**Cordova安装创建**

1. 安装cordova

命令：npm install -g cordova

1. 创建cordova项目

命令：cordova create cordovaApp //cordovaApp: 创建项目名称



说明：第一个qingzhen表示文件夹的名称

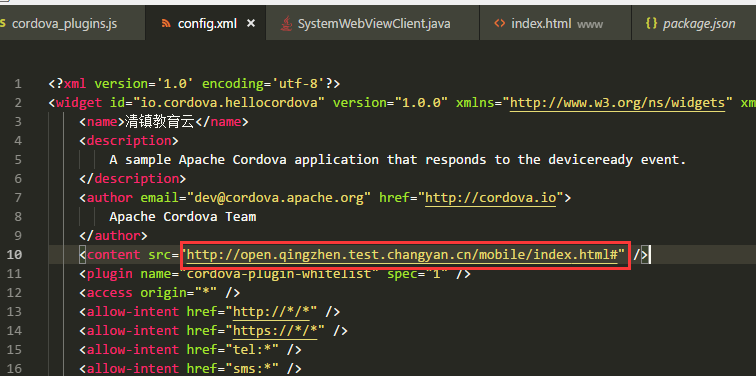
第二个com.qingzhen.app 表示项目id

第三个qingzhne表示项目名称

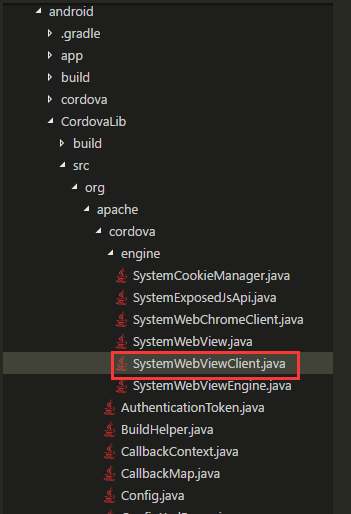
cd cordovaApp //转到你创建的目录

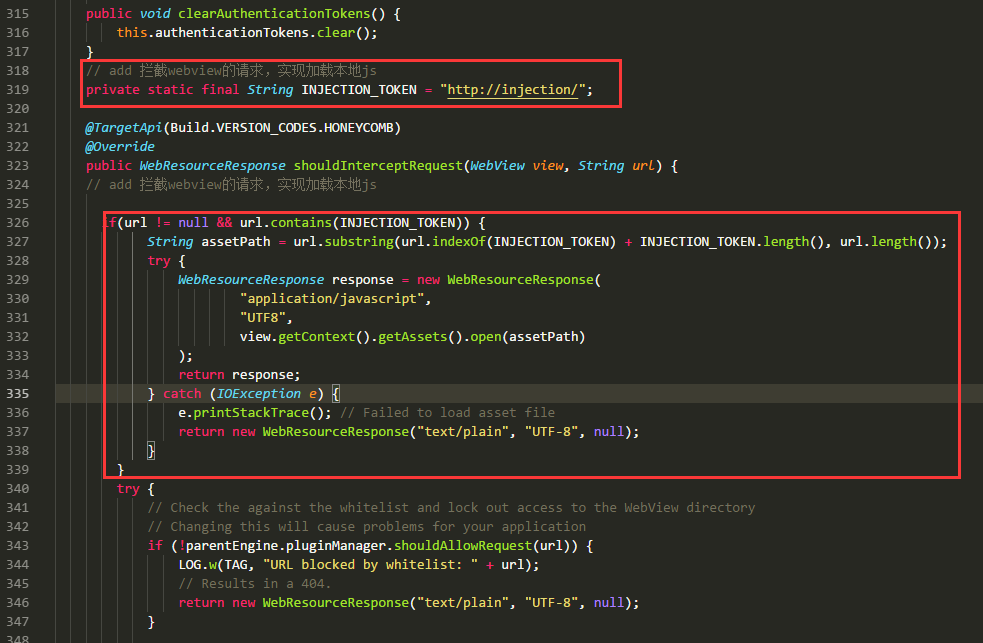
cordova platform add android//添加构建app的平台

1. 打包构建（测试使用apk）
2. 命令：cordova run android//测试 手机连上电脑
3. 命令：cordova build android//生成apk
4. 路径：platforms\android\app\build\outputs\apk\debug
5. 如果你打包的是一个服务器的地址，那么你需要以下cordova文件：
6. 将cordova文件中的config.xml的content修改成服务器地址



1. 修改下面这个文件 新增两段代码





// add 拦截webview的请求，实现加载本地js

private static final *String* INJECTION\_TOKEN = "http://injection/";

// add 拦截webview的请求，实现加载本地js

if(url != null && url.contains(INJECTION\_TOKEN)) {

*String* assetPath = url.substring(url.indexOf(INJECTION\_TOKEN) + INJECTION\_TOKEN.length(), url.length());

try {

*WebResourceResponse* response = new WebResourceResponse(

"application/javascript",

"UTF8",

view.getContext().getAssets().open(assetPath)

);

return response;

} catch (*IOException* *e*) {

e.printStackTrace(); // Failed to load asset file

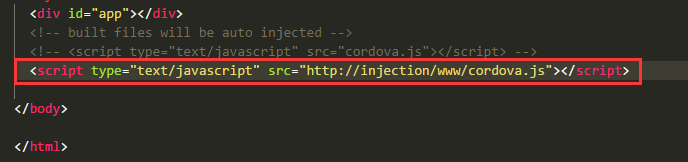
return new WebResourceResponse("text/plain", "UTF-8", null);

}

}

1. 修改你所在代码引入的cordova.js

<script type="text/javascript" src="http://injection/www/cordova.js"></script>



4. 修改安装的背景图标

1. 安装 cordova-plugin -splashscreen

命令：cordova plugin add cordova-plugin-splashscreen

1. 替换原有cordova的图片
2. 在cordova config.xml 文件中引入以下代码

<platform name="android">

<icon density="ldpi" src="res/icon/android/icon-36-ldpi.png" />

<icon density="mdpi" src="res/icon/android/icon-48-mdpi.png" />

<icon density="hdpi" src="res/icon/android/icon-72-hdpi.png" />

<icon density="xhdpi" src="res/icon/android/icon-96-xhdpi.png" />

<splash density="port-hdpi" src="res/screen/android/screen-hdpi-portrait.png" />

<splash density="port-ldpi" src="res/screen/android/screen-ldpi-portrait.png" />

<splash density="port-mdpi" src="res/screen/android/screen-mdpi-portrait.png" />

<splash density="port-xhdpi" src="res/screen/android/screen-xhdpi-portrait.png" />

</platform>

5.下载文件

1. 安装 cordova-plugin -file-transfer

命令：cordova plugin add cordova-plugin-file-transfer

地址：<https://cordova.apache.org/docs/en/latest/reference/cordova-plugin-file-transfer/>

//HTML

<div class="app">

<div class="line"><button id="download">下载</button></div>

<div class="line"><img id="myImage"></img></div>

<div class="line"><span id="text"></span></div>

</div>

<script type="text/javascript" src="cordova.js"></script>

*//JS代码*

*var* app = {

// Application Constructor

initialize: *function*() {

document.addEventListener('deviceready', this.onDeviceReady.bind(this), false);

},

// deviceready Event Handler

onDeviceReady: *function*() {

this.receivedEvent();

},

// get DOM

$$: *function*(*id*) {

return document.getElementById(id);

},

receivedEvent: *function*() {

*var* \_this = this;

*var* dlDom = this.$$('download');

dlDom.onclick = *function*() {

\_this.createFilePath();

}

},

// 创建文件路径

createFilePath: *function*() {

*var* \_this = this;

window.requestFileSystem(*LocalFileSystem*.PERSISTENT, 0, *function*(*fs*) {

fs.root.getFile("downloadedImage.png", { create: true, exclusive: false }, *function*(*fileEntry*) {

*console*.log(fileEntry);

//调用fileTransfer插件，下载图片

\_this.downLoadImg(fileEntry.nativeURL);

}, *function*(*err*) {

*console*.log(err);

});

}, *function*(*error*) {

*console*.log(error);

});

},

// fileTransfer plugin

downLoadImg: *function*(*fileURL*) {

*var* \_this = this;

// 初始化FileTransfer对象

*var* fileTransfer = new FileTransfer();

// 服务器下载地址

*var* uri = encodeURI("http://avatar.csdn.net/7/E/0/1\_michael\_ouyang.jpg");

// 调用download方法

fileTransfer.download(

uri, //uri网络下载路径

fileURL, //url本地存储路径

*function*(*entry*) {

*console*.log("download complete: " + entry.toURL());

\_this.$$('myImage').src = entry.toURL();

},

*function*(*error*) {

*console*.log("download error source " + error.source);

*console*.log("download error target " + error.target);

*console*.log("upload error code" + error.code);

}

);

}

};

app.initialize();

6.回退按钮

1. 安装 cordova-plugin-backbutton

命令：cordova plugin add cordova-plugin-backbutton

地址：[https://cordova.apache.org/docs/en/5.1.1/cordova/events/events.backbutton.html](https://cordova.apache.org/docs/en/5.1.1/cordova/events/events.backbutton.html%20)

/\* 初始化设备 \*/

initialize() {

document.addEventListener(

'deviceready',

this.onDeviceReady.bind(this),

false

);

},

/\* 设备初始化之后监听返回按钮事件 \*/

onDeviceReady() {

this.ready = true;

document.addEventListener('backbutton', this.onBackDown, false);

},

/\* 返回按钮事件 \*/

onBackDown() {

/\* 当设备已准备并且当前页时在主页的时候 退出程序 \*/

navigator.app.exitApp();

}

7. 检测跳转第三方app

1. 检测是否安装了第三方app：

安装：cordova-plugin- appavailability

命令：cordova plugin add appavailability

地址:  [https://www.npmjs.com/package/cordova-plugin-appavailability](%20https://www.npmjs.com/package/cordova-plugin-appavailability)

1. 跳转第三方APP：

安装：cordova-plugin-com.lampa.startapp

命令：cordova plugin add com.lampa.startapp

地址：<https://github.com/lampaa/com.lampa.startapp>

appAvailability.check(

'cn.com.lezhixing.clover.xf', // 校信 Package Name

*function*(*info*) {

// Success callback

// Info parameter is available only for android

window.location.href = scheme;

},

*function*() {

\_this.popupAppVisible = true;

}

);

注：如果你跳转的第三方是特殊的带头地址，需要在cordova的config.xml中添加一下代码

<allow-intent href="XXX /\*" />

**打包带签名的apk**

1. 生成不带签名的apk

命令：cordova build --release android

生成的文件名称是app-release-unsigned.apk

1. 在生成apk的目录下（只是便于查找），生成签名

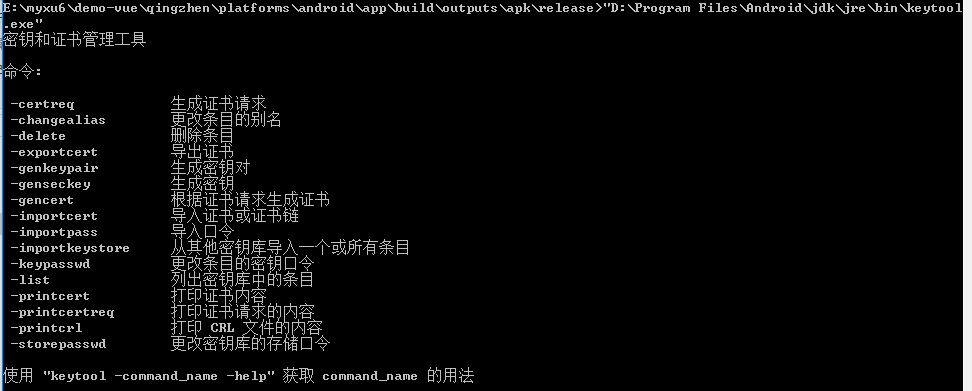
命令：keytool -genkeypair -alias demo.keystore -keyalg RSA -validity 4000 -keystore demo.keystore

将name替换成你所需要的生成签名文件的名称

这里需要注意，当你输入keytool时，可能会出现

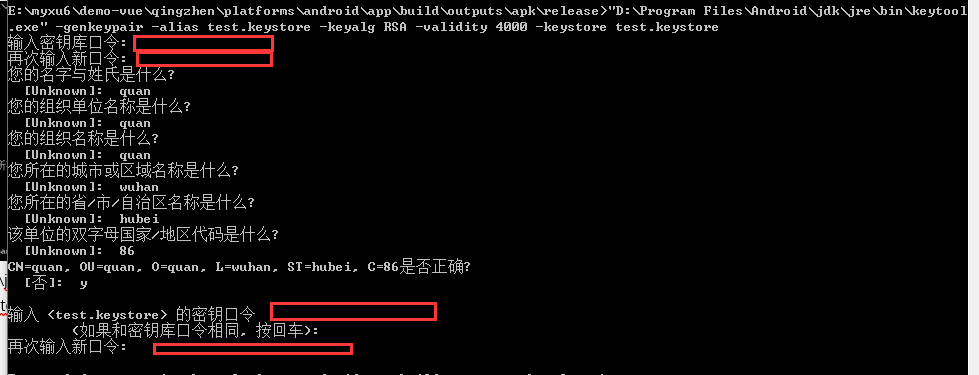


有可能时因为你的环境变量配置的有问题，你也可以将所在目录下的执行文件拖入cmd中，如果你使用的是git也可以直接使用keytool（我自己的是乱码，但是不影响使用，对应的输入信息是一样的）



例如："D:\Program Files\Android\jdk\jre\bin\keytool.exe" -genkeypair -alias test.keystore -keyalg RSA -validity 4000 -keystore test.keystore

将会输入一些信息：这里会要求输入两次密码，作为之后你使用签名的凭证





1. 使用签名生成带签名的apk

命令：jarsigner -verbose -keystore name.keystore -signedjar name.apk name\_unsigned.apk name.keystore

name.keystore 替换成 你所生成的签名文件 我的就是test.keystore

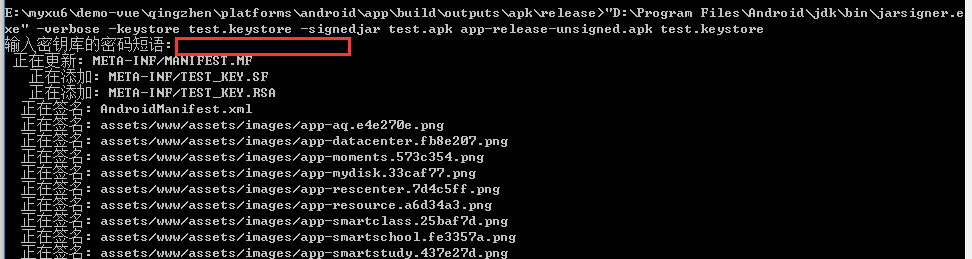
name.apk 的name替换成你所需要生成的apk的名称 我这边是test

name\_unsigned.apk 替换成未带签名的apk 我这边就是 app-release-unsigned.apk

这里需要注意，当你输入jarsigner时，可能会出现和keytool一样的问题，操作方式也是相同的

例如：“D:\Program Files\Android\jdk\bin\jarsigner.exe” -verbose -keystore test.keystore -signedjar test.apk app-release-unsigned.apk test.keystore

这里输入你所需要的密码，然后就会生成apk





1. 这里是注意

Keytool和jarsigner都是java下的工具，一个在bin文件目录下，一个在jre\bin文件目录下，如果你安装了jdk，但是你输入keytool或者jarsigner都是不可运行的程序的话，

1. 你可以直接拖入exe文件替换相应的名称文件，
2. 你也可以用管理员方式执行cmd然后输入keytool或者jarsigner，就会有相应的信息
3. 你也可以使用git打开也是可以执行的

注：后续会相继更新cordova相关内容