如何使用Delphi結合分散式和移動式開發

Gordon Li

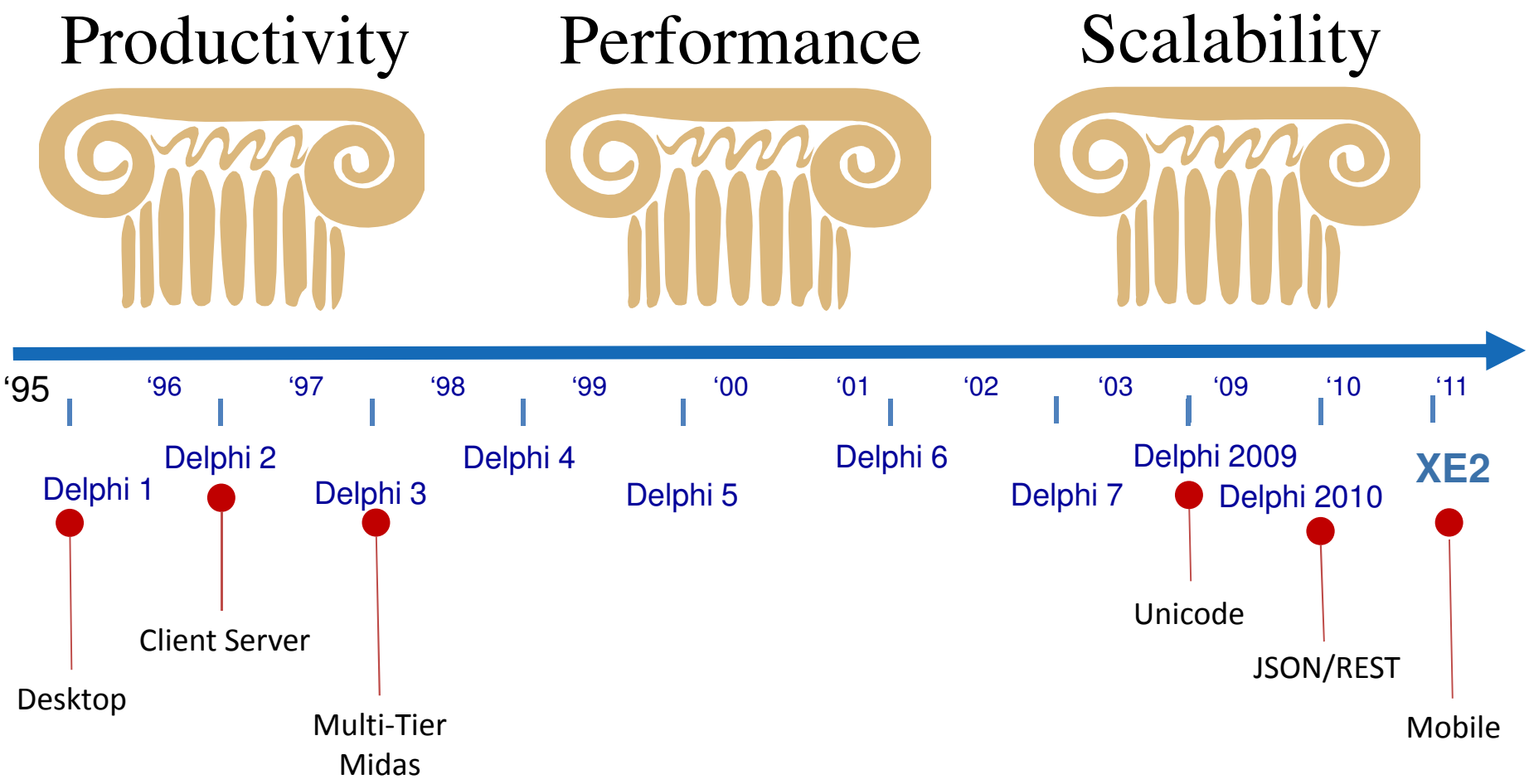
5/17 成都, 5/19 上海
5/29 台北, 5/30 台中, 5/31 高雄
6/5 雅加達, 6/7 泗水

Agenda

- DataSnap XE2
- Mobile Connectors
- 結合分散式和移動式開發
  - XE2 Update 4
- Q&A

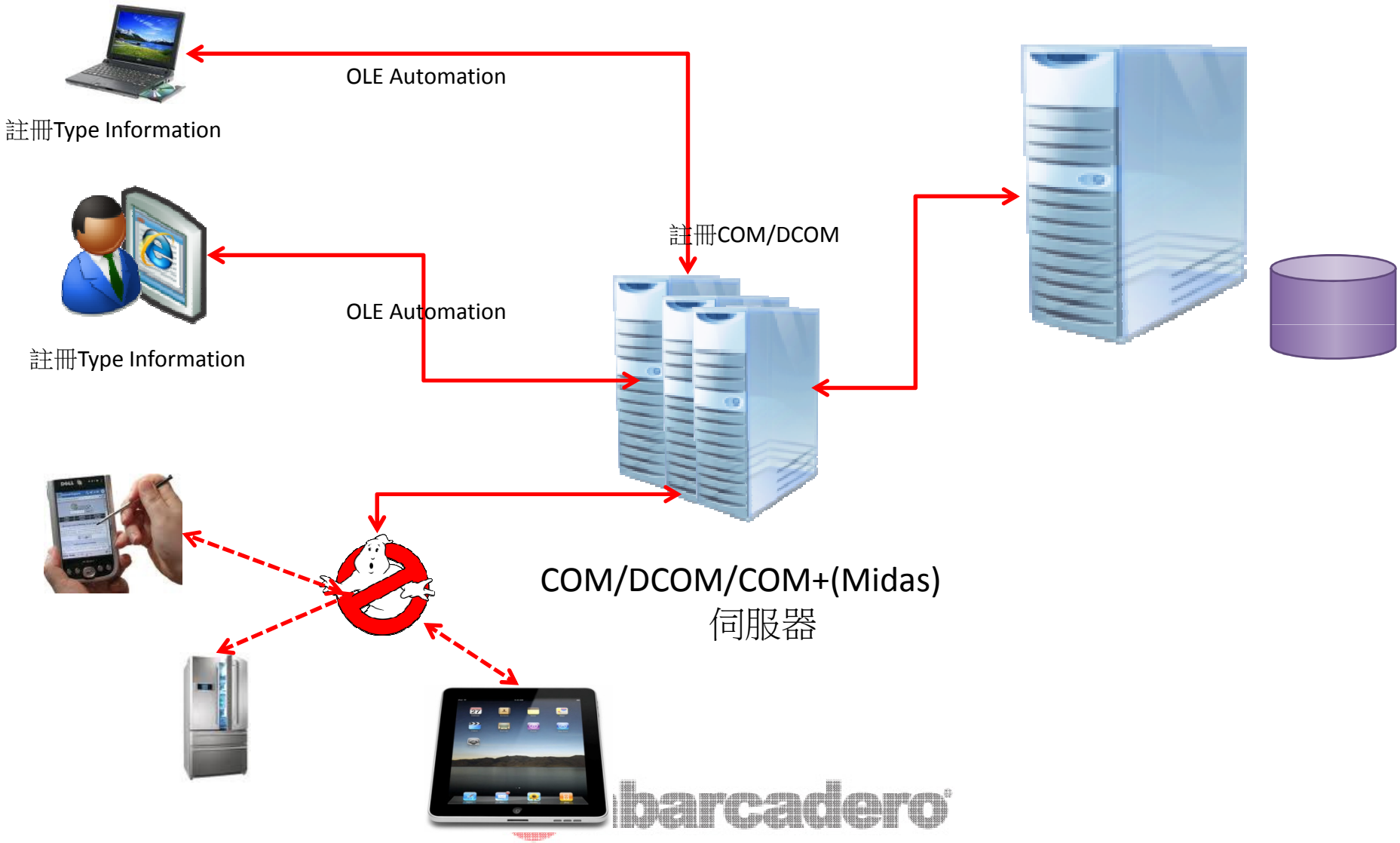
DataSnap XE2

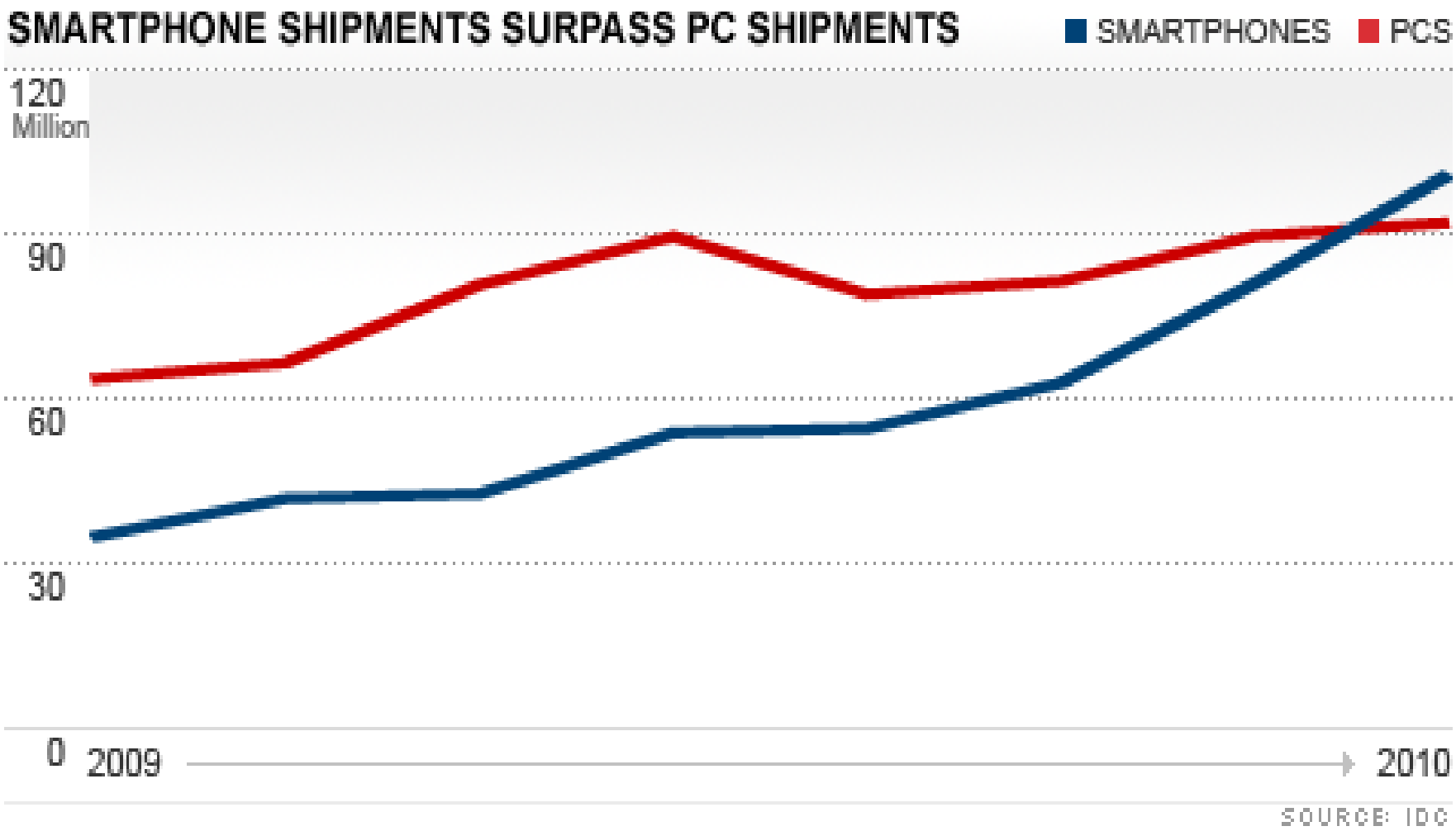
Delphi Heritage



embarcadero

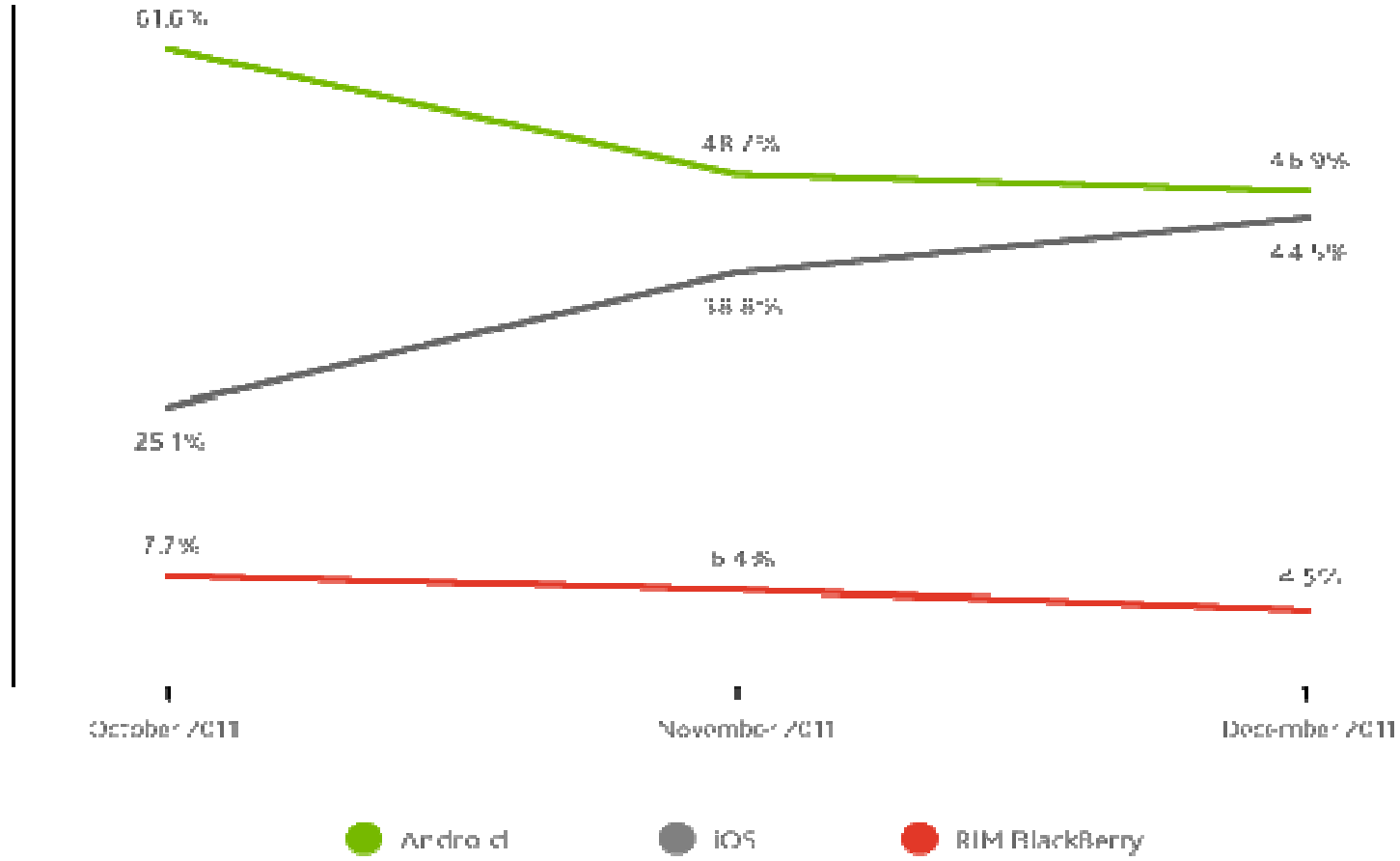
分散式系統(回到COM/DCOM的時代)





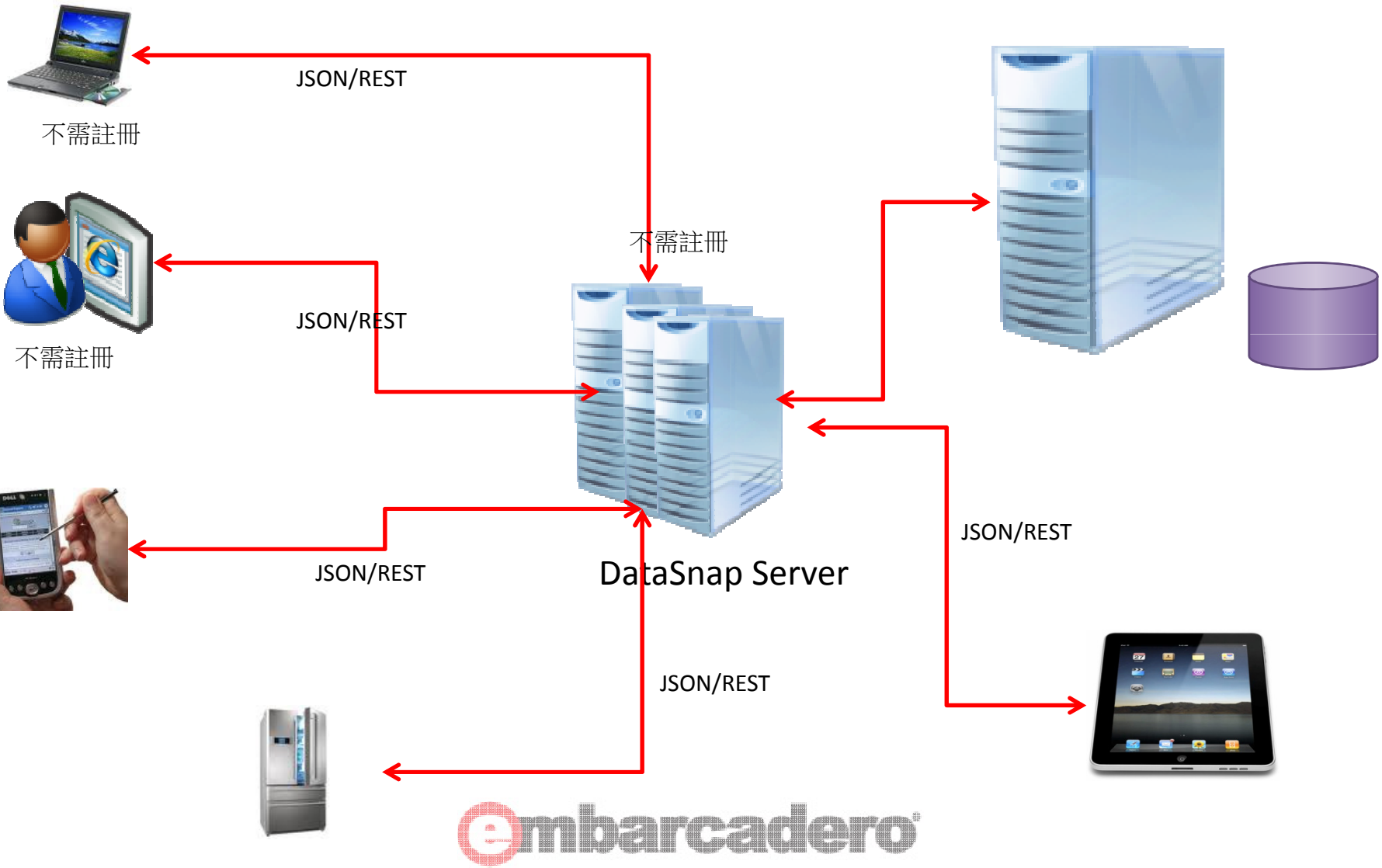
Smartphone Operating System Share – Recent Smartphone Acquirers

Oct - Dec 2011, Nielsen Mobile Insights



Source: Nielsen

分散式系統


JSON/REST
不需註冊
不需註冊
JSON/REST
不需註冊
JSON/REST
DataSnap Server
JSON/REST
JSON/REST
embarcadero

DataSnap XE2

- 從Midas/DataSnap到DataSnap XE2
  - Protocol ：COM/DCOM/COM+ ---→ TCP/HTTP
  - 呼叫方式 : Private call ---→ REST
  - 資料封裝 : OLE Automation ---→ JSON
  - 平台 : Windows Only ---→ 跨平台
  - 便利度 ：複雜,困難 ---→ 簡易
  - 執行效率 : 二位元資料封裝 ---→ 字串
  - ?

- DataSnap XE2保留IAppServer介面
  - 可把Midas系統昇級到DataSnap XE2

從Midas/DataSnap到DataSnap XE2

- TRemoteDatModule(COM/DCOM/COM+)
  - {$M+}
  - IAppServer = interface(IDispatch)
  - ['{1AEFCC20-7A24-11D2-98B0-C69BEB4B5B6D}']
  - function AS_ApplyUpdates(const ProviderName: WideString; Delta: OleVariant;
  - MaxErrors: Integer; out ErrorCount: Integer; var OwnerData: OleVariant): OleVariant; safecall;
  - function AS_GetRecords(const ProviderName: WideString; Count: Integer; out RecsOut: Integer;
  - Options: Integer; const CommandText: WideString;
  - var Params: OleVariant; var OwnerData: OleVariant): OleVariant; safecall;
  - function AS_DataRequest(const ProviderName: WideString; Data: OleVariant): OleVariant; safecall;
  - function AS_GetProviderNames: OleVariant; safecall;
  - function AS_GetParams(const ProviderName: WideString; var OwnerData: OleVariant): OleVariant; safecall;
  - function AS_RowRequest(const ProviderName: WideString; Row: OleVariant; RequestType: Integer;
  - var OwnerData: OleVariant): OleVariant; safecall;
  - procedure AS_Execute(const ProviderName: WideString; const CommandText: WideString;
  - var Params: OleVariant; var OwnerData: OleVariant); safecall;
  - end;
  - {$M-}

從Midas/DataSnap到DataSnap XE2

```
TDSProviderDataModuleAdapter = class(TDSAdapterClass) //DataSnap XE2
 private
  FProviderDataModule: TProviderDataModule;
 public
  constructor Create(AdapteeInstance: TObject); override;
  destructor Destroy; override;
  // IAppServerFastCall interface methods.
  //
  function  AS_ApplyUpdates(const ProviderName: WideString; DeltaStream: OleVariant;
    MaxErrors: Integer; out ErrorCount: Integer; OwnerDataStream: TDBXStreamValue): OleVariant;
  function  AS_GetRecords(const ProviderName: WideString; Count: Integer;
    out RecsOut: Integer; Options: Integer; const CommandText: WideString;
    ParamReader: TDBXStreamValue; OwnerDataStream: TDBXStreamValue): OleVariant;
  function  AS_DataRequest(const ProviderName: WideString;
    DataStream: OleVariant): OleVariant;
  function  AS_GetProviderNames: string;
  function  AS_GetParams(const ProviderName: WideString;
    OwnerDataStream: TDBXStreamValue): OleVariant;
  function  AS_RowRequest(const ProviderName: WideString; RowStream: OleVariant;
    RequestType: Integer; OwnerDataStream: TDBXStreamValue): OleVariant;
  procedure AS_Execute(const ProviderName, CommandText: WideString;
    ParamReader: TDBXStreamValue; OwnerDataStream: TDBXStreamValue);

    property ProviderDataModule: TProviderDataModule read FProviderDataModule write FProviderDataModule;
 end;
```

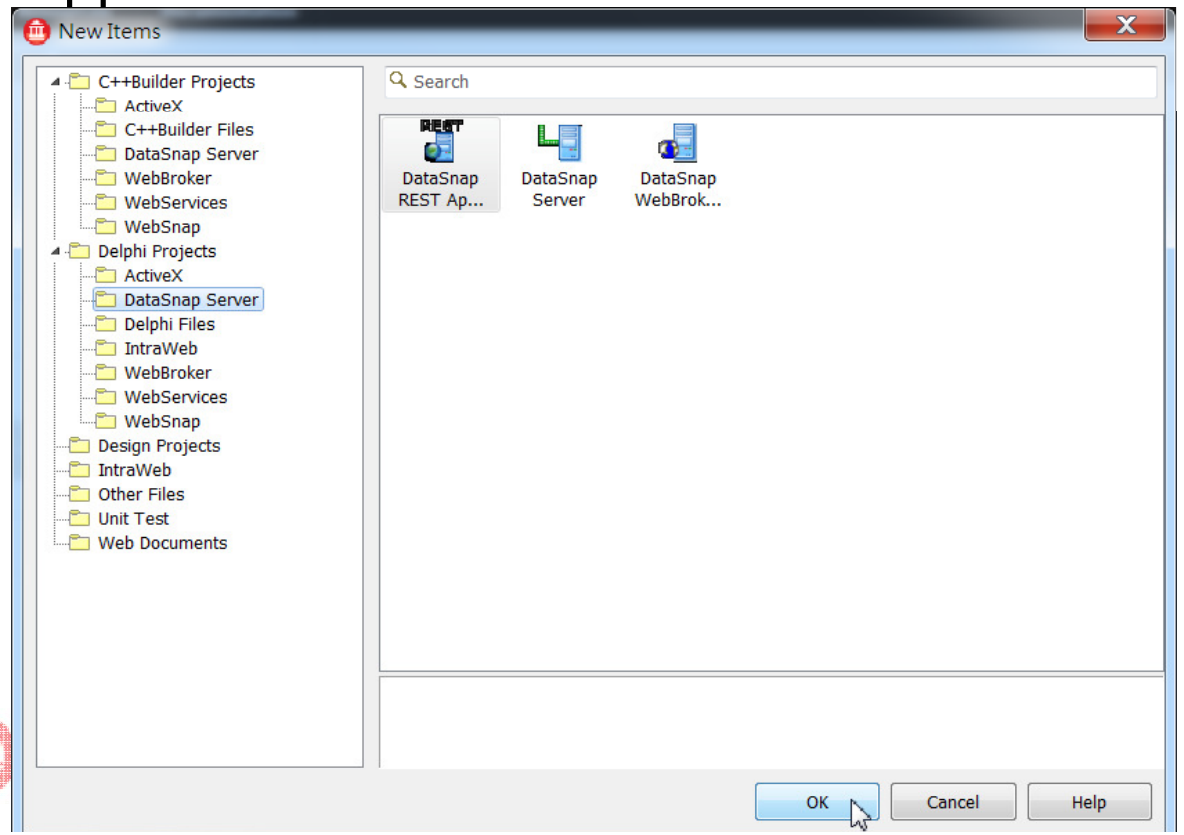
從Midas/DataSnap到DataSnap XE2

```
{$IFDEF MSWINDOWS}
{$MethodInfo ON}
 TRemoteDataModule = class(TProviderDataModule, IAppServer)
 private
  FLock: TObject;
 protected
  class procedure UpdateRegistry(Register: Boolean; const ClassID, ProgID: string); override;
  { IAppServer }
  function AS_GetProviderNames: OleVariant; safecall;
  function AS_ApplyUpdates(const ProviderName: WideString; Delta: OleVariant;
   MaxErrors: Integer; out ErrorCount: Integer;
   var OwnerData: OleVariant): OleVariant; safecall;
  function AS_GetRecords(const ProviderName: WideString; Count: Integer;
   out RecsOut: Integer; Options: Integer; const CommandText: WideString;
   var Params, OwnerData: OleVariant): OleVariant; safecall;
  function AS_DataRequest(const ProviderName: WideString;
   Data: OleVariant): OleVariant; safecall;
  function AS_GetParams(const ProviderName: WideString; var OwnerData: OleVariant): OleVariant; safecall;
  function AS_RowRequest(const ProviderName: WideString; Row: OleVariant;
   RequestType: Integer; var OwnerData: OleVariant): OleVariant; safecall;
  procedure AS_Execute(const ProviderName: WideString;
   const CommandText: WideString; var Params, OwnerData: OleVariant); safecall;

 public
  constructor Create(AOwner: TComponent); override;
  destructor Destroy; override;
  procedure Lock; virtual;
  procedure Unlock; virtual;
```

DataSnap XE2

- XE2提供3種DataSnap伺服器
  - DataSnap REST Application
  - DataSnap Server
  - DataSnap WebBroker Application



DataSnap XE2

DataSnap Server型態	說明
DataSnap REST Application	為DataSnap伺機器加入REST的功能, 可執行在Web Server中
DataSnap Server	DataSnap Server 是提供一般應用程式形態的DataSnap伺機器, 最適合Midas系統昇級使用
DataSnap WebBroker Application	是為WebBroker技術加入DataSnap功能, 主要目的應該是為了把早期WebBroker或是Internet Express應用程式昇級成DataSnap伺機器

DataSnap XE2

DataSnap伺服器生命週期	說明
Server	在整個DataSnap伺服器中只會建立一個服務類別物件以服務所有的用戶端，只有當DataSnap伺服器結束時才會釋放此服務類別物件
Session	在DataSnap伺服器中會為每一個連結的用戶端建立一個專屬的服務類別物件服務此用戶端，一旦用戶端結束或是關閉TSQLConnection的連結，此服務類別物件便會被釋放
Invocation	在DataSnap伺服器中每當用戶端執行一次請求時，在DataSnap伺服器便會為這個請求建立一個服務類別物件服務此用戶端請求，當請求執行結束後，DataSnap伺服器便會釋放此服務類別物件

DataSnap XE2

生命週期	說明
Server	提供所有用戶端公用的服務，由於所有用戶端都使用單一的服端服務類別物件，因此使用這種生命週期的服務物件負荷都比較大，使用這種生命週期的服務物件適合提供快速，簡單，無狀態的服務為主。
Session	由於這種生命週期型態的服端服務類別物件為會每一個用戶端的連結建立一個專屬的服務物件，因此可提供用戶端無狀態以及有狀態的服務，也可提供長期，負荷較大的服務。
Invocation	這種生命週期型態的服端服務類別物件只存在於每一個用戶端的呼叫周期，因此適合提供可在背景執行的服務，或是執行資料庫的預儲程序，或是批次處理等和用戶端較無相關的服務。不過由於使用這種生命週期的服務類別會被頻繁的建立和釋放，因此這種服務類別應該儘量精簡，如果需要使用資料庫，那麼也應該搭配使用dbExpress的連結池功能以加快服務速度。

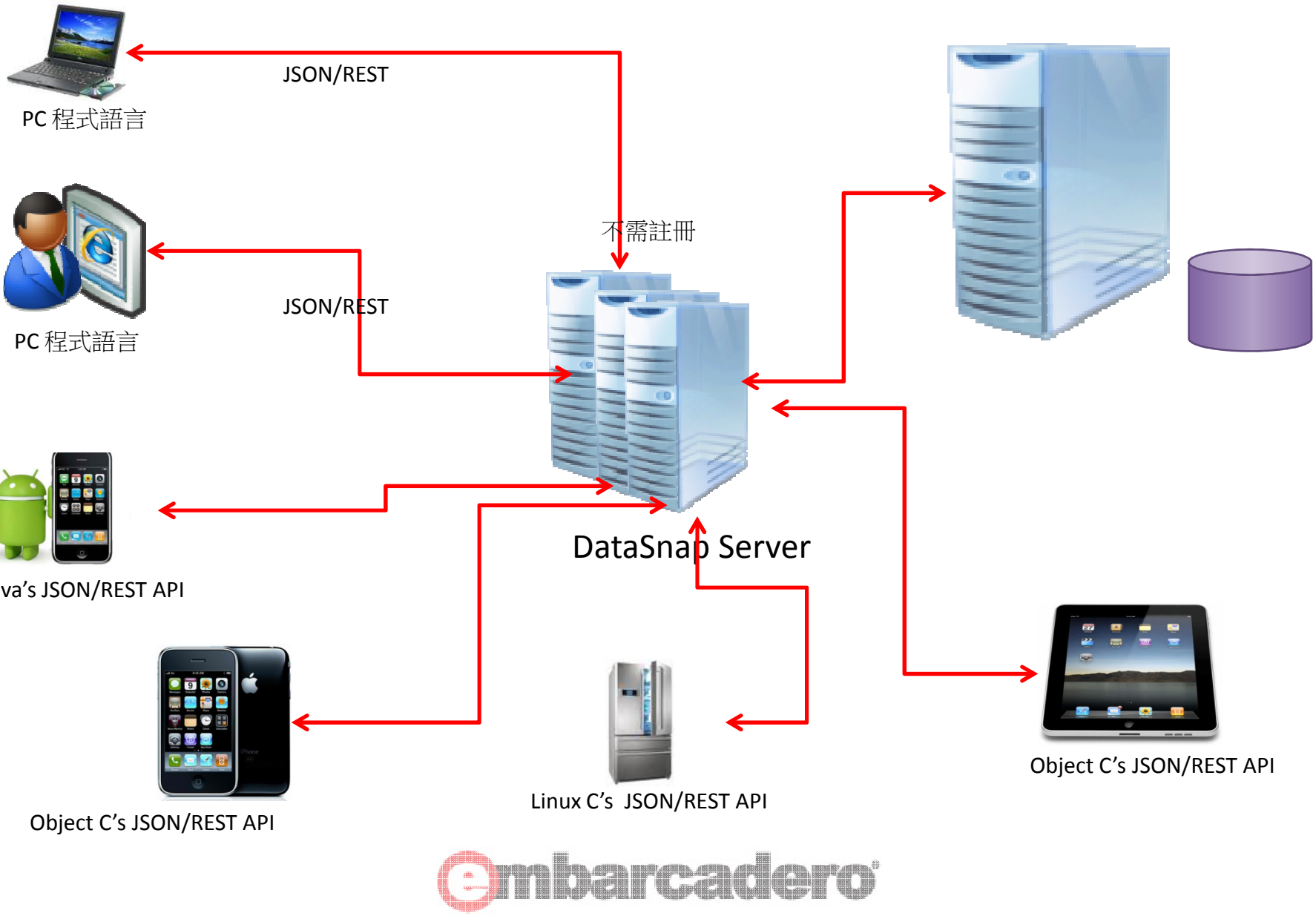
DataSnap XE2

- 範例
  - 執行效率
  - Server生命週期
  - Session生命週期
  - Invocation生命週期
  - 修改Midas伺服器為DataSnap伺服器

DataSnap XE2

- 修改Midas伺服器為DataSnap伺服器步驟
  1. 原有Object Pascal程式碼可保留使用
  2. 刪除專案中和COM/DCOM/COM+相關的程式碼
     - 例如Type library, COM介面, COM相關的程式單元
     - COM/DCOM/COM+註冊程式碼, 初使化程式碼
  3. 把對外輸出的方法從private改為public
  4. 移除safecall呼叫宣告
  5. 加入TDataMdule或其衍生類別的程式單元, 並且在其中加入TDSXXXX相關的元件以使其成為JSON/REST伺服器

現在的分散式系統

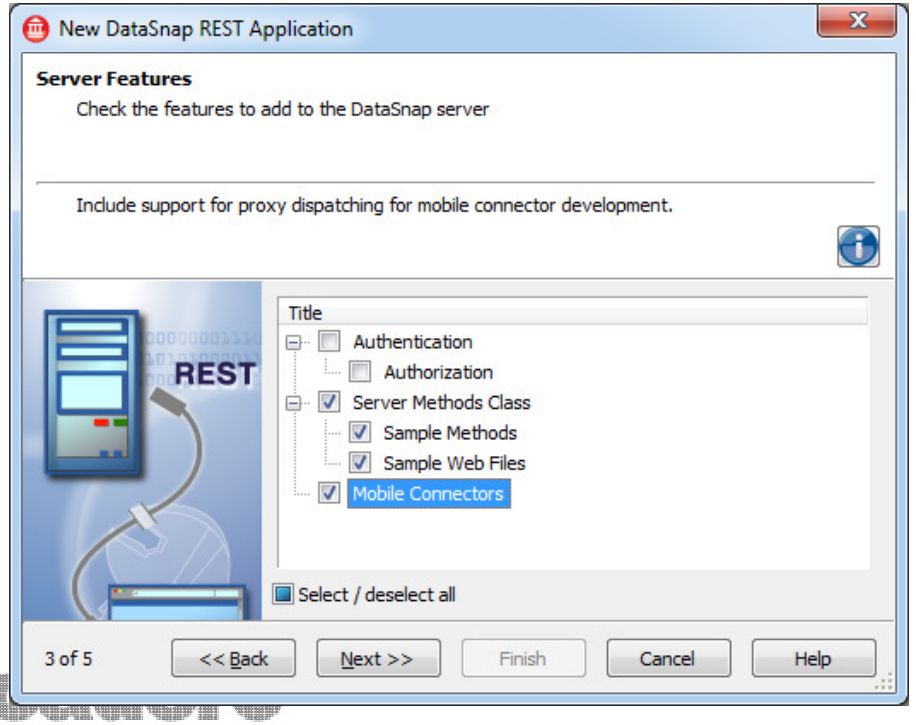

JSON/REST
PC 程式語言
不需註冊
JSON/REST
PC 程式語言
DataSnap Server
Java's JSON/REST API
Object C's JSON/REST API
Linux C's JSON/REST API
Object C's JSON/REST API
embarcadero

DataSnap Mobile Connectors

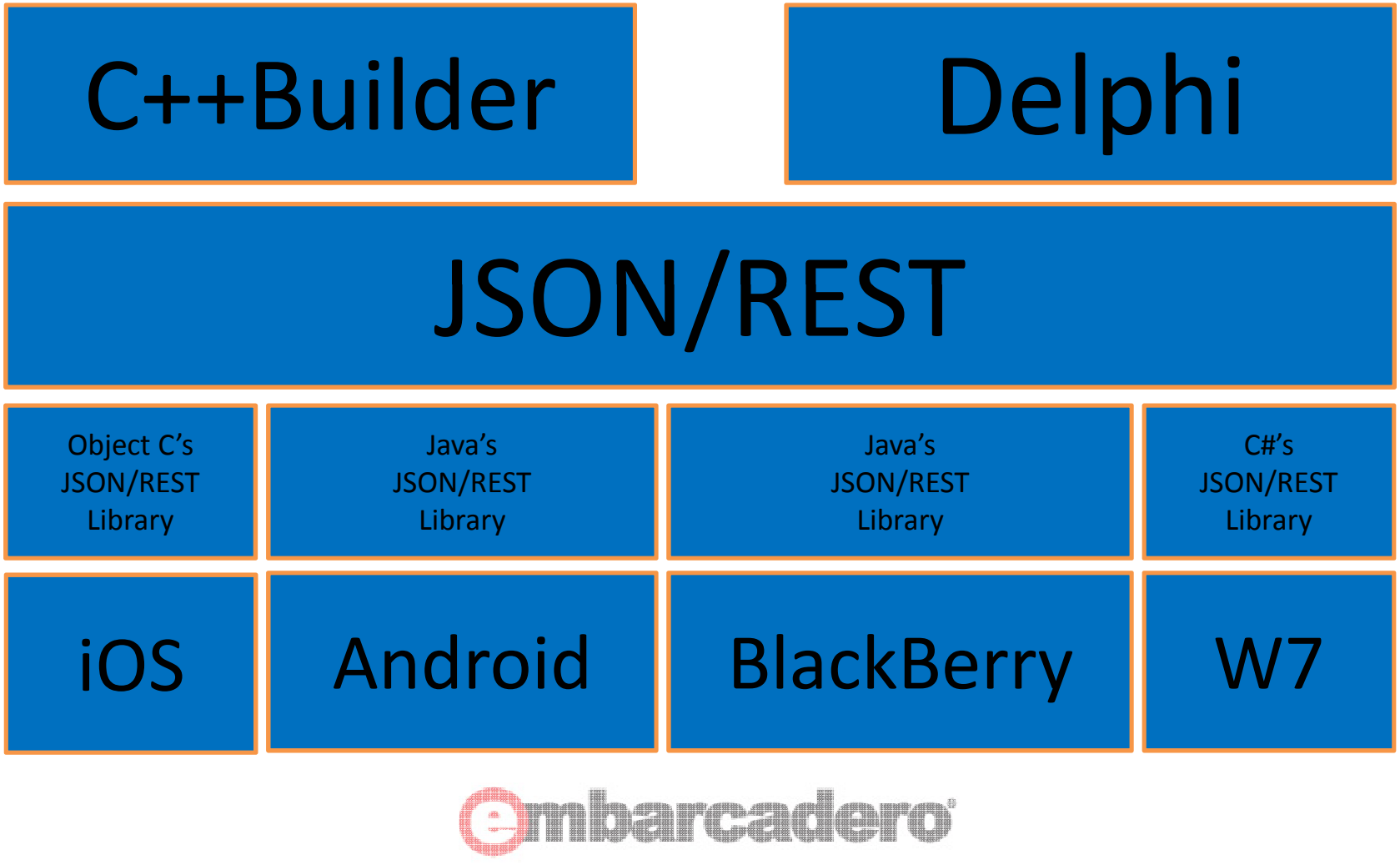
結合分散式和手機系統

Mobile Connectors for DataSnap

- DataSnap Mobile Connectors
  - iOS – Objective C
  - Android – Java
  - BlackBerry – Java
  - Windows Phone 7 – C#,

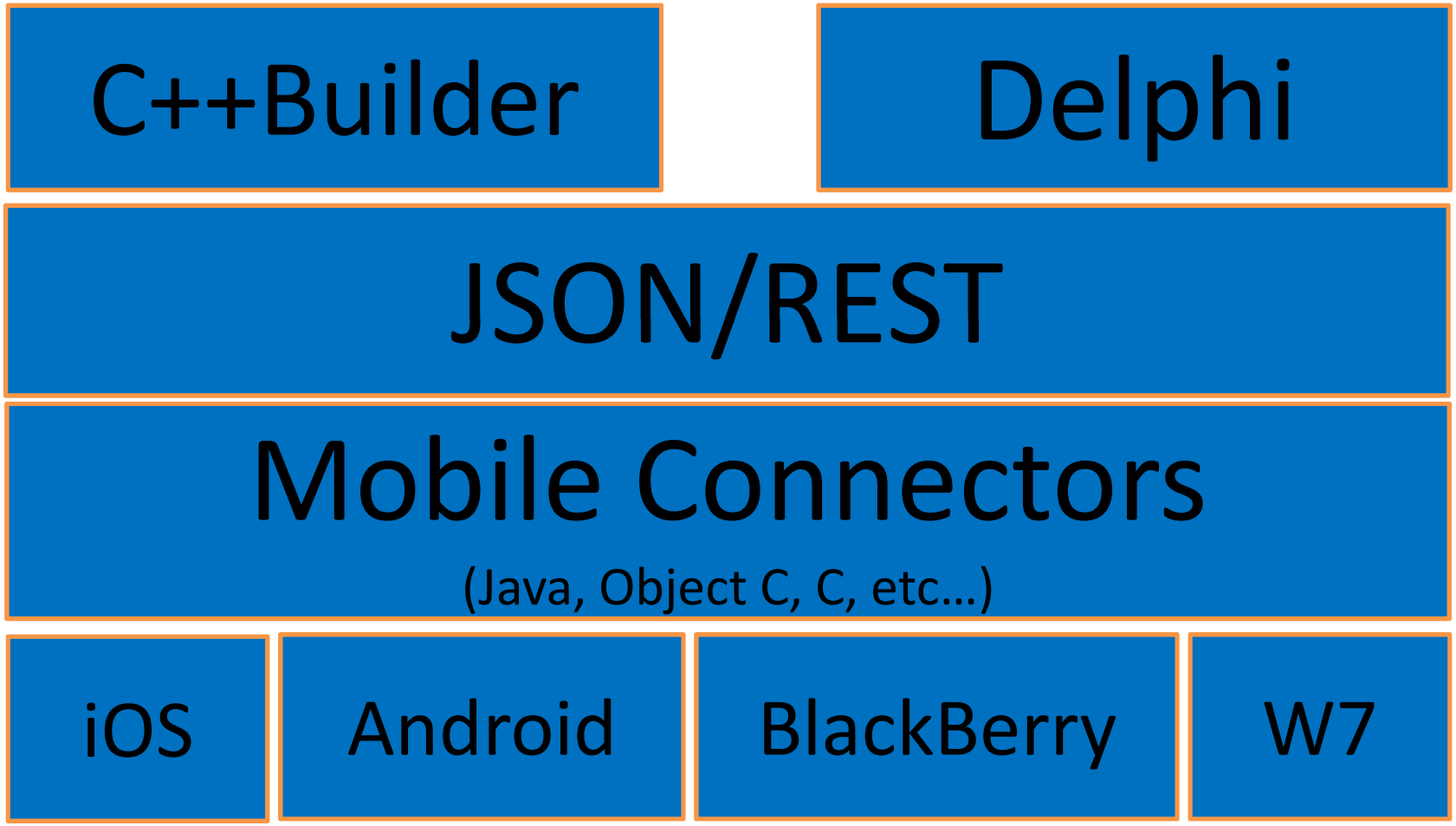


Mobile Connectors for DataSnap

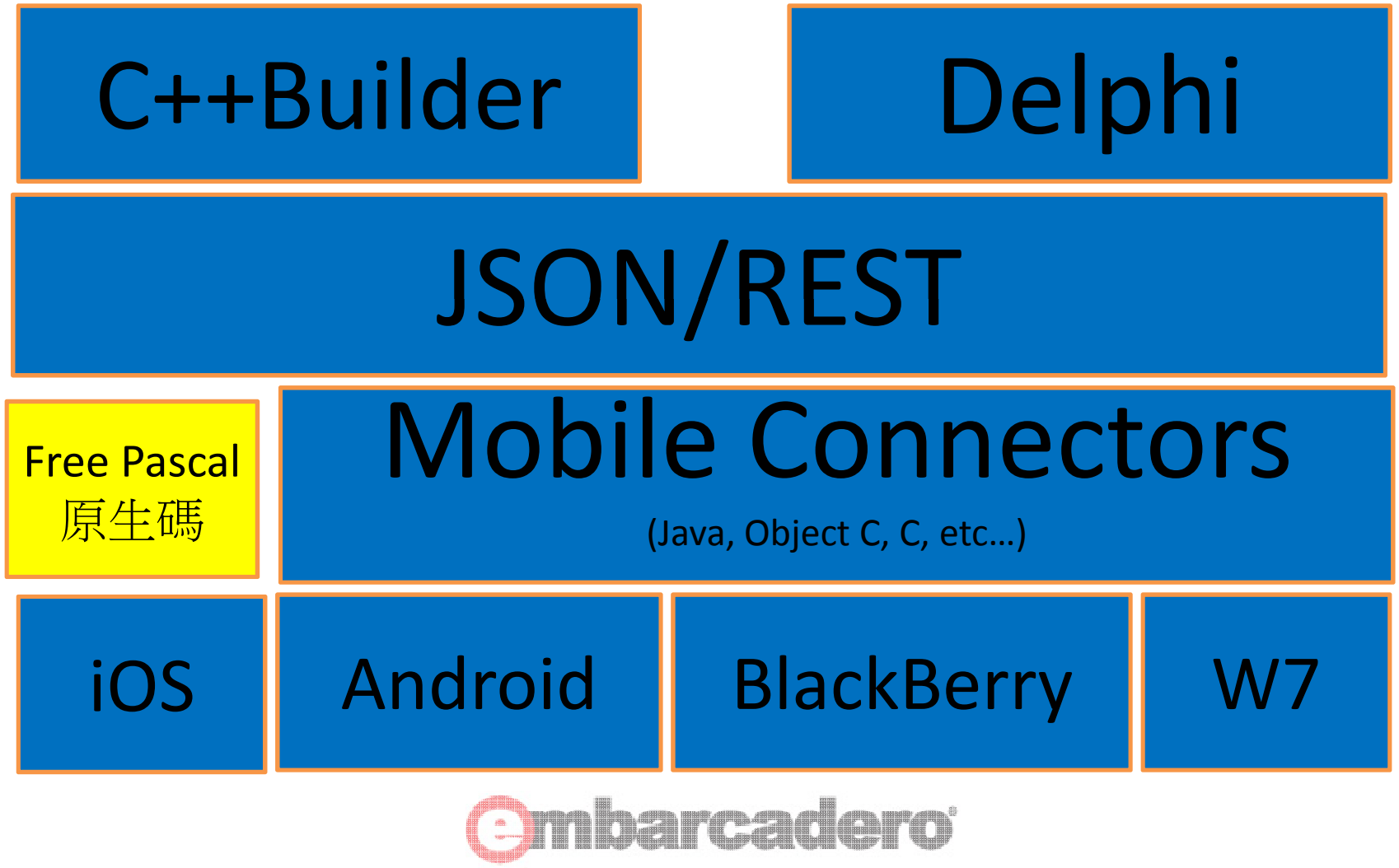


embarcadero

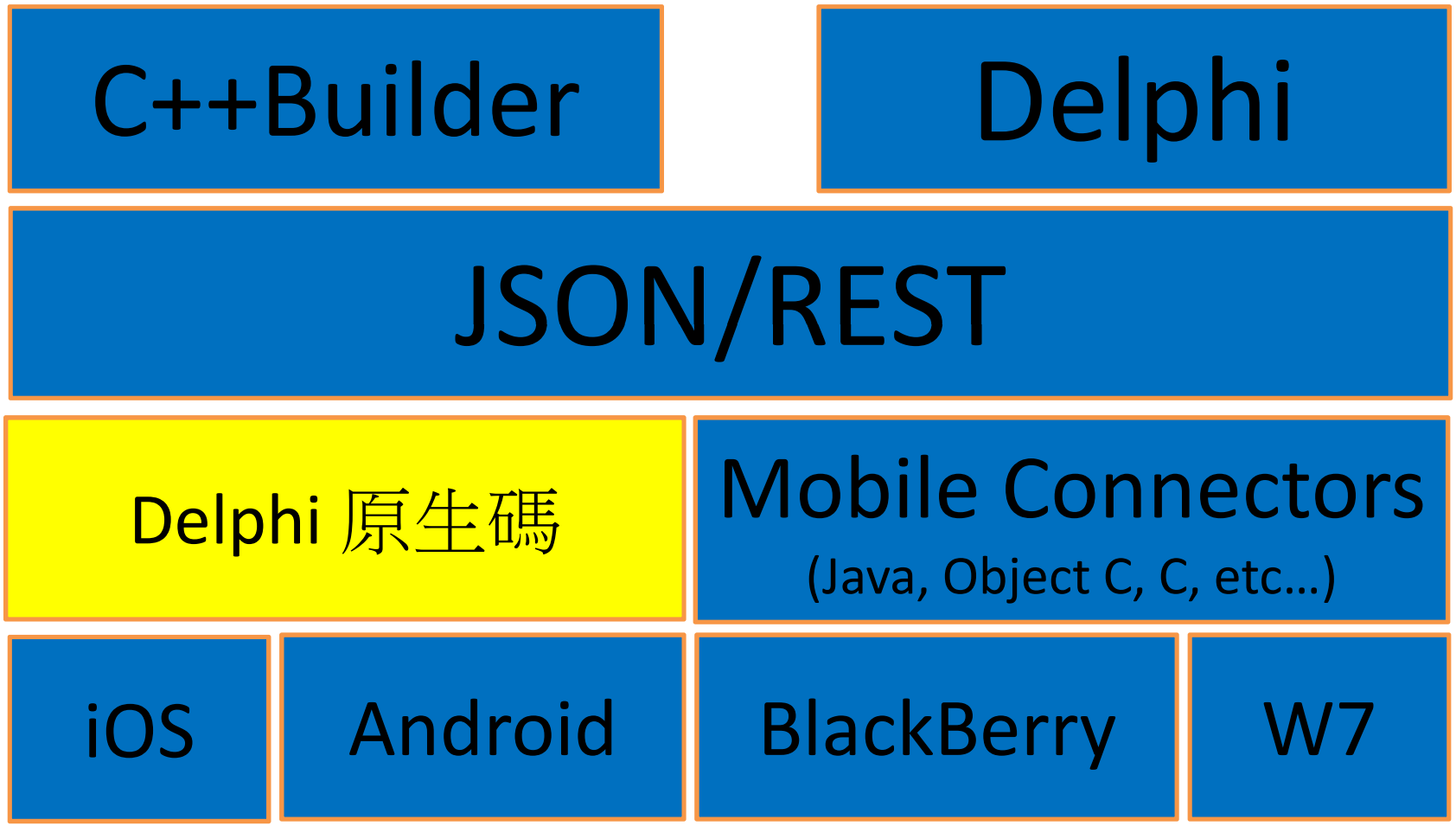
Mobile Connectors for DataSnap



Mobile Connectors for DataSnap(U4)



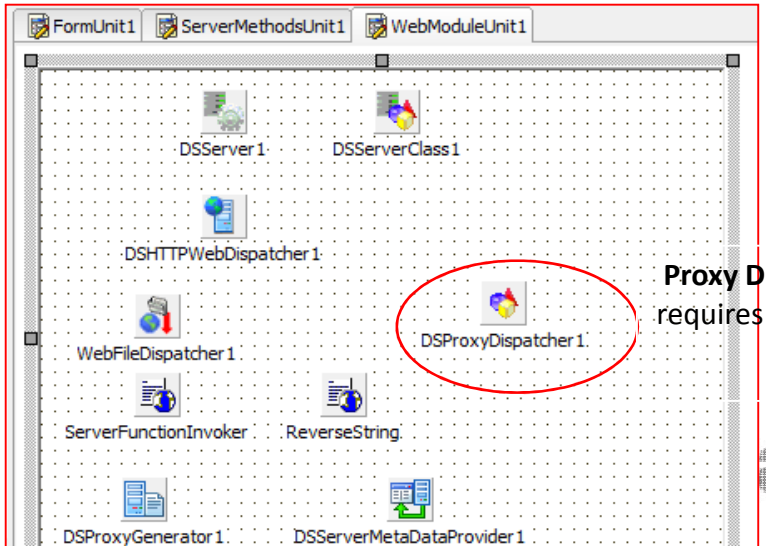
Mobile Connectors for DataSnap(*)



embarcadero

DataSnap Mobile Connectors

- How it works
  - DataSnap Connectors provide a new **Proxy Dispatcher.**
    - **Proxy Dispatcher** works together with the pre-existing proxy generator, metadata provider and server class components.
    - **Proxy Dispatcher** provides functionality which allows for developers on remote machines to generate a proxy on the server and download it onto the developer's machine.



Proxy Dispatcher handles the http requests that requires a specific proxy.

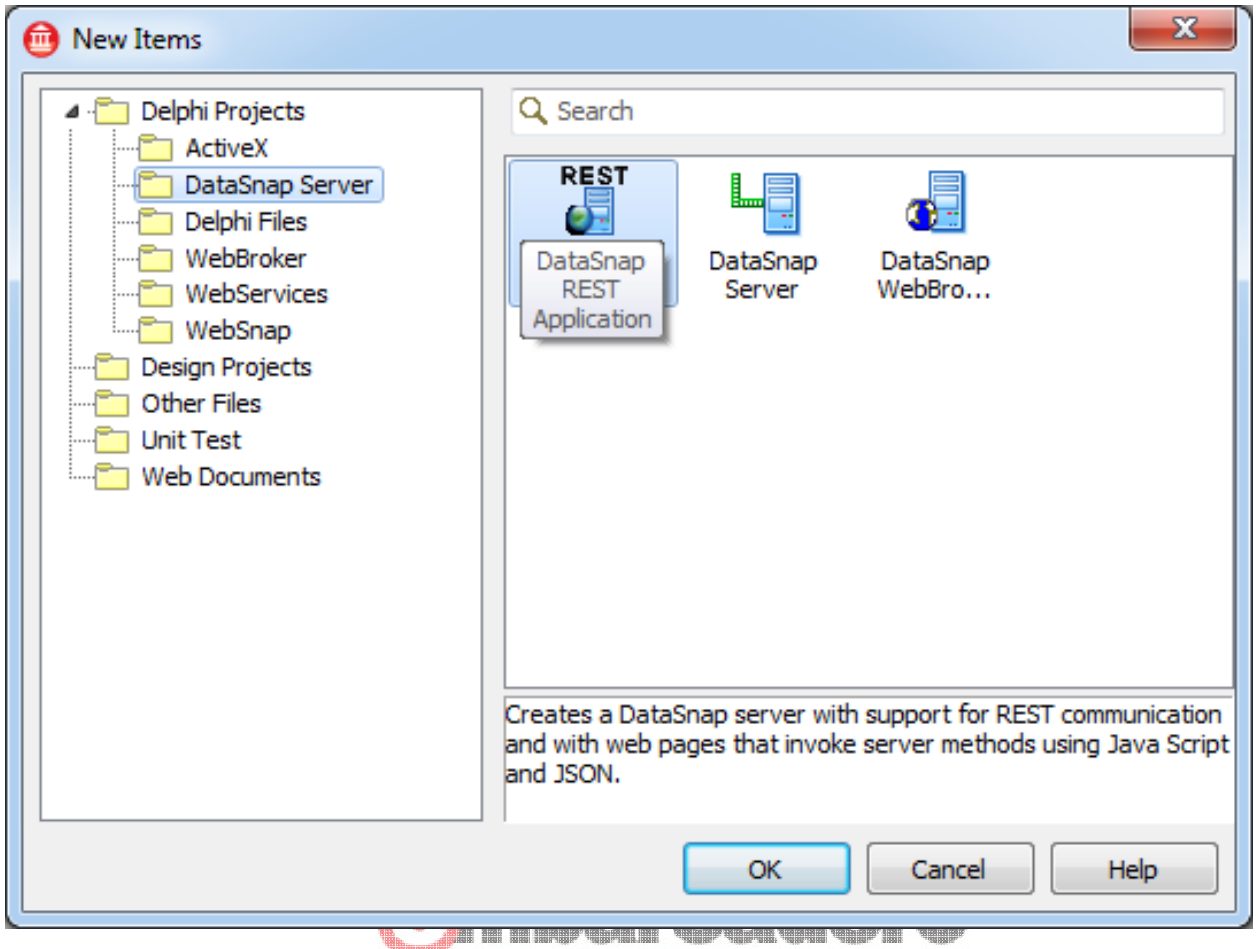
arcadero

DataSnap Mobile Connectors

- To begin mobile development with DataSnap:
  1. Properly setup your DataSnap Server for proxy dispatching, so you can generate the mobile platform's proxy.
     - Once your DataSnap server is running, anyone can manually request the proxy by opening a browser and going to: http://host:port/proxy/[DEVICE].zip
       - csharp_silverlight
       - objective_ios42
       - java_blackberry
       - java_android
     - Alternatively, you can use the "**Win32ProxyDownloader.exe**" to create the proxy source code.
  2. For example, an Android (Java) application needs the **DSProxy.java** file which contains the code generated by the server.

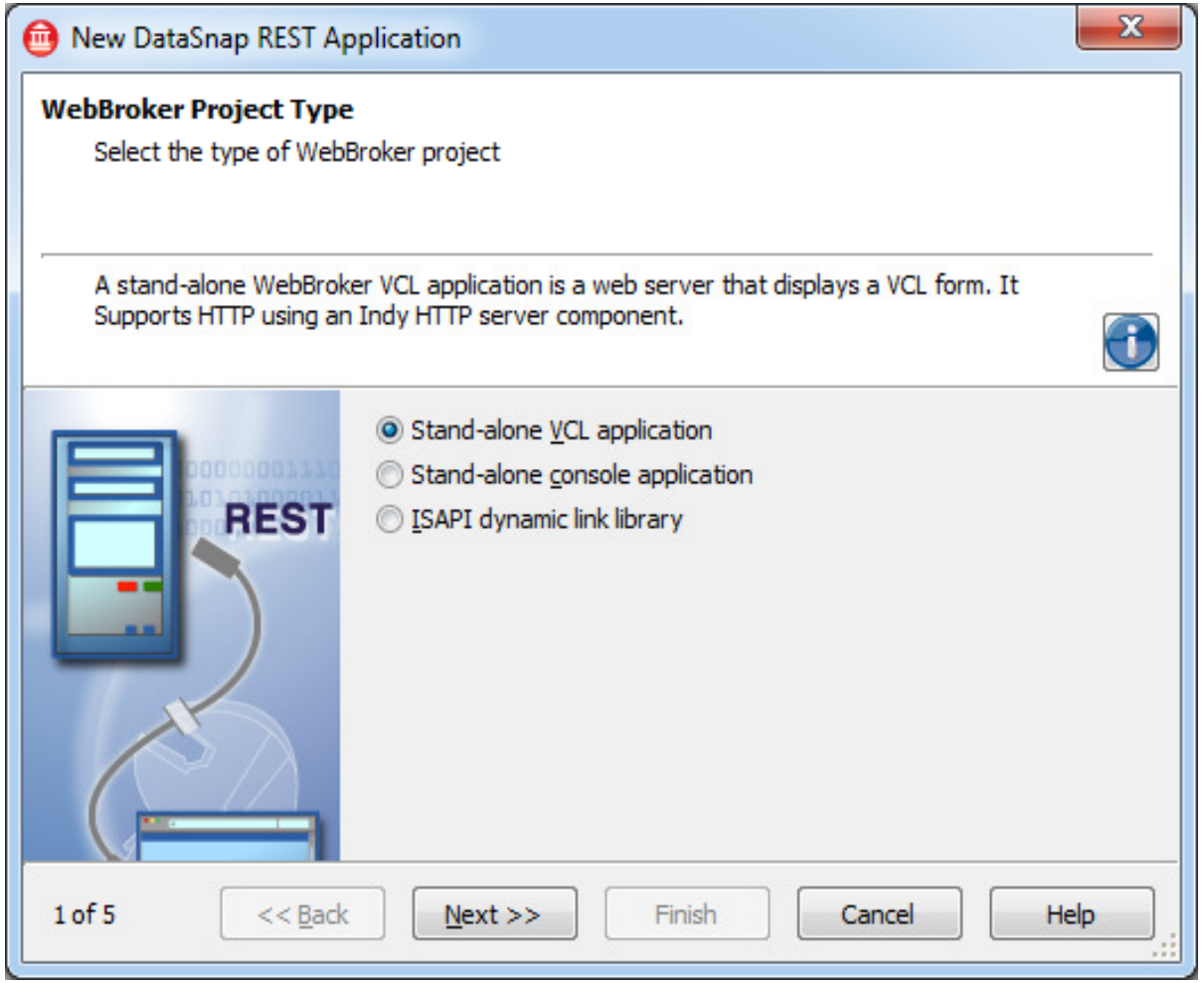
DataSnap Mobile Connectors Step-By-Step Details

Create a DataSnap REST Server using wizard.



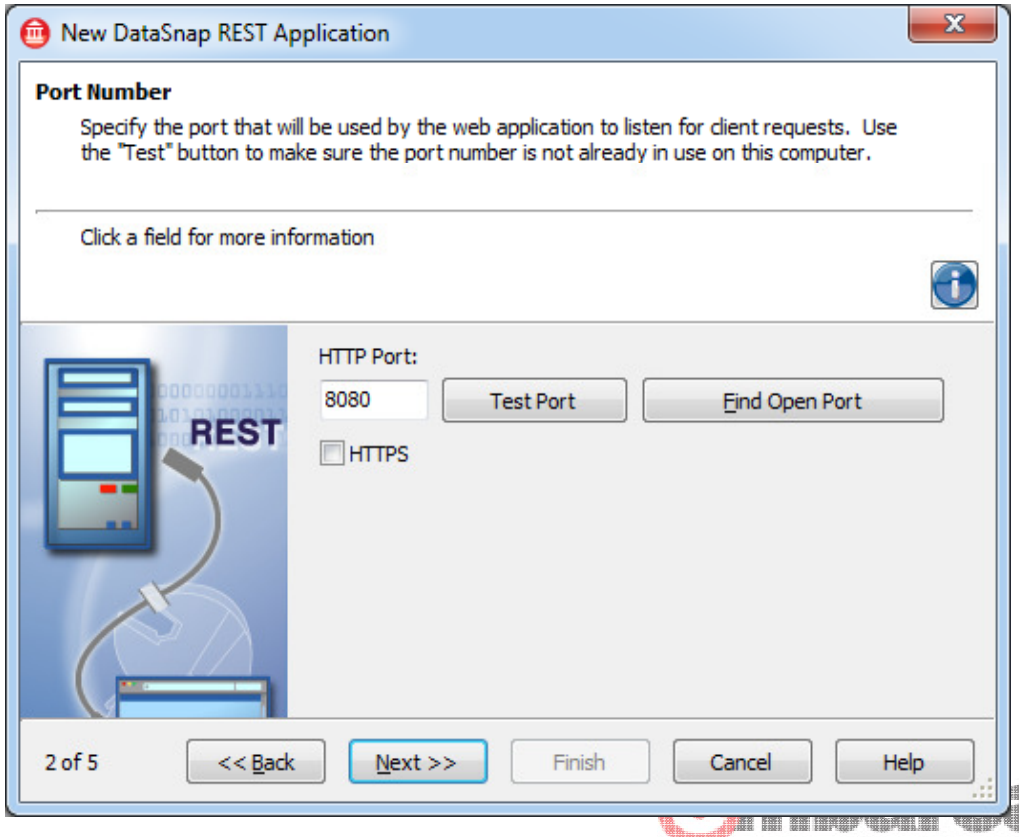
DataSnap Mobile Connectors Step-By-Step Details

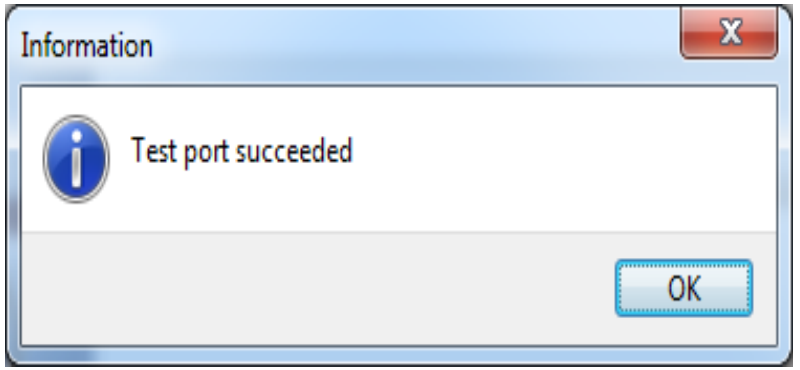
1. Create a Stand-alone VCL Application.



DataSnap Mobile Connectors Step-By-Step Details

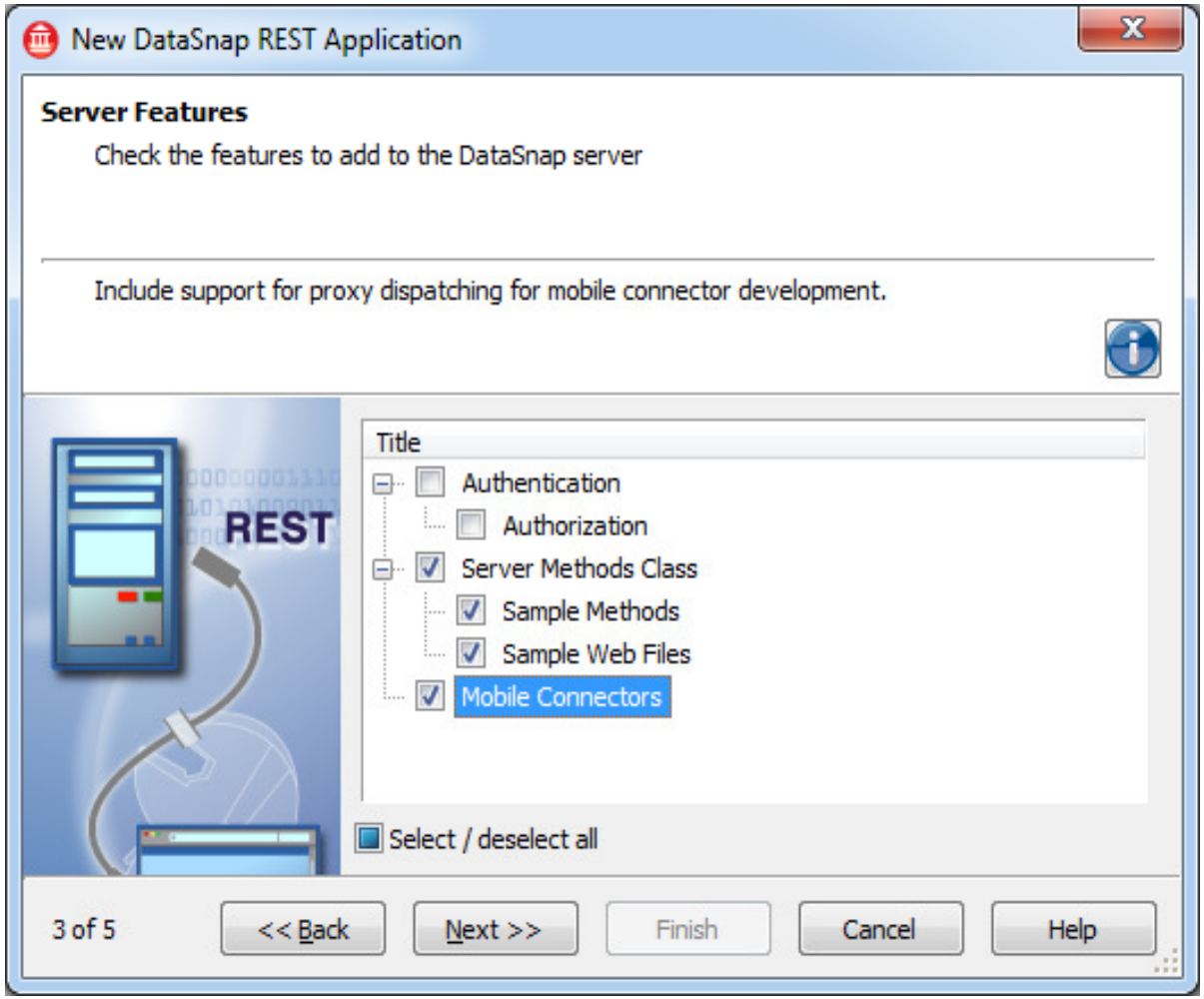
2. Specify the port that will be used by the web application to listen for client requests.





DataSnap Mobile Connectors Step-By-Step Details

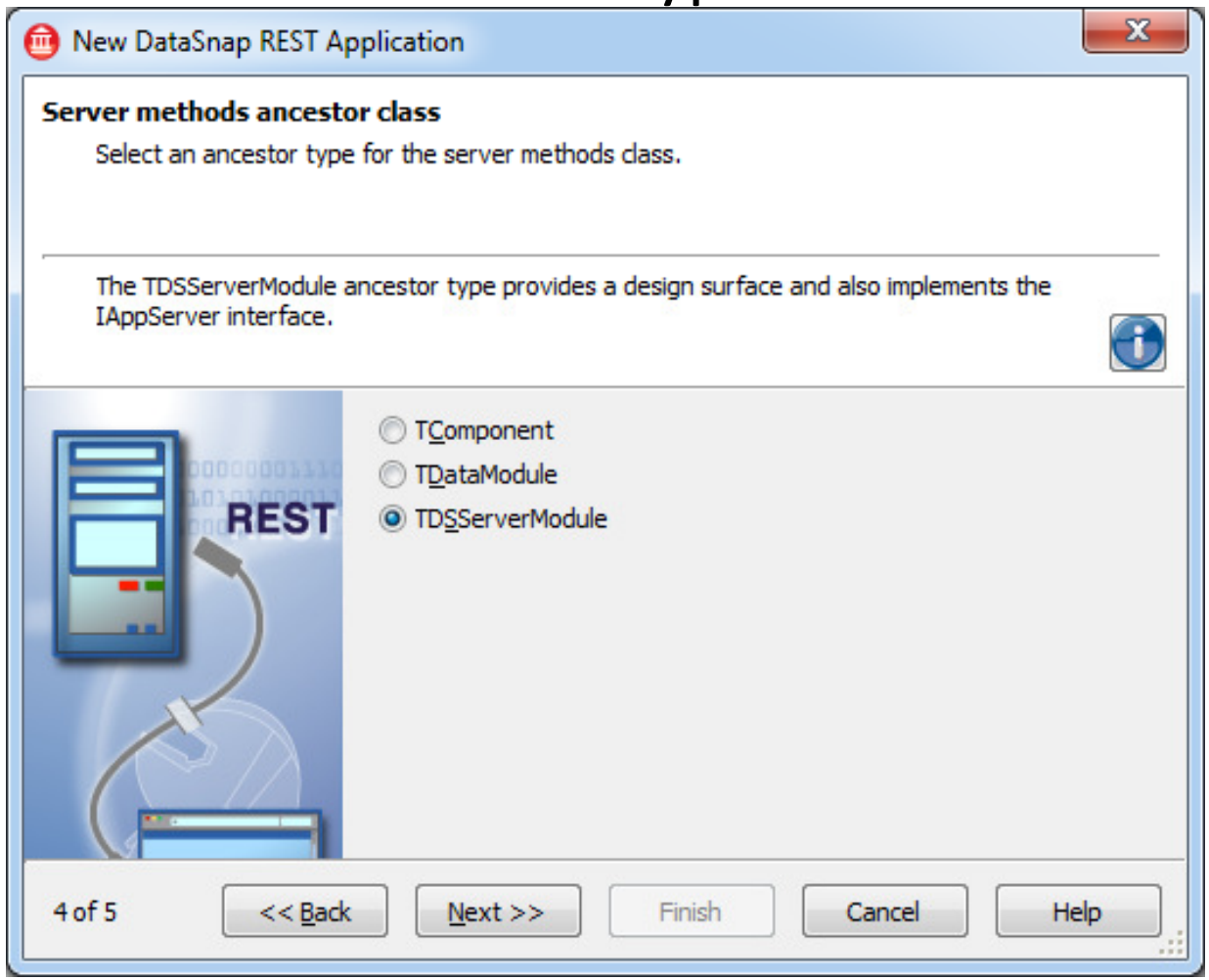
3. Add Mobile Connectors.



“Mobile Connectors" adds support for proxy dispatching for mobile connector development.

DataSnap Mobile Connectors
Step-By-Step Details

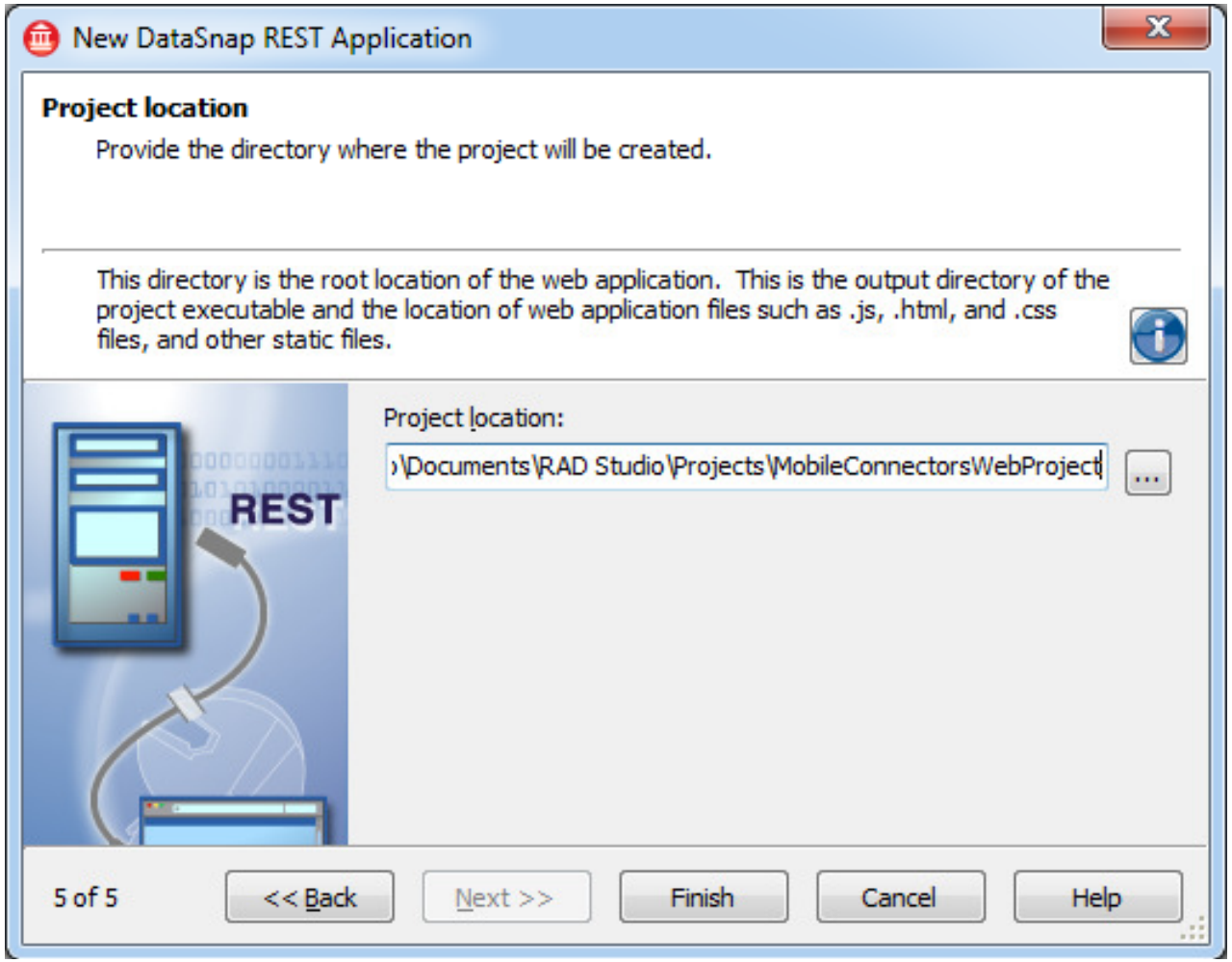
4. Select ancestor type of the server methods class.



TDSServerModule to expose datasets from the server to client applications.

DataSnap Mobile Connectors Step-By-Step Details

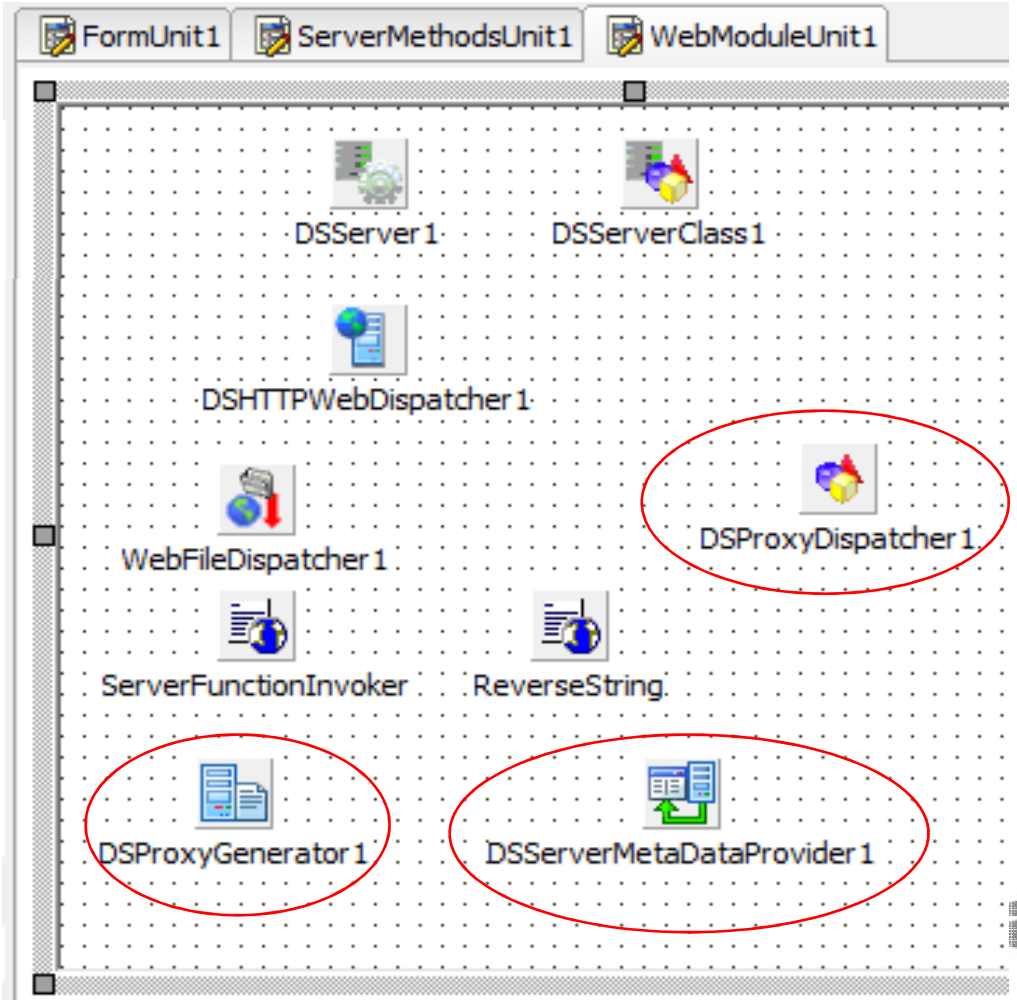
5. Enter the root directory of the web application.



This is the output directory of the project executable and the location of web application files such as .js, .html, and .css files, and other static files

DataSnap Mobile Connectors Step-By-Step Details

DataSnap REST Application with Mobile Connectors.



Components needed to enable a DataSnap REST server to generate the proxies:

TDSProxyGenerator
TDSServerMetaDataProvider
TDSProxyDispatcher

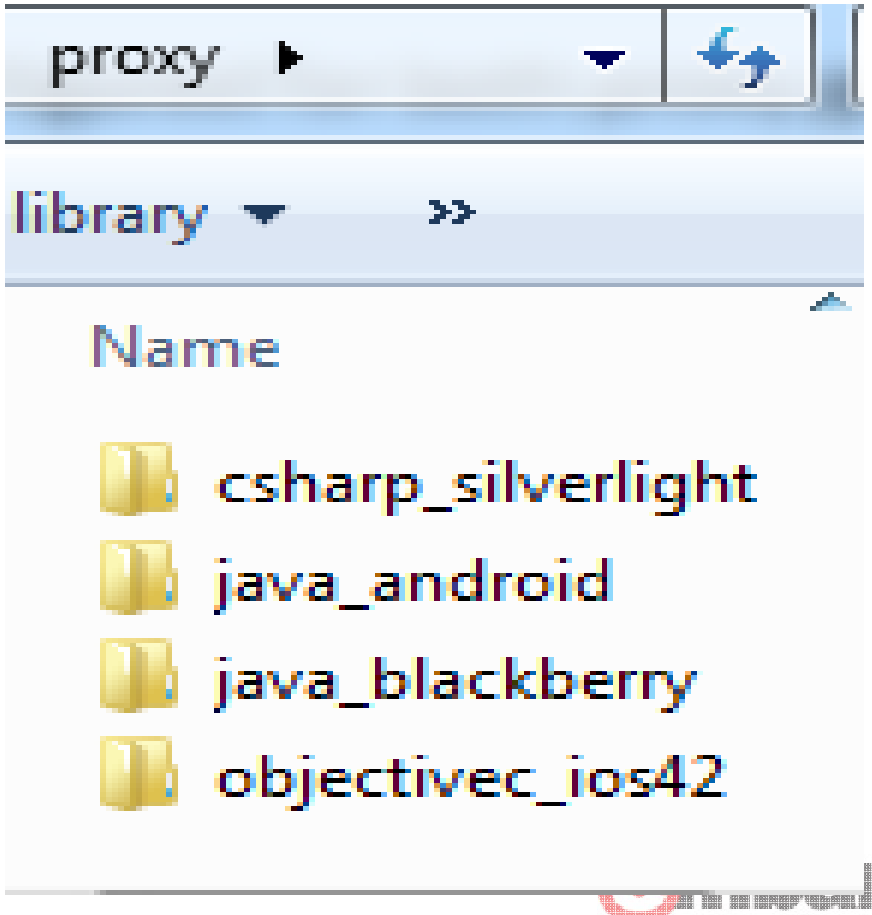
Your DataSnap server is now configured for both proxy generation and proxy dispatching!

adero

DataSnap Mobile Connectors

Step-By-Step Details

Getting the Generated Proxy from the Wizard.



Two ways to request the proxy:

1. http://host:port/proxy/[DEVICE].zip
2. **Win32ProxyDownloader.exe**

This creates the proxy source code for the specified **[DEVICE]** mobile device

DataSnap Mobile Connectors

Step-By-Step Details

Generated Proxy Source Code

```
// Created by the DataSnap proxy generator.

package com.embarcadero.javaandroid;
   /**
    * @param Value [in] - Type on server: string
    * @return result - Type on server: string
    */
   public String EchoString(String Value) throws DBXException {
     DSRESTCommand cmd = getConnection().CreateCommand();
     cmd.setRequestType(DSHTTPRequestType.GET);
     cmd.setText("TServerMethods1.EchoString");
     cmd.prepare(get_TServerMethods1_EchoString_Metadata());
     cmd.getParameter(0).getValue().SetAsString(Value);
     getConnection().execute(cmd);
     return  cmd.getParameter(1).getValue().GetAsString();
   }
```

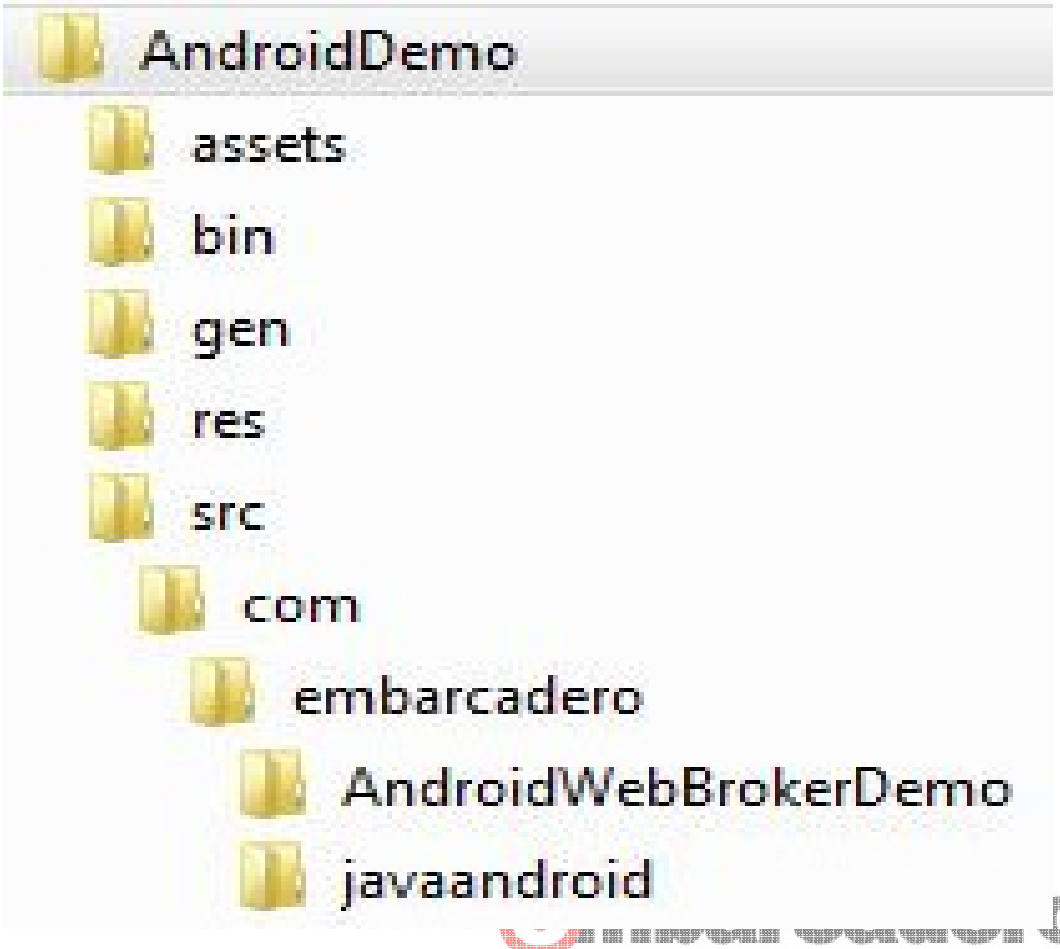
The proxy source code created includes:

- Static Java source files required by the proxy
- **DSProxy.java**, the generated proxy

DataSnap Mobile Connectors

Step-By-Step Details

Import Proxy Source Code into your Mobile client.



DataSnap Mobile Connectors - Step-By-Step Details

How to use DSProxy class

```
public class DSProxy {
  public static class TServerMethods1 extends DSAdmin {
    public TServerMethods1(DSRESTConnection Connection) {
      super(Connection);
    }

    private DSRESTParameterMetaData[] TServerMethods1_EchoString_Metadata;
    private DSRESTParameterMetaData[] get_TServerMethods1_EchoString_Metadata() {
      if (TServerMethods1_EchoString_Metadata == null) {
        TServerMethods1_EchoString_Metadata = new DSRESTParameterMetaData[]{
          new DSRESTParameterMetaData("Value", DSRESTParamDirection.Input,
                                      DBXDataTypes.WideStringType, "string"),
          new DSRESTParameterMetaData("", DSRESTParamDirection.ReturnValue,
                                      DBXDataTypes.WideStringType, "string"),
        };
      }
      return TServerMethods1_EchoString_Metadata;
    }
```

DataSnap Mobile Connectors - Step-By-Step Details

How to use DSProxy class (continued)

```
/**
   * @param Value [in] - Type on server: string
   * @return result - Type on server: string
   */
  public String EchoString(String Value) throws DBXException {
    DSRESTCommand cmd = getConnection().CreateCommand();
    cmd.setRequestType(DSHTTPRequestType.GET);
    cmd.setText("TServerMethods1.EchoString");
    cmd.prepare(get_TServerMethods1_EchoString_Metadata());
    cmd.getParameter(0).getValue().SetAsString(Value);
    getConnection().execute(cmd);
    return cmd.getParameter(1).getValue().GetAsString();
  }
```

DataSnap Mobile Connectors - Step-By-Step Details

Create and setup a connection to the server using the proxy class.

```
import com.embarcadero.javaandroid.DSRESTConnection;
...
//The host property specifies the host name or  IP address.
//The port property specifies the port  which the servers is listening on.
//The protocol property  specifies the protocol to use for the connection. Specifically; http or https.
DSRESTConnection conn = new DSRESTConnection();
conn.setHost("host");
conn.setPort(port);
conn.setProtocol( "http“ );
```

Class **DSRESTConnection** encapsulates all the properties and methods needed for connecting to the server.

DataSnap Mobile Connectors - Step-By-Step Details

To remotely invoke the server methods, just create the proxy class, passing connection, and then call the method:

```
DSRESTConnection conn = new DSRESTConnection();
conn.setHost ( "10.40.30.24“);
conn.setPort(8080);
conn.setProtocol( "http“ );

TServerMethods1 proxy = new TServerMethods1(conn);
String Result = proxy.EchoString( "Hello, World!“ );
```

DataSnap Mobile Connectors - Step-By-Step Details

Example of a server method with a *var* parameter

```
function TServerMethods1.VarParamTest(var Value: string): string;
begin
  Value := StrUtils.ReverseString(Value);
  Result := Value;
end;
```

DataSnap Mobile Connectors - Step-By-Step Details

Generated proxy code to invoke a server method **with a *var* parameter**

```
private DSRESTParameterMetaData[] TServerMethods1_VarParamTest_Metadata;
private DSRESTParameterMetaData[] get_TServerMethods1_VarParamTest_Metadata() {
  if (TServerMethods1_VarParamTest_Metadata == null) {
    TServerMethods1_VarParamTest_Metadata = new DSRESTParameterMetaData[]{
      new DSRESTParameterMetaData("Value", DSRESTParamDirection.InputOutput,
                                  DBXDataTypes.WideStringType, "string"),
      new DSRESTParameterMetaData("", DSRESTParamDirection.ReturnValue,
                                  DBXDataTypes.WideStringType, "string"),
    };
  }
  return TServerMethods1_VarParamTest_Metadata;
}
```

DataSnap Mobile Connectors - Step-By-Step Details

Generated proxy code to invoke a server method **with a *var* parameter (continued)**

```
/**
 * @param Value [in/out] - Type on server: string
 * @return result - Type on server: string
 */
public static class VarParamTestReturns {
  public String Value;
  public String returnValue;
}

public VarParamTestReturns VarParamTest(String Value) throws DBXException {
  DSRESTCommand cmd = getConnection().CreateCommand();
  cmd.setRequestType(DSHTTPRequestType.GET);
  cmd.setText("TServerMethods1.VarParamTest");
  cmd.prepare(get_TServerMethods1_VarParamTest_Metadata());
  cmd.getParameter(0).getValue().SetAsString(Value);
  getConnection().execute(cmd);
  VarParamTestReturns ret = new VarParamTestReturns();
  ret.Value = cmd.getParameter(0).getValue().GetAsString();
  ret.returnValue = cmd.getParameter(1).getValue().GetAsString();
  return ret;
}
```

DataSnap Mobile Connectors - Step-By-Step Details

The example shows how to invoke the method and read the resulting values.

```
TServerMethods1 proxy = new TServerMethods1(conn);

VarParamTestReturns Result = proxy.VarParamTest( "Hello, World!" );

System.out.println(Result.Value);

System.out.println(Result.returnValue);
```

Both the new value of the parameter passed in and the result value will be held by the *Result* instance.

DataSnap Mobile Connectors - Step-By-Step Details

How to use heavyweight callbacks

```
public class MyCallback extends DBXCallback {
   public TJSONValue execute(TJSONArray params) {
      System.out.println(params.toString());
      return new TJSONTrue();
  }
}
```

```
DSCallbackChannelManager manager  =
 new
DSCallbackChannelManager(conn,"chname",DSCallbackChannelManager.getNew
ManagerID());
```

embarcadero

DataSnap Mobile Connectors - Step-By-Step Details

How to use heavyweight callbacks (continued)

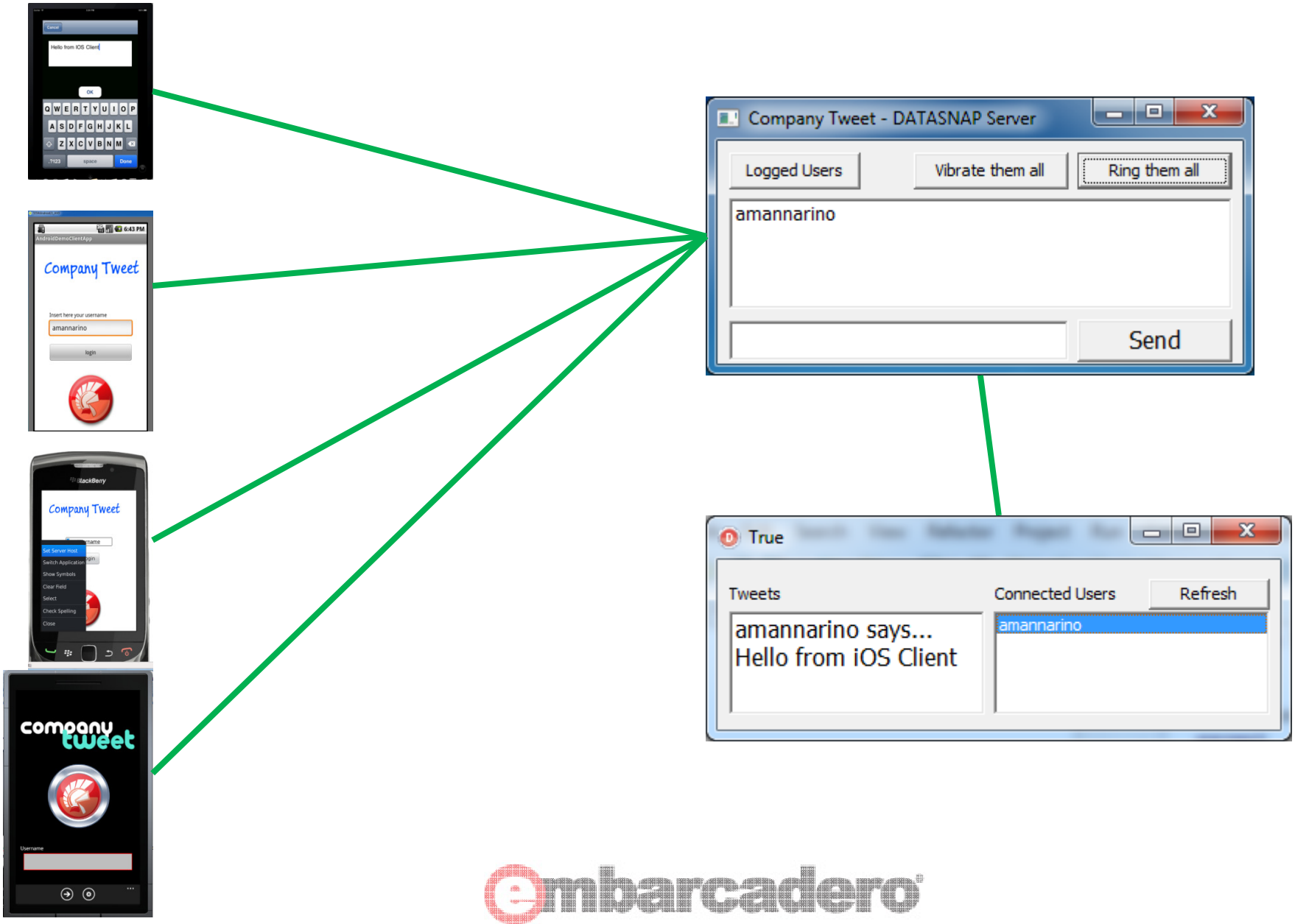
Example of registering a callback and then broadcasting to the same channel it is registered on:

```
manager.registerCallback("mycb01", new MyCallback());
```

To stop the connection we can unregister the callback and close the channel:

```
manager.unregisterCallback(“mycb01“);
manager.closeClientChannel();
```

DataSnap Mobile Connectors - Demo



embarcadero

Mobile Connectors

- 範例
  - Android客戶端使用Mobile Connectors呼叫DataSnap Server

結合分散式和移動式開發

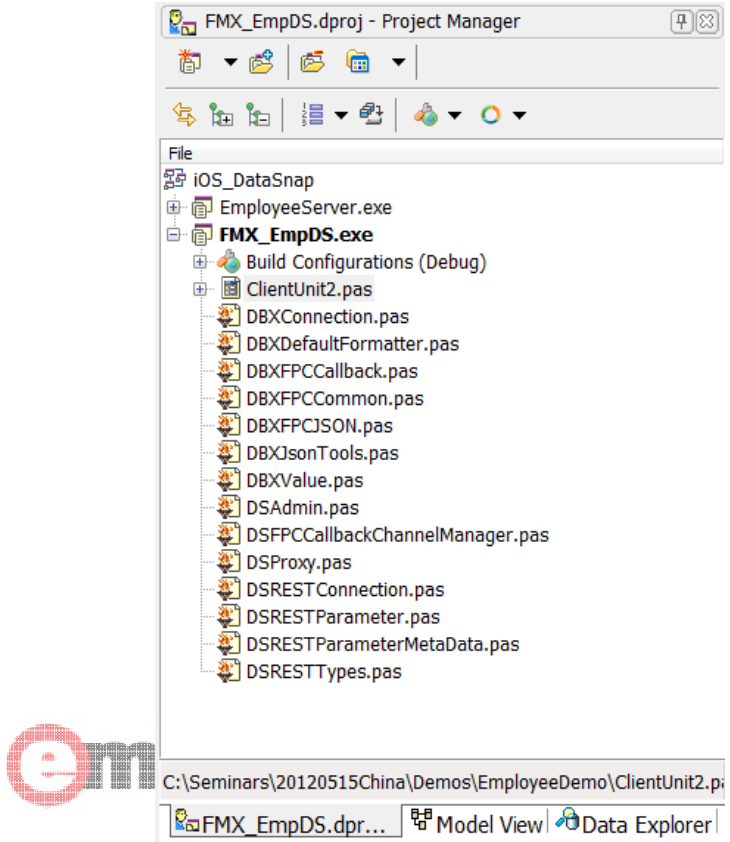
原生式開發

結合分散式和移動式開發

- 在XE2 Update 4之前只能藉由Mobile Connectors開發DataSnap客戶端
- XE2 Update 4對iOS的支提供了更進一步的支援, 允許開發人員直接使用Object Pascal藉由dbExpress在iPhone/iPad上直接連結DataSnap Server
- 整個開發步驟如下:

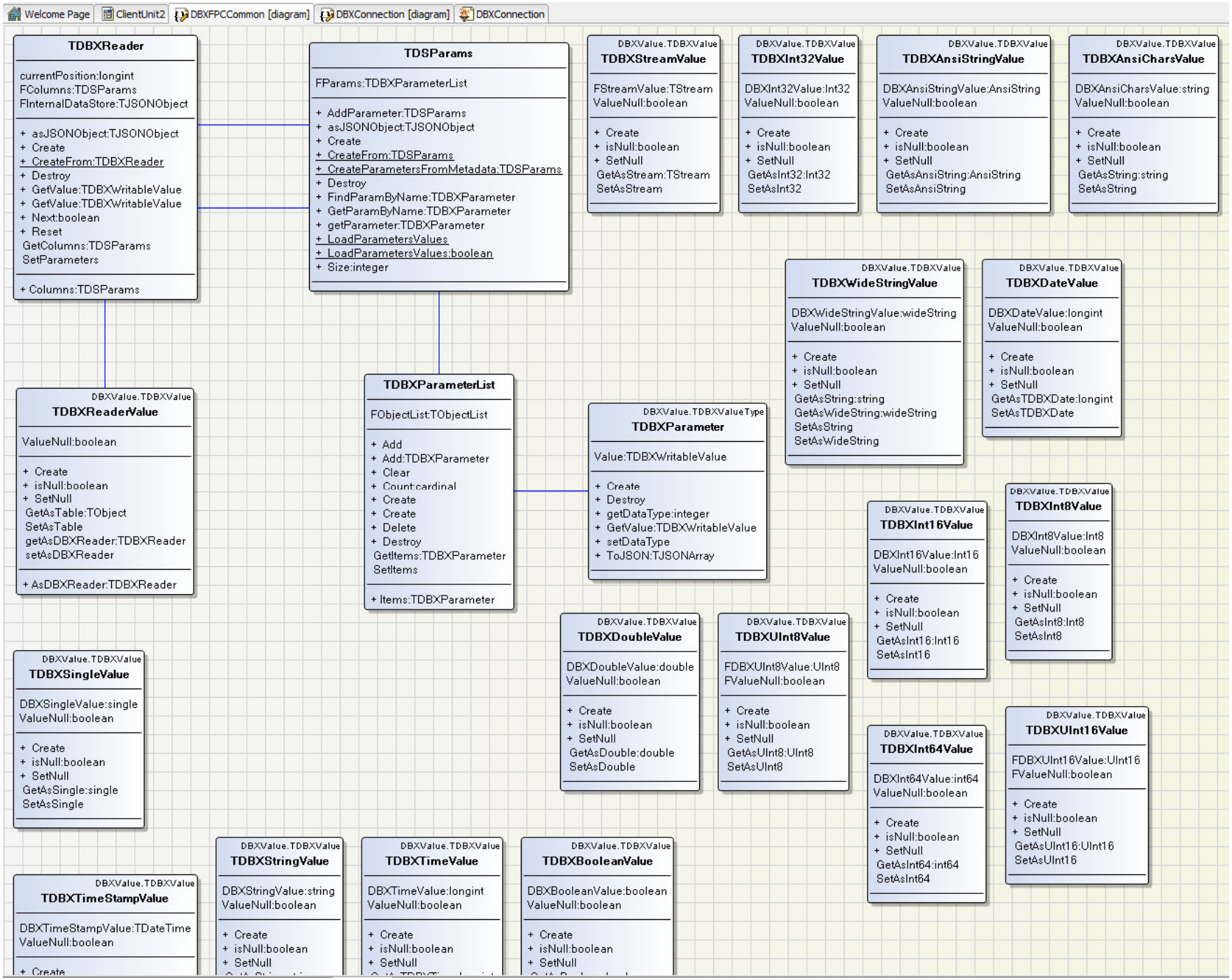
結合分散式和移動式開發

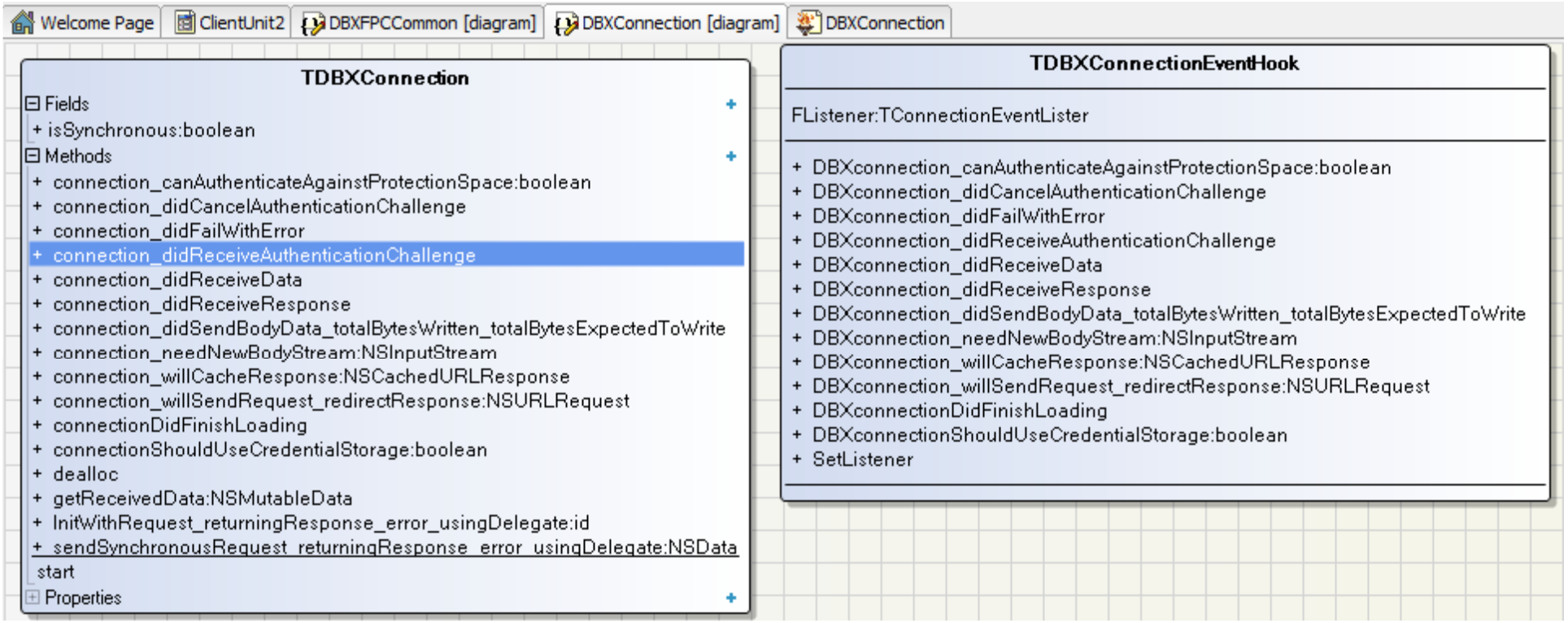
1) 開發DataSnap Server
2) 使用瀏覽器執行下列URL取得iOs能使用的dbExpress程式碼
   - http://yourdatasnapserver:8080/proxy/freepascal_ios50.zip
3) 解開上面的ZIP檔到你的DataSnap客戶端專案目錄



結合分散式和移動式開發

4) 使用其中的DBXConnection.pas, DBXFPCCallback.pas, DBXFPCCommon.pas, DBXFPCJSON.pas等程式單元, 如同Windows DataSnap客戶端一樣呼叫DataSnap Server


Welcome Page
ClientUnit2
DBXFPCCommon [diagram]
DBXConnection [diagram]
DBXConnection
TDBXReader
currentPosition:longint
FColumns:TDSParams
FInternalDataStore:TJSONObject
+ asJSONObject:TJSONObject
+ Create
+ CreateFrom:TDBXReader
+ Destroy
+ GetValue:TDBXWritableValue
+ GetValue:TDBXWritableValue
+ Next:boolean
+ Reset
GetColumns:TDSParams
SetParameters
+ Columns:TDSParams
TDSParams
FParams:TDBXParameterList
+ AddParameter:TDSParams
+ asJSONObject:TJSONObject
+ Create
+ CreateFrom:TDSParams
+ CreateParametersFromMetadata:TDSParams
+ Destroy
+ FindParamByName:TDBXParameter
+ GetParamByName:TDBXParameter
+ getParameter:TDBXParameter
+ LoadParametersValues
+ LoadParametersValues:boolean
+ Size:integer
DBXValue.TDBXValue
TDBXStreamValue
FStreamValue:TStream
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsStream:TStream
SetAsStream
DBXValue.TDBXValue
TDBXInt32Value
DBXInt32Value:Int32
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsInt32:Int32
SetAsInt32
DBXValue.TDBXValue
TDBXAnsiStringValue
DBXAnsiStringValue:AnsiString
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsAnsiString:AnsiString
SetAsAnsiString
DBXValue.TDBXValue
TDBXAnsiCharsValue
DBXAnsiCharsValue:string
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsString:string
SetAsString
DBXValue.TDBXValue
TDBXWideStringValue
DBXWideStringValue:wideString
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsString:string
GetAsWideString:wideString
SetAsString
SetAsWideString
DBXValue.TDBXValue
TDBXDateValue
DBXDateValue:longint
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsTDBXDate:longint
SetAsTDBXDate
DBXValue.TDBXValue
TDBXReaderValue
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsTable:TObject
SetAsTable
getAsDBXReader:TDBXReader
setAsDBXReader
+ AsDBXReader:TDBXReader
TDBXParameterList
FObjectList:TObjectList
+ Add
+ Add:TDBXParameter
+ Clear
+ Count:cardinal
+ Create
+ Create
+ Delete
+ Destroy
GetItems:TDBXParameter
SetItems
+ Items:TDBXParameter
DBXValue.TDBXValueType
TDBXParameter
Value:TDBXWritableValue
+ Create
+ Destroy
+ getDataType:integer
+ GetValue:TDBXWritableValue
+ setDataType
+ ToJSON:TJSONArray
DBXValue.TDBXValue
TDBXInt16Value
DBXInt16Value:Int16
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsInt16:Int16
SetAsInt16
DBXValue.TDBXValue
TDBXInt8Value
DBXInt8Value:Int8
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsInt8:Int8
SetAsInt8
DBXValue.TDBXValue
TDBXSingleValue
DBXSingleValue:single
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsSingle:single
SetAsSingle
DBXValue.TDBXValue
TDBXDoubleValue
DBXDoubleValue:double
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsDouble:double
SetAsDouble
DBXValue.TDBXValue
TDBXUInt8Value
FDBXUInt8Value:UInt8
FValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsUInt8:UInt8
SetAsUInt8
DBXValue.TDBXValue
TDBXInt64Value
DBXInt64Value:int64
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsInt64:int64
SetAsInt64
DBXValue.TDBXValue
TDBXUInt16Value
FDBXUInt16Value:UInt16
FValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
GetAsUInt16:UInt16
SetAsUInt16
DBXValue.TDBXValue
TDBXTimeStampValue
DBXTimeStampValue:TDateTime
ValueNull:boolean
+ Create
DBXValue.TDBXValue
TDBXStringValue
DBXStringValue:string
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
DBXValue.TDBXValue
TDBXTimeValue
DBXTimeValue:longint
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull
DBXValue.TDBXValue
TDBXBooleanValue
DBXBooleanValue:boolean
ValueNull:boolean
+ Create
+ isNull:boolean
+ SetNull



結合分散式和移動式開發

◆範例1

1. 建立DataSnap XE2伺服器
2. 建立FireMonkey iOS HD客戶端應用程式
3. 使用XE2 Update 4提供的Free Pascal版的DBX類別連結DataSnap XE2伺服器取得服務
   - http://yourdatasnapserver:8080/proxy/freepascal_ios50.zip

結合分散式和移動式開發

- 範例1
  - 簡單的iOS DataSnap客戶端
- 範例2
  - 存取資料庫的iOS DataSnap客戶端
- 範例3
  - FishFact iOS客戶端

謝謝您的參加!

embarcadero