Faculty of Science,
Technology
and Medicine

Web Programming

Volker Müller
University of Luxembourg

PHP 8 Released

26.11.2020: PHP 8 officially released

Improvements:

- JIT compiler → better performance (??)
- Union types: Foo|Bar
- Named function arguments (pass values into function by name, not necessarily order)
- "Attributes" (= Annotations in Java)
- Match expression (better switch)
- Some new useful functions (str_contains( ), ….) …..

React (reactjs.org)

Declarative, efficient, reactive JS library to build user interfaces – not a full MVC framework

Maintained by Facebook + community

Basis for development of single-page or mobile applications

Uses JSX = JS extension to directly embed HTML into JS

Provides components in JS, binding, stateful components

Single Page Applications (SPA)

Web application interacting with users by dynamically rewriting current page rather than loading new pages from server

More similar to Desktop applications

Avoid interruption of user experience (caused by page loading)

Often uses intensively dynamic interaction with web server in background (using Ajax, Web sockets)

SPA must be stateful to record the current "situation"

Example: React Component

```
class Hello extends React.Component {
  render() {  return <button onclick = {
       function( ){ alert('click');}} >
      Hello {this.props.name} </button>  ;  }
} .....

ReactDOM.render(
<Hello name="Volker" />, document.getElementById("i"))
```

Usage of JSX

Many more ex. on React website

Flux

React provides VIEW only, does not include controller or model

Flux = controller / model architecture used by FB for React

Flux flow: actions → dispatcher → data store, changes to store are propagated back to view

Properties in React should not be changed directly, but via callbacks which trigger actions

Conclusion on JS Frameworks

Frameworks can simplify the work of a developer since many standard tasks are provided (with some speed overhead)

Different frameworks have different focus

Tendency that JS frameworks also cover mobile apps based on HTML5, CSS3, JS → progressive web apps

→ It is not easy to decide which framework to use in a concrete case

WebAssembly (webassembly.org)

WebAssembly (Wasm) = binary instruction format that allows C/C++/Rust/.Net code to be run inside a virtual machine in a browser

Code runs almost at native speed in a sandboxed-virtual environment

Wasm must currently be loaded and compiled by JS (future: external loading possible)

Can be combined with WebWorkers & local storage

Rust-WASM Example (on Moodle)

Example for WASM with Rust language available on Moodle

README.md with all steps to setup environment, build and run the example included

Rust (rust-lang.org) is prog. language designed for performance and safety, especially safe concurrency, guarantees memory safety

Becoming increasingly popular in last years

Web Performance Optimization

What is Web Performance Optimization?

Web performance optimization (WPO) = field of knowledge about increasing speed in which web pages are downloaded and displayed

“1-second delay in page load time yields 11% fewer page views and 16% decrease in customer satisfaction” [Aberdeen Group]

“47% of people expect a web page to load in two seconds or less” [Akamai]

Various Resources available on Web

Many websites provide information on WPO:

https://developers.google.com/web/fundamentals/performance

https://developers.google.com/speed/docs/insights/rules

http://yslow.org/

Many Support Tools Exist

Chrome DevTools

Google PageSpeed

Yahoo YSlow, included in FireBug → provides detailed suggestions on what to do

Page speed analysis: http://www.webpagetest.org/

Link checker: https://validator.w3.org/checklink

Google Timing Recommendations

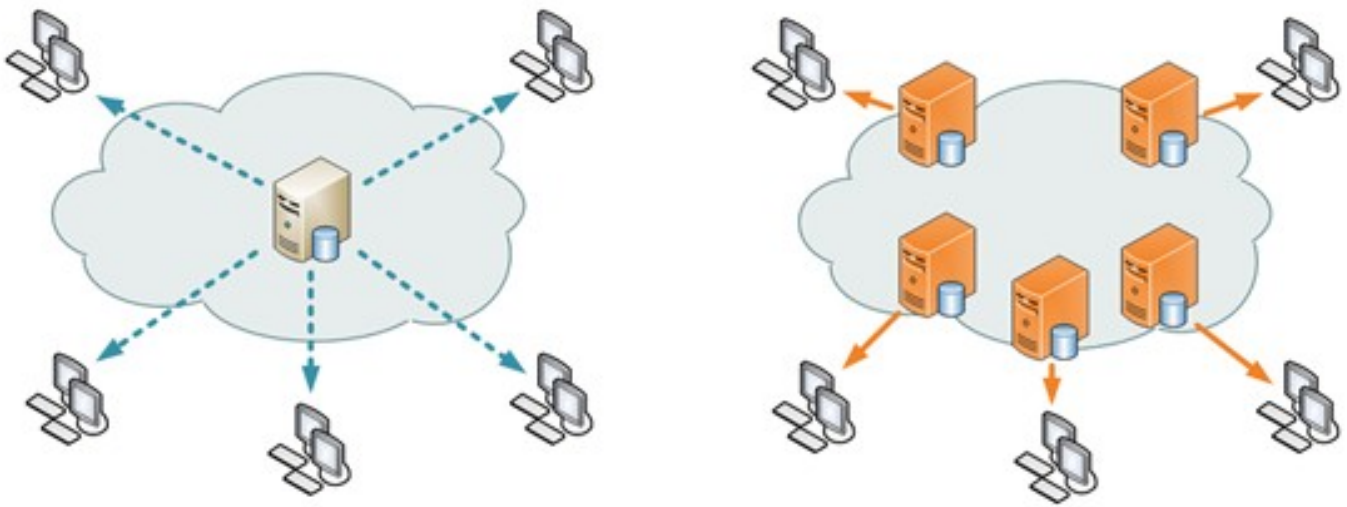
Respond to users immediately; acknowledge user input in under 100ms

When animating or scrolling, produce a frame in under 10ms

Keep users engaged; deliver interactive content in under 1000ms

Web Application Hosting

Many browsers perform max. two simultaneous requests to same domain → host assets (CSS, JS, images, etc.) separated on CDN (Content Delivery Network) allows parallel downloads of resources



Web Application Setup

Reduce server response time under 200 ms:

- Use separate DB server
- Check hardware configuration / consider clustering

Minimize # of redirects

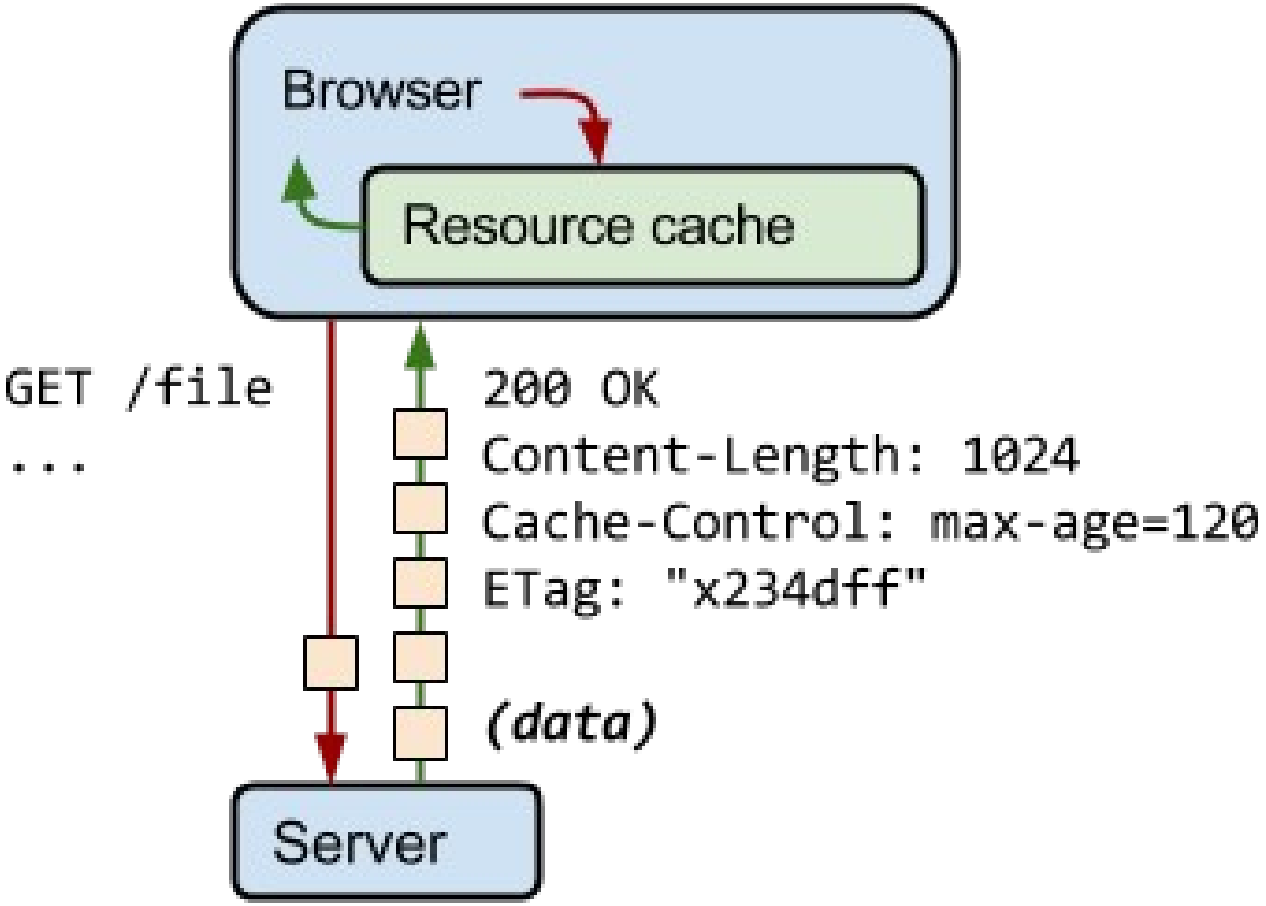
Reduce DNS lookups

Merge multiple JS into one file, also reduce number of CSS files

Avoid bad requests (404) by link checking

HTTP Caching

Every browser ships with HTTP cache



HTTP Caching

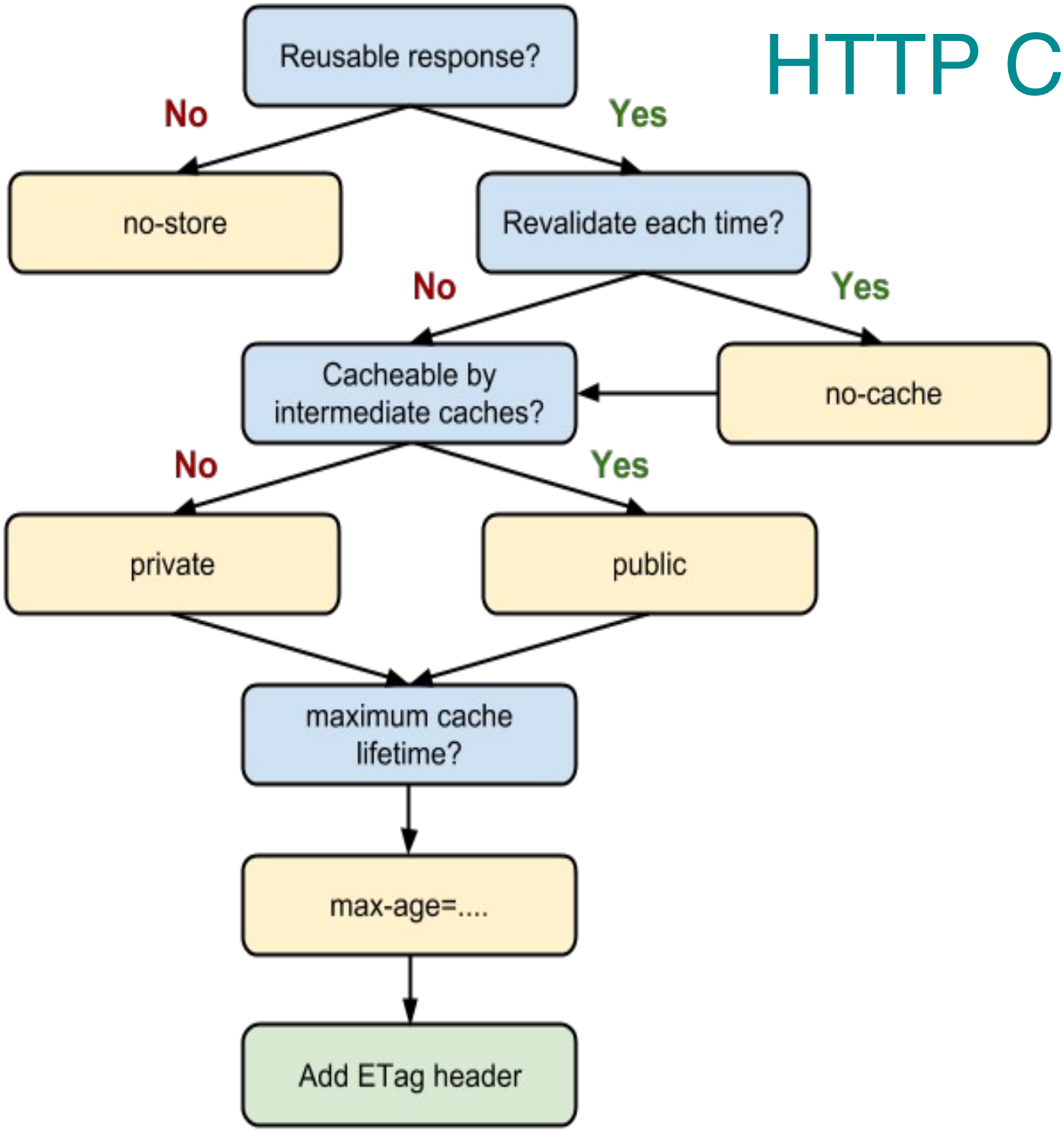
→ Configure web server such that HTTP headers for caching are sent (“ETag”, “Cache-Control”, “max-age”, …)

If cache expired, browser will send ETag; if data not changed, HTTP status “304 Not Modified” returned

Choose different cache configurations for different types of data

If web server runs CMS, consider using a reverse proxy (squid, …) to take load away from server

HTTP Caching-Policy



Optimizing Content Efficiency (1)

Common website: 13 KB HTML, 528 KB images, 207 KB JS, 24 KB CSS, 282 KB other → > 1 MB

⇨ Eliminate unnecessary downloads

⇨ Optimize encoding / transfer size:

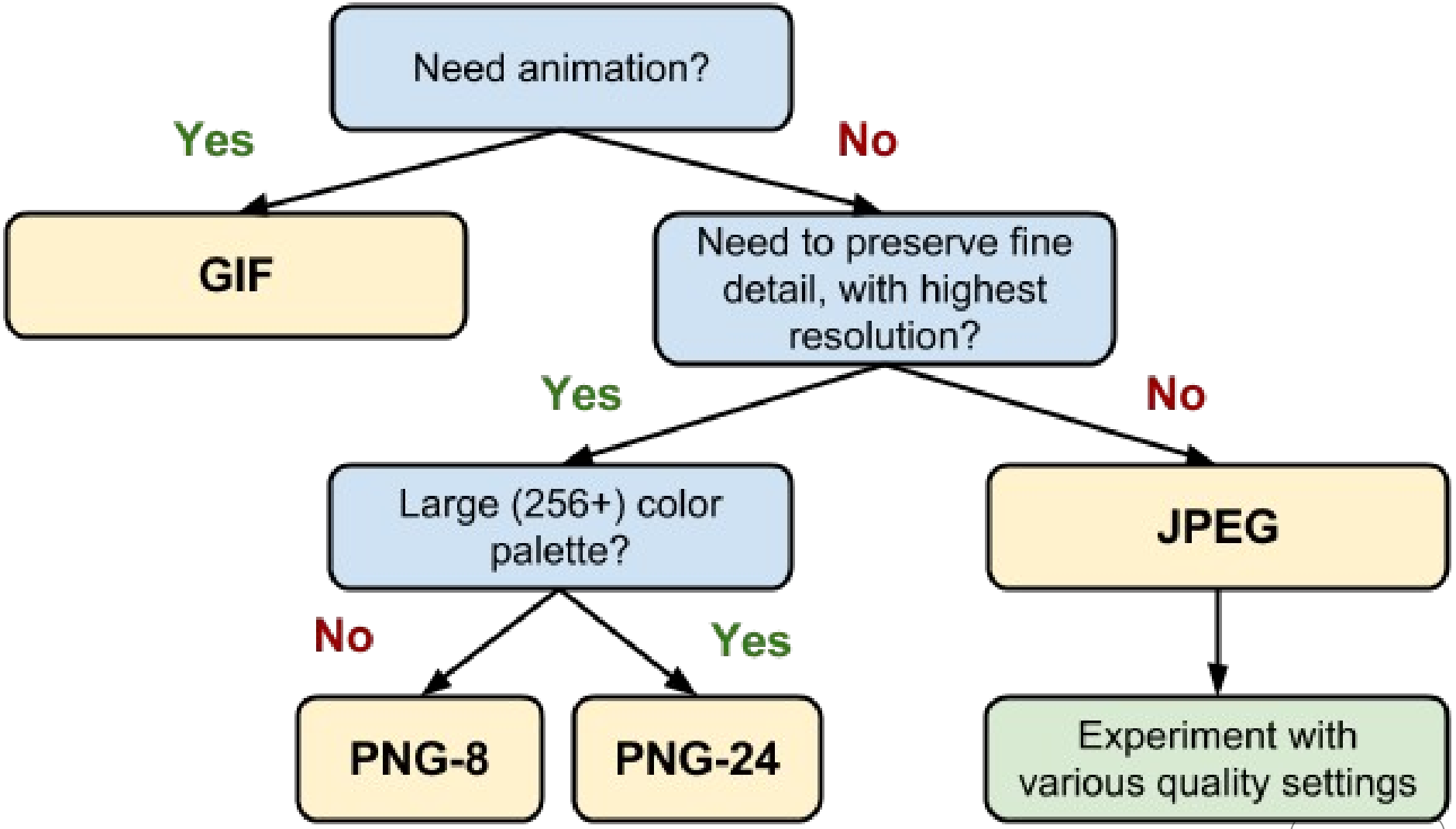
- Image optimization
- Minification of JS and CSS
- Text compression with gzip (server config!!)

Optimizing Content Efficiency (2)

Image optimization:

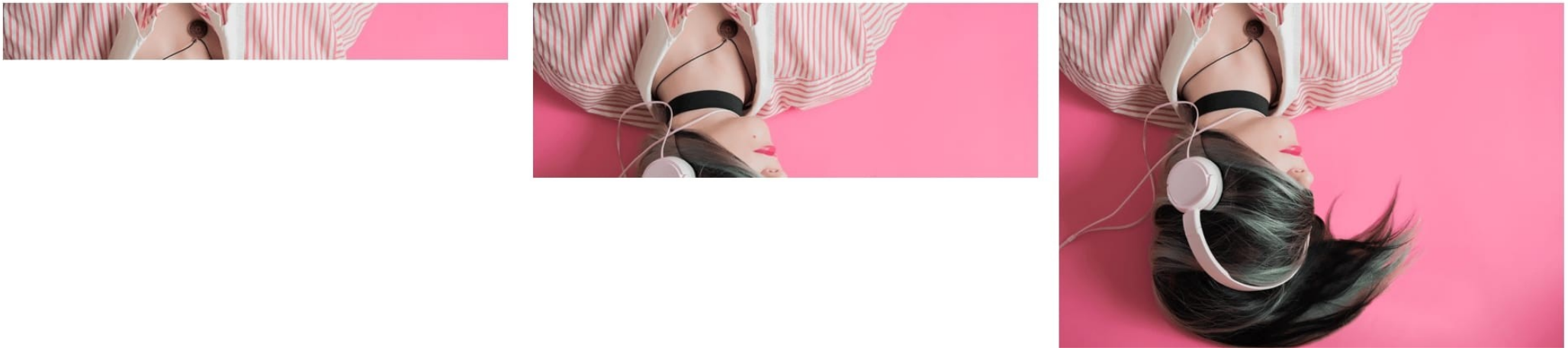
- Use CSS 3 techniques where possible
- Consider using vector graphics (SVG)
- Choose correct image size (no HTML scaling)
- Consider lossy compression (JPEG, WebP)
- Web font optimization
- Use HTTP caching

Choosing the Right Image Format



Progressive JPEG

Baseline JPEG - Loads from top-to-bottom



Progressive JPEG - Loads from low-quality to high-quality

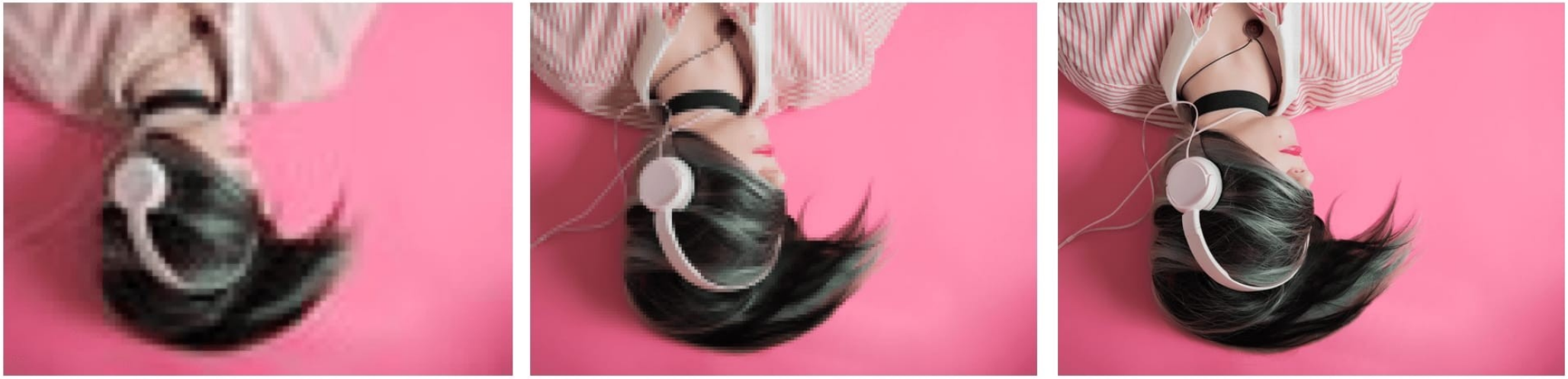
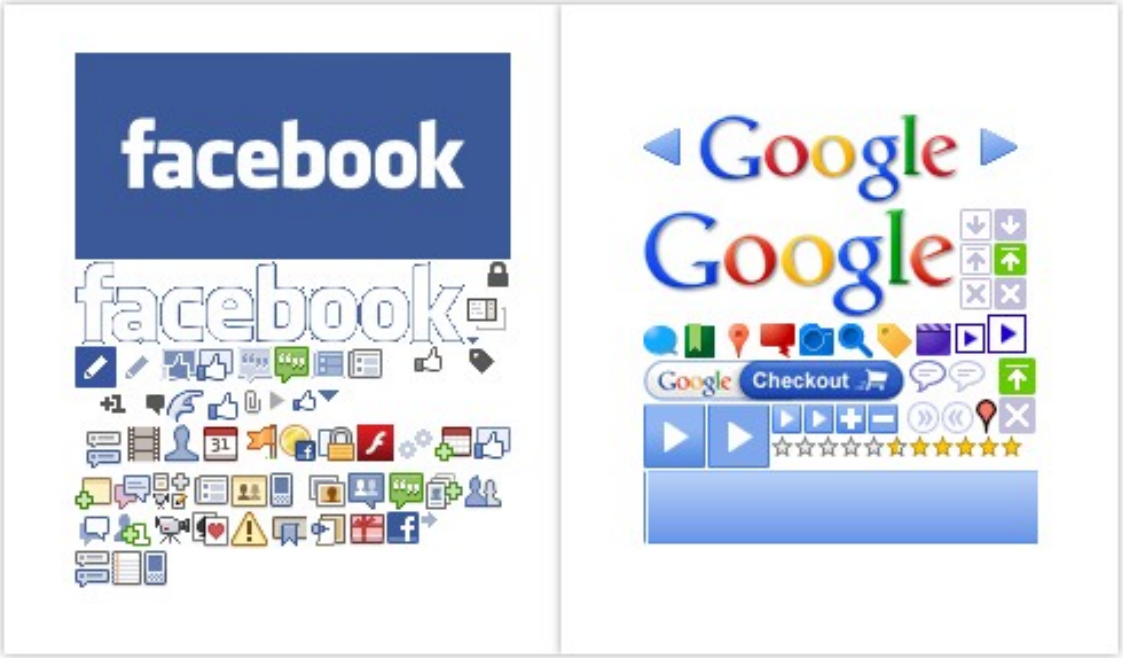


Image Sprites

Collection of images put into a single image

Selection of partial image done with CSS

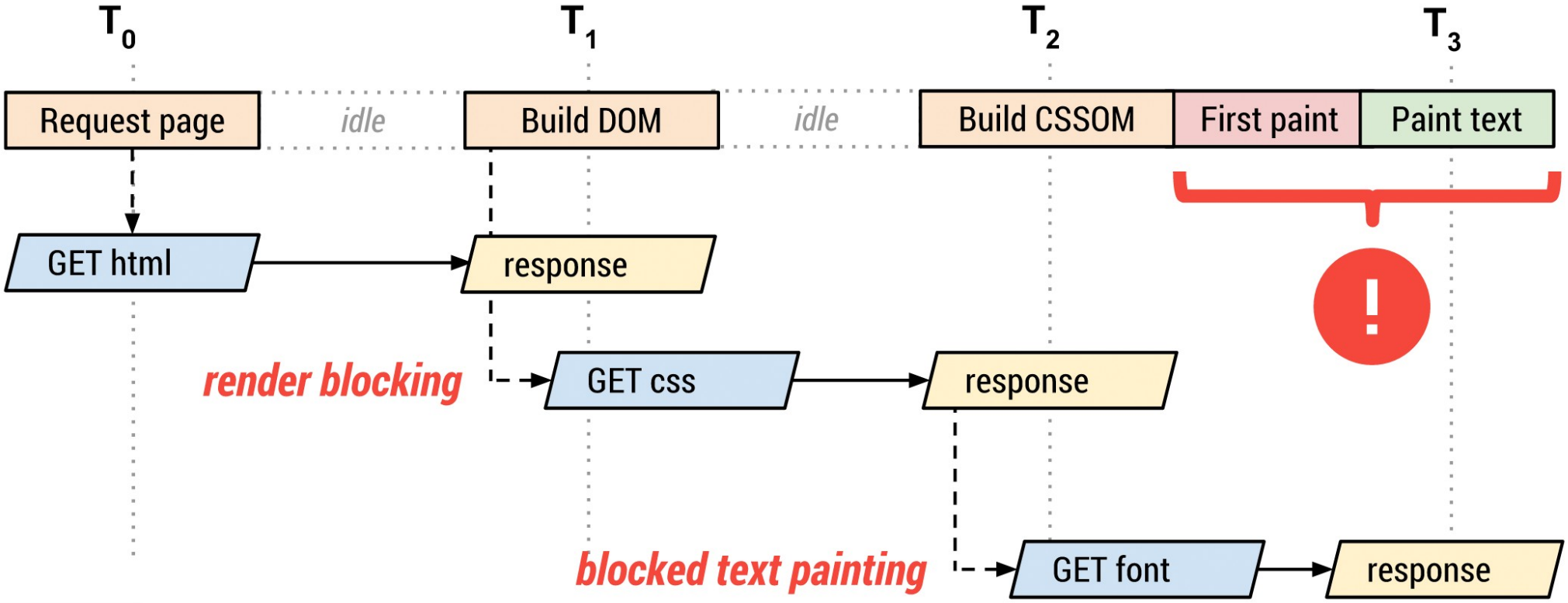
Generation tools exist



Optimizing Web Fonts

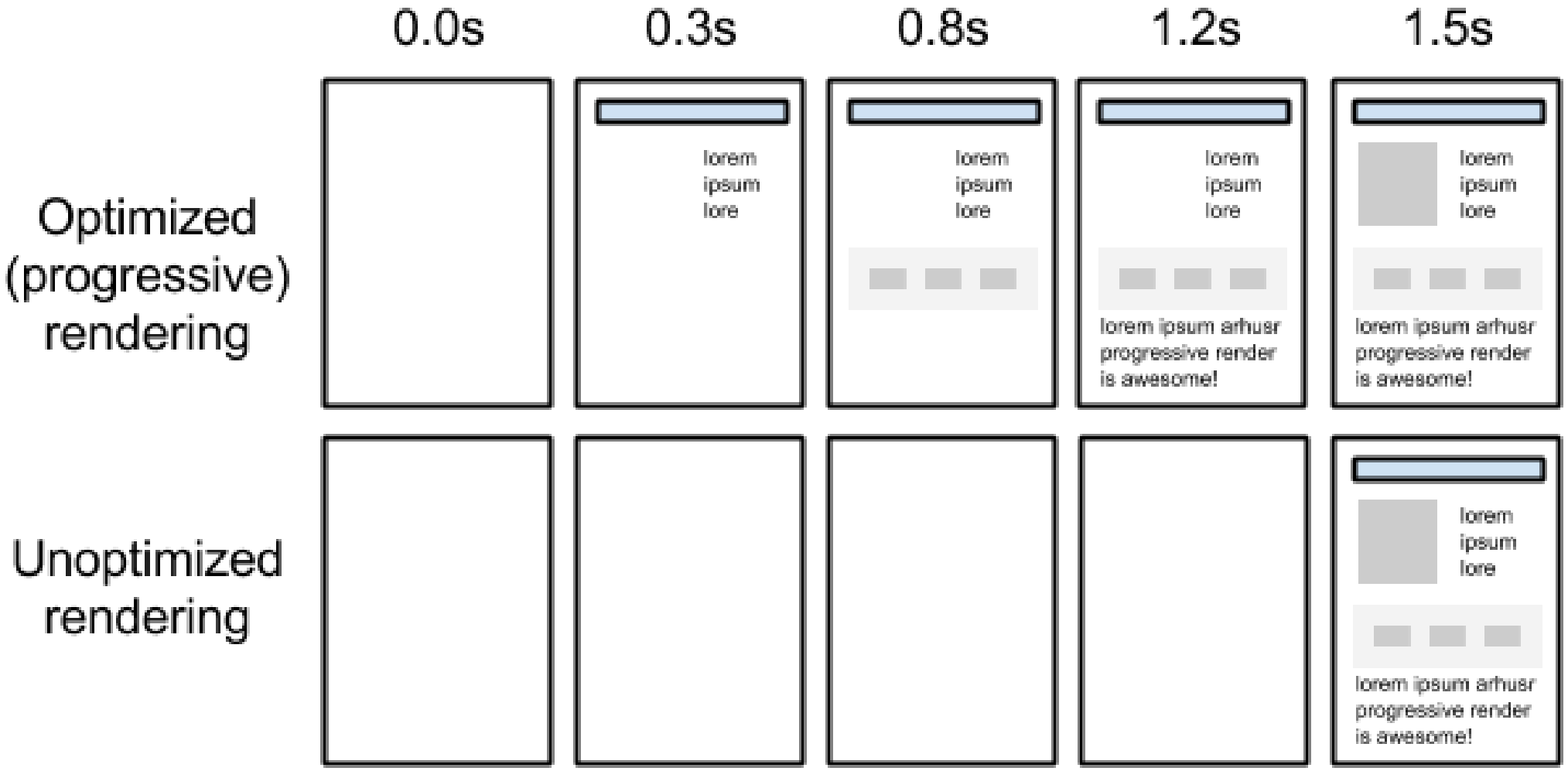
Define font family with @font-face

Define Unicode-subrange



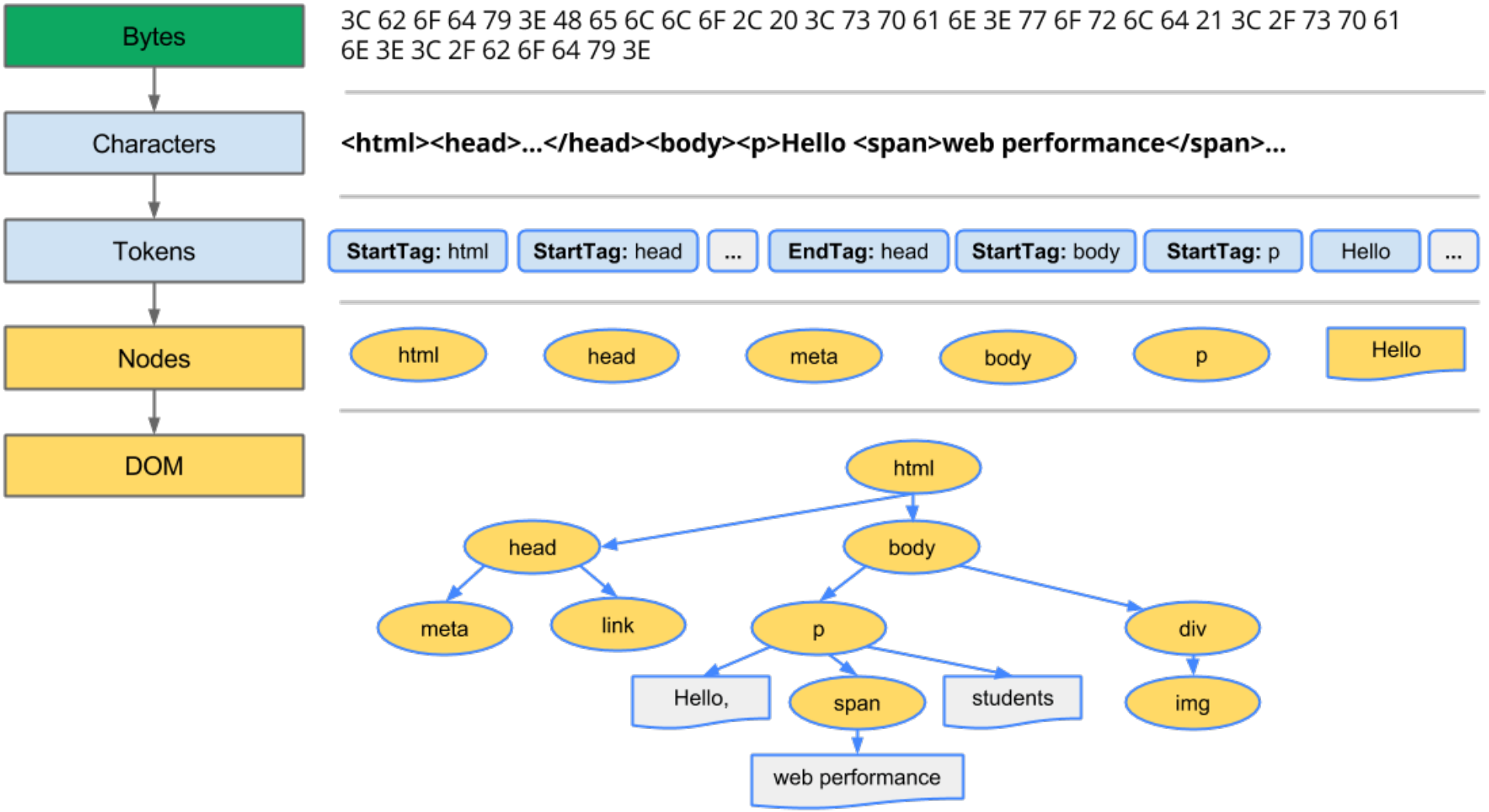
→ Use JS or inline css to fetch web fonts early

Critical Path



HTML, CSS and JS are potentially blocking rendering

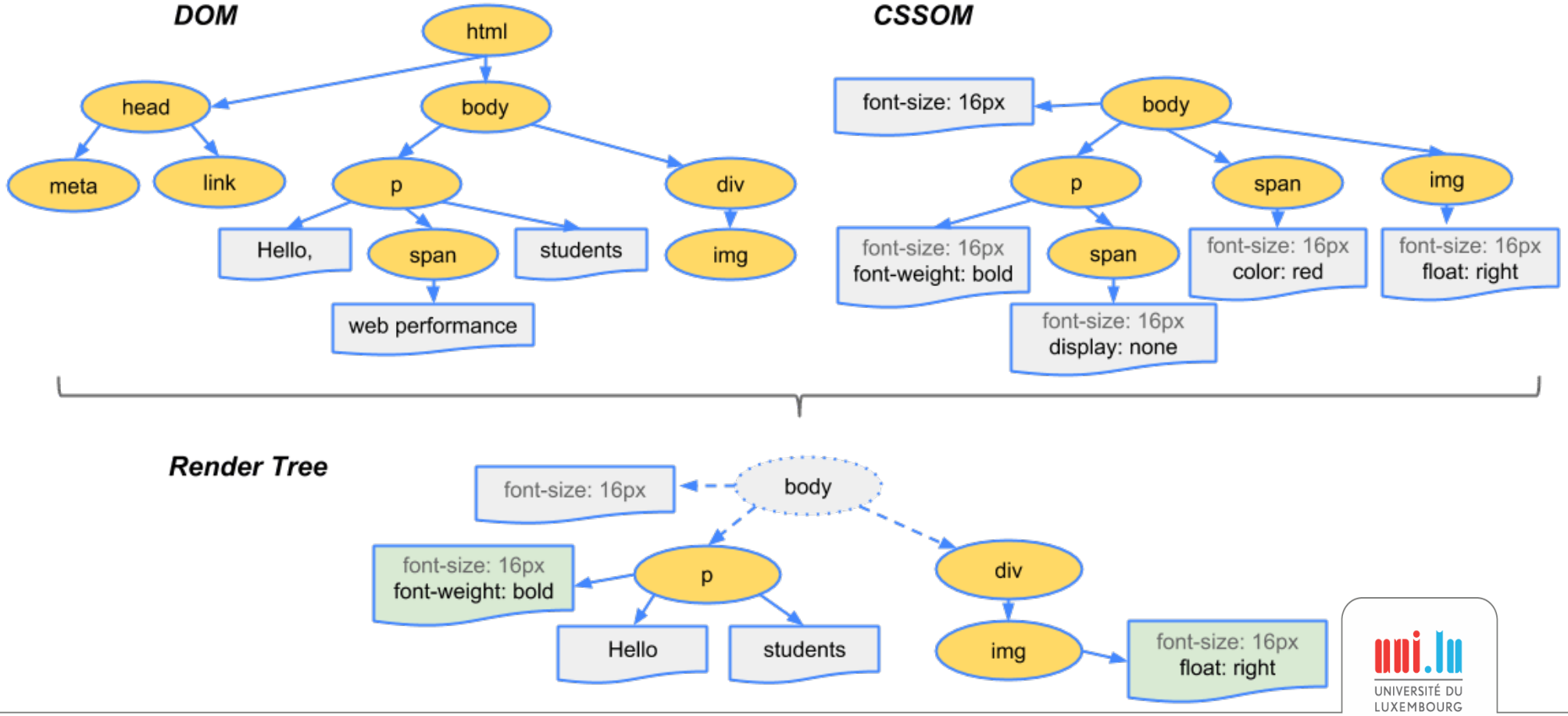
Constructing the DOM



Render Tree

Similar to DOM, CSSOM constructed

Both are merged into a Render Tree



Implications

CSS is a render blocking resource

→ use media queries where possible: <link href="other.css" rel="stylesheet" media="(min-width: 40em)">

JavaScript can query and modify the DOM and the CSSOM

→ JavaScript execution blocks on the CSSOM

→ JavaScript blocks DOM construction unless explicitly declared as “async”

CSS Suggestions

Put CSS in header

Avoid css imports (to minimize HTTP requests)

Small css code can be inlined in header, but don't inline in HTML elements

Use the media attribute in <link> such that CSS is only loaded under certain conditions

Minify css: remove white spaces, duplicate entries, unneeded code → Tools: csstidy, many IDEs, http://www.cleancss.com/

JS Suggestions

Minify and cleanse JS files (remember webpack)

Put JS scripts at bottom of page if not needed for rendering

Prefer <script> to inline JS (allows caching !!)

<script> has “async” attribute if script does not interfere with DOM (similarly: attribute “defer”)

Make Ajax cacheable (with HTTP headers!)

HTML / PHP Tricks

Remove unneeded or empty tags (“tidy” : tool for clean up)

Remove white space where possible

Avoid empty src attribute in image tags

Consider prefetching (using browser idle time):

```
<link rel = "dns-prefetch" href = "https://api.twitter.com" />

<link rel = "prefetch" href="…" />
```

PHP: Flush the buffer as early as possible

HTML Tricks (2)

Minimize number of iframes

Reduce cookie size and # of cookies

Use cookie-free domains for static resources

Consider post-load components which are not immediately needed for rendering (using existing JS code: YUI Imageloader, jQuery load function)

Pre-load components (take advantage of idle time of browser), request components (like images, styles and scripts) you'll need in future

Rendering performance

Optimize JavaScript execution:

- Use efficient algorithms
- Use “precise” selectors

Reduce scope and complexity of style calculations

Avoid large, complex layouts and layout thrashing

Another Type of Optimization

Search engine optimization (SEO) = affecting the visibility of website / web page in a search engine's unpaid results

Many documents on tricks exist on the web, for different search engines

Conclusion

Only a few common suggestions given, more resources available on the web (especially at Google developer site)

In the end, the skills of the programmer to write efficient and correct code also plays an important role

Next Week

Security of Web Applications