30TH EMBARCADERO DEVELOPER CAMP

第30回 エンバカデロ・デベロッパーキャンプ・ツアー

【G1】ケーススタディセッション

「C++Builder を用いたマルチ OS 対応一般コンシューマ向けスマートフォンアプリ開発事例」

カシオ計算機 (株) コンシューマ事業部 企画部
園城 浩行

embarcadero

アジェンダ

- 会社紹介 & 自己紹介
- 開発の概要
  - 「撮ってキャラスタジオ」アプリのご紹介
  - C++Builder を導入した経緯について
- 「撮ってキャラスタジオ」のソフトウェア論理構成と、それぞれの開発事例
- Android/iOS それぞれで苦労したこと
- まとめ

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

「C++Builder を用いたマルチ OS 対応
一般コンシューマ向けスマートフォンアプリ開発事例」

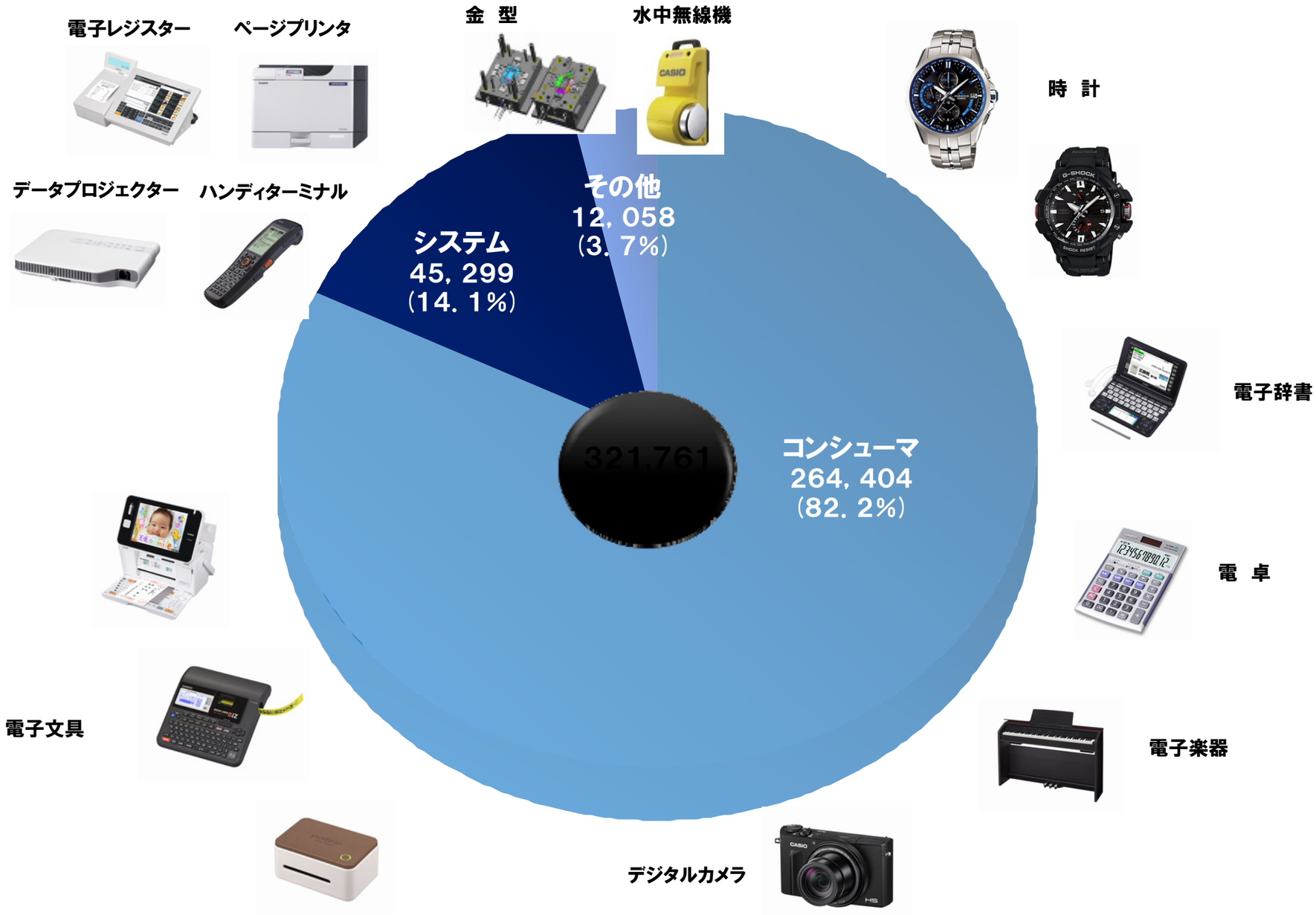
会社紹介 & 自己紹介

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

会社紹介

- 社名　カシオ計算機株式会社（Casio Computer Co.,Ltd.）
- 創立　1957年6月
- 資本金　485億9，200万円
- 連結売上高　3，217億6，100万円（2013年度）
- 従業員数　10，992名（2014/3/31連結）
- 社長　樫尾　和雄

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。



本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

embarcadero

自己紹介

- 名前: 園城 (おんじょう) 浩行
- 所属: カシオ計算機 (株)
- 趣味: 野球観戦 (関東生まれ関東育ちのトラキチ)
- 仕事: ソフトウェアエンジニア
  - 2001年入社
  - 入社-現在: サーバーサイドのエンジニア
    - 開発言語・フレームワーク (最近): JRuby on Rails
    - ミドルウェア: Apache, Tomcat
    - データベース: MySQL
  - **昨年4月に新組織発足！⇒ スマフォアプリも担当することに (しかも、Android/iOS 両方。。。)**

スマフォアプリ開発未経験者しかいない

今日のプレゼン: スマフォアプリ開発ド素人チームによるドタバタ開発事例 (^_^;;;;

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

「C++Builder を用いたマルチ OS 対応
一般コンシューマ向けスマートフォンアプリ開発事例」

「撮ってキャラスタジオ」アプリのご紹介と、C++Builder を導入した経緯について

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

人の写真から”いい感じ” のスタンプ画像を生成

写真を撮影します

Step1　画像の読み込み

カメラで撮影 or 写真の選択

顔の特徴
抽出

雰囲気の似たキャラクターを自動生成

Step2　テーマの選択

テーマを選択するとキャラ画像を作成
（髪型とメガネが変更できます）

ポーズと
セリフを
合成

メッセンジャのスタンプに

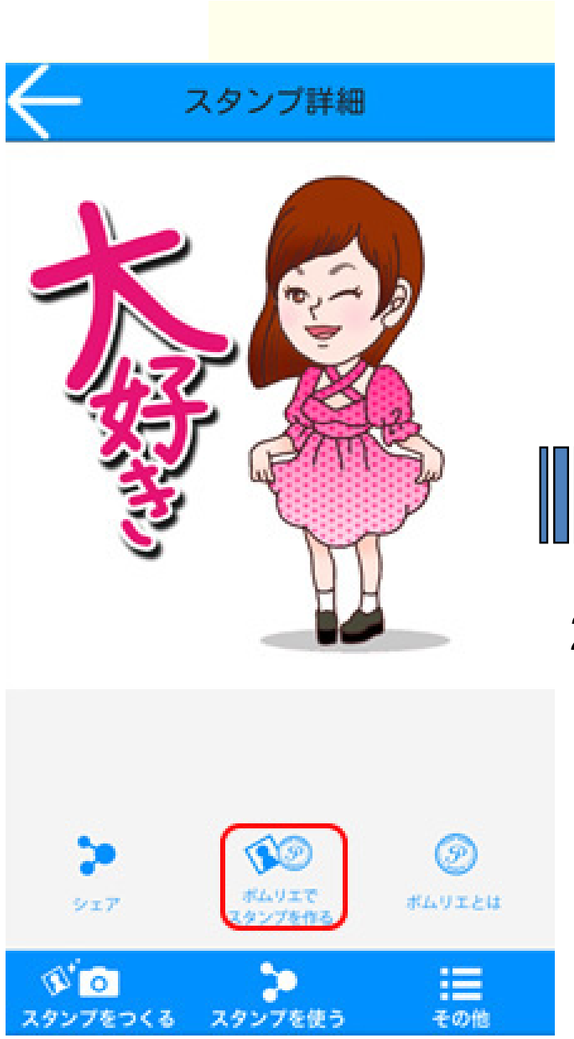
Step3　スタンプ画像の作成

21 個のスタンプ画像を自動作成します

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

embarcadero

弊社製スタンプメーカー「ポムリエ」を繋げてリアルスタンプも



2値化

WiFi でポムリエに
送信するとリアルな
スタンプも作れます

2013年11月発売

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

「撮ってキャラスタジオ」開発の概要

- 開発スタート: 2014年11月
- 開発人員: (スマフォアプリ開発素人) 1.7人
  ※ UI のパーツデザインはデザイン会社様に外注
- 開発ツール: C++Builder
- 最初のリリース
  - Android アプリ: 2015年03月26日
  - iOS アプリ: 2015年04月06日

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

開発に使った C++Builder のバージョン

	開発スタート	初回リリース	イマココ
Android	XE7	XE7	XE8
iOS	XE7	XE8 Beta 10	XE8

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

C++Builder 導入を決めた経緯-1

- ワンソースマルチ OS 開発のアプローチを取りたかった
  - コンシューマ向けアプリ ⇒ 開発スピードが大事
  - しかし
    - 開発人数少ない
    - スマフォアプリ開発以外の仕事もある
- HTML5 ベースのツールも話題にあがったけれど
  - サクサク動くか不安
  - 完全にワンソースにするのは難しい
  - メンテナンス性に不安

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

C++Builder 導入を決めた経緯-2

- 上司が C++Builder の情報を知っていた
  - 完全にデバイスネイティブになるみたいだ
    - HTML5 ベースよりはサクサク動かせそう
    - 全レイヤーをワンソースにできそう
  - C++ なら経験者がいるぞ
    - ただし、Delphi も経験者は誰もおらず

じゃあ、C++Builder でチャレンジしてみよう

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

C++Builder に対する印象の変化

- 最初の印象
  - Android/iOS それぞれの開発環境に比べると、できることは限られているが、両 OS 対応アプリが簡単に作れる
- 使ってみて
  - Android/iOS それぞれでできることは (大抵のことは) なんでもできる。ただし、ちゃんとしたアプリを作るためにはがっつり勉強する必要がある

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

「C++Builder を用いたマルチ OS 対応
一般コンシューマ向けスマートフォンアプリ開発事例」

「撮ってキャラスタジオ」のソフトウェア論理構成と、それぞれの開発事例

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

「撮ってキャラスタジオ」ソフトウェア論理構造

(1) UI

アプリ全体制御・各機能の呼び出し

(2) キャラ変換エンジン

- 顔パーツ検出
- 似顔絵パーツ決定
- スタンプ合成

(3) ポムリエ連動モジュール

- スタンプ画像2値化
- ポムリエとの WiFi 通信

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(1) UI 部分の開発について

(1) UI

アプリ全体制御・各機能の呼び出し

(2) キャラ変換エンジン

- 顔パーツ検出
- 似顔絵パーツ決定
- スタンプ合成

(3) ポムリエ連動モジュール

- スタンプ画像2値化
- ポムリエとの WiFi 通信

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(1) UI 部分の開発について: スタイルデザイン

- 標準のコンポーネントだけで構成されるアプリはさほど問題なく作れそう
- コンシューマ向けアプリの場合、凝った UI を求められる
  ⇒ スタイルデザインがさっぱりわからない
- エークリッパー・インク 羽生さんに講義をお願い
  – 3日かかるというものを1日で教育していただくことに

⇒ スタイルデザインは試行錯誤するしかない
  – それにしても IDE がよく落ちる。。。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

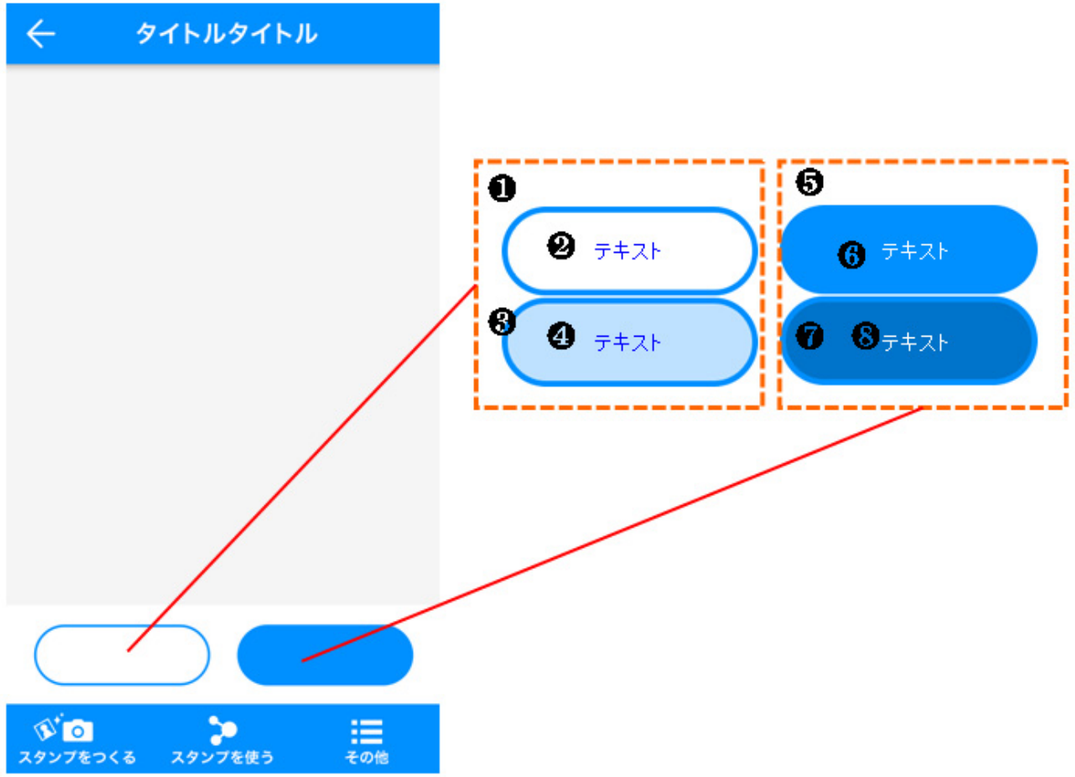
(1) UI 部分の開発について:スタイルデザイン

- 羽生さんからいただいたアドバイス: ワークフローにスタイルデザインの工程を入れること

要件
↓
仕様
↓
デザイン(モック)
↓
スタイル設計→スタイルファイル
※エフェクトやアニメの組み合わせ
↓
UI レイアウトの作成
↓
アプリ実装

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(1) UI 部分の開発について: UI デザイナさんとのやりとり



No.	種別	width	height	Font	X座標	Y座標	配置	色	部品	備考
❶	BG	460	344	–	58px	1540px	–	#333333	back_btn.png	–
❷	ボタンテキスト	–	–	40pt	–	–	❶の範囲で、上下センタリング	#0080ff	–	–
❸	BG	460	344	–	562px	1540px	–	#333333	next_btn.png	–
❹	ボタンテキスト	–	–	40pt	–	–	❶の範囲で、上下センタリング	#ffffff	–	–

従来の Android/iOS の開発環境とは
UI のはめ込み方が異なるので、
C++Builder に合わせたパーツの組込み指示書の
作成をデザイナさんに依頼

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(2) キャラ変換エンジンの開発について

(1) UI

アプリ全体制御・各機能の呼び出し

(2) キャラ変換エンジン

- 顔パーツ検出
- 似顔絵パーツ決定
- スタンプ合成

(3) ポムリエ連動モジュール

- スタンプ画像2値化
- ポムリエとの WiFi 通信

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(2) キャラ変換エンジンの開発について: 概要

- 弊社研究部門 (別部門)と協力会社さんの共同研究でアルゴリズムを開発
- 研究成果のソースコードをもらい、アプリに組込み実装
  - プログラムは Visual Studio のプロジェクトで開発されていた
    - そのままでは C++Builder ではビルドすらできない。。。
  - 粛々と C++Builder に移植

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(2) キャラ変換エンジンの開発について: 手順

- C++Builder で1つ1つコンパイルして環境依存のコードを修正 ⇒ 動作確認
  - とても泥臭い作業でした
    - とはいえ、C++ 同士なので普通にソースを読み込めば特別なことをしなくても作業開始できます
- <u>効率のよい方法をご存知の方、教えてください＞＜</u>

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(3) ポムリエ連動モジュールの開発について

(1) UI

アプリ全体制御・各機能の呼び出し

(2) キャラ変換エンジン

- 顔パーツ検出
- 似顔絵パーツ決定
- スタンプ合成

(3) ポムリエ連動モジュール

- スタンプ画像2値化
- ポムリエとの WiFi 通信

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(3) ポムリエ連動モジュールの開発について: 前提

- 別部門が開発したポムリエと連動するアプリがあった
  - Android/iOS それぞれの開発環境で構築

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(3) ポムリエ連動モジュールの開発について: 最初の設計戦略

- 折角動いているプログラムがある
- ちゃんとライブラリ化もされている

じゃあ、C++Builder から Android 用/iOS 用
それぞれのライブラリをリンクして使えば効率よさそう！

などと思っていたらドツボにハマりました orz

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(3) ポムリエ連動モジュールの開発について: Android の場合

- java2op で jar ファイル ⇒ Delphi への I/F を作成
  - jdk1.8 では動かず、jdk1.7 へ path を変えたり。。。
- java2op で作成された *.pas を C++Builder でビルド ⇒ *.hpp ファイルを作成
  - これが I/F になるんだな、どれどれ (ソースを見てみる)
  - Delphi へのラッパとおぼしき記述がずらり
- Delphi の知識がないので、どう使っていいかわからず
  - 1W くらい試行錯誤してみたものの、やっぱりわからず

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

RAD Studio Android アプリケーションでのカスタム Java ライブラリ セットの使用

RAD Studio Android アプリケーションでの Java ライブラリの使用 への移動

RAD Studio では、RAD Studio アプリケーションで独自のまたはサードパーティ製の Java ライブラリを使用することができます。

RAD Studio Android アプリケーションでカスタム ライブラリ セットを使用するには：

1. お使いの Java ライブラリが RAD Studio の Java ライブラリの要件を満たしていることを確認します。
2. Delphi または C++ アプリケーションで Java ライブラリ API へのアクセスに使用できるネイティブ ブリッジ ファイルを作成します。
3. アプリケーションに Java ライブラリを追加します。

目次 [非表示]

1 RAD Studio の Java ライブラリの要件
2 Java ライブラリのネイティブ ブリッジ ファイルの作成
3 Java ライブラリのアプリケーションへの追加
4 関連項目

RAD Studio の Java ライブラリの要件

RAD Studio では、Android API レベル 10(Android 2.3.3)以降のみサポートしています。お使いの Java ライブラリで Android API を使用している場合は、サポートされている Android API を使って、それらのライブラリをコンパイルする必要があります。

お使いの Java ライブラリが他の Java ライブラリに依存している場合は、自分の Java ライブラリとその依存先ライブラリの両方の JAR ファイルが必要です。後で有効な classes.dex ファイルを生成するために、これらすべての Java ライブラリの JAR ファイルが必要なのです。

各ライブラリにどの JAR ファイルが属しているかについては、「classes.dex ファイルの生成」セクションの表を参照してください。

Java ライブラリのネイティブ ブリッジ ファイルの作成

Java ライブラリができたら、ネイティブ ブリッジ ファイルを 1 つ以上作成する必要があります。これらは、お使いの Java ライブラリの API にアクセスするためにネイティブ RAD Studio アプリケーションに組み込むことができる Delphi ファイルまたは C++ ファイルです。

Java2OP コマンドライン ツールを使って、Delphi のネイティブ ブリッジ ファイルを作成できます。その後は、Delphi ネイティブ ブリッジ ファイルを C++ に手動で変換できます。

RAD Studio インストール フォルダ(C:¥Program Files (x86)¥Embarcadero¥Studio¥15.0)内の次のパスにある既存のブリッジ ファイル(Androidapi.JNI で始まるファイル)を調べると参考になるでしょう。

- source¥rtl¥android(Delphi の場合)
- include¥android¥rtl(C++ の場合)

Java ライブラリのアプリケーションへの追加

Android 向けアプリケーションに Java ライブラリを追加するには、2 つの異なるアプローチがあります。

- プロジェクト マネージャを使用して、Java ライブラリをアプリケーションに追加。これは、もっともシンプルな方法で、必要なのはクリック 2 回のみです。この方法は、大抵の人が利用できるでしょう。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

embarcadero

ページ | 議論 | ソースを表示 | 履歴

表示設定

検索

表示

検索

RAD STUDIO XE7

- XE7 コード例(英語)
- XE7 ライブラリ
- メイン ページ - XE7 トピック Wiki
- XE7 の新機能
- チュートリアル
- FireMonkey アプリケーション プラットフォーム
- マルチデバイス アプリケーション
- 入門
- プロジェクト開発のステップ
- 主要なアプリケーション種別
- FireDAC
- Windows 開発者ガイド
- モデリング ツール
- IDE リファレンスとコマンドライン ユーティリティ
- Delphi リファレンス
- C++ リファレンス
- 主要インデックス
- HTML5 Builder

以前のバージョン

- XE6 のトピック
- XE5 のトピック

Java2OP.exe - Android 向けネイティブ ブリッジ ファイル ジェネレータ

コマンドライン ユーティリティ:インデックス への移動

Java2OP.exe("Java To Object Pascal")は、Java ライブラリ(JAR またはクラス ファイル)から Delphi ネイティブ ブリッジ ファイルを生成するのに使用できるコマンドライン ツールです。生成されたファイルを使用すると、Android 上で Delphi アプリケーションからそれらの Java ライブラリにアクセスできるようになります。「**RAD Studio Android アプリケーションでのカスタム Java ライブラリ セットの使用**」を参照してください。

Java2OP.exe は登録ユーザー向けサイト(http://cc.embarcadero.com/myreg)で入手可能です。

目次 [非表示]

1 使用法
1.1 入力オプション
1.2 出力オプション
2 例
3 トラブルシューティング
3.1 JDK が不明
3.2 EListError エラー
4 関連項目

使用法

このツールを使用するには、オプションを任意に組み合わせてツールを呼び出すだけです。

```
Java2OP.exe [<オプションの任意の組み合わせ>]
```

出力される Delphi ネイティブ ブリッジ ファイルにどのような内容を含めるかを示す入力オプションを少なくとも 1 つ指定する必要があります。

`-classes` パラメータを使って明示的に指定しない限り、出力されるネイティブ ブリッジ ファイルには、Android 向けの組み込み RAD Studio Java ライブラリの一部となっているものは含まれていません。指定したクラスのいずれかが Android 向けの組み込み RAD Studio Java ライブラリの構成要素に依存している場合、結果として生成されるネイティブ ブリッジ ファイルではそれらの要素は再宣言されず、代わりに、それらの要素を既に宣言している RAD Studio ユニットがネイティブ ブリッジ ファイルにインクルードされます。

入力オプション

オプション	説明
`-classes`	エクスポートするクラスまたはパッケージの名前をスペースで区切ったリストです。 `-classes` では、指定の Java ソース(`-jar` オプションまたは `-source` オプション)または Android API の特定のサブセットを明示することができます。
`-jar`	入力となるコンパイル済み Java ライブラリ(`.jar` ファイル)をスペースで区切ったリストです。指定されたファイルのすべての API メンバが解析され、生成されるネイティブ ブリッジ ファイルで宣言されます。
`-source`	Java ソース ファイル(`.java` ファイル)が格納されている入力フォルダをスペースで区切ったリストです。指定されたフォルダに含まれているファイルのすべての API メンバが解析され、生成されるネイティブ ブリッジ ファイルで宣言されます。

出力オプション

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

embarcadero

(3) ポムリエ連動モジュールの開発について: iOS の場合

- (当時) java2op に相当するツールが見当たらず。。。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

(3) ポムリエ連動モジュールの開発について: 結局

- 既存のライブラリをリンクして使うことをあきらめる
- C++Builder でフルスクラッチで開発
  - 2値化エンジンはもともと C++ で作られていたので流用
- 教訓
  - Android/iOS それぞれの開発環境で作られた資産を使いたい、と思ってもそれが本当に最適解か計画段階で検証すべき
    - 想像するよりハードルが高いです

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

「C++Builder を用いたマルチ OS 対応
一般コンシューマ向けスマートフォンアプリ開発事例」

Android/iOS それぞれで苦労したこと

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

Android でアプリが固まる

- 処理中に画面を連打するとアプリが固まる (ANR)
  - メインスレッドで5秒以上かかる処理を実行していた
    - メインスレッドで何をするか、サブスレッドで何をするか、をきちんと再設計した (その他、細かい工夫を色々追加)
- 処理中にアプリを最小化するとアプリが固まる
  - ライフサイクルの監視と適切な処理をしていなかった orz
    - Embarcadero さんにサポートをいただき、ライフサイクルを監視して、適切な処理を行うよう再設計
    - その他、実装を見直し

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

```
_di_IFMXApplicationEventService AppEventService;
if (TPlatformServices::Current->SupportsPlatformService(__uuidof(IFMXApplicationEventService)) &&
          (AppEventService = TPlatformServices::Current->GetPlatformService(__uuidof(IFMXApplicationEventService)))) {
        AppEventService->SetApplicationEventHandler(HandleAppEvent);
 } else {
        // Application Event Service is not supported
 }
```

```
bool __fastcall TSampleForm::HandleAppEvent(TApplicationEvent AAppEvent, System::TObject* AContext)
{
      bool _return = true;
      switch (AAppEvent) {
      case TApplicationEvent::EnteredBackground:
            // バックグラウンドに入ったときにすべき処理が入る
            break;
      case TApplicationEvent::BecameActive:
            // アクティブに戻ったときにすべき処理が入る
            break;
        // その他コードの記述が続く。。。
        }
        return _return;
}
```


本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

iOS 64ビットサポートが必須に！

- 2015年2月1日以降 ⇒ XE7 は iOS 64 bit 未対応
  - 仕方ないので XE8 のベータ版で申請することに
- iOS 側の開発は XE8 ベータに移行
  - ベータ版なのでちょいちょい落ちる
  - 毎週のようにアップデートが
  - 作成される ipa がおかしいことが？？？

⇒ バタバタしながらも なんとか審査に出す。。。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

結果、リジェクト orz

iOS データストレージガイドライン違反

Apps must follow the iOS Data Storage Guidelines or they will be rejected

On launch and content download, your app stores ○○ MB,
which does not comply with the iOS Data Storage Guidelines.

Next Steps

アプリを入れただけで iCloud にデータが
いっぱいバックアップされちゃってるよ

Please verify that only the content that the user creates using your app, e.g., documents, new files, edits, etc. is backed up by iCloud as required by the iOS Data Storage Guidelines.
Also, check that any temporary files used by your app are only stored in the /tmp directory; please remember to remove or delete the files stored in this location when it is determined they are no longer needed.

Data that can be recreated but must persist for proper functioning of your app - or because users expect it to be available for offline use - should be marked with the "do not back up" attribute. For NSURL objects, add the NSURLIsExcludedFromBackupKey attribute to prevent the corresponding file from being backed up. For CFURLRef objects, use the corresponding kCRUFLIsExcludedFromBackupKey attribute.

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

リジェクトの原因

- 本アプリはプログラム内部データを山盛り持っている (パーツ画像、学習データ等)
- それらを StartUp¥Documents に配置していた

[docwiki: iOS アプリケーションの作成]

ファイルの読み込みと配置

iOS デバイス ターゲット プラットフォーム向けにアプリケーションを配置する際、ファイルをアプリケーションと一緒に提供してそれらのファイルを実行時に読み込ませるつもりであれば、配置マネージャを使用して、配置時にファイルを StartUp¥Documents フォルダに格納します。

All configurations - iOS Device platform

Local Path	Local Name	Type	Platforms	Remote Path	Remote Name
..\..\..\..\..\Public\Do...	dbdemos.gdb	File	[Android,iOSDevice,iOSSimulator]	StartUp\Documents\	dbdemos.gdb

以下のコードを使用すると、これらのファイルを実行時に特定できます。

Delphi の場合:

```
TPath.Combine(TPath.GetDocumentsPath, 'filename')
```

C++ の場合:

```
System::Ioutils::TPath::Combine(System::Ioutils::TPath::GetDocumentsPath(), L"filename")
```

ここで、filename は、上記スクリーンショットの 'dbdemos.gdb' のようなファイルの名前です。TPath レコードは System.IOUtils ユニットで宣言されているため、uses 句に System.IOUtils を追加する必要があります。

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

iOS Data Storage Guidelines

これに違反

1. Documents フォルダに置いていいのはユーザー作成のデータのみおくべし
2. ユーザ生成のデータではないファイルや、キャッシュデータなどは置いてはいけないです。
3. 再ダウンロード出来るデータ、再作成できるデータは /Library/Cachesに置くべし
4. テンポラリーファイルは /tmp に置くべし
5. バックアップが不要なファイルは "do not back up" attribute を付けるべし

参考: http://qiita.com/tototti/items/8646405f47cc56a59722

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

結局どうしたか？

- 本アプリはサーバー連動などはしていないので、データが消される可能性のある /tmp や /Library/Caches には置けない
- プログラム内部データをあるフォルダにまとめておいてそのフォルダに“do not back up” 属性を付与して対処

```
_di_NSURL NSU = TNSURL::Wrap(TNSURL::OCClass->fileURLWithPath(StrToNSStr(path)));
_di_NSString key = StrToNSStr("NSURLIsExcludedFromBackupKey");
System::PPointer err;
NSU->setResourceValue(TNSNumber::OCClass->numberWithBool(true), key, err);
```

これでなんとかアクセプトされました！

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

「C++Builder を用いたマルチ OS 対応
一般コンシューマ向けスマートフォンアプリ開発事例」

まとめ

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

開発経験から考えたこと

- 素人がどたばた開発を経てなんとかリリースできました
- C++Builder は、<u>ちゃんと使ってあげると</u>効果絶大！
  - 素人 1.7人で両 OS 対応アプリをリリースできたのは C++Builder のおかげ
  - ただし、銀の弾丸ではない
    - <u>**ちゃんと使うには iOS/Android の知識は必須**</u>
      - Apple/Google 各社が出している開発者向けドキュメントはきちんと読み込むべし (ここが不十分で色々痛い目に)
    - Delphi もちゃんと知っておいたほうが確実に良い
      - が、私もぜんぜんできていない＞＜

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

C++Builder or 従来の開発環境

- C++Builder はワンソースで大抵のことは実現できるが学習コストは高い
- 何を目指すか？
  - 工数の削減？機能の追及？細部へのこだわり？
- C++Builder が向いていそうなケース
  - 短期間で Android/iOS 同時リリースをしたい
  - 開発ラインを2ライン組むだけのリソースがない
  - C++ の中級者以上が複数人存在する

本文書の一部または全部の転載を禁止します。本文書の著作権は、著作者に帰属します。

30TH EMBARCADERO DEVELOPER CAMP

第30回 エンバカデロ・デベロッパーキャンプ・ツアー

Thank you!

embarcadero