



# Less & Sass

---

CSS inteligente

# ¿Qué es Less.js?

---

- Less es un preprocesador para CSS.
- Nos permite trabajar con hojas de estilo con funcionalidades de un lenguaje de programación.
- Existen otros preprocesadores como SASS o Stylus, aunque sólo LESS.js se puede integrar desde el front.



# Sus principales ventajas son:

---

- Nos permite ser más productivos
- Ayuda a organizar mejor nuestras hojas de estilo y a la compatibilidad entre navegadores.
- El código que generamos es reutilizable

# Instalación y Uso

---

# Instalación

---

- Descargar el archivo de <http://lesscss.org/>
- Cargar dentro del head de la página el archivo .css o .less. No importa la extensión pero hay que indicar el atributo 'rel' como stylesheet/less

<head>

<link rel="stylesheet/less" type="text/css" href="estilo.less">

<link rel="stylesheet/less" type="text/css" href="estilo.css">

<script type="text/javascript" src="js/less-1.2.1.min.js"></script>

</head>

No es  
recomendable  
usarlo así en  
producción

A large, stylized feather graphic in a light beige color, positioned on the left side of the slide. It has a central rachis with many fine, radiating barbs, giving it a delicate, organic appearance.

# Conceptos fundamentales

---

- Variables
- Operaciones
- Anidamiento
- Mixins
- Funciones

# Variables

---



# Las variables se definen de la siguiente forma:

---

@size: 100px;

@color: #ff3;

@ruta: 'images/background.gif';



# Variables globales y locales

---

Global



```
@colorbase: red;
```

```
body{
```

Local



```
@colorbase: blue;
```

```
color: @colorbase;
```

```
}
```

```
body{  
  color: blue;  
}
```





# Nota:

---


Las variables del tipo *string* se pueden embeber dentro de otra cadena, utilizando llaves:

```
@rutabase: "https://less.org/";
```

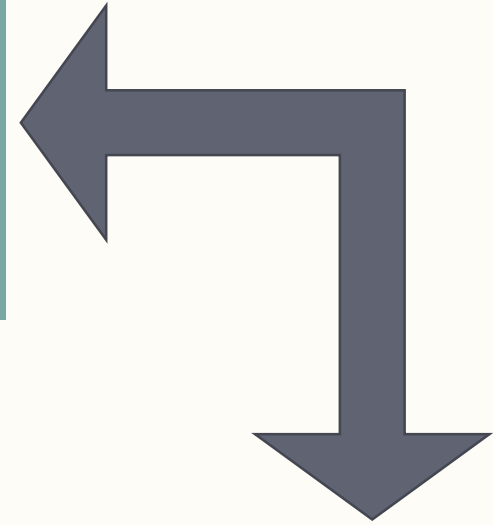
```
background-image: url("@{rutabase}/images/fondo.png");
```

# Operaciones

---



```
@ancho: 100px;  
div{  
    width: @ancho * 2;  
}
```




```
div{  
    width: 200px;  
}
```

Jerarquía de  
operadores:  
\*, /, +, -

# Alteración de jerarquía

---



```
@distancia: 10px;  
@size: @distancia / (8 - 3);
```



```
@size: 2px;
```

# Nota: sentencias shorthand

---

```
@distancia: 10px;  
margin: @distancia @distancia / 2;
```



```
margin: 10px 5px;
```


No olvidar  
separar por  
espacios



# Anidamiento & Concatenación

---

# Anidamiento de clases



```
body{  
  width: 100px;  
  article{  
    width: 200px;  
  }  
}
```



```
body{ width: 100px; }  
body article{width: 100px; }
```





# Concatenación de clases

---

Para indicar que concatenamos en lugar de anidar usamos el  
operador **&**


```
div{  
  width: 100px;  
  &.clase{  
    background: red;  
  }  
}
```

```
div{width: 100px;}  
div.clase{  
  width: 100px;  
  background: red;  
}
```

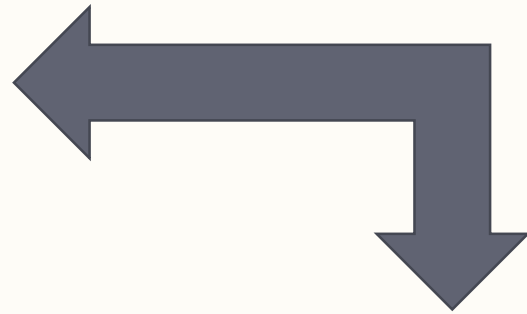
# Concatenar pseudoclasses

:hover, :active, :visited

---



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



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

# Mixins

---

Funciones en CSS


A large, light green feather graphic is positioned on the left side of the slide, extending from the bottom towards the top. It has a central rachis with many fine barbs branching out.

# Mixins

---

Los **Mixins** son conjuntos de reglas que se pueden anidar dentro de otras reglas.

Su comportamiento es similar al de las funciones de otros lenguajes de programación.



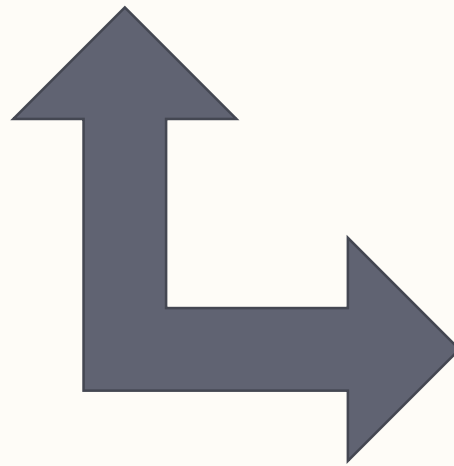
```
.derecha{  
  float: right;  
  text-align: right;  
}
```

Regla I

```
div{  
  width: 100px;  
  height: auto;  
  .derecha;  
}
```


Regla I

Regla II aplicando  
conjuntamente la  
Regla I



```
div{  
  width: 100px;  
  height: auto;  
  float: right;  
  text-align: right;  
}
```

# Mixins con parámetros




```
.rounded-corners (@radius: 5px) {  
  -webkit-border-radius: @radius;  
  -moz-border-radius: @radius;  
  -ms-border-radius: @radius;  
  -o-border-radius: @radius;  
  border-radius: @radius; }
```

Definimos el Mixin para que reciba un parámetro.

```
#header { .rounded-corners; }  
#footer { .rounded-corners(10px); }
```

Aplicamos la regla dentro del div, sin pasarle parámetros al Mixin.

Aplicamos la regla dentro del div, pasándole el parámetro al Mixin.

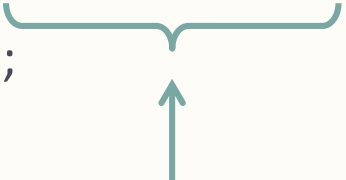


```
#header {  
    -webkit-border-radius: 5px;  
    -moz-border-radius: 5px;  
    -ms-border-radius: 5px;  
    -o-border-radius: 5px;  
    border-radius: 5px; }  
  
#footer {  
    -webkit-border-radius: 10px;  
    -moz-border-radius: 10px;  
    -ms-border-radius: 10px;  
    -o-border-radius: 10px;  
    border-radius: 10px;  
}
```

CSS  
Compilado

# Mixins multiparámetro

```
.borderradius(@a:0px, @b:0px, @c:0px, @d:0px){  
  border-radius: @a @b @c @d;  
}
```



Tener en cuenta que *Less*,  
diferencia las unidades.

Nota: pueden enviar y recibir varios parámetros siempre que en la definición del mixin se separen por comas.



# Funciones de color

---



```
lighten(@color, 30%);  
darken(@color, 30%);
```

- ***lighten***: para aclarar un color.
- ***darken***: para oscurecerlo.
- ***saturate***: para saturarlo, o “aumentar el color”.
- ***desaturate***: para desaturarlo, o “reducir el color”.
- ***fadein***: para resaltarlo quitándole transparencia.
- ***fadeout***: para disimularlo usando transparencia.
- ***fade***: para cambiar la transparencia a 50%.
- ***spin***: para cambiar el tono de color.
- ***mix***: para mezclar dos colores.

Las funciones de color son funciones pre-definidas de Less CSS que permiten alterar un color, para hacerlo más claro, oscuro, saturado, cambiarle la tonalidad, etc.