《[javascript](http://caibaojian.com/t/javascript)图片预加载详解》图片的加载速度往往影响着网站整体的用户体验，尤其对于包含大量图片的网站。对图片进行预加载，不失为一个高效的解决方案。如何实现预加载？本文将例举利用[CSS](http://caibaojian.com/css3/)、JavaScript及[ajax](http://caibaojian.com/t/ajax)实现图片预加载的三大方法。

**Perishable Press网站近日发表了一篇文章《**[**3 Ways to Preload Images with CSS, JavaScript, or Ajax**](http://perishablepress.com/3-ways-preload-images-css-javascript-ajax/)**》，分享了利用CSS、JavaScript及Ajax实现图片预加载的三大方法。下面为译文。**

预加载图片是提高用户体验的一个很好方法。图片预先加载到浏览器中，访问者便可顺利地在你的网站上冲浪，并享受到极快的加载速度。这对图片画廊及图片占据很大比例的网站来说十分有利，它保证了图片快速、无缝地发布，也可帮助用户在浏览你网站内容时获得更好的用户体验。本文将分享三个不同的预加载技术，来增强网站的性能与可用性。

**方法一：用CSS和JavaScript实现预加载**

实现预加载图片有很多方法，包括使用CSS、JavaScript及两者的各种组合。这些技术可根据不同设计场景设计出相应的解决方案，十分高效。

单纯使用CSS，可容易、高效地预加载图片，[代码](http://caibaojian.com/c/code)如下：

#preload-01 { background: url(http://domain.tld/image-01.png) no-repeat -9999px -9999px; }

#preload-02 { background: url(http://domain.tld/image-02.png) no-repeat -9999px -9999px; }

#preload-03 { background: url(http://domain.tld/image-03.png) no-repeat -9999px -9999px; }

将这三个ID选择器应用到(X)[html](http://caibaojian.com/t/html)元素中，我们便可通过CSS的background属性将图片预加载到屏幕外的背景上。只要这些图片的路径保持不变，当它们在[web](http://caibaojian.com/c/web)页面的其他地方被调用时，浏览器就会在渲染过程中使用预加载（缓存）的图片。简单、高效，不需要任何JavaScript。

该方法虽然高效，但仍有改进余地。使用该法加载的图片会同页面的其他内容一起加载，增加了页面的整体加载时间。为了解决这个问题，我们增加了一些JavaScript代码，来推迟预加载的时间，直到页面加载完毕。代码如下：

// better image preloading @ <http://perishablepress.com/press/2009/12/28/3-ways-preload-images-css-javascript-ajax/>

function preloader() {

if (document.getElementById) {

document.getElementById("preload-01").style.background = "url(http://domain.tld/image-01.png) no-repeat -9999px -9999px";

document.getElementById("preload-02").style.background = "url(http://domain.tld/image-02.png) no-repeat -9999px -9999px";

document.getElementById("preload-03").style.background = "url(http://domain.tld/image-03.png) no-repeat -9999px -9999px";

}

}

function addLoadEvent(func) {

var oldonload = window.onload;

if (typeof window.onload != 'function') {

window.onload = func;

} else {

window.onload = function() {

if (oldonload) {

oldonload();

}

func();

}

}

}

addLoadEvent(preloader);

在该脚本的第一部分，我们获取使用类选择器的元素，并为其设置了background属性，以预加载不同的图片。

该脚本的第二部分，我们使用addLoadEvent()函数来延迟preloader()函数的加载时间，直到页面加载完毕。

如果JavaScript无法在用户的浏览器中正常运行，会发生什么？很简单，图片不会被预加载，当页面调用图片时，正常显示即可。

**方法二：仅使用JavaScript实现预加载**

上述方法有时确实很高效，但我们逐渐发现它在实际实现过程中会耗费太多时间。相反，我更喜欢使用纯JavaScript来实现图片的预加载。下面将提供两种这样的预加载方法，它们可以很漂亮地工作于所有现代浏览器之上。

**JavaScript代码段1**

只需简单编辑、加载所需要图片的路径与名称即可，很容易实现：

<div class="hidden">

<script type="text/javascript">

<!--//--><![CDATA[//><!--

var images = new Array()

function preload() {

for (i = 0; i < preload.arguments.length; i++) {

images[i] = new Image()

images[i].src = preload.arguments[i]

}

}

preload(

"http://domain.tld/gallery/image-001.jpg",

"http://domain.tld/gallery/image-002.jpg",

"http://domain.tld/gallery/image-003.jpg"

)

//--><!]]>

</script>

</div>

该方法尤其适用预加载大量的图片。我的画廊网站使用该技术，预加载图片数量达50多张。将该脚本应用到登录页面，只要用户输入登录帐号，大部分画廊图片将被预加载。

**JavaScript代码段2**

该方法与上面的方法类似，也可以预加载任意数量的图片。将下面的脚本添加入任何Web页中，根据程序指令进行编辑即可。

<div class="hidden">

<script type="text/javascript">

<!--//--><![CDATA[//><!--

if (document.images) {

img1 = new Image();

img2 = new Image();

img3 = new Image();

img1.src = "http://domain.tld/path/to/image-001.gif";

img2.src = "http://domain.tld/path/to/image-002.gif";

img3.src = "http://domain.tld/path/to/image-003.gif";

}

//--><!]]>

</script>

</div>

正如所看见，每加载一个图片都需要创建一个变量，如“img1 = new Image();”，及图片源地址声明，如“img3.src = "../path/to/image-003.gif";”。参考该模式，你可根据需要加载任意多的图片。

我们又对该方法进行了改进。将该脚本封装入一个函数中，并使用 addLoadEvent（），延迟预加载时间，直到页面加载完毕。

function preloader() {

if (document.images) {

var img1 = new Image();

var img2 = new Image();

var img3 = new Image();

img1.src = "http://domain.tld/path/to/image-001.gif";

img2.src = "http://domain.tld/path/to/image-002.gif";

img3.src = "http://domain.tld/path/to/image-003.gif";

}

}

function addLoadEvent(func) {

var oldonload = window.onload;

if (typeof window.onload != 'function') {

window.onload = func;

} else {

window.onload = function() {

if (oldonload) {

oldonload();

}

func();

}

}

}

addLoadEvent(preloader);

**方法三：使用Ajax实现预加载**

上面所给出的方法似乎不够酷，那现在来看一个使用Ajax实现图片预加载的方法。该方法利用DOM，不仅仅预加载图片，还会预加载CSS、JavaScript等相关的东西。使用Ajax，比直接使用JavaScript，优越之处在于JavaScript和CSS的加载不会影响到当前页面。该方法简洁、高效。

window.onload = function() {

setTimeout(function() {

// XHR to request a JS and a CSS

var xhr = new XMLHttpRequest();

xhr.open('GET', 'http://domain.tld/preload.js');

xhr.send('');

xhr = new XMLHttpRequest();

xhr.open('GET', 'http://domain.tld/preload.css');

xhr.send('');

// preload image

new Image().src = "http://domain.tld/preload.png";

}, 1000);

};

上面代码预加载了“preload.[JS](http://caibaojian.com/t/javascript)”、“preload.css”和“preload.png”。1000毫秒的超时是为了防止脚本挂起，而导致正常页面出现功能问题。

下面，我们看看如何用JavaScript来实现该加载过程：

window.onload = function() {

setTimeout(function() {

// reference to <head>

var head = document.getElementsByTagName('head')[0];

// a new CSS

var css = document.createElement('link');

css.type = "text/css";

css.rel = "stylesheet";

css.href = "http://domain.tld/preload.css";

// a new JS

var js = document.createElement("script");

js.type = "text/javascript";

js.src = "http://domain.tld/preload.js";

// preload JS and CSS

head.appendChild(css);

head.appendChild(js);

// preload image

new Image().src = "http://domain.tld/preload.png";

}, 1000);

};

这里，我们通过DOM创建三个元素来实现三个文件的预加载。正如上面提到的那样，使用Ajax，加载文件不会应用到加载页面上。从这点上看，Ajax方法优越于JavaScript。