【C1】Delphi/iOSチュートリアルセッション

Delphi for iOS開発 ファーストステップ

エンバカデロ・テクノロジーズ
エヴァンジェリスト 高橋　智宏

embarcadero Developer Camp

アジェンダ

- OS X側の準備
- Windows側の準備
- 画面の作り方(メインフォームとサブフォーム)
- デバッグの基礎(ログの出力と確認)
- デバイスの回転に対応するには?
- iPhone, iPad への対応方法
- アプリのローカライズ手順
- Delphi言語の変更点(モバイル向け)

OS X側の準備

Mac, SDK, ブリッジ, 実機

モバイルアプリ実行&デバッグの仕組み

Windows上のIDE

ローカル or リモート

Windows or Mac
実行&デバッグ用ブリッジ
(RAD PAServer XE4)

iOS
シミュレータ

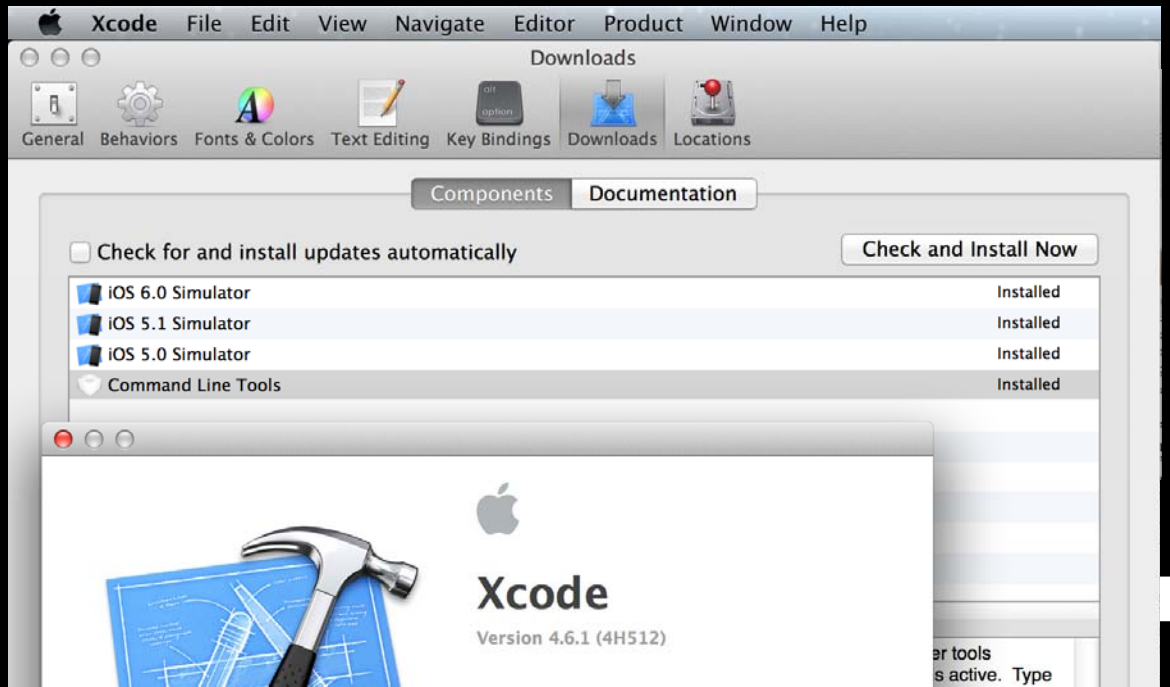
iOS実機
iOS5.1以降

Android
エミュレータ

Android実機

Macマシン

- OS X 10.8 (Mountain Lion) または 10.7 (Lion)
- Xcode 4.6.1
  - iOS 6.1 SDK & シミュレータ
  - Command Line Tools (実機向けコード署名)

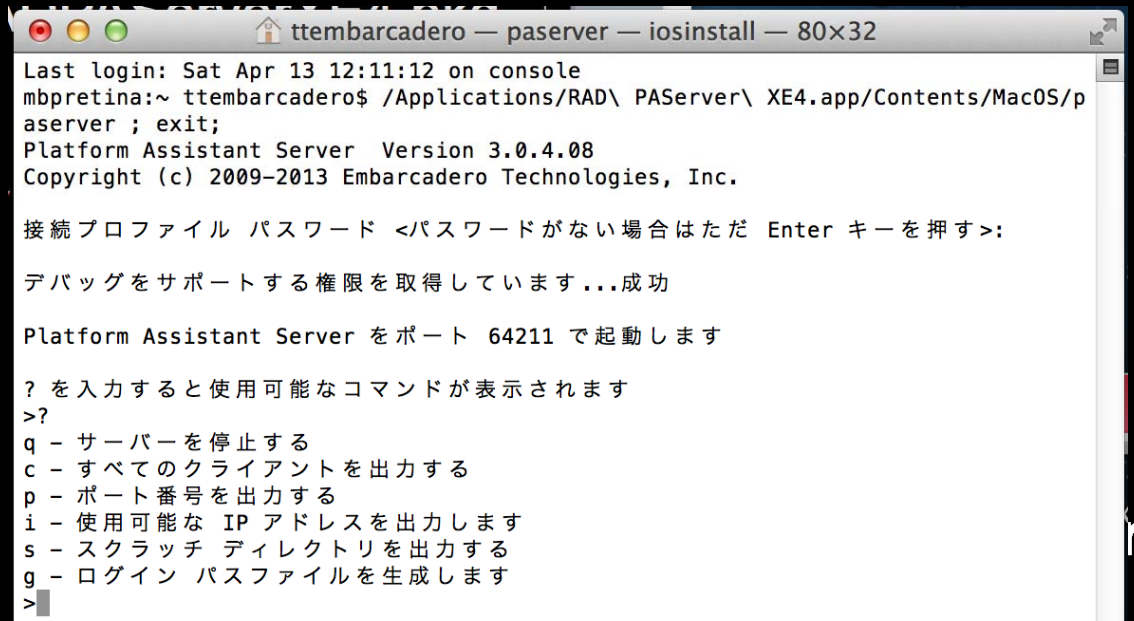


ブリッジ

- RAD PAServer XE4をOS Xにインストール
  - …¥RAD Studio¥11.0¥PAServer¥RADPAServerXE4.pkg

  をMacにコピーしてインストールを行う
  - アプリケーション – RAD PAServer XE4.app

  をダブルクリックして、Enterキーを押す
  - OS X側のファイアウォールは切っておく



実機の準備

- iOS 5.1.x～6.1.x の iPhone 4～5, iPod touch 4～5
- iOS 5.1.x～6.1.x の iPad 2～4, iPad mini
- iOS Developer [Enterprise] Program
  - https://developer.apple.com/jp/support/ios/enrollment.html
- アプリを実機に転送するための証明書
  - アプリケーション – キーチェーンアクセス.app で作成
  - iOS Provisioning Portal に登録
  - アプリケーション – キーチェーンアクセス.app に登録
- 実機のUDIDをiOS Provisioning Portal に登録
- 実機をXcodeのOrganizerに認識させる

Windows側の準備

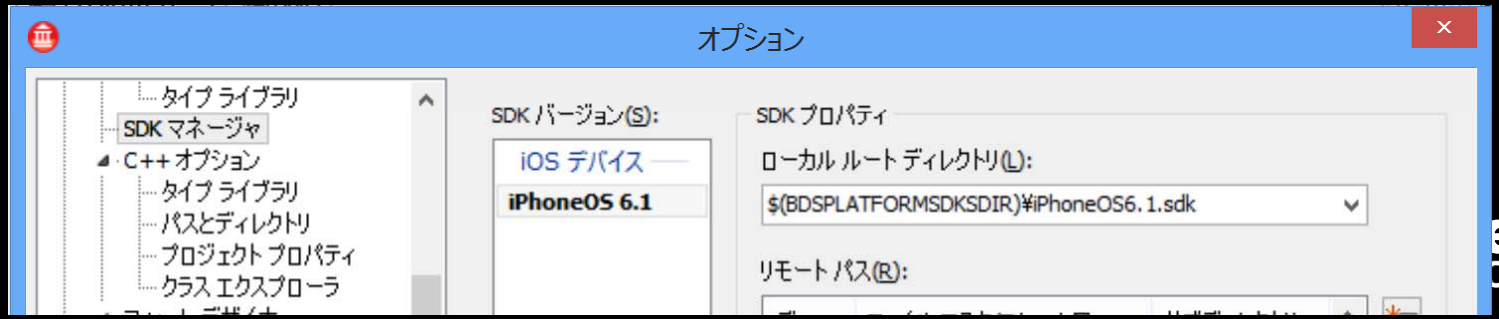
物理/仮想OS, IDE, ブリッジ接続

Windows OSの準備

- Vista, Windows 7 または Windows 8
  - 32bit または 64bit
  - 物理マシン または 仮想OS
- OS X向けのVM製品を使う場合には
  - VMware Fusion 4 または 5
    - http://www.vmware.com/jp/products/desktop_virtualization/fusion/
  - Parallels Desktop 7 または 8 for Mac
    - http://www.parallels.com/jp/products/desktop/
  - Oracle VirtualBox 4.x for OS X
    - https://www.virtualbox.org

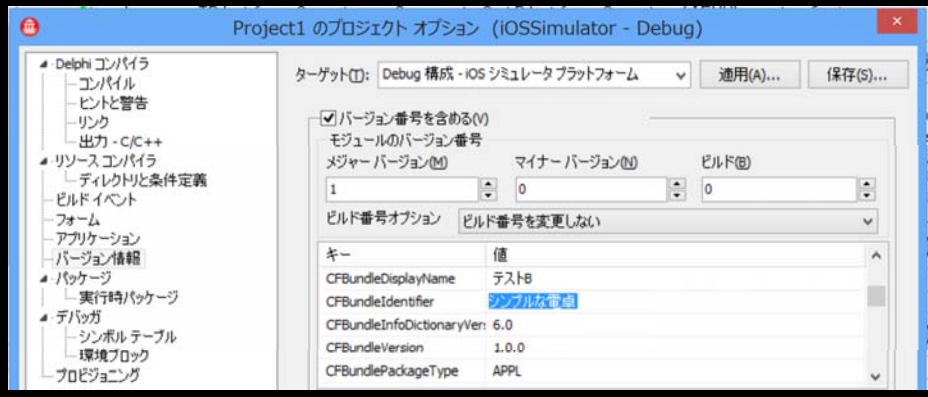
IDEの[環境オプション]の設定

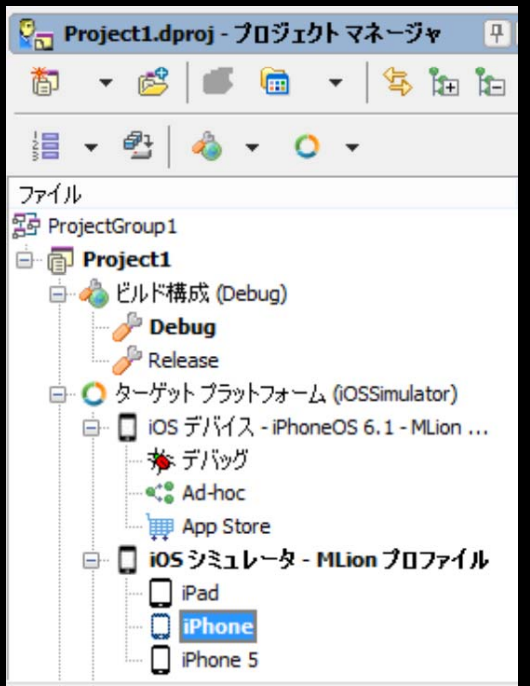
- [自動保存の設定] – [エディタファイル] を ON
- [接続プロファイルマネージャ] – [追加...]
  - プラットフォーム: OS X
- [SDKマネージャ] – [追加...]
  - プラットフォーム: iOS デバイス
  - 接続するプロファイル: 上で作成したもの
  - SDKバージョン: iPhoneOS 6.1
- iOSシミュレータ向けのSDK設定は不要!!



モバイルプロジェクト

- プロジェクト名(英数字のみ)
  - Project1.dproj
- アプリ名(日本語もOK)
  - バージョン情報の CFBundleDisplayName で設定
- ターゲットプラットフォーム
  - iOS デバイス (iOSDevice)
  - iOS シミュレータ (iOSSimulator)
- ビルド構成
  - Debug
  - Release



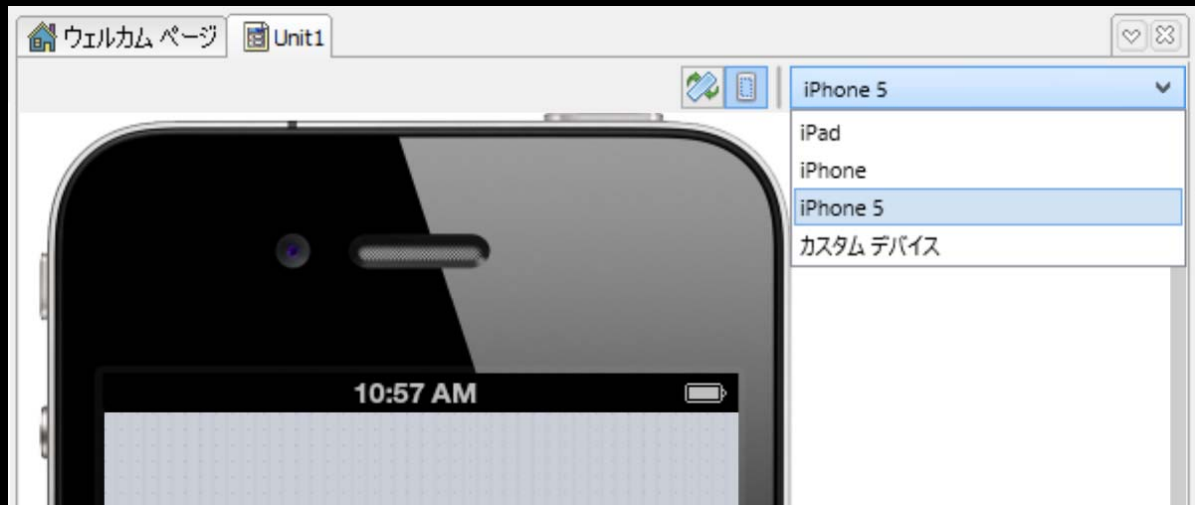


画面の作り方(メインフォームとサブフォーム)

TForm, Show/ShowModal, 入力

フォーム(TForm)の基礎

- 常にiOSの全領域(ステータスバーを除く)を覆う
  - サイズ(Width x Height)は3種類
- フォーム自体にスクロール機能は無い
  - ScrollBox系のコンポーネントを利用する
- Fontプロパティは無い
  - 各コンポーネントはデフォルトで、iOSのシステムフォントを利用



Show/ShowModal

- メインフォームとは、最初に生成されるTForm
  - iPhoneとiPadで別個のフォームを用意するか否か?
- それ以外のフォーム(TForm)を表示するには?
  - subform1.Show(); → モードレス
  - subform1.ShowModal(); → モーダル
  - やはり、iOSの画面全体を覆ってしまう
    - ダイアログボックス的な表示は出来ない
    - TFormのTransparencyプロパティは使えない!?
    - TPanelやフレーム(TFrame)を使う!!

InputQuery/ShowMesage

- InputQuery → ユーザーに入力を求める
- ShowMessage → メッセージを表示する

```
procedure TForm1.Button1Click(Sender: TObject);
var
  result: string;
begin
  if InputQuery('タイトル', 'メッセージ', result) then
  begin
    ShowMessage(result);
  end;
end;
```

- iOS標準のダイアログボックスを使用

デバッグの基礎(ログの出力と確認)

iOSシミュレータ, iOS実機

ログの出力

- iOS APIを使用すると…

```
uses iOSapi.Foundation;

procedure TForm1.Button1Click(Sender: TObject);
var
  i: Integer;
  f: Single;
  e: Extended; // Double
begin
  i := Random(100);
  f := 123.456;
  e := Random;
  NSLog(TNSString.OCClass.stringWithString(NSSTR('%@:%d:%f:%f')),
        TNSString.OCClass.stringWithString(NSSTR('これはログメッセージ!')),
        i, f, e);
end;
```

ログの出力

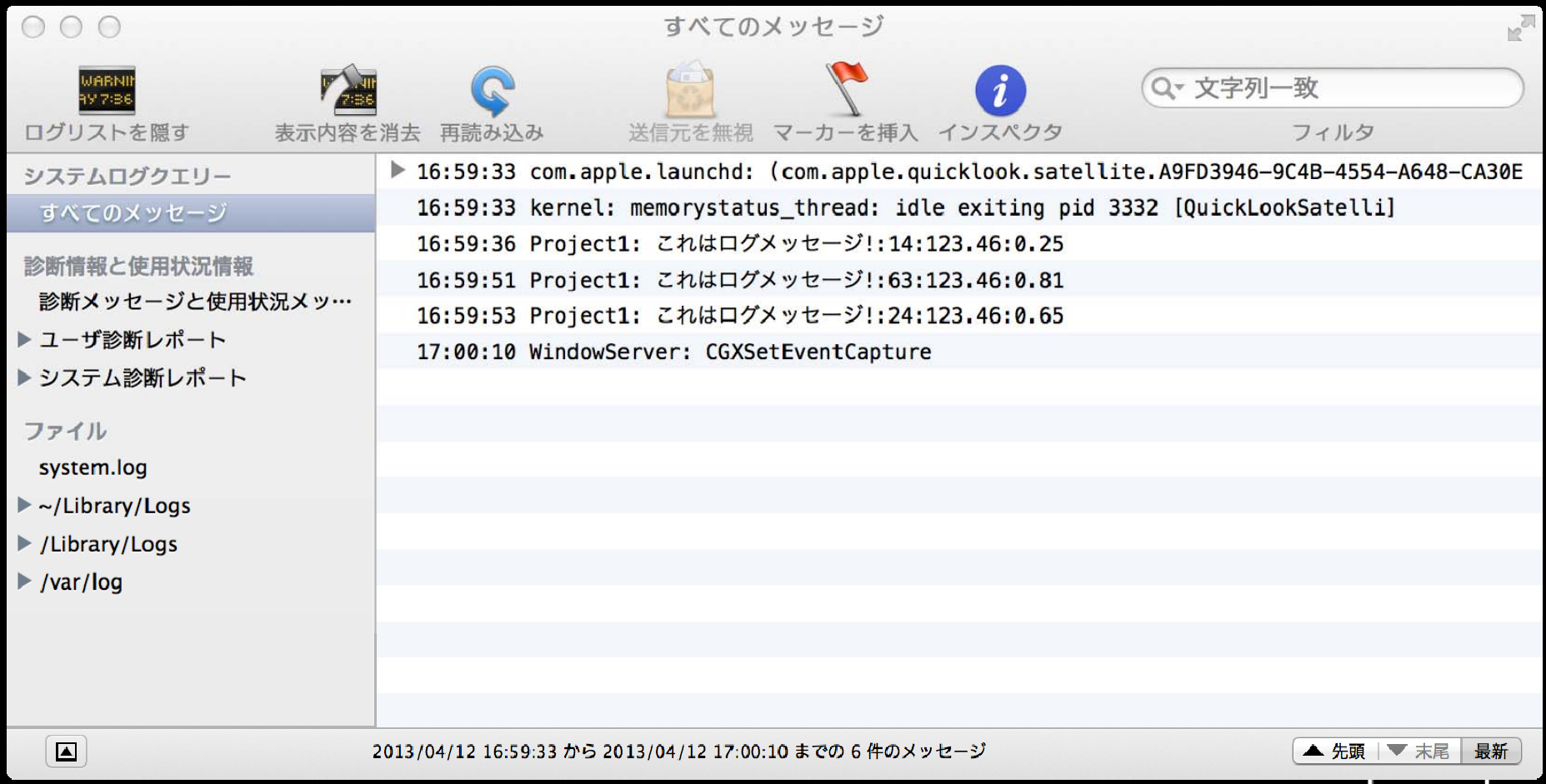
- Delphiのライブラリを使用しましょう!!

```
uses FMX.Platform;

procedure TForm1.Button1Click(Sender: TObject);
var
  i: Integer;
  f: Single;
  e: Extended; // Double
  log: IFMXLoggingService;
begin
  i := Random(100);
  f := 123.456;
  e := Random;
  log := TPlatformServices.Current.GetPlatformService(IFMXLoggingService)
                                                    as IFMXLoggingService;
  log.Log('%s:%d:%f:%f', ['これはログメッセージ!', i, f, e]);
end;
```

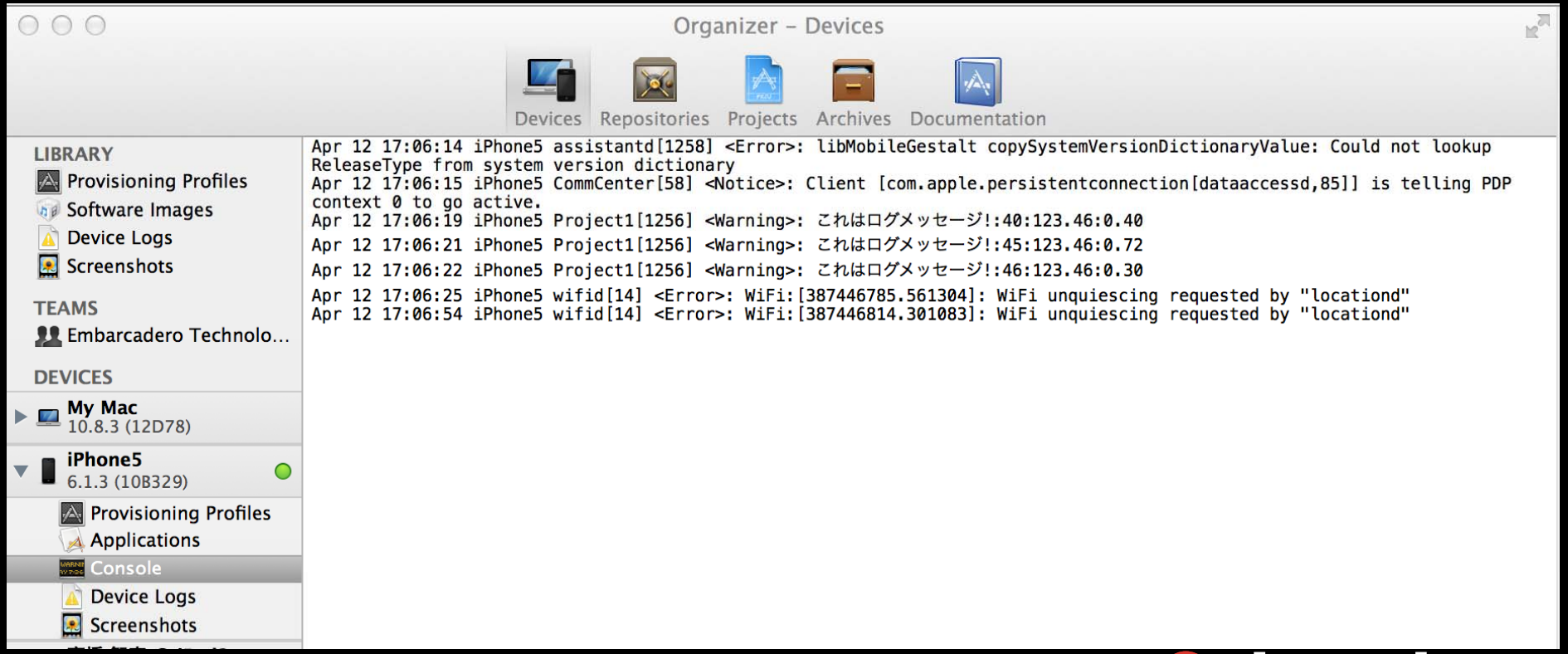
iOSシミュレータ

- アプリケーション – ユーティリティ – コンソール.app



iOS実機

- Xcode – Window – Organizer – DEVICES – 実機 – Console



デバイスの回転に対応するには?

縦・横, イベント, プロジェクト設定

デバイスの向き(縦・横)とイベント

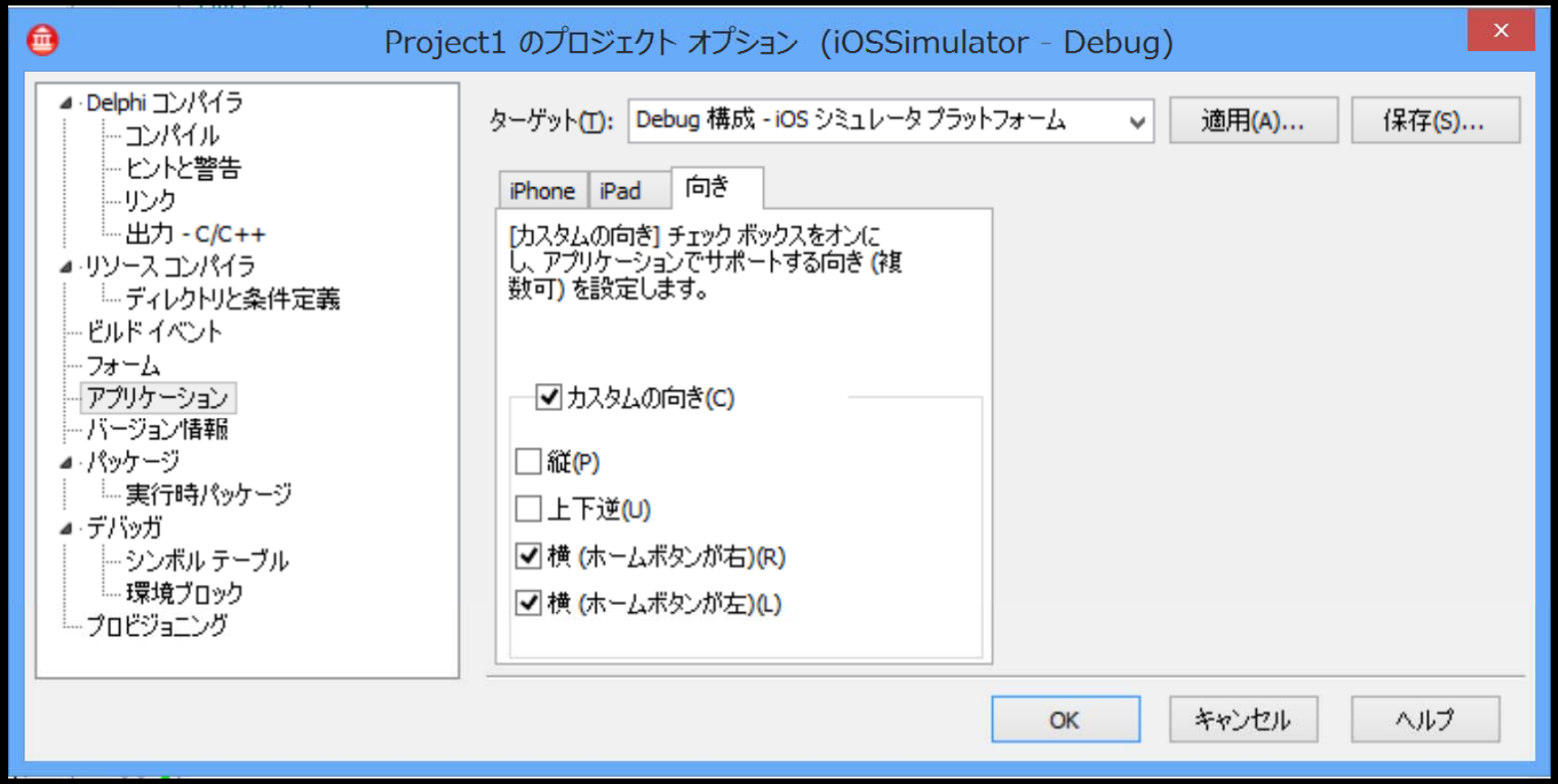
- 表示される前にTForm.OnResizeイベントが発生
  - ただし、Screen.ActiveFormのみ
  - TForm.Width/Height が縦・横に応じて変化
  - Screen.Size.Width/Height は縦向きで固定
- 向きは4種類

```
uses FMX.Platform, System.TypInfo;

procedure TForm1.FormResize(Sender: TObject);
Var
  screen: IFMXScreenService;
  ori: TScreenOrientation;
begin
  screen := TPlatformServices.Current.GetPlatformService(IFMXScreenService) as
IFMXScreenService;
  ori := screen.GetScreenOrientation;
  log.Log('%s', [GetEnumName(TypeInfo(TScreenOrientation), Integer(ori))]);
end;
```

デバイスの向きを固定するには?

- [プロジェクトオプション] – [アプリケーション] – [向き]
  - カスタムの向きを ON にして、4種類から選択



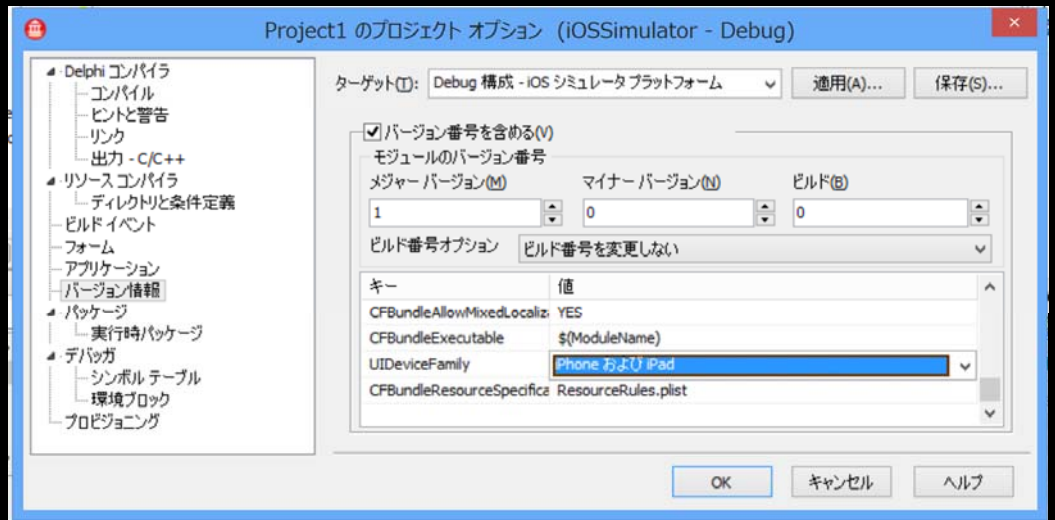
iPhone, iPad への対応方法

ユニバーサルアプリ, メインフォーム

embarcadero Developer Camp

ユニバーサルアプリ

- 単体で、iPhone(iPod touch)とiPadの両方に対応しているアプリのこと
  - iPhone(iPod touch)のみ対応のアプリも可
    - iPadにインストールすると、等倍または2倍で表示される
  - iPadのみ対応のアプリも可
- [プロジェクトオプション] – [バージョン情報]で指定
  - UIDeviceFamily
    - iPhone & iPad



デバイス情報を取得

- モデル名とバージョン番号 – Delphi RTL版

```
uses FMX.Platform;

procedure TForm1.Button1Click(Sender: TObject);
var
  device: IFMXDeviceService;
  os: TOSVersion;
begin
  device := TPlatformServices.Current.GetPlatformService(IFMXDeviceService)
                                                    as IFMXDeviceService;
  ShowMessage(device.GetModel +':’+ IntToStr(os.Major)+'.'+IntToStr(os.Minor));
end;
```

- バージョン番号は2つの整数値(メジャー&マイナー)
  - 例: iPhone:6.1

デバイス情報を取得(続き)

- モデル名とバージョン番号 – iOS API版

```
uses iOSapi.UIKit;

procedure TForm1.Button1Click(Sender: TObject);
var
  device: UIDevice;
  model: string;
  version: string;
begin
  device := TUIDevice.Wrap(TUIDevice.OCClass.currentDevice);
  model := UTF8ToString(device.model.UTF8String);
  version := UTF8ToString(device.systemVersion.UTF8String);
  ShowMessage(model + ':' + version);
end;
```

- バージョン番号は文字列
  - 例: iPad:5.1.1

メインフォームを切り替える

- メインフォームは Project名.dpr のコードで決定される
  - FormFamilyでFireMonkeyに自動選択させる方法もあり

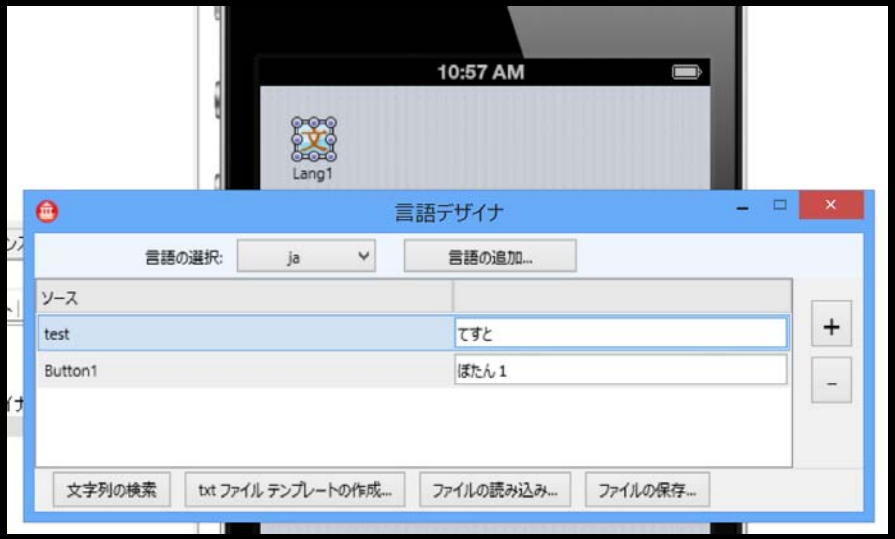
```
program Project1;
uses
  System.StartUpCopy,
  FMX.Forms, FMX.Platform,
  Unit1 in 'Unit1.pas' {Form1},
  Unit2 in 'Unit2.pas' {Form2};
{$R *.res}
var
  device: IFMXDeviceService;
begin
  Application.Initialize;
  device := TPlatformServices.Current.GetPlatformService(IFMXDeviceService)
                                                    as IFMXDeviceService;
  if Pos('iPad', device.GetModel) > 0 then
    Application.CreateForm(TForm2, Form2)   // iPad
  else
    Application.CreateForm(TForm1, Form1);  // iPhone,iPod touch
  Application.Run;
end.
```

アプリのローカライズ手順

TLang, ロケール

TLangコンポーネント

- 各言語向けの翻訳リストを管理
- GUIの文字列プロパティなどを動的に自動置換




```
procedure TForm1.Button1Click(Sender: TObject);
var
  msg: string;
begin
  msg := 'test';
  ShowMessage(msg); // てすと(ja)
end;
```

ロケールの取得

- iOSのロケール一覧(ja, en など)

```
uses iOSapi.Foundation, iOSapi.UIKit;

var
  la: NSArray;
  i: Integer;
  lname: NSString;
begin
  la := TNSLocale.OCClass.preferredLanguages;
  for i := 0 to la.count-1 do
  begin
    lname := TNSString.Wrap(la.objectAtIndex(i));
    ListBox1.Items.Add(UTF8ToString(lname.UTF8String));
  end;
end;
```

- 選択されている言語(ja, en など)

```
uses FMX.Platform;

var
  locale: IFMXLocaleService;
begin
  locale := TPlatformServices.Current.GetPlatformService(IFMXLocaleService) as IFMXLocaleService;
  ShowMessage(locale.GetCurrentLangID);
end;
```

Delphi言語の変更点 (モバイル向け)

文字列, ARC

文字列

- 型
  - UTF-16のstring(UnicodeString)のみ
- ポインタ
  - PChar(PWideChar)のみ
- 文字へのアクセス
  - インデックスは 0 から始まる

```
var
  device: string;
begin
  device := 'My iPhone Simulator';
  ShowMessage(device[4]);                          // P(iOS)
  ShowMessage(device[Pos('iPhone', device)]);      // P(iOS)
  ShowMessage(device[3]);                          // i(iOS)
  ShowMessage(device[device.IndexOf('iPhone')]); // i(iOS)
end;
```

ARC(Automatic Reference Counting)

- NSAutoRelasePoolのGCでもなく、Objective-CのARCでもない、Delphi独自のARC
  - Delphi言語のオブジェクトの解放は不要

```
type
  TMyClass = class
  public
    constructor Create;
    destructor Destroy; override;
  end;
…
…
procedure TForm1.Button1Click(Sender: TObject);
var
  obj: TMyClass;
begin
  obj := TMyClass.Create;
  // obj.Free;
  // FreeAndNil(obj);
  // obj := nil;
end;
```

ヘルプ & 資料

- 多くのサンプルプロジェクト
  - C:¥Users¥Public¥Documents¥RAD Studio¥11.0¥Samples¥FireMonkeyMobile
- 日本語のヘルプ&チュートリアル
  - 製品付属のヘルプ & オンラインのdocwiki
- ホワイトペーパー
  - 「The Delphi Language for Mobile Development」
    - マルコ・カントゥ著

- 参考書籍
  - 「DelphiでかんたんiOSアプリプログラミング」
    - 著者: 細川 淳(ほそかわ・じゅん)
    - 出版: カットシステム http://www.cutt.co.jp/book/978-4-87783-310-7.html

Q&A

Thank you!

embarcadero®
Developer Camp