**Sass**

1. Store Data with Sass Variables

<style type='text/sass'>

$text-color: red;

.header{

text-align: center;

}

.blog-post, h2 {

color: $text-color;

}

</style>

<h1 class="header">Learn Sass</h1>

<div class="blog-post">

<h2>Some random title</h2>

<p>This is a paragraph with some random text in it</p>

</div>

<div class="blog-post">

<h2>Header #2</h2>

<p>Here is some more random text.</p>

</div>

<div class="blog-post">

<h2>Here is another header</h2>

<p>Even more random text within a paragraph</p>

</div>

2. Nest CSS with Sass

<style type='text/sass'>

.blog-post {

h1 {

text-align: center;

color: blue;

}

p {

font-size: 20px;

}

}

</style>

<div class="blog-post">

<h1>Blog Title</h1>

<p>This is a paragraph</p>

</div>

3. Create Reusable CSS with Mixin

<style type='text/sass'>

@mixin border-radius($radius) {

-webkit-border-radius: $radius;

-moz-border-radius: $radius;

-ms-border-radius: $radius;

border-radius: $radius;

}

#awesome {

width: 150px;

height: 150px;

background-color: green;

@include border-radius(15px);

}

</style>

<div id="awesome"></div>

4. Use @if and @else to Add Logic To Your Styles

<style type='text/sass'>

@mixin border-stroke($val) {

@if $val == light {

border: 1px solid black;

}

@else if $val == medium {

border: 3px solid black;

}

@else if $val == heavy {

border: 6px solid black;

}

@else {

border: none;

}

}

#box {

width: 150px;

height: 150px;

background-color: red;

@include border-stroke(medium);

}

</style>

<div id="box"></div>

**5. Use @for to Create a Sass Loop**

<style type='text/sass'>

@for $j from 1 to 6 {

.text-#{$j} {font-size: 10px \* $j};

}

</style>

<p class="text-1">Hello</p>

<p class="text-2">Hello</p>

<p class="text-3">Hello</p>

<p class="text-4">Hello</p>

<p class="text-5">Hello</p>

6. Use @each to Map Over Items in a List

<style type='text/sass'>

@each $color in blue, black, red {

.#{$color}-bg {background-color: $color;}

}

div {

height: 200px;

width: 200px;

}

</style>

<div class="blue-bg"></div>

<div class="black-bg"></div>

<div class="red-bg"></div>

\*\*\*A map has slightly different syntax. Here's an example:

$colors: (color1: blue, color2: red, color3: green);  
  
@each $key, $color in $colors {  
  .#{$color}-text {color: $color;}  
}

7. Apply a Style Until a Condition is Met with @while

<style type='text/sass'>

$t: 1;

@while $t < 11 {

.text-#{$t} {

font-size: 5px \* $t;

}

$t: $t + 1;

}

</style>

<p class="text-1">Hello</p>

<p class="text-2">Hello</p>

<p class="text-3">Hello</p>

<p class="text-4">Hello</p>

<p class="text-5">Hello</p>

<p class="text-6">Hello</p>

<p class="text-7">Hello</p>

<p class="text-8">Hello</p>

<p class="text-9">Hello</p>

<p class="text-10">Hello</p>

8. Split Your Styles into Smaller Chunks with Partials

// The main.scss file

@import 'variables'

9. Extend One Set of CSS Styles to Another Element

<style type='text/sass'>

h3{

text-align: center;

}

.info{

width: 200px;

border: 1px solid black;

margin: 0 auto;

}

.info-important {

@extend .info;

background-color: magenta;

}

</style>

<h3>Posts</h3>

<div class="info-important">

<p>This is an important post. It should extend the class ".info" and have its own CSS styles.</p>

</div>

<div class="info">

<p>This is a simple post. It has basic styling and can be extended for other uses.</p>

</div>