uexWidget
管理当前应用。

方法
startWidget 加载一个 widget
finishWidget 退出一个 widget
removeWidget 删除一个 widget
checkUpdate 检查更新
loadApp 启动第三方应用（iOS）
startApp 启动第三方应用（Android）
getOpenerInfo 获取 widget 的相关信息。
installApp 安装 apk
getPushInfo 获取推送消息
setPushNotifyCallback 设置推送消息到达时的回调函数
setPushInfo 设置推送用户信息
setPushState 设置推送服务的状态
getPushState 获取推送服务的状态

回调方法
cbStartWidget 加载 widget 完成时的回调方法
cbRemoveWidget 删除 widget 完成时的回调方法
cbCheckUpdate 检查更新完成时的回调方法
cbGetOpenerInfo 获取 widget 相关信息的回调方法
cbGetPushInfo 获取推送消息的回调方法
cbGetPushState 获取推送状态的回调方法

监听方法
onSuspend 程序挂起的监听方法
onResume 程序恢复的监听方法
onTerminate 程序终止的监听方法

startWidget

加载一个 widget
uexWidget.startWidget(appId, animiId, funName, info, animDuration)
说明：
在当前 widget 加载一个子 widget。
参数：
appId: *(String 类型)必选* 子 widget 的 appId
animiId: *(Number 类型)必选* 子 widget 载入时的动画 id,详见 CONSTANT 中 Window Animi ID
funName: *(String 类型)必选* 方法名，子 widget 结束时将 String 型的任意字符回调给该方法，可为空。注意：只在主窗口中有效，浮动窗口中无效

info: *(String 类型)必选* 传给子 widget 的信息

animDuration: *(Number 类型)必选* 动画持续时长，单位为毫秒，默认 200 毫秒

平台支持：

Android 2.2+

iOS 4.3+

版本支持：

3.0.0+

示例：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script type="text/javascript">
    function startWidget(appId){
            uexWidget.startWidget(appId, "2", "pluginCallback", "plugin  插件调用");
    }
    function pluginCallback(resultInfo){
        alert("pluginCallback"+resultInfo);
    }
    function cbStartWidget(opId, dataType,data){
        alert(data);
    }
    window.uexOnload = function() {
        uexWidget.cbStartWidget = cbStartWidget;
    }
</script>
  </head>
<body>
    <input class="btn" type="button" value="加载 widget" onclick=startWidget("123456789")>
</body>
</html>
```

finishWidget

退出一个 widget

uexWidget.finishWidget(resultInfo, appId, isWgtBG)

说明：

退出一个 widget。

参数：

resultInfo: *(String 类型)必选* 此 widget 结束时，传递给 opener 的信息。

appId: *(String 类型)可选* 要结束的 widget 的 appId，为空时退出的是当前的 widget。

isWgtBG: *(Number 类型)可选* 结束此 widget 的方式，0 表示销毁该 widget，下次再调用 startWidget 时，重新打开；1 表示把该 widget 置于后台，下次再调用 startWidget 时，不重新

打开，操作数据全部保存。不传或为空时，默认为 0。注意传该参数时，必须要传 appId 参数。
平台支持：
Android 2.2+
iOS 4.3+
版本支持：
3.0.0+
示例：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script type="text/javascript">
    function finishWidget(){
        uexWidget.finishWidget("uexWidget.finishWidget");
    }
   </script>
</head>
<body class="um-vp" ontouchstart>
    <input class="btn" type="button" value="结束 widget" onclick=finishWidget()>
</body>
</html>
```

removeWidget

删除一个 widget
uexWidget.removeWidget(appId)
说明：
删除一个 widget。
参数：
appId: *(String 类型)必选* widget 的 appId，主 widget 不能被删除。
平台支持：
Android 2.2+
iOS 4.3+
版本支持：
3.0.0+
示例：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script type="text/javascript">
    function delWidget(appId){
```

```
        uexWidget.removeWidget(appId);
    }
    function cbRemoveWidget(opCode, dataType, data){
        if(data == 0){
            alert("删除成功！");
        }else{
            alert("删除失败！");
        }
    }
    window.uexOnload = function() {
        uexWidget.cbRemoveWidget = cbRemoveWidget;
    }
</script>
</head>
<body class="um-vp" ontouchstart>
        <input class="btn" type="button" value="删除widget"
onclick=delWidget("11111111111")>
</body>
</html>
```

checkUpdate

检查更新

uexWidget.checkUpdate()

说明：

检查当前 widget 是否有更新。

参数：

无

平台支持：

Android 2.2+

iOS 4.3+

版本支持：

3.0.0+

示例：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script type="text/javascript">
    function cbCheckUpdate(opCode, dataType, jsonData){
        alert(jsonData);
    }
    window.uexOnload = function() {
```

```
            uexWidget.cbCheckUpdate = cbCheckUpdate;
        }
</script>
</head>
<body class="um-vp" ontouchstart>
        <input class="btn" type="button" value="检查widget更新"
onclick="uexWidget.checkUpdate();">
</body>
</html>
```

loadApp

启动第三方应用
uexWidget.loadApp(appInfo)
说明：
根据相关信息启动一个第三方应用。
参数：
appInfo: *(String 类型)必选* 第三方应用的 URL Schemes。
平台支持：
iOS 4.3+
版本支持：
3.0.0+
示例：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script type="text/javascript">
        function loadappo(){
        var appInfo = "http://www.baidu.com";
        uexWidget.loadApp(appInfo);
    }
</script>
</head>
<body>
    <input class="btn" type="button" onclick="loadappo()" value="启动第三方应用(iOS)"/>
</body>
</html>
```

startApp

启动第三方应用
uexWidget.startApp(startMode,mainInfo, addInfo, optInfo)

说明：

根据相关信息启动一个第三方应用。

参数：

startMode: *(String 类型)必选* 启动方式，0 表示通过包名和类名启动，1 表示通过 Action 启动。

startMode 为 0：

mainInfo: *(String 类型)必选* 包名

addInfo: *(String 类型)可选* 类名，为空时启动应用入口类

startMode 为 1：

mainInfo: *(String 类型)必选* action

addInfo: *(String 类型)可选* category 或 data，json 格式如下：

{"category":["android.intent.category.WID","android.intent.category.WID1"],"data":{"mimeType":"image/png","scheme":"sip"}}

各字段含义如下:

参数	是否必须	说明
category	否	category 属性
data	否	data 属性
mimeType	否	mineType 属性
scheme	否	scheme 属性

optInfo: *(String 类型)可选* 附加参数，键值对，{key: value}格式，多个用英文","隔开，如： "{'key1': 'value1'},{'key2': 'value1'}"

平台支持：

Android 2.2+

版本支持：

3.0.0+

示例：

1.要启动的 AndroidManifest.xml 文件如下：

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.dongjf.mytest"//com.dongjf.mytest 即为包名
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-sdk
        android:minSdkVersion="11"
        android:targetSdkVersion="18" />
    <application
        android:allowBackup="true"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="MainActivity"
            android:label="@string/app_name"
            android:icon="@drawable/icon1"
```

```
            android:windowSoftInputMode="stateHidden|adjustResize"
             >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
            <intent-filter>
                <action android:name="com.djf.test.main" />
                <category android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity android:name="com.dongjf.mytest.SecondActivity">
            //通过包名启动，<intent-filter>标签非必须
            <intent-filter>
                <action android:name="com.djf.test.second" />
                //通过 action 启动时， category DEFAULT 属性必须，否则无法调起
                <category android:name="android.intent.category.DEFAULT" />
                <data android:scheme="tel"/>
            </intent-filter>
        </activity>
        <activity android:name="com.dongjf.mytest.ThirdActivity">
            <intent-filter>
                <action android:name="com.djf.test.second" />
                <category android:name="android.intent.category.DEFAULT" />
                <category android:name="android.intent.category.WID" />
                <category android:name="android.intent.category.WID1" />
                <data android:scheme="sip" android:mimeType="image/png"/>
            </intent-filter>
        </activity>
        <activity    android:name="com.dongjf.mytest.ForthActivity">
            <intent-filter>
                <action android:name="com.djf.test.second" />
                <category android:name="android.intent.category.WID" />
                <category android:name="android.intent.category.DEFAULT" />
                <data android:mimeType="text/html"/>
            </intent-filter>
        </activity>
    </application>
</manifest>
```

2.启动该应用对应界面示例如下：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
```

```
    <meta name="viewport" content="target-densitydpi=device-dpi, width=device-width, initial-scale=1, user-scalable=no, minimum-scale=1.0, maximum-scale=1.0">
    <style>
        input{
            margin:.4em;
            font-size: 1.2em !important;
        }
        input[type="text"]{
            color: black;
        }
        p{
            font-size: 14px !important;
        }
    </style>
  <script type="text/javascript">
    function startAppP(mode){
        var main,add,opt;
        main = "com.dongjf.mytest";
        switch(mode){
            case 0:
                uexWidget.startApp(0, main);//启动应用的入口主类
                break;
            case 1:
                add = "com.dongjf.mytest.SecondActivity";
                uexWidget.startApp(0, main, add);//启动应用的 SecondActivity 类
                break;
            case 2:
                add = "com.dongjf.mytest.ThirdActivity";
                opt = "{'key1': 'value1'},{'key2': 'value2'}";
                uexWidget.startApp(0, main, add, opt);//启动应用的 ThirdActivity 类
                break;
        }
    }
    function startAppA(mode){
        var main,add,opt;
        switch(mode){
            case 0:
                main = "com.djf.test.main";
                uexWidget.startApp(1, main);//启动应用的 MainActivity 类
                break;
            case 1:
                main = "com.djf.test.second";
                add = '{"data":{"scheme":"tel"}}';
                uexWidget.startApp(1, main, add);//启动应用的 SecondActivity 类
```

```
                    break;
                case 2:
                    main = "com.djf.test.second";
                    add = '{"data":{"mimeType":"text/html"}}';
                    opt = "{'key1': 'value1'},{'key2': 'value2'}";
                    uexWidget.startApp(1, main, add, opt);//启动应用的 ForthActivity 类
                    break;
                case 3:
                    main = "com.djf.test.second";
                    add = '{"data":{"mimeType":"image/png","scheme":"sip"}}';
                    opt = "{'key1': 'value1'},{'key2': 'value2'}";
                    uexWidget.startApp(1, main, add, opt);//启动应用的 ThirdActivity 类
                    break;
                case 4:
                    main = "com.djf.test.second";
                    add =
'{"category":["android.intent.category.WID","android.intent.category.WID1"],"data":{"mimeType
":"image/png","scheme":"sip"}}';
                    uexWidget.startApp(1, main, add);//启动应用的 ThirdActivity 类
                    break;
            }
        }
        </script>
    </head>
<body class="um-vp" ontouchstart>
        <input class="btn" type="button" value="通过包名启动 0" onclick=startAppP(0)>
        <input class="btn" type="button" value="通过包名启动 1" onclick=startAppP(1)>
        <input class="btn" type="button" value="通过包名启动 2" onclick=startAppP(2)>
        <br><br>
        <input class="btn" type="button" value="通过 Action 启动 0" onclick=startAppA(0)>
        <input class="btn" type="button" value="通过 Action 启动 1" onclick=startAppA(1)>
        <input class="btn" type="button" value="通过 Action 启动 2" onclick=startAppA(2)>
        <input class="btn" type="button" value="通过 Action 启动 3" onclick=startAppA(3)>
        <input class="btn" type="button" value="通过 Action 启动 4" onclick=startAppA(4)>
</html>
```

getOpenerInfo

获取 widget 的相关信息
uexWidget.getOpenerInfo()
说明：
　　获取打开者传入此 widget 的相关信息。即调用 startWidget 时传入的 info 参数值。
参数：

无
平台支持：
Android 2.2+
iOS 4.3+
版本支持：
3.0.0+
示例：
```
<!DOCTYPE html>
<html>
  <head>
    <script type="text/javascript">
    window.uexOnload = function(){
        uexWidget.cbGetOpenerInfo = cbGetOpenerInfo;
    }
    function cbGetOpenerInfo(opCode,dataType,data){
        alert(data);
    }
    </script>
  </head>
  <body class="um-vp " ontouchstart>
    <input   class="btn" type="button" value="获取" onclick="uexWidget.getOpenerInfo();">
  </body>
</html>
```

installApp

安装 apk
uexWidget.installApp(appPath)
说明：
根据安装包所在路径安装一个 apk。
参数：
appPath: *(String 类型)必选* apk 所在路径，支持的路径协议，详见 CONSTANT 中的 Path Types
平台支持：
Android 2.2+
版本支持：
3.0.0+
示例：
```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script type="text/javascript">
    function install(){
```

```
            var path = "res://pay.apk";
            uexWidget.installApp(path);
        }
</script>
</head>
<body>
        <input class="btn" type="button" value="安装" onclick=install()>
</body>
</html>
```

getPushInfo

获取推送消息
uexWidget.getPushInfo()
说明：
　　获取推送消息,上报消息到管理后台
参数：
　　无
平台支持：
　　Android 2.2+
　　iOS 4.3+
版本支持：
　　3.0.0+
示例：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script type="text/javascript">
        function cbGetPushInfo(opCode,dataType,data){
            alert(data);
        }
        window.uexOnload = function() {
            uexWidget.cbGetPushInfo = cbGetPushInfo;
       }
</script>
</head>
<body class="um-vp" ontouchstart>
    <input class="btn" type="button" value="获取推送消息数据 "
onclick="uexWidget.getPushInfo();">
</body>
</html>
```

setPushNotifyCallback

设置 Push 消息到达时的回调函数
uexWidget.setPushNotifyCallback(cbFunction)
说明：
　　如果应用开启了推送功能，那么当有消息推送进来时，平台将调用指定的 cbFunction 函数通知页面 。
参数：
　　cbFunction:　*(String 类型)必选* 方法名
平台支持：
　　Android 2.2+
　　iOS 4.3+
版本支持：
　　3.0.0+
示例：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script type="text/javascript">
        function pushCallback(){
         alert("收到推送消息");
        }
    window.uexOnload = function() {
          uexWidget.setPushInfo('bbb','姓名');
          uexWidget.setPushNotifyCallback('pushCallback');
    }
</script>
</head>
</html>
```

setPushInfo

设置推送用户信息
uexWidget.setPushInfo(uId, uNickName)
说明：
　　设置推送用户信息
参数：
　　uId: *(String 类型)必选* 用户 ID
　　uNickName : *(String 类型)必选* 用户昵称
平台支持：

Android 2.2+
iOS 4.3+

版本支持：

3.0.0+

示例：

见 setPushNotifyCallback 中的示例。

setPushState

设置推送服务的状态

uexWidget.setPushState(state)

说明：

设置推送服务的状态

参数：

state: *(Number 类型)必选* 推送服务状态。0，关闭。1，开启。

平台支持：

Android 2.2+
iOS 4.3+

版本支持：

3.0.0+

示例：

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <script type="text/javascript">
    function cbGetPushState(opId, dataType, data){
        alert(data);
    }
    function ss(){
        uexWidget.setPushState(0);
        uexWidget.getPushState();
    }
</script>
</head>
<body>
    <input class="btn" type="button" onclick="ss()" value="设置消息推送状态"/>
</body>
</html>
```

getPushState

获取推送服务的状态
uexWidget.getPushState()
说明：
　　获取推送服务的状态
参数：
　　无
平台支持：
　　Android 2.2+
　　iOS 4.3+
版本支持：
　　3.0.0+
示例：
　　见 setPushState 中的示例

cbStartWidget

加载 widget 完成时的回调方法
　　uexWidget.cbStartWidget(opId, dataType, data)
参数：
　　opId:　*(Number 类型)必选* 操作 ID，在此函数中不起作用，可忽略
　　dataType:　*(Number 类型)必选* 参数类型 详见 CONSTANT 中 Callback 方法数据类型
　　data: *(Number 类型)必选*　返回 uex.cSuccess 或 uex.cFailed，详见 CONSTANT 中 Callback int 类型数据
版本支持：
　　3.0.0+

cbRemoveWidget

删除 widget 完成时的回调方法
　　uexWidget.cbRemoveWidget(opId, dataType, data)
参数：
　　opId:　*(Number 类型)必选* 操作 ID，在此函数中不起作用，可忽略
　　dataType:　*(Number 类型)必选* 参数类型 详见 CONSTANT 中 Callback 方法数据类型
　　data: *(Number 类型)必选*　返回 uex.cSuccess 或 uex.cFailed，详见 CONSTANT 中 Callback int 类型数据
版本支持：
　　3.0.0+

cbCheckUpdate

检查更新完成时的回调方法
　　uexWidget.cbCheckUpdate(opId, dataType, data)
参数：

opId: *(Number 类型)必选* 操作 ID，在此函数中不起作用，可忽略

dataType: *(Number 类型)必选* 参数类型 详见 CONSTANT 中 Callback 方法数据类型

data: *(Number 类型)必选* 返回 uex.jvUpdate、uex.jvNoUpdate 或 uex.jvError，详见 CONSTANT 中 Callback int 类型数据

版本支持：

3.0.0+

cbGetOpenerInfo

获取 widget 相关信息的回调方法

uexWidget.cbGetOpenerInfo(opId, dataType, data)

参数：

opId: *(Number 类型)必选* 操作 ID，在此函数中不起作用，可忽略

dataType: *(Number 类型)必选* 参数类型 详见 CONSTANT 中 Callback 方法数据类型

data: *(String 类型)必选* 返回的数据。本 widget 的打开者通过 startWidget 函数打开本 widget 时传入的 info 参数值。

版本支持：

3.0.0+

cbGetPushInfo

获取推送消息的回调方法

uexWidget.cbGetPushInfo(opId, dataType, data)

参数：

opId: *(Number 类型)必选* 操作 ID，在此函数中不起作用，可忽略

dataType: *(Number 类型)必选* 参数类型 详见 CONSTANT 中 Callback 方法数据类型

data: *(String 类型)必选* 返回的数据。根据 iOS 平台推送的特殊性，推送服务器发出的推送的 json 格式为：{"aps":{"alert":"你收到推送.","badge":1,"sound":"sound.wav","userInfo”:"接到的参数”}}

各字段含义如下:

参数	是否必须	说明
alert	是	推送消息的标题
badge	是	应用图标上显示的通知数
sound	是	收到推送消息的声音文件
userInfo	是	推送收到的数据

版本支持：

3.0.0+

cbGetPushState

获取推送状态的回调方法

uexWidget.cbGetPushState(opId, dataType, data)

参数：

opId: *(Number 类型)必选* 操作 ID，在此函数中不起作用，可忽略

dataType: *(Number 类型)必选* 参数类型 详见 CONSTANT 中 Callback 方法数据类型

data: *(Number 类型)必选* 返回的数据。1: 推送服务的状态是开启的。0: 推送服务的状态是关闭的。

版本支持：

3.0.0+

onSuspend

程序挂起的监听方法

uexWidget.onSuspend()

参数：

无

版本支持：

3.0.0+

onResume

程序恢复的监听方法

uexWidget.onResume()

参数：

无

版本支持：

3.0.0+

onTerminate

程序终止的监听方法

uexWidget.onTerminate()

参数：

无

版本支持：

3.0.0+