PLARIUM

tuchk4

CSS in JS

ВАЛЕРИЙ Сорокобатько

v.sorokobatko@gmail.com

PLARIUM

tuchk4

ВАЛЕРИЙ СОРОКОБАТЬКО

- **Create-react-app** контрибьютор
- автор **Awesome-create-react-app**
- автор **React Storybook README Addon**

PLARIUM

CSS in JS

PLARIUM

ПЛАН

- В чем польза?
- Существующие решения
- Требования к библиотеке
- Что используем мы
- Нюансы в реализации CSS in JS подхода

PLARIUM

В чем польза?

PLARIUM

Удобство разработки

Удобство разработки

Andrey Okonetchnikov
@okonetchnikov

*“Investing into **#DX** you’ll get a better **#UX** in the result”*

PLARIUM

Удобство разработки

Не нужно думать о подключении CSS файлов

Достаточно подключить только JS модуль

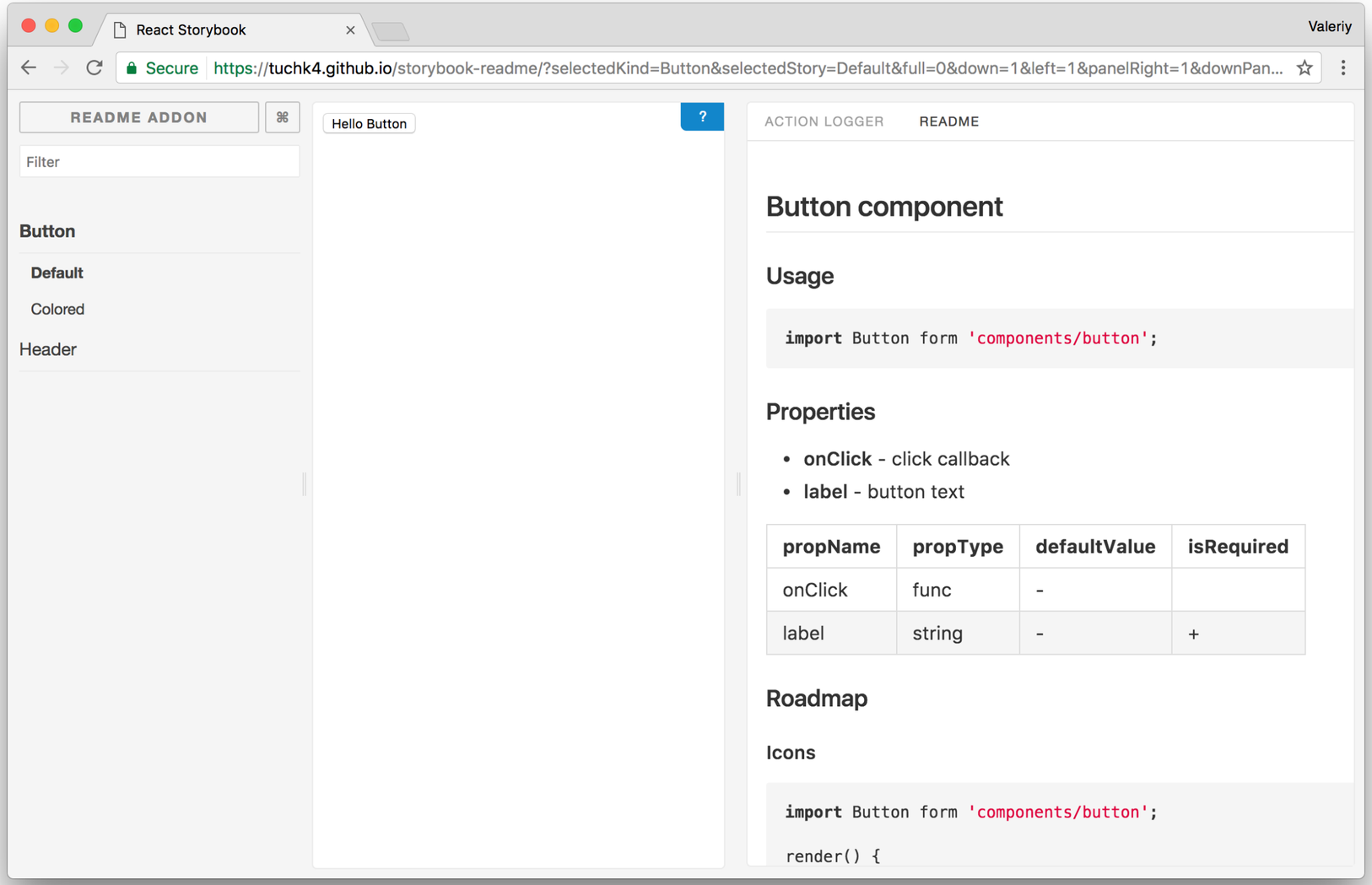
PLARIUM

Удобство разработки

Особенно удобно при шаринге коде

PLARIUM

Удобство разработки



PLARIUM

Удобство разработки

Более мощный CSS

PLARIUM

Удобство разработки

Более мощный CSS

```
import color from 'color'
const red = Color('red')

const styles = {
  color: red.lighten(10).toHex()
}
```

PLARIUM

Удобство разработки

Более мощный CSS

```
const styles = {
  width: window.innerWidth
}
```

PLARIUM

Удобство разработки

Более мощный CSS

```
.col-1 .col-2 .col-3 .col-4 ...
.col-md-1 .col-md-2 .col-md-3 .col-md-4 ...
.col-xs-1 .col-xs-2 .col-xs-3 .col-xs-4 ...
```

PLARIUM

Удобство разработки

Более мощный CSS

```
.col-1 .col-2 .col-3 .col-4 ...
.col-md-1 .col-md-2 .col-md-3 .col-md-4 ...
.col-xs-1 .col-xs-2 .col-xs-3 .col-xs-4 ...
```

```
const css = props => ({
  width: `12 / ${props.size}%`
})
```

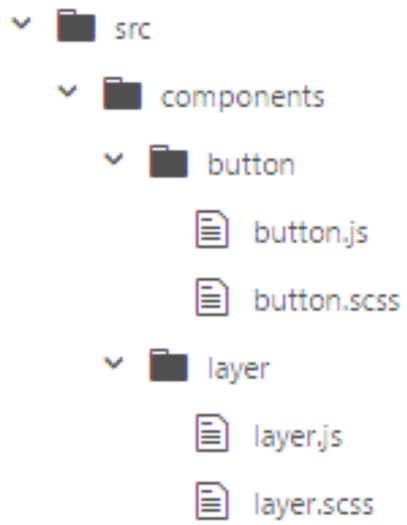
PLARIUM

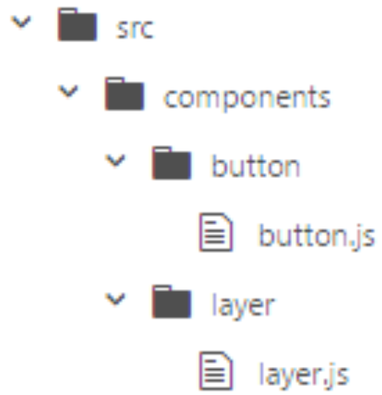
Удобство разработки

Меньше код и файлов

Удобство разработки

Меньше код и файлов





PLARIUM

Удобство разработки

Не повторяйте себя
(DRY)

PLARIUM

Удобство разработки

Не повторяйте себя

```
const ANIMATION_DURATION = 2.5

const styles = {
  animation: `${ANIMATION_DURATION}s pulse`
};
```

PLARIUM

Удобство разработки

Не повторяйте себя

```
const ANIMATION_DURATION = 2.5

const styles = {
  animation: `${ANIMATION_DURATION}s pulse`
};

mountElement()
setTimeout(() => {
   unmountElement()
}, ANIMATION_DURATION * 1000)
```

PLARIUM

Удобство разработки

Чистый код

PLARIUM

Удобство разработки

~~Чистый код~~

```
import styles from './button.css'
import Button from './button'

const MyButton = ({ primary, raised, large, children, ...props }) => {
  const classList = classnames(styles.button, {
    [styles.primary]: primary,
    [styles.large]: large,
    [styles.raised]: raised,
  })

  return <Button className={classList} {...props}>{children}</Button>
}
```

PLARIUM

Удобство разработки

Чистый код

```
import Button from './button'

const MyButton = injectStyles({
  color: props => props.primary ? 'blue' : 'black',
  fontSize: props => props.large ? '2em' : '1em',
  boxShadow: props => props.raised ? '1px 1px #ccc' : 'none',
})(Button)
```

Удобство разработки

Без конфигов

PLARIUM

Удобство разработки

Без конфигов

CSS in JS – это нативный JS

PLARIUM

Удобство разработки

Без конфигов

CSS in JS – это нативный JS

- Не нужны лоадеры и плагины
- Не нужно настраивать webpack и babel
- Проще запускать тесты

PLARIUM

Удобство разработки

lodash для CSS in JS

PLARIUM

Inline styles или CSS?

PLARIUM

Проблемы inline styles

```
<div style="padding-bottom: 8px;">
▼<div style="padding: 16px; font-weight: 500; box-sizing: border-box; position: relative; white-space: nowrap;
cursor: pointer;">
  ▼<div style="display: inline-block; vertical-align: top; white-space: normal; padding-right: 90px;">
    ▶<span style="color: rgba(0, 0, 0, 0.87); display: block; font-size: 15px;">…</span>
    ▶<span style="color: rgba(0, 0, 0, 0.54); display: block; font-size: 14px;">…</span>
    </div>
  ▼<button tabindex="0" type="button" style="border: 10px; box-sizing: border-box; display: inline-block; font-
  family: Roboto, sans-serif; -webkit-tap-highlight-color: rgba(0, 0, 0, 0); cursor: pointer; text-decoration:
  none; margin: auto; padding: 12px; outline: none; font-size: 0px; font-weight: inherit; position: absolute; z-
  index: 1; overflow: visible; transition: all 450ms cubic-bezier(0.23, 1, 0.32, 1) 0ms; width: 48px; height:
  48px; top: 0px; bottom: 0px; right: 4px; background: none;">
    ▶<div>…</div>
    </button>
  </div>
▼<div style="padding: 16px; font-size: 14px; color: rgba(0, 0, 0, 0.87);">
  ▶<div>…</div>
  ▼<button tabindex="0" type="button" style="border: 10px; box-sizing: border-box; display: inline-block; font-
  family: Roboto, sans-serif; -webkit-tap-highlight-color: rgba(0, 0, 0, 0); cursor: pointer; text-decoration:
  none; margin: 0px; padding: 0px; outline: none; font-size: inherit; font-weight: inherit; position: relative;
  z-index: 1; height: 36px; line-height: 36px; min-width: 88px; color: rgba(0, 0, 0, 0.87); transition: all
  450ms cubic-bezier(0.23, 1, 0.32, 1) 0ms; border-radius: 2px; user-select: none; overflow: hidden; background-
  color: rgba(64, 168, 45, 0.21); text-align: center;">
```

PLARIUM

Проблемы inline styles

- Невозможно менять стили для всех компонентов одного типа

PLARIUM

Проблемы inline styles

- Невозможно менять стили для всех компонентов одного типа
- Нет стандартной поддержки псевдоклассов

PLARIUM

Проблемы inline styles

- Невозможно менять стили для всех компонентов одного типа
- Нет стандартной поддержки псевдоклассов
- Много «лишнего» кода

PLARIUM

Поговорим о CSS in JS

PLARIUM

Проблемы CSS in JS

PLARIUM

Проблемы CSS in JS

- Дополнительные операции в рантайме

PLARIUM

Проблемы CSS in JS

- Дополнительные операции в рантайме
- Плохая интеграция с IDE

PLARIUM

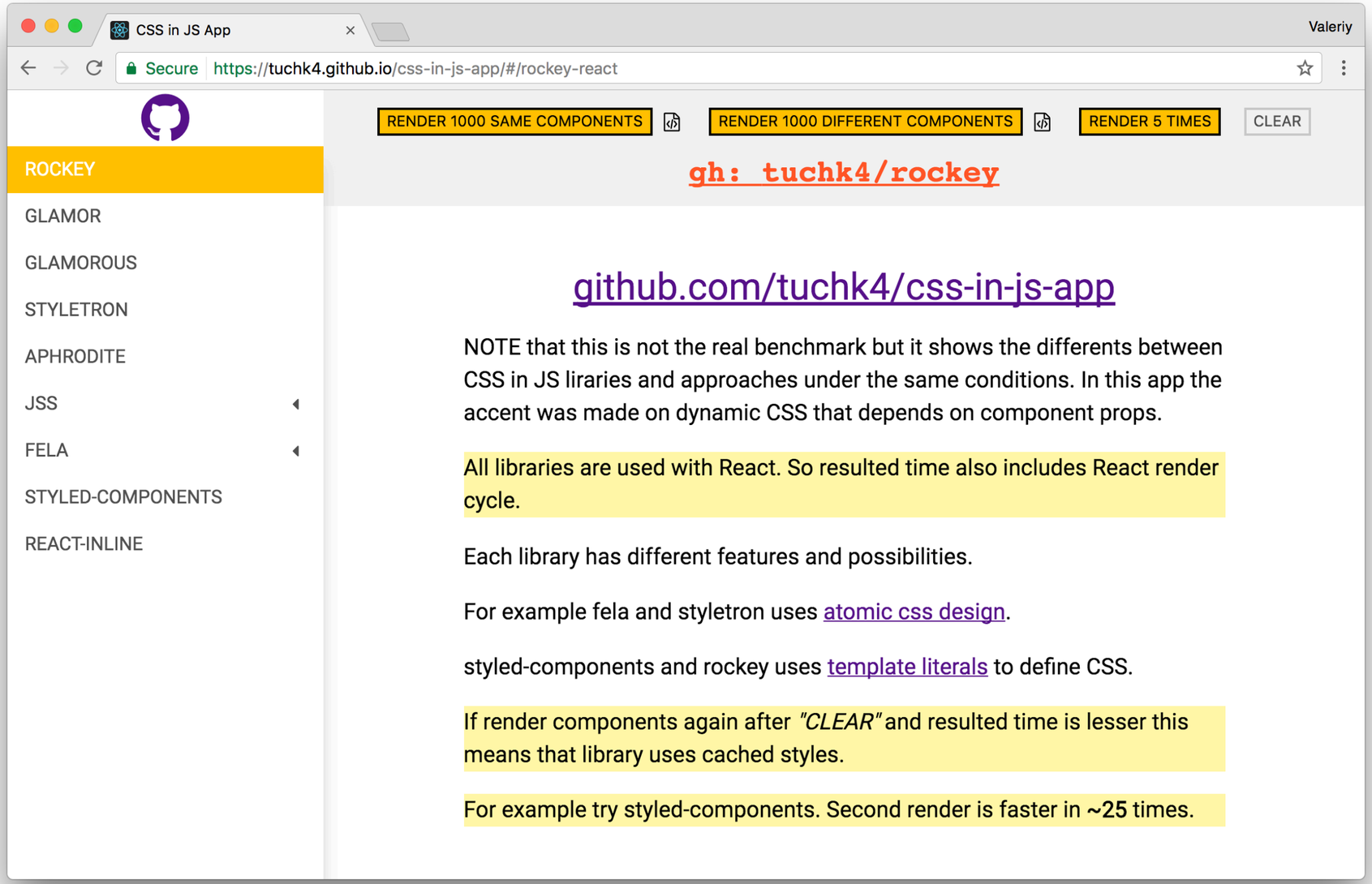
Проблемы CSS in JS

- Дополнительные операции в рантайме
- Плохая интеграция с IDE
- Дополнительная работа с DOM

Существующие решения

PLARIUM

Пример



PLARIUM

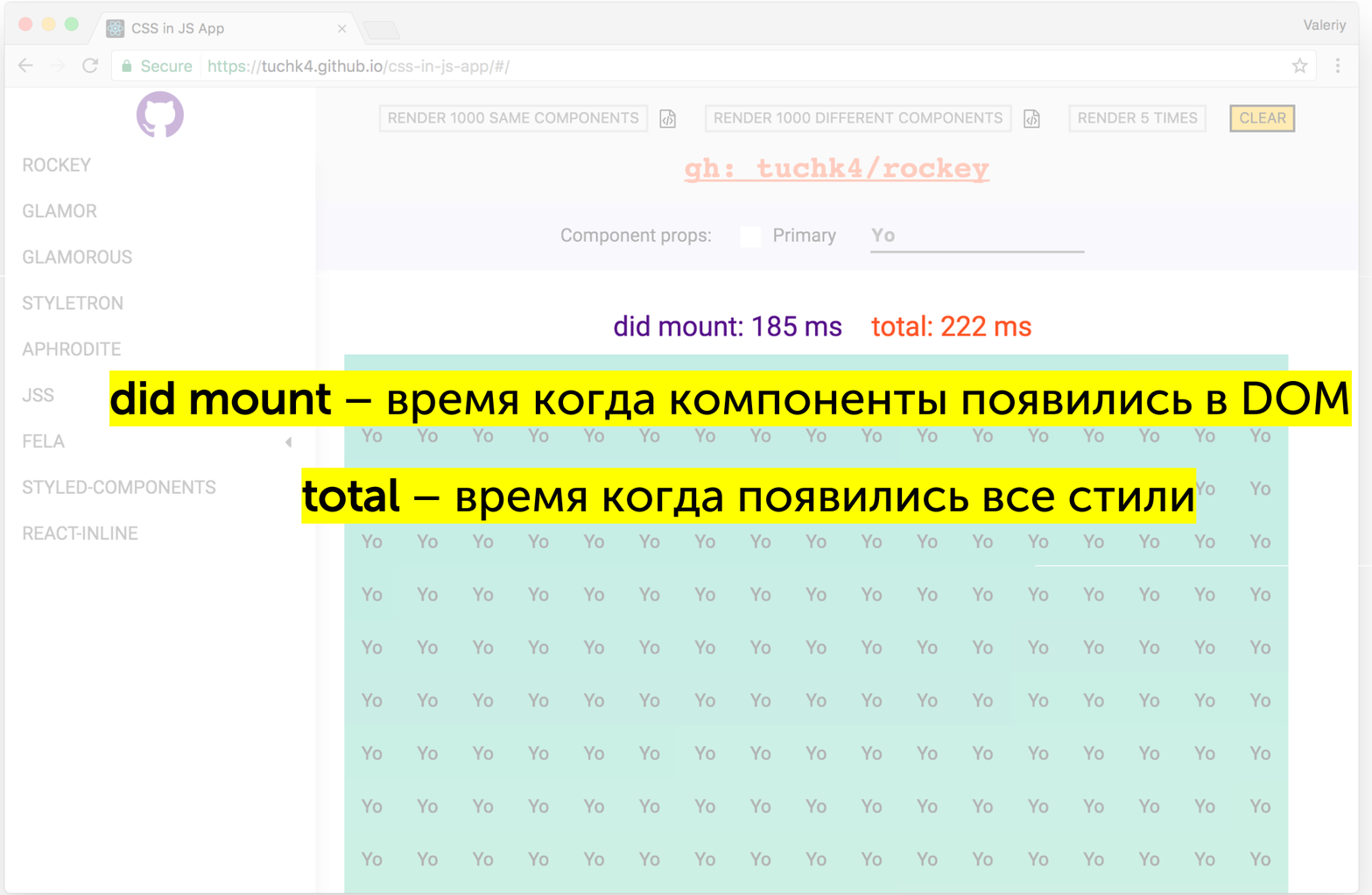
Пример

- Рендерит 4000 компонентов
- С акцентом на динамический CSS

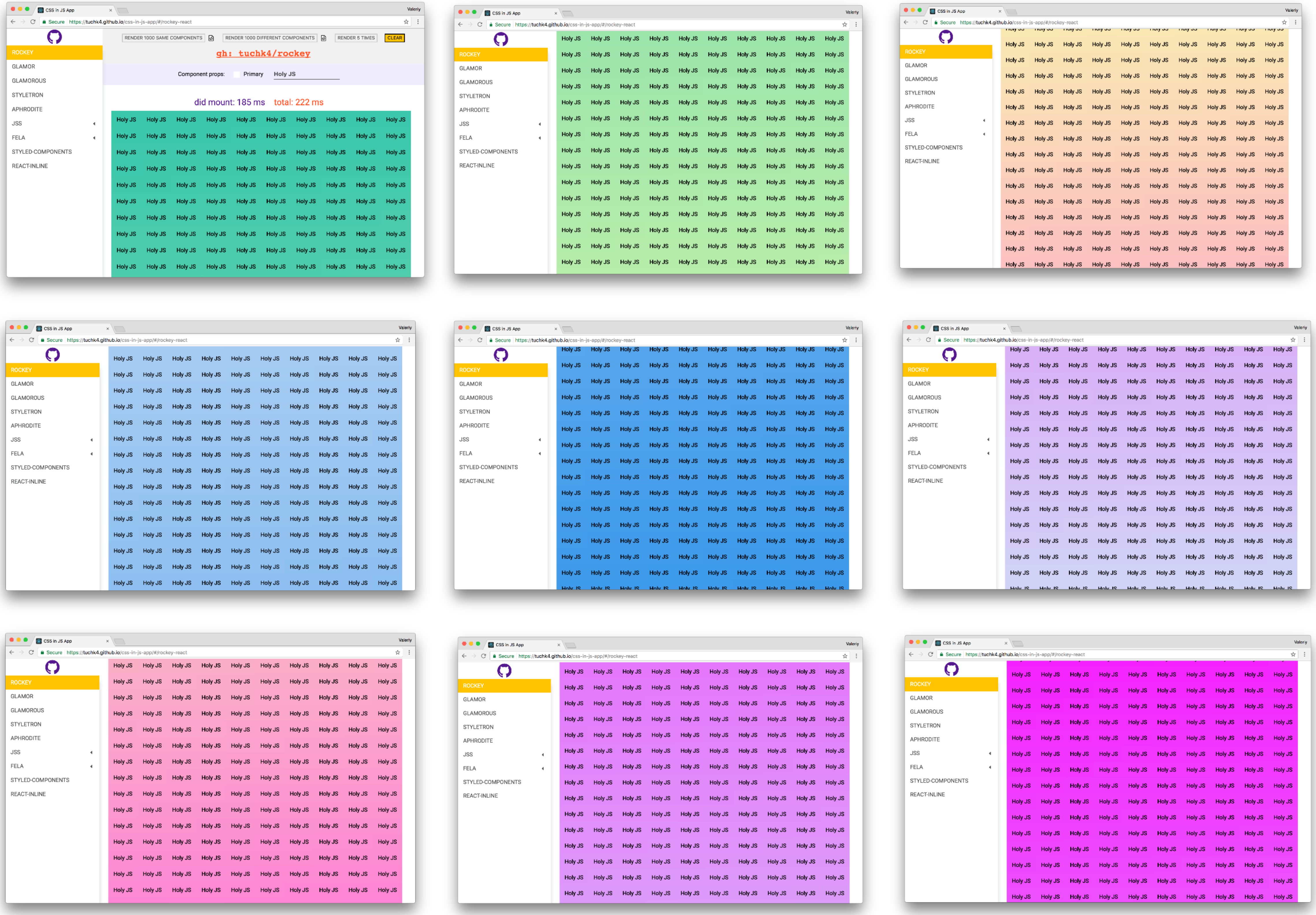
```
{
  minWidth: '16px',
  height: '16px',
  display: 'inline-block',
  backgroundColor: props => colors[props.i]
}
```

PLARIUM

Пример



Пример


gh: tuchk4/rockey
ROCKEY
GLAMOR
GLAMOROUS
STYLETRON
APHRODITE
JSS
FELA
STYLED-COMPONENTS
REACT-INLINE
did mount: 185 ms
total: 222 ms
Holy JS

PLARIUM

PLARIUM

JSS

JSS

```
const styles = {
  block: {
    minWidth: '16px',
    height: '16px',
    display: 'inline-block',
    backgroundColor: props => colors[props.i]
  }
}
```

PLARIUM

JSS

JSS

```
const styles = {
  block: {
    minWidth: '16px',
    height: '16px',
    display: 'inline-block',
    backgroundColor: props => colors[props.i]
  }
}
```

PLARIUM

JSS

JSS

```
const styles = {
  block: {
    minWidth: '16px',
    height: '16px',
    display: 'inline-block',
    backgroundColor: props => colors[props.i]
  }
}
```

PLARIUM

JSS

JSS

```
.block-0-1 {
    background-color: #dc2430;
}
```

```
.block-0-0 {
    height: 16px;
    display: inline-block;
    min-width: 16px;
}
```

PLARIUM

JSS

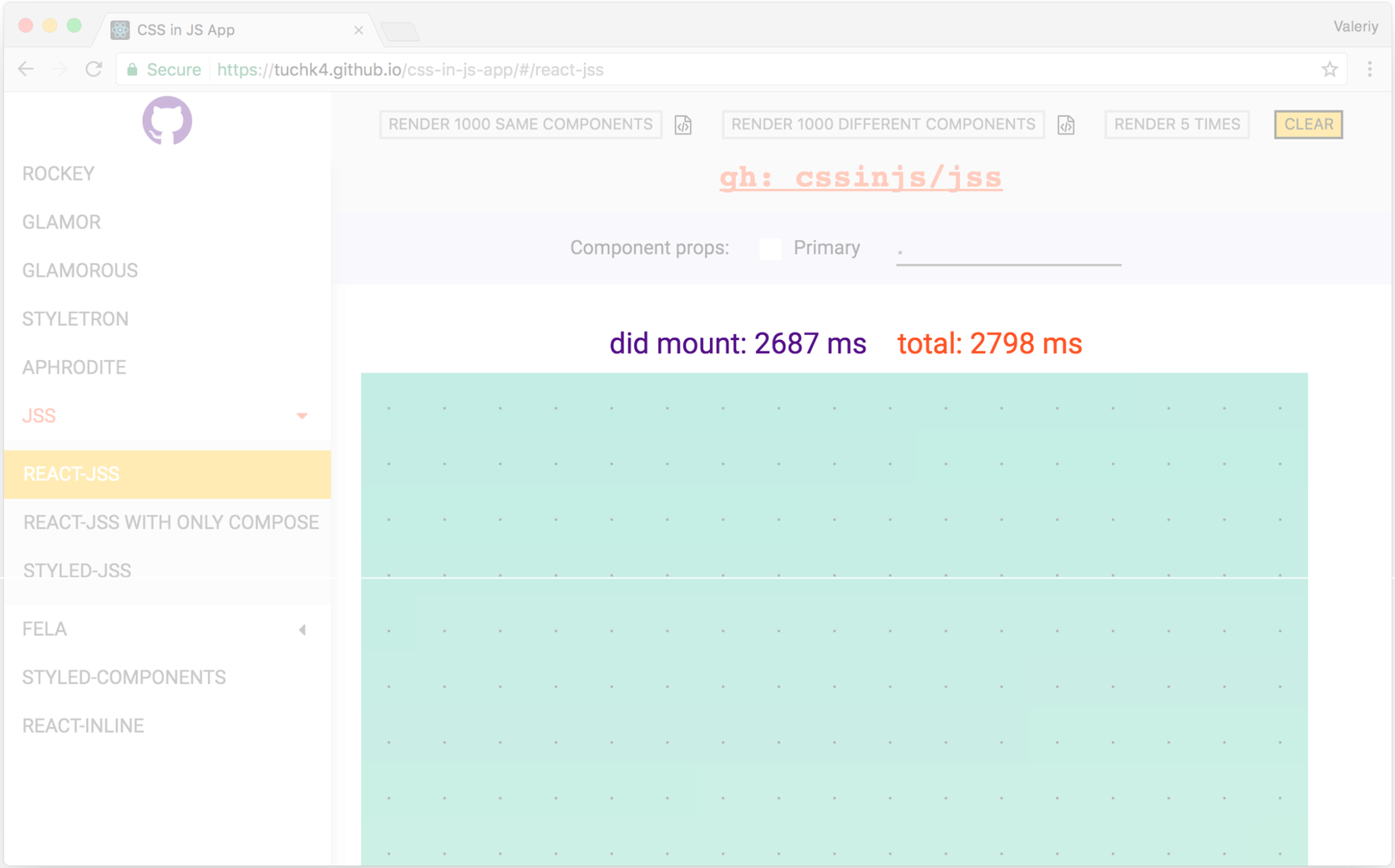
JSS

```
.block-0-1 {
  background-color: #dc2430;
}

.block-0-0 {
  height: 16px;
  display: inline-block;
  min-width: 16px;
}
```

PLARIUM

JSS

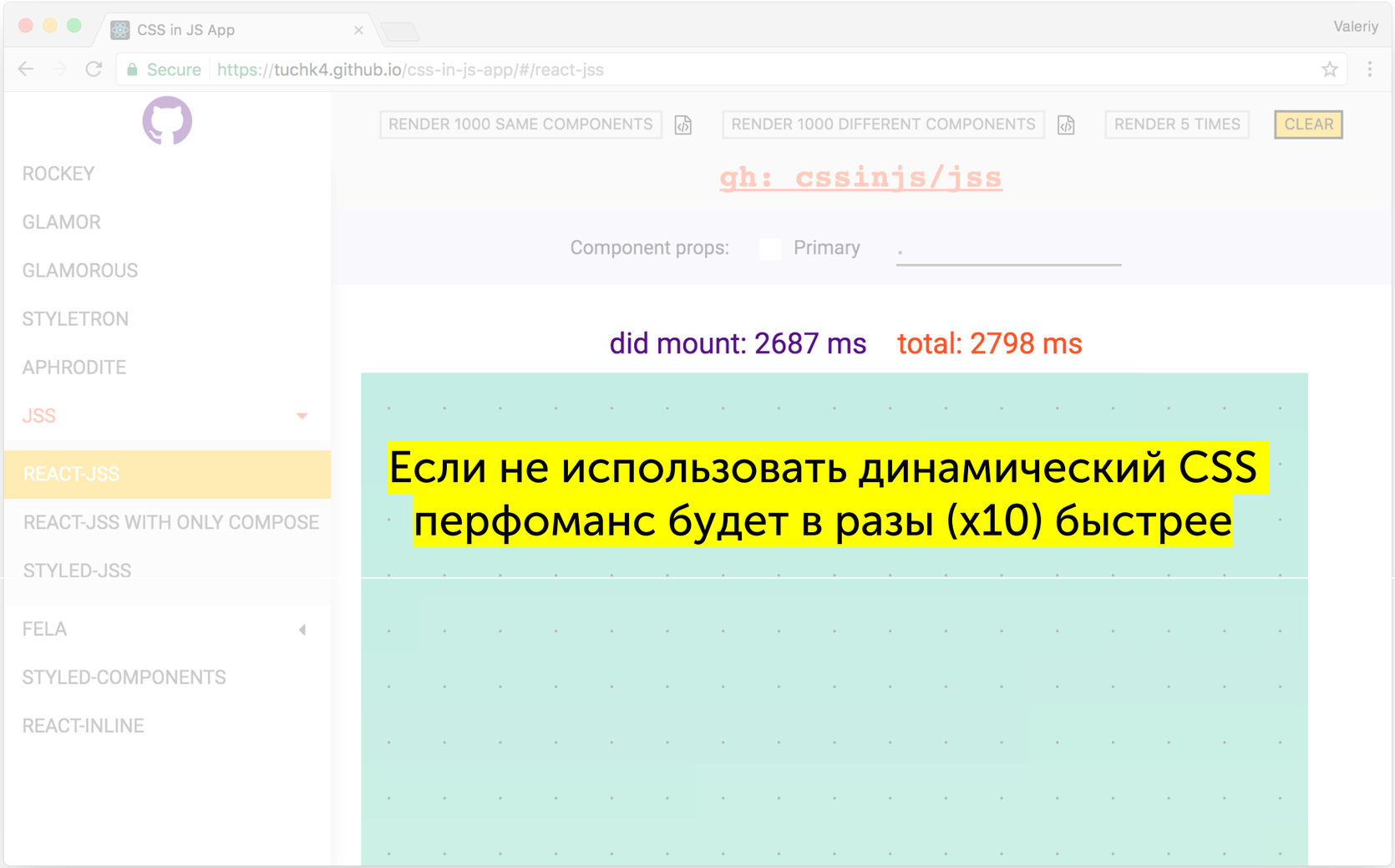
JSS



PLARIUM

JSS

JSS



PLARIUM

JSS

JSS

Мы не используем:

- Низкий перфоманс для динамических правил
- Не хотим описывать стили объектом

PLARIUM

FELA

PLARIUM

FELA

“If the view is a function of state, your CSS should be too”

```
Style = Function(props)
```

PLARIUM

FELA

```
const rule = props => ({
  minWidth: '16px',
  height: '16px',
  display: 'inline-block',
  backgroundColor: colors[props.i]
})
```

PLARIUM

FELA

```
const rule = props => ({
  minWidth: '16px',
  height: '16px',
  display: 'inline-block',
  backgroundColor: colors[props.i]
})
```

PLARIUM

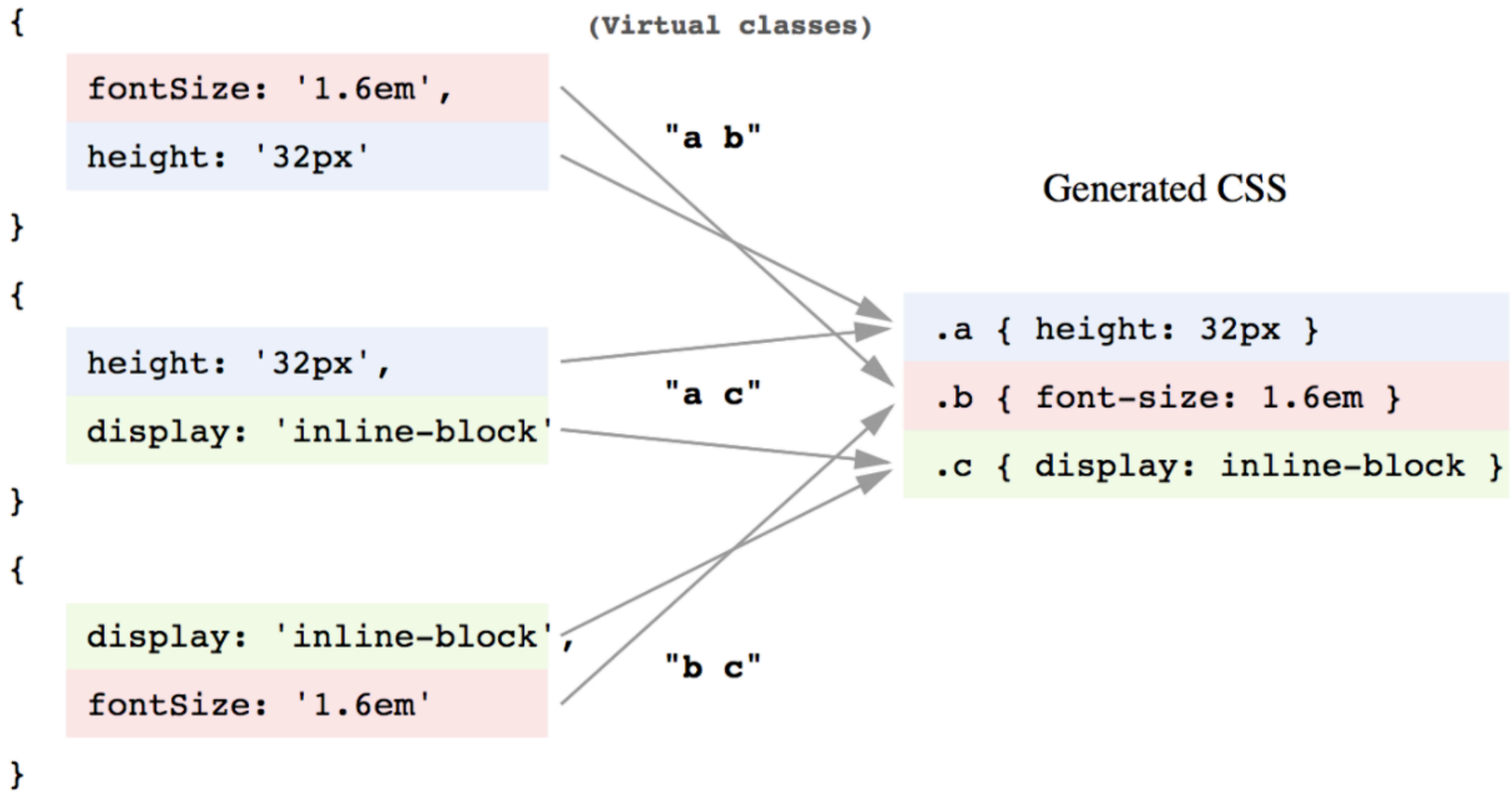
FELA

```
const rule = props => ({
  minWidth: '16px',
  height: '16px',
  display: 'inline-block',
  backgroundColor: colors[props.i]
})
```

PLARIUM

FELA

Atomic CSS design



PLARIUM

FELA

```
.a {
    background-color: #7b4397;
}
.b {
    display: inline-block;
}
.c {
    height: 16px;
}
.d {
    min-width: 16px;
}
```

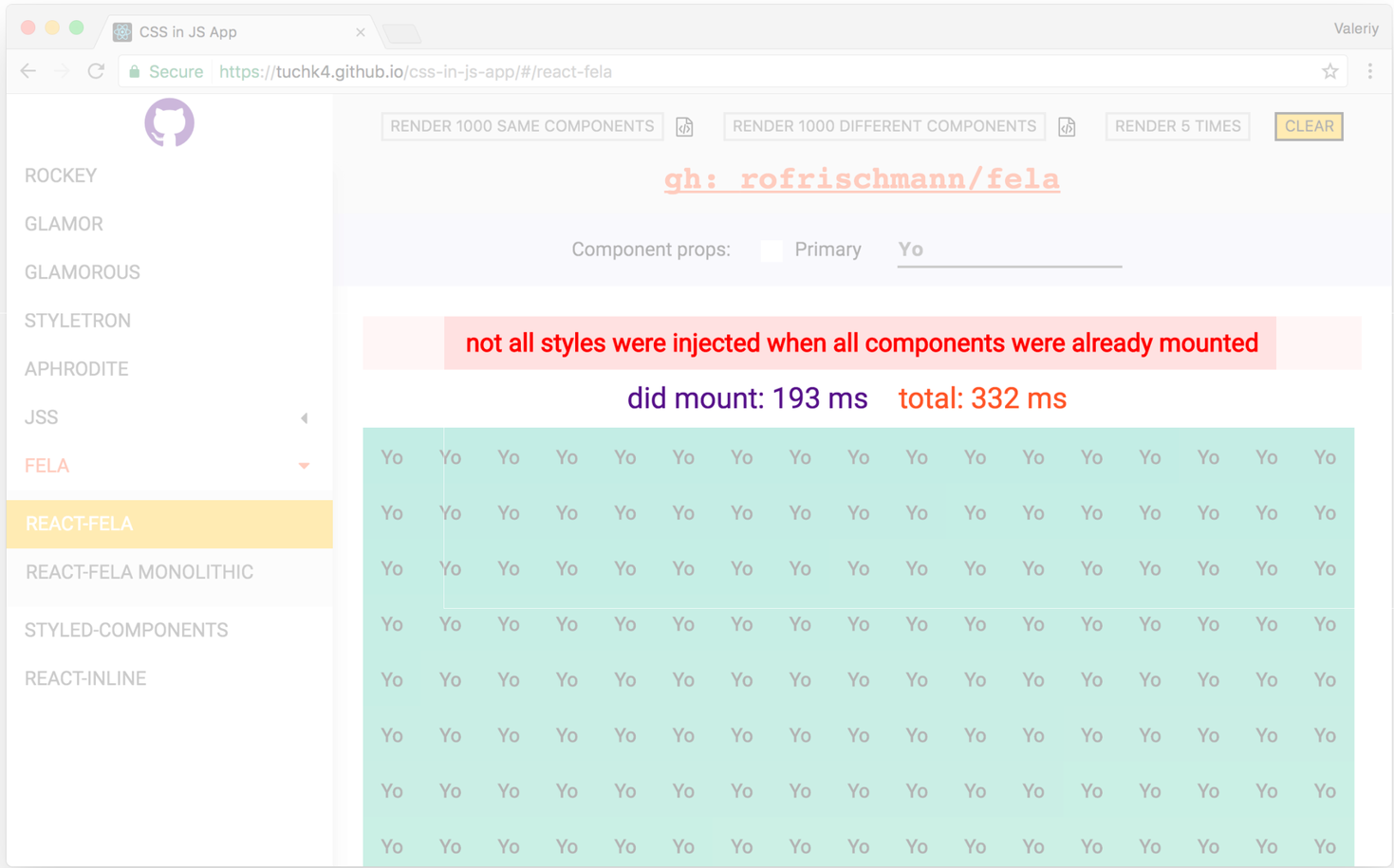
PLARIUM

FELA

```
<div class="a b c d e f g h anx j k">Yo</div>
<div class="a b c d e f g h any j k">Yo</div>
<div class="a b c d e f g h any j k">Yo</div>
<div class="a b c d e f g h any j k">Yo</div>
<div class="a b c d e f g h anz j k">Yo</div>
<div class="a b c d e f g h anz j k">Yo</div>
<div class="a b c d e f g h anz j k">Yo</div>
<div class="a b c d e f g h aoa j k">Yo</div>
<div class="a b c d e f g h aoa j k">Yo</div>
<div class="a b c d e f g h aob j k">Yo</div>
<div class="a b c d e f g h aob j k">Yo</div>
<div class="a b c d e f g h aob j k">Yo</div>
<div class="a b c d e f g h aob j k">Yo</div>
<div class="a b c d e f g h aoc j k">Yo</div>
<div class="a b c d e f g h aoc j k">Yo</div>
<div class="a b c d e f g h aoc j k">Yo</div>
<div class="a b c d e f g h aoc j k">Yo</div>
<div class="a b c d e f g h aoc j k">Yo</div>
```

PLARIUM

FELA



PLARIUM

FELA

Мы не используем:

- Не хотим описывать стили объектом
- Не удобно работать в devtools (Atomic CSS design)

PLARIUM

GLAM

PLARIUM

GLAM

```
<div css={`
    color: ${color};
    font-size: 128px;
    font-family: sans-serif;
    margin: 0;
`}>Hello World!</div>
```

PLARIUM

GLAM

babel →

```
<div className="css-19w4ohx">Hello World!</div>
```

PLARIUM

GLAM

```
.vars-l37dqe {
    --css-19w4ohx-0: #4263eb;
}

.css-19w4ohx {
    color: var(--css-19w4ohx-0);
    font-size: 128px;
    font-family: sans-serif;
    margin: 0;
}
```

PLARIUM

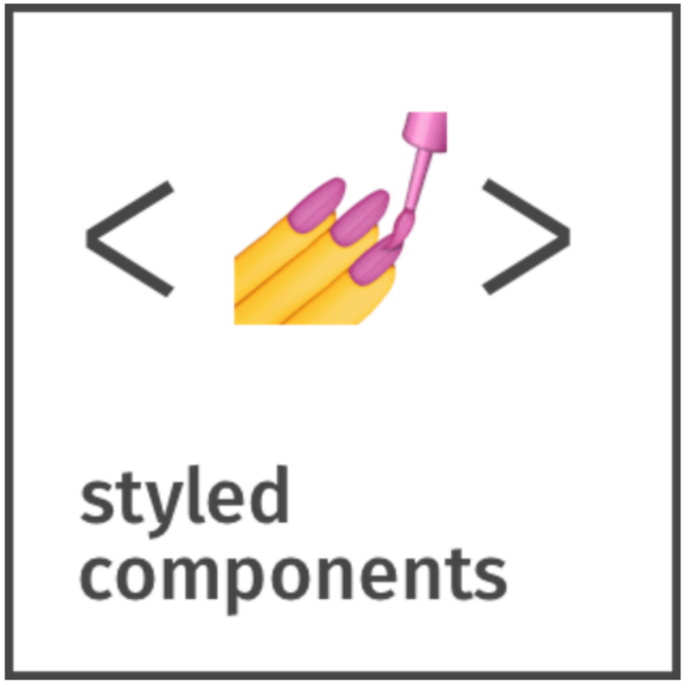
GLAM

Мы не используем:

- Нужен babel plugin и webpack loader
- Много чего еще не реализовано

 Create React App

PLARIUM



PLARIUM

styled-components


```
const Block = styled.div`
  min-width: 16px;
  height: 16px;
  display: inline-block';
  background-color: ${props => colors[props.i]}
`
```

PLARIUM

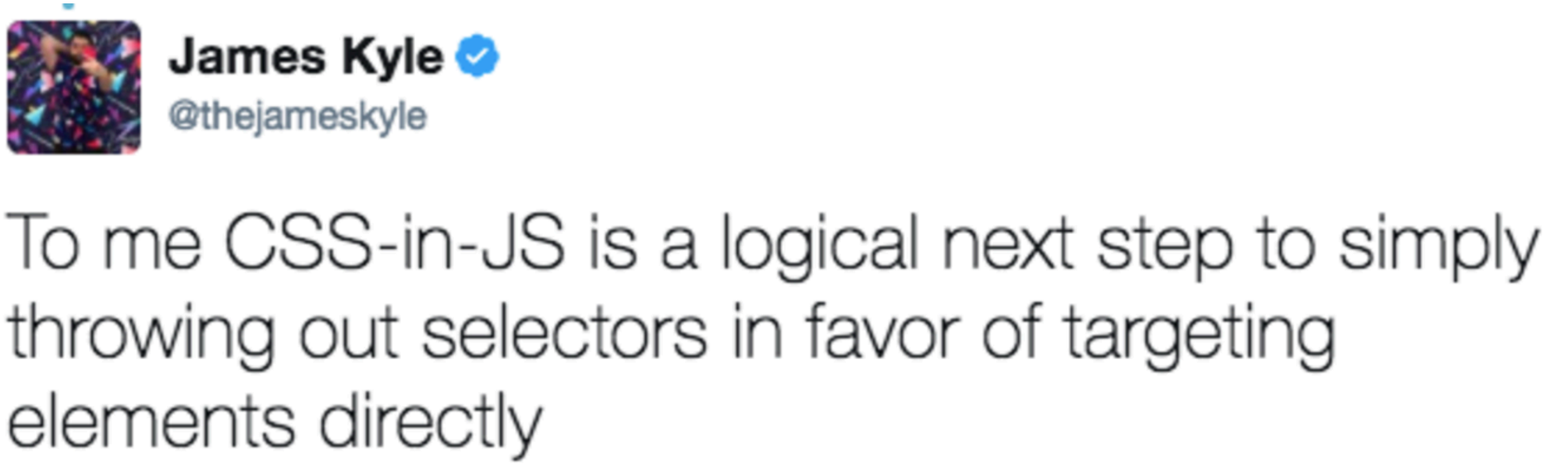
styled-components


```
const Block = styled.div`
  min-width: 16px;
  height: 16px;
  display: inline-block';
  background-color: ${props => colors[props.i]}
`
```

PLARIUM

styled-components

styled components



PLARIUM

styled-components

styled components

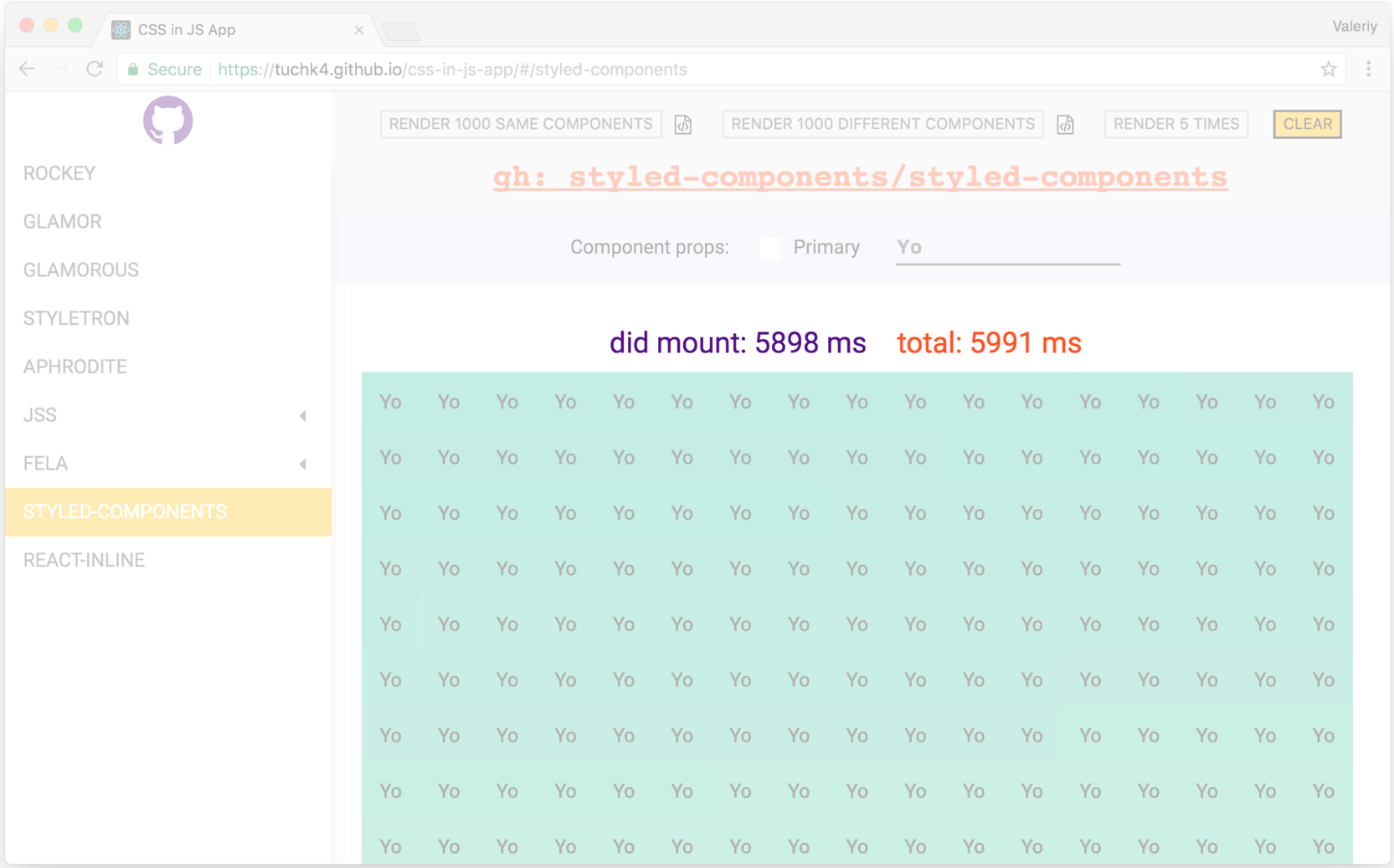
```
.cJoRIh {
    min-width: 16px;
    height: 16px;
    display: inline-block;
    background-color: #00dbde;
}
```

```
.sc-bdVaJa {
}
```

PLARIUM

styled-components

styled components



styled-components

Мы не используем:

- Очень низкий перфоманс
- Не удобно работать в devtools

PLARIUM

Требования к библиотеке

Требования к библиотеке

- Должна быть быстрой!

Требования к библиотеке

- Должна быть быстрой!
- Небольшой размер

Требования к библиотеке

- Должна быть быстрой!
- Небольшой размер
- Максимально совместим с devtools

Требования к библиотеке

- Должна быть быстрой!
- Небольшой размер
- Максимально совместим с devtools
- Читаемые имена CSS классов

PLARIUM

Требования к библиотеке

- Должна быть быстрой!
- Небольшой размер
- Максимально совместим с devtools
- Читаемые имена CSS классов
- Валидация правил и autoprefix

PLARIUM

Требования к библиотеке

- Должна быть быстрой!
- Небольшой размер
- Максимально совместим с devtools
- Читаемые имена CSS классов
- Валидация правил и autoprefix
- Динамический CSS

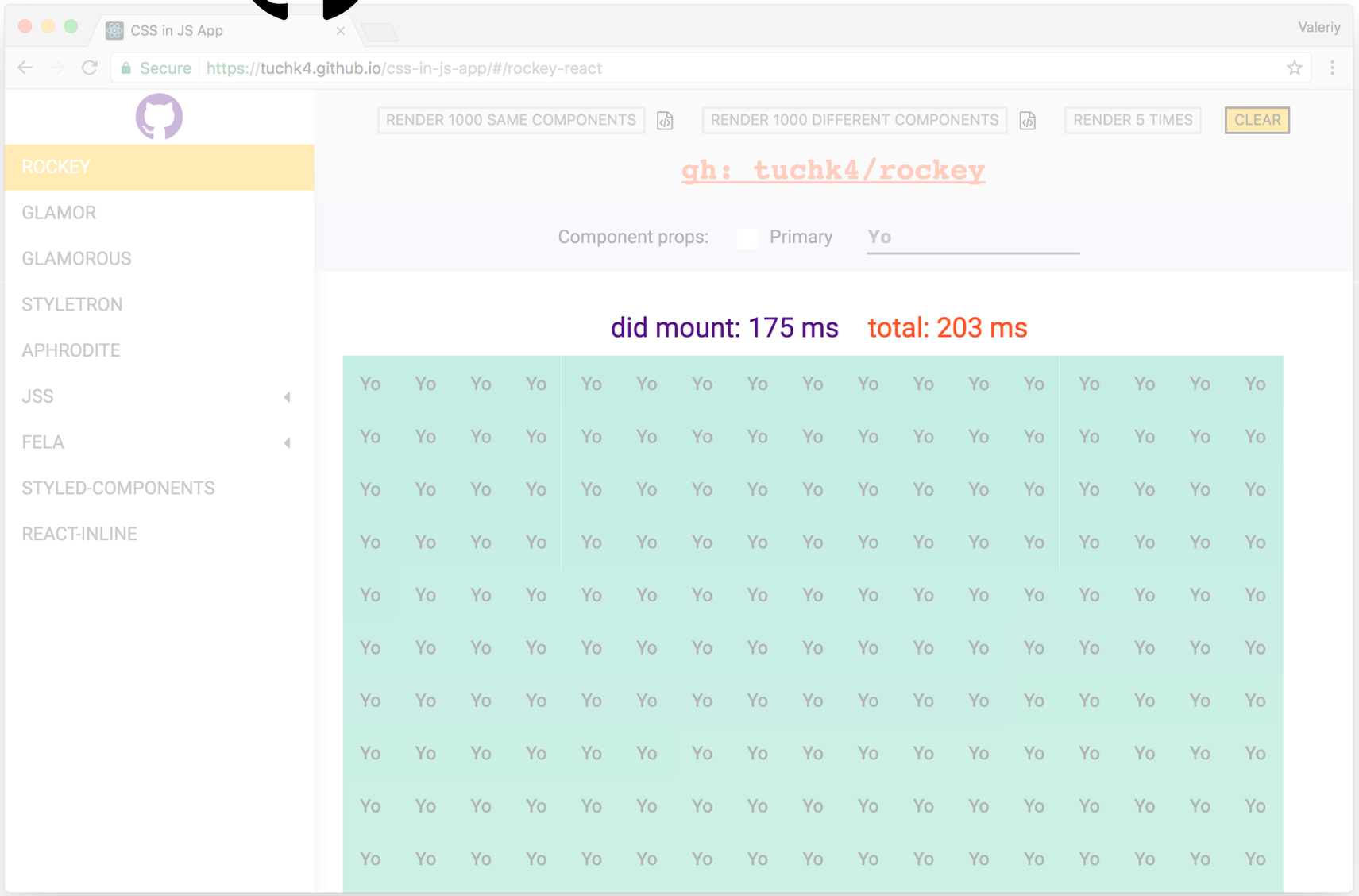
PLARIUM

ЧТО и ПОЧЕМУ используем мы?

PLARIUM

tuchk4/rockey

tuchk4/rockey



PLARIUM

rockey + recompose

```
compose(
  css`
    font-size: 20px;
    padding: 5px;
    margin: 5px;
    background: white;
    border: 1px solid #ccc;
  `,

  css`
    ${props => props.isRaised && `
      box-shadow: 1px 1px #ccc;
    `}
  `
)(Button)
```

PLARIUM

tuchk4/rockey

Компоненты как селекторы

PLARIUM

tuchk4/rockey

Компоненты как селекторы

```
const Card = (props) => (
  <div>
    <CardHeader>
      <Title>I am  CardHeader</Title>
      <CloseIcon/>
    </CardHeader>
    <CardBody>I am Body</CardBody>
    <CardActions>
      <Button>Click me</Button>
    </CardActions>
  </div>
)
```

PLARIUM

tuchk4/rockey

Компоненты как селекторы

```
rockey(Card)`
  Card {
    width: 100px;

    CardHeader {
      background: #fc3;
      CloseIcon {
        float: right;
      }
    }

    CardActions {
      Button {
        float: right;
      }
    }
  }
`
```

PLARIUM

tuchk4/rockey

«Читабельные» имена CSS классов

PLARIUM

tuchk4/rockey

«Читабельные» имена CSS классов

```
▼<div class="Card-66f5">
    <div class="CardHeader-be1e">I'm CardHeader</div>
    <div class="CardBody-8ad4">I'am Body</div>
  </div>
▼<div class="PrimaryCard-2da2 Card-66f5">
    <div class="CardHeader-be1e">I'm CardHeader</div>
    <div class="CardBody-8ad4">I'am Body</div>
  </div>
```

PLARIUM

tuchk4/rockey

Правильное наследование

PLARIUM

tuchk4/rockey

Правильное наследование

```
const Button = rockey.button`
  font-size: 20px;
  padding: 5px;
  margin: 5px;
  background: white;
  border: 1px solid #ccc;
`

const PrimaryButton = Button`
  color: blue;
`

const SuperButton = PrimaryButton`
  color: red;
`
```

PLARIUM

tuchk4/rockey

Правильное наследование

```
<button class="Button-b95b">Button</button>
<button class="PrimaryButton-aee2 Button-b95b">PrimaryButton</button>
<button class="SuperButton-baa8 PrimaryButton-aee2 Button-b95b">SuperButton</button>
```

PLARIUM

tuchk4/rockey

Правильное наследование

```
<button class="Button-b95b">Button</button>
<button class="PrimaryButton-aee2 Button-b95b">PrimaryButton</button>
<button class="SuperButton-baa8 PrimaryButton-aee2 Button-b95b">SuperButton</button>
```

PLARIUM

tuchk4/rockey

Правильное наследование

```
<button class="Button-b95b">Button</button>
<button class="PrimaryButton-aee2 Button-b95b">PrimaryButton</button>
<button class="SuperButton-baa8 PrimaryButton-aee2 Button-b95b">SuperButton</button>
```

PLARIUM

tuchk4/rockey

Looks

PLARIUM

tuchk4/rockey

Looks

```
<Button primary={true}>I am m Button</Button>

<PrimaryButton>I am PrimaryButton</PrimaryButton>
```

PLARIUM

tuchk4/rockey

Looks

```
<Button primary={true}>I am m Button</Button>

<PrimaryButton>I am PrimaryButton</PrimaryButton>
```

PLARIUM

tuchk4/rockey

Looks

Button	raised	disabled	success	warning	primary	ripple
raised	-					
disabled		-				
success			-			
warning				-		
primary					-	
ripple						-

PLARIUM

tuchk4/rockey

Looks

Button	raised	disabled	success	warning	primary	ripple
raised	-	✅	✅	✅	✅	✅
disabled	✅	-	✅	✅	✅	✅
success	✅	✅	-	❌	❌	✅
warning	✅	✅	❌	-	❌	✅
primary	✅	✅	❌	❌	-	✅
ripple	✅	✅	✅	✅	✅	-

PLARIUM

tuchk4/rockey

Looks

```
const Button = rockey.button`
  padding: 5px;
  color: ${props => props.disabled ? '#ccc' : '#000'};
`

const { PrimaryButton, SuccessButton } = Button.look`
  PrimaryButton {
    color: blue;
  }

  SuccessButton {
    color: green;
  }
`
```

PLARIUM

tuchk4/rockey

Looks

```
const Button = rockey.button`
  padding: 5px;
  color: ${props => props.disabled ? '#ccc' : '#000'};
`

const { PrimaryButton, SuccessButton } = Button.look`
  PrimaryButton {
    color: blue;
  }

  SuccessButton {
    color: green;
  }
`
```

PLARIUM

Реализация

PLARIUM

Реализация

С использованием шаблонных строк

```
const Block = styled.div`
  min-width: 16px;
  height: 16px;
  display: inline-block';
  background-color: ${props => colors[props.i]}
`
```

PLARIUM

Реализация

С использованием шаблонных строк

- Быстрее писать код

PLARIUM

Реализация

С использованием шаблонных строк

- Быстрее писать код
- ctrl+c / ctrl+v из CSS файлов и примеров

PLARIUM

Реализация

С использованием шаблонных строк

- Быстрее писать код
- ctrl+c / ctrl+v из CSS файлов и примеров
- Более привычный синтаксис

PLARIUM

Этапы

PLARIUM

Этапы

PLARIUM

Нужен ли настоящий AST?

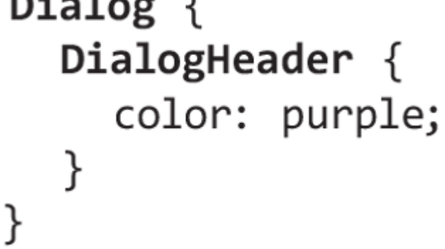
PLARIUM

Нужен ли настоящий AST?

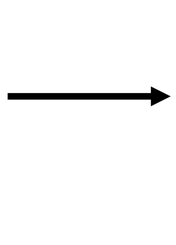
postCSS

Нужен ли настоящий AST?

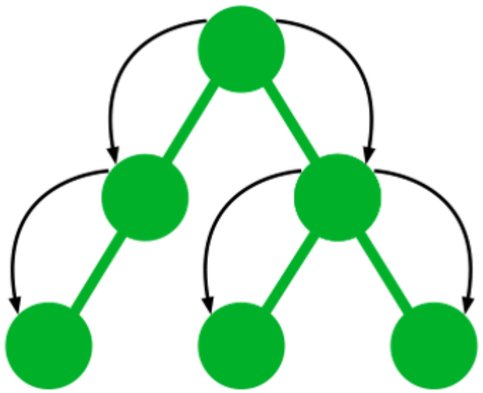
```
Dialog {
  DialogHeader {
    color: purple;
  }
}
```

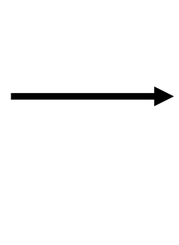


Разбор CSS строки



Разбор AST





CSS

- Вложенные селекторы
- Уникальные имена классов (cssmodules)
- Плагины
- Генерируем финальный CSS

PLARIUM

Нужен ли настоящий AST?

Заранее знаем задачи

- Вложенный CSS
- Autoprefix
- Уникальные имена CSS классов

PLARIUM

Нужен ли настоящий AST?

Задаем функцию определения имени CSS класса при инициализации парсера

```
const parser = createParser({
  getClassName: name => `${name}-${hash()}`
})
```

Нужен ли настоящий AST?

```
Dialog {
  DialogHeader {
    color: purple;
  }
}
```

- Генерируем финальный CSS

PLARIUM

Парсинг

PLARIUM

Парсинг

```
for (let symbol of raw) {
  if (symbol === '{') {

  } else if (symbol === '}') {

  } else if (symbol === ':') {

  }
}
```

PLARIUM

Парсинг

```
for (let symbol of raw) {
    if (isOpenBracket(symbol)) {
        openBlock();
    } else if (isCloseBracket(symbol)){
        closeBlock();
        processBlock();
    }
    ...
}
```

PLARIUM

Парсинг

```
function render(raw) {
  const isOpenBracket = symbol => {...}
  const isCloseBracket = symbol => {...}
  const openBlock = () => {...}
  const closeBlock = () => {...}
  const processBlock = () => {...}

  for (let symbol of raw) {
     if (isOpenBracket(symbol)) {
        openBlock();
     } else if (isCloseBracket(symbol)) {
        closeBlock();
        processBlock();
     }
     ...
  }
}
```

PLARIUM

Парсинг

```
function render(raw) {
  const isOpenBracket = symbol => {...}
  const isCloseBracket = symbol => {...}
  const openBlock = () => {...}
  const closeBlock = () => {...}
  const processBlock = () => {...}

  for (let symbol of raw) {
     if (isOpenBracket(symbol)) {
        openBlock();
     } else if (isCloseBracket(symbol)) {
        closeBlock();
        processBlock();
     }
     ...
  }
}
```

Парсинг

Автоматически заменяем вызов функции на её тело

PLARIUM

Парсинг

```
if (symbol === '{') {
  if (!!components.length) {
    openedBrackets++;
    if (openedBrackets !== 1) {
      current += symbol;
    }
  } else if (!!modificator) {
    openedModificatorBrackets++;
    if (openedModificatorBrackets !== 1) {
      current += symbol;
    }
  }
} else if (symbol === '}') {
  if (!!components.length) {
    openedBrackets--;
  } else if (!!modificator) {
    openedModificatorBrackets--;
  } else {
    throw new RockeySyntaxError(`${raw.slice(i - 1, i + 1)}`);
  }
  if (!!components.length) {
    if (openedBrackets === 0) {
      const selector = [];
    }
  }
}
```

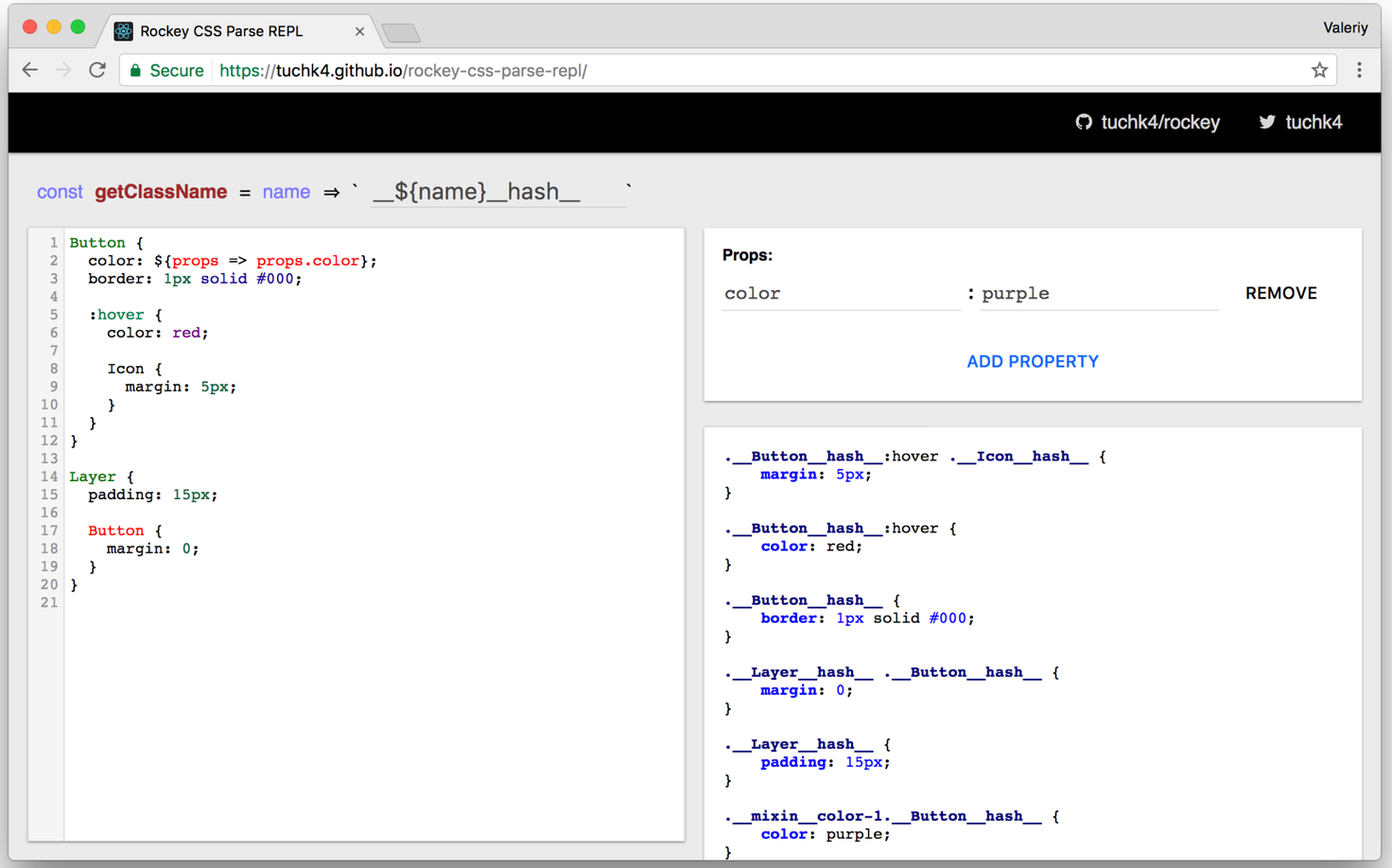
PLARIUM

Парсинг

+40% перфоманса

PLARIUM

tuchk4.github.io/rockey-css-parse-repl



PLARIUM

Как вставить CSS в DOM в рантайме

PLARIUM

Как вставить CSS в DOM в рантайме

- Создать **<style/>** и вставить в **<head/>**
- CSSStyleSheet Api

PLARIUM

CSSStyleSheet Api

PLARIUM

CSSStyleSheet Api

```
const node = document.createElement('style')
document.head.appendChild(style)

const sheet = node.sheet

sheet.insertRule(css, sheet.cssRules.length)
```

PLARIUM

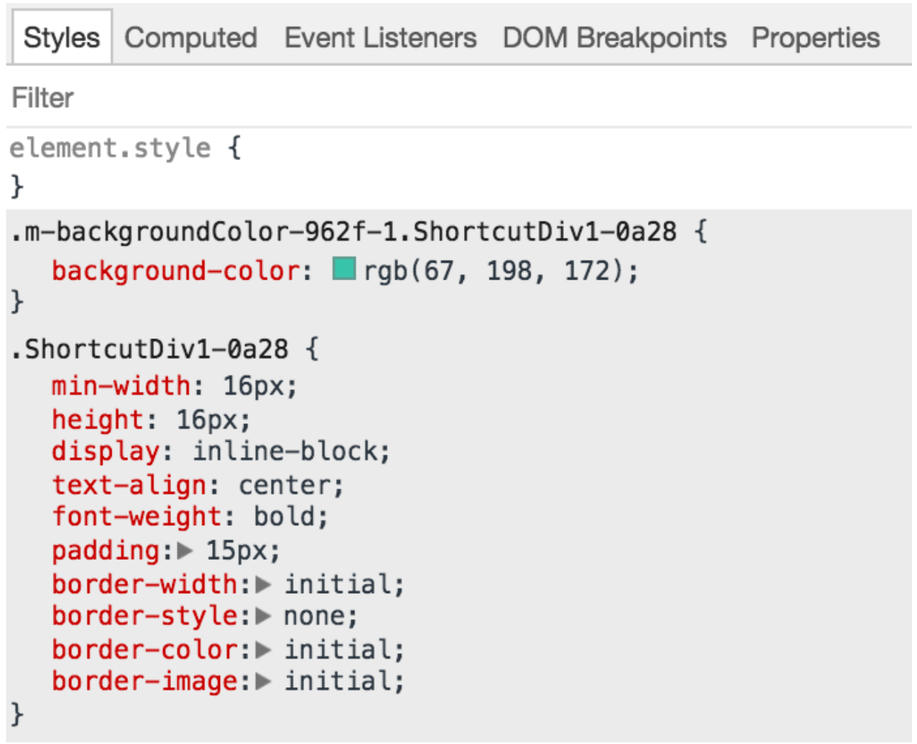
CSSStyleSheet Api

```
const node = document.createElement('style')
document.head.appendChild(style)

const sheet = node.sheet

sheet.insertRule(css, sheet.cssRules.length)
```

PLARIUM

CSSStyleSheet Api

Невозможно редактировать значения



PLARIUM

<style>

PLARIUM

<style>

<style>

```
.button {
  color: red;
}
```

```
.button .icon {
  color: red;
}
```

```
.icon {
  color: black;
}
```

</style>

```
insert(css)
  const textNode = document.createTextNode(css);
  styleNode.appendChild(textNode);
}
```

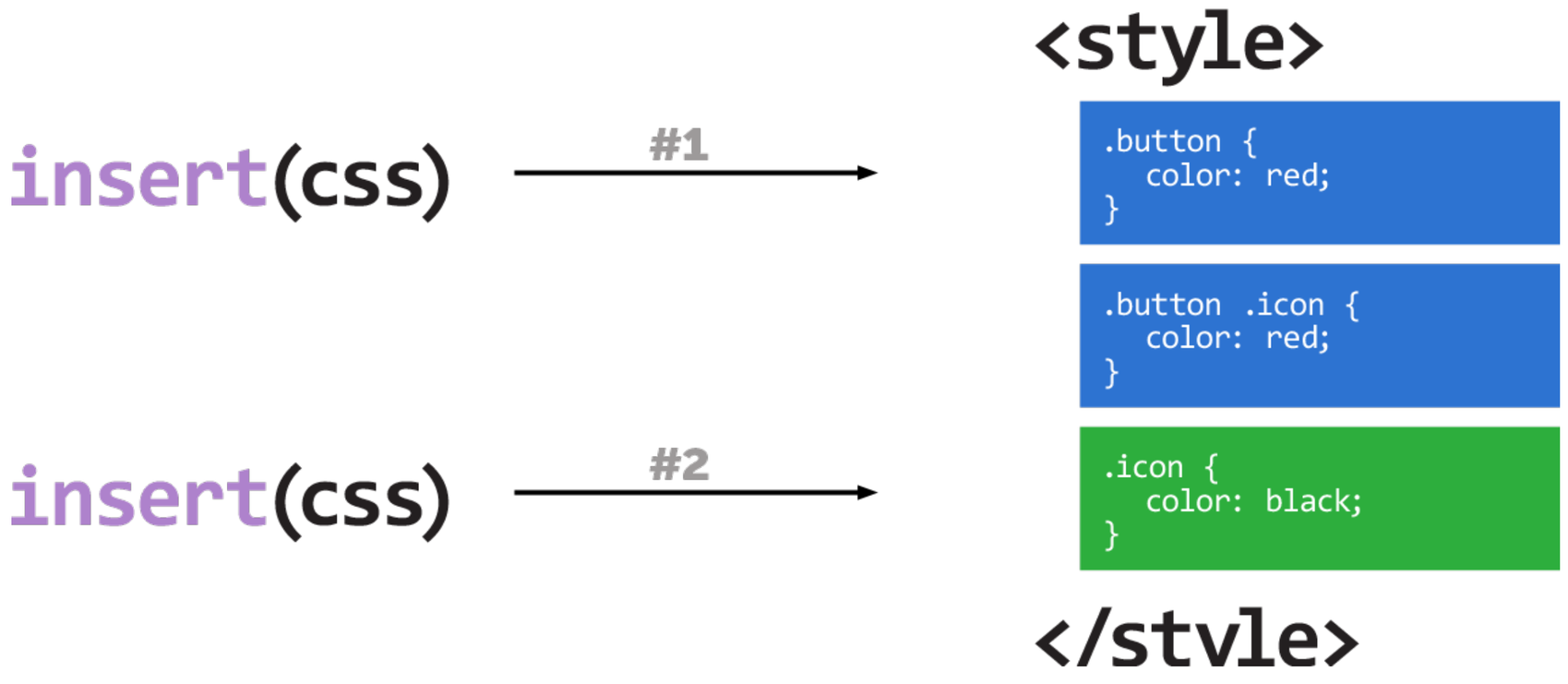
PLARIUM

<style>

Вставлять в момент рендера компонента

PLARIUM

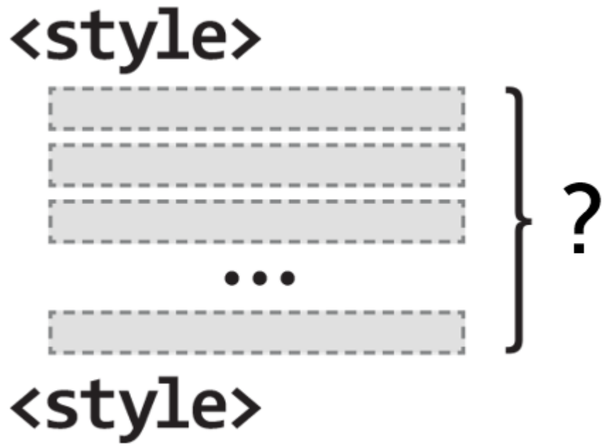
<style>



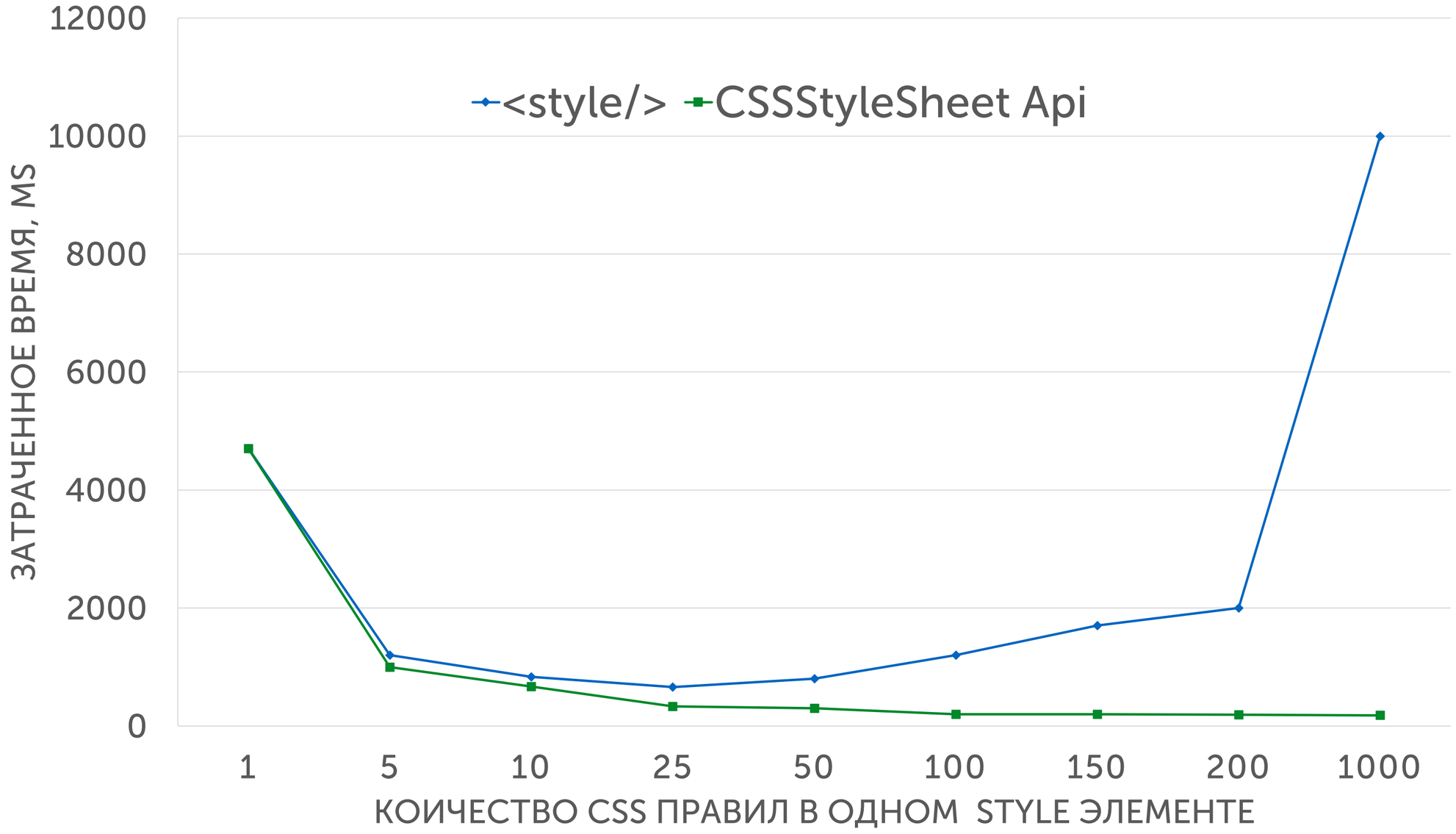
PLARIUM

<style>

Количество стилей в одном <style> элементе



Вставляем 5000 CSS правил



PLARIUM

<style>

Собирать очередь?

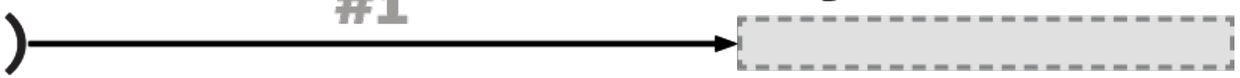
PLARIUM

<style>

Собираем очередь

```
insert(css)
insert(css)
insert(css)
insert(css)
  flushQueue()
```

#1



<style>

```
insert(css)
insert(css)
  flushQueue()
```

#2

<style>

PLARIUM

<style>

Ограничивать размер очереди?

PLARIUM

<style>

MAX_QUEUE_SIZE = 2

PLARIUM

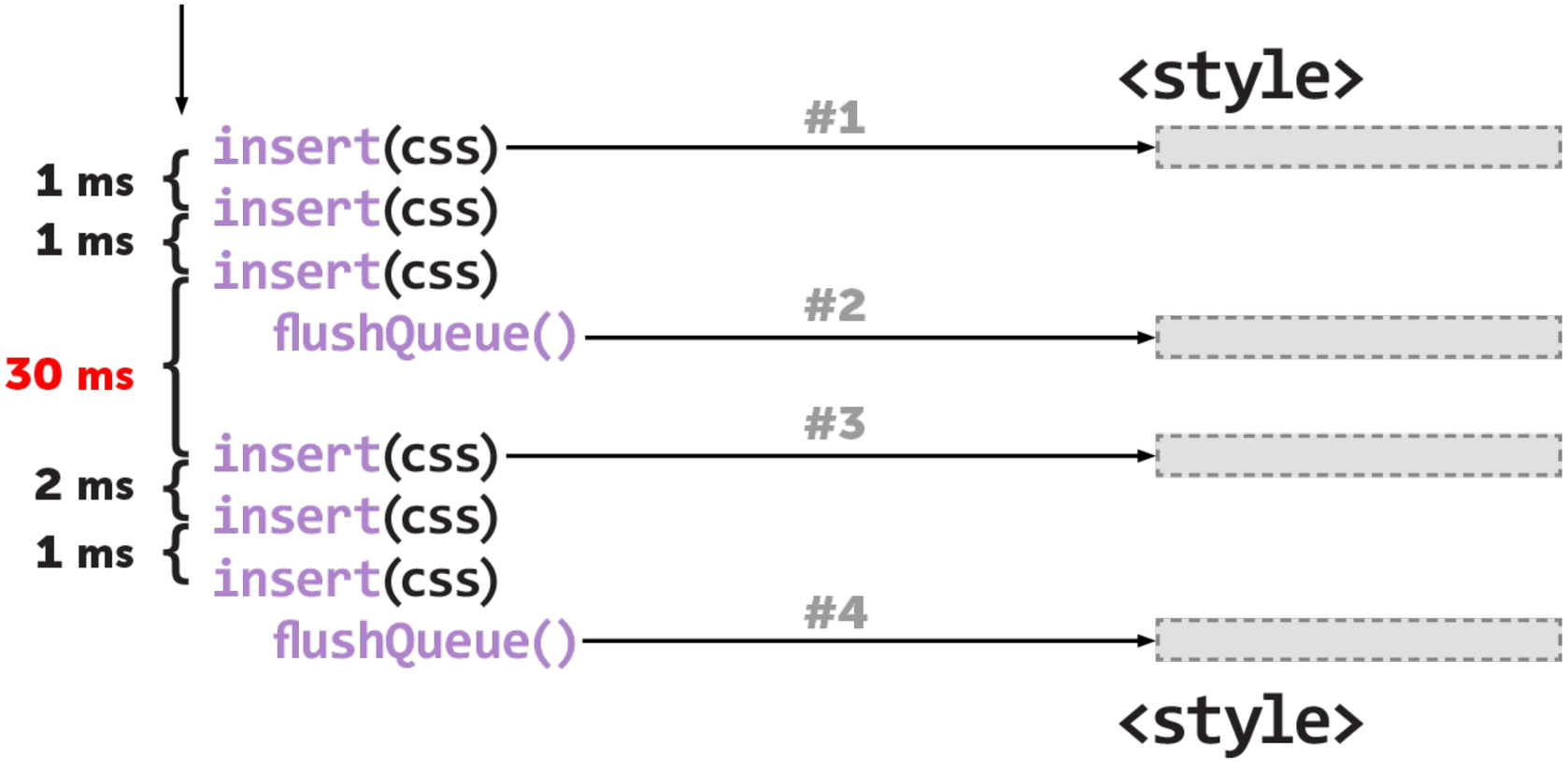
<style>

Замерять частоту вызова `insert()`?

PLARIUM

<style> CALL_FREQUENCY

Собираем очередь если между вызовами менее 5ms

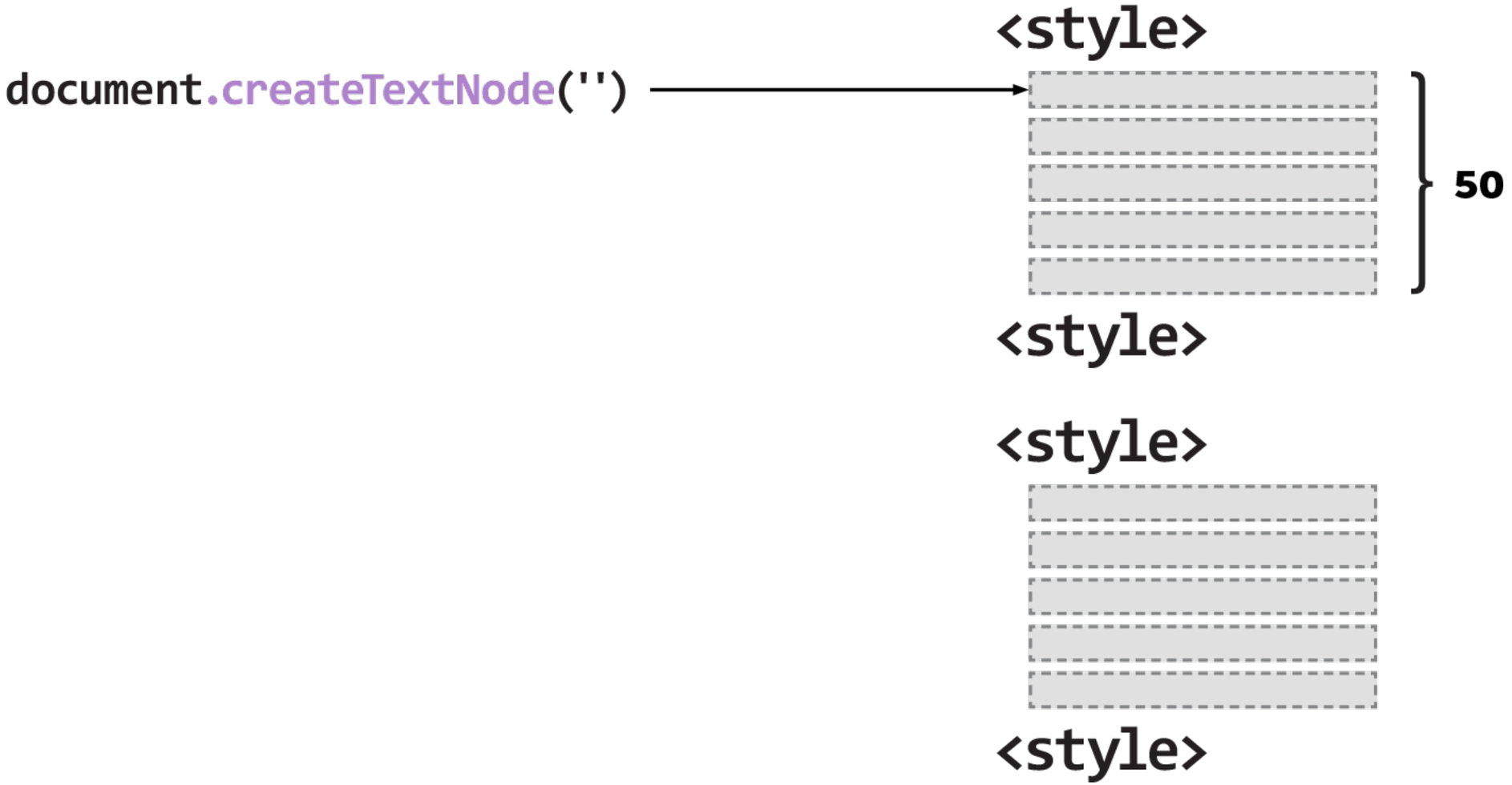


PLARIUM

<style>

При старте приложения создаем **N** textNode
и сразу вставляем их в **<style/>**

<style>



PLARIUM

<style>

```
insert(css) {
  const textNode = requestFreeTextNode(css)
  textNode.textContent = css
}
```

PLARIUM

<style>

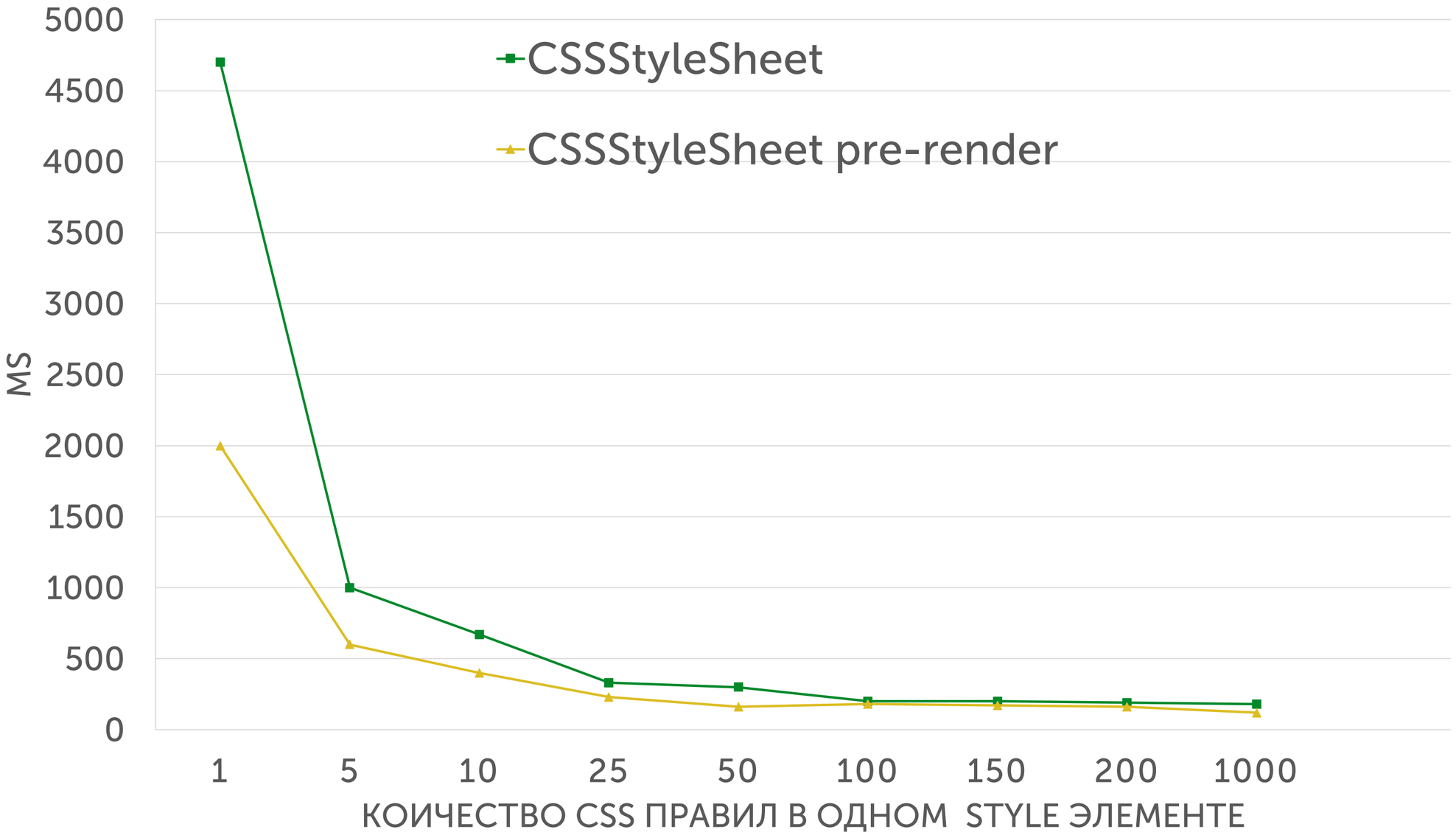
```
insert(css) {
  const textNode = requestFreeTextNode(css)
  textNode.textContent = css
}
```

<style>


<style>
.button {
color: red;
}
requestFreeTextNode()
textNode.textContent = css
</style>

PLARIUM

Вставляем 5000 CSS правил



PLARIUM

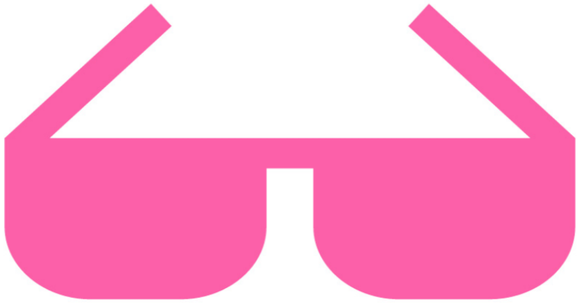
ИТОГ

production – **CSSStyleSheetApi**
1000 CSS правил в одном <style/>

development – работа как с обычным DOM
25 CSS правил в одном <style/>

PLARIUM

tuchk4



tuchk4/awesome-css-in-js

PLARIUM

tuchk4

Спасибо

ВАЛЕРИЙ Сорокобатько