wantsome
The friendly IT Academy

Saptamana 4

Partea 2

Programare Front-End

1. CSS Preprocessors

“Preprocesor”

- program care ne permite sa generam cod CSS pe baza unui cod scris intr-o sintaxa specifica
- exista o varietate de “preprocesoare”
- le folosim pentru capabilitatile si functionalitatile pe care le ofera si care nu exista in CSS implicit: posibilitatea reutilizarii codului prin intermediul declararii unor variabile si mixin-uri, mostenirea selectorilor, extensii, combinare, etc…
- functionalitatile oferite in plus fac codul mult mai usor de citit, mentinut si cresc rapiditatea dezvoltarii, in special atunci cand vine vorba despre aplicatii de dimensiuni mari

Referinte: *positioning explained*

SASS - “CSS with superpowers”

- Avem 2 posibilitati pentru a alege una din cele doua sintaxe disponibile specifice SASS :

SCSS(.scss)

```scss
$background-prime-color: #ffabff;
.myClass {
  background-color: $background-prime-color;
  width: 200px;
  height: 200px;
  h1 {
    color: $background-prime-color;
  }
}
```

SASS(.sass)

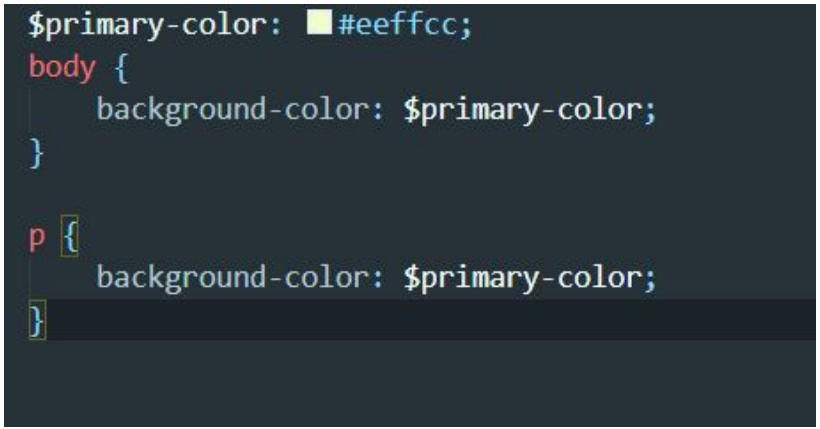
```sass
$background-prime-color: #ffabff
.myClass
  background-color: $background-prime-color
  width: 200px
  height: 200px
    h1
      color: $background-prime-color
```

SASS - functionalitati

Variables

- se foloseste simbolul $ pentru a declara o variabila care retine in memorie o anumita valoare

  ex: $primary-color: #eeffcc;



SASS

Nesting

CSS OUTPUT

```
nav ul {
  margin: 0;
  padding: 0;
  list-style: none;
}
nav li {
  display: inline-block;
}
nav a {
  display: block;
  padding: 6px 12px;
  text-decoration: none;
}
```

SCSS SYNTAX

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

  li { display: inline-block; }

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

SASS SYNTAX

```
nav
  ul
    margin: 0
    padding: 0
    list-style: none

  li
    display: inline-block

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

SASS

Partials

- Putem face fisiere partiale folosing o sintaxa: _[partial-file-name]**.scss**
- Preprocesorul SASS va sti ca fisierele care incep cu _ (underscore) nu vor trebui transformate in fisier .css
- Folosim aceasta tehnica pentru modularizarea fisierelor SASS: *arhitectura mai buna, cod mai usor de urmarit*

SASS

Imports

- Sintaxa folosita este **@import 'nume-fisier'**
- Cu ajutorul *import*-ului incarcam unul sau mai multe fisiere partiale intr-un singur fisier ce va fi preprocesat in fisier CSS - ulterior utilizat intr-o pagina .html a aplicatiei noastre

```scss
// _reset.scss

html,
body,
ul,
ol {
  margin:  0;
  padding: 0;
}
```

```scss
// base.scss

@import 'reset';

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef;
}
```

```css
html,
body,
ul,
ol {
  margin:  0;
  padding: 0;
}

body {
  font: 100% Helvetica, sans-serif;
  background-color: #efefef;
}
```

SASS

Mixins

- Un *mixin* incapsuleaza si retine mai multe proprietati care pot fi reutilizate ulterior

```scss
@mixin abutton {
    a {
      background-color: blue;
      color: white;
      radius: 3px;
      margin: 2px;

      &:hover {
        color: red;
      }

      &:visited {
        color: green;
      }
    }
  }

@include abutton();
 .menu-button {
  @include abutton;
}
```

```css
.menu-button a {
  background-color: blue;
  color: white;
  radius: 3px;
  margin: 2px;
}

.menu-button a:hover {
  color: red;
}

.menu-button a:visited {
  color: green;
}
```

SASS

Mixins Parameters and Variables

```
@mixin a-button($base, $hover, $link) {
    a {
        background-color: $base;
        color: white;
        radius: 3px;
        margin: 2px;

        &:hover {
            color: $hover;
        }

        &:visited {
            color: $link;
        }
    }
}

.menu-button {
    @include a-button(blue, red, green);
}
.text-button {
    @include a-button(yellow, black, grey);
}
```

https://medium.freecodecamp.org/sass-mixins-and-loops-171122499a2

SASS

Extend/Inheritance

```
%message-shared {
  border: 1px solid #ccc;
  padding: 10px;
  color: #333;
}
```

```
.message {
  @extend %message-shared;
}

.success {
  @extend %message-shared;
  border-color: green;
}

.error {
  @extend %message-shared;
  border-color: red;
}
```

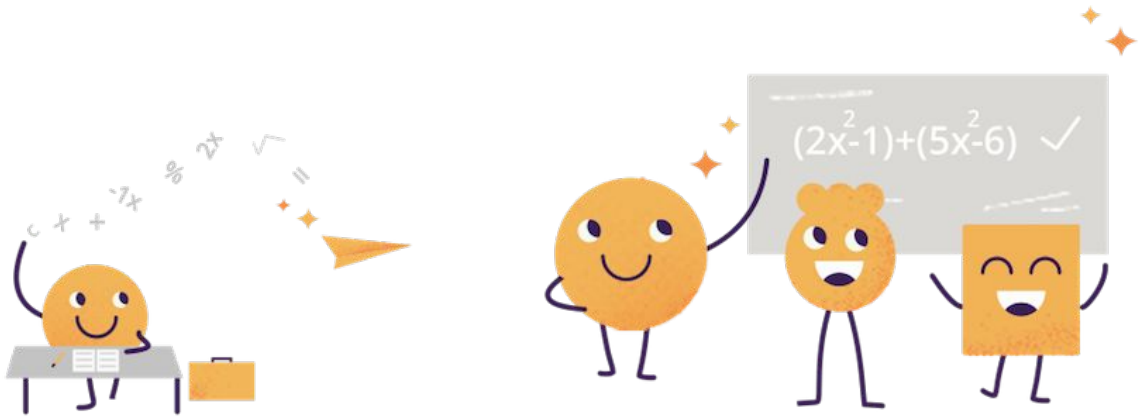
SASS

Operators

```
article[role="main"]
  float: left
  width: 600px / 960px * 100%
```

```
article[role="main"] {
  float: left;
  width: 62.5%;
}
```

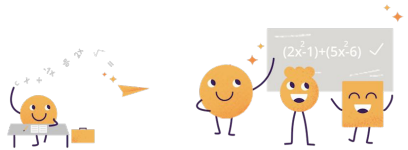
PRACTICE: **SASS**



PRACTICE: SASS

Cerinte:

1. *Instalati extensia **live-sass-compiler** pentru editorul vostru*
2. *Scrieti cate un fisier pentru fiecare feature din SASS (variable.scss, nesting.scss, mixins.scss, inheritance.scss, operators.scss)*
3. *Fiecare fisier trebuie sa demonstreze (aplice) functionalitatea pe care o reprezinta - exemplu: pentru _variables.scss, declarati o variabila si in cadrul aceluiasi fisier declarati o clasa in cadrul careia sa o folositi*
4. *Folosind import, creati un fisier principal care sa includa toate modulele*
5. *Utilizati foaia de stiluri generata pentru a aplica proprietatile definite, intr-o pagina HTML*



wantsome
The friendly IT Academy

2. Animations with CSS

2.1. CSS Transitions

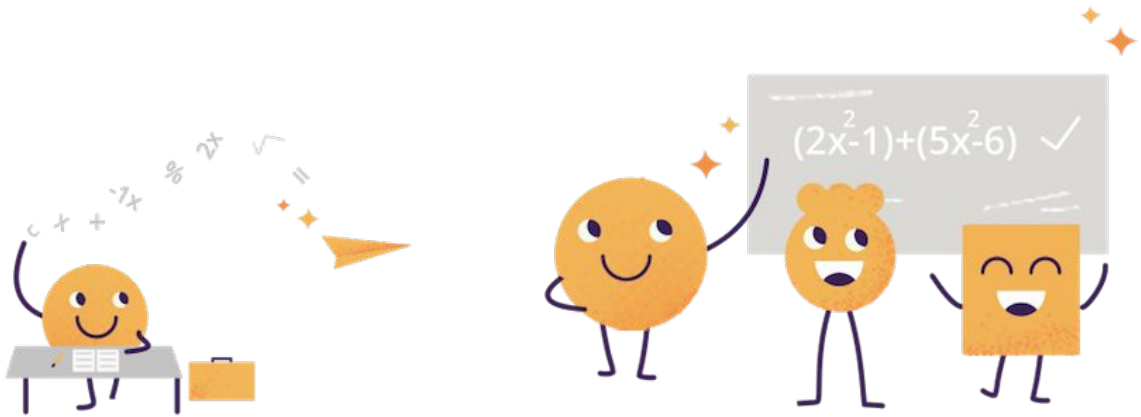
CSS TRANSITIONS

- o tranzitie se petrece atunci cand o valoare a unei proprietati tranzitioneaza catre o alta valoare iar aceasta actiune se petrece vizibil
- Avem 4 caracteristici specifice unei tranzitii:
    1. **Transition-property** (ce proprietate sa fie ‘transformata’)
    2. **Transition-duration** (time)
    3. **Transition-timing-function** (linear, ease, ease-in, ease-out, ease-in-out)
    4. **Transition-delay** (time)

  Se pot specifica separat sau in cadrul aceleiasi proprietati *transition*

  *Ex: **transition**: **background-color 1s ease 2s**;*

PRACTICE: **Transitions**



PRACTICE: Transitions

Cerinte:

1. https://codepen.io/oviduzz/pen/aMqJoZ (**http://bit.do/exAnimation**)
2. https://codepen.io/oviduzz/pen/bZLqLP (**http://bit.do/exAnimation2**)

wantsome
The friendly IT Academy

2.2. CSS Animations

CSS Animations

- O animatie schimba forma vizuala a unui element, de la o stare la alta

- Starile unui element la un moment dat (stilurile aplicate) pot fi definite utilizand ***@keyframes***

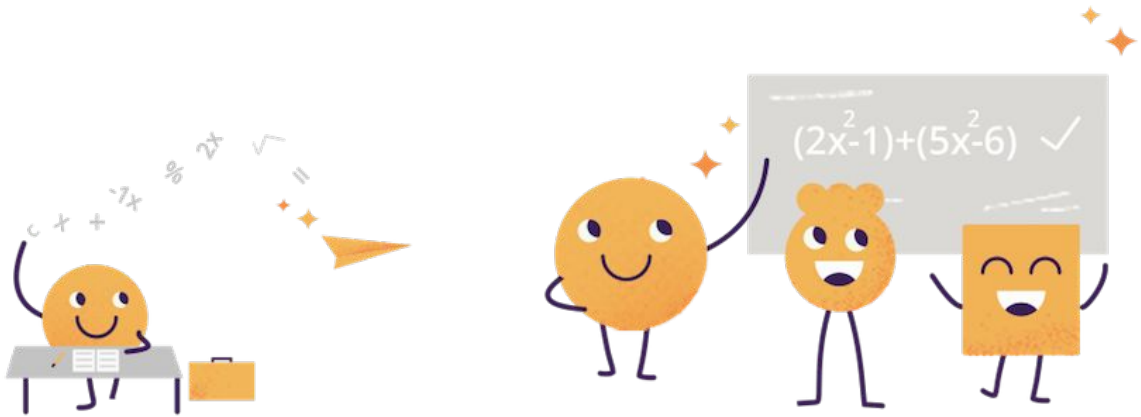
- Pentru ca o animatie sa functioneze aceasta trebuie atasata unui element html prin intermediul unui selector CSS

- Animatiile sunt o aternativa mai puternica pentru tranzitii

- Diferentele? - Tranzitiile merg de la *A la B* pe cand cu animatii putem face de la *A la B la C la D* etc.

- O animatie prezinta mai multe caracteristici specifice:
  animation-name / animation-duration / animation-timing-function / animation-delay /
  animation-iteration-count / animation-direction / animation-fill-mode/ animation-play-state

PRACTICE: **Transitions**



PRACTICE: Animations

Cerinte:

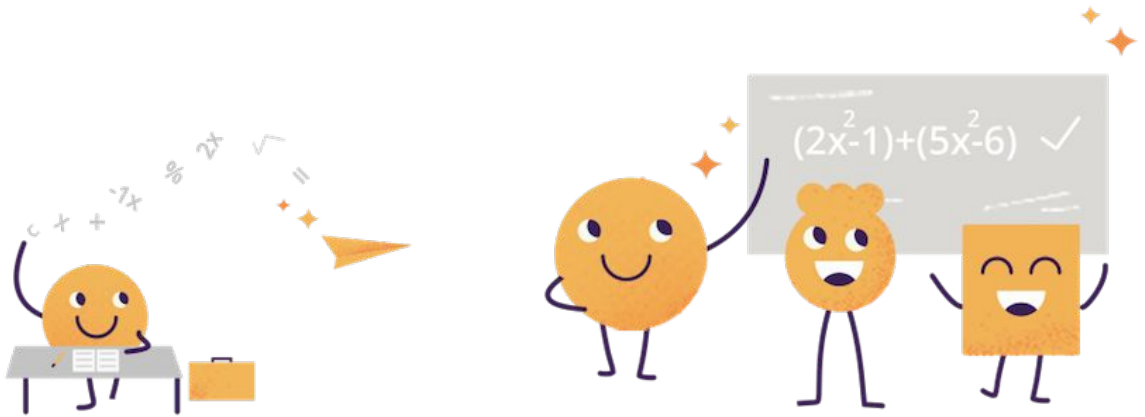
1. https://codepen.io/oviduzz/pen/LaQyWM (http://bit.do/exAnimation3)
2. https://codepen.io/oviduzz/pen/rRJmzd (http://bit.do/exAnimation4)

wantsome
The friendly IT Academy

Cool stuff

1. http://animista.net/play
2. https://codepen.io/patrikhjelm/pen/hItqn
3. https://codepen.io/mariosmaselli/pen/ghmwq
4. https://codepen.io/Maseone/pen/rGapf
5. https://codepen.io/drygiel/pen/KbhmA

PRACTICE: **BOOM ! “Implement a real design” time!**



https://ufile.io/13io4