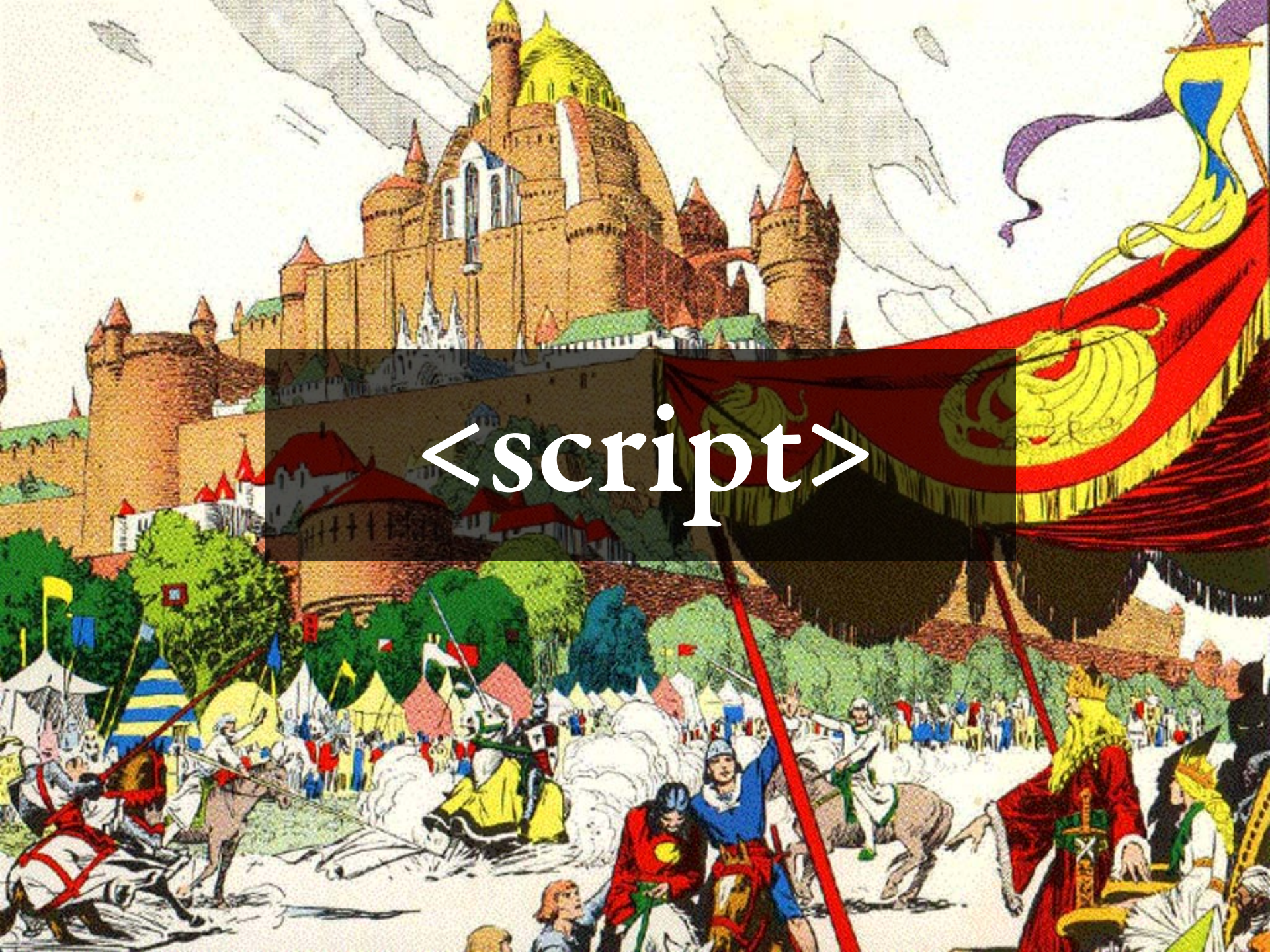
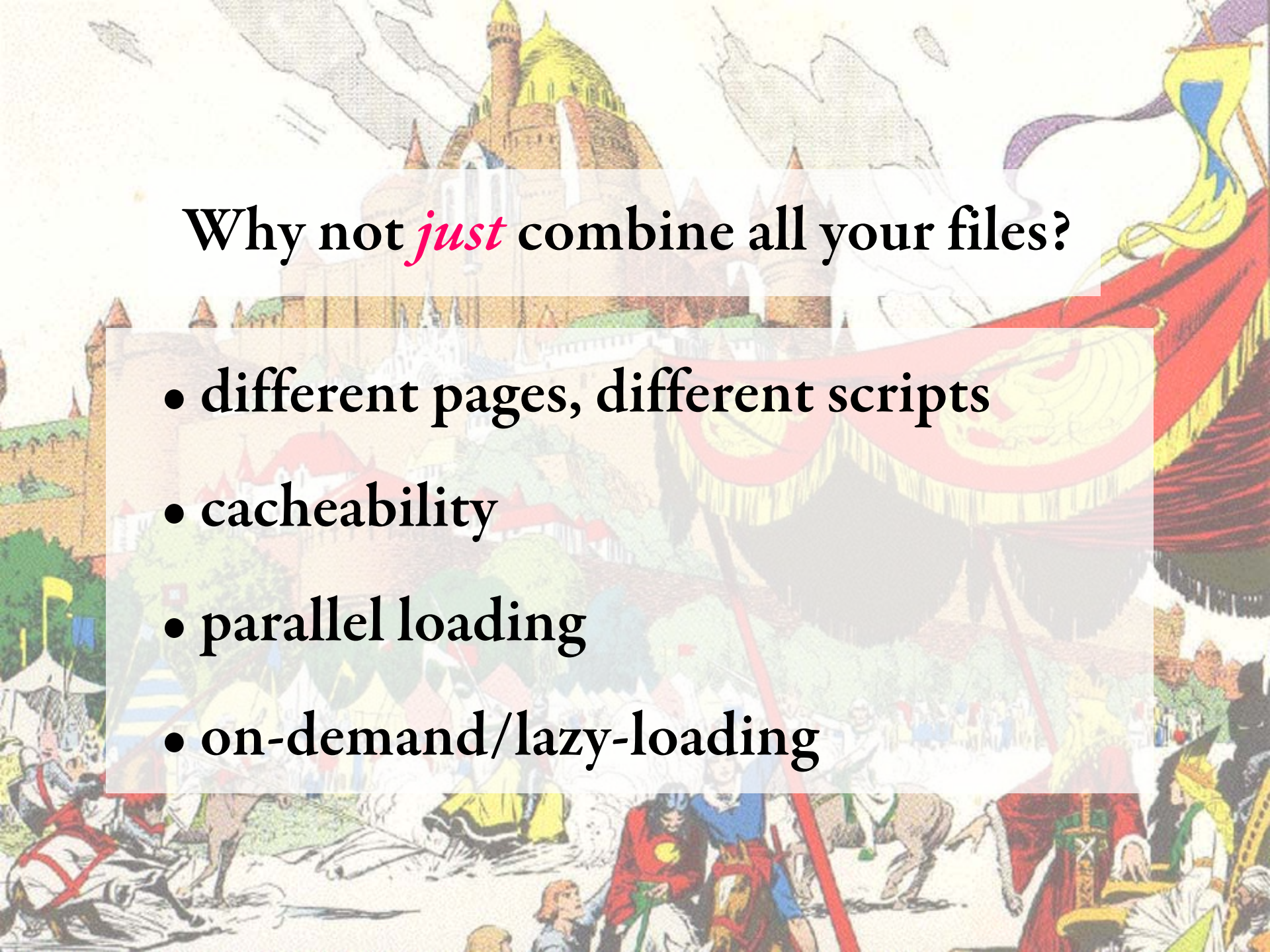


[illegible]

Kyle Simpson
@getify
<http://getify.me>

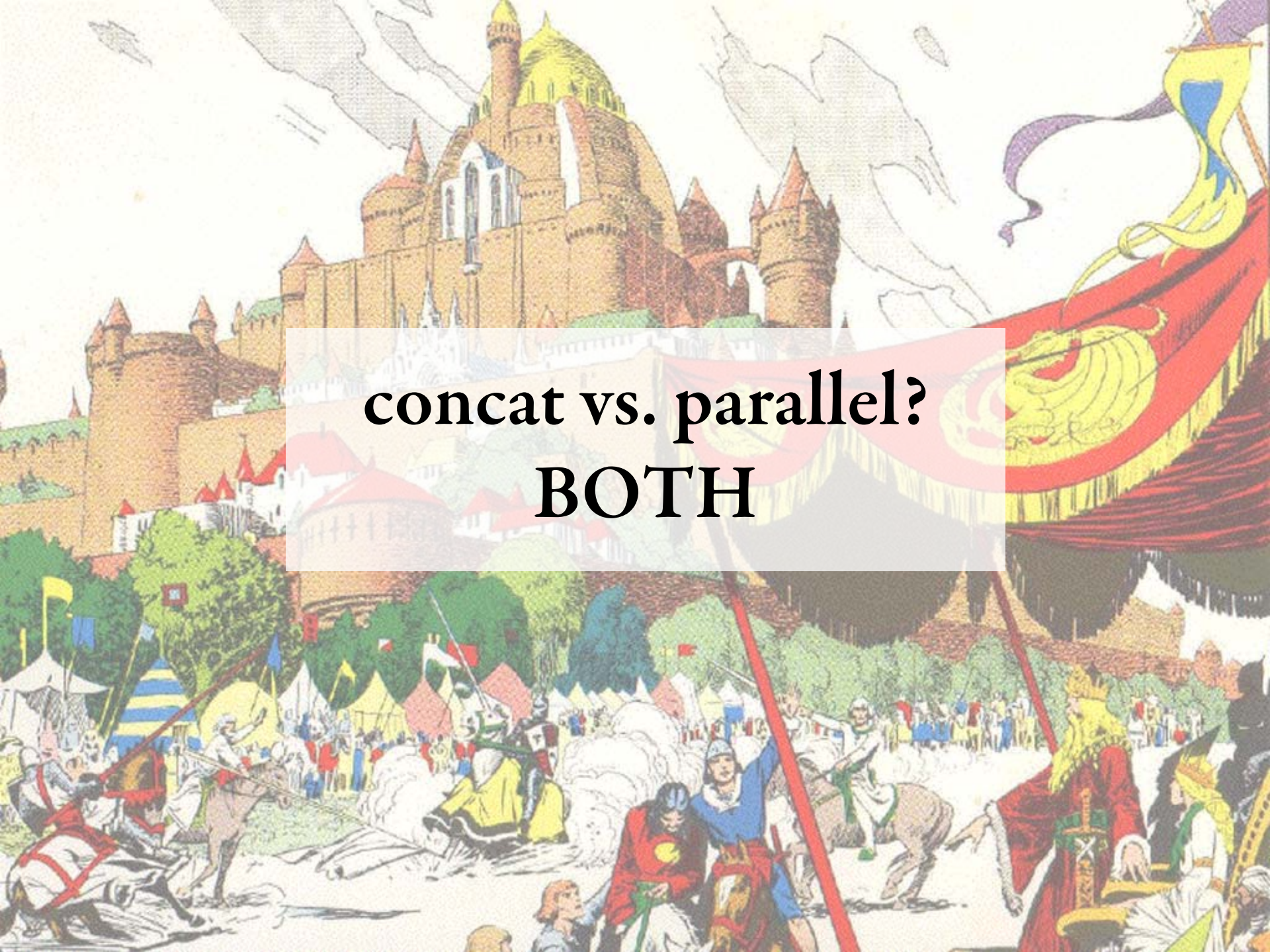


<script>



Why not *just* combine all your files?

- different pages, different scripts
- cacheability
- parallel loading
- on-demand/lazy-loading



concat vs. parallel?
BOTH

`document.write()`

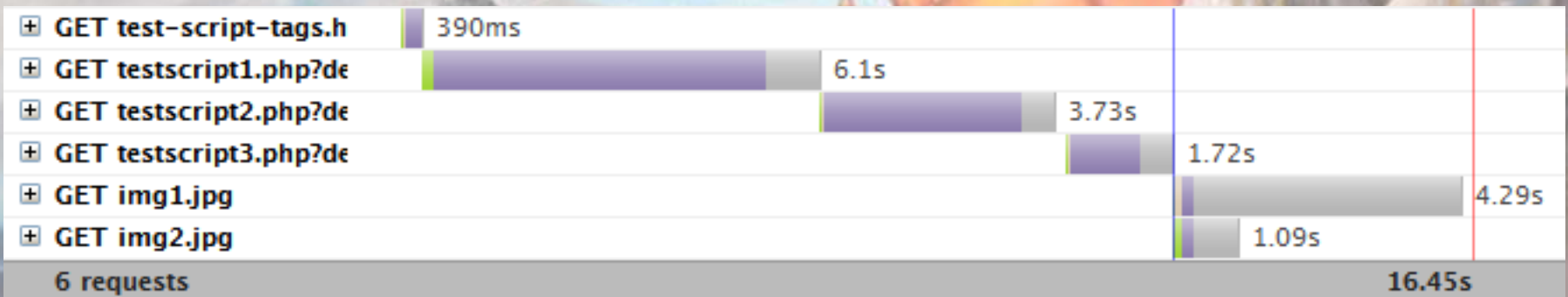


A historical painting depicting a public execution scene. In the foreground, a man in a white robe is being led towards a guillotine. In the background, a man in a crown and armor is visible. The scene is set outdoors with a crowd of people in the background.

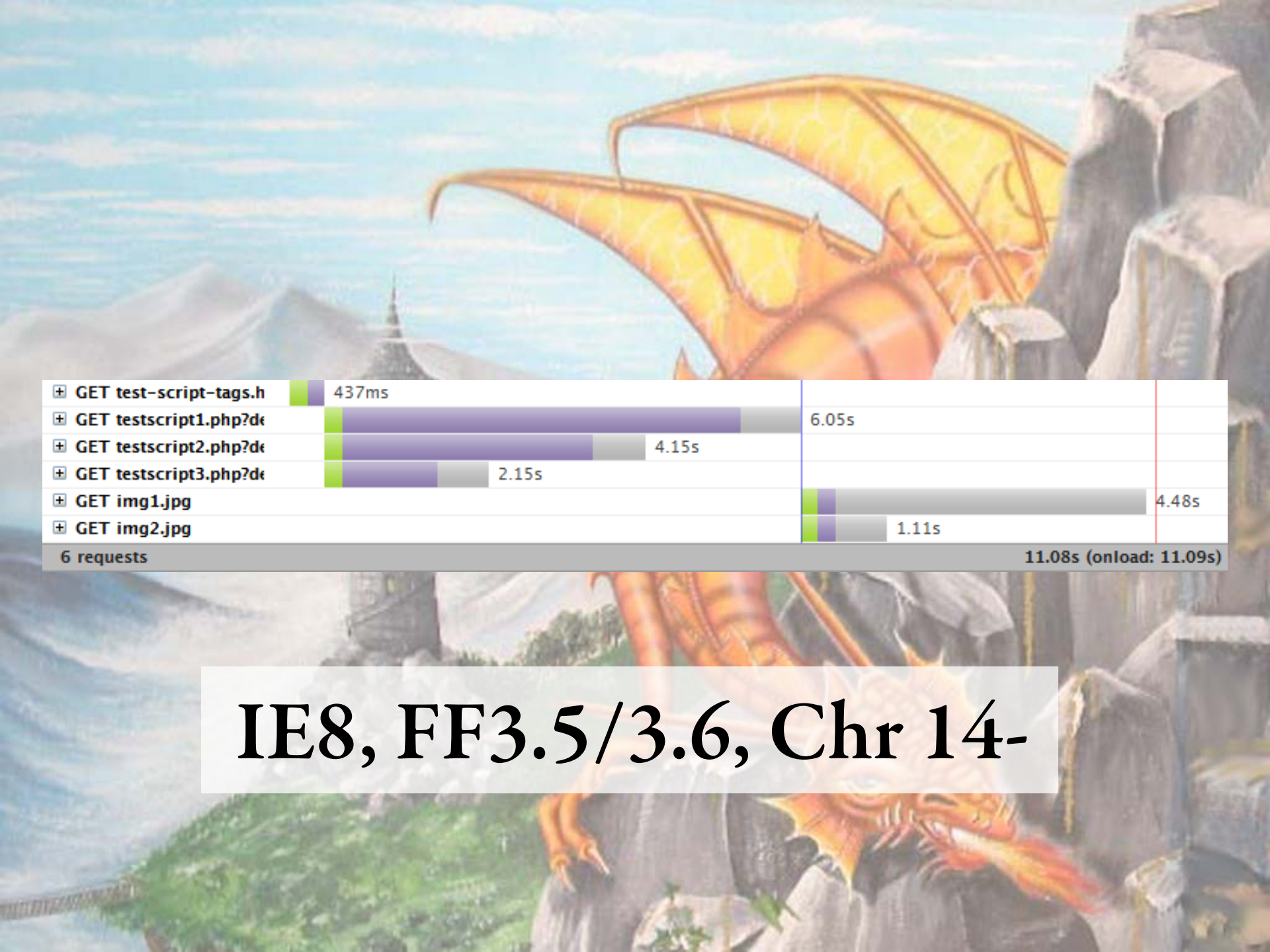
**document.write()
Must
Die!**



Performance

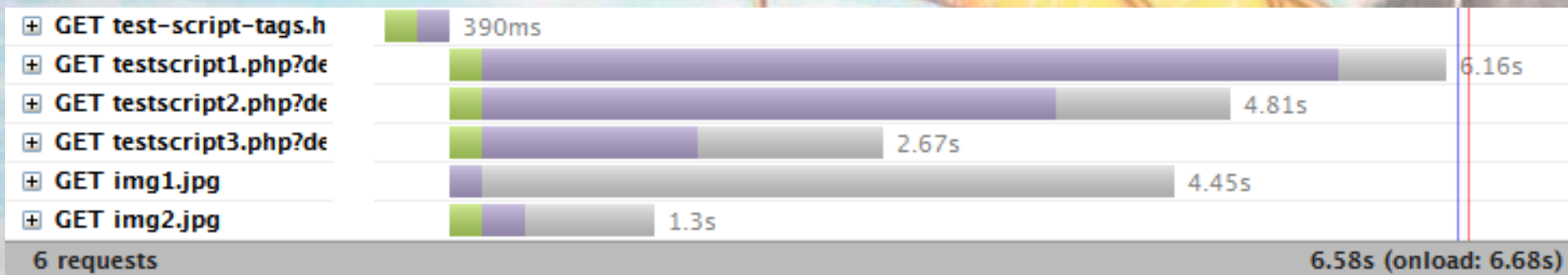


IE7-, FF3-, Opera

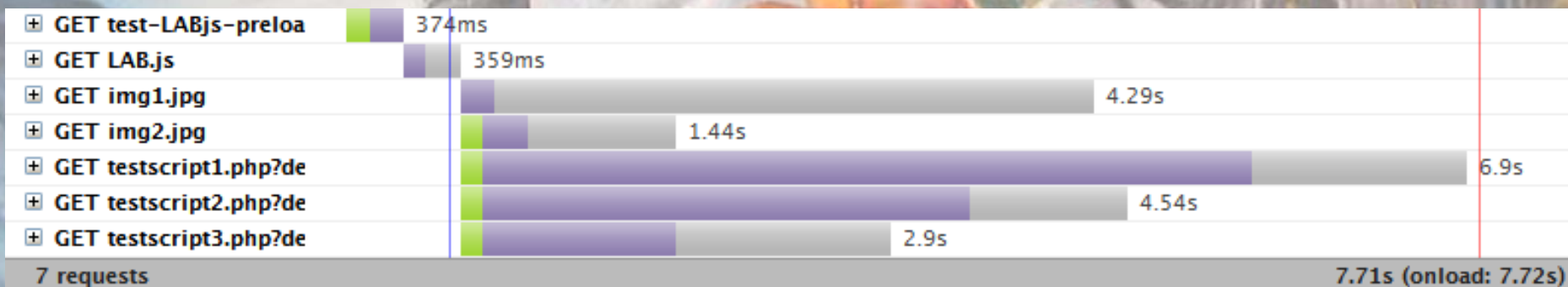


| | | |
|--------------------------|-------|-------------------------|
| ⊕ GET test-script-tags.h | 437ms | |
| ⊕ GET testscript1.php?de | | 6.05s |
| ⊕ GET testscript2.php?de | 4.15s | |
| ⊕ GET testscript3.php?de | 2.15s | |
| ⊕ GET img1.jpg | | 4.48s |
| ⊕ GET img2.jpg | 1.11s | |
| 6 requests | | 11.08s (onload: 11.09s) |

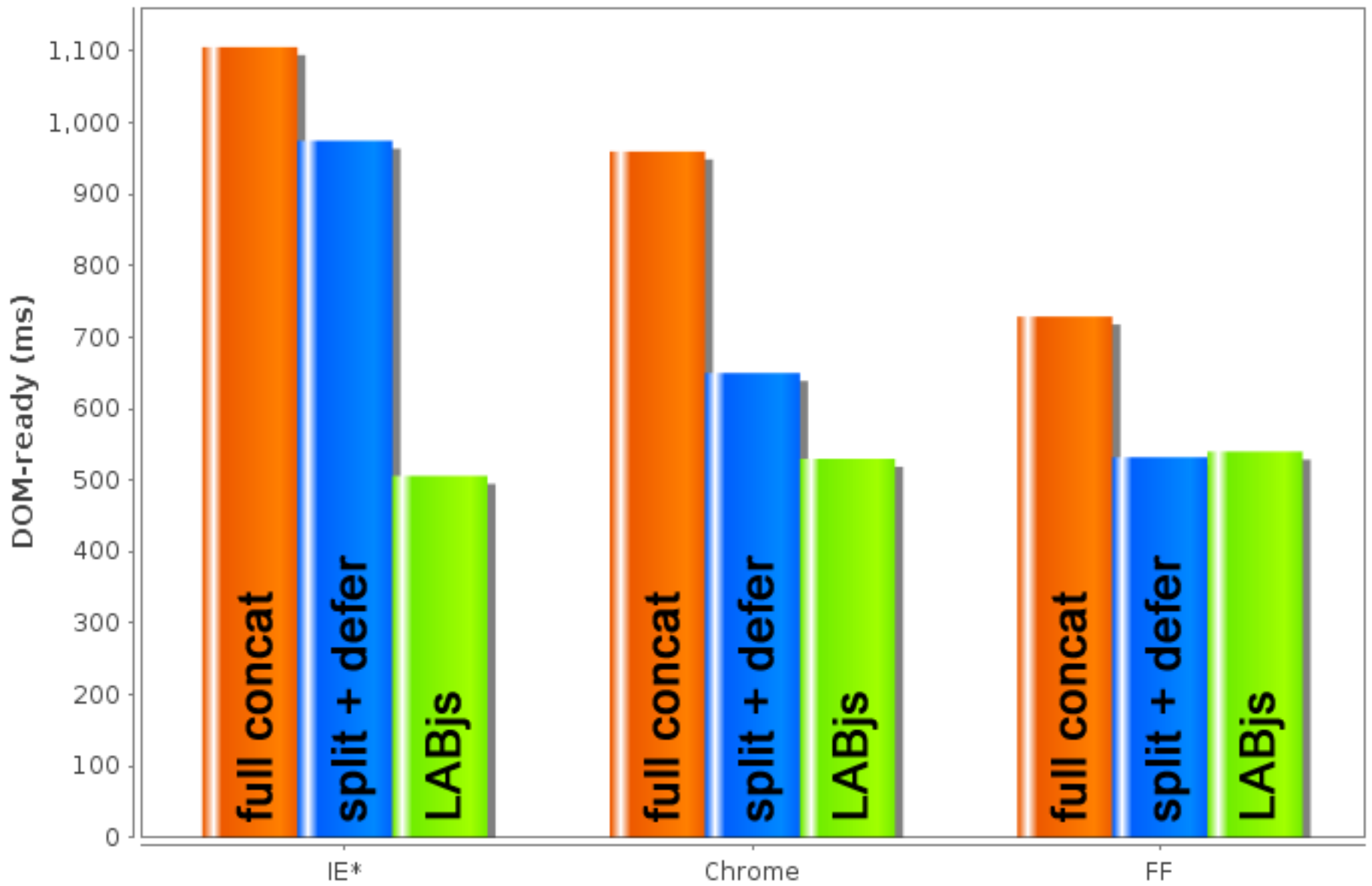
IE8, FF3.5/3.6, Chr 14-



IE9+, FF4+



Script Loaders



comparing DOM-ready times across loading techniques



`<script>` tags also suck because...

- browser-specific scripts
- conditional loading/URLs
- event handling



WTF Loader

```
document.write("<script src='...'></sc"+"ript>");
```


A man in a conical helmet and chainmail armor, holding a sword, looking upwards with a determined expression. The background is a bright, overcast sky. The man is positioned in the center-right of the frame, with a large rock formation on the left. The text "Surely we can do better!?" is overlaid on a black banner at the bottom.

Surely we can do better!?

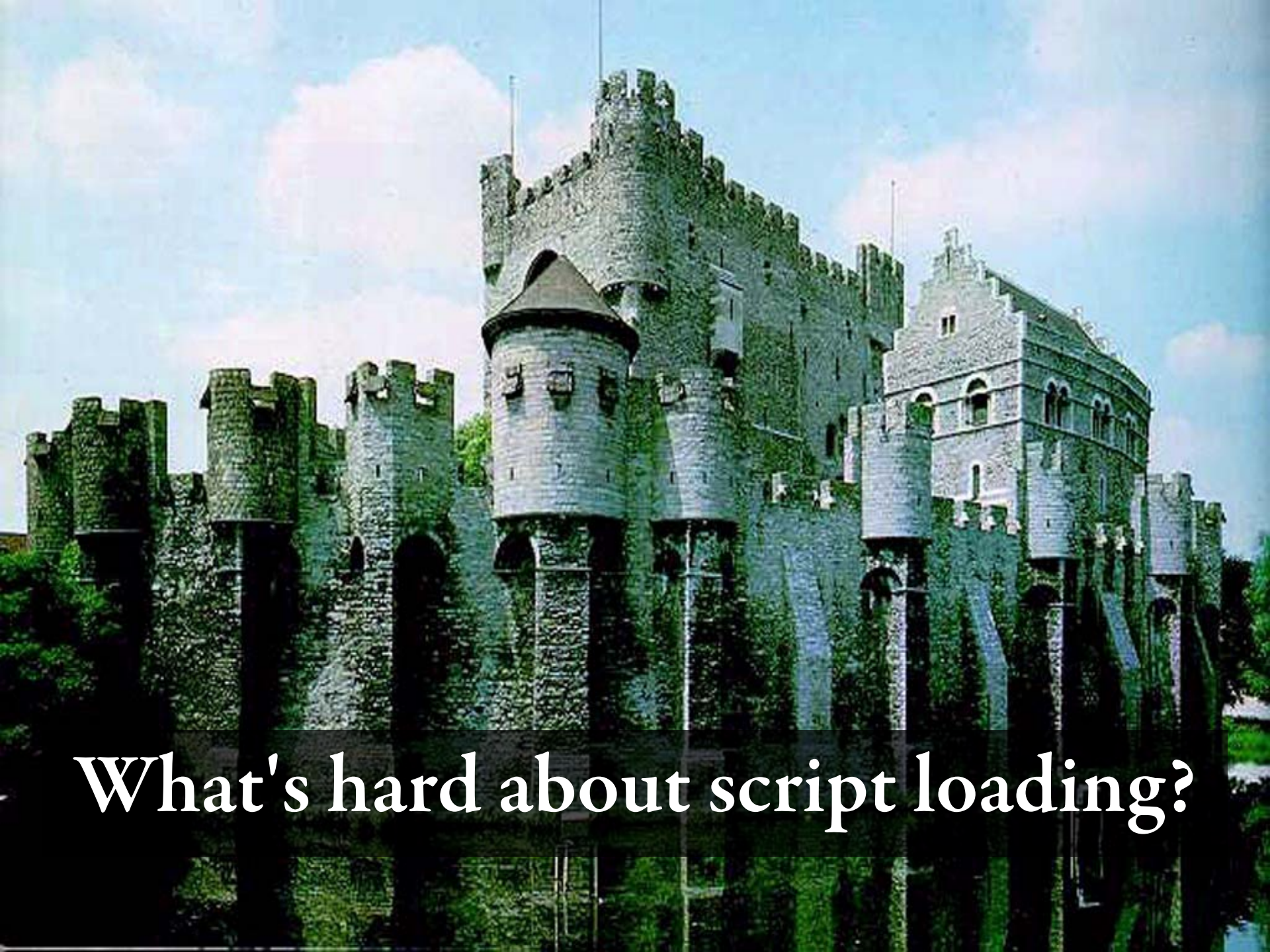
XHR?

```
1 function loadScript(src,cb) {
2     var xhr = XMLHttpRequest ? new XMLHttpRequest() : new ActiveXObject("Microsoft.XMLHTTP")
3     head = document.head || document.getElementsByTagName("head")[0]
4     ;
5     xhr.onreadystatechange = function() {
6         if (xhr.readyState == 4) {
7             var script = document.createElement("script");
8             script.text = xhr.responseText; // script injected.. could also eval()
9             head.insertBefore(script,head.firstChild);
10            cb();
11        }
12    };
13    xhr.open("GET",src);
14    xhr.send();
15 }
```

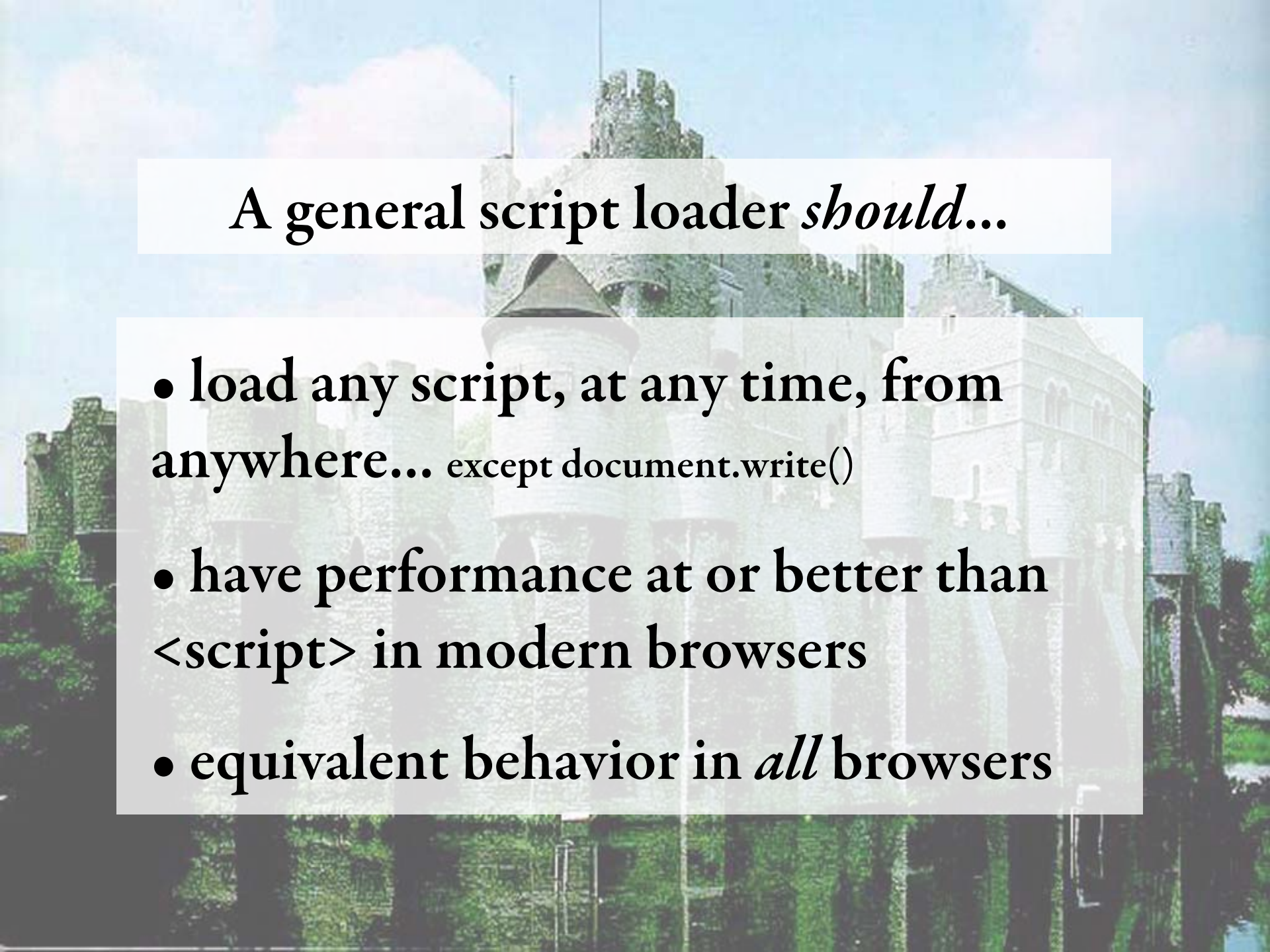


```
1 function loadScript(src,cb) {  
2     var script = document.createElement("script"),  
3     head = document.head || document.getElementsByTagName("head")[0]  
4     ;  
5     script.onload = script.onreadystatechange = function() {  
6         if ( (elem.readyState && elem.readyState != "complete" &&  
7             elem.readyState != "loaded") || script.isDone) return;  
8         script.isDone = true;  
9         cb();  
10    };  
11    script.src = src;  
12    head.insertBefore(script,head.firstChild);  
13 }
```

Making progress... but not *there* yet



What's hard about script loading?



A general script loader *should*...

- load any script, at any time, from anywhere... except `document.write()`
- have performance at or better than `<script>` in modern browsers
- equivalent behavior in *all* browsers



A general script loader should *also*...

- avoid: hacks, UA sniffing
- feature-detect
- have as few exception-cases as possible

General script loader **feature creep**...

- *trying* to handle `document.write()`
- loading CSS
- dependency management
- delaying DOM-ready

Script loading requires...

- loading many scripts in parallel (race to finish loading ASAP)
- ensuring execution order (*not* ASAP)

LABjs



LABjs: *performance* script loader

```
1 <html>
2   <head>
3     <title>Script Loading</title>
4
5     <script src="script1.js" type="text/javascript"></script>
6     <script src="http://some.tld/script2.js" type="text/javascript"></script>
7     <script src="script3.js" type="text/javascript"></script>
8
9   </head>
10  <body>
11    
12  </body>
13 </html>
```

```
1 <html>
2   <head>
3     <title>Script Loading</title>
4     <script>
5       $LAB
6         .script("script1.js")
7         .script("http://some.tld/script2.js")
8         .script("script3.js");
9     </script>
10  </head>
11  <body>
12    
13  </body>
14 </html>
```

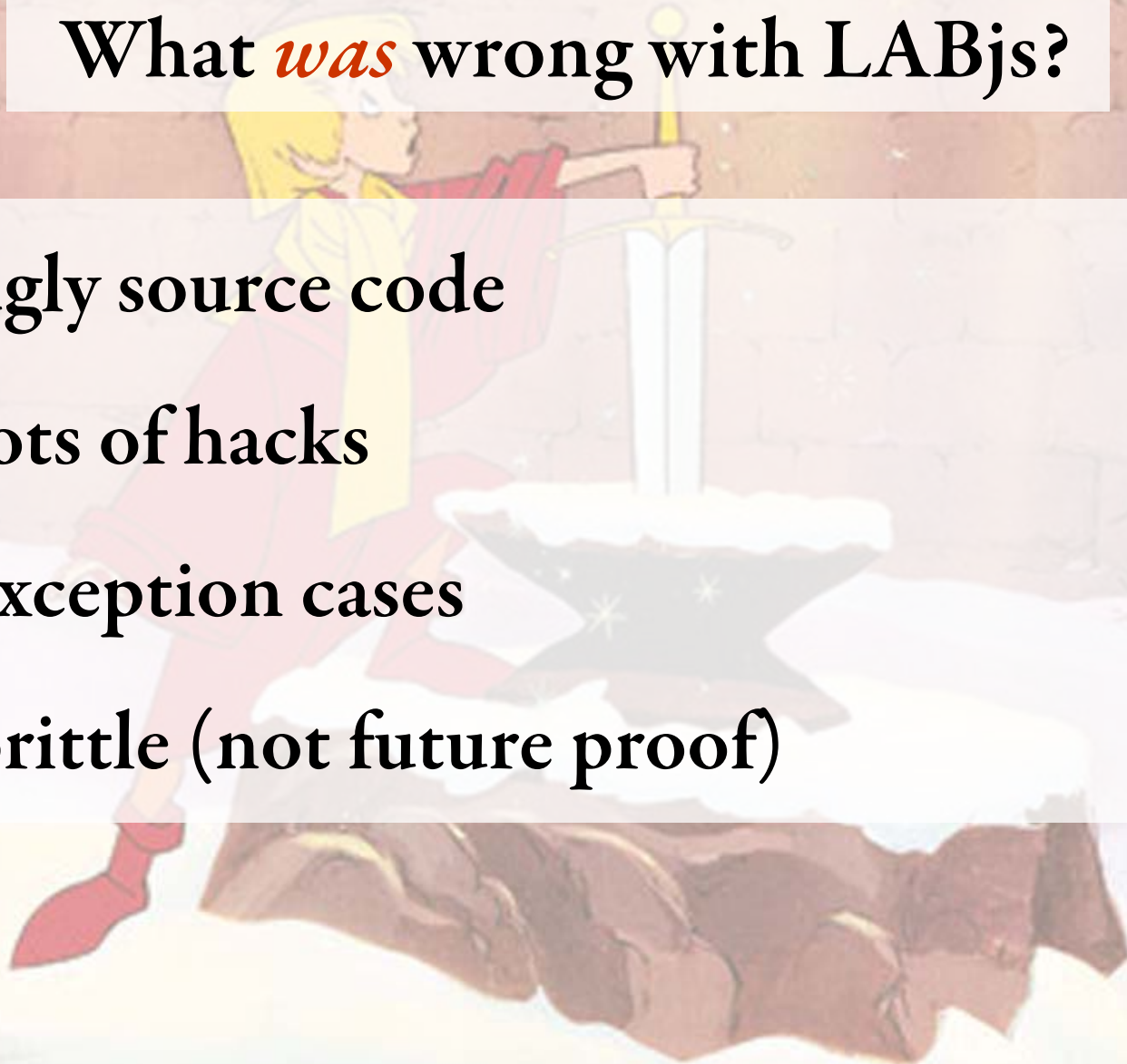

LABjs: *performance* script loader

```
1 <html>
2   <head>
3     <title>Script Loading</title>
4
5     <script src="script1.js" type="text/javascript"></script>
6     <script src="http://some.tld/script2.js" type="text/javascript"></script>
7     <script>script2Init("Hi");</script>
8     <script src="script3.js" type="text/javascript"></script>
9     <script>script3Init();</script>
10
11   </head>
12   <body>
13     
14   </body>
15 </html>
```

```
1 <html>
2   <head>
3     <title>Script Loading</title>
4     <script>
5       $LAB
6       .script("script1.js")
7       .script("http://some.tld/script2.js")
8       .wait(function(){script2Init("Hi");})
9       .script("script3.js")
10      .wait(script3Init);
11    </script>
12  </head>
13  <body>
14    
15  </body>
16 </html>
```


What *was* wrong with LABjs?

- ugly source code
- lots of hacks
- exception cases
- brittle (not future proof)





LABjs 2.0

- *readable* source code
- "future proof" feature detects
- *fewer* exception cases
- better performance

<http://labjs.com>




Anyone can write a loader!


(and *many* devs have)

| LABjs | HeadJS | ControlJS | RequireJS | Load.js | YepNope.js | \$script.js | LazyLoad | curl.js | JsDefer | jquery.defer |
|----------------------------------|---------|-----------|---|---------------------------------------|---|---------------------------------------|------------------------------|---|-------------|--------------|
| Y | Y | Y | Y | Y | Y | Y | N | Y | Y | Y |
| Y | | Y | Y(API) | Y? -- see note about IE/Webkit/Chrome | Y | Y? -- see note about IE/Webkit/Chrome | n/a | N (cujo is meant to be used with an optimizer which bundles dependencies) | Y | Y |
| Y | | Y | Y | N | Y | N | N | N | N | N |
| Y | | | Y(using path config) | N (chains negate the need for ids) | Y | Y | n/a | Y(using path config) | Y | Y |
| N | | | | N (chains negate the need for ids) | Y | | N | N | N | N |
| N IE/Webkit/Chrome
Y FF/Opera | Y | N | N (UNLESS you use the Order plugin explicitly on each dependency, which is your own fault) -- see note about FF/Opera | N -- see note about FF/Opera | Y (On purpose, as a default, pluginable (unreleased/unsupported), though) | N -- see note about FF/Opera | N -- see note about FF/Opera | N | N | N |
| Y | | N | Y | Y | Y | Y | Y | Y | Y | Y |
| Y (queueing) | | N | Y | Y | Y | Y | N | Y | Y | Y |
| N (sorta) | | N | Y | Y | Y | | N | Y (CommonJS AMD) | Y | Y |
| N | N | N | Y | N | N | Y | N | Y | N | N |
| Y | | Y | Y | Y | Y | Y | Y | Y | Y | Y |
| Y | | Y | Y | n/a | Y | n/a | Partial | Y (feature detection only!) | n/a | n/a |
| N | | Y | N | N | N | N | Partial | N | N | N |
| Y | N (N/A) | N (N/A) | Y(partially) | N | N (N/A) | N | Y | N | N | N |
| N | N | | Y(w/plugin) | N | Y | N | Y | Y (via cssx/css plugin) | N (planned) | N (planned) |
| N (but wrappable) | N | N | Y | N | Y | | N | Y (plugins and extensions) | Y | Y |
| N | N | Y | N | N | N | N | N | N | Y | Y |
| Y | N | N | Y | N | Y | Y | N | N (planned via extension) | Y | Y |
| N (possibly coming) | N | Y | N | N | Y | N | N | N | Y | Y |


<https://spreadsheets0.google.com/spreadsheet/ccc?key=tDdcrv9wNQRCNCRCfWxhYQ>

A medieval painting depicting three monks in a forest. The monks are wearing white and brown robes with hoods. They are standing in a line, looking towards the left. The background shows trees and foliage. A semi-transparent white box with black text is overlaid on the center of the image.

**Loaders are competing
more on APIs than on
features or performance**

A scene from a medieval manuscript depicting three knights in armor standing in a forest. They are looking at a fourth knight who is holding a shield. The text "They're also copying each other (good and bad)" is overlaid on the image.

**They're also copying
each other (good and bad)**

A man in a conical helmet and chainmail armor is pointing his finger at a woman. The woman has a long, yellow, pointed nose and is wearing a grey headscarf. They are standing in front of a stone wall.

But, can their
functionality be
trusted?



Testing is *much* harder than
most realize

A close-up photograph of a white horse in traditional jousting armor. The horse is wearing a brown leather bridle with gold-colored metal plates and a decorative collar. Its mane is long and white. In the background, a jousting arena is visible with a fence and other participants on horseback. A semi-transparent black box with white text is overlaid on the left side of the image.

What *should* a
loader do?

A white horse in traditional armor with a rider in the background. The horse is wearing a brown leather bridle with gold-colored metal ornaments. The rider is wearing a white helmet and armor. The background is a blurred outdoor setting with yellow and red vertical stripes.

Real Preloading

http://wiki.whatwg.org/wiki/Script_Execution_Control


```
1 function executePreloadedScript(script) {
2     var head = document.head || document.getElementsByTagName("head")[0];
3     head.insertBefore(script, head.firstChild);
4 }
5 function preloadScript(src, cb) {
6     var script = document.createElement("script");
7     // explicit preloading (Zakas)
8     if (typeof script.preload == "boolean") {
9         script.preload = true;
10        script.onpreload = function() {
11            cb(script);
12            script.onpreload = null;
13        };
14        script.src = src;
15    }
16    // implicit preloading (WHATWG, IE4+)
17    else if (script.readyState && script.readyState == "uninitialized") {
18        script.onreadystatechange = function() {
19            if (script.readyState == "loaded") onload(script);
20            script.onreadystatechange = null;
21        };
22        script.src = src;
23    }
24    else {
25        // ...
26    }
27 }
28 preloadScript("../", function(script) {
29     setTimeout(function() { executePreloadedScript(script); }, 5000);
30 });
```

IE4+ ftw?



Ordered Async

`async=false`

FF4+, Chr 12+, IE10p2+, Webkit/Safari, Opera (soon!)

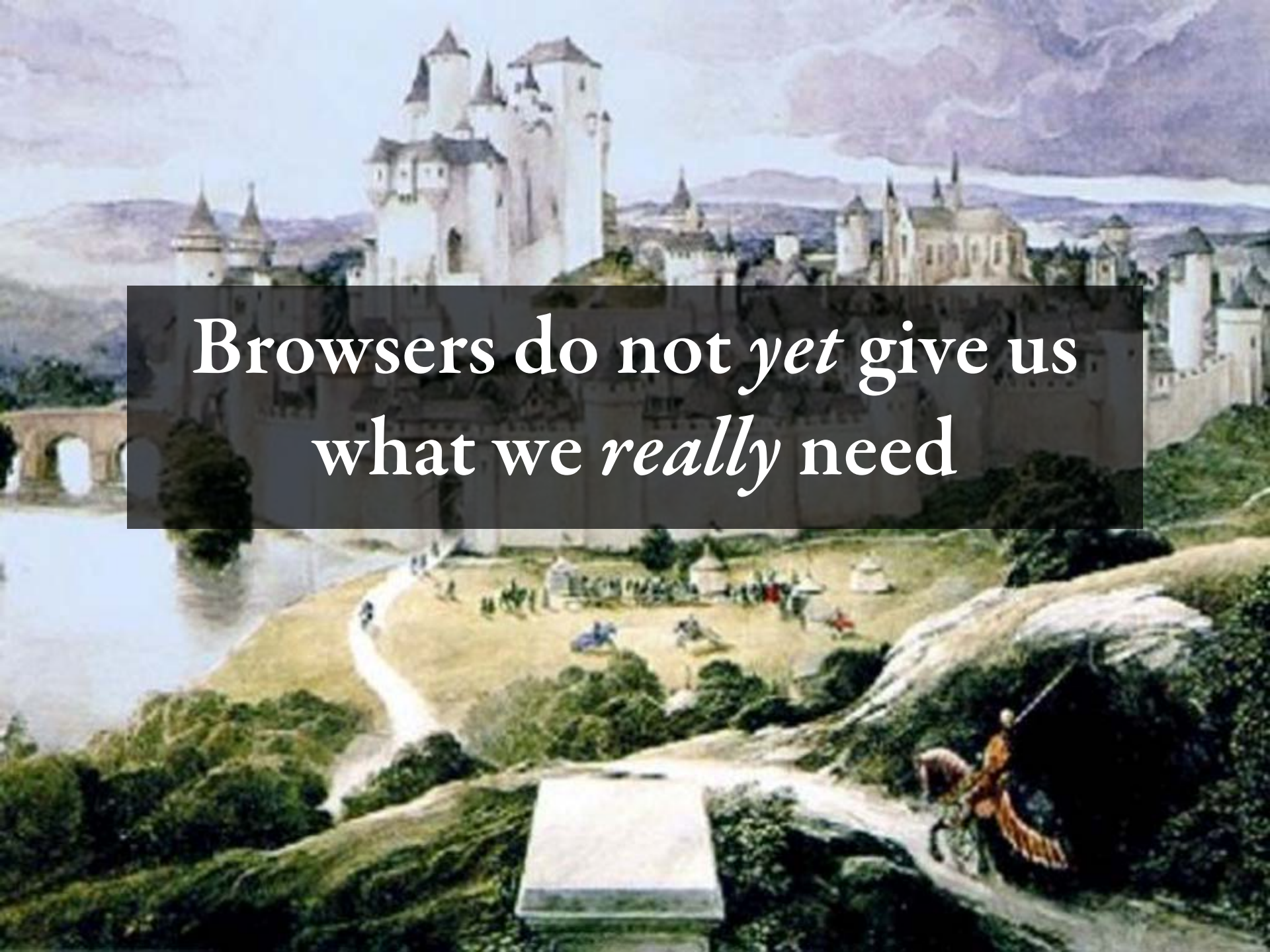
http://wiki.whatwg.org/wiki/Dynamic_Script_Execution_Order

How?

1. try real preloading
2. try ordered async
3. try same-domain XHR
4. fall back on "cache preloading"



Competition is good, *only* if
community is educated

A medieval-style painting of a castle on a hill. The castle has multiple towers with conical roofs and is surrounded by a river. In the foreground, a dragon is breathing fire, and a path leads up to the castle. The scene is set in a lush, green landscape with a river and a bridge in the background.

Browsers do not *yet* give us
what we *really* need



**Co-opetition is much
healthier for the
community**



W3C, WHATWG

<http://wiki.whatwg.org/wiki/Category:Proposals>



W3C, WHATWG

<http://ygp.go.ly/script-preloading>


<http://odq.go.ly/load-error-events>



We'll accomplish more if we
work together



Future, The Script Loader



Preloading

(deferred execution)



Modules

(CommonJS, AMD, etc)

The background of the slide is a religious painting. It depicts a bearded man, likely a saint or a deity, with a long, dark beard and a red crown adorned with jewels. He has striking yellow eyes and is surrounded by a bright, golden halo. The scene is set against a backdrop of soft, white and yellow clouds.

Native Modules

(ES-Harmony?)

HOLY GRAIL





"Script Loader of my dreams"

What else?

- **timeouts**
- **load abort**
- **error handling**
- **load priority**

A close-up shot of three LEGO knights in medieval armor. The knight on the left has a brown helmet and a blue tunic with a black and white crest. The knight in the middle has a grey helmet and a red tunic with a blue crest. The knight on the right has a grey helmet and a red tunic with a black and gold crest. They are standing in front of a stone wall with a checkered pattern and a small window. The text "And now for something completely different..." is overlaid in white on a dark grey background.

And now for something
completely different...

Questions?



Kyle Simpson
@getify
<http://getify.me>

<http://wiki.whatwg.org/wiki/Category:Proposals>

<http://labjs.com>