

BIBLIOTECA LIBDISP



EXEMPLO DE USO

Tutorial de utilização da Biblioteca LibDisp para comunicação com painéis de chamada Wyma no sistema operacional Windows.

Rev.0 – 19/01/2017 – Revisão inicial

Rev.1 – 30/05/2017 – Acrescentados métodos para painel chamada fiscal.

Rev.2 – 19/09/2018 – Modificado método “CallTicket”

Índice

1. Introdução	2
2. Requisitos para iniciar	2
3. Instalando	3
4. Software aplicativo exemplo	6
5. Biblioteca LibDisp	7
6. Código exemplo em C#.NET	8
7. Registrando o arquivo typeLib (tlb), no Windows.....	9
8. Criando os componentes no Delphi	11
9. Criando um projeto no Delphi.....	12

1. Introdução

A Biblioteca de Comunicação LibDisp é um arquivo de extensão de aplicativo (DLL) com diversas funções prontas, que embutem e simplificam a implementação de um protocolo de comunicação.

A biblioteca foi concebida usando a tecnologia Microsoft .NET 4, portanto, para sua utilização será necessário possuir no mínimo a mesma versão, além das ferramentas de desenvolvimento. O framework pode ser adquirido no link abaixo:

<https://www.microsoft.com/pt-br/download/details.aspx?id=24872>

Para este projeto, utilizou-se o Visual Studio 2017, que pode ser instalado gratuitamente a partir do site da Microsoft, disponível no link:

<http://www.visualstudio.com/downloads/download-visual-studio-vs>

Selecione uma versão com linguagem C#.

2. Requisitos para iniciar

Antes de iniciar o projeto, verifique se recebeu todos os arquivos necessários. Você deve ter os seguintes arquivos:

1-setup.exe = Instalador dos arquivos de código em C#.NET para testar a comunicação, através do aplicativo exemplo e arquivos de biblioteca “DLL” e “TLB”.

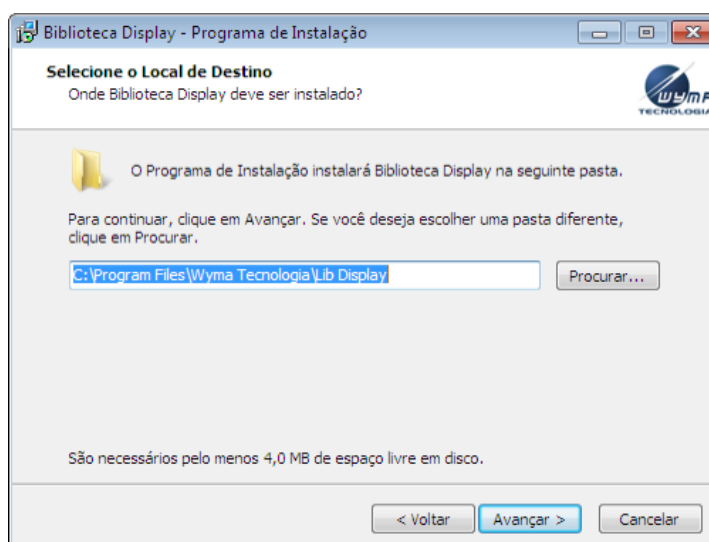
Também deve ter seu endereço IP configurado (consulte o manual de instalação do painel).

3. Instalando

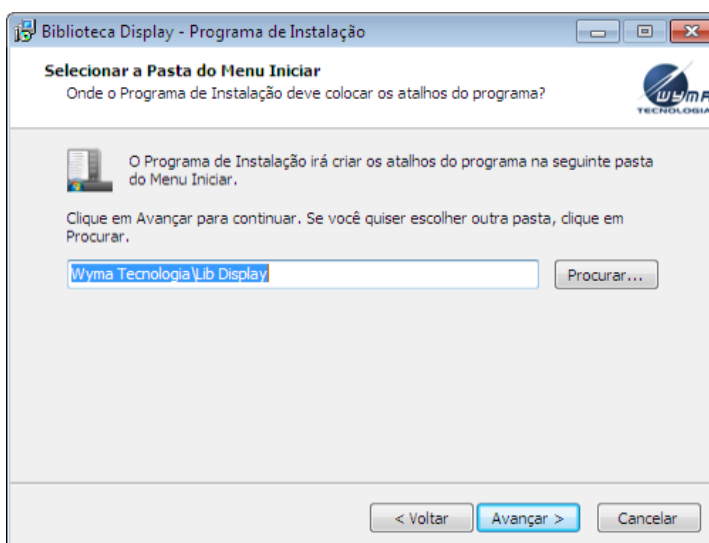
Localize o arquivo “**setup.exe**” e com dois cliques abra o instalador. Clique no botão “**Avançar**” para iniciar a instalação.



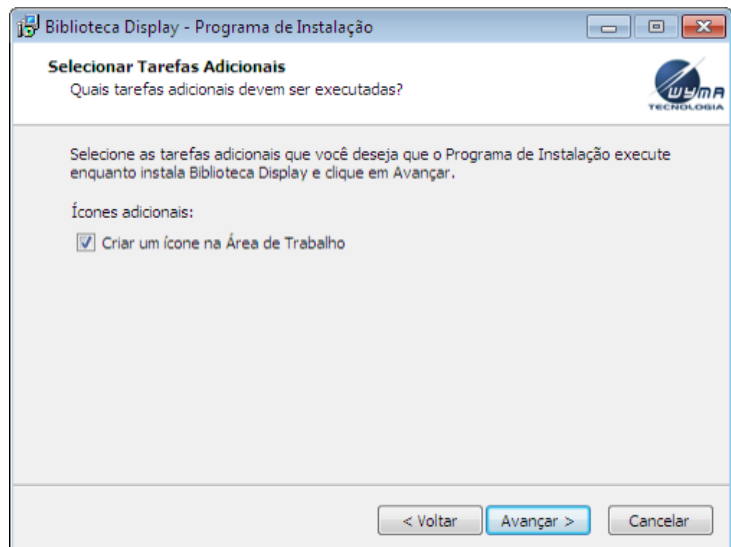
Sugerimos deixar a mesma pasta de instalação que será usada como referência neste documento.



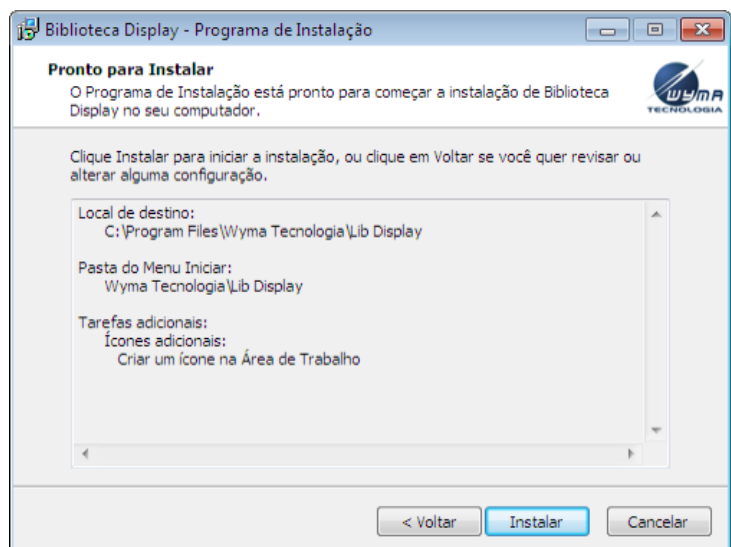
Esta é a pasta de atalhos para o aplicativo de teste.



Se desejar criar um atalho na área de trabalho marque a opção ao lado.



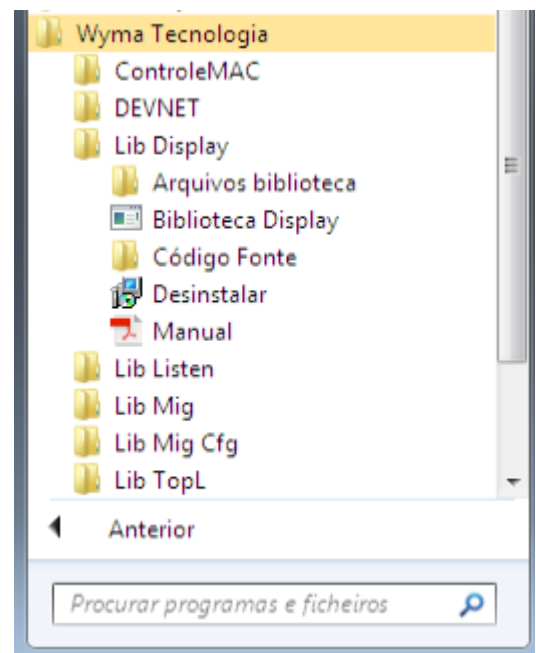
Este é o resumo das opções escolhidas para a instalação, clique em instalar para finalizar.



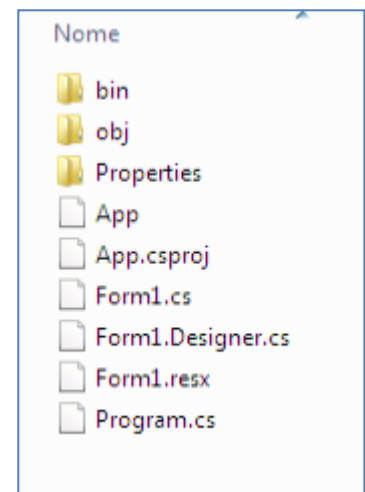
Marque a opção "Executar" para iniciar o aplicativo assim que concluir a instalação.



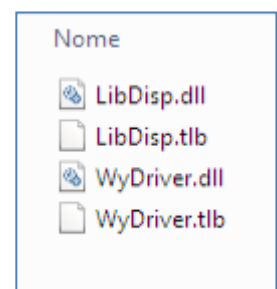
Após a instalação alguns atalhos serão criados dentro da pasta “Wyma Tecnologia” o nome do atalho para o aplicativo será “Lib Display”. Note que existe um atalho para a pasta “**Código fonte**” com o código de um projeto exemplo e também um atalho para uma pasta “**Arquivos biblioteca**”, onde estão os arquivos dll e tlb.



Clique no atalho “**Código Fonte**” para abrir a pasta com os arquivos do projeto.



Clique no atalho “**Arquivos biblioteca**” para abrir a pasta com os arquivos “dll” e “tlb”.



4. Software Aplicativo exemplo

O software aplicativo exemplo deve ser usado para testar a chamada do painel e verificar se toda a infraestrutura de rede esta funcionando.

O uso do software é bem simples:

1º Abra o software aplicativo: **“Biblioteca Display”**.

2º Preencha o campo IP com o endereço IP do painel, preencha também qual a linha de display, caso exista mais de 1 linha será necessário especificar e finalmente preencha a letra/número do ticket, número do local de atendimento e nome do atendimento.

3º Com o botão de número específico, você pode escolher a informação que aparece no display.

4º Com os botões com gerenciamento no painel, você ativa uma chamada informando qual o local de atendimento, porém, o número do ticket é controlado pelo próprio sempre chamando o próximo número.

5º Com os botões de chamada fiscal você insere ou retira o número do caixa de um painel de chamada de fiscal.

The screenshot shows a Windows-style application window titled 'Form1'. It contains several sections of controls:

- IP do Display:** A text box containing '192.168.0.113' and an 'Aplicar' button.
- Configuration Fields:** Four spinners for 'Linha:' (value 1), 'Letra:' (value A), 'Ticket:' (value 1), and 'Mesa:' (value 1). Below them is a text box for 'Nome atendimento:'.
- Comando para chamar número específico:** A button labeled 'Clique para exibir chamada especifica no painel'.
- Comandos para gerenciamento no painel :** Three buttons: 'Clique para solicitar ao painel chamar o próximo', 'Clique para repetir a chamada no painel', and 'Clique para ajustar o número da chamada'.
- Comando de chamada de fiscal:** Two buttons: 'Inserir mesa' and 'Retirar mesa'.
- Comando para leitura de tempo de atd:** A button labeled 'Informação Atendimento'.
- Display Area:** A large, empty rectangular area at the bottom, likely for displaying the output of the commands.

5. Biblioteca LibDisp

Propriedades:

String **_serverIp** – Endereço IP do painel (servidor), remoto (RW).

String **_status** – Informativo sobre status do último comando chamado (R).

Métodos:

Nome:

CallTicket

Descrição:

Envia comando para o painel com as informações de chamada.

Cabeçalho:

bool CallTicket(byte dispRow, char alpha, ushort ticketNum, byte boxNum, string depName)

Parâmetros:

dispRow – Informa qual a linha do painel, em caso de painel multilinhas.

alpha – Informa a letra do ticket a ser mostrado na chamada.

ticketNum – Número do ticket a ser mostrado na chamada.

boxNum – Número do local a ser mostrado na chamada.

depName – Nome do departamento a ser mostrado na chamada.

Nome:

CallNext

Descrição:

Envia comando para o painel mostrar o próximo número de ticket no display com o local de atendimento informado. Neste comando o número do ticket é controlado pelo painel.

Cabeçalho:

bool CallNext(byte dispRow, byte boxNum)

Parâmetros:

dispRow – Informa qual a linha do painel, em caso de painel multilinhas.

boxNum – Número do local a ser mostrado na chamada.

Nome:

RepeatCall

Descrição:

Repete a última chamada realizada com o comando CallNext.

Cabeçalho:

bool RepeatCall(byte dispRow, byte boxNum)

Parâmetros:

dispRow – Informa qual a linha do painel, em caso de painel multilinhas.

boxNum – Número do local a ser mostrado na chamada.

Nome:

AdjustNum

Descrição:

Ajusta o número inicial da contagem do número do ticket. Configure antes de começar a usar o comando CallNext.

Cabeçalho:

bool AdjustNum(byte dispRow, byte boxNum, ushort ticketNum)

Parâmetros:

dispRow – Informa qual a linha do painel, em caso de painel multilinhas.

boxNum – Número do local.

ticketNum – Número do ticket a ser ajustado no painel.

Nome:

InsertCall

Descrição:

Solicita ao painel de chamada de fiscal a inclusão do caixa na fila de chamada.

Cabeçalho:

bool InsertCall(byte boxNum)

Parâmetros:

boxNum – Número do local.

Nome:

DeleteCall

Descrição:

Solicita ao painel de chamada de fiscal a retirada do caixa na fila de chamada.

Cabeçalho:

bool DeleteCall(byte boxNum)

Parâmetros:

boxNum – Número do local.

6. Código exemplo em C#.NET

Iniciaremos a análise do código abrindo o Visual Studio e clicando na sequência de menus FILE → OPEN PROJECT, abra o projeto instalado em “c:\Program Files\Wyma Tecnologia\Lib Display\source\App\App.csproj”.

Com o projeto aberto localize “**Solution Explorer**” e expanda o projeto “**App**” e em seguida expanda a pasta “**References**” note que a dll “**LibDisp**” aparece nas referências do projeto.

Agora clique em **Form1.cs** e vamos abrir o código para ver como é feito o uso da DLL.

Note que é criada uma nova instância do objeto com a linha de código abaixo, onde “**DisplayCall**” é o construtor da DLL “**LibDisp**”:

```
private IDisplayCall disp = new DisplayCall();
```

No carregamento do form1 preencha as seguintes propriedades:

Precisamos informar o endereço do servidor remoto, altere e preencha com o IP do seu painel.

```
disp._serverIp = "192.168.0.112";
```

O restante são os códigos dos botões, onde cada botão executa um comando diferente no painel e aqui entra a DLL “**LibDisp**” que tem um comando para simplificar cada uma das funções do painel.

Chamada de próximo ticket, informando os números de ticket e local de atendimento a serem mostrados no display:

```
disp.CallTicket(Convert.ToByte(numericUpDown_row.Value),  
Convert.ToChar(comboBox_alpha.SelectedItem),  
Convert.ToUInt16(numericUpDown_ticket.Value),  
Convert.ToByte(numericUpDown_mesa.Value),textBox_depName.Text);
```

Chamada do próximo ticket no display:

```
disp.CallNext(Convert.ToByte(numericUpDown_row.Value),  
Convert.ToByte(numericUpDown_mesa.Value)); Repete chamada do ticket:
```

Repetição da última chamada no display:

```
disp.RepeatCall(Convert.ToByte(numericUpDown_row.Value),  
Convert.ToByte(numericUpDown_mesa.Value));
```

Ajuste do número de chamada:

```
disp.AdjustNum(Convert.ToByte(numericUpDown_row.Value),  
Convert.ToByte(numericUpDown_mesa.Value),  
Convert.ToUInt16(numericUpDown_ticket.Value));
```

Insere um novo valor na fila do painel para chamada de fiscal:

```
disp.InsertCall (Convert.ToByte(numericUpDown_mesa.Value));
```

Retira um valor da fila no painel para chamada de fiscal:

```
disp.DeleteCall(Convert.ToByte(numericUpDown_mesa.Value));
```

Para auxiliar na compreensão do que ocorre internamente na DLL, você deve ler o parâmetro:

```
disp._status
```

Isto é feito no método abaixo, todas as informações são mostradas na caixa de texto na tela do aplicativo de exemplo.

```
textBox1.Text += disp._status + Convert.ToChar(13) + Convert.ToChar(10);
```

7. Registrando o arquivo typeLib (tlb), no Windows

Para usar a DLL em sistemas não Microsoft .net, é necessário registrar os arquivos TLB que são distribuídos na instalação junto com os arquivos DLL, verifique na pasta de atalhos do programa o atalho para “**LibDisplay → Arquivos biblioteca**”, este é um atalho para a pasta onde são copiadas as DLLs e TLBs necessários para o funcionamento do software.

Para registrar os arquivos TLB, você precisa abrir o **Prompt de comandos** do Windows em modo **administrador**, clicando o botão direito do mouse sobre o atalho esta opção estará disponível.

Certifique-se de ter instalado o Framework indicado no começo do manual, ou será necessário mudar o path abaixo para o caminho de um framework instalado em seu computador.

Digite o seguinte comando:

```
set path="%path%";C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319
```

A linha de comando acima apenas vai facilitar o uso do aplicativo “**regasm**”, assim não teremos que digitar todo o caminho dele para que o aplicativo funcione, basta digitar “**regasm**”.

Agora vamos mudar o diretório do prompt de comandos para a pasta onde estão os arquivos TLB para executar o comando regasm.

Para registrar o WyDriver.TLB execute o comando:

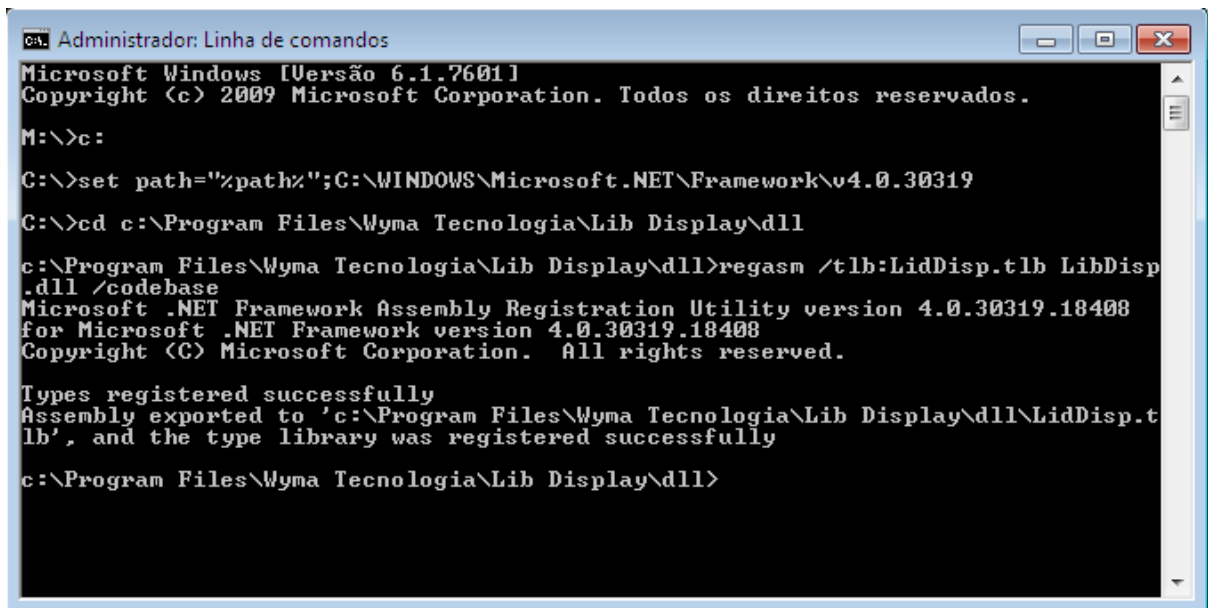
```
regasm /tlb:WyDriver.tlb WyDriver.dll /codebase
```

Para registrar o LibDisp.TLB execute o comando:

```
regasm /tlb:LibDisp.tlb LibDisp.dll /codebase
```

Após este comando tudo vai estar registrado no Windows para começar a usar como um componente.

A tela abaixo exemplifica alguns dos comandos digitados no prompt, note que ao final do registro de cada um dos TLB é indicado se o registro obteve sucesso.



```
Administrador: Linha de comandos
Microsoft Windows [Versão 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. Todos os direitos reservados.

M:\>c:

C:\>set path="%path%";C:\WINDOWS\Microsoft.NET\Framework\v4.0.30319

C:\>cd c:\Program Files\Wyma Tecnologia\Lib Display\dll

c:\Program Files\Wyma Tecnologia\Lib Display\dll>regasm /tlb:LidDisp.tlb LibDisp.dll /codebase
Microsoft .NET Framework Assembly Registration Utility version 4.0.30319.18408
for Microsoft .NET Framework version 4.0.30319.18408
Copyright (C) Microsoft Corporation. All rights reserved.

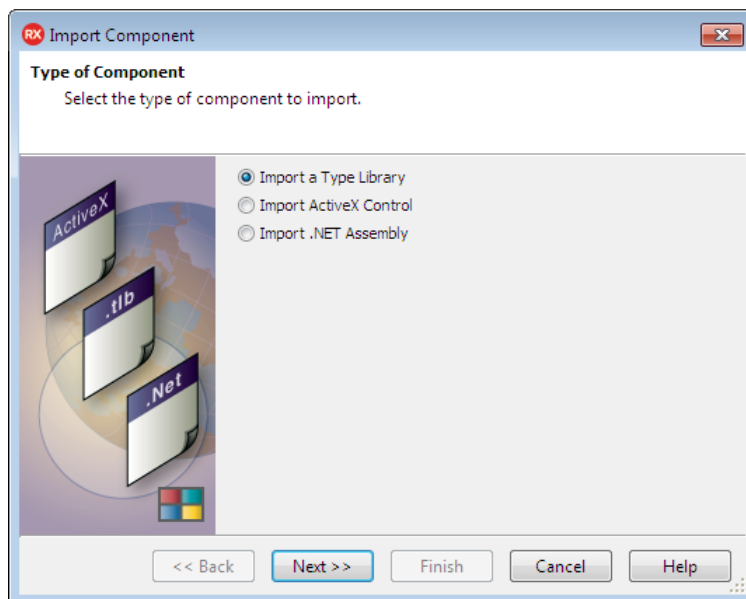
Types registered successfully
Assembly exported to 'c:\Program Files\Wyma Tecnologia\Lib Display\dll\LidDisp.tlb', and the type library was registered successfully

c:\Program Files\Wyma Tecnologia\Lib Display\dll>
```

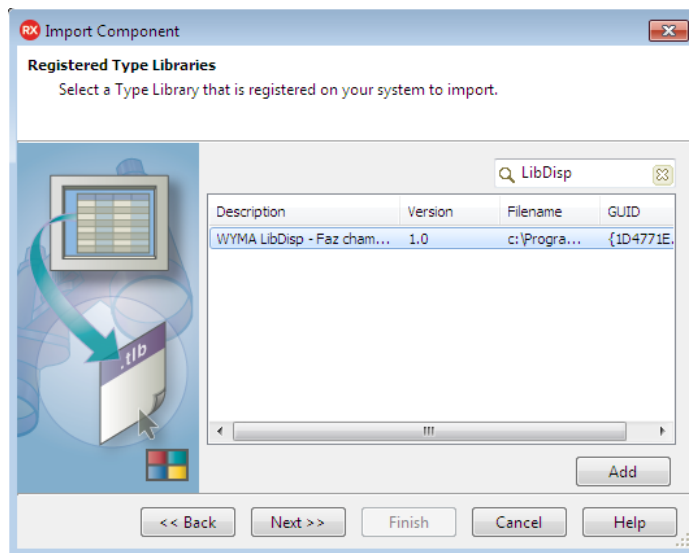
8. Criando um componente WyDriver, no Delphi

O procedimento para usar a DLL como um componente no Delphi é bem simples, o exemplo abaixo foi feito no RAD Studio 10.1 Berlin.

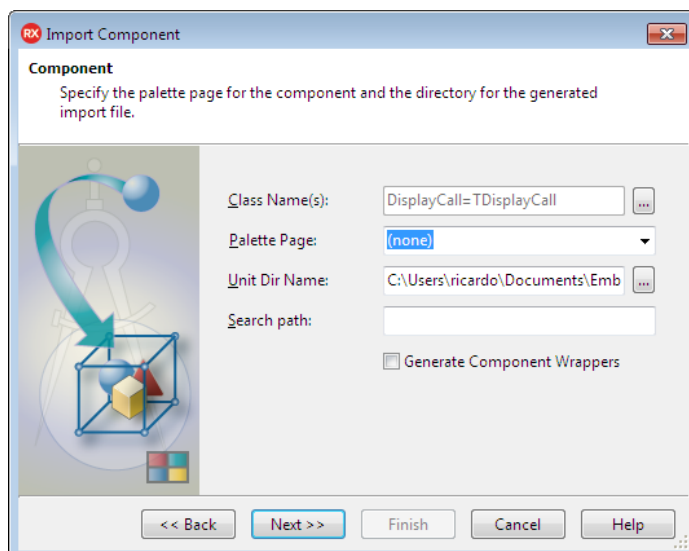
- a) Para criar os componentes vá até o menu em: **“Component”**, depois em **“Import Component”**.
- b) Selecione a opção: **“Import a Type Library”** e clique em **“Next”**:



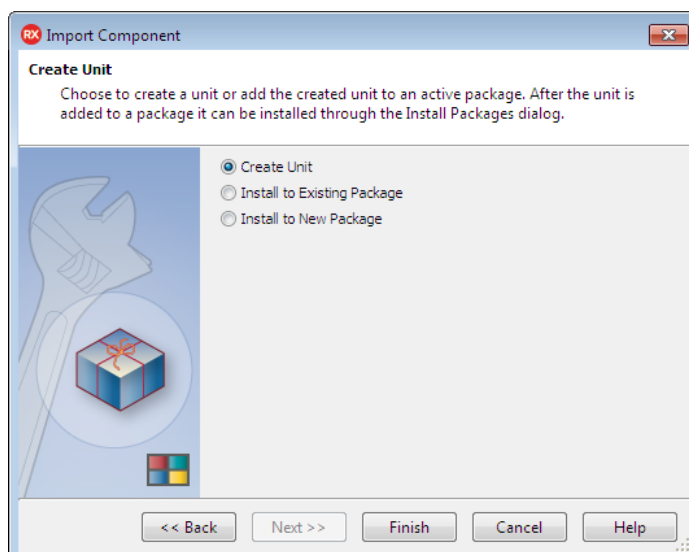
- c) Digite **“LibDisp.tlb”** na pesquisa escolha o item WYMA LibDisp e clique em **“Next”**:



d) Clique em **“Next”**:



e) Por fim escolha **“Create Unit”** e clique em **“Finish”**:



- f) Este componente **“LibDisp.tlb”** depende de outro componente chamado **“WyDriver.tlb”**, é necessário seguir as mesmas etapas acima (passos “a” até “e”), para importar esta outra biblioteca também.

9. Criando um projeto no Delphi

Agora podemos criar um projeto para usarmos estes novos componentes em nosso projeto, siga as etapas abaixo para começar um projeto novo:

- 1) Clique em **“File”**, depois em **“New”** e finalmente em **“VCL Forms application – Delphi”** para criar um projeto novo.
- 2) Localize a caixa de ferramentas **“Tool Palette”** e digite no campo de busca o texto: **“TButton”**, com isto deverá aparecer o componente **“TButton”** logo abaixo, clique duas vezes sobre ele para que ele seja acrescentado ao formulário do projeto.
- 3) Verifique que agora o formulário **“Form1”** tem um botão, clique duas vezes sobre ele para irmos até o código.
- 4) Localize mais acima no código a palavra chave **“uses”**. Em **“uses”**, acrescente as bibliotecas: **ActiveX**, **WyDriver_TLB** e **LibDisp_TLB**, estas bibliotecas são necessárias para o nosso projeto.
- 5) Agora digite o código abaixo para o componente TButton.

```
procedure TForm1.Button1Click(Sender: TObject);
var
  iLibrary: IDisplayCall;
begin
  iLibrary:=CoDisplayCall.Create();      // Cria instância da dll
  iLibrary._serverIp:='192.168.0.112';  // Configura o Ip do servidor Display

  iLibrary.CallTicket();                 // Envia requisição para o servidor Display.
end;
```

- 6) Antes de rodar o exemplo verifique qual o IP do servidor Display e ajuste no código a linha abaixo com o IP correto:

```
iLibrary.serverIp:='192.168.0.112';    // Configura o Ip do servidor Display
```

- 7) Agora basta adequar ao seu programa..