

# iPhone WebApp

## 开发指南

作者：流螢飛舞



WeiPhone Tech Team

## 目录

一、前言	2
二、入门	3
三、框架	5
四、属性	8
五、事件	12
六、特性	14
七、CSS	15
八、封装	20
九、后记	23

▲ 本文所有蓝色文字为标准语句；紫色文字为标准标识字。

▲ iPhone Safari 对字符大小写具有敏感性，因此必须注意文件名的格式，尤其不要使用中文作为文件名。

☆ 感谢 [hhytt](#) 精心编写的外壳程序，这使得 WebApp 更加规范、便捷的安装成为可能。

☆ 感谢威锋网技术组 ( WeiPhone Tech Team ) 各位同仁的帮助！尤其是 [飘sir](#) 超版对本文给予了大力支持，并审核了全文。在此致以诚挚的谢意。



## 一、前言

Apple iPhone 推出以后，以其动人的外观、超强的性能和丰富的软件资源吸引了众多的拥趸者。iPhone 的原始出厂功能比较基本，仅能完成一般的电话、短信、时钟、邮件、上网以及定位的较少的功能。然而，其基于 Apple Mac OS X 的操作系统，为用户提供了编写应用软件的可能。目前仅在 AppStore 上架的软件就已经达到近十万个。可以形象的比喻，原始的 iPhone 像个刚买来的电脑，但众多的第三方软件使其可以完成几乎任何你可以想象到的功能。

你的创意总是独特的，或许那些收费的或免费的软件不能满足你的愿望，您也想试图编写一些应用程序来实现你的需求。那么，我可以告诉你，这点上，很难、也很容易。

**难：**Apple 公司为企图开发 iPhone 应用软件的公司和个人提供了开发所必须的软件开发工具包 SDK ( Software Develoment Kit )，然而，使用它，必须先向 Apple 注册申请，开发的软件要提交 Apple Store 审核，然后才能上架发布。虽然现在已经出现了盗版的 SDK，可以免注册在 MAC 或 Windows 环境下进行开发，但其开发环境搭建的复杂，以及基于 Object-C 相对较难的编程语言，也阻挡的很多爱好者试图开发程序的脚步。

**易：**所幸的是，Apple 为我们提供了另一条简洁的开发途径，也就是在基于 WebKit 的 Safari 上，用户可以编写自己所需的 WebApp，也就是网络应用软件。简单的说，应用程序可严格编写为服务器上的 HTML、CSS 和 JavaScript 文件，实现大多数可以实现的功能需求。Web 开发和 SDK 开发是两种截然不同、各有利弊的开发方式。

WebKit 是一种浏览器引擎，支撑着 iPhone 内的 Mobile Safari 浏览器背后的技术。WebKit 是一个开源项目，它优先支持 HTML 和 CSS 特性。实际上，WebKit 还支持尚未被其他浏览器采纳的一些诸如 HTML5 规范 CSS 样式。

iPhone 上的 Safari 支持的标准：

- ✓ HTML 4.01
- ✓ XHTML 1.0
- ✓ CSS 2.1 以及部分 CSS 3
- ✓ JavaScript (ES3)
- ✓ DOM (Level 2)
- ✓ AJAX (XMLHttpRequest)

熟悉这些标准并且平常也坚持 Web Standards 实践的朋友估计要笑出来了——就这些吗？我们天天在用啊，还有必要专门写文章来说明吗？事实上，Safari 之前作为一款无 PC 版的浏览器，一直用户数量就不高，因此对它的研究也就不多，然而 Safari 其实有不少自己的扩展，因此还是很值得研究的。既然我们是针对 iPhone 设置，其实就是针对 Safari 设计，无需考虑兼容其它浏览器，这时候为什么不好好利用这些扩展增强自己的应用程序的可用性呢？

虽然目前 iPhone WebApp 开发已经有诸如 iUI、Canvas、Dashcode 等很好的开发工具，但了解掌握最基本的 html、javascript、css 知识仍是十分必要的。本文竭力提供通俗易懂的内容，以大量的例子入手，突出特性，适合有一定 Web 开发基础，并试图开发 iPhone WebApp 开发的人士参考。

## 二、入门

1、首先要说的就是 **viewport**，也就是可视区域。对于桌面浏览器，我们都很清楚 viewport 是什么，就是出去了所有工具栏、状态栏、滚动条等等之后用于看网页的区域，这是真正有效的区域。

对于传统 Web 页面直接在 iPhone 上面显示来说是很好的事情，因为如果传统 Web 页面在 980 宽度的桌面浏览器 viewport 中显示正常的话，iPhone 上显示也绝对正常。然而这对于 Web 应用程序来说则不是好事，因为我们需要按照 980 宽度来设计将来会以 320 宽度显示的页面，一个应该显示为 320\*80 的元素，必须设计为 980\*245，这多麻烦！因此我们需要改变 viewport。

实际上应该怎么做呢？我们有几个选择，因此先让我们看看到底我们能够设置哪些属性吧。我们可以操作的属性有 4 个：

- **width** - viewport 的宽度（范围从 200 到 10,000，默认为 980 像素）
- **height** - viewport 的高度（范围从 223 到 10,000）
- **initial-scale** - 初始的缩放比例（范围从 >0 到 10）
- **minimum-scale** - 允许用户缩放到的最小比例
- **maximum-scale** - 允许用户缩放到的最大比例
- **user-scalable** - 用户是否可以手动缩放

这 6 个属性，我们可以设置其中的一个或者多个，iPhone 会根据你设置的属性自动推算其他属性值，而非直接采用默认值。这点很重要，在完全不设置的时候有默认 viewport，在你设置一个属性后其它值是自动推算出来的，不再是默认的。

如果你把

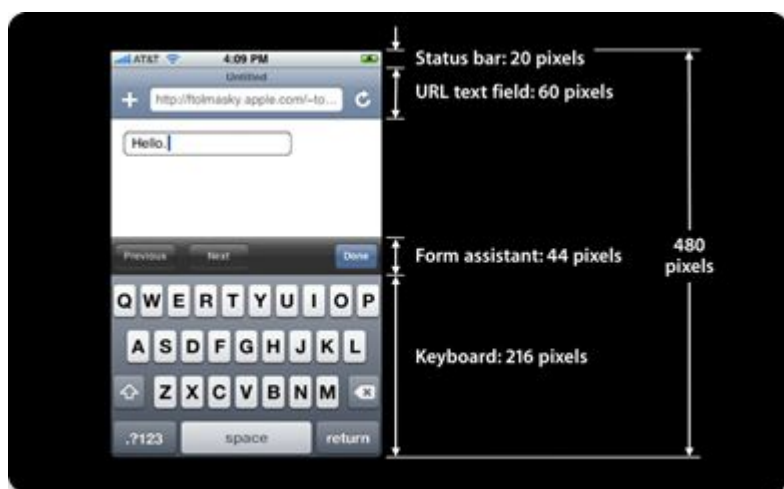
**initial-scale=1**，那么 **width** 和 **height** 在竖屏时自动为 320\*356（不是 320\*480 因为地址栏等都占据空间），横屏时自动为 480\*208。

类似地，如果你仅仅设置了 **width**，就会自动推算出 **initial-scale** 以及 **height**。例如你设置

了 **width=320**，竖屏时 **initial-scale** 就是 1，横屏时则变成 1.5 了。

那么到底这些设置如何让 Safari 知道？其实很简单，就一个 meta，形如：

```
<meta id="viewport" name="viewport" content="width=320; initial-scale=1.0; maximum-scale=1.0; user-scalable=0;" />
```



在设置了 **initial-scale=1** 之后，我们终于可以以 1:1 的比例进行页面设计了。

关于 viewport，还有一个很重要的概念是：iphone 的 safari 浏览器完全没有滚动条，而且不是简单的“隐藏滚动条”，是根本没有这个功能。iphone 的 safari 浏览器实际上从一开始就完整显示了这个网页，然后用 viewport 查看其中的一部分。当你用手指拖动时，其实拖的不是页面，而是 viewport。

浏览器行为的改变不止是滚动条，交互事件也跟普通桌面不一样。这在后面会进行详细说明。

2、必须注意到，为了让你的页面更好地受到 Safari 的支持。必须：

- ◆ 声明正确的 doctype；
- ◆ 避免使用 frameset；
- ◆ 每一个独立的资源文件，HTML、CSS、JavaScript、以及非流媒体的其他多媒体文件，限制在 10m 之内；
- ◆ 顶级入口的 JavaScript 执行时间限制为 5 秒，超时将自动终止；
- ◆ JavaScript 分配内存上限为 10m；
- ◆ 同一时间最多在 Safari 内打开 8 个子窗口（同时浏览的页面）。

Safari 本身还对图片有如下的限制：

- ◆ GIF（包括 GIF 动画）、PNG 与 TIFF 解压后的体积小于 2m。意思是，原图的长度乘以宽度再乘以每一个像素的位数，得出来的大小要小于 2m；
- ◆ JPEG 解压后最大的体积是 32m。解压体积大于 2m 的 JPG 会被进行二次抽样，最终显示给用户的是二次抽样后的结果。显示时实际上是降低了精度的，以提高程序的执行效率。

可以看出 iPhone 对 HTML 的支持与桌面端的 Safari 是类似的，只是加入了更多扩展功能而已。使用 HTML 作为框架，适当嵌入 javascript，灵活运用 CSS，即可实现你编写 iPhone 应用软件的快乐梦想。

3、HTML 的基本结构。很多人对 HTML 已经很熟了。在此仅做一简单复习。

```
<html>
<head>
  <meta ...../>
  <link ...../>
  <title>.....</title>
  <script type="text/javascript" src="xxx.js"></script>
  <style type="text/css">.....</style>
</head>
<script Language="JavaScript">.....</script>
<body>.....</body>
</html>
```

下面，侧重 iPhone Safari 的特性，详细介绍其各部分的内容。

## 三、框架

基于 WebKit 的 iPhone Safari，有一些与电脑上的 Safari，特别是与 IE(Internet Explorer) 不同之处，以下的说明中会特别注明。

### 1、头<head>部分：

#### ①<meta>

定义网页语言：（如果使用了扩展字符，请选用 gb18030）

```
<meta http-equiv="Content-Type" content="text/html; charset=gb2312"/>
```

定义用户界面，宽度为设备宽度（等同于 320 像素），并不可缩放）

```
<meta name="viewport" content="width=device-width, user-scalable=no"/>
```

定义显示的数字串，不被自动识别为电话号码。否则会自动转换为拨号超链接。

```
<meta name="format-detection" content="telephone=no" />
```

定义开发者

```
<meta name="Author" content="LSi"/>
```

#### ②<link>

web app 可以像原生应用一样，在 home 界面里上添加一个快捷方式图标

```
<link rel="apple-touch-icon" href="icon.png" />
```

这是针对单个页面的，如果你把图片命名为“apple-touch-icon.png”，放在网站的某个目录里面，该目录下所有页面都会获得添加图标的功能.....

图标的要求是尺寸 57×57，png 格式，不用画蛇添足的去圆角渐变或玻璃反光效果，iphone 系统会把图片自动裁剪和渲染成统一的风格.....

#### ③<title>

定义可被搜索的网页关键词

```
<title>择吉老黄历</title>
```

#### ④<script>

定义外部 javascript 文件

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

#### ⑤<style>

定义所用的 CSS（层叠样式表 Cascading Style Sheet）

```
<style type="text/css">.....</style>
```

iPhone Safari 支持的 CSS3 的很多新特性。灵活运用 CSS，不仅可以简化程序，而且可以轻松实现很多特殊的效果。这一点，会在后面详细说明。

### 2、JavaScript 部分

```
<script Language="JavaScript">.....</script>
```

这部分是程序实现各种功能的关键。一般包括：

#### ■ 变量定义



## ■ 自定义函数

简单示例：

```
<script Language="JavaScript">
    var MsgArr=new Array()
    var pPot=pMin=0; pMax=9
    var wAddr="http://bbs.weiphone.com/?u=61542"

    function pSetup(s)
    {
        return (s<pMax) ? true : false
    }
</script>
```

当外部 js 文件与内嵌 javascript 段存在变量或函数冲突时，按加载顺序，以最后的定义为准。

Javascript 段亦可放在 <body> 段之后，可实现先加载 <body>，后加载 <javascript>。但必须注意，由于是顺序加载，<body> 段不能引用后面的变量或函数，但可以向后调用。

## 3、<body>部分

这是程序界面的主要部分。程序的显示、交互一般都放到这个段。

简单示例：

```
<body style="margin: 0; font-size: 12pt; font-family: Trebuchet MS; color: #FFFFFF"
    ontouchmove="event.preventDefault()" onload="tImg.style.visibility='visible'" >
    
    <table id="layerT" style=" position: absolute; width: 320px; height: 480px">
        <tr><th id="TitleBar" colspan="2" height="48px"> </th></tr>
        <script Language="JavaScript">
            for (i=0; i<=pMax; i++)
            {
                document.write("<tr ontouchend ='DispImg(" + i + ")' style='border-top:
1px solid #CCC'><td id='M' + i + "></td><td width='12' >></td></tr>");
            }
        </script>
        <tr><td id="ToolBar" colspan="2" height="42px"> </td></tr>
    </table>
    <div id="layerH" style="position: absolute; width: 320px; height: 480px; left: 0px; top:
0px; background-color: RGBA(0,0,0,0.3); z-index: 1; visibility:hidden">
        <div id="layerM" style="-webkit-border-radius:12px; border:2px solid #FFF;
```

```

-webkit-box-shadow: 0px 2px 4px #888;
position: absolute; left: 24px; top: 106px;
width: 256px; height: 268px; padding-left: 8px; padding-right: 8px;
color: #FFFFFF; text-shadow: 1px 1px 1px #000; text-align: center;
background-color: RGBA(32,48,96,0.9);
background-image:url('BG-Msg.png'); background-repeat:no-repeat;
z-index: 1; visibility: hidden; ">
<p><span style="font-size: 16pt; font-weight: bold">使用说明</span></p>
<hr noshade size="1">
<div id="HelpText" style="height: 120px">说明文字</div>
<hr noshade size="1">
<form name="formV" method="POST">
    <input type="button" class="sButton" value="确认" name="B1"
        onclick=" layerH.style.visibility='hidden'" />
</form>
</div>
</div>
</body>

```

简要说明：

`ontouchmove="event.preventDefault()"` //锁定 viewport，任何屏幕操作不移动用户界面（弹出键盘除外）。

`onload="tImg.style.visibility='visible'"` //<body>加载完成后，显示该图片。

`onerror="pError()"` //图片加载错误时，执行错误处理过程。（**特别说明**：在 IE 中，只有图片文件不存在，才返回错误；而在 iPhone Safari 中，图片文件不存在，或格式不正确，均返回错误。因此这种方式，只能用来判断图片文件是否存在。）

`position: absolute; width: 320px; height: 480px` //设定该容器绝对定位位置，正好为完全的 viewport 尺寸。（**特别说明**：除非参数值为 0，建议数值参数均带上单位。）

`id="TitleBar"` //定义单元标志，其后可以使用：`TitleBar.innerHTML = "标题文字"` 或 `TitleBar.innerHTML = "<b>标题文字</b>"` 来重置该单元内容。不同的是 innerText 是纯文本内容，而 innerHTML 可以使用规范的 html 格式。

`width: 256px; height: 268px; padding-left: 8px; padding-right: 8px;` //由于设定了容器的 padding 值，所以，该容器的宽度实际是： $256 + 8 + 8 = 272$  px。

`z-index: 1` //设定该层的顺序。层号越大越靠上；若有相同层号，按程序中定义顺序，后边的靠上；设定为 -1 会不显示，但为显示方便，避免层设定混乱，隐藏某层一般使用：`visibility: hidden`。



## 四、属性

### 1、字体

① **font-family**: 字体名称 (iPhone 自带字体参见右表)

② **font-size**: 字号参数 (一般标题使用 18 pt, 正文使用 12pt, 按钮使用 14pt, 工具条使用 12px)

③ **font-style**: 斜体字的名称 (italic)

④ **font-weight**: 字体粗细 (取值范围从数字 100~900, 浏览器默认的字体的粗细为 400, 默认加粗 bold 为 700。另外可以通过参数 lighter 和 bolder 使得字体在原有基础上显得更细或更粗些。)

⑤ **text-transform**: 文字大小写

uppercase: 所有文字大写显示

lowercase: 所有文字小写显示

capitalize: 每个单词的头字母大写显示

none: 不继承母体的文字变形参数。

⑥ **text-decoration**: 文字修饰

underline: 为文字加下划线

overline: 为文字加上划线

line-through: 为文字加删除线

blink: 使文字闪烁

none: 不显示上述任何效果。

⑦ **text-align**: 文本水平对齐

left: 左对齐

right: 右对齐

center: 居中对齐

justify: 相对左右对齐

但需要注意的是, text-align 是块级属性, 只能用于 <p>、<blockquote>、<ul>、<h1>~<h6> 等标识符里。

⑧ **vertical-align**: 文本垂直对齐

top: 顶对齐

bottom: 底对齐

text-top: 相对文本顶对齐

text-bottom: 相对文本底对齐

baseline: 基准线对齐

middle: 中心对齐

sub: 以下标的形式显示

super: 以上标的形式显示)

⑨ **text-shadow**: 1px 1px 1px #000: 文字阴影。(参数分别为: 水平 X 方向偏移量; 垂直 Y

### Fonts on the iPhone

- American Typewriter
- Arial
- Arial Rounded MT Bold
- Courier
- Courier New
- Georgia
- Helvetica
- Helvetica Neue
- Marker Felt
- Times New Roman
- Trebuchet MS
- Verdana

• Zapfino

方向偏移量；高斯模糊半径值；阴影颜色值)

## 2、颜色

















① **color** : 颜色属性 ( 颜色参数取值范围：以 RGB 值表示；以 16 进制 ( hex ) 的色彩值表示；以默认颜色的英文名称表示，注意大小写！ )

例如：

**color: #0080FF** (可以使用#abc 缩写来表示#aabbcc)

**color: RGBA(00,33,66,0.5)** (Red:0 Green:33 Blue:66 opacity:50%)

**color: red** (颜色代码保留字，参见下表)

 Black="#000000"	 Green="#008000"	 Gray="#808080"	 White="#FFFFFF"
 Silver="#C0C0C0"	 Lime="#00FF00"	 Yellow="#FFFF00"	 Olive="#808000"
 Fuchsia="#FF00FF"	 Maroon="#800000"	 Red="#FF0000"	 Purple="#800080"
 Aqua="#00FFFF"	 Teal="#008080"	 Blue="#0000FF"	 Navy="#000080"

② **background-color** : 背景颜色

参数取值和颜色属性一样。

## 3、背景

① **background-image: url(URL)**

URL 就是背景图片的存放路径。如果用 "none" 来代替背景图片的存放路径，将什么也不显示

② **background-repeat** : 背景图片重复 ( 默认的是水平、垂直两个方向上平铺 )

no-repeat : 不重复平铺背景图片

repeat-x : 只在水平方向上平铺

repeat-y : 只在垂直方向上平铺

③ **background-position** : 背景定位 ( 用于控制背景图片在网页中显示的位置。其参数为带长度单位的数字参数，其取值范围：

top : 相对前景对象顶对齐

bottom : 相对前景对象底对齐

left : 相对前景对象左对齐

right : 相对前景对象右对齐

center : 相对前景对象中心对齐

比例关系

参数中的 center 如果用于另外一个参数的前面，表示水平居中；如果用于另外一个参数的后面，表示垂直居中。)

## 4、容器

① **margin** 边框空白。位于 BOX 模型的最外层，包括四项属性。格式分别如下：

margin-top：顶部空白距离

margin-right：右边空白距离

margin-bottom：底部空白距离

margin-left：左边空白距离

如果使用上述属性的简化方式 margin，可以在其后连续加上四个带长度单位的数字，来分别表示 margin-top、margin-right、margin-bottom、margin-left，每个数字中间要用空格分隔。

② **border** 对象边框。位于边框空白和对象空隙之间，包括了七项属性。格式分别如下：

border-top：顶边框宽度

border-right：右边框宽度

border-bottom：底边框宽度

border-left：左边框宽度

border-width：所有边框宽度

border-color：边框颜色

border-style：边框样式参数

其中 border-width 可以一次性设置所有的边框宽度，border-color 同时设置四面边框的颜色时，可以连续写上四种颜色，并用空格分隔。上述连续设置的边框都是按 border-top、border-right、border-bottom、border-left 的顺序。

Border-style 相对别的属性而言稍稍复杂些，因为它还包括了多个边框样式的参数：

none：无边框。

dotted：边框为点线。

dashed：边框为长短线。

solid：边框为实线。

double：边框为双线。

groove：根据 color 属性显示不同效果的 3D 边框

ridge：根据 color 属性显示不同效果的 3D 边框

inset：根据 color 属性显示不同效果的 3D 边框

outset：根据 color 属性显示不同效果的 3D 边框

③ **padding** 对象间隙。位于对象边框和对象之间，包括了四项属性。基本格式如下：

padding-top：顶部间隙

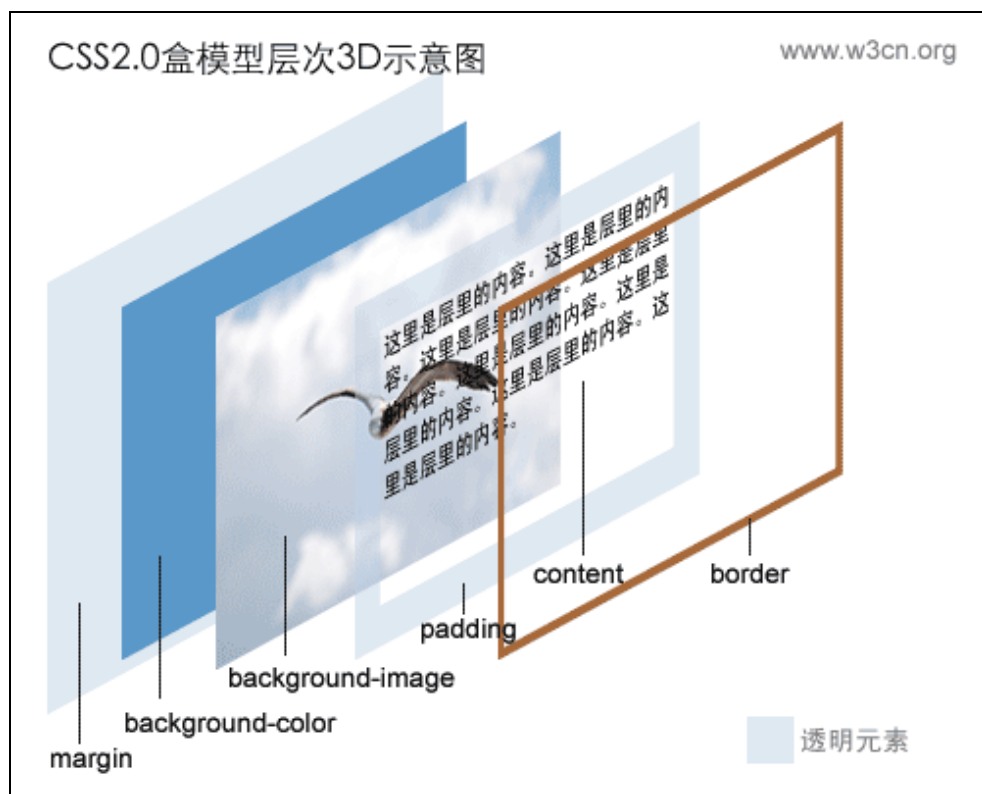
padding-right：右边间隙

padding-bottom：底部间隙

padding-left：左边间隙

和 margin 类似，padding 也可以一次性设置所有的对象间隙。

④ **background: url(..), url(..), url(..), url(..)** 多重背景。



## 五、事件

### 1、

在第二章中提到过，iPhone Safari 浏览器行为的改变不止是滚动条，交互事件也跟普通桌面不一样：所有 hover 动作，还有 mouseover，都不存在。

点击页面元素之后发生的事情很复杂：比如当你用单指按住可点击元素，然后放开，首先触发的 event 居然是 mousemove！接下来如果元素内容不变，会陆续触发 mousedown ,mouseup ,click，如果内容改变，就不会再触发任何事件。

如果按住元素之后移动手指，当然就不要指望会触发 mousemove 了，啥事情都没有，不过停下的时候，会触发 onscroll（因为你刚才移动了 viewport 嘛.....）

如果你用两个手指做缩放的动作，啥事件都不会有啦，但是如果你用两个手指在屏幕上一同移动，而且所在的位置是一个本来有滚动条的页面元素（比如 iframe 罢），会触发一个叫 mousewheel 的事件（但是别指望 iframe 本身的内容会滚动），停止移动时同样触发 onscroll。

### 2、

如果你熟悉 IE 下的网页制作，那必须告诉你 iPhone Safari 的交互事件的一些特殊之处。

相对于 IE，iPhone Safari 增加了一些屏幕操作的事件：

- **Touchstart** //当手指接触屏幕时触发
- **Touchmove** //当已经接触屏幕的手指开始移动后触发
- **Touchend** //当手指离开屏幕时触发
- **touchcancel**
- **gesturestart** //当两个手指接触屏幕时触发
- **gesturechange** //当两个手指接触屏幕后开始移动时触发
- **gestureend**

运用这几个交互事件，完成某一元素的移动，或者检测手指在屏幕上的划动，已经十分简单。

例如：

```
onTouchstart="startX = event.touches[0].pageX; endX = event.touches[0].pageX"
onTouchmove="endX = event.touches[0].pageX; pMove()"
onTouchend="pClick()"
```

### 3、

对于某一可点击表格元素使用 click 和 onTouchend 均可完成点击触发事件的功能，二者的区别在于：ontouchend 不可用于 IE，并且不会出现点击后元素的外观变化（例如被点击单元格会高亮显示），这也许正是你所需要的。当然，被点击元素的外观变化，可以使用样式来设定：

**-webkit-tap-highlight-color: 颜色**

另外，在 iPhone Safari 上，onclick 会有残留。例如点击后显示某一个层，如果该层在此位置也定义了可触发，则可能会也触发这个事件。



## 4、

屏幕旋转事件：onorientationchange，这当然是 iPhone 所特有的了。添加屏幕旋转事件侦听，可随时发现屏幕旋转状态（左旋、右旋还是没旋）。例子：

```
function orientationChange() {  
    switch(window.orientation)  
    {  
        case 0: alert( "没旋" ); break;  
        case -90: alert( "左旋" ); break;  
        case 90: alert( "右旋" ); break;  
    }  
}  
addEventListener('load', function() {  
    orientationChange();  
    window.onorientationchange = orientationChange; } );
```

## 5、

隐藏地址栏 & 处理事件的时候，防止滚动条出现：

```
addEventListener('load', function() {  
    setTimeout(function() { window.scrollTo(0, 1); }, 100); } );
```

## 6、

双手指滑动事件：

```
addEventListener('load', function() {  
    window.onmousewheel = twoFingerScroll; }, false);  
function twoFingerScroll(ev) {  
    var delta=ev.wheelDelta/120;    //对 delta 值进行判断(比如正负)，而后执行相应操作  
    return true;  
}
```

## 7、

判断是否为 iPhone：

```
function isAppleMobile() {  
    return ((navigator.platform.indexOf("iPhone") != -1) }
```



## 六、特性

本章介绍 iPhone Safari 上所特有的一些性能。

### 1、localStorage

众所周知，为了保护 Web 用户的安全性，HTML 程序禁止向客户端写入任何文件。这一程度上限定了一些 WebApp 的功能范围。IE 可以变通使用 Microsoft 的 FileSystemObject 来写文件，这在 iPhone 上则无法实现。如果用户想保存一些数据，比如某些设定或用户登录数据，待程序下次执行时可以直接读取，怎么办呢？

“使用 Cookie ！”相信很多朋友会这样回答。是的，可以使用 Cookie，但是，使用 Cookie 有很多缺陷，除了读写比较复杂外，还有：用户没有打开 Cookie 怎么办？期间用户删除了 Cookie 怎么办？其实，iPhone Safari 有一项符合 HTML5 规范的扩展功能：**localStorage**。它使得开发者的这种需求变得十分简单。

例子：（注意数据名称 n 要用引号引起来）

`var v = localStorage.getItem('n') ? localStorage.getItem('n') : "";` //如果名称是 n 的数据存在，则将其读出，赋予变量 v。

`localStorage.setItem('n', v);` //写入名称为 n、值为 v 的数据

`localStorage.removeItem('n');` //删除名称为 n 的数据

### 2、电话

如果你关闭自动识别后，又希望某些电话号码能够链接到 iPhone 的拨号功能，那么可以通过这样来声明电话链接：

`<a href="tel:13800138000">13800138000</a>` 或用于单元格：

`<td onclick="location.href='tel:122'">`

### 3、自动大写与自动修正

在文本框中输入时，iPhone 提供了自动大写与自动更正两项功能。自动大写的意思是，在输入开始的时候，以及在一个句号并空一个格后，自动会启用 shift，输入一个字母后该 shift 自动消失。自动修正的意思是，iPhone 会自动根据词库，包括自带的以及从你过往输入分析而来的，来对你的输入进行自动更正。我们都知道用手指点击那么小一个软键盘很容易误按旁边的键，这时候你可以不用忙于修正，只要 iPhone 提示的自动修正的词正是你想要的，你就可以按空格然后输入下一个词，iPhone 会自动修正前面那个词。

要关闭这两项功能，可以通过 **autocapitalize** 与 **autocorrect** 这两个选项：

`<input type="text" autocapitalize="off" autocorrect="off" />`

### 4、WebKit

基于 WebKit 的 iPhone Safari 还有一些特有的样式，为有别与其它浏览器，它使用：

`-webkit-` //详见第七章 CSS。



## 七、CSS

CSS ( Cascading Stylesheets, 层叠样式表 ) 是一种制作网页的新技术, 现在已经为大多数的浏览器所支持, 成为网页设计必不可少的工具之一。

W3C ( The World Wide Web Consortium ) 把动态 HTML ( Dynamic HTML ) 分为三个部分来实现 : 脚本语言( 包括 JavaScript、Vbscript 等 )、支持动态效果的浏览器( 包括 Internet Explorer、Netscape Navigator、Safari 等 ) 和 CSS 样式表。

1、定义方式, 包括外部调用和内部嵌入两种 :

```
<script type="text/javascript" src="myClass.css"></script>
<style type="text/css">.....</style>
```

2、一般格式为 :

```
.sImg { position: absolute; left: 0; top: 0; width: 320px; height: 480px; border: 0; }
th { -webkit-border-radius: 4px }
```

对于第一种, 使用 : `` 或在 javascript 段, 使用 : `tImg.className = "sImg"` 来引用。

对于第二种, 无需引用, 其对所有表格中的标题单元格 ( th ) 均起作用。

3、作用级别 :

如果相应的样式在多处均有描述, 则其作用级别 ( 从低到高 ) 为 :

外部 CSS 文件、内嵌 CSS 段 ( `.className` 引用、`.style.cssText` 重设 ) 元素定义 ( `class="....."`、`style="....."` ) 。

4、特有样式 **webkit**。实施大子的 CSS 2.1 规格所界定的万维网联盟 ( W3C ), 以及部分的 CSS 3 规格。(为便于查询, 也包括了一些同类别的非 iPhone 样式)。

① “盒模型” 的具体描述性质的包围盒块内容, 包括边界, 填充等等。

**-webkit-border-bottom-left-radius:** radius;

**-webkit-border-top-left-radius:** horizontal\_radius vertical\_radius;

**-webkit-border-radius:** radius; //容器圆角

**-webkit-box-sizing:** sizing\_model; 边框常量值 : border-box/content-box

**-webkit-box-shadow:** hoff voff blur color; //容器阴影 ( 参数分别为 : 水平 X 方向偏移量 ; 垂直 Y 方向偏移量 ; 高斯模糊半径值 ; 阴影颜色值 )

**-webkit-margin-bottom-collapse:** collapse\_behavior; 常量值 : collapse/discard/separate

**-webkit-margin-start:** width;

**-webkit-padding-start:** width;

**-webkit-border-image:** url(borderimg.gif) 25 25 25 25 round/stretch round/stretch;





**-webkit-appearance:** push-button; //内置的 CSS 表现，暂时只支持 push-button

② “视觉格式化模型” 描述性质，确定了位置和大小块元素。

**direction:** rtl

**unicode-bidi:** bidi-override; 常量：bidi-override/embed/normal

③ “视觉效果” 描述属性，调整的视觉效果块内容，包括溢出行为，调整行为，能见度，动画，变换，和过渡。

**clip:** rect(10px, 5px, 10px, 5px)

**resize:** auto; 常量：auto/both/horizontal/none/vertical

**visibility:** visible; 常量：collapse/hidden/visible

**-webkit-transition:** opacity 1s linear; 动画效果 ease/linear/ease-in/ease-out/ease-in-out

**-webkit-backface-visibility:** visible; 常量：visible(默认值)/hidden

**-webkit-box-reflect:** right 1px; 镜向反转

**-webkit-box-reflect:** below 4px -webkit-gradient(linear, left top, left bottom, from(transparent), color-stop(0.5, transparent), to(white));

**-webkit-mask-image:** -webkit-gradient(linear, left top, left bottom, from(rgba(0,0,0,1)), to(rgba(0,0,0,0)));

//CSS 遮罩/蒙板效果

**-webkit-mask-attachment:** fixed; 常量：fixed/scroll

**-webkit-perspective:** value; 常量：none(默认)

**-webkit-perspective-origin:** left top;

**-webkit-transform:** rotate(5deg);

**-webkit-transform-style:** preserve-3d; 常量：flat/preserve-3d; (2D 与 3D)

④ “生成的内容，自动编号，并列出” 描述属性，允许您更改内容的一个组成部分，创建自动编号的章节和标题，和操纵的风格清单的内容。

**content:** “Item” counter(section) “ ”;

This resets the counter.

First section

>two section

three section

counter-increment: section 1;

counter-reset: section;

⑤ “分页媒体” 描述性能与外观的属性，控制印刷版本的网页，如分页符的行为。

**page-break-after:** auto; 常量：always/auto/avoid/left/right

**page-break-before:** auto; 常量：always/auto/avoid/left/right

**page-break-inside:** auto; 常量：auto/avoid

⑥ “颜色和背景” 描述属性控制背景下的块级元素和颜色的文本内容的组成部分。

**-webkit-background-clip:** content; 常量：border/content/padding/text

**-webkit-background-origin:** padding; 常量：border/content/padding/text

**-webkit-background-size:** 55px; 常量：length/length\_x/length\_y



⑦ “字型” 的具体描述性质的文字字体的选择范围内的一个因素。报告还描述属性用于下载字体定义。

**unicode-range:** U+00-FF, U+980-9FF;

⑧ “文本” 描述属性的特定文字样式，间距和自动滚屏。

**text-shadow:** #00FFFC 10px 10px 5px;

**text-transform:** capitalize; 常量：capitalize/lowercase/none/uppercase

**word-wrap:** break-word; 常量：break-word/normal

**-webkit-marquee:** right large infinite normal 10s; 常量：direction(方向) increment(迭代次数) repetition(重复) style(样式) speed(速度);

**-webkit-marquee-direction:** ahead/auto/backwards/down/forwards/left/reverse/right/up

**-webkit-marquee-increment:** 1-n/infinite(无穷次)

**-webkit-marquee-speed:** fast/normal/slow

**-webkit-marquee-style:** alternate/none/scroll/slide

**-webkit-text-fill-color:** #ff6600; 常量：capitalize, lowercase, none, uppercase

**-webkit-text-security:** circle; 常量：circle/disc/none/square

**-webkit-text-size-adjust:** none; 常量:auto/none;

**-webkit-text-stroke:** 15px #fff;

**-webkit-line-break:** after-white-space; 常量：normal/after-white-space

**-webkit-appearance:** caps-lock-indicator;

**-webkit-nspace-mode:** space; 常量：normal/space

**-webkit-rtl-ordering:** logical; 常量：visual/logical

**-webkit-user-drag:** element; 常量：element/auto/none

**-webkit-user-modify:** read-only; 常量：read-write-plaintext-only/read-write/read-only

**-webkit-user-select:** text; 常量：text/auto/none

⑨ “表格” 描述的布局和设计性能表的具体内容。

**-webkit-border-horizontal-spacing:** 2px;

**-webkit-border-vertical-spacing:** 2px;

**-webkit-column-break-after:** right; 常量：always/auto/avoid/left/right

**-webkit-column-break-before:** right; 常量：always/auto/avoid/left/right

**-webkit-column-break-inside:** logical; 常量：avoid/auto

**-webkit-column-count:** 3; //分栏

**-webkit-column-rule:** 1px solid #fff;

style:dashed,dotted,double,groove,hidden,inset,none,outset,ridge,solid

⑩ “用户界面” 描述属性，涉及到用户界面元素在浏览器中，如滚动文字区，滚动条，等等。报告还描述属性，范围以外的网页内容，如光标的标注样式和显示当您按住触摸触摸目标，如在 iPhone 上的链接。

**-webkit-box-align:** baseline,center,end,start,stretch 常量：baseline/center/end/start/stretch

**-webkit-box-direction:** normal;常量：normal/reverse



**-webkit-box-flex:** flex\_valuet  
**-webkit-box-flex-group:** group\_number  
**-webkit-box-lines:** multiple; 常量 : multiple/single  
**-webkit-box-ordinal-group:** group\_number  
**-webkit-box-orient:** block-axis; 常量 : block-axis/horizontal/inline-axis/vertical/orientation  
**-webkit-box-pack:** alignment; 常量 : center/end/justify/start

## 5、动画过渡

这是 Webkit 中最具创新力的特性：使用过渡函数定义动画。

**-webkit-animation:** title infinite ease-in-out 3s;

animation 有这么几个属性：

**-webkit-animation-name :** //属性名，就是我们定义的 keyframes  
**-webkit-animation-duration : 3s** //持续时间  
**-webkit-animation-timing-function :** //过渡类型：ease/ linear(线性) /ease-in(慢到快) /ease-out(快到慢) /ease-in-out(慢到快再到慢) /cubic-bezier  
**-webkit-animation-delay : 10ms** //动画延迟(默认 0)  
**-webkit-animation-iteration-count :** //循环次数(默认 1)，infinite 为无限  
**-webkit-animation-direction :** //动画方式 :normal(默认 正向播放)；alternate(交替方向，第偶数次正向播放，第奇数次反向播放)

这些同样是可以简写的。但真正让我觉的很爽的是 keyframes，它能定义一个动画的转变过程供调用，过程为 0%到 100%或 from(0%)到 to(100%)。简单点说，只要你有想法，你想让元素在这个过程中以什么样的方式改变都是很简单的。

**-webkit-transform:** 类型 ( 缩放 scale/旋转 rotate/倾斜 skew/位移 translate )  
*scale(num,num)* 放大倍率。scaleX 和 scaleY(3)，可以简写为：scale(\*, \*)  
*rotate(\*deg)* 转动角度。rotateX 和 rotateY，可以简写为：rotate(\*, \*)  
*Skew(\*deg)* 倾斜角度。skewX 和 skewY，可简写为：skew(\*, \*)  
*translate(\*,\*)* 坐标移动。translateX 和 translateY，可简写为：translate(\*, \*)。

### 实现模拟弹出消息框 ( Alert ) 的例子：

①定义过渡 ( 在 <style type="text/css">段中描述 keyframes )：

```

@-webkit-keyframes DivZoom
{
    0% { -webkit-transform: scale(0.01) }
    60% { -webkit-transform: scale(1.05) }
    80% { -webkit-transform: scale(0.95) }
    100% { -webkit-transform: scale(1.00) }
}
.sZoom { -webkit-animation: DivZoom 0.5s ease-in-out }
  
```



( 很容易看懂, 将元素从缩小的 0.01 倍--很小但不能为 0 倍, 放大到 1.05 倍, 再缩小到 0.95 倍, 最后到 1 倍即正常大小。整个过渡过程事件为 0.5 秒, 动画方式为 **ease-in-out**, 即慢到快再到慢, 默认只进行 1 次过渡。这正是大家经常看到的 iPhone 弹出的提示信息的动画效果! )

②定义元素 ( 在 `<body>` 段中 ) :

```
<div id="layerH" style="-webkit-border-radius:12px; border:2px solid #FFF;
    -webkit-box-shadow: 0px 2px 4px #888;
    position: absolute; left: 24px; top: 106px;
    width: 256px; height: 268px; padding-left: 8px; padding-right: 8px;
    color: #FFFFFF; text-shadow: 1px 1px 1px #000; text-align: center;
    background-color: RGBA(32,48,96,0.9);
    background-image:url('BG-Msg.png'); background-repeat:no-repeat;
    z-index: 1; visibility: hidden; ">
<p><span style="font-size: 16pt; font-weight: bold">使用说明</span></p>
<hr noshade size="1">
<div id="HelpText" style="height: 120px">说明文字</div>
<hr noshade size="1">
<form name="formV" method="POST">
    <input type="button" value="确认" name="B1"
        style="width: 100%; height: 40px; font-size: 14pt; font-weight: bold;
        color: #FFFFFF; text-shadow: 0px -1px 1px #000;"
        onclick=" layerH.style.visibility='hidden'">
</form>
</div>
```

③启动动画 ( 在 `javascript` 定义的函数中 )

```
function pHelp()
{
    layerH.style.visibility = 'visible'
    layerH.style.cssText = "-webkit-animation-delay: " + Math.random() + "ms"
    layerH.className = 'sZoom'
}
```

(这个启动函数就很好理解了。但是为什么要使用 `-webkit-animation-delay` 这句呢? 因为当一个元素过渡显示完成后, 若其样式没有变化, 下一次将无法进行过渡动画显示。我们巧妙的利用其动画延迟时间定义, 使其有所变化, 就避免了上述问题。其中使用随机数函数 `Math.random()`, 产生一个大于 0 小于 1 的随机数。当然, 延迟零点几毫秒, 用户是不会察觉的。)



## 八、封装

至此为止，我们把开发一个 iPhone WebApp 的全部关键问题都了解了，看来写一个实际的应用软件已经没有问题了。现在的问题是：程序写好了，怎么安装到 iPhone 上来运行？

1、立刻有人回答：我们的 WebApp 不是设计在 iPhone 上运行吗？使用 iPhone 本身自带的 Safari 来运行不就可以了？

是的！按以下步骤来操作吧：

①将开发完成的 WebApp 全部文件，包括 HTML 文件、所用到的图片、或许需要的外部 js 文件和 css 文件，统统复制进 iPhone。为了便于管理，最好是复制到 iPhone 的：

`private/var/mobile/Document\` 之下，新建的某个目录，比如：`My WebApp` 中。

②为使 iPhone Safari 能够使用本地文件，需要安装一个插件：**file://Schema in Safari**，这个插件可在 Cydia 上下载安装。

③启动 iPhone Safari，在地址栏中输入：

`File:\\private\\var\\mobile\\Document\\My WebApp\\index.html`

终于启动运行了。

④此时，点击 Safari 窗口底部工具栏是的 “+” 图标，生成桌面图标。以后就可以直接在桌面上快捷地来运行这个 WebApp 程序了。

好复杂啊！！界面上 Safari 固有的工具栏总是存在，影响了我的程序界面！这还不算，用户操作 iPhone 水平不一，安装一个这样的程序太不友好了！！总之，很多人会说，这个程序太不专业了，不像个软件，使用 Webkit 方式开发的软件就是不如 SDK 方式.....

2、但是，**现在不同了！**威锋网技术组的 hhytt 成功编写了一个外壳程序，可以很容易的将 WebApp 封装打包为 iPhone 上使用的标准 ipa 软件格式，用户可以像众多的其它软件一样，更加规范、便捷的进行安装使用。这也极大的促进了 WebApp 软件的流传。

关于使用这个外壳程序的一些说明：

①在电脑上，将 **WebViewer by hhytt.rar** 解压后，生成一个 `WebViewer by hhytt` 目录，其中有以下文件夹及文件（加粗字体表示文件夹，普通字体表示文件）：

**WebViewer by hhytt**

└ iTunesArtwork

└ **Payload**

└ **WebViewer.app**

└ Default.png

└ icon.png

└ Info.plist

└ WebViewer

└ PkgInfo

└ **html**



将你的 WebApp 程序全部文件复制到 **html** 目录中，主文件取名为：index.html。

②修改有关文件名（例如你的 WebApp 程序取名为 MyWabApp）：

将：**WebViewer** 文件更名为 **MyWabApp**；

将：**WebViewer.app** 目录更名为 **MyWabApp.app**。

**icon.png** 是你的程序的图标，57×57 像素，png 格式；

**Default.png** 是你的程序启动时的封面图片，尺寸为 320×480 像素，png 格式。

**iTunesArtwork** 是你的程序的插图图片，也就是在 iTunes 中应用软件资料库中显示的图片，最大为 512×512 像素，也可借用 icon.png。可以为 jpeg 或 png 格式。注意，这个文件做好后，要去掉扩展名。

这 3 个文件需要你自己制作，复制进来覆盖原来的文件即可。

③修改 **Info.plist** 文件。这很关键，以下详细介绍：

Info.plist 在 iPhone 上任何一款软件中都存在，是重要的安装配置文件。其格式比较特殊，可使用专用程序 **Pledit.exe** 来打开并编辑它。

其基本内容为（**红色**为需要修改的部分，//为笔者添加的注释，实际文件中不应存在）：

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
<key>CFBundleDevelopmentRegion</key>
<string>zh</string>                //该软件默认运行语言环境
<key>CFBundleDisplayName</key>
<string>MyWabApp</string>         //桌面名称（可使用中文）
<key>CFBundleExecutable</key>
<string>MyWabApp</string>         //主程序文件名称
<key>CFBundleIconFile</key>
<string>icon.png</string>         //图标文件名称
<key>CFBundleIdentifier</key>
<string>com.lh.MyWabApp</string>  //作者身份（必须唯一）
<key>CFBundleInfoDictionaryVersion</key>
<string>6.0</string>
<key>CFBundleName</key>
<string>MyWabApp</string>         //主程序目录名称
<key>CFBundlePackageType</key>
<string>APPL</string>
<key>CFBundleSignature</key>
<string>????</string>
<key>CFBundleVersion</key>
```



```
<string>0.1</string>           //该软件版本号
<key>DTPlatformName</key>
<string>iphoneos</string>
<key>DTSDKName</key>
<string>iphoneos2.0</string>
<key>LSRequiresiPhoneOS</key>
<string>YES</string>
<key>MinimumOSVersion</key>
<string>2.0</string>           //运行时所需最低 iPhone OS 版本
<key>SignerIdentity</key>
<string>Apple iPhone OS Application Signing</string>
<key>UIInterfaceOrientation</key>
<string>UIInterfaceOrientationLandscapeRight</string>
                                   //是否横屏，可用...LandscapeRight/Left/No/Auto
<key>UIStatusBarHidden</key>
<true/>                         //是否隐藏状态栏，可用 true/false
<key>UIPrerenderedIcon</key>
<true/>                         //是否使用软件自带原始图标，true-原始/false-使用系统遮罩图标
</dict>
</plist>
```

#### ④打包：

在 **WebViewer by hhytt** 目录下，同时选中其下的 iTunesArtwork 文件和 **Payload** 文件夹，生成 ZIP 格式压缩文件 WebViewer by hhytt.zip。然后就其更名为 MyWabApp.ipa。一个标准的 iPhone WebApp 软件诞生了。

然后，你就可以将其安装到 iPhone 上运行了。



## 九、后记

成功是令人愉悦的。解决通往成功路上的一个个难题是完成挑战的过程,更是充满了苦恼与欢乐。虽然,通过 WebApp 可以完成很多应用程序的开发,但,我还想知道:

### 1、数据库操作

虽然 WebKit 提供了构建于 SQLite 之上的客户端数据库,并提供在 javascript 中以下方法:

①加载数据库:

```
var myDB = openDatabase (DBFileName, version, DisplayName, MaxSize)
```

②运行事务:

```
myDB.transaction (
    function (transsction) {
        transaction.executeSql ( SQL, [array of ?s], dataHandler, errorHandler);
    }
);
```

可是,我暂时还没有试验成功 .....

### 2、播放声音

在 iPhone Safari 上,播放声音必须通过其 QuickPlayer。这使得用户需要实现的某些功能.....例如最简单的点击某元素时的声音反馈,难以实现。

### 3、对设备有关数据的读取

最简单的,如何知道当前 iPhone 的 IMEI ? 如何知道当前设置的语言环境( English 还是 简体中文 )?

加速计、光线感应器、接近性传感器,现在还不能获取其信息。

由于 WebApp 自身的限制,无法进行较底层的开发编程。在这个方面,SDK 开发的优势就彰显出来了。

( 未完待续..... )

