

Flexbox

Основные flexbox свойства



```
// define flexbox container  
// by default behaves ~ inline-block  
display: flex;  
  
// sets how flex items are placed in the flex  
// container defining the main axis and the direction  
flex-direction: row | column | row-reverse | ...;  
  
// distribution or position of elements along main-axis  
justify-content: center | start | end | ...;  
  
// distribution or position of elements along cross-axis  
align-items: center | start | end | ...;  
  
// sets whether flex items are forced onto one  
// line or can wrap onto multiple lines  
flex-wrap: wrap | nowrap | wrap-reverse | ...;  
  
// sets the flex grow factor of a flex item  
flex-grow: <number>;
```

Лучший cheat-sheet по flexbox

<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

Игра для запоминания свойств

<https://flexboxfroggy.com/#ru>

Node-sass

Модульные стили и css препроцессоры



```
npm install node-sass --save
```

Scss



```
// nested rules
```

```
// css
```

```
.container .nested-container {}
```

```
// scss
```

```
.container {  
  .nested-container {}  
}
```

Scss



```
// variables and imports
```

```
// vars.module.scss
```

```
$my-var-name: 1234;
```

```
$primary-color: #eee;
```

```
// Component.module.scss
```

```
@import './vars.module.scss'
```

```
.container {
```

```
  border: 1px solid $primary-color;
```

```
}
```

Scss

// & character

// CSS

```
.container {}
```

```
.container:hover {}
```

```
.container_modifier {}
```

```
.container + .container {}
```

// SCSS

```
.container {
```

```
  &:hover {}
```

```
  &_modifier {}
```

```
  & + & {}
```

```
}
```

Scss



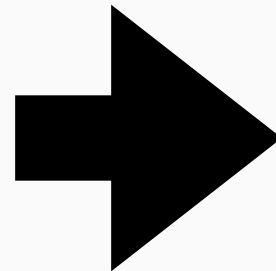
```
// mixins
@mixin flex-centered($arg) {
  font-size: $arg;

  display: flex;
  justify-content: center;
  align-items: center;
}

.container {
  @include flex-centered(26px);
}
```


Модульные стили

styles.css



styles.module.scss

Scss



```
// import as object in js
```

```
// Component.jsx
```

```
import styles from './Component.module.scss';
```

Какие плюсы?

- Независимость стилей в разных файлах. Теперь не нужно выдумывать сложные составные названия для классов. Можно, например, иметь 2 .scss файла и в каждом из них класс `.container { ... }`
- Увеличение скорости рендеринга, тк упрощаются селекторы
- Все префиксы для различных браузеров будут проставлены автоматически

classnames



```
npm install classnames --save
```



```
import React from 'react';
import classNames from 'classnames/bind';

import styles from './NodeSass.module.scss';

const cx = classNames.bind(styles);

export const ClassNames = () => (
  <div className={cx('container', { rotate: true })}>Hello</div>
);
```

Применение conditional styles



```
.container {
  width: 500px;
  height: 300px;
  background: red;
  display: flex;
  justify-content: center;
  align-items: center;

  &:hover {
    background: green;
  }
}

.rotate {
  transform: rotate(45deg);
}
```

Этот код в результате дает
элемент с такими стилями

```
<div class="NodeSass_container__1UmiZ NodeSass_rotate__2dZmZ">Hello</div>
```

Полученный класс -
<имя файла>_<имя класса>_<hash>



```
import styles from "../App.module.scss";

const cx = classNames.bind(styles);

const MyComponent = ({ error }) => (
  <div className={cx("container", { "container-error": error
  })}>Content</div>
);
```

Применение модификаторов



```
.container {
  width: 100px;
  height: 100px;

  background: plum;

  &-error {
    border: 1px solid red;
  }
}
```

**Этот код в результате дает
элемент с такими стилями**

```
<div class="App_container__eSJ6i App_container-error__W-p9H">Content</div>
```

**Полученный класс -
<имя файла>_<имя класса>_<hash>**



Применение модификаторов

```
const cx = classNames.bind(styles);
```

```
const MyComponent = ({ error, color }) => (  
  <div className={cx("container", { [`container-color-${color}`]: true  
})}>Content</div>  
);
```

```
const App = ({ error }) => <MyComponent error={true} color="red"/>;
```



```
.container {  
  width: 100px;  
  height: 100px;
```

```
  background: plum;
```

```
  &-color-red {  
    border: 1px solid red;  
  }  
}
```

Этот код в результате дает
элемент с такими стилями

```
<div class="App_container__eSJ6i App_container-color-red__3Agot">Content</div>
```

Полученный класс -
<имя файла>_<имя класса>_<hash>

Добавить возможность реагировать на изменение темы для

- **Container**
- **Input**