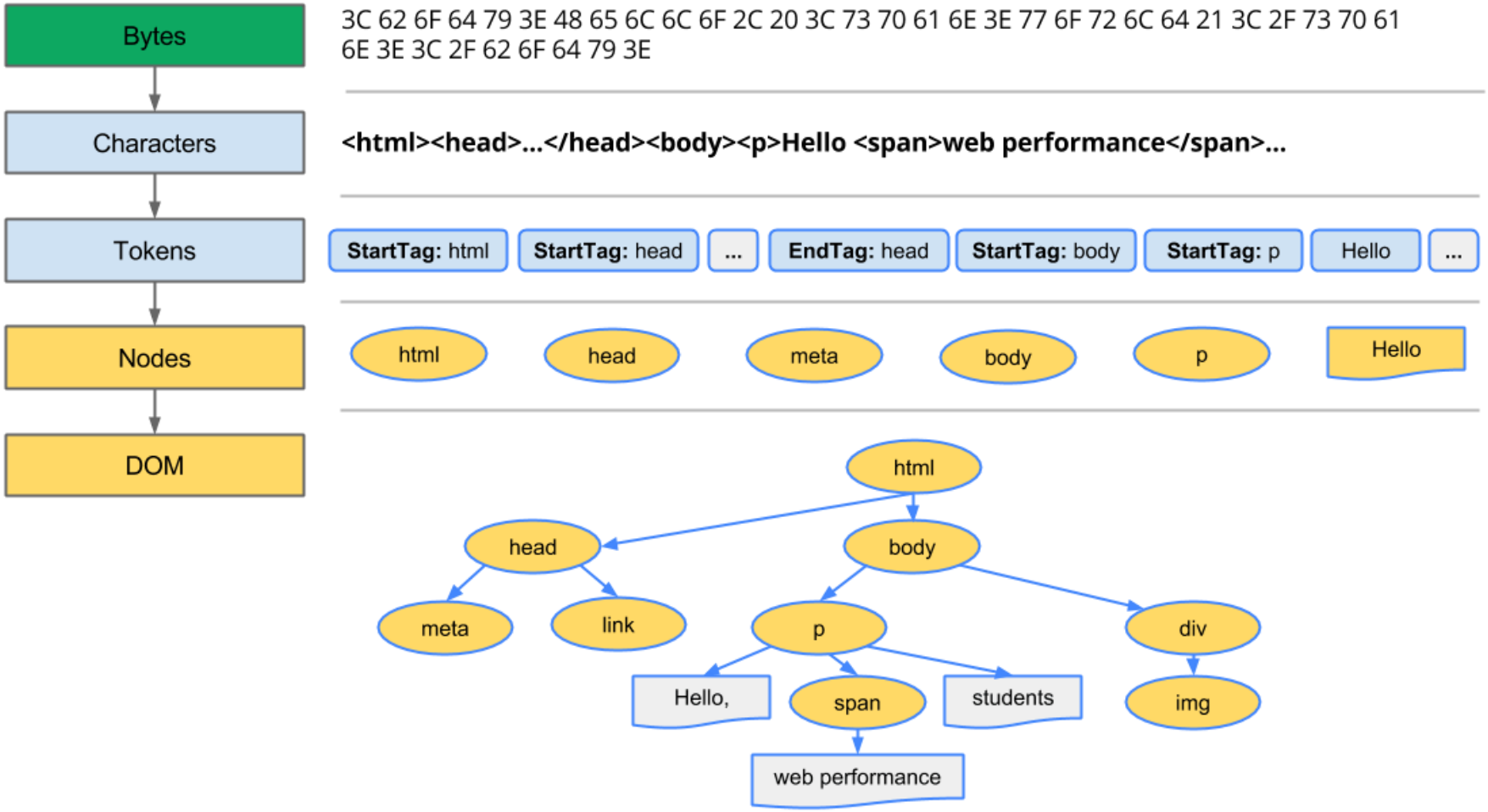
Critical Rendering Path

Constructing Document Object Model

```
<html>
<head>
    <meta name="viewport" content="width=device-width,initial-scale=1">
    <link href="style.css" rel="stylesheet">
    <title>Critical Path</title>
</head>
<body>
    <p>Hello <span>web performance</span> students!</p>
    <div>
        <img src="awesome-photo.jpg">
    </div>
</body>
</html>
```

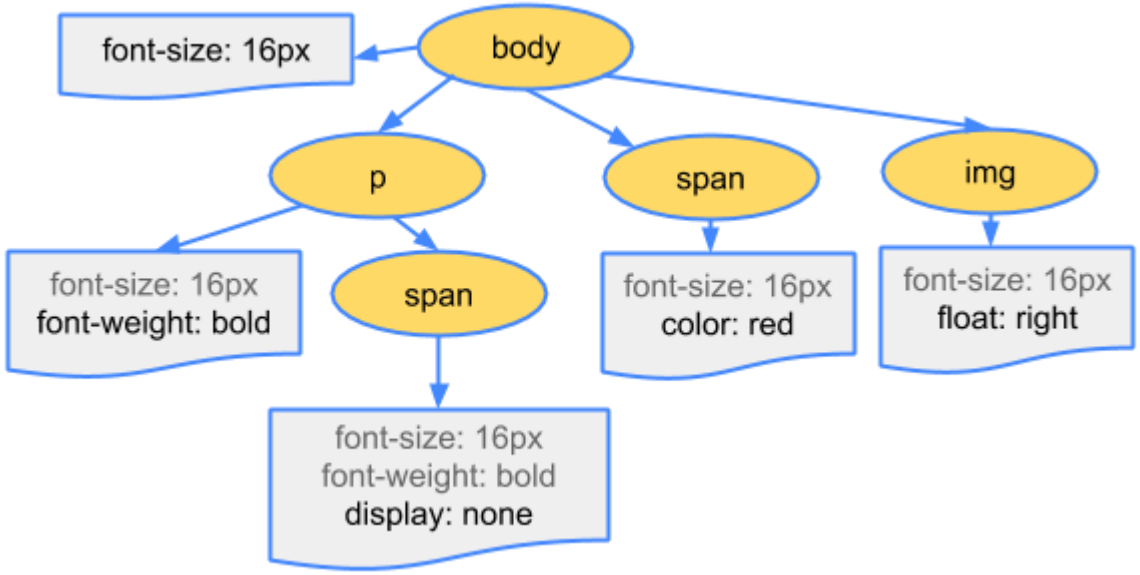
Как браузер обрабатывает данный HTML?

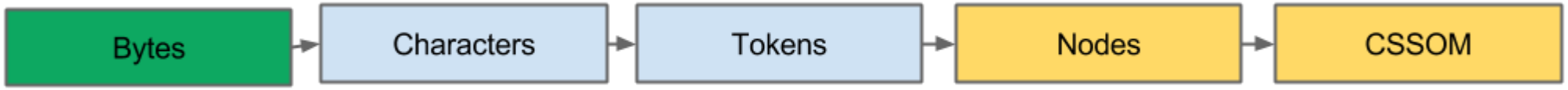
Constructing Document Object Model



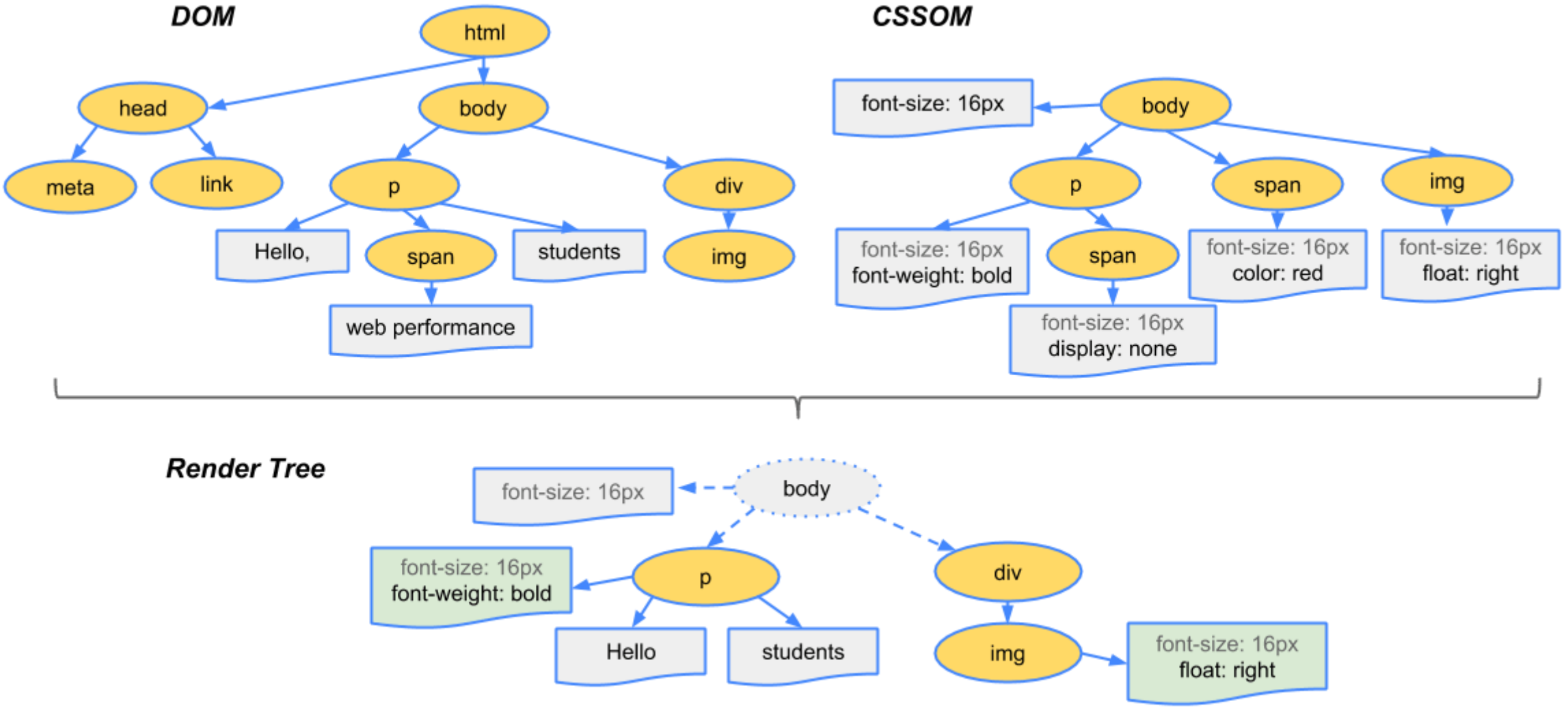
Constructing CSS Object Model

```
body { font-size: 16px }

p { font-weight: bold }

span { color: red }

p span { display: none }

img { float: right }
```





Constructing Render Tree



Reflow (Layout) and Repaint (Rasterizing)

1.
Во время **Reflow** вычисляются координаты и размеры DOM элементов из Render Tree.
Все относительные величины транслируются в абсолютные.
Результатом является **Box Model** — свой прямоугольник каждого DOM элемента.

2.
Во время **Repaint** происходит отрисовка DOM элементов из Render Tree.
К этому моменту уже вычислены их размеры, положение, визуальное оформление.

JavaScript execution blocks on the CSSOM

```
<html>
<head>
    <link href="style.css" rel="stylesheet">
    <title>Critical Path</title>
</head>
<body>
    <p>
        Hello<span>web performance</span>
students!
    </p>
    <div>
        <img src="awesome-photo.jpg">
    </div>
    <script src="app.js"></script>
</body>
</html>
```

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

// change DOM text content
span.textContent = 'interactive’;

// change CSSOM property
span.style.display = 'inline';

// create a new element, style it, and append to DOM
var loadTime = document.createElement('div');
loadTime.textContent = 'Loaded on: ' + new Date();

loadTime.style.color = 'blue';
document.body.appendChild(loadTime);
```

What triggers a reflow or a repaint?

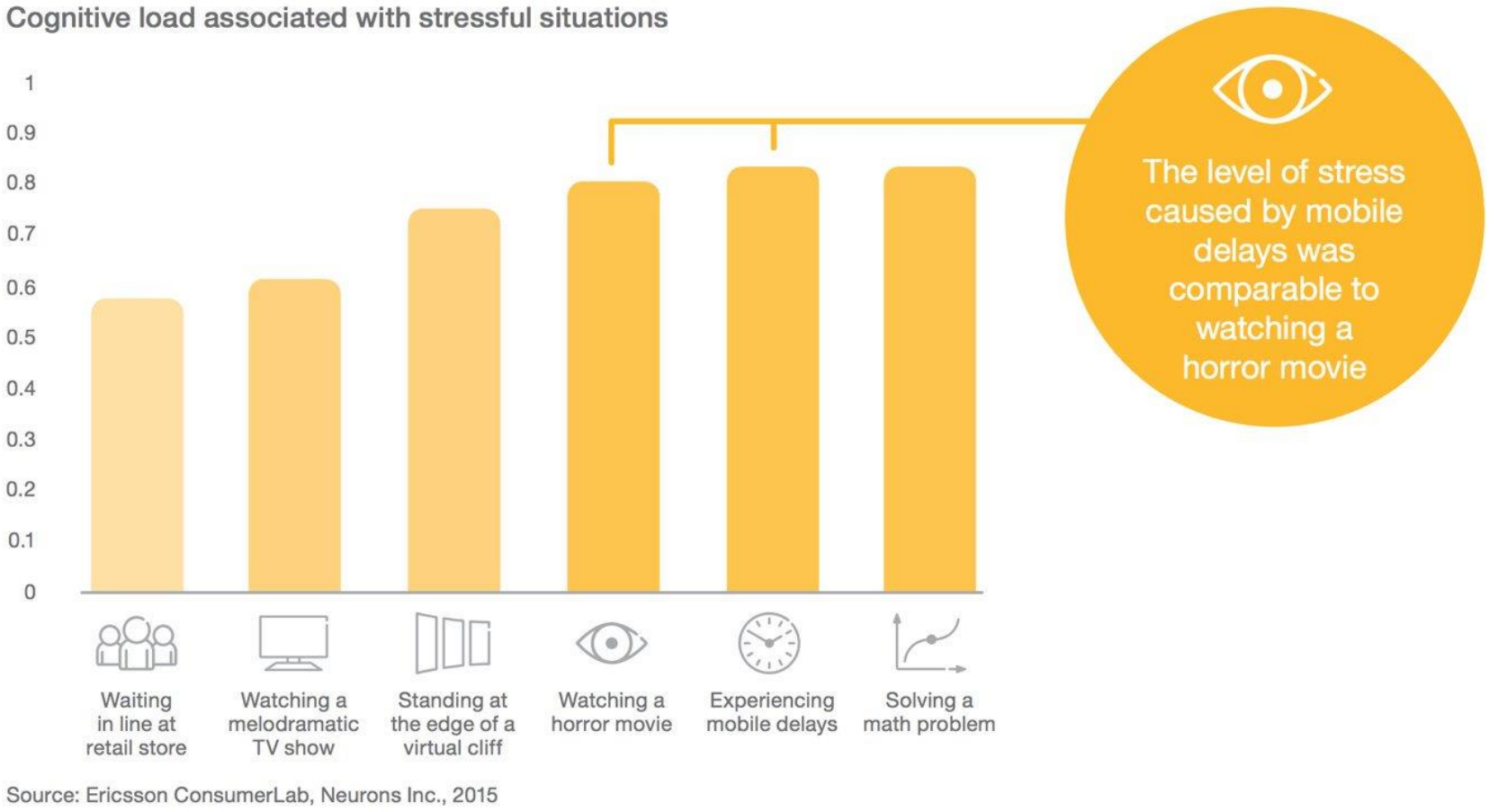
```
let bstyle = document.body.style;

bstyle.padding = "20px"; // reflow, repaint

bstyle.border = "10px solid red"; // another reflow and a repaint

bstyle.color = "blue"; // repaint only, no dimensions changed

bstyle.backgroundColor = "#fad"; // repaint

bstyle.fontSize = "2em"; // reflow, repaint

// new DOM element - reflow, repaint
document.body.appendChild(document.createTextNode('dude!'));
```

What else trigger reflow?

```
let dStyle = document.getElementById('abc').style;

let offset = dStyle.offsetLeft;

let scroll = dStyle.scrollTop;

let width = dStyle.offsetWidth;

let computed = dStyle.getComputedStyle();
```

Web Performance

Slow site is very uncomfortable



Performance affects business

— In 2016, AliExpress made their site faster by a third and received 10.5% more orders

— Back in 2006, Google tried making the search slower by half-a-second and discovered that users were making 25% fewer requests

— In 2008, Aberdeen Group discovered that slowing a site down by one second decreases the user satisfaction by 16%

What is performance?

1. Server responds quickly
2. **Web app loads and renders quickly**
3. Web app works quickly

JavaScript code minification

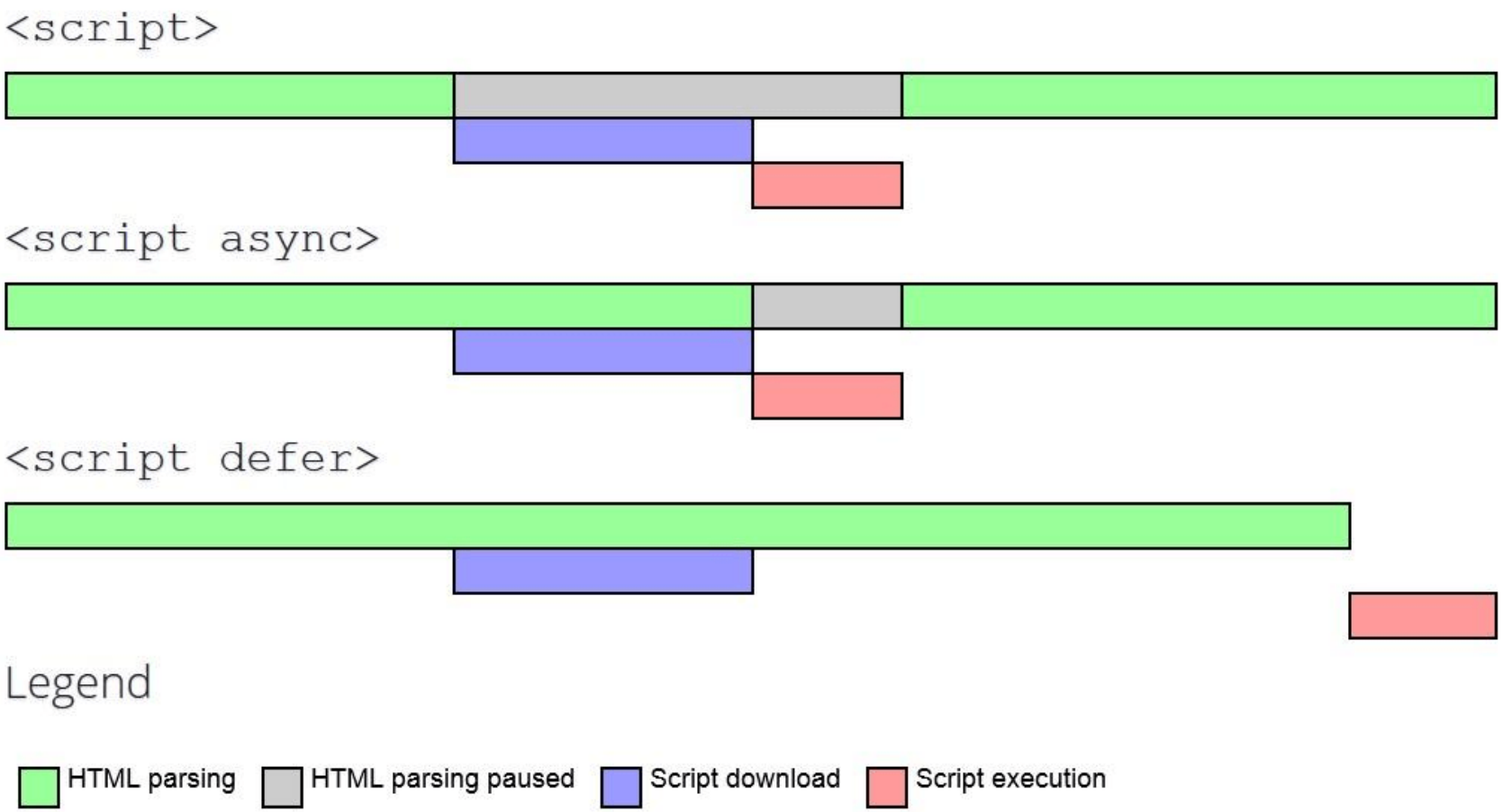
Original:

```
function logArrayItems(element, index) {
    console.log('a[' + index + '] = ' + element);
}

[1, 2, 3, 5, 8].forEach(logArrayItems);
```

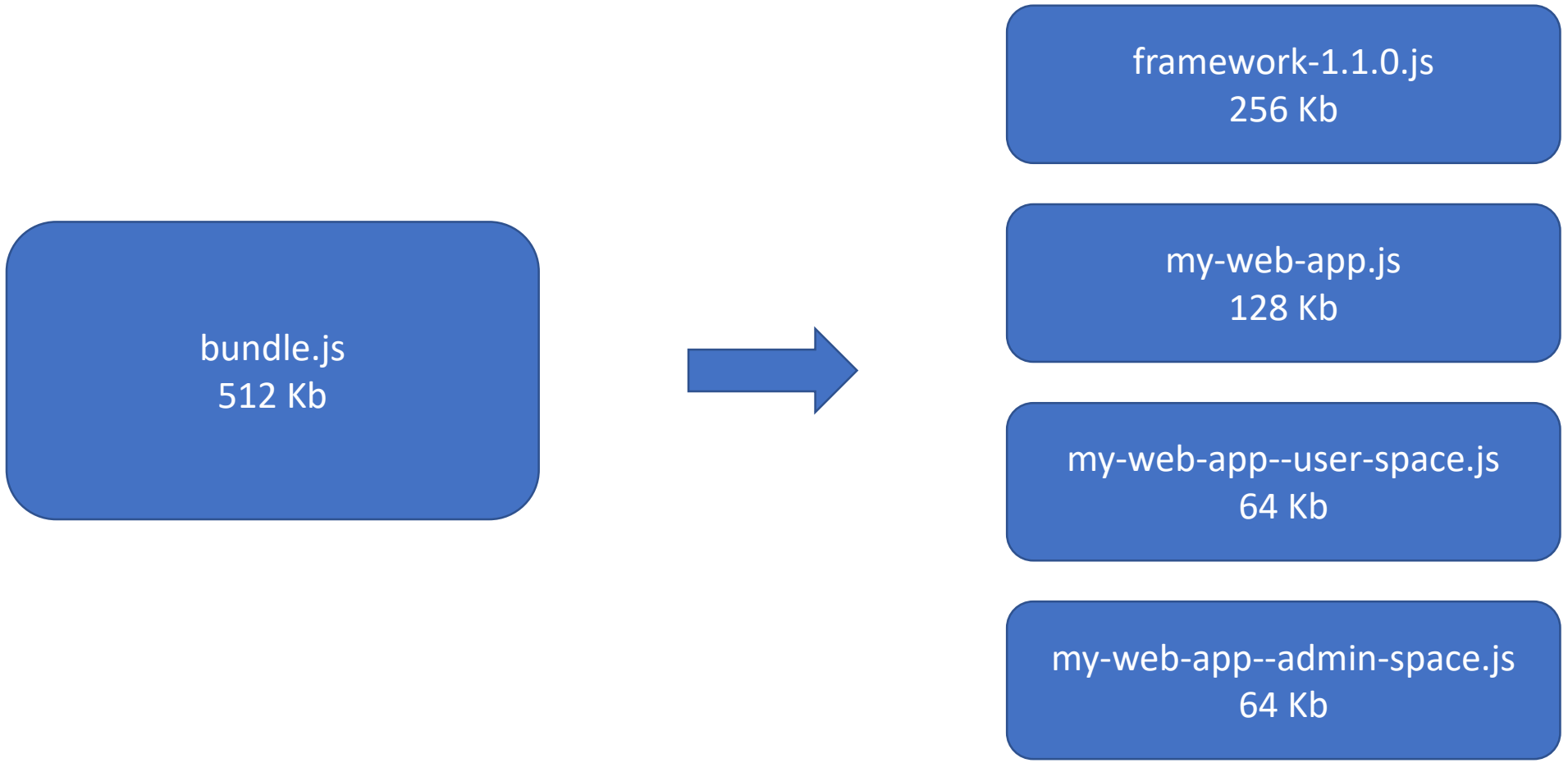
Minified:

```
function logArrayItems(o,r){console.log("a["+r+"] = "+o)}
[1,2,3,5,8].forEach(logArrayItems);
```

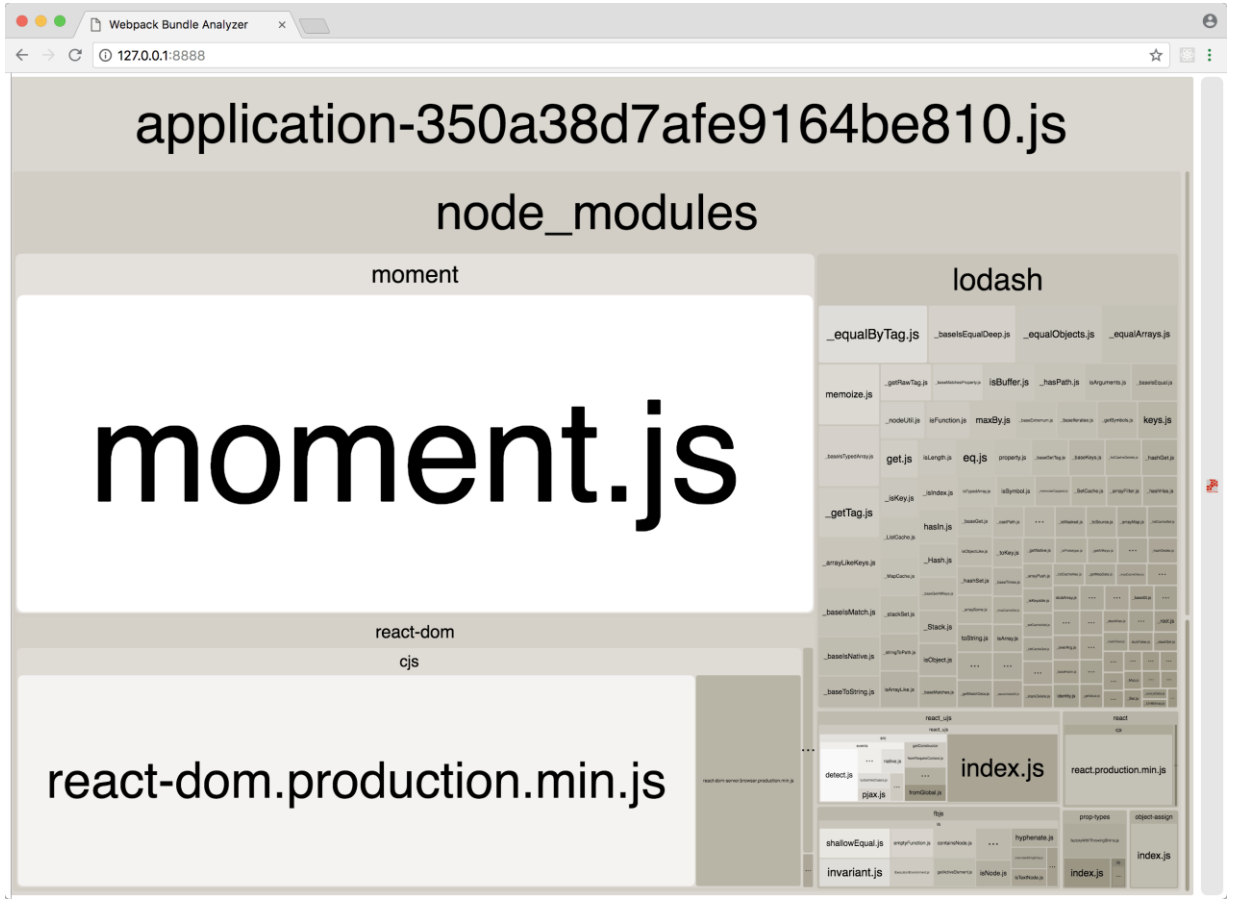
Asynchronous loading JavaScript



Code splitting



Unused code in dependencies



CSS Minification & Optimization

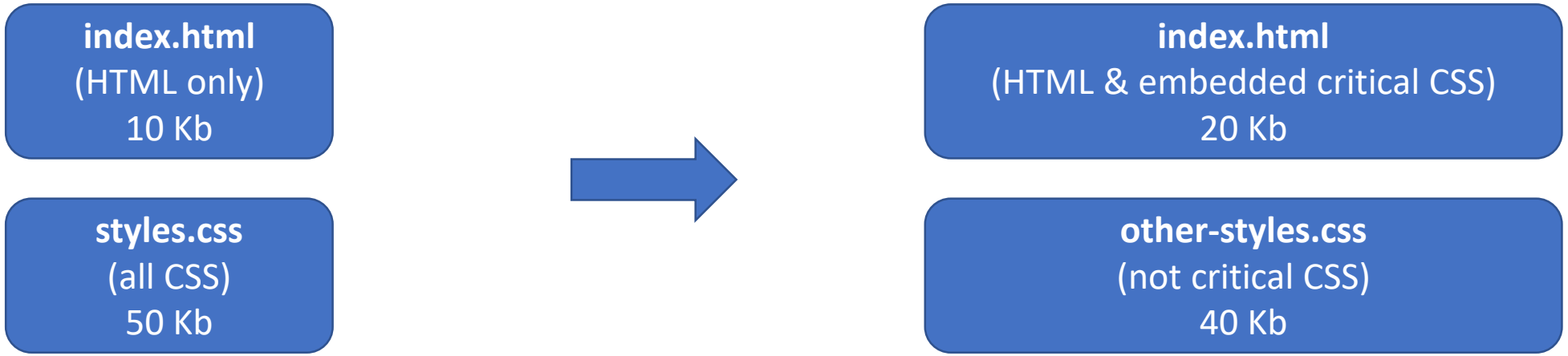
Original:

```
.foo {
    width: 200px;
    height: 100px;
}

.bar {
    border-width: 1px;
    border-style: solid;
    border-color: red;
}
```

Optimized:

```
.foo{width:200px;height:100
px}.bar{border:1px solid
red}
```

Embedding critical styles



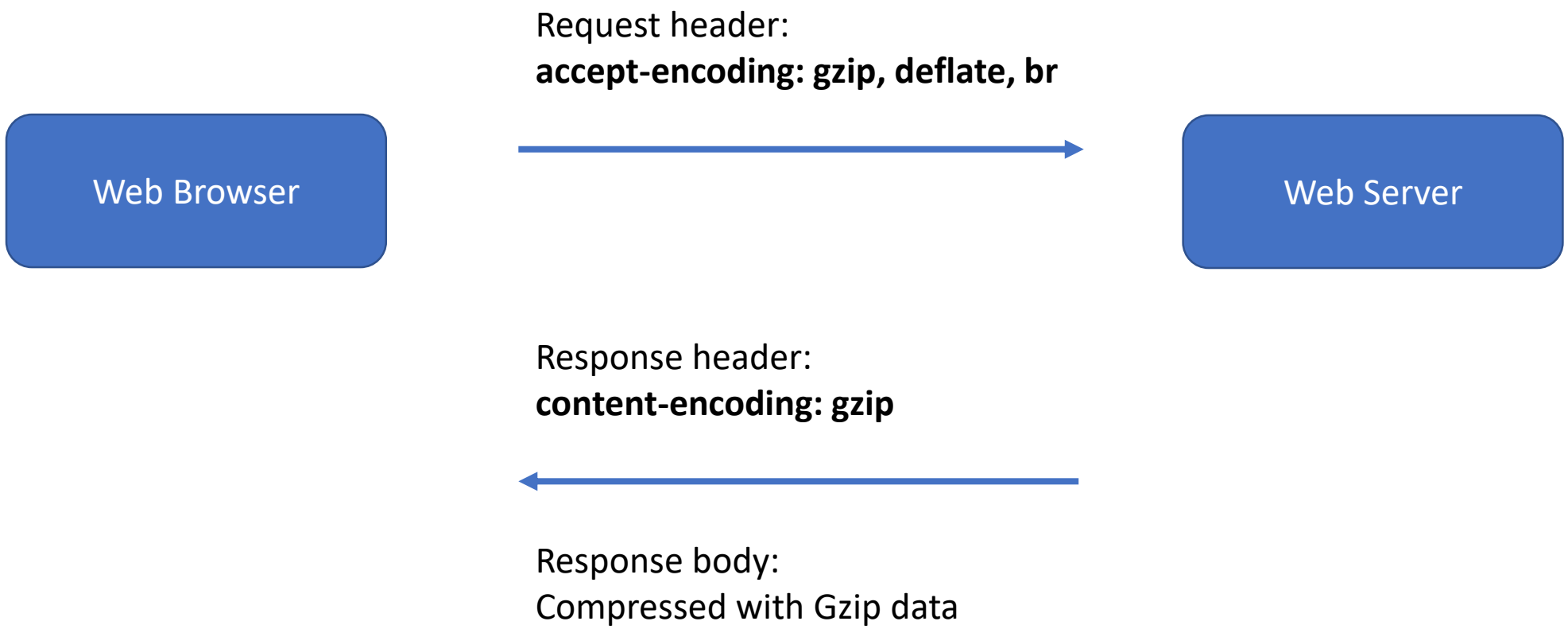

```
<style>
    /* critical styles */
</style>
<link rel="preload" href="other-styles.css" as="style" onload="this.rel = 'stylesheet'" />
```

HTML Minification

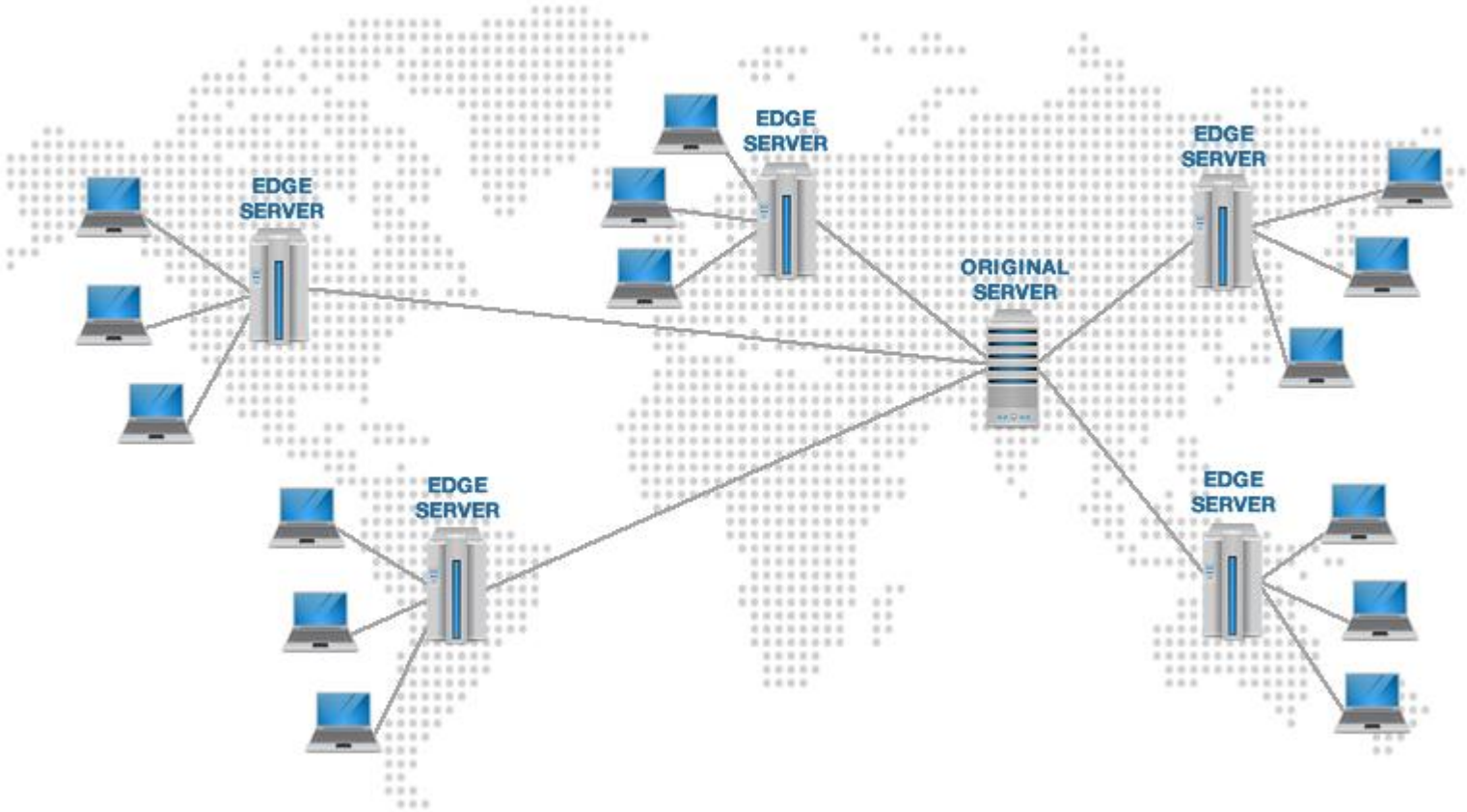
```
<!DOCTYPE html>
<html>

<head>
    <title>Page Title</title>
</head>

<body>
    <h1>This is a Heading</h1>
    <!-- comment -->
    <p>This is a paragraph.</p>
</body>

</html>
```

```
<!DOCTYPE
html><html><head><title>Page
Title</title></head><body><h1>T
his is a Heading</h1><p>This is
a paragraph.</p></body></html>
```

HTTP data compression



Content Delivery Network



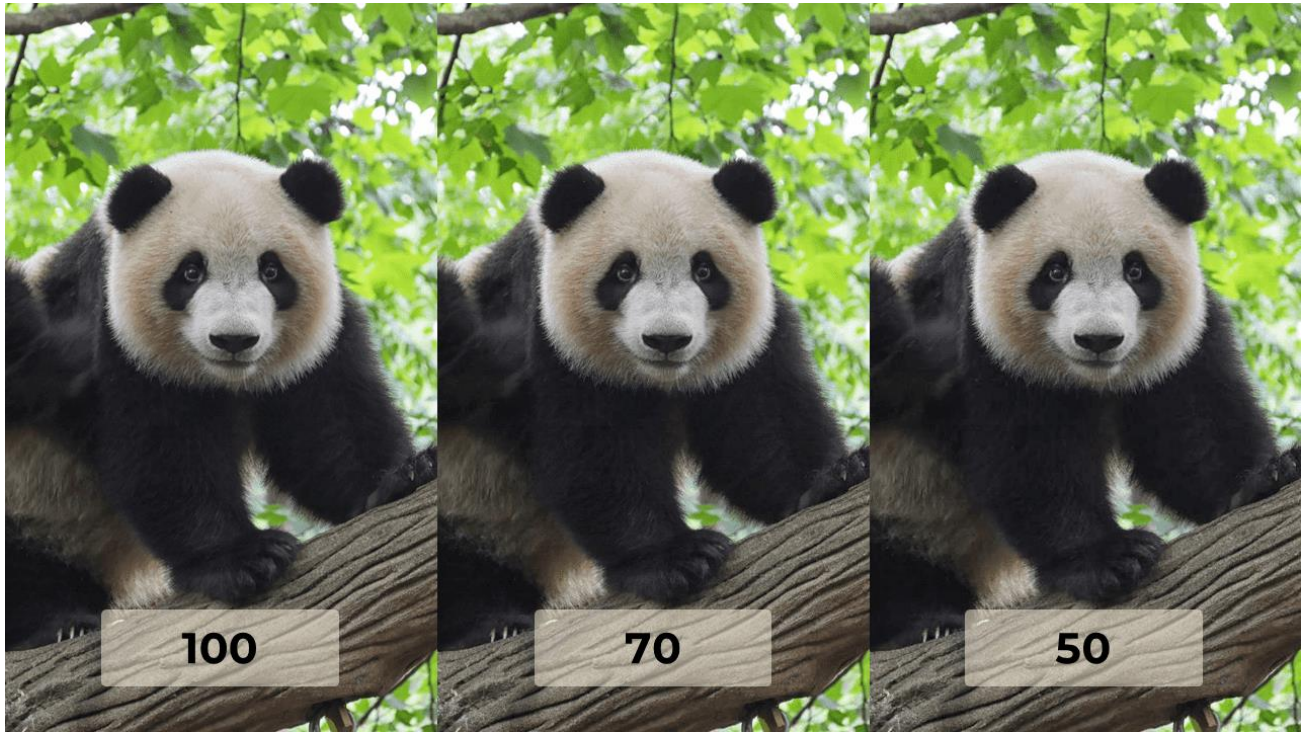
Preloading resources

```
<!-- DNS resolve -->
<link rel="dns-prefetch" href="//example.com">

<!-- TCP (TLS) connection establish -->
<link rel="preconnect" href="http://css-tricks.com">

<!-- Preload with low priority -->
<link rel="prefetch" href="image.png">

<!-- Preload with high priority -->
<link rel="preload" href="image.png">

<!-- Prerender page -->
<link rel="prerender" href="http://css-tricks.com">
```

Suitable image format

- **svg** is best for vector images such as icons or logos
- **jpg** is best for photos because it compresses images with a slight quality loss not visible by the human eye
- **png** is best for raster graphics that you want to display without any quality losses – e.g., raster icons or pixel art
- **mp4** is best for animation (don’t use **gif** at all)

JPG compression level



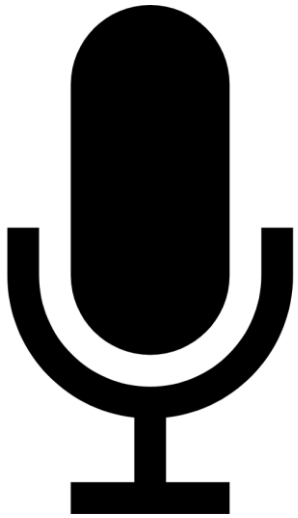
669 Kb

100 Kb

74 Kb

SVG Minification & Optimization

```
<!-- Generated by IcoMoon.io -->
<svg version="1.1"
xmlns="http://www.w3.org/2000/svg"
width="512" height="512" viewBox="0 0
512 512">
    <title></title>
    <g id="icomoon-ignore">
    </g>
    <path
        d="M240 352c44.183 0 80-35.817
80-80v-192c0-44.183-35.817-80-80-80s-
80 35.817-80 80v192c0 44.183 35.818 80
80 80zM352 224v48c0 61.855-50.145 112-
112 112s-112-50.145-112-112v-48h-
32v48c0 74.119 56.002 135.15 128
143.11v64.89h-64v32h160v-32h-64v-
64.89c71.997-7.96 128-68.991 128-
143.11v-48h-32z">
    </path>
</svg>
```



```
<svg
xmlns="http://www.w3.org/2000/svg"
width="512" height="512" viewBox="0
0 512 512"><path d="M240 352c44.2 0
80-35.8 80-80v-192c0-44.2-35.8-80-
80-80s-80 35.8-80 80v192c0 44.2
35.8 80 80 80zM352 224v48c0 61.9-
50.1 112-112 112s-112-50.1-112-
112v-48h-32v48c0 74.1 56 135.2 128
143.1v64.9h-64v32h160v-32h-64v-
64.9c72-8 128-69 128-143.1v-48h-
32z"/></svg>
```