


Lesson 1: Turbo Start

- [1. Turbo Start](#)
- [2. EuroCC](#)
- [3. Properties](#)

[More lessons](#)

[DelphiLand Online](#)

 **Delphi** combines the user-friendliness of **Visual Basic** with the precise control and speed of **C++**. Without lots of "real" programming, you can develop very efficient and fast applications for Windows 95/98 or Windows NT.



Delphi is a so-called **RAD (Rapid Application Development)** tool. Delphi reduces the complicated task of programming Windows applications to the handling of "objects" in a visual environment.


Typing of source code is limited to a strict minimum. As a result, you can fully concentrate on what the program should **do**: this is top-down programming at its highest level! Designing a nice Windows GUI interface becomes a breeze. You don't have to **program** the standard Windows elements: just a few mouse clicks, and there's your fully functional *listbox*, *file dialog box*, or even a full fledged *database grid*!

As a result, debugging is limited to the program lines that you entered yourself, because all these ready-made modules that you use are tested and ready-to-go.

Compared to the limited possibilities of **Turbo Pascal**, or the complexity of **C++**, Delphi is really refreshing. Once you've used it, you'll be fascinated by its sheer development speed.


For starters, let me tell you a few general things about my lessons.

-  Although the lessons were initially written for **Delphi 4/5**, most of the stuff will hold for **D3** and **D2**.
-  **Hyperlinks** are used abundantly for explanations and tips. So you'd better start practicing with the BACK-button of your browser, because that is the **only way** to return to the current lesson ;-)

 **VCL, IDE, OOP...?** In order to get you up to speed quickly on **acronyms**, in one of the first lessons of our second tutorial, we'll make the program **Acron**.

For those who are too impatient: the exe-file of Acron can be found in the **Downloads** section as ACRONX.ZIP. Also download ACRODAT3.ZIP: it contains the list of acronyms. This list will grow. Every new and enhanced version will get a higher number (yep, the next one will be ACRODAT4.ZIP, and afterwards..., and so on).

By the way (BTW), **D1** means **Delphi 1**, so **D5** is...? Correct. For "the 32-bits versions of Delphi, D2, D3, D4 and D5" I came up with the very original **D2+**.

 **Run before you can walk?** Yes, that's possible with Delphi! No need to start studying Turbo Pascal, no need to know anything about object orientation. Just learn as you practice, and we'll fill in the gaps later on.

So, no traditional "Hello, World" proggy in my courses. We'll start immediately with a **real** application. After all, it's R.A.D time! Let's go!



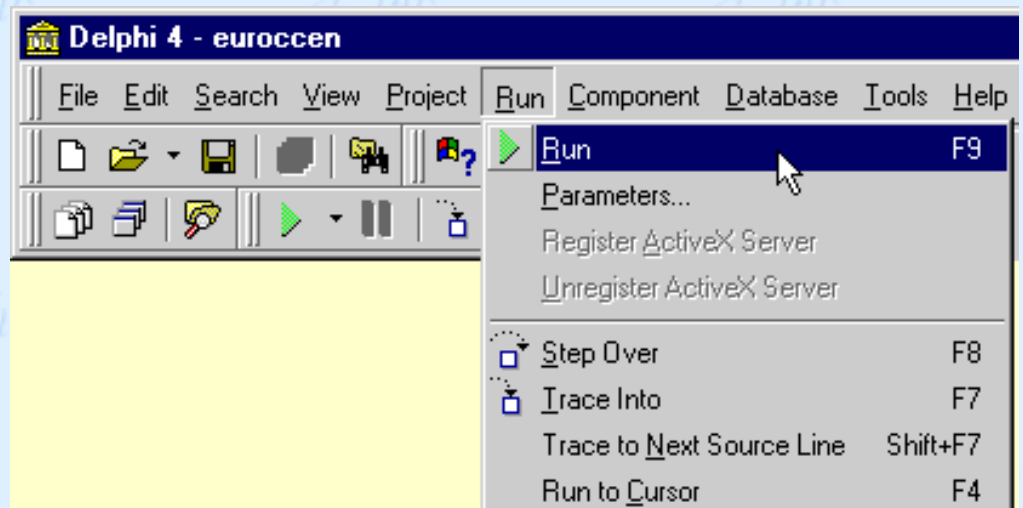
Preparations

1. If you haven't done so already, create a new folder (directory) **\DelphiLa** on your harddisk (C:, D:, E:...). Next, download the first series of lessons **les0103.zip** from the **Downloads** section to this directory and unzip it.
2. If you haven't done so already, create directory **EuroCCEN** "under" \DelphiLa. Next, download **euroccen.zip** (source code for the first finished project) from the **Downloads** section to this directory and unzip it. When checking with Windows Explorer, you'll see euroccen.dpr, euroform.dfm and euroform.pas.
3. Now start the introduction lesson: in Windows Explorer, doubleclick on **index.html** (in folder \DelphiLa). You're all set to follow the first lesson offline!

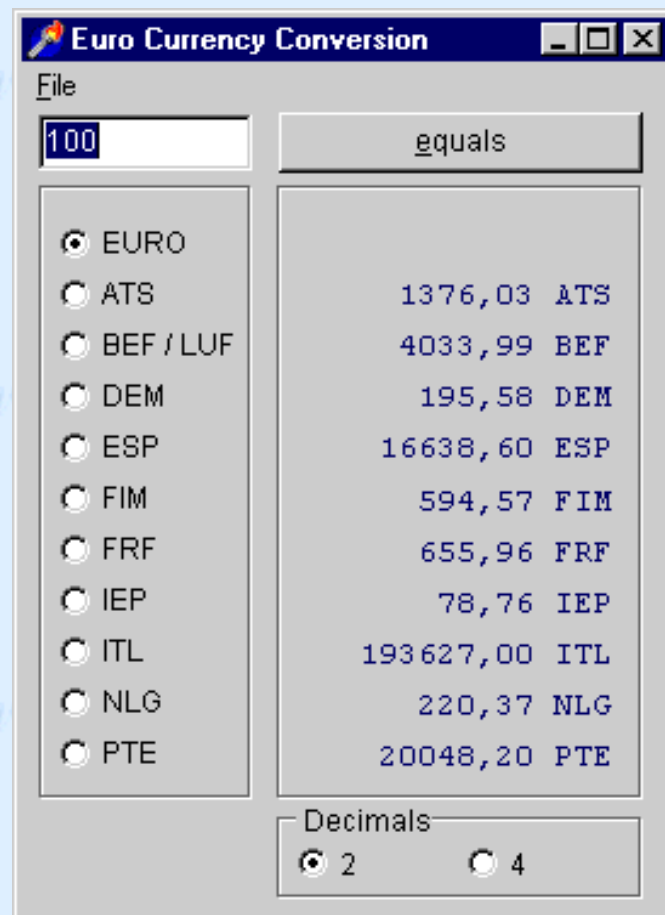
Euro Currency Convertor: compilation

Our very first project is a **Euro Currency Convertor**. The English ready-to-go version is called EuroCCEN. You compile and test it as follows:

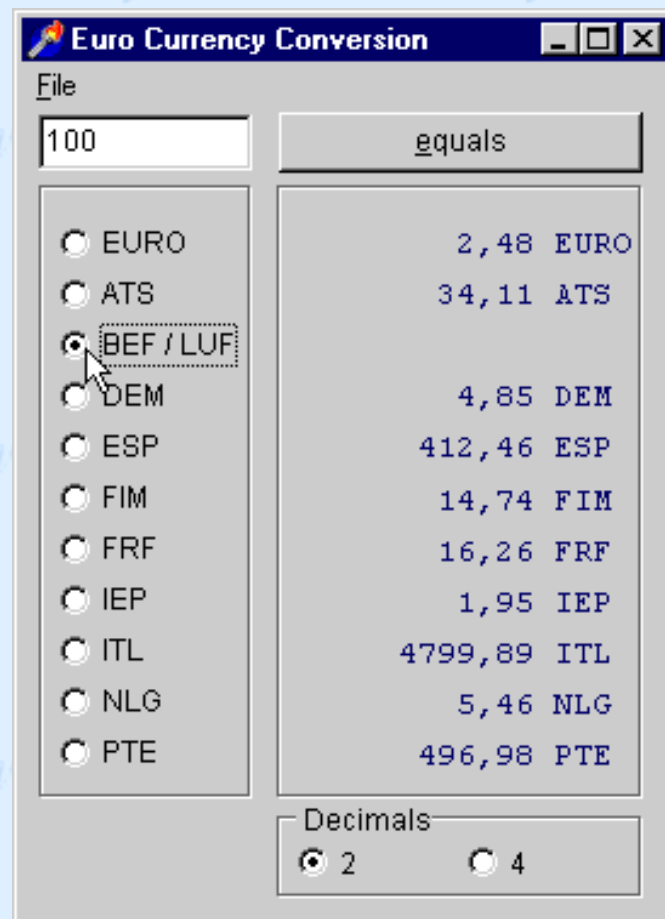
1. Doubleclick in Windows Explorer (or in My Computer) on euroccen.dpr: Delphi starts and it opens the project "euroccen".
2. Open Delphi's Run menu and click Run.



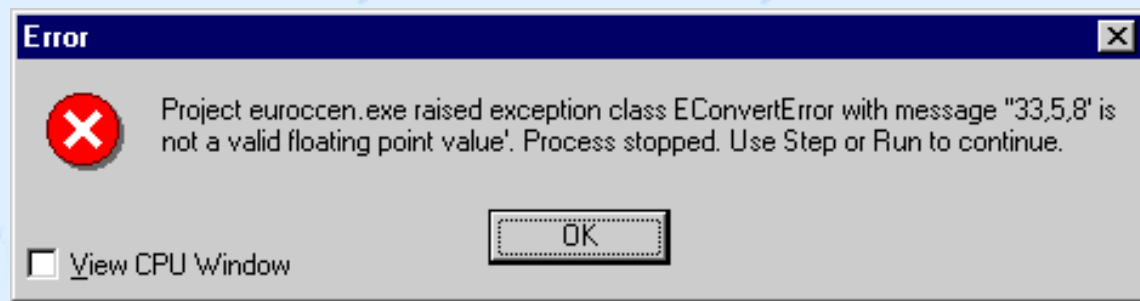
3. Your project is compiled and linked: the result is euroccen.exe. Immediately after that, the program euroccen.exe is launched. All this happens so fast, that you probably even didn't notice that your source code was compiled into an executable program!



4. Enter a new **valid** number in the **EDIT-box**, for example: 25. Click on the button with the caption "equals". In the right frame, you should see the corresponding amounts in the other European currencies.
5. Click on one of the **radiobuttons** for the starting currency, for example BEF/LUF. Note that the corresponding **label** for Belgian en Luxemburg frank becomes invisible. That's because I found it quite silly if the program should show that 25 BEF = 25 BEF... On top of that, this trick gives a clear visual feedback as to the starting currency.



6. Click the radiobutton **4** for the number of decimals. All converted numbers now are shown with 4 decimals (after the *decimal point* or *decimal comma*). This is useful when converting small amounts of certain currencies, e.g. Lire (ITL = Italian Lire). For a display with two decimals, click on the corresponding radiobutton.
7. Enter an **invalid** value in the EDIT-box (like 20,5,7 or ABC) and try a conversion. Delphi responds with an error message and pauses the execution of the application: when trying to convert the **text** "20,5,7" or "ABC" to a **number**, Delphi's *Debugger* noted the error condition. The *Debugger* shows **design** errors as well as

runtime errors!

8. Click the **OK** button in order to close the error-dialogbox. Then press key **F9** (this is the same as the menu-command **Run**). Execution of the program is resumed and you see a second error message. This is the *warning* we programmed ourselves. Only this message will be shown when there is an invalid input when you run the program from *Windows Explorer* or from *My Computer*. Also close this dialogbox by clicking it's **OK** button.



9. Enter a number with a decimal fraction in the EDIT-box. For the decimal separator, you can type **.** (point, dot) as well as **,** (comma): not only 10.5 but also 10,5 is accepted. The program does this by looking at Windows' "Regional Settings" and by replacing the **comma** with a **point**. Or vice versa, of course. This is a golden tip if you make applications for both Anglo-Saxon and non-Anglo-Saxon users. Especially the Americans tend to forget that there are a few countries besides the US of A...
10. Type a number in the EDIT-box and press the ENTER key: just as if you clicked the "equals" button! This gimmick is programmed as follow: if the EDIT-box has the **focus** (meaning: if the text-cursor is visible in this object) and if you press ENTER, a click on the "equals" button is **simulated**.
11. Stop the program via it's menu and Exit, via it's sytem menu, or via it's Close-button (top right of the window).
12. Stop Delphi. If you get the question "Save changes to...?", say no.

Because maybe you changed something in the source code and for the moment we don't want to save these changes.

Euro Currency Convertor as a standalone program

1. Start Windows Explorer and look in the folder where our first project is located. You will see several additional files:
 - o **euroccnl.exe** is the executable file
 - o **euroform.dcu** (dcu= Delphi compiled unit) is the file that was compiled from the source files euroform.pas (pas= Pascal) and euroform.dfm (dfm= Delphi Form)
 - o **euroccnl.dsk** (dsk= Desktop) is the file that remembers how your Delphi-*desktop* looked like when you stopped Delphi. Next time when you open the same project, the positions and dimensions of all the windows will be restored. Nifty!
 - o **other** files: if you fooled around with the project. In the next lessons I'll discuss all those Delphi-files in detail.
2. Doubleclick on euroccnl.exe: the program starts.
3. Try again to convert an illegal value. Now, you will only see the **programmed** error message.
4. Stop the program.
5. If you want to see how EuroCC reacts to a different setting of the "decimal separator", you open Windows' Control Panel, you open the *Regional Settings* and you change the "Decimal symbol": enter a **point** if it was a *comma* (or the opposite: enter a **comma** if it was a *point*). Close the window *Regional Settings*.
6. Start EuroCCEN from the Explorer and note that the results are displayed differently: with decimal **points** instead of *comma*'s (or the opposite, of course). Try a value with a decimal fraction: also in the EDIT-box this same decimal separator is shown, no matter if you type *point* or *comma*.
7. Stop the EuroConvertor. Afterwards, don't forget to reset your *Regional Settings* back to the way they were.

Navigation

Not the **only** way of course... You can also return to the current subject by means of the

navigation menu and then work your way back to where you were. But since the browser has a **BACK** button... ;-)


You can freely [download lessons 1 to 3](#) and the associated source code files.

Members of the [DelphiLand Club](#) can also download the full version of lessons 4 to 10, including all fully commented source code files.

[[TOP](#)]

© Copyright 2000
Studiebureau Festrtaets

Lesson 1: Turbo Start

 **Delphi** combines the user-friendliness of **Visual Basic** with the precise control and speed of **C++**. Without lots of "real" programming, you can develop very efficient and fast applications for Windows 95/98 or Windows NT.



Delphi is a so-called **RAD (Rapid Application Development)** tool. Delphi reduces the complicated task of programming Windows applications to the handling of "objects" in a visual environment.


Typing of source code is limited to a strict minimum. As a result, you can fully concentrate on what the program should **do**: this is top-down programming at its highest level! Designing a nice Windows GUI interface becomes a breeze. You don't have to **program** the standard Windows elements: just a few mouse clicks, and there's your fully functional *listbox*, *file dialog box*, or even a full fledged *database grid*!

As a result, debugging is limited to the program lines that you entered yourself, because all these ready-made modules that you use are tested and ready-to-go.


Compared to the limited possibilities of **Turbo Pascal**, or the complexity of **C++**, Delphi is really refreshing. Once you've used it, you'll be fascinated by its sheer development speed.

For starters, let me tell you a few general things about my lessons.

-  Although the lessons were initially written for **Delphi 4/5**, most of the stuff will hold for **D3** and **D2**.
-  **Hyperlinks** are used abundantly for explanations and tips. So you'd better start practicing with the BACK-button of your browser, because that is the **only way** to return to the current lesson ;-)

 **VCL, IDE, OOP...?** In order to get you up to speed quickly on **acronyms**, in one of the first lessons of our second tutorial, we'll make the program **Acron**. For those who are too impatient: the exe-file of Acron can be found in the **Downloads** section as ACRONX.ZIP. Also download ACRODAT3.ZIP: it contains the list of acronyms. This list will grow. Every new and enhanced version will get a higher number (yep, the next one will be ACRODAT4.ZIP, and afterwards..., and so on).

By the way (BTW), **D1** means **Delphi 1**, so **D5** is...? Correct. For "the 32-bits versions of Delphi, D2, D3, D4 and D5" I came up with the very original **D2+**.

 **Run before you can walk?** Yes, that's possible with Delphi! No need to start studying Turbo Pascal, no need to know anything about object orientation. Just learn as you practice, and we'll fill in the gaps later on. So, no traditional "Hello, World" proggie in my courses. We'll start immediately with a **real** application. After all, it's R.A.D time! Let's go!



Preparations

1. If you haven't done so already, create a new folder (directory) **\DelphiLa** on your harddisk (C:, D:, E:...). Next, download the first series of lessons **les0103.zip** from the **Downloads** section to this directory and unzip it.
2. If you haven't done so already, create directory **EuroCCEN** "under" \DelphiLa. Next, download **euroccen.zip** (source code for the first finished project) from the **Downloads** section to this directory and unzip it. When checking with Windows Explorer, you'll see euroccen.dpr, euroform.dfm and euroform.pas.
3. Now start the introduction lesson: in Windows Explorer, doubleclick on **index.html** (in folder \DelphiLa). You're all set to follow the first lesson offline!

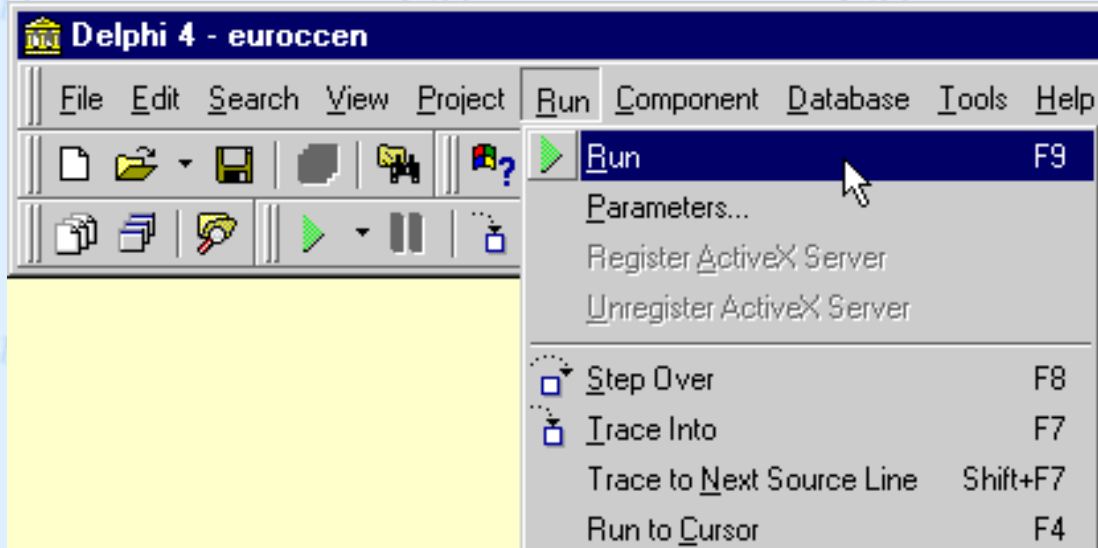
Euro Currency Converter: compilation

Our very first project is a **Euro Currency Converter**. The English ready-to-go version is called EuroCCEN. You compile and test it as follows:

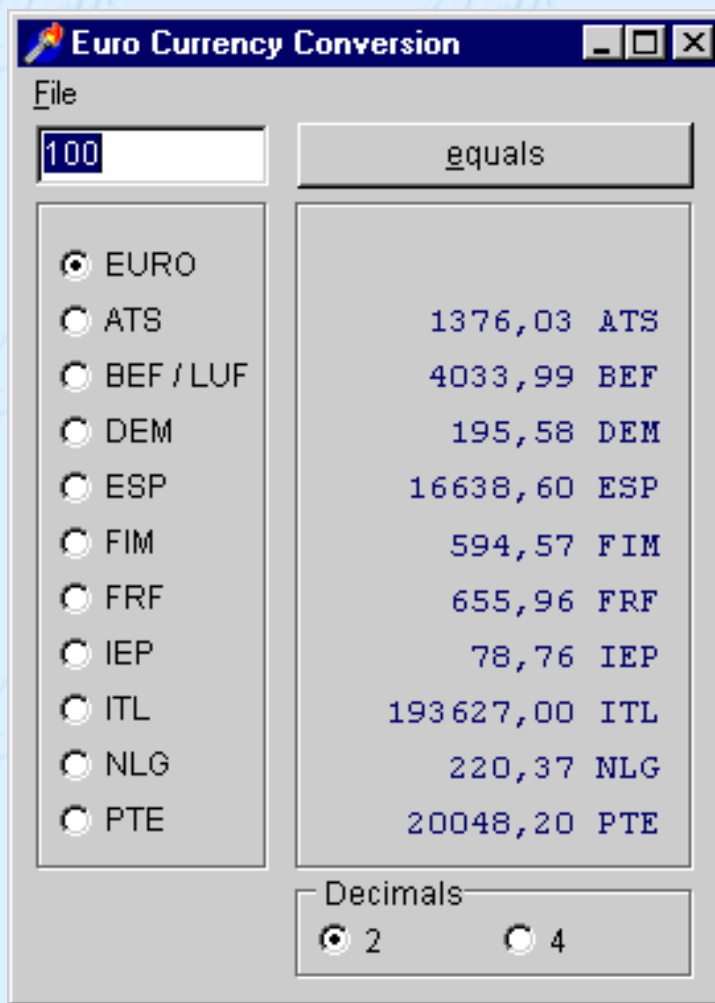
1. Doubleclick in Windows Explorer (or in My Computer) on euroccen.dpr: Delphi

starts and it opens the project "euroccen".

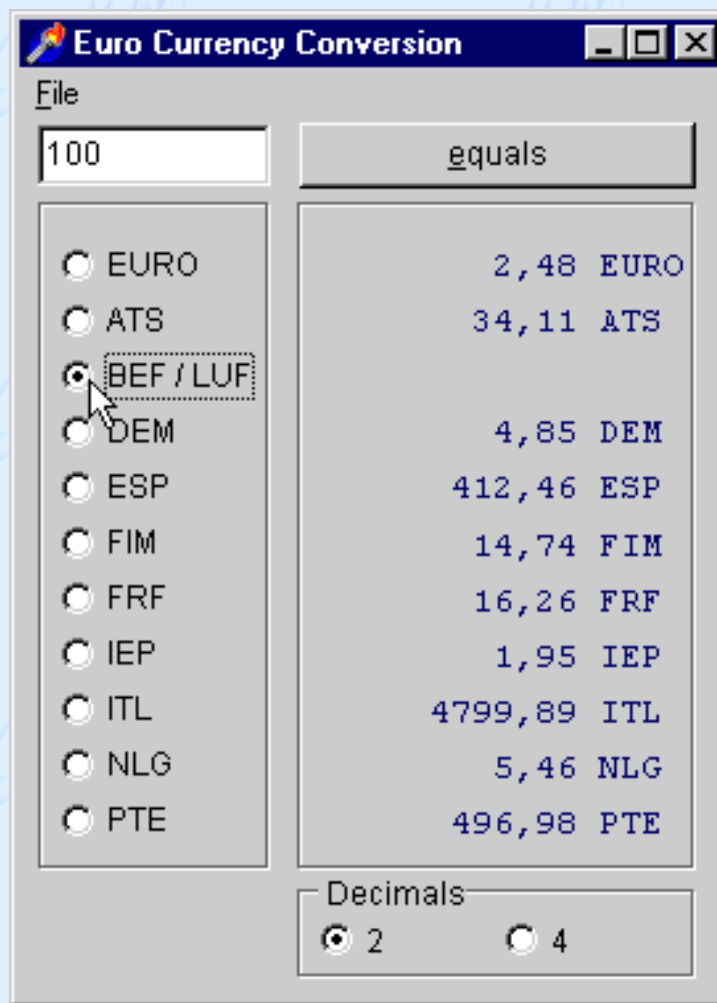
2. Open Delphi's Run menu and click Run.



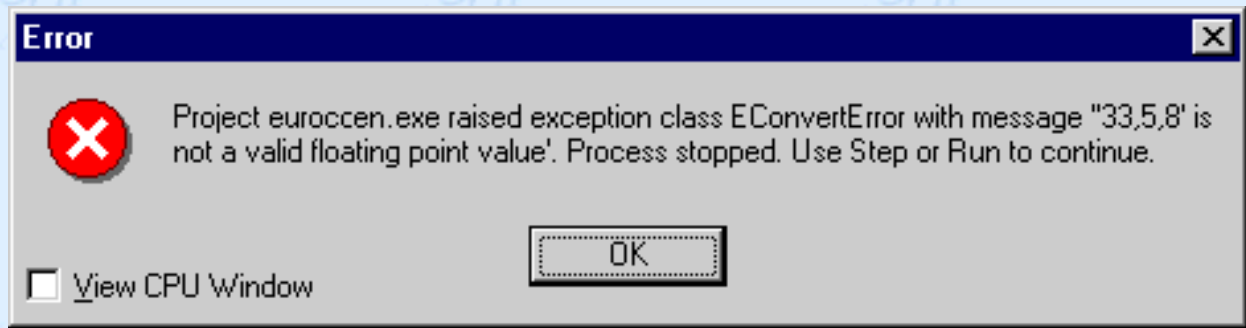
3. Your project is compiled and linked: the result is **euroccen.exe**. Immediately after that, the program **euroccen.exe** is launched. All this happens so fast, that you probably even didn't notice that your source code was compiled into an executable program!



4. Enter a new **valid** number in the **EDIT-box**, for example: 25. Click on the button with the caption "equals". In the right frame, you should see the corresponding amounts in the other European currencies.
5. Click on one of the **radiobuttons** for the starting currency, for example BEF/LUF. Note that the corresponding **label** for Belgian en Luxemburg frank becomes invisible. That's because I found it quite silly if the program should show that 25 BEF = 25 BEF... On top of that, this trick gives a clear visual feedback as to the starting currency.



- Click the radiobutton **4** for the number of decimals. All converted numbers now are shown with 4 decimals (after the *decimal point* or *decimal comma*). This is useful when converting small amounts of certain currencies, e.g. Lire (ITL = Italian Lire). For a display with two decimals, click on the corresponding radiobutton.
- Enter an **invalid** value in the EDIT-box (like 20,5,7 or ABC) and try a conversion. Delphi responds with an error message and pauses the execution of the application: when trying to convert the **text** "20,5,7" or "ABC" to a **number**, Delphi's *Debugger* noted the error condition. The *Debugger* shows **design** errors as well as **runtime** errors!



8. Click the **OK** button in order to close the error-dialogbox. Then press key **F9** (this is the same as the menu-command **Run**). Execution of the program is resumed and you see a second error message. This is the *warning* we programmed ourselves. Only this message will be shown when there is an invalid input when you run the program from *Windows Explorer* or from *My Computer*. Also close this dialogbox by clicking it's **OK** button.



9. Enter a number with a decimal fraction in the EDIT-box. For the decimal separator, you can type **.** (point, dot) as well as **,** (comma): not only 10.5 but also 10,5 is accepted. The program does this by looking at Windows' "Regional Settings" and by replacing the **comma** with a **point**. Or vice versa, of course. This is a golden tip if you make applications for both Anglo-Saxon and non-Anglo-Saxon users. Especially the Americans tend to forget that there are a few countries besides the US of A...
10. Type a number in the EDIT-box and press the ENTER key: just as if you clicked the "equals" button! This gimmick is programmed as follow: if the EDIT-box has the **focus** (meaning: if the text-cursor is visible in this object) and if you press ENTER, a click on the "equals" button is **simulated**.
11. Stop the program via it's menu and Exit, via it's sytem menu, or via it's Close-button (top right of the window).
12. Stop Delphi. If you get the question "Save changes to...?", say no. Because

maybe you changed something in the source code and for the moment we don't want to save these changes.

Euro Currency Convertor as a standalone program

1. Start Windows Explorer and look in the folder where our first project is located. You will see several additional files:
 - **euroccnl.exe** is the executable file
 - **euroform.dcu** (dcu= Delphi compiled unit) is the file that was compiled from the source files euroform.pas (pas= Pascal) and euroform.dfm (dfm= Delphi Form)
 - **euroccnl.dsk** (dsk= Desktop) is the file that remembers how your Delphi-*desktop* looked like when you stopped Delphi. Next time when you open the same project, the positions and dimensions of all the windows will be restored. Nifty!
 - **other** files: if you fooled around with the project. In the next lessons I'll discuss all those Delphi-files in detail.
2. Doubleclick on euroccnl.exe: the program starts.
3. Try again to convert an illegal value. Now, you will only see the **programmed** error message.
4. Stop the program.
5. If you want to see how EuroCC reacts to a different setting of the "decimal separator", you open Windows' Control Panel, you open the *Regional Settings* and you change the "Decimal symbol": enter a **point** if it was a *comma* (or the opposite: enter a **comma** if it was a *point*). Close the window *Regional Settings*.
6. Start EuroCCEN from the Explorer and note that the results are displayed differently: with decimal **points** instead of *comma's* (or the opposite, of course). Try a value with a decimal fraction: also in the EDIT-box this same decimal separator is shown, no matter if you type *point* or *comma*.
7. Stop the EuroConvertor. Afterwards, don't forget to reset your *Regional Settings* back to the way they were.

Navigation

Not the **only** way of course... You can also return to the current subject by means of the **navigation menu** and then work your way back to where you were. But since the browser has a BACK button... ;-)

You can freely [download lessons 1 to 3](#) and the associated source code files.

Members of the [DelphiLand Club](#) can also download the full version of lessons 4 to 10, including all fully commented source code files.

[[TOP](#)]

© Copyright 2000
Studiebureau Festraets

Lesson 2: EuroCC: forms, edits, buttons



In this lesson we'll create a simplified version of **EuroCC**, the application that was introduced in lesson 1.

For now, the accent will not be on what the program does, but rather on how to start a project, how to save the files, add components, and so on. But very soon, we'll continue in a higher gear!



Preparation

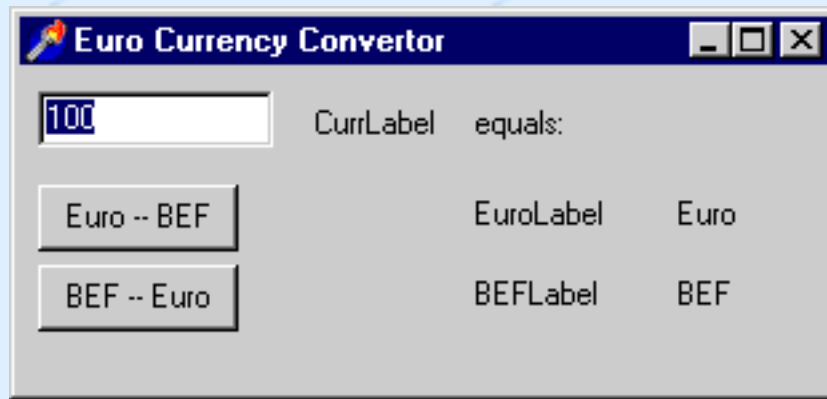
1. If you haven't done so yet, download [euroen01.zip](#) to \DelphiLa.
Hold your horses! Don't unzip it now: every project will get it's own directory, and it is best to keep all the original files in \DelphiLa.
2. Create a new directory: **\DelphiLa\EuroEN01**. Of course, you can choose other names for your directories and projects. But if you use the structure that I propose here, it will be a lot easier to understand each other.
3. "Unzip" euroen01.zip to directory \DelphiLa\EuroEN01. Check if you've got the files euroen01.dpr, euroform.dfm and euroform.pas.



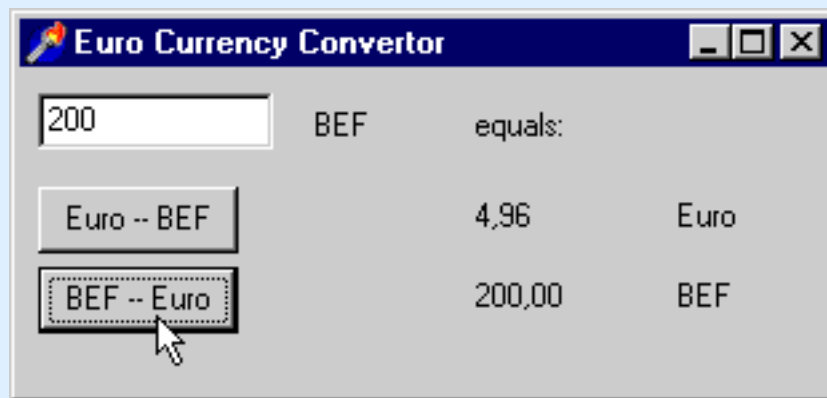
Starting at the end

As usual, we'll start with the end: how should or simplified EuroCC look like and what should it do?

1. Start Delphi and open the project euroen01.dpr through the **File** menu and **Open Project...**
2. Compile the application. If you have forgotten how to do that, have a look at [lesson 1](#).
3. As with every succesful compilation, Delphi will start the application in the "environment" of the debugger.



4. Just play around with the program. Notice that Delphi gives an *error message* and that it pauses the program, if you enter an invalid value and then click a button. To continue, click **Run** in the **Run** menu, or press **F9**. In the message-box that appears, click **OK**. Afterwards you do not get a second error message, because in this simple version of EuroCC we are not checking for a valid input (compare this behaviour with lesson 1).



5. Stop the application. In the **File** menu, click **Close All**. Delete all files from `\DelphiLa\EuroEN01`, using your favorite file manager (Windows Explorer,...).

Your Turn !

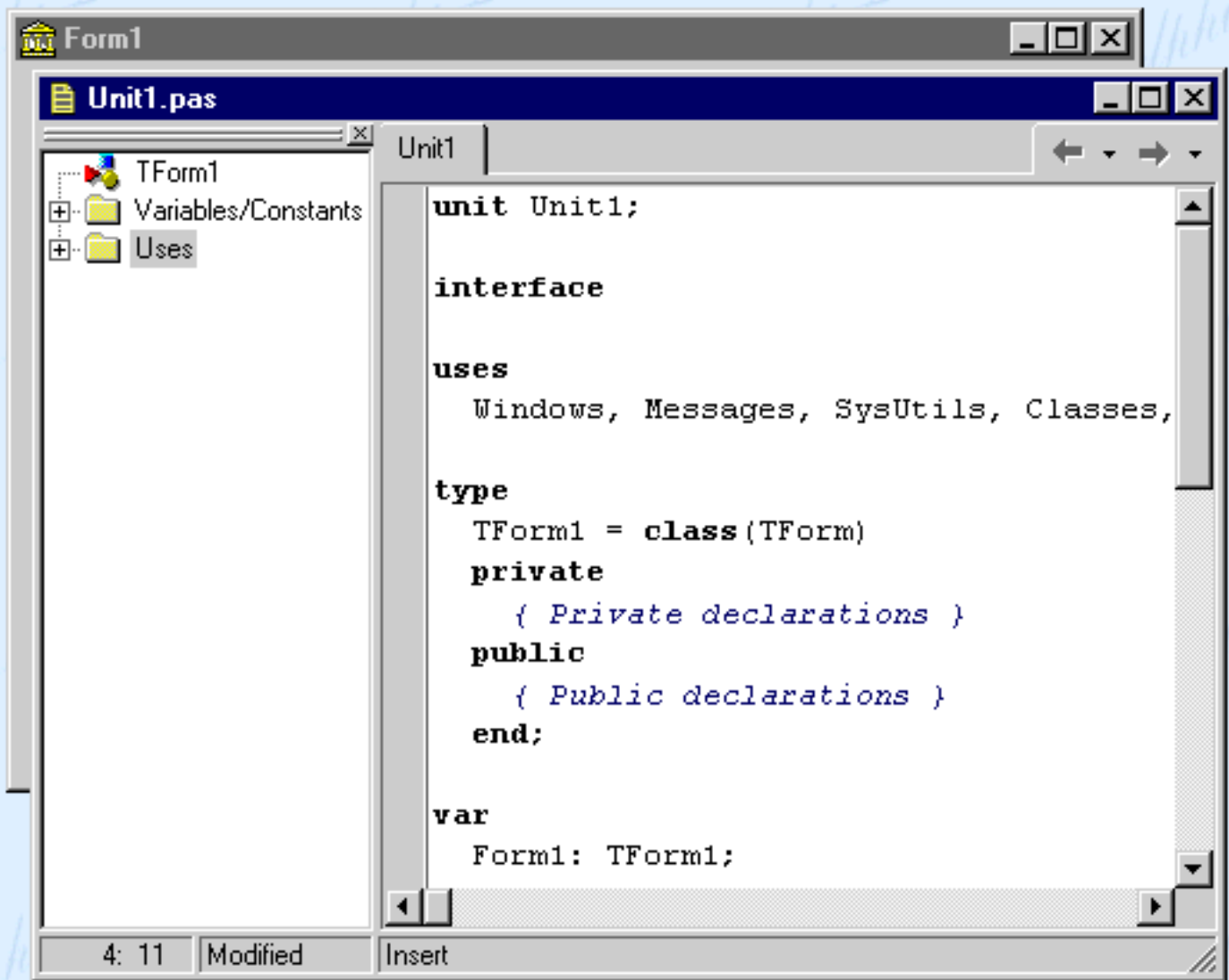
- Open the **File** menu and click **New Application**. Delphi will start a new project:
 - Delphi's title bar shows the name of the new project: **Project1**. This means that Delphi created a dpr-file (project file) called Project1.dpr. Note: if you start several new projects one after the other during the same session, the second project gets the name Project2.dpr, the next one will be Project3,... Delphi consequently gives names along these lines to new files and new components.
 - Notice the window with the title **Form1**. That window (*form*) is the basis for your

project. Delphi-applications are based on *Forms*. Every application has one or more Forms. A Form is a component in the shape of a window. That's why Bill called it "Windows"... On the form you put other components, such as *buttons*, *Edit-boxes*, *Radiobuttons*, *ListBoxes*, *ComboBoxes* and other well known Windows-creatures.

- Behind the window "Form1" you'll find the **Editor** window. That's where the source code can be viewed and edited. The source code is the result of the cooperation between Delphi and yourself: for every unit, Delphi creates a template that you can complete. For now, you'll probably see nothing but the bottom of the Editor window (its status bar).



2. Click the Editor window, e.g. its status bar (the strip with the indication *1: 1 Modified Insert*). The entire Editor window comes into view:



Unit1 is the name of the Unit that goes with Form1. Delphi created the file Unit1.pas, containing the source code for Form1, the main form of the application.

For the moment, all these files only exist in the RAM of your computer. You have to save them to disk files.

3. Open the File menu and click **Save All**.

In the dialog *Save Unit1 as*, select directory **\DelphiLa\EuroEN01**. Enter **euroform** as the file name. Delphi saves the text of the unit in file **euroform.pas**. The other data for Form1 are saved automatically in **euroform.dfm**.

4. In the next dialog, *Save Project1 AS*, the right directory is already selected. Enter **euroen01** as file name. Delphi saves the *composition*-data of the project as **euroen01.dpr** (this file contains the names of the units and the forms that you created, the names of other units used by the project, and so on).

Notice that the title bar of the Editor window has changed to euroform.pas. But where is the project

file, the dpr-file? It isn't visible at this moment, but that isn't important anyhow. Don't change anything in dpr-files if you do not know exactly what you are doing, better leave the management of the project to Delphi.



Files and file naming

It is best to save all files of a new project as soon as possible. That way, you won't experience nasty surprises as lost projects (what was that name again?), or an existing project being overwritten by a new one (and that's a lot worse...)

- You are completely free in naming the first unit (in our case: the only unit).
- Of course, all next units have to be given different names.
- The name of the project file must be different from all unit-names. In our case, the name *euroen01* is not allowed for both the unit and the project, although the file extensions are different!
- Out of the project name, Delphi creates the name for the compiled executable. In our case this will be **euroen01.exe**. Therefore it is worthwhile thinking of a suitable name BEFORE starting a project.



Quick Analysis



Analysing before you start is half of the game.

We know that 1 Euro equals 40.3399 BEF (Belgian frank). BTW: the same is true for the Luxemburg frank).

Clicking a button should convert the value in the edit-box to another currency. Also the name of that currency should be shown.

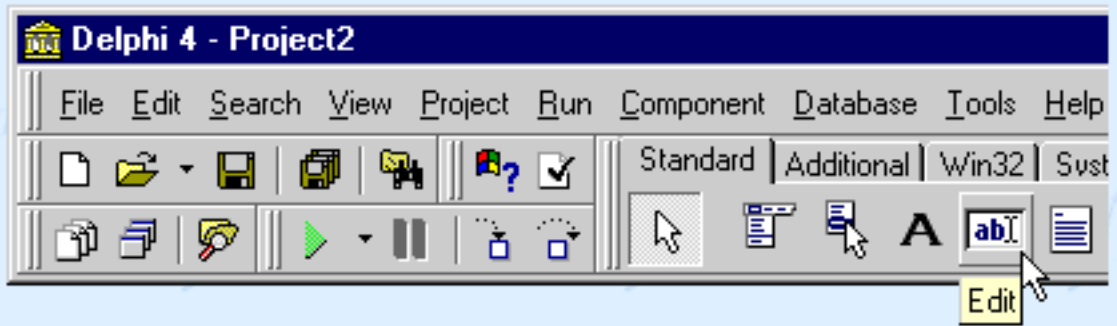
Components for this application: 1 edit-box for data entry, 2 buttons (one for the conversion Euro - Belgian franks, one for the reverse), 6 labels (1 for the name of the starting currency, 1 for the text "equals", 2 for the results, 2 for the currency names).



Adding the Components

The next steps explain how to add the components to the form:

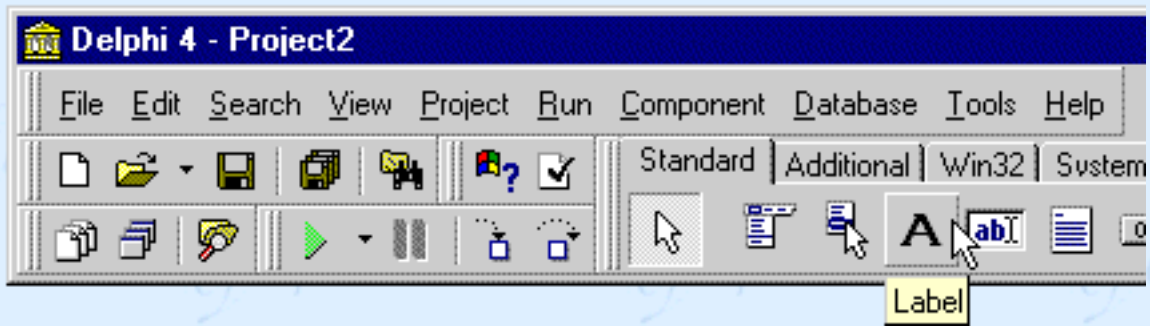
1. Click the form window: it comes to the front. Tip: you also can press **F 12**, which switches between the "unit" in the editor and the "form".
Probably you will see a raster on the form, facilitating the alignment of the components. For reasons of clarity I omitted this raster in the pictures. On my PC, I switched off the form-raster: about everything can be personalized in Delphi. But don't change too much if you only just started with Delphi, because otherwise it would be quite difficult to restore the original settings. In the worst case scenario, you would have to reinstall Delphi :-(((
2. In the component palette, click the icon of the Edit component. It is on the **standard** page of the palette:



3. Click somewhere in the form. An Edit-box appears on the form, with the text Edit1 (this is the name of the component).
4. Drag component Edit1 to the upper left-hand corner of Form1, right below the title bar.



5. Select the Label component from the component palette.



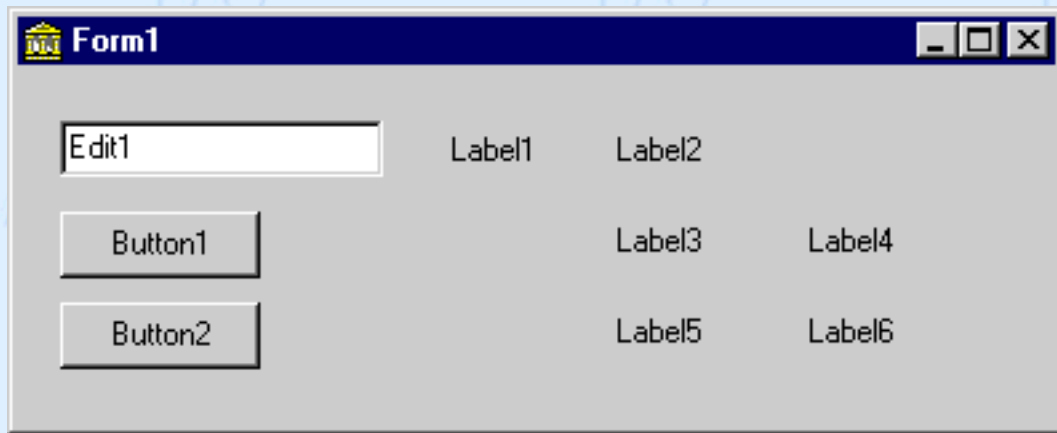
Move the cursor somewhere right of Edit1 and click: Label1 is placed on the form.



6. Once again, select the Label from the component palette. Then, click to the right of Label1. Label 2 is born.
7. Select the Button component from the component palette (also on the **standard** page). add it to the form, somewhere below the Edit-box. A button with the name Button1 appears.



8. Below Button1, add a second button to the form.
9. Add two Labels to the right of Button1 (Label3 and Label4). Add two more labels to the right of Button2 (Label5 and Label6). By now, your form should look similar to this:



10. Drag the components until Form1 more or less matches the picture shown above.

Probably your form is a lot bigger than the one that I show here. Size the form by dragging the bottom left corner of the window's frame, just as you would do with any other window.

11. Time to save your work: in the File menu, select **Save All**.
12. Let's test. In the Run menu, select **Run**. Or press key **F9**.

Delphi will compile and run your program.

The program doesn't seem to do anything, but make no mistake: this is a full blown Windows application! You can size and move the window, enter text in the Edit, click the buttons,...

13. Stop the application.

Every application contains one or more Forms...

But to every rule there is an exception... ;-) Delphi also allows the creation of applications *without forms*.

This is the last word you hear about this, because our lessons are about RAD for *Windows!*

You can freely [download lessons 1 to 3](#).

Members of the [DelphiLand Club](#) can also download the full version of lessons 4 to 10, including all fully commented source code files.

[[TOP](#)]

Lesson 3: Properties

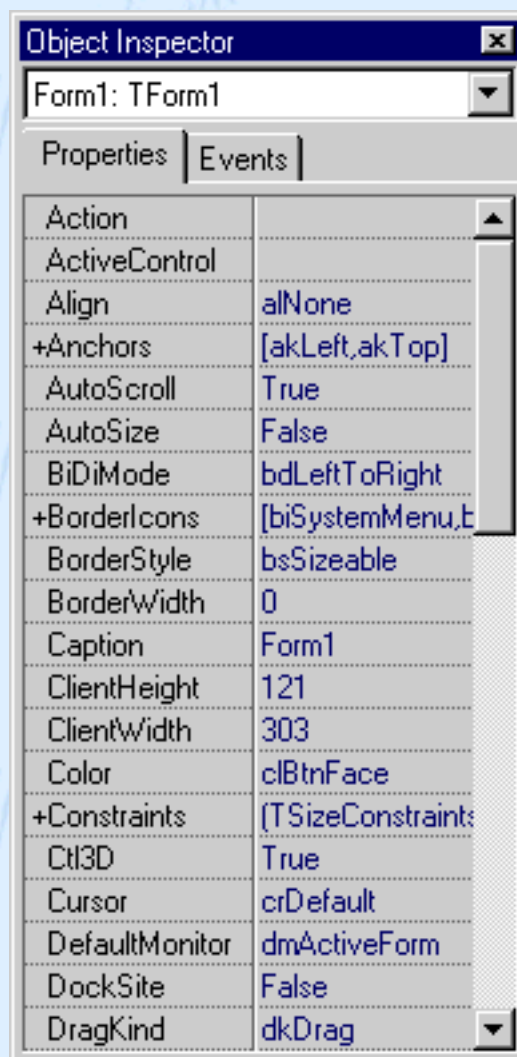


Time to bring some life into our project.

Let's change the **properties** of some of the components: their caption, color, size, font, and so on. And let's add just a little bit of code. Finally, I hear you sigh... isn't that all is programming about...? But Delphi is at least as much about components and their properties!

The Object Inspector: Properties and Events

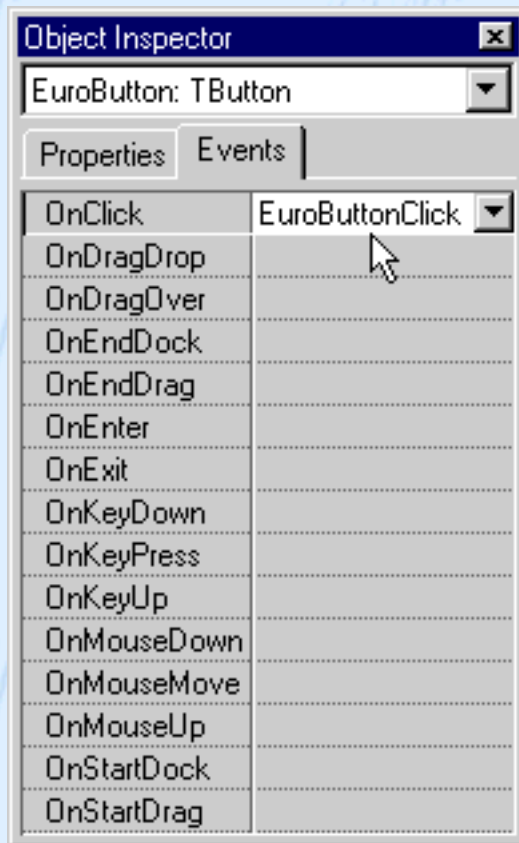
1. In the **Object Inspector** you can change the **properties** of the components at *design time*. If you don't see the Object Inspector, open it thru' the command in the **View** menu, or press key **F11**.



2. Make sure that **Form1** is selected in the Object Inspector, as shown in the picture above (select

Form1 in the Inspector's listbox). Then, click on the **Caption** property and type: **Euro Currency Convertor**. The text in the title bar of the "form" (the window) changes as you type.

3. On the form, click Label2, in order to select it. In the Object Inspector, change the label's **Caption** to **equals** (you can also select component Label2 in the Inspector's listbox). Then, change the Caption of Label4 to **Euro** and change the Caption of Label5 to **BEF**. Change the Caption of Button1 to **Euro -- BEF**. Change the Caption of Button2 to **BEF -- Euro**.
4. The names of our components are not very descriptive. Let's at least change the names of the components that will be used in the program's source code. That is, the components whose properties will be changed or used when the program is running. Select Edit1 and change its' **Name** property to **InputEdit**. Likewise, change the names of the following components:
 - Label1 : **CurrLabel**
 - Label3 : **EuroLabel**
 - Label5 : **BEFLabel**
 - Button1 : **EuroButton**
 - Button2 : **BEFButton**
5. Change the **Text** property of InputEdit to **100**. This assures that the program has a value to start working with, i.e.: "100 currency units".
6. In the Object Inspector, you also indicate the **events** to which a component should respond. Select EuroButton. Next, select the Events tab of the Inspector: you'll see the events OnClick, OnDoubleClick,...
7. Double-click in the white field to the right of the OnClick event. Delphi will write **EuroButtonClick** into this field. That will be the name of the **Event Handler**, the routine that is executed each time EuroButton is clicked.



8. In the Editor, Delphi has already created a template for this Event Handler:

```

procedure TForm1.EuroButtonClick(Sender: TObject);
begin

end;

```

9. Let's experiment for a moment with some (temporary) source code. Between the words **begin** and **end** of the Event Handler, you type the code to be executed when EuroButton is clicked:

```

CurrLabel.Caption := 'Euro';
EuroLabel.Caption := InputEdit.Text;

```

The entire Event Handler should read as follows:

```
procedure TForm1.EuroButtonClick(Sender: TObject);  
begin  
    CurrLabel.Caption := 'Euro';  
    EuroLabel.Caption := InputEdit.Text;  
end;
```

Attention! The word *Euro* is between **single** quotes, not double quotes!

10. Compile and run the program. Type something in the Edit-box and click the EuroButton: the text 'Euro' appears in CurrLabel and the text of the Edit-box appears in EuroLabel. In Delphi-language:
 - the **string** 'Euro' is assigned to the property **Caption** of CurrLabel;
 - the property **text** of InputEdit is assigned to the property **Caption** of EuroLabel.
11. Stop the application and save the project (Save All).

Well, I hope you're beginning to see the idea behind this tutorial: getting you up and running in as little time as possible -- practice first, and then theory. At first, I'll show you what Delphi *DOES*. And when you're becoming too curious, I'll tell you the *why* and the *how*. Sounds fair? Surely sounds fun, I hope. So, up to [lesson 4](#), where we shall make things happen in *runtime*: let's code!

You can freely [download lessons 1 to 3](#) and the associated source code files.

Members of the [DelphiLand Club](#) can also download the full version of lessons 4 to 10, including all fully commented source code files.

[[TOP](#)]

© Copyright 2000
Studiebureau Festraets

Lesson 4: Let's code !

Our Euro Currency Converter still doesn't do anything useful. In this lesson, we'll add some real code to the event handlers.

In lesson 3 we've put some temporary code in the event handler of EuroButton. But what should this event handler *really* do?

- the text "Euro" should be shown in CurrLabel.
 - the **text** of the InputEdit should be converted to a **numerical value**.
 - From this number (the value in Euro) the value in BEF should be calculated (multiply it with 40.3399).
 - This number is converted again to 'text', which is shown in BEFLabel.
 - Finally, the text of the InputEdit should also be shown in EuroLabel.
-

Lesson 5: EuroCC with RadioButtons

In this version of EuroCC, we'll replace the *Buttons* with **RadioButtons**.

We'll also introduce subroutines, which constitute the power of every real programming language.

Lesson 6: Making decisions: If... Then... Else

To make decisions, a program has to determine whether a condition is TRUE or FALSE. This is done in a **conditional statement**, where the program flow splits up in two or more "branches".

In lesson 6, we talk about the **IF** statement.

Lesson 7: Error handling in EuroCC

In the final version of **EuroCC** we add some whistles and bells. An extra radiogroup allows setting the decimal fraction to 2 or to 4 digits. And we 'll handle the errors, in case some invalid value is entered.

In this lesson, we introduce the **Case** instruction, that replaces a number of nested *if...then...else...* statements.

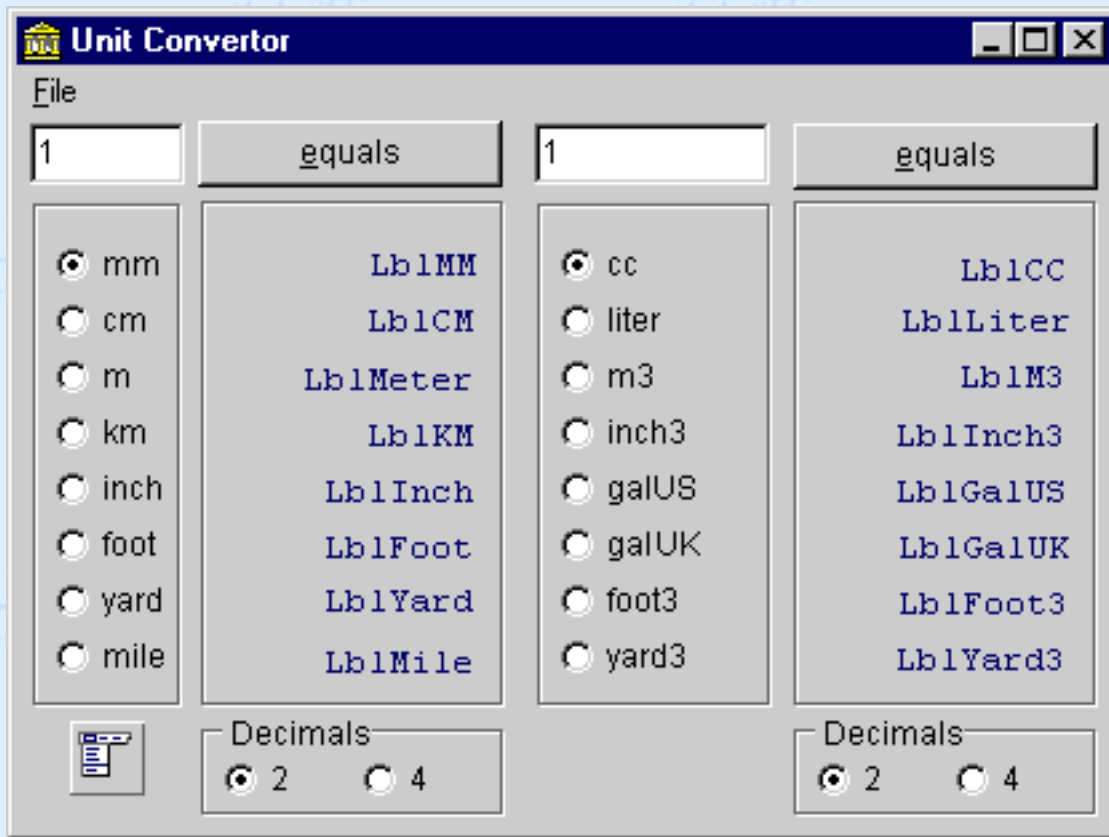
Lesson 8: EuroCC completed

By making our EuroCC program look at Windows' decimal separator, we "internationalize" it, so the user won't get an error in typing the wrong decimal separator. If necessary, a comma will be converted to a point, and vice versa.

And last but not least, we add a menu. Kind of kinky, adding the menu right at the end ;-)

Lesson 9: Units Converter

Let's make some exercises. In lesson 9, you make a general **Units Converter**, based on the "EuroCC" currency convertor. This application should convert length units (cm, meter, inch,...), volumes (liter, gallon,...) or whatever you want it to convert.

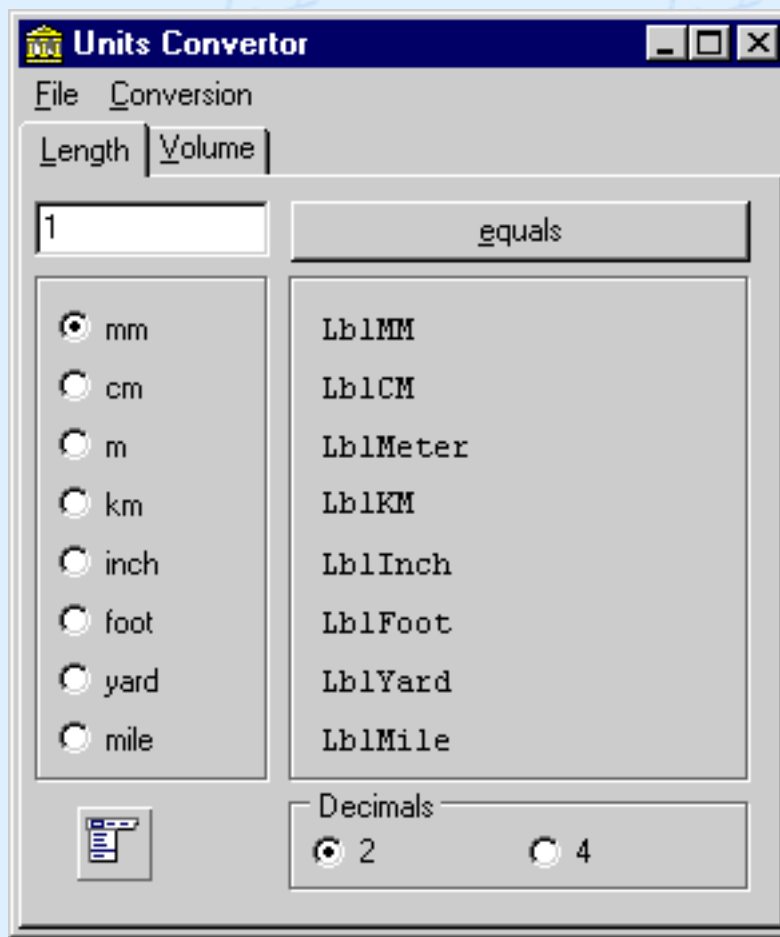


We give you a template project in file [unicoen1.zip](#), where you have to fill in most of the coding. Afterwards, you can compare with the solution contained in [unicoen2.zip](#).

Next, we'll discuss some ways to make the source code more compact and "structured". The source code for this part of the lesson is available as [unicoen3.zip](#)

Lesson 10: Units Converter revisited

In lesson 10 we create an extended version of the Units Converter, using a **TabControl**.



You can download all of the lessons and their associated source code files, if you are a member of the [DelphiLand Club](#).

Your membership allows you to download all of the lessons of this tutorial, plus the lessons of new tutorials, plus the mini-tutorials, function library, and so on...!

© Copyright 2000
Studiebureau Festraets