Lecture # 7 & 8

Course: Web Development

Ms. Hafiza Alia
(alia@TheProTec.com)

ROBERT W. SEBESTA

Html Forms

- The <form> Element
  - The HTML **<form>** element defines a form that is used to collect user input:
    - **<form>**
      .
      form elements
      .
      </form>
- The <input> Element
  - The <input> element can be displayed in several ways, depending on the **type** attribute.

Input Elements

Type	Description
<input type="text">	Defines a one-line text input field
<input type="radio">	Defines a radio button (for selecting one of many choices)
<input type="submit">	Defines a submit button (for submitting the form)

Text Input

```
<form>
  First name:<br>
  <input type="text" name="firstname"><br>
  Last name:<br>
  <input type="text" name="lastname">
</form>
```

Radio Button Input

- <form>

```
<input type="radio" name="gender" value="male" checked> Male<br>
<input type="radio" name="gender" value="female"> Female<br>
<input type="radio" name="gender" value="other"> Other
```

- </form>

The Submit Button

```
<form action="/action_page.php">
  First name:<br>
  <input type="text" name="firstname" value="Mickey"><br>
  Last name:<br>
  <input type="text" name="lastname" value="Mouse"><br><
br>
  <input type="submit" value="Submit">
</form>
```

Label element

- The <label> tag defines a label for many form elements.
- The <label> element is useful for screen-reader users, because the screen-reader will read out loud the label when the user focus on the input element.
- The <label> element also help users who have difficulty clicking on very small regions (such as radio buttons or checkboxes) - because when the user clicks the text within the <label> element, it toggles the radio button/checkbox.
- The for attribute of the <label> tag should be equal to the id attribute of the <input> element to bind them together.

The Name Attribute

```
<form action="/action_page.php">
  First name:<br>
  <input type="text" value="Mickey"><br>
  Last name:<br>
  <input type="text" name="lastname" value="Mouse"><br><
br>
  <input type="submit" value="Submit">
</form>
```

Grouping Form Data with <fieldset>

- The **<fieldset>** element is used to group related data in a form.
- The **<legend>** element defines a caption for the <fieldset> element

```
<form
  <fieldset>
    <legend>Personal information:</legend>
    First name:<br>
    <input type="text" name="firstname" value="Mickey"><br>
    Last name:<br>
    <input type="text" name="lastname" value="Mouse"><br><br>
    <input type="submit" value="Submit">
  </fieldset>
</form>
```

The <select> Element

The **<select>** element defines a **drop-down list:**

```
<select name="cars">
  <option value="volvo">Volvo</option>
  <option value="saab">Saab</option>
  <option value="fiat">Fiat</option>
  <option value="audi">Audi</option>
</select>
```

- The **<option>** elements defines an option that can be selected.
- By default, the first item in the drop-down list is selected.
- To define a pre-selected option, add the **selected** attribute to the option:
- Example <option value="fiat" selected>Fiat</option>

The <textarea> Element

- The **<textarea>** element defines a multi-line input field (**a text area**):
- Example

```
<textarea name="message" rows="3" cols="30">
The cat was playing in the garden.
</textarea>
```

The <button> Element

- The **<button>** element defines a clickable **button**:
- Example
- <button type="button">Click Me!</button>

List Attribute

```
<form>
  <input list="browsers" name="browser">
  <datalist id="browsers">
    <option value="Internet Explorer">
    <option value="Firefox">
    <option value="Chrome">
    <option value="Opera">
    <option value="Safari">
  </datalist>
  <input type="submit">
</form>
```

Input types

- <input type="submit" value="Submit">
- <input type="reset">
- <input type=“password">
- <input type="checkbox"> defines a checkbox.
- <input type=“file”>

HTML5 Input Types

- Color
- Placeholder
- date
- datetime-local
- email
- Month (month & year)
- number
- range
- search
- tel
- time
- url
- week

Input type Color

```
<form action="/action_page.php">
Select your favorite color:
 <input type="color" name="favcolor" value="#ff0000">
 <input type="submit">
</form>
```

Input Type Date

```
<form>
  Birthday:
  <input type="date" name="bday">
</form>
```

You can also add restrictions to dates:

```
<form>
  Enter a date before 1980-01-01:
  <input type="date" name="bday" max="1979-12-31"><br>
  Enter a date after 2000-01-01:
  <input type="date" name="bday" min="2000-01-02"><br>
</form>
```

HTML Input Attributes

- The value Attribute
  - The **value** attribute specifies the initial value for an input field:

Example

- <form action="">
  First name:
  <input type="text" name="firstname" value="John">
  </form>
- The readonly Attribute
  - The **readonly** attribute specifies that the input field is read only (cannot be changed):

Example

- <form action="">
  First name:
  <input type="text" name="firstname" value="John" readonly>
  </form>

HTML Input Attributes cont..

- The disabled Attribute
  - The **disabled** attribute specifies that the input field is disabled.
- A disabled input field is unusable and un-clickable, and its value will not be sent when submitting the form:

Example

- <form action="">
  First name:
  <input type="text" name="firstname" value="John" disabled>
  </form>

HTML Input Attributes cont..

- The size Attribute
  - The **size** attribute specifies the size (in characters) for the input field:

Example

- <form action="">
  First name:
  <input type="text" name="firstname" value="John" size="40">
  </form>

HTML Input Attributes cont..

- The maxlength Attribute
  - The **maxlength** attribute specifies the maximum allowed length for the input field:

Example

- <form action="">
  First name:
  <input type="text" name="firstname" maxlength="10">
  </form>

HTML Input Attributes cont..

- Method
- Autocomplete (on/off)
- autofocus
- form
- novalidate
- target
- height and width
- list
- min and max
- multiple
- pattern (regexp)
- placeholder
- step

HTML Input Attributes cont..

1. autocomplete
2. novalidate

The multiple Attribute

- <form action="/action_page.php">
- Select images: <input type="file" name="img" multiple>
- <input type="submit">
- </form>

The pattern Attribute

- <form action="/action_page.php">
  - Country code: <input type="text" name="country_code" pattern="[A-Za-z]{3}" title="Three letter country code">

    <input type="submit">
  - </form>
- Format: 123-45-678
  - http://rubular.com/r/ubZM1E6vol

The required Attribute

- Username: <input type="text" name="usrname" required>

The step Attribute

- The **step** attribute specifies the legal number intervals for an <input> element.
- Example: if step="3", legal numbers could be -3, 0, 3, 6, etc.
- <input type="number" name="points" step="3">

HTML5 Introduction

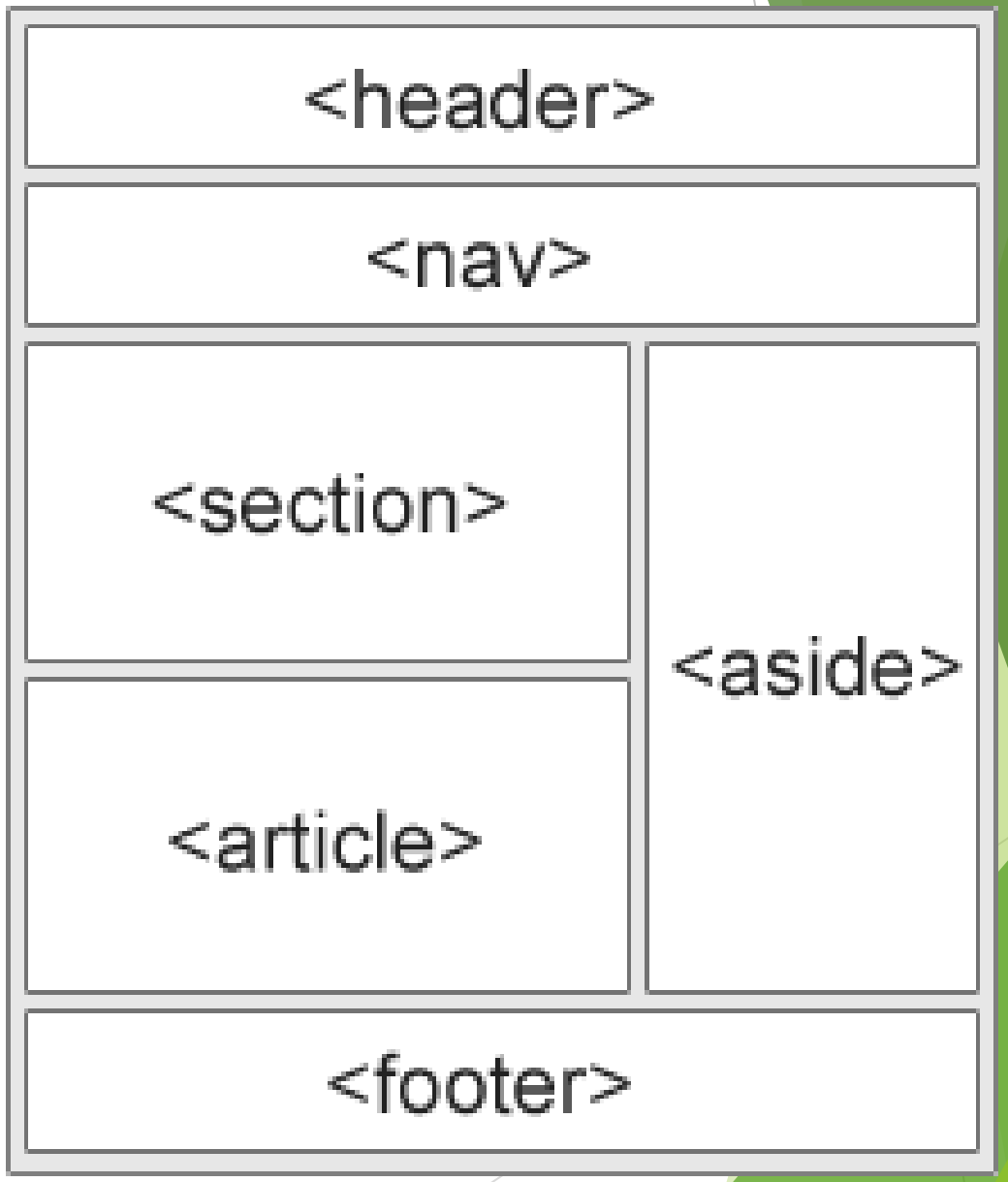
- New Elements
- Semantics

New Semantic Elements

- A semantic element clearly describes its meaning to both the browser and the developer.
- Examples of non-semantic elements: and - Tells nothing about its content.
- Examples of semantic elements: <form>, <table>, and <article> - Clearly defines its content

New Semantic Elements

- <article>
- <aside>
- <details>
- <figcaption>
- <figure>
- <footer>
- <header>
- <main>
- <mark>
- <nav>
- <section>
- <summary>
- <time>



Website Layout

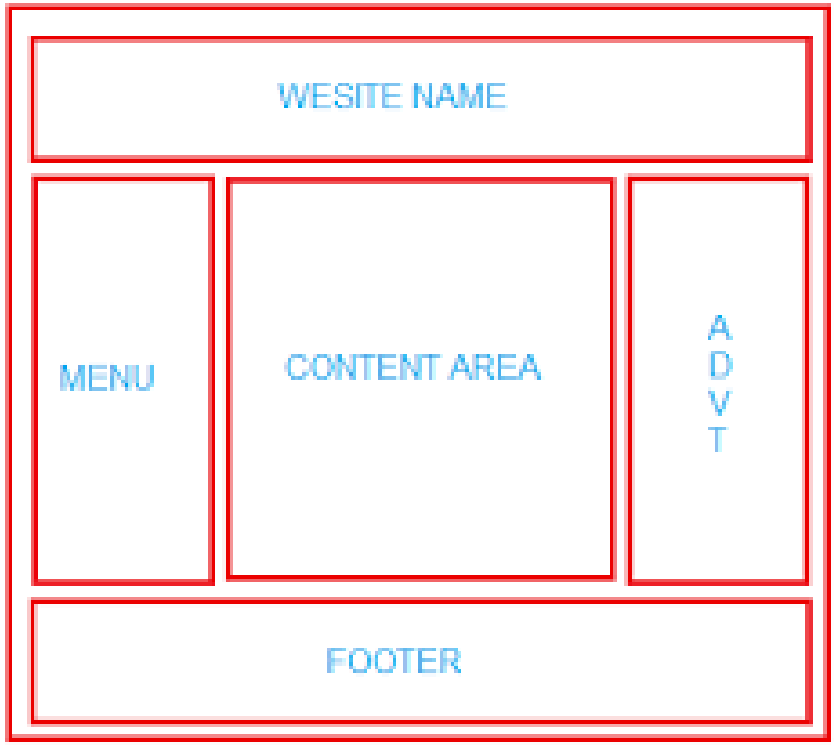
- There are five different ways to create multicolumn layouts. Each way has its pros and cons:
  - HTML tables (not recommended)
  - CSS float property
  - CSS flexbox
  - CSS grid

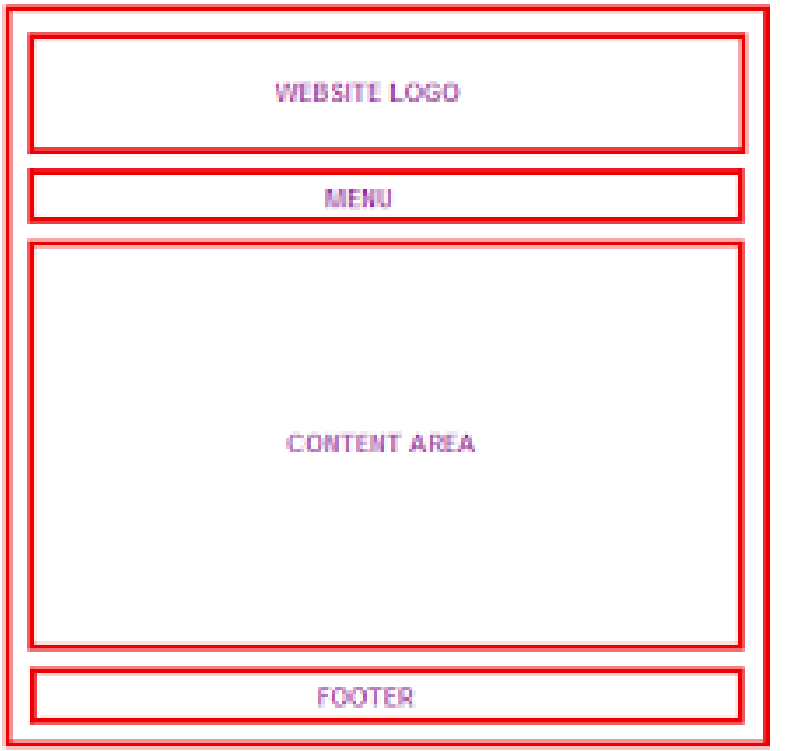
CSS Floats

- The CSS float property specifies how an element should float.
- The CSS clear property specifies what elements can float beside the cleared element and on which side.

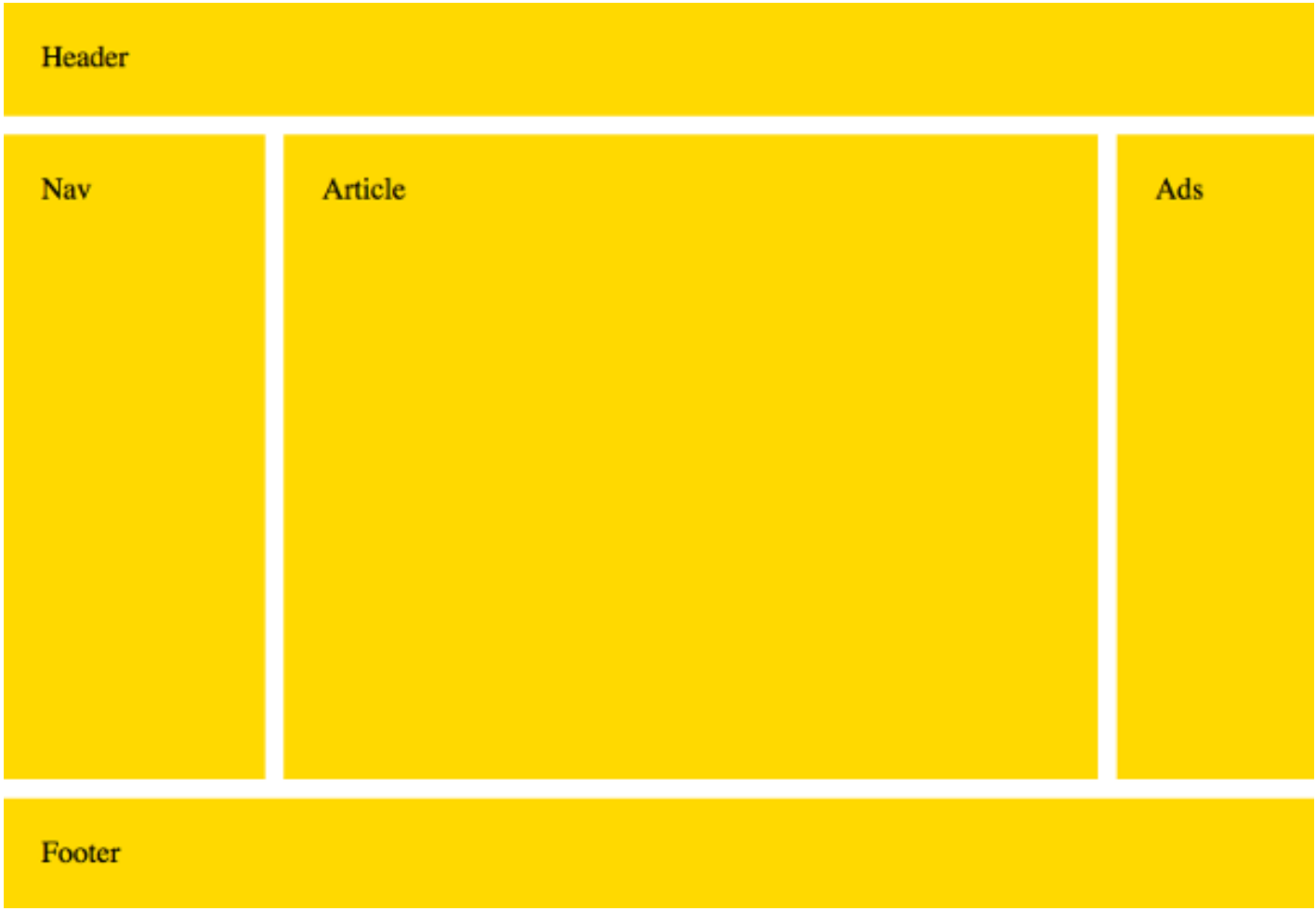
Website Layout

- A website is often divided into headers, menus, content and a footer:

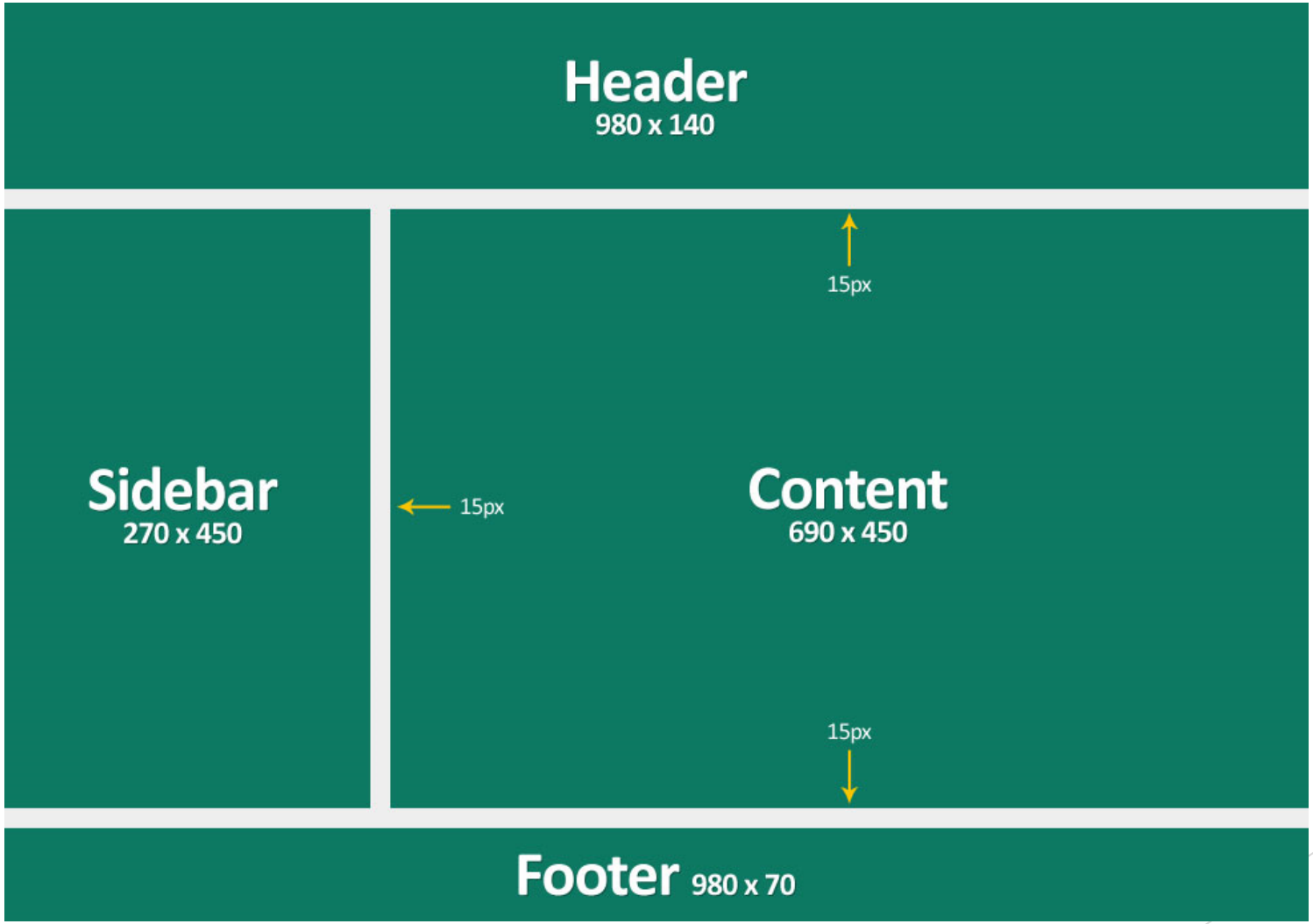




Website Layouts



Website Layouts



CSS clear property

- The clear property specifies what elements can float beside the cleared element and on which side.

- The clear property can have one of the following values:
  - none - Allows floating elements on both sides. This is default
  - left - No floating elements allowed on the left side
  - right- No floating elements allowed on the right side
  - both - No floating elements allowed on either the left or the right side

- The most common way to use the clear property is after you have used a float property on an element.

CSS Flexbox Layout

Use of flexbox ensures that elements behave predictably when the page layout must accommodate different screen sizes and different display devices.

CSS Flexbox Layout Module

- Before the Flexbox Layout module, there were four layout modes:
  - Block, for sections in a webpage
  - Inline, for text
  - Table, for two-dimensional table data
  - Positioned, for explicit position of an element

CSS Flexbox Layout Module

The Flexible Box Layout Module, makes it easier to design flexible responsive layout structure without using float or positioning.

How to make a Flexbox model?

Step 1:

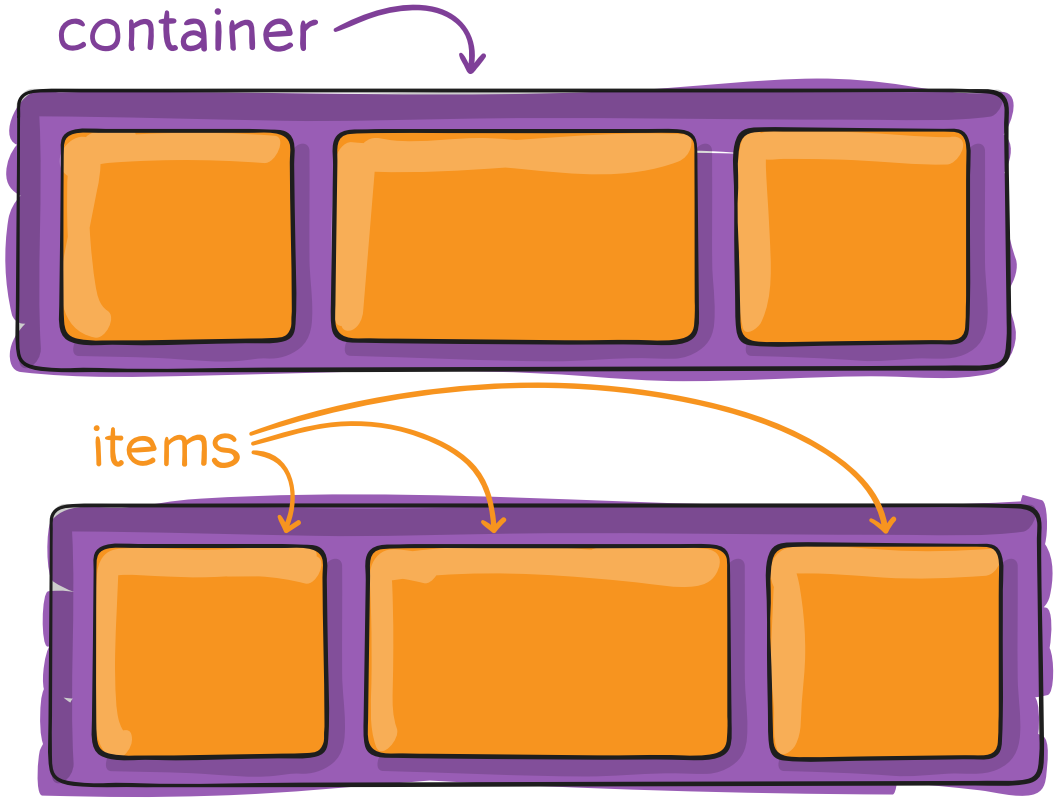
Define a flex container

```
<div class="flex">


</div>
```

Step 2:

Define display flex for parent div which is flex

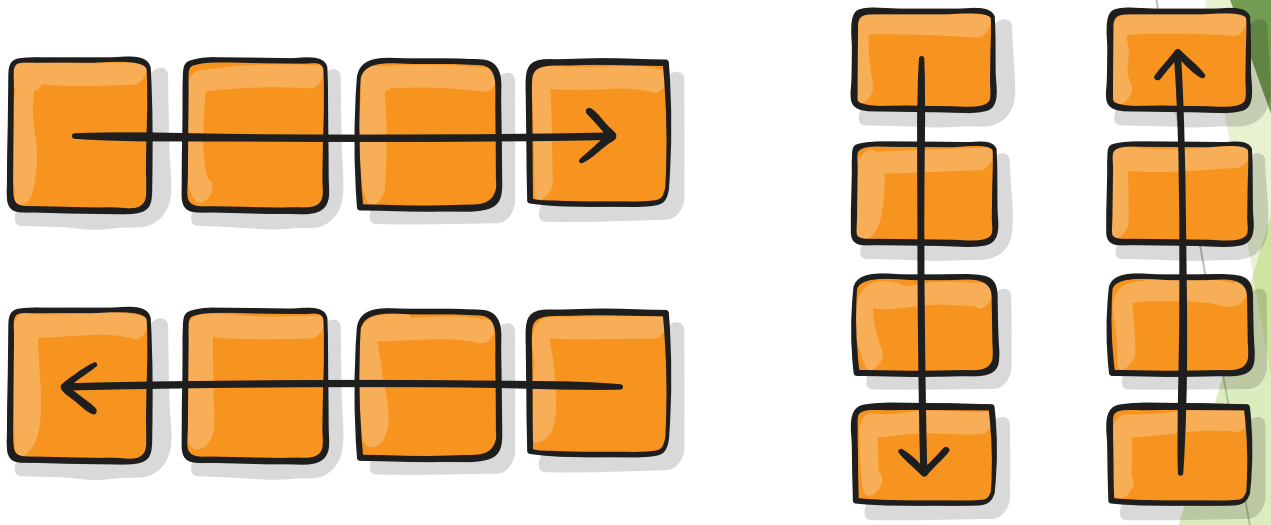


The flex container properties

- flex-direction
- flex-wrap
- justify-content
- align-items
- align-content

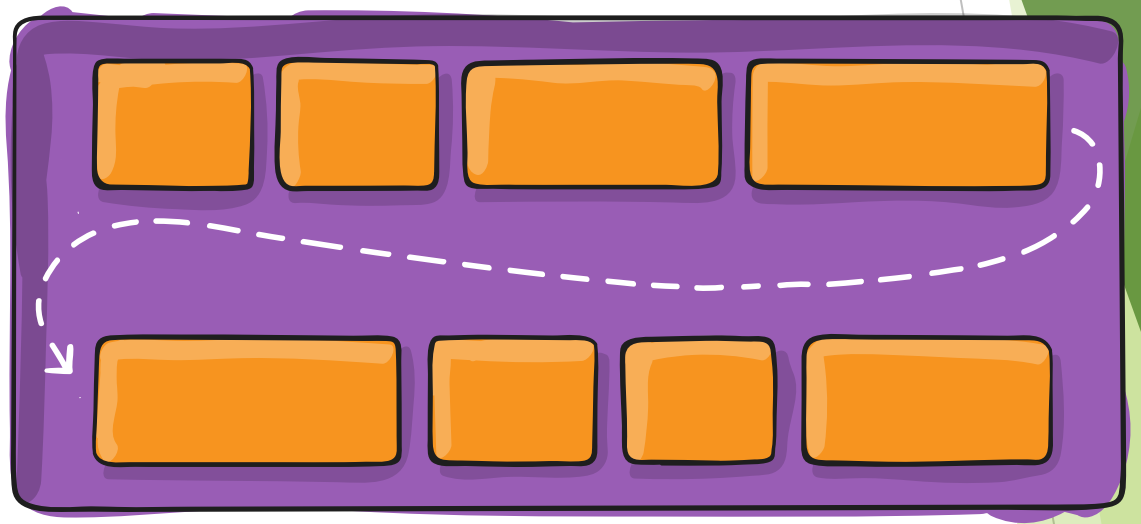
flex-direction

- Values
  - Column
  - column-reverse
  - Row
  - row-reverse



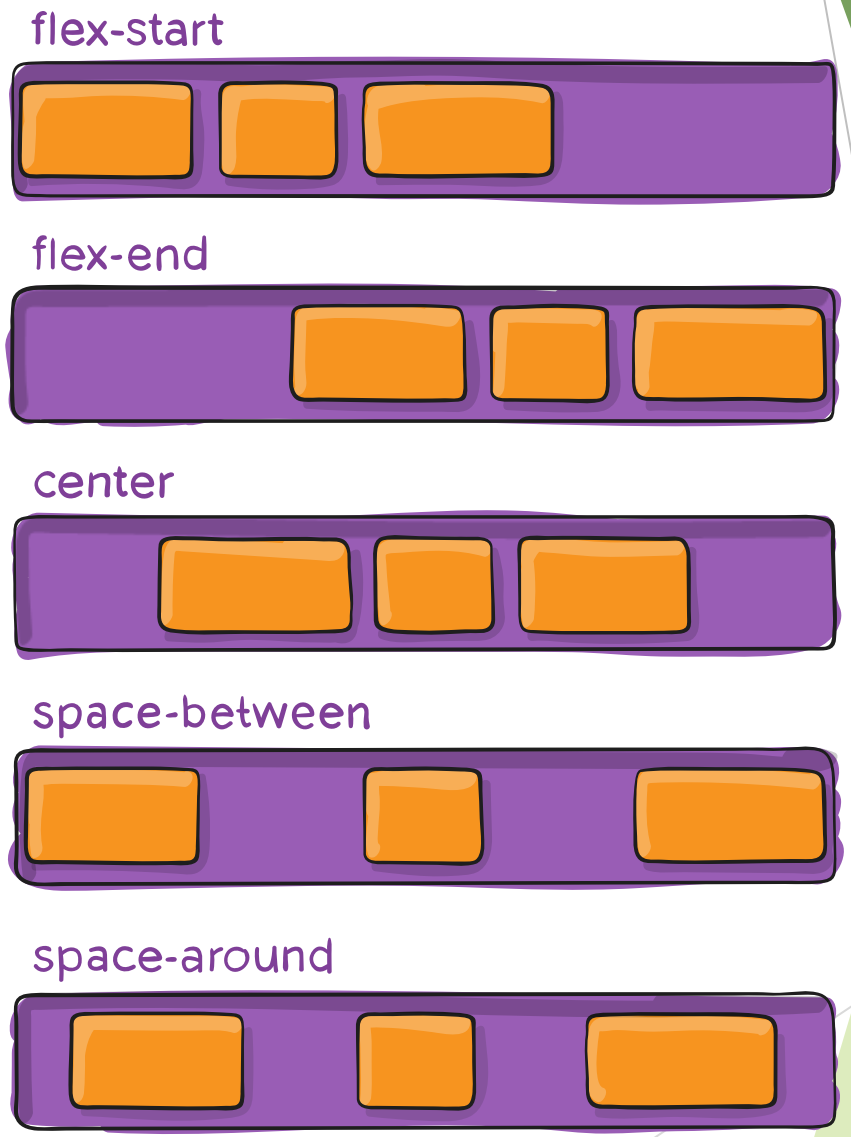
flex-wrap

- Values
  - wrap
  - nowrap
  - wrap-reverse



justify-content

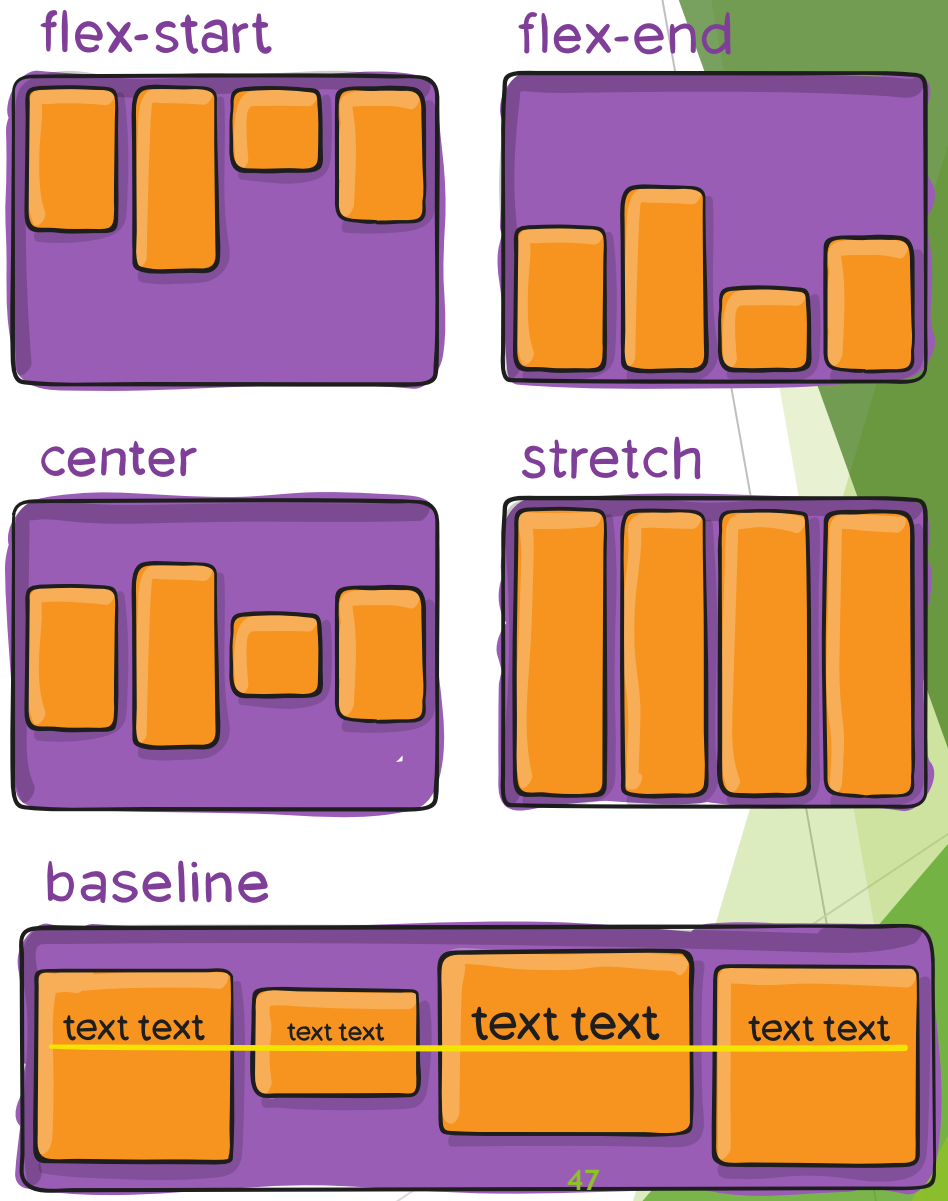
- Align items horizontally
- Values
  - center
  - flex-start
  - flex-end
  - space-around
  - space-between



align-items

- The align-items property is used to align the flex items vertically.

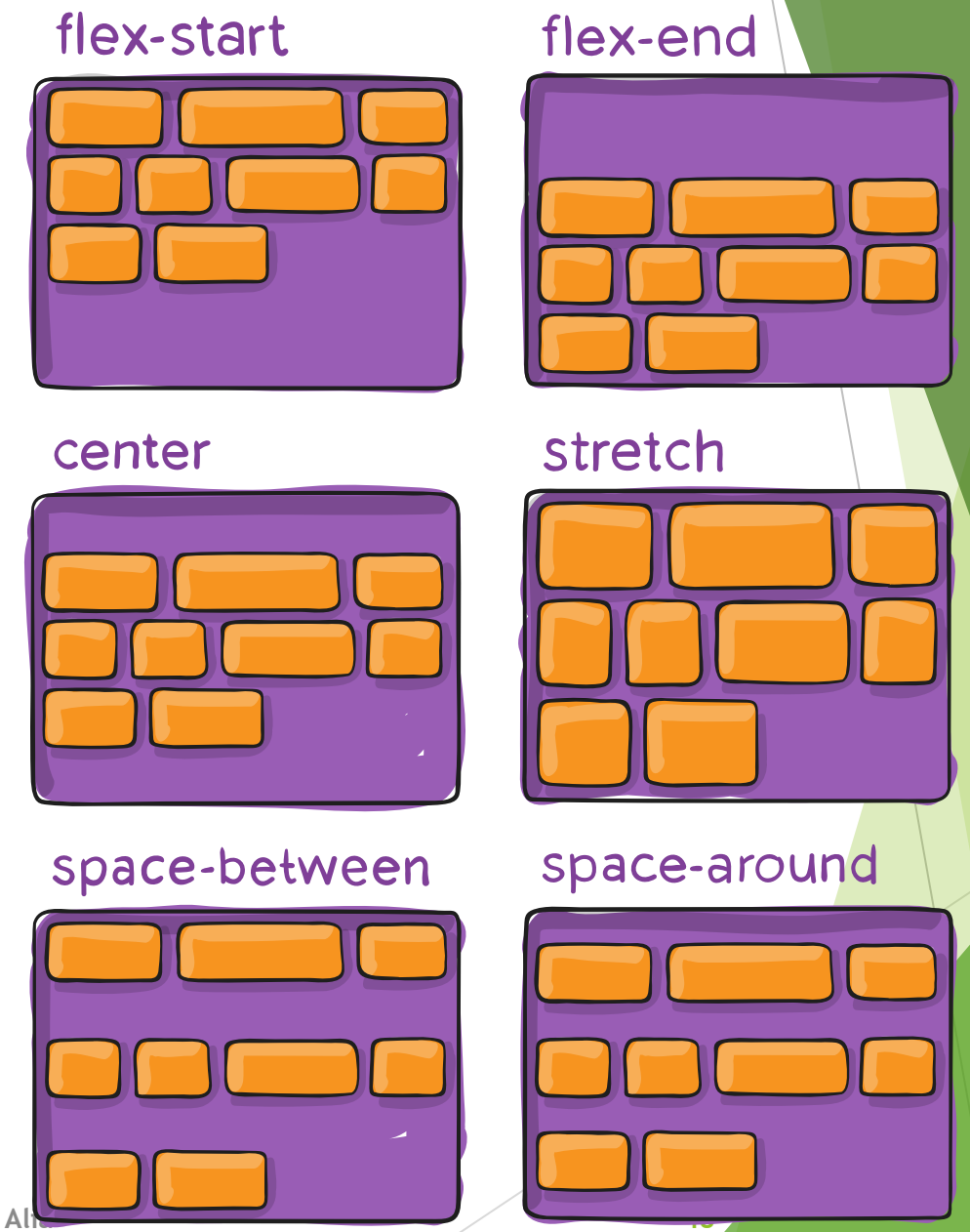
- Values
  - center
  - flex-start
  - flex-end
  - Stretch (default)



align-content

- Used to align flex lines
- Values
  - Space-between
  - Space-around
  - Stretch (default)
  - center

the flex-wrap property set to *wrap*, to better demonstrate the `align-content` property.



Flex Item or Child Elements

The flex item properties are:

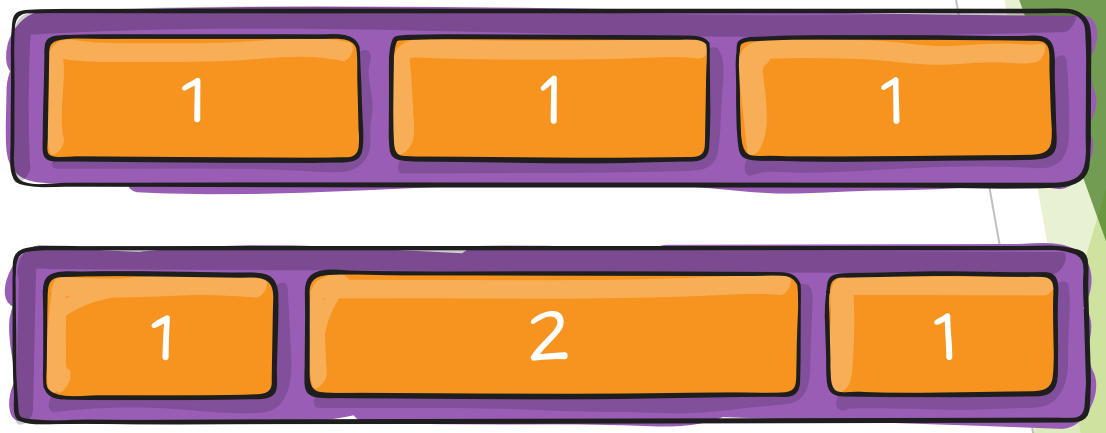
- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

Flex order (Flex Item) or Child Elements

```
<div class="flex-container">
  <div style="order: 3">1</div>
  <div style="order: 2">2</div>
  <div style="order: 4">3</div>
  <div style="order: 1">4</div>
</div>
```

Flex grow(Flex Item)

- how much it grow
- flex-grow:number default value is 0
- This defines the ability for a flex item to grow if necessary.
- If all items have flex-grow set to 1, the remaining space in the container will be distributed equally to all children.




```
<div class="flex-container">
  <div style="flex-grow: 1">1</div>
  <div style="flex-grow: 1">2</div>
  <div style="flex-grow: 8">3</div>
</div>
```

Flex shrink (Flex Item)

- This defines the ability for a flex item to shrink if necessary.

```
.item {
 flex-shrink: <number>; /* default 1 */
}
```

```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div style="flex-shrink: 0">3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
</div>
```

Flex basis

- specifies initial length

  If set to 0, the extra space around content isn't factored in.

```
<div class="flex-container">
  <div>1</div>
  <div>2</div>
  <div style="flex-basis: 200px">3</div>
  <div>4</div>
</div>
```

Note

- All the div inside a flex container are flexible and are flex items
- The flex item properties are:
  - Order (set the order of flex order:number)
  - flex-grow (how much it grow flex-grow:number default value is 0)
  - flex-shrink ()
  - flex-basis (specifies initial length)
  - Flex (shorthand property for the flex-grow, flex-shrink, and flex-basis)
  - align-self (override the default alignment)

Flex Responsive

```
/* Responsive layout - makes a one column-layout instead of two-column layout */
@media (max-width: 800px) {
  .flex-container {
    flex-direction: column;
  }
}
```

HTML5 Style Guide and Coding Conventions

- Use Correct Document Type
- Use Lower Case Element Names
- Close All HTML Elements
- Close Empty HTML Elements
- Use Lower Case Attribute Names
- Quote Attribute Values
- Image Attributes(Alt)
- Spaces and Equal Signs
  - HTML5 allows spaces around equal signs. But space-less is easier to read and groups entities better together.
  - <link rel="stylesheet" href="styles.css">
- Use Lower Case File Names

HTML Multimedia

- Multimedia on the web is sound, music, videos, movies, and animations.
  - Html Audio
  - Html Video

The HTML <video> Element

- <video width="320" height="240" controls autoplay muted>
- <source src="m.mp4" type="video/mp4">
- <source src="mo.ogg" type="video/ogg">
- Your browser does not support the video tag.
- </video>

The HTML <audio> Element

- <audio controls autoplay muted>
- <source src="abc.ogg" type="audio/ogg">
- <source src="abc.mp3" type="audio/mpeg">
- Your browser does not support the audio element.
- </audio>

- Auto play works on muted videos only

HTML YouTube Videos

- <iframe width="420" height="315"
  src="https://www.youtube.com/embed/Yf39EqXYiy8">
  </iframe>

- YouTube Autoplay
  - <iframe width="420" height="315"
    src="https://www.youtube.com/embed/XGSy3_Czz8k?autoplay=0">
    </iframe>

YouTube Autoplay

- <iframe width="420" height="315" src="https://www.youtube.com/embed/tgbNymZ7vqY?autoplay=1&mute=1"> </iframe>

- **Add mute=1 after autoplay=1 to let your video start playing automatically (but muted).**

YouTube Playlist

<iframe width="420" height="315"
src="https://www.youtube.com/embed/tgbNymZ7vqY?list=tgbNymZ7vqY
&">
</iframe>

- **YouTube Loop**
  - Value 0 (default): The video will play only once.
  - Value 1: The video will loop (forever).
- <iframe width="420" height="315"
  src="https://www.youtube.com/embed/tgbNymZ7vqY?loop=1">
  </iframe>

YouTube Controls

- Value 0: Player controls does not display.
- Value 1 (default): Player controls display

- <iframe width="420" height="315" src="https://www.youtube.com/embed/XGSy3_Czz8k? controls=0 </iframe>

HTML Responsive Web Design Introduction

- What is Responsive Web Design?
  - Responsive Web Design is about using HTML and CSS to automatically resize, hide, shrink, or enlarge, a website, to make it look good on all devices (desktops, tablets, and phones):

Setting The Viewport

- <meta name="viewport" content="width=device-width, initial-scale=1.0">

Responsive images

- Responsive images are images that scale nicely to fit any browser size.
- Use the width Property
  - If the CSS width property is set to 100%, the image will be responsive and scale up and down:
- Use the max-width Property
  - If the max-width property is set to 100%, the image will scale down if it has to, but never scale up to be larger than its original size:
  - <img src="img.jpg" style="max-width:100%;height:auto;">

Media queries

- In addition to resize text and images, it is also common to use media queries in responsive web pages.
- With media queries you can define completely different styles for different browser sizes.

Media queries

```
@media screen and (max-width: 800px) {
  .footer {
    width: 100%; /* The width is 100%, when the viewport is 800px or
smaller */
  }


@media screen and (max-width: 400px) {
  .footer {
    width: 300px; /* The width is 100%, when the viewport is 800px or
smaller */
  }
```

Media queries

```
@media screen and (min-width: 800px) {
  .footer {
    width: 100%; /* The width is 100%, when the viewport is 800px or
larger */
  }


@media screen and (min-width: 400px) {
  .footer {
    width: 300px; /* The width is 100%, when the viewport is 800px or
larger */
  }
```

Responsive Web Design - Frameworks

- **Bootstrap**
- W3.css
- HTML5 Boilerplate. ...
- HTML KickStart. ...
- Montage HTML5 Framework. ...
- SproutCore. ...
- Zebra.

End Of Lecture 7 & 8