Appcan 2.0 Native插件扩展机制

功能描述：

Native插件机制，是指针对native 开发者封装将具有特定功能但是平台没有封装的接口，封装成js调用Native功能，然后通过js扩展接口调用，以达到功能扩展的目的。

2.0的扩展机制与1.2扩展机制有所差别，增加了资源文件的添加，动态获取，Activity回调，自定义View到当前window，接口参数可任意定制等等。代码上，主要体现在**EUExBase.java**的变化，接口函数的参数个数由0个或者多个改为了1个数据类型为String数组的参数，即所有的扩展对象中的接口必须有且仅有一个参数(String[])，可在示例代码中查看到。

Android SDK

**Appcan native插件SDK，由一个基础工程构成，主要包括引擎jar包，接口类，android工程资源文件夹res，插件配置xml，widget文件夹等等。**

**详细描述：**

1. 引擎jar包AppCanEngine\_xxx.jar

位于工程目录的libs目录下，是利用appcan平台扩展native plugin的基础。

1. 接口类

主要包括以下3个：

**EUExBase.java：**封装了JS调用native 以及native回调JS的桥接函数的父类，任何 扩展的对象均需要继承此类；

**EUExUtil.java：**提供动态获取本应用资源id等功能的工具类。注意：Appcan中的 所有资源文件，包括字符串资源等，都必须使用此工具类当中的相 关函数动态获取其资源ID，而不能直接使用R文件引用！具体如何 使用可在示例代码中看到。

**EUExCallback.java：**与plugin的callback相关的一些常量。

1. 资源文件夹（res文件夹）

此文件夹中（包括子目录）已经定义或者命名的任何资源文件，不允许有任何重命名，删除等操作，这些资源文件属于AppCan平台所使用，一旦平台找不到相关资源文件，将报错退出。

1. Error文件夹（assets目录下的error文件夹）

error文件夹下存放网页访问错误时候显示的界面，必须拷贝到资源下，如果想定制自己的可以替换文件夹下资源但是不能修改文件名称，有可能导致文件找不到，加载方式同图片资源

1. plugin.xml

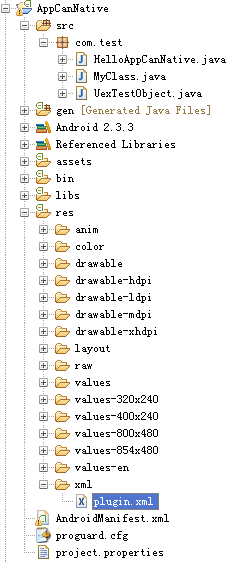
plugin.xml 文件位于res目录下的xml目录中，是配置自定义native Plugin调用对象的xml文件，如果需要自定义对象和开发natvie plugin，必须在此文件中配置自定义js对象名和java类的包名类名。

1. widget 文件夹（assets目录下的widget文件夹）

widget 文件夹为网页包，默认有config.xml 文件，基本数据已经默认填写，开发者只需修改部分参数即可，切勿删除config.xml 文件，widget文件也必须加载到工程中；

配置工程

1. 打开eclipse，将appcan android工程导入。如图：



等待eclipse自动编译完后，检查是否报错；如未报错，直接运行即可。

其中：UexTestObject.java为自定义plugin的测试实例。

API接口

**平台接入类**

平台的自带的生命周期函数进行了封装；

**JS调用Native plugin基础类------EUExBase.java**

EUExBase对象成员描述：

**mBrwView**：

由构造函数传入的继承了Webview并封装了appcan 平台接口的EBrowserView的对象实例；

**mContext**：AppCan平台的上下文，AppCan主Activity的实例。

EUExBase对象成员函数描述：

**- public void jsCallback(String inCallbackName, int inOpCode, int inDataType, String inData)**

自定义对象完成一系列操作后回调网页js时的函数，其中：

**inCallbackName：**网页中js函数的名称即所属js对象名称的集合。如：uexDemo.cbFunc1，即为uexDemo对象的func1函数被调用后，java回调js时的callback函数。

**inOpCode：**网页中随机分配的不重复int类型的数值，主要用于区分一个js接口函数多次操作java中不同的对象。由网页调用native接口时传入。

**inDataType：**本次回调网页js时，传给js的数据的数据类型，可参考EUExCallback对象下的静态值：F\_C\_TEXT，F\_C\_JSON，F\_C\_INT，F\_C\_STATUS。

**inData：**回调给页面js的值。

**- public void jsCallback(String inCallbackName, int inOpCode, int inDataType, int inData)**

同上。当返回的数据类型为F\_C\_INT等数值型时，需走这个callback。

**-** **public void errorCallback(int inOpCode, int inErrorCode, String inErrorInfo)**

当调用接口发生错误时，java层通过此回调通知js。其中：

**inOpCode：**同上inOpCode。

**inErrorCode：**错误码，可自定义。

**inErrorInfo：**返回给js的错误信息。

**-** **public void onCallback(String inScript)**

Java层延时主动回调js的函数。如GPS定位时，会有多次回调，可通过此接口回调。

**-** **public String getCookie(String inUrl)**

获取inUrl对应页面的cookie。

**-** **public void setCookie(String inUrl, String cookie)**

设置inUrl对应页面的cookie。

**-** **public abstract boolean clean()**

抽象函数，必须实现。当网页切换或者销毁时，AppCan会主动调用此函数，所以应该在此函数中做一些释放你占用的资源的操作。

- **public void startActivityForResult(Intent intent, int requestCode)**

运行一个Activity,并要求被运行的Activity在finish时有返回值.

你的plugin中,如果需要运行另一个Activity,并且需要此Activity返回数据时,必须要通过此接口调用，返回的数据将通过onActivityResult函数回调,可在onActivityResult函数中做相关处理。

- **public void startActivity(Intent intent)**

运行一个不需要返回值的Activity。

**-** **public void onActivityResult(int requestCode, int resultCode, Intent data)**

当自定义对象有startActivityForResult操作，目标Activity结束后返回结果时可通过重载这个函数回调。

- **public void addViewToCurrentWindow(View child, RelativeLayout.LayoutParams parms)**

添加一个你的自定义view覆盖到当前window中的最上层。

- **public void removeViewFromCurrentWindow(View child)**

从当前window移除一个你的自定义view。

**Plugin动态获取资源id及提供平台相关功能操作的工具类---- EUExUtil.java**

EUExUtil对象函数描述：

-**public static int getResDrawableID(String resName)**

获取DrawableI类型资源的id。

-**public static int getResLayoutID(String resName)**

获取Layout类型资源的id。

-**public static int getResAnimID(String resName)**

获取Anim类型资源的id。

-**public static int getResAttrID(String resName)**

获取Attr类型资源的id。

-**public static int getResColorID(String resName)**

获取Color类型资源的id。

-**public static int getResDimenID(String resName)**

获取Dimen类型资源的id。

-**public static int getResIdID(String resName)**

获取Id类型资源的id。

-**public static int getResRawID(String resName)**

获取Raw类型资源的id。

-**public static int getResStringID(String resName)**

获取String类型资源的id。

-**public static int getResStyleID(String resName)**

获取Style类型资源的id。

-**public static int getResXmlID(String resName)**

获取Xml类型资源的id。

以上接口具体使用可参考SDK工程中的示例UexTestObject对象的定义，以及assets目录下，widget文件中index.html网页的书写。

EUExCallback对象成员描述：

EUExCallback对象为AppCan引擎为EUExBase配套提供的一个存放常量的类，包括回调js是，当传递数据为json数据时的key值，平台类型，传递给js的数据类型的描述等常量。

Samples

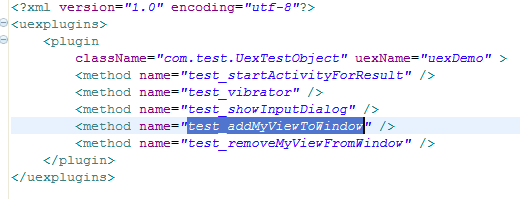
以下内容以扩展uexDemo对象为例，描述为appcan扩展一个native plugin 的开发流程：

1. 配置plugin.xml ：

找到工程中res目录下的xml目录，在plugin.xml文件中配置你的扩展对象相关信息。

(1)**、配置自定义对象**：

示例对象uexDemo对象配置如下：



*uexName*： 为封装的js对象的名称，可任意命名；

*className：* 与js对象映射的java对象的路径包名及类名；

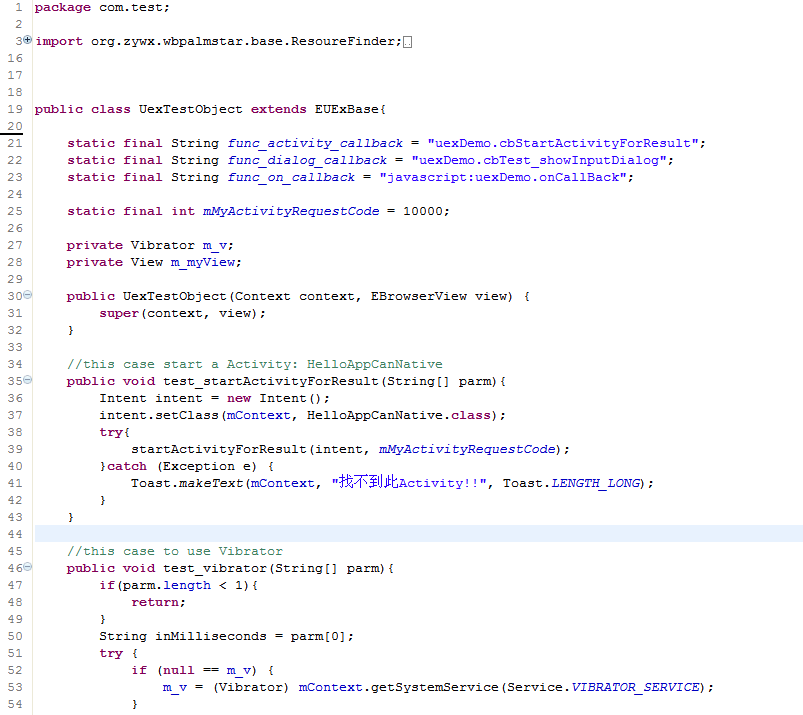
可定义多个plugin。

(2)**、定义与js对象映射的java类：**

此类必须继承自plugin的基础类：EUExBase，并实现或重写相关函数。

新建java类UexTestObject.java并继承自EUExBase。其中：

UexTestObject.java类似于下图：



其中：所有接口函数的参数均为1个string数组。此数组的长度即为js传过来的参数个数，此数组的index与js中参数的index相对应。

例如，在js中有类似调用：

uexDemo. test\_vibrator(p1，p2，p3，p4)

那么，当它映射到UexTestObject.java的：

public void test\_vibrator(String[] parm)

函数中时，parm的长度将为4，可通过parm[0]取得p1，parm[1]取得p2，parm[2]取得p3，parm[3]取得p4，以此类推。

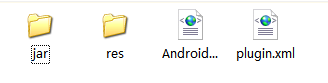
这与1.2中，JS中有4个参数，那么java中也必须有4个参数与之对应是不同的。

具体书写及回调网页的方式请参考工程中UexTestObject.java的详细代码。

提交plugin

开发完自己的plugin后，将你的java代码导出成jar包，并连同此plugin所使用到的资源文件，以规定的格式打包成zip包，并提交给appcan官方。

一个带有资源文件的native插件包目录结构一般类似于下图：

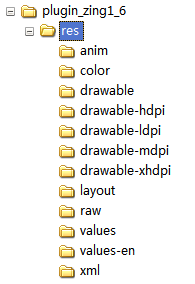


其中，jar目录为存放你的jar文件的目录，jar目录和 plugin.xml文件为必须的（如果jar包对应有jni接口，则含有so文件），其他不带资源文件的插件可能没有res目录或者没有AndroidManifest.xml文件。

相应目录及文件解释：

**jar目录：**存放你的jar包的目录，如果有多个jar包，都放在这个目录下。其中，jar包的命名规则为：plugin打头，加你的扩展对象名，再加具体的后缀，如：plugin\_uexDemo.jar。

res目录：此目录中的所有目录同中间件基础工程的res目录相对应。类似于下图：

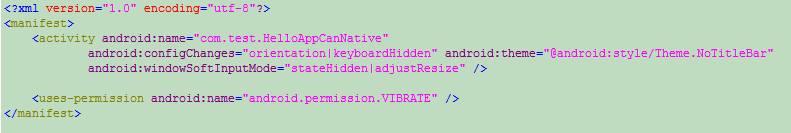


对于所有res资源，其命名规则为：plugin打头，加你的plugin对象名再加其他信息。如：

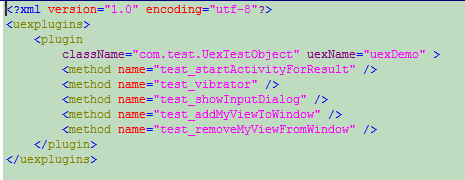
plugin\_uexdemo\_xxx.png、plugin\_uexdemo\_yyy.xml、

<string name=" plugin\_uexdemo\_zzz ">等等。

AndroidManifest.xml文件：配置本native插件中用到的activty，service，receiver，权限等的主配置文件。只配置此plugin用到的。其结构类似于下图：



plugin.xml文件：定义本native插件扩展的uex对象及其接口函数的文件。只配置此plugin用到的。其结构类似于下图：



**注：本sdk目录下有一个已经打好的uexDemo的插件包示例，可做参考。**

**至此，一个native plugin开发完毕。**