CS472 WAP

Responsive Design

Page Layout

The Whole is Greater than the Sum of the Parts

Maharishi International University Fairfield, Iowa © 2020

Main Point Preview

- **Responsive design** is the strategy of making a site that responds to the browser and device width. Responsive design utilizes media queries to determine the available display area and new CSS techniques such as flexbox and grid to make designs more flexible.
- *The unified field is the source of all possibilities and when we think from this level our actions are spontaneously responsive to whatever situation we encounter.*

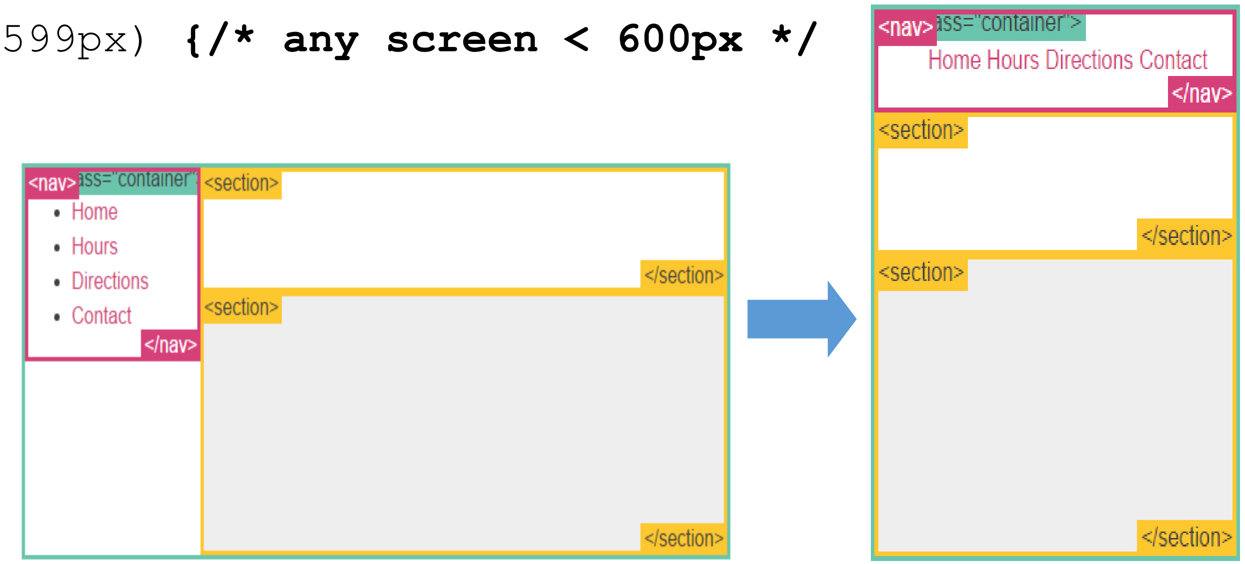
Responsiveness guidelines

- (Mobile) users scroll websites vertically not horizontally!
- if forced to scroll horizontally or zoom out it results in a poor user experience.
- Some additional rules to follow:
  - 1. Do NOT use large fixed width elements
  - 2. Do NOT let the content rely on a fixed viewport width to render well
  - 3. Do Use CSS media queries to apply different styling for small and large screens
  - https://www.w3schools.com/css/css_rwd_viewport.asp
  - https://www.quirksmode.org/blog/archives/2010/09/combining_meta.html
  - https://developer.mozilla.org/en-US/docs/Mozilla/Mobile/Viewport_meta_tag

Media queries

Media queries specify a screen size for a set of CSS rules

```
@media screen and (min-width:600px) { /* any screen > 600px */
 nav { float: left; width: 25%; }
 section { margin-left: 25%; }
}
@media screen and (max-width:599px) {/* any screen < 600px */
 nav li { display: inline; }
}
```



- See lesson3_examples\mediaquery.html
- example similar to figure

Design for Mobile First

- Mobile First means designing for mobile before designing for desktop or any other device
- Instead of changing styles when the width gets *smaller* than 768px, we should change the design when the width gets *larger* than 768px.

- @media only screen and (max-width: 600px) /* applies to any screen 600px or smaller -- avoid this */

- @media only screen and (min-width: 600px) /* applies to any screen 600px or larger -- **favor this** */

meta viewport

- For responsive design include meta viewport in head
  - <meta name=viewport content="width=device-width, initial-scale=1">
  - Sets the viewport to be same as the actual device width

- Some mobile browsers, notably Safari iPhone, have a default layout viewport of around 850 to 1000 pixels — much larger than the device width.
  - Why? to accommodate desktop sites that did not test on mobile
  - Such sites usually stretch to roughly that width

- media queries without a meta viewport tag will not look good on such devices
  - code would conclude browser is larger than 600 pixels,
  - and apply desktop styles instead of the mobile ones
  - meta viewport tag solves that problem

  - View from tablet or phone
    - Without viewport tag
    - With viewport tag

Flexbox Layout

- responsive layout without float or positioning
- one-dimensional layout model
  - space distribution between items
  - powerful alignment capabilities.
- display property flex on containing element
  - display: flex;
  - direct child elements automatically flexible items
  - simple example

Flex Container Properties

- flex direction: column or row
- flex wrap: wrap or nowrap
- center vertically and horizontally via justify-content and align-items
- Responsive image grid
- Responsive website

Flex Container Properties

display	Specifies the type of box used for an HTML element
flex-direction	Specifies the direction of the flexible items inside a flex container
justify-content	Horizontally aligns the flex items when the items do not use all available space on the main-axis
align-items	Vertically aligns the flex items when the items do not use all available space on the cross-axis
flex-wrap	Specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line
align-content	Modifies the behavior of the flex-wrap property. It is similar to align-items, but instead of aligning flex items, it aligns flex lines
flex-flow	A shorthand property for flex-direction and flex-wrap

Grid Layout

- CSS Grid offers a grid-based layout system, with rows and columns, making it easier to design web pages without having to use floats and positioning.

- A Grid Layout must have a parent element with the display property set to grid

- Direct child element(s) of the grid container automatically becomes grid items.

Grid vs Flexbox vs Multi-column

- **Grid is designed to be used *with* flexbox, not instead of it**
- Flexbox is for one dimensional layout (row or column).
- CSS grid is for two dimensional page layout.
- Multi-column best for columns with flow of content across the columns
    - E.g., flowing body of text across several columns

grid-template-columns property

- grid-template-columns: number and width of columns

```
.grid-container {
   display: grid;
   grid-template-columns: 80px 200px auto 40px;
}
```

- grid-template-rows: height of rows

```
grid-template-rows: 80px 200px;
```

Child Elements (Items)

- A grid *container* contains grid *items*.
- Default: one grid item for each column in each row
  - can also span multiple columns and/or rows

grid-column Property

- grid-column: which columns an item covers

```
/* starts on line 1 and ends on line 5: */
.item1 {
  /* n columns have n+1 lines , from line 1 at left side to line n+1 on right */
  grid-column: 1 / 5;
}

/* starts on column 2 and spans 3 columns: */
.item2 {
  grid-column: 2 / span 3;
}
```

grid-row property

- grid-row: which rows an item covers

```
 /* start on row-line 1 and end on row-line 4: */
.item1 {
  grid-row: 1 / 4;
}

 /* start on row 2 and span 2 rows: */
.item2 {
  grid-row: 2 / span 2;
}
```

grid-template-areas property

grid-template-areas: name items and use the names to layout columns and rows

```
.item1 { grid-area: header; }  /* assign names */
.item2 { grid-area: menu; }
.item3 { grid-area: main; }
.item4 { grid-area: right; }
.item5 { grid-area: footer; }

.grid-container {
  grid-template-areas:
    'header header header header header header'
    'menu main main main right right'
    'menu footer footer footer footer footer';
}
```

good example and exercise https://www.w3schools.com/css/css_templates.asp

CSS Frameworks

Because CSS layout is so tricky, there are CSS frameworks out there to help make it easier. Here are a few if you want to check them out. Using a framework is only a good idea if the framework really does what you need your site to do. They're no replacement for knowing how CSS works.

- Bootstrap
- Foundation
- SemanticUI

Bootstrap

Foundation

Semantic UI

Bootstrap grid system has responsive breakpoints

- create a row (<div class="row">).
  - add columns (tags with appropriate .col-*-* classes)
  - first star (*) represents the responsiveness: sm, md, lg or xl
    - lg means columns require a lot of space to look good
    - start stacking when screen is pretty large in order to keep them wide
  - second star represents a number, which should add up to 12 for each row

```
<div class="row">
  <div class="col-sm-4">this column will be 4 of 12 grid elements </div>
  <div class="col-sm-8"> this column will be 8 of 12 grid elements </div>
</div>
```

- col-lg, stack when the width is < 1200px. (columns always pretty large)
- col-md, stack when the width is < 992px.
- col-sm, stack when the width is < 768px.
- col-xs, then the columns will never stack. (columns might get very small)

Main Point

- **Responsive design** is the strategy of making a site that responds to the browser and device width. Responsive design utilizes media queries to determine the available display area and new CSS techniques such as flexbox and grid to make designs more flexible.
- *The unified field is the source of all possibilities and when we think from this level our actions are spontaneously responsive to whatever situation we encounter.*

CONNECTING THE PARTS OF KNOWLEDGE WITH THE WHOLENESS OF KNOWLEDGE

Page Layout: Whole Is Greater than the Sum of the Parts

1. You can use floats and positioning to change where elements are displayed.
2. Modern web apps use responsive design principles including media queries, viewport, Flexbox, grid, and frameworks such as Bootstrap

3. **Transcendental consciousness** is the experience of pure wholeness.
4. **Impulses within the Transcendental field**: At quiet levels of awareness thoughts are fully supported by the wholeness of pure consciousness.
5. **Wholeness moving within itself:** In Unity Consciousness, one appreciates all parts in terms of their ultimate reality in wholeness.

MAHARISHI INTERNATIONAL UNIVERSITY

HIGHER EDUCATION FOR HIGHER CONSCIOUSNESS