Cascading Style Sheets

Kianoosh Abbasi

CSC309 Winter 2022

So far

- How web works
  - Client/server – request/response - HTTP
- Web browsers
  - Sends requests & renders response
- HTML
  - Tags: headers, inputs, etc.

This session

- Adding style to HTML
- Basic CSS rules
  Styles, selectors, precedence, units
- Spacing
  Box model: margin, border, padding
- Layout
  Positioning, flexbox, grid

CSS styles

- HTML elements & attributes

  `<img src="a.jpg" />, <a href="http://google.com">`

- Style attribute

  Describes how the element should look like

- Usage:

```
<h1 style="color: red"> Hello </h1>
<div style="width: 50px; height: 60px"> … </div>
```

CSS basic properties

- color
- background-color
- font-size
- text-align
- width
- height
- z-index
- font-family
- list-style-type
- opacity

Stylesheets

- Inline styles make our HTML crowded and dirty
  Generally not recommended
- Cannot be applied to multiple elements
- Solution: style tags
- Or stylesheets: a separate CSS file

Stylesheets

- Style tag:

```
<style>
    (CSS content)
</style>
```

- Recommended to be placed inside <head> element

- CSS file

  Put css content in a file (name usually ending with .css)

  Then inside <head>: `<link rel="stylesheet" href="style.css" />`

Selectors

- Styles can be applied to an arbitrary set of elements
- All elements of a certain tag
  All elements (the <html> element)
- A specific element: the id attribute
- A set of elements: the class attribute

Selectors

- CSS content (inside <style> or stylesheet)

```
/* Makes all paragraphs red */
p {
    color: red;
}

/* Specifies the size of the element with id `big-text` */
#big-text {
    font-family: Arial;
    font-size: 20px;
}

/* Aligns all elements from class `center` */
.center {
    text-align: center;
}
```

Combined Selectors

- The AND condition

```
h1#big-text {
.text.center {
.text#big-text {
button.btn {
```

- The OR condition

```
h1, h2, p {
h1, .center {
```

- Descendant condition

```
div #big-text {
.center .text {
```

- Immediate child condition

```
p > span {
.form-field > div {
```

- Adjacent sibling condition

```
p + div {
```

Pseudo-classes

- Mouse over a link

```
a:hover {
    cursor: pointer;
}
```

- First & last child

```
div#main:first-child {
    width: 100%;
}
```

- Nth child!

```
.red:nth-child(5) {
    z-index: 999;
}
```

Precedence

Visit https://wattenberger.com/blog/css-cascade

- Inline css overrides stylesheets and <style> tags
- More specific selectors overrides less specific ones
  #ID > .class > tag
  .class1.class2 > .class1
- Later rules override earlier ones
- To evade these:
  Use the !important keyword

Units

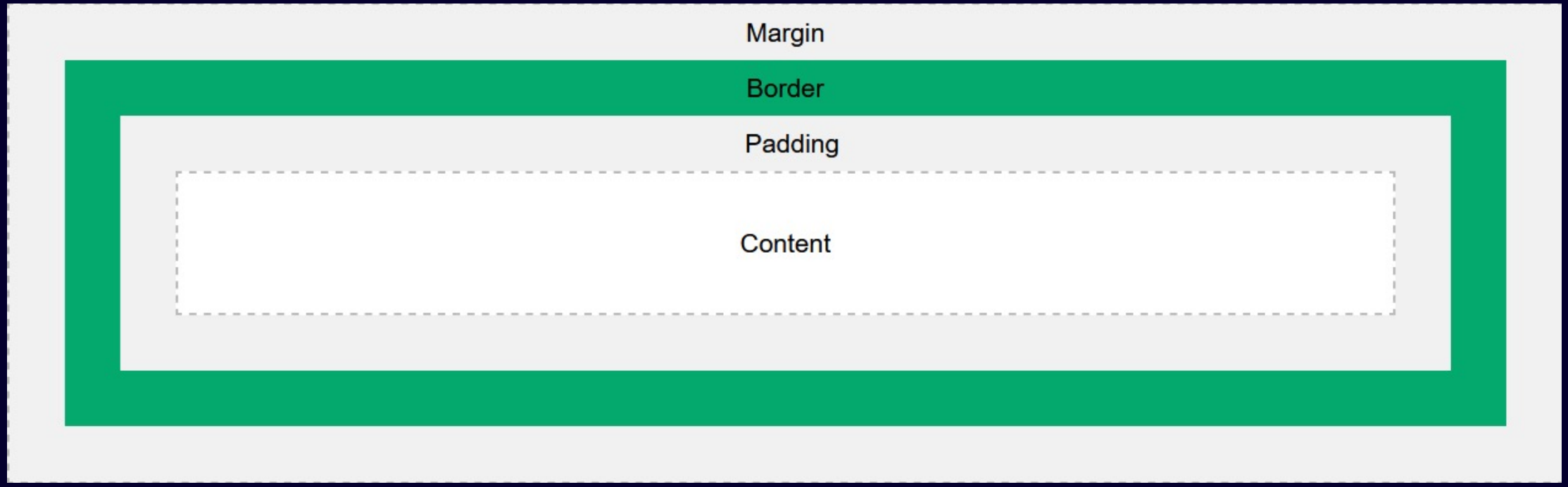
- For width, margin, font-size, etc.

- Absolute lengths
  cm, in, px

- Relative lengths
  %,
  vw: 1% of viewpoint width
  em: element's font-size,
  rem: root element's font-size (usually for font-size)
  fr: fraction (of the available space)

Questions?

Box Model



Source: https://www.w3schools.com/css/css_boxmodel.asp

Spacing

- Border

```
border-style: solid/dotted/…
border-width: 10px/thin/0px
border-color: red/white/#fa23ca
border-radius: 5px
```

- The shortcut

```
border: 10px solid red;
```

- Style a specific edge

```
border-top-color, border-left-
width
```

- Combine edges

  Top-right-bottom-left

```
border-width: 1px 2px 3px 5px;
```

- Alternative

  Top&bottom - left&right

```
border-width: 1px 2px;
```

Margin vs padding

- Margin: the space around element's borders
- Padding: the space between borders and content
- Similar rules as border-width

```
padding: 0;
margin-left: 10px;
padding: 10px 0;
```

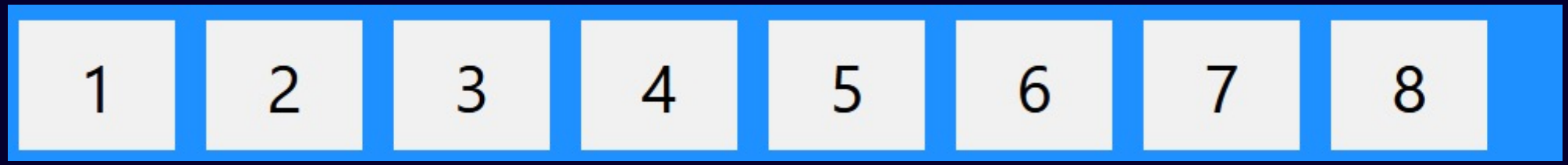
- Margin can have negative values

Layout

Visit https://www.w3schools.com/css/css_positioning.asp

- Properties
  - Top, bottom, left, and right
  - Position specifies how they should be interpreted
- Default is position: static
  - No specific positioning
  - Ignores the 4 properties
- position: relative
  - Relative to its normal (static) position
- position: fixed
  - relative to the browser's view
- position: absolute
  - Relative to nearest positioned ancestor
- position: sticky
  - Sticks to top if user scrolls

Flexbox



Source: https://www.w3schools.com/css/css3_flexbox.asp

Flexbox

- Flexibly places items inside the parent element (aka container)

- Container's style
  ```
  display: flex;
  ```

- To wrap items on overflow
  ```
  flex-wrap: wrap;
  ```

- Space between items with justify-content
  ```
  right, left, center, space-evenly, space-between, space-around
  ```

Flex items

Visit https://flexboxfroggy.com/

- flex-grow: how much it should grow if there is extra space
- flex-shrink: how much it can shrink if there is not enough space
- Both are relative to other items
- flex-basis: initial length of the item
- flex: shortcut for flex-grow, flex-shrink, and flex-basis

Grid Layout

Visit https://www.w3schools.com/css/css_grid.asp

1	2	3
4	5	6
7	8	9

Grid layout

- Parent element or container

```
display: grid;
```

- Specify how it looks like

```
grid-template-columns: auto auto auto;
grid-template-columns: 20px 10% auto;
grid-template-rows: 50px 100px;
```

- Gaps and Space between columns/rows

```
grid-column-gap: 10px;
grid-row-gap: 1em;
justify-content: space-evenly; /* columns */
align-content: center; /* rows */
```

Grid Item

Visit https://www.w3schools.com/css/css_grid_item.asp

- Specifies which columns/rows to place the item

- Start & end

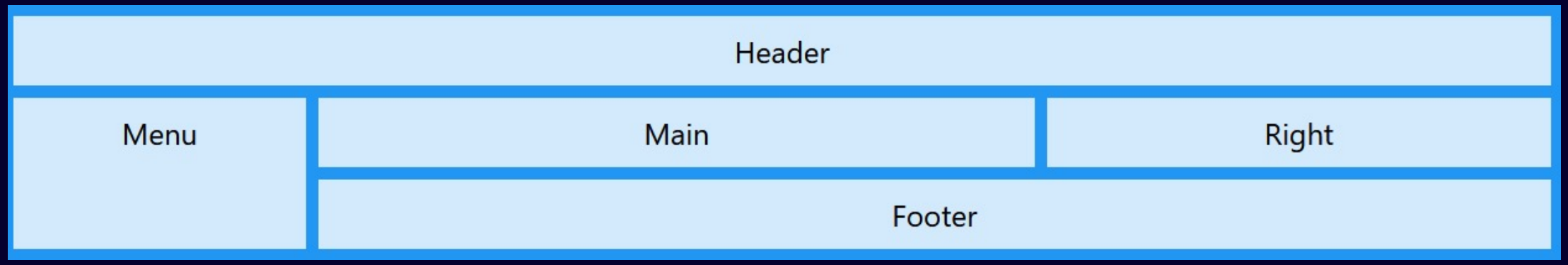
```
grid-column: 1 / 5;
```

- Start & length

```
grid-row: 2 / span 4;
```

- Assign names with grid-area

Exercise



Questions?

Also visit https://cssgridgarden.com/

Responsive design

- Should render well in different devices
  Wide screeners, laptops, tablets, smart phones

- General tip: avoid absolute lengths
  Use width: 10% or 2em instead of 100px
  Use rem for font-sizes

- Most times, a new design is needed for phones
  @media queries
  https://www.w3schools.com/css/css_rwd_mediaqueries.asp

CSS frameworks

- Include thousands of pre-defined classes
- Bootstrap
  https://getbootstrap.com
- Semantic
  https://semantic-ui.com
- Bulma
  https://bulma.io
- React-only frameworks: Material UI, Ant Design, etc.

This session

- Adding style to HTML
- Basic CSS rules
  Styles, selectors, precedence, units
- Spacing
  Box model: margin, border, padding
- Layout
  Positioning, flexbox, grid

Next session

- Back-end development & frameworks
- Python projects
  Virtual environment & pip
- Django
  Setup, simple views, forms, templates

Final notes

- Assignment 1 due is next Friday!
- Submit your code through Markus
- You are not allowed to use frameworks for A1!
- Register project teams on Markus