor-fold>//GEN-END:initComponents

    private void formKeyPressed(java.awt.event.KeyEvent evt) {//GEN-
FIRST:event_formKeyPressed
// TODO add your handling code here:
    }//GEN-LAST:event_formKeyPressed


    // Variables declaration - do not modify//GEN-BEGIN:variables
    // End of variables declaration//GEN-END:variables

}
/*
 * Beeps.java
 *
 * Created on 4 de Novembro de 2004, 02:11
 */

/**
```

```
 *
 * @author  lagar
 */
package br.ufsc.inf.lagar.verify.animated1;

import br.ufsc.inf.lagar.verify.animated1.osc.AudioPlayer;
import java.io.*;
import java.net.*;


public class Beeps {

    byte[] beep;
    byte[] goodBeep;
    byte[] badBeep;
    byte[] greatBeep;

    /** Creates a new instance of Beeps */
    public Beeps() throws IOException{

        InputStream is = getClass().getResourceAsStream("/resources/snd/miss.au");
        ByteArrayOutputStream bos = new ByteArrayOutputStream();
        byte[] buf = new byte[8192];
        int n = 0;
        while((n = is.read(buf)) != -1){
            bos.write(buf, 0, n);
        }

        this.beep = bos.toByteArray();

        is = getClass().getResourceAsStream("/resources/snd/match.au");
        bos = new ByteArrayOutputStream();
        n = 0;
        while((n = is.read(buf)) != -1){
            bos.write(buf, 0, n);
        }

        this.goodBeep = bos.toByteArray();

        is = getClass().getResourceAsStream("/resources/snd/win2.au");
        bos = new ByteArrayOutputStream();
        n = 0;
        while((n = is.read(buf)) != -1){
            bos.write(buf, 0, n);
        }

        this.greatBeep = bos.toByteArray();

    }

    public void beep() {

        ByteArrayInputStream b = new ByteArrayInputStream(this.beep);
        try {
            AudioPlayer.main(b);
        } catch (Exception ex) {
            ex.printStackTrace();
        }
    }


    public void goodBeep() {
        ByteArrayInputStream b = new ByteArrayInputStream(this.goodBeep);
        try {
            AudioPlayer.main(b);
        } catch (Exception ex) {
            ex.printStackTrace();
        }

    }

    public void greatBeep() {

        ByteArrayInputStream b = new ByteArrayInputStream(this.greatBeep);
        try {
            AudioPlayer.main(b);
        } catch (Exception ex) {
            ex.printStackTrace();
        }

    }


    public void badBeep(){



    }

}
/*
 * CaixaLoader.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.resultloader;

import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.LagarLotteryGlobals;
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.MalformedURLException;
import java.net.URL;
import java.util.List;
import java.util.zip.ZipEntry;
import java.util.zip.ZipInputStream;

public abstract class CaixaLoader implements ResultLoader{

    protected URL resultPath;

    protected InputStream getHtmlStream() throws IOException{

        if(this.resultPath.getProtocol().equalsIgnoreCase("http") ||
           this.resultPath.getProtocol().equalsIgnoreCase("ftp") ){

            InputStream in = this.resultPath.openStream();
            byte[] buf = new byte[8192];

            String localPath = LagarLotteryGlobals.LL_HOME;
            int pos = this.resultPath.getPath().lastIndexOf("/");
            localPath = localPath + this.resultPath.getPath().substring(pos);
            File localFile = new File(localPath);
            if(!localFile.exists()){
                localFile.createNewFile();
            }
            BufferedOutputStream bout = new BufferedOutputStream(new
FileOutputStream(localPath));

            int read = 0;
            while((read = in.read(buf)) != -1){
                bout.write(buf, 0, read);
            }
            bout.flush();
            bout.close();
            in.close();


            this.resultPath = localFile.toURI().toURL();

        }

        ZipInputStream in = new ZipInputStream(new
BufferedInputStream(this.resultPath.openStream()));

        ZipEntry entry = in.getNextEntry();

        if(entry == null) throw new IllegalArgumentException("BAD ZIP");

        while(!entry.getName().contains("HTM")){
            entry = in.getNextEntry();
            if(entry == null) throw new IllegalArgumentException("BAD ZIP");
        }

        return in;
    }

    public void setResultPath(String url) throws MalformedURLException {
        this.resultPath = getClass().getResource(url);

        if(this.resultPath == null){
            this.resultPath = new URL(url);
        }
    }

    public String getResultPath(){
        return this.resultPath.toString();
    }
```

```
}
/*
 * CardReaderWriterTest.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.zmisc.test;

import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.LagarLotteryGlobals;
import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.generator.GeneratorHeader;
import java.io.BufferedInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.util.zip.ZipEntry;
import java.util.zip.ZipInputStream;

public class CardReaderWriterTest {

    /** Creates a new instance of CardReaderWriterTest */
    public CardReaderWriterTest() {
    }

    public void readCards() throws Exception{

        FileInputStream fin = new FileInputStream("/home/lagar/teste/bb.llg");

        ZipInputStream zin = new ZipInputStream(fin);

        ZipEntry e = null;
        while( (e = zin.getNextEntry()) != null ){
            if(e.getName().equals(LagarLotteryGlobals.GENERATED_HEADER_NAME)){
                break;
            }
        }
        GeneratorHeader header = GeneratorHeader.loadFromStream(zin);

        zin.closeEntry();
        zin.close();

    }

    public void testGenCardIO() throws IOException{
        File f = new File("/home/lagar/teste/bb.llg");
        GenCardIO io = new GenCardIO(f);

        GameCard buff = new GameCard(0, null, io.getBallsPerCard());

        io.skipToCard(buff, io.size());
        System.out.println(buff);
    }



    public static void main(String[] args) throws Exception{

        new CardReaderWriterTest().testGenCardIO();
    }

}
/*
 * CellData.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis;

import java.awt.Color;

public class CellData {

    private String text;
    private Color bgColor;
    private Color fgColor;
    private boolean hit;

    public CellData(){

    }

    public CellData(String text, Color bg, Color fg, boolean hit){
        this.text = text;
        this.bgColor = bg;
        this.setFgColor(fg);
        this.setHit(hit);
    }

    public String getText() {
        return text;
```

```java
    }

    public void setText(String text) {
        this.text = text;
    }

    public Color getBgColor() {
        return bgColor;
    }

    public void setBgColor(Color c) {
        this.bgColor = c;
    }

    public boolean isHit() {
        return hit;
    }

    public void setHit(boolean hit) {
        this.hit = hit;
    }

    public Color getFgColor() {
        return fgColor;
    }

    public void setFgColor(Color fgColor) {
        this.fgColor = fgColor;
    }

}
/*
 * CellDataRenderer.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis;

import java.awt.Color;
import java.awt.Component;
import javax.swing.JButton;
import javax.swing.JLabel;
import javax.swing.JTable;
import javax.swing.SwingConstants;
import javax.swing.border.Border;
import javax.swing.table.TableCellRenderer;
```

```java
public class CellDataRenderer extends JLabel implements TableCellRenderer{

    private static Color DEFAULT_COLOR = new JButton("a").getBackground();

    public CellDataRenderer(){
        this.setOpaque(true);
        this.setHorizontalAlignment(SwingConstants.CENTER);
    }

    public Component getTableCellRendererComponent(JTable table, Object value,
boolean isSelected, boolean hasFocus, int row, int column) {

        CellData c = (CellData) value;
        Color bgColor = c.getBgColor();
        Color fgColor = c.getFgColor();
        if(bgColor == null) bgColor = DEFAULT_COLOR;

        if(!isSelected){
            this.setBackground(bgColor);
        }
        else{
            this.setBackground(bgColor.darker());
        }
        this.setText(c.getText());
        this.setForeground(fgColor);

        return this;
    }

}
/*
 * Chartable.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis;

import org.jfree.chart.JFreeChart;

/**
 *
 * @author lagar
 */
public interface Chartable {

    public JFreeChart getChart();
}
/*
 * ChartAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis;

import java.awt.CardLayout;
import java.util.ArrayList;
import java.util.List;
import javax.swing.JPanel;
import org.jfree.chart.ChartPanel;

public abstract class ChartAnalysis extends Analysis implements Chartable {

    public List<JPanel> getPanels() {

        List<JPanel> panels = new ArrayList<JPanel>();

        ChartPanel panel = new ChartPanel(getChart());
        panel.setLayout(new CardLayout());
        panel.setName("Chart");
        panels.add(panel);

        return panels;
    }

}
/*
 * LineAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis.gameanalysis;

import br.ufsc.inf.lagar.analysis.CellData;
import br.ufsc.inf.lagar.analysis.DefaultTableChartAnalysis;
import br.ufsc.inf.lagar.main.GameCard;
import java.awt.Color;
import java.lang.reflect.InvocationTargetException;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Set;
import javax.swing.JOptionPane;
import javax.swing.SwingUtilities;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.labels.PieSectionLabelGenerator;
import org.jfree.chart.labels.StandardPieSectionLabelGenerator;
import org.jfree.chart.plot.MultiplePiePlot;
import org.jfree.chart.plot.PiePlot;
import org.jfree.data.category.DefaultCategoryDataset;
import org.jfree.data.general.DefaultPieDataset;
import org.jfree.util.TableOrder;

public class ColumnAnalysis extends DefaultTableChartAnalysis {

    private int[][] colsData;

    /** Creates a new instance of LineAnalysis */
    public ColumnAnalysis() {
    }

    protected void fillTableChart() {

        GameCard cardBuffer = new GameCard();
        int perLine = getCurrentGameConfiguration().getNumbersPerLine();
        int total = getCurrentGameConfiguration().getNumbersPerCard();
        int numCols = perLine;
        int sizeCols = total / perLine;
        int[] sum = new int[numCols+1];

        for(int index = 0; this.gci.hasNext(); ++index){
            this.gci.next(cardBuffer);
            byte[] balls = cardBuffer.getBalls();

            for(int i = 0; i < balls.length; ++i){
                int ball = balls[i];

                for(int k = 1; k <= numCols; ++k){
                    int[] col = this.colsData[k];
                    if(Arrays.binarySearch(col, ball) >= 0){
                        ++sum[k];
                        break;
                    }
                }
            }
            for(int i = 1; i <= numCols; ++i){
                String category = String.valueOf(i);
                this.tcols.get(category)[index] = this.getHitz(sum[i]);
                Integer dc = this.dataChart.get(category);
                this.dataChart.put(category, dc + sum[i]);
            }

            this.tcols.get(this.columnOne)[index] = new
CellData(String.valueOf(cardBuffer.getSerial()), null, Color.BLACK, false);

            Arrays.fill(sum, 0);
        }

    }

    protected List<String> buildCategories() {

        List<String> result = new ArrayList<String>();

        int perLine = getCurrentGameConfiguration().getNumbersPerLine();
        int total = getCurrentGameConfiguration().getNumbersPerCard();
        int numCols = perLine;
        int sizeCols = total / perLine;


        for(int i = 1; i <= numCols; ++i){
            result.add(String.valueOf(i));
            this.colsData = new int[numCols+1][perLine];
        }

        for(int i = 1; i <= numCols; ++i){

            int currNum = i;
            for(int j = 0; j < sizeCols; ++j){

                this.colsData[i][j] = currNum;
                currNum += perLine;
            }

        }

        return result;
    }

    public String getName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Column_analysis");
    }

    public String getHelpInfo() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Column_analysis_help_info");
    }




}
/**
 * CombinationGenerator.java
 *
 * Combination Generator
 *  by Michael Gilleland, Merriam Park Software
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */
package br.ufsc.inf.lagar.generator;

// --------------------------------------
// Systematically generate combinations.
// --------------------------------------

import java.math.BigInteger;

public class CombinationGenerator
{

  private int[] a;
  private int n;
  private int r;
  private BigInteger numLeft;
  private BigInteger total;

// ------------
// Constructor
// ------------

public CombinationGenerator(int n, int r)
{
    if(r > n)
    {
        throw new IllegalArgumentException();
    }
    if(n < 1)
    {
        throw new IllegalArgumentException();
    }
    this.n = n;
    this.r = r;
    a = new int[r];
    BigInteger nFact = getFactorial(n);
    BigInteger rFact = getFactorial(r);
    BigInteger nminusrFact = getFactorial(n - r);
    total = nFact.divide(rFact.multiply(nminusrFact));
    reset();
}

// ------
// Reset
// ------
```

```
public void reset()
{
    for(int i = 0; i < a.length; i++)
    {
        a[i] = i;
    }
    numLeft = new BigInteger(total.toString());
}

// ------------------------------------------------
// Return number of combinations not yet generated
// ------------------------------------------------

public BigInteger getNumLeft()
{
    return numLeft;
}

// -----------------------------
// Are there more combinations?
// -----------------------------

public boolean hasMore()
{
    return numLeft.compareTo(BigInteger.ZERO) == 1;
}

// ------------------------------------
// Return total number of combinations
// ------------------------------------

public BigInteger getTotal()
{
    return total;
}

// ------------------
// Compute factorial
// ------------------

public static BigInteger getFactorial(int n)
{
    BigInteger fact = BigInteger.ONE;
    for(int i = n; i > 1; i--)
    {
        fact = fact.multiply(new BigInteger(Integer.toString(i)));
    }
    return fact;
}

// --------------------------------------------------------
// Generate next combination (algorithm from Rosen p. 286)
// --------------------------------------------------------

public int[] getNext()
{

    if(numLeft.equals(total))
    {
        numLeft = numLeft.subtract(BigInteger.ONE);
        return a;
    }

    int i = r - 1;
    while(a[i] == n - r + i)
    {
        i--;
    }
    a[i] = a[i] + 1;
    for(int j = i + 1; j < r; j++)
    {
        a[j] = a[i] + j - i;
    }

    numLeft = numLeft.subtract(BigInteger.ONE);
    return a;

}
}/**
 * CombinationGeneratorTest.java
 *
 * Copyright (C) 2006 Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */
package br.ufsc.inf.lagar.utils;

import br.ufsc.inf.lagar.generator.CombinationGenerator;
import junit.framework.TestCase;

/**
 * @author lagar
 *
 */
public class CombinationGeneratorTest extends TestCase
{

public void testCombinationGenerator()
{
    int[] elements = { 10, 20, 30, 40, 50 };
    int[] indices;
    CombinationGenerator x = new CombinationGenerator(elements.length, 3);
    StringBuffer combination = new StringBuffer();
    while(x.hasMore())
    {

        indices = x.getNext();
        for(int i = 0; i < indices.length; i++)
        {
            combination.append(elements[indices[i]]);
            if(indices.length - i != 1)
                combination.append(" - ");
        }
```

```java
            combination.append("\n");
        }

        StringBuffer expectedValue = new StringBuffer();
        expectedValue.append("10 - 20 - 30");
        expectedValue.append("\n");
        expectedValue.append("10 - 20 - 40");
        expectedValue.append("\n");
        expectedValue.append("10 - 20 - 50");
        expectedValue.append("\n");
        expectedValue.append("10 - 30 - 40");
        expectedValue.append("\n");
        expectedValue.append("10 - 30 - 50");
        expectedValue.append("\n");
        expectedValue.append("10 - 40 - 50");
        expectedValue.append("\n");
        expectedValue.append("20 - 30 - 40");
        expectedValue.append("\n");
        expectedValue.append("20 - 30 - 50");
        expectedValue.append("\n");
        expectedValue.append("20 - 40 - 50");
        expectedValue.append("\n");
        expectedValue.append("30 - 40 - 50");
        expectedValue.append("\n");

        System.out.println(combination.toString());

        assertEquals("CombinationGeneratorTest", combination.toString(),
                     expectedValue.toString());
    }

}
/*
 * DecisionDialog.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.ui.dialog;

/**
 *
 * @author  lagar
 */
public class DecisionDialog extends javax.swing.JDialog {

    /** Creates new form DecisionDialog */
    public DecisionDialog(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
        this.setLocationRelativeTo(null);
    }



    public br.inf.ufsc.lagar.swingutil.SimpleInternalFrame getTitlePanel() {
        return titlePanel;
    }

    public javax.swing.JButton getCancelButton() {
        return cancelButton;
    }

    public javax.swing.JPanel getMainPanel() {
        return mainPanel;
    }

    public javax.swing.JButton getOkButton() {
        return okButton;
    }

    public javax.swing.JPanel getButtonsPanel() {
        return buttonsPanel;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        titlePanel = new br.inf.ufsc.lagar.swingutil.SimpleInternalFrame();
        mainPanel = new javax.swing.JPanel();
        buttonsPanel = new javax.swing.JPanel();
        cancelButton = new javax.swing.JButton();
        okButton = new javax.swing.JButton();
        jSeparator1 = new javax.swing.JSeparator();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        setTitle(bundle.getString("Option")); // NOI18N
        titlePanel.setTitle("Title");

        mainPanel.setLayout(new java.awt.CardLayout());

        cancelButton.setText(bundle.getString("Cancel")); // NOI18N

        okButton.setText(bundle.getString("Ok")); // NOI18N

        org.jdesktop.layout.GroupLayout buttonsPanelLayout = new
org.jdesktop.layout.GroupLayout(buttonsPanel);
        buttonsPanel.setLayout(buttonsPanelLayout);
        buttonsPanelLayout.setHorizontalGroup(
            buttonsPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
buttonsPanelLayout.createSequentialGroup()
                .addContainerGap(343, Short.MAX_VALUE)
                .add(okButton)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(cancelButton)
                .addContainerGap())
        );

        buttonsPanelLayout.linkSize(new java.awt.Component[] {cancelButton,
okButton}, org.jdesktop.layout.GroupLayout.HORIZONTAL);

        buttonsPanelLayout.setVerticalGroup(
            buttonsPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
            .add(buttonsPanelLayout.createSequentialGroup()
                .add(buttonsPanelLayout.createParallelGroup(org.jdesktop.layout.Group
Layout.BASELINE)
                    .add(cancelButton)
                    .add(okButton))
                .addContainerGap(12, Short.MAX_VALUE))
        );

        buttonsPanelLayout.linkSize(new java.awt.Component[] {cancelButton,
okButton}, org.jdesktop.layout.GroupLayout.VERTICAL);

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(titlePanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 511,
Short.MAX_VALUE)
            .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 511,
Short.MAX_VALUE)
            .add(buttonsPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(jSeparator1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 511,
Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(titlePanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 233,
Short.MAX_VALUE)
                .add(2, 2, 2)
                .add(jSeparator1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 10,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(buttonsPanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
        );
        pack();
    }// </editor-fold>//GEN-END:initComponents


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JPanel buttonsPanel;
    private javax.swing.JButton cancelButton;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JPanel mainPanel;
    private javax.swing.JButton okButton;
    private br.inf.ufsc.lagar.swingutil.SimpleInternalFrame titlePanel;
    // End of variables declaration//GEN-END:variables

}
/*
 * DefaultGameCardPrinter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.printer;

import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameCardLayout;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Point;
import java.awt.print.PageFormat;
import java.awt.print.Paper;
import java.awt.print.PrinterException;
import java.io.File;
import java.io.IOException;
import java.util.List;

public class DefaultGameCardPrinter extends GameCardPrinter {

    private GenCardIO cards;
    private GameCard cardBuffer = new GameCard();

    /** Creates a new instance of DefaultGameCardPrinter */
    public DefaultGameCardPrinter() {
    }

    public void setFile(File gameCard) throws IOException {

        this.cards = new GenCardIO(gameCard);
    }

    public int print(Graphics g, PageFormat pf, int pageIndex) throws
PrinterException {

        if(cards.size() < (pageIndex+1)){
            return NO_SUCH_PAGE;
        }

        try {
            this.cards.skipToCard(cardBuffer, pageIndex + 1);
        } catch (IOException ex) {
            throw new RuntimeException(ex);
        }

        this.printCard(this.cardBuffer, g, pf);
        this.printExtendedBalls(this.cardBuffer, g, pf, pageIndex);
        this.printExtendedText(this.cardBuffer, g, pf, pageIndex);

        return PAGE_EXISTS;
    }

    private void printCard(GameCard gc, Graphics g, PageFormat pf){

        /* User (0,0) is typically outside the imageable area, so we must
         * translate by the X and Y values in the PageFormat to avoid clipping
         */
        Graphics2D g2d = (Graphics2D) g;

        Paper p = (Paper) pf.getPaper().clone();
        p.setImageableArea(0, 0, p.getWidth(), p.getHeight());
        pf.setPaper(p);

        double x = (pf.getWidth() / 2) - cmToPixel(4.1);
        double y = 0;

        g2d.translate(x, y);

        GameCardLayout gcl = GameManager.getCurrentGame().getCardLayout();
        double width = gcl.getBallWidth();
        double height = gcl.getBallHeigth();

        for(int i: gc.getBalls()){

            Point.Double p2d = gcl.getBalls().get(i);
            /* Now we perform our rendering */
            g2d.fillRect((int)cmToPixel(p2d.x), (int)cmToPixel(p2d.y),
(int)cmToPixel(width), (int)cmToPixel(height));
        }

    }

    protected void printExtendedText(GameCard gc, Graphics g, PageFormat pf, int
pageIndex) throws PrinterException {

        GameCardLayout gcl = GameManager.getCurrentGame().getCardLayout();
        GameCardLayout.ExtendedText ext = gcl.getExtendedText();

        double x = (ext.getX());
        double y = (ext.getY());

        Graphics2D g2d = (Graphics2D) g;
        g2d.drawString("# "+gc.getSerial(), (int)cmToPixel(x), (int)cmToPixel(y));
    }

    protected void printExtendedBalls(GameCard gc, Graphics g, PageFormat pf, int
pageIndex) throws PrinterException {

    }

}
```

```java
/*
 * DefaultTableChartAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis;

import br.ufsc.inf.lagar.main.GameCardIterator;
import java.awt.Color;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
import java.util.Set;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.labels.PieSectionLabelGenerator;
import org.jfree.chart.labels.StandardPieSectionLabelGenerator;
import org.jfree.chart.plot.PiePlot;
import org.jfree.data.general.DefaultPieDataset;

public abstract class DefaultTableChartAnalysis extends TableChartAnalysis{

    protected GameCardIterator gci;
    protected List<String> columns;
    protected Map<String, CellData[]> tcols;
    protected Map<String, Integer> dataChart;
    protected String columnOne = "n";

    protected CellData hit = new CellData("x", new Color(40,210,20), Color.BLACK,
true);
    protected CellData miss = new CellData("", new Color(255,255,255),Color.BLACK,
false);

    private Map<Integer, CellData> misses = new HashMap<Integer, CellData>();
    private Map<Integer, CellData> hitz = new HashMap<Integer, CellData>();

    /** Creates a new instance of DefaultTableChartAnalysis */
    public DefaultTableChartAnalysis() {
    }

    public int getRowCount() {
        return (int) this.gci.size();
    }

    public int getColumnCount() {
        return this.columns.size();
    }

    public String getColumnName(int columnIndex) {
        return this.columns.get(columnIndex);
    }

    public CellData getCellDataAt(int rowIndex, int columnIndex) {

        String cat = this.columns.get(columnIndex);
        CellData[] rows = this.tcols.get(cat);
        CellData cd = rows[rowIndex];
//        if(!cd.isHit() && columnIndex != 0 && cd.getText().equals("")){
//
//            if(rowIndex == 0){
//                rows[rowIndex] = this.getMiss(1);
//                return rows[rowIndex];
//            }
//            //CellData up = this.getCellDataAt(rowIndex-1, columnIndex);
//            CellData up = this.tcols.get(cat)[rowIndex-1];
//            if(up.isHit()){
//                rows[rowIndex] = this.getMiss(1);
//                return rows[rowIndex];
//            }
//            //if(up.getText() == null || up.getText().equals("")){
//                //up = this.getCellDataAt(rowIndex, columnIndex);
//                this.preloadMisses(rowIndex, columnIndex);
//            //}
//            //int value = Integer.parseInt(up.getText());
//            //rows[rowIndex] = this.getMiss(++value);
//            return rows[rowIndex];
//        }
//        else{
          return cd;
//        }
    }

    public void calculateAnalysis() {

        this.gci = getGameCardIterator();

        List<String> categories = this.buildCategories();
        this.columns = new ArrayList<String>(1+categories.size());
        this.columns.add(this.columnOne);
        this.columns.addAll(categories);

        this.tcols = new HashMap<String, CellData[]>();
        this.dataChart = new LinkedHashMap<String,Integer>();

        for(String c: this.columns){
            CellData[] line = new CellData[(int)this.gci.size()];
            Arrays.fill(line, this.miss);
            this.tcols.put(c, line);

            this.dataChart.put(c, new Integer(0));
        }
        this.dataChart.remove(this.columnOne);

        this.fillTableChart();

        this.gci.close();
        this.preloadMisses();

    }

    protected abstract void fillTableChart();

    protected abstract List<String> buildCategories();

    public JFreeChart getChart() {
        DefaultPieDataset dataset = new DefaultPieDataset();
        Set<String> keys = this.dataChart.keySet();
        for(String key: keys){
            dataset.setValue(key, this.dataChart.get(key));
        }

        JFreeChart chart = ChartFactory.createPieChart(this.getName(),
                dataset,true,true,false);

        PiePlot plot = (PiePlot) chart.getPlot();
        PieSectionLabelGenerator generator = new StandardPieSectionLabelGenerator(
                "{0} - {2}", new DecimalFormat("0"), new DecimalFormat("0.00%")
                );
        plot.setLabelGenerator(generator);

        return chart;
    }

    public CellData getMiss(int miss){

        Integer key = Integer.valueOf(miss);
        CellData cd = this.misses.get(key);
        if(cd == null){
            cd = new CellData(String.valueOf(miss), Color.WHITE, Color.LIGHT_GRAY,
false);
            this.misses.put(key, cd);
        }
        return cd;
    }

    public CellData getHitz(int hitz){

        Integer key = Integer.valueOf(hitz);
        CellData cd = this.hitz.get(key);
        if(cd == null){
            cd = new CellData(String.valueOf(hitz), Color.WHITE, Color.BLACK, true);
            this.hitz.put(key, cd);
        }
        return cd;
    }

    private void preloadMisses() {
        for(String s : this.columns){

            if(s.equals(this.columnOne)) continue;

            CellData[] rows = this.tcols.get(s);

            if(!rows[0].isHit()) rows[0] = this.getMiss(1);

            for(int i = 1; i < rows.length; ++i){
                CellData cd = rows[i];
                if(!cd.isHit()){
                    cd = rows[i-1];
                    if(cd.isHit()){
                        rows[i] = this.getMiss(1);
                    }
                    else{
                        int up = Integer.parseInt(cd.getText());
                        rows[i] = this.getMiss(++up);
                    }
                }

            }
        }
    }

}
/*
 * FileChooserLLG.java
 *
 * Created on 22 de Março de 2007, 20:44
 */

package br.ufsc.inf.lagar.ui.panels;

import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.generator.GeneratorHeader;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.MainFrame;
import br.ufsc.inf.lagar.ui.FileExtensionFilter;
import br.ufsc.inf.lagar.ui.dialog.DecisionDialog;
import java.awt.Dimension;
import java.io.File;
import java.io.IOException;
import java.text.DateFormat;
import java.text.NumberFormat;
import java.util.Locale;
import javax.swing.JEditorPane;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.JScrollPane;
import javax.swing.SwingUtilities;
import javax.swing.filechooser.FileFilter;

/**
 *
 * @author  lagar
 */
public class FileChooserLLG extends javax.swing.JPanel {


    public enum ChooserMode {
        SAVE_MODE,
        LOAD_MODE
    }

    private ChooserMode mode = ChooserMode.LOAD_MODE;

    private JFileChooser fileChooser;

    /** Creates new form FileChooserLLG */
    public FileChooserLLG() {
        initComponents();
        this.fileChooser = new JFileChooser();
        this.fileChooser.setMultiSelectionEnabled(false);
        this.fileChooser.setAcceptAllFileFilterUsed(false);

        FileFilter[] filters = fileChooser.getChoosableFileFilters();

        for (FileFilter f : filters) {
            fileChooser.removeChoosableFileFilter(f);
        }

        fileChooser.addChoosableFileFilter(new FileExtensionFilter("llg",
                "LagarLottery Games (*.LLG)"));

        this.infoLabel.setVisible(false);
        this.numberInfoLabel.setVisible(false);
        this.infoButton.setVisible(false);
        this.gameTypeLabel.setVisible(false);
        this.gameTypeInfo.setVisible(false);
        this.spacerLabel.setVisible(false);
    }

    @Override
    public void setEnabled(boolean enabled){
        super.setEnabled(enabled);
        this.fileField.setEnabled(enabled);
        this.fileButton.setEnabled(enabled);
        this.fileLabel.setEnabled(enabled);
    }

    private void chooseFile() {

        int returnVal = JFileChooser.CANCEL_OPTION;
        switch(mode){
            case SAVE_MODE: returnVal = this.fileChooser.showSaveDialog(null); break;
            case LOAD_MODE: returnVal = this.fileChooser.showOpenDialog(null); break;
        }

        if(returnVal == JFileChooser.APPROVE_OPTION){
            String selectedFile = this.fileChooser.getSelectedFile().toString();
            if(!selectedFile.toLowerCase().endsWith(".llg")){
                selectedFile = selectedFile + ".llg";
            }
            this.fileField.setText(selectedFile);

            if(this.mode == ChooserMode.LOAD_MODE){
                GeneratorHeader header;

                try {
                    header = GeneratorHeader.fetchGeneratorHeader(new
File(selectedFile));
                    NumberFormat nf = NumberFormat.getInstance();
                    this.numberInfoLabel.setText(nf.format(header.getNumberOfCards())
);
                    this.gameTypeInfo.setText(header.getGameName());
                    this.infoLabel.setVisible(true);
                    this.numberInfoLabel.setVisible(true);
                    this.infoButton.setVisible(true);
                    this.gameTypeLabel.setVisible(true);
                    this.gameTypeInfo.setVisible(true);
                    this.spacerLabel.setVisible(true);

                } catch (IOException ex) {
                    ex.printStackTrace();
                }
            }

        }

    }

    public String getFilePath(){
        return this.fileField.getText();
    }

    public ChooserMode getMode() {
        return mode;
    }

    public void setMode(ChooserMode mode) {
        this.mode = mode;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        fileLabel = new javax.swing.JLabel();
        fileField = new javax.swing.JTextField();
        fileButton = new javax.swing.JButton();
        infoLabel = new javax.swing.JLabel();
        infoButton = new javax.swing.JButton();
        gameTypeLabel = new javax.swing.JLabel();
        numberInfoLabel = new javax.swing.JLabel();
        gameTypeInfo = new javax.swing.JLabel();
        spacerLabel = new javax.swing.JLabel();

        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        fileLabel.setText(bundle.getString("File:")); // NOI18N

        fileButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/fileopen.png")));
        fileButton.setText(bundle.getString("Choose..")); // NOI18N
        fileButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                fileButtonActionPerformed(evt);
            }
        });

        infoLabel.setText(bundle.getString("Number_of_cards:")); // NOI18N

        infoButton.setText(bundle.getString("Show_cards")); // NOI18N
        infoButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                infoButtonActionPerformed(evt);
            }
        });

        gameTypeLabel.setText(bundle.getString("Game_type:")); // NOI18N

        numberInfoLabel.setText("xxxx");
        gameTypeInfo.setText("xxxx");

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(fileLabel)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
                    .add(layout.createSequentialGroup()
                        .add(infoLabel)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(numberInfoLabel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 369, Short.MAX_VALUE)
                        .addContainerGap())
                    .add(layout.createSequentialGroup()
                        .add(fileField, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
380, Short.MAX_VALUE)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(fileButton))
                    .add(layout.createSequentialGroup()
                        .add(gameTypeLabel)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(gameTypeInfo,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 70,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(infoButton)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(spacerLabel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 219, Short.MAX_VALUE))))
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASEL
INE)
                    .add(fileLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
20, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(fileButton)
                    .add(fileField, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
22, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASEL
INE)
                    .add(gameTypeLabel)
                    .add(gameTypeInfo)
                    .add(infoButton)
                    .add(spacerLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
24, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASEL
INE)
                    .add(infoLabel)
                    .add(numberInfoLabel)))
        );
    }// </editor-fold>//GEN-END:initComponents

    private void infoButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_infoButtonActionPerformed
```

```
        GameCardsInfo gci = new GameCardsInfo(null, false);
        gci.setGameFilePath(this.getFilePath());
        gci.showGameCardsInfo();

    }//GEN-LAST:event_infoButtonActionPerformed

    private void fileButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_fileButtonActionPerformed
        this.chooseFile();
    }//GEN-LAST:event_fileButtonActionPerformed


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton fileButton;
    private javax.swing.JTextField fileField;
    private javax.swing.JLabel fileLabel;
    private javax.swing.JLabel gameTypeInfo;
    private javax.swing.JLabel gameTypeLabel;
    private javax.swing.JButton infoButton;
    private javax.swing.JLabel infoLabel;
    private javax.swing.JLabel numberInfoLabel;
    private javax.swing.JLabel spacerLabel;
    // End of variables declaration//GEN-END:variables

}
/*
 * FileExtensionFilter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.ui;

import java.io.File;
import javax.swing.filechooser.FileFilter;

public class FileExtensionFilter extends FileFilter {

        private String m_description = null;

        private String m_extension = null;

        public FileExtensionFilter(String extension, String description) {
                m_description = description;
                m_extension = extension;
        }

        public String getDescription() {
                return m_description;
        }

        public boolean accept(File f) {
                if (f == null)
                        return false;

                if(f.isDirectory()) return true;

                return f.getName().toLowerCase().endsWith(m_extension);
        }
}
/*
 * Filterable.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter;

import br.ufsc.inf.lagar.main.GameCard;
import javax.swing.JTextField;

/**
 *
 * @author lagar
 */
public interface Filterable {

    public static final String OR_SEPARATOR = "\\|"; //just the "|"

    public String getShortName();

    public String getFullName();

    public String getDescription();

    public String getFilterHelp();

    /**
     * Set if the filter will be in accept or reject mode.
     * @param mode true means accept mode, false means reject mode.
     */
    public void setAcceptReject(boolean mode);

    public void setTextInput(String input) throws TextInputFilterException;

    /**
```

```
     * Submit the GameCard to the filter.
     * @param card the GameCard to be tested against the filter
     * @return true if the card passed the filter test, false if the filter rejected
the card.
     */
    public boolean filter(GameCard card);

    public boolean acceptGame(String gameName);

}
/*
 * FilterManager.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter;

import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.generator.GeneratorHeader;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.main.LagarLotteryGlobals;
import br.ufsc.inf.lagar.ui.ProgressInterface;
import br.ufsc.inf.lagar.ui.panels.filter.SingleFilterPanel;
import java.io.DataOutputStream;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.text.NumberFormat;
import java.util.List;
import java.util.zip.Deflater;
import java.util.zip.ZipEntry;
import java.util.zip.ZipOutputStream;

public class FilterManager implements ProgressInterface{

    private boolean cancel;
    private int passedCards = 0;
    private int currentCard = 0;
    private long total;
    private List<SingleFilterPanel> filters;
    private File loadCards;
    private File saveCards;
    private GeneratorHeader header;

    /** Creates a new instance of FilterManager */
    public FilterManager(File loadCards, File saveCards, List<SingleFilterPanel>
filters) throws IOException{

        this.filters = filters;
        this.loadCards = loadCards;
        this.saveCards = saveCards;

        if(!saveCards.isFile() || !saveCards.canWrite() ||
saveCards.equals(loadCards)){
            throw new IOException("Can't save filtered cards on choosen location.
Choose another.");
        }

        this.header = GeneratorHeader.fetchGeneratorHeader(loadCards);
        this.total = this.header.getNumberOfCards();

    }

    public void runFilters() throws IOException{


        FileOutputStream fos = new FileOutputStream(saveCards);
        ZipOutputStream zos = new ZipOutputStream(fos);
        zos.setLevel(Deflater.BEST_SPEED);

        DataOutputStream dos = new DataOutputStream(zos);

        zos.putNextEntry(new ZipEntry(LagarLotteryGlobals.GENERATED_CARDS_NAME));

        GenCardIO input = new GenCardIO(loadCards);
        while(input.hasNext()){

            boolean ok = true;
            ++currentCard;

            GameCard gc = input.next();
            for(SingleFilterPanel sfp: filters){
                Filterable f = sfp.getFilter();
                if(!f.filter(gc)){
                    ok = false;
                    break;
                }
            }

            if(ok){
                dos.write(gc.getBalls());
                ++passedCards;
            }

            if(this.cancel){
                zos.closeEntry();
                dos.close();
                this.saveCards.delete();
                throw new IOException("Cancelled");
            }
        }

        dos.flush();
        zos.closeEntry();

        header.setNumberOfCards(getPassedCards());

        zos.putNextEntry(new ZipEntry(LagarLotteryGlobals.GENERATED_HEADER_NAME));
        header.saveToStream(dos);
        dos.flush();
        zos.closeEntry();

        dos.close();

    }

    public void cancel() {
        this.cancel = true;
    }

    public String getProgressText() {
        NumberFormat nf = NumberFormat.getInstance();

        return nf.format(this.currentCard) + " of " + nf.format(total) + "
filtered.";
    }

    public int getProgressValue() {
        return this.currentCard;
    }

    public int getMaxProgressValue() {

        return (int)this.total;
    }

    public int getPassedCards() {
        return passedCards;
    }

}
/*
 * FilterPanel.java
 *
 * Created on 29 de Março de 2007, 21:02
 */

package br.ufsc.inf.lagar.ui.panels;

import br.ufsc.inf.lagar.filter.FilterManager;
import br.ufsc.inf.lagar.filter.TextInputFilterException;
import br.ufsc.inf.lagar.ui.dialog.ProgressDialog;
import br.ufsc.inf.lagar.ui.panels.filter.SingleFilterPanel;
import java.io.File;
import java.io.IOException;
import java.text.NumberFormat;
import java.util.List;
import javax.swing.JOptionPane;
import javax.swing.SwingUtilities;

/**
 *
 * @author  lagar
 */
public class FilterPanel extends javax.swing.JPanel {

    /** Creates new form FilterPanel */
    public FilterPanel() {
        initComponents();
        this.fileChooserSave.setMode(FileChooserLLG.ChooserMode.SAVE_MODE);
    }
    public void loadAvailableFilters() throws Exception{
        this.filterSelectPanel1.loadAvailableFilters();
    }

    public boolean validateInput(){

        if(this.fileChooserLoad.getFilePath() == null ||
           this.fileChooserLoad.getFilePath().equals("") ){

            JOptionPane.showMessageDialog(
                    this,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Choose_a_game_to_filter."),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
            return false;
        }

        if(this.filterSelectPanel1.getAddedFilters().size() == 0){

            JOptionPane.showMessageDialog(
                    this,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Choose_a_filter."),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
            return false;
        }

        if(this.fileChooserSave.getFilePath() == null ||
           this.fileChooserSave.getFilePath().equals("") ){

            JOptionPane.showMessageDialog(
                    this,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Choose_a_file_to_save_the_filtered_cards."),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
            return false;
        }

        List<SingleFilterPanel> filters = this.filterSelectPanel1.getAddedFilters();
        for(SingleFilterPanel sfp: filters){
            try {
                sfp.getFilter().setTextInput(sfp.getFilterField().getText());
            } catch (TextInputFilterException ex) {
                JOptionPane.showMessageDialog(
                        null,
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("<html>Filter_<b>")+sfp.getPosition()+java.util.ResourceBundle.getB
undle("resources/bundles/lagarLottery").getString("</b>_has_input_errors.</html>"),
                        java.util.ResourceBundle.getBundle("resources/bundles/lagarLo
ttery").getString("Error"),
                        JOptionPane.ERROR_MESSAGE);

                return false;
            }
        }
        return true;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        simpleInternalFrame1 = new br.inf.ufsc.lagar.swingutil.SimpleInternalFrame();
        infoPanel = new javax.swing.JPanel();
        infoLabel = new javax.swing.JLabel();
        loadPanel = new javax.swing.JPanel();
        fileChooserLoad = new br.ufsc.inf.lagar.ui.panels.FileChooserLLG();
        mainPanel = new javax.swing.JPanel();
        filterSelectPanel1 = new
br.ufsc.inf.lagar.ui.panels.filter.FilterSelectPanel();
        savePanel = new javax.swing.JPanel();
        fileChooserSave = new br.ufsc.inf.lagar.ui.panels.FileChooserLLG();
        sp1 = new javax.swing.JLabel();
        sp2 = new javax.swing.JLabel();
        runFiltersButton = new javax.swing.JButton();

        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        simpleInternalFrame1.setTitle(bundle.getString("Filters")); // NOI18N

        infoLabel.setText(bundle.getString("Use_this_module_to_eliminate_cards_from_a
_generated_cards_file_(LLG)."")); // NOI18N

        org.jdesktop.layout.GroupLayout infoPanelLayout = new
org.jdesktop.layout.GroupLayout(infoPanel);
        infoPanel.setLayout(infoPanelLayout);
        infoPanelLayout.setHorizontalGroup(
            infoPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(infoPanelLayout.createSequentialGroup()
                .addContainerGap()
                .add(infoLabel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 561,
Short.MAX_VALUE)
                .addContainerGap())
        );
        infoPanelLayout.setVerticalGroup(
            infoPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(infoPanelLayout.createSequentialGroup()
                .addContainerGap()
                .add(infoLabel)
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );

        loadPanel.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.getSt
ring("Cards_to_filter"))); // NOI18N

        org.jdesktop.layout.GroupLayout loadPanelLayout = new
org.jdesktop.layout.GroupLayout(loadPanel);
        loadPanel.setLayout(loadPanelLayout);
        loadPanelLayout.setHorizontalGroup(
            loadPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(fileChooserLoad, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 575,
Short.MAX_VALUE)
        );
        loadPanelLayout.setVerticalGroup(
            loadPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(loadPanelLayout.createSequentialGroup()
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .add(fileChooserLoad, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
        );

        mainPanel.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.getSt
ring("Filters"))); // NOI18N

        org.jdesktop.layout.GroupLayout mainPanelLayout = new
org.jdesktop.layout.GroupLayout(mainPanel);
        mainPanel.setLayout(mainPanelLayout);
        mainPanelLayout.setHorizontalGroup(
            mainPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(filterSelectPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
575, Short.MAX_VALUE)
        );
        mainPanelLayout.setVerticalGroup(
            mainPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(filterSelectPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
294, Short.MAX_VALUE)
        );

        savePanel.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.getSt
ring("Save_filtered_cards"))); // NOI18N

        org.jdesktop.layout.GroupLayout savePanelLayout = new
org.jdesktop.layout.GroupLayout(savePanel);
        savePanel.setLayout(savePanelLayout);
        savePanelLayout.setHorizontalGroup(
            savePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(fileChooserSave, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 575,
Short.MAX_VALUE)
        );
        savePanelLayout.setVerticalGroup(
            savePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(savePanelLayout.createSequentialGroup()
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .add(fileChooserSave, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
        );

        runFiltersButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/runprog.png")));
        runFiltersButton.setText(bundle.getString("Run_filters")); // NOI18N
        runFiltersButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                runFiltersButtonActionPerformed(evt);
            }
        });
        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(simpleInternalFrame1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
585, Short.MAX_VALUE)
            .add(infoPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(loadPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .add(sp1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 221,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(runFiltersButton)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(sp2, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 221,
Short.MAX_VALUE))
            .add(org.jdesktop.layout.GroupLayout.TRAILING, savePanel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(simpleInternalFrame1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(infoPanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(loadPanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(savePanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAIL
ING)
                    .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.B
ASELINE)
                        .add(sp1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                        .add(sp2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(runFiltersButton)))
        );

        layout.linkSize(new java.awt.Component[] {sp1, sp2},
org.jdesktop.layout.GroupLayout.VERTICAL);

    }// </editor-fold>//GEN-END:initComponents
    private void runFiltersButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_runFiltersButtonActionPerformed
        if(this.validateInput()){

            File load = new File(this.fileChooserLoad.getFilePath());
            File save = new File(this.fileChooserSave.getFilePath());

            try {

                if(load.equals(save)){
                    JOptionPane.showMessageDialog(
                            null,
                            java.util.ResourceBundle.getBundle("resources/bundles/lag
arLottery").getString("Can't_save_on_same_place_that_is_loading_the_cards."),
                            java.util.ResourceBundle.getBundle("resources/bundles/lag
arLottery").getString("Error"),
                            JOptionPane.ERROR_MESSAGE);
                    return;

                }

                if(save.exists()){
                    save.delete();
                }
                save.createNewFile();

                final FilterManager filterManager = new FilterManager(
                        load,
                        save,
                        this.filterSelectPanel1.getAddedFilters());

                final ProgressDialog pd = new ProgressDialog(null, true,
filterManager);
                pd.setLocationRelativeTo(null);
                pd.getTitleLabel().setText(java.util.ResourceBundle.getBundle("resour
ces/bundles/lagarLottery").getString("Please_wait.._filtering_your_cards."));

                Thread t = new Thread(new Runnable() {
                    public void run() {
                        try {
                            filterManager.runFilters();

                            JOptionPane.showMessageDialog(
                                 null,
                                java.util.ResourceBundle.getBundle("resources/bundles
/lagarLottery").getString("Run_filters_successfully") + "\n" +
                                    java.util.ResourceBundle.getBundle("resources/bun
dles/lagarLottery").getString("Number_of_cards_that_passed_the_filters:_")+
                                  NumberFormat.getInstance().format(filterManager.ge
tPassedCards()) + "\n" +
                                  java.util.ResourceBundle.getBundle("resources/bund
les/lagarLottery").getString("Number_of_rejected_cards:_")+
                                    NumberFormat.getInstance().format(filterManager.g
etMaxProgressValue() - filterManager.getPassedCards()),
                                java.util.ResourceBundle.getBundle("resources/bundles
/lagarLottery").getString("Ok"),
                                JOptionPane.INFORMATION_MESSAGE);

                        } catch (IOException ex) {
                            ex.printStackTrace();
                        }
                        finally{
```

```
                        SwingUtilities.invokeLater(new Runnable() {
                            public void run() {
                                pd.exit();
                            }
                        });

                    }
                }
            });
            t.start();
            pd.showProgressDialog();

        } catch (IOException ex) {
            ex.printStackTrace();
        }


    }
}//GEN-LAST:event_runFiltersButtonActionPerformed




    // Variables declaration - do not modify//GEN-BEGIN:variables
    private br.ufsc.inf.lagar.ui.panels.FileChooserLLG fileChooserLoad;
    private br.ufsc.inf.lagar.ui.panels.FileChooserLLG fileChooserSave;
    private br.ufsc.inf.lagar.ui.panels.filter.FilterSelectPanel filterSelectPanel1;
    private javax.swing.JLabel infoLabel;
    private javax.swing.JPanel infoPanel;
    private javax.swing.JPanel loadPanel;
    private javax.swing.JPanel mainPanel;
    private javax.swing.JButton runFiltersButton;
    private javax.swing.JPanel savePanel;
    private br.inf.ufsc.lagar.swingutil.SimpleInternalFrame simpleInternalFrame1;
    private javax.swing.JLabel sp1;
    private javax.swing.JLabel sp2;
    // End of variables declaration//GEN-END:variables

}
/*
 * FilterSelectPanel.java
 *
 * Created on 29 de Março de 2007, 21:13
 */

package br.ufsc.inf.lagar.ui.panels.filter;

import br.ufsc.inf.lagar.filter.Filterable;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.ui.dialog.DecisionDialog;
import br.ufsc.inf.lagar.utils.RTSI;
import java.awt.Color;
import java.awt.Rectangle;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.List;
import java.util.ResourceBundle;
import javax.swing.JScrollPane;
import javax.swing.JTextPane;
import javax.swing.border.Border;
import javax.swing.border.LineBorder;
import javax.swing.border.MatteBorder;

/**
 *
 * @author  lagar
 */
public class FilterSelectPanel extends javax.swing.JPanel {

    private ArrayList<SingleFilterPanel> addedFilters = new
ArrayList<SingleFilterPanel>();

    private SingleFilterPanel selectedFilter;

    private Border defaultBorder = new MatteBorder(0,1,1,1,Color.LIGHT_GRAY);
    private Border selectedBorder = new LineBorder(Color.BLACK, 2);

    /** Creates new form FilterSelectPanel */
    public FilterSelectPanel(){
        initComponents();
        this.buttonBar.setBackground(this.getBackground());

        this.changeFilterButtonsState();
    }

     public void loadAvailableFilters() throws Exception{

        GameConfiguration current = GameManager.getCurrentGame();

        String packageName = "br.ufsc.inf.lagar.filter.gamefilter";

        ArrayList<String> classes = RTSI.find(packageName, Filterable.class);
        for(String clazz: classes){

            clazz = packageName+"."+clazz;
            Filterable filter = (Filterable) Class.forName(clazz).newInstance();

            if(filter.acceptGame(current.getName())){
                this.filterCombo.addItem(filter);
            }
        }

        this.filterCombo.setSelectedIndex(0);

    }

    public void moveUp(SingleFilterPanel sfp){
        int pos = sfp.getPosition() - 1;

        if(pos == 0) return;

        this.getAddedFilters().remove(pos);

        --pos;

        this.getAddedFilters().add(pos, sfp);

        this.rebuildFilterList();
    }

    public void moveDown(SingleFilterPanel sfp){

        int pos = sfp.getPosition() - 1;
        if(pos == this.getAddedFilters().size()-1) return;

        this.getAddedFilters().remove(pos);

        ++pos;

        this.getAddedFilters().add(pos, sfp);

        this.rebuildFilterList();
    }

    public void removeSfp(SingleFilterPanel sfp){

        int pos = sfp.getPosition() - 1;
        this.getAddedFilters().remove(pos);

        if(pos > 0){
            --pos;
        }

        if(pos < this.getAddedFilters().size()){
            this.setSelectedFilter(this.getAddedFilters().get(pos));
        }
        else{
            this.setSelectedFilter(null);
        }

        this.changeFilterButtonsState();

        this.rebuildFilterList();
    }

    public void setSelectedFilter(SingleFilterPanel sfp){

        this.selectedFilter = sfp;
        for(SingleFilterPanel s: addedFilters){
            s.setBorder(this.defaultBorder);
        }

        if(this.selectedFilter != null){
            this.selectedFilter.setBorder(this.selectedBorder);
        }

        this.repaint();

    }

    private void rebuildFilterList(){

        this.selFilterPanel.removeAll();
        for(int i = 0; i < this.getAddedFilters().size(); ++i){

            SingleFilterPanel sfp = this.getAddedFilters().get(i);
            sfp.setPosition(i + 1);

            this.selFilterPanel.add(sfp);
        }

        this.scrollPane.revalidate();
        this.repaint();
    }

    /** This method is called from within the constructor to
```

```
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        fLabel = new javax.swing.JLabel();
        filterCombo = new javax.swing.JComboBox();
        addButton = new javax.swing.JButton();
        mainPanel = new javax.swing.JPanel();
        scrollPane = new javax.swing.JScrollPane();
        selFilterPanel = new javax.swing.JPanel();
        filterButtonsPanel = new javax.swing.JPanel();
        buttonBar = new com.l2fprod.common.swing.JButtonBar();
        helpButton = new javax.swing.JButton();
        removeButton = new javax.swing.JButton();
        upButton = new javax.swing.JButton();
        downButton = new javax.swing.JButton();

        fLabel.setLabelFor(filterCombo);
        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        fLabel.setText(bundle.getString("Filter:")); // NOI18N

        filterCombo.setBackground(new java.awt.Color(255, 255, 255));

        addButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/add.png")));
        addButton.setText(bundle.getString("Add")); // NOI18N
        addButton.setMargin(new java.awt.Insets(2, 2, 2, 2));
        addButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                addButtonActionPerformed(evt);
            }
        });

        scrollPane.setBorder(null);
        scrollPane.setDoubleBuffered(true);
        selFilterPanel.setLayout(new javax.swing.BoxLayout(selFilterPanel,
javax.swing.BoxLayout.Y_AXIS));

        scrollPane.setViewportView(selFilterPanel);

        filterButtonsPanel.setBorder(javax.swing.BorderFactory.createMatteBorder(1,
1, 0, 0, new java.awt.Color(243, 243, 243)));
        buttonBar.setBorder(null);
        helpButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/help.png")));
        helpButton.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
        helpButton.setMargin(new java.awt.Insets(1, 1, 1, 1));
        helpButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                helpButtonActionPerformed(evt);
            }
        });

        removeButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/cancel.png")));
        removeButton.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
        removeButton.setMargin(new java.awt.Insets(1, 1, 1, 1));
        removeButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                removeButtonActionPerformed(evt);
            }
        });

        upButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/up.png")));
        upButton.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
        upButton.setMargin(new java.awt.Insets(1, 1, 1, 1));
        upButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                upButtonActionPerformed(evt);
            }
        });

        downButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/down.png")));
        downButton.setHorizontalTextPosition(javax.swing.SwingConstants.CENTER);
        downButton.setMargin(new java.awt.Insets(1, 1, 1, 1));
        downButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                downButtonActionPerformed(evt);
            }
        });

        org.jdesktop.layout.GroupLayout buttonBarLayout = new
org.jdesktop.layout.GroupLayout(buttonBar);
        buttonBar.setLayout(buttonBarLayout);
        buttonBarLayout.setHorizontalGroup(
            buttonBarLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(helpButton)
            .add(upButton)
            .add(downButton)
            .add(removeButton)
        );
        buttonBarLayout.setVerticalGroup(
            buttonBarLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(buttonBarLayout.createSequentialGroup()
                .add(28, 28, 28)
                .add(helpButton)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(upButton)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(downButton)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(removeButton)
                .addContainerGap(157, Short.MAX_VALUE))
        );

        org.jdesktop.layout.GroupLayout filterButtonsPanelLayout = new
org.jdesktop.layout.GroupLayout(filterButtonsPanel);
        filterButtonsPanel.setLayout(filterButtonsPanelLayout);
        filterButtonsPanelLayout.setHorizontalGroup(
            filterButtonsPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLay
out.LEADING)
            .add(buttonBar, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
        );
        filterButtonsPanelLayout.setVerticalGroup(
            filterButtonsPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLay
out.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, buttonBar,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );

        org.jdesktop.layout.GroupLayout mainPanelLayout = new
org.jdesktop.layout.GroupLayout(mainPanel);
        mainPanel.setLayout(mainPanelLayout);
        mainPanelLayout.setHorizontalGroup(
            mainPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
mainPanelLayout.createSequentialGroup()
                .add(scrollPane, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 448,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(filterButtonsPanel,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
        );
        mainPanelLayout.setVerticalGroup(
            mainPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(scrollPane, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 292,
Short.MAX_VALUE)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, filterButtonsPanel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(fLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 35,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(filterCombo, 0, 369, Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(addButton))
            .add(org.jdesktop.layout.GroupLayout.TRAILING, mainPanel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
                    .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.B
ASELINE)
                        .add(addButton)
                        .add(filterCombo,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(fLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
```

```
    }// </editor-fold>//GEN-END:initComponents

    private void downButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_downButtonActionPerformed
        this.moveDown(this.selectedFilter);
    }//GEN-LAST:event_downButtonActionPerformed

    private void upButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_upButtonActionPerformed
        this.moveUp(this.selectedFilter);
    }//GEN-LAST:event_upButtonActionPerformed

    private void removeButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_removeButtonActionPerformed
        this.removeSfp(this.selectedFilter);
    }//GEN-LAST:event_removeButtonActionPerformed

    private void helpButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_helpButtonActionPerformed

        DecisionDialog dd = new DecisionDialog(null, false);
        JTextPane textPane = new JTextPane();
        textPane.setText(this.selectedFilter.getFilter().getFilterHelp());
        JScrollPane pane = new JScrollPane(textPane,
                JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);

        dd.getMainPanel().add(pane, pane.getClass().getName());
        dd.getButtonsPanel().setVisible(false);
        dd.getTitlePanel().setTitle(this.selectedFilter.getFilter().getShortName());

        dd.setTitle(ResourceBundle.getBundle("resources/bundles/lagarLottery").getStr
ing("Help"));
        dd.setVisible(true);

    }//GEN-LAST:event_helpButtonActionPerformed

    private void addButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_addButtonActionPerformed

        Filterable filterOriginal = (Filterable) this.filterCombo.getSelectedItem();
        Filterable filter;
        try {
            filter = (Filterable) filterOriginal.getClass().newInstance();
        } catch (Exception ex) {
            throw new RuntimeException(ex);
        }

        SingleFilterPanel sfp = new SingleFilterPanel(filter,
this.getAddedFilters().size() + 1, this);
        sfp.setBorder(this.defaultBorder);

        this.getAddedFilters().add(sfp);

        this.setSelectedFilter(sfp);
        this.selFilterPanel.add(sfp);
        this.scrollPane.revalidate();
        this.changeFilterButtonsState();
        this.repaint();
    }//GEN-LAST:event_addButtonActionPerformed

    private void changeFilterButtonsState() {
        boolean enabled = this.selectedFilter != null;
        this.helpButton.setEnabled(enabled);
        this.removeButton.setEnabled(enabled);
        this.upButton.setEnabled(enabled);
        this.downButton.setEnabled(enabled);
    }

    public ArrayList<SingleFilterPanel> getAddedFilters() {
        return addedFilters;
    }



    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton addButton;
    private com.l2fprod.common.swing.JButtonBar buttonBar;
    private javax.swing.JButton downButton;
    private javax.swing.JLabel fLabel;
    private javax.swing.JPanel filterButtonsPanel;
    private javax.swing.JComboBox filterCombo;
    private javax.swing.JButton helpButton;
    private javax.swing.JPanel mainPanel;
    private javax.swing.JButton removeButton;
    private javax.swing.JScrollPane scrollPane;
    private javax.swing.JPanel selFilterPanel;
    private javax.swing.JButton upButton;
    // End of variables declaration//GEN-END:variables

}
/*
 * GameCardIterator.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import java.io.IOException;
import java.util.Iterator;

/**
 *
 * @author lagar
 */
public interface GameCardIterator extends Iterator<GameCard> {

    public long size();
    public int getBallsPerCard();

    public void next(GameCard cardBuffer);

    public void close();

}
/**
 * DrawResult.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import java.text.SimpleDateFormat;
import java.util.*;

/**
 * @author Euclides Pinheiro de Melo
 * Represents one draw result information.
 */
public class GameCard implements Comparable<GameCard>{
    int serial;
    Date date;
    private byte[] balls;

    public GameCard(){

    }

    public GameCard(int serial, Date date, int size) {
        // TODO Auto-generated constructor stub
        this.setBalls(new byte[size]);
        this.date = date;
        this.serial = serial;
    }

    @Override
    public String toString(){

        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");
        StringBuilder sb = new StringBuilder(100);

        sb.append(this.getSerial());
        sb.append(" :: ");
        for(byte i: this.balls){
            sb.append(i);
            sb.append(" - ");
        }

        if(this.date != null)
        sb.append(sdf.format(this.date));

        return sb.toString();
    }

    public void setBall(byte ball, int pos) {
        this.balls[pos] = ball;
    }

    /**
     * @return Returns the balls.
     */
    public byte[] getBalls() {
        return this.balls;
    }

    public void setBalls(byte[] balls) {
        this.balls = balls;
    }

    /**
     * @return Returns the date.
     */
    public Date getDate() {
        return this.date;
    }

    /**
     * @param date The date to set.
     */
    public void setDate(Date date) {
        this.date = date;
    }

    /**
     * @return Returns the serial.
     */
    public int getSerial() {
        return this.serial;
    }

    /**
     * @param serial The serial to set.
     */
    public void setSerial(int serial) {
        this.serial = serial;
    }

    public int compareTo(GameCard o) {
        if(this.serial > o.getSerial()){
            return 1;
        }
        else if(this.serial < o.getSerial()){
            return -1;
        }

        return 0;
```

```java
    }



}
/*
 * GameCardLayout.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import java.awt.geom.Point2D;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;

public class GameCardLayout {

    private double width;
    private double heigth;
    private double ballWidth;
    private double ballHeigth;

    private HashMap<Integer, Point2D.Double> balls = new HashMap<Integer,
Point2D.Double>();
    private List<ExtendedBall> extendedBalls = new ArrayList<ExtendedBall>();
    private ExtendedText extendedText = null;

    /**
     * Creates a new instance of GameCardLayout
     */
    public GameCardLayout() {
    }




    public static class ExtendedBall{

        private String description;
        private boolean required;
        private HashMap<Integer, Point2D.Double> balls = new HashMap<Integer,
Point2D.Double>();

        public boolean isRequired() {
            return required;
        }

        public void setRequired(boolean required) {
            this.required = required;
        }

        public HashMap<Integer, Point2D.Double> getBalls() {
            return balls;
        }

        public void setBalls(HashMap<Integer, Point2D.Double> balls) {
            this.balls = balls;
        }

        public String getDescription() {
            return description;
        }

        public void setDescription(String description) {
            this.description = description;
        }
    }

    public static class ExtendedText{

        private double x;
        private double y;
        private double width;
        private double height;

        public double getX() {
            return x;
        }

        public void setX(double x) {
            this.x = x;
        }

        public double getY() {
            return y;
        }

        public void setY(double y) {
            this.y = y;
        }

        public double getWidth() {
            return width;
        }

        public void setWidth(double width) {
            this.width = width;
        }

        public double getHeight() {
            return height;
        }

        public void setHeight(double height) {
            this.height = height;
        }
    }
    public double getWidth() {
        return width;
    }

    public void setWidth(double width) {
        this.width = width;
    }

    public double getHeigth() {
        return heigth;
    }

    public void setHeigth(double heigth) {
        this.heigth = heigth;
    }

    public double getBallWidth() {
        return ballWidth;
    }

    public void setBallWidth(double ballWidth) {
        this.ballWidth = ballWidth;
    }

    public double getBallHeigth() {
        return ballHeigth;
    }

    public void setBallHeigth(double ballHeigth) {
        this.ballHeigth = ballHeigth;
    }

    public HashMap<Integer, Point2D.Double> getBalls() {
        return balls;
    }

    public void setBalls(HashMap<Integer, Point2D.Double> balls) {
        this.balls = balls;
    }

    public List<ExtendedBall> getExtendedBalls() {
        return extendedBalls;
    }

    public void setExtendedBalls(List<ExtendedBall> extendedBalls) {
        this.extendedBalls = extendedBalls;
    }

    public ExtendedText getExtendedText() {
        return extendedText;
    }

    public void setExtendedText(ExtendedText extendedText) {
        this.extendedText = extendedText;
    }
}


/*
 * GameCardPrinter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.printer;

import br.ufsc.inf.lagar.main.GameCard;
import java.awt.Graphics;
import java.awt.print.PageFormat;
import java.awt.print.Printable;
import java.awt.print.PrinterException;
import java.io.File;
import java.io.IOException;

public abstract class GameCardPrinter implements Printable {

    /** Creates a new instance of GameCardPrinter */
    public GameCardPrinter() {
    }

    public abstract void setFile(File gameCard) throws IOException;

    public abstract int print(Graphics g, PageFormat pf, int pageIndex) throws
PrinterException;

    protected abstract void printExtendedText(GameCard gc, Graphics g, PageFormat
pf, int pageIndex) throws PrinterException;

    protected abstract void printExtendedBalls(GameCard gc, Graphics g, PageFormat
pf, int pageIndex) throws PrinterException;


    protected double cmToPixel(double cm){

        double inch = this.cmToInch(cm);

        return inch * 72;
    }

    protected double cmToInch(double cm){

        return cm / 2.54;
    }

}
/*
 * GameCardsInfo.java
 *
 * Created on 17 de Abril de 2007, 21:58
 */
```

```
package br.ufsc.inf.lagar.ui.panels;

import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.ui.FileExtensionFilter;
import java.io.BufferedWriter;
import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.OutputStreamWriter;
import java.io.StringWriter;
import java.io.Writer;
import java.text.NumberFormat;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.ProgressMonitor;
import javax.swing.SwingUtilities;
import javax.swing.filechooser.FileFilter;

/**
 *
 * @author  lagar
 */
public class GameCardsInfo extends javax.swing.JDialog {

    private String gameFilePath;

    /** Creates new form GameCardsInfo */
    public GameCardsInfo(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
        this.setLocationRelativeTo(parent);
    }

    public void showGameCardsInfo(){

        Thread t = new Thread(new Runnable() {
            public void run() {

                StringWriter sb = new StringWriter();
                try {
                    writeCards(sb, true);
                } catch (IOException ex) {
                    ex.printStackTrace();
                    JOptionPane.showMessageDialog(
                    null,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error_loading_cards"),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
                    return;
                }

                jEditorPane1.setText(sb.toString());

                SwingUtilities.invokeLater(new Runnable() {
                    public void run() {
                        GameCardsInfo.this.setVisible(true);
                    }
                });
            }
        });
        t.start();

    }

    private void writeCards(Writer sb, boolean onScreen) throws IOException{

        final String filePath = this.gameFilePath;

        GenCardIO io = new GenCardIO(new File(filePath));

        try{

            int digits = String.valueOf(io.size()).length();

            if(onScreen){
                if(io.size() > 1000){
                    sb.write(java.util.ResourceBundle.getBundle("resources/bundles/la
garLottery").getString("Too_much_cards_to_show_on_screen._Please_export_to_file_using
_the_button_on_the_menu."));
                    return;
                }
            }

            ProgressMonitor pm = null;
            if(!onScreen){
                pm = new ProgressMonitor(
                    null,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Exporting.."),
                    "",
                    0,
                    (int)io.size());
                pm.setMillisToDecideToPopup(50);
                pm.setMillisToPopup(50);
            }

            NumberFormat nf = NumberFormat.getIntegerInstance();
            nf.setMinimumIntegerDigits(digits);
            NumberFormat nfb = NumberFormat.getIntegerInstance();
            nfb.setMinimumIntegerDigits(2);
            sb.append("<html>");

            GameCard gc = new GameCard(0, null, io.getBallsPerCard());

            int p = 0;
            while(io.hasNext()){

                io.next(gc);

                sb.append("<b>");
                sb.append(nf.format(gc.getSerial()));
                sb.append("</b>");
                sb.append(" -");

                for(byte i: gc.getBalls()){
                    sb.append("- ");
                    sb.append(nfb.format(i));
                    sb.append(" ");
                }

                sb.append("<br>\n");
                if(pm != null){
                    if(pm.isCanceled()){
                        break;
                    }
                    pm.setProgress(++p);

                }
            }
            sb.append("</html>");
        }
        finally{
            io.close();
        }

    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        jPanel1 = new javax.swing.JPanel();
        jScrollPane1 = new javax.swing.JScrollPane();
        jEditorPane1 = new javax.swing.JEditorPane();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu1 = new javax.swing.JMenu();
        jMenuItem1 = new javax.swing.JMenuItem();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        setTitle(bundle.getString("Cards_on_file")); // NOI18N
        jEditorPane1.setContentType("text/html");
        jScrollPane1.setViewportView(jEditorPane1);

        org.jdesktop.layout.GroupLayout jPanel1Layout = new
org.jdesktop.layout.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 574,
Short.MAX_VALUE)
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 303,
Short.MAX_VALUE)
        );

        jMenu1.setText(bundle.getString("Menu")); // NOI18N
        jMenuItem1.setText(bundle.getString("Export_cards_to_file")); // NOI18N
        jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem1ActionPerformed(evt);
            }
        });

        jMenu1.add(jMenuItem1);

        jMenuBar1.add(jMenu1);
        setJMenuBar(jMenuBar1);

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_jMenuItem1ActionPerformed

        JFileChooser chooser = new JFileChooser();
        chooser.setMultiSelectionEnabled(false);
        chooser.setAcceptAllFileFilterUsed(false);

        FileFilter[] filters = chooser.getChoosableFileFilters();

        for (FileFilter f : filters) {
            chooser.removeChoosableFileFilter(f);
        }

        chooser.addChoosableFileFilter(new FileExtensionFilter("html",
                java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").
getString("Html_file_(*.html)")));

        int ret = chooser.showSaveDialog(this);
        if(ret == JFileChooser.APPROVE_OPTION){

            String fp = chooser.getSelectedFile().getPath();
            if(!fp.toLowerCase().endsWith(".html") &&
               !fp.toLowerCase().endsWith(".htm")){

                fp = fp + ".html";
            }

            final File f = new File(fp);

            Thread t = new Thread(new Runnable() {
                public void run() {

                    Writer w = null;
                    try {
                        w = new BufferedWriter(new OutputStreamWriter(new
FileOutputStream(f)));
                        writeCards(w, false);

                        JOptionPane.showMessageDialog(
                                null,
                                java.util.ResourceBundle.getBundle("resources/bundles
/lagarLottery").getString("File_exported"),
                                java.util.ResourceBundle.getBundle("resources/bundles
/lagarLottery").getString("OK"),
                                JOptionPane.INFORMATION_MESSAGE);

                    } catch (Exception ex) {
                        ex.printStackTrace();
                        JOptionPane.showMessageDialog(
                                null,
                                java.util.ResourceBundle.getBundle("resources/bundles
/lagarLottery").getString("Failed"),
                                java.util.ResourceBundle.getBundle("resources/bundles
/lagarLottery").getString("Error"),
                                JOptionPane.ERROR_MESSAGE);
                    }
                    finally {
                        if(w != null){
                            try {w.close();
                            } catch (IOException ex) {

                            }
                        }
                    }
                }
            });
            t.start();

        }

    }//GEN-LAST:event_jMenuItem1ActionPerformed


    public String getGameFilePath() {
        return gameFilePath;
    }

    public void setGameFilePath(String gameFilePath) {
        this.gameFilePath = gameFilePath;
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JEditorPane jEditorPane1;
    private javax.swing.JMenu jMenu1;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JMenuItem jMenuItem1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    // End of variables declaration//GEN-END:variables

}
/*
 * GameChooserPanel.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.ui.panels;

import br.ufsc.inf.lagar.ui.dialog.DecisionDialog;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.Enumeration;
import java.util.List;
import javax.swing.AbstractButton;
import javax.swing.JPanel;
import javax.swing.JRadioButton;

/**
 *
 * @author  lagar
 */
public class GameChooserPanel extends JPanel implements ActionListener{

    private List<RadioPanel> radioButtons = new ArrayList<RadioPanel>();
    private DecisionDialog decisionDialog = null;
    private String selectedGame = null;

    /** Creates new form GameChooserPanel */
    public GameChooserPanel() {
        initComponents();
    }

    public GameChooserPanel(int numOptions){
        this();

    }

    public void addOption(String option){

        RadioPanel rp = new RadioPanel();
        rp.getRadioButton().setText(" "+option+" ");
        this.radioGroup.add(rp.getRadioButton());
        this.bodyPanel.add(rp);
        this.radioButtons.add(rp);

        radioButtons.get(0).getRadioButton().setSelected(true);
    }

    public String getSelectedGame(){

        this.decisionDialog = new DecisionDialog(null, true);
        this.decisionDialog.getOkButton().addActionListener(this);
        this.decisionDialog.getCancelButton().addActionListener(this);
        this.decisionDialog.getMainPanel().add(this, "gameChooser");
        this.decisionDialog.getTitlePanel().setTitle(java.util.ResourceBundle.getBund
le("resources/bundles/lagarLottery").getString("Choose_a_game"));

        this.decisionDialog.pack();
        this.decisionDialog.setVisible(true);

        return this.selectedGame.trim();
    }


    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        radioGroup = new javax.swing.ButtonGroup();
        bodyPanel = new javax.swing.JPanel();

        bodyPanel.setLayout(new javax.swing.BoxLayout(bodyPanel,
javax.swing.BoxLayout.Y_AXIS));

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .add(27, 27, 27)
                .add(bodyPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 361,
Short.MAX_VALUE)
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(bodyPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 276,
Short.MAX_VALUE)
                .addContainerGap())
        );
    }// </editor-fold>//GEN-END:initComponents

    public void actionPerformed(ActionEvent e) {

        String action = e.getActionCommand();

        if(action.equals(this.decisionDialog.getOkButton().getText())){

            Enumeration<AbstractButton> buttons = this.radioGroup.getElements();
            while(buttons.hasMoreElements()){
                AbstractButton b = buttons.nextElement();
                if(this.radioGroup.isSelected(b.getModel())){
                    this.selectedGame = b.getText();
                    break;
                }
            }
        }
        this.decisionDialog.dispose();

    }


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JPanel bodyPanel;
    private javax.swing.ButtonGroup radioGroup;
    // End of variables declaration//GEN-END:variables
```

```
}
/*
 * GameConfiguration.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import br.ufsc.inf.lagar.main.GameResults;
import br.ufsc.inf.lagar.printer.DefaultGameCardPrinter;
import br.ufsc.inf.lagar.printer.GameCardPrinter;

public class GameConfiguration {

    private String name;
    private int numbersPerCard;
    private int minNumbersToChoose;
    private int maxNumbersToChoose;
    private int numbersPerDrawing;
    private int numbersPerLine;

    private GameCardLayout cardLayout;

    private GameResults gameResults;

    private GameCardPrinter gameCardPrinter = new DefaultGameCardPrinter(); //
default

    /** Creates a new instance of GameConfiguration */
    public GameConfiguration() {
    }

    public GameResultIterator getGameCardIterator(int begin, int end, int jump){

        return new GameResultIterator(this.getGameResults(), begin, end, jump);
    }

    public String getName() {
        return name;
    }

    protected void setName(String name) {
        this.name = name;
    }

    public GameResults getGameResults() {
        return gameResults;
    }

    protected void setGameResults(GameResults gameResults) {
        this.gameResults = gameResults;
    }

    public int getNumbersPerCard() {
        return numbersPerCard;
    }

    public void setNumbersPerCard(int numbersPerCard) {
        this.numbersPerCard = numbersPerCard;
    }

    public int getMinNumbersToChoose() {
        return minNumbersToChoose;
    }

    public void setMinNumbersToChoose(int minNumbersToChoose) {
        this.minNumbersToChoose = minNumbersToChoose;
    }

    public int getMaxNumbersToChoose() {
        return maxNumbersToChoose;
    }

    public void setMaxNumbersToChoose(int maxNumbersToChoose) {
        this.maxNumbersToChoose = maxNumbersToChoose;
    }

    @Override
    public String toString(){
        StringBuilder sb = new StringBuilder(50);
        sb.append("Name: "+this.name); sb.append("\n");
        sb.append("NumbersPerCard: "+this.numbersPerCard); sb.append("\n");
        sb.append("NumbersPerDrawing: "+this.numbersPerDrawing); sb.append("\n");
        sb.append("minNumbersChoose: "+this.minNumbersToChoose); sb.append("\n");
        sb.append("maxNumbersChoose: "+this.maxNumbersToChoose); sb.append("\n");

        return sb.toString();
    }

    public int getNumbersPerDrawing() {
        return numbersPerDrawing;
    }

    public void setNumbersPerDrawing(int numbersPerDrawing) {
        this.numbersPerDrawing = numbersPerDrawing;
    }

    public GameCardLayout getCardLayout() {
        return cardLayout;
    }

    public void setCardLayout(GameCardLayout cardLayout) {
        this.cardLayout = cardLayout;
    }

    public GameCardPrinter getGameCardPrinter() {
        return gameCardPrinter;
    }
    public void setGameCardPrinter(GameCardPrinter gameCardPrinter) {
        this.gameCardPrinter = gameCardPrinter;
    }

    public int getNumbersPerLine() {
        return numbersPerLine;
    }

    public void setNumbersPerLine(int numbersPerLine) {
        this.numbersPerLine = numbersPerLine;
    }

}
/*
 * GameConfigurationParser.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import br.ufsc.inf.lagar.printer.GameCardPrinter;
import br.ufsc.inf.lagar.resultloader.ResultLoader;
import br.ufsc.inf.lagar.utils.LocalEntityResolver;
import java.awt.geom.Point2D;
import java.io.IOException;
import java.io.InputStream;
import java.net.MalformedURLException;
import java.util.List;
import org.jdom.Document;
import org.jdom.Element;
import org.jdom.JDOMException;
import org.jdom.input.SAXBuilder;

public class GameConfigurationParser {

    /** Creates a new instance of GameConfigurationParser */
    public GameConfigurationParser() {
    }

    public GameConfiguration parse(InputStream in) throws ClassNotFoundException,
InstantiationException, IllegalAccessException, MalformedURLException,
JDOMException, IOException {

        GameConfiguration gc = new GameConfiguration();

        SAXBuilder saxBuilder = new SAXBuilder(true);
        saxBuilder.setEntityResolver(new LocalEntityResolver("lagarLottery gametype
dtd", "/gametypes/gametype.dtd"));
        Document doc = saxBuilder.build(in);

        Element root = doc.getRootElement();

        gc.setName(root.getChildTextTrim("name"));
        gc.setNumbersPerCard(Integer.parseInt(root.getChildTextTrim("numbersPerCard")
));
        gc.setNumbersPerDrawing(Integer.parseInt(root.getChildTextTrim("numbersPerDra
wing")));
        gc.setMinNumbersToChoose(Integer.parseInt(root.getChildTextTrim("minNumbersTo
Choose")));
        gc.setMaxNumbersToChoose(Integer.parseInt(root.getChildTextTrim("maxNumbersTo
Choose")));
        gc.setNumbersPerLine(Integer.parseInt(root.getChildTextTrim("numbersPerLine")
));

        //result loader
        Element resultLoader = root.getChild("resultLoader");

        String className = resultLoader.getChildTextTrim("class");
        String resultPath = resultLoader.getChildTextTrim("resultPath");

        Class rlClazz = Class.forName(className);
        ResultLoader rl = (ResultLoader) rlClazz.newInstance();
        rl.setResultPath(resultPath);
        GameResults gameResults = new GameResults();
        gameResults.setResultLoader(rl);

        gc.setGameResults(gameResults);

        //game card layout
        GameCardLayout gcl = new GameCardLayout();
        Element cardLayout = root.getChild("cardLayout");

        String cardPrinterclazz =
cardLayout.getChild("gamePrinter").getAttributeValue("class").trim();
        Class cpClazz = Class.forName(cardPrinterclazz);
        GameCardPrinter gameCardPrinter = (GameCardPrinter) cpClazz.newInstance();
        gc.setGameCardPrinter(gameCardPrinter);

        gcl.setWidth(Double.parseDouble(cardLayout.getChildTextTrim("width")));
        gcl.setHeigth(Double.parseDouble(cardLayout.getChildTextTrim("height")));
        gcl.setBallWidth(Double.parseDouble(cardLayout.getChildTextTrim("ballWidth"))
);
        gcl.setBallHeigth(Double.parseDouble(cardLayout.getChildTextTrim("ballHeight"
)));

        List<Element> balls = cardLayout.getChildren("ball");
        for(Element b: balls){
            Integer n = Integer.valueOf(b.getAttributeValue("n"));
            double x = Double.parseDouble(b.getAttributeValue("x"));
            double y = Double.parseDouble(b.getAttributeValue("y"));

            Point2D.Double point = new Point2D.Double(x,y);

            gcl.getBalls().put(n, point);
        }

        //extended balls
        List<Element> extBalls = cardLayout.getChildren("extendedBall");
        for(Element exBall: extBalls){
            GameCardLayout.ExtendedBall gcex = new GameCardLayout.ExtendedBall();

            String desc = exBall.getChildTextTrim("description");
            boolean required =
Boolean.parseBoolean(exBall.getChildTextTrim("required"));

            gcex.setDescription(desc);
            gcex.setRequired(required);

            List<Element> bbx = exBall.getChildren("ball");

            for(Element bx: bbx){
            Integer n = Integer.valueOf(bx.getAttributeValue("n"));
            double x = Double.parseDouble(bx.getAttributeValue("x"));
            double y = Double.parseDouble(bx.getAttributeValue("y"));

            Point2D.Double point = new Point2D.Double(x,y);

            gcex.getBalls().put(n, point);

            gcl.getExtendedBalls().add(gcex);
        }
        }

        Element extText = cardLayout.getChild("extendedText");
        if(extText != null){
            GameCardLayout.ExtendedText xt = new GameCardLayout.ExtendedText();
            xt.setX(Double.parseDouble(extText.getAttributeValue("x")));
            xt.setY(Double.parseDouble(extText.getAttributeValue("y")));
            xt.setWidth(Double.parseDouble(extText.getAttributeValue("width")));
            xt.setHeight(Double.parseDouble(extText.getAttributeValue("height")));

            gcl.setExtendedText(xt);
        }

        gc.setCardLayout(gcl);

        return gc;
    }


}
/*
 * GameFactoryTest.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */
```

```
package br.ufsc.inf.lagar.core.base;

import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import junit.framework.*;
import br.ufsc.inf.lagar.utils.RTSI;
import java.util.ArrayList;
import java.util.List;

/**
 *
 * @author lagar
 */
public class GameFactoryTest extends TestCase {


    public void testLoadGames(){

        List<GameConfiguration> availableGames = GameManager.getAvailableGames();

        assertNotNull(availableGames);
        assertTrue(availableGames.size() != 0);

        for(GameConfiguration gc: availableGames){
            System.out.println(gc);
        }
    }
}
/*
 * GameManager.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import br.ufsc.inf.lagar.main.GameConfigurationParser;
import br.ufsc.inf.lagar.utils.RTSI;
import java.io.InputStream;
import java.util.ArrayList;
import java.util.List;

public class GameManager {

    private static List<GameConfiguration> availableGames = null;
    private static GameConfiguration currentGame = null;
```

```
/**
 * Creates a new instance of GameManager
 */
private GameManager() {

}

private static void loadGames(){

    GameManager.availableGames = new ArrayList<GameConfiguration>();

    String packageName = "/gametypes";
    List<String> xmls = RTSI.findFiles(packageName, "xml");
    GameConfigurationParser gcp = new GameConfigurationParser();

    for(String xml: xmls){
        try {
            if(!xml.startsWith("/")){
                xml = "/"+xml;
            }
            if(!xml.startsWith(packageName)){
                xml = packageName + "/" + xml;
            }
            System.out.println(xml);
            InputStream in = GameManager.class.getResourceAsStream(xml);
            System.out.println(in);
            GameConfiguration gc = gcp.parse(in);

            GameManager.availableGames.add(gc);

        } catch (Exception ex) {
            throw new RuntimeException("Error in game factory load games", ex);
        }

    }
}

public static List<GameConfiguration> getAvailableGames() {

    if(GameManager.availableGames == null){
        loadGames();
    }

    return availableGames;
}


public static void setCurrentGame(GameConfiguration aCurrentGame) {
    currentGame = aCurrentGame;
}

public static void setCurrentGame(String gameName) {

    for(GameConfiguration gc: availableGames){
        if(gc.getName().equals(gameName)){
            currentGame = gc;
            break;
        }
    }

    if(currentGame == null)
    throw new IllegalArgumentException(gameName + " is not a valid gameName");
}
```

```
    public static GameConfiguration getCurrentGame() {
        return currentGame;
    }


}
/*
 * GameResultIterator.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import java.util.Iterator;
import java.util.List;

public class GameResultIterator implements GameCardIterator{

    private GameResults gr;
    private List<GameCard> games;
    private int position;

    /**
     * Creates a new instance of GameResultIterator
     */
    public GameResultIterator(GameResults gr, int begin, int end, int jump) {
        this.gr = gr;
        this.games = gr.getSubset(begin, end, jump);
    }

    public boolean hasNext() {
        return position < this.games.size();
    }

    public GameCard next() {

        GameCard gc = this.games.get(position);
        ++position;

        return gc;
    }

    public void next(GameCard cardBuffer) {
        GameCard next = this.next();

        cardBuffer.setDate(next.getDate());
        cardBuffer.setSerial(next.getSerial());
        cardBuffer.setBalls(next.getBalls());
    }

    public long size(){

        return this.games.size();
    }

    public void remove() {
        throw new UnsupportedOperationException("Not allowed to remove");
    }

    public int getBallsPerCard() {
        return this.games.get(0).getBalls().length;
    }

    public void close() {
    }


}
/*
 * GameResults.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import br.ufsc.inf.lagar.resultloader.ResultLoader;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;

public class GameResults {

    private ResultLoader resultLoader;
    private List<GameCard> loadedResults = new ArrayList<GameCard>();

    /** Creates a new instance of GameResults */
    public GameResults() {
    }

    public ResultLoader getResultLoader() {
        return resultLoader;
    }
    public void setResultLoader(ResultLoader resultLoader) {
        this.resultLoader = resultLoader;
    }

    public List<GameCard> getLoadedResults() {
        return loadedResults;
    }

    public void loadResults() throws IOException{
        this.loadedResults = this.resultLoader.loadResults();
        Collections.sort(this.loadedResults);
    }

    public List<GameCard> getSubset(int begin, int end, int jump){

        //special case: all results
        if(begin == 1 && end == this.loadedResults.get(this.loadedResults.size()-
1).serial && jump == 1){
            return this.loadedResults;
        }

        ArrayList<GameCard> subset = new ArrayList<GameCard>();

        for(int i = begin - 1; i < end; i+= jump){
            subset.add(this.loadedResults.get(i));
        }
        return subset;
    }

}
/*
 * GenCardIO.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.generator;

import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameCardIterator;
import br.ufsc.inf.lagar.main.LagarLotteryGlobals;
import java.io.DataInputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStream;
import java.util.zip.ZipEntry;
import java.util.zip.ZipInputStream;

public class GenCardIO implements GameCardIterator {

    private DataInputStream dis;
    private GeneratorHeader generatorHeader;
    private long lastCardRead;

    /**
     * Creates a new instance of GenCardIO
     */
    public GenCardIO(File llgFile) throws IOException{

        this.generatorHeader = GeneratorHeader.fetchGeneratorHeader(llgFile);
        this.dis = this.openGeneratedCardsStream(llgFile);
    }

    public boolean skipToCard(GameCard cardBuffer, long cardNumber) throws
IOException{

        if(cardNumber <= this.lastCardRead){
            //throw new IllegalArgumentException("cardNumber must be greater than
lastCardRead");
            return false;
        }

        long alreadySkip = this.lastCardRead *
this.generatorHeader.getBallsPerCard();

        this.lastCardRead = cardNumber - 1;

        long bytesToSkip = (cardNumber - 1) * this.generatorHeader.getBallsPerCard();
        bytesToSkip = bytesToSkip - alreadySkip;

        while(bytesToSkip != 0){
            long n = this.dis.skip(bytesToSkip);
            bytesToSkip -= n;
        }

        this.next(cardBuffer);

        return true;
    }

    public void close(){
        try {
            this.dis.close();
        } catch (IOException ex) {
            ex.printStackTrace();
        }
    }

    private DataInputStream openGeneratedCardsStream(File llgFile) throws
IOException{

        FileInputStream fin = new FileInputStream(llgFile);

        ZipInputStream zin = new ZipInputStream(fin);

        DataInputStream dis = new DataInputStream(zin);

        ZipEntry e = null;
        while( (e = zin.getNextEntry()) != null ){
            if(e.getName().equals(LagarLotteryGlobals.GENERATED_CARDS_NAME)){
                break;
            }
        }

        if(e == null) throw new IOException("No generated cards found.");

        return dis;
    }

    public long size() {
        return this.generatorHeader.getNumberOfCards();
    }

    public boolean hasNext() {
        return this.lastCardRead < this.size();
    }

    public GameCard next() {

        GameCard g = new GameCard(0,null,this.getBallsPerCard());
        this.next(g);

        return g;
    }

    public void remove() {
        throw new UnsupportedOperationException("Not allowed to remove");
    }

    public void next(GameCard cardBuffer){

        ++lastCardRead;

        if(cardBuffer.getBalls() == null){
            cardBuffer.setBalls(new byte[this.getBallsPerCard()]);
        }

        try {
            this.dis.readFully(cardBuffer.getBalls());
            cardBuffer.setSerial((int)lastCardRead);
        } catch (IOException ex) {
            throw new RuntimeException(ex);
        }

    }

    public int getBallsPerCard() {
        return this.generatorHeader.getBallsPerCard();
    }

}
/*
 * GenerateBallPositions.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.utils;

import java.text.NumberFormat;
import java.util.Locale;
import java.util.regex.Matcher;

public class GenerateBallPositions {

    /** Creates a new instance of GenerateBallPositions */
    public GenerateBallPositions() {
    }

    public void generatePositions(int firstBall, int lastBall, int endLine, double
xStart,
                                    double yStart, double xInc, double yInc){

        NumberFormat f = NumberFormat.getNumberInstance(Locale.US);
        f.setMaximumFractionDigits(2);

        String s = "<ball n=\"?\" x=\"?\" y=\"?\"/>";

        double startY = yStart;
        int m = 0;
        for(int i = firstBall; i <= lastBall; ++i){


            String c = s.replaceFirst("\\?", String.valueOf(i));

            double incX = xStart + (xInc * m);

            c = c.replaceFirst("\\?", f.format(incX));

            c = c.replaceFirst("\\?", f.format(startY));

            System.out.println(c);

            if(++m % 10 == 0){

                m = 0;
                startY = yStart + (yInc * (i/10));
            }
        }
    }

    public static void main(String[] args){

        new GenerateBallPositions().generatePositions(
                1,80,10,1.2,5.82,0.65,0.325);
    }

}
/*
 * GeneratorAllCombPanel.java
 *
 * Created on 28 de Fevereiro de 2007, 19:42
 */

package br.ufsc.inf.lagar.generator.impl;

import br.ufsc.inf.lagar.generator.GeneratorInterface;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.generator.AllCombinationGeneratorService;
import br.ufsc.inf.lagar.ui.dialog.ProgressDialog;
import br.ufsc.inf.lagar.ui.panels.*;
import java.awt.Color;
import java.io.IOException;
import java.text.NumberFormat;
import java.util.ArrayList;
import java.util.Arrays;
import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.SwingUtilities;
import org.apache.commons.lang.StringUtils;

/**
 *
 * @author  lagar
 */
public class GeneratorAllCombPanel extends javax.swing.JPanel implements
GeneratorInterface{

    /**
     * Creates new form GeneratorAllCombPanel
     */
    public GeneratorAllCombPanel() {
        initComponents();

        this.setNumbersAreaEnabled(false);

        GameConfiguration gc = GameManager.getCurrentGame();
        for(int i = gc.getMinNumbersToChoose(); i <= gc.getMaxNumbersToChoose();
++i){
            this.numbersPerCardCombo.addItem(i);
        }

        this.fileChooserLLG1.setMode(FileChooserLLG.ChooserMode.SAVE_MODE);

    }

    private void setNumbersAreaEnabled(boolean enabled){

        if(enabled){
            this.numbersArea.setBackground(Color.WHITE);
        }
        else{
            this.numbersArea.setBackground(this.getBackground());
        }
        this.numbersArea.setEditable(enabled);
    }


    private void generateCards() {

        try {
            int[] numbers = null;
            if(this.allPossibleRadio.isSelected()){
                int maxNumbers = GameManager.getCurrentGame().getNumbersPerCard();
                numbers = new int[maxNumbers];
                for(int i = 0; i < numbers.length; ++i){
                    numbers[i] = i + 1;
                }
            }
            else{
                numbers = this.parseUserNumbers(this.numbersArea.getText());

                if(numbers.length <
(Integer)this.numbersPerCardCombo.getSelectedItem()){
                    JOptionPane.showMessageDialog(
                            null,
                            java.util.ResourceBundle.getBundle("resources/bundles/lag
arLottery").getString("You_typed_few_numbers"),
                            java.util.ResourceBundle.getBundle("resources/bundles/lag
arLottery").getString("Error"),
                            JOptionPane.ERROR_MESSAGE);
                    return;
                }

                StringBuilder sb = new StringBuilder();
                sb.append(java.util.ResourceBundle.getBundle("resources/bundles/lagar
Lottery").getString("You_typed_the_numbers_below"));
                sb.append("\n");

                NumberFormat nf = NumberFormat.getNumberInstance();
                nf.setMinimumIntegerDigits(2);

                for(int i = 0; i < numbers.length; ++i){
                    sb.append(nf.format(numbers[i]));
                    sb.append(" - ");
                    if((i + 1) % 10 == 0){
                        sb.delete(sb.length()-3, sb.length());
                        sb.append("\n");
                    }
                }
                if(sb.charAt(sb.length()-1) == ' '){
                    sb.delete(sb.length()-3, sb.length());
                    sb.append("\n");
                }
                sb.append(java.util.ResourceBundle.getBundle("resources/bundles/lagar
Lottery").getString("Is_it_correct?"));

                if(JOptionPane.showConfirmDialog(null, sb.toString()) !=
JOptionPane.YES_OPTION){
                    return;
                }

            }

            final AllCombinationGeneratorService service = new
AllCombinationGeneratorService(
                    this.fileChooserLLG1.getFilePath().trim(),
                    numbers,
                    (Integer)this.numbersPerCardCombo.getSelectedItem());

            final ProgressDialog pd = new ProgressDialog(null, true, service);
            pd.setLocationRelativeTo(null);
            Thread t = new Thread(new Runnable() {
                public void run() {
                    try {
                        service.generateAllCombinations();
                        JOptionPane.showMessageDialog(
                             null,
                            java.util.ResourceBundle.getBundle("resources/bundles/lag
arLottery").getString("Cards_generated_successfully"),
                            java.util.ResourceBundle.getBundle("resources/bundles/lag
arLottery").getString("Ok"),
                            JOptionPane.INFORMATION_MESSAGE);

                    } catch (IOException ex) {
                        ex.printStackTrace();
                    }
                    finally{
                        SwingUtilities.invokeLater(new Runnable() {
                            public void run() {
                                pd.exit();
                            }
                        });

                    }
                }
            });
            t.start();
            pd.showProgressDialog();

        } catch (NumberFormatException ex) {
            ex.printStackTrace();
        } catch (IOException ex) {
            ex.printStackTrace();
        } catch(IllegalArgumentException iaex){
            JOptionPane.showMessageDialog(
                    null,
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Error
_parsing_numbers._Make_sure_your_numbers_are_typed_correctly,_are_inside_the_valid_ra
nge_area_and_try_again."),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
        }

    }

    private int[] parseUserNumbers(String userData) throws IllegalArgumentException{

        StringBuilder sb = new StringBuilder(userData);
        int index = sb.indexOf("\n");
        while(index != -1){
            sb.replace(index, index + 1, "-");
            index = sb.indexOf("\n");
        }

        int minNumber = 1;
        int maxNumber = GameManager.getCurrentGame().getNumbersPerCard();

        String[] parts = sb.toString().trim().split("-");
        ArrayList<Integer> numbers = new ArrayList(parts.length);
        for(int i = 0; i < parts.length; ++i){
            String s = StringUtils.deleteWhitespace(parts[i]);
            try{
                if(s == null || s.equals("")) continue;
                Integer n = Integer.valueOf(s);

                if(numbers.contains(n)) continue;

                numbers.add(n);

                if(n > maxNumber || n < minNumber){
                    throw new IllegalArgumentException();
                }
            }
            catch(Exception ex){
                throw new IllegalArgumentException(ex);
            }

        }

        int[] nums = new int[numbers.size()];
        for(int i = 0; i < nums.length; ++i){
            nums[i] = numbers.get(i);
        }

        Arrays.sort(nums);

        return nums;
    }

    private boolean validaForm() {

        StringBuilder errors = new StringBuilder();

        if(this.numbersRadio.isSelected()){
            if(this.numbersArea.getText().equals("")){
                errors.append(java.util.ResourceBundle.getBundle("resources/bundles/l
agarLottery").getString("-_You_must_type_the_numbers."));
                errors.append("\n");
            }
        }

        String file = this.fileChooserLLG1.getFilePath();
        if(file == null || file.equals("")){
            errors.append(java.util.ResourceBundle.getBundle("resources/bundles/lagar
Lottery").getString("-_You_must_choose_a_file_to_save."));
            errors.append("\n");
        }


        if(errors.length() != 0){
            JOptionPane.showMessageDialog(
                    null, errors.toString(),
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Error
"), JOptionPane.ERROR_MESSAGE);
            return false;
        }

        return true;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
```

```
BEGIN:initComponents
    private void initComponents() {
        radioGroup = new javax.swing.ButtonGroup();
        simpleInternalFrame1 = new br.inf.ufsc.lagar.swingutil.SimpleInternalFrame();
        topPanel = new javax.swing.JPanel();
        textLabel = new javax.swing.JLabel();
        numbersPanel = new javax.swing.JPanel();
        allPossibleRadio = new javax.swing.JRadioButton();
        numbersRadio = new javax.swing.JRadioButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        numbersArea = new javax.swing.JTextArea();
        numbersPerCardLabel = new javax.swing.JLabel();
        numbersPerCardCombo = new javax.swing.JComboBox();
        hintLabel = new javax.swing.JLabel();
        filePanel = new javax.swing.JPanel();
        fileChooserLLG1 = new br.ufsc.inf.lagar.ui.panels.FileChooserLLG();
        bottomPanel = new javax.swing.JPanel();
        generateButton = new javax.swing.JButton();
        jLabel4 = new javax.swing.JLabel();
        jLabel5 = new javax.swing.JLabel();

        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        simpleInternalFrame1.setTitle(bundle.getString("Generators")); // NOI18N

        textLabel.setText(bundle.getString("This_module_generates_all_combinations"))
; // NOI18N

        org.jdesktop.layout.GroupLayout topPanelLayout = new
org.jdesktop.layout.GroupLayout(topPanel);
        topPanel.setLayout(topPanelLayout);
        topPanelLayout.setHorizontalGroup(
            topPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADIN
G)
            .add(topPanelLayout.createSequentialGroup()
                .addContainerGap()
                .add(textLabel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 481,
Short.MAX_VALUE)
                .addContainerGap())
        );
        topPanelLayout.setVerticalGroup(
            topPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADIN
G)
            .add(textLabel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 76,
Short.MAX_VALUE)
        );

        numbersPanel.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.ge
tString("Numbers"))); // NOI18N
        radioGroup.add(allPossibleRadio);
        allPossibleRadio.setSelected(true);
        allPossibleRadio.setText(bundle.getString("Generate_all_possible_combinations
_(all_numbers)")); // NOI18N
        allPossibleRadio.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0,
0, 0));
        allPossibleRadio.setMargin(new java.awt.Insets(0, 0, 0, 0));
        allPossibleRadio.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                allPossibleRadioActionPerformed(evt);
            }
        });

        radioGroup.add(numbersRadio);
        numbersRadio.setText(bundle.getString("Generate_cards_only_with_the_numbers_b
elow:")); // NOI18N
        numbersRadio.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0,
0));
        numbersRadio.setMargin(new java.awt.Insets(0, 0, 0, 0));
        numbersRadio.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                numbersRadioActionPerformed(evt);
            }
        });

        jScrollPane1.setHorizontalScrollBarPolicy(javax.swing.ScrollPaneConstants.HOR
IZONTAL_SCROLLBAR_NEVER);
        jScrollPane1.setVerticalScrollBarPolicy(javax.swing.ScrollPaneConstants.VERTI
CAL_SCROLLBAR_ALWAYS);
        numbersArea.setColumns(20);
        numbersArea.setLineWrap(true);
        numbersArea.setRows(5);
        numbersArea.setWrapStyleWord(true);
        jScrollPane1.setViewportView(numbersArea);

        numbersPerCardLabel.setText(bundle.getString("Numbers_per_card:")); // NOI18N

        hintLabel.setForeground(new java.awt.Color(153, 153, 153));
        hintLabel.setText(bundle.getString("<html>_Type_the_numbers_separated_by___'<
b>-</b>'____Ex:_1_-_2_-_3_-_4_-_5_-_6_</html>")); // NOI18N

        org.jdesktop.layout.GroupLayout numbersPanelLayout = new
org.jdesktop.layout.GroupLayout(numbersPanel);
        numbersPanel.setLayout(numbersPanelLayout);
        numbersPanelLayout.setHorizontalGroup(
            numbersPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
            .add(numbersPanelLayout.createSequentialGroup()
                .add(numbersPanelLayout.createParallelGroup(org.jdesktop.layout.Group
Layout.LEADING)
                    .add(numbersPanelLayout.createSequentialGroup()
                        .addContainerGap()
                        .add(numbersPanelLayout.createParallelGroup(org.jdesktop.layo
ut.GroupLayout.LEADING)
                            .add(allPossibleRadio)
                            .add(numbersRadio)))
                    .add(numbersPanelLayout.createSequentialGroup()
                        .addContainerGap()
                        .add(numbersPerCardLabel)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(numbersPerCardCombo,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 72,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                    .add(numbersPanelLayout.createSequentialGroup()
                        .add(29, 29, 29)
                        .add(numbersPanelLayout.createParallelGroup(org.jdesktop.layo
ut.GroupLayout.LEADING)
                            .add(hintLabel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 454, Short.MAX_VALUE)
                            .add(jScrollPane1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 454, Short.MAX_VALUE))))
                .addContainerGap())
        );
        numbersPanelLayout.setVerticalGroup(
            numbersPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
            .add(numbersPanelLayout.createSequentialGroup()
                .add(numbersPanelLayout.createParallelGroup(org.jdesktop.layout.Group
Layout.BASELINE)
                    .add(numbersPerCardLabel)
                    .add(numbersPerCardCombo,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(allPossibleRadio)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(numbersRadio)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 90,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(hintLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 15,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );

        filePanel.setBorder(javax.swing.BorderFactory.createTitledBorder("File to
save"));

        org.jdesktop.layout.GroupLayout filePanelLayout = new
org.jdesktop.layout.GroupLayout(filePanel);
        filePanel.setLayout(filePanelLayout);
        filePanelLayout.setHorizontalGroup(
            filePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(filePanelLayout.createSequentialGroup()
                .addContainerGap()
                .add(fileChooserLLG1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
471, Short.MAX_VALUE)
                .addContainerGap())
        );
        filePanelLayout.setVerticalGroup(
            filePanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(filePanelLayout.createSequentialGroup()
                .add(fileChooserLLG1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
26, Short.MAX_VALUE)
                .addContainerGap())
        );

        generateButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/runprog.png")));
        generateButton.setText(bundle.getString("Generate")); // NOI18N
        generateButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                generateButtonActionPerformed(evt);
            }
        });

        org.jdesktop.layout.GroupLayout bottomPanelLayout = new
org.jdesktop.layout.GroupLayout(bottomPanel);
        bottomPanel.setLayout(bottomPanelLayout);
        bottomPanelLayout.setHorizontalGroup(
            bottomPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEA
DING)
            .add(bottomPanelLayout.createSequentialGroup()
                .addContainerGap()
                .add(jLabel4, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 174,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(generateButton)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jLabel5, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 174,
Short.MAX_VALUE)
                .addContainerGap())
        );
        bottomPanelLayout.setVerticalGroup(
            bottomPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEA
DING)
            .add(jLabel4, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 23,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
            .add(jLabel5, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 23,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
            .add(generateButton)
        );

        bottomPanelLayout.linkSize(new java.awt.Component[] {jLabel4, jLabel5},
org.jdesktop.layout.GroupLayout.VERTICAL);

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(simpleInternalFrame1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
505, Short.MAX_VALUE)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, filePanel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(bottomPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(topPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, numbersPanel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(simpleInternalFrame1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(topPanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(numbersPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(filePanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(bottomPanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
    }// </editor-fold>//GEN-END:initComponents

    private void numbersRadioActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_numbersRadioActionPerformed
```

```
        this.setNumbersAreaEnabled(true);
    }//GEN-LAST:event_numbersRadioActionPerformed

    private void allPossibleRadioActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_allPossibleRadioActionPerformed
        this.setNumbersAreaEnabled(false);
    }//GEN-LAST:event_allPossibleRadioActionPerformed

    private void generateButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_generateButtonActionPerformed
        boolean ok = this.validaForm();

        if(ok) this.generateCards();
    }//GEN-LAST:event_generateButtonActionPerformed

    public JPanel getGeneratorPanel() {
        return this;
    }

    public String getGeneratorName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/generator/impl/generatorTr").ge
tString("All_combination_generator");
    }



    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JRadioButton allPossibleRadio;
    private javax.swing.JPanel bottomPanel;
    private br.ufsc.inf.lagar.ui.panels.FileChooserLLG fileChooserLLG1;
    private javax.swing.JPanel filePanel;
    private javax.swing.JButton generateButton;
    private javax.swing.JLabel hintLabel;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextArea numbersArea;
    private javax.swing.JPanel numbersPanel;
    private javax.swing.JComboBox numbersPerCardCombo;
    private javax.swing.JLabel numbersPerCardLabel;
    private javax.swing.JRadioButton numbersRadio;
    private javax.swing.ButtonGroup radioGroup;
    private br.inf.ufsc.lagar.swingutil.SimpleInternalFrame simpleInternalFrame1;
    private javax.swing.JLabel textLabel;
    private javax.swing.JPanel topPanel;
    // End of variables declaration//GEN-END:variables

}
/*
 * GeneratorHeader.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.generator;

import br.ufsc.inf.lagar.main.LagarLotteryGlobals;
import java.beans.XMLDecoder;
import java.beans.XMLEncoder;
import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.OutputStream;
import java.util.Date;
import java.util.zip.ZipEntry;
import java.util.zip.ZipInputStream;

public class GeneratorHeader {

    private int version;
    private String gameName;
    private long numberOfCards;
    private int ballsPerCard;
    private Date date;
    private int[] numbers;


    /** Creates a new instance of GeneratorHeader */
    public GeneratorHeader() {
        this.setVersion(1);
        this.setDate(new Date());
    }

    public static GeneratorHeader fetchGeneratorHeader(File llgFile) throws
IOException{

        FileInputStream fin = new FileInputStream(llgFile);

        ZipInputStream zin = new ZipInputStream(fin);

        ZipEntry e = null;
        while( (e = zin.getNextEntry()) != null ){
            if(e.getName().equals(LagarLotteryGlobals.GENERATED_HEADER_NAME)){
                break;
            }
        }
        GeneratorHeader header = GeneratorHeader.loadFromStream(zin);

        zin.closeEntry();
        zin.close();

        return header;
    }

    public static GeneratorHeader loadFromStream(InputStream is) throws IOException{
        byte[] buff = new byte[8192];
        ByteArrayOutputStream bos = new ByteArrayOutputStream(8192);
        int n = 0;
        while((n = is.read(buff)) >= 0){
            bos.write(buff, 0, n);
        }

        ByteArrayInputStream bis = new ByteArrayInputStream(bos.toByteArray());

        XMLDecoder decoder = new XMLDecoder(bis);
        GeneratorHeader header = (GeneratorHeader) decoder.readObject();

        decoder.close();

        return header;

    }

    public void saveToStream(OutputStream os) throws IOException{

        ByteArrayOutputStream out = new ByteArrayOutputStream();
        XMLEncoder encoder = new XMLEncoder(out);
        encoder.writeObject(this);
        encoder.close();

        os.write(out.toByteArray());

    }

    public int getVersion() {
        return version;
    }

    public void setVersion(int version) {
        this.version = version;
    }

    public String getGameName() {
        return gameName;
    }

    public void setGameName(String gameName) {
        this.gameName = gameName;
    }

    public long getNumberOfCards() {
        return numberOfCards;
    }

    public void setNumberOfCards(long numberOfCards) {
        this.numberOfCards = numberOfCards;
    }

    public int getBallsPerCard() {
        return ballsPerCard;
    }

    public void setBallsPerCard(int ballsPerCard) {
        this.ballsPerCard = ballsPerCard;
    }

    public Date getDate() {
        return date;
    }

    public void setDate(Date date) {
        this.date = date;
    }

    public int[] getNumbers() {
        return numbers;
    }

    public void setNumbers(int[] numbers) {
        this.numbers = numbers;
    }

}
/*
 * GeneratorInterface.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.generator;

import javax.swing.JPanel;

/**
 *
 * @author lagar
 */
public interface GeneratorInterface {

    public String getGeneratorName();

    public JPanel getGeneratorPanel();

}
/*
 * GeneratorMetaPanel.java
 *
 * Created on 18 de Abril de 2007, 22:18
 */

package br.ufsc.inf.lagar.ui.panels;

import java.awt.CardLayout;

/**
 *
```

```java
 * @author  lagar
 */
public class GeneratorMetaPanel extends javax.swing.JPanel {

    private GeneratorPanel gp;

    /** Creates new form GeneratorMetaPanel */
    public GeneratorMetaPanel() {
        initComponents();
        this.gp = new GeneratorPanel();
        this.gp.setMetaPanel(this);
        this.add(gp, gp.getClass().getName());
    }

    public void loadAvailableGenerators() throws Exception{
        gp.loadAvailableGenerators();
    }

    public void resetPanel(){
        CardLayout cardLayout = (CardLayout) this.getLayout();
        cardLayout.show(this, gp.getClass().getName());
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {

        setLayout(new java.awt.CardLayout());

    }// </editor-fold>//GEN-END:initComponents


    // Variables declaration - do not modify//GEN-BEGIN:variables
    // End of variables declaration//GEN-END:variables

}
/*
 * GeneratorPanel.java
 *
 * Created on 18 de Abril de 2007, 22:00
 */

package br.ufsc.inf.lagar.ui.panels;

import br.ufsc.inf.lagar.generator.GeneratorInterface;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.utils.RTSI;
import java.awt.CardLayout;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.ArrayList;
import java.util.HashMap;
import javax.swing.JPanel;

/**
 *
 * @author  lagar
 */
public class GeneratorPanel extends javax.swing.JPanel implements ActionListener {

    private HashMap<String, GeneratorInterface> vs = new HashMap<String,
GeneratorInterface>();
    private String selection;
    private JPanel metaPanel;

    /** Creates new form GeneratorPanel */
    public GeneratorPanel() {
        initComponents();
    }

    public void loadAvailableGenerators() throws Exception{

        GameConfiguration current = GameManager.getCurrentGame();

        String packageName = "br.ufsc.inf.lagar.generator.impl";

        boolean first = true;

        ArrayList<String> classes = RTSI.find(packageName, GeneratorInterface.class);
        for(String clazz: classes){

            clazz = packageName+"."+clazz;
            GeneratorInterface generator = (GeneratorInterface)
Class.forName(clazz).newInstance();

            this.vs.put(generator.getGeneratorName(), generator);
            RadioPanel rp = new RadioPanel();
            rp.getRadioButton().setText(generator.getGeneratorName());
            rp.getRadioButton().setActionCommand(generator.getGeneratorName());
            rp.getRadioButton().addActionListener(this);

            this.buttonGroup1.add(rp.getRadioButton());

            this.versPanel.add(rp);

            this.metaPanel.add(generator.getGeneratorPanel(),
generator.getGeneratorName());

            if(first){
                rp.getRadioButton().setSelected(true);
                first = false;
                this.selection = generator.getGeneratorName();
            }

        }

        this.jScrollPane1.revalidate();
        this.repaint();


    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        buttonGroup1 = new javax.swing.ButtonGroup();
        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jScrollPane1 = new javax.swing.JScrollPane();
        versPanel = new javax.swing.JPanel();
        labelSepLeft = new javax.swing.JLabel();
        openButton = new javax.swing.JButton();
        labelSepRight = new javax.swing.JLabel();
        simpleInternalFrame1 = new br.inf.ufsc.lagar.swingutil.SimpleInternalFrame();

        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        jLabel1.setText(bundle.getString("choose_one_generator_available")); //
NOI18N

        org.jdesktop.layout.GroupLayout jPanel1Layout = new
org.jdesktop.layout.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .add(jLabel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 482,
Short.MAX_VALUE)
                .addContainerGap())
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .add(jLabel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 43,
Short.MAX_VALUE)
                .addContainerGap())
        );

        jScrollPane1.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.ge
tString("Generators"))); // NOI18N
        versPanel.setLayout(new javax.swing.BoxLayout(versPanel,
javax.swing.BoxLayout.Y_AXIS));

        jScrollPane1.setViewportView(versPanel);

        openButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/runprog.png")));
        openButton.setText(bundle.getString("Open")); // NOI18N
        openButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                openButtonActionPerformed(evt);
            }
        });

        simpleInternalFrame1.setTitle(bundle.getString("Generators")); // NOI18N

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(labelSepLeft, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 198,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
```

```
                    .add(openButton)
                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                    .add(labelSepRight, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
198, Short.MAX_VALUE))
                .add(simpleInternalFrame1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
506, Short.MAX_VALUE)
                .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 506,
Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(simpleInternalFrame1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jPanel1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 175,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
                    .add(labelSepLeft,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(openButton)
                    .add(labelSepRight,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );

        layout.linkSize(new java.awt.Component[] {labelSepLeft, labelSepRight},
org.jdesktop.layout.GroupLayout.VERTICAL);

    }// </editor-fold>//GEN-END:initComponents

    private void openButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_openButtonActionPerformed

        CardLayout cardLayout = (CardLayout) this.metaPanel.getLayout();
        cardLayout.show(this.metaPanel, this.selection);

    }//GEN-LAST:event_openButtonActionPerformed

    public void actionPerformed(ActionEvent e) {
        this.selection = e.getActionCommand();
    }

    public JPanel getMetaPanel() {
        return metaPanel;
    }

    public void setMetaPanel(JPanel metaPanel) {
        this.metaPanel = metaPanel;
    }
```

```java
    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.ButtonGroup buttonGroup1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JLabel labelSepLeft;
    private javax.swing.JLabel labelSepRight;
    private javax.swing.JButton openButton;
    private br.inf.ufsc.lagar.swingutil.SimpleInternalFrame simpleInternalFrame1;
    private javax.swing.JPanel versPanel;
    // End of variables declaration//GEN-END:variables

}
/*
 * IWizardPage.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.ui.wizard;

public interface IWizardPage {

    public void setWizardPanel(WizardPanel wizardPanel);

    public void activatePage();

    public void deactivatePage();

    public String getPageName();

    public String getNextPageName();
}
/*
 * JTableTest.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.ui;

import br.ufsc.inf.lagar.ui.dialog.DecisionDialog;
import java.awt.Color;
import java.awt.Component;
import java.awt.Dimension;
import java.awt.Frame;
import java.util.Enumeration;
import javax.swing.JDialog;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.SwingConstants;
import javax.swing.event.TableModelListener;
import javax.swing.table.TableCellRenderer;
import javax.swing.table.TableColumn;
import javax.swing.table.TableModel;
import junit.framework.TestCase;

public class JTableTest extends TestCase implements TableModel{

    private CellData one = new CellData("AB", Color.GREEN);
    private CellData two = new CellData("CD", Color.RED);

    /**
     * Creates a new instance of JTableTest
     */
    public JTableTest() {

    }

    public int getRowCount() {
        return 500;
    }

    public int getColumnCount() {
        return 22;
    }

    public String getColumnName(int columnIndex) {
        return String.valueOf(columnIndex);
    }

    public Class<?> getColumnClass(int columnIndex) {
        return CellData.class;
    }

    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return false;
    }

    public Object getValueAt(int rowIndex, int columnIndex) {
        if(rowIndex == columnIndex){
            return this.one;
        }
        return this.two;
    }

    public void setValueAt(Object aValue, int rowIndex, int columnIndex) {
    }

    public void addTableModelListener(TableModelListener l) {
    }

    public void removeTableModelListener(TableModelListener l) {
    }

    public void testTable() {

        DecisionDialog d = new DecisionDialog(null, true);

        JTable table = new JTable(new JTableTest());
        table.setAutoResizeMode( JTable.AUTO_RESIZE_OFF );
        table.setDefaultRenderer(CellData.class, new CellDataRenderer());

        //table.setPreferredScrollableViewportSize(new Dimension(20,20));
//        table.createDefaultColumnsFromModel();

        Enumeration<TableColumn> columns = table.getColumnModel().getColumns();
        while(columns.hasMoreElements()){
            TableColumn t = columns.nextElement();
            t.setMinWidth(30);
            t.setPreferredWidth(30);
            //t.setResizable(false);
        }

        JScrollPane pane = new JScrollPane(table,
JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                                                 JScrollPane.HORIZONTAL_SCROLLBAR_AS
_NEEDED);

        d.getMainPanel().add(pane, "table");

        d.pack();
        d.setVisible(true);

    }
}

class CellData {

    private String text;
    private Color color;

    public CellData(){

    }

    public CellData(String text, Color c){
        this.text = text;
        this.color = c;
    }

    public String getText() {
        return text;
    }

    public void setText(String text) {
        this.text = text;
    }

    public Color getColor() {
        return color;
    }

    public void setColor(Color c) {
        this.color = c;
    }


}
class CellDataRenderer extends JLabel implements TableCellRenderer {

    public CellDataRenderer(){
        this.setOpaque(true);
        this.setHorizontalAlignment(SwingConstants.CENTER);
    }

    public Component getTableCellRendererComponent(JTable table, Object value,
boolean isSelected, boolean hasFocus, int row, int column) {

        CellData c = (CellData) value;
        if(!isSelected)this.setBackground(c.getColor());
        else{
            Color col = c.getColor();
            this.setBackground(col.darker());
        }
        this.setText(c.getText());

        return this;
    }

}
/*
 * LagarLotteryGlobals.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

public class LagarLotteryGlobals {

    public static final String GENERATED_HEADER_NAME = "gheader";
    public static final String GENERATED_CARDS_NAME = "cards";
    public static final String GENERATED_FILTER_NAME = "filters";

    public static final String LL_HOME =
System.getProperty("user.home")+"/.lagarLottery";

}
/*
 * LineAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis.gameanalysis;

import br.ufsc.inf.lagar.analysis.CellData;
import br.ufsc.inf.lagar.analysis.DefaultTableChartAnalysis;
import br.ufsc.inf.lagar.main.GameCard;
import java.awt.Color;
import java.lang.reflect.InvocationTargetException;
import java.text.DecimalFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Set;
import javax.swing.JOptionPane;
import javax.swing.SwingUtilities;
import org.jfree.chart.ChartFactory;
import org.jfree.chart.JFreeChart;
import org.jfree.chart.labels.PieSectionLabelGenerator;
import org.jfree.chart.labels.StandardPieSectionLabelGenerator;
import org.jfree.chart.plot.MultiplePiePlot;
import org.jfree.chart.plot.PiePlot;
import org.jfree.data.category.DefaultCategoryDataset;
import org.jfree.data.general.DefaultPieDataset;
import org.jfree.util.TableOrder;

public class LineAnalysis extends DefaultTableChartAnalysis {

    private String input = null;
    private List<String> cats;
    private Map<String, int[]> mChartData = new HashMap<String, int[]>();

    /** Creates a new instance of LineAnalysis */
    public LineAnalysis() {
    }
    protected void fillTableChart() {

        int lines = Integer.valueOf(this.input);
        int numDrawn = getCurrentGameConfiguration().getNumbersPerDrawing();
        int maxCat = numDrawn > lines ? lines : numDrawn;

        for(String cat: cats){
            this.mChartData.put(cat, new int[maxCat+1]);
        }

        GameCard cardBuffer = new GameCard();
        int[] catsCount = new int[this.cats.size()];

        for(int index = 0; this.gci.hasNext(); ++index){
            this.gci.next(cardBuffer);
            byte[] balls = cardBuffer.getBalls();

            for(int i = 0; i < balls.length; ++i){
                int ball = balls[i];

                int cat = (ball-1) / lines;
                ++catsCount[cat]; //increment sum on the column
            }

            for(int j = 0; j < catsCount.length; ++j){
                String category = this.cats.get(j);
                this.tcols.get(category)[index] = this.getHitz(catsCount[j]);
                ++this.mChartData.get(category)[catsCount[j]]; //increment sum of 
category
            }

            this.tcols.get(this.columnOne)[index] = new 
CellData(String.valueOf(cardBuffer.getSerial()), null, Color.BLACK, false);

            Arrays.fill(catsCount, 0);
//              Integer dc = this.dataChart.get(category);
//              this.dataChart.put(category, ++dc);
        }

    }

    protected List<String> buildCategories() {

        List<String> result = new ArrayList<String>();

        this.fetchNumberOfLines();

        final int linesOrig = Integer.valueOf(this.input.trim());
        int lines = linesOrig;
        int numbers = getCurrentGameConfiguration().getNumbersPerCard();
        int first = 1;

        while(numbers > 0){
            String cat = null;
            if(numbers >= linesOrig){
                cat = String.valueOf(first) + "-" + lines;
                result.add(cat);
            }
            else{
                lines-=numbers;
                cat = String.valueOf(first) + "-" + lines;
                result.add(cat);
                break;
            }
            first += linesOrig;
            numbers -= linesOrig;
            lines += linesOrig;
        }

        this.cats = result;
        return result;
    }

    public String getName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Line_analysis");
    }

    public String getHelpInfo() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Line_analysis_help_info");
    }

    public JFreeChart getChart() {
        DefaultCategoryDataset dataset = new DefaultCategoryDataset();
        for(String key: this.cats){
            int[] rows = this.mChartData.get(key);
            for(int i = 0; i < rows.length; ++i){
                dataset.setValue(rows[i], Integer.valueOf(i), key);
            }
        }

        JFreeChart chart = ChartFactory.createMultiplePieChart(this.getName(),
                dataset, TableOrder.BY_COLUMN, true,true,false);

        MultiplePiePlot plot = (MultiplePiePlot) chart.getPlot();
        PiePlot piePlot = (PiePlot) plot.getPieChart().getPlot();
        PieSectionLabelGenerator generator = new StandardPieSectionLabelGenerator(
                "{0} - {2}", new DecimalFormat("0"), new DecimalFormat("0.00%")
                );
        piePlot.setLabelGenerator(generator);

        return chart;
    }


    private void fetchNumberOfLines(){

        Runnable r = new Runnable() {
            public void run() {
                String s = JOptionPane.showInputDialog(
                null,
                java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gamean
alysis/analysisTr").getString("Enter_the_number_of_balls_per_line"),
                java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gamean
alysis/analysisTr").getString("Input"),
                JOptionPane.QUESTION_MESSAGE);
                if(s != null && !s.equals("")){
                    LineAnalysis.this.input = s;
                }
                else{
                    LineAnalysis.this.input =
String.valueOf(getCurrentGameConfiguration().getNumbersPerLine());
```

```java
                }
            }
        };

        if(SwingUtilities.isEventDispatchThread()){
            r.run();
        }
        else{
            try {
                SwingUtilities.invokeAndWait(r);
            } catch (InterruptedException ex) {
                ex.printStackTrace();
            } catch (InvocationTargetException ex) {
                ex.printStackTrace();
            }
        }
    }



}

package br.ufsc.inf.lagar.utils;

import java.io.IOException;
import java.io.InputStream;
import org.xml.sax.EntityResolver;
import org.xml.sax.InputSource;
import org.xml.sax.SAXException;

/**
 * This SAX EntityResolver allows us to validate against local DTDs.
 *
 * public static final String DSML_DTD_PUBLIC_ID = "http://www.dsml.org/DSML";
 * public static final String DSML_DTD_RESOURCE =
"/net/socialchange/bob/dtds/dsml.dtd";
 * ..
 * ..
 * builder.setEntityResolver(new LocalEntityResolver(DSML_DTD_PUBLIC_ID,
DSML_DTD_RESOURCE));
 * ..
 * SAXBuilder builder = new SAXBuilder(true);
 *
 *
 * Stolen from jakarta tomcat 3.2.2, share/org/apache/jasper/compiler/JspUtil.java
 */
public class LocalEntityResolver implements EntityResolver {

    String dtdId;
    String dtdResource;

    public LocalEntityResolver(String id, String resource) {
        this.dtdId = id;
        this.dtdResource = resource;
    }

    public InputSource resolveEntity(String publicId, String systemId)
        throws SAXException, IOException
        {
            System.out.println ("publicId = " + publicId);
            System.out.println ("systemId is " + systemId);
            System.out.println ("resource is " + dtdResource);
            if (publicId == null) {return null; }
            if (publicId.equals(dtdId)) {
                InputStream input =
                    this.getClass().getResourceAsStream(dtdResource);
                InputSource isrc =
                    new InputSource(input);
                return isrc;
            }
            else {
                System.out.println ("returning null");
                return null;
            }
        }
}
/*
 * LotofacilLoader.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.resultloader;

import br.ufsc.inf.lagar.main.GameCard;
import java.io.IOException;
import java.io.InputStream;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import org.htmlparser.Node;
import org.htmlparser.Parser;
import org.htmlparser.util.NodeList;
import org.htmlparser.util.ParserException;
import org.htmlparser.visitors.TagFindingVisitor;

public class LotofacilLoader extends CaixaLoader{

    /** Creates a new instance of LotofacilLoader */
    public LotofacilLoader() {
        //try {
            //this.resultPath = new
URL("http://www1.caixa.gov.br/loterias/_arquivos/loterias/D_quina.zip");
        //    this.resultPath = getClass().getResource("/D_lotfac.zip");
        //} catch (MalformedURLException ex) {
        //    ex.printStackTrace();
        //}
    }
```

```java
    public List<GameCard> loadResults() throws IOException {

        List<GameCard> lotofacilResults = new ArrayList<GameCard>();
        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

        InputStream in = this.getHtmlStream();

        byte[] buf = new byte[1024];
        StringBuilder sb = new StringBuilder(300);
        int len = 0;

        while((len = in.read(buf)) > 0){

            String s = new String(buf, 0, len);
            sb.append(s);
        }
        in.close();

        try {
            Parser parser = new Parser(sb.toString());
            sb = null;
            String [] tagsToBeFound = {"TR"};
            TagFindingVisitor visitor = new TagFindingVisitor (tagsToBeFound);

            parser.visitAllNodesWith (visitor);
            // First tag specified in search
            Node[] trTags = visitor.getTags(0);

            for(int i = 1; i < trTags.length; ++i){

                Node n = trTags[i];
                NodeList nl = n.getChildren();

                GameCard r = new GameCard(-1, null, 15);

                Node serial = nl.elementAt(1);
                r.setSerial(Integer.parseInt(serial.getChildren().toHtml()));

                Node date = nl.elementAt(3);
                try {
                    r.setDate(sdf.parse(date.getChildren().toHtml()));
                } catch (ParseException ex) {
                    throw new IOException("Error parsing date in result xml");
                }
                int index = 0;
                for(int j = 5; j <= 33; j+=2 ){

                    Node ball = nl.elementAt(j);
                    r.setBall((byte)Integer.parseInt(ball.getChildren().toHtml()),
index);
                    ++index;
                }
                Arrays.sort(r.getBalls());

                lotofacilResults.add(r);
            }

        } catch (ParserException ex) {
            ex.printStackTrace();
        }

        return lotofacilResults;
    }

}
/*
 * LotofacilLoaderTest.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.persistence.rs;

import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.resultloader.LotofacilLoader;
import br.ufsc.inf.lagar.resultloader.ResultLoader;
import java.io.IOException;
import java.util.List;
import junit.framework.*;
import java.net.URL;

/**
 *
 * @author lagar
 */
public class LotofacilLoaderTest extends TestCase {


    public void testLotofacilLoader() throws IOException{

        ResultLoader loader = new LotofacilLoader();

        List<GameCard> lotofacilResults = loader.loadResults();

        for(GameCard d: lotofacilResults){
            System.out.println(d);
        }
    }

}
/*
 * LowHighAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis.gameanalysis;

import br.ufsc.inf.lagar.analysis.CellData;
import br.ufsc.inf.lagar.analysis.DefaultTableChartAnalysis;
import br.ufsc.inf.lagar.main.GameCard;
import java.awt.Color;
import java.util.ArrayList;
import java.util.List;

public class LowHighAnalysis extends DefaultTableChartAnalysis{

    /** Creates a new instance of LowHighAnalysis */
    public LowHighAnalysis() {
    }


    protected void fillTableChart() {

        GameCard cardBuffer = new GameCard();

        for(int index = 0; this.gci.hasNext(); ++index){
            this.gci.next(cardBuffer);
            byte[] balls = cardBuffer.getBalls();

            int divider = getCurrentGameConfiguration().getNumbersPerCard() / 2;
            int lows    = 0;
            int highs   = 0;

            for(int i = 0; i < balls.length; ++i){
                int ball = balls[i];

                if(ball <= divider) ++lows;
                else                ++highs;
            }

            String category = lows+"-"+highs;
            this.tcols.get(category)[index] = this.hit;
            this.tcols.get(this.columnOne)[index] = new
CellData(String.valueOf(cardBuffer.getSerial()), null, Color.BLACK, false);

            Integer dc = this.dataChart.get(category);
            this.dataChart.put(category, ++dc);
        }


    }

    public String getName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Low_x_High_Analysis");
    }
    public String getHelpInfo() {
        return 
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Low_x_High_Analysis_Help_Info");
    }
    
    protected List<String> buildCategories(){
        
        List<String> result = new ArrayList<String>();
        
        int numbersPerCard = getCurrentGameConfiguration().getNumbersPerCard();
        int numbersPerDraw = getCurrentGameConfiguration().getNumbersPerDrawing();
        
        int numLows = (numbersPerCard / 2);
        
        if( numLows < numbersPerDraw ){
            
            int numHighs = numbersPerCard - numLows;
            int firstLow = numbersPerDraw - numHighs;
            
            for(int i = firstLow; i <= numLows; ++i){
                String cat = i+"-"+(numbersPerDraw-i);
                result.add(cat);
            }
            
        } else{
            for(int i = 0; i <= numbersPerDraw; ++i){
                String cat = i+"-"+(numbersPerDraw-i);
                result.add(cat);
            }
        }
        
        return result;
        
    }
    
}
/*
 * LowHighFilter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter.gamefilter;

import br.ufsc.inf.lagar.filter.AbstractGameFilter;
import br.ufsc.inf.lagar.filter.Filterable;
import br.ufsc.inf.lagar.filter.TextInputFilterException;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameManager;
import org.apache.commons.lang.StringUtils;

public class LowHighFilter extends AbstractGameFilter {

    private String[] categs;
    private int half;

    /** Creates a new instance of LowHighFilter */
    public LowHighFilter() {
        this.half = GameManager.getCurrentGame().getNumbersPerCard() / 2;
    }

    public void validateInput(String input) throws TextInputFilterException {

        String[] params = input.split(Filterable.OR_SEPARATOR);
        for(String s: params){

            String[] numbers = s.split("-");
            if(numbers.length != 2) throw new TextInputFilterException();

            for(String n: numbers){
                try{
                    Integer.parseInt(n.trim());
                }
                catch(NumberFormatException nfex){
                    throw new TextInputFilterException(nfex);
                }
            }

        }

        this.categs = new String[params.length];
        for(int i = 0; i < categs.length; ++i){
            categs[i] = StringUtils.deleteWhitespace(params[i]);
        }
    }

    public String getShortName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Low_x_High");
    }

    public String getFullName() {
        return getShortName();
    }

    public String getDescription() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Low_x_High_description");
    }

    public String getFilterHelp() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Low_x_High_help");
    }

    public boolean filter(GameCard card) {
```

```java
        String categ = this.getLowHighCategory(card);

        for(String p: this.categs){

            if(categ.equals(p)){
                return this.isMode();
            }
        }

        return !(this.isMode());
    }

    protected String getLowHighCategory(GameCard card){

        int lows = 0;
        byte[] balls = card.getBalls();

        for(byte b: balls){
            if(b <= this.half){
                ++lows;
            }
        }

        int highs = balls.length - lows;

        return lows + "-" + highs;

    }

}
/*
 * MainFrame.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import br.ufsc.inf.lagar.ui.dialog.DecisionDialog;
import br.ufsc.inf.lagar.ui.panels.AnalysisControlPanel;
import br.ufsc.inf.lagar.ui.panels.FilterPanel;
import br.ufsc.inf.lagar.ui.panels.GameChooserPanel;
import br.ufsc.inf.lagar.generator.impl.GeneratorAllCombPanel;
import br.ufsc.inf.lagar.ui.panels.GeneratorMetaPanel;
import br.ufsc.inf.lagar.ui.panels.GeneratorPanel;
import br.ufsc.inf.lagar.ui.panels.PrinterPanel;
import br.ufsc.inf.lagar.ui.panels.UpdateDbPanel;
import br.ufsc.inf.lagar.ui.panels.VerificatorPanel;
import java.awt.CardLayout;
import java.awt.Dimension;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.File;
import java.io.IOException;
import java.text.DateFormat;
import java.text.NumberFormat;
import java.util.List;
import java.util.Locale;
import javax.swing.JDialog;
import javax.swing.JEditorPane;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.SwingUtilities;
import org.apache.commons.lang.LocaleUtils;

/**
 *
 * @author  lagar
 */
public class MainFrame extends javax.swing.JFrame {

    private static final String PROGRAM_NAME = "lagarLottery 1.0";
    private JPanel homePanel = new JPanel();
    private AnalysisControlPanel analysisPanel = null;
    private GeneratorMetaPanel generatorPanel = null;
    private PrinterPanel printerPanel = null;
    private FilterPanel filterPanel = null;
    private VerificatorPanel verificatorPanel = null;

    /** Creates new form MainFrame */
    public MainFrame() {
        initComponents();

        this.buttonBar.setBackground(this.getBackground());
        this.setLocationRelativeTo(null);
        this.setExtendedState(JFrame.MAXIMIZED_BOTH);

    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        menuGroup = new javax.swing.ButtonGroup();
        jSplitPane = new javax.swing.JSplitPane();
        mainPanel = new javax.swing.JPanel();
        menuPane = new javax.swing.JScrollPane();
        buttonBar = new com.l2fprod.common.swing.JButtonBar();
        analysisButton = new javax.swing.JToggleButton();
        filterButton = new javax.swing.JToggleButton();
        generatorButton = new javax.swing.JToggleButton();
        printerButton = new javax.swing.JToggleButton();
        verificatorButton = new javax.swing.JToggleButton();
        statusPanel = new javax.swing.JPanel();
        menuBar = new javax.swing.JMenuBar();
        menuBarItem = new javax.swing.JMenu();
        listResMenuItem = new javax.swing.JMenuItem();
        updateResMenuItem = new javax.swing.JMenuItem();
        exitMenuItem = new javax.swing.JMenuItem();
        helpMenu = new javax.swing.JMenu();
        jMenuItem1 = new javax.swing.JMenuItem();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        jSplitPane.setDividerLocation(115);
        jSplitPane.setDividerSize(7);
        jSplitPane.setOneTouchExpandable(true);
        mainPanel.setLayout(new java.awt.CardLayout());

        mainPanel.setMinimumSize(new java.awt.Dimension(300, 300));
        jSplitPane.setRightComponent(mainPanel);

        menuPane.setBorder(null);
        menuPane.setMinimumSize(new java.awt.Dimension(115, 22));
        buttonBar.setBackground(new java.awt.Color(238, 238, 238));
        buttonBar.setPreferredSize(new java.awt.Dimension(50, 50));
        menuGroup.add(analysisButton);
        analysisButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/32x32/edu_mathematics.png"))
);
        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        analysisButton.setText(bundle.getString("Analysis")); // NOI18N
        analysisButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                analysisButtonActionPerformed(evt);
            }
        });

        menuGroup.add(filterButton);
        filterButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/32x32/filter.png")));
        filterButton.setText(bundle.getString("Filters")); // NOI18N
        filterButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                filterButtonActionPerformed(evt);
            }
        });

        menuGroup.add(generatorButton);
        generatorButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/32x32/gear.png")));
        generatorButton.setText(bundle.getString("Generators")); // NOI18N
        generatorButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                generatorButtonActionPerformed(evt);
            }
        });

        menuGroup.add(printerButton);
        printerButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/32x32/fileprint.png")));
        printerButton.setText(bundle.getString("Print")); // NOI18N
        printerButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                printerButtonActionPerformed(evt);
            }
        });
        menuGroup.add(verificatorButton);
        verificatorButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/32x32/button_ok.png")));
        verificatorButton.setText(bundle.getString("Verificators")); // NOI18N
        verificatorButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                verificatorButtonActionPerformed(evt);
            }
        });

        org.jdesktop.layout.GroupLayout buttonBarLayout = new
org.jdesktop.layout.GroupLayout(buttonBar);
        buttonBar.setLayout(buttonBarLayout);
        buttonBarLayout.setHorizontalGroup(
            buttonBarLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(buttonBarLayout.createSequentialGroup()
                .addContainerGap()
                .add(buttonBarLayout.createParallelGroup(org.jdesktop.layout.GroupLay
out.LEADING)
                    .add(generatorButton,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 86, Short.MAX_VALUE)
                    .add(filterButton, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
86, Short.MAX_VALUE)
                    .add(analysisButton,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 86, Short.MAX_VALUE)
                    .add(org.jdesktop.layout.GroupLayout.TRAILING, printerButton,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 86,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(verificatorButton,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 86,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );

        buttonBarLayout.linkSize(new java.awt.Component[] {analysisButton,
filterButton, generatorButton}, org.jdesktop.layout.GroupLayout.HORIZONTAL);

        buttonBarLayout.setVerticalGroup(
            buttonBarLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADI
NG)
            .add(buttonBarLayout.createSequentialGroup()
                .addContainerGap()
                .add(analysisButton, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
56, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(filterButton, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
56, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(generatorButton, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
66, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(printerButton, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
66, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(verificatorButton,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 66,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(77, Short.MAX_VALUE))
        );

        buttonBarLayout.linkSize(new java.awt.Component[] {analysisButton,
filterButton, generatorButton}, org.jdesktop.layout.GroupLayout.VERTICAL);
```

```
        menuPane.setViewportView(buttonBar);

        jSplitPane.setLeftComponent(menuPane);

        org.jdesktop.layout.GroupLayout statusPanelLayout = new
org.jdesktop.layout.GroupLayout(statusPanel);
        statusPanel.setLayout(statusPanelLayout);
        statusPanelLayout.setHorizontalGroup(
            statusPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEA
DING)
            .add(0, 640, Short.MAX_VALUE)
        );
        statusPanelLayout.setVerticalGroup(
            statusPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEA
DING)
            .add(0, 20, Short.MAX_VALUE)
        );

        menuBarItem.setText(bundle.getString("Menu")); // NOI18N
        listResMenuItem.setText(bundle.getString("List_results_DB")); // NOI18N
        listResMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                listResMenuItemActionPerformed(evt);
            }
        });

        menuBarItem.add(listResMenuItem);

        updateResMenuItem.setText(bundle.getString("Update_results_DB")); // NOI18N
        updateResMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                updateResMenuItemActionPerformed(evt);
            }
        });

        menuBarItem.add(updateResMenuItem);

        exitMenuItem.setText(bundle.getString("Exit")); // NOI18N
        exitMenuItem.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                exitMenuItemActionPerformed(evt);
            }
        });

        menuBarItem.add(exitMenuItem);

        menuBar.add(menuBarItem);

        helpMenu.setText(bundle.getString("About")); // NOI18N
        jMenuItem1.setText(bundle.getString("About_item")); // NOI18N
        jMenuItem1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jMenuItem1ActionPerformed(evt);
            }
        });

        helpMenu.add(jMenuItem1);

        menuBar.add(helpMenu);

        setJMenuBar(menuBar);

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jSplitPane, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 640,
Short.MAX_VALUE)
            .add(statusPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .add(jSplitPane, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 433,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(statusPanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
        );
        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void jMenuItem1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_jMenuItem1ActionPerformed

        new AboutDialog(null, false).setVisible(true);

    }//GEN-LAST:event_jMenuItem1ActionPerformed

    private void verificatorButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_verificatorButtonActionPerformed
        CardLayout cardLayout = (CardLayout) this.mainPanel.getLayout();
        cardLayout.show(this.mainPanel, VerificatorPanel.class.getName());
    }//GEN-LAST:event_verificatorButtonActionPerformed

    private void listResMenuItemActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_listResMenuItemActionPerformed

        Thread t = new Thread(new Runnable() {
            public void run() {
                List<GameCard> results =
GameManager.getCurrentGame().getGameResults().getLoadedResults();
                JEditorPane editor = new JEditorPane("text/html", "");
                StringBuilder sb = new StringBuilder();
                DateFormat df = DateFormat.getDateInstance(DateFormat.SHORT);
                NumberFormat nf = NumberFormat.getIntegerInstance();
                nf.setMinimumIntegerDigits(4);
                NumberFormat nfb = NumberFormat.getIntegerInstance();
                nfb.setMinimumIntegerDigits(2);
                sb.append("<html>");
                for(GameCard gc: results){
                    sb.append("<b>");
                    sb.append(nf.format(gc.getSerial()));
                    sb.append("</b>");
                    sb.append(" :  ");

                    for(byte i: gc.getBalls()){
                        sb.append(nfb.format(i));
                        sb.append(" - ");
                    }

                    sb.append(" ");
                    sb.append(df.format(gc.getDate()));
                    sb.append("<br>");
                }
                sb.append("</html>");

                editor.setText(sb.toString());

                JScrollPane scroll = new JScrollPane(editor,
                        JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                        JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);

                final DecisionDialog dd = new DecisionDialog(MainFrame.this, false);
                dd.getMainPanel().add(scroll, scroll.getClass().getName());
                dd.getButtonsPanel().setVisible(false);
                dd.getTitlePanel().setTitle(java.util.ResourceBundle.getBundle("resou
rces/bundles/lagarLottery").getString("Results_on_DB"));
                Dimension d = new Dimension(700, 400);
                dd.setPreferredSize(d);
                dd.setSize(d);
                dd.setTitle("");

                SwingUtilities.invokeLater(new Runnable() {
                    public void run() {
                        dd.setVisible(true);
                    }
                });
            }
        });

        t.start();


    }//GEN-LAST:event_listResMenuItemActionPerformed

    private void exitMenuItemActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_exitMenuItemActionPerformed

        this.exit();
    }//GEN-LAST:event_exitMenuItemActionPerformed

    private void updateResMenuItemActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_updateResMenuItemActionPerformed
        this.loadResults(true);

        this.analysisPanel.updateMaxCardsResultsDb(GameManager.getCurrentGame().getGa
meResults().getLoadedResults().size());
    }//GEN-LAST:event_updateResMenuItemActionPerformed

    private void filterButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_filterButtonActionPerformed

        CardLayout cardLayout = (CardLayout) this.mainPanel.getLayout();
        cardLayout.show(this.mainPanel, FilterPanel.class.getName());
    }//GEN-LAST:event_filterButtonActionPerformed

    private void printerButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_printerButtonActionPerformed

        CardLayout cardLayout = (CardLayout) this.mainPanel.getLayout();
        cardLayout.show(this.mainPanel, PrinterPanel.class.getName());
    }//GEN-LAST:event_printerButtonActionPerformed

    private void generatorButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_generatorButtonActionPerformed
```

```
        this.generatorPanel.resetPanel();
        CardLayout cardLayout = (CardLayout) this.mainPanel.getLayout();
        cardLayout.show(this.mainPanel, GeneratorMetaPanel.class.getName());
    }//GEN-LAST:event_generatorButtonActionPerformed

    private void analysisButtonActionPerformed(java.awt.event.ActionEvent evt)
{//GEN-FIRST:event_analysisButtonActionPerformed

        CardLayout cardLayout = (CardLayout) this.mainPanel.getLayout();
        cardLayout.show(this.mainPanel, AnalysisControlPanel.class.getName());
    }//GEN-LAST:event_analysisButtonActionPerformed

    public void startApp() {

        List<GameConfiguration> games = GameManager.getAvailableGames();

        GameChooserPanel panel = new GameChooserPanel(games.size());

        for(GameConfiguration gc: games){
            panel.addOption(gc.getName());
        }

        String game = null;
        try{
            game = panel.getSelectedGame();
        }
        catch(Exception ex){
            this.exit();
        }

        if(game == null){
            this.exit();
        }

        GameManager.setCurrentGame(game);

        this.setTitle(game + " - " + PROGRAM_NAME);

        this.loadResults(false);
        this.loadModules();
    }

    private void exit(){
        System.exit(0);
    }

    private void loadResults(boolean onMainMenu){
        try {

            final UpdateDbPanel upPanel = new UpdateDbPanel();

            final GameResults gr = GameManager.getCurrentGame().getGameResults();

            final String originalResultPath = gr.getResultLoader().getResultPath();
            String localPath = LagarLotteryGlobals.LL_HOME;
            int pos = originalResultPath.lastIndexOf("/");
            localPath = localPath + originalResultPath.substring(pos);
            System.out.println(localPath);

            File localFile = new File(localPath);
            if(localFile.exists()){

                gr.getResultLoader().setResultPath(localFile.toURI().toURL().toString
());
                gr.loadResults();

                gr.getResultLoader().setResultPath(originalResultPath);

                if(!onMainMenu){
                    return;
                }

                List<GameCard> cards = gr.getLoadedResults();
                DateFormat format = DateFormat.getDateInstance(DateFormat.SHORT);
                GameCard last = cards.get(cards.size()-1);
                StringBuilder sb = new StringBuilder();
                sb.append(java.util.ResourceBundle.getBundle("resources/bundles/lagar
Lottery").getString("<html>Your_results_database_ends_in_card:<br>"));
                sb.append(java.util.ResourceBundle.getBundle("resources/bundles/lagar
Lottery").getString("Number:_"));
                sb.append(last.getSerial());
                sb.append("<br>");
                sb.append(java.util.ResourceBundle.getBundle("resources/bundles/lagar
Lottery").getString("Date:_"));
                sb.append(format.format(last.getDate()));
                sb.append("<br>");
                sb.append("<br>");
                sb.append(java.util.ResourceBundle.getBundle("resources/bundles/lagar
Lottery").getString("Do_you_want_to_update_your_results_database?</html>"));

                upPanel.getInfoLabel().setText(sb.toString());

            }
            else{
               upPanel.getInfoLabel().setText(java.util.ResourceBundle.getBundle("res
ources/bundles/lagarLottery").getString("You_don't_have_a_local_results_database._Loa
d_from_the_web?"));

            }
            upPanel.getUrlField().setText(originalResultPath);

            final DecisionDialog dd = new DecisionDialog(this, true);
            dd.getTitlePanel().setTitle(java.util.ResourceBundle.getBundle("resources
/bundles/lagarLottery").getString("Update_db?"));
            dd.getMainPanel().add(upPanel, upPanel.getClass().getName());

            ActionListener l = new ActionListener() {
                public void actionPerformed(ActionEvent e) {
                    if(e.getActionCommand().equals(dd.getOkButton().getText())){

                        upPanel.getInfoLabel().setText(java.util.ResourceBundle.getBu
ndle("resources/bundles/lagarLottery").getString("Updating_results.._please_wait."));
                        dd.getOkButton().setEnabled(false);
                        dd.getCancelButton().setEnabled(false);
                        dd.setDefaultCloseOperation(JDialog.DO_NOTHING_ON_CLOSE);

                        Thread t = new Thread(new Runnable() {
                            public void run() {
                                try {
                                    String web =
upPanel.getUrlField().getText().trim();
                                    if(!originalResultPath.equals(web)){
                                        gr.getResultLoader().setResultPath(web);
                                    }
                                    gr.loadResults();
                                    JOptionPane.showMessageDialog(
                                    null,
                                    java.util.ResourceBundle.getBundle("resources/bun
dles/lagarLottery").getString("Updated_successfully"),
                                    java.util.ResourceBundle.getBundle("resources/bun
dles/lagarLottery").getString("Ok"),
                                    JOptionPane.INFORMATION_MESSAGE);
                                } catch (Exception ex) {
                                    ex.printStackTrace();
                                    JOptionPane.showMessageDialog(
                                            null,
                                            java.util.ResourceBundle.getBundle("resou
rces/bundles/lagarLottery").getString("Error_loading_results"),
                                            java.util.ResourceBundle.getBundle("resou
rces/bundles/lagarLottery").getString("Error"),
                                            JOptionPane.ERROR_MESSAGE);
                                }

                                SwingUtilities.invokeLater(new Runnable() {
                                    public void run() {
                                        dd.dispose();
                                    }
                                });

                            }

                        });
                        t.start();

                    }
                    else{
                        dd.dispose();
                    }
                }
            };

            dd.getOkButton().addActionListener(l);
            dd.getCancelButton().addActionListener(l);

            dd.setVisible(true);


        } catch (IOException ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(
                    null,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error_loading_results_database"),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Fatal_Error"),
                    JOptionPane.ERROR_MESSAGE);
        }
    }

    private void loadModules(){

        this.analysisPanel = new AnalysisControlPanel();
        try{
            this.analysisPanel.fillAvailableAnalysis();
        }
        catch(Exception ex){
            ex.printStackTrace();
            JOptionPane.showMessageDialog(this,
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Fatal
_error_loading_available_analysis_-_will_exit_now"),
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Error
"), JOptionPane.ERROR_MESSAGE);
            this.exit();
        }

        this.generatorPanel = new GeneratorMetaPanel();
        try {
            this.generatorPanel.loadAvailableGenerators();
        } catch (Exception ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(this,
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Fatal
_error_loading_available_generators_-_will_exit_now"),
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Error
"), JOptionPane.ERROR_MESSAGE);
            this.exit();
        }

        this.printerPanel = new PrinterPanel();

        this.filterPanel = new FilterPanel();
        try {
            this.filterPanel.loadAvailableFilters();
        } catch (Exception ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(this,
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Fatal
_error_loading_available_filters_-_will_exit_now"),
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Error
"), JOptionPane.ERROR_MESSAGE);
            this.exit();
        }

        this.verificatorPanel = new VerificatorPanel();
        try {
            this.verificatorPanel.loadAvailableVerificators();
        } catch (Exception ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(this,
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Fatal
_error_loading_available_verificators_-_will_exit_now"),
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Error
"), JOptionPane.ERROR_MESSAGE);
            this.exit();
        }

        this.mainPanel.add(this.homePanel, "homePanel");
        this.mainPanel.add(this.analysisPanel, AnalysisControlPanel.class.getName());
        this.mainPanel.add(this.generatorPanel, GeneratorMetaPanel.class.getName());
        this.mainPanel.add(this.filterPanel, FilterPanel.class.getName());
        this.mainPanel.add(this.printerPanel, PrinterPanel.class.getName());
        this.mainPanel.add(this.verificatorPanel, VerificatorPanel.class.getName());
    }


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JToggleButton analysisButton;
    private com.l2fprod.common.swing.JButtonBar buttonBar;
    private javax.swing.JMenuItem exitMenuItem;
    private javax.swing.JToggleButton filterButton;
    private javax.swing.JToggleButton generatorButton;
    private javax.swing.JMenu helpMenu;
    private javax.swing.JMenuItem jMenuItem1;
```

```
    private javax.swing.JSplitPane jSplitPane;
    private javax.swing.JMenuItem listResMenuItem;
    private javax.swing.JPanel mainPanel;
    private javax.swing.JMenuBar menuBar;
    private javax.swing.JMenu menuBarItem;
    private javax.swing.ButtonGroup menuGroup;
    private javax.swing.JScrollPane menuPane;
    private javax.swing.JToggleButton printerButton;
    private javax.swing.JPanel statusPanel;
    private javax.swing.JMenuItem updateResMenuItem;
    private javax.swing.JToggleButton verificatorButton;
    // End of variables declaration//GEN-END:variables

}
/*
 * Main.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.main;

import br.ufsc.inf.lagar.analysis.CellDataRenderer;
import br.ufsc.inf.lagar.main.MainFrame;
import com.jgoodies.looks.plastic.Plastic3DLookAndFeel;
import com.jgoodies.looks.plastic.PlasticLookAndFeel;
import com.jgoodies.looks.plastic.PlasticXPLookAndFeel;
import com.jgoodies.looks.plastic.theme.Silver;
import com.jgoodies.looks.plastic.theme.SkyBluer;
import java.awt.Color;
import java.io.File;
import java.util.List;
import javax.swing.UIManager;

public class Main {

    /** Creates a new instance of Main */
    public Main() {
    }

    public static void main(String[] args){

        double mega = 1024d * 1024d;

        System.out.println("TotalMemory = "+Runtime.getRuntime().totalMemory() / mega
+" MB");
        System.out.println("MaxMemory = "+Runtime.getRuntime().maxMemory() / mega +"
MB");
        System.out.println("Using java = "+System.getProperty("java.version"));

        setLookAndFeel();

        File localHome = new File(LagarLotteryGlobals.LL_HOME);
        if(!localHome.exists()){
            localHome.mkdir();
        }

        javax.swing.SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                createAndShowGUI();
            }
        });

    }

    private static void createAndShowGUI(){

        MainFrame frame = new MainFrame();
        frame.pack();
        frame.setVisible(true);
        frame.startApp();
    }

    private static void setLookAndFeel(){
        try{

            PlasticLookAndFeel.setTabStyle(PlasticLookAndFeel.TAB_STYLE_METAL_VALUE);
            PlasticXPLookAndFeel xp = new PlasticXPLookAndFeel();
            xp.setPlasticTheme(new SkyBluer());

            UIManager.setLookAndFeel(xp);
//            UIManager.setLookAndFeel("com.sun.java.swing.plaf.gtk.GTKLookAndFeel");
        }
        catch(Exception ex){
            try {
                UIManager.setLookAndFeel("javax.swing.plaf.metal.MetalLookAndFeel");
            } catch (Exception ex2){}
        }
    }

}
/*
 * OddEvenAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis.gameanalysis;

import br.ufsc.inf.lagar.analysis.CellData;
import br.ufsc.inf.lagar.analysis.DefaultTableChartAnalysis;
import br.ufsc.inf.lagar.main.GameCard;
import java.awt.Color;
import java.util.ArrayList;
import java.util.List;

public class OddEvenAnalysis extends DefaultTableChartAnalysis{


    /** Creates a new instance of OddEvenAnalysis */
    public OddEvenAnalysis() {
    }

    protected void fillTableChart() {

        GameCard cardBuffer = new GameCard();

        for(int index = 0; this.gci.hasNext(); ++index){
            this.gci.next(cardBuffer);
            byte[] balls = cardBuffer.getBalls();

            int evens = 0;
            int odds  = 0;

            for(int i = 0; i < balls.length; ++i){
                int ball = balls[i];

                if((ball % 2) == 0) ++evens;
                else                ++odds;
            }

            String category = evens+"-"+odds;
            this.tcols.get(category)[index] = this.hit;
            this.tcols.get(this.columnOne)[index] = new
CellData(String.valueOf(cardBuffer.getSerial()), null,Color.BLACK, false);

            Integer dc = this.dataChart.get(category);
            this.dataChart.put(category, ++dc);
        }

    }

    public String getName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Even_x_Odd_analysis");
    }

    public String getHelpInfo() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/analysis/gameanalysis/analysisT
r").getString("Even_x_Odd_analysis_help_info");
    }


    protected List<String> buildCategories(){

        List<String> result = new ArrayList<String>();
        int numbersPerCard = getCurrentGameConfiguration().getNumbersPerCard();
        //int numbersPerDraw = getCurrentGameConfiguration().getNumbersPerDrawing();
        int numbersPerDraw = this.gci.getBallsPerCard();

        int numEvens = (numbersPerCard / 2);

        if( numEvens < numbersPerDraw ){

            int numOdds = numbersPerCard - numEvens;
            int firstEven = numbersPerDraw - numOdds;

            for(int i = firstEven; i <= numEvens; ++i){
                String cat = i+"-"+(numbersPerDraw-i);
                result.add(cat);
            }

        } else{
            for(int i = 0; i <= numbersPerDraw; ++i){
                String cat = i+"-"+(numbersPerDraw-i);
                result.add(cat);
            }
        }

        return result;
    }


}
/*
 * OddEvenAnalysisTest.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.core.analysis.gameanalysis;

import br.ufsc.inf.lagar.analysis.CellDataRenderer;
import br.ufsc.inf.lagar.analysis.gameanalysis.OddEvenAnalysis;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.ui.dialog.DecisionDialog;
import java.io.IOException;
import java.util.Enumeration;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.ToolTipManager;
import javax.swing.table.TableColumn;
import junit.framework.*;
import br.ufsc.inf.lagar.analysis.CellData;
import br.ufsc.inf.lagar.analysis.TableChartAnalysis;
import br.ufsc.inf.lagar.main.GameResultIterator;
import java.util.ArrayList;
import java.util.List;
import org.jfree.chart.JFreeChart;

/**
 *
 * @author lagar
 */
public class OddEvenAnalysisTest extends TestCase {


    public void testTable() throws IOException{

        DecisionDialog d = new DecisionDialog(null, true);

        List<GameConfiguration> availableGames = GameManager.getAvailableGames();
        GameConfiguration gc = availableGames.get(1);
        //gc.gameResults.loadResults();

        assertNotNull(availableGames);
        assertTrue(availableGames.size() != 0);

        OddEvenAnalysis instance = new OddEvenAnalysis();
        instance.setBegin(1);
        instance.setEnd(20);
        instance.setJump(1);
        instance.setCurrentGameConfiguration(gc);
        instance.calculateAnalysis();

        JTable table = new JTable(instance);
        table.setAutoResizeMode( JTable.AUTO_RESIZE_OFF );
        table.setDefaultRenderer(CellData.class, new CellDataRenderer());

        //table.setPreferredScrollableViewportSize(new Dimension(20,20));
//        table.createDefaultColumnsFromModel();

        Enumeration<TableColumn> columns = table.getColumnModel().getColumns();
        while(columns.hasMoreElements()){
            TableColumn t = columns.nextElement();
            t.setMinWidth(50);
            t.setPreferredWidth(50);
            //t.setResizable(false);
        }

        JScrollPane pane = new JScrollPane(table,
JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                                                  JScrollPane.HORIZONTAL_SCROLLBAR_AS
_NEEDED);

        ToolTipManager.sharedInstance().unregisterComponent(table);
        ToolTipManager.sharedInstance().unregisterComponent(table.getTableHeader());

        d.getMainPanel().add(pane, "table");

        d.pack();
        d.setVisible(true);

    }
    /**
     * Test of buildEvenOddCategories method, of class
br.ufsc.inf.lagar.core.analysis.gameanalysis.OddEvenAnalysis.
     */
    public void testBuildOddEvenCategories() {
        System.out.println("buildOddEvenCategories");

        List<GameConfiguration> availableGames = GameManager.getAvailableGames();

        assertNotNull(availableGames);
        assertTrue(availableGames.size() != 0);

        OddEvenAnalysis instance = new OddEvenAnalysis();
        instance.setCurrentGameConfiguration(availableGames.get(0));

        List<String> expResult = null;
        List<String> result = instance.buildEvenOddCategories();

        for(String s: result){
            System.out.println(s);
        }
    }

}
/*
 * OddEvenFilter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter.gamefilter;

import br.ufsc.inf.lagar.filter.AbstractGameFilter;
import br.ufsc.inf.lagar.filter.Filterable;
import br.ufsc.inf.lagar.filter.TextInputFilterException;
import br.ufsc.inf.lagar.main.GameCard;
import org.apache.commons.lang.StringUtils;

public class OddEvenFilter extends AbstractGameFilter{

    private String[] categs;

    /** Creates a new instance of OddEvenFilter */
    public OddEvenFilter() {
    }

    public String getShortName() {
        return
```

```
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Even_x_Odd");
    }

    public String getFullName() {
        return this.getShortName();
    }

    public String getDescription() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Even_x_Odd_description");
    }

    public String getFilterHelp() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("oddeven_filter_help");
    }


    public void validateInput(String input) throws TextInputFilterException {

        String[] params = input.split(Filterable.OR_SEPARATOR);
        for(String s: params){

            String[] numbers = s.split("-");
            if(numbers.length != 2) throw new TextInputFilterException();

            for(String n: numbers){
                try{
                    Integer.parseInt(n.trim());
                }
                catch(NumberFormatException nfex){
                    throw new TextInputFilterException(nfex);
                }
            }

        }

        this.categs = new String[params.length];
        for(int i = 0; i < categs.length; ++i){
            categs[i] = StringUtils.deleteWhitespace(params[i]);
        }
    }

    public boolean filter(GameCard card) {

        String categ = this.getOddEvenCategory(card);

        for(String p: this.categs){

            if(categ.equals(p)){
                return this.isMode();
            }
        }

        return !(this.isMode());
    }

    protected String getOddEvenCategory(GameCard card){

        int evens = 0;
        byte[] balls = card.getBalls();
        for(byte b: balls){
            if(b % 2 == 0){
                ++evens;
            }
        }

        int odds = balls.length - evens;

        return evens + "-" + odds;

    }



}
/*
 * PrinterManager.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.printer;

import br.ufsc.inf.lagar.main.GameManager;
import java.awt.print.Book;
import java.awt.print.PageFormat;
import java.awt.print.PrinterException;
import java.awt.print.PrinterJob;
import java.io.File;
import java.io.IOException;

public class PrinterManager {

    /** Creates a new instance of PrinterManager */
    public PrinterManager() {
    }

    public void printGameFile(File llgFile) throws IOException, PrinterException{

        GameCardPrinter printer = GameManager.getCurrentGame().getGameCardPrinter();

        printer.setFile(llgFile);

        PrinterJob job = PrinterJob.getPrinterJob();
        job.setPrintable(printer);

        boolean doPrint = job.printDialog();
        if (doPrint) {
            job.print();
        }
    }

}
/*
 * PrinterPanel.java
 *
 * Created on 29 de Março de 2007, 20:10
 */

package br.ufsc.inf.lagar.ui.panels;

import br.ufsc.inf.lagar.printer.PrinterManager;
import java.awt.print.PrinterException;
import java.io.File;
import java.io.IOException;
import javax.swing.JOptionPane;

/**
 *
 * @author  lagar
 */
public class PrinterPanel extends javax.swing.JPanel {

    /** Creates new form PrinterPanel */
    public PrinterPanel() {
        initComponents();
    }

    public void printGames(){

        String message = null;

        try{
            File llgFile = new File(this.fileChooserLLG1.getFilePath());
            PrinterManager m = new PrinterManager();
            m.printGameFile(llgFile);
        }
        catch(IOException ioex){
            message =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Error
_loading_file.");
        }
        catch(PrinterException pex){
            message =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery").getString("Error
_printing_file.");
        }

        if(message != null){
            JOptionPane.showMessageDialog(
                    this,
                    message,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
        }

    }

    /** This method is called from within the constructor to
```

```
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        simpleInternalFrame1 = new br.inf.ufsc.lagar.swingutil.SimpleInternalFrame();
        jPanel1 = new javax.swing.JPanel();
        fileChooserLLG1 = new br.ufsc.inf.lagar.ui.panels.FileChooserLLG();
        jPanel2 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jPanel3 = new javax.swing.JPanel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        printButton = new javax.swing.JButton();

        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        simpleInternalFrame1.setTitle(bundle.getString("Print_cards")); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.getStri
ng("Choose_cards_to_print"))); // NOI18N

        org.jdesktop.layout.GroupLayout jPanel1Layout = new
org.jdesktop.layout.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(fileChooserLLG1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 516,
Short.MAX_VALUE)
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(fileChooserLLG1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
        );

        jLabel1.setText(bundle.getString("This_module_will_print_generated_cards_dire
ctly_into_bet_cards_using_your_printer.")); // NOI18N

        org.jdesktop.layout.GroupLayout jPanel2Layout = new
org.jdesktop.layout.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(
            jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jPanel2Layout.createSequentialGroup()
                .addContainerGap()
                .add(jLabel1)
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
        jPanel2Layout.setVerticalGroup(
            jPanel2Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jPanel2Layout.createSequentialGroup()
                .addContainerGap()
                .add(jLabel1)
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
```

```
        );

        printButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/runprog.png")));
        printButton.setText(bundle.getString("Print")); // NOI18N
        printButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                printButtonActionPerformed(evt);
            }
        });

        org.jdesktop.layout.GroupLayout jPanel3Layout = new
org.jdesktop.layout.GroupLayout(jPanel3);
        jPanel3.setLayout(jPanel3Layout);
        jPanel3Layout.setHorizontalGroup(
            jPanel3Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jPanel3Layout.createSequentialGroup()
                .add(jLabel2, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 209,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(printButton)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jLabel3, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 210,
Short.MAX_VALUE))
        );
        jPanel3Layout.setVerticalGroup(
            jPanel3Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
jPanel3Layout.createSequentialGroup()
                .addContainerGap()
                .add(jPanel3Layout.createParallelGroup(org.jdesktop.layout.GroupLayou
t.TRAILING)
                    .add(org.jdesktop.layout.GroupLayout.LEADING, jLabel3,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 25, Short.MAX_VALUE)
                    .add(org.jdesktop.layout.GroupLayout.LEADING, printButton,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .add(org.jdesktop.layout.GroupLayout.LEADING, jLabel2,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 25, Short.MAX_VALUE))
                .addContainerGap())
        );

        jPanel3Layout.linkSize(new java.awt.Component[] {jLabel2, jLabel3},
org.jdesktop.layout.GroupLayout.VERTICAL);

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(simpleInternalFrame1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
526, Short.MAX_VALUE)
            .add(jPanel2, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(jPanel3, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(simpleInternalFrame1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jPanel2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jPanel1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jPanel3, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(145, Short.MAX_VALUE))
        );
    }// </editor-fold>//GEN-END:initComponents

    private void printButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_printButtonActionPerformed

        this.printGames();
    }//GEN-LAST:event_printButtonActionPerformed


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private br.ufsc.inf.lagar.ui.panels.FileChooserLLG fileChooserLLG1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JButton printButton;
    private br.inf.ufsc.lagar.swingutil.SimpleInternalFrame simpleInternalFrame1;
    // End of variables declaration//GEN-END:variables

}
/*
 * PrinterTest.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.zmisc.test;
```

```
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.print.PageFormat;
import java.awt.print.Paper;
import java.awt.print.Printable;
import java.awt.print.PrinterException;
import java.awt.print.PrinterJob;
import javax.swing.JFrame;

/**
 *
 * @author lagar
 */
public class PrinterTest {

    public PrinterTest() {

    }

    public void testPrinter(){

        PrinterJob job = PrinterJob.getPrinterJob();
        job.setPrintable(new HelloWorldPrinter());

        boolean doPrint = job.printDialog();

        if (doPrint) {
          try {
               job.print();
          } catch (PrinterException e) {
            e.printStackTrace();
          }
        }

//        JFrame abc = new JFrame();
//        abc.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
//        abc.setVisible(true);

    }

    public static void main(String[] args){
        new PrinterTest().testPrinter();
    }
}

class HelloWorldPrinter implements Printable{

    public int print(Graphics g, PageFormat pf, int pageIndex) throws
PrinterException {

        if (pageIndex > 0) { /* We have only one page, and 'page' is zero-based */
            return NO_SUCH_PAGE;
        }

        /* User (0,0) is typically outside the imageable area, so we must
         * translate by the X and Y values in the PageFormat to avoid clipping
         */
        Graphics2D g2d = (Graphics2D) g;

        Paper p = (Paper) pf.getPaper().clone();
        p.setImageableArea(0, 0, p.getWidth(), p.getHeight());
        pf.setPaper(p);
        double x = (pf.getWidth() / 2) - cmToPixel(4.1);
        double y = 0;

        g2d.translate(x, y);

        /* Now we perform our rendering */
        g2d.fillRect((int)cmToPixel(1.2), (int)cmToPixel(5.82), (int)cmToPixel(0.3),
(int)cmToPixel(0.18));

        g2d.fillRect((int)cmToPixel(1.2+0.65), (int)cmToPixel(5.82),
(int)cmToPixel(0.3), (int)cmToPixel(0.18));

        g2d.fillRect((int)cmToPixel(1.2+0.65+0.65), (int)cmToPixel(5.82),
(int)cmToPixel(0.3), (int)cmToPixel(0.18));

        g2d.fillRect((int)cmToPixel(1.2+0.65+0.65+0.65), (int)cmToPixel(5.82),
(int)cmToPixel(0.3), (int)cmToPixel(0.18));

        g2d.fillRect((int)cmToPixel(1.2+0.65+0.65+0.65+0.65), (int)cmToPixel(5.82),
(int)cmToPixel(0.3), (int)cmToPixel(0.18));



        g2d.fillRect((int)cmToPixel(1.2), (int)cmToPixel(5.82+0.325),
(int)cmToPixel(0.3), (int)cmToPixel(0.18));

        g2d.fillRect((int)cmToPixel(1.2+0.65), (int)cmToPixel(5.82+0.325+0.325),
(int)cmToPixel(0.3), (int)cmToPixel(0.18));

        g2d.fillRect((int)cmToPixel(1.2+0.65+0.65),
(int)cmToPixel(5.82+0.325+0.325+0.325), (int)cmToPixel(0.3), (int)cmToPixel(0.18));

        g2d.fillRect((int)cmToPixel(1.2+0.65+0.65+0.65),
(int)cmToPixel(5.82+0.325+0.325+0.325+0.325), (int)cmToPixel(0.3),
(int)cmToPixel(0.18));

        g2d.fillRect((int)cmToPixel(1.2+0.65+0.65+0.65+0.65),
(int)cmToPixel(5.82+0.325+0.325+0.325+0.325+0.325), (int)cmToPixel(0.3),
(int)cmToPixel(0.18));

        /* tell the caller that this page is part of the printed document */
        return PAGE_EXISTS;
    }

    private double cmToPixel(double cm){

        double inch = this.cmToInch(cm);

        return inch * 72;
    }

    private double cmToInch(double cm){

        return cm / 2.54;
    }

}
/*
 * PrinterTestV2.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.zmisc.test;

import br.ufsc.inf.lagar.main.GameCardLayout;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.printer.PrinterManager;
import java.awt.Dimension;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Point;
import java.awt.print.Book;
import java.awt.print.PageFormat;
import java.awt.print.Pageable;
import java.awt.print.Paper;
import java.awt.print.Printable;
import java.awt.print.PrinterException;
import java.awt.print.PrinterJob;
import java.io.File;
import java.util.List;
import javax.print.attribute.HashPrintRequestAttributeSet;
import javax.print.attribute.PrintRequestAttributeSet;
import javax.print.attribute.standard.Copies;
import javax.print.attribute.standard.MediaSize;
import javax.print.attribute.standard.PageRanges;
import javax.swing.JFrame;

/**
 *
 * @author lagar
 */
public class PrinterTestV2 {

    public PrinterTestV2() {

    }

    public void testPrinter(){

        PrinterJob job = PrinterJob.getPrinterJob();
        CardPrinter cardPrinter = new CardPrinter();

        job.setPrintable(cardPrinter);

//        PrintRequestAttributeSet attr_set = new HashPrintRequestAttributeSet();
//        attr_set.add(new PageRanges(1,5));

        boolean doPrint = job.printDialog();
        if (doPrint) {
          try {
               job.print();
          } catch (PrinterException e) {
            e.printStackTrace();
          }
        }

//          JFrame abc = new JFrame();
//          abc.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
//          abc.setVisible(true);

    }

    public static void main(String[] args) throws Exception{

        List<GameConfiguration> availableGames = GameManager.getAvailableGames();
        GameConfiguration gc = availableGames.get(0);
        gc.getGameResults().loadResults();
        GameManager.setCurrentGame(gc);

        //new PrinterTestV2().testPrinter();
        File llgFile = new File("/home/lagar/teste/bb6.llg");
        new PrinterManager().printGameFile(llgFile);

//          JFrame abc = new JFrame();
//          abc.setMinimumSize(new Dimension(500,500));
//          abc.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
//          abc.setVisible(true);
    }
}

class CardPrinter implements Printable{

    GameConfiguration gc = GameManager.getCurrentGame();

    public int print(Graphics g, PageFormat pf, int pageIndex) throws
PrinterException {

        if (pageIndex > 5) { /* We have only one page, and 'page' is zero-based */
            return NO_SUCH_PAGE;
        }

        /* User (0,0) is typically outside the imageable area, so we must
         * translate by the X and Y values in the PageFormat to avoid clipping
         */
        Graphics2D g2d = (Graphics2D) g;

        Paper p = (Paper) pf.getPaper().clone();
        p.setImageableArea(0, 0, p.getWidth(), p.getHeight());
        pf.setPaper(p);

        double x = (pf.getWidth() / 2) - cmToPixel(4.1);
        double y = 0;

        g2d.translate(x, y);

        GameCardLayout gcl = gc.getCardLayout();
        double width = gcl.getBallWidth();
        double height = gcl.getBallHeigth();

        int[] nums = {1, 25, 58, 74, 79};
        for(int i: nums){
            Point.Double p2d = gcl.getBalls().get(i);
            /* Now we perform our rendering */
            g2d.fillRect((int)cmToPixel(p2d.x), (int)cmToPixel(p2d.y),
(int)cmToPixel(width), (int)cmToPixel(height));
        }

        List<GameCardLayout.ExtendedBall> exbs = gcl.getExtendedBalls();
        Point.Double p2d = exbs.get(0).getBalls().get(nums.length);
        g2d.fillRect((int)cmToPixel(p2d.x), (int)cmToPixel(p2d.y),
(int)cmToPixel(width), (int)cmToPixel(height));

        /* tell the caller that this page is part of the printed document */
        return PAGE_EXISTS;
    }

    private double cmToPixel(double cm){

        double inch = this.cmToInch(cm);

        return inch * 72;
    }

    private double cmToInch(double cm){

        return cm / 2.54;
    }

}/*
 * ProgressDialog.java
 *
 * Created on 9 de Março de 2007, 20:54
 */

package br.ufsc.inf.lagar.ui.dialog;

import br.ufsc.inf.lagar.ui.ProgressInterface;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.PrintStream;
import java.text.NumberFormat;
import java.text.SimpleDateFormat;
import java.util.Date;
import javax.swing.Timer;

/**
 *
 * @author  lagar
 */
public class ProgressDialog extends javax.swing.JDialog  implements ActionListener {

    private ProgressInterface pi;
    private Timer timer;
    private boolean exited = false;
    private long time;
    SimpleDateFormat sdf = new SimpleDateFormat("mm:ss");

    /** Creates new form ProgressDialog */
    public ProgressDialog(java.awt.Frame parent, boolean modal, ProgressInterface
pi){
        super(parent, modal);
        initComponents();
        this.timer = new Timer(1000, this);
        this.setProgressTarget(pi);
```

```
}

public void setProgressTarget(ProgressInterface pi){
    this.pi = pi;
    this.jProgressBar1.setValue(0);
    this.jProgressBar1.setMinimum(0);
    this.jProgressBar1.setMaximum(pi.getMaxProgressValue());
}

public void showProgressDialog(){
    time = System.currentTimeMillis();
    this.timer.start();
    this.updateProgress();
    this.setVisible(true);
}

public void exit(){

    if(this.exited) return;

    this.exited = true;
    this.cancelButton.setEnabled(false);
    if(this.timer.isRunning()){
        this.timer.stop();
    }
    this.dispose();
}

private void updateProgress(){
    int value = pi.getProgressValue();
    String text = pi.getProgressText();
    this.jProgressBar1.setValue(value);
    this.jProgressBar1.setString(text);

    double dval = value;
    double max = pi.getMaxProgressValue();
    double perc = ((dval / max) * 100.0);
    int intperc = (int) perc;
    this.setTitle(intperc + "%");

    if(value == pi.getMaxProgressValue() && this.timer.isRunning()){
        this.timer.stop();
    }
}

public void actionPerformed(ActionEvent e) {
    long now = System.currentTimeMillis();
    long diff = now - this.time;
    this.updateProgress();
    Date d = new Date(diff);
    this.timeLabel.setText(sdf.format(d));
}

public javax.swing.JLabel getTitleLabel() {
    return titleLabel;
}

/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this method is
 * always regenerated by the Form Editor.
 */
// <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
```

```
BEGIN:initComponents
    private void initComponents() {
        titleLabel = new javax.swing.JLabel();
        jProgressBar1 = new javax.swing.JProgressBar();
        s1 = new javax.swing.JLabel();
        s2 = new javax.swing.JLabel();
        cancelButton = new javax.swing.JButton();
        timeLabel = new javax.swing.JLabel();

        setDefaultCloseOperation(javax.swing.WindowConstants.DO_NOTHING_ON_CLOSE);
        setTitle("");
        addWindowListener(new java.awt.event.WindowAdapter() {
            public void windowClosing(java.awt.event.WindowEvent evt) {
                formWindowClosing(evt);
            }
        });

        titleLabel.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        titleLabel.setText(bundle.getString("Please_wait_generating_your_cards")); //
NOI18N

        jProgressBar1.setString("");
        jProgressBar1.setStringPainted(true);

        cancelButton.setText(bundle.getString("Cancel")); // NOI18N
        cancelButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cancelButtonActionPerformed(evt);
            }
        });

        timeLabel.setFont(new java.awt.Font("Dialog", 0, 12));
        timeLabel.setForeground(new java.awt.Color(102, 102, 102));
        timeLabel.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
        timeLabel.setText("00:00:00");

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .addContainerGap()
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAIL
ING)
                    .add(timeLabel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
396, Short.MAX_VALUE)
                    .add(org.jdesktop.layout.GroupLayout.LEADING, jProgressBar1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 396, Short.MAX_VALUE)
                    .add(org.jdesktop.layout.GroupLayout.LEADING, titleLabel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 396, Short.MAX_VALUE)
                    .add(org.jdesktop.layout.GroupLayout.LEADING,
layout.createSequentialGroup()
                        .add(s1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 148,
Short.MAX_VALUE)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(cancelButton)
                        .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                        .add(s2, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 149,
Short.MAX_VALUE)))
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(titleLabel)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jProgressBar1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
37, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .add(timeLabel)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASEL
INE)
                    .add(cancelButton)
                    .add(s1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(s2, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );
        pack();
    }// </editor-fold>//GEN-END:initComponents

    private void formWindowClosing(java.awt.event.WindowEvent evt) {//GEN-
FIRST:event_formWindowClosing
        this.pi.cancel();
        this.jProgressBar1.setString(java.util.ResourceBundle.getBundle("resources/bu
ndles/lagarLottery").getString("Cancelled"));
        this.exit();
    }//GEN-LAST:event_formWindowClosing

    private void cancelButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_cancelButtonActionPerformed
        this.pi.cancel();
        this.jProgressBar1.setString(java.util.ResourceBundle.getBundle("resources/bu
ndles/lagarLottery").getString("Cancelled"));
        this.exit();
    }//GEN-LAST:event_cancelButtonActionPerformed



    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton cancelButton;
    private javax.swing.JProgressBar jProgressBar1;
    private javax.swing.JLabel s1;
    private javax.swing.JLabel s2;
    private javax.swing.JLabel timeLabel;
    private javax.swing.JLabel titleLabel;
    // End of variables declaration//GEN-END:variables

}
/*
 * ProgressInterface.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
```

```java
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.ui;

/**
 *
 * @author lagar
 */
public interface ProgressInterface {

    public void cancel();

    public String getProgressText();

    public int getProgressValue();

    public int getMaxProgressValue();

}
/*
 * QuinaGameCardPrinter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.printer;

import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameCardLayout;
import br.ufsc.inf.lagar.main.GameManager;
import java.awt.Graphics;
import java.awt.Graphics2D;
import java.awt.Point;
import java.awt.print.PageFormat;
import java.awt.print.PrinterException;
import java.util.List;

public class QuinaGameCardPrinter extends DefaultGameCardPrinter{
    private GameCardLayout.ExtendedBall markNumber;

    /** Creates a new instance of QuinaGameCardPrinter */
    public QuinaGameCardPrinter() {

    }

    @Override
    protected void printExtendedBalls(GameCard gc, Graphics g, PageFormat pf, int
pageIndex) throws PrinterException {

        if(this.markNumber == null){
            GameCardLayout gcl = GameManager.getCurrentGame().getCardLayout();
            List<GameCardLayout.ExtendedBall> exts = gcl.getExtendedBalls();

            for(GameCardLayout.ExtendedBall ext: exts){
                if(ext.getDescription().equals("Numeros por jogo")){
                    this.markNumber = ext;
                    break;
                }
            }
        }

        Graphics2D g2d = (Graphics2D) g;

        GameCardLayout gcl = GameManager.getCurrentGame().getCardLayout();
        double width = gcl.getBallWidth();
        double height = gcl.getBallHeigth();

        Point.Double p2d = this.markNumber.getBalls().get(gc.getBalls().length);
        /* Now we perform our rendering */
        g2d.fillRect((int)cmToPixel(p2d.x), (int)cmToPixel(p2d.y),
(int)cmToPixel(width), (int)cmToPixel(height));


    }

}
/*
 * QuinaLoader.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.resultloader;

import br.ufsc.inf.lagar.main.GameCard;
import java.io.IOException;
import java.io.InputStream;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
import org.htmlparser.Node;
import org.htmlparser.Parser;
import org.htmlparser.util.NodeList;
import org.htmlparser.util.ParserException;
import org.htmlparser.visitors.TagFindingVisitor;

public class QuinaLoader extends CaixaLoader{

    /** Creates a new instance of QuinaLoader */
    public QuinaLoader() {
        //try {
            //this.resultPath = new
URL("http://www1.caixa.gov.br/loterias/_arquivos/loterias/D_quina.zip");
        //    this.resultPath = getClass().getResource("/D_quina.zip");
        //} catch (MalformedURLException ex) {
        //    ex.printStackTrace();
        //}
    }

    public List<GameCard> loadResults() throws IOException {

        List<GameCard> quinaResults = new ArrayList<GameCard>();
        SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy");

        InputStream in = this.getHtmlStream();

        byte[] buf = new byte[1024];
        StringBuilder sb = new StringBuilder(300);
        int len = 0;

        while((len = in.read(buf)) > 0){

            String s = new String(buf, 0, len);
            sb.append(s);
        }
        in.close();

        try {
            Parser parser = new Parser(sb.toString());
            sb = null;
            String [] tagsToBeFound = {"TR"};
            TagFindingVisitor visitor = new TagFindingVisitor (tagsToBeFound);

            parser.visitAllNodesWith (visitor);
            // First tag specified in search
            Node[] trTags = visitor.getTags(0);

            for(int i = 1; i < trTags.length; ++i){

                Node n = trTags[i];
                NodeList nl = n.getChildren();

                GameCard r = new GameCard(-1, null, 5);

                Node serial = nl.elementAt(1);
                r.setSerial(Integer.parseInt(serial.getChildren().toHtml()));
                Node date = nl.elementAt(3);
                try {
                    r.setDate(sdf.parse(date.getChildren().toHtml()));
                } catch (ParseException ex) {
                    throw new IOException("Error parsing date in result xml");
                }
                int index = 0;
                for(int j = 5; j <= 13; j+=2 ){

                    Node ball = nl.elementAt(j);
                    r.setBall((byte)Integer.parseInt(ball.getChildren().toHtml()),
index);
                    ++index;
                }
                Arrays.sort(r.getBalls());

                quinaResults.add(r);
            }

        } catch (ParserException ex) {
            ex.printStackTrace();
        }

        return quinaResults;
    }

}
/*
 * QuinaLoaderTest.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.persistence.rs;

import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.resultloader.QuinaLoader;
import br.ufsc.inf.lagar.resultloader.ResultLoader;
import java.io.IOException;
import java.util.List;
import junit.framework.*;
import java.net.URL;

/**
 *
 * @author lagar
 */
```

```
public class QuinaLoaderTest extends TestCase {


    public void testQuinaLoader() throws IOException{

        ResultLoader loader = new QuinaLoader();

        List<GameCard> quinaResults = loader.loadResults();

        for(GameCard d: quinaResults){
            System.out.println(d);
        }
    }

}
/*
 * RadioPanel.java
 *
 * Created on 12 de Abril de 2007, 20:32
 */

package br.ufsc.inf.lagar.ui.panels;

/**
 *
 * @author  lagar
 */
public class RadioPanel extends javax.swing.JPanel {

    /**
     * Creates new form RadioPanel
     */
    public RadioPanel() {
        initComponents();
    }

    public javax.swing.JRadioButton getRadioButton() {
        return radioButton;
    }

    public void setRadioButton(javax.swing.JRadioButton radioButton) {
        this.radioButton = radioButton;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        radioButton = new javax.swing.JRadioButton();

        setMaximumSize(new java.awt.Dimension(32767, 39));
        radioButton.setText("jRadioButton1");
        radioButton.setBorder(javax.swing.BorderFactory.createEmptyBorder(0, 0, 0,
0));
        radioButton.setMargin(new java.awt.Insets(0, 0, 0, 0));

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(radioButton, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 277,
Short.MAX_VALUE)
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(radioButton)
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
    }// </editor-fold>//GEN-END:initComponents


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JRadioButton radioButton;
    // End of variables declaration//GEN-END:variables

}
/*
 * ResultLoader.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.resultloader;

import br.ufsc.inf.lagar.main.GameCard;
import java.io.IOException;
import java.net.MalformedURLException;
import java.util.List;


public interface ResultLoader {

    public List<GameCard> loadResults() throws IOException;

    public void setResultPath(String url) throws MalformedURLException;

    public String getResultPath();

}
package br.ufsc.inf.lagar.utils;
```

```java
/**
 * RTSI.java
 *
 * Created: Wed Jan 24 11:15:02 2001
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 *
 */

import java.io.*;
import java.net.URL;
import java.net.JarURLConnection;
import java.util.ArrayList;
import java.util.jar.*;
import java.util.zip.*;
import java.util.Enumeration;

/**
 * This utility class is looking for all the classes implementing or
 * inheriting from a given interface or class.
 * (RunTime Subclass Identification)
 *
 * Modified by Euclides (lagar)
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 *
 * @author <a href="mailto:daniel@satlive.org">Daniel Le Berre</a>
 * @version 1.0
 */
public class RTSI {


    /**
     * Display all the classes inheriting or implementing a given
     * class in the currently loaded packages.
     * @param tosubclassname the name of the class to inherit from
     */
    public static void find(String tosubclassname) {
        try {
            Class tosubclass = Class.forName(tosubclassname);
            Package [] pcks = Package.getPackages();
            for (int i=0;i<pcks.length;i++) {
                find(pcks[i].getName(),tosubclass);
            }
        } catch (ClassNotFoundException ex) {
            System.err.println("Class "+tosubclassname+" not found!");
        }
    }

    /**
     * Display all the classes inheriting or implementing a given
     * class in a given package.
     * @param pckgname the fully qualified name of the package
     * @param tosubclass the name of the class to inherit from
     * @return list with the names (String) of classes found
     */
    public static ArrayList<String> find(String pckname, String tosubclassname) {
        try {
            Class tosubclass = Class.forName(tosubclassname);
            return find(pckname,tosubclass);
        } catch (ClassNotFoundException ex) {
            System.err.println("Class "+tosubclassname+" not found!");
        }
        return new ArrayList<String>();
    }

    /**
     * Display all the classes inheriting or implementing a given
     * class in a given package.
     * @param pckgname the fully qualified name of the package
     * @param tosubclass the Class object to inherit from
     * @return list with the names (String) of classes found
     */
    public static ArrayList<String> find(String pckgname, Class tosubclass) {

        // Code from lagar
          ArrayList<String> result = new ArrayList<String>();

        // Code from JWhich
        // ======
        // Translate the package name into an absolute path
        String name = new String(pckgname);
        if (!name.startsWith("/")) {
            name = "/" + name;
        }
        name = name.replace('.','/');

        // Get a File object for the package
        URL url = RTSI.class.getResource(name);
        // URL url = tosubclass.getResource(name);
        // URL url = ClassLoader.getSystemClassLoader().getResource(name);
        //SysSystem.out.println(name+"->"+url);

        // Happens only if the jar file is not well constructed, i.e.
        // if the directories do not appear alone in the jar file like here:
        //
        //          meta-inf/
        //          meta-inf/manifest.mf
        //          commands/                  <== IMPORTANT
        //          commands/Command.class
        //          commands/DoorClose.class
        //          commands/DoorLock.class
        //          commands/DoorOpen.class
        //          commands/LightOff.class
        //          commands/LightOn.class
        //          RTSI.class
        //
        if (url==null) return result;

        String urlDir = url.getFile();
        urlDir = urlDir.replaceAll("%20", " ");
        File directory = new File(urlDir);

        // New code
        // ======
        if (directory.exists()) {
            // Get the list of the files contained in the package
            String [] files = directory.list();
            for (int i=0;i<files.length;i++) {

                // we are only interested in .class files
                if (files[i].endsWith(".class")) {
                    // removes the .class extension
                    String classname = files[i].substring(0,files[i].length()-6);
                    try {
                        // Try to create an instance of the object
                        Object o =
Class.forName(pckgname+"."+classname).newInstance();
                        if (tosubclass.isInstance(o)) {
                            //System.out.println(classname);
                            result.add(classname);
                        }
                    } catch (ClassNotFoundException cnfex) {
                        System.err.println(cnfex);
                    } catch (InstantiationException iex) {
                        // We try to instanciate an interface
                        // or an object that does not have a
                        // default constructor
                    } catch (IllegalAccessException iaex) {
                        // The class is not public
                    }
                }
            }
        } else {
            try {
                // It does not work with the filesystem: we must
                // be in the case of a package contained in a jar file.
                JarURLConnection conn = (JarURLConnection)url.openConnection();
                String starts = conn.getEntryName();
                JarFile jfile = conn.getJarFile();
                Enumeration e = jfile.entries();
                while (e.hasMoreElements()) {
                    ZipEntry entry = (ZipEntry)e.nextElement();
                    String entryname = entry.getName();
                    if (entryname.startsWith(starts)
                        &&(entryname.lastIndexOf('/')<=starts.length())
                        &&entryname.endsWith(".class")) {
                        String classname = entryname.substring(0,entryname.length()-
6);
                        if (classname.startsWith("/"))
                            classname = classname.substring(1);
                        classname = classname.replace('/','.');
                        try {
                            // Try to create an instance of the object
                            Object o = Class.forName(classname).newInstance();
                            if (tosubclass.isInstance(o)) {
                                //System.out.println(classname.substring(classname.la
stIndexOf('.')+1));
                                result.add(classname.substring(classname.lastIndexOf(
'.')+1));
                            }
                        } catch (ClassNotFoundException cnfex) {
                            System.err.println(cnfex);
                        } catch (InstantiationException iex) {
                            // We try to instanciate an interface
                            // or an object that does not have a
                            // default constructor
                        } catch (IllegalAccessException iaex) {
                            // The class is not public
                        }
                    }
                }
            } catch (IOException ioex) {
                System.err.println(ioex);
            }
        }

        return result;
    }

    /**
     * Display all the files in a package.
     * @param pckgname the fully qualified name of the package
     * @param fileFilter the file suffix ex: xml - null if want all files
     * @return list with the names (String) of files found
     */
    public static ArrayList<String> findFiles(String pckgname, String fileFilter) {

        // Code from lagar
          ArrayList<String> result = new ArrayList<String>();

        // Code from JWhich
        // ======
        // Translate the package name into an absolute path
        String name = new String(pckgname);
        if (!name.startsWith("/")) {
            name = "/" + name;
        }
        name = name.replace('.','/');

        // Get a File object for the package
        URL url = RTSI.class.getResource(name);
        // URL url = tosubclass.getResource(name);
        // URL url = ClassLoader.getSystemClassLoader().getResource(name);
        //SysSystem.out.println(name+"->"+url);

        // Happens only if the jar file is not well constructed, i.e.
        // if the directories do not appear alone in the jar file like here:
        //
        //          meta-inf/
        //          meta-inf/manifest.mf
        //          commands/                  <== IMPORTANT
        //          commands/Command.class
        //          commands/DoorClose.class
        //          commands/DoorLock.class
        //          commands/DoorOpen.class
        //          commands/LightOff.class
        //          commands/LightOn.class
        //          RTSI.class
        //
        if (url==null) return result;

        String urlDir = url.getFile();
        urlDir = urlDir.replaceAll("%20", " ");
        File directory = new File(urlDir);

        // New code
        // ======
        if (directory.exists()) {
            // Get the list of the files contained in the package
            String [] files = directory.list();
            for (int i=0;i<files.length;i++) {

                if(fileFilter != null){

                    if (files[i].toLowerCase().endsWith(fileFilter.toLowerCase())) {
                        result.add(files[i]);
                    }
                }
                else{
                    result.add(files[i]);
                }


            }
        } else {
            try {
                // It does not work with the filesystem: we must
                // be in the case of a package contained in a jar file.
                JarURLConnection conn = (JarURLConnection)url.openConnection();
                String starts = conn.getEntryName();
                JarFile jfile = conn.getJarFile();
                Enumeration e = jfile.entries();
                while (e.hasMoreElements()) {
                    ZipEntry entry = (ZipEntry)e.nextElement();
                    String entryname = entry.getName();
                    if (entryname.startsWith(starts)
                        &&(entryname.lastIndexOf('/')<=starts.length())) {

                       if(fileFilter != null){

                            if
(entryname.toLowerCase().endsWith(fileFilter.toLowerCase())) {
                                result.add(entryname);
                            }
                        }
                        else{
                            result.add(entryname);
                        }

                    }
                }
            } catch (IOException ioex) {
                System.err.println(ioex);
            }
        }

        return result;
    }

}// RTSI
/*
 * SerialCardFilter.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter.gamefilter;

import br.ufsc.inf.lagar.filter.AbstractGameFilter;
import br.ufsc.inf.lagar.filter.TextInputFilterException;
import br.ufsc.inf.lagar.main.GameCard;
import org.apache.commons.lang.StringUtils;

public class SerialCardFilter extends AbstractGameFilter {

    private int jump;

    /** Creates a new instance of SerialCardFilter */
    public SerialCardFilter() {
    }

    public void validateInput(String input) throws TextInputFilterException {

        try{
            String[] arr = input.split("=");
            for(int i = 0; i < arr.length; ++i){

                arr[i] = StringUtils.deleteWhitespace(arr[i]);
            }
            this.jump = Integer.parseInt(arr[1]);
        }
        catch(Exception ex){
            throw new TextInputFilterException(ex);
        }

    }

    public String getShortName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Serial_filter");
    }

    public String getFullName() {
        return this.getShortName();
    }

    public String getDescription() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Serial_filter_description");
    }

    public String getFilterHelp() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/filter/gamefilter/filtersTr").g
etString("Serial_filter_help");
    }

    public boolean filter(GameCard card) {

        int serial = card.getSerial();

        if(serial < this.jump) return !(this.isMode());

        if(serial % this.jump == 0) return this.isMode();

        return !(this.isMode());
    }

}
/*
 * SimpleVerificator.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.verify.verificators;

import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.ui.dialog.DecisionDialog;
import br.ufsc.inf.lagar.verify.VerificatorInterface;
import java.io.File;
import java.io.IOException;
import java.text.NumberFormat;
import java.util.Arrays;
import javax.swing.JEditorPane;
import javax.swing.JScrollPane;
import javax.swing.SwingUtilities;

public class SimpleVerificator implements VerificatorInterface {

    private GameCard result;
    private File cardsFile;
    private NumberFormat nf;
    private NumberFormat nfb;

    /** Creates a new instance of SimpleVerificator */
    public SimpleVerificator() {
        nf = NumberFormat.getIntegerInstance();
        nf.setMinimumIntegerDigits(4);
        nfb = NumberFormat.getIntegerInstance();
        nfb.setMinimumIntegerDigits(2);
    }

    public void setCards(File llgFile) throws IOException {
        this.cardsFile = llgFile;
    }

    public String getVName() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/verify/verificators/verificator
sTr").getString("Simple_verificator");
    }

    public String getMoreInfo() {
        return
java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/verify/verificators/verificator
sTr").getString("todo_simple_verificator_more_info");
    }

    public void runVerificator() {

        final DecisionDialog dd = new DecisionDialog(null, false);
        dd.getTitlePanel().setTitle(this.getVName());
        dd.getButtonsPanel().setVisible(false);
        JEditorPane editor = new JEditorPane("text/html", "");
        JScrollPane scroll = new JScrollPane(
                editor,
                JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);

        dd.getMainPanel().add(scroll, scroll.getClass().getName());

        editor.setText(this.computeResult());

        scroll.revalidate();
        dd.repaint();

        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                dd.setVisible(true);
            }
        });
    }

    private String computeResult(){

        GameConfiguration gc = GameManager.getCurrentGame();

        StringBuilder sb = new StringBuilder();
        sb.append("<html>");
        GenCardIO io = null;
        try{
            io = new GenCardIO(this.cardsFile);

            int[] cats = new int[gc.getNumbersPerDrawing()+1];
            while(io.hasNext()){
                GameCard gameCard = io.next();
                int hits = this.computeHits(gameCard);
                ++cats[hits];

                this.writeGame(sb, gameCard, hits);

            }

            this.writeSummary(sb, cats);

        }
        catch(IOException ioex){

            ioex.printStackTrace();
            sb.append(java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/verify/ve
rificators/verificatorsTr").getString("Error_computing_results"));
            return sb.toString();
        }
        finally {
            sb.append("</html>");
            if(io != null) io.close();
        }
        return sb.toString();
    }

    private int computeHits(GameCard currGame){

        byte[] r = this.result.getBalls();
        byte[] g = currGame.getBalls();

        int hits = 0;

        for(int i = 0; i < g.length; ++i){
            if(Arrays.binarySearch(r, g[i]) >= 0){
                ++hits;
            }
        }

        return hits;
    }

    private void writeGame(StringBuilder sb, GameCard gc, int hits){

        sb.append("<b>");
        sb.append(nf.format(gc.getSerial()));
        sb.append("</b>");
        sb.append(" :  ");
        sb.append(nfb.format(hits));
        sb.append(java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/verify/verifi
cators/verificatorsTr").getString("_hits"));
        sb.append("<br><br>");
    }

    private void writeSummary(StringBuilder sb, int[] hits){

        sb.append("<b>");
        sb.append(java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/verify/verifi
cators/verificatorsTr").getString("===========_Summary_==========="));
        sb.append("</b>");
        sb.append("<br>");
        sb.append(java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/verify/verifi
cators/verificatorsTr").getString("Hits_x_number_of_cards"));
        sb.append("<br><br>");
        for(int i = 0; i < hits.length; ++i){
            sb.append(nfb.format(i));
            sb.append(" ");
            sb.append(java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/verify/ve
rificators/verificatorsTr").getString("hits"));
            sb.append(" = ");
            sb.append(nfb.format(hits[i]));
            sb.append(" ");
            sb.append(java.util.ResourceBundle.getBundle("br/ufsc/inf/lagar/verify/ve
rificators/verificatorsTr").getString("cards."));
            sb.append("<br>");
        }
    }

    public void setResult(GameCard gc){
        this.result = gc;
    }

}
/*
 * SingleFilterPanel.java
 *
 * Created on 29 de Março de 2007, 21:36
 */

package br.ufsc.inf.lagar.ui.panels.filter;

import br.ufsc.inf.lagar.filter.Filterable;
import java.awt.Color;
import java.awt.Component;
import java.awt.event.MouseListener;
import javax.swing.border.LineBorder;

/**
 *
 * @author  lagar
 */
public class SingleFilterPanel extends javax.swing.JPanel {

    private Filterable filter;
    private FilterSelectPanel parentFp;
    private int position;
    private String ACCEPT = "Accept";
    private String REJECT = "Reject";

    /** Creates new form SingleFilterPanel */
    public SingleFilterPanel(Filterable f, int position, FilterSelectPanel parentFp)
{
        initComponents();
        this.parentFp = parentFp;
        this.filter = f;
        this.setPosition(position);
        this.numberLabel.setText(String.valueOf(position) + "- ");
        this.filterLabel.setText(f.getShortName());

        this.acRjCombo.addItem(ACCEPT);
        this.acRjCombo.addItem(REJECT);
        this.acRjCombo.setSelectedItem(REJECT);

        //mouse listener
        MouseListener ml = this.getMouseListeners()[0];
        Component[] cs = this.acRjCombo.getComponents();
        for (int i = 0; i < cs.length; i++) {
            cs[i].addMouseListener(ml);
        }
        this.getFilterField().addMouseListener(ml);
    }

    public Filterable getFilter() {
        return filter;
    }

    public int getPosition() {
        return position;
    }

    public void setPosition(int position) {
        this.position = position;
        this.numberLabel.setText(position + "- ");
    }

    public javax.swing.JTextField getFilterField() {
        return filterField;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        filterLabel = new javax.swing.JLabel();
        filterField = new javax.swing.JTextField();
        acRjCombo = new javax.swing.JComboBox();
        numberLabel = new javax.swing.JLabel();

        setMaximumSize(new java.awt.Dimension(32767, 50));
        setMinimumSize(new java.awt.Dimension(200, 50));
        addMouseListener(new java.awt.event.MouseAdapter() {
            public void mousePressed(java.awt.event.MouseEvent evt) {
                formMousePressed(evt);
            }
        });

        filterLabel.setText("Filter name");

        acRjCombo.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                acRjComboActionPerformed(evt);
            }
        });

        numberLabel.setText("1");

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .addContainerGap()
                .add(numberLabel)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(filterLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 70,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(acRjCombo, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(filterField, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 223,
Short.MAX_VALUE)
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.BASEL
INE)
                    .add(numberLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
24, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(filterLabel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
26, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(acRjCombo, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
24, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(filterField, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
26, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addContainerGap(org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE))
        );
    }// </editor-fold>//GEN-END:initComponents

    private void acRjComboActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_acRjComboActionPerformed
        String selectedItem = this.acRjCombo.getSelectedItem().toString();
        this.filter.setAcceptReject(selectedItem.equals(ACCEPT));
    }//GEN-LAST:event_acRjComboActionPerformed

    private void formMousePressed(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_formMousePressed
        this.parentFp.setSelectedFilter(this);
    }//GEN-LAST:event_formMousePressed


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JComboBox acRjCombo;
    private javax.swing.JTextField filterField;
    private javax.swing.JLabel filterLabel;
    private javax.swing.JLabel numberLabel;
    // End of variables declaration//GEN-END:variables

}
/*
 * TableAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis;

import java.awt.CardLayout;
import java.awt.FlowLayout;
import java.util.ArrayList;
import java.util.Collections;
import java.util.Enumeration;
import java.util.List;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.ToolTipManager;
import javax.swing.event.TableModelListener;
import javax.swing.table.TableColumn;
import javax.swing.table.TableModel;
import org.jfree.chart.JFreeChart;

public abstract class TableAnalysis extends Analysis implements TableModel{

    /**
     * Creates a new instance of TableAnalysis
     */
    public TableAnalysis() {
    }

    public abstract int getRowCount();

    public abstract int getColumnCount();

    public abstract String getColumnName(int columnIndex);

    public abstract CellData getCellDataAt(int rowIndex, int columnIndex);

    public List<JPanel> getPanels() {
        JPanel panel = new JPanel();
        panel.setLayout(new CardLayout());
        panel.setName("Table");

        JScrollPane scroll = new JScrollPane(getTable(),
                JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                JScrollPane.HORIZONTAL_SCROLLBAR_AS_NEEDED);

        panel.add(scroll, "scroll");

        List<JPanel> panels = new ArrayList<JPanel>();
        panels.add(panel);

        return panels;
    }

    protected JTable getTable(){

        JTable table = new JTable(this);
        table.setAutoResizeMode( JTable.AUTO_RESIZE_OFF );
        table.setDefaultRenderer(CellData.class, new CellDataRenderer());

        Enumeration<TableColumn> columns = table.getColumnModel().getColumns();
        while(columns.hasMoreElements()){
            TableColumn t = columns.nextElement();
            t.setMinWidth(20);
            t.setPreferredWidth(43);
            //t.setResizable(false);
        }

        return table;
    }

    public Class<?> getColumnClass(int columnIndex) {
        return CellData.class;
    }

    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return false;
    }

    public Object getValueAt(int rowIndex, int columnIndex) {
        return this.getCellDataAt(rowIndex, columnIndex);
    }

    public void setValueAt(Object aValue, int rowIndex, int columnIndex) {
    }

    public void addTableModelListener(TableModelListener l) {
    }

    public void removeTableModelListener(TableModelListener l) {
    }


}
/*
 * TableChartAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.analysis;

import java.util.List;
import javax.swing.JPanel;
import org.jfree.chart.ChartPanel;
import org.jfree.chart.JFreeChart;
```

```java
public abstract class TableChartAnalysis extends TableAnalysis implements Chartable
{

    public List<JPanel> getPanels() {

        List<JPanel> panels = super.getPanels();
        JFreeChart chart = getChart();

        if(chart != null){
            ChartPanel panel = new ChartPanel(getChart());
            panel.setName("Chart");
            panels.add(panel);
        }


        return panels;
    }

}
/*
 * TermsDialog.java
 *
 * Created on 16 de Maio de 2007, 22:30
 */

package br.ufsc.inf.lagar.main;

import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;

/**
 *
 * @author  lagar
 */
public class TermsDialog extends javax.swing.JDialog {

    /** Creates new form TermsDialog */
    public TermsDialog(java.awt.Frame parent, boolean modal) {
        super(parent, modal);
        initComponents();
        this.setLocationRelativeTo(null);

        try{
            InputStream is = getClass().getResourceAsStream("/resources/LICENSE");
            BufferedReader r = new BufferedReader(new InputStreamReader(is));
            StringBuilder sb = new StringBuilder();
            String line = null;
            while( (line = r.readLine()) != null ){
                sb.append(line);
                sb.append("\n");
            }
            this.jTextArea1.setText(sb.toString());
        }
        catch(Exception ex){
            this.jTextArea1.setText("See file LICENSE");
        }

    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        jScrollPane1 = new javax.swing.JScrollPane();
        jTextArea1 = new javax.swing.JTextArea();

        setDefaultCloseOperation(javax.swing.WindowConstants.DISPOSE_ON_CLOSE);
        setTitle("Terms");
        jTextArea1.setColumns(20);
        jTextArea1.setEditable(false);
        jTextArea1.setLineWrap(true);
        jTextArea1.setRows(5);
        jTextArea1.setWrapStyleWord(true);
        jScrollPane1.setViewportView(jTextArea1);

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 600,
Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 400,
Short.MAX_VALUE)
        );
        pack();
    }// </editor-fold>//GEN-END:initComponents

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new TermsDialog(new javax.swing.JFrame(), true).setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTextArea jTextArea1;
    // End of variables declaration//GEN-END:variables

}
/*
 * TesteMain.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.verify.animated1.osc;

import java.io.ByteArrayInputStream;
import java.io.ByteArrayOutputStream;
import java.io.InputStream;

public class TesteMain {

    /** Creates a new instance of TesteMain */
    public TesteMain() {
    }

    public static void main(String[] args) throws Exception{

        InputStream is =
TesteMain.class.getResourceAsStream("/resources/snd/pop.au");
        ByteArrayOutputStream bos = new ByteArrayOutputStream();
        byte[] buf = new byte[8192];
        int n = 0;
        while((n = is.read(buf)) != -1){
            bos.write(buf, 0, n);
        }

        byte[] arrFinal = bos.toByteArray();
        ByteArrayInputStream ohh = new ByteArrayInputStream(arrFinal);

        int i = 0;
        while(i < 5){
            AudioPlayer.main(ohh);
            ++i;
            //is = TesteMain.class.getResourceAsStream("/resources/snd/pop.au");
            ohh = new ByteArrayInputStream(arrFinal);
        }

    }

}
/*
 * TestOddEvenAnalysis.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.zmisc.test;

import br.ufsc.inf.lagar.analysis.CellDataRenderer;
import br.ufsc.inf.lagar.analysis.gameanalysis.OddEvenAnalysis;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.ui.dialog.DecisionDialog;
import java.io.IOException;
import java.util.Enumeration;
import javax.swing.JScrollPane;
import javax.swing.JTable;
import javax.swing.ToolTipManager;
import javax.swing.table.TableColumn;
import br.ufsc.inf.lagar.analysis.CellData;
import br.ufsc.inf.lagar.analysis.TableChartAnalysis;
import br.ufsc.inf.lagar.main.GameResultIterator;
import java.util.ArrayList;
import java.util.List;
import org.jfree.chart.JFreeChart;

public class TestOddEvenAnalysis {

    /** Creates a new instance of TestOddEvenAnalysis */
    public TestOddEvenAnalysis() {
    }

    public static void testTable() throws IOException{

        DecisionDialog d = new DecisionDialog(null, true);

        List<GameConfiguration> availableGames = GameManager.getAvailableGames();
        GameConfiguration gc = availableGames.get(0);
        gc.getGameResults().loadResults();


        OddEvenAnalysis instance = new OddEvenAnalysis();
        instance.setBegin(1);
        instance.setEnd(179);
        instance.setJump(1);
        instance.setCurrentGameConfiguration(gc);
        instance.calculateAnalysis();

        JTable table = new JTable(instance);
        table.setAutoResizeMode( JTable.AUTO_RESIZE_OFF );
        table.setDefaultRenderer(CellData.class, new CellDataRenderer());

        //table.setPreferredScrollableViewportSize(new Dimension(20,20));
//        table.createDefaultColumnsFromModel();

        Enumeration<TableColumn> columns = table.getColumnModel().getColumns();
        while(columns.hasMoreElements()){
            TableColumn t = columns.nextElement();
            t.setMinWidth(50);
            t.setPreferredWidth(50);
            //t.setResizable(false);
        }

        JScrollPane pane = new JScrollPane(table,
JScrollPane.VERTICAL_SCROLLBAR_AS_NEEDED,
                                                  JScrollPane.HORIZONTAL_SCROLLBAR_AS
_NEEDED);

        ToolTipManager.sharedInstance().unregisterComponent(table);
        ToolTipManager.sharedInstance().unregisterComponent(table.getTableHeader());

        d.getMainPanel().add(pane, "table");

        d.pack();
        d.setResizable(true);
        d.setVisible(true);

    }

    public static void main(String[] args) throws IOException{
        testTable();
    }

}
/*
 * TextInputFilterException.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.filter;

public class TextInputFilterException extends Exception{

    /** Creates a new instance of TextInputFilterException */
    public TextInputFilterException() {
    }

    public TextInputFilterException(String message){
        super(message);
    }

    public TextInputFilterException(String message, Throwable t){
        super(message,t);
    }

    public TextInputFilterException(Throwable t){
        super(t);
    }

}
/*
 * UpdateDbPanel.java
 *
 * Created on 8 de Abril de 2007, 16:58
 */

package br.ufsc.inf.lagar.ui.panels;

/**
 *
 * @author  lagar
 */
public class UpdateDbPanel extends javax.swing.JPanel {

    /** Creates new form UpdateDbPanel */
    public UpdateDbPanel() {
        initComponents();
    }

    public javax.swing.JLabel getInfoLabel() {
        return infoLabel;
    }

    public javax.swing.JTextField getUrlField() {
        return urlField;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        infoLabel = new javax.swing.JLabel();
        fromUrlPanel = new javax.swing.JPanel();
        urlField = new javax.swing.JTextField();

        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        fromUrlPanel.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.ge
tString("update_from_url"))); // NOI18N

        org.jdesktop.layout.GroupLayout fromUrlPanelLayout = new
org.jdesktop.layout.GroupLayout(fromUrlPanel);
        fromUrlPanel.setLayout(fromUrlPanelLayout);
        fromUrlPanelLayout.setHorizontalGroup(
            fromUrlPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
            .add(urlField, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 498,
Short.MAX_VALUE)
        );
        fromUrlPanelLayout.setVerticalGroup(
            fromUrlPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
            .add(urlField, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 31,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
        );

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .addContainerGap()
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAIL
ING)
                    .add(org.jdesktop.layout.GroupLayout.LEADING, fromUrlPanel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .add(org.jdesktop.layout.GroupLayout.LEADING, infoLabel,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 508, Short.MAX_VALUE))
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(infoLabel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 83,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(fromUrlPanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
    }// </editor-fold>//GEN-END:initComponents


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JPanel fromUrlPanel;
    private javax.swing.JLabel infoLabel;
    private javax.swing.JTextField urlField;
    // End of variables declaration//GEN-END:variables

}
/*
 * VerificatorInterface.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.verify;

import br.ufsc.inf.lagar.main.GameCard;
import java.io.File;
import java.io.IOException;

/**
 *
 * @author lagar
 */
public interface VerificatorInterface {

    public void setCards(File llgFile) throws IOException;

    public void setResult(GameCard gc);

    public String getVName();

    public String getMoreInfo();

    public void runVerificator();

}
/*
 * VerificatorPanel.java
 *
 * Created on 12 de Abril de 2007, 20:23
 */

package br.ufsc.inf.lagar.ui.panels;

import br.ufsc.inf.lagar.generator.GenCardIO;
import br.ufsc.inf.lagar.main.GameCard;
import br.ufsc.inf.lagar.main.GameConfiguration;
import br.ufsc.inf.lagar.main.GameManager;
import br.ufsc.inf.lagar.utils.RTSI;
import br.ufsc.inf.lagar.verify.VerificatorInterface;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.File;
import java.io.IOException;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.HashMap;
import javax.swing.JOptionPane;
import javax.swing.JRadioButton;
import org.apache.commons.lang.StringUtils;

/**
 *
 * @author  lagar
 */
public class VerificatorPanel extends javax.swing.JPanel implements ActionListener{

    private HashMap<String, VerificatorInterface> vs = new HashMap<String,
VerificatorInterface>();
    private String selection;
    private GameCard result;

    /** Creates new form VerificatorPanel */
    public VerificatorPanel() {
        initComponents();
        this.fileChooserLLG1.setMode(FileChooserLLG.ChooserMode.LOAD_MODE);
    }

    public void loadAvailableVerificators() throws Exception{

        GameConfiguration current = GameManager.getCurrentGame();

        String packageName = "br.ufsc.inf.lagar.verify.verificators";

        boolean first = true;
        ArrayList<String> classes = RTSI.find(packageName,
VerificatorInterface.class);
        for(String clazz: classes){

            clazz = packageName+"."+clazz;
            VerificatorInterface verificator = (VerificatorInterface)
Class.forName(clazz).newInstance();

            this.vs.put(verificator.getVName(), verificator);
            RadioPanel rp = new RadioPanel();
            rp.getRadioButton().setText(verificator.getVName());
            rp.getRadioButton().setActionCommand(verificator.getVName());
            rp.getRadioButton().addActionListener(this);

            this.buttonGroup1.add(rp.getRadioButton());

            this.versPanel.add(rp);

            if(first){
                rp.getRadioButton().setSelected(true);
                first = false;
                this.selection = verificator.getVName();
            }

        }

        this.jScrollPane1.revalidate();
        this.repaint();


    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        buttonGroup1 = new javax.swing.ButtonGroup();
        simpleInternalFrame1 = new br.inf.ufsc.lagar.swingutil.SimpleInternalFrame();
        jScrollPane1 = new javax.swing.JScrollPane();
        versPanel = new javax.swing.JPanel();
        labelSepRight = new javax.swing.JLabel();
        labelSepLeft = new javax.swing.JLabel();
        infoButton = new javax.swing.JButton();
        runButton = new javax.swing.JButton();
        fileChooserLLG1 = new br.ufsc.inf.lagar.ui.panels.FileChooserLLG();
        jPanel1 = new javax.swing.JPanel();
        jScrollPane2 = new javax.swing.JScrollPane();
        resultArea = new javax.swing.JTextArea();

        java.util.ResourceBundle bundle =
java.util.ResourceBundle.getBundle("resources/bundles/lagarLottery"); // NOI18N
        simpleInternalFrame1.setTitle(bundle.getString("Verificators")); // NOI18N

        jScrollPane1.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.ge
tString("Verificators_impls"))); // NOI18N
        versPanel.setLayout(new javax.swing.BoxLayout(versPanel,
javax.swing.BoxLayout.Y_AXIS));
```

```
        jScrollPane1.setViewportView(versPanel);

        infoButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/info.png")));
        infoButton.setText(bundle.getString("Info")); // NOI18N
        infoButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                infoButtonActionPerformed(evt);
            }
        });

        runButton.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/resources/16x16/runprog.png")));
        runButton.setText(bundle.getString("Run")); // NOI18N
        runButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                runButtonActionPerformed(evt);
            }
        });

        fileChooserLLG1.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle
.getString("Game_to_verify"))); // NOI18N

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(bundle.getStri
ng("Result"))); // NOI18N
        jPanel1.setMinimumSize(new java.awt.Dimension(100, 60));
        jPanel1.setPreferredSize(new java.awt.Dimension(100, 60));
        resultArea.setColumns(20);
        resultArea.setLineWrap(true);
        resultArea.setRows(3);
        resultArea.setWrapStyleWord(true);
        jScrollPane2.setViewportView(resultArea);

        org.jdesktop.layout.GroupLayout jPanel1Layout = new
org.jdesktop.layout.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jScrollPane2, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 535,
Short.MAX_VALUE)
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING
)
            .add(jScrollPane2, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 54,
Short.MAX_VALUE)
        );

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(layout.createSequentialGroup()
                .add(labelSepLeft, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 178,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(runButton)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(infoButton)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(labelSepRight, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
179, Short.MAX_VALUE))
```

```
            .add(simpleInternalFrame1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
545, Short.MAX_VALUE)
            .add(fileChooserLLG1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 545,
Short.MAX_VALUE)
            .add(jPanel1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 545,
Short.MAX_VALUE)
            .add(org.jdesktop.layout.GroupLayout.TRAILING, jScrollPane1,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 545, Short.MAX_VALUE)
        );

        layout.linkSize(new java.awt.Component[] {infoButton, runButton},
org.jdesktop.layout.GroupLayout.HORIZONTAL);

        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .add(simpleInternalFrame1,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(fileChooserLLG1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jPanel1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 78,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(jScrollPane1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 193,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.TRAIL
ING)
                    .add(layout.createParallelGroup(org.jdesktop.layout.GroupLayout.B
ASELINE)
                        .add(infoButton)
                        .add(runButton))
                    .add(labelSepRight,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .add(labelSepLeft,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 25,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
                .addContainerGap())
        );

        layout.linkSize(new java.awt.Component[] {labelSepLeft, labelSepRight},
org.jdesktop.layout.GroupLayout.VERTICAL);

        layout.linkSize(new java.awt.Component[] {infoButton, runButton},
org.jdesktop.layout.GroupLayout.VERTICAL);

    }// </editor-fold>//GEN-END:initComponents

    private void runButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_runButtonActionPerformed
        if(this.validateInput()){
            this.processRunButton();
        }
    }//GEN-LAST:event_runButtonActionPerformed

    private void infoButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_infoButtonActionPerformed
```

```
        this.processInfoButton();
    }//GEN-LAST:event_infoButtonActionPerformed

    private void processRunButton() {

        final VerificatorInterface vi = this.vs.get(this.selection);
        vi.setResult(this.result);
        try {
            vi.setCards(new File(this.fileChooserLLG1.getFilePath()));
        } catch (IOException ex) {
            ex.printStackTrace();
        }

        Thread t = new Thread(new Runnable() {
            public void run() {
                vi.runVerificator();
            }
        });
        t.start();
    }

    private void processInfoButton() {
        throw new UnsupportedOperationException("Not yet implemented");
    }

    public void actionPerformed(ActionEvent e) {
        this.selection = e.getActionCommand();
    }

    private boolean validateInput() {

        GameConfiguration gc = GameManager.getCurrentGame();

        String filePath = this.fileChooserLLG1.getFilePath();
        GenCardIO io = null;
        try {

            io = new GenCardIO(new File(filePath));

            String result = this.resultArea.getText();
            String[] rn = result.trim().split("-");
            byte[] arr = new byte[rn.length];
            for(int i = 0; i < rn.length; ++i){
                String s = StringUtils.deleteWhitespace(rn[i]);
                arr[i] = Byte.parseByte(s);
            }
            Arrays.sort(arr);

            if(gc.getNumbersPerDrawing() != arr.length){
                JOptionPane.showMessageDialog(
                    this,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Invalid_result._You_must_type_the_correct_amount_of_numbers."),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
                return false;
            }

            if(arr[arr.length-1] > gc.getNumbersPerCard()){
                JOptionPane.showMessageDialog(
                    this,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Invalid_result._You_typed_one_ball_that_is_impossible_to_occur."),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
                return false;
            }

            this.result = new GameCard(0,null,arr.length);
            this.result.setBalls(arr);

        } catch (IOException ex) {
            ex.printStackTrace();
            JOptionPane.showMessageDialog(
                    this,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Invalid_file"),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
            return false;
        }
        catch (NumberFormatException nfex) {
            nfex.printStackTrace();
            JOptionPane.showMessageDialog(
                    this,
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Invalid_result._Use_character_'-'_as_a_separator."),
                    java.util.ResourceBundle.getBundle("resources/bundles/lagarLotter
y").getString("Error"),
                    JOptionPane.ERROR_MESSAGE);
            return false;
        }
        finally {
            if(io != null) io.close();
        }

        return true;
    }


    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.ButtonGroup buttonGroup1;
    private br.ufsc.inf.lagar.ui.panels.FileChooserLLG fileChooserLLG1;
    private javax.swing.JButton infoButton;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JScrollPane jScrollPane2;
    private javax.swing.JLabel labelSepLeft;
    private javax.swing.JLabel labelSepRight;
    private javax.swing.JTextArea resultArea;
    private javax.swing.JButton runButton;
    private br.inf.ufsc.lagar.swingutil.SimpleInternalFrame simpleInternalFrame1;
    private javax.swing.JPanel versPanel;
    // End of variables declaration//GEN-END:variables

}
/*
 * WizardPanel.java
 *
 * Copyright (C) 2006  Euclides Pinheiro de Melo  <lagar_lottery@yahoo.com.br>
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU Library General Public
```

```java
 * License as published by the Free Software Foundation; either
 * version 2 of the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
 * Library General Public License for more details.
 *
 * You should have received a copy of the GNU Library General Public
 * License along with this program; if not, write to the
 * Free Software Foundation, Inc., 59 Temple Place - Suite 330,
 * Boston, MA 02111-1307, USA.
 *
 */

package br.ufsc.inf.lagar.ui.wizard;

import java.awt.CardLayout;
import java.awt.Component;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Stack;


public class WizardPanel extends javax.swing.JPanel {

    private final String NEXT = "Next >";
    private final String FINISH = " Finish";

    private Stack<String> pagePath = new Stack<String>();
    private Map<String, IWizardPage> pageMap = new HashMap<String, IWizardPage>();
    private List<String> pages = new ArrayList<String>();

    /** Creates new form WizardPkcs11Panel */
    public WizardPanel() {
        initComponents();
        posInit();
    }

    private void posInit() {

        this.getBackButton().setEnabled(false);
        this.getNextButton().setText(NEXT);
    }

    public void addPage(IWizardPage panel){

        String pageName = panel.getPageName();

        if(pageName == null || pageName.equals("")){
            throw new IllegalArgumentException("Page name cannot be empty or null");
        }
        if(this.pageMap.containsKey(pageName)){
            throw new IllegalArgumentException("Already have one page with name:
"+pageName);
        }

        this.pageMap.put(pageName, panel);
        this.pages.add(pageName);

        if(this.pagePath.isEmpty()){
            this.pagePath.push(pageName);
        }

        panel.setWizardPanel(this);

        this.mainPanel.add((Component) panel, pageName);
    }

    protected void showPage(String pageName){

        if(this.pagePath.size() <=1) this.getBackButton().setEnabled(false);
        else this.getBackButton().setEnabled(true);

        CardLayout cards = (CardLayout) this.mainPanel.getLayout();
        cards.show(this.mainPanel, pageName);
    }

    protected void goNextPage(){

        this.deactivateActualPage();

        String current = this.pagePath.peek();
        IWizardPage panel = this.pageMap.get(current);

        String nextPage = panel.getNextPageName();

        if(nextPage == null){
            nextPage = this.getNextPage(current);
        }

        this.pagePath.push(nextPage);
        IWizardPage currentPanel = this.pageMap.get(nextPage);
        currentPanel.activatePage();

        if(this.isLastPage(nextPage)){
            this.nextButton.setText(FINISH);
        }

        this.showPage(nextPage);
    }

    protected void goPreviousPage(){

        this.nextButton.setText(NEXT);

        this.deactivateActualPage();
        this.pagePath.pop();

        String page = this.pagePath.peek();
        IWizardPage panelAtual = this.pageMap.get(page);
        panelAtual.activatePage();

        this.showPage(page);
    }

    protected void deactivateActualPage(){
        if(!this.pagePath.isEmpty()){
            IWizardPage panel = this.pageMap.get(this.pagePath.peek());
            panel.deactivatePage();
        }
    }

    protected boolean isLastPage(String currentPage){
        return currentPage.equals(this.getNextPage(currentPage));
    }

    protected String getNextPage(String currentPage){

        int index = this.pages.indexOf(currentPage);

        if(index == -1) return null;
        if(index == this.pages.size()-1) return currentPage;

        return this.pages.get(++index);
    }


    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    // <editor-fold defaultstate="collapsed" desc=" Generated Code ">//GEN-
BEGIN:initComponents
    private void initComponents() {
        buttonsPanel = new javax.swing.JPanel();
        cancelButton = new javax.swing.JButton();
        nextButton = new javax.swing.JButton();
        backButton = new javax.swing.JButton();
        jSeparator1 = new javax.swing.JSeparator();
        titlePanel = new br.inf.ufsc.lagar.swingutil.SimpleInternalFrame();
        mainPanel = new javax.swing.JPanel();

        cancelButton.setText("Cancel");
        cancelButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                cancelButtonActionPerformed(evt);
            }
        });

        nextButton.setText("Next >");
        nextButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                nextButtonActionPerformed(evt);
            }
        });

        backButton.setText("< Back");
        backButton.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                backButtonActionPerformed(evt);
            }
        });

        org.jdesktop.layout.GroupLayout buttonsPanelLayout = new
org.jdesktop.layout.GroupLayout(buttonsPanel);
        buttonsPanel.setLayout(buttonsPanelLayout);
        buttonsPanelLayout.setHorizontalGroup(
            buttonsPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
buttonsPanelLayout.createSequentialGroup()
                .addContainerGap(227, Short.MAX_VALUE)
                .add(backButton, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 97,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
```

```
                    .add(nextButton)
                    .add(17, 17, 17)
                    .add(cancelButton)
                    .addContainerGap())
                .add(jSeparator1, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 531,
Short.MAX_VALUE)
        );

        buttonsPanelLayout.linkSize(new java.awt.Component[] {backButton,
nextButton}, org.jdesktop.layout.GroupLayout.HORIZONTAL);

        buttonsPanelLayout.setVerticalGroup(
            buttonsPanelLayout.createParallelGroup(org.jdesktop.layout.GroupLayout.LE
ADING)
                .add(org.jdesktop.layout.GroupLayout.TRAILING,
buttonsPanelLayout.createSequentialGroup()
                    .add(jSeparator1, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE, 10,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                    .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .add(buttonsPanelLayout.createParallelGroup(org.jdesktop.layout.Group
Layout.BASELINE)
                        .add(cancelButton)
                        .add(nextButton)
                        .add(backButton))
                    .addContainerGap())
        );

        buttonsPanelLayout.linkSize(new java.awt.Component[] {backButton,
nextButton}, org.jdesktop.layout.GroupLayout.VERTICAL);

        titlePanel.setTitle("Wizard");

        mainPanel.setLayout(new java.awt.CardLayout());

        org.jdesktop.layout.GroupLayout layout = new
org.jdesktop.layout.GroupLayout(this);
        this.setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(buttonsPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .add(titlePanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 531,
Short.MAX_VALUE)
            .add(layout.createSequentialGroup()
                .addContainerGap()
                .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 507,
Short.MAX_VALUE)
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(org.jdesktop.layout.GroupLayout.LEADING)
            .add(org.jdesktop.layout.GroupLayout.TRAILING,
layout.createSequentialGroup()
                .add(titlePanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(mainPanel, org.jdesktop.layout.GroupLayout.DEFAULT_SIZE, 289,
Short.MAX_VALUE)
                .addPreferredGap(org.jdesktop.layout.LayoutStyle.RELATED)
                .add(buttonsPanel, org.jdesktop.layout.GroupLayout.PREFERRED_SIZE,
org.jdesktop.layout.GroupLayout.DEFAULT_SIZE,
org.jdesktop.layout.GroupLayout.PREFERRED_SIZE))
```

```
        );
    }// </editor-fold>//GEN-END:initComponents

    private void cancelButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_cancelButtonActionPerformed
        this.cancelWizard();
    }//GEN-LAST:event_cancelButtonActionPerformed

    private void nextButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_nextButtonActionPerformed
        if(this.nextButton.getText().equals(FINISH)){
            this.endWizard();
        } else{
            this.goNextPage();
        }
    }//GEN-LAST:event_nextButtonActionPerformed

    private void backButtonActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_backButtonActionPerformed
        this.goPreviousPage();
    }//GEN-LAST:event_backButtonActionPerformed


    private void endWizard() {
        // TODO Auto-generated method stub

    }

    private void cancelWizard() {
        // TODO Auto-generated method stub

    }

    public javax.swing.JButton getBackButton() {
        return backButton;
    }

    public void setBackButton(javax.swing.JButton backButton) {
        this.backButton = backButton;
    }

    public javax.swing.JButton getCancelButton() {
        return cancelButton;
    }

    public void setCancelButton(javax.swing.JButton cancelButton) {
        this.cancelButton = cancelButton;
    }

    public javax.swing.JButton getNextButton() {
        return nextButton;
    }

    public void setNextButton(javax.swing.JButton nextButton) {
        this.nextButton = nextButton;
    }




    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton backButton;
    private javax.swing.JPanel buttonsPanel;
    private javax.swing.JButton cancelButton;
    private javax.swing.JSeparator jSeparator1;
    private javax.swing.JPanel mainPanel;
    private javax.swing.JButton nextButton;
    private br.inf.ufsc.lagar.swingutil.SimpleInternalFrame titlePanel;
    // End of variables declaration//GEN-END:variables



}
```

11 Apêndice B - Artigo

Desenvolvimento de um software para jogadores de loterias

Euclides Pinheiro de Melo

Departamento de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88040-900 – Santa Catarina – SC – Brasil

lagar@inf.ufsc.br

Abstract. *Winning in a lottery has always been the dream of many people since the early days of lottery drawings, which started to take place many centuries ago. At the same time, people have always been seeking ways to improve their chances of becoming a winner, usually trying to discover patterns that might be present in the past results, and putting their bets on the possible results that seemed to them the most probable to occur. However, both the statistical analysis of previous data and the choice of a combination of bets satisfying some established criteria often implies in hard and repetitive mechanical work, which turns out to be even worse when it becomes necessary to fill a large number of bet cards manually. With this scenario in mind, the objective of this work is to supply a software that can help a wagerer to formulate and lay his bets without wasting a long time with dull tasks, as mentioned.*

Resumo. *Vencer na loteria tem sido o sonho de muitas pessoas ao longo da história da humanidade. As pessoas vêm constantemente criando e inventando meios para fazer o sonho se tornar o mais próximo e real, e em geral, estes meios exigem "trabalho braçal", como a realização de longas análises sobre resultados passados, o desdobramento de apostas ou o preenchimento manual de um grande número de cartões. O objetivo deste trabalho consiste em fornecer um software que auxilie o jogador a ter um meio de pôr em prática suas idéias, sem precisar dispender um longo período de tempo com tarefas manuais, deixando o apostador concentrar-se na parte que demanda mais inteligência.*

1. O problema da loteria

O problema das loterias consiste em descobrir que dezenas serão sorteadas no próximo concurso, ou que aposta, ou conjunto de apostas, terá acertado o maior número de dezenas possível no próximo concurso.

Um fato aceito é que não é possível prever exatamente o próximo resultado, uma vez que uma loteria é teoricamente um sistema aleatório. Mas isso não impede a mente humana de imaginar meios de chegar perto, tentando analisar as variáveis que estão presentes e imaginar planos, cenários possíveis para tentar tomar uma decisão melhor. Cada apostador tem sua grande idéia única de como resolver este problema. Por exemplo, há os que acreditam que o melhor meio é utilizar números presentes no seu dia a dia, como idade dos filhos, número de documentos, entre outros. Há quem acredite que sonhar com os números é um bom meio. Há quem acredite que produzir um conjunto de cartões com determinadas características é o melhor meio. Independente da validade destes meios, o que o usuário precisa, em muitos casos, é de um suporte ferramental para pôr em prática a sua "grande idéia única".

É neste cenário que entra um software para jogadores de loterias. Um software para jogadores de loterias não é uma caixa mágica, onde aperta-se um botão e consegue-se as dezenas que irão sair no próximo concurso. Tal recurso, apesar de desejável, não é possível de ser concebido, devido a natureza do problema. O que um bom software para loterias pode fazer, para auxiliar o jogador, é reunir as informações disponíveis e permitir que o usuário possa visualizar elas de maneira agradável (utilizando-se de interface gráfica, dados sob a forma de gráficos), auxiliando a sua tomada de decisão para o próximo concurso. Outra característica é de automatizar tarefas repetitivas, que podem ser feitas rapidamente utilizando-se um computador, que se fossem feitas pelo homem, seriam tediosas e passíveis de erros.

2. Software para jogadores de loteria

Um software para jogadores de loterias tem como propósito principal aliviar o usuário de tarefas que não demandam inteligência, ou seja, tarefas mecânicas e repetitivas; e não apenas aliviar o usuário, como também dar agilidade e conveniência ao processo de se jogar na loteria.

O processo de jogar na loteria, consiste, basicamente, nos seguintes passos representados na figura abaixo:

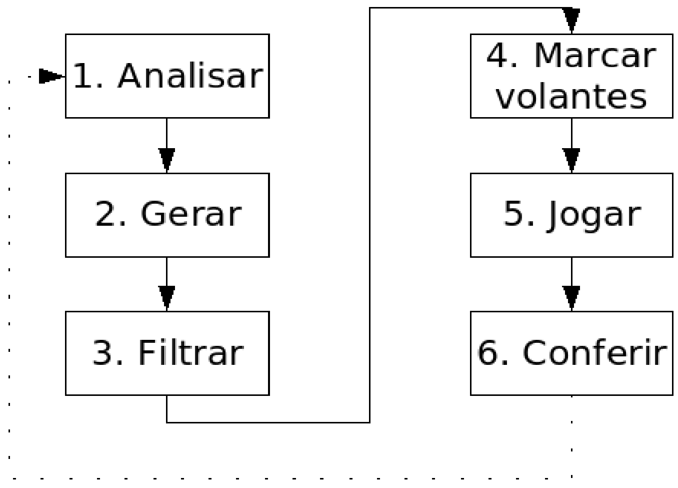


Figura 1: processo para jogar na loteria

1. Analisar: consiste do passo do usuário pensar, pesquisar, coletar dados para realizar alguma análise em resultados passados ou outra fonte que julgue adequada para escolher dezenas a jogar ou gerar um bolão. Este passo é opcional.
2. Gerar: consiste em o usuário inventar um meio de combinar suas dezenas, gerando tipicamente um bolão. Este passo é opcional.
3. Filtrar: restringir, usando alguma regra, os jogos gerados, a fim de reduzir os custos do jogo, ou fazer caber a quantidade de cartões na verba disponível. Este passo é opcional.
4. Marcar volantes: consiste no preenchimento físico dos volantes de loteria, a fim de se poder realizar a aposta em uma casa lotérica credenciada. Passo requerido.
5. Jogar: consiste no ato de submeter os volantes ao agente lotérico e realizar o pagamento. Passo requerido.
6. Conferir: averiguar se houve sucesso ou fracasso no jogo que apostou. Passo requerido.

Observando-se cada passo descrito, nota-se que existe uma série de atividades manuais que o usuário precisaria executar se não usar um software para jogador de loterias. Vamos a algumas:

- No caso da análise, em particular de resultados passados, teria que manualmente colher todos os resultados de loteria, e aplicar meios de visualizar estes dados de forma mais agradável. Ex: visualizar a distribuição de pares e ímpares nos resultados passados: colher cada resultado, contar em cada resultado quantos pares e ímpares deram, montar uma tabela ou gráfico exibindo o resultado da análise. Manualmente, é mais de um dia de trabalho;

- No caso de gerar, em particular em bolões, onde se costumam jogar, por exemplo, centenas ou milhares de volantes. Enumerar todos os números de cada volante é um trabalho extremamente repetitivo e demorado;
- No caso de filtrar, aplicar regras de rejeite ou aceite em cada volante, novamente é um trabalho demorado;
- No caso de marcar volantes, pintar cada célula de cada volante de loteria usando caneta é algo que pode de dias a semanas, dependendo do tamanho do jogo;
- No caso de conferir, examinar cada cartão para determinar a quantidade de erros e acertos, no caso de haver muitos volantes, demanda muito tempo e esforço;

Em suma, o propósito básico de um software para jogadores de loterias consiste em automatizar as tarefas descritas acima (podendo ainda oferecer outras facilidades mais) e deixar com o jogador a estratégia que ele julgue adequada para vencer. Deste modo, um software para jogadores de loteria é visto como uma ferramenta, algo que promove a eficiência, minimizar esforço para aquele que tem de executar alguma tarefa. Uma analogia: imagine que você precisa pregar um prego na parede. Você tem o prego, você tem a parede. É possível ser feito só com as mãos, mas o esforço e tempo serão muito grandes. A tarefa se torna mais fácil quando você tem uma ferramenta adequada para o problema: no caso, um martelo. Encare o software para jogadores de loteria como o martelo: ter a ferramenta não garante que o prego estará pregado, mas facilita a tarefa. Depende do utilizador da ferramenta para alcançar o objetivo.

2.1 Limitações

É comum, quando as pessoas ouvem sobre softwares de loteria, pensar que "inventou-se" um jeito garantido de ganhar, ou então pensar que "o software vai ganhar para mim". O software é uma ferramenta que você usa para facilitar a vida ao jogar na loteria, e depende da estratégia, de entrada do usuário para fornecer saída.

Ao acessar *sites* de produtos comerciais de loteria, é comum anunciarem que o software vai te fazer ganhar, mas esteja ciente que todo software deste tipo sofre desta limitação, que é a impossibilidade de garantir ganhos (claro, é possível garantir ganho se especificar alguns cenários, mas pense de modo geral).

Outra limitação está no fato de que o software nunca vai satisfazer todos os desejos de todos os jogadores: cada jogador tem sua estratégia única e particular que acha adequado para vencer na loteria. Os softwares disponíveis reúnem algumas estratégias comuns que boa parte dos jogadores gostam / costumam jogar e disponibilizam no software. A falta de extensibilidade por parte dos softwares disponíveis tentará ser resolvida neste trabalho. Softwares existentes podem ser encontrados nas referências [SPLOTO01], [PAPASORTE01] e [GERASORTE01].

3. Projeto do software

O software foi projetado para suportar o processo para jogar na loteria. Utilizou-se os passos macro do processo para dividir o software em módulos, essencialmente um para cada passo. São eles:

- Módulo núcleo – Classes e abstrações comuns usadas por todos os módulos;
- Módulo análise – Classes e abstrações relacionadas com requisitos de análise;
- Módulo filtro – Classes e abstrações relacionadas com os requisitos de filtro;
- Módulo gerador – Classes e abstrações relacionadas com os requisitos de geração;
- Módulo conferência - Classes e abstrações relacionadas com os requisitos de conferência;
- Módulo impressão – Classes e abstrações relacionadas com os requisitos de impressão.

A representação gráfica pode ser encontrada na figura abaixo:

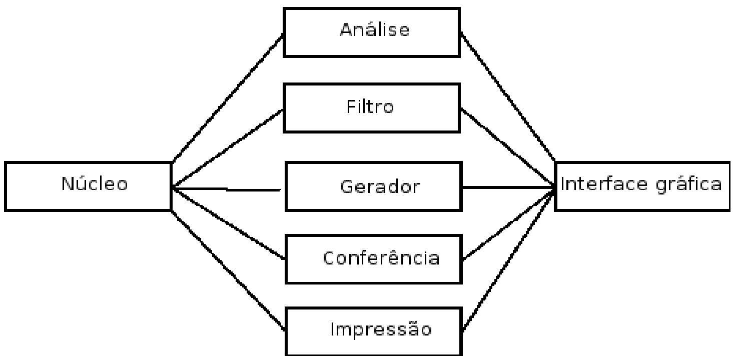


Figura 2: Representação dos módulos

Este foi o primeiro esforço de divisão do problema, o de tentar quebrar o projeto de software em unidades menores de software, para facilitar a codificação, validação dos requisitos e testes.

3.1 Casos de uso

Para facilitar a visualização das funcionalidades do sistema, segue abaixo o diagrama de casos de uso:

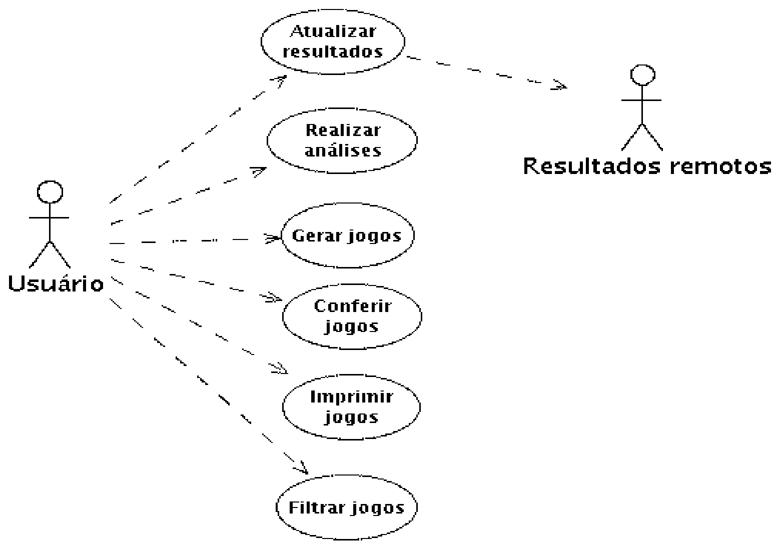


Figura 3: Diagrama de casos de uso

Novamente, os casos de uso são tarefas que estão presentes no processo para se jogar na loteria.

3.2 Tornando o software extensível

O sistema usa, em vários módulos, a noção de uma arquitetura plugin, no sentido que código novo introduzido não requer mudança em código existente para que o código existente perceba e possa fazer uso das novas funcionalidades adicionadas. Alguns exemplos clássicos, para facilitar o entendimento:

Navegadores de internet suportam plugins para adicionar novos tratadores de tipo de mídia: por exemplo, o usuário instala o plugin do Macromedia Flash para suportar este tipo de mídia no navegador, e o código do navegador não é alterado ou recompilado;

Editores de imagem, como o Photoshop, Gimp, Corel suportam plugins para aplicar efeitos nas imagens em edição;

Ambientes de desenvolvimento, como o Eclipse e o Netbeans, permitem que novas funcionalidades sejam agregadas ao ambiente. O Eclipse levou o conceito tão longe que o próprio software é apenas um grande gerenciador de plugins, não fornecendo ele próprio funcionalidade de uso.

No caso deste software de loterias, foi utilizado este conceito para que outros programadores possam adicionar novas análises, filtros, tipos de jogos, verificadores e geradores sem precisar tocar em uma única linha de código sequer de todo o sistema. Isso facilita muito a adição destas novas funcionalidades, pois quem assim o quiser fazer, apenas precisa escrever seu novo código de interesse, sem se preocupar em como tornar isso disponível para a estrutura existente. A estrutura existente é capaz de detectar que algo novo está presente e tratar de encaixá-la no contexto. Deste modo, o requisito de extensibilidade facilitada fica garantido, que é um dos maiores diferenciais deste software em comparação com outros softwares existentes.

4. Tecnologias utilizadas

Como este trabalho é de código aberto, a escolha das tecnologias tem de levar este aspecto em conta, uma vez que se há o desejo da sociedade em poder acessar e editar este código, há a preferência que este o seja feito usando tecnologias livres, ou no mínimo gratuitas, para que, quem for manusear o fonte, possa instalar as mesmas ferramentas que o desenvolvedor principal utilizou. Seria ruim, imaginando que eu fosse um desenvolvedor novo, querendo conhecer e editar o código do projeto, tivesse que comprar uma ferramenta cara para poder abrir a estrutura de projeto de código e editar os fontes.

Utilizou-se das seguintes tecnologias:

- Linguagem Java versão 5 [SUN01].
- Netbeans 5.5 [NETBEANS01].
- Hospedagem em SourceForge [SOURCEFORGE01].

5. LagarLottery 1.0

LagarLottery foi o nome dado ao software. Por se tratar de um código que está disponível em um repositório americano (SourceForge) e estar disponível para o mundo, escolheu-se um nome em inglês para o produto (e é por este mesmo motivo que o código fonte está em inglês).

O fonte, binário e documento completos podem ser obtidos no site do projeto:

http://sourceforge.net/projects/lagar-lottery

6. Considerações finais / trabalhos futuros

Este artigo demonstra que um software para jogadores de loteria é uma ferramenta útil, e que uma ferramenta desta natureza pode ser muito interessante, até para quem não costuma jogar muito.

Um grande desafio deste trabalho foi tentar contornar o problema da extensibilidade: cada jogador tem sua idéia única e particular de gerar um jogo, realizar uma análise, filtrar cartões, em fim, cada jogador de loteria tem a sua grande idéia, e não é possível prever isso de antemão. A solução adotada neste trabalho foi deixar o software com vários pontos de extensibilidade (um em cada módulo) para que o jogador pudesse ele mesmo (se tiver conhecimento de programação) ou pedir para que outro o faça (o código é livre) para adicionar a sua tão particular funcionalidade. Se o software não tivesse sido projetado para ter os pontos de extensibilidade e usar a "arquitetura plugin", essa tarefa de adicionar código novo seria muito mais demorada e penosa. Mas com essa solução, uma nova análise ou filtro pode ser adicionado em questão de minutos.

Neste ponto temos uma possibilidade de trabalho futuro (versão 2.0) do software: permitir a inserção de código novo (análise, filtro, etc) através de uma linguagem de script, no estilo que ferramentas de escritório como o Microsoft Excel permite adicionar através de macros: o usuário poder, através de uma linguagem simples, especificar uma nova funcionalidade e executar esta nova funcionalidade no próprio ambiente.

Outra possibilidade de trabalho futuro está em capturar regras de análise do usuário, para que o software possa, através destas regras fornecidas, processar os dados da análise automaticamente. Por exemplo: imagine que você sempre olha a característica X da análise de pares x ímpares, procurando sempre por um padrão Y. Se você pudesse dizer isso para o software, este poderia colorir a tabela resultante com cores diferentes indicando um padrão fornecido.

Por fim, deixo o convite a realizar o download do projeto em http://sourceforge.net/projects/lagar-lottery/ , experimentar o software e contribuir com suas idéias e código para que este seja um dos melhores entre os softwares para jogadores de loterias existentes.

Referências

[SUN01] Linguagem java. Disponível em: http://java.sun.com. Acesso em: abril de 2006.

[NETBEANS01] Netbeans IDE. Disponível em: http://www.netbeans.org. Acesso em: abril de 2006.

[SOURCEFORGE01] SourceForge. Disponível em: http://sourceforge.net/. Acesso em: abril de 2006.

[SPLOTO01] SpLoto. Disponível em: http://www.spolti.com.br/sploto/. Acesso em: maio de 2006.

[GERASORTE01] Gerasorte. Disponível em: http://www.gerasorte.com.br/. Acesso em: maio de 2006.

[PAPASORTE01] Papasorte. Disponível em: http://www.loteria.com.br/. Acesso em: maio de 2006.