# Web Technologies — Week 3

Mikheil Rukhaia

International Black Sea University,
Tbilisi, Georgia

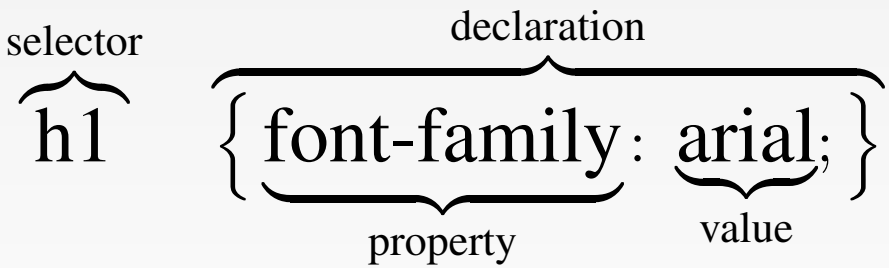
October 22, 2015

# Outline

# Introducing CSS

# Cascading Style Sheet

- Cascading Style Sheet, CSS, works by associating rules with elements, describing how the content of the elements should be displayed.
- Example:



# Rule components

- Selector: comma separated list of elements, to which declaration applies.
- Declaration: list of semi-colon separated pairs, setting how elements should be styled:
  - Property: affecting the property of the elements.
  - Value: comma separated list of specifications for the property.
- Example:

```
h1, h2, h3 {
  font-weight: bold;
  font-family: arial, verdana, sans-serif;
  color: #000000;
  background-color: #FFFFFF;
}
```

# Adding CSS to HTML

- External style sheet: separate `.css` file, linked to the document in the header using `<link />` element (preferable).
- Internal style sheet: CSS rules are listed inside a `<style>` element in the header of the document (usually used to override the external style sheet rules).
- Inline style rules: CSS rules are defined for concrete element via its `style` attribute (usually used to override the rules defined in either way).

## The <link> element

- <link /> element is contained inside <head> element and must contain rel, type and href attributes to specify external style sheet.
- In this case values for rel and type are fixed to stylesheet and text/css respectively.
- href must specify the URL to the .css file.
- Example:

```
<link rel="stylesheet" type="text/css"
                       href="style.css" />
```

# Advantages of external CSS

- Can be used for several pages on the site.
- Makes HTML documents shorter and less information is downloaded from the server (i.e. the site works faster).
- Just changing one file you can alter style of multiple pages.
- Different style files can be attached to the document for different displays.
- If browser does not support CSS, simply ignores external one, while displays content of the `<style>` element on the screen.

# Text Formatting

# Affecting text

- There are two ways to control how text appears in the browser.
- Directly affect the font and its appearance, or
- Have other formatting effects to the text.

# Font properties

| Property | Description |
|---|---|
| `font` | Combine several properties into one |
| `font-family` | Specified font must be installed on user's computer |
| `font-size` | Usually specified in `pt` or `px` |
| `font-weight` | `lighter`, `normal`, `bold` or `bolder` |
| `font-style` | `normal`, `italic`, or `oblique` (a slant version of normal font) |
| `font-variant` | `normal` or `small-caps` |
| `font-stretch` | Control the width of the actual letters in a font (not spaces between them) |
| `font-size-adjust` | Alter the aspect ratio of the size of characters of the font |

## Font families

- There are five general font families: `serif`, `sans-serif`, `monospace`, `cursive` and `fantasy`
- Serif has extra curls on letters, while sans-serif has straight ends to the letters.
- Monospace is a serif font having each letter of the same width.
- In printing serif is better readable, while on screen sans-serif is preferred.
- Cursive fonts are emulating handwriting and fantasy are decorative fonts.

# Font families (ctd.)

- In `font-family` property you should specify several fonts in case user does not have one installed.
- Font names containing spaces must be specified in double quotes.
- The property specification should end with one of the general families.
- Example:

```
p {
 font-family: times, "times new roman", serif;
}
```

# Font size

- Absolute size: `xx-small`, `x-small`, `small`, `medium`, `large`, `x-large` and `xx-large`
- Relative size: `smaller` and `larger`
- Length can be expressed in `px`, `em`, `ex`, `pt`, `pc`, `in`, `cm` and `mm`
- Percentage is calculated as a proportion of the element containing the text.
- Example:

```
p {
 font-family: times, "times new roman", serif;
 font-size: 1pc; /* 12pt */
}
```

# Text properties

| Property | Description |
|---|---|
| `color` | Either a hex code or a color name |
| `vertical-align` | `baseline` (default), `sub`, `super`, `top`, `middle` or `bottom` |
| `text-align` | `left`, `right`, `center` or `justify` |
| `text-decoration` | `line-through`, `underline` or `overline` |
| `text-indent` | Indent the first line of text within an element |
| `text-transform` | `none`, `capitalize`, `uppercase` or `lowercase` |

# Text properties (ctd.)

| Property | Description |
|---|---|
| `letter-spacing` | Either `normal` or a unit of length |
| `word-spacing` | Controls the amount of space between each word |
| `white-space` | `normal`, `pre` or `nowrap` |
| `direction` | `ltr`, `rtl` or `inherit` |
| `unicode-bidi` | Allows you to create bidirectional text |

- Example:

```
p {
  font-family: times, "times new roman", serif;
  font-size: 1pc; /* 12pt */
  color: #000000; /* black */
  text-align: justify;
  text-indent: 1em;
}
```

## Pseudo-classes

- There are two pseudo-classes in CSS related to text formatting.
- `first-letter` pseudo-class specifies a rule just for the first letter of an element.
- `first-line` pseudo-class allows to render the first line of an element differently from the rest.
- The name of the pseudo-class is separated from the element by a colon.
- Example

```
p:first-letter { font-size: 24pt; }
p:first-line { color: #0000ff; /* blue */ }
```

# Selectors

# Universal selector

- Universal selector is an asterisk `* {}`.
- Rules written using the universal selector apply to the whole document.
- Usually it is used to specify default values for the document.
- Example:

```
* {
  font-family: sans-serif;
  font-size: 1pc;
}
```

# Class selector

- Class selector is a point followed by the class name.
- Class selector can be used in two ways:
  - if no element is specified before, then it applies to all elements that carry `class` attribute with the class name as a value.
  - if element name is specified before, then it applies to this element only when `class` attribute has the value of the class name.
- Example:

```
.negative {
  color: #FFFFFF;
  background-color: #000000;
}
p.background {
  background-color: #00FF00;
}
```

# ID selector

- ID selector is a hash symbol followed by the ID name.
- ID selector is very similar to class selector, but works with `id` attribute.
- Since `id` must be unique in the document, the ID selector can be applied only in one element. (some browsers ignore this rule)
- Example:

```
#negative {
  color: #FFFFFF;
  background-color: #000000;
}
p#background {
  background-color: #00FF00;
}
```

## Child selector

- Child selector is a > symbol and matches an element that is a direct child of another.
- Drawback: only latest versions of the browsers support this selector.
- Example:

```
p>b {
  color: #0000FF;
  text-decoration: underline;
}
```

# Descendent selector

- Descendent selector is a space and matches an element that is a descendent of another.
- In contrast to child selector, descendent applies not only direct, but to all children of the element.
- Example:

```
table th {
  color: #FFFFFF;
  background-color: #111111;
  text-transform: capitalize;
  font-size: 14pt;
  vertical-align: middle;
  text-align: center;
}
```

# Adjacent sibling selector

- Adjacent sibling selector is a plus symbol and matches an element that is the next sibling of another.
- Drawback: only latest versions of the browsers support this selector.
- Example:

```
h1+p:first-letter { font-size: 2pc; }
```

# Attribute selectors

- Attribute selectors are written inside `[]` and matches to the element carrying the corresponding attributes and/or values.

- Drawback: last three selectors are only supported by the latest versions of some browsers.

| Name | Example |
|---|---|
| Existence | `p[id]` |
| Equality | `p[id="summary"]` |
| Space | `p[class~="XHTML"]` |
| Hyphen | `p[lang\|="en"]` |
| Prefix | `p[attr^="b"]` |
| Substring | `p[attr*="on"]` |
| Suffix | `p[attr$="x"]` |

# The Box Model

# Box model

- Every element is treated as a box.
- Each box has border, margin and padding.
- Border separates the edge of the box from other boxes.
- Margin is the distance between the edge of a box and the box next to it.
- Padding is the space between the content of the box and its border.

# Box model (ctd.)

- It is possible to individually control the top, bottom, left, and right border, margin, and padding of each box.
- Margin and padding are important in creating white spaces between parts of the page.
- Important: when a bottom margin of one element meets the top margin of another, only the larger of the two will show.

## Border

- `border-color` indicates the color of a border.
- Use
  `border-bottom-color`,
  `border-top-color`,
  `border-left-color`,
  `border-right-color`
  to individually change respective border parts.
- Note: each property we consider later can be applied to individual parts using the above scheme.

## Border (ctd.)

- `border-style` specifies the style of the borderline.
- Values are: `none`, `solid`, `dashed`, `double`, `dotted`, `inset`, `outset`.
- `border-width` specifies the width of the border and can be either length or `thin`, `medium`, `thick`
- Example:

```
p { border: solid 3px blue; }
.dotted {
  border-style:dotted;
  border-color:#999999;
  border-width:2px;
}
```

## Margin and Padding

- `margin` and `padding` can be a length, percentage or `inherit`
- Percentage is relative to the containing box.
- Similar to border, it is possible to change margin and padding only to specific parts.
- Example:

```
em { margin-right: 5px; padding-left: 5px; }

td { margin: 5%; padding: 10%; }
```

## Dimensions

- `height` and `width` can be used to specify box dimensions via length, percentage or `auto` (default).
- `min-height`, `min-width`, `max-height` and `max-width` specify minimal and maximal dimensions, respectively, via length or percentage.
  (only supported by the latest browser versions)
- `line-height` specifies the space between lines of text via length or percentage.
- `overflow` deals with the situations when content cannot be fitted into a box; values can be `hidden` or `scroll`

# Advanced Formatting

## Links

- `link` pseudo-class styles links in general.
- `visited` pseudo-class styles links that have already been visited.
- `active` pseudo-class styles links that are currently active (being clicked).
- `hover` pseudo-class styles links when someone is hovering over them.
- Example:

```
a: hover { background-color: green; }
a: visited { color: pink; }
```

# Background

| Property | Values |
|---|---|
| `background-color` | color in hex code, color name or rgb value. |
| `background-image` | `url("IMAGE URL")` |
| `background-repeat` | `repeat`, `repeat-x`, `repeat-y`, `no-repeat` |
| `background-attachment` | `fixed`, `scroll` |
| `background-position` | x% y%, x y, `left`, `right`, `center`, `top`, `bottom` |
| `background` | A shorthand form to specify all of these properties. |

# Lists

| Property | Values |
|---|---|
| `marker-offset` | distance between a marker and the text. |
| `list-style` | shorthand form to specify the following properties. |
| `list-style-position` | `inside`, `outside` |
| `list-style-image` | `url("IMAGE URL")` |
| `list-style-type` | `none`, `disc` (default), `circle`, `square` `decimal` (default), `decimal-leading-zero`, `lower-alpha`, `upper-alpha`, `lower-roman`, `upper-roman` |

# Tables

| Property | Values |
|---|---|
| `border-collapse` | `collapse`, `separate` |
| `border-spacing` | one or two length values separated by ; (the first apply to the horizontal and the second to the vertical spacing) |
| `empty-cells` | `show` (default in FF), `hide` (default in IE), `inherit` |
| `caption-side` | `top` (default), `bottom`, `left`, `right` |
| `table-layout` | `fixed`, `auto` (default), `inherit` |

# The `cursor` Property

| Value | Description |
|---|---|
| `auto` | Browser defaults |
| `crosshair` | Crosshair or plus sign. |
| `default` | Arrow |
| `pointer` | Pointing `hand` |
| `move` | Grasping hand |
| `text` | vertical bar |
| `wait` | Hourglass |
| `help` | Question mark |
| `X-resize` | `X` must be replaced by either `e`, `w`, `n`, `s`, `ne`, `nw`, `se` or `sw`, indicating that an edge can be moved from that corner. |

# Image replacement

- Before fonts were available in web, it was necessary to place an image if someone wanted a fancy text.
- Nowadays, there is some cases when it is still necessary to replace several words with image (e.g. company logo, etc.)
- This can be done either by directly placing `img` element, or by a CSS technique called image replacement.
- Image replacement is better, because semantic is not lost (used by search engines, screen readers, etc.).

# Image replacement (ctd.)

- To replace text with an image, place the image on the background and use `text-indent` property to move text from visible area.
- The technique works with elements having `display` set to `block`, `inline-block`, and the like.
- Example:

```
<h1 class="logo">My Logo</h1>
h1.logo {
  background: url(logo.png) no-repeat;
  width: 100px;              // width of logo.png
  height: 50px;              // height of logo.png
  overflow: hidden;
  text-indent: 100%;
  white-space: no-wrap;   // might be necessary
}
```

# CSS sprites

- Images are good, but they slow down the site performance, since for each image additional HTTP request is needed.
- This can be improved by producing so called image sprite, that is a big image consisting of several small ones.
- The image sprite is positioned using `background-position` in such a way that only the relevant part is visible.
- There are several tools doing all these automatically (e.g. SpriteMe).

# CSS sprites (ctd.)

- Example: HTML part

```
<ul>
  <li>
    <a class="hide twitter">Twitter</a>
  </li>
  <li>
    <a class="hide fb">Facebook</a>
  </li>
  <li>
    <a class="hide gplus">Google+</a>
  </li>
  <li>
    <a class="hide linkedin">LinkedIn</a>
  </li>
</ul>
```

## CSS sprites (ctd.)

- Example: CSS part

```
.hide {
  text-indent: 100%;
  overflow: hidden;
}
li a {
  display: block;
  width: 29px;
  height: 18px;
  background-image: url(social.png);
}
li a.twitter { background-position: 0 0;}
li a.fb { background-position: 0 -20px;}
li a.gplus { background-position: 0 -40px;}
li a.linkedin { background-position: 0 -60px; }
```

# CSS sprites (ctd.)

- CSS sprites are useful for small pictures or icons.
- Do not combine large pictures in sprite.
- It is not mandatory that all pictures in a sprite have same dimensions.
- In the latter case, you need more complicated adjustment.