

A thorough analysis of

CSS *in* **JS**



ANDREI PFEIFFER

Timișoara / RO

[e-spres-oh]

Code Designer



Co-Organizer



@pfeiffer_andrei

2+ months study

2+ months study

in 40 minutes

***For more context, details, or
uncovered topics***

*For more context, details, or
uncovered topics*

Please post your questions

MOTIVATION

CSS *isn't trivial to* **scale**

Best practices

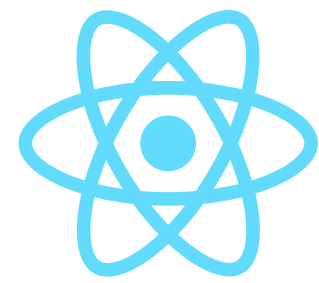
are way too complex to teach
and impossible to enforce

Methodologies aren't trivial to scale

OOCSS, BEM, SMACCS

CSS encapsulation

is a great step forward



CSS Modules
(optional)



**Emulated
Scoped CSS**



**Emulated / ShadowDOM
Encapsulation**

CSS encapsulation
is not enough

CSS *is not type-safe*

Explore type-safe CSS

*Type-checking, Goto definition,
Safe refactoring, Unused code detection,
Typed design tokens*

CSS *in* **JS**

CSS *in* **TS**

DISCLAIMERS

*I have not built
my own CSS-in-JS library*

***I have no motivation to
promote or trash either of them***

***I have no prior experience
with CSS-in-JS***

I've equally used all libraries

... but have no extensive experience

***This analysis is a pursuit towards
better understanding***

Based on limited know-how

Research, Experimentation & Discussions with maintainers

OVERVIEW

SSR (Server-Side Rendering)

Easy integration with **Next.js**

TypeScript support



Cxs



Emotion

Styled JSX



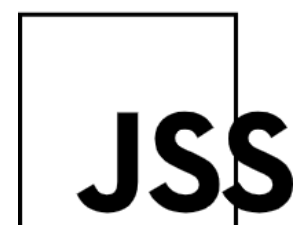
style9

Styled Components

Glamor



Stitches



glamorous

Compiled

TypeStyle

Aphrodite



Astroturf



Treat

Radium



Cxs



Emotion

Styled JSX



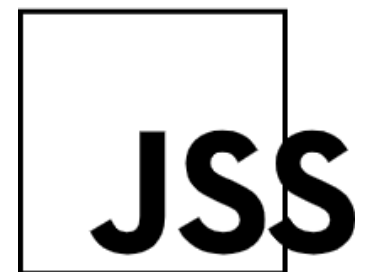
style9

Styled Components

Glamor



Stitches



Compiled

Aphrodite

TypeStyle



Treat

Radium



Astroturf



Styled JSX



TypeStyle



Compiled

LIBRARIES

FEATURES



COMMON



AMBIGUOUS



DISTINCT



COMMON

FEATURES



SSR

Server-Side Rendering



No inline styles



Styles encapsulation

Uniquely generated CSS class names



Global styles



Full CSS syntax support

Pseudos, Keyframe animations, Media queries



AMBIGUOUS

FEATURES



Dead code removal

Dead code removal



Works at component level

- Removing component



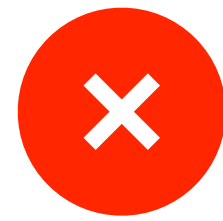
Doesn't work at CSS rule level

- Nesting: `"& span"`
- Pseudos: `"&:first-child"`
- Parents: `".parent &"`
- Dynamic: `` .color-${active} ``

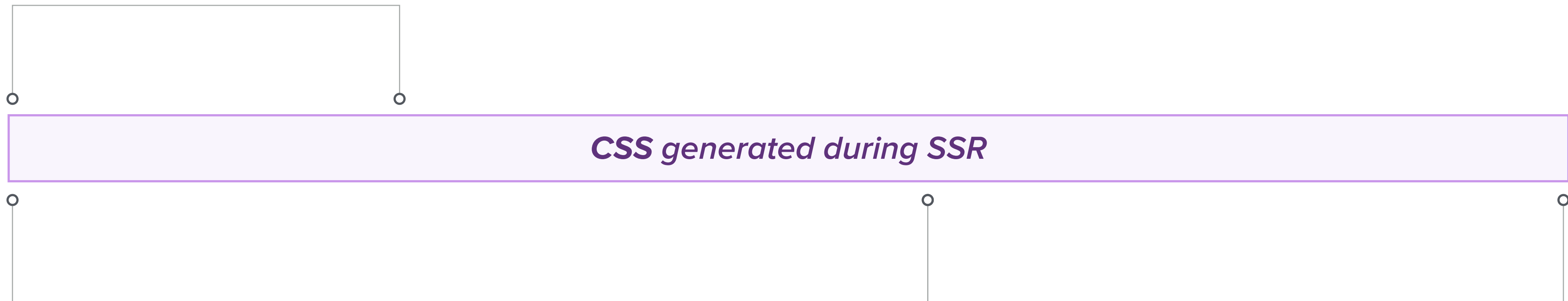


Critical CSS extraction

Critical CSS extraction



Above the fold CSS



CSS required for initial render

Removes dynamic styles





DISTINCT

FEATURES



Styles definition syntax

Styles definition syntax

Tagged Templates

```
const title = css`  
  font-size: 2rem;  
  border-color: ${COLOR_BLUE};  
`;
```

Object Styles

```
const title = css({  
  fontSize: "2rem",  
  borderColor: COLOR_BLUE,  
});
```

Tagged Templates

```
const title = css`  
  font-size: 2rem;  
  border-color: ${COLOR_BLUE};  
`;
```

Syntax highlight & Code suggestions

Requires code editor/IDE plugin



Easier migration from plain CSS

Object Styles




























```
const title = css({  
  fontSize: "2rem",  
  borderColor: COLOR_BLUE,  
});
```

Syntax highlight: out-of-the-box

Code suggestions: via @types



Simpler & lighter

	<i>Styled JSX</i>	 SC	 Emotion	 Treat	TypeStyle	 fela	 Stitches	 JSS	 goober	Compiled
<i>Tagged Templates</i>										
<i>Object Styles</i>										



*Full out-of-the-box
support*



*Optional with
separate plugin*



*Lack of any
support*



Styles output

Styles output

Runtime stylesheets

// styles get bundled with the components
`<script src="bundle.js"></script>`

// injects styles to DOM
`<script src="library_runtime.js"></script>`

Styles output

Static CSS extraction

```
// styles extracted as static .css files  
<link rel="stylesheet" href="styles.css" />
```

```
// includes the components  
<script src="bundle.js"></script>
```

PERFORMANCE

HTML

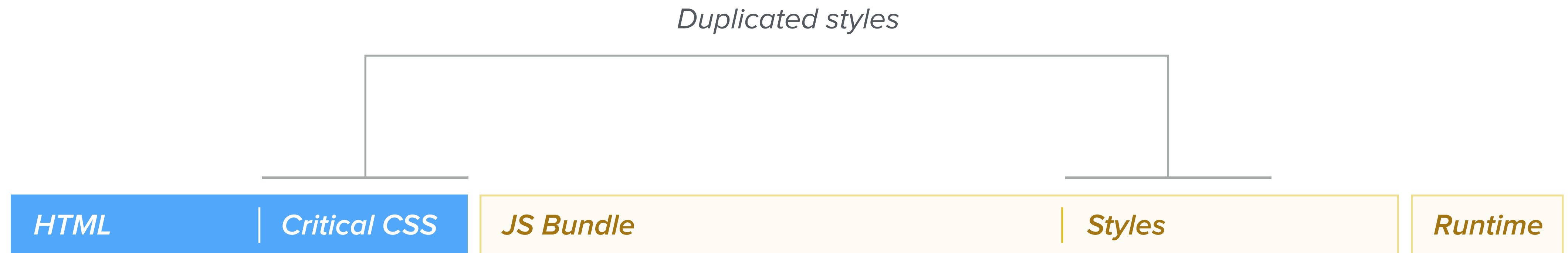


JS Bundle

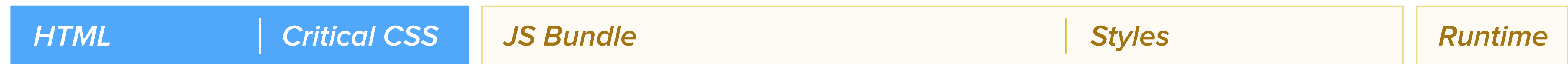
Styles

Runtime

CSR (Client-Side Rendering) **with Runtime stylesheets**



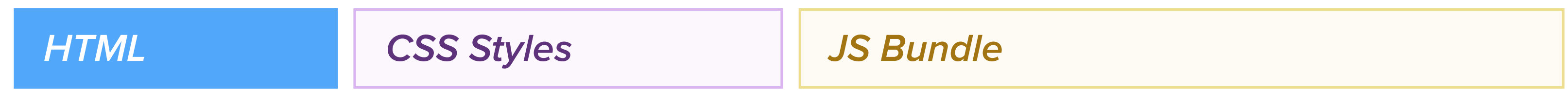
SSR (Server-Side Rendering) **with Runtime stylesheets**



File size

Runtime stylesheets

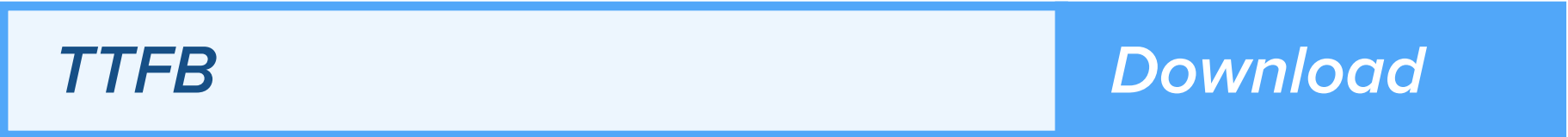
Static CSS extraction

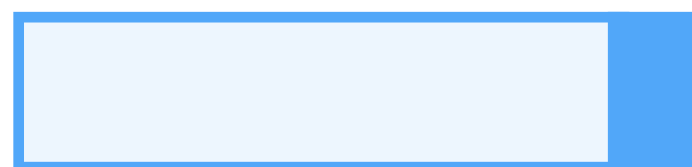


Difficult to cache

Easy to cache

HTTP Request





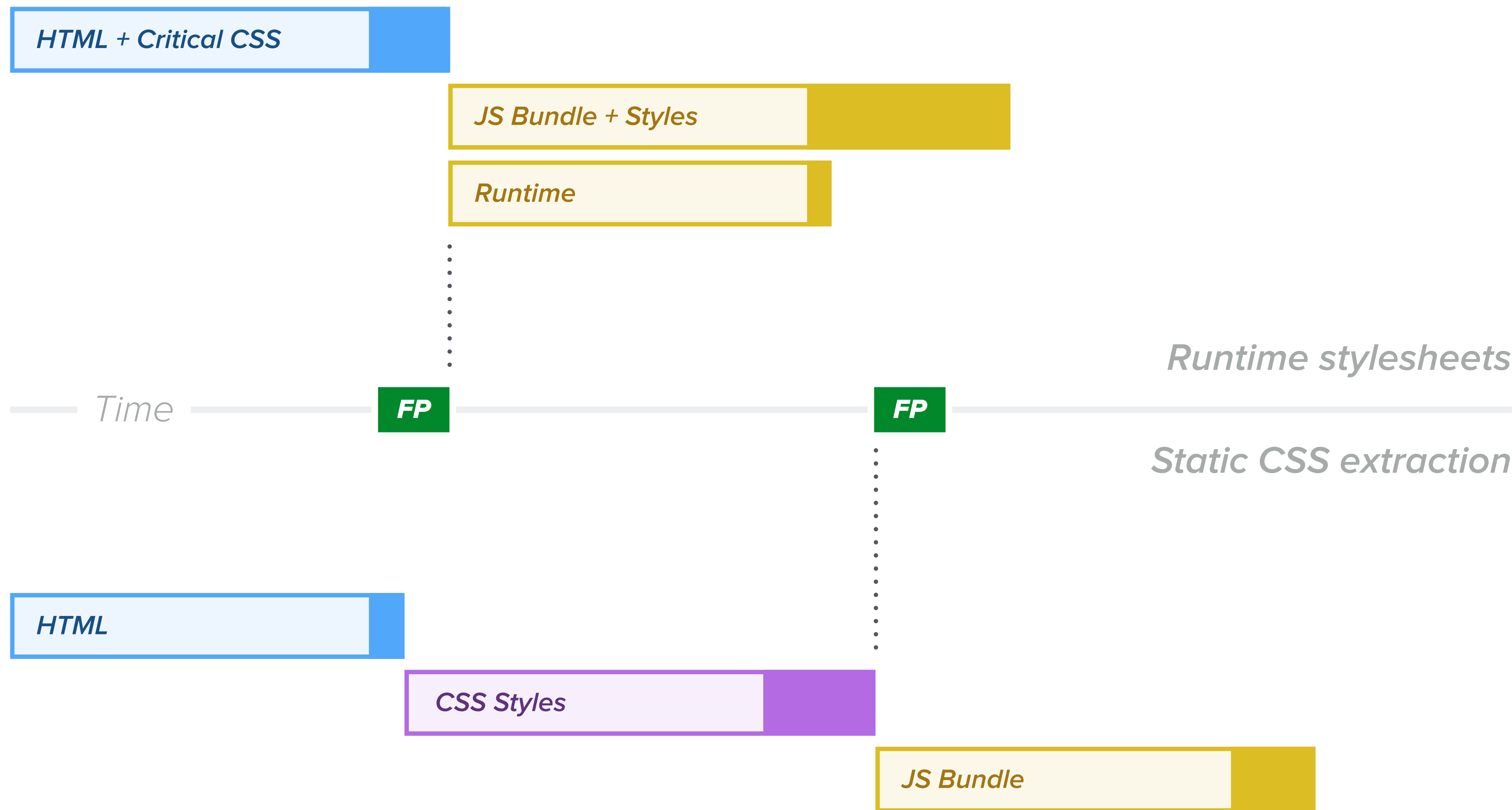
Fast Wi-Fi

Time

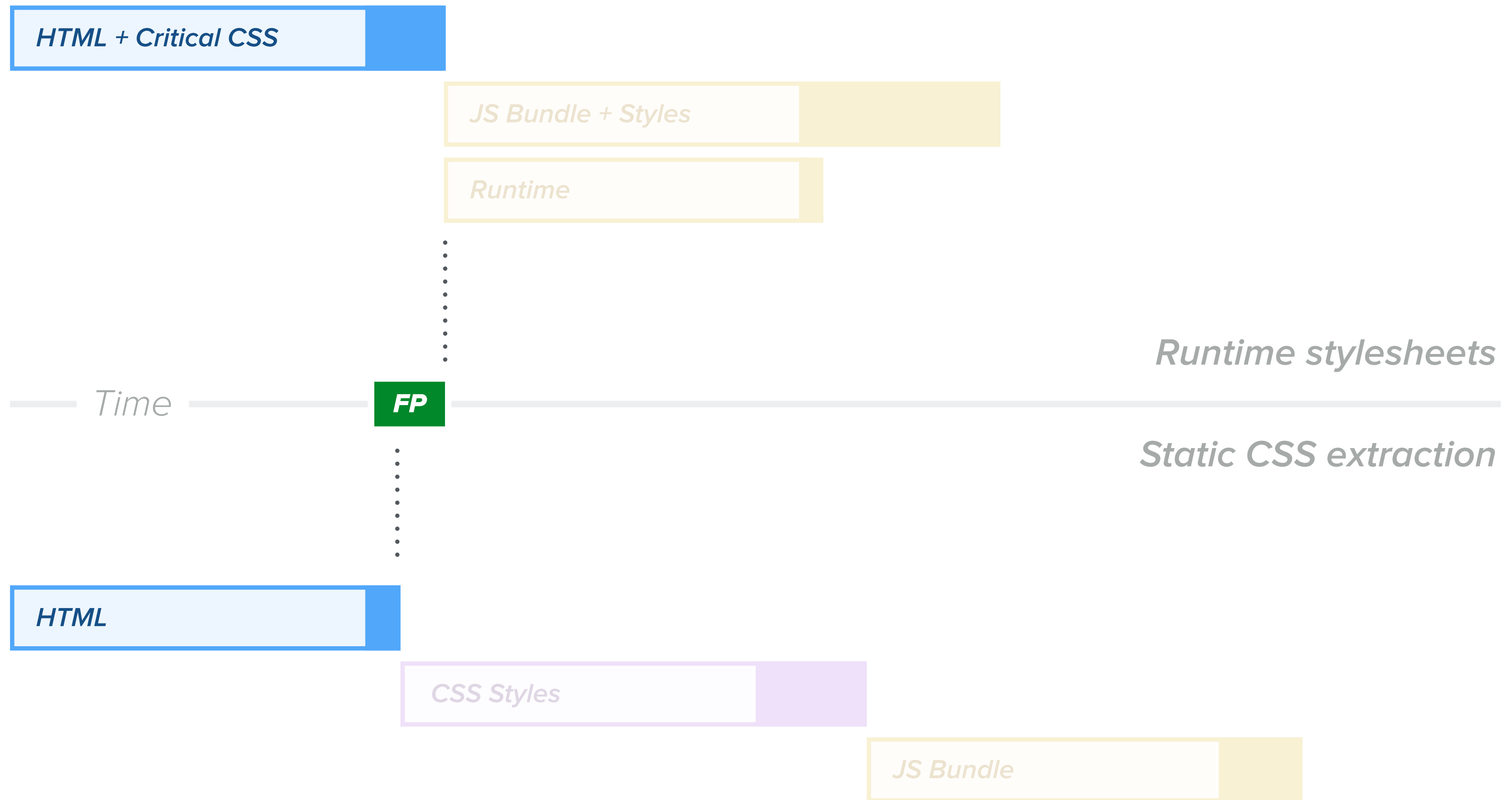
Slow mobile



EMPTY CACHE



FULL CACHE



HTML + Critical CSS

JS Bundle + Stylesheets

Runtime stylesheets

Static CSS extraction

Static CSS extraction

Static CSS extraction

JS Bundle

JS Bundle

Styles output

Runtime stylesheets




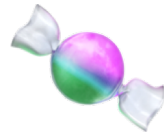


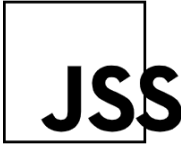





















*Looks more suitable for **CSR / SPA***

*Faster **First Paint** metrics*

Static CSS extraction


*Looks more suitable for **SSR***

Less bytes** transferred, better **caching

	 <i>Styled JSX</i>	 <i>SC</i>	 <i>Emotion</i>	 <i>Treat</i>	<i>TypeStyle</i>	 <i>fela</i>	 <i>Stitches</i>	 <i>JSS</i>	 <i>goober</i>	<i>Compiled</i>
<i>Static CSS extraction</i>										
<i>Runtime stylesheets</i>										


Supported


Experimental


Not supported

Understanding our tools

helps us make

better educated decisions

Checkout the full analysis

github.com/andreipfeiffer/css-in-js

THANK **YOU**



andreipfeiffer.dev