CSS-препроцессоры

Обзор и сравнение с современным CSS

Гулеватая Вероника
Цыганков Тимофей

6 декабря 2025

Что такое CSS?

CSS (Cascading Style Sheets) – язык описания внешнего вида веб-страниц. Позволяет сделать страницу красивой и удобной.

Отвечает за:

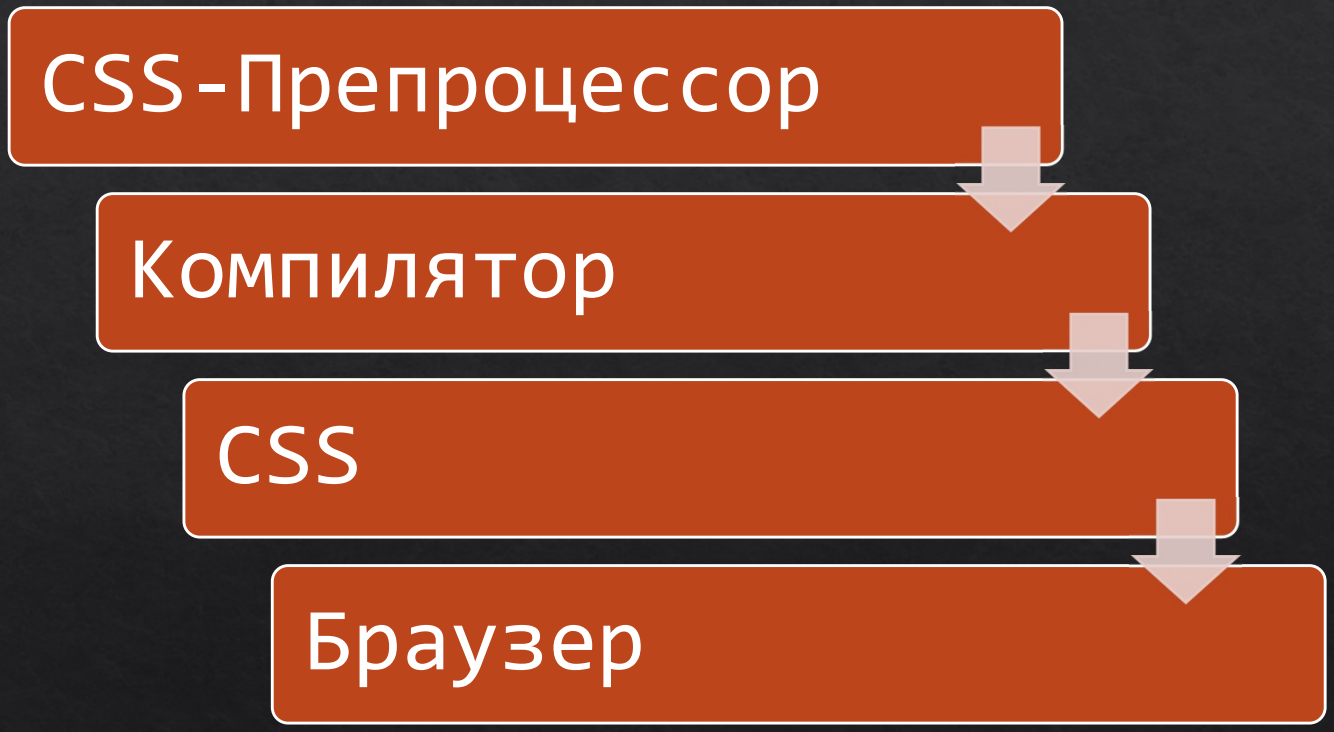
- Цвета и шрифты
- Размеры и отступы
- Расположение элементов
- Анимации и оформление

Что такое CSS-препроцессор?

CSS-препроцессор – надстройка над CSS с расширенным синтаксисом

- Добавляет переменные, вложенность, миксины, функции
- Компилируется в обычный CSS
- Браузер видит только CSS, не препроцессор

Как работает CSS-препроцессор?

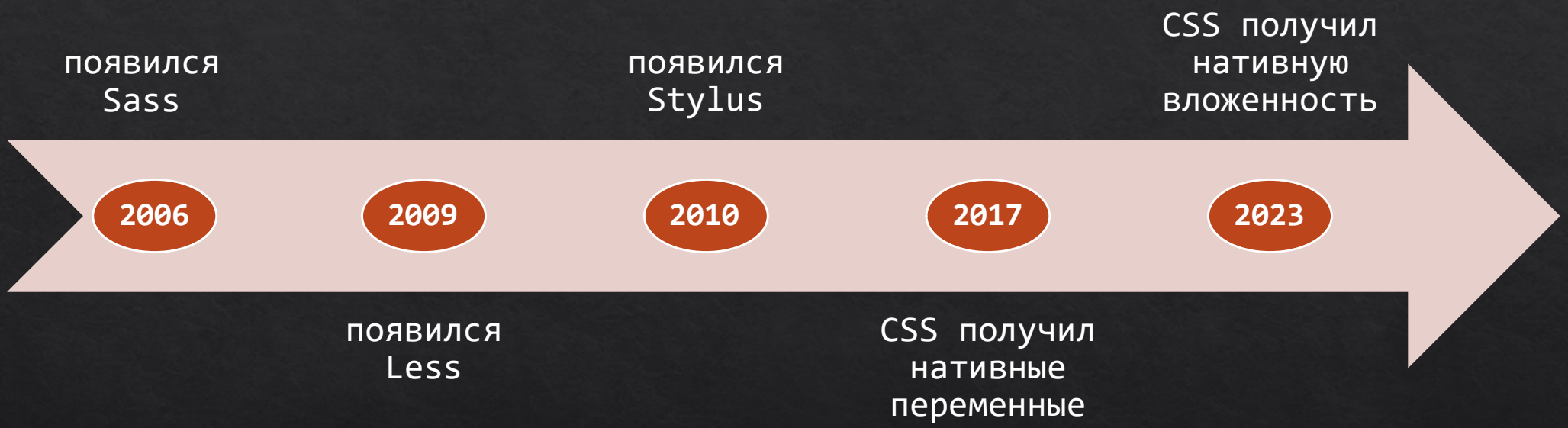


Какие проблемы CSS решают препроцессоры?

Нет переменных	приходится копировать цвета, размеры, шрифты
Нет вложенности	длинные повторяющиеся селекторы
Нет логики	невозможно использовать циклы, условия
Сложность поддержки	изменение цвета бренда = поиск и замена в 100 местах

```
.card
.card .title
.card .title:hover
.card .button
```

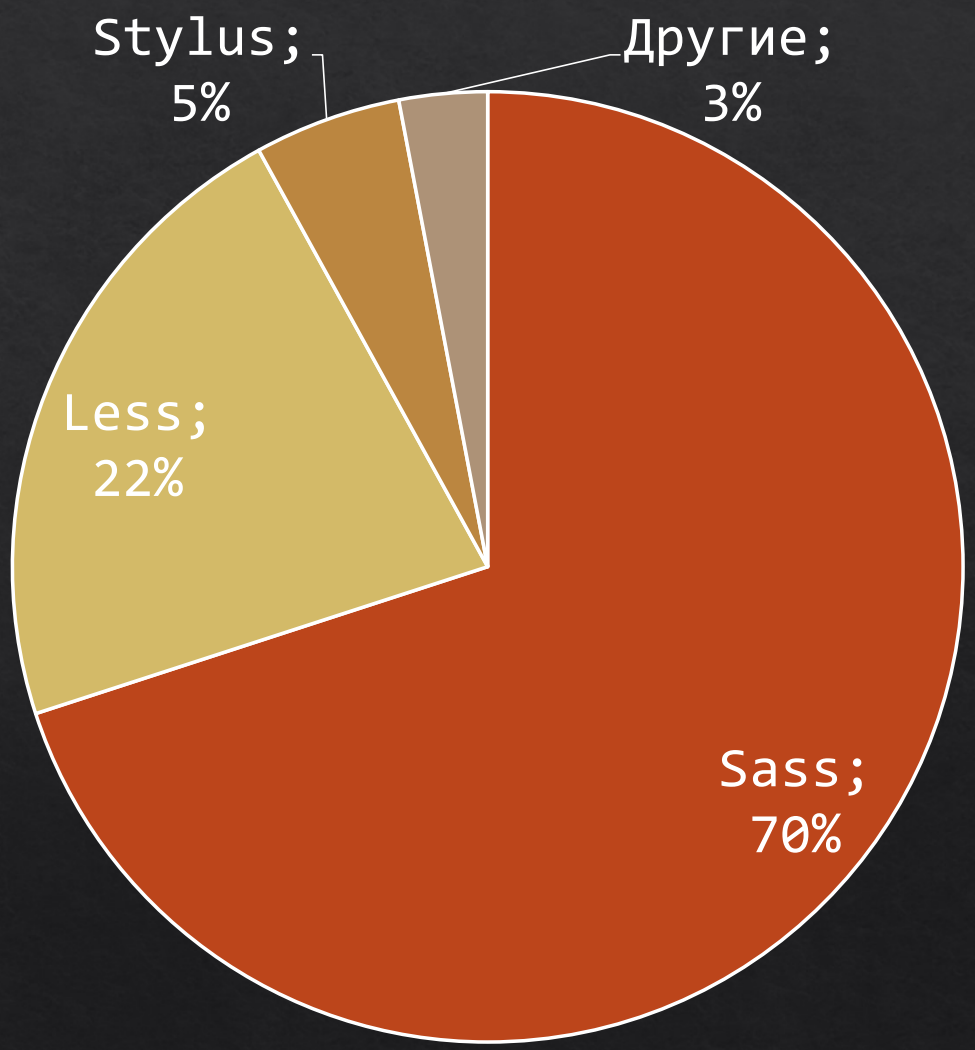
История препроцессоров



Sass: Syntactically Awesome Style Sheets

Less: Leaner Style Sheets

Популярность препроцессоров на сегодняшний день



Переменные — синтаксис


```
$primary-color: #3498db;
$font-size: 16px;
```


```
@primary-color: #3498db;
@font-size: 16px;
```


```
primary-color = #3498db
font-size = 16px
```

Демо: Переменные

demo/01-variables/

- problem.css — CSS с повторяющимися значениями
- solution.scss — SCSS с переменными
- result.css — Скомпилированный результат

→ Переходим в VS Code

Вложенность

Обычный CSS

```
.card { }
.card .title { }
.card .title:hover
  { }
.card--featured { }
```

SCSS

```
.card {
  .title {
     &:hover { }
  }
  &--featured { }
}
```

Демо: Вложенность

demo/02-nesting/

- `flat.css` — Плоский CSS
- `nested.scss` — SCSS с вложенностью
- `result.css` — Скомпилированный результат

→ Переходим в VS Code

Миксины

Миксины — аналог функции, но для стилей. Позволяют создавать переиспользуемые блоки CSS с параметрами.

```scss
// Объявление
@mixin button($color, $size: medium) {
  background: $color;
  padding: if($size == large, 16px 32px, 12px 24px);
  &:hover {
    background: darken($color, 10%);
  }
}

// Использование
.btn-primary { @include button(#3498db); }
.btn-danger  { @include button(#e74c3c, large); }
```

Миксин vs Extend vs Класс: Когда что использовать?

Инструмент	Когда использовать	Плюсы	Минусы
Миксин (@mixin)	Необходимость вставить повторяющийся код, часто с параметрами	• Гибкость • Возможность передачи значений • Лёгкая модифицируемость	• Дублирует CSS в выходном файле (может увеличивать размер)
Extend (@extend)	Несколько селекторов разделяют один и тот же набор свойств	• Объединяет селекторы • Экономит место в итоговом CSS	• Может приводить к запутанным цепочкам селекторов • Нельзя использовать параметры
CSS-класс	Необходимо повторно применять стиль в HTML, а не только в CSS	• Простое и нативное решение • Не требует препроцессора • Минимальный размер CSS	• Меньше гибкости • Иногда нарушает семантику HTML

Демо: Миксины

demo/03-mixins/

- **buttons.scss** — Миксин для кнопок
- **buttons.css** — Сгенерированный CSS

→ Переходим в VS Code

Встроенные функции

Функции для работы с цветами

```scss
$base: #3498db;

.element {
  background: $base;
  border-color: darken($base, 15%);    // Темнее на 15%
  color: lighten($base, 40%);          // Светлее на 40%
  box-shadow: 0 0 10px rgba($base, 0.5); // С прозрачностью
}
```

Математические операции

```scss
$base-size: 16px;

h1 { font-size: $base-size * 2; }      // 32px
h2 { font-size: $base-size * 1.5; }    // 24px
p  { font-size: $base-size; }          // 16px
```

Циклы @for и @each

Цикл @for

```scss
@for $i from 1 through 5 {
  .mt-#{$i} {
    margin-top: $i * 8px;
  }
}
```

Цикл @each

```scss
$colors: (
  primary: #3498db,
  success: #27ae60,
  danger: #e74c3c
);

@each $name, $color in $colors {
  .text-#{$name} { color: $color; }
  .bg-#{$name} { background-color: $color; }
}
```

Используются для генерации утилитарных классов, сеток, палитр

Демо: Циклы

demo/04-loops/

- palette.scss — 15 строк кода
- palette.css — 60+ строк результата

→ Переходим в VS Code

Сравнение препроцессоров

	Sass/SCSS	Less	Stylus
Синтаксис	Строгий, два варианта	CSS-подобный	Гибкий, минималистичный
Переменные	$var	@var	var = value
Миксины	@mixin / @include	.mixin()	mixin()
Функции	Очень обширные	Базовые	Общирные
Условия/циклы	Полноценные	Ограниченные	Полноценные

Где используются представленные препроцессоры

Sass/SCSS

- Bootstrap 4/5
- Foundation
- Bulma
- большинство enterprise-проектов

Less

- Bootstrap 3
- Ant Design

Stylus

- Исторически: некоторые проекты на Node.js

Рекомендуем выбирать SCSS для ваших проектов

Современный CSS: Переменные

! CSS за последние годы получил много возможностей, которые раньше были только в препроцессорах

Sass-переменные	CSS Custom Properties
Компиляция	Живые переменные
Статичные	Можно менять через JS
Нет каскада	Работает наследование

CSS Код с использованием переменных

```css
:root {
  --primary: #3498db;
  --spacing: 16px;
}

.button {
  background: var(--primary);
  padding: var(--spacing);
}
```

Современный CSS: Вложенность

Нативный CSS Nesting

```css
.card {
  padding: 20px;

  & .title {
    font-size: 24px;

    &:hover {
      color: blue;
    }
  }
}
```

!

В 2023 году CSS получил нативную вложенность
Поддержка: Chrome 112+, Firefox 117+, Safari 17.2+

Что умеет CSS, а что — только препроцессоры?

Возможность	CSS	Препроцессоры
Переменные	✓	✓
Вложенность	✓	✓
Миксины	✗	✓
Циклы	✗	✓
Условия	✗	✓
Функции (кастомные)	✗	✓
Модули (@use, @forward)	✗	✓

Если нужна генерация кода, логика, модульная архитектура — препроцессоры незаменимы

Демо: Сравнение SCSS и CSS

demo/05-comparison/

- `component.scss` — Компонент на SCSS
- `component.css` — Тот же компонент на чистом CSS

→ Переходим в VS Code

Когда использовать препроцессоры в нынешнее время?

Используем препроцессор

- Большой проект
- Дизайн-система
- Генерация классов
- Сложные темы
- Команда знает Sass

Достаточно CSS

- Небольшой проект
- Tailwind CSS
- Важна динамическая смена тем через JS
- CSS-in-JS решения

Альтернативы

PostCSS

модульный трансформер CSS (Autoprefixer, CSS Modules)

CSS-in-JS

стили в JavaScript (styled-components, Emotion)

Tailwind CSS

утилитарный подход, "препроцессор наоборот"

Lightning CSS

очень быстрый парсер и минификатор

Ресурсы и полезные ссылки

sass-lang.com
документация
Sass

lesscss.org
документация
Less

Спасибо за внимание!