VARIABLES

La declaración de una variable se asemeja mucho a la declaración de una propiedad de CSS normal, `<variable>: <expression>`. Puedes usar esa variable dónde tú quieras.

Ejemplo:

```
$base-color: #c6538c;
$border-dark: rgba($base-color, 0.88);

.alert {
border: 1px solid $border-dark;
}
```

NESTING

SCSS te permite anidar selectores y clases de la siguiente manera:

```
nav {
      ul {
            margin: 0;
            padding: 0;
            list-style: none;
      }

      li {
            display: inline-block;
      }

      a {
            display: block;
            padding: 6px 12px;
            text-decoration: none;
      }
}
```

Muchas propiedades de CSS tienen el mismo prefijo, por ejemplo `font-family`, `font-weight` y `font-size`. SCSS nos permite agruparlas anidándolas así:

```
font: {
      family: Helvetica, sans-serif;
      size: 18px;
      weight: bold;
}

text: {
      align: center;
      transform: lowercase;
      overflow: hidden;
}
```

@MIXIN

Gracias a `@mixin` podemos crear bloques de código CSS que necesitemos repetir múltiples veces de manera que podamos reutilizarlo. La sintáxis es la siguiente:

```
@mixin name {
      property: value;
      property: value;
      ...
}
```

Este sería un ejemplo de uso:

```
@mixin important-text {
      color: red;
      font-size: 25px;
      font-weight: bold;
      border: 1px solid blue;
}
```

Para usar nuestro mixin recurrimos al `@include`. Esta es su sintáxis:

```
selector {
      @include mixin-name;
}
```

Y este sería el ejemplo de uso:

```
.danger {
      @include important-text;
      background-color: green;
}
```

➔ Podemos incluir mixins dentro de otros.

Los mixins aceptan argumentos, de esta forma podemos pasarles variables. Por ejemplo:

```
/* Define un mixin con dos argumentos */
@mixin bordered($color, $width) {
      border: $width solid $color;
}

.myArticle {
      @include bordered(blue, 1px); // Llama al mixin con dos
argumentos.
}
```

También se pueden definir valores por defecto para los argumentos, de esta forma solo hay que pasarle el valor que queremos que cambie:

```
@mixin bordered($color: blue, $width: 1px) {
      border: $width solid $color;
}

.myTips {
      @include bordered($color: orange);
}
```

@EXTEND

El `@extend` nos permite compartir un conjunto de CSS de un selector a otro.

Esto es útil cuando nos encontramos con dos elementos que son prácticamente idénticos a nivel de estilos pero que cambian en algunos detalles.

Un ejemplo de este caso:

```
.button-basic {
	border: none;
	padding: 15px 30px;
	text-align: center;
	font-size: 16px;
	cursor: pointer;
}

.button-report {
	@extend .button-basic;
	background-color: red;
}

.button-submit {
	@extend .button-basic;
	background-color: green;
	color: white;
}
```