Estructuras de control

Módulo 03

Bucles

Implementación

SASS

Para comenzar con estructuras de control, vamos a primero generar nuestro for,

```
@for $i from 1  through 3 {  }
```

Luego vamos a asignar los valores también generando de forma dinámica una clase,

```
@for $i from 1  through 3 {

.seccion-#{$i * 1}{}

}
```

Implementación

SASS

Al código anterior le vamos a sumar una propiedad en este caso elegimos width, y la **multiplicamos en cada repetición por 30%**,

```
@for $i from 1  through 3 {

.seccion-#{$i * 2}{

width: $i * 30%;

}

}
```

Implementación

SASS

El [resultado anterior], en nuestro **estilos.css** luego de compilado será,

```
.seccion-2 {
  width: 30%;
}

.seccion-4 {
  width: 60%;
}

.seccion-6 {
  width: 90%;
}
/*# sourceMappingURL=estilos.css.map */
```

Implementación

SASS

Vamos a modificarlo **para lograr algo más razonable por ejemplo cubrir el 100% y que las secciones sean consecutivas,**

```
@for $i from 1  through 3 {

.seccion-#{$i * 1}{

  width: $i * 33.33%;

  }

}
```

Implementación

SASS

El resultado luego de **compilador será,**

```
.seccion-1 {
  width: 33.33%;
}

.seccion-2 {
  width: 66.66%;
}

.seccion-3 {
  width: 99.99%;
}
/*# sourceMappingURL=estilos.css.map */
```

Implementación

sass

Empecemos a trabajar!

El código anterior funciona, en una estructura pero no de forma responsiva. Entonces vamos a **suponer que cambiamos el width el contenedor principal,**

```
main {
  width:100%;
}

#seccion1 {
  width: 400px;
}

#seccion2 {
width: 500px;

}
```

Implementación

sass

En el caso anterior, debemos entonces **calcular a cuánto sería equivalente cada una de las columnas** para poder trabajarlas porcentualmente. Sin embargo **sass** nos permite hacer el cálculo directamente en nuestro **.scss,**

```
}

#seccion1 {
  width: 400px/900px * 100%;
}

#seccion2 {
width: 500px/900px * 100%;

}
```

Implementación

sass

El resultado del código anterior será luego de compilado,

```
main {
  width: 100%;
}

#seccion1 {
  width: 44.44444%;
}

#seccion2 {
  width: 55.55556%;
}
```

Condicionales

Implementación

SASS

En este preprocesador también **podemos utilizar condicionales,**

```
p {

@if($width == 200px){

background-color: blue;

}

    }
```

Implementación

SASS

El código anterior podemos trabajarlo si por ejemplo contamos con el siguiente aditamento,

```
$width: 200px;

p { @if($width == 200px){

background-color : blue;}


}
```

,

Implementación

¿Cómo empezar a trabajar?

El código anterior en una estructura en nuestro **index.html,**

```
<p> Parrafo de 200px de ancho </p>
```

Dará el siguiente resultado,

Parrafo de 200px de ancho

Implementación

¿Cómo empezar a trabajar?

El paso anterior nos permite trabajar con un **condicional simple** pero también podemos **utilizar else,**

```
p { @if($width == 200px){
background-color : blue;}
@else { background-color: green;}

}
```

Implementación

sass

También podemos utilizar un condicional **con else if,**

```
p { @if($width == 200px){

background-color : blue;}


@else if($width ==600px){

background-color: gray;
}

@else { background-color: green;}


}
```

Implementación

sass

También podemos trabajar con condicionales y errores con @error.

```
@mixin reflexive-position($property, $value) {
  @if $property != left and $property != right {
    @error "Property #{$property} must be either left or right.";
  }

  $left-value: if($property == right, initial, $value);
  $right-value: if($property == right, $value, initial);

  left: $left-value;
  right: $right-value;
  [dir=rtl] & {
    left: $right-value;
    right: $left-value;
  }
}

.sidebar {
  @include reflexive-position(top, 12px);
  //        ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  // Error: Property top must be either left or right.
}
```

For each

Implementación

sass

También podemos trabajar con for each de la siguiente forma,

```
$arreglo-colores:(
  gris1: #ccc,
  gris2: #aeaeae,
  gris3: #a4a4a4
);
@each $color, $valor in $arreglo-colores {
  .fondo-#{$color} {
    background: $valor;
  }
}
```

Implementación

sass

Al momento de compilarlo obtendremos el siguiente resultado,

```
.fondo-gris1 {
  background: #ccc;
}

.fondo-gris2 {
  background: #aeaeae;
}

.fondo-gris3 {
  background: #a4a4a4;
}
```

While

Implementación

sass

También SASS nos permite trabajar con **@while** como estructura de control, en este ejemplo tomado de la página oficial de **SASS.**

SCSS Sass

```scss
/// Divides `$value` by `$ratio` until it's below `$base`.
@function scale-below($value, $base, $ratio: 1.618) {
  @while $value > $base {
    $value: $value / $ratio;
  }
  @return $value;
}

$normal-font-size: 16px;
sup {
  font-size: scale-below(20px, 16px);
}
```

CSS

```css
sup {
  font-size: 12.36094px;
}
```

Revisión

- Repasar los conceptos vistos de **sass.**
- Trabajar con el **bonus track** debajo de todo para practicar los **elementos individualmente.**
- Ver todos los videos y materiales necesarios antes de continuar

¡Muchas gracias!

¡Sigamos trabajando!