1. Переменные упрощают разработку и поддержку кода, позволяя изменять значения из одного места в программе

1.1. Переменные значений

Создание переменной начинается с символа **@,** далее без пробелов следует имя переменной, состоящее из латинских букв, цифр, тире и знака подчеркивания.

```
// main: styles.less;
@textColor: #eee;
@bgColor: #000;

@grey: #eee;

@colorAccent: #3CFFFF;
@colorSecond: #26ADFF;
@colorDanger: #FC3F4A;
@lightGrey: #8f9496;


@linkColor: @colorSecond;
```

Результат выполнения кода:

```
.block{
      background-color:@color;
      color: @textColor;
}
a{
       color: @linkColor
}
```

1.2. Переменные «Селекторы»

Переменная может содержать имя класса или идентификатора

```
/*переменная — имя селектора*/
@selector-class: border;
@selector-id: nav;

.@{selector-class}
```

{ border: 5px dotted @color; }

#@{selector-id}
{ padding: 10px; }

Результат выполнения кода CSS:

```
.border {border: 5px dotted @color;}
#nav{ padding: 10px; }
```

Использование HTML:

```
<div class=”border”>
```

Арифметические операции (обычный режим)

Допускается производить операции с выражениями, значения которых соответствуют строкам.

Пример	Название	Результат	Числа	Строки
-@a	Отрицание	Смена знака @a	+	ошибка
@a + @b	Сложение	Сумма @a и @b	+	игнорируется*
@a - @b	Вычитание	Разность @a и @b	+	игнорируется*
@a * @b	Умножение	Произведение @a и @b	+	игнорируется*
@a / @b	Деление	Частное от деления @a на @b	+	игнорируется*

```
@fontSize: 16px;
@h1: @fontSize*1.6;
@h2: @fontSize*1.4;
@h3: @fontSize*1.2;

@pad1: 5vw;
@pad2: 5px;
```

Пример

```
padding: (@pad1 + @pad2);
```

Результат выполнения кода CSS:

```
padding: (5wv + 5px);
```

1.3. Переменные «Свойства»

```
/*переменная — имя свойства*/

@prop: color;
@selector-id: nav;

#@{selector-id} {
        @{prop}: pink;
        padding: 10px;
}
```

Результат выполнения кода CSS:

```
#nav{
    color:  pink;
    padding: 10px;
}
```

2. Вложенные правила

В хорошо структурированных таблицах стилей нет необходимости присваивать каждому элементу классы. Достаточно лишь более подробно описывать стили элементов, используя возможность вкладывать селекторы в другие селекторы. К слову, такие селекторы называются вложенными и представляют собой объёмную структуру.

01. Пример вложения

```
#header{
        .menu{
```

```
        color: red;
    }
    .logo{
        width: 100px;
    }
}
```

Результат выполнения кода:

```
#header .menu {
    color: red;
}
#header .logo {
    width: 100px;
}
```

К тому же во вложенной конструкции можно ссылаться на родительский класс, используя знак &.

02. Пример со ссылкой на родителя

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

Результат выполнения кода:

```
a {
    color: blue;
}
a:hover {
    color: red;
}
```

03. Пример объединения

```
.class-1 {
  background-color: #fff;

  &.class-2 {
    color: #000;
  }
}
```

На выходе компилятора получается сдвоенный селектор:

```
.class-1 {
  background-color: #fff;
}
.class-1.class-2 {
  color: #000;
}
```

04. Пример (склеивание) селекторов

```
.button {
  background-color: #ddd;
  color: #000;

  &-add {
    background-color: green;
    color: #fff;
  }

  &-remove {
    background-color: red;
    color: #fff;
  }
}
```

На выходе компилятора получается

```
.button {
  background-color: #ddd;
  color: #000;
}

.button-add {
  background-color: green;
  color: #fff;
}

.button-remove {
  background-color: red;
  color: #fff;
}
```

3. Миксины (примеси)

Препроцессор Less позволяет выполнять примеси — добавлять существующие свойства к вновь создаваемым свойствам. Этот инструмент существенно упрощает написание кода путем исключения повторного написания стилей

Если вы не хотите, чтобы сама примесь не присутствовала в коде CSS после компиляции, то необходимо поставить скобки после определения примеси

Примесь (от англ. mix-in) — набор свойств и селекторов, расширяющий поведение другой сущности (селектора).

Пример css

```
.bordered() {
  border-top: dotted 1px #333;
  border-bottom: solid 2px #333;
}
```

Использование

```
article {
  .bordered;
  color: #443d3d;
```

```
}
```

После компиляции селектор `.bordered` безвозмездно отдаст все свои свойства и вложенные правила селектору `article`. При этом в конечном CSS-файле будут объявлены оба этих селектора.

Реализация:

```
article {
  border-top: dotted 1px #333;
  border-bottom: solid 2px #333;
  color: #443d3d;
}
```

при компиляции не будет создан класс `.bordered`, так как у него указаны скобки после имени. Такая конструкция говорит компилятору, что она чистейшая примесь, которая не хочет быть скомпилирована без явных на то причин.

3.1. Примеси с параметрами

Для того, чтобы примеси были переиспользуемыми в различных контекстах, у них могут быть указаны параметры, в зависимости от которых может меняться цвет, фон и другие значения. Параметры указываются в скобках после имени примеси и представляют собой обычные локальные переменные.

```
.bordered(@color) {
  border-top: dotted 1px @color;
  border-bottom: solid 2px @color;
}

article {
  .bordered(#ccc);
  color: #443d3d;
}
```

Как результат

```
.article {
  border-top: dotted 1px #cccccc;
  border-bottom: solid 2px #cccccc;
  color: #443d3d;
}
```

3.2. Значения параметров по умолчанию

Важной отличительной чертой примесей является возможность указывать значения по умолчанию для переменных. То есть в случае, если примесь вызвана, а значения для параметров не были переданы или переданы частично, то ошибки компилятор не выдаст, а возьмёт значение, указанное по умолчанию.

```
.bordered(@color: #ccc) {
  border-top: dotted 1px @color;
  border-bottom: solid 2px @color;
}

article {
  .bordered();
  color: #443d3d;
}
```

Как результат

```
article {
  border-top: dotted 1px #cccccc;
  border-bottom: solid 2px #cccccc;
  color: #443d3d;
}
```

Пример с несколькими переменными

```
.flex(@dir, @con){
      display: flex;
     flex-direction: @dir;
     justify-content: @con;
}


.menu{
      .flex(row,space-between);
}
```

Функции

Функция — представляет из себя именованную часть кода программы, к которой можно обратиться из другого места программы.