Responsive design (continuation)

MGR. DANIELA PONCE, PHD.

(2019)

Lecture content

- CSS modul Flexbox
- Links styling
- Responsive images

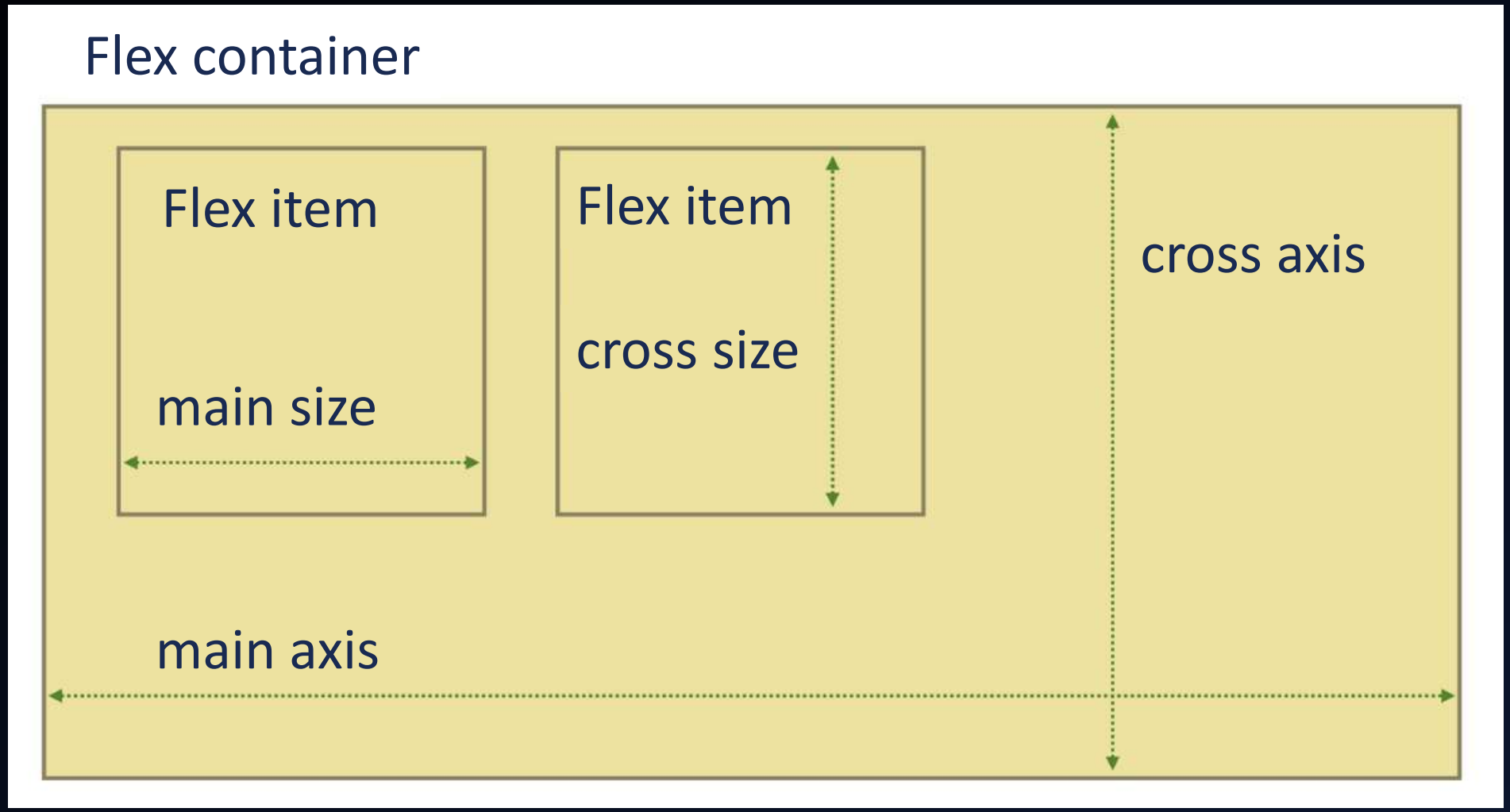
Layout modes

- Block
- Inline
- Table
- Position
- **Flex**

FlexBox

- Modul for flexible layout creation
  - Not one property but a set of styling properties
  - Containing element „flexible container“ + contained elements
- The basic idea is an automic distribution of the free space based on modifications of contained element sizes
- Solution to many styling problems previously solved by JS (e.g. problem of equally sized columns)
- W3C Candidate recommendation (as of 19 November, 2018)
- Many possible ways of deployment in responsive design
- Simple way to set the optimal display

FlexBox



- Image taken from https://www.vzhurudolu.cz/

Basic terms

- Flex container
  - Parent element (a container, which contains the items to be displayed)
- Flex item
  - Child of the flex container (direct descendant)
- Main axis
  - Gives the direction of the main flow of items rendering inside of the flex container
  - Its default orientation is horizontal (can be changed)
- Cross axis
  - Perpendicular to the main axis
- Main and cross size
  - For flex items, analogosous to axes

Key properties of the container (CSS attributes)

- Display: flex for displaying as flexbox
- Flex-direction direction of rendering (main axis)
- Flex-wrap break, tj. possibility to move flex items to the next row (if not told differently, the flex container will attempt to display all flex elements at one row)
- Justify-content .. Justifying flex items inside the flex container
- Align-items aligning flex items along the cross axis
- Align-content aligning rows of flex container along the cross axis (space between rows)

Example: centering element vertically

```
<body>
<main>
    <h1>Wanna be centered!</h1>
</main>
</body>
```

```
body {
    display: flex;
    min-height: 100vh;
    margin: 0;
}
main {
    margin: auto;
}
```

Example (blackboard with messages from the exercise 5.1)

```
<article>
    <section>
        <p>Lorem ipsum dolor …</p>
    </section>
    <section>
        <p>Lorem ipsum dolor …</p>
    </section>
    <section>
        <p>Lorem ipsum dolor …</p>
    </section>
    …
</article>
```

```
article {
    display: flex;
    flex-wrap: wrap;
    justify-content: space-around;
    margin: .5em;
}
section {
    width: 10em; height:7.5em;
    overflow: hidden;
    margin: .5em;
    padding: 0.5em;
}
```

Key properties of flex item (CSS attributes)

- Order
  - Order of items inside the container
- Flex-grow
  - An option to make the item wider, if space is available
- Flex-shrink
  - An option to make the item narrower, if necessary
- Flex-basis
  - Basic size of the item before the redistribution of the free space
- Align-self
  - Individual alignment of the item

Example: forced placement of the footer on a short page

```
<header>
   <h1>Site name</h1>
</header>
<main>
   <p>Bacon Ipsum dolor sit …</p>
</main>
<footer>
   <p>© 2015 No rights reserved.</p>
</footer>
```

```
body {
   display: flex;
   flex-flow: column;
   min-height: 100vh;
}

main {
   flex-grow: 1;
}
```

1vh = 1% of the height of the viewport

FlexBox

- Examples, references
  - Lea Verou: CSS Secrets: Better Solutions to Everyday Web Design Problems (it can be found at Internet as pdf file, too)
  - http://www.w3schools.com/css/css3_flexbox.asp
  - https://css-tricks.com/snippets/css/a-guide-to-flexbox/
  - https://www.w3.org/TR/css-flexbox-1/ (W3 specification)
  - https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Flexible_Box_Layout/Using_CSS_flexible_boxes (MDN)
- For maximum compatibility it is very convenient to prefix the properties (e.g. display: -webkit-box)

Link styling

- Any attribute can be used (e.g. color, background color, font, decoration, margins, display mode)
- By using pseudoclass selector, styling can take into account the state of the link:

a:link	normal state, not visited link
a:visited	visited link
a:hover	link with mouse over it
a:active	link in the moment of being clicked

- The LVHA order of styling rules in the stylesheet is important for proper styling:
  - :link {} :visited {} :hover {} :active {}

Example

```
a:link, a:visited {
    background-color: white;
    color: black;
    border: 2px solid green;
    padding: 10px 20px;
    text-align: center;
    text-decoration: none;
}
a:hover, a:active {background-color: red;}
```

Responsive image (HTML5 element picture)

- Automatic change of sizes insufficient -> composed element picture
- Conditional usage of image variants in specified in source elements
- In the media attribute, usage condition is formulated (in the same way as in @media)
- One „backfall“img element for not supporting browsers (which?)
- Example:

```
<picture>
  <source srcset=„flowers-small.jpg" media="(max-width: 400px)">
  <source srcset=„flowers.jpg">
  <img src=„flowers.jpg" alt="Flowers">
</picture>
```