TMS FNC WX Pack

Developer guide

tmssoftware.com

Copyright © 2022 tmssoftware.com

Table of contents

1. Getting started 6
1.1 Overview 6
1.1.1 Availablility 6
1.1.2 Android Support 6
1.1.3 Licensing 6
2. Reference 8
2.1 TTMSFNCWXBarcode 8
2.1.1 Creating barcodes 8
2.1.2 Retrieving barcodes 8
2.1.3 Customization 9
2.2 TTMSFNCWXCamera 10
2.2.1 Requirements 10
2.2.2 Control the camera 10
2.2.3 List of available camera devices 10
2.2.4 Take a snapshot 11
2.3 TTMSFNCWXHTMLMemo 12
2.3.1 Load and save HTML content 12
2.3.2 Formatting 12
2.3.3 Toolbar 13
2.4 TTMSFNCWXPDFViewer 15
2.4.1 Load a PDF 15
2.4.2 Thumbnails 15
2.4.3 Methods 16
2.4.4 Events 16
2.5 TTMSFNCWXQRCode 17
2.5.1 Creating QR codes 17
2.5.2 Retrieving QR codes 17
2.5.3 Customization 17
2.6 TTMSFNCWXVideoPlayer 19
2.6.1 Properties 19
2.6.2 Methods 19
2.6.3 Events 20

Copyright © 2022 tmssoftware.com

2.7 TTMSFNCWXJSONFormatter 21
2.7.1 Properties 21
2.8 TTMSFNCWXSpeechSynthesis 22
2.8.1 Properties 22
2.8.2 Methods 22
2.8.3 Events 23
2.9 TTMSFNCWXOCR 24
2.9.1 Language 24
2.9.2 WorkerCount 24
2.9.3 Methods 24
2.9.4 Events 25
2.9.5 TTMSFNCOCRResult 26
2.9.6 TTMSFNCOCRParagraph 26
2.9.7 TTMSFNCWXOCRLine 27
2.9.8 TTMSFNCWXOCRWord 27
2.10 TTMSFNCWXOCRHelper 28
2.10.1 Getting started 28
2.10.2 Feedback 29
2.10.3 Properties 30
2.10.4 Methods 30
2.10.5 TTMSFNCWXOCRHelperSelectionResult 32
2.10.6 TTMSFNCWXOCRHelperRectangle 32
2.11 TTMSFNCWXPDFThumbnailList 33
2.11.1 Properties 33
2.11.2 Events 33
2.12 TTMSFNCWXPDFToolbar 34
2.12.1 Adding buttons 34
2.12.2 Properties 34
2.12.3 Events 35
2.13 TTMSFNCWXBarcodeDecoder 36
2.13.1 Properties 36
2.13.2 Methods 36
2.13.3 Events 36
2.14 TTMSFNCWXQRDecoder 37
2.14.1 Properties 37

Copyright © 2022 tmssoftware.com

2.14.2 Methods 37
2.14.3 Events 37
2.15 TTMSFNCWXAudioPlayer 38
2.15.1 Properties 38
2.15.2 Methods 38
2.15.3 Events 39
2.16 TTMSFNCWXDocx 40
2.16.1 Building a document 40
2.16.2 Document 40
2.16.3 Sections 40
2.16.4 Paragraphs 40
2.16.5 Tables 40
2.16.6 Rows 41
2.16.7 Cells 41
2.16.8 Text 41
2.16.9 Image 41
2.16.10 Bookmark & Internal Hyperlink 42
2.16.11 External Hyperlink 42
2.16.12 Table Of Contents 42
2.16.13 PageNumber 42
2.16.14 Headers and footers 43
2.16.15 Templating 43
2.16.16 Generating Docx 43
2.17 TTMSFNCWXContainer 44
2.17.1 Adding links 44
2.17.2 Adding Containers 44
2.17.3 Adding Actions 45
2.17.4 ExecuteJavaScript 46
2.17.5 JavaScript Bridge 46
2.18 TTMSFNCWXMathEditor 47
2.18.1 Geting Started 47
2.18.2 Keyboard Shortcuts 47
2.18.3 LaTeX Commands 47
2.18.4 Properties 48
2.18.5 Methods 48

Copyright © 2022 tmssoftware.com

2.18.6 Events 48

Copyright © 2022 tmssoftware.com

1. Getting started

1.1 Overview

Included in the TMS FNC WX Pack is a set of components that enable you to use certain web technologies inside your native apps. When creating a new project, the tool palette will allow you to select and use these components.

1.1.1 Availablility

Supported frameworks and platforms

- VCL Win32/Win64
- FMX Win32/Win64, macOS, iOS, Android, Linux
- LCL Win32/Win64, macOS, iOS, Android, numerous Linux variants including Raspbian
- WEB: Chrome, Edge, Firefox, …

Supported IDE's

- Delphi XE7 and C++ Builder XE7 or newer releases
- Lazarus 2.0 with FPC 2.6.4 or newer official releases
- TMS WEB Core for Visual Studio Code 1.8 or newer releases

1.1.2 Android Support

For deploying on android, you'll need to add both the FNCWebChromeClient.jar and the JavascriptHelper.jar files to your project. You can read the instructions on how to this here.

1.1.3 Licensing

Several TMS FNC WX Pack components use open-source JavaScript libraries.

This includes:

Summernote distributed under the MIT license.
License: https://github.com/summernote/summernote/blob/develop/LICENSE

PDFJS is distributed under the Apache License 2.0
License: https://github.com/mozilla/pdf.js/blob/master/LICENSE

EasyQRCodeJS QR code generator is distributed under the MIT license
License: https://github.com/ushelp/EasyQRCodeJS/blob/master/LICENSE

BWIP-JS Bar code generator is distributed under the MIT license
License: https://github.com/metafloor/bwip-js/blob/master/LICENSE

Copyright © 2022 tmssoftware.com

JSON formatter is distributed under the MIT license
License: https://github.com/mohsen1/json-formatter-js/blob/master/LICENSE

Tesseract.js is distributed under the Apache License 2.0
License: https://github.com/naptha/tesseract.js/blob/master/LICENSE.md

Quagga2 barcode decoder is distributed under the MIT license
License: https://github.com/ericblade/quagga2/blob/master/LICENSE

jsQR QR code decoder is distributed under the Apache License 2.0
License: https://github.com/cozmo/jsQR/blob/master/LICENSE

Copyright © 2022 tmssoftware.com

2. Reference

2.1 TTMSFNCWXBarcode

The `TTMSFNCWXBarcode` is a component to generate barcodes. It uses the bwip-js JavaScript library.

There are various supported barcode formats. To familiarize yourself with the supported formats and the settings that applies to them, please visit the documentation page of the original bwipp library: https://github.com/bwipp/postscriptbarcode/wik

2.1.1 Creating barcodes

To create a barcode, all you need to do is set the `TTMSFNCWXBarcode.BarcodeType` and `TTMSFNCWXBarcode.Text` properties. This can be done both at designtime and runtime. Giving an unsupported format as `Text` or selecting a `BarcodeType` that does not support the current `Text` will cause an exception.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
 try
 TMSFNCWXBarcode1.BarcodeType := bciCode93Ext;
 except
 //handle exception here
 end;
 try
 TMSFNCWXBarcode1.Text := 'My code';
 except
 //handle exception here
 end;
end;
```

2.1.2 Retrieving barcodes

To retrieve the barcodes, implement the `TTMSFNCWXBarcode.OnGetBarcode` event. Access to the generated barcode is provided as a `TTMSFNCBitmap`. Process the `TTMSFNCBitmap` parameter as it fits best:

```
procedure TForm1.TMSFNCWXBarcode1GetBarcode(Sender: TObject;
 ABitmap: TTMSFNCBitmap);
begin
 //Save into a file, for example:
 ABitmap.SaveToFile('path_to\my_code.png');
end;
```

To retrieve a barcode syncronously, use the `TMSFNCWXBarcode.GetBarcode(AText: string)` method. Keep in mind, this method won't work on Android platforms.

```
procedure TForm1.Button1Click(Sender: TObject);
var
 bmp: TTMSFNCBitmap;
begin
 //Save into a file, for example:
 bmp := TMSFNCWXBarcode1.GetBarcode('codetocreate');
 try
 bmp.SaveToFile('path_to\my_code.png');
```


Copyright © 2022 tmssoftware.com

```
  finally
  bmp.Free;
  end;
end;
```

2.1.3 Customization

It's possible to customize the barcode with type specific settings though the provided `TMSFNCWXBarcode`.Settings property. To customize the look of the barcode, use the `TTMSFNCWXBarcode.BarcodeBorder, TTMSFNCWXBarcode.Scale` and `TTMSFNCWXBarcode.Color` properties.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
  TMSFNCWXBarcode1.BarcodeBorder.Color := gcRed;
  TMSFNCWXBarcode1.BarcodeBorder.Width := 2;
  TMSFNCWXBarcode1.BarcodeBorder.Visible := True;
  TMSFNCWXBarcode1.Color := gcAliceblue;
  TMSFNCWXBarcode1.Scale := 2; //Make it bigger while maintaining correct
line ratio
end;
```



Copyright © 2022 tmssoftware.com

2.2 TTMSFNCWXCamera

The `TTMSFNCWXCamera` is a component to access and show a stream of the selected camera.

2.2.1 Requirements

`macOS` requires internet connection.

The minimum required `iOS` version is `14.6`.

2.2.2 Control the camera

On `macOS` an internet connection is required as the camera file needs to be loaded from an online source. So as a first step, set the public `FileLocation` property to `cflOnline`.

The camera stream needs to be started manually:

```
procedure TForm1.Button1Click(Sender: TObject);
begin
 TMSFNCWXCamera1.Start;
end;
```

The camera feed can be paused or stopped by calling `TTMSFNCWXCamera.Pause` and `TTMSFNCWXCamera.Stop`.

2.2.3 List of available camera devices

The `TTMSFNCWXCamera` picks a default camera automatically. It's possible to change this behaviour by requesting a list of available camera devices, but keep in mind this functionality requires internet connection in native applications. For `WEB` it's expected that the application will be hosted in a secure context.

Change the `TTMSFNCWXCamera.CameraType` property to `wctSelected` to enable camera device selection. After that, call `TTMSFNCWXCamera.RequestDevices` to request a list of the available devices. This list of devices can be accessed once the `OnCameraDevicesInitialized` event is triggered.

```
procedure TForm2.FormCreate(Sender: TObject);
begin
 TMSFNCWXCamera1.CameraType := wctSelected;
 TMSFNCWXCamera1.RequestDevices;
end;
procedure TForm2.TMSFNCWXCamera1CameraDevicesInitialized(Sender: TObject);
begin
 if TMSFNCWXCamera1.Devices.Count > 0 then
 begin
 TMSFNCWXCamera1.SelectDevice(TMSFNCWXCamera1.Devices.Items[0]);
 TMSFNCWXCamera1.Start;
 end;
end;
```


Copyright © 2022 tmssoftware.com

2.2.4 Take a snapshot

Taking a snapshot of the current camera feed is as simple as calling `TTMSFNCWXCamera.TakeSnapshot` and implementing the `TTMSFNCWXCamera.OnSnapshot` event. The `TakeSnapshot` method has an `AFormat` parameter which will determine if the snapshould should be created as PNG or JPEG. By default it is created as PNG.

```
procedure TForm1.FormCreate(Sender: TObject);
begin
 TMSFNCWXCamera1.TakeSnapshot;
end;
procedure TForm1.TMSFNCWXCamera1Snapshot(Sender: TObject;
 ABitmap: TTMSFNCBitmap);
begin
 ABitmap.SaveToFile('path\to\picture.png');
end;
```


Copyright © 2022 tmssoftware.com

2.3 TTMSFNCWXHTMLMemo

Edit HTML directly with the `TTMSFNCWXHTMLMemo` component. It has an optional embedded format toolbar or can be used with external toolbar. Programmatic access to all formatting capabilities is available. Content is loaded and saved directly as HTML. It uses the summernote JavaScript library.

2.3.1 Load and save HTML content

There are 2 ways to load some content into the `TTMSFNCWXHTMLMemo`.

The first way is use the `LoadHTMLContent(AHTMLText: string)` method to load an HTML string into the editor.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
 TMSFNCWXHTMLMemo1.LoadHtmlContent('<b>Bold text</b>');
end;
```

The second way is to take advantage of the `HTML TStrings` property and use the loading mechanisms that are exposed through `TStrings` or set the `TStrings.Text` property. This method also applies to saving the contents.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
 //Load from file:
 TMSFNCWXHTMLMemo1.HTML.LoadFromFile('path\to\file');
 //Alternatively assign the Text property:
 TMSFNCWXHTMLMemo1.HTML.Text := '<b>Bold text</b>';
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
begin
TMSFNCWXHTMLMemo1.HTML.SaveToFile('path\to\file');
end;
```

2.3.2 Formatting

The `TTMSFNCWXHTMLMemo` is a WYSWYG editor. Via toolbar interaction the text can be formatted. However it’s not required to depend on the toolbar and the formatting is possible through exposed methods. This way the toolbar can be disabled and the `TTMSFNCWXHTMLMemo` formatting can be done through custom toolbars.

Copyright © 2022 tmssoftware.com

The following methods are available:

- ResetContent
- Undo
- Redo
- ToggleCodeview
- TextBold
- TextItalic
- TextUnderline
- TextStrikeThrough
- RemoveFormat
- FormatToParagraphs
- Indent
- Outdent
- InsertList(AOrdered: Boolean)
- InsertParagraph
- JustifyLeft
- JustifyRight
- JustifyCenter
- JustifyFull
- SetLineHeight(ALineHeight: Integer)
- SetFontSize(AFontSize: Integer)
- InsertText(AText: string)
- SetFontName(AFontName: string)

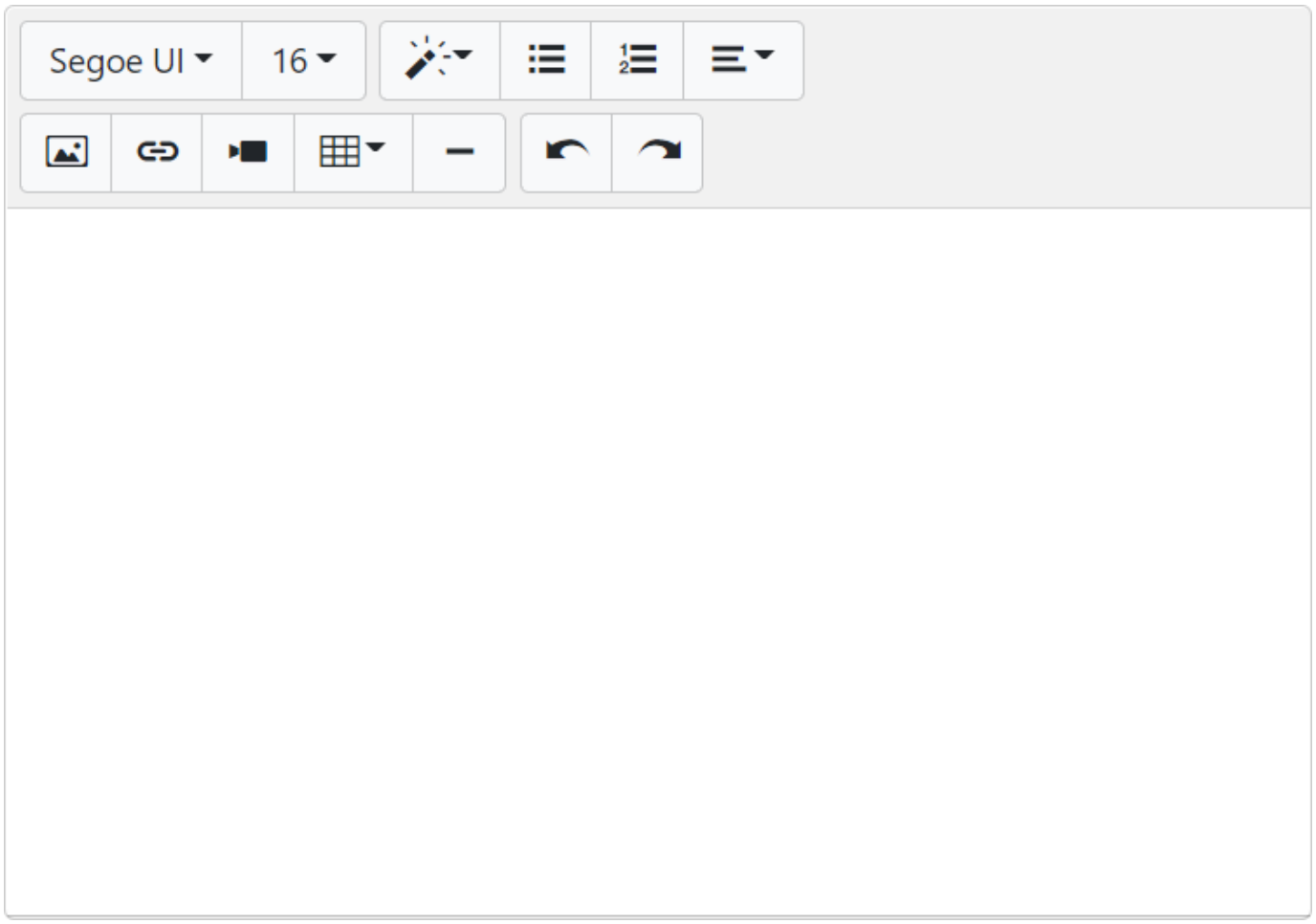
2.3.3 Toolbar

The toolbar can be disabled by setting `TTMSFNCWXHTMLMemo.Toolbar.Visible` to False.

It is also configurable which buttons should be visible in the toolbar. Disable or enable the buttons through the `TTMSFNCWXHTMLMemo.Toolbar.Buttons` property. By default all buttons are set to enabled:

```
procedure TForm1.FormCreate(Sender: TObject);
begin
 TMSFNCWXHTMLMemo1.BeginUpdate;
 TMSFNCWXHTMLMemo1.Toolbar.Buttons.FontStyle.BtColor := False;
 TMSFNCWXHTMLMemo1.Toolbar.Buttons.FontStyle.BtClear := False;
 TMSFNCWXHTMLMemo1.Toolbar.Buttons.FontStyle.BtStrikeThrough := False;
 TMSFNCWXHTMLMemo1.Toolbar.Buttons.FontStyle.BtUnderline := False;
 TMSFNCWXHTMLMemo1.Toolbar.Buttons.FontStyle.BtBold := False;
 TMSFNCWXHTMLMemo1.Toolbar.Buttons.FontStyle.BtItalic := False;
 TMSFNCWXHTMLMemo1.EndUpdate;
end;
```


Copyright © 2022 tmssoftware.com



Copyright © 2022 tmssoftware.com

2.4 TTMSFNCWXPDFViewer

The `TTMSFNCWXPDFViewer` is a component to display PDF files in your applications, including page thumbnails. There is programmatic access to navigate through pages, search text and more. It uses the PDF.js JavaScript library.

2.4.1 Load a PDF

You can load a PDF through different methods:

To load from a URL, use `TTMSFNCWXPDFViewer.LoadFromURL`. Keep in mind that the PDF will be retrieved via an HTTP request in this case, so any URLs that are CORS protected won't be able to load.

A PDF can be load as a base64 encoded string by using the `TTMSFNCWXPDFViewer.LoadFromBase64` method.

For VCL, FMX and LCL there is also an extra method available which enables PDF loading directly from files.

```
procedure TForm1.Button1Click(Sender: TObject);
begin
 //From URL:
 TMSFNCWXPDFViewer1.LoadFromUrl('my/url');
 //From base64:
 TMSFNCWXPDFViewer1.LoadFromBase64('base64encodedfile');
 //From file (only VCL, FMX and LCL):
 TMSFNCWXPDFViewer1.LoadFromFile('path\to\pdf_file.pdf');
end;
```

2.4.2 Thumbnails

By default thumbnail generation is disabled. To enable this functionality, set the `TTMSFNCWXPDFViewer.CalculateThumbnails` property to True. After that, whenever a PDF is loaded, the thumbnails will be calculated. These thumbnails can be accessed by implementing the `OnGenerateThumbnail` event.

```
procedure TForm1.TMSFNCPDFViewer1GenerateThumbnail(Sender: TObject;
 NumPage: Integer; Base64Img: string);
var
 ms: TMemoryStream;
 b: TBytes;
 Base64DataImg: string;
begin
 Base64DataImg := Copy(Base64Img, Pos(',', Base64Img) + 1,
Length(Base64Img));
 b := TTMSFNCUtils.Decode64ToBytes(Base64DataImg);
 ms := TMemoryStream.Create;
 try
 ms.Write(b, Length(b));
 ms.Position := 0;
 //Use your image here
 //For example: load it into a bitmap from a stream
 finally
 ms.Free;
 end;
end;
```


Copyright © 2022 tmssoftware.com

2.4.3 Methods

Property name	Description
NextPage	Navigate to the next page.
PreviousPage	Navigate to the previous page.
ShowPage(AIndex: Integer)	Show a page with a given index.
ZoomIn	Zoom into the PDF.
ZoomOut	Zoom out of the PDF.
TurnAngleLeft	Rotate the PDF to the left by 90 degrees.
TurnAngleRight	Rotate the PDF to the right by 90 degrees. GenerateImageActualPage
Get the current page as an image.	Search for a given text.

2.4.4 Events

Property name	Description
OnLoaded	Event triggers when the PDF has loaded.
OnPageLoaded	Event triggers when the current page has loaded.
OnPageRendered	Event triggers when the current page has been rendered.
OnLoadError	Event triggers when an error happens during loading.
OnGenerateThumbnail	Event triggers for each generated thumbnail.
OnGenerateImagePage	Event triggers as a result of the GenerateImageActualPage method.
OnHyperlinkClick	Event triggers when a hyperlink is clicked in the PDF.
OnThumbnailProgress	Event triggers as the thumbnails are being loaded. The progress can be tracked here.
OnViewerInitialized	Event triggers when the TTMSFNCWXPDFViewer is initialized.
OnFoundSearchText	Event triggers for each match for the SearchText method.
OnNotFoundSearchText	Event triggers when no search results were found for a given text.
OnSearchTextFinished	Event triggers when the text search finished.

Copyright © 2022 tmssoftware.com

2.5 TTMSFNCWXQRCode

The `TTMSFNCWXQRCode` is a component to generate and show highly customizable QR codes. It uses the EasyQRCodeJS JavaScript library.

2.5.1 Creating QR codes

Creating a QR code is as simple as setting the `TTMSFNCWXQRCode.Text` property either at designtime or at runtime. The default error correction level is set to M, it can be changed via the `TTMSFNCWXQRCode.ErrorCorrectionLevel` property:

```
procedure TForm2.Button1Click(Sender: TObject);
begin
 TMSFNCWXQRCode1.ErrorCorrectionLevel := eclH;
 TMSFNCWXQRCode1.Text := 'text to be encoded';
end;
```

2.5.2 Retrieving QR codes

To retrieve the QR codes, implement the `TTMSFNCWXQRCode.OnGetQRCode` event. Access to the generated QR code is provided as a `TTMSFNCBitmap`. Process the `TTMSFNCBitmap` parameter as it fits best:

```
procedure TForm1.TMSFNCWXQRCode1GetBarcode(Sender: TObject;
 ABitmap: TTMSFNCBitmap);
begin
 //Save into a file, for example:
 ABitmap.SaveToFile('path_to\my_code.png');
end;
```

2.5.3 Customization

The `TTMSFNCWXQRCode` component provides various properties to customize the final QR code. For all the available properties please check the Options part of the EasyQRCodeJS library: https://github.com/ushelp/EasyQRCodeJS#options

```
procedure TForm2.Button1Click(Sender: TObject);
begin
 TMSFNCWXQRCode1.Text := 'my cusomized barcode';
 TMSFNCWXQRCode1.Settings.ColorDark := gcDeepskyblue;
 TMSFNCWXQRCode1.Settings.ColorLight := gcAzure;
 TMSFNCWXQRCode1.Settings.TimingHorizonalColor := gcDeepskyblue;
 TMSFNCWXQRCode1.Settings.TimingVerticalColor := gcDeepskyblue;
 TMSFNCWXQRCode1.Settings.PositionColor.TopLeftOuterColor := gcDodgerblue;
 TMSFNCWXQRCode1.Settings.PositionColor.TopLeftInnerColor := gcDodgerblue;
TMSFNCWXQRCode1.Settings.PositionColor.TopRightOuterColor :=
gcDodgerblue;
 TMSFNCWXQRCode1.Settings.PositionColor.TopRightInnerColor :=
gcDodgerblue;
 TMSFNCWXQRCode1.Settings.PositionColor.BottomRightOuterColor :=
gcDodgerblue;
 TMSFNCWXQRCode1.Settings.PositionColor.BottomRightInnerColor :=
gcDodgerblue;
end;
```


Copyright © 2022 tmssoftware.com



Copyright © 2022 tmssoftware.com

2.6 TTMSFNCWXVideoPlayer

The `TTMSFNCWXVideoPlayer` is a component that lets you play a video from a URL. You can toggle the controls inside the video. But you can also use all features from outside the video.

2.6.1 Properties

Property name	Description
Autoplay	Automatically plays a video when loaded.
Controls	Toggles controls inside video component.
CurrentTime	Get or set the current time of the video.
Duration (ReadOnly)	Get the duration of the video.
IsPlaying	Checks if the video is playing.
Loop	Restart the video when it ended.
Muted	Mute the video.
PlayBackRate	Change how fast the video plays.
DisablePictureInPicture	Disables the `PictureInPicture` feature.
AutoPictureInPicture	Automatically toggles `PictureInPicture` when switching tabs (web only).
Poster	Loads an image when video data is unavailable.
PictureInPicture	Makes the video come out of the component.
URL	Sets the video source to load. Can be a local file using `"file://"`.
Volume	Set the volume of the video.

2.6.2 Methods

Property name	Description
ForwardVideo	Skips the video with x seconds.
RewindVideo	Return x seconds in the video.
Play	Start playing the video.
Pause	Stop Playing the video.

Copyright © 2022 tmssoftware.com

2.6.3 Events

Property name	Description
OnPlay	Triggers when the video starts playing.
OnPause	Triggers when the video is paused.
OnEnded	Triggers when the video is ended.
OnRateChange	Triggers when the `PlayBackRate` has changed.
OnVolumeChange	Triggers when the volume has been changed.
OnTimeUpdate	Triggers when the `CurrentTime` has been changed.
OnError	Triggers when an error has occurred playing the video.
OnPictureInPictureEnter	Triggers when the video enters `PictureInPicture`.
OnPictureInPictureLeave	Triggers when the video leaves `PictureInPicture`.
OnMetaDataLoaded	Triggers when the video metadata has been loaded.
OnVideoInitialized	Triggers when the video has been initialized.

Copyright © 2022 tmssoftware.com

2.7 TTMSFNCWXJSONFormatter

The `TTMSFNCWXJSONFormatter` lets you input a JSON string and renders it in a readable JSON object. This component uses the JSONFormatter.js Library.

2.7.1 Properties

Property name	Description
JSON	Sets the JSON string to format
HoverPreviewEnabled	Enable preview on hover
HoverPreviewArrayCount	Number of array items to show in preview Any array larger than this number will be shown as `Array[XXX]` where `XXX` is length of the array
HoverPreviewFieldCount	Number of object properties to show for object preview. Any object with more properties that thin number will be truncated.
Theme	Sets the theme. Theme can be Light, Dark or Custom. When set to custom, it takes the color settings found under `Appearance`.
AnimateOpen	Enable animations when expanding JSON object
Animate Close	Enable animations when closing JSON Object
OpenAtDepth	Set How many levels of the rendered tree should open. `-1` for expanding all levels and `0` to collapse all levels
Appearance	Set the colors for the Custom Theme
UseToJSON	Internally uses the toJSON method to render an object as a string as available. Useful for objects like Dates or JavaScript objects used in TMS WEB Core, that might make more sense as a string than as empty objects. True by default

Copyright © 2022 tmssoftware.com

2.8 TTMSFNCWXSpeechSynthesis

This component lets you convert text to speech. You can use an editor to set the voice at design and runtime or you can set them manually using the properties.

Keep in mind

This component is not supported on Android.

https://bugs.chromium.org/p/chromium/issues/detail?id=487255

2.8.1 Properties

Property name	Description
Pitch	Get or set the pitch of the speaking voice
Rate	Set the rate at how quick the voice is speaking
Volume	Set how loud the voice is speaking
Voices	Get a list of all available voices
Voice	Set a voice to use
IsSpeaking	Check if the component is already speaking

2.8.2 Methods

Property name	Description
Speak	Let’s you convert text to speech
Pause	Pauses the text to speech
Resume	Resumes the text to speech
Cancel	Cancels all queued text to speech
ConfigureVoices	Opens an editor at runtime to let the user configure the voice

Copyright © 2022 tmssoftware.com

2.8.3 Events

Property name	Description
OnVoiceReady	Triggers when all voices are finished loading and available in the `Voices` property
OnStart	Triggers when a text has started speaking
OnPause	Triggers when the text to speech has been paused
OnResume	Triggers when the text to speech has been resumed
OnEnd	Triggers when the text to speech has ended

Copyright © 2022 tmssoftware.com

2.9 TTMSFNCWXOCR

Component that recognizes text in an image. This component uses Tesseract.js library.

> **Keep in mind**
>
> This component is not supported on **Linux**!
>
> Having multiple instances of a **WEB** project using this component open, might cause out-of-memory crashes!

2.9.1 Language

Property to set the language to use for the OCR. There are over a 100 language packs available. You can see all available languages here: https://tesseract-ocr.github.io/tessdoc/Data-Files#datafiles-for-version-400-november-29-2016.

You can load multiple languages by combining them with a `+`. Example: `eng+deu+ned`

2.9.2 WorkerCount

This property sets the number of workers that will be initialized. Assigning more workers will let you complete more OCR processes simultaneously. The number of workers is minimum `1` and is capped at `15`. More than `15` of these workers, cause the application to go out of memory and crash. Please note that the more workers are needed, the longer the initialization of the component is going to take, and the more resources are required. When you start an OCR process the component will assign it to a scheduler and will start the process the moment a worker is available.

2.9.3 Methods

There are 4 methods for starting a OCR on an image. The difference between both is the way that the images are loaded. For getting the results of these methods it's important to subscribe to the `OnFinishOCR` event which forwards the result. As the whole process is something that happens asyncronous. Only PNG, JPG and JPEG are supported file formats for these methods. The results of these 2 methods will also be added to a `ResultList` which you can access like this: `TTMSFNCWXOCR.ResultList`. This list is of the type `TObjectList`, so you can manipulate this list al you want.

RecognizeURL

This method accepts a URL of an image somewhere on the internet and then starts the OCR Progress You can call this method like this:

```
TMSFNCWXOCR1.RecognizeURL('https://tesseract.projectnaptha.com/img/eng_bw.png');
```

RecognizeFile

This method accepts a `TTMSFNCUtilsFile` which can be a local file and converts it to a base64 image and then start the OCR progress.

Copyright © 2022 tmssoftware.com

For uploading a file in TMS Web Core you first have to convert from a `TFile` to a `TTMSFNCUtilsFile`. Here is a snippet on how to do this using a `TWebFilePicker`:

```
procedure TForm3.WebFilePicker1Change(Sender: TObject);
begin
TTMSFNCUtils.LoadFile(WebFilePicker1.Files[0].FileObject, @DoFileLoaded);
end;
procedure TForm3.DoFileLoaded(const AFile: TTMSFNCUtilsFile);
begin
 TMSFNCWXOCR1.RecognizeFile(AFile);
end;
```

In VCL an FMX you can just use this:

```
OpenDialog1.InitialDir := GetCurrentDir;
OpenDialog1.Filter := 'Images |*.PNG;*.JPEG;*.JPG';
if OpenDialog1.Execute then
begin
 Image1.Picture.LoadFromFile( OpenDIalog1.Files[0]);
 TMSFNCWXOCR1.RecognizeFile(OpenDialog1.Files[0]);
end;
```

RecognizeBytes

This method accepts a `TBytes` array and converts it to `base64` and starts the OCR Progress.

RecognizeBase64

This method accepts a `base64` string and starts the OCR Progress.

Terminate

This method cancels all processes and terminates all workers. This causes all workers to reinitialize as well.

2.9.4 Events

OnProgressOCR

This event triggers every time there is an update progress. This send a `TTMSFNCWXOCRProgress` object along. This object contains a status, progress and a `WorkerID` which can be used to show status updates of the OCR Process.

```
procedure TForm3.TMSFNCWXOCR1ProgressOCR(Sender: TObject;
 Progress: TTMSFNCWXOCRProgress);
begin
 Label1.Text := Progress.Status + ': ';
 ProgressBar1.Value := Progress.Progress * 100;
end;
```

OnFinishOCR

This Event Triggers when the OCR Progress has finished. This event exposes a `TTMSFNCOCRResult` object.

Copyright © 2022 tmssoftware.com

This object contains all information needed for handling the OCR.

```
procedure TForm3.TMSFNCWXOCR1FinishOCR(Sender: TObject;
 Result: TTMSFNCWXOCRResult);
begin
 TMSFNCWXJSONFormatter1.JSON.Text := Result.JSONString;
 WebMemo1.Text := Result.Text;
 WebLabel2.caption := 'Confidence: ' + FloatToStr(Result.Confidence);
end;
```

OnOCRError

This event triggers when an error occurs during the OCR process. This error is forwarded as a string in the event listener. This error is a direct message from the library. This means that we have no influence on the messages sent.

OnOCRInitialized

This event triggers when all workers have been initialized. After this you can freely start OCR processes.

2.9.5 TTMSFNCOCRResult

Property name	Description
Text	The result in plain text
HOCR	The result in HTML format
TSV	The result in TSV format
Confidence	Score of how certain the process is about this text
Paragraphs	List of TTMSFNCParagraphs
Lines	List of TTMSFNCLines
Words	List of TTMSFNCWords
JSONString	Full JSON of TTMSFNCOCRResult

2.9.6 TTMSFNCOCRParagraph

Property name	Description
Text	Plain text in this paragraph
Confidence	Score of how certain the process is about this paragraph
Lines	List of TTMSFNCLines in this paragraph
BBox	Object containing coordinates of where this paragraph is situated in the image.

Copyright © 2022 tmssoftware.com

2.9.7 TTMSFNCWXOCRLine

Property name	Description
Text	Plain text of this line
Confidence	Score of how certain the process is about this line
Words	List of TTMSFNCWords in this line
Bbox	Object containing the coordinates of where this line is situated in the image

2.9.8 TTMSFNCWXOCRWord

Property name	Description
Text	Plain text of this word
Confidence	Score of how certain the process is about this word
BBox	Object containing the coordinates of where this word is situated in the image

Copyright © 2022 tmssoftware.com

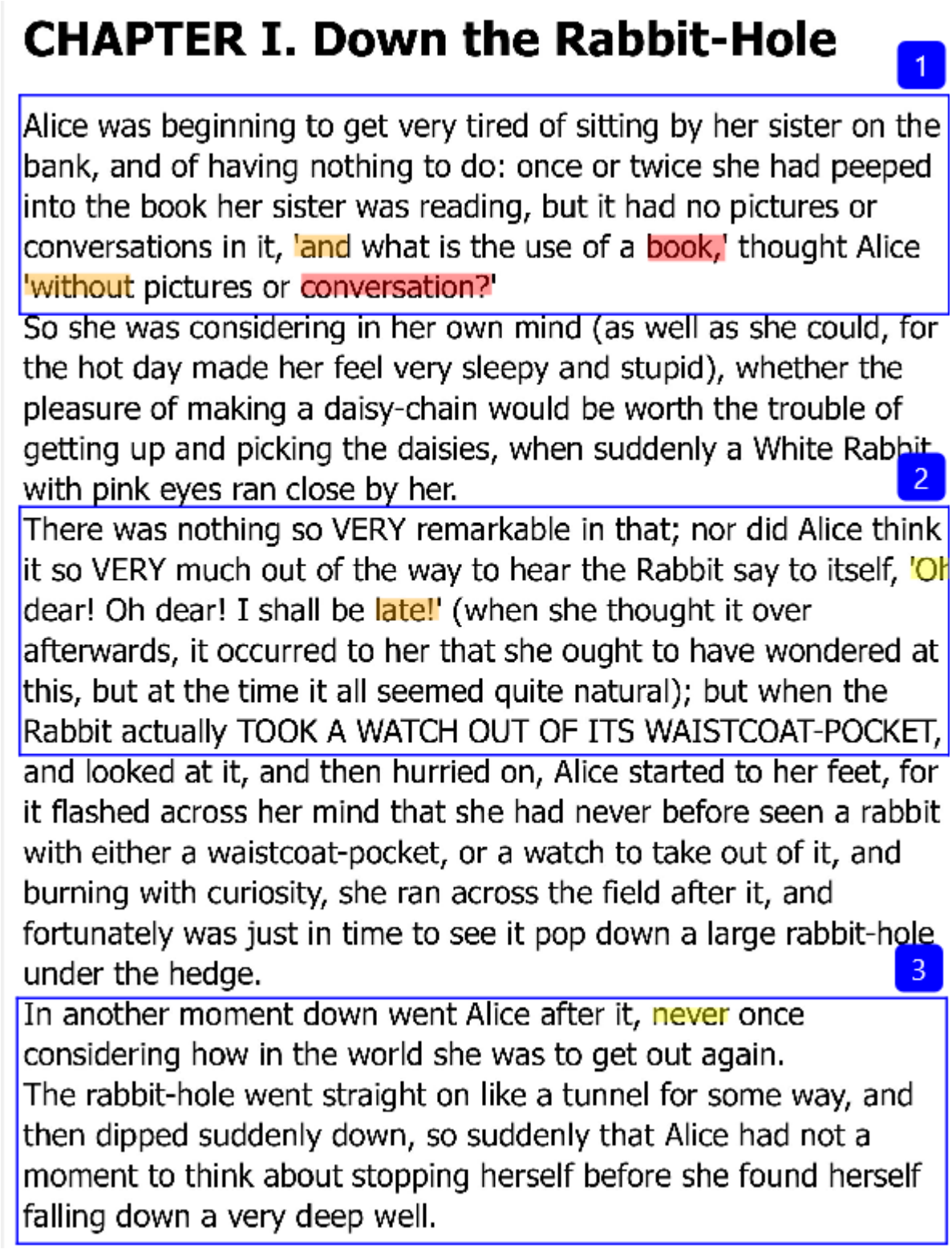
2.10 TTMSFNCWXOCRHelper

This component is designed to be used in connection with the `TTMSFNCWXOCR` component. You can use this component to load an image on which you can select multiple parts of an image to have OCR performed with the `TTMSFNCWXOCR` component on the selected text. The component will also show feedback about which words might not be correct. It will highlight these words with a color indicating the confidence of the OCR process. You can set 3 levels of confidence for this and three colors for these confidence levels.

2.10.1 Getting started

Simply dropping this component on the form and linking a `TTMSFNCWXOCR` component in the object is enough to get started. You can then start drawing rectangles. And start the OCR process using the `StartOCR` method. You can also set the `AutoStart` property on true. This means that the OCR Process automatically starts when you draw a rectangle on the component. All finished processes are stored in the `SelectionResults` collection. This object contains the OCRResult, the Bitmap, the drawn Rectangle, and the `ID` of the object.

Copyright © 2022 tmssoftware.com



2.10.2 Feedback

This component lets you set limits at which the component will highlight the uncertain words. When setting the `LowConfidence` property, all words which have a confidence lower than this will be highlighted. The same happens for the `ModerateConfidence` and `ReasonableConfidence`. You can disable these limits by setting the value to 0 or lower

Copyright © 2022 tmssoftware.com

2.10.3 Properties

Property name	Description
AutoStart	Setting this property to true, starts the OCR Process whenever you draw a rectangle in the component
EnableSelection	Setting this to false, disables the drawing of rectangles.
EnableFeedback	Setting this to false, disables showing feedback of the process
OCR	Property that lets you link a TTMSFNCWXOCR component. This is required to use the component to start the OCR Process
Bitmap	Property That lets you load an image in the component
LowConfidence	Set the bounds to highlight the words which have a confidence lower than this number.
ModerateConfidence	Set the bounds to highlight the words which have a confidence lower than this number
ReasonableConfidence	Set the bounds to highlight the words which have a confidence lower then this number
Appearance	Set the different colours for feedback and selecting rectangles on the component.

2.10.4 Methods

StartFullImageOCR

This method lets you start an OCR process for the whole Image.

StartOCR

This method starts all OCR processes which aren't already in progress. If there aren't any processes available, it will start a full image OCR Process

SaveToTemplate

This method returns a list of all drawn rectangles on the form. This list contains `TTMSFNCWXOCRHelperRectangle` objects. These objects contain the coordinates of the drawn rectangles.

SaveTemplateToJSON

This method returns a JSON string of the drawn template.

LoadTemplate

This method will load a template in the component. You can start the process for this template using the `StartOCR` method. If you want to re-use the template you will need to reload the template like this before starting the OCR process.

```
TMSFNCWXOCRHelper1.LoadTemplateFromJSON(JSONString);
TMSFNCWXOCRHelper1.StartOCR;
```


Copyright © 2022 tmssoftware.com

LoadTemplateFromJSON

Use this method to load a template from a JSON string. This JSON needs to use this template:


```
[
 {
 "Bottom": 803.559265136719,
 "DPIScale": 1.5,
 "Left": 11.9266719818115,
 "Right": 2319.73681640625,
 "Top": 190.826690673828
 },
 {
 "Bottom": 1738.31176757813,
 "DPIScale": 1.5,
 "Left": 41.743335723877,
 "Right": 2295.88330078125,
 "Top": 977.986755371094
 },
 {
 "Bottom": 2489.69189453125,
 "DPIScale": 1.5,
 "Left": 35.7799987792969,
 "Right": 1923.17492675781,
 "Top": 1854.59692382813
 }
]
```


ClearNonProcessSelections

This method will delete all Rectangles which are not in progress or are not already finished.

ClearAll

This method will delete all Rectangles. Please note that Rectangles which are in progress will be deleted but the OCR process will continue. The component will just skip over these results.

ResetOCR

This method will Terminate all workers and reinitialize them. This will force stop everything. This will also delete all rectangles and clear the results.

CheckRectanglesAvailableForOCR

This method checks if there are any rectangles that still needs to be processed and don’t have an OCR result, returns true if there are.

Copyright © 2022 tmssoftware.com

2.10.5 TTMSFNCWXOCRHelperSelectionResult

Property name	Description
ID	The id of the OCR Process, is automatically assigned
Bitmap	The part of the image that is used for the OCR.
SelectionDPIScale	Used to draw the rectangles correctly when showing it on the image. This property is set when you draw a rectangle.
FeedbackDPIScale	Used to show the feedback correctly. This property is set when you start the OCR Process.
OCRResult	The Result that returns from the `TTMSFNCWXOCR` component.
InProgress	Boolean to check if this entry has already been started.
Rectangle	This object of type TRectF contains the drawn rectangle.

2.10.6 TTMSFNCWXOCRHelperRectangle

This object is used to load and save to templates. This contains the data needed to redraw a rectangle. This contains the top, left, right, bottom and DPI scale of the drawn rectangl

Copyright © 2022 tmssoftware.com

2.11 TTMSFNCWXPDFThumbnailList

The `TTMSFNCWXPDFThumbnailList` component serves as an add-on for `TTMSFNCWXPDFViewer`. It displays the thumbnails of the currently loaded PDF and allows navigation by clicking the items. To use it, assign a `TTMSFNCWXPDFViewer` component to the `Viewer` property.

2.11.1 Properties

Property name	Description
Appearance	Appearance settings for the items.
Items	Public access to the thumbnail items.
Orientation	Change the orientation between horizontal and vertical.
Viewer	Assign the `TTMSFNCWXPDFViewer` to this property.

2.11.2 Events

Property name	Description
OnAfterDraw	Event triggered after drawing.
OnBeforeDraw	Event triggered before drawing.
OnItemAfterDrawBackground	Event triggered after drawing the item background.
OnItemAfterDrawContent	Event triggered after drawing the item content.
OnItemAfterDrawText	Event triggered after drawing the item text.
OnItemBeforeDrawBackground	Event triggered before drawing the item background.
OnItemBeforeDrawContent	Event triggered before drawing the item content.
OnItemBeforeDrawText	Event triggered before drawing the item text
OnItemClick	Event triggered when an item is clicked.
OnItemSelected	Event triggered when an item is selected

Copyright © 2022 tmssoftware.com

2.12 TTMSFNCWXPDFToolbar

The `TTMSFNCWXPDFThumbnailList` component serves as an add-on for `TTMSFNCWXPDFViewer`. This toolbar allows default PDF navigation capabilities as well as an option to add custom buttons.

2.12.1 Adding buttons

By default 9 buttons are added. It's possible to remove/edit/add buttons through the `Items` property.

```
procedure TForm1.DoCustomEvent(Sender: TObject);
begin
 //Handle any PDF related operations here. For now,
 //just show a message:
 ShowMessage('hello world');
end;
procedure TForm1.FormCreate(Sender: TObject);
var
 itm: TTMSFNCWXPDFToolbarItem;
begin
 itm := TMSFNCWXPDFToolbar1.Items.Add;
 itm.PDFRole := trCustom;
 itm.OnCustomEvent := DoCustomEvent;
 //Load a bitmap:
 //itm.Bitmap := TTMSFNCBitmap.CreateFromResource('MyBitmap');
end;
```

2.12.2 Properties

Property name	Description
Appearance	Appearance settings for the items.
Items	Public access to the thumbnail items.
Orientation	Change the orientation between horizontal and vertical.
Viewer	Assign the TTMSFNCWXPDFViewer to this property.

Copyright © 2022 tmssoftware.com

2.12.3 Events

Property name	Description
OnAfterDraw	Event triggered after drawing.
OnBeforeDraw	Event triggered before drawing
OnItemAfterDrawBackground	Event triggered after drawing the item background.
OnItemAfterDrawContent	Event triggered after drawing the item content.
OnItemAfterDrawText	Event triggered after drawing the item text.
OnItemBeforeDrawBackground	Event triggered before drawing the item background.
OnItemBeforeDrawContent	Event triggered before drawing the item content.
OnItemBeforeDrawText	Event triggered before drawing the item text.
OnItemClick	Event triggered when an item is clicked.
OnItemSelected	Event triggered when an item is selected.

Copyright © 2022 tmssoftware.com

2.13 TTMSFNCWXBarcodeDecoder

The `TTMSFNCWXBarcodeDecoder` component allows decoding multiple types of barcodes from images. It also supports decoding barcodes when connected to a `TTMSFNCWXCamera`. This component uses the quagga2 library.

2.13.1 Properties

Property name	Description
BarcodeType	Set the barcode type to read. By default `bdtCode128` is selected.
Camera	Assign a `TTMSFNCWXCamera` to this property to read barcode from the camera.

2.13.2 Methods

Property name	Description
DecodeFromBase64(ABase64; ACallback = nil)	Decode from a `base64` encoded image. If `ACallback` is assigned it's called after processing the image.
DecodeFromFile(AFileName; ACallback = nil)	Decode from a file. Not available for `WEB`. If `ACallback` is assigned it's called after processing the image.
DecodeFromImage(ABitmap; ACallback = nil)	Decode from a `TTMSFNCBitmap`. If `ACallback` is assigned it's called after processing the image.

2.13.3 Events

Property name	Description
OnDecoded	Event triggered each time the image is processed. Check the `AFound Boolean` parameter to see if the reading was successful. The `AResult string` parameter contains the decoded result.

Copyright © 2022 tmssoftware.com

2.14 TTMSFNCWXQRDecoder

The `TTMSFNCWXQRDecoder` component allows decoding QR codes from images. It also supports decoding QR codes when connected to a `TTMSFNCWXCamera`. This component uses the jsQR library.

2.14.1 Properties

Property name	Description
Camera	Assign a `TTMSFNCWXCamera` to this property to read QR codes from the camera.

2.14.2 Methods

Property name	Description
DecodeFromBase64(ABase64; ACallback = nil)	Decode from a `base64` encoded image. If `ACallback` is assigned it's called after processing the image.
DecodeFromFile(AFileName; ACallback = nil)	Decode from a file. Not available for `WEB`. If `ACallback` is assigned it's called after processing the image.
DecodeFromImage(ABitmap; ACallback = nil)	Decode from a `TTMSFNCBitmap`. If `ACallback` is assigned it's called after processing the image.

2.14.3 Events

Property name	Description
OnDecoded	Event triggered each time the image is processed. Check the `AFound` `Boolean` parameter to see if the reading was successful. The `AResult` `string` parameter contains the decoded result.

Copyright © 2022 tmssoftware.com

2.15 TTMSFNCWXAudioPlayer

The `TTMSFNCWXAudioPlayer` is a component that lets you play an audio from a URL. You can toggle the controls inside the audio player. You can also use all features from outside of the player. Make sure to familiarize yourself with the supported codes.

2.15.1 Properties

Property name	Description
Autoplay	Automatically plays an audio when loaded.
Controls	Toggles controls inside the component.
CurrentTime	Get or set the current time of the audio.
Duration (ReadOnly)	Get the duration of the audio.
IsPlaying	Checks if the audio is playing
Loop	Restart the audio when it ended.
Muted	Mute the audio.
PlayBackRate	Change how fast the audio plays.
URL	Sets the audio source to load. Can be a local file using the file:// protocol.
Volume	Set the volume of the audio.

2.15.2 Methods

Property name	Description
ForwardAudio	Skips the audio ahead with x seconds.
RewindAudio	Return x seconds in the audio.
Play	Start playing the audio.
Pause	Stop Playing the audio.

Copyright © 2022 tmssoftware.com

2.15.3 Events

Property name	Description
OnPlay	Triggers when the audio starts playing
OnPause	Triggers when the audio is paused
OnEnded	Triggers when the audio is ended
OnRateChange	Triggers when the PlayBackRate has changed
OnVolumeChange	Triggers when the CurrentTime has been changed
OnTimeUpdate	Triggers when the CurrentTime has been changed
OnError	Triggers when an error has occurred playing the audio
OnMetaDataLoaded	Triggers when the audio metadata has been loaded
OnAudioInitialized	Triggers when the audio component has been initialized

Copyright © 2022 tmssoftware.com

2.16 TTMSFNCWXDocx

The `TTMSFNCWXDocx` is a non-visual component that let's you build and generate Docx files on the fly. Almost everything that can be done in Word, can be used in this component. You can add images, tables, bookmarks, links and even a table of contents. TTMSFNCWXDocx makes it easy to build your documents. This component uses the Docx.js library.

2.16.1 Building a document

Before building a document it's important to understand the structure of the document and all the available building blocks to build your document.

2.16.2 Document

First you have the document level. At this level you can set the creator, description, title, … of the document you want to generate. This is the top level of the document. At this level you can add sections.

2.16.3 Sections

Sections are basically chapters. Every section you add, will start a new page on your document. Most of the time, you'll want different sections for a cover sheet, table of contents and your main content. At this level, you can set per section the headers and footers, the page orientation, … in these sections you can tables, paragraphs and tables of content.

Section in landscape mode:

```
section := TMSFNCWXDocx1.Document.AddSection;
section.Page.Orientation := poLandscape;
```

2.16.4 Paragraphs

Paragraphs are basically lines where you put content. Every paragraph starts on a new line. In Paragraphs you will have the possibility to add tables, text, images, bookmarks, hyperlinks, page references, line breaks… On this level you can set the heading style, set the bullet points, Alignment, …

Paragraph with text and image right aligned and heading2 style:

```
paragraph := section.AddParagraph;
paragraph.Heading := hlHeading2;
paragraph.Alignment := taRight;
paragraph.AddText('image');
paragraph.AddImage(TMSFNCBitmapContainer1.Bitmaps[0],500,500);
```

2.16.5 Tables

Tables have a collection of rows. These rows contain a collection of cells. A Table always must have rows and cells otherwise it won't be added to the document. At this level you can set table width and the indenting.

Copyright © 2022 tmssoftware.com

Example of a table:

```
table := section.AddTable;
 for i := 0 to 3 do
 begin
 tableRow := Table.AddRow;
 for j := 0 to 3 do
 begin
 tableCell := tableRow.AddCell;
 paragraph := TableCell.AddParagraph;
 paragraph.AddText('row ' + IntToStr(i) + ' cell ' + IntToStr(j));
 end;
 end;
```

This code snippet generates following table:

row 0 cell 0	row 0 cell 1	row 0 cell 2	row 0 cell 3
row 1 cell 0	row 1 cell 1	row 1 cell 2	row 1 cell 3
row 2 cell 0	row 2 cell 1	row 2 cell 2	row 2 cell 3
row 3 cell 0	row 3 cell 1	row 3 cell 2	row 3 cell 3

2.16.6 Rows

At row level you can set the height, if the row is the header, if the rows can't be split if this doesn't fit on the page anymore, … a `Row` must always have at least 1 cell.

2.16.7 Cells

At cell level you can set the borders, the cell width, the `ColumnSpan` / `RowSpan`, shading, … Here you can add `Paragraphs`, or another `Table`. Cells do not need to be filled with children.

2.16.8 Text

Here you can add your text. You can set a ton of properties like font, color, size, underline, … If you add linebreaks inside text, these will just be filtered away and not shown. It's better to add a new paragraph or add a new text and set the linebreak property on how many lines of whitespace you want.

Text Example:

```
text := paragraph.AddText('Hello world');
text.Font.Color := gcRed;
text.Font.Size := 50;
Text.Font.Name := 'arial';
```

2.16.9 Image

You can add an image to a paragraph. You can set the bitmap and it will render the image. It's important to set the transformation (height and width) to the desired values. These are by default at `100`. You can choose where you want to

Copyright © 2022 tmssoftware.com

position the image by using the floating property. To enable this, you'll have to set the `FloatingEnabled` to true. In the floating class you'll find the properties margins, horizontal and vertical positions. These will allow you to place the item where you want.

This code snippet generates an image that will align in the top right corner:

```
i := Paragraph.AddImage;
i.Bitmap.Assign(WebImageControl1.Picture);
i.Transformation.Width := 100;
i.Transformation.Height := 100;
i.Floating.HorizontalPosition.Relative := hpRightMargin;
i.Floating.HorizontalPosition.Align := hpaInside;
i.Floating.VerticalPosition.Align := vpaInside;
i.Floating.VerticalPosition.Relative := vpTopMargin;
i.FloatingEnabled := true;
```

2.16.10 Bookmark & Internal Hyperlink

A Bookmark is used for an internal hyperlink. Adding a bookmark and setting its text will display it as simple text. However, with the internal hyperlink you will be able to link to this text. You can do this by setting the ID of the bookmark and referencing it in the internal hyperlink.

Bookmark example:

```
paragraph.AddBookmark('titleBookmark', 'Demo: A docx.js Story');
```

Referencing Bookmark:

```
internalHyperlink := paragraph.AddInternalHyperlink;
internalHyperlink.Anchor := 'titleBookmark';
```

2.16.11 External Hyperlink

This is used to link to other documents, web pages, … You can set an alternative text by adding extra text elements to the children collection of the external hyperlink. This is done like so:

```
externalHyperlink := paragraph.AddExternalHyperlink;
externalHyperlink.Link := 'https://www.tmssoftware.com';
```

2.16.12 Table Of Contents

This is autogenerated content that will show all headers and their pages. You can set it to auto link to the corresponding heading or not. Note that using this autogenerated content will give a warning that this has to be generated when opening word.

2.16.13 PageNumber

Page numbers have the same properties as the text. The only exception is when adding a page number, you'll have to supply a format to show the text.

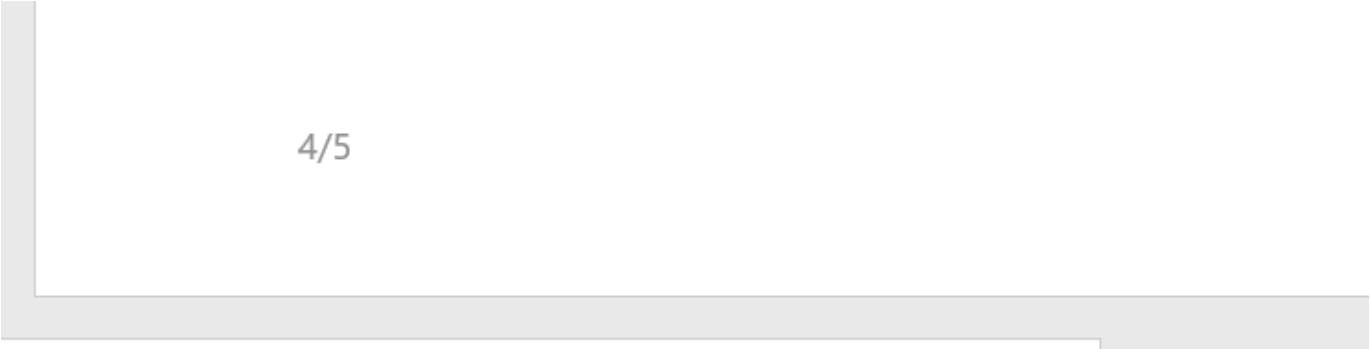
Copyright © 2022 tmssoftware.com

Example

```
paragraph.AddPageNumber('page %s of total pages %s');
```

the first %s is the current page and the second one is the total pages. The total pages is optional here.

The standard format for pagenumbering is the following layout:



2.16.14 Headers and footers

For headers and footers, it's important to know that they are only used inside that section. You have respectively the first, even and default headers and footers. So, you can set it however you want. These footers have the same options as the sections for adding items. You can add paragraphs and tables here as suited for your needs. You can access these headers and footers like this:

```
paragraph := section.Footers.DefaultFooter.AddParagraph;
```

This will add a paragraph to all footers in that section.

2.16.15 Templating

This component let you construct documents and export them to a JSON. You can use this json to persist where you want. And when you need it, you can simply load it in. For every document child you can set an ID. You can search the document for this ID. This will return the Object as a `TPersistent`. This means that you will have to cast it to the correct object. Example of using a template:

```
AsText(TMSFNCWXDocx1.FindByID('Company') as TTMSFNCWXDocxChild).Text :=
'MailMerge';
```

2.16.16 Generating Docx

You can generate the documents as a `base64` string or save it directly to a file. To do this, you'll have to subscribe to the `OnDownloadAsBase64` or `OnDownloadAsFile` events. To start the generation, you'll need to use the `GetDocxAsFile` and `GetDocxAsBase64` procedures. The `GetDocxAsFile` requires a filename with full path to start generating Docx files

Copyright © 2022 tmssoftware.com

2.17 TTMSFNCWXContainer

The `TTMSFNCWXContainer` is a component to create your own HTML elements and integrate JavaScript libraries into your applications.

TTMSFNCWXContainer

2.17.1 Adding links

To integrate external libraries, you'll need to add a link to the required JavaScript, Stylesheet or other data. To do that you'll need to add `Link` to the `Links` collection. It's Important to now that when you want to call or initialize a JS Library that you wait until the control is initialized. To do this you can call the JS code with an ExecuteJavaScript within the `OnControlInitialized` Event. This event triggers when the control is fully initialized.

Properties

Property name	Description
Async	When set to true the script will be fetched in parallel to parsing and will be evaluated as soon as it is available.
Charset	Sets the required charset (e.g. UTF-8).
Content	Set the inline content when the `kind` is `mlkStyle`.
Defer	When set to true the Script will be executed after the document has been parsed.
Enabled	If Set to true, the link will be added to the document.
Kind	Set what kind of link you want to add: `mlkLink`, `mlkScript`, `mlkStyle`.
Rel	Set the relationship between the current document and the linked document/resource. (e.g. 'stylesheet' for CSS)
Type	Specifies the media type of the linked document/resource/script.
URL	Set the link to the required document/resource/script.

2.17.2 Adding Containers

To create HTML objects, all you need to do is add a container to the `ElementContainers` collection. Here you add the HTML, CSS, JavaScript and `Actions`. With these actions you can use HTML events to trigger Pascal code.

Copyright © 2022 tmssoftware.com

Properties

Property name	Description
Actions	Collection of Action items which lets you asign Pascal events to HTML elements.
HTML	Set the HTML of this container
HTMLElementClassName	Set the name of the class for the outer HTML Element.
HTMLElementID	Set the ID of the outer HTML Element.
Margins	Set the margins for this container.
Position	Set the position of the container within the component.
Script	Set the script for this HTML Element.
Style	Set the style for this HTML Element.
UseDefaultStyle	toggling this enables the default style of the container.
Visible	Toggling this hides or shows the container.

2.17.3 Adding Actions

To allow the HTML Elements to interact with Delphi you can use `Actions`. With these actions you can listen to various events from HTMLElements and assign delphi procedures that have to be executed when these events trigger.

Properties

Property name	Description
CustomEvent	Set the eventname when using a Custom Event
Enabled	Toggle this to disable/enable the Action
Event	Set the HTML Event to listen to
EventReturnValue	Set the value that has to be returned to Delphi code
HTMLElementID	Set the id of the HTML Element to which the action has to listen.
Name	Set the name of the action
Tag	Set the tag of this action

Events

You have the OnExecute event. This event triggers when the HTML Event to which the action is listening has been triggered. This event returns a `TTMSFNCCustomWebControlEventData` Object. This object contains the Event, EventName, ID, and the customData (most of the time the `EventReturnData`). This CustomData is a JSON string.

Copyright © 2022 tmssoftware.com

2.17.4 ExecuteJavaScript

The `TTMSFNCWXContainer` component has a ExecuteJavaScript procedure. You can use this procedure to call JavaScript functions and pass parameters to them. The JavaScript string can only be 1 line. This example will show a alert.

```
TMSFNCWXContainer1.ExecuteJavaScript('alert("test")');
```

2.17.5 JavaScript Bridge

This bridge is used to trigger the actions in the `ElementContainers`. But you can access this bridge through JavaScript as well. You can call the JS function `callback()`. This function requires an EventName and the data you want to send.

This code adds an eventlistener and sends back the data from this event to the Pascal side.

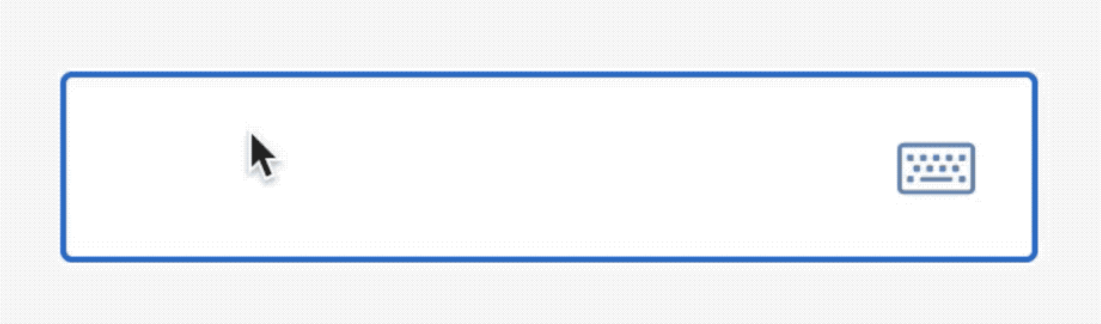
```
function InitCarousel() {
  $('#carouselExampleCaptions').on('slide.bs.carousel', function (data) {
    CallBack('CarouselSlide', data);
  });
}
```

In Delphi you can use the `OnCustomEvent` from the `TTMSFNCWXContainer` to listen to these events. This `OnCustomEvent` is the same as the `OnExecute` from the `Actions` collection.

Copyright © 2022 tmssoftware.com

2.18 TTMSFNCWXMathEditor

The `TTMSFNCWXMathEditor` is a component that enables you to quickly render math functions and edit them. this component uses the MathLive library.



2.18.1 Geting Started

Most of the functionality is inside the editor itself. You don't have to have knowledge of LaTeX. Simply start typing and use a backslash for things like square root and the editor will render the rest.

2.18.2 Keyboard Shortcuts

The component has a set of keyboard shortcuts that can be used. You can find all of them using this link.

2.18.3 LaTeX Commands

The component supports over 800 LaTeX commands. you can find a list of commands here.

Copyright © 2022 tmssoftware.com

2.18.4 Properties

Property name	Description
VirtualKeyboardLayout	changes the layout of the virtual keyboard. i.e. azerty, qwerty, ...
VirtualKeyboardMode	change how the virtual keyboard is activated. i.e. Manual, on focus, ...
VirtualKeyboardTheme	change the theme of the Keyboard
VirtualKeyboards	set all the virtual keyboards that need to be shown.
MathFormat	change the format to export the math function to

2.18.5 Methods

Property name	Description
CaptureScreenshot	Take a screenshot of the component. Triggers the OnCaptureScreenshot Event
GetMath	Exports the math function in the chosen format. Triggers the OnGetMath Event
InsertMath	Import a LaTeX string to show in the editor
Insert	Insert a string at the chosen location

There are a ton of other methods not discribed here. You can find a list of all possible methods here. These methods like an Undo can simply be called by using the the following syntax:

```
TTMSFNCWXMathEditor.Undo;
```

2.18.6 Events

Property name	Description
OnInput	Triggers when the user inputs characters
OnChange	Triggers when the field has been updated
OnGetMath	Triggers when the function GetMath has been called

Copyright © 2022 tmssoftware.com