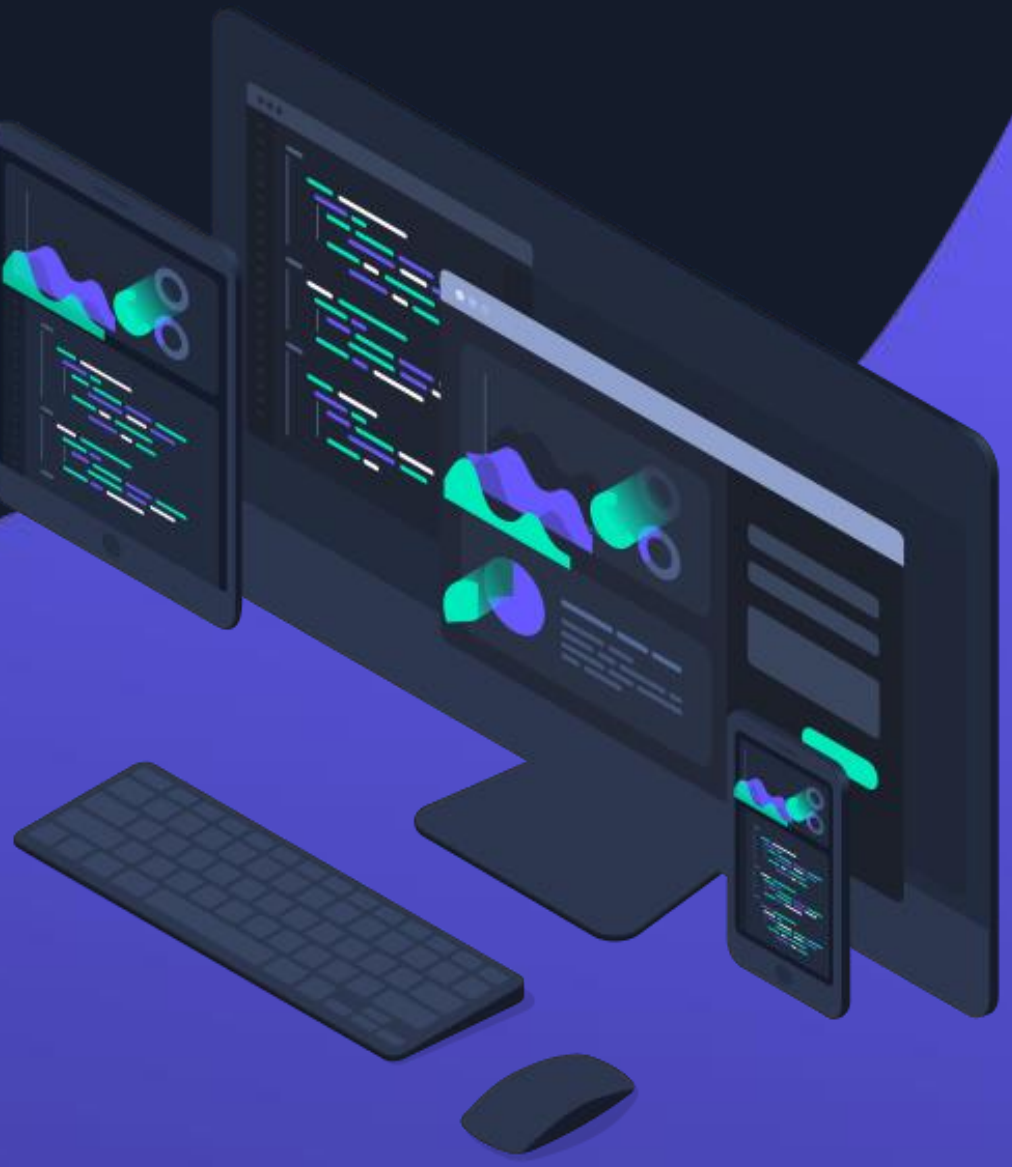
ITEA

ONLINE



ITEA ONLINE

На сегодняшнем занятии:

1. Sass
2. Переменные в Sass
3. Вложенность в Sass
4. Многоразовый CSS (mixin)
5. Модули Sass
6. Наследование в Sass
7. CSS Transform
8. 3D преобразования в CSS
9. CSS Transitions
10. CSS Animation

ITEA ONLINE

Syntactically Awesome StyleSheets

Sass, или «Syntactically Awesome StyleSheets», является языковым расширением CSS. Он добавляет функции, недоступные с использованием базового синтаксиса CSS.

Sass помогает разработчикам упрощать и поддерживать таблицы стилей для своих проектов. Sass может расширить язык CSS, потому что это препроцессор.

Он берет код, написанный с использованием синтаксиса Sass, и преобразует его в базовый CSS. Это позволяет создавать переменные, встраивать одни правила CSS в другие, импортировать другие файлы Sass, и т. д.

В результате получается более компактный, удобный для чтения код.

ITEA ONLINE

Переменные в Sass

Одна из особенностей Sass, отличная от CSS, — это использование переменных. Они объявляются и устанавливаются для хранения данных. В Sass переменные начинаются с $, за которым следует имя переменной. Вот несколько примеров:
Одним из примеров полезности переменных является то, что ряд элементов должен быть одного цвета. Если этот цвет изменен, единственным местом для редактирования кода является значение переменной.

```
$main-fonts: Arial, sans-serif;
$headings-color: green;

//To use variables
h1 {
  font-family: $main-fonts;
  color: $heading-color;
}
```

I T E A ONLINE

Вложенность в Sass

Sass допускает вложение правил CSS, что является полезным способом организации таблицы стилей.

Для большого проекта в файле CSS будет много строк и правил. Вот где вложенность может помочь организовать ваш код, поместив правила дочернего стиля в соответствующие родительские элементы

Sass

```
nav {
    background-color: red;

    ul {
        list-style: none;

        li {
            display: inline-block;
        }
    }
}
```

CSS

```
nav {
    background-color: red;
}

nav ul {
    list-style: none;
}

nav ul li {
    display: inline-block;
}
```

ITEA ONLINE

Многоразовый CSS (mixin)

В Sass mixin представляет собой группу объявлений CSS, которые можно использовать повторно в таблице стилей.
Новые функции CSS требуют времени, прежде чем они будут полностью приняты и готовы к использованию во всех браузерах. По мере добавления функций в браузеры, правила CSS, использующие их, могут потребоваться вендорные префиксы.
Вендорные префиксы — это приставки к названию CSS свойства, которые добавляют производители браузеров для нестандартизированных свойств

```scss
@mixin transform($property) {
  -webkit-transform: $property;
  -ms-transform: $property;
  transform: $property;
}

// Using mixin
.box {
  @include transform(rotate(30deg));
}
```

```css
.box {
  -webkit-transform: rotate(30deg);
  -ms-transform: rotate(30deg);
  transform: rotate(30deg);
}
```

ITEA ONLINE

Модули Sass

Модули в Sass — это отдельные файлы, содержащие сегменты кода CSS. Они импортируются и используются в других файлах Sass. Это отличный способ группировать аналогичный код в модуль, чтобы поддерживать его.

Имена модулей начинаются с символа подчеркивания (_), который сообщает Sass, что это небольшой сегмент CSS. Кроме того, файлы Sass заканчиваются расширением .scss. Чтобы привести код в частичном файле в другой файл Sass, используйте директиву @import.

```
@import "foo.scss";
@import "foo";
@import "rounded-corners", "text-shadow";
```

Например, если все ваши миксины сохранены в частичном имени «_mixins.scss», и они необходимы в файле «main.scss», так их можно использовать в основном файле:

Обратите внимание, что подчеркивание не требуется в операторе импорта — SASS понимает, что он является частичным. После частичного импортирования файлу доступны все переменные, миксины и другой SASS код.

I T E A
ONLINE

Наследование в Sass

У Sass есть функция, называемая расширением/наследованием, которая упрощает заимствование правил CSS из одного элемента и построение на них в другом.
Например, нижний блок правил CSS создает класс `.panel`. Он может имеет фоновый цвет, высоту и границу.
Теперь вам нужна другая панель под названием `.big-panel`. Он имеет те же базовые свойства, что и `.panel`, но также требует ширину и размера шрифта. Можно скопировать и вставить исходные правила CSS из `.panel`, но код становится повторяющимся, когда вы добавляете больше типов панелей.
Директива `extend` — это простой способ повторного использования правил для разных элементов:
`.big-panel` будет иметь те же свойства, что и `.panel`, в дополнение к новым стилям.
Обратите внимание, что подчеркивание не требуется в операторе импорта - SASS понимает, что он является частичным. После частичного импортирования в файлу доступны все переменные, миксины и другой SASS код.

```
.panel {
  background-color: red;
  height: 70px;
  border: 2px solid green;
}

.big-panel {
  @extend .panel;
  width: 150px;
  font-size: 2em;
}
```

ITEA ONLINE

CSS Transform

Трансформ CSS позволяют вам перемещать, поворачивать, масштабировать и изменять элементы. Трансформ (преобразование) — это эффект, который позволяет элементу изменять форму, размер и положение. CSS поддерживает преобразования 2D и 3D.

Метод `translate()` перемещает элемент из его текущего положения (в соответствии с параметрами, заданными для оси X и оси Y).

Метод `rotate()` вращает элемент по часовой стрелке или против часовой стрелки в соответствии с заданной степенью.

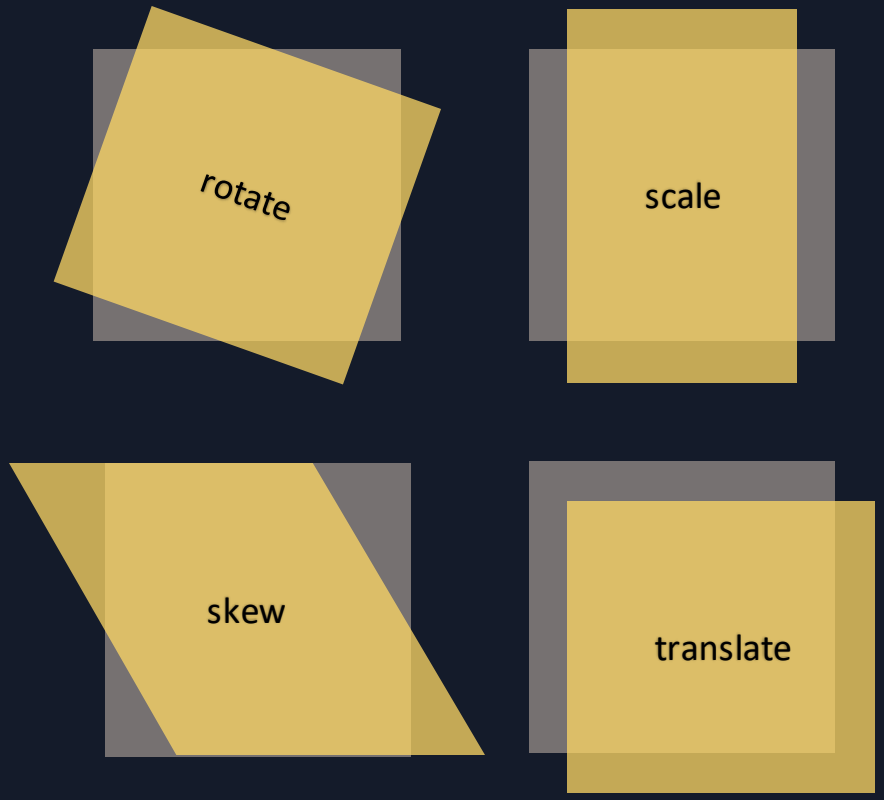
Метод `scale()` увеличивает или уменьшает размер элемента (в соответствии с параметрами, заданными для ширины и высоты).

Метод `skewX()` перемещает элемент вдоль оси X на заданный угол.

Метод `skewY()` искажает элемент вдоль оси Y на заданный угол.

Метод `skew()` искажает элемент вдоль оси X и Y по заданным углам.

Метод `matrix()` объединяет все методы двумерного преобразования в один. Параметры следующие: matrix (`scaleX()`, `skewY()`, `skewX()`, `scaleY()`, `translateX()`, `translateY()`)



I T E A
ONLINE

Наследование в Sass

У Sass есть функция, называемая расширением/наследованием, которая упрощает заимствование правил CSS из одного элемента и построение на них в другом.
Например, нижний блок правил CSS создает класс `.panel`. Он может имеет фоновый цвет, высоту и границу.
Теперь вам нужна другая панель под названием `.big-panel`. Он имеет те же базовые свойства, что и `.panel`, но также требует ширину и размера шрифта. Можно скопировать и вставить исходные правила CSS из `.panel`, но код становится повторяющимся, когда вы добавляете больше типов панелей.
Директива `extend` — это простой способ повторного использования правил для разных элементов:
`.big-panel` будет иметь те же свойства, что и `.panel`, в дополнение к новым стилям.
Обратите внимание, что подчеркивание не требуется в операторе импорта - SASS понимает, что он является частичным. После частичного импортирования в файлу доступны все переменные, миксины и другой SASS код.

```
.panel {
  background-color: red;
  height: 70px;
  border: 2px solid green;
}

.big-panel {
  @extend .panel;
  width: 150px;
  font-size: 2em;
}
```

I T E A
ONLINE

3D преобразования в CSS

CSS позволяет форматировать элементы с помощью 3D-преобразований.

Метод `rotateX()` вращает элемент вокруг своей оси X в заданной степени

Метод `rotateY()` вращает элемент вокруг своей оси Y в заданной степени

Метод `rotateZ()` вращает элемент вокруг своей оси Z в заданной степени

`rotateX(0)`

`rotateX(45deg)`

`rotateX(80deg)`

CSS Transitions

Переходы CSS позволяют вам изменять значения свойств плавно (от одного значения к другому) в течение заданной продолжительности.
Чтобы создать эффект перехода, вы должны указать две вещи:

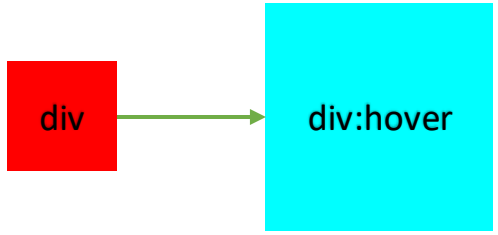
- свойство CSS, которое вы хотите добавить
- продолжительность эффекта

Кривая скорости перехода (Speed Curve of the Transition)
transition-timing-function указывает кривую скорости эффекта перехода и может принимать следующие значения:

- ease,
- linear,
- ease-in,
- ease-out,
- ease-in-out,
- cubic-bezier(n,n,n,n)

```
div {
  width: 100px;
  height: 100px;
  transition: width 2s, height 2s,
background-color 2s;
  background-color: red;
}

div:hover {
  width: 200px;
  height: 200px;
  background-color: aquamarine;
}
```



CSS Animation

Анимация позволяет элементу постепенно меняться от одного стиля к другому. Вы можете изменить столько свойств CSS, сколько хотите, столько раз, сколько хотите.

Чтобы использовать анимацию CSS, вы должны сначала указать ключевые точки (`@keyframes`) для анимации. Ключевые точки (`@keyframes`) сохраняют, какие стили элемент будет иметь в определенное время.

Свойство `animation-duration` определяет, сколько времени потребуется анимации для завершения. Если animation-duration не указано, анимации не будет, потому что значение по умолчанию равно 0 с (0 секунд).

Свойство `animation-delay` определяет задержку для начала анимации.

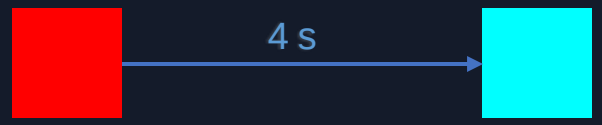
Свойство `animation-iteration-count` указывает количество раз, когда анимация должна запускаться.

Свойство `animation-direction` определяет, следует ли воспроизводить анимацию вперед, назад или в чередующихся циклах.

Свойство `animation-timing-function` определяет кривую скорости анимации.

```
@keyframes example {
  from {background-color: red;}
  to {background-color: aquamarine;}
}

div {
  width: 100px;
  height: 100px;
  background-color: red;
  animation-name: example;
  animation-duration: 4s;
}
```



ITEA ONLINE

Q&A

ITEA

ONLINE