Delphi XE2 xPlatform FireMonkey aka FMX

Jeroen Pluimers
jpluimers@better-office.com
Edwin van der Kraan
evdkraan@better-office.com
better office benelux

EKON15 2011
DE, Düsseldorf, 20111028

FireMonkey on Windows and Mac

- Share the same FMX classes
- Have very similar runtime behaviour
- Windows
  - DirectX
  - Can run in a Windows VM
- Mac
  - OpenGL
  - Physical Mac is needed

Mac / Windows cross platform demo

- FishFact using InterBase
- Getting the components right
- Getting the data binding right
- Getting the initialization right
- Debugging

Exercise

- Do something similar as the FishFact FireMonkey demo
- Use any of these databases.tables that has images (fieldname: blob type)
  - DBDEMOS.ANIMALS
    - (BMP: BMP image)
  - DBDEMOS.VENUES
    - (VENUE_MAP: BMP image, REMARKS: text)
- Try a master detail relation between for instance these two tables
  - DBDEMOS: CUSTOMER -> ORDER
  - EMPLOYEE: DEPARTMENT -> EMPLOYEE
  - MASTAPP: CUSTOMERS -> ORDERS -> ITEMS

FireMonkey on iOS

- Uses Xcode for compiling and Debugging
  - Needs a way of sharing files between Mac and Windows
- Uses the FreePascal (FPC) compiler for compiling and Xcode integration
  - FPC is used in “Delphi” mode
    - Requires FPC 2.5.1
- You use Delphi on Windows to develop and pre-test your app
- You use Xcode on Mac to debug and fine-tune your app

Limitations of FPC (and therefore FireMonkey apps on iOS)

- FPC “Delphi” language support is similar, but not the same as the Delphi compiler on Windows
- Most important shortcomings:
  - No Unit Scope Names (see Demo)
    - https://forums.embarcadero.com/thread.jspa?threadID=60270&tstart=15
  - No Generics
  - No support for .obj file format
  - No support for spaces in path names
    - https://forums.embarcadero.com/message.jspa?messageID=398897
  - Limited RTTI support (no TValue, certainly no TValueInterface)

Limitations of FPC (and therefore FireMonkey apps on iOS)

- Result:
  - FireMonkey is using FMX_*.pas on iOS
    - Basically those are copies of the FMX.*.pas files, without the Unit Scope Name
  - FireMonkey is using the unscoped RTL unit names on iOS
    - Don't by accident use System.SysUtils, but in iOS always do SysUtils
  - Limited RTL support
  - No TClientDataSet (it depends on .obj files)
  - No TDataSet (uses Unit Scope Names)
  - No Data Binding (uses Unit Scope Names)
  - No DBX JSON units (uses Unit Scope Names)
  - No IOUtils unit (use SysUtils in stead)

Current (update 1) import limitations

- Many Objective C headers have not been implemented
- Embarcadero knows
  - they are working on getting this to you as soon as they can
  - Anders Ohlsson did a lot of demos on CodeRage 6
    and just published the sample code online
  - See CodeRage video around 50:00

So why still do iOS in FireMonkey?

- Because things on iOS work very differently from Windows and Mac
  - Many things use URLs to communicate between applications
    - See CodeRage 6 demos by Anders Ohlsson
- Because it is fun doing iOS stuff
- Because developing in Delphi 1 was fun too!
- You need to get started somewhere
- This is way better than just Xcode and Objective C on the Mac
- You will get updates that are going to be better

Two basic solutions to work around this

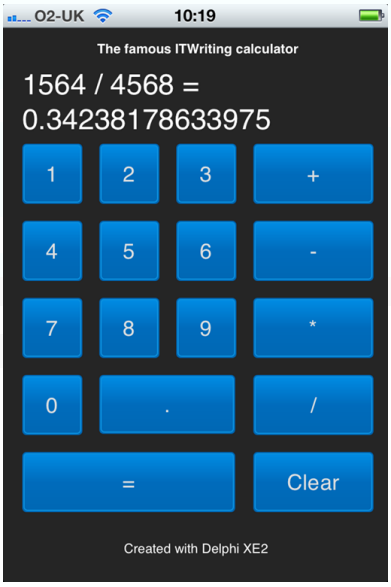
- Use {$IFDEF FPC} for iOS specific things
- Find cross platform projects that support FPC and Delphi:
  - SuperObject (JSON support)
    - http://code.google.com/p/superobject/
    - Needs a patch for FPC 2.5.1
      - http://free-pascal-general.1045716.n5.nabble.com/Unable-to-compile-JSON-SuperObject-on-FPC-2-5-1-td3258462.html
      - http://free-pascal-general.1045716.n5.nabble.com/file/n4665724/superobject_lg.patch
  - Synapse TCP/IP
    - http://www.ararat.cz/synapse/doku.php/start

JSON – xPlatform data for iOS

- Delphi JSON Viewer (Paweł Głowacki – note both "l"s are in fact "ł")
  - http://edn.embarcadero.com/article/40882
- Delphi JSON serializing/deserializing starter
  - http://stackoverflow.com/questions/3820996/delphi-2010-how-to-save-a-whole-record-to-a-file/3821990#3821990
  - Look for TBaseObject
- SuperObject serializing/deserializing starter
  - http://code.google.com/p/superobject/wiki/first_steps
  - http://2dev.info/showthread.php?4-SuperObject-A-fast-Delphi-JSON-parser
  - Look for TSuperRttiContext

Excersise

- Write yourself a nice cross platform calculator
- 2 Separate projects
  - 1 for Windows/Mac
  - 1 for iOS
- Share business logic in 1 unit across the projects
  - Easiest to have this unit in the directory of the iOS app
- Remember
  - 1024x768 == iPad
  - 320x480 == iPhone/iPod-Touch



Hot link

- iOS Clock app in iTunes made in Delphi XE2
  - http://itunes.apple.com/us/app/anders-analog-clock/id475178939?mt=8
- Consumer apps are usually 'simple' things, so start thinking that way.

References

- Anders Ohlsson
  - Blog http://blogs.embarcadero.com/ao
  - CodeRage 6 sessions
    - Multi-platform Development with FireMonkey
      - http://cc.embarcadero.com/item/28571
    - iOS Development with FireMonkey
      - http://cc.embarcadero.com/item/28553
    - iOS DataSnap using FireMonkey and InterBase
      - http://cc.embarcadero.com/item/28593
- CodeRage 6 sessions with FireMonkey
  - http://www.google.com/search?q=site:cc.embarcadero.com+coderage+6+firemonkey