

# HTML and CSS

Margit Tennosaar

# Last session

**HTML basic layout** // <doctype>, <html>, <head>, <body>, <title>

**Including CSS** // External file, internal style, inline style

**HTML attributes** // name=„value“

**HTML comments** // <!-- Write your comments here -->

**HTML typography** // <h1>-<h6>, <p>, <strong>, <em>

**Links** // <a>, external (<http://...>) and internal (.../about.html or #footer)

**Images in HTML and CSS** // external and internal.



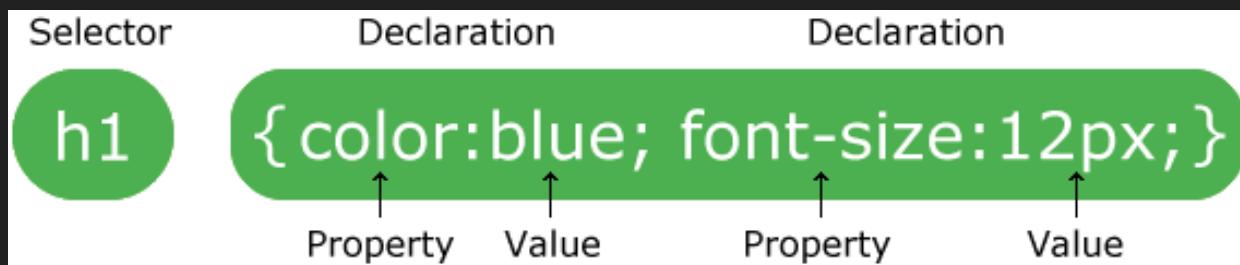
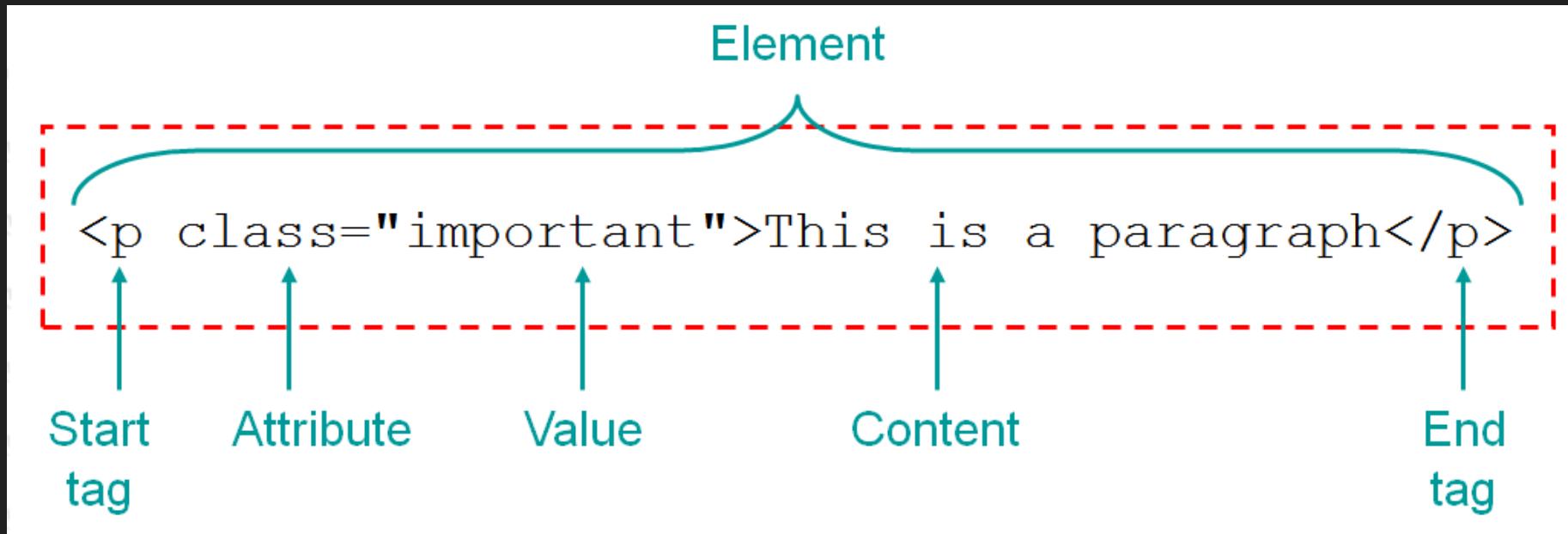
# What is CSS?

**CSS describes how HTML elements are to be displayed on the screen, paper, or in other media.**

**CSS saves a lot of work.** It can control the layout of multiple web pages all at once.

External stylesheets are stored in **CSS files**.

# Syntax and Selectors



- HTML elements
- IDs #
- Classes .

# CSS selectors

HTML element

p {color:blue}

id

#my\_id {color:blue}

class

.my\_class {color:blue}

The difference between an **ID** and a **class** is that an **ID** can identify one element, whereas a **class** can identify more than one.

# Comments

<!-- Write your HTML comments here -->

/\* Write your CSS comments here \*/

# Box model

## Box model

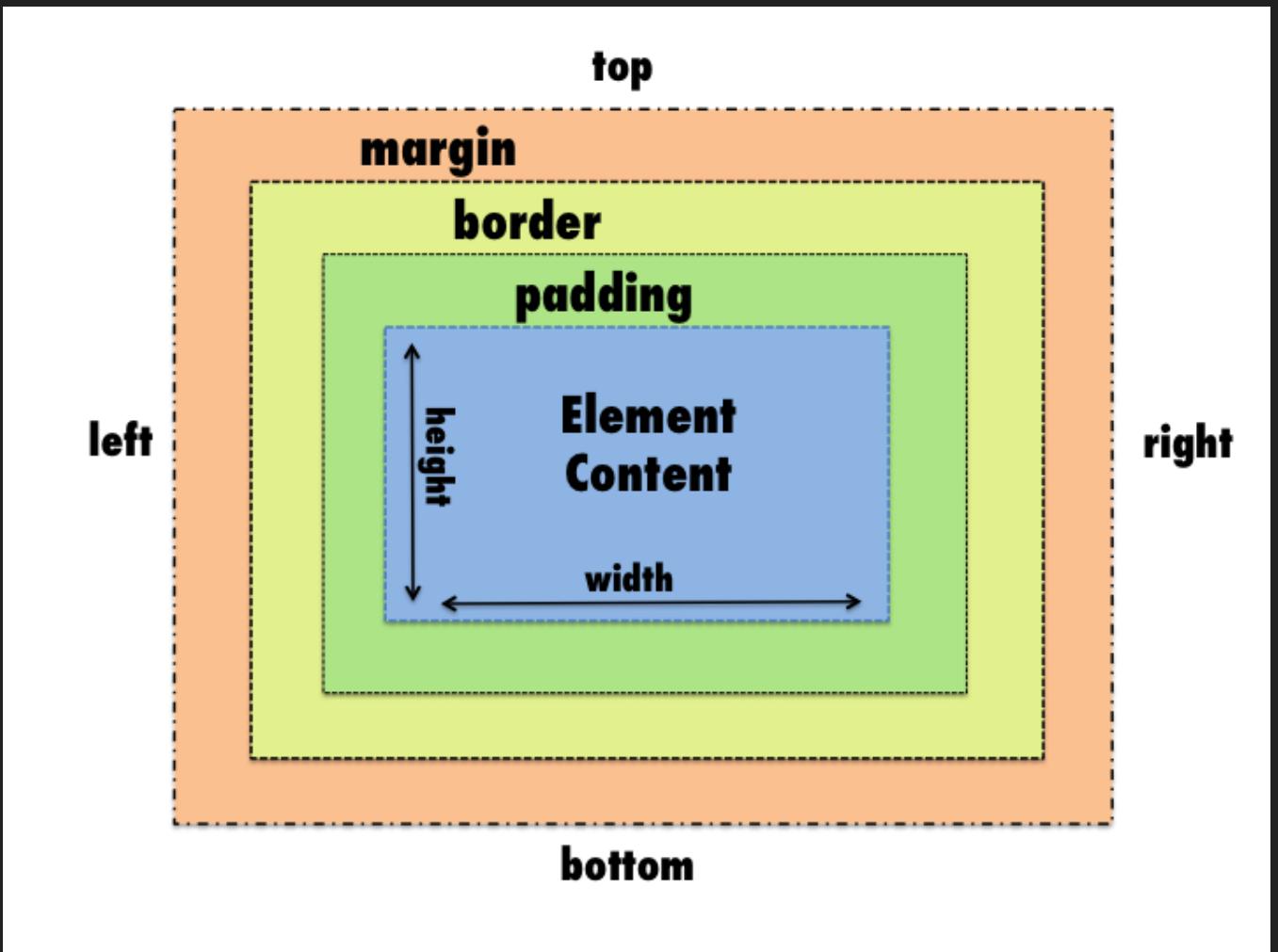


Image source: <http://espezua.github.io/blog/css-concepts.html>

# Padding

- **Padding** defines the space between an element's content and border, adding breathing room inside the element.
- Padding can be applied to individual sides or all sides simultaneously.

# Padding

```
div {  
  padding: 20px;  
}
```

```
div {  
  padding: 20px 10px;  
}
```

```
div {  
  padding: 20px 15px 10px 5px;  
}
```

```
div {  
  padding-top: 30px;  
  padding-left: 15px;  
}
```

# Border

- Borders define the **edge** of an element.
- It can be customised with width, style, and colour. These three properties always need to be visual.
- Applied individually to each side or all sides simultaneously.

# Border

```
div {  
    border: 2px solid black; /* 2px solid black border on all sides */  
}
```

```
div {  
    border-top: 3px dashed red; /* Top border only */  
    border-left: 5px double green; /* Left border only */  
}
```

```
div {  
    border-width: 4px; /* Thickness */  
    border-style: dotted; /* Style */  
    border-color: blue; /* Color */  
}
```

# Border styles

- solid: continuous line
- dotted: dots
- dashed: dashed line
- double: two lines
- groove: 3D grooved effect
- ridge: opposite of groove
- none: no border

# Advanced borders

Border radius

Border image

Outline

# Box sizing

- Determines whether padding and borders are included in an element's dimensions.
- Helps prevent layout issues caused by unexpected size calculations.

# content-box (default)

- Padding and borders are **not included** in the element's width and height.
- Total size = width/height + padding + border.

```
div {  
  box-sizing: content-box;  
  width: 200px;  
  padding: 20px;  
  border: 10px solid black;  
}
```

Total width = **200px + 20px (padding) + 10px (border) = 260px.**

# border-box

- Padding and borders are **included** in the element's width and height.
- Total size = width/height (remains fixed).

```
div {  
  box-sizing: border-box;  
  width: 200px;  
  padding: 20px;  
  border: 10px solid black;  
}
```

Total width = **200px (includes padding and border)**.

# Margin

- Margin defines the **outside** space around an element.
- Margins do not affect the element's size but determine its positioning relative to other elements.

# Margin

```
div {  
  margin: 20px;  
}
```

```
div {  
  margin: 20px 10px;  
}
```

```
div {  
  margin: 20px 15px 10px 5px;  
}
```

```
div {  
  margin-top: 30px;  
  margin-right: 15px;  
  margin-bottom: 20px;  
  margin-left: 10px;  
}
```

# Box model recap

CSS box model is the **base of CSS layouts**

All elements have **height** and **width**

**Inside** space is created with **padding**

**Outside** space is created with **margin**

CSS "starts counting" sides from **top, right, bottom, left**

# CSS universal selector

The CSS **universal selector** (\*) matches elements of any type.

Common use is for "universal reset"

```
* {  
margin: 0;  
padding: 0;  
box-sizing: border-box;  
}
```

# CSS naming

# CSS naming

- Clear names make your code easy to understand for you and others.
- Consistent naming makes it easier to add new features without breaking existing ones.
- Working in a team is smoother when everyone follows the same rules.

# Naming tips

- Choose names that describe the purpose or role of the element, not its appearance.
- Use lowercase letters and hyphens to separate words.
- Avoid spaces, special characters, or camelCase in class names.
- Use unique names to prevent conflicts with other classes or libraries.

# Bad naming

```
<div class="red-box">This is a red box</div>
<div class="big-title">Main Heading</div>
```

```
.red-box {
  background-color: red;
  padding: 10px;
}
```

```
.big-title {
  font-size: 32px;
  font-weight: bold;
}
```

# Good naming

```
.alert {  
  padding: 10px;  
  border: 1px solid #f00;  
  background-color: #ffe5e5;  
}
```

```
.error-message {  
  color: red;  
}
```

```
.heading {  
  font-size: 16px;  
  font-weight: normal;  
}
```

```
.large-heading {  
  font-size: 32px;  
  font-weight: bold;  
}
```

# BEM method

The BEM method is one of the most popular naming conventions and a great starting point for beginners.

**Block** - A standalone component or section (e.g., a card, a form).

**Element** - A part of the block (e.g., a button in a form).

**Modifier** - A variation or state of the block or element (e.g., a highlighted card).

```
<div class="card card--featured">
  <h2 class="card__title">Card Title</h2>
  <p class="card__description">This is a card description.</p>
</div>
```

```
/* Block */
.card {
  border: 1px solid #ddd;
  padding: 16px;
  background-color: #fff;
}
```

```
/* Modifier */
.card--featured {
  border-color: gold;
}
```

```
/* Element */
.card__title {
  font-size: 18px;
  margin-bottom: 8px;
}
```

# CSS combinator

**descendant** selector (space)

main p {background:green}

*All paragraphs which are in the main*

**child** selector (>)

main > p {background:green}

*All paragraphs which are in the main and are direct children*

**adjacent sibling** selector (+)

main + p {background:green}

*The paragraph which is directly after the main*

**general sibling** selector (~)

main ~ p {background:green}

*All paragraphs which are coming after the main*

# CSS specificity

Specificity determines which CSS rule is applied when multiple rules target the same element.

Higher specificity overrides lower specificity.

Inline styles: 1000

ID selectors: 100

Classes, attributes, pseudo-classes: 10

Type selectors and pseudo-elements: 1

# Inheritance

Some CSS properties, like color and font-family, are inherited by child elements, while others (e.g., margin and padding) are not.

```
p {  
  color: inherit;  
}  
  
<div style="color: red;">  
  <p>This paragraph inherits red color.</p>  
</div>
```

# Float

```
Main #past img {  
    float:left;  
}
```

The **float** CSS property places an element on its container's left or right side, allowing text and inline elements to wrap around it.

When you want to clear floating for the next elements, use the **clear** property