UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE CIÊNCIAS DA COMPUTAÇÃO

Beta Browser

Browser Estruturado em Tecnologias Assistivas

Victor Hugo Marques Costa

Florianópolis
Julho 2004

UNIVERSIDADE FEDERAL DE SANTA CATARINA
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CURSO DE CIÊNCIAS DA COMPUTAÇÃO

Beta Browser

Browser Estruturado em Tecnologias Assistivas

Victor Hugo Marques Costa

Trabalho apresentado à disciplina de Projetos II, orientado pelo professor João Bosco da Motta Alves.

Florianópolis
Julho 2004

“A suprema felicidade da vida é a convicção de que somos amados, acima de tudo, por nós próprios – ou melhor, apesar de nós próprios”

Victor Hugo

À minha querida esposa, Roberta, que me apoiou durante todo o projeto e faculdade, e aos meus pais Nelson e Bernadete que me guiaram durante toda a vida.

Agradeço a minha família que me acompanhou com paciência e carinho. Aos amigos que ajudaram na hora do aperto e aos meus orientadores João Bosco da Motta Alves, Andréa Miranda e Carlos Eduardo Gonçalves.

SUMÁRIO

LISTA DE FIGURAS

RESUMO

As pessoas que apresentam algum tipo de limitação se vêem impedidas, por vários motivos, de ter acesso às Tecnologias de Informação e Comunicação. Entretanto, a revolução informacional é uma realidade e um processo irreversível na história da sociedade pós-moderna. Portanto, o desenvolvimento de tecnologias alternativas que tenham a acessibilidade como conceito primordial, é fundamental para que as desigualdades sociais e o número de excluídos digitais não aumentem ainda mais. Diante desta realidade, este projeto apresenta uma proposta de estudo e desenvolvimento de um browser utilizando a ferramenta Delphi. Esse contará com sistema de varredura de teclas e com um teclado virtual para a digitação dos endereços, afim de que os usuários com limitação física e/ou motora possam navegar na Web, com maior facilidade e menor desgaste físico e emocional. Para tanto, será feita a avaliação da usabilidade da ferramenta proposta, objetivando adaptá-la ao máximo às necessidades deste grupo de usuários com tais limitações.

1 INTRODUÇÃO

O mundo em que vivemos traz aos deficientes físicos grandes dificuldades ao acesso à informação. Estas dificuldades se evidenciam, e na era da informação chegam através das tecnologias e páginas da Web pouco acessíveis. O principal equívoco cometido pelos fabricantes de dispositivos e softwares é a definição de usuário padrão, cujo perfil pode ser descrito como o de um indivíduo do sexo masculino com 25 anos de idade, anglo-saxão, formado em ciências da computação e viciado em tecnologia, NEWEL (4).

Uma das abordagens adotadas para minimizar este problema é a das tecnologias assistivas, que são quaisquer dispositivos capazes de facilitar a execução de uma tarefa. Essas tecnologias podem ser dividas em 5 grupos distintos:

- Tecnologias alternativa/aumentativas de acesso à informação;
- Tecnologias de acesso;
- Tecnologias alternativo/aumentativas para a comunicação;
- Tecnologias de mobilidade;
- Tecnologias de controle ambiente.

Este trabalho tem como principal objetivo desenvolver um browser capaz de facilitar o acesso às informações contidas na internet, por pessoas com deficiência física e/ou motora. Assim, esta ferramenta terá características do primeiro e do terceiro grupo de tecnologias assistivas acima enumeradas, pois além de contribuir para o aumento do acesso à informação, aumentará também a comunicação deste grupo de usuários através de webmail's e outras ferramentas disponíveis em páginas Web.

Esta ferramenta será direcionada, a princípio, a pessoas com deficiência física e/ou motora, sendo que suas limitações não impossibilitem a leitura e compreensão da interface, pois, caso contrário, a ferramenta perderia sua utilidade.

1.1 Justificativa

De acordo com o mapa de exclusão digital divulgado pela fundação Getúlio Vargas – FGV e pelo comitê para democratização da Informática – CDI, o uso da Internet aumentou de 10% para 15% e, segundo dados do Censo/2000, cerca 24,5 milhões de pessoas apresentam algum tipo de incapacidade ou deficiência. Entretanto, observa-se que nas Instituições de Ensino Superior as iniciativas que minimizem este cenário ainda são muito insipientes. Paralelamente a isto, há pouco interesse por parte não só das Universidades, como também das empresas, em desenvolver tecnologias acessíveis que possibilitam a inclusão digital e, conseqüentemente, o exercício da cidadania de forma efetiva por parte dos Portadores de Necessidades Especiais.

Diante deste cenário, a motivação que levou a desenvolver um trabalho nesta área, está ligada à necessidade trazida pelos portadores de deficiência física na obtenção de informações disponíveis na internet, e na dificuldade por parte deste grupo de usuários em acessar as tecnologias ligadas à informática. Configura-se, assim, relevância deste estudo.

1.2 Objetivos

1.2.1 Geral

Desenvolver um browser baseado no conceito de acessibilidade, afim de que as informações contidas na internet possam ser acessadas também por Portadores de Necessidades Especiais.

1.2.2 Específicos

- Facilitar o manuseio da interface do browser, através da implementação da varredura das teclas de navegação;
- Implementar um teclado virtual que possibilite ao usuário, deficiente físico/ motora, digitar os endereços web.

2 REVISÃO BIBLIOGRÁFICA

2.1 Fundamentação Teórica

2.1.1 Classificação de Pessoas Portadoras de Necessidades Especiais

A Classificação dada em 1980 pela OMS (O*rganização Mundial da Saúde*), que faz parte do documento conhecido como ICIDH (*International Classification of Impairments, Disabilities, and Handicap*), apresentadas aqui com base em MONTOYA (3), pode ser vista abaixo:

- Deficiência (*Impairment*): perda ou anormalidade de uma estrutura ou função psicológica, fisiológica ou anatômica;
- Incapacidade (*Disability*): restrição ou ausência da capacidade de realizar uma atividade na forma ou dentro da margem que se considera normal para o ser humano;
- Desvantagem (*Handicap*): é a situação desvantajosa em que se encontra um indivíduo, em conseqüência de uma deficiência ou de uma incapacidade, que lhe limita e impede de desempenhar um rol de atividades que seria considerado normal para pessoas da mesma idade, sexo e nível sociocultural.

2.1.2 Classificação das Tecnologias Assistivas

As tecnologias assistivas também conhecidas como ajudas técnicas, podem ser definidas em várias áreas da vida e da ciência, que vão desde uma cadeira de rodas até softwares de reconhecimento de voz.

Define-se formalmente tecnologia assistiva como "*dispositivos que correspondem a qualquer item, peça de equipamento ou produto, seja ele adquirido comercialmente ou não, modificado ou construído, que é utilizado para aumentar, manter ou aperfeiçoar capacidades funcionais de indivíduos com deficiências físicas"* DATI (1).

Segundo KOON(2), dentro da área de computação, pode-se dividir as tecnologias assistivas em 5 grupos distintos:

1- Tecnologias alternativo/aumentativo de acesso a informação

Este tipo de tecnologia encaixa-se melhor às necessidades dos usuários com limitações sensorias, pois cria ferramentas que ampliam as possibilidades de acesso à informação. São exemplos destas tecnologias os programas de reconhecimento de voz, multimídia, aplicações de computação móvel e conectividade.

2- Tecnologias de acesso

As pessoas com deficiência física ficam impossibilitadas ou com dificuldades ao tentarem manusear um computador. Por este motivo, surgem estes tipos de tecnologias que são representadas pelos sistemas magnificadores de tela, simuladores de teclado, linha Braille e sintetizadores de voz.

3- Tecnologias alternativo/aumentativas para a comunicação

Toda pessoa que não consegue comunicar-se através do código verbal – oral – beneficia-se destas tecnologias. A única diferença entre as duas é que a primeira não utiliza a linguagem falada, mas sim outras linguagens, como a dos sinais. Já a segunda, beneficia-se de alguns recursos para facilitar a comunicação verbal.

4- Tecnologias de mobilidade

Alguns ambientes contam com facilitadores para pessoas com deficiência fisíca, criando instrumentos que facilitem sua movimentação. Este tipo de tecnologia envolve recursos avançados da ciência, com características de ficção, como implantes cibernéticos.

5- Tecnologias de controle ambiente

Todas as tecnologias que permitem o controle de um ambiente, como o abrir e fechar de portas e o acionamento de eletrodomésticos. As aplicações em realidade virtual são destaque nesta área.

2.2 Interface com o usuário

Observa-se que grande parte das empresas de desenvolvimento de software não leva em conta o tipo de usuário que estará manipulando as interfaces de tais produtos.

Segundo CYBIS (5) *"A dificuldade no desenvolvimento de interfaces com usabilidade se deve ao fato delas constituírem fundamentalmente, sistemas abertos, probabilísticos, não determinísticos, sujeitos as influências do ambiente e as interpretações dos usuários."*. Um dos primeiros problemas a serem resolvidos pelos desenvolvedores de interfaces é a questão da interpretação da face do programa. Para algumas empresas que não se preocupam com este problema, toda a interface baseia-se na interpretação do programador, e não é realizado nenhum tipo de estudo de caso, onde essa seria apresentada a usuários comuns e verificadas as facilidades de uso.

Para o desenvolvimento de interfaces amigáveis e ergonômicas, o desenvolvedor deve conhecer profundamente seu usuário final, ou seja, prever seus movimentos, suas interpretações e evitar ao máximo influências do ambiente, que no caso dos programas, são as mensagens de erro e paradas críticas.

2.2.1 Características ergonômicas da Interface Humano-Computador (IHC)

Bastien e Scapin *apud* CIBYS (5), elucidam que a interface humano-computador, segundo a engenharia de usabilidade, deve apresentar algumas características básicas de ergonomia, além das potencialidades do software. São elas:

- A condução: diz respeito à capacidade da interface de guiar e orientar o usuário através de mensagens, rótulos, alarmes e etc., possibilitando a localização pelo usuário (capacidade de localizar-se dentro do programa e saber exatamente em que ponto está a execução de sua tarefa), o conhecimento das conseqüências de suas ações (descriminação explícita e clara das funcionalidades da interface) e, por fim, a obtenção de informações suplementares.
    - Presteza: caracteriza-se pela qualidade que a interface possui de orientar o usuário nas suas ações, ou seja, o usuário deve saber exatamente onde está e quais ações pode tomar, bem como suas conseqüências.
    - Feedback imediato: o sistema deve ter respostas rápidas e apresentar informações sobre as transações efetuadas, assim como seus resultados. Uma interface com um bom feedback imediato aumenta o nível de confiança do usuário que, caso contrário, pode realizar ações impróprias por julgar falha do sistema, e isto pode prejudicar o andamento da tarefa.
    - Legibilidade: a interface do sistema deve levar em conta a facilidade que a informação é transmitida. Por exemplo, letras escuras em fundo claro trazem uma maior facilidade na leitura do que letras claras em fundo escuro, assim como a leitura de palavras com letras maiúsculas e minúsculas é mais rápida do que com todas as letras maiúsculas.
    - Agrupamento/distinção de itens: é a organização dos objetos dentro da tela que podem ser distribuídos de duas maneiras: por localização ou por formato.
- Carga de trabalho: uma interface de sistema deve oferecer ao usuário um número de ações mínimas para executar uma tarefa. Quanto maior o número de ações maior será a carga de trabalho.
    - Brevidade: pela pequena capacidade da memória de curto prazo, quanto menos entradas menor a probabilidade de cometer erros. Além disso, quanto mais sucintos forem os itens, menor será o tempo de leitura e quanto mais complexas forem as ações para

realizar-se uma tarefa maior será a carga de trabalho e maior será a chance de se cometer erros.

    - Densidade Informacional: a carga de memorização deve ser minimizada, pois se o usuário tem de decorar listas de dados ou procedimentos complicados que não dizem respeito à tarefa em si, ele terá desviado a sua atenção e provavelmente ocasionará erros.

- Controle explícito: o usuário deve ter o controle total do sistema, quando suas entradas estão sobcontrole, os erros e ambigüidades reduzem.
    - Ações explícitas do usuário: o programa deve estar na "mão" do usuário, ou seja, tudo aquilo que ele solicitar deve ser atendido e processado imediatamente.
    - Controle do usuário: refere-se ao fato de que apenas os usuários deveriam estar no controle do sistema (ex. cancelar, confirmar, etc).
- Adaptabilidade: a interface do sistema deverá possuir a capacidade de se adaptar às necessidades do usuário, moldando-se as suas exigências.
    - Flexibilidade: o usuário do sistema deve contar com esta característica para manusear a interface e configurá-la de forma a facilitar a sua tarefa.
    - Consideração da experiência do usuário: em alguns casos o sistema deve detectar as tarefas que são comuns e muito executadas pelo usuário e criar atalhos ou diminuir o número de ações para sua realização. Isto decorre do uso do sistema e da experiência que este usuário adquiriu.
- Gestão de erros: o software deve apresentar soluções ao usuário, que muitas vezes devem ser transparentes a este. Um erro de sistema pode ser desde a entrada incorreta de dados, onde esse deve sugerir a formatação e o tipo correto destes, até mensagens de erros críticos, onde o sistema deve possibilitar o contorno desses.
    - Proteção contra erros: trata e previne erros do usuário e do próprio sistema para que não ocorram danos irreparáveis.

    - Qualidade das mensagens de erro: o usuário, a partir de uma mensagem clara e completa, deve poder contornar o erro sem danos irreversíveis, sendo que o sistema ainda deve indicar a maneira de corrigí-lo.
    - Correção de erros: é a capacidade do sistema de reconhecer o tipo de erro e guiar o usuário na sua correção. Quando os passos para corrigir os erros são simples e fáceis causam menos "traumas" nesses usuários.
- A homogeneidade/coerência: em tarefas ou contextos distintos, as identificações de campos e nomes de comandos devem ser diferentes para não gerar uma associação incorreta entre as tarefas. Em tarefas e contextos iguais, os termos e nomeações destes campos devem manter-se.

2.2.2 Os componentes da Interação Humano-Computador

Os componentes da IHC são aqueles que orientam e ensinam a utilização do sistema ao seu usuário. Estes componentes podem-se dividir em:

- Diálogos;
    - As ações;
        - Ação de entrada de dados ou comandos (ex.: sistema avalia e corrige a entrada de um campo);
    - As tarefas;
        - Tarefa de diagnóstico (ex.: em casos de campos de dados críticos o sistema deve fornecer ajuda detalhada do formato dos dados) ;
        - Tarefa corretiva (ex.: sistemas que apresentam mensagens claras e que possam ser configuradas pelo usuário);
        - Tarefa destrutiva (ex.: em diálogos de confirmação de deleção, a tecla default deve ser a não destrutiva, para que não ocorra perda de dados);

- Os estilos dos diálogos;
    - Diálogo por menu (ex.: Menus principais do Windows XP que filtram os comandos mais utilizados);
    - Diálogo por linguagem de comandos (ex.: Diálogos no estilo terminal, onde os comandos devem ser rigidamente corretos, exigindo experiência do usuário);
    - Diálogo por preenchimento de formulários (ex.: Formulários comuns de entrada de dados, mas que tenha um controle rígido do formato destes);
    - Diálogo por manipulação direta (ex.: São entradas de dados complexas que usam abstrações de objetos reais para facilitar a tarefa do usuário);
- Estruturas de interações;
    - Estruturas seqüenciais (ex.: passo a passo, wizards, etc...);
    - Estrutura paralela (ex.: tarefas que podem ser realizadas em qualquer ordem, supondo-se que o usuário saiba a melhor ordem);
    - Estrutura repetitiva (ex.: entradas de dados onde a maior parte das informações são iguais, sendo que a repetição destas entradas devem ser facilitadas pelo sistema);

- Os objetos de interação;
    - Painéis de controle;
        - Janela (ex.: tela principal de um aplicativo e suas telas secundárias);
        - Caixa de diálogo (ex.: Caixas de confirmação de deleção de arquivos);
        - Caixa de ação/tarefa (ex.: Caixas que solicitam parâmetros para realização da tarefa ou confirmação para isto);
        - Formulário e a tela de consulta (entrada e consulta de dados);
        - Caixa de mensagem (ex.: avisos de sistema, advertências, ajudas, etc...);

- Os controles compostos;
    - Painel de menu (conjunto de menus que aparecem no topo das janelas do ambiente Windows – arquivo, exibir, etc...);
    - Página de menu (ex.: como em sistemas antigos que o menu é fixo e identificado por números);
    - Barra de menu (semelhante ao Painel de Menu mas com orientação horizontal);
    - Hipertexto (menu imbricado – Helps, Manuais, onde as páginas possuem relacionamentos como na internet);
    - Barra de ferramentas (ex.: Barras como no Word que podem ser fixas ou flutuantes);
    - Lista de seleção (opções fixas com barra de rolagem);
    - Caixa de combinação (ex.: seleção do tipo da fonte no Word através do menu Formatar->Fontes);
- Os grupos de controles;
    - Grupo de botões de comando;
    - Grupo de botões de rádio (ex.: menu formatar->tabulação do Word, na opção de selecionar o alinhamento);
    - Grupos de caixa de atribuição (são caixas que podem ser marcadas para configuração de uma opção, geralmente com um “x”);
    - Grupos de campos/mostradores de dados;
- Os controles simples;
    - Botão de comando;
    - Botão de seleção;
    - Cursor do dispositivo de apontamento;
    - Escala;
    - Dial;
- Os campos de entrada;
    - Linha e área de comando;
    - Campo de dados;
    - Campo de texto;
    - Campo gráfico;

    - Os mostradores estruturados;
        - Lista/coluna de dados (cabeçalhos de tabelas, identificação de linhas numa tabela);
        - Tabela de dados;
        - Texto;
        - Gráfico;
        - Diagrama de figura;
        - Diagrama de fluxo;
        - Mapa;
    - Os mostradores simples;
        - Mostrador de dados;
    - As orientações;
        - Rótulo (identificação de campos de entradas de dados, geralmente a esquerda ou acima deste);
        - Mensagem;
        - Indicador de progressão;
        - Efeito sonoro;
        - Motivo melódico;
        - Locução e fala;
- Os sistemas de significado;
    - Ícones;
    - Códigos de formas;
    - Denominações;
    - Abreviaturas;
    - Códigos alfanuméricos;
    - Códigos de cores;
    - Códigos de fontes;
    - Códigos de textura;
    - Códigos de vídeo reverso;
    - Códigos de intermitência visual (pisca-pisca);
- As primitivas;
    - As formas visuais;
        - Cores;
        - Fontes;

        - Bordas;
        - Arranjo ou layout;
        - Fundos ou background;
    - Formas sonoras;

3 METODOLOGIA

3.1 Tipo de estudo

Foram utilizadas como forma de estudo a leitura de bibliografias como livros e a própria internet, para auxiliar o desenvolvimento do projeto.

3.2 Local/Contexto

O projeto foi desenvolvido no domicílio do autor, mas contou com as dependências do Rexlab, bem como a sala do orientador do projeto.

3.3 Fontes de Informação

As fontes de informação foram basicamente documentais, e foram coletadas ou pelo autor ou pelo orientador do projeto, mas todos os dois passaram pelos critérios de escolha do último. As características do conteúdo foram voltadas a acessibilidade e tecnologias assistivas tanto de cunho geral quanto de cunho específico.

3.4 Técnicas de coleta e análise de dados

Foi feita a inspeção da usabilidade para qualificar a interface do sistema.

4 DESENVOLVIMENTO DO PROJETO

A aplicação recebeu uma interface inicial, uma para configuração e uma interface principal, que resumem o navegador. Ainda sobre as configurações do sistema, este é capaz de armazenar configurações por usuário do Windows, o que se faz necessário, pois é comum em escolas de ensino especial, a utilização de um computador, por mais de uma pessoa. Quanto a navegação, esta conta com um dicionário de palavras que auxiliará a digitação dos endereços de internet.

4.1 Interfaces

4.1.1 Tela de Entrada

A tela de entrada resume-se em três botões: Navegar, Configurar e Sair. Tendo cada um a função que a própria nomeação sugere. Na Figura 1, apresenta-se a referida tela.

Figura 1 – Tela de Entrada

Como podemos notar, o botão navegar encontra-se destacado, pois pelo método escolhido de seleção de teclas o botão ativo aumenta seu tamanho destacando ao usuário sua ativação.

4.1.2 Tela de Configuração

Clicando no botão Configurar descrito no item anterior, a tela da Figura 2 irá se abrir.

Figura 2 – Tela de Configuração

Como se pode notar existem três abas com funções específicas. Na primeira podemos configurar os Dispositivos de Entrada, que são o Mouse e o Teclado. Para o Mouse pode-se alternar os botões que serão utilizados para ativação das teclas e para o Teclado pode-se desativá-lo, para que não haja interferência no uso do sistema. Nota-se a observação que se encontra nesta última opção (Win9x), pois não é possível utilizar esta função nas versões Windows NT, Windows 2000 e Windows XP.

A segunda aba apresenta configurações das teclas de utilização do sistema, como se pode verificar na Figura 3.

Figura 3 – Tela de Configuração (Teclas)

No primeiro grupo de configurações, Destaque de Teclas, pode-se ajustar a velocidade com que estas destacam-se possibilitando o ajuste para os diferentes usuários do sistema. No segundo grupo, Fontes, pode-se alterar as fontes de apresentação do sistema bem como seu tamanho, adaptando assim as diferentes dificuldades de visão dos possíveis usuários.

Na última aba tem-se a possibilidade de configurar os dicionários, como está visível na figura 4. Nesta tela seleciona-se o dicionário e define-se se este é padrão ou não.

Nesta tela pode-se clicar em “Importar” ou “Excluir Dicionário”. Na primeira opção tem-se a seguinte tela apresentada da figura 5.

Configurando o campo Nome e definindo o caminho do dicionário importa-se um novo dicionário para o sistema.

Figura 4 - Tela de Configuração (Dicionários)

Figura 5 – Tela de Importação de Dicionários

4.1.3 Tela de Navegação

A tela de navegação tem características parecidas a de um navegador de páginas da internet comum. Ao topo, como se pode ver na Figura 6, é apresentado o endereço de navegação atual. Logo abaixo aparecem os botões de controle da navegação como Voltar, Avançar, Atualizar, Parar e Url, nesta ordem.

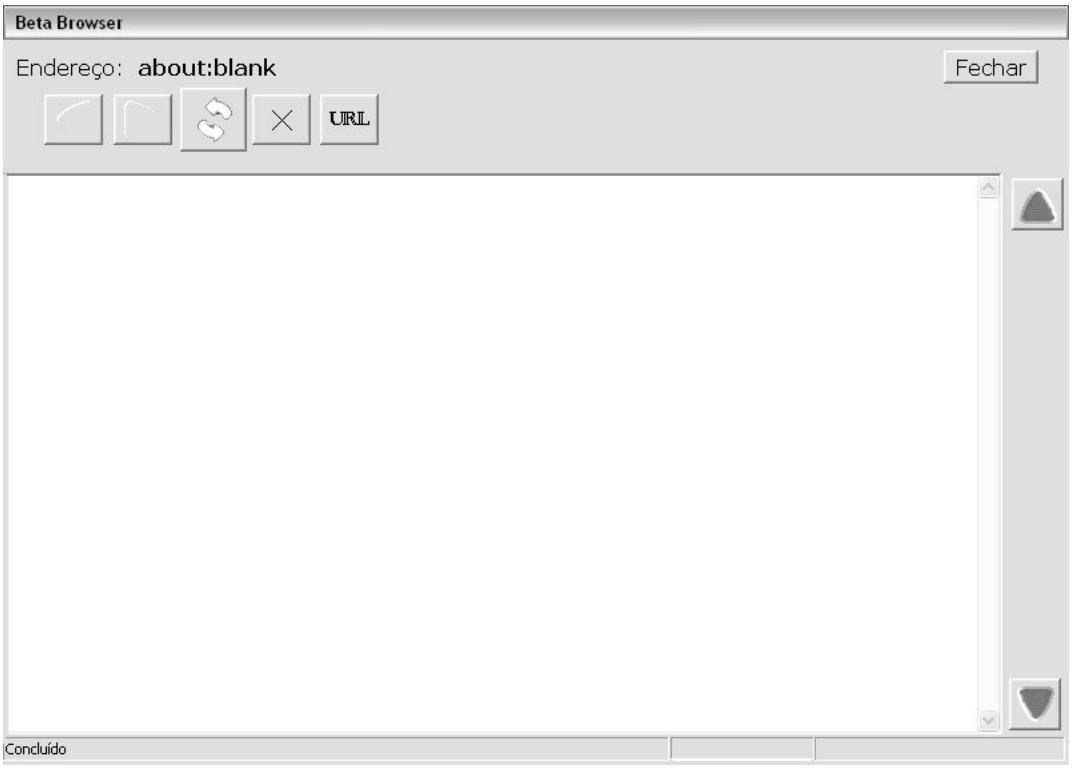

Figura 6 – Tela de Navegação

Os quatro primeiros botões apresentam as funções comuns a todos os navegadores. Já o último abre o teclado virtual para que o usuário possa digitar seu endereço de internet para navegação. Os outros botões, Fechar, Seta para cima e Seta para Baixo, tem respectivamente as funções de fechar o programa, rolar a janela, onde aparecem as páginas da internet, para cima e rolar para baixo.

Já o teclado virtual, que é ativado pelo botão URL, aparece como na Figura 7.

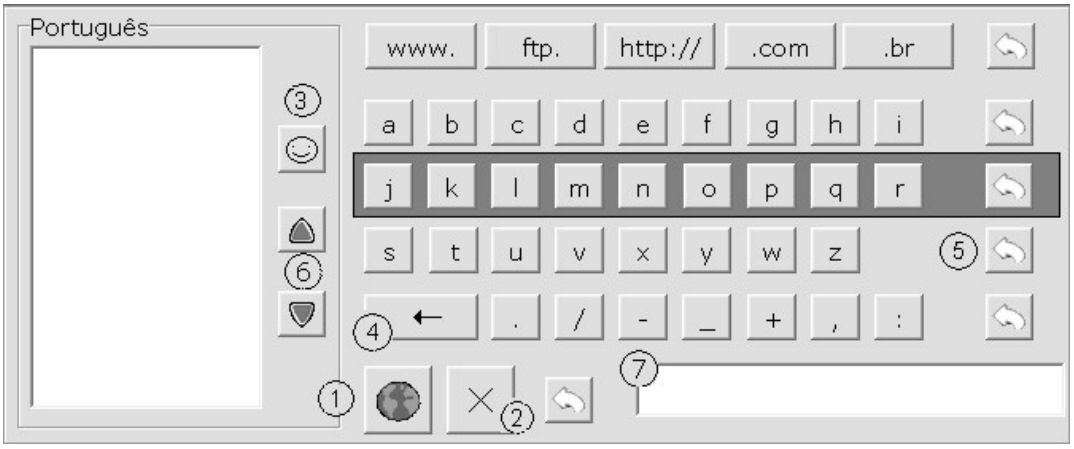

Figura 7 – Teclado Virtual

As linhas do teclado são realçadas através de uma moldura que aparece ao fundo das teclas pertencentes a linha ativa. Ao clicar o botão do mouse, esta moldura fica travada e os botões começam a se alternar, uma coluna de cada vez. Clicando novamente o botão do mouse, aciona-se a tecla que esta ativa e seu texto aparecerá no campo indicado pelo número 7. Ao mesmo tempo, o teclado verifica o histórico de endereços e mostra o que mais se aproxima do texto digitado. Além disso aparecerão no campo identificado como “Português”, possui este nome pois é o dicionário padrão, as sugestões de palavras que podem completar o endereço que está sendo digitado.

Neste teclado foram criados teclas fixas com as extensões de endereços de internet como ‘.com’ e ‘.br’, e ainda os prefixos de endereços como ‘www.’, ‘ftp.’, e ‘http://’.

Para iniciar a navegação com o endereço digitado, usa-se o botão com a legenda número 1, como mostra a figura 7, que irá copiar este endereço para o navegador e fechar o teclado. Também nesta linha, só que com legenda número 2, está o botão para cancelar a digitação.

Caso o usuário queira utilizar a palavra sugerida pelo teclado através do dicionário, basta acionar o botão do mouse no momento em que a tecla, identificada com o número 3, estiver ativa. Com isso, o teclado irá copiar a palavra para o campo 7, voltando a alternar as teclas. Neste mesmo espaço encontram-se duas teclas, identificadas com o número 6, que irão rolar a seleção das palavras sugeridas, permitindo ao usuário selecionar uma palavra sem que esta tenha sido digitada completamente.

O botão número 4, exerce a mesma função do BackSpace do teclado convencional. Além disso cada linha possui um botão para retornar a alternância das linhas caso o usuário tenha entrado na linha errada. Este botão é o número 5, representado por uma seta.

4.2 Códigos de Implementação

Neste tópico serão apresentados trechos de código na linguagem Object Pascal que resolvem alguns dos problemas encontrados no desenvolvimento deste projeto, afim de facilitar a compreensão do leitor ao ler o código fonte (ver Anexo 1).

4.2.1 Restringindo a área de movimentação do Mouse

O navegador Beta Browser delimita o movimento do mouse numa região muito restrita das suas interfaces, para que o usuário, portador de dificuldades motoras, não clique em qualquer lugar da tela, prejudicando o uso da aplicação. Estas áreas restritas resumem-se em regiões sobre a superfície dos botões, fazendo que o usuário, ao clicar no botão do mouse, certifique-se que está clicando em cima do botão ativo.

O procedimento abaixo representa a solução dada a este problema.

```
procedure tBetaLib.RestrictMouse(left,top,rigth,bottom : integer) ;
var
  r : TRect ;
begin
  r := Rect(left,top,rigth,bottom) ;
  ClipCursor( @r ) ;
end;
```

Neste procedimento são utilizadas as funções Rect e ClipCursor. Na primeira é criado um objeto Trect, que define as coordenadas na tela para que a função ClipCursor restrinja o movimento do mouse.

Ao finalizar o programa deve-se liberar o cursor do mouse para que não atrapalhe a execução do Windows. Para isso usa-se o procedimento abaixo.

```
procedure tBetaLib.UnRestrictMouse ;
begin
    ClipCursor( nil ) ;
end;
```

Nele é chamada a função ClipCursor novamente, só que sem passar o objeto Trect como no procedimento para restringir. Com isso, a função interpreta que o mouse irá percorrer toda a tela disponível.

4.2.2 Ocultando o ponteiro do Mouse

Para facilitar o uso da interface do sistema e não causar confusão aos usuários, optou-se pela ocultação do ponteiro do mouse, pois a sua "aparição" em cima dos botões iria obstruir a visão do usuário e causar confusão naquele que fica atento ao movimento do ponteiro. Desta forma o usuário acostuma-se com sua ausência e prende sua atenção nos botões que estão alternando.

Para esta tarefa foi criada a função MouseShowCursor, como apresentada abaixo.

```
function tBetaLib.mouseshowcursor(const Show: boolean): boolean;
var
   I: integer;
begin
   I := ShowCursor(LongBool(true));
   if Show then
   begin
     Result := I >= 0;
     while I < 0 do
     begin
        Result := ShowCursor(LongBool(true)) >= 0;
```

```
      Inc(I);
    end;
  end
  else
  begin
    Result := I < 0;
    while I >= 0 do
    begin
      Result := ShowCursor(LongBool(false)) < 0;
      Dec(I);
    end;
  end;
end;
```

4.2.3 Rolagem da tela

Algumas páginas da internet definem as setas de rolagem em locais diferentes do que o canto superior e inferior direito do navegador, devido aos *frames* do html. Para isso foi utilizado um componente do Delphi chamado ScrollBox. Este componente é colocado no formulário em um tamanho fixo e todo aquele componente adicionado dentro dele que exceder este tamanho, fará com que o ScrollBox cria um barra de rolagem no lado direito. Sendo assim, foi colocado dentro deste o componente que abre as páginas da internet já com um tamanho maior que do ScrollBox. Quando o usuário clica no botão de rolagem para cima ou para baixo, o seguinte procedimento é executado.

```
procedure tBetaLib.SimulaClick(x,y : integer);
begin
  Mouse_Event(MOUSEEVENTF_ABSOLUTE or
                MOUSEEVENTF_LEFTDOWN, x, y, 0, 0);
  Mouse_Event(MOUSEEVENTF_ABSOLUTE or MOUSEEVENTF_LEFTUP,
                x, y, 0, 0);
end;
```

Nota-se que os parâmetros x e y definem a posição em que o mouse será posicionado para a simulação do clique. Este parâmetros, no caso da rolagem, devem ser exatamente a posição por sobre os botões da barra de rolagem do componente ScrollBox.

4.2.4 Configuração por usuário

Para garantir a configuração do sistema específica para cada usuário, foi utilizado o usuário do windows para criação dos arquivos de configuração. Na criação do aplicativo o seguinte trecho de código é executado.

```
Registro := TRegistry.Create;
try
begin
   Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',
                              false);
   if Registro.KeyExists('\Software\Microsoft\Windows\CurrentVersion\Explorer')
     and  registro.ValueExists('Logon User Name') then
   begin
      Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\
                                  Explorer',false);
      if not Fileexists(getcurrentdir + '\' + registro.ReadString('Logon User
                             Name') +  '.ini') then
         copyfile(pchar(getcurrentdir + '\betaconf.ini'),pchar(getcurrentdir + '\' +
                    registro.ReadString('Logon User Name') + '.ini'),true);
         iniconf := tinifile.Create(getcurrentdir + '\' + registro.ReadString('Logon
                                      User  Name') + '.ini');
   end;
end;
except
   On E: Exception do
        iniconf := tinifile.Create(getcurrentdir + '\betaconf.ini');
end;
```

```
try
begin
   Registro.RootKey := HKEY_LOCAL_MACHINE;
   Registro.OpenKey('\Network\Logon',false);
   if Registro.KeyExists('\Network\Logon') and registro.ValueExists('username')
   then
   begin
      if not Fileexists(getcurrentdir + '\' + registro.ReadString('username') + '.ini')
      then
        copyfile(pchar(getcurrentdir + '\betaconf.ini'),pchar(getcurrentdir + '\' +
                  registro.ReadString('username') + '.ini'),true);
        iniconf := tinifile.Create(getcurrentdir + '\' +
                                    Registro.ReadString('username') +  '.ini');
   end;
end;
except
  On E: Exception do  iniconf := tinifile.Create(getcurrentdir + '\betaconf.ini');
end;
```

Este procedimento faz a leitura do registro do Windows e busca o usuário que está logado no momento no sistema operacional, então cria o arquivo .ini de configuração do sistema, caso não exista. Se já existir faz uma cópia do arquivo betaconf.ini.

5 CONCLUSÃO

Consciente da importância deste trabalho para a sociedade, tendo em vista a escassez de pesquisa nesta área, e que este facilitará o manuseio das informações da internet pelos portadores de necessidades especiais classificados no trabalho como beneficiários da ferramenta, conclui-se que esta demonstrou-se muito útil, mas que ainda necessita de aperfeiçoamentos.

O autor deixa aqui a necessidade da continuidade do desenvolvimento desta aplicação, seja por ele mesmo ou por um terceiro, agregando novas funcionalidades e características à ferramenta.

Como nova funcionalidade, propõe-se a inclusão de soluções para outros tipos de deficiências ou incapacidades, como a falta de visão, ou paralisias dos membros superiores e inferiores, o que exigiria a capacidade de reconhecimento de voz ou algum outro método que possibilite o manuseio da interface.

6 REFERÊNCIAS

[1] DATI. **FAQ: Frequently Asked Questions**. Disponível na internet. **http://www.asel.udel.edu/dati/atdef.html**. 20 março 2001.

[2] KOON, Ricardo A, VEGA, Maria Eugenia de la. **El impacto tecnológico en las personas com discapacidad**. In: Congresso International de Informática Educativa Especial, 2, 2000.

[3] MONTOYA, Rafael S. **Los sistemas de ayudas basados en la tecnologia de la informacíon**. In: Congresso Internacional de Informática Educativa Especial, 1, 1998, Neuquén – Argentina.

[4] NEWELL, Alan F, GREGOR, Peter. Human Computer Interfaces for people with disabilities. In: ELSEVIER SCIENCE. **Handbook of Human-Computer Interaction**. 1997, p.813-824.

[5] CYBIS, Walter de Abreu. Engenharia de Usabilidade: Uma Abordagem Ergonômica. Disponível em: http://www.labiutil.inf.ufsc.br/apostila.htm. Acesso em: 5 de julho de 2003.

ANEXOS

ANEXO I

- CÓDIGO FONTE -

1 - Código da tela Inicial:

```
unit UInicial;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls, Buttons, UBetaLib, inifiles;

type
  TFInicial = class(TForm)
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    SpeedButton3: TSpeedButton;
    Timer1: TTimer;
    NavegarSpeedButton: TSpeedButton;
    ConfigurarSpeedButton: TSpeedButton;
    SairSpeedButton: TSpeedButton;
    procedure Timer1Timer(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormActivate(Sender: TObject);
    procedure NavegarSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure ConfigurarSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure SairSpeedButtonMouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);

  private
    procedure AjustaTela(tela : tscreen);
```

```
  public
    { Public declarations }
  end;

var
  FInicial: TFInicial;
  activebutton : integer;
  iniconf : tinifile;

implementation

Uses UPrincipal, UConfigura, Registry;

var
  Tempo : integer;

{$R *.DFM}

procedure TFInicial.Timer1Timer(Sender: TObject);
begin
  Tempo := iniconf.ReadInteger('Botoes', 'Velocidade', 10) * 100;
  timer1.Interval := Tempo;

  case activebutton of
    0 : begin
          NavegarSpeedButton.Visible := true;
          SairSpeedButton.Visible := false;
          BetaLib.RestrictMouse(NavegarSpeedButton.Left +
(NavegarSpeedButton.Width
                                      div 2) + Finicial.Left, NavegarSpeedButton.Top +
                                       NavegarSpeedButton.height + Finicial.top,
                                       NavegarSpeedButton.Left +
(NavegarSpeedButton.Width
                                       div 2) + Finicial.left, NavegarSpeedButton.Top +
```

```
                                    NavegarSpeedButton.height + Finicial.Top);
        ActiveButton := 1;
       end;
   1 : begin
        ConfigurarSpeedButton.Visible := true;
        NavegarSpeedButton.Visible := false;
        BetaLib.RestrictMouse(ConfigurarSpeedButton.Left +
                              (ConfigurarSpeedButton.Width div 2) +
Finicial.Left,
                              ConfigurarSpeedButton.Top +
                              ConfigurarSpeedButton.height + Finicial.top,
                              ConfigurarSpeedButton.Left +
                              (ConfigurarSpeedButton.Width div 2) +
Finicial.left,
                              ConfigurarSpeedButton.Top +
                              ConfigurarSpeedButton.height + Finicial.Top);
        ActiveButton := 2;
       end;
   2 : begin
        SairSpeedButton.Visible := true;
        ConfigurarSpeedButton.Visible := false;
        BetaLib.RestrictMouse(SairSpeedButton.Left + (SairSpeedButton.Width
div 2) +
                              Finicial.Left, SairSpeedButton.Top +
                              SairSpeedButton.height + Finicial.top,
                              SairSpeedButton.Left + (SairSpeedButton.Width
div 2) +
                              Finicial.left, SairSpeedButton.Top +
                              SairSpeedButton.height + Finicial.Top);
        ActiveButton := 0;
       end;
  end;
end;
```

```
procedure TFInicial.FormShow(Sender: TObject);
begin
  activebutton := 0;
  AjustaTela(screen);
end;


procedure TFInicial.AjustaTela(tela : tscreen);
var
  i : integer;
begin
  FInicial.Font.Size := iniconf.ReadInteger('Fonte','Tamanho',8);
  FInicial.Font.Name := iniconf.ReadString('Fonte','Tipo','Verdana');

  tela.Realign;
  if (tela.Width = 1024) and (tela.Height = 768) then
  begin
    for i := 0 to ComponentCount -1 do
    begin
      if pos('Timer',twincontrol(components[i]).name) = 0 then
      begin
        TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.18);
        TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.18);
        TWinControl(Components[i]).Width :=
Round(TWinControl(Components[i]).Width * 1.10);
        TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height * 1.10);
      end;
      if pos('Speed',TControl(Components[i]).Name) <> 0 then
      begin
        TSpeedButton(Components[i]).Font.Size :=
iniconf.ReadInteger('Fonte','Tamanho',8);
```

```
        TSpeedButton(Components[i]).Font.Name :=
iniconf.ReadString('Fonte','Tipo','Verdana');
      end;
      width := round(width * 1.02);
      height := round(height * 1.02);
    end;
    FPrincipal.Font.Size := Round(FPrincipal.Font.Size * 1.11);
  end;

  if (tela.Width = 1280) and (tela.Height = 1024) then
  begin
    for i := 0 to ComponentCount -1 do
    begin
      if pos('Timer',twincontrol(components[i]).name) = 0 then
      begin
        TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.63);
        TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top *1.90);
        TWinControl(Components[i]).Width :=
Round(TWinControl(Components[i]).Width * 1.60);
        TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height * 1.60);
      end;
      if pos('Speed',TControl(Components[i]).Name) <> 0 then
      begin
        TSpeedButton(Components[i]).Font.Size :=
iniconf.ReadInteger('Fonte','Tamanho',8);
        TSpeedButton(Components[i]).Font.Name :=
iniconf.ReadString('Fonte','Tipo','Verdana');
      end;
      width := round(width * 1.07);
      height := round(height * 1.07);
    end;
```

```
  end;
  tela.ResetFonts;
end;

procedure TFInicial.FormCreate(Sender: TObject);
var
  Registro : TRegistry;
begin
  BetaLib.mouseshowcursor(false);

  Registro := TRegistry.Create;
  try
  begin

Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
    if Registro.KeyExists('\Software\Microsoft\Windows\CurrentVersion\Explorer')
and
       registro.ValueExists('Logon User Name') then
    begin

Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
      if not Fileexists('.\' + registro.ReadString('Logon User Name') + '.ini') then
        copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('Logon User
Name') +
                    '.ini'),true);
        iniconf := tinifile.Create('.\' + registro.ReadString('Logon User Name') +
'.ini');
    end;
  end;
  except
    On E: Exception do iniconf := tinifile.Create('.\betaconf.ini');
  end;
```

```
  try
  begin
    Registro.RootKey := HKEY_LOCAL_MACHINE;
    Registro.OpenKey('\Network\Logon',false);
    if Registro.KeyExists('\Network\Logon') and registro.ValueExists('username')
then
    begin
      if not Fileexists('.\' + registro.ReadString('username') + '.ini') then
      copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('username') +
                '.ini'),true);
      iniconf := tinifile.Create('.\' + registro.ReadString('username') + '.ini');
    end;
  end;
  except
    On E: Exception do iniconf := tinifile.Create('.\betaconf.ini');
  end;
end;

procedure TFInicial.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  iniconf.free;
  BetaLib.UnRestrictMouse;
end;

procedure TFInicial.FormActivate(Sender: TObject);
begin

BetaLib.RestrictMouse(FInicial.Left+10,FInicial.Top+40,FInicial.Left+10,FInicial.
Top + 40);
  BetaLib.delay(iniconf.ReadInteger('Botoes','Velocidade',10)*100);
  Timer1.Enabled := true;
end;
```

```
procedure TFInicial.NavegarSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  Timer1.Enabled := false;
  FPrincipal.showmodal;

BetaLib.RestrictMouse(FInicial.Left+5,Finicial.Top+40,FInicial.Left+5,FInicial.To
p+40);
  Timer1.Enabled := true;
end;

procedure TFInicial.ConfigurarSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  Timer1.Enabled := false;

  BetaLib.UnRestrictMouse;
  BetaLib.mouseshowcursor(true);

  FConfigura.ShowModal;

  FInicial.Font.Size := iniconf.ReadInteger('Fonte','Tamanho',8);
  FInicial.Font.Name := iniconf.ReadString('Fonte','Tipo','Verdana');

  BetaLib.mouseshowcursor(false);
  BetaLib.RestrictMouse(ConfigurarSpeedButton.Left +
(ConfigurarSpeedButton.Width
                        div 2) + Finicial.Left, ConfigurarSpeedButton.Top +
                        ConfigurarSpeedButton.height + Finicial.top,
                        ConfigurarSpeedButton.Left +
(ConfigurarSpeedButton.Width
                        div 2) + Finicial.left, ConfigurarSpeedButton.Top +
                        ConfigurarSpeedButton.height + Finicial.Top);
```

```
  Timer1.Enabled := true;
end;

procedure TFInicial.SairSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  close;
  BetaLib.UnRestrictMouse;
end;

end.
```

2 - Código da tela Principal:

```
unit UPrincipal;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, OleCtrls, SHDocVw, ComCtrls, Buttons, comobj, Db, AdsData,
  AdsFunc, AdsTable, ADSSET, dm, DBCtrls, Mask, ToolEdit, CurrEdit,
  RXDBCtrl, ExtCtrls, OleCtnrs, DBLookup, sCtrlResize, TeeProcs,
  TeEngine, Chart, UBetaLib, inifiles;

type
  TFprincipal = class(TForm)
    Label1: TLabel;
    StatusBar1: TStatusBar;
    SpeedButton1: TSpeedButton;
    SpeedButton2: TSpeedButton;
    SpeedButton3: TSpeedButton;
```

ProgressBar1: TProgressBar;
AlternaBotoesTimer: TTimer;
SpeedButton5: TSpeedButton;
SpeedButton6: TSpeedButton;
SpeedButton7: TSpeedButton;
SpeedButton4: TSpeedButton;
SpeedButton8: TSpeedButton;
GroupBox1: TGroupBox;
GroupBox2: TGroupBox;
SpeedButton49: TSpeedButton;
DicListBox: TListBox;
SpeedButton9: TSpeedButton;
SpeedButton10: TSpeedButton;
SpeedButton11: TSpeedButton;
SpeedButton12: TSpeedButton;
SpeedButton13: TSpeedButton;
SpeedButton22: TSpeedButton;
SpeedButton21: TSpeedButton;
SpeedButton20: TSpeedButton;
SpeedButton19: TSpeedButton;
SpeedButton18: TSpeedButton;
SpeedButton17: TSpeedButton;
SpeedButton16: TSpeedButton;
SpeedButton15: TSpeedButton;
SpeedButton14: TSpeedButton;
SpeedButton23: TSpeedButton;
SpeedButton24: TSpeedButton;
SpeedButton25: TSpeedButton;
SpeedButton26: TSpeedButton;
SpeedButton27: TSpeedButton;
SpeedButton28: TSpeedButton;
SpeedButton29: TSpeedButton;
SpeedButton30: TSpeedButton;
SpeedButton31: TSpeedButton;

SpeedButton39: TSpeedButton;
SpeedButton38: TSpeedButton;
SpeedButton37: TSpeedButton;
SpeedButton36: TSpeedButton;
SpeedButton35: TSpeedButton;
SpeedButton34: TSpeedButton;
SpeedButton33: TSpeedButton;
SpeedButton32: TSpeedButton;
SpeedButton50: TSpeedButton;
SpeedButton40: TSpeedButton;
SpeedButton41: TSpeedButton;
SpeedButton42: TSpeedButton;
SpeedButton43: TSpeedButton;
SpeedButton46: TSpeedButton;
SpeedButton47: TSpeedButton;
SpeedButton48: TSpeedButton;
TecladoTimer: TTimer;
SpeedButton45: TSpeedButton;
SpeedButton44: TSpeedButton;
Shape1: TShape;
Shape2: TShape;
Shape3: TShape;
Shape4: TShape;
Shape5: TShape;
Shape6: TShape;
wwwSpeedButton: TSpeedButton;
ftpSpeedButton: TSpeedButton;
httpSpeedButton: TSpeedButton;
comSpeedButton: TSpeedButton;
brSpeedButton: TSpeedButton;
aSpeedButton: TSpeedButton;
bSpeedButton: TSpeedButton;
cSpeedButton: TSpeedButton;
dSpeedButton: TSpeedButton;

eSpeedButton: TSpeedButton;
fSpeedButton: TSpeedButton;
gSpeedButton: TSpeedButton;
hSpeedButton: TSpeedButton;
iSpeedButton: TSpeedButton;
jSpeedButton: TSpeedButton;
kSpeedButton: TSpeedButton;
lSpeedButton: TSpeedButton;
mSpeedButton: TSpeedButton;
nSpeedButton: TSpeedButton;
oSpeedButton: TSpeedButton;
pSpeedButton: TSpeedButton;
qSpeedButton: TSpeedButton;
rSpeedButton: TSpeedButton;
sSpeedButton: TSpeedButton;
tSpeedButton: TSpeedButton;
uSpeedButton: TSpeedButton;
vSpeedButton: TSpeedButton;
xSpeedButton: TSpeedButton;
ySpeedButton: TSpeedButton;
wSpeedButton: TSpeedButton;
zSpeedButton: TSpeedButton;
backSpeedButton: TSpeedButton;
pontoSpeedButton: TSpeedButton;
barraSpeedButton: TSpeedButton;
hifenSpeedButton: TSpeedButton;
underSpeedButton: TSpeedButton;
maisSpeedButton: TSpeedButton;
virgulaSpeedButton: TSpeedButton;
doispontosSpeedButton: TSpeedButton;
navegarSpeedButton: TSpeedButton;
cancelarSpeedButton: TSpeedButton;
smileSpeedButton: TSpeedButton;
VoltarSpeedButton: TSpeedButton;

```
AvancarSpeedButton: TSpeedButton;
AtualizarSpeedButton: TSpeedButton;
PararSpeedButton: TSpeedButton;
URLSpeedButton: TSpeedButton;
FecharSpeedButton: TSpeedButton;
SubirSpeedButton: TSpeedButton;
DescerSpeedButton: TSpeedButton;
SpeedButton51: TSpeedButton;
SpeedButton52: TSpeedButton;
SpeedButton53: TSpeedButton;
SpeedButton54: TSpeedButton;
SpeedButton55: TSpeedButton;
SpeedButton56: TSpeedButton;
Voltar1SpeedButton: TSpeedButton;
Voltar2SpeedButton: TSpeedButton;
Voltar3SpeedButton: TSpeedButton;
Voltar4SpeedButton: TSpeedButton;
Voltar5SpeedButton: TSpeedButton;
Voltar6SpeedButton: TSpeedButton;
URLEdit: TEdit;
Label2: TLabel;
ScrollBox1: TScrollBox;
WebBrowser1: TWebBrowser;
SpeedButton57: TSpeedButton;
SpeedButton58: TSpeedButton;
SobeDicSpeedButton: TSpeedButton;
DesceDicSpeedButton: TSpeedButton;
procedure FormCreate(Sender: TObject);
procedure WebBrowser1TitleChange(Sender: TObject;
  const Text: WideString);
procedure WebBrowser1ProgressChange(Sender: TObject; Progress,
  ProgressMax: Integer);
procedure FormActivate(Sender: TObject);
procedure salvaurl(url : string);
```

```
    procedure navega;
    procedure AlternaBotoesTimerTimer(Sender: TObject);
    procedure WebBrowser1NewWindow2(Sender: TObject; var ppDisp:
IDispatch;
      var Cancel: WordBool);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure FormShow(Sender: TObject);
    function  CaptureScreenRect( ARect: TRect ): TBitmap;
    procedure AjustaTela(tela : tscreen);
    procedure TecladoTimerTimer(Sender: TObject);
    procedure Shape1MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Shape2MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Shape3MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Shape4MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Shape5MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Shape6MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure LocalizaUrl(URL : string);
    procedure URLSpeedButtonMouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure VoltarSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure AvancarSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure AtualizarSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure PararSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
```

```
    procedure FecharSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure wwwSpeedButtonMouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure ftpSpeedButtonMouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure httpSpeedButtonMouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure comSpeedButtonMouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure brSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Voltar1SpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure aSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure bSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure cSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure dSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure eSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure fSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure gSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure hSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
```

```
procedure iSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure Voltar2SpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
procedure jSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure kSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure lSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure mSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure nSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure oSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure pSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure qSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure rSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure Voltar3SpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
procedure sSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure tSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure uSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure vSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
procedure xSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
```

```
    procedure ySpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure wSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure zSpeedButtonMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Voltar4SpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure backSpeedButtonMouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure pontoSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure barraSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure hifenSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure underSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure maisSpeedButtonMouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure virgulaSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure doispontosSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure Voltar5SpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure navegarSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure cancelarSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure Voltar6SpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
```

```
    procedure ScrollBox1MouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure DescerSpeedButtonMouseDown(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure SubirSpeedButtonMouseDown(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure CarregaListBox(Lista : TStrings);
    procedure smileSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure SobeDicSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure DesceDicSpeedButtonMouseUp(Sender: TObject;
      Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
    procedure ButtonTask(caption : TCaption; var visible : boolean);
    procedure WebBrowser1StatusTextChange(Sender: TObject;
      const Text: WideString);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Fprincipal: TFprincipal;
  NumButton, NumShape, NumButtonsShape : integer;
  ButtonsShape1 : array [1..6] of TSpeedButton;
  ButtonsShape2, ButtonsShape3 : array [1..10] of TSpeedButton;
  ButtonsShape4, ButtonsShape5 : array [1..9] of TSpeedButton;
  ButtonsShape6 : array [1..3] of TSpeedButton;
  control : tcontrol;
  shape, ActiveButton, iniciado : boolean;
  Texto, auxURL : String;

implementation
```

```
uses UInicial, registry;

var
  Tempo, IndicePalavra : integer;
  iniconf : Tinifile;
  PalavraDgt : array [1..20] of string;
  SelDic : boolean;
  Lista : TStringList;
  auxLista : TStrings;

{$R *.DFM}

procedure TFprincipal.FormCreate(Sender: TObject);
var
  Registro : TRegistry;
begin
  Label2.Caption := 'about:blank';
  webbrowser1.Navigate('about:blank');

  speedbutton1.Enabled := false;
  speedbutton2.Enabled := false;

  Registro := TRegistry.Create;
  try
  begin

Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
    if Registro.KeyExists('\Software\Microsoft\Windows\CurrentVersion\Explorer')
and registro.ValueExists('Logon User Name') then
    begin
```

```
Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
     if not Fileexists('.\' + registro.ReadString('Logon User Name') + '.ini') then
       copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('Logon User
Name')
                  +'.ini'),true);
       iniconf := tinifile.Create('.\' + registro.ReadString('Logon User Name') +
'.ini');
    end;
  end;
  except
    On E: Exception do iniconf := tinifile.Create('.\betaconf.ini');
  end;

  try
  begin
    Registro.RootKey := HKEY_LOCAL_MACHINE;
    Registro.OpenKey('\Network\Logon',false);
    if Registro.KeyExists('\Network\Logon') and registro.ValueExists('username')
then
    begin
      if not Fileexists('.\' + registro.ReadString('username') + '.ini') then
      copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('username') +
'.ini'),true);
      iniconf := tinifile.Create('.\' + registro.ReadString('username') + '.ini');
    end;
  end;
  except
    On E: Exception do iniconf := tinifile.Create('.\betaconf.ini');
  end;

  MD.URLS.DataBaseName := getcurrentdir;
  MD.URLS.Active := true;
```

```
  SelDic := false;
end;

procedure TFprincipal.WebBrowser1TitleChange(Sender: TObject;
  const Text: WideString);
begin
  if webbrowser1.LocationURL <> 'about:blank' then
    URLEdit.text := webbrowser1.LocationURL;
end;

procedure TFprincipal.WebBrowser1ProgressChange(Sender: TObject;
Progress,
  ProgressMax: Integer);
begin
  progressbar1.Max := progressmax;
  progressbar1.Position := progress;
end;

procedure TFprincipal.FormActivate(Sender: TObject);
begin

BetaLib.RestrictMouse(FPrincipal.Left+10,FPrincipal.Top+40,FPrincipal.Left+10
,FPrincipal.Top
                                +40);

  Lista := TStringList.Create;

  Lista.LoadFromFile(iniconf.ReadString('Dicionario','Default',''));

  numbutton := 1;
  NumShape := 1;
  NumButtonsShape := 1;
  shape := true;
```

```
ActiveButton := false;

md.URLS.IndexName := 'urls';

WebBrowser1.Height := 2000;
ScrollBox1.VertScrollBar.Range := WebBrowser1.Height;
webbrowser1.SetFocus;

ButtonsShape1[1] := wwwSpeedButton;
ButtonsShape1[2] := ftpSpeedButton;
ButtonsShape1[3] := httpSpeedButton;
ButtonsShape1[4] := comSpeedButton;
ButtonsShape1[5] := brSpeedButton;
ButtonsShape1[6] := Voltar1SpeedButton;

ButtonsShape2[1] := aSpeedButton;
ButtonsShape2[2] := bSpeedButton;
ButtonsShape2[3] := cSpeedButton;
ButtonsShape2[4] := dSpeedButton;
ButtonsShape2[5] := eSpeedButton;
ButtonsShape2[6] := fSpeedButton;
ButtonsShape2[7] := gSpeedButton;
ButtonsShape2[8] := hSpeedButton;
ButtonsShape2[9] := iSpeedButton;
ButtonsShape2[10] := Voltar2SpeedButton;

ButtonsShape3[1] := jSpeedButton;
ButtonsShape3[2] := kSpeedButton;
ButtonsShape3[3] := lSpeedButton;
ButtonsShape3[4] := mSpeedButton;
ButtonsShape3[5] := nSpeedButton;
ButtonsShape3[6] := oSpeedButton;
ButtonsShape3[7] := pSpeedButton;
ButtonsShape3[8] := qSpeedButton;
```

```
  ButtonsShape3[9] := rSpeedButton;
  ButtonsShape3[10] := Voltar3SpeedButton;

  ButtonsShape4[1] := sSpeedButton;
  ButtonsShape4[2] := tSpeedButton;
  ButtonsShape4[3] := uSpeedButton;
  ButtonsShape4[4] := vSpeedButton;
  ButtonsShape4[5] := xSpeedButton;
  ButtonsShape4[6] := ySpeedButton;
  ButtonsShape4[7] := wSpeedButton;
  ButtonsShape4[8] := zSpeedButton;
  ButtonsShape4[9] := Voltar4SpeedButton;

  ButtonsShape5[1] := backSpeedButton;
  ButtonsShape5[2] := pontoSpeedButton;
  ButtonsShape5[3] := barraSpeedButton;
  ButtonsShape5[4] := hifenSpeedButton;
  ButtonsShape5[5] := underSpeedButton;
  ButtonsShape5[6] := maisSpeedButton;
  ButtonsShape5[7] := virgulaSpeedButton;
  ButtonsShape5[8] := doispontosSpeedButton;
  ButtonsShape5[9] := Voltar5SpeedButton;

  ButtonsShape6[1] := navegarSpeedButton;
  ButtonsShape6[2] := cancelarSpeedButton;
  ButtonsShape6[3] := Voltar6SpeedButton;

  BetaLib.delay(iniconf.ReadInteger('Botoes','Velocidade',10)*100);

  AlternaBotoesTimer.Enabled := true;
end;

procedure TFprincipal.salvaurl(url : string);
begin
```

```
  if not md.URLS.AdsSeek(url,sthard) then
  begin
    md.urls.append;
    md.urlsurl.value := URLEdit.text;
    md.urls.post;
  end;
end;

procedure TFprincipal.navega;
begin
  speedbutton1.Enabled := true;
  webbrowser1.Navigate(URLEdit.text);
  salvaurl(URLEdit.text);
end;

procedure TFprincipal.AlternaBotoesTimerTimer(Sender: TObject);
begin
  case numbutton of
    1 : begin
          VoltarSpeedButton.Visible := true;
          DescerSpeedButton.Visible := false;
          BetaLib.RestrictMouse(VoltarSpeedButton.Left +
(VoltarSpeedButton.Width div 2),
                                VoltarSpeedButton.Top +
VoltarSpeedButton.height,
                                VoltarSpeedButton.Left +
(VoltarSpeedButton.Width div 2),
                                VoltarSpeedButton.Top +
VoltarSpeedButton.height);
          numbutton := 2;
        end;
    2 : begin
          AvancarSpeedButton.Visible := true;
          VoltarSpeedButton.Visible := false;
```

```
        BetaLib.RestrictMouse(AvancarSpeedButton.Left +
(AvancarSpeedButton.Width div 2),
                              AvancarSpeedButton.Top +
AvancarSpeedButton.height,
                              AvancarSpeedButton.Left +
(AvancarSpeedButton.Width div 2),
                              AvancarSpeedButton.Top +
AvancarSpeedButton.height);
        numbutton := 3;
       end;
   3 : begin
        AtualizarSpeedButton.Visible := true;
        AvancarSpeedButton.Visible := false;
        BetaLib.restrictmouse(AtualizarSpeedButton.Left +
(AtualizarSpeedButton.Width div 2),
                              AtualizarSpeedButton.Top +
AtualizarSpeedButton.height,
                              AtualizarSpeedButton.Left +
(AtualizarSpeedButton.Width div 2),
                              AtualizarSpeedButton.Top +
AtualizarSpeedButton.height);
        numbutton := 4;
       end;
   4 : begin
        PararSpeedButton.Visible := true;
        AtualizarSpeedButton.Visible := false;
        BetaLib.restrictmouse(PararSpeedButton.Left +
(PararSpeedButton.Width div 2),
                              PararSpeedButton.Top +
PararSpeedButton.height,
                              PararSpeedButton.Left +
(PararSpeedButton.Width div 2),
                              PararSpeedButton.Top +
PararSpeedButton.height);
```

```
        numbutton := 5;
      end;
   5 : begin
        URLSpeedButton.Visible := true;
        PararSpeedButton.Visible := false;
        BetaLib.restrictmouse(URLSpeedButton.Left + (URLSpeedButton.Width
div 2),
                              URLSpeedButton.Top + URLSpeedButton.height,
                              URLSpeedButton.Left + (URLSpeedButton.Width
div 2),
                              URLSpeedButton.Top + URLSpeedButton.height);
        numbutton := 6;
      end;
   6 : begin
        FecharSpeedButton.Visible := true;
        URLSpeedButton.Visible := false;
        BetaLib.restrictmouse(FecharSpeedButton.Left +
(FecharSpeedButton.Width div 2),
                              FecharSpeedButton.Top +
FecharSpeedButton.height,
                              FecharSpeedButton.Left +
(FecharSpeedButton.Width div 2),
                              FecharSpeedButton.Top +
FecharSpeedButton.height);
        numbutton := 7;
      end;
   7 : begin
        SubirSpeedButton.Visible := true;
        FecharSpeedButton.Visible := false;
        BetaLib.restrictmouse(SubirSpeedButton.Left +
10,SubirSpeedButton.Top +
                             30,SubirSpeedButton.Left +
10,SubirSpeedButton.Top + 30);
        numbutton := 8;
```

```
        end;
    8 : begin
          DescerSpeedButton.Visible := true;
          SubirSpeedButton.Visible := false;
          BetaLib.restrictmouse(DescerSpeedButton.Left +
10,DescerSpeedButton.Top +
                                      30,DescerSpeedButton.Left +
10,DescerSpeedButton.Top + 30);
          numbutton := 1;
        end;
  end;
  AlternaBotoesTimer.Interval := Tempo;
end;

procedure TFprincipal.WebBrowser1NewWindow2(Sender: TObject;
  var ppDisp: IDispatch; var Cancel: WordBool);
begin
  cancel := true;
end;

procedure TFprincipal.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  AlternaBotoesTimer.Enabled := false;
  BetaLib.mouseshowcursor(false);
  BetaLib.UnRestrictMouse;
end;

procedure TFPrincipal.AjustaTela(tela : tscreen);
var
  i : integer;
begin

  FPrincipal.Font.Size := iniconf.ReadInteger('Fonte','Tamanho',8);
  FPrincipal.Font.Name := iniconf.ReadString('Fonte','Tipo','Verdana');
```

```
  FecharSpeedButton.Font.Size := iniconf.ReadInteger('Fonte','Tamanho',8);
  FecharSpeedButton.Font.Name := iniconf.ReadString('Fonte','Tipo','Verdana');

  SpeedButton8.Font.Name := iniconf.ReadString('Fonte','Tipo','Verdana');
  SpeedButton8.Font.Size := iniconf.ReadInteger('Fonte','Tamanho',8);

  tela.Realign;
  if (tela.Width = 1024) and (tela.Height = 768) then
  begin
    for i := 0 to ComponentCount -1 do
    begin { Varre todos os componentes do form que possam ser redefinidos
(classe TWinControl) }
      if pos('Speed',twincontrol(components[i]).name) > 0 then
      begin
        if (pos('DescerSpeedButton',twincontrol(components[i]).name) > 0) or
(pos('SpeedButton7',twincontrol(components[i]).name) > 0) then
        begin
          TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.28);
          TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.315);
        end
        else
        begin
          if (pos('SpeedButton7',twincontrol(components[i]).name) > 0) then
          begin
            TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.295);
            TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.32);
          end
          else
```

```
          if (pos('SubirSpeedButton',twincontrol(components[i]).name) > 0) or
(pos('SpeedButton6',twincontrol(components[i]).name) > 0) then
          begin
            TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.28);
            TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.37);
          end
          else
          begin
            TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.29);
            TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.37);
          end
       end;
     end
     else
     begin
       if pos('Timer',twincontrol(components[i]).name) = 0 then
       begin
         if pos('WebBrowser',twincontrol(components[i]).name) > 0 then
         begin
           TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.32);
           TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.36);
         end
         else
         begin
           if pos('Shape',TWinControl(Components[i]).Name) > 0then
           begin
             TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.31);
```

```
            TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.35);
          end
          else
          begin
            TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.31);
            TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.32);
          end;
        end;
      end;
    end;

    if (Components[i] is TWinControl) or (Components[i] is TbuttonControl) or
(Components[i] is TGraphicControl) then
    begin { Redefine os componentes em proporção ao original }
      if twincontrol(components[i]).name <> 'StatusBar1' then
      begin
        if pos('Shape',TGraphicControl(components[i]).Name) > 0 then
        begin
          TGraphicControl(Components[i]).Width :=
                          Round(TGraphicControl(Components[i]).Width * 1.29);
          TGraphicControl(Components[i]).Height :=
                          Round(TGraphicControl(Components[i]).Height * 1.38);
        end
        else
        begin
          if pos('WebBrowser',TwinControl(components[i]).Name) > 0 then
          begin
            TWinControl(Components[i]).Width :=
Round(TWinControl(Components[i]).Width *
                                                        1.27);
```

```
            TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height *
                                                        1.31);
          end
          else
          begin
            if pos('Label',TwinControl(components[i]).Name) > 0 then
            begin { Redefine os componentes em proporção ao original }
              TLabel(Components[i]).Width :=
Canvas.TextWidth(TLabel(Components[i]).Caption);
              TLabel(Components[i]).Height :=
                                Canvas.TextHeight(TLabel(Components[i]).Caption);
            end
            else
            begin
              if pos('ScrollBox',TwinControl(components[i]).Name) > 0 then
              begin
                TWinControl(Components[i]).Width :=
Round(TWinControl(Components[i]).Width *
                                                          1.28);
                TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height *
                                                    1.31);
              end
              else
              begin
                TWinControl(Components[i]).Width :=
Round(TWinControl(Components[i]).Width *
                                                  1.29);
                TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height *
                                                  1.31);
              end
            end
```

```
          end
        end
      end
      else
      begin
        TWinControl(Components[i]).Width := FPrincipal.Width - 10;
        TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height *
                                                1.20);
        statusbar1.Panels.Items[0].Width := round(progressbar1.Left * 1.31);
        statusbar1.Panels.Items[1].Width := progressbar1.width;
        StatusBar1.Font.Size := iniconf.ReadInteger('Fonte','Tamanho',8);
        StatusBar1.Font.Name := iniconf.ReadString('Fonte','Tipo','Verdana');
      end;
    end;
  end;
 end;

 if (tela.Width = 1280) and (tela.Height = 1024) then
 begin
   for i := 0 to ComponentCount -1 do
   begin { Varre todos os componentes do form que possam ser redefinidos
(classe
           TWinControl) }
     if (pos('SpeedButton',twincontrol(components[i]).name) > 0) or
       (pos('Speed',twincontrol(components[i]).name) > 0) then
     begin
       if (pos('DescerSpeedButton',twincontrol(components[i]).name) > 0) or
         (pos('SpeedButton7',twincontrol(components[i]).name) > 0) then
       begin
         TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.60);
         TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.79);
```

```
        end
        else
        begin
          TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.61);
          TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.61);
        end;
      end
      else
        if pos('Timer',twincontrol(components[i]).name) = 0 then
          if pos('WebBrowser',twincontrol(components[i]).name) = 0 then
            if pos('Progress',twincontrol(components[i]).name) > 0 then
            begin
              TWinControl(Components[i]).Left := StatusBar1.Panels.Items[0].Width
+ 1;
              TWinControl(Components[i]).Top := StatusBar1.Top + 2;
            end
            else
            begin
              TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.61);
              TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.80);
            end
          else
          begin
            TWinControl(Components[i]).Left :=
Round(TWinControl(Components[i]).Left * 1.61);
            TWinControl(Components[i]).Top :=
Round(TWinControl(Components[i]).Top * 1.63);
          end;
```

```
    if (Components[i] is TWinControl) or (Components[i] is TbuttonControl)or
(Components[i] is
       TGraphicControl) then
      if (twincontrol(components[i]).name <> 'StatusBar1') then
       if (twincontrol(components[i]).name <> 'ProgressBar1') then
        if (twincontrol(components[i]).name <> 'WebBrowser1') then
        begin
          TWinControl(Components[i]).Width :=
Round(TWinControl(Components[i]).Width *
                                                          1.66);
          TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height *
                                                  1.86);
        end
        else
        begin
          TWinControl(Components[i]).Width :=
Round(TWinControl(Components[i]).Width *
                                                 1.59);
          TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height *
                                                  1.84);
        end
       else
       begin
        TWinControl(Components[i]).Width := statusbar1.Panels.Items[1].Width;
        TWinControl(Components[i]).Height := statusbar1.Height - 2;
       end
      else
      begin
       TWinControl(Components[i]).Width := FPrincipal.Width - 10;
       TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height *
                                               1.36);
```

```
        statusbar1.Panels.Items[0].Width := round(progressbar1.Left * 1.49);
        statusbar1.Panels.Items[1].Width := progressbar1.width;
        statusbar1.Font.Size := iniconf.ReadInteger('Fonte','Tamanho',8);
        statusbar1.Font.Name := iniconf.ReadString('Fonte','Tipo','Verdana');
      end
    else
    begin
      if Components[i] is TLabel then
      begin { Redefine os componentes em proporção ao original }
        TLabel(Components[i]).Width :=
Canvas.TextWidth(TLabel(Components[i]).Caption); //
         Round(TWinControl(Components[i]).Width * (Screen.Width / iWidth));
        TLabel(Components[i]).Height :=
Canvas.TextHeight(TLabel(Components[i]).Caption); //
         Round(TWinControl(Components[i]).Height * (Screen.Height / iHeight));
        TLabel(Components[i]).Font.Size :=
iniconf.ReadInteger('Fonte','Tamanho',8);
        TLabel(Components[i]).Font.Name :=
iniconf.ReadString('Fonte','Tipo','Verdana');
      end;
      if Components[i] is TImage then
      begin
        TWinControl(Components[i]).Width :=
Round(TWinControl(Components[i]).Width * 1.36);
        TWinControl(Components[i]).Height :=
Round(TWinControl(Components[i]).Height *
                                                1.36);
      end;
    end;
  end;
 end;

 label2.Left := Label1.Left + Label1.Width + 10;
```

```
  Label2.Font.Size := iniconf.ReadInteger('Fonte','Tamanho',8);
  Label2.Font.Name := iniconf.ReadString('Fonte','Tipo','Verdana');

  tela.ResetFonts;
end;

procedure TFprincipal.FormShow(Sender: TObject);
var
  iniconf : tinifile;
  Registro : TRegistry;
begin
  Registro := TRegistry.Create;
  try
  begin

Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
    if Registro.KeyExists('\Software\Microsoft\Windows\CurrentVersion\Explorer')
and
       registro.ValueExists('Logon User Name') then
    begin

Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
      if not Fileexists('.\' + registro.ReadString('Logon User Name') + '.ini') then
        copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('Logon User
Name') +
                    '.ini'),true);
        iniconf := tinifile.Create('.\' + registro.ReadString('Logon User Name') +
'.ini');
    end;
  end;
  except
    On E: Exception do iniconf := tinifile.Create('.\betaconf.ini');
```

```
  end;

  try
  begin
    Registro.RootKey := HKEY_LOCAL_MACHINE;
    Registro.OpenKey('\Network\Logon',false);
    if Registro.KeyExists('\Network\Logon') and registro.ValueExists('username')
then
    begin
      if not Fileexists('.\' + registro.ReadString('username') + '.ini') then
      copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('username') +
'.ini'),true);
      iniconf := tinifile.Create('.\' + registro.ReadString('username') + '.ini');
    end;
  end;
  except
    On E: Exception do iniconf := tinifile.Create('.\betaconf.ini');
  end;

  Tempo := iniconf.ReadInteger('Botoes', 'Velocidade', 2) * 100;
  AlternaBotoesTimer.Interval := Tempo;
  TecladoTimer.Interval := Tempo;

  GroupBox2.Caption :=
copy(iniconf.ReadString('Dicionario','DefaultNome',''),1,14);

  if not iniciado then
    AjustaTela(screen);

  iniciado := true;

  iniconf.Free;
end;
```

```
function TFprincipal.CaptureScreenRect( ARect: TRect ): TBitmap;
var
  ScreenDC: HDC;
begin
  Result := TBitmap.Create;
  with Result, ARect do
  begin
    Width := Right - Left;
    Height := Bottom - Top;
    ScreenDC := GetDC( 0 );
    try
      BitBlt( Canvas.Handle, 0, 0, Width, Height, ScreenDC, Left, Top, SRCCOPY
);
    finally
      ReleaseDC( 0, ScreenDC );
    end;
  end;
end;

procedure TFprincipal.TecladoTimerTimer(Sender: TObject);
var
  auxNumShape, auxNumButtonsShape : integer;
begin
  if Shape then
    case NumShape of
      1 : begin
            BetaLib.RestrictMouse(Groupbox1.Left + Shape1.Left + 1,
GroupBox1.top +
                                  Shape1.Top + 30, GroupBox1.Left + Shape1.Left + 2,
                                   GroupBox1.Top + Shape1.Top + 32);
            SmileSpeedButton.Visible := false;
            Shape1.Visible := true;
            NumShape := 2;
          end;
```

```
    2 : begin
        BetaLib.RestrictMouse(Groupbox1.Left + Shape2.Left + 1,
GroupBox1.top +
                              Shape2.Top + 30, GroupBox1.Left + Shape2.Left
+ 2,
                        GroupBox1.Top + Shape2.Top + 32);
        Shape1.Visible := false;
        Shape2.Visible := true;
        NumShape := 3;
       end;
    3 : begin
        BetaLib.RestrictMouse(Groupbox1.Left + Shape3.Left + 1,
GroupBox1.top +
                              Shape3.Top + 30, GroupBox1.Left + Shape3.Left
+ 2,
                        GroupBox1.Top + Shape3.Top + 32);
        Shape2.Visible := false;
        Shape3.Visible := true;
        NumShape := 4;
       end;
    4 : begin
        BetaLib.RestrictMouse(Groupbox1.Left + Shape4.Left + 1,
GroupBox1.top +
                               Shape4.Top + 30, GroupBox1.Left + Shape4.Left
+ 2,
                        GroupBox1.Top + Shape4.Top + 32);
        Shape3.Visible := false;
        Shape4.Visible := true;
        NumShape := 5;
       end;
    5 : begin
        BetaLib.RestrictMouse(Groupbox1.Left + Shape5.Left + 1,
GroupBox1.top +
```

```
                                    Shape5.Top + 30, GroupBox1.Left +
Shape5.Left + 2,
                            GroupBox1.Top + Shape5.Top + 32);
          Shape4.Visible := false;
          Shape5.Visible := true;
          NumShape := 6;
        end;
      6 : begin
          BetaLib.RestrictMouse(Groupbox1.Left + Shape6.Left + 1,
GroupBox1.top +
                                     Shape6.Top + 30, GroupBox1.Left + Shape6.Left
+ 2,
                            GroupBox1.Top + Shape6.Top + 32);
          Shape5.Visible := false;
          Shape6.Visible := true;
          NumShape := 7;
        end;
      7 : begin
          BetaLib.RestrictMouse(Groupbox1.Left + DesceDicSpeedButton.Left +
20,
                                    GroupBox1.top + DesceDicSpeedButton.Top +
50,
                                    GroupBox1.Left + DesceDicSpeedButton.Left +
20,
                                    GroupBox1.Top + DesceDicSpeedButton.Top +
50);
          Shape6.Visible := false;
          DesceDicSpeedButton.Visible := true;
          NumShape := 8;
        end;
      8 : begin
          BetaLib.RestrictMouse(Groupbox1.Left + SobeDicSpeedButton.Left +
20,
```

```
                                    GroupBox1.top + SobeDicSpeedButton.Top +
50,
                                    GroupBox1.Left + SobeDicSpeedButton.Left +
20,
                            GroupBox1.Top + SobeDicSpeedButton.Top + 50);
          SobeDicSpeedButton.Visible := true;
          DesceDicSpeedButton.Visible := false;
          NumShape := 9;
        end;
      9 : begin
          BetaLib.RestrictMouse(Groupbox1.Left + SmileSpeedButton.Left + 20,
GroupBox1.top +
                                    SmileSpeedButton.Top + 50, GroupBox1.Left +
                                    SmileSpeedButton.Left + 20, GroupBox1.Top +
                                    SmileSpeedButton.Top + 50);
          SobeDicSpeedButton.Visible := false;
          SmileSpeedButton.Visible := true;
          NumShape := 1;
        end;
    end;

  if ActiveButton then
  begin
    if NumShape = 1 then
      auxNumShape := 6
    else
      auxNumShape := NumShape - 1;

    case auxNumShape of
      1 : begin
          if NumButtonsShape <= 6 then
          begin
            ButtonsShape1[NumButtonsShape].Visible := true;
            if NumButtonsShape-1 <> 0 then
```

```
            ButtonsShape1[NumButtonsShape-1].Visible := false
          else
            ButtonsShape1[6].Visible := false;
          BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape1[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape1[NumButtonsShape].Top
                                  + 40, GroupBox1.Left +
ButtonsShape1[NumButtonsShape].Left
                                  + 20, GroupBox1.Top +
ButtonsShape1[NumButtonsShape].Top
                                  + 40);
          NumButtonsShape := NumButtonsShape + 1;
        end
        else
        begin
          NumButtonsShape := 1;
          ButtonsShape1[NumButtonsShape].Visible := true;
          ButtonsShape1[6].Visible := false;
          BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape1[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape1[NumButtonsShape].Top +
                                  40, GroupBox1.Left +
ButtonsShape1[NumButtonsShape].Left +
                                          20, GroupBox1.Top +
ButtonsShape1[NumButtonsShape].Top +
                                  40);
          NumButtonsShape := NumButtonsShape + 1;
        end;
      end;
    2 : begin
        if NumButtonsShape <= 10 then
        begin
```

```
            ButtonsShape2[NumButtonsShape].Visible := true;
            if NumButtonsShape-1 <> 0 then
              ButtonsShape2[NumButtonsShape-1].Visible := false
            else
              ButtonsShape2[10].Visible := false;
            BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape2[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape2[NumButtonsShape].Top +
                                  40, GroupBox1.Left +
ButtonsShape2[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape2[NumButtonsShape].Top +
                                  40);
            NumButtonsShape := NumButtonsShape + 1;
          end
          else
          begin
            NumButtonsShape := 1;
            ButtonsShape2[NumButtonsShape].Visible := true;
            ButtonsShape2[10].Visible := false;
            BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape2[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape2[NumButtonsShape].Top +
                                  40, GroupBox1.Left +
ButtonsShape2[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape2[NumButtonsShape].Top +
                                  40);
            NumButtonsShape := NumButtonsShape + 1;
          end;
        end;
    3 : begin
```

```
          if NumButtonsShape <= 10 then
          begin
            ButtonsShape3[NumButtonsShape].Visible := true;
            if NumButtonsShape-1 <> 0 then
              ButtonsShape3[NumButtonsShape-1].Visible := false
            else
              ButtonsShape3[10].Visible := false;
            BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape3[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape3[NumButtonsShape].Top +
                                  40, GroupBox1.Left +
ButtonsShape3[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape3[NumButtonsShape].Top +
                                 40);
            NumButtonsShape := NumButtonsShape + 1;
          end
          else
          begin
            NumButtonsShape := 1;
            ButtonsShape3[NumButtonsShape].Visible := true;
            ButtonsShape3[10].Visible := false;
            BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape3[NumButtonsShape].Left +
                                   20, GroupBox1.Top +
ButtonsShape3[NumButtonsShape].Top
                                  + 40, GroupBox1.Left +
ButtonsShape3[NumButtonsShape].Left
                                  + 20, GroupBox1.Top +
ButtonsShape3[NumButtonsShape].Top
                                  + 40);
            NumButtonsShape := NumButtonsShape + 1;
          end;
```

```
        end;
      4 : begin
          if NumButtonsShape <= 9 then
          begin
            ButtonsShape4[NumButtonsShape].Visible := true;
            if NumButtonsShape-1 <> 0 then
              ButtonsShape4[NumButtonsShape-1].Visible := false
            else
              ButtonsShape4[9].Visible := false;
            BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape4[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape4[NumButtonsShape].Top +
                                  40, GroupBox1.Left +
ButtonsShape4[NumButtonsShape].Left +
                                  20, GroupBox1.Top +
ButtonsShape4[NumButtonsShape].Top +
                                  40);
            NumButtonsShape := NumButtonsShape + 1;
          end
          else
          begin
            NumButtonsShape := 1;
            ButtonsShape4[NumButtonsShape].Visible := true;
            ButtonsShape4[9].Visible := false;
            BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape4[NumButtonsShape].Left +
                                   20, GroupBox1.Top +
ButtonsShape4[NumButtonsShape].Top
                                  + 40, GroupBox1.Left +
ButtonsShape4[NumButtonsShape].Left
                                  + 20, GroupBox1.Top +
ButtonsShape4[NumButtonsShape].Top
                                  + 40);
```

```
          NumButtonsShape := NumButtonsShape + 1;
        end;
      end;
    5 : begin
        if NumButtonsShape <= 9 then
        begin
          ButtonsShape5[NumButtonsShape].Visible := true;
          if NumButtonsShape-1 <> 0 then
            ButtonsShape5[NumButtonsShape-1].Visible := false
          else
            ButtonsShape5[9].Visible := false;
          BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape5[NumButtonsShape].Left +
                                20, GroupBox1.Top +
ButtonsShape5[NumButtonsShape].Top
                                + 40, GroupBox1.Left +
                                ButtonsShape5[NumButtonsShape].Left + 20,
GroupBox1.Top
                                + ButtonsShape5[NumButtonsShape].Top +
40);
          NumButtonsShape := NumButtonsShape + 1;
        end
        else
        begin
          NumButtonsShape := 1;
          ButtonsShape5[NumButtonsShape].Visible := true;
          ButtonsShape5[9].Visible := false;
          BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape5[NumButtonsShape].Left +
                                 20, GroupBox1.Top +
ButtonsShape5[NumButtonsShape].Top
                                + 40, GroupBox1.Left +
                                ButtonsShape5[NumButtonsShape].Left + 20,
GroupBox1.Top
```

```
                                    + ButtonsShape5[NumButtonsShape].Top +
40);
          NumButtonsShape := NumButtonsShape + 1;
         end;
        end;
     6 : begin
         if NumButtonsShape <= 3 then
         begin
          ButtonsShape6[NumButtonsShape].Visible := true;
          if NumButtonsShape-1 <> 0 then
           ButtonsShape6[NumButtonsShape-1].Visible := false
          else
           ButtonsShape6[3].Visible := false;
          BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape6[NumButtonsShape].Left +
                                    20, GroupBox1.Top +
ButtonsShape6[NumButtonsShape].Top +
                                    40, GroupBox1.Left +
ButtonsShape6[NumButtonsShape].Left +
                                    20, GroupBox1.Top +
ButtonsShape6[NumButtonsShape].Top +
                                    40);
          NumButtonsShape := NumButtonsShape + 1;
         end
         else
         begin
          NumButtonsShape := 1;
          ButtonsShape6[NumButtonsShape].Visible := true;
          ButtonsShape6[3].Visible := false;
          BetaLib.RestrictMouse(GroupBox1.Left +
ButtonsShape6[NumButtonsShape].Left +
                                     20, GroupBox1.Top +
ButtonsShape6[NumButtonsShape].Top
```

```
                                        + 40, GroupBox1.Left +
ButtonsShape6[NumButtonsShape].Left
                                        + 20, GroupBox1.Top +
ButtonsShape6[NumButtonsShape].Top
                                        + 40);
            NumButtonsShape := NumButtonsShape + 1;
          end;
        end;
    end;
  end;
end;

procedure TFprincipal.Shape1MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := true;
  shape := false;
  NumButtonsShape := 1;
end;

procedure TFprincipal.Shape2MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := true;
  shape := false;
  NumButtonsShape := 1;
end;

procedure TFprincipal.Shape3MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := true;
  shape := false;
  NumButtonsShape := 1;
```

```
end;

procedure TFprincipal.Shape4MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := true;
  shape := false;
  NumButtonsShape := 1;
end;

procedure TFprincipal.Shape5MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := true;
  shape := false;
  NumButtonsShape := 1;
end;

procedure TFprincipal.Shape6MouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := true;
  shape := false;
  NumButtonsShape := 1;
end;

procedure TFPrincipal.LocalizaUrl(URL : string);
var
  Tamanho : integer;
begin
  Tamanho := length(URL);
  Texto := URL;
  MD.URLS.AdsSeek(Texto,stSOFT);
  if pos(URL,MD.URLSURL.Value) = 1 then
```

```
  begin
    URLEdit.Text := MD.URLSURL.Value;
    URLEdit.SetFocus;
    URLEdit.SelStart := 0;
    URLEdit.SelLength := Tamanho;
  end
  else
  begin
    URLEdit.Text := Texto;
    URLEdit.SetFocus;
    URLEdit.SelStart := 0;
    URLEdit.SelLength := length(URLEdit.Text);
  end;
end;

procedure TFprincipal.URLSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  i : integer;
begin
  BetaLib.CriaLista;
  BetaLib.RestrictMouse(URLSpeedButton.Left + URLSpeedButton.Width + 5,
                        URLSpeedButton.Top + URLSpeedButton.height,
                        URLSpeedButton.Left + URLSpeedButton.Width + 5,
                        URLSpeedButton.Top + URLSpeedButton.height);
  AlternaBotoesTimer.Enabled := false;
  ScrollBox1.Top := GroupBox1.Top + GroupBox1.Height + 20;
  ScrollBox1.Height := StatusBar1.Top - ScrollBox1.Top;

  BetaLib.delay(iniconf.ReadInteger('Botoes','Velocidade',10)*100);
  TecladoTimer.Enabled := true;

  NumShape := 1;
  NumButtonsShape := 1;
```

```
  URLEdit.Text := '';

  IndicePalavra := 1;

  for i:=1 to 20 do
    PalavraDgt[i] := '';
end;

procedure TFprincipal.VoltarSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  try
    webbrowser1.GoBack;
  except
    on EOleSysError do speedbutton1.Enabled := false;
  end;
  speedbutton2.Enabled := true;
end;

procedure TFprincipal.AvancarSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  try
    webbrowser1.GoForward;
  except
    on EOleSysError do speedbutton2.Enabled := false;
  end;
  speedbutton2.Enabled := true;
end;

procedure TFprincipal.AtualizarSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
```

```
  webbrowser1.Refresh2;
end;

procedure TFprincipal.ParaSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  webbrowser1.Stop;
end;

procedure TFprincipal.FecharSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  AlternaBotoesTimer.Enabled := false;
  Close;
end;

procedure TFprincipal.wwwSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + wwwSpeedButton.Caption;

  ActiveButton := false;
  Shape := true;

  auxURL := auxURL + wwwSpeedButton.Caption;

  wwwSpeedButton.Visible := false;

  LocalizaUrl(auxURL);

  SelDic := false;

  DicListBox.Items.Clear;
end;
```

```
procedure TFprincipal.ftpSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + ftpSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  ftpSpeedButton.Visible := false;

  auxURL := auxURL + ftpSpeedButton.Caption;

  LocalizaUrl(auxURL);

  SelDic := false;

  DicListBox.Items.Clear;
end;

procedure TFprincipal.httpSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + httpSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  httpSpeedButton.Visible := false;

  auxURL := auxURL + httpSpeedButton.Caption;

  LocalizaUrl(auxURL);

  SelDic := false;

  DicListBox.Items.Clear;
end;
```

```
procedure TFprincipal.comSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + comSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  comSpeedButton.Visible := false;

  auxURL := auxURL + comSpeedButton.Caption;

  LocalizaUrl(auxURL);

  SelDic := false;

  DicListBox.Items.Clear;
end;

procedure TFprincipal.brSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + brSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  BrSpeedButton.Visible := false;

  auxURL := auxURL + brSpeedButton.Caption;

  LocalizaUrl(auxURL);

  SelDic := false;

  DicListBox.Items.Clear;
end;
```

```
procedure TFprincipal.Voltar1SpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := false;
  Shape := true;
  Voltar1SpeedButton.Visible := false;
end;

procedure TFprincipal.aSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := aSpeedButton.visible;
  ButtonTask(aSpeedButton.caption,btnvisible);
  aSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.bSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := bSpeedButton.visible;
  ButtonTask(bSpeedButton.caption,btnvisible);
  bSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.cSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
```

```
  btnvisible := cSpeedButton.visible;
  ButtonTask(cSpeedButton.caption,btnvisible);
  cSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.dSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := dSpeedButton.visible;
  ButtonTask(dSpeedButton.caption,btnvisible);
  dSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.eSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := eSpeedButton.visible;
  ButtonTask(eSpeedButton.caption,btnvisible);
  eSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.fSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := fSpeedButton.visible;
  ButtonTask(fSpeedButton.caption,btnvisible);
  fSpeedButton.Visible := btnVisible;
end;
```

```
procedure TFprincipal.gSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := gSpeedButton.visible;
  ButtonTask(gSpeedButton.caption,btnvisible);
  gSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.hSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := hSpeedButton.visible;
  ButtonTask(hSpeedButton.caption,btnvisible);
  hSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.iSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := iSpeedButton.visible;
  ButtonTask(iSpeedButton.caption,btnvisible);
  iSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.Voltar2SpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
```

```
  ActiveButton := false;
  Shape := true;
  Voltar2SpeedButton.Visible := false;
end;

procedure TFprincipal.jSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := jSpeedButton.visible;
  ButtonTask(jSpeedButton.caption,btnvisible);
  jSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.kSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := kSpeedButton.visible;
  ButtonTask(kSpeedButton.caption,btnvisible);
  kSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.lSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := lSpeedButton.visible;
  ButtonTask(lSpeedButton.caption,btnvisible);
  lSpeedButton.Visible := btnVisible;
end;
```

```
procedure TFprincipal.mSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := mSpeedButton.visible;
  ButtonTask(mSpeedButton.caption,btnvisible);
  mSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.nSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := nSpeedButton.visible;
  ButtonTask(nSpeedButton.caption,btnvisible);
  nSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.oSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := oSpeedButton.visible;
  ButtonTask(oSpeedButton.caption,btnvisible);
  oSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.pSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
```

```
  btnVisible : boolean;
begin
  btnvisible := pSpeedButton.visible;
  ButtonTask(pSpeedButton.caption,btnvisible);
  pSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.qSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := qSpeedButton.visible;
  ButtonTask(qSpeedButton.caption,btnvisible);
  qSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.rSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := rSpeedButton.visible;
  ButtonTask(rSpeedButton.caption,btnvisible);
  rSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.Voltar3SpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := false;
  Shape := true;
  Voltar3SpeedButton.Visible := false;
end;
```

```
procedure TFprincipal.sSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := sSpeedButton.visible;
  ButtonTask(sSpeedButton.caption,btnvisible);
  sSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.tSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := tSpeedButton.visible;
  ButtonTask(tSpeedButton.caption,btnvisible);
  tSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.uSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := uSpeedButton.visible;
  ButtonTask(uSpeedButton.caption,btnvisible);
  uSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.vSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
```

```
  btnVisible : boolean;
begin
  btnvisible := vSpeedButton.visible;
  ButtonTask(vSpeedButton.caption,btnvisible);
  vSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.xSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := xSpeedButton.visible;
  ButtonTask(xSpeedButton.caption,btnvisible);
  xSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.ySpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := ySpeedButton.visible;
  ButtonTask(ySpeedButton.caption,btnvisible);
  ySpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.wSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := wSpeedButton.visible;
  ButtonTask(wSpeedButton.caption,btnvisible);
```

```
  wSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.zSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  btnVisible : boolean;
begin
  btnvisible := zSpeedButton.visible;
  ButtonTask(zSpeedButton.caption,btnvisible);
  zSpeedButton.Visible := btnVisible;
end;

procedure TFprincipal.Voltar4SpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := false;
  Shape := true;
  Voltar4SpeedButton.Visible := false;
end;

procedure TFprincipal.backSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  auxPalavra : String;
begin
  TecladoTimer.Enabled := false;

  URLEdit.Text := copy(URLEdit.Text,1,length(URLEdit.Text)-1);

  if copy(URLEdit.Text,pos('.com',URLEdit.Text),length(URLEdit.Text)-
pos('.com',URLEdit.Text))
     = '.com' then
  begin
```

```
    URLEdit.Text := copy(URLEdit.Text,1,length(URLEdit.Text)-
length('.com')+1);
    auxURL := copy(auxURL,1,length(auxURL)-length('.com') + 1);
  end
  else
    if copy(URLEdit.Text,pos('.br',URLEdit.Text),length(URLEdit.Text)-
pos('.br',URLEdit.Text)) =
       '.br' then
    begin
      URLEdit.Text := copy(URLEdit.Text,1,length(URLEdit.Text)-length('.br')+1);
      auxURL := copy(auxURL,1,length(auxURL)-length('.br') + 1);
    end
    else
    begin
      if (copy(URLEdit.Text,pos('www.',URLEdit.Text), URLEdit.SelLength) =
'www.')
         or (copy(URLEdit.Text,pos('ftp.',URLEdit.Text), URLEdit.SelLength) =
'ftp.')
         or (copy(URLEdit.Text,pos('http://',URLEdit.Text), URLEdit.SelLength) =
'http://')then
      begin
        URLEdit.Text := copy(URLEdit.Text,1,URLEdit.SelLength);
        auxURL := copy(auxURL,1,length(auxURL));
      end
      else
      begin
        PalavraDgt[IndicePalavra] :=

copy(PalavraDgt[IndicePalavra],1,length(PalavraDgt[IndicePalavra])-1);
        auxURL := copy(auxURL,1,length(auxURL)-1);
      end;
    end;

  LocalizaUrl(auxURL);
```

```
  if SelDic then
  begin
    auxPalavra := copy(URLEdit.SelText,pos(PalavraDgt[IndicePalavra-
                        1],URLEdit.SelText),length(URLEdit.SelText));
    if auxPalavra <> PalavraDgt[IndicePalavra-1] then
      PalavraDgt[IndicePalavra-1] := auxPalavra;
    IndicePalavra := IndicePalavra - 1;
  end;

  if (PalavraDgt[IndicePalavra] = '') and (IndicePalavra > 1) then
  begin
    IndicePalavra := IndicePalavra - 1;
    auxPalavra :=
copy(URLEdit.SelText,pos(PalavraDgt[IndicePalavra],URLEdit.SelText),length(
URLEdit.SelText
        ));
    if auxPalavra <> PalavraDgt[IndicePalavra] then
      PalavraDgt[IndicePalavra] := auxPalavra;
  end;

  SelDic := false;

  CarregaListBox(BetaLib.ConsultaDicionario(PalavraDgt[IndicePalavra],Lista));

  BetaLib.delay(iniconf.ReadInteger('Botoes','Velocidade',10)*100);

  Shape := true;
  ActiveButton := false;

  TecladoTimer.Enabled := true;
  backSpeedButton.Visible := false;
end;
```

```
procedure TFprincipal.pontoSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + pontoSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  pontoSpeedButton.Visible := false;

  auxURL := auxURL + pontoSpeedButton.Caption;

  LocalizaUrl(auxURL);

  IndicePalavra := IndicePalavra + 1;

  DicListBox.Items.Clear;

  SelDic := false;
end;

procedure TFprincipal.barraSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + barraSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  barraSpeedButton.Visible := false;

  auxURL := auxURL + barraSpeedButton.Caption;

  LocalizaUrl(auxURL);

  IndicePalavra := IndicePalavra + 1;

  DicListBox.Items.Clear;
```

```
  SelDic := false;
end;

procedure TFprincipal.hifenSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + hifenSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  hifenSpeedButton.Visible := false;

  auxURL := auxURL + hifenSpeedButton.Caption;

  LocalizaUrl(auxURL);

  IndicePalavra := IndicePalavra + 1;

  DicListBox.Items.Clear;

  SelDic := false;
end;

procedure TFprincipal.underSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + underSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  underSpeedButton.Visible := false;

  auxURL := auxURL + underSpeedButton.Caption;

  LocalizaUrl(auxURL);
```

```
  IndicePalavra := IndicePalavra + 1;

  DicListBox.Items.Clear;

  SelDic := false;
end;

procedure TFprincipal.maisSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + maisSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  maisSpeedButton.Visible := false;

  auxURL := auxURL + maisSpeedButton.Caption;

  LocalizaUrl(auxURL);

  IndicePalavra := IndicePalavra + 1;

  DicListBox.Items.Clear;

  SelDic := false;
end;

procedure TFprincipal.virgulaSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + virgulaSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  virgulaSpeedButton.Visible := false;
```

```
  auxURL := auxURL + virgulaSpeedButton.Caption;

  LocalizaUrl(auxURL);

  IndicePalavra := IndicePalavra + 1;

  DicListBox.Items.Clear;

  SelDic := false;
end;

procedure TFprincipal.doispontosSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  URLEdit.Text := URLEdit.Text + doispontosSpeedButton.Caption;
  ActiveButton := false;
  Shape := true;
  doispontosSpeedButton.Visible := false;

  auxURL := auxURL + doispontosSpeedButton.Caption;

  LocalizaUrl(auxURL);

  IndicePalavra := IndicePalavra + 1;

  DicListBox.Items.Clear;

  SelDic := false;
end;

procedure TFprincipal.Voltar5SpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
```

```
  ActiveButton := false;
  Shape := true;
  Voltar5SpeedButton.Visible := false;
end;

procedure TFprincipal.navegarSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin

BetaLib.RestrictMouse(FPrincipal.Left+10,FPrincipal.Top+40,FPrincipal.Left+10
,FPrincipal.Top
                                +40);

  Label2.Caption := URLEdit.Text;

  TecladoTimer.Enabled := false;

  ScrollBox1.Top := SubirSpeedButton.Top;
  ScrollBox1.Height := StatusBar1.Top - ScrollBox1.Top;

  AlternaBotoesTimer.Enabled := true;

  Shape := true;
  ActiveButton := false;

  Shape6.Visible := false;
  navegarSpeedButton.Visible := false;

  auxURL := '';
  SelDic := false;

  Navega;
end;
```

```
procedure TFprincipal.cancelarSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin

BetaLib.RestrictMouse(FPrincipal.Left+10,FPrincipal.Top+40,FPrincipal.Left+10
,FPrincipal.Top
                              +40);
  BetaLib.DestroiLista;
  TecladoTimer.Enabled := false;

  ScrollBox1.Top := SubirSpeedButton.Top;
  ScrollBox1.Height := StatusBar1.Top - ScrollBox1.Top;

  AlternaBotoesTimer.Enabled := true;

  Shape := true;
  ActiveButton := false;

  Shape6.Visible := false;
  cancelarSpeedButton.Visible := false;

  auxURL := '';
  SelDic := false;
end;

procedure TFprincipal.Voltar6SpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  ActiveButton := false;
  Shape := true;
  Voltar6SpeedButton.Visible := false;
end;

procedure TFprincipal.ScrollBox1MouseDown(Sender: TObject;
```

```
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  AlternaBotoesTimer.Enabled := false;
  BetaLib.delay(AlternaBotoesTimer.Interval);
  AlternaBotoesTimer.Enabled := true;
end;

procedure TFprincipal.DescerSpeedButtonMouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  BetaLib.restrictmouse(ScrollBox1.Left + (ScrollBox1.Width -
10),ScrollBox1.Top +
                        ScrollBox1.height + 10,ScrollBox1.Left +
ScrollBox1.Width –
                        10,ScrollBox1.Top + ScrollBox1.height + 10);
  BetaLib.SimulaClick(ScrollBox1.Left + (ScrollBox1.Width - 10), ScrollBox1.Top
+
                      ScrollBox1.height + 10);
  BetaLib.restrictmouse(DescerSpeedButton.Left + 10,DescerSpeedButton.Top
+
                        30,DescerSpeedButton.Left +
10,DescerSpeedButton.Top + 30);

  AlternaBotoesTimer.Enabled := false;
  BetaLib.delay(iniconf.ReadInteger('Botoes','Velocidade',10)*100);
  AlternaBotoesTimer.Enabled := true;
end;

procedure TFprincipal.SubirSpeedButtonMouseDown(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
begin
  BetaLib.restrictmouse(ScrollBox1.Left + (ScrollBox1.Width -
10),ScrollBox1.Top +
```

```
                                30,ScrollBox1.Left + ScrollBox1.Width -
10,ScrollBox1.Top + 30);
  BetaLib.SimulaClick(ScrollBox1.Left + (ScrollBox1.Width - 10), ScrollBox1.Top
+ 30);
  BetaLib.restrictmouse(SubirSpeedButton.Left + 10,SubirSpeedButton.Top +
                                30,SubirSpeedButton.Left + 10,SubirSpeedButton.Top +
30);

  AlternaBotoesTimer.Enabled := false;
  BetaLib.delay(iniconf.ReadInteger('Botoes','Velocidade',10)*100);
  AlternaBotoesTimer.Enabled := true;
end;

procedure TFPrincipal.CarregaListBox(Lista : TStrings);
begin
  try
    if Lista.Count > 0 then
    begin
      DicListBox.Clear;
      DicListBox.Items.Capacity := lista.count;
      DicListBox.Items := Lista;
      DicListBox.Selected[0] := true;
    end;
  except
    on E:exception do
    begin
      DicListBox.Selected[0] := true;
    end;
  end;

  if DicListBox.Items.Capacity = 0 then
  begin
    IndicePalavra := IndicePalavra + 1;
  end;
```

```
end;

procedure TFprincipal.smileSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  i : integer;
begin
  for i:=0 to DicListBox.Items.Capacity - 1 do
    if DicListBox.Selected[i] then
      break;

  URLEdit.Text := URLEdit.Text +
copy(DicListBox.Items.Strings[i],length(PalavraDgt[IndicePalavra]) +
       1,length(DicListBox.Items.Strings[i]));

  auxURL := auxURL +
copy(DicListBox.Items.Strings[i],length(PalavraDgt[IndicePalavra]) +
                1,length(DicListBox.Items.Strings[i]));

  LocalizaUrl(auxURL);

  if DicListBox.Items.Capacity > 0 then
    IndicePalavra := IndicePalavra + 1;

  SelDic := true;
end;

procedure TFprincipal.SobeDicSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  i : integer;
begin
  TecladoTimer.Enabled := false;
```

```
  if DicListBox.Items.Capacity > 0 then
  begin
    for i:=0 to DicListBox.Items.Capacity - 1 do
      if DicListBox.Selected[i] then
        break;
    if i <> 0 then
      DicListBox.Selected[i - 1] := true;
  end;

  BetaLib.delay(iniconf.ReadInteger('Botoes','Velocidade',10)*100);
  TecladoTimer.Enabled := true;
end;

procedure TFprincipal.DesceDicSpeedButtonMouseUp(Sender: TObject;
  Button: TMouseButton; Shift: TShiftState; X, Y: Integer);
var
  i : integer;
begin
  TecladoTimer.Enabled := false;

  if DicListBox.Items.Capacity > 0 then
  begin
    for i:=0 to DicListBox.Items.Capacity - 1 do
      if DicListBox.Selected[i] then
        break;
    if DicListBox.Items.Capacity - 1  > i then
      DicListBox.Selected[i + 1] := true;
  end;

  BetaLib.delay(iniconf.ReadInteger('Botoes','Velocidade',10)*100);
  TecladoTimer.Enabled := true;
end;

procedure TFPrincipal.ButtonTask(caption : TCaption; var visible : boolean);
```

```
begin
  TecladoTimer.Enabled := false; //para que o teclado nao corra sem que a lista
de palavras
                                        tenha sido carregada.

  URLEdit.Text := URLEdit.Text + Caption;
  ActiveButton := false;
  Shape := true;
  Visible := false;

  auxURL := auxURL + Caption;

  LocalizaUrl(auxURL);

  PalavraDgt[IndicePalavra] := PalavraDgt[IndicePalavra] + Caption;

  auxLista := BetaLib.ConsultaDicionario(PalavraDgt[IndicePalavra], Lista);

  while auxLista.Count = 0 do
  begin
    PalavraDgt[IndicePalavra + 1] :=
copy(PalavraDgt[IndicePalavra],length(PalavraDgt[IndicePalavra]),1);
    PalavraDgt[IndicePalavra] :=
              copy(PalavraDgt[IndicePalavra],1,length(PalavraDgt[IndicePalavra])-
1);
    IndicePalavra := IndicePalavra + 1;
    auxLista := BetaLib.ConsultaDicionario(PalavraDgt[IndicePalavra], Lista);
  end;

  CarregaListBox(auxLista);

  SelDic := false;

  TecladoTimer.Enabled := true; //Reabilita o teclado
```

```
end;

procedure TFprincipal.WebBrowser1StatusTextChange(Sender: TObject;
  const Text: WideString);
begin
  statusbar1.Panels.Items[0].Text := text;
end;

end.
```

3 - Código da tela de Configuração:

```
unit UConfigura;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls, ComCtrls, Spin, inifiles;

type
  TFconfigura = class(TForm)
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    PageControl1: TPageControl;
    TabSheet1: TTabSheet;
    TabSheet2: TTabSheet;
    MouseCheckBox: TCheckBox;
    TecladoCheckBox: TCheckBox;
    GroupBox1: TGroupBox;
    MouseImage: TImage;
```

```
  GroupBox2: TGroupBox;
  GroupBox3: TGroupBox;
  Label1: TLabel;
  DestaqueTrackBar: TTrackBar;
  Label2: TLabel;
  TabSheet3: TTabSheet;
  GroupBox4: TGroupBox;
  Label3: TLabel;
  Label4: TLabel;
  FontDialog1: TFontDialog;
  Button4: TButton;
  Label5: TLabel;
  Label6: TLabel;
  ComboBox1: TComboBox;
  Label7: TLabel;
  Label8: TLabel;
  Button5: TButton;
  CheckBox1: TCheckBox;
  Button6: TButton;
  procedure Button3Click(Sender: TObject);
  procedure MouseCheckBoxClick(Sender: TObject);
  procedure TecladoCheckBoxClick(Sender: TObject);
  procedure Button2Click(Sender: TObject);
  procedure FormShow(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  procedure DestaqueTrackBarChange(Sender: TObject);
  procedure Button4Click(Sender: TObject);
  procedure Button5Click(Sender: TObject);
  procedure ComboBox1Change(Sender: TObject);
  procedure CheckBox1Enter(Sender: TObject);
  procedure CheckBox1Click(Sender: TObject);
  procedure Button6Click(Sender: TObject);
private
  { Private declarations }
```

```
  public
    { Public declarations }
  end;

var
  Fconfigura: TFconfigura;

  DesabilitarTeclado, InverterMouse : boolean;

implementation

uses UImporta, Registry;

var
  iniconf : tinifile;

{$R *.DFM}

Procedure TFconfigura.Button3Click(Sender: TObject);
begin
  iniconf.Free;
  Close;
end;

procedure TFconfigura.MouseCheckBoxClick(Sender: TObject);
begin
  if not MouseCheckBox.Checked then
  begin
    SwapMouseButton(false);
    InverterMouse := false;
    MouseImage.Picture.LoadFromFile(getcurrentdir +
'\imagens\mouseesquerdo.bmp');
  end
  else
```

```
  begin
    SwapMouseButton(true);
    InverterMouse := true;
    MouseImage.Picture.LoadFromFile(getcurrentdir +
'\imagens\mousedireito.bmp');
  end;
  Button2.Enabled := true;
end;

procedure TFconfigura.TecladoCheckBoxClick(Sender: TObject);
var
  npos : pointer;
begin
  if TecladoCheckBox.Checked then
  begin
    SystemParametersInfo(SPI_SCREENSAVERRUNNING, word(True),
@nPos, 0);
    DesabilitarTeclado := true;
  end
  else
  begin
    SystemParametersInfo(SPI_SCREENSAVERRUNNING, word(False),
@nPos, 0);
    DesabilitarTeclado := false;
  end;
  Button2.Enabled := true;
end;

procedure TFconfigura.Button2Click(Sender: TObject);
begin
  iniconf.WriteInteger('Botoes', 'Velocidade', DestaqueTrackBar.Position);
  iniconf.WriteBool('Mouse', 'Inverter', InverterMouse);
  iniconf.WriteBool('Teclado', 'Desabilitar', DesabilitarTeclado);
```

```
  iniconf.WriteInteger('Fonte','Tamanho',FontDialog1.Font.Size);
  iniconf.WriteString('Fonte','Tipo',FontDialog1.Font.Name);

  Button2.Enabled := false;
end;

procedure TFconfigura.FormShow(Sender: TObject);
var
  Registro : TRegistry;
begin
  Registro := TRegistry.Create;
  try
  begin

Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
    if Registro.KeyExists('\Software\Microsoft\Windows\CurrentVersion\Explorer')
and
       registro.ValueExists('Logon User Name') then
    begin

Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
      if not Fileexists('.\' + registro.ReadString('Logon User Name') + '.ini') then
        copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('Logon User
Name') +
                    '.ini'),true);
        iniconf := tinifile.Create('.\' + registro.ReadString('Logon User Name') +
'.ini');
    end;
  end;
  except
    On E: Exception do iniconf := tinifile.Create('.\betaconf.ini');
  end;
```

```
  try
  begin
    Registro.RootKey := HKEY_LOCAL_MACHINE;
    Registro.OpenKey('\Network\Logon',false);
    if Registro.KeyExists('\Network\Logon') and registro.ValueExists('username')
then
    begin
      if not Fileexists('.\' + registro.ReadString('username') + '.ini') then
      copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('username') +
'.ini'),true);
      iniconf := tinifile.Create('.\' + registro.ReadString('username') + '.ini');
    end;
  end;
  except
    On E: Exception do iniconf := tinifile.Create('.\betaconf.ini');
  end;

  MouseCheckBox.Checked := iniconf.ReadBool('Mouse', 'Inverter', false);
  TecladoCheckBox.Checked := iniconf.ReadBool('Teclado', 'Desabilitar', false);
  DestaqueTrackBar.Position := iniconf.ReadInteger('Botoes', 'Velocidade', 2);
  label2.Caption := floattostr(DestaqueTrackBar.Position/10) + ' seg.';
  label5.Caption := inttostr(iniconf.ReadInteger('Fonte','Tamanho',8));
  Label6.Caption := iniconf.ReadString('Fonte','Tipo','Verdana');

  FontDialog1.Font.Name := iniconf.ReadString('Fonte','Tipo','Verdana');
  FontDialog1.Font.Size := iniconf.ReadInteger('Fonte','Tamanho',8);

  combobox1.Clear;
  ComboBox1.Items.Add(iniconf.ReadString('Dicionario','Nome0',''));
  ComboBox1.Items.Add(iniconf.ReadString('Dicionario','Nome1',''));
  ComboBox1.Items.Add(iniconf.ReadString('Dicionario','Nome2',''));
  ComboBox1.Items.Add(iniconf.ReadString('Dicionario','Nome3',''));
```

```
  Label8.Caption := 'Arquivo: ' + iniconf.ReadString('Dicionario','Default','');
  Combobox1.Text := iniconf.ReadString('Dicionario','DefaultNome','');

  CheckBox1.Checked := true;

  Button2.Enabled := false;
end;

procedure TFconfigura.Button1Click(Sender: TObject);
begin
  if iniconf.ReadString('Dicionario','DefaultNome','') = '' then
    showmessage('Não há Dicionário Padrão definido!! Selecione um dos
dicionários e clique em
                        "Dicionário Padrão".')
  else
  begin
    if Button2.Enabled then
      Button2.Click;
    iniconf.Free;
    close;
  end;
end;

procedure TFconfigura.DestaqueTrackBarChange(Sender: TObject);
begin
  Button2.Enabled := true;
  label2.Caption := floattostr(DestaqueTrackBar.Position/10) + ' seg.';
end;

procedure TFconfigura.Button4Click(Sender: TObject);
begin
  if FontDialog1.Execute then
  begin
     Label5.Caption := inttostr(FontDialog1.Font.Size);
```

```
    Label6.Caption := FontDialog1.Font.Name;

    Button2.Enabled := true;
  end;
end;

procedure TFconfigura.Button5Click(Sender: TObject);
var
  i : integer;
begin
  FImporta.ShowModal;
  for i:=0 to 3 do
    combobox1.Items.Strings[i] := iniconf.ReadString('Dicionario','Nome' +
inttostr(i), '');

  Label8.Caption := 'Arquivo: ' + iniconf.ReadString('Dicionario','Arquivo0', '');
  combobox1.Text := iniconf.ReadString('Dicionario','Nome0', '');
  button2.Enabled := true;
end;

procedure TFconfigura.ComboBox1Change(Sender: TObject);
begin
  if combobox1.ItemIndex = 0 then
    Label8.Caption := 'Arquivo: ' + iniconf.ReadString('Dicionario','Arquivo0','');
  if combobox1.ItemIndex = 1 then
    Label8.Caption := 'Arquivo: ' + iniconf.ReadString('Dicionario','Arquivo1','');
  if combobox1.ItemIndex = 2 then
    Label8.Caption := 'Arquivo: ' + iniconf.ReadString('Dicionario','Arquivo2','');
  if combobox1.ItemIndex = 3 then
    Label8.Caption := 'Arquivo: ' + iniconf.ReadString('Dicionario','Arquivo3','');

  if Combobox1.Text = iniconf.ReadString('Dicionario','DefaultNome','') then
    Checkbox1.Checked := true
  else
```

```
    Checkbox1.Checked := false;
  Button2.Enabled :=true;
end;

procedure TFconfigura.CheckBox1Enter(Sender: TObject);
begin
  if checkbox1.Checked then
  begin
    iniconf.WriteString('Dicionario','Default','');
    iniconf.WriteString('Dicionario','DefaultNome','');
  end
  else
  begin

iniconf.WriteString('Dicionario','Default',copy(label8.Caption,10,length(Label8.ca
ption) - 9));
    iniconf.WriteString('Dicionario','DefaultNome',Combobox1.Text);
  end;
end;

procedure TFconfigura.CheckBox1Click(Sender: TObject);
begin
  button1.SetFocus;
  button2.Enabled := true;
end;

procedure TFconfigura.Button6Click(Sender: TObject);
begin
  if combobox1.Text = iniconf.ReadString('Dicionario','DefaultNome','') then
  begin
    iniconf.WriteString('Dicionario','DefaultNome','');
    iniconf.WriteString('Dicionario','Default','');
    CheckBox1.Checked := false;
  end;
```

```
  iniconf.WriteString('Dicionario','Arquivo' + inttostr(combobox1.ItemIndex),'');
  iniconf.WriteString('Dicionario','Nome' + inttostr(combobox1.ItemIndex),'');
  iniconf.DeleteKey('Dicionario','Arquivo' + inttostr(combobox1.ItemIndex));
  iniconf.DeleteKey('Dicionario','Nome' + inttostr(combobox1.ItemIndex));
  combobox1.Items.Strings[combobox1.ItemIndex] := '';
  Combobox1.ItemIndex := 0;
  if combobox1.Items.Strings[0] =
iniconf.ReadString('Dicionario','DefaultNome','') then
    CheckBox1.Checked := true;
  Label8.Caption := 'Arquivo: ' + iniconf.ReadString('Dicionario','Arquivo0','');
  button2.Enabled := true;
end;

end.
```

4- Código da tela de Importação do Dicionário:

```
unit UImporta;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls;

type
  TFImporta = class(TForm)
    Button3: TButton;
    GroupBox1: TGroupBox;
```

```
    LabeledEdit1: TLabeledEdit;
    LabeledEdit2: TLabeledEdit;
    Button1: TButton;
    Button4: TButton;
    procedure Button1Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  FImporta: TFImporta;

implementation

uses UOpen, inifiles, Registry;

{$R *.dfm}

procedure TFImporta.Button1Click(Sender: TObject);
begin
  FProcura.ShowModal;
  LabeledEdit2.Text := FProcura.FileListBox1.FileName;
end;

procedure TFImporta.Button4Click(Sender: TObject);
var
  betaconf : tinifile;
  arquivo : TstringList;
  Registro : TRegistry;
begin
```

```
  arquivo := TstringList.Create;

  Registro := TRegistry.Create;
  try
  begin

Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
    if Registro.KeyExists('\Software\Microsoft\Windows\CurrentVersion\Explorer')
and
       registro.ValueExists('Logon User Name') then
    begin

Registro.OpenKey('\Software\Microsoft\Windows\CurrentVersion\Explorer',false
);
      if not Fileexists('.\' + registro.ReadString('Logon User Name') + '.ini') then
        copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('Logon User
Name') +
                 '.ini'),true);
        betaconf := tinifile.Create('.\' + registro.ReadString('Logon User Name') +
'.ini');
    end;
  end;
  except
    On E: Exception do betaconf := tinifile.Create('.\betaconf.ini');
  end;

  try
  begin
    Registro.RootKey := HKEY_LOCAL_MACHINE;
    Registro.OpenKey('\Network\Logon',false);
    if Registro.KeyExists('\Network\Logon') and registro.ValueExists('username')
then
    begin
```

```
    if not Fileexists('.\' + registro.ReadString('username') + '.ini') then
    copyfile(pchar('.\betaconf.ini'),pchar('.\' + registro.ReadString('username') +
'.ini'),true);
    betaconf := tinifile.Create('.\' + registro.ReadString('username') + '.ini');
   end;
  end;
  except
   On E: Exception do betaconf := tinifile.Create('.\betaconf.ini');
  end;

  if LabeledEdit1.Text <> '' then
  begin
   if fileexists(LabeledEdit2.Text) then
   begin
    if not betaconf.ValueExists('Dicionario','Arquivo0') then
    begin
     betaconf.WriteString('Dicionario','Arquivo0',LabeledEdit2.Text);
     betaconf.WriteString('Dicionario','Nome0',LabeledEdit1.Text);
     arquivo.LoadFromFile(LabeledEdit2.Text);
     arquivo.Sort;
     arquivo.SaveToFile(LabeledEdit2.Text);
    end
    else
     if not betaconf.ValueExists('Dicionario','Arquivo1') then
     begin
      betaconf.WriteString('Dicionario','Arquivo1',LabeledEdit2.Text);
      betaconf.WriteString('Dicionario','Nome1',LabeledEdit1.Text);
      arquivo.LoadFromFile(LabeledEdit2.Text);
      arquivo.Sort;
      arquivo.SaveToFile(LabeledEdit2.Text);
     end
     else
      if not betaconf.ValueExists('Dicionario','Arquivo2') then
      begin
```

```
        betaconf.WriteString('Dicionario','Arquivo2',LabeledEdit2.Text);
        betaconf.WriteString('Dicionario','Nome2',LabeledEdit1.Text);
        arquivo.LoadFromFile(LabeledEdit2.Text);
        arquivo.Sort;
        arquivo.SaveToFile(LabeledEdit2.Text);
       end
       else
        if not betaconf.ValueExists('Dicionario','Arquivo3') then
        begin
         betaconf.WriteString('Dicionario','Arquivo3',LabeledEdit2.Text);
         betaconf.WriteString('Dicionario','Nome3',LabeledEdit1.Text);
         arquivo.LoadFromFile(LabeledEdit2.Text);
         arquivo.Sort;
         arquivo.SaveToFile(LabeledEdit2.Text);
        end
        else
         showmessage('Você deve excluir um dos dicionários!');
   betaconf.Free;
   close;
  end
  else
  begin
   Beep;
   showmessage('Arquivo não encontrado!!');
   LabeledEdit2.SetFocus;
  end;
end
else
begin
  Beep;
  showmessage('Defina um nome para o dicionário!!');
  LabeledEdit1.SetFocus;
end;
```

```
end;

procedure TFImporta.Button3Click(Sender: TObject);
begin
  close;
end;

end.
```

5- Código da tela de Procura do Dicionário:

```
unit UOpen;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  StdCtrls, FileCtrl;

type
  TFProcura = class(TForm)
    DirectoryListBox1: TDirectoryListBox;
    DriveComboBox1: TDriveComboBox;
    Label1: TLabel;
    Button1: TButton;
    Button2: TButton;
    Label2: TLabel;
    FileListBox1: TFileListBox;
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure DirectoryListBox1Change(Sender: TObject);
    procedure FormCreate(Sender: TObject);
```

```
    procedure FormShow(Sender: TObject);
  private
    { Private declarations }
  public
    tipo : integer;
  end;

var
  FProcura: TFProcura;

implementation

{$R *.DFM}
(*$I-*)

procedure TFProcura.Button1Click(Sender: TObject);
begin
  Close;
end;

procedure TFProcura.Button2Click(Sender: TObject);
begin
  Label1.Caption := '';
  Close;
end;

procedure TFProcura.DirectoryListBox1Change(Sender: TObject);
begin
  //label1.caption:=directorylistbox1.Directory+'\';
end;

procedure TFProcura.FormCreate(Sender: TObject);
begin
  label1.caption:=directorylistbox1.Directory+'\';
```

```
end;

procedure TFProcura.FormShow(Sender: TObject);
begin
  drivecombobox1.Drive := directorylistbox1.Drive;
end;

end.
```

6- Código da biblioteca de funções:

```
unit UBetaLib;

interface

uses dialogs, sysutils, ExtCtrls, StdCtrls, forms, Controls, Windows, classes,
inifiles;

type
  tBetaLib = class
  function retiracaracteres(listadecaracteres : string; variavel : string) : string;
  function strzero (tamanho : integer; variavel : string) : string;
  procedure delay(tempo : integer);
  procedure alteraacabou(sera : boolean);
  function  devolvebranco (qtd : integer) : string;
  function  retirazeroesq (variavel : string) : string;
  function  dow(ddata : string) : integer;
  procedure mouseparacontrole(controle : tcontrol);
  function  mouseshowcursor(const show : boolean) : boolean;
  procedure RestrictMouse(left,top,rigth,bottom : integer);
  procedure UnRestrictMouse;
  function VerificaAcabou : boolean;
```

```
  procedure SimulaClick(x,y : integer);
  function ConsultaDicionario(palavra : string; Lista : TStringList) : Tstrings;
  procedure CriaLista;
  procedure DestroiLista;

public

end;

var
   Betalib : tBetaLib;
   acabou : boolean;
   arquivo : textfile;
   auxLista : TStringList;

implementation

uses auxlib, StrUtils;

type TReplaceFlags = set of (rfReplaceAll, rfIgnoreCase);

procedure tBetaLib.CriaLista;
begin
  auxLista := TStringList.Create;
end;

procedure tBetaLib.DestroiLista;
begin
  auxLista.Free;
end;

function tBetaLib.retiracaracteres(listadecaracteres : string; variavel : string) :
string;
var
```

```
  caracteres : array [1..10] of string;
  i, j : integer;
  aux : string;
  achou : boolean;
begin
  for i:=1 to length(listadecaracteres) do
    caracteres[i] := copy(listadecaracteres,i,1);

  for i:=1 to length(variavel) do
  begin
    for j:=1 to length(listadecaracteres) do
    begin
      if copy(variavel,i,1)=caracteres[j] then
        achou := true
    end;

    if not achou then
      aux := aux + copy(variavel,i,1)
    else
      achou := false;
  end;

  result := aux;
end;

function tBetaLib.strzero (tamanho : integer; variavel : string) : string;
begin
  while length(variavel) < tamanho do
    variavel := '0' + variavel;

  result := variavel;
end;

procedure tBetaLib.delay(tempo : integer);
```

```
begin
  auxBetaLib.Timer1.Interval := tempo;
  auxBetaLib.timer1.Enabled := true;

  alteraacabou(true);

  while verificaacabou do
    application.processmessages;
end;

procedure tBetaLib.alteraacabou(sera : boolean);
begin
  acabou := sera;
end;

function  tBetaLib.devolvebranco (qtd : integer) : string;
var
  aux : string;
  i : integer;
begin
  for i:=1 to qtd do
     aux := aux + ' ';

  result := aux;
end;

function  tBetaLib.retirazeroesq (variavel : string) : string;
var
  i : integer;
  aux : string;
begin
  i := 1;

  while copy(variavel,i,1) = '0' do
```

```
      i := i + 1;

   aux := copy(variavel,i,length(variavel)-i+1);
   result := aux;
end;

function  tBetaLib.dow(ddata : string) : integer;
begin
   result := dayofweek(strtodate(ddata));
end;

procedure tBetaLib.mouseparacontrole(Controle: TControl);
var
  IrPara: TPoint;
begin
  if pos('Web',controle.Name) <> 0 then
  begin
    if (Controle.Visible) and (Controle.name = 'SubirImage') then
    begin
      IrPara.X := Controle.Left + (Controle.Width - 10);
      IrPara.Y := Controle.Top + 10;
      if Controle.Parent <> nil then
        IrPara := Controle.Parent.ClientToScreen(IrPara);
    end
    else
    begin
      IrPara.X := Controle.Left + (Controle.Width - 10);
      IrPara.Y := Controle.Top + (Controle.Height - 10);
      if Controle.Parent <> nil then
        IrPara := Controle.Parent.ClientToScreen(IrPara);
    end;
  end
  else
  begin
```

```
    IrPara.X := Controle.Left + (Controle.Width div 2);
    IrPara.Y := Controle.Top + (Controle.Height div 2);
    if Controle.Parent <> nil then
      IrPara := Controle.Parent.ClientToScreen(IrPara);
  end;
  SetCursorPos(IrPara.X, IrPara.Y);
end;

function tBetaLib.mouseshowcursor(const Show: boolean): boolean;
var
  I: integer;
begin
  I := ShowCursor(LongBool(true));
  if Show then
  begin
    Result := I >= 0;
    while I < 0 do
    begin
      Result := ShowCursor(LongBool(true)) >= 0;
      Inc(I);
    end;
  end
  else
  begin
    Result := I < 0;
    while I >= 0 do
    begin
      Result := ShowCursor(LongBool(false)) < 0;
      Dec(I);
    end;
  end;
end;

procedure tBetaLib.RestrictMouse(left,top,rigth,bottom : integer) ;
```

```
var
  r : TRect ;
begin
  r := Rect(left,top,rigth,bottom) ;
  ClipCursor( @r ) ;
end;

procedure tBetaLib.UnRestrictMouse ;
begin
    ClipCursor( nil ) ;
end;

function tBetaLib.VerificaAcabou : boolean;
begin
 result := acabou;
end;

procedure tBetaLib.SimulaClick(x,y : integer);
begin
 Mouse_Event(MOUSEEVENTF_ABSOLUTE or
MOUSEEVENTF_LEFTDOWN, x, y, 0, 0);
 Mouse_Event(MOUSEEVENTF_ABSOLUTE or MOUSEEVENTF_LEFTUP, x,
y, 0, 0);
end;

function tBetaLib.ConsultaDicionario(palavra : string; Lista : TStringList) :
Tstrings;
var
 i, tentativas : integer;
 achou : boolean;
begin
 auxLista.Clear;
 achou := false;
 tentativas := 0;
```

```
  for i:=0 to Lista.Count - 1 do
  begin
    application.ProcessMessages;
    if pos(palavra,Lista.Strings[i]) = 1 then
    begin
      auxLista.Add(Lista.Strings[i]);
      achou := true;
    end
    else
      if achou then
        inc(tentativas);
  end;

   result := auxLista;
end;
end.
```

7- Código do formulário auxiliar a biblioteca de funções:

```
unit auxlib;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
  ExtCtrls;

type
  TauxBetaLib = class(TForm)
    Timer1: TTimer;
    procedure Timer1Timer(Sender: TObject);
```

```
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
   auxBetaLib: TauxBetaLib;

implementation

uses UBetaLib;
{$R *.DFM}

procedure TauxBetaLib.Timer1Timer(Sender: TObject);
begin
  BetaLib.alteraacabou(false);
  Timer1.Enabled := false;
end;

end.
```

ANEXO II

- ARTIGO -

BETA BROWSER – BROWSER ESTRUTURADO EM TECNOLOGIAS ASSISTIVAS

João Bosco da Motta Alves, Andréa Miranda e Carlos Eduardo Gonçalves

RESUMO

As pessoas que apresentam algum tipo de limitação se vêem impedidas, por vários motivos, de ter acesso às Tecnologias de Informação e Comunicação. Entretanto a revolução informacional é uma realidade e um processo irreversível na história da sociedade pós-moderna. Portanto, o desenvolvimento de tecnologias alternativas que tenham a acessibilidade como conceito primordial, é fundamental para que as desigualdades sociais e o número de excluídos digitais não aumentem ainda mais. Diante desta realidade, este projeto apresenta uma proposta de estudo e desenvolvimento de um browser utilizando a ferramenta Delphi. Esse contará com sistema de varredura de teclas e com um teclado virtual para a digitação dos endereços, afim de que os usuários com limitação física e/ou motora possam navegar na Web, com maior facilidade e menor desgaste físico e emocional. Para tanto, será feita a avaliação da usabilidade da ferramenta proposta objetivando adaptá-la ao máximo, às necessidades deste grupo de usuários com tais limitações.

ABSTRACT

People who need more assistence because have any kind of handicap, feel obstructed to have access to the Information and Comunication Tecnology. However the information revolution is a reality and a irreversible process in the history of postmodern society. So, the development of alternative tecnologies who has the acessibility as prime concept, is basic to stabilish the socials disparates and the number of digital excluded. In face of this reality, this project shows a propose of studie and development of a browser with Delphi tool. With a system of scanning buttons and a virtual keyboard computer, the users who as fisical and/or hand coordinating impairment will have the ability to surf the web, with major facilities and minor fisical and emotional consuming. As much, will be necessary the avaliation of the usability of the proposed tool with the purpose to maximize the adaptation to the necessity of this group of users.

INTRODUÇÃO

O mundo em que vivemos traz aos deficientes físicos grandes dificuldades ao acesso à informação. Estas dificuldades se evidenciam, e na era da informação chegam através das tecnologias e páginas da Web pouco acessíveis.

De acordo com o mapa de exclusão digital divulgado pela fundação Getúlio Vargas – FGV e pelo comitê para democratização da Informática – CDI, o uso da Internet aumentou de 10% para 15% e, segundo dados do Censo/2000, cerca 24,5 milhões de pessoas apresentam algum tipo de incapacidade ou deficiência. Entretanto, observa-se que nas Instituições de Ensino Superior as iniciativas que minimizem este cenário ainda são muito insipientes. Paralelamente a isto, há pouco interesse por parte, não só das Universidades, como também das empresas em desenvolver tecnologias acessíveis que possibilitam a inclusão digital e conseqüentemente o exercício da cidadania de forma efetiva por parte dos Portadores de Necessidades Especiais.

Uma das abordagens adotadas para minimizar este problema são as tecnologias assistivas, que são quaisquer dispositivos capazes de facilitar a execução de uma tarefa.

Surge então a necessidade de desenvolvimento de ferramentas que auxiliem estes usuários a acessarem as páginas de internet, a exemplo dos navegadores para deficientes visuais.

A FERRAMENTA

A ferramenta proposta deve possibilitar ao usuário a capacidade de acessar as páginas da web, pois dispende a necessidade de total coordenação dos membros superiores, bastando apenas um mínimo de controle do indicador, para acionamento do botão do mouse ou dispositivo que o substitua.

Isto é possível pois os botões da interface não necessitam do apontamento do ponteiro do mouse com o posterior clique do botão do mesmo. Esta interface tem a capacidade de guiar o ponteiro do mouse para cima do botão ativo, bastando a este clicar o botão. O ativamento dos botões é percebido pelo destaque (aumento de tamanho) que cada botão recebe no momento da sua ativação. Esta ativação ocorre de forma sequencial e pode ser ajustado o tempo em que os botões alternam sua ativação. Este tipo de configuração, assim como outras, deverão ser feitas por usuários que auxiliem o portador de deficiência, pois a interface de configuração não possui o mesmo comportamento do resto da interface do aplicativo, tendo então características comuns a interfaces de outros programas.

Além destes recursos o sistema possui um teclado virtual, a exemplo dos teclados virtuais do Windows, mas com a diferença de que os botões deste possuem o mesmo tipo de ativação dos outros já descritos. Além disso as linhas desse teclado recebem o primeiro destaque através de um quadro cinza que aparece atrás delas, também sequencialmente. Caso a letra que o usuário queira esteja na linha ativa, ele aciona o botão do mouse, passando a alternarem-se os botões da linha com o mesmo destaque dos outros botões.

Como último recurso, mas não menos importante, o dicionário de palavras que reduz consideravelmente o tempo de digitação dos endereços no

teclado virtual, sugerindo palavras que completem a que o usuário esteja digitando.

Pela limitação a que o projeto submete-se (por ser estudo acadêmico e não ferramenta comercial ou de estudo mais aprofundado), não é possível navegar dentro das páginas que o usuário entrar, então o usuário deverá conhecer previamente o endereço que deseja visualizar no navegador. Esta etapa será uma sugestão ao leitor de trabalho futuro e que complementará o estudo.

REFERÊNCIAS

[1] DATI. FAQ: Frequently Asked Questions. Disponível na internet. http://www.asel.udel.edu/dati/atdef.html. 20 março 2001.

[2] KOON, Ricardo A, VEGA, Maria Eugenia de la. El impacto tecnológico en las personas com discapacidad. In: Congresso International de Informática Educativa Especial, 2, 2000.

[3] MONTOYA, Rafael S. Los sistemas de ayudas basados en la tecnologia de la informacíon. In: Congresso Internacional de Informática Educativa Especial, 1, 1998, Neuquén – Argentina.

[4] NEWELL, Alan F, GREGOR, Peter. Human Computer Interfaces for people with disabilities. In: ELSEVIER SCIENCE. Handbook of Human-Computer Interaction. 1997, p.813-824.

[5] CYBIS, Walter de Abreu. Engenharia de Usabilidade: Uma Abordagem Ergonômica. Disponível em: http://www.labiutil.inf.ufsc.br/apostila.htm. Acesso em: 5 de julho de 2003.