Site layout and responsive design

MGR. DANIELA PONCE, PHD.

(2019)

Lecture content

- Block and line elements
- Page layout
- Responsive design
- Media queries
- Placing elements on the page (position attribute)

Block and line elements

- Each HTML element has its initial way of display depending on the type of the element: display attribute
- Most elements have the initial way of display:
  - block: div, h1 to h6, p, header, footer, section, …
  - inline: span, a, img
  - Other values exist, e.g. inline-block, table, list-item, … (see in detail)
- The way of display can be changed, if needed
- Influence on the direction of element rendering in the document flow, other attributes are not affected

```
<span class=„no-nested“>
    <div>…</div>
</span>
. no-nesteddisplay: block;}
```

Block elements

- Always start on a new line
- Use all device width available (strech to maximum on both sides)
- In normal order, they are rendered one beneath other

- Examples:
  - <h1> - <h6>, <p>
  - Styling rule in CSS: .inlineDiv{display: block;}

(In)line elements

- They do not start on a new line
- They are as wide as needed
- In normal order, they are rendered one beside other

- Examples:
  - elements <a>, <img>
  - Styling rule in CSS: li {display: inline;}

Inline-block elements (display: inline-block)

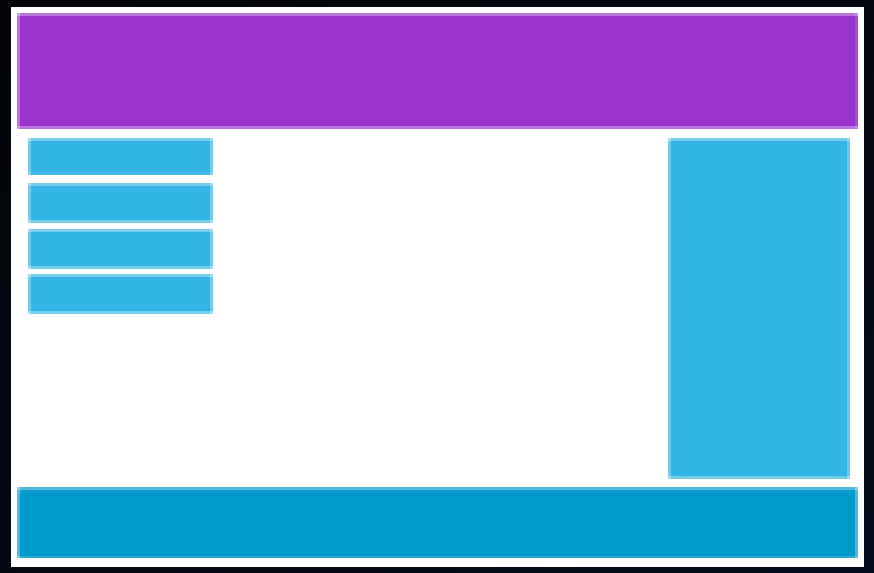
- Inline element with properties of block element:
  - Width and height of the element can be set (not true for inline elements)
  - Top and bottom paddings and margins will be applied (not true for inline elements)
  - A new row is inserted after the element (true for block element)

- Common usage: links in navigation (example); in a narrow window, a new row will be inserted

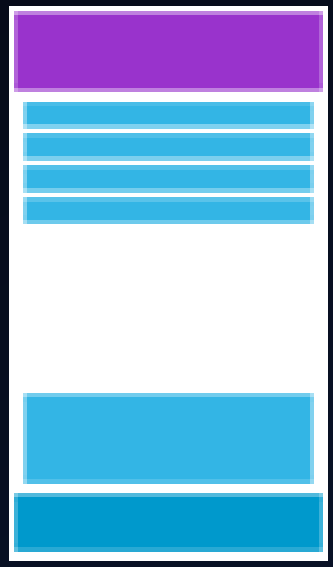
Page layout assembling

- The page is understood as a group of areas
- Each area has its purpose (menu, content, header…)
- Definition of page areas
  - General-purpose elements
  - Semantic HTML5 elements (header, nav, article, section, aside, footer)
- Several basic layout types:
  - Single column – common for mobile browsers
  - Two columns – common for tablet and laptop bowsers
  - Three /multiple columns – for desktop browsers only

PC



Mobil



Responsive design

- „Such a design of web page that the page matches the window size, display type or orientation“
- Responsive page changes in reaction to the surrounding factors
- What can change:
  - Page layout, visibility of elements
  - Sizes of used fonts
  - Image sizes
  - Indentation
  - Element types can be interchanged, e.g. List element ul (navigation on desktop) is replaced by drop-down menu element select (navigation in mobile)

Why responsive design?

- Styles for different media exist
- Mobile or printer can use different styles
- Without responsive design, it is not possible to respond to display resolution, density (dpi), orientation
- 10px font on FullHD display is almost unreadable
- Achieving the same font size is not sufficient
- Displaying the same page on mobile device is impractical
- The user must zoom in, scroll down
- The page is controlled by touching, not but pointing (by cursor)

Media queries

- CSS technique for creating responsive design
- A query on the medium will be included in the set of rules when a certain set of conditions is met
- The conditions apply to the display device of the user
- Example:

```
@media all (max-width: 480px) {
        .container { width: auto }
}
```

- They are commonly used in combination with other sets of rules that are valid in all cases.

Styles for different media

- HTML page can have several stylesheets attached for different media (=output devices).
- The stylesheet for another medium contains the same rule sets with the same selectors; however, attributes are set different values or even different attributes are set
- Example for printed version of page (print medium):
  - Element hiding (menu, login, search), content extension to free areas, text color changes
- Three ways to connent medium and style (example for print medium):

1. `<link rel="stylesheet" href="style.css" type="text/css" media="print" />`
2. `@import url(“style.css“) print;`
3. `@media print {  /* style rules*/ }`

Media query (@media) structure

- @media not|only *medium-type* and (*properties*) {
  CSS declarations;
  }
- The only keyword is for older browsers that do not understand @media queries (5% of user in Czech Republic as of Oct 30, 2018 see statistics statistics)
- The not keyword for negation of the condition following
- *Medium-type* can have these values :
  - all, embossed, handheld, print, projection, screen, speech, tty, tv
- The *(properties)* part is optional
- The *all* value can be omitted

Composed conditions in media queries

- Several media types can be given
- The or logical connector is used, specified as comma, example:
  - @media (max-width: 400px), print { … *declarations* …}
- The comma separated list of media can also contain reference to property, example :
  - @media screen and (monochrome), print { … *declarations* …}
- The and logical connector can be used:
  - @media screen and (min-width: 400px) and (max-height: 600px) { … *declarations* …}

Media groups

- W3C specification of properties is formulated for media groups (no one-by-one medium listing)
- The specification uses these groups of media:
  - continuous or paged
  - visual, audio, speech or tactile
  - grid or bitmap
  - interactive or static
  - all

Assignment of medium type to media groups

- The most common media types (screen, print) belong to these groups:
  - Print:
    - Paged, visual, bitmap, static
  - Screen:
    - Continuous, visual, audio, bitmap, static, not static

- For your curiosity, the tv medium is:
  - Paged, continuous, visual, audio, bitmap, static, not static

Properties that can be detected

- Width, height, device-width, device-height, orientation, aspect-ratio, device-aspect-ratio, color, color-index, monochrome, resolution, scan, grid
- Most of the properties can be also used with min- and max-prefices

Rendering area (element viewport)

- Area within browser where the page will be rendered
- The size of rendering area can be specified in a meta element
- Example – the rendering area width is set to the width of the device:

  ```
  <meta name="viewport" content="width=device-width, initial-scale=1">
  ```

- For mobiles, the rendering will use the full width of the screen
- Not recommended: to disable user change of the rendering area (user-scalable attribute)
  - <!– NOT RECOMMENDED! -->
  - <meta name="viewport" content="user-scalable=no">
  - <meta name="viewport" content="width=device-width, maximum-scale=1.0">

Selected properties

- Width, device-width:
  - Applicable for media types visual, tactile
- Width:
  - Continuous media – viewport width including scrollbar is used as base
  - Paged- media – page box (what is it?) is used as base
- Device-width:
  - Continuous media – screen width is used as base
  - Paged- media – paper width is used as base
- Similarly height, device-height

Further selected properties

- Color, monochrome:
  - For visual media type
- Color:
  - „number of bits in one pixel in color component of the device “
  - Value is zero for monochrome devices
  - Style for color printing: media=“print and (color)“
- Monochrome:
  - „ number of bits in one pixel in monochromatic buffer“
  - Value is zero for non-monochrome devices
  - Style for black and white printing: media=“print and (monochrome)“

More selected properties

- Resolution:
  - For bitmap media
  - In dpi, dpcm values
  - It is recommended to use min- and max- variants

- Example:
  - @media print and (min-resolution: 120dpcm) {…}

How to create a responsive site

- Design of general selectors
  - Headlines
  - Paragraphs
  - Navigation elements
  - Font size is usually defined in em units (relative way)
- Definice of the individual size variants
- Consideration of how exactly the site should look like and which parts will be changed across variants
- Writing media queries according to the previous point
  - Control the font size, image sizes, layout proportion in every layout variant

Hints

- The entire layout may be fixed; however, for different resolutions different variants exist
- Images should be wrapped into containers and their width should be set to 100%, ensuring flexibility
- In some cases two variants of one functionality have to be included in the HTML document and complementary switching on/off of functionalities has to be implemented (by display: block and display: none)

Testing

- Most of funcionality can be tested within browser
- Use developer tools (F12 key) for this purpose
- Chrome browser has a bulit-in emulator of mobile browsers, it is possible:
  - To set manually screen parameter
  - To select a device from the list

Layout modes

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

Element positioning

- Positioning schemas:
  - Normal flow (block and inline formating, relative positioning)
  - Floating model (float property)
  - Absolute positioning
- Relative positioning: the box remains in the flow, it is just a little bit „deflected“
- Absolute positioning: the box is cut out from the flow and it is placed at new position

- Shared by relative and absolute positioning: position attribute

Placing an element on a page with attribute position

- Static .. The element is rendered in the order of its location in the document flow
- Absolute .. With respect to the first non-static ancestor
- Fixed .. With respect to the browser window
- Relative .. With respect to the normal position of the element itself
- Sticky .. Static until it reaches the specified position, then fixed
- Initial .. Set to the default value (static)
- Inherit .. Sets to the value from the parent element

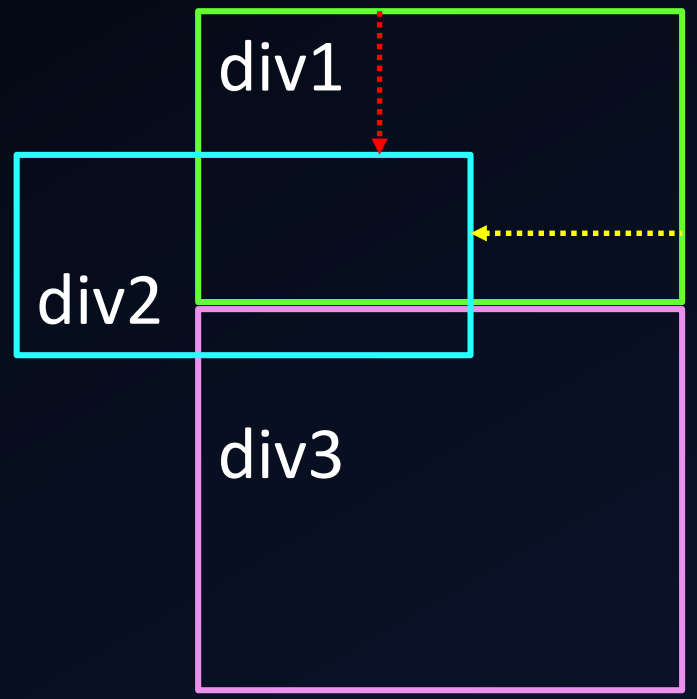
See also: http://www.barelyfitz.com/screencast/html-training/css/positioning/

Consequences of element positioning

- Absolute positioning
  - The element is cut out from the normal rendering flow of the document
  - If the document is scrolled, the element is scrolled in the same way
- Fixed positioning
  - The element is fixed on the screen
  - If the document is scrolled, the element does not scroll
  - If the document is printed, the element is printed at the same position on all pages.
- Relative positioning
  - The element is not cut out from the normal rendering flow of the document
  - Its normal position remains preserved
- Element in position absolute or fixed can overlap other elements
- Elements overlapping can be controlled by z-index attribute of elements
  - Both negative and positive values are allowed; the higher the value, the closer to the user the element is
  - Only for elements with absolute, relative or fixed position

Example: absolute positioning

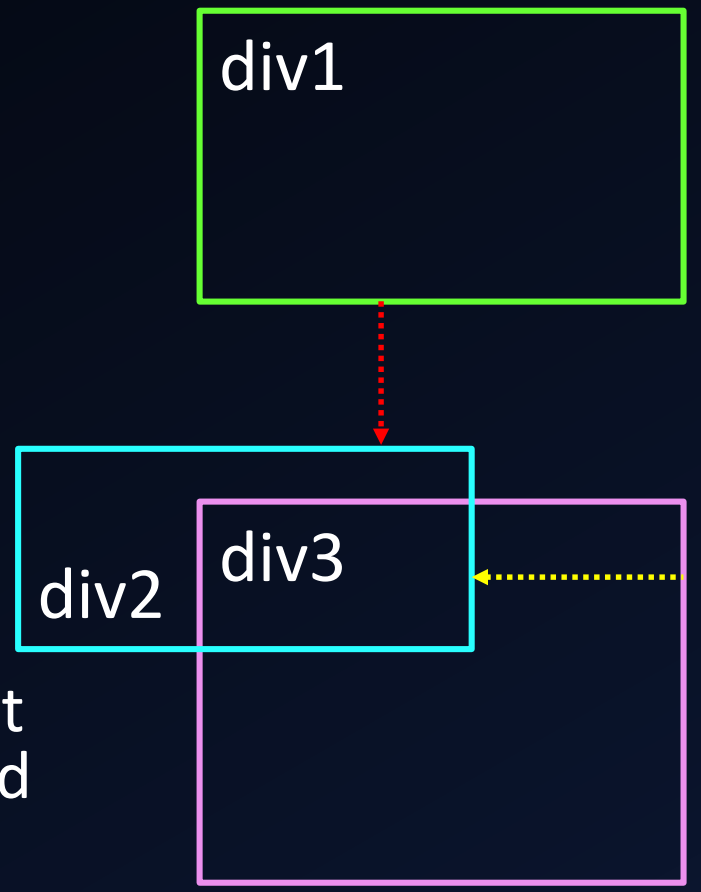
```
#div2 {
    position: absolute;
    width: 100px;
    height: 70px;
    right: 50px;
    top: 30px;
}
```



- The element div2 will be offset **50px from the right** margin of the page, and **30px from the top** of the page.
- Its original location between the elements div1 and div3 will be released, and div3 will be rendered immediately after div1.

Example: relative positioning

```
#div2 {
    position: relative;
    width: 100px;
    height: 70px;
    right: 50px;
    top: 30px;
}
```



- The element div2 will be offset 50px from the right of the original position of the original position, and 30px from the top of the original position.
- The normal position of the element remains preserved.

Responsive design - summary

- modern trend (for the last 10 years ☺)
- Increasing share of mobile devices (mobile+tablet outnumber desktops worldwide) changed WWW - design of majority of web sites is responsive
- Knowledge of responsive design basics is a must
- How to continue?
  - http://designmodo.com/responsive-design-examples/
  - http://www.smashingmagazine.com/2011/01/12/guidelines-for-responsive-web-design/
  - … search yourself