BLAISE PASCAL MAGAZINE 90

Object Pascal / Internet / JavaScript / WebAssembly / Pas2Js / Databases
CSS Styles / Progressive Web Apps
Android / IOS / Mac / Windows & Linux

Blaise Pascal

BLAISE PASCAL MAGAZINE 90

Object Pascal / Internet / JavaScript / WebAssembly / Pas2Js / Databases
CSS Styles / Progressive Web Apps
Android / IOS / Mac / Windows & Linux

Blaise Pascal

CONTENT

ARTICLES

ADVERTISERS

Niklaus Wirth

Pascal is an imperative and procedural programming language, which Niklaus Wirth designed in 1968–69 and published in 1970, as a small, efficient language intended to encourage good programming practices using structured programming and data structuring. A derivative known as Object Pascal designed for object-oriented programming was developed in 1985. The language name was chosen to honour the Mathematician, Inventor of the first calculator: Blaise Pascal (see top right).

Publisher: PRO PASCAL FOUNDATION in collaboration © Stichting Ondersteuning Programmeertaal Pascal

Stephen Ball
http://delphiaball.co.uk
@DelphiABall

Peter Bijlsma -Editor
peter @ blaisepascal.eu

Dmitry Boyarintsev
dmitry.living @ gmail.com

Michaël Van Canneyt,
michael @ freepascal.org

Marco Cantù
www.marcocantu.com
marco.cantu @ gmail.com

David Dirkse
www.davdata.nl
E-mail: David @ davdata.nl

Benno Evers
b.evers
@ everscustomtechnology.nl

Bruno Fierens
www.tmssoftware.com
bruno.fierens @ tmssoftware.com

Holger Flick
holger @ flixments.com

Primož Gabrijelčič
www.primoz @ gabrijelcic.org

Mattias Gärtner
nc-gaertnma@netcologne.de

Peter Johnson
http://delphidabbler.com
delphidabbler @ gmail.com

Max Kleiner
www.softwareschule.ch
max @ kleiner.com

John Kuiper
john_kuiper @ kpnmail.nl

Wagner R. Landgraf
wagner @ tmssoftware.com

Vsevolod Leonov
vsevolod.leonov@mail.ru

Andrea Magni www.andreamagni.eu
andrea.magni @ gmail.com
www.andreamagni.eu/wp

Paul Nauta PLM Solution Architect
CyberNautics
paul.nauta @ cybernautics.nl

Kim Madsen
www.component4developers

Boian Mitov
mitov @ mitov.com

Jeremy North
jeremy.north @ gmail.com

Detlef Overbeek - Editor in Chief
www.blaisepascal.eu
editor @ blaisepascal.eu

Howard Page Clark
hdpc @ talktalk.net

Heiko Rompel
info @ rompelsoft.de

Wim Van Ingen Schenau -Editor
wisone @ xs4all.nl

Peter van der Sman
sman @ prisman.nl

Rik Smit
rik @ blaisepascal.eu

Bob Swart
www.eBob42.com
Bob @ eBob42.com

B.J. Rao
contact @ intricad.com

Daniele Teti
www.danieleteti.it
d.teti @ bittime.it

Anton Vogelaar
ajv @ vogelaar-electronics.com

Robert Welland
support @ objectpascal.org

Siegfried Zuhr
siegfried @ zuhr.nl

Editor - in - chief
Detlef D. Overbeek, Netherlands Tel.: Mobile: +31 (0)6 21.23.62.68
News and Press Releases email only to **editor@blaisepascal.eu**

Editors
Peter Bijlsma, W. (Wim) van Ingen Schenau, Rik Smit

Correctors
Howard Page-Clark, Peter Bijlsma

Trademarks All trademarks used are acknowledged as the property of their respective owners.

Caveat Whilst we endeavour to ensure that what is published in the magazine is correct, we cannot accept responsibility for any errors or omissions. If you notice something which may be incorrect, please contact the Editor and we will publish a correction where relevant.

Subscriptions (2019 prices)

	Internat. excl. VAT	Internat. incl. 9% VAT	Shipment
Printed Issue ±60 pages	€ 250	€ 261,60	€ 85,00
Electronic Download Issue 60 pages	€ 60	€ 65,40	———
Printed Issue inside Holland (Netherlands) ±60 pages	———	€ 200,00	€ 60,00

Member and donator of WIKIPEDIA
Member of the **Royal Dutch Library**

Subscriptions can be taken out online at **www.blaisepascal.eu** or by written order, or by sending an email to **office@blaisepascal.eu**
Subscriptions can start at any date. All issues published in the calendar year of the subscription will be sent as well.
Subscriptions run 365 days. Subscriptions will not be prolonged without notice. Receipt of payment will be sent by email.
Subscriptions can be paid by sending the payment to:
ABN AMRO Bank Account no. 44 19 60 863 or by credit card or Paypal
Name: Pro Pascal Foundation-Foundation for Supporting the Pascal Programming Language (Stichting Ondersteuning Programeertaal Pascal)
IBAN: NL82 ABNA 0441960863 BIC ABNANL2A VAT no.: 81 42 54 147 (Stichting Programmeertaal Pascal)
Subscription department
Edelstenenbaan 21 / 3402 XA IJsselstein, The Netherlands
Mobile: + 31 (0) 6 21.23.62.68 **office@blaisepascal.eu**

From your editor

In this issue I have tried to introduce some new programs, that can be of great influence for your programming skills:
Fast Report
– which makes reporting extremely easy, Database Workbench
- helping, designing and creating Databases and MMX code
– a free tool, an extremely great tool for coding.
I was also surprised by the large number of books that now were published and the last member of this sprout:
Marco Cantù's book – free as download – the Object Pascal Handbook, and of course not to forget the Lazarus Handbook from Michael Canneyt and others.

That is all very good news.

Because of the vaccination program that has started by now it might be possible to setup again some events which we all long for…
to meet and have fun again together.
It should be possible in the coming year, and I am looking forward to it.
I have some plans but want to be a little more secure about how things will evolve.
But you know: an event planning takes months and there are quite some unknown issues.

What I am sure of for the next year is that Michael van Canneyt will come with something for Pascal which will give anyone the opportunity to build their own VS Code and Atom examples. Which is of course open source.
It will become embedded in the Lazarus environment and the first examples will be available in the next issue.
In the same issue 91 we plan to let you know the roadmap of FPC/Lazarus and there are some overwhelming items.
Hopefully we all can go to 2021 as a year with greater confidence. It might be possible that in 2021 the issue number 100 will be published.
But that's far ahead.

We all have had a lesson in humility.
Lets be careful . Lets be optimistic…

From our Technical advisor: Cartoons from Jerry King

"All I got was a giant lump of coal.
I knew I should've never spammed santa."

BOOK REVIEW:
The Delphi 10.4 Sydney Edition of the Object Pascal Handbook

Marco's blog explaines:
I've finished working on a complete update of my book on the Delphi Object Pascal language, covering all of the new features in the 10.4 release.

Over the last few months, I've been working on a new edition of my Object Pascal Handbook. The first and only printed edition was done for 10 Seattle, and there was a PDF-only update for Berlin. Over Delphi 10.3 and 10.4 the language saw significant changes, from inline variables to custom managed records and the removal of ARC for mobile. For this reason, the book was starting to have gaps and show inconsistencies: it was about time to review it.

So I've done a complete editing and also a good cleanup of the source code of the companion demos, adding some new ones (and removing a few). The changes are not only for new features, but also overall improvements to the book flow and content.

The new book is a complete guide of the Delphi language in Sydney, and would help newcomers learn the language and developers with some experience in Delphi understand what changed over recent releases -- although the book is not organized in a chronological order, but it progresses through features.

The Complete Guide to the Object Pascal programming language for Delphi developers, updated to version 10.4 Sydney

Author: Marco Cantù
ISBN-coming
Pages: 570
Second edition: December 2020.
The ebook (in PDF) draft is currently available as part of a promotion by Embarcadero Technologies. The printed book will be available shortly.
The **code repository** for the application projects discussed in the book is at `https://github.com/MarcoDelphiBooks/ObjectPascalHandbook104`.

part i: foundations
01: coding in pascal
02: variables and data types
03: language statements
04: procedures and functions
05: arrays and records
06: all about strings
part ii: oop in
object pascal
07: objects
08: inheritance
09: handling exceptions
10: properties and events
11: interfaces
12: manipulating classes
13: objects and memory
part iii: advanced features
14: generics
15: anonymous methods
16: reflection and attributes
17: tobject and the system unit in closing
18: other core rtl classes

a: the evolution of object pascal
b: glossary
c: index

Generic Key-Value Pairs

As a first example of a generic class, I've implemented a key-value pair data structure. The first code snippet below shows the data structure written in a traditional fashion, with an object used to hold the value:

```
type
  TKeyValue = class
  private
    FKey: string;
    FValue: TObject;
    procedure SetKey(const Value: string);
    procedure SetValue(const Value: TObject);
  public
    property Key: string read FKey write SetKey;
    property Value: TObject read FValue write SetValue;
  end;
```

To use this class you can create an object, set its key and value, and use it, as in the following snippets of various methods of the main form of the `KeyValueClassic` application project:

```
// FormCreate
Kv := TKeyValue.Create;

// Button1Click
Kv.Key := 'mykey';
Kv.Value := Sender;

// Button2Click
Kv.Value := self; // the form

// Button3Click
ShowMessage('[' + Kv.Key + ',' + Kv.Value.ClassName + ']');
```

What if you need a similar class, holding an Integer rather than an object? Well, either you make a very unnatural (and dangerous) type cast, or you create a new and separate class to hold a string key with a numeric value. Although copy and paste of the original class might sound a solution, you end up with two copies for a very similar piece of code, you are going against good programming principles... and you'll have to update with new features or correct the same bugs two, or three or twenty times.

Generics make it possible to use a much broader definition for the value, writing a single generic class. Once you instantiate the key-value generic class, it becomes a specific class, tied to a given data type. So you still end up with two, or three, or twenty classes compiled into your application, but you have a single source code definition for all of them, still replying on proper string type checking and without a

runtime overhead. But I'm getting ahead of myself: let's start with the syntax used to define the generic class:

```
type
  TKeyValue<T> = class
  private
    FKey: string;
    FValue: T;
    procedure SetKey(const Value: string);
    procedure SetValue(const Value: T);
  public
    property Key: string read FKey write SetKey;
    property Value: T read FValue write SetValue;
  end;
```

In this class definition, there is one unspecified type, indicated by the placeholder `T`, placed within angle brackets. The symbol `T` is frequently used by convention, but as far as the compiler is concerned you can use just any symbol you like. Using `T` generally makes the code more readable when the generic class uses only one parametric type; in case the class needs multiple parametric types it is common to name them according to their actual role, rather than using a sequence of letters (`T`, `U`, `V`) as it happened in C++ during the early days.

note "T" has been the standard name, or placeholder, for a generic type since the days the C++ language introduced *templates* in the early 1990s. Depending on the authors, the "T" stands for either "Type" or "Template type".

The generic `TKeyValue<T>` class uses the unspecified type as the type of one of its two fields, the property value, and the setter method parameter. The methods are defined as usual, but notice that regardless of the fact they have to do with the generic type, their definition contains the complete name of the class, including the generic type:

```
procedure TKeyValue<T>.SetKey(const Value: string);
begin
  FKey := Value;
end;

procedure TKeyValue<T>.SetValue(const Value: T);
begin
  FValue := Value;
end;
```

To use the class, instead, you have to fully qualify it, providing the actual type of the generic type. For example, you can now declare a key-value object hosting buttons as values by writing:

```
var
  Kv: TKeyValue<TButton>;
```

Marco Cantù, Object Pascal Handbook 10.4

The full name is required also when creating an instance, because this is the actual type name (while the generic, uninstantiated type name is like a type construction mechanism).

Using a specific type of the value of the key-value pair makes the code much more robust, as you can now only add `TButton` (or derived) objects to the key-value pair and can use the various methods of the extracted object. These are some snippets from the main form of the `KeyValueGeneric` application project:

```
// FormCreate
Kv := TKeyValue<TButton>.Create;

// Button1Click
Kv.Key := 'mykey';
Kv.Value := Sender as TButton;

// Button2Click
Kv.Value := Sender as TButton; // was "self"

// Button3Click
ShowMessage ('[' + Kv.Key + ',' + Kv.Value.Name + ']');
```

When assigning a generic object in the previous version of the code we could add either a button or a form, now we can use only button, a rule enforced by the compiler. Likewise, rather than a generic `kv.Value.ClassName` in the output we can use the component `Name`, or any other property of the `TButton` class.

Of course, we can also mimic the original program by declaring the key-value pair with an object type, like:

```
var
  Kvo: TKeyValue<TObject>;
```

In this version of the generic key-value pair class, we can add any object as value. However, we won't be able to do much on the extracted objects, unless we cast them to a more specific type. To find a good balance, you might want to go for something in between specific buttons and any object, requesting the value to be a component:

```
var
  Kvc: TKeyValue<TComponent>;
```

You can see corresponding code snippets in the same `KeyValueGeneric` application project. Finally, we can also create an instance of the generic key-value pair class that doesn't store object values, but rather plain integers:

```
var
  Kvi: TKeyValue<Integer>;
begin
  Kvi := TKeyValue<Integer>.Create;
  try
    Kvi.Key := 'object';
    kvi.Value := 100;
```

Marco Cantù, Object Pascal Handbook 10.4

```
    Kvi.Value := Left;
    ShowMessage ('[' + Kvi.Key + ',' +
      IntToStr (Kvi.Value) + ']');
  finally
    Kvi.Free;
  end;
```

Inline Variables and Generics Type Inference

When you are declaring a variable of a generic type, the declaration can be fairly long. As you create an object of that type, you have to repeat the same declaration. That is, unless you take advantage of inline variable declarations and their ability to infer the variable type. The last code fragment above can be written as:

```
begin
  var Kvi := TKeyValue<Integer>.Create;
  try
    ...
```

In this code you don't have to repeat the full generic type declaration twice. This is particularly handy when using containers, as we'll see later.

Type Rules on Generics

When you declare an instance of a generic type, this type gets a specific version, which is enforced by the compiler in all subsequent operations. So if you have a generic class like:

```
type
  TSimpleGeneric<T> = class
    Value: T;
  end;
```

as you declare a specific object with a given type, you cannot assign a different type to the `Value` field. Given the following two objects, some of the assignments below (part of the `TypeCompRules` application project) are incorrect:

```
var
  Sg1: TSimpleGeneric<string>;
  Sg2: TSimpleGeneric<Integer>;
begin
  Sg1 := TSimpleGeneric<string>.Create;
  Sg2 := TSimpleGeneric<Integer>.Create;

  Sg1.Value := 'foo';
  Sg1.Value := 10; // Error
  // E2010 Incompatible types: 'string' and 'Integer'
```

maXbox Starter 78 Image Classifier

Author: Max Kleiner

The Portable Pixmap Format

Today I want to introduce a system independent image format for any images, machine learning data like feature maps in a **CNN (Convolutional Neural Network)** or just to exchange picture data on various platforms.

The **Portable PixMap** format uses an uncompressed and inefficient format so that it is seldom used for storing large images but on the other side this is an advantage.
We use this format for converting system-independent images to machine learning feature maps in a **CNN.**

Besides **PPM,** other 2 popular **Netpbm** file formats include the portable bitmap format `(.PBM)` and the portable graymap format `(.PGM)`.
Sometimes, they are also collectively referred to the portable anymap format `(*.PNM)`.

- PBM is for bitmaps (black and white, no grays).
- PGM is for grayscale (0–255 or 0–65535).
- PPM is for "pixmaps" which means full color.

Figure 1: this is a Conversion from BMP to PPM to JPG

One of the image file formats that is developed by the Netpbm project. For cross-platform purpose, the images are encoded using a text format as plain. `PPM` files specify the color of the pixel using a number from 0 to 65536 (or 0–255 for each channel), in addition to the image resolution height, width and so on.

For details go to
`https://en.wikipedia.org/wiki/Netpbm#File_formats`

So beside the plain format also a raw (binary) format exists. The ASCII ("plain") formats (P1-P3) allow for human readability and easy transfer to other platforms; the binary ("raw") formats (P4-P6) are more efficient in file size but may have native byte-order issues. This is how we define the header first:

```
Header:= Format('P6'#10'%d %d'#10'255'#10,
[bmp.Width,bmp.Height]);
```

In my opinion the PPM format was devised to be an intermediate format for use in developing file format conversion systems. The PPM idea is that we have one format that any other format can be converted into.
As I said The PPM format is not intended to be an archival format, so it does not need to be too storage efficient. Thus, it is one of the simplest formats as a common denominator.

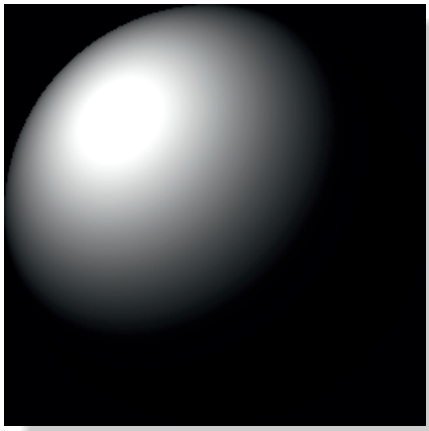


*Figure 2: PPM to PNG with 251*251 Dimension*

Let's start with the code.
Each file opens with a two-byte magic number (in ASCII) that identifies the type of file, the magic number is either an ASCII character string "P1", "P2","P3","P4", "P5" or "P6" depending upon the storage method used.

"P1" and "P4" indicate that the image data is in a bitmap as black and white.
We use a pattern helper class as a procedure() to save the bitmap as a PPM file: (*next page* ➔)

```
procedure TBitmapHelperSaveAsPPM_4(FileName: TFileName; bmp: TBitmap; useGrayScale: Boolean);
var
 i,j: Integer;
 Header: AnsiString;
 ppm: TMemoryStream;
 agb: TBytes;
```

We use a memory-stream and a byte-array for storage. Using a byte array can lead to some optimizations that can make accessing and changing information in the array faster than it would be with arrays of other types, in our case we define a 3 bytes array:

```
begin
 ppm:= TMemoryStream.Create;
 try
  Header:= Format('P6'#10'%d %d'#10'255'#10,[bmp.Width, bmp.Height]);
  writeln(Header);
  ppm.WriteBuffer((Header), Length(Header));
  setlength(agb,3)
  for i:= 0 to bmp.Width- 1 do
   for j:= 0 to bmp.Height- 1 do begin
     if useGrayScale then
      agb:= InttoBytes(ColorToGray(ColorToRGB(bmp.Canvas.Pixels[j,i])))
     else
      agb:= InttoBytes(ColorToRGB(bmp.Canvas.Pixels[j,i]));
     ppm.Write(stringOf(agb), 3);
     //ppm.Write(BytetoString(rgb),3);
   end;
  ppm.SaveToFile(FileName);
 finally
  ppm.Free;
 end;
end;
```

```
procedure TBitmapHelperSaveAsPPM_4(FileName: TFileName; bmp: TBitmap; useGrayScale: Boolean);
var
 i,j: Integer;
 Header: AnsiString;
 ppm: TMemoryStream;
 agb: TBytes;
```

We use a memory-stream and a byte-array for storage. Using a byte array can lead to some optimizations that can make accessing and changing information in the array faster than it would be with arrays of other types, in our case we define a 3 bytes array:

```
begin
 ppm:= TMemoryStream.Create;
 try
  Header:= Format('P6'#10'%d %d'#10'255'#10,[bmp.Width, bmp.Height]);
  writeln(Header);
  ppm.WriteBuffer((Header), Length(Header));
  setlength(agb,3)
  for i:= 0 to bmp.Width- 1 do
   for j:= 0 to bmp.Height- 1 do begin
     if useGrayScale then
      agb:= InttoBytes(ColorToGray(ColorToRGB(bmp.Canvas.Pixels[j,i])))
     else
      agb:= InttoBytes(ColorToRGB(bmp.Canvas.Pixels[j,i]));
     ppm.Write(stringOf(agb), 3);
     //ppm.Write(BytetoString(rgb),3);
   end;
  ppm.SaveToFile(FileName);
 finally
  ppm.Free;
 end;
end;
```

A PPM file consists of two parts, a header and the image data stored within `WriteBuffer()`. The header consists of at least three parts normally delineated by carriage returns and/or linefeeds but the PPM specification only requires white space.

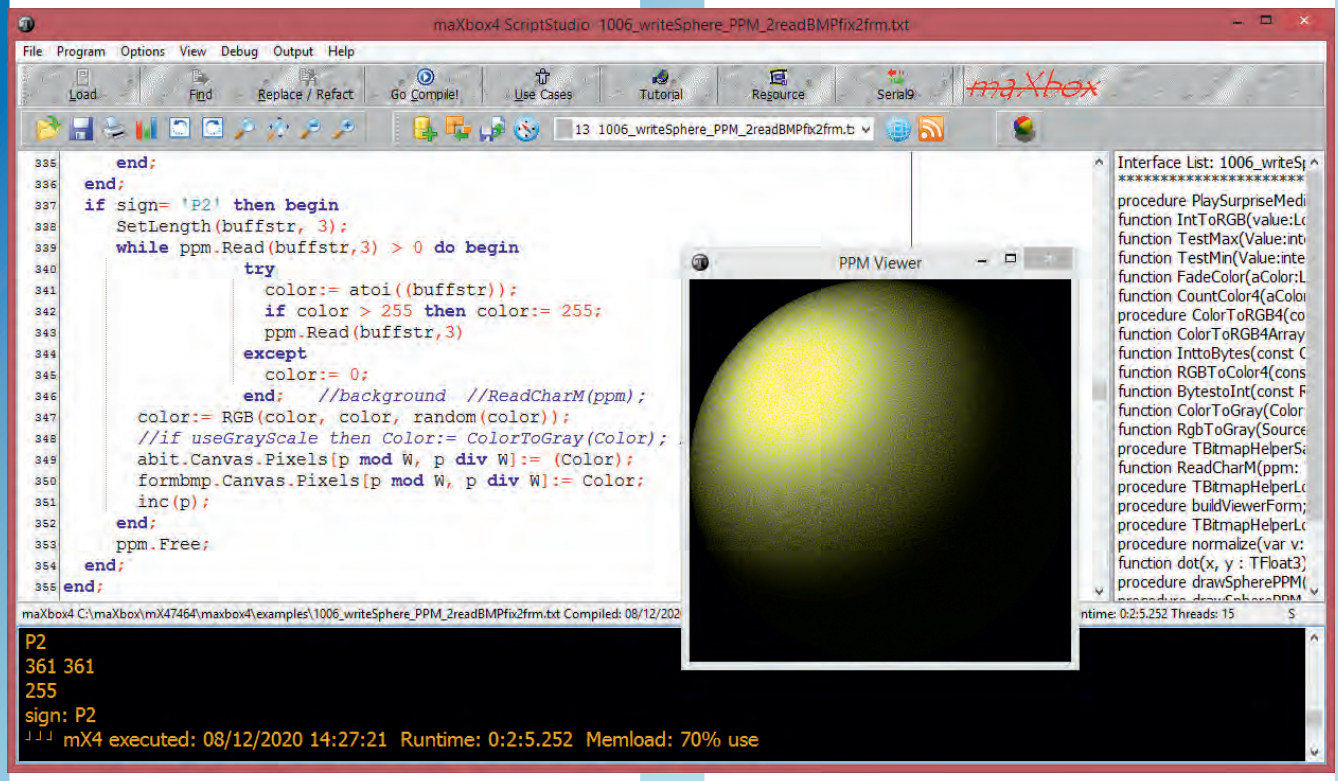


Figure 3: maXbox viewer on a simple canvas

Interesting is a grayscale function as boolean decision `useGrayScale`:

```
function ColorToGray(Color: TColor): TColor;
var L: Byte;
begin
 L:= round(0.2126*GetRValue(Color)+0.7152*GetGValue(Color)
+0.0722 * GetBValue(Color));
 Result:= RGB(L, L, L);
end;
```

When converting from RGB to grayscale, it is said that specific weights to channels R,G, and B should be applied.
These weights are: 0.2989, 0.5870, 0.1140., but I changed these weights as an optimized discriminator for feature maps.

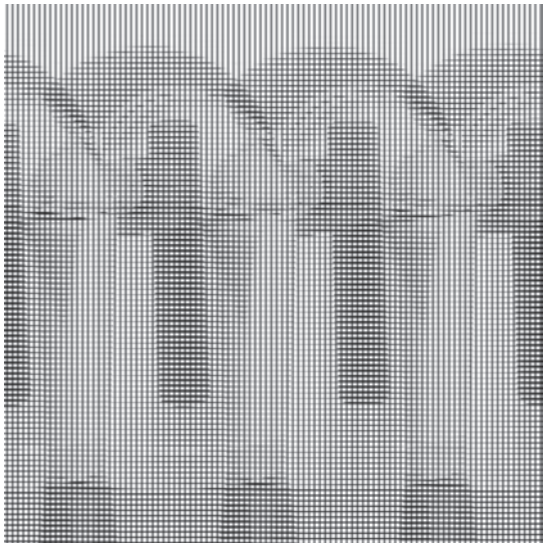


Figure 4: Explains weights of grayscale

It is said that the reason for this is different human perception or sensibility towards these three colors, anyway it shows “tons” of different methods to generate grayscale images with different outcomes!
By the way you can test your PPM online:
`http://paulcuth.me.uk/netpbm-viewer/`
Just drag your PPM, PGM or PBM files onto the dashed area below the site to convert them to PNG images in a browser.
After convert bitmaps to PPM we step to the viewer to be independent of another app, for example with a P6 signature:
P6 - 361 * 361
Example bitmap
of resolution 361 with 255 colors
on RGB channels 255

The last part of the header gives the maximum value of the colour components for the pixels, this allows the format to describe more than some single byte(0..255) colour values.

```
if sign = 'P6' then begin
  SetLength(buffstr, 3);
  while ppm.Read(buffstr, 3) > 0 do
  begin
   //color:= Bytestoint(bytesof(buffstr))
   color:= RGB(ord(buffstr[1]),ord(buffstr[2]),ord(buffstr[3]))
   if useGrayScale then  Color := (ColorToGray(Color));
   abit.Canvas.Pixels[p mod W, p div W] := Color;
   inc(p);
  end;
end;
```

While not required by the format specification it is a standard convention to store the image in top to bottom, left to right order. Each pixel is stored as a byte, value 0 = black, value 255 = white. The components are stored in the usual order, Red - Green - Blue.

```
if sign = 'P2' then begin  SetLength(buffstr, 3);
  while ppm.Read(buffstr,3) > 0 do begin
        try
         color:= atoi((buffstr));
         if color > 255 then color:= 255;
         ppm.Read(buffstr,3)
        except
         color:= 0;
        end;  //background  //ReadCharM(ppm);
   color:= RGB(random(color), (color), (color));
   //if useGrayScale then Color:= ColorToGray(Color); //dark mode
   abit.Canvas.Pixels[p mod W, p div W]:= (Color);
   inc(p);
  end;
  ppm.Free;
end;
```

You can find also in the script the `drawSpherePPM_Save Procedure` to generate a PPM from the Scratch. Now at last this is our process for object detection:
an image to detect in the input folder, e.g.: `manmachine.ppm` then extract the feature map and mark the probability to convert back as a `PNG`. It goes on with the declaration of the created paths:

```
model_path = "./models/yolo-tiny.h5"
input_path = "./input/manmachine.ppm"
output_path = "./output/manmachineout.png"
```

Figure 5: Object Detection from PPM Intermediate

To detect only some of the objects above, I will need to call the `CustomObjects` method and set the name of the object(s) we want to detect to.
Netpbm contains over 220 separate programs in the package, most of which have "pbm", "pgm", "ppm", "pam", or "pnm" in their names. The programs are designed to be minimal building blocks that can be used in various combinations to do other things like image detection on an **Arduino** with **TensorFlowLite.**

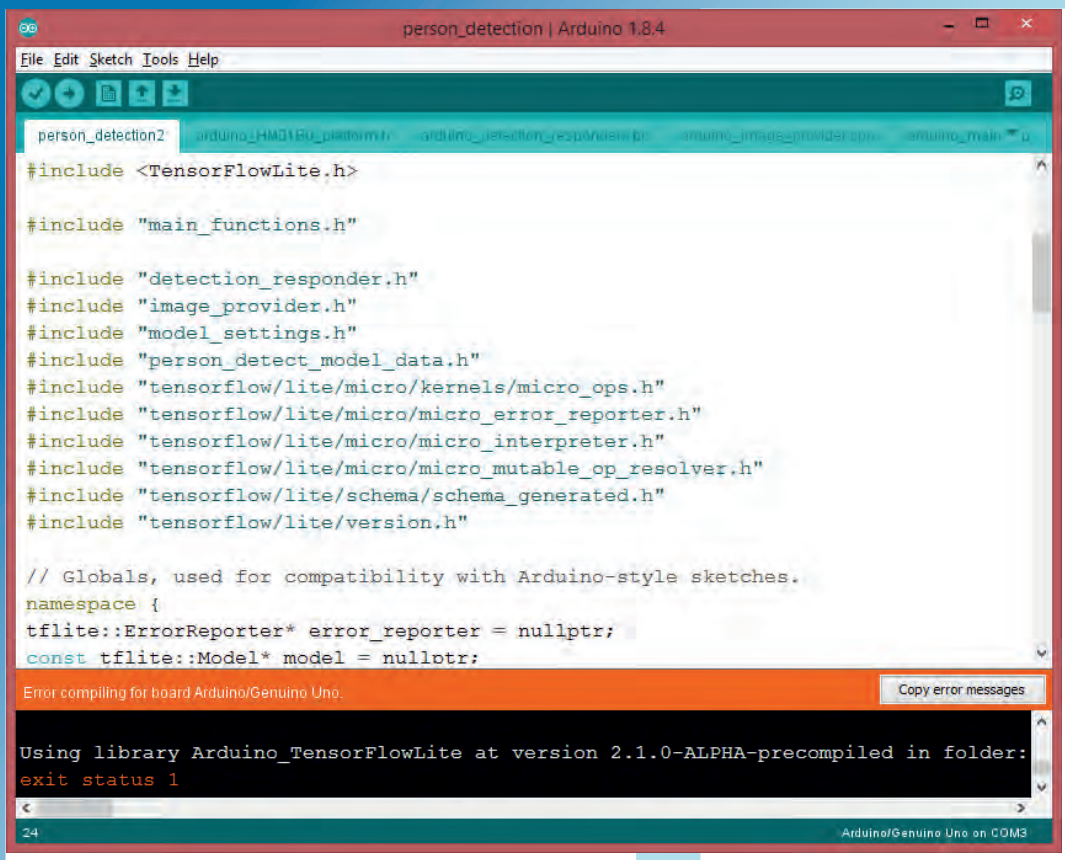


Figure 6: PPM on an Arduino

Conclusion:

The portable pixmap format(PPM), the portable graymap format(PGM) and portable bitmap format(PBM) are image file formats designed to be easily exchanged between platforms. They are also sometimes referred collectively as the portable anymap format(PNM). These formats are a convenient (simple) method of saving image data.
And the format is not even limited to graphics, its definition allowing it to be used for arbitrary three-dimensional matrices or cubes of unsigned integers.

The script and data can be found:

http://www.softwareschule.ch/examples/sphere2.txt
http://www.softwareschule.ch/examples/sphere2.htm
http://www.softwareschule.ch/examples/detector2.htm

Author: Max Kleiner
Ref:
https://people.cs.clemson.edu/~dhouse/courses/405/notes/ppm-files.pdf
http://paulbourke.net/dataformats/ppm/

By Detlef Overbeek

starter expert

Introduction

This article is about the new version of FastReport for VCL (Delphi) and LCL for Lazarus. Each section has a different colour! I have splitted the items apart because they need very much different handling.
The program is not a direct installer for your FastReport components but actually a selection app to pick the correct version of Delphi that you want to use. They call it **"FR6 recompile wizard 1.22"**

Fast Reports
Reporting must be Fast!

INSTALLING FOR DELPHI.
Because Delphi needs to be installed under Windows one needs to notice that Installing anything under hard disk C:\ (or where your programs are installed) has a small hurdle: You can only use the instalment fully if you start Delphi with Admin rights. Otherwise all the demos and extra information **FastReport** needs will bring you into trouble, because you can not save your project: actually it is opened, but refuses to save the necessary files, even starting with an Error message. (*see page 18 Figure 8*).
I wasn't aware of this and FastReport promised to make changes to this in future updates. The problem is solved instantly if you start Delphi with Administrator-rights: (see figure 1). Of course that will be updated.
Go to the taskbar
➔ Choose RAD Studio 10.4
➔ Right click with your mouse,
➔ Run As administrator.

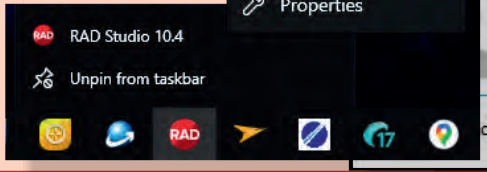


Figure 1:

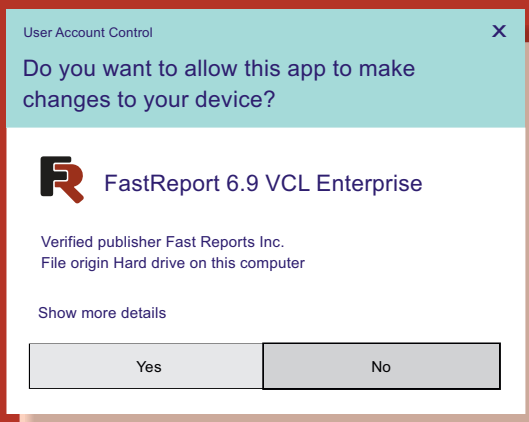


Figure 2:

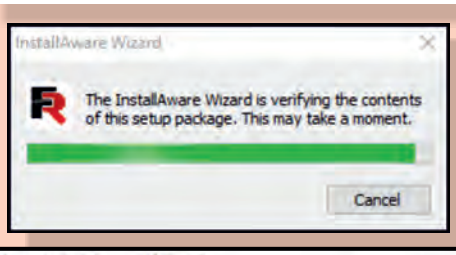


Figure 3:

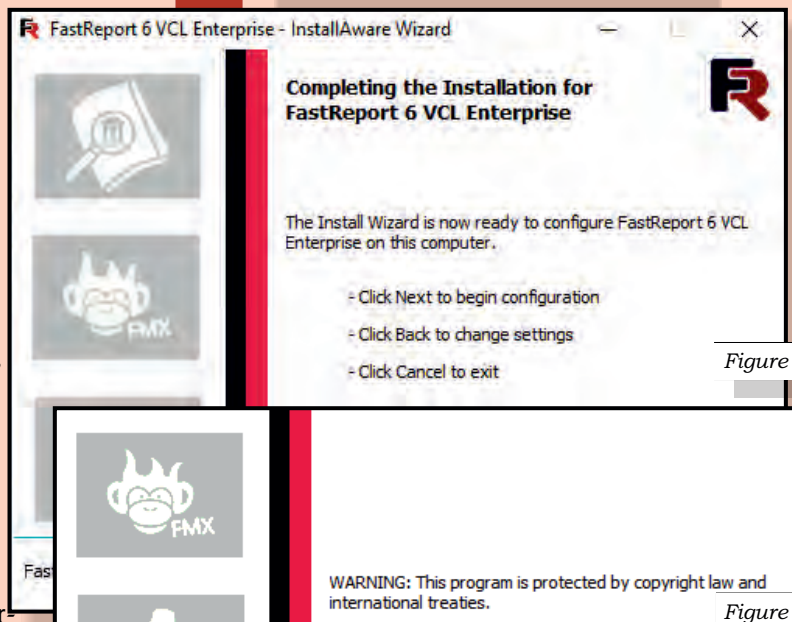


Figure 4

Figure 5:

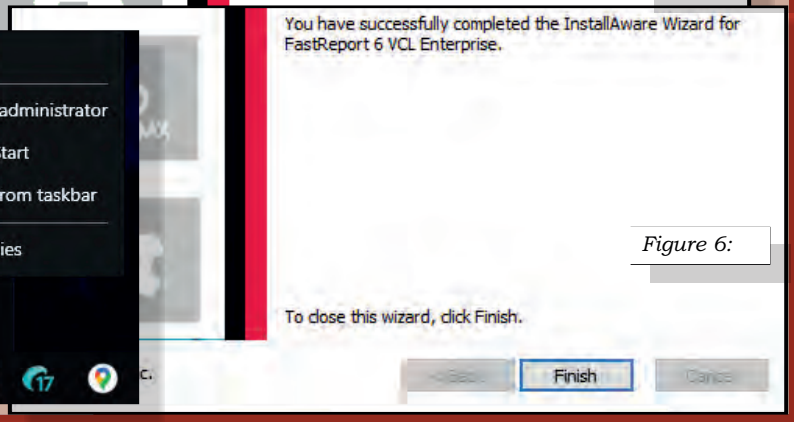


Figure 6:

For **installation** simply click the install file they have made available and do the usual proceedings. The Figures below show what happens. All very easy and quick. After having done this the program **"FR6 recompile wizard 1.22"** can be selected in your windows program list. Remember it is an "FR6 **recompile wizard** 1.22". In itself it is very simple, but I looked for Delphi 10.4 Sydney - it wasn't there - but choose then 10.3 Rio and compiled, and it all worked. Even for Delphi Sydney.

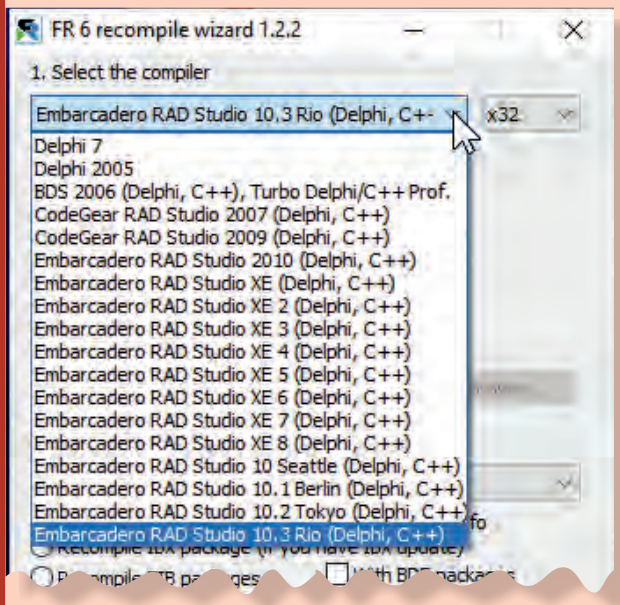


Figure 7: The list of candidates

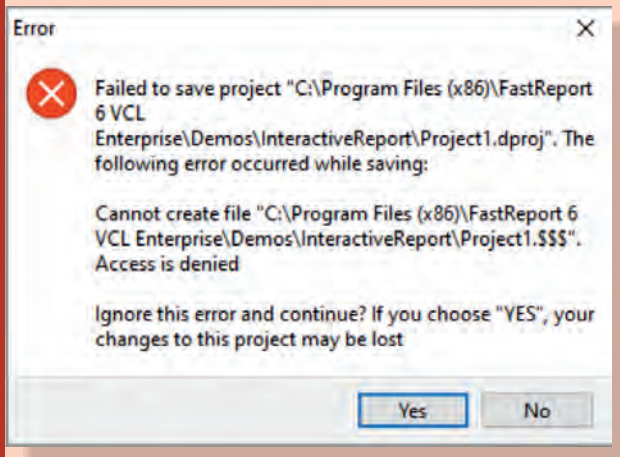


Figure 8: Error message on saving

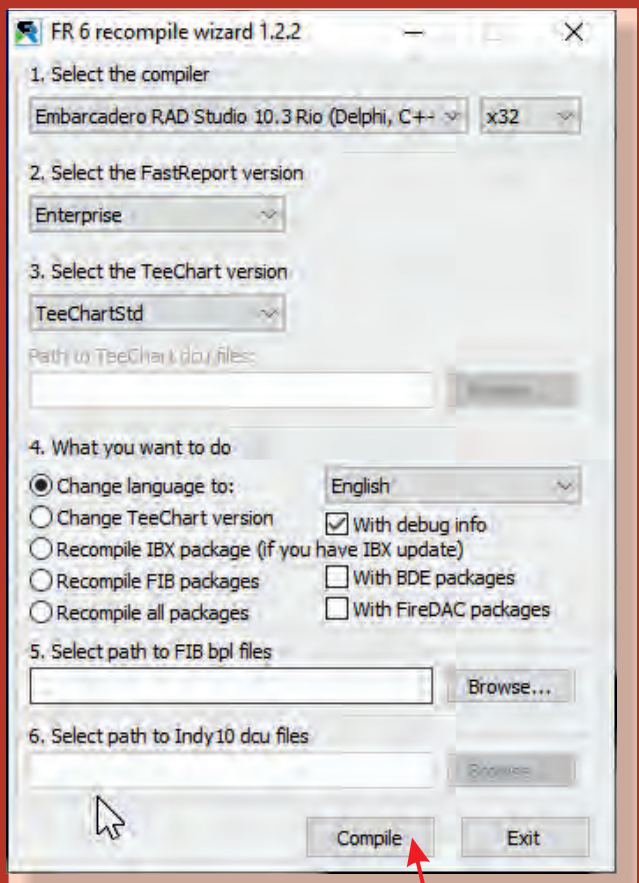


Figure 9: Make your choices.

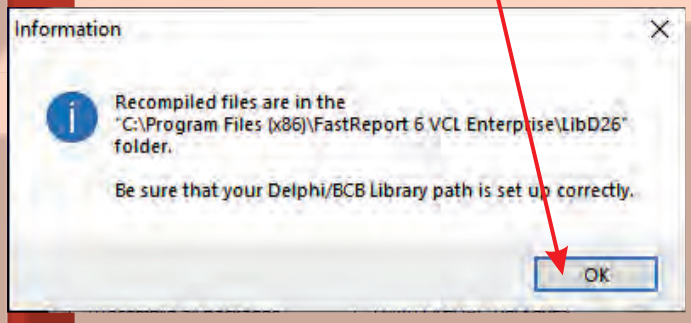


Figure 10: After compiling it show the path of the library

So to solve that simply run Delphi in administrator mode. You must of course reinstall everything, if you save the projects which is inevitable if you want to run the demos at least.

This new version of FastReport VCL 6.9 has of course some nice new improvements.
In the new version **interactive forms** support in **PDF export** for objects is added:
Text, CheckBox and **Picture.**
With the **ability to include only the required font glyphs into the interactive** form via the `InteractiveFormsFontSubset` property.
Learn more in our blog:
`https://www.fast-report.com/en/blog/.`

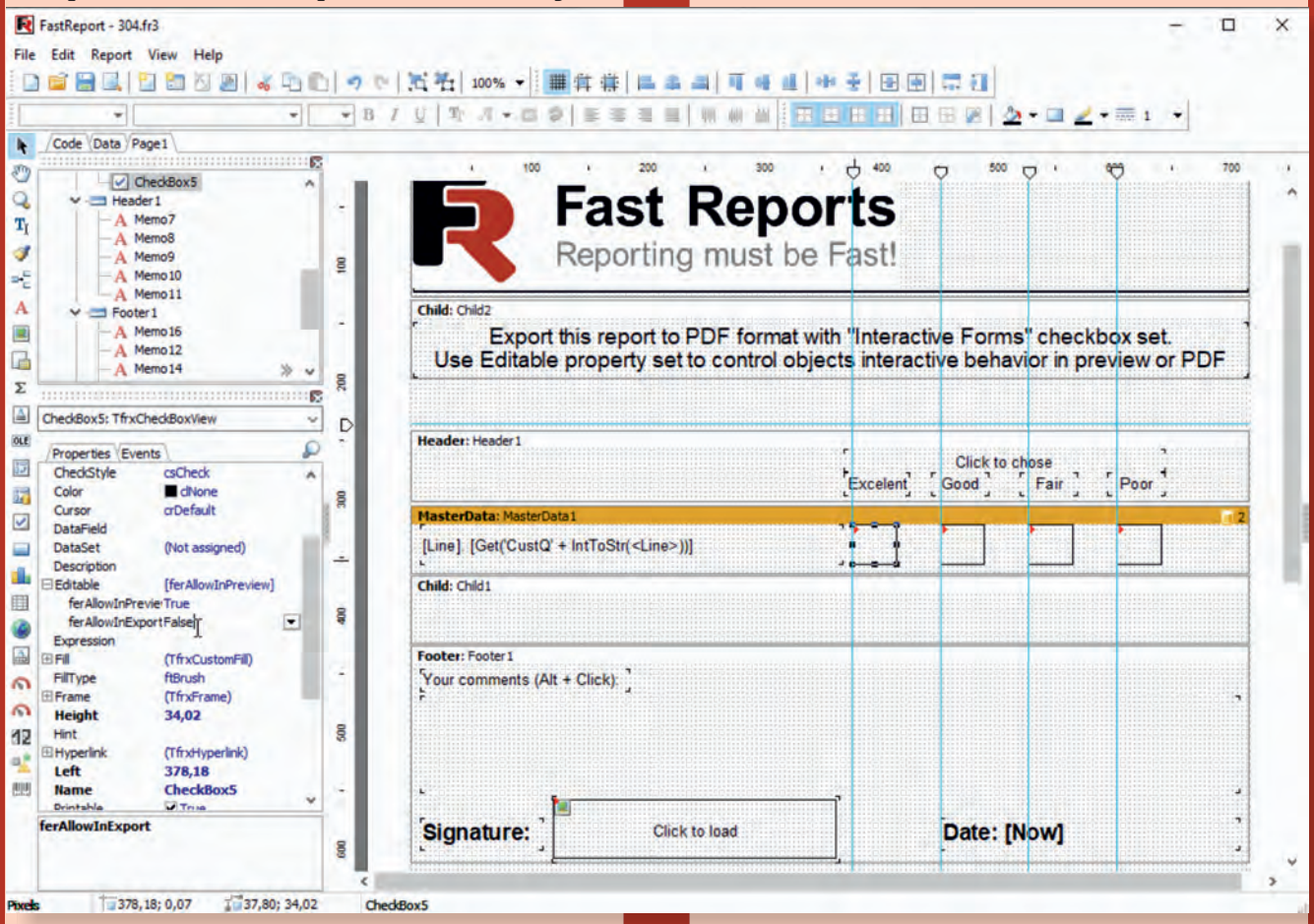


Figure 11: HTML exports now can export Outline tree.

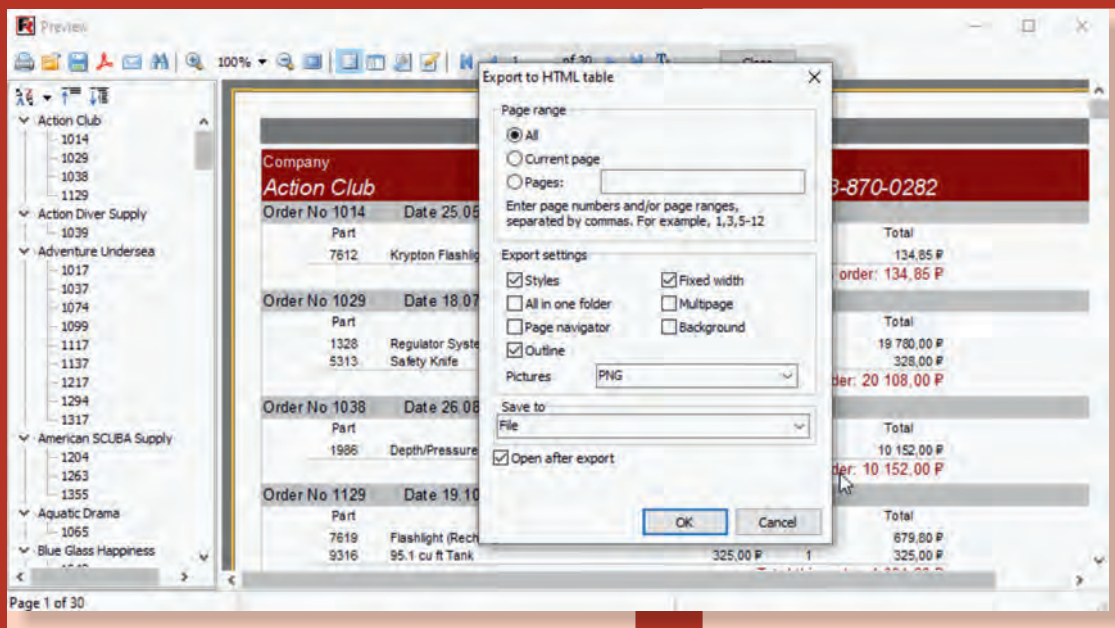


Figure 12: Editing of the dialog pages in the report becomes easier with guide lines support.

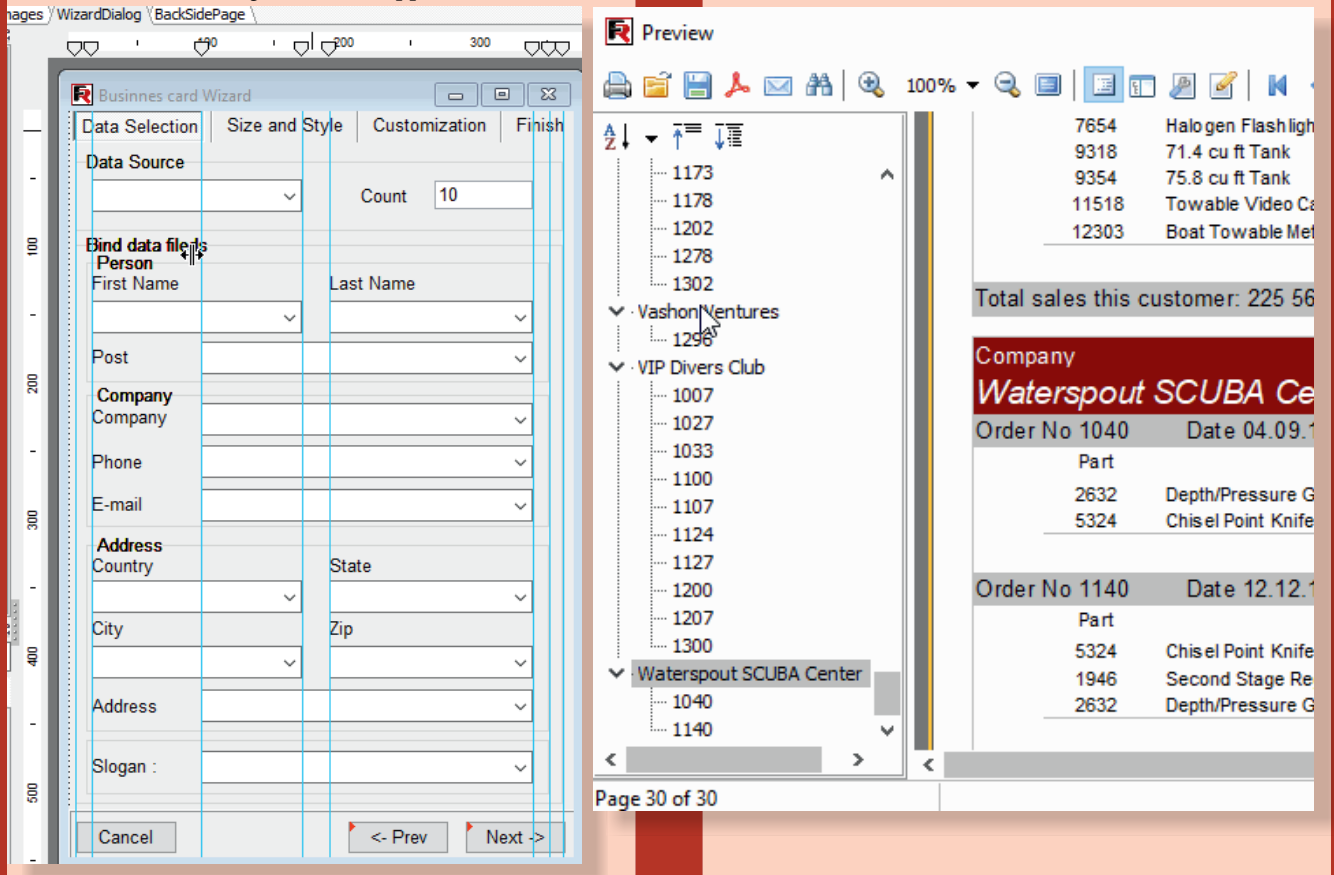


Figure 13: Added filtration support in preview window outline tree for fast search and navigation.

Added two new linear barcodes **Deutsche Post Identcode** & **Deutsche Post Leitcode.**

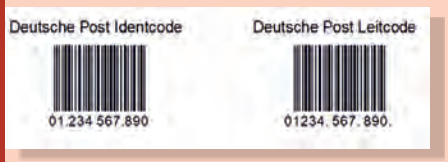


Figure 14:

Also, they added a Beta version of Client-Server components for **Lazarus** (In the FastReport Enterprise version). *(See page magazine page 30 or Page 15/ 18 of the article).*

❶ Added support of **interactive forms in PDF export for Text, CheckBox and Picture objects** (Use `Editable` property with `[ferAllowInExport]` flag)

❷ Added `InteractiveFormsFontSubset` property in PDF export which allows to set used glyph for interactive forms with embedded fonts via an expression like : `A-Z,a-z,0-9,#43-#47,!@#$`

❸ Added **outline support to HTML export filters** (`frxExportHTML` and `frxExportHTML`)

❹ Added **support of Client-Server components for Lazarus in Beta stage**

❺ Added support of **Guide lines** for dialog pages in the report designer workspace

❻ Added **filtration support in preview outline tree** for fast search of nodes

❼ Added support of **save and load for SQL editor presets settings**

❽ Add **ZUGFeRD** minimum level to PDF export filter

❾ Added **CC** and **BCC** fields support for E-mail export **Indy** and **Outlook**

❿ Add **preset helper class for SBER QR** code

⓫ Added **RTL support to DOCX Export** filter

⓬ Added **new linear Barcodes:** Deutsche Post Identcode and Deutsche Post Leitcode

Changed behavior of interactive objects.
Restrictions → `[DontEditInPreview]` is deprecated.
Now all Text objects is disabled for editing in preview by default.
To allow editing use Editable - `[ferAllowInPreview, ferAllowInExport]` for text objects (it impacts only text objects)

FIXING LIST:

- *Copy-paste bugs* fixed when operation doesn't copy child objects of container (Dialog page containers)
- *Fixed bug when copy-paste operation* assigns child objects to container before it's getting pasted
- *Fixed processing of FIB UTF8* blob field
- *Fixed bug with merge of duplicates* in multi-column report
- *Fixed bug in 2D barcodes* with long data
- *Fixed object inspector* for multy monitor configurations
- *Fixed Integer Overflow error* during draw operation for some objects
- *Fixed bug with OnLoadTemplate* event in inherited reports
- *Fixed bug when calls NewPage inside* `OnBeforePrint` event changes current band which affects aggregates calculation
- *Fixed Dropbox transport* processing of root folder.
- *Fixed bug in RTFexport* with color table
- *Fixed bug when report Engine forces Keeping mode* for child bands sequence linked with a report title
- *Fixed memory leaks* with an empty detail reports
- *Fixed Barcode draw* on screen for 100% scale (incorrect stretch 1:1)
- *Fixed removing of temporary folder* after mail export
- *Fixed bug with split big pages printing mode* when additional page was added
- *Multi byte input support for Syntax memo* moved from define to the report designer settings
- *Fixed Ctrl + backspace hotkey* in Syntax memo

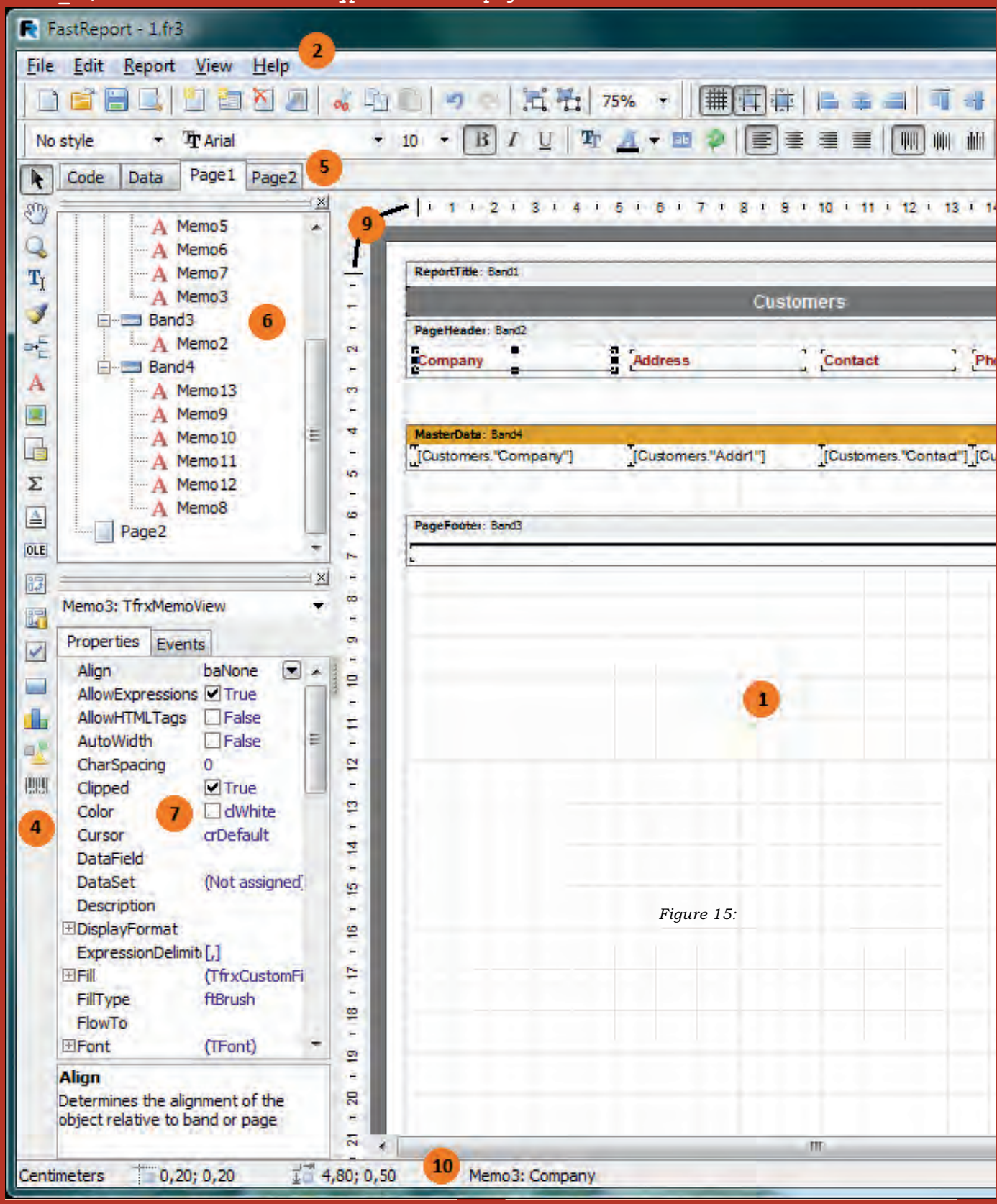
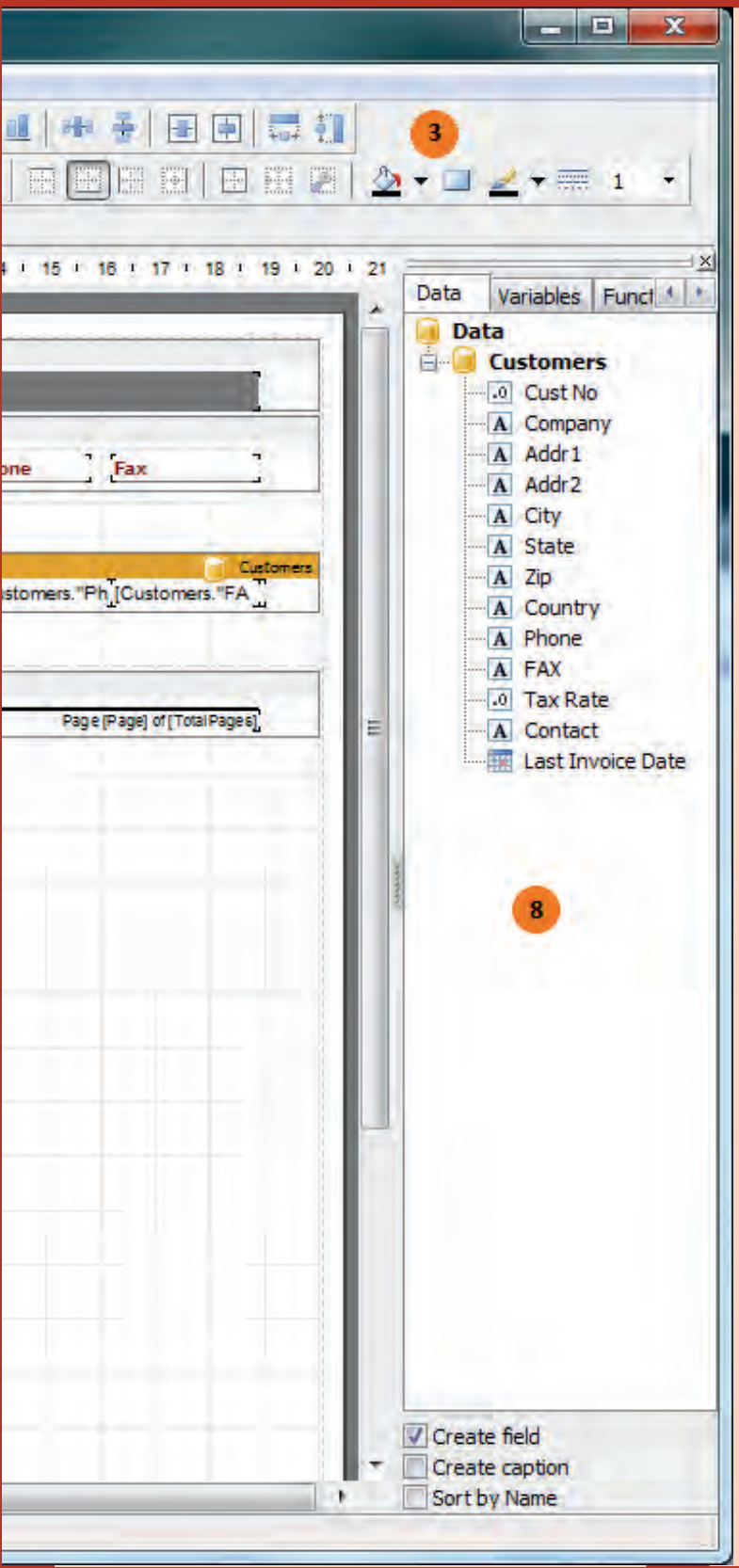

FastReport - 1.fr3
File Edit Report View Help
2
No style
Arial
10
75%
Code
Data
Page1
Page2
5
Memo5
Memo6
Memo7
Memo3
Band3
6
Memo2
Band4
Memo13
Memo9
Memo10
Memo11
Memo12
Memo8
Page2
9
Memo3: TfrxMemoView
Properties
Events
Align baNone
AllowExpressions True
AllowHTMLTags False
AutoWidth False
CharSpacing 0
Clipped True
Color clWhite
7
Cursor crDefault
DataField
DataSet (Not assigned)
Description
DisplayFormat
ExpressionDelimit [,]
Fill (TfrxCustomFi
FillType ftBrush
FlowTo
Font (TFont)
4
Align
Determines the alignment of the object relative to band or page
ReportTitle: Band1
Customers
PageHeader: Band2
Company
Address
Contact
MasterData: Band4
[Customers."Company"]
[Customers."Addr1"]
[Customers."Contact"]
PageFooter: Band3
1
Centimeters
0,20; 0,20
4,80; 0,50
10
Memo3: Company

Figure 15:



The report component is supplied with an embedded visual report designer, which can be opened at design-time by double-clicking on the TfrxReport component. The designer provides the user with all the tools necessary for designing and previewing reports. The designer's interface meets current requirements. It contains several toolbars, which can be docked wherever wanted. These toolbar locations are restored every time the designer is opened. Toolbar locations, together with other designer settings, are stored in the registry or, should one be assigned, in an ini-file.

To give the end user of your project the ability to design reports, you should either place a "TfrxDesigner" component from the FastReport component palette onto a Delphi form or add the "frxDesgn" unit to the unit's Uses list. Using the designer at run-time allows the user to change the report's appearance, as well as to edit the finished report.

Note: you should also place any other Tfrx components that will be used on the Delphi form.

KEY TO REPORT DESIGNER FEATURES:

❶ report design workspace
❷ menu bar
❸ toolbars
❹ object toolbar
❺ report page tabs
❻ "Report tree" pane
❼ "Object inspector" pane
❽ "Data tree" pane :
elements can be dragged from this pane onto a report page
❾ rulers :
a ruler can be dragged onto a report page to create a blue guideline on the page (objects snap to nearby guidelines)

❿ status bar

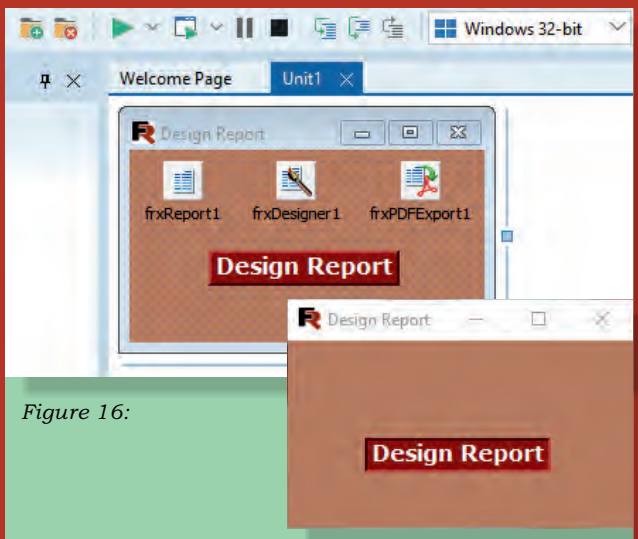


Figure 16:

In this very simple project. I added an old acquaintance, the "Color Button" to make the form more attractive.
I have explained the working in older articles: but here is the install explanation again.

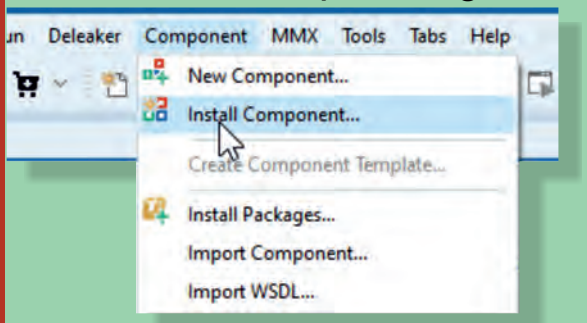


Figure 17:

Nothing else is changed in the demo. The code is of course down loadable from your subscription page. The project is implemented into an new project which is called the Invoice Demo. It will show the design as well the creating of an invoice and export it as a PDF.

THE COLORBUTTON INSTALL

Installing the button component is simple:
It has a few steps: I created a zip file which contains all the necessary items:

❶ `Color_Button.Zip`
unpack this to the directory of your choice: probably `C:\Components\ColorButton`
❷ Start an empty Delphi
(Do not load a project or close it)
❸ Choose at the top: (Figure 17). Component → Install Component. A new window pop up: → select the `Unit filename:` → Select "Install into a new package".
❹ The Unit file name will be now `ColorButton.pas` (*as you can see in figure 18 and 19*).
❺ The Package name: `ColorBtn`
❻ Package description: `Color Button` → is the name on the Tab.
❼ After clicking Finish will yet create an other Window which you need to simply click OK. After that the component will compile and be installed. *Figure 23*
❽ Go to Tools → Options → Language → Delphi → Library → Browsing path and add the directory `C:\Components`.

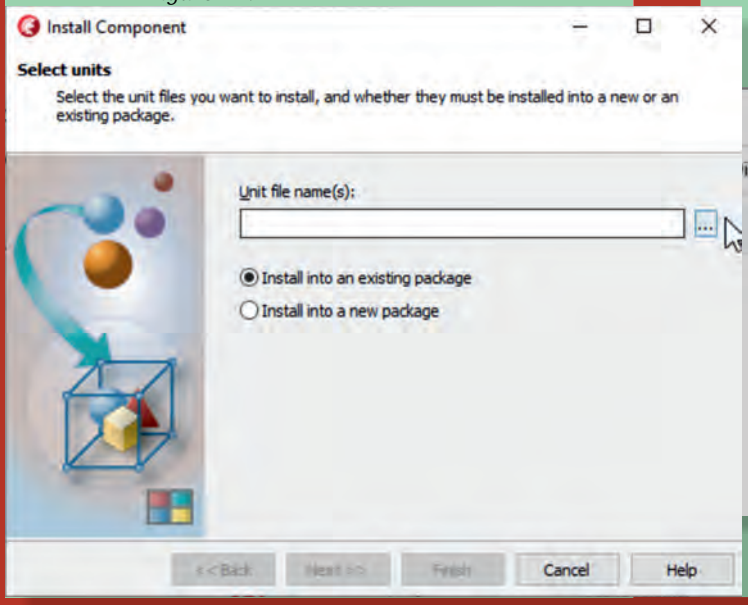


Figure 18:

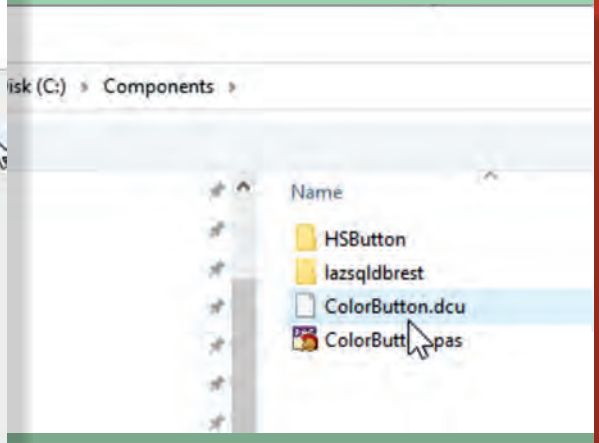


Figure 19:

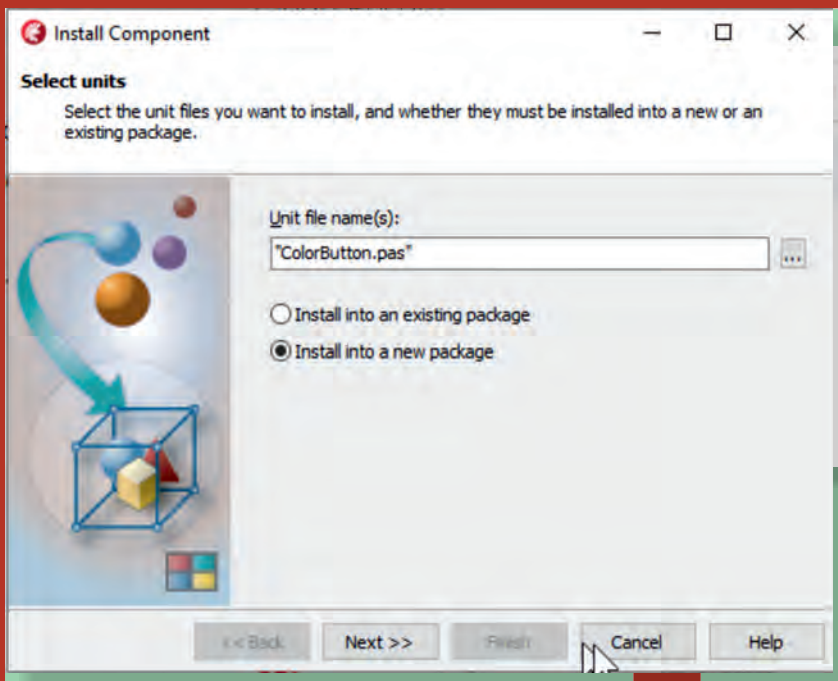


Figure 20: You can chose the file itself (see figure 21)

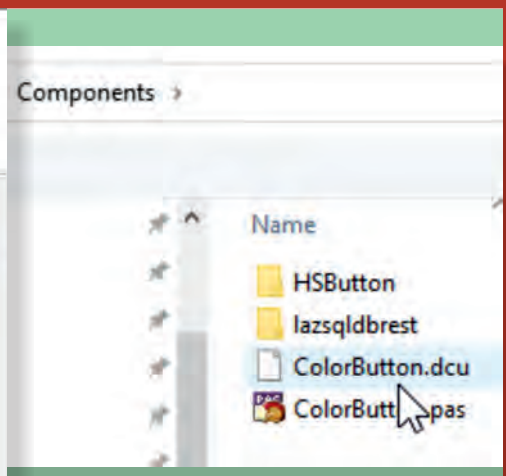


Figure 21: Simply select the .pas file

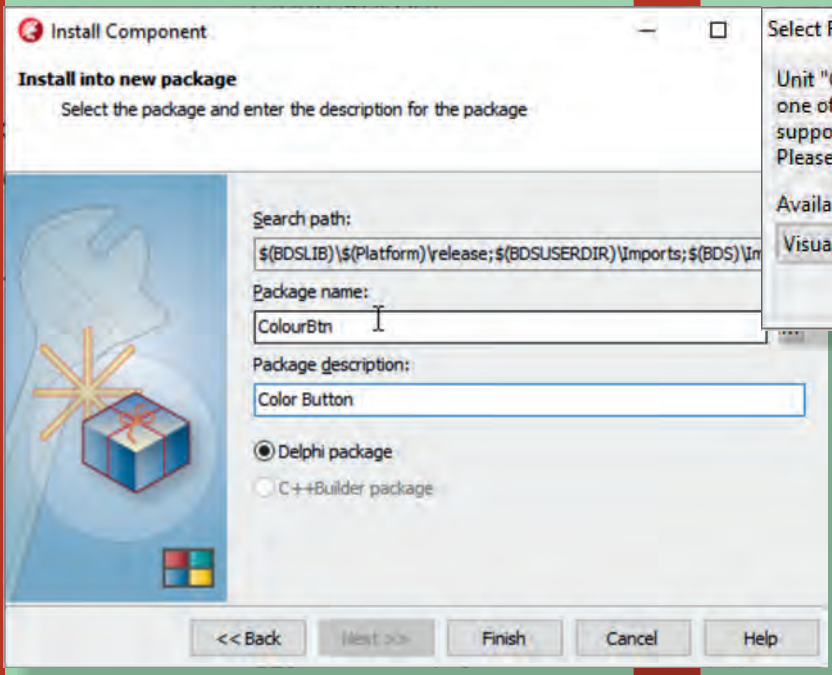


Figure 22: If possible you should set the search path even before you install the component. In the list of page 23 of the magazine - page 9 of the article

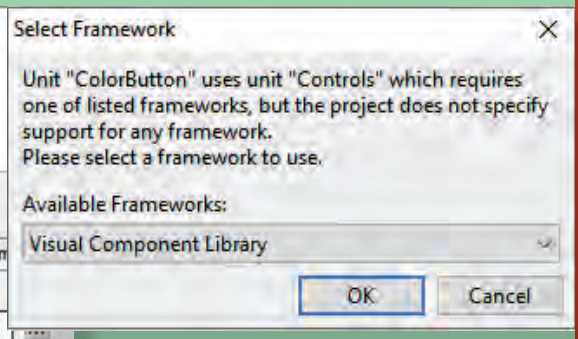


Figure 23: You will have to embed the component into the VCL -framework

If you have problems with installing the color button, please send an email to `office@blaisepascal.eu`

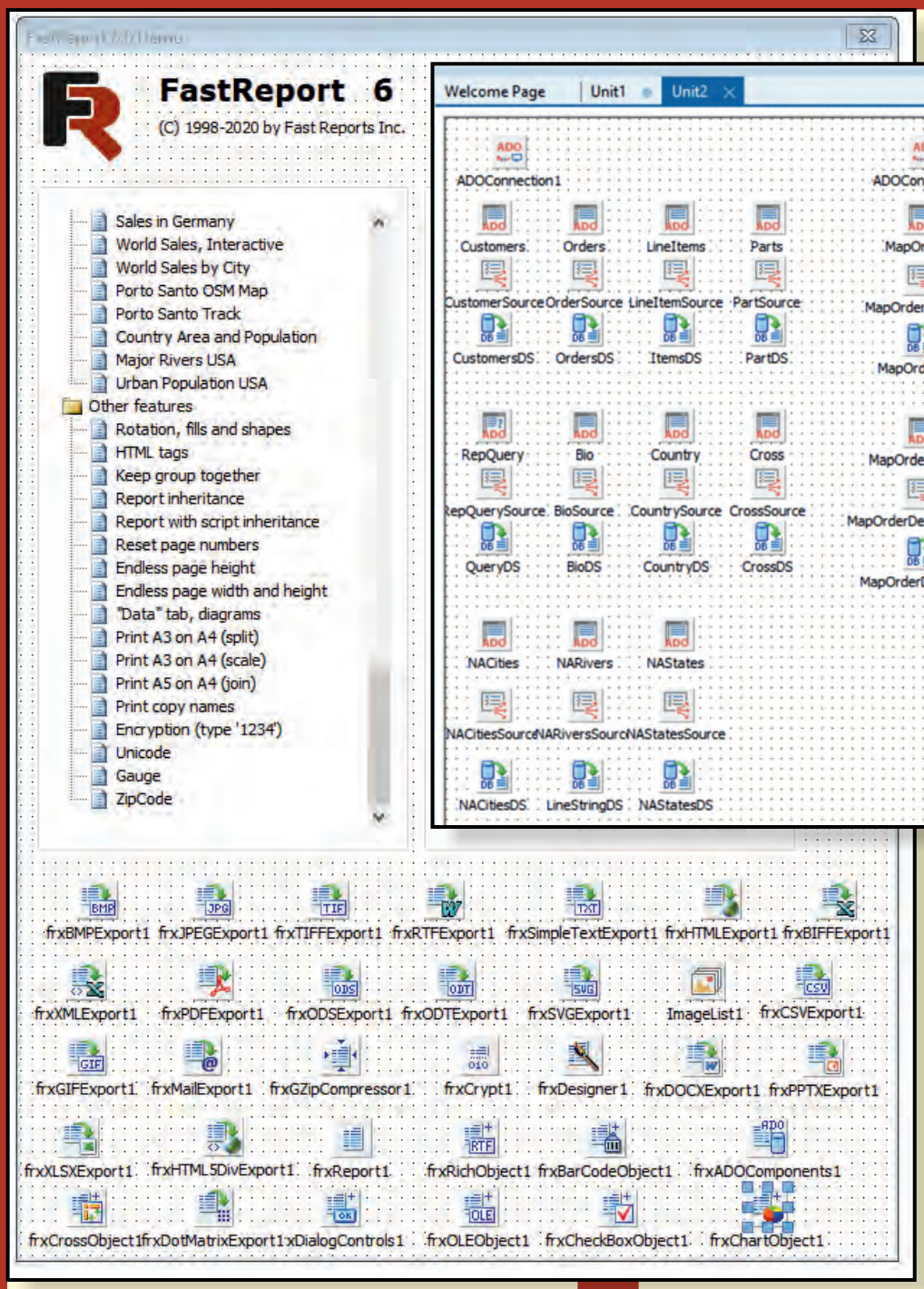


Figure 25: The Unit2

Figure 24: Unit 1

FastReport has designed a very nice project of an enormous number of items you might use as example for your own project. It is available under c:\Program Files (x86)\FastReport 6 VCL Enterprise\Demos\Main\FRDemo.dproj or .dpr. You best make a copy first.

For this example I have enlarged the original form and spread the components so you can have a good overview of what all is there to be used.

Figure 26: the running demo application

The very special thing is: if you compile it you can chose an item and then either see the project as is or have a look at the coding.
On page 25 of the Magazine /Article page 11 there is an overview in the working app a list of items in the left column and here you can choose the item.
In the right pane a corresponding explanation is shown. On the bottom there is a button that will show the report details (design) and a possible preview button. A fantastic way of showing and learning.

I picked under What's New - Text object processing as illustrating items. Don't worry about all the components - the text is very good and explaining. You even have the full sources, though the structure can be confusing but it is very nicely ordered:
Unit 1 (Magazine page 25 /article page 10) has the User Interface and Unit 2 has all the connections.

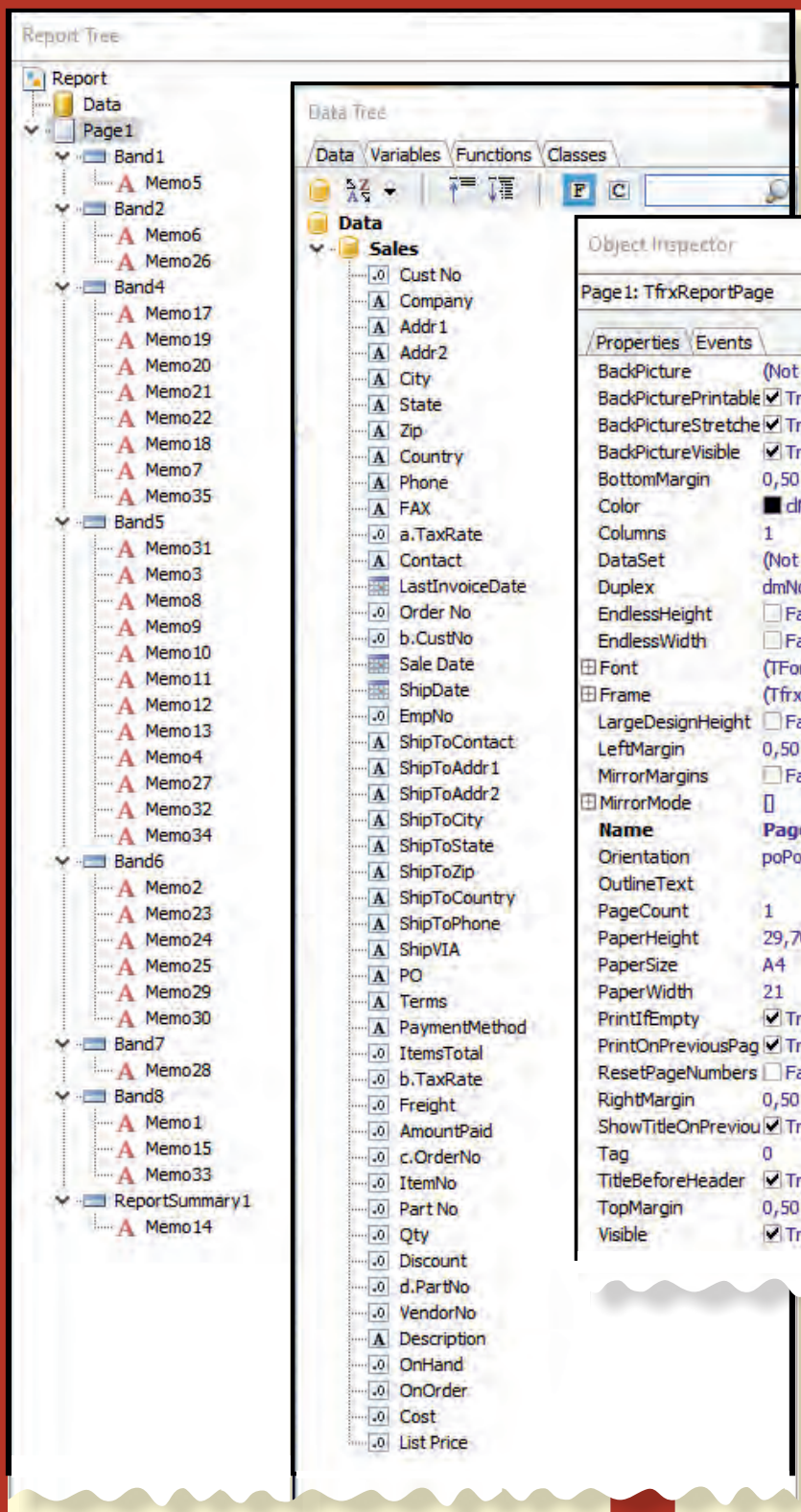


Figure 27: On page 1 of the report you see all the text objects, double click them and you can change the text.

Figure 28: Data Details

Figure 29: The object inspector of FastReport shows the settings

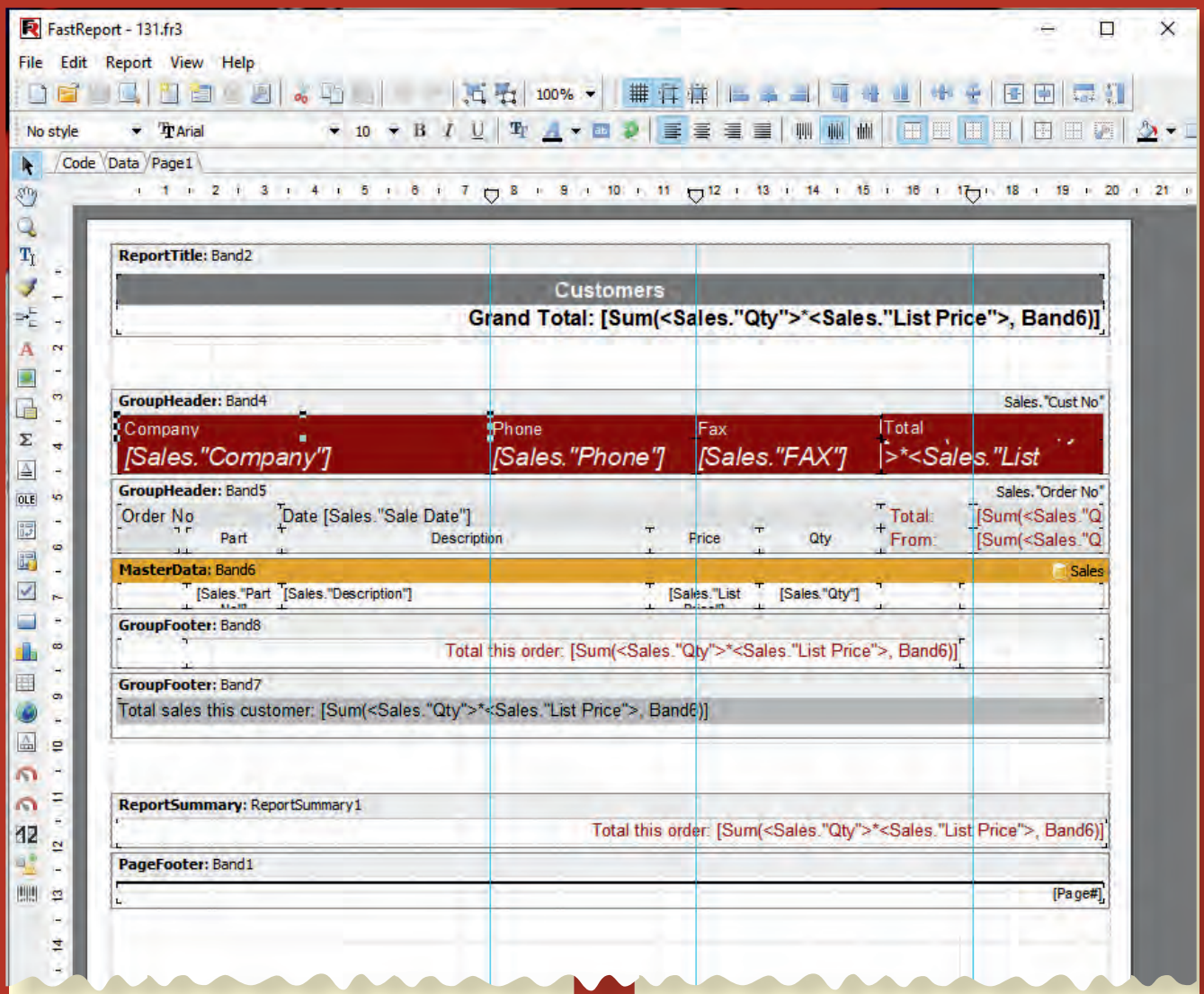


Figure 30: Overview of the complete Form.
Note: *before you play with this make a copy!*

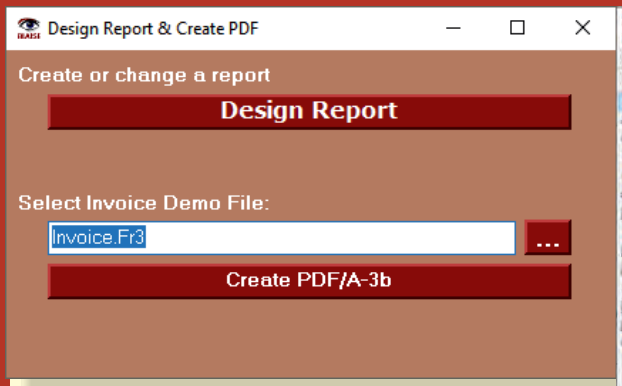


Figure 31: The running app

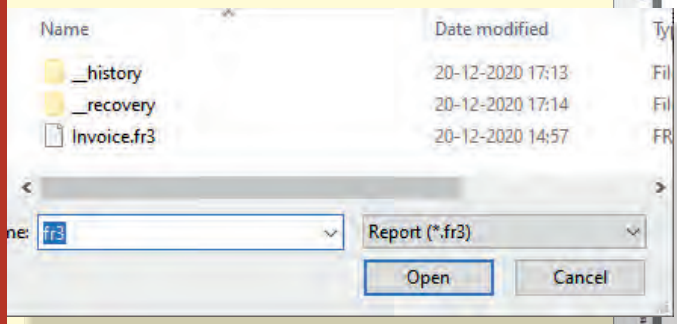


Figure 32: The report you choose will become available

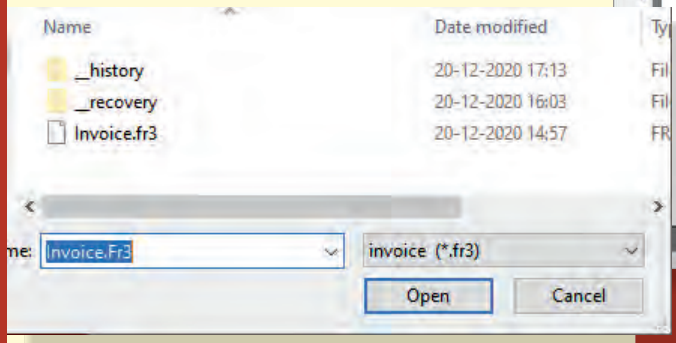


Figure 33: that specific report will now be exported as a PDF File

Because I wanted to make an easy example for you here is the **project Invoice Demo.**

It is easy to understand and does mainly two things: It creates a form or lets you collect an existing one and eventually save it.
After all this you can decide to create an invoice from that. I did that for you as an example. You can export the report as PDF. The code will of course be down loadable from your subscription page.

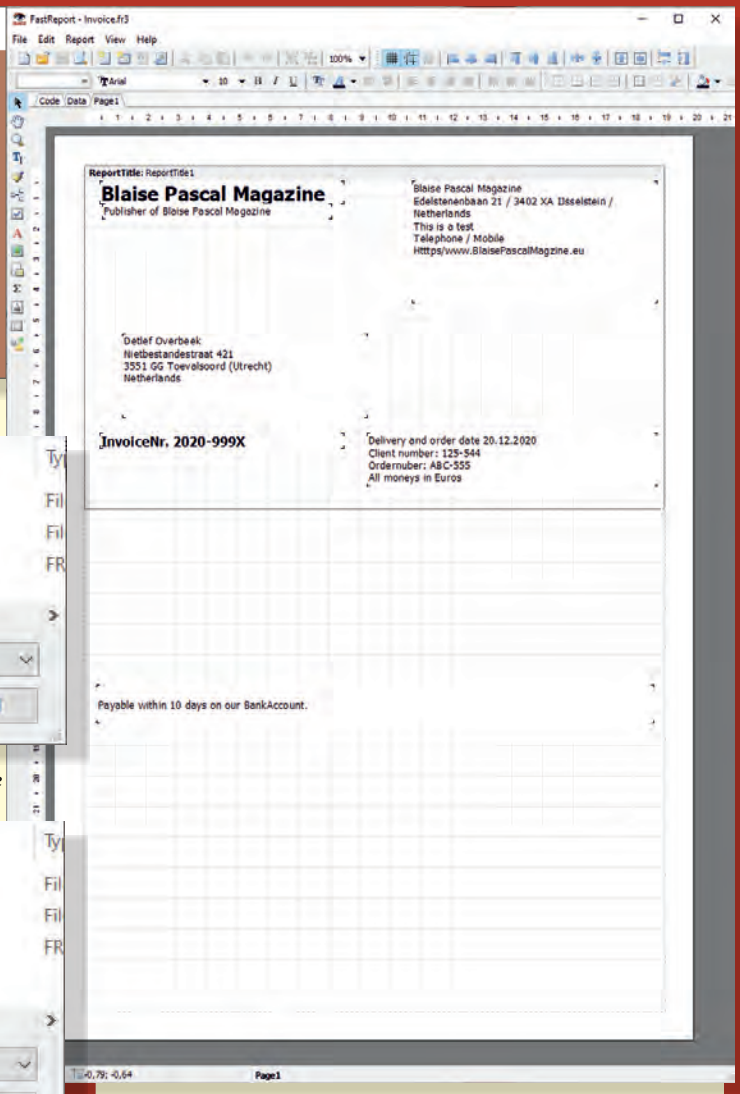


Figure 34: The opened report

Web reports in Lazarus on Linux and or Windows

Until recently, the task of generating reports through the Web from an application written in Lazarus was, let's say, very creative.
Today we are pleased to announce the release of **client-server components for Lazarus** in beta testing.
They can be used to create and customise a server for remote report generation and exporting, while your program and Web browser can act as a client.
The client-server components in **VCL FastReport** were implemented long ago, but before that they had not been adapted for Lazarus and operated only in Rad Studio Delphi and C++Builder.

This article will be useful not only for those who have never used them before, but also for those who want to migrate their FR server to Linux, as it honestly describes all the advantages and disadvantages.

We also discovered a problem in Lazarus itself and found the way to solve it.
And now let's move on to the technical side of the issue.

CLIENT-SERVER COMPONENTS FOR REPORTING IN LAZARUS

The client-server components in Lazarus are available under two operating systems: Windows and Linux (GTK2 only).
They are implemented in the "fr6CS_lazarus.lpk" package, the dependencies of which specify the export package, and **therefore the export package located in the "Source/ExportPack" folder must be installed before installation.**

After installing the client/server components package in Lazarus, you will have a new tab called "FastReport 6.0 Client/Server", where you will find 4 components:

❶ TfrxReportServer is a server component, a report server and a two-in-one HTTP server. Its most important field is `ConfigFileName`, where the `path to the xml` file with settings must be specified.
You can specify many settings in this file, these are just the most important of them:

- Server settings itself, such as port and waiting time;
- Settings for the FR engine, such as disabling the execution of scripts;
- Huge export settings;
- Cache settings, such as cache file lifetime, location of cache storage or you can disable it altogether;
- Logging system settings;
- Database connection settings.

You can read more about all the settings here.

❷ `TfrxServerConnection` is a client component that contains information for remote connection to `TfrxReportServer`, such as: host, port, number of error attempts, timeout and so on.
❸ `TfrxReportClient`, a client component, analogous to `TfrxReport`, performs a report request on the server and displays it on the client. It uses this `TfrxServerConnection` component to obtain information about the server. It can receive a built report from the server and display or export it in-house (on the client's machine).
❹ TfrxHTTPClient is a client component designed to receive arbitrary files via the HTTP protocol. For example, it can request reports already exported from the server.

We also provide
3 demonstration applications:
1 **server** immediately together with reports and databases;
2 **clients** "simple" and "advanced".

As an "Advanced" client, you can perform load-bearing multithreaded testing by pressing the Start button:

We would also like to remind you that the client can be a normal browser and the generation of the website for the web client can also be configured to his or her taste, while the demo server is already configured for the web client.

"Bottle neck" GTK2

Lazarus is a cross-platform open source IDE the Linux server.
GTK2 has one property: single threading.
This means that even if two programmes will draw figures on their BitMap in an endless cycle, GTK2 will narrow everything down to one stream. Therefore, in total, these two programs will draw fewer figures than one program would have drawn, because part of the time will be spent on synchronisation.
But in this case synchronisation is organised without our involvement.
Now take, for example, the situation when the figures are drawn by one programme in several streams (each stream has its own local BitMap). In this case you will have to create a locking system in your code, because we remember about the single threads of this library.

But these should not be ordinary critical sections, but global critical sections at GTK2 level. And when one of the threads enters this critical function, even the basic form hangs until it comes out of it.
Multithreading with pictures can only be organised by finding a similar drawing component that is independent of GTK2 (for example, for this example with pictures), in our case another component is not suitable. Therefore, some server operations are narrowed to one stream.

And because of this feature, in particular, it is better not to use dialogues, because as long as one customer answers a dialogue, the building of reports for other customers will be on pause.
The question may arise: "Why do you even need a single-threaded server?" and you will be partially right, as it is similar to the "bottle neck" problem on roads where the total road capacity equals the road capacity in the narrowest place.
But you will be right only partly.
Only three large operations are narrowed down to one flow: uploading reports, building and exporting.
Receipt and transfer are carried out in parallel, and thanks to the caching system of built reports in some cases you will not need to build them again.
And this, in combination with a powerful server, will make this problem much less visible. It is also a problem only for Linux (GTK2), because this problem does not exist in Windows, and with the release of GTK3 (*which already supports multithreading*) in Lazarus it will disappear altogether.

FastReport VCL Enterpris
127.0.0.1:8097
FastReport VCL Enterprise
Version: 6.8.0
© Copyright 1998-2020 by Fast Reports Inc.
On-line statistic
Contact e-mail
FastReport home site
List of available reports at the server
1.Basic reports
01.Simple list
Demonstrates how to create simple list report.
02.Simple group
Demonstrates how to create simple report with one group. Group breaked each time when first letter of company name changes.
03.Nested groups
This report shows how to use multiple groups.
04.Multi-column list
Demonstrates how to create simple report with columns. Just set number of columns in page options.
05.Multi-column bands
This example shows how to print multi-column bands. To make data-band multi-column, set its Columns property to 2 or more.
06.Memos and pictures
Demonstrates how to create simple list report with images and memos from DB fields.
07.Split bands
Demonstrates how to break long memos. Set "Stretched" and "AllowSplit" options of the MasterData band and "Stretched" option of the object with long memo.
08.Side-by-Side subreports
Demonstrates how to create side-by-side subreports. Each subreport can have different number of records.
09.Report with title page
This report contains two pages (title and list). You can have several pages in your report. Each page can contains one report and can have own paper settings (size, margins, orientation and etc).
10.URLs, anchors
This report demonstrates how to use URLs and anchors. To assign an URL to an object, just fill in its URL property. If first symbol of this property is '#', this is an anchor (it should be added in the script by Engine.AddAnchor method). First '@' symbol means a page number. Otherwise it is treated as an URL (for example, http://www.someurl.com).
11.Keep group together
This report shows how to keep all group records together on the page.
12.Totals in group header
This report demonstrates how to show group total in the group header. The report is two-pass report. On first pass the group totals are accumulated in the variables, on final pass they are printed in the group header.
2.Cross-tabs
01.One row
Demonstrates how to print cross-tab reports with TfrxCrossView object. To use this object, include TfrxCrossObject component from FR component palette into your project. To configure cross-tab, double-click it. This table has no row items and displays only grand total row.
02.One column
This table has no column items and displays only grand total column.

OTHER POSSIBLE PROBLEMS
A small part of the functionality was cut out during porting, namely the CGI and authorization system. How soon we implement them depends on real requests for them from our customers.

Exporting to HTML for Lazarus is still far from perfect, the web client can and should be used, because it is convenient, but the reports will not display perfectly.
However, you can set up the system so that the web client receives reports exported not in HTML but in PDF, for example.
After all, most modern browsers can display PDF files perfectly. We have implemented this export format very well, which you can see for yourself.
Correction of an error in the Lazarus message queue 2.0.10
For those who want to use our new components in Linux, one important thing is worth mentioning: we encountered an error in blocking the message queue and could not bypass it. We were able to contact the programmers in Lazarus, they have already fixed this problem and in the next version of Lazarus everything will work steadily. For now, in version 2.0.10 and later this bug will have to be fixed locally. It will not take much effort or time, all you have to do is add three lines to one of the Lazarus files and rebuild it:

1. open the folder where Lazarus is located (on **Ubuntu,** for example `/usr/share/lazarus/2.0.10`)
2. open the file `~/lcl/interfaces/gtk2/gtk2msgqueue.pp` in it.
3. find the procedure:
   `procedure TGtkMessageQueueue.Lock;`
4. In this procedure, find the line:
   `g_main_context_acquire(FMainContext);`

   This code disregards the fact that the `g_main_context_acquire` function may fail.

   So replace it with:
   `g_main_context_acquire (FMainContext);`

That is to say, we call `g_main_context_acquire` in a cycle until it returns True (completed successfully).
We invite you to try out our new components for Lazarus, but we would like to remind you that this is a beta version and it will improve according to your needs. In case of any errors, be sure to write to us in support.

The rest of the problems are many times smaller and even tiny compared to this one:

WHAT IS LEFT BEHIND
A small part of the functionality was cut out during porting, namely the CGI and authorization system.
How soon we implement them depends on real requests for them from our customers.

Exporting to HTML for Lazarus is still far from perfect, the web client can and should be used, because it is convenient, but the reports will not display perfectly.

However, you can set up the system so that the web client receives reports exported not in HTML but in PDF, for example. After all, most modern browsers can display PDF files perfectly. We have implemented this export format very well, which you can see for yourself.

starter expert

INTRODUCTION

Cyclic Redundancy Checking (CRC) is a way to insure the integrity of data.
During transmission data may be disturbed by atmospheric interference, bad contacts or other hardware failures.
Also damaged magnetic media cause corrupted data. Data also may be manipulated by interested parties.
The basic idea of CRC checking is to attach a unique number to the data.
This number is generated in the following way:
Regard the data (message M) as one big binary number.

Choose a number k (called the key) and divide `M` by `k`, the remainder is `r`.
The quotient `Q` is discarded. The remainder `r` is called the checksum and this number is attached to the message,
`M = Q.k + r .`
`(Message = quotient times key + remainder)`
Look at the next picture for the case of data transmission:

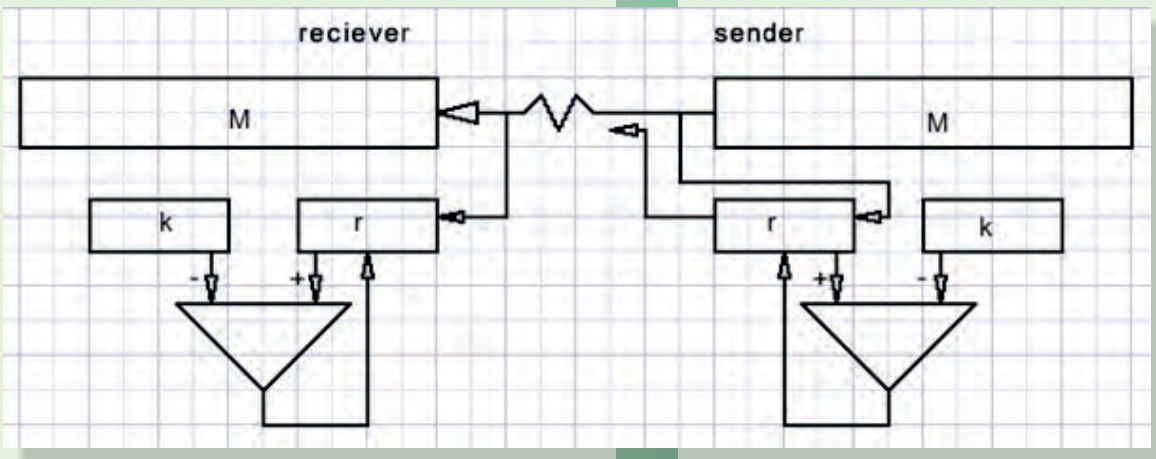


Figure 1: the case of data transmission:

At the sender side, bits of message M send are also shifted into register `r` while `k` is subtracted from `r`. After message `M`, the checksum `r` is transmitted.
At the receiver side also the checksum is generated.
When `M` bits are transmitted error free, the generated checksum at the receiver side equals the checksum send by the sender.

Tested with: Delphi 7

CAPABILITIES

In case of a message length of `32 bits` and a checksum of `16 bits`, there are `232/216 = 65536` messages that share the same checksum.
This looks bad at first glance but consider a randomly damaged message `M`: it only goes undetected if `1` out of `65536` checksums is generated.
With a n bit checksum the chances for a random error staying undetected is `1 / 2n .`

DECIMAL EXPLANATION

To understand CRC checking, take the example of normal decimal arithmetic and a `key (k)` of `10`.
Then message `192743` generates a checksum of `3` regardless of the other digits.
Any error except for the last digit `3` goes undetected.
Reason is that `40, 700, 2000, 90000, 100000` are multiples of the key k.

Now consider a key of
`11. 19274310 = 1218a111 {a = 10}`
This message will generate a checksum of `1`.
Next an **error** is imposed on `M`, which becomes `19574310 = 12407911` and now checksum `9` is generated. The error is detected.
Let our `key k` be `125`.

`8 * 125 = 1000` so, errors in digits `3,4,`...etc will go undetected because they impose errors on `M` that are a multiple of `1000`.
Here we reach an important conclusion:
an error `E` superimposed on message `M` goes undetected if `E` is a multiple of `k`.
Working decimal, `10n` may not be a multiple of `k for n = 1,2,3,….`
This is the case when the number system base and the key have no common factors.

With a key of k bits, producing checksums of `n = k-1` bits long, any single error burst within n bits will be detected.

PUBLIC OR SECRET KEY?
In the case of data transfers a public key, known by everyone, is fine.
Another situation occurs when **CRC checking** is used as a signature to protect data against unauthorized modification.
In this case the checksum is generated with a secret key, only known by the application.

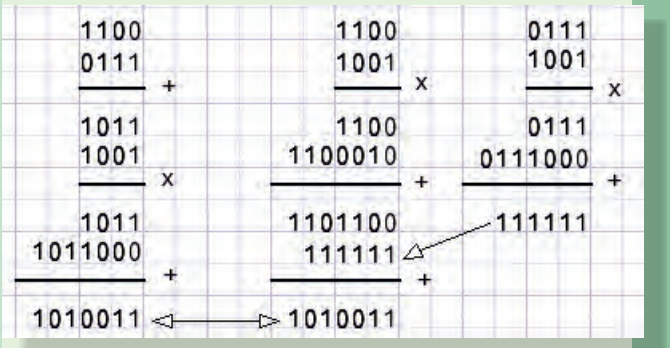


Figure 2: CRC cheking

SIMPLIFIED ARITHMETIC
Before, I mentioned that the checksum is generated by division.
Division implies borrows.
Borrows (and carries) may be avoided by defining addition (subtraction) as exclusive or (xor) operations. This simplifies hardware without reducing effectiveness.

Binary xor operations are
`0+0 = 0; 0+1=1; 1+0=1; 1+1=0;`
No carries, no borrows, they are simply ignored. There is no difference between addition and subtraction. xor is also called: “logical difference”.

In `x or , 0` is the unity element of operation: `1+0 = 1, 0+1 = 1.`
1 is it's own inverse : `1 + 1 = 0.`
The associative law holds:
`(a+b) + c = a + (b+c)`
The commutative law holds : `a+b = b+a`
The distributive law holds: `a(b+c) = ab + ac`
see the next example as proof where `a=1001, b=1100` and `c=0111`

The left column shows `a(b+c)`, the right column `ab + ac`.
So we have a valid arithmetic system.
Here is another reassuring example:
the calculation `a * b / b = a for a=010111` and `b=101101`

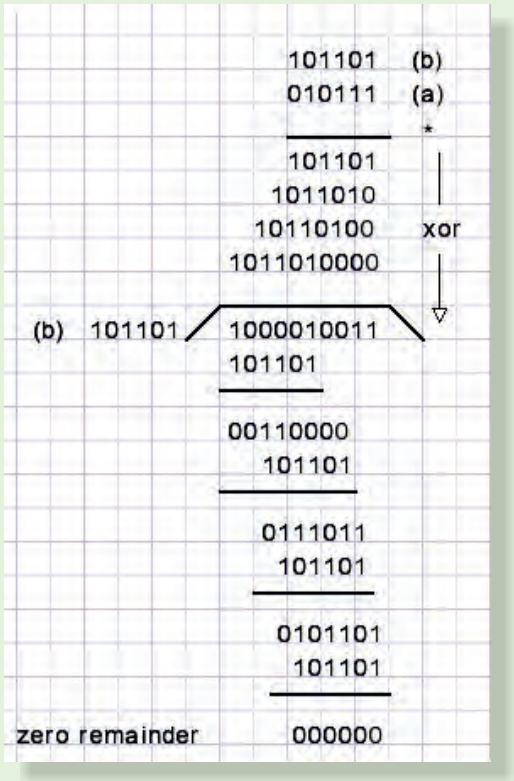


Figure 3: a valid arithmetic system.

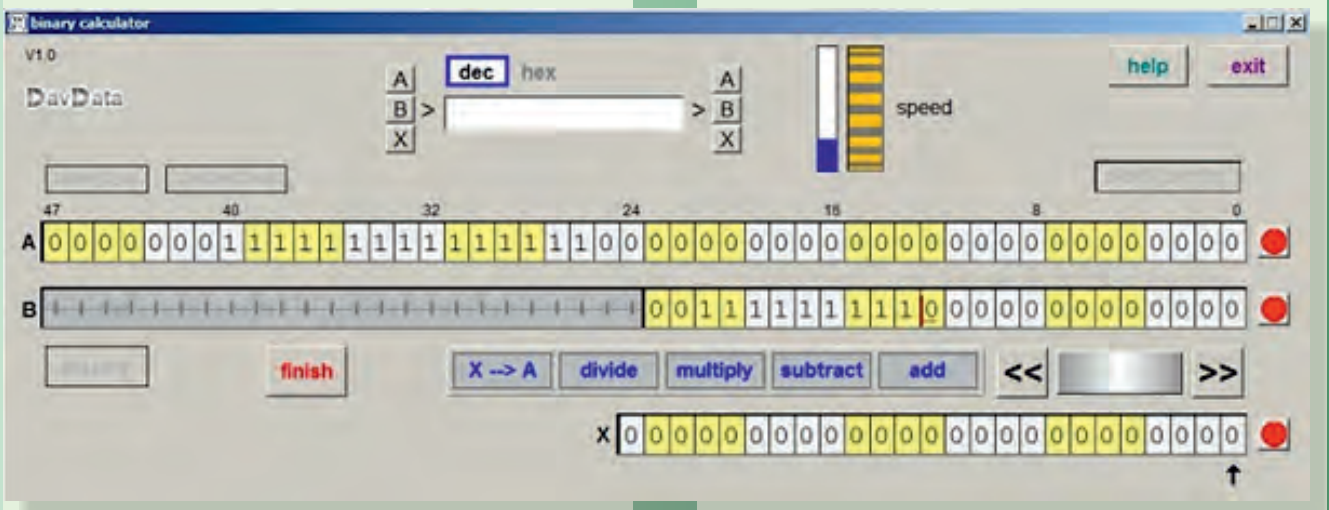


Figure 4: a simple calculator

Note: we are not interested in the quotient.

CHOOSING A KEY

A key of `1 0000 00002` will generate checksums that equal the 8 least significant bits of `M` because any message of the form `xxx 0000 0000` is a multiple of the key.
Any other key is good.
The choice of a key depends on the type of errors expected. Many keys are designed to detect double bit errors that are far apart.
In that case errors like `1000000000000000 1` may not be a multiple of the key.
To choose a key may start with factorizing expected errors to make sure they are not a multiple of the key. This arithmetic, of course, must be performed using the above rules ignoring carries and borrows.

POPULAR KEYS

A popular `16 bit` key is
`1 0001 0000 0010 0001` which is known as the `X25` standard.
Another 16 bit key is
`1 1000 0000 0000 0101` called the CRC-16 protocol, used in **modems.**
The **Ethernet** standard uses the 32 bit key:
`1 0000 0100 1100 0001 0001 1101 1011 0111.`

The keys are designed to detect double bit errors that are many bits apart.
Choosing a certain key is based on the assumption that some errors are more likely to occur than others.

A BIT CALCULATOR

To explore and learn binary arithmetic with suppressed carries and borrows I have programmed a simple calculator. Below is a reduced image.

Calculations may be performed with or without carries.

The speed is adjustable from 1 to 25 operations per second.
It uses 3 registers:
`A : 48 bits.`
`B : 24 bits`, may be shifted left to represent value `B*2n` where n is the shift count.
`X : 25 bits.`

Operations are:
`Add: A = A + B.`
Overflow sets if the sum exceeds `248 -1`
`Subtract: A = A - B`
Underflow is set when A becomes negative.
`Multiply: A = A + B.X`
(clear A before operation)
`Divide: X = A / B`
A holds remainder after division.

Numbers `(0,1)` may be entered directly into the register or entered in decimal or hexadecimal format in an edit box.
Register values may be displayed in decimal or hexadecimal format in this edit box.

Image below lists the controls and registers:

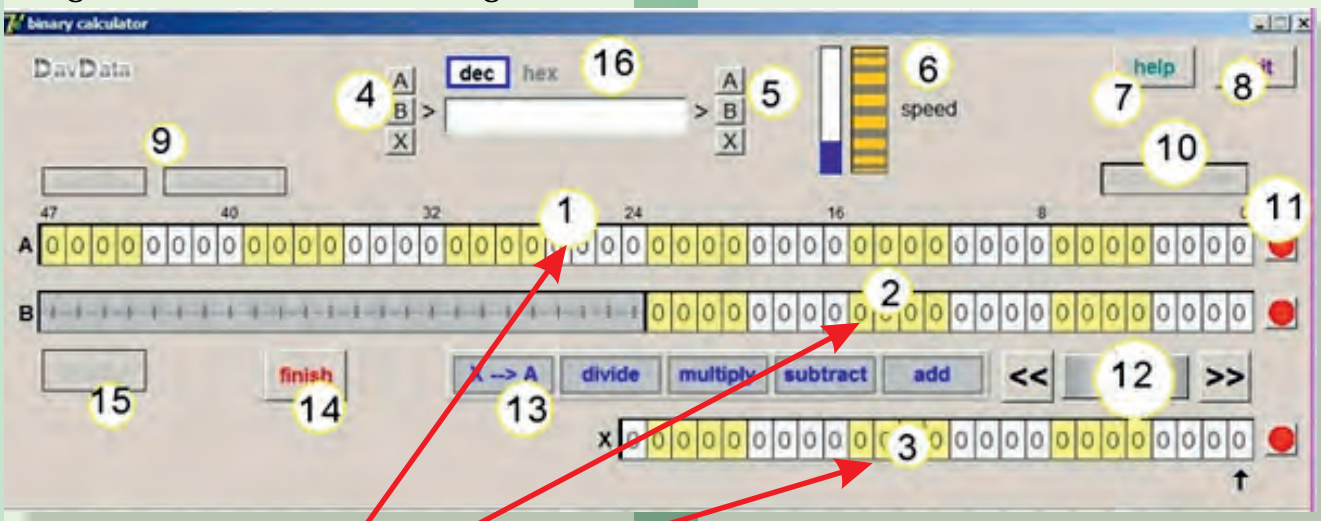


Figure 5: The controls and registers

1. **A register.**
2. **B register.**
3. **C register.**
4. **Buttons to transfer register to edit box.**
5. **Buttons to transfer edit box to registers.**
6. **Speed control**
7. **On-Line help**
8. **Close exerciser and exit.**
9. **Overflow / underflow indicators.**
10. **Carry suppression button.**
11. **Clear buttons for A,B,X registers.**
12. **B register left – right shift handles.**
13. **Operation buttons**
14. **Finish operation in progress, bypassing delays.**
15. **Operation busy indicator.**
16. **Edit component for decimal or hexadecimal display of numbers.**

THE DELPHI (7) PROJECT

`Unit 1` handles button events and indicators.
`Unit 2` consists of a class called `TEdit64` which is a descendant of `TObject`.
`TEdit64` has a box property pointing to a paintbox for display of register values.

The `TEdit64` class does not perform arithmetic operations, but only displays and edits data, shifts the B register and manipulates a cursor.
Data is held in property digits, an `array[…]` of byte.

Byte layout is:
`1xxx xxxx` right pointing arrow displayed at top of bit to indicate a borrow
`x1xx xxxx` left pointing arrow displayed at top of bit to indicate a carry
`xx10 xxxx` bit displayed in red color (not black). Not used.
`xxx1 xxxx` bit displayed bold. (not used)
`xxxx nnnn` number `0..15` .
Only `0,1` used here.
Some properties of TEdit64:

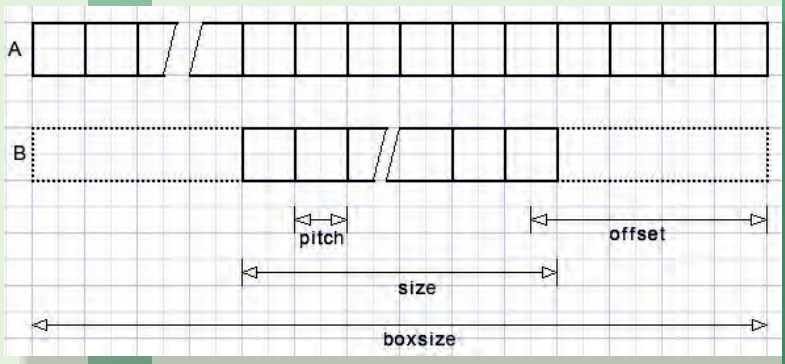


Figure 6: Some properties of TEdit64:

Size, boxsize and offset are counted in binary digits.
Pitch is the pixelcount per digit.
Binvalue: read (write) register value as 64 bit integer.
Box: associated paintbox for display of register.

`Unit 3` handles arithmetic operations.
Operations are performed bit-wise.
A `rotationbutton` component (simulation of a rotary button as found in laboratory equipment) allows for adjustable speed.

The button is set for values `0..20`.
The delay is obtained from a microseconds timer component.
Value `0` causes `1Hz` operation speed, `20` causes `25Hz`.
Each step must cause the same relative speed increment so `25 = step20`.
`Step = e0.05.ln(25) = e0.1609`.
So, `delaytime = 1e6 / exp(n*0.1609)` where `n` is the `rotation-button` position.
This unit has the following procedures:
`procedure ADDstep(n : byte);`
Adds bit `n` of `B` to corresponding bit of `A`.
`procedure carrystep(n : byte);`
is called to propagate a carry through `A`.
`procedure AddBtoA;`
calls `ADDstep` and `carrystep` to make full addition `A = A + B`.

A similar case is subtraction with `procedures subtractstep(n:byte)`, `borrowstep(n:byte)` being called by `procedure subtractBfrom A`.
Multiplication is done by calling `AddBtoA` repeatedly if the corresponding `X` bit is `1`.
`Division` calls `subtractBfrom A` repeatedly, setting an `X` bit if subtraction is done.
For each position `(offset)` of B a check is needed because subtraction may not be possible.
Different checks are used for the case of carries and carry suppression.

`function AminBOK: boolean;` returns true if subtraction with carries is possible.
Note: in computer hardware the subtraction is actually made and negative results are restored. However, non restore divide algorithms also exist.
`function AxorBOK: byte;` checks for carry suppressed division and returns
- 0 : continue shifting B register right
- 1 : xor B to A and shift B right
- 2 : error

See examples below:

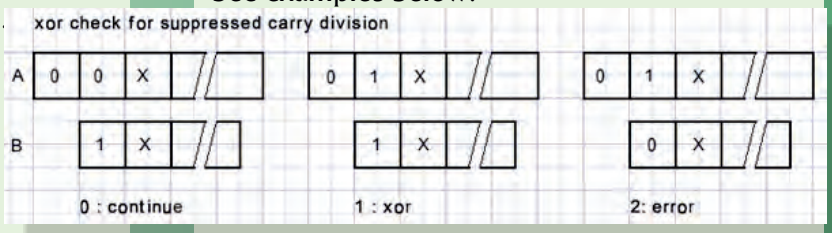
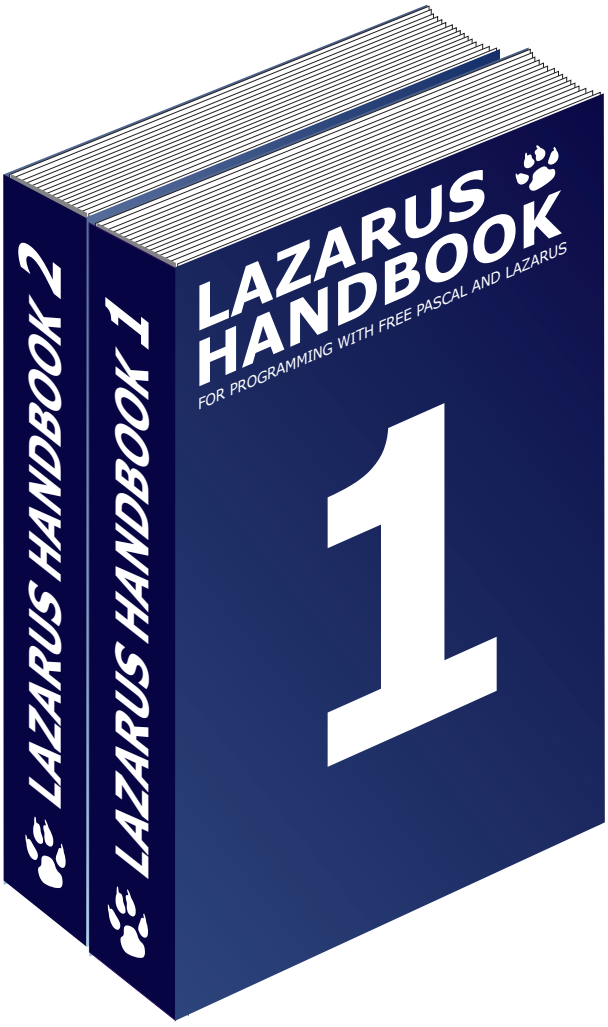
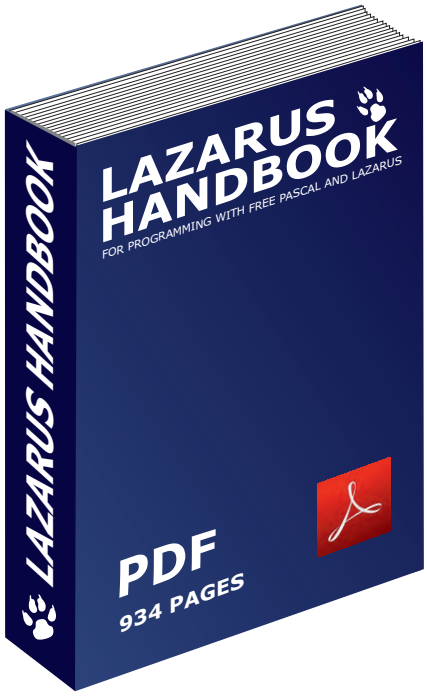


Figure 7: Examples
Please refer to the source code for details.
The `rotation button` *component was described some years ago in Blaise Pascal Magazine.*
For the `microseconds timer and the array button component` *(used to select operations) see my book:* "Computer Math and Games in Pascal".
The buttons and extra examples are available via download from you registered subscription address:

`https://www.blaisepascalmagazine.eu/your-downloads/`





Sewn POCKET edition with PDF

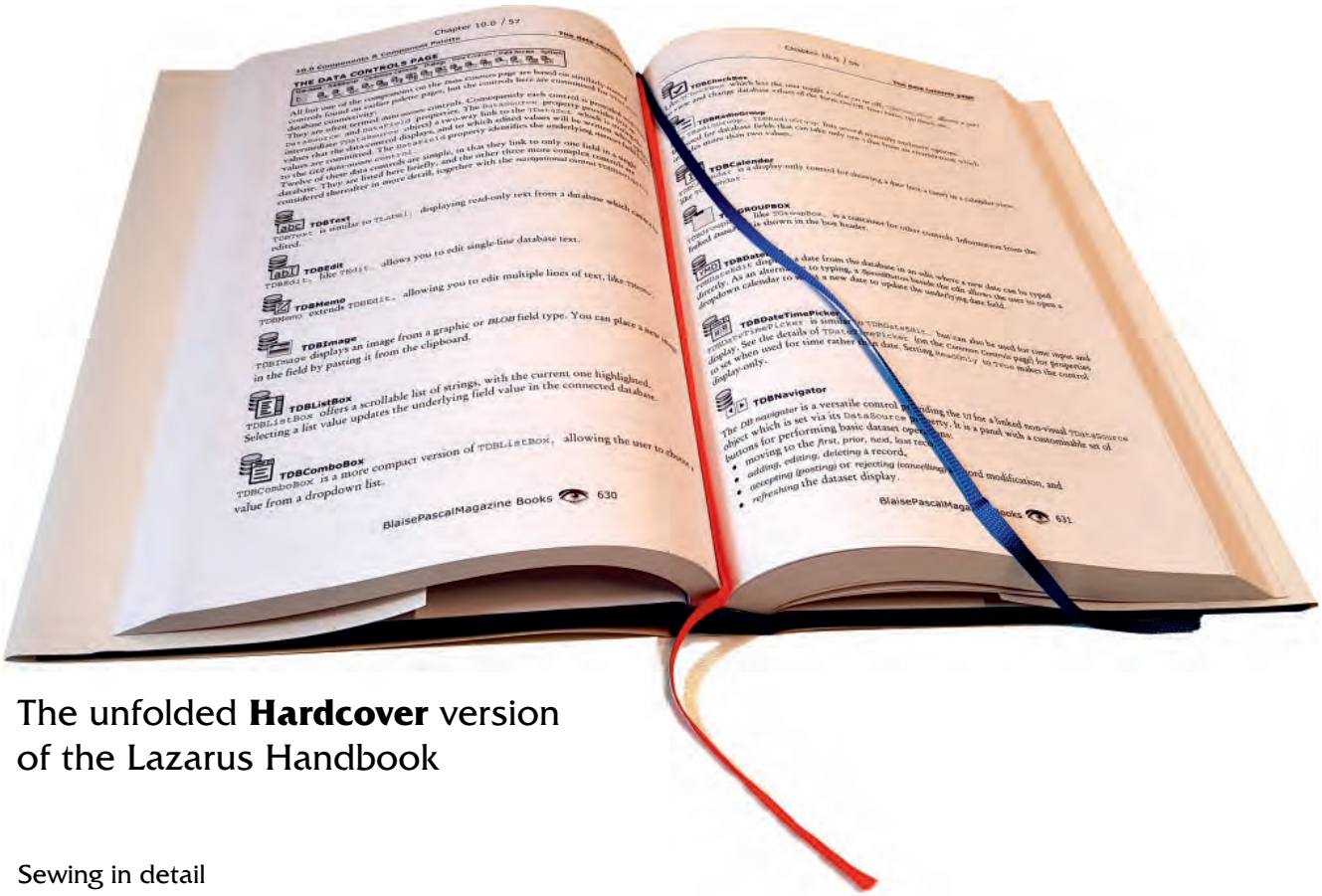
50 euro ex Vat and shipping
inc. PDF

LAZARUS HANDBOOK POCKET edition is also sewn, (see next page) to make sure you will not lose pages after a while. It is printed on 100 percent guaranteed FSC certified Paper

INCLUDED:
bookmark - creditcard - usb stick which contains the personalized pdf file of the book and the extra program files. So you have your electronic as wel the printed book in one product.

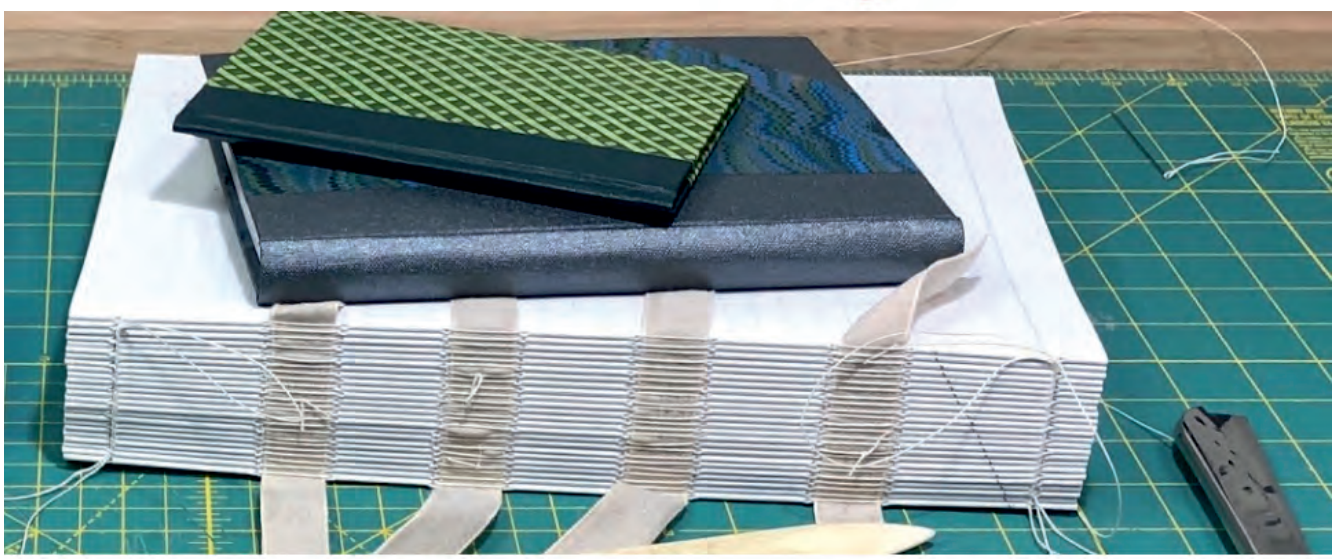
934 pages in two books

For ordering go to:
`https://www.blaisepascalmagazine.eu/product-category/books/`



The unfolded **Hardcover** version of the Lazarus Handbook

Sewing in detail



Details of the book:
CONTENT
https://www.blaisepascalmagazine.eu/blog/lazarus-handbook-content-overview-21-2-2020/
ALL FIRST PAGES
https://www.blaisepascalmagazine.eu/blog/lazarus-handbook-all-first-pages-of-chapters/

For ordering go to:
https://www.blaisepascalmagazine.eu/product-category/books/

Database Workbench 5.78 a must have tool for Databases Page 1/10

By Detlef Overbeek

INTRODUCTION

I have tested and tried many "Database Structure Design Packages" in the past.
Most of them were nice but not convincing, and yet they had too few possibilities to create all aspects of the design, construction and overview.
Eventually I ended up at "DataBase Workbench" and there is a huge amount of tools attached to it.
It is modular (suitable per group of databases) and also expandable and cheap. Especially when you realize that you already have to pay 100 euros for a simple app. To convince yourself go to
`https://www.upscene.com/database_workbench/`
or
`https://www.upscene.com/downloads/` .

In this article I'll give you some examples of how to work with it and what I learned from it. One of the most inspiring features is the table overview.

The main new features which are added in this version are:

❶ SQL Azure support
❷ Windows Terminal Server support (*multiple users using Database Workbench via remote terminals*)
❸ Data Compare tool
❹ PostgreSQL support, including stored routine debugger, SSQL and SSH tunnel connections
❺ Report Writer tool for creating customer data-bases reports
❻ MySQL Azure support
❼ command line version of the DataPump tool
❽ 'favorite databases' in Enterprise Manager
❾ MySQL 'prepared statements' en 'session variables' support in stored procedure debugger

❶ SQL Azure support

Microsoft Azure SQL Database (formerly SQL Azure, SQL Server Data Services, SQL Services, and Windows Azure SQL Database) is a managed cloud database (PaaS Platform as a service or application platform as a service (aPaaS)) provided as part of Microsoft Azure.
**A cloud database is a database that runs on a cloud computing platform, and access to it is provided as a service. Managed database services take care of scalability, backup, and high availability of the database. Azure SQL Database is a managed database service which is different from AWS RDS which is a container service.*
Please Note: My personal opinion about Cloud databases is negative: It is nothing but a marketing concept of what in earlier days was server technique. The problem is that as soon as you use this external space and it is not fully under your control: it is vulnerable. The data are not yours any more. In practice these data can get lost, stolen, or misused by others. My advice is create your own Cloud! So that you have full control.

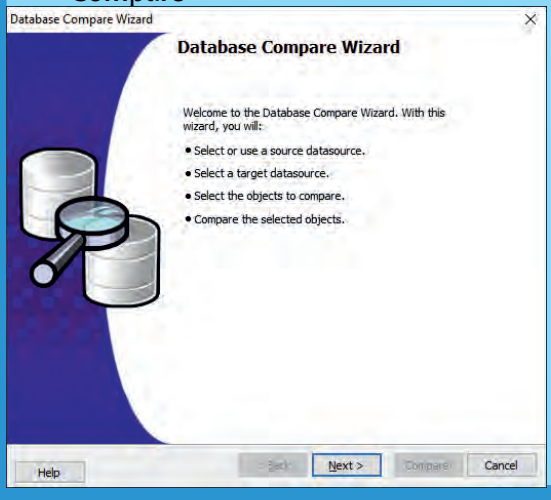
❷ **Windows Terminal Server** support

(*multiple users using Database Workbench via remote terminals*)

❸ Data Compare tool

This is a wizard that makes the comparison of database possible:

- Select or use a source.
- Select the target
- Select the objects to compare and then
- Compare



❹ **PostgreSQL support,**
including stored routine debugger, SSQL and SSH tunnel connections

PostgreSQL s a free and open-source relational database management system (RDBMS) emphasizing extensibility and SQL compliance. It was originally named POSTGRES, referring to its origins as a successor to the Ingres database developed at the University of California, Berkeley. In 1996, the project was renamed to PostgreSQL to reflect its support for SQL.

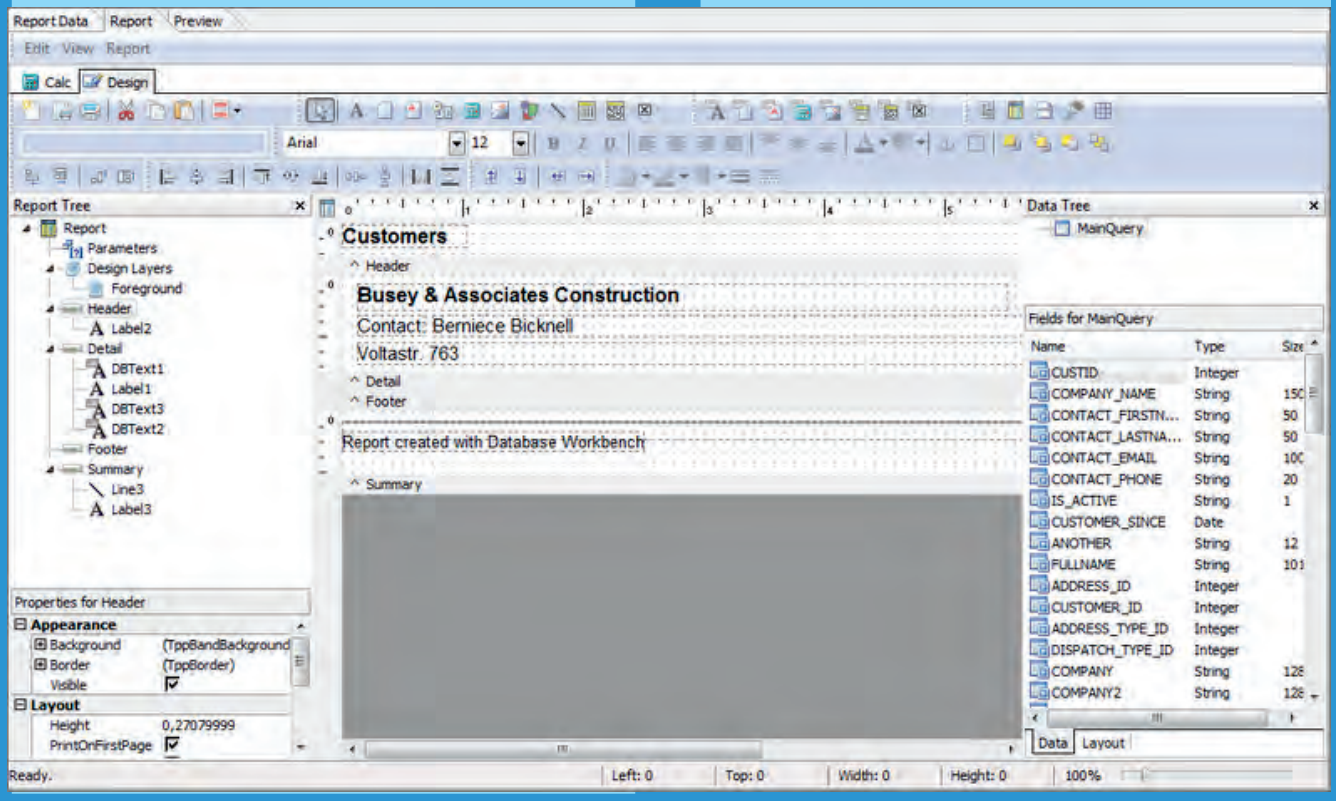
PostgreSQL features transactions with Atomicity, Consistency, Isolation, Durability (ACID) properties, automatically updatable views, materialized views, triggers, foreign keys, and stored procedures. It is designed to handle a range of workloads, from single machines to data warehouses or Web services with many concurrent users. It is the default database for macOS Server, and is also available for Linux, FreeBSD, OpenBSD, and Windows.

❺ **Report Writer** tool for creating customer data-bases reports

❻ MySQL Azure support
MySQL is an open-source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language.

MySQL has stand-alone clients that allow users to interact directly with a MySQL database using SQL, but more often MySQL is used with other programs to implement applications that need relational database capability. MySQL is a component of the "LAMP" web application software stack (and others), which is an acronym for Linux, Apache, MySQL, Perl/PHP/Python. MySQL is used by many database-driven web applications, including Drupal, Joomla, phpBB, and WordPress. MySQL is also used by many popular websites, including Facebook, Flickr, MediaWiki, Twitter and YouTube.

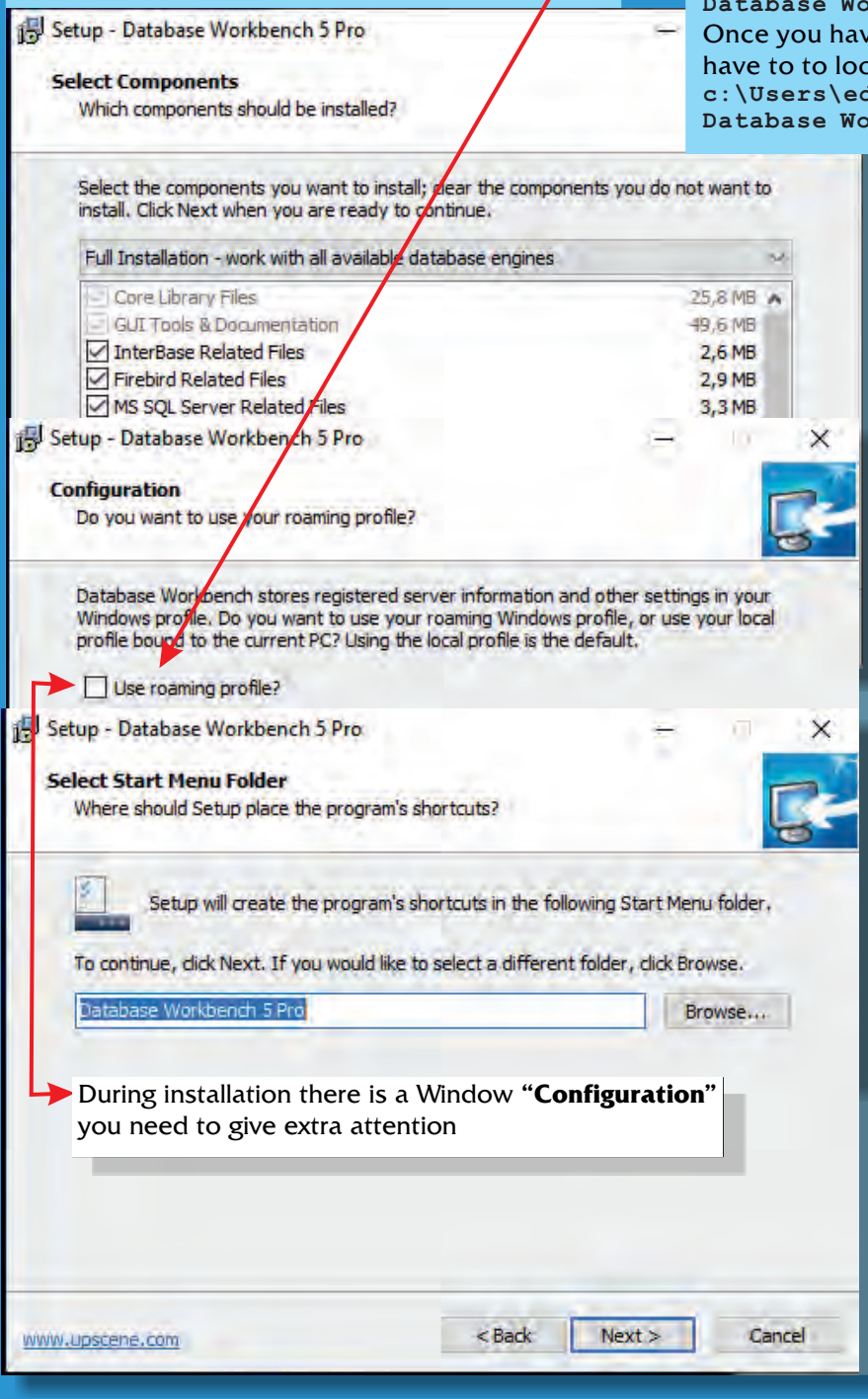
❼ command line version of the DataPump tool



❽ 'favorite databases' in Enterprise Manager Puts your most used database in front.

❾ MySQL 'prepared statements' and'session variables' support in stored procedure debugger

INSTALLING



During installation there is a Window **"Configuration"** you need to give extra attention

Roaming:
This question has to be thought of first. If you only use your database locally there is no need for this, otherwise if you want to use the installation from start on a server you better choose this option. Here are the differences:
The details of your settings will be saved under
`c:\Users\edito\AppData\Local\Database Workbench 5 Pro\.`
Once you have chosen for roaming you will have to to look under
`c:\Users\edito\AppData\Roaming\Database Workbench 5 Pro\.`

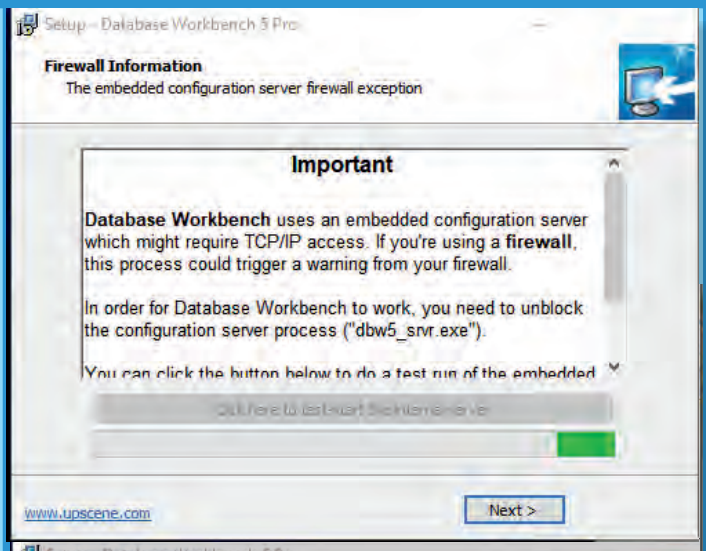
Explanation of Roaming
Technical definition
In more technical terms, roaming refers to the ability for a cellular customer to automatically make and receive voice calls, send and receive data, or access other services, including home data services, when travelling outside the geographical coverage area of the home network, by means of using a visited network. For example: should a subscriber travel beyond their cell phone company's transmitter range, their cell phone would automatically hop onto another phone company's service, if available.
The process is supported by the Telecommunication processes of mobility management, authentication, authorization and accounting billing procedures (known as AAA or 'triple A').

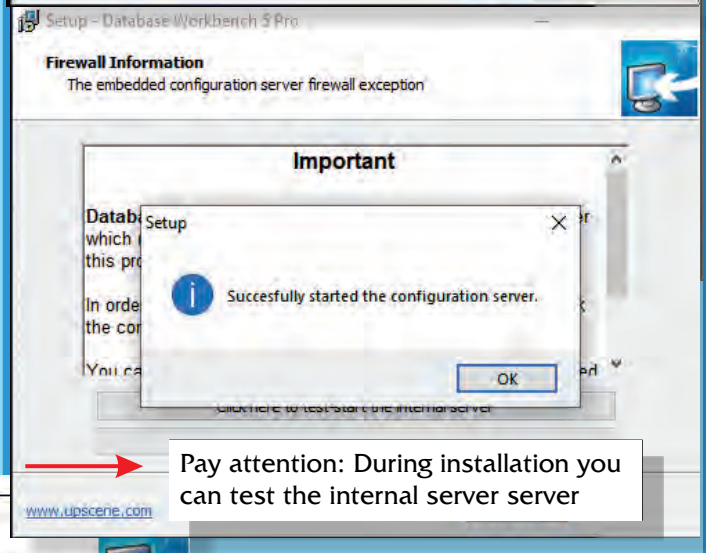
Roaming in general
Roaming is divided into "SIM-based roaming" and "username/password-based roaming", whereby the technical term "roaming" also encompasses roaming between networks of different network standards, e.g. WLAN (Wireless Local Area Network) or GSM (Global System for Mobile Communications).

Device equipment and functionality, such as SIM card capability, antenna and network interfaces, and power management, determine the access possibilities.
Using the example of ***WLAN/GSM roaming,*** *the following scenarios can be differentiated*

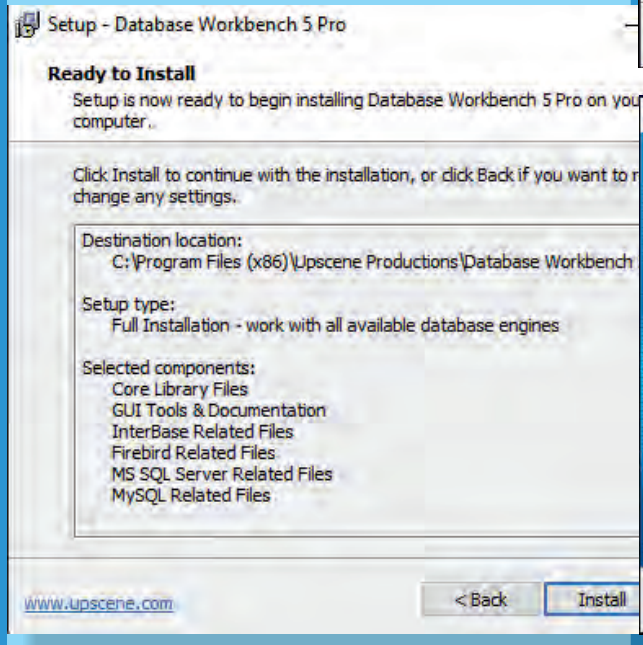
- ***SIM-based*** *(roaming):*
  GSM subscriber roams onto a public WLAN operated by:
  their GSM operator, or another operator who has a roaming agreement with their GSM operator.
- ***Username/password based*** *roaming:*
  GSM subscriber roams onto a public WLAN operated by: their GSM operator, or another operator who has a roaming agreement with their GSM operator.

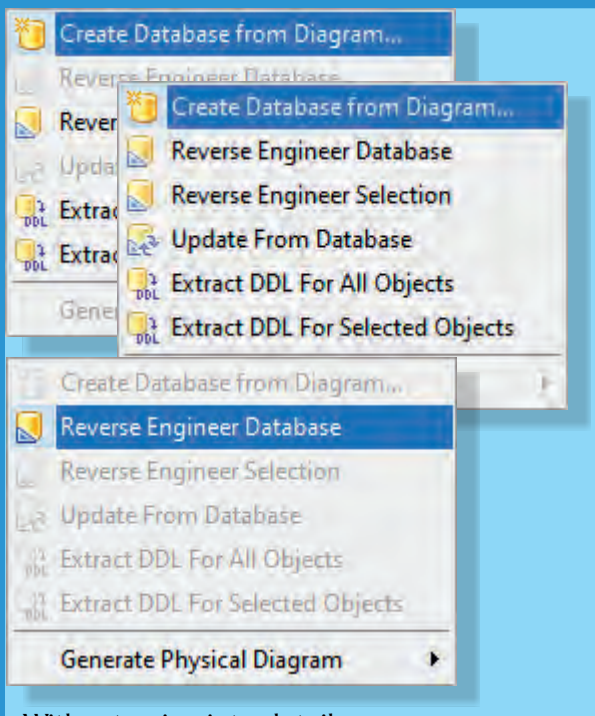
Although these user/ network scenarios focus on roaming from GSM network operator's networks, clearly roaming can be bi-directional, i.e. from public WLAN operators to GSM networks.
Traditional roaming in networks of the same standard, e.g. from a WLAN to a WLAN or a GSM network to a GSM network,
has already been described above and is likewise defined by the foreignness of the network based on the type of subscriber entry in the home subscriber register.
In the case of session continuity, seamless access to these services across different access types is provided.



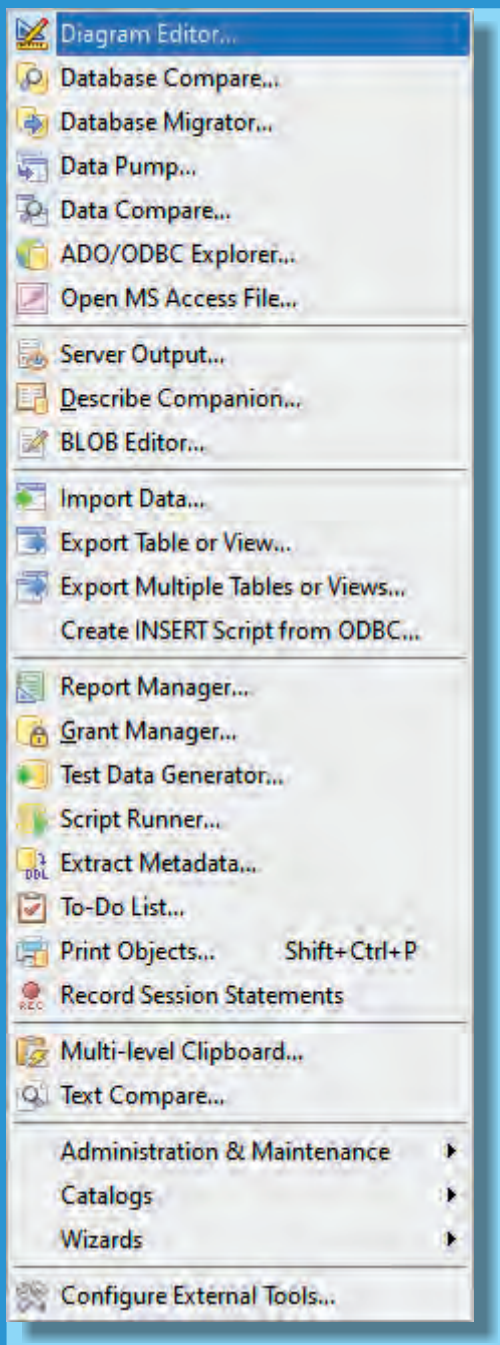


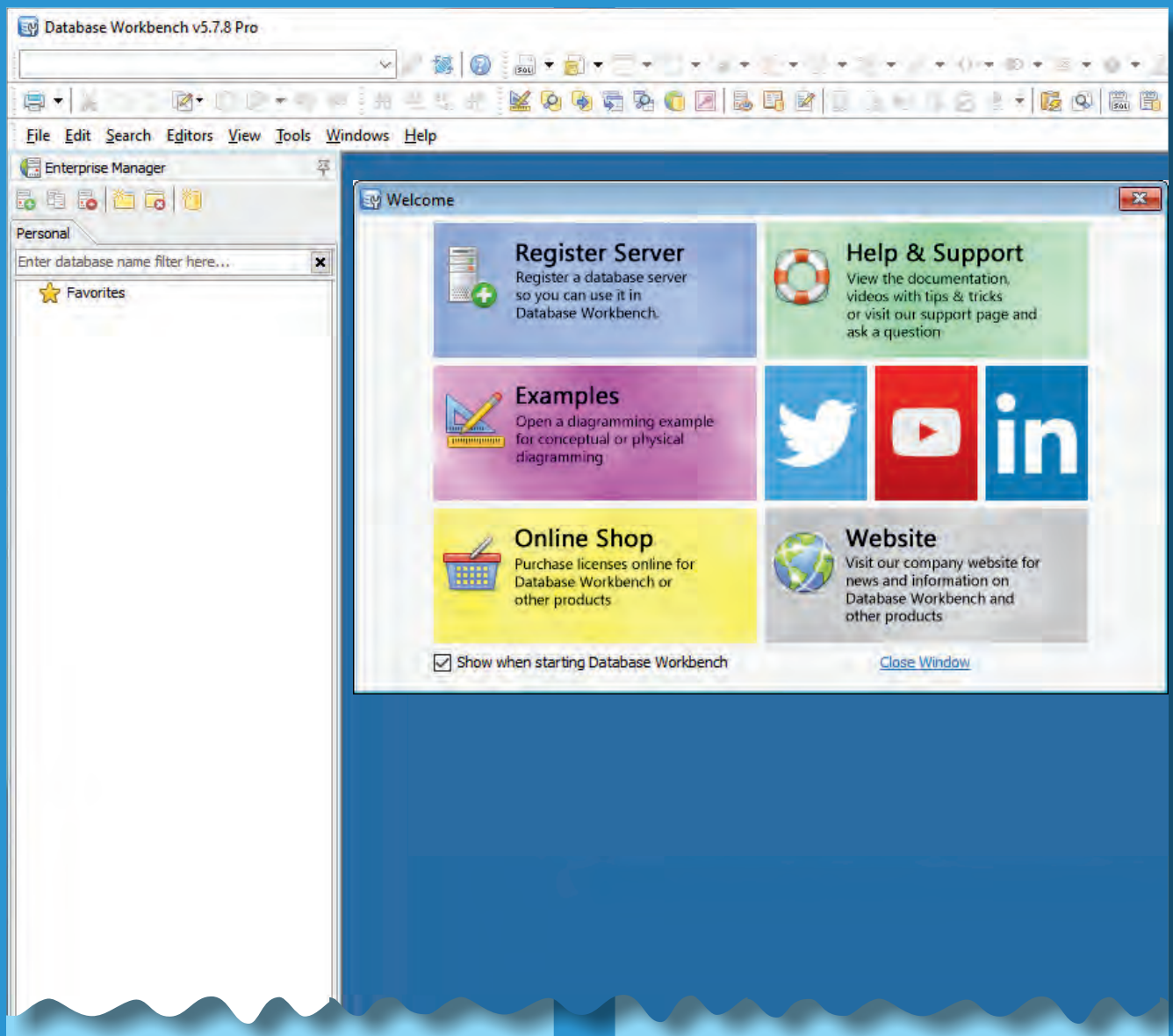
Pay attention: During installation you can test the internal server server



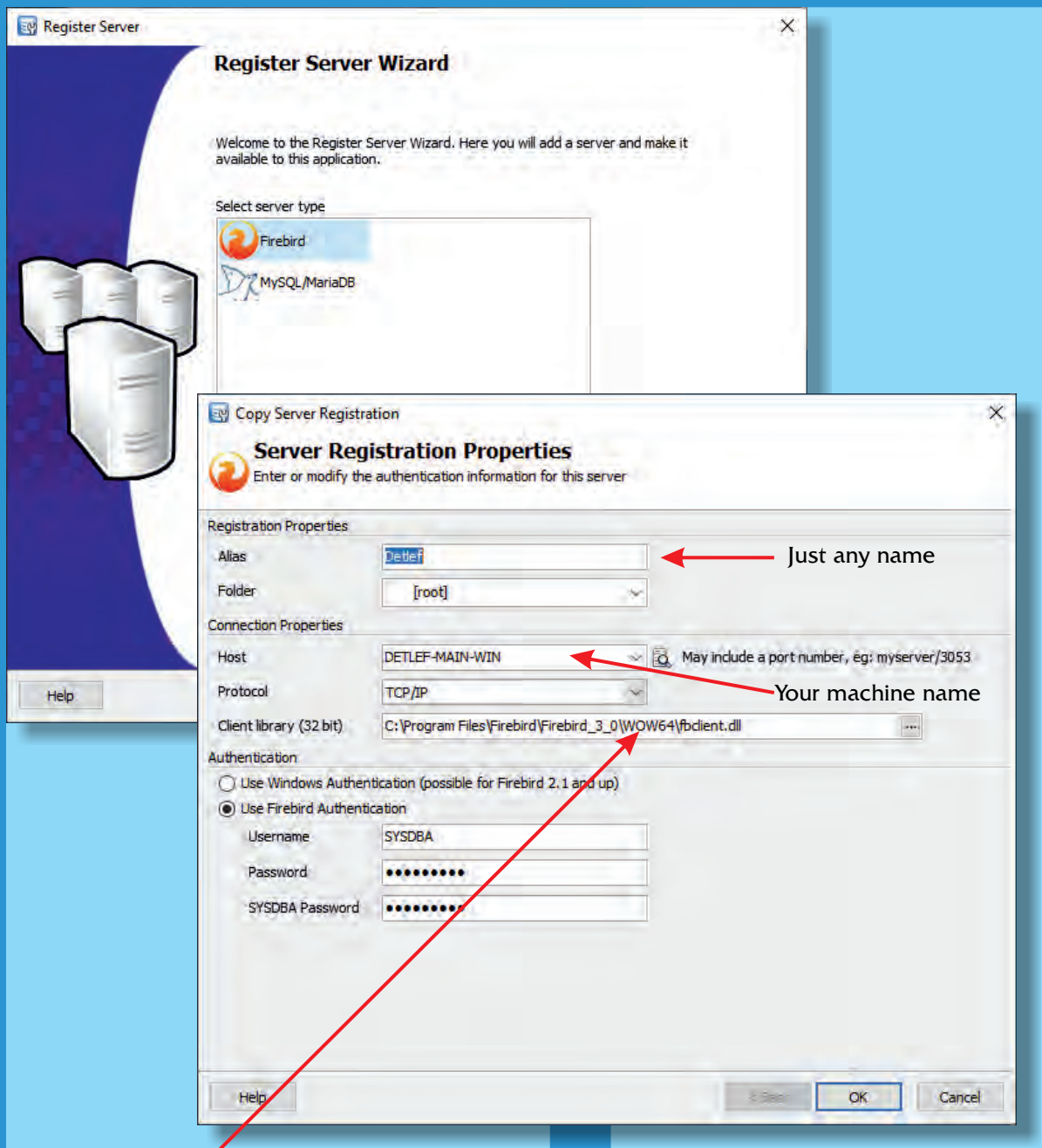


Without going into details:
If you have a Database already, you can let it be analysed from the database towards an overview, or the other way around.
As you can see you can even automatically generate a Diagram or let the Diagram create your database, make updates etc.
A very powerful tool. As you will see in the next pages you can even make some kind of graphic design from the Diagram, which is very instructive if you need to document it. Colorize special tables, change the font, group the tables. (See page 51 article page 9)
Fantastic!
The tools list in the right column is very instructive to show and find anything you might need or want. If you want to know more about a special item you use just as in Delphi and Lazarus F1 and the associated help will appear. (See page 52 article page 10)





The still empty IDE Integrated Dvelopment Environment. On the next page you can see the possible choices I made.

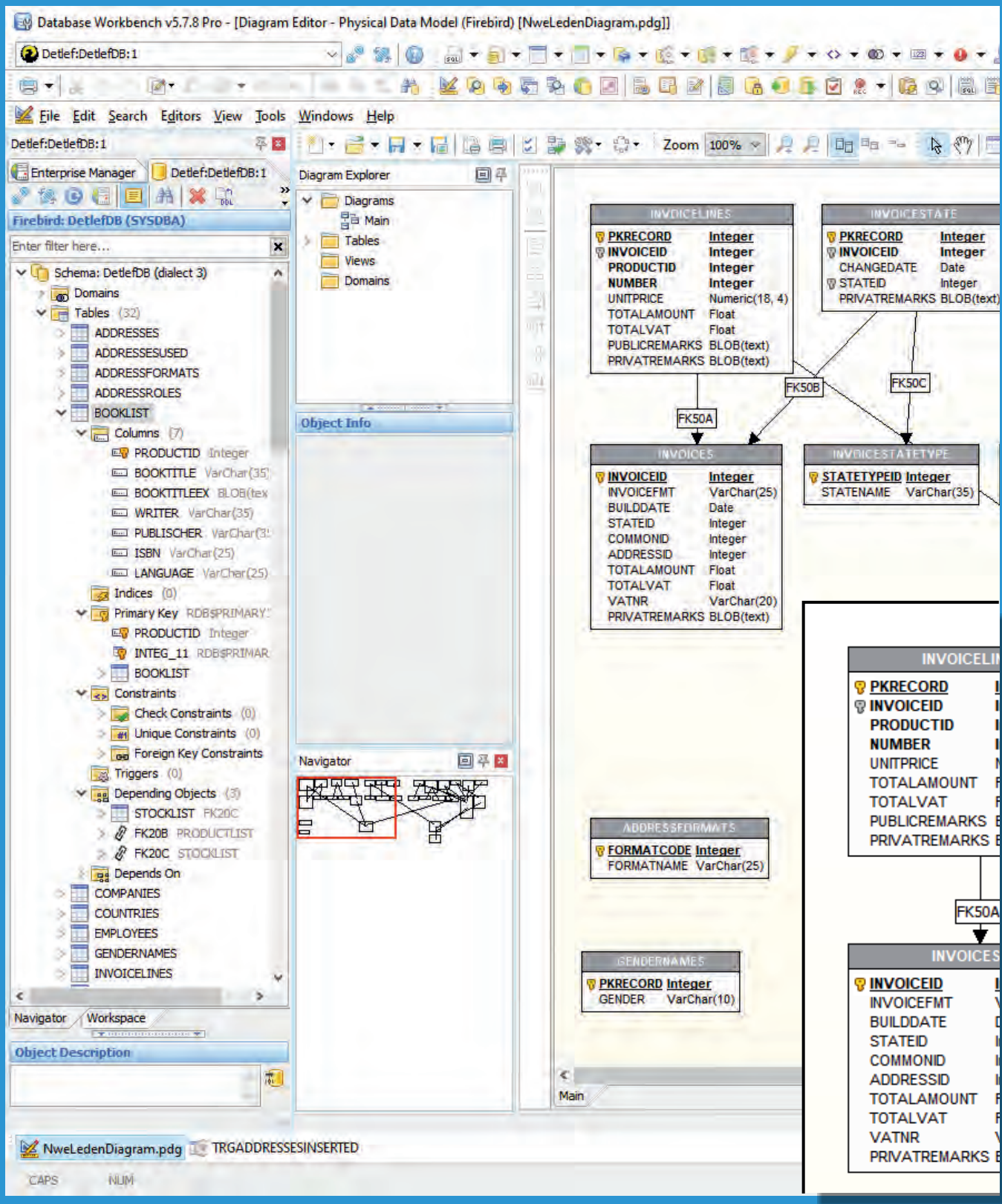


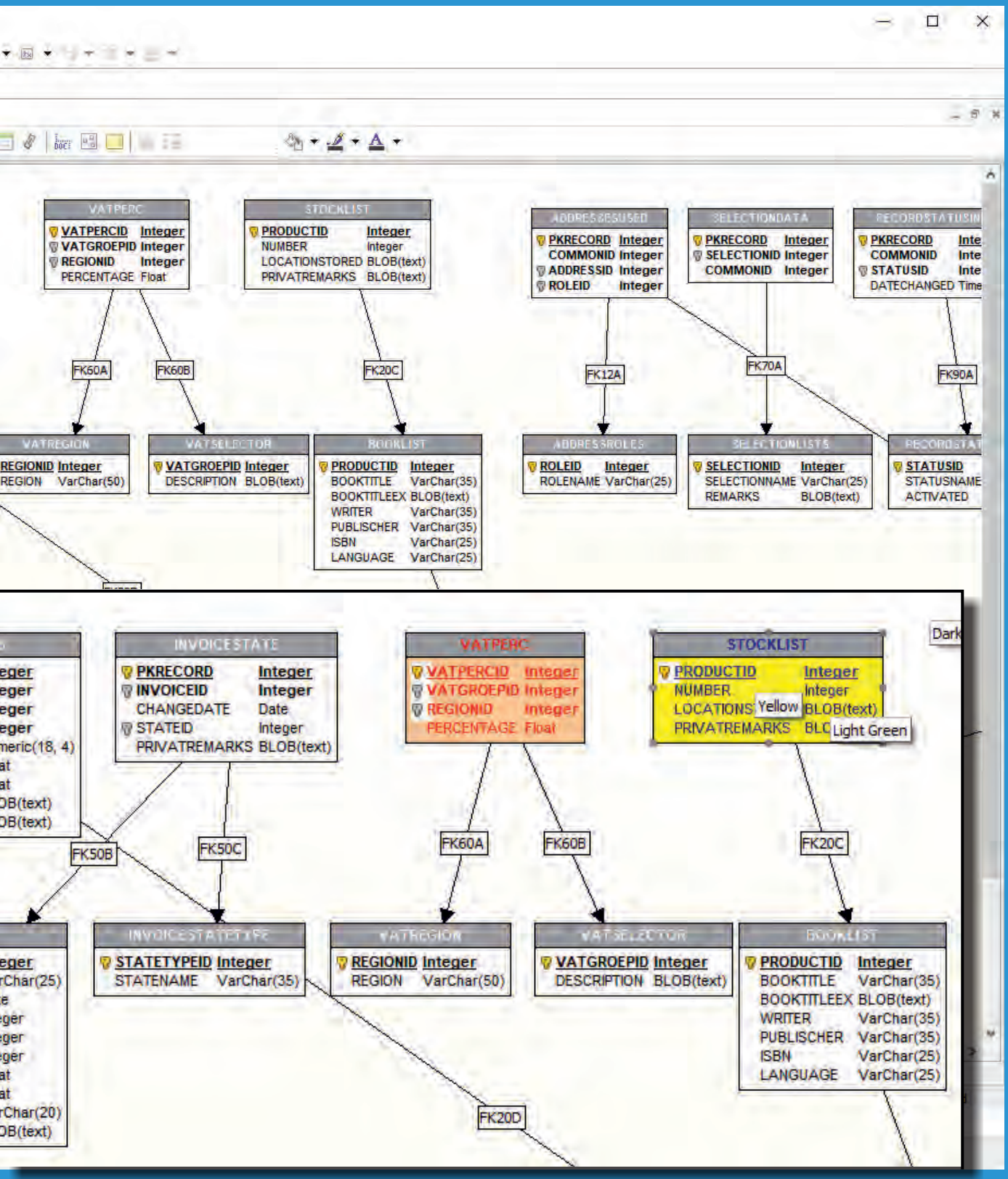
I chose Firebird and after clicking this wizard pops up. The client Library has to show the path to the correct dll. In this case the first DLL youll find is a 64 bit DLL. You'll need the 32 bit dll which is the `\WOW64\` dir.

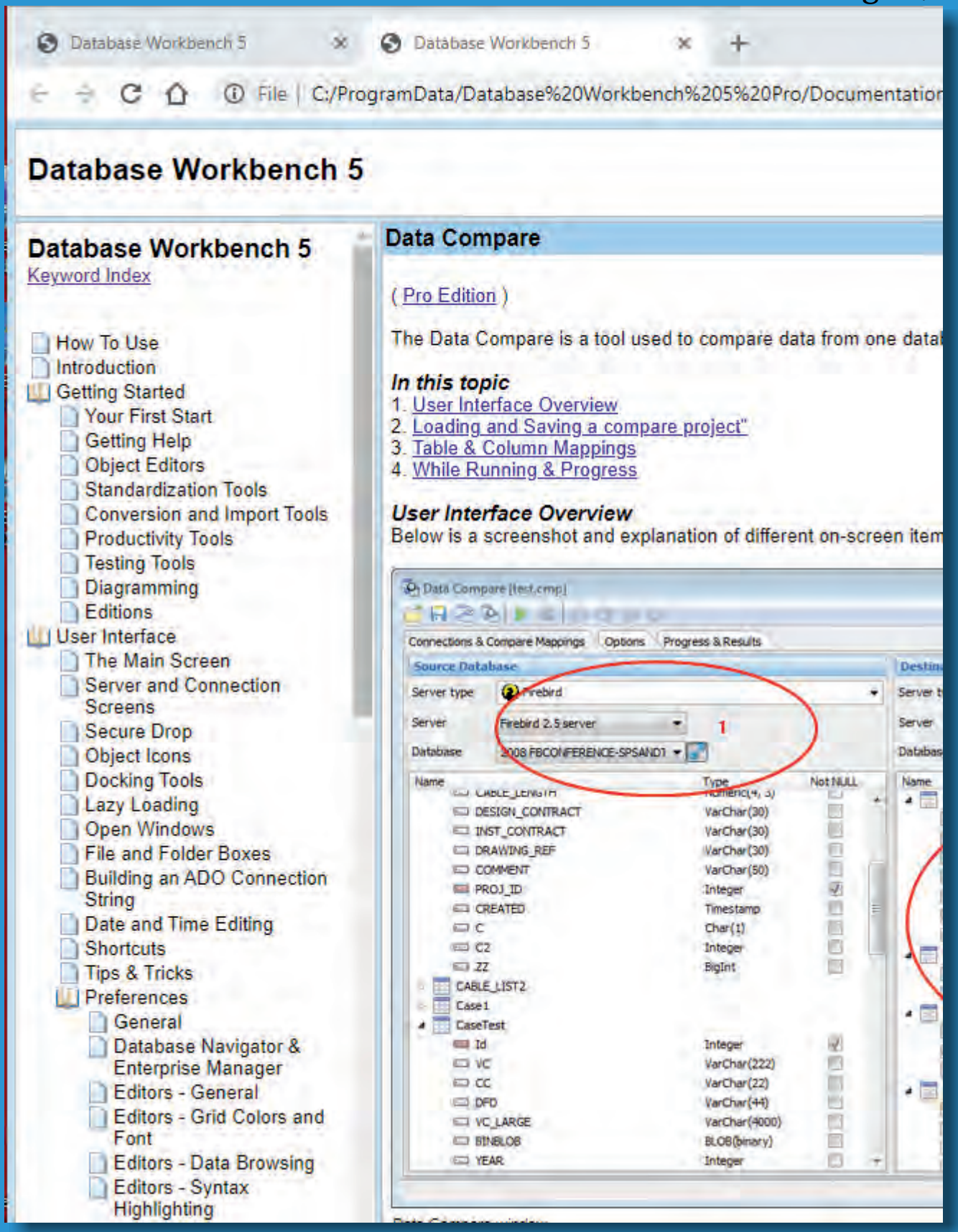
WOW64

In computing on Microsoft platforms, WoW64 (`Windows 32-bit On Windows 64-bit`) is a subsystem of the Windows operating system capable of running 32-bit applications on 64-bit Windows.

It is included in all 64-bit versions of Windows—including Windows XP Professional x64 Edition, IA-64 and x64 versions of Windows Server 2003, as well as 64-bit versions of Windows Vista, Windows Server 2008, Windows 7, Windows 8, Windows Server 2012, Windows 8.1 and Windows 10. In Windows Server 2008 R2 Server Core, it is an optional component, but not in Nano Server[clarification needed]. WoW64 aims to take care of many of the differences between 32-bit Windows and 64-bit Windows, particularly involving structural changes to Windows itself.


Database Workbench v5.7.8 Pro - [Diagram Editor - Physical Data Model (Firebird) [NweLedenDiagram.pdg]]
File Edit Search Editors View Tools Windows Help
Enterprise Manager
Firebird: DetlefDB (SYSDBA)
Schema: DetlefDB (dialect 3)
Diagram Explorer
Object Info
Navigator
Object Description
INVOICELINES
INVOICESTATE
INVOICES
INVOICESTATETYPE
ADDRESSFORMATS
GENDERNAMES
FK50A
FK50B
FK50C
NweLedenDiagram.pdg
TRGADDRESSESINSERTED


VATPERC
VATPERCID Integer
VATGROEPID Integer
REGIONID Integer
PERCENTAGE Float
STOCKLIST
PRODUCTID Integer
NUMBER Integer
LOCATIONSTORED BLOB(text)
PRIVATREMARKS BLOB(text)
FK60A
FK60B
FK20C
FK12A
FK70A
FK90A
VATREGION
REGIONID Integer
REGION VarChar(50)
VATSELECTOR
VATGROEPID Integer
DESCRIPTION BLOB(text)
BOOKLIST
PRODUCTID Integer
BOOKTITLE VarChar(35)
BOOKTITLEEX BLOB(text)
WRITER VarChar(35)
PUBLISCHER VarChar(35)
ISBN VarChar(25)
LANGUAGE VarChar(25)
ADDRESSROLES
ROLEID Integer
ROLENAME VarChar(25)
SELECTIONLISTS
SELECTIONID Integer
SELECTIONNAME VarChar(25)
REMARKS BLOB(text)
RECORDSTAT
STATUSID
STATUSNAME
ACTIVATED
INVOICESTATE
PKRECORD Integer
INVOICEID Integer
CHANGEDATE Date
STATEID Integer
PRIVATREMARKS BLOB(text)
FK50B
FK50C
INVOICESTATETYPE
STATETYPEID Integer
STATENAME VarChar(35)
FK20D
Yellow
Light Green


Database Workbench 5
Database Workbench 5
File | C:/ProgramData/Database%20Workbench%205%20Pro/Documentation
Database Workbench 5
Database Workbench 5
Keyword Index
How To Use
Introduction
Getting Started
Your First Start
Getting Help
Object Editors
Standardization Tools
Conversion and Import Tools
Productivity Tools
Testing Tools
Diagramming
Editions
User Interface
The Main Screen
Server and Connection Screens
Secure Drop
Object Icons
Docking Tools
Lazy Loading
Open Windows
File and Folder Boxes
Building an ADO Connection String
Date and Time Editing
Shortcuts
Tips & Tricks
Preferences
General
Database Navigator & Enterprise Manager
Editors - General
Editors - Grid Colors and Font
Editors - Data Browsing
Editors - Syntax Highlighting
Data Compare
(Pro Edition)
The Data Compare is a tool used to compare data from one datal
In this topic
1. User Interface Overview
2. Loading and Saving a compare project"
3. Table & Column Mappings
4. While Running & Progress
User Interface Overview
Below is a screenshot and explanation of different on-screen item
Data Compare [test.cmp]
Connections & Compare Mappings
Options
Progress & Results
Source Database
Server type
Firebird
Server
Firebird 2.5 server
1
Database
2008 FBCONFERENCE-SPSAND1
Name
Type
Not NULL
DESIGN_CONTRACT
VarChar(30)
INST_CONTRACT
VarChar(30)
DRAWING_REF
VarChar(30)
COMMENT
VarChar(50)
PROJ_ID
Integer
CREATED
Timestamp
C
Char(1)
C2
Integer
ZZ
BigInt
CABLE_LIST2
Case1
CaseTest
Id
Integer
VC
VarChar(222)
CC
VarChar(22)
DFD
VarChar(44)
VC_LARGE
VarChar(4000)
BINBLOB
BLOB(binary)
YEAR
Integer

starter expert

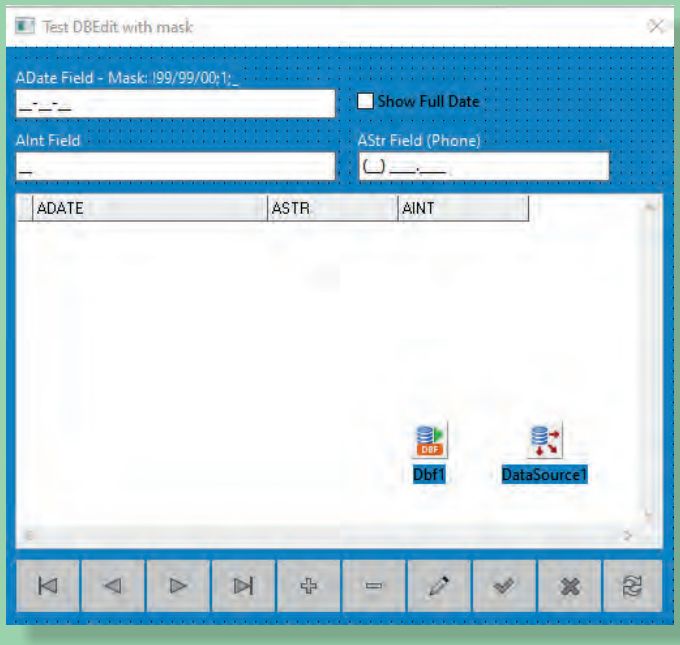
This is a Lazarus version, the Delphi version will be published in the next issue 91.

INTRODUCTION:
The maskedit is a very nice component to use but needs some special knowledge.
In this lazarus example some of the possibilities are explained, the code is available from your personal subscription download address:
https://www.blaisepascalmagazine.eu/your-downloads/
and you need to login first.

Running the app means you will also create a small database on the fly. You ca see what you need for
special masks to force the edit field to be filled in correctly. I need to find out why the Checkbox prevents the font to become white.
This is what you need to put on the form:

3 Labels
3 DbEdit
1 DbGrid
1 DB Navigator
1 CheckBox
1 DBF Component
1 DataSource

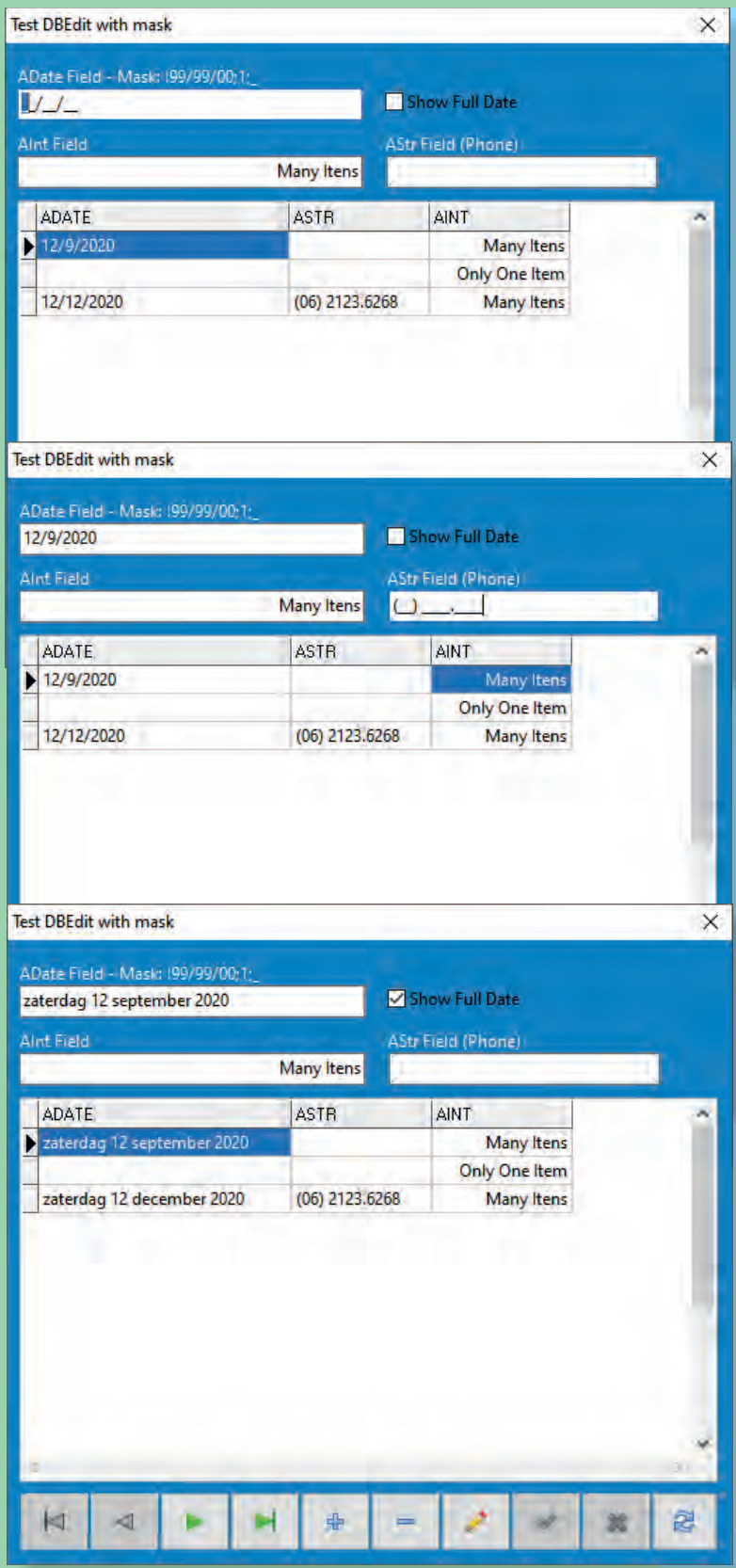
Please note how this example shows you to create a Database.



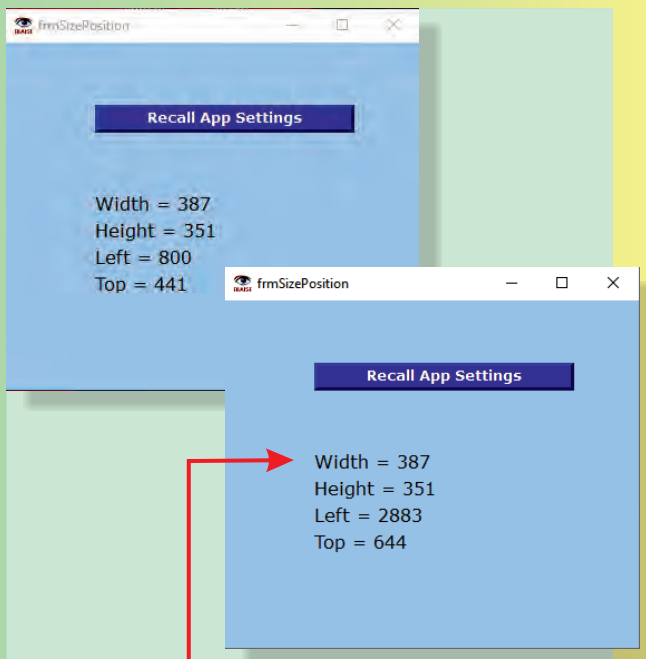

```
procedure TForm1.Dbf1ADATESetText(Sender: TField;
        const aText: string);
var
 FixedStr: String;
begin
 //workaround to fpc bug 15039
 FixedStr := AnsiReplaceStr(aText, ' ', '');
 Sender.AsString := FixedStr;
end;

procedure TForm1.ShowLongDateCheckBoxChange
        (Sender: TObject);
var
 DateField: TDateTimeField;
begin
 DateField :=
        Dbf1.FieldByName('ADate') as TDateTimeField;
 if ShowLongDateCheckBox.Checked then
  DateField.DisplayFormat :=
DefaultFormatSettings.LongDateFormat
 else
  DateField.DisplayFormat := '';
end;
```

```
procedure TForm1.FormCreate(Sender: TObject);
begin
 DefaultFormatSettings.ShortDateFormat := 'd/m/yyyy';
 DefaultFormatSettings.DateSeparator := '/';
 if not FileExistsUTF8(Dbf1.TableName) then
 begin
  Dbf1.FieldDefs.Clear;
  Dbf1.FieldDefs.Add('ADate', ftDate);
  Dbf1.FieldDefs.Add('AStr', ftString, 50);
  Dbf1.FieldDefs.Add('AInt', ftLargeint);
  Dbf1.CreateTable;
  //add some data
  Dbf1.Open;
  Dbf1.Append;
  Dbf1.FieldByName('ADate').AsString := '24/12/2020';
  Dbf1.Post;
  Dbf1.Append;
  Dbf1.FieldByName('AInt').AsInteger := 1;
  Dbf1.Post;
  Dbf1.Append;
  Dbf1.FieldByName('ADate').AsString := '24/12/2020';
  Dbf1.FieldByName('AInt').AsInteger := 30;
  Dbf1.Post;
 end
 else
  Dbf1.Open;
end;
```

```
 procedure TForm1.Dbf1AINTGetText(Sender: TField;
var aText: string; DisplayText: Boolean);
begin
 if DisplayText then
 begin
  if Sender.IsNull then
   aText := '(Undefined)'
  else if Sender.AsInteger = 0 then
   aText := 'No Item'
  else if Sender.AsInteger = 1 then
   aText := 'Only One Item'
  else if Sender.AsInteger < 10 then
   aText := 'Few Itens'
  else
   aText := 'Many Itens';
 end
 else
  aText := Sender.AsString;
end;
```


Test DBEdit with mask
ADate Field - Mask: !99/99/00;1;_
Show Full Date
AInt Field
Many Items
AStr Field (Phone)
ADATE
ASTR
AINT
12/9/2020
Many Items
Only One Item
12/12/2020
(06) 2123.6268
Many Items
Test DBEdit with mask
ADate Field - Mask: !99/99/00;1;_
12/9/2020
Show Full Date
AInt Field
Many Items
AStr Field (Phone)
(__) ___-____
ADATE
ASTR
AINT
12/9/2020
Many Items
Only One Item
12/12/2020
(06) 2123.6268
Many Items
Test DBEdit with mask
ADate Field - Mask: !99/99/00;1;_
zaterdag 12 september 2020
Show Full Date
AInt Field
Many Items
AStr Field (Phone)
ADATE
ASTR
AINT
zaterdag 12 september 2020
Many Items
Only One Item
zaterdag 12 december 2020
(06) 2123.6268
Many Items

starter expert



In this series we want to show and create little nice code creations that can be helpful and very easy to replicate.

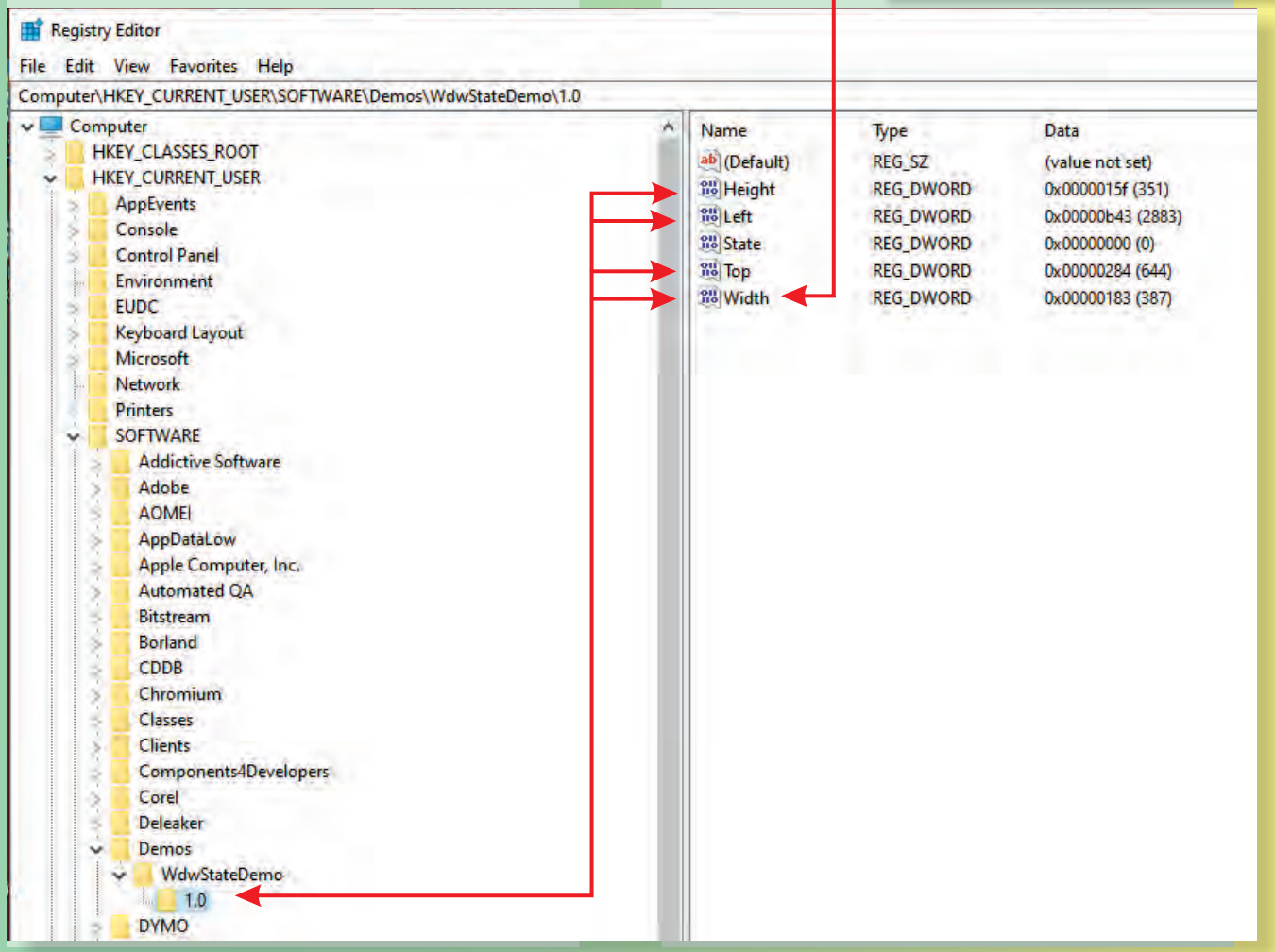
INTRODUCTION:
This example allows your application to be started and saved at the place you want and at the size you want. It is handled trough the Registry of Windows. There are two functions involved:

```
Function ReadIntFromReg
Function NewWindowProc
```

You need to add to the „Uses" clause

```
Registry, ColorButton;
```

There is a procdure onCreate which reads from the Registry all details you can see here.
The `ClrBtnWindowsetting` reads the new settings. The `onFormshow` sets the old window



```
unit UFormSizePosistion;

interface

uses
 Winapi.Windows, Winapi.Messages, System.SysUtils, System.Variants, System.Classes,
 Vcl.Graphics, Vcl.Controls, Vcl.StdCtrls, Vcl.Forms, Vcl.Dialogs,Vcl.ExtCtrls,
 TypInfo, Registry, ColorButton; //need to be added

const
  CRegKey = 'Software\Demos\WdwStateDemo\1.0'; //  This is important because this is the input to the registry
  //  If you create several programs with this remember  to create your own constant
  //  CRegKey = the actual replacement registration in the Registry
Var
 OldWindowProc   : Pointer; //Variable for the old windows proc
 MyMsg       : DWord;
type
 TfrmSizePosition = class(TForm)
  ClrBtnWindowSetting: TColorButton;
  Label1: TLabel;
  Label2: TLabel;
  Label3: TLabel;
  Label4: TLabel;
  procedure FormCreate(Sender: TObject);
  procedure ClrBtnWindowSettingClick(Sender: TObject);
  procedure FormShow(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 frmSizePosition: TfrmSizePosition;

implementation

{$R *.dfm}

function ReadIntFromReg(Reg: TRegistry; Name: string; Def: Integer): Integer;
 //Reads integer with given name from registry and returns it  // If no such value exists, returns Def default value
begin
   if Reg.ValueExists(Name)
   then  Result := Reg.ReadInteger(Name)
   else  Result := Def;
end;

Function NewWindowProc(WindowHandle : hWnd;
            TheMessage  : DWord ;  //LongInt
            ParamW    : LongInt;
            ParamL    : LongInt) : LongInt Stdcall;
Begin
  If TheMessage = MyMsg  Then
  Begin
  {Tell the application to restore, let it restore the form}
  SendMessage(Application.handle, WM_SYSCOMMAND, SC_RESTORE, 0);
  SetForegroundWindow(Application.Handle);
  {We handled the message - we are done}
  Result := 0;
  exit;
  End;
  {Call the original winproc}
  Result := CallWindowProc(OldWindowProc,
             WindowHandle,
             TheMessage,
             ParamW,
             ParamL);
End;
```

```
procedure TfrmSizePosition.FormCreate(Sender: TObject);
begin
 OldWindowProc := Pointer(SetWindowLong(frmSizePosition.Handle,
                  GWL_WNDPROC,
                  LongInt(@NewWindowProc)));
end;

procedure TfrmSizePosition.ClrBtnWindowSettingClick(Sender: TObject);
var
   Reg: TRegistry;           // the registry
   State: Integer;           // state of window
   Pl : TWindowPlacement;    // used for API call
   R: TRect;                 // used for wdw pos
begin

   // Calculate window's normal size and position using Windows API call - the form's Width, Height, Top
   // and Left properties will give maximized window size if form is maximised, which is not what we want here
    Pl.Length := SizeOf(TWindowPlacement);
   GetWindowPlacement(Self.Handle, @Pl);
   R := Pl.rcNormalPosition;
   Reg := TRegistry.Create;

   try
   // Open required key - and create it if it doesn't exist
   Reg.RootKey := HKEY_CURRENT_USER;
   Reg.OpenKey(CRegKey, True);
   // Write window size and position
   Reg.WriteInteger('Width', R.Right-R.Left);
   Reg.WriteInteger('Height', R.Bottom-R.Top);
   Reg.WriteInteger('Left', R.Left);
   Reg.WriteInteger('Top', R.Top);

   // Write out state of window
   // Record window state (maximised, minimised or normal)
   // - special case when minimized since form window is simply
   // hidden when minimised, and application window is actually
   // the one minimised - so we check to see if application window *is*

   Label1.Caption := 'Width = '      + IntToStr( R.Right-R.Left);
   Label2.Caption := 'Height = '     + IntToStr( R.Bottom-R.Top);
   Label3.Caption := 'Left = '       + IntToStr( R.Left);
   Label4.Caption := 'Top = '        + IntToStr( R.Top);


   if IsIconic(Application.Handle)
   then  State := Ord(wsMinimized) // minimized - write that state
   else
      // not mimimized - we can rely on window state of form
      State := Ord(Self.WindowState);
      Reg.WriteInteger('State', State);


 finally
  Reg.Free;
 end;
end;
```

```
procedure TfrmSizePosition.FormShow(Sender: TObject);
var
 Reg: TRegistry;  // the registry
 State: Integer;  // state of window
begin
 Reg := TRegistry.Create;

 try
  // Open required key - and exit it if it doesn't exist
  Reg.RootKey := HKEY_CURRENT_USER;

  if not Reg.OpenKey(CRegKey, False) then Exit;

  // Read the window size and position - designed form sizes are defaults

  Self.Width  := ReadIntFromReg(Reg, 'Width',  Self.Width);
  Self.Height := ReadIntFromReg(Reg, 'Height', Self.Height);
  Self.Left   := ReadIntFromReg(Reg, 'Left',   Self.Left);
  Self.Top    := ReadIntFromReg(Reg, 'Top',    Self.Top);

  // Now get window state and restore
  State := ReadIntFromReg(Reg, 'State', Ord(wsNormal));

  {check if window was minimised
   - we have special processing for minimized state since Delphi doesn't
   minimize windows - it uses application window instead}

 if State = Ord(wsMinimized) then
  begin
    {we need to set visible true else form won't restore properly
     - but this causes a brief display of form}
    Self.Visible := True;
    Application.Minimize;
  end
 else
  Self.WindowState := TWindowState(State);
 finally
  Reg.Free;
 end;


end;

end.
```

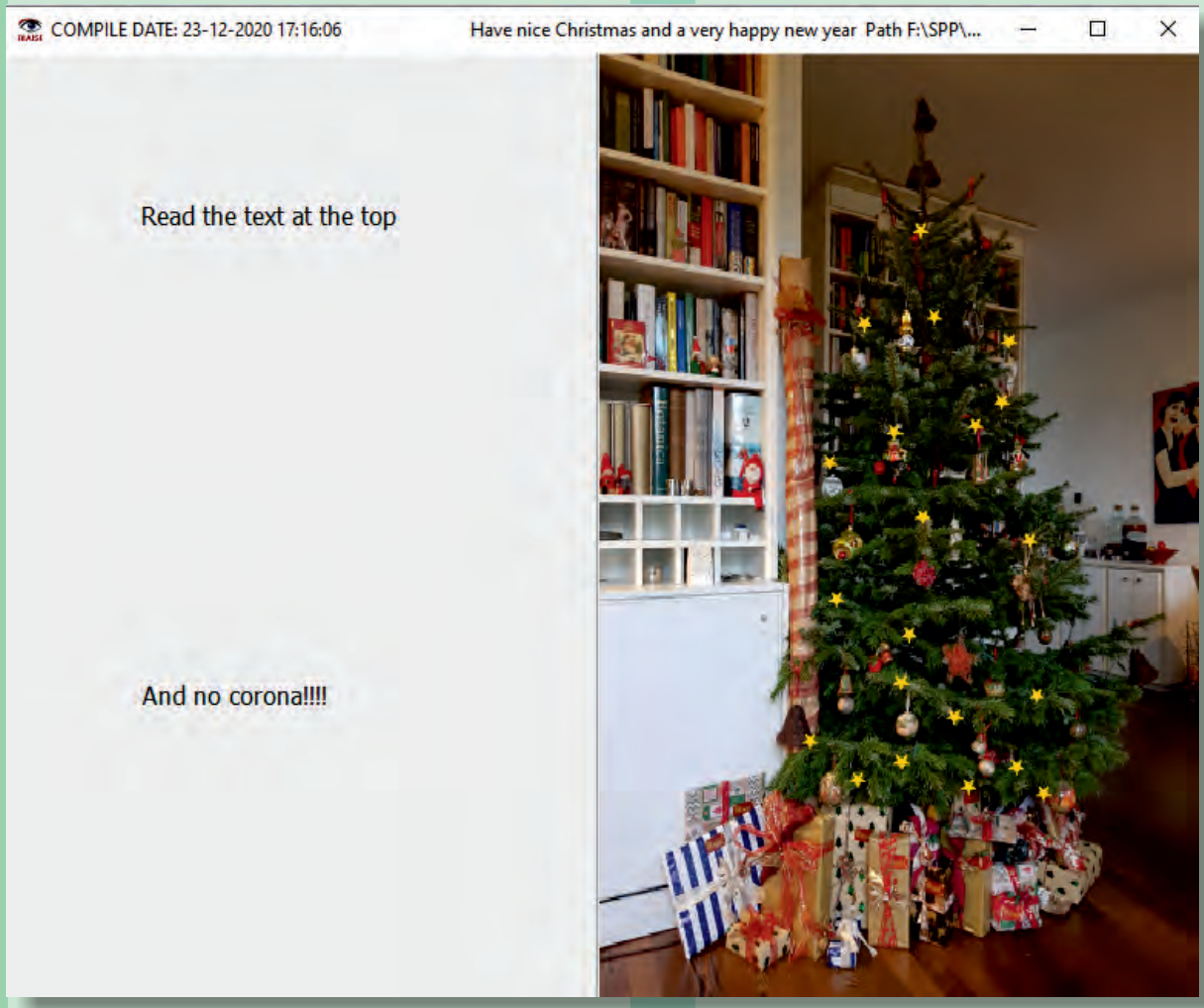
I have tried to do the same with an inifile but that doesn't handle it all very well. So to make it more robust under windows, you need to use the registry.
In Lazarus it is quite something different. I'll try to that in the next issue: that should be working on Windows, Linux and Mac.

starter expert

In this example I dropped some code that will always show the correct compile- date and path of your application.
And a Christmas an Newyears wish.
The code is downloadable of course.

For Lazarus will be available in the next issue

```
procedure TMainform.FormCreate(Sender: TObject);
Var Compiledate : String;
Begin
 Compiledate:=DateTimeToStr(FileDateToDateTime(FileAge(ExtractFileName(Application.ExeName))));

 Caption := 'COMPILE DATE:' + ' ' +Compiledate +'                    '
    + 'Have nice Christmas and a very happy new year'
    + '  Path ' + ExtractFilePath(Application.ExeName) + 'Compiledate code'; // Path
end;
```



"According to this, 2021 will be a better year than 2020. I don't need a prediction app to tell me that."

LAZARUS FOR INSTALL FOR BIG SUR (Mac)

By Detlef Overbeek

starter ▲ expert

Figure 1: Logo

INTRODUCTION

Because of the new Operating System of the macOS - Big Sur we had extra problems to create a good installer.
Now Mattias Gaertner succeeded to create one: But this is a Trunk Version. It works just as good but it is own update, the final release will follow later (next year).

Figure 2: Big Sur

With cliffs plunging hundreds of feet down to rocky coves churning with foamy surf, it's no wonder that many people consider Big Sur the most dramatic stretch of coastline anywhere in the world. But along with its rugged natural beauty, Big Sur is a region with a long artistic history, as well as creative restaurants and unique resorts that let you immerse yourself in this world of fogs, redwood forests, and incomparable coastal views.

So here is the package that you need to work with the latest tested version for the **macOS:**

```
Lazarus-2.0.11-r64280-x86_64-macosx.pkg
```

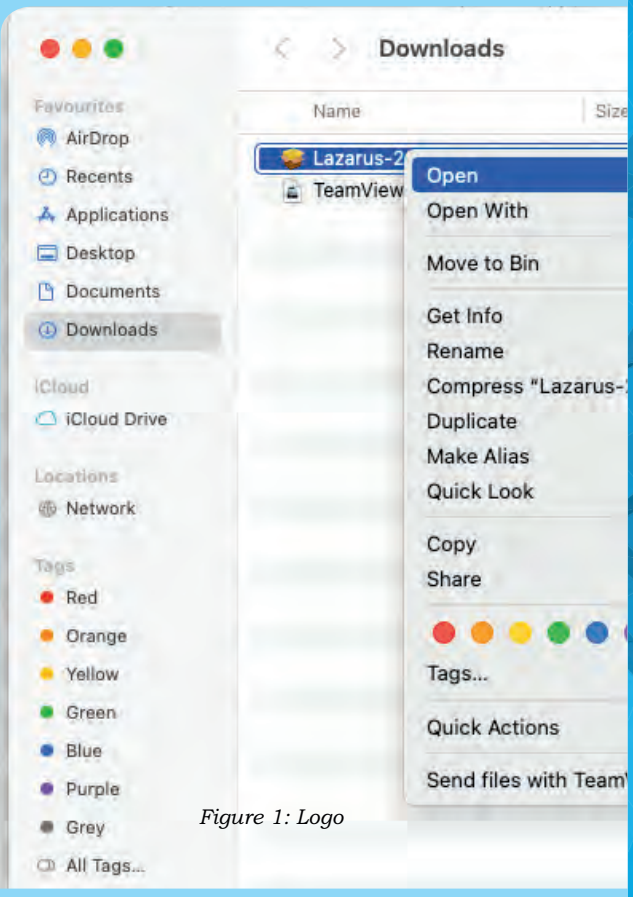
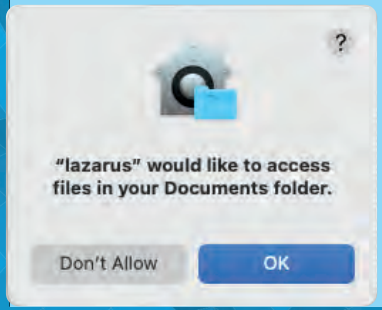


Figure 1: Logo

I tested it 26th december together with Mattias, to make sure that you can install it without problems. If you find any problems just let me know: `office@blaisepascal.eu`.
The promised free version fot the TMS WebCore Components is the next step Mattias is working on. Though it's not for this year. It will be end january before we have updated that.



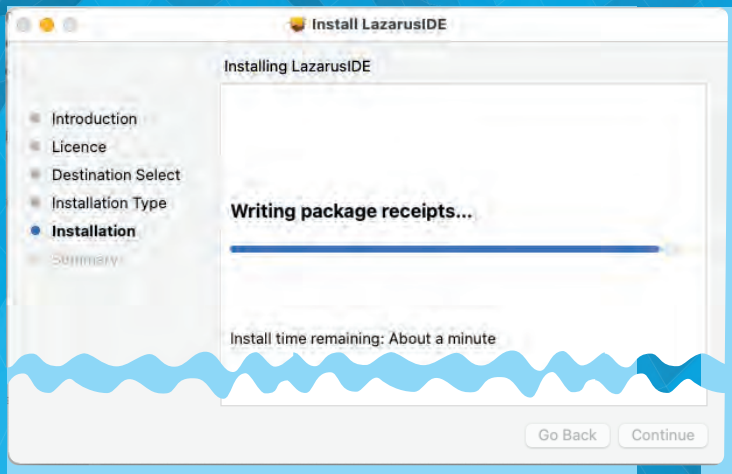


Figure 5: Installing

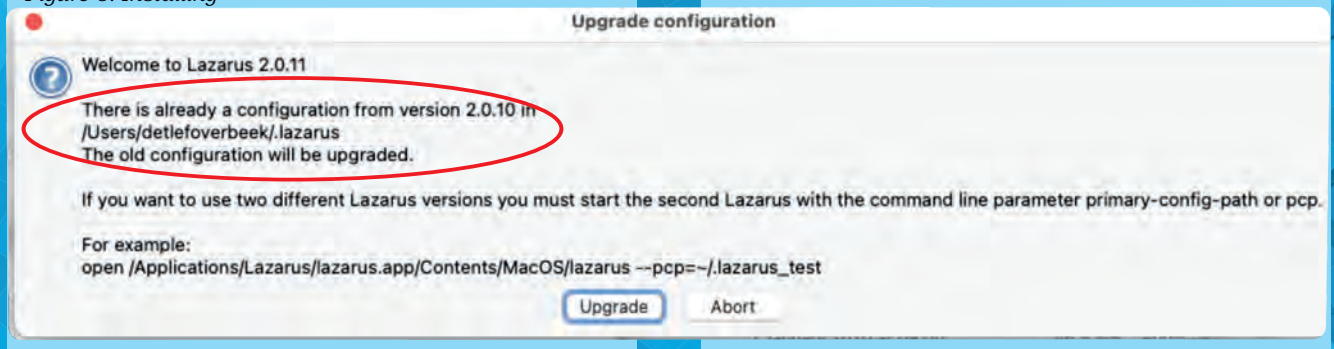


Figure 6: upgrade the version

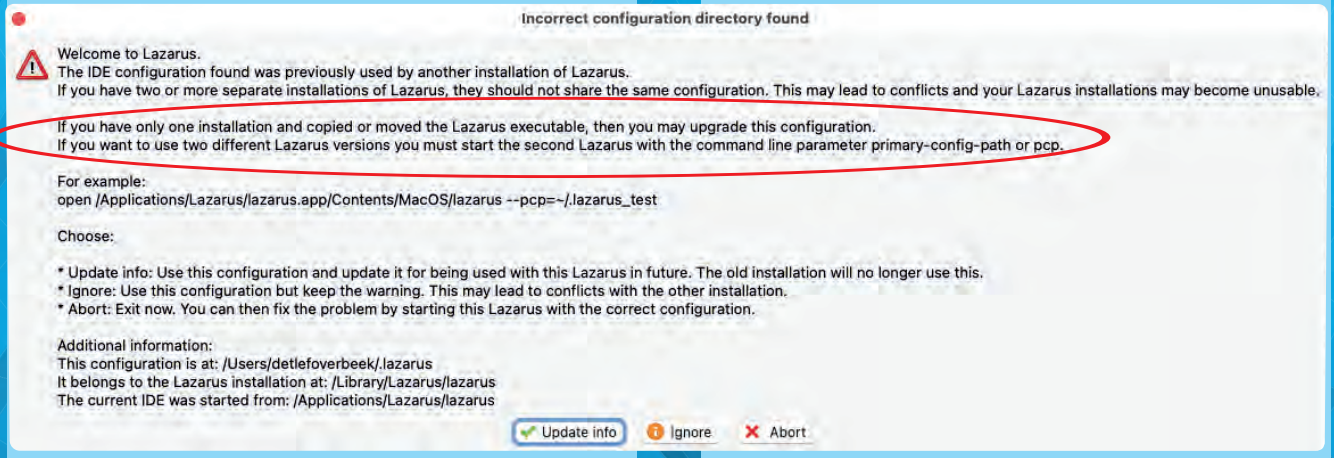
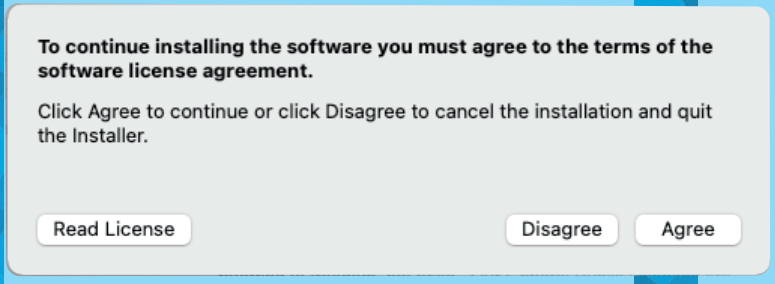
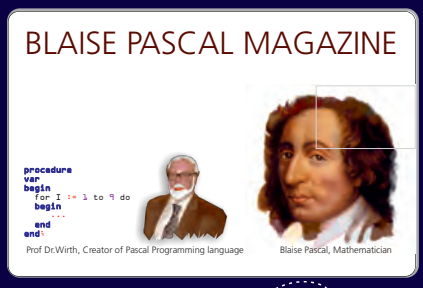


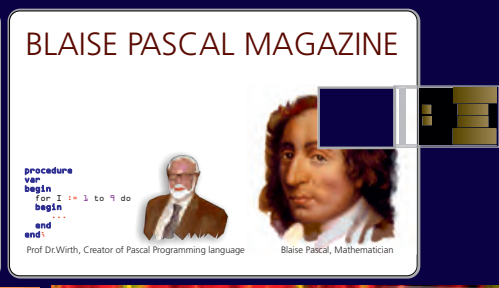
Figure 7: Read carefully



Apple makes it sometimes very difficult to keep up with them. A new OS after an other means usually extra problems.

Figure 8: Answer

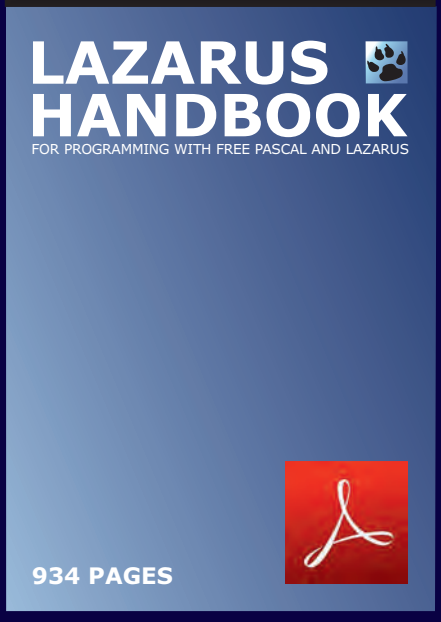


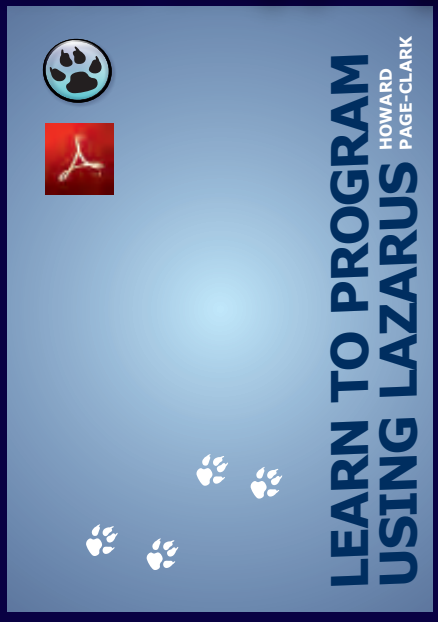


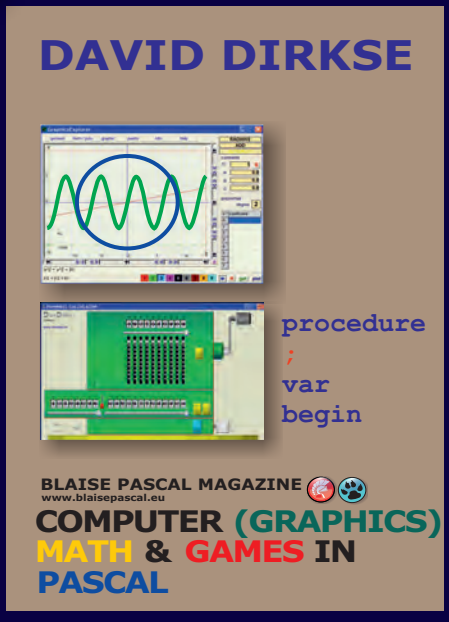
SUPER OFFER

€ 150 ex Vat

free shipping







1. One year **Subscription**
2. **The newest LIB Stick**
   - including Credit Card USB stick
3. **Lazarus Handbook** - Personalized -PDF including Code
4. Book **Learn To Program** using Lazarus PDF including 19 lessons and projects
5. **Book Computer Graphics Math & Games** book + PDF including ±50 projects

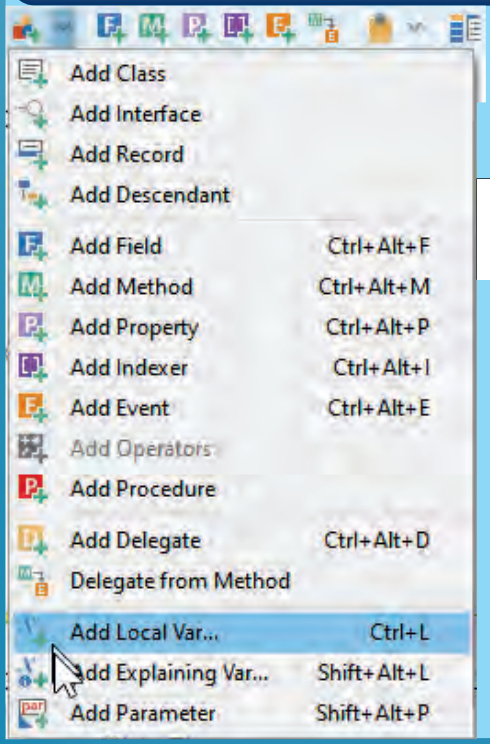
MMXcode for Delphi - Introduction and overview Page 1 /7

By Detlef Overbeek

INTRODUCTION:

MMX is a **Refactoring Browser** with Delphi Pascal support. It is Integrated with Delphi 10 – 10.4 Sydney

- You can improve the design of existing code with build-in refactorings
- It has Instant, two-way navigation.
- Point-and-click, drag-and-drop programming.
- **IntelliReplace™** propagates renames in code.
- You can create and edit classes, members and procedures.
- Instantly copy, paste and on-the-fly convert classes, methods, properties etc. even between classes and modules.
- Use Entity Insight™ to change attributes for (multiple) selected members.
- Rearrange and Sort code, maintaining region definitions.
- Improve Navigation with History, Method Favorites and Member Search Bar
- Live Documentation support
- Live Metrics™
- Source Indexer
- IDE Editor Refactorings bring refactorings right in the Editor.



History

For over two decades ModelMaker Code Explorer™ is an add-on to the Delphi IDE for developers. Speaking for myself, a Delphi installation is simply incomplete without MMX. Uwe Rabe has taken over the distribution and further development of MMX since ModelMaker Tools BV is concentrated on Mind Mapping tool SimpleMind.
There is a slight change in the name: **ModelMaker Code Explorer™** will become **MMX Code Explorer** from now on, still shortened as **MMX.**
The new website can be found here:
`https://www.mmx-delphi.de/`

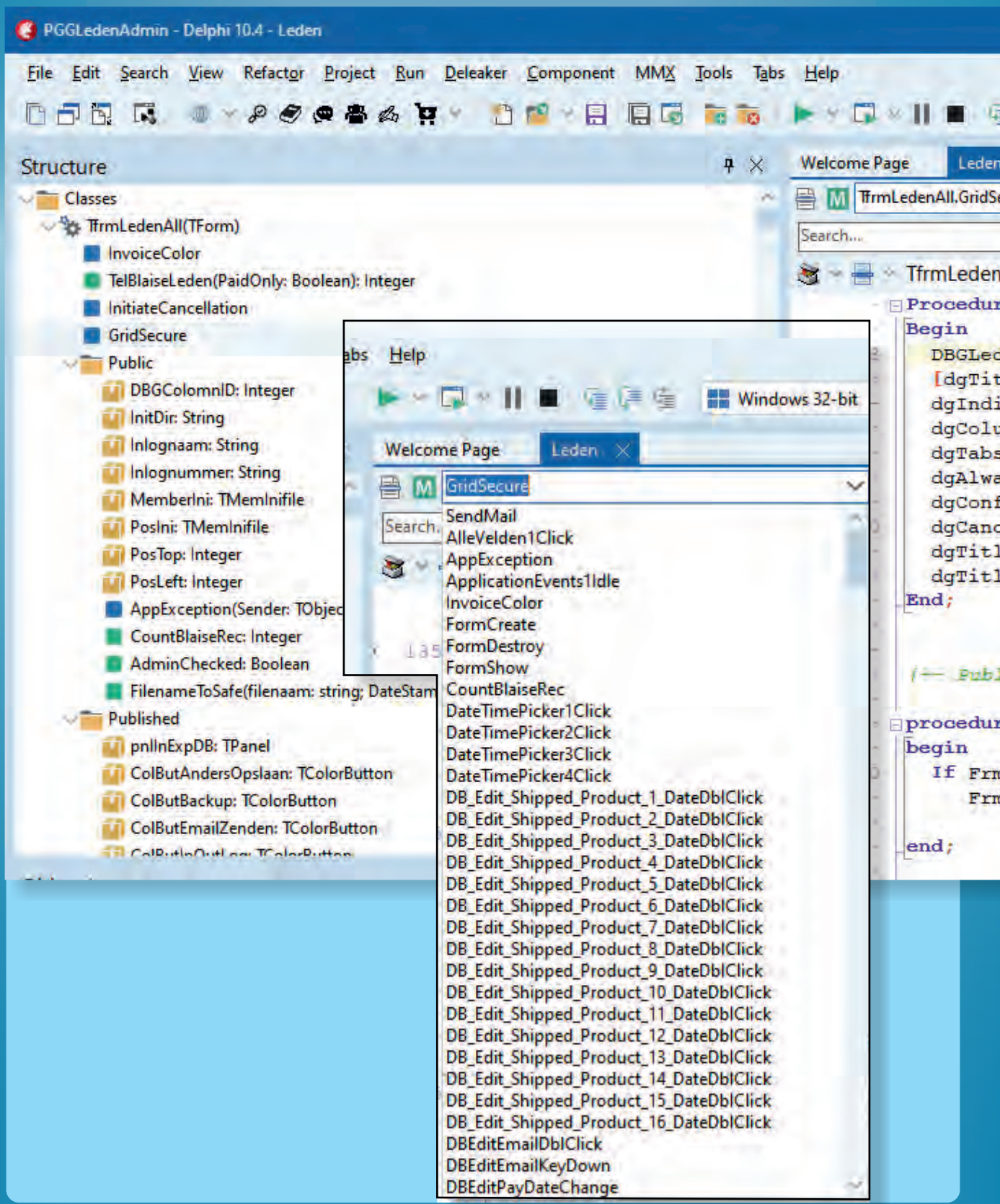
A very good development is that MMX is available for free now.
This allows also hobbyist developers to make use of this tool.
MMX is still closed source, though.

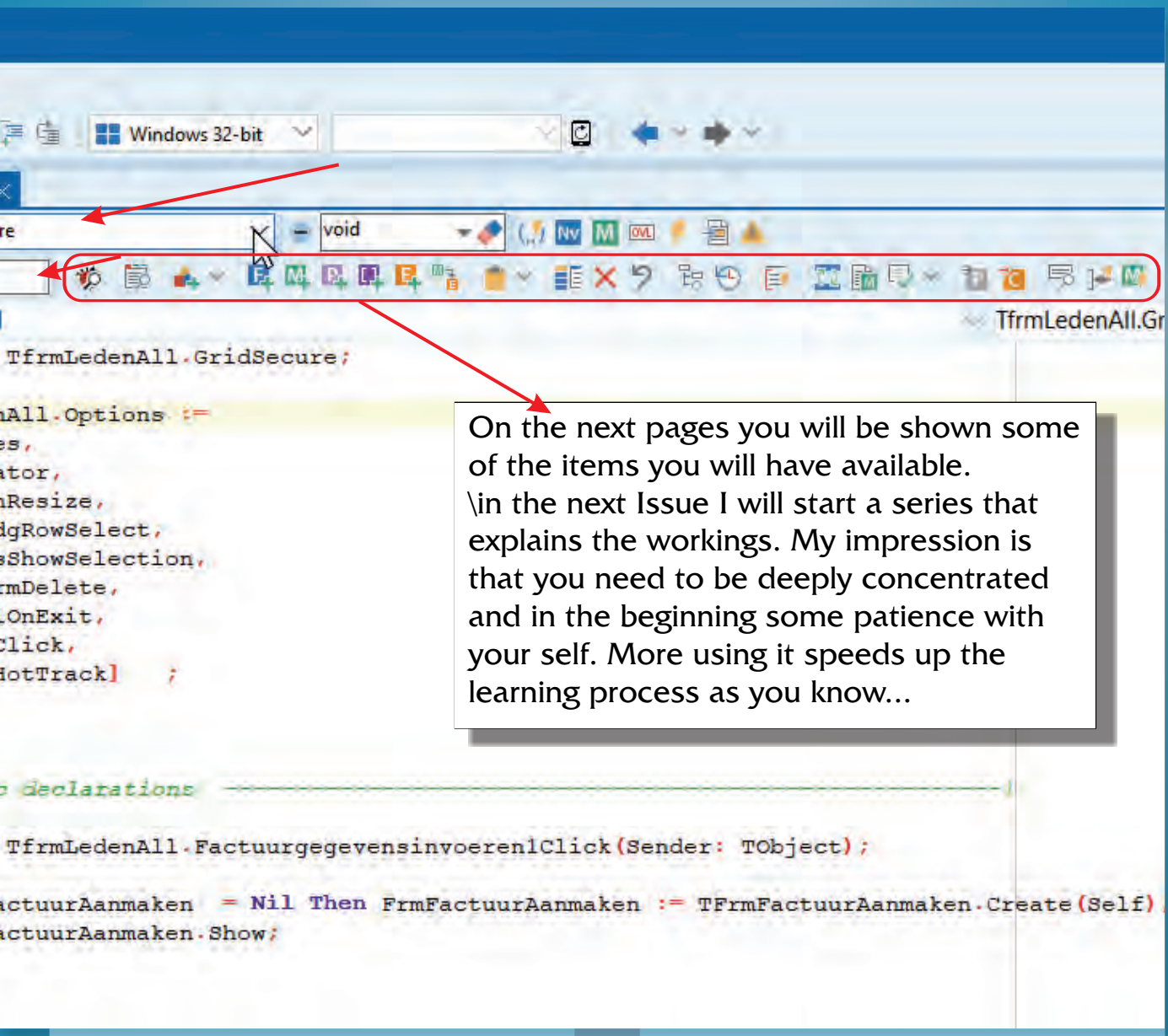
Alas, the Delphi 5 and 6 versions didn't make it into this transition, as they would have required too much special treatment.
The current release thus supports
Delphi 7,
Delphi 2007 – 2010,
Delphi XE – XE8,
Delphi 10 Seattle,
Delphi 10.1 Berlin and
Delphi 10.2 Tokyo,
Delphi 10.3 Rio and
Delphi 10.4 Sydney.

MMX is available for free now.

The latest MMX supporting retired versions are still available for download.
As of 18 May 2018 the Delphi and C# related ModelMaker Tools products are no longer available. This concerns:

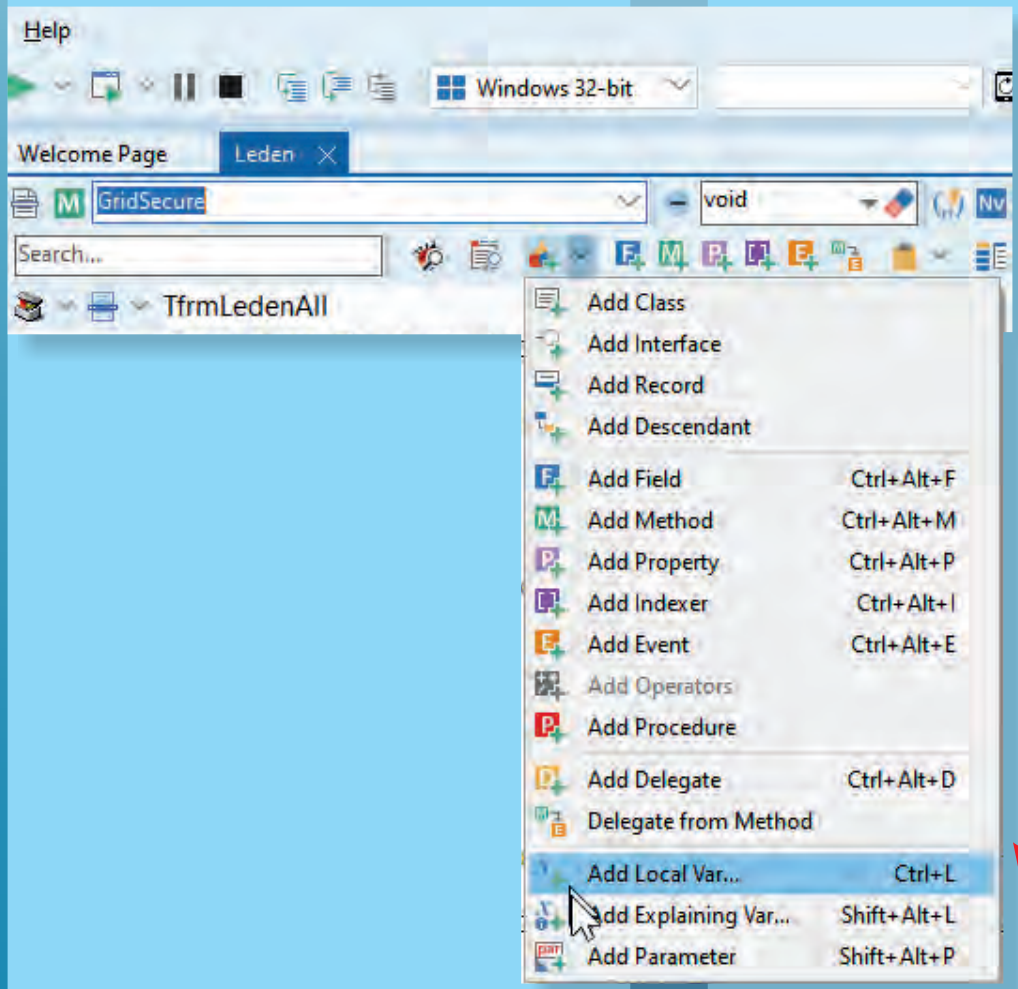
ModelMaker - stand alone UML Modeler for Delphi and C# ModelMaker Code Explorer - Delphi IDE add-in Structured Difference Viewer

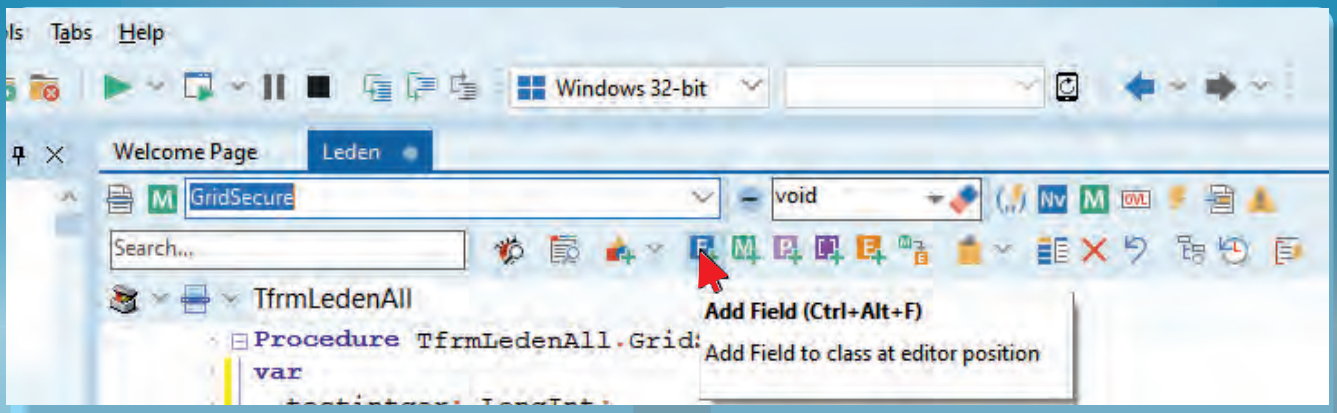

PGGLedenAdmin - Delphi 10.4 - Leden
File Edit Search View Refactor Project Run Deleaker Component MMX Tools Tabs Help
Structure
Classes
TfrmLedenAll(TForm)
InvoiceColor
TelBlaiseLeden(PaidOnly: Boolean): Integer
InitiateCancellation
GridSecure
Public
DBGColomnID: Integer
InitDir: String
Inlognaam: String
Inlognummer: String
MemberIni: TMemInifile
PosIni: TMemInifile
PosTop: Integer
PosLeft: Integer
AppException(Sender: TObjec
CountBlaiseRec: Integer
AdminChecked: Boolean
FilenameToSafe(filenaam: string; DateStam
Published
pnlInExpDB: TPanel
ColButAndersOpslaan: TColorButton
ColButBackup: TColorButton
ColButEmailZenden: TColorButton
Welcome Page
Leden
TfrmLedenAll.GridSe
Search...
TfrmLeden
Procedur
Begin
DBGLed
[dgTit
dgIndi
dgColu
dgTabs
dgAlwa
dgConf
dgCanc
dgTitl
dgTitl
End;
procedur
begin
If Frm
Frm
end;
Help
Windows 32-bit
Welcome Page
Leden
GridSecure
Search...
SendMail
AlleVelden1Click
AppException
ApplicationEvents1Idle
InvoiceColor
FormCreate
FormDestroy
FormShow
CountBlaiseRec
DateTimePicker1Click
DateTimePicker2Click
DateTimePicker3Click
DateTimePicker4Click
DB_Edit_Shipped_Product_1_DateDblClick
DB_Edit_Shipped_Product_2_DateDblClick
DB_Edit_Shipped_Product_3_DateDblClick
DB_Edit_Shipped_Product_4_DateDblClick
DB_Edit_Shipped_Product_5_DateDblClick
DB_Edit_Shipped_Product_6_DateDblClick
DB_Edit_Shipped_Product_7_DateDblClick
DB_Edit_Shipped_Product_8_DateDblClick
DB_Edit_Shipped_Product_9_DateDblClick
DB_Edit_Shipped_Product_10_DateDblClick
DB_Edit_Shipped_Product_11_DateDblClick
DB_Edit_Shipped_Product_12_DateDblClick
DB_Edit_Shipped_Product_13_DateDblClick
DB_Edit_Shipped_Product_14_DateDblClick
DB_Edit_Shipped_Product_15_DateDblClick
DB_Edit_Shipped_Product_16_DateDblClick
DBEditEmailDblClick
DBEditEmailKeyDown
DBEditPayDateChange

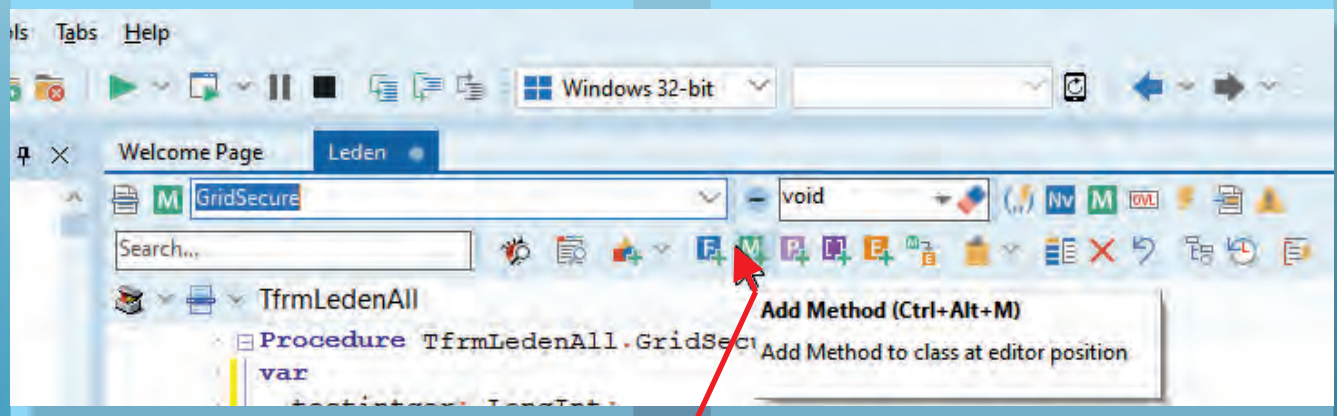


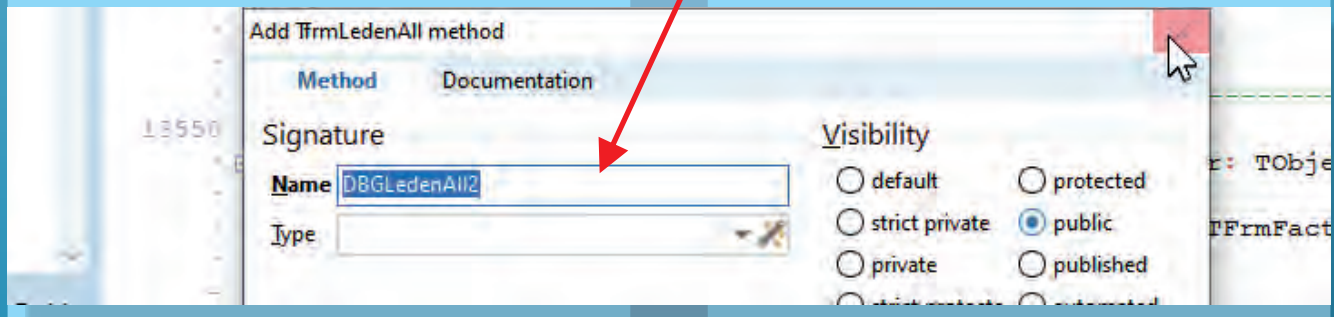
On the next pages you will be shown some of the items you will have available.
\in the next Issue I will start a series that explains the workings. My impression is that you need to be deeply concentrated and in the beginning some patience with your self. More using it speeds up the learning process as you know...

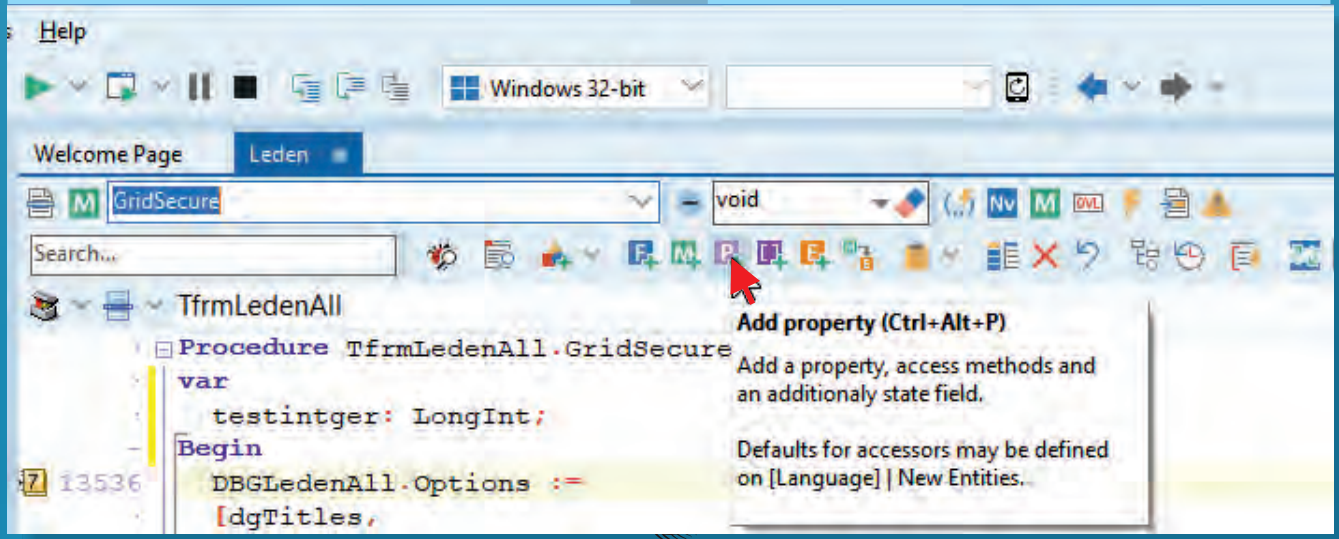
You can make a choice of the kind of commands you want to use

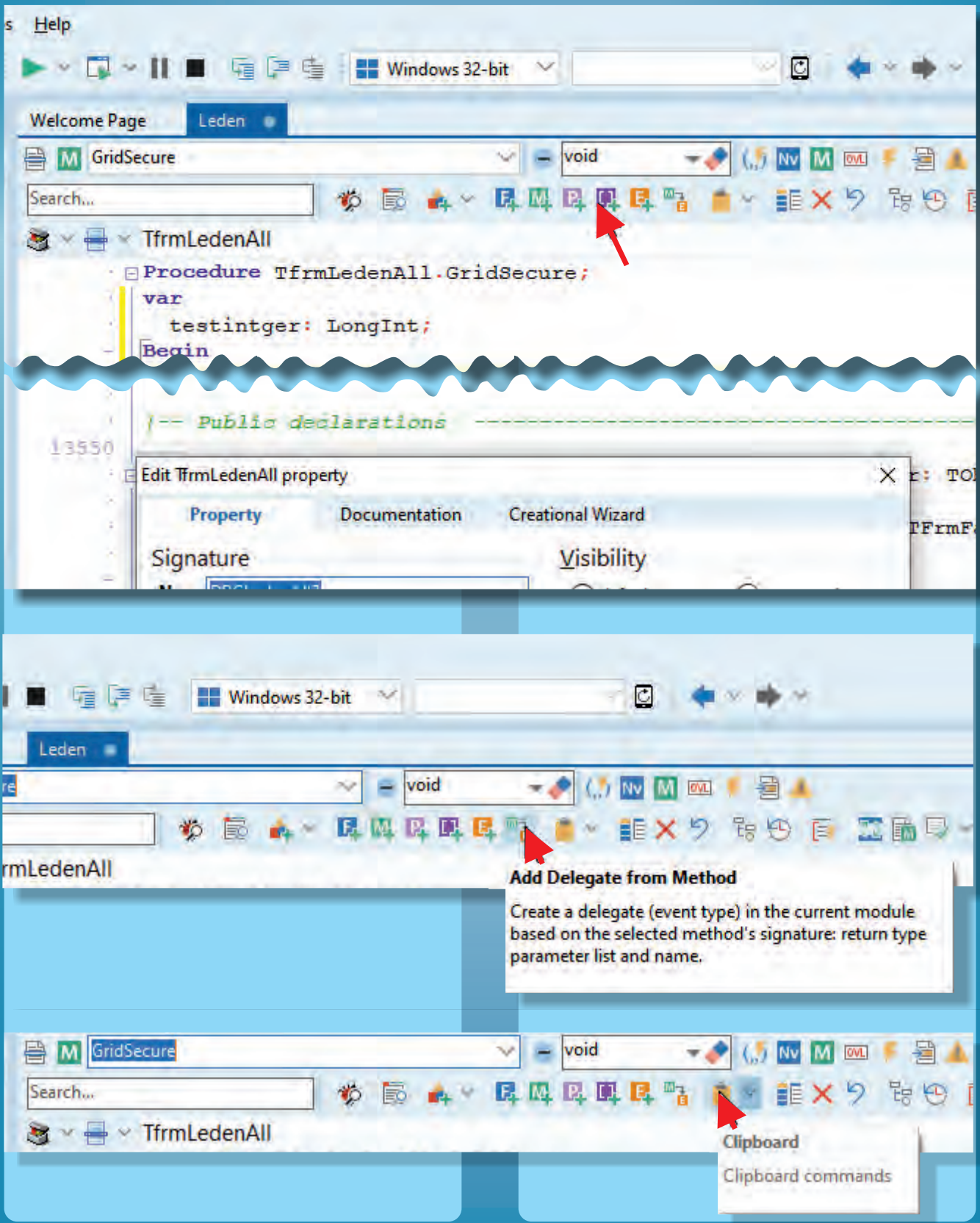


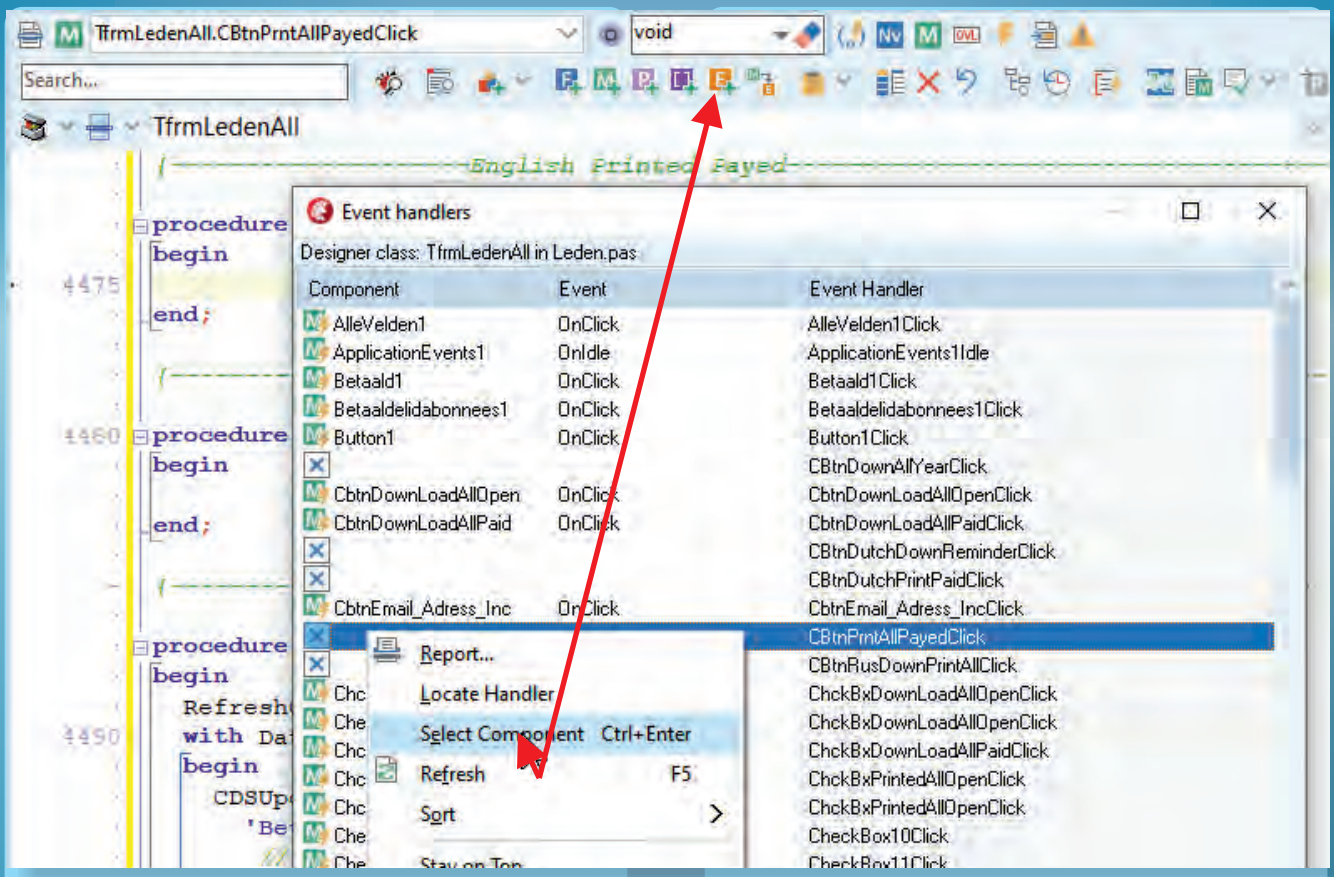

Tabs Help
Windows 32-bit
Welcome Page
Leden
GridSecure
void
Search...
TfrmLedenAll
Procedure TfrmLedenAll.Grid
var
Add Field (Ctrl+Alt+F)
Add Field to class at editor position


Tabs Help
Windows 32-bit
Welcome Page
Leden
GridSecure
void
Search...
TfrmLedenAll
Procedure TfrmLedenAll.GridSec
var
Add Method (Ctrl+Alt+M)
Add Method to class at editor position


Add TfrmLedenAll method
Method
Documentation
Signature
Name
DBGLedenAll2
Type
Visibility
default
protected
strict private
public
private
published


Help
Windows 32-bit
Welcome Page
Leden
GridSecure
void
Search...
TfrmLedenAll
Procedure TfrmLedenAll.GridSecure
var
testintger: LongInt;
Begin
DBGLedenAll.Options :=
[dgTitles,
Add property (Ctrl+Alt+P)
Add a property, access methods and an additionaly state field.
Defaults for accessors may be defined on [Language] | New Entities.


Help
Windows 32-bit
Welcome Page
Leden
GridSecure
void
Search...
TfrmLedenAll
Procedure TfrmLedenAll.GridSecure;
var
testintger: LongInt;
Begin
{-- Public declarations
13550
Edit TfrmLedenAll property
Property
Documentation
Creational Wizard
Signature
Visibility
Add Delegate from Method
Create a delegate (event type) in the current module based on the selected method's signature: return type parameter list and name.
Clipboard
Clipboard commands



The possibilities MMX offers are not only beautiful but also very supportive and even challenging.
It gives you new insights and will deepen your understanding of the Pascal language and forces you to rethink the design of your app. Since there is no charge at all you should try.

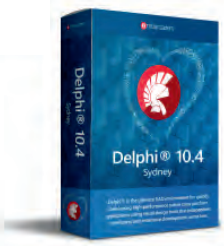
It costs extra time to learn the immense possibilitie's - but the use of it is so much rewarding that you will use a lot of it in no time at all and choose to make your own alterations because even that is possible. It is complex but not difficult to use…

barnsten

RAD Studio 10.4 Sydney

RAD Studio 10.4 Sydney is the ultimate RAD environment loved by developers for quickly building high-performance native cross-platform applications in Modern C++ and Delphi using powerful visual design tools and integrated toolchains.

Shop RAD Studio



Delphi 10.4 Sydney

A powerful RAD environment for quickly developing high-performance native cross-platform applications using powerful visual design tools and integrated toolchains that independent developers and enterprise development teams love.

Shop Delphi

C++Builder 10.4 Sydney

C++Builder 10.4 Sydney is a complete RAD environment, loved by developers, with an integrated toolchain for modern C++ to help quickly build high-performance native Windows apps 10x faster than competing solutions.

Shop C++ Builder

The Barnsten team wishes you a good and healthy 2021

www.barnsten.com / info@barnsten.com
France: Téléphone +33 (0)9 72 19 28 87
Benelux: Telefoon +31 (0)2 35 42 22 27

Figure 1: About box

Introduction
Since years programmes exist that want to help you by organizing, creating structures, planing thoughts etc. I started with Freemind years ago, saw some others but either they were costly or too complicated to use.
Until I found out that my greatly admired Gerrit Beuze had created one:
Simple Mind
This is truly an absolute winner:
The simplicity of the User Interface and the absolute logical way of handling shows he has designed it by the habits of a non-developers mentality. The customer is the only important person. Fantastic!

Above this all he sells the program very low in price, so that I just for fun bought also the Android version (8,45 €).
The Windows version costs 25 € and there are also macOs and iOS versions for about the same price, there is also a trial version.

In this article I will give an overview of the program. As said the handling is very intuitive.

What is meant by a MindMap tool?
A mind map is a diagram used to visually organize information.
A mind map is hierarchical and shows relationships among pieces of the whole.
It is often created around a single concept, drawn as an image in the center of a blank page, to which associated representations of ideas such as images, words and parts of words are added.
Major ideas are connected directly to the central concept, and other ideas branch out from those major ideas.
Mind maps can also be drawn by hand, either as "notes" during a lecture, meeting or planning session, for example, or as higher quality pictures when more time is available.
Mind maps are considered to be a type of spider diagram. A similar concept in the 1970s was "idea sun bursting".

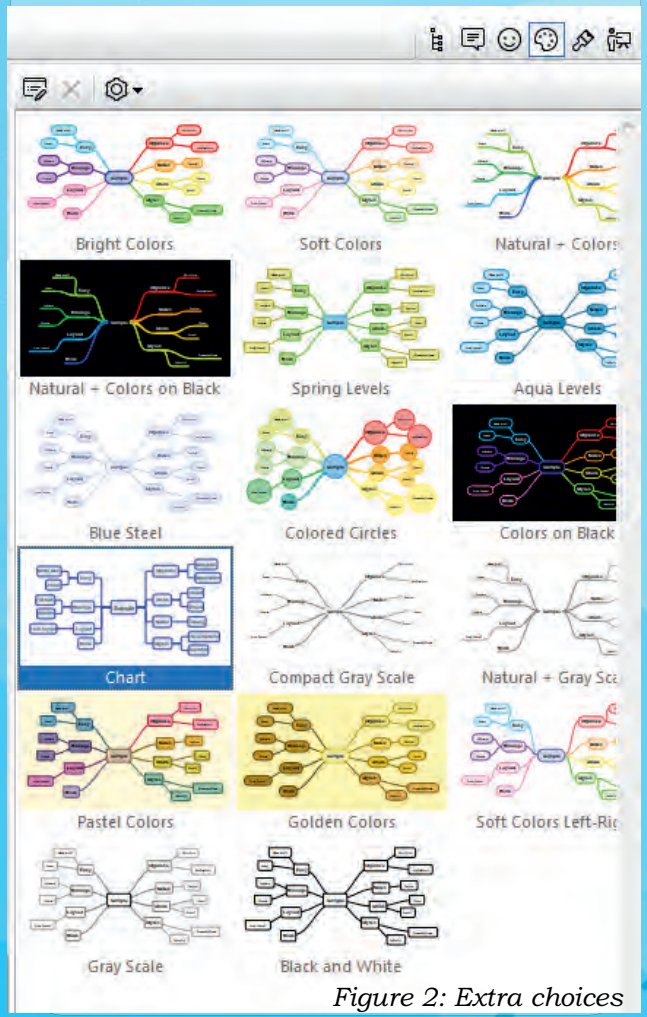


Figure 2: Extra choices

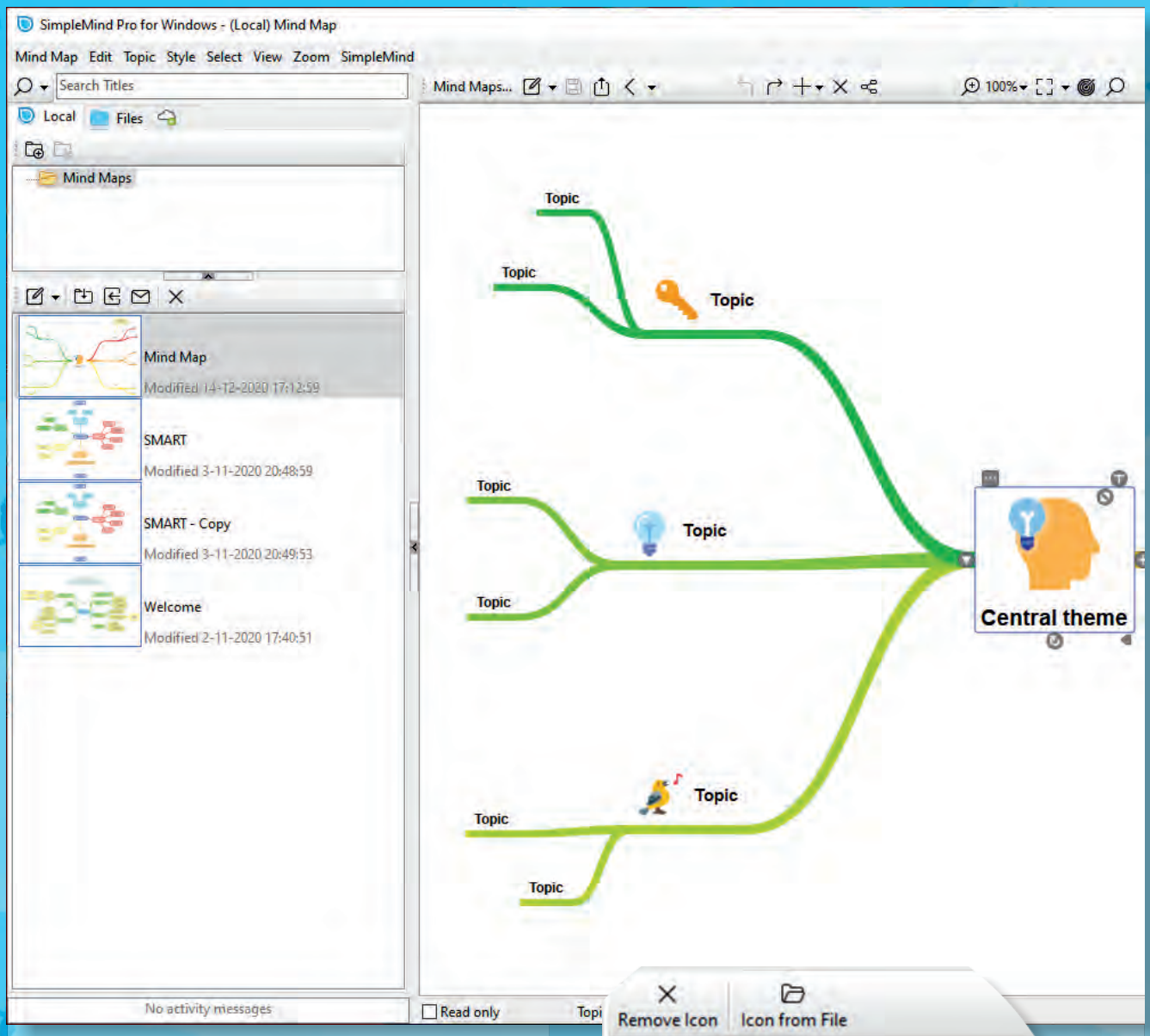


Figure 3: an example of a mind map

At the left you can choose one of the basic models but you can choose from your own altered models. Select a branch and alter anything you want insert more items and pictures or images.

There are more to find: In the right column, click the painting palette and a lot of variations are available. see *Figure 2: Extra choices.*

Figure 4: You can load your own icons

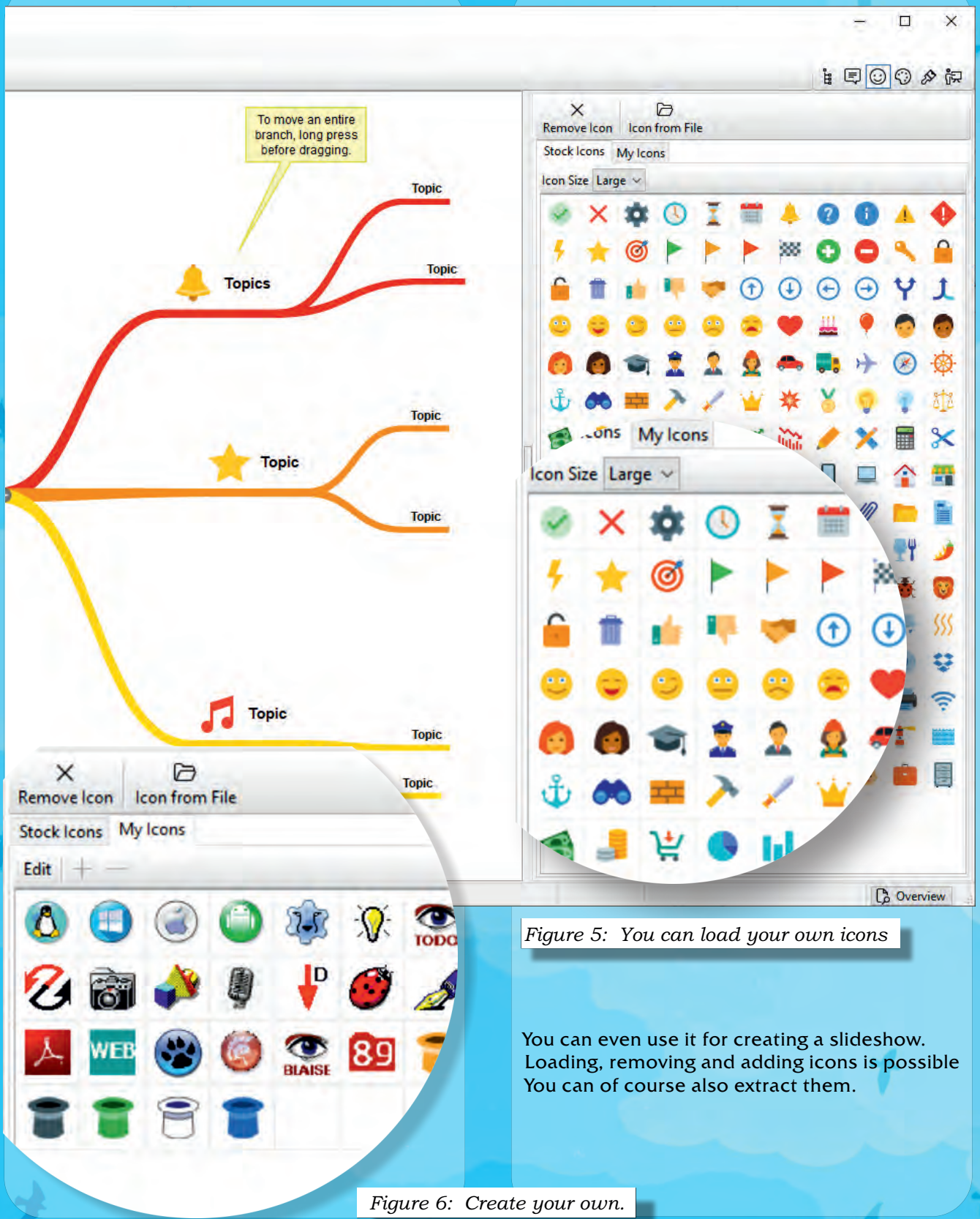
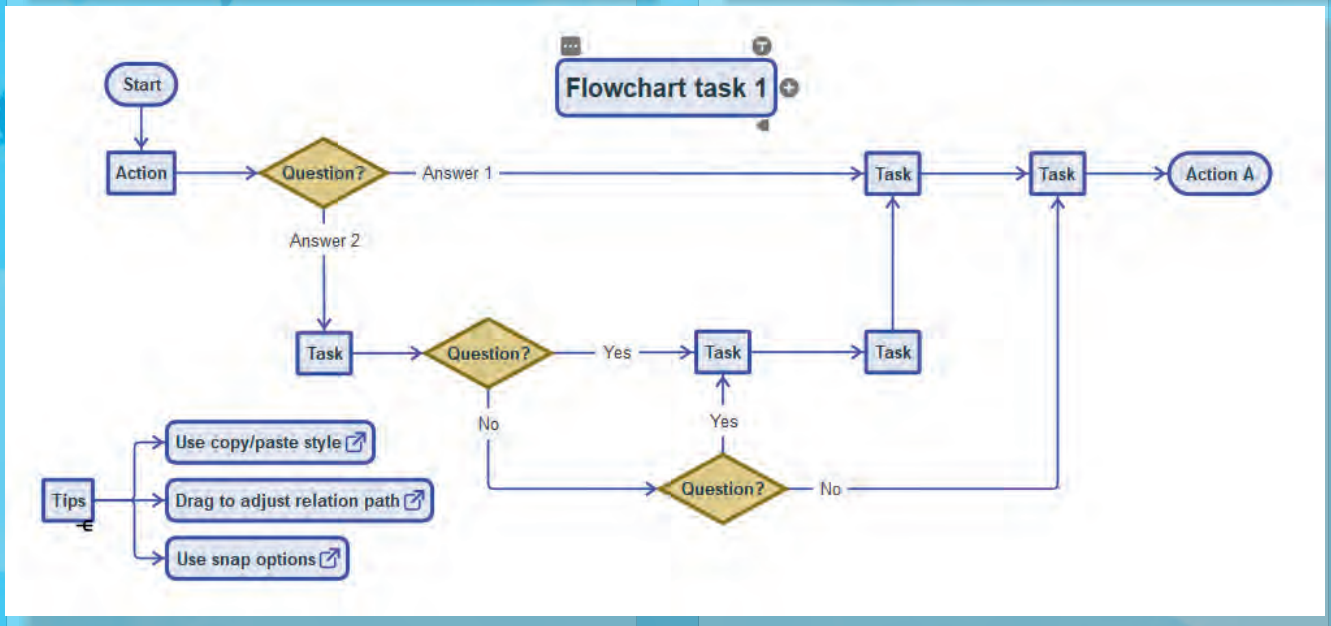


Figure 5: You can load your own icons

You can even use it for creating a slideshow.
Loading, removing and adding icons is possible
You can of course also extract them.

Figure 6: Create your own.



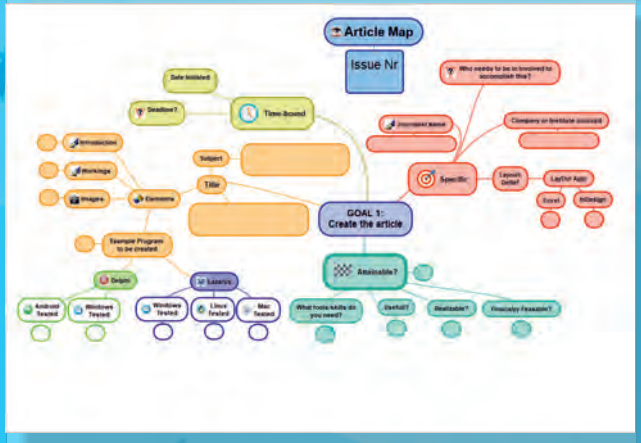


Figure 7: Extra overview
Flowchart items can also be created.

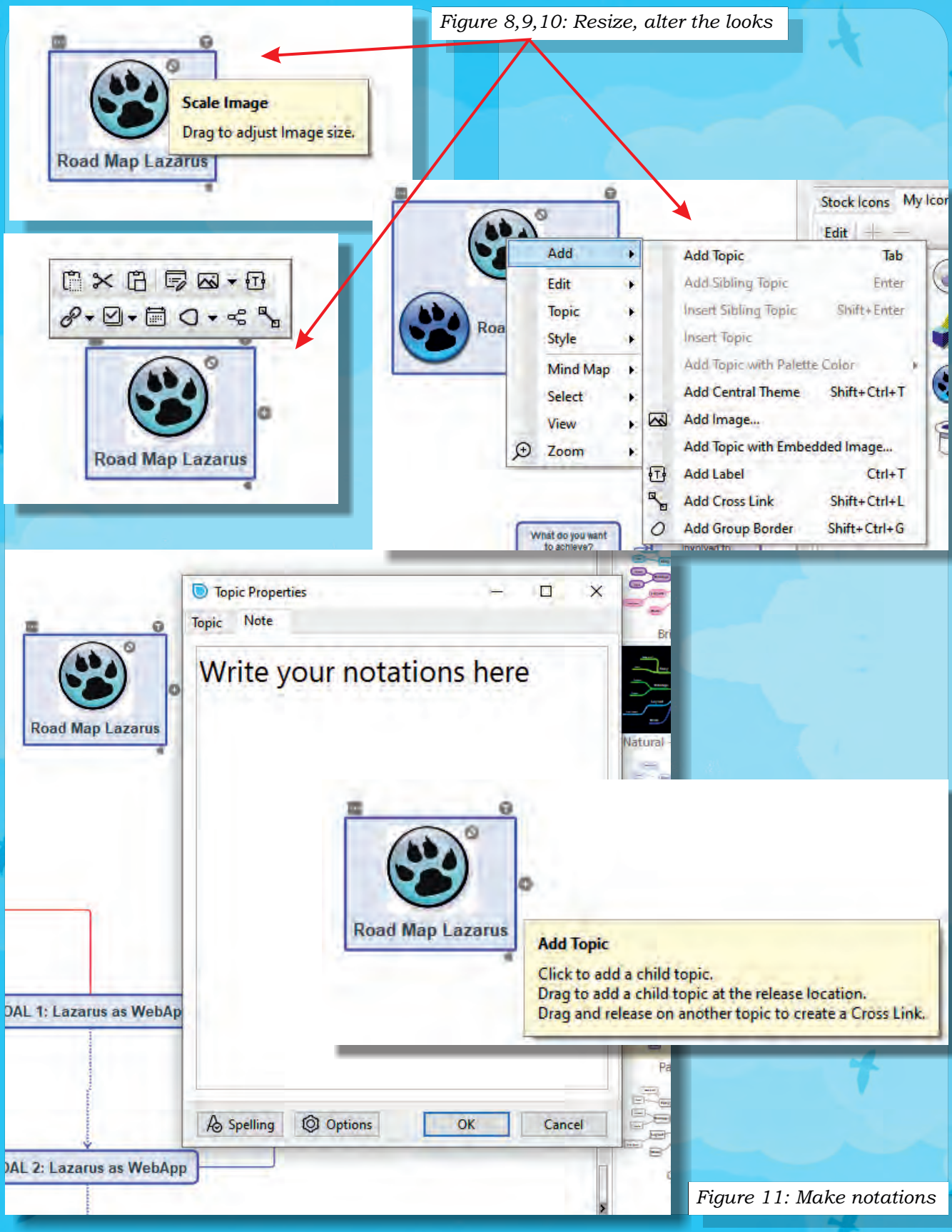
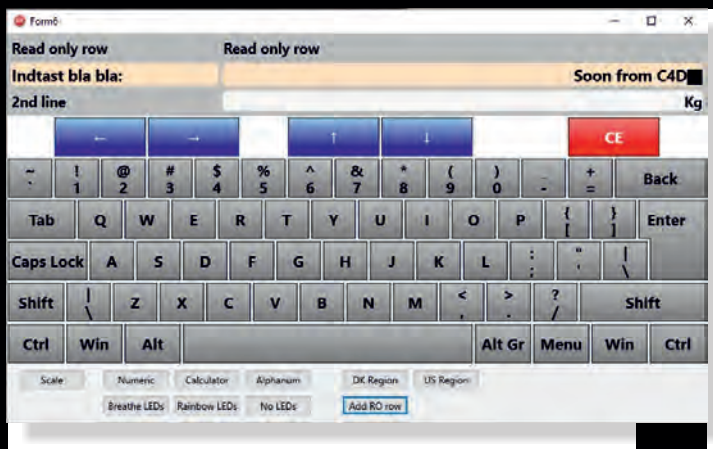


Figure 8,9,10: Resize, alter the looks

Figure 11: Make notations



We are happy to announce not only an update to kbmFMX Std Edition (which is currently bundled with kbmMemTable), but also kbmFMX Professional Edition which adds our new superb virtual keyboard and breadcrumb controls.

The virtual keyboard exists with various customizable layouts (alphanumeric, numeric, calculator), Danish, English and German region (also easily customizable), masking and animation features, and it can show as an embedded keyboard, an embedded keyboard with multiline input/editor, and as a complete replacement virtual popup keyboard optionally replacing the standard Android, IOS and Windows virtual keyboards, directly interacting with any Firemonkey keyboard aware control.

Read more on our site at http://www.components4developers.com/

kbmFMX Professional comes with full source at can be purchased for only US$50 per named developer license and includes 12 months SAU (service and update subscription)

kbmFMX Professional Edition includes all features from kbmFMX Standard Edition, including TDataSource databound readonly grid, TDataSource databound image and memo controls, signature capture control, numeric edit control, advanced pan/zoom/rotate image control with optional areas of interest zones, busy screen overlay and more.

TEASER: KBMFMX MOBILE VIRTUAL KEYBOARD

Next release of kbmFMX will contain even more features for defining keyboard layouts.

Mobile small screen keyboards usually follow a different operating concept than workstation virtual keyboards running on self service like terminals or even 9 inch tablets.

Basically small screen keyboards operate with multiple “layers” of keyboards each holding a number of keys with exactly one base character on each + capital character for alpha characters. That means that the keyboard usually do not have AltGr or other “shift” type keys except for a capital shift key.

The small screen keyboard also have no navigation keys like arrow or tabulator keys, because it is assumed the user will move the caret (if needed) using touch within an editable field.

Instead the small screen keyboard comes with one or more layer selection keys, like 123 or ABC for shifting forth and back between defined keyboard layers.

kbmFMX’s virtual keyboard now supports 3 layers per keyboard layout. Non small screen keyboards typically only use the default layer 1. Further it supports custom virtual keycodes for selecting a layer.

A new keyboard layout (`TkbmFMXMobileAlphaNumericKeyboardLayout`) has been added, which fairly closely emulates a virtual keyboard on a small screen mobile phone and which currently uses two layers.

Are Bumblebees intelligent?!

"Our study puts the final nail in the coffin of the idea that small brains constrain insects to have limited behavioural flexibility and only simple learning abilities," Lars Chittka, the project supervisor, co-author, and a professor at QMUL's School of Biological and Chemical Sciences, said in a press release.
Scientists have shown that bees can grasp the abstract concept of using an object to achieve a goal – in this case, a gulp of sugar water – even if they aren't likely to engage in such behavior in the wild.

It turns out you can teach an old bee new tricks. The fuzzy insects can even improve on the method, scientists say.

Bees join the ranks of tool-capable organisms, according to a paper published Thursday in the journal Science. A team of researchers from Queen Mary University of London (QMUL) showed bees to be cleverer than anyone expected, teaching them to push a ball into a goal for a sweet reward. Surprisingly, the insects' behavior implied an abstract understanding of the simple soccer game, one that that went beyond mere mimicry.
Once considered the exclusive domain of humans, then primates, and then birds and cetaceans, tool-use has historically been thought a hallmark of highly intelligent animals. Now, bees are buzzing into the club.

Bees in a previous experiment succeeded in pulling a string to get their ambrosia, but since they often carry out similar tasks in the wild, such behavior could have been an extension of natural instinct. This time, researchers wanted to test whether bees could solve a problem involving a nearby object with no obvious connection to the reward.

They set up a test where a bee could land on a small table with a ball and a central hole. If the bee rolled the ball into the hole, scientists would reward it with a snack of sweet sucrose solution.

But while a few super-smart bees were able to solve the problem on their own, most needed some help. Researchers trained initial bees by using a bumblebee-decorated stick to show how to push the ball to the goal. Later bees could watch previously-trained live bee demonstrators for a chance to learn "socially." Another group watched a "ghost" demonstration where a magnet hidden under the table dragged the ball into the hole, and a control group was presented with a ball already sitting on top of the hole.

It turns out bumblebees catch on quickly when there's sugar involved. Rookies had three chances to watch a veteran do the trick, after which all were able to imitate the teacher."It may be that bumblebees, along with many other animals, have the cognitive capabilities to solve such complex tasks, but will only do so if environmental pressures are applied to necessitate such behaviours," explained joint-lead author Olli J. Loukola in the release.

Even more impressive was how the trainees showed off their new skills. When presented with a new situation where three balls sat on the table at varying distances from the hole, seven out of 10 socially trained bees and six out of 10 "ghost"-taught bees went for the closest one, even if they had watched an experienced bee move the farthest ball during training.

From this innovation, the researchers could conclude two things: that the bees had an abstract notion of what they were doing, rather than just mindlessly following the leader; and that rolling a relatively big ball all the way across a table is tough for little bees.

"The bees solved the task in a different way than what was demonstrated, suggesting that observer bees did not simply copy what they saw, but improved on it," Dr. Loukola said.

Repeating the experiment with a black ball rather than a yellow one produced almost identical performance in trained bees, further underscoring the idea that the insects knew exactly what they were doing, in a display of what the researchers wrote was "unprecedented cognitive flexibility."

It seems we've been underestimating the mental capabilities of the very small.

"The old-fashioned view is if an animal has a small brain, it's not intelligent or smart," Loukola told New Scientist.

Even though a bee brain has about 100,000 times fewer neurons than the one between a human's ears, it seems that in at least one simple situation they have the ability to learn and improvise a new strategy just like a person.

As the team wrote in the paper, "We present here an example in which an insect displays a goal-directed behavior for which evolution has not provided them with a rigid adaptation."

Eirik Søvik, who studies bees and animal problem-solving at Volda University College in Norway, suggests that we're only just beginning to scratch the surface of insect cognition.

"We just have not been very good at designing experiments that allow us to probe insect cognition very well," he told New Scientist. "That's probably because it is so incredibly difficult to imagine how bees experience the world, and if you want to give them tasks they can succeed at, that is key. I think the authors here really succeed at taking the bees' view of the world."

BUMBLEBEE

KBMMW PROFESSIONAL AND ENTERPRISE EDITION V. 5.13 RELEASED!

- **RAD Studio XE5 to 10.4.1 Sydney supported**
- Win32, Win64, Linux64, Android, IOS 32, IOS 64 and OSX client and server support
- Native high performance 100% developer defined application server
- Full support for centralized and distributed load balancing and failover
- Advanced ORM/OPF support including support of existing databases
- Advanced logging support
- Advanced configuration framework
- Advanced scheduling support for easy access to multithread programming
- Advanced smart service and clients for very easy publication of functionality
- High quality random functions.
- High quality pronouncable password generators.
- High performance LZ4 and Jpeg compression
- Complete object notation framework including full support for YAML, BSON, Messagepack, JSON and XML
- Advanced object and value marshalling to and from YAML, BSON, Messagepack, JSON and XML
- High performance native TCP transport support
- High performance HTTPSys transport for Windows.
- CORS support in REST/HTML services.
- Native PHP, Java, OCX, ANSI C, C#, Apache Flex client support!

kbmMemTable is the fastest and most feature rich in memory table for Embarcadero products.

- **Easily supports large datasets with millions of records**
- **Easy data streaming support**
- **Optional to use native SQL engine**
- **Supports nested transactions and undo**
- **Native and fast build in M/D, aggregation/grouping, range selection features**
- **Advanced indexing features for extreme performance**

- **Improvement to ORM including support for rewriting JOINS.**
- **Improvements to SmartBind**
- **Improvements to SmartEvent**
- **Improvements to LINQ**
- **Support of XML mixed text documents**
- **New TkbmMWXMLReformatter which can analyze and reformat even damaged XML**
- **New support for USB/HID devices including MagTek barcode reader class (Windows/Ent)**
- **More features improvements and fixes.**

Please visit
http://www.components4developers.com
for more information about kbmMW

- High speed, unified database access (35+ supported database APIs) with connection pooling, metadata and data caching on all tiers
- Multi head access to the application server, via REST/AJAX, native binary, Publish/Subscribe, SOAP, XML, RTMP from web browsers, embedded devices, linked application servers, PCs, mobile devices, Java systems and many more clients
- Complete support for hosting FastCGI based applications (PHP/Ruby/Perl/Python typically)
- Native complete AMQP 0.91 support (Advanced Message Queuing Protocol)
- Complete end 2 end secure brandable Remote Desktop with near realtime HD video, 8 monitor support, texture detection, compression and clipboard sharing.
- Bundling kbmMemTable Professional which is the fastest and most feature rich in memory table for Embarcadero products.

COMPONENTS 4 DEVELOPERS

DX

EESB, SOA,MoM, EAI TOOLS FOR INTELLIGENT SOLUTIONS. kbmMW IS THE PREMIERE N-TIER PRODUCT FOR DELPHI / C++BUILDER