BLAISE PASCAL MAGAZINE 92

Blaise Pascal

ARTICLES

Google qubit sculpture
Google's Sycamore quantum computing chip has 54 qubits in a two-dimensional array. Running a program on the chip means changing the configuration of the qubits. This sculpture symbolizes the different states of the qubits with different layers as time passes.

ADVERTISERS

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).

Niklaus Wirth

Contributors

Stephen Ball http://delphiaball.co.uk @DelphiABall		**Peter Bijlsma -Editor** peter @ blaisepascal.eu
Dmitry Boyarintsev dmitry.living @ gmail.com	**Michaël Van Canneyt,** michael @ freepascal.org	**Marco Cantù** www.marcocantu.com marco.cantu @ gmail.com
David Dirkse www.davdata.nl E-mail: David @ davdata.nl	**Benno Evers** b.evers @ everscustomtechnology.nl	**Bruno Fierens** www.tmssoftware.com bruno.fierens @ tmssoftware.com
Holger Flick holger @ flixments.com		
Primož Gabrijelčič 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.blaisepascal.eu editor @ blaisepascal.eu	**Howard Page Clark** hdpc @ talktalk.net	**Heiko Rompel** info @ rompelsoft.de
Wim Van Ingen Schenau -Editor wisone @ xs4all.nl	**Peter van der Sman** sman @ prisman.nl	**Rik Smit** rik @ blaisepascal.eu
Bob Swart www.eBob42.com Bob @ eBob42.com	**B.J. Rao** contact @ intricad.com	**Daniele Teti** www.danieleteti.it d.teti @ bittime.it
Anton Vogelaar ajv @ vogelaar-electronics.com	**Danny Wind** dwind @ delphicompany.nl	**Siegfried Zuhr** siegfried @ zuhr.nl

Subscriptions (201**9** 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**	————	€ 240,00	€ 70,00

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

From your editor

Dear Reader,
here is our February issue of this year.
Especially in this issue is an article about the Easter holidays
and what influence they actually have on many holidays. Just for fun and out of curiosity.
I never understood all these changes each year again and therefore the funny behaviour of the planned dates…always changing. Next year even on my birthday…

On the cover you find a beautiful cube constructed by google for quantum purposes. It looks like
an artificial diamond of science. The quantum era has definitively made its entrance into our world.
I suppose the actual large-scale application of it will take a few more "days". The article I wrote about this is actually to make you aware of the becoming more and more reliable of that science.
But it is already in use by many experiments and BMW is making progress to solve the problem of buying parts the cheapest way, finding still the nearest supplier to Europe and best product.
Since they are using a huge number of parts this is monstrous a task that a normal computer can not solve in a short period of time.

I have a lot of great news to announce:
Lazarus has released its newest stable version: see page 70.
The final arrival of the TMS WebCore for free under the macOS Version Big Sure is described.
One of the members of the Lazarus team (Martin Friebe) has been able to create a special WebForm for Lazarus that will enable you to create your webforms and applications with "What you see is what you get: WYSIWIG".
We will go on developing that during the year and I will update you about this fantastic new feture constantly. But for now it is still under construction. There is more to come.

Delphi came wit and update: 10.4.2 Sydney. In the next issue I will dive into that version and write an article about updating without loosing all your installed components. They will soon come with a new version of the Community edition.

Components4Developers has announced its new version 5.14 and Kim Madsen wrote an article to secure your USB HID' s, which could become necessary, and of course there is a lot of other articles that might find your interest.

All new things happening and this beautiful weather makes me feel like spring is in the air. Refreshing, it already smells like the first flowers.

Lets enjoy spring!

Detlef

This article was taken out of the following address:
`https://www.howtogeek.com/711656/`
`what-is-end-to-end-encryption-`
`and-why-does-it-matter/`

INTRODUCTION

End-to-end encryption (E2EE) ensures that your data is encrypted (kept secret) until it reaches an intended recipient. Whether you're talking about end-to-end encrypted messaging, email, file storage, or anything else, this ensures that no one in the middle can see your private data.

In other words: If a chat app offers end-to-end encryption, for example, only you and the person you're chatting with will be able to read the contents of your messages. In this scenario, not even the company operating the chat app can see what you're saying.

ENCRYPTION BASICS

First, let's start with the basics of encryption. Encryption is a way of scrambling (encrypting) data so that it can't be read by everyone. Only the people who can unscramble (decrypt) the information can see its contents. If someone doesn't have the decryption key, they won't be able to unscramble the data and view the information.

(*This is how it's supposed to work, of course. Some encryption systems have security flaws and other weaknesses.*)

Your devices are using various forms of encryption all the time.

For example, when you access your online banking website—or any website using HTTPS, which is most websites these days—the communications between you and that website are encrypted so that your network operator, internet service provider, and anyone else snooping on your traffic can't see your banking password and financial details.

Wi-Fi uses encryption, too.
That's why your neighbors can't see everything you're doing on your Wi-Fi network—assuming that you use a modern Wi-Fi security standard that hasn't been cracked, anyway.
Encryption is also used to secure your data. Modern devices such as iPhones, Android phones, iPads, Macs, Chromebooks, and Linux systems (but not all Windows PCs) store their data on your local devices in Encryption "in Transit" and "at Rest": Who Holds the Keys?

So encryption is everywhere, and that's great. But when you're talking about communicating privately or storing data securely, the question is: Who holds the keys?
For example, let's think about your Google account. Is your Google data—your Gmail emails, Google Calendar events, Google Drive files, search history, and other data—secured with encryption?

Well, yes. In some ways.

Google uses encryption to secure data "in transit." When you access your Gmail account, for example, Google connects via secure HTTPS.
This ensures that no one else can snoop on the communication going on between your device and Google's servers.
Your internet service provider, network operator, people within range of your Wi-Fi network, and any other devices between you and Google's servers can't see the contents of your emails or intercept your Google account password.

Google also uses encryption to secure data "at rest." Before the data is saved to disk on Google's servers, it is encrypted.
Even if someone pulls off a heist, sneaking into Google's data center and stealing some hard drives, they wouldn't be able to read the data on those drives.

Both encryption in transit and at rest are important, of course. They're good for security and privacy.
It's much better than sending and storing the data unencrypted!
But here's the question:
Who holds the key that can decrypt this data?
The answer is Google.
Google holds the keys.

Since Google holds the keys, this means that Google is capable of seeing your data—emails, documents, files, calendar events, and everything else.

If a rogue Google employee wanted to snoop on your data—and yes, its happened—encryption wouldn't stop them.

If a hacker somehow compromised Google's systems and private keys (admittedly a tall order), they would be able to read everyone's data.

If Google was required to turn over data to a government, Google would be able to access your data and hand it over.

Other systems may protect your data, of course. Google says that it has implemented better protections against rogue engineers accessing data.
Google is clearly very serious about keeping its systems secure from hackers.
Google has even been pushing back on data requests in Hong Kong, for example.

So yes, those systems may protect your data. **But that's not encryption protecting your data from Google.**
It's just Google's policies protecting your data.
Don't get the impression that this is all about Google.
It's not—not at all.
Even Apple, so beloved for its privacy stances, does not end-to-end encrypt iCloud backups.
In other words:
Apple keeps keys that it can use to decrypt everything you upload in an iCloud backup.

HOW END-TO-END ENCRYPTION WORKS
Now, let's talk chat apps.
For example: Facebook Messenger. When you contact someone on Facebook Messenger, the messages are encrypted in transit between you and Facebook, and between Facebook and the other person.
The stored message log is encrypted at rest by Facebook before it's stored on Facebook's servers.
But Facebook has a key.
Facebook itself can see the contents of your messages.

THE SOLUTION IS END-TO-END ENCRYPTION.
With end-to-end encryption, the provider in the middle—whoever you replace Google or Facebook with, in these examples—will not be able to see the contents of your messages.

They do not hold a key that unlocks your private data. Only you and the person you're communicating with hold the key to access that data.
Your messages are truly private, and only you and the people you're talking to can see them—not the company in the middle.

WHY IT MATTERS

End-to-end encryption offers much more privacy. For example, when you have a conversation over an end-to-end encrypted chat service like Signal, you know that only you and the person you're talking to can view the contents of your communications.

However, when you have a conversation over a messaging app that isn't end-to-end encrypted—like Facebook Messenger—you know that the company sitting in the middle of the conversation can see the contents of your communications.

IT'S NOT JUST ABOUT CHAT APPS.

For example, email can be end-to-end encrypted, but it requires configuring PGP encryption or using a service with that built in, like ProtonMail.
Very few people use end-to-end encrypted email.
End-to-end encryption gives you confidence when communicating about and storing sensitive information, whether it's financial details, medical conditions, business documents, legal proceedings, or just intimate personal conversations you don't want anyone else having access to.

END-TO-END ENCRYPTION ISN'T JUST ABOUT COMMUNICATIONS

End-to-end encryption was traditionally a term used to describe secure communications between different people.
However, the term is also commonly applied to other services where only you hold the key that can decrypt your data.with a secret only you know.

**FOR EXAMPLE, PASSWORD MANAGERS such as 1PASSWORD, BITWARDEN, LASTPASS, AND DASHLANE ARE END-TO-END ENCRYPTED.
THE COMPANY CAN'T RUMMAGE THROUGH YOUR PASSWORD**

In a sense, this is arguably "end-to-end" encryption—except that you're on both ends.

No one else—not even the company that makes the password manager—holds a key that lets them decrypt your private data.

You can use the password manager without giving the password manager company's employees access to all your online banking passwords.

ANOTHER GOOD EXAMPLE:

If a file storage service is end-to-end encrypted, that means that the file storage provider can't see the contents of your files.

If you want to store or sync sensitive files with a cloud service—for example, tax returns that have your social security number and other sensitive details—encrypted file storage services are a more secure way to do that than just dumping them in a traditional cloud storage service such as **Dropbox, Google Drive, or Microsoft OneDrive.**

ONE DOWNSIDE:

Don't Forget Your Password!
There's one big downside with end-to-end encryption for the average person:
If you lose your decryption key, you lose access to your data.
Some services may offer recovery keys that you can store, but if you forget your password and lose those recovery keys, you can no longer decrypt your data.

That's one big reason that companies like Apple, for example, might not want to end-to-end encrypt iCloud backups.

Since Apple holds the encryption key, it can let you reset your password and give you access to your data again.

This is a consequence of the fact that Apple holds the encryption key and can, from a technical perspective, do whatever it likes with your data.
If Apple didn't hold the encryption key for you, you wouldn't be able to recover your data.
Imagine if, every time someone forgets a password to one of their accounts, their data in that account would be wiped out and become inaccessible.

Forget your Gmail password? Google would have to erase all your Gmails to give you your account back. That's what would happen if end-to-end encryption was used everywhere.

Examples of Services That Are End-to-End Encrypted

Here are some basic communication services that offer end-to-end encryption.
This isn't an exhaustive list—it's just a short introduction.

For chat apps, **Signal** offers end-to-end encryption for everyone by default.
Apple iMessage offers end-to-end encryption, but Apple gets a copy of your messages with the default iCloud backup settings.

WhatsApp says that every conversation is end-to-end encrypted,

Some other apps offer end-to-end encryption as an optional feature that you have to enable manually, including **Telegram** and Facebook Messenger.
For end-to-end encrypted email, you can use PGP—however, it's complicated to set up.

Thunderbird now has integrated **PGP** support.

There are encrypted email services like ProtonMail and Tutanota that store your emails on their servers with encryption and make it possible to more easily send encrypted emails.
For example, if one ProtonMail user emails another ProtonMail user, the message is automatically sent encrypted so that no one else can see its contents.

However, if a ProtonMail user emails someone using a different service, they'll need to set up PGP to use encryption. (*Note that encrypted email doesn't encrypt everything: While the message body is encrypted, for example, subject lines aren't.*)

RELATED: What Is Signal, and Why Is Everyone Using It?

End-to-end encryption is important.

IF YOU'RE GOING TO HAVE A PRIVATE CONVERSATION OR SEND SENSITIVE INFORMATION, DON'T YOU WANT TO MAKE SURE THAT ONLY YOU AND THE PERSON YOU'RE TALKING TO CAN SEE YOUR MESSAGES?

VIRTUAL BOX PROBLEM

Since I build a new computer having a AMD Ryzen CPU and installed Windows on it I found it very disappointing my VirtualBox doesn't work on that any more.
I found out it actually it's very simple to be solved:
in the settings of the BIOS go to Advanced CPU Settings and than it will come up with the SVM Mode: this is set standard to disable, but it needs to be enabled. Problem gone.

"Apparently, he's having a hard time understanding quantum computing. His brain locked up when I tried explaining it to him."

maXbox

Author: Max Kleiner

"Time goes, you say?
Ah, no! alas, time stays, we go."
 - *Henry Austin Dobson*

At its core, RSS refers to simple text files (XML/RDF) with more or less important, updated information - news pieces, articles, weather info, opinion mining that sort of thing.
In the following I want to show this topic thing with the BBC-News feeder. News feeds allow you to see when websites have added new content. You can get the latest headlines and video in one place, as soon as it's published, without having to visit the websites you have taken the feed from.

BBC News as our example provides feeds for both the desktop website as well as for our mobile site and the most popular feeds are listed here:
https://www.bbc.co.uk/news/10628494

First we define the URL to get the content from:

```
Const
  RSS_NewsFeed = 'http://feeds.bbci.co.uk/news/world/rss.xml';
```

RSS is an **XML** based document format for syndicating news and other timely news-like information.
It provides headlines, URLs to the source document and brief description information in an easy to understand and use format.
RSS based "News Readers" and **"News Aggregators"** allow the display of **RSS** headlines on workstation desktops.
Users of RSS content use apps called feed 'readers' or 'aggregators' (*newer versions of Web browsers offer built in support for RSS feeds*): a user subscribes to a feed by entering the link of the RSS feed into their RSS feed reader;

of course software libraries exist to read the RSS format and present RSS headlines on webpages and other online applications like in our script example with SimpleRSS. So you can feed a memo or text component in your form.

In the XML baseline you see this:

```
<?xml version="1.0"?>
- <rdf:RDF xmlns:rdf=
    "http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns="http://purl.org/rss/1.0/">
- <!-- XML Generated by SimpleRSS
  http://simplerss.sourceforge.net at Sat, 15 Jan 2021 11:43:18 -->
- <channel xmlns="" rdf:about="">
  <title>Title Required</title>
  <link>Link Required</link>
  <description>Description Required</description>
- <items>
  <rdf:Seq />
  </items>
  </channel>
  </rdf:RDF>
```

This structure is a convention and suitable for a record:

```
type
 TRSSItem2 = record
  FPubDate: TDate;
  FLink: string;
  FTitle: string;
  FDescription: string;
 end;
```

The **FPubDate** made me a bit upset because I had to convert it in **TRFC822DateTime Format**. This specified date-time value, while technically valid, is likely to cause interoperability issues.

The value specified must meet the Date and Time specifications as defined by **RFC822,** with the exception that the year should be expressed as four digits.
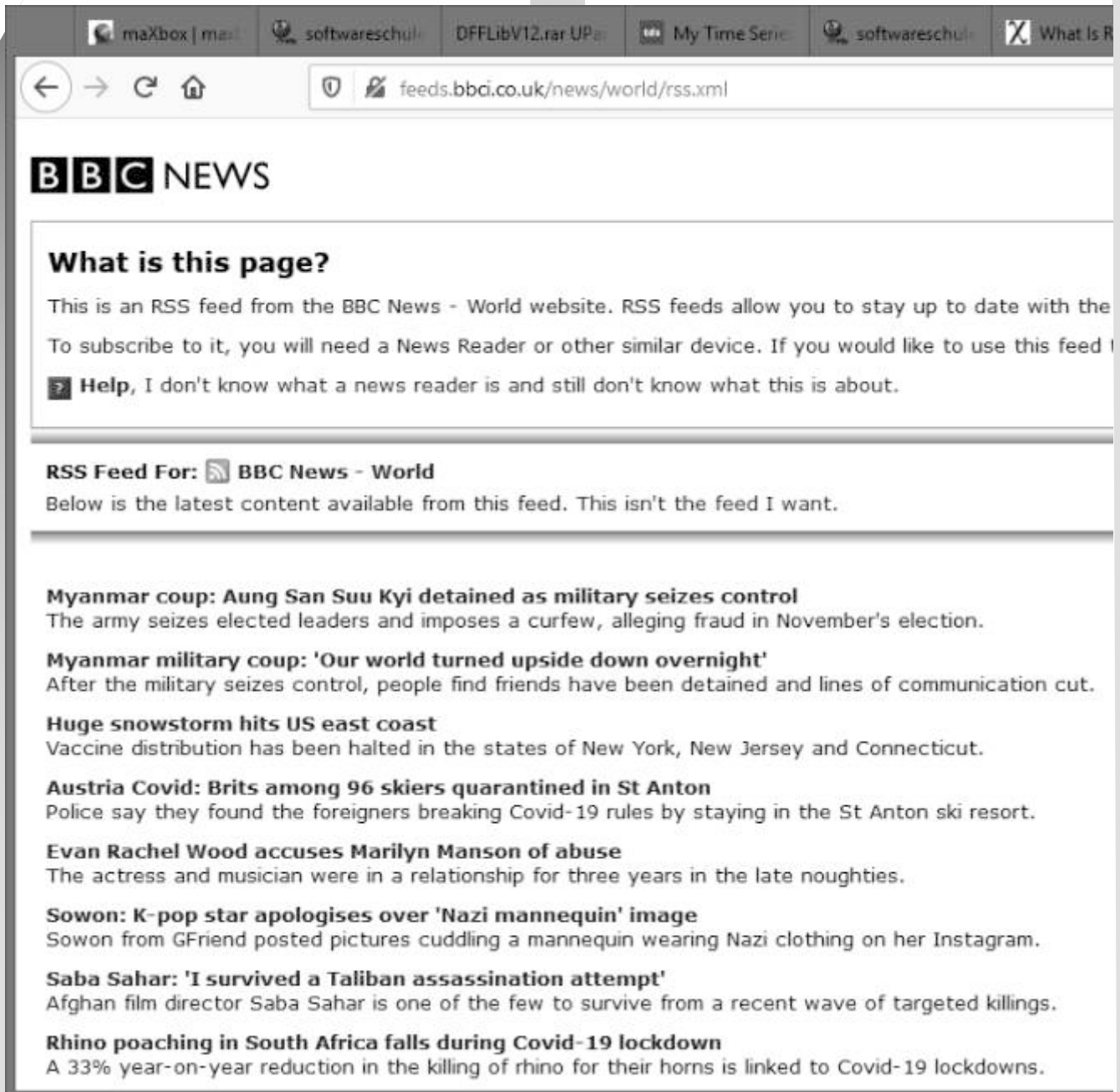The syntax is like (*all single spaced and no comments*):

```
<pubDate>Wed, 02 Oct 2020 08:00:00 EST</pubDate>
<pubDate>Wed, 02 Oct 2020 13:00:00 GMT</pubDate>
```

There, of course, is knowledge on the web, but in my case I use a component with specific objects to adapt. But the good news is the core code is straight and simple:

```
//RSS Script Feed Snippet:
with TSimpleRSS.create(self) do begin
 XMLType:= xtRDFrss;
 IndyHTTP:= TIdHTTP.create(self);
 LoadFromHTTP(RSS_NewsFeed);
 //LoadFromHTTP(Climatefeed);
 writeln('RSSVersion: '+Version)
 writeln('SimpleRSSVersion: '+SimpleRSSVersion)
 for it:= 0 to items.count-1 do
  writeln(itoa(it)+': '+Items[it].title+':
'+items[it].pubdate.getdatetime);
 end;
```

maXbox | ma⋯   softwareschul⋯   DFFLibV12.rar UP⋯   My Time Seri⋯   softwareschul⋯   What Is R

← → C ⌂   🔓 🔏 feeds.bbci.co.uk/news/world/rss.xml

B B C NEWS

What is this page?

This is an RSS feed from the BBC News - World website. RSS feeds allow you to stay up to date with the

To subscribe to it, you will need a News Reader or other similar device. If you would like to use this feed

❔ **Help**, I don't know what a news reader is and still don't know what this is about.

RSS Feed For: 📶 BBC News - World

Below is the latest content available from this feed. This isn't the feed I want.

Myanmar coup: Aung San Suu Kyi detained as military seizes control
The army seizes elected leaders and imposes a curfew, alleging fraud in November's election.

Myanmar military coup: 'Our world turned upside down overnight'
After the military seizes control, people find friends have been detained and lines of communication cut.

Huge snowstorm hits US east coast
Vaccine distribution has been halted in the states of New York, New Jersey and Connecticut.

Austria Covid: Brits among 96 skiers quarantined in St Anton
Police say they found the foreigners breaking Covid-19 rules by staying in the St Anton ski resort.

Evan Rachel Wood accuses Marilyn Manson of abuse
The actress and musician were in a relationship for three years in the late noughties.

Sowon: K-pop star apologises over 'Nazi mannequin' image
Sowon from GFriend posted pictures cuddling a mannequin wearing Nazi clothing on her Instagram.

Saba Sahar: 'I survived a Taliban assassination attempt'
Afghan film director Saba Sahar is one of the few to survive from a recent wave of targeted killings.
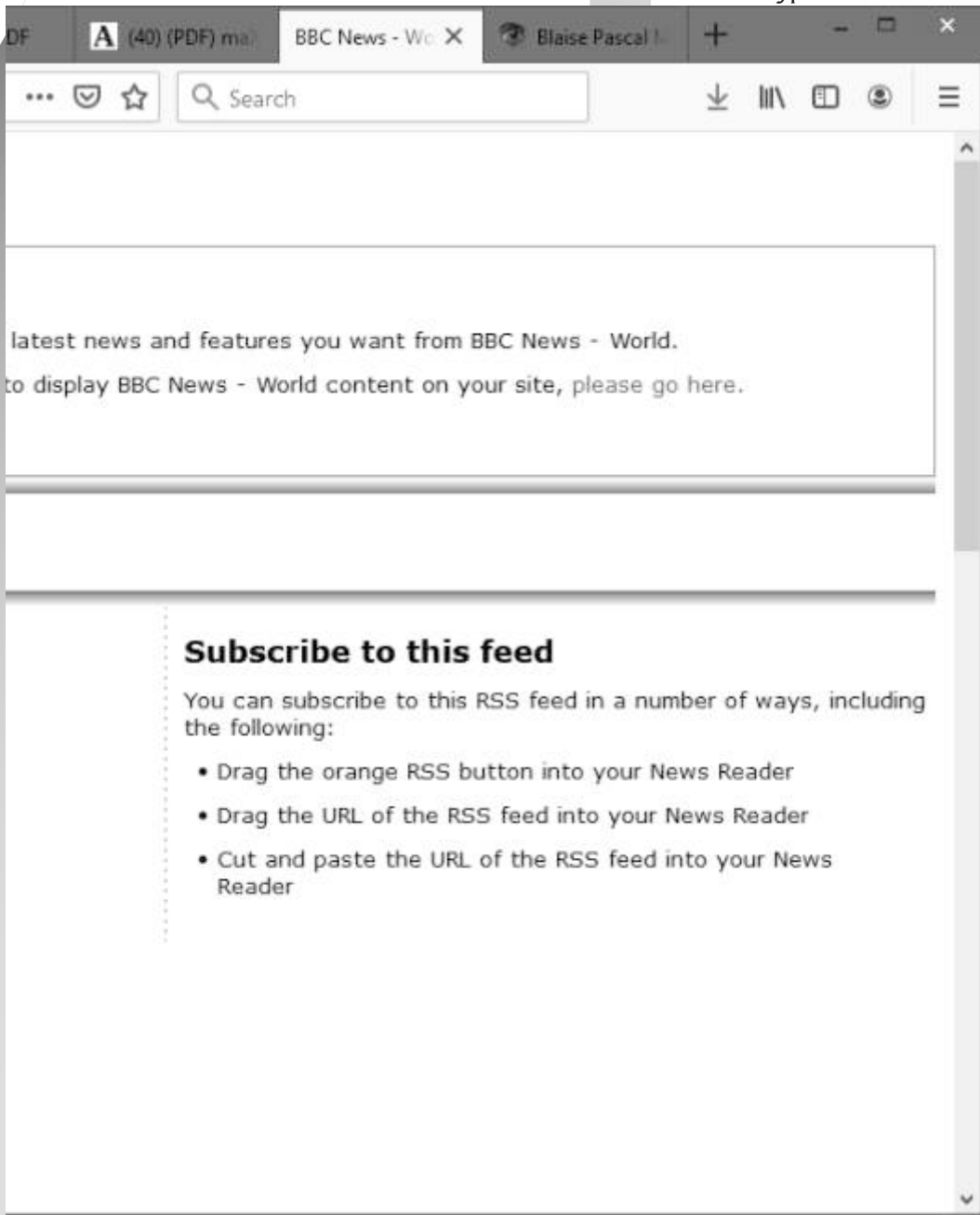
Rhino poaching in South Africa falls during Covid-19 lockdown
A 33% year-on-year reduction in the killing of rhino for their horns is linked to Covid-19 lockdowns.

Items are sticked together in the class TRSSItems and inherits from TOwnedCollection.
These CollectionItem objects in turn contain their own published child property which descends from TCollection and contain their own TCollectionItem descendant - so a nested TCollection/TCollectionItem scenario.

The http provider is by default Indy and we can load the feeds as a stream with the LoadFromHTTP() Method. The RSS Version is based with 2. These modern supplied RSS documents use the RSS 2.0 format. Each RSS item links to the html/web documents are described. Additional technical information is available from the following non-US Government website like BBC News. The XMLType is based on the RDF proposal.

latest news and features you want from BBC News - World.

to display BBC News - World content on your site, please go here.

Subscribe to this feed

You can subscribe to this RSS feed in a number of ways, including the following:

- Drag the orange RSS button into your News Reader
- Drag the URL of the RSS feed into your News Reader
- Cut and paste the URL of the RSS feed into your News Reader

RDF is a kind of sematic web. The Semantic Web enables devices to seek out knowledge distributed throughout the Web, mesh or mix it, and then take action based on it. Simply said:

The **Resource Description Framework (RDF)** is the **W3C** standard for encoding knowledge. After running the script you get this similar output:

```
29: Life in a Day: Kevin Macdonald says
film 'reinforces everyones similarities':
Mon, 01 Feb 2021 00:00:57 GMT
```

link:
```
https://www.bbc.co.uk/news/
entertainment-arts-55861945
```

descript:
```
Kevin Macdonalds documentary features
personal videos from across the world
- all shot on the same day.
```

rssitem.title
```
BBC News - World
```

rssitem.link
```
https://www.bbc.co.uk/news/
```

To build your own provider for example with TLS1.3 or to link with your own **XML-Parser/DOM** vendor you can use the `LoadFromStream()` method.
Now the code block for a weather service with **HTTPS** and `LoadFromStream()`:

```
Const Weatherfeed5Bern=
   'https://weather-broker-cdn.api.bbci.co.uk/en/forecast/rss/3day/2661552';
```

```
function GetBlogStream8(const S_API,pData: string;
                 astrm: TStringStream): TStringStream;
begin
 HttpGET(S_API, astrm) //maps HTTPS from WinInet_HttpGet
 result:= astrm;
end;

strm:= TStringStream.create('');
strm:= GetBlogStream8(WeatherFeed5Bern,'', strm);

 with TSimpleRSS.create(self) do begin
  XMLType:= xtRDFrss;   // bbcnews: xtRDFrss;
  //( xtRDFrss, xtRSSrss, xtAtomrss, xtiTunesrss )');
  //GenerateXML;
  LoadFromStream((strm));
  SaveToFile('C:\maXbox\Lazarus\rssbbctest.xml');
  writeln('RSSFeedVersion: '+Version)
  writeln('SimpleRSSVersion: '+SimpleRSSVersion)
  for it:= 0 to items.count-1 do
   writeln(itoa(it)+': '+Items[it].title+':
     '+items[it].pubdate.getdatetime);
  strm.Free;
 end;
```

And the output as iterated items from **RSS-Reader** will be (3-day forecast):

```
RSSFeedVersion: 2.0
SimpleRSSVersion: ver 0.4 (BlueHippo) Release 1
   0:  Today: Light Snow, Min Temperature:-5°C (23°F)
       Max Temperature: 0°C (32°F): Sat, 15 Jan 2021 10:37:30 Z
   1:  Saturday:Light Cloud,Min Temperature:-3°C (27°F)
       Max Temperature:-1°C (30°F):Sat,15 Jan 2021 10:37:30 Z
   2:  Sunday:Sleet Showers,Min Temperature:-1°C (31°F)
       Max Temperature:3°C (38°F):Sat,15 Jan 2021 10:37:30 Z
```

By the way, weather data have their own format, which is called **NWS,** and is not to be confused with **RSS** and cannot be read by **RSS** readers and aggregators. These files present more detailed information than the **RSS** feeds in strings friendly for parsing. Both the **RSS** and **XML** feeds offer URLs to icon images.

Conclusion:
Really Simple Syndication (RSS) is a family of web formats used to publish frequently updated digital content. Most commonly used to update news articles, weather reports or traffic services and other content that changes quickly, RSS feeds may also include audio files (PodCasts) or even video files (VodCasts). SimpleRSS components provides methods for accessing, importing, exporting and working with RSS, RDF, Atom & iTunes Feeds. This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License.

Next time I show an an online translation service with sentiment analysis:
Trump's false election fraud claims face a dead end
Trumps falsche Wahlbetrugsansprüche stehen vor einer Sackgasse
Trump's valse verkiezingsfraudeclaims lopen dood
Les fausses allégations de fraude électorale de Trump font face à une impasse

Ref Script & Component:
```
http://www.softwareschule.ch/
   examples/bbcnews.txt
http://simplerss.sourceforge.net
script: 1017_XmlDocRssParser.pas
```
Doc:
```
https://maxbox4.wordpress.com
http://feedvalidator.org/docs/rss2.html
http://web.resource.org/rss/1.0/modules/
   content/
```

In this article we summon the latest developments of Quantum computing. After having written several articles (Issue 62 – Quantum computing, Issue 63 - Part2 Building a quantum computer, 66 Majorana, the new solution for QantumBits, Issue 73/74 Quantum Internet) I write about a new development: larger numbers of QBits and increased certainty for computations.

QUANTUM COMPUTING
Quantum computing is the use of quantum phenomena such as superposition and entanglement to perform computation. Computers that perform quantum computations are known as quantum computers.

Quantum computers are believed to be able to solve certain computational problems, such as integer factorization (*which underlies RSA encryption*), substantially faster than classical computers.
The study of quantum computing is a subfield of quantum information science.

Quantum computing began in the early 1980s, when physicist Paul Benioff proposed a quantum mechanical model of the Turing machine. Richard Feynman and Yuri Manin later suggested that a quantum computer had the potential to simulate things that a classical computer could not.

In 1994, Peter Shor developed a quantum algorithm for factoring integers that had the potential to decrypt RSA-encrypted communications.
Despite ongoing experimental progress since the late 1990s, most researchers believe that "fault-tolerant quantum computing is still a rather distant dream. In recent years, investment into quantum computing research has increased in both the public and private sector.

The article was gathered from various articles about Quantum Computing, News Items, WikiPedia explanations a full list of used information is shown at the end of the article.

PART1: QUANTUM VOLUME:
OVERVIEW OF QUANTIFYING
Of course all these giants will try to be the first or the greatest and time has shown that all of them win - now and then.
Up to this moment there are the well-known names that work on **Quantum Computers:**

❶ **IBM**
is developing a Software language tool for development: **Qiskit**
`https://qiskit.org/`

❷ **Google**
An open source framework for programming quantum computers: **Cirq**
Cirq is a Python software library for writing, manipulating, and optimizing quantum circuits, and then running them on quantum computers and quantum simulators.
`https://quantumai.google/cirq`

❸ **Microsoft**
The **Quantum Development Kit for Q#** and **Azure Quantum.**
The Quantum Development Kit is the development kit for Q#, quantum - focused programming language and Azure Quantum, quantum cloud platform. Build and run Q# programs on quantum hardware or formulate solutions that execute optimization.

❹ **Honeywell**
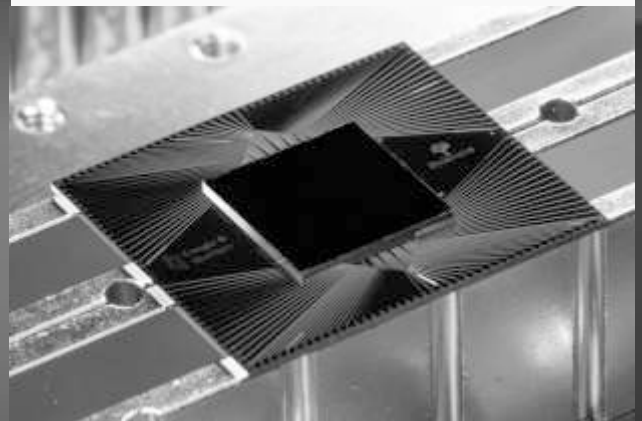uses Microsoft's **Quantum Development Kit for Q#** and **Azure Quantum.**

Figure 1: The Sycamore chip is composed of 54 qubits, each made of superconducting loops. Credit: Erik Lucero.

QBit

*In quantum computing, a **qubit** or quantum bit (sometimes **qbit** is the basic unit of quantum information - the quantum version of the classical 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 which is fundamental to quantum mechanics and quantum computing.*

BINARY DIGIT

A binary digit, characterized as 0 or 1, is used to represent information in classical computers.
*When averaged over both of its states (0,1), a binary digit can represent up to one bit of **Shannon information,** where a bit is the basic unit of information.*

*However, in this article, **the word bit is synonymous with a binary digit.***

In classical computer technologies, a processed bit is implemented by one of two levels of low DC voltage, and whilst switching from one of these two levels to the other, a so-called forbidden zone must be passed as fast as possible, as electrical voltage cannot change from one level to another instantaneously.

There are two possible outcomes for the measurement of a qubit - usually taken to have the value "0" and "1", like a bit or binary digit.

However, whereas the state of a bit can only be either 0 or 1, the general state of a qubit according to quantum mechanics can be a coherent superposition of both.

*Moreover, whereas a measurement of a classical bit would not disturb its state, **a measurement of a qubit would destroy its coherence and irrevocably disturb the superposition state.** It is possible to fully encode one bit in one qubit.*
However, a qubit can hold more information, e.g. up to two bits using superdense coding.

*In quantum information theory, **superdense coding (or dense coding)** is a quantum communication protocol to transmit two classical bits of information (i.e., either 00, 01, 10 or 11) from a sender (often called Alice) to a receiver (often called Bob), by sending only one qubit from Alice to Bob, under the assumption of Alice and Bob pre-sharing an entangled state.*

By performing one of four quantum gate operations on the (entangled) qubit she possesses, Alice can pre-arrange the measurement Bob makes.

After receiving Alice's qubit, operating on the pair and measuring both, Bob has two classical bits of information.

*If Alice and Bob do not already share entanglement before the protocol begins, then it is impossible to send two classical bits using 1 qubit, as this would violate **Holevo's** theorem it is an important limitative theorem in quantum computing, an interdisciplinary field of physics and computer science.*
It is sometimes called Holevo's bound, since it establishes an upper bound to the amount of information that can be known about a quantum state (accessible information).

Superdense coding is the underlying principle of secure quantum secret coding.
The necessity of having both qubits to decode the information being sent eliminates the risk of eavesdroppers intercepting messages.

It can be thought of as the opposite of quantum teleportation, in which one transfers one qubit from Alice to Bob by communicating two classical bits, as long as Alice and Bob have a pre-shared Bell pair.

For a system of n components, a complete description of its state in classical physics requires only n bits, whereas in quantum physics it requires 2n complex numbers.

TRANSMONS, ION TRAPS AND TOPOLOGICAL QUANTUM COMPUTERS.
Progress towards building a physical quantum computer focuses on technologies such as transmons, ion traps and topological quantum computers, which aim to create high-quality qubits.

These qubits may be designed differently, depending on the full quantum computer's computing model, whether quantum logic gates, quantum annealing, or adiabatic quantum computation.

There are currently a number of significant obstacles in the way of constructing useful quantum computers.
In particular, it is difficult to maintain the quantum states of qubits as they suffer from quantum decoherence and state fidelity.
Quantum computers therefore require error correction.

Any computational problem that can be solved by a classical computer can also be solved by a quantum computer.
Conversely, any problem that can be solved by a quantum computer can also be solved by a classical computer, at least in principle given enough time. In other words, quantum computers obey the Church–Turing thesis.

While this means that quantum computers provide no additional advantages over classical computers in terms of computability, quantum algorithms for certain problems have significantly lower time complexities than corresponding known classical algorithms.

Notably, quantum computers are believed to be able to quickly solve certain problems that no classical computer could solve in any feasible amount of time - a feat known as QUANTUM SUPREMACY.
The study of the computational complexity of problems with respect to quantum computers is known as quantum complexity theory.

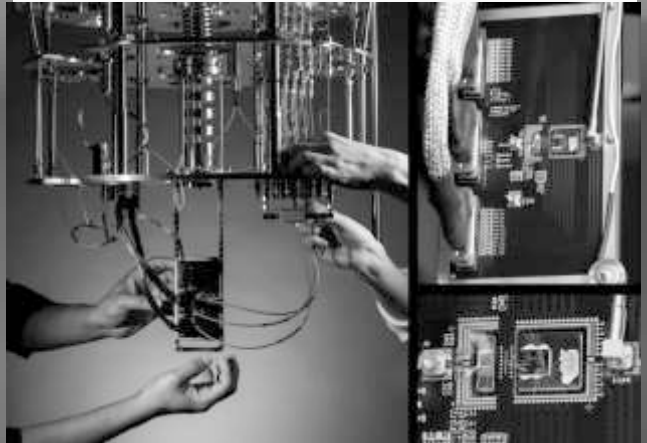*Figure 2:*

Figure 3: An IBM quantum computer where you can see the processor placed in the container at the bottom.
Stephen Shankland/CNET

IBM

says that software development is best done collaboratively, since open-source approaches are based around the understanding that an ecosystem of different human needs drives the best outcomes - and quantum computing is no different.
They expect developers to work in each of three key segments laying the groundwork for those working higher up in the stack.

- At the lowest level, quantum kernel developers are creating high-performance quantum circuits with timing and pulse-level controls.
- Quantum algorithm developers rely on these circuits to develop groundbreaking quantum algorithms that might provide an advantage over present-day classical computing solutions.
- Finally, quantum model developers apply these algorithms to real-world use cases in order to develop quantum models for chemistry, physics, biology, machine learning, optimization, or even finance.

Workloads with both quantum and classical components is not constrained by origin or the nature of integration, and a hybrid cloud will allow these workloads to run everywhere that our cloud native systems run today and in the future.IBM is developing a Software language tool for development: **Qiskit**

IBM promises 100x faster quantum computers through new software foundations.
Big Blue's open-source software efforts span the basics of quantum computing to higher-level jobs like AI and molecular simulations.

IBM expects it will increase performance of its complex machines by a factor of 100, a development that builds on Big Blue's progress in making the advanced computing hardware.

Much of the software will be written using open-source technology that outsiders can contribute to and benefit from, IBM mentions that adding the improvements will lead to a 100x speedup.

IBM's planned quantum computing software releases are part of an effort to hide away as much of the quantum computing complexity as possible for ordinary folks.
Getting a quantum computer to do useful work today is intricate, even with specialists like **Zapata Computing** and **Cambridge Quantum Computing**.

The software speedups could mean jobs that took months on a classical computer can be finished in a few hours. That would bring quantum computers closer to fulfilling their potential of solving problems out of reach of classical machines.
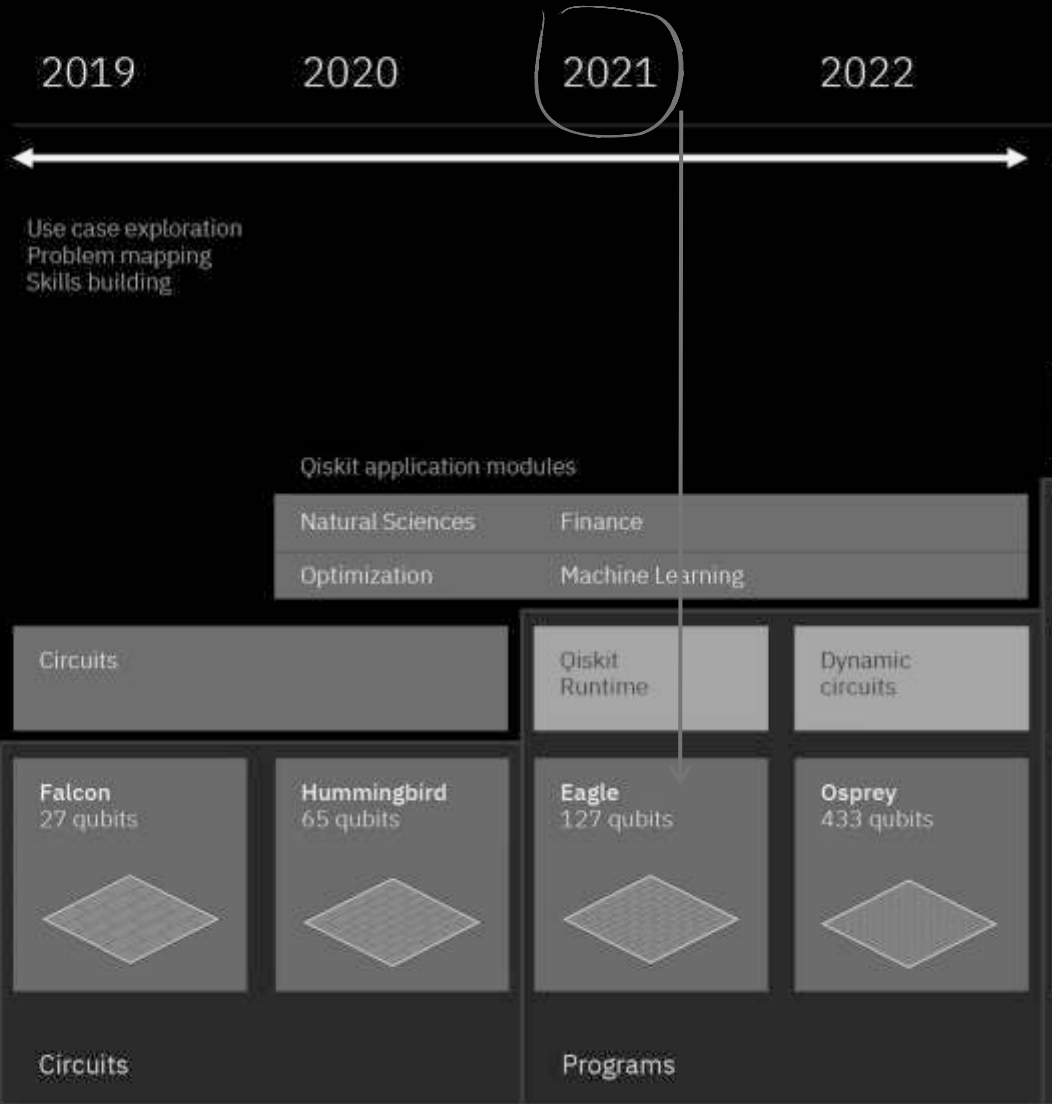
QUANTUM CONNECTION
IBM is among competitors like Google, Intel, Microsoft, IonQ, Rigetti Computing and Honeywell racing for leadership in quantum computing.

Where classical computing technology is relatively settled, quantum computers employ a wide variety of approaches. It's not yet clear which among them will prevail as the technology fledges from research labs into real-world use.

At the heart of quantum computers are qubits*, data storage and processing elements that can store a combination of one and zero. ➔

Development Roadmap

	2019	2020	2021	2022
Enterprise Clients	Use case exploration Problem mapping Skills building			
Model developers				
Algorithm developers		Qiskit application modules: Natural Sciences / Finance / Optimization / Machine Learning		
Kernel developers	Circuits		Qiskit Runtime	Dynamic circuits
Quantum systems	Falcon 27 qubits	Hummingbird 65 qubits	Eagle 127 qubits	Osprey 433 qubits
IBM Cloud	Circuits		Programs	

Quantum computers connect qubits through a quantum physics phenomenon called entanglement that lets a machine encompass an enormous number of possible solutions to a problem.

Operating a quantum computer involves applying a series of manipulations called gates to the qubits. With a specific sequence of gates, called a circuit, quantum computers can perform computations for particular tasks like simulating molecules or optimizing parts purchases, which BMW is trying on its complex supply chain.
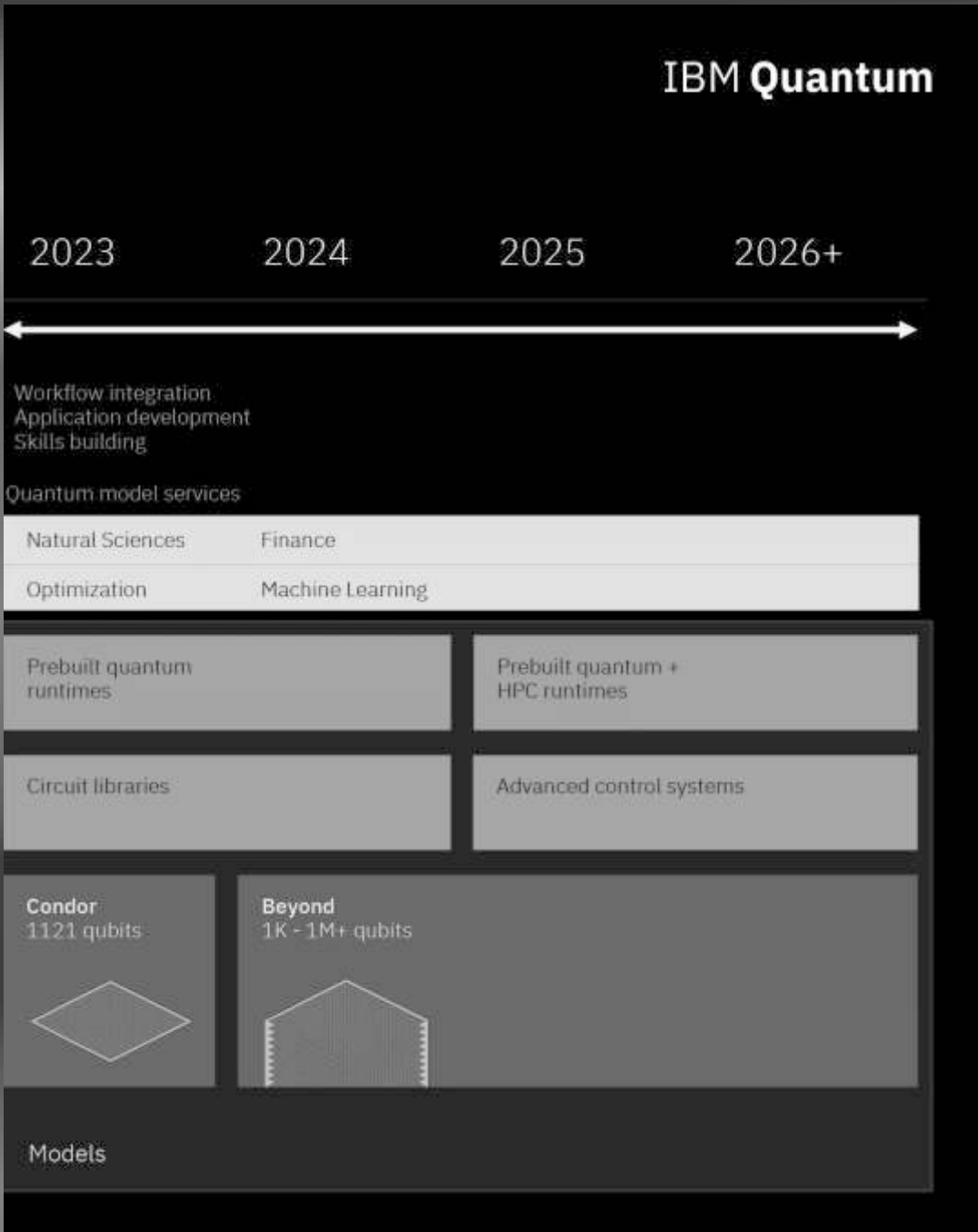
Figure 4: The IBM roadmap

On 23 October 2019, **Google AI**, in partnership with the **U.S. National Aeronautics and Space Administration** (**NASA**), claimed to have performed a quantum computation that is infeasible on any classical computer.

There are several models of quantum computers (or rather, quantum computing systems), including the quantum circuit model, quantum Turing machine, adiabatic* quantum computer, *(*relating to or denoting a process or condition in which heat does not enter or leave the system concerned*)
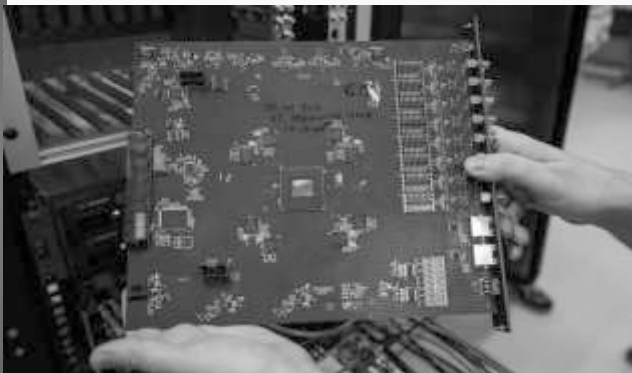
Figure 5:

GOOGLE
after some troubles a new era of computing seems to be here.
Researchers at Google claim their quantum computer has solved a problem that would take even the very best conventional machine thousands of years to crack.

The milestone, known as QUANTUM SUPREMACY, represents a long-sought stride towards realising the immense promise of quantum computers, devices that exploit the strange properties of quantum physics to speed up certain calculations.
It shows that quantum computing is really hard but not impossible.

The paper demonstrates that a quantum processor consisting of 54 superconducting quantum bits, or qubits, was able to perform a random sampling calculation – essentially verifying that a set of numbers is randomly distributed – exponentially faster than any standard computer.

Google's SYCAMORE device did it in just 3 minutes and 20 seconds, although one of the qubits had to be turned off as it wasn't working properly. It stands by the claims that the calculation would have taken IBM's Summit,
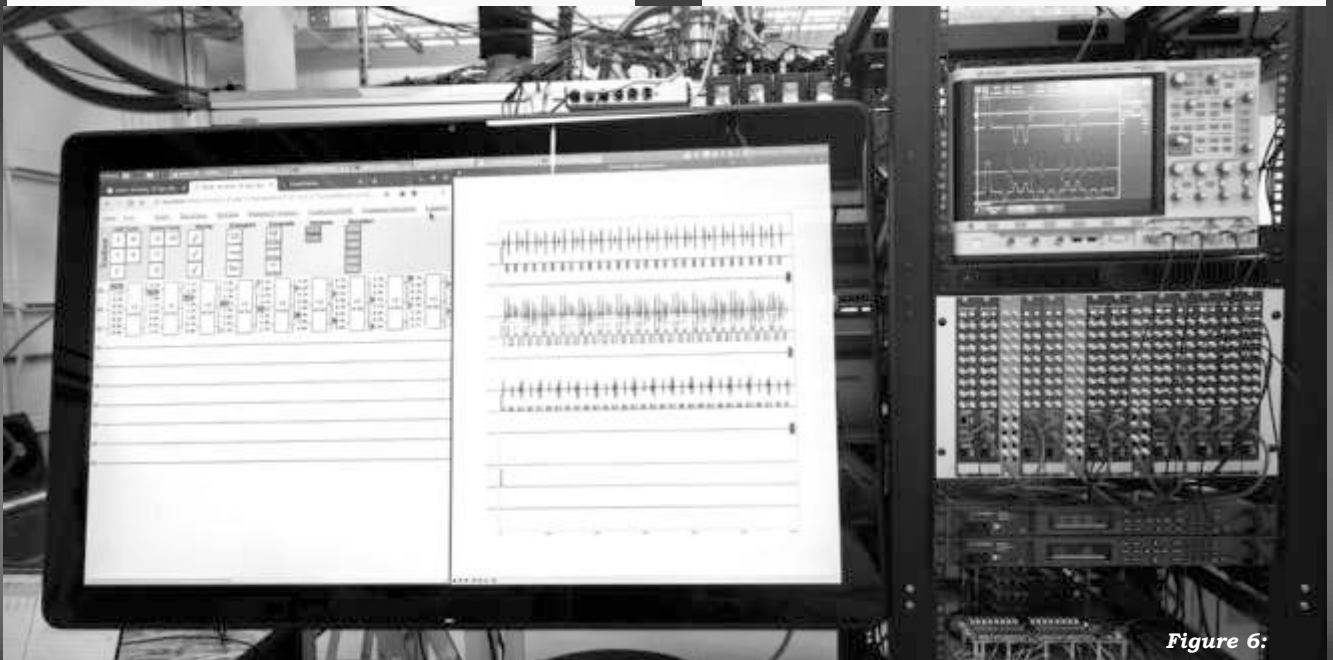

Figure 6:

Figure 6:

IBM has already pushed back on that claim, insisting that with some clever classical programming, its machine can solve the problem in 2.5 days.
Indeed, IBM, which has its own 53-qubit quantum computer, prefers a higher threshold for quantum supremacy, which explains its argument that Google has not yet reached the milestone.

But such caveats should not detract too much from Google's achievement.
There wil always be clever ways to tweak classical algorithms, but until Google has had a chance to digest the methodology IBM are proposing, it is hard to judge.

We will see more of this sort of back and forth when it comes to claims of quantum supremacy.

Even if you accept IBM's claims at face value, Google's quantum computer is still a big step forward, says Ciarán Gilligan-Lee at University College London.

"IBM is claiming that, even when running the world's largest computer for two and half days, and running petabytes of memory, they can simulate what the quantum chip does in 200 seconds. When you put it into context, it is still a pretty impressive achievement."

It doesn't mean quantum computers are ready to tackle real-world problems though – that remains decades away. Instead, it is a proof of concept.

We are looking forward to the next milestone: proof that we have sufficient control over the qubits that we can overcome the small errors they accumulate during calculations.

We are now in a phase we call **N**oisy **I**ntermediate - **S**cale **Q**uantum computing, or **NISQ**.
TO GET BEYOND THAT, WE NEED TO START DOING ERROR CORRECTION.
The architecture of the Google chip is already optimised for that.

Figure 7:
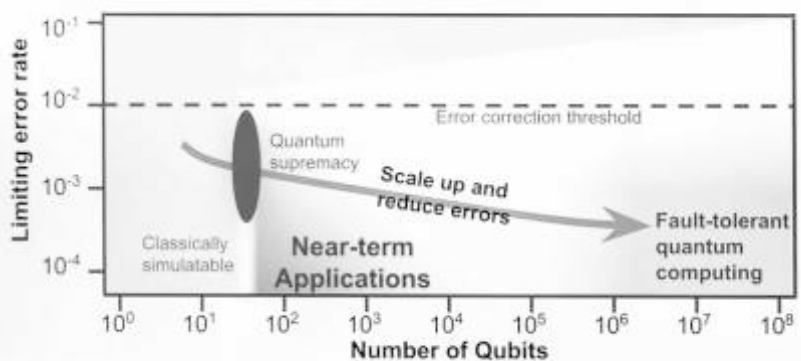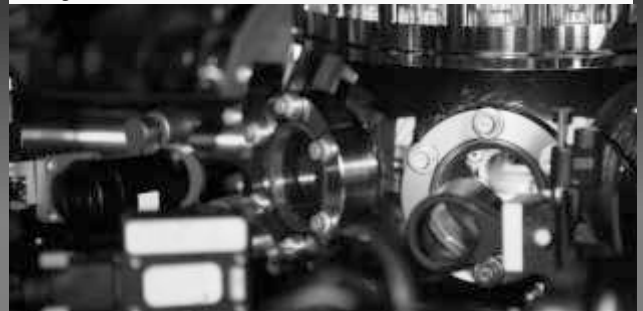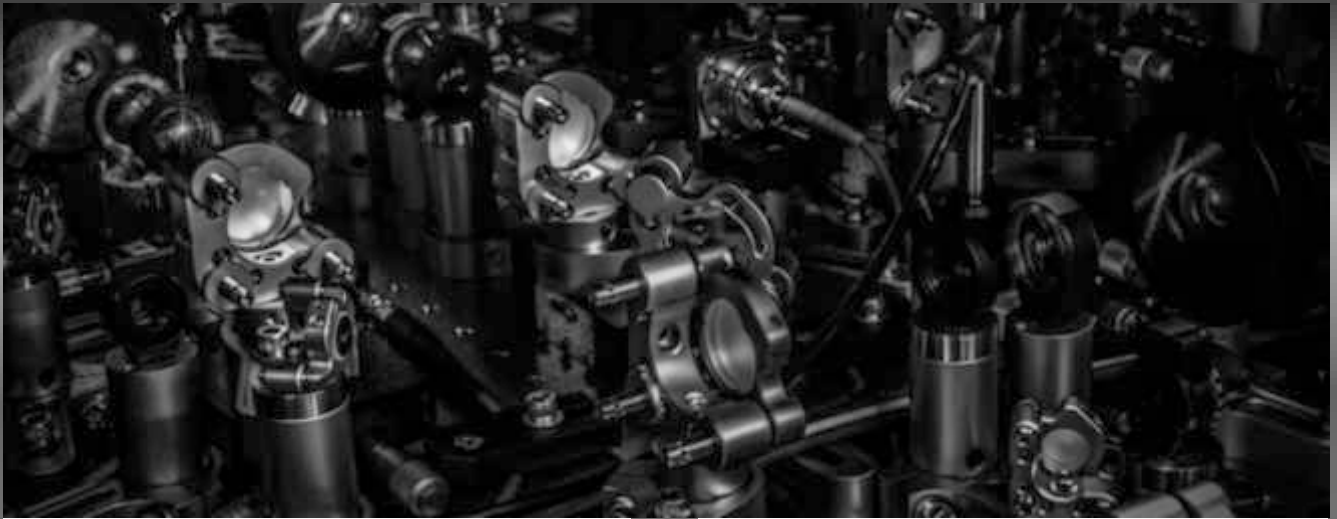

Figure 8:

Figure 9: Honeywell

QUANTUM VOLUME:
OVERVIEW OF QUANTIFYING

Measuring the capabilities of a quantum computer requires a measurement that can summarize this complex operation.
Quantum Volume is a metric that can be used to express the effectiveness of a given quantum computer.
The various quantum hardware platforms available today have a wide array of specifications, making it difficult to differentiate the machines' overall capabilities.

IBM developed a simple measurement like Quantum Volume, which is absolutely necessary. The larger the quantum volume, the more complex problems you can solve. When Honeywell releasd its quantum computer by mid-2020 its quantum volume was 128. To compare for that moment the maximum was about 50.

HOW QUANTUM VOLUME IS CALCULATED

Quite a large variety of factors and complex calculations determines the Quantum Volume.

One factor is the number of qubits in a quantum computer. Qubits are the computing bits that take advantage of quantum physics.

Other factors include errors present in the quantum operations used for calculations, how connected the qubits within the system are to other qubits, cross-talk between those qubits and the efficiency of the compiler running the operations.

Unlike the volume of a cube, Quantum Volume measures are not computed by simple multiplication, but require a complicated set of statistical tests.

Here are the main components:

❶ **Number of qubits**
The simplest measure to quantify is the number of physical qubits in the quantum computer.
In traditional computers, the bits are in a state of "0" or "1."
However, in quantum computers, the qubits can be in the states "0" or "1" or both at the same time.

That's a property of quantum physics called superposition.
The number of qubits is important because they are a basic element of the computer. But to determine the quantum computers full capability, two more factors must be considered.

❷ Error rates
When it comes to error rates, like golf, the lower the number the better.
The limiting error rate of the system quantifies how often it gets the wrong answer.
Error rates encompass systematic errors that might not show up in a single qubit.
That comprehensive error rate is important in determining how often calculations will be accurate.

❸ Connectivity of qubits
Lastly, connectivity is a key factor in determining quantum volume.
Connectivity defines which pairs of qubits can be entangled in a quantum computer.
Some hardware can directly apply entanglement operations to any two physical qubits in the system.
Other hardware can only operate on qubit pairs that are physically located next to each other.
When qubits are fully connected, they can execute algorithms more efficiently, solving problems with fewer steps and taking full advantage of the qubits' limited coherence time.

As the Quantum Volume increases, the ability to use quantum computers to solve really complicated problems will accelerate.
"It means that there will be a lot more things that we can do with quantum computing that we thought were further out on the horizon," said analyst Smith-Goodson.

Figure 10:

HONEYWELL ACHIEVED A QUANTUM VOLUME OF 128 ON THEIR QUANTUM COMPUTER IN LATE SEPTEMBER 2020.

What does that really mean?

Quantum volume is a metric that measures the capabilities and error rates of a quantum computer.

It expresses the maximum size of square quantum circuits that can be implemented successfully by the computer.

The form of the circuits is independent from the quantum computer architecture, but compiler can transform and optimize it to take advantage of the computer's features.

Thus, quantum volumes for different architectures can be compared.

In 2020, the highest achieved quantum volume (per § IBM's modified definition) rose from 32 for IBM's computer "Raleigh" to 128 for Honeywell's "H1",[2] i.e. quantum circuits of size up to 7×7 have been implemented successfully.

This and future increases in the capabilities of quantum computers will empower customers to achieve better and more results.

The differentiated technology, exemplified by the high-fidelity and fully-connected qubits with mid-circuit measurement* and qubit reuse, enables the customers to push the frontier of quantum computing applications. *(Mid-Circuit Measurement is a unique feature that allows qubits to be selectively measured at a point other than the end of a quantum circuit. The quantum information of a measured qubit collapses to a classical state (zero or one), but the non-measured qubits retain their quantum state.)
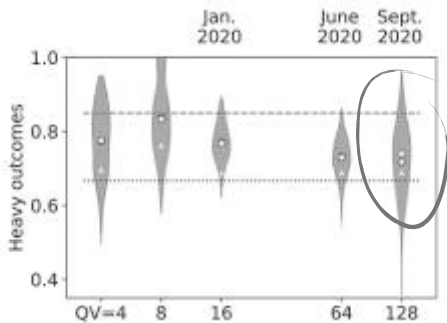

Figure 11: Plot for the outcomes of the respective Quantum Volumes

The figure of the plot at the bottom-left column shows the heavy outcomes for Honeywell Quantum Solutions' tests of quantum volume and the dates when each test passed.

All tests are above the 2/3 threshold to pass the respective Quantum Volume.

Circles indicate heavy outcome averages and the violin plots show the histogram distributions. Data colored in blue shows system performance results and red shows modeled, noise-included simulation data. White markers are the lower 2-sigma error bounds.

The system successfully passed the Quantum Volume 128 test outputting heavy outcomes 71.78% of the time, which is above 2/3 threshold with 99.934% confidence. The average single-qubit fidelity is 99.97(1)% and the average two-qubit gate fidelity is 99.54(7)% with fully-connected qubits.
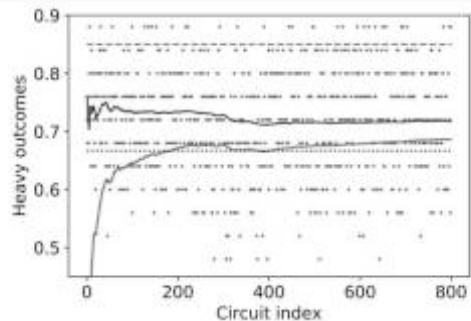

Figure 12: Plot for the heavy out comes

The figure above shows the individual heavy outcomes for each Quantum Volume 128 run. The blue line is an average of heavy outcomes and the red line is the lower 2-sigma error bar which crosses the 2/3 threshold after 186 circuits.
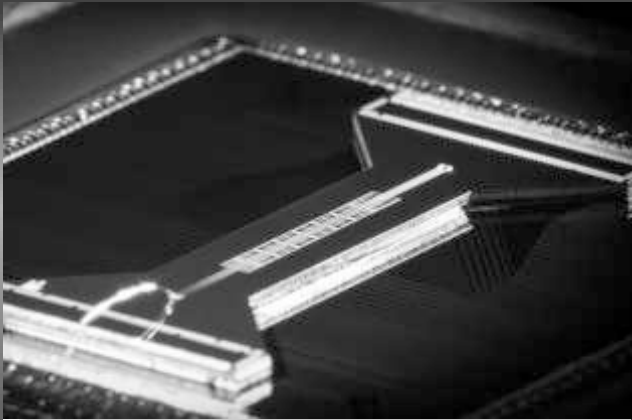
Figure 13: First-Generation Ion Trap

The new kid in town:

The Ion Trap

Currently, Honeywell Quantum Solutions *(part of Honeywell ofcourse)* has started testing its first generation of commercial micro-fabricated trapped-ion qubit devices. See the image above.

These devices, or traps, hold the qubits which are integrated into Honeywell's quantum computing systems.
Traps are fabricated in the Minneapolis microfabrication production facility.
The first generation of traps incorporates includes different arrangements, or architectures, of qubits.
The different architectures under evaluation include one-dimensional (1D) chains of ions as well as a scalable design of a two-dimensional (2D) arrays which eliminate qubit count limitations imposed by a linear qubit arrangement.
Thi provides an enabling component for Honeywell's first quantum computer.

The fabrication of traps also demonstrates a unique feature of our end-to-end quantum computing technologies. This includes all components from the traps that form computational qubits to the augmented control electronics and user interfaces.

Honeywell's quantum computer stores processing elements called qubits in a row along the center of a device called an H trap. The company is working on second-generation machines that look more like a grid than a line.

TRAPPED ION QUANTUM COMPUTER

A **trapped ion** quantum computer is one proposed approach to a large-scale quantum computer.
Ions, or charged atomic particles, can be confined and suspended in free space using electromagnetic fields.

Qubits are stored in stable electronic states of each ion, and quantum information can be transferred through the collective quantized motion of the ions in a shared trap (interacting through the Coulomb force).
Lasers are applied to induce coupling between the qubit states (for single qubit operations) or coupling between the internal qubit states and the external motional states
(for entanglement between qubits).

The fundamental operations of a quantum computer have been demonstrated experimentally with the currently highest accuracy in trapped ion systems.

Promising schemes in development to scale the system to arbitrarily large numbers of qubits include transporting ions to spatially distinct locations in an array of ion traps, building large entangled states via photonically connected networks of remotely entangled ion chains, and combinations of these two ideas.

This makes the trapped ion quantum computer system one of the most promising architectures for a scalable, universal quantum computer. one-way quantum computer, and various quantum cellular automata.

The most widely used model is the quantum circuit. Quantum circuits are based on the quantum bit, or "qubit", which is somewhat analogous to the bit in classical computation. Qubits can be in a 1 or 0 quantum state, or they can be in a superposition of the 1 and 0 states. However, when qubits are measured the result of the measurement is always either a 0 or a 1; the probabilities of these two outcomes depend on the quantum state that the qubits were in immediately prior to the measurement.

Over decades of research and development, **Microsoft** has achieved advancements across the quantum stack - including software, applications, devices, and controls.

Their approach to quantum takes a comprehensive approach to delivering all the technology needed to enable commercial impact - encompassing everything from development to deployment.

It includes major ongoing focus to develop the topological qubit to help make scalable, stable quantum computing a reality.

With the **Azure Quantum** open cloud ecosystem you can find a lot of what you might need to accelerate your app-development and quantum computing: quantum software, hardware, as well as learning resources for developers, researchers, and students.
You can find pre-built optimization solvers that borrow from quantum principles running on classical resources, and write quantum algorithms designed to run on quantum hardware.

For developers
Bring quantum apps to life with the quantum development kit for the **Q#** quantum programming language and **Azure Quantum**. In this open-source kit, you'll find tools to formulate and run optimization problems on large-scale or hardware-accelerated Azure compute resources, as well as for developing durable quantum apps for quantum hardware.

Get the quantum development kit:
`https://azure.microsoft.com/en-gb/resources`
`/development-kit/quantum-computing/`

The development kit for quantum computing and optimization
The open-source for Q# and Azure Quantum enables you to develop durable quantum applications for quantum hardware and for scalable hardware.
Optimization is a class of problems whose solutions are primary candidates for running on scalable quantum computers.
The **Quantum Development Kit** also has tools to formulate optimization problems to run on large-scale or hardware-accelerated computer resources in Azure.

Figure 14: Microsoft overview of seervices related to Quantum
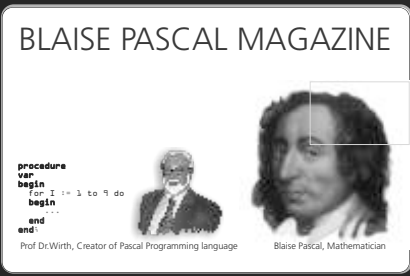
Quantum Volume and potential - applied to scope

INDUSTRY

INTELLIGENCE

RESEARCH

Chemicals & Petroleum
Surfactants, Catalysts
Chemical product design

Chemicals & Petroleum
Oil Shipping / Trucking
Refining Processes
Feedstock To Product
Drilling Locations

Distribution & Logistics
Vehicle Routing
Network Optimization

Scenario Simulation
Freight Forecasting
Irregular Behaviors (ops)
Disruption Management

Distribution Supply chain

Financial Services
Credit/Asset
Scoring

Transaction
Settlement
PortFolio
Management

Seismic Imaging   **Consumer goods**

**Diseas Risk
Predictions**

Financial Services
Derivalives Pricing
Irregular Behaviors
(Fraud)

Investment
Risk Analysis

**Artificial
Intelligence &
Machine Learning**
Quality / Control
Process Planning
Manufacturing
Supply Chain

Fabrication
Optimization

**Finance
Recommender**

**Structural Design
& Fluid Dynamics**

**Health Care
& Life Sciences**
Drug Discovery /
International licensing
Protein Structure
Prediction

**Optimization
Accelerated Diagnosis**
Clinical Trial Enhancements
Genomic Ananlyis
Medical / Drug Supply Chain

Manufacturing
Quantum Chemistry
Materials Discovery

A COMBINATIONS COUNTER
By David Dirkse

starter ▲ expert ▲ 7

INTRODUCTION
In the mathematics field of **combinatorics,** a combination is a choice of elements from a collection. The sequence of choosing is unimportant.
Note that this part of two articles is meant to preapare the for the second part: \
HASHI LOGIC PUZZLES
An example is where a team of cleaners is chosen from a group. The question is
1. How many different choices can we make
2. How to systematically generate and number these choices

First some theory.
For elements we take the characters A,B,C,D,E
These 5 elements can be ordered in
$5.4.3.2.1 = 120$ sequences.
From ABCDE to EDCBA.

Now take the case where the collection consists of 2 A's and 3 B's.

To start we make these elements different by attaching an index, so we have
A1,A2,B1,B2,B3 and 120 number of sequences again.

The A elements can be placed in $2.1 = 2$ sequences, the B elements in $3.2.1= 6$ sequences which actually are the same if the sequence is unimportant.

For the real number of sequences, without the indexes, we have to divide 120 by 2 and by 6,
so $120/(2.6)=10$ unique sequences remain.

(AABBB , ABABB , ABBAB , ABBBA , BAABB , BABAB , BABBA , BBAAB , BBABA , BBBAA)

While choosing from a group of people we may attach character N to a person which is not chosen and character Y to a person which is chosen.

If we select 3 people out of a group of 8 then we notice sequences like YYYNNNNN, YNNNYYNN.

Combinatorics is an area of mathematics primarily concerned with counting, both as a means and an end in obtaining results, and certain properties of finite structures.
It is closely related to many other areas of mathematics and has many applications ranging from logic to statistical physics, from evolutionary biology to computer science, etc.

Insofar as an area can be described by the types of problems it addresses, combinatorics is involved with the enumeration (counting) of specified structures.
Sometimes referred to as arrangements or configurations in a very general sense, associated with finite systems.
They need to be "largest", "smallest" or satisfy some other optimality criterion.

Combinatorics is a range of linked studies which have something in common and yet diverge widely in their objectives, their methods, and the degree of coherence they have attained.
One way to define combinatorics is, to describe its subdivisions with their problems and techniques. (make the problem smaller, cut it into segments).

Combinatorics is well known for the breadth of the problems it tackles.
Combinatorial problems arise in many areas of pure mathematics, notably in algebra, probability theory, topology, and geometry,
as well as in its many application areas.

In the later twentieth century, powerful and general theoretical methods were developed, making combinatorics into an independent branch of mathematics in its own right.

COMBINATORICS IS USED FREQUENTLY IN COMPUTER SCIENCE TO OBTAIN FORMULAS AND ESTIMATES IN THE ANALYSIS

One of the oldest and most accessible parts of combinatorics is graph theory, which by itself has numerous natural connections to other areas. Combinatorics is used frequently in computer science to obtain formulas and estimates in the analysis of algorithms.

A mathematician who studies combinatorics is called a combinatorialist.

So the number of combinations equals the number of ways we can write sequences of 3 Y's and 5 N's which is
`8.7.6.5.4.3.2.1/(1.2.3.4.5.1.2.3) = 56.`

{division by 3.2.1 and also by 5.4.3.2.1}

In general:
Writing 1.2.3….N as N! (called N faculty) and the number of combinations of k choices out of N elements as C(N,k) then C(N,k) = N!/(k! . (N-k)!) because there are k elements chosen and N-k element not chosen.

This project was started because I needed a combinations counter to solve logic puzzles called Hashi, also called Bridges.
A next article covers the Hashi project.

Add, Shift, Backup

It's about elements which are chosen or not. For this project I choose binary digits 0 (not chosen) and 1 (chosen) , placed in an array of bytes.

Take the combinations of 3 out of 5:

1: 11100	6: 10011
2: 11010	7: 01110
3: 11001	8: 01101
4: 10110	9: 01011
5: 10101	10: 00111

Indeed, C(5,3) = 5!/(3!)*(2!) = 10

Flowchart
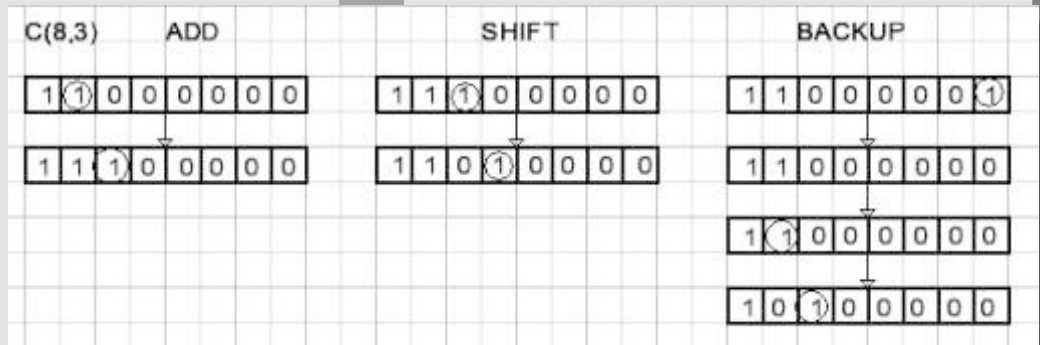(read 1 for bridges, it was about solving Hashi puzzles)

While "counting" from 00000 we notice three conditions:
1. Adding a 1, so counting 00000,10000,11000,11100: the first combination
2. Shifting the most right 1: 11100,11010,10011: three next combinations
3. Backup, if the rightmost digit is 1
   a. Remove this digit
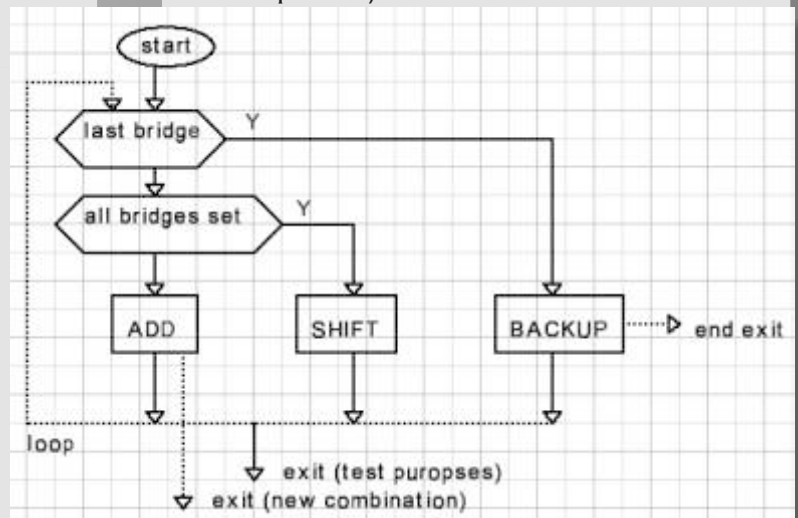   b. Search left for 1 digit
   c. Shift this digit right

Repeat steps 1..3
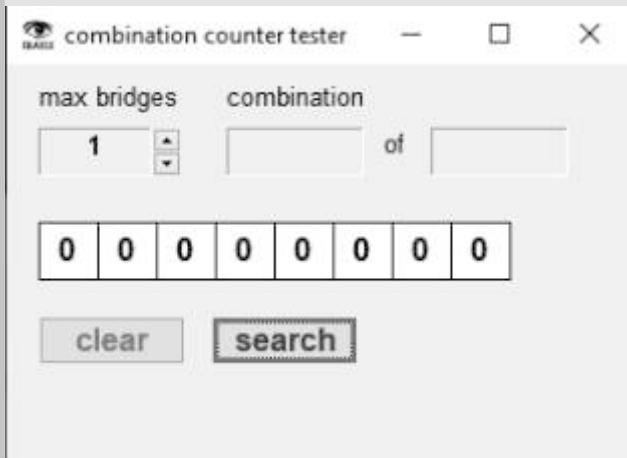The end is reached when the backup finds no more 1's.

The counter in action Delphi 7

This is the advance function which produces a new combination in the counter[ ] array on being called:

Delphi 10.4

Lazarus

```
function advance : byte;
label loop,Ladd,Lshift,Lback;
begin
Loop:
if p = 8 then goto Lback;
if count = maxcount then goto Lshift;
Ladd:
 inc(p);
 counter[p] := 1;
 inc(count);
 if count = maxcount then begin
             result := 1;
             exit;
             end;
 goto loop;
Lshift:
 counter[p] := 0;
 inc(p);
 counter[p] := 1;
 result := 1;
 exit;
Lback:
 counter[p] := 0;
 dec(count);
 while (p > 0) and (counter[p] = 0) do dec(p);
 if p = 0 then begin
        result := 0;
        exit;       //report end
        end;
 counter[p] := 0;
 inc(p);
 counter[p] := 1;
 if (8-p) < maxcount - count then goto LBack; //no space for
                        //next digits
 goto Loop;
end;
```

p : index for counter array.
Count : the number of 1's placed
Maxcount : the maximum number of 1's {k in
C(N,k)}
Note: N = 8 in this program.

Calculation of the number of combinations:

```
 function combinations(n,k : byte) : dword;
 //calculate number of combinations
 var i : byte;
   x : double;
 begin
  x := 1;
  for i := 1 to n-k do x := x * (i+k)/i;
  result := round(x);
 end;
```

The function avoids very large intermediate results.
For higher numbers of N the number of combinations explodes.
C(40,20) = 137846528800. (rounded to 100's)
Please refer to the source code for details.

7.

Hashi (Hashiwokakero) also called Bridges is a Japanese logic puzzle.

Hashiwokakero, Hashi o kakero; literally **build bridges!** is a type of logic puzzle published by Nikoli.
It has also been published in English under the name Bridges or Chopsticks
(based on a mistranslation:

the hashi of the title, 橋, means bridge;

hashi written with another character, 箸,
means chopsticks).
It has also appeared in The Times under the name Hashi. In France, Denmark, the Netherlands, and Belgium it is published under the name Ai-Ki-Ai.

RULES
Hashiwokakero is played on a rectangular grid with no standard size, although the grid itself is not usually drawn.
Some cells start out with (usually encircled) numbers from 1 to 8 inclusive; these are the "islands". The rest of the cells are empty.

The goal is to connect all of the islands by drawing a series of bridges between the islands. The bridges must follow certain criteria:
- They must begin and end at distinct islands, travelling a straight line in between.
- They must not cross any other bridges or islands.
- They may only run orthogonally (i.e. they may not run diagonally).
- At most two bridges connect a pair of islands.
- The number of bridges connected to each island must match the number on that island.
  The bridges must connect the islands into a single connected group.

Below is pictured an initial puzzle (left) and the solved state (right):
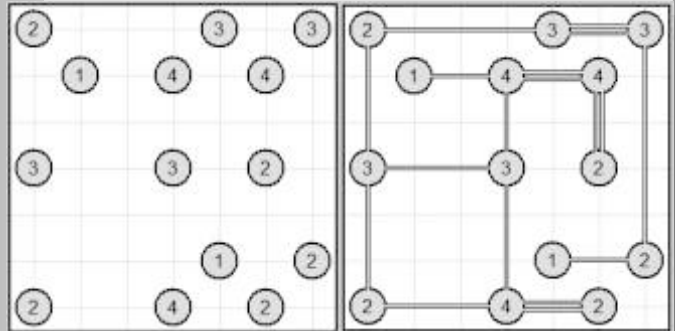
The circles are islands which have to be connected by bridges.
The number in the circle is the sum of bridges connecting to the island.
A maximum of two bridges may connect two islands. Bridges may not cross.
The puzzle is solved when all bridges have been installed and all islands are interconnected.
Hashi puzzles exist in dimensions from 7x7 to 25x25.
The difficulty ranges from easy to extreme.

PASCAL DELPHI /LAZARUS PROJECT.
The purpose of this Hashi project is to:
- Replace pencil, eraser and paper
- Save and open puzzles
- Supply warnings
- Supply hints
- Supply full solutions
- Test for multiple solutions

Coding the puzzle
The puzzle is divided in squares (fields),
a field may contain:
- nothing
- An island
- A horizontal bridge
- A vertical bridge

```
type TFieldType = (ftNone,ftIsland,ftHbridge,ftVbridge);
  TField = record
     ftype  : TFieldType;
     Idx    : byte;   //entry in island list
     brCount : byte; //bridgecount for bridge and islands fields
     end;
const maxgame = 15;      //maximum game size 15x15
  maxIsland = 100;       //maximum number of islands
var game : array[1..maxgame,1..maxgame] of TField;
```

This supplies enough information to paint the game, add/remove islands by mouseclicks, add/remove bridges by mouseclicks or – movements.
A two dimensional array with hidden islands and bridges however is inconvenient to generate warnings and hints or to recognize a solved puzzle.

It is more convenient to have the islands separately listed in an array.

In above Tfield we notice a variable Idx which points to an island in the island[ ] list.

Instead of (x,y) to indicate an island in the game[.] array, now one number suffices to identify an island in the island array.
Also this list holds the number of bridges installed in each direction and the island indexes of the neighbours. The island list is made when play mode is switched on.

```
type TVector = array[1..4] of byte;//[1] right; [2] up; [3] left; [4] down
   TIsland = record
      x,y    : byte;  //field coords in game array
      brCount : byte;  //nr of bridges to install
      brConn  : Tvector; //bridges connected per direction
      neighbrs : Tvector; //[right,top,left,down] neighbour
      groupNr : byte;  //for isolation and solved puzzle testing
      end;
var Island : array[1..maxIsland] of TIsland;
   IslandCount : byte; //total islands present
   BridgeCount : byte; //total bridges added in game
   maxBridgeCount : byte; //total bridges to add
```

NOTE:
the number of bridges to install is the sum of all islands bridge counts divided by two.
So, if this sum is odd, the puzzle cannot be solved.
All islands are placed in an Island[ ] list.
Picture below shows an island with 4 neighbours:
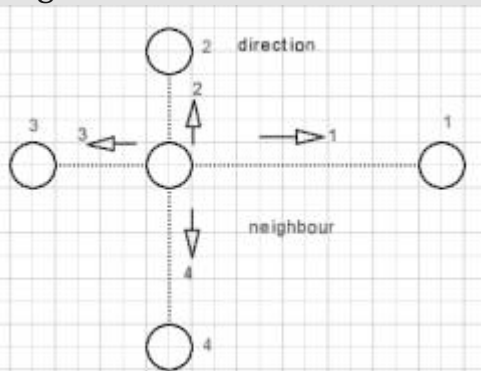
In this list we see the neighbours vector neighbrs[ ] which holds the island list index of the 1 to 4 neighbours,
Vector brConn[ ] holds the number of installed bridges for directions 1..4.
x,y are the coordinates of the island in the game[,] array;
The byte variable groupNr serves the checking for a solved puzzle and also the recognition of an isolated island or group of islands. See later.
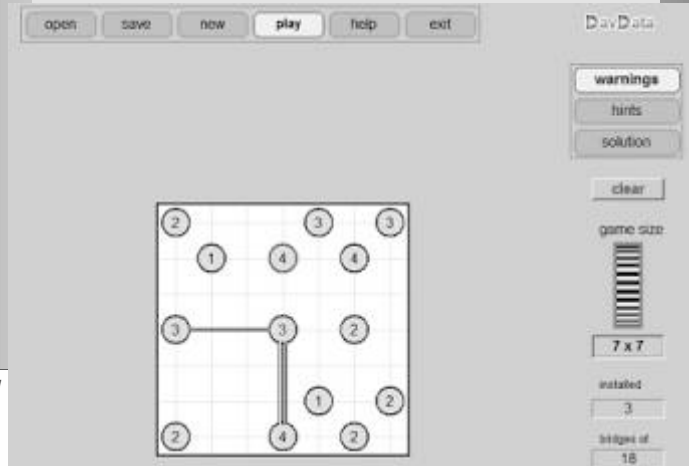
There are two type of warnings:
1.   An island is unable to place all of the required bridges

cannot add all bridges
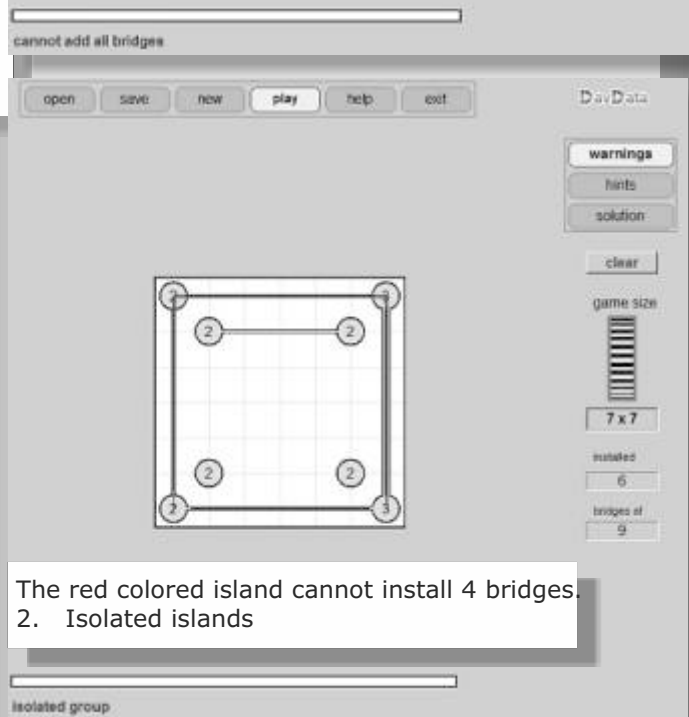
The red colored island cannot install 4 bridges.
2.   Isolated islands

isolated group

Before descending into the generation of warnings and hints, good notation is convenient.
B is the number of bridges to install at an island.
a_i {i: 1..4} is the number of bridges (0..2) that neighbour i is able to supply.
$A = a_1 + a_2 + a_3 + a_4$

NOTE: if a vertical bridge crosses between horizontally oriented islands, the a value is zero.
For an island, a warning (type 1) must be generated if A < B.
To check for isolated groups (warning type 2), first the groupNr variable is set to the island list index, so each island has its own unique group number.

Then scanning the island list, the connection with neighbours 1 and 4 (right, down) is examined.

If a connection exists, the island with the highest group number has his group number replaced by the lowest group number.

Repeating this process until no group number changes have occurred, the case of a solved game is where only one group (nr. 1, of course) remains.
If more than one group remains, the groups are isolated.
To check for isolated groups, a new list is built:

```
const maxgroup = 100;
type TGroupData = record
        IsGCount : byte;      //islands group count
        BrSum : byte;          //bridges summed
        BrConnected : byte; //actual bridges
        end;
var grouplist : array[1..maxgroup] of TGroupData;
```

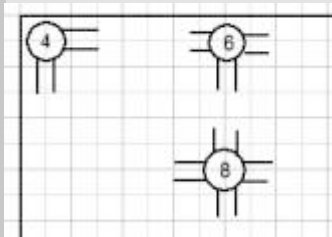IsGcount : number of islands in the group.
brSum : bridges to install in the group.
brConnected : actual number of bridges installed in the group.
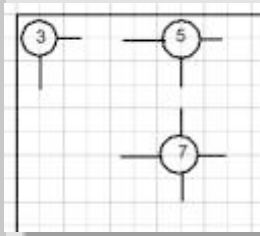The groupNr variable in the island[ ] list is the index for the grouplist.

HINTS
Several websites offer Hashi puzzles and supply instructions how to solve them.
These cases are obvious:

4 in the corner, 6 on a side, 8 in the center: 2 bridges are needed in all directions.
At least one bridge must be present in these cases:

3 in the corner, 5 at a side and 7 in the center require at least one bridge per direction. We feel however that the hints above are in fact examples of one and the same general rule. Let's investigate.
Starting at an island without any bridge, we count for each neighbour the number (a) of bridges it allows to connect. This depends on the number of bridges already installed at this neighbour, its own bridge count and a possible blockade by bridge crossings.

$$A = a_1 + a_2 + a_3 + a_4$$
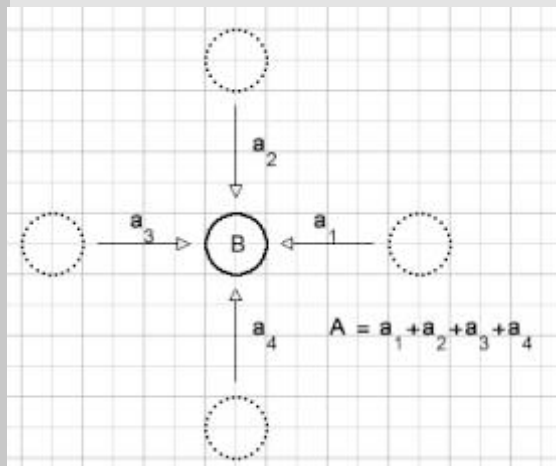
We ask the question: "can we skip painting a bridge in direction 1"?
This is the case if directions 2,3,4 supply enough bridges
$B <= a2+a3+a4$ or
$B <= A-a1$ so
$A-B-a1 >= 0$
For direction i, in case of
$A-B-ai = -1$ 1 bridge is missing
$A-B-ai = -2$ 2 bridges are missing.
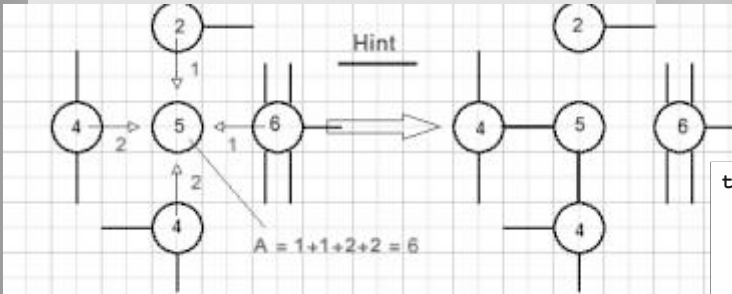So if pi is the number of bridges to paint in direction i we see:
$-pi = A-B-ai$
$pi = (B-A) + (ai - bi)$
If bi are the bridges already installed in direction i.
Negative values of p are replaced by 0.
Example:

Hint

A = 1+1+2+2 = 6

$p1 = p2 = (5-6) + 1 = 0$
$p3 = p4 = (5-6) + 2 = 1$

Example (reduced picture)

hint stop. Press <SPACE> to add bridge, <ESC> to ignore

FULL SOLUTIONS

So called extreme puzzles reach a point where no hints can be generated.
In these cases a so called brute force approach always is able to find a solution.

Brute force means that analytic reasoning is abandoned. Simply all possibilities are tried. Problem however is how to count all possibilities in a systematical way, being sure that no one is skipped.
Also processing time must be kept within reasonable boundaries.
So, assume we have reached the point in solving a Hashi puzzle where no hint is found but some bridges are still missing.
Of all possibly missing bridges a list is generated. In most cases this list will be longer than the actual number of bridges that need to be installed.
If 2 bridges may connect to an island in the same direction, 2 identical entries are made in the bridgelist.

```
type Tline = record
       x1,y1,x2,y2 : byte;   //islands coords of bridge
       end;
  TBridge = record
       line : Tline;         //x1,y1 x2,y2 islands
                               coordinates in game[ ]
       AB  : byte;           //0: no bridge; 1: bridge present
       Skip10 : boolean;     //skip double 01-10 counts
       end;
var bridgelist  : array[1..50] of Tbridge;
  maxbridgelist : byte;   //top of list
```

If the solve button is pressed an entry in the bridgelist is made for each bridge which may possibly be added.
Say this are 10 entries for 10 possible bridges. However, we know how many bridges were added already, say we have 4 bridges to go. This means that we have to select 4 bridges out of 10 in a systematical way to see if any selection represents a solution.
In math this is known as a combination of 4 out of 10.
This number of combinations is written as $C(10,4) = 10!/(4!)*(6!) = 210$.
Please refer to my previous article about combinations counting.
The skip10 variable in the bridgelist is set for the second of two identical entries.

In this case bridge selections 01 and 10 are the same, so counting must be 00,10,11 skipping the 01 selection.(do not select bridges, select 1 bridge, select both bridges).
Byte array CC[ ] is the combinations counter.
1: bridge; 0:no bridge
Next picture shows what to do after a CC counter update:
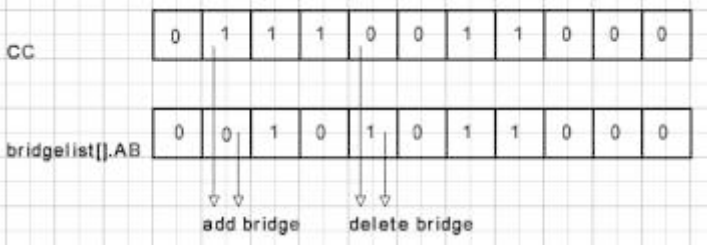
For each new combination in CC bridges are added/removed from the island list and game[,]
Then a check for solution is made. The number of bridges in the bridgelist is limited to 40 to avoid extremely long processing times.

Multiple solutions
These are simply checked by generating more combinations and testing for solutions.
When the last combination is reached,
no more solutions are possible.

This Delphi/ Lazarus project.
The menu buttons are supplied by the `TDavArrayBtn` component, see my Math and Delphi programming book.

The gamesize selection is made by a `TDavRotation` button component, which imitates buttons found on labotory equipment. See an earlier publication.
Space in this article is limited so I do not show code, please refer to the Delphi project.
Variable gamesize holds the selected size of the Hashi puzzle.
Array game[1..15,1..15] uses only fields 1..gamesize.
The game is painted in bitmap Gmap, which is created in the onCreate event.
Gmap is copied to Paintbox1 to become visible.

Hints and warnings are painted directly on paintbox1 of form1.

The effect of events from keyboard, mouse or buttons depends on the gamestate.
```
Type TGamestate =
(gsInitial,gsNew,gsPlaying,gsHintWaiting,gsSolving);
```
`gsNew`: allows adding/removing islands in game[,]
`gsPlaying`: allows adding/removing bridges, selection of warnings and hints.
`gsHintWaiting`: a hint is found, waiting for user decision: install or neglect.
`gsSolving`: searching for solution in progress.

Gamestates are updated by procedure `controlmessage(cm : Tcontrolmesage)`
This is the central place to change the gamestates.
Mouse- and keyboard events and also button clicks call the controlmessage procedure with a message indicating the originator. Then the controlmessage procedure, after approval, calls a specific procedure to do the job.
Procedure controlmessage in fact is a table like construct of case statements:
Imagine the gamestate as row, the message type as a column.
This structure may look exaggerated at first (*why not performing functions directly in the event methods*) but centralizing control adds to the clearness and increases maintainability. The project has 6 units:

```
Unit1:
     event handling, control,
     adding/removing islands
Game_unit:
     for testing of a solved puzzle, warning
     and hints, adding or removing bridges
Paint_unit:
     painting procedures for Gmap bitmap
Solve_unit:
     procedures to solve a puzzle by brute
     force, combinations counter
Progress_unit:
     procedures to make progress bar in
     paintbox progressbox on form1
IO_unit:
     procedures to save and open puzzles.
```

While adding islands in the `gsNew` gamestate, checks are made to keep a distance between islands and avoids `bridgecounts` which are too high. Also, islands having bridges connected cannot be changed.

Progressbar
During the search for solution counter `CCupdates` in the `solve_unit` counts the combinations.
`Maxcombinations (unit1)` is the total number of combinations calculated earlier. The value `CCupdates/maxcombinations *100` is passed to the `setProgress(perc : single)` procedure in the `progress_unit`.

Hashi Help
Most buttons are self explanatory.

Adding/removing an island:
Menu: New
Mousedown on game places island with bridgecount 1.
Next left-mouseclicks increment the bridgecount. Right-mouseclicks decrement the bridgecount.
A bridgecount of 0 removes the island.

Adding/removing bridges
Menu: Play
Position mousepointer over island. Press left mousebutton and hold down while moving mouse in direction of bridge. If bridge appears, release mousebutton.
Repeating this procedure adds another bridge. Also a second bridge may be added by moving over an existing bridge and pressing the left mousebutton. Pressing the right-mousebutton removes the bridge.

Clear button
If menu:new then all islands are removed.
If menu:play then all bridges are removed

Save/open
The saved hashi files have no extension.
For easy recognition, the filename is prefixed by hashi!

Solving a puzzle
1. On your own by adding bridges
2. Pressing the warning button to report errors
3. Pressing the hint button to receive hints which solve most puzzles
4. Pressing the Solve button if hints are not found.

For bridges added by hints, there is simply no choice, these bridges are always necessary even in case of multiple solutions.
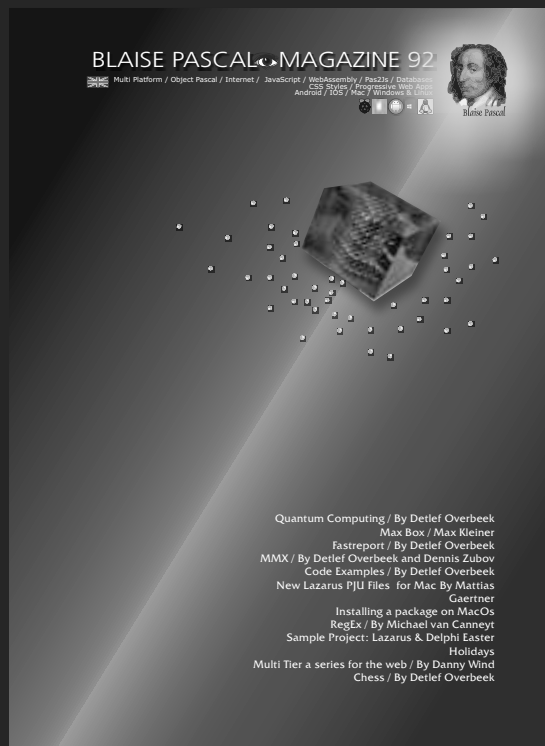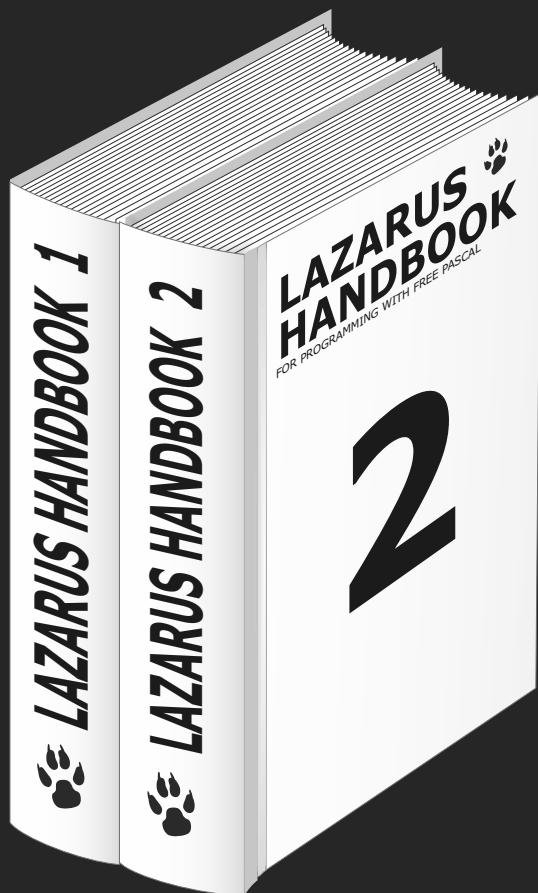The solve button may be pressed any time during play mode.
It does not change any installed bridge but tries to find a solution by adding missing bridges.
For very hard puzzles, start with adding bridges indicated by hints and press the solve button if no hints are found.

By Michael van Canneyt

ABSTRACT
Regular expressions should be a part of every programmer's toolbox. Once grasped, they are easy to use and can accomplish otherwise difficult or time-consuming tasks for you.
A short introduction in Regular Expressions.

REG EX & PASCAL

starter ————— expert

INTRODUCTION

Regular Expressions have been around for almost 70 years, and are commonly used to perform search and replace operations.
They can be found in most IDEs or text editors.
Still, surprisingly many people do not know how to use them. Therefor we give a simple introduction on how to use them.
In what follows, the POSIX conventions for regular expressions are described.
Some editors and IDEs use slightly different conventions.
A good overview of all possibilities (and in particular the Perl Compatible Regular Expressions (PCRE)) can be found on the wikipedia page:

`https://en.wikipedia.org/wiki/Regular_expression`

Many programmers will be familiar with pattern matching for filenames:
A mechanism used in the `FindFirst/Findnext` calls: They allow for simple wildcard matching, where `*` stands for an arbitrary sequence of characters, and `?` stands for a single character.

Thus `?.pas` will match all filenames of 1 character long with extension `.pas`, and `System.*.pas` will match all filenames that start with the System namespace.
Similarly, many programmers fluent in SQL will know the `LIKE` operator that allows the use of `%` and `_` wildcards.
So, how to match these filenames with regular expressions ? Like in filename wildcard matching, letters and digits (and many symbols) in a regular expressions match only that letter or digit. But a regular expression can contain some special characters.
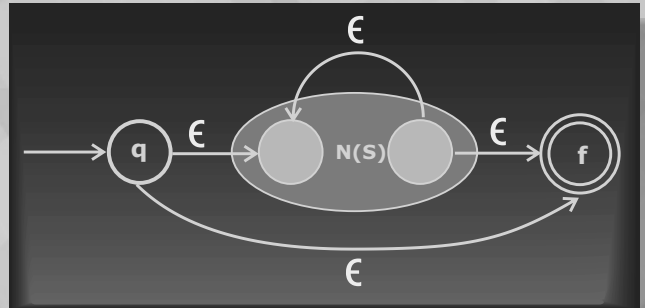

Figure 1: Trapmoth - Own work - Altered by Overbeek
A nondeterministic finite automaton (NFA) used as a construction block for Thompson's construction algorithm converting a regular Kleene Star expression into a NFA.

A regular expression (shortened as regex or regexp; also referred to as rational expression) is a sequence of characters that define a search pattern. Usually such patterns are used by string-searching algorithms for "find" or "find and replace" operations on strings, or for input validation. It is a technique developed in theoretical computer science and formal language theory.

The concept arose in the 1950s when the American mathematician Stephen Cole Kleene formalized the description of a regular language. The concept came into common use with Unix text-processing utilities. Different syntaxes for writing regular expressions have existed since the 1980s, one being the POSIX standard and another, widely used, being the Perl syntax.

Regular expressions are used in search engines, search and replace dialogs of word processors and text editors, in text processing utilities such as sed and AWK and in lexical analysis. Many programming languages provide regex capabilities either built-in or via libraries.
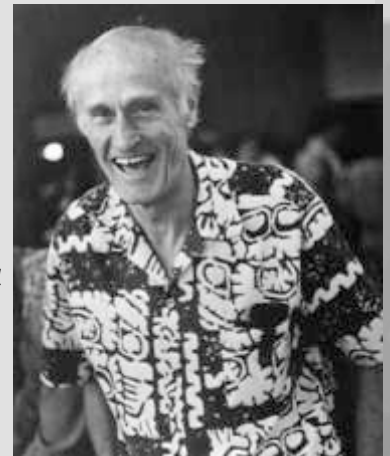

Figure2: American mathematician Stephen Cole Kleene

Here are some of them:
- `^` The `^` matches the beginning of the input string (usually a line of text).
- `.` The dot `.` matches any single character. It is equivalent to the `?` wildcard in filename matching.
- `*` The asterisk matches the previous element any number of times (also zero).
- `$` The `$` matches the end of a search string (usually a line of text).

If you want to include a literal dot or asterisk in your regular expression, you must escape them with a backslash.
With these 2 characters, we can already emulate the filename pattern matching.

The following regular expression:
`^.\.pas$`
This will match all filenames with a name of 1 character long, and extension `.pas`.

Likewise, the following:
`^System\..*\.pas$`
will match all filenames that start with the System namespace.

However, you can do more with Regular expressions than just emulate file name wildcard matching. For instance, you might want to search for files that are either a `.pas` file or a `.dcu` file. To be able to do this, we need 2 more concepts in Regular Expressions:

`( )` The brackets mark a subexpression.

`|` The vertical bar matches the element before or the element after.

To find either a `.pas` file or a `.dcu` file, we can combine these two with
`^.*\.pas|dcu$`

If you want to find files that end on
`-1, -2` or `-3`, then you can use the following operator:

`[ ]` This will match any letter between the brackets.

`[^ ]` This will match any letter not between the brackets

You can specify a range of characters by separating them with a dash, for example a-z or 0-9, just as you specify a range in Pascal.

Thus, the following expression will match all .txt filenames that end on `-1`, `-2` or `-3`:
`^.*-[123]\.txt$`
The following will also do:
`^.*-[1-3]\.txt$`

The POSIX standard specifies also several named ranges of character sets.
For example
`[:alpha:]` denotes all alphanumerical characters, whereas
`[:digit:]` denotes all digits and
`[:space:]` denotes whitespace.
Some languages (notably, Perl) or editors use escaped characters
for this: `\w` means words, `\s` means spaces etc..

NOTE that there are no operators to specify **case-insensitive** or **case sensitive** match: this kind of property of a search must be specified separately.

2 SEARCH AND REPLACE

The Delphi IDE and the Lazarus IDE both allow you to use regular expressions:
the search (and replace) dialog has a checkbox 'Regular expression'. When checked, the IDE will interpret the text in the search box as a regular expression.
For example

`^ *F.* : .*;$`
Will match all field or variable definitions starting with an F. (they must have a space before
and after the colon)
`Items\[.*\]:=`
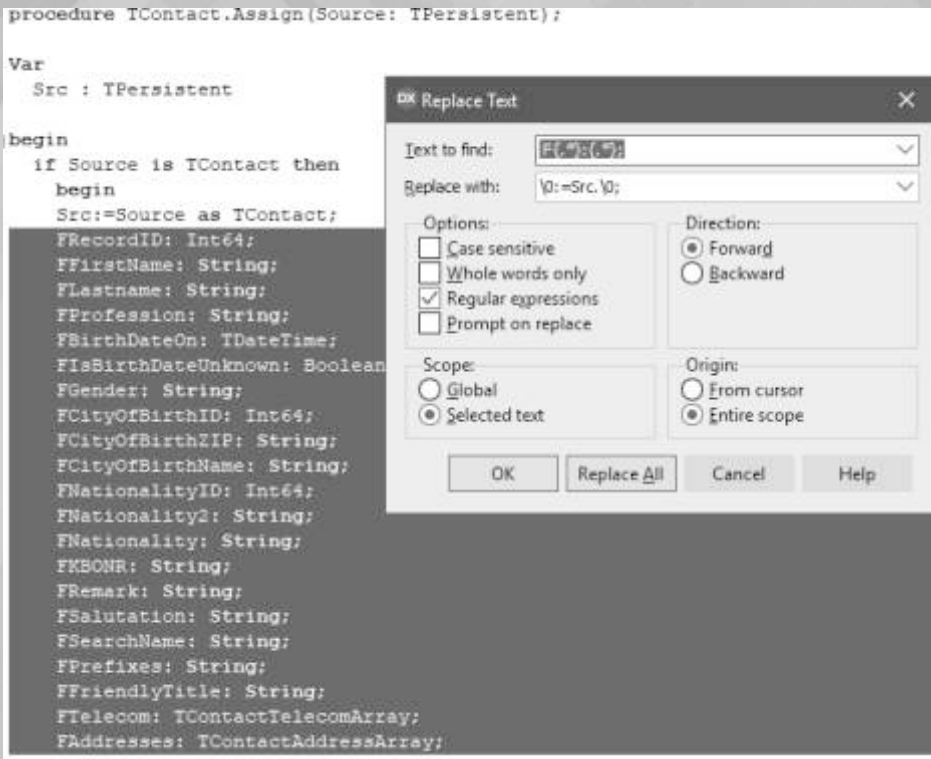will find all assignments to an Items array element.

```
procedure TContact.Assign(Source: TPersistent);

Var
    Src : TPersistent

begin
    if Source is TContact then
        begin
        Src:=Source as TContact;
        FRecordID: Int64;
        FFirstName: String;
        FLastname: String;
        FProfession: String;
        FBirthDateOn: TDateTime;
        FIsBirthDateUnknown: Boolean
        FGender: String;
        FCityOfBirthID: Int64;
        FCityOfBirthZIP: String;
        FCityOfBirthName: String;
        FNationalityID: Int64;
        FNationality2: String;
        FNationality: String;
        FKBONR: String;
        FRemark: String;
        FSalutation: String;
        FSearchName: String;
        FPrefixes: String;
        FFriendlyTitle: String;
        FTelecom: TContactTelecomArray;
        FAddresses: TContactAddressArray;
```

Replace Text ✕

Text to find: ▮▮▮▮▮▮
Replace with: \0:=Src.\0;

Options:
☐ Case sensitive
☐ Whole words only
☑ Regular expressions
☐ Prompt on replace

Direction:
◉ Forward
○ Backward

Scope:
○ Global
◉ Selected text

Origin:
○ From cursor
◉ Entire scope

OK | Replace All | Cancel | Help

Figure 3: Search and replace in an Assign method

In the IDE you will usually want to use the replace functionality when using regular expressions.

When you want to replace the found matches with something else, you will most likely want to refer to part of the matched term in your replacement text.
In order to do so, you can use the { } brackets to group particular parts in your expression.
You can refer in your replacement text to these groups by \0 to \9.

A good example where this can be useful is the Assign method, which must be implemented in descendants of **TPersistent** to copy all properties from one to another instance.

The text and the search and replace dialog to do so are shown in figure 3 on page 3/8 of the article, page 43 of the issue.
When implementing such a method you start by copying all private fields that make up your persistent class to the body of your Assign method.

Keep the list selected, and in the search and replace dialog, you can enter the following:
F{.*} : {.*};
This will match all field definitions in the selection.
Because we used the grouping operators, we can use the group in the replace text:
\0 := Src.\0;
This will replace something like
FFirstName : String;
With:
FirstName := Src.FirstName;
When we hit 'Replace all' the IDE will do all replacements, and the result can be seen in figure 4 on page 4/8 of the article, page 44 of the issue.

```
if Source is TContact then
  begin
  Src:=Source as TContact;
  RecordID:=Src.RecordID;
  FirstName:=Src.FirstName;
  Lastname:=Src.Lastname;
  Profession:=Src.Profession;
  BirthDateOn:=Src.BirthDateOn;
  IsBirthDateUnknown:=Src.IsBirthDateUnknown;
  Gender:=Src.Gender;
  CityOfBirthID:=Src.CityOfBirthID;
  CityOfBirthZIP:=Src.CityOfBirthZIP;
  CityOfBirthName:=Src.CityOfBirthName;
  NationalityID:=Src.NationalityID;
  Nationality2:=Src.Nationality2;
  Nationality:=Src.Nationality;
  KBONR:=Src.KBONR;
  Remark:=Src.Remark;
  Salutation:=Src.Salutation;
  SearchName:=Src.SearchName;
  Prefixes:=Src.Prefixes;
  FriendlyTitle:=Src.FriendlyTitle;
  Telecom:=Src.Telecom;
  Addresses:=Src.Addresses;
  end
```

Figure 4: The completed Assign method

3 USING REGULAR EXPRESSIONS IN DELPHI OR LAZARUS

Newer versions of Delphi come with the RegularExpressionsCore unit which contains the TPerlRegEx class.
It uses an external library which contains the actual implementation of the regular expression engine.

The commercial tool RegexBuddy is a GUI program that assists you in creating a regular expression and can generate the necessary Delphi code for handling the regular expresssion.

There is also an open source all-native Object Pascal implementation of regular expressions:
`https://github.com/andgineer/TRegExpr`
It is implemented by **Andrey V. Sorokin,** has been around for a long time, works for all versions of **Delphi,** and also for **Lazarus/Free Pascal.**
In fact, Free Pascal ships with a version of `TRegexpr`. Therefor, we'll use this implementation to demonstrate how regular expressions can be used.

The `TRegexpr` source code comes with a demo application **REStudio**, which is similar in purpose to **RegexBuddy.** It needs some care, but after some small fixes it can be made to work in Delphi.
The necessary fixes to let it compile in Lazarus and Delphi Rio (and newer) have been pushed to the following repository:
`https://github.com/mvancanneyt/TRegExpr`
When started, it looks like figure 5 on page 5/8 of the article, page 45 of the issue.
Its working is quite simple: in the top memo, you can enter a regular expression.
The program will give you feedback, so you will immediately see if the expression is valid.
The bottom memo allows you to enter a text, and with the **Exec.../ExecNext** buttons you can let the regular expression execute its action.

The `TRegExpr` class is not a component.
It must be created in code, and has the following properties:

- **Expression**
  The regular expression.
- **ModifierStr** (string)
  set the regular expression modifier options as a string (as one would in Perl or Javascript). The string consists of a concatenation of the uppercase letter of the next properties.
- **ModifierI (boolean)**
  The I modifier, for case insensitive search.
- **ModifierR (boolean)**
  The R modifier: enables some extented Russian characters.
- **ModifierS (boolean)**
  The S modifier: when True, the . character also matches a newline.
- **ModifierG (boolean)**
  The G modifier: turn on greedy matching.
- **ModifierM (boolean)**
  The M modifier: enables multiline mode.

  In that mode, the ^ and $ characters match the beginning and end of each line in the search text, not just the beginning and end of the whole search text.
- **ModifierX (boolean)**
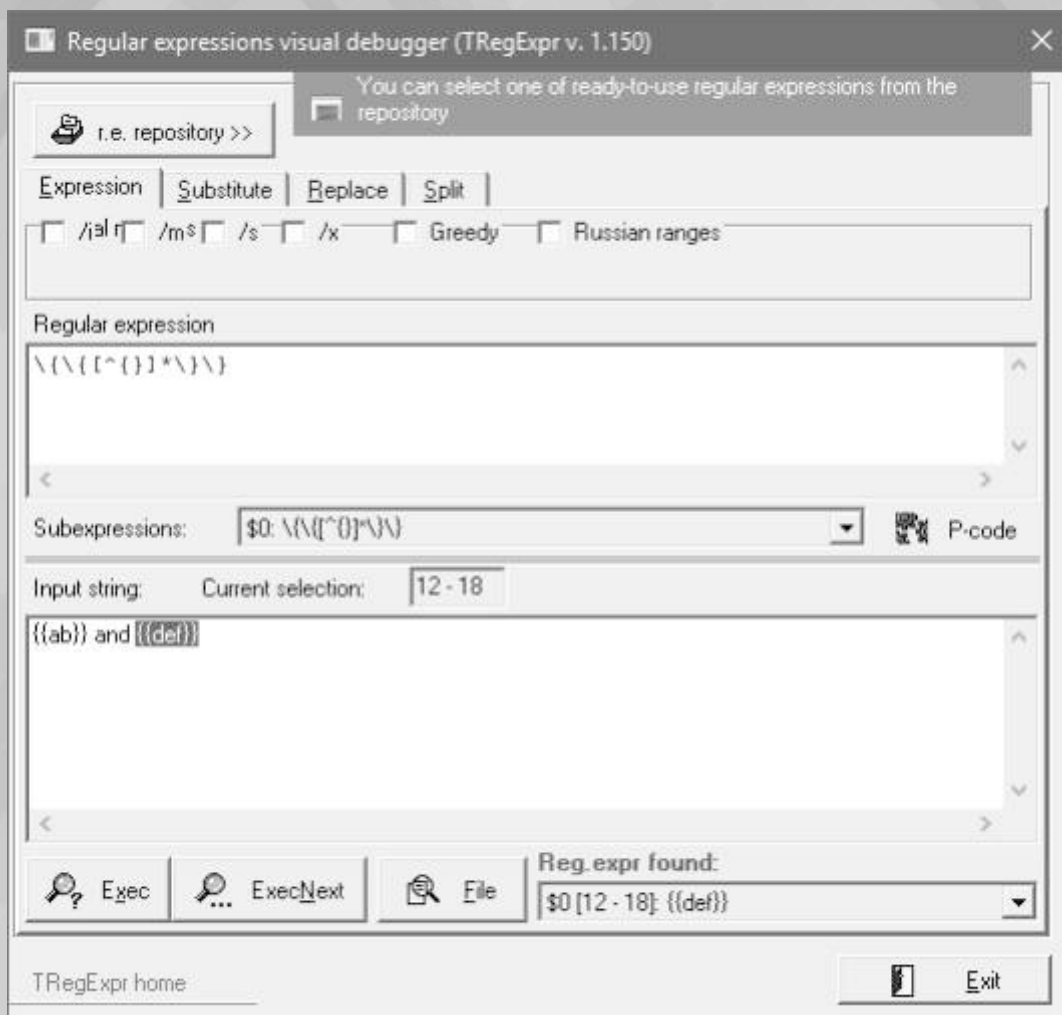 The X modifier: Allows you to use comments in your regex using the # character.

REG EX & PASCAL

Figure 5: RE Studio in action

- **SubExprMatchCount (integer)**
  the amount of sub expression matches
  after an **Exec** or **ExecNext.**
- **MatchPos (array of integer)**
  a 0-based array giving you the position of
  the subexpressions in the search string.
  The 0-th entry is the position of the whole
  regular expression match.
- **MatchLen (array of integer)**
  a 0-based array giving you the lengths of
  the subexpressions in the search string.
  The 0-th entry is the length of the whole
  regular expression match.
- **Match (array of string)**
  a 0-based array giving you the actual
  matches of the subexpressions in the
  search string. The 0-th entry is the text of
  the whole regular expression match.

- **CompilerErrorPos**
  if an error occured during compilation of
  your Regular Expression, this gives the
  position of the error.
  There are many more properties, but the
  above ones are the main properties.
  The class also has the following important
  methods:

```pascal
function Exec(const AInputString: RegExprString): boolean;
function ExecNext: boolean; overload;
function ExecNext(ABackward: boolean): boolean; overload;
function Substitute(const ATemplate: RegExprString): RegExprString;
procedure Split(const AInputStr: RegExprString; APieces: TStrings);
function Replace(const AInputStr: RegExprString;

const AReplaceStr: RegExprString; AUseSubstitution: boolean = False) // ###0.946 : RegExprString; overload;
function ReplaceEx(const AInputStr: RegExprString; AReplaceFunc: TRegExprReplaceFunction): RegExprString;
```

- **Exec**
  Starts a search for matches on `aInputString`. Returns true if a match was found.
- **ExecNext**
  Will search for the next occurence.
  If `aBackWard` is true, the search is done backwards. Returns true if a match was found.
- **Substitute**
  will replace the text in `ATemplate` with the currently found matches:
  The template can contain placeholders for the subexpressions:
  `$&` or `$0` is replaced by the whole match, and `$1` till `$n` are replaced by their respective subexpression.
- **Split**
  will split `aInputStr` into various items split by the regular expression matches, and put all found items in `aPieces`.
- **Replace**
  will replace aInputStr all the matches with the `AReplaceStr` term. If `AUseSubstitution` is true, the replacement term is not treated as a regular text, instead it is used as a pattern for the Substitute to replace all items. The function returns the input string with all matches replaced.
- **ReplaceEx**
  will replace `aInputStr` all the matches with the result of the `AReplaceFunc` callback; the callback is called for each match in the input string, and the result of the callback is then used to replace that particular match in the input string. This is a very powerful mechanism.

So, how can we use this class to implement search and replace?

To demonstrate how to search, we create a small application in Lazarus with a memo `(mdemo)`, a button `(bsearch)` and a `TFindDialog (FDRE)`. The `OnClick` handler of the button shows the find dialog:

```pascal
procedure TMainForm.btnSearchClick(Sender: TObject);
begin
  FDRE.Execute;
end;
```

This is nothing special.
The `OnFind` event handler of the find dialog is called every time the user clicks the `Find` button in the find dialog. Here we implement the logic for searching.
Because the `TRegexpr` has 2 methods to search, depending on whether it is the first time you search `(Exec)` or it is a next search `(NextExec)`, we must keep track of whether it is a new search or the next time a search is done.
To do so, we store the regular expression in a variable, and if the regular expression has changed, we assume it is a new search:

```pascal
procedure TMainForm.FDREFind(Sender: TObject);
var
  Found: Boolean;
begin
If FDRE.FindText<>FLastRE then
  begin
    FreeAndNil(FRE);
    FRE:=TRegexpr.Create;
    FLastRE:=FDRE.FindText;
    FRE.Expression:=FLastRE;
    Found:=FRE.Exec(MDemo.Lines.text);
  end
else
  Found:=FRE.ExecNext;
if Found then
  begin
    MDemo.SelStart:=FRE.MatchPos[0]-1;
    MDemo.SelLength:=FRE.MatchLen[0];
  end
else
  ShowMessage('Search term not found');
end;
```
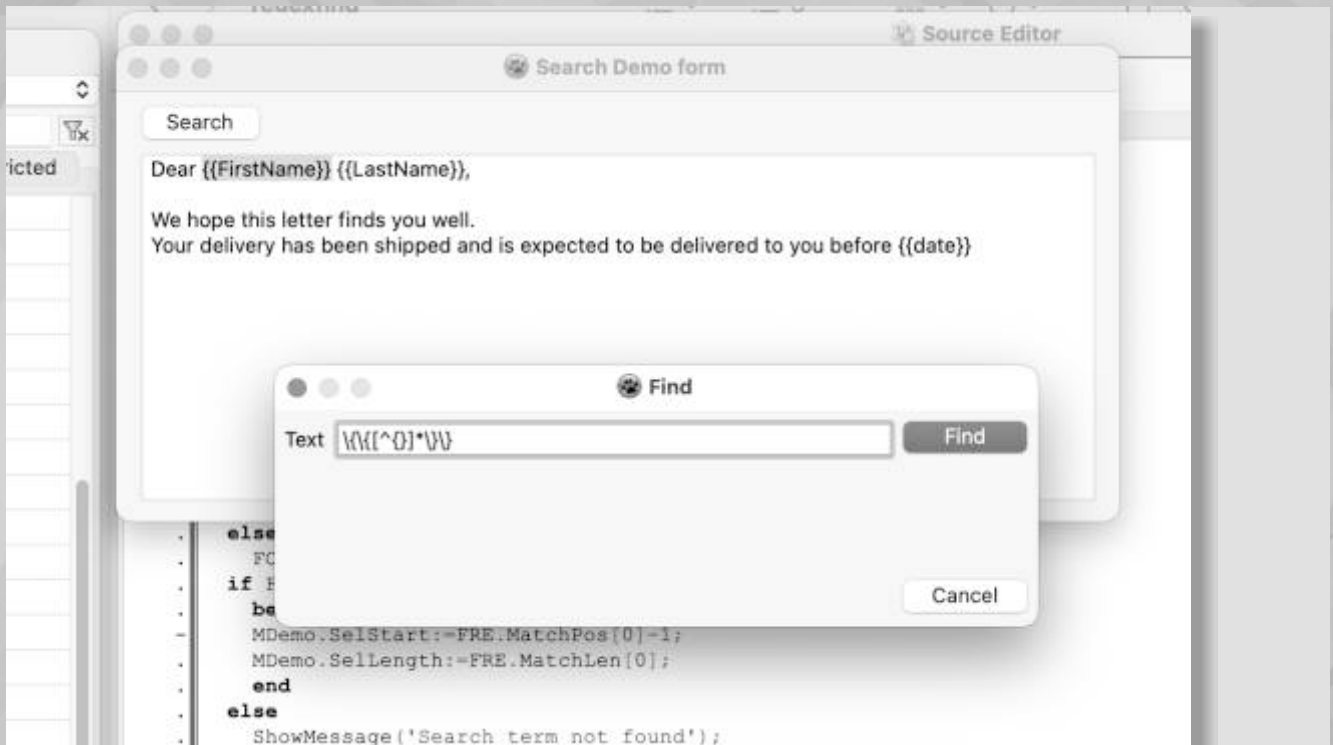
Figure 6: Search in action

When we find a match, we use the `MatchPos` and `MatchLen` properties to set the selection of the memo. (note that `Matchpos` is 1-based, and `SelStart` is 0-based) If no match is found, we show a message, but we don't close the dialog.

NOTE that if the user changes the text between searches, the matches will no longer correspond to actual positions, so in a real program, it would be best to make the memo readonly while the search is active. The result of this can be seen in figure 6 on page 7/8 of the article, page 47 of the issue. To demonstrate the power of the TRegexpr class, we'll also implement a small algorithm that transforms a template text with some data placeholders to a ready-to-read text: a kind of mailmerge, or the kind of functionality often found in templating engines such as Mustache. A template engine such as Mustache will replace a template `{{SomeVariable}}` with the value of `SomeVariable`. Usually more complex logic is also available, but for this example, we limit ourselves to simple variables, which are kept in a `Key=Value` list.

This time we use a Delphi program. We drop 2 `TMemo` components on it, one for the template (`MTemplate`), one for the result (`MResult`). We also drop a `TValueListEditor` on the form. Here the user can enter the values for the template variables. Last item is a button (`Generate`) to generate the final text based on the template and the user-provided variables and their values. The `OnClick` event of the button is implemented with the `ReplaceEx` call:

```
procedure TMainForm.GenerateClick(Sender: TObject);
Var
  Regex : TRegexpr;
begin
  Regex:=TRegexpr.Create;
  try
    Regex.Expression:='\{\{[^{}]*\}\}';
    MResult.Lines.Text:=
       Regex.ReplaceEx(MTemplate.lines.Text,
    ReplaceCallback);
  finally
    Regex.Free;
  end;
end;
```

As you can see, the code is almost disappointingly simple. A regexp instance is created, the regular expression to find template names is entered, and the `ReplaceEx` is called with the `MTemplate` text as input. The result of the function is assigned to the text in `MResult`.

The real magic happens in the `ReplaceCallback` callback:

```pascal
function TMainForm.ReplaceCallBack(Sender: TRegexpr): String;
Var
   aMatch,aName : String;
begin
   aMatch:=Sender.Match[0];
   aName:=Copy(aMatch,3,Length(aMatch)-4);
   Result:=vleKeys.Strings.Values[aName];
end;
```

Again, the function is simple. It could be implemented as a one-liner, but for clarity, it is split out in 3 lines: First the regular expression match is taken, then the curly braces are stripped off to get the variable name, and the last line uses the variable name to look up the replacement value in the value list editor. That's it. It is so simple to create a simple templating engine. The result can be seen in figure 7 on page 8/8 of the article, page 48 of the issue.

5 CONCLUSION

Being able to use regular expressions can make life a lot easier, both when coding in the IDE as in an actual program: The examples presented here hopefully demonstrated that using simple regular expression is really not difficult and that they can easily be implemented in a Delphi or Lazarus program.

Michael van Canneyt has created 2 projects : one for Delphi and one for Lazarus. They are of course downloadadble from your login: `https://www.blaisepascalmagazine.eu/your-downloads/`
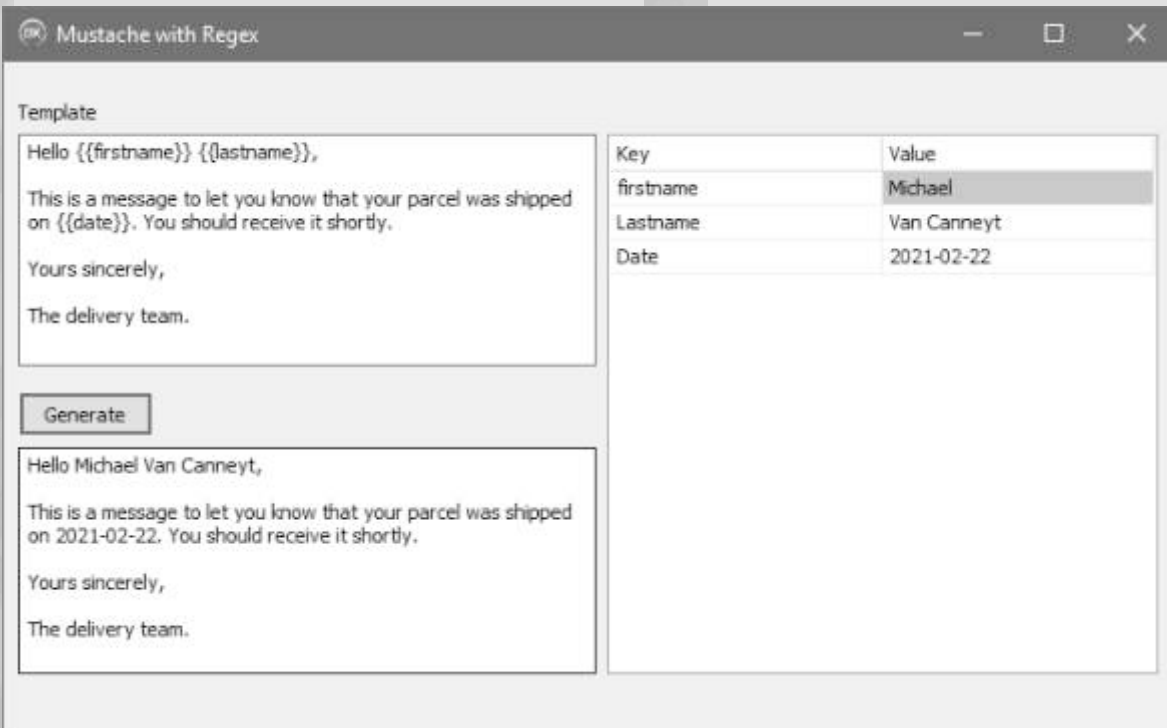
Figure 7: A simple template engine in action
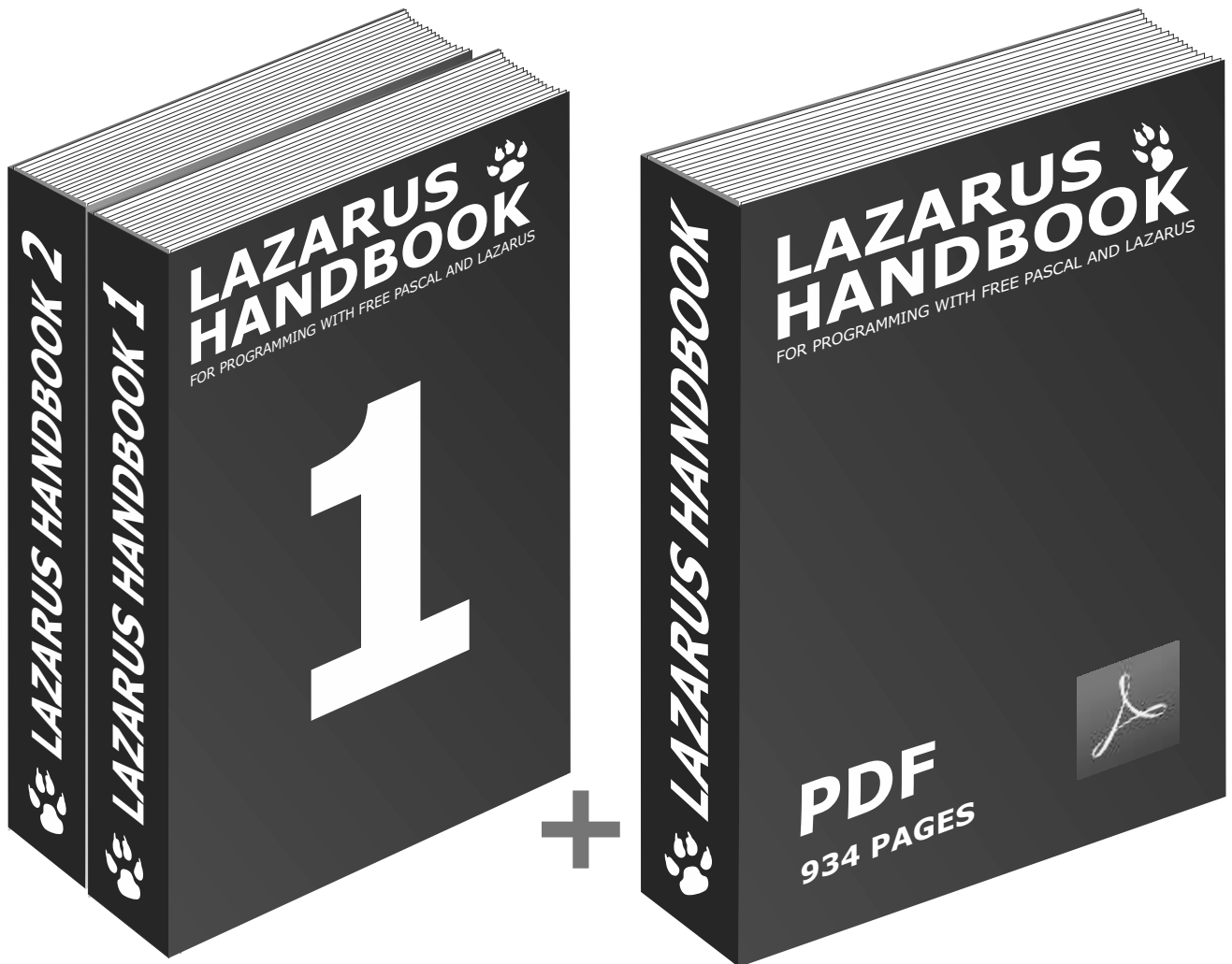
Galactic Center Sonification

(Credit: NASA/CXC/SAO/K.Arcand, SYSTEM Sounds (M. Russo, A. Santaguida))

`https://chandra.si.edu/photo/2020/sonify/`
`sonify_galactic_all.mp4`

INTRODUCTION:

Sonification is the process that translates data into sound, and a new project brings the center of the Milky Way to listeners for the first time.
The translation begins on the left side of the image and moves to the right, with the sounds representing the position and brightness of the sources.
The light of objects located towards the top of the image are heard as higher pitches while the intensity of the light controls the volume.

Stars and compact sources are converted to individual notes while extended clouds of gas and dust produce an evolving drone.
The crescendo happens when we reach the bright region to the lower right of the image. This is where the 4-million-solar-mass supermassive black hole at the center of the Galaxy, known as Sagittarius A* (A-star), resides, and where the clouds of gas and dust are the brightest.

Users can listen to data from this region, roughly 400 light years across, either as "solos" from NASA's Chandra X-ray Observatory, Hubble Space Telescope, and Spitzer Space Telescope, or together as an ensemble in which each telescope plays a different instrument.

Each image reveals different phenomena happening in this region about 26,000 light years from Earth.

The Hubble image outlines energetic regions where stars are being born, while Spitzer's infrared data show glowing clouds of dust containing complex structures.
X-rays from Chandra reveal gas heated to millions of degrees from stellar explosions and outflows from Sagittarius A.
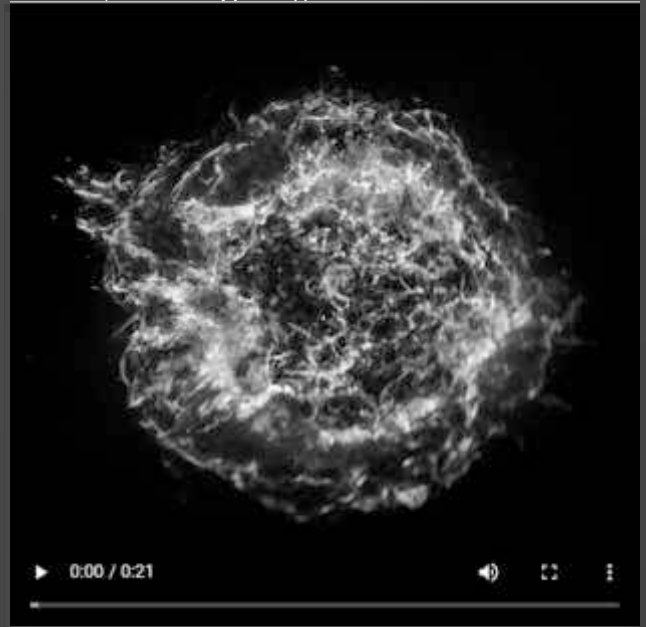
`https://chandra.si.edu/photo/2020/sonify/`
`sonify_casa_all_at_once.mp4`

`https://chandra.si.edu/photo/2020/sonify/`
`sonify_m16_composite.mp4`

For more detailed information:
`https://chandra.si.edu/photo/2020/`
`sonify/animations.html#M16`

starter   expert

In these code snippets I want to make interesting code available for everyone, and if possible do that for Lazarus and Delphi. One of the ways to find nice pieces of code is to search in the example list for projects that comes with Lazarus. Or create things myself so everybody can use them.

Since I have a vast imagination I can think of thousands, but they need to have a purpose: one must learn something out of it that is not easy to gain and can be an example in other situations…

In this case it was even more adventurous. I found three things that are interesting to understand: **the first**

Once I tested the Easter-date program I found that the program under Lazarus had a command prompt running during the final starting of the program. Rather annoying, I want to get rid of that, and than asked Mattias Gärtner - the Lazarus Guru –and he immediately had an answer to it:

there was something wrong with the project settings:
go to project settings and from there to
Compiler options → Config and Target → Target-specific options: (*see Figure 1*)
check the box **Win32 gui application(-WG)**
Win32 should be understood as Win32 and 64.
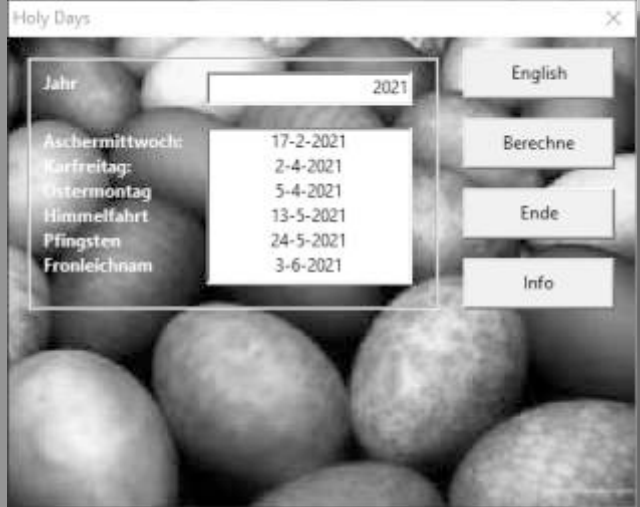
Figure 2: The Easter-date application
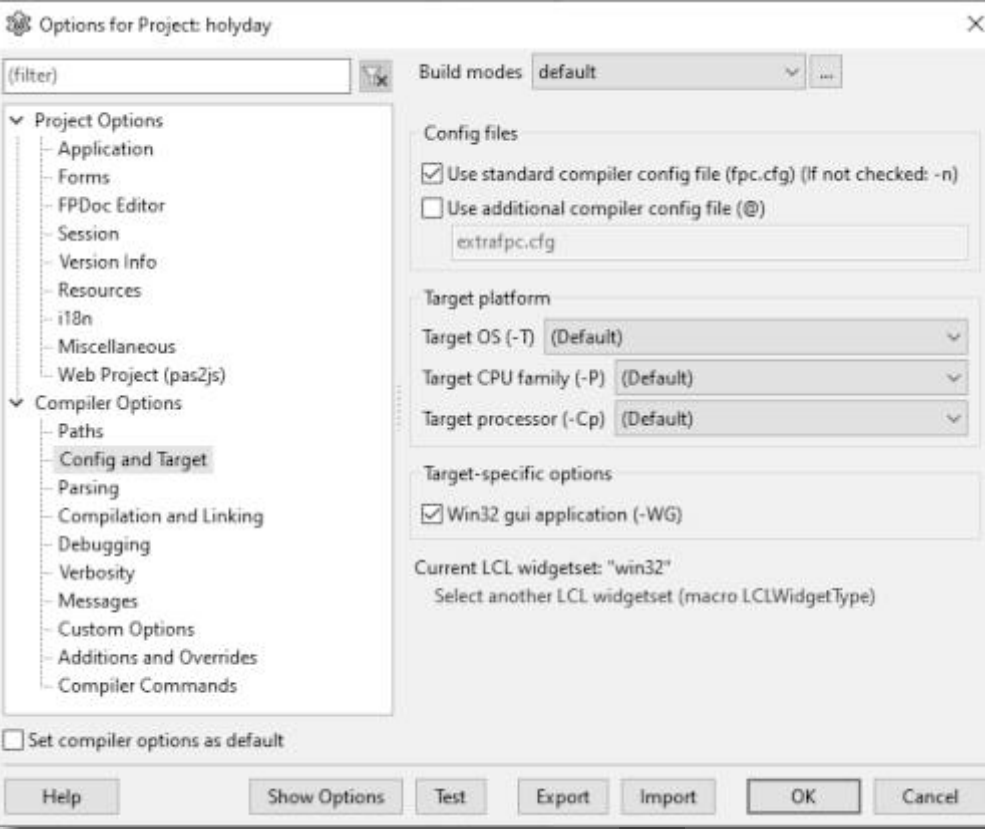
Then there is a little extra to be looked after:
`{$apptype console}`
either delete this or comment it .

Figure 1: Win32 Check

```
FUNCTION TForm1.CalcEasterday(aYear: WORD): TDateTime;
VAR
 A, B, C, D, E, F, G, H, I, J, K, L, M, N: INTEGER;
 vDay, vMonth: WORD;
BEGIN
 A := aYear MOD 19;
 B := aYear DIV 100;
 C := aYear MOD 100;

 D := B DIV 4;
 E := B MOD 4;

 F := (B + 8) DIV 25;
 G := (B - F + 1) DIV 3;
 H := (19 * A + B - D - G + 15) MOD 30;
 I := C DIV 4;
 J := C MOD 4;
 K := (32 + 2 * E + 2 * I - H - J) MOD 7;
 L := (A + 11 * H + 22 * K) DIV 451;
 M := (H + K - 7 * L + 114) DIV 31;
 N := (H + K - 7 * L + 114) MOD 31;

 vDay := N + 1;
 IF M = 3 THEN vMonth := 3 ELSE vMonth := 4;
 Result := EncodeDate(aYear, vMonth, vDay);
END;
```


Figure 3: Google Eggs

Google had a nice easter logo which I used here. And now the funny part comes: the project was developed in Delphi and then converted to Lazarus. But I did not have the original file, so I had to recreate it, and if possible make it a even „better".

The **second** item is that I wanted to know how the Easter code algorithm was composed, you best take a look your self:
`Tform.CalcEasterday`
In this app the Easter related Holidays are calculated. For the calculation there is a Function created which gives the result and a procedure that sets the final dates
`Procedure T.FormButton1Click`.

```
procedure TForm1.Listbox1DrawItem(AControl: TWinControl;
    Index: Integer;  ARect: TRect; AState: TOwnerDrawState);
var
 ts: TTextStyle;
begin
 Listbox1.Canvas.FillRect(ARect);
 ts := Listbox1.Canvas.TextStyle;
 ts.Alignment := taCenter;
 Listbox1.Canvas.TextRect(ARect, ARect.Left+2, ARect.Top,
 Listbox1.Items[Index], ts);
end;
```

In the about box you will find the name of the provider. I am not sure, they were the designer of this peace of code. If you are the designer, please make contact!

```
PROCEDURE TForm1.Button1CLICK(Sender: TObject);
VAR
 Easter: TDateTime;
 aYear : WORD;
BEGIN
 ListBox1.Clear;
 TRY
  aYear := StrToInt(Edit1.Text);
 EXCEPT
  if LanguageButton.Caption = 'English' then
   ShowMessage('Fehlerhafte Eingabe des Jahrs!')
  else
   ShowMessage('Incorrect input of the year!');
  Exit;
 END;
 Easter := CalcEasterday(aYear);
 ListBox1.Items.Add(DateToStr(Easter - 46));
 ListBox1.Items.Add(DateToStr(Easter - 2));
 ListBox1.Items.Add(DateToStr(Easter + 1));
 ListBox1.Items.Add(DateToStr(Easter + 39));
 ListBox1.Items.Add(DateToStr(Easter + 50));
 ListBox1.Items.Add(DateToStr(Easter + 60));
END;
```

Figure 4: The Delphi Version

Figure 6: The Google eggs are available for Delphi as wel.

Because I want to make these little projects available, I had to re-design it for Delphi, even though the original design was already done in Delphi. But since I discovered this project in the Lazarus Example list and wanted to review it, I not only had to redesign it, but also needed to create some code that was not available for Delphi which Lazarus does have:

TTextstyle which means you can centre the text in the listbox, very nice! **The third item:** But here is the code to do it in Delphi: not as easy as in Lazarus, but it still can be done…

Figure 6: The listbox text is now centered

```
procedure TForm1.ListBox1DrawItem(Control: TWinControl; Index: Integer;
 Rect: TRect; State: TOwnerDrawState);
var
 C: TCanvas;
 s: string;
 w: integer;
begin
 C:=ListBox1.Canvas;
 C.FillRect(Rect);
 s:=ListBox1.Items[Index];
 w:=C.TextWidth(s);
 C.TextRect(Rect, Rect.Left+2+(Rect.Right-Rect.Left-w) div 2, Rect.Top,
Listbox1.Items[Index]);
end;
```

Christian Feast Days Dependent on Easter Sunday
The date of Easter Sunday is crucial for the sequence of the Christian liturgical year as many Christian feast days are celebrated at a fixed interval of days (or weeks) before or after Easter Sunday.

As an extra I set the Delphi version to English, so you'll have to translate by using the Language Button. Of course all the code is downloadable so you can have a closer look. Anyway: Happy Easter! Next year Easter will be be on my Birthdate (18 april 2022)

The table at
https://webspace.science.uu.nl/ ~gent0113/easter/easter_text2c
lists the most important days in the Christian liturgical year and their relation with the date of Easter Sunday. Nominally, the dates are given for the current (Gregorian) calendar year, but the dates can be adjusted for any other year in the Gregorian calendar

By Detlef Overbeek

starter   expert

In this example - which is somewhat divergent in Delphi to the original In Lazarus - I want to show some little pieces of code that can become handy: How to change your Form icon on the fly and the use of the Track-bar.

Actually you can read the commentary notes in the Code example and see what is going on. Using it is best done by testing…The code examples are of course available at your personal Downloadpage:
https://www.blaisepascalmagazine.eu/your-downloads/

BLAISE PASCAL MAGAZINE

HOME   ▾ YOUR SUBSCRIPTION DOWNLOADS   REGISTER   GET SUBSCRIPTIONS ▾

```delphi
procedure TFrmGUI.BtnFormIcoClick(Sender: TObject);
begin
 // Add the application icon under Windows
 FrmGUI.Icon.Assign(Image1.Picture.Graphic);
end;

procedure TFrmGUI.FormCreate(Sender: TObject);
begin
 // Fetch the Path where the app is hosted and
 // insert the variable with that for global use
 S:= (ExtractFilePath(Application.ExeName)) ;
end;

procedure TFrmGUI.BtnLoadWinClick(Sender: TObject);
begin
 Image1.Picture.LoadFromFile((S) + 'icons' + PathDelim + 'Delphi.ico');
 // Extracts the path from where the program starts +subdir +file
 Caption := 'Icon found at ' + S + 'icons' + PathDelim + 'Delphi.ico' ;

 UpdateTrackbar; // See the procedure
end;

procedure TFrmGUI.BtnOSxClick(Sender: TObject);
begin
 ShowMessage('Delphi can not show icons for other OSx')
 // A button taken over from the Lazarus version, where this can be done
end;

procedure TFrmGUI.BtnSearch_IconClick(Sender: TObject);
begin

 try
  If OpenDialog1.Execute
  Then Image1.Picture.LoadFromFile(OpenDialog1.FileName);
  Caption := 'Icon found at Path: ' + OpenDialog1.FileName;
  // Show in the caption area the path
 Except
  MessageDlg('Loading the icon failed', mtError, [mbOK], 0);
 End;

 UpdateTrackbar; // See the procedure
end;
```
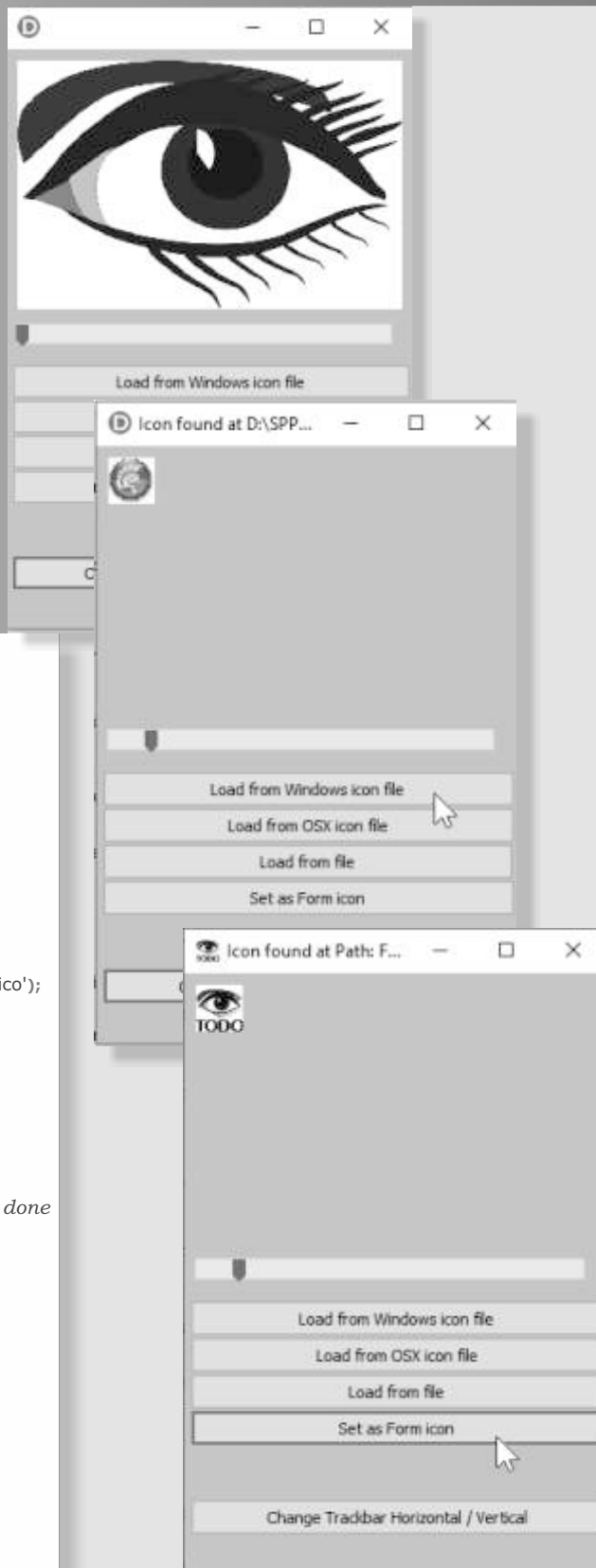
```pascal
procedure TFrmGUI.Btn_Hor_VertClick(Sender: TObject);
begin
 If Trackbar1.Orientation  = trHorizontal
 then
  begin // Set the orientation of the trackbar
   Trackbar1.Orientation := trVertical;

   // Change the width of the buttons
   BtnLoadWin.left       := + (Trackbar1.Width + 2);
   BtnLoadWin.width      :=  (BtnLoadWin.Width - 25);

   BtnOSx.left          := + (Trackbar1.Width + 2);
   BtnOSx.width         :=  (BtnOSx.Width - 25);

   BtnSearch_Icon.left   := + (Trackbar1.Width + 2);
   BtnSearch_Icon.width  :=  (BtnSearch_Icon.Width - 25);

   BtnFormIco.left       := + (Trackbar1.Width + 2);
   BtnFormIco.width      :=  (BtnFormIco.Width - 25);

   Btn_Hor_Vert.left    := + (Trackbar1.Width + 2);
   Btn_Hor_Vert.width    :=  (Btn_Hor_Vert.Width - 25);

  end
 else if Trackbar1.Orientation = trVertical
 then
  begin
  Trackbar1.Orientation   := trVertical;

   BtnLoadWin.left      := + 4;
   BtnLoadWin.width     := (BtnLoadWin.Width + 26);

   BtnOSx.left         := + 4;
   BtnOSx.width        := (BtnOSx.Width + 26);

   BtnSearch_Icon.left  := + 4;
   BtnSearch_Icon.Width := (BtnSearch_Icon.Width + 26);

   BtnFormIco.left      := + 4;
   BtnFormIco.width     := (BtnFormIco.width + 26);

   Btn_Hor_Vert.left    := + 4;
   Btn_Hor_Vert.width   := (Btn_Hor_Vert.Width + 26);

   Trackbar1.Orientation := trHorizontal ;
  end;

end;

procedure TFrmGUI.TrackBar1Change(Sender: TObject);
begin
 TrackBar1.Position := ((Image1.Picture.Icon.Width)div 2);
end;

procedure TFrmGUI.UpdateTrackBar;
begin
 if TrackBar1.Enabled then
 begin
  TrackBar1.Min := 0;
  TrackBar1.Max := 160;
  TrackBar1Change(TrackBar1);
 end;
```
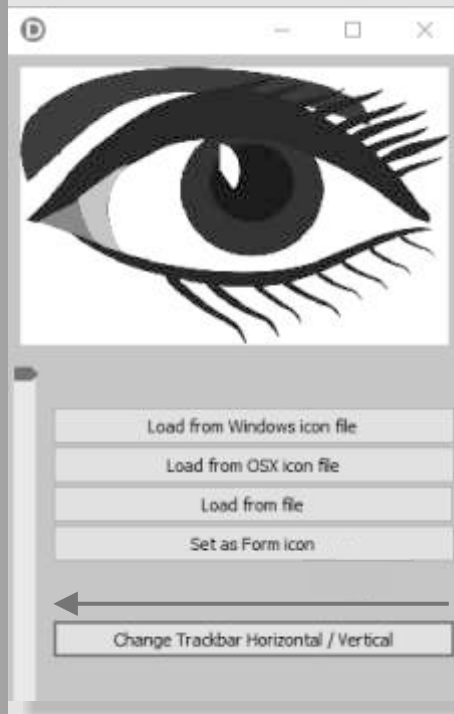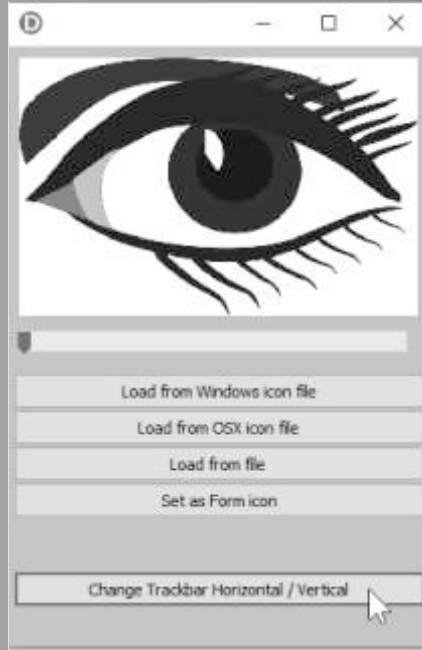

In the example of the code at the left is the code written for making sure you can see the Button width will be changed if you have the track bar horizontally activated or vertically. The option Load from OSx is not available because Delphi does not run on other OSx

starter   expert

This is the Lazarus version of the same project. Here you can use Load from OSX and it works. I have tried to find the immediate change for the Application Icon but that doesn't work. Would have been a nice feature… Or Not?

You can load a png file or what ever bitmaps. The result will however be always an icon. Lazarus has because of the Multiple OSs an extension that can contain multiple icons: `.icns`

Here is the loaded icon from OSX

```pascal
procedure TForm1.FormCreate(Sender: TObject);
begin
 S:= (ExtractFilePath(ParamStrUTF8(0))) ;
end;

procedure TForm1.BtnLoadWinClick(Sender: TObject);
begin
 Image1.Picture.LoadFromFile(ExtractFilePath
    (ParamStrUTF8(0)) + 'icons' + PathDelim + 'lazarus.ico');
 // extracts the path where the progarma starts +subdir
 //+ file
 Caption := 'Icon found at '
    + S + 'icons' + PathDelim + 'lazarus.ico' ;

 UpdateTrackbar;
end;

procedure TForm1.BtnOSxClick(Sender: TObject);
begin
 Image1.Picture.LoadFromFile(ExtractFilePath
    (ParamStrUTF8(0)) + 'icons' + PathDelim +
'lazarus.icns');
 Caption := 'Icon found at '
    + S + 'icons' + PathDelim + 'lazarus.icns' ;

 UpdateTrackbar;
end;

procedure TForm1.BtnFormIcoClick(Sender: TObject);
begin
 Icon.Assign(Image1.Picture.Graphic);
end;

procedure TForm1.BtnSearch_IconClick
        (Sender: TObject);
begin
 Try
  If OpenDialog1.Execute
  Then Image1.Picture.LoadFromFile
     (OpenDialog1.FileName);
  Caption := 'Icon found at ' + ' Path: '
     + OpenDialog1.FileName;
 Except
   MessageDlg('Loading the icon failed', mtError, [mbOK], 0);
 End;
 UpdateTrackbar;
end;

procedure TForm1.TrackBar1Change(Sender: TObject);
begin
 if TCustomIcon(Image1.Picture.Graphic).Count >
    TrackBar1.Position
 then TCustomIcon(Image1.Picture.Graphic).Current :=
    TrackBar1.Position;
end;

procedure TForm1.UpdateTrackBar;
begin
 TrackBar1.Enabled :=
   Image1.Picture.Graphic is TCustomIcon;
 if TrackBar1.Enabled then
 begin
  TrackBar1.Min := 0;
  TrackBar1.Max :=
    TCustomIcon(Image1.Picture.Graphic).Count - 1;
  TrackBar1.Position :=
    TCustomIcon(Image1.Picture.Graphic).Current;
  TrackBar1Change(TrackBar1);
 end;
end;
```

Embarcadero is pleased to announce the release of Delphi, C++Builder and RAD Studio 10.4.2. With new features and much improved quality, the new release builds on the work done in 10.4 Sydney and the 10.4.1 quality release.

RAD Studio 10.4.2 continues expanding some of the key cornerstones of the product, from Windows to multi-device support, from IDE modernization to libraries quality and compiler performance.

In the following blog post, we want to highlight some of the main new features and enhancements in 10.4.2, including the best-in-class windows application development capabilities, the new developer productivity and user experience features, the expanded FireMonkey platforms support, the new Delphi and C++ features, and the improvements to quality.

Read More

On February 14th 1995, Borland introduced a new tool for developers, one that sparked a lot of enthusiasm and over 26 years has been used to build applications used by billions of people (think about the good old Skype) and it's still being used today for building apps for many incredibly different tasks.

Embarcadero is running a Showcase project for that. Marco Cantu's blog post does not cover the launch day (you can refer to Marco's old birthdays site for that) or the Showcase, but rather looks at how things have changed over the years and how a number of important milestones have preserved their original value.

barnsten

Online Delphi FMX Power Trainingen

Leer ontwikkelen voor 4 platformen tegelijkertijd - Windows, macOS, iOS en Android

4 en 5 maart van 09.00-17.00 uur
We starten weer met de trainingen!
Online deze keer, maar met evenveel enthousiasme en passie én met een kleine groep deelnemers.
Vooraf krijgen de deelnemers het Nederlandstalig lesmateriaal toegestuurd.
Om het nog leuker te maken ontvangt u ook box met versnaperingen om tijdens de training van te genieten.

Overzicht trainingen

Delphi FMX Power Training op 4 maart
Laat u verrassen over de mogelijkheden van het FMX Framework voor cross-platform ontwikkeling vanuit één enkele broncode en ga er na deze training zelf mee aan de slag.
Delphi FMX Android Power Training op 5 maart
Bent u al bekend met FMX en klaar voor het ontwikkelen van apps? Deze training gaat specifiek in op het ontwikkelen van mobile apps met Delphi. Na deze training kunt u direct beginnen. Ook geschikt als vervolg van de FMX Power training op 4 maart.
Interesse in beide trainingen? Dan krijgt u een combinatie korting.

Deelnemers ontvangen na afloop een certificaat van deelname.

Danny Wind Delphi MVP, trainer en ontwikkelaar

INTRODUCTION

This series of articles is about writing your own web services server and client in Delphi. The approach of all articles is pragmatic. This first article introduces some of the concepts you need to know and shows you how to create and consume your own web service in Delphi.

What is a web service? A web service sends and receives data over the world wide web. Web services mostly communicate over the internet through the HTTP protocol and send and receive data in one of the web formats, such as JSON, XML or HTML.

Why would you want to build a web service? Web services are used in almost every app, website and desktop application to get local and remote data. Web services are also used for interoperability and import and export. For instance accounting software usually has an interface to its locally or remotely stored data using a local or remote web service. Web services are also easily scalable. Start with a simple locally run web service on a laptop and then scale up into a server park or into the cloud.

In a sense even a website is a web service. When you open an URL (`https://www.blaisepascalmagazine.eu/`) in your web browser the browser sends a HTTP GET command to the website URL and in response receives data in the form of a HTML page.

However, when software developers mention web services, they mostly mean a web service that uses REST (REpresentational State Transfer) over HTTP to send and receive JSON formatted data. For instance the DuckDuckGo API that resolves your question into a direct answer from WikiPedia, GitHub and more. Just try this link in your browser `https://api.duckduckgo.com/ ?q=Blaise&format=json&pretty=1`

When you access a web service you supply an URL combined with an identifier into an URI for a specific resource.

URL - Uniform Resource Locator

This is the human readable address that a resource (a web service) can be found at. An example would be https://duckduckgo.com/. It's translated to a physical IP address through DNS. This way the resource can be located over a TCP/IP network. In analogies an URL would be the home address for the house where your resource lives.

URI - Uniform Resource Identifier

There is also a thing called URI. This identifies a specific resource. If you just go with the idea that this adds a specific resource identifier to retrieve from the URL location, you're not far off. An example of an URI is https://duckduckgo.com/index.html. In analogies the URI would be a specific bookcase inside the house.

URL vs URI

It's fun to debate what is or isn't an URL, or an URI, because the definition in the RFC documents leaves some of it open to different interpretations. So you may get into a discussion at the coffee machine on URL vs URI, after which you can relax with your cup of coffee, as it really doesn't matter. It's just the location of your web service.

The protocol used is HTTP, denoted by the prefix http:. Even if the URL has HTTPS in its prefix this is still talking HTTP, it's just encrypted.

HTTP - Hyper Text Transfer Protocol

HTTP is the protocol used to communicate over TCP/IP with your web service. In analogies HTTP is a very limited language used to exchange data.

HTTP has nine commands that are used to request and receive data. Each of these commands has a specific purpose. For web services there are four commands you should start with.

HTTP GET	Retrieves data from the resource
idempotent, cacheable	
usage in our web service	**SELECT** (get existing record, disallow caching so we get new data each time)
HTTP POST	Appends data to the existing resource
not idempotent, not cacheable/stale	
usage in our webservice	**UPDATE** existing (partial update of fields in a record, not updating the primary key)
HTTP PUT	Replace the existing resource or inserts data as a new resource
idempotent, not cacheable/stale	
usage in our web service	**INSERT** new (or **REPLACE**) (insert new record with new primary key, or replace entire record)
HTTP DELETE	Deletes the resource
idempotent, not cacheable/stale	
usage in our web service	**DELETE** (delete existing record or return error if it doesn't exist)

In our article we will use a simplified mapping of HTTP commands to actions we want our web service to perform. For more complete mapping you could take a look at some of the open source REST web service frameworks available.

Some Open Source web service frameworks
MARS Curiosity Framework:
`https://github.com/andrea-magni/MARS`
mORMot ORM Framework
`https://github.com/synopse/mORMot`
WiRL RESTful library
https://github.com/delphi-blocks/WiRL

We will not be using these existing frameworks, instead we will be building a simple web service server and client from scratch.

STARTING THE BUILDING OF THE PROGRAM
on the next page we will create the web service client.
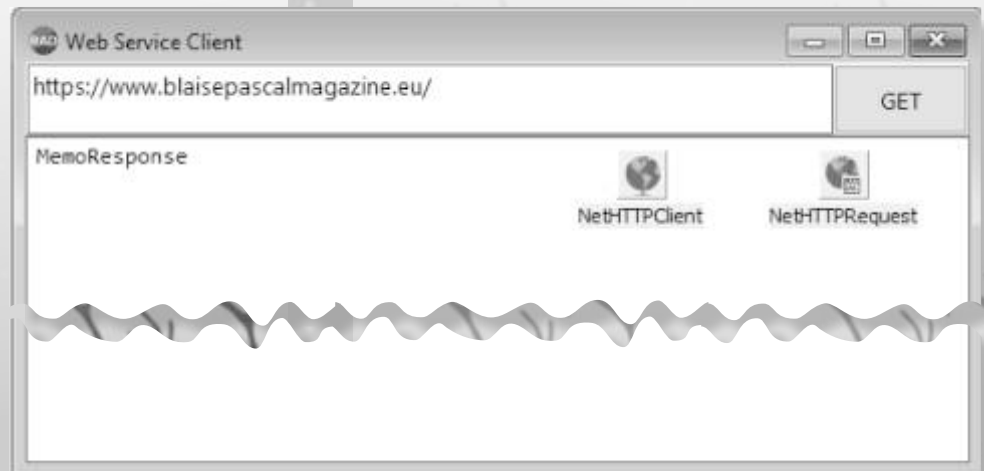
Figure 1: The Web Service Client to be build

Start with either a VCL or FireMonkey application
1. Add a Panel and align Top
2. Add a Button to this Panel and rename to ButtonGet, align Right
3. Add an Edit to this Panel and rename to EditURL, align Client
4. Add a Memo to the Form and align Client
5. Add a `NetHTTPClient` to the Form
6. Add a `NetHTTPRequest` to the Form and link the Client property to `NetHTTPClient`
7. Add an `OnClick` event-handler to the `ButtonGet`

```
procedure TFormMain.ButtonGETClick(Sender: TObject);
begin
 NetHTTPRequest.MethodString := 'GET';
 NetHTTPRequest.URL := EditURL.Text;
 NetHTTPRequest.Execute();
end;
```

8. The `HTTP GET` request is executed asynchronously, which means the response will appear at some time in the future. When the response arrives the `OnRequestCompleted` event-handler of the `NetHTTPRequest` will be called.
9. Add an `OnRequestCompleted` event-handler to the `NetHTTPRequest`

```
procedure TFormMain.NetHTTPRequestRequestCompleted(const Sender: TObject;
 const AResponse: IHTTPResponse);
begin
 MemoResponse.Text := AResponse.ContentAsString;
end;
```

10. Run and test using a website URL, for instance
    `https://www.blaisepascalmagazine.eu/`
11. The GET will return a HTML page
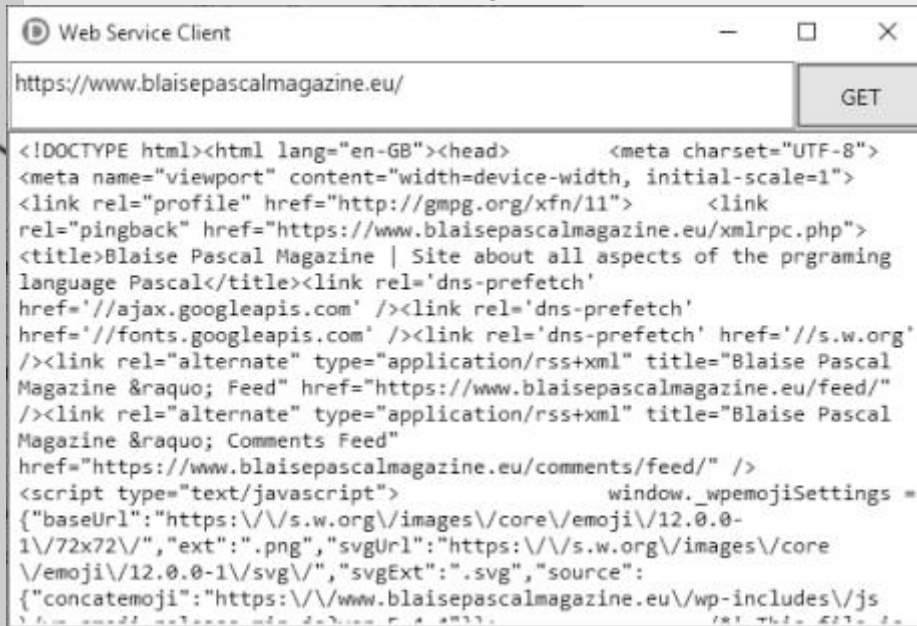
Figure 2: The result of the GET order event

12. When you use an URL to a REST endpoint from an existing Web Service, you get JSON data. Try this URLs in the Web Service Client
    https://api.discogs.com/artists/457265

13. The result is JSON data, recognizable due to the curly braces {}


```
{"name": "Wende Snijders", "id": 457265, "resource_url":
"https://api.discogs.com/artists/457265", "uri":
"https://www.discogs.com/artist/457265-Wende-Snijders", "releases_url":
"https://api.discogs.com/artists/457265/releases", "images": [{"type":
"primary", "uri": "", "resource_url": "", "uri150": "", "width": 160,
"height": 160}, {"type": "secondary", "uri": "", "resource_url": "",
"uri150": "", "width": 336, "height": 448}, {"type": "secondary", "uri":
"", "resource_url": "", "uri150": "", "width": 596, "height": 600},
{"type": "secondary", "uri": "", "resource_url": "", "uri150": "",
"width": 450, "height": 463}, {"type": "secondary", "uri": "",
"resource_url": "", "uri150": "", "width": 337, "height": 250}],
"realname": "Wende Snijders", "profile": "Talent has no boundaries for
Dutch singer-songwriter Wende, who has enjoyed one of the most remarkable
musical careers seen in the Benelux countries in recent years. As a fresh
graduate from the Amsterdam Theatre School, in 2004 Wende released her
debut album of brilliant interpretations of a repertoire of French
chansons, supported by the renowned Metropole Orchestra. It became a
staggering success, setting the foundations for her to become established
```

Figure 3: The result is JSON data

14. The next step is to create our own Web Service Server
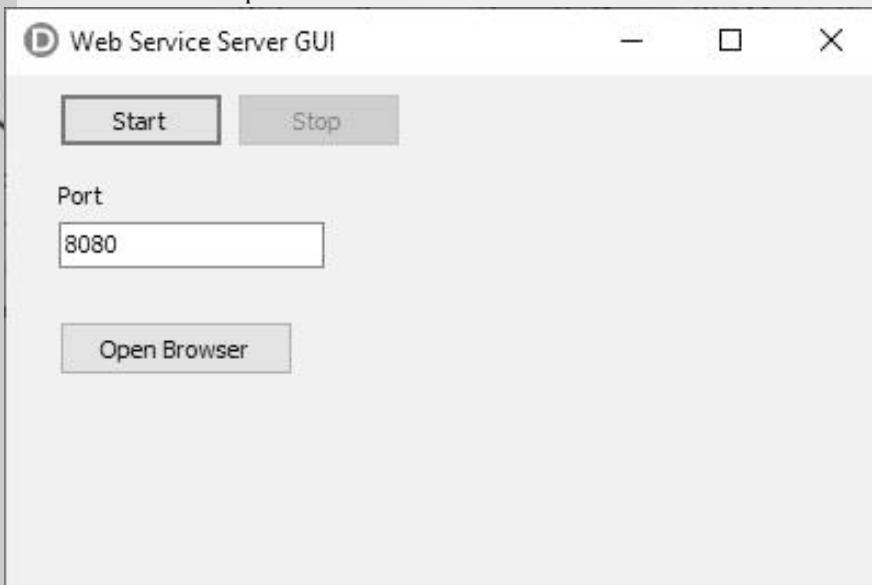
Figure 4: The Web Service Server Gui

15. We start with the New Item wizard for Web | Web Server Application
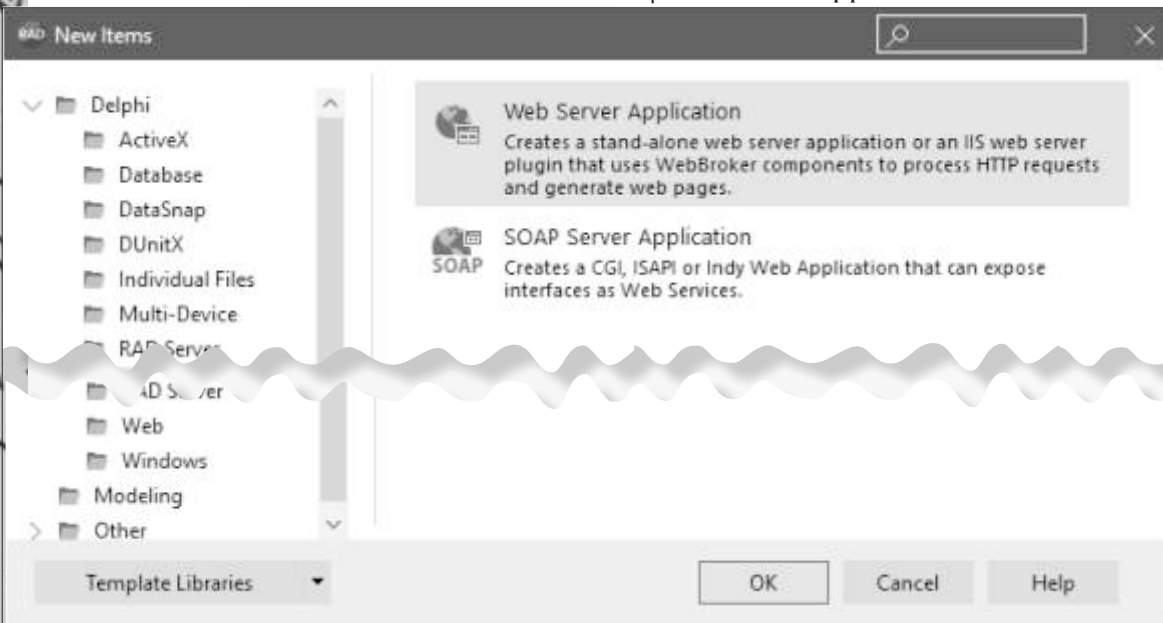
Figure 5: starting with the Web Server Appication: New ➔Other➔Web

16. This will allow you to create a web server that runs standalone or as a library in **Apache (Linux)** or **IIS (Windows)**. It will process HTTP requests and you can write the code on how it should respond, with plain text, a JSON string, HTML page or even an image or a file.
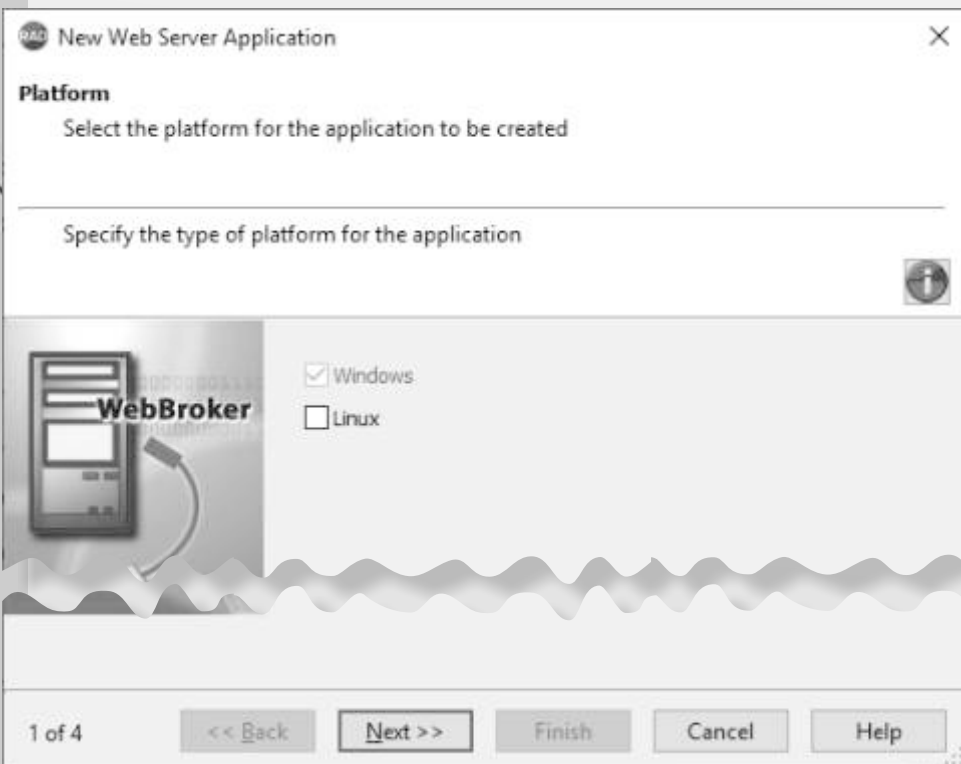
Figure 6: Standard for Windows

17. For now we go with Windows. We can add Linux support later on.
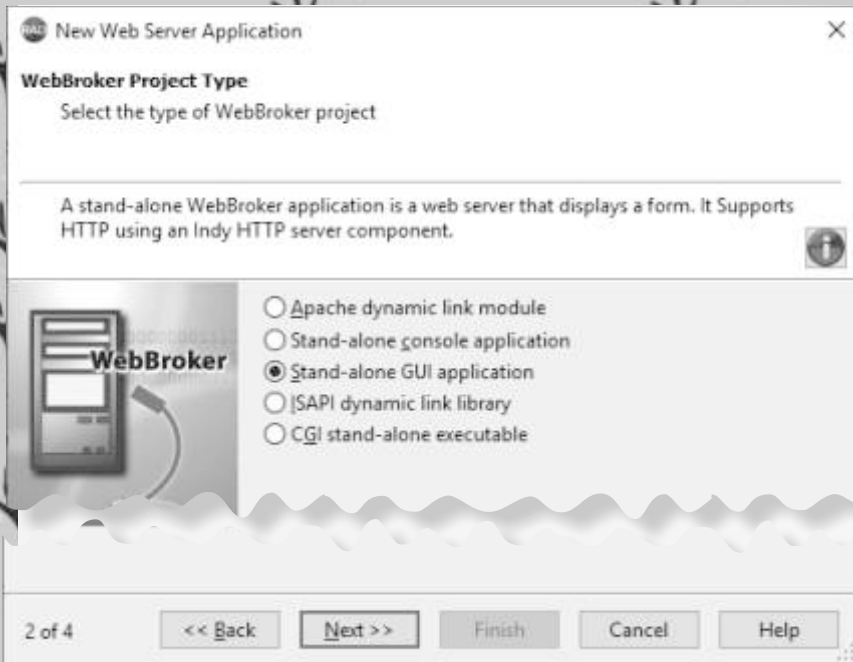
Figure 7: Select the type of WebBroker

18. For now we create a stand-alone GUI application. If we run this wizard again later on we can combine these options into one Project Group and share the base code files between these options.
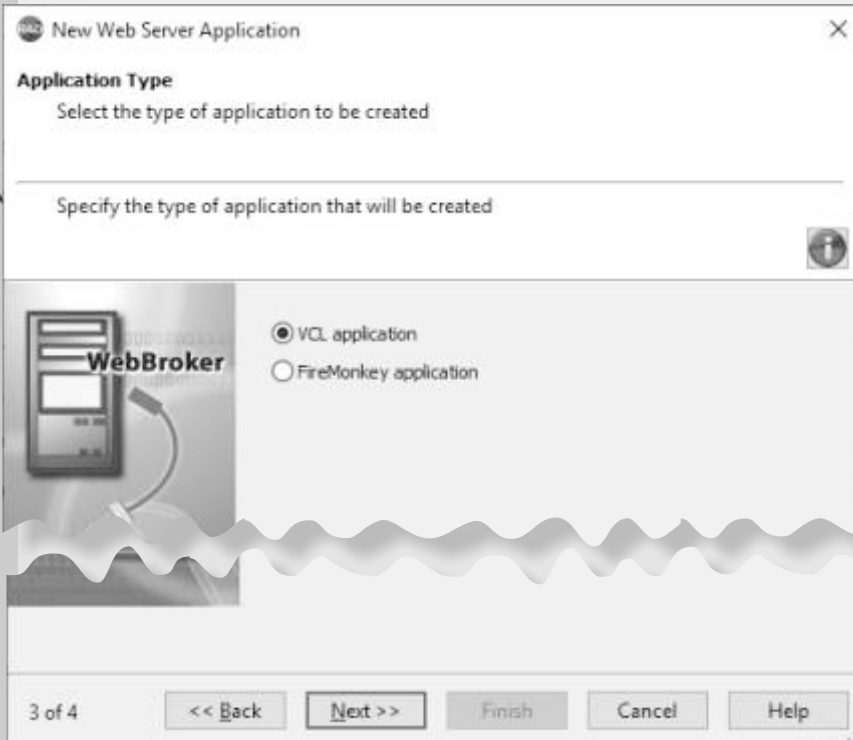
Figure 8: VCL

19. VCL is OK
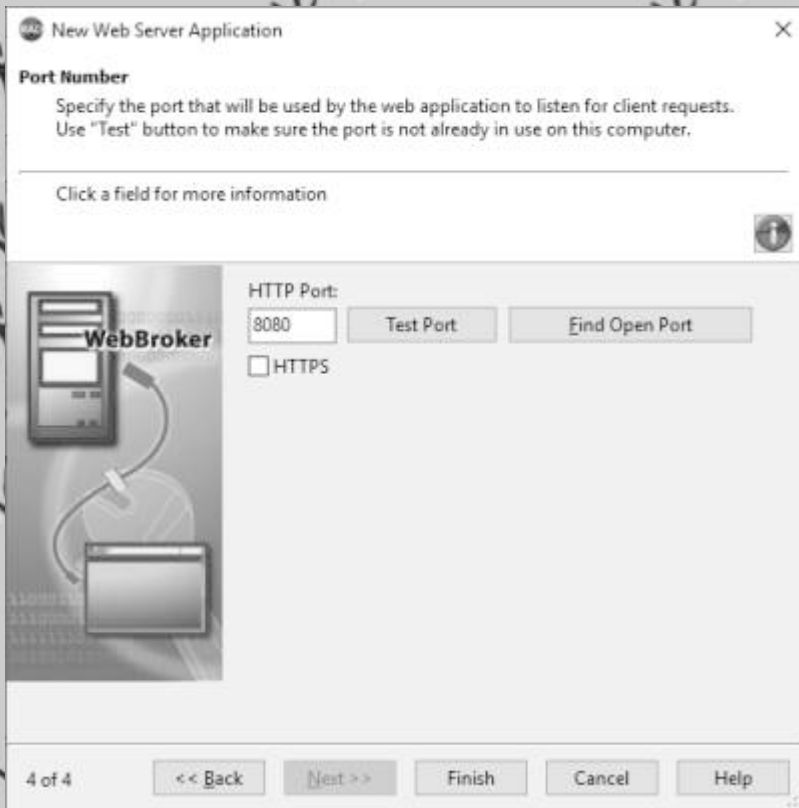
Figure 9: The port number

20. Test if the default port is not already taken on your machine.
     If it is you can Find Open Port or just try another port, such as 8088 or 8888.
21. Save the files into a separate directory and save the Project as WebServiceServerGUI
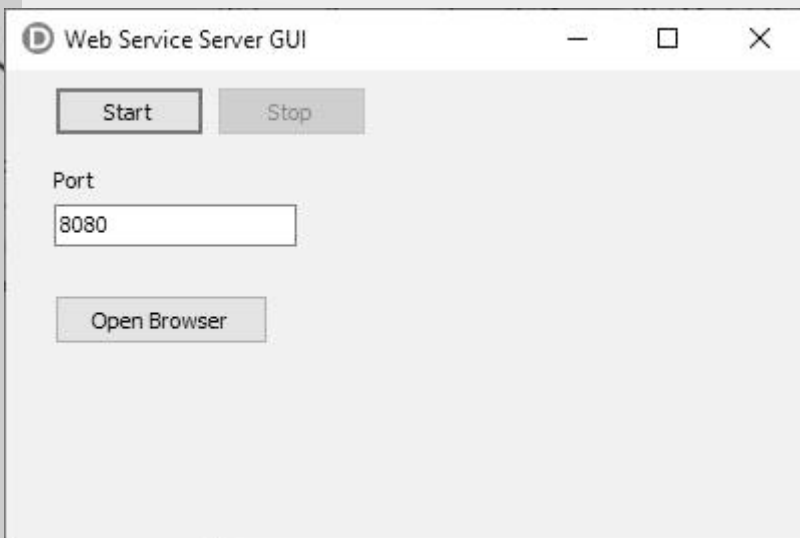
Figure 10: Run the Web Service Server GUI

22. Run the Web Service Server GUI
23. Click on Start, this should open the Windows Firewall configuration, depending on your local network configuration (Private or Public) you can choose to open the port for private networks only or for public networks (if you have your computer configured as such). Please note that when opening it up for public networks would mean the port is also open when you visit an airport.
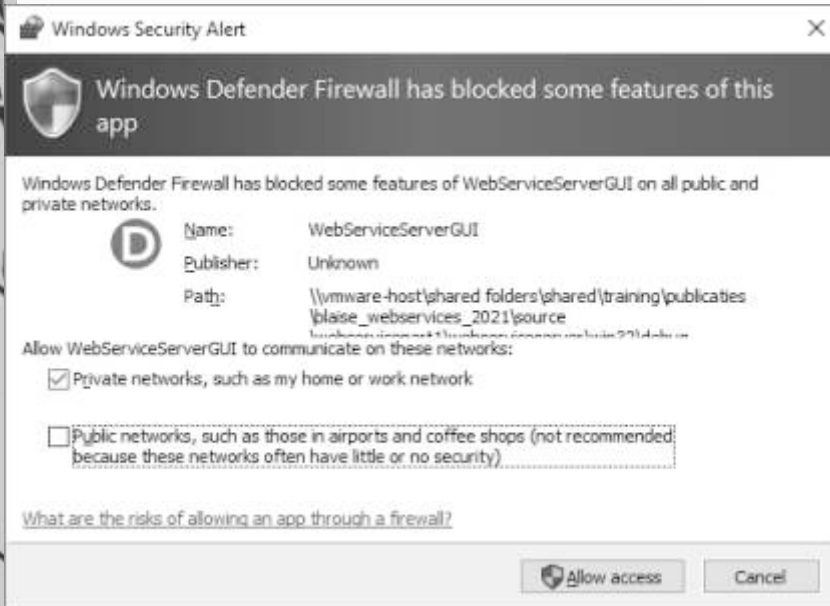
Figure 11: The Windows Firewall config

24. Make note of the URL that is then opened in the web browser
    `http://localhost:8080/`
25. Run the Web Service Client and open this URL.
26. The :8080 in this URL is the internet port. An internet port is like a door, each door has its own number. If the port is opened in the firewall, traffic is allowed through.
27. The localhost in this URL translates to a loopback to this machine. It's translated to a local IP address (127.0.0.1) that points to the machine you are running on.
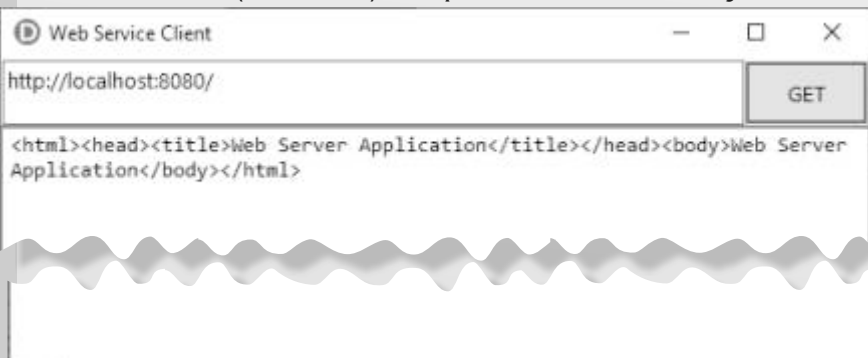
Figure 12: Run the Web Service Client and open this URL.

28. This is the default handler for the Web Server Service
29. Let's add a new GET handler to the Web Server Service
30. Open the Web Server Service GUI project and open the WebModuleUnit
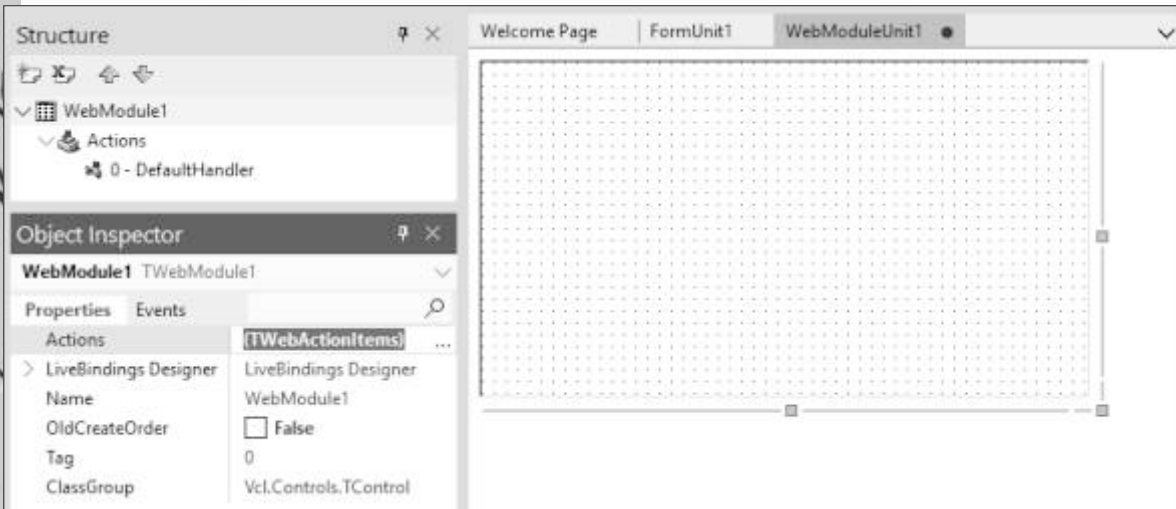
Figure 13: WebModuleUnit

31. HTTP requests to the Web Server are mapped to WebActionItems (Action Handlers).
    Add a new WebActionItem by clicking the three-dots in the property Actions
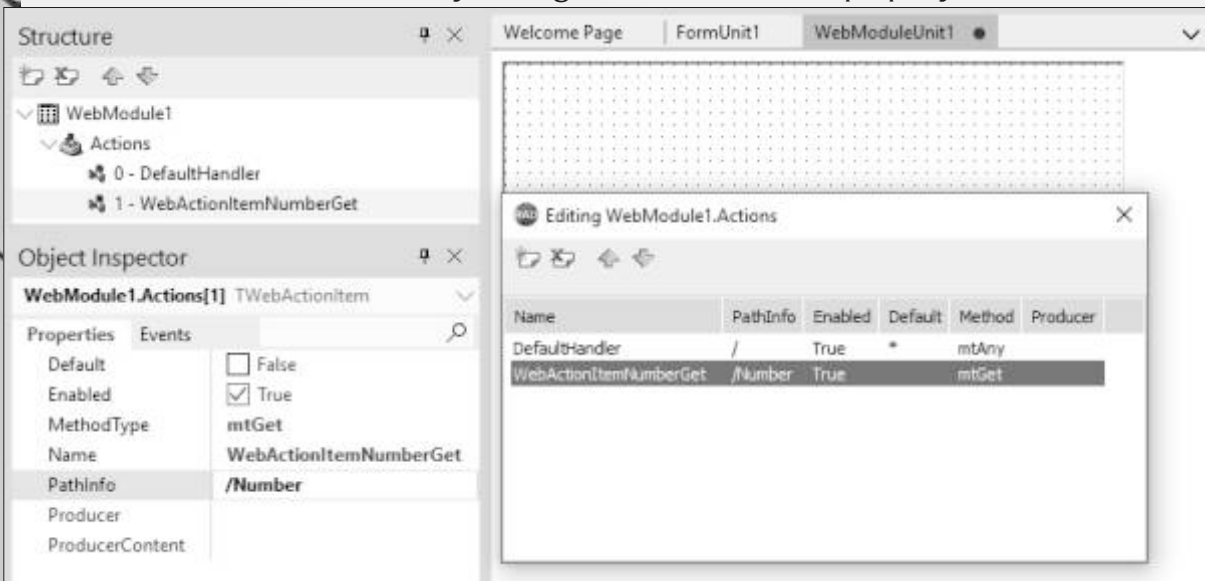
Figure 14:

32. Modify MethodType to mtGet and PathInfo to /Number
33. Add an OnAction event-handler to this WebActionItem and code a response

```
procedure TWebModule1.WebModule1WebActionItemNumberGetAction(Sender:
TObject;
 Request: TWebRequest; Response: TWebResponse; var Handled: Boolean);
begin
 Response.ContentType := 'application/json; charset=UTF-8';
 Response.Content := Random(100).ToString;
end;
```

34. Run the Web Service Server, and Start it with the Button
35. Now open the Number URL in the Web Service Client
    `http://localhost:8080/Number`
36. Request a couple of random numbers in the Web Service Client

Figure 15: Opening the Number Url

37. By the way, this is valid JSON, because a Number is a primitive
    datatype is where no encapsulation or serialization is needed.
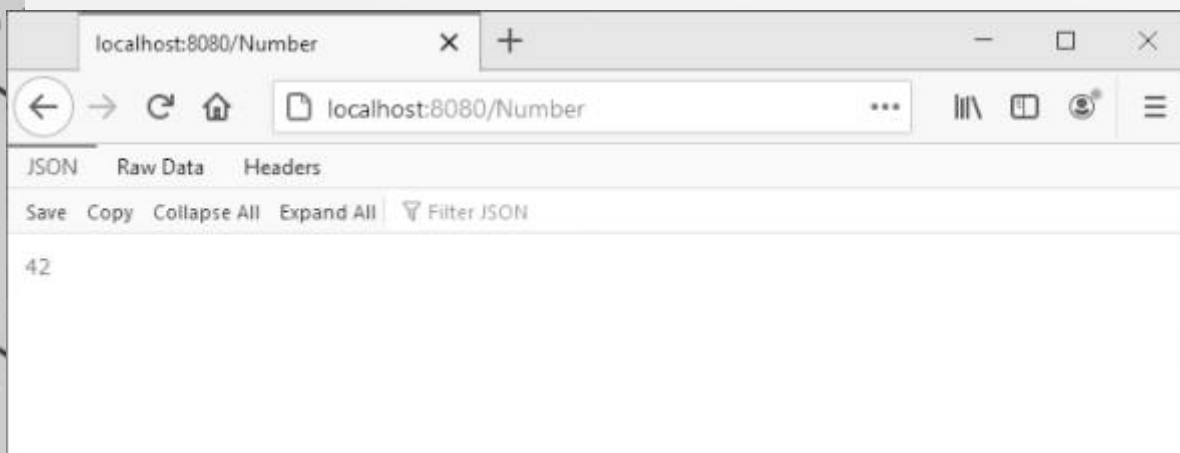
Figure 16:

38. See here also for a brief description of JSON structures
`https://www.json.org/json-nl.html`
You already have the basis for your own Web Service. You can use this code for example to use the
values from your own weather station via GET requests available and displayed in an Android App.

In the following articles, we will change data in the Web Service by adding a PUT, POST and DELETE.
We will also use more complex JSON structures.

You can download the source code for this article from your subscription webpage:
`https://www.blaisepascalmagazine.eu/your-downloads/`

By Mattias Gärtner   starter   expert

Figure 1: Logo   2.0.12

INTRODUCTION:
We have been developing the free version of
TMS WebCore and that worked fine for Windows
and for Linux. **macOS** was however a problem.
Not because we could not install it, but because
of the manifold popping up of warnings and
requests to accept to use your password.
Up to 8 times per install. That is ridiculous.
So we decided to find a way to get it done
without all that hassle. **Apple** has still its very
own ideas of protecting and User Interface.
I had promised that about three months a go.
It's a long way to Tipperary, an old song and
especially this project needed quite some time.
But we have solved it. Here is the story of how
to install it.

The version is downloadable for free from:
`http://downloads.blaisepascal.eu/`
`TMSWebcoreLazarusDemoFree1.5.4.zip`

By the way: now you also can download the
latest **stable update**(no new features only bugs
reduction) of **Lazarus** from:
`https://sourceforge.net/projects/lazarus/`
`files/Lazarus%20Windows%2064%20bits/`
`Lazarus%202.0.12/`
`lazarus-2.0.12-fpc-3.2.0-win64.exe/download`

or go to:
`https://www.lazarus-ide.org/`

(see figure 4 on the next page➔2/4 or 71)
Once you have done this we can start the
installation of the package. First of all: the
installation under Win10 responds with a
message of distrust. Don't worry, simply click on
the more info text. In the next screen you can
simply run it.

Figure 2/3: the warning that is of no use...

NOW BACK TO macOS.
Here you need to prepare for the usual steps:

1. Install your macOS version of Lazarus.
    We have tested this with the version
    2.0.11(the latest version was not available
    yet by writing this article…). The version is
    a so called trunk version, but works no
    other than 2.010

2. Download the version of TMS Webcore for
    free from our website or this address
    `http://downloads.blaisepascal.eu/`
    `TMSWebcoreLazarusDemoFree1.5.4.zip`
    (*See figure 6 at page 1/7 or issue page 71*)

3. You also need to collect your License
    number: it's for free available at:
    `https://www.tmssoftware.com/site/`
    `trialkey.asp`

4. Now unzip the download to a directory of
    your choice

5. Start Lazarus

6. Go to Package ➔ Open Package file (lpk)
    ➔ select in the directory of your choice
    `tmswebcorepkgliblaz.lpk`.
    That file will handle the other `.lpk` files
    as well. The package window opens.

Figure 4: the header of the lazarus pages
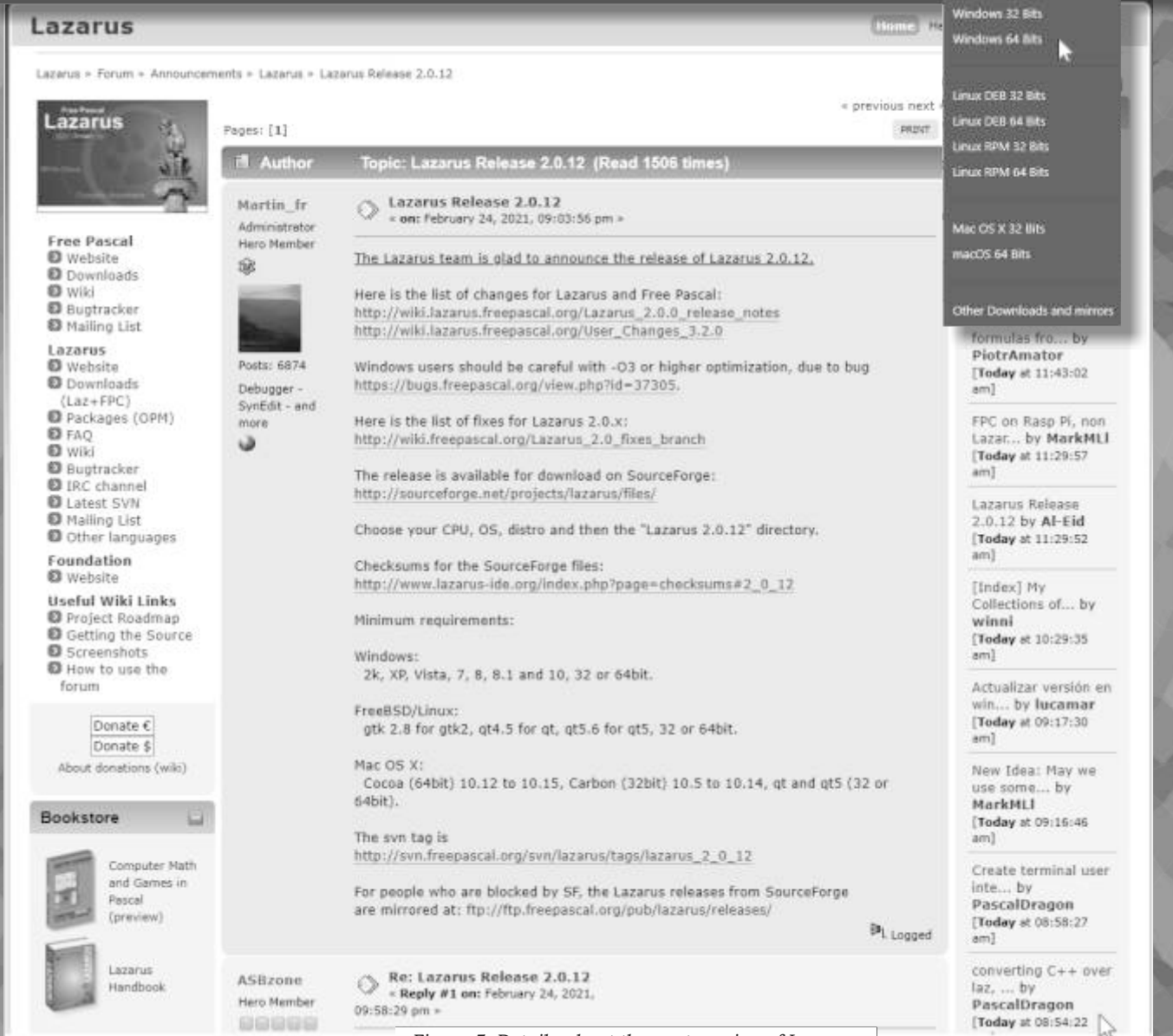
Figure 5: Details about the next version of Lazarus
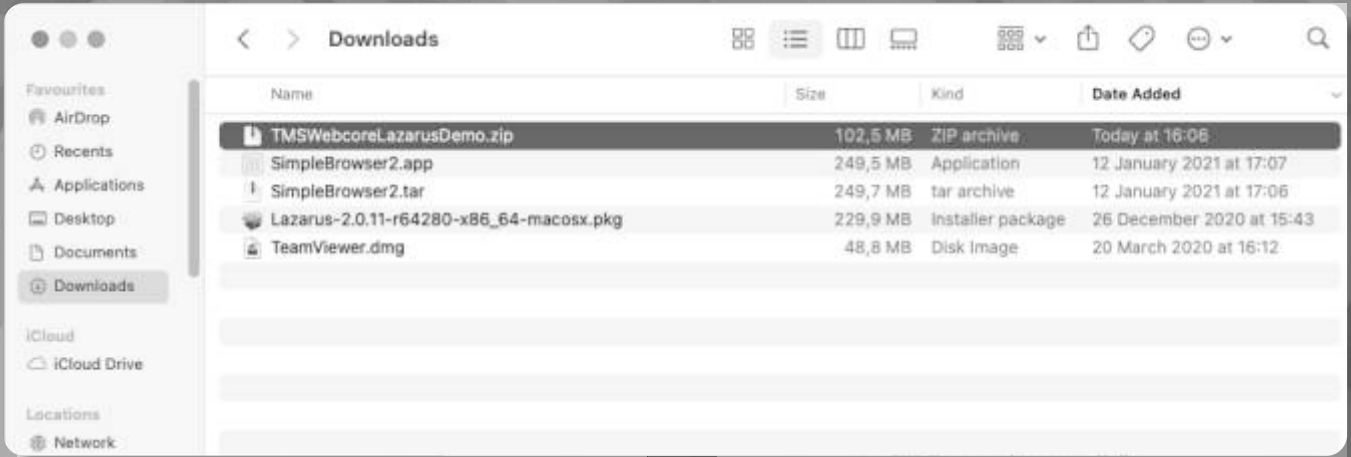
Figure 6: The zip file
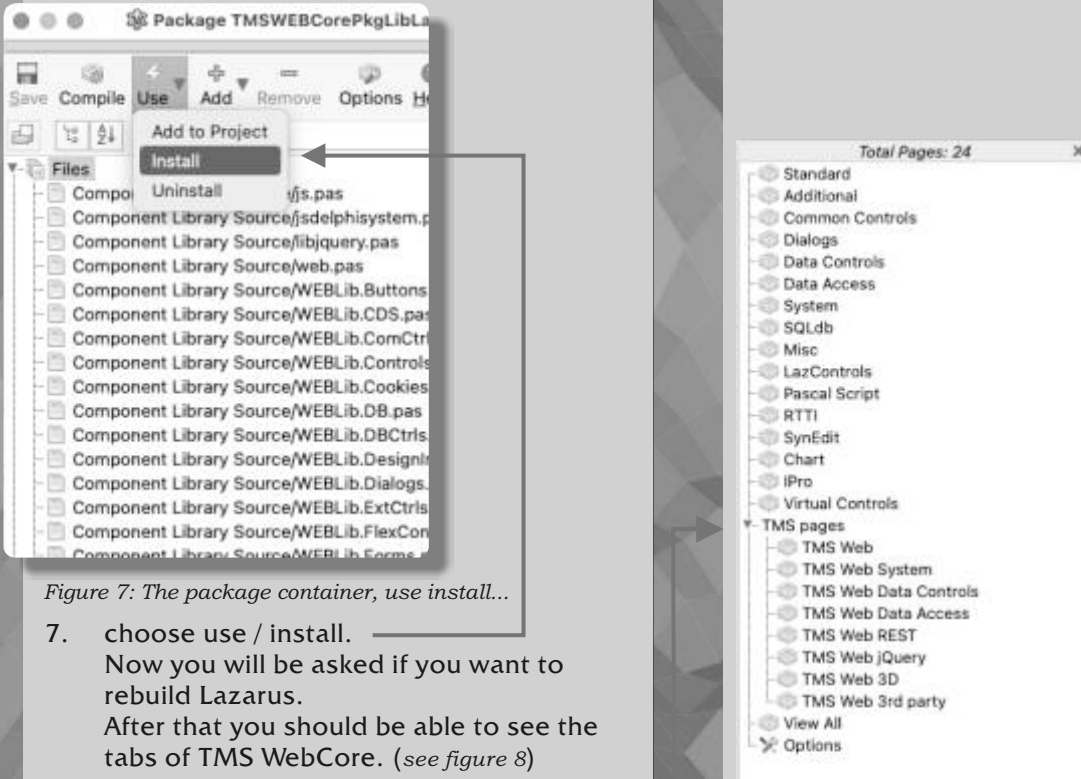
Figure 7: The package container, use install...

7. choose use / install.
    Now you will be asked if you want to
    rebuild Lazarus.
    After that you should be able to see the
    tabs of TMS WebCore. (*see figure 8*)
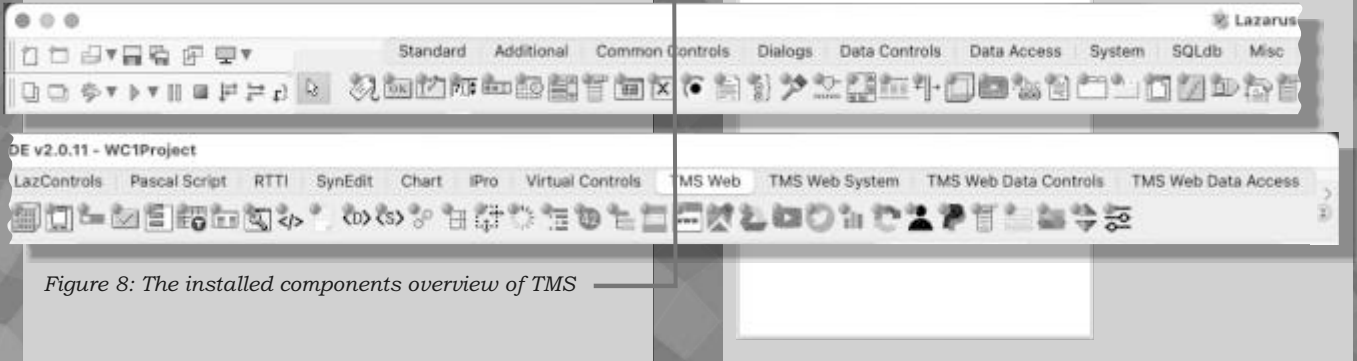
Figure 8: The installed components overview of TMS

Now build a small project and compile that. Immediately you'll be urged to enter your free code of TMS.

Try to run the project. If it works: that's all. Tested with macOS – Big Sur
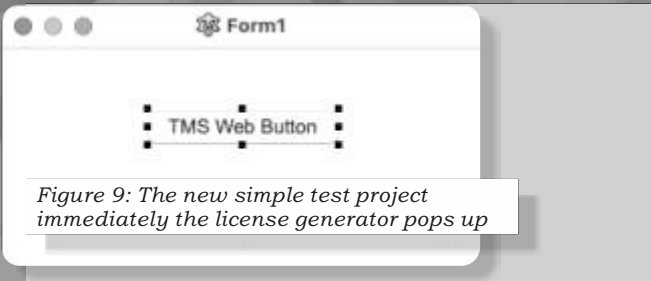
Figure 9: The new simple test project immediately the license generator pops up

TMSWebcompiler license not found

The TMSWebCompiler requires a license file "/Users/detlefoverbeek/Downloads/TMSWebcoreLazarusDemo/Bin/macOS/License.lic". This license is generated using your email address and TMS Webcore key.

[Ignore] [Ignore until IDE restarts] [Run license generator] [Cancel]

Figure 10: The License generator

Figure 11:
After that you need to save the license in that directory
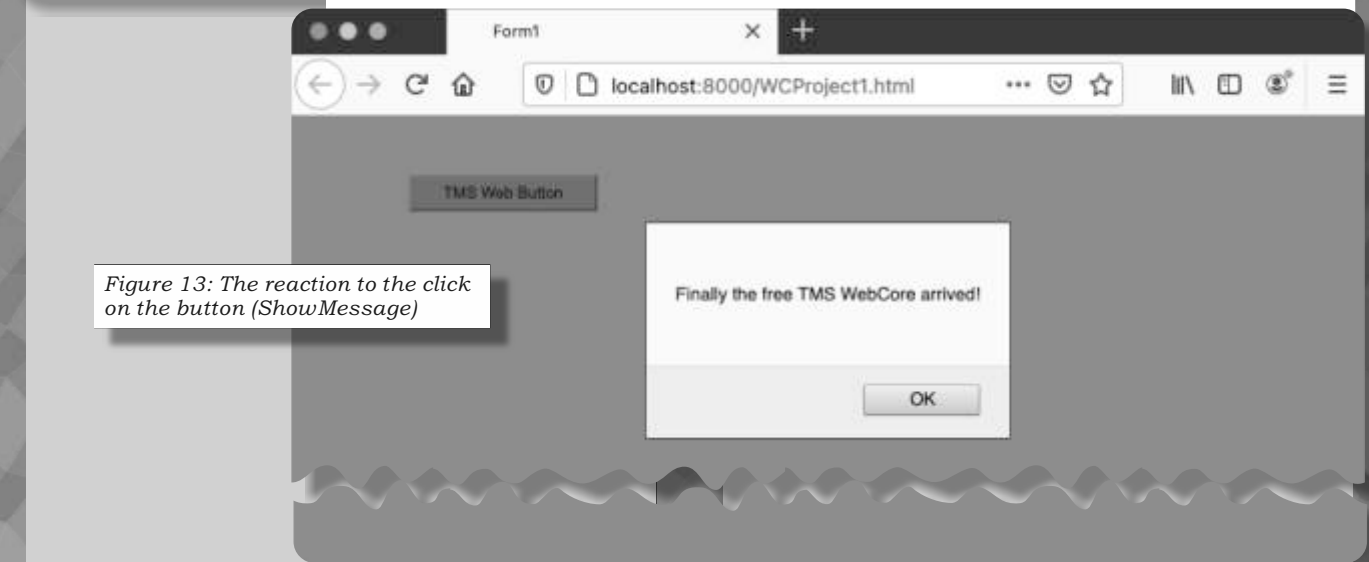
Figure 12: The running App
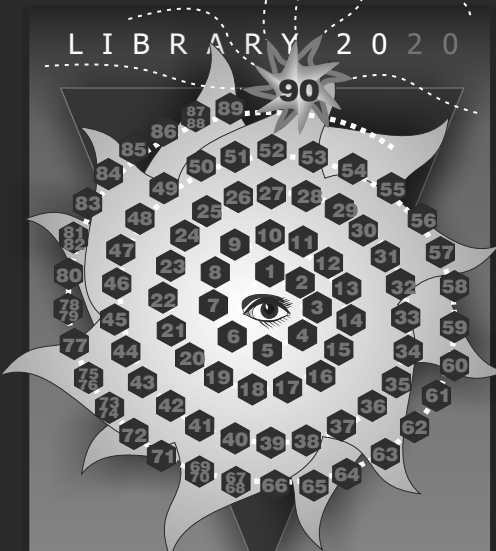
Figure 13: The reaction to the click on the button (ShowMessage)

Finally the free TMS WebCore arrived!

OK

USB HID support in kbmMW
#2 – 24×7 stability on Win10
By Kim Bo Madsen

HOW TO IMPROVE STABILITY OF YOUR USB HID DEVICES ON WIN10

I've noticed when attaching a barcode scanner like the Honeywell 1300g to the USB port of a computer running Win10, the scanner works perfectly….
for as long as it is being used, or some additional hours, until power save mode or a Windows house keeping kicks in.

After then, the only way to get the scanner to work again, is to disconnect it physically and reconnect it.

First I suspected power save mode of the USB device. It is important to configure the USB devices that you want running 24×7, to not adhere to power save modes.

Turning off power saving for USB devices
This can be done a couple of places. I recommend updating settings for all the places:
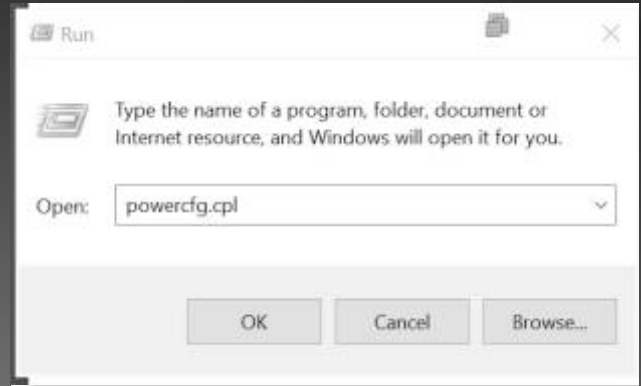
Figure 1: In the Windows power configuration control panel, which can be opened by running powercfg.cpl
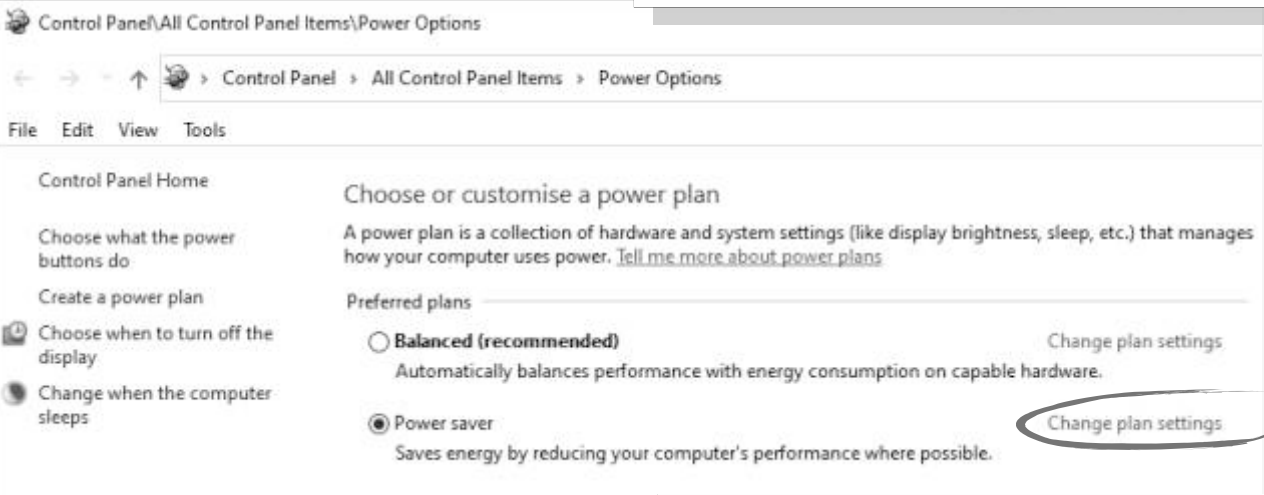
Figure 2: Select Change plan settings for your selected power plan:
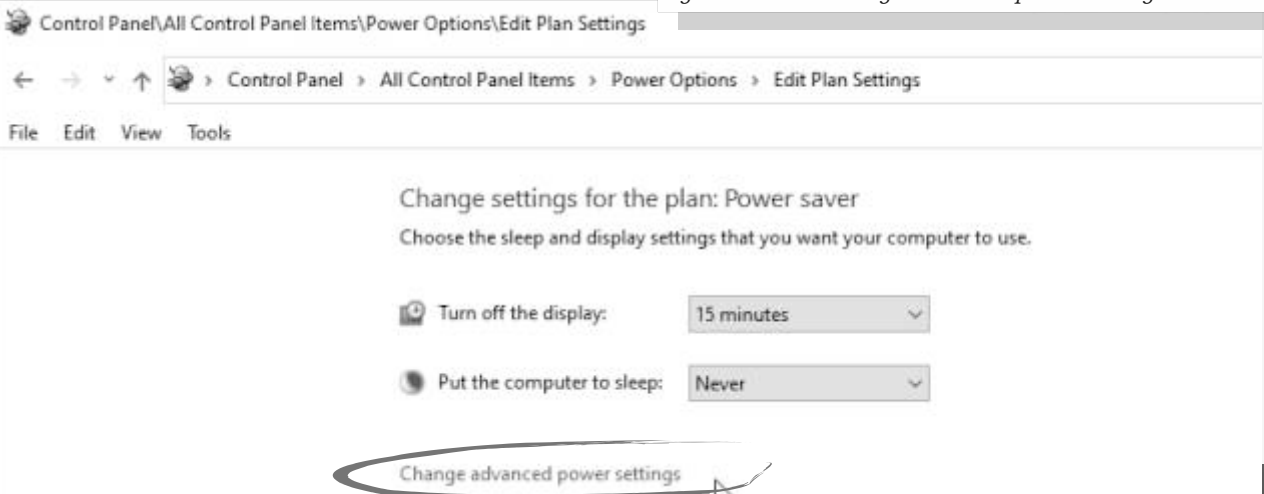
Figure 3: Select Change advanced power settings

Figure 4:
Change USB selective suspend settings to DISABLED

2. On the specific USB driver in device management. Some USB device drivers allows for defining their power save settings.

Open the device manager
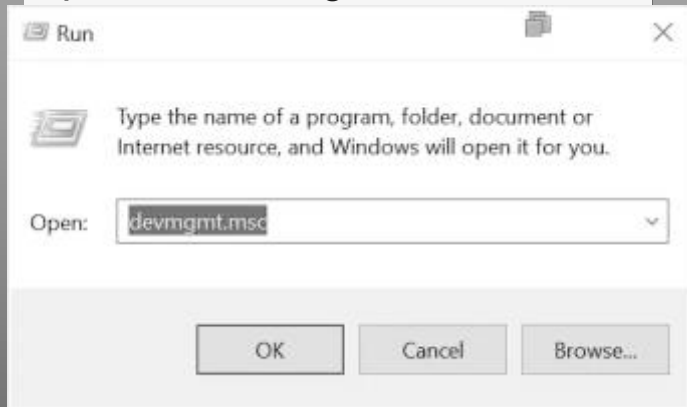
Figure 5: Open the device manager

Figure 6: Locate

In the device manager, locate the Universal Serial Bus controllers section. Each of the USB hub entries usually have power save options you can tune or turn off.
Do that for at least the USB hub which you connected the device to, but I recommend to do it for all USB hubs to avoid problems if you one day choose to plug in the USB device into another port on your computer, which may be driven by a different built in USB hub.

Right click the USB hub entries, one by one, and click Properties
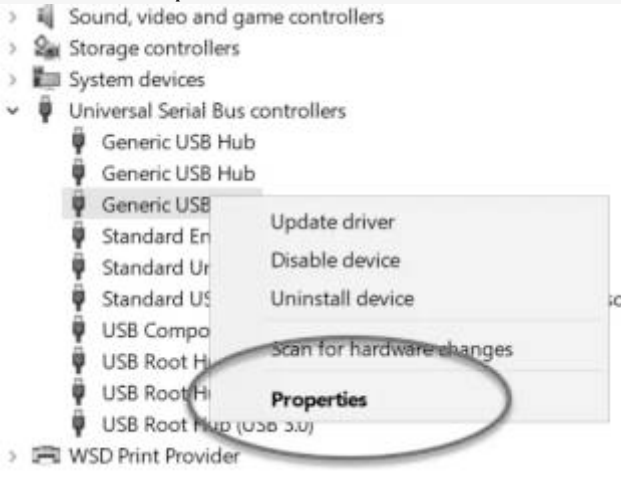
Figure 7: Properties
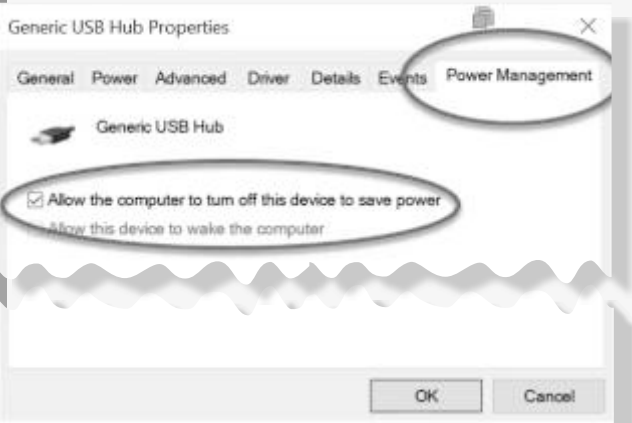
Figure 8: Alter the setting

Make sure to uncheck "Allow the computer to turn of this device to save power".

If you do not see the Power Management tab, the device do not provide additional features for setting its power management, and you can ignore it.

BUT IT DIDN'T REALLY SOLVE THE PROBLEM
After disabling power saving features for the USB (which is a requirement to do to get a stable, always running USB device).
I however still experienced problems when the devices were not used over night.

At first I suspected the kbmMW HID code to be the culprit, but it turned out it was not.

Instead it turns out that the default Microsoft driver for Barcode scanners configured in HID mode, simply dies after some hours of non use. The exact reason for it to die, is unknown to me, but I can guess it has to do with some Win10 scheduled operations, like updates, system clean up and even power save modes that may be triggered even though we have tried our best to disable those.
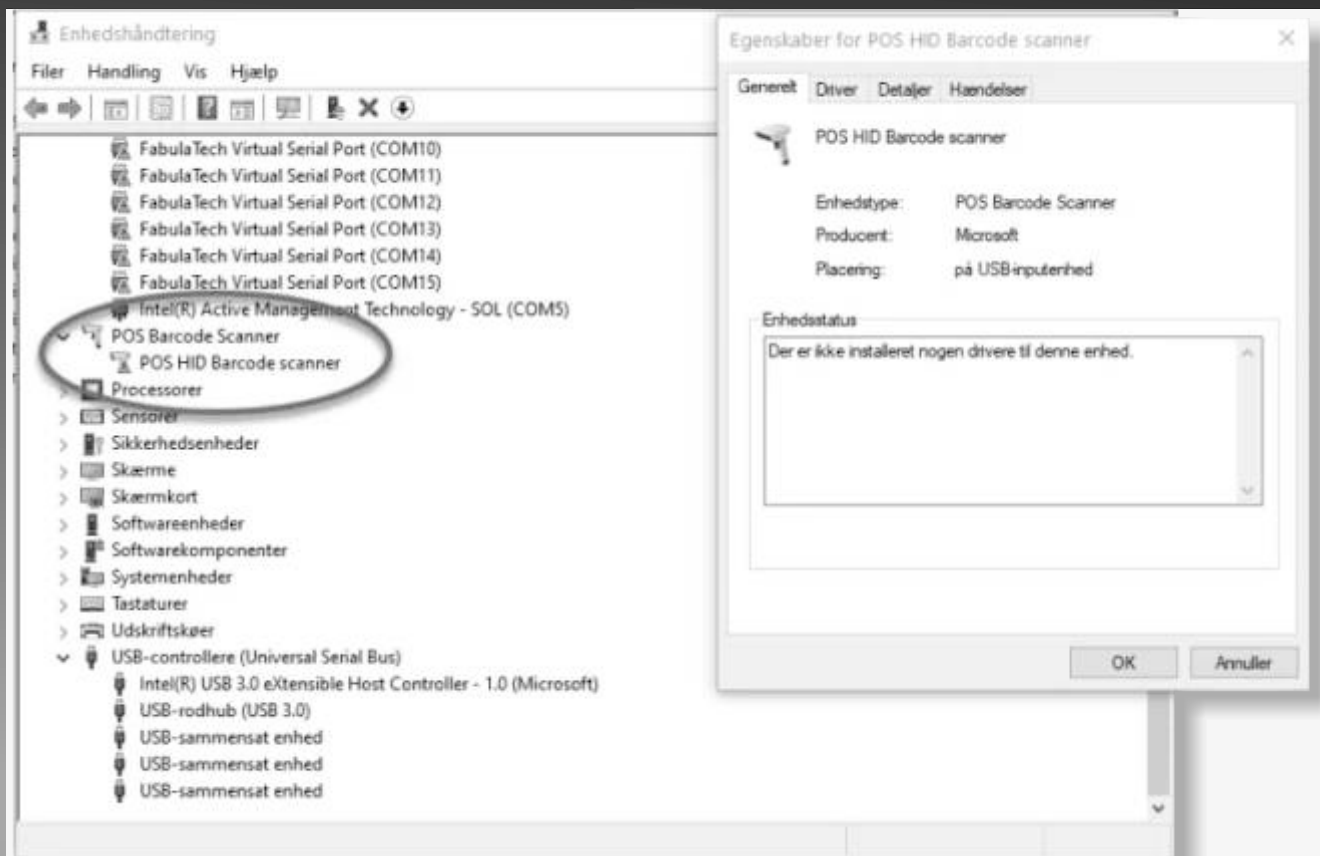
Figure 9: Pos HID Barcode Sanner

As you can see, there is a small yellow triangle on the POS HID Barcode scanner. It is an indicator of that the Windows HID driver has stopped communicating with the device. The device is actually very much alive, and I have verified that with USB debug/trace/scanner software, but the driver simply died, and will not "undie" until the device is reconnected.

Searching high and low for a solution to this problem found no real solution, using the driver default used by Windows 10.

However the salvation came, when I discovered that Win10 also comes with the older Win7 HID driver installed out of the box. And that driver does not die randomly.

So the solution is to right click the HID device, click on the Driver tab, and select Update driver. Then select "Browse my computer for driver software"

Update drivers - Logitech USB Input Device

How do you search for drivers?

→ Automatically search for drivers
Windows searches your computer for the best available driver and installs it on your device

→ Browse the computer for drivers
Manually locate and install a driver

Cancel
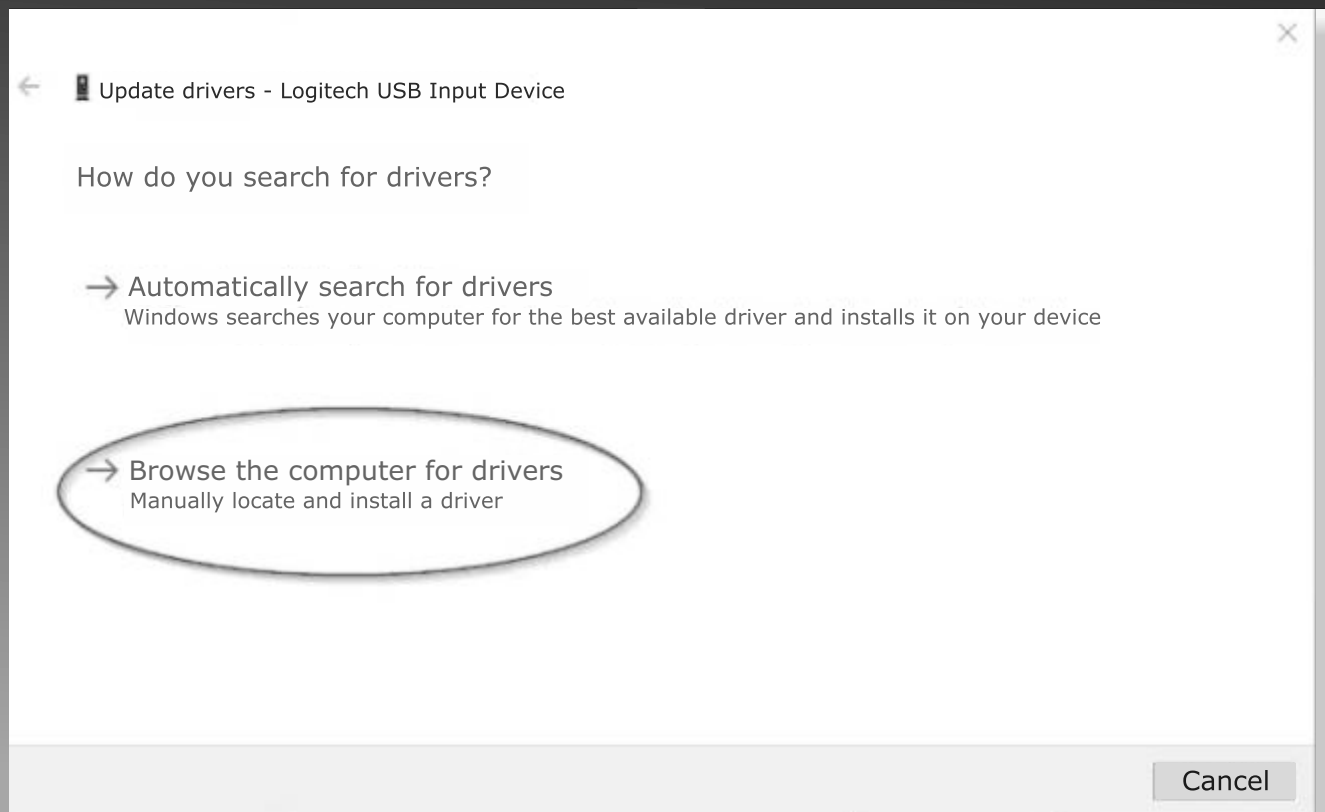
Figure 10: Browse the computer

Update drivers - Logitech USB Input Device

Browse your computer for drivers

Search for drivers in this location

C:\Users\Kim Bo Madsen\Documents   Browse

☑ Include subfolders

→ Let me choose from a list of available drivers on my computer
This list shows available drivers that are compatible with the device, as well as all
drivers in the same category as the device

Next   Cancel
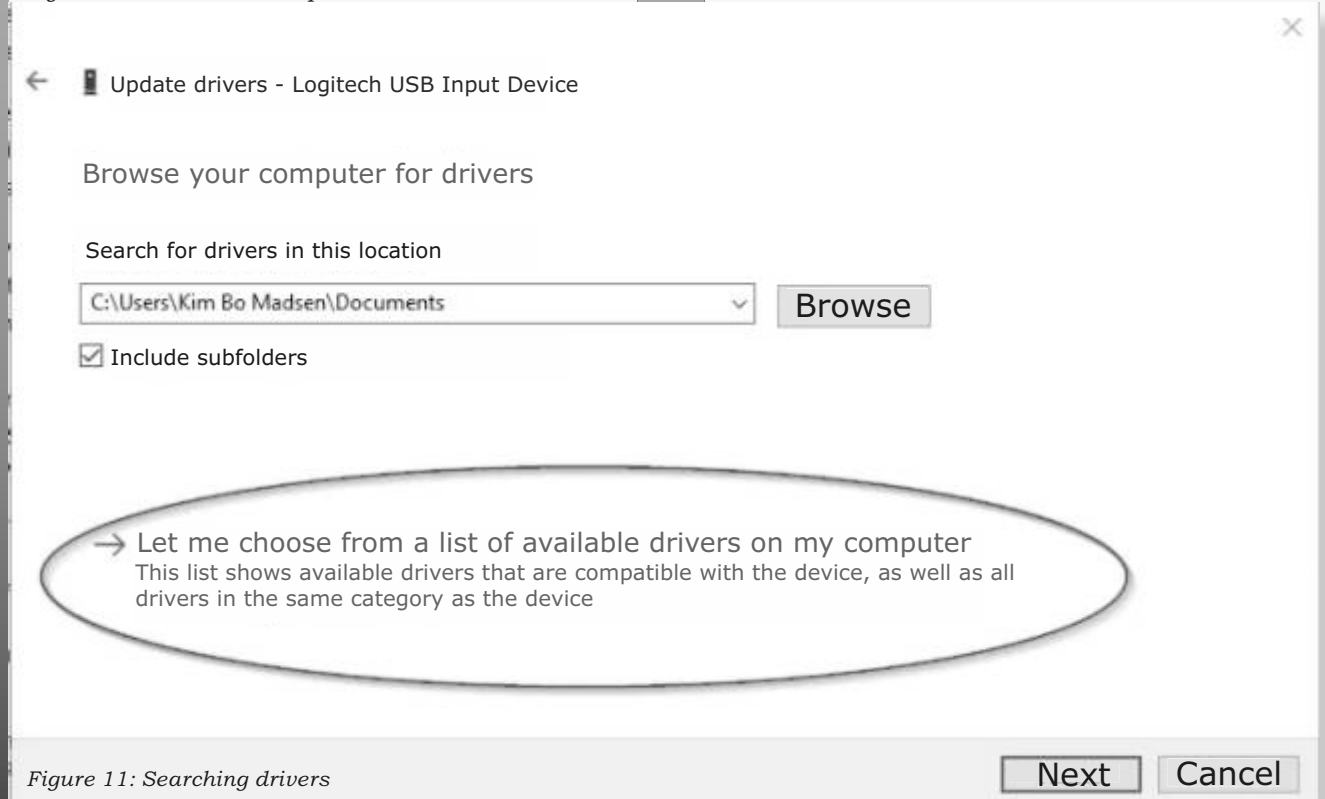
Figure 11: Searching drivers

Update drivers - Logitech USB Input Device

Select the device driver you want to install for the hardware

Select the manufacturer and model of your hardware device, then click Next.
Click Have Disk / CD if an installation disk came with the hardware

☑ Vis kompatibel hardware

Model
HID-compliant device
POS HID Barcode scanner

The driver is digitally signed.
Tell me why driver signings are important

Has floppy disk / CD

Next   Cancel

Figure 12: Found

Let it install and voila, your HID driver is now stable and your device can run 24×7 on Win10.

I have chosen to do this change also for USB connected magnetic stripe readers, even though they seemingly do not use the unstable Win10 driver from outset, but making the change to the Win7 driver (which works for those magnetic stripe readers too), makes me more confident in those also running 24×7 and I have not had an issue with them with the older driver.

By Kim Bo Madsen

This blog post is about a new discovery I have made in Delphi, and the way to circumvent it.

THE GENERICS COMPILER BUG

I had a weird issue in kbmMW SmartBinding's DefineData feature, which is used as a way to live push external data into kbmMW's binding mechanism, making it possible to update visual components on the fly based on external data.

```
...
 TMyObject = class
 private
 public
   destructor Destroy; override;
 end;

...
implementation

...
destructor TMyObject.Destroy;
begin
   inherited;
end;


procedure TForm6.FormCreate(Sender: TObject);
var o:TMyObject;
begin
   o:=TMyObject.Create;
   Binding.DefineData('object',o,true);
   o:=TMyObject.Create;
   Binding.DefineData('object',o,true);
   Binding.UndefineAllData;
end;
```

The DefineData method exists in 4 different overloaded variations, 2 of generic types and 2 of non generic types.

```
function DefineData(const AName:string; const AData:IkbmMWBindingData):TValue; overload;
```

```
function DefineData(const AName:string; const AData:TValue; const AManaged:boolean = false):TValue; overload;
```

```
function DefineData<T:class>(const AName:string; const AData:T):TValue; overload;
```

```
function DefineData<T>(const AName:string; const [ref] AData:T; const AAsReference:boolean):TValue; overload;
```

DefineData can be told to manage the data provided for it, meaning it is responsible for freeing the data when it is no longer used by the binding framework, which is a powerful way to leave the responsibility of the lifetime of the data to `SmartBind`..... if it only worked!

I discovered that despite my best efforts, the data were not automatically freed.
I looked high and low to made sure that the intricate internal routines handling all this, actually worked, before I realized what the problem was.
First some test code showing the use of `DefineData` where I noticed it failing:

(*See right top*).

Finally **UndefineAllData** removes all data definitions regardless of names. The second instance of **TMyObject** ought to disappear here, but alas it did not.

So OBVIOUSLY there must be a bug in **DefineData** and **UndefineAllData**.

But no, that is not the case.

The problem in this case is that the compiler chooses the wrong **overloaded** method when it compiles the code.

Stepping into the code, I realised that I stepped into the one with this signature:

```
function DefineData<T>(const AName:string; const [ref] AData:T; const AAsReference:boolean):TValue;
```

Basically I create a simple object, defined it to **SmartBind** with the argument **true**, meaning that **SmartBind** must take ownership of the object instance.
Then I created a new one and defined that under the same name as the previous one, which should result in the previous **TMyObject** should be **freed**, which did not happen.

Which you can see is a generic version, and thus should not be selected unless I had used the syntax: → *next page*

```
Binding.DefineData<TObject>('object',o,true);
```

It should have selected the one with the signature:

```
function DefineData(const AName:string; const AData:TValue; const AManaged:boolean = false):TValue;
```

Impressively the compiler actually figured out (by the argument given for **AData)** that the generic type of **DefineData** should be

```
DefineData<TMyObject>.
```

That is impressive, but absolutely not intended. The compiler should never deduce the generic type based on the type of a **T argument**, but only on what has been given as the type between the generics brackets <>.

The only way to circumvent this bug is to make sure that there are no two methods (generics or not) that have potentially compatible argument lists while having the same method name.

Usually the compiler would issue a compile error, in case multiple overloaded methods matched the parameter list, but not in this case, which is not a bug, but correct behaviour. The bug is that it selects the wrong one.

As it is calling an incorrect method, it is a rather serious bug, which needs to be fixed.
I have for the moment only tested this issue in 10.3.
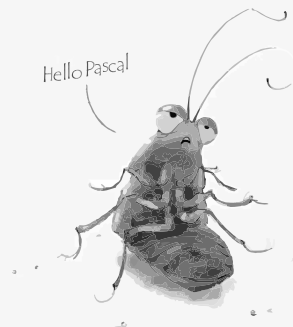
Bumblebees fly at a speed of 54 km

 Biting open the stem of a flower…
 …and using its tongue to drink the nectar.

Bumblebees generally visit flowers that exhibit the bee pollination syndrome and these patches of flowers may be up to 1–2 km from their colony. They tend to visit the same patches of flowers every day, as long as they continue to find nectar and pollen there, a habit known as pollinator or flower constancy.
While foraging, bumblebees can reach ground speeds of up to 15 m/s (54 km/h).

Bumblebees use a combination of colour and spatial relationships to learn which flowers to forage from.
They can also detect both the presence and the pattern of electric fields on flowers, which occur due to atmospheric electricity, and take a while to leak away into the ground.
They use this information to find out if a flower has been recently visited by another bee.
Bumblebees can detect the temperature of flowers, as well as which parts of the flower are hotter or cooler and use this information to recognise flowers.
After arriving at a flower, they extract nectar using their long tongues ("glossae") and store it in their crops.
Many species of bumblebees also exhibit "nectar robbing": instead of inserting the mouthparts into the flower in the normal way, these bees bite directly through the base of the corolla to extract nectar, avoiding pollen transfer.

A bumblebee "nectar robbing" a flower:Pollen is removed from flowers deliberately or incidentally by bumblebees. Incidental removal occurs when bumblebees come in contact with the anthers of a flower while collecting nectar. When it enters a flower, the bumblebee's body hairs receive a dusting of pollen from the anthers. In queens and workers this is then groomed into the corbiculae (pollen baskets) on the hind legs where it can be seen as bulging masses that may contain as many as a million pollen grains.
Male bumblebees do not have corbiculae and do not purposively collect pollen.

Bumblebees are also capable of buzz pollination, in which they dislodge pollen from the anthers by creating a resonant vibration with their flight muscles.

In at least some species, once a bumblebee has visited a flower, it leaves a scent mark on it. This scent mark deters bumblebees from visiting that flower until the scent degrades.
This scent mark is a general chemical bouquet that bumblebees leave behind in different locations (*e.g. nest, neutral, and food sites*), and they learn to use this bouquet to identify both rewarding and unrewarding flowers, and may be able to identify who else has visited a flower.
Bumblebees rely on this chemical bouquet more when the flower has a high handling time, that is, where it takes a longer time for the bee to find the nectar once inside the flower.

Once they have collected nectar and pollen, female workers return to the nest and deposit the harvest into brood cells, or into wax cells for storage.
Unlike honeybees, bumblebees only store a few days' worth of food, so are much more vulnerable to food shortages. Male bumblebees collect only nectar and do so to feed themselves. They may visit quite different flowers from the workers because of their different nutritional needs.

Asynchronous flight muscles
Bees beat their wings about 200 times a second.
Their thorax muscles do not contract on each nerve firing, but rather vibrate like a plucked rubber band.
This is efficient, since it lets the system consisting of muscle and wing operate at its resonant frequency, leading to low energy consumption.
Further, it is necessary, since insect motor nerves generally cannot fire 200 times per second.
These types of muscles are called asynchronous muscles and are found in the insect wing systems in families such as Hymenoptera, Diptera, Coleoptera, and Hemiptera. Bumblebees must warm up their bodies considerably to get airborne at low ambient temperatures.
Bumblebees can reach an
 internal thoracic
temperature of 30 °C (86 °F)
using this method.

James Lindsey at Ecology of Commanster, CC BY-SA 2.5, via Wikimedia Commons

Bernie, Public domain, via Wikimedia Commons

KBMMW PROFESSIONAL AND ENTERPRISE EDITION
V. **5.14** RELEASED!

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

kbmMemTable is the fastest and most feature rich in memory table for Embarcadero products.
- **Easily supports large datasets with millions of records**
- **Easy data streaming support**
- **Optional to use native SQL engine**
- **Supports nested transactions and undo**
- **Native and fast build in M/D, aggregation/grouping, range selection features**
- **Advanced indexing features for extreme performance**

♦ **New HTTP and kbmMW server status message handling support in Smart services**

♦ **New support for TkbmMWGenericMagneticStripeReaderHID and TkbmMWGenericBarcodeReaderHID**

♦ **New support for pivot based counters in the ORM**

♦ **New support for transport native file sending for HTTP.Sys transport**

♦ **New support for automatic GZIP compression of responses from HTTP Smart services**

♦ **Several feature improvements and fixes.**

♦ **More features improvements and fixes.**

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

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

.::: COMPONENTS
      DEVELOPERS 4