6 MATH + COLOR

Number Operations

- `+` addition
- `-` subtraction
- `*` multiplication
- `/` division
- `%` modulo

Modulo = **remainder** from a division operation. `12 % 3` results in `0`, while `12 % 5` returns `2`.

Assembly Tip

Sass defaults to returning (up to) five digits after a decimal point.

Division

- The trickiest of the number operations, due to `font`:

```
font: normal 2em/1.5 Helvetica, sans-serif;
```


Triggering Division

- Variable involved - `$size / 10`
- Parenthesis - `(100px / 20)`
- Another arithmetic operation - `20px * 5 / 2`

String Addition

Addition on strings concatenates them:

```
$family: "Helvetica " + "Neue";  // "Helvetica Neue"
```

Initial left-side string determines post-concatenation quotes:

```
$family: 'sans-' + serif  // 'sans-serif'
$family: sans- + 'serif'  // sans-serif
```


If the units differ, Sass attempts combination:

application.scss

```
h2 {
  font-size: 10px + 4pt;
}
```

application.css

```
h2 {
  font-size: 15.33333px;
}
```

Incompatible units will throw an error:

application.scss

```
h2 {
  font-size: 10px + 4em;
}
```

application.css

```
Incompatible units: 'em'
and 'px'.
```

Pre-Defined Math Utilities

- `round($number)` – round to closest whole number
- `ceil($number)` – round up
- `floor($number)` – round down
- `abs($number)` – absolute value
- `min($list)` – minimum list value
- `max($list)` – maximum list value
- `percentage($number)` – convert to percentage

Called the same way as custom functions:

application.scss

```
h2 {
  line-height: ceil(1.2);
}
```

application.css

```
h2 {
  line-height: 2;
}
```

`percentage()` replaces our custom `fluidize()`:

application.scss

```
.sidebar {
  width: percentage(350px/1000px);
}
```

application.css

```
.sidebar {
  width: 35%;
}
```

`percentage()` replaces our custom `fluidize()`:

application.scss

```
$context: 1000px;

.sidebar {
  width: percentage(450px/$context);
}
```

application.css

```
.sidebar {
  width: 45%;
}
```

Color Juggling

- Easier recall through variables
- Simplified alteration via **color utility functions**
- Faster representation using **shorthand**

application.scss

```
$color-base: #333333;

.addition {
  background: $color-base + #112233;
}
.subtraction {
  background: $color-base - #112233;
}
.multiplication {
  background: $color-base * 2;
}
.division {
  background: $color-base / 2;
}
```

application.css

```
.addition {
  background: #445566;
}
.subtraction {
  background: #221100;
}
.multiplication {
  background: #666666;
}
.division {
  background: #191919;
}
```

6.4 Math + Color

Assembly Tip

Where possible, use color utility functions instead of color arithmetic: easier to predict and maintain.

application.scss

```
$color: #333333;

.alpha {
  background: rgba(51,51,51,0.8);
}
```

manually finding the rgb value of a color we already have stored

application.css

```
.alpha {
  background: rgba(51,51,51,0.8);
}
```

6.5 Color Shortcuts

application.scss

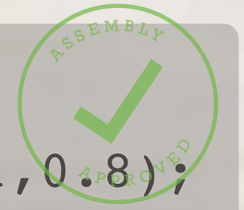
```scss
$color: #333333;

.alpha {
  background: rgba($color,0.8);
}
.beta {
  background: rgba(#000,0.8);
}
```

can also use hex values where appropriate

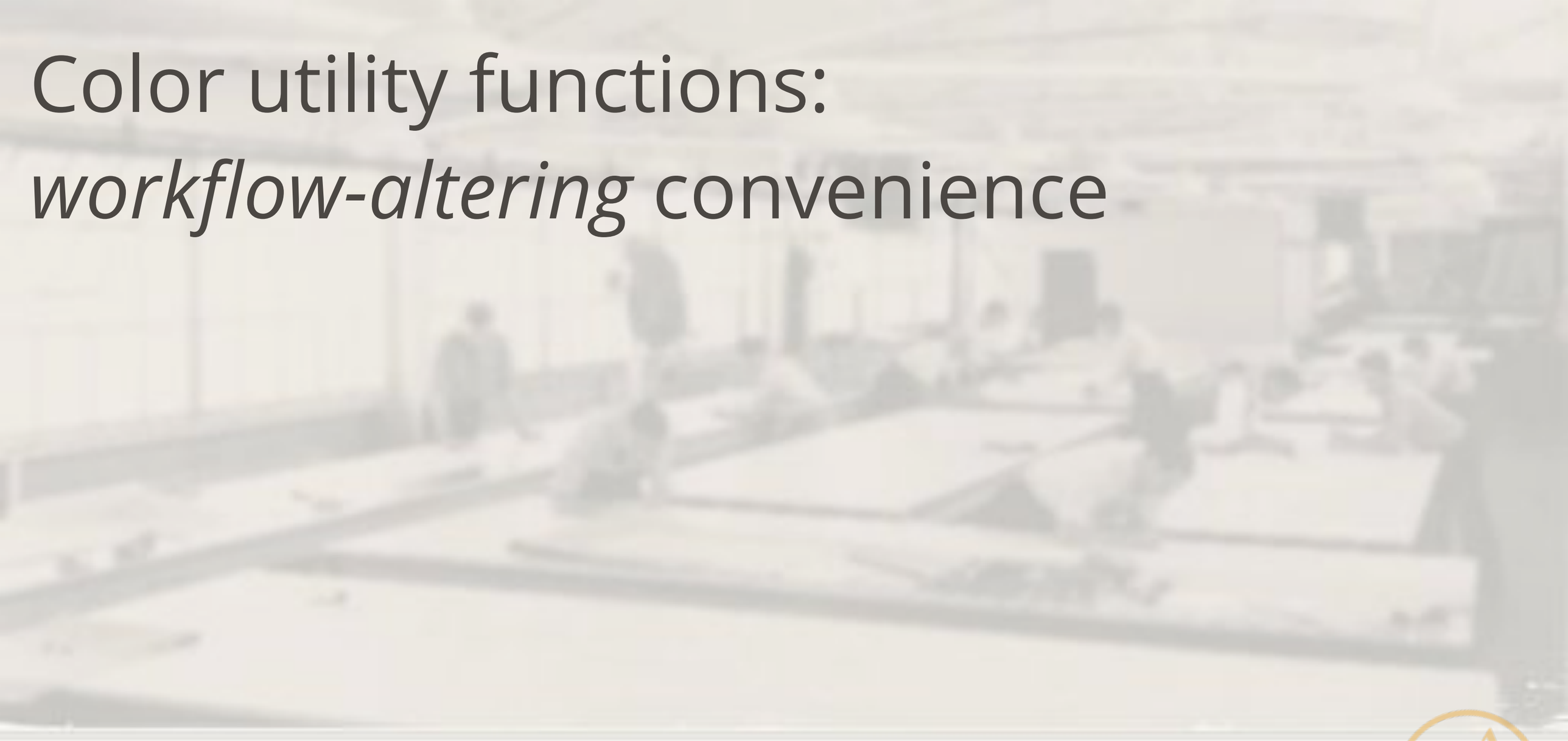
application.css

```css
.alpha {
  background: rgba(51,51,51,0.8);
}
.beta {
  background: rgba(0,0,0,0.8);
}
```



6.5 Color Shortcuts

Color utility functions: *workflow-altering* convenience



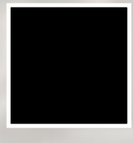
application.scss

```
$color: #333;

.lighten {
  color: lighten($color, 20%);
}
.darken {
  color: darken($color, 20%);
}
```

application.css

```
.lighten {
  background: #666666;
}
.darken {
  background: black;
}
```



6.6 Color Functions

application.scss

```
$color: #87bf64;

.saturate {
  color: saturate($color, 20%);
}
.desaturate {
  color: desaturate($color, 20%);
}
```

application.css

```
.saturate {
  background: #82d54e;
}
.desaturate {
  background: #323130;
}
```

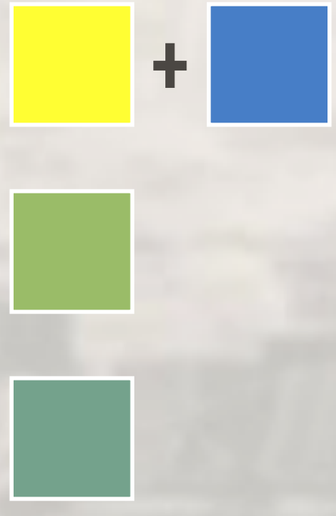
6.6 Color Functions

application.scss

```
.mix-a {
  color: mix(#ffff00, #107fc9);
}
.mix-b {
  color: mix(#ffff00, #107fc9, 30%);
}
```

application.css

```
.mix-a {
  background: #87bf64;
}
.mix-b {
  background: #57a58c;
}
```



6.6 Color Functions

application.scss

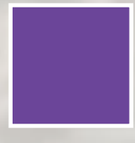
```
$color: #87bf64;

.grayscale {
  color: grayscale($color);
}
.invert {
  color: invert($color);
}
.complement {
  color: complement($color);
}
```

application.css

```
.grayscale {
  color: #929292;
}
.invert {
  color: #78409b;
}
.complement {
  color: #9c64bf;
}
```






6.6 Color Functions

Assembly Tip

But wait, there's more!

http://sass-lang.com/docs/yardoc/Sass/Script/Functions.html


ASSEMBLING SASS
PROPERTY OF:
CODE SCHOOL
PROJECT DATE:
SEP 2012
STANDARD ISSUE
NO 18
CODE SCHOOL
ASSEMBLY