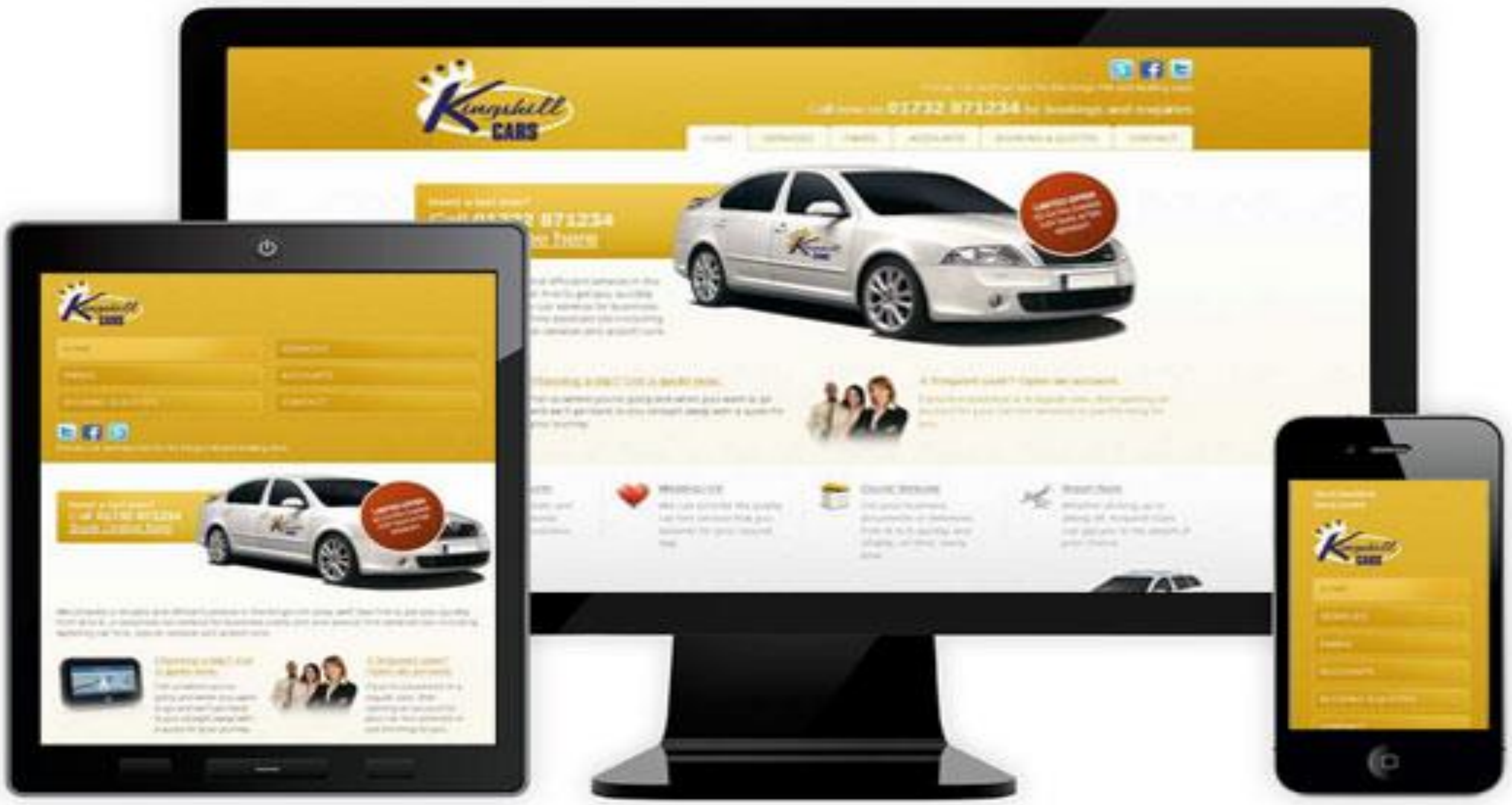
Creating Responsive Websites

ITI – Assiut Branch
Eng. Hany Saad

Responsive Design…

“The layout changes based on the size and capabilities of the device”

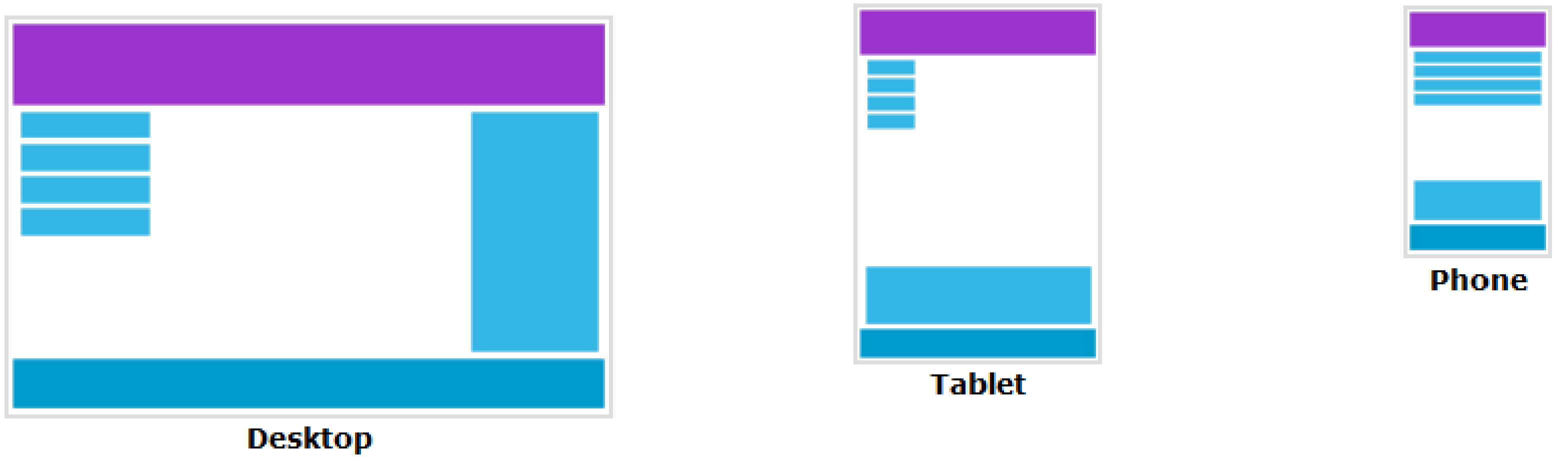


Responsive Design (Cont.)

- ❑ Responsive Web Design was first introduced by Ethan Marcotte in 2009
- ❑ An approach to web design that provides an optimal viewing experience across a wide range of devices.
- ❑ Think responsively.. Think Mobile First
- ❑ Mobile first design: Design for mobile, then for desktop.
- ❑ Responsive design is where website isn’t fixed with single size, It responds to users’ device automatically
- ❑ Adaptive design is where created website redesign itself as per the device size (Mostly different versions of the website).
- ❑ Negative space is empty space between elements to it more readable & standout
  - padding or margins are great strategy to create it

Responsive Design (Cont.)

- **Designing For The Best Experience For All Users:**
  - Web pages can be viewed using many different devices: desktops, tablets, and phones. Your web page should look good, and be easy to use, regardless of the device.



Guides to create responsive website

- **Guides:**
  - ✓ Use HTML5
  - ✓ Set the viewport.
  - ✓ Size content to the viewport
  - ✓ Use CSS layout to make the website responsive
  - ✓ Use CSS media queries for responsiveness

Set the Viewport

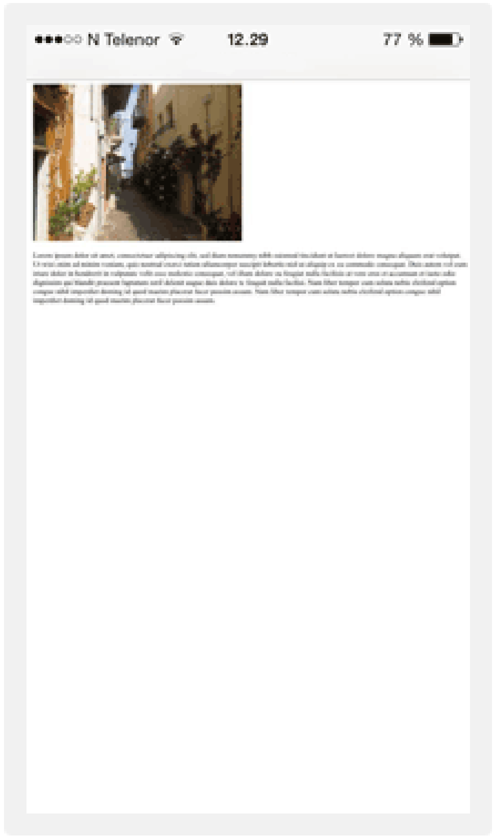
Set the viewport

- ❑ The viewport is the area to display website independent on device screen .
- ❑ Pages optimized for a variety of devices must include a meta viewport element in the head of the document.
- ❑ A meta viewport tag gives the browser instructions on how to control the page's dimensions and scaling.
- ❑ Example:

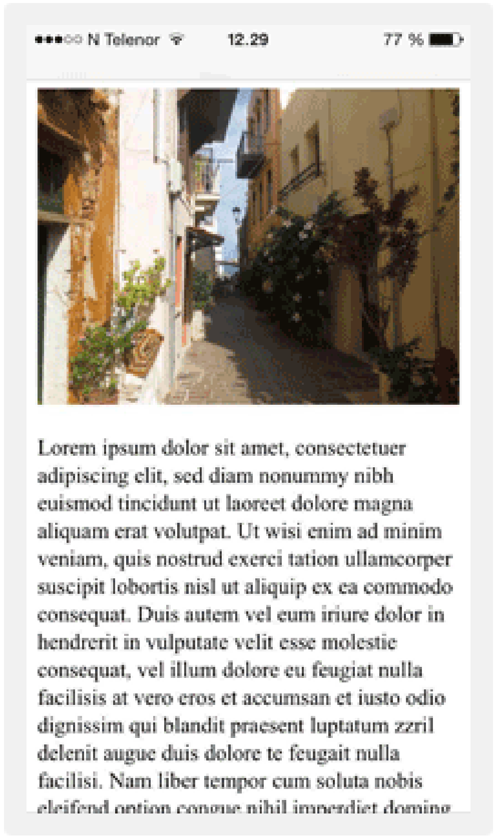
```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

- ❑ Include width=device-width to match the screen's width in device independent pixels.
- ❑ Include initial-scale=1 to establish a 1:1 relationship between CSS pixels and device independent pixels.
- ❑ Ensure your page is accessible by not disabling user scaling (don’t use: minimum-scale, maximum-scale, user-scalable).

Set the viewport (Cont.)



Without the viewport meta tag

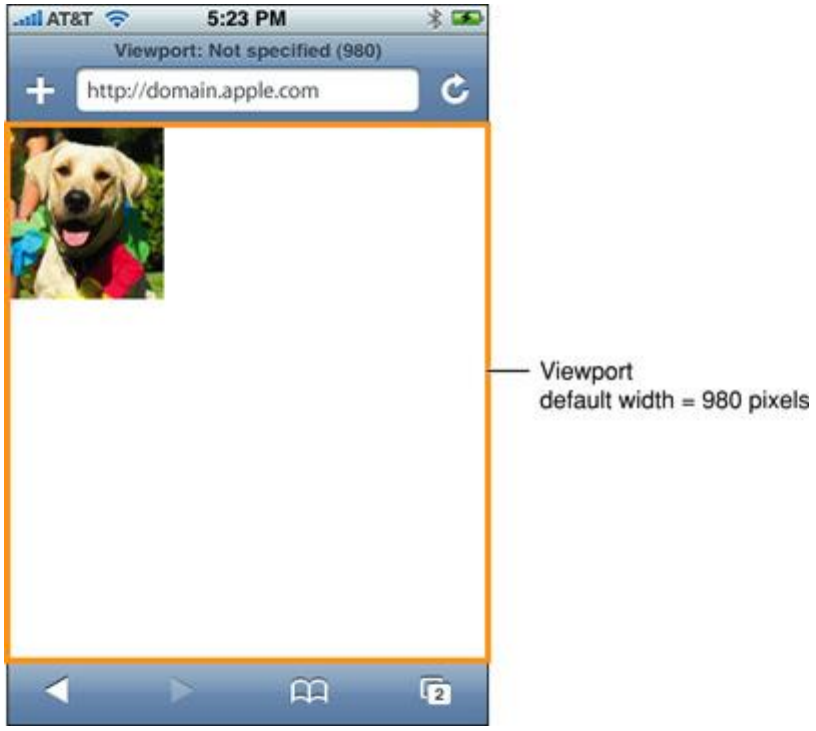


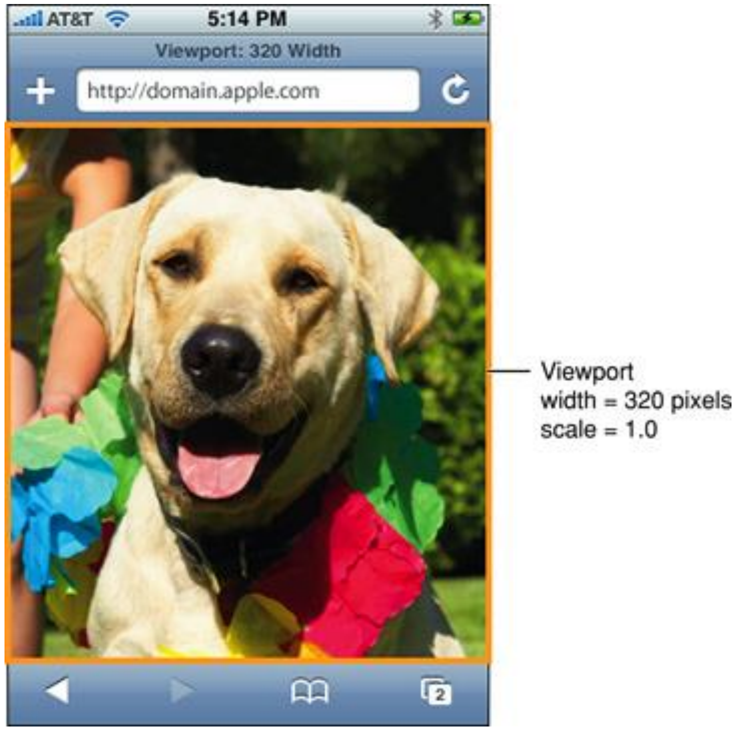
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla facilisi. Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming

With the viewport meta tag

Set the viewport (Cont.)

❑ viewport Examples:



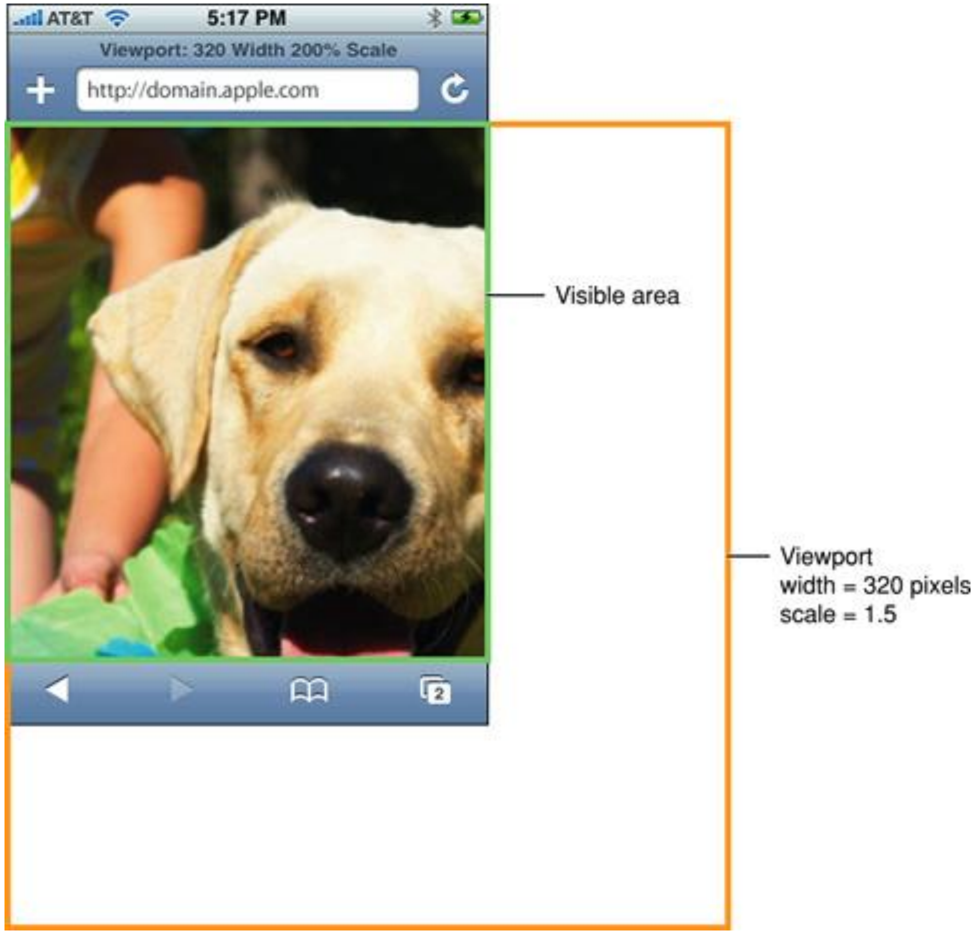


shows a webpage on iPhone, containing a single 320 x 356 pixel image, that is rendered for the first time using the default viewport settings.

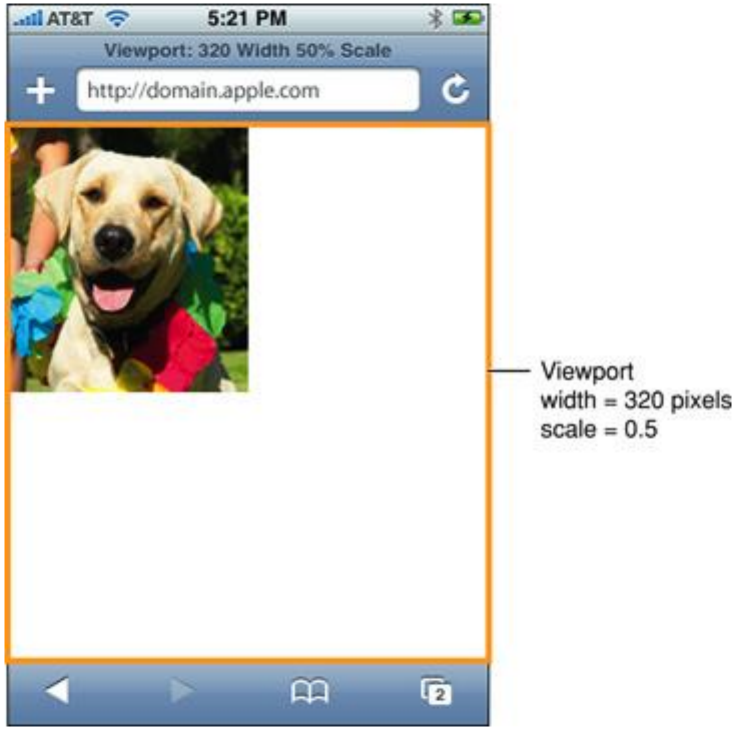
shows the same webpage with the viewport set to the size of the visible area, which is also the size of the image.

Set the viewport (Cont.)

❑ viewport Examples:



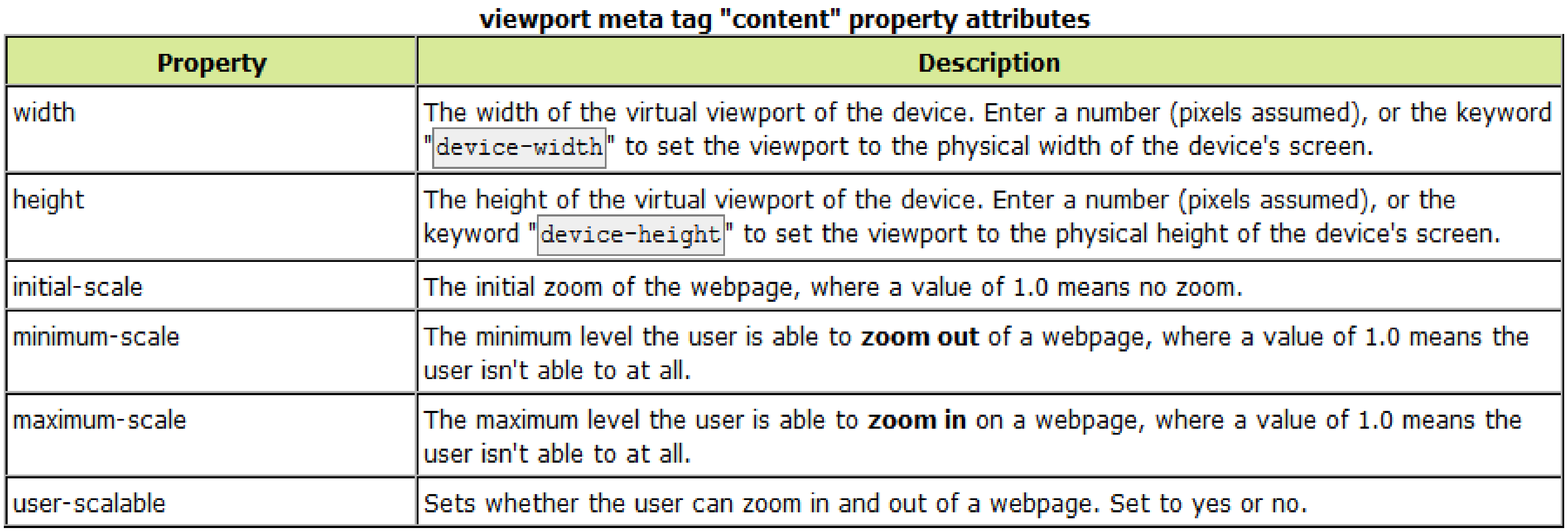
Viewport with width set to 320 and scale set to 150%



Viewport with width set to 320 and scale set to 50%

Set the viewport (Cont.)

❑ viewport meta tag "content" property attributes

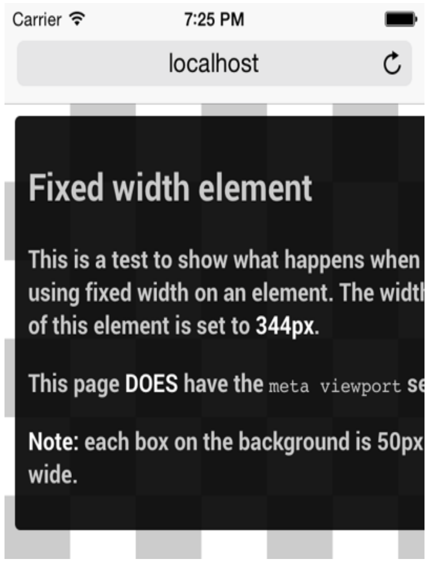


viewport meta tag "content" property attributes

Property	Description
width	The width of the virtual viewport of the device. Enter a number (pixels assumed), or the keyword "`device-width`" to set the viewport to the physical width of the device's screen.
height	The height of the virtual viewport of the device. Enter a number (pixels assumed), or the keyword "`device-height`" to set the viewport to the physical height of the device's screen.
initial-scale	The initial zoom of the webpage, where a value of 1.0 means no zoom.
minimum-scale	The minimum level the user is able to **zoom out** of a webpage, where a value of 1.0 means the user isn't able to at all.
maximum-scale	The maximum level the user is able to **zoom in** on a webpage, where a value of 1.0 means the user isn't able to at all.
user-scalable	Sets whether the user can zoom in and out of a webpage. Set to yes or no.

Size content to the viewport

- Setting large absolute CSS widths for page elements), will cause the elements to be too wide for the viewport on a narrower device.
- Instead, consider using relative width values, such as width: 100%.
- Similarly, beware of using large absolute positioning values that may cause the element to fall outside the viewport on small screens.

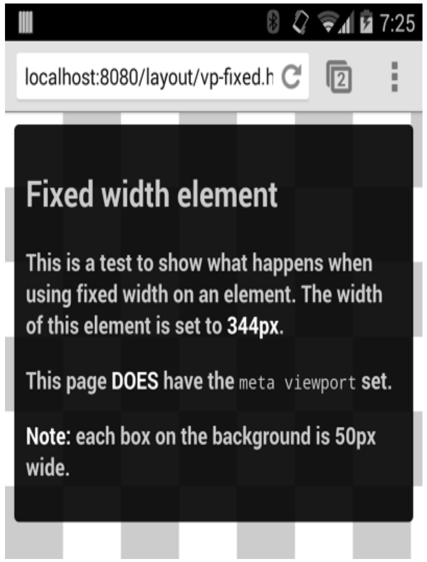

Carrier 7:25 PM
localhost

Fixed width element

This is a test to show what happens when using fixed width on an element. The widt of this element is set to **344px**.

This page **DOES** have the `meta viewport` se

Note: each box on the background is 50px wide.


7:25
localhost:8080/layout/vp-fixed.h

Fixed width element

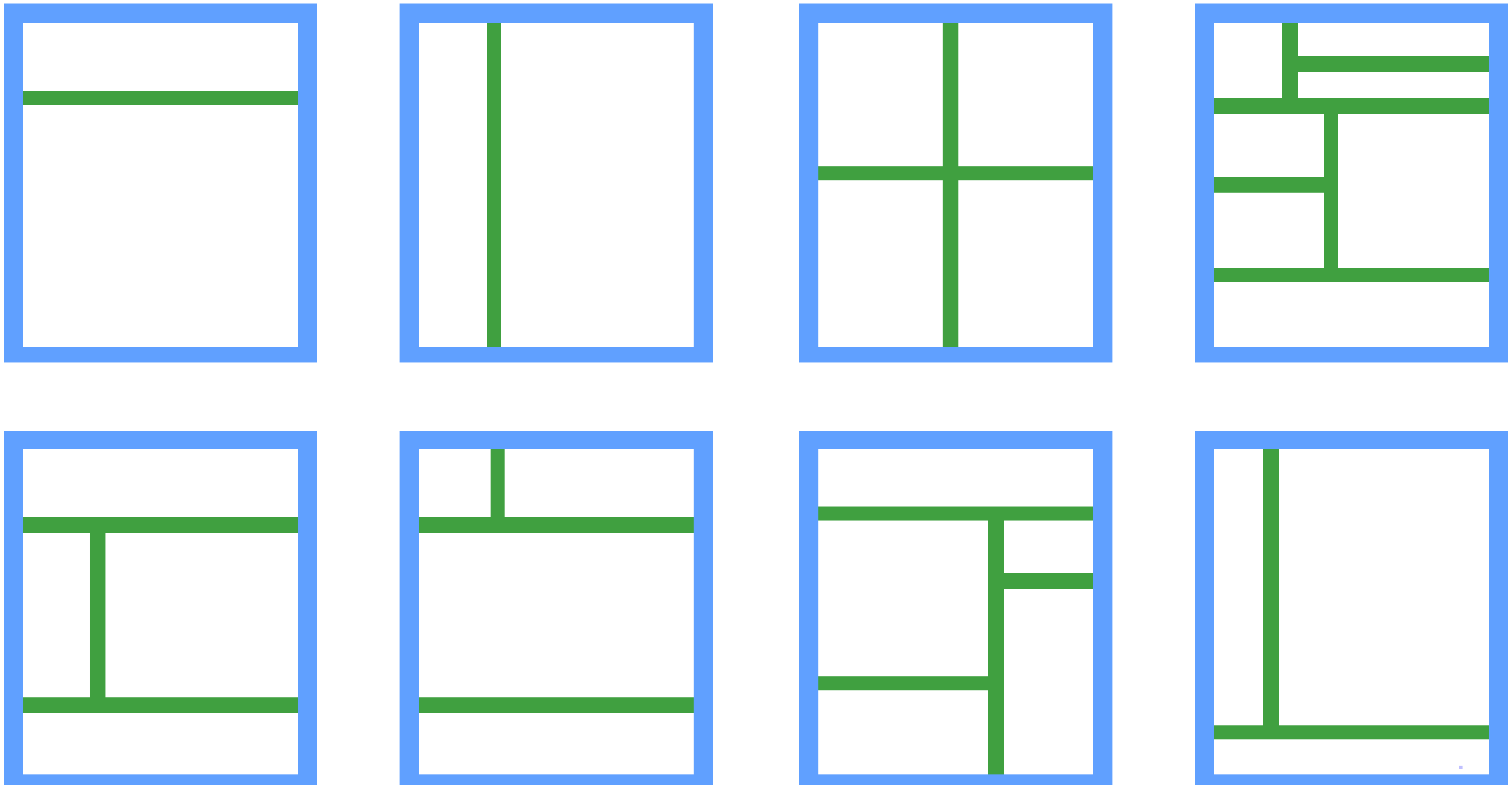
This is a test to show what happens when using fixed width on an element. The width of this element is set to **344px**.

This page **DOES** have the `meta viewport` set.

Note: each box on the background is 50px wide.

Website layout

Website Layout



Common HTML Layout Techniques

- ~~**HTML tables**~~
  - ~~The <table> element was not designed to be a layout tool! The purpose of the <table> element is to display tabular data. So, **do not use tables for your page layout**! They will bring a mess into your code. And imagine how hard it will be to redesign your site after a couple of months.~~

- **CSS float property**
  - It is common to do entire web layouts using the CSS float property. Floats is a technique that allows the elements to float to the left or right of one another, rather than the default of sitting on top of one another. Disadvantages: Floating elements are tied to the document flow, which may harm the flexibility

- **CSS flexbox**
  - Flexbox is a new layout mode in CSS3. Use of flexbox ensures that elements behave predictably when the page layout must accommodate different screen sizes and different display devices.

Common HTML Layout Techniques (Cont.)

- **Grid-based layout**
  - It makes your content more flexible and organized. It also makes your content easier to view and experience.
  - Some will use a 12-column grid, some will use a 16-column grid, some will use a 24-column grid, and others will use anything in between.

- **CSS framework**
  - Easy and fast.
  - Like: Bootstrap, foundation, W3,…

CSS Flexbox

- **The Flexible Box Layout**, makes it easier to design flexible responsive layout structure without having to use floats or positioning.
- **Flexbox Elements:**
  - A flexbox layout consists of **a parent element, with one or more child elements.**
  - The **parent** element becomes flexible by setting the **display property to flex**
  - The **flex-direction property** defines in which direction the container wants to stack the items.

CSS Flexbox (Cont.)

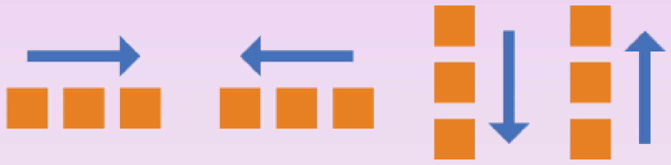
display

This defines a flex container; inline or block depending on the given value. It enables a flex context for all its direct children.

CSS

```css
.container {
  display: flex; /* or inline-flex */
}
```

flex-direction



This establishes the main-axis, thus defining the direction flex items are placed in the flex container. Flexbox is (aside from optional wrapping) a single-direction layout concept. Think of flex items as primarily laying out either in horizontal rows or vertical columns.

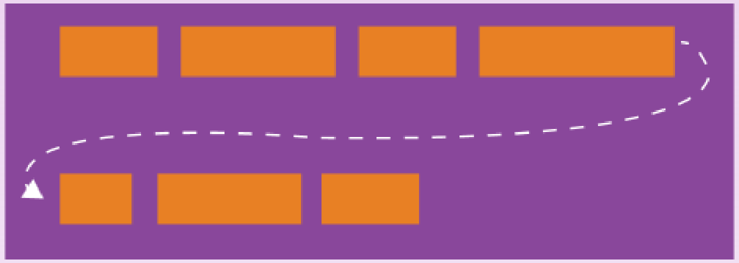
CSS

```css
.container {
  flex-direction: row | row-reverse | colum
}
```

- `row` (default): left to right in `ltr`; right to left in `rtl`
- `row-reverse`: right to left in `ltr`; left to right in `rtl`
- `column`: same as `row` but top to bottom
- `column-reverse`: same as `row-reverse` but bottom to top

CSS Flexbox (Cont.)

flex-wrap



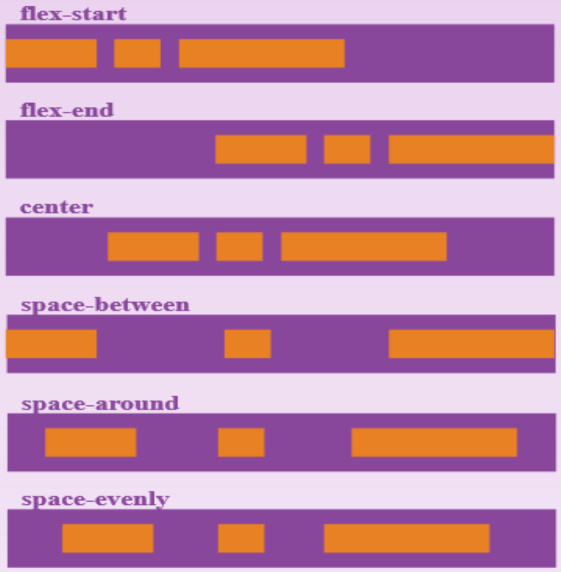
By default, flex items will all try to fit onto one line. You can change that and allow the items to wrap as needed with this property.

CSS

```css
.container{
  flex-wrap: nowrap | wrap | wrap-reverse;
}
```

- `nowrap` (default): all flex items will be on one line
- `wrap` : flex items will wrap onto multiple lines, from top to bottom.
- `wrap-reverse` : flex items will wrap onto multiple lines from bottom to top.

justify-content

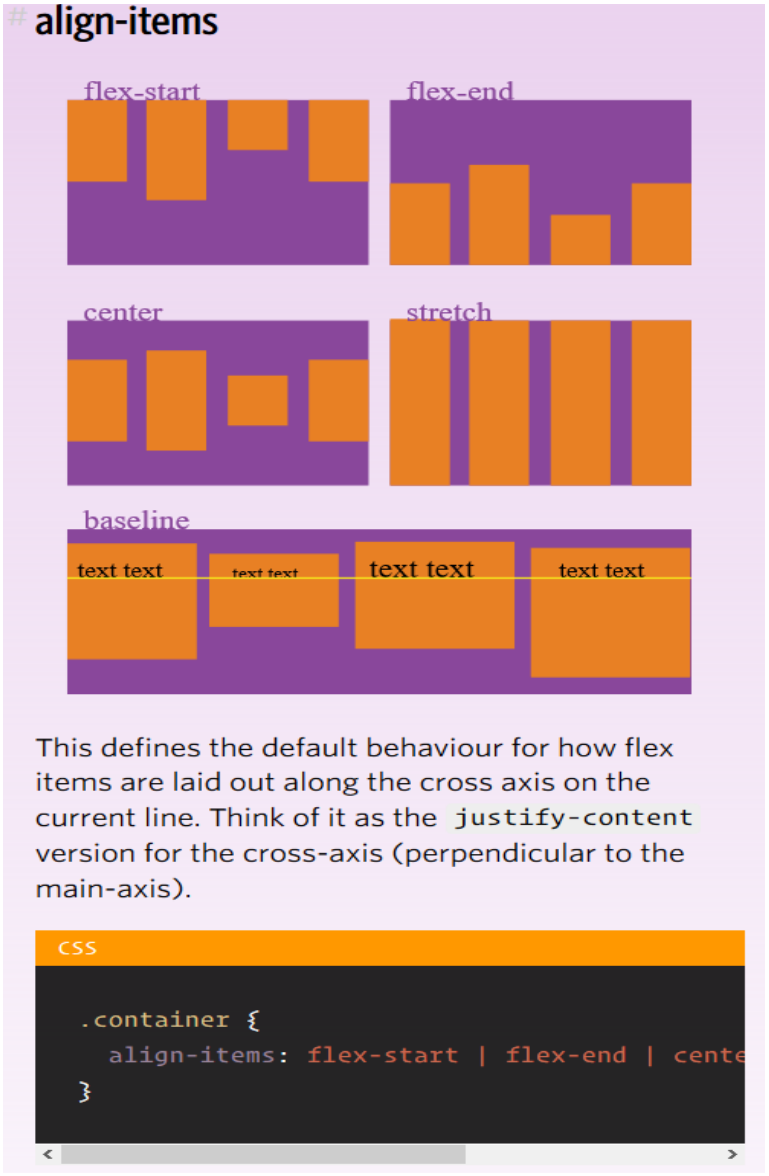


This defines the alignment along the main axis. It helps distribute extra free space left over when either all the flex items on a line are inflexible, or are flexible but have reached their maximum size. It also exerts some control over the alignment of items when they overflow the line.

CSS

```css
.container {
  justify-content: flex-start | flex-end | c
}
```

CSS Flexbox (Cont.)

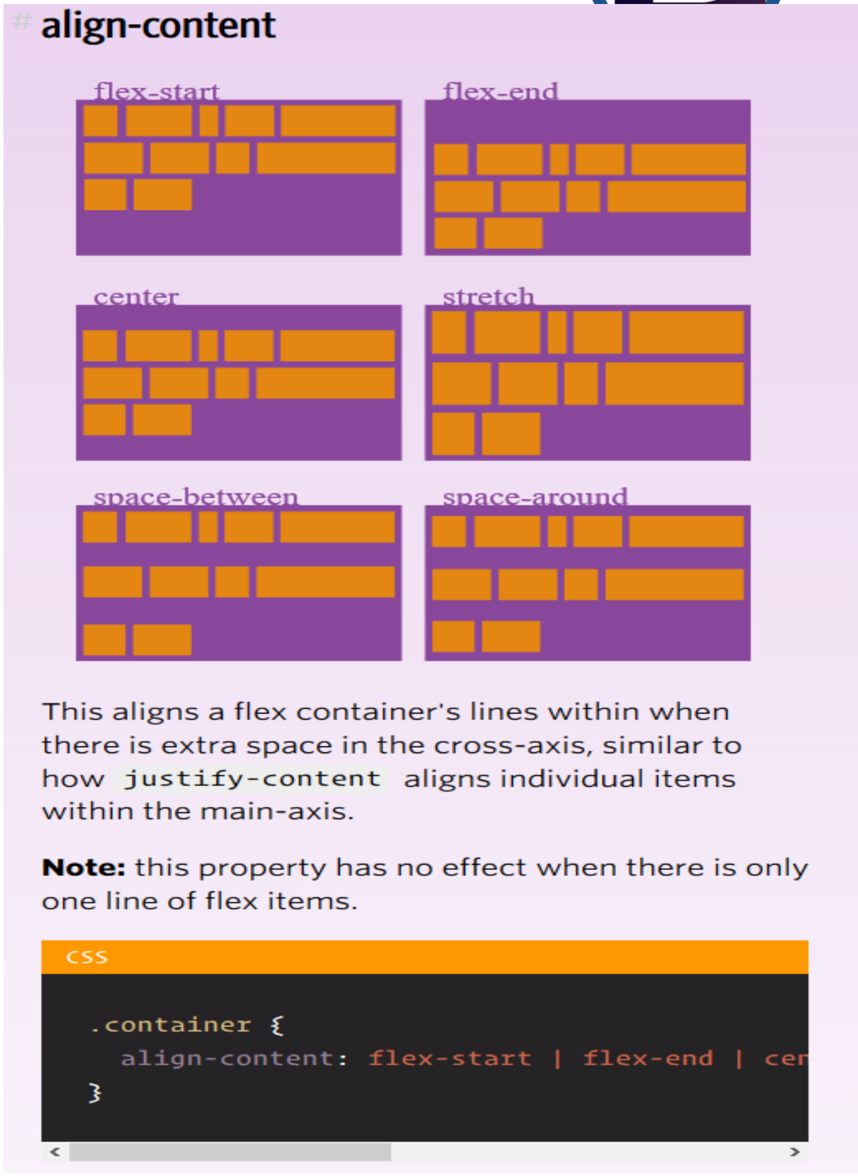


align-items

This defines the default behaviour for how flex items are laid out along the cross axis on the current line. Think of it as the `justify-content` version for the cross-axis (perpendicular to the main-axis).

CSS

```
.container {
  align-items: flex-start | flex-end | cente
}
```



align-content

This aligns a flex container's lines within when there is extra space in the cross-axis, similar to how `justify-content` aligns individual items within the main-axis.

Note: this property has no effect when there is only one line of flex items.

CSS

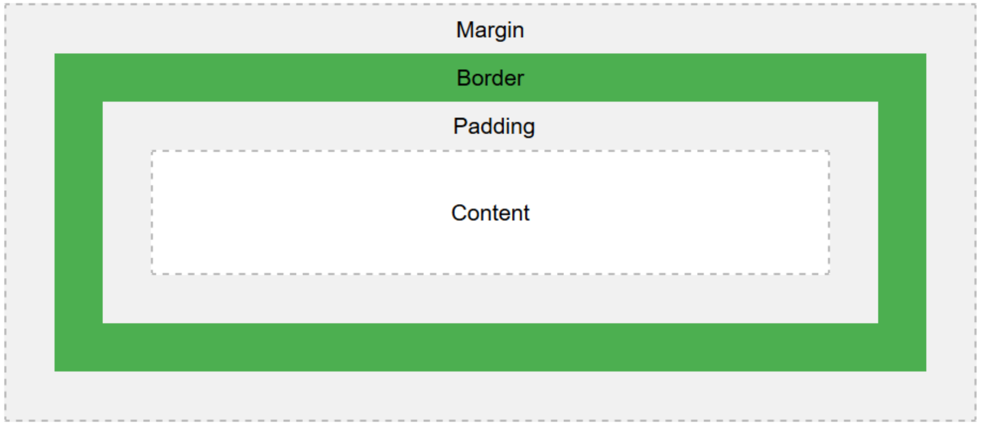
```
.container {
  align-content: flex-start | flex-end | cer
}
```

CSS Flexbox (Cont.)

- **Examples and more details:**
  - https://css-tricks.com/snippets/css/a-guide-to-flexbox/
  - https://www.w3schools.com/css/css3_flexbox.asp

CSS box-sizing Property

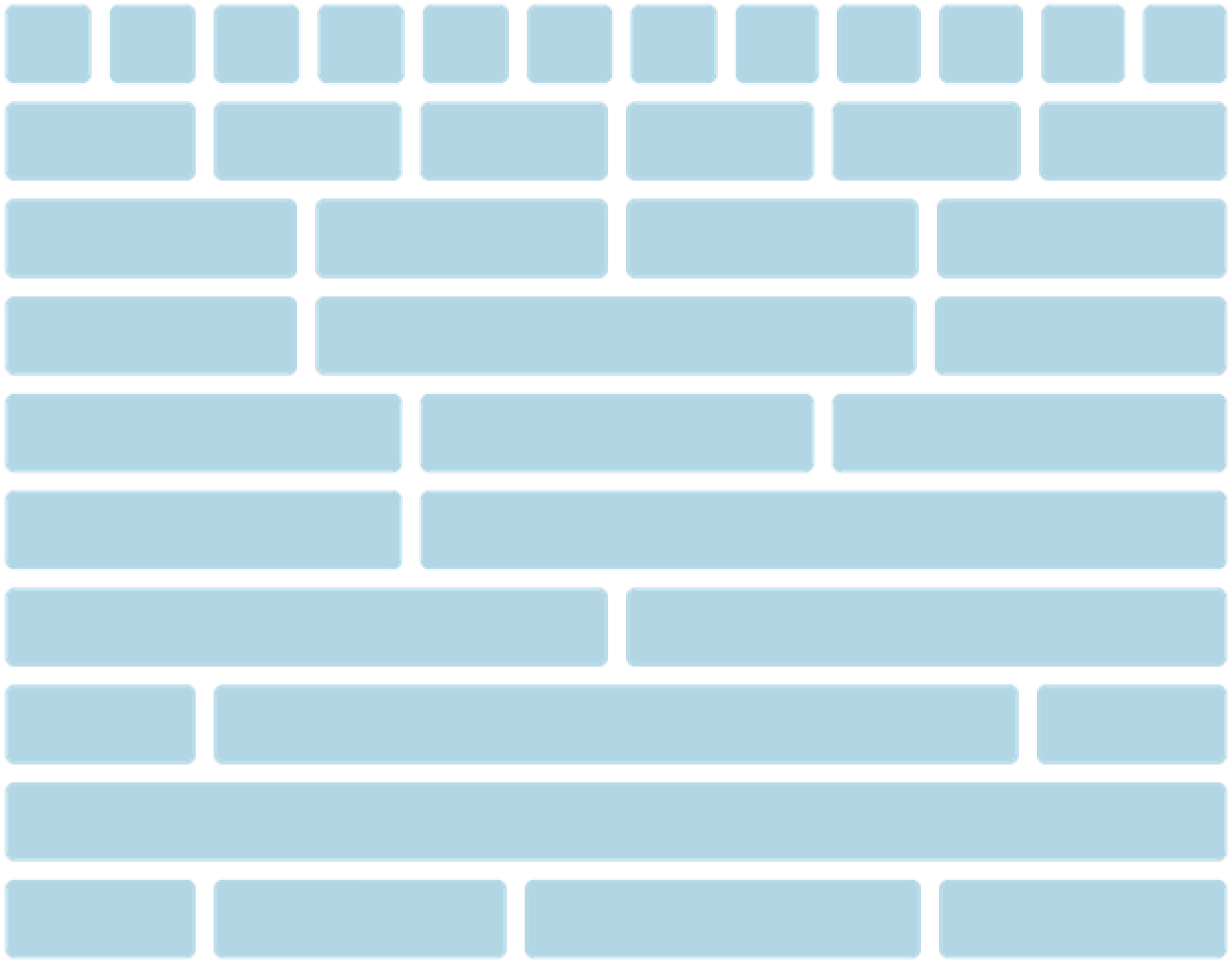
- The **box-sizing property** defines how **the width and height of an element are calculated: should they include padding and borders, or not**.
- **Property values:**
  - **content-box:** Default. The width and height properties (and min/max properties) **includes only the content. Border and padding are not included**.
  - **border-box:** The width and height properties (and min/max properties) **includes content, padding and border**.



- **Examples and more details:**
  - https://www.w3schools.com/css/css3_box-sizing.asp

Grid-based design

12 cols is a guideline for developing responsive css framework



1col→8.33%

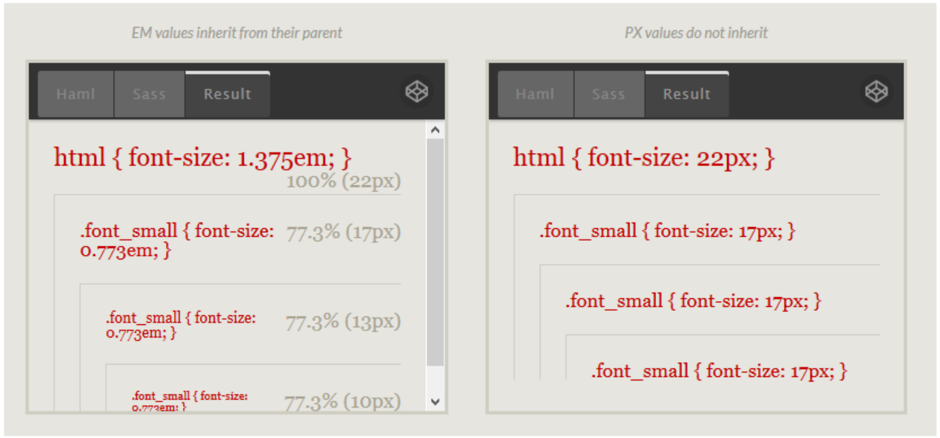
2col→16.66%

12col→100%

- **Examples and more details:**
  - https://www.w3schools.com/css/css_rwd_grid.asp

CSS Units to be used in responsive design

- **Relative units:**
  - **<u>Percent (%)</u>:** The percent unit is much like the “em” unit (Generally, 1em = 12pt = 16px = 100%.)
  - **<u>em</u>:** Relative to the font-size of the element (2em means 2 times the size of the current font).
  - **<u>Rem</u>:** Relative to font-size of the root element (html)
- **In Relative units:** each nested child assumes the parent is 1em(100%). Thus children inherit size by scaling in relation to the parent font size.



EM values inherit from their parent

```
html { font-size: 1.375em; }                100% (22px)
  .font_small { font-size: 0.773em; }       77.3% (17px)
    .font_small { font-size: 0.773em; }     77.3% (13px)
      .font_small { font-size: 0.773em; }   77.3% (10px)
```

PX values do not inherit

```
html { font-size: 22px; }
  .font_small { font-size: 17px; }
    .font_small { font-size: 17px; }
      .font_small { font-size: 17px; }
```

CSS Units to be used in responsive design (Cont.)

- **REM as in Root EM:**
  - While **em is relative to the font size of the element itself**, **rem is only relative to the html (root) font-size**.
  - Both **em and rem are flexible, scalable units which are translated by the browser into pixel values**, depending on the font size settings in your design.
    - Translation of **rem** units to pixel value is **determined by the font size of the html element**. This font size is influenced by inheritance from the browser font size setting unless explicitly overridden with a unit not subject to inheritance.
    - Translation of **em** units to pixel values **is determined by the font size of the element they’re used on**. This font size is influenced by inheritance from parent elements unless explicitly overridden with a unit not subject to inheritance.

CSS Units to be used in responsive design (Cont.)

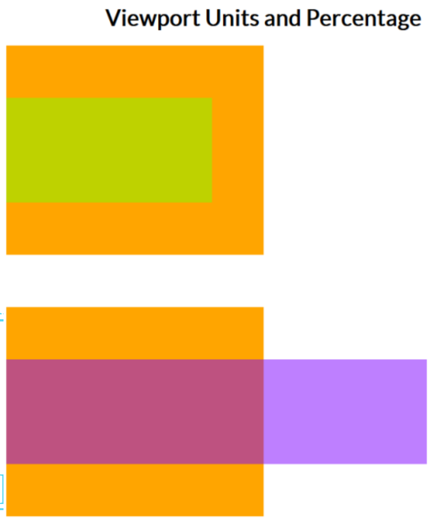
- **Viewport units:**
  - **<u>Viewport Width (vw)</u>:** A percentage of the **full viewport width**. 10vw will resolve to 10% of the current viewport width, or 48px on a phone that is 480px wide **(Equal to 1% of the width of the initial containing block)**.
    - The difference between % and vw: **A % length is relative to local context (containing element) width, while a vw length is relative to the full width** of the browser window.
  - **<u>Viewport Height (vh)</u>:** A percentage of the full viewport height. 10vh will resolve to 10% of the current viewport height (Equal to 1% of the height of the initial containing block).

CSS Units to be used in responsive design (Cont.)

- **Viewport units (Cont.):**
  - **Viewport Minimum (vmin):** A percentage **of the viewport width or height, whichever is smaller**. 10vmin will resolve to 10% of the current viewport width in portrait orientations, and 10% of the viewport height on landscape orientations.
  - **Viewport Maximum (vmax):** A percentage of the viewport width or height, whichever is larger. 10vmax will resolve to 10% of the current viewport height in portrait orientations, and 10% of the viewport width on landscape orientations.

CSS Units to be used in responsive design (Cont.)

- **Things to Keep in Mind with Viewport units:**
  - The width of the child element is set to be equal to 80% of its parent's width. However, another child element has a width of 80vw, which makes it wider than its parent.
  - When the overflow property on the root element is set to auto, the browsers will assume that the scrollbars don't exist. This will make the elements slightly wider than you expect them to be.
  - You should **use percentages when setting width for block** elements so that the scrollbars don't interfere with the computation of their width.



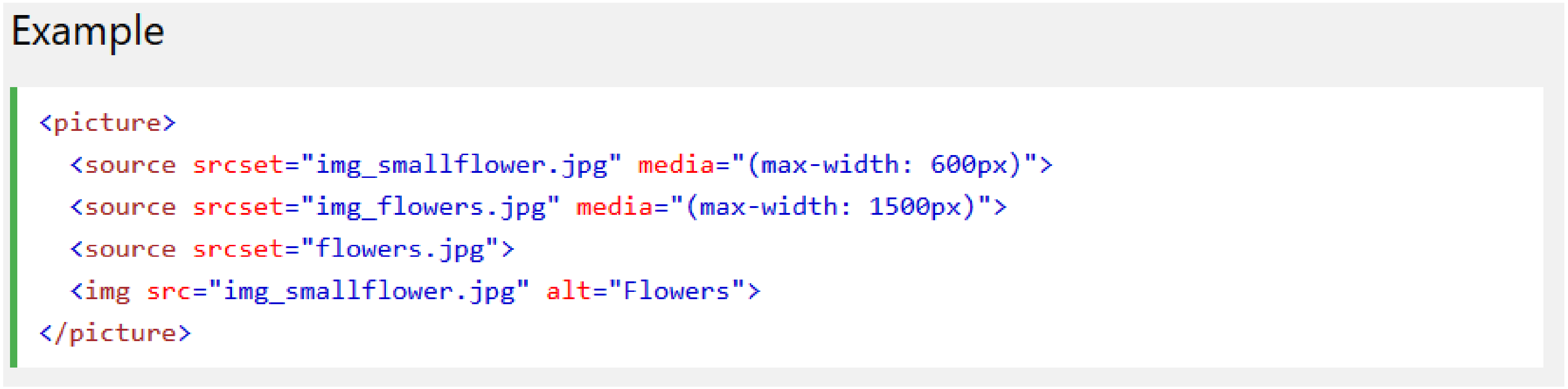
  - **More details:**
    - https://www.w3schools.com/cssref/css_units.asp
    - https://webdesign.tutsplus.com/articles/7-css-units-you-m about--cms-22573
    - https://css-tricks.com/fun-viewport-units/
    - https://webdesign.tutsplus.com/tutorials/comprehensive-guide-w rem--cms-22984

Responsive elements

- **Responsive images and videos:**
  - **Using the width Property (%):** If the width property is set to 100%, the **image will be responsive and scale up and down**. Notice that in that case, the **image can be scaled up to be larger than its original size**. A better solution, in many cases, will be to use the max-width property instead.
  - **Using 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**.
  - Make images responsive: https://www.w3schools.com/css/css_rwd_images.asp

Responsive elements (Cont.)

- **Responsive images and videos (cont.):**
  - **Show Different Images Depending on Browser Width:** The HTML <picture> element allows you to define different images for different browser window sizes.



Example

```
<picture>
  <source srcset="img_smallflower.jpg" media="(max-width: 600px)">
  <source srcset="img_flowers.jpg" media="(max-width: 1500px)">
  <source srcset="flowers.jpg">
  <img src="img_smallflower.jpg" alt="Flowers">
</picture>
```

  - Demo: https://www.w3schools.com/tags/tag_picture.asp
- **Responsive typography (text):**
  - **Use responsive units:**
    - <p style="**font-size:10vw**">Hello World</p>
- **Centering Elements:**
  - https://www.w3schools.com/css/css_align.asp
- **Responsive Web Design Patterns:**
  - https://developers.google.com/web/fundamentals/design-and-ux/responsive/patterns

CSS media queries

Use CSS media queries

- ❑ **The @media rule is used to define different style rules for different media types/devices.**
- ❑ **Media queries look at the capability of the device, and can be used to check many things, such as:**
  - ✓ width and height of the browser window
  - ✓ width and height of the device
  - ✓ orientation (is the tablet/phone in landscape or portrait mode?)
  - ✓ resolution
  - ✓ and much more

Use CSS media queries (Cont.)

- **General Syntax:**

```
@media not|only mediatype and (media feature) {
    CSS-Code;
}
```

- **You can also have different stylesheets for different media:**

```
<link rel="stylesheet" media="mediatype and|not|only (media feature)"
href="mystylesheet.css">
```

- **Example: (between 520 and 699px, we're going to use that extra space that opens up in the sidebar)**

```
@media all and (max-width: 699px) and (min-width: 520px) {
  #sidebar ul li a {
    padding-left: 21px;
    color: #666;
  }
}
```

Use CSS media queries (Cont.)

- **Example: (calling an external stylesheet when media type is screen with specified width)**

```
<link rel='stylesheet' media='screen and (min-width: 701px) and (max-width: 900px)' href='css/medium.css' />
```

- **Demo:**

  http://googlesamples.github.io/web-fundamentals/samples/layouts/rwd-fundamentals/media-queries.html

Use CSS media queries (Cont.)

- **Common Media Types:**
  - ✓ All: Used for all media type devices
  - ✓ Screen: Used for computer screens, tablets, smart-phones etc.

- **Common Media Features:**
  - ✓ Width: Specifies the width of the display area, such as a browser window.
  - ✓ min-width: Specifies the minimum width of the display area, such as a browser window.
  - ✓ min-device-width: Specifies the minimum width of the device, such as a computer screen.
  - ✓ Orientation: Specifies the whether the display is in landscape mode or portrait mode.

- **Other Media types and Features:**
  - ✓ http://www.w3schools.com/cssref/css3_pr_mediaquery.asp

How to choose breakpoints

- Create breakpoints based on content, never on specific devices.
- Design for the smallest mobile device first, then progressively enhance the experience as more screen real estate becomes available.
- To insert a breakpoint at 600px (for example), create two new stylesheets, one to use when the browser is 600px and below, and one for when it is wider than 600px.

```
<link rel="stylesheet" href="weather.css">

<link rel="stylesheet" media="(max-width:600px)" href="weather-2-small.css">

<link rel="stylesheet" media="(min-width:601px)" href="weather-2-large.css">
```

How to choose breakpoints (Cont.)

- Pick minor breakpoints between major breakpoints when necessary:

```css
@media (min-width: 360px) {
  body {
    font-size: 1.0em;
  }
}

@media (min-width: 500px) {
  .seven-day-fc .temp-low,
  .seven-day-fc .temp-high {
    display: inline-block;
    width: 45%;
  }
```

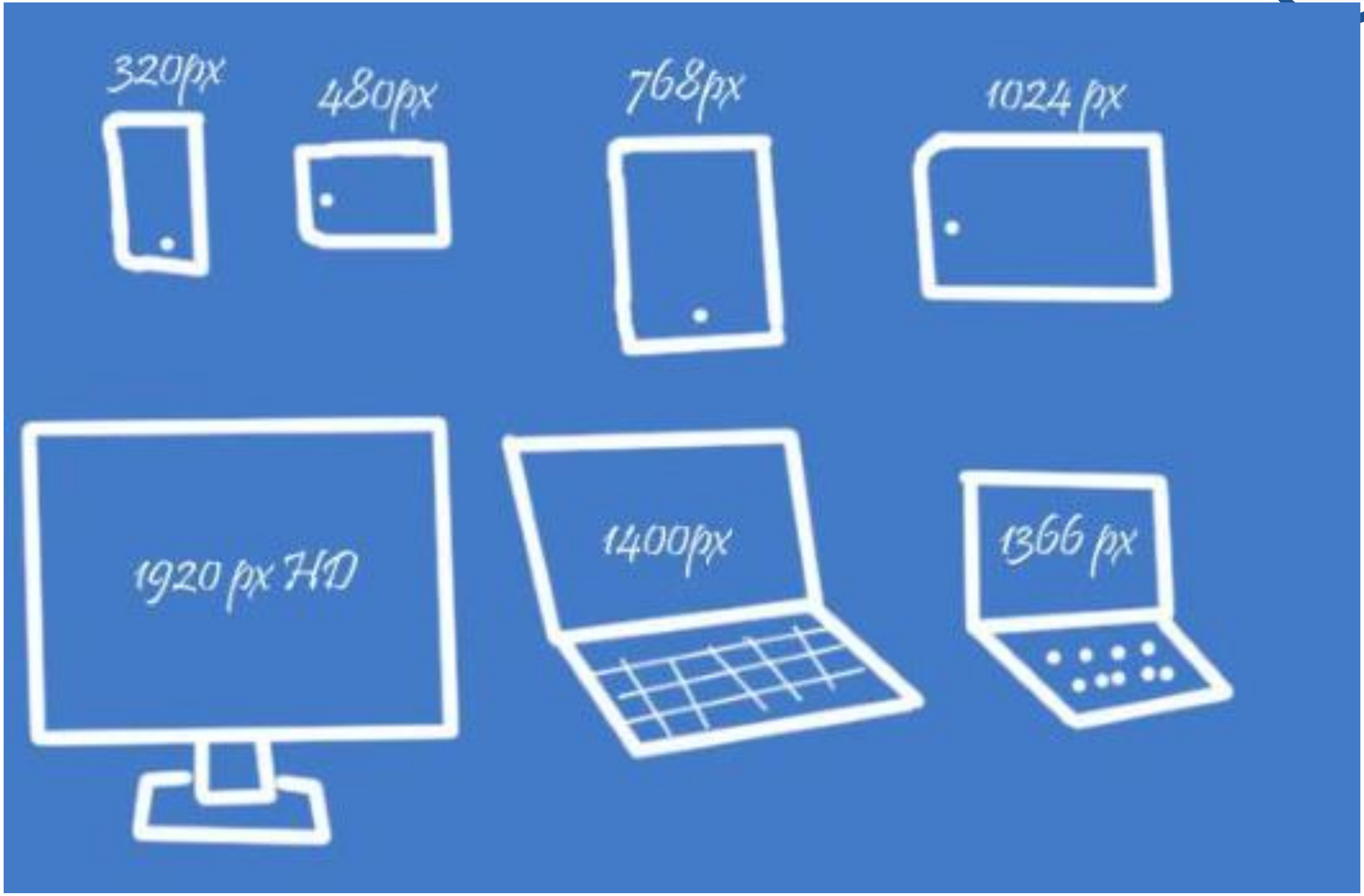
- Full Demo:

  http://googlesamples.github.io/web-fundamentals/samples/layouts/rwd-fundamentals/weather-2.html

- Responsive Web Design Patterns

  https://developers.google.com/web/fundamentals/design-and-ux/responsive/patterns

How to choose breakpoints (Cont.)



How to choose breakpoints (Cont.)

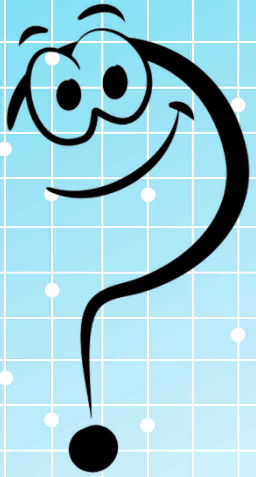
- Breakpoint is the moment when layout changes from layout to another
- Breakpoints are created with media query
- Usually triggered by viewport width
- Define breakpoint for layout and ensure that there is no overlap between them

Device	Width	Breakpoint expression
Smartphones	< 480px	(max-width:480px)
Portrait Tables	481px to 768px	(min-width:481px) and (max-width:768px)
Landscape Tablets| desktop	768px to 940px	(min-width:769px) and (max-width:940px)
Default	940px and up	(min-width:941px)
Large Screens	1210px and up	(min-width:480px)

CSS Frameworks & pre-processors

- **CSS Frameworks:** are pre-prepared software frameworks that are meant to allow for easier, more standards-compliant web design using the Cascading Style Sheets language.
  - Twitter BootStrap framework
    - http://getbootstrap.com
  - Foundation CSS framework
    - http://foundation.zurb.com/docs/

- **CSS pre-processors:** helps you write maintainable, future-proof code and it will seriously reduce the amount of CSS you have to write. Where these tools shine best are in large-scale user interfaces that require huge stylesheets and many style rules.
  - Sass (Syntactically Awesome Style Sheets):
    - http://sass-lang.com
  - Less
    - http://lesscss.org



Thanks….