HTML5发展历史

# 一、概述

2007年W3C(万维网联盟)立项HTML5，直至2014年10月底，这个长达八年的规范终于正式封稿。

过去这些年，HTML5颠覆了PC互联网的格局，优化了**移动互联网**的体验，接下来，HTML5将**颠覆原生App**世界。这听起来有点危言耸听，但若认真分析HTML5的发展史，你会发现，这个世界的发展趋势确实就是这样。

在互联网的早期，对用户而言，能打开浏览器接入到互联网世界就是一个神奇的事情，但互联网发展到2005年前后，开始出现下一个变化，就是宽带互联。

随着宽带的普及和电脑性能的增强，人们不再满足于单纯的通过互联网看新闻、收发邮件，消耗更高带宽的娱乐产品开始出现，就是流视频和网页游戏。其实视频和游戏是古老的需求，在互联网不普及的时候，需求的满足方式是离线传输的VCD和游戏光盘;后来互联网逐渐普及，人们更改了使用方式，通过下载软件+本地媒体播放器来看视频，下载体积较大的端游玩游戏。

但是对消费者体验更好的新方式还是出现并颠覆了以前的一切，那就是**流媒体和网页游戏**。Youtube等公司把握住潮流飞速崛起，各种页游公司也如雨后春笋。

但是HTML标准没有把握住产业的变化及时演进，浏览器产品也未升级，这块新需求被浏览器插件满足了，那就是**Flash**。这个部署在亿万浏览器里的商业插件俨然成为事实标准。2005年Adobe巨资收购Macromedia，把Flash收归旗下，紧接着大幅推广FLV流媒体和action script语言，很明显这桩收购可以列为IT并购的经典案例，FLV流媒体和Flash游戏风靡互联网，Adobe在新的产业升级中攫取了大量的利润。

除了Flash这个商业产品成为了事实标准，W3C还面临一个尴尬，就是另一个私有扩展协议的制造者—IE。IE当时在桌面浏览器占有垄断地位，并且扩展了大量的IE Only语法，开发者完全不知道这些语言是谁定义的。整个web世界，就被两家公司微软+Adobe绑架了。

很多IT巨头都坐不住了，尤其是苹果和Google。PC操作系统的世界难有突破，Web浏览器被苹果寄予厚望，而且第一代iPhone只支持网页，那时还没有Appstore，Safari是乔布斯非常看重的产品；新贵Google虽然大量赞助Mozilla，但并未对IE的地位产生实质影响，收购了YouTube后发现底层被Adobe控制，也是非常难过，而且Google每年给IE的搜索框和AdobleFLV缴纳的费用真不是小数目。

既然大家都是W3C的主席单位，好吧，我们重新开始做HTML5吧。

是的，HTML5其实就是这么诞生的。那是2007年，IE和Flash由盛转衰的转折点。

# 二、从IETF到W3C：HTML 4之路

HTML 1并不曾存在，HTML的第一个官方版本就是由IETF（互联网工程任务组）推出的**HTML 2.0**。问世之前，这个版本中的很多细则已经被实现，比如，1994年的Mosaic浏览器已经实现了在文档中嵌入图片的方法，后来HTML 2.0便吸纳了img这个标签。

后来，W3C取代IETF的角色，成为HTML的标准组织，二十世纪90年代的后半叶，HTML的版本被频繁修改，直到1999年的HTML 4.01， 至此，HTML到达了它的第一个拐点。

# 三、XHTML 1：XML风格的HTML

HTML在HTML 4.01之后的第一个修订版本就是**XHTML 1.0**，其中X代表“eXtensible”（扩展），当然也有人将之解读为 “eXtreme”（极端）。**XHTML 1.0是基于HTML 4.01的**，并没有引入任何新标签或属性，唯一的区别是语法，HTML对语法比较随便，而 XHTML则要求XML般的严格语法。

使用严格的语法规范并非坏事，要求开发者使用单一的代码风格，比如，HTML4.01允许你使用大写或小写字母标识标记元素和属性，XHTML则只允许小写字母。**XHTML1.0的推出刚好碰上了CSS的崛起**，Web开发设计者们开始意识到Web标准问题，基于XHTML的严格语法规范被视为编写 HTML代码的最佳实践。

# 四、W3C推出XHTML 1.1

如果说XHTML 1.0是XML风格的HTML，**XHTML 1.1则是货真价实的XML**。这意味着XHTML 1.1无法使用 text/html MIME-type直接输出，然而，如果Web开发者使用XMLmime-type，则当时的主流浏览器IE则压根不支持。看上去，W3C似乎正在与当时的Web脱节。

# 五、出力不讨好的XHTML 2

对W3C而言，到了HTML 4已经是功德圆满，他们的下一步工作是**XHTML 2**，希望将Web带向XML的光明未来。虽然XHTML 2听上去和XHTML 1类似，它们却有很多差别，XHTML 2不向前兼容，甚至不兼容之前的HTML。它是一种全新的语言，赤条条来去无牵挂。这实在是一场灾难。

# 六、WHATWG：与W3C决裂

W3C闭门造车的作风引起了一些人的不满，来自**Opera（** 2016年2月360 和昆仑万维收购**）**，**Apple**，**以及Mozilla**的代表开始表达反对声音。2004年，Opera的 Ian Hickson提议在HTML基础上进行扩展以适应新的Web应用，该提议遭到W3C的拒绝。于是，他们自发组织成立了超文本应用技术工作组，就是**WHATWG**。

# 七、从WebApps1.0到HTML 5

从一开始，WHATWG就和W3C走不同的路线，W3C对问题的讨论是集体投票，而WHATWG则由主笔IanHickson定度。表面上看，W3C更民主，然而事实上，各种内部纷争会使一些决议限于泥潭，在WHATWG，事情的进展会更容易，不过，主笔的权力并非无限大，他们的委员会可以对那些过于偏执的主笔进行弹劾。

一开始，WHATWG的主要工作包括两部分，Web Forms 2.0和Web Apps 1.0，它们都是HTML的扩展，后来，他们合并到一起成为现在的HTML 5规范。在WHATWG致力于HTML 5的同时，W3C继续他们的XHTML 2.0，然而，他们慢慢地陷入困境。

2006年10月，Web之父Tim Berners-Lee发表了一篇博客文章，表示，从HTML走向XML的路是行不通的，几个月后，W3C组建了一个新的HTML工作组，他们非常明智地 选择了WHATWG的成果作为基础。这一转变带来一些困惑，W3C同时进行这两套规范，XHTML2和HTML 5（注意，W3C的HTTML5在5之前有个空格，而WHATWG的HTML 5则没有空格），而WHATWG也在进行着同样的工作。

# 八、XHTML已死：XHTML语法永存

这一混乱局面到了2009年开始变得清晰，W3C宣布终止XHTML2的工作，这是一份关于XHTML2的迟到的讣告。这一消息被那些XML的反对者视为珍宝，他们借此嘲笑那些使用XHTML1规范的人，然而他们似乎忘记了，XHTML1和XHTML2是截然不同的东西。于此同时，XHTML1规范的制定者担心，XHTML1中的严格语法规范会被HTML 5弃用，这种担心后来证明是多余的，HTML 5既支持松散语法，也支持XHTML1般的严格语法。

# 九、HTML5第一阶段： Web 增强与破垄断

自HTML5诞生以来，一共经历了两个阶段，分别是**Web增强和移动互联网**。我们先从Web 增强说起。

web体验的丰富增强主要表现在：1. webapp，比如gmail;2. 流媒体;3. 游戏。我们就这3个方面来讲HTML5做了什么。

## 1. webApp

HTML5新增了离线存储、更丰富的表单(比如Input type=date)、js线程、socket（套接字）、标准扩展embed、以及很多css3新语法…

## 2. 流媒体

HTML5新增了audio、video

## 3. 游戏

HTML5新增了canvas、webgl

当然还有Google努力在HTML5中推进Header和Section等标签，以利于搜索引擎分析，这些不多述。

HTML5补充流媒体和游戏能力后，加上苹果强势拒绝在iOS上引入Flash，成功的遏制了Flash的发展，然后就该遏制IE私有语法了。

在HTML5标准的升级过程中，苹果和Google同时也看到了浏览器市场重新洗牌的机会，他们一方面参与HTML5的规范，一边在浏览器产品上发力。Apple首先开始大力发展Safari，**建立WebKit开源项目**，Mac、iOS、Windows多平台齐发力；Google起初是赞助Mozilla开发Firefox，后来自己开发了v8引擎，合并WebKit，于2008年正式推出Chrome。“IE的私有规范+Flash不是标准，我们才是标准”这样的口号在新一代浏览器大战中打响，IE瞬间成为千夫所指的垄断代表，甚至成了阻碍Web发展的罪人(当时IE6已数年未更新，并且丝毫不惧Firefox的发展)。

偏偏微软此时也出了晕招，推出了一系列即不完整支持规范又互相不兼容的IE7、8、9、10，彻底失去了开发者的心。

Adobe的Flash被遏制，与Web霸主的位子擦肩而过;IE的私有标准被遏制，并且造成IE市场份额不停下滑，直到IE最新的移动版本反过来开始支持WebKit私有语法，真是令人唏嘘。不知道HTML6是不是该打倒WebKit垄断了。

# 十、HTML5第二阶段： 移动互联网

随着Chrome和Safari的高歌猛进，以及IE+Flash的衰落，HTML5告一段落，进入了下一个时代——**移动互联网**。HTML5的跨平台优势在移动互联网时代被进一步凸显。**HTML5是唯一一个通吃PC、Mac、iPhone、iPad、Android、Windows Phone等主流平台的跨平台语言。**Java和Flash都曾梦想这个位置，但梦断于iOS。此时人们纷纷开始研究基于HTML5开发跨平台手机应用。很多人当时认为，原生应用只是过渡，就像当年从C/S结构转变为B/S结构一样。而且学习Objective-C和Java很费劲，我既然会网页开发，为何不试试HTML5。

W3C此时成立了DeviceAPI工作组，为HTML5扩展了Camera、GPS等手机特有的API，然而麻烦的是，移动互联网初期的迭代太快了，手机OS在不停的扩展硬件API，陀螺仪、距离感应器、气压计。。。每年手机OS都有大版本更新。而W3C作为一个数百家会员单位共同决策的组织，从标准草案的提出到达成一致是非常复杂的过程，跟不上移动互联网初期的快速迭代。

PhoneGap的出现，给开发者打开了一扇窗。很多人期待PhoneGap不停扩展API，来补充浏览器的不足。Adobe看到PhoneGap仿佛看到了重振江湖地位的希望，但在Adobe收购PhoneGap后，又发现这个东西可商用性不足，而且开源使得Adobe无法像Flash那样获取商业利益，于是就把PhoneGap捐给了Apache，改名为Cordova。

因为各种原因，Cordova的定位最终没有成为浏览器的强化，而走向了混合式开发。基于当时的背景，他们认为原生是不可替代的，“原生+HTML5”的混合模式更有意义。所以现在Cordova的使用模型是“原生工程师+HTML5工程师”一起协作完成App。

这时Facebook加入了W3C，牵头成立了MobileWeb工作组。Facebook是混Web圈的，并且在手机OS上没有自己的领地，他不喜欢被苹果和Google掌控的原生应用生态系统。Mobile Web这个工作组的重要目标就是让HTML5开发的网页应用达到原生应用的体验。然而，事与愿违，它不努力也就算了，结果是努力了却失败了。2012年，Facebook放弃了HTML5的新闻充斥了全世界的IT媒体，HTML5瞬间被打入冷宫。

Facebook为何放弃HTML5？核心是当时基于HTML5真的做不出好的移动App。对比Twritter等竞争对手的原生App，Facebook的HTML5版本实在无法让用户满意。比如Push功能，到现在HTML5的推送和原生的推送体验差距依然巨大，更不用说HTML5应用的页面切换白屏、下拉刷新/侧滑菜单不流畅等众多问题。看着原生工程师轻松实现摇一摇、二维码、语音输入、分享到朋友圈等功能，更是让HTML5工程师感觉自己站错了队。

即使Facebook不喜欢被控制，也不能拿被用户抛弃来冒险。而且Facebook并没有掌握关键点—手机浏览器内核。如果浏览器不跟上，徒然定一堆标准草案落不了地。

而浏览器在手机上的表现是什么呢？先看Google，Chrome性能虽高，但Android上的浏览器却并非Chrome，而是WebKit改出来的一个蹩脚的Android浏览器;再看苹果，iOS上不允许其他浏览器引擎上架App Store，而且其他使用Safari引擎的应用也无法调用苹果自己的JavaScript加速引擎Nitro。结果是苹果和Google不但不在浏览器上积极实现HTML5关于移动App所需的规范，反而对HTML5做出种种限制。

不管是当时硬件能力不足，还是手机OS厂商的故意限制，总之结果就是：在移动互联网的初期，一定是原生应用生态系统的天下，iOS和Android首先自己的地盘稳固后，产业才会向下个阶段升级。

Facebook也好，PhoneGap也好，想在移动互联网初期就分一杯羹是分不到的，但坚持下来，机会往往会出现。

# 十一、 HTML5这回真的来了

**终于，在2014年10月底，W3C宣布HTML5正式定稿。这个时间，不晚不早，硬件性能更强、手机OS迭代速度下降。**

随着HTML5标准定稿，一切纷争将告一段落，现在，属于HTML5的时代到来了。

有人说，光标准定稿没用啊，配套起来了吗？HTML5做的应用究竟能否匹敌原生App？答案是，HTML5不但可以匹敌原生App，甚至它天然的很多特性超越了原生App。

我们先谈谈HTML5原来不如原生应用的地方，业内俗称HTML5有“性工能”障碍。即HTML5性能不如原生、开发工具不如原生、能力调用不如原生。

这几个问题导致开发者无法使用HTML5做出与原生一样的App。然而，不管是硬件升级还是OS厂商策略变化，以及相关软件技术的成熟，已解决了HTML5的“性工能”障碍。

## 1. 硬件升级

2011年，iPhone 4s的CPU是A5，现在iPhone 6是A8，按苹果的历次发布会的说法，速度共提升了7.5倍。这3年间7.5倍的速度提升，抹平了太多HTML5的性能问题。

## 2. 苹果、Google的策略变化

Google在2013年底发布的Android 4.4，内置的Webview不再是蹩脚的Android WebKit浏览器，而是Chromium，性能大幅提升。从最新的Android5.0开始，Webview可以通过Google Play Store实时更新，和Chrome的升级保持一致，用户就可以不刷机享受到最新的浏览器引擎;再看Apple方面，2012年iPhone 5发布后，HTML5在iOS上的表现已令人满意，Safari独家的JavaScript加速引擎Nitro不再那么重要，不过在iOS 8发布后，苹果还是很识趣地取消了三方程序调用Nitro的限制，现在任意浏览器或应用调用iOS的UIWebview都可以利用Nitro加速，这样在前端使用JS做大型运算也成为可能。两大手机操作系统霸主和浏览器巨头的态度发生了变化，使得HTML5在手机上的发展不再受限，而且这个变化不可逆只能继续向前，这种变化势必会产生深远的影响。

## 3. 软件技术的成熟

PhoneGap的发展虽然放缓了，但其他产品技术却成熟了。2014年的iWeb大会上，众多厂商的产品提供了面向开发者免费或开源的HTML5性工能障碍的解决方案。

DCloud公司在iWeb大会上发布了系统的HTML5“性工能缺失”的解决方案，包括：

a) 性能：提升HTML5性能的手机端引擎，让侧滑菜单、下拉刷新等动态交互卡顿的问题得以解决;

b) 工具：HTML5开发IDE产品HBuilder, 超快的编程利器;

c) 能力：把40万原生API封装成JavaScript对象，以解决HTML5能力不足问题的Native.js技术;

d) 最接近原生体验的高性能框架：MUI框架，体积只有几十K，加载、运行远快于一般框架。

基于该方案开发的HTML5应用完全可以达到原生App的功能和体验。

使用HBuilder开发HTML5应用

英特尔公司发布了Crosswalk引擎，可以让Android 4.0-4.3的手机上的应用打包Chromium引擎而不是Android WebKit。毕竟目前市场上存在大量Android 4.0-4.3的手机，同时统一的webview也避免了兼容性的烦恼。

在专业方向上很多公司也做出了不错的成绩。触控的Cocos2d-html5、Egret runtime和Ludei CocoonJS强化了Canvas的表现，让HTML5游戏体验更好;UC、猎豹等手机浏览器都强化了音视频播放的表现。

不管是硬件升级、软件成熟，还是操作系统厂商策略变化，都在强力推动HTML5的爆发。

不过要注意，我说的HTML5爆发，不是指手机浏览器会替代桌面成为应用入口。有人说HTML5不好，因为用户讨厌打开浏览器输入URL的过程。我想说这种想法是对HTML5的片面理解。HTML5!=传统浏览器，虽然编程语言还是HTML、Javascript、CSS，但发行方式绝不是传统网站那么简单。HTML5应用的入口，反而很少是启动浏览器输入URL，它可以是存在于手机桌面的图标、也可以来自超级App(如微信朋友圈)、以及搜索引擎、应用市场、广告联盟。。。到处都是它的入口。它的入口，比原生App更多。

# 十二、原生App的颠覆

HTML5的“性工能”障碍得到解决，可以接近原生App的效果，所以它就可以替代原生App吗？很多人认为，即使HTML5会发展的比现在好，也将是与原生App各占一部分市场的格局，要求不高的长尾应用会使用HTML5，而主流应用仍是原生App的天下。

但我认为这样的想法很危险，就像Apple成立前，HP的高层告诉沃兹：谁会在家里摆一台电脑呢?未来HTML5肯定会颠覆原生App。“性工能”障碍的消除，只是HTML5的劣势被削弱，但劣势被消除后，它的优势就会大放异彩，HTML5的优势是什么?我们分别就开发者和最终用户来看。

# 十三、HTML5对开发者的7大优势

## 跨平台：

在多屏年代，开发者的痛苦指数非常高，人人都期盼HTML5能扮演救星。多套代码、不同技术工种、业务逻辑同步，这是折磨人的过程。有点类似个人电脑早期世界，那个时候的每家电脑都有自己的操作系统和编程语言，开发者疲于做不同版本，其实DOS的盛行也很大程度是因为开发者实在没精力给其他电脑写程序。跨平台技术在早期大多因为性能问题夭折，但中后期硬件能力增强后又会占据主流，因为跨平台确实是刚需。

## 快速迭代：

移动互联网是一个快鱼吃慢鱼的时代，谁对用户的需求满足的更快，谁的试错成本更低，谁就拥有巨大的优势。互联网产品大多免费、且有网络效应，后入者抢夺用户的难度非常大。使用原生开发，从招聘、开发、上线各个环节的效率都慢一倍以上，而且参与的人越多，沟通效率往往拖慢不止一倍。

## 持续交付：

很多人有这样的体会，一个原生应用上线Appstore，突然有一个大bug，只好连夜加班修复，然后静静等待2周或更长时间的Apple审核，这2个星期被用户的涂抹淹死，市场上一片差评，用户大量流失。等新应用被审核上线了，用户已经卸载了。但是，HTML5没有这些问题，你可以实时更新，有问题立即响应。

## 大幅下降成本：

创业者融资并不容易，如何花钱更高效非常重要。如果你使用原生开发的App和竞争对手使用HTML5开发的App没什么区别，但你的开发成本高出一倍，我相信没有投资人会喜欢给你投钱。

## 开源生态系统发达：

HTML5前端是开放的正反馈循环生态系统，大量的开源库可以使用，开发应用变得更轻松、更敏捷，当然这也体现在了快速迭代和成本下降上。不过更重要的是，这种开放的正反馈循环生态系统未来的生命力是比原生生态系统更强劲的。

## 开放的数据交换：

HTML是以page为单元开放代码的，它无需专门开发SDK，只要不混淆，就能与其他应用交互数据。开发者可以让手机搜索引擎很容易检索到自己的数据， 也更容易通过跨应用协作来满足最终用户需求。

## 更容易推广、更容易爆发：

导流入口多：HTML5应用导流非常容易，超级App(如微信朋友圈)、搜索引擎、应用市场、浏览器，到处都是HTML5的流量入口。而原生App的流量入口只有应用市场。聪明的HTML5开发者当然会玩转各种流量入口从而取得更强的优势。

流量大：前段时间微信朋友圈风靡一时《神经猫》，这个游戏如果放到Appstore，绝对没有那么多流量，超级App带来的流量，远大于原生应用市场。假如微信允许游戏在桌面创建快捷方式、假如游戏后续升级解决持续娱乐问题，未来不可想象。

导流效率高：除了入口多、流量大，导流效率高也不可忽视，谁都知道：页游和端游打同样的广告，广告变用户的转化率，页游远远高于端游。

可精准导流到二级页：我们都知道搜索引擎可以直接进入到

# 十四、HTML5对最终用户的3大优势

## 大幅降低使用门槛

为什么流媒体会替代下载视频成为主流?为什么页游会如此火爆?只因用户太“懒”。让用户更方便的满足需求，有时效果好于更多的满足需求。

用户眼睛看到一个兴趣点，点击后，就应该立即开始满足用户需求。比如流媒体可以立即看，页游可以立即玩。而目前的原生应用市场，用户需要这样操作：选一个应用、等待下载、确认权限、等待安装，然后点击打开。这样糟糕的体验迟早要被颠覆。

不管是App、游戏还是音视频，未来都将即点即用。谁先满足用户这个需求，谁就制胜。

## 实时更新、差量更新的优秀体验

HTML5应用可以绕开应用市场的限制进行自主实时更新，用户可以快速享受新服务。

而且这种更新完全可以是差量更新，比如某个HTML页面或某个js文件有问题，只更新这个几k的小文件就可以了，这比原生应用的更新体验好太多。

## 跨应用的使用体验

目前手机应用切换是以桌面或任务管理器为中心的，但事实上这些中心很影响效率和体验。用户想出差三亚，先打开去哪App订票，然后切回桌面，再找到并打开天气App，搜索输入三亚，再切到桌面，找到并打开航旅纵横App，输入航班号值机，哦对了，航班号多少来着，再切到桌面，找到并打开去哪App看航班号，最后找到并打开租车App，输入租车地点，然后再切回桌面。。。

在原生应用体系下，用户只能这样。但在HTML5体系下，他不需要切回桌面，他可以在App间方便的直接跳来跳去，而不是使用一个一个孤岛App;他更不用重复录入数据，应用间可以方便的互相传递数据。

这种模式需要一点想象力，但未来迟早会来。

分析至此，我们可以明显的看出，不管是站在最终用户角度、还是站在开发者角度，HTML5必将取代原生应用当前的位置。并由此引发一系列颠覆。

# 十五、还有什么会被改变?

HTML5的爆发，原生App生态系统的颠覆，是一场产业革命，很多角色都会受到影响，我们来预测一番。

## 新型HTML5引擎战火将烧起

标准的HTML5引擎并不能解决HTML5的所有问题，拥有大流量入口的互联网巨头，莫不在思考内嵌更优秀的增强引擎。腾讯推出了X5浏览器引擎，就是看中这个机会。目前各路浏览器厂商、应用市场厂商、甚至rom厂商，都在努力整合更优质的浏览器引擎。假使微信内嵌的Webview可以运行更优秀的canvas游戏、假使360手机助手可以发行即点即用的HTML5应用并且能力体验与原生一致、假使小米rom内置更强大的webview使得所有HTML5应用在小米手机上运行的更流畅。。。

一个巨头开始行动，所有巨头都会闻风而动，没错，这场战役会是移动互联网世界的二次世界大战。

## 应用发行市场将洗牌

由于超级App的巨大流量能轻易成为HTML5应用的入口，并且会形成大者更大的效应，传统的应用商店、甚至线下预装，这些流量不足和效率偏低的发行模式将被挤出市场主流。本身也是超级App的大流量应用商店，如果转型得当，也将以发行HTML5应用为主。

## 广告和统计市场

原生的广告和统计SDK提供商会面临尬尴，Google、百度等基于网页的广告和统计服务会取得更大的优势。开发者不再需要打包SDK，引入一个Script即可。

## 开源技术将在移动互联网领域更加流行

HTML的开放性造就了大量的开源产品，也反向促进了HTML的繁荣。在Github上有大量的JS框架，而原生的开源代码数量相比甚少。而未来移动互联网世界将因为开源而发展的更迅速，这里也同样存在类Github厂商的机遇。

## 开发工具的变化

早期HTML只需要记事本写几个Tag，中期的HTML、JS、CSS比较复杂，需要更高级的文本编辑器，但HTML5到来后，它的代码量、复杂度、开发模型将与原生开发看齐，需要类似XCode、Eclipse等专业的IDE工具来解决开发、调试的问题。一些以会使用记事本写代码为荣的开发者，将面临思路转换甚至被更高效的开发者淘汰。

## 性能分析调优

目前很多针对原生应用的性能分析调优工具或服务，未来也面临转型，HTML5应用的性能分析调优是另一个世界。

## 混淆与产权保护

HTML5是开放代码的，好处也带来弊端，有些东西开发者希望暴露，但有些东西开发者希望保护。混淆技术就变得更有商业机会。PC Web上Gmail的混淆就做的不错。除了JS混淆，离线数据加密相信也有不少空间。

## 安全厂商的新机会

HTML5的强大会引发很多安全问题，并且解决思路与原生不一样，业内有可能会出现新的安全厂商领导者。