【セッションNo. 2】

Delphi/400 最新技術情報

Delphi/400 XE5
ーこんなに簡単！IBM iスマートデバイスネイティブ開発ー

株式会社ミガロ.
RAD事業部　技術支援課
吉原　泰介

【アジェンダ】

1.企業導入が進むスマートデバイス

2.スマートデバイスアプリケーションの種類

3.ネイティブアプリケーションの開発
- 3-1. ネイティブアプリケーションの開発環境
- 3-2. 簡単なネイティブアプリケーションの開発
- 3-3. IBM i に接続するネイティブアプリケーションの開発
- 3-4.ネイティブアプリケーションの配布

4.まとめ

1.企業導入が進むスマートデバイス

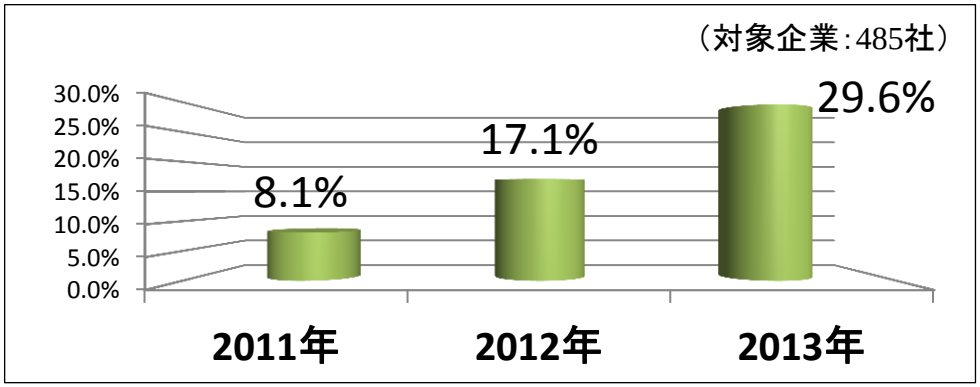
1.企業導入が進むスマートデバイス

- スマートデバイス企業導入率と導入OSの傾向

この1～2年でスマートデバイスの法人への導入が急速に進んできます。
また企業で導入されるスマートデバイスはiOS、Androidが主流になっている傾向です。

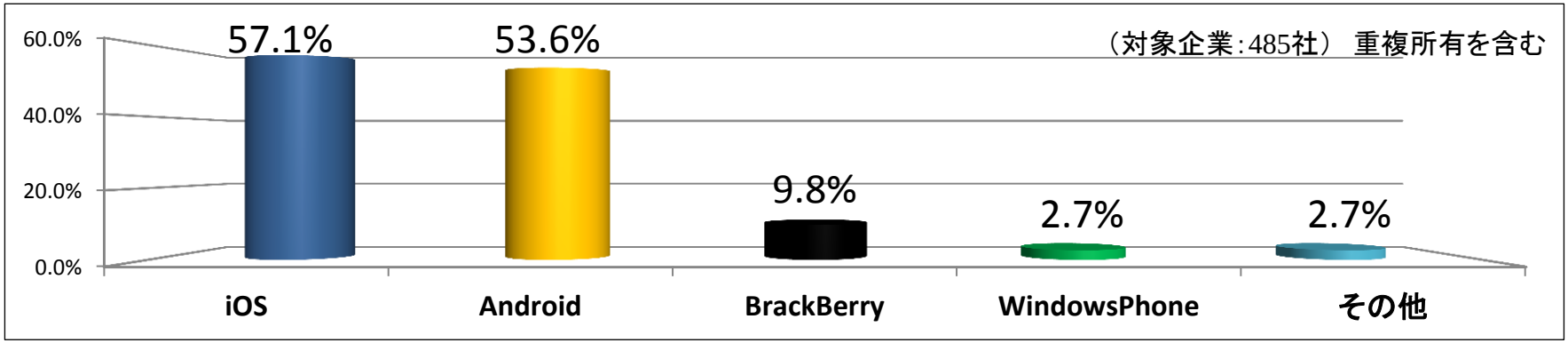
スマートデバイス企業導入率の遷移

2011年に比べると2013年では
導入率が3.5倍以上に増加



企業導入スマートデバイス比率（2013年）

スマートデバイスを導入している企業の大半が iOS 、Android を採用



参考：キーマンズネット業務用スマートフォン導入状況2013

1.企業導入が進むスマートデバイス

- スマートデバイスアプリケーションの業務利用

スマートデバイス導入企業の多くは「業務効率化」を導入目的としています。

→ 今後はPCアプリケーション（Windows ）だけではなく、
スマートデバイスアプリケーションも業務利用が増加

PCアプリケーションとスマートデバイスアプリケーションの特徴の違い

特徴が違う

PCアプリケーションの特徴	スマートデバイスアプリケーションの特徴
細かいキーボード入力ができる	どこでもすぐに使用できる
画面が大きく見やすい	タッチで感覚的に操作できる
ネットワークが安定している	カメラ、GPS、センサー等が利用できる
携帯性が低い	細かいキーボード入力は不向き

2.スマートデバイスアプリケーションの種類

2.スマートデバイスアプリケーションの種類

- スマートデバイスで利用されるアプリケーション
  スマートデバイス（iOS、Android）で利用することができるアプリケーションは、大きく2種類のアプリケーションに分かれます。

ネイティブアプリケーション

Webアプリケーション

2.スマートデバイスアプリケーションの種類

ネイティブアプリケーション

ネイティブアプリケーションは、スマートデバイス端末上で動作してデバイス機能と連携ができるアプリケーションです。
App Store や GooglePlay といったストアや、社内向けに公開したWebサーバからインストールして利用することができます。

特徴

- デバイス機能（カメラ、GPS等）が利用できる
- レスポンスが良い
- オフラインでも利用できる

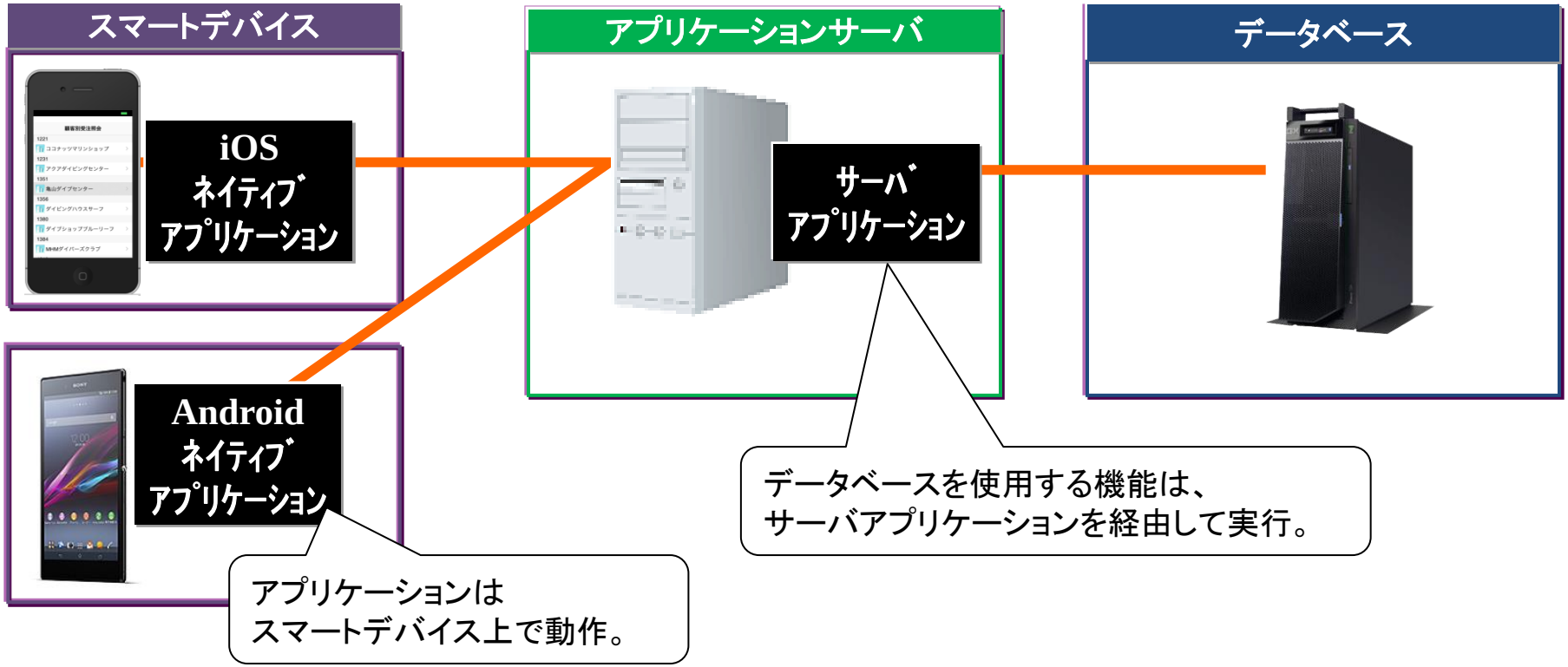
開発言語例

- iOS：　Objective-C
- Android：　Java等

2.スマートデバイスアプリケーションの種類

ネイティブアプリケーション

ネイティブアプリケーションの環境構成



2.スマートデバイスアプリケーションの種類

Webアプリケーション

Webアプリケーションは、Webサーバ上で動作するプログラムを
PC同様にブラウザから利用できるアプリケーションです。
スマートデバイス端末にアプリケーションはインストールされないため、
ブラウザのブックマーク等を使って利用することができます。

特徴

・ブラウザで実行するため、プラットフォームを問わず
　汎用的に開発・利用できる。
・インストールが不要なため、利用が容易。

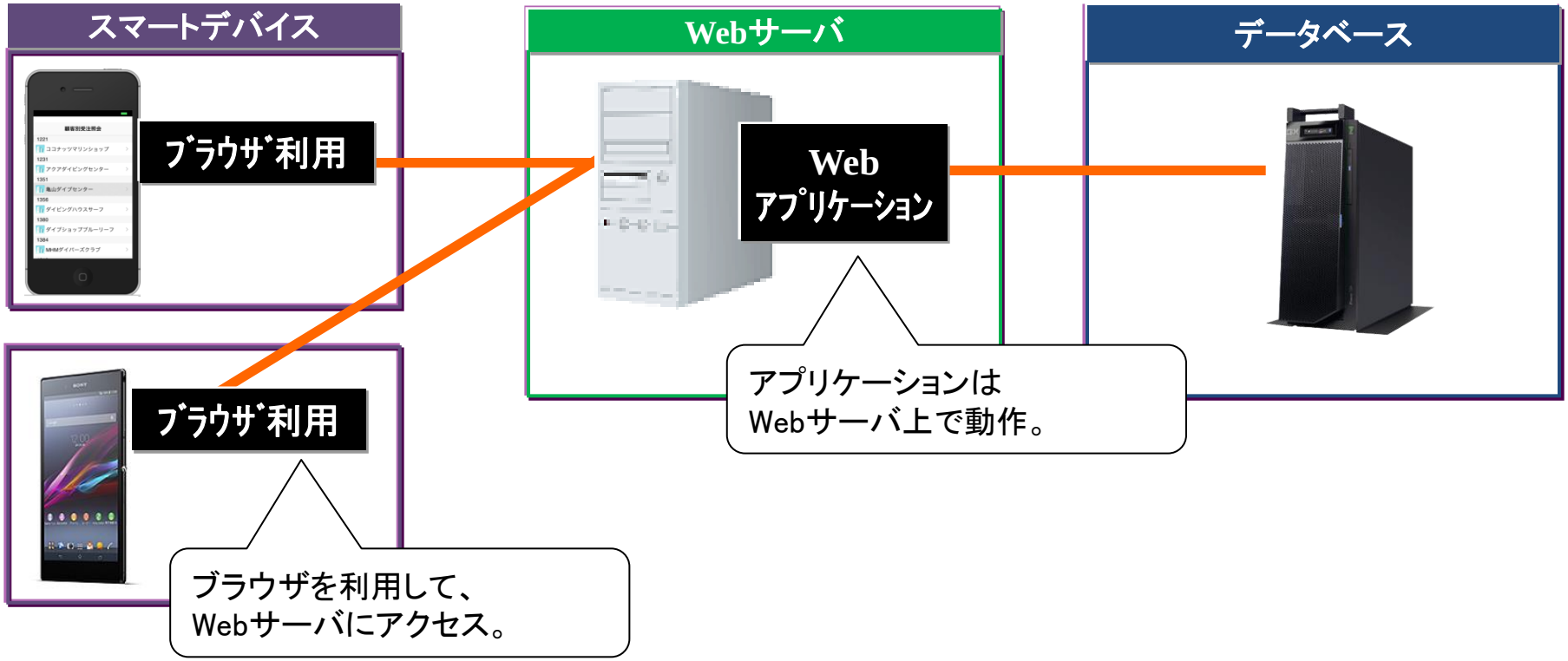
開発言語例

・HTML5、javaScript、PHP、java等

2.スマートデバイスアプリケーションの種類

Webアプリケーション

Webアプリケーションの環境構成



2.スマートデバイスアプリケーションの種類

- ネイティブ / Webアプリケーションの特徴

	ネイティブ	Web
開発言語	iOS:Objective-C Android:Java	HTML javascript等
開発生産性	△	○
デバイス機能	◎	△
パフォーマンス	◎	○
オフライン動作	◎	×
配布	△	◎

アプリケーションによって
多くの開発言語習得が必要

言語によって開発環境が
煩雑になるため、
生産性が低くなる

2.スマートデバイスアプリケーションの種類

- ネイティブ / Webアプリケーションの特徴（Delphi）

	ネイティブ	Web
開発言語	Delphi	Delphi
開発生産性	◎	◎
デバイス機能	◎	△
パフォーマンス	◎	○
オフライン動作	◎	×
配布	△	◎

開発言語を
Delphiで統一できる

Delphiの開発機能は
生産性が高い

2.スマートデバイスアプリケーションの種類

- Delphi/400 XE5 ネイティブアプリケーションの強み

DelphiスキルでiOS / Android ネイティブ開発ができる

開発言語はDelphiだけで iOS / Android のネイティブ開発ができます。
またコンパイルの設定切り替えだけで、1つのプログラムから
iOS / Android 両方のスマートデバイスに対応できます。

従来と同じ手法でネイティブ開発ができる

コンポーネントで画面設計して、イベントでプログラムコーディングする
従来の開発手法でネイティブアプリケーションが開発できます。

デバイス連携機能を簡単に開発することができる

スマートデバイス連携機能（カメラやGPS等）を専用コンポーネントで、
簡単に開発することができます。

2.スマートデバイスアプリケーションの種類

- ネイティブアプリケーションのデバイス機能連携
  ネイティブアプリケーションではカメラ連携、バーコード連携、GPS連携
  といったデバイス連携機能をアプリケーションへ簡単に実装できます。

カメラ撮影

バーコード読取

GPS位置情報取得

2.スマートデバイスアプリケーションの種類

- ネイティブアプリケーションのデバイス機能連携例1
  カメラ機能を連携したネイティブアプリケーション

2.スマートデバイスアプリケーションの種類

- ネイティブアプリケーションのデバイス機能連携例2
  バーコード、QRコード読取りを活用したネイティブアプリケーション

2.スマートデバイスアプリケーションの種類

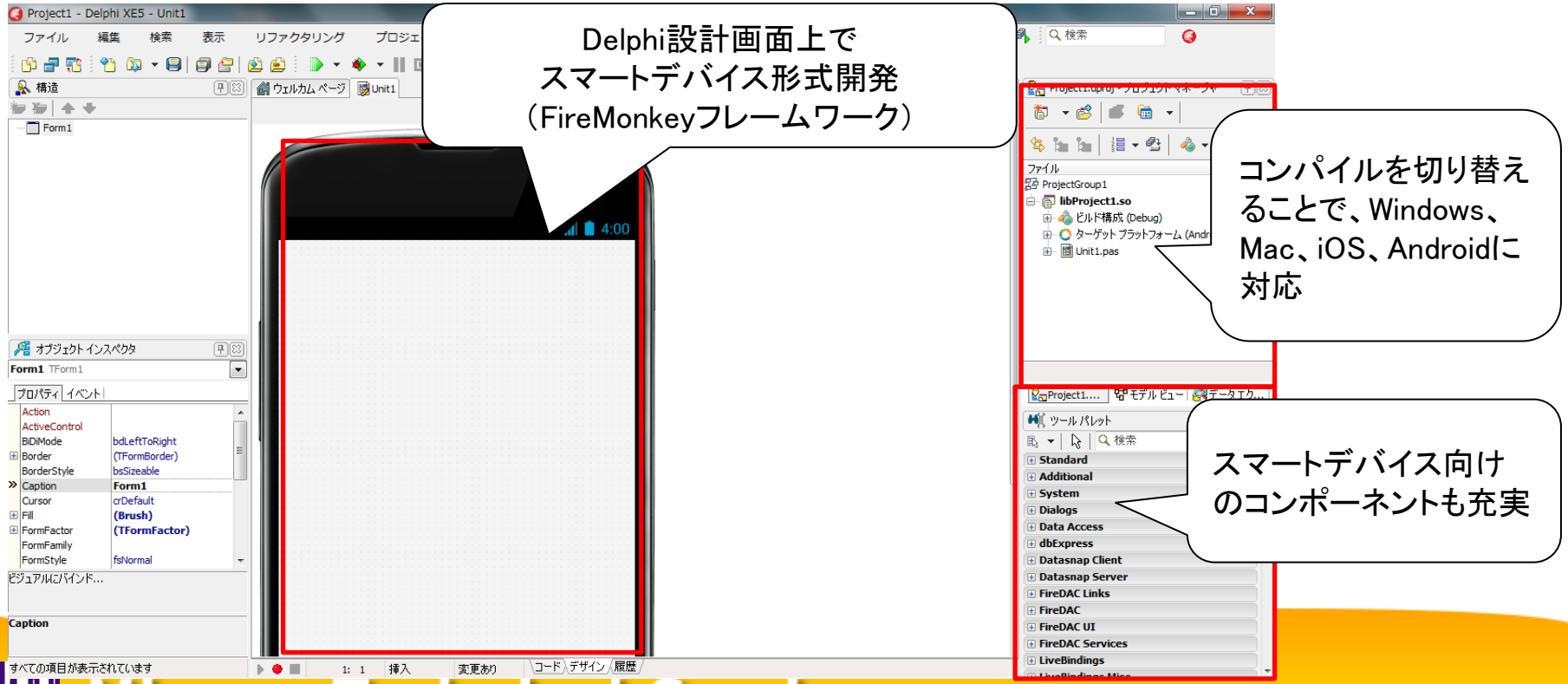
- ネイティブアプリケーションのデバイス機能連携例3
  GPSを使って地図連携を活用したネイティブアプリケーション

3.ネイティブアプリケーションの開発

3-1. ネイティブアプリケーションの開発環境

- Delphi/400 XE5 のネイティブアプリケーション開発機能

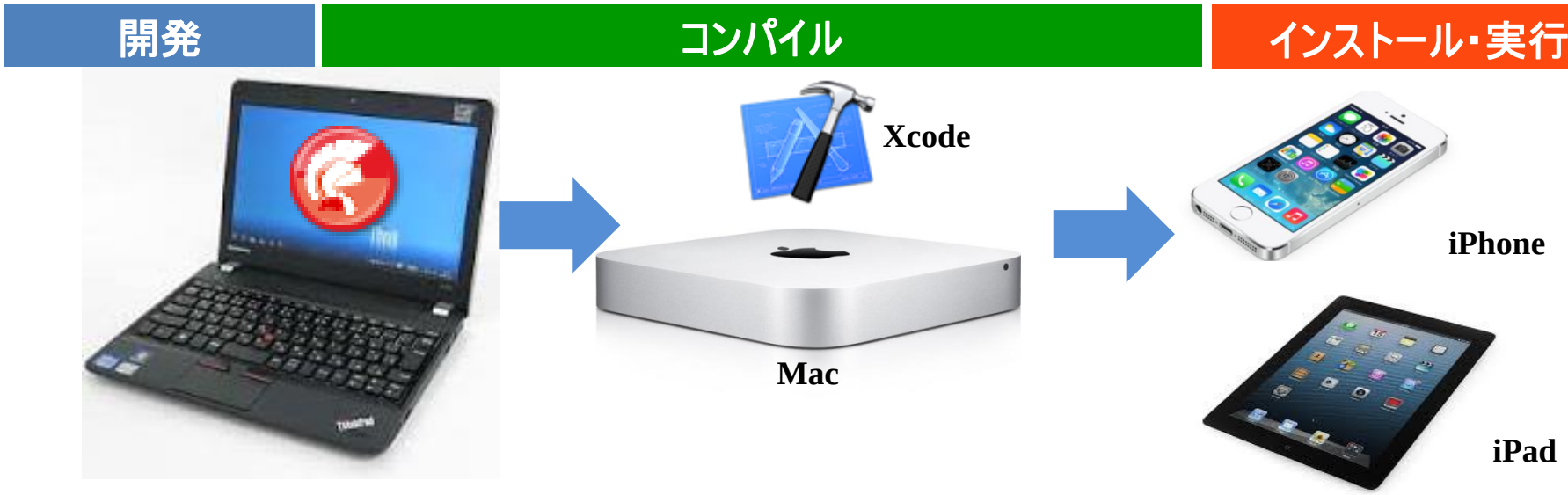
  Delphi/400 XE5 では、Windowsアプリケーションの開発機能に加え、
  スマートデバイスのネイティブアプリケーション開発機能が追加されました。
  FireMonkeyフレームワークを利用することで、Delphiスキルを使って
  iOS、Androidアプリケーションが開発できます。



3-1. ネイティブアプリケーションの開発環境

- iOSネイティブアプリケーション開発に必要な環境
  - Windows端末（Delphi/400 XE5）
  - Mac端末（OSX 10.7～10.9）
  - iOS Developer Program(Xcode,配布)
  - iOS実機(iPhone、iPad等 iOS6.0～7.1)

Mac上にWindows仮想環境を作成して、1台で構成することも可能です



3-1. ネイティブアプリケーションの開発環境

- Androidネイティブアプリケーション開発に必要な環境
  - Windows端末（Delphi/400 XE5）
  - Android実機
    （Android 2.3.3以降のARM7 + NEON対応デバイス）

開発	コンパイル	インストール・実行

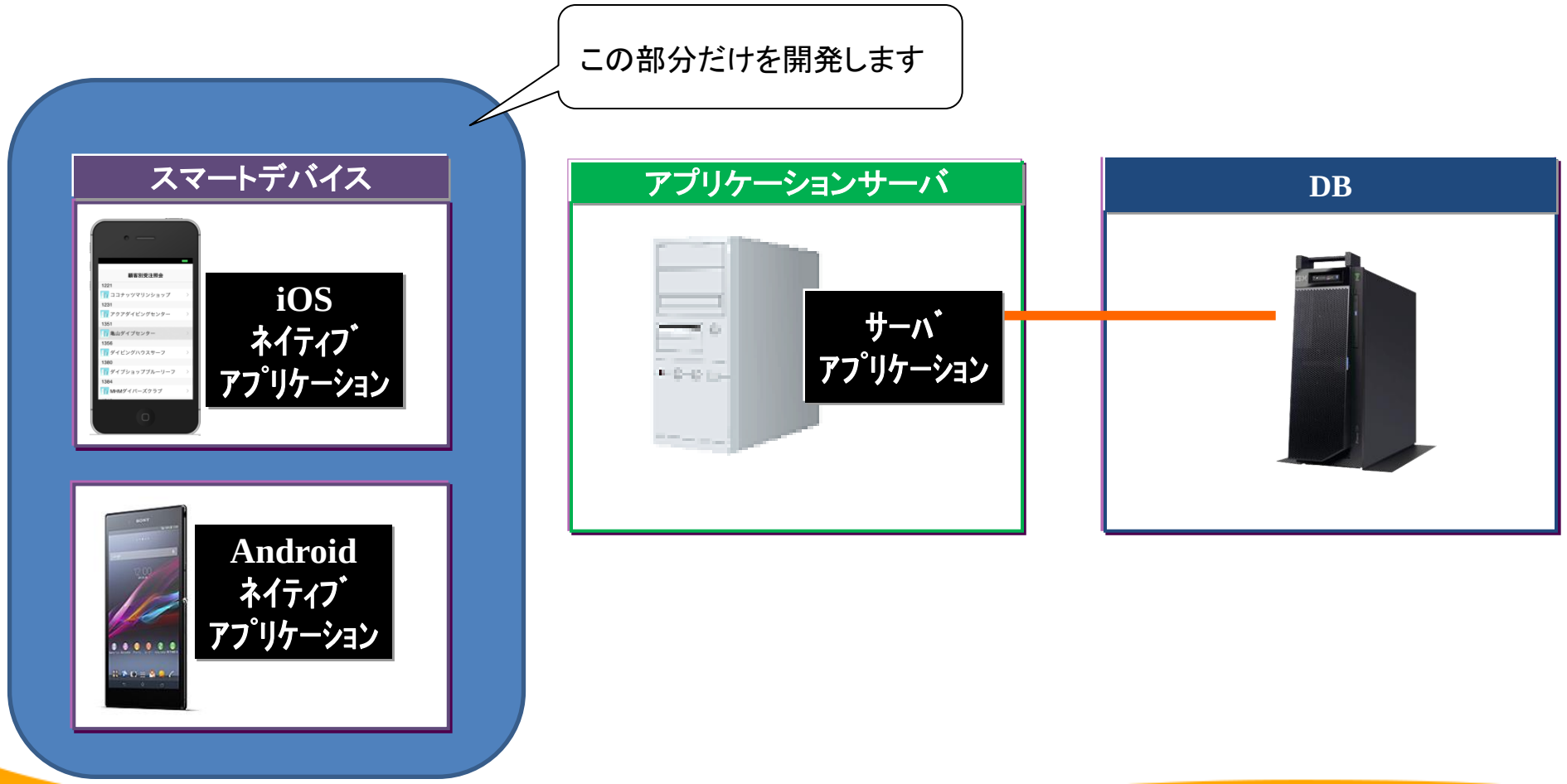
Windows

ANDROID SDK

Android
Phone/Tablet

3-2. 簡単なネイティブアプリケーションの開発

- 簡単なネイティブアプリケーションの開発



3-2. 簡単なネイティブアプリケーションの開発

- デモで開発するネイティブアプリケーション
  スマートデバイス機能（カメラ）を連携したアプリケーションを開発

iPhone

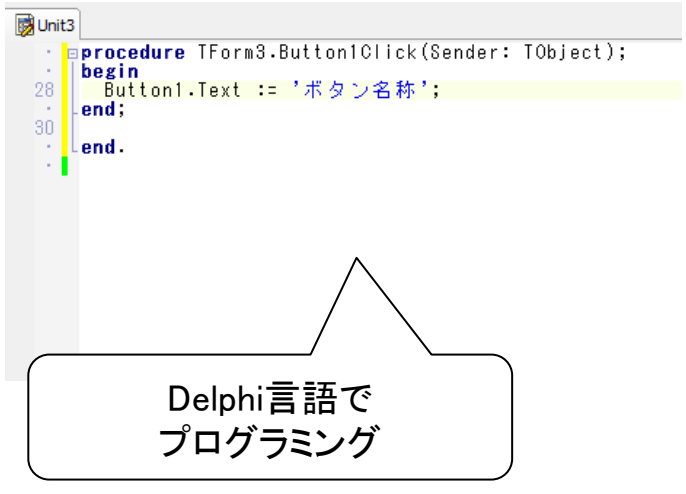
Android

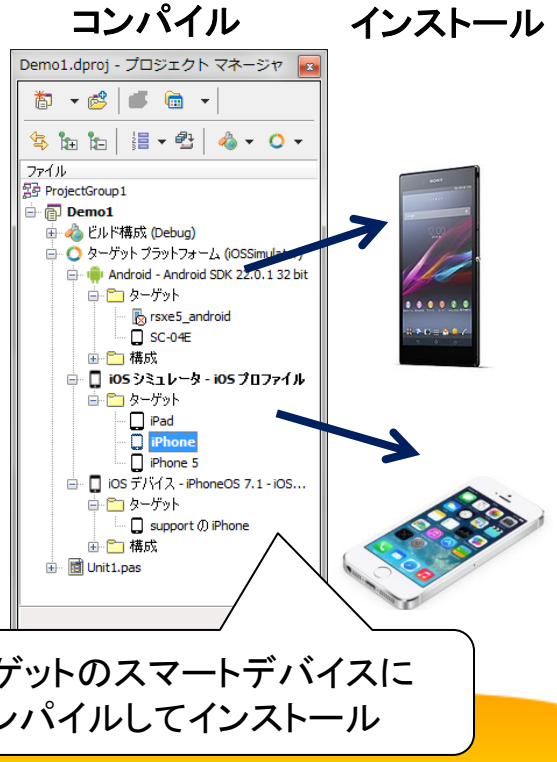
ボタンをタッチするとカメラで撮影が行え
撮影した写真を画面に取り込みます。

3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発の流れ
  ネイティブアプリケーションの開発は、従来のC/S、Web開発手法と同じです。

① コンポーネントの配置 → ② イベントにプログラミング → ③ コンパイル/インストール

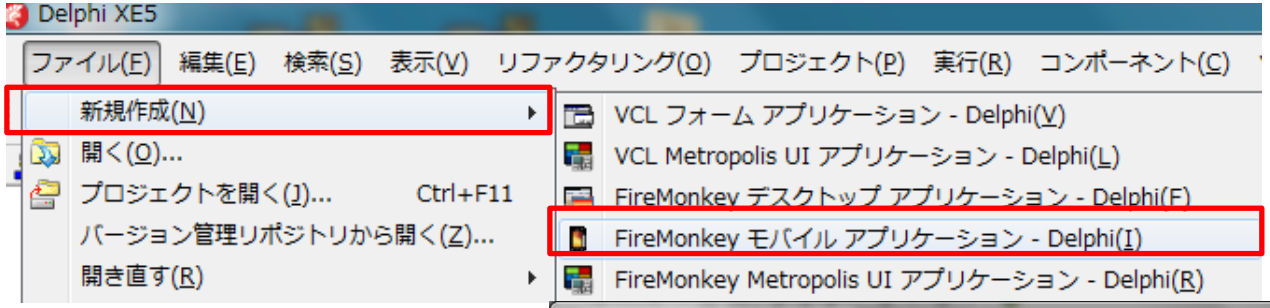


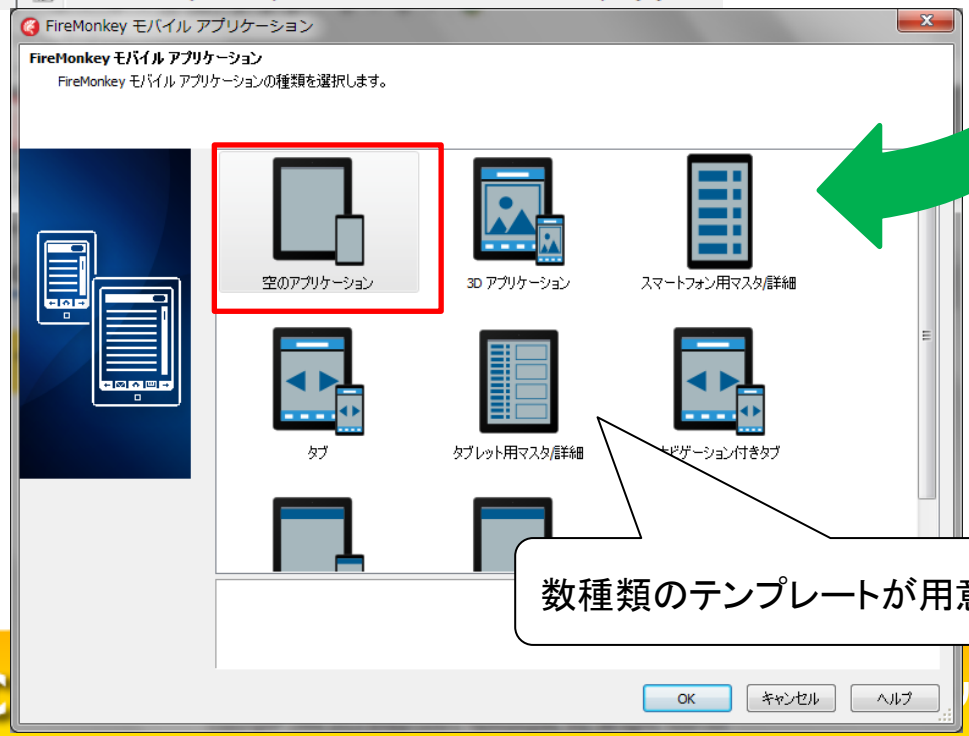


3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順1

  新規作成よりモバイルアプリケーションを選択



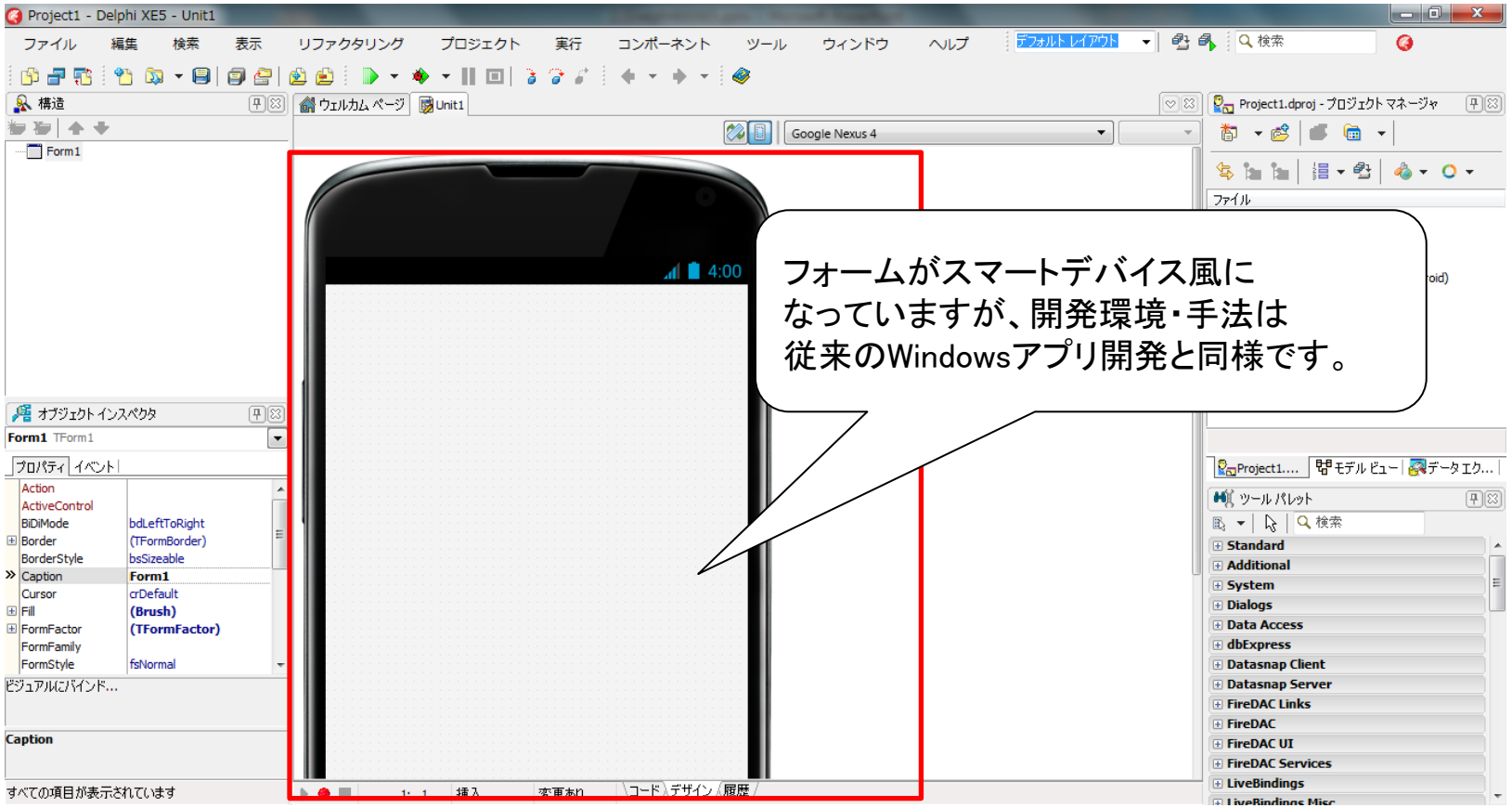


数種類のテンプレートが用意されています。

3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順2

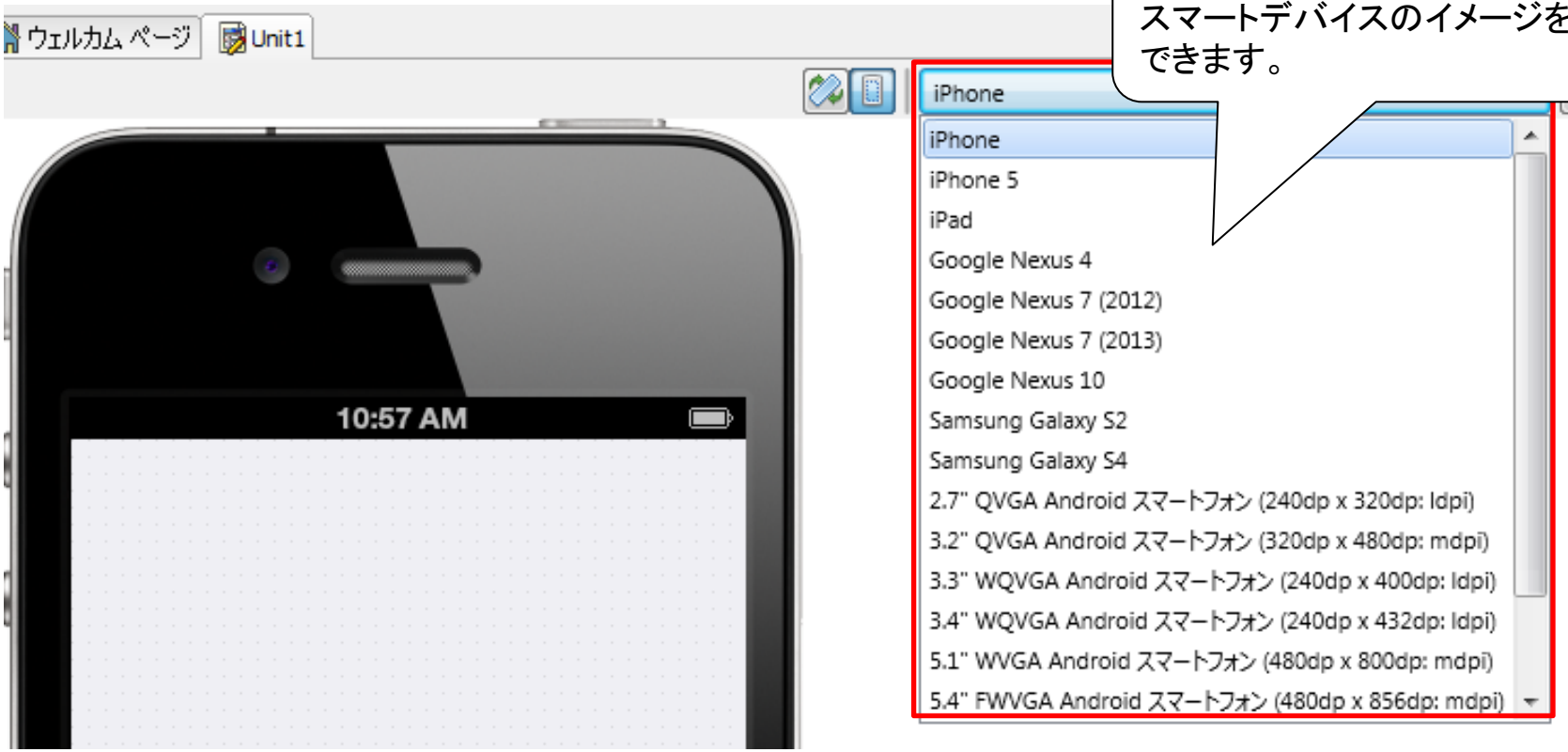
ネイティブアプリケーション開発画面



3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順3

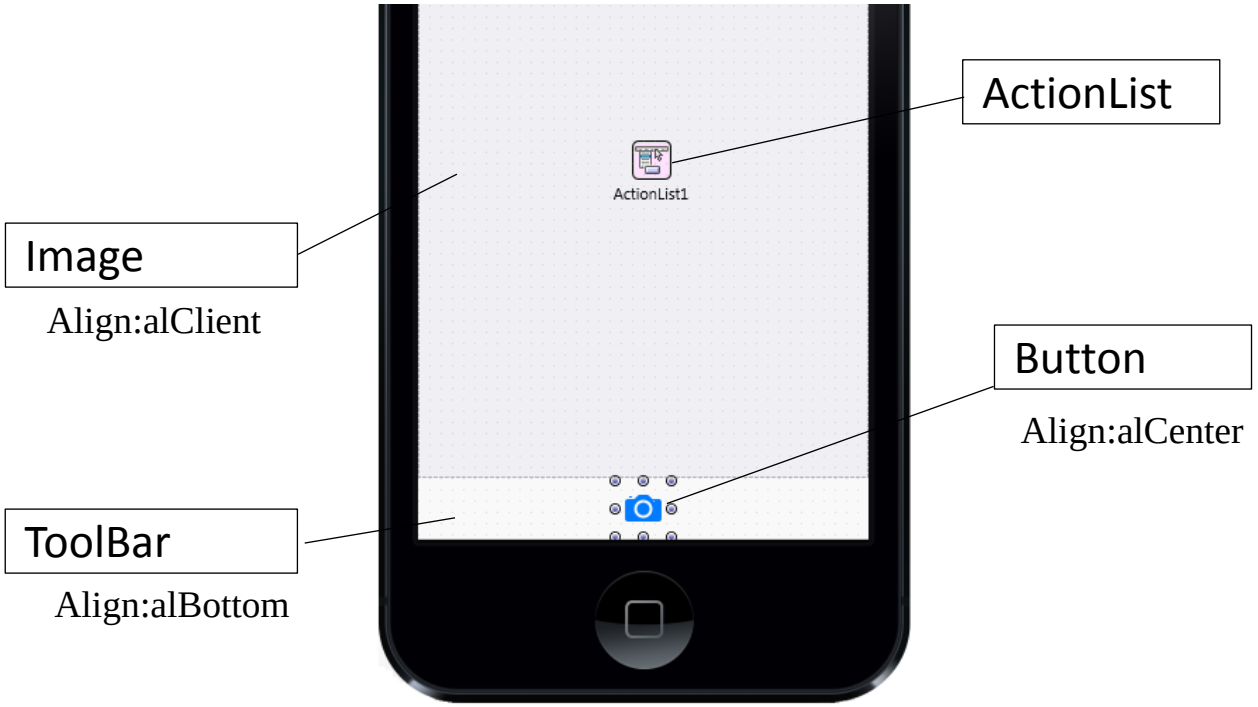
  フォームテンプレートを選択

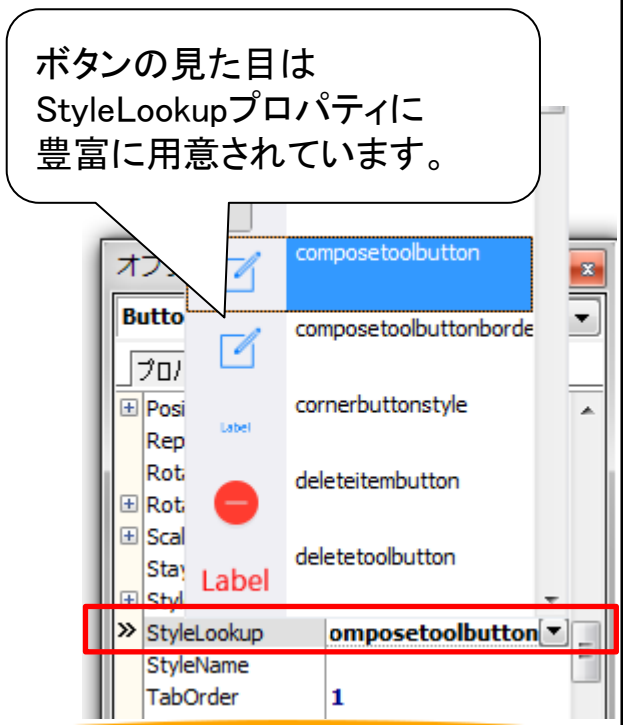


3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順4

  フォームに次のコンポーネントを配置
  ToolBar、Button、Image、ActionList

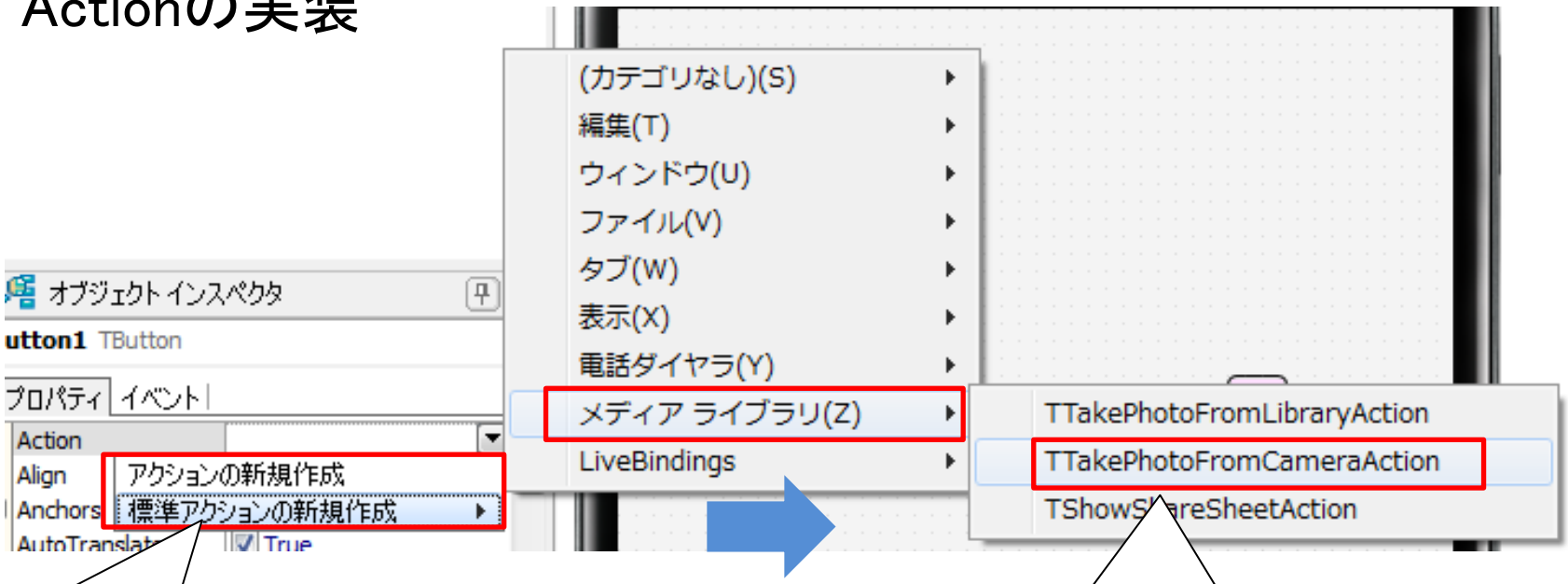




3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順5

Actionの実装



ButtonのActionプロパティで
標準アクションの新規追加

TTakePhotoFromCameraAction
を選択

3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順5

  Actionのイベントにプログラムを実装

OnDidFinishTakingイベントを作成

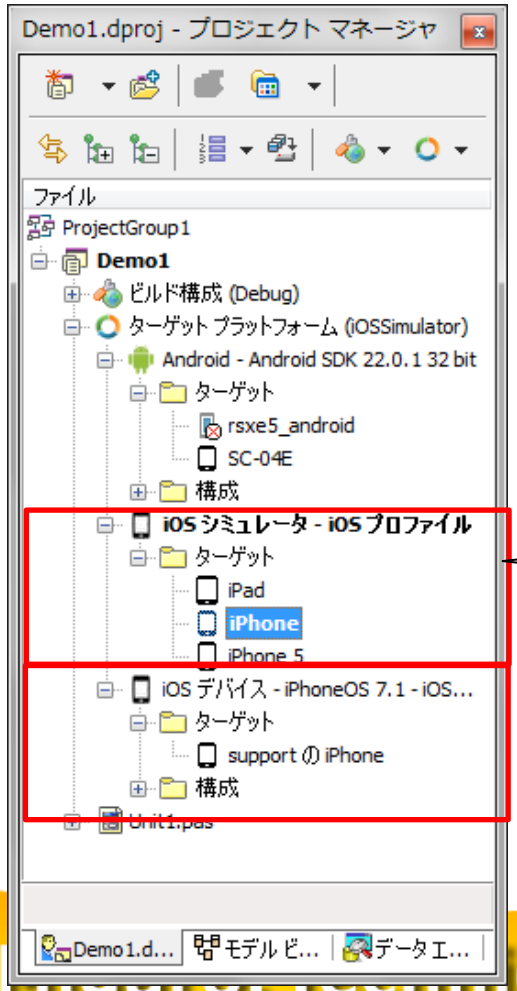
OnDidFinishTaking処理(カメラ撮影終了処理)

```
procedure TForm1.TakePhotoFromCameraAction1DidFinishTaking(Image: TBitmap);
begin
  Image1.Bitmap.Assign(Image);
end;
```

3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順6

  iOS実機向けにコンパイル



iOSシミュレータ向けのコンパイルも可能

iOS実機向けコンパイル

3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順7

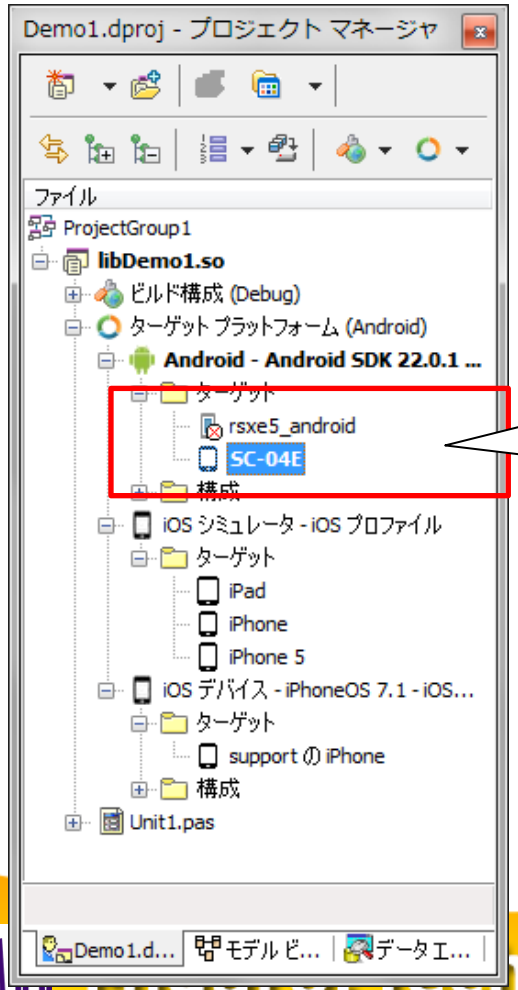
  iOSアプリケーションの実行

スマートデバイスの機能を使えば、カメラ撮影を連携したアプリケーションも
PCやWebのアプリケーションと比べて、簡単に実現ができます。

3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順8

Android実機向けにコンパイル



同じプログラムを
Android実機向けに
コンパイル

3-2. 簡単なネイティブアプリケーションの開発

- ネイティブアプリケーション開発手順9

  Androidアプリケーションの実行

1つのプログラムからiOS、Androidのネイティブアプリケーションを開発できます。

3-2. 簡単なネイティブアプリケーションの開発

- 補足：iOSとAndroidの違い1
  ハードウェアキーの違い
  Androidには「戻るボタン」や「メニューボタン」が物理的に存在しますが、
  iOSには「ホームボタン」しかありません。
  例えばiOSで「戻るボタン」が前提のアプリを作成してしまうと意図した画面遷移
  操作が行えなくなります。そのため、OS・ハードの違いを把握した画面設計は
  非常に重要となってきます。

iOS

ホームボタン
しかない

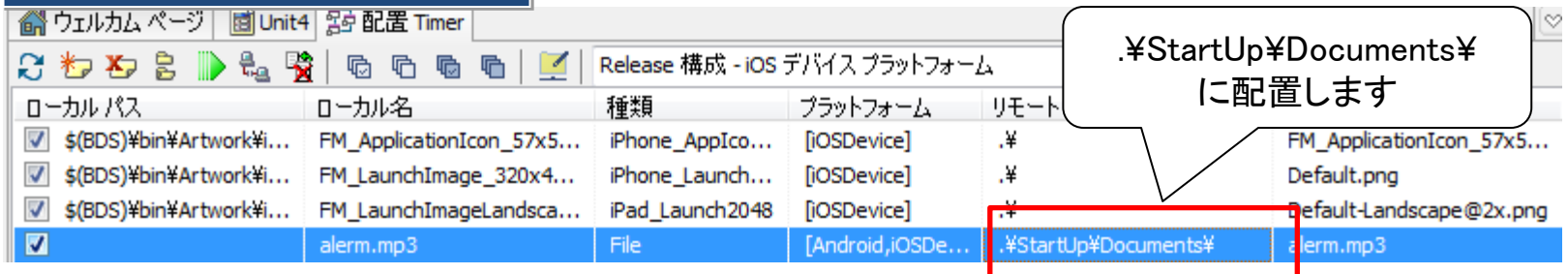
Android

ホームボタン、メニューボタン、
戻るボタンがある

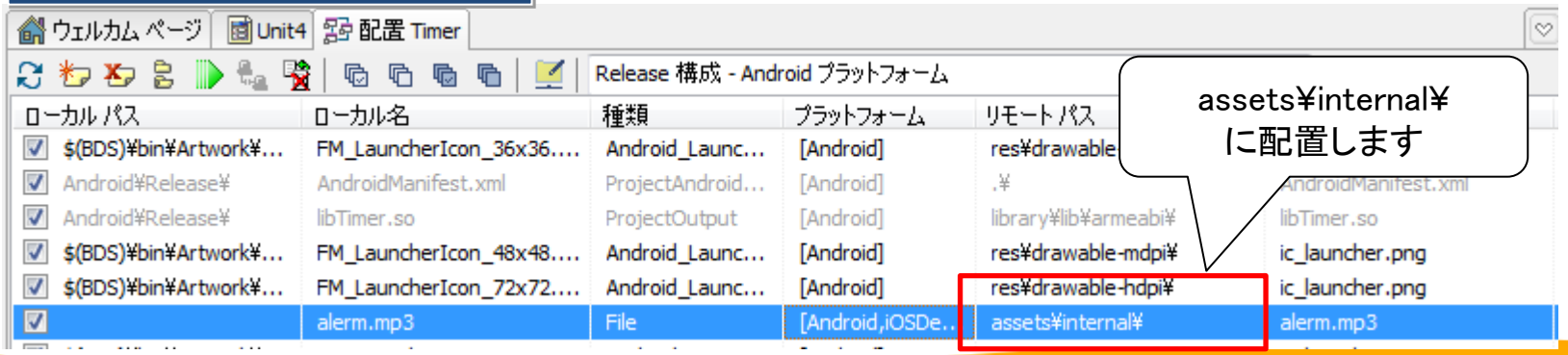
3-2. 簡単なネイティブアプリケーションの開発

- 補足：iOSとAndroidの違い2
  ファイル配置の違い（プロジェクト|配置から設定）
  音源ファイルや動画ファイルなど、アプリケーション内で固有で持ちたい場合、配置（保存）先のパスはプラットフォームによって異なります。

iOS

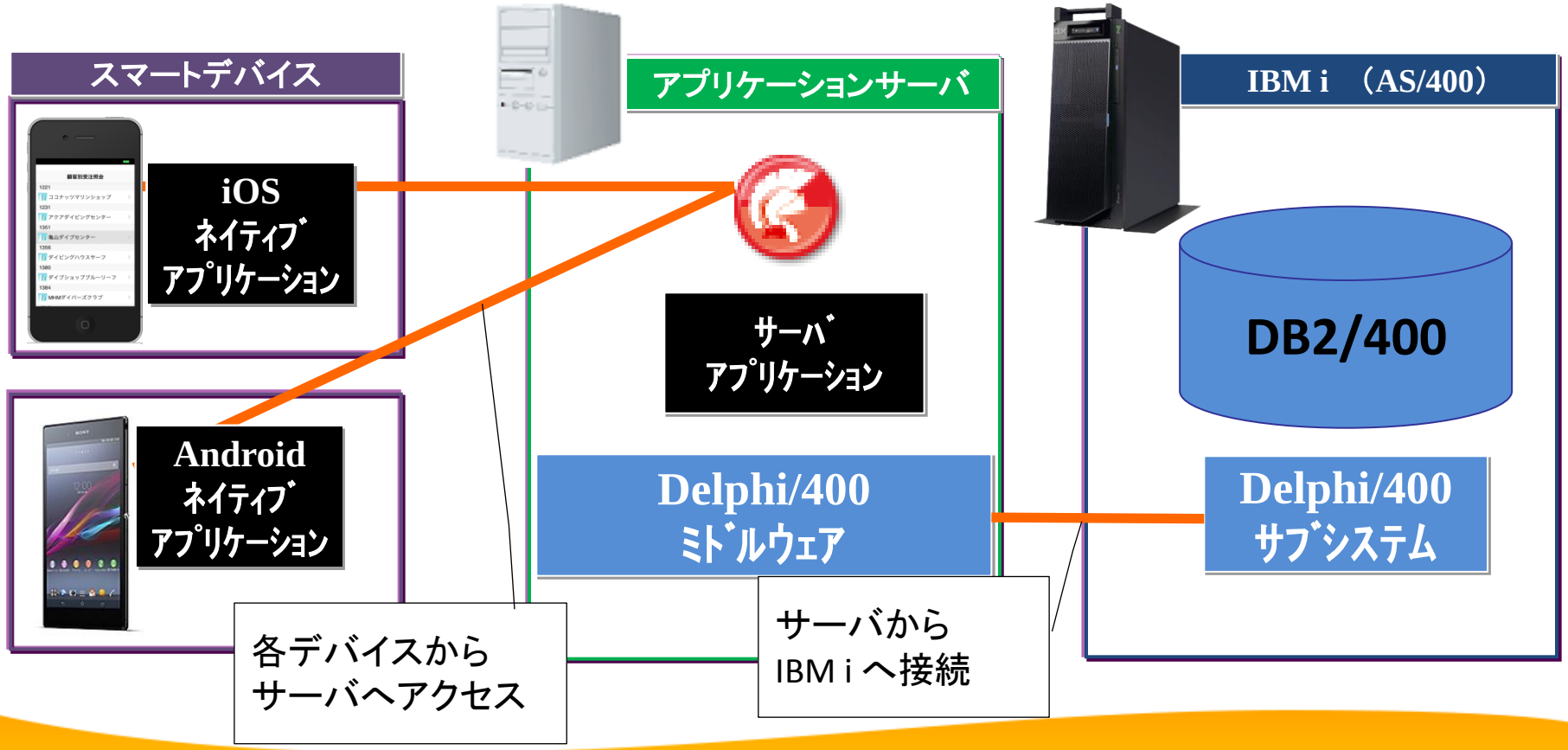


Android



3-3. IBM i に接続するネイティブアプリケーションの開発

- IBM i に接続するネイティブアプリケーションの仕組み
  ネイティブアプリケーションからIBM i に接続する仕組みは、
  Webアプリケーションに近い、サーバを経由した3階層方式になります。



3-3. IBM i に接続するネイティブアプリケーションの開発

- サーバアプリケーションとは？
  アプリケーションサーバ（中間サーバ）からデータベースに接続、処理を行うアプリケーションです。
  ネイティブアプリケーションは、サーバアプリケーションを経由してデータベースにアクセスすることができます。

Delphi/400で開発するサーバアプリケーション

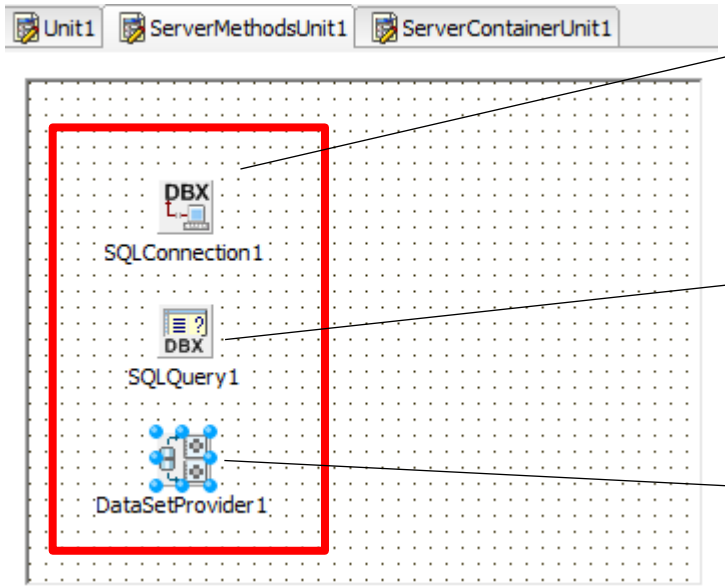
Delphi/400ではサーバアプリケーションを『DataSnap』で簡単に開発できます。
『DataSnap』はサーバアプリケーション専用の開発機能です。
サーバアプリケーションは、SQLConnectionやSQLQuery等のDBコンポーネントを設定したり、関数をプログラミングすることで機能を実装できます。

サーバアプリケーション開発手順の詳細はP61以降の補足資料記載

3-3. IBM i に接続するネイティブアプリケーションの開発

- DataSnapサーバアプリケーションの設定概要

【スマートデバイスから IBM i へ SQLを実行できるサーバ機能を実装】
SQLConnection、SQLQuery、DataSetProviderを配置して構成します。



【SQLConnectionコンポーネント】
dbExpressドライバを“CO400”に設定

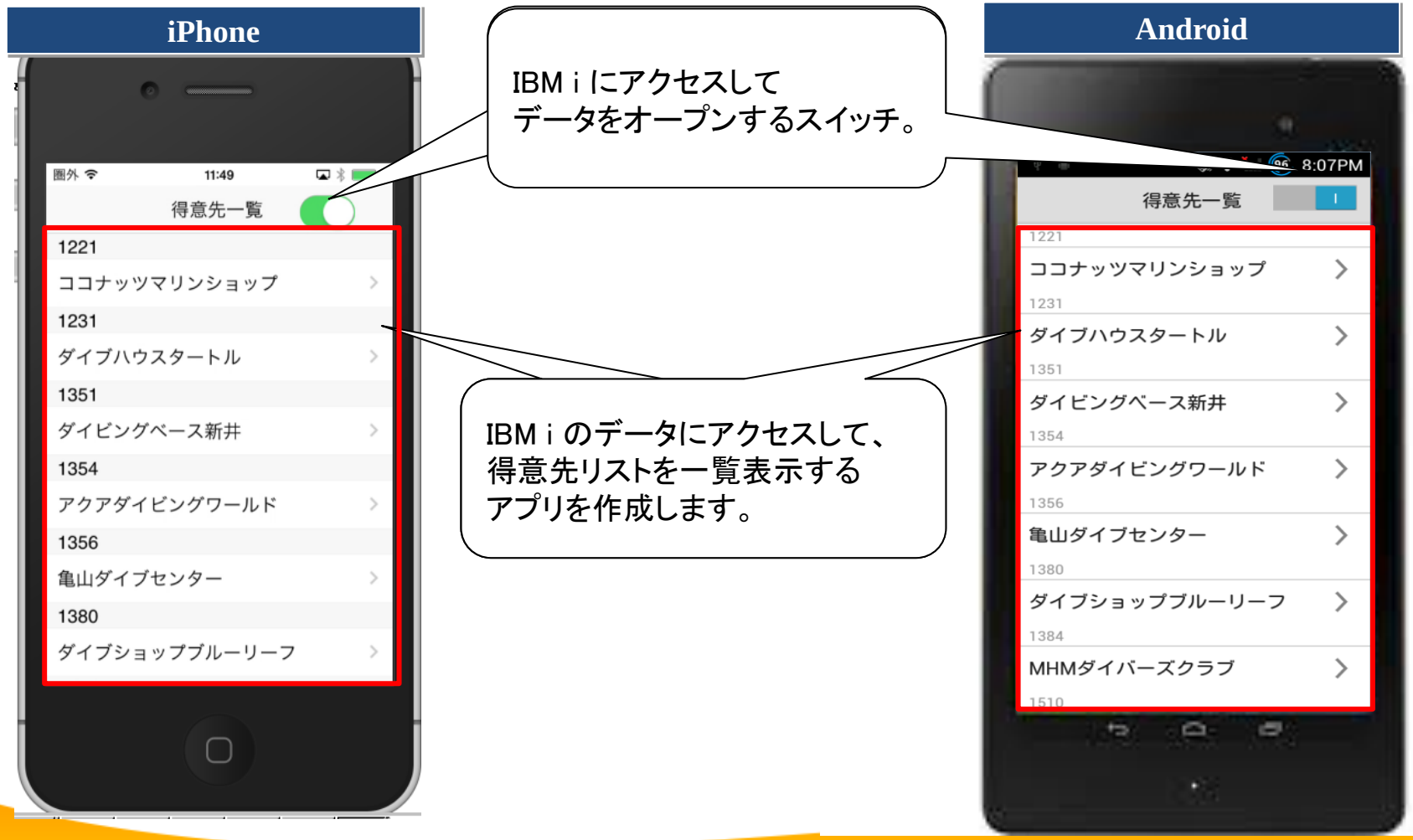
【SQLQueryコンポーネント】
接続を” SQLConnection1”に設定

【 DataSetProviderコンポーネント】
Option”poAllowCommandText”を
“True”に設定
**この設定でクライアントアプリから
SQLが自由に発行できます**

今回は、このプログラムをアプリケーションサーバ上で起動しておきます。

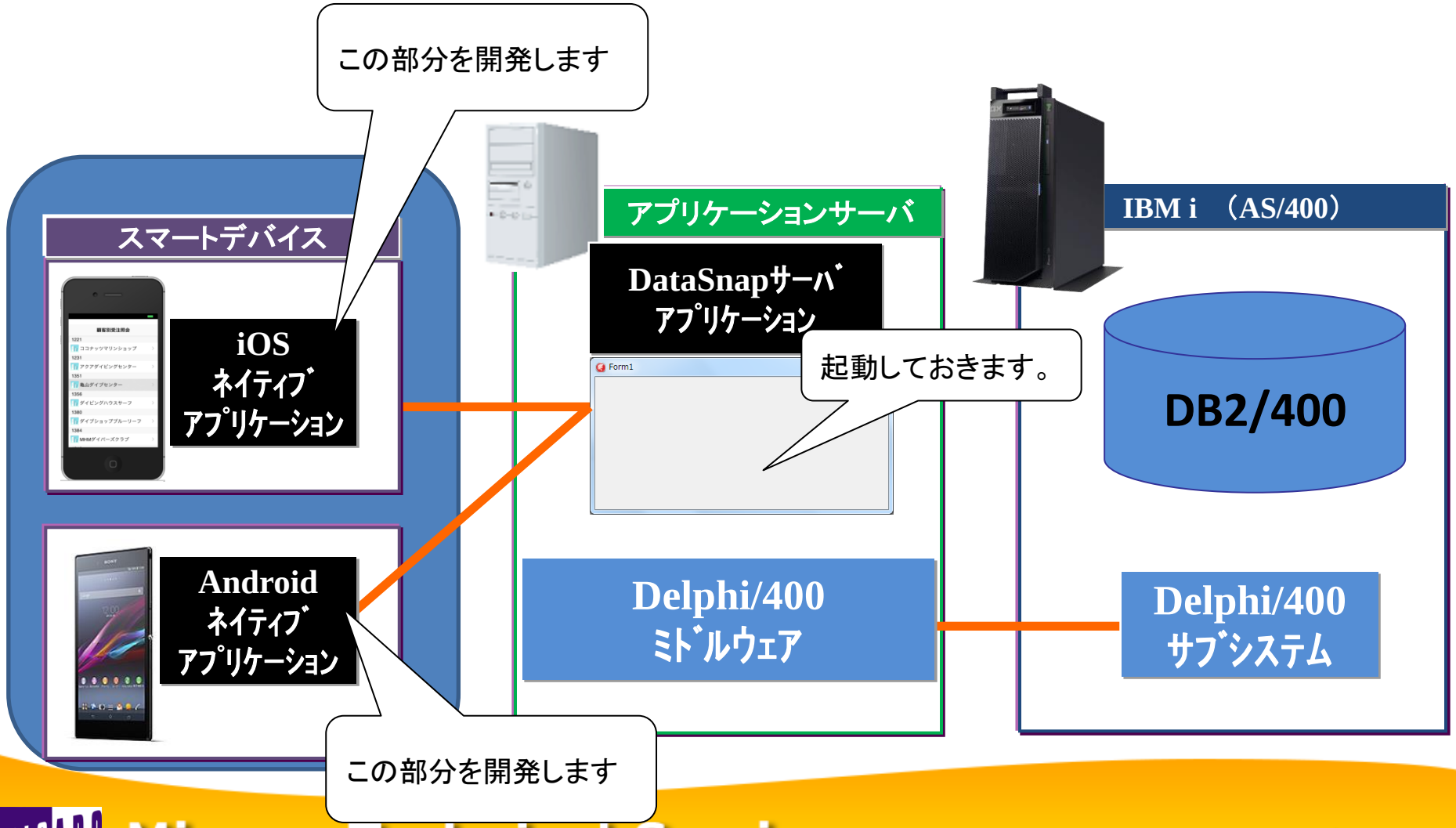
3-3. IBM i に接続するネイティブアプリケーションの開発

- デモで開発するネイティブアプリケーション



3-3. IBM i に接続するネイティブアプリケーションの開発

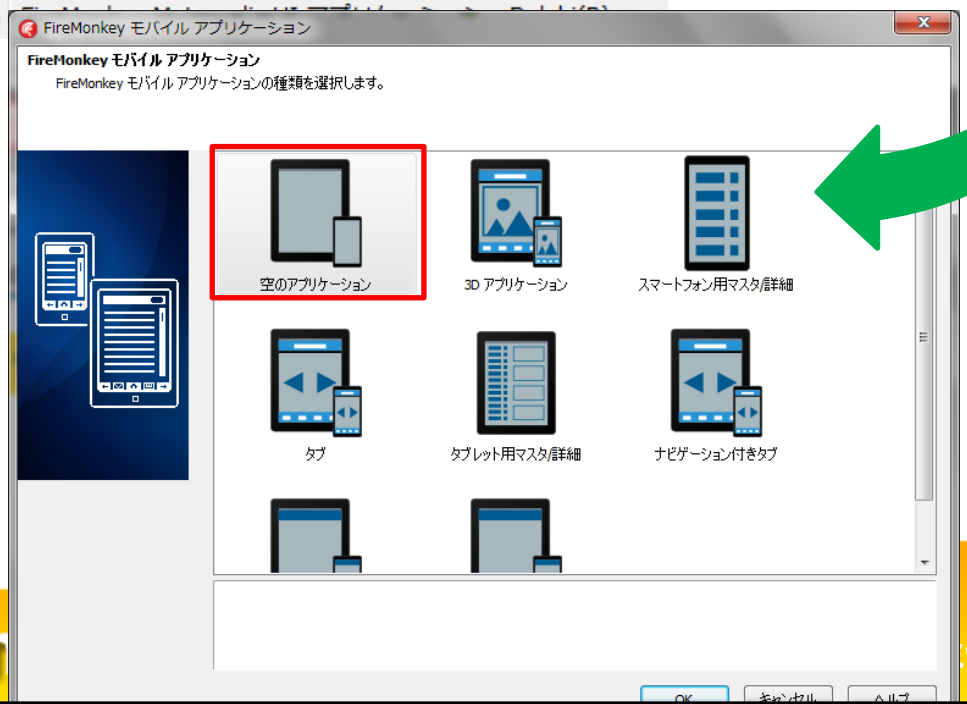
- IBM i に接続するネイティブアプリケーション



3-3. IBM i に接続するネイティブアプリケーションの開発

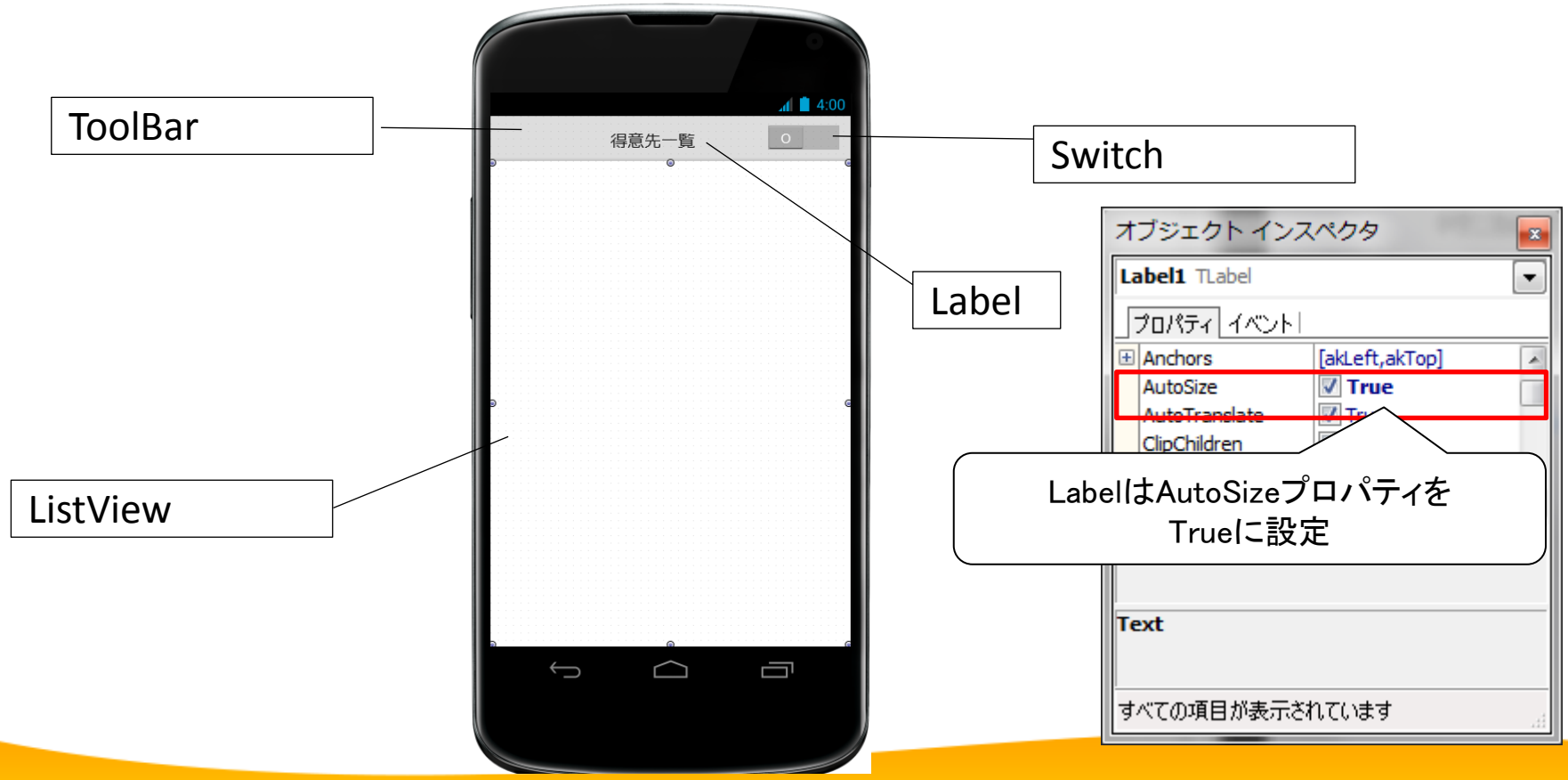
- IBM i に接続するネイティブアプリケーション開発手順1

メニューの[ファイル|新規作成]から
『モバイルアプリケーション』を選択



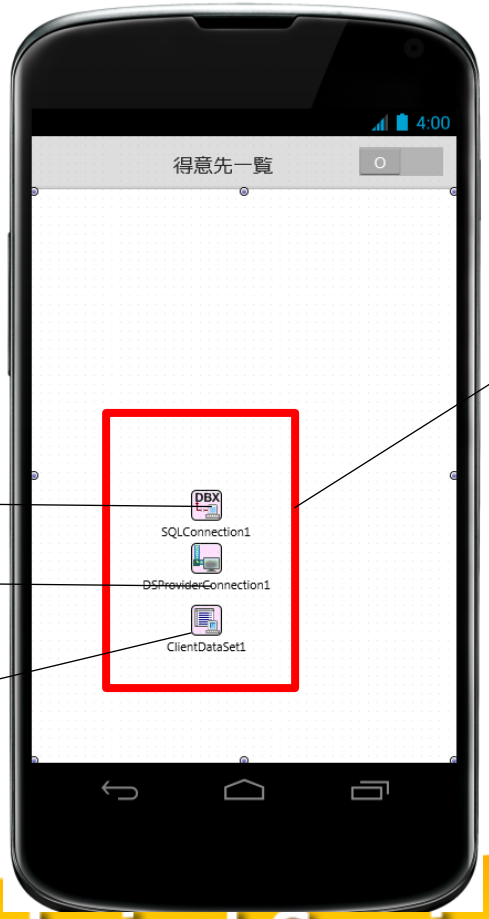
3-3. IBM i に接続するネイティブアプリケーションの開発

- IBM i に接続するネイティブアプリケーション開発手順2
  フォームに次のコンポーネントを配置
  ToolBar、Label、Switch、ListView



3-3. IBM i に接続するネイティブアプリケーションの開発

- IBM i に接続するネイティブアプリケーション開発手順3
  フォームに次のコンポーネントを配置
  SQLConnection、DSProviderConnection、ClientDataSet



SQLConnection

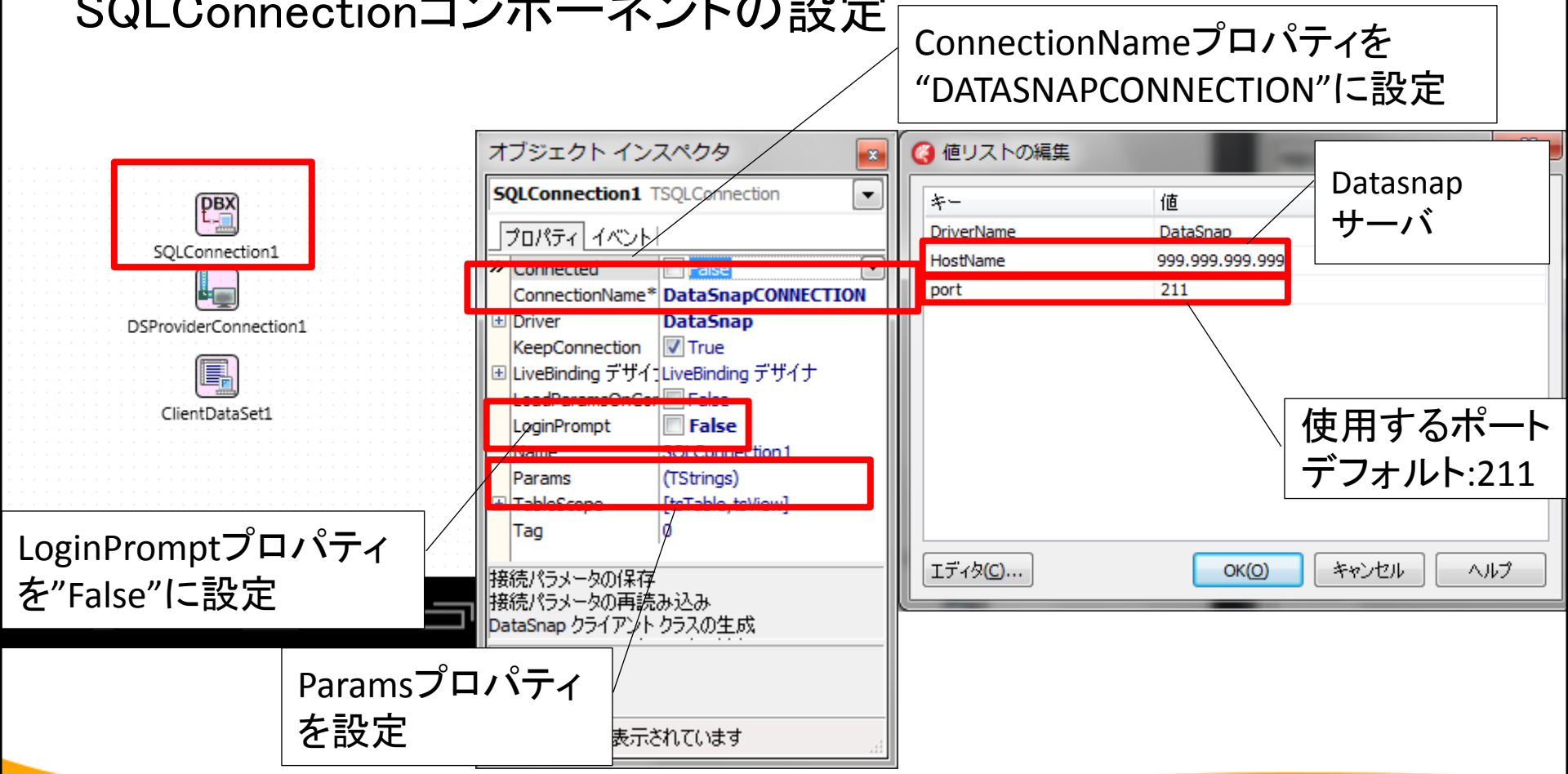
DSProviderConnection

ClientDataSet

今回はサーバアプリケーションを利用して、IBM i のデータを表示する機能を実装します

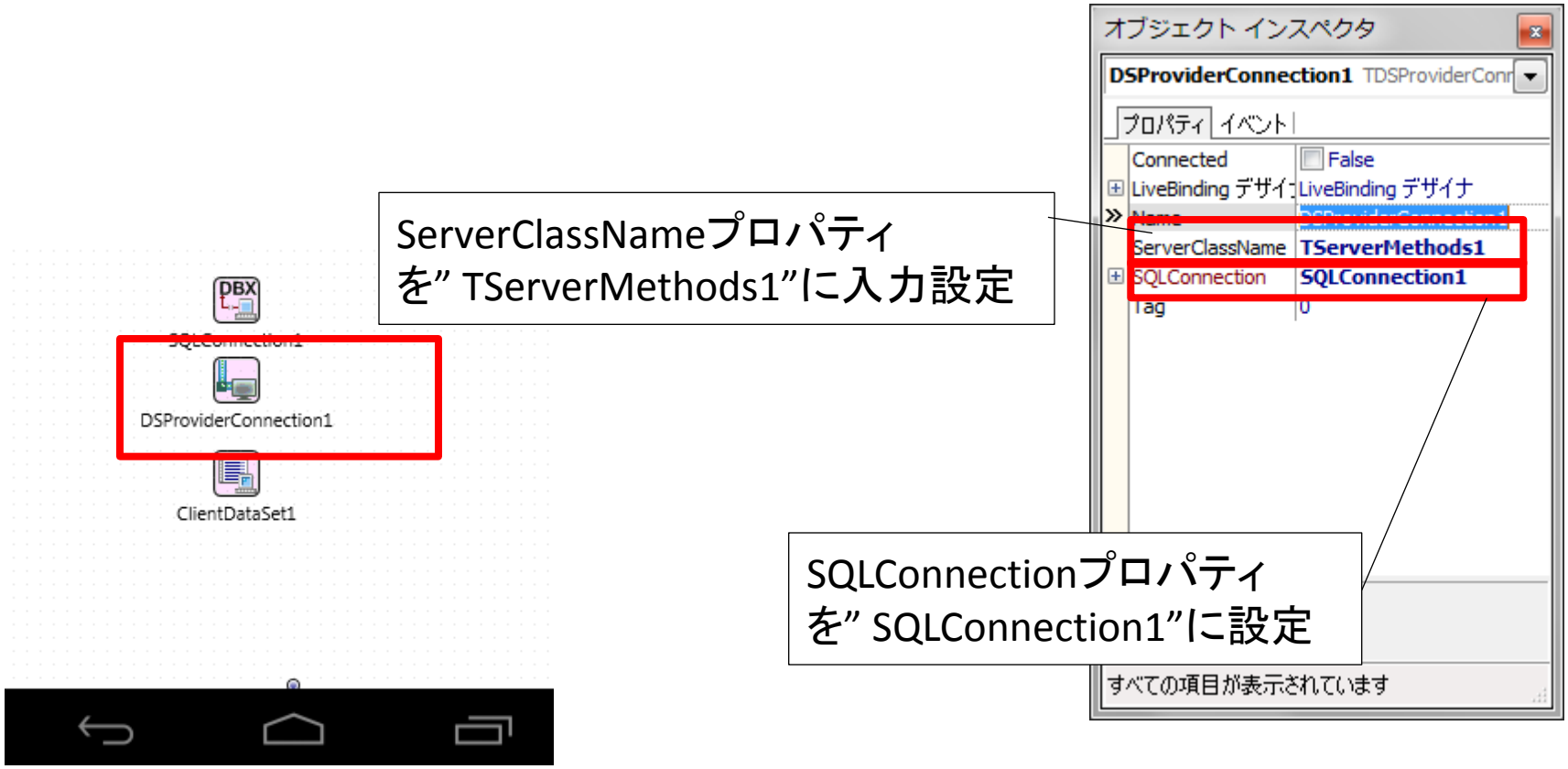
3-3. IBM i に接続するネイティブアプリケーションの開発

- IBM i に接続するネイティブアプリケーション開発手順4
  SQLConnectionコンポーネントの設定



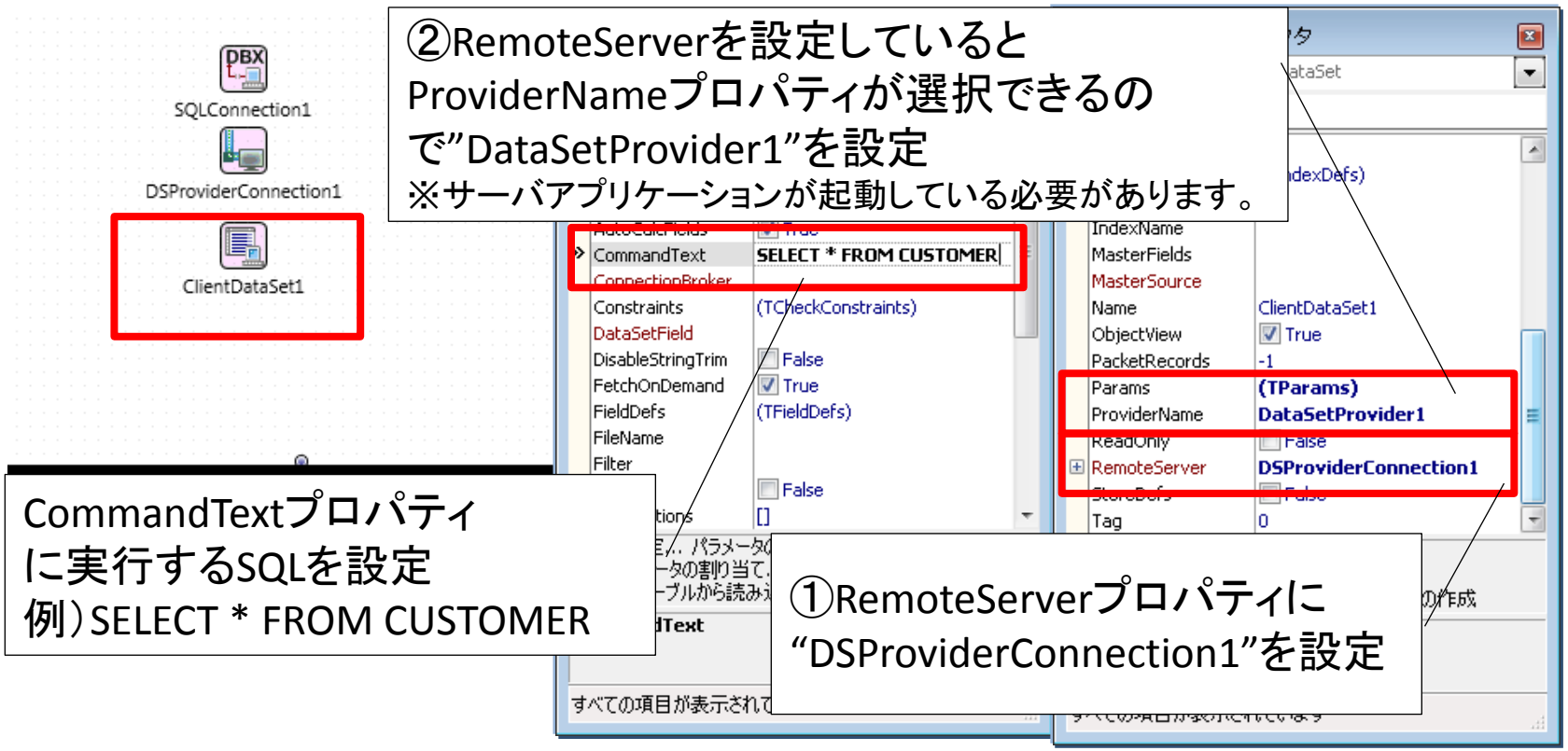
3-3. IBM i に接続するネイティブアプリケーションの開発

- IBM i に接続するネイティブアプリケーション開発手順5
  DSProviderConnectionコンポーネントの設定



3-3. IBM i に接続するネイティブアプリケーションの開発

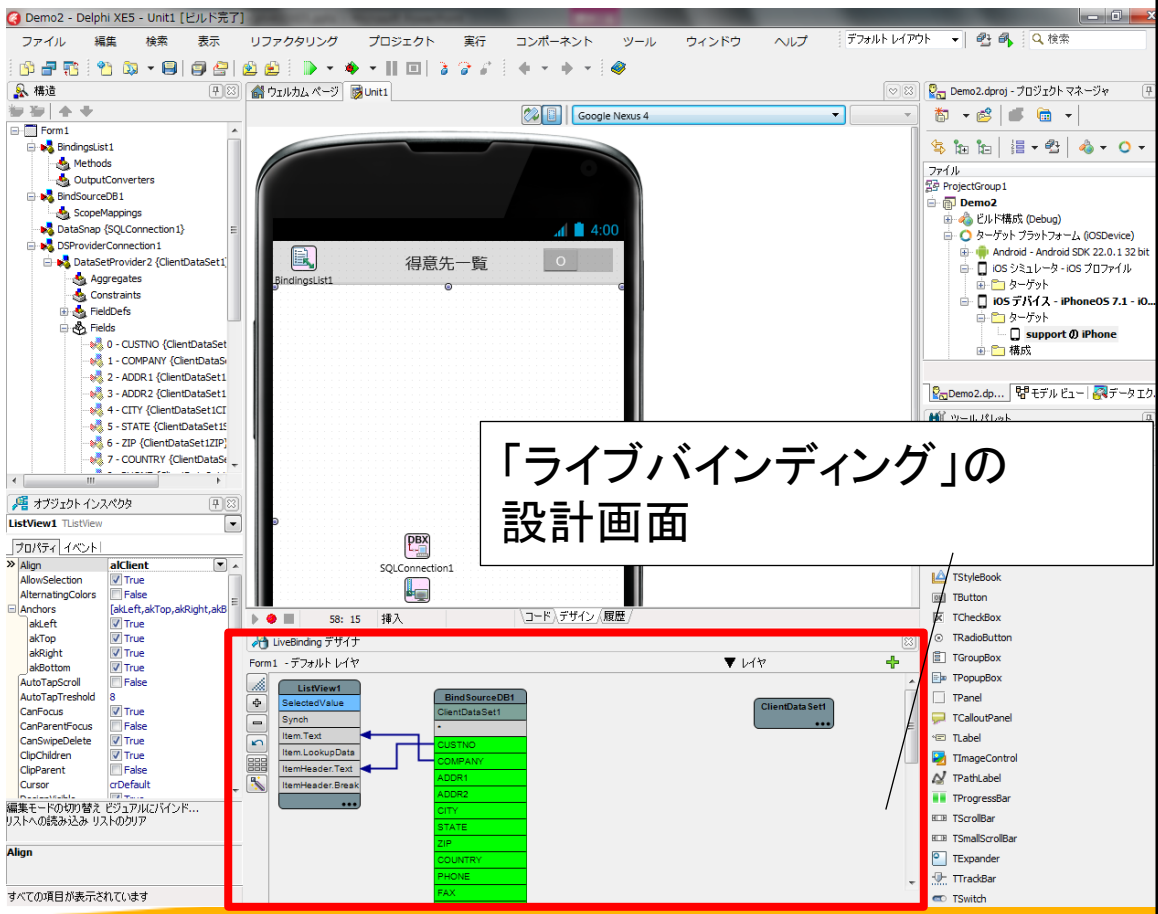
- IBM i に接続するネイティブアプリケーション開発手順6
  ClientDataSetコンポーネントの設定



3-3. IBM i に接続するネイティブアプリケーションの開発

- IBM i に接続するネイティブアプリケーション開発手順7
  データ表示をライブバインディング機能で実装

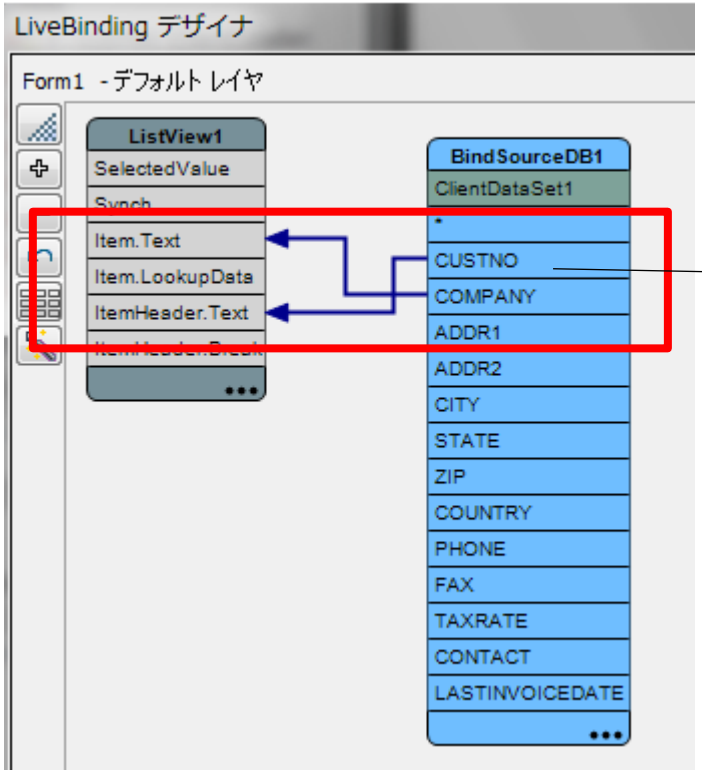
右クリックから
「ビジュアルにバインド」を起動



「ライブバインディング」の
設計画面

3-3. IBM i に接続するネイティブアプリケーションの開発

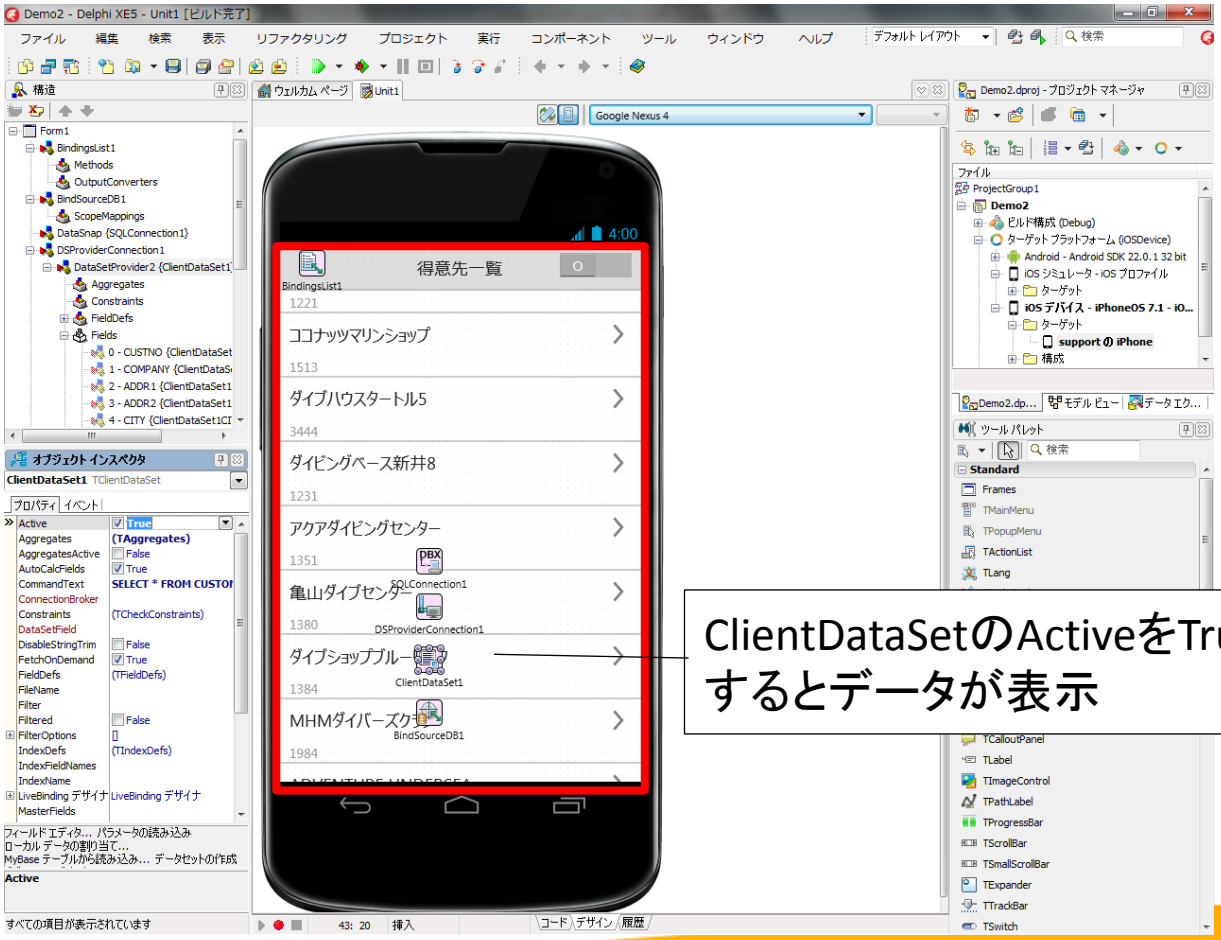
- IBM i に接続するネイティブアプリケーション開発手順8
  ClientDataSetの項目をListViewにドラッグ&ドロップでリンク



CUSTNOをItemHeaderTextにリンク
COMPANYをItemTextにリンク

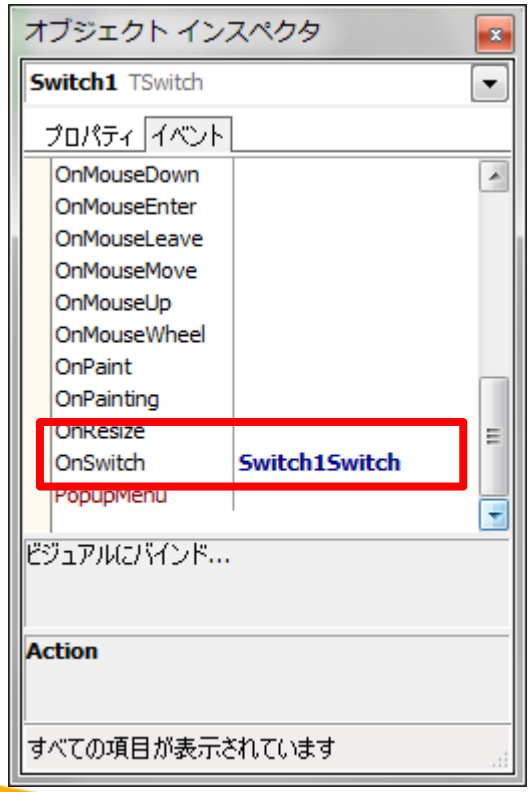
3-3. IBM i に接続するネイティブアプリケーションの開発

- IBM i に接続するネイティブアプリケーション開発手順9
  データ表示の確認



3-3. IBM i に接続するネイティブアプリケーションの開発

- IBM i に接続するネイティブアプリケーション開発手順10
  SwitchのonSwitchイベントにプログラムを実装

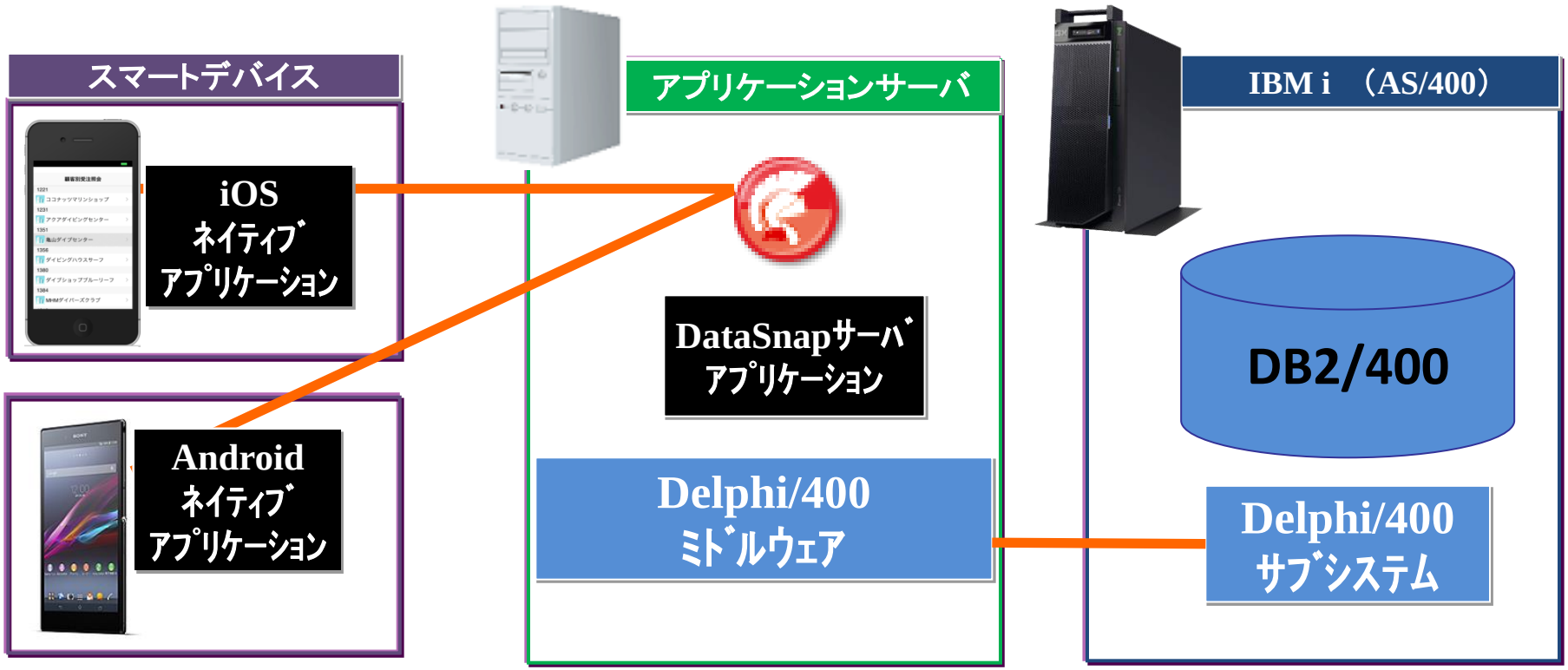


スイッチ切り替え処理

```
procedure TForm1.Switch1Switch(Sender: TObject);
begin
  ClientDataSet1.Active := Switch1.IsChecked;
end;
```

3-3. IBM i に接続するネイティブアプリケーションの開発

- 3階層によってスマートデバイスからIBM i へ接続完成



本資料では設定内容を全て記載していますが、
開発時はプロパティ設定をしていくだけなので簡単！

3-3. IBM i に接続するネイティブアプリケーションの開発

- 補足:カスタマイズ例
  選択先の受注明細画面を作成すれば、
  得意先別受注照会のようなカスタマイズも簡単に実現できます。

3-4.ネイティブアプリケーションの配布

- ネイティブアプリケーションはスマートデバイスに配布・インストールする方法が2つあります。

社内公開での配布

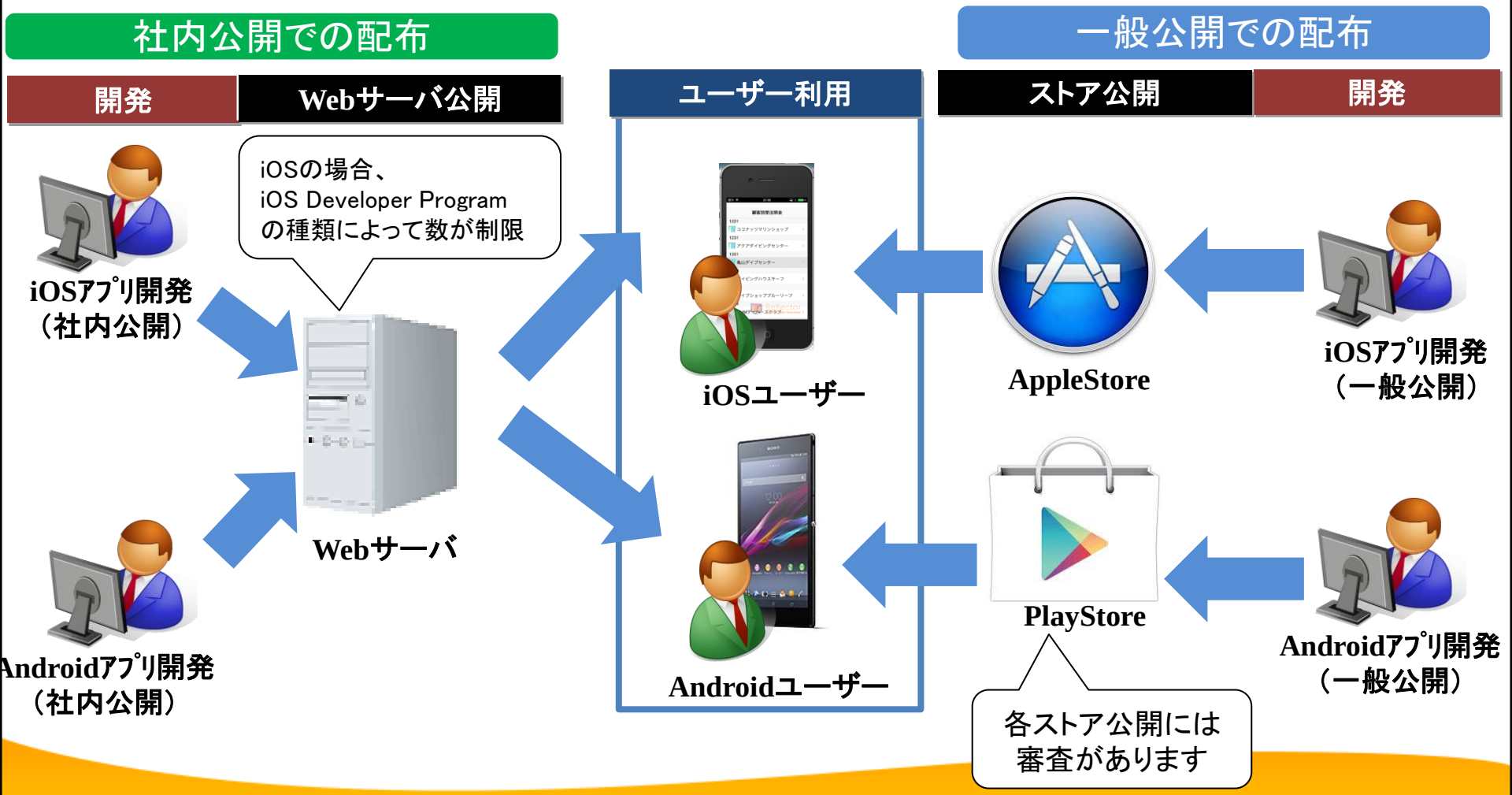
開発者はWebサーバ上にネイテブアプリケーションのファイルを公開して配布します。

一般公開での配布

開発者はiOSであればAppleStore、AndroidであればPlayStoreにネイティブアプリケーションを公開して配布します。

3-4.ネイティブアプリケーションの配布

- 社内公開と一般公開の配布方法の違い



3-4.ネイティブアプリケーションの配布

- 社内公開と一般公開のメリット/デメリット

	社内公開	一般公開
メリット	・社内だけで配布・利用できる。 ・審査がないため、社内専用のアプリケーションが開発できる。	・ストアで公開するため、どこからでもすぐにインストールして利用できる。
デメリット	・Webサーバ等を用意して、配布環境の構築・運用が必要。	・誰でも利用できてしまう。 ・公開には審査が必要。 （自社用アプリの公開は難しい）

社内公開するアプリは、配布用のコンパイルしたファイルをWebサーバ上に配置して、リンクでダウンロードできるように準備します。

【配布用のコンパイルファイル】
iOS:　ipa、ipListファイル　　Android:　apkファイル

4.まとめ

4.まとめ

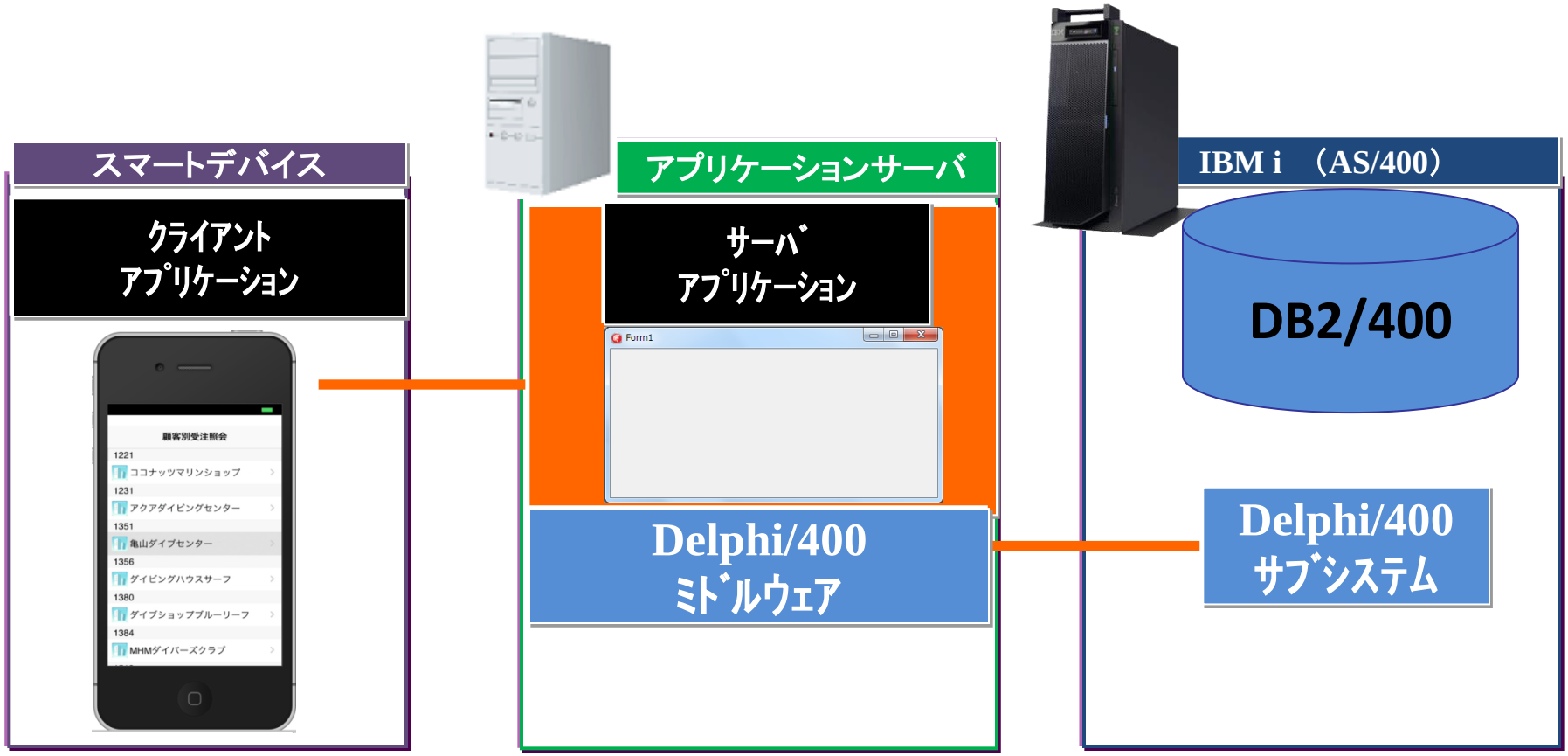
- 企業導入されるスマートデバイスはiOS、Androidが主流
- Delphi/400XE5ではiOSとAndroidのネイティブ開発が可能
- IBM i への接続はDataSnapサーバを使った3階層方式
- 配布は社内公開と一般公開で方法が異なる

ご静聴ありがとうございました

補足資料

補足資料

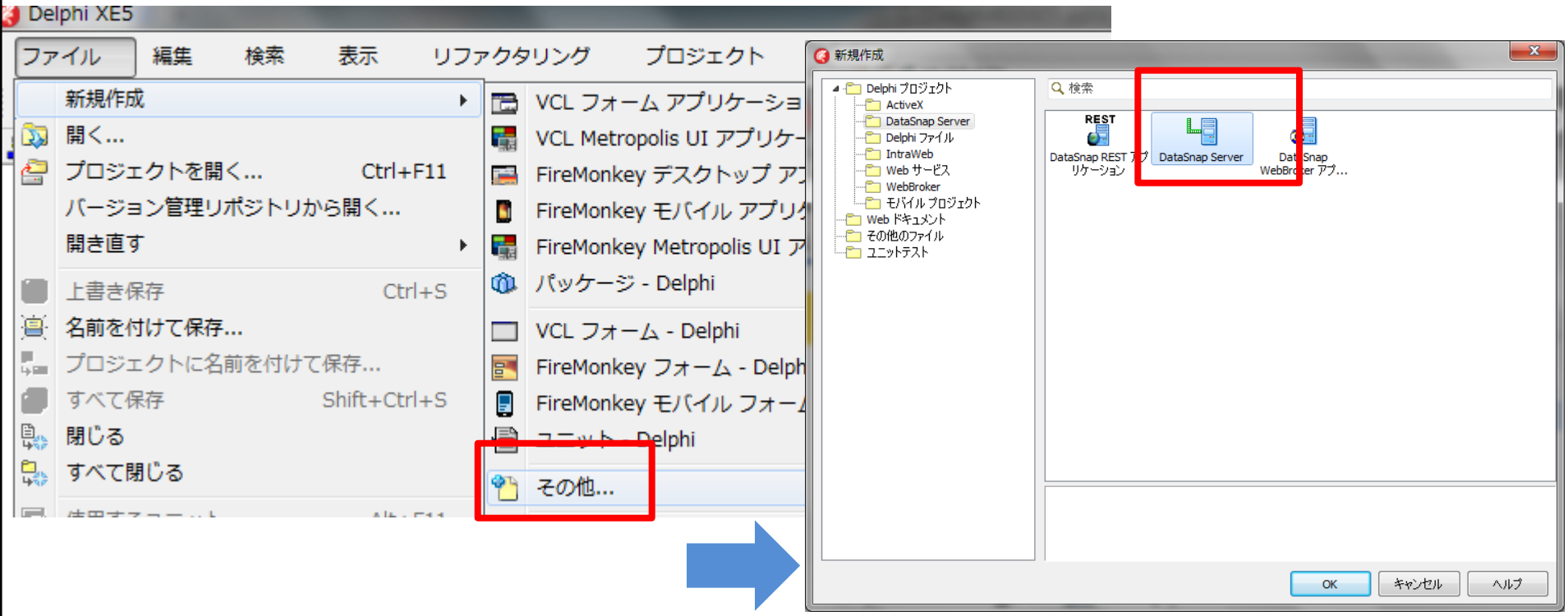
- サーバアプリケーションの開発手順



補足資料

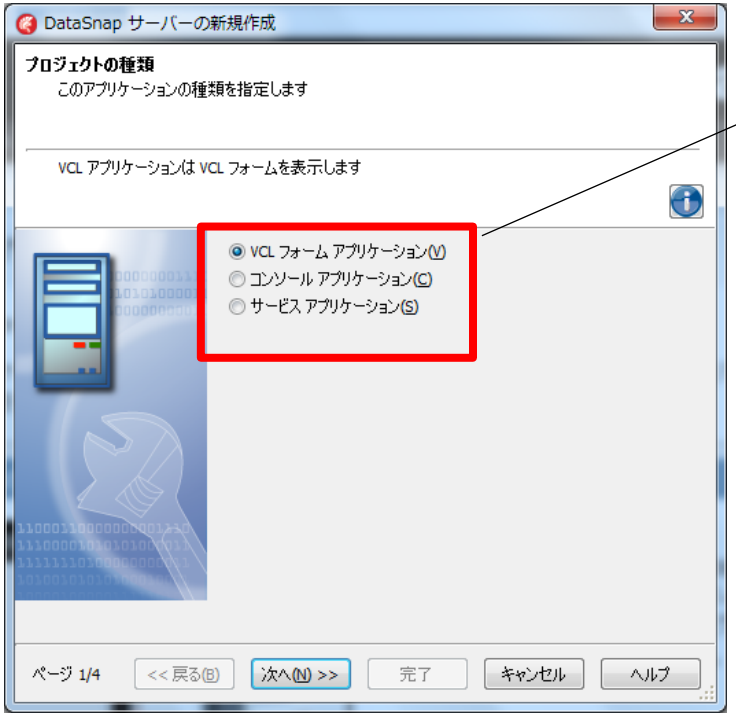
- サーバアプリケーションの開発手順1

メニューの[ファイル|新規作成|その他]から『DataSnapServer』を選択



補足資料

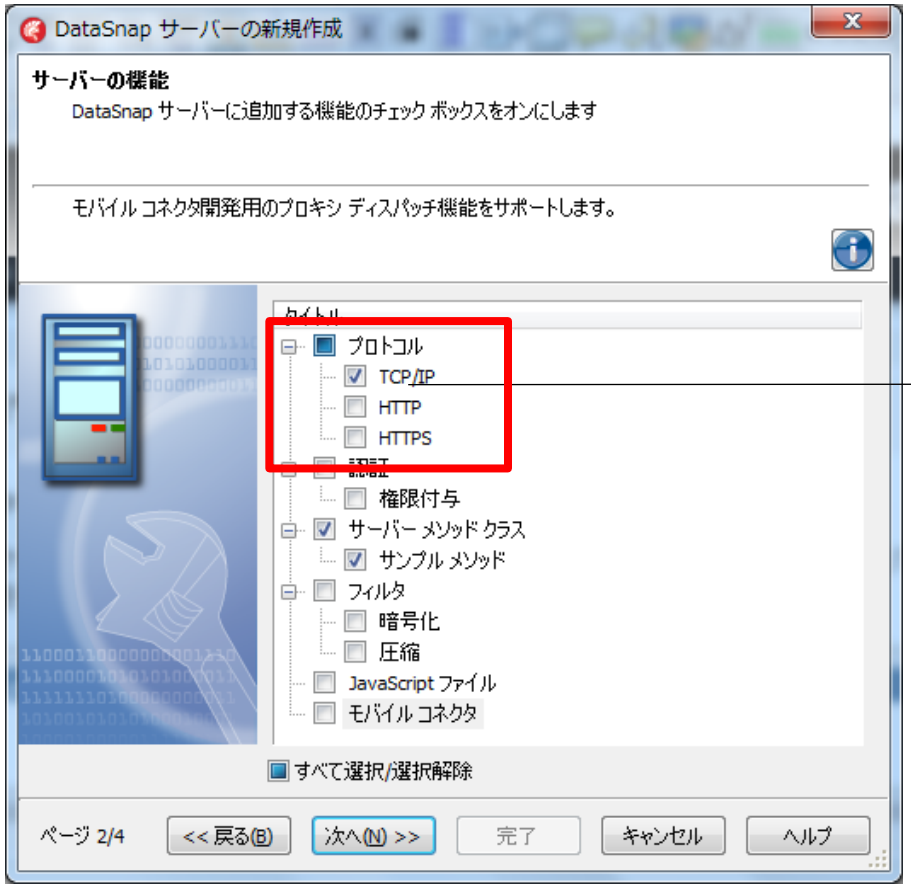
- サーバアプリケーションの開発手順2
  DataSnapServerの形式を選択



サーバ上でどのように実行するかを選択します。
サーバに常駐で起動させる場合は
サービスアプリケーションを使います。

補足資料

- サーバアプリケーションの開発手順3
  通信プロトコルを設定



接続通信の設定
標準はTCP/IPを使います。
※HTTP経由での通信も可能です。

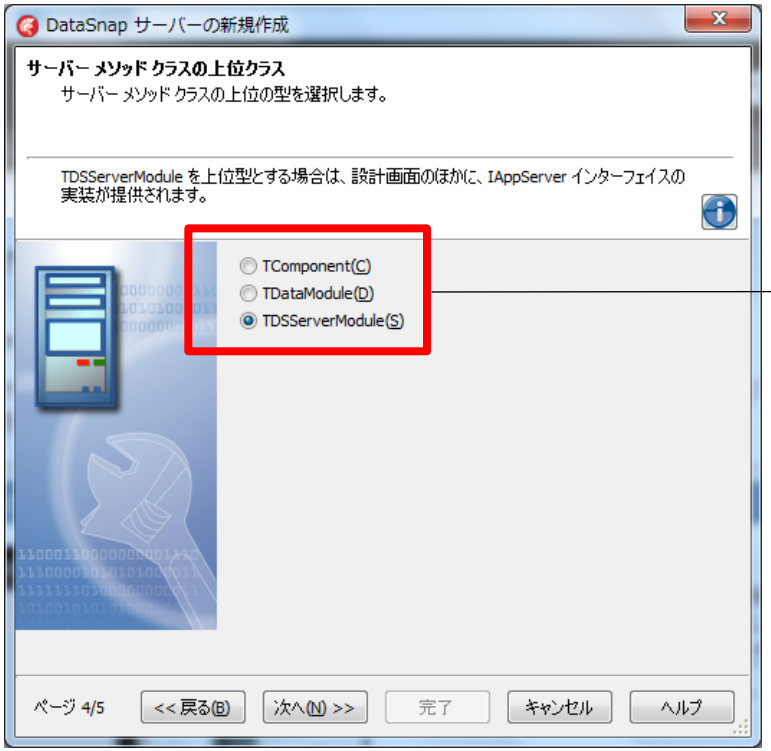
補足資料

- サーバアプリケーションの開発手順4
  使用するポート番号を指定

サーバ接続に使用するポート番号を指定します。(デフォルト211)

補足資料

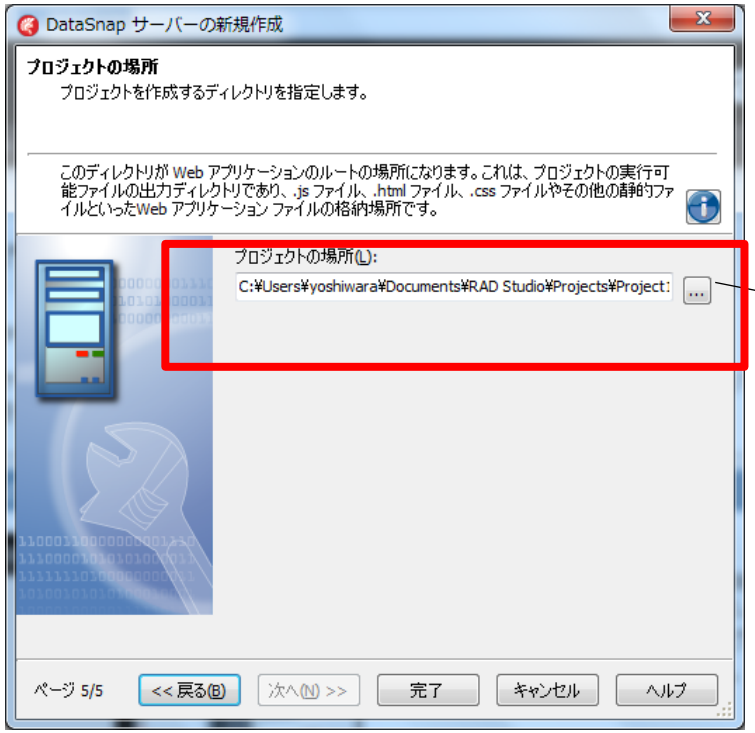
- サーバアプリケーションの開発手順5
  サーバーメソッドの上位クラスを選択



データセットを公開する場合は
TDSServerModuleを選択します。

補足資料

- サーバアプリケーションの開発手順6
  ソースの保存先を選択

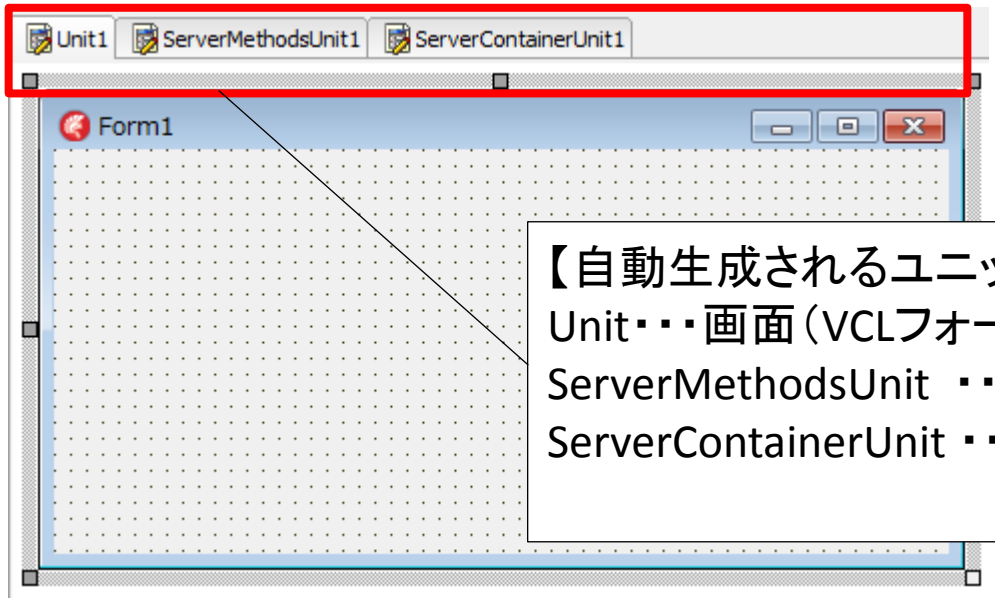


プロジェクトソースの保存場所を指定します。

補足資料

- サーバアプリケーションの開発手順7

  選択した構成によってモジュールが生成

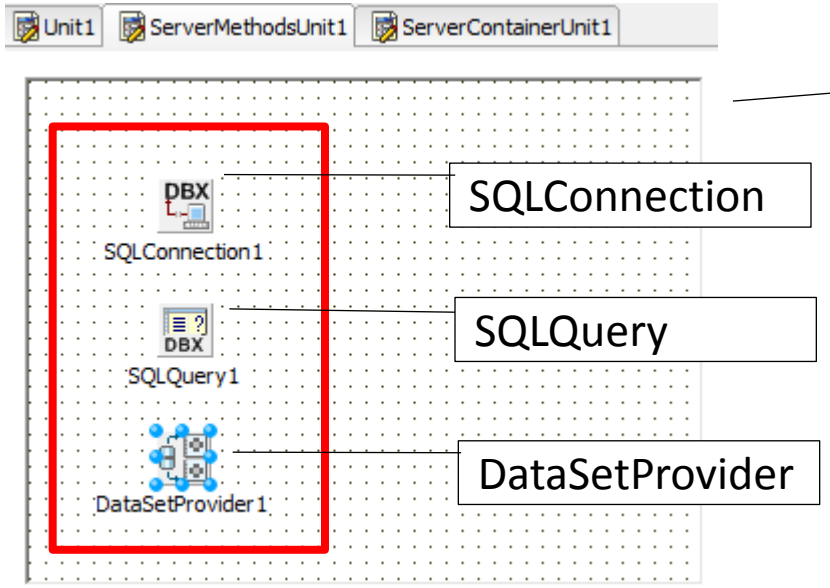


【自動生成されるユニット】
Unit・・・画面（VCLフォームアプリケーション時のみ）
ServerMethodsUnit ・・・サーバが提供する機能
ServerContainerUnit ・・・通信を制御する機能

補足資料

- サーバアプリケーションの開発手順8

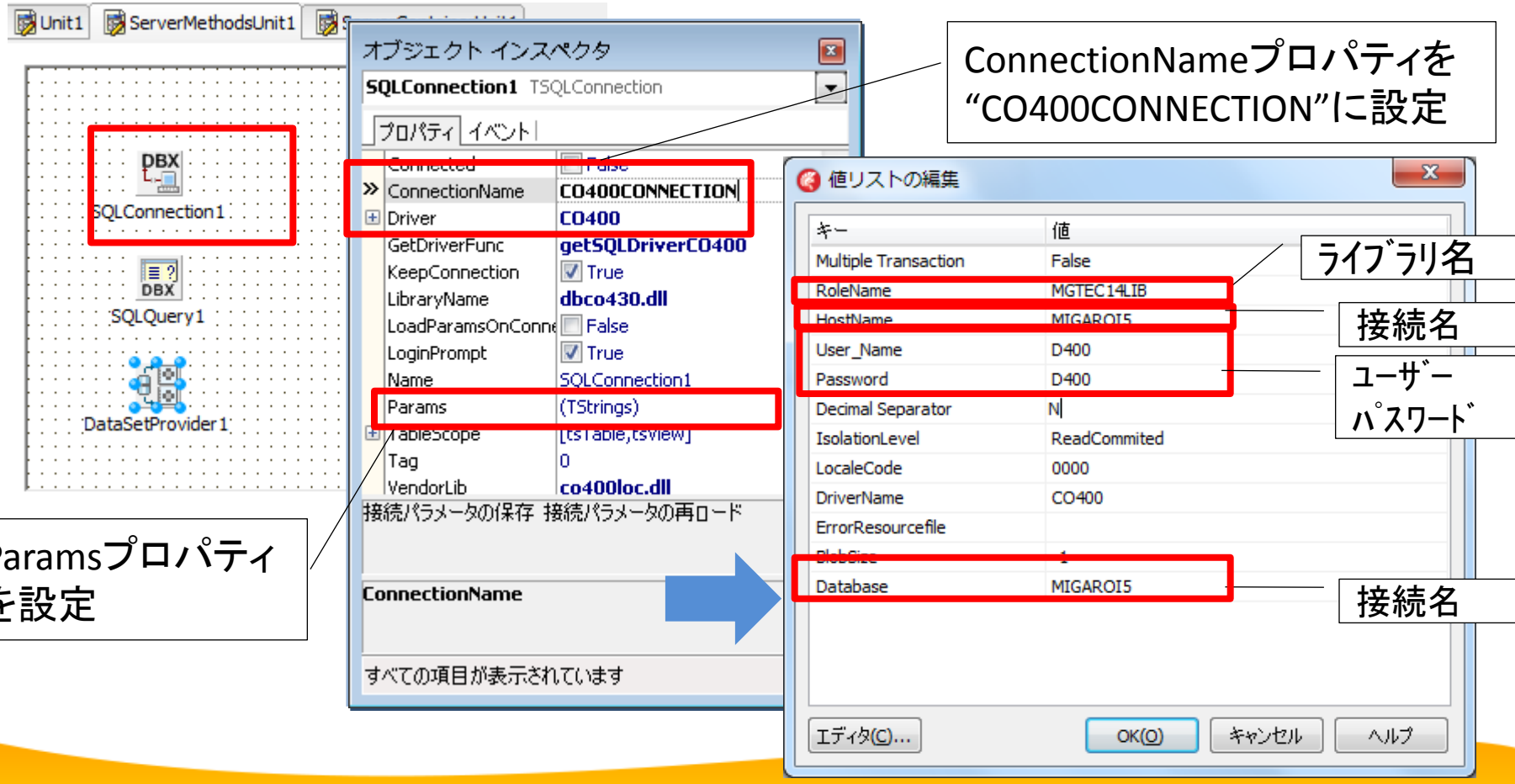
  ServerMethodsに次のコンポーネントを配置
  SQLConnection、SQLQuery、DataSetProvider



今回はIBM i へSQL実行する
サーバアプリケーションの機能だけ
実装しています。

補足資料

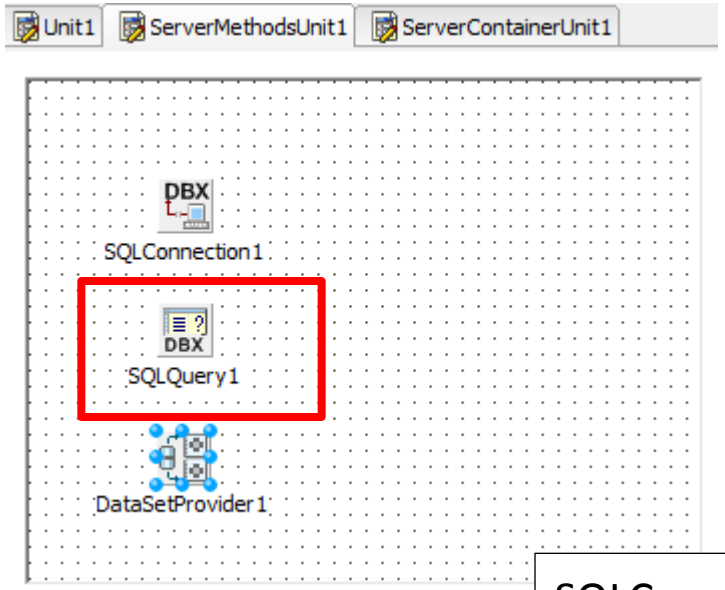
- サーバアプリケーションの開発手順9
  SQLConnectionコンポーネントの設定



補足資料

- サーバアプリケーションの開発手順10

  SQLQueryコンポーネントの設定

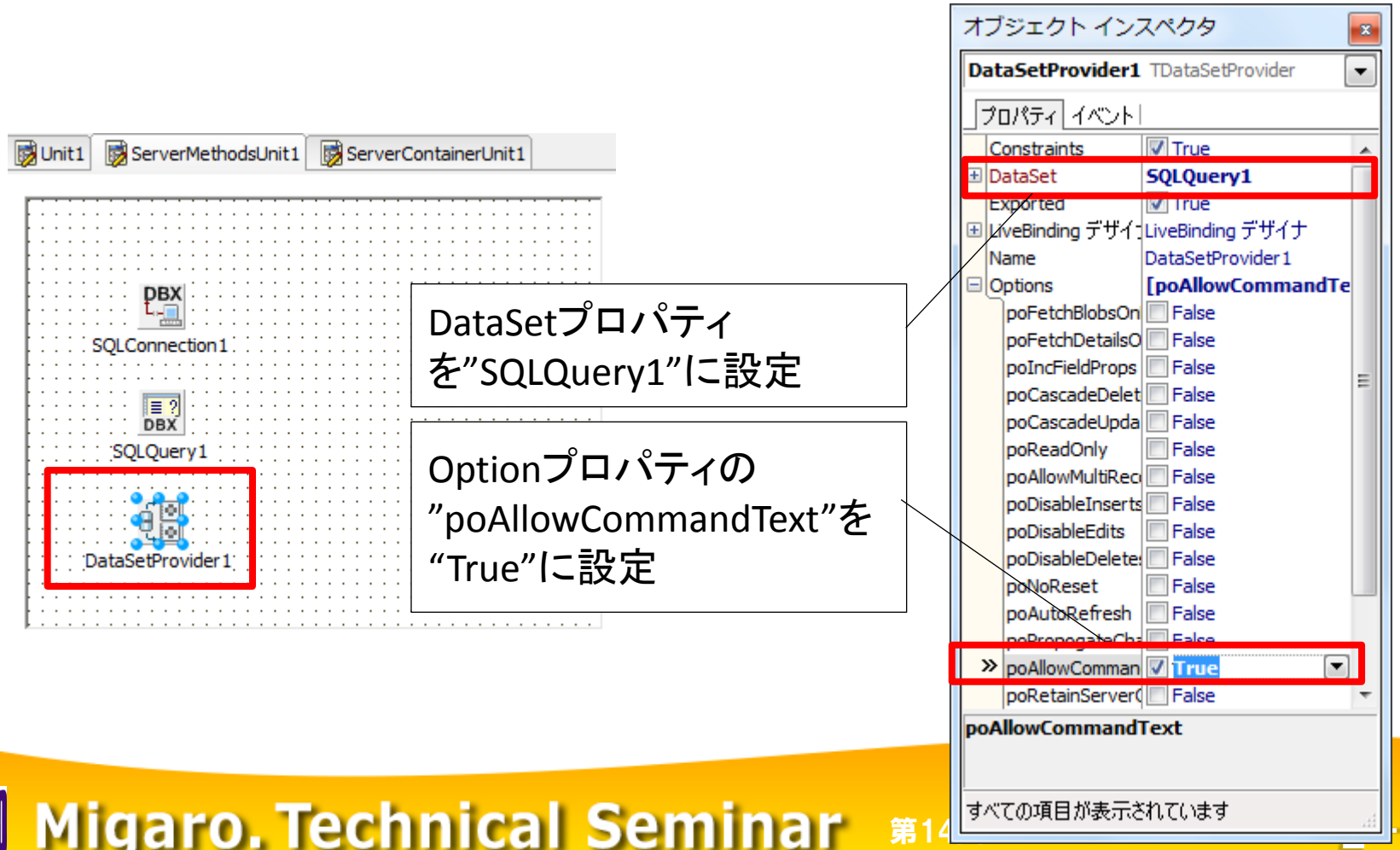


SQLConnectionプロパティ
を” SQLConnection1”に設定

補足資料

- サーバアプリケーションの開発手順11

  DataSetProviderコンポーネントの設定



補足資料

- サーバアプリケーションの開発手順12
  完成したらコンパイルして、アプリケーションサーバ上で起動