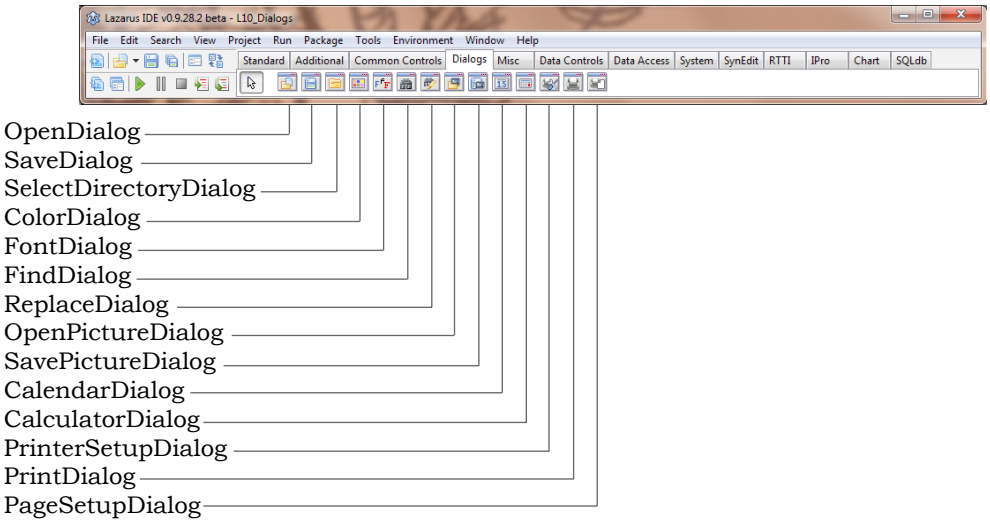
Chapter 10
Dialogs

- Open and save files
- Changing fonts and colours
- Printer dialogs

10

Dialogs

The tab Dialogs offers a number of Dialogs for files, colours, fonts, images, calendar, calculator and printer.



10 We prepare for a new session:

- Create a new sub-directory in the lessons-directory: **`L10_Dialogs`**
- Double click on **`L03_Start.lpi`** to start this project
- Modify the constants **`modTxt`** and **`verTxt`** into **`'L10_Dialogs. '`** and **`'Ver: 1.0.'`**
- With **`Files|Save As`** save the pascalfile as **`L10_DialogsP.pas`**.
- Answer with **`Yes`** on the question "add directory to path".
- Answer with **`No`** on the question "remove old files".
- With **`Project|Project Save As`** save the project as: **`L10_Dialogs.lpr`**.
- With **`Project|Project Options adjust title to: L10_Dialogs`**.

The way to use a Dialog component is to call the method: Execute. We have used this method already in the previous lessons. The Execute method returns "True" if a selection is made and "False" if on "Abort" or the close button is clicked. In this case we don't want to take action.

It is possible to set a number of initial conditions and filters. A title can add some useful information about the dialog. There is no property hint.

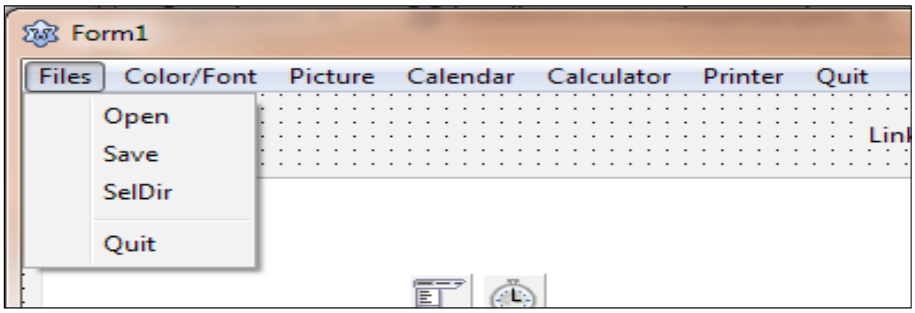
OpenDialog, SaveDialog, SelectDirectoryDialog

Let's start with the file Open- Save- and SelectDir Dialogs.

Add an Open, a Save and a SelectDirectory dialog to the Form by double clicking on the icons for the components: OpenDialog, SaveDialog and SelectDirectoryDialog in the tab Dialogs. Name the dialogs: **DlOpen**, **DlSave** and **DlSelDir**.

We need a mechanism to start these Dialogs. And we need to show an opened file.

Add the next sub menu-items to the menu: File. This is how it will look:



We normally do this by double clicking on the component MainMenu. The Menu-editor opens and we can add before Quit an item "Files". Right click on "Quit", next select Insert New Item (before), call this new menu-item: **MnFiles** and change the caption in: **Files**.

Now right click on the menu-item **Files** and select "Create submenu". A New Item appears. Right click on this new item, select "Insert new item (after)" and repeat this 4 times. Adjust names and captions.

Name	***Caption***
MnFlOpen	Open
MnFlSave	Save
MnFlSelDir	SelDir
MnFlSpace	-
MnFlQuit	Quit

The caption of **MnFlSpace** is used to display a separator in the submenu.

In the files sub-menu, select **MnFlQuit**. In the Events tab, select OnClick and select **MQuitClick**.

Add a memo to the Form and call the memo: **MeData**.

With these components we have all we need to do some opening and saving.

Select **MnFlOpen**, go to the tab events and double click on OnClick. Lazarus will create a new procedure:

```
procedure TForm1.MnFlOpenClick(Sender: TObject);
begin
  ...
end;
```

This procedure will be called when we click on Files|Open and here we can add the actions we should like to perform. Type in the next code:

```
procedure TForm1.MnFlOpenClick(Sender: TObject);
var
s: string;
begin
①  DlOpen.Title:= modTxt + mes01;
②  Sb.Panels[0].Text:= '';
③  If DlOpen.Execute = true then
   begin
④    s:= DlOpen.FileName;
⑤    MeData.Lines.LoadFromFile(s);
⑥    Sb.Panels[0].Text:= s;
   end;
end;
```


① First we adjust the title of the Open Dialog. It makes sense to try to explain what is the intention of a dialog screen. Mes01 stands for **'Select file.'**.

② We like to show the name in the StatusBar. If we do not select anything, we want to show an empty string.

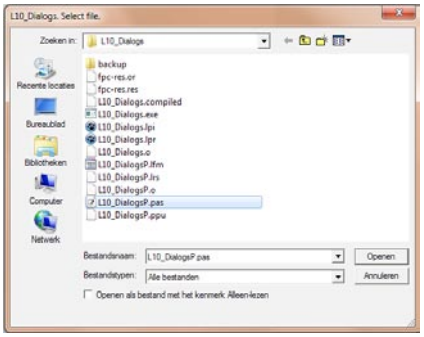
③ The Opendialog is started with the function: **DlOpen.**. This function returns **true** when a file is selected or **false** when no selection is made or when "abort" is clicked.

④ When a file is selected (result is **true**), we copy the full path of this file into **s**.

⑤ Next we load the file in the memo **MeData** using the function: **MeData.Lines.LoadFromFile(s)** where **s** contains the full path file name.

⑥ And we show the selected file name in **panel[0]** van the statusbar.

Now we can compile the program, and when we start it we can use the menu-item File|Open, by clicking on this item:



It is possible to choose any file to open. It will be clear, only files with text produce a readable result. Files with the extension *.o or *.exe will not show much useful information in **MeData**, but files with an extension of *.lpr, *lpi or *.pas can be examined after opening.

Click on a file, f.i. **L10_DialogsP.pas** and the text will show up in **MeData**. If you would make changes in this file, the next time this

It is possible to adjust a couple of parameters in the Open Dialog.

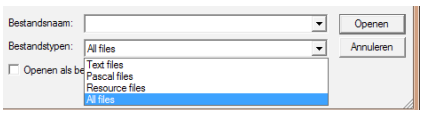
DefaultExt defines the default extension to look for. Doesn't seems to work yet.

It is possible to define a file in the property "FileName". The dialog will point to this file. It is useful in the case a "default" file name can be used, but also an alternative should be available for selection.

The "Filter" is used for the selection of files, depending on the extension. The filter has a description and an extension, separated by "|".

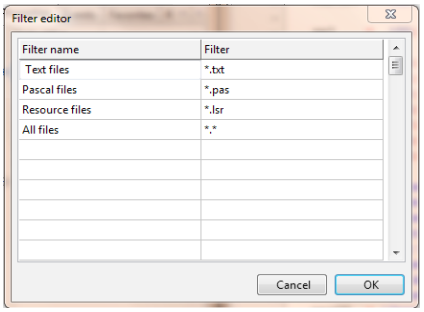

```
Text files|*.txt|Pascal files|*.pas|Resource  files |*.lsr|All files|*.*
```

This filter result in the next file types:



It is by the way a good idea to include "All files | *.*". If users have generated files, that are valid but have changed extensions, it still is possible to select these files.

It is possible to change the filter of an Open Dialog. The type of filter can be made dependent of previous selections or one Open Dialog can be used for several parts in the program.



To adjust the filter during run time:

```
DlOpen.Filter:= 'Text files|*.txt
                |Pascal files|*.pas
                |Resource files|*.lsr
                |Alle files|*.*';
```

It is also possible to use the Filter Editor. Double click on the property Filter in the Object-Inspector of DlOpen.

The property FilterIndex indicates which filter is shown in the extension field of the dialog when started.

If a filterindex is used indexes indicate, index indicates "no preference"; index 1 points to the 1st entry: "Text files | *.txt"; index 2 points to the 2nd entry: "Pascal files | *.pas", and so on.

To start the Open Dialog with the Pascal files filter, set the index to 2.

The property InitialDir indicates the map that will be shown when the Open Dialog is started:

```
DlOpen.InitialDir:= 'e:\';
```

This command will show the root for drive e: when the OpenDialog is started.

Another elegant solution is to set InitialDir to the "current directory". This can be done simply by using the function GetCurrentDir of the **MnFlOpen** procedure:

```
DlOpen.InitialDir:= GetCurrentDir;
```

10 There are quite a lot of options in the property "Options". It is possible to inhibit a change of the map, only ReadOnly files can be shown, etc. In total 24 options are available.

Another good practice is to set the tile of the Open Dialog with a meaningful title. Again all these properties are adjustable during design time, but also during run time.

SaveDialog

The SaveDialog offers the possibility to save data to files. The procedure will be called when we click on Files|Save and we add the next code to the OnClick event of this menu-item:

```
procedure TForm1.MnFlSaveClick(Sender: TObject);
begin
①  DlSave.Title:= modTxt + mes02;
②  Sb.Panels[1].Text:= '';
③  If DlSave.Execute = true
   then
   begin
④    Sb.Panels[1].Text:= DlSave.FileName;
⑤    MeData.Lines.SaveToFile(DlSave.FileName);
   end;
end;
```

① First we set the title in the SaveDialog **`DlSave`**:

```
'L10_Dialogs' + 'Save file.';
```

② Next we clean the contents of **`Sb.Panels[0]`**.

③ Next we start the function **`DlSave.Execute`**. If no selection is made, we do not perform any action.

④ But if a file is selected, we show this file name and path in **`Sb.Panels[0]`** and

⑤ we write the text from **`MeData`** to the selected file.

Of course one has to be careful with overwriting existing files. For this reason it is wise to use the option:

```
ofOverwritePromt
```

If an attempt is made to overwrite an existing file, a warning- and confirmation message is shown. Only if confirmation is given, the file will be overwritten with the new data.