CodeRage™ 8

embarcadero

Beyond the App

Jim McKeeth

Developer Relations, Embarcadero Technologies

jim.mckeeth@embarcadero.com

You Rock.
You Rule.
You Rage.

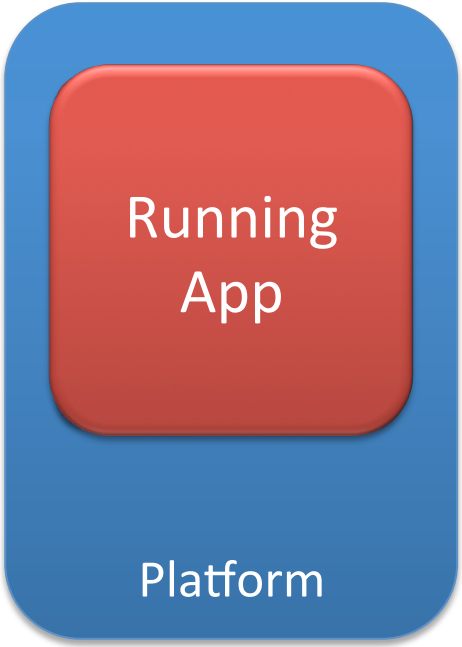
Oct 14, 2013

Agenda

- Defining an App
- Notifications
- Share Sheet
- Debug Logging
- Inter-app communication

Defining an App

- An app is a single “window” of information and interaction.
- It may contain multiple displays or screens.
- App runs in a sandbox on the platform.
- All its interaction is contained within the app.

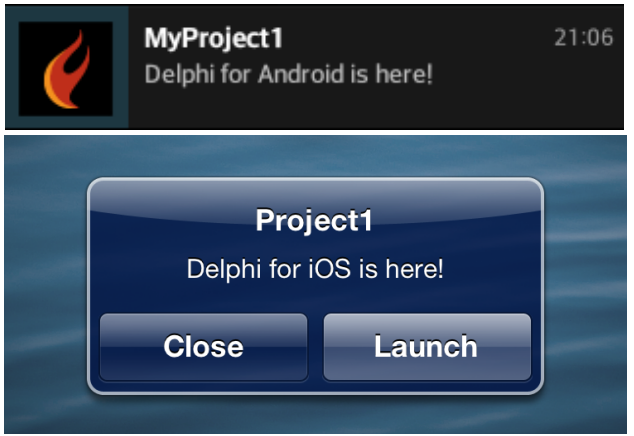


Examples of going beyond

- Notification on iOS & Android
- Badges on iOS
- Messaging between apps
- Accessing camera roll or contacts
- Other “shared storage”
- Debug logging
- Inter-app communication

Notification Service

- Accessible via the TNotificationCenter component
- Create a TNotification via the CreateNotification method
- Set properties on Notification
- Send with ScheduleNotification or PresentNotification methods



http://sn.im/xe5-notifications

embarcadero

Share Sheet

- Simple Inter-app communication
- Leverage social media accounts managed by platform
- Avoids need to communicate and authenticate with service directly
- Can send picture and/or text
- Provided via a Standard Action
- On Android it is sent to other apps
- On iOS it is sent to iOS

http://sn.im/xe5-sharesheet

Invoking Share Sheet

- Add TActionList
- Add Standard Action TShowShareSheetAction
- Assign action (to control, gesture, etc.)
- Or manually invoke by calling ExecuteAction(ShowShareSheetAction1) on a control
- Handle OnBeforeExecute event of Action to assign TextMessage and Bitmap

Debug Logging

- Log debug messages to the “console” to track internal state of app
- May be left in for shipping apps, but be careful!
- Different calls and console locations for each platform
- Good candidate for abstraction library
  - Recommended: DXLibrary’s DX.Utils.Logger
  - code.google.com/p/dx-library

Debug Logging on Windows

- Uses Windows
- Call OutputDebugString method:
  - OutputDebugString(pchar(DelphiString));
- Shows up in the Event Log window in Delphi when debugging
- A number of 3rd party debug string viewers available
  - DebugView (Microsoft Sysinternals) http://sn.im/debugview
  - GExpertsDebugWindow.exe http://sn.im/xe5-gexperts
  - CnDebugViewer.exe http://www.cnpack.org/downbuilds.php

Alternatives to OutputDebugString on Windows

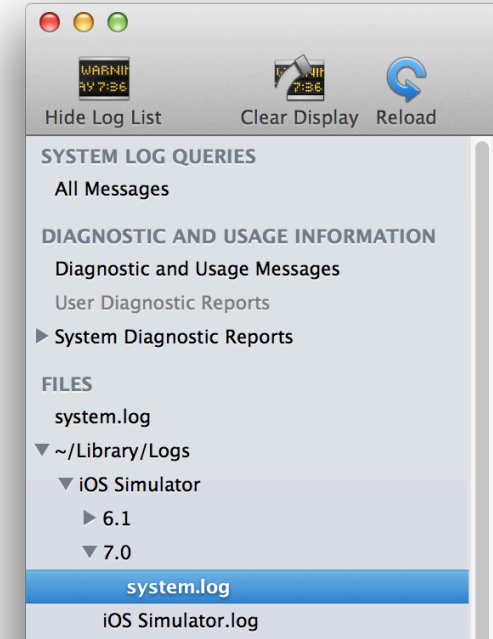
- CodeSite by Raize Software
  - www.raize.com
  - Included with Delphi XE5
- SmartInspect by GurockSoftware
  - www.gurock.com

Logging on iOS

- Use iOSapi.Foundation Unit
- Call NSLog method:
  - NSLog(PtrForObject(NSSTR(DelphiString)));
- Shows up in Console app on OS X

Viewing Logs from iOS from the Simulator

- Run the Console app (OS X)
  - Navigate to Files -> ~/Library/Logs/iOS Simulator/7.0/system.log



Viewing Logs from iOS Device

- Run Xcode
- Open the Organizer window (⇧⌘2)
- Navigate to the Console node for your selected attached device
- Whenever you are using a real device you will see a lot of other log messages

Logging on Android

- Use Androidapi.Log unit
- Call the one of the methods:
  - LOGI *Informational messages*
  - LOGW *Warning messages*
  - LOGE *Error messages*
  - LOGF *Fatal messages*
- Example:
  - LOGI(LMarshaller.AsAnsi(DelphiString).ToPointer);

Viewing Logs from Android Device or Emulator

- Use `adb` – the Android Debug Bridge (command line)
- Found in (default)
  - C:\Users\Public\Documents\RAD Studio\12.0\PlatformSDKs\adt-bundle-windows-x86-20130522\sdk\platform-tools *[add to path!]*
- Usage:
  - (from command line) `adb logcat`
- More information:
  - http://sn.im/android-adb & http://sn.im/android-logcat

Viewing Logs from Android Device or Emulator

- Use Android Debug Monitor (GUI)
- Found in (default)
  - C:\Users\Public\Documents\RAD Studio\12.0\PlatformSDKs\adt-bundle-windows-x86-20130522\sdk\tools *[add to path!]*
- Launch `monitor.bat` (replace `DDMS.bat` but both work similar)
- More information: http://sn.im/android-ddms

Logging on OS X

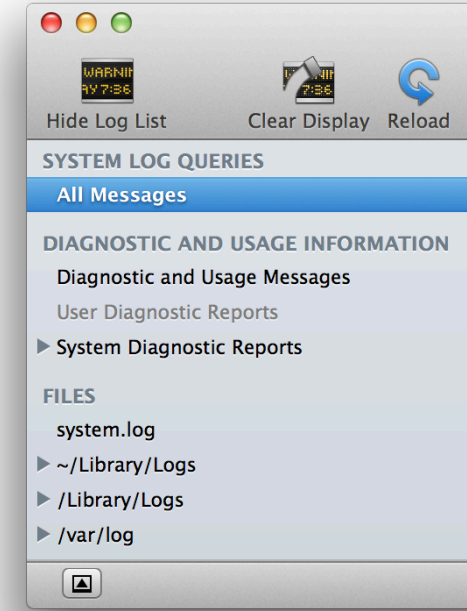
- NSLog isn't imported.
- Use the following code to import:

```
{$IFDEF MACOS}
uses Macapi.ObjectiveC, Macapi.ObjCRuntime, Macapi.Foundation;
type PNSString = Pointer;

const libFoundation = '/System/Library/Frameworks/Foundation.framework/Foundation';

procedure NSLog(format: PNSString); cdecl; varargs; external libFoundation name _PU + 'NSLog';
{$ENDIF MACOS}
```

- Call:
  - `NSLog(PtrForObject(NSSTR(DelphiString)));`

Viewing Logs from OS X

- Use Console app on OS X
- Shows up in “All Messages” node
- Alternate method for only app messages
  - Launch app via terminal
    - ~/RADPAServer/scratch-dir/[Connection]/[Project].app/Contents/MacOS/[Project]



Using DX-Library

- Use Open From Version control to save locally
  - URL: http://dx-library.googlecode.com/svn/trunk/
- Add path of to search path
  - Path to DX-Library
  - C:\Users\Public\Documents\RAD Studio\12.0\Samples\Delphi\RTL\CrossPlatform Utils
- `Uses DX.Utils.Logger`
- Call `Log(DelphiString)`

Calling Other Apps on Android

- Android uses “Intent” to call other apps

```
uses FMX.Helpers.Android, Androidapi.JNI.GraphicsContentViewText,
  Androidapi.JNI.Net, Androidapi.JNI.JavaTypes;
var
  Intent: JIntent;
begin
  Intent := TJIntent.JavaClass.init(TJIntent.JavaClass.ACTION_VIEW,
    TJnet_Uri.JavaClass.parse(StringToJString(URL)));
  SharedActivity.startActivity(Intent);
end;
```

- URL can follow special formats:
  - http, tel, sms, fb, mailto, twitter, geo, etc.

Calling Other Apps on iOS

- Use openURL method
-
```
Uses IdURI, iOSapi.Foundation, FMX.Helpers.iOS;
var
  NSU: NSUrl;
begin
  NSU := StrToNSUrl(TIdURI.URLEncode(URL));
  if SharedApplication.canOpenURL(NSU) then
    SharedApplication.openUrl(NSU);
end;
```
- URL can follow special formats:
  - http, tel, sms, fb, mailto, twitter, etc.

Summary

- Notifications with TNotificationCenter
- Share Sheet
- Debug Logging – Use DX-Library
- Inter-app communication
  - Intent or openUrl

Thank-You

- Jim McKeeth
- jim.mckeeth@embarcadero.com
- www.delphi.org/coderage8/
- Twitter @jimmckeeth