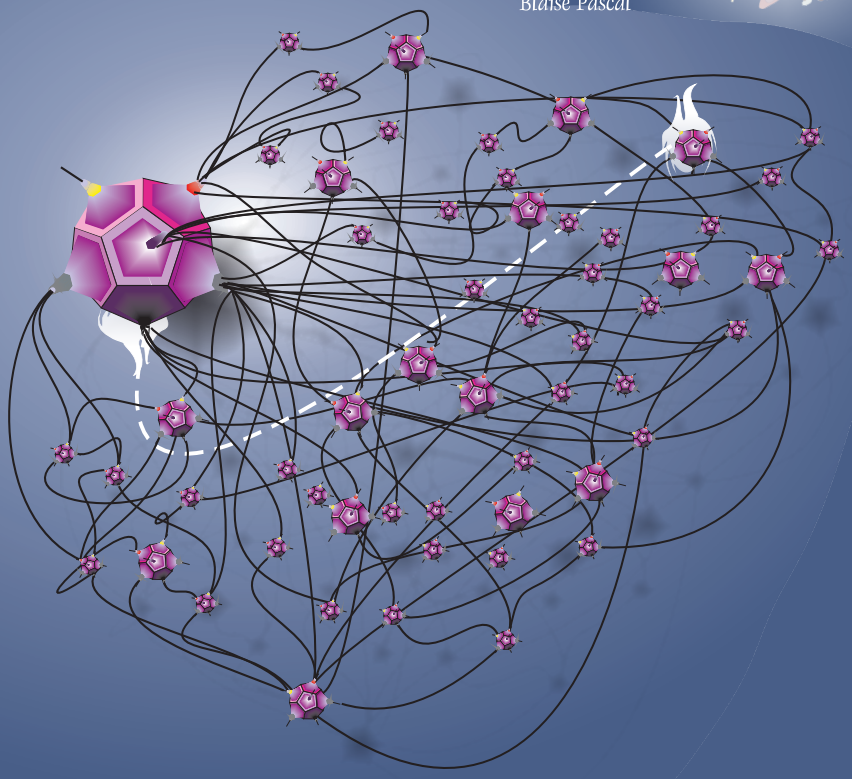
BLAISE PASCAL MAGAZINE 87/88

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

Blaise Pascal



Artificial Intelligence: AI Deep Learning
Artificial Intelligence: The Neural Network
Artificial Intelligence: Creating a Decision Tree
maXbox Starter Image Classifier 76 Machine Learning with CAI
Debugging your memory leaks with kbmMW
What is G5 en what does it mean to us
The new Delphi Sydney
Image Line - FL Studio
TMS Webcore for Visual Studio
kbmMW The compile tool

BLAISE PASCAL MAGAZINE 87/88

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

Blaise Pascal

CONTENT

ARTICLES

ADVERTISERS

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

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

Niklaus Wirth

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

Peter Bijlsma -Editor
peter @ blaisepascal.eu

Dmitry Boyarintsev
dmitry.living @ gmail.com

Michaël Van Canneyt,
michael @ freepascal.org

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

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

Benno Evers
b.evers
@ everscustomtechnology.nl

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

Holger Flick
holger @ flixments.com

Primož Gabrijelčič
www.primoz @ gabrijelcic.org

Mattias Gärtner
nc-gaertnma@netcologne.de

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

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

John Kuiper
john_kuiper @ kpnmail.nl

Wagner R. Landgraf
wagner @ tmssoftware.com

Vsevolod Leonov
vsevolod.leonov@mail.ru

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

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

Kim Madsen
www.component4developers

Boian Mitov
mitov @ mitov.com

Jeremy North
jeremy.north @ gmail.com

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

Howard Page Clark
hdpc @ talktalk.net

Heiko Rompel
info @ rompelsoft.de

Wim Van Ingen Schenau -Editor
wisone @ xs4all.nl

Peter van der Sman
sman @ prisman.nl

Rik Smit
rik @ blaisepascal.eu

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

B.J. Rao
contact @ intricad.com

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

Anton Vogelaar
ajv @ vogelaar-electronics.com

Robert Welland
support @ objectpascal.org

Siegfried Zuhr
siegfried @ zuhr.nl

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

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

Correctors
Howard Page-Clark, Peter Bijlsma

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

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

Subscriptions (2019 prices)

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

Member and donator of WIKIPEDIA

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

From your editor

This very special item has taken quite sometime to create.
First of all because without having a break – we finished the Lazarus Handbook – I started on this issue and then something I hadn't counted on: I had an accident – broke my hip. So I must apologize that it took so long to create a new issue.

Here is a short overview of the various articles that are published in this item:
In this issue I try to do several things: I wrote an article about Artificial Intelligence, it explains for the very beginner what it is and how it really works, not in depth but mostly explaining how that "engine" runs.

There is also an article on how to use it for recognizing human persons and that is a project you could run yourself as developer. This subject will become more and more interesting because of its social significance: from frightening up to beautiful vistas and incredible solutions. It is a technique that we urgently need and that enables us to make better but also more responsible decisions. So I think we as developers will need to know what it is and how to handle.
Already now there are a large number of solutions that we can try and use.
Google is very good in it, and often you can use these as building blocks.

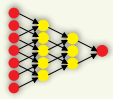
A solution will never give you an exact answer but will always be a percentage that comes very close to your goal… It' s the first time that a programmatic maybe becomes possible.

5G is an other article that tries to explain about technique behind this yet to come mobile frequency.

I wrote also about a very interesting company: Image line, the story is in itself incredible and they wrote their program in Pascal and assembler. The beautiful user interfaces makes one jealous.
And the number of downloads: 30.000 per day.

Lazarus has published the latest version 2.0.10, actually only a bug release no great new technical experiences, FPC had been updated to version 3.0.2. Pas2Js was now finally extended with Delphi – compatible generics.
This time we are lucky to have two book reviews: One about TMS Webcore (English and German) and one about the Lazarus Handbook. I also found out that the TMSSoftware WEB core developers guide is available and contains an enormous number of examples and how to's. For everyone that has bought the Webcore framework a must, and maybe you might even print it as a reference guide. From now on I will try to make several examples available to you in as well Lazarus as Delphi, starting this issue.
Talking about Webcore Bruno Fierens created something very special: TMS WEB Core for Visual Studio Code, which means that you can do things in that special environment, that will amaze you.
Read the article at page 86!

Because of these corona times we will have to find new way of creating events and we think we can do something about that: we will try to make a free version of a Zoom-like app that will be able to connect you all and in person. These are services which should be available for everyone without cost and we try to achieve that…



INTRODUCTION:

In this article I would like to explain in a very understandable way how the subject of Artificial Intelligence can be used and in this edition I would like to list all the various terms that are used. This article is not difficult to read or understand on one condition: it must be read in great concentration and details are very demanding. It is not a walk over but no rocket science as well.
How a simple Neural Network functions by creating an example - and for the future: I will try to create the same project as a peace of code so that it might become a starting point of building your own and better and deeper understanding of the subject. Some names of books that have helped will also be listed. I found over the research I had to do for this article, that all details of the subjects are available but often very shattered or forgotten to explain: people often don't realize that for starters in this field the terms are not only very confusing but also very much hidden, they are NOT explained. I have tried to do so and it took me very much time to get that done.

TERMS

The easiest way to think about artificial intelligence is in the context of a human. After all humans are the most intelligent creatures we know off. **(1)AI** is a broad branch of computer science: the goal of **AI** is to create systems that can function intelligently and independently.
Humans can speak and listen to communicate through language.
This is the field of **(2) speech recognition**. Much of speech recognition is statistically based. Hence it's called **(3)statistical learning .**

Humans can write and read text in a language. This is the field of **(4)NLP** or **natural language processing.**
Humans can see with their eyes and process what they see. This is the field of **computer vision. (5)Computer vision** falls under the symbolic way for computers to process information. Recently there has been another way. Humans recognize the scene around them through their eyes which create images of that world.

This is the field of **(6)image processing** - which even though is not directly related to **AI** - is required for **computer vision.**
Humans can understand their environment and move around fluidly: this is the field of **(7) robotics.**

Humans have the ability to see patterns such as grouping of like objects. This is the field of **(8)pattern recognition.**
Machines are even better at **pattern recognition** because they can use more data and dimensions of data. This is the field of **(9)machine learning.**

Now let's talk about the human brain: the human brain is a network of neurons and we use these to learn things. If we can replicate the structure and the function of the human brain we might be able to get cognitive capabilities in machines. This is the field of **(10)neural networks.**
If these networks are more complex and deeper and we use those to learn complex things, That is the field of **(11)deep learning.**

There are different types of **deep learning** and machines which are essentially different techniques to replicate what the human brain does.
If we get the network to scan images from left to right top to bottom it's a **(12)convolution neural network.** A **CNN** is used to recognize objects in a scene. this is how computer vision fits in. An object recognition is accomplished through AI. Humans can remember the past, like what you had for dinner last night. Well at least most of you. We can get a **neural network** to remember a limited past.

This is a **(13)recurrent neural network**. As you see there are two ways an eye works: one is *symbolic* based and another is *data* based for the database side called **(9)machine learning.** We need to feed the Machine lots of data before it can learn. For example: if you had lots of data for sales versus advertising spend.
You can plot that data to see some kind of a pattern. If the machine can learn this pattern then it can make predictions based on what it has learned.

Overview and coherence

While one or two or even three dimensions is easy for humans to understand and learn machines can learn in many more dimensions like even hundred or thousands.
That's why machines can look at lots of high dimensional data and determine patterns.

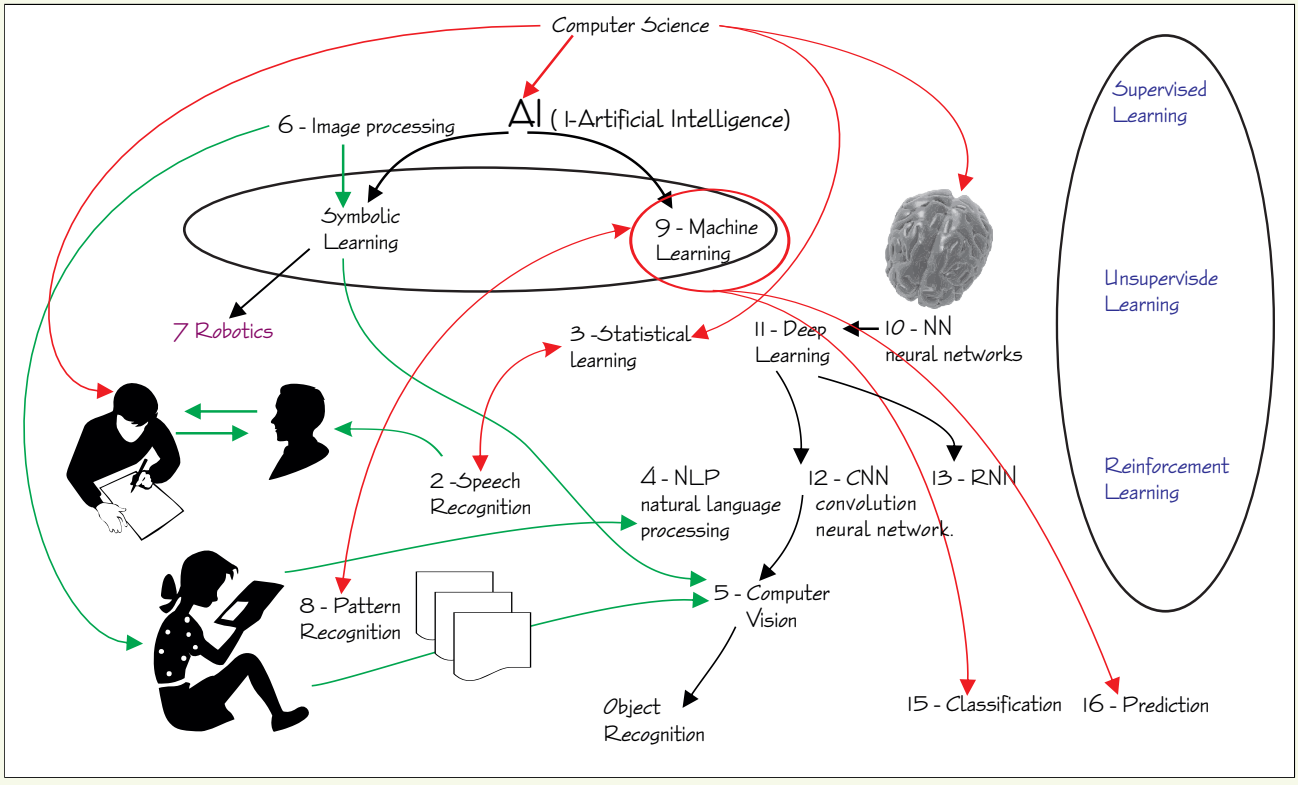
ONCE IT LEARNS THESE PATTERNS IT CAN MAKE PREDICTIONS THAT HUMANS CAN'T EVEN COME CLOSE TO.

We can use all these machine learning techniques to do one of two things:
(15)classification or **(16)prediction.**
As an example: when you use some information about customers to assign new customers to a group - like young adults - then you are classifying that customer if you use data to predict if they're likely to defect to a competitor.
Then you're making a **prediction.**

There is another way to think about **learning algorithms used for AI:** if you train an algorithm with data that also contains the answer then it's called **(17)supervised learning** for example.

When you **train a machine** to recognize your friends by name, you'll need to identify them for the computer.
If you train an **algorithm** with data, where you want the machine to figure out the patterns then it's **(18)unsupervised learning**.
For example: you might want to feed the data about celestial objects in the universe and expect the machine to come up with patterns.

In that data by itself, if you give any algorithm a goal and expect the Machine through trial-and-error to achieve that goal then it's called **(19)reinforcement learning.** *(See page 14)*
A robot's attempt to climb over a wall until it succeeds is an example of it.



LIST OF TERMS

1. AI (Artificial Intelligence)
2. speech recognition
3. statistical learning
4. NLP Natural Language Processing
5. computer vision
6. image processing
7. robotics
8. pattern recognition
9. machine learning
10. neural networks
11. deep learning
12. convolution neural network. CNN
13. recurrent neural network
14. classification
15. prediction.

learning algorithms used for AI:

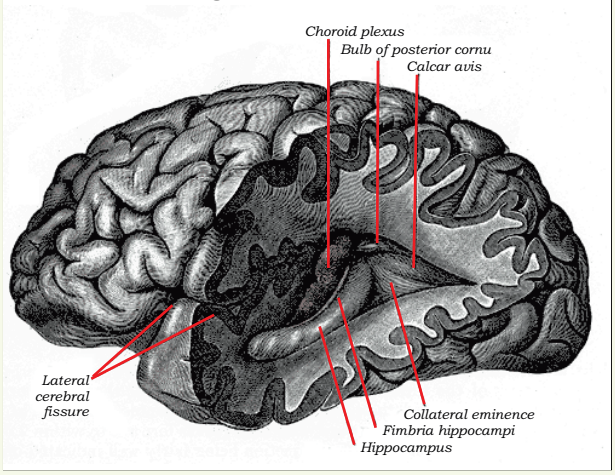
16. supervised learning
17. unsupervised learning.
18. reinforcement learning.

For this article we had the help of WIKIPEDIA

1. AI (ARTIFICIAL INTELLIGENCE)

Artificial intelligence (AI), sometimes called machine intelligence, is intelligence demonstrated by machines, unlike the natural intelligence displayed by humans and animals.

Leading AI textbooks define the field as the study of "intelligent agents":
any device that perceives its environment and takes actions that maximize its chance of successfully achieving its goals. Colloquially, the term "artificial intelligence" is often used to describe machines (or computers) that mimic "cognitive" functions that humans associate with the human mind, such as "learning" and "problem solving".



SOME EXAMPLES:
Here are some examples of applications.

- Siri.
Siri is one of the most popular personal assistant offered by Apple in iPhone and iPad.

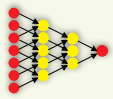
- Tesla.
Not only smartphones but automobiles are also shifting towards Artificial Intelligence. This is one of the best automobiles available until now. The car has not only been able to achieve many accolades but also features like self-driving, predictive capabilities, and absolute technological innovation.

- Alexa
Amazon Alexa, also known simply as Alexa, is a virtual assistant AI technology developed by Amazon, first used in the Amazon Echo smart speakers developed by Amazon Lab126.
It is **capable of voice interaction**, music playback, **making to-do lists, setting alarms, streaming podcasts, playing audiobooks, and providing weather, traffic, sports, and other real-time information, such as news**.

Alexa can also control several smart devices **using itself as a home automation system**. Users are able to extend the Alexa capabilities by installing "skills" (additional functionality developed by third-party vendors, in other settings more commonly called apps such as weather programs and audio features).

Most devices with Alexa allow users to activate the device using a wake-word (such as Alexa or Amazon); other devices (such as the Amazon mobile app on iOS or Android and Amazon Dash Wand) require the user to push a button to activate Alexa's listening mode, although, some phones also allow a user to say a command, such as "Alexa" or "Alexa wake".

Currently, interaction and communication with Alexa are available in several languages.
The problem with these new developments is always privacy regulation, if you use them they wiil make your notations calls and details available to Amazon, maybe even to others.



The problem is to find the same sort possibilities that do not have a built in commercials background and which could **be localized.**

- Cogito
Cogito originally co-founded by Dr. Sandy and Joshua is one of the best examples of the **behavioral version to improve the intelligence of customer support representatives**, currently on the market.
The company is a synthesis of machine **learning and behavioral science to enhance customer collaboration for phone professionals.**
Cogito is applicable on millions of voice calls that take place on a daily basis.
The AI solution analyzes the human voice and provides real-time guidance to enhance behavior.

SOME EXAMPLES:
It employs a very intelligent machine learning process that learns the temperature you like and programs itself in about a week.
Moreover, it will automatically turn off to save energy, if nobody is at home.
In fact, it is a combination of both – artificial intelligence as well as Bluetooth low-energy because some components of this solution will use BLE services and solutions.

- Boxever
Boxever is a company that heavily relies on machine learning to **enhance the customer experience in the travel industry** and conveys micro-moments or experiences that can please the customers.
Boxever significantly improves customer engagement through machine learning and Artificial Intelligence to rule the playing field, helping customers to find new ways and make memorable journeys.
`https://www.boxever.com/`

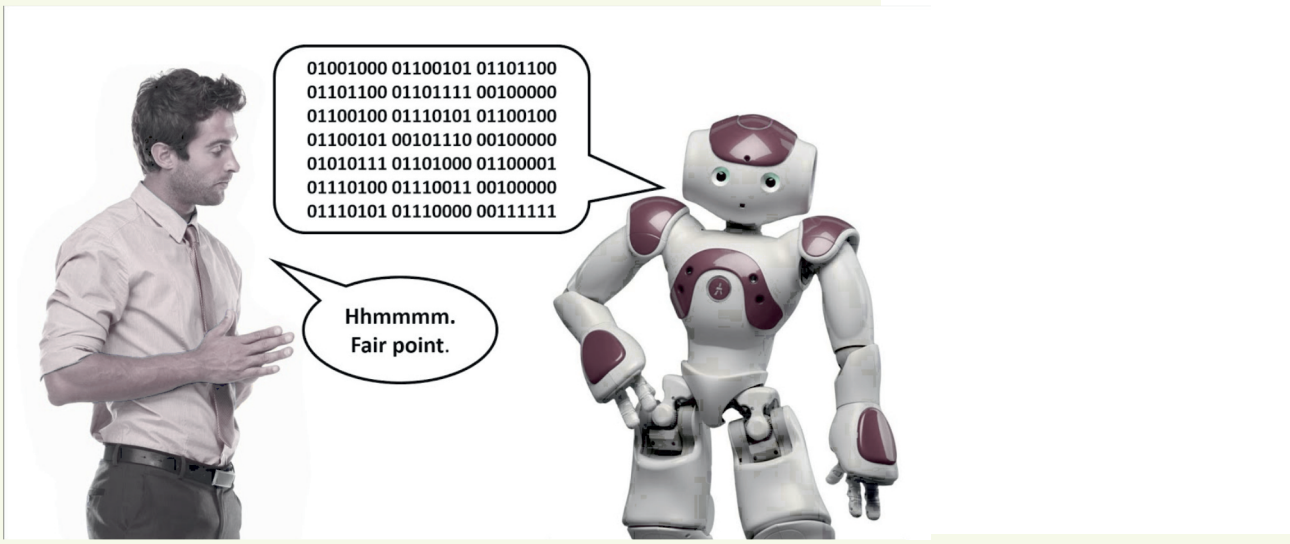
- Flying Drones
The **flying drones** are already shipping products to customers home - though on a test mode. They indicate a powerful machine learning system that can **translate the environment into a 3D model** through sensors and video cameras.
more - using the **Alexa Voice Service**.The sensors and cameras are able to notice the position of the drones in the room by attaching them to the ceiling. Trajectory generation algorithm guides the drone on how and where to move. Using a Wi-Fi system, we can control the drones and **use them for specific purposes – product delivery, video-making, or news reporting**
`https://www.faa.gov/uas/recreational_fliers/`

- Echo
Echo was launched by **Amazon,** which is getting smarter and adding new features.
It is a revolutionary product that can help you to **search the web for information, schedule appointments, shop, control lights, switches, thermostats, answers questions, reads audiobooks, reports traffic and weather, gives info on local businesses, provides sports scores and schedules.**

2. SPEECH RECOGNITION

These addressess give some extra information
`http://shapeshed.com/html5-speech-recognition-api/`
`http://www.google.com/intl/en/chrome/demos/speech.html`
`http://sourceforge.net/projects/tpapro` In addition to this my article about
BRAIN - COMPUTER INTERFACE:
MECHANICAL TELEPATHY PAGE 1/5
BLUETOOTH - INTERNET - BLUETOOTH
in Blaise Pascal Magazine issue nr 82/83
Speech recognition is an interdisciplinary **subfield of computer science and computational linguistics** that develops methodologies and technologies that enable the recognition and translation of s**poken language into text by computers.**
It is also known as **automatic speech recognition (ASR), computer speech recognition or speech to text (STT).**

2 SPEECH RCOGNITION



It incorporates knowledge and research in the computer science, linguistics and computer engineering fields.
Some **speech recognition** systems require "training" (also called "enrollment") where an individual speaker reads text or isolated vocabulary into the system.
The system analyzes the person's specific voice and uses it to fine-tune the recognition of that person's speech, **resulting in increased accuracy.** Systems that do not use training are called "speaker independent systems. Systems that use training are called "speaker dependent".

Speech recognition applications include voice user interfaces such as **voice dialing** (e.g. "call home"), call routing (e.g. "I would like to make a collect call"), domotic appliance control, search key words (*e.g. find a podcast where particular words were spoken*), simple data entry (*e.g., entering a credit card number*), preparation of structured documents (*e.g. a radiology report*), determining speaker characteristics, **Speech-to-text processing** (e.g., word processors or emails), and aircraft (usually termed direct voice input).

The term „voice recognition“ or **speaker identification** refers to identifying the speaker, rather than what they are saying.

Some **speech recognition** systems require "training" (also called "enrollment") where an individual speaker reads text or isolated vocabulary into the system.
The system analyzes the person's specific voice and uses it to fine-tune the recognition of that person's speech, **resulting in increased accuracy.**
Systems that do not use training are called "speaker independent systems.
Systems that use training are called "speaker dependent".

Recognizing the speaker can simplify the task of translating speech in systems that have been trained on a specific person's voice or it can be used to authenticate or verify the identity of a speaker as part of a security process.

- Netflix. ...
Netflix needs no introduction – it is a widely popular content-on-demand service that uses predictive technology to offer recommendations on the basis of consumers' reaction, interests, choices, and behavior.
The **technology examines from a number of records to recommend movies based on your previous liking and reactions.**

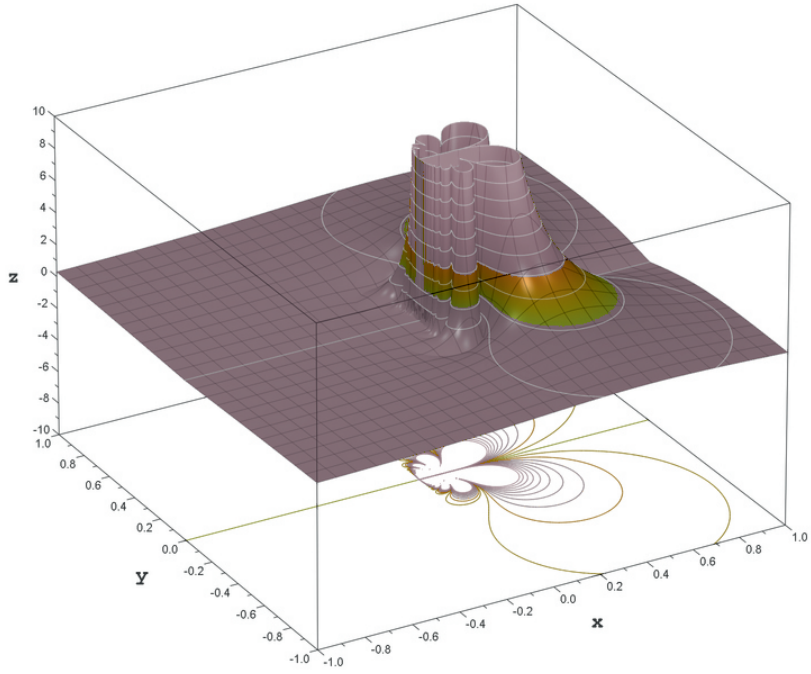
It is turning more intelligent with each passing year. The only the drawback of this technology is that small movies go unnoticed while big films grow and propagate on the platform. But it is still improving and learning to be more intelligent.
https://www.netflix.com/

- Pandora.
Pandora is one of the most popular and highly demanded tech solutions that exist.
It is also called **the DNA of music.**
Depending on 400 musical characteristics, The team of expert musicians individually analyzes the song.
The system is also good at recommending the track record for recommending songs that would never get noticed, despite people's liking. In europe it's not yet established

- Nest (Google)
Nest was one of the most famous and successful artificial intelligence startups and it was acquired by Google in 2014 .
The Nest Learning Thermostat uses behavioral algorithms **to save energy based on your behavior and schedule.**

From the technology perspective, **speech recognition** has a long history with several waves of major innovations.
Most recently, the field has benefited from advances in deep learning and big data.
https://nest.com/
The advances are evidenced not only by the surge of academic papers published in the field, but more importantly by the worldwide industry adoption of a variety of deep learning methods in designing and deploying speech recognition systems.

3. STATISTICAL LEARNING
Statistical learning theory is a framework for machine learning drawing from the fields of statistics and functional analysis.
Statistical learning theory deals with the problem of finding a predictive function based on data.



Statistical learning theory has led to successful applications in fields such as **computer vision, speech recognition, and bioinformatics.**

4. NLP NATURAL LANGUAGE PROCESSING

Natural language processing (NLP) is a subfield of linguistics, computer science, information engineering, and artificial intelligence **concerned with the interactions between computers and human (natural) languages, in particular how to program computers to process and analyze large amounts of natural language data.**

Challenges in natural language processing frequently involve speech recognition, natural language understanding, and natural-language generation.

5. COMPUTER VISION

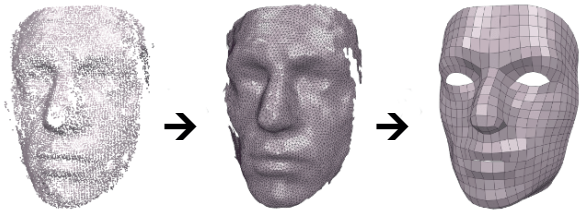
Computer vision is an interdisciplinary scientific field that deals with how computers can gain **high-level understanding from digital images or videos.**
From the perspective of engineering, it seeks to understand and **automate tasks that the human visual system** can do.

Computer vision tasks include methods for acquiring, processing, analyzing and understanding digital images, and **extraction of high-dimensional data from the real world** in order to produce numerical or symbolic information, e.g. in the forms of decisions.
Sub-domains of computer vision include scene reconstruction, event detection, video tracking, object recognition, 3D pose estimation, learning, indexing, motion estimation, visual servoing, 3D scene modeling, and image restoration.

6. IMAGE PROCESSING

Digital image processing is the use of a digital computer to process digital images through an algorithm.
As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing.

It allows a much wider range of algorithms to be applied to the input data and **can avoid problems such as the build-up of noise and distortion during processing**.



Since images are defined over two dimensions (perhaps more) digital image processing may be modeled in the form of multidimensional systems.
Understanding in this context means the transformation of visual images **(the input of the retina)** into descriptions of the world that make sense to thought processes and can elicit appropriate action.

This **image understanding** can be seen as the disentangling of symbolic information from image data using models constructed with the aid of geometry, physics, statistics, and learning theory.

The scientific discipline of computer vision is concerned with the theory behind artificial systems that extract information from images. The image data can take many forms, such as **video sequences, views from multiple cameras, multi-dimensional data from a 3D scanner or medical scanning device**.
The technological discipline of computer vision seeks to apply its theories and models to the construction of computer vision systems.

The generation and development of digital image processing are mainly affected by three factors:

first: the development of computers;
second: the development of mathematics (especially the creation and improvement of discrete mathematics theory);
third: the demand for a wide range of applications in environment, agriculture, military, industry and medical science has increased networks, large-scale simulations of neural microcircuits) and actual biological systems (e.g. in vivo and in vitro neural nets).

Such neural systems can be embodied in machines with mechanic or any other forms of physical actuation.
This includes **robots, prosthetic or wearable systems but also, at smaller scale, micro-machines and, at the larger scales, furniture and infrastructures.**

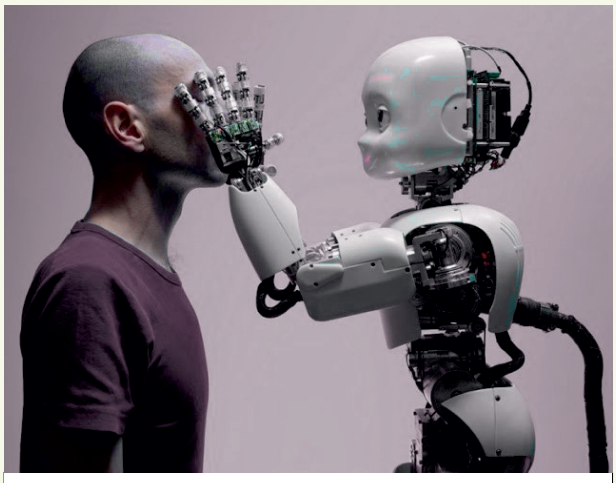
Neurorobotics is that branch of neuroscience with robotics, which deals with the study and application of science and technology of embodied autonomous neural systems like brain-inspired algorithms.
At its core, neurorobotics is based on the idea that the brain is embodied and the body is embedded in the environment.
Therefore, most neurorobots are required to function in the real world, as opposed to a simulated environment.

Beyond brain-inspired algorithms for robots neurorobotics may also involve the design of brain-controlled robot systems.

7. ROBOTICS

Neurorobotics represents the two-front approach to the **study of intelligence.**

Neuroscience attempts to discern what intelligence consists of and how it works **by investigating intelligent biological systems,** while the study of **artificial intelligence** attempts to **recreate intelligence through non-biological, or artificial means.**



Neurorobotics is the overlap of the two, where biologically inspired theories are tested in a grounded environment, with a physical implementation of said model.
The successes and failures of a neurorobot and the model it is built from can provide evidence to refute or support that theory, and give insight for future study.

8. PATTERN RECOGNITION

Pattern recognition is the automated recognition of patterns and regularities in data.

The field of pattern recognition is concerned with the **automatic discovery of regularities in data** through the use of computer algorithms and with the use of these regularities to take actions such as classifying the data into different categories.

This text focuses on machine learning approaches to pattern recognition.
Pattern recognition systems are in many cases trained from labeled "training" data **(supervised learning),** but when no labeled data are available other algorithms can be used to discover previously unknown patterns (**unsupervised learning**).
Machine learning is strongly related to pattern recognition and originates from artificial intelligence. **KDD (knowledge discovery in databases**) and data mining have a larger focus on unsupervised methods and stronger connection to business use.
Pattern recognition focuses more on the signal and also takes acquisition and **Signal Processing** into consideration.

Its originated in engineering, and the term is popular in the context of computer vision: a leading computer vision conference is named **Conference on Computer Vision and Pattern Recognition.**

In **pattern recognition**, there may be a higher interest to formalize, explain and visualize the pattern, while machine learning traditionally focuses on maximizing the recognition rates.

Yet, all of these domains have evolved substantially from their roots in artificial intelligence, engineering and statistics, and they've become increasingly similar by integrating developments and ideas from each other.

An example of **pattern recognition** is **classification,** which attempts to assign each input value to one of a given set of classes (for example, **determine whether a given email is "spam" or "non-spam"**).

Pattern recognition is a more general problem that encompasses other types of output as well.
Other examples are **regression,** a defensive reaction to some unaccepted impulses which assigns a real-valued output to each input; sequence labeling, which assigns a class to each member of a sequence of values (*for example, part of speech tagging, which assigns a part of speech to each word in an input sentence*); and parsing, which assigns a parse tree to an input sentence, describing the syntactic structure of the sentence.

Pattern recognition algorithms generally aim to provide a reasonable answer for all possible inputs and to perform "most likely" matching of the inputs, taking into account their statistical variation. **This is opposed to pattern matching algorithms, which look for exact matches in the input with pre-existing patterns.**
A common example of a pattern-matching algorithm is regular expression matching, which looks for patterns of a given sort in textual data and is included in the search capabilities of many text editors and word processors.

9. MACHINE LEARNING

illustration found at: https://axveco.com/en/why-should-top-managers-learn-to-apply-automated-machine-learning/

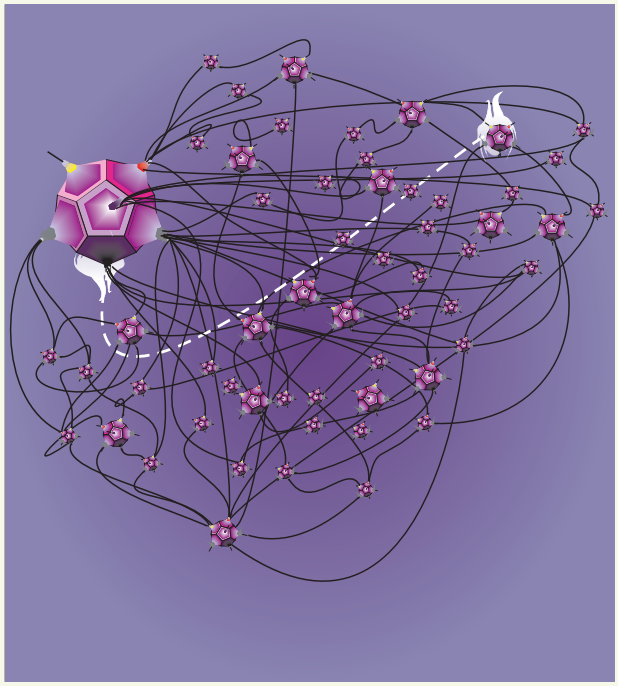
Machine learning (ML) is the study of computer **algorithms that improve automatically through experience**.
It is seen as a subset of artificial intelligence. Machine learning algorithms build a mathematical model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so.
Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or infeasible to develop conventional algorithms to perform the needed tasks.

Machine learning is closely related to computational statistics, which focuses on making predictions using computers. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

10. NEURAL NETWORKS

A **neural network** is a network or circuit of neurons, or in a modern sense, an artificial neural network, composed of artificial neurons or nodes. Thus a neural network is either a **biological neural network**, made up of real biological neurons, or an **artificial neural network,** for solving artificial intelligence (AI) problems.

The connections of the biological neuron are **modeled as weights**. A positive weight reflects an excitatory connection, while negative values mean inhibitory connections. All inputs are modified by a weight and summed. This activity is referred to as a linear combination. Finally, an activation function controls the amplitude of the output.

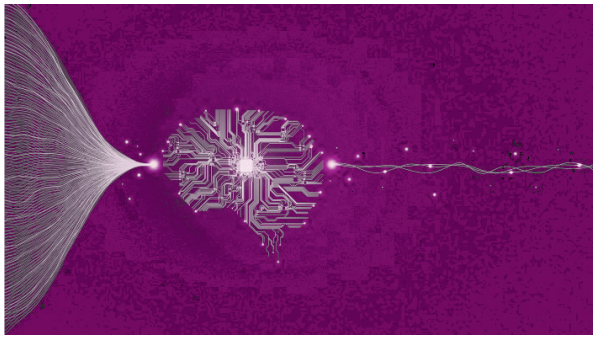


For example, an acceptable range of output is usually between 0 and 1, or it could be −1 and 1.

These artificial networks may be used for predictive modeling, adaptive control and applications where they can be trained via a dataset.

Self-learning resulting from experience can occur within networks, which can derive conclusions from a complex and seemingly unrelated set of information.

11. DEEP LEARNING



Illsutration found at https://becominghuman.ai/https-medium-com-katdare-swanand-college-student-to-machine-learning-enthusiast-af7e288e21a7

Deep learning is part of a broader family of machine learning methods based on artificial neural networks with representation learning. Learning can be supervised, semi-supervised or unsupervised.

Deep learning architectures such as deep neural networks, deep belief networks, recurrent neural networks and convolutional neural networks have been applied to fields including computer vision, machine vision, speech recognition, natural language processing, audio recognition, social network filtering, machine translation, bioinformatics, drug design, medical image analysis, material inspection and board game programs, where they have produced results comparable to and in some cases surpassing human expert performance.

Artificial neural networks (ANNs) were inspired by information processing and distributed communication nodes in biological systems. ANNs have various differences from biological brains.

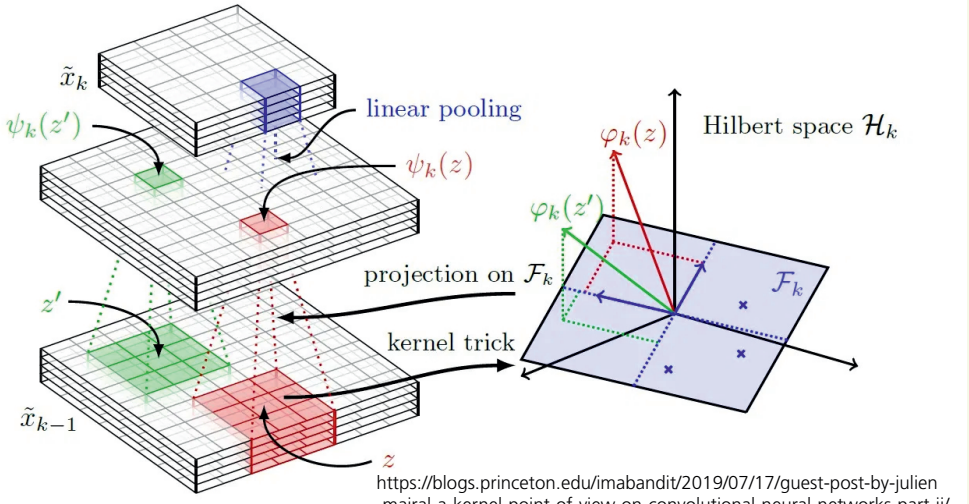
11. DEEP LEARNING / 12 CONVOLUTIONAL NEURAL NETWORK 13 RECURRENT NEURAL NETWORKS

Specifically, neural networks tend to be static and symbolic, while the biological brain of most living organisms is dynamic (plastic) and analog.

The adjective "deep" in deep learning comes from the use of multiple layers in the network. Early work showed that a linear perceptron cannot be a universal classifier, and then that a network with a nonpolynomial activation function with one hidden layer of unbounded width can on the other hand so be.

Deep learning is a modern variation which is concerned with an unbounded number of layers of bounded size, which permits practical application and optimized implementation, while retaining theoretical universality under mild conditions.

In deep learning the layers are also permitted to be heterogeneous and to deviate widely from biologically informed connectionist models, for the sake of efficiency, trainability and understandability, whence the "structured" part.



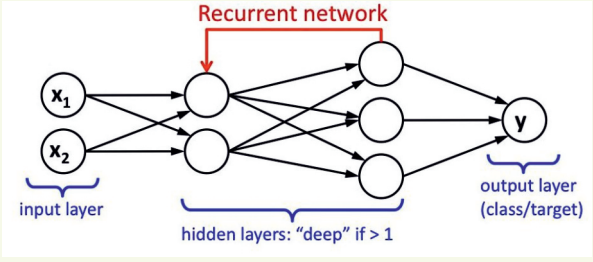
https://blogs.princeton.edu/imabandit/2019/07/17/guest-post-by-julien-mairal-a-kernel-point-of-view-on-convolutional-neural-networks-part-ii/

12. CONVOLUTION NEURAL NETWORK.(CNN)

In deep learning, a convolutional neural network (CNN, or ConvNet) is a class of deep neural networks, most commonly applied to **analyzing visual imagery.**
They are also known as **shift invariant** or **space invariant artificial neural networks (SIANN)**, based on their shared-weights architecture and translation invariance characteristics.
They have applications in image and video recognition, recommender systems, image classification, medical image analysis, natural language processing, and financial time series.
Convolutional networks were inspired by biological processes in that the connectivity pattern between neurons resembles the organization of the animal visual cortex. Individual cortical neurons respond to stimuli only in a restricted region of the visual field known as the receptive field. The receptive fields of different neurons partially overlap such that they cover the entire visual field. CNNs use relatively little pre-processing compared to other image classification algorithms. This means that the network learns the filters that in traditional algorithms were hand-engineered. This independence from prior knowledge and human effort in feature design is a major advantage.

13. RECURRENT NEURAL NETWORK

A recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence.
This allows it to exhibit temporal dynamic behavior. Derived from feedforward neural networks, RNNs can use their internal state (memory) to process variable length sequences of inputs.



This makes them applicable to tasks such as **unsegmented, connected handwriting recognition** or **speech recognition.**

The term **"recurrent neural network"** is used indiscriminately to refer to **two broad classes of networks** with a similar general structure, where one is finite impulse and the other is infinite impulse.
Both classes of networks exhibit temporal dynamic behavior. A finite impulse recurrent network is a directed acyclic graph that can be unrolled and replaced with a strictly feedforward neural network, while an infinite impulse recurrent network is a directed cyclic graph that can not be unrolled.

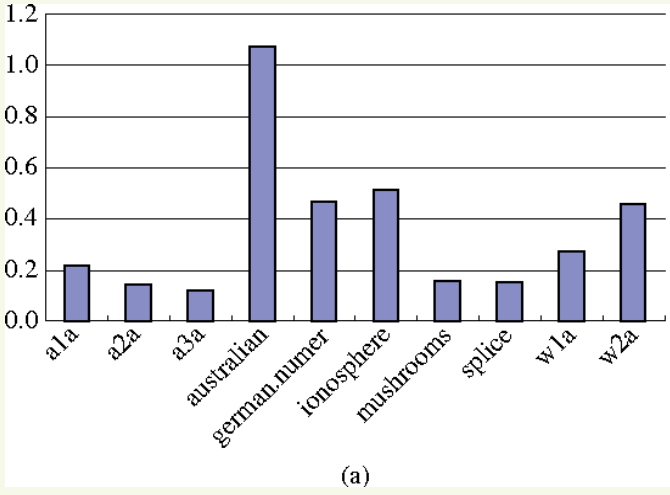
Both finite impulse and infinite impulse recurrent networks can have additional stored states, and the storage can be under direct control by the neural network.
The storage can also be replaced by another network or graph, if that incorporates time delays or has feedback loops. Such controlled states are referred to as gated state or gated memory, and are part of long short-term memory networks (LSTMs) and gated recurrent units. This is also called Feedback Neural Network.

The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

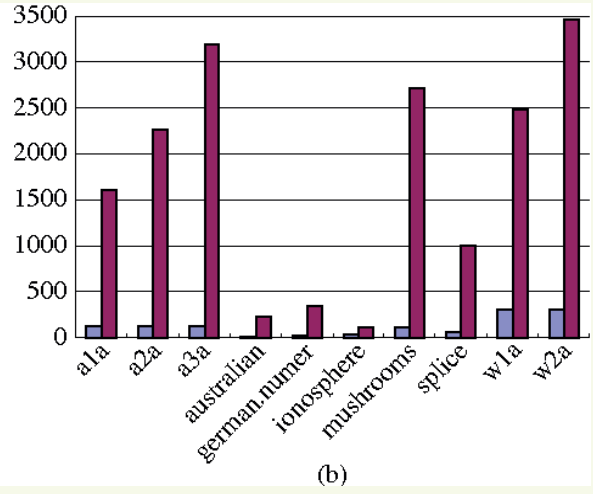
14. CLASSIFICATION
In statistics, **classification** is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known.
Examples are assigning a given email to the "spam" or "non-spam" class, and assigning a diagnosis to a given patient based on observed characteristics of the patient (sex, blood pressure, presence or absence of certain symptoms, etc.). Classification is an example of pattern recognition.

In the terminology of machine learning, classification is considered **an instance of supervised learning**, i.e., learning where a training set of correctly identified observations is available.
The corresponding unsupervised procedure is known as clustering, and involves grouping data into categories based on some measure of inherent similarity or distance.

An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. The term "classifier" sometimes also refers to the mathematical function, implemented by a classification algorithm, that maps input data to a category.



(a)



(b)

15. PREDICTION.

In statistics, prediction is a part of statistical inference. One particular approach to such inference is known as **predictive inference,** but the prediction can be undertaken within any of the several approaches to statistical inference.
Indeed, one possible description of statistics is that it provides a means of transferring knowledge about a sample of a population to the whole population, and to other related populations, which is not necessarily the same as prediction over time. **When information is transferred across time, often to specific points in time, the process is known as forecasting. Forecasting** usually requires time series methods, while prediction is often performed on cross-sectional data.

Statistical techniques used for prediction include regression analysis and its various sub-categories such as linear regression, generalized linear models (*logistic regression, Poisson regression, Probit regression*), etc. In case of forecasting, autoregressive moving average models and vector autoregression models can be utilized. When these and/or related, generalized set of regression or machine learning methods are deployed in commercial usage, the field is known as **predictive analytics.**

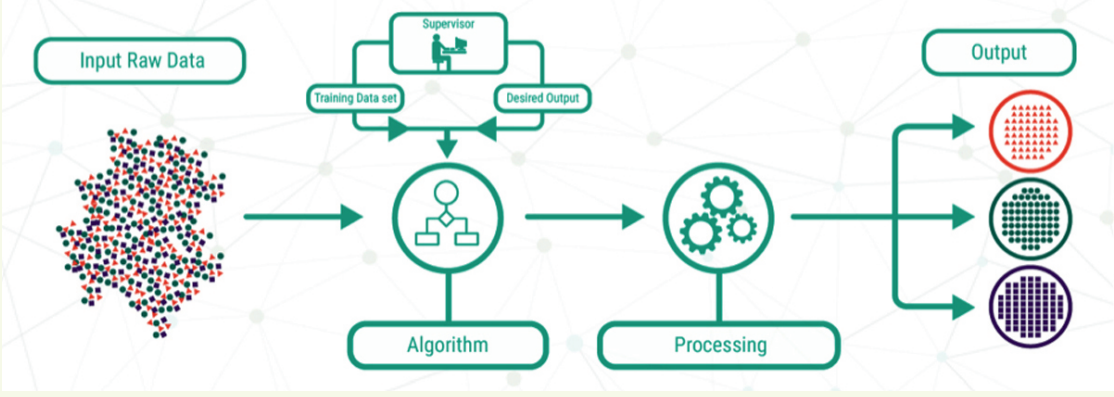
16. SUPERVISED LEARNING

Supervised learning is the machine learning task of learning a function that maps an input to an output based on example input-output pairs. It **infers a function from labeled training data** consisting of a set of training examples.In **supervised learning,** each example is a pair consisting of an input object (typically a vector) and a desired output value (also called the supervisory signal).
A **supervised learning algorithm** analyzes the training data and produces an inferred function, which can be used for mapping new examples.

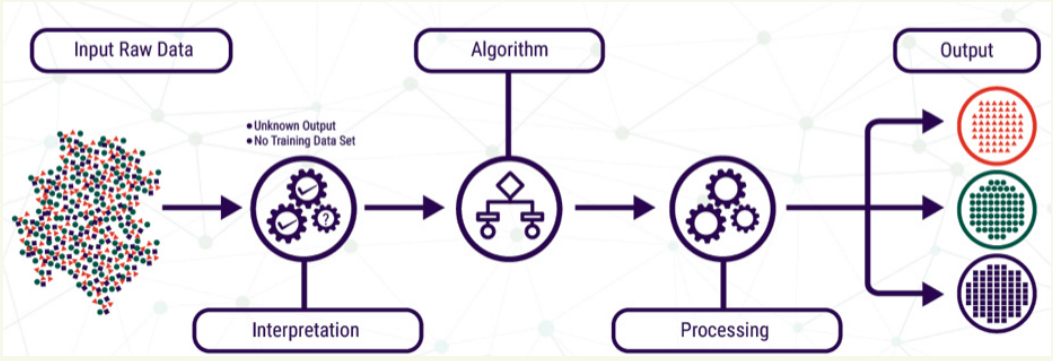
An optimal scenario will allow for the algorithm to correctly determine the class labels for unseen instances. This requires the learning algorithm to generalize from the training data to unseen situations in a "reasonable" way (see inductive bias). The parallel task in human and animal psychology is often referred to as concept learning.
Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output.

```
Y = f(X)
```

The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data. It is called supervised learning because the process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers, the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance.

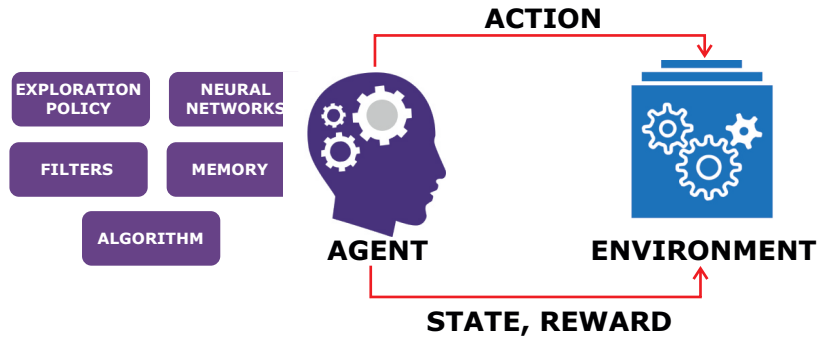


17. UNSUPERVISED LEARNING.
Unsupervised learning is a type of machine learning that looks for previously undetected patterns in a data set with no pre-existing labels and with a minimum of human supervision.



In contrast to supervised learning that usually makes use of human-labeled data, unsupervised learning, also known as self-organization allows for **modeling of probability densities over inputs.** It forms one of the three main categories of machine learning, along with supervised and reinforcement learning. Semi-supervised learning, a related variant, makes use of supervised and unsupervised techniques.

18. REINFORCEMENT LEARNING.
Reinforcement learning (RL) is an area of machine learning concerned with how software agents ought to take actions in an environment in order to maximize the notion of cumulative reward.
Reinforcement learning is one of **three basic machine learning paradigms,** alongside supervised learning and unsupervised learning.



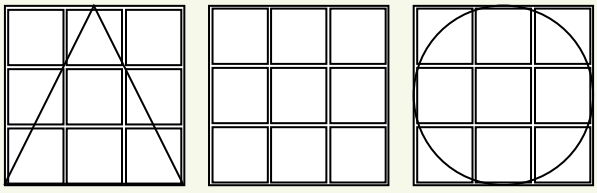
Reinforcement learning differs from supervised learning in **not needing labelled input/output pairs be presented,** and in **not needing sub-optimal actions to be explicitly corrected.**
Instead the focus is on finding a balance between exploration and exploitation.

https://medium.com/ai%C2%B3-theory-practice-business/reinforcement-learning-part-1-a-brief-introduction-a53a849771cf

In this example I want to show how to detect through a neural network the chosen figure is either a triangle a circle or a square.

Because we want to explain it all at the simplest level, we need to use just as little as possible components.
One of the basic rules which exist for solving problems at whatever level you are working, is cut your problem in to various elements, solve them separately and then go back to solve the whole problem.

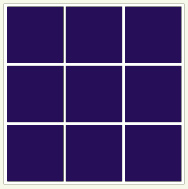
Because we are using the simplest shapes, it is a good idea to measure them. How to do that? Just give them a surface size and the principally break them down to numbers.
We could do that like a screen.
As humans we see images and of course we immediately have a vision of these shapes. So think of these shapes to be an image.
An image has a size so it might be reasonable to expect that as the Screen has as size measured in pixels, the image also will have one.
This gives us a reason to count the pixels.
Since we have a number, we can find the number of pixels per Square/Circle/Triangle.
Or maybe we even simply say if I have a square the rest of the images should fit in.
So if I would make a square like 9 pixels I must make the circle as big as the outer lines of the Square and then figure out how many pixel are in a circle.



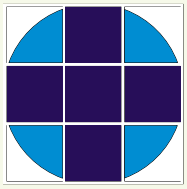
This is necessary so we can distinguish between the items without having to look at a drawn picture but get number which is easier to compute. Now if you look at the circle fitting into this square you can find out how many pixels will be used by the circle: there will be pixels that are only partially fitting, whole pixels and pixels with only very small overlap. This is important to define the differences between the three images.

For illustrational reasons I have used small squares as a pixel and call it boxes.
The actual size is of course not relevant.

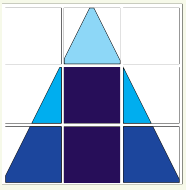
So now we have three values:
1. The square:
   9 boxes completely fitting



2. The circle :
   5 boxes completely fitting,
   4 boxes only partly 55% covered.



3. The Triangle: 4 pixels completely fitting,
   2 boxes 60 % covered,
   1 boxes 70% covered and
   2 boxes10% covered.

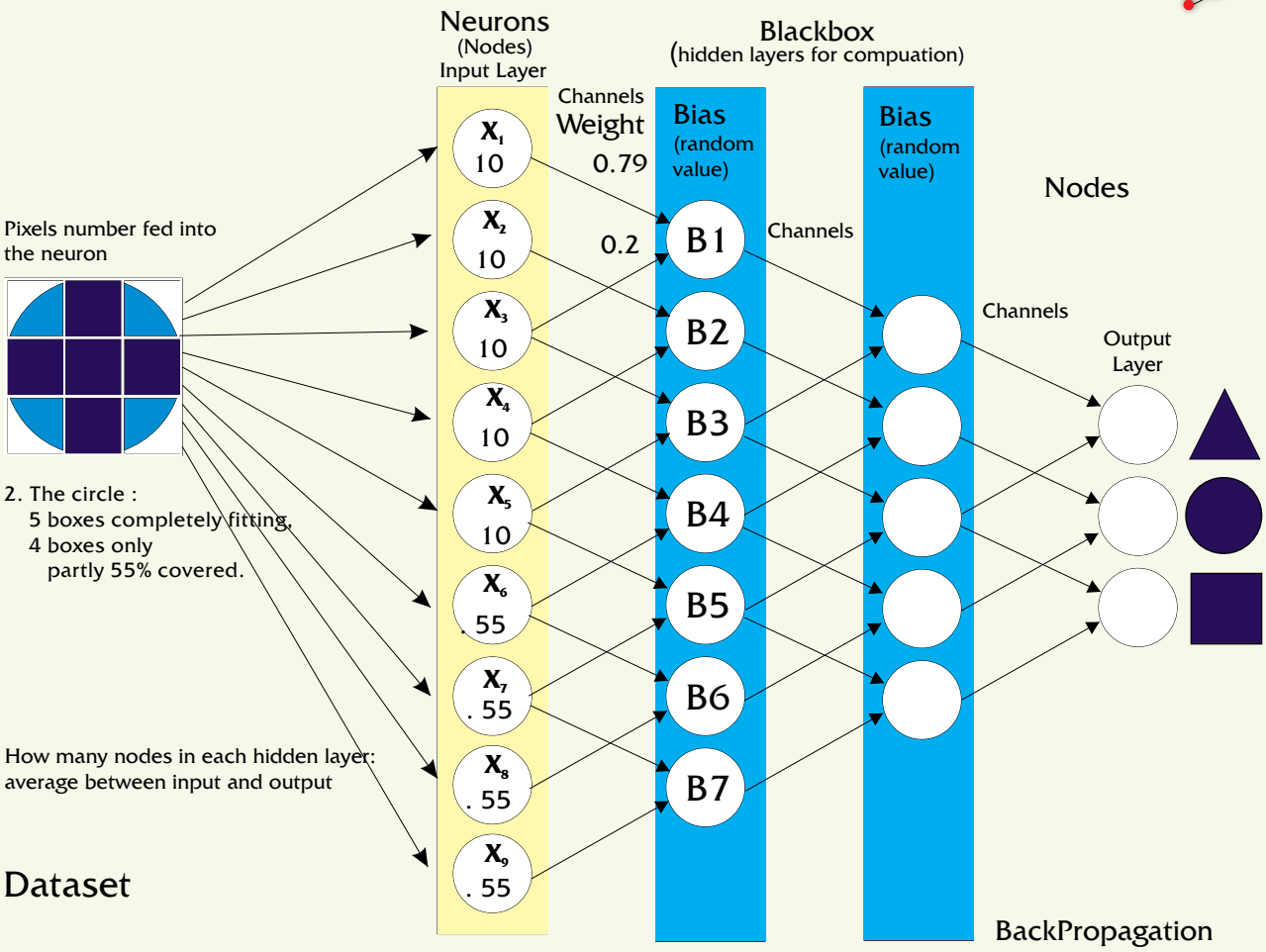


Here we have the basic number we could use for our computation.

In theory now already we could recognize the right sort of shape(Image).

Now to show what should be done by the steps in the model of the neural network we first must explain the various parts we need to use for each iteration.
(We need to do several iterations to get a better result on the outcome, lets say the predictability of the final answer).
There is one thing to say about Neural Networking: the result should always be as high as possible but it never will be 100%, but even 95 percent prediction can be a secure outcome.

Now we have started on creating the model for neural networking we need some further explanation.



So in the first row we are exposing all the values we found per item (Shape). We put these into the (Neurons) that are placed on an input layer. Now to let the computer decide what it can give as an outcome we need to create an extra factor which is added to the "weight". (Weight is the number we entered by the channels.) This extra factor is only needed to ad to the calculation for the final outcome. It is completely random and has in itself no value. It only makes sure that after a while the outcome is narrowed down to reality. Through out each iteration it will or can be an other number.

So here is the order of the performance of the program:

1. the program is started and the data will be inserted
2. after being gone through each channel it will put its computed numbers in to the output layer (the number of channels can vary per program)
3. This is only a first computation and it is absolutely not certain that this outcome is correct.

Therefore the same computation has to be done again, with different **weights** and **biasses** and that will be done again and again. The reason is simply that we are not looking for an exact outcome but for percentage which is as big as possible. If this grows to a very high percentage (95% - as an example) it is taken as a prediction with a high certainty of truth. In all these steps the program learns what might be the closest answer.

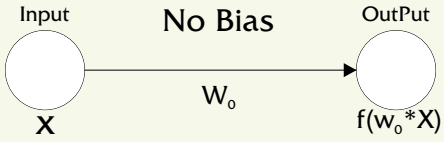
That is why iterations can exceed an outrageous number of iterations like (±)35.000 or even more if necessary.

The percentage will grow after each iteration until the percentage is high enough. These iterations include so called (for each step) activation and back propagation. So at each step the back propagation and activation has to take place and after one an other.

Here is an explanation of **Activation** and **Backpropagation.**
Of course there are many extras but that is not necessary to understand the beginning of the mode.

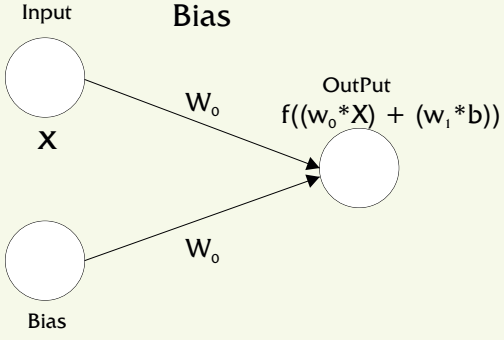
Activation function

The **activation function** in Neural Networks takes an input 'x' multiplied by a weight 'w'.
Bias allows you to shift the activation function by adding a constant (i.e. the given bias) to the input.
Bias in Neural Networks can be thought of as analogous to the role of a **constant** in a **linear function**, whereby the line is effectively transposed by the constant value.



In a scenario with **no bias**, the input to the activation function is 'x' multiplied by the connection weight 'w0'.

In a scenario **with bias,** the input to the activation function is 'x' times the connection weight 'w_0' plus the bias times the connection weight for the bias 'w1'.
This has the effect of shifting the activation function by a **constant amount** (b * w1).



Backpropagation,
short for *"backward propagation of errors,"* is an algorithm for supervised learning of artificial neural networks using gradient descent.
Given an artificial neural network and an error function, the method calculates the gradient of the error function with respect to the neural network's weights.

The "backwards" part of the name stems from the fact that calculation of the gradient proceeds backwards through the network, with the gradient of the final layer of weights being calculated first and the gradient of the first layer of weights being calculated last.

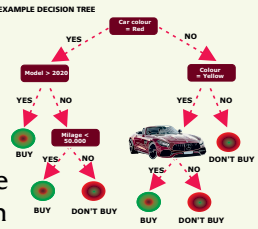
Partial computations of the gradient from one layer are reused in the computation of the gradient for the previous layer.
This backwards flow of the error information allows for efficient computation of the gradient at each layer versus the naive approach of calculating the gradient of each layer separately.

Backpropagation's popularity has experienced a recent resurgence given the widespread adoption of deep neural networks for image recognition and speech recognition.

CREATING A DECISION TREE

In the neural Network world we can do great jobs. But sometimes it is even better to start with a simple idea. We always have problems and usually we want to solve them. It can give a good insight in the sort of problem that you want to solve to create a decision tree for yourself.
To my surprise I found no program at all (for Pascal) that was capable of building it. Therefore I started to build something primitive which did the job. (It is for beginners and needs to be refined.) So I found that the technique that one would need for this, wasn't available as components or any other solution. Usually its schematic and needs some graphical items. I needed something that was showing a simple arrow, (a line + a triangle) that has to be drawn (the arrowhead), as well some elements where I could put on some information that was editable.

Since I did not find anything that could do this, this might be a nice task for the future. In that way one would be able to build a little schema which would be easy for explanatory reasons and demonstrating the structure of a problem. Nice to have.



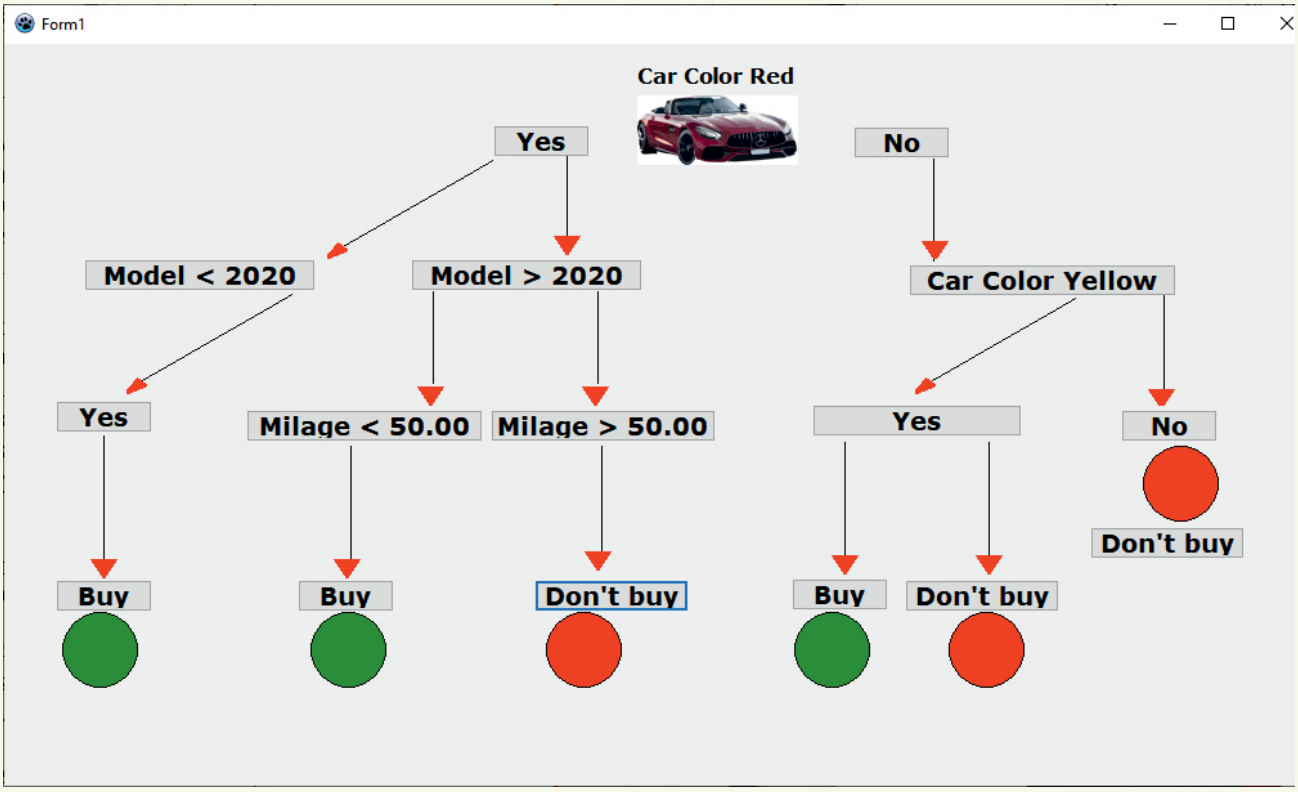
To give an impression of what this very simple program should be capable of, I build a little app which you can see here. The code for this small app is of course available as download.

Just a few words to spend on the construction: because anything had to be drawn on the fly, I was looking for the easiest components to use and found out that some obvious things like a shape was not going to help. It lacks a lot of convenience because a simple action like flood fill on it was not available.

The best thing to start drawing is using a PaintBox, just to make sure you will not loose a lot of time to find out. Because of using this PaintBox you should keep in mind that drawing arrow is not an easy thing: You will have to create it. The angle you want to set to the arrow is neither simple but has to be designed:

```
PBYellowDontArrow.Canvas.Polygon([Point(5,10)
,Point(15,25),Point(25,10)],false);
```

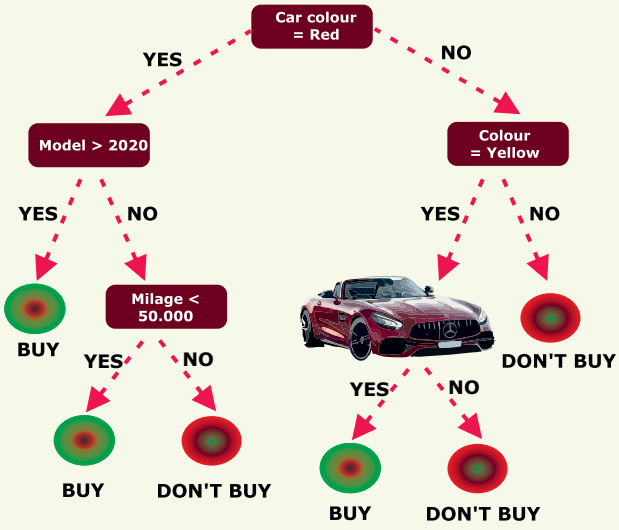
for the arrowhead,



and

```
PBYellowBuy.Canvas.Line(25,{beginpoint}0,25,200);
```

for the arrow itself.
Since we are using a square we have to compute these angles carefully before they are useful.
How simple it seems - it cost a lot of time.



maXbox Starter 76 Image Classifier

Author: Max Kleiner

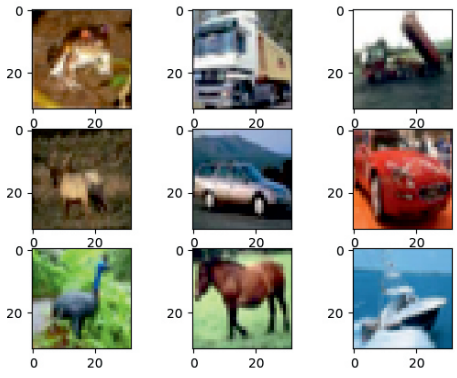
From Document to Real World Recognition

This tutor explains a trip to the kingdom of object recognition with computer vision knowledge and an image classifier from the CAI framework in Lazarus and Delphi, the so called CIFAR-10 Image Classifier.

CAI NEURAL API is a **Pascal based neural network API** optimized for AVX, AVX2 and AVX512 instruction sets plus OpenCL capable devices including AMD, Intel and NVIDIA for GPU capabilities.
This API has been tested under Windows and Linux.
On January this year we got in **Delphi support for OpenCL and Threads**.
This project and API is a sub-project from a bigger and older project called CAI and its sister to **Keras/TensorFlow** based K-CAI NEURAL API.

Image detection has been witnessing a rapid revolutionary change in some fields of computer vision.

Its involvement in the combination of object classification as well as object recognition makes it one of the most challenging topics in the domain of machine learning & vision.



How to Start:

First we need the library with modules.

Neural-API is a Pascal library built to empower developers to build and run applications and systems with **self-contained deep learning** and Computer Vision capabilities using a few lines of straight forward code.
You'll need a **Lazarus** or **Delphi** development environment. If you have an **OpenCL** capable device, you'll need its **OpenCL** GPU drivers.
But to use CAI first you need to install a few dependencies as units namely:

- dglOpenGL
- OpenCV as CL or OpenCL
- CL_Platform
- Neuralvolume, neuralnetwork, neuralab, etc.,

itself to install with **Lazarus** and with the help of **Git** so clone this project, add the neural folder to your Lazarus unit search path and you'll be almost ready to go!

```
https://github.com/joaopauloschuler
/neural-api
```

After installing CAI, you can find documentation in a readme.
Concerning **Delphi** a number of units do compile with Delphi and **you can create and run neural networks with Delphi** or in my case with the community edition 10.3 see picture in the left column. You'll be able to compile these units with Delphi: neuralvolume, neuralnetwork, neuralab, neuralabfun, neuralbit, neuralbyteprediction, neuralcache, neuraldatasets, neuralgeneric, neuralplanbuilder and neuralfit.
For Keras and tensorflow you get it also with git:

git clone
```
https://github.com/joaopauloschuler
/k-neural-api.git
```

Another fascinating way is to run the whole system on google **colab** or **colab.research container** with the help of a **Jupyter notebook*** running on Ubuntu in the cloud including the build of Free Pascal with Lazarus as I did and tested too:

https://github.com/maxkleiner/maXbox/blob/master/EKON24_SimpleImageClassificationCPU.ipynb

!apt-get install fpc fpc-source lazarus git subversion.
*(*The Jupyter Notebook is an open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text. Uses include: data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more.*
https://jupyter.org/)

Jupyter is a spin-off project from **IPython**, aiming to standardize interactive computing in any programming languages.
The kernel provides interactive environment that executes user code as a server, connected with a frontend through socket.
Those who need rapid prototyping or quick coding might be the best target users.
I don't see any reason why Pascal or Free Pascal should implement one, but it's up to those who want to implement, just like in other languages.

Now download the **CIFAR model file** (162 MB) that contains 5 volumes for the classification model that will be used for object training and detection:

https://www.cs.toronto.edu/~kriz/cifar-10-binary.tar.gz

```
if not os.path.isfile('cifar-10-batches-bin/data_batch_1.bin'):
print("Downloading CIFAR-10 Files")
url = 'https://www.cs.toronto.edu/~kriz/cifar-10-binary.tar.gz'
```

The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

The dataset is divided into five training batches (data_batch_1.bin -5) and one test batch, each with 10000 images.
The test batch contains exactly 1000 randomly-selected images from each class.

The training batches contain the remaining images in random order, but some training batches may contain more images from one class than another.
Between them, the training batches contain exactly 5000 images from each class.

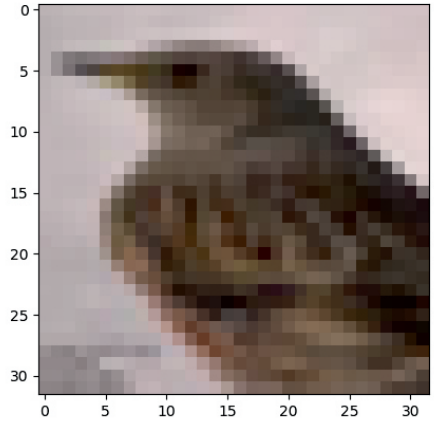
Then we need 3 necessary folders or files:

```
- neural\
- neural\data_batch_1.bin -
                    data_batch_5.bin
- neural\test_batch.bin
```

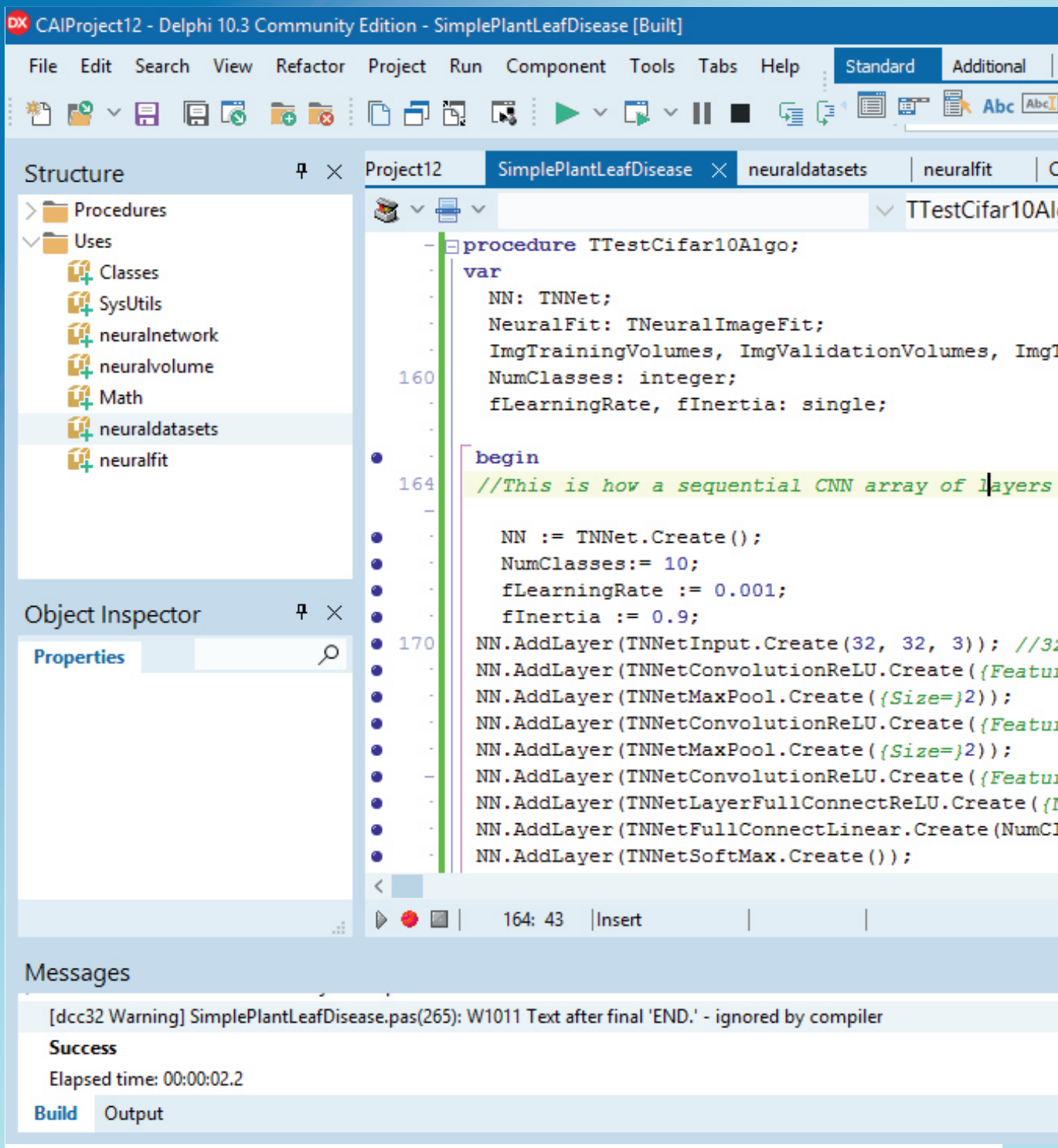
The classes are completely mutually exclusive. There is no overlap between automobiles and trucks or cat and dog. The classes are:

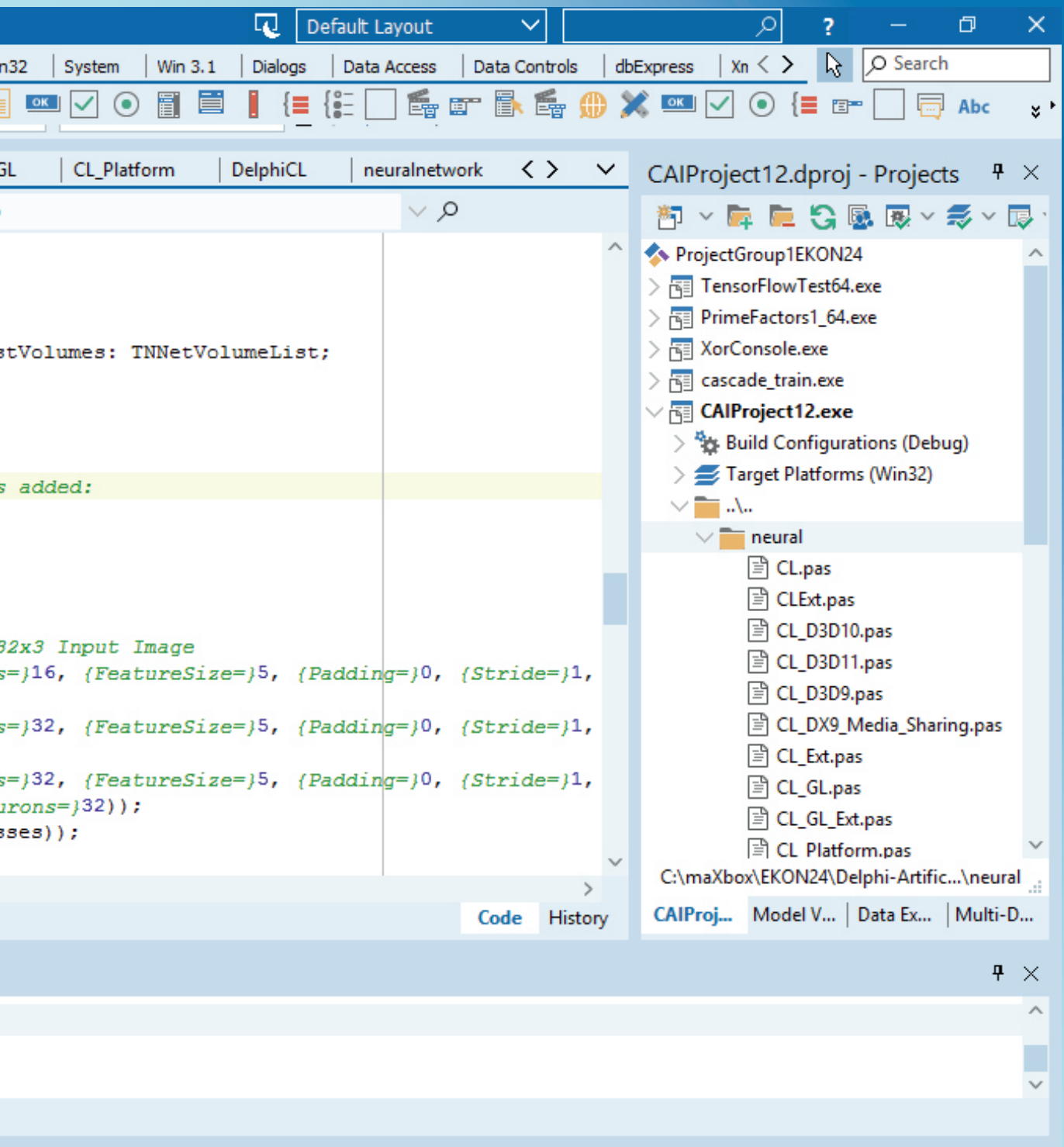
```
classes=('plane','car','bird','cat',
    'deer','dog','frog','horse','ship','truck')
```

These results were obtained with a convolutional neural network. Briefly, they are 18% test error *without* data augmentation and 11% *including* data.
A single image has a low resolution for training the data features:



Open now your preferred code editor for writing Pascal code (in my case maXbox and Delphi Community Edition) and load the file examples/SimpleImageClassifier/SimpleImageClassifier.lpi or some valid file name like in my case SimpleImageClassifier_CPU_Cifar.pas


CAIProject12 - Delphi 10.3 Community Edition - SimplePlantLeafDisease [Built]
File Edit Search View Refactor Project Run Component Tools Tabs Help
Standard Additional
Structure
Procedures
Uses
Classes
SysUtils
neuralnetwork
neuralvolume
Math
neuraldatasets
neuralfit
Object Inspector
Properties
Project12
SimplePlantLeafDisease
neuraldatasets
neuralfit
TTestCifar10Al
procedure TTestCifar10Algo;
var
NN: TNNet;
NeuralFit: TNeuralImageFit;
ImgTrainingVolumes, ImgValidationVolumes, ImgT
NumClasses: integer;
fLearningRate, fInertia: single;
begin
//This is how a sequential CNN array of layers
NN := TNNet.Create();
NumClasses:= 10;
fLearningRate := 0.001;
fInertia := 0.9;
NN.AddLayer(TNNetInput.Create(32, 32, 3)); //32
NN.AddLayer(TNNetConvolutionReLU.Create({Featur
NN.AddLayer(TNNetMaxPool.Create({Size=}2));
NN.AddLayer(TNNetConvolutionReLU.Create({Featur
NN.AddLayer(TNNetMaxPool.Create({Size=}2));
NN.AddLayer(TNNetConvolutionReLU.Create({Featur
NN.AddLayer(TNNetLayerFullConnectReLU.Create({N
NN.AddLayer(TNNetFullConnectLinear.Create(NumCl
NN.AddLayer(TNNetSoftMax.Create());
164: 43
Insert
Messages
[dcc32 Warning] SimplePlantLeafDisease.pas(265): W1011 Text after final 'END.' - ignored by compiler
Success
Elapsed time: 00:00:02.2
Build
Output



This example has interesting aspects to look at: Its source code is very small and Layers are added sequentially. Then the Training hyper-parameters are defined before calling the fit method. You need fit() to train the model! In line 155 we start with the test-class from the CAI library and we create the neural net layers too:

```
procedure TTestCifar10Algo;
var
 NN: TNNet;
 NeuralFit: TNeuralImageFit;
 ImgTrainingVolumes, ImgValidationVolumes, ImgTestVolumes: TNNetVolumeList;
 NumClasses: integer;
 fLearningRate, fInertia: single;

begin
 //This is how a sequential CNN array of layers is added:

  NN := TNNet.Create();
  NumClasses:= 10;
  fLearningRate := 0.001;
  fInertia := 0.9;
 NN.AddLayer(TNNetInput.Create(32, 32, 3)); //32x32x3 Input Image
 NN.AddLayer(TNNetConvolutionReLU.Create({Features=}16, {FeatureSize=}5, {Padding=}0, {Stride=}1, {SuppressBias=}0));
 NN.AddLayer(TNNetMaxPool.Create({Size=}2));
 NN.AddLayer(TNNetConvolutionReLU.Create({Features=}32, {FeatureSize=}5, {Padding=}0, {Stride=}1, {SuppressBias=}0));
 NN.AddLayer(TNNetMaxPool.Create({Size=}2));
 NN.AddLayer(TNNetConvolutionReLU.Create({Features=}32, {FeatureSize=}5, {Padding=}0, {Stride=}1, {SuppressBias=}0));
 NN.AddLayer(TNNetLayerFullConnectReLU.Create({Neurons=}32));
 NN.AddLayer(TNNetFullConnectLinear.Create(NumClasses));
 NN.AddLayer(TNNetSoftMax.Create());
 writeln(NN.SaveDataToString);
 //readln;
```

Then we load the data volumes for training. There is a trick that you can do with this API or any other API when working with image classification: you can increase the input image size. As per the following example (train, test and validate), by increasing CIFAR-10 input image sizes from 32x32 to 48x48, you can gain up to 2% in classification accuracy.

```
CreateCifar10Volumes(ImgTrainingVolumes,
ImgValidationVolumes, ImgTestVolumes);

 WriteLn
   (
    'Training Images:', ImgTrainingVolumes.Count,
    ' Validation Images:', ImgValidationVolumes.Count,
    ' Test Images:', ImgTestVolumes.Count
   ); //*)

WriteLn('Neural Network will minimize error with:');
WriteLn(' Layers: ', NN.CountLayers());
WriteLn(' Neurons: ', NN.CountNeurons());
WriteLn(' Weights: ', NN.CountWeights());
writeln('Start Convolution Net...');
readln;
```

As an output on the shell we see the Neural Network will minimize error with:

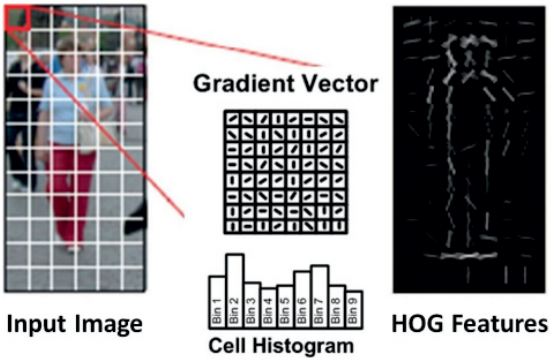
```
Layers: 9
Neurons: 122
Weights: 40944
Start Convolution Net...
```

Later on, this is how the training/fitting method is called:

```
 NeuralFit := TNeuralImageFit.Create;
//readln;
NeuralFit.FileNameBase :=
'EKONSimpleImageClassifier2';
NeuralFit.InitialLearningRate := fLearningRate;
NeuralFit.Inertia := fInertia;
NeuralFit.LearningRateDecay := 0.005;
  NeuralFit.StaircaseEpochs := 17;
 //  NeuralFit.Inertia := 0.9;
  NeuralFit.L2Decay := 0.00001;

//readln;  best fit: batch 128 epochs 100
// just for test and evaluate the process - epochs = 1,
          otherwise 10 or 100!

NeuralFit.Fit(NN, ImgTrainingVolumes,
ImgValidationVolumes, ImgTestVolumes, NumClasses,
                  {batchsize}128, {epochs}1);
  writeln('End Convolution Net...');
  readln;
  NeuralFit.Free;

  NN.Free;
  ImgTestVolumes.Free;
  ImgValidationVolumes.Free;
  ImgTrainingVolumes.Free;
end;
```

The Learning rate in `NeuralFit.InitialLearningRate := fLearningRate;` is an important hyperparameter that controls how much we adjust the weights in the network according to the gradient.

Like in a **Histogram of Oriented Gradients** (HOG) is basically a feature descriptor that is used to detect objects in image processing and other computer vision techniques. The Histogram of oriented gradients descriptor technique includes occurrences of gradient orientation in localised portions of an image, such as detection window, region of interest (ROI), among others. Advantage of HOG-like features is their simplicity, and it is easier to understand information they carry.



In the last step we see an image recognition that specializes in people or humans. Surprisingly, from a well known picture in Delft (The Night Watch), the people in the background are more likely to be recognized than the real person.
That has to do with the image section in the sense of focus of the rectangle.

```
person : 11.248066276311874
person : 14.372693002223969
person : 19.247493147850037
person : 34.878602623939514
person : 77.2484838962555
image detector compute ends…
```

#loads model from path specified above using the `setModelPath()` class method.
`detector.loadModel()`

To detect only some of the objects above, I will need to call a CustomObjects method and set the name of the object(s) we want to detect to through. The rest are False by default. In our example, we detect customized only person, laptop, cat and dogs. I will explain that in one of the next magazine.

So we get the whole validation for demo purpose in a total time about only 12 minutes, but with a weak accuracy.
But the real validation has a longer duration or must have to proof the concept (best fit: batch 128 - epochs 100).

Starting Testing.

```
Epochs: 50
Examples seen: 2000000
Test Accuracy: 0.8383
Test Error: 0.4463 Test Loss: 0.4969
Total time: 162.32min
Epoch time: 2.7 minutes. 100 epochs:
4.5 hours.
Epochs: 50. Working time: 2.71
hours.
Finished.
```

Starting Validation Demo
- just to make it shorter.
VALIDATION RECORD!

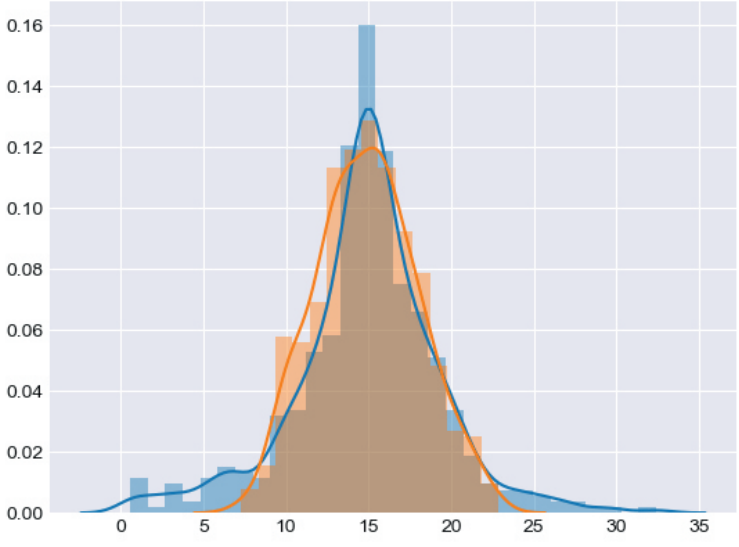
```
Saving NN at
EKONSimpleImageClassifier.nn
Epochs: 1 Examples seen:40000
Validation Accuracy: 0.4608
Validation Error: 1.38
55 Validation Loss: 1.4801
Total time:  11.39 min
Epoch time: 8.6 minutes.
100 epochs: 14 hours.
Epochs: 1.
Working time: 0.19 hours.
Finished.
```

Conclusion:
So how can we shorten the validation time?
With a pretrained model!
Pretrained models are a wonderful source of help for people looking to learn an algorithm or try out an existing framework online or offline.

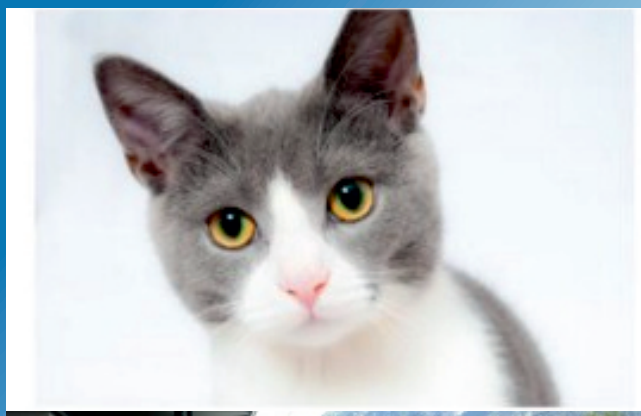
But the predictions made using pretrained models would not be so effective as you train the model with your data, that's the trade-off. Due to time restrictions or computational restraints, it's not always possible to build a model from scratch (like we did) which is why pretrained models exist like the pretrained-yolov3.h5!

The script and data can be found:

```
http://www.softwareschule.ch/example
s/detector2.htm

https://sourceforge.net/projects/
maxbox/files/Examples/EKON/EKON24/
ImageDetector/

https://sourceforge.net/projects/
maxbox/files/Examples/EKON/EKON24/
SimpleImageClassifier_CPU_Cifar2.pas
/download

http://www.softwareschule.ch/box.htm
https://maxbox4.wordpress.com
```



A human observation of a cat:



A computer registration of a cat:

```
[[[ 1  29  70 114  76   0   8   4   5   5   0 111 162   9   8  62  62]
  [ 0  33  61 102 106  34   0   0   0   0  49 182 150   1  12  65  62]
  [ 0  40  54 123  90  72  77  52  51  49 121 205  98   0  15  67  59]
  [ 1  41  57  74  54  96 181 220 170  90 149 208  56   0  16  69  59]
  [ 1  32  36  47  81  85  90 176 206 140 171 186  22   3  15  72  63]
  [ 1  31  39  66  71  71  97 147 214 203 190 198  22   6  17  73  65]
  [ 3  15  30  52  57  68 123 161 197 207 200 179   8   8  18  73  66]
  [ 2  17  37  34  40  78 103 148 187 205 225 165   1   8  19  76  68]
  [ 3  20  44  37  34  35  26  78 156 214 145 200  38   2  21  78  69]
  [ 2  20  34  21  43  70  21  43 139 205  93 211  70   0  23  78  72]
  [ 4  16  24  14  21 102 175 120 130 226 212 236  75   0  25  78  72]
  [ 5  13  21  28  28  97 216 184  90 196 255 255  84   4  24  79  74]
  [ 5  15  25  30  39  63 105 140  66 113 252 251  74   4  28  79  75]
  [ 5  16  32  38  57  69  85  93 120 128 251 255 154  19  26  80  76]
  [ 5  20  42  55  62  66  76  86 104 148 242 254 241  83  26  80  77]
  [ 3  20  38  55  64  69  80  78 109 195 247 252 255 172  40  78  77]
  [ 8  23  34  44  64  88 104 119 173 234 247 253 254 227  66  74  74]
  [ 6  24  37  45  63  85 114 154 196 226 245 251 252 250 112  66  71]
```

Later versions of kbmMW contain more and more nice to have features for general logging, auditing, runtime exception handling with stack-trace and now also memory usage debugging.
These features are actually available for any application, even ones not using other parts of kbmMW.
I've already been writing some articles about the logging and auditing system in kbmMW which also covered exception handling with stack-trace, but latest addition is the ability to real time trace each and every memory allocation done by your application.

WHY USE KBMMW'S MEMORY DEBUGGER, when for example FastMM have leak detection build in?

- FastMM only tracks memory allocations done via the regular GetMem etc memory allocations.
- It does not track allocations made via any of the Virtual/Heap/Global/Local allocation methods available in Windows.
- Further kbmMW's debugger still works even if FastMMs leak detection is disabled, and provide features for logging memory use and allocations at any time in your application.

STARTING OUT
You will need to add **kbmMWDebugMemory** to the uses clause of your application, and you will have to make sure that the following defines are set in your `kbmMWConfig.inc` file:

```
{$DEFINE KBMMW_SUPPORT_DEBUGMEMORY}
{$DEFINE KBMMW_INSTALL_DEBUGMEMORY_HANDLERS}
```

Otherwise all memory debugging will be disabled.

If **KBMMW_SUPPORT_DEBUGMEMORY** is omitted, then no memory debugging functionality (including all functions/methods) are available.

If **KBMMW_INSTALL_DEBUGMEMORY_HANDLERS** is omitted, then the memory debugging system will not automatically install the hooks and handlers, and thus the functions may be available (see above), but no memory tracing is happening.

When both defines are set, adding `kbmMWDebugMemory` to your uses clause of your application or unit, will automatically install kbmMW's memory debugging features and hooks.

To get the best out of it, you should also make sure your application is build with debug, stack-trace and an external TDS file alternatively a detailed `MAP` file.
The produced `*.tds` or `*.map` file must be in the same directory as your executable when you run it if you want to do memory debugging outside the IDE.

CONCEPT
kbmMW automatically hooks Borland type memory allocations, Microsoft Windows Virtualxxx, Globalxxx, Localxxx and Heapxxx allocation methods. It plays nicely with any third dparty memory manager including **FastMM**.

Each allocation and reallocation made, is assigned a unique incrementing 64 bit number by **kbmMW**.
This number can be used for tracking all allocations made between two points in time, called **checkpoints**.

Basically a checkpoint is just an 64 bit number.
This way you can check exactly what allocations have happened in subsets of your code, and even get a stack-trace to where the allocations took place.

STATISTICS

kbmMW maintains statistics over allocated memory and allocation counts.

These can be obtained at any time like this:

```
lLiveAllocationsCount.Caption:=inttostr(TkbmMWDebugMemory.LiveAllocationCount)
 +' ('+inttostr(TkbmMWDebugMemory.LiveAllocationCountPerSec)+'/sec)';

lLiveAllocSize.Caption:=inttostr(TkbmMWDebugMemory.LiveAllocationSize)
 +' ('+inttostr(TkbmMWDebugMemory.LiveAllocationSizePerSec)+'/sec)';

lMaxAllocationCount.Caption:=inttostr(TkbmMWDebugMemory.MaxAllocationCount);

lMaxAllocationSize.Caption:=inttostr(TkbmMWDebugMemory.MaxAllocationSize);

lMaxCapacity.Caption:=inttostr(TkbmMWDebugMemory.CurrentAllocationCountCapacity);
```

`LiveAllocationCount` is the number of active, in use, memory allocations detected.

It counts allocations of all types (Borland – object/string/memory, Local memory, Global memory, Virtual memory, Heap memory).
Since for example **FastMM** will allocate large chunks of memory via typically `VirtualAlloc`, and hand out pieces of this memory to applications using `GetMem` (Borlands memory manager interface), you will see an imprecise count (and size), since the count will include both the VirtualAlloc made by **FastMM,** and the individual **GetMem** (etc) calls made by the application.

Thus the live values are comparable values, but not exact values. You can depend on them to show for example increasing use of memory (perhaps indicating a leak), but you cant depend directly on the exact absolute value, as some allocations are counted twice due to the above situation.

DETECTING LEAKS AT SHUTDOWN

What is a leak? It's a resource that has been allocated, but which is never deallocated.

Some leaks are bad, some are not. The ones that are not bad, are leaks that are happening due to single, non repetitive allocations, typically made during startup of an application, which are never explicitly freed. In fact the RTL and VCL contains numerous such leaks.

Those leaks are not bad, because the operating system (Windows in this case) will automatically release all memory used by an application the moment the application shuts down.

The bad leaks are those that repeatedly allocates more memory, but never deallocates it again. These leaks will eventually make the application run out of memory space and/or make the system go extremely slow due to exhaust of physical RAM which results in paging.
Paging is where currently less important memory segments are written to disk, to make room for currently more important memory segments, that are currently being allocated, or being read in from disk (page file).

It is normal for some paging happening on a system, but it is not normal if an application allocates as much memory as to slow all other processes significantly down, just due to paging happening.

A bad leak has occurred (repeatedly).

The only time to reasonably reliable to detect if those bad leaks has occurred, is at application shutdown time. Why? Because at that time you know that all (most) allocated memory should have been released by your applications destructors / FormClose events etc.

kbmMW makes this simple to check for. Some place, very early in your application's startup cycle, put these lines:

```
TkbmMWDebugMemory.ReportDestination('c:\temp\leaks.txt');

TkbmMWDebugMemory.ReportLeaksOnShutdown:=true;
TkbmMWDebugMemory.StartLeakChecking;
```


Now you have defined where leak reports should be dumped, that you want an automatic leak report to be created on shutdown, and that you want leak detection to start immediately. In fact all that StartLeakChecking does, is to load any TDS/MAP debug information (for stack trace purpose) and then register a baseline checkpoint from which it will check that allocations has been freed. All allocations before this time will be ignored and thus all build in VCL/RTL leaks and all objects allocated for the TDS/MAP info.

You can actually see the baseline value by checking:

```
ShowMessage('Baseline='+inttostr(TkbmMWDebugMemory.Baseline));
```

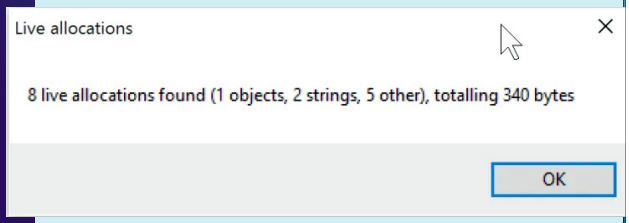
If you want the leak detection to include everything since the memory allocation hooks was installed. Set baseline to zero. Eg.

```
TkbmMWDebugMemory.Baseline:=0;
```

The checkpoint for last allocation made, can be obtained via:

```
var
 cp:TkbmMWDebugMemoryAllocationKey;
...
  cp:=TkbmMWDebugMemory.Checkpoint;
  ShowMessage('Checkpoint='+inttostr(key));
```

Now when you run your application and then closes it again, kbmMW will produce a report of non freed allocations.
Default it will present an overview status on screen like this:



And output details to the designated file on the next page(35):
As you can see, the debugger is able to determine if its objects, strings or other types of memory that has been leaked, and gives you in the first section, statistics over how many instances of a particular class are leaked. In this case just one instance of a TkbmMWInnerThread.

This one is a safe leak, that just exists because the kbmMW scheduler have a relaxed event thread running handling events. The memory debugger has registered a scheduled event for calculating allocations/sec.

The stack-trace may be more or less precise, depending on the amount of debug information compiled into your application (stack frames must be enabled), and depending on if you let Delphi generate an external TDS or MAP file which kbmMW's stack trace functionality can use.

You can choose not to collect stack-trace for allocations. This will save some memory and CPU time, and will make your report less verbose. This can be done by adding the following line before the StartLeakChecking call:

```
TkbmMWDebugMemory.CollectStacks:=false;
```

Tracing allocations in specific situations
You may want to report all allocations made within a specific time interval, or between two locations in your code. You do this by using the checkpoint method to obtain a number at the two locations and then select what you want in your report. For example:

```
var
 FMyCP1,FMyCP2: TkbmMWDebugMemoryAllocationKey;
...
  FMyCP1:=TkbmMWDebugMemory.Checkpoint;
  <your interesting code>
  FMYCP2:= TkbmMWDebugMemory.Checkpoint;
  MyReport(FMyCP1,FMyCP2);

...

procedure MyReport(ACP1,ACP2:
                          TkbmMWDebugMemoryAllocationKey);
var sr:TkbmMWDebugMemoryScanResult;
begin
  sr:=TkbmMWDebugMemoryScanResult.Create;
  try
    TkbmMWDebugMemory.Scan(sr,FMyCP1,FMyCP2,[mwdmstObject]);
    sr.Log(mwltDebug,mwllDetailed,[mwsrpoNoStack]);
  finally
    sr.Free;
  end;
end;
```

The above example will only report object instance allocations, and will report them via the kbmMW log system as debug log, without any stack-trace.

```
Class:TkbmMWInnerThread Count:1 TotalSize:84
278956) Object:<unnamed> Class:TkbmMWInnerThread Addr:02F4C498 Size:84 hModule=00400000
<406A8A> : System(System.pas) : Line 4570 : @GetMem$qqri
<40791B> : System(System.pas) : Line 15975 : TObject.NewInstance$qqrv
<408126> : System(System.pas) : Line 17293 : @ClassCreate$qqrpvzc
<6085F6> : kbmMWGlobal(kbmMWGlobal.pas) : Line 8833 : TkbmMWCustomThreadPool.GetIdleCount$qq
<60899E> : kbmMWGlobal(kbmMWGlobal.pas) : Line 9199 : TkbmMWLockFreeHashArray32.Increment$qq
<608E71> : kbmMWGlobal(kbmMWGlobal.pas) : Line 9346 : TkbmMWLockFreeHashArray64.Increment$qq
<68E938> : kbmMWDebugMemory(kbmMWGlobal.pas) : Line 9176 : %TkbmMWLockFreeHashArray__1$44Kbm
<68E972> : kbmMWDebugMemory(kbmMWGlobal.pas) : Line 9179 : %TkbmMWLockFreeHashArray__1$44Kbm

278958) Unicode string:TkbmMWScheduledRelaxedEventThread(kbmMWSystemScheduler) RefCount:2 Ad
<4097D3> : System(System.pas) : Line 24231 : @NewUnicodeString$qqri
<40A71A> : System(System.pas) : Line 29853 : @UStrCatN$qqrv
<68C4BD> : kbmMWDebugMemory(kbmMWDebugMemory.pas) : Line 1243 : TkbmMWDebugMemory.Scan$qqrxp
<608736> : kbmMWGlobal(kbmMWGlobal.pas) : Line 8859 : TkbmMWCustomThreadPool.Reserve$qqrxo
<4BB92C> : System.Classes(System.Classes.pas) : Line 12585 :
<4BB98C> : System.Classes(System.Classes.pas) : Line 12585 :
<40971E> : System(System.pas) : Line 24006 : ThreadWrapper$qqspv
Address <0> unknown

279038) Unicode string:I RefCount:1 Addr:06AD0230 Size:16 hModule=00400000
<409C60> : System(System.pas) : Line 25201 : @UStrAsg$qqrr20System.UnicodeStringx20System.Un
<5F7549> : kbmMWDateTime(kbmMWDateTime.pas) : Line 2820 : TkbmMWDateTime.TrySetRFC1123DateTi
<5F9027> : kbmMWDateTime(kbmMWDateTime.pas) : Line 3654 : TkbmMWDateTime.$o20System.UnicodeS
<68D143> : kbmMWDebugMemory(kbmMWDebugMemory.pas) : Line 1495 : kbmMWDebugMemoryGlobalFree$q
<68D198> : kbmMWDebugMemory(kbmMWDebugMemory.pas) : Line 1496 : kbmMWDebugMemoryGlobalFree$q
<68D306> : kbmMWDebugMemory(kbmMWDebugMemory.pas) : Line 1520 : kbmMWDebugMemoryGlobalReAllo
<608D7D> : kbmMWGlobal(kbmMWGlobal.pas) : Line 9299 : TkbmMWLockFreeHashArray32.Decrement$qq
<68C5A6> : kbmMWDebugMemory(kbmMWDebugMemory.pas) : Line 1270 : TkbmMWDebugMemory.Scan$qqrxp

278927) Data(BORLAND) Addr:02F36A38 Size:64 hModule=00400000
<408770> : System(System.pas) : Line 17807 : TMonitor.GetMonitor$qqrxp14System.TObject
<4BB721> : System.Classes(System.Classes.pas) : Line 12585 :
<5CC939> : Vcl.Forms(Vcl.Forms.pas) : Line 10181 : Forms.TApplication.SetTitle$qqrx20System.
<5CC9BB> : Vcl.Forms(Vcl.Forms.pas) : Line 10204 : Forms.TApplication.SetHandle$qqrp6HWND__
<5CC9E9> : Vcl.Forms(Vcl.Forms.pas) : Line 10216 : Forms.TApplication.IsDlgMsg$qqrr6tagMSG
<5CBBF4> : Vcl.Forms(Vcl.Forms.pas) : Line 9768 : Forms.TApplication.CheckIniChange$qqrr24Wi
<5CBF1A> : Vcl.Forms(Vcl.Forms.pas) : Line 9833 : Forms.TApplication.WndProc$qqrr24Winapi.Me
<5CBF5D> : Vcl.Forms(Vcl.Forms.pas) : Line 9842 : Forms.TApplication.WndProc$qqrr24Winapi.Me

278835) Data(OS_LOCAL) Addr:01334398 Size:8 hModule=00400000
<53302E> : Vcl.Controls(Vcl.Controls.pas) : Line 15003 : Controls.TDockTree.PaintSite$qqrp5H
<52AD56> : Vcl.Controls(Vcl.Controls.pas) : Line 9822 : Controls.TWinControl.ControlAtPos$qq
<52AD6C> : Vcl.Controls(Vcl.Controls.pas) : Line 9825 : Controls.TWinControl.ControlAtPos$qq
<52AB31> : Vcl.Controls(Vcl.Controls.pas) : Line 9761 : Controls.TWinControl.SetParentWindow
<524F04> : Vcl.Controls(Vcl.Controls.pas) : Line 6004 : Controls.TControl.SetName$qqrx20Syst
<529A46> : Vcl.Controls(Vcl.Controls.pas) : Line 9036 : Controls.TWinControl.FlipChildren$qq
<52CA0C> : Vcl.Controls(Vcl.Controls.pas) : Line 10838 : Controls.TWinControl.DockDrop$qqrp2
<5C1B70> : Vcl.Forms(Vcl.Forms.pas) : Line 3247 : Forms.TScrollingWinControl.IsTouchProperty

278955) Data(BORLAND) Addr:02F21110 Size:12 hModule=00400000
<4404DB> : System.Generics.Collections(System.Generics.Collections.pas) : Line 2632 : Generi
<43F454> : System.Generics.Collections(System.Generics.Collections.pas) : Line 1815 : Generi
<43F479> : System.Generics.Collections(System.Generics.Collections.pas) : Line 1826 : Generi
<43E3FF> : System.Generics.Defaults(System.Generics.Defaults.pas) : Line 1513 :
<609718> : kbmMWGlobal(kbmMWGlobal.pas) : Line 9629 : TkbmMWLock.BeginWrite$qqrv
<6097C8> : kbmMWGlobal(kbmMWGlobal.pas) : Line 9664 : TkbmMWLock.BeginWrite$qqrv
```

C++Builder 10.4 SydneyProfessional

€ 1.087,- ~~€ 1.699,-~~

C++Builer 10.4 Sydney Enterprise

€ 2.559,- ~~€ 3.999,-~~

C++Builder 10.4 Sydney Architect

€ 4.159,- ~~€ 6.499,-~~

RAD Studio 10.4 SydneyProfessional

€ 1.919,- ~~€ 2.999,-~~

RAD Studio 10.4 Sydney Enterprise

€ 3.199,- ~~€ 4.999,-~~

RAD Studio 10.4 Sydney Architect

€ 4.479,- ~~€ 6.999,-~~

Any questions? Contact us directly.

With the RAD Studio 10.4 tools you can build the most beautiful applications and apps! Order your license directly online, request a quote or speak to a migration specialist via telephone +31 23 542 22 27 or +33 972 19 28 87

Barnsten BV - Embarcadero Distributor for Benelux, France and Africa
Zijlstraat 47, 2011 TK Haarlem, The Netherlands
https://www.barnsten.com/promotions/

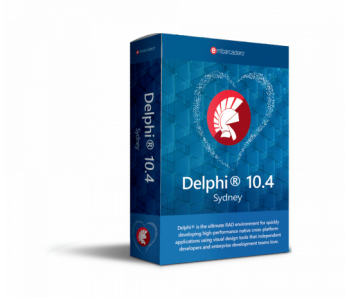
barnsten

The UPGRADE possibility is back in September

More than 35% discount on all 10.4 Sydney products + FREE Component Pack with purchase of Enterprise & Architect versions!

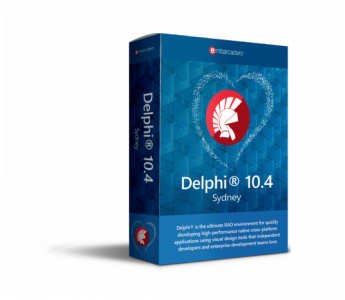
Take advantage of **the limited time** to purchase Delphi, C ++ Builder and RAD Studio for the old upgrade price! This means a discount of at least 35%. When purchasing an Enterprise or Architect version you will also receive a unique Component Pack worth € 10,000. This pack contains products from e.g. InstallAware, TMS, Steema and Woll2woll.

All offers are valid until September 25, 2020.



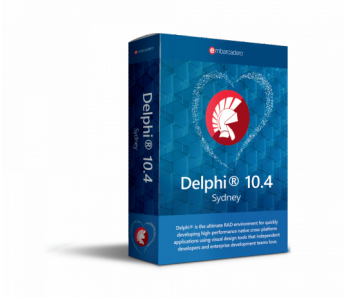
Delphi 10.4 Sydney Professional

€ 1.087,- ~~€ 1.699,-~~



Delphi 10.4 Sydney Enterprise

€ 2.559,- ~~€ 3.999,-~~



Delphi 10.4 Sydney Architect

€ 4.159,- ~~€ 6.499,-~~

In telecommunications, 5G is the fifth generation technology standard for cellular networks.
Cellular phone companies began deploying 5G worldwide in 2019, the planned successor to the 4G networks which provide connectivity to most current cellphones.
5G networks the service area is divided into small geographical areas called cells. All 5G wireless devices in a cell are connected to the Internet and telephone network by radio waves through a local antenna in the cell.

❶ THE MAIN ADVANTAGE OF THE NEW NETWORKS
is that they will have greater bandwidth, providing far better download speeds- *up to 10 gigabits per second (Gbit/s)).*
Due to the increased bandwidth, the new networks will not just serve mobile phones like existing cellular networks, but also be used as **general internet service providers** for laptops and desktop computers, competing with existing **ISPs** such as cable internet (*Creating separate bindings always available as long your phoneprovider is available - so anywhere in the world*), and also will create new techniques for applications in the **internet of things (IoT)** and **machine to machine areas, probably even in yet non existing projects.** Current 4G cellphones will not be able to use the new networks, which will require new 5G enabled wireless devices. The increased speed is achieved partly by using higher-frequency radio waves than current cellular networks.

❷ HIGHER-FREQUENCY RADIO WAVES
have a **shorter range** than the frequencies used by previous cell phone towers, requiring smaller cells. So to ensure wide service, 5G networks operate on up to three frequency bands, low, medium, and high.

CURRENT 4G CELLPHONES WILL NOT BE ABLE TO USE THE NEW NETWORKS, WHICH WILL REQUIRE NEW 5G ENABLED WIRELESS DEVICES.

❸ A 5G network is composed of several networks of up to 3 different types of cells, each requiring different antennas, each type giving a different tradeoff of download speed vs. distance and service area.

5G cellphones and wireless devices will connect to the network through the highest speed antenna within range at their location:

LOW-BAND 5G
uses a similar frequency range to current 4G cellphones, 600-700 MHz, providing download speeds a little higher than 4G: 30-250 megabits per second (Mbit/s). Low-band cell towers will have a range and coverage area similar to current 4G towers.

MID-BAND 5G
uses microwaves of 2.5-3.7 GHz, currently allowing speeds of 100-900 Mbit/s, with each cell tower providing service up to several miles in radius.
This level of service is the most widely deployed, and is available in most metropolitan areas in 2020. Some countries are not implementing low-band, making this the minimum service level.

HIGH-BAND 5G
currently uses frequencies of 25-39 GHz, near the bottom of the millimeter wave band, although higher frequencies may be used in the future. It often achieves **download speeds of a gigabit per second** (Gbit/s), comparable to cable internet.

Millimeter waves (mmWave or mmW) have a more limited range, requiring many small cells.They have trouble passing through some types of walls and windows. Due to their higher costs, current plans are to deploy these cells only in dense urban environments and areas where crowds of people congregate such as sports stadiums and convention centers. The above speeds are those achieved in actual tests in 2020, and speeds will increase during rollout.

The industry consortium setting standards for 5G is the **3rd Generation Partnership Project (3GPP).**
It defines any system using 5G NR (5G New Radio) software as "5G", a definition that came into general use by late 2018.
Minimum standards are set by the **International Telecommunications Union (ITU).** Previously, some reserved the term 5G for systems that deliver download speeds of 20 Gbit/s as specified in the ITU's IMT-2020 document.

5G ENABLES A NEW KIND OF NETWORK THAT IS DESIGNED TO CONNECT VIRTUALLY EVERYONE AND EVERYTHING TOGETHER INCLUDING MACHINES, OBJECTS, AND DEVICES.

5G is the 5th generation mobile network.
It is a new global wireless standard after 1G, 2G, 3G, and 4G networks.
5G enables a new kind of network that is designed to connect virtually everyone and everything together including machines, objects, and devices.

WHO INVENTED 5G?
A: No one company or person owns 5G, but there are several companies within the mobile ecosystem that are contributing to bringing 5G to life.

3GPP is driving many essential inventions across all aspects of 5G design, from the air interface to the service layer. Other 3GPP 5G members range from infrastructure vendors and component/device manufacturers to mobile network operators and vertical service providers.
The project covers cellular tele-communications technologies, including radio access, core network and service capabilities, which provide a complete system description for mobile telecommunications.

5G is based on OFDM (Orthogonal frequency-division multiplexing), a method of modulating a digital signal across several different channels to reduce interference.
5G uses 5G NR air interface alongside OFDM principles. 5G also uses wider bandwidth technologies such as sub-6 GHz and mmWave.

DIFFERENCES BETWEEN THE PREVIOUS GENERATIONS OF MOBILE NETWORKS AND 5G
A: The previous generations of mobile networks are 1G, 2G, 3G, and 4G.
First generation - 1G 1980s: 1G delivered analog voice.
Second generation - 2G Early 1990s: 2G introduced digital voice (e.g. CDMA- Code Division Multiple Access).
Third generation - 3G. Early 2000s: 3G brought mobile data (e.g. CDMA2000).
Fourth generation - 4G LTE
2010s: 4G LTE ushered in the era of mobile broadband. 1G, 2G, 3G, and 4G all led to 5G, which is designed to provide more connectivity than was ever available before.

WHERE IS 5G BEING USED?
Broadly speaking, 5G is used across three main types of connected services, including enhanced mobile broadband, mission-critical communications, and the massive IoT. A defining capability of 5G is that it is designed for forward compatibility—the ability to flexibly support future services that are unknown today.

ENHANCED MOBILE BROADBAND
In addition to making our smartphones better, 5G mobile technology can usher in new immersive experiences such as VR and AR with faster, more uniform data rates, lower latency, and lower cost-per-bit.

MISSION-CRITICAL COMMUNICATIONS
5G can enable new services that can transform industries with ultra-reliable, available, low-latency links like remote control of critical infrastructure, vehicles, and medical procedures.

MASSIVE IOT
5G is meant to seamlessly connect a massive number of embedded sensors in virtually everything through the ability to scale down in data rates, power, and mobility-providing extremely lean and low-cost connectivity solutions.

WHAT CAPACITY CAN WE EXPECT?
The average consumer is expected to go from being able to consume 2.3 GB of data per month today to close to 11 GB of data per month on their smartphone in 2022.

This is driven by explosive growth in video traffic as mobile is increasingly becoming the source of media and entertainment, as well as the massive growth in always-connected cloud computing and experiences.

A UNIFORM USER EXPERIENCE SO THAT THE DATA RATES STAY CONSISTENTLY HIGH -EVEN WHEN USERS ARE MOVING AROUND.

5G will expand the mobile ecosystem to new industries. This will contribute to cutting-edge user experiences such as boundless **extreme reality (XR),** seamless IoT capabilities, new enterprise applications, local interactive content and instant cloud access.

BUSINESSES EXPECTATIONS:
With high data speeds and superior network reliability, 5G will have a tremendous impact on businesses. The benefits of 5G will enhance the efficiency of businesses while also giving users faster access to more information.

Depending on the industry, some businesses can make full use of 5G capabilities, especially those needing the high speed, low latency, and network capacity that 5G is designed to provide.
For example, smart factories could use 5G to run industrial Ethernet to help them increase operational productivity and precision.

HOW FAST IS 5G?

A: 5G is designed to deliver peak data rates up to 20 Gbps based on IMT-2020 requirements.
In addition to higher peak data rates, 5G is designed to provide much more network capacity by expanding into new spectrum, such as mmWave.

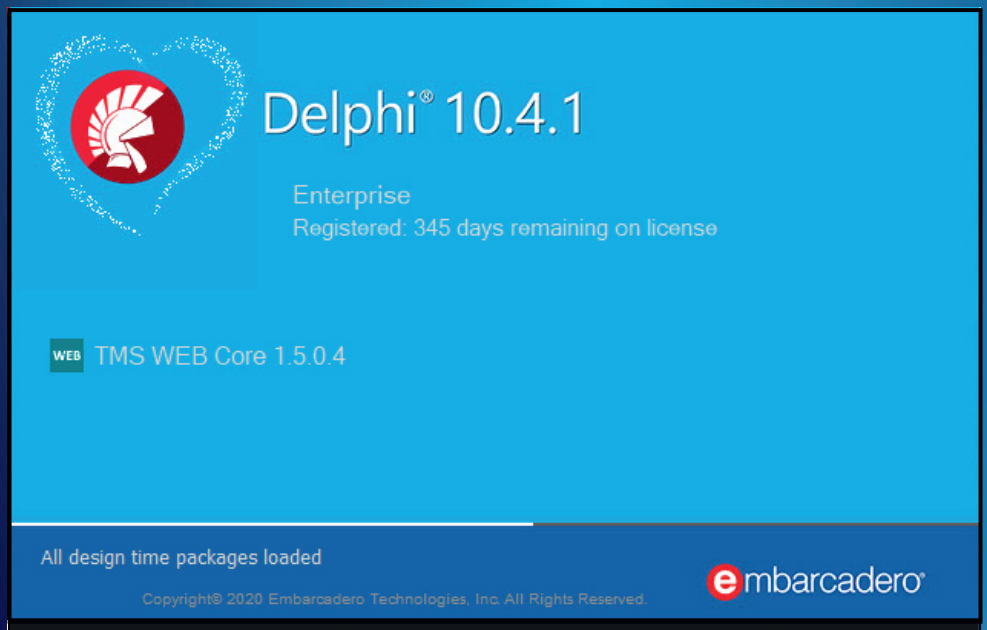
5G can also deliver much lower latency for a more immediate response and can provide an overall more uniform user experience so that the data rates stay consistently high - even when users are moving around.
And the new 5G NR mobile network is backed up by a Gigabit LTE coverage foundation, which can provide ubiquitous Gigabit-class connectivity.

5G has been deployed in 35+ countries and counting. We are seeing much faster rollout and adoption compared with 4G.
Consumers are very excited about the high speeds and low latencies. But 5G goes beyond these benefits by also providing the capability for mission-critical services, enhanced mobile broadband and massive IoT.

ONLY NEW PHONES.
You will need to get a smartphone that supports 5G if you want to be able to use the network.
There are several new mobile phones available that are designed to support 5G, and multiple carriers across the world support the 5G wireless network.
As the 5G rollout timeline progresses, more smartphones and carrier subscriptions will become available, as 5G technology and 5G compatible devices becoming more mainstream.

To make it simple: For us it might be possible (Blaise Pascal Magazines) to use 5G and we will try that. It might mean we can forget about the various providers but have one instead and have a very good up and download connection - not only for our mobiles but also for our other connections like the glass fibre network. But one restriction: it must first be rolled out in our country and we nee to se the effective speed we can achieve.

The text of this article was created by using the website of https://www.qualcomm.com/ and others


Delphi® 10.4.1
Enterprise
Registered: 345 days remaining on license
WEB
TMS WEB Core 1.5.0.4
All design time packages loaded
Copyright© 2020 Embarcadero Technologies, Inc. All Rights Reserved.
embarcadero


RX
RAD Studio Additional Options
Additional Languages
French Language Pack
German Language Pack
Japanese Language Pack
Samples
Help
TeeChart Standard
DUnit Unit Testing Frameworks
InterBase Express (IBX) Components
InterBase 2020 Developer Edition
Download Size: ~ 2,0 GB
Download Time: ~ 33 min
Required Space: ~ 12,3 GB
Back
Install

Code completion and other improvements

After having downloaded numerous updates of Delphi Sidney, I come to the conclusion that Delphi is becoming better, actually very much quicker during installation.
There is one thing I hate:
things that had a certain place in the structure and needed to be found again.
It' s like someone has rebuild your house without asking. Of course this is minor but I had some difficulties to find things under TOOLS and it costed a lot of time to find items again, especially when the naming also had changed.

The first hurdle I run into was the **Code completion issue**.
As soon as I started a project I run into that. It did not work or behave in the way it should, so I wanted to find out how I could overcome this.
After downloading this version of Delphi Studio I already had deinstalled anything old and cleaned directories as well as the registry. So I was quite surprised.
Drinking coffee and reviewing the problem I thought what I had done just after installation: I loaded the project and the because I use a very large screen had to shrink the opened project so therefore I saved that as my own settings.
Just after that I tried to set back everything to **„startup**" settings and repeated the whole procedure.
Now it was saved as **„startup**"I tried to load my project and yes - now the code completion worked correct.
I tried all other settings and that worked also. I mentioned this to the team and created a small video for them. Doing this is always very instructive - if you can reproduce the error.

So now for the first time I can use my fresh Delphi and created one of the projects for TMSwebcore. I have been looking into the new features of Delphi:

A good improvement is Delphi's **code tooling** , 10.4 provides **Code Insight** using a Delphi implementation of the Language Server Protocol (LSP).
LSP is a technique for calculating results for code completion, navigation or similar in a separate process. This means that the IDE will never block while completing and Code Insight will provide accurate results. 10.4 provides a much enhanced developer productivity experience when working with large projects with a vast number of lines of code.

It now has a **Unified Installer for Online & Offline installations,** which comes in handy.
I had sverall oportunities to try that.
In 10.4, a unified installer using the GetIt installer technology was introduced. This provides a single installer that supports both **online** (internet connected) installations and **offline** installations (via an ISO).
Now both online and offline installations allow you to select an initial set of RAD Studio features to install, such as support for specific combinations of programming languages and target platforms, language support, or help resources, and add or remove them at any time. **The removing is not any more a time consuming adventure!**

- There is Added support for the **Metal API on macOS and iOS**

(metal API) Metal *is a low-level, low-overhead hardware-accelerated 3D graphic and compute shader application programming interface (API) developed by Apple Inc., and which debuted in iOS Metal combines functions similar to OpenGL and OpenCL under one API. It is intended to improve performance by offering low-level access to the GPU hardware for apps on iOS, iPadOS, macOS, and tvOS. It can be compared to low-level APIs on other platforms such as Vulkan and DirectX 12.*

Code completion and other improvements

Metal is an object-oriented API that can be invoked using the Swift or Objective-C programming languages. Full-blown GPU execution is controlled via the Metal Shading Language. According to Apple promotional materials: "MSL Metal Shading Language is a single, unified language that allows tighter integration between the graphics and compute programs.

- In addition to supporting the latest iOS SDK, using RAD Studio 10.4 can also address Apple's new launch screen storyboard requirement through built-in IDE support.
- The Sidney release includes a new **FMX** implementation for the **styled** `TMemo` component on the Windows platform, offering better support for IME* and additional enhancements.

*IME - *An input method (or input method editor, commonly abbreviated IME) is an operating system component or program that enables users to generate characters not natively available on their input devices by using sequences of characters (or mouse operations) that are natively available on their input devices. Using an input method is usually necessary for languages that have more graphemes than there are keys on the keyboard.*
For instance, on the computer, this allows the user of Latin keyboards to input Chinese, Japanese, Korean and Indic characters; on many hand-held devices, such as mobile phones, it enables using the numeric keypad to enter Latin alphabet characters (or any other alphabet characters) or a screen display to be touched to do so. On some operating systems, an input method is also used to define the behaviour of the dead keys.

- **Enterprise and Architect Edition** customers can take advantage of FMXLinux integration for building Linux GUI applications.
- The `TWebBrowser` control for **iOS** is now implemented using the `WKWebView API`
- The macOS implementation of Media Player control now uses AVFoundation

The GetIt Package Manager in the IDE includes significant enhancements in 10.4. This includes **displaying release dates** for each package with the ability to sort them by release date; new **filtering options** for installed packages, exclusive content available for update subscription customers, packages for which updates are available.

One thing il myself like very much:
VCL Style Changes for High DPI
In 10.4, the VCL Styles architecture has been significantly extended to support **High DPI and 4K monitors.**
All UI controls on the VCL form are now automatically scaled for the proper resolution of the monitor the form is displayed on. The style API has been fully revised to support high DPI styles.
Each UI element can be selected from a library of multi-scale versions and scaled to any DPI, resulting in crisp UI elements on all monitors.

New High DPI S
There are a large number of updates of the built-in and premium **VCL** styles to provide support for the new High-DPI style mode, letting you design visually beautiful applications for any monitor.

VCL developers can now use multiple VCL styles in different forms within a **single application or even different visual controls** that are on the same form.
This also includes support for styling any element using the default platform theme. Beside allowing more flexibility in styling, this also enables you to use third-party unstyled controls within a styled VCL application.

Code completion and other improvements

Toolchain* performance and quality improvements

- A large number of STL improvements from Dinkumware
- Several key RTL methods and areas improved, based on work done to improve compatibility with common C++ libraries
- Several improvements to CMake support
- A large number of quality and stability improvements
- Windows API Updates
  They enhanced many API declarations and added additional ones, to further improve the Windows platform integration.
- General enhancements to the FireDAC database access library and also updated the drivers for FireBird, PostgreSQL and SQLite.
  Choose SQLite static or dynamic linking.

In software, a **toolchain is a set of programming tools that is used to perform a complex software development task or to create a software product, which is typically another computer program or a set of related programs. In general, the tools forming a toolchain are executed consecutively so the output or resulting environment state of each tool becomes the input or starting environment for the next one, but the term is also used when referring to a set of related tools that are not necessarily executed consecutively.*

A simple software development toolchain may consist of a compiler and linker (which transform the source code into an executable program), libraries (which provide interfaces to the operating system), and a debugger (which is used to test and debug created programs).

A complex software product such as a video game needs tools for preparing sound effects, music, textures, 3-dimensional models and animations, together with additional tools for combining these resources into the finished product.

```
10  type
      TMyRecord = record
12      Value: Integer;
        class operator Initialize(out Dest: TMyRecord);
        class operator Finalize(var Dest: TMyRecord);
      end;

    class operator TMyRecord.Initialize(out Dest: TMyRecord);
    begin
      Dest.Value := 10;
      Form1.Memo1.Lines.Append('created ' + IntToHex(Integer(Pointer(@Dest))));
    end;
40
    class operator TMyRecord.Finalize(var Dest: TMyRecord);
    begin
      Form1.Memo1.Lines.Append('destroyed ' + IntToHex(Integer(Pointer(@Dest))));
    end;

    procedure TForm1.GoButtonClick(Sender: TObject);
    var
      LMyRecord: TMyRecord;
    begin
50    Memo1.Lines.Append(LMyRecord.Value.ToString);
    end;
```

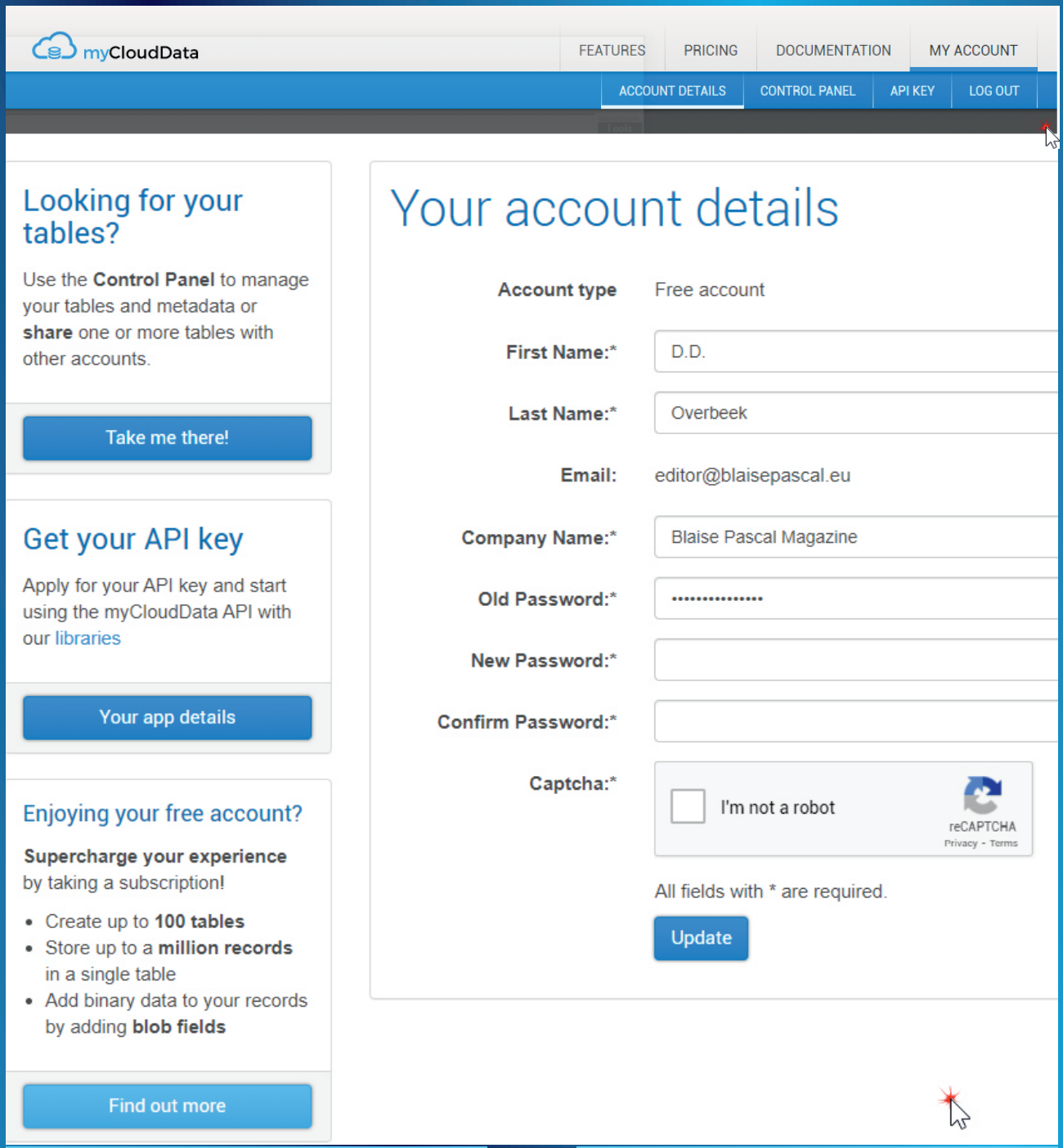
Delphi Custom Managed Records
A key language addition to the Delphi language, **the Delphi record type now supports custom initialization, finalization, and copy operations**.

Developers now have the ability to customize how records get created, copied, and destroyed, by writing the code to be executed at the various steps.

This adds additional power to records in Delphi, a construct used to achieve better efficiency compared to classes, see image.

CREATE A WEB APP FOR BOOK AUTHORS

A first small web app with Client server Model:using TWebmyCloudDbClientDataset of TMS WEBCore



INTRODUCTION

The component TWebmyCloudDbClientDataset makes it easy for a Delphi Web Application to use database tables on the "myCloudData.net" service by a familiar syntax of using ClientDataSet.

It also allows a seamless integration of the "myCloudData.net" data tables with data-aware components like TWebDBGrid.
All the database operations can be done in the standard Delphi way through the TWebmyCloudDbClientDataset component.

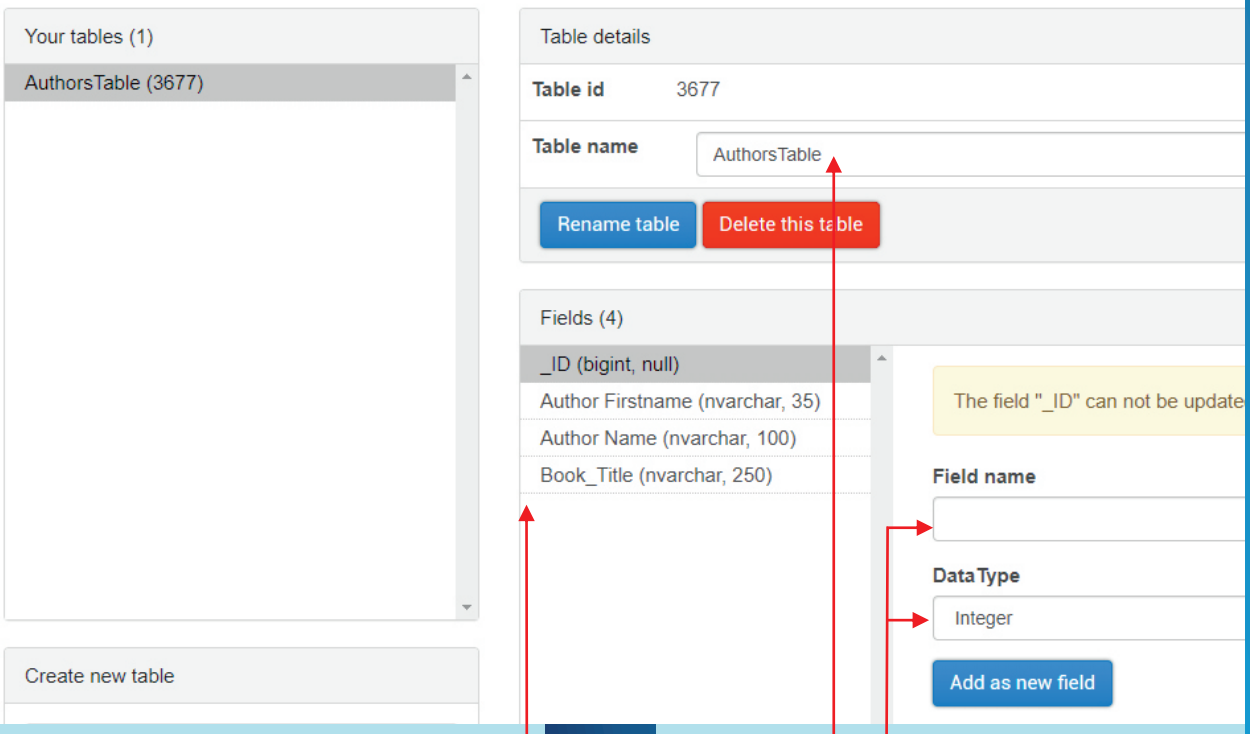
CREATE A WEB APP FOR BOOK AUTHORS

A first small web app with Client server Model: using TWebmyCloudDbClientDataset of TMS WEBCore

All you need to do is specify the **myCloudData** properties and add the field definitions either in design more or in code in a standard **Delphi** syntax.

Then connect a **DataSource** and **Data components** to it and make the dataset active.

Let's create the web application using `TwebmyCloudDbClientDataset`.
Set up your **myCloudData** project in the **myCloudData** console first.



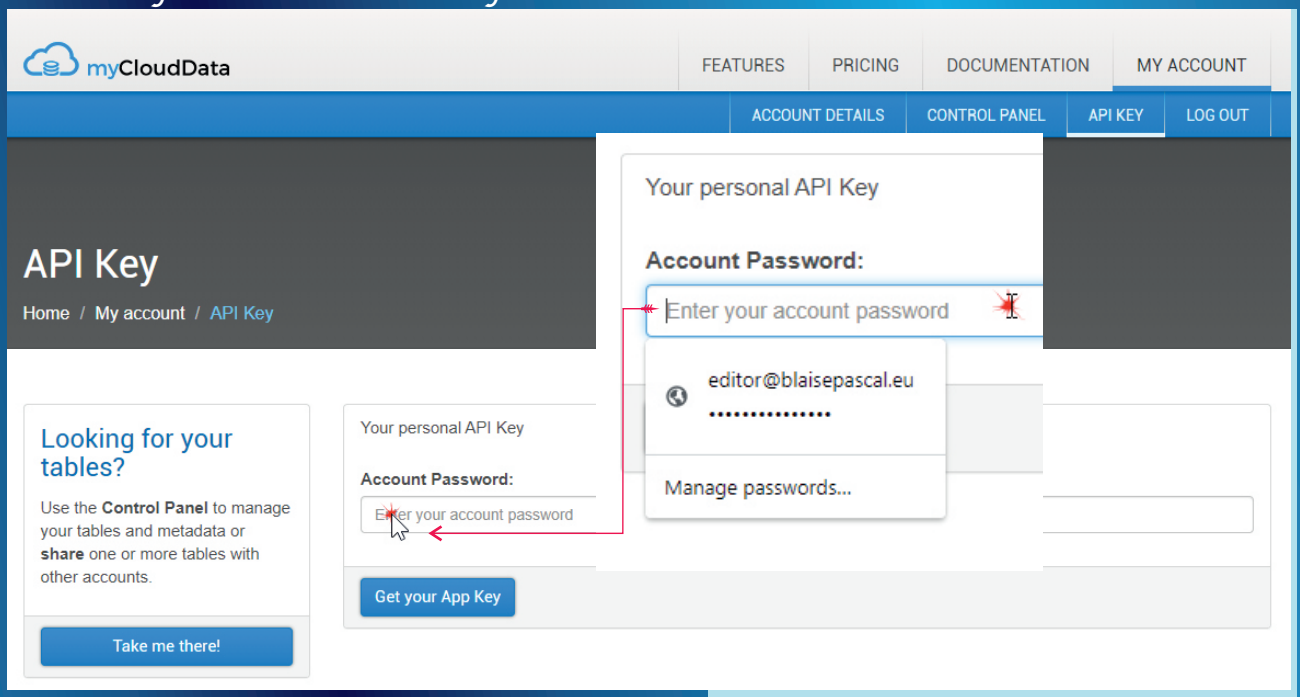
Follow these steps:

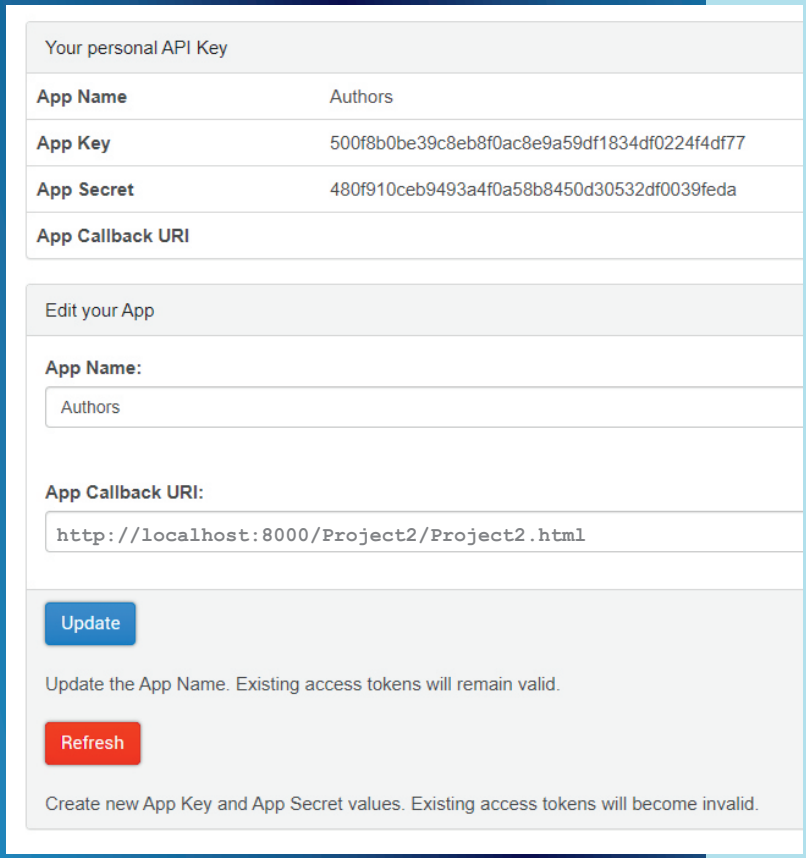
1. Navigate to `https://www.myclouddata.net/` and sign up for **myCloudData**
2. Go to **My Account → Control Panel**
3. Create a new Table and add the required Fields
4. Note the **Table** name.
   This will be used for the `TableName property` later. Creating this table means that you first give in the table name (*It must exist first, so needs a name*) and later the field names and their values

CREATE A WEB APP FOR BOOK AUTHORS

A first small web app with Client server Model: using TWebmyCloudDbClientDataset of TMS WEBCore

Go to My Account → API Key





❺ Enter your **myCloudData** password and click "Get your App Key"

❻ Note the App Key and App Callback URL values. These will be our properties AppKey and AppCallbackURL to be used later later.

Note the **App Secret** value. This will be our property **AppSecret** we need during the setup.

Note that the `AppCallbackURL` should be set to the URL of your web application. This can be different in debugging (typically something like `http://localhost:8000/Project1/Project1.html`) as from a deployed application. (Not my address but it will look like it).

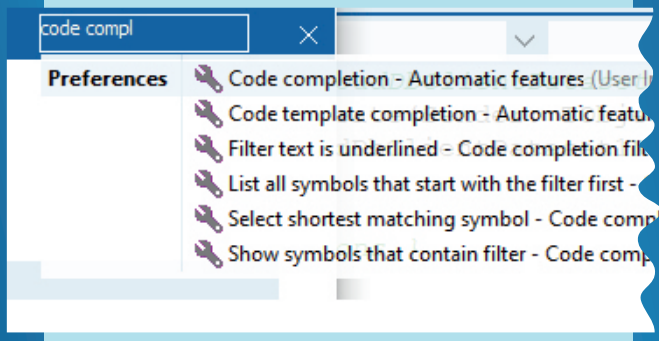
If you do not know what it is just run a very simple project and it will show you the local address
Remember this little app is a client server app and we will create now the server side : https://www.myclouddata.net where it is located and the next step is creating the client side. So even though it's only a Client Dataset it has the same principles as a larger database.

STARTUP DELPHI:

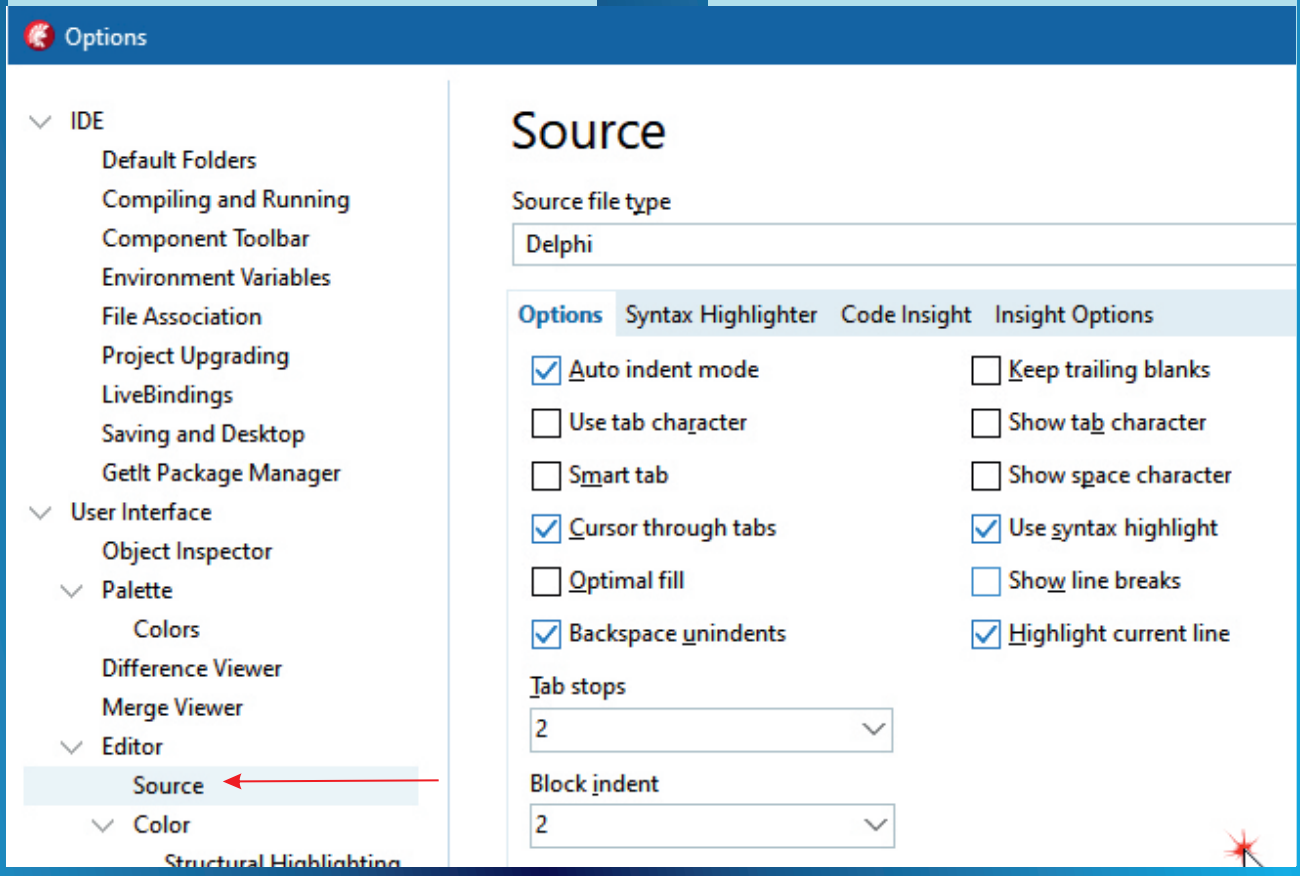
If you use the latest version and have trouble with code completion - there is away to get around this:

Since I had created my own desktop just after starting up, I thought that in this case it might be wise to reset it to Startup Layout and try if then the code completion would work and yes it did after that. I once again tested it by saving this under each kind of desktop and it kept on functioning.

I tried to find the **code completion** in Delphi and looked at all the places after stepping through the tools menu and then in the right top box typing in code completion there was a sign of it:



So I tried to find it in the list under **Tools → Options → Userinterface → Editor** and finally found it under **Source** after having typed in Code… For years I had not looked into that and then you could simple find it under code completion.



Now simply chose **File, New, Other, TMS Web Application**.
A new web form is created. To find out what your URL will be, simply run this empty app and pick up the URL in your browser.
Verify that with what you have already setup in **myCloudData.**

Go to the **Tool Palette** and select the `TWebmyCloudDbClientDataset` component from the "TMS Data Access" section and drop it on the web form.

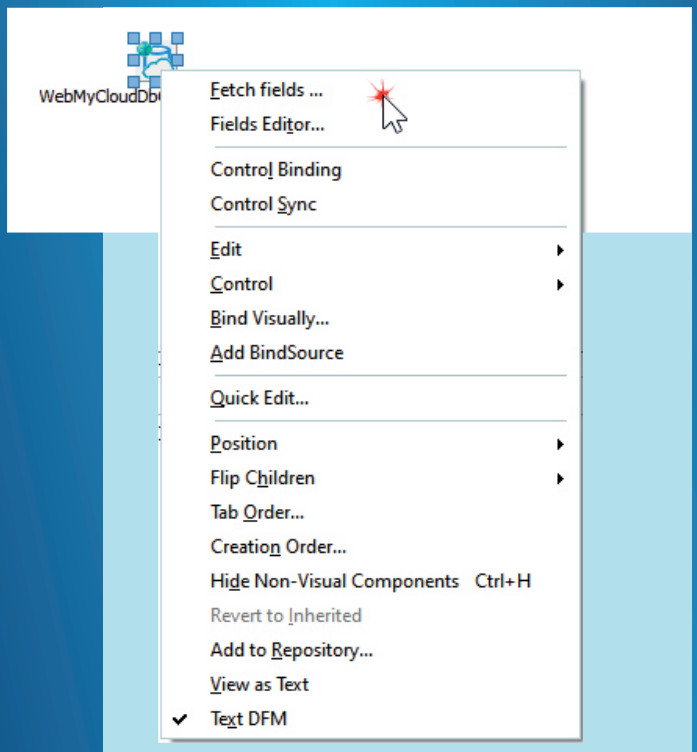
Specify the Component Properties.
Set up the properties either in code or in the **Object Inspector** by right-clicking on the "Fields Editor":

- **AppKey:** from the "My Account → API Key" section of `myCloudData.net`
- **AppCallbackURL:** from the "My Account → API Key" section of `myCloudData.net`
- **TableName:** from the "My Account →Control Panel" section of **`myCloudData.net`.**

Create the Fields or Properties of each object in the Object Store.
The DataSet field definitions need to be set up either in **Object Inspector** by right-clicking on the **"Fields Editor"** or must be created in the **WebFormCreate** event code.

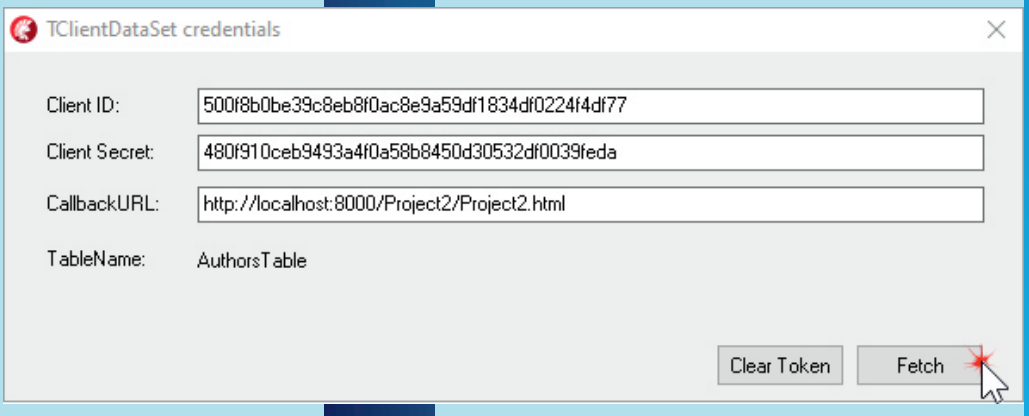
Select the fields in the **Object Inspector**
Follow these steps:

❶ Right-click the `TWebmyCloudDBClientDataset` and select "Fetch Fields" (See right top)

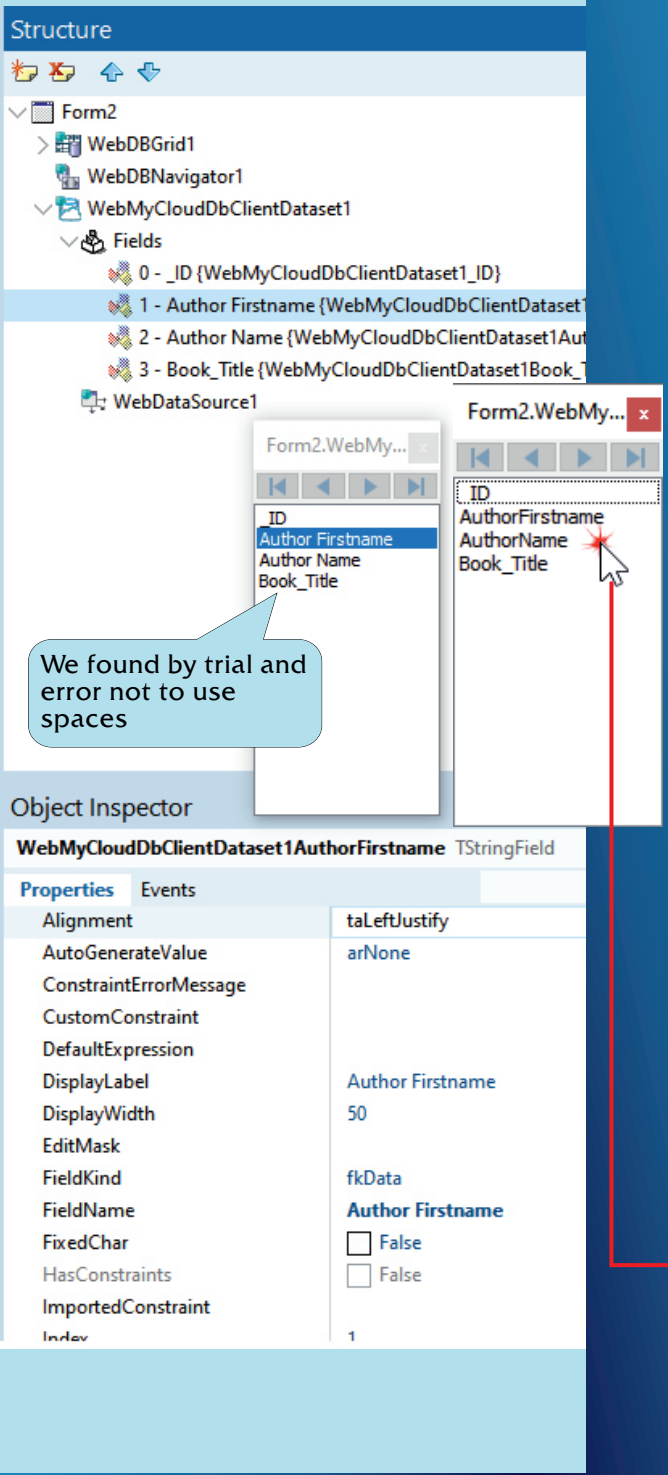


❷. Enter the Client ID (AppKey), Client Secret (AppSecret) and CallbackURL (*A local URL is required here, for example: http://127.0.0.1:8888*) values. Note that the TableName is retrieved automatically from the `TableName` property.

❸ Click the "Fetch" button and follow the authentication instructions.
If the process is successful, a dialogue with the list of available fields is displayed.



4. Right-click the `TWebmyCloudDBClientDataset` and select "Fields Editor"
5. Select the required fields



Create the Fields in code
Here is an example of adding the field definitions in code in the `OnCreate` event.

In the **Object Inspector**, double-click on `OnCreate` event of the Web Form.

This creates an event handler procedure `WebFormCreate`. The following code in it sets up the field definitions.
What fields you add are based on how you defined them for the Table in `myCloudData.net`.

Note that `_ID` field must be defined as data type `ftString`.

Now select and drop a `TWebDataSource`, `TWebDBGrid` and `TWebDBNavigator` component on the Web Form.

Setting up the DataSource and Data components.

Set the `DataSource's DataSet` property to `WebMyCloudDbClientDataset1`.

Then set the `DataSource` property of the grid and navigator to point to `TWebDataSource1`.

Set up the `Columns` of the `DBGrid`
Do that by clicking on the Columns property of the `DBGrid`.

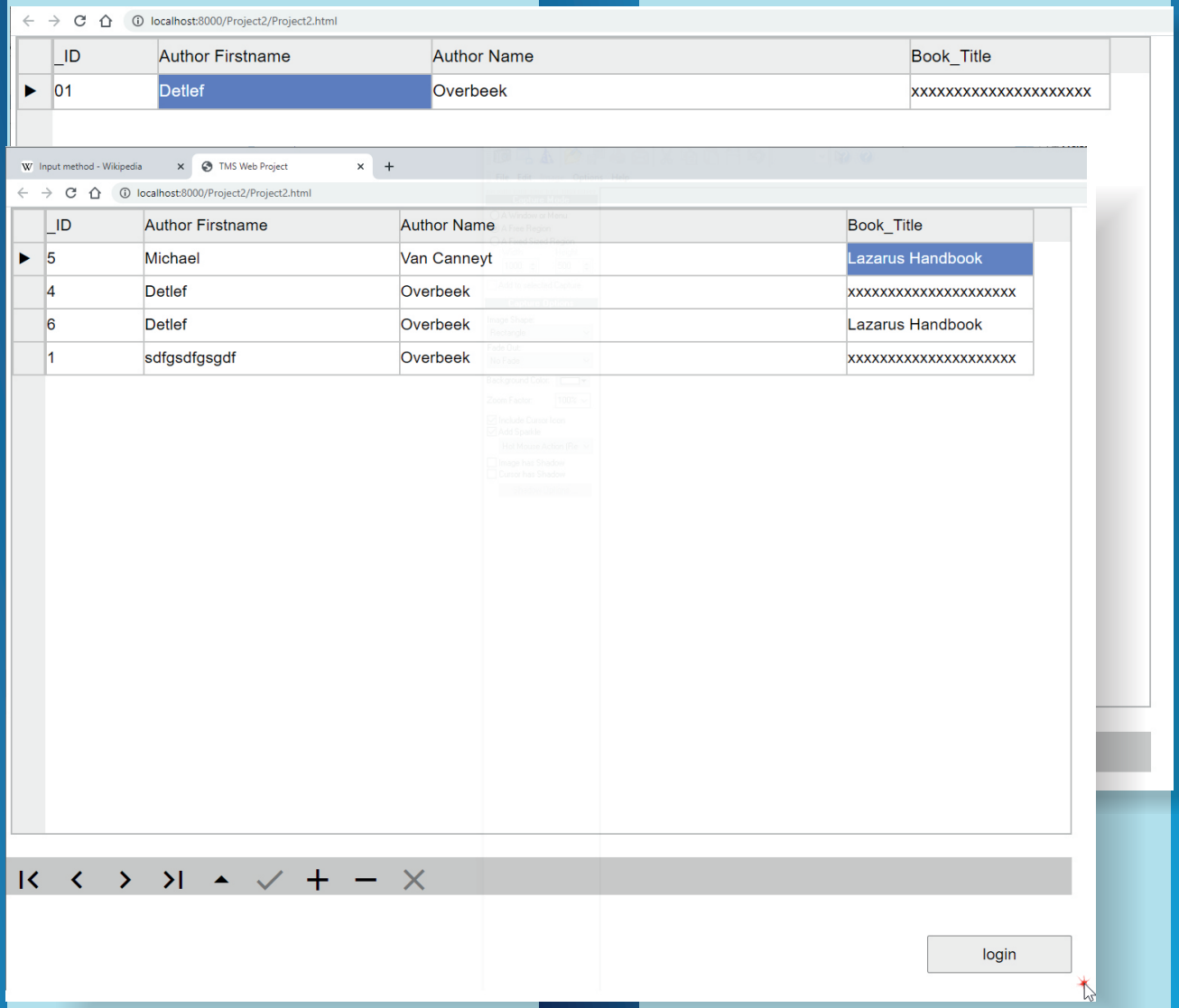
SET UP A NEW RECORD EVENT
Since we will be adding **New Records** with the **DB Navigator**, we need to set up the default values of the record.
For this, we set up an `OnNewRecord` event procedure for the **`myCloudDb Client Data Set`** in the **Object Inspector** and type the following code in it:

VERY IMPORTANT
Under no circumstances you are allowed to use field names that contain spaces to setup in your MyCloudData:
`AuthorFirstname` **= Correct**
`Author Firstname` **= NOT ALLOWED**
`Author_Firstname` **= ALLOWED**

CREATE A WEB APP FOR BOOK AUTHORS

A first small web app with Client server Model: using TWebmyCloudDbClientDataset of TMS WEBCore

```
procedure TForm2.WebMyCloudDbClientDataset1NewRecord(DataSet: TDataSet);
begin
 DataSet.Active := True;
 DataSet.Edit;
 Dataset.FieldByName('AuthorFirstname').AsString := 'Detlef';
 Dataset.FieldByName('AuthorName').AsString     := 'Overbeek';
 Dataset.FieldByName('Book_Title').AsString     := 'xxxxxxxxxxxxxxxxxxxx';
end;
```



Reference Section

`TWebMyCloudDbClientDataset`

Below is a list of the most important properties and methods of TWebIndexedDbClientDataSet component.

Properties of TWebmyCloudDbClientDataSet

PROPERTIES OF TWEBMYCLOUDDBCLIENTDATASET

PROPERTY	DESCRIPTION
`Active`	Set this to True to activate the DataSet. Field definitions must be present along with other properties described below.
`AppKey`	Get from the "**API Key**" section of `myCloudData.net`.
`AppCallbackURL`	Get from the "**API Key**" section of `myCloudData.net`.
`TableName`	Specify a table name to connect to from the "**Control Panel**" section of `myCloudData.net`.
`OnError`	This is an event property that notifies the application of any errors from `myCloudData.net`. The event can be set up at design time in `Object Inspector` by double-clicking on it. If the Application does not subscribe to this event, an Exception is raised on such errors. If subscribed, the application can then decide what to do. For example, show error, raise exception or take some corrective action. Note that hard errors (Delphi Exceptions) are not passed in this event. Rather, they cause an Exception that appears in a red alert. But in any case, all errors are always logged to the browser console.

Methods of `TWebmyCloudDbClientDataset`

Only the methods specific to **myCloudData** are listed below. Other methods from the base DataSet classes are used in the standard way.

REFRESH

```
procedure Refresh(Force: Boolean=False);
```

Refresh reloads all the objects from the database. If `AddSortFieldDef` has been used to set up sorting definitions, the objects are loaded in the order specified. In addition, the current record pointer is restored after the Reload which is convenient for the user interface of the web application. Refresh is internally postponed till all the pending updates started asynchronously are finished.
The Force parameter ignores the pending updates and forces a reload.

`AddSortFieldDef` and `ClearSortFieldDefs`

Use `AddSortFieldDef` to add one or more sort definitions for loading the data.
Before using a series of these calls, you must clear all sort definitions by calling `ClearSortFieldDefs`.

```
procedure AddSortFieldDef(aField: String;
isAscending: Boolean));
```

`Where`

- aField - the field name for the sorting order
- isAscending - Set True for ascending order.

`ClearTokens`

After a successful authentication & authorization, the `TWebmyCloudDbClientDataset` will store the obtained access tokens in the local storage so that a next time, this does not need to be obtained again. If for some reason this needs to be removed, call procedure ClearTokens;

The project is downloadable including code. In the next issue I will extend this app to extra functionality

Image- Line is one of those company's I had no clue they existed, nor what they did or mean today.
Out of the blue I received a message which made me very aware of them: they are the company for musicians from very young to aged people, and the typical songs and sounds they produce are from this very time: the future?
I had always an interest in music but the shear number of youngsters using FL Studio I couldn't have imagined. The songs that are created are thousands per day and all transformed into video and online, on the Internet. It must be millions of them…

All this came to my mind at the moment, later as I realized there is a world out there I do not know…
That morning we had a telephone message who made clear that someone wanted to help us and sponsor us. As I am very suspicious I said thank you and told him to call me an other time and that I was in a conference meeting – actually I was very busy trying several calls at the same time.

But he insisted and kept being polite and seriously called me back a few days later. Than we had a better conversation: I found out they wanted to support us for the Lazarus/FPC project and that they were so much content with the development we had with this project that they wanted to spend money so we could do more development for the future.

Truly to good to know! But because of this we had to find out who these people are and that's how this article started: I went to the internet and found out.

I was quite right about the numbers: it are millions of people – the program is downloaded each day by 30.000 people, all new tryers. Incredible.
Now the program they created is of the category **Digital Audio Workstations** for creating music.

If you realize it's been used by the greatest musicians: Not only youngsters but people like Mike Oldfield – who doesn't remember Tubular Bells, even I was young at that time – Martin Garrix, one of the very well known, Avicii the Swedish DJ.

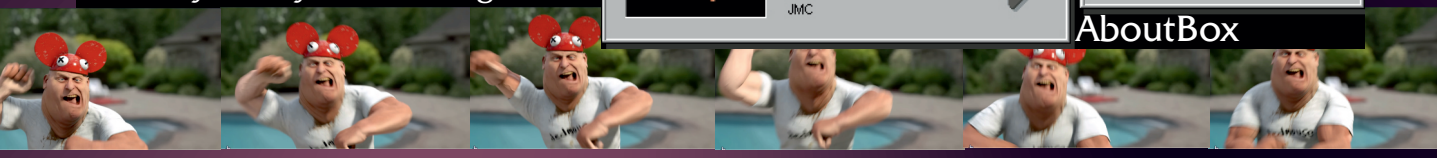
"Tubular Bells"

So after finding out what company was behind this all I was not any more suspicious but became very enthusiastic to get to know what they had achieved: written in Pascal about 2.5 million lines of code, they created a very special version of the VCL and adding 300 assembler functions. Jean Marie Curie, the founder and CTO of Image line software company explained to me that they started the whole project in turbo pascal (that old, yes) and naturally went to Delphi , C++ was late added to do the mobile version.
They wanted to create for the Mac as well and recently Mac had a new condition: the ARM chip. The new Macs are going to work on ARM and their new OS Catalina works only with the new Cocoa and 64 bit.
They alreday had started with FreePascal, because of the lack of 64-bit MacOS in Delphi.
This software company from Ghent /Belgium changed the Digital Musical world into the most downloaded and loved creative audio tool for singers, songwriters and composers. Crazy and beloved people…And they are young, (some older artists - already made their brake through) but an ocean of creativity all made possible through this program: **FL Studio**. The sheer fun of it - almost radiating - is created with not only the music but also the video-creativity is really overwhelming.

I can imagine that artists are becoming addicted to work with this… Now to explain things from history, the first version of **FL Studio** was released twenty years ago. Jean-Marie Cannie, founder of "Image-Line" explained, run me through the years and told about the success and history of the most popular music program in the world.

Exactly twentytwo years ago, Didier Dambrin, an employee of the Belgian software company Image-Line, quietly unleashed a musical revolution. What the **Fender** (*1) was to rock and roll and the Roland TR-808 (*2) to hip hop,
"FruityLoops" - now called FL Studio – brought all this to digital music production. A new legend was created.
The democratization of music had begun. The program's accessibility enticed a generation of pockmarked teens to hammer beats into the night on -cheap?- Windows computers. (Mac is status, Windows is….) and status is important on that age.

AboutBox



Pockmarked teens grow up, sometimes become larger than life. More than ten years after Didier put the first version online, the experimental hobbyist **Soulja Boy** released the hit **Crank Dat** - a bold statement that turned the music world upside down.
The song was only created with presets from the DEMO VERSION of: FruityLoops.

Today, much of the music on the iTunes charts is produced with **FL Studio.**
From the rattling hi-hats and deep 808 thumps of Metro Boomin, Hit-Boy, Zaytoven, Lex Luger, Sonny Digital and 808 Mafia to the pompous electro-kitsch of Avicii, Afrojack, Basshunter, **Deadmau5** and **Martin Garrix**. Benga and Skream made their first Dubstep beats with it, **Boi-1da** and 9th Wonder won Grammys with it. PartyNextDoor has named itself after a preset from the program and try to find a reggaeton or bubbling producer who doesn't use it.

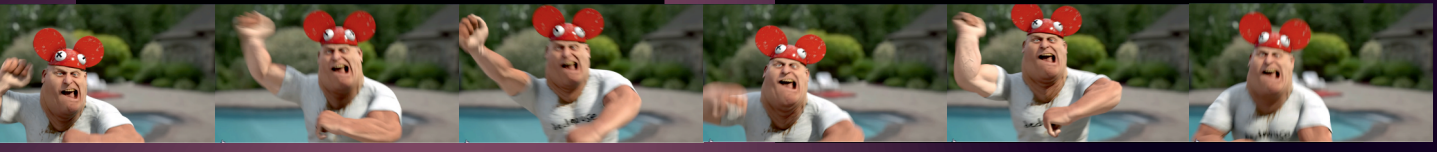
In **1992 Jean Marie founded** the game company Image-Line together with Frank Van Biesen. "At that time everyone actually wanted to develop games.
That was the most fun and got the most response."
The brand new company directly focused on a striking niche in the world: erotic games.
The first game that the Image-Line released was the cult classic Porntris: Tetris, but with

"I wrote that myself at the time.
Unfortunately, it turned out that the erotic world was barely breaking even. There was not much money to be made."
In 1995 IBM organized a game competition where you could win a color laptop. For me that was the main reason to participate.
Twenty years ago, such a device cost a fortune. Five thousand euros, something like that. The game I developed, **"The Da Vinci Connection" won first prize in its category.**

Didier - nineteen at the time - won all other categories, plus the overarching prize: a visit to IBM headquarters.
The game he created was way above all other entries.
I immediately saw that he was exceptionally talented and I really wanted him to work for Image-Line.
At the time, Didier was at home on benefits and was not exactly looking for a job.
He thought it was fine like that. It wasn't until I promised him a new computer if he'd come to work for me that he agreed.
Anyway, not long after,

Didier put together a midi drum sequencer in his spare time.
Voila, FruityLoops was born.

Did you immediately see the potential in **FL Studio?**
No. Not at all. We knew nothing about music. The first time we threw it online, the server went down. Then I thought: oops! What is going on here? Then we looked for a bigger server. Flat again. An even bigger server. Flat again. This went on for years. Everywhere we tried to host it, the servers went down. After that it was pretty clear it was a hot item.



And Didier? He was glad that something happened to it at all. He himself had neither the social nor the commercial attitude to do anything with it. He didn't really care about that aspect. Probably, Didier had just dumped it somewhere on the web for free and it died a silent death.

Talking about Didier: He is very introverted and hesitant about all forms of contact.
That is probably the personality you need to program 17 hours a day, 7 days a week for 20 years.
Didier was able to work for months at a time without leaving his home. In total he has written 2.5 million lines of code.
That is immense! **Someday we will print it on paper as a piece of art.** It is truly unbelievable what that man has created!
Its funny to realize -we had no musical past - maybe that's a reason for the success.
Philosophically you can think of it like this: **FL Studio** is a tool that produces music, rather than a musical tool. Didier has always seen **FL Studio** as a game. He just wanted something that looked nice, moved smoothly and was easy to operate.

Something all user interfaces should have. Hence, our piano roll, highly praised by many people, looks the way it does.
Funny to know that For Didier the music was not important at all. He thought how the notes were entered was more important than how they sounded. He himself almost made no music with it.
In the beginning we didn't make any money. Funny? Not at all... From 1998 to 2004 it yielded only small amounts.
Although it was one of the most popular software programs, we didn't sell any licenses. The income increased slightly over the years. But now we are at **30.000 downloads per day.**
The few that bought it wasn't even enough to keep it online. It were the other products that paid for it's Existence.
If **FL Studio** had been our only product, it could never have survived. Fortunately, we had a lot of other programs we had revenues from. We had quite a bit of extra work. We have created software for, among others, Radio 538 and for the Belgian game show Blokken. Those projects were funding **FL Studio**.

Of course it was a shame to have to do additional work to cover the costs. It seems very demotivating to young programmers. Anyone looking to start something new will run into the same problem.
You first need millions of users before any revenues will come. There are tons of good products that will never be developed because of illegal downloads.
We got to know that our software was radically changing the musical landscape.

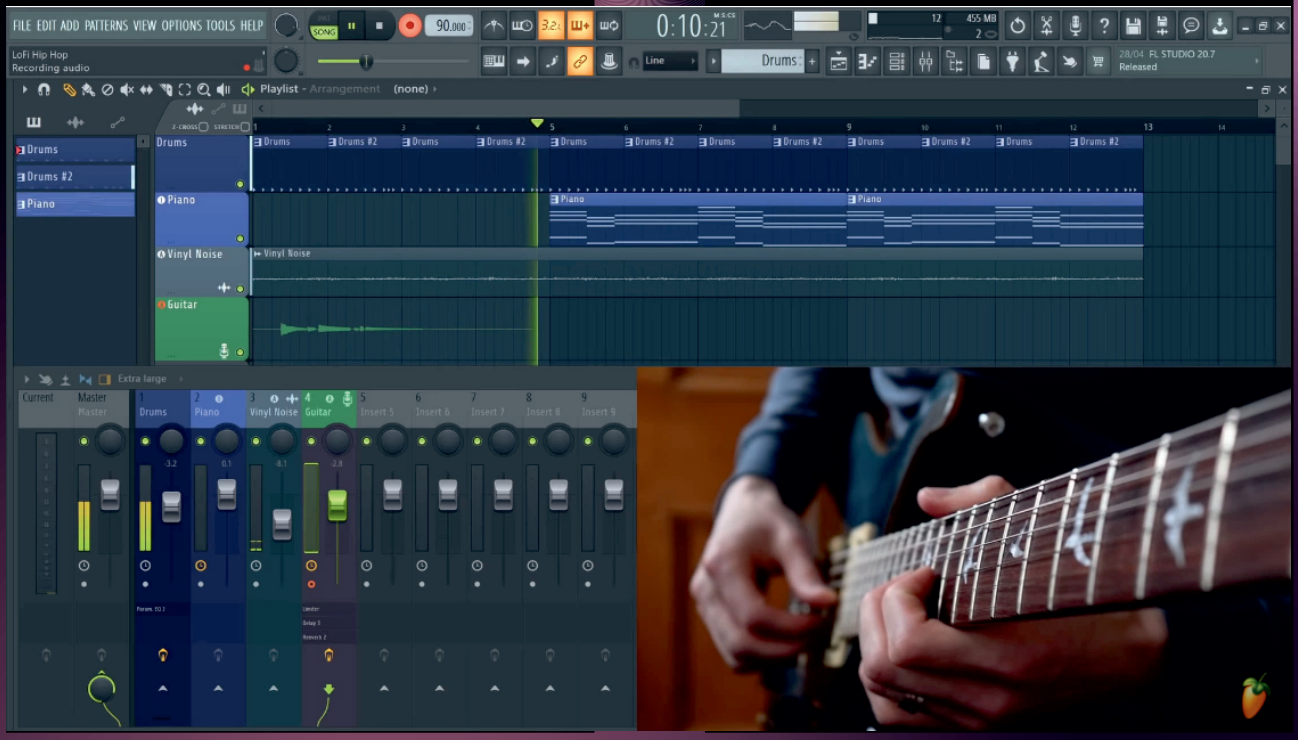
That took some time to realize.
In time saw an interview in “The New York Times" with my idol: Mike Oldfield.
“A photo accompanying the article showed his studio and a computer with FL Studio open.
I couldn't believe my eyes. Then I just sent him an email to see if what I saw was correct.
He replied immediately. "Yes," he wrote, "I use it regularly. I had a few million pound studio built next to my house, but I haven't been in since I met FL Studio."
Gradually we saw more famous musicians emerge. In the hip hop scene,

9th Wonder and Soulja Boy have been very important to us. Before they started publicly saying that they were making music with FL Studio(it was not done to use the program).

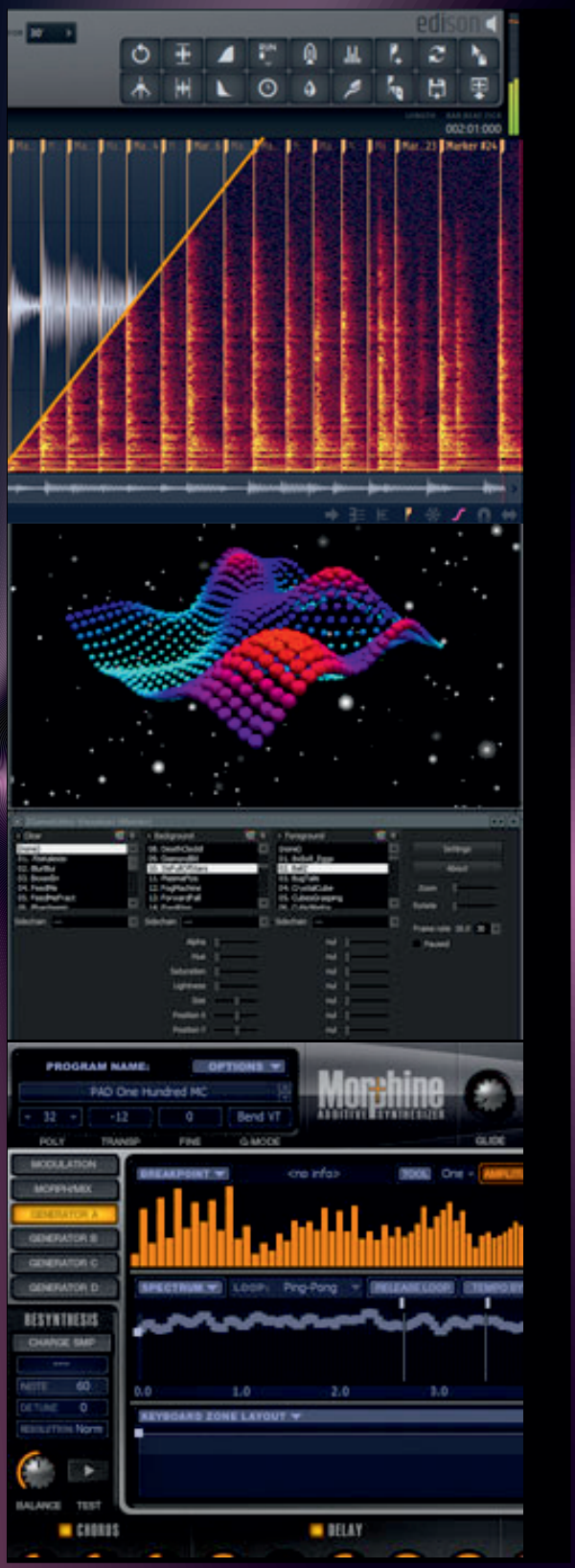
This was mainly due to the confusing name. At the time, the **“Dance eJay”** program was popular. That was a program that allowed you not to do anything else but merge pre-made loops.
A sort of copy and paste. You didn't actually make music. Anyone who heard the name 'FruityLoops' (Now FL Studio) immediately thought our program also worked with loops and acted alike. Of course, none of that is true.

We had to change the name to FL Studio because the name had some wordly connections which were very damaging.
For us it was important to get rid of that awkward story.

In the United States, the word 'Fruity' is also often associated with homo-sexuality.
That was not really accepted in the group of tough hip-hoppers by that time.



Plugins & Instruments



On top of that came the "Kellogg's" story. FruityLoops (now FL Studio) was too similar to Froot Loops.
We had a trademark in the Benelux, but as we grew, that name caused problems. Kellogg's has reportedly done something with music in the past as well. For years they supplied CDs with the cornflakes and they even bundled their own sequencer in the past.

So their case was completely justified. Not many famous musicians contacted us. Which was very disappointing. We had hoped for a lot of mutual feedback.
The company had been around for a very long time (twenty years) now and we had hoped for a nice compilation video with famous users congratulating us, but it is almost impossible to get hold of those guys, - to busy on their agenda.

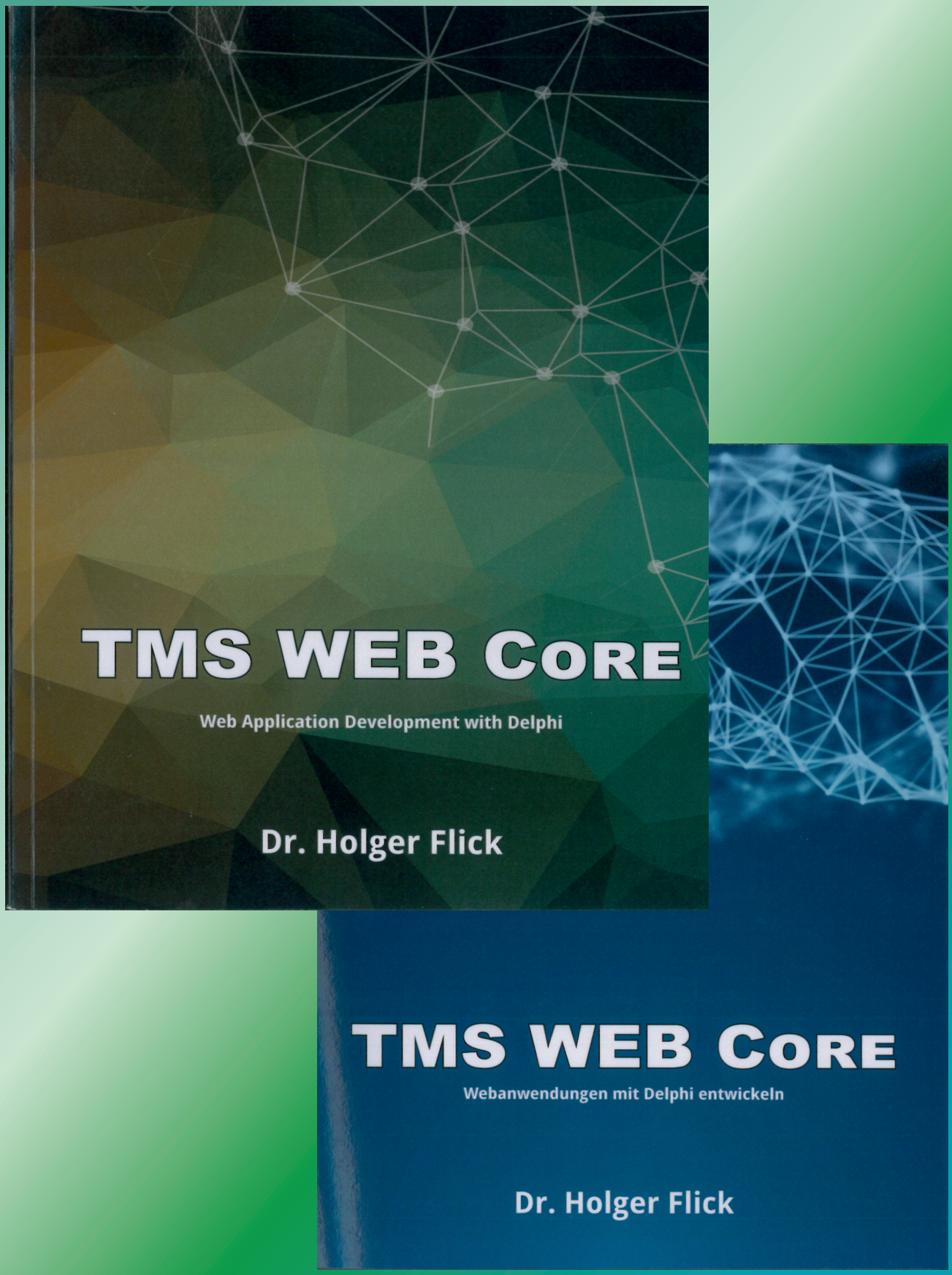
And the Mac version?
We've been working on it for a quite a while and now it has been released since about 2years.
It may seem strange that we have so many users while nowadays everyone uses Mac, but we Europeans sometimes forget that in Africa, South America and the middle of the United States, the old fashioned Windows is the main OS. But anyway, the Mac version is now the number one priority.
If we had written the program directly for Mac, a lot of things would have turned out differently.
Deadmau5 (one of the artist that became famous) for example - he lived and worked with us for a while - produced with FL but wanted to perform live with his Mac.
He switched between PC and Mac for a while. He would load everything into **Ableton** for a performance. In the end, he chose Ableton completely.
If that Mac version had existed before, it probably never would have switched.

It may seem strange that we have so many users while nowadays everyone uses Mac, but we Europeans sometimes forget that in Africa, South America and the middle of the United States, the old fashioned Windows is the main OS.

He was one of our testers. Sometimes we see talented users who make beautiful things. We then recruit them for our Alpha Team. They are allowed to test and comment on the first versions. That's how we noticed **Deadmau5** too. He was not yet known at all and was unemployed in Canada. I then asked him if he would like to live here in Belgium for a year and work with one of our programmers.
He even lived in my house for a while. Actually, it has yielded little.
He made some loop kits for us, but we got into trouble with that later.

The Faxing Berlin loop, for example.
He actually exploded because of that.
Anyway, he went back to Canada and became a world star, while that loup could still be used in **FL Studio**.
When someone else released a song with that loop kit, it suddenly caused a conflict with Deadmau5's record company.

Actually, we mainly used Deadmau5 to give demonstrations and master classes at music fairs. A lot of people have probably had a demo of Deadmau5 without knowing it.


TMS WEB CORE
Web Application Development with Delphi
Dr. Holger Flick
TMS WEB CORE
Webanwendungen mit Delphi entwickeln
Dr. Holger Flick

BOOKS: TMS WEB APP DEVELOPMENT WITH DELPHI
BY HOLGER FLICK

AUTHOR
Holger Flick is the author whome has written this book

BOOK:
Published on “Amazon”.
If you want to order the book:
Go to Amazon or search for the **ISBN Number 9798615692895.**
If you live in the US that is no problem, but if you live elswhere you need to go for the English version to Amazon UK or for the German version to Amazon Germany.

TITLE:
Web Application Development with Delphi
That is because US does not ship over here in Europe. You can not buy or order it in any bookshop.
In the coming pages, we give a complete overview of all chapters and some pages so you can get an insight in the content of the book.

The printed quality of this book is reasonable, sometimes the images are very small but it gives a hint what you need to or can do.

This kind of prepared books can be very nice for quick publishing, but the quality of printing and binding is low, of course the book is glued and not sewn.

It was created in two languages: English and German, which can be an enormous advantage if you are talking in development terms. English is the standard, but German is the very much wanted extra language edition.

That said - it is very good we have finally an Author that deepens out the subject and does research on - in this case the Web-core Suite and its components.

The content is suitable for both beginners and advanced developers interested in creating web applications with TMS WEB Core.
Knowledge of Delphi (Object Pascal) and the Visual Component Library VCL) is required.
To reproduce the numerous examples, ou need a current version of Delphi and TMS WEB Core. The free Delphi Community Edition is sufficient as well as the trial version of TMS WEB Core. I

For more than two decades, the development environment Delphi is known for the rapid programming of Windows applications. Especially the easy creation of desktop database applications and the uncomplicated deployment of the applications to customer systems made Delphi popular compared to other programming languages. For several years now, software can be created for the target platforms Linux, MacOS as well as the mobile operating systems iOS and Android. With TMS WEB Core, modern web applications can be programmed with Delphi since the beginning of 2018. These applications can be executed completely in the web browser because they are translated into JavaScript.

- Detailed description of the basics, the functionality, and the transpiler (based on pas2js)
- Step-by-step creation of the first web application
- Progressive Web Applications (PWA) for offline use
- Electron applications: Cross-platform Desktop applications based on web applications
- Integration of JavaScript classes and controls
- Creating web services for databases with TMS XData
- Integration of databases with TDataset controls
- XData-specific functionality for use in web applications
- Responsive web design (form designer, HTML, CSS, Bootstrap)
- The final chapter provides a comprehensive and practical example of server and web application with Google Maps and Google Charts

Content Overview

Content Overview

Content Overview

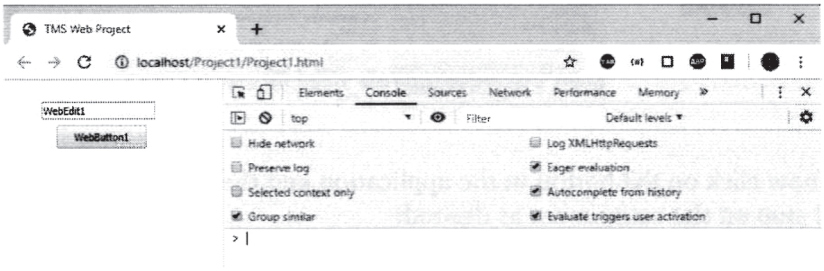
Content Overview

3.5.2 Using the web browser developer console

For debugging, run your application and start the developer console of your web browser. In Google Chrome and Mozilla Firefox, you can reach this console by pressing F12.
As an example, we will create a simple application consisting of a button *TWebButton* and a *TWebEdit* edit control. The button executes the following source code when clicked:

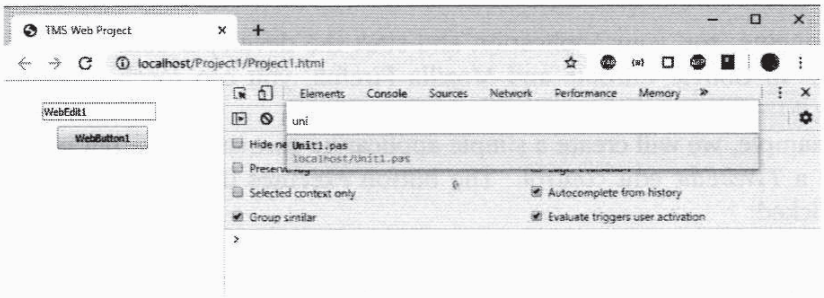
```
1  procedure TForm1.WebButton1Click(Sender: TObject);
2  var
3    LDate: TDateTime;
4  begin
5    LDate := Now;
6    WebEdit1.Text := DateToStr( LDate );
7  end;
```

If we start the application and open the web console, we are presented with the following screenshot when using Google Chrome:

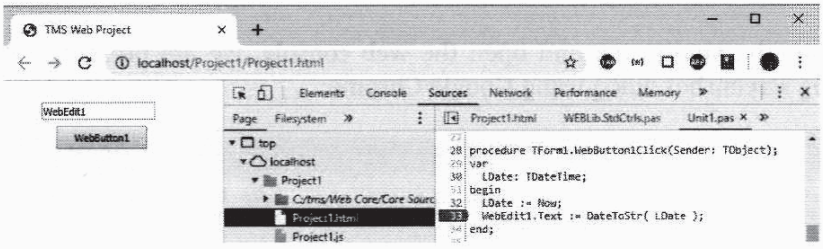


At the moment, we have no access to the source code. With the shortcut Ctrl+P we can select a unit. It is also possible to enter parts of the file name. Here is the opened console with the name *uni* as search term:

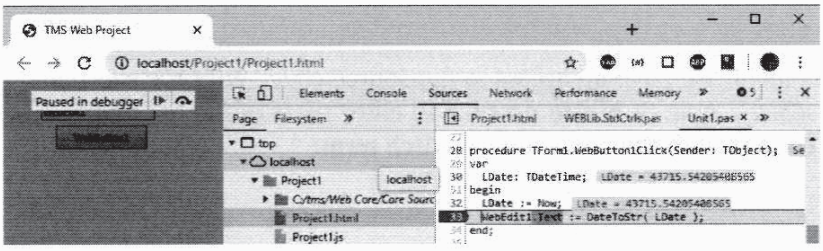
Page views



If we select *Unit1.pas*, we can set a breakpoint by clicking on the line number as we know it from Delphi. Here the breakpoint was set on the assignment of the variable to the graphical component:



We can now click on the button in the application and the execution of the application will stop on the assignment as desired:



You can now very comfortably view the values and even set other values for variables. Consult the documentation of your web browser, which functions are provided and how you can use them. Keep in mind that this is a web application like any other and therefore you are not subject to any restrictions. However, you have the advantage that you can work with Delphi source code and do not have to learn

The pages are shown on a readable size so you can see what the quality of the text is

Page views

Note

The category TMS Web » Compile is specific to RAD Studio 10.3 and later. The appearance of the dialogs from version 10.3 and later differs due to the modernized layout introduced with that version.

3.7 Building blocks

TMS WEB Core has a modular structure. It consists of the basic module called *Core*. Based on this core, further modules can be used. Which modules are available depends on the selected licensing model.
The following modules are available. Figure 3.7 on page 86 presents the modular structure of the framework graphically as a mind map.

TMS WEB Core
The basic package as the basis for all further modules. Besides the basic components it also contains all elements for IDE integration. The special compiler (transpiler) for creating web applications is also included in this package.

TMS FNC for the Web
FNC components for the Web. Many of the components from the FNC UI package are also available for TMS WEB Core with this module. These kind of components form the highest level of abstraction, since the components do not differ in the different frameworks of Delphi. This means that you can use the FNC components in the web in the same way as you are used to from VCL or FireMonkey applications with FNC components.

jQuery components
Components from the jQuery JavaScript framework can be used with the help of components from this module. At design time, the components are indicated by white frames in the form designer. WYSIWYG is not supported because the graphical representation is only available in JavaScript.

Cloud services
Components for direct use of cloud services, such as Google Calendar. When using these components, you do not need to worry about implementing the communication interface between your application and the cloud. Especially aspects such as encryption, authentication, and authorization are made easy with these components.

XData
External databases can easily be accessed implementing an XData server. These

The pages are shown on a readable size so you can see what the quality of the text is

servers are REST web services that are automatically provided by the TMS XData technologies. Especially by using TMS Aurelius the development process chain of your application development comes full circle: You develop your database structure, use TMS Aurelius to connect your database to your application logic and then use XData to make the information available to the outside world. Your graphical user interface on the Web, developed with TMS WEB Core, can access the data via this general, standardized interface and even transfer new records or changes to the database.

Note

Even though there is only one building block for the jQuery JavaScript framework so far, TMS WEB Core can still be used together with other JavaScript frameworks that offer visual components.
Read in section 3.8.4 on page 70 how to use or include external JavaScript source code.
It is recommended to contact TMS to express interest in special frameworks. Without being aware of the need for a specific framework, TMS will certainly not provide any building block or individual components.

3.8 Pascal to JavaScript Transpiler

As explained in the previous sections, the web application is created by converting the Pascal source code into JavaScript. The software that performs this kind of translation is formally called *transpiler*. The resulting JavaScript web application can then be executed completely on the client side in any web browser. The connection to the web server is usually established via HTTP or WebSocket communication channels. To convert Pascal source code to JavaScript, the *pas2js*-transpiler is used.[1] The transpiler is developed as an open source project by the Free Pascal Compiler Team and is based on years of experience.[2]

3.8.1 Supported Delphi language features

At the moment, the transpiler is almost identical to the Delphi language, but some elements are not yet supported. Currently, it is **lacking** support for the following language elements:

Generics:

Generic classes, for example *TList<string>*. Generic classes guarantee a reli-

[1] short for *Pascal to JavaScript*
[2] `http://wiki.freepascal.org/pas2js`

The pages are shown on a readable size so you can see what the quality of the text is

Page views

able type security already during the compiler run. Many runtime errors occur due to typecasting. Using generic classes makes many typecastings unnecessary, so that many runtime errors can be prevented.

Custom Attributes:

The labelling of classes and methods. This functionality is often used when objects are to be automatically stored in streams. Below is an example of an interface that serves as the basis of a web service. For example, the method *Sum* is defined for access via an HTTP GET Request:

```
type
  [ServiceContract]
  IAccountingService = interface(IInvokable)
    ['{E5A9D9F0-28F1-4B6B-AA26-9995034BF8F4}']

    [HttpGet]
    // Berechnet die Summe der Zahlen A und B
    function Sum(A, B: double): double;
  end;
```

Advanced Records:

Records with type declarations, variables, properties and methods. Developers who still prefer records over classes - for whatever reason - can now even define methods in records in Delphi.

```
type
  TMyRecord = record
    type
      TInnerColorType = Integer;
    var
      Red: Integer;
    class var
      Blue: Integer;
    procedure printRed();
    constructor Create(val: Integer);
    property RedProperty: TInnerColorType read Red write Red;
    class property BlueProp: TInnerColorType read Blue write Blue;
 end;
```

Advanced RTTI:

The query of runtime properties has become popular through Java and C#. Both languages offer exemplary support for querying the runtime environment. For example, you can query which methods a class provides, from which base class a class was derived...

The pages are shown on a readable size so you can see what the quality of the text is

Page views

Since some versions the Delphi runtime library contains the record *TRttiContext* for querying runtime properties. The record is an advanced record and also uses generic data types, so that for these reasons it is not possible to query the runtime environment with it in TMS WEB Core.

```
1  TRttiContext = record
2    private
3      FContextToken: IInterface;
4    private class var
5      FGlobalContextCounter: Integer;
6      FGlobalContextToken: IInterface;
7    public
8      class function Create: TRttiContext; static;
9      procedure Free;
10     class procedure KeepContext; static;
11     class procedure DropContext; static;
12     function GetType(ATypeInfo: Pointer): TRttiType; overload;
13     function GetType(AClass: TClass): TRttiType; overload;
14     function GetTypes: TArray<TRttiType>;
15     function FindType(const AQualifiedName: string): TRttiType;
16     function GetPackages: TArray<TRttiPackage>;
17   end;
```

Type Helpers:

Extending classes without inheriting or modifying the class or extending data types. In newer Delphi versions the method *ToString()* can be executed on integer variables to get the numeric value as a string. This type of method was implemented as a type extension and is therefore not available in TMS WEB Core so far; the use of the function *IntToStr()* is still required.

The extension of the transpiler with these language elements is planned and can be expected in the coming years.

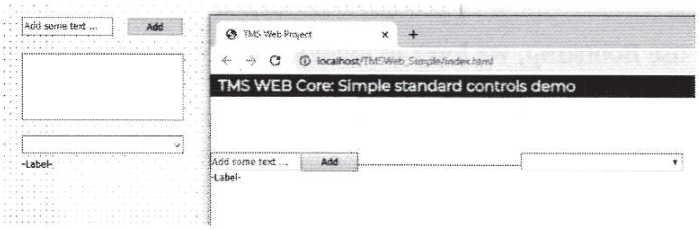
Note

Language elements from JavaScript, which you cannot use in Object Pascal because the transpiler does not know them yet, can be included as JavaScript within your application, if absolutely necessary. Section 3.8.4 on page 70 explains how to use JavaScript directly.

TMS WEB Core is delivered with a special variant of the transpiler. It is not recommended to exchange the transpiler delivered with TMS WEB Core for another version. TMS ensures that all components and the integration into the development environment with the used transpiler work as expected.

The pages are shown on a readable size so you can see what the quality of the text is

the DOM of the browser if desired. For this purpose, the properties *HeightStyle* and *WidthStyle* have to be set to the value *ssAuto*. The other values for this property are explained in section 4.5.5 on page 140.
The order of the components is determined by the *ChildOrder* property.
The following example shows the form from the previous section with relative positioning. The order is not determined, because all components have the value 0 for the *ChildOrder* property. Since there is also no grouping of the components (e.g. using a *TWebPanel*), all elements are arranged according to the standard alignment of the DOM.



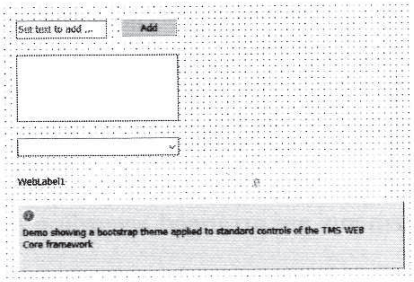
The positioning and scaling can be modified by CSS or HTML, but by using grouping components like *TWebGroupBox* or *TWebPanel* a lot of manual positioning effort in web design can be avoided.

4.5.3 Design with CSS and Bootstrap

Bootstrap[10] is formally a JavaScript library that offers simple design options for all modern web browsers. In particular, the developer can concentrate on defining the graphical elements and the library ensures that it is displayed identically on every web browser. This sounds very complex and also discourages many developers. However, after just a quick look at the documentation, you can see that many things can be used in the library with very little effort. Bootstrap offers numerous predefined layouts that turn a simple web application into a modern web application that can be used on many devices.
The key lies in CSS class definitions, which you can also use directly in TMS WEB Core. Every graphical control in TMS WEB Core has the property *ElementClassName*. It refers to a CSS class that is to be defined or referenced in the associated HTML document.
By defining a class for each of the form elements, we can ensure that our application is visually 'state of the art' and is displayed identically on every web browser, including smartphones and tablets.

[10] URL: `https://getbootstrap.com/docs/4.4/getting-started/introduction/`

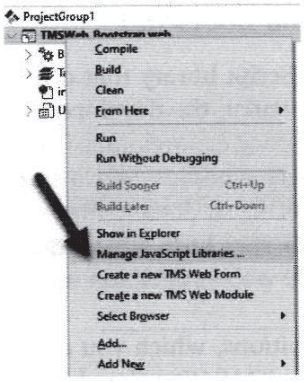
The pages are shown on a readable size so you can see what the quality of the text is



To be able to use Bootstrap, we have to include the library in our HTML file:

```
1 <link rel="stylesheet"
  ↪ href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css">
```

However, we do not have to insert this reference manually. It can be done with the help of a dialog. After all, there is hardly any need to implement HTML or JavaScript - one of the main arguments for application development with TMS WEB Core. Therefore, the context menu of the project manager offers the JavaScript Manager which is accessible using the menu item *Manage JavaScript Libraries ...*.

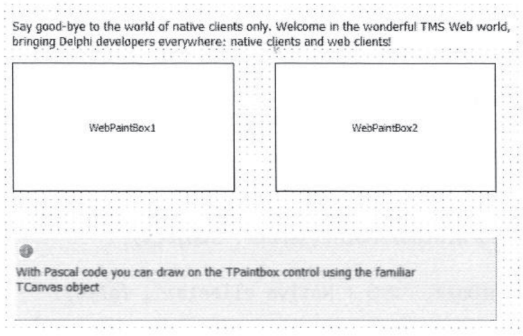


All supported libraries are displayed. If you place a check mark in front of an entry, the corresponding references are inserted. Of course, you can delete the reference from the project at any time by unchecking it. TMS WEB Core remembers which references belong to which entry.
You can still make manual entries in the HTML files if you use the manager. Using the manager does not restrict the direct use of the HTML files.

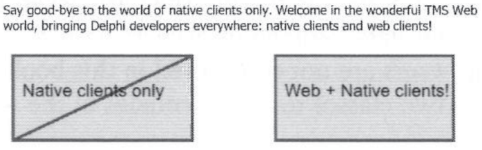
The pages are shown on a readable size so you can see what the quality of the text is

Page views

TMS WEB Core provides the control *TWebPaintBox* accordingly. The properties and events are identical to the VCL variant, so that we will only give an example for use here.
You can find this example in your installation in the directory Demo » Basics » PaintBox.



In the example, two areas on the form are defined for drawing using *TWebPaintBox*. Both components implement the *OnPaint* event (line 21 or 26) and then call the method *PaintSign*. The *PaintSign* method shows an example of how to use the *Canvas* event to draw graphical shapes and output text, similar to the VCL.



This way, you can easily transfer existing drawing operations from your VCL application to your web applications without having to learn how to use JavaScript.

```
1  procedure TForm4.PaintSign(Control: TWebPaintBox; AText: string; Cross:
   ↪ boolean);
2  begin
3    Control.Canvas.Pen.Width := 3;
4    Control.Canvas.Pen.Color := clRed;
5    Control.Canvas.Brush.Color := clYellow;
6    Control.Canvas.Brush.Style := bsSolid;
7    Control.Canvas.Rectangle(10,10,200,100);
8
9    if Cross then
10   begin
11     Control.Canvas.MoveTo(10,100);
```

The pages are shown on a readable size so you can see what the quality of the text is

Page views


```
13      function Add(const AName, AValue: string): TCookie; overload;
14      function Add(const AName, AValue, APath: string): TCookie; overload;
15      function Add(const AName, AValue, APath: string; Expiry: TDateTime):
        ↪  TCookie; overload;
16      property Items[Index: integer]: TCookie read GetItem write SetItem;
17      function Find(const AName: string): TCookie;
18    end;
```

TCookie:

Represents a cookie with name, value, path, and expiration date.

```
1   TCookie = class(TCollectionItem)
2
3     protected
4       property Changed: boolean read FChanged write FChanged;
5     public
6       function CookieAsString: string;
7     published
8       property Name: string read FName write SetName;
9       property Value: string read FValue write SetValue;
10      property Expiry: TDateTime read FExpiry write SetExpiry;
11      property Path: string read FPath write SetPath;
12    end;
```

To access cookies, first create an instance of *TCookies*.

```
1   uses
2      DateUtils;
3
4   procedure TForm1.WebFormCreate(Sender: TObject);
5   var
6     LCookies: TCookies;
7
8   begin
9     LCookies := TCookies.Create;
10    LCookies.Add('user', '4711', Tomorrow);
11    LCookies.SetCookies;
12  end;
```

In the example, the instance is stored in the variable *LCookies* (line 9). Then the methods *Add*, *Find*, and *Delete* can be used. Access to individual cookies is possible via the *Items* property. The number of cookies cannot be queried in Items. The *TCookies* class is derived from the *TCollection* class and thus has the *Count* property, which returns the number of cookies. The example creates a cookie with the name

The pages are shown on a readable size so you can see what the quality of the text is

Page views

user and the value *4711* (line 10). It is valid until the next day. The function *Tomorrow* is part of the unit *DateUtils*. To save the created cookies or changes to existing cookies, the method *SetCookies* must be called (line 11).
To load the existing cookies, proceed as follows:

```
1  procedure TForm1.WebFormCreate(Sender: TObject);
2  var
3    LCookies: TCookies;
4    LCookie: TCookie;
5    i: Integer;
6
7  begin
8    LCookies := TCookies.Create;
9    LCookies.GetCookies;
10
11   Memo.Lines.Clear;
12   for i:= 0 to LCookies.Count -1 do
13   begin
14     LCookie := LCookies.Items[i];
15     Memo.Lines.Add(LCookie.CookieAsString );
16   end;
17 end;
```

The example assumes that a *TWebMemo* component named *Memo* exists in the form class. In the *TWebMemo* information of all cookies is output as a string (line 15) using the method *CookieAsString*. Before that, the instance of *TCookies* is created and assigned to the variable *LCookie* (line 8), analogous to the creation and storage of cookies. The *GetCookies* method reads all cookies (line 9).
Existing content in the *TWebMemo* is deleted (line 11) and each *TCookie* element of the *Items* property is iterated in a *for*-loop (line 14).

Warning

The expiration date of cookies cannot be retrieved.[a] This means that when you create and save a cookie, its validity is saved. However, the date is not read when the web application is restarted.

[a] https://stackoverflow.com/questions/1532193/reading-cookie-expiration-date

4.13 Working with JSON

The JavaScript Object Notation (JSON) data format has become the leading data exchange format on the internet. The reason for this is that its structure makes it

The pages are shown on a readable size so you can see what the quality of the text is

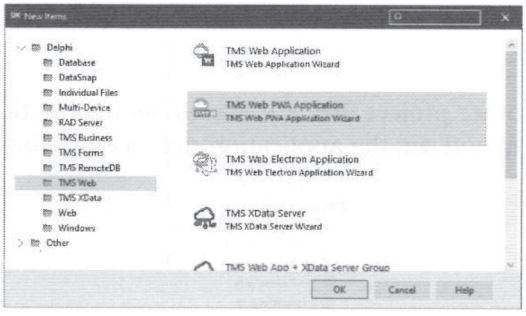
Page views

5.1 Progressive Web Applications

The development of special web applications, the Progressive Web Applications, or PWAs in short, is also possible.[1] PWAs are mainly used because they can be used both online and offline. They also adapt to the conditions of the device on which the web application is used and can be installed as a shortcut like a native application. On a mobile iOS or Android device, one almost has the impression of using a native application.

5.1.1 Creating a PWA

In RAD Studio, PWA applications can be created using a wizard. Via File › New › Other... › TMS Web › TMS Web PWA Application a new project is created.



The project then also contains the necessary files that enable the characteristics that are special to PWA:

IconRes$\{High|Mid|Low\}$**.png**
Icons of the application in different sizes for main screen, bookmarks, browser tab, etc. Replace the PNG files with the graphics for your application. The used platform will then use the icon itself in the required size. To be provided are:

Name	Width	Height
High	512	512
Mid	256	256
Low	64	64

You can customize the file name in the project options (↦ section 5.1.2, page 220).

[1] https://developers.google.com/web/progressive-web-apps/

The pages are shown on a readable size so you can see what the quality of the text is

Page views

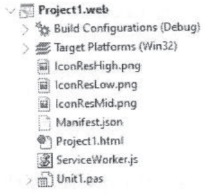
Manifest.json

The so-called manifest of the application. As is now common practice for desktop applications, meta-information required by the operating system is stored in manifest files. TMS WEB Core creates this file for you completely automatically based on the project options and project properties. The manifest contains the name and description of the application as well as information about the icons. It also contains other parameters for configuring the application.

ServiceWorker.js

The 'Service Worker' is a JavaScript component that turns a web application into a progressive web application. The module contains all functions for caching the application for offline use etc. and is automatically generated by TMS WEB Core. The functionality is developed by Google, so that new versions of TMS WEB Core are always provided with the current version from Google.

Needless to say, all other project files are also generated so that you can start designing the main form and run the application right away using default values.



Note

It is recommended to save the entire project immediately after creation. Otherwise, the IDE may have problems finding the icon files etc. After the project has been saved, the paths for the IDE are defined and the icons and other files generated by TMS WEB Core can be found without issues.

5.1.2 Projekt options

The project options for Project > Options... contain further parameters if a PWA is edited in the IDE.

The pages are shown on a readable size so you can see what the quality of the text is

Figure 6.1: Non-visual database components in TMS WEB Core identical to the VCL components *TSQLConnection*, *TClientDataSet* and *TDataSource*. Via *TWebClientConnection* data from any end point can be accessed in JSON encoding via HTTP. *TWebDataSet* provides information about fields and records. The connection to the graphical controls is done via a *TWebDataSource*.

6.1 TWebDataset: A ClientDataset for web applications

Since its first version, Delphi has been characterized by the ability to create database applications with just a few clicks. TMS WEB Core follows the RAD approach and enables the rapid creation of web applications with database access using non-visual components and visual controls.
The understanding of section 4.13.4 on page 190 is required. This section explains how to retrieve and process data from a web service in JSON format. The processing step is now performed for you by the non-visual components. The precondition is that your data source is encoded in JSON format and the individual records (tuples) are stored in a JSON array. This array does not necessarily have to be stored on the first level of the JSON structure. You can specify the path to the data records with a *DataNode*.
The non-visual database components are connected to each other as in the VCL:

$$TWebClientConnection \Leftarrow TWebDataSet \Leftarrow TWebDataSource$$

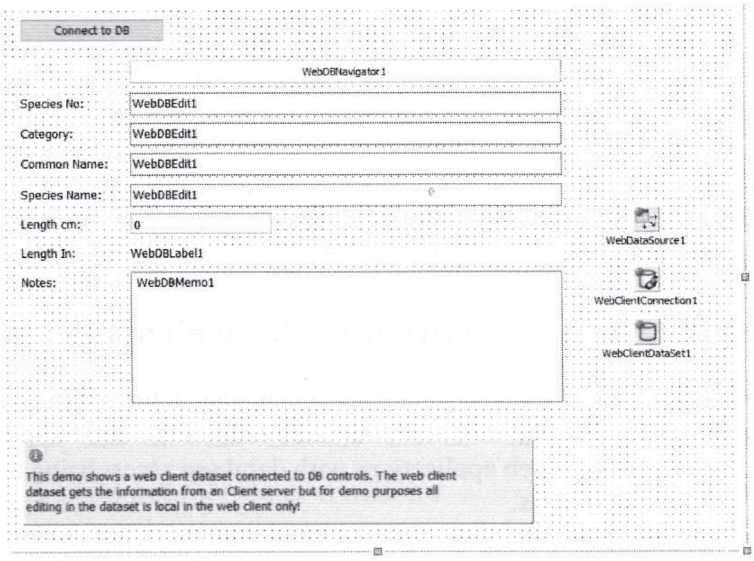
The *TWebClientConnection* establishes the connection to a data source specified by a URL via HTTP. The data records including the field descriptions can be found in the *TWebDataSet*. The connection to the visual controls is established via a *TWebDataSource*.

6.1.1 Using TWebClientConnection

The example presented here can be found in the TMS WEB Core installation in the directory Demo » Basics » Dataset with the project name *TMSWeb_Dataset*.
The main form contains non-visual database components as well as visual components that are connected to the *TWebDataset* via the *TWebDataSource*. The components *TWebDBEdit*, *TWebDBLabel*, *TWebDBMemo* and *TWebDBNavigator* all have a *DataSource* property pointing to *WebDataSource1*. The components behave in the same way as the VCL components of the same name (without "*Web*" in its name).

The pages are shown on a readable size so you can see what the quality of the text is

Page views



As usual, the database components are assigned values at runtime in the source code, so that the relevant properties do not have to be searched for in the object inspector.

```
1  procedure TForm1.WebButton1Click(Sender: TObject);
2  begin
3    WebClientConnection1.URI :=
     ↪  'https://download.tmssoftware.com/tmsweb/fishfacti.json';
4    WebClientConnection1.DataNode := 'ROW';
5
6    WebClientConnection1.Active := true;
7    WebButton1.Enabled := false;
8  end;
```

The endpoint of the web service is specified by the property *URI* (not URL!) (line 3). This is the 'good old' Fish Facts database which is shipped with Delphi since Delphi 1. It has been converted to JSON format:

```
1  "ROW": [
2    {
3      "_Species_No": "90020",
4      "_Category": "Triggerfishy",
5      "_Common_Name": "Clown Triggerfish",
```

The pages are shown on a readable size so you can see what the quality of the text is

Page views

Chapter 7

Examples

In the previous chapters, you have learned about the development of web and desktop applications with TMS WEB Core in detail. Up to now, however, the examples were only brief and always related to a specific topic.
This chapter will now present a cross-technology example application.
In particular, step-by-step descriptions are provided for creating and easily understanding the examples on the following aspects:

- Deploying a database with XData as a web service.
- Accessing the database from a web application.
- Using queries to retrieve subsets from the database.
- Extending the web service to provide data for complex SQL queries.
- Creation of charts with Google Charts.
- User interaction with charts and retrieval of detailed information.
- Integration of modern web designs with HTML, CSS, and Bootstrap independent of the application logic.
- Display of information from the database on a map (Google Maps).

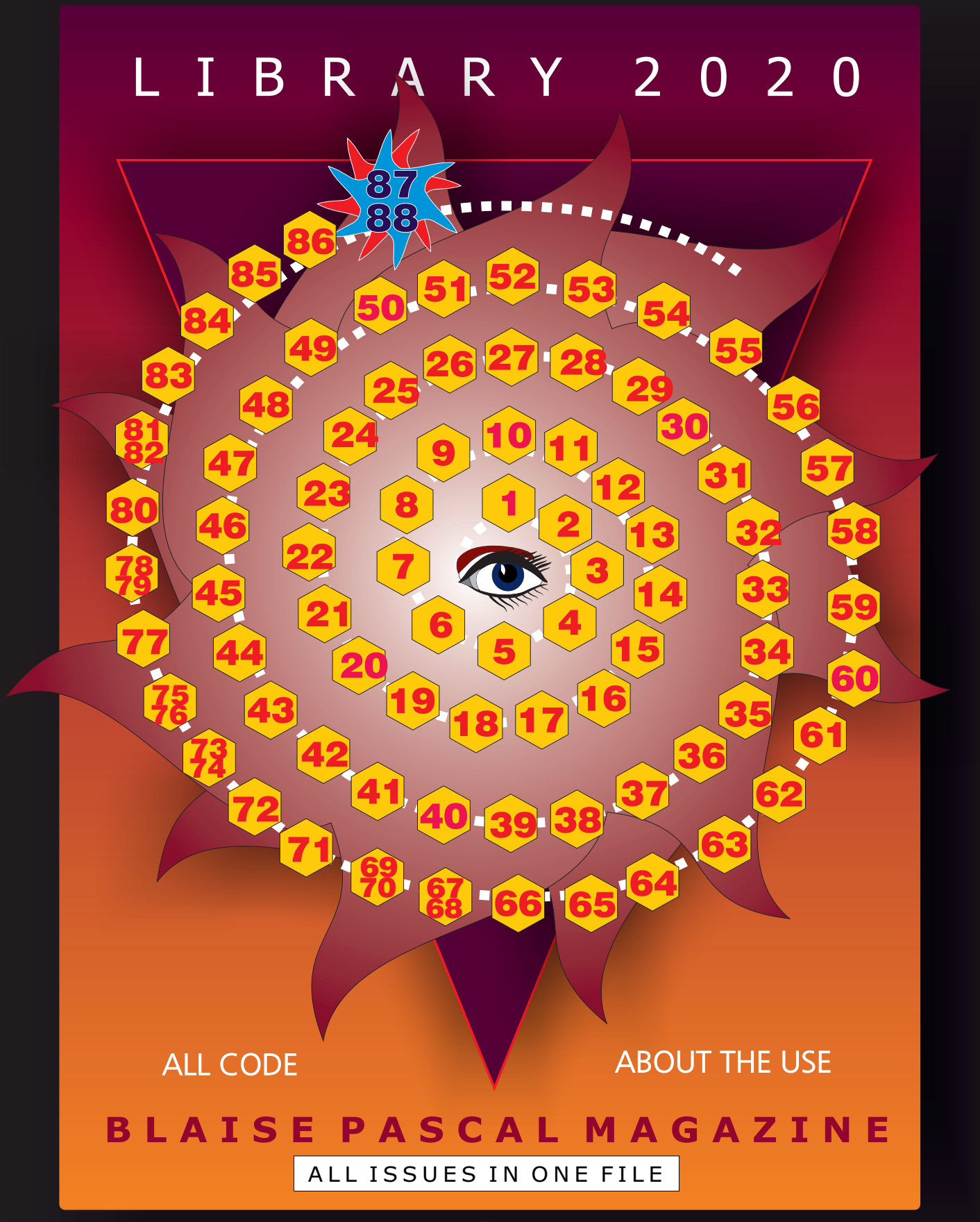
In addition, to presenting an example application, this chapter also serves as a learning check to see whether the aspects from the previous chapters have been fully understood. The theoretical knowledge is consolidated for your future application development by means of concrete examples.

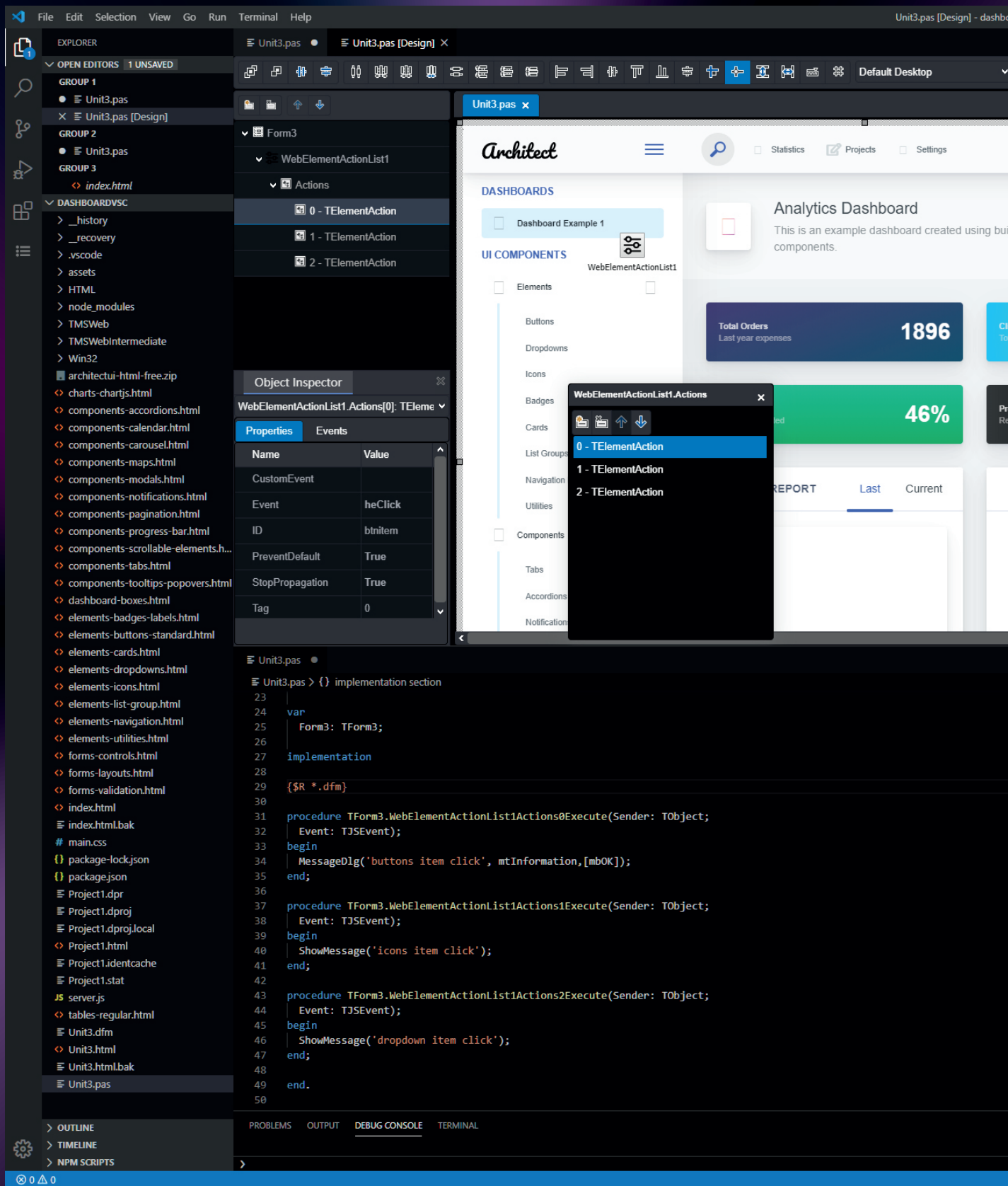
The pages are shown on a readable size so you can see what the quality of the text is

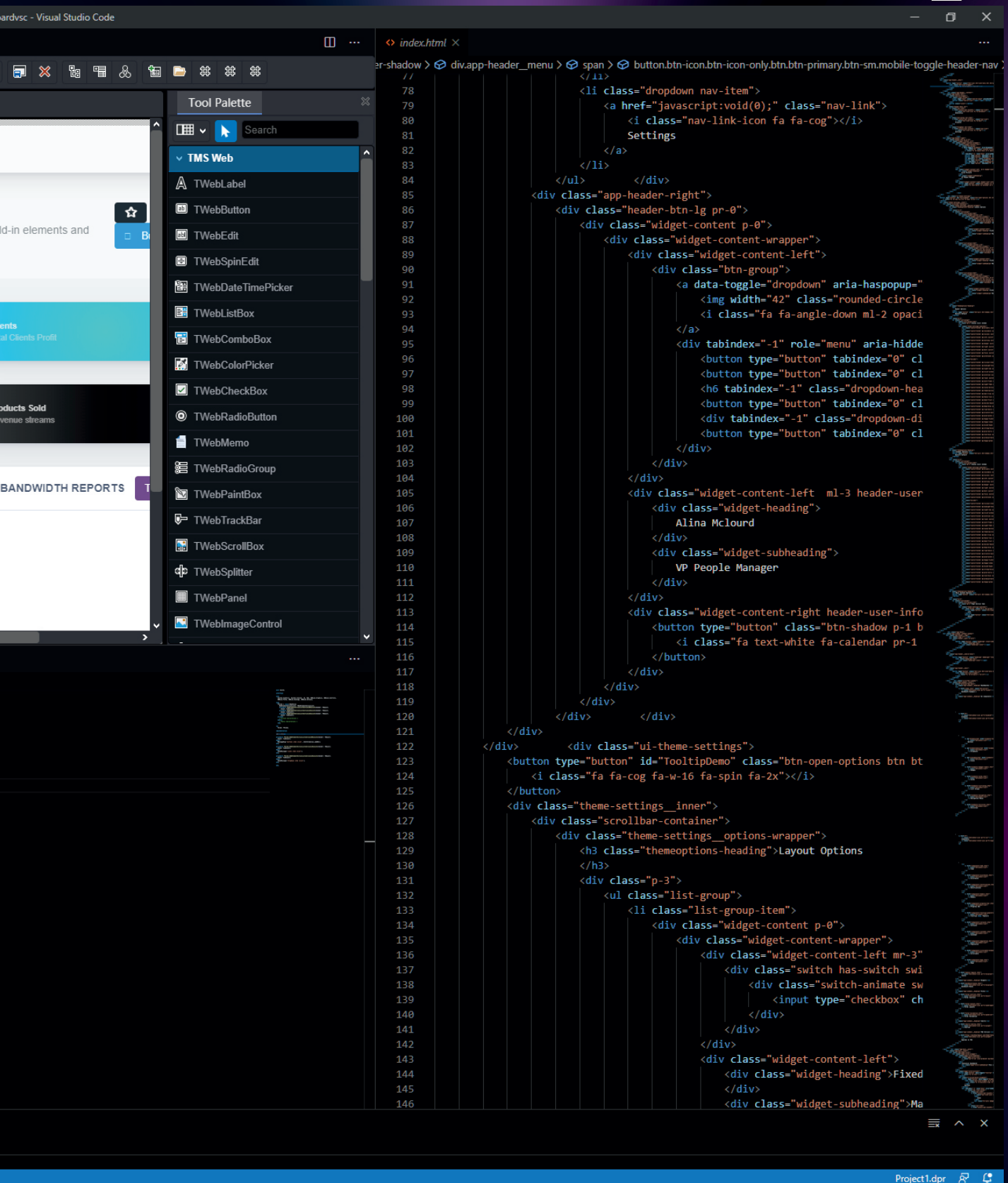
Page views

Contents

The pages are shown on a readable size so you can see what the quality of the text is


LIBRARY 2020
87
88
ALL CODE
ABOUT THE USE
BLAISE PASCAL MAGAZINE
ALL ISSUES IN ONE FILE


File Edit Selection View Go Run Terminal Help
Unit3.pas [Design] - dashbo
EXPLORER
OPEN EDITORS 1 UNSAVED
GROUP 1
Unit3.pas
Unit3.pas [Design]
GROUP 2
Unit3.pas
GROUP 3
index.html
DASHBOARDVSC
_history
_recovery
.vscode
assets
HTML
node_modules
TMSWeb
TMSWebIntermediate
Win32
architectui-html-free.zip
charts-chartjs.html
components-accordions.html
components-calendar.html
components-carousel.html
components-maps.html
components-modals.html
components-notifications.html
components-pagination.html
components-progress-bar.html
components-scrollable-elements.h...
components-tabs.html
components-tooltips-popovers.html
dashboard-boxes.html
elements-badges-labels.html
elements-buttons-standard.html
elements-cards.html
elements-dropdowns.html
elements-icons.html
elements-list-group.html
elements-navigation.html
elements-utilities.html
forms-controls.html
forms-layouts.html
forms-validation.html
index.html
index.html.bak
main.css
package-lock.json
package.json
Project1.dpr
Project1.dproj
Project1.dproj.local
Project1.html
Project1.identcache
Project1.stat
server.js
tables-regular.html
Unit3.dfm
Unit3.html
Unit3.html.bak
Unit3.pas
OUTLINE
TIMELINE
NPM SCRIPTS
Unit3.pas
Unit3.pas [Design]
Default Desktop
Form3
WebElementActionList1
Actions
0 - TElementAction
1 - TElementAction
2 - TElementAction
Object Inspector
WebElementActionList1.Actions[0]: TEleme
Properties
Events
Name
Value
CustomEvent
Event
heClick
ID
btnitem
PreventDefault
True
StopPropagation
True
Tag
0
Unit3.pas
Architect
Statistics
Projects
Settings
DASHBOARDS
Dashboard Example 1
UI COMPONENTS
WebElementActionList1
Elements
Buttons
Dropdowns
Icons
Badges
Cards
List Groups
Navigation
Utilities
Components
Tabs
Accordions
Notifications
Analytics Dashboard
This is an example dashboard created using bui
components.
Total Orders
Last year expenses
1896
46%
REPORT
Last
Current
WebElementActionList1.Actions
0 - TElementAction
1 - TElementAction
2 - TElementAction
Unit3.pas
Unit3.pas > {} implementation section
23
24 var
25 Form3: TForm3;
26
27 implementation
28
29 {$R *.dfm}
30
31 procedure TForm3.WebElementActionList1Actions0Execute(Sender: TObject;
32 Event: TJSEvent);
33 begin
34 MessageDlg('buttons item click', mtInformation,[mbOK]);
35 end;
36
37 procedure TForm3.WebElementActionList1Actions1Execute(Sender: TObject;
38 Event: TJSEvent);
39 begin
40 ShowMessage('icons item click');
41 end;
42
43 procedure TForm3.WebElementActionList1Actions2Execute(Sender: TObject;
44 Event: TJSEvent);
45 begin
46 ShowMessage('dropdown item click');
47 end;
48
49 end.
50
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
0 0


Visual Studio Code
Tool Palette
Search
TMS Web
TWebLabel
TWebButton
TWebEdit
TWebSpinEdit
TWebDateTimePicker
TWebListBox
TWebComboBox
TWebColorPicker
TWebCheckBox
TWebRadioButton
TWebMemo
TWebRadioGroup
TWebPaintBox
TWebTrackBar
TWebScrollBox
TWebSplitter
TWebPanel
TWebImageControl
index.html
Project1.dpr

INTRODUCTION

Object Pascal developers are mainly focused on native Windows application and server development with the VCL framework in Delphi as well as cross platform native application development for iOS, Android, macOS and Windows with the FireMonkey framework in Delphi and the LCL framework in Lazarus.

Due to the significant increase in power and capabilities of the browsers since the introduction of HTML5 and CSS3, the browser itself has become very interesting as a target for rich client application development.
That's why tmssoftware.com has developed the TMS WEB Core framework for about 2 years. This framework allows Delphi and Lazarus developers to do RAD component based rich web client development from their preferred IDE. And now tmssoftware.com introduces support to build TMS WEB Core web client application from Visual Studio Code IDE.

WHAT IS TMS WEB CORE?

TMS WEB Core is a framework and accompanying toolchain support to make optimal use of this framework from an IDE. This framework is written in the **Object Pascal programming language**.

With this code, a web client application can be made thanks to the open-source **pas2js compiler.** The pas2js compiler also comes with a run-time library that closely matches the RTL of Delphi or Lazarus.
Object Pascal users can therefore simply reuse a lot of existing code in a web client application.

The TMS WEB Core framework consists of a visual and non-visual component library that allows to quickly develop web client applications with RAD component based development.

It allows the Delphi or Lazarus developers to work on TMS WEB Core web client applications in much the same way from Visual Studio Code.

This framework has been developed in such a way that it is highly compatible with the VCL or LCL framework and Object Pascal developers are therefore immediately intuitively familiar with the environment.

WHAT IS VISUAL STUDIO CODE?

For software developers unfamiliar with Visual Studio Code, this is a free, open-source, extensible, and platform-independent IDE. This can be downloaded from `https://code.visualstudio.com/`.

This means that the Visual Studio Code IDE works exactly the same on **Windows, macOS** and **Linux** and also comes with full high-DPI support. Visual Studio Code is an initiative of Microsoft and focuses on being open and extensible.
It is open-source and completely free and should not be confused with the regular Microsoft Visual Studio IDE.
Meanwhile, there is a very rich range of extensions for Visual Studio Code with numerous supported programming languages and all kinds of interesting IDE tools, utilities, useful extensions. Currently more than 3000 such extensions can be found at `https://marketplace.visualstudio.com/`.
The latest development of Visual Studio Code is even an online version, so you can also use the IDE from the cloud.

WHAT DOES TMS WEB CORE FOR VISUAL STUDIO CODE OFFER?

At this time of writing and to the best of our knowledge, neither Visual Studio Code nor any of its many extensions offer a **RAD, visual component based form designer** for building applications[2]. Since RAD has always been at the heart of what Delphi or Lazarus means to Object Pascal developers, TMS WEB Core for Visual Studio Code provides a RAD component-based application development tool.

This includes: tool palette, object inspector, structure window, form designer, code window, debugging. In other words, it allows the Delphi or Lazarus developers to work on TMS WEB Core web client applications in much the same way from Visual Studio Code.

WHY THE CHOICE FOR VISUAL STUDIO CODE?

The reason for the efforts to develop a purely TMS WEB Core for Visual Studio Code extension is technology-driven. Visual Studio Code itself is built using web technology. The Visual Studio Code user interface is fully rendered using HTML / CSS / JavaScript through the Chrome browser engine.

This opens the fantastic opportunity to build a form designer based on web technology, which means that on the form designer we can see the TMS WEB Core web components 'live'. Where in the Delphi IDE the form designer is a **VCL based** form designer and during design the web components are simulated as VCL controls, here we have the real web component displayed in the form designer just as it is displayed in the browser when the application is generated.

HERE WE HAVE THE REAL WEB COMPONENT DISPLAYED IN THE FORM DESIGNER JUST AS IT IS DISPLAYED IN THE BROWSER WHEN THE APPLICATION IS GENERATED.

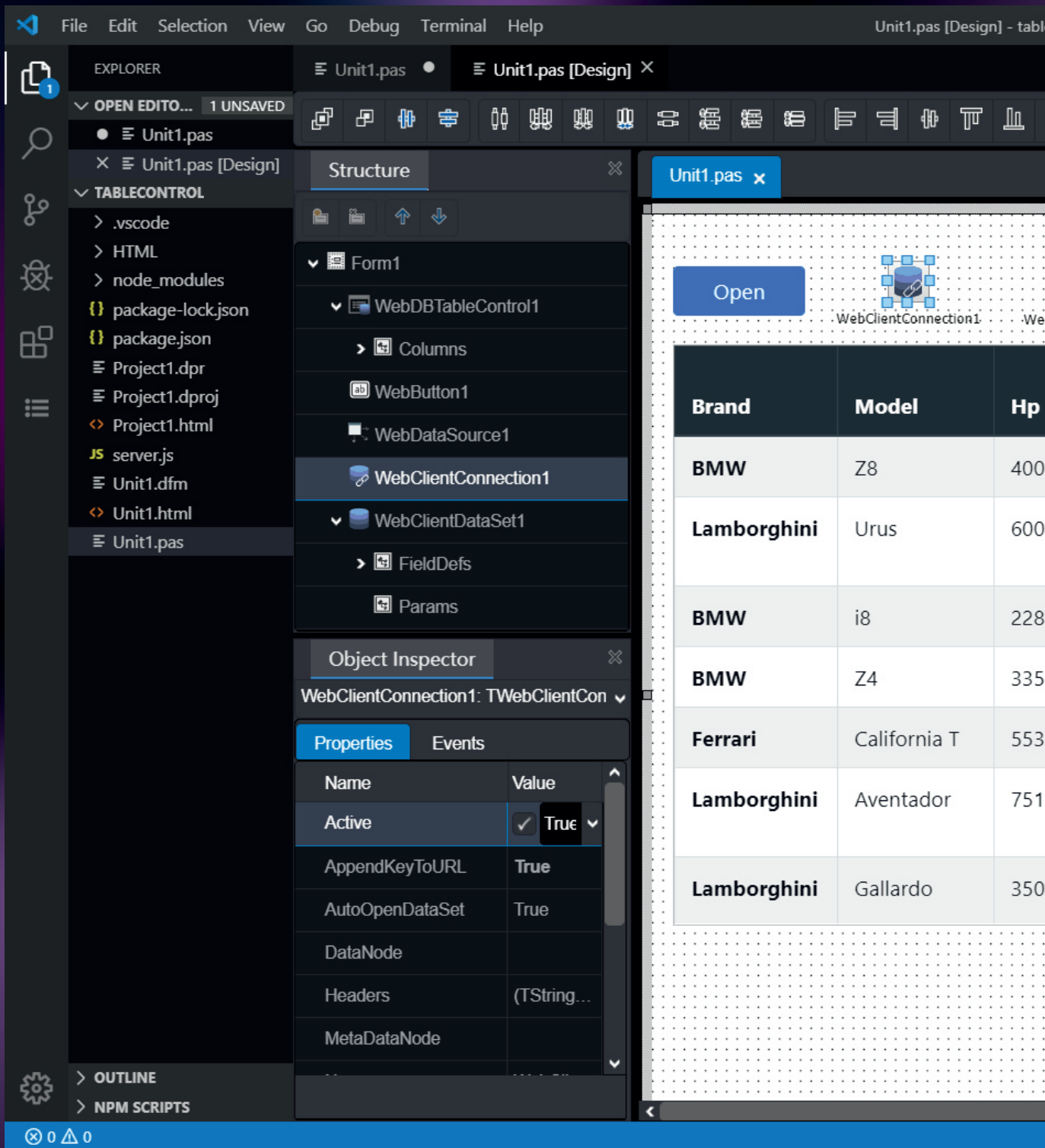
Even during authoring, a CSS library such as **Bootstrap** can be used and it shows the user interface controls on the form designer using the selected Bootstrap classes & themes. The fact that the IDE itself is free, light, fast, modern and runs on Windows, macOS and Linux is of course a fantastic and welcome added benefit. Another not to be overlooked technical advantage is that Visual Studio Code provides browser communication to make debugging Object Pascal code from the IDE easy (as opposed to debugging JavaScript code from the browser console).

REQUIREMENTS?

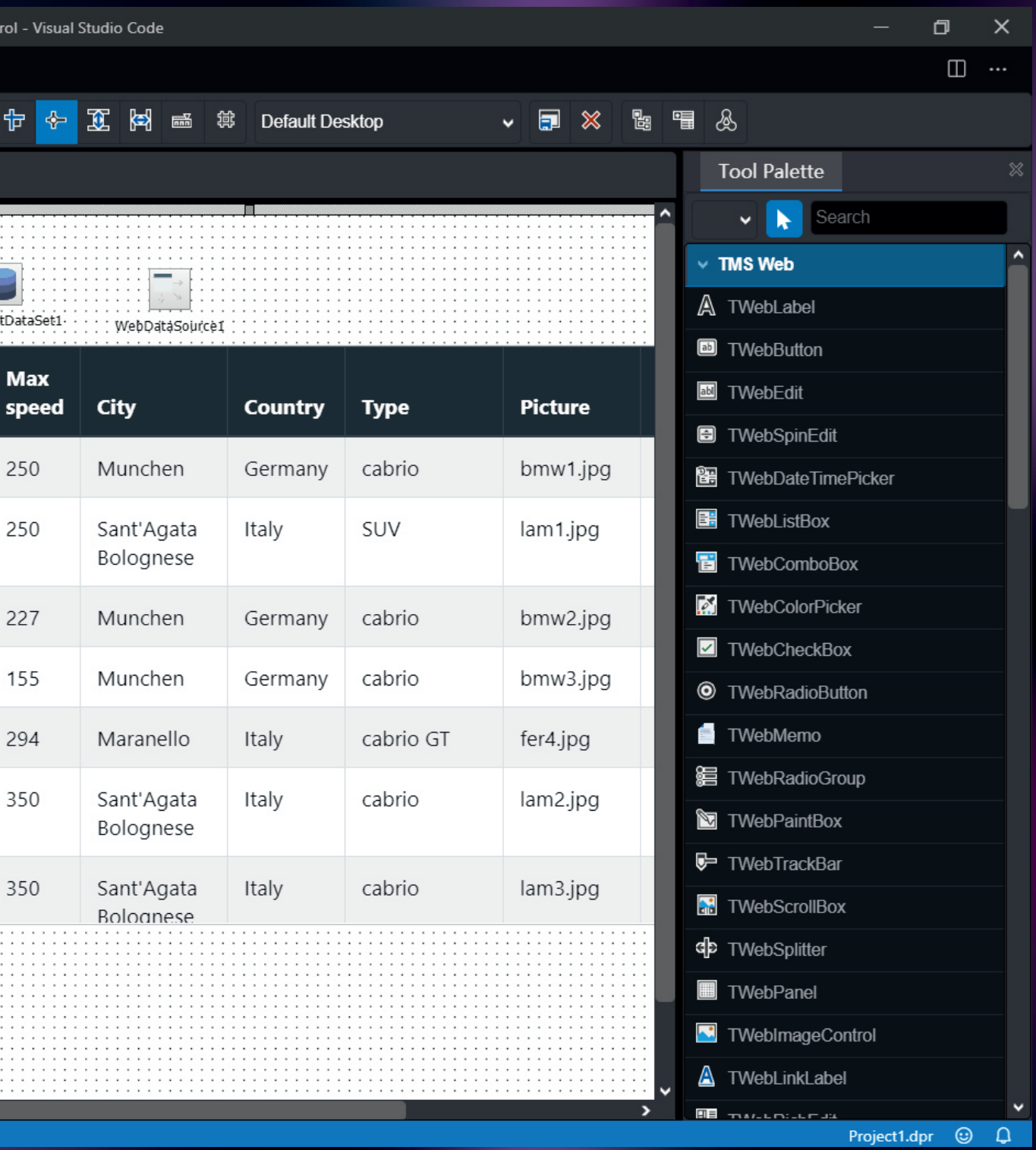
The OmniPascal Visual Studio Code extension (`https://www.omnipascal.com/`) already provides excellent Object Pascal syntax highlighting, code completion, and several other features to facilitate Object Pascal coding.
An interesting side effect is that **OmniPascal** uses internally the Delphi AST engine which was written by Roman Yankovski, TMS FixInsight product manager (`https://www.tmssoftware.com/site/fixinsight.asp`).
Another requirement is of course the pas2js compiler (`https://wiki.freepascal.org/pas2js`). This is the compiler that does the magic of compiling (or transpiling if you prefer this terminology) the Object Pascal code to JavaScript in the browser. This means Delphi or Lazarus developers can write all UI control logic with their beloved strictly typed and object-oriented Object Pascal language. Fortunately, the extension TMS WEB Core for Visual Studio Code has been developed in such a way that both of these prerequisites are added fully automatically in the Visual Studio Code IDE at install-time.

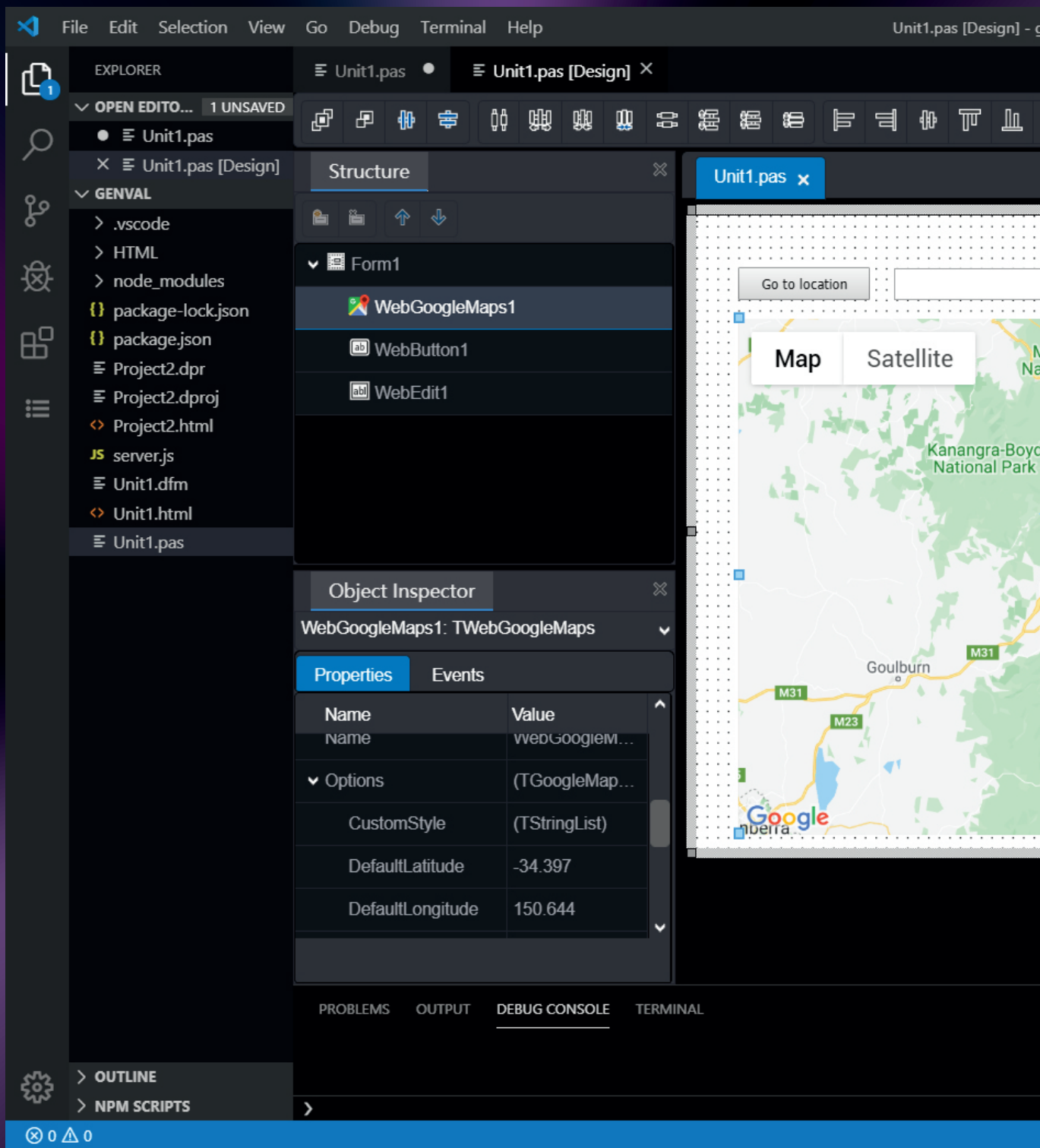
HOW DOES IT COMPARE TO TMS WEB CORE FOR DELPHI OR LAZARUS?

The good news here is that the entire TMS WEB Core framework works unchanged under Visual Studio Code. So there is full parity of the framework feature set between Delphi, Lazarus and Visual Studio Code. Early on, it was decided to make it a development requirement to provide the ability to open Delphi created TMS WEB Core projects from Visual Studio Code and vice versa. Because TMS WEB Core for Visual Studio Code uses the exact same framework code as Delphi, this means that when features or enhancements are added

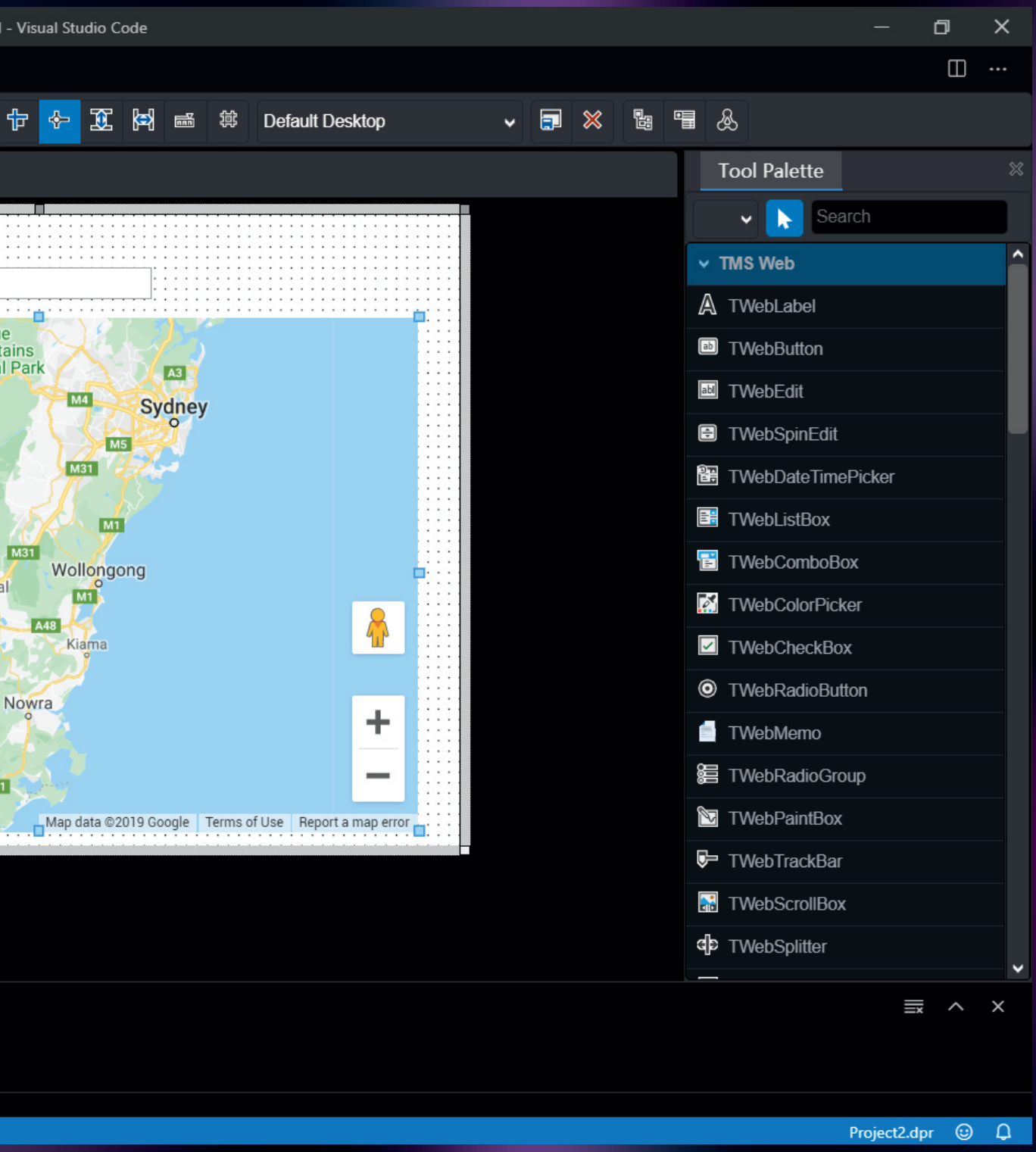


Live data from a server visualized in the form designer


rol - Visual Studio Code
Default Desktop
Tool Palette
Search
TMS Web
TWebLabel
TWebButton
TWebEdit
TWebSpinEdit
TWebDateTimePicker
TWebListBox
TWebComboBox
TWebColorPicker
TWebCheckBox
TWebRadioButton
TWebMemo
TWebRadioGroup
TWebPaintBox
TWebTrackBar
TWebScrollBox
TWebSplitter
TWebPanel
TWebImageControl
TWebLinkLabel
tDataSet1
WebDataSource1
Max speed
City
Country
Type
Picture
250 Munchen Germany cabrio bmw1.jpg
250 Sant'Agata Bolognese Italy SUV lam1.jpg
227 Munchen Germany cabrio bmw2.jpg
155 Munchen Germany cabrio bmw3.jpg
294 Maranello Italy cabrio GT fer4.jpg
350 Sant'Agata Bolognese Italy cabrio lam2.jpg
350 Sant'Agata Bolognese Italy cabrio lam3.jpg
Project1.dpr



The embedded Google Maps control live in the form designer


- Visual Studio Code
Default Desktop
Tool Palette
Search
TMS Web
TWebLabel
TWebButton
TWebEdit
TWebSpinEdit
TWebDateTimePicker
TWebListBox
TWebComboBox
TWebColorPicker
TWebCheckBox
TWebRadioButton
TWebMemo
TWebRadioGroup
TWebPaintBox
TWebTrackBar
TWebScrollBox
TWebSplitter
Sydney
Wollongong
Kiama
Nowra
A3
M4
M5
M31
M1
A48
Map data ©2019 Google
Terms of Use
Report a map error
Project2.dpr

to the Delphi product, it will be automatically carried over to the Visual Studio Code version. The reverse is of course already true. An important difference, however, is that Visual Studio Code works according to the concept: one project is one folder.
This means that you can only manage one TMS WEB Core project within a folder with Visual Studio Code. This in itself is a minor limitation as most developers work the same way from Delphi or Lazarus.

WHAT DOES THIS MEAN FOR TMS WEB CORE FOR DELPHI?

TMS WEB Core for Visual Studio Code does not replace TMS WEB Core for Delphi. It's nothing more and nothing less than an additional IDE choice. So each developer can decide for themselves which IDE they prefer: Delphi, Lazarus or Visual Studio Code. The fact that Visual Studio Code & Lazarus are platform independent, i.e. can be used directly from macOS or Linux, can of course be a deciding factor. Furthermore, Delphi or Lazarus is used for much more than just developing TMS WEB Core web client applications, for example the development of a TMS XData or Embarcadero RAD server REST back-end.

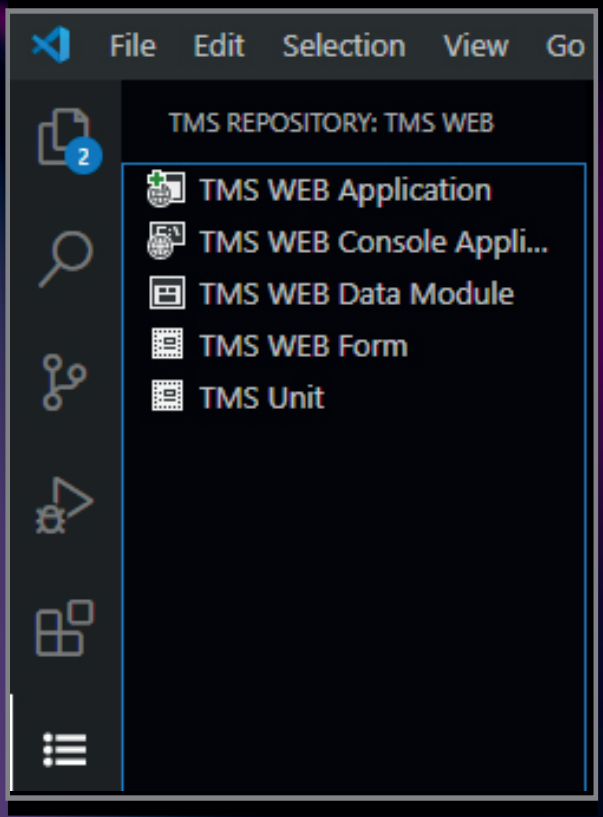
WHAT IS STANDARD INCLUDED IN TMS WEB CORE?

The basic TMS WEB Core framework already comes with a whole set of visual and non-visual components. All standard user interface controls such as an edit, checkbox, memo, button, listbox, grid, treeview, menu etc... are all present. The visual controls are built in such a way that their programming interface matches the VCL equivalents as closely as possible. For example, there is TWebEdit which is the equivalent for the VCL TEdit and therefore also has a TWebEdit.Text: string property. Or so there is TWebStringGrid as equivalent to TStringGrid where similarly there is a property TWebStringGrid.Cells [Col, Row: integer]: string for access to the cell data.

DOES TMS WEB CORE ALSO WORK WITH VISUAL STUDIO CODE ONLINE?

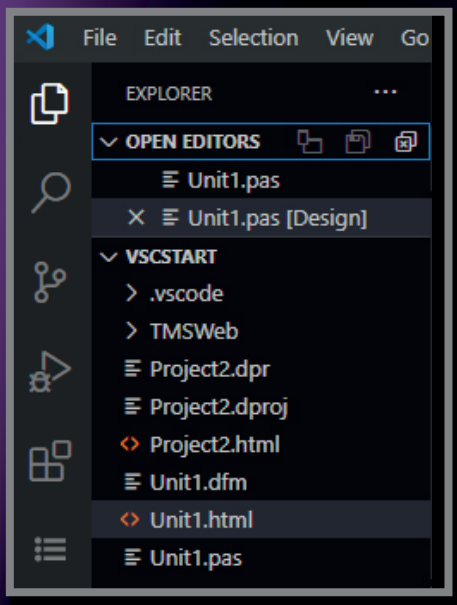
Visual Studio Code Online is currently in beta and promises to bring the IDE to the cloud. An IDE that you can easily use from the browser. Any machine with a browser can be used anywhere, anytime to develop, develop, develop ... (I hear Steve Ballmer here somewhere).
TMS WEB Core has been tested with Visual Studio Code Online and it is also possible to create web client applications for the browser from the browser. The circle is also closed in this area.

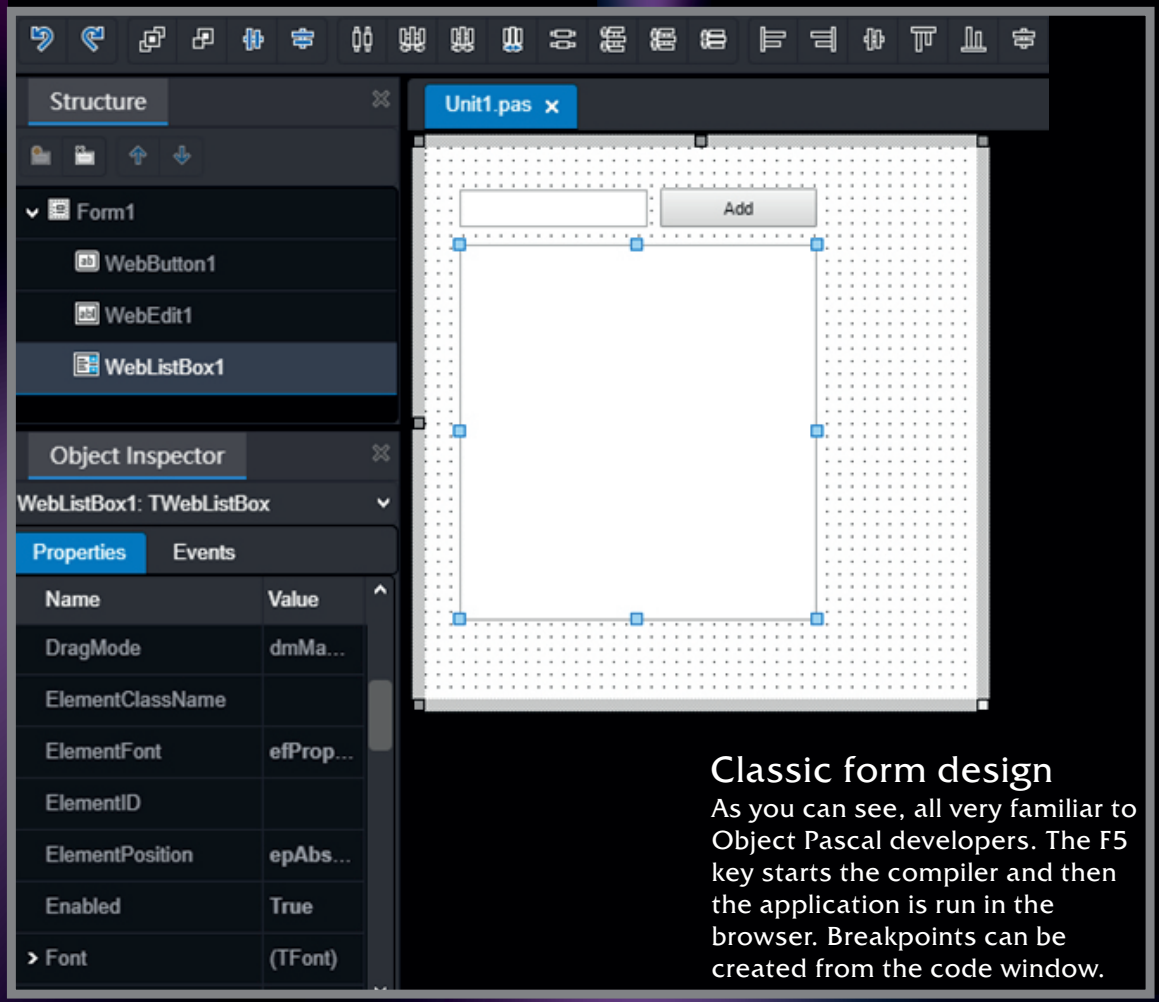
Get started with TMS WEB Core for Visual Studio Code
After installing TMS WEB Core in Visual Studio Code, the user gets an extra icon in the activity bar (left side IDE). This opens the TMS Repository from where you can choose to start a new project or add a new form, data module or unit to a project.



After creating a project, the files included in this project also appear on the left in the Explorer (just like for other Visual Studio Code projects). For a TMS WEB Core application this means the project .DPROJ, .DPR and .HTML file as well as the .PAS, .DFM and .HTML file of a first form within this project. This is an identical project structure to the one used in the Delphi IDE or Lazarus IDE.

When opening the source code of the first form unit1.pas, the code appears. The form designer is activated with the shortcut Ctrl + F12. This gives you the structure view and object inspector to the left of the form designer and the tool palette to the right. You can then drag components onto the form and use the object inspector to change properties and add events.





Classic form design

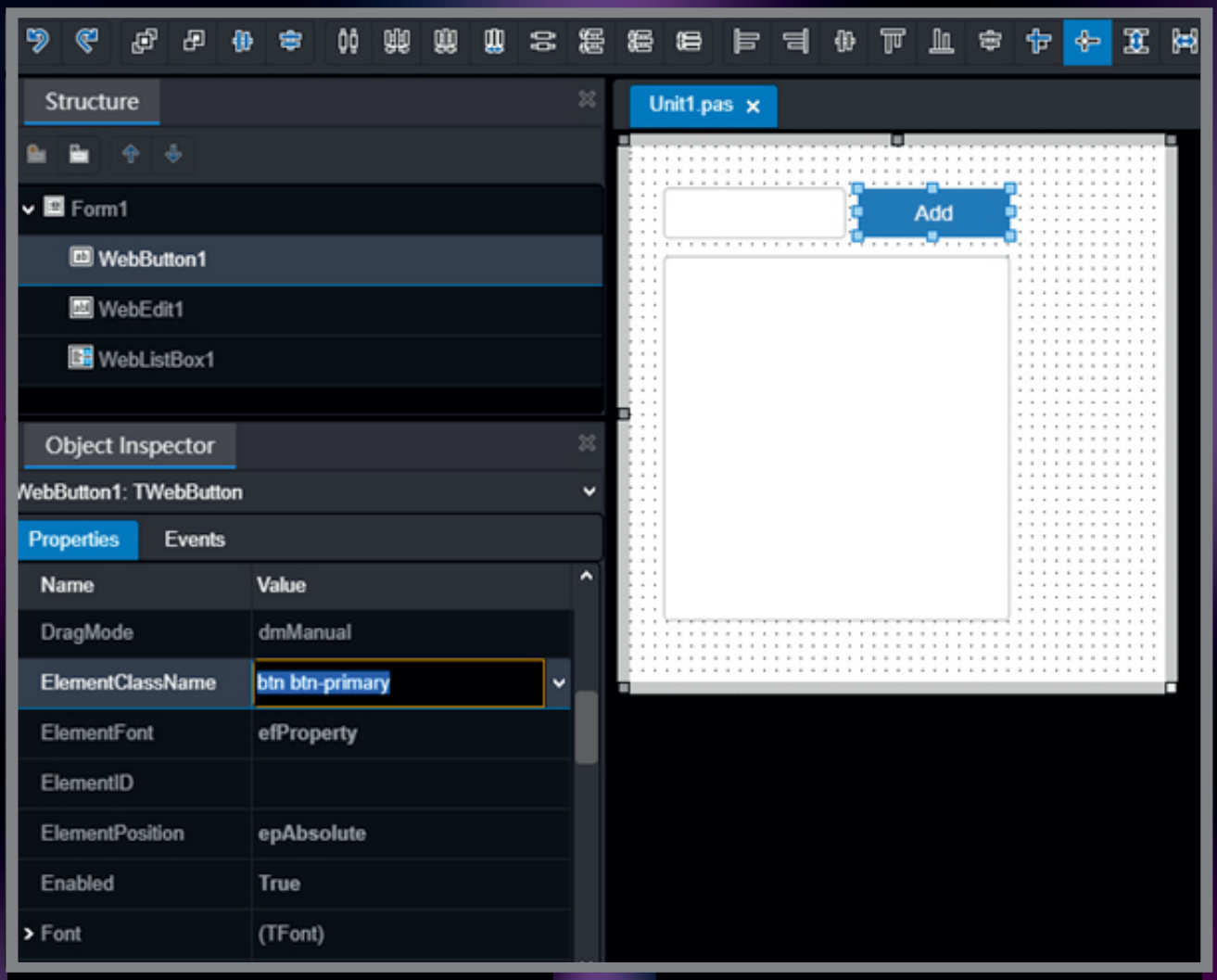
As you can see, all very familiar to Object Pascal developers. The F5 key starts the compiler and then the application is run in the browser. Breakpoints can be created from the code window.

Unit1.pas | Unit1.pas [Design] | Unit1.html | Project2.html

Unit1.pas > {} interface section

```
1   unit Unit1;
2
3   interface
4
5   uses
6     System.SysUtils, System.Classes, JS, Web, WEBLib.Graphics, WEBLib.Controls,
7     WEBLib.Forms, WEBLib.Dialogs, WEBLib.StdCtrls;
8
9   type
10    TForm1 = class(TWebForm)
11      WebButton1: TWebButton;
12      WebEdit1: TWebEdit;
13      WebListBox1: TWebListBox;
14      procedure WebButton1Click(Sender: TObject);
15    private
16      { Private declarations }
17    public
18      { Public declarations }
19    end;
20
21  var
22    Form1: TForm1;
23
24  implementation
25
26  {$R *.dfm}
27
28  procedure TForm1.WebButton1Click(Sender: TObject);
29  var
30    s: string;
31  begin
32    s := WebEdit1.Text;
33    WebListBox1.Items.Add(s);
34  end;
```

So far this is all very similar to building a Windows application with the VCL or LCL framework. The moment you start using templates, the power of using web design (HTML / CSS) opens up for TMS WEB Core. To illustrate this, in a first step we can start by including the Bootstrap library for styling the application. This is done by adding a link in the project HTML and then specifying the CSS class to be used on the controls via the ElementClassName property. In this simple example, we put the "form-control" CSS class on TWebEdit and TwebListBox and the "btn btn-primary" class on TWebButton.



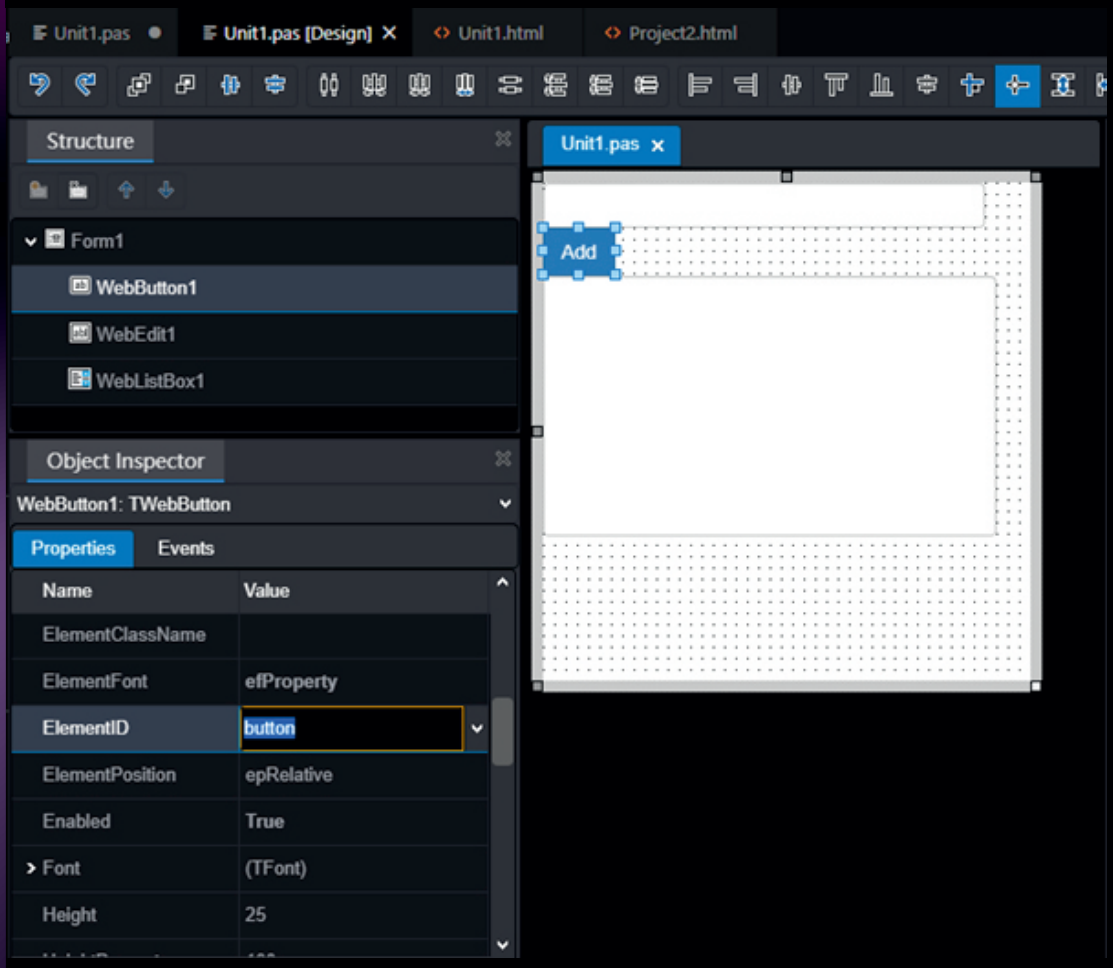
Claissik form design but with Bootstrap styling

In a next step we will again move away from the classic VCL form design and specify the layout of the form fully in HTML. Finally, to connect the user-interface logic to the HTML, we link the user-interface controls to the HTML elements by setting in the object inspector the control.ElementID property to the ID of the HTML elements. The HTML for the form will be:

HTML template code for the form

```
Unit1.html > ...
1   <html>
2   <head>
3     <meta http-equiv="Content-type" content="text/html; charset=utf-8" />
4     <title>TMS Web Project</title>
5     <style>
6     </style>
7   </head>
8   <body>
9     <div class="row">
10    <div class="col-md-6"><input class="form-control" type="text" id="edit"></input></div>
11    <div class="col-md-6"><button class="btn btn-primary" id="button">Add</button></div>
12    </div>
13    <div><select class="form-control" id="listbox" size="8"></select></div>
14  </body>
15  </html>
16
```

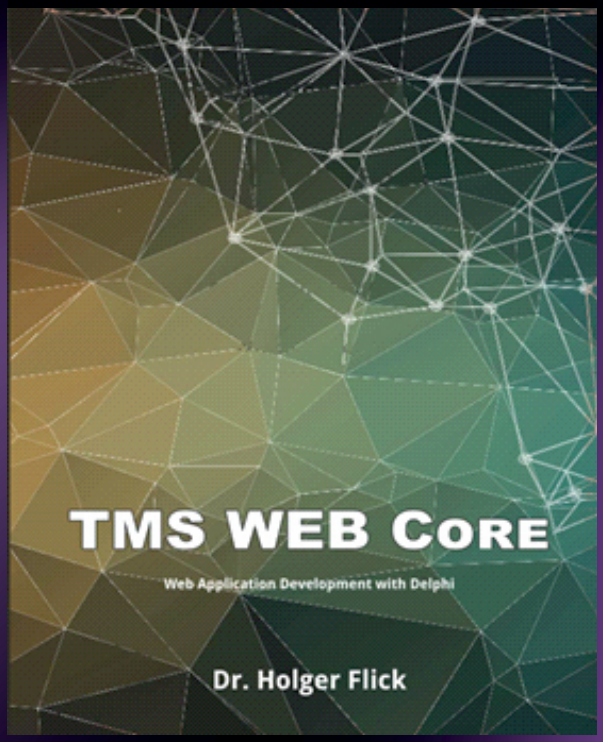
Connected to the control on the form designer this looks like in the IDE:



Layout fully controlled by HTML / CSS and UI logic connected in form designer

Here we see WebButton1 linked to the HTML button element via ElementID "button". **Bootstrap** classes have been used for the DIV elements that immediately give this form a basic responsive design.

If the browser window is wide enough, the DIV elements for the HTML INPUT control linked here to TWebEdit1 and the HTML BUTTON will be in one row.
For a smaller screen, it automatically switches to column display. In other words, there is a separation between the layout of the form (via HTML / CSS) and the code in **Object Pascal.** When wanted, a template for the application could have been purchased or a web designer could have independently from the application developer created the form layout.
Of course, in this introductory article, we only scratched the surface of the possibilities of TMS WEB Core.
TMS WEB Core now comes with almost 100 different demos, dozens of videos and there is also a book on the market that describes the framework: see page 62 where the book is described in depth



https://www.amazon.com/TMS-WEB-Core-Application-Development/dp/B086G6XDGW

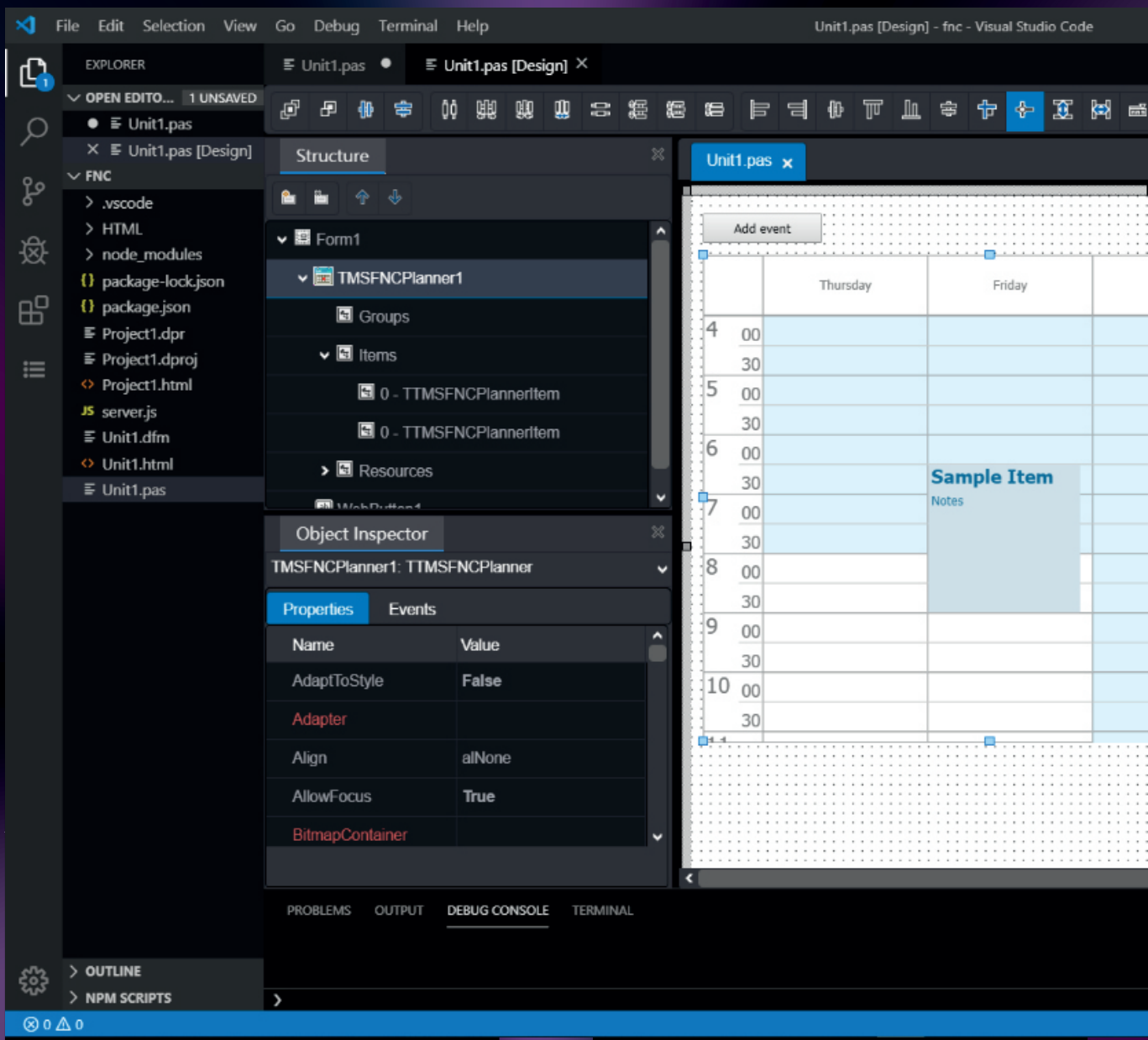
or you can get started by watching an online video series

https://www.youtube.com/playlist?list=PLaczILq2zKMm3xBTth2AE7CKFWGj8DLvs

ROADMAP

v1.0 is clearly the first step, the foundation for developing web client applications with the TMS WEB Core framework in Visual Studio Code. There are already many further developments in the pipeline. Some of the major and already publicly announced features for subsequent versions of TMS WEB Core for Visual Studio Code are:

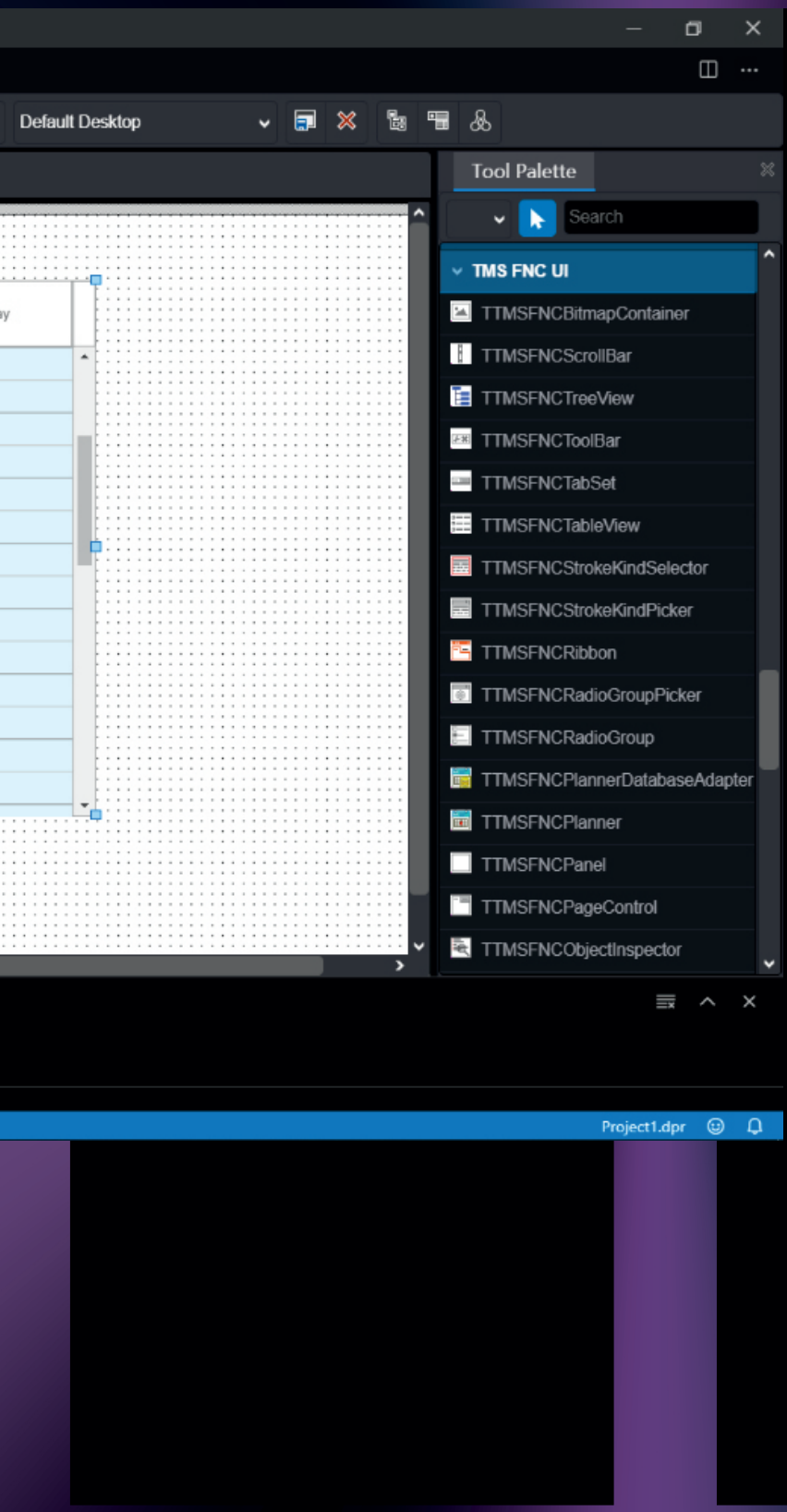
- Built-in support for creating PWA projects
- Built-in support for creating cross-platform desktop applications with the Electron framework
- Support for installable 3rd party components including the full TMS FNC libraries of cross-platform / cross-framework components
- Support for new Object Pascal language properties
- Templates and wizards for automatic application generation



A look into the future, TMS FNC Planner on the Visual Studio Code form designer

CONCLUSION

Visual Studio Code is a relatively young and web technology based IDE. It was a technically logical and obvious choice to build the necessary support to integrate the TMS WEB Core framework for RAD component based web client development. With version 1.0 the first step has been taken and further developments are being made to expand this into an ultra productive and rich platform.


Default Desktop
Tool Palette
Search
TMS FNC UI
TTMSFNCBitmapContainer
TTMSFNCScrollBar
TTMSFNCTreeView
TTMSFNCToolBar
TTMSFNCTabSet
TTMSFNCTableView
TTMSFNCStrokeKindSelector
TTMSFNCStrokeKindPicker
TTMSFNCRibbon
TTMSFNCRadioGroupPicker
TTMSFNCRadioGroup
TTMSFNCPlannerDatabaseAdapter
TTMSFNCPlanner
TTMSFNCPanel
TTMSFNCPageControl
TTMSFNCObjectInspector
Project1.dpr

BOOKS: LAZARUS HANDBOOK

Authors:

Michaël van Canneyt - Main Author.
Martin Friebe
Mattias Gärtner
Inoussa Ouedraogo
Detlef Overbeek
Howard Page Clark
Werner Pamler

Title: Lazarus Handbook

Book

The book is created in three versions:
PDF (Electronic),
HardCover printed and sewn, linen cover and white paper cover for protection.
Softcover printed and sewn.
The luxurious version (hardcover has also two ribbons).
This edition is registered by the Royal Dutch Library (Nederlandse Koninklijke Bibliotheek}

ISBN:

978-94-90968-02-1 for the PDF version
978-94-90968-13-7 for the Hardcover version
978-94-90-968-97-7 for the Softcover version

You can order the book on the website of BLaise Pascal Magazine:
`https://www.blaisepascalmagazine.eu/product-category/books/`

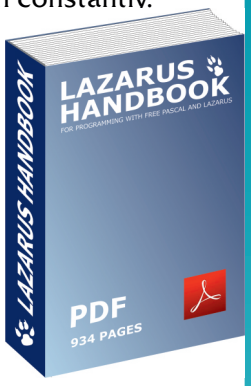
The book is of great value not only for the reader but also for the community.
Half of the proceeds are intended for the community, which means that projects to develop the language can be financed.
As a member of the community you can suggest new items for development.

The book has been developed during the last 5 years and is for the moment very much up to date. The latest developments are handled in it.
For Pas2Js however it was impossible to integrate. That item will because of its enormous size, impact and possibilities become a title in its own. We will start with the PDF file first. Pas2JS is also to be used for the Pascal language and therefore also available for Delphi.

The book is a technical reference as well explanatory and heavily illustrated. For quite a number of items code is also available.
The latest version of Lazarus and FPC is used: 2.0.10 and 3.02.

You receive together with the book always a PDF file so that you can use it in conjunction with the chapter overview – you can simply click on an item and will land on the corresponding page, as wel you can use the PDF Index in the last part of the book. Page numbers are co-respondent.

Since the language is subject to constant development, new items and extensions we will update the PDF version constantlv.
The first year all updates of the PDF are available for free. After that we will offer you a yearly upgrade subscription.



Please send your notifications to
`Admin@BlaisePascalMagzine.eu`

Chapter Overview Lazarus Handbook

Chapter Overview Lazarus Handbook

Chapter Overview Lazarus Handbook

Chapter Overview Lazarus Handbook

Chapter Overview Lazarus Handbook

Introduction

These days INSTALLING LAZARUS has become really easy, unlike the struggles that were common when installing early versions of LAZARUS. Installers are available for all major platforms, and unless you need something special you can install LAZARUS in no time.

Your first step in installing LAZARUS is to download it from the LAZARUS WEBSITE:

`http://www.lazarus-ide.org/`

On the main page, you can see a big DOWNLOAD NOW button. Your browser

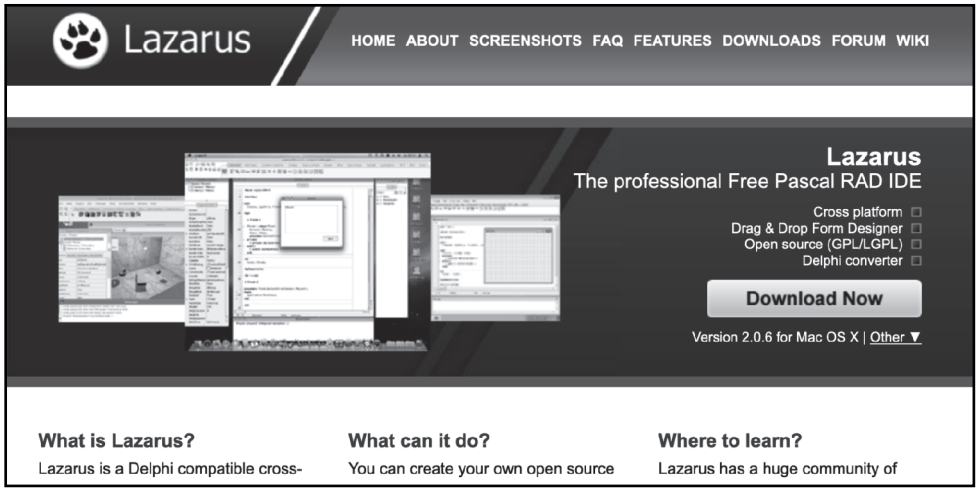


Figure 1: The download website

Clicking the download button or selecting another version takes you to the SourceForge download pages. Depending on the choice you made, either the download starts at once, or you may need to choose further files.

What to do next varies according to the platform you are installing on. The following sections discuss specific steps needed for each major supported platform: WINDOWS, LINUX and MAC.

1.3 Installing Lazarus on macOS

by Detlef Overbeek

1. INSTALLING

Installing LAZARUS on a MAC is not particularly difficult but it is critical that you perform the various installation steps outlined below in the correct order. If you skip steps, or do them out of order you will almost certainly be disappointed.

The detailed steps to follow are listed below, and assume you have a recent version of MACOS, a recent version of XCODE from APPLE, and a recent LAZARUS version.

These instructions apply to both the **Carbon** and **Cocoa** widgetsets. While the **Carbon LCL** implementation may still be seen as slightly more stable, as of

1.1 INSTALLING XCODE, IF YOU DO NOT HAVE IT INSTALLED

http://wiki.lazarus.freepascal.org/Installing_Lazarus_on_MacOS_X

You need the APPLE DEVELOPER tools, which are a part of the XCODE development environment. If you do not have XCODE installed, your first step is to download and install XCODE.

If you do not have them installed, these tools can be installed from the original MACOS installation disks (in which case you will then need an update to obtain the very latest tools version). Or you can download the tools from the **Apple Developer Connection (ADC)**, which requires free registration:

http://developer.apple.com/

When you download the XCODE archive, it should end up in your **Downloads** directory as a **.zip** file. Right-click the **.zip** to un-archive the files into your **Downloads** directory.
You may be happy with XCODE there, but perhaps not, since other users will see the path
to it but be unable to use it. Because of its unsatisfactory location in **Downloads**,
you can move it, and then inform XCODE-SELECT where you moved it to.
In a terminal type these commands:

```
mv Downloads/Xcode-beta.app /Developer/
```

6 Debugging

by Martin Friebe

AN OVERVIEW OF DEBUGGING

THE BASIC SET-UP REQUIRED TO DEBUG

Before you can successfully debug projects the Lazarus debugger must be configured correctly. The original Lazarus installation usually sets up the debugger correctly. Nevertheless, it is prudent to check that your set-up and configuration conforms to the basic requirements given below. The need for this arises because Lazarus currently (at version 2.0) makes use of an external tool (the GDB executable, GNU DeBugger) as its debugging engine. For the Apple Macintosh LLDB can be used as an alternative. See the setup chapter for Mac OS X on Chapter page *35*.

Setting the path to GDB

When you start Lazarus for the first time the IDE tries to detect GDB's presence and its path setting. If GDB is not found a start-up prompt is shown indicating the problem. In that case you will need to specify GDB's location yourself. The GDB path is OS-dependent. The Windows Lazarus installer installs GDB in your installation's `mingw` folder. On Linux the OS will usually have installed GDB in `/usr/bin/gdb`. A few Linux distros may require you to install GDB as an additional package. GDB installation on Macintosh and iOS requires extra work. The section on Initial set-up for Mac OS X near the end of this chapter has full details of the steps needed.

Setting project options

Every Lazarus project has its own Project Options. Certain options in the Project Options dialog (Project → Project Options..., Ctrl+Shift+F11) must be set appropriately for the debugger to work with your project. Click the Debugging node to open the Debugging page. Here you must check Generate info for the debugger.
Type of debug info can be set to any of Automatic, Stabs or Dwarf with sets.
Click the Compilation and Linking node to open that page.
Optimization levels must be set to 0 or 1 (no higher).
Under Linking, Link smart (-XX) must be unchecked.

THE MEANING OF THE GUTTER SYMBOLS FOR DEBUGGING

When using the debugger, the editor's left gutter shows the current debug status of each line in your program's source code.

The executable line marker (small blue dot)

Once your application is started in the debugger, a small blue dot appears to the left of each executable line in the source. This blue dot signifies that executable code has been generated for this line, and that the unit was compiled with debug information included.

Note that including debug information will increase the size of the executable file considerably. (See also Use external gdb debug symbols file (-Xg) on this chapter page *50*).

6 Debugging

by Martin Friebe

AN OVERVIEW OF DEBUGGING

THE BASIC SET-UP REQUIRED TO DEBUG

Before you can successfully debug projects the Lazarus debugger must be configured correctly. The original Lazarus installation usually sets up the debugger correctly. Nevertheless, it is prudent to check that your set-up and configuration conforms to the basic requirements given below. The need for this arises because Lazarus currently (at version 2.0) makes use of an external tool (the GDB executable, GNU DeBugger) as its debugging engine. For the Apple Macintosh LLDB can be used as an alternative. See the setup chapter for Mac OS X on Chapter page *35*.

Setting the path to GDB

When you start Lazarus for the first time the IDE tries to detect GDB's presence and its path setting. If GDB is not found a start-up prompt is shown indicating the problem. In that case you will need to specify GDB's location yourself. The GDB path is OS-dependent. The Windows Lazarus installer installs GDB in your installation's `mingw` folder. On Linux the OS will usually have installed GDB in `/usr/bin/gdb`. A few Linux distros may require you to install GDB as an additional package. GDB installation on Macintosh and iOS requires extra work. The section on Initial set-up for Mac OS X near the end of this chapter has full details of the steps needed.

Setting project options

Every Lazarus project has its own Project Options. Certain options in the Project Options dialog (Project → Project Options..., Ctrl+Shift+F11) must be set appropriately for the debugger to work with your project. Click the Debugging node to open the Debugging page. Here you must check Generate info for the debugger.
Type of debug info can be set to any of Automatic, Stabs or Dwarf with sets.
Click the Compilation and Linking node to open that page.
Optimization levels must be set to 0 or 1 (no higher).
Under Linking, Link smart (-XX) must be unchecked.

THE MEANING OF THE GUTTER SYMBOLS FOR DEBUGGING

When using the debugger, the editor's left gutter shows the current debug status of each line in your program's source code.

- **The executable line marker** (small blue dot)

Once your application is started in the debugger, a small blue dot appears to the left of each executable line in the source. This blue dot signifies that executable code has been generated for this line, and that the unit was compiled with debug information included.

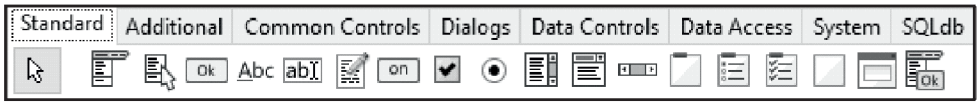
Original, DB and RTTI components

Many of the simpler controls in the LCL have DB and RTTI counterpart controls, and so are supplied in three flavours. For instance, as well as `Tedit`, there is also a `TDBEdit` and a `TTIEdit`. As well as a `TLabel` there is a `TDBLabel` and a `TTILabel`, and so on. The DB controls extend the standard controls with a built-in ability to interchange data with a database.

The RTTI controls extend the standard controls with a built-in ability to exchange data with some linked object's published property. The DB controls are grouped together on the Data Controls page of the component palette, and the RTTI controls are grouped together on their own RTTI page.

THE STANDARD PAGE

The Standard palette page holds the most frequently used visual controls and one non-visual component. It is similar to the corresponding Standard tab in Delphi.



TMainMenu

The main menu appears at the top of the main window of most GUI desktop apps. It provides easy access to selectable commands. The user selects and executes the desired command by a simple click on the menu item describing it. After dropping a `TMainMenu` on the form at design time, double-click its icon (or use its context menu) to open the menu editor, which provides an easy, straight-forward way to create a new menu skeleton. The menu editor has three regions (see Figure 6):

- **the title bar**

gives the name of the parent form (or datamodule) and the name of the menu being edited. When a menu item is selected it also shows the menu item's name, and the status of its `OnClick` event (the principal event of interest for a menu item).

- **a left panel**

which has eight toolbuttons used to rearrange or delete menu items, and below the buttons an area showing summary statistics about the menu, with a Help button at the bottom.

- **the main editing region**

on the right (for a newly created menu the editor is initially empty except for a single Add button). This area shows a copy or skeleton indicating what the main menu will look like at runtime when it is fully dropped down. Editing the menu is a process of adding the desired items one by one, growing the skeleton dynamically as you proceed. An Add button appears dynamically wherever there is the potential of adding a new item to the developing menu.

The `brush` bitmap is created when the form is created, and recreated whenever the button with bitmap is clicked:

```
procedure TMainForm.CreateBrushBitmap;
begin
  FreeAndNil(FBrushBitmap);
  FBrushBitmap:=TBitmap.Create;
  FBrushBitmap.Width:=BImage.Glyph.Width;
  FBrushBitmap.Height:=BImage.Glyph.Height;
  FBrushBitmap.Canvas.Draw(0,0,BImage.Glyph);
end;

procedure TMainForm.BImageClick(Sender: TObject);
begin
  With OPD do
   If Execute then
     begin
       BImage.Glyph.LoadFromFile(FileName);
       CreateBrushBitmap;
       RefreshPanels(Self);
```

The following illustration shows the effect:

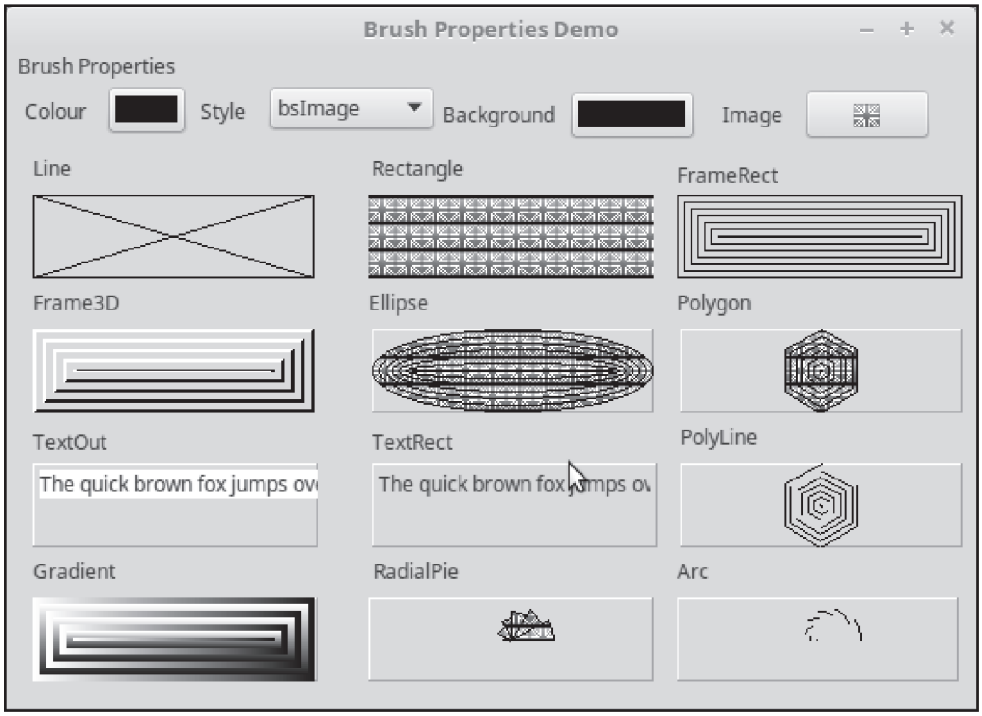


Figure 8: The brush properties

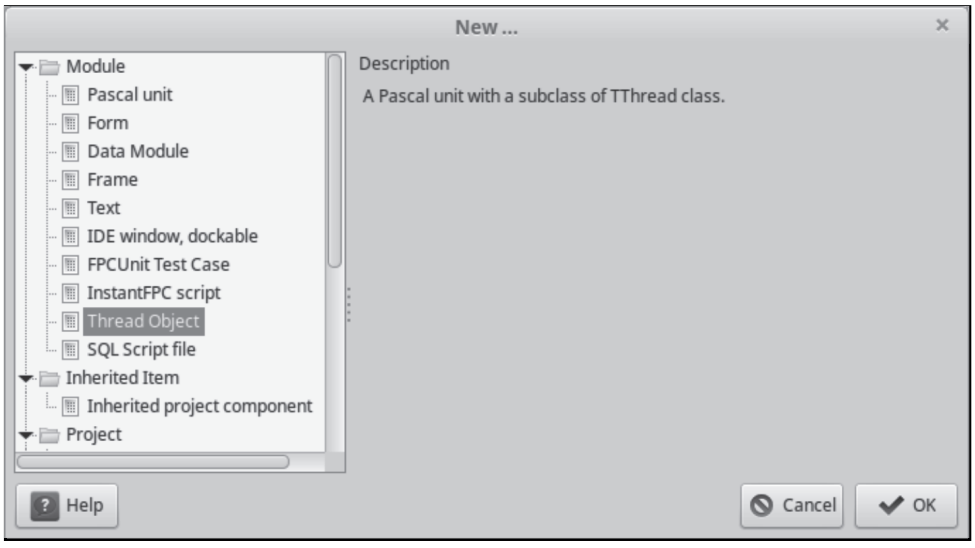


Figure 1: After installing the lazthread package, File->New gets a "Thread Object" option

Your thread's Execute method must contain the code you want to see executed in the thread. For instance, to sort a stringlist which has millions of entries, this could be as simple as:

```
var aList : TStringList;
procedure TMyThread.Execute;
begin
  aList.Sort;
end;
```

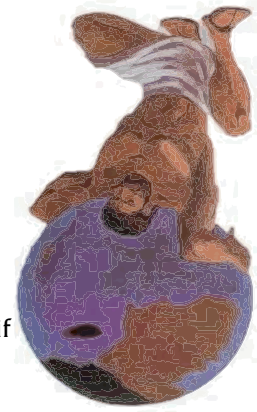
To sort a stringlist named `MyStringList` in the background, you can do this:

```
aList := MyStringList;
with TMyThread.Create(True) do
  begin
```

That looks easy enough and will do the work. However, this simple example has serious shortcomings:

- first of all, after calling `Start`, you do not get notified when the stringlist is actually sorted
- secondly, because the list is stored in a global variable you can sort only one `stringlist` at a time
- the combination of the two problems means effectively you can only do the sorting once

By Kim Madsen



Some people may have wondered if I have fallen off the face of the earth as I have been less vocal the last couple of weeks.

It has nothing to do with the dreadful COVID-19 infection I suppose most of us, one way or the other, are affected by. It rather has to do with being overly busy with various things.

One of the things, that relate to kbmMW and kbmMemTable, is the development of a brand new CompileTool which will be included in next release of kbmMemTable and kbmMW.

The purpose of the CompileTool is …
TA DAAA…. TO COMPILE STUFF

So what's new about that?
Not really much… but let me explain the rationale behind my apparant brain damage.

THE COMPILE TOOL

Compiling and installing **kbmMemTable,** has in Delphi always been fairly easy.
In C++Builder only mode, not so much, partly because the C++ only environment diverge more and more from what kbmMemTable originally supported, and the matching C++ project files.

To complicate matters even more, **kbmMW** can be a pain to install in **Delphi** due to kbmMW's ability to seemlessly integrate with loads of 3rdparty stuff. Paths and requirements and more, need to be provided. In C++ only mode it is even more complex to get it going, and despite the Compile Tool helping much on the situation, it is not fully solved with **kbmMW** yet, because C++ Builder exhibits random crashes and unexplained compile/link errors (*internal errors*).

But one of the things **Delphi** did very well… in the early days (*I suppose until XE got to see the light*), was to consistently tell you that some 3rdparty packages should be referenced to compile kbmMW's packages nicely. Delphi was even nice enough to add the relevant **requirements** to the **kbmMW** package so everything just worked.

Unfortunately, Delphi has stopped to do that stabily since many years.
I have reported it to Embarcadero on numerous occations, and they have acklowledged the issue, but have not been able to figure out why it has stopped working. Mind you.. sometimes it works… but then suddenly it does not, usually when that happens, it stops working for good in my experience.

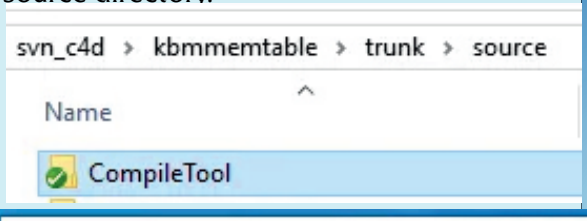
So the Compile Tool has as goal to:

1. Know everything about the units and requirements of the project at hand
2. Be able to manage knowledge about 3rdparty libraries/packages
3. Produce correct valid project files for the project that the Compile Tool has been prepared for.
4. Compile and install the projects automatically
5. Be able to recompile and restart itself, so it is up to date with whatever settings you may have made in for example kbmMWConfig.inc and thus based on those settings, is able to produce correct project files.

At first I created it to make compilation and installation of kbmMW easier, but it soon dawned to me that kbmMemTable should be supported too (standalone) and that it as such, could be a generic tool that will work for other developers too. (Currently it is not released with license for other 3rdparty developers to use it, but ping me if you have interest in that.)

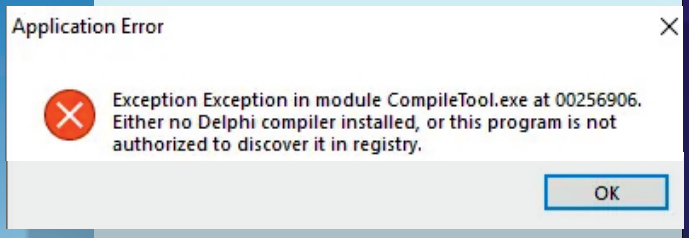
COMPILE TOOL FOR KBMMEMTABLE

Let us have a look at the **Compile Tool** for kbmMemTable. It will usually be found, as `CompileTool.exe`, in the source directory of the project for which it is supposed to support. Further, the source of the Compile Tool will be found in the subdirectory CompileTool under the managed projects source directory.

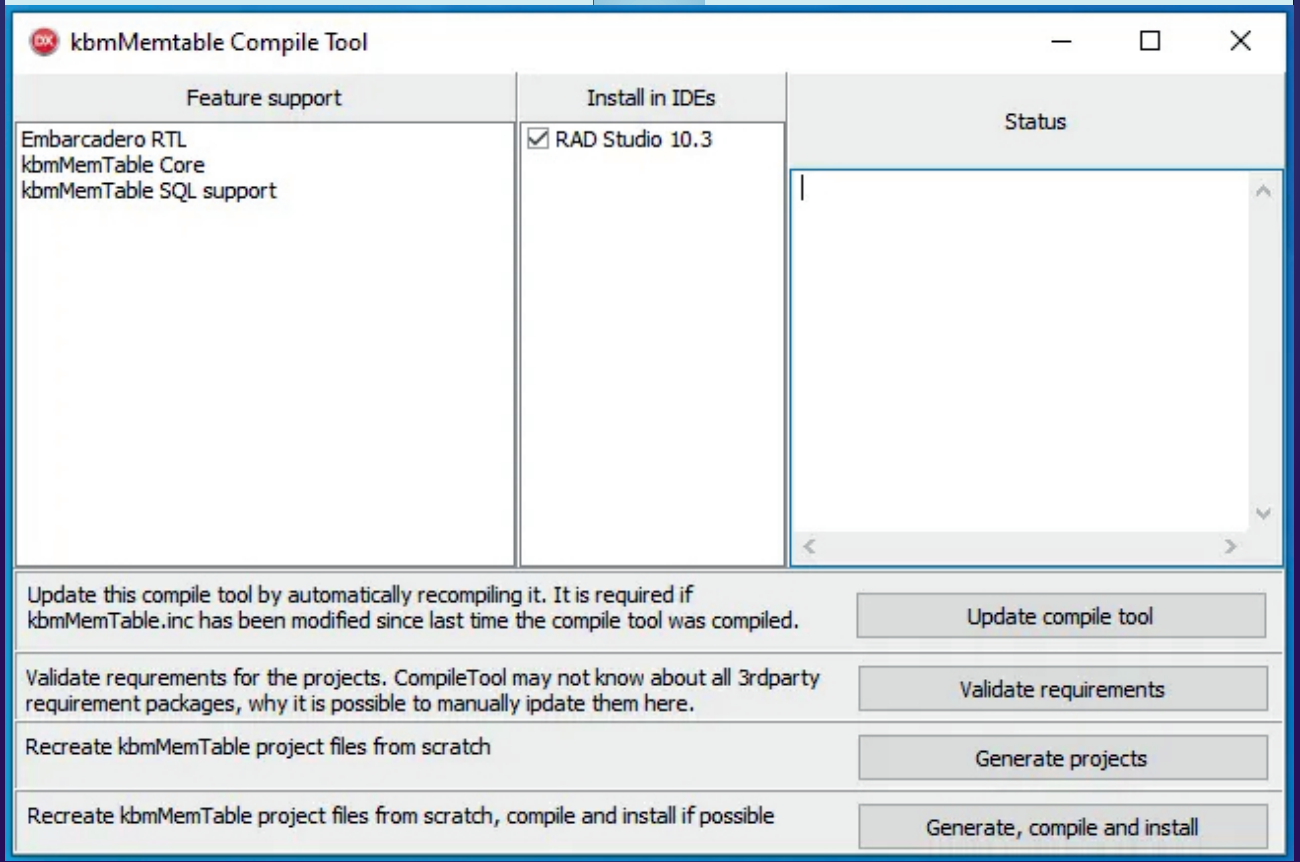


If you loose CompileTool.exe, it can be recompiled by opening and building the project in the CompileTool directory.
You can not use the CompileTool source/executable from a different project (like kbmMW), because they contain different settings, specially in the `uCompileToolFeatures.pas` file, which is specialized for each project.

If you start `CompileTool.exe` on a computer on which Delphi, C++Builder or RAD Studio is not installed, you will get an exception and the tool will shut down.



So let us start it on a computer with RAD Studio installed.



On the left side, a list of supported features for the project, is listed. In the case of kbmMemTable it is pretty simple, and will not change, since all features are available for all versions of kbmMemTable. However look later for how the Compile Tool for kbmMW looks.

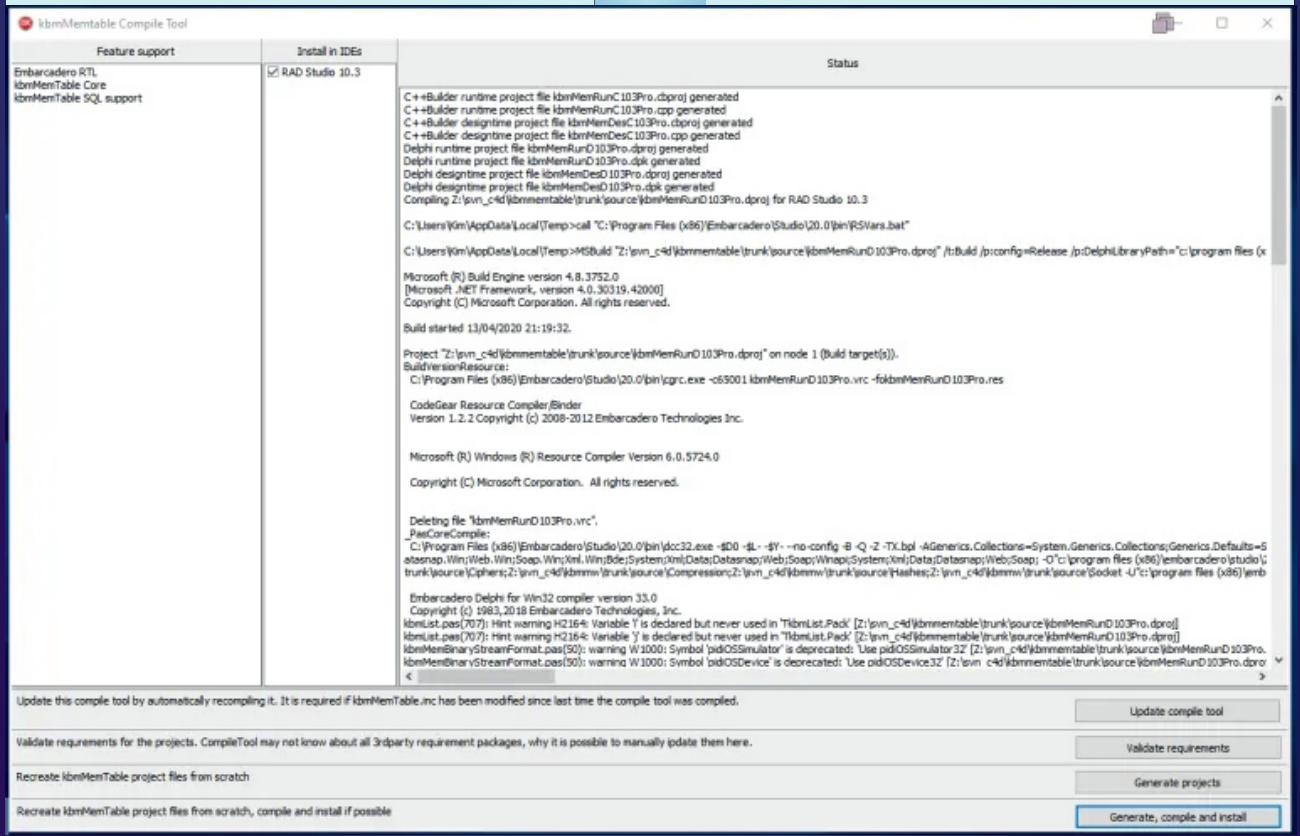
In the center, the discovered versions of Delphi, C++Builder or RAD Studio is listed. You can select, in which of them, kbmMemTable should be installed.
On the right side, you can follow the status of what is happening.
At the bottom, various buttons are available.

Update compile tool
– Will rebuild the Compile Tool itself, and restart it with the newly compiled executable.
For kbmMemTable it is not often needed to do, but in the case of kbmMW, you will want to do that, everytime you have changed something in kbmMWConfig.inc.

Validate requirements
– It will show a dialog, where 3rdparty requirements, which the Compile Tool has not been able to resolve itself, can be defined. The definitions made here will be remembered for next time in the file `CompileTool.ini`, making it easier to recompile/install without having to reconfigure each time. The `.ini` file will never be overwritten by kbmMemTable or kbmMW installations. kbmMemTable will usually not need any settings in this dialog. See section about kbmMW installation further on these pages for more information.

Generate projects
– It will produce new kbmMemTable project files matching the selected IDE. The project files will automatically be written to the parent directory (which is the kbmMemTable source directory).

Generate, compile and install
– It will generate projects, as described above, →



compile the projects and automatically install the resulting packages in the IDE, for all selected IDE's.
Usually you will be required to **close the selected IDE before being able to compile and install**. You can start with the option -F: `CompileTool.exe -F` to override the requirement to stop the IDE.
However the packages will not show up until you restart the IDE later on, and if the packages already was in use by the IDE you will get compile/linker errors.

This is an example of the result. You may notice that there are various paths shown in the status. Those paths are automatically picked up from your current installation, and provided for the compiler by the Compile Tool.
After it succeeds, you can close the tool, and start the IDE.
Now kbmMemTable will be available and installed with all paths for **Windows 32** compilation, correctly setup automatically.

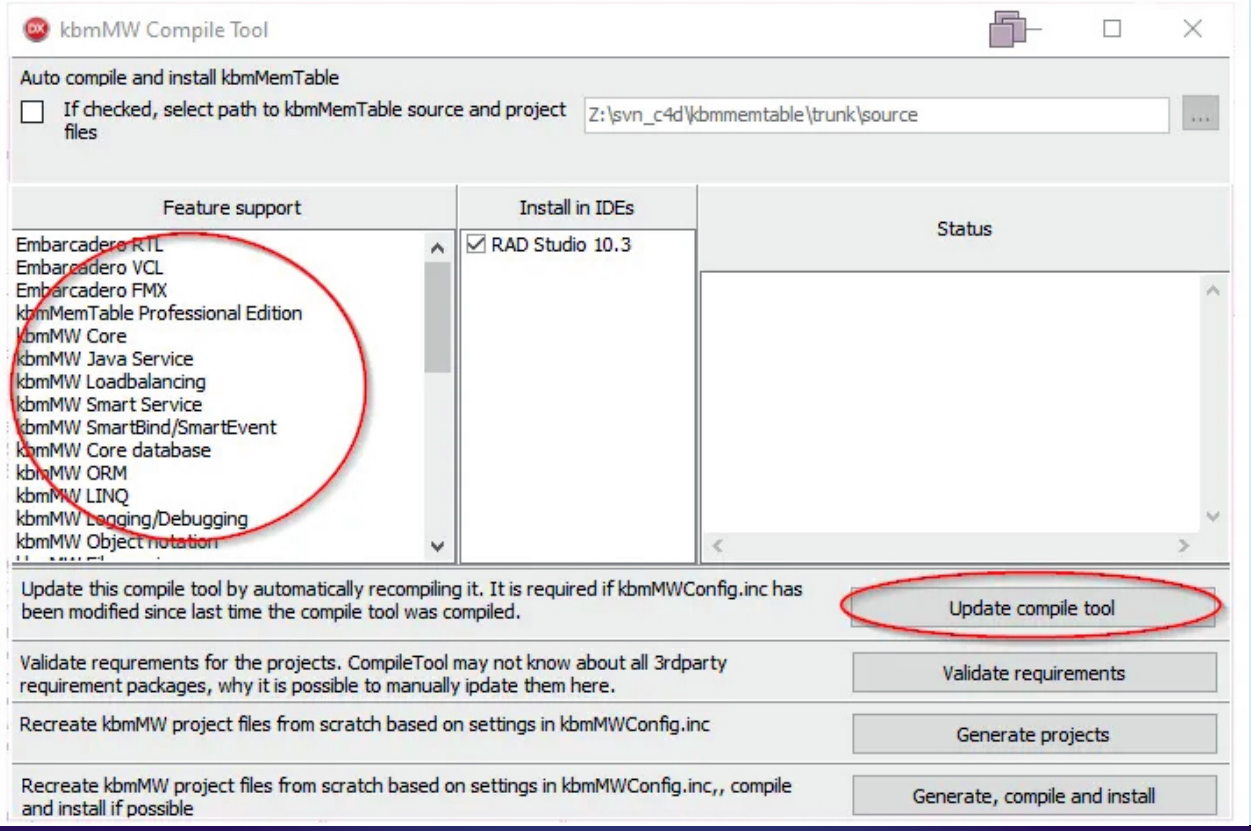
So let us look at how it works when installing kbmMW.

Compile Tool for kbmMW
In this case, I have, for the demo, opened `kbmMWConfig.inc`, uncommented the line:

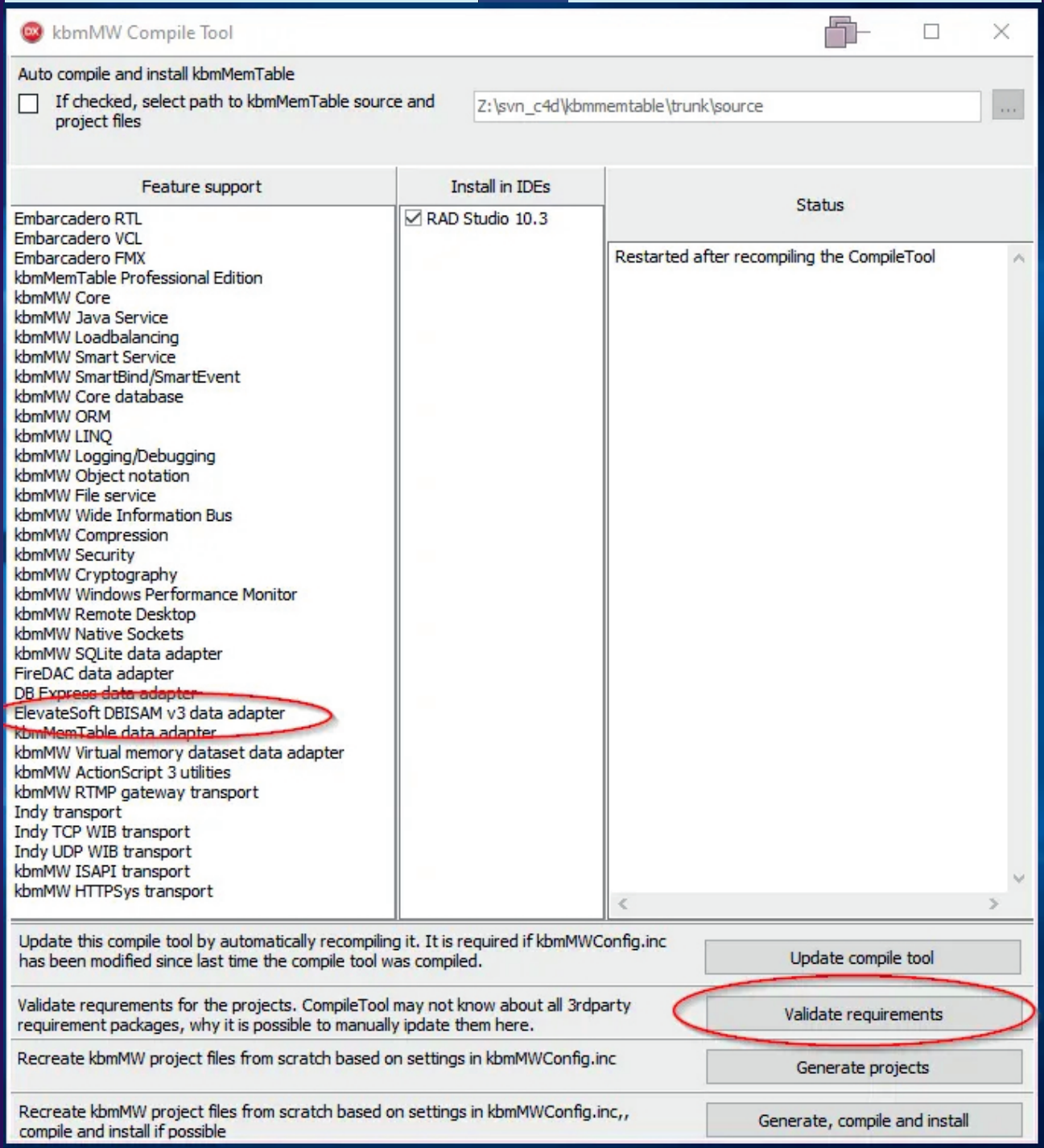
```
{$DEFINE KBMMW_DBISAM3_SUPPORT} // DBISAM 3 support.
```

and saved the file again, which essentially tells kbmMW that we want full support for DBISAM v3.
(FTR (*First Time Right*) there are similar defines for 36 other databases too, incl. DBISAM v4, ElevateDB, NexusDB and many many more).

Since this is a new setting, I first start the Compile Tool, where the Feature support at this time do not include DBISAM3, and let it recompile itself.

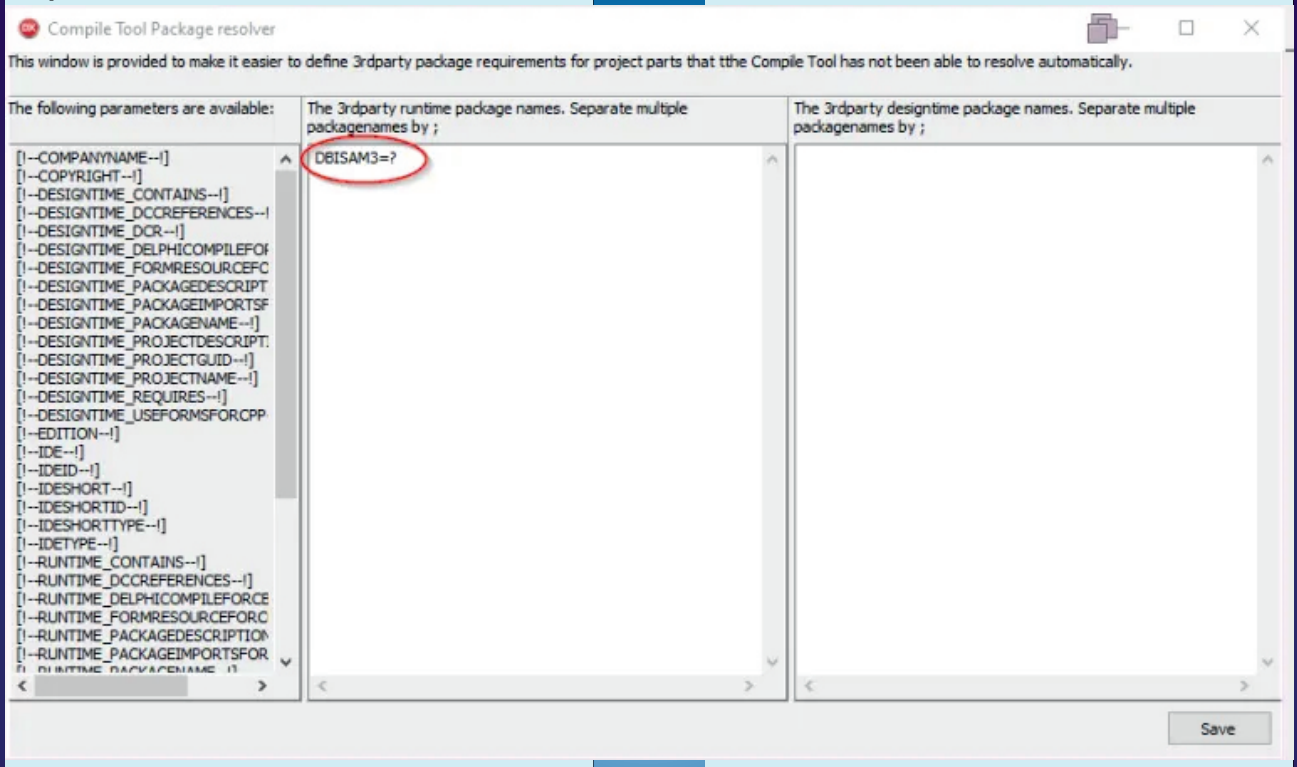


When it restarts, it looks like this:



Now the Compile Tool recognize the request for feature support for ElevateSoft DBISAM v3.
Since it is a new 3rdparty tool that kbmMW should support, compared to previous settings, we want to check the requirements dialog by clicking on the Validate requirements button.

If you need to refer to multiple packages/requirements for the DBISAM3 selection, you can separate those with a semicolon ;
Click save, and the Compile Tool will remember your settings, also for next time you start the Compile Tool.



You can see that there is a runtime package requirement that is currently unresolved.
To resolve it, simply type the name of the DBISAM v3 runtime package, typically something like db300d20 or something along those lines. As the structure of the package name can vary wildly between various 3rdparty projects, it is left for you to type the right value. Remember that this value will be used for all the IDE's that the Compile Tool will compile for.

You can include parameters in the name. Eg. `DBISAM3=db300[!-IDESHORTYPE-!][!-IDESHORTID-!]` which will replace `[!-IDESHORTYPE-!]` with D for Delphi or C for C++Builder and `[!-IDESHORTID-!]` with 20 for Delphi 10.3.

Clicking either Create projects or Create, compile and install, will ensure the kbmMW project files contains the relevant requirements.

Also notice that there is an extra Prerequisites section at the top of the Compile Tools window. It is there because kbmMW require compilation and installation of kbmMemTable beforehand.
You can point out where it's source is, and click the checkbox, then it will automatically recompile and install it when you recompile and install kbmMW via the Create, compile and install button.

BEHIND THE SCENES

So what is happening behind the scenes? Well… I told about the `uCompileToolFeatures.pas` file which is special for each project. It is in that file of the Compile Tool sources, where the project specialities are defined.

```
type
TkbmCTFeatures = class
const
  LOWEST_SUPPORTED_BDS_VERSION = 12.0; // XE5
public
  class function GetText(const ATextInfo:TkbmCTTextInfo):string;
  class procedure RegisterRuntimeFeatures(const AInfo:TProjectInfos);
  class procedure RegisterDesigntimeFeatures(const AInfo:TProjectInfos);
  class function BuildParameters(const AMain:TfrmMain; const ACpp:boolean; const AIDE:TIDEInfo):TStringList;
  class function GenerateProjectFileName(const ACpp:boolean; const AIDE:TIDEInfo; const ADesignTime:boolean):string;
end;
```

It contains a class definition which groups a few methods which should be defined

```
class function TkbmCTFeatures.GetText(const ATextInfo:TkbmCTTextInfo):string;
begin
  case ATextInfo of
    ctttiCaption:          Result:='kbmMW Compile Tool';
    ctttiRebuildToolCaption:    Result:='Update this compile tool by automatically recompiling it. It is required if
        kbmMWConfig.inc has been modified since last time the compile tool was compiled.';
    ctttiRecreateProjectsCaption: Result:='Recreate kbmMW project files from scratch based on settings in
        kbmMWConfig.inc';
    ctttiRecreateInstallCaption: Result:='Recreate kbmMW project files from scratch based on settings in
        kbmMWConfig.inc,, compile and install if possible';
    ctttiPrerequisiteCaption:   Result:='Auto compile and install kbmMemTable';
    ctttiPrerequisiteExplanation: Result:='If checked, select path to kbmMemTable source and project files';
  end;
end;
```

The above specifies the texts to be shown and thus can be configured for other projects (like kbmMemTable which

```
class function TkbmCTFeatures.BuildParameters(const AMain:TfrmMain; const ACpp:boolean;
const AIDE:TIDEInfo):TStringList;
```

This method builds relevant parameters that must exist for the project file generation. It includes version numbers, project names and descriptions and more. The method will be called multiple times during project generation.

```
class procedure TkbmCTFeatures.RegisterRuntimeFeatures(const AInfo:TProjectInfos);
begin
   AInfo.AddProjectInfo('RTL','Embarcadero RTL','rtl','',true);

   AInfo.AddProjectInfo('VCL','Embarcadero VCL','vcl;vclimg','',true);

   AInfo.AddProjectInfo('FMX','Embarcadero FMX','fmx','',true);

{$IFDEF KBMMW_ENTERPRISE_EDITION}
   AInfo.AddProjectInfo('KBMMEMTABLE','kbmMemTable Professional Edition','kbmMemRun[!--IDE--!]Pro','',true);
{$ELSE}
 {$IFDEF KBMMW_PROFESSIONAL_EDITION}
   AInfo.AddProjectInfo('KBMMEMTABLE','kbmMemTable Professional Edition','kbmMemRun[!--IDE--!]Pro','',true);
 {$ELSE}
   AInfo.AddProjectInfo('KBMMEMTABLE','kbmMemTable Standard Edition','kbmMemRun[!--IDE--!]Std','',true);
 {$ENDIF}
{$ENDIF}
...
 {$IFDEF KBMMW_DBISAM3_SUPPORT}
   AInfo.AddProjectInfo('DBISAM3','ElevateSoft DBISAM v3 data adapter','?','kbmMWDBISAM3',true);
 {$ENDIF}

...
```

This section defines all the features that can exist in a kbmMW runtime package, and their library requirements and units, including the DBISAM v3 option.

`AddProjectInfo` takes 5 arguments:

The unique ID of the project part.
For example KBMMEMTABLE.
Any ID can be used, as long as it is unique.
The descriptive name of the project part.
The libraries that are required for the project part, separated by semicolon and without file extensions. If it is an empty string, there are no requirements for that particular project part. If it is a question mark, it is unknown, and thus can be handled by the user in the Compile Tool package resolver dialog.
It is allowed to include paths if needed, but recommended to only use relative paths from the Source directory.

Further it is legal to prefix each library with either < or }.
Doing so will ensure to sort the item first (<) or last (}), rather just according to its regular name, when project files are generated.

The unit names (without extension) that are to be part of this project part.
Multiple unit names can be specified separated by semicolon (;).
If the unit also encompasses a form or datamodule file, use this syntax:
`unitname=formname:formclass.`
Eg.
`kbmMWCustomJavaService=kbmMWCustomJavaService:TkbmMWSimpleService;}JNI`

A boolean indicating if a requirement is mandatory for this project part. It is used for validation/warning that the project file may not have been generated correctly, if the requirement value has not been made available.

```
class procedure TkbmCTFeatures.RegisterDesigntimeFeatures(const AInfo:TProjectInfos);
begin
   AInfo.AddProjectInfo('RTL','Embarcadero RTL','rtl','',true);

   AInfo.AddProjectInfo('VCL','Embarcadero VCL','vcl;vclimg;vclx','',true);

   AInfo.AddProjectInfo('IDE','Embarcadero IDE','designide','',true);

   AInfo.AddProjectInfo('FMX','Embarcadero FMX','fmx','',true);

{$IFDEF KBMMW_LICENSE_DATABASE}
   AInfo.AddProjectInfo('KBMMW core database','kbmMW Core database','dbrtl;vcldb','',true);
{$ENDIF}
...
```

This section defines all the features that can exist in a kbmMW designtime package, and their library requirements and units.

It follows the same explanation as for the runtime part.

```
class function TkbmCTFeatures.GenerateProjectFileName(
const ACpp:boolean; const AIDE:TIDEInfo; const ADesignTime:boolean):string;
```

The above code is used for creating the relevant project file name for a specific IDE. The file name should not include any file extensions. The method will be called multiple times during project creation.

Final plea

If you have reached here, then you are qualified to assist me making the Compile Tool better.
Since kbmMW supports so many 3rdparty tools, I have for now only specified library requirements in the Compile Tool for some of them.
For the remaining, I have left it for you to define, simply because I do not know the naming rules for all the various 3rdparty libraries.
Please comment here on this thread if you have some pet libraries that kbmMW supports and that you would like the Compile Tool to know about,
preferably with the complete description of how the library name is defined. I.e. what each part of the name consists of.

If the name does not change based on the version of the 3rdparty library, it most likely will be possible to add automatic support in the Compile Tool for that particular library, making it even easier to compile and install kbmMW.

KBMMW PROFESSIONAL AND ENTERPRISE EDITION V. 5.10.20 RELEASED!

- RAD Studio XE2 to 10.3 Rio 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! SmartBind now fully supports VCL, FMX, including image/graphics and TListView**
- **NEW! SmartBind data generators and data proxies for easy separation of data sharing concerns in modular applications**
- **NEW! SmartEvent for easy separation of event and execution workflow based concerns for the ultimate in modular application design**
- **NEW! Native highly scalable TCP server transport now also supports REST**
- **Significant improvements and fixes in many areas including**
  - **RTTI**
  - **Scheduler**
  - **LINQ**
  - **Object Notation**
  - **ORM**

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

COMPONENTS 4 DEVELOPERS

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