**web开发文档**

一.fis3打包工具（http://fex-team.github.io/fis3/docs/beginning/intro.html）

FIS3 是面向前端的工程构建工具。解决前端工程中性能优化、资源加载（异步、同步、按需、预加载、依赖管理、合并、内嵌）、模块化开发、自动化工具、开发规范、代码部署等问题。

就项目而言，打包工具能够做到资源的定位、压缩，产生文件指纹，对于发布、更新等步骤起到帮助。

1.安装node和npm（[**https://nodejs.org**](https://nodejs.org/)）

安装node和npm（新版本的node自带npm）

Windows 用户安装完成后需要在 CMD 下确认是否能执行 node 和 npm

命令：node -v npm -v

2.安装fis3（cmd下执行）

npm **install** -**g** fis3

安装完成后执行 fis3 -v 判断是否安装成功

（

升级fis3

npm update -g fis3

重装fis3

npm install -g fis3

）

3.命令使用

进入项目根目录，执行命令：

**fis3 release -d <path>**

<path> 为任意目录，即打包后文件生成的目录

全局或本地安装插件

npm install [-g] fis3-hook-relative

启用插件

fis.hook('relative');

二.接口的调用

采用cordova框架，接口调用应用以下步骤

1.  webdemo工程地址

svn://172.20.5.249:3692/Branch/TianciAndroid/OS/Task-5.0/AS\_CoocaaWebView/assets/www

将该目录下的所有文件复制到工作目录下

（开发仅仅是对index.html页面以及相对应的css和js进行更改）

2.重写html页面以及js脚本

注:

html内容需要包含在一个id = "deviceready" 的div中，在js文件中，将对其进行监听

demo中的js/index.js可作为一个完整的参考例子，包含所有的接口调用方式

重点在于对triggleButton:function()函数的重写/修改

在此之前的函数可归结为事件的监听加载过程，可不做修改

（receivedEvent: function(id)的内容根据需求进行增减）

另：

fis3打包工具能够针对png，jpg等文件产生文件指纹，并进行资源定位。

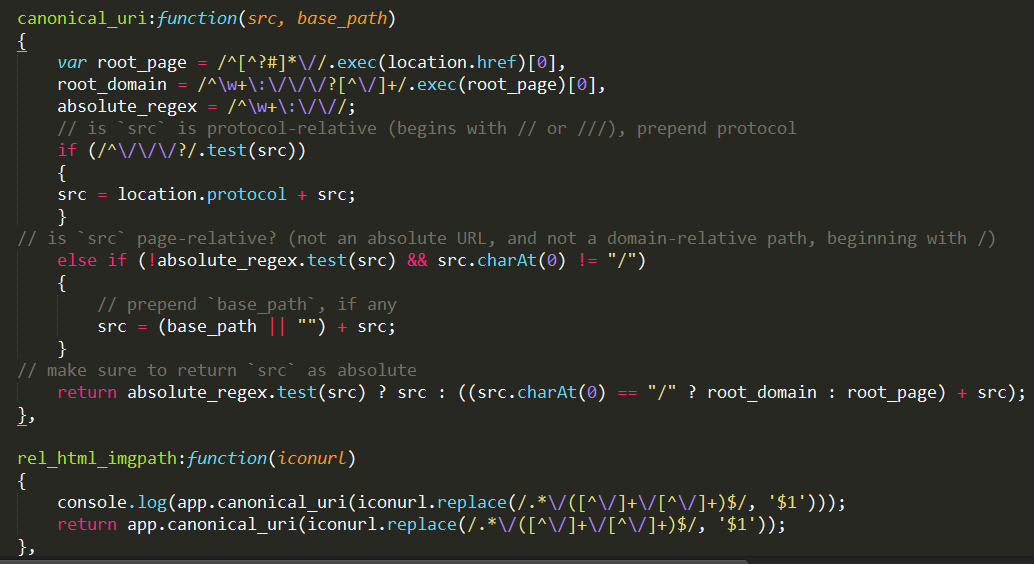
但该方式在js文件或 html中的script标签内的内容中出现的文件将不起作用，需另作处理，使用编译函数 **\_\_uri(path)** 来定位资源

例如：



在css文件中或html中style标签内的内容，则采用url(path) 以及 src=path 字段来替换

而app.rel\_html\_imgpath()函数则是对编译后的指纹进行的一个处理，代码如下：



rel\_html\_imgpath:function(iconurl)

{

console.log(app.canonical\_uri(iconurl.replace(/.\*\/([^\/]+\/[^\/]+)$/, '$1')));

return app.canonical\_uri(iconurl.replace(/.\*\/([^\/]+\/[^\/]+)$/, '$1'));

}

canonical\_uri:function(src, base\_path)

{

var root\_page = /^[^?#]\*\//.exec(location.href)[0],

root\_domain = /^\w+\:\/\/\/?[^\/]+/.exec(root\_page)[0],

absolute\_regex = /^\w+\:\/\//;

// is `src` is protocol-relative (begins with // or ///), prepend protocol

if (/^\/\/\/?/.test(src))

{

src = location.protocol + src;

}

// is `src` page-relative? (not an absolute URL, and not a domain-relative path, beginning with /)

else if (!absolute\_regex.test(src) && src.charAt(0) != "/")

{

// prepend `base\_path`, if any

src = (base\_path || "") + src;

}

// make sure to return `src` as absolute

return absolute\_regex.test(src) ? src : ((src.charAt(0) == "/" ? root\_domain : root\_page) + src);

}

在完成页面开发之后，先自行在本地服务器上进行测试，代码上传到服务器上，在电视上通过CordovaTest.apk进行测试。在确认没有问题之后，发出邮件给运维人员进行正式环境（http://webapp.skysrt.com）的部署。

**支持的接口:（建议在酷开5.5（含）以上的版本使用）**

启动本地媒体（经测试4.x的版本不支持）

调用方式：监听click函数，执行

coocaaosapi.startLocalMedia( function(message) {console.log(message); },function(error) {console.log(error);});

返回结果：打开本地媒体界面

启动电视设置 （经测试4.x的版本不支持）

调用方式：监听click函数，执行

coocaaosapi.startTVSetting(function(message) {console.log(message); },function(error){console.log(error);});

返回结果：打开电视设置界面

启动信号源 （经测试4.x的版本不支持）

coocaaosapi.startSourceList( function(message){console.log(message);},function(error){console.log(error); });

打开信号源选择界面

启动二维码 （经测试4.x的版本可以支持，但是启动的二维码为电视派下载，现不再建议使用）

启动影视历史 （经测试4.x的版本不支持）

coocaaosapi.startMovieHistory(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：打开影视历史界面

启动我的游戏 （经测试4.x的版本不支持）

coocaaosapi.startMyGames(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：打开我的游戏界面

启动我的应用 （经测试4.x的版本不支持）

coocaaosapi.startMyApps("",function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：启动我的应用界面

启动儿童我的应用 （儿童模式在5.5版本中正式启用，在此之前版本不支持）

coocaaosapi.startMyApps("child",function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：启动儿童模式下我的应用

启动给用户设置

coocaaosapi.startUserSetting(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：启动用户账户中心

启动网络设置

coocaaosapi.startNetSetting(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：打开网络设置界面

启动蓝牙设置

coocaaosapi.startBlueToothSetting(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：打开蓝牙设置界面

启动消息盒子 （酷开5.5版本以上才具有消息盒子功能）

coocaaosapi.startMessageBox(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：打开消息盒子

启动系统升级

coocaaosapi.startSystemUpgrade(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：打开系统的升级页面

获取定位信息

coocaaosapi.getDeviceLocation(function(message) {

console.log("location " + message.location);

document.getElementById('locationid').value = JSON.stringify(message);

},function(error) { console.log(error);})

返回结果：设备定位信息的json串

获取accesstoken

coocaaosapi.getUserAccessToken(function(message) {

console.log("usertoken " + message.accesstoken);

document.getElementById('usertokenid').value = JSON.stringify(message);

},function(error) { console.log(error);})

返回结果：用户的token值

影视相关：

id跳转影视列表

coocaaosapi.startMovieList(listid,function(message) {console.log(message); },function(error) { console.log(error);});

参数listid为输入/获取到的影视id

返回结果：跳转打开该id的影视列表

id跳转影视详情

coocaaosapi.startMovieDetail(detailid,function(message) {console.log(message); },function(error) { console.log(error);});

参数detailid为输入/获取到的影视详情id

返回结果：跳转打开该id的影视详情

id跳转影视专题

coocaaosapi.startMovieTopic(topicid,function(message) {console.log(message); },function(error) { console.log(error);});

参数topicid为输入/获取到的影视专题的id

返回结果：跳转打开该id对应的影视专题页

启动影视会员中心

coocaaosapi.startMovieMemberCenter('qq',function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：启动打开影视会员中心

启动影视主页

coocaaosapi.startMovieHome(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：打开影视主页面

获取设备信息

coocaaosapi.getDeviceInfo(function(message)

{ },function(error) { console.log(error);})

返回结果：

包含有屏幕尺寸（panel）、机芯（chip）、机型（model）、mac地址、安卓版本（androidsdk）、酷开版本（version）、激活id（activeid）、电视id（devid）、chipid等信息的json串

获取网络连接状态

coocaaosapi.isNetConnected(function(message)

{ },function(error) { console.log(error);})

返回结果：

true（联网状态）/false（无网络状态）

获取网络连接类型

coocaaosapi.getNetType(function(message)

{ },function(error) { console.log(error);})

返回结果：

WIFI / Ethernet

获取网络ip地址

coocaaosapi.getIpInfo(function(message) {

console.log("dns0 " + message.dns0);

console.log("dns1 " + message.dns1);

console.log("gateway " + message.gateway);

console.log("ip " + message.ip);

console.log("mac " + message.mac);

console.log("netmask " + message.netmask);

},function(error) { console.log(error);})

返回结果：

json串，包括网关、mac、ip等信息

应用圈相关：

启动应用圈

coocaaosapi.startAppStore(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：

打开应用圈

启动应用圈榜单

coocaaosapi.startAppStoreBD(1,function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：跳转打开应用圈榜单页面

id跳转应用圈分类

coocaaosapi.startAppStoreSort(sortid,function(message) {console.log(message); },function(error) { console.log(error);});

参数sortid为输入/获取到的应用圈分类id

返回结果：

跳转到该id对应的应用圈分类页面

id跳转应用圈列表

coocaaosapi.startAppStoreList(listid,function(message) {console.log(message); },function(error) { console.log(error);});

参数listid为输入/获取到的应用圈列表id

返回结果：跳转到该id对应的应用圈列表

id跳转应用圈详情

coocaaosapi.startAppStoreDetail(detailid,function(message) {console.log(message); },function(error) { console.log(error);});

参数detailid为输入/获取到的应用圈详情id

返回结果：跳转到该id对应的应用圈详情页面

id跳转应用圈专题

coocaaosapi.startAppStoreZone(zoneid,function(message) {console.log(message); },function(error) { console.log(error);});

参数zoneid为输入/获取到的应用圈专题id

返回结果：跳转到该id对应的应用圈专题页面

下载一个任务

coocaaosapi.startOrCreateDownloadTask(

"https://qd.myapp.com/myapp/qqteam/AndroidQQ/mobileqq\_android.apk",

'',

'qq移动版',

'com.tencent.mobileqq',

'123123', 'http://img.zcool.cn/community/01559e565d84d832f875964706920d.png',

function(message) {console.log(message); },

function(error) { console.log(error);});

参数：

1.下载路径

2.MD5校验值

3.下载文件名称

4.下载的包名

5.appid

6.icon图片地址

返回结果：

在应用圈中进行下载

启动酷游吧

coocaaosapi.startGameCenter(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：打开酷游吧界面

启动游戏军火库

coocaaosapi.startGameArsenal(function(message) {console.log(message); },function(error) { console.log(error);});

返回结果：打开酷游吧游戏军火库页面

id跳转酷游吧列表

coocaaosapi.startGameCenterList(gamelistid,gametitleid,function(message) {console.log(message); },function(error) { console.log(error);})

参数：gamelistid为酷游吧列表id，gametitleid为名称

返回结果：打开该id对应的酷游吧列表

用户是否登录

coocaaosapi.hasCoocaaUserLogin(function(message) {

},function(error) { console.log(error);})

返回结果：

true：已经登录 false：未登录

获取用户信息

coocaaosapi.getUserInfo(function(message) {

console.log("address " + message.address);

console.log("avatar " + message.avatar);

console.log("avatars " + message.avatars);

console.log("birthday " + message.birthday);

console.log("city " + message.city);

console.log("corp " + message.corp);

console.log("district " + message.district);

console.log("education\_grade " + message.education\_grade);

console.log("email " + message.email);

console.log("gender " + message.gender);

console.log("idcard " + message.idcard);

console.log("line " + message.line);

console.log("mobile " + message.mobile);

console.log("nick\_name " + message.nick\_name);

console.log("occupation " + message.occupation);

console.log("open\_id " + message.open\_id);

console.log("oss\_id " + message.oss\_id);

console.log("postcode " + message.postcode);

console.log("province " + message.province);

console.log("qq " + message.qq);

console.log("region " + message.region);

console.log("region\_id " + message.region\_id);

console.log("revenue " + message.revenue);

console.log("score " + message.score);

console.log("sky\_id " + message.sky\_id);

console.log("skype " + message.skype);

console.log("slogan " + message.slogan);

console.log("tel1 " + message.tel1);

console.log("tel2 " + message.tel2);

console.log("visit\_num " + message.visit\_num);

console.log("wechat " + message.wechat);

console.log("weibo " + message.weibo);

console.log("balance " + message.balance);

console.log("third\_account " + message.third\_account);

console.log("external\_info " + message.external\_info);

document.getElementById('userinfoid').innerHTML = JSON.stringify(message);

},function(error) { console.log(error);})

返回结果：当前用户的信息json串

启动支付订单

coocaaosapi.purchaseOrder('1001',math+'','包月','product detail','虚拟',{'notify\_url':'http://42.121.113.121:8090/aqiyiOrder/viewMain.html'},0.01,0,'','',

function(success)

{

},

function(error)

{

console.log(error);

});

web兼容性问题

测试多台电视，屏幕分辨率有1920\*1080 与 1280\*720两种

可见宽高发现有980\*551 1050\*591

去除head头里面

<meta name="viewport" >允许进行缩放

则按照1920\*1080分辨率进行页面制作，可自适应不同电视

最外层div设置大小1920\*1080（16:9）

首先适应宽度。若最外层宽小于可见宽，则页面在电视上必然无法全屏，反之必为全屏

若最外层宽大于可见宽，则首先横向全屏，但若宽高比不为16:9，则会出现：

1.不足，则高度无法全屏

2.超过，则纵向可滚动