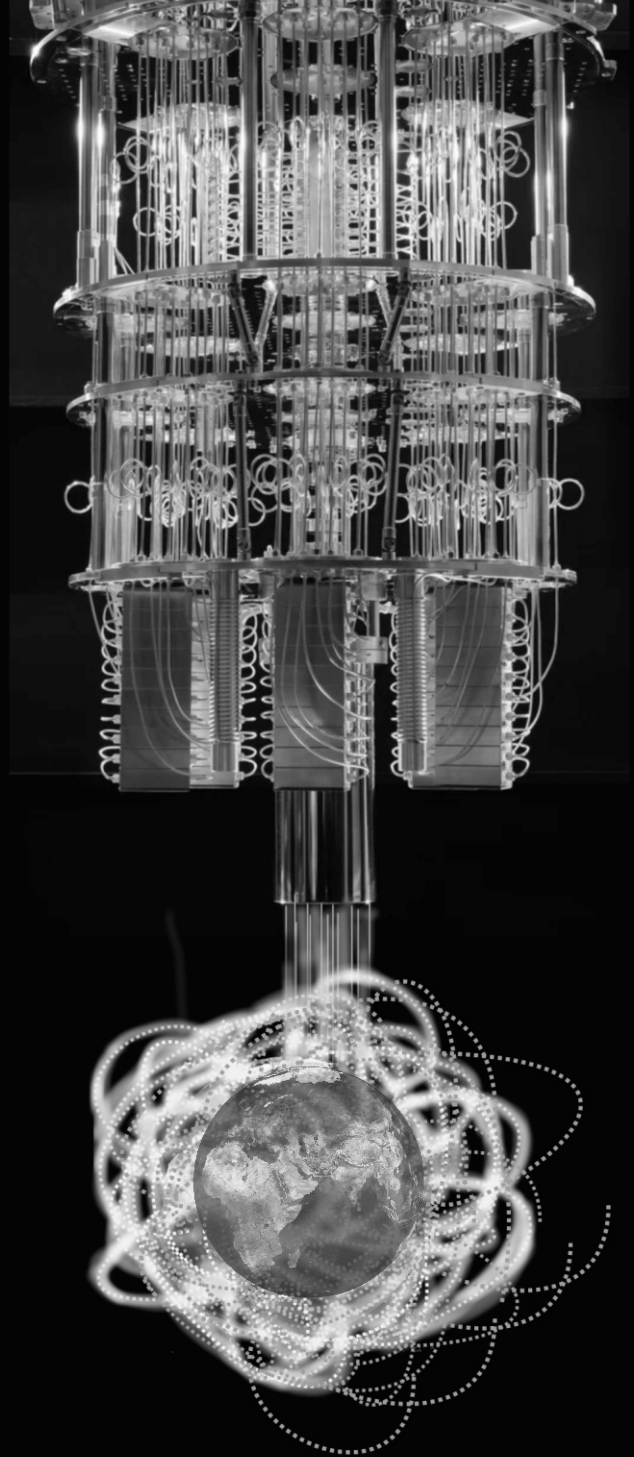




BLAISE PASCAL MAGAZINE 96



From our technical advisor Cartoon from Jerry King
Python for Delphi with Python4Delphi by Max kleiner
Mustache Templates in Pascal,

One popular template language definition is called Mustache, by Michael van Canneyt
Quantum Internet already active, by Detlef Overbeek

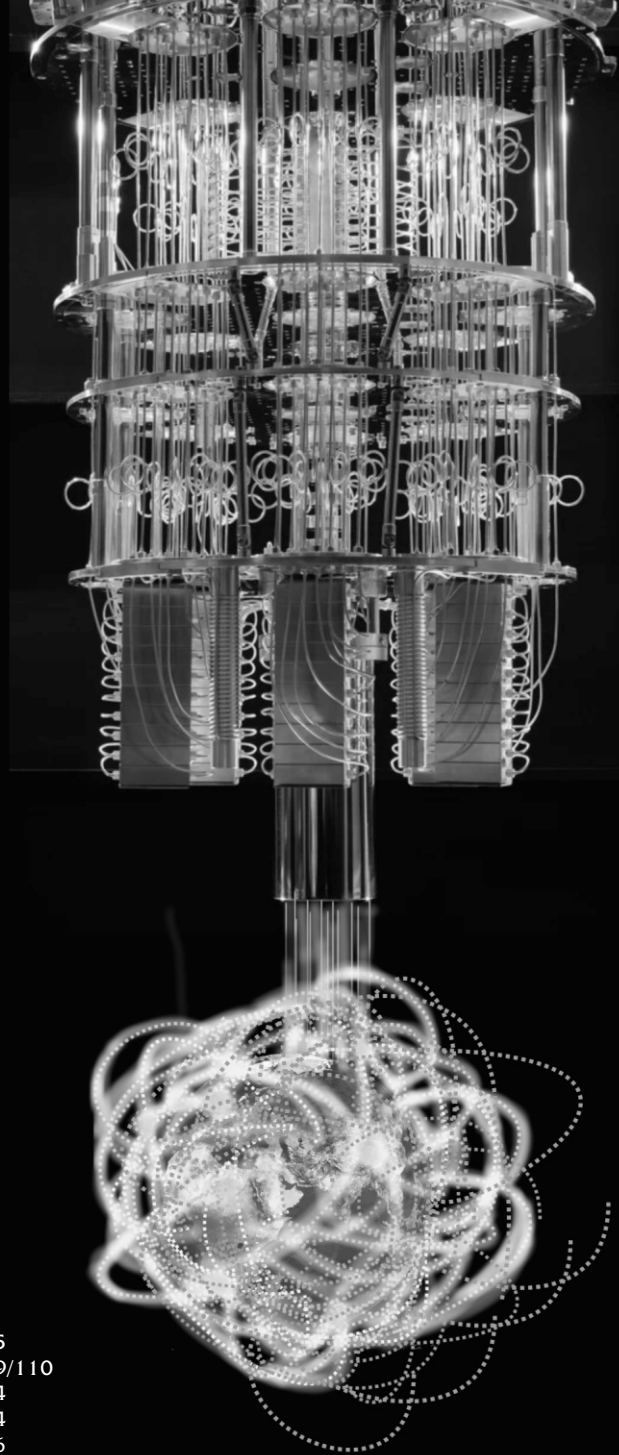
The 15 Puzzel Explorer, by David Dirkse
The Chamelon Game for the Web, written in Pascal,
by Detlef Overbeek & Mattias Gärtner

CODE SNIPPET TMS RichEdit for VCL/FMX/FNC Delphi and Lazarus by Detlef Overbeek
Web Service Part 4, Deploy to Internet Information Services, by Danny Wind



BLAISE PASCAL MAGAZINE 96

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



CONTENT

ARTICLES

From your editor Page 4

From our technical advisor

Cartoon sfrom Jerry King Page 5

Python for Delphi with Python4Delphi

by Max kleiner Page 7

Mustache Templates in Pascal, One popular template
language definition is called Mustache,

by Michael van Canneyt Page 15

Quantum Internet already active,

by Detlef Overbeek Page 27

The 15 Puzzel Explorer,

by David Dirkse Page 46

The Chamelon Game for the Web, written in Pascal,

by Detlef Overbeek & Mattias Gärtner Page 62

CODE SNIPPET TMS RichEdit

for VCL/FMX/FNC Delphi and Lazarus

by Detlef Overbeek Page 78

Web Service Part 4,

Deploy to Internet Information Services,

by Danny Wind Page 86

ADVERTISERS

Barnsten	Page 85
Components4Developers	Page 109/110
Delphi Community version	Page 14
Delphi Company	Page 84
Lazarus Handbook - Pocket (softcover) SUMMER SALE	Page 6
Lazarus Handbook - Hardcover SUMMER SALE	Page 13
Subscription+Hardcover Lazarus Handbook	Page 61
Subscription+SoftCover Lazarus Handbook	Page 77
Subscription+ Library USB Stick	Page 45
Super Offer Bundle	Page 26



Niklaus Wirth

Pascal is an imperative and procedural programming language, which Niklaus Wirth designed (left below) 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 - Netherlands



Contributors

Stephen Ball http://delphiaball.co.uk @DelphiABall		Peter Bijlsma -Editor peter @ blaiseascal.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 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.com
Boian Mitov mitov @ mitov.com		Jeremy North jeremy.north @ gmail.com
Detlef Overbeek - Editor in Chief www.blaiseascal.eu editor @ blaiseascal.eu	Howard Page Clark hdpc @ talktalk.net	Heiko Rempel info @ rompelsoft.de
Wim Van Ingen Schenau -Editor wisone @ xs4all.nl	Peter van der Sman sman @ prisman.nl	Rik Smit rik @ blaiseascal.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	Danny Wind dwind @ delphicompany.nl	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@blaiseascal.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	€ 155,96	€ 250	€ 80,00
Electronic Download Issue 60 pages	€ 64,20	€ 70	—
Printed Issue inside Holland (Netherlands) 60 pages	—	€ 250,00	€ 70,00

Subscriptions can be taken out online at www.blaiseascal.eu or by written order, or by sending an email to office@blaiseascal.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 Programmeertaal 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@blaiseascal.eu

Copyright notice

All material published in Blaise Pascal is copyright © SOPP Stichting Ondersteuning Programmeertaal Pascal unless otherwise noted and may not be copied, distributed or republished without written permission. Authors agree that code associated with their articles will be made available to subscribers after publication by placing it on the website of the PGG for download, and that articles and code will be placed on distributable data storage media. Use of program listings by subscribers for research and study purposes is allowed, but not for commercial purposes. Commercial use of program listings and code is prohibited without the written permission of the author.



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

KB



From your editor

This is a special edition with a lot of news: new versions of the Delphi Community Edition, Lazarus and FPC and Components4Developers (kbnMW). For Lazarus can be said this is a bug-update, with FreePascal having an extra update, but also mainly a bug update. About Delphi Community Edition: it was a long time ago since the last. Components4Developers has a great new item: Internationalisation: the article about that was published in 94/95.

There finally has arrived a new kind of internet: The Quantum Internet. I try to explain in a special article not only how it works but try to mention some special aspects and what the implications are. I made contact with the TU Delft (Netherlands) and they promised to keep us informed about their new development. I want to show them what we can do with our Pascal for the future development of special applications. Read the article on page 27. It offers new chances.

At the end of the year we want to create a special event: Pascal has this year its 50th birthday and BPM publishes its English spoken 100th Issue. If you have any request for this or ideas let me know: editor@blaise Pascal.eu we will work together with "Barnsten" and organise an event that will be done online and if possible (Corona) we will do that in a not yet known location.

David Dirkse has created yet another game: a very simple game. But to solve it is very hard. Because of this game was as always written in Delphi 7, I have transformed it to work under the latest version "Sidney". It will also become available under Lazarus and that has two consequences: first of all I needed to transfer two components to Lazarus that yet didn't exist and after that the game has to be transformed to Lazarus. So that will be published in the next Issue.

I wish you all a very nice summer and - if possible - holiday.
Hoping you are interested in this new Issue,
if not let me know.

Detlef Overbeek

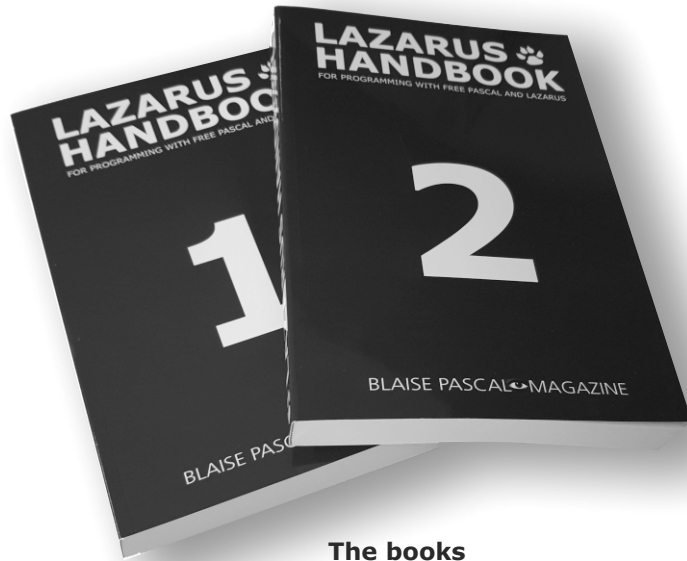




*“I’m not familiar with quantum computing,
but I have heard they may have heating problems.
Thank you for calling tech-support.”*



SUMMERSALE



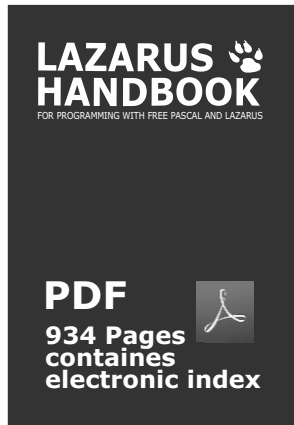
The books

Sewn POCKET , almost thousand pages

written by the makers of FPC and Lazarus

934 Pages in two books **40 €** (euro)

+



+



Including the PDF and Code Examples

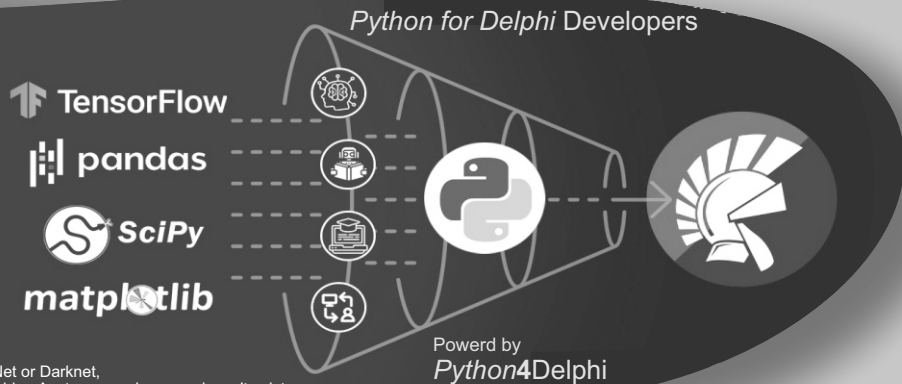
<https://www.blaisepascalmagazine.eu/product/lazarus-handbook-pocket/>

We have a new service at our website, for shipping you can make three choices:

1. The shipping cost depending on which part of the world you want the book to be shipped:
Europe or the other countries: the World

SHIPPING COST Europe	SHIPPING COST Europe + Registered	SHIPPING COST Europe + Track & Trace	SHIPPING COST World	SHIPPING COST World + Registered	SHIPPING COST World + Track & Trace
----------------------	-----------------------------------	--------------------------------------	---------------------	----------------------------------	-------------------------------------

maXbox starter 86



In a future world you can decide belonging to SkyNet or Darknet, but you can not find the difference between an Android or Avatar cause humans doesn't exist anymore.

Python for Delphi (P4D) is a set of free components that wrap up the Python DLL into Delphi and Lazarus (FPC). A DLL could be for example the 'python37.dll'. They let you almost easily execute Python scripts, create new Python modules and new Python types. You can create Python extensions as DLLs and much more like scripting or automating your console. P4D provides different levels of functionality:

- Low-level access to the Python API
- High-level bi-directional interaction with Python
- Access to Python objects using Delphi custom variants (VarPyth.pas)
- Wrapping of Delphi objects for use in Python scripts using RTTI (WrapDelphi.pas)
- Creating Python extension modules with Delphi classes, records and functions
- Generate Scripts in maXbox from a Python engine.

P4D makes it, after some (tough) studies, very easy to use Python as a scripting language for Delphi applications. It also comes with an extensive range of demos and useful (video) tutorials. So a very first simple approach is to call the Python dll without a wrapper or mapper e.g. call the copyright function:

```
//if fileExistst(PYDLLPATH+ 'python37.dll';
function getCopyRight: Pchar;
external 'Py_GetCopyRight@C:\maXbox\EKON25\python37.dll stdcall';
```

Then we call the function with a pre-test:

```
function IsDLLOnSystem(DLLName:string): Boolean;
var ret: integer; good: boolean;
begin
    ret:= LoadLibrary(pchar(DLLNAME));
    Good:= ret>0;
    if good then FreeLibrary(ret);
    result:= Good;
end;
if isDLLOnSystem(PYDLLPATH+PYDLLNAME) then begin
    showMessage('py dll available');
    writeln(getCopyRight)
end;
```

Copyright (c) 2001-2019 Python Software Foundation.
All Rights Reserved.
Copyright (c) 2000 BeOpen.com.
All Rights Reserved.
Copyright (c) 1995-2001 Corporation for National Research Initiatives.
All Rights Reserved.
Copyright (c) 1991-1995 Stichting Mathematisch Centrum, Amsterdam.
All Rights Reserved.

We also use to invoke a Python script as an embedding const and use the dll functionality of `Import('PyRun_SimpleString');`

```
procedure InitSysPath;
var _path: PPyObject;
    const Script =
        'import sys' + sLineBreak+
        'sys.executable = r"%s"' + sLineBreak+
        'path = sys.path' + sLineBreak+
        'for i in range(len(path)-1, -1, -1):' + sLineBreak+
        '    if path[i].find("site-packages") > 0:' + sLineBreak+
        '        path.pop(i)' + sLineBreak +
        'import site' + sLineBreak+
        'site.main()' + sLineBreak+
        'del sys, path, i, site';
begin
    if VenvPythonExe <> '' then
        ExecString(AnsiString(Format(Script, [VenvPythonExe])));
    _path := PySys_GetObject('path');
    if Assigned(FOnSysPathInit) then
        FOnSysPathInit(Self, _path);
end;
```

HOW TO USE PYTHON4DELPHI

The best way to learn about how to use P4D is to try the extensive range of demos available. Also studying the unit tests for `VarPyth` and `WrapDelphi` can help understand what is possible with these two units and the main `PythonEngine.pas`.

Lets start with the `VarPyth.pas` unit.

This allows you to use Python objects like COM automation objects, inside your Delphi source code. This is a replacement, bugfixed of the former `PythonAtom.pas` that uses the new custom variant types introduced since Delphi6.

You may use these Python Variants in expressions just as you would any other Variants or automation e.g.:

```
var
    a: Integer; v: Variant;
begin
    v:= VarPythonEval('2 ** 3');
    a:= v;
```


The functions provided by this unit `VarPyth.pas` are largely selfexplanatory.

WHAT ABOUT THE `WrapDelphi.pas` UNIT?

You can use P4D to create Python extension modules too that expose your own classes and functions to the Python interpreter. You may package your extension with `setuptools` and distribute it through `PyPi`.

So if you have an existing object or unit in Delphi that you'd like to use in Python but you don't want to modify that object to make it Python-aware, then you can wrap that object just for the purposes of supplying it to Python with a package. Using `TPyDelphiObject` you can wrap any Delphi object exposing published properties and methods.

Note that the conditional defines `TYPEINFO` and `METHODINFO` need to be on. As an example, if you have an existing Delphi class simply called `TRGBColor`:

```
TRGBColor = class
  private
    fRed, fGreen, fBlue: Integer;
  public
    property Red: read fRed write fRed;
    property Green: read fGreen write fGreen;
    property Blue: read fBlue write fBlue;
end;
```

You want to use `Color` within some Python code but you don't want to change anything about the class. So you make a wrapper inherited from `TPyObject` that provides some very basic services, such as getting and setting attributes of a `Color`, and getting a string representation:

```
TPyColor = class(TPyObject)
  private
    fColor: TRGBColor;
  public
    constructor Create( APythonType: TPythonType ); override;
    // Py Basic services
    function GetAttr(key: PChar): PPyObject; override;
    function SetAttr(key: PChar; value: PPyObject): Integer; override;
    function Repr: PPyObject; override;
end;
```

The project in the subdirectory Delphi generates a Python extension module (a DLL with extension “pyd” in Windows) that allows you to create user interfaces using Delphi from within Python. The whole VCL (almost and maybe) is wrapped with a few lines of code! The small demo TestApp.py gives you a flavour of what is possible. The machinery by which this is achieved is the WrapDelphi unit.

The subdirectory DemoModule demonstrates how to create Python extension modules using Delphi, that allow you to use in Python, functions defined in Delphi. Compile the project and run test.py from the command prompt (e.g. py test.py). The generated pyd file should be in the same directory as the Python file. This project should be easily adapted to use with Lazarus and FPC.

The subdirectory RttiModule contains a project that does the same as the DemoModule, but using extended RTTI to export Delphi functions. This currently is not supported by FPC. The unit PythonEngine.pas is the main core-unit of the framework. You are responsible for creating one and only one TPythonEngine. Usually you just drop it on your main form. Most of the Python/C API is presented as member functions of the engine.

```

type
  TPythonVersionProp = record
    DllName : string;
    RegVersion : string;
    APIVersion : Integer;
  end;
...
Py_BuildValue := Import('Py_BuildValue');
Py_Initialize := Import('Py_Initialize');
PyModule_GetDict := Import('PyModule_GetDict');
PyObject_Str := Import('PyObject_Str');
PyRun_String := Import('PyRun_String');
PyRun_SimpleString := Import('PyRun_SimpleString');
PyDict_GetItemString := Import('PyDict_GetItemString');
PySys_SetArgv := Import('PySys_SetArgv');
Py_Exit := Import('Py_Exit');
...

```



Let's take a last look at the functionality of PyRun_SimpleString mentioned first within the const script.
http://www.softwareschule.ch/examples/1016_newsfeed_sentiment_integrate2.txt

```
PyRun_SimpleString: function(str: PAnsiChar): Integer; cdecl;
```

We see that we have to pass a PAnsiChar in cdecl convention and map to ExecString (PyRun_String):

```

procedure TPythonEngine.ExecString(const command: AnsiString);
begin
  Py_XDECREF( Run_CommandAsObject( command, file_input ) );
end;

```



Figure :1 tutor86_pythondll_spy.png

CONCLUSION

The P4D library provides a bidirectional bridge between Delphi and Python.

It allows Delphi applications to access Python modules and run Python scripts.

On the other side it makes Delphi/Lazarus objects, records, interfaces, and classes accessible to Python, giving Python the ability to use this methods.

Before you try the demos please see the Wiki topic "How Python for Delphi finds your Python distribution" at <https://github.com/pyscripter/python4delphi/wiki/FindingPython>

You will need to adjust the demos accordingly, to successfully load the Python distribution that you have installed in your PC, e.g. C:\Users\max\AppData\Local\Programs\Python\Python36\.

PYTHONENGINE:

The core of Python for Delphi. Provides the Python API with dll mapper and runtime configuration.

VarPyth: VarPyth wraps Python types as Delphi custom variants.

WrapDelphi: Uses RTTI (in a DLL) so you can use Delphi objects from Python without writing individual wrapper classes or methods.

TPythonGUIInputOutput: Provides a Python console you can drop on a form and execute a Python script from a memo. The TPythonEngine class is the single global engine block shared by all Python code.

VarPyth wraps Python types as Delphi custom variants.

VarPyth requires at least Delphi v6.

This Tutorials folder contains text and video, webinar tutorials accompanied with slides and demo source code. Next time I'll show a few bidirectional demos with implementation details.

WIKI P4D TOPICS

- Installation
- Supported Platforms
- How Python for Delphi finds your Python distribution (read before trying the demos)

LEARN ABOUT PYTHON FOR DELPHI

- Tutorials
- Demos: <https://github.com/maxkleiner/python4delphi>

Dealing with internals of Python means also you get the original stack-trace errors back like
 (TPythonEngine.RaiseError;): File
 C:\Users\max\AppData\Local\Programs\Python\Python36\lib\sitepackages\pandas\core\indexing.py,
 line 1304, in
 _validate_read_indexer raise KeyError(f"{not_found} not in index")
 KeyError: "['id', 'links', 'published_parsed', 'published', 'title_detail', 'guidislink', 'link', 'summary_detail'] not in index"

The script can be found:

http://www.softwareschule.ch/examples/1016_newsfeed_sentiment_integrate2.txt

Ref Video:

<https://www.youtube.com/watch?v=jLuxTfct3CU>

<https://github.com/pyscripter/python4delphi/wiki/FindingPython>

You will need to adjust the demos accordingly, to successfully load the

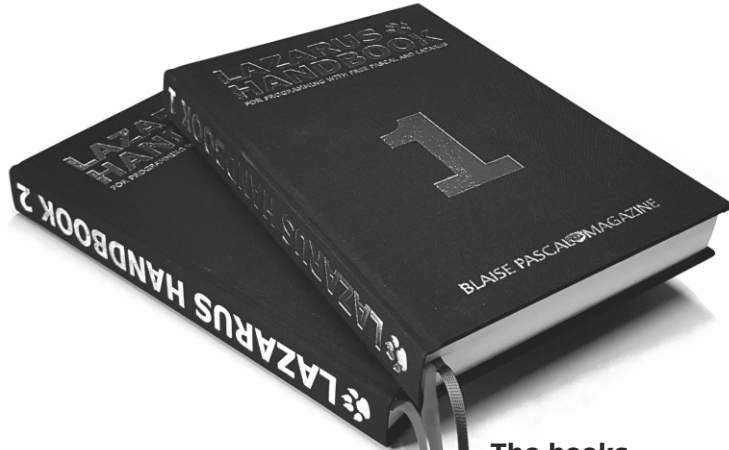
Python distribution that you have installed on your computer. Doc: <https://maxbox4.wordpress.com>

APPENDIX: CATALOG OF THE DEMOS:

- Demo01 A simple Python evaluator
- Demo02 Evaluate a Python expression
- Demo03 Defining Python/Delphi vars
- Demo04 Defining Python/Delphi vars (advanced case)
- Demo05 Defining a new Module
- Demo06 Defining a new Type
- Demo07 Using Delphi methods as Python functions
- Demo08 Using Delphi classes for new Python types
- Demo09 Making a Python module as a Dll
- Demo10 FireDAC Database demo using FireDAC
- Demo11 Using Threads inside Python
- Demo16 Using a TDelphiVar or Module methods
- Demo17 Using variant arrays of 2 dimensions
- Demo19 C++ Builder: this is a replicate of the Delphi Demo05
- Demo20 C++ Builder: this is a replicate of the Delphi Demo08
- Demo21 Using Events in TPythonModule or TPythonType
- Demo22 Using Threading, Windows Console and Command line arguments
- Demo23 Using Threading and Delphi log window
- Demo25 Using VarPyth.pas
- Demo26 Demo08 revisited to allow the new Python type to be subclassed
- Demo27 Container indexing
- Demo28 Iterator
- Demo29 Using Python Imaging Library (PIL)
- Demo30 Using Named Parameters
- Demo31 Using WrapDelphi to access Delphi Form attributes
- Demo32 Demo08 revisited using WrapDelphi
- Demo33 Using Threads inside Python
- Demo34 Dynamically creating, destroying and recreating PythonEngine.



Summersale

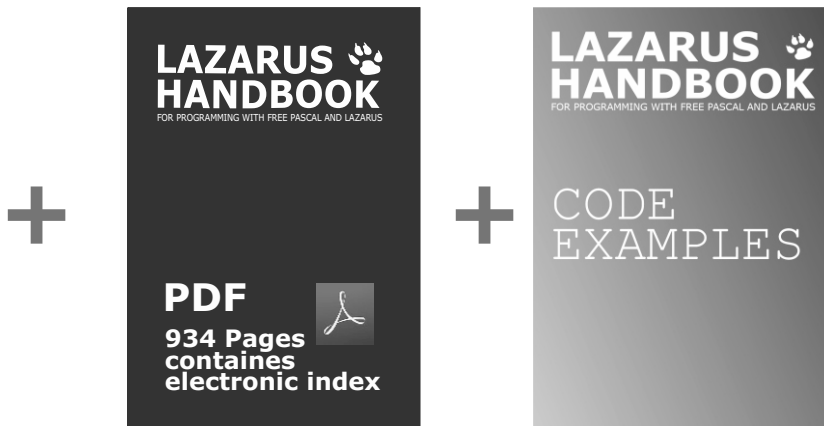


The books

Sewn Hardcover, almost thousand pages

written by the makers of FPC and Lazarus

934 Pages in two books **65 €** (euro)



Including the PDF and Code Examples

<https://www.blaisepascalmagazine.eu/product/lazarus-handbook-hardcover/>

We have a new service at our website, for shipping you can make three choices:

1. The shipping cost depending on which part of the world you want the book to be shipped:
Europe or the other countries: the World

SHIPPING COST Europe	SHIPPING COST Europe + Registered	SHIPPING COST Europe + Track & Trace	SHIPPING COST World	SHIPPING COST World + Registered	SHIPPING COST World + Track & Trace
----------------------------	--	---	---------------------------	---	--

Download Now

<https://www.embarcadero.com/products/delphi/starter>

**Delphi & C++Builder
Community Editions Now
Available in Version 10.4.2!**



embarcadero®



Download Now

**Delphi & C++Builder Community Editions Now Available in
Version 10.4.2!**

The free and full-featured Community Editions of Delphi and C++Builder are designed to help you get started programming with these powerful IDEs quickly and explore their robust app development features.

Now, the Delphi & C++Builder Community Editions are now available in version 10.4.2, the latest version released in February of this year!

MUSTACHE TEMPLATES IN PASCAL

By Michael van Canneyt

Templates are widely used in websites and web applications. One popular template language definition is called Mustache. For a long time dmustache was the only Pascal implementation of the mustache template language. Now there is a second implementation available.

1 INTRODUCTION

Templates are very old. You can find them in many places: from Microsoft Mail merge to many web templating languages: smarty, twig etc. One such templating system is Mustache: <http://mustache.github.io/>

The nice thing about mustache is that it is not bound to a programming language: it is just a description of how the templating should work, with a reference implementation. As can be seen on the website, implementations in many programming languages are available.

Mustache is also very low-level: its motto is 'Logic-less templates'. That means there is no way to put logic in the templates. The only logic are sections, which are an encapsulation of array data.

In its most basic form, Mustache will change a template `{{Something}}` by the value of the "variable" Something. You can also use scopes: `{{Customer.Name}}` will be replaced by the Name property of the Customer object.

It further supports sections, and some special whitespace handling.

Mustache is geared towards generating HTML in that the specification mandates that substitutions must be html escaped (although this can be disabled). The complete specification is available at the above website.

Mustache assumes JSON as the source of data, so the above 2 templates would need JSON data in the following form:

```
{
  "Something" : 12,
  "Customer" : {
    "Name" : "Pan",
    "FIRSTNAME" : "PETER"
  }
}
```



But the source of the template data can be anything, an implementation of Mustache can provide alternate sources of data. The dMustache implementation exists since many years.

So why a new implementation of mustache templates? There are 2 reasons. The first reason is that dMustache, available at:

<https://github.com/synopse/dmustache>

depends on the Synopse mORMot framework (part of it is included in the repository). This means it does not compile for all platforms that Delphi supports. The second reason is that it fails several key tests in the Mustache testsuite. If you don't care about these reasons, dmustache is a good choice, it is highly optimized and offers several extensions to the Mustache templating specification, including limited logic.

2 FPMUSTACHE AND MUSTACHED

Free Pascal now contains a fpMustache unit in its distribution. It has no dependencies except the JSON support of Free Pascal. To be able to use the new engine in Delphi, it has been ported to Delphi, and can be downloaded freely at

<https://gitlab.com/mvancanneyt/mustached>

The unit in the Delphi implementation is called mustache, in Free Pascal it is called fpmustache.

The Delphi implementation comes with 2 GUI demo programs. The mustache unit contains many objects, most of which are auxiliary classes used to compile the template. The central object in the implementation is the TMustache class.

It is a component and can be dropped on a form or datamodule.

```

Tmustache = class(TComponent)
Public
  Procedure Compile;
  Procedure Render(aContext : TMustacheContext; aOutput : TMustacheOutput);
  Function Render(aContext : TMustacheContext) : TMustacheString;
  Function Render(const aJSON : TJSONValue) : TMustacheString;
  Function Render(const aJSON : String) : TMustacheString;
  Class function CreateMustache(aOwner: Tcomponent; aTemplate: TMustacheString): TMustache;
  Class Function Render(aTemplate : TMustacheString; const aJSON : String) : TMustacheString;
Published
  Property Template : TMustacheString;
  Property OnGetValue : TGetTextValueEvent;
  Property StartTag : TMustacheString;
  Property StopTag : TMustacheString;
  Property Partial : TStrings;
end;

```



From this declaration, you can see that the most simple usage of this component is as follows:

```

program t1;
  {$APPTYPE CONSOLE}
  // For delphi, only the mustache unit needs to be used.

  uses jsonparser, fpmustache;

  Const
    JSON = '{ "products" : [ { "name" : "BMW" }, '+
           '{ "name" : "Mercedes" }, '+
           '{ "name" : "Audi" } ] }';

  // Mock markdown table
  Template =
    '| name |'+sLineBreak+
    '|-----|'+sLineBreak+
    '| { # products} | { {name} } |'+sLineBreak+
    '| { /products} |';

  begin
    Writeln(TMustache.Render(Template,JSON));
  end.

```

The output will look like this:

```

| name |
|-----|
| BMW |
| Mercedes |
| Audi |

```

Which is a markdown table, containing the 3 cars specified in the JSON. As you can see, for simple cases like this it is not even necessary to create an instance of the `TMustache` class. However, if you do create an instance of `TMustache` by putting it on a form, You can set the following properties and events:

- **Template** A String with the Mustache template.
- **StartTag** The starting characters for a tag, by default this is `{{` .
- **StopTag** The ending characters of a tag, by default this is `}}` .

With these basic properties, one can get started. However, for more advanced usage, the following properties are also available:

OnGetValue This is an event that allows you to return values for template names that are not in the JSON which will be fed to the mustache engine.

It has the following signature:

```

TGetValueEvent = Procedure (Const aName : TMustacheString;
                             var aHandled : Boolean;
                             var aValue : TMustacheString) of Object;

```



The `aName` parameter contains the name of the value the engine needs, and you must return the value for this name in `aValue`. On return, `aHandled` must be set to `True`. If not, an empty value will be assumed.

- **Partials** Mustache specifies that you can include other templates, called partials:

```
{{> subtemplate}}
```

When the engine encounters this, it must locate the `subtemplate` partial template, and process it as if its contents was inserted instead of the above template. The `Partials` property allows you to map partial names to actual partial templates:

```
name={{#names}}{{name}}{/names}}
```

With all these properties set, the methods of `TMustache` can be used to actually render a template:

- **Compile** Compiles the template: this method will parse the template and compile it into a form that will allow it to be rendered multiple times without the need to parse it again at each invocation of the `Render` method. Under normal circumstances, it should not be necessary to call this method: the `Render` method will call it when needed.
- **Render** Will render the template with given data. This method has several forms. The simplest form takes a string with JSON data as input, and returns the template as rendered with the JSON data:

```
Function Render(const aJSON : String) : TMustacheString;
```

The second form is actually used by the first form: here the data to use when rendering is actually specified as a JSON object:

```
Function Render(const aJSON : TJSONValue) : TMustacheString;
```

The third form uses a context object: the `TMustacheContext` class, which is responsible for providing the data to the render process:

```
Function Render(aContext : TMustacheContext) : TMustacheString;
```

The 2 `Render` objects that accept JSON actually call this method and use a descendent of the `TMustacheContext` class called `TMustacheJSONContext`, to fetch the data from the provided JSON string (or object). The output of the above `Render` methods is always a string. But in fact, the output can be sent to whatever destination you want: a stream, a file, whatever.



This brings us to the last form of the Render method:

```
Procedure Render (aContext : TMustacheContext; aOutput : TMustacheOutput);
```

The TMustacheOutput class is very simple

```

TMustacheOutput = Class(TObject)
Public
Procedure Output(Const aText : TMustacheString); virtual; abstract;
Procedure Reset; virtual; abstract;
end;
    
```

The output method is called whenever rendered text needs to be appended to the output. The standard Render methods use a TMustacheStringOutput descendent of this class, which simply concatenates all text into one string, which is then returned by the Render function.

3 A SIMPLE DEMO

Mustache template demo
- 🔍 ✕

<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Template <pre> <h1>{{header}}</h1> {{#bug}} {{/bug}} {{#items}} {{#first}} {{name}} {{/first}} {{#link}} {{name}} {{/link}} {{/items}} {{#empty}} <p>The list is empty.</p> {{/empty}} </pre> </div>	<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> JSON <pre> { "header": "Colors", "items": [{ "name": "red", "first": true, "url": "#Red" }, { "name": "green", "link": true, "url": "#Green" }, { "name": "blue", "link": true, "url": "#Blue" }], "empty": false } </pre> </div>
<div style="border: 1px solid #ccc; padding: 5px; display: inline-block; margin: 0 auto;">Render</div>	
<div style="border: 1px solid #ccc; padding: 5px; margin-bottom: 5px;"> Result <pre> <h1>Colors</h1> red green blue </pre> </div>	

Figure 1: A simple demo





To demonstrate this, we will recreate the demo on the mustache website as a Lazarus program. (the same program exists for Delphi in the gitlab repository) To this end, we put 3 memos on a form, and a button to render the content. The `mTemplate` memo component contains the sample template from the website, the `mJSON` memo contains the JSON to feed as data to the rendering engine. The result will be rendered in the `mResult` memo. The Mustache component is created in the Form's `OnCreate` event.

```

{ procedure TMainForm.FormCreate(Sender: TObject);
  begin
    FMustache:=TMustache.Create(Self);
  end;
}
    
```

Doing so we don't need to put the mustache component on the component palette. We do need to provide the path to the source of the `fpMustache` unit in the project's compiler settings. (The same is done in the Delphi demo) When the user presses the `bRender` button, the following code is executed:

```

{ procedure TMainForm.BRenderClick(Sender: TObject);
  begin
    MResult.Text := FMustache.Render(mTemplate.Text,mJSON.Text);
  end;
}
    
```

The result of all this can be seen in figure 1 on page 5.

4 FEEDING DATASET DATA TO THE RENDERER

The above example uses plain JSON to feed the data to the template renderer. But what if your data is in a dataset? One way would be to transform the dataset data to a JSON array of records, and then feed that JSON data to the renderer. However, this would involve a lot of copying of data, slowing things down and increasing memory requirements.

Instead, the `TMustacheContext` can be expanded: the unit `dbmustache` (or `fpdbmustache`) contains a descendant of `TMustacheContext` called `TMustacheDBContext`.

This context object will fetch the data from one or more datasets that can be attached to it.

Each dataset is given a section name, which can be used in the template to refer to a dataset in the collection of datasets.

Given a dataset with name `dsFamily`, the following template

```

{
  {{#dsFamily}}
  <tr>
  <td>{{name}}</td><td>{{age}}</td>
  </tr>
  {{/dsFamily}}
}
    
```

would produce a set of HTML table rows with the contents of the fields name and age.



Likewise, a template

```
{{dsFamily.name}}
```

would render the field name of the dsFamily dataset.

The `TMustacheDBContext` class (a class descendent from `TObject`, so not a `TComponent`) has the following methods and properties:

```
TMustacheDBContext = Class(TMustacheContext)
Public
  Procedure Clear;
  Procedure AddDataset(aDataset : TDataset; aSectionName : String = "");
  Procedure RemoveDataset(aDataset : TDataset);
  Property StaticValues : TStrings;
  Property Datasets[aIndex : Integer] : TDatasetCollectionItem;
  Property DatasetCount : Integer;
end;
```

The meaning of these methods is pretty straightforward:

- **Clear**
Clear the dataset list and the `StaticValues`.
- **AddDataset**
Add a dataset to the list of datasets. The section name can be specified. When the name is omitted, the name of the dataset component is used.
- **RemoveDataset**
Remove a dataset from the list of datasets.
- **StaticValues**
A stringlist containing name=value pairs.
This can be used to provide the template renderer with values that are not available as a field in one of the provided datasets.
- **Datasets**
Indexed access to the Dataset/Section name items.
- **DatasetCount**
Number of registered dataset items.

We can change the GUI example program to use this dataset context: we drop a CSV dataset on the form (csvFamily), connect it to a grid (GData) and a DBNavigator (NavData), which we put instead of the MJSON memo.

The data of the CSV dataset is loaded in the form OnCreate handler:

```
procedure TMainForm.FormCreate(Sender: TObject);
begin
  FMustache:=TMustache.Create(Self);
  csvFamily.FileName:=ExtractFilePath(ParamStr(0))+‘family.csv’;
  csvFamily.Open;
end;
```



The `OnClick` method of the `BRender` button now must use the `TMustacheDBContext` class to feed the data to the render method:

```

procedure TMainForm.BRenderClick(Sender: TObject);
Var
  C: TMustacheDBContext;
begin
  C:=TMustacheDBContext.Create(Null);
  try
    C.AddDataset(csvFamily,'data');
    C.StaticValues.Values['title']:= 'Our family';
    fMustache.Template:=mTemplate.Text;
    MResult.Text:=fMustache.Render(C);
  finally
    C.Free;
  end;
end;

```

In the code, a static value called `title` is added to the context, which will contain the title of the rendered HTML page. After setting the `Template` property, and calling the `Render` method using the created `TMustacheDBContext` instance, the result will look something like figure 2 on page 8.

5 ADDING SIMPLE LOGIC: USING EXPRESSIONS

The `TMustache` component is set up in such a manner that descendents can be created which initialize their own template parser, which must be an instance of `TMustacheParser`.

This allows you to extend the template parser with support for new kinds of templates.

The template is parsed into a DOM-like structure, based on `TMustacheElement` classes.

When the template is compiled, the parser uses a factory function to create the necessary mustache elements for the various template elements. You can override this function (called `CreateDefault`) in a descendent of `TMustacheParser`, to check for special characters and create a special element with customized handling for the template.

The examples from the previous sections and the above explanation will work both in Delphi and Lazarus. What follows next is only available in Free Pascal: the `fpexmustache` unit contains a descendent of the `TMustache` component, called `TMustacheExpr`.



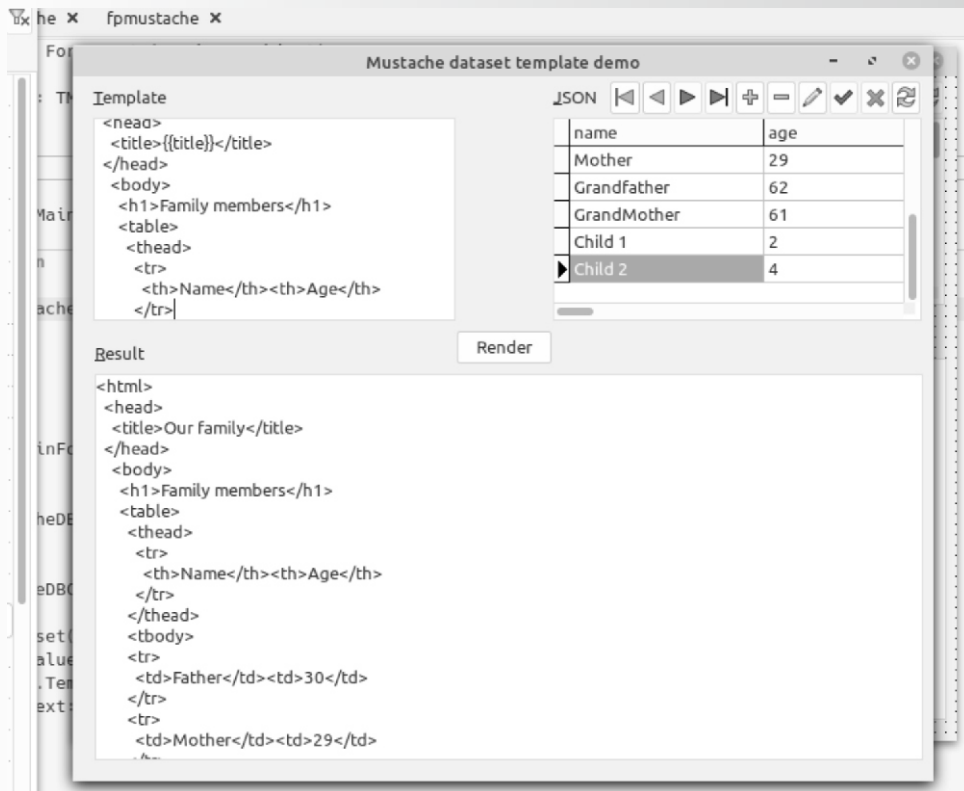


Figure 2: Rendering using a dataset

This component uses an expression calculation engine only available in Free Pascal, hence the component is only available for Free Pascal/Lazarus, and not for Delphi. The `TMustacheExpr` class creates a `TMustacheExprParser` parser class. It overrides the `CreateDefault` method to allow you to enter an expression:

```
[[IF (age >= 18, 'adult', 'minor')]]
```

The square brackets inside the template indicate that the content of the template is an expression. All expressions supported by the `TFPEXpressionParser` component of Free Pascal are allowed. The above is quite simple: if the variable `age` is larger than 18, then `adult` will be rendered, else `minor`.

The expression engine supports various datatypes: whatever the result type, it will be converted to a string using standard functions in the `SysUtils` unit, and rendered. The expression engine of Free Pascal itself of course offers a lot of conversion functions (most of them mappings from various `SysUtils` functions), so you can format the result yourself in your expression.

When using the expression engine, it needs to know what variables are available and what their types are. The available variable names are the same ones as in the JSON data given to the renderer. However, they must be explicitly registered with the expression engine.



This is done using the `RegisterVariables` method of the `TMustacheExpr` class, which comes in several overloaded versions:

```

Procedure RegisterVariables (aContext : TMustacheJSONContext;
                             aPath : TJJSONStringType = "";
                             UseEvent : Boolean = True);

Procedure RegisterVariables (aJSON : String;
                             aPath : TJJSONStringType = "";
                             UseEvent : Boolean = True);

Procedure RegisterVariables (aJSON : TJJSONObject;
                             aPath : TJJSONStringType = "";
                             UseEvent : Boolean = True);
    
```

As you can see, the variable values can be registered through multiple mechanisms: as a JSON string, a JSON object, or a JSON context. The `aPath` parameter is a path in the JSON object: only keys below this path will be registered. For example, given the following JSON:

```

{
  "data" : {
    "customer" : {
      „firstname" : "John",
      "lastname" : "Doe"
    }
  }
}
    
```

using the path `data.custmer` will only register the `firstname` and `lastname` values.

The use of `UseEvent` needs some explanation. If `UseEvent` is false, the variables will be registered as static values: their names, types and values will be taken from the JSON given to the `RegisterVariables` call. That means if you call `Render` several times, each time using a different JSON, `firstname` and `lastname` will have the value `John` and `Doe` every time when you render the template.

This may be what you want, but it is more likely that this is not what you want: in `RegisterVariables` you just want to register the names using a sample JSON, but the actual value must be fetched when rendering from the JSON supplied to the `Render` method.

If `UseEvent` is `True` (the default), then only the names and types of the variables are registered, but the actual values of these variables will be retrieved while rendering with an event. The values will be retrieved using the same mechanism as used when getting template values. We'll demonstrate this with a small example, which demonstrates the expression given in the example above.




```

program demo2;

uses Classes, fpjson, jsonparser, fpmustache, fpexmustache;

Const // Mock markdown table
  Template =
    '| name | age |'+sLineBreak+
    '|-----|-----|'+sLineBreak+
    '{{#data}}| {{name}} |'+
    '{{IF(age>=18,"adult","minor')}} |'+sLineBreak+
    '{{/data}}';
Var M: TMustacheExpr; C: TJSONObject; F: TFileStream;
begin
  M:=TMustacheExpr.Create(Nil);
  try
    F:=TFileStream.Create('family.json',fmOpenRead);
    C:=GetJSON(F) as TJSONObject;
    M.RegisterVariables(C,'data[0]');
    M.Template:=Template;
    Writeln(M.Render(C));
  finally
    M.Free;
    F.Free;
    C.Free;
  end;
end.

```

As you can see, the program is again simplicity itself.
 The JSON is read from a file family.json that looks like this:

```

{
  "data" : [
    { "name" : "Father", "age": 30 },
    { "name" : "Mother", "age": 29 },
    { "name" : "Grandfather", "age": 62 },
    { "name" : "GrandMother", "age": 61 },
    { "name" : "Child 1", "age": 2 },
    { "name" : "Child 2", "age": 4 }
  ]
}

```

The path data[0] used in the RegisterVariables call indicates that the first record in the data array must be used to register the variables: this means name and age are registered as variables using an event to get the actual values.

The result of this program is the following markdown table:

```

| name | age |
|-----|-----|
| Father | adult |
| Mother | adult |
| Grandfather | adult |
| GrandMother | adult |
| Child 1 | minor |
| Child 2 | minor |

```

Which is the expected outcome.



6 CONCLUSION

Having a standard template engine available can be useful: it can be used to generate markdown, HTML, plain text, or any other text format, starting from a template and data. This could be used for a website, but also for reporting, and it is perfect for generating and sending mails, allowing the user to specify a template for the mail (html or plain text) and send the mail based on data in a database.



Advertisement

BLAISE PASCAL MAGAZINE

```

procedure
var
begin
for i := 1 to 9 do
begin
...
end
end
    
```

Prof Dr.Wirth, Creator of Pascal Programming language

Blaise Pascal, Mathematician



Editor in Chief: Detlef Overbeek
Edelstenenbaan 21 3402 xA
IJsselstein Netherlands



Prof Dr.Wirth, Creator of Pascal Programming language

editor@blaisepascalmagazine.eu
https://www.blaisepascalmagazine.eu

BLAISE PASCAL MAGAZINE

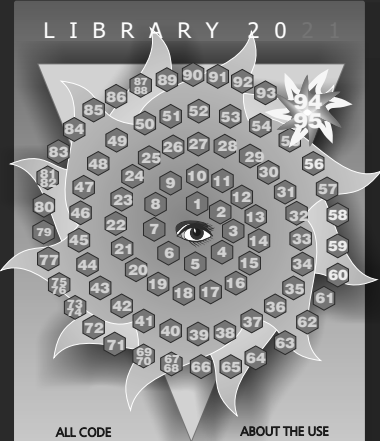
```

procedure
var
begin
for i := 1 to 9 do
begin
...
end
end
    
```

Prof Dr.Wirth, Creator of Pascal Programming language

Blaise Pascal, Mathematician

LIBRARY 2021



ALL CODE ABOUT THE USE

BLAISE PASCAL MAGAZINE

ALL ISSUES IN ONE FILE

LIBRARY 2020

Programming Language
PASCAL

for Delphi and Lazarus

VIDEO


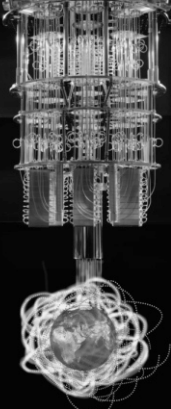
Blaise Pascal / Internet / Lazarus / WebAssembly
Pascal / Database / CSS / Ajax / Programming / Web Pages
Android / PDF / Mac / Windows / Linux

BLAISE PASCAL MAGAZINE

SUPER OFFER (5)
€ 150 ex Vat

SUMMERSALE

BLAISE PASCAL MAGAZINE 96

From our technical advisor Cannon from early King Python for Delphi with Python/Delphi by Max Steiner
Macros/Template by Michael van Gemert
One popular template language definition is called Macrotex by Michael van Gemert
Questions/Interest already active by Detlef Overbeek
The 15 Puzzle Explorer by David Dirkse
The Chessboard Game for the Nix by Detlef Overbeek
CODL SNIPPET TMS RichEdit in VCL/MFC: Death and Learning: Detlef Overbeek
Web Service Part 4: Deploy to Internal Information Services, by Danny Wind

LAZARUS HANDBOOK

FOR PROGRAMMING WITH FREE PASCAL AND LAZARUS

including 30 example projects

934 PAGES



LEARN TO PROGRAM HOWARD PAGE-CLARK
USING LAZARUS





DAVID DIRKSE



```

procedure ;
var
begin
for i := 1
to 9 do
begin
end;
end;
    
```

BLAISE PASCAL MAGAZINE
www.blaisepascal.eu

COMPUTER (GRAPHICS) MATH & GAMES IN PASCAL

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

By Detlef Overbeek

ABSTRACT:

To make a description of the intent of these articles (including future ones) I will try to put in some a basic structure and then explain every issue. The purpose of this article is to clarify what the quantum internet is, why it is necessary for us Pascal Programmers to delve into it - not out of sheer curiosity - but because of the awareness that the future is already here and that to understand everything we need to get deep into this new challenge (and opportunity), because - then with the help of Pascal - we can create new applications which can interact with the Quantum Internet or as a utility, or even as a component.

It's not easy, but cut a problem into pieces and it becomes manageable. Since all programmers have a very important mental resource: abstraction - and I hope fantasy too, it should be possible with the help of serious study to understand what is coming our way.

It offers many opportunities.



picture by Qtech

OVERVIEW

first of all there is this Hierarchy

- QUANTUM INTERNET - a structure it self
- Quantum Network Explorer - an intermediate between the "normal" Internet and the Quantum Internet
- The "normal" INTERNET - The Internet as we know it

there are 5 main items which we will handle:

- ① The TERMS - used in this article; they will be each handled and explained separately so you can in the middle of an article jump back to your basic information.
- ② The QUANTUM THEORY - the quantum environment. Don't be afraid. Its only asking a lot of your imagination. Here I will sometimes go back to already written articles that explain things even in detail
- ③ The QUANTUM INTERNET itself.
- ④ The Quantum Network Explorer - quantum internet kit
- ⑤ The societal and social consequences.

Before we start the article we need to explain THE TERMS:

- The Advanced Research Projects Agency Network (ARPANET)
- QuBits
- Entanglement
- Quantum Memory
- The nitrogen-vacancy center (N-V center or NV center)
- Quantum key distribution (QKD)
 - Quantum Repeater
 - Quantum Teleportation.





THE ADVANCED RESEARCH PROJECTS AGENCY NETWORK (ARPANET)

The history overview has been published as an extra at the end of the book *Learn to program using Lazarus*. See the specific article below. *The ARPANet was the first wide-area packet-switched network with distributed control and one of the first networks to implement the TCP/IP protocol suite. Both technologies became the technical foundation of the Internet. The ARPANET was established by the Advanced Research Projects Agency (ARPA) of the United States Department of Defense.*

A

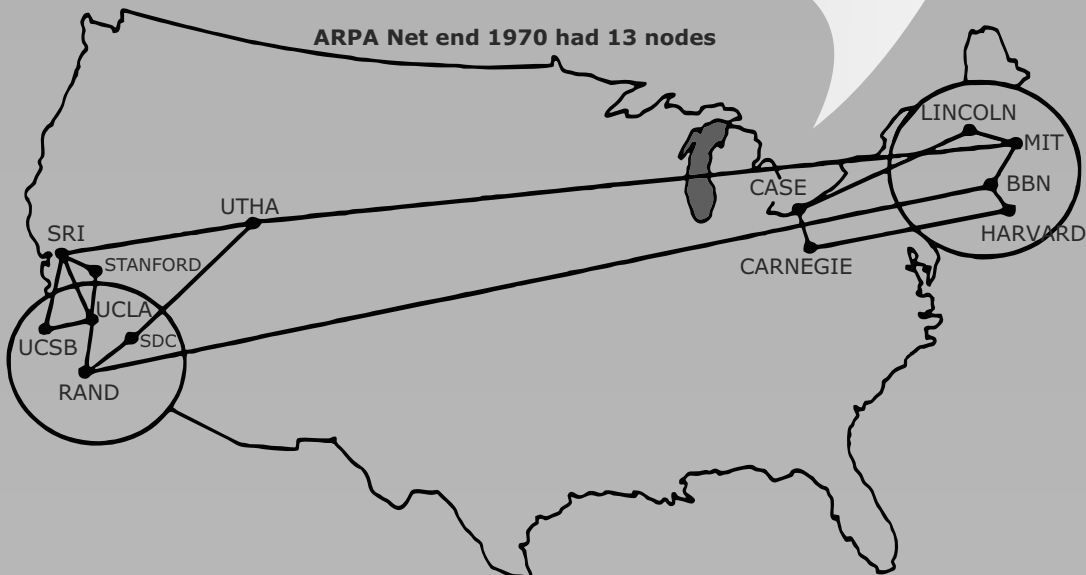
The reason for ARPANET's creation was economic: computers were expensive to buy, so it was beneficial if laboratories working on projects with a military purpose could share each other's equipment, even though spread out.

Because in the late 1960s neither equipment nor communication lines were very reliable, it was a requirement for ARPANET that in the event of inevitable local technical problems as many operations could continue as possible.

It is a popular misconception that the ARPANET was intended as a communications network in times of nuclear war. There was the idea of a network that could survive a nuclear war (suggested by Paul Baran, who proposed a similar packet switching architecture) but this suggestion did not play a role in ARPA's decision to set up a network.

The first concrete plans for the ARPANET were drafted in 1968, and in 1969 ARPA decided to outsource the implementation to the firm Bolt, Beranek and Newman. A major problem in the designing ARPANET was that computers of very different sorts needed to be inter-connected.

The designers of ARPANET solved this potential incompatibility by adding a minicomputer at each network location to work as an Interface Message Processor (IMP). This IMP interfaced the computer and the network connect. These IMPs were 16-bit DDC-516 computers from Honeywell.





QUANTUM ENTANGLEMENT

→ see also Blaise Nr 62 page 10 explains this in great detail

Quantum entanglement is a physical phenomenon that occurs when a group of particles are generated, interact, or share spatial proximity in a way such that the quantum state of each particle of the group cannot be described independently of the state of the others, including when the particles are separated by a large distance.

The topic of Quantum Entanglement is at the heart of the disparity between classical and quantum physics: entanglement is a primary feature of quantum mechanics lacking in classical mechanics.

Measurements of physical properties such as position, momentum, spin, and polarization performed on entangled particles can, in some cases, be found to be perfectly correlated. For example, if a pair of entangled particles is generated such that their total spin is known to be zero, and one particle is found to have clockwise spin on a first axis, then the spin of the other particle, measured on the same axis, is found to be counterclockwise. However, this behavior gives rise to seemingly paradoxical effects: any measurement of a particle's properties results in an irreversible wave function collapse of that particle and changes the original quantum state. With entangled particles, such measurements affect the entangled system as a whole.

E



QUANTUM MEMORY

WIKIPEDIA

In Quantum Computing, Quantum Memory is the quantum-mechanical version of ordinary computer memory. Whereas ordinary memory stores information as binary states (represented by "1"s and "0"s), Quantum Memory stores a quantum state for later retrieval.

These states hold useful computational information known as QuBits.

*Unlike the classical memory of everyday computers, the states stored in Quantum Memory can be in a Quantum Superposition, **giving much more practical flexibility in quantum algorithms than classical information storage.***

Quantum memory is essential for the development of many devices in Quantum Information processing, including a synchronization tool that can match the various processes in a Quantum Computer, a Quantum Gate that maintains the identity of any state, and a mechanism for converting predetermined photons into on-demand photons.

*Quantum Memory can be used in many aspects, such as Quantum Computing and Quantum Communication. Continuous research and experiments have **enabled Quantum Memory to realize the storage of QuBits.***

M

Quantum Memory can be used in many aspects, such as Quantum Computing and Quantum Communication.



THE NITROGEN-VACANCY CENTER (N-V center or NV center)

WIKIPEDIA

is one of numerous point defects in diamond. Its most explored and useful property is its photoluminescence, which allows observers to read out its spin-state.

*The NV center's electron spin, localized at atomic scales, **can be manipulated at room temperature by external factors such as magnetic, or electric fields, microwave radiation, or light, resulting in sharp resonances in the intensity of the photoluminescence.***

*These resonances can be explained in terms of electron spin related phenomena such as quantum entanglement, spin-orbit interaction and Rabi oscillations, and analysed using advanced quantum optics theory. An individual **NV center** can be used as a basic unit for a quantum computer, a qubit, used f.e. for quantum cryptography. Further potential applications in novel fields of electronics and sensing include spintronics, masers, and quantum sensors. If the charge is not specified the term "NV center" refers to the negatively charged NV- center.*

N





QUANTUM KEY DISTRIBUTION (QKD)

is a secure communication method which implements a cryptographic protocol involving components of quantum mechanics. It enables two parties to produce a shared random secret key known only to them, which can then be used to encrypt and decrypt messages. It is often incorrectly called quantum cryptography, as it is the best-known example of a quantum cryptographic task. An important and unique property of quantum key distribution is the ability of the two communicating users to detect the presence of any third party trying to gain knowledge of the key. This results from a fundamental aspect of quantum mechanics: the process of measuring a quantum system in general disturbs the system.

A third party trying to eavesdrop on the key must in some way measure it, thus introducing detectable anomalies. By using quantum superpositions or quantum entanglement and transmitting information in quantum states, a communication system can be implemented that detects eavesdropping. If the level of eavesdropping is below a certain threshold, a key can be produced that is guaranteed to be secure (i.e., the eavesdropper has no information about it), otherwise no secure key is possible and communication is aborted.



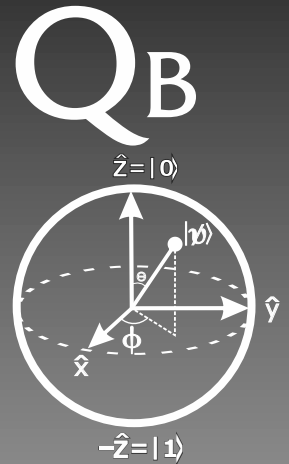
QUBITS → see also BlaiseNr 62 page 9 explains this in great detail

In Quantum Computing, a QuBit or Quantum Bit (sometimes qbit) is the basic unit of quantum information - the quantum version of the classic binary bit physically realized with a two-state device.

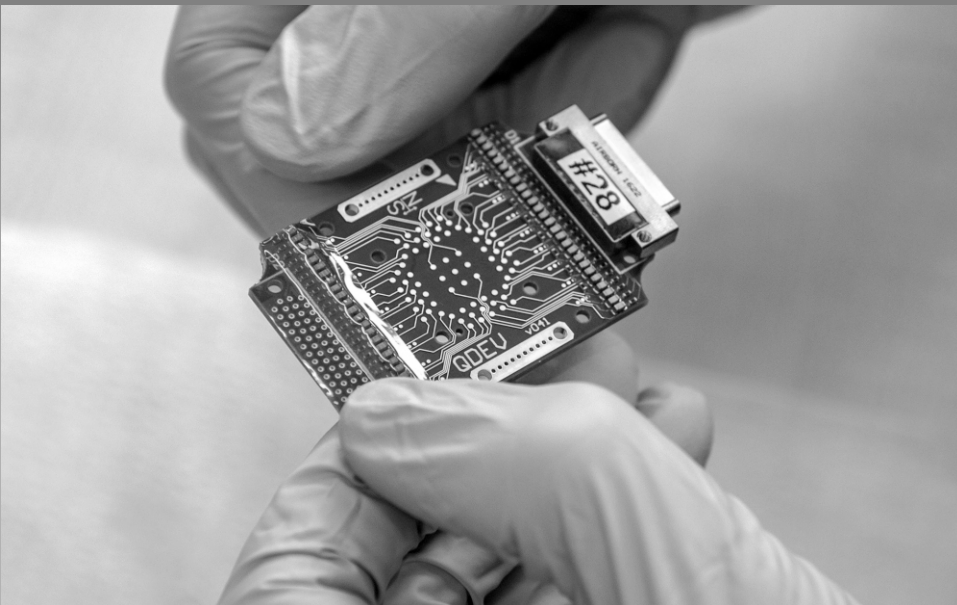
A qubit is a two-state (or two-level) quantum-mechanical system, one of the simplest quantum systems displaying the peculiarity of quantum mechanics. Examples include the spin of the electron in which the two levels can be taken as spin up and spin down; or the polarization of a single photon in which the two states can be taken to be the vertical polarization and the horizontal polarization.

In a classical system, a bit would have to be in one state or the other. However, quantum mechanics allows the qubit to be in a coherent superposition of both states simultaneously, a property that is fundamental to quantum mechanics and quantum computing.

Qkd



The Bloch sphere is a representation of a qubit, the fundamental building block of quantum computers.



Developing a topological qubit
<https://cloudblogs.microsoft.com/quantum/2018/09/06/developing-a-topological-qubit/>





QUANTUM REPEATER

Repeaters

Long-distance communication is hindered by the effects of signal loss and decoherence inherent to most transport mediums such as optical fiber. In classical communication, amplifiers can be used to boost the signal during transmission, but in a quantum network amplifiers cannot be used since qubits cannot be copied – known as the no-cloning theorem.

That is, to implement an amplifier, the complete state of the flying qubit would need to be determined, something which is both unwanted and impossible.

R

TRUSTED REPEATERS

*An intermediary step which allows the testing of communication infrastructure are **trusted repeaters**. Importantly, a trusted repeater cannot be used to transmit qubits over long distances.*

Instead, a trusted repeater can only be used to perform quantum key distribution with the additional assumption that the repeater is trusted.

Consider two end nodes A and B, and a trusted repeater R in the middle. A and R now perform quantum key distribution to generate a key. The key is secure from an outside eavesdropper, but clearly the repeater R also knows. This means that any subsequent communication between A and B does not provide end to end security, but is only secure as long as A and B trust the repeater R.

QUANTUM REPEATERS

Diagram for quantum teleportation of a photon

A true quantum repeater allows the end to end generation of quantum entanglement, and thus - by using quantum teleportation - the end to end transmission of qubits.

In quantum key distribution protocols one can test for such entanglement.

This means that when making encryption keys, the sender and receiver are secure even if they do not trust the quantum repeater.

Any other application of a quantum internet also requires the end to end transmission of qubits, and thus a quantum repeater.

Quantum repeaters allow entanglement and can be established at distant nodes without physically sending an entangled qubit the entire distance.

These initial entangled qubits can be easily created, for example through parametric down conversion, with one qubit physically transmitted to an adjacent node.

*At this point, the repeater can perform a **bell measurement** on the qubits.*

This has the effect of "swapping" the entanglement such that they are now entangled at a distance twice that of the initial entangled pairs.

It can be seen that a network of such repeaters can be used linearly or in a hierarchical fashion to establish entanglement over great distances.

*The **BELL MEASUREMENT** is an important concept in quantum information science: It is a joint **quantum-mechanical measurement** of two qubits that determines which of the four Bell states the two qubits are in.*

*Hardware platforms suitable as end nodes can also function as **Quantum Repeaters**. However, there are also hardware platforms specific only to the task of acting as a repeater, **without the capabilities** of performing quantum gates*

*A **QUANTUM LOGIC GATE** (or simply quantum gate) is a basic quantum circuit operating on a small number of qubits. They are the building blocks of quantum circuits, like classical logic gates are for conventional digital circuit.*





QUANTUM TELEPORTATION

Quantum Teleportation is a technique for transferring quantum information from a sender at one location to a receiver some distance away. While teleportation is commonly portrayed in science fiction as a means to transfer physical objects from one location to the next, quantum teleportation only transfers quantum information. Moreover, the sender may not know the location of the recipient, and does not know which particular quantum state will be transferred.

One of the first scientific articles to **investigate quantum teleportation** is "Teleporting an Unknown Quantum State via Dual Classical and Einstein-Podolsky-Rosen Channels" in 1993, in which they used dual communication methods to send/receive quantum information. It was experimentally realized in 1997

Experimental determinations of quantum teleportation have been made in information content - including photons, atoms, electrons, and superconducting circuits - as well as distance with 1,400 km (870 mi) being the longest distance of successful teleportation by the group of Jian-Wei Pan using the Micius satellite for space-based quantum teleportation.

Challenges faced in quantum teleportation include the no-cloning theorem which sets the limitation that **creating an exact copy of a quantum state is impossible**, the no-deleting theorem that states that quantum information cannot be destroyed, the size of the information teleported, the amount of quantum information the sender or receiver has before teleportation, and noise that the teleportation system has within its circuitry.

T
creating
an exact
copy
of a
quantum state
is impossible



Quantum teleportation shows up in 3D for the first time
<http://chinaplus.cri.cn/news/china/9/20190819/335607.html>



THE RACE HAS BEGUN

to create a super-secure online space that channels the unimaginable power of the quantum world. We've uploaded a lot in our lives related to banking, emails, social media, profiles, records - a lot of sensitive information. It's a worrying fact that the internet regularly shows major security flaws. It's not insurmountable yet, but our private data is usually fairly secure.

But today or tomorrow we may all be faced with cracked encryption algorithms that should have protected us.

The race has begun to create a super-secure online space that channels the unimaginable power of the quantum world. We've uploaded a lot in our lives related to banking, emails, social media, profiles, records - a lot of sensitive information. It's a worrying fact that the internet regularly shows major security flaws. It's not insurmountable yet, but our private data is usually fairly secure. But today or tomorrow we may all be faced with cracked encryption algorithms that should have protected us.

That is the urgent task for a new, safer kind of internet. I Always asked myself why the construction of addressing is so easy to circumvent: if you connect a real address to the internet or a personal number that is unique and belongs to that person, the person can be traced, the problem would solve itself. The free internet has changed from an free to a dangerous environment.

AND YET WE CANNOT LET IT GO.

The Quantum Internet uses the power of the Quantum Environment. When implemented, these systems will provide much better protection than just our data. We will bring all-new, unimaginable Quantum apps and perhaps become the platform for a global Quantum computer of unimaginable power. This will of course be an enormous and all-encompassing technical challenge, but the start is already there! And it is expanding and improving every day. Anyone who still remembers the „ARPA NET“ can make a comparison.

We are in that phase now. Cable networks are being laid everywhere. Now whole-world satellite belts are emerging specifically to support the Internet, and scientists can chat their secrets through local networks. There are even developments that also use small satellites to enable quantum connections at great distances. The time has come for us to join the QUANTUM INFORMATION HIGHWAY.

Our culture and industries are of course based on a lot, a large quantity of information and therefore knowledge. If we use this to get the right kind of information and make it accessible - and share it - we can gain a lot of power and therefore executive power and thus a lot of social gain.



2. THE QUANTUM THEORY

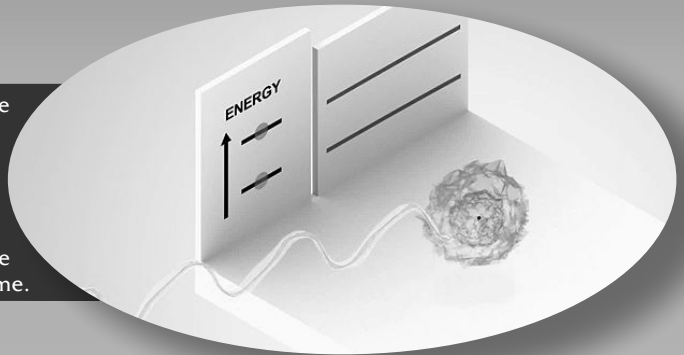
This part has been taken out at wiki:
https://en.wikipedia.org/wiki/Quantum_mechanics

Quantum mechanics allows the calculation of properties and behaviour of physical systems. It is typically applied to microscopic systems: molecules, atoms and sub-atomic particles. It has been demonstrated to hold for complex molecules with thousands of atoms, but its application to human beings raises philosophical problems. Predictions of quantum mechanics have been verified experimentally to an extremely high degree of accuracy.

A FUNDAMENTAL FEATURE of the theory is that it usually **cannot predict with certainty** what will happen, but only give probabilities.

One consequence of the mathematical rules of quantum mechanics is a tradeoff in predictability between different measurable quantities. The most famous form of this uncertainty principle says that no matter how a quantum particle is prepared or how carefully experiments upon it are arranged, it is impossible to have a precise prediction for a measurement of its position and also at the same time for a measurement of its momentum.

The green line shows the “wave” which means the superposition changing through time. The two red circles show the superpositions of the Qubit: the outer ring is active state, the inner is the inactive state. They can be active and inactive at the same time.



Another consequence of the mathematical rules of quantum mechanics is the phenomenon of quantum interference, which is often illustrated with the double-slit experiment. *(Read BPM issue 62 page 9)* In the basic version of this experiment, a coherent light source, such as a laser beam, illuminates a plate pierced by two parallel slits, and the light passing through the slits is observed on a screen behind the plate.

The wave nature of light causes the light waves passing through the two slits to interfere, producing bright and dark bands on the screen – a result that would not be expected if light consisted of classical particles. However, the light is always found to be absorbed at the screen at discrete points, as individual particles rather than waves; the interference pattern appears via the varying density of these particle hits on the screen.



2. THE QUANTUM THEORY

Another counter-intuitive phenomenon predicted by quantum mechanics is quantum tunnelling:

a particle that goes up against a potential barrier can cross it, even if its kinetic energy is smaller than the maximum of the potential. In classical mechanics this particle would be trapped.

Quantum tunnelling has several important consequences, enabling radioactive decay, nuclear fusion in stars, and applications such as scanning tunnelling microscopy and the tunnel diode,

allow more exact predictions than quantum theory can provide.

A collection of results have demonstrated that broad classes of such hidden-variable theories are in fact incompatible with quantum physics.

**entanglement does not allow
sending signals faster than light,
as demonstrated by the
no-communication theorem.**

When quantum systems interact, the result can be the creation of QUANTUM ENTANGLEMENT:

their properties become so intertwined that a description of the whole solely in terms of the individual parts is no longer possible.

QUANTUM ENTANGLEMENT (*See page 3 of this article*) enables the counter-intuitive properties of quantum pseudo-telepathy, and can be a valuable resource in communication protocols, such as quantum key distribution and superdense coding. Contrary to popular misconception, entanglement does not allow sending signals faster than light, as demonstrated by the no-communication theorem.

Another possibility opened by entanglement is testing for "hidden variables", hypothetical properties more fundamental than the quantities addressed in quantum theory itself, knowledge of which would allow more exact predictions than quantum theory can provide.



2. THE QUANTUM THEORY

RELATION TO GENERAL RELATIVITY

Even though the predictions of both quantum theory and general relativity have been supported by rigorous and repeated empirical evidence, their abstract formalisms contradict each other and they have proven extremely difficult to incorporate into one consistent, cohesive model. Gravity is negligible in many areas of particle physics, so that unification between general relativity and quantum mechanics is not an urgent issue in those particular applications. However, the lack of a correct theory of quantum gravity is an important issue in physical cosmology and the search by physicists for an elegant "Theory of Everything" (TOE). Consequently, resolving the inconsistencies between both theories has been a major goal of 20th- and 21st-century physics. This TOE would combine not only the models of subatomic physics but also derive the four fundamental forces of nature from a single force or phenomenon.

One proposal for doing so is string theory, which posits that the point-like particles of particle physics are replaced by one-dimensional objects called strings. String theory describes how these strings propagate through space and interact with each other. On distance scales larger than the string scale, a string looks just like an ordinary particle, with its mass, charge, and other properties determined by the vibrational state of the string. In string theory, one of the many vibrational states of the string corresponds to the graviton, a quantum mechanical particle that carries gravitational force.

Another popular theory is **loop quantum gravity** (LQG), which describes quantum properties of gravity and is thus a theory of quantum spacetime. LQG is an attempt to merge and adapt standard quantum mechanics and standard general relativity. This theory describes space as an extremely fine fabric "woven" of finite loops called spin networks. The evolution of a spin network over time is called a spin foam.

3. THE QUANTUM INTERNET

The rise of the internet has made the role of information clearer and we are only now beginning to feel its far-reaching effects. Now we are again at the dawn of a new information age, that is likely to turn the world upside down: We used to need industry to produce large quantities of products, now we have the possibility via computers, supercomputers and quantum computers to create highly individualized industrial scale products: buy a suit and you can have it custom made via a computer that makes it to your special size. If you are ill, individualized medicines can even be created for you.

Normal (conventional) computers function with digital units called Bits. This is the information usually displayed as a value of 1 or 0. Every email, status update or photo on your phone is stored entirely in bits.



3. THE QUANTUM INTERNET

The task that is slowly emerging has already yielded impressive new technologies, such as ultra-sensitive detectors LHC (Large Hadron Collider) at Cern, of gravity and magnetic fields. Physicists can now control dozens of interconnected qubits at once. Now they can make Prototypes of real QUANTUM COMPUTERS if they get enough capacity.

It looks like they will outperform any computer that could ever be built, at least in special types of calculations.

Quantum computers can simulate chemical processes to design new drugs and advanced materials and solve difficult problems in engineering and logistics.

Their full potential is unimaginable, all because they are able to perform all calculations simultaneously via an array of Qubits. A "normal computer" wouldn't get around to that in years.

One thing we can predict with certainty is that these incredible machines will make us arrange a QUANTUM INTERNET because it is Quantum Computers that threaten our security.

Our current encryption algorithms are based on mathematical solutions that are impossible for a classical computer to calculate on: factoring large prime numbers takes an awful lot of time. That way we can't keep the internet safe enough in the long run.

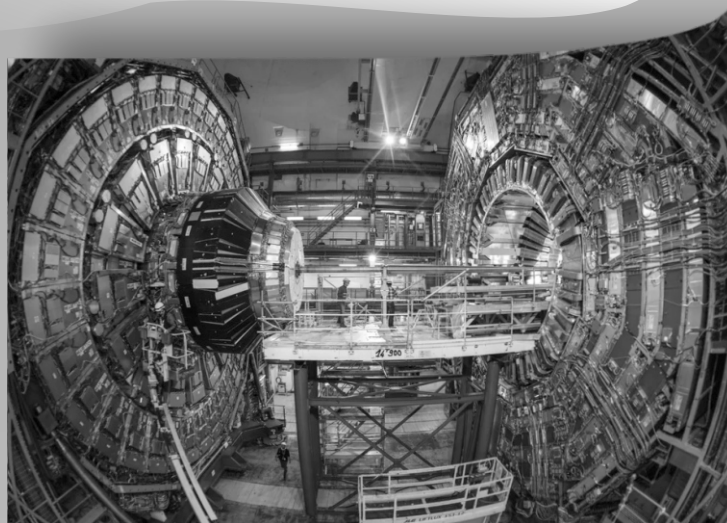
A Quantum Computer could do this in a flash, using an algorithm designed way back in 1994 by mathematician Peter Shor of MIT (Massachusetts Institute of Technology).

(See the article of Issue 63 page 11/12 "Basic insights")

QUANTUM ENCRYPTION

From the perspective of the quantum world, that's rather limited because we know that particles behave in ways that are very strange. An atom, electron or photon can be in a state where its properties are not determined. For instance: it can have two different energies at once. These quantum states are extremely fragile, but we learn to manipulate them and deal with particles that store a quantum unit or information.

A QUBIT encoding knows not only 0 or 1, but any combination of 0 and 1 together. (The equivalent of "maybe" for humans, but computers can't use that.)



Cern



3. THE QUANTUM INTERNET

QUANTUM ENCRYPTION

Security then becomes impossible for everything that depends on online communication, from e-mail to water management systems and electricity networks and all the IOT (Internet of things). Our vital infrastructure is still based on this, especially the financial world.

I predict that such an extremely powerful QUANTUM MACHINE will become possible in 5 years. But this problem needs to be solved. Modifying crypto systems takes a lot of time: so computer time and data being sent now could be intercepted and unravelled if a powerful Quantum Computer is available then.

The consequences are vital: even politics would be shown for the umpteenth time that one lies. You can immediately see the political vicissitudes and, if possible, a threatening lead would affect world peace. There are still leaders who think if only they have an advantage, they can take on world domination. As old as mankind is, did that not lead to prosperity...

Conversely, if you're communicating using QuBits' quantum states, you can't eavesdrop on them, and if you do, it won't work, because just looking at the signal will change that delicate state.

This will not mean replacing the entire Internet, **but building an extra layer of quantum communication links on top** so that users can share a key that keeps their online exchanges secret. Internet traffic would still go through the cables that do it now. It would only be encrypted and extracted using those keys.

Connections with large numbers of entangled qubits can enable even more impressive applications, such as sending messages expressed entirely in quantum states. In the short term, quantum computers will not be as powerful and will probably be located at a greater distance, for example in universities or research centers. But using Quantum Communication links, we could create a quantum supercomputer by making the Quantum Computers collaborate.

They could also allow users to remotely run programs on quantum computers, ensuring security that even the owners of the computer cannot snoop on. This is called blind quantum computing and it could allow anyone to use Quantum Computers without any risk of sensitive data being stolen.

DIAMOND COMMUNICATION

A fantastic example of the emerging QUANTUM INTERNET has been shown in a laboratory in Delft, at the Technical University of Delft, Netherlands. There, three very small diamonds communicate with each other, forming a miniature but fully functioning prototype network of entwined links. In the lattice of carbon atoms of each diamond is a defect containing a single nitrogen atom. An electron pair in this spot can emit a photon entangled with it. Each diamond also contains a one-qubit quantum memory, which enables basic quantum information processing.

In an article published in April, Ronald Hanson and his team from QuTech (a research institute in Delft affiliated with the Technical University of Delft, where Stephanie Wehner made her prediction three years ago that a Quantum Internet would soon come) showed that they could network three diamonds and exchange quantum information between them. In principle, this technology can be scaled up, allowing entanglement to be shared between any number of nodes. To cut costs: The hardware doesn't have to be a diamond.



4 THE QUANTUM INTERNET EXPLORER

QUTECH INSTITUTE, Netherlands

Quantum error correction is essential for large-scale quantum systems that are prone to decoherence: interaction of the quantum system with the surrounding environment leads to a loss of quantum information, which hinders the development of scalable universal quantum computing. Correcting errors in the qubits poses special challenges:

- ① First
a quantum bit is not only subject to bit flip errors (a 0 turning into a 1 or the other way around), but can suffer gradual errors.
- ② Second,
according to the no-cloning theorem, it is impossible to make perfect copies of qubit state.
- ③ Lastly,
measuring a qubit in an attempt to discover errors immediately destroys the qubit state.

In a quantum ERROR CORRECTING CODE, quantum information is distributed among many qubits, so that the dominant noise processes affect this information in a reversible manner. This means that there exists an error reversal procedure by which one can detect and correct the errors. By encoding a logical quantum bit in multiple physical qubits it becomes possible to detect and correct errors so that the quantum system becomes more stable with increasing size.

FUNDAMENTALLY NEW INTERNET TECHNOLOGY

The internet – a vast network connecting devices anywhere on earth using long-range classical communication – has had a revolutionary impact on our world. It provided us with new ways of sharing and accessing information from all around the world. The long-term vision of a quantum internet is to provide a fundamentally new internet technology by enabling the transmission of quantum bits. Such a quantum internet will – in synergy with the ‘classical’ internet that we have today – connect quantum processors in order to provide unparalleled capabilities that are impossible using classical communication. One aim of the Quantum Internet Demonstrator is to connect several cities in the Area Delft, The Hague and Leiden with a quantum network and showcase the operational capabilities of a quantum internet to the world.

A second aim is to lay a basis for national Quantum-secure network, including access for the testing of relevant innovation questions and industrial applications. We will develop and integrate new hardware and software components for quantum networks, and test innovations in a concept phase. In addition, the **Quantum Internet Demonstrator** provides an ideal starting point for growing a **EUROPEAN QUANTUM NETWORK**.

One of the reasons that the classical internet was able to grow so quickly was that, from the outset, access for network engineers, programmers and users was straightforward and cheap. The creation of a similar environment can give the Quantum Internet the boost needed to take it to the next level.

Indeed, we are already working together with several partners

(see here

<https://qutech.nl/putting-quantum-bits-into-the-fiber-optic-network-launching-the-qfc-4-1qid-project/> and here

<https://qutech.nl/kpn-and-qutech-join-forces-to-make-quantum-internet-a-reality/>) to realize the first quantum internet demonstrator.



4 THE QUANTUM INTERNET EXPLORER



From left to right:

Stephanie Wehner (QuTech), Jaya Baloo (KPN), Jan Kees de Jager (KPN), Paul de Krom (TNO), Paul Althuis (TU Delft), Ronald Hanson (QuTech)

Currently QuTech is working on two complementary network technologies within the DEMONSTRATOR PROGRAM:

The explorer will become available after the summer. We will tell you!

- ◆ The worlds first QUANTUM INTERNET based on linked qubit systems based on Nitrogen-Vacancies (NV)*¹ centres in diamond. *¹(See explantion on page 3 of this article)
- ◆ A MDI QKD (Measurement-Device-Independent \ Quantum Key distribution*²) network that will enable end to end security to all clients connected to a central point, and is fundamentally more secure than point-to-point QKD systems due to its quantum measurement in the central point.*²(See explantion on page 3 of this article)

With the QUANTUM NETWORK EXPLORER, planned to be released by the end of 2021, everybody will be able to gain online access to our quantum network, and learn about its applications and capabilities.

Connecting millions of users around the world with super-secure encryption keys, across countries and continents is an ultimate goal. WE WILL HAVE TO.

Working together with the existing network of optical cables that carry all current internet traffic and other telecom data is the basis. However, there are technical problems: optical fibers are completely transparent. Even if you use the optimal wavelength of light. After a short distance, 90 percent of the photons disappear. That limits Quantum-By-Fibre to a range of a few hundred kilometers at most.

Even if you use the successors to IBM's quantum computer, this security of the Internet can be broken. So there has to be a much better solution. And it is already there: The principle of QUANTUM MEMORY (see page 3 of this article).

Today's fiber optic network uses amplifiers to fortify signals. It is impossible to send Quantum Signals through an amplifier. Amplifiers measure the signal and that directly changes the quantum data!

This is not the case with QUANTUM MEMORY: they do not change anything, but they check the whole via a calculation method. (See the explanation at page 4 of this article). By extending the available time of the presence of a QuBit, a larger cluster can be prepared and therefore more capacity.



4 THE QUANTUM INTERNET EXPLORER

QUANTUM REPEATER

To extend the reach of QKD (Quantum Key Distribution - see page 4 of this article) you need to build on trusted nodes, devices that forward a message by decoding and recoding it to send it to the next part of the fiber. China already has an impressive network. with a 500-mile series of 32 trusted nodes totaling hundreds of links. Problem solved? No. Each node possesses a security risk that, if compromised, could leak your message. This is not an optimal improvement.

In order to transport our Quantum data at all, we need a device known as a QUANTUM REPEATER. (See page 5 of this article). Imagine two users named A and B who want to chat.

They each make a pair of entangled QUBITS and each sends one of their own to a Quantum Repeater in the middle. The Repeater performs some kind of simultaneous measurement of the states of the two qubits it has received, designed to entangle them. (See explanation on page 8) According to the rules of quantum physics, this then entangles the two QuBits held by A and B, a process called entanglement swapping. String many QUANTUM REPEATERS together in a line and you can get entangled qubits at a much greater distance and now we will talk about something very special:

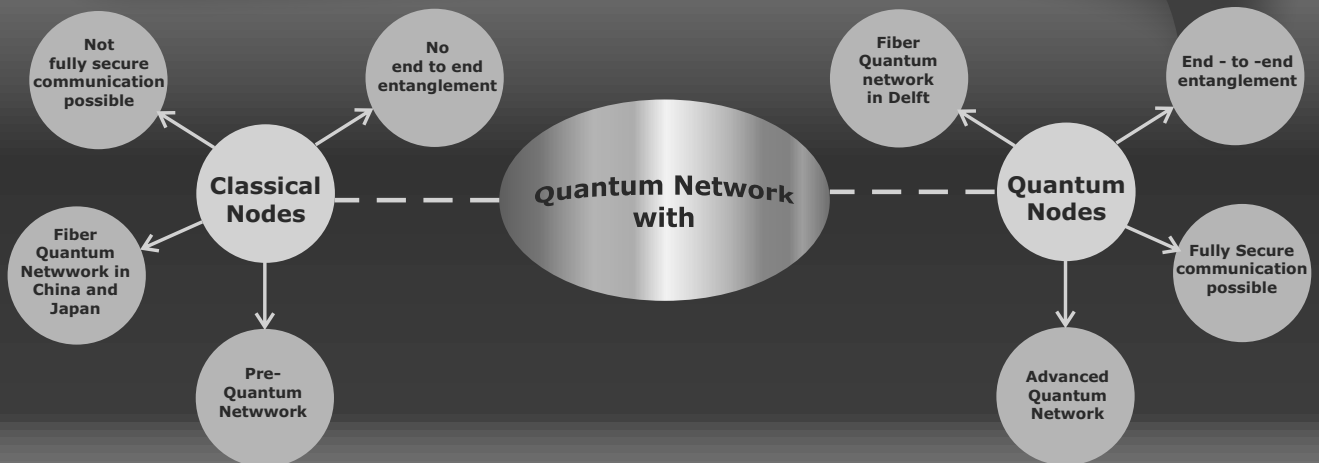
REFRESHING THE SIGNAL WITH CLASSICAL NODES

In a pre-quantum network, such as the ones in China and Japan, this is the job of classical nodes. At the node, the photons that arrive are collected and their state is measured. Afterwards, fresh photons in the right state are sent towards the next node in the chain.

QUANTUM REPEATERS AND TELEPORTATION

In space, photons can travel for hundreds and hundreds of kilometers with a small probability of getting lost. **This is a luxury that doesn't apply to fiber.** Imagine that you have a photon source that releases 10 billion photons every second.

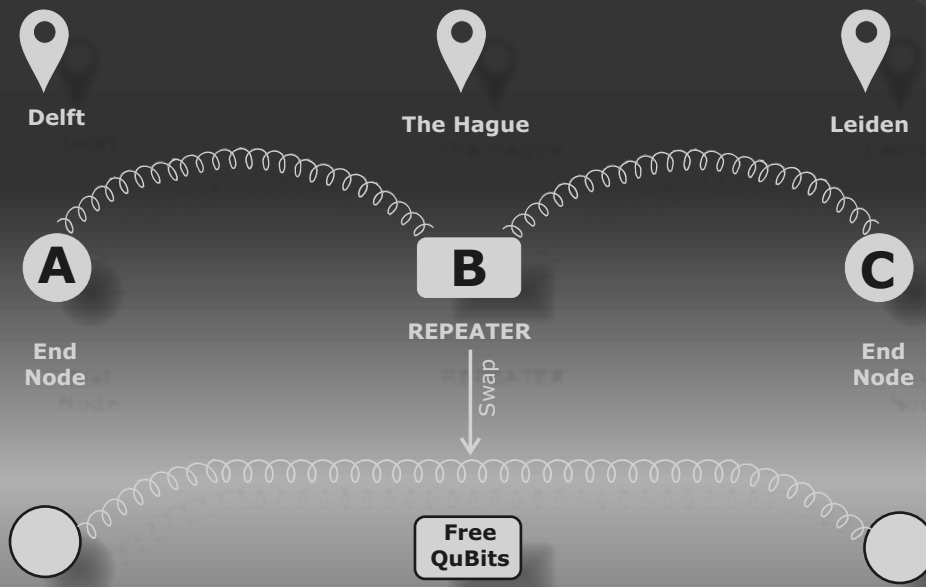
With a fiber connection in which a bit less than one out of twenty photons gets lost every kilometer, after 500 kilometers the rate has dropped to roughly one photon arriving every second. After 1000 kilometers of fiber there is such an immense drop in the rate that you might not even be able to measure any photon in your own lifetime anymore: after 1000 kilometers of fiber around one photon will arrive every 300 years. Therefore, the signal needs to be refreshed.



4 THE QUANTUM INTERNET EXPLORER

THE NEXT STAGE: QUANTUM REPEATERS

In stages beyond the pre-quantum network, the aim is to refresh the signal with a different kind of node: a quantum repeater. True quantum repeaters have not been realized yet, but in the near future the basis for such repeaters will be laid on the Dutch Quantum Network. In order to understand how QUANTUM REPEATERS work, it's important to understand a different concept first: QUANTUM TELEPORTATION.



STAR TREK

If you've watched Star Trek, or any other science fiction film where teleportation is commonplace, you probably already have an idea of what teleportation should look like: an object disappears and reappears somewhere else. But while teleportation in science fiction films often ignores physical laws, QUANTUM TELEPORTATION works just because of that. Performing QUANTUM TELEPORTATION means that a quantum state disappears on one side and then reappears again at the other side. The quantum state doesn't physically travel from one side to the other, but it is teleported. Also note that the state gets destroyed at the original location.

TELEPORTATION OF INFORMATION QUBITS

Suppose that we have a Quantum Network between DELFT and THE HAGUE. The qubits in DELFT and THE HAGUE, qubit D and qubit H, are entangled. In Delft there is another qubit, the information qubit, that we want to teleport over the network to THE HAGUE. First, the entanglement between qubits D and H needs to be stored in a quantum memory. Afterwards, qubit D and the information qubit can get entangled. In DELFT, we perform certain measurements on the qubit D and the information qubit and afterwards we communicate our measurement outcomes to THE HAGUE.

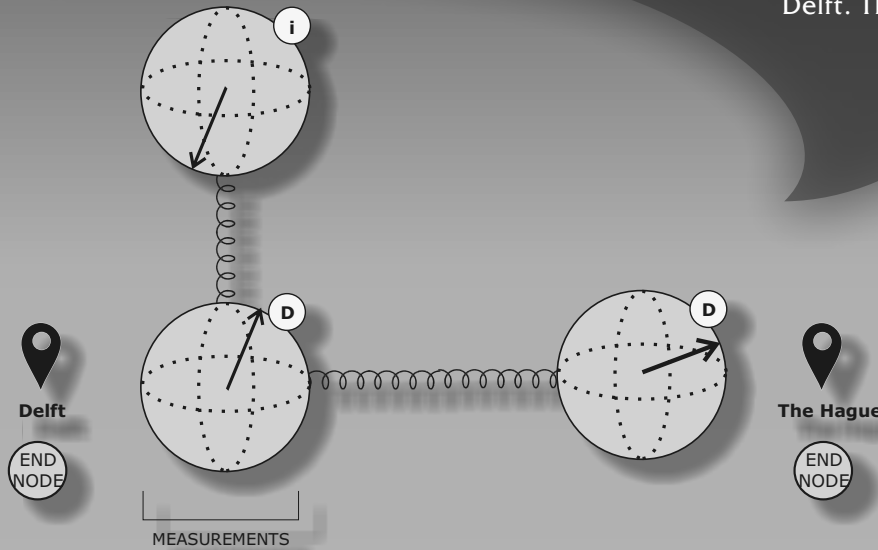
NOTE: this communication with THE HAGUE is classical, not quantum. After receiving the classical communication, the node at THE HAGUE performs certain operations to compensate for the measurement outcome which lead to a QUBIT having the same state as the information QUBIT originally had. In other words: the information QUBIT has been teleported.



4 THE QUANTUM INTERNET EXPLORER

QUANTUM REPEATERS use this concept of QUANTUM TELEPORTATION.

Imagine that Delft and Leiden are part of a **Quantum Network** and they are connected via a **Quantum Repeater** in The Hague. First, entanglement can be generated between the end node in Delft and the quantum repeater in The Hague. Also, entanglement can be generated between the quantum repeater in The Hague and the end node in Leiden. Once both links succeed in generating entanglement, the quantum repeater can employ the following trick: it can use the qubit that is entangled with Leiden to **teleport** one qubit from The Hague to Leiden. In particular, it can teleport the qubit that is entangled with Delft. Hence, after performing the teleportation protocol Leiden obtains a qubit which is entangled with Delft. This procedure is known



China is working on a quantum communication network in space.

In space, there is negligible photon loss and photons can remain in a superposition state for a long time, meaning that no repeaters are needed.

Recently, a trusted repeater network was established between a satellite and three ground locations, two in China and one in Austria, Quantum Key Distribution (see page 4/18 of this article) has already been performed. In 2017, a secure videoconference was held between scientists in Austria and China with this network. Since the rate at which keys were generated did not allow for direct encryption, the keys were used to feed a symmetric cypher (ADVANCED ENCRYPTION STANDARD PROTOCOL), resulting in the secure videoconference lasting for 75 minutes.

Furthermore, the Chinese team has also demonstrated quantum teleportation from the ground to the satellite. Since fiber networks have the advantage of efficiently and conveniently connecting several users inside a city, the satellite has already been connected to the Chinese pre-quantum network.

A Quantum Network in space has two main drawbacks. The first is a low transmission rate since out of every 6 million photons sent from the satellite every second, only about one photon was detected on the ground.

The second drawback is that today, a quantum network in space can only operate at night. Therefore, China's National Space Science Center is expected to launch other satellites that have stronger and cleaner beams such that it can also operate during the day.



5. THE SOCIETAL AND SOCIAL CONSEQUENCES

If you have a lot (I mean really many) ground stations, a few large satellites cannot serve them. Instead we need extensive coverage.

The **QUANTUM INTERNET KIT** at **QuTech** in the Netherlands is one of the most advanced constellations of small satellites to date.

Several projects are making their way, including a UK-Singapore mission called **SPEQTR***, and **ROKS***, a satellite built by a private consortium. Both will be launched in 2022.

SPEQTR:

<https://www.isispace.nl/news/isispace-selected-to-develop-12-unit-cubesat-platform-for-emerging-quantum-key-distribution-mission/>

ROKS:

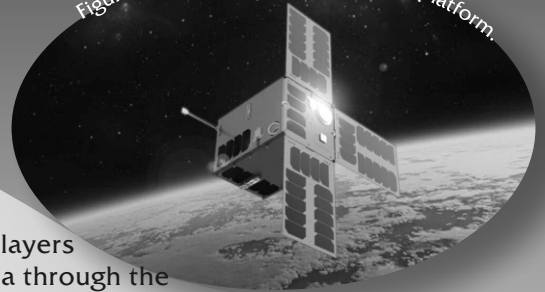
<https://www.nanosats.eu/sat/roks>

and

<https://craftprospect.com/portfolioitems/roks/>

#::~text=The%20Responsive%20Operations%20Key%20Services,Key%20Distribution%20(QKD)%20technology.

Figure 1 – Render image of the 12U platform.



To create a **Global Quantum Web**, we need the kind of software that makes it possible to use apps already available using the normal internet. Several layers of software, known as the Internet Stack, route data through the existing network so that the average user doesn't have to worry about plumbing. How about working with the actual apps? We have no clue. Will we have the chance to create new forms of communication?

3D projections in the middle of the room? Not to talk a bout the gaming industries.

We are still Homo Ludens! Teleportation of goods will not be available for the coming year, but maybe we could even better understand the Physics then.

When these extraordinary technologies have come to the world, we will become more and more aware of them. They will alter our lives and maybe even our social structures.

Many actual problems will be solved: danger of kidnapping your bank account, creating better and securer environments where elections won't be hacked, the lights won't disappear. Of course science will be better off: Space flight to our neighbor planet and even stars will come to reality and why not Warp Drive?

We can develop in Pascal what we need in future...Pascal has it all. Or else we will create it.

There is always a downside to all the good news, the new Quantum Web makes bigger Quantum Computation possible. The Dark web should be made impossible. This new internet will make it possible to deal in much better way of creating a more social world, where we all are equal. In reality.

For any extra explanations see or previous issues about this subject

Issue: 62/63/66/73&74

This article was created by making use of the following sources:

New Scientist: <https://www.newscientist.com/>

QuTech: <https://qutech.nl/>

SciTechDaily: <https://scitechdaily.com/impenetrable-encryption-for-data-communication-researchers-take-quantum-key-distribution-out-of-the-lab/>

Tech Target: <https://searchsecurity.techtarget.com/definition/quantum-key-distribution-QKD>

WikiPedia: https://en.wikipedia.org/wiki/Quantum_key_distribution

IDQ: <https://www.idquantique.com/quantum-safe-security/overview/>

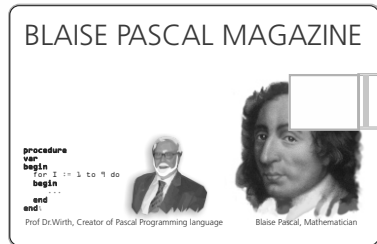
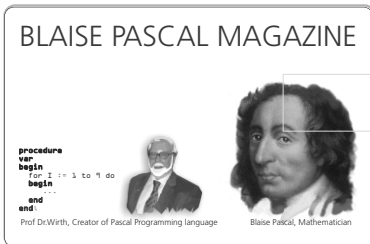
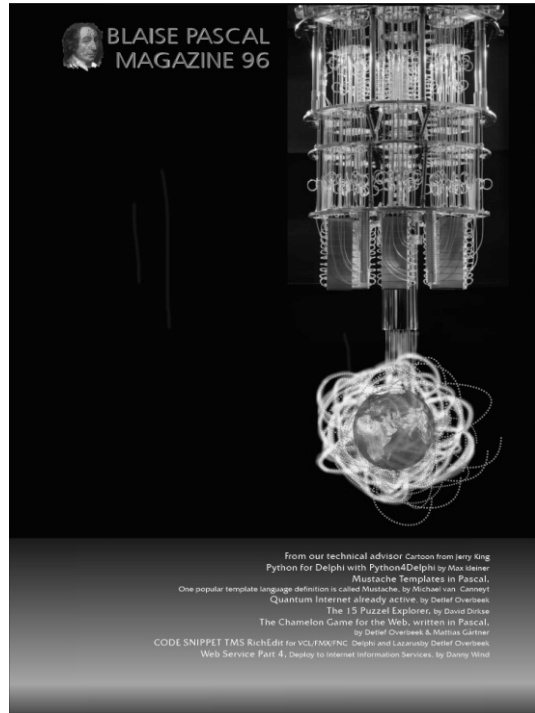
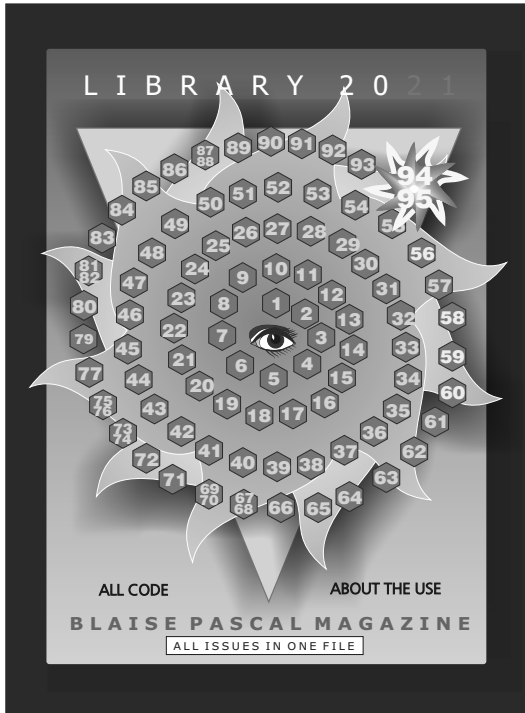
TU Delft: <https://www.tudelft.nl/over-tu-delft/strategie/vision-teams/quantum-internet-vision-team/impact-governance/the-possible-dark-side-of-quantum-communication-technologies>

If you are interested in **Quantum History**:

https://en.wikipedia.org/wiki/Quantum_mechanics



Summersale:

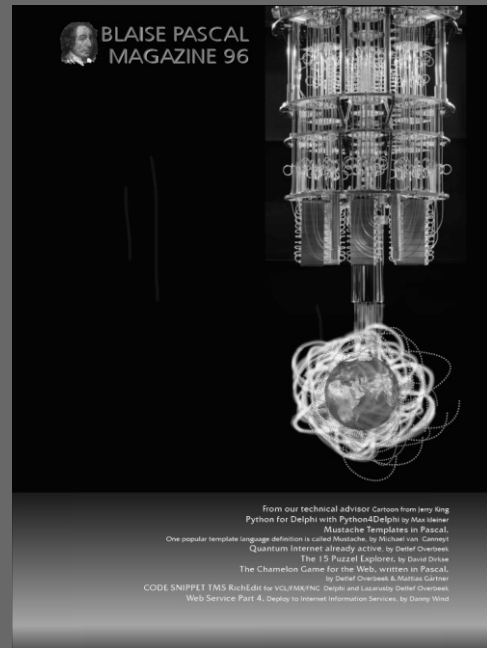
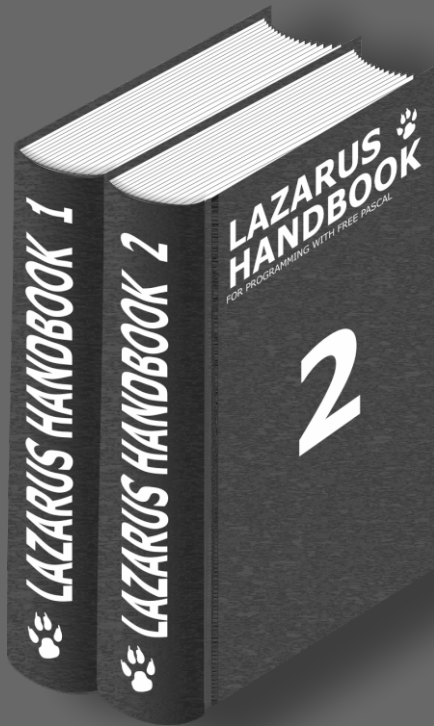


**1 year PDF
 Subscription
 + the new LibStick
 (on USB Card - 95 Issues)**

€ 70
 ex vat / no shipment



ADVERTISEMENT



**Sewn Hardcover,
almost thousand pages**
written by the makers of FPC and Lazarus

**Subscription
Combi
Subscription +
Lazarus Handbook^(hardcover)
+ Lazarus Handbook PDF
+ Lazarus Handbook Examples**
€100

Ex Vat 9% Ex Shipment

<https://www.blaisepascalmagazine.eu/product/lazarus-handbook-hardcover-subscription/>

A 15-PUZZLE EXPLORER

The 15-puzzle was around 1870 invented in New England and became very popular after 1880, sweeping across America and Europe. Below are two pictures of 15-puzzles, the left in an initial-, the right in the final (solved) state.

15	2	1	12	1	2	3	4
8	5	6	11	5	6	7	8
4	9	10	7	9	10	11	12
3	14	13		13	14	15	

By David Dirkse

INTRODUCTION

A 4x4 square holds 15 tiles numbered 1 to 15 and one empty space.

A tile may not be lifted but can shift to its neighbouring empty space.

A puzzle is solved by shifting tiles until the solved state is reached in which the tiles are sequentially ordered..

This Delphi project allows for generating 15-puzzle games with selectable difficulty, solving a game manually or have the computer search for solutions.

It helps the user to develop strategies.

Games together with solutions may be saved to disc.

For the computer search choice is between breadth-first or depth-first search.

15-puzzles are not difficult to solve, however finding the shortest solution is very hard. It has been verified that 4*4 size puzzles can be solved in 80 moves or less.

Despite the simplicity of this puzzle, much computer time is needed.

Searching for a solution takes about $t = 0.136 * 2.1^s$ microseconds where s is the search depth in moves. (3GHz clock). To explore 20 moves deep requires about 400 milliseconds, this time increases to 10 minutes for a 30 moves deep search and about doubles for each extra move.





15-puzzle: - □ ×

file set game play solve help DavData

open solved store original

save extreme original partial

save as random back reduce

STOP

1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

offset move count 10 search depth 20

tile checks clear set all

search mode breadth depth

first

move 0 + of 0

-

0

Welcome at the 15-puzzle explorer. Open or enter a new game.

This illustrates the impossibility to find the shortest solution for hard puzzles by brute force: 80 moves would take a single processor longer than the time that our solar system exists. So, in this project we are limited to approximations of the shortest solution. **NOT EVERY PUZZLE IS SOLVABLE.**

It is rather simple to shift the -1-tile to the left top corner, the -2- tile right next to it, etc.

Finally then we end up with 3 remaining tiles and 1 empty space in the right bottom corner.

These 3 tiles (11,12,15, empty) have $4! = 24$ sequences but only the 12 below lead to a solution:

11 12	11 12	12	12	12 15	12 15
15	15	11 15	11 15	11	11
15	15	15 11	15 11	11	11
12 11	12 11	12	12	15 12	15 12



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	

A single exchange of two tiles in above picture makes the puzzle unsolvable. To distinguish solvable and unsolvable puzzles needs a variable that is invariant to moves. Calculating this variable for the solved game then sets the solvability criterion.

Here is the algorithm:

1. Write the tiles left-top to right-bottom as a permutation (sequence)
2. Count the number of inversions (an inversion is a pair a,b where a > b)
3. Add "1" if the empty space is at an even row (counting rows 1,2,3,4)

A solvable puzzle has odd parity sum:

1,2,3,4,5,6,7,8,9,10,11,12,13,14,150 inversions, + 1 for empty space at row 4

Moving the empty space horizontally does not change anything.

Moving the empty space upward :

1,2,3,4,5,6,7,8,9,10,11,13,14,15,121 inversion (12,15) empty space at row 3

Odd parity, solvable puzzle.

The following game has no solution:

1	10	3	4		1	2	3	4
5	6	7	8		5	6	7	8
9	2	11	12		9	10		12
13	14	15			13	14	15	11

The permutation is 1,10,3,4,5,6,7,8,9,2,11,12,13,14,15

The right picture shows the game where all tiles except 11,12,15 are placed in position.

The number of inversions is:

For the 10 at position 2: count = 8 (10 compared to ,3 4 5 6 7 8 9 2)

For the 2 at position 10: count =7 (2 compared to 3 4 5 6 7 8 9)

1 more for the empty space at row 4.

Together 8+7+1 = 16, unsolvable puzzle.





CODING THE GAME

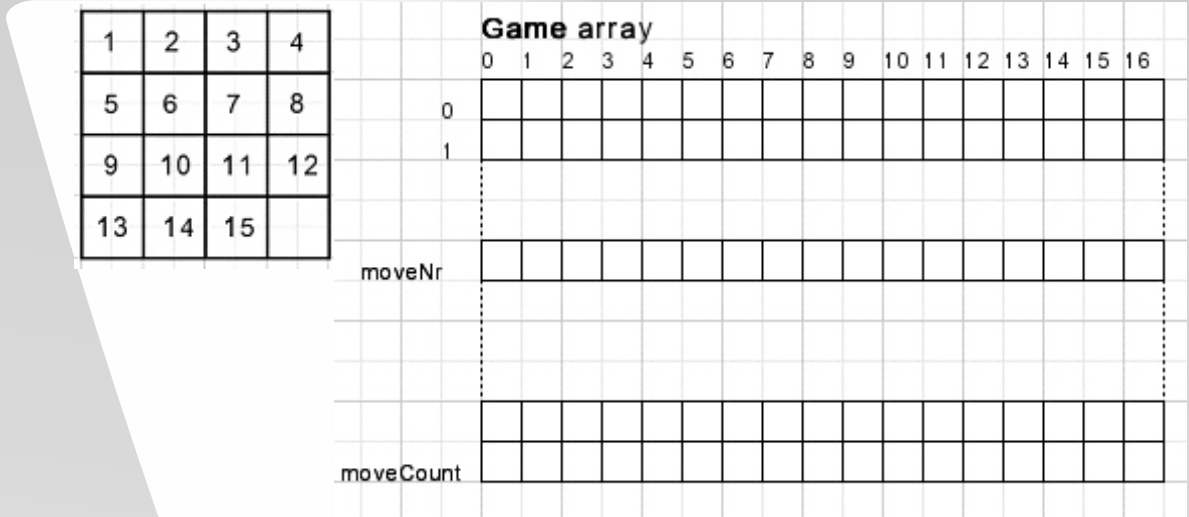
The tile and empty space positions on the board are saved as a one dimensional array[0..16].

[1..16] hold the tile number for that position.

[0] holds the position of the empty space.

A complete game is coded as an array of board positions.

Board positions:



```
const maxmove = 250;
var game      : array[0..maxmove,0..16] of byte;
moveNr       : byte;//the number of moves, points to game[moveNr,1..16]
moveCount    : byte;//total moves done
```

Game[0,0..16.] holds the original puzzle, Game[1,0..16] is the board state after one move, etc.

So, game[moveNr,0..16] is the current board state.

This coding is convenient for step by step showing of a solution.

SOLVING A GAME

The above coding is inconvenient for puzzle solving.

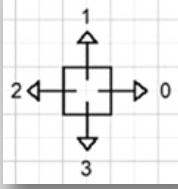
Here a counter is needed for a systematic search.

A way is to move the empty space and save the direction of moves as digits of a counter.





MOVE DIRECTIONS:



```

const gamedirections : array[1..16] of byte = ($9,$d,$d,$c,$b,$f,$f,$e,$b,$f,$f,$e,$3,$7,$7,$6);
//bit 0..3 set if direction allowed for this position
var Xfield:byte; //0 element position
var Xmove : array[0..maxmove] of byte; // directions list
XNr : byte; //current trial move number
Xdir : byte; //trial move direction
PXnr : byte; //needed for breadth first search, see later
    
```

Before searching for a solution, array XGame[1..16] is copied from the Game[n,1..16] array.

Xfield = Game[n,0] holds the position of the empty field.

The Xmove[n,0] array holds the subsequent moves (directions) of the empty field..

Array GameDirections holds a byte for each position which has its bits set for the move directions that are allowed for the empty field.

For the left top corner this value is \$9 = 1001 binary, directions 0 and 3 are allowed.

So, the search for a solution is disjunct from the game[,] array.
 In case of a failing search for a solution the search maybe reselected but now with another search depth or tile selection.

PARTIAL SOLUTIONS

To find a solution requiring 30 moves takes about 10 minutes.
 For each extra move this time doubles.

How to solve more difficult games?

The answer is to search first for game states that have only a few tiles at the right place, starting normally with tiles 1,2 .

The next search than tries to position the 3,4 tiles at the right place, then 5,6,7,8 then 9,10,13,14 and finally 11,12,15.

But of course other choices are possible.

Tile positions to check for this partial solutions are selectable by a mouseclick.
 A selected tile is indicated by its blue edges.

```

Var Gmask : array[1..16] of boolean; //true enables solution check per tile position 1..16
    
```



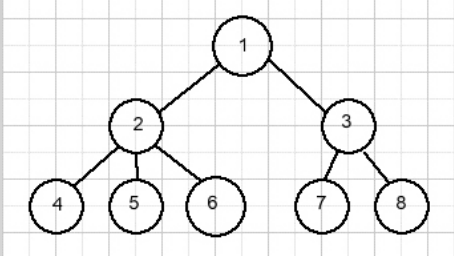


Breadth- vs depth-first search

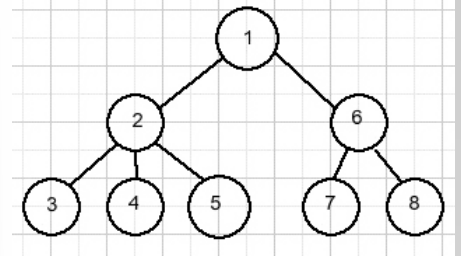
We write a solution (or search) as a graph where circles represent board states and lines between circles are moves.

Pictures below illustrate the difference between breath-first and depth-first search for a search level of 2 deep. The numbers in the circles indicate the search sequence.

BREADTH-FIRST SEARCH:



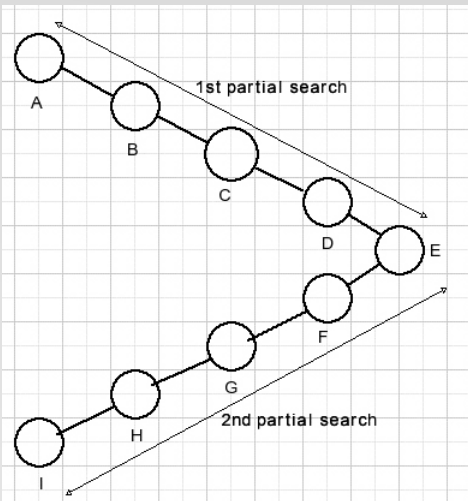
DEPTH-FIRST SEARCH:



Breadth first search always finds the shortest solution but in general depth-first search finds a solution faster. Reason is that breadth-first needs to take moves back a lot more times.

MOVE REDUCTION

For each partial solution, the number of moves from start to end game positions is minimal (when using breadth first search) .



The path from A to E is minimal and so is E to I.

However there may be a shorter path from A to F,G..or C to F,G....etc.

A shorter path from C to H eliminates board positions D,E,F,G while adding less other moves.

By varying the tile selection of partial solutions I found a 86 move solution of the most difficult puzzle in a few minutes and a 84 move solution in an hour.

Addition: I now have a 78 move solution for the “extreme” puzzle.

Selection of higher search depths and selecting more tiles per partial solution will result in shorter solutions.





BREADTH FIRST SEARCH FLOWCHART

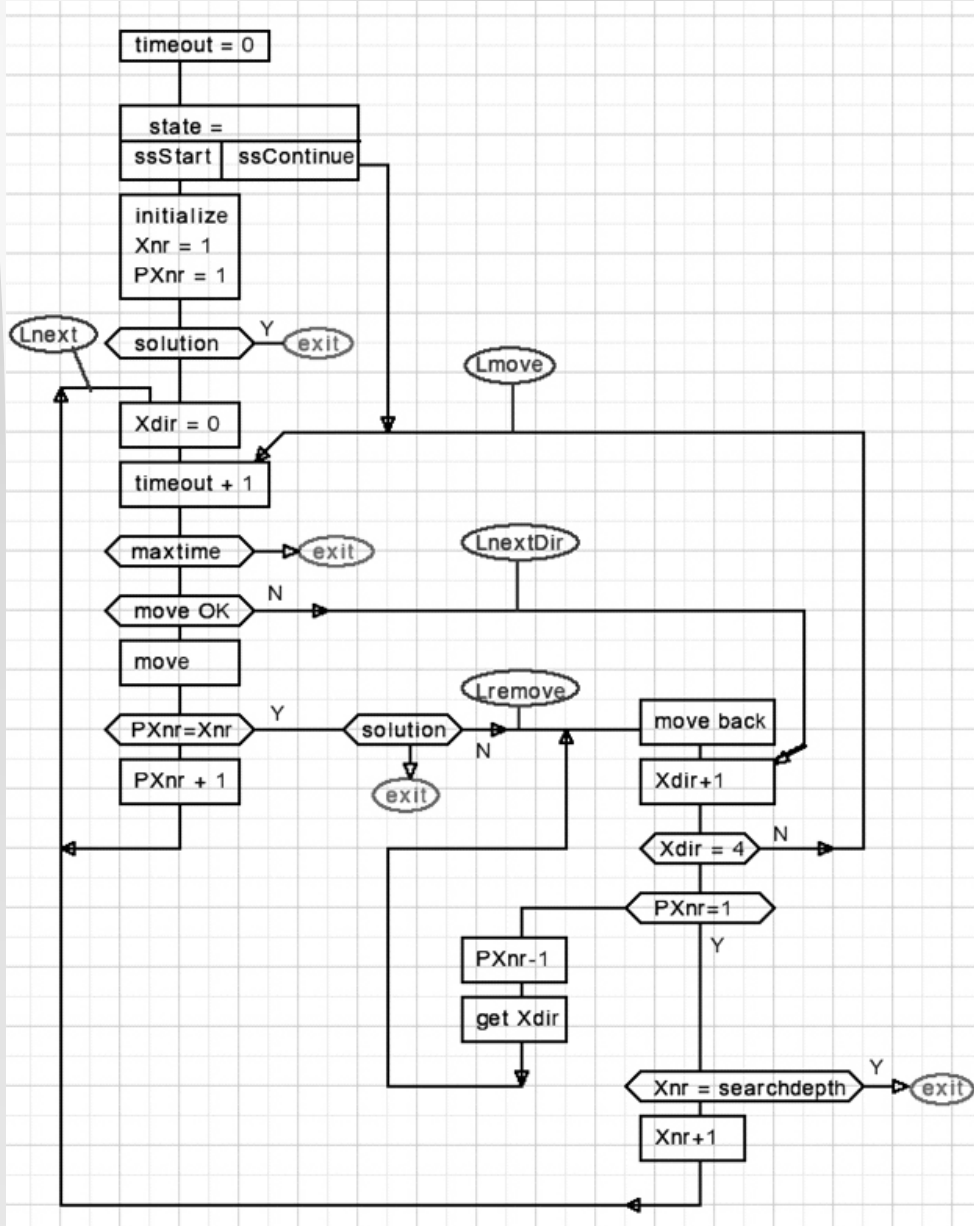
Xnr is the move number at search depth. Xnr is never decremented.

PXnr (Previous Xnr) differs from Xnr after taking back a move, so being at a previous board state.

Always $PXnr \leq Xnr$.

A check for solution is only made if $PXnr = Xnr$

A timer is added for regular interruption of the search to communicate with the Windows operating system.



Blue ovals are labels.





The search is initially started with `ssStart` followed by initialization of `Xnr,PXnr,Xgame,Xdir`.

After a time-out re-entry is with `ssContinue`.

This is the code to detect a partial solution:

```
function testPartialSolution: boolean;
//test XGame for solution
var i: byte;
begin
  result := true;
  i := 1;
  while result and (i < 16) do
  begin
    result := (Gmask[i]=false) or (Xgame[i]=i);
    inc(i);
  end;
end;
```

After a partially solved game, the `XMove[]` list with direction codes must be converted to game states and added to the `Game[. . .]` array.

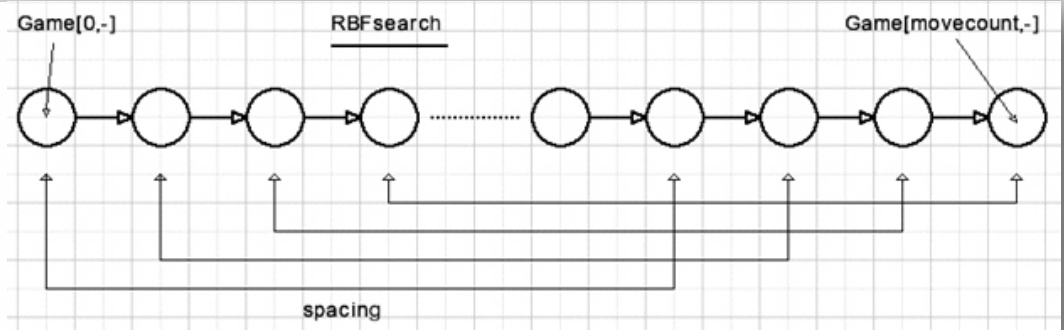
This procedure does the job :

```
function UpdateGame: boolean;
//called after (partial) solved game
//insert Xmoves into game[... ] array
var i,j,zf1,zf2: byte;
begin
  if moveNr + XNr > maxmove then begin
    result := false;
    exit;
  end;
  zf1 := Game[moveNr,0];
  zf2 := zf1;
  for i := 1 to 16 do XGame[i] := Game[moveNr,i];
  for i := 1 to Xnr do
  begin
    inc(moveNr);
    for j := 1 to 16 do Game[moveNr,j] := Game[moveNr-1,j];
    case Xmove[i] of
      0: zf2 := zf1 + 1;
      1: zf2 := zf1 - 4;
      2: zf2 := zf1 - 1;
      3: zf2 := zf1 + 4;
    end;
    Game[moveNr,0] := zf2;
    Game[moveNr,zf1] := Game[moveNr,zf2];
    Game[moveNr,zf2] := 0;
    zf1 := zf2;
  end;
  result := true;
end;
```

A full solution in most cases will be a sequence of partial solutions.

To find a shorter solution, the procedure `RBFsearch` tries to find a shorter path between two `Game[]` entries. If found, procedure `GReduce` inserts the new game states in array `Games[]`.





The RBFsearch procedure is very similar to the Breadth first search procedure. The spacing is the value of the search depth rotation button. Please refer to the source code for details.

THE PROGRAM

15-puzzle: DavData

file set game play solve help

open solved store original

save extreme original partial

save as random back reduce

STOP

partial solution search

interrupt searches

try move reduction

take move back

search depth selection

random moves from solved state

offset move count 10

search depth 20

tile with check mark

tile without check mark

masks clear set all

search breadth depth

first

show highest move

next move

previous move

original game

move 0

generate random puzzle with offset move count

select most difficult puzzle

Welcome at the 15-puzzle explorer. Open or enter a new game.





Painting of the tiles is done in a TImage component.

Some home made components are used (see previous articles or my book):

- Menu buttons are DavArrayButtons
- Search depth- and offset count are Dav7RotationButtons
- Search times are obtained from a Dav7Timer microseconds counter

HELP INFORMATION

Menu: file – open,save,save asself explanatory

File names have no extension but are automatically prefixed with P15-

Menu: Set game – solvedsets the solved game state

Extreme.....sets most difficult puzzle

Random.....offset count of random moves from the solved state
to select game of known difficulty

Click on tile next to empty field to move tile.

To exchange two tiles: mouse button down on 1st tile, keep button pressed and move to other tile, release button.

Menu: play – store.....sets the current game as the original

Original.....sets the game to its original state (no moves done)

Click on tile next to empty field to move tile.

Menu: solve – original.....set original board state

- partial.....search for partial solution

- reduce.....try to shorten a solution

Click on tile to add/remove selection for partial solution check.

Stepping through a solution

In solve or play mode:

Mouse click on "T" shows board after last move.

Mouse click on "0" shows original game, before moves.

Left mouse click on "+" . "-" shows next , previous move.

Right mouse click on "+", "-" shows all next, previous moves continuously.

PROGRAM EXAMPLE: PLAYERMOVE

Next follows a description of the first player move.

The tile positions are stored in array game[0,0..16]

moveNr = 0; movecount = 0;

menubutton = mbPlay;

A tile is moved by a mouseclick on that tile.

A mousedown event on GImage calls function XYtoField with mouse (x,y) coordinates and the field number 1..16 is returned.

Then procedure playermove is called with this field number. If the maximal move was reached, the procedure playermove exits, nothing is done.



1	2	3	4
5	6	7	8
9	10	11	12
13	14	15	16

```
procedure playermove(f1 : byte); //f1 is field number of mousedown
```

```

zf := Game[moveNr,0]; //zero field loaded in zf
n := abs(zf-f1);
if (n <> 1) and (n <> 4) then exit; //no adjacent empty field
//
m1 := moveNr + 1;
for n := 1 to 16 do Game[m1,n] := Game[moveNr,n]; //copy to next
setMoveInfo(moveNr+1,moveNr+1); //update moveNr , movecount
game[moveNr,zf] := game[moveNr,f1]; //do move
game[moveNr,f1] := 0;
game[moveNr,0] := f1;
slowmove(moveNr-1,moveNr); //tile moves to next board state
s := 'move '+inttostr(movecount);
if testsolution then
begin
  s := s + ' game solved';
end;
form1.msglabel.Caption := s;

```

```

Procedure slowmove (in paint-unit) calls
procedure paintTile(m,pos : byte; dx,dy : shortInt);
m is the move number (= index to game[] array.)

```

pos is the tile position (1..16)
dx is an offset to the tile X position
dy is an offset to the tile Y position.

The sliding move is obtained by repeatedly calling `painttile()`
with incrementing or decrementing values of dx,dy.
The speed of moving is set by calling

```

procedure delay(msec : dword); //milliseconds elay
//use dav7timer1 on form1
begin
  msec := msec * 1000;
  with form1.dav7Timer1 do
    begin
      start;
      repeat
        application.ProcessMessages;
        stop;
      until Elapsedtime >= msec; //microseconds returned
    end;
end;

```

This concludes the 15-puzzle description.



David Dirkse / Website <http://www.davdata.nl/>



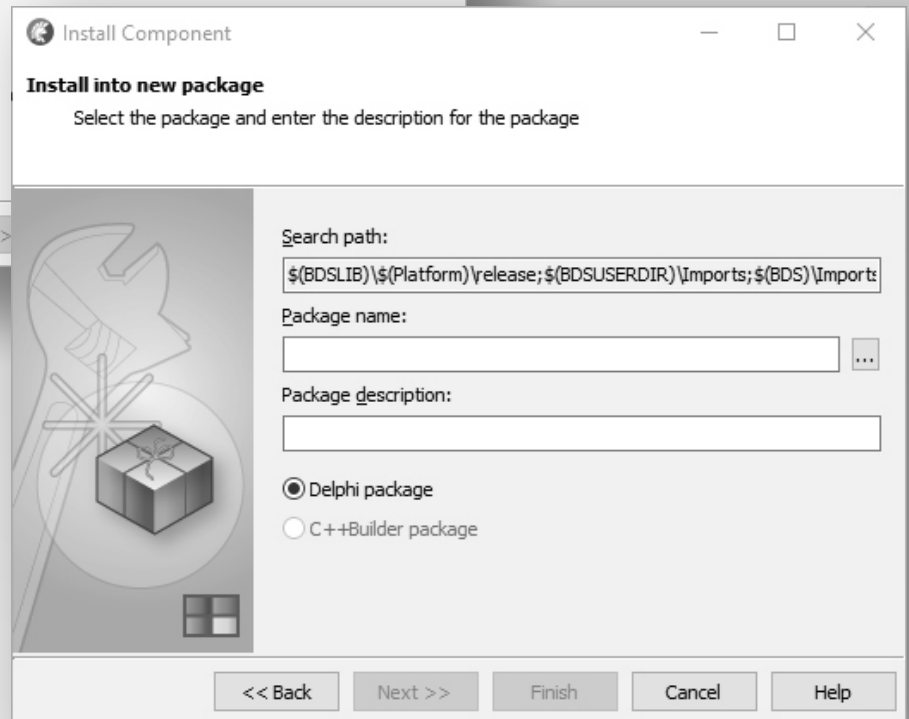
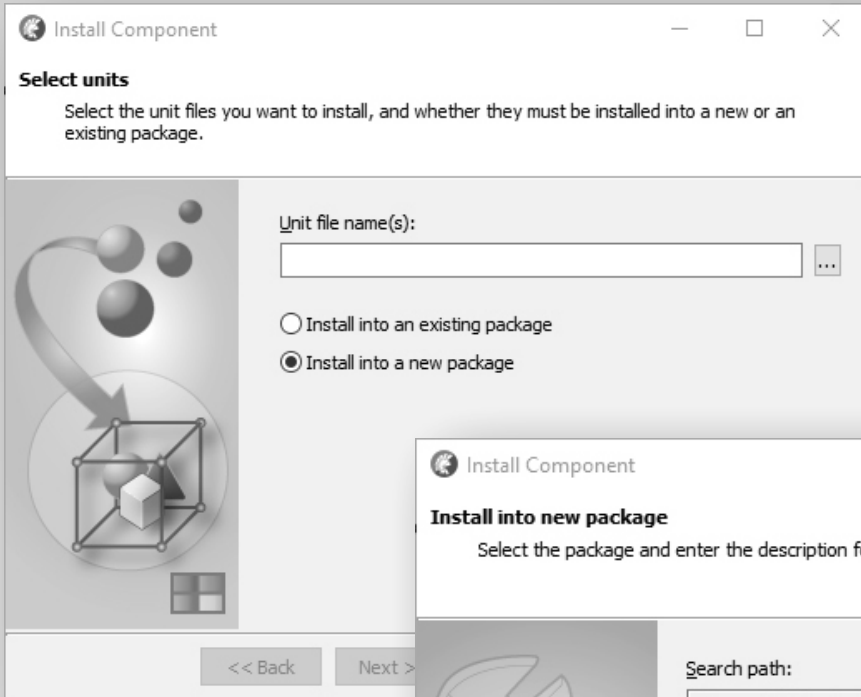


A short advice to install the components under Delphi(10.4). Delphi 7 still works. I have tested the program and succeeded after some research: first install the components and after that the project. Here are the steps how to:

Start Delphi, or Rad studio,

Go to the tab Component → Install Component → a new window starts:

add the unit File name, a .pas or .dcu file. Install into a new package, enter any given name.



In the next window enter a package name and eventually a description.

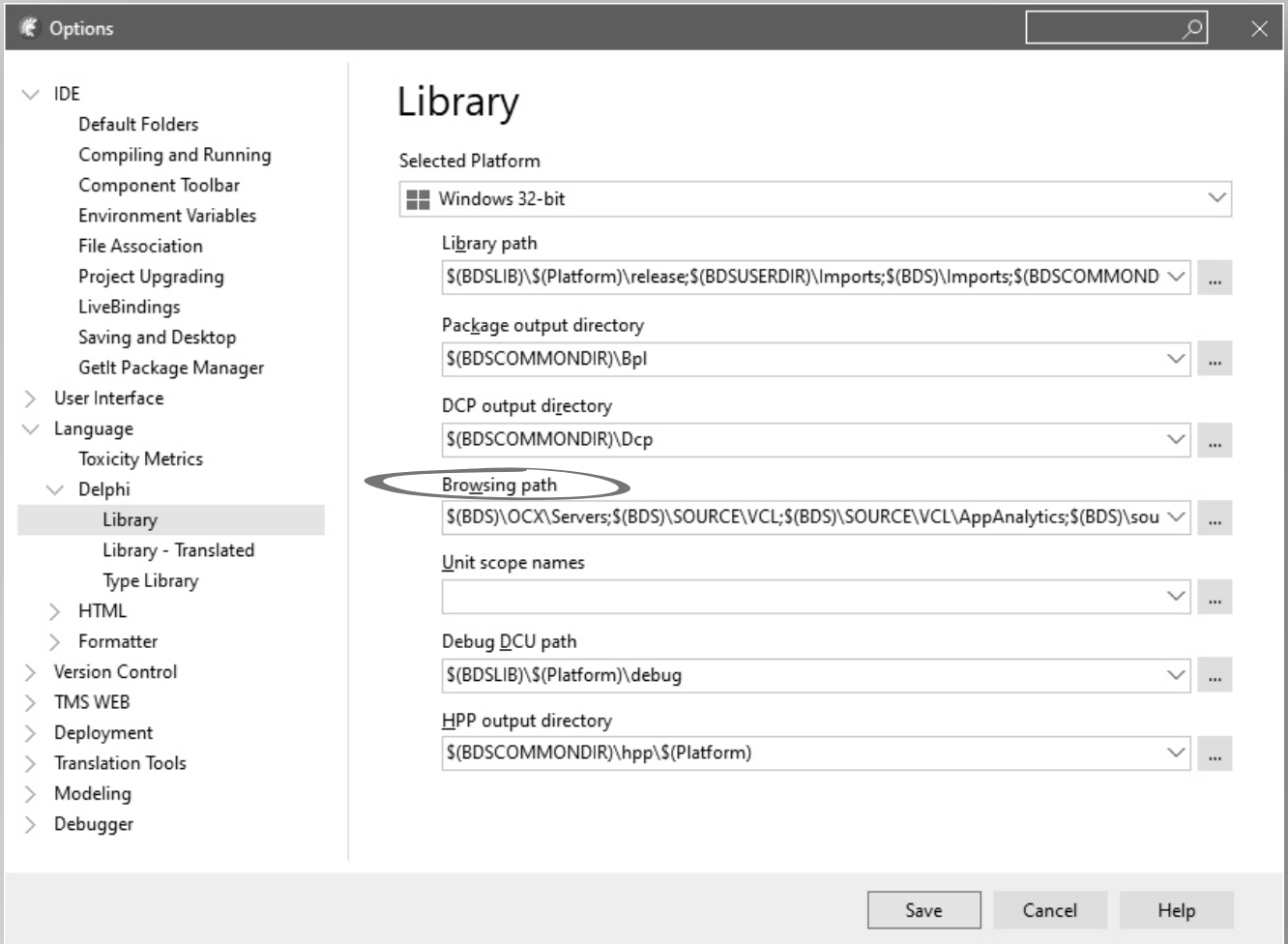
Click Finish. A new window appears and you need to accept that.

The component will now be compiled and added to (In this case) the Tab **system**.





TO make sure you have done everything correctly you can check the following:
 Click on the Tab Tools: go to the section Language: (very strange) and then Delphi
 → **Library**. Look at the **Browsing Path** and enter the path to the Dir of the component:

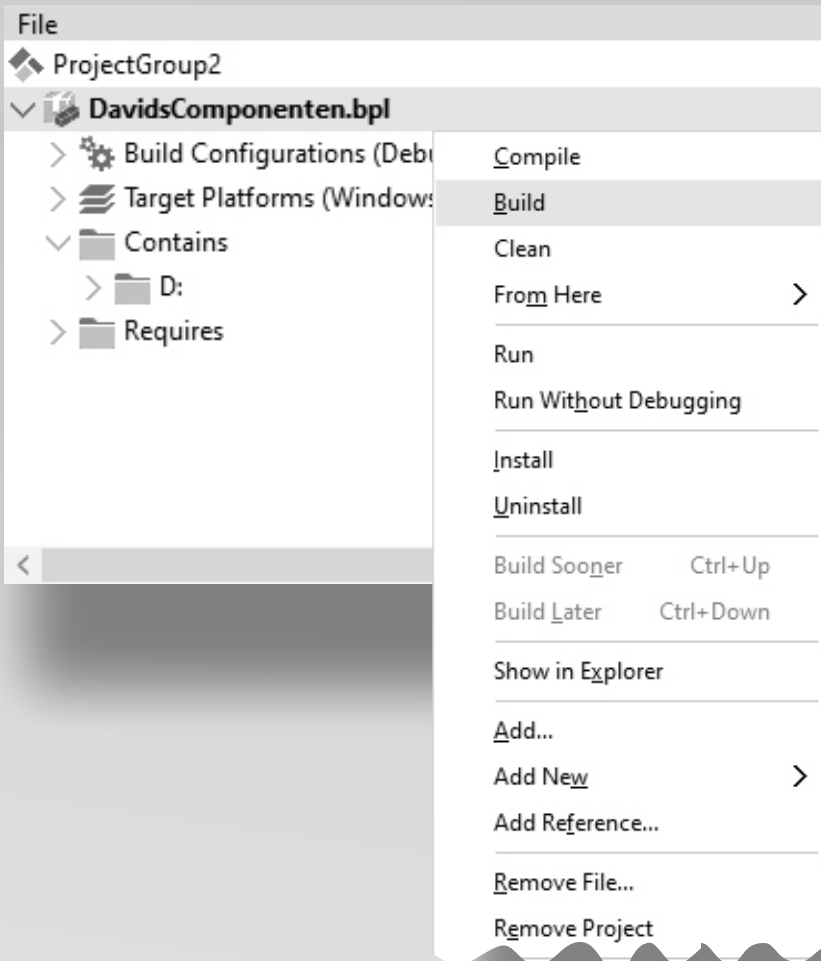


`$(BDS)\OCX\Servers;`

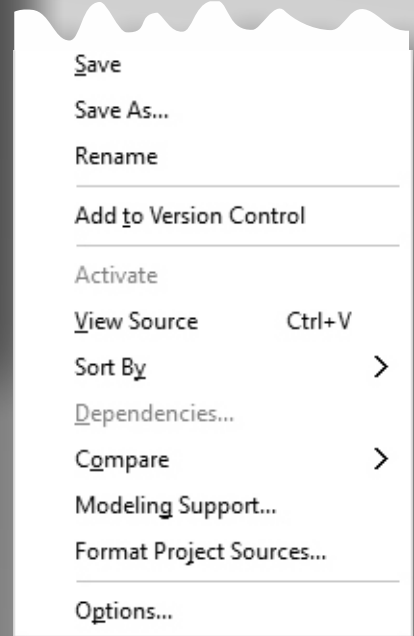
...

`C:\Program Files (x86)\FastReport VCL Enterprise\LIBD27;`
`D:\SPP\Blaise\Blaise_UK_96_2021\ Authors\David\PuzzleExplorer\Components\dav7comps ;`
`D:\SPP\Blaise\Blaise_UK_96_2021\Authors\David\PuzzleExplorer\Components\rotationBtn`
 or yours of course.



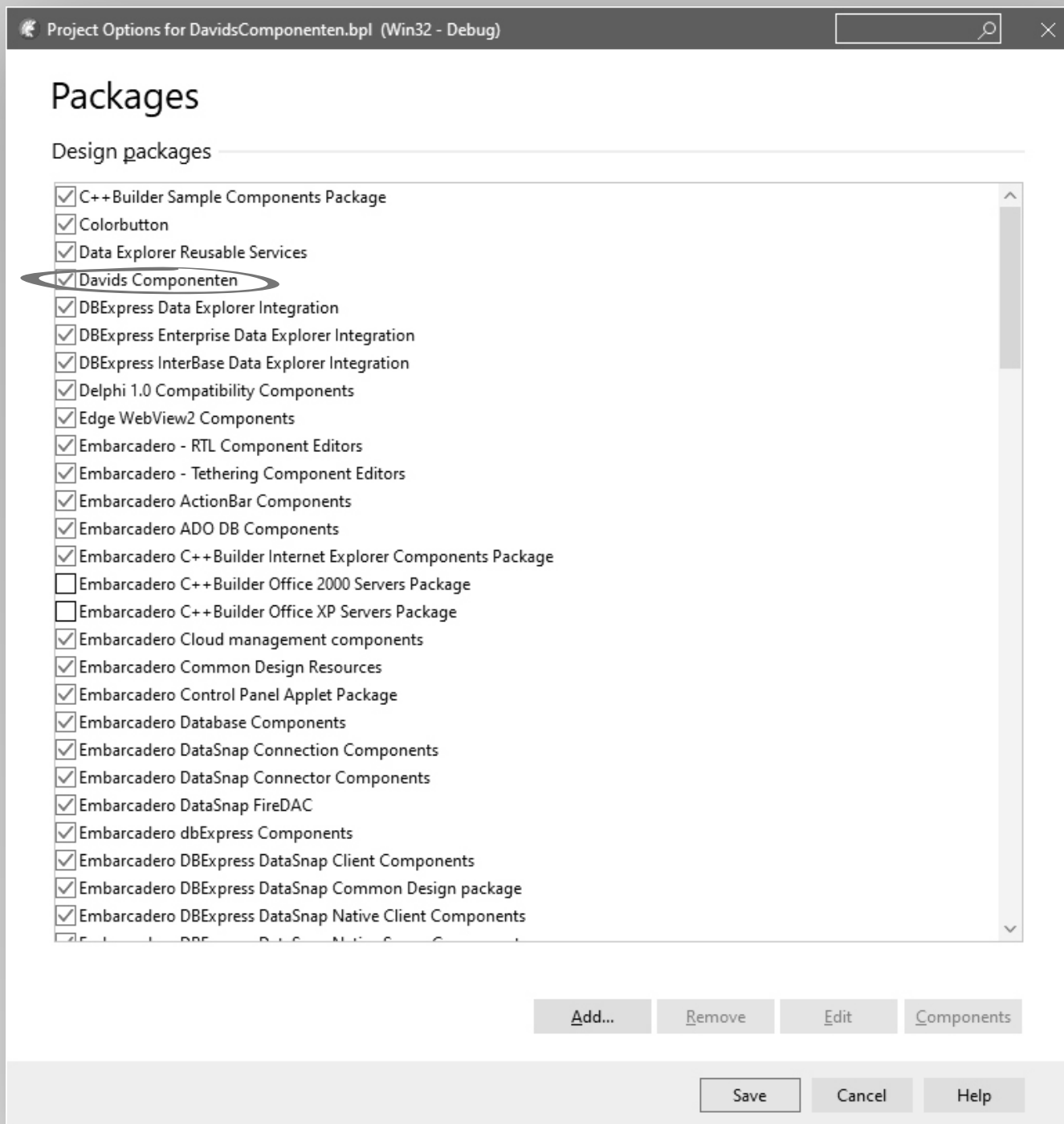


Now recompile and build the component.



The second necessary step is go to **Component** again and then to **Install Packages**. If your component is not in the list click Add. (see the figure on the next page)
 Now is the complete path for the BPL asked:
 c:\Users\Public\Documents\Embarcadero\Studio\21.0\Bp1\
 after that Save and add the other component.





You now should be able to start the project Puzzle Explorer itself including its components.

We succeeded to port these components all to Lazarus. Since they are free you can use them for other Projects as well (Delphi too), which is very kind of David Dirkse.

In the next issue I'll make a short article how the Program Works under Lazarus.



Chameleon Game for WordPress created in Lazarus

By Detlef Overbeek & Coding Mattias Gärtner



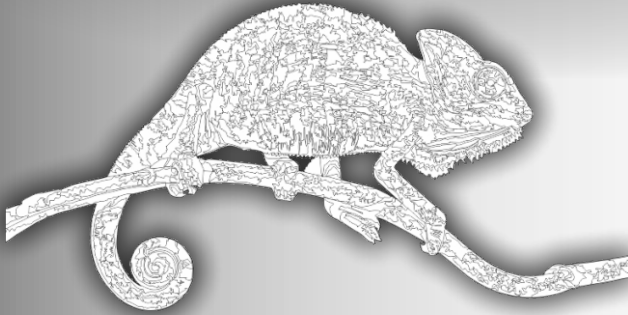
starter



expert

INTRODUCTION:

In this article I'll try to show how Pas2Js can be used to create a small pastime game. To reset and refill your creativity(?). You can learn a lot of how to use the Pas2JS by unravelling the project as I did here in this article...I show the pieces of code and make them available in Pascal and show what the result would be in HTML and or JavaScript.



The game is easy: give the chameleon its colour back. And if you take a deeper dive into the coding elements, you could learn something about coding and building it. Though it is not complex, it needs quite some methods and functions and they all will be set to JavaScript through **PAS2JS**.

To start with:

lets sum up the necessities for this game:

- 1 Elucidate some of the history of the project
- 2 Creating a list of the requirements
- 3 Explain the coding

1 THE HISTORY OF THE PROJECT

The basic idea was to create a new element in advertising in a **PDF**: an interactive advertisement. So we tried to create an array in a **PDF** created by **Adobe** by using **JavaScript**: possible in **Adobe Acrobat**.

So finally we found out, it could work,

but the grid needs quite a lot of pieces for the puzzle.

Because of this and to show the grid it took too much time, so it wasn't possible.

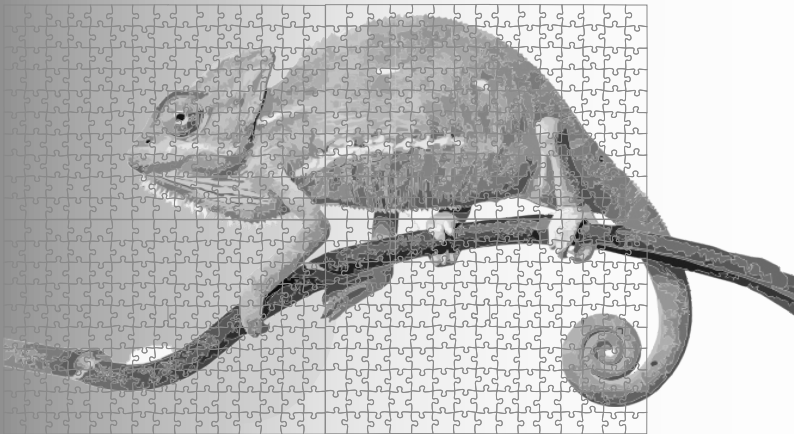
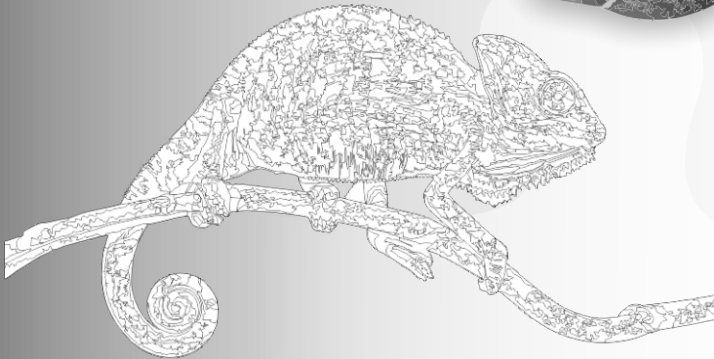
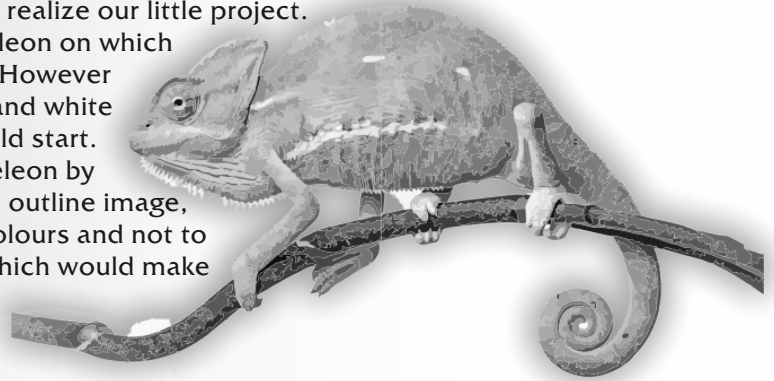
No one wants to wait for a number of minutes before anything was shown.

We still wanted to try and finally found another medium: a browser.





The web has of course enough speed to realize our little project. The image started as a coloured Chameleon on which we tried to change the original colours. However we needed something simpler: a black and white version. I created that, so finally we could start. I created an outline image of the Chameleon by converting the bitmap in to a vectorised outline image, this because of the separation for the colours and not to have to create a special array-overlay which would make it much more complex.



The original puzzle elements are difficult because each of them has to have a different outline which can only fit in one way. So you would need to have more than thousand pieces and that was not to be done in a short while.

I created the first basic grid of the puzzle pieces, but then came to a much simpler idea: why not simply use the pixel sizes. The browsers are made for that. All you need is the coordinates of the mouse and you could create an event at a given point.





② REQUIREMENTS

Lazarus Minimal Version 2.0.12

download: <https://www.lazarus-ide.org/index.php?page=downloads>

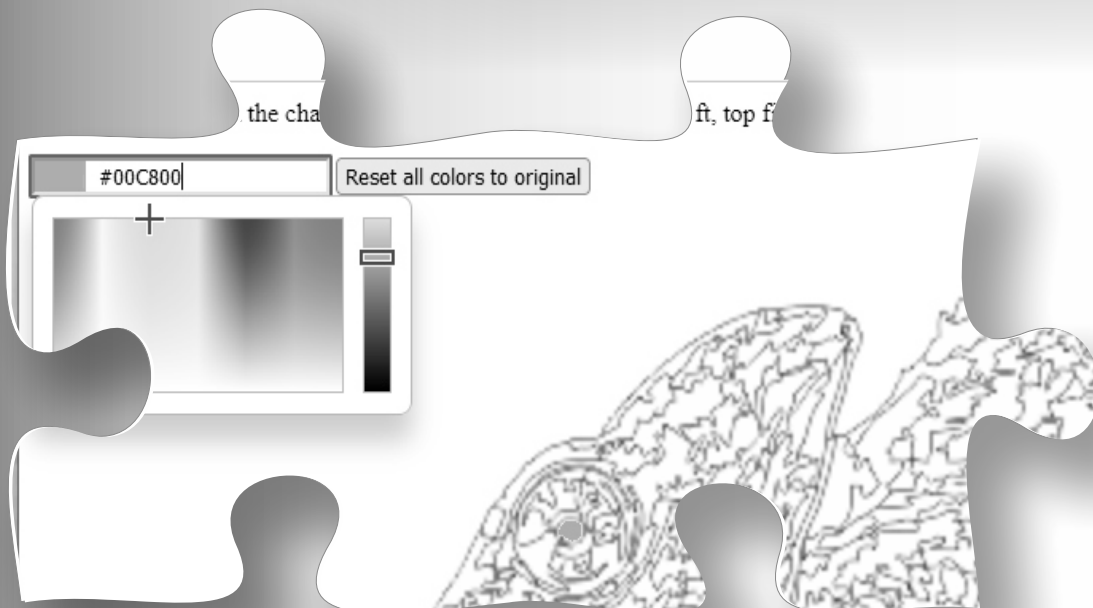
Pas2JS Minimal Version 206

download: <https://wiki.freepascal.org/pas2js>

An OS that can run any Browser – (Windows , Linux, macOS). I used Chrome.

For the Color Picker:

```
/**
 * jscolor - JavaScript Color Picker
 *
 * @link    http://jscolor.com
 * @license For open source use: GPLv3
 *          For commercial use: JSColor Commercial License
 * @author  Jan Odvarko - East Desire
 * @version 2.4.5
 *
 * See usage examples at http://jscolor.com/examples/
 */
```





③ THE CODING

Lets start wit the final code and explain how to show the result of the program- the website running and show its functionality. Since the program is complete now we should first of all compile it. The code is situated in my case at d:\MattiasChamelon_Black_White\ in the command line below you can find that path at the end of the line.

So to explain the command line: The first part is the path for **Pas2JS** followed by the directory we need. Here you will find the compileserver.exe.

Now some help for this:

```
Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\Users\edito>C:\Pas2JS_206\pas2js-windows-2.0.6\bin\i386-
win32\compileserver.exe -help
Error: Invalid option at position 1: "e"
Usage compileserver.exe [options]
Where options is one or more of :
-d --directory=dir      Base directory from which to serve files.
                        Default is current working directory: C:\Users\Yourname
-h --help              This help text
-i --indexpage=name    Directory index page to use (default: index.html)
-n --noindexpage      Do not allow index page.
-p --port=NNNN        TCP/IP port to listen on (default is 3000)
-q --quiet            Do not write diagnostic messages
-w --watch            Watch directory for changes
-c --compile[=proj]   Recompile project if pascal files change.
                        Default project is app.lpr
-m --mimetypes=file   filename of mimetypes.
                        Default is C:\Pas2JS_206\pas2js-windows-2.0.6\bin\
                        i386-win32\mime.types
-s --simpleserver     Only serve files, do not enable compilation.
```

After the exe file `-n --noindexpage` and `-p --port=NNNN` (portnumber 3000) followed by `-s --simpleserver` only serves files, does not enable compilation and `-d --directory=dir` is the base directory from which to serve files. Default is current working directory: `C:\Users\(your name)`.

```
Command Prompt - C:\Pas2JS_206\pas2js-windows-2.0.6\bin\i386-win32\compileserver.exe -n -p 3000 -s -d "D:\MattiasChamelon_Black_White"
Microsoft Windows [Version 10.0.19043.1052]
(c) Microsoft Corporation. All rights reserved.

C:\Users\edito>C:\Pas2JS_206\pas2js-windows-2.0.6\bin\i386-win32\compileserver.exe -n -p 3000 -s -d "D:\MattiasChamelon_
Black_White"
2021-07-06 11:30:03.186 [etInfo] Listening on port 3000, serving files from directory: D:\MattiasChamelon_Black_White
2021-07-06 11:30:03.186 [etInfo] Compile requests will be ignored.
2021-07-06 11:30:51.241 [etInfo] 200 serving "ChameleonBlackWhite1.html" -> "D:\MattiasChamelon_Black_White\ChameleonBl
ackWhite1.html"
```





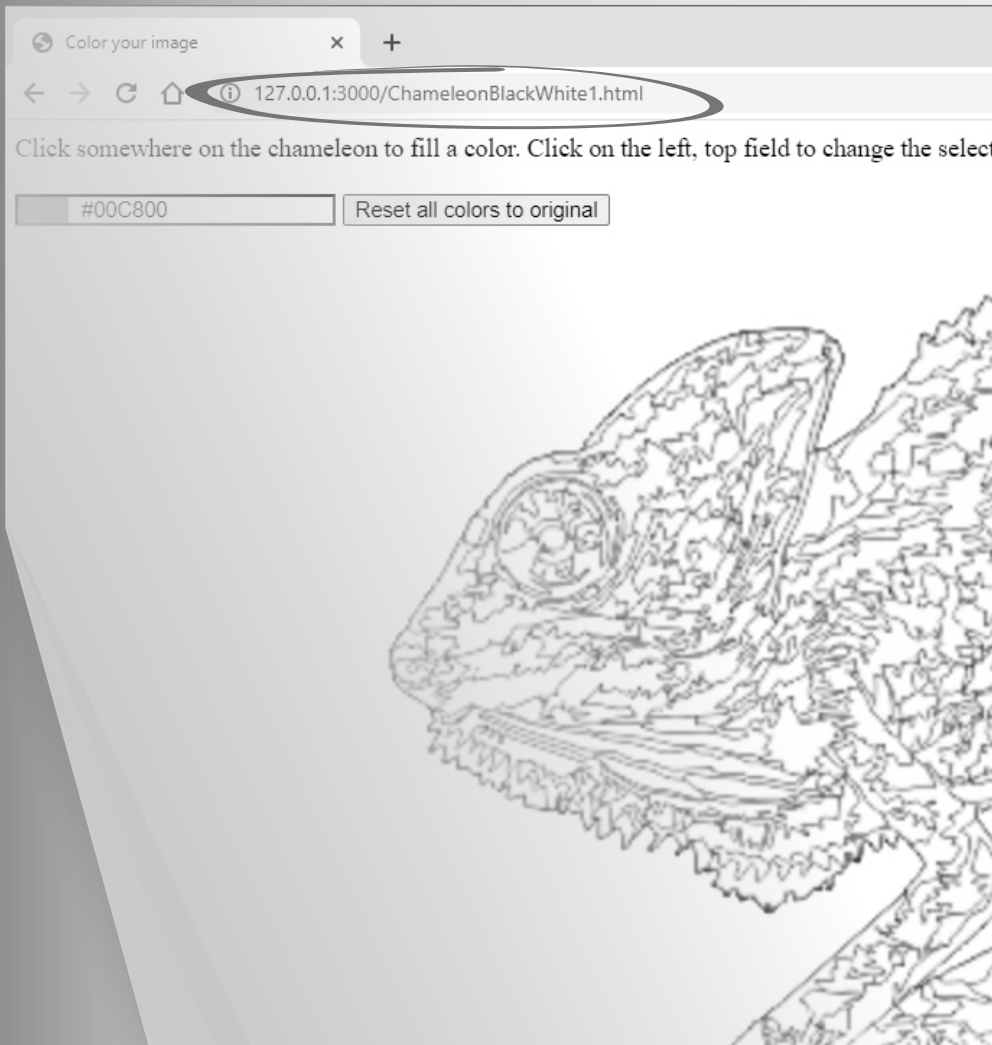
So if you want to start the program you need to open a command prompt (or Linux: a terminal) under macOS I haven't tried but it should be the same.

and then add the following line of text:

```
C:\Pas2JS_206\pas2js-windows-2.0.6\bin\i386-win32\compileserver.exe -n -p 3000 -s -d "D:\MattiasChamelon_Black_White".
```

Then open this page in a browser:

<http://127.0.0.1:3000/ChameleonBlackWhite1.html>.





Now to get this all done in a WordPress website you do not need to install special things. I'll explain some items. You need to install a new widget: There is one that simply adds an Image and one you need to use for the code a custom HTML widget. They are standard available.

We have separated the JavaScript so you can have a view at that. We tried to find the best **ColorPicker** but the problem was that only the one we used was responding in the way we wanted. Writing one your self is out of the scope: it would take days to do so and than debug it.

HERE IS HOW IT WORKS:

Chose a colour in the Picker and click on what ever point in the Chameleon. The colour will be added. If you want to save your work, you can save it locally and even if you want to start all over again you can go back to the blanc and empty settings. There are a few cluster of pixels you will find popping up information. You can play with that in the settings. I'll explain that in the coding.

Here comes the implementation of the coding part:
Pascal - Html – JavaScript:

```
{
  C:\Pas2JS_206\pas2js-windows-2.0.6\bin\i386-win32\compileserver.exe -n -p 3000 -s -d
  „D:\MattiasChamelon_Black_White" http://127.0.0.1:3000/ChameleonBlackWhite1.html
}
program ChameleonBlackWhite1;

{$mode objfpc}
{$ModeSwitch externalclass}

uses JS, Classes, SysUtils, Web, Types;

const
  LocalStorageHistoryName = 'FloodFillHistory';
  WhiteMin = 230;
  AlphaMin = 254;

type // This section of Type is fore creating some Functions and procedures for overall use
  TJSCPChangeEvent = reference to procedure;

  TJSCColorPicker = class external name 'JSColor'
  public
    constructor New(Input: TJSHtmlInputElement);
    OnChange: TJSCPChangeEvent; external name 'onChange';
    function toHexString: string; external name 'toHEXString';
    procedure fromRGBA(r, g, b, a: byte);
  end;

  TFloodFillHistoryItem = class
  public
    x,y: NativeInt;
    Color: string; // hex value
  end;
  TFloodFillHistoryItems = array of TFloodFillHistoryItem;

function RGBToHexValue(r, g, b: byte): string;
begin
  Result:='#'+HexStr(r,2)+HexStr(g,2)+HexStr(b,2);
end;

procedure HexValueToRGB(Value: string; out Red, Green, Blue: byte);
begin
  Red:=parseInt(copy(value,2,2),16);
  Green:=parseInt(copy(value,4,2),16);
  Blue:=parseInt(copy(value,6,2),16);
end;
```





```
type ///////////////////////////////////////////////////Begin of the app itself //////////////////////////////////////
```

```
{ TChameleonApp }
```

```
TChameleonApp = class
```

```
private
```

```
img: TJSHTMLImageElement;
Canvas: TJSHTMLCanvasElement;
ctx: TJSCanvasRenderingContext2D;
ColorPickerInput: TJSHTMLInputElement;
Picker: TJSColorPicker;
ResetButton, ConfirmResetButton: TJSHTMLButtonElement;
History: TFloodFillHistoryItems;
OrigPixel: TJSImageData;
OrigData: TJSUint8ClampedArray;
function OnCanvasClick(Event: TEventListenerEvent): boolean;
function OnConfirmResetClick(Event: TEventListenerEvent): boolean;
function OnLoad(Event: TEventListenerEvent): boolean;
procedure OnPickerChange;
function OnResetClick(Event: TEventListenerEvent): boolean;
function ClickChameleon(Event: TJSMouseEvent): boolean;
function FloodFill(CurData: TJSUint8ClampedArray; w,h,x,y: NativeInt; NewRed,NewGreen,NewBlue: byte): boolean;
procedure ShowHidePopup(ElName: string; px, py: integer; Show: boolean);
procedure ShowPopupHideOther(ElName: string; px, py: integer);
function GetPopupAt(px, py: integer): string;
procedure AddHistoryItem(x,y: NativeInt; NewRed,NewGreen,NewBlue: byte);
procedure SaveHistory;
function LoadHistory: TFloodFillHistoryItems;
procedure ApplyHistory;
procedure DrawImage;
public
procedure InitChameleon;
end;
```

```
function TChameleonApp.OnLoad(Event: TEventListenerEvent): boolean;
```

```
var r: TJSRect; w, h: NativeInt;
```

```
begin
```

```
Result:=true;
DrawImage;
img.style.setProperty('display','none');
if Event=nil then; // Nothing
r:=Canvas.getBoundingClientRect; w:=round(r.width); h:=round(r.height);
OrigPixel:=ctx.getImageData(0,0,w,h);
OrigData:=OrigPixel.data;
// load history
History:=LoadHistory;
ApplyHistory;
end;
```

```
procedure TChameleonApp.ShowHidePopup(ElName: string; px, py: integer; Show: boolean);
```

```
var El: TJSHTMLElement; // eyeurl, leftbottomeyeurl, tailurl in javascript : line 1777 see next page 7
```

```
begin
```

```
El:=TJSHTMLElement(document.getElementById(ElName));
if Show then
begin
El.style.setProperty('left',IntToStr(px)+'px');
El.style.setProperty('top',IntToStr(py)+'px');
El.style.removeProperty('display');
end
else
El.style.setProperty('display','none');
end;
```

```
procedure TChameleonApp.ShowPopupHideOther(ElName: string; px, py: integer);
```

```
procedure ShowHide(CurElName: string);
```

```
begin
```

```
ShowHidePopup(CurElName,px,py,ElName=CurElName);
```

```
end;
```

```
begin
```

```
ShowHide('eyeurl');
ShowHide('leftbottomeyeurl');
ShowHide('tailurl');
end;
```





```

function TChameleonApp.GetPopupAt(px, py: integer): string;
begin // Create popup points
  if (px>325) and (py>180) and (px<346) and (py<206) then exit('yeurl');
  if (px>300) and (py>196) and (px<337) and (py<225) then exit('leftbottomeyeurl');
  if (px>1060) and (py>570) and (px<1200) and (py<700) then exit('tailurl');
  writeln('GetPopupAt Mouse=',px,',',py);
end;

////////////////////////////////////// HISTORY ////////////////////////////////////////
procedure TChameleonApp.AddHistoryItem(x, y: NativeInt; NewRed, NewGreen, NewBlue: byte);
var Item: TFloodFillHistoryItem;
begin
  Item:=TFloodFillHistoryItem.Create;
  Item.x:=x;
  Item.y:=y;
  Item.Color:=RGBToHexValue(NewRed,NewGreen,NewBlue);
  TJSArray(History).push(Item);
  SaveHistory;
end;

procedure TChameleonApp.SaveHistory;
var s: String; i: Integer; Item: TFloodFillHistoryItem;
begin // store as json
  s:='[';
  for i:=0 to length(History)-1 do
  begin
    Item:=History[i];
    s:='{"x":'+str(Item.x)+'"', 'y":'+str(Item.y)+'"', 'c":'+Item.Color+'"}';
    if i<length(History)-1 then s:=s+',';
  end;
  s:=s+']';
  writeln('SaveHistory ',s);
  Window.LocalStorage.setItem(LocalStorageHistoryName,s);
end;

function TChameleonApp.LoadHistory: TFloodFillHistoryItems;
var json: String; Arr: TJSArray; i: Integer; Obj: TJSObject; Item: TFloodFillHistoryItem;
begin
  Result:=nil;
  json:=Window.localStorage.getItem(LocalStorageHistoryName);
  if (not hasValue(json)) or (json='') then exit;
  writeln('LoadHistory json=',json);
  try Arr:=TJSArray(TJSJSON.parse(json));
  except
    writeln('LoadHistory json=',json);
    writeln('LoadHistory json parse error: ',JSExceptValue);
    exit(nil);
  end;
  writeln('LoadHistory arr=',Arr.length);
  SetLength(Result,Arr.length);
  for i:=0 to Arr.Length-1 do
  begin
    Obj:=TJSObject(Arr[i]);
    Item:=TFloodFillHistoryItem.Create;
    Item.x:=NativeInt(Obj['x']); Item.y:=NativeInt(Obj['y']);
    Item.Color:=String(Obj['c']);
    Result[i]:=Item;
  end;
end;

procedure TChameleonApp.ApplyHistory;
var r: TJSRect; Item: TFloodFillHistoryItem; w, h: NativeInt; pixel: TJSImageData;
Data: TJSUint8ClampedArray; Red, Green, Blue: byte; i: Integer;
begin
  r:=Canvas.getBoundingClientRect; w:=round(r.width); h:=round(r.height);
  pixel:=ctx.getImageData(0,0,w,h);
  Data:=pixel.data;

  for i:=0 to length(History)-1 do
  begin
    Item:=History[i];
    HexValueToRGB(Item.Color,Red,Green,Blue);
    FloodFill(Data,w,h,Item.x,Item.y,Red,Green,Blue);
  end;

  ctx.putImageData(pixel,0,0);
end;
////////////////////////////////////// HISTORY ////////////////////////////////////////

```





```
<div id="eyeurl" style="display:none;
position: absolute; z-index:1; background-color: white; border-style: solid;
border-color: black; border-width: 1px; padding: 3px">

<a href="https://www.blaisepascalmagazine.eu"
target="_blank">https://www.blaisepascalmagazine.eu</a><br>
You clicked the eye!<br>

</div>
```

```
<div id="leftbottomeyeurl" style="display:none;
position: absolute; z-index:1; background-color: white; border-style:
solid;
border-color: black; border-width: 1px; padding: 3px">

<a href="https://www.blaisepascalmagazine.eu"
target="_blank">https://www.blaisepascalmagazine.eu</a><br>
Clicked left below the eye!<br>

</div>
```

```
<div id="tailurl" style="display:none; position: absolute; z-index:1;
background-color: white; border-style: solid;
border-color: black; border-width: 1px; padding: 3px">

<a href="https://www.blaisepascalmagazine.eu"
target="_blank">https://www.blaisepascalmagazine.eu</a><br>
Clicked on tail<br>

</div>
```





```

procedure TChameleonApp.DrawImage;
begin
  ctx.drawImage(img,0,0,1520,752);
end;

```

```

function TChameleonApp.ClickChameleon(Event: TJSMouseEvent): boolean;
var ex,ey: Double; pixel: TJSImageData; Data: TJSUint8ClampedArray;
  value: string; r: TJSRect; x,y,w,h: NativeInt; NewRed,NewGreen,NewBlue: Byte;
begin
  Result:=true;
  r:=Canvas.getBoundingClientRect;
  w:=round(r.width);
  h:=round(r.height);
  ex:=Event.clientX - r.left;
  ey:=Event.clientY - r.top;
  x:=round(ex);
  y:=round(ey);

  pixel:=ctx.getImageData(0,0,w,h);
  Data:=pixel.data;

```

```

// flood fill

```

```

value:=Picker.toHexString;
WriteLn('picked color: '+value);
HexValueToRGB(value,NewRed,NewGreen,NewBlue);

```

```

if FloodFill(Data,w,h,x,y,NewRed,NewGreen,NewBlue) then

```

```

begin
  AddHistoryItem(x,y,NewRed,NewGreen,NewBlue);
  ctx.putImageData(pixel,0,0);
end;

```

```

ShowPopupHideOther(GetPopupAt(x,y),round(Event.clientX)+20,round(Event.clientY));
end;

```

```

function TChameleonApp.FloodFill(CurData: TJSUint8ClampedArray; w, h, x,
  y: NativeInt; NewRed, NewGreen, NewBlue: byte): boolean;
var p: NativeInt; OldRed, OldGreen, OldBlue, OldAlpha: Byte; PxStack: TNativeIntDynArray;

```

```

procedure Check(ax,ay: NativeInt);

```

```

var ap: NativeInt;

```

```

begin

```

```

if (ax<0) or (ay<0) or (ax>=w) or (ay>=h) then exit; // outside Canvas

```

```

ap:=(ax+ay*w)*4; // series of rows, each pixel as red, green, blue, alpha bytes.

```

```

// writeLn('Check ',ax,',',ay,',',CurData[ap+0],',',CurData[ap+1],',',CurData[ap+2],',',CurData[ap+3]);

```

```

if (OrigData[ap+0]<WhiteMin) or (OrigData[ap+1]<WhiteMin) or (OrigData[ap+2]<WhiteMin)
or (OrigData[ap+3]<AlphaMin) then exit; // not a white, opaque pixel

```

```

if (CurData[ap+0]=NewRed) and (CurData[ap+1]=NewGreen) and (CurData[ap+2]=NewBlue) then exit; // already colored

```

```

CurData[ap+0]:=NewRed;

```

```

CurData[ap+1]:=NewGreen;

```

```

CurData[ap+2]:=NewBlue;

```

```

TJSArray(PxStack).push(ap);

```

```

end;

```

```

begin

```

```

  Result:=false;

```

```

if (x>=w) or (y>=h) then exit;

```

```

if OrigData=nil then exit;

```

```

p:=(x+y*w)*4;

```

```

OldRed:=OrigData[p];

```

```

OldGreen:=OrigData[p+1];

```

```

OldBlue:=OrigData[p+2];

```

```

OldAlpha:=OrigData[p+3];

```

```

writeLn('TChameleonApp.FloodFill ',x,',',y,' rgba=',OldRed,',',OldGreen,',',OldBlue,',',OldAlpha);

```

```

  Check(x,y);

```

```

while length(PxStack)>0 do

```

```

begin

```

```

  Result:=true;

```

```

  p:=NativeInt(TJSArray(PxStack).pop) div 4;

```

```

  x:=p mod w;

```

```

  y:=p div w;

```

```

  Check(x,y-1);

```

```

  Check(x,y+1);

```

```

  Check(x-1,y);

```

```

  Check(x+1,y);

```

```

end;

```

```

end;

```



```
function TChameleonApp.OnCanvasClick(Event: TEventListenerEvent): boolean;
begin
  ConfirmResetButton.style.setProperty('display','none');
  Result:=ClickChameleon(TJSMouseEvent(Event));
end;
```

```
function TChameleonApp.OnResetClick(Event: TEventListenerEvent): boolean;
begin
  Result:=true;
  ConfirmResetButton.style.removeProperty('display');
  if Event=nil then ;
end;
```

```
function TChameleonApp.OnConfirmResetClick(Event: TEventListenerEvent): boolean;
begin
  Result:=true;
  ConfirmResetButton.style.setProperty('display','none');

  DrawImage;
  SetLength(History,0);

  if Event=nil then ; //DO nothing
  SaveHistory; //Set a memory point
end;
```

```
procedure TChameleonApp.OnPickerChange;
var value: String;
begin
  ConfirmResetButton.style.setProperty('display','none');

  value:=Picker.toHexString;
  Writeln('picked color: '+value);
end;
```

```
procedure TChameleonApp.InitChameleon;
var imgdiv: TJSHTMLDivElement; i: Integer;
begin
  img:=nil;
  imgdiv:=TJSHTMLDivElement(document.getElementById('originalimage'));
  for i:=0 to imgdiv.children.length-1 do
    if imgdiv.children.item(i) is TJSHTMLImageElement then
      img:=TJSHTMLImageElement(imgdiv.children.item(i));
    if img=nil then
      begin
        writeln('InitChameleon img not found');
        exit;
      end;
  Canvas:=TJSHTMLCanvasElement(document.getElementById('canvas'));
  ctx:=Canvas.getContextAs2DContext('2d');
  ColorPickerInput:=TJSHTMLInputElement(document.getElementById('colorpicker'));
  ResetButton:=TJSHTMLButtonElement(document.getElementById('resetbutton'));
  ResetButton.addEventListener('click',@OnResetClick);
  ConfirmResetButton:=TJSHTMLButtonElement(document.getElementById('confirmresetbutton'));
  ConfirmResetButton.addEventListener('click',@OnConfirmResetClick);

  Canvas.addEventListener('click',@OnCanvasClick);

  Picker:=TJSColorPicker.New(ColorPickerInput);
  Picker.OnChange:=@OnPickerChange;
  Picker.fromRGBA(0,200,0,255);
  ColorPickerInput.value:=RGBToHexValue(0,200,0);

  if not img.complete then img.onload:=@OnLoad else OnLoad(nil);
end;
```

```
var App: TChameleonApp;
begin //Main program
  App:=TChameleonApp.Create;
  App.InitChameleon;
```





```

<!doctype html>
<html lang="en">
<head>
  <meta http-equiv="Content-type" content="text/html; charset=utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Color your image</title>
  <script src="jscolor.js"></script>
  <script src="ChameleonBlackWhite1.js"></script>
</head>

<body>
  <div>Click somewhere on the chameleon to fill a color.
    Click on the left,
    top field to change the selected color.</div><br>

  <input id="colorpicker">
  <button id="resetbutton"
    title="Set original colors"
    type="button">Reset all colors to original</button>

  <button id="confirmresetbutton"
    title="Confirm reset all colors, all your changes will be lost"
    type="button" style="display:none">Confirm reset all color</button>
  <br>
  <div id="originalimage">
    
  </div>

  <canvas id="canvas" width=1520 height=752></canvas>

  <div id="eyeurl" style="display:none;
    position: absolute; z-index:1; background-color: white; border-style: solid;
    border-color: black; border-width: 1px; padding: 3px">

  <a href="https://www.blaisepascalmagazine.eu"
    target="_blank">https://www.blaisepascalmagazine.eu</a><br>
    You clicked the eye!<br>
    
  </div>

  <div id="leftbottomeyeurl" style="display:none;
    position: absolute; z-index:1; background-color: white; border-style: solid;
    border-color: black; border-width: 1px; padding: 3px">

  <a href="https://www.blaisepascalmagazine.eu"
    target="_blank">https://www.blaisepascalmagazine.eu</a><br>
    Clicked left below the eye!<br>
    
  </div>

  <div id="tailurl" style="display:none; position: absolute; z-index:1;
    background-color: white; border-style: solid;
    border-color: black; border-width: 1px; padding: 3px">

  <a href="https://www.blaisepascalmagazine.eu"
    target="_blank">https://www.blaisepascalmagazine.eu</a><br>
    Clicked on tail<br>
    
  </div>

  <script>
    rtl.run();
  </script>

</body>

</html>

```

HTML Page



```

<div>Click somewhere on the chameleon to select a color. Click on the left, top field to change
the selected color.</div><br>

<input id="colorpicker">
<button id="resetbutton" title="Set original colors" type="button">Reset all colors to
original</button>
<button id="confirmresetbutton" title="Confirm reset all colors, all your changes will be
lost" type="button" style="display:none">Confirm reset all color</button>
<br>

<canvas id="canvas" width=1520 height=752></canvas>

<div id="eyeurl" style="display:none; position: absolute; z-index:1000; background-color:
white; border-style: solid; border-color: black; border-width: 1px; padding: 3px">
<a href="https://www.blaisepascalmagazine.eu"
target="_blank">https://www.blaisepascalmagazine.eu</a><br>
You clicked the eye!
<!-- Image -->
<a href="https://www.blaisepascalmagazine.eu" target="_blank"></a>
</div>

<div id="leftbottomeyeurl" style="display:none; position: absolute; z-index:1000;
background-color: white; border-style: solid; border-color: black; border-width: 1px; padding:
3px">
<a href="https://www.blaisepascalmagazine.eu"
target="_blank">https://www.blaisepascalmagazine.eu</a><br>
Clicked left below the eye!<br>

</div>

<div id="tailurl" style="display:none; position: absolute; z-index:1000; background-color:
white; border-style: solid; border-color: black; border-width: 1px; padding: 3px">
<a href="https://www.blaisepascalmagazine.eu"
target="_blank">https://www.blaisepascalmagazine.eu</a><br>
Tail clicked!<br>

</div>

```

The rest of the JavaScript is available in the „JavascriptWordpress.txt“ because that file would take about 10 pages of code. You can download it as a subscriber from <https://www.blaisepascalmagazine.eu/your-downloads/> after you have logged in.



Edit Page < Blaise Pascal Magazin x +

blaisepascalmagazine.eu/wp-admin/post.php?post=11582&action=edit

Blaise Pascal Magazine 2 0 + New View Page Delete Cache WPForms 2

- Dashboard
- Posts
- Media
- Pages
- All Pages
 - Add New
 - Restrict Access
- Comments
- Projects
- Testimonials
- Downloads
- WooCommerce
- Products
- Analytics
- Marketing
- WPForms

WordPress 5.7.2 is available! [Please update now.](#)

Edit Page

There is an autosave of this post that is more recent than the version below. [View the autosave](#)

Color Game Chameleon

Permalink: <https://www.blaisepascalmagazine.eu/colorgame/>

- Image
 - full
- Custom HTML
 - Chameleon

Word count: 9704



Custom HTML

meleon

Content:

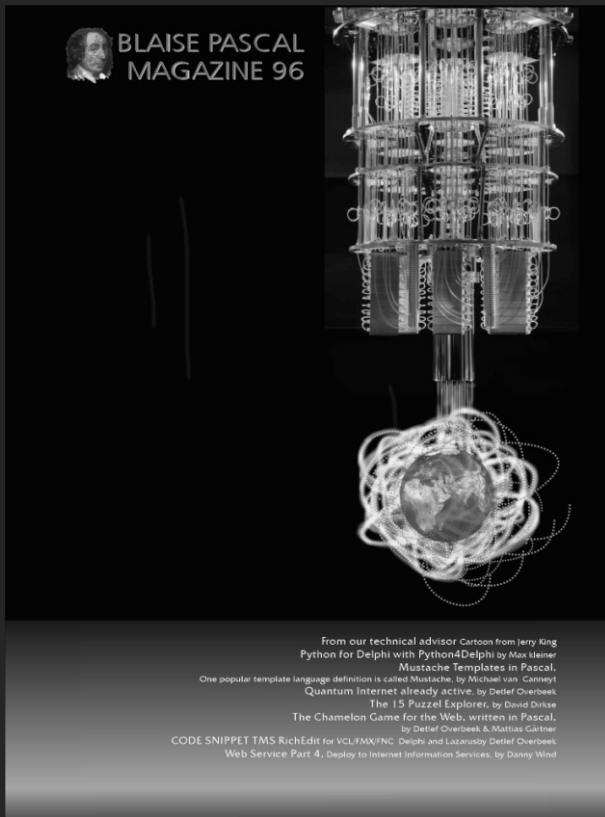
```

1 <div>Click somewhere on the chameleon to select a color. Click on the left, top field to change the selected color
  </div><br>
2
3 ~ <input id="colorpicker">
4 ~ <button id="resetbutton" title="Set original colors" type="button">Reset all colors to original</button>
5 ~ <button id="confirmresetbutton" title="Confirm reset all colors, all your changes will be lost" type="button"
  style="display:none">Confirm reset all color</button>
6 ~ <br>
7 ~
8 ~ <canvas id="canvas" width=1520 height=752</canvas>
9
10 ~ <div id="eyeurl" style="display:none; position: absolute; z-index:1000; background-color: white; border-style: s
  border-color: black; border-width: 1px; padding: 3px">
11 ~ <a href="https://www.blaisepascalmagazine.eu" target="_blank">https://www.blaisepascalmagazine.eu</a><br>
12 ~ You clicked the eye!
13 ~ </div>

```

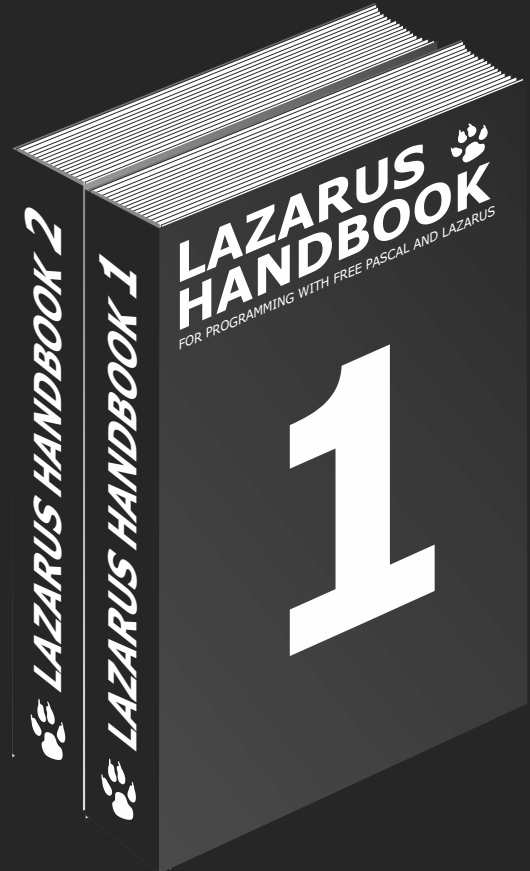


ADVERTISEMENT



BLAISE PASCAL
MAGAZINE 96

From our technical advisor: Cartoon from Jerry King
Python for Delphi with Python4Delphi by Max Klein
Mustache Templates in Pascal,
One popular template language definition is called Mustache, by Michael van Canneyt
Quantum Internet already active, by Detlef Overbeek
The 15 Puzzle Explorer, by David Dinsse
The Chamelon Game for the Web, written in Pascal,
by Detlef Overbeek & Matthias Gartner
CODE SNIPPET TMS RichEdit for VCL/FMX/FNC Delphi and Lazarus by Detlef Overbeek
Web Service Part 4, Deploy to Internet Information Services, by Danny Wind



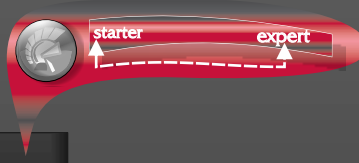
Subscription Combi

Subscription + Lazarus Handbook
(Download) (Softcover + PDF)

€ 75

normal price: 40 + 70 = € 110

Ex Vat 9% Ex shipment



ABSTRACT

This article is about the TMS VCL Component the Rich Editor. It is an advanced version of the Delphi equivalent. It is not an Editor, but get's close to it.

INTRODUCTION

Because I promised to let you know about a Rich Editor that does all in an easy way I wanted to, I have downloaded the Rich editor from TMS in my case for VCL.

I found out that there is a version that you can use in combination with a Database, and that is a very nice extra.

To make that all visible I created a small app: The TMS RichEditor is a WYSIWYG editor for formatted text. **TAdvRichEditor** can include formatted text with:

bullets, hyperlinks, images, indenting, aligned paragraphs, actually it is almost a little word editor. It offers an enormous number of functions:

merging / highlighting text / find & replace, undo/redo / clipboard / printing, auto-correct and emoticons.

You can use a large set of buttons, via ribbons and if you buy the TMS VCL UI Pack it offers even a spelling checker and export to PDF, RTF and HTML files

The core component is **TAdvRichEditor**. This is a standalone component that can be used as-is for WYSIWYG editing of formatted text. Internally the RichEditor consists of a simple DOM. This DOM is a generic list of document elements. Different types of document elements exist such as a text element, image element, linebreak element, bullet element, ...

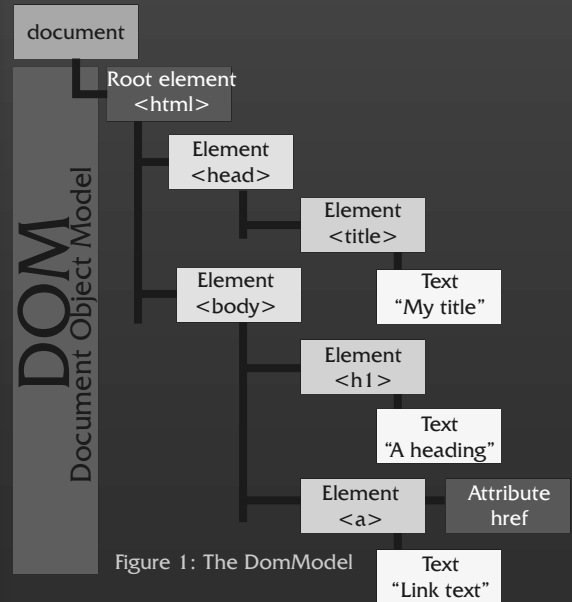


Figure 1: The DomModel



The

WIKIPEDIA **Document Object**

Model (DOM) is a cross-platform and language-independent interface that treats an XML or HTML document as a tree structure wherein each node is an object representing a part of the document.

The DOM represents a document with a logical tree. Each branch of the tree ends in a node, and each node contains objects.

DOM methods allow programmatic access to the tree; with them one can change the structure, style or content of a document. Nodes can have event handlers attached to them. Once an event is triggered, the event handlers get executed. The principal standardization of the DOM was handled by the World Wide Web Consortium (W3C), which last developed a recommendation in 2004. WHATWG (Web Hypertext Application Technology Working Group) took over the development of the standard, publishing it as a living document. The W3C now publishes stable snapshots of the WHATWG standard.



Each document element has several attributes that determine the appearance in the document. While the `TAdvRichEditor` provides a large series of methods to add or remove elements from the DOM, it is also accessible via `TAdvRichEditor.Context.Content`. It is recommended though that the API is used instead of direct DOM manipulation. As always I try to use the component as is. That is the best way to understand and try the component. You 'll immediately find out if it does what you want it to do. And it does even without one written line of code.
My compliments.

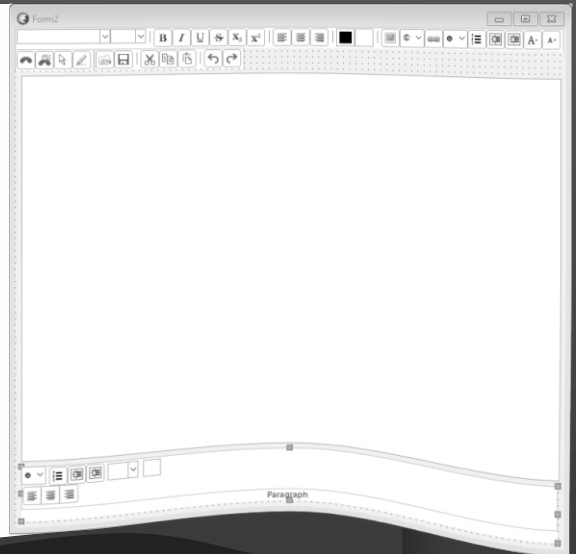


Figure 2: The project



Figure 3: The used ButtonBars

To create the project I started a VCL project and added only a few things to the form:

`A TAdvRichEditor`

The component with its default settings is ready for use. Entering of text can be done with default font & alignment.

For ease of use, connect a `TAdvRichEditorEditButtonBar` and `TAdvRichEditorFormatButtonBar` that presents most of the built-in actions as a button bar or use the docking toolbars `TAdvRichEditorEditToolBar`, `TAdvRichEditorFormatToolBar`, `TAdvRichEditorEditingToolBar`, `TAdvRichEditorParagraphToolBar` to apply all kinds of formatting to the text without writing any code or use its ribbon equivalents for a WYSIWYG editor with ribbon UI. See figure 3



What I very much appreciate is a new service: when you rightclick on the component a list appears that gives extra options

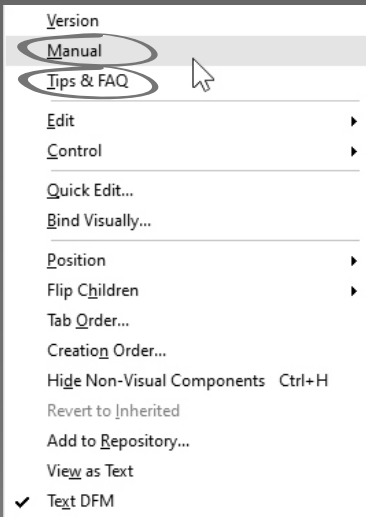


Figure 4: Right click on the component

Yet an other one:

Tips and FAQ. A list of tips complete with extra code ready for use.

This is very well thought of. The manual itself contains all the Properties etc.

No use to add them here – there are too many.

If you would like to try all this yourself you can simply download the trial version.

I advise you to. It is always an extra pleasure to have a small kind of word editor in your larger project. That makes you program independent of the editor your client eventually has...

Below you find a thumbnail of the website. On the next page the important part of the site is shown.

You'll get there if you choose to click Tips and FAQ:

<https://www.tmssoftware.com/site/advricheditor.asp?s=faq>

There is also a version of the Rich Editor available for FNC, as well a Rich Editor voor FNC cross-platform, cross-IDE & cross-framework, so TMS FNC Rich Editor can also be used with Windows, Android, iOS, macOS, Linux apps both from DELPHI as LAZARUS.



Figure 5: The thumbnail with the address, see next page



Tips and Frequently Asked Questions

◀ Color text on the fly while typing

It is possible to check rules after the user types a word in TAdvRichEditor and based on this, apply formatting to the entered text. This sample code snippet will apply a red color when the user has entered a number value in the TAdvRichEditor:

[view plain text](#)

```
procedure TForm1.AdvRichEditor1EnterWord(Sender: TObject; AWord: string);
var
  v,e,ci,ss,s1: integer;
  el: TREElement;
begin
  val(aword, v, e);
  if (e = 0) then
  begin
    // is a number

    // store current selection
    ss := AdvRichEditor1.SelStart;
    s1 := AdvRichEditor1.SelLength;
    ci := AdvRichEditor1.Caret.CharIndex;

    if AdvRichEditor1.Caret.Element.Text[ci] = ' ' then
      dec(ci);

    // set selection to entered word to apply new color
    AdvRichEditor1.Selection.FromElement := AdvRichEditor1.Caret.Element;
    AdvRichEditor1.Selection.FromChar := ci - Length(AWord);
    AdvRichEditor1.Selection.ToElement := AdvRichEditor1.Caret.Element;
    AdvRichEditor1.Selection.ToChar := ci;

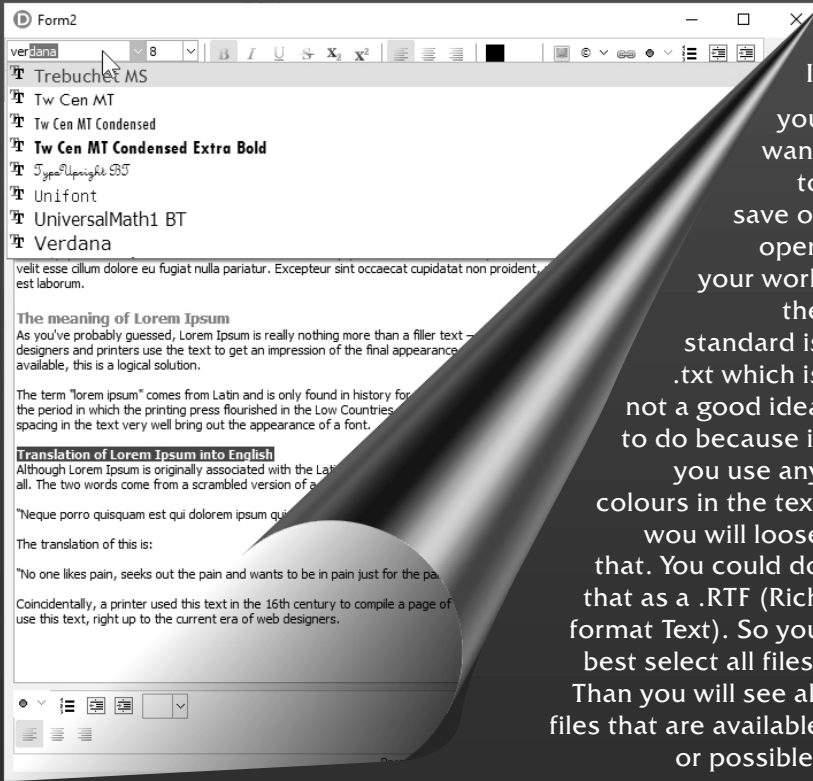
    AdvRichEditor1.SetSelectionColor(clRed);

    // restore current selection
    AdvRichEditor1.SelStart := ss;
    AdvRichEditor1.SelLength := 0;
    AdvRichEditor1.SelectionToCaret;
    AdvRichEditor1.SetSelectionColor(clBlack);
  end;
end;
```

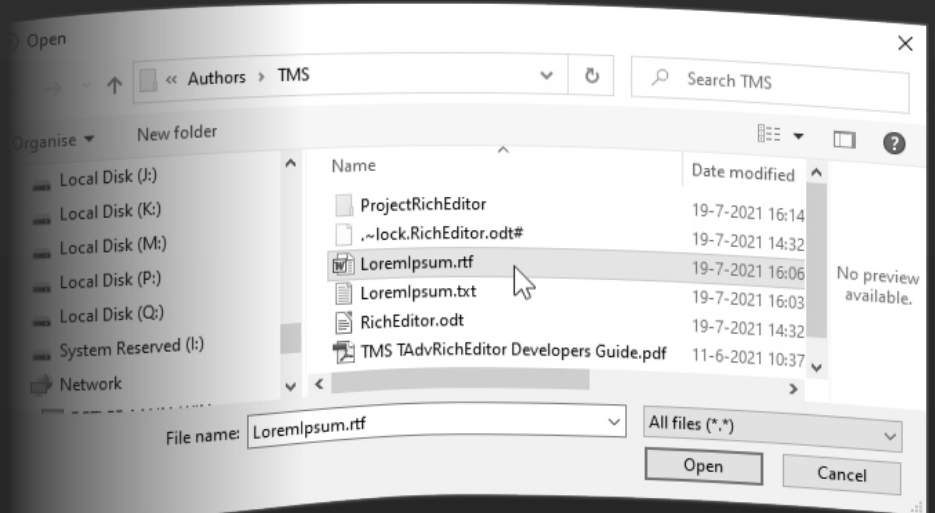
Bruno Fierens (Nov 28, 2019)

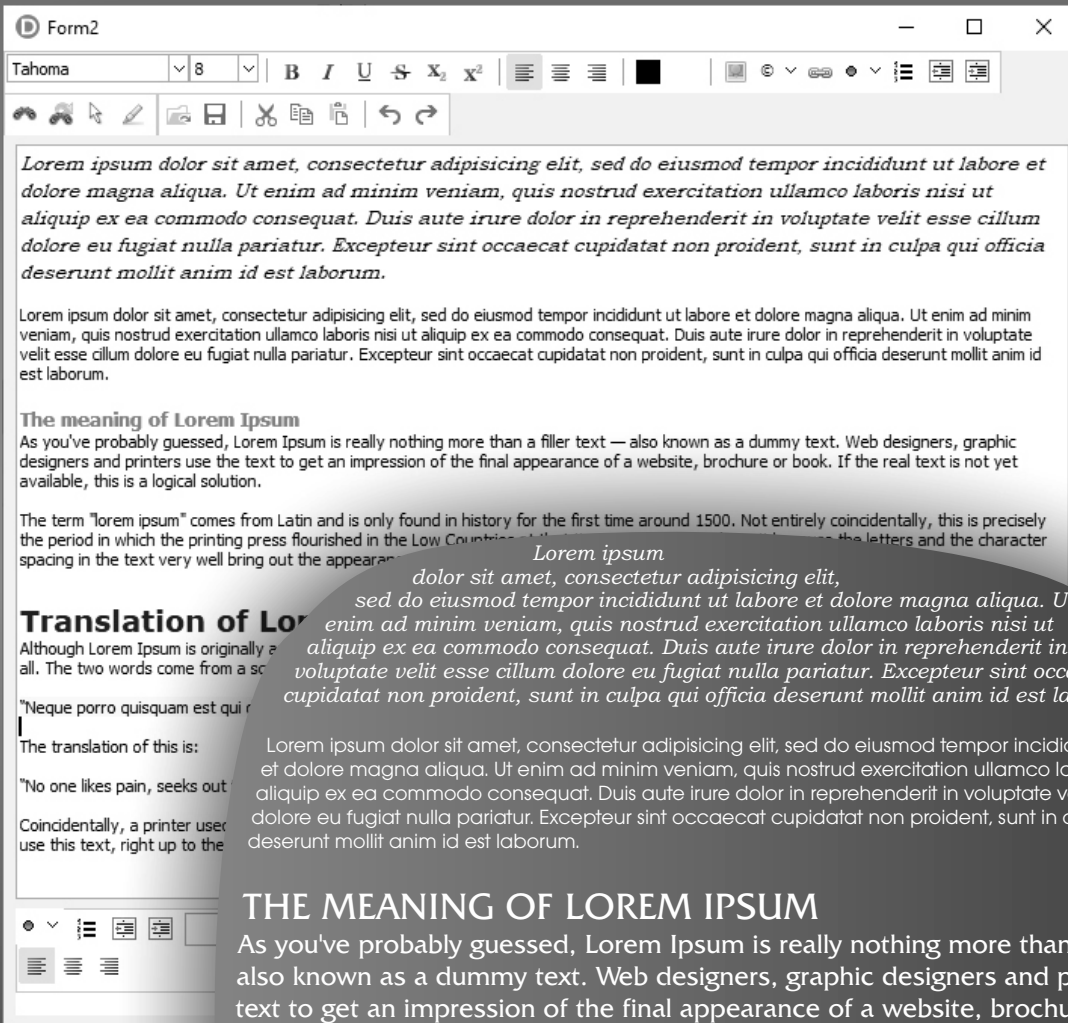
- ▶ [Controlling clipboard format](#)
- ▶ [Use TAdvRichEditor as logging tool](#)
- ▶ [Customizing the popup menu](#)
- ▶ [How to use a watermark](#)
- ▶ [How to add color & font style formatted text](#)
- ▶ [Supported Delphi versions](#)
- ▶ [Sending TAdvRichEditor formatted text by email](#)
- ▶ [How to copy formatted text from a TAdvRichEditor to a TPlannerItem](#)
- ▶ [How to copy contents from one TAdvRichEditor instance to another](#)





If you want to save or open your work the standard is .txt which is not a good idea to do because if you use any colours in the text you will loose that. You could do that as a .RTF (Rich format Text). So you best select all files. Than you will see all files that are available or possible.





Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

The meaning of Lorem Ipsum

As you've probably guessed, Lorem Ipsum is really nothing more than a filler text — also known as a dummy text. Web designers, graphic designers and printers use the text to get an impression of the final appearance of a website, brochure or book. If the real text is not yet available, this is a logical solution.

The term "lorem ipsum" comes from Latin and is only found in history for the first time around 1500. Not entirely coincidentally, this is precisely the period in which the printing press flourished in the Low Countries at that time. Printers then chose it because the letters and the character spacing in the text very well bring out the appearance of a font.

Lorem ipsum

dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Translation of Lorem Ipsum into English

Although Lorem Ipsum is originally associated with the Latin idiom, the term is linguistically incorrect. In fact, the text actually means nothing at all. The two words come from a scrambled version of a passage from Cicero's Finibus Bonorum et Malorum, in which he writes:

"Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit..."

The translation of this is:

"No one likes pain, seeks out the pain and wants to be in pain just for the pain itself..."

Coincidentally, a printer used this text, right up to the present day.

THE MEANING OF LOREM IPSUM

As you've probably guessed, Lorem Ipsum is really nothing more than a filler text — also known as a dummy text. Web designers, graphic designers and printers use the text to get an impression of the final appearance of a website, brochure or book. If the real text is not yet available, this is a logical solution.

The term "lorem ipsum" comes from Latin and is only found in history for the first time around 1500. Not entirely coincidentally, this is precisely the period in which the printing press flourished in the Low Countries at that time. Printers then chose it because the letters and the character spacing in the text very well bring out the appearance of a font.

Translation of Lorem Ipsum into English

Although Lorem Ipsum is originally associated with the Latin idiom, the term is linguistically incorrect. In fact, the text actually means nothing at all. The two words come from a scrambled version of a passage from Cicero's Finibus Bonorum et Malorum, in which he writes:

"Neque porro quisquam est qui dolorem ipsum quia dolor sit amet, consectetur, adipisci velit..."

The translation of this is:

"No one likes pain, seeks out the pain and wants to be in pain just for the pain itself..."



THE DELPHI COMPANY

-est 1998-

OS X Android iOS Windows

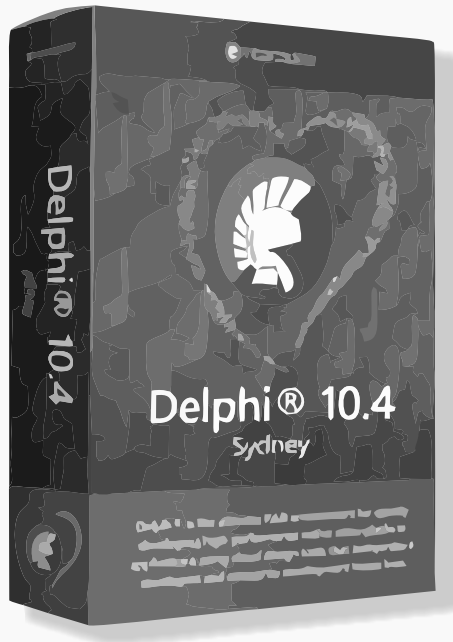


Four Platforms
One develop environment
One Expertise

Vier platforms
Eén ontwikkelomgeving
Eén expertise

DELPHI

www.delphicompany.nl
info@delphicompany.nl

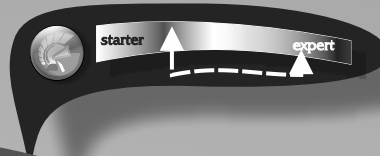


Delphi 10.4 Sydney Professional
Delphi €1.699,00

<https://www.barnsten.com/promotions/>



By Danny Wind



This series of articles is about writing your own web services server and client in Delphi. The approach of all articles is pragmatic. The first article introduced some of the concepts you need to know and shows you how to create and consume your own web service in Delphi with just the GET request. The second article showed you how to update the data in the web service and how to create in-memory storage for the web service. The third article showed how to consume and use your web service from both Delphi clients on Windows and from a web page with JavaScript. This fourth article is about deploying your web service to the Internet Information Services (IIS) server on Windows.

What are the deployment options for a web service written in Delphi? If you create a new web application in Delphi you have the following options in the wizard

Apache dynamic link module	An Apache module. Apache has support for HTTP and HTTPS. The current Apache 2.4 is supported on x64 Linux.
Stand-alone console application	A stand-alone WebBroker console application is a web server that has a text-only user interface. It supports HTTP using an Indy HTTP server component.
Stand-alone GUI application	A stand-alone WebBroker application is a web server that displays a form. It supports HTTP using an Indy HTTP server component.
ISAPI dynamic link library	An ISAPI library integrates with IIS. IIS has support for HTTP and HTTPS.
CGI stand-alone executable	A CGI executable integrates with a web server. Note that CGI is typically slower and more difficult to debug than ISAPI or an Apache module.

The stand-alone options do not use the IIS or Apache web server platform for receiving and handling HTTP requests, but instead rely on the Indy HTTP web server component. This works reliably and can even support HTTPS with OpenSSL, although the wizard suggests otherwise.

However it is less suited for server environments exposed to the public internet, which need to be maintained, updated and monitored to remain secure. Stand-alone is more suited for static installations hidden inside a VPN or for embedded industrial installations.

One stand-alone option is missing from this list; it's also possible to manually create a Windows Service application and embed the web service into it. This uses the same Indy web server as the other Stand-alone options.



Of these options ISAPI and Apache make the most sense for regular deployment. Both ISAPI and Apache can be run on default web server installations of Windows and Linux. You can just order a Virtual Private Server with any hosting provider and have your web service up and running on the internet within a couple of hours. Of course you can also use your own server.

Because IIS and Apache are widely supported you can set up monitoring on the web server with one of the many available tools or delegate maintenance and administration to a third-party system administrator. System administrators are usually well versed in handling security, updates and monitoring of IIS and Apache installations. Both IIS and Apache also support HTTPS encryption, made easy through Let's Encrypt certificates or, if you need, higher level certificates from one of the other Certificate Authorities.

In this article we will focus on using IIS and ISAPI for our web service and we will deploy it to a development machine.

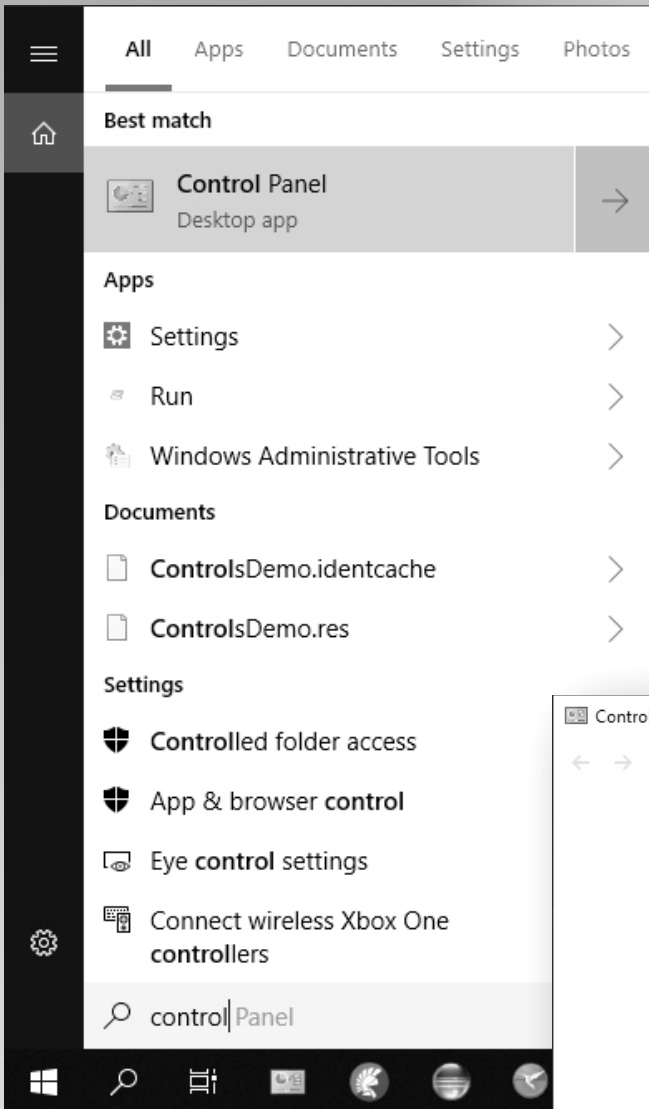
We start by enabling Internet Information Server on a plain Windows-10 machine. This could be the same (virtual) machine where you have your Delphi development installed, but any other Windows-10 machine or virtualized environment is fine as well.

If you are running a Windows Server environment you probably already have Internet Information Services installed through your server roles administration panel. In that case you can skip ahead towards creating the ISAPI itself in Delphi at step 7.

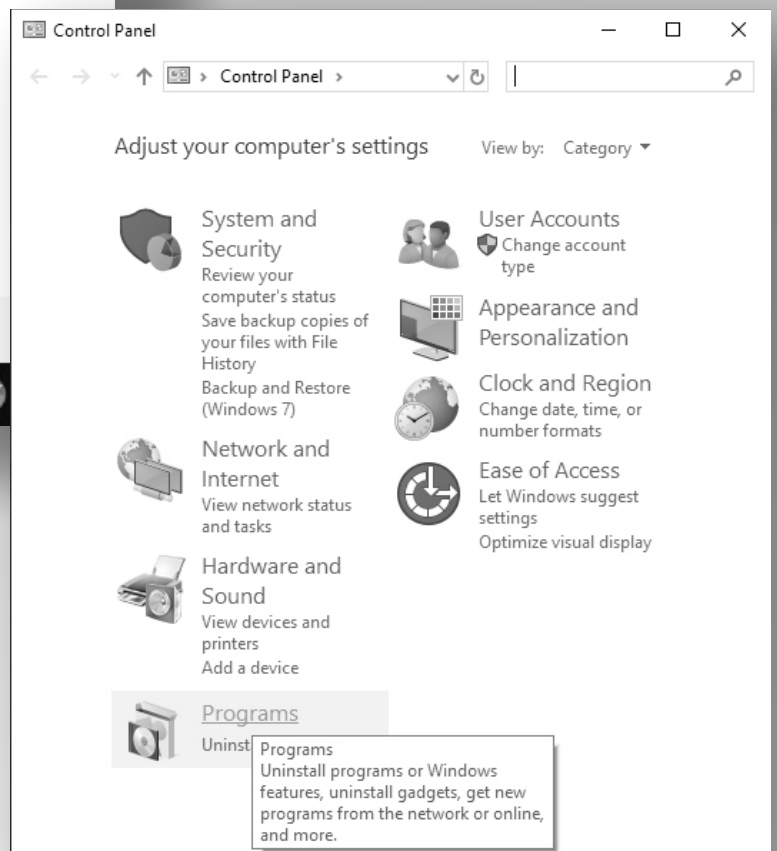
For those of us who are using plain Windows-10 we install the Internet Information Services server through the Control Panel with Programs and Features



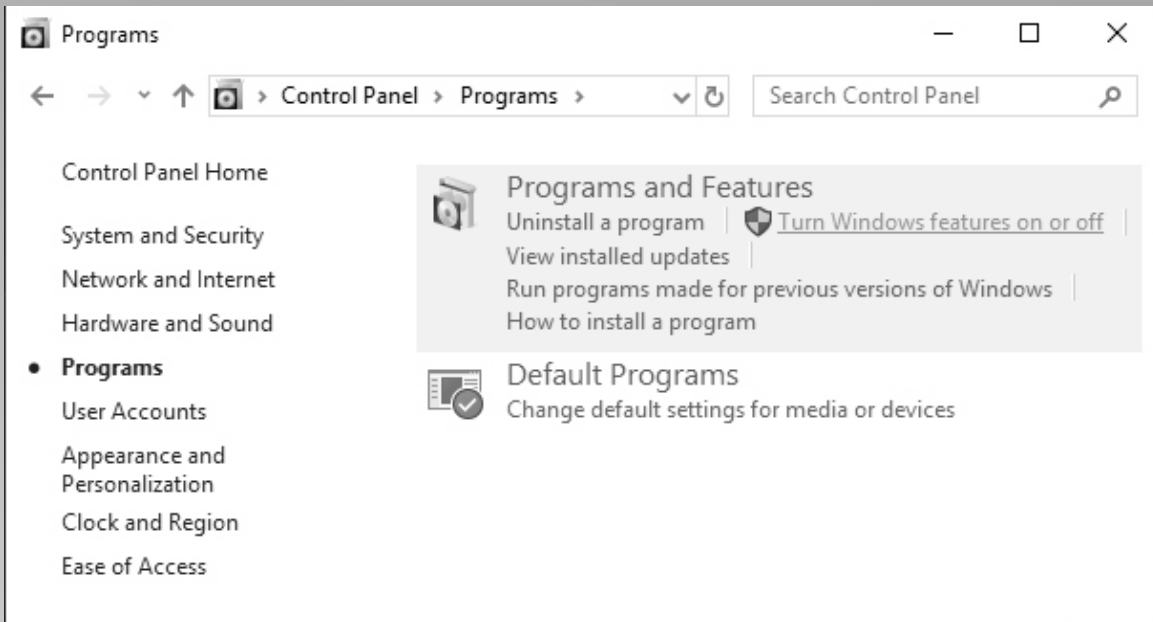
- 1 Click on the search icon next to the start menu and type "Control". Note that even if you are using a non English language version of Windows this will still lead you to the Control Panel Desktop app.



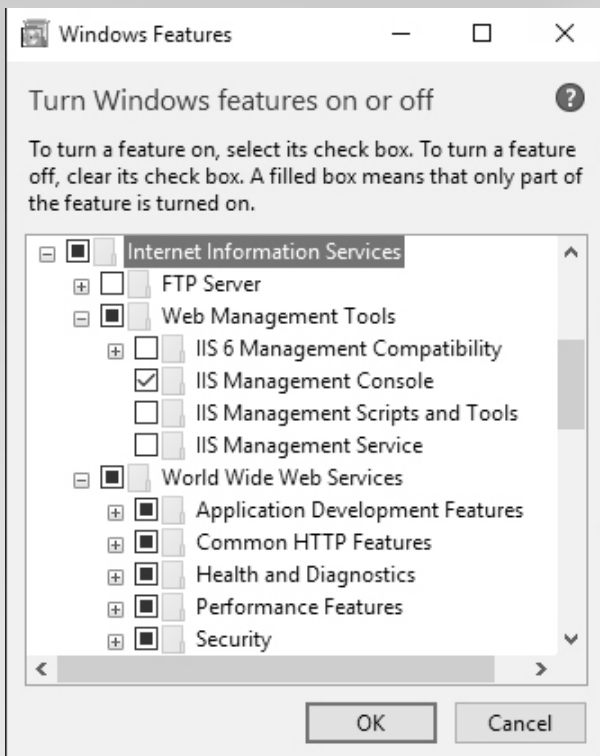
- 2 In the Control Panel select "Programs"



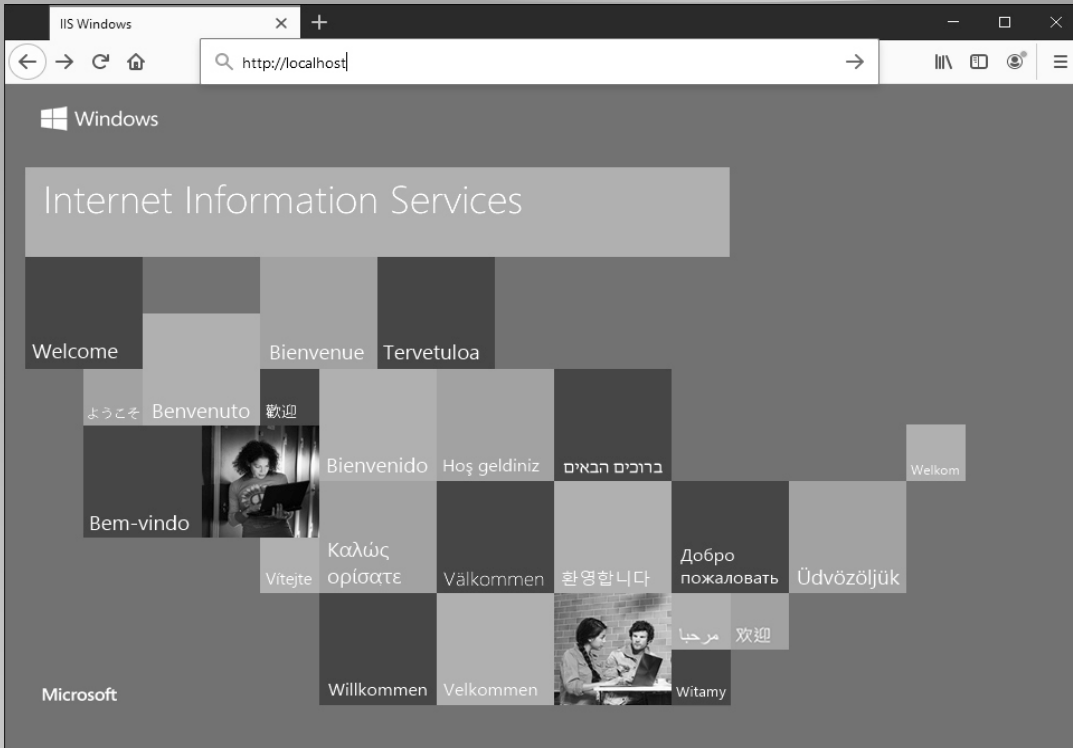
3 Under Programs click on "Turn Windows features on or off"



4 Inside the Turn Windows Features on or off selection box just turn Internet Information Services on. This will automatically select almost all of the features that we need. We will do additional configuration steps later on



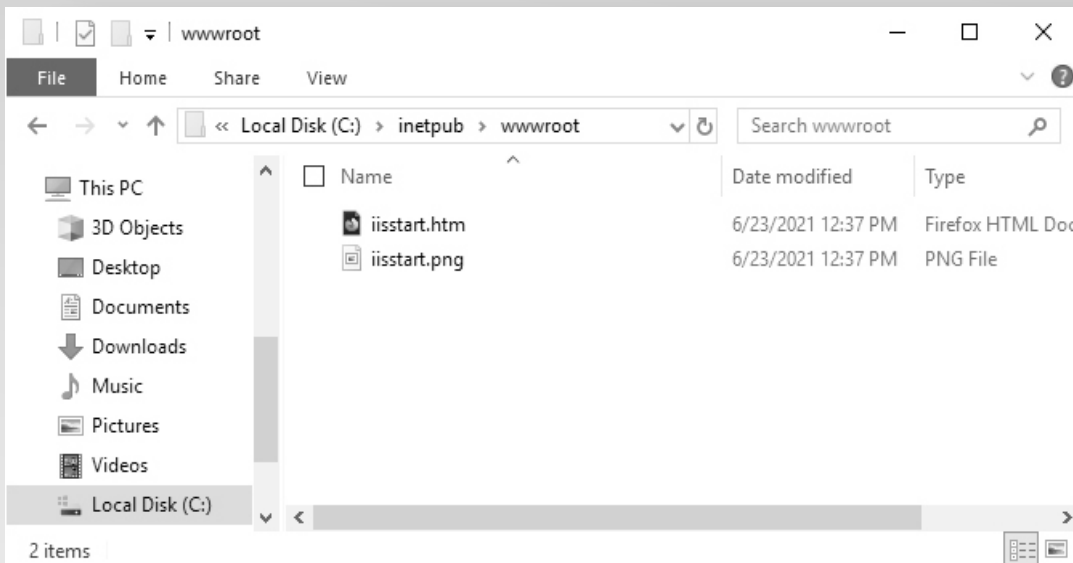
5 Windows will now install IIS. We can check if the install was successful by navigating to localhost in our browser `http://localhost`



6 We now have an IIS server running on our Windows-10 machine

The Internet Information Services server, or IIS server, can be used to serve several types of web site and web service content. The most apparent one is the html page that we just requested by visiting localhost. It's simply a file located in the IIS root folder.

`C:\inetpub\wwwroot`



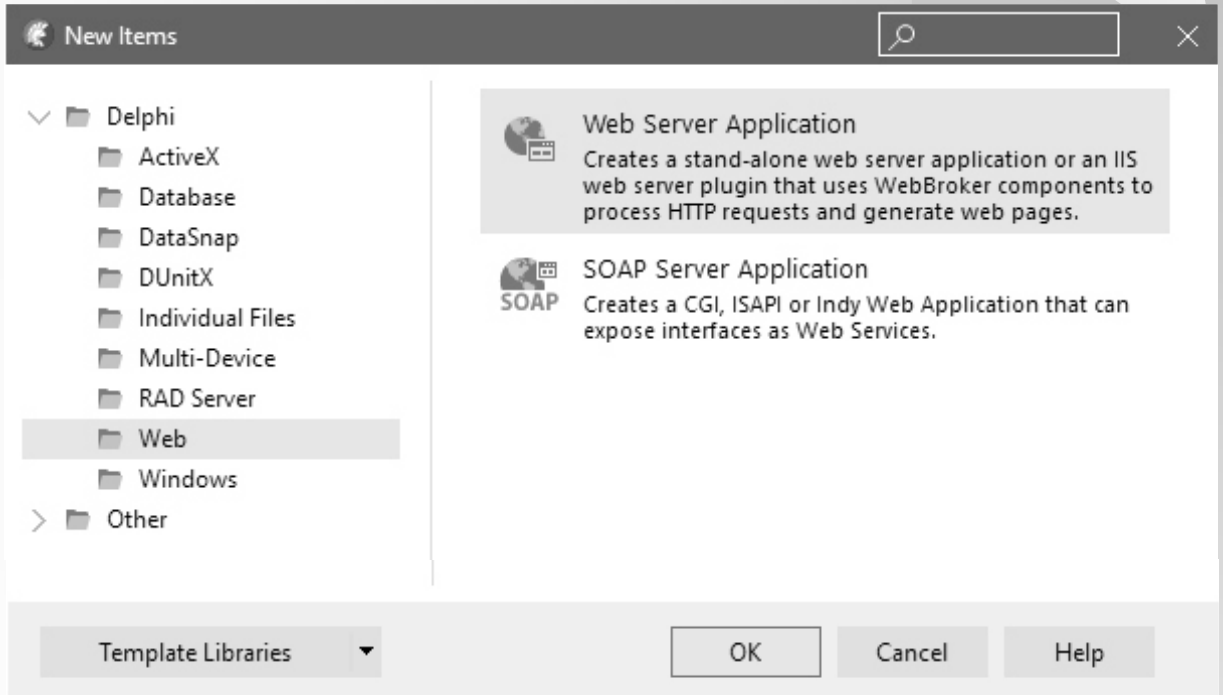
Any other htm file that you place at this location will be available in the IIS web server as a web page.

Please note that with installing IIS on your machine it also opened port 80 (HTTP) and port 443 (HTTPS) for external connections. At the end of the article we'll show you how to easily block or allow traffic to these ports in the Windows Firewall, as you may not want IIS available to everyone all the time, especially if you are on a public internet location, such as an airport.

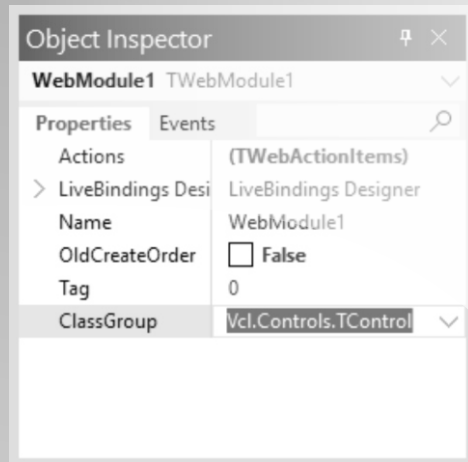
One of the things that can be hosted by an IIS server is an ISAPI extension. ISAPI stands for Internet Server Application Programming Interface, and it allows us to create a dynamic link library (DLL) that adds to the functionality of the IIS server.

Our next step is to create an ISAPI dll in Delphi. The code from the web service we wrote in the previous article will be shared and embedded within this ISAPI dll.

- 7 Create a new temporary Project Folder, eg \ISAPI
- 8 In Delphi create a new Project with File | New - Other and under Delphi - Web choose Web Server Application

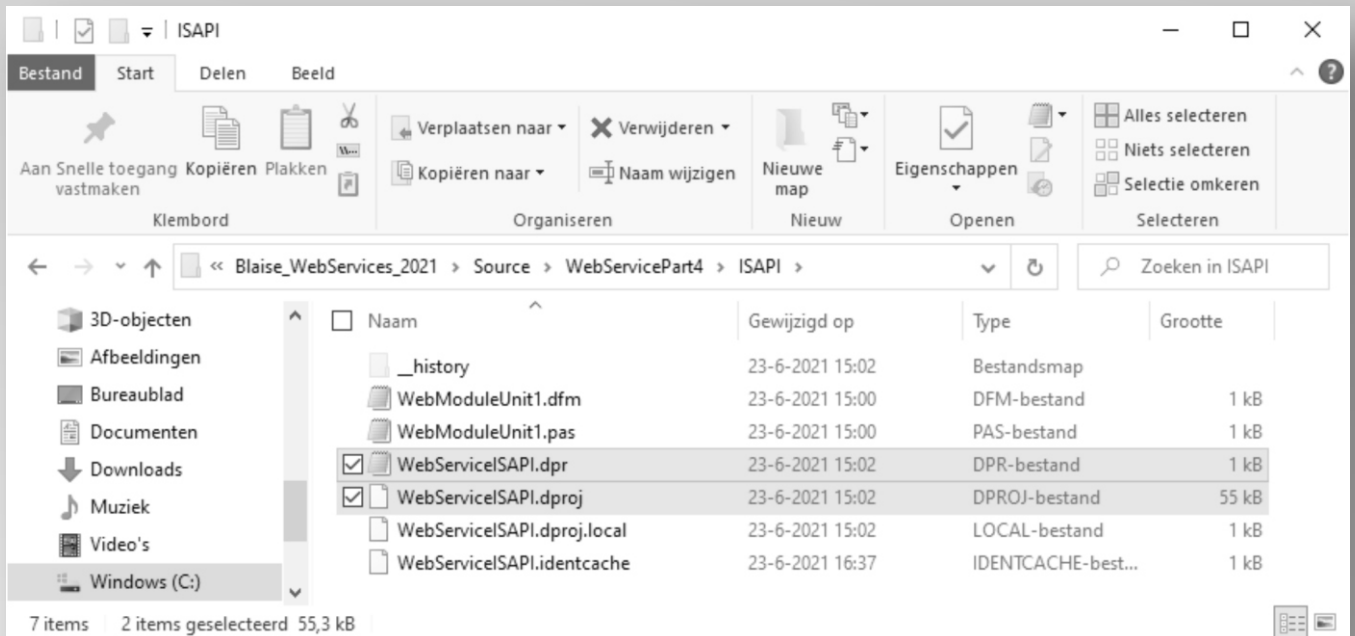


- 9 In the wizard choose Windows and then ISAPI Dynamic Link Library
- 10 Save the project as WebServiceISAPI in the temporary folder \ISAPI
- 11 Now take a look at the project manager. Notice how the ISAPI project has the same structure as the previously created WebServiceServerGUI project. It is just a project file and a WebModuleUnit
- 12 There is a difference between the two WebModuleUnit files. The one from WebServiceServerGUI has the ClassGroup property set to the Vcl framework , while the one from the WebServiceISAPI has the ClassGroup property set to the framework neutral setting. We will change that after we merge the two projects

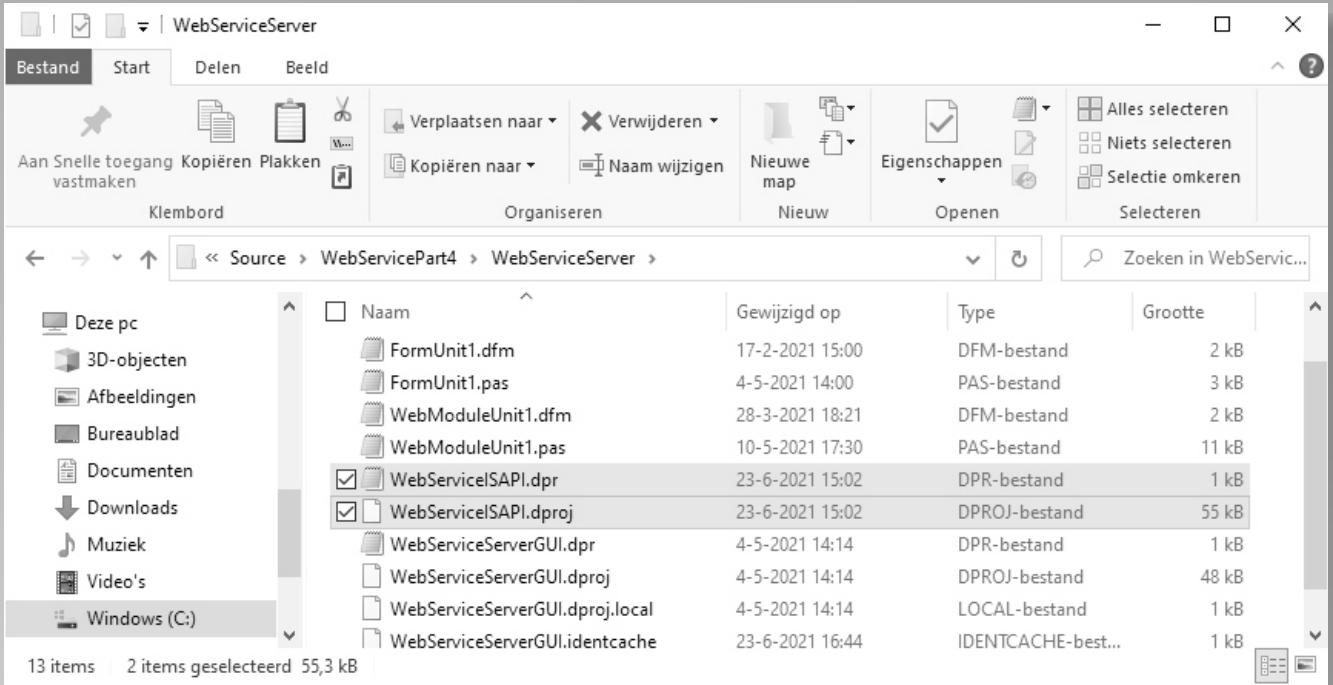


In order to merge the new
13 WebServiceISAPI with the existing WebServiceServerGUI we can simply copy the two WebServiceISAPI project files over to the WebServiceServerGUI directory. By doing that we let the new ISAPI project use the existing WebModuleUnit from the WebServiceServerGUI as they are in the same folder

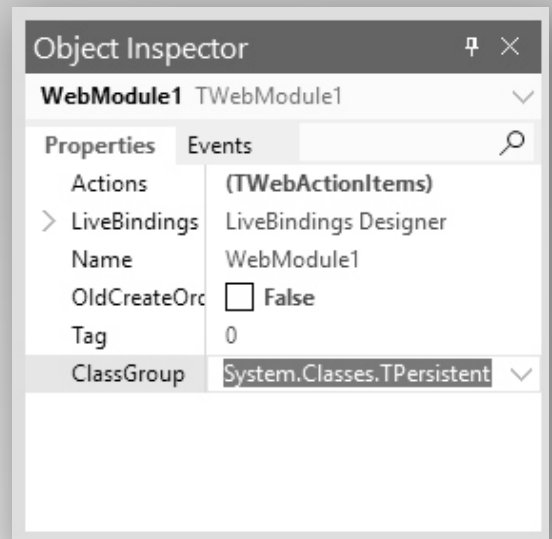
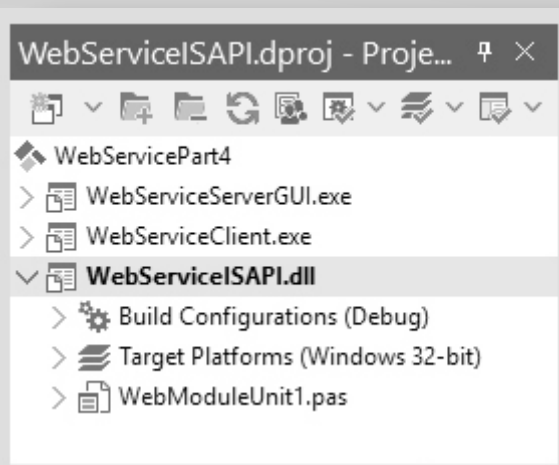
```
copy \ISAPI\WebServiceISAPI.dpr \WebServiceServerGUI
copy \ISAPI\WebServiceISAPI.dproj \WebServiceServerGUI
```



14 After copying these files over to the existing WebService project folder



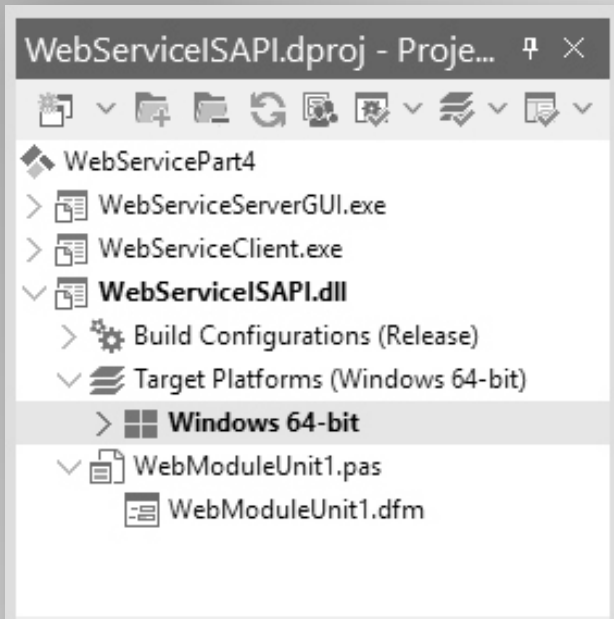
15 we can now add this new project to the Project Group with a right-click in the Project Manager and Add existing Project



16 Now open the WebServiceISAPI project from the Project Manager and open the WebModuleUnit. As this WebModuleUnit is now shared with the VCL WebServiceServerGUI we need to change the property ClassGroup to the framework neutral System.Classes.TPersistent



- 17 We also modify the target platform for this project to 64-bit, as ISAPI extensions in Windows-10 are by default 64-bit. Right click on the project in the Project manager and Add the Windows-64 bit platform



- 18 We do not need the 32-bit target platform for the ISAPI dll so you can remove it
19 Now build the WebServiceISAPI dll to test if merging the projects was successful

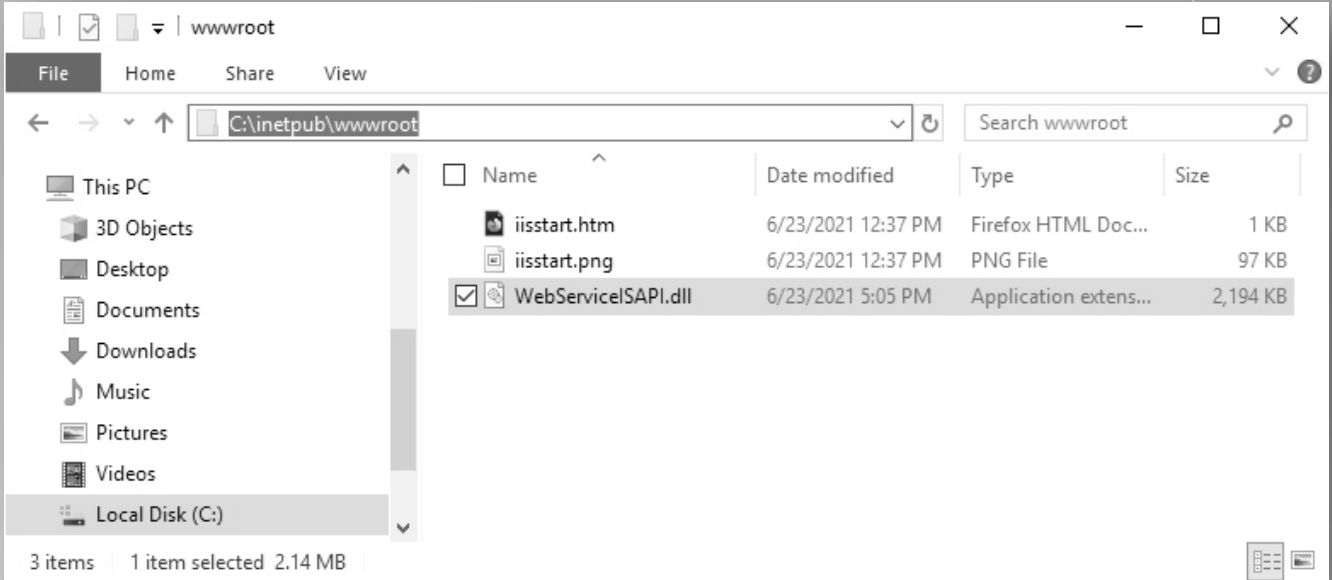
By merging the files from the VCL standalone application `WebServiceServerGUI` with the `WebServiceISAPI` dll project we can just develop and debug our web service with the VCL application, and then deploy it with a build of the `WebServiceISAPI` dll. It's much easier to debug a VCL application than debugging an ISAPI dll. Because the VCL version of our web service uses port 8080 and IIS by default uses port 80 we can even run them side by side.

Installing the ISAPI into IIS requires copying the dll file over to a directory that the IIS process can access as well as configuring IIS to load the ISAPI dll as an extension. For this article we will just use the `DefaultAppPool` and the `wwwroot` directory to run our ISAPI dll, because we are deploying to a development machine. In a production environment you may want to change the IIS configuration and create your own secured `AppPool` and website.

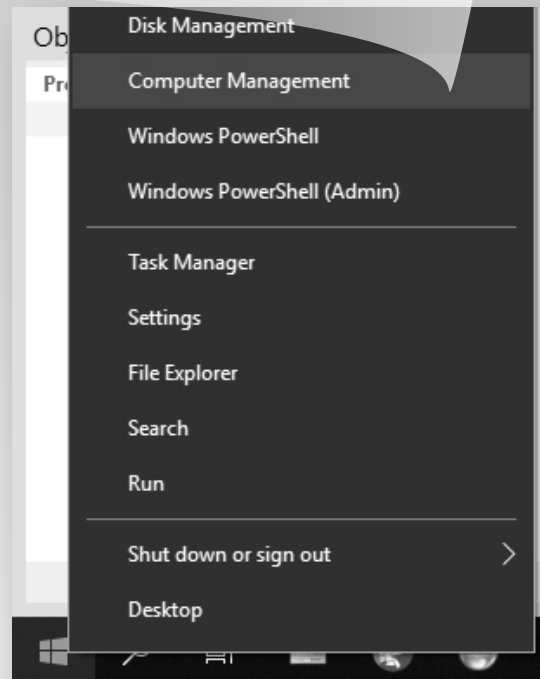
An Application Pool is a pool of resources within which an IIS application or website is run and which can be configured for the websites and applications that it governs. IIS comes pre-configured with the `DefaultAppPool` that services the default website at `wwwroot`.



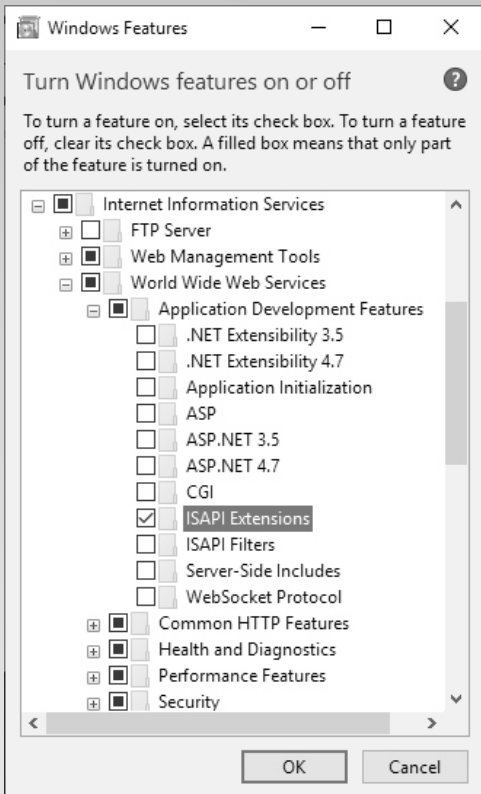
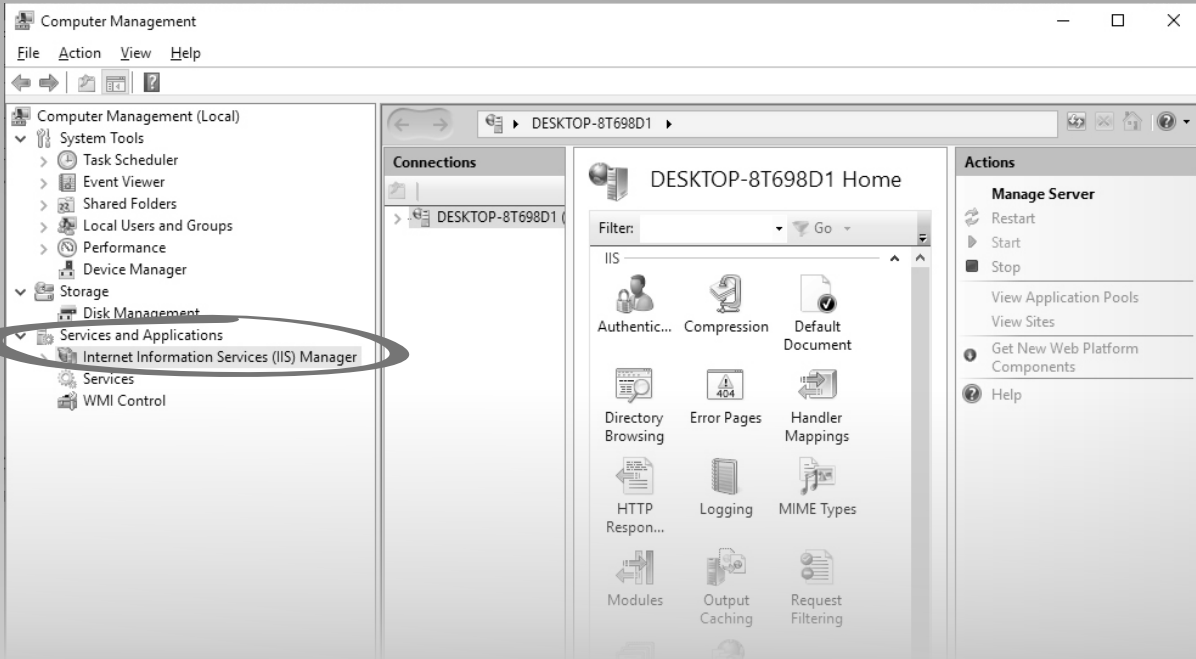
- 20 Copy the `WebServiceISAPI.dll` to the `inetpub\wwwroot` folder. Windows may block copying from network locations to the protected `inetpub` folder, so you may need to copy it to an intermediate local folder (`C:\TEMP`) first
copy `WebServiceISAPI.dll` `C:\inetpub\wwwroot`



- 21 Next we need to configure IIS to allow execution of this ISAPI extension
- 22 Open Computer Management with a right-click on the start button



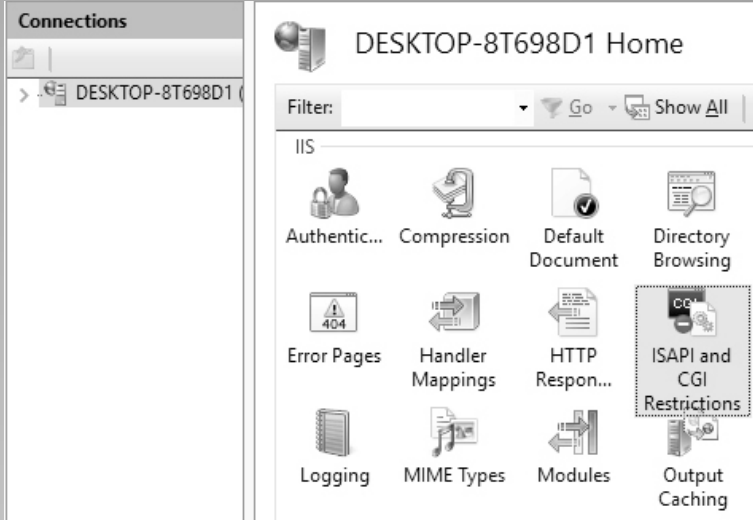
- 23 In Computer Management select the Internet Information Services manager under Services and Applications



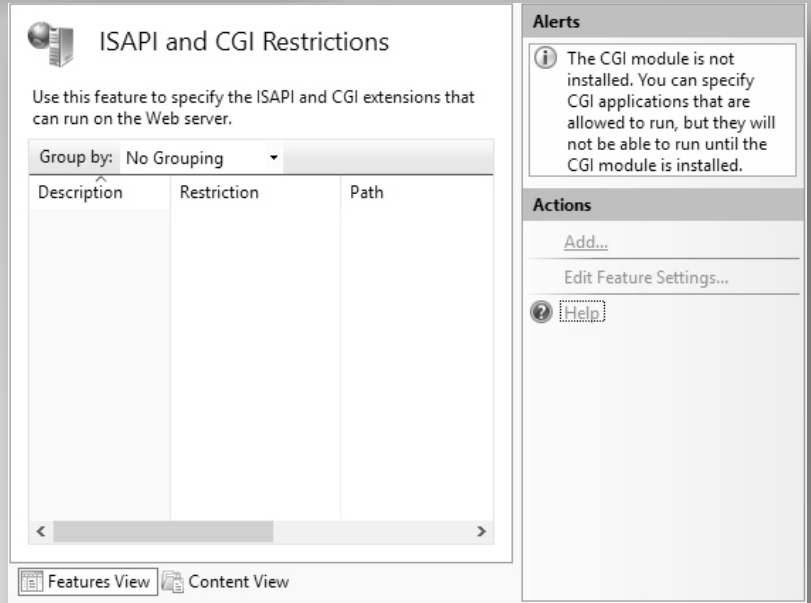
- 24 We need to use ISAPI/CGI Restrictions to configure our ISAPI, but as you can see the icon for that is not available. This is because in newer Windows and IIS versions ISAPI extensions are by default disabled and we need to enable it first.
- 25 Go back to the Control Panel as we did earlier and navigate to Turn Windows Features on and off. In the Internet Information Services branch we enable ISAPI extensions



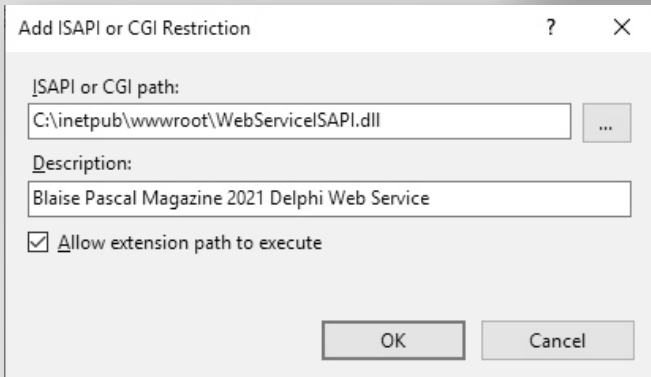
26 After enabling this we should see ISAPI/CGI Restrictions turn up in the IIS manager



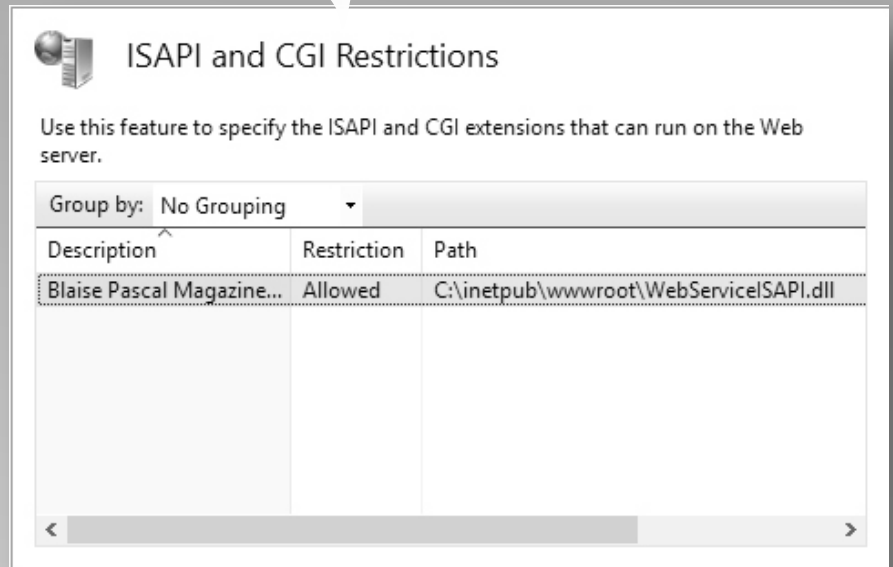
27 Open the feature and use the Add Action on the right to Add our ISAPI dll to IIS



28 We configure only this single ISAPI dll and allow it to be executed

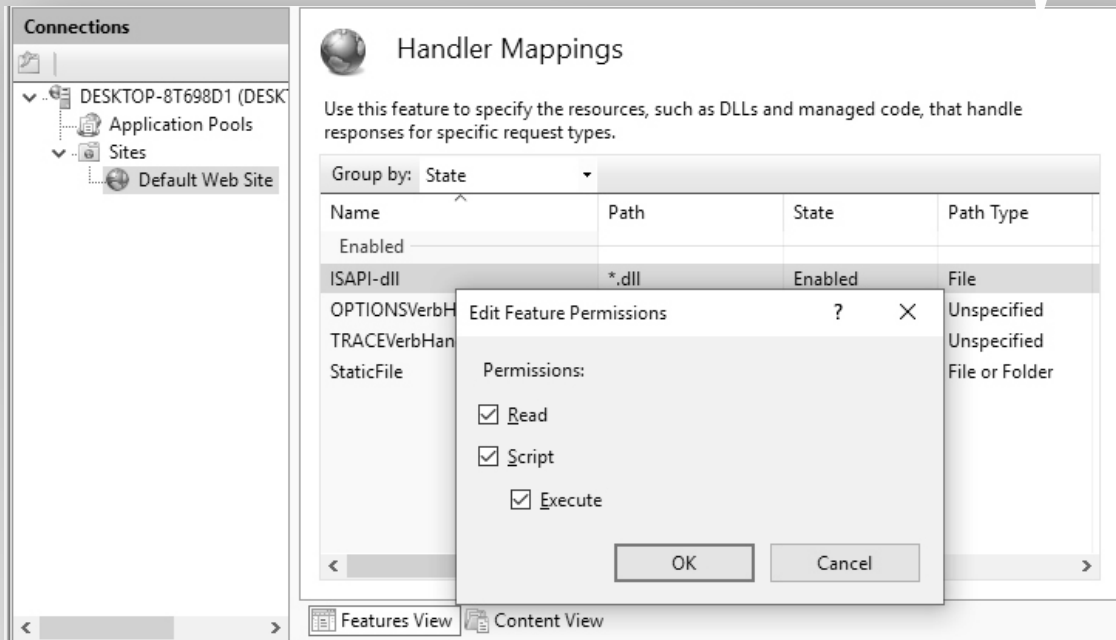


29 After adding the ISAPI dll you'll find it in the list

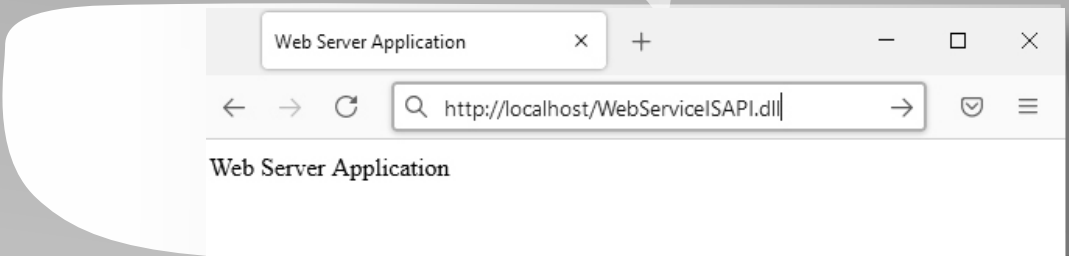


30 After adding the specific ISAPI to the allowed list we also need to configure handler mappings for wwwroot

31 Select the Default Web Site and open the Handler Mappings with the icon. Then right click on the ISAPI dll line and Edit Feature Permissions and enable execution



- 32 By pre-selecting the Default Web Site we can configure Handler Mappings just on this website. Handler mappings on server-level will still have ISAPI-dll as disabled
- 33 Test if this works by requesting the WebServiceISAPI.dll in a browser `http://localhost/WebServiceISAPI.dll`



The web service is now up and running as an ISAPI extension. Because the web service uses transient in-memory storage and ISAPI extensions are unloaded by IIS when not in use, our web service currently suffers from forgetfulness. We need to reconfigure IIS to hold our web service in memory for a longer time period to be able to use it properly.

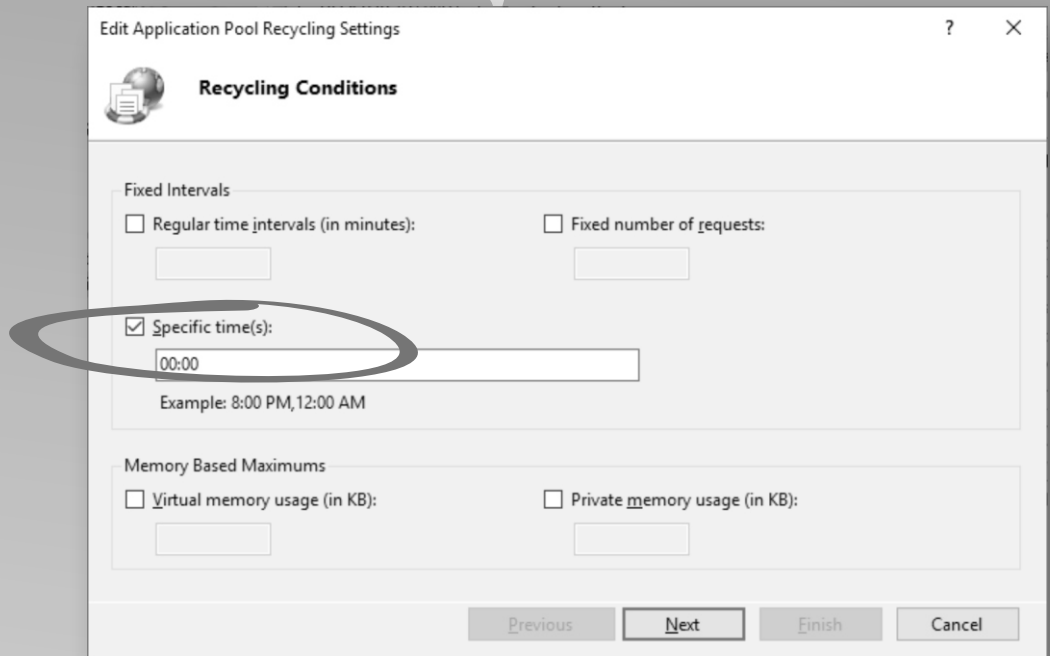
- 34 Select the Default Application Pool and open Recycling on the right-hand side under Edit Application Pool

A screenshot of the IIS Management Console. The left pane shows the "Connections" tree with "Application Pools" selected. The main pane is titled "Application Pools" and contains a table with the following data:

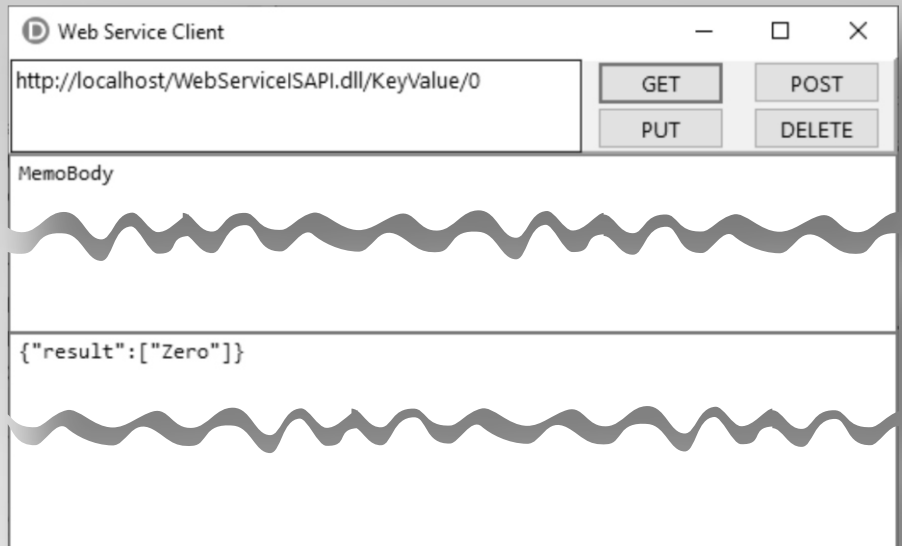
Name	Status	.NET CLR V...	Managed Pipel...	Id
DefaultAppPool	Started	v4.0	Integrated	Aj

The right pane shows "Actions" for the selected application pool, including "Add Application Pool...", "Set Application Pool Defaults...", "Application Pool Tasks" (Start, Stop, Recycle...), and "Edit Application Pool" (Basic Settings..., Recycling...).

- 35 Modify the Recycle time and set it to 00:00 specific time



- 36 The IIS manager will translate this to the local time format, so it may end up as 12:00 AM depending on your international settings. This is fine, its intended to Recycle at midnight, once every 24 hours
- 37 Test if it all works by reconfiguring the URL for the WebServiceClient application in the EditURL text `http://localhost/WebServiceISAPI.dll/KeyValue/0`



- 38 You can now POST and PUT new values with the Delphi web service client. If you close the Delphi client and re-open a new web browser and request the POSTed value it will still be there. If this does not work check the Recycling settings of the Default App Pool.



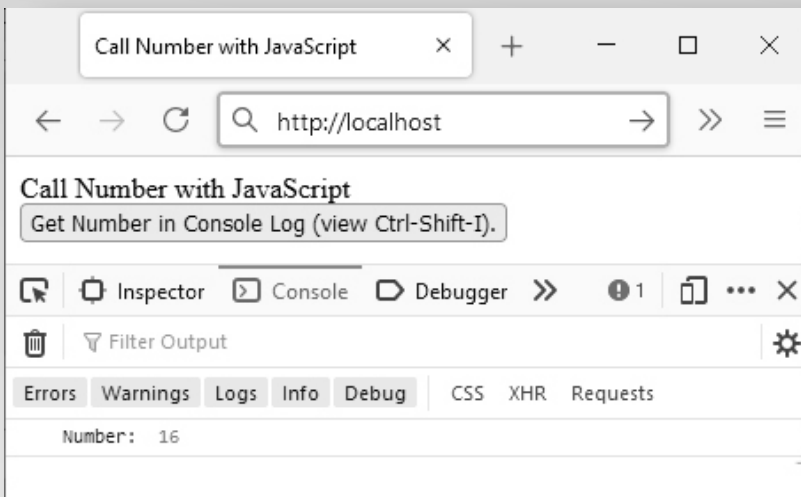
Our web service is now up and running on an IIS web server. The IIS web server can also be used to serve static HTML and JavaScript pages. In our previous article we added a /JavaScript Web Action Handler that returned some HTML and JavaScript code that in turn used the web service itself to return a number through a REST request on localhost\Number. It's a bit of a convoluted path. Now that we are running on a web server it makes much more sense to save this HTML with JavaScript page to a regular static html file in the default website and let the IIS web server handle the request.

39 Copy the HTML and JavaScript code from the /JavaScript Web Action Handler to Notepad and change the embedded url to our new WebServiceISAPI. Don't forget to remove the additional single quotes or just use the below text

```
<html>
<head><title>Call Number with JavaScript</title></head>
<body>Call Number with JavaScript <button onclick="getNumber()">Get Number in
Console Log (view Ctrl-Shift-I).</button>
<script type="text/javascript">
function getNumber() {
let url = 'http://localhost/WebServiceISAPI.dll/Number';
fetch(url).then(resp=> resp.json()).then(j=> console.log('\nNumber: ', j));
}
</script>
</html>
```

Note that later on you may want to change the reference of localhost to your external IP or your DNS registered domain name, as localhost only makes sense from within your development machine. For now we need it to be localhost.

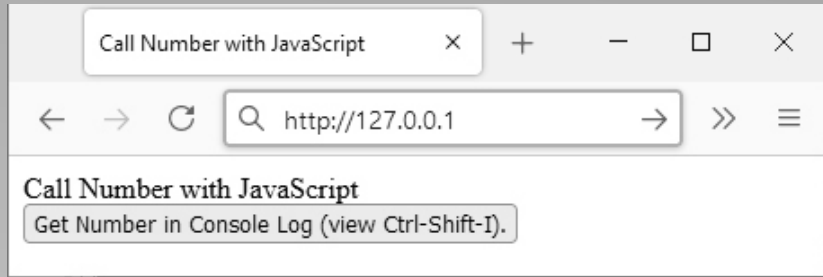
40 Save this file as default.htm in a folder \HTML and copy it to the IIS wwwroot folder
C:\inetpub\wwwroot\default.htm



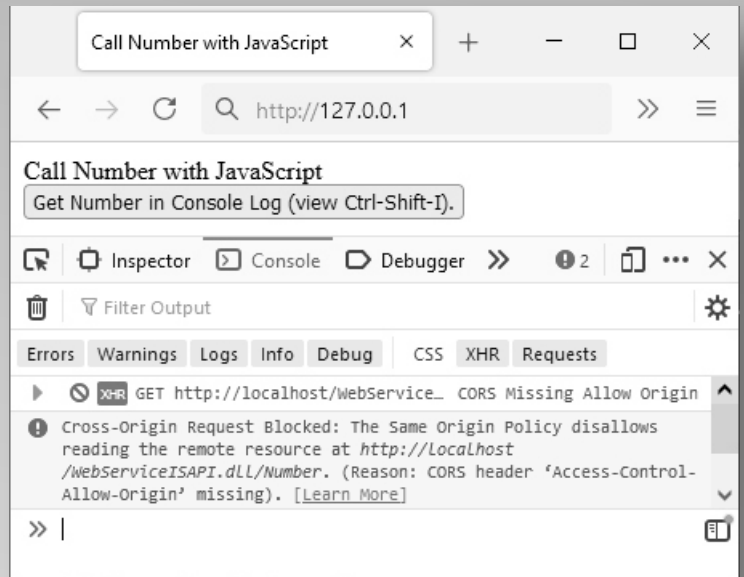
- 41 IIS will prefer loading default.htm instead of iisstart.htm, and the default HTML page will be displayed when visiting localhost
- 42 Test if it works by opening the localhost url in a browser from inside your development machine. Remember to open the Console Log in Firefox or Chrome/Edge with **Ctrl-Shift-I**



- 43 This all looks fine and the button works and requests a new Number from the new ISAPI web service
- 44 What if instead of using localhost we would use the url `http://127.0.0.1` ?



- 45 The page opens just fine. But if you now click on the Get Number button we get an error in the Console

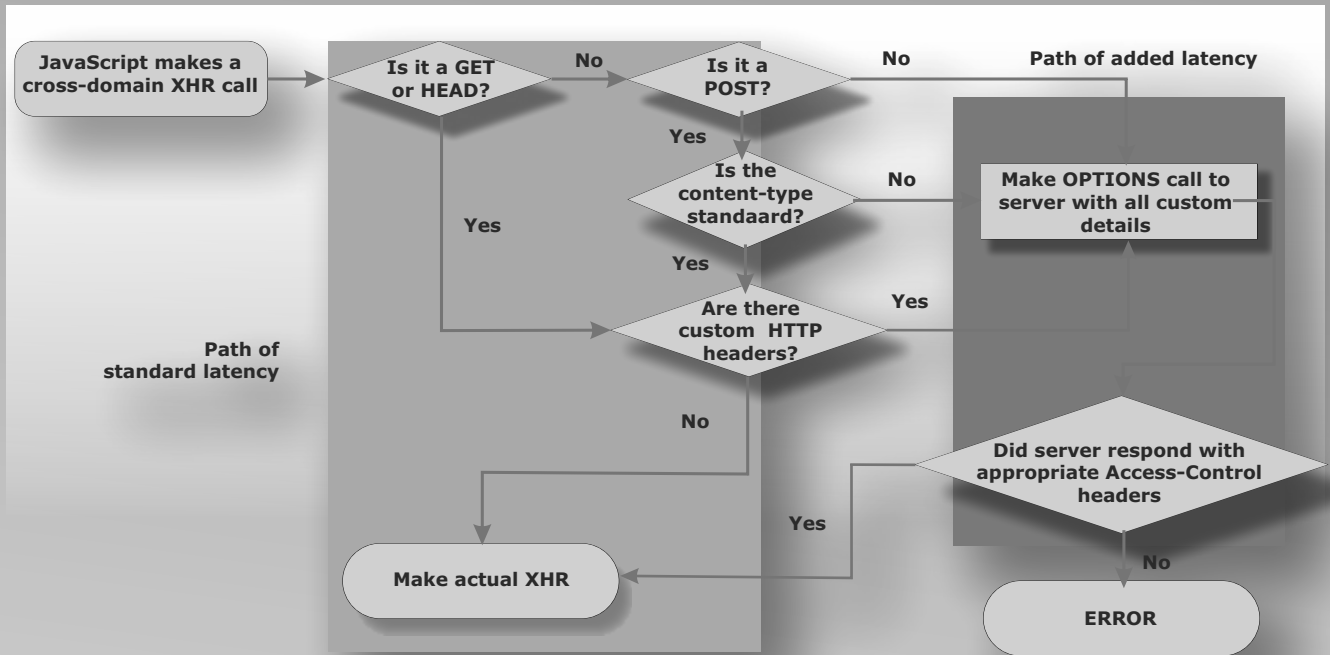


- 46 Let's investigate why this is happening
We visit the default.htm web page with the IP address 127.0.0.1. When we then click on the Get Number button, the JavaScript performs a XHR GET request from the web page at address 127.0.0.1 to a web service at localhost. We know that these are one and the same, but in fact, the localhost hostname and 127.0.0.1 are considered to be different by the browser, so the browser handles it as a cross-site event. The browser therefore asks the web service at localhost if this cross-site request should be allowed. Because the web service does not indicate that such a Cross-Origin Request is allowed, the request is blocked by the browser.

CORS Cross Origin Resource Sharing.
A web service can indicate access from other domains is allowed, by responding with 'Access-Control-Allow-Origin' in the HTTP header.
https://en.wikipedia.org/wiki/Cross-origin_resource_sharing



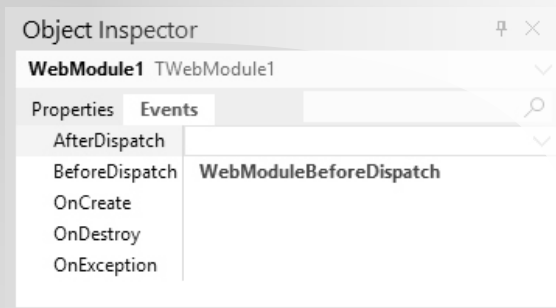
The following image from wikipedia on Cross Origin Resource Sharing illustrates this path.



"Path of an XMLHttpRequest (XHR) through CORS." by Bluesmoon is licensed under CC-BY-SA 4.0

So what we need to do is have our web service allow **CORS** by adding the appropriate `Access-Control*` headers.

- 47 Open the **WebModule** unit of the project and add a **BeforeDispatch** event-handler to the WebModule



- 48 Add **CORS** code to the **WebModule BeforeDispatch** event-handler, to indicate that calling our web service from other domains is OK.



```
procedure TWebModule1.WebModuleBeforeDispatch(Sender: TObject;  
Request: TWebRequest; Response: TWebResponse; var Handled: Boolean);  
begin  
  {Report back to the caller/browser that we allow Cross Origin Resource Sharing (CORS) for all calling domains}  
  Response.SetCustomHeader('Access-Control-Allow-Origin','*');  
  
  if Trim(Request.GetFieldByName(  
    'Access-Control-Request-Headers')) <> '' then  
    begin  
      Response.SetCustomHeader( 'Access-Control-Allow-Headers',  
        Request.GetFieldByName('Access-Control-Request-Headers'));  
      Handled := True;  
    end;  
end;
```

The code above is the most permissive

response you can have for a web service.

If someone initiates a CORS preflight request, all of the Access-Control-Request-Headers are allowed by simply copying them over to the Access-Control-Allow-Headers in the response. If a client asks if POST is supported, we indicate that it is, same for any of the other HTTP methods. We could also limit this a bit, to indicate that we support just a limited subset of HTTP methods by using Access-Control-Allow-Methods: POST, PUT, GET, DELETE.

In the same code snippet, by setting Access-Control-Allow-Origin to '*', we are allowing CORS calls from all origins. You could limit this to be less permissive by setting specific domain names in Access-Control-Allow-Origin.

If you start using your Delphi web service from other client platforms or browsers you may also run into caching issues. To prevent a web service consumer from caching previously retrieved values for the idempotent and cacheable HTTP GET command, you can set some headers in the web service to inform the web service consumer that it should not cache anything.

49

Add these Custom Headers to the code in the WebModuleBeforeDispatch handler to request no-caching

```
{ Set additional headers to ask client-side to not cache locally  
Cache-Control=no-cache, no-store, must-revalidate  
Pragma=no-cache  
Expires=0 }
```

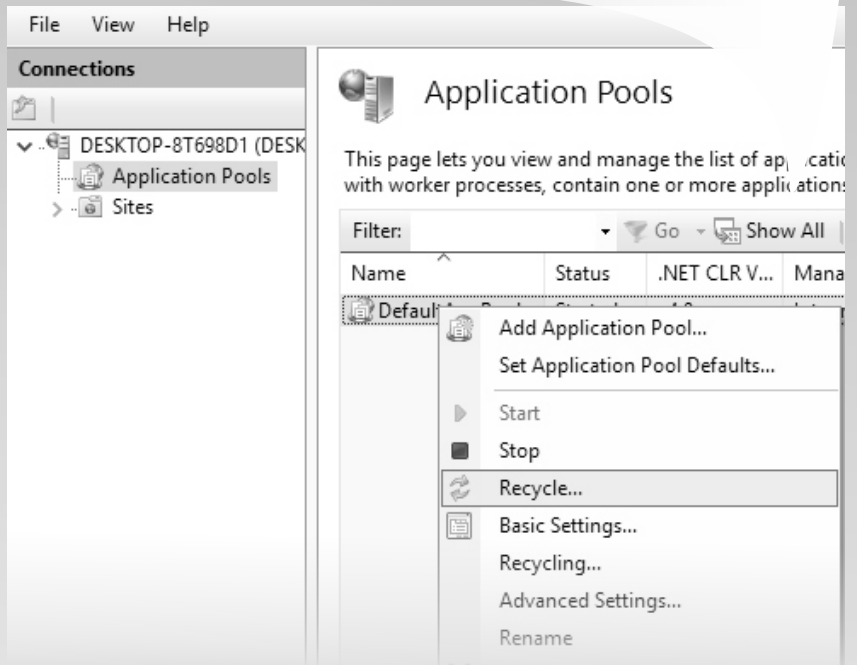
```
Response.CustomHeaders.AddPair('Cache', 'no-cache, no-store, must-revalidate');  
Response.CustomHeaders.AddPair('Pragma', 'no-cache');  
Response.CustomHeaders.AddPair('Expires', '0');
```



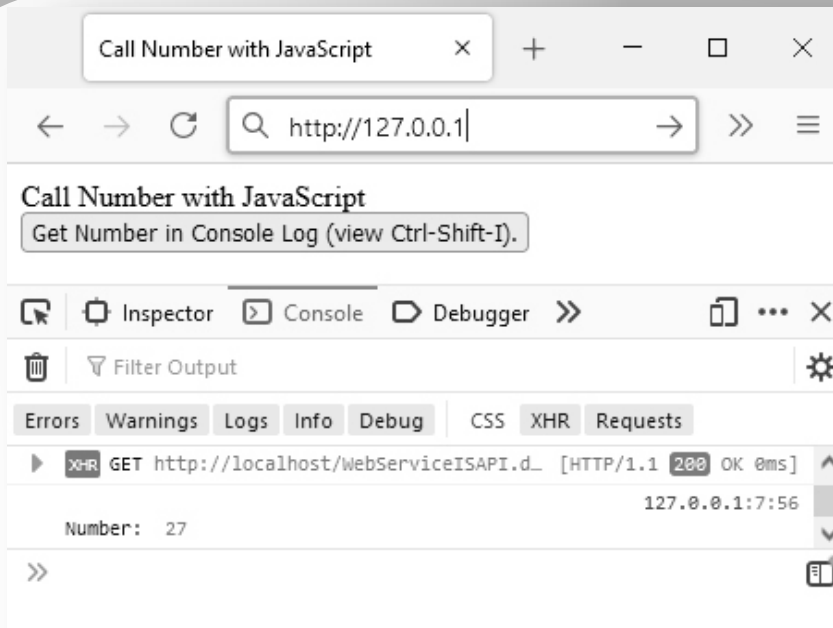
These added headers ask other web service consumers to always retrieve new values for each GET request. This is not entirely foolproof though. You may run into caching proxy servers that choose to ignore these headers, in which case you would need to take additional steps. One of the tricks commonly used is to add a dummy parameter to the URL that the client then changes with each request. This dummy parameter can be ignored by the web services server, but any in-between caching proxy will see this as a completely new request and not serve the result from its cache. One example:

```
{This URL is different}  
http://localhost:8080/KeyValue?key=0&unique=random42  
{each time}  
http://localhost:8080/KeyValue?key=0&unique=random84  
{but gets the same value from the key value store}
```

- 50 We are now ready to deploy the next version of our ISAPI dll.
To update the dll, recompile the ISAPI library in Delphi and perform the following steps to replace the existing ISAPI dll
- 51 Open the IIS manager by using the **Search** field next to the Start button. Search for "inetmgr"
- 52 Open the Application Pools setting, select the Default Application Pool, right-click and choose Recycle.
By manually recycling the Application Pool of the ISAPI dll it will be unloaded from memory and you will be able to replace the dll file on disk



- 53 Then using the File Explorer remove the existing `WebServiceISAPI.dll` from `wwwroot` and copy the new version of this dll to the same location. If you are unable to replace the dll it may still be in use, you can release it by stopping and starting the IIS web server.
- 54 If we now try it again with `http://127.0.0.1` it works OK

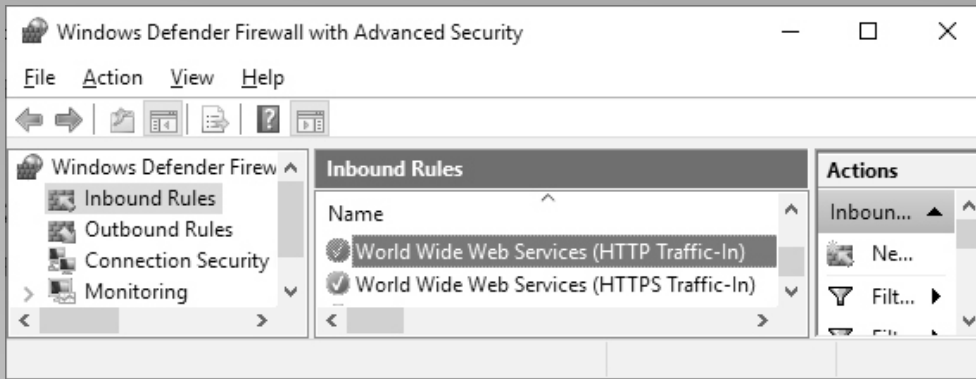


With these modifications to the web service CORS preflight our web service will be accessible from external domains. Note that the localhost reference in the JavaScript snippet only makes sense from within the development machine. If you want to access the default.htm from outside of the development machine you need to change it to the external IP or hostname. For a production machine you'd change it to the domain name.

The web service is now running under IIS and the coding is done, but there are some additional tips and hints that I'd like to share.

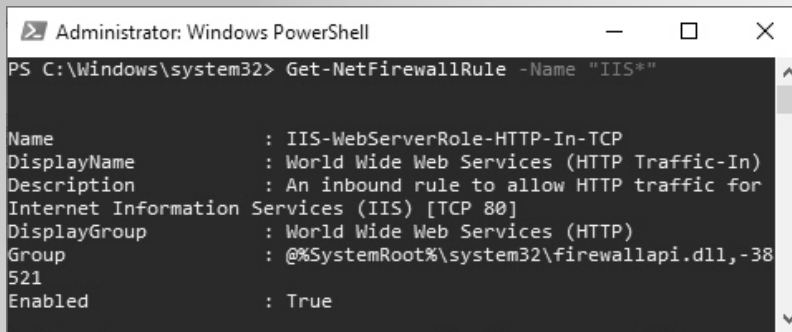
When we installed IIS on the development machine it also opened up port 80 (HTTP) and port 443 (HTTPS) for all types of networks (Private, Public, Domain). When you visit a location with public internet, such as an airport, you may not want to have these IIS ports open. Of course you can easily change these firewall rules, and close these ports, using the Advanced Firewall configuration.



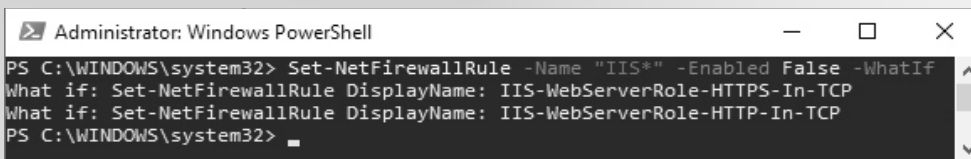


But this does require some mouse clicks and there is an easier way of doing this using PowerShell. Right-click on the start menu and start Windows PowerShell (Admin).

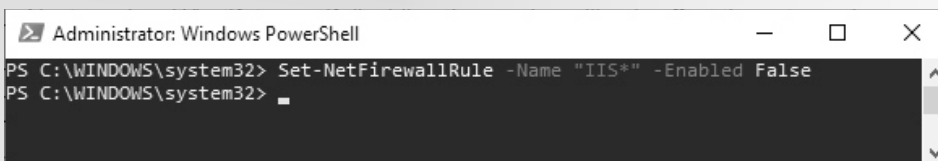
- 55 Get a list of the two firewall rules for IIS with this command
`Get-NetFirewallRule -Name "IIS*"`



- 56 Verify that you see just two rules, for port 80 and 443, both for IIS
- 57 Next we do a WhatIf, to see if disabling these rules will only affect these two rules
`Set-NetFirewallRule -Name "IIS*" -Enabled False -WhatIf`



- 58 If that looks good then execute the actual disable command
`Set-NetFirewallRule -Name "IIS*" -Enabled False`



- 59 Now test if IIS can still be reached on your machine from another machine using your external IP address. It should now be blocked. Note that your localhost will remain operational, as it's only blocking external connections.
- 60 Unblocking is easy as well, just use the `-Enabled True` command
`Set-NetFirewallRule -Name "IIS*" -Enabled True`

```
Administrator: Windows PowerShell
PS C:\WINDOWS\system32> Set-NetFirewallRule -Name "IIS*" -Enabled True
PS C:\WINDOWS\system32> █
```

- 61 And IIS can again be reached from other machines.

While you are actively developing and testing with Delphi on IIS on your development machine or laptop you can easily enable the firewall rule for full access. In all other cases you can disable it and have IIS safely hidden behind the firewall.

In this deployment we chose to host our ISAPI dll in the Default Website and the Default Application Pool, which is fine for a development machine. However for deployment on a production environment you will probably create a new website and a new Application Pool for each ISAPI dll. This makes it easier to manage each ISAPI extension as well as more secure. Additionally you can shorten the URL to the `WebServiceISAPI.dll` with an URL Rewrite, which makes it both easier to access the web service as well as obscures the actual ISAPI dll filename. For production you will also need to harden your IIS installation, making it even more secure.

In our next article we'll be deploying our web service to Apache on Linux. Stay tuned!

Source code for this article can be downloaded here:
<https://www.blaisepascalmagazine.eu/your-downloads/>



kbmMW Professional and Enterprise Edition v. 5.16.00 released!

5.16.00 Jul 17 2021

**THIS IS A MAJOR NEW RELEASE WITH
THE INTRODUCTION OF OUR NEW
CONTEXT SENSITIVE I18N
INTERNATIONALIZATION FRAMEWORK.**

Important notes (changes that may break existing code)

- * Updated messaging subject header to v5 to support packet size. Notice newer servers will adjust to old clients, but newer clients will not be able to communicate with older servers, unless you specifically downgrade the header version (client side), or update the server. Downgrade the header version by setting the transport.Params property like this: `yourtransport.Params.Values[KBMMW_TRANSPORT_PARAM_SUBJECTHEADERVERSION]='4'`;
- * Updated TkbmMWORM's LINQ to support Delete and Update. May require minor changes to existing code to compile.

New stuff

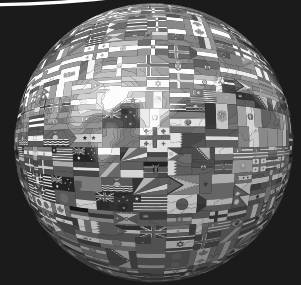
- Added kbmMWI18N.pas, kbmMWI18NVCL.pas and kbmMWI18NFMX.pas to Enterprise Edition. It is a complete context sensitive translation framework.
- Added kbmMWs4TLSTransportPlugin for StreamSec v4 TLS support.
- Updated TkbmMWORM's LINQ to support Delete and Update. May require minor changes to existing code to compile.
- Added Value2String class method to TkbmMWRTTI.
- Added VarRec2String class method to TkbmMWRTTI.
- Added VarRec2Variant class method to TkbmMWRTTI.
- Added ConvertToBoolean, ConvertToInt, ConvertToString, ConvertToFloat, ConvertToDateTime, Convert class functions to TkbmMWRTTI.
- Added AsString to TkbmMWNullable. Useful for debugging/logging purposes.
- Added AsString to TkbmMWDateTime. Useful for debugging/logging purposes.
- Added AsFieldValueArray to TkbmMWORMFieldList.
- Added two variations of DatasetFieldValueArray to DatasetFieldValueArray. Returns an array of field values.
- Added global kbmMWDebugEncoding:TEncoding which defines how to encode debug messages. Default encodes as ASCII.
- Added kbmMW SQL functions:
 - NumberToFixed(double)->string,
 - FixedToNumber(string)->double,
 - ToHex(int)->string or ToHex(digits,int)->string,
 - FromHex(string)->int
- Added IsLogLevel(const ALevel:TkbmMWLogLevel):boolean to TkbmMWLog.
- Added PrepareMessagePackStorage to TkbmMWConfiguration.

Changes/minor additions

- Cleaned up and improved data conversion in SmartBind.
- Improved SmartBind rebinding to a TDataSet/TDataSource.
- Improved shutdown detection in SmartBind.
- Improved shutdown detection in SmartEvent.
- Improved TkbmMWThreadObjectDictionary to optionally support an ordered list, which can be fetched using BeginReadOrdered and EndReadOrdered functions. Specify which to support ordering at construction time.
- Changed TkbmMWGenericMagneticStripeReaderHID.DecodeInputReport and TkbmMWMagTekMSRHID.DecodeInputReport to set status to an empty string if decoded ok.
- Improved TkbmMWYAMLStreamer to read and write comments. It will attempt to preserve comments when possible. Further it supports inserting comments before and/or after a node.
- Improved YAML character support during parsing.
- Added PreComment and PostComment properties to TkbmMWONCustomObject.
- Optimized all value setters of TkbmMWONCustomObject and TkbmMWONObject.
- Added general new MOBILE definition support for mobile non NEXTGEN.
- Improved TkbmMWLinQStage.AsON to accept optional extra parameter to return a plain object notation object (native or object) that is not embedded in an array (default).

Fixes

- Fixed kbmMWGetFileSize platform bug.
- Fixed TkbmMWTiming.GetTimeUS platform bug.
- Fixed MacOS compilation bugs.
- Fixed bug in TkbmMWHID, detecting HID candidates.
- Fixed bug in gzip detection in kbmMWCustomHTTPService.pas
- Changed misspelled 'tagString' to 'tagStrings' in SmartBind
- Fixed bug when no rows in TkbmMWBindingStringGridControlFMXProxy.FillGridColumn.





Ivar Leidus, CC BY-SA 4.0 <<https://creativecommons.org/licenses/by-sa/4.0/>>, via Wikimedia Commons

KBMMW PROFESSIONAL AND ENTERPRISE EDITION V. 5.16.00 RELEASED!

- **RAD Studio XE5 to 10.4.2 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**



THIS IS A MAJOR NEW RELEASE WITH THE INTRODUCTION OF OUR NEW CONTEXT SENSITIVE I18N INTERNATIONALIZATION FRAMEWORK.

- New I18N context sensitive internationalization framework to make your applications multilingual.
- New ORM LINQ support for Delete and Update.
- Comments support in YAML.
- New StreamSec TLS v4 support (by StreamSec)
- Many other feature 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.

