How well do you know the web?

Script Loading

刘丽慷

加载脚本

```
<script src="//other-domain.com/1.js"></script>
<script src="2.js"></script>
```

阻塞渲染

- 为什么会阻塞渲染
- DOM API 允许向正在处理的内容追加字符串
- document.write

怎么办

- 将脚本元素放置到文档底部

IE 的解决方案

```
<script src="//other-domain.com/1.js" defer></script>
<script src="2.js" defer></script>
```

defer

- Internet Explorer 4 中引入了 “defer”
- 我保证不使用像 document.write 这样的功能来注入内容。要是我说谎，任你处罚
- 在 DOMContentLoaded 触发前执行它们
- 按顺序

一团糟

- src / defer
- html 标签
- 动态添加
- 6 种方式添加一个脚本
- 浏览器们对于它们的执行顺序并没有达成一致

WHATWG

- 动态添加时 defer 不生效
- 缺少 src 属性时 defer 不生效
- 否则，script 在文档解析完毕后，按添加顺序执行

bug

```
// demo.html
<script src="//other-domain.com/1.js" defer></script>
<script src="2.js" defer></script>


// 1.js
console.log('1');
document.getElementsByTagName('p')[0].innerHTML = 'Changing some content';
console.log('2');


// 2.js
console.log('3');
```

HTML5 async

```
<script src="//other-domain.com/1.js" async></script>
<script src="2.js" async></script>
```

完美的场景

- 多个脚本能立即下载
- 不会阻塞渲染
- 下载完毕 按照添加的顺序执行

HTML 不允许你这么做

- RequireJS 包裹在一个回调函数中
- XHR eval() 不适用于跨域脚本
- LabJS hack <script type="script/cache" src="...">

DOM 伸出援手

```
[
  '//other-domain.com/1.js',
  '2.js'
].forEach(function(src) {
  var script = document.createElement('script');
  script.src = src;
  document.head.appendChild(script);
});
```

改进

```
[
  '//other-domain.com/1.js',
  '2.js'
].forEach(function(src) {
  var script = document.createElement('script');
  script.src = src;
  script.async = false;
  document.head.appendChild(script);
});
```

预加载

```
<link rel="prefetch" href="//other-domain.com/1.js">
<link rel="prefetch" href="2.js">
```

HTTP2/SPDY

```
<script src="dependencies.js"></script>
<script src="enhancement-1.js"></script>
<script src="enhancement-2.js"></script>
<script src="enhancement-3.js"></script>
...
<script src="enhancement-10.js"></script>
```

IE 特性

```
var script = document.createElement('script');
script.src = 'whatever.js';
```

readystatechange

```
var script = document.createElement('script');

script.onreadystatechange = function() {
  if (script.readyState == 'loaded') {
    // 脚本下载完毕，但还没有执行。
    // 在做如下操作前它不会执行：
    document.body.appendChild(script);
  }
};

script.src = 'whatever.js';
```

综合的方案

```
var scripts = [
  '1.js',
  '2.js'
];
var src;
var script;
var pendingScripts = [];
var firstScript = document.scripts[0];


// 监视 IE 中的脚本加载
function stateChange() {
  // 尽可能多的按顺序执行脚本
  var pendingScript;
  while (pendingScripts[0] && pendingScripts[0].readyState == 'loaded') {
    pendingScript = pendingScripts.shift();
    // 避免该脚本的加载事件再次触发(比如修改了 src 属性)
    pendingScript.onreadystatechange = null;
    // 不能使用 appendChild，在低版本 IE 中如果元素没有闭合会有 bug
    firstScript.parentNode.insertBefore(pendingScript, firstScript);
  }
}
```

接上一页

```
// 循环脚本地址
while (src = scripts.shift()) {
  if ('async' in firstScript) { // 现代浏览器
    script = document.createElement('script');
    script.async = false;
    script.src = src;
    document.head.appendChild(script);
  }
  else if (firstScript.readyState) { // IE<10
    // 创建一个脚本并添加进待执行队列中
    script = document.createElement('script');
    pendingScripts.push(script);
    // 监听状态改变
    script.onreadystatechange = stateChange;
    // 必须在添加 onreadystatechange 监听后设置 src
    // 否则会错过缓存脚本的加载事件
    script.src = src;
  }
  else { // 退化使用延迟加载
    document.write('<script src="' + src + '" defer></'+'script>');
  }
}
```

参考

- A deep dive into script loading
- ES6 modules in depth
- ES6 modules – the browser-specific stuff
- Using <link rel=preload>
- The order of caches
- Chrome bug: Compositing within SVG
- Anatomy of a frame
- The browser's event loop: Tasks vs microtasks
- visibility: visible undoes visibility: hidden
- The code behind Big Web Quiz
- How the Big Web Quiz music was scheduled

Thanks