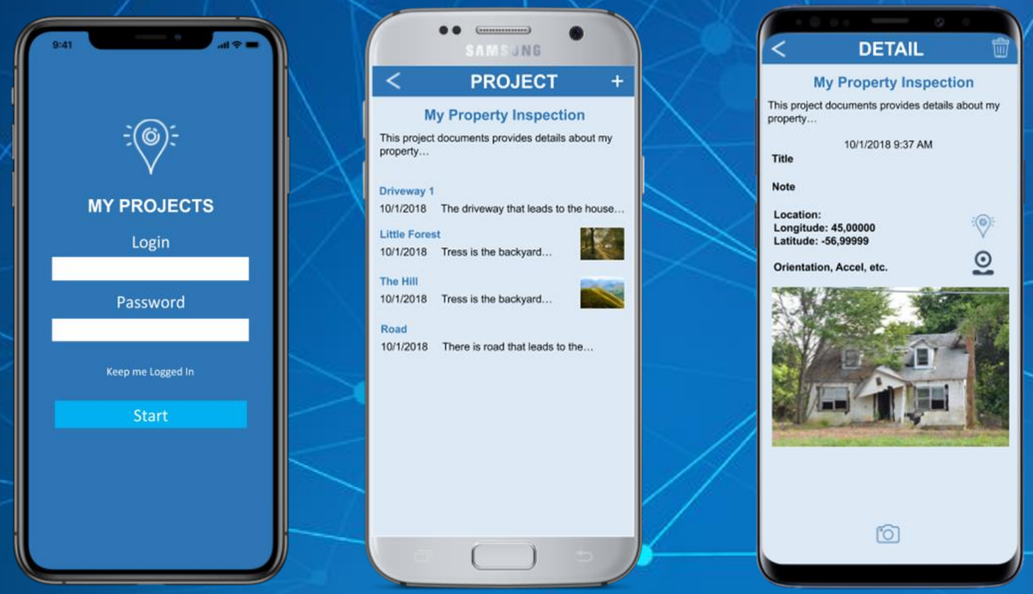


Building Mobile Apps with *FireMonkey*

Fast Track to Mobile Development Training for Delphi & RAD Studio

embarcadero®

About Jim McKeeth

- Embarcadero's Chief Developer Advocate & Engineer
- Long time software developer
- Invented and patented pattern and swipe to unlock
  - US Patent # 8352745 & 6766456, etc.
- Built thought controlled drone with Google Glass
- Host of Podcast at Delphi.org
- Lives near Boise, Idaho, USA with family & dogs
- Improvisational ComedySportz performer
- Contributor to *Internet of Things and Data Analytics Handbook*

Rest of the Team

Al Mannarino
Principal Software Consultant
Al.Mannarino@embarcadero.com

Craig Chapman
Software Consultant
Craig.Chapman@embarcadero.com

Mary Kelly
Software Consultant
Mary.Kelly@embarcadero.com

Jim McKeeth
Chief Developer Advocate
and Engineer
Jim.McKeeth@embarcadero.com

Training Overview

Training Overview

- Goals for this training
- Agenda and overview
- Where to *Git* the Code
- General Useful Information
- Specs for the App

Goals

- Help you get up to speed for mobile development with FireMonkey
- This is a workshop - we are developing an app together
- Expectations
  - Experienced with Delphi or C++Builder
  - Some experience with database development
  - Follow along with the exercises
- We are showing Delphi, but you can also do it in C++Builder
- Built with 10.3 Rio & should work with any edition
  - Including the free Community Edition

Agenda

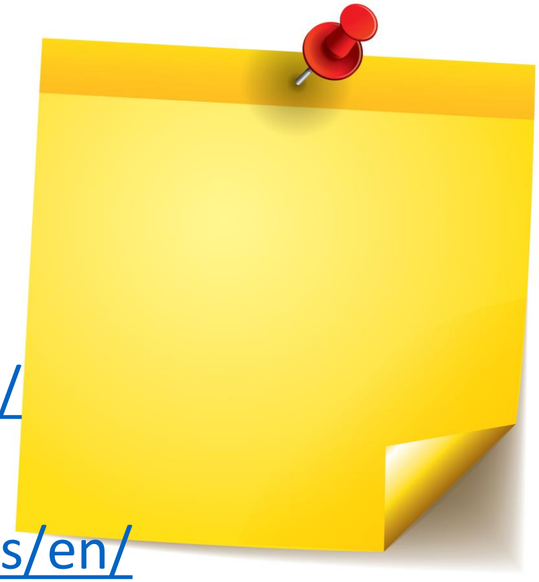
- The App Specs: What we are building
- Introduction to FireMonkey
- Setting Up for Mobile Development
- Working with Embedded InterBase: IB ToGo & IBLite
- Building the User Interface with Styles
- LiveBinding: *Connecting the* UI *to* `Data`
- Using Sensors: Hello World!
- Reporting and Sharing: *Will you be my friend?*
- Architecture Considerations: Android vs. iOS
- Publishing and Sharing your App
  - Ad-Hoc as well as Google and Apple stores

Image by Tumisu
https://pixabay.com/photo-1919292/

Git the Code

- Available on GitHub
  - https://github.com/Embarcadero/FieldLogger-FMXTraining
  - Lab Docs in GitHub ⇨ /tree/master/Lab%20Exercises
- Delphi, FMX, and SQL code
- Includes the app at various stages
- Also resources and libraries
- These slides: https://embt.co/FmxMobileAppTraining
- Master folder on Google Drive: https://drive.google.com/drive/u/0/folders/1pH-3UPc9x0I6jF1MWyw2aI1Mf2ewUkPX

Useful Information



- There are many links to the DocWikis
  - To save space we'll use shorthand
    - docwiki:RAD = docwiki.embarcadero.com/RADStudio/en/
    - docwiki:lib = docwiki.embarcadero.com/Libraries/en/
    - docwiki:code = docwiki.embarcadero.com/CodeExamples/en/
    - docwiki:ib = docwiki.embarcadero.com/InterBase/2017/en/
- Example:
  - **docwiki**:RAD/FireMonkey_Platform_Services
  - **http://docwiki.embarcadero.com/**RADStudio/en/FireMonkey_Platform_Services
- You have a copy of the slides and there are notes with more information and comments in the "speaker notes" section

App Specs

- Project log collection application
- Uses Embedded InterBase ToGo (or the free IBLite)
- Database has projects with child log entries
  - Log entries include:
    - DateTime, Picture, Geolocation, Orientation, Accelerometer, User notes
- Screens
  - Edit project details
  - Add logs to project
  - Browse & edit projects & log entries
  - Reporting
    - Export project with log entries as JSON or HTML
    - Save to file or share via email, etc.

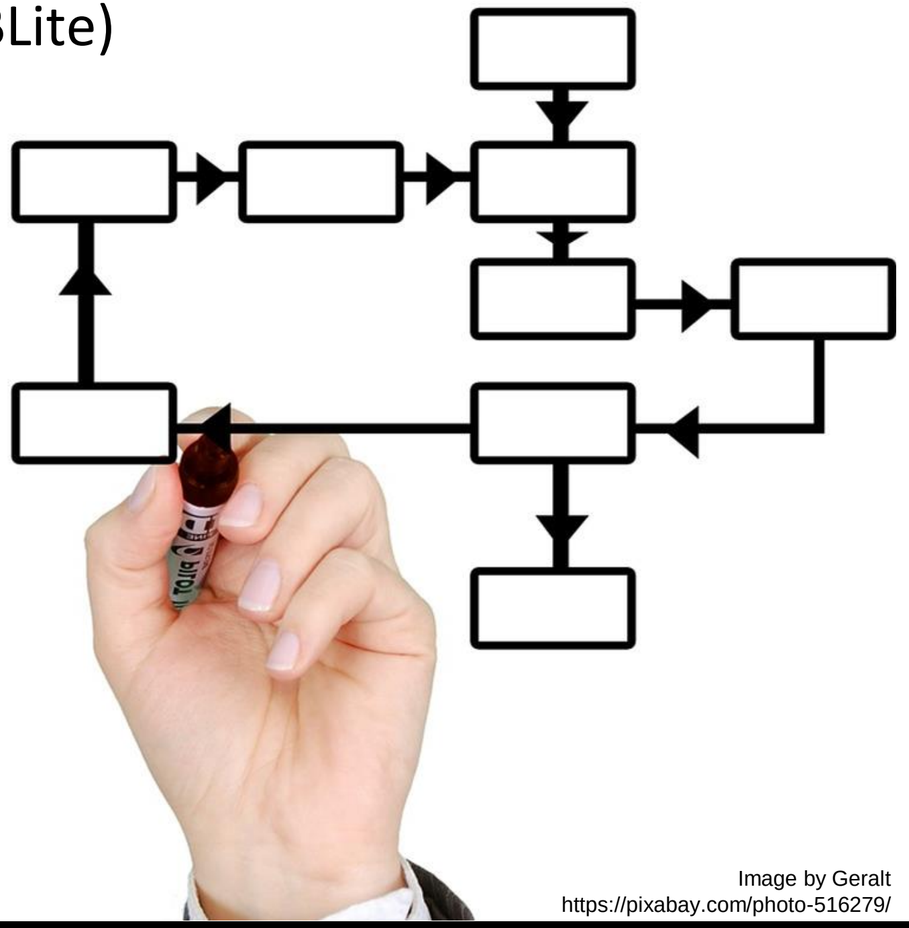


Image by Geralt
https://pixabay.com/photo-516279/

Up Next...

Introduction to FireMonkey

The cross platform application development framework!

Introduction to FireMonkey

The cross platform application development framework!

Windows, iOS, macOS, Android, and Linux

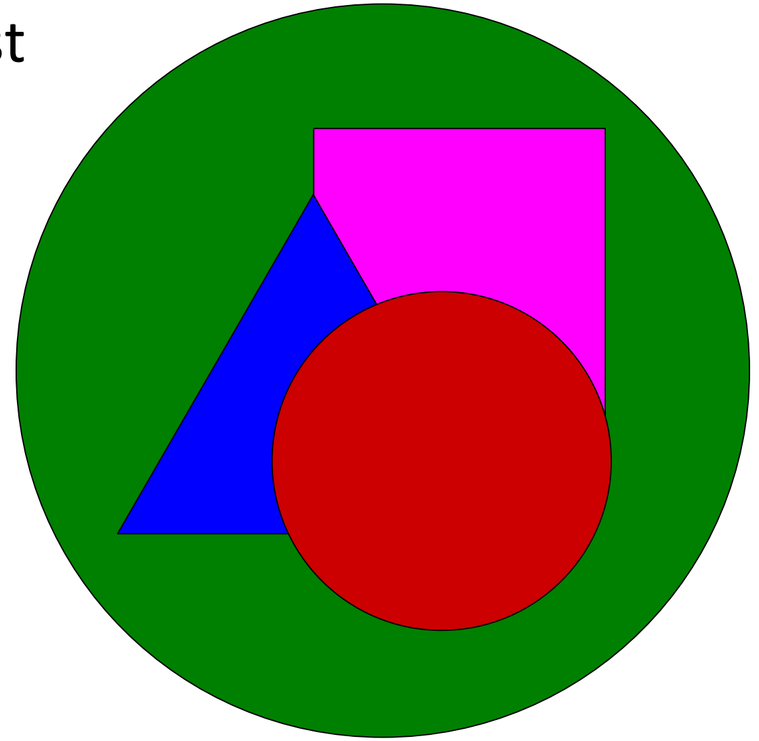
In this section we cover

- What is FireMonkey?
  - The FMX Form
  - FMX Layouts
- FireMonkey Platform Services
  - Platform Default Behaviors
- FireUI - Technology to Fine Tune Your UI
- FMX Compared to VCL
- Building Your First FireMonkey App

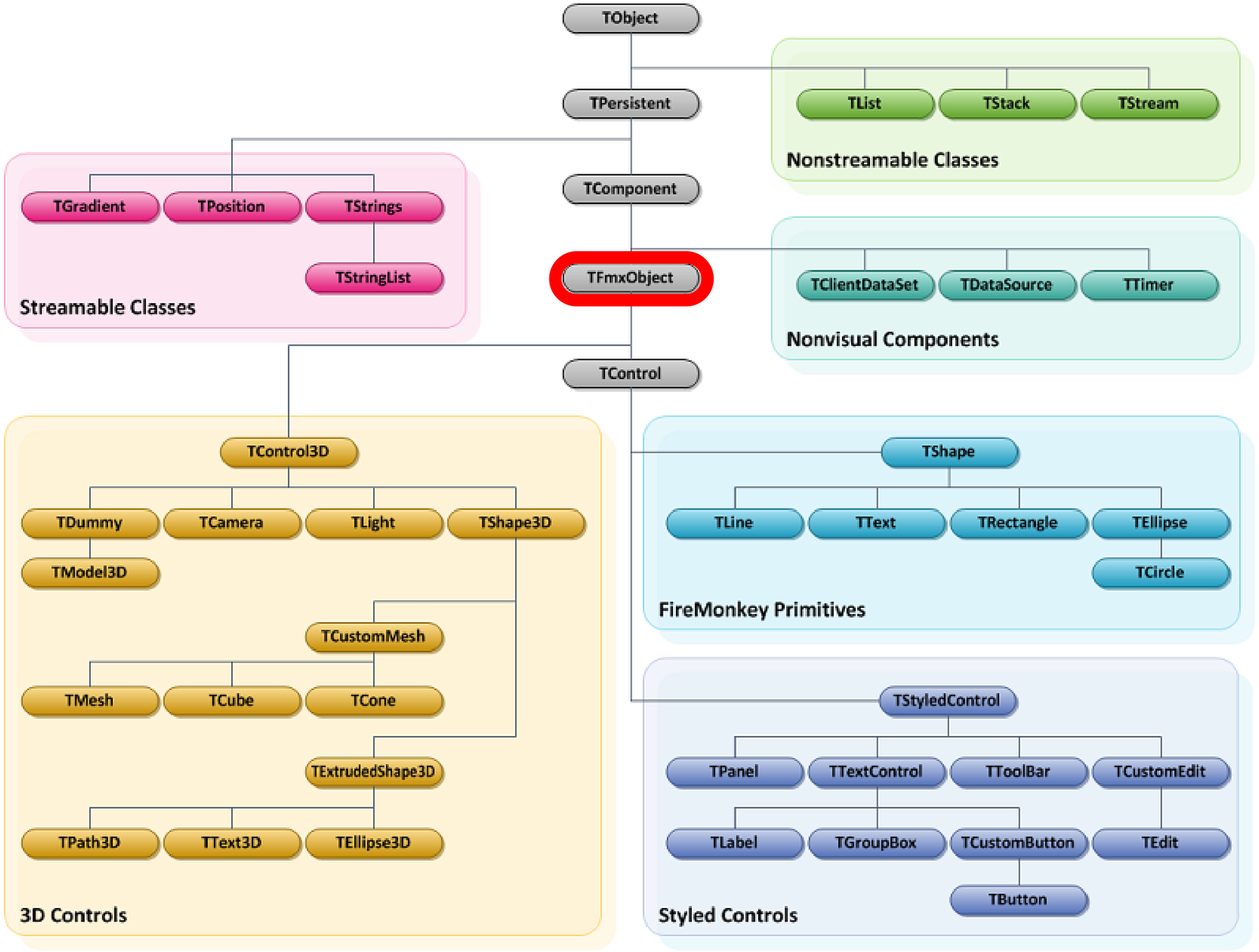
Follow along: https://embt.co/FmxMobileAppTraining

FireMonkey is Similar to VCL

- Your VCL experience is applicable with FireMonkey
  - Many of the same components and concepts exist
  - It is not a 1:1 mapping of the VCL
    - Eg: TLabel.Text instead of TLabel.Caption
- Designed to be cross-platform:
  - iOS, Android, macOS, & Windows
  - Other platforms like Linux via 3rd parties
  - Cross platform is in its DNA
- Still uses the **RTL** you know and love



FireMonkey Object Hierarchy



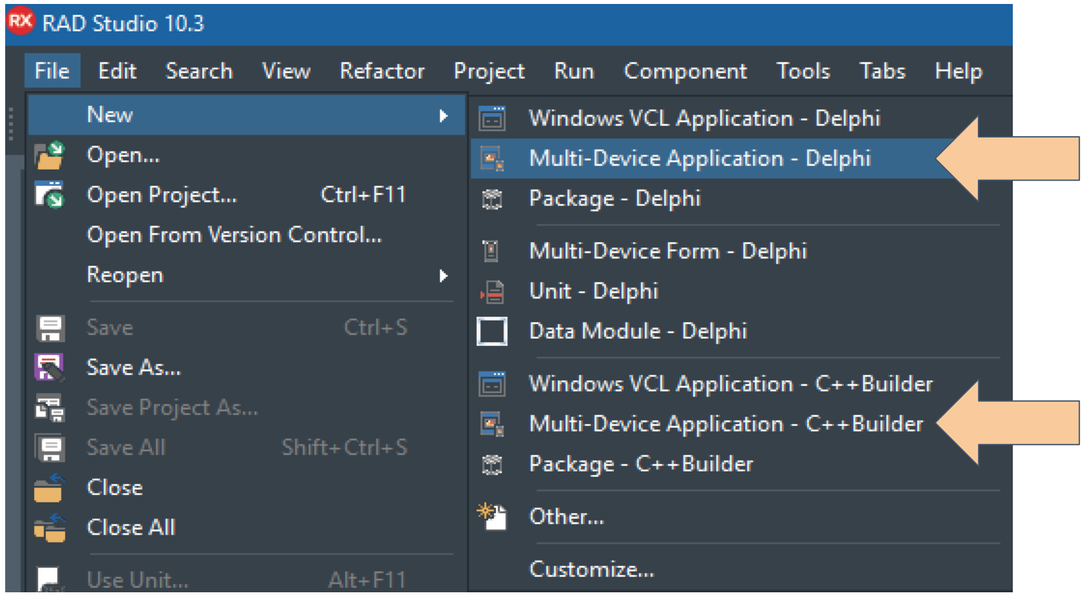
FireMonkey Overview

- FireMonkey also includes platform services and other non-visual components
- Rendered by GPU
  - Uses DirectX on Windows
  - OpenGL on macOS
  - OpenGL ES on iOS & Android
  - Many animations and graphical effects
- Check out the Quick Start Guide
  - docwiki:RAD/FireMonkey_Quick_Start_Guide_-_Introduction

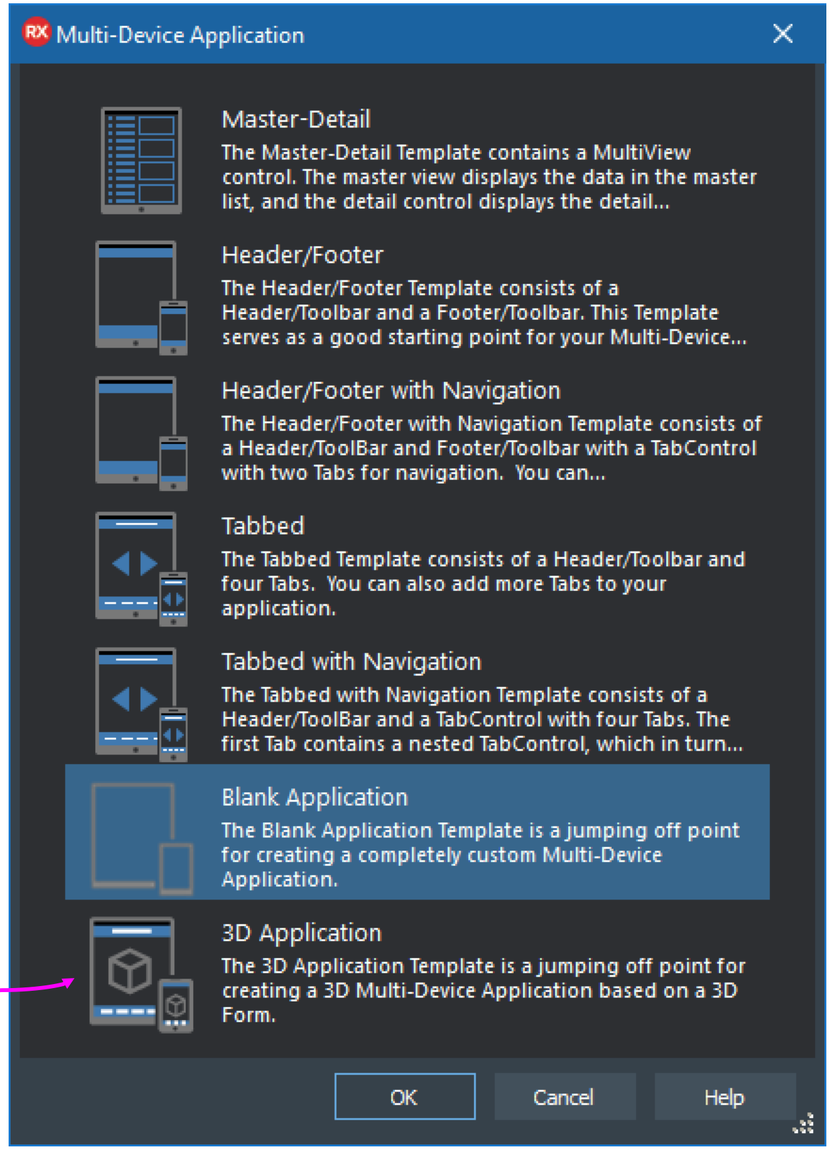
The FireMonkey (.FMX) Form

- Uses floating point numbers for positions, sizes, etc.
- Supports animation and graphical effects
- Very flexible controls: You can do more with fewer components
- Many different layout & alignment options
  - TLayout, TFlowLayout, TGridLayout, TGridPanelLayout, TScrollBox, & TScaledLayout
  - None, Top, Left, Right, Bottom, MostTop, MostBottom, MostLeft, MostRight, Client, Contents, Center, VertCenter, HorzCenter, Horizontal, Vertical, Scale, Fit, FitLeft, FitRight
- Robust and customizable styling system
- All components are nestable: Instead of TBitBtn, put TImage on TButton
- The .FMX file is very similar to a VCL .DFM file
- The Enumerated properties are Scoped Enums, and don't typically have type prefixes (i.e. Align of Client vs. alClient)

Creating a FireMonkey Project



- Listed as “Multi-Device Application”
- Available for Delphi or C++Builder
- Presents you a dialog to choose a template
- In some versions 2D FMX is called HD



3D FMX applications are different from 2D, & won’t be covered.
You can learn more about them in the DocWiki though!

Hide/Show Non-Visual Components

Design Style
Platform style to show in designer

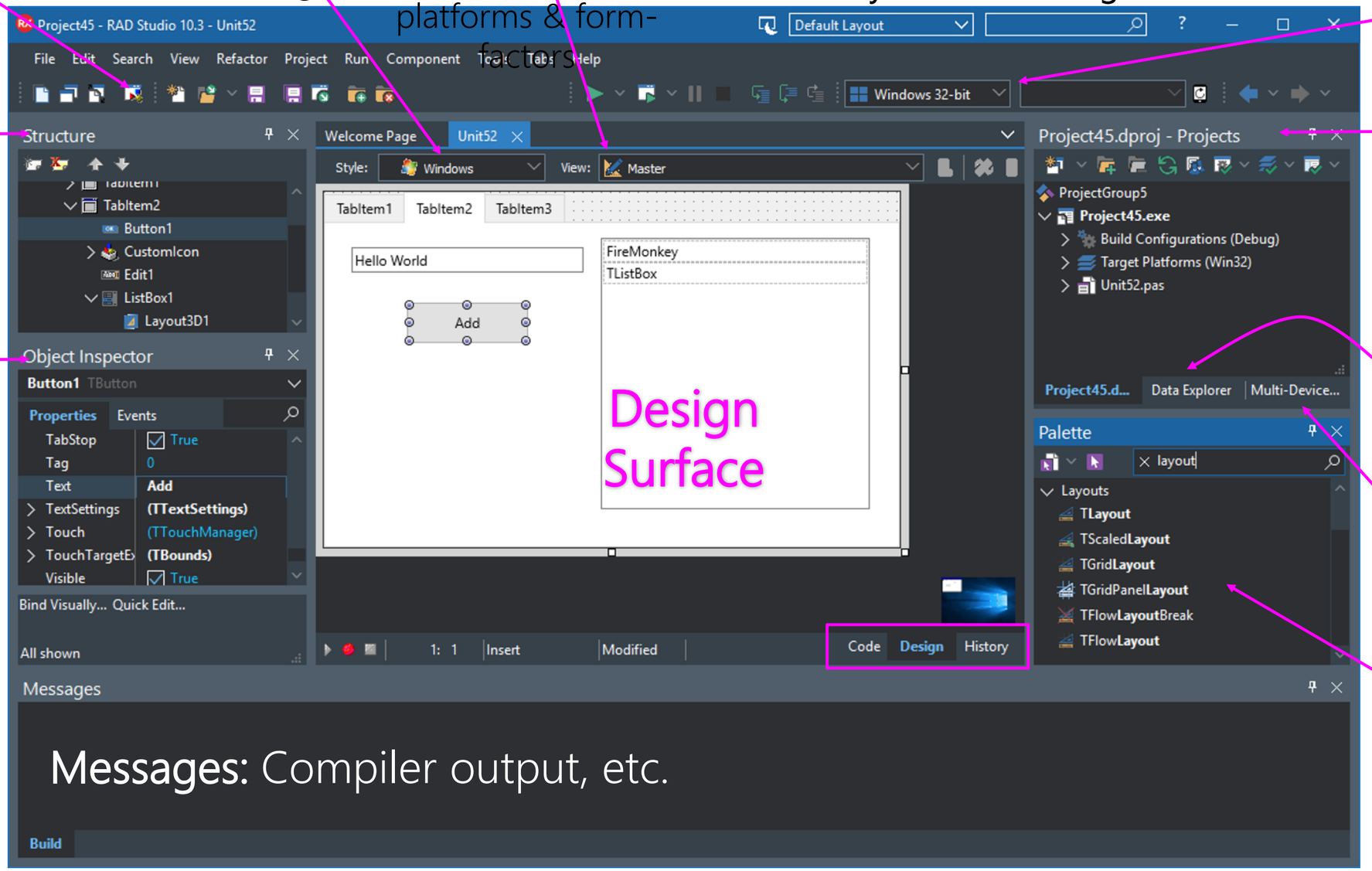
Custom Views
Created & edited for specific platforms & form-factors

Theme **Layout** **IDE Insight**

Platform Selector
Choose target for compiler, and device for deployment.

Structure view
Hierarchical view of visible and non-visible components on form

Object Inspector
View and edit properties and events of selected component(s) or form.
Notice the **TButton** has a **Text** property instead of **Caption**, otherwise very similar to **VCL**.



Project Manager
Hierarchical view of all files in project or project group

Data Explorer
Access & explore data sources

Multi-Device Preview
Preview your UI across different form factors

Palette
Only shows FireMonkey controls. They might appear disabled if not supported on the

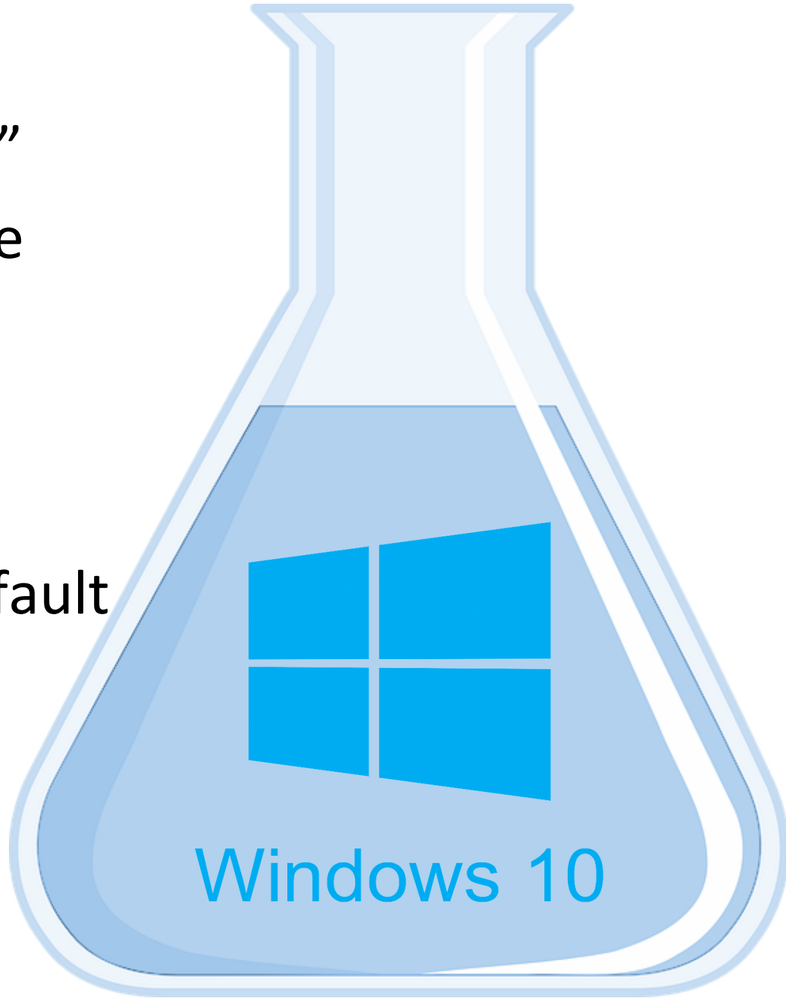
Messages: Compiler output, etc.

Understanding Platform Default Behavior

- Many properties have an option of PlatformDefault value
  - This will change the value based on the platform
  - Tab Controls PlatformDefault property http://embt.co/TutorialTabPages
- The Style can apply properties too based on platform
  - This is controlled with the StyledSettings property
  - Settings text parameters http://embt.co/SettingTextParameters
- Change the ControlType property from Styled to Platform
  - Currently supporting iOS, Android, and Windows coming soon
  - More information: http://embt.co/FMXNative

Lab - Hello FireMonkey

- Create a FireMonkey app for Windows
- Add a TTabControl, then right click and “Add TTabItem”
- Include TLabel, TButton, TEdit, & TListBox on one of the TTabItems
- Notice TLabel & TButton have **Text** instead of **Caption** properties
- Notice the TTabControl has TabPosition of PlatformDefault
- In the OnClick event handler add the following code:
  - ListBox1.Items.Add(Edit1.Text);
- Compile and run your app on Windows



Beaker image https://pixabay.com/photo-23417/

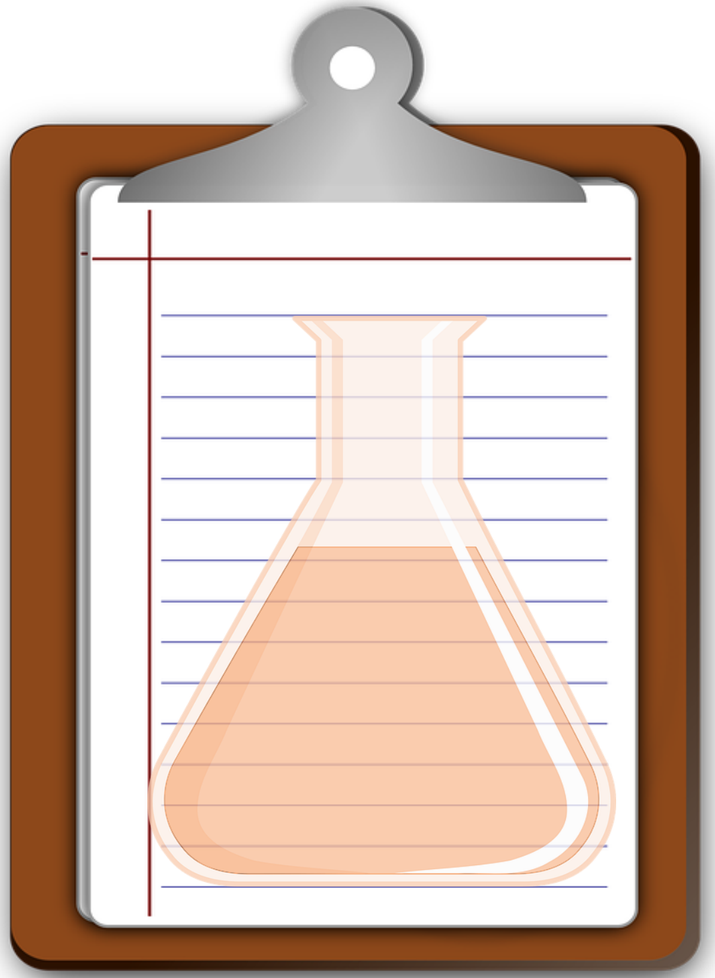
FireMonkey Platform Services

- A FireMonkey interface that defines functionality that might or might not be implemented on a particular run-time platform
  - Allows for different functionality and implementation per platform
- FireMonkey implements over 50 platform services
- You can implement your own platform services
  - Use TPlatformServices.AddPlatformService and TPlatformServices.RemovePlatformService
  - For example, you can unregister one of the built-in platform services and replace it with a new implementation of the platform service that is tailored to fit your needs.
- More information on Platform Services http://embt.co/PlatformServices

Lab - FireMonkey Platform Services

- Working with your lab from earlier, add Clipboard support
- Take a look at http://embt.co/PlatformServices
- Add FMX.Platform & FMX.Clipboard to your uses clause
- Interact with the clipboard via Platform Services

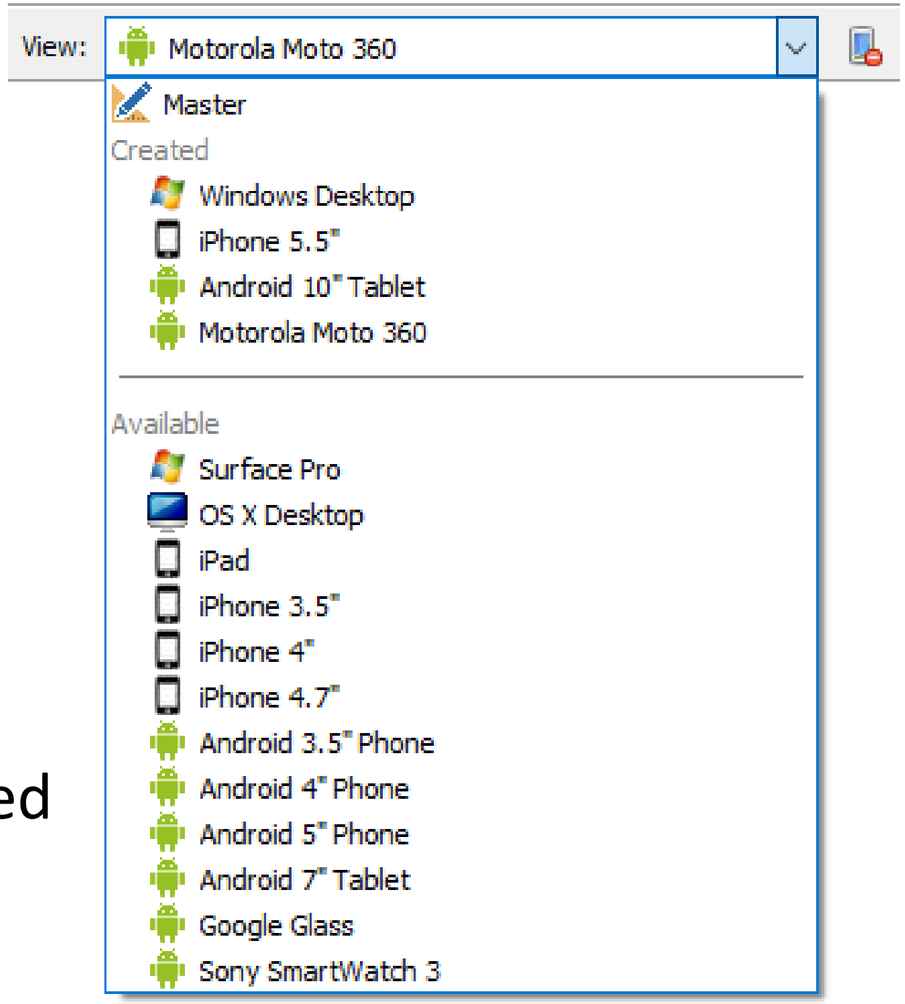
```
uses
  FMX.Platform, FMX.Clipboard;

procedure TFormFMXLab2.Button2Click(Sender: TObject);
var
  ClipboardService: IFMXExtendedClipboardService;
begin
  if TPlatformServices.Current.SupportsPlatformService(
              IFMXExtendedClipboardService, ClipboardService)
  then
  begin
    if ClipboardService.HasText then
      ListBox1.Items.Add(ClipboardService.GetText);
    ClipboardService.SetText('Hello FireMonkey');
  end;
end;
```



Clipboard image https://pixabay.com/photo-155885/

FireUI Device Views

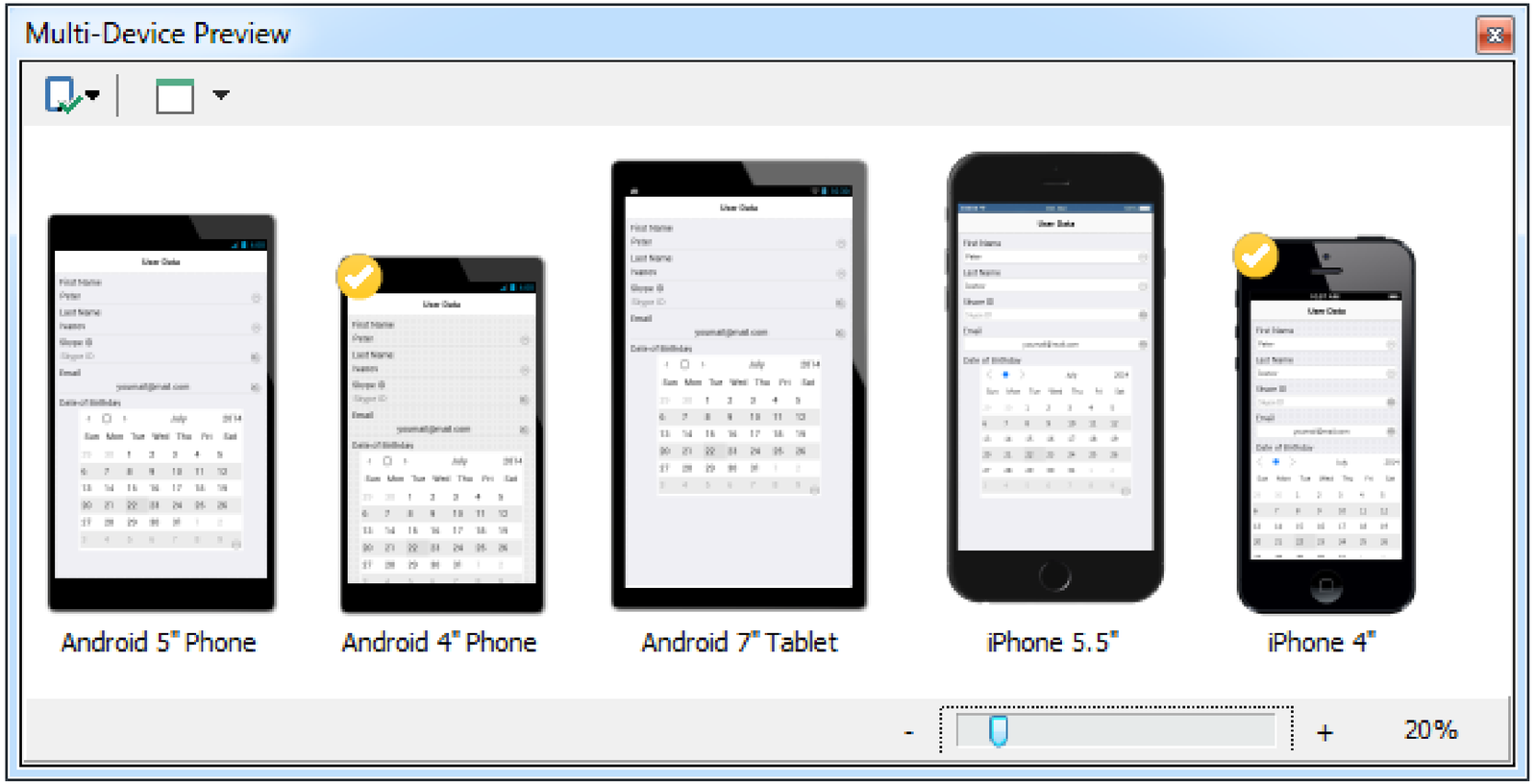
- Allows you to add platform specific customized views to your layout
- Each device specific view is a collection of changes to the master view
- Allows for customization based on OS, form factor, etc.
- Easily customize to add your own devices
- At runtime the closest view is automatically used



docwiki:RAD/Using_FireMonkey_Views

FireUI Multi-Device Preview

- Accessible via: View > Tool Windows > Multi-Device Preview

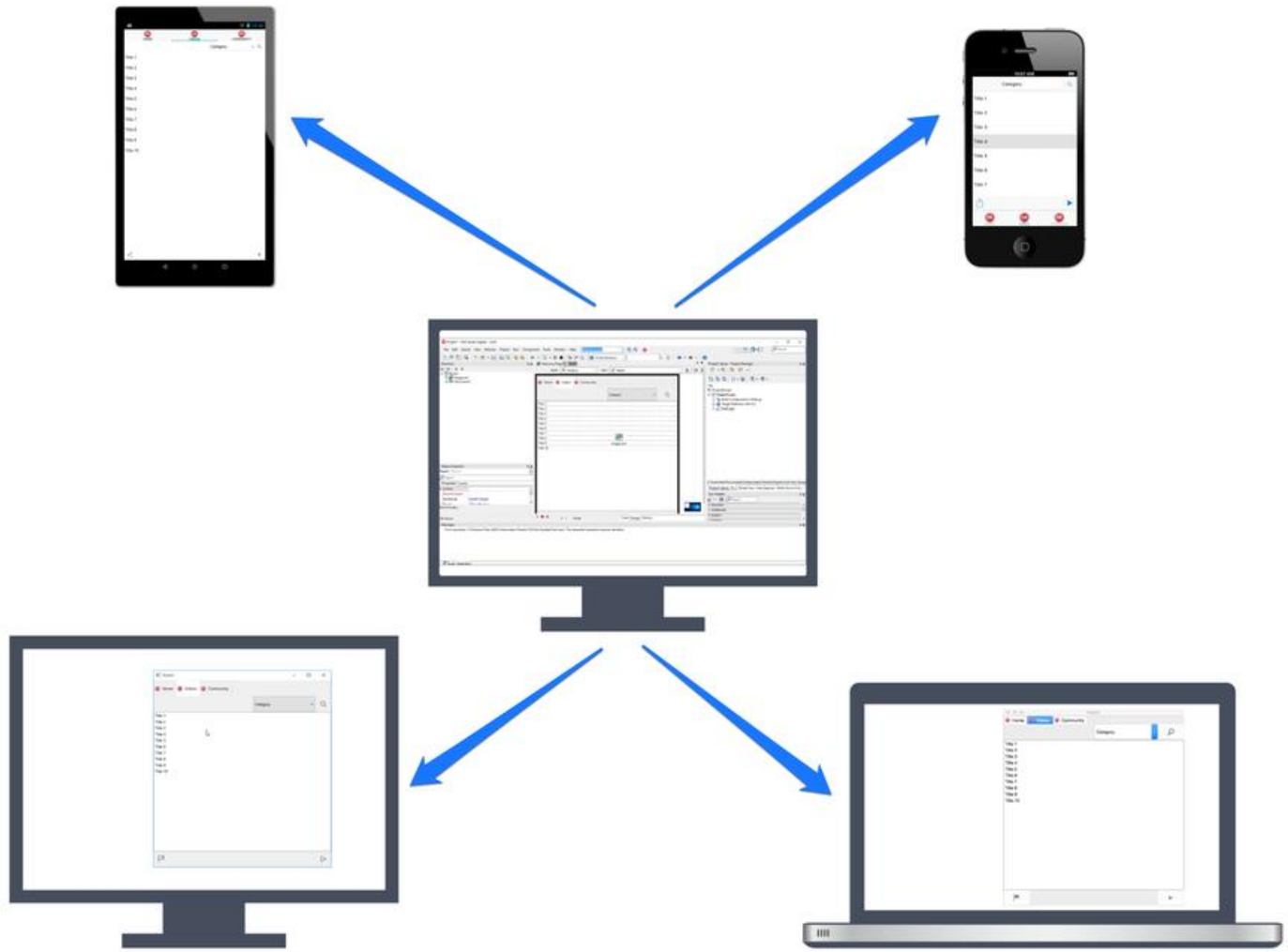


- docwiki:RAD/Multi-Device_Preview

FireUI Live Preview

- Live UI preview on physical device
- Uses app tethering technology to connect over local network
- Preview app available via the app store
- Recompile from code to add custom components

docwiki:RAD/FireUI_Live_Preview

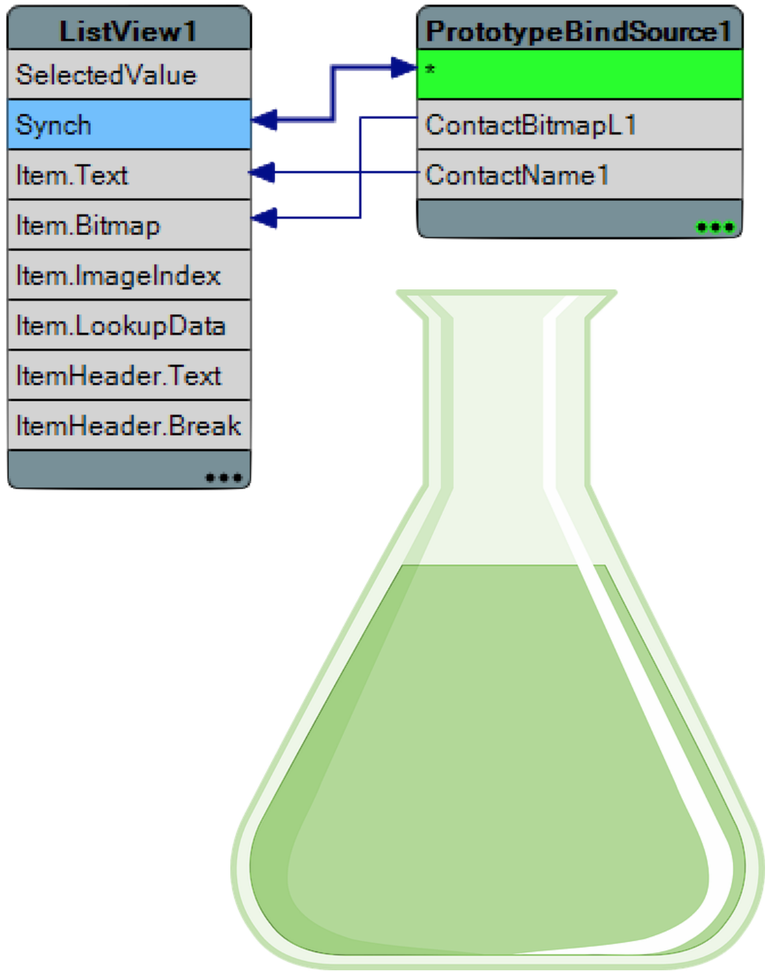


Working with Databases

- *(Just an introduction - more on this later)*
- FireMonkey works with any database components
- FireDAC is the new, high performance, cross platform, multi-database component framework that is included “in the box”
  - On mobile FireDAC supports SQLite, IBLite (Free), & IBToGo (IBToGo adds encryption and more for a license fee)
  - FireDAC Overview: embt.co/FireDACOverview
  - FireDAC Mobile Tutorial: embt.co/FireDACMobile
- FireMonkey doesn’t have specific data aware controls, instead any component can be connected to any data via LiveBindings
  - embt.co/LiveBindings

Lab - FireMonkey LiveBindings

- Add a TPrototypeBindSource
  - Dbl Click -> Add ContactBitmapsL & ContactNames
- Add TListView
  - Find the ItemAppearance.ItemAppearance property
    - Change it from ListItem to ImageListItem
  - Set ItemAppearanceObjects.ItemObjects.Visible to False
- Right Click ListView1 -> Bind Visually…
  - Connect:
    - Sync -> *
    - Item.Text -> ContactName1
    - Item.Bitmap -> ContactBitmatL1



delphi.org/2016/10/firemonkey-vs-vcl/

More FireMonkey UI

- More on layouts and UIs https://embt.co/MoreFmxUI
- DocWiki
  - docwiki:RADStudio/**FireMonkey**
  - docwiki:RADStudio/**FireMonkey_Quick_Start_Guide_**-_Introduction
  - docwiki:RADStudio/**FireMonkey_Components_Guide**
  - docwiki:RADStudio/**FireMonkey_Applications_Guide**
  - docwiki:RADStudio/**Multi-Device_Applications_Index**
  - docwiki:RADStudio/**FireMonkey_Tutorials**
  - docwiki:RADStudio/**FireMonkey_3D**
  - docwiki:RADStudio/**Tutorial:_Creating_a_FireMonkey_3D_Application**
  - docwiki:RADStudio/**FireMonkey_Component_Library**
  - docwiki:Libraries/**FMX** **Unit List**

In this section we covered

- What is FireMonkey?
  - The FMX Form
  - FMX Layouts
- FireMonkey Platform Services
  - Platform Default Behaviors
- FireUI - Technology to Fine Tune Your UI
- FMX Compared to VCL
- Building Your First FireMonkey App

Up Next...

Setting Up Mobile Development

Focus on Android, but also touch on iOS