**基本目录结构**

一个基本的 HTML5 Boilerplate 站点，最初的目录结构就像这样：

.

├── css

│ ├── main.css

│ └── normalize.css

├── doc

├── img

├── js

│ ├── main.js

│ ├── plugins.js

│ └── vendor

│ ├── jquery.min.js

│ └── modernizr.min.js

├── .editorconfig

├── .htaccess

├── 404.html

├── apple-touch-icon.png

├── browserconfig.xml

├── index.html

├── humans.txt

├── robots.txt

├── crossdomain.xml

├── favicon.ico

├── tile-wide.png

└── tile.png

下面是有关该目录的组成及使用方式的整体概述：

**css**

/css 文件夹保存项目中的所有CSS文件。文件夹中包含了一些用于初始化的 CSS，为开发者在构建项目前提供标准化的开发环境。

**doc**

/doc 文件夹保存 HTML5 Boilerplate 的所有开发文档。可以使用它作为开发者个人项目的基础文档。

**js**

/js 文件夹保存项目中的所有 js 文件。库、插件和自定义代码都可以放在这里，此外还包括了一些用于初始化的 js，以帮助开发者构建项目。

**.htaccess**

该文件默认为 Apache 的网络服务器配置信息。更多信息，请参考 [Apache Server Configs repository](https://github.com/h5bp/server-configs-apache)。

如果网站托管的服务器不是 Apache，那么可以在 [Server Configs](https://github.com/h5bp/server-configs/blob/master/README.md)项目中找到相关服务器的配置信息。

**404.html**

一个自定义的404页面,有益于开发者快速开始构建项目。

**browserconfig.xml**

该文件包含 IE11 动态瓷贴的设置信息。

更多信息，请参考[MSDN](http://msdn.microsoft.com/en-us/library/ie/dn455106.aspx)。

**.editorconfig**

该文件的作用在于，帮助和激励开发者及其团队，在编辑器和 IDE 之的范。

**index.html**

该页面包含默认的 HTML 结构，建议开发者使用它作为页面的基础结构。如果开发者正在使用一个服务器端的模版引擎，那么可能需要在项目构建之初，将该页面结构整合入模版引擎。

如果开发者修改了目录结构，请确保同时更新了 CSS 和 JavaScript 的引用地址。

如果开发者使用了谷歌分析脚本（Google Analytics），请确保网页底部相关脚本的 Analytics ID 书写正确。

**humans.txt**

该文件记录网站、应用程序的开发团队及开发技术等信息。

**robots.txt**

该文件记录对搜索引擎屏蔽的页面。

**crossdamin.xml**

一个用作跨域请求的模板。

**Icons**

可以根据需要替换默认的 favicon.ico，tile.png，tile-wide.png 和 Apple Touch Icon。

更多信息，可以参考  Hans 的这篇文章——[HTML5 Boilerplate Favicon and Apple Touch Icon PSD-Template](https://drublic.de/blog/html5-boilerplate-favicons-psd-template/" \t "_blank)。

**HTML 解析**

HTML5 Boilerplate 默认提供两个 html 页面：

* index.html
* 404.html

**index.html**

**no-js类**

no-js 类可以让开发者依据 JavaScript 被启用（.js）或被禁止(.no-js)两种状态，更轻易准确地添加自定义样式。

使用这个技巧也有助于[避免浏览器闪烁（FOUC）](http://paulirish.com/2009/avoiding-the-fouc-v3/)。

**lang 属性**

请认真考虑在 <html> 中添加 lang 属性，从而对页面内容所使用的语种加以说明，举例如下：

<html class="no-js" lang="en">

**<title> 和 <meta> 标签的顺序**

置顶 <title> 和 <meta> 标签的顺序意义重大，原因如下：

**字符集声明（<meta charset="utf-8">）：**

* 该声明必须完整地包含在[文档内容最开始的 1024 字节中](https://www.whatwg.org/specs/web-apps/current-work/multipage/semantics.html#charset)。
* 该声明应该尽早出现（早于可能被攻击者利用的任何内容，比如 <title> 元素）,以避免 IE 潜在的[编码安全问题](https://code.google.com/p/doctype-mirror/wiki/ArticleUtf7" \t "_blank)。

**兼容模式的元标签（<meta http-equiv="x-ua-compatible" content="ie=edge">）：**

除<title> 和其他 <meta> 标签外，[该标签必须在其他标签之前声明](http://msdn.microsoft.com/en-us/library/cc288325.aspx" \t "_blank)。

**x-ua-compatible**

IE 8/9/10 支持[文档兼容模式](http://msdn.microsoft.com/en-us/library/cc288325.aspx" \t "_blank)——该模式会对文档解析和页面渲染产生影响。因此，即使网站的访问者使用了 IE 9 或更高版本的浏览器，IE 也有可能不使用最新的渲染引擎，而会使用 IE 5.5 渲染引擎解析你的页面。

x-ua-compatible 元标签的详细内容如下：

<meta http-equiv="x-ua-comptible" content="ie=edge">

此外，可以使用 HTTP 响应头信息 x-ua-comptible: ie=edge 来发送网页数据。这种方式将强制 IE 8/9/10 使用最新的可用模式来渲染页面，[即使某些情况下并不适合使用该模式](https://hsivonen.fi/doctype/" \l "ie8" \t "_blank)。因此，要确保浏览该网站的所有浏览器，都尽可能提供最佳的用户体验。

如有可能，我们建议移除 meta 标签，而只发送 HTTP 响应头信息。这么做的原因是，如果你的网站运行在非标准的端口上， IE 默认会 “在兼容性视图下显示内部网站”，而此时 meta 标签就不会起作用了。

如果你正在使用 Apache 作为网络服务器，那么可以使用 [.htaccess](https://github.com/h5bp/server-configs-apache) 文件来配置 HTTP 响应头信息。如果是其他网络服务器，可以点击这里查看[其他服务器配置信息](https://github.com/h5bp/server-configs" \t "_blank)。

从 IE 11 开始，[文档模式已经被弃用了](http://msdn.microsoft.com/en-us/library/ie/bg182625.aspx" \l "docmode" \t "_blank)。如果你的业务仍然依赖于陈旧的网页应用，或者你的业务就是为老版本 IE 设计的，那么可以考虑在全公司使用 [企业版模式（Enterprise Mode）](http://blogs.msdn.com/b/ie/archive/2014/04/02/stay-up-to-date-with-enterprise-mode-for-internet-explorer-11.aspx)。

**移动端视图**

使用 [viewport 元标签](https://docs.google.com/present/view?id=dkx3qtm_22dxsrgcf4)时有些许不同。更多信息详见[苹果开发者文档](https://developer.apple.com/library/safari/documentation/AppleApplications/Reference/SafariWebContent/UsingtheViewport/UsingtheViewport.html)。HTML5 Boilerplate 初始化了一些简单的设置，希望在多种使用情境下取得良好平衡。

<meta name="viewport" content="width=device-width, initial-scale=1">

**Favicons 和 Touch Icon**

整站的快捷方式图标应该存放在根目录。HTML5 Boilerplate 默认提供了一组图标（包括 favicon 和 Apple Touch Icon）,为开发者提供设计参考。

**Modernizr**

HTML5 Boilerplate 使用了一份自定义的 Modernizr。众所周知，[Modernizr](http://modernizr.com/" \t "_blank) 是一个 JavaScript 库，作用是确保所有浏览器都可以使用 HTML 5 元素（它包含了 HTML 5 shiv），同时还会根据功能检测的结果为 html 元素添加不同的类名。便于开发者明确某个浏览器所支持的 CSS 和 JavaScript 特性。

通常来说，为了最大程度地减少页面加载时间，最好的方式是在页面底部调用 JavaScript，以此避免加载外部脚本的过程阻塞整个网页的渲染和解析进度。但是，Modernizr 脚本应该在浏览器渲染页面前就加载执行，这样才能让浏览器正确处理尚未支持的 HTML5 元素。因此，Modernizr 脚本应该是唯一可以在文档头部加载的 JavaScript 脚本。

**腻子脚本**

如果你需要在项目中加载 [腻子脚本（polyfills）](https://remysharp.com/2010/10/08/what-is-a-polyfill) ，那么你必须确保腻子脚本在其他 JavaScript 之前加载。如果你正在使用类似[cdn.polyfill.io](http://cdn.polyfill.io/" \t "_blank)的 ployfill CDN 服务，只需在页面底部将其放在其他脚本之前即可。

<script src="//cdn.polyfill.io/v1/polyfill.min.js"></script>

<script src="//ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>

<script>window.jQuery || document.write('<script src="js/vendor/jquery-1.11.2.min.js"><\/script>')</script>

<script src="js/plugins.js"></script>

<script src="js/main.js"></script>

如果只是想方便自己使用腻子脚本，可以将它们包含到 js/plugins.js 中。当你有了一堆腻子脚本需要加载时，可以在 js/vendor 文件夹创建一个 polyfills.js 文件。当然使用这个技巧也要注意，确保腻子脚本在其他 JavaScript 脚本前完全加载。

有些人对于 Modernizr 和腻子脚本的认知存在一些误区。正确理解 Modernizr 很重要：它只进行功能检测而不执行功能修复。Modernizr 唯一关注的修复工作，是[有关跨浏览器方面腻子脚本的冗长列表](https://github.com/Modernizr/Modernizr/wiki/HTML5-Cross-Browser-Polyfills" \t "_blank)。

**内容区**

HTML5 Boilerplate 模版的内容基本是空的。这么做是有意而为的，以方便 Boilerplate 模版同时适配网页（web page）和网页应用（web app）的开发。

**浏览器升级提示**

Boilerplate 的内容区包含了一条提示，建议 IE 6/7 的用户安装最新版本浏览器。如果你打算支持 IE 6/7，那么你需要移除相关代码。

**由 Google CDN 分发的 jQuery**

Google CDN 提供的 jQuery ，使用 protocol-independent 路径放置在了页面底部。此外还设置了一个本地的 jQuery 版本，从而在 CDN 版本不可用或启用离线版本的时候调用。

Google CDN 的版本之所以能从[众多的可选版本（比如 jQuery CDN）](https://jquery.com/download/" \l "using-jquery-with-a-cdn" \t "_blank)脱颖而出，一方面是因为它响应速度快，另一方则是因为它具有最佳的[穿透力（获取的成功率）](http://httparchive.org/trends.php" \l "perGlibs" \t "_blank)，这有助于提高将库文件传输到用户浏览器缓存的成功率。

虽然 Google CDN 是模版中开发网页和应用默认的分发网络，但你可能还需要做进一步的配置。你可以使用类似 [WebPageTest](http://www.webpagetest.org/" \t "_blank) 的服务和类似 [PageSpeed Insights](https://developers.google.com/speed/pagespeed/insights/" \t "_blank) 或 [YSlow](https://developer.yahoo.com/yslow/" \t "_blank) 的浏览器工具测试网站，这有助于帮你测试网站的实际表现，并且指出特定网页或应用有待优化的地方。

**谷歌分析代码**

最后，是一段谷歌分析代码。谷歌建议将该脚本置于文档头部。其考虑如下：如果将该脚本置于网页头部，那么即使页面未完全加载，也可以统计用户访问量，并且可以激发浏览器的最大并发连接数。

更多信息请参考：

* [谷歌通用分析代码优化](https://mathiasbynens.be/notes/async-analytics-snippet#universal-analytics)
* [Analytics.js 入门](https://developers.google.com/analytics/devguides/collection/analyticsjs/)
* [谷歌分析示例和工具](https://ga-dev-tools.appspot.com/)

**注意**：之所以默认包含谷歌分析代码，是因为谷歌分析是当前[最流行的追踪解决方案之一](https://trends.builtwith.com/analytics/Google-Analytics)。然而，它的用法并不是一成不变的，你应该查看[可替代方案](https://en.wikipedia.org/wiki/List_of_web_analytics_software)，以选择最适合自己的。

**CSS 解析**

模版中默认的 CSS 文件并不依赖 [conditional class names](http://www.paulirish.com/2008/conditional-stylesheets-vs-css-hacks-answer-neither/)，[conditional style sheets](http://css-tricks.com/how-to-create-an-ie-only-stylesheet/" \t "_blank)，或 [Modernizr](http://modernizr.com/" \t "_blank) 的表现效果，尽可以放心按自己的偏好放手去干。

**Normalize.css**

为了确保所有浏览器的渲染效果更加一致且符合标准，我们引入了 [Normalize.css](https://necolas.github.io/normalize.css/)——这是一个现代化、紧密结合 HTML 5 的可选方案，可以用来实现浏览器的 CSS 重置。

Normalize.css 本质上并不是重置 CSS：

* 只处理需要标准化修正的样式
* 保留浏览器已有的默认样式，而不是全盘替换它们
* 修正浏览器的缺陷和常见性差异
* 通过细微的改进提高可用性
* 不干扰调试工具
* 拥有良好的开发文档

更多详情请参考[项目网站](https://necolas.github.com/normalize.css/" \t "_blank)，或者[博客文章](http://nicolasgallagher.com/about-normalize-css/)。

**默认样式**

基于 Normalize.css，编写了一些基础的样式，具体如下：

* 提供了基础的排版样式，以改善文本可读性
* 文本高亮时，默认取消 text-shadow
* 修改了一些元素默认的对齐方式，比如: img，video，fieldset，textarea
* 修改了老版本浏览器的提示符样式

非常激励开发者根据实际需求和添加样式到默认样式中版。

**通用样式**

在基础样式之外，我们也提供了一些通用的辅助样式。

**.hidden**

如果你想在视觉上隐藏任何元素，或是对屏幕阅读器屏蔽某个元素的内容，那么都可以为其添加 hidden 样式。这些元素可以是后续填充或显示的，也可以是使用 JavaScript 隐藏的。

**.visuallyhidden**

如果你想在视觉上隐藏任何元素，那么可以为其添加 .visuallyhidden 样式，但此时屏幕阅读器仍然可以识别它。

更多信息可参考：

* [CSS in Action: Invisible Content Just for Screen Reader Users](http://www.webaim.org/techniques/css/invisiblecontent/)
* [Hiding content for accessibility](http://snook.ca/archives/html_and_css/hiding-content-for-accessibility)
* [HTML5 Boilerplate - Issue #194](https://github.com/h5bp/html5-boilerplate/issues/194/)

**.invisible**

如果你想在视觉上隐藏任何元素，或是对屏幕阅读器屏蔽某个元素的内容，那么都可以为其添加 .invisible 样式，但这种方式并不会影响页面布局。

与 hidden 样式从布局中移除元素的方式不同，invisible 样式只会让元素不可见，不会影响其在文档流中的位置，也不会影响其附近元素的位置。

**注意**：绝不应该使用上述样式进行[关键字堆彻](https://en.wikipedia.org/wiki/Keyword_stuffing)，那必将有损网站在搜索引擎中的排名。

**.clearfix**

给任何元素添加 clearfix 样式，可以确保它始终包裹浮动的子元素。

过去几年，出现了 clearfix hack 的多种变体，但在这里，我们使用 [micro clearfix](http://nicolasgallagher.com/micro-clearfix-hack/)。

**媒体查询**

使用 HTML5 Boilerplate 可以很轻松地进行 [移动优先](http://www.lukew.com/presos/preso.asp?26) 和 [响应式设计](http://www.alistapart.com/articles/responsive-web-design/) 的开发。但有必要提醒的是，这里面没有[银弹（引申含义：具有极端有效性的解决方法）](http://www.cloudfour.com/css-media-query-for-mobile-is-fools-gold/" \t "_blank)。

我们插入了媒体查询的语句，帮助你构建宽幅和高分辨率的移动样式。建议根据网站内容调节断点，而不是简单参考具体设备的固定尺寸。

如果并不需要注重移动优先原则，那么你也可以修改或移除媒体查询的语句。当视窗宽度确定下来后，你可以使用 max-width 来替代固定尺寸，比如这样来编写：@media only screen and (max-width: 480px)。

更多有关移动开发的信息，请看 [Mobile Boilerplate](https://github.com/h5bp/mobile-boilerplate)。

**印刷样式**

最后，我们提供了一些有用的印刷样式，以优化印刷流程，同时还可以提高印刷品的可读性。

印刷时，这些样式会：

* 去除所有背景色，将字体颜色改为黑色，移除 text-shadow——[以节省印刷颜料并加快印刷进程](http://www.sanbeiji.com/archives/953" \t "_blank)。
* 为链接添加下划线并附加 URL 地址——以方便用户了解具体的引用地址。不过有两种例外：[片段标示符（比如 #href）](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/a" \l "attr-href" \t "_blank) 和 [javascript: 伪协议](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/void" \l "JavaScript_URIs" \t "_blank))。
* 展开缩写，方便用户了解缩写的具体含义。
* 指定了浏览器的分页方式，以及分页时[页面顶部和底部的最少行数](https://en.wikipedia.org/wiki/Widows_and_orphans" \t "_blank)。所有[支持该属性的浏览器](https://en.wikipedia.org/wiki/Comparison_of_layout_engines_%28Cascading_Style_Sheets%29#Grammar_and_rules)都会有如下解析：

1. 确保表格在每个分页都有表头（<thead>）
2. 防止块引用、预格式文本、图片和表格分页时被截断而进入不同页面。
3. 除相关页面外，确保标题不会出现在其他页面
4. 确保 [orphans and widows](https://en.wikipedia.org/wiki/Widows_and_orphans) 属性[不会出现在印刷品上](http://css-tricks.com/almanac/properties/o/orphans/" \t "_blank)。

该印刷样式与其他 CSS 样式被包裹在了一起，以[减少 HTTP 请求](http://www.phpied.com/delay-loading-your-print-css/" \t "_blank)。同样，它们应该始终被置于样式表的尾部，方便覆盖其他样式。

**JS 解析**

**main.js**

该文件可以用来包含或引用网站/应用程序内的 JavaScript 代码。对于大型项目，可以使用一个类似 [Require.js](http://requirejs.org/) 的模块加载器，以方便加载其他脚本。

**plugins.js**

该文件可以用来包含所有的插件，比如 jQuery 插件和其他第三方插件。

这里面的一个技巧是，将 jQuery 插件放入 (function($){ ...})(jQuery); 闭包中，确保它们安全地处于 jQuery 命名空间下。更多信息请参考 [jQuery 插件开发文档](http://docs.jquery.com/Plugins/Authoring#Getting_Started)。

plugins.js 文件默认保存了一小段代码，防止浏览器禁用 console 而引发的 console 错误。如果控制台方法不可用，那么这段代码将确保相应的方法为空函数值，由此，防止浏览器报错。

**vendor**

该目录可以用来保存所有的第三方库。最新的 jQuery 和 Modernizr 压缩版本就默认保存在这里。

**扩展和定制**

这里是优化 HTML5 Boilerplate 主题的一些建议。因为并不是所有的特性都适合具体的需求，所以我们没有默认导入这些特性。

**App Stores**

**安装 Chrome 插件**

用户可以直接从网站安装 Chrome 应用，前提是该应用已经通过谷歌的网络管理工具接驳了网站。更多信息详见[内置 Chrome 商店的开发文档](https://developer.chrome.com/webstore/inline_installation" \t "_blank)。

<link rel="chrome-webstore-item" href="https://chrome.google.com/webstore/detail/APP\_ID">

**iOS 6+ Safari 中的 Smart App Banners**

不用再向用户介绍复杂的方式进入 App Store 获取相关的应用程序了。引用下面的[元标签](https://developer.apple.com/library/IOS/documentation/AppleApplications/Reference/SafariWebContent/PromotingAppswithAppBanners/PromotingAppswithAppBanners.html" \l "//apple_ref/doc/uid/TP40002051-CH6-SW2" \t "_blank)，将会给用户提供更友好的方式下载你的 iOS App，或者用来用户的当前状态优化网站体验。

<meta name="apple-itunes-app" content="app-id=APP\_ID,app-argument=SOME\_TEXT">

**DNS 预加载**

简而言之，DNS 预加载是告知浏览器当前网站域名的一种方法，这样一来客户端才能解析拥有该 DNS 的主机，缓存这些网站，当再次需要使用时，请求速度就会更快。

**隐式预加载**

浏览器会自动为用户预加载诸多信息。当浏览器在 html 中检测到一个链接标签时，并不会分配给它浏览器当前请求的域名，而是从客户端系统，根据 IP 地址分配域名。客户端首先检测缓存，如果缓存中不存在，则从 DNS 服务器发送请求。这些请求发生在后台，并不会阻塞页面渲染进程。

采用这种方式，可以让需要的外部 IP 地址预加载到客户端缓存中，同时又不会阻塞外部内容的加载。请求越少，则页面渲染越快。在移动端这种感觉会更明显，因为移动端的延迟更强烈。

**禁用隐式预加载**

<meta http-equiv="x-dns-prefetch-control" content="off">

即使禁用 X-DNS-Prefetch-Control 元标签或者 HTTP 头信息，浏览器也会预加载所有显式的 dns-prefetch 链接。

**注意**：如果你的网站依赖于外部域名的资源，那么这种做法将会降低加载速度。

**显式预加载**

通常，浏览器扫描 html 只会预加载外部域名。如果所需资源在当前 html 之外（比如说，需要请求一个远程服务器的 javascript，或是需要一个存储了所有网页信息的 CDN）， 那么你就可以将需要预加载的域名全部列出来。

<link rel="dns-prefetch" href="//example.com">

<link rel="dns-prefetch" href="//ajax.googleapis.com">

如果有很多外部域名需要请求，建议使用上述方式。如果能够将它们编写在[Meta Charset](https://developer.mozilla.org/en-US/docs/Web/HTML/Element/meta" \l "attr-charset" \t "_blank) 元素之后就更棒了，这样浏览器将会尽快加载它们。

**常用预加载链接**

Amazon S3:

<link rel="dns-prefetch" href="//s3.amazonaws.com">

Google APIs:

<link rel="dns-prefetch" href="//ajax.googleapis.com">

Microsoft Ajax Content Delivery Network:

<link rel="dns-prefetch" href="//ajax.microsoft.com">

<link rel="dns-prefetch" href="//ajax.aspnetcdn.com">

更多 DNS 预加载信息请参考：

* <https://developer.mozilla.org/en-US/docs/Controlling_DNS_prefetching>
* <https://dev.chromium.org/developers/design-documents/dns-prefetching>
* <http://blogs.msdn.com/b/ie/archive/2011/03/17/internet-explorer-9-network-performance-improvements.aspx>
* <http://dayofjs.com/videos/22158462/web-browsers_alex-russel>

**谷歌通用分析**

**更多追踪设置**

HTML5 Boilerplate 内部优化过的谷歌通用分析代码段[optimized Google Universal Analytics snippet](https://mathiasbynens.be/notes/async-analytics-snippet" \l "universal-analytics" \t "_blank)的内容大致如下：

ga('create', 'UA-XXXXX-X', 'auto'); ga('send', 'pageview');

如果想要进一步修改，可以查看谷歌的[高级设置](https://developers.google.com/analytics/devguides/collection/analyticsjs/advanced" \t "_blank)， [Pageview](https://developers.google.com/analytics/devguides/collection/analyticsjs/pages" \t "_blank)，和 [Event](https://developers.google.com/analytics/devguides/collection/analyticsjs/events) 开发文档。

**隐藏 IP 地址**

在某些国家，如果两个司法管辖区之间没有相同严厉的法律，那么个人信息就不允许在相互间传递，比如从德国向欧盟之外传送。因此，网络管理员需要确保使用谷歌通用分析时，不能将个人信息从德国传到美国。开发者可以在发送 events/pageviews 前，设置 [ga('set', 'anonymizeIp', true);](https://developers.google.com/analytics/devguides/collection/analyticsjs/advanced" \l "anonymizeip" \t "_blank)。

ga('create', 'UA-XXXXX-X', 'auto');

ga('set', 'anonymizeIp', true);

ga('send', 'pageview');

**追踪 jQuery AJAX 请求**

这里有一篇来自 Jango Steve 的文章，介绍了[如何在谷歌分析中追踪 jQuery AJAX 请求](http://www.alfajango.com/blog/track-jquery-ajax-requests-in-google-analytics/" \t "_blank)。

在 plugins.js 中添加如下代码：

*/\**

*\* Log all jQuery AJAX requests to Google Analytics*

*\* See: http://www.alfajango.com/blog/track-jquery-ajax-requests-in-google-analytics/*

*\*/*

if (typeof ga !== "undefined" && ga !== null) {

$(document).ajaxSend(function(event, xhr, settings){

ga('send', 'pageview', settings.url);

});

}

**追踪 JavaScript 错误**

在 ga 定义之后，添加如下函数：

(function(window){

var undefined,

link = function (href) {

var a = window.document.createElement('a');

a.href = href;

return a;

};

window.onerror = function (message, file, line, column) {

var host = link(file).hostname;

ga('send', {

'hitType': 'event',

'eventCategory': (host == window.location.hostname || host == undefined || host == '' ? '' : 'external ') + 'error',

'eventAction': message,

'eventLabel': (file + ' LINE: ' + line + (column ? ' COLUMN: ' + column : '')).trim(),

'nonInteraction': 1

});

};

}(window));

**追踪页面滚动**

在 ga 定义之后，添加如下函数：

$(function(){

var isDuplicateScrollEvent,

scrollTimeStart = new Date,

$window = $(window),

$document = $(document),

scrollPercent;

$window.scroll(function() {

scrollPercent = Math.round(100 \* ($window.height() + $window.scrollTop())/$document.height());

if (scrollPercent > 90 && !isDuplicateScrollEvent) { //page scrolled to 90%

isDuplicateScrollEvent = 1;

ga('send', 'event', 'scroll',

'Window: ' + $window.height() + 'px; Document: ' + $document.height() + 'px; Time: ' + Math.round((new Date - scrollTimeStart )/1000,1) + 's'

);

}

});

});

**Internet Explorer**

**在 IE 10 中提示用户切换到“桌面模式”**

在 Metro 模式下，IE 10 并不支持插件，比如 Flash。如果你的网站需要使用插件，那么可以使用 x-ua-comptible 元标签，提醒用户切换到桌面模式。

<meta http-equiv="x-ua-comptible" content="requiresActiveX=true">

下面是 HTML5 Boilerplate 中 x-ua-comptible 的默认值：

<meta http-equiv="x-ua-comptible" content="ie=edge,requiresActiveX=true">

更多信息请参考 [Microsoft's IEBlog post about prompting for plugin use in IE10 Metro Mode](http://blogs.msdn.com/b/ie/archive/2012/01/31/web-sites-and-a-plug-in-free-web.aspx)。

**IE 9+ 固定网站**

如果启用了固定功能，那么 IE 9 的用户就可以将应用程序添加到任务栏或者开始菜单。这一功能也带来了一系列的工具，方便用户对元素进行个性化配置。更多信息请参考 [documentation on IE9 Pinned Sites](http://msdn.microsoft.com/en-us/library/gg131029.aspx)。

**为固定的网站命名**

如果不使用这条标签，Windows 将会使用页面标题作为应用程序的标题。

<meta name="application-name" content="Sample Title">

**为固定的网站添加提示**

如你所知，这里讲的是一个提示工具。当用户将鼠标悬停在固定网站的图标上边时，就会出现一个预览窗口。

<meta name="msapplication-tooltip" content="A description of what this site does.">

**为固定的网站设置默认页面**

如果当网站固定时需要制定一个特定的 URL（比如首页），那就在这里设置。一个好的想法是让固定网站发送特别编写的 URL，这样开发者就可以追踪到有多少用户使用了固定网站，就像下面这样：

<meta name="msapplication-starturl" content="http://www.example.com/index.html?pinned=true">

**自定义 IE 的控制按钮色彩**

IE 9+ 会自动使用固定网站图标的颜色，给浏览器按钮添加相应的阴影颜色。除非开发者自定义其他色彩，同时自定义色彩只能使用颜色关键字 (red) 和十六进制色彩 (#ff0000)。

<meta name="msapplication-navbutton-color" content="#ff0000">

**自定义窗口大小**

如果固定网站打开时需要使用确定的大小，那么可以在这里自定义尺寸。该功能只支持静态的像素尺寸，且最小为 800x600。

<meta name="msapplication-window" content="width=800;height=600">

**添加跳转列表**

开发者可以给固定网站添加跳转列表，当鼠标点击右键时，就可以快速显示跳转列表了。每个列表项都指向特定的 URL，并拥有自己的图标（通常为 16x16 的图标）。开发者可以添加任意数量的列表项。

<meta name="msapplication-task" content="name=Task 1;action-uri=http://host/Page1.html;icon-uri=http://host/icon1.ico">

<meta name="msapplication-task" content="name=Task 2;action-uri=http://microsoft.com/Page2.html;icon-uri=http://host/icon2.ico">

**（Windows 8)高级质量视觉效果**

Windows 8 允许开发者提供一张 PNG 瓷贴图片，也允许自定义瓷贴的背景色。[Full details on the IE blog](http://blogs.msdn.com/b/ie/archive/2012/06/08/high-quality-visuals-for-pinned-sites-in-windows-8.aspx" \t "_blank)。

* 为网站创建一个 144x144 分辨率的图标，将其填充满整个画布，再将其背景色改为透明。
* 将图片保存为 32-bit PNG 格式，同时在不降低画质的前提下进行优化处理。然后重命名为任何你希望的名字，比如 metro-tile.png。
* 可以在 IE 博文中添加 HTML meta 元素引用瓷贴及其颜色。

**Windows 8 固定网站的信息识别**

IE 10 可以通过轮询网站的 XML 文档识别主要信息，继而将其展示在首屏的应用图标上。这样即使用户没有打开应用，也可以接受更新的信息。该识别值可以是数字，也可以是预定义符号列表中的一个。

* [Tutorial on IEBlog with link to badge XML schema](http://blogs.msdn.com/b/ie/archive/2012/04/03/pinned-sites-in-windows-8.aspx)
* [Available badge values](http://msdn.microsoft.com/en-us/library/ie/br212849.aspx)

<meta name="msapplication-badge" value="frequency=NUMBER\_IN\_MINUTES;polling-uri=http://www.example.com/path/to/file.xml">

**在 IE 10 中禁用点击时链接的高亮效果**

这非常类似于 iOS Safari 中的 [-webkit-tap-highlight-color](http://davidwalsh.name/mobile-highlight-color)。但与这个 CSS 属性有所区别的是，这里使用的是一个 HTML 元标签，并且是一个布尔值而不是颜色值。使用这个功能就会对全体使用或禁用。

<meta name="msapplication-tap-highlight" content="no" />

更多帮助信息和技巧，请参考 [Microsoft's documentation on adapting WebKit-oriented apps for IE10](http://blogs.windows.com/windows_phone/b/wpdev/archive/2012/11/15/adapting-your-webkit-optimized-site-for-internet-explorer-10.aspx)。

**搜索**

**为搜索引擎蜘蛛设置 sitemap**

[了解如何创建 sitemap](http://www.sitemaps.org/protocol.html)。

<link rel="sitemap" type="application/xml" title="Sitemap" href="/sitemap.xml">

**对搜索引擎屏蔽页面**

根据 FLickr 前社区经理 Heather Champ 的看法，如果你足够理智，那么就不应该允许搜索引擎检索 “联系” 和 “投诉” 页面。

<meta name="robots" content="noindex">

**提醒**：不应该在页面中出现展示给搜索引擎的信息。

**Firefox 和 IE 搜索插件**

内置搜索功能的网站，可以考虑使用浏览器搜索插件提高性能。“搜索插件” 本质上就是一个 XML 文件，它定义了插件于浏览器的信息交互行为。[How to make a browser search plugin](https://www.google.com/search?ie=UTF-8&q=how+to+make+browser+search+plugin" \t "_blank)。

<link rel="search" title="" type="application/opensearchdescription+xml" href="">

**其他参数**

* 使用[polyfills](https://github.com/Modernizr/Modernizr/wiki/HTML5-Cross-browser-Polyfills)。
* 通过 [microdata](http://microformats.org/wiki/microdata) 使用 [Microformats](http://microformats.org/wiki/Main_Page" \t "_blank) 优化[搜索结果的准确性](http://googlewebmastercentral.blogspot.com/2009/05/introducing-rich-snippets.html" \t "_blank)。
* 如果你正在构建一个网页应用，那么你可能会考虑在 [iOS 5+ 中滚动时，通过-webkit-overflow-scrolling: touch 调用本地样式](http://www.johanbrook.com/articles/native-style-momentum-scrolling-to-arrive-in-ios-5/)。
* 如果想要屏蔽 Chrome 的翻译提示，或在网页中的禁用 Google 翻译，那么可以使用 [<meta name="google" value="notranslate">](https://support.google.com/translate/?hl=en#2641276)。如果只是在某一个部分禁用翻译，那么可以添加 [class="notranslate"](https://support.google.com/translate/?hl=en#2641276)。
* 如果想在 iOS 中，禁用 Safari 对手机号的自动检测和格式化功能，那么可以使用 [<meta name="format-detection" content="telephone=no">](https://developer.apple.com/library/safari/documentation/AppleApplications/Reference/SafariHTMLRef/Articles/MetaTags.html/#//apple_ref/doc/uid/TP40008193-SW5)。
* 可以通过使用 [implementing X-Robots-tag headers](https://github.com/h5bp/html5-boilerplate/issues/804)，避免开发阶段的网站被搜索引擎抓取。
* 当前的屏幕阅读器对 HTML5 的支持还不是很好，建议通过使用 [accessifyhtml5.js](https://github.com/yatil/accessifyhtml5.js) 给 HTML5 元素添加 ARIA roles 增强无障碍使用体验。

**订阅**

**RSS**

需要一个 RSS 订阅？请看这里的教程。[learn how to write an RSS feed from scratch](http://www.rssboard.org/rss-specification" \t "_blank)。

<link rel="alternate" type="application/rss+xml" title="RSS" href="/rss.xml">

**Atom**

Atom 和 RSS 非常类似，你可能会很中意它，[那么看看 Atom 的文档吧](http://www.atomenabled.org/developers/syndication/" \t "_blank)。

<link rel="alternate" type="application/atom+xml" title="Atom" href="/atom.xml">

**Pingbacks**

当其他网站链接到你的网站时，你的服务器可能会收到通知。该 href 属性需要包含你在 pingback 上的服务地址。

<link rel="pingback" href="">

**更多信息**

* [High-level explanation](https://codex.wordpress.org/Introduction_to_Blogging#Pingbacks)
* [Step-by-step example case](http://www.hixie.ch/specs/pingback/pingback-1.0#TOC5)
* [PHP pingback service](https://web.archive.org/web/20131211032834/http:/blog.perplexedlabs.com/2009/07/15/xmlrpc-pingbacks-using-php/)

**社交网络**

**Facebook 图谱**

当用户分享当前站点时，开发者可以设置其分享到 Facebook 或其他社交网络的信息内容。以下就是开发者需要的最基本信息。更多具体的内容类型，请参考 [Facebook's built-in Open Graph content templates](https://developers.facebook.com/docs/opengraph/objects/builtin/)。如果想要使用 Facebook 支持的高级特性，可以参考[Open Graph tutorial](https://developers.facebook.com/docs/opengraph/tutorial/" \t "_blank)。

<meta property="og:title" content="">

<meta property="og:description" content="">

<meta property="og:image" content="">

**Twitter 卡贴**

Twitter 卡贴提供了类似 Facebook 图谱的功能。实际上，当卡贴不能使用时，Twitter 也会使用类似图谱的功能。注意，对于这种方式，Twitter 要求开发者在每个基本域名上都激活卡贴功能。更多格式和应用处理方式，请参考 [official Twitter Cards documentation](https://dev.twitter.com/docs/cards)。

<meta name="twitter:card" content="summary">

<meta name="twitter:site" content="@site\_account">

<meta name="twitter:creator" content="@individual\_account">

<meta name="twitter:url" content="http://www.example.com/path/to/page.html">

<meta name="twitter:title" content="">

<meta name="twitter:description" content="">

<meta name="twitter:image" content="http://www.example.com/path/to/image.jpg">

**URL**

**Canonical URL**

通过在 URL 后面追加参数 # 或 ? 来显示页面状态，对浏览器或其他使用者都会有所帮助。http://www.example.com/cart.html?shopping-cart-open=true 就可以比 http://www.example.com/cart.html 更精确地定义页面。

<link rel="canonical" href="">

**Official shortlink**

向搜索引擎或该网站的使用者提示“这是本网站的短地址”，这种方式已经不被支持。更多信息请参考 [article about shortlinks onthe Microformats wiki](http://microformats.org/wiki/rel-shortlink)。

<link rel="shortlink" href="h5bp.com">

**独立的移动端地址**

如果在桌面端和移动端分别使用独立的 URLs，那么就要考虑好如何让搜索引擎算法更好地解析网站的配置信息。

在 HTML 页面中添加如下注释会对此有所帮助：

* 对于桌面端网页，添加 link rel="alternate" 标签指向相关的移动端地址，比如 <link rel="alternate" media="only screen and (max-width: 640px)" href="http://m.example.com/page.html" >。
* 对于移动端网页，添加 link rel="canonical" 标签指向相关的桌面端地址，比如 <link rel="canonical" href="http://www.example.com/page.html">。

**更多信息请参考：**

* <https://developers.google.com/webmasters/smartphone-sites/details#separateurls>
* <https://developers.google.com/webmasters/smartphone-sites/feature-phones>

**网页应用**

当网页应用在 iOS 中被添加到桌面后，可以使用如下标签获取信息：

使用 apple-mobile-web-app-capable，可以减少网页应用对 Chrome 的依赖，并提供 IOS App 的视图支持。可以通过使用 apple-mobile-web-app-status-bar-style，控制默认视图的色彩模式。

<meta name="apple-mobile-web-app-capable" content="yes">

<meta name="apple-mobile-web-app-status-bar-style" content="black">

使用 apple-mobile-web-app-title，可以为桌面图标重命名。该功能支持 iOS 6+。

<meta name="apple-mobile-web-app-title" content="">

更多信息请参考[苹果官方文档](https://developer.apple.com/library/safari/" \l "documentation/AppleApplications/Reference/SafariHTMLRef/Articles/MetaTags.html" \t "_blank)。

**Apple Touch Icons**

Apple Touch Icons 相当于 iOS 设备的桌面图标。Apple Touch Icons 的主要尺寸如下：

* 57×57px – iPhone with @1x display and iPod Touch
* 72×72px – iPad and iPad mini with @1x display running iOS ≤ 6
* 76×76px – iPad and iPad mini with @1x display running iOS ≥ 7
* 114×114px – iPhone with @2x display running iOS ≤ 6
* 120×120px – iPhone with @2x and @3x display running iOS ≥ 7
* 144×144px – iPad and iPad mini with @2x display running iOS ≤ 6
* 152×152px – iPad and iPad mini with @2x display running iOS 7
* 180×180px – iPad and iPad mini with @2x display running iOS 8

显示尺寸含义：

* @1x - non-Retina
* @2x - Retina
* @3x - Retina HD

更多有关 iOS 设备的显示信息，请参考 [List of iOS devices display](https://en.wikipedia.org/wiki/List_of_iOS_devices#Display)。

大多数情况下，可以使用 180×180px 大小的图标，命名为 apple-touch-icon.png 后包含在页面的 <head> 即可：

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

如果希望每个设备有不同的内容，那么可以添加多个 Apple Touch Icons。更多信息情参考 [article on Touch Icons](https://mathiasbynens.be/notes/touch-icons)。

**Apple Touch Startup Image**

除此之外，可以给 iOS 上的网页应用添加启动界面。该功能需要使用 apple-touch-startup-image，并附加相关的图片链接。由于 iOS 应用于多种尺寸的屏幕下，所以有必要使用媒体查检测尺寸，然后再加载图片。这里是一个在 retina iPhone 的示例：

<link rel="apple-touch-startup-image" media="(max-device-width: 480px) and (-webkit-min-device-pixel-ratio: 2)" href="img/startup-retina.png">

不过，该功能可能需要使用 JavaScript 检测启动界面的图片。Mobile Boilerplate 提供了一个有效地功能来解决这个问题，请参考[helpers.js](https://github.com/h5bp/mobile-boilerplate/blob/v4.1.0/js/helper.js" \l "L336-L383" \t "_blank) 的实现方法。

**Chrome Mobile web apps**

Chrome 移动端拥有一个专有的元标签，[用来在桌面安装网页应用](https://developer.chrome.com/multidevice/android/installtohomescreen" \t "_blank)，该标签比苹果的属性属性标签更具有通用性。

<meta name="mobile-web-app-capable" content="yes">

同样适用于 touch icons:

<link rel="icon" sizes="192x192" href="highres-icon.png">

**其他**

**.gitignore**

HTML5 Boilerplate 引入了一个基础性的、项目级的 .gitignore。主要用来让源代码忽略对项目中特定文件和目录的管理。在不同的开发环境使用不同的忽略列表，将会大有裨益。

特定系统和特定编辑器的文件应该使用 “global ignore” ，从而忽略系统中所有库对相关文件的依赖。

比如，在希望全局忽略的 HOME 目录，将如下内容放入 .gitignore 文件。

[core]

excludesfile = ~/.gitignore

更多信息请参考：

* [More on global ignores: https://help.github.com/articles/ignoring-files](More on global ignores: https://help.github.com/articles/ignoring-files)
* [Comprehensive set of ignores on GitHub: https://github.com/github/gitignore](Comprehensive set of ignores on GitHub: https://github.com/github/gitignore)

**.editorconfig**

.editorconfig 文件用来激励和帮助开发者/开发团队，在不同的编辑器和 IDE 下，定义和维护一致性的代码风格。

默认情况下，.editorconfig 包含了一些基本[属性](http://editorconfig.org/" \l "supported-properties" \t "_blank)，用来体现代码风格，但是开发者可以根据需求自定义相关属性。

为了让编辑器、IDE 更好的使用 .editorconfig 配置文件，开发者需要[安装一个插件](http://editorconfig.org/" \l "download" \t "_blank)。

**注意**：如果你不需要使用 HTML5 Boilerplate 提供的服务器配置，我们强烈建议不要允许你服务器使用 .editorconfig 文件，因为该文件会屏蔽敏感信息。

更多信息请参考 [EditorConfig project](http://editorconfig.org/" \t "_blank)。

**服务器配置**

HTML5 Boilerplate 为 Apache HTTP 服务器配置了 .htaccess 文件。如果不是使用 Apache 服务器，建议下载 [server configuration](https://github.com/h5bp/server-configs) 并适配所用服务器。

**Servers and Stacks**

介绍 WEB 服务器和堆栈的内容就远远超过了本文档的范围，但这里有一些常用的资料：

* [Apache HTTP Server](https://httpd.apache.org/docs/trunk/getting-started.html)
* [LAMP](https://en.wikipedia.org/wiki/LAMP_%28software_bundle%29) (Linux, Apache, MySQL, and PHP). Other variants include [MAMP](https://www.mamp.info/en/index.html), [WAMP](http://www.wampserver.com/en/), or [XAMPP](https://www.apachefriends.org/index.html).
* LAPP uses PostgreSQL instead of MySQL
* [Nginx](http://wiki.nginx.org/GettingStarted)
* [LEMP](http://www.chrisjohnston.org/ubuntu-tutorials/setting-up-a-lemp-stack-ubuntu-904) is similar to the LAMP stack but uses Nginx
* [IIS](https://en.wikipedia.org/wiki/Internet_Information_Services)
* [ASP.NET](http://www.asp.net/get-started)
* [MEAN](http://mean.io/) (MongoDB, Express, AngularJS, Node.js)

**.htaccess**

.htaccess（超文本存取）文件就是[Apache HTTP 服务器的配置文件](https://github.com/h5bp/server-configs-apache" \t "_blank)。常用于：

* Rewriting URLs
* Controlling cache
* Authentication
* Server-side includes
* Redirects
* Gzipping

如果你使用过主流服务器的配置文件（通常称为 httpd.conf），那么你应该熟悉往 .htaccess 文件中添加逻辑处理。比如， 部分位于主要的配置文件中。因为 .htaccess 文件会拖慢 Apache，所以通常建议这么做。

为了启用本地的 Apache 模块，请参考[这里](https://github.com/h5bp/server-configs-apache/wiki/How-to-enable-Apache-modules" \t "_blank)。

.htaccess 主要用于：

* 允许跨源请求 web 字体
* 当浏览器请求图片时，使用跨域资源共享头
* 将 404.html 作为 404 错误文档
* 为 IE 用户提供更好的用户体验
* 将 UTF-8 作为 text/html 和 text/plain 的字符编码
* 启用 URLs 重写引擎
* 强制或移除 URL 开头的 www.
* 缺少默认文档时阻塞对目录的调用
* 隔离文件访问，防止敏感信息泄露
* 降低了 MIME 类型的安全风险
* 强制压缩
* 通知浏览器是否需要从服务器请求特定文件，或者是否需要从浏览器缓存获取特定文件

当使用 .htaccess 时，我们建议阅读一次所有的内部注释。其中有一些是可选的。

更多有关 .htaccess 文件的信息请参考[这里](https://httpd.apache.org/docs/current/howto/htaccess.html" \t "_blank)。

注意，.htaccess 的源库在[这里](https://github.com/h5bp/server-configs-apache" \t "_blank)。

**crossdomain.xml**

跨域策略文件是一个 XML 文档，其内容来自 web 客户端（比如，Adobe Flash Player, Adobe Reader），可以允许处理来自多个域名的数据：

* 授权读取数据
* 允许客户端在跨域请求中导入自定义的头部信息
* 授权许可基本的套接字连接

**注意**：如果一个客户端从一个特定的源域名获取内容，然后该内容请求重定向其他域名，那么远程域名就需要使用跨域策略文件，从而获得源域名的授权，最终允许客户端继续处理相关事务。

更多信息请参考 [cross-domain policy file specification](https://www.adobe.com/devnet/articles/crossdomain_policy_file_spec.html)。

**robots.txt**

robots.txt 文件用来告知搜索引擎网站中可以抓取的页面。

默认情况下，文件中包含下面两行信息：

* User-agent: \* - 以下规则适用于所有的搜索引擎
* Disallow: - 网站中的所有页面都可以被抓取

如果想屏蔽某些页面，那么你需要在 Disallow 参数后面做出具体声明（比如: Disallow: /path）。如果你想屏蔽所有内容，只需 Disallow: /。

/robots.txt 并不是用来访问控制的，所以请不要这样使用。可以将其视为一个 “禁止通行” 标志，而不是一扇锁上的门。通过 robots.txt 文件屏蔽 URLs，即使未被抓取仍有可能被定位，而且 robots.txt 文件中的内容也可以被任何人访问到，这样就间接透漏了私有内容的位置。所以，如果想屏蔽访问私人信息，建议使用合理的验证机制。

关于 /robots.txt 文件的更多信息请参考：

* [robotstxt.org](http://www.robotstxt.org/)
* [How Google handles the robots.txt file](https://developers.google.com/webmasters/control-crawl-index/docs/robots_txt)

**browserconfig.xml**

用户在 Windows 8.1 启动界面固定的应用图标，可以通过 browserconfig.xml 文件进行个性化定制。在该文件中，可以自定义瓷贴颜色、图片，甚至是[动态瓷贴](http://msdn.microsoft.com/en-us/library/ie/dn455106.aspx" \l "CreatingLiveTiles" \t "_blank)。

默认情况下，该文件指向两个既有的瓷贴图片：

* tile.png (558x558px): used for Small, Medium and Large tiles.如有必要该图片可以自动修改尺寸。
* tile-wide.png (558x270px): user for Wide tiles.

注意，当收藏网站时，IE 11 将收藏夹中使用相同的图片。

有关 browserconfig.xml 文件的跟多信息，请参考 [MSDN](http://msdn.microsoft.com/en-us/library/ie/dn320426%28v=vs.85%29.aspx)。

**FAQ**

**为什么 jQuery 的链接没有加http？**

这是因为使用了 [protocol-relative URLs](http://paulirish.com/2010/the-protocol-relative-url/)。

**注意**:如果你尝试直接在浏览器中预览本地网页，那么浏览器将无法加载资源，特别是使用 protocol-relative URLs，因为它会尝试从本地文件系统获取相关资源。建议使用本地的 HTTP 服务器，或者是允许在线预览网页的文件托管服务（比如[Dropbox](https://www.dropbox.com/" \t "_blank)）来测试网页。

建立本地 HTTP 服务器可以使用多种[短命令](https://gist.github.com/willurd/5720255" \t "_blank)：

* PHP 5.4.0+ 使用 [php -S localhost:8080](https://php.net/manual/en/features.commandline.webserver.php" \t "_blank) 命令从本地目录启动并运行。
* Python 2.x 使用 python -m SimpleHTTPServer 命令从本地目录启动并运行。
* Python 3.x 使用 python -m http.server 命令从本地目录启动并运行。
* Ruby 1.9.2+ 使用 ruby -run -ehttpd . -p8080 命令从本地目录启动并运行。
* Node.js 使用 [static -p 8080](https://www.npmjs.org/package/node-static) 或 [http-server -p 8080](https://www.npmjs.org/package/http-server) 命令安装和启动服务器。

**为什么不从 Google CDN 自动加载最新版本的 jQuery？**

首先，Google [CDN](https://en.wikipedia.org/wiki/Content_delivery_network) 所指向的 [jQuery 文件](https://ajax.googleapis.com/ajax/libs/jquery/1/jquery.js)[已不再更新，并且将锁定在1.11.1版本，以防止新版本变动引发的网页脚本失效](http://blog.jquery.com/2014/07/03/dont-use-jquery-latest-js/)。

其次，通常来说，更新版本应该是个慎重的决定！网页中不应该一直引用最新版本的原因如下：

* 可能与现有插件、代码不兼容
* 与固定引用某一版本的方式相比，一直引用最新版本将会减少本地缓存时间，这意味着用户无法受益于长周期缓存带来的便利。

**为什么将 Google Analytics 代码置于网页底部？而 Google 则建议将其置于<head>。**

将其置于<head>的主要优势是，在页面加载完成之前，即使用户离开当前网页，也可以获得页面浏览量（PV）。然而，将其置于页面底部则[有助于改善整体性能](http://stevesouders.com/efws/inline-scripts-bottom.php" \t "_blank)。

**如何在 HTML5 Boilerplate 中整合**[**Bootstrap**](http://getbootstrap.com/)**。**

一个简单的方法是使用 [Initializr](http://initializr.com/" \t "_blank)，并创建自定义的构建环境——在其中同时包含 HTML5 Boilerplate 和 [Bootstrap](http://getbootstrap.com/)。

更多信息请参考文章： [HTML5 Boilerplate 和 Bootstrap 的互相整合](https://www.quora.com/Is-Bootstrap-a-complement-or-an-alternative-to-HTML5-Boilerplate-or-viceversa/answer/Nicolas-Gallagher)。

**每当 HTML5 Boilerplate 释放出新版本，开发者是否必须更新网站？**

完全没有必要！这就像是房子盖完之后，虽然需要时不时的修护一下，但通常没必要重新打地基。当然，如果想尝试新版本的变动，也是完全可以的，但最好正确权衡这么做的成本和收获。

**哪里可以获取技术支持？**

请使用 [StackOverflow](https://stackoverflow.com/questions/tagged/html5boilerplate" \t "_blank) 寻求帮助。

著作权归作者所有。  
商业转载请联系作者获得授权,非商业转载请注明出处。  
原文: <http://www.w3cplus.com/html5/html5-boilerplate.html> © [w3cplus.com](http://www.w3cplus.com/)