



# VCL Tips & Triks

Samuel “Muka” David

Embarcadero

Conference

# Apresentação



## Samuel “Muka” David

- Especialista Delphi - Softplan
- Instrutor Oficial Delphi
- Certificado Delphi [7..Master]
- MVP Embarcadero
  - Delphi Conference [2009..2017], Extreme Delphi, DephiSquad
  - 4 anos colunista do Study Guide, revista Active Delphi
  - Ex-Coordenador DUG-RS



# O que fazemos?



Não construímos aviões ou naves espaciais





# O que fazemos?



Reformamos carros antigos



# Agenda



Handle e Mensagens – Identificando Controles

Hook de Teclado

Mesclando Browser a Aplicação

Disparando eventos Delphi em um elemento HTML

Usando Google Maps em um app Vcl

Interagindo com Google Translate

Manipulando DFMs - Como localizar e substituir todas suas imagens.

# Handle e Mensagens



- No windows, tudo possui um Handle
- Handle é um ID inteiro que identifica um controle no OS
- VCL é toda baseada em controles do Windows
- WinControls x GraphicControls
- WinControls são manipulados através de mensagens do Windows
- Mensagens podem ser enviadas para qualquer Handle de qualquer aplicação.

# Identificando Controles



Identificar o Handle de um controle baseado em sua posição.

- `WindowFromPoint;`

Buscar a posição atual do mouse na área de trabalho.

- `GetCursorPos;`

Identificar nome de uma classe.

- `GetClassName;`

Obter o texto de um controle.

- `GetWindowText;`



# Identificando Controles



## Obter o nome de um controle Delphi

- Global Atom Table
- Controls.pas - InitControls populando a GlobalAtom.
- WindowAtomString := Format('Delphi%.8X',[GetCurrentProcessID]);

## Destacar um controle

- GetClientRect(pHandle,IRect)
- GetDC(pHandle)
- CreatePen(PS\_SOLID, 2, ColorToRGB(clRed))

## Identificar onde o controle está inserido

- GetParent(IHandle);

# Hook de teclado



Hook pode ser Global ou Local, Global deve estar em uma Dll;

## Implementar 4 método

- Instalar o Hook;
- Método chamado pelo Hook;
- Chamada para o método propagar o Hook;
- Desinstalar o Hook;

# Hook de teclado



## Instalar o Hook;

- `SetWindowsHookEx(idHook: Integer; lpfn: TFNHookProc; hmod: HINST; dwThreadId: DWORD): HHOOK; stdcall;`
- "idHook": Tipo do hook (WH\_KEYBOARD);
- "lpfn" : Método que será executado pelo hook;
- "hmod" : handle indentificador da DLL (HInstance);
- "HINST" : Handle do App que o hook estará associado. Zero para todos;
- Retorna o identificador do Hook.

# Hook de teclado



## Método chamado pelo Hook;

- KeyboardProc(nCode: Integer; wParam: WPARAM; lParam: LPARAM): LRESULT; stdcall;
- nCode determina como a mensagem é processada pelo OS (HC\_ACTION);
- wParam retorna a tecla que foi pressionada;
- lParam traz informações adicionais, entre elas se a tecla ALT está pressionada;
- lParam ALT -> ((HiWord(Msg.lParam) and KF\_ALTDOWN) <> 0);



# Hook de teclado

Chamada para o método propagar o Hook;

- `Result := CallNextHookEx(Fhook, nCode, wParam, lParam);`

Desinstalar o Hook;

- `UnhookWindowsHookEx(Fhook): BOOL;`

Comunicação entre Dll e App

- `Function InstallHook(pHandle: Integer): integer;`
- `PostMessage(FHndWinHook, HOOK_MSG, wParam, lParam);`

# Mesclando Browser a Aplicação



TWebBrowser - API do IE/Edge, default IE7

Determinar a versão do browser usada pelo app

- FEATURE\_BROWSER\_EMULATION

[HKEY\_CURRENT\_USER\Software\Microsoft\Internet Explorer\Main\FeatureControl\FEATURE\_BROWSER\_EMULATION]  
"MeuExecutavel.exe"=dword:0

# Mesclando Browser a Aplicação



## Carregar pagina web por um arquivo externo

- TWebBrowser.Navigate

## Carregar pagina web por um Stream

- TPageProducer – Armazenar o HTML
- WebBrowser.Document – Interface para o Documento do HTML
- (WebBrowser.Document as IPersistStreamInit).Load(IStream);
- TStreamAdapter - Implementa interface IStream e consome um TStream;

# Adicionando eventos em uma pg web



## Acessar elementos da pagina

- IDocument := WebBrowser.Document as IHTMLDocument2
- IBody := IDocument.body as IHTMLElement;
- IElementCollection := IBody.all as IHTMLElementCollection;
- IHTMLElement -> id



# Adicionando eventos em uma pg web



## Adicionar evento

- IHTMLElement2 -> attachEvent(Event, pdisp);
- Event -> String com o nome do evento Html (onclick, onkeyup);
- Pdisp -> Objeto que implemente a interface Idispatch

## Pdisp

- Criar classe herdada de TComponent;
- Implementar \_AddRef, \_Release e QueryInterface;
- Criar Propriedade do tipo TNotifyEvent;
- Executar evento no método Invoke;

## Adicionar uma Mapa em um form VCL

- Chave para uso das Api Google;
- Montar HTML com API do Google Maps;
- Criar um Elemento HTML para notificar o Delphi;
- Criar um evento no Delphi e passar;
- Interação toda feita por JSON serializado pelo JavaScript;

## Json

- `TJSONObject.ParseJSONValue(const Data: string)`
- `TJsonValue -> TJSONObject, TJSONArray, TJSONString`
- `TJSONPair -> JSONPair.JsonString.Value e JSONPair.JsonValue.Value`

## Criar um tradutor de texto

- Chave para uso das Api Google;
- <https://www.googleapis.com/language/translate/v2?>
- Parâmetros: **key**=%s&**q**=%s&**source**=%s&**target**=%s
- XMLHttpRequest - Get para o Google translate;
- ResponseText : Retorna uma string Json com a tradução;

## Criar um aplicativo de fala

- Chave para uso das Api Google;
- [https://translate.google.com/translate\\_tts?](https://translate.google.com/translate_tts?)
- Parâmetros: **ie=UTF-8&client=tw-ob&q=%s&tl=%s**
- XMLHttpRequest - Get para o Google translate;
- ResponseStream: Retorna um objeto que implementa a interface IStream;
- TOutputStream faz o meio de campo de um IStream para um TMemoryStream;
- Salvar o Stream em um arquivo mp3.



# Manipulando DFMs



## Listar DFMs do Diretório

- `TDirectory.GetFiles(Diretorio, '*.dfm', TSearchOption.soAllDirectories);`

## Extrair componentes do DFM

- Token para identificar a linha que tenha 'object' e 'inherited' mais a classe do componente que se quer extrair.
- `TImageList -> Bitmap`
- `TImage -> Picture.Data`
- `TSpeedButton, TBitBtn -> Glyph.Data`
- `ObjectTextToBinary -> TMemoryStream` possui `ReadComponent` que retorna um `TComponent`, com ele extraímos a imagem de cada propriedade.

# Manipulando DFMs



## Substituindo imagens:

- Imagens extraídas são salvas, com o nome da Unit, Classe do componente, nome do componente e em caso de um TImageList, o índice da imagem dentro da lista, criando um catálogo de imagens.
- Escolhe-se a Imagem que se quer substituir e busca por todas as imagens iguais no catálogo;
- `Bitmap.Canvas.Pixels[X,Y]`; Extrai a cor do pixel selecionado.
- Buscamos as Units novamente extraímos os componentes conforme o catálogo, e substituímos as imagens.
- `ObjectBinaryToText` transforma nosso componente em Texto onde substituímos através de um `StringReplace` a Imagem antiga pela nova.



# BRIGADO



Samuel “Muka” David

[MukaDavid@gmail.com](mailto:MukaDavid@gmail.com)

[linkedin.com/in/mukadavid](https://linkedin.com/in/mukadavid)

[facebook.com/mukadavid](https://facebook.com/mukadavid)

Embarcadero

Conference





# Embarcadero Conference