

10.1 Mobile websites and browsers

Mobile web browsers

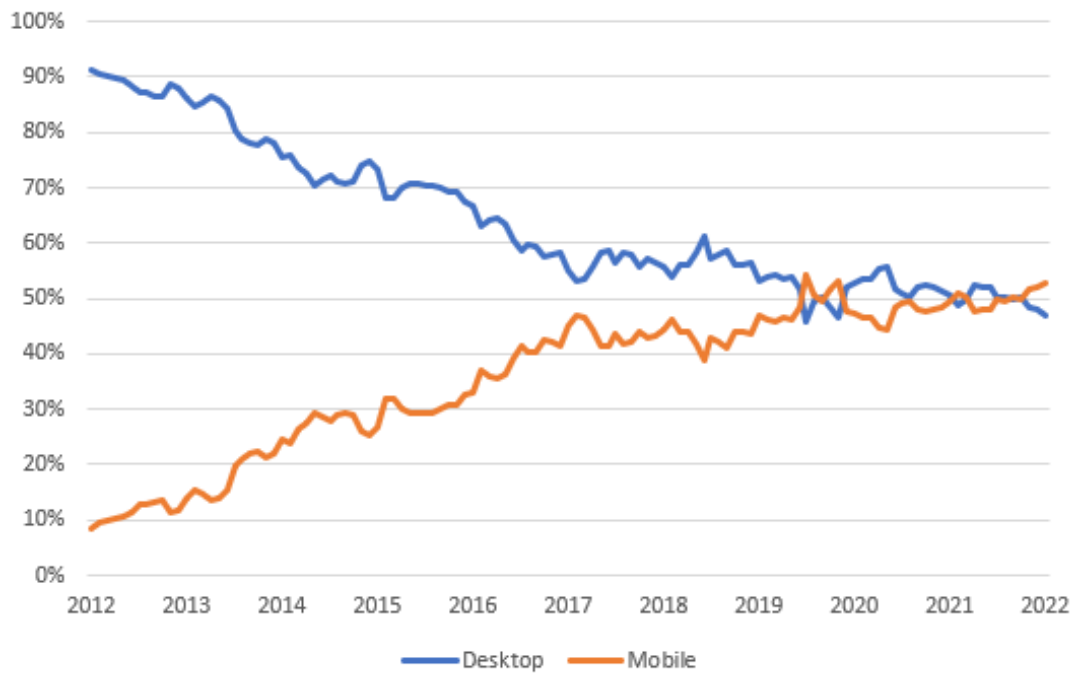
Prior to 2007, a number of markup languages were used to create simple webpages for first-generation smartphones and personal digital assistants (PDAs). Ex: HDML, WML, cHTML, iHTML, and XHTML-MP. The introduction of the iPhone in 2007 inaugurated the smartphone era and led to the development of innovative mobile web browsers that could render the same webpages designed for desktop web browsers. A **mobile web browser** is a web browser designed for mobile devices that can display webpages using HTML, CSS, and JavaScript.

Today's mobile browsers are much like their desktop counterparts, although mobile browsers often lack the ability to use plugins developed for desktop browsers. The popularity of smartphones has encouraged web developers to create mobile websites. A **mobile website** is a website that is designed for mobile devices with smaller screen sizes and touch interfaces.

Figure 10.1.1: CNN.com for desktop and mobile browsers.

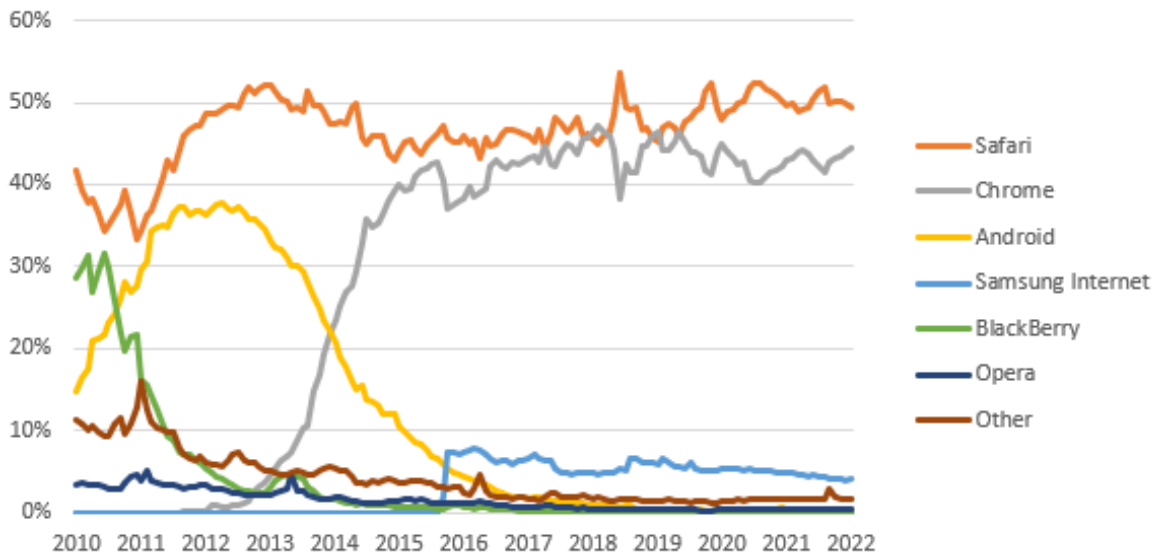


Figure 10.1.2: Desktop vs. mobile access in North America.



Source: [StatCounter.com](https://www.statcounter.com)

Figure 10.1.3: Top six mobile browsers in North America.



Source: [StatCounter.com](https://www.statcounter.com)

**PARTICIPATION
ACTIVITY**

10.1.1: Mobile web browsers.



Refer to the figures above.

1) According to StatCounter, the first time mobile browsers accounted for more web traffic than desktop browsers in North America was in 2017.



- ☐ True
- ☐ False

2) According to StatCounter, the Safari mobile web browser is most often used to access the web in North America.



- ☐ True
- ☐ False

3) From 2013 to 2016, Android browser usage steadily declined while Chrome increased.



- ☐ True
- ☐ False

Designing for mobile browsers

Mobile and desktop websites are created with the same technologies: HTML, CSS, and JavaScript. However, mobile websites differ from desktop websites in some significant ways. Designers of mobile websites must consider:

- Screen size is much smaller than desktops.
- User interaction is with touch, not a mouse.
- Mobile devices may have limited or slower Internet connectivity.
- Many users have data plans that limit how much content can be downloaded.
- Users might have a different purpose for accessing a website when using a mobile device vs.

using a desktop.

- Limited memory and CPU speed of mobile devices means mobile browsers are not as powerful as desktop browsers.
- Mobile websites may take advantage of GPS and accelerometer data.

**PARTICIPATION
ACTIVITY**

10.1.2: Designing mobile websites.



- 1) Mobile websites often show less information than their desktop counterparts.
☐ True
☐ False
- 2) Links are generally larger on mobile websites.
☐ True
☐ False
- 3) Mobile websites should not rely on mouse hovering to trigger actions on the website.
☐ True
☐ False
- 4) Phone numbers and addresses are typically of equal importance to desktop and mobile website users.
☐ True
☐ False
- 5) Navigation links are often just as visible on mobile websites as on desktop websites.
☐ True
☐ False



6) JavaScript does not run as quickly on a mobile web browser.

- ☐ True
- ☐ False

Implementing mobile websites

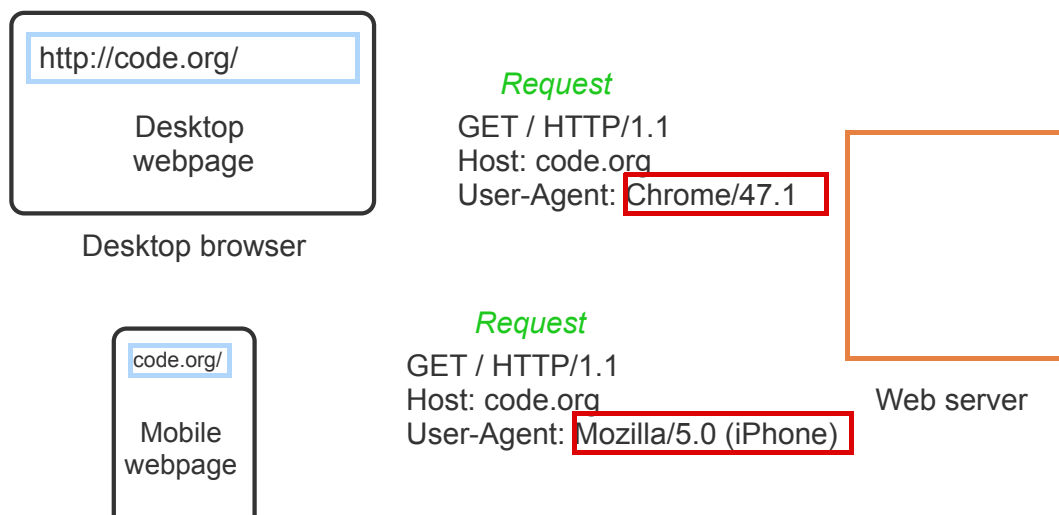
Developers implement mobile websites using three main techniques:

1. **Separate websites:** Two different websites are created, one for desktop and one for mobile.
2. **Dynamic serving:** The same URL is requested by both desktop and mobile web browsers, but the web server sends back a desktop version to desktop browsers and a mobile version to mobile browsers.
3. **Responsive web design:** The web server sends back the same HTML to both desktop and mobile browsers, but the browsers alter the appearance of the webpage to match the device size.

The first strategy is straightforward and easy to implement. Developers can create the desktop website using one set of URLs and the mobile site using another. Ex: `http://example.com/` vs. `http://example.com/mobile`. The second strategy is illustrated in the animation below.

PARTICIPATION ACTIVITY

10.1.3: Dynamic serving.





Mobile browser

Animation content:

A web server is displayed with a request of "GET/HTTP/1.1" with a host of "code.org" and a user-agent of "Chrome/47.1". This corresponds to a display of a desktop browser that shows the desktop webpage with the URL "http://code.org/". Another request is displayed as "GET/HTTP/1.1" with a host of "code.org" and a user-agent of "Mozilla/5.0 (iPhone)". This corresponds to a display of a mobile browser that shows the mobile webpage with the URL "http://code.org/".

Animation captions:

1. User types a URL into the desktop browser, and an HTTP request is sent to the web server.
2. Web server examines the User-Agent string indicating a desktop Chrome browser made the request, so the desktop page is returned.
3. User types a URL into the mobile browser, and an HTTP request is sent to the web server.
4. Web server examines the User-Agent string indicating an iPhone browser made the request, so the mobile page is returned.

Table 10.1.1: Example user agent strings for mobile browsers.

Mobile browser	User agent
Chrome on Nexus 10	Mozilla/5.0 (Linux; Android 4.3; Nexus 10 Build/JSS15Q) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/46.0.2490.76 Safari/537.36
Safari on iPhone	Mozilla/5.0 (iPhone; CPU iPhone OS 9_1 like Mac OS X) AppleWebKit/601.1.46 (KHTML, like Gecko) Version/9.0 Mobile/13B137 Safari/601.1
BlackBerry Z30	Mozilla/5.0 (BB10; Touch) AppleWebKit/537.10+ (KHTML, like Gecko) Version/10.0.9.2372 Mobile Safari/537.10+
MSIE on Nokia Lumia 520	Mozilla/5.0 (compatible; MSIE 10.0; Windows Phone 8.0; Trident/6.0; IEMobile/10.0; ARM; Touch; NOKIA; Lumia 520)

**PARTICIPATION
ACTIVITY**

10.1.4: Implementing mobile websites.

1) When implementing separate websites, a mobile browser accessing the desktop website may be redirected by the server to the mobile site automatically.

- ☐ True
- ☐ False

2) A mobile user should never have access to the desktop version of a website when a mobile version is available.

- ☐ True
☐ False

3) The web server detects mobile browsers by examining the User-Agent request header.

- ☐ True
☐ False

4) Keeping the content on a desktop website and a separate mobile website consistent can be a challenge.

- ☐ True
☐ False

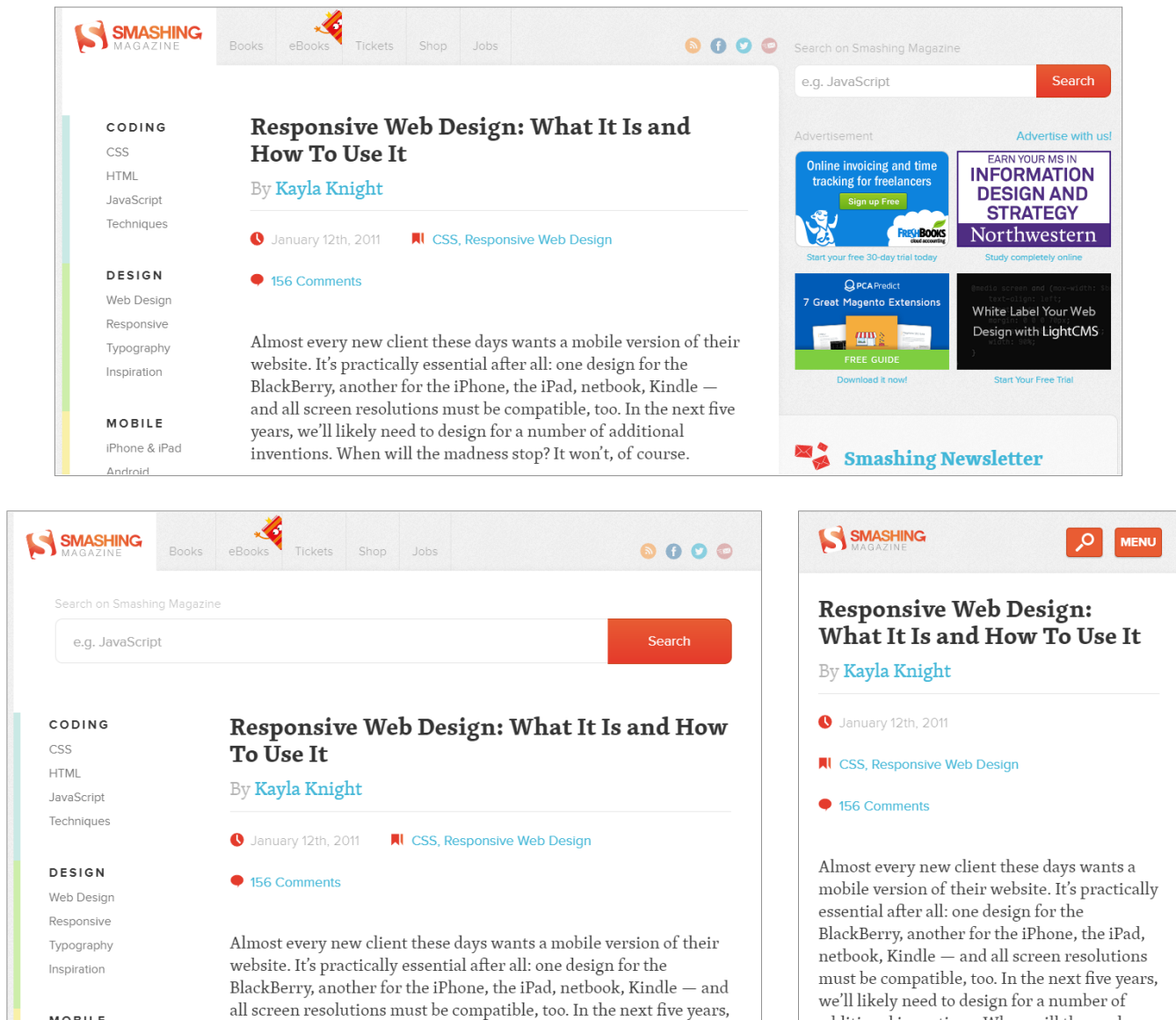
Responsive web design

Responsive web design (RWD) is a collection of techniques to create webpages that adapt to the browser's size. RWD is characterized by three things:

1. Elements are laid out on fluid, proportion-based grids that use relative units like percentages instead of absolute units like pixels.
2. Images are sized with relative units to adapt to various screen sizes.
3. CSS media queries apply different CSS styles depending on the browser's width.

RWD allows developers to create a single webpage that looks good on mobile, desktop, and tablet browsers. The figure below shows a webpage whose contents change depending on the browser's width. Ex: On the desktop version, the search box is on the right side of the window. On the tablet version, the search box occupies the entire width of the browser. And on the mobile version, the search box is replaced with a search button. This material discusses designing websites using RWD.

Figure 10.1.4: Webpage using responsive web design for desktop, tablet, and mobile.



Source: www.smashingmagazine.com

Responsive vs. adaptive

The term "adaptive design" is sometimes used by web designers to describe a website design philosophy that is similar to responsive web design. An **adaptive website** adapts to the width of the browser at specific widths. Ex: A container is 400 pixels wide when the browser is wider than 500 pixels, but the container shrinks to 200 pixels when the browser is less than 500 pixels wide. A responsive website will instead smoothly adjust the width of the container to fit the browser width.

The differences between adaptive and responsive design are not always easy to pin down, and even seasoned web designers [debate the differences](#) between the two design methodologies. This material uses the term "responsive" in contexts where some designers would argue the term "adaptive" should be used instead.

PARTICIPATION ACTIVITY

10.1.5: Responsive web design.

- 1) RWD requires the web server to examine the user agent string in HTTP requests.

☐ True

☐ False
- 2) RWD uses JavaScript to alter the page contents to fit the browser.

☐ True

☐ False
- 3) Decreasing a desktop browser's width is an easy way to determine if RWD techniques are being used.

☐ True

☐ False

Exploring further:

- [Google's Mobile SEO Overview](#)
- [Mobile Web from W3C](#)
- [The Difference Between Responsive and Adaptive Design](#)

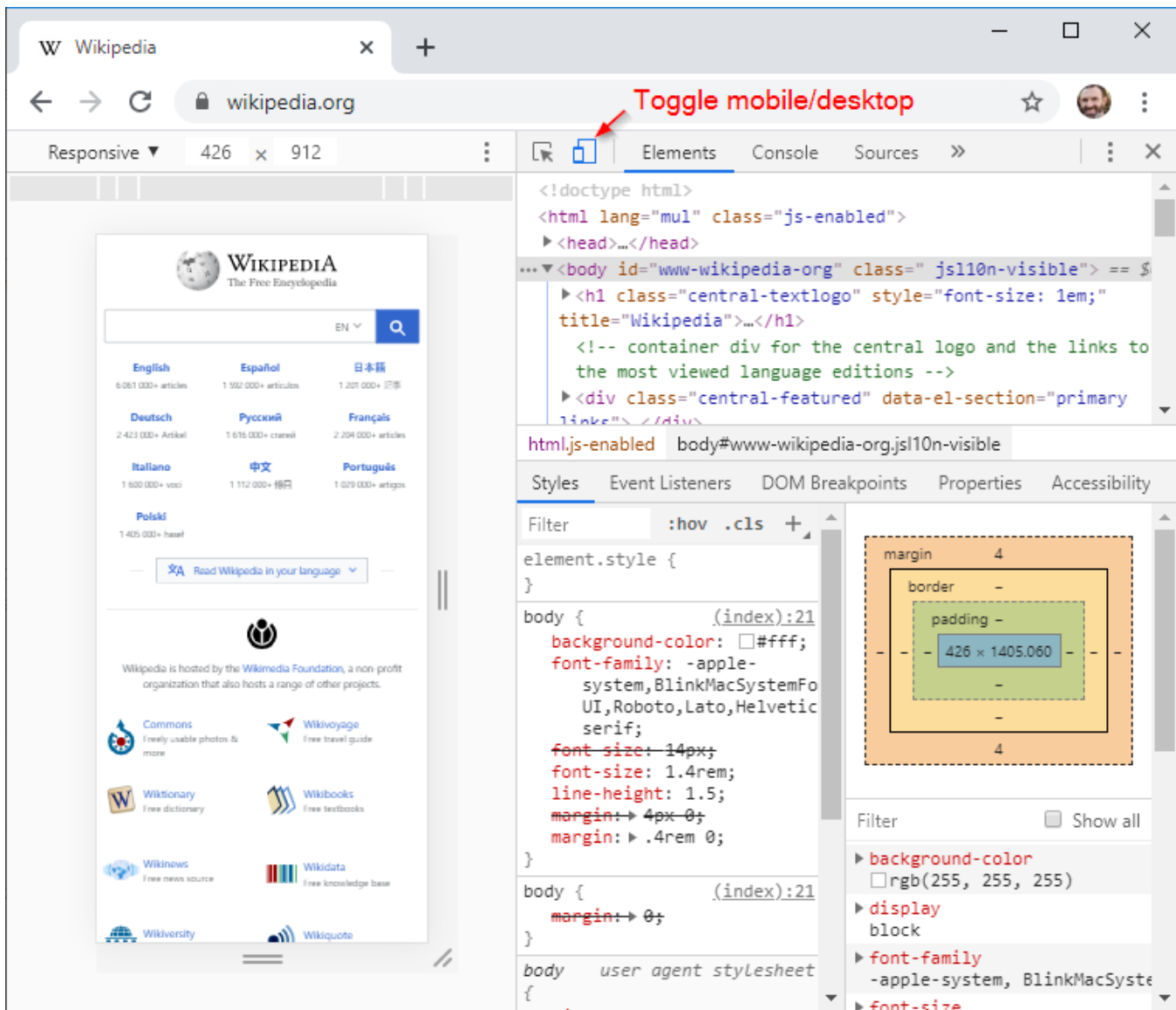
10.2 Mobile development tools

Screen emulator

Developers must test their mobile websites on a variety of mobile devices to ensure the websites work properly for all users. However, many desktop browsers contain development tools that aid mobile website development without using a mobile device.

The Chrome desktop browser's DevTools provides a screen emulator that can mimic a wide range of smartphones and tablets. A **screen emulator** is software that simulates how mobile device screens operate. Developers access the DevTools screen emulator by typing Ctrl-Shift-I in Chrome (Windows) or Command-Option-I (Mac).

Figure 10.2.1: Chrome DevTools screen emulator.

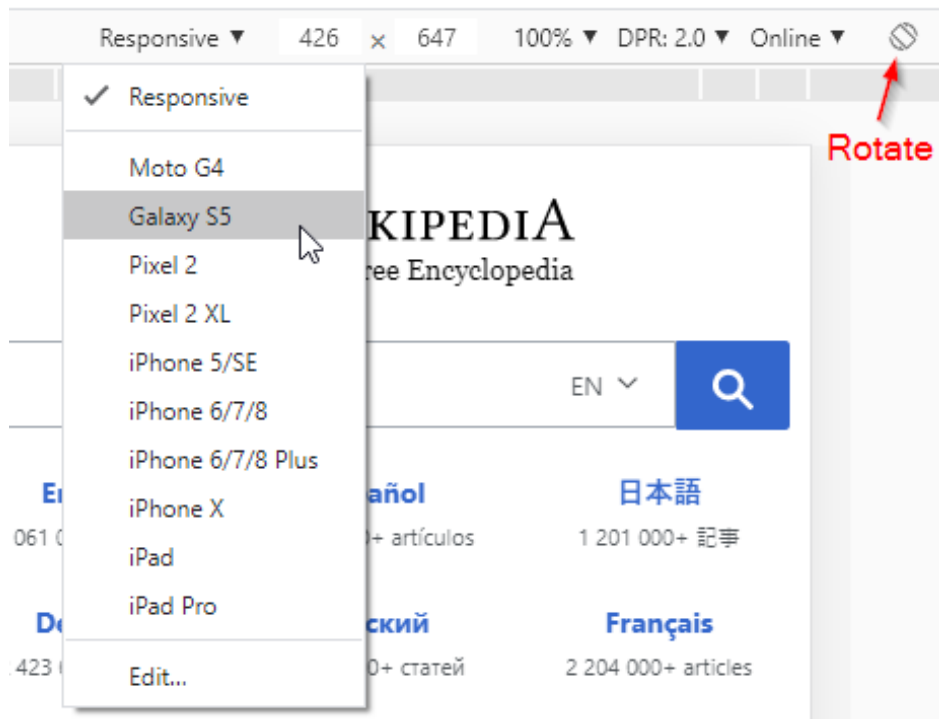


Source: en.wikipedia.org

The mouse pointer appears as a circle in the emulator to simulate a finger. Pressing the mouse button simulates touching the mobile device. Holding down the Shift key while dragging the mouse simulates a pinch gesture.

A variety of mobile devices can be selected from the dropdown list at the top of the DevTools. Each device has a set width and height, and pressing the rotate button swaps the width and height to emulate rotating the device. When a device is selected, the emulator is in Device mode. Selecting "Responsive" from the dropdown list puts the emulator in Responsive mode and allows the developer to change the width and height to any value.

Figure 10.2.2: Selecting a device to emulate.



**PARTICIPATION
ACTIVITY**

10.2.1: Chrome screen emulator.

- 1) A pinch gesture usually zooms in and out of the webpage.
 - ☐ True
 - ☐ False
- 2) The Chrome screen emulator can display a mobile website in portrait or landscape.
 - ☐ True
 - ☐ False

- 3) The emulator is only capable of emulating the iPhone.
- ☐ True
- ☐ False
- 4) The developer can change the emulated width and height when the emulator is in Device mode.
- ☐ True
- ☐ False
- 5) The www.whitehouse.gov home page looks identical in desktop Chrome and in an emulated iPhone 6 browser.
- ☐ True
- ☐ False

Device pixel ratio

©zyBooks 04/15/24 16:44 2071381
Marco Aguilar

Chrome's screen emulator allows developers to view or change the device pixel ratio. The **device pixel ratio (DPR)** is the ratio between device pixels and logical pixels. Ex: A DPR of 2 means 1 logical pixel is 2 device pixels wide and 2 device pixels tall, 4 device pixels altogether. A **logical pixel** is also called a **device-independent pixel (DIP)**. The CSS "px" unit is a logical pixel unit. Ex: `<div style="width:20px">` makes the div 20 logical pixels wide.

Manufacturers usually build screens with DPR dependent on the distance between the viewer and screen. DPR values of 2 and above are common on high-end mobile devices, where the viewer is close to the screen. Most desktop monitors have a DPR of 1 because the viewer is farther from the screen.

Figure 10.2.3: Four logical pixels are 4 device pixels on 1 DPR screen, 16 device pixels on 2 DPR screen, and 36 device pixels on 3 DPR screen.

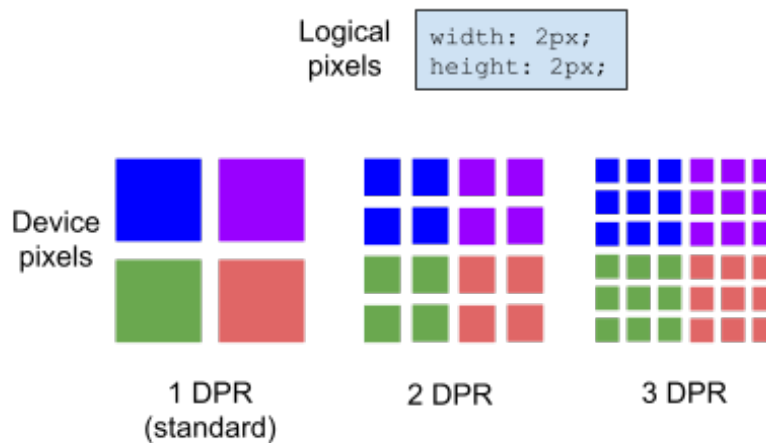
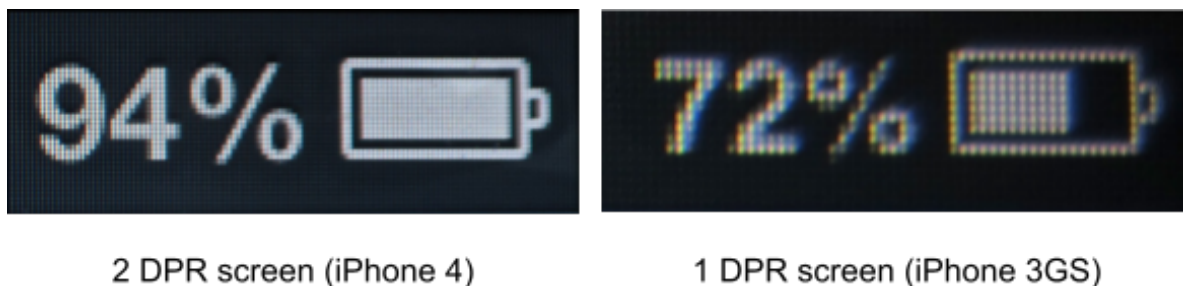


Figure 10.2.4: DPR settings in Chrome's screen emulator.



If DPR settings are not visible, click the vertical ellipsis on the far right side of the toolbar and choose "Add device pixel ratio" from the menu.

Figure 10.2.5: Image sharpness increases on higher DPR screen.



Source: [Retina display - Haotian0905](#) / Public Domain, [Non-Retina display - Haotian0905](#) / Public Domain



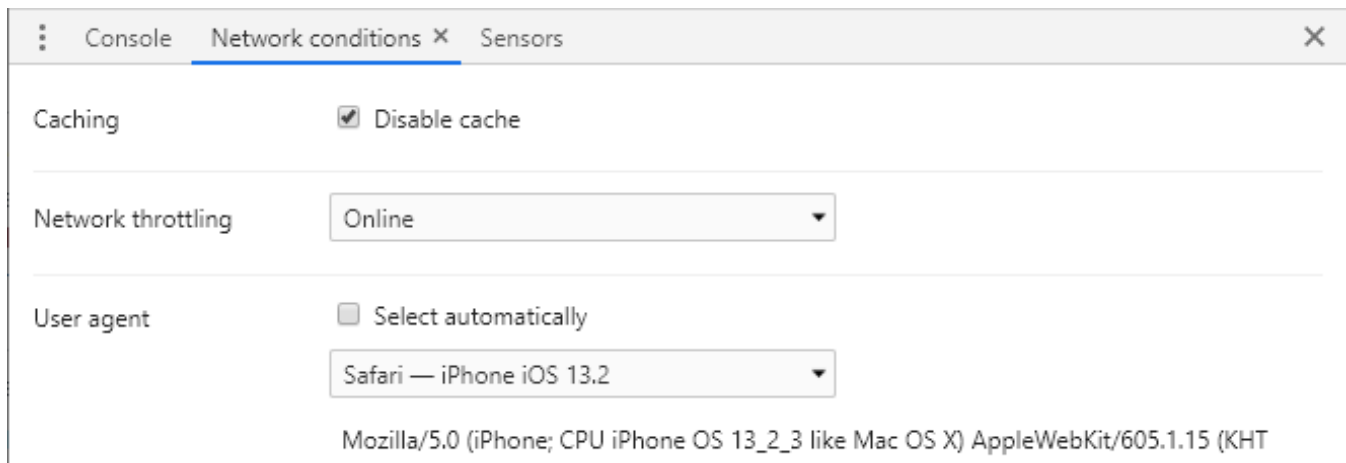
- 1) An image is 50×50 logical pixels. How large in device pixels is the image on a 1 DPR display?
- ☐ 50×50
- ☐ 25×25
- ☐ 100×100
- 2) An image that is 50×50 device pixels is created for a 1 DPR display. If the image is modified for a 2 DPR display, how large in device pixels is the new image on a 1 DPR display?
- ☐ 50×50
- ☐ 25×25
- ☐ 100×100
- 3) An image that is 50×50 logical pixels will use how many total device pixels on a 3 DPR display?
- ☐ 250
- ☐ 2,500
- ☐ 22,500

Emulating network conditions

Pressing Esc in DevTools shows "Network conditions" and "Sensors" tabs. If the tabs are not visible, clicking the `:` button to the left of "Console" reveals a menu of extra tools that includes "Network conditions" and "Sensors". The Network conditions tab allows the browser to modify the following:

- **Caching** - Disabling the cache forces the browser to download all webpage resources every time
- **Network throttling** - Simulate slower networks like 2G or 3G or simulate being offline
- **User agent** - Change the user agent string to any number of browser user agents

Figure 10.2.6: Emulator configuration options.



**PARTICIPATION
ACTIVITY**

10.2.3: Emulating network conditions.

- 1) If disk cache is disabled, the cache remains disabled when the DevTools are closed.
 - ☐ True
 - ☐ False
- 2) Changing the Network throttling dropdown to "Offline" allows the developer to test how the website behaves with no Internet connection.
 - ☐ True
 - ☐ False
- 3) Changing the device in the emulator changes the user agent in the "Network conditions" tab.
 - ☐ True
 - ☐ False

Emulating sensors

Most desktop computers do not have touch screens, GPS sensors, or accelerometers, so Chrome provides sensor emulators. The "Sensors" tab is shown in the figure below.

Figure 10.2.7: Sensors to emulate device location and orientation.

The screenshot shows the 'Sensors' tab in the Chrome DevTools interface. It is divided into two main sections: 'Location' and 'Orientation'. The 'Location' section includes a dropdown menu set to 'London' with a 'Manage' button next to it. Below this are four input fields: '51.507351' for Latitude, '-0.127758' for Longitude, 'Europe/London' for Timezone ID, and 'en-GB' for Locale. The 'Orientation' section features a dropdown menu set to 'Custom orientation'. Below it are three input fields: '332.5922' for α (alpha), '43.063' for β (beta), and '21.3383' for γ (gamma). A 'Reset' button is located at the bottom of the Orientation section. To the right of the Orientation inputs is a graphic of a smartphone tilted at an angle.

Location	
London	Manage
51.507351	Latitude
-0.127758	Longitude
Europe/London	Timezone ID
en-GB	Locale

Orientation	
Custom orientation	
332.5922	α (alpha)
43.063	β (beta)
21.3383	γ (gamma)
Reset	

PARTICIPATION ACTIVITY

10.2.4: Sensors emulator.

- 1) The geolocation (latitude and longitude) of the emulated device is modified by moving the desktop computer to a different location.

- ☐ True
- ☐ False

2) A developer can drag the image that looks like a mobile device to modify the accelerometer's alpha (rotation around the z-axis), beta (front-to-back tilt), and gamma (left-to-right tilt) values.

- ☐ True
- ☐ False

Exploring further:

- [Chrome DevTools](#)

10.3 Viewport

Viewport meta tag

A **viewport** is the visible area of a webpage. Before smartphones, many webpages were designed to fit a browser that was nearly 1000 pixels wide. Mobile browsers were not wide enough to show all the content without forcing the user to scroll horizontally. So, mobile browsers pretended the browser's viewport was 980 pixels wide, which enabled most websites to fit in the mobile browser. Using a viewport that was wider than the mobile device's screen required the user to zoom-in to comfortably read the text.

Figure 10.3.1: 320px wide mobile browser displaying 980px wide webpage.



The viewport meta tag allows developers to specify viewport properties. Some common viewport properties include:

- **width** - The viewport width, which is typically set to **device-width**, the device's screen width.
- **initial-scale** - The initial zoom level, which is typically set to 1 so the webpage is initially zoomed in at 100%.
- **user-scalable** - A boolean value that, when set to **no**, prevents the user from zooming in and out.

The viewport properties above are specified in the viewport meta tag's **content** attribute, separated by commas.

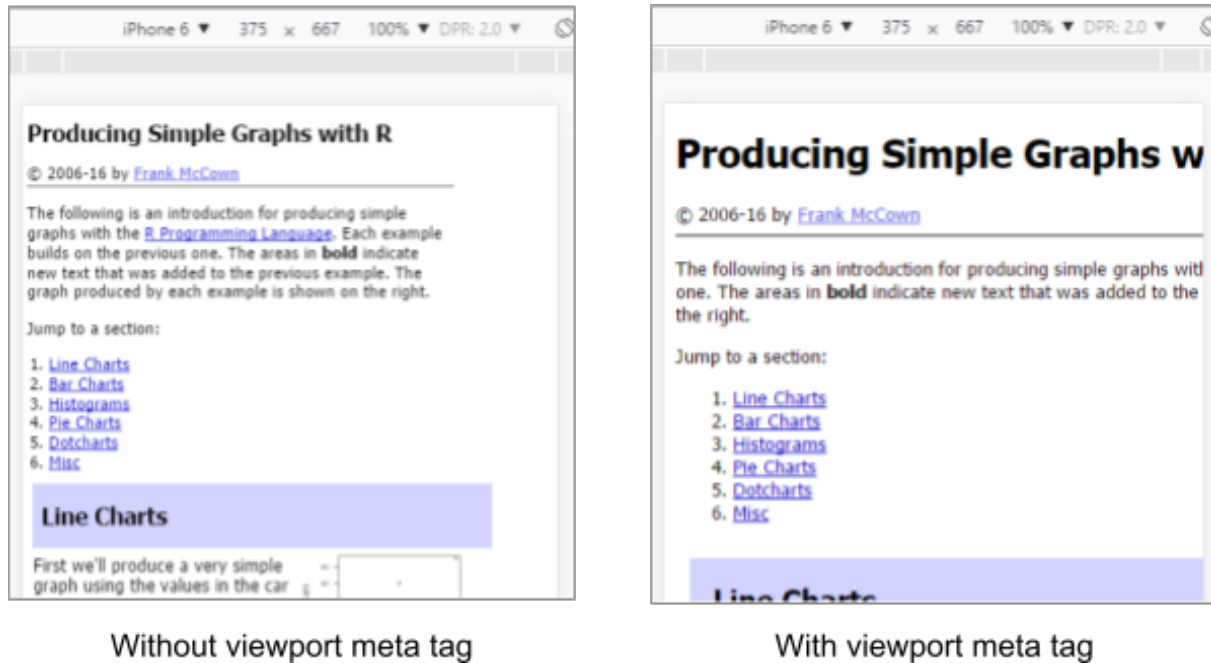
Figure 10.3.2: Viewport meta tag with standard parameters.

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

A webpage not optimized for mobile devices is shown in the figure below. When the viewport meta tag is not used, the default viewport of 980 pixels is used, and the page is shrunk to fit the mobile

viewport. When the viewport meta tag is used, the text is much larger and easier to read. However, the webpage must be scrolled horizontally to see the entire page, since the webpage hasn't been optimized for mobile.

Figure 10.3.3: Viewport meta tag used in webpage.



PARTICIPATION ACTIVITY

10.3.1: Viewport meta tag.

- 1) The `height` property is often used in the viewport meta tag.
 - ☐ True
 - ☐ False
- 2) Best practice is to set the `initial-scale` property to 1.
 - ☐ True
 - ☐ False

3) Setting the viewport meta tag property `user-scalable=no` disables zooming.

- ☐ True
- ☐ False

Viewport units

Developers frequently want the content of a webpage to occupy a fixed percentage of the viewport. Most desktop and mobile browsers support CSS viewport units. A **viewport unit** (**vw** and **vh**) is a percentage of the browser viewport's width or height, where 1vw = 1% of viewport width and 1vh = 1% of viewport height. CSS also defines minimum and maximum viewport units **vmin** and **vmax** where 1vmin = 1vw or 1vh, whichever is smaller and 1vmax = 1vw or 1vh, whichever is larger.

PARTICIPATION ACTIVITY

10.3.2: Viewport units.

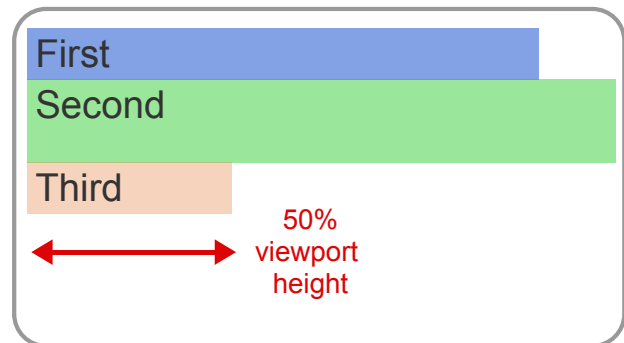
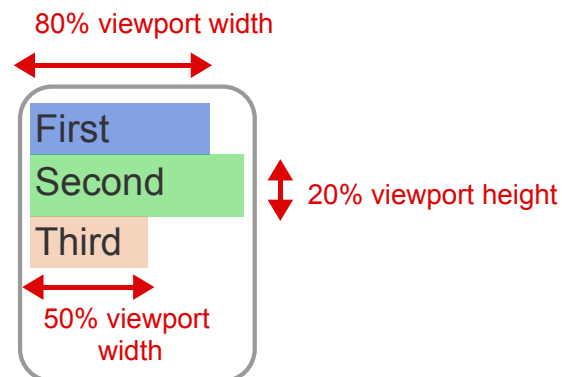
```
<!DOCTYPE html>
<html>
  <title>Viewport units</title>
  <style type="text/css">

#first {
  background: cornflowerblue;
  width: 80vw;
}

#second {
  background: lightgreen;
  height: 20vh;
}

#third {
  background: mistyrose;
  width: 50vmin;
}

</style>
<body>
  <div id="first">First</div>
  <div id="second">Second</div>
  <div id="third">Third</div>
</body>
</html>
```



Animation content:

The following code snippet is displayed:

```
<!DOCTYPE html>
<html>
  <title>Viewport units</title>
  <style type="text/css">
#first {
  background: cornflowerblue;
  width: 80vw;
}
#second {
  background: lightgreen;
  height: 20vh;
}
#third {
  background: mistyrose;
  width: 50vmin;
}
</style>
<body>
  <div id="first">First</div>
  <div id="second">Second</div>
  <div id="third">Third</div>
</body>
</html>
```

End of code snippet. A box representing the browser displays the three divs. The "First" div has a width that is 80% of the viewport width. The "Second" div has a height that is 20% of the viewport height. The "Third" div has a width that is 50% of the viewport width, which is not as wide as the "First" div. A larger browser also displays the three divs. Compared to the smaller browser, the "First" div is wider, the "Second" div is taller, and the "Third" div is wider.

Animation captions:

1. The First <div> has a width of 80vw, which is 80% of the viewport width.
2. The First <div> width grows wider as the browser's viewport is made wider.
3. The Second <div> has a height of 20vh, which is 20% of the viewport height.
4. Second <div> height increases as the viewport height increases.

5. The Third <div> has a width of 50vmin, which is 50% of the viewport width when the viewport width < height.
6. 50vmin is 50% of the viewport height when the viewport width > height.

**PARTICIPATION
ACTIVITY**

10.3.3: Viewport units.



- 1) If a browser viewport is 400px wide, how wide is an element with **width:50vw**?
 - ☐ 400px
 - ☐ 200px
 - ☐ 100px
- 2) If a browser viewport is 300px high, how wide is an element with **width:10vh**?
 - ☐ 300px
 - ☐ 30px
 - ☐ 150px
- 3) If a browser viewport is 400px wide and 300px high, how wide is an element with **width:50vmax**?
 - ☐ 400px
 - ☐ 100px
 - ☐ 200px



- 4) If a browser viewport is 400px wide, how wide are the two inner `<div>` elements?



```
<div width="300px">
  <div width="50vw">A</div>
  <div width="50%">B</div>
</div>
```

- ☐ A = 200px, B = 150px
- ☐ A = 150px, B = 150px
- ☐ A = 200px, B = 200px

PARTICIPATION ACTIVITY

10.3.4: Practice with viewport units.



The webpage below displays three quotes that are embedded in `<div>` tags using the CSS class "quote". The "quote" class uses a fixed width of 180px. Resize the webpage by grabbing the right edge of the rectangle surrounding the page with the mouse and moving left or right. Note that the three quotes do not change in size. Do the following:

1. Change the "quote" class's width to 20vw and re-render the page. Resize the webpage width and observe how the quotes' width changes with the webpage's width.
2. Add "height: 80vh;" to the "quote" class and re-render the page. Resize the webpage height and observe how the height of the colored part of the quotes changes with the webpage height. The text remains fixed.
3. Add "overflow: auto;" to the "quote" class and re-render the page. Resize the webpage height and observe how the quotes' height with the webpage height, and a vertical scrollbar is added to the quotes for viewing the hidden text.
4. Change the "quote" class's width to 120vw and re-render the page. The quotes are 20% wider than the webpage, so a horizontal scrollbar appears. Resize the webpage width and observe how the horizontal scrollbar is always necessary to view the hidden 20% of the webpage.

[HTML](#) [CSS](#)

```
1 <div class="quote quote1">
2 Love is friendship that has caught fire. It is quiet
3 understanding, mutual confidence, sharing and forgiving.
4 It is loyalty through good and bad times. It settles for
5 less than perfection and makes allowances for human weaknesses.
6 <br>- Ann Landers
7 </div>
8
9 <div class="quote quote2">
10 Love is the most important thing in the world, but baseball
11 is pretty good too.
12 <br>- Yogi Berra
13 </div>
14
15 <div class="quote quote3">
16 Love is patient. love is kind. It does not envy. it does not
```

[Render webpage](#)[Reset code](#)

Your webpage

Love is friendship that has caught fire. It is quiet understanding, mutual confidence, sharing and forgiving. It is loyalty through good and bad times. It settles for less than perfection and makes allowances for human weaknesses.
- Ann Landers

Love is the most important thing in the world, but baseball is pretty good too.
- Yogi Berra

Love is patient, love is kind. It does not envy, it does not boast, it is not proud. It does not dishonor others, it is not self-seeking, it is not easily angered, it keeps no record of wrongs. Love does not delight in evil but rejoices with the truth.
- Paul the Apostle

[► View solution](#)

**CHALLENGE
ACTIVITY**

10.3.1: Viewport.



550544.4142762.qx3zqy7

Start

Add a viewport <meta> tag with user-scalable set to no, width set to device-width, and initial-scale set to 1.

```
1  
2 <!-- Your solution goes here -->  
3
```

1

2

Check**Next**

Exploring further:

- [CSS length \(MDN\)](#)
- [Using the viewport meta tag to control layout on mobile browsers](#)

10.4 Media queries

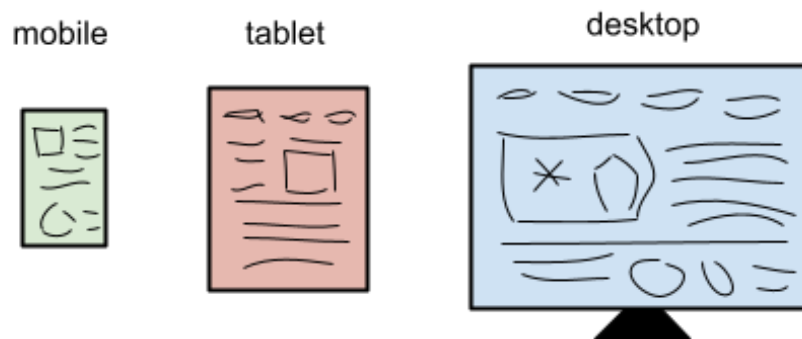
Designing for different screen sizes

Web developers typically design websites to make optimal use of the entire viewport. Viewports vary widely in size, so three platforms developers consider are: desktop, tablet, and mobile. Developers follow two general strategies to design websites for all three platforms:

1. **Graceful degradation**: Design the desktop website first and modify the design to fit smaller screens.
2. **Progressive enhancement**: A "mobile first" design methodology that begins with designing the website for the smallest device and then adapts the design for larger screens.

Good practice is to use progressive enhancement to build websites that show equivalent text and images and provide the same functionality on all platforms.

Figure 10.4.1: Designing for three platforms.



PARTICIPATION ACTIVITY

10.4.1: Targeting different screens.



1) A developer that modifies an existing desktop website for a mobile device is using a progressive enhancement design strategy.

- ☐ True
☐ False

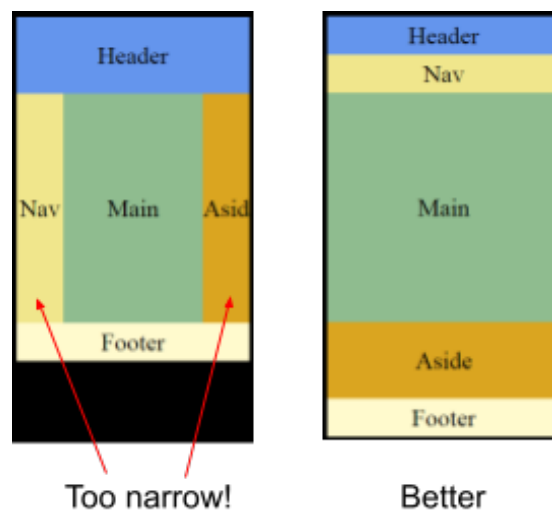
2) Progressive enhancement helps designers to simplify the amount of text displayed on a webpage and remove unnecessary clutter.

- ☐ True
☐ False

Media queries

A fluid layout is ideal for both desktop and mobile devices. However, a fluid layout that works well on a desktop does not always work well on a mobile device. Developers use responsive web design to modify a webpage's layout depending on the browser's width.

Figure 10.4.2: A fluid design for desktop should be altered for mobile.



Webpages using responsive web design adapt to different viewport sizes using media queries. A CSS **media query** is a combination of media type and optionally one or more media feature

expressions that evaluate to true or false.

- A **media type** is a device category that a media query applies to. Ex: **all**, **print**, and **screen**.
- A **media feature** is a characteristic of a browser, device, or environment. Ex: **aspect-ratio**, **height**, and **orientation**.

Some media features may be prefixed with "min-" or "max-" to express \geq or \leq constraints. Ex: **min-width:200px** means $\text{width} \geq 200\text{px}$.

Table 10.4.1: Media types used in media queries.

Media type	Description
all	All devices (default value if no media type is listed)
print	For printers and documents viewed on screen in print preview mode
screen	Intended for screens

Table 10.4.2: Example media features used in media queries.

Media features	Description
aspect-ratio	Viewport's width-to-height aspect ratio
height	Viewport's height
width	Viewport's width
orientation	Viewport's orientation: portrait or landscape
resolution	Device's pixel density

Media queries often are used in `<link>` elements and in stylesheets.

- A media query in a `<link>` is specified with the **media attribute**. When the media query matches (evaluates to true), the link's stylesheet is downloaded.
- A media query in a stylesheet is specified with the **@media** at-rule. An **at-rule** is a CSS statement that starts with the @ character and instructs the CSS engine how to behave. When a media query in a stylesheet matches, the @media's inner rules are defined.

PARTICIPATION ACTIVITY

10.4.2: Evaluating media queries.

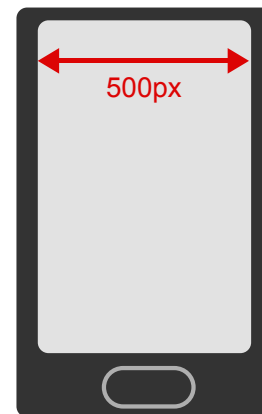


```

<!-- CSS media query on a link element -->
<link rel="stylesheet"
false media="print and (min-width: 400px)"
      href="styles.css">

<!-- CSS media query in a stylesheet -->
<style> true true
@media screen and (max-width: 800px) {
  .proposal {
    width: 350px; 500px <= 800px
  }
}
</style>

```



Animation content:

The following code snippet is displayed:

```
<!-- CSS media query on a link element -->
```

```
<link rel="stylesheet" media="print and (min-width: 400px)" href="styles.css" />
```

```
<!-- CSS media query in a stylesheet -->
```

```
<style>
```

```
@media screen and (max-width: 800px) {
```

```
  .proposal {
```

```
    width: 350px;
```

```
  }
```

```
}
```

```
</style>
```

End of code snippet. A mobile screen is 500px wide.

Animation captions:

1. A webpage with media queries is being viewed on a mobile device with a viewport that is 500px wide.
2. The link element defines a media query: Is the webpage being printed and the viewport's width ≥ 400 pixels?
3. The "print" media type is false because the webpage is displayed on a mobile device. The media query does not match, so styles.css is not downloaded.
4. The stylesheet uses a media query: Is the webpage displayed on a viewport that is ≤ 800 pixels wide?
5. The "screen" media type is true and $500\text{px} \leq 800\text{px}$, so the media query matches. The .proposal class is defined.

PARTICIPATION ACTIVITY

10.4.3: Media queries.

- 1) What media type is used if no media type is specified in the media query?

```
@media (max-width: 500px) {  
    ...  
}
```

- ☐ all
- ☐ print
- ☐ screen

- 2) Are media queries required to have at least one expression after the media type?

- ☐ Yes
- ☐ No

- 3) A webpage is using the stylesheet below. What color is the webpage's text when the webpage is printed?



```
body {  
    color: blue;  
}  
  
@media print {  
    body {  
        color: red;  
    }  
}
```

- ☐ black
- ☐ blue
- ☐ red
- 4) Is there a media type that specifically targets mobile devices?
- ☐ Yes
- ☐ No
- 5) Which expression is true for a viewport that is 1000px wide?
- ☐ (width: 500px)
- ☐ (max-width: 500px)
- ☐ (min-width: 500px)
- 6) Which media query matches a 320px wide viewport whose width is larger than the viewport's height?
- ☐ @media screen and (min-width: 320px) and (orientation: landscape) { ... }
- ☐ @media screen and (orientation: landscape) { ... }
- ☐ @media screen and (device-width: 320px) { ... }

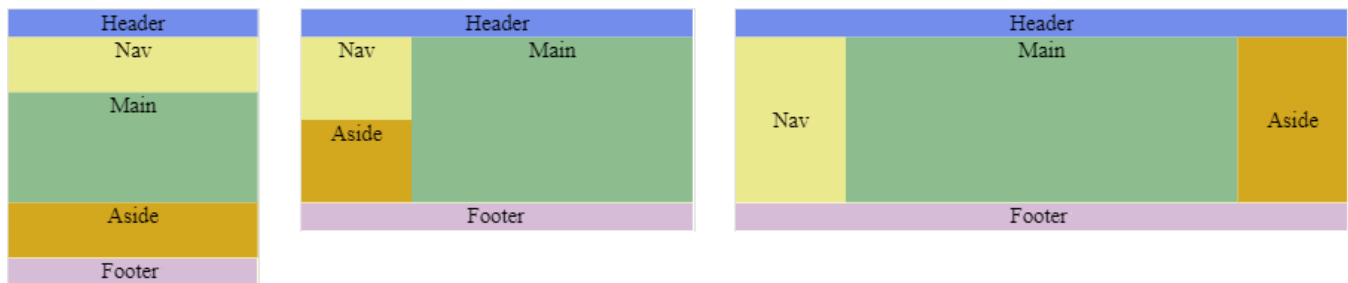


Breakpoints

Developers choose "breakpoints" to determine when content should be displayed, resized, or hidden in a webpage. A **breakpoint** is the screen width that activates a media query. Ex: In the media query `@media screen and (max-width: 800px)`, 800 is the breakpoint. *Good practice is to use breakpoints that target the content rather than a specific device. Ex: Using a breakpoint to hide a `<div>` when the device width is larger than 700px is better than a breakpoint to hide a `<div>` when the page is viewed on an iPhone X.*

Media query breakpoints do not work properly unless the viewport meta tag sets the viewport width to the device width. A common error is to forget to include the viewport meta tag `<meta name="viewport" content="width=device-width, initial-scale=1">` in the webpage when using breakpoints.

Figure 10.4.3: Mobile, tablet, and desktop wireframes for the Participation Activity below.



PARTICIPATION ACTIVITY

10.4.4: Create a responsive layout.

The webpage below displays a wireframe with a fluid layout ideal for a mobile device. A grid layout positions the page contents in a single column. Add the following CSS to the bottom of the existing CSS to create a responsive layout that is ideal for tablets and desktops:

1. Tablet: Add a media query at breakpoint 600px that creates 2 columns for the page contents and places the Nav and Aside on the same row as Main.

```
/* Tablet */
@media all and (min-width: 600px) {
  body {
    grid-template-columns: 200px auto;
    grid-template-rows: 50px 150px 150px 50px;
    grid-template-areas:
      "head head"
      "nav main"
      "aside main"
      "foot foot";
  }
}
```

2. Render the webpage. Resize the webpage width and observe how the layout changes when the width is 600px wide.
3. Desktop: Add a media query at breakpoint 800px that creates 3 columns and places Nav, Main, and Aside on the same row.

```
/* Desktop */
@media all and (min-width: 800px) {
  body {
    grid-template-columns: 200px auto 200px;
    grid-template-rows: 50px 300px 50px;
    grid-template-areas:
      "head head head"
      "nav main aside"
      "foot foot foot";
  }
}
```

4. Render the webpage. Resize the webpage width and observe how the layout changes when the webpage is 600px and 800px wide.

[HTML](#)[CSS](#)

```
1 <meta name="viewport" content="width=device-width, initial-scale=1">
2
3 <body>
4   <header>Header</header>
5   <nav>Nav</nav>
6   <main>Main</main>
7   <aside>Aside</aside>
8   <footer>Footer</footer>
9 </body>
10
```

[Render webpage](#)[Reset code](#)

Your webpage

Header

Nav

Main

**PARTICIPATION
ACTIVITY**

10.4.5: Breakpoints.



Refer to the CSS below.

```
body {  
  background-color: green;  
}  
  
@media screen and (min-width: 600px) {  
  body {  
    background-color: blue;  
  }  
}  
  
@media screen and (min-width: 800px) {  
  body {  
    background-color: red;  
  }  
}
```

- 1) Which of the media queries evaluate to true on a browser that is 900px wide?
- ☐ Neither of the media queries.
- ☐ The second media query only.
- ☐ Both media queries.
- 2) What is the body's background color for a browser that is 900px wide?
- ☐ green
- ☐ blue
- ☐ red
- 3) What is the body's background color on a mobile device in portrait mode with a viewport width of 375px?
- ☐ green
- ☐ blue
- ☐ red



4) What is the body's background color on a mobile device in landscape mode with a viewport width of 667px?

- ☐ green
- ☐ blue
- ☐ red

Exploring further:

- [How Does Progressive Enhancement Relate to Mobile-First Strategy?](#)
- [MDN: Using media queries](#)
- [Media Queries for Common Device Breakpoints](#)

10.5 Responsive images

Introduction

Most high-end mobile devices have high resolution screens with device pixel ratios (DPR) greater than 1. Ex: Pixel 2 has 2.6 DPR, iPhone X has 3 DPR, and Galaxy S10 has 4 DPR. DPR is often designated with an "x", so a standard screen is 1x, a 2 DPR screen is 2x, a 3 DPR screen is 3x, etc. A webpage's images need to have resolutions at least as high as the DPR of the mobile device in order to avoid looking pixelated or blocky.

Figure 10.5.1: Low DPR images appear pixelated on high DPR screens.



A **responsive image** is an image that scales to fit different layouts in a responsive website. Web developers must consider several issues when using responsive images:

- Responsive images need to scale to various sizes without looking pixelated or blocky.
- Higher resolution images should be used for devices with higher DPRs to avoid images looking pixelated or blocky.
- Images with the minimum DPR supported by a device should be used to avoid sending larger image files that waste network bandwidth.

**PARTICIPATION
ACTIVITY**

10.5.1: Responsive images.



1) A 1x image looks pixelated on a 2 DPR device.

- ☐ True
☐ False



2) A 2x image will look the same as an equivalent 1x image on a 1 DPR device.

- ☐ True
☐ False



3) A website should serve a 3x image to a 2 DPR device even if a 2x image is available.

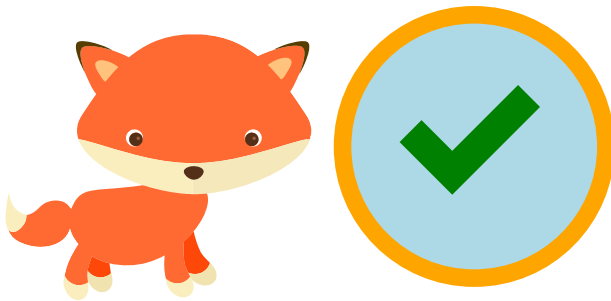
- ☐ True
- ☐ False

SVG images

SVG images are ideal for responsive images. A **Scalable Vector Graphics (SVG) image** is a vector image that is defined with XML. A **vector image** is an image defined with lines, curves, and points that scale nicely at any resolution and always appear crisp.

An SVG image may be stored a .svg file and displayed with an `` tag. Or an `<svg>` element may define the SVG image parts in the HTML document.

Figure 10.5.2: SVG image file and `<svg>` element.



```
<!-- Resize the browser width to see the images grow or shrink -->


<!-- Circle with checkmark inside -->
<svg viewBox="15 15 70 70" style="width:10vw">
  <circle cx="50" cy="50" r="30" stroke="orange" stroke-width="4"
fill="lightblue" />
  <path d="M60 36l-15 15-7-7-5 5 12 12 20-20z" fill="green"></path>
</svg>
```




```
<!-- Resize the browser width to see the images grow or shrink -->


<!-- Circle with checkmark inside -->
<svg viewBox="15 15 70 70" style="width:25vw">
  <circle cx="50" cy="50" r="30" stroke="orange" stroke-width="4"
fill="lightblue" />
  <path d="M60 36l-15 15-7-7-5 5 12 12 20-20z" fill="green"></path>
</svg>
```

Source: [Cute Fox](#) (KBJ-77) / Public Domain)

**PARTICIPATION
ACTIVITY**

10.5.2: SVG images.

1) Developers normally create 1x, 2x, and 3x SVG images to target various DPR devices.

- ☐ True
☐ False

2) An SVG image may be defined in a .svg file or in the HTML with the `<svg>` tag.

- ☐ True
☐ False




3) SVG is ideal for photographs.

- ☐ True
☐ False

srcset attribute

To display raster images like JPEG and PNG at the correct resolution, developers create different versions of the same image at different resolutions. Ex: A 3x device that needs to display a 100 x 75 pixel image needs an image with resolution 300 x 225. The `` **srcset attribute** specifies which image should be displayed for specific DPR values.

Figure 10.5.3: JPG images at different resolutions.

		
300 x 225 for 3x	200 x 150 for 2x	100 x 75 for 1x (standard)

**PARTICIPATION
ACTIVITY**

10.5.3: Browsers request images at correct resolution.

2x screen



dog-medium.jpg

```

```

Standard screen (1x)



dog-small.jpg

3x screen



dog-large.jpg



Web server



Animation content:

The following code snippet is displayed:

```

```

End of code snippet. A box labeled "web server" is displayed for the screen requests. A standard 1x screen is displayed with the "dog-small.jpg" image. A 2x screen is displayed with the "dog-medium.jpg" image. A 3x screen is displayed with the "dog-large.jpg" image.

Animation captions:

1. Three browsers have different DPRs. All three browsers read the tag.
2. Browser with 2 DPR screen requests dog-medium.jpg.
3. Browser with 3 DPR screen requests dog-large.jpg.
4. Browser with standard (1 DPR) screen requests dog-small.jpg.

PARTICIPATION ACTIVITY

10.5.4: Image srcset attribute.



Refer to the HTML below.

```

```

1) Which image is requested by a device with a 1.5 DPR?



- ☐ dog-small.jpg
- ☐ dog-medium.jpg
- ☐ dog-large.jpg

2) Which image is requested by an old browser that doesn't understand the **srcset** attribute?

- ☐ dog-small.jpg
- ☐ dog-medium.jpg
- ☐ All three images

3) A browser on a 3x screen downloaded dog-large.jpg, which was 300 x 225 pixels. What is the image size in logical pixels when displayed in the browser?

- ☐ 300 x 225
- ☐ 900 x 675
- ☐ 100 x 75

sizes attribute

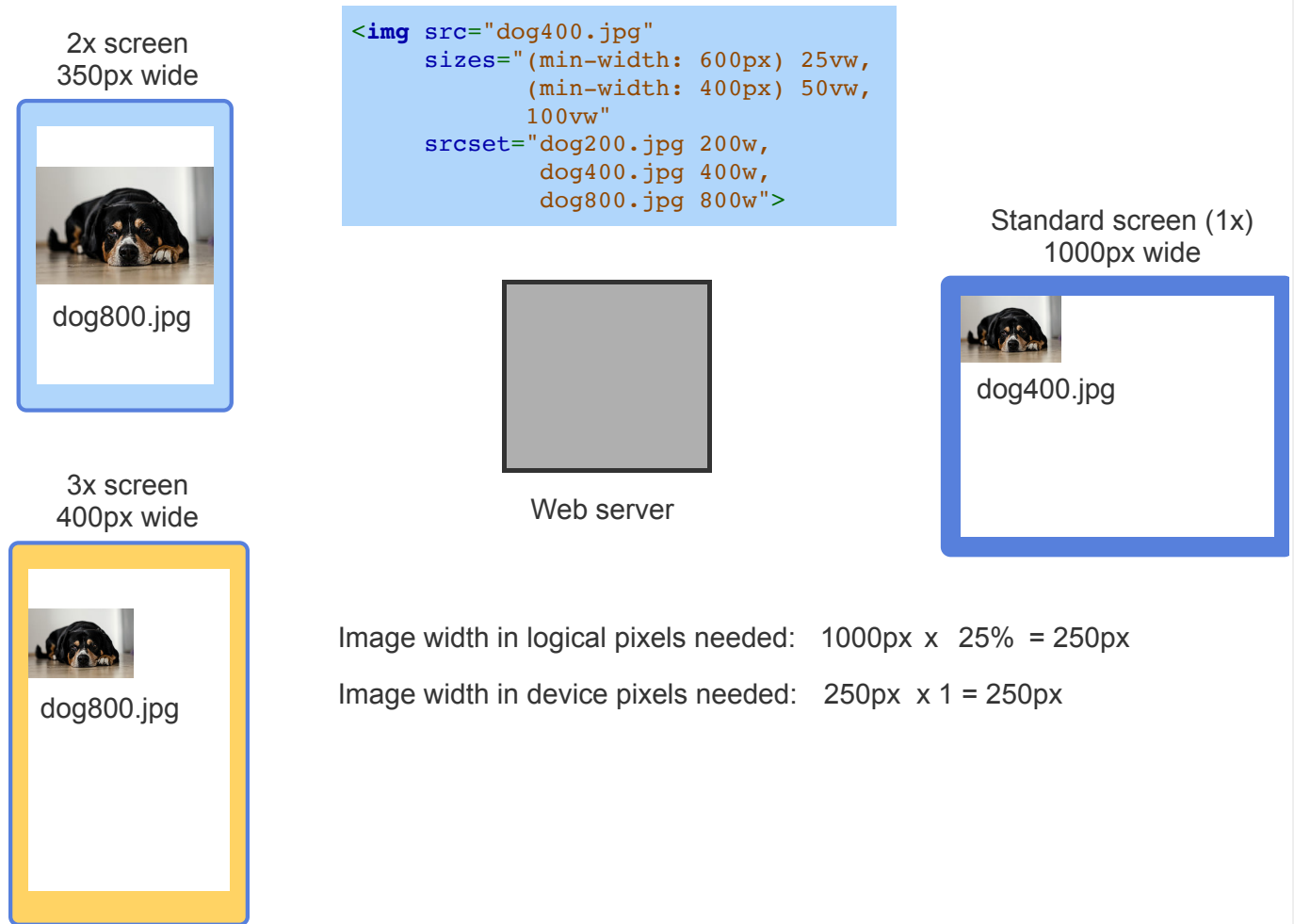
When a responsive image is displayed in a webpage that spans a proportional width of the browser, the browser needs to choose the image that best fits the proportional size at the given DPR. The browser uses an image's **sizes** and **srcset** attributes to calculate the needed image size for a given DPR and downloads the best image.

The image's **sizes attribute** specifies one or more pairs of media conditions and relative image sizes. Ex: **(min-width: 400px) 30vw** specifies a media condition (viewport width must be at least 400px wide) and a relative image size (30% of the viewport width). The media condition must be omitted for the last (default) image size.

The **srcset** attribute is used with the **sizes** attribute to indicate the actual image width for each image in the set. Ex: **fish.jpg 300w** indicates fish.jpg is 300 pixels wide. So if a browser uses **sizes** to calculate a 500px wide image is needed, the browser uses **srcset** to determine which image is at least 500px wide and downloads the appropriate image.

PARTICIPATION ACTIVITY

10.5.5: Browsers request images with optimal widths.



Animation content:

The following code snippet is displayed:

```

```

End of code snippet. A box labeled "web server" is displayed for the screen requests. A standard 1x screen that is 1000px wide is displayed with the "dog400.jpg" image. The image width in logical pixels needed is calculated by multiplying 350px by 100% equalling 350px. The image width in device pixels needed is calculated by multiplying 350px by 2 equalling 700px. A 2x

screen that is 350px wide is displayed with the "dog800.jpg" image. The image width in logical pixels needed is calculated by multiplying 400px by 50% equalling 200px. The image width in device pixels needed is calculated by multiplying 200px by 3 equalling 600px. A 3x screen that is 400px wide is displayed with the "dog800.jpg" image. The image width in logical pixels needed is calculated by multiplying 1000px by 25% equalling 250px. The image width in device pixels needed is calculated by multiplying 250px by 1 equalling 250px.

Animation captions:

1. Three web browsers have different DPRs and widths. All three browsers read the tag.
2. Browser with width = 350px needs an image that spans 100% of the browser width.
3. Browser calculates image width needed is 700px, so the best available image is dog800.jpg.
4. Browser with width = 400px needs an image that spans 50% of browser width.
5. Browser calculates image width needed is 600px, so the best available image is dog800.jpg.
6. Browser with width = 1000px needs an image that spans 25% of browser width.
7. Browser calculates image width needed is 250px, so the best available image is dog400.jpg.

PARTICIPATION ACTIVITY

10.5.6: Image sizes and srcset attributes.

Refer to the HTML below.

```

```

1) What image is requested by a 1x device with viewport width = 600px?

- ☐ dog200.jpg
- ☐ dog400.jpg
- ☐ dog800.jpg

2) What image is requested by a 2x device with viewport width = 400px?

- ☐ dog200.jpg
- ☐ dog400.jpg
- ☐ dog800.jpg

3) What image is requested by a 3x device with viewport width = 800px?

- ☐ dog800.jpg
- ☐ dog1600.jpg
- ☐ dog2000.jpg

4) What image is requested by an old browser that does not know what the `sizes` and `srcset` attributes are?

- ☐ dog400.jpg
- ☐ dog800.jpg
- ☐ dog2000.jpg

<picture> element

Another way browsers match images with device characteristics is with the `<picture>` element. The `<picture>` element is especially useful when implementing art direction. **Art direction** is the process of swapping out images with different proportions for different screen sizes. Ex: A wider image may be appropriate for a mobile device in landscape mode, but a skinnier image is more appropriate for portrait mode.

The `<picture>` element contains one or more `<source>` elements that specify a media condition and image. If the media condition is true, the associated image is downloaded and displayed. An `` element must also be specified as the image to download and display if none of the `<source>` images are selected.

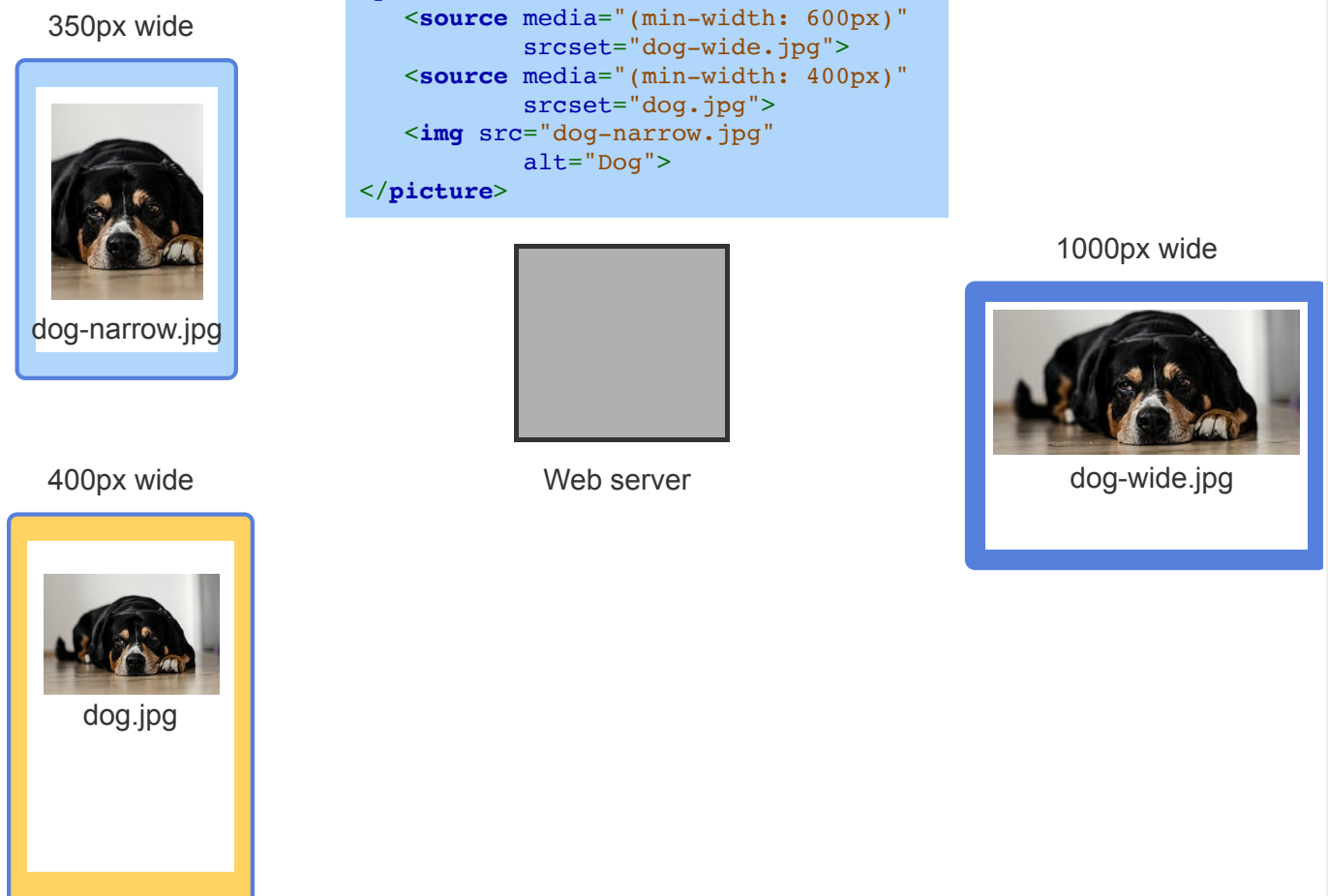
Most modern browsers support `<picture>`, and a polyfill called [Picturefill](#) is available for supporting older browsers. A **polyfill** is code that supports some functionality that is missing from a browser.

Figure 10.5.4: Art direction for various screen sizes.



**PARTICIPATION
ACTIVITY**

10.5.7: Browsers request images with <picture> element.



Animation content:

The following code snippet is displayed:

```
<picture>
  <source media="(min-width: 600px)"
    srcset="dog-wide.jpg">
  <source media="(min-width: 400px)"
    srcset="dog.jpg">
  
</picture>
```

End of code snippet. A box labeled "web server" is displayed for the screen requests. A 1000px wide screen is displayed with the "dog-wide.jpg" image. A 350px wide screen is displayed with the "dog-narrow.jpg" image. A 400px wide screen is displayed with the "dog.jpg" image.

Animation captions:

1. Three browsers have different widths. All three browsers read the <picture> element.
2. Browser with width = 350px does not match media query, requests dog-narrow.jpg.
3. Browser with width = 400px requests dog.jpg.
4. Browser with width = 1000px requests dog-wide.jpg.

PARTICIPATION ACTIVITY

10.5.8: Practice with the picture element.



The webpage below displays redrocks2v1.jpg when the webpage's width is at least 600px and redrocks1v1.jpg when the webpage's width is less than 600px. Resize the webpage width to verify the images swap when the browser width is 600px.

Modify <picture> to display redrocks3v1.jpg when the webpage's width is between 500px and 600px. redrocks3v1.jpg is accessible from the same directory as the other redrock images.

```
1 <picture>
2   <source media="(min-width: 600px)"
3     srcset="https://resources.zybooks.com/WebProgramming/redrocks2v1
4   
7 </picture>
8
```

[Render webpage](#)[Reset code](#)

Your webpage

[► View solution](#)

PARTICIPATION ACTIVITY

10.5.9: Picture element.



Refer to the HTML below.

```
<picture>
  <source media="(min-width: 600px)"
    srcset="dog-wide.jpg,
            dog-wide@1.5x.jpg 1.5x,
            dog-wide@2x.jpg 2x">
  <source media="(min-width: 400px)"
    srcset="dog.jpg,
            dog@1.5x.jpg 1.5x
            dog@2x.jpg 2x">
  
</picture>
```

- 1) What image is requested by a 1x device with viewport width = 650px?
 - ☐ dog-wide.jpg
 - ☐ dog-wide@1.5x.jpg
 - ☐ dog-narrow.jpg
- 2) What image is requested by a 2x device with viewport width = 400px?
 - ☐ dog.jpg
 - ☐ dog@2x.jpg
 - ☐ dog-narrow.jpg
- 3) What image is requested by a 3x device with viewport width = 350px?
 - ☐ dog@2x.jpg
 - ☐ dog-narrow.jpg
 - ☐ dog-narrow@2x.jpg

Exploring further:

- [Web Fundamentals: Images in Markup](#)
- [Responsive Images in Practice](#)

- [Built-in Browser Support for Responsive Images](#)
- [A Look Into: CSS4 Image Set](#)

10.6 Bootstrap

Introduction to Bootstrap

Bootstrap is one of the most popular frameworks for creating responsive websites because Bootstrap simplifies the task of creating fluid layouts and provides many common website components. Bootstrap includes:

- Numerous CSS classes that aid in styling text, tables, images, etc.
- A grid system for designing layouts that work on mobile, tablet, and desktop screens.
- Reusable components like dropdowns, alerts, navigation, modal dialogs, etc.

History of Bootstrap

Bootstrap was created by Mark Otto and Jacob Thornton at Twitter in mid-2010. Bootstrap was originally called Twitter Blueprint but was renamed Bootstrap and released as an open source project in late 2011.

A developer can download the Bootstrap CSS file from getbootstrap.com and host the file on a web server. Alternatively, Bootstrap can be downloaded from a CDN. A **Content Delivery Network (CDN)** hosts popular web files around the globe and automatically routes requests to the closest server, thus speeding up delivery of the files.

Figure 10.6.1: Downloading Bootstrap CSS from a CDN.

```
<link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
rel="stylesheet" crossorigin="anonymous"
integrity="sha384-
1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3">
```

**PARTICIPATION
ACTIVITY**

10.6.1: Practice with Bootstrap.



The webpage below uses the Bootstrap CSS library. Bootstrap provides consistent styling for all browsers using CSS selector rules called **Reboot**. Ex: The `<button>` element is given the same padding on Chrome, Firefox, and all other browsers. Bootstrap adds additional classes to change the appearance of an element.

Add some Bootstrap classes to modify the webpage. After each modification, render the webpage to see the change.

1. Add the class `display-1` to the `<h1>` element to create a larger display heading. Ex: `<h1 class="display-1">`. Try other display sizes: `display-2`, `display-3`, and `display-4`.
2. Add the class `text-primary` to the `<p>` element to make the heading blue. Try other text colors: `text-success`, `text-warning`, and `text-danger`.
3. Add the class `bg-success` to the `<p>` element to add a green background to the paragraph. Try other background colors: `bg-primary`, `bg-danger`, and `bg-dark`.
4. Add the following classes to the `<blockquote>` element:
 1. `border` to create a border
 2. `w-50` to make the width 50%
 3. `ms-3` to add a left margin
 4. `shadow` to cast a shadow
 5. `text-center` to center the text
5. Add the class `fw-bold` to the blockquote's `<p>` to make the font bold.
6. Add the `blockquote-footer` class to the blockquote's `<footer>` to put a dash before the name and decrease the font size.

7. Add the `table` class to the `<table>` to make the table span the browser width and to add some padding around the table contents.
8. Add the `table-striped` class to the `<table>` to make the table rows zebra-striped.

```
1 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css" integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oE" rel="stylesheet">
2
3
4 <h1>My First Bootstrap Page</h1>
5
6 <p>Experiment with different color classes.</p>
7
8 <blockquote>
9   <p>"Learning never exhausts the mind."</p>
10  <footer>Leonardo da Vinci</footer>
11 </blockquote>
12
13 <table>
14   <thead>
15     <tr>
16       <th>Title</th>
```

[Render webpage](#)[Reset code](#)

Your webpage

My First Bootstrap Page

Experiment with different color classes.

"Learning never exhausts the mind."

Leonardo da Vinci

Title	Author	Published
<i>Don Quixote</i>	Miguel de Cervantes	1605/1615
<i>Wuthering Heights</i>	Emily Brontë	1847
<i>Ulysses</i>	James Joyce	1922
<i>The Great Gatsby</i>	F. Scott Fitzgerald	1925

**PARTICIPATION
ACTIVITY**

10.6.2: Bootstrap library.



- 1) According to the figure above, what version of Bootstrap is being downloaded from a CDN?
 - ☐ cdn.jsdelivr.net
 - ☐ 5.1.3
 - ☐ bootstrap.min.css
- 2) What default body font does Reboot use?
 - ☐ Times New Roman
 - ☐ Courier New
 - ☐ A sans-serif font
- 3) What CSS Bootstrap class makes the text red?
 - ☐ text-primary
 - ☐ text-danger
 - ☐ text-success
- 4) What CSS Bootstrap class only affects an element's margin?
 - ☐ ms-3
 - ☐ width
 - ☐ table
- 5) What CSS Bootstrap class only affects an element's width?
 - ☐ ms-3
 - ☐ w-50
 - ☐ table



Grid system

Bootstrap uses a responsive **grid system** that allows developers to create layouts for a variety of screen sizes. The grid system is divided into 12 columns or less that fluidly resize as the browser size changes. The grid system is applied to a block element using one of two classes:

- **container** - Creates a responsive fixed-width container with a **max-width** that changes at various breakpoints.
- **container-fluid** - Creates a responsive container with a 100% width.

Grid columns are placed inside of block elements that use the **row** class. Columns use block elements with one of the class names from the table below. When the browser width is less than the breakpoint, the column is placed in a separate row. Ex: **col-md** runs horizontally when the browser's width is $\geq 768\text{px}$, but the column occupies an entire row when the browser's width is $< 768\text{px}$.

Table 10.6.1: Bootstrap breakpoints.

Description	Example	Breakpoint	Class
Extra small devices	Portrait phones	$< 576\text{px}$	col
Small devices	Landscape phones	$\geq 576\text{px}$	col-sm
Medium devices	Tablets	$\geq 768\text{px}$	col-md
Large devices	Desktops	$\geq 992\text{px}$	col-lg
Extra large devices	Large desktops	$\geq 1200\text{px}$	col-xl

The grid system requires the viewport meta tag be set appropriately.

Figure 10.6.2: Viewport meta tag required by Bootstrap grid system.

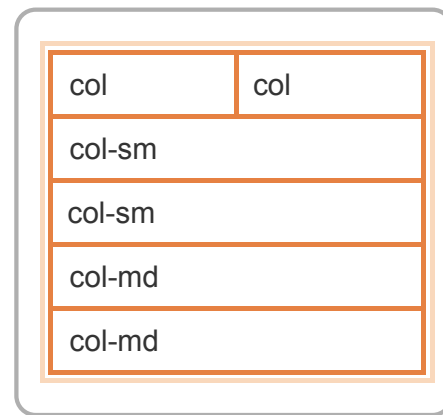
```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

PARTICIPATION ACTIVITY

10.6.3: Bootstrap grid system.



```
<div class="container-fluid">
  <div class="row">
    <div class="col">col</div>
    <div class="col">col</div>
  </div>
  <div class="row">
    <div class="col-sm">col-sm</div>
    <div class="col-sm">col-sm</div>
  </div>
  <div class="row">
    <div class="col-md">col-md</div>
    <div class="col-md">col-md</div>
  </div>
</div>
```



575 width

Animation content:

The following code snippet is displayed:

```
<div class="container-fluid">
  <div class="row">
    <div class="col">col</div>
    <div class="col">col</div>
  </div>
  <div class="row">
    <div class="col-sm">col-sm</div>
    <div class="col-sm">col-sm</div>
  </div>
  <div class="row">
    <div class="col-md">col-md</div>
    <div class="col-md">col-md</div>
  </div>
```

```
</div>  
</div>
```

End of code snippet. An 800px wide browser displays a grid with 2 columns and 3 rows. The first row contains 2 "col" labels in the columns. The second row contains 2 "col-sm" labels in the columns. The third row contains 2 "col-md" labels in the columns. After the browser is resized to 767px, the first and second rows are unchanged, and the third row has expanded into 2 rows (row 3 and 4), where each row has only 1 column labeled "col-md". After the browser is resized to 575px, the first row is unchanged, and the second row has expanded into 2 rows, each with 1 column labeled "col-sm". The fourth and fifth rows contain 1 column with the label "col-md".

Animation captions:

1. The `<div>` element uses the `container-fluid` class to become a responsive grid container that spans the browser width.
2. Three `<div>` elements use the `row` class to create a grid with 3 rows.
3. The `<div>` elements in each row use `col`, `col-sm`, and `col-md` classes to form equal-width columns.
4. When the browser is resized and is less than the 768 breakpoint, the `col-md` columns are placed on separate rows.
5. When the browser is resized and is less than the 576 breakpoint, the `col-sm` columns are placed on separate rows.

PARTICIPATION ACTIVITY

10.6.4: Practice with Bootstrap's grid system.



The webpage below uses the Bootstrap grid system. To create columns of different sizes, column class names end with a number to indicate the column width. The maximum width is 12. Ex: Class `col-md-4` means the column spans 4/12 or one third of the row's width, and `col-sm-6` spans 6/12 or half of the row's width.

Resize the webpage so the width is less than 576px, and the columns will form a single stacked column. Try other grid system layouts by adding the following HTML:

1. Add a new row with a single column that uses the `col-sm-9` class. Render the page and verify that the column only occupies 9/12 of the row.
2. Add a new row with three new columns that each use one of the following classes: `col-sm-10`, `col-sm-4`, `col-sm-4`. Render the page and verify that the

`col-sm-10` column fits on a row by itself because $10 + 4 = 14$ which is > 12 , so the `col-sm-4` column is forced onto the next row.

3. Add a new row with two new columns. The first column should use both classes: `col-sm-8` and `col-md-6`. The second column should use both classes: `col-sm-4` and `col-md-6`. Render the page and resize the webpage's width. Verify that when the rendered page is ≥ 768 px wide that both columns span half the row instead of the entire row because of the `col-md` classes. When the rendered page is ≥ 576 px and < 768 px, the columns span $8/12$ and $4/12$ of the row because of the `col-sm` classes.
4. Change the grid container class from `container-fluid` to `container`. Render the page and resize the webpage's width. Verify that the grid width is fixed at 768px when the browser width is ≥ 768 px and 576px when the browser width is ≥ 576 px and < 768 px.

HTML

CSS

```
1 <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5
2 integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oE
3 <meta name="viewport" content="width=device-width, initial-scale=1">
4
5 <div class="container-fluid">
6   <div class="row">
7     <div class="col-sm-1">col-sm-1</div>
8     <div class="col-sm-1">col-sm-1</div>
9     <div class="col-sm-1">col-sm-1</div>
10    <div class="col-sm-1">col-sm-1</div>
11    <div class="col-sm-1">col-sm-1</div>
12    <div class="col-sm-1">col-sm-1</div>
13    <div class="col-sm-1">col-sm-1</div>
14    <div class="col-sm-1">col-sm-1</div>
15    <div class="col-sm-1">col-sm-1</div>
16    <div class="col-sm-1">col-sm-1</div>
```

Render webpage

Reset code

Your webpage

col-sm-1	col-sm-1	col-sm-1	col-sm-1	col-sm-1	col-sm-1	col-sm-1	col-sm-1	col-sm-1	col-sm-1	col-sm-1	col-sm-1
col-sm-8								col-sm-4			
col-sm-2		col-sm-2		col-sm-4				col-sm-4			
col-sm-6						col-sm-6					

[► View solution](#)PARTICIPATION
ACTIVITY

10.6.5: Bootstrap grid system.

1) Should column classes always be nested inside of a `row` class?

- ☐ Yes
- ☐ No

2) How many columns does A span?

```
<div class="row">
  <div class="col-sm">A</div>
  <div class="col-sm">B</div>
</div>
```

- ☐ 4
- ☐ 6
- ☐ 8

3) If a column uses the class `col-lg-6`, how much of the row does the column occupy when the browser is 1000px wide?

- ☐ 50%
- ☐ 60%
- ☐ 100%

4) If a column uses the class `col-lg-6`, how much of the row does the column occupy when the browser is 900px wide?

- ☐ 25%
- ☐ 50%
- ☐ 100%

5) Are columns A and B ever on the same row?

```
<div class="row">
  <div class="col-sm-9">A</div>
  <div class="col-sm-4">B</div>
</div>
```

- ☐ Yes
- ☐ No

- 6) How wide must the browser be to display A and B each occupying 50% of the row?



```
<div class="row">
  <div class="col-md-9 col-sm-6">A</div>
  <div class="col-md-3 col-sm-6">B</div>
</div>
```

- ☐ < 576px
- ☐ ≥ 576px and < 768px
- ☐ ≥ 768px
- 7) What is the grid's width if the browser is 800px wide?



```
<div class="container">
  <div class="row">
    <div class="col-sm-10">A</div>
    <div class="col-sm-2">B</div>
  </div>
</div>
```

- ☐ 576px
- ☐ 768px
- ☐ 800px

Images

Bootstrap provides three CSS classes to place images into shapes: **rounded**, **rounded-circle**, and **img-thumbnail**.

Figure 10.6.3: Bootstrap image shapes.

```
  
  

```



Bootstrap provides the CSS class `img-fluid` that makes an image scale fluidly inside a parent element. The `img-fluid` class adds the following CSS styles to the image:
`display:block; max-width:100%; height:auto.`

**PARTICIPATION
ACTIVITY**

10.6.6: Fluid image.



The rendered webpage shows a cat image.

1. Resize the rendered webpage and verify the cat image does not change sizes when the webpage width is less than the cat image width.
2. Add `class="img-fluid"` to the `` element, and render the webpage.
3. Resize the rendered webpage and verify the cat image changes size when the webpage width is less than the cat image width.


```
1 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/boot
2   rel="stylesheet" crossorigin="anonymous"
3   integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1
4 <meta name="viewport" content="width=device-width, initial-scale=1">
5
6 
  
</div>
```

- ☐ True
- ☐ False

3) Assuming no other CSS properties apply to the images below, the images appear directly next to each other.

```


```

- ☐ True
- ☐ False

Components

Bootstrap provides many **components** for developing interactive user interfaces. Ex:

- Button to display a custom button
- Carousel to cycle through images or text in a slideshow
- Dropdown to display a dropdown menu
- Progress to display a progress bar

Some components require the Popper.js and Bootstrap JavaScript libraries to operate, as shown in the figure below. Familiarity with JavaScript programming is helpful to use some components effectively.

Figure 10.6.4: Accessing Popper and Bootstrap JavaScript libraries from a CDN.

```
<!-- Popper.js -->
<script
src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js"
  integrity="sha384-
7+zcNj/IqJ95wo16oMtfSKbZ9ccEh31eOz1HGyDuCQ6wgnyJNSYdrPa03rtR1zdB"
crossorigin="anonymous">
</script>

<!-- Bootstrap JavaScript -->
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js"
  integrity="sha384-
QJHtvGhmr9XOIpI6YVutG+2QOK9T+ZnN4kzFN1RtK3zEFEIsxhlmWl5/YESvpZ13"
crossorigin="anonymous">
</script>
```

PARTICIPATION ACTIVITY

10.6.8: Practice with Modal and Alert.



Two popular Bootstrap components are Modal and Alert:

- A **Modal** is a Bootstrap component that displays a configurable modal dialog box.
- An **Alert** is a Bootstrap component that shows a short message.

The webpage below displays an Email button, but pressing the button does nothing. Make the following modifications to display a modal dialog box and show alerts based on what buttons were clicked in the dialog box:

1. Add the following HTML attributes to the Email `<button>` element:
`data-bs-toggle="modal" data-bs-target="#emailModal"`. Render the page and verify that clicking on the Email button displays a modal dialog box. Clicking the Yes button in the dialog box closes the dialog box and displays an Alert confirming the email was sent. (No email is actually being sent.)
2. Add a new Alert for the No button:
 1. Add another Alert in the HTML like the alert with ID `yesAlert`, but make the new Alert use ID `noAlert`.
 2. Change the wording in the `noAlert` to indicate that the email was not sent.

3. Change the Alert's class from `alert-success` to `alert-danger`.

3. Finally, add some JavaScript that will show the Alert with ID `noAlert` when the No button is clicked:

```
const modalNoBtn = document.querySelector("#emailModal .btn-default");
modalNoBtn.addEventListener("click", function () {
  showAlert("noAlert");
});
```

Render the page and verify that pressing No in the Modal displays the new Alert with red text.

[HTML](#)[JavaScript](#)

```
1 <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
2   rel="stylesheet" crossorigin="anonymous"
3   integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1o9/6QbrFC3Qec4gtL9Qd3"
4   crossorigin="anonymous">
5 <!-- JavaScript libraries required for components -->
6 <script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.10.2/dist/umd/popper.min.js"
7   integrity="sha384-7+zCNj/IqJ95wo16oMtfsKbZ9ccEh31e0z1HGyDuCQ6wgnyJyrB516f8Dh8f0f3"
8   crossorigin="anonymous">
9 </script>
10 <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
11   integrity="sha384-QJHtvGhmr9X0IpI6YVutG+2Q0K9T+ZnN4kzFN1RtK3zEFEIsxhlmWl9/7vpDtppL2PvkY9J"
12   crossorigin="anonymous">
13 </script>
14
15 <meta name="viewport" content="width=device-width, initial-scale=1">
16
```

[Render webpage](#)[Reset code](#)

Your webpage

Email

► View solution

PARTICIPATION ACTIVITY

10.6.9: Bootstrap Modals and Alerts.



Refer to the Participation Activity above.

1) What `data-bs-target` attribute value makes a button display a Modal?



- ☐ `modal`
- ☐ modal's ID
- ☐ modal's class name

2) What CSS class initially hides an Alert?



- ☐ `alert-success`
- ☐ `alert`
- ☐ `collapse`

3) What happens when the Modal is displayed and the user clicks the dark part of the screen outside of the Modal or presses the ESC key?

- ☐ The Modal hides, and the green Alert displays.
- ☐ The Modal remains visible.
- ☐ The Modal hides, and no Alerts are displayed.

Exploring further:

- [Introduction to Bootstrap](#)
- [Reboot](#)

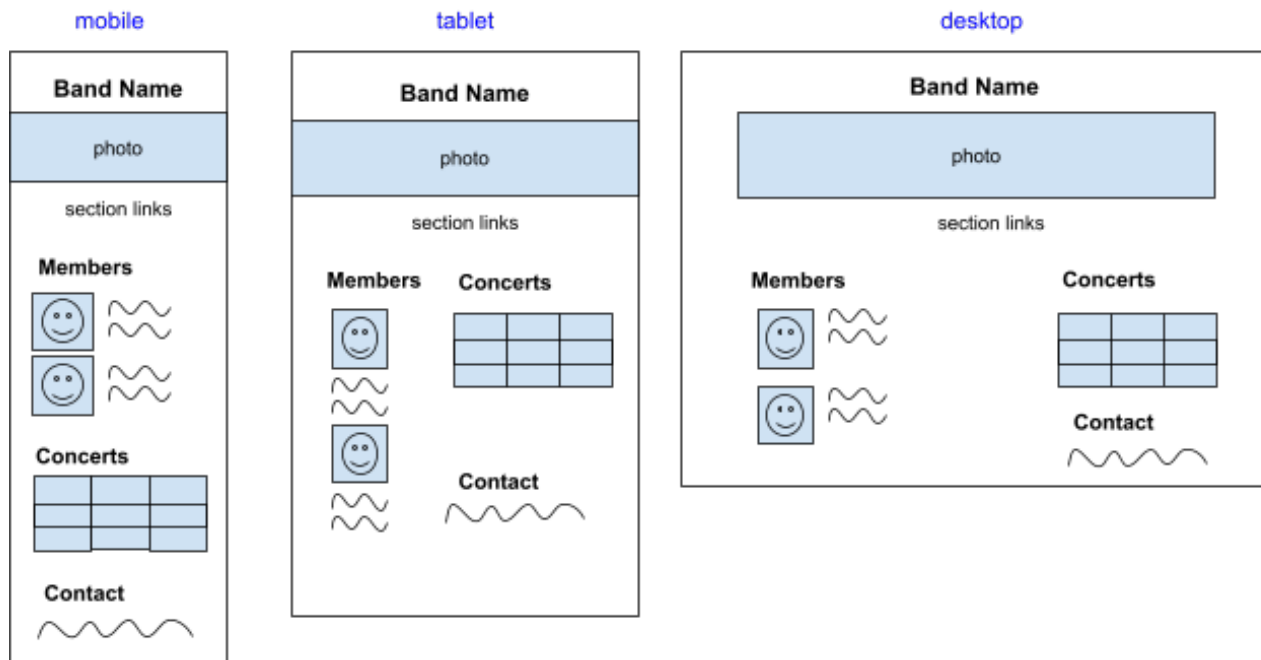
10.7 Example: Responsive band webpage

Designing for mobile first

This section modifies the band webpage from an earlier example that displays information about the fictional band *Reach Out*. Responsive web design is applied to create optimal designs for mobile, tablets, and desktops.

A progressive enhancement approach is taken: construct the mobile design first, then modify the design for larger screens. The webpage will use a grid layout to position the header, members, concerts, and contact information. The mobile design places the entire grid into a single column, but the tablet and desktop design use 2 columns for the sections.

Figure 10.7.1: Mobile, tablet, and desktop wireframes.



Mobile design.

band.html is the band's webpage, and styles.css is an external stylesheet that applies CSS to the webpage. The CSS creates a grid container for the `<body>` and places the header and 3 sections into grid items that are arranged in a single column. The design is optimal for mobile but not for larger screens.

The band photo at the top of the page uses the `width`, `max-width`, and `height` properties to make the image responsive. Resize the rendered webpage to see the photo resizing automatically to fill the browser width.

Images from [Wikimedia.org](https://commons.wikimedia.org/)

[index.html](#) [styles.css](#)

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>Reach Out</title>
7   <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10  <header>
11    <h1>Reach Out</h1>
12    
14    <div class="nav">
15      <a href="#members">Members</a> &nbsp;
16      <a href="#concerts">Concerts</a> &nbsp;
```

[Render webpage](#)[Reset code](#)

Your webpage

REACH OUT



PARTICIPATION ACTIVITY

10.7.1: Mobile design.



1) The `<body>` grid container has 4 grid items.

- ☐ True
☐ False

2) The `body` selector's `grid-gap` property may be modified to put more space between the header and each section.

- ☐ True
☐ False

3) Removing `display:grid;` from the `body` selector changes the rendered webpage significantly.

- ☐ True
☐ False

4) Flexbox is not used in the CSS.

- ☐ True
☐ False

Tablet design

The tablet's wider screen allows the Concerts and Contact section to appear next to the Members section. However, not enough room exists for the unordered lists, so each list must be moved under each photo.

The media query in the figure below does two things if the browser's minimum width is 600 pixels:

1. The grid container `<body>` is changed to 2 columns so the Concerts and Contact section appear next to the Members section.
2. The flexbox container with the class `band-member` is switched to the column direction so the member's unordered list appears under the member's photo.

Figure 10.7.2: Media query for tablets.

```
/* tablets */
@media all and (min-width: 600px) {
  body {
    grid-template-columns: auto
    auto;
    grid-template-areas:
      "head head"
      "members concerts"
      "members contact";
  }
  .band-member {
    flex-direction: column;
  }
}
```

Tablet design.

Resize the rendered webpage to see the changes take effect when the browser window is just below or just above 600 pixels wide.

Images from [Wikimedia.org](https://www.wikimedia.org/)

index.html

styles.css

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>Reach Out</title>
7   <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10  <header>
11    <h1>Reach Out</h1>
12    
14    <div class="nav">
15      <a href="#members">Members</a> &nbsp;
16      <a href="#concerts">Concerts</a> &nbsp;
```

Render webpage

Reset code

Your webpage

REACH OUT

PARTICIPATION
ACTIVITY

10.7.2: Tablet design.



1) What breakpoint is used in the media query for detecting a tablet?

- ☐ 500px
- ☐ 600px
- ☐ 768px

2) Will the media query evaluate to true when the webpage is being printed?

- ☐ Yes
- ☐ No

3) Where is the media query placed in the CSS?

- ☐ Above the **body** rule.
- ☐ Inside the **body** rule.
- ☐ Below the **body** rule.

4) What CSS property names the sections used in the **grid-template-areas** value?

- ☐ **grid-template-columns**
- ☐ **grid-name**
- ☐ **grid-area**

5) What **grid-template-areas** value places the Members to the right of the Concerts and Contact section?

- ☐ `"head head"`
`"concerts members"`
`"contact members"`

- ☐ `"members members"`
`"concerts head"`
`"contact head"`

- ☐ `"head head"`
`"concerts contact"`
`"members members"`

Desktop design

The wider desktop screen provides enough room so the band member's unordered list may appear to the right of the band member's photo. The media query in the figure below changes the flex container with the class `band-member` so the flex items are ordered on the same row when the minimum browser width is 800 pixels.

Figure 10.7.3: Media query for desktop.

```
/* desktop */
@media all and (min-width: 800px)
{
    .band-member {
        flex-direction: row;
    }
}
```

©zyBooks 04/15/24 16:44 2071381
Marco Aguilar
CIS192_193_Spring_2024

Desktop design.

Resize the rendered webpage to see the changes take effect when the browser window is just below or just above 800 pixels wide.

Images from [Wikimedia.org](https://www.wikimedia.org/)

index.html

styles.css

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>Reach Out</title>
7   <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10   <header>
11     <h1>Reach Out</h1>
12     
14     <div class="nav">
15       <a href="#members">Members</a> &nbsp;
16       <a href="#concerts">Concerts</a> &nbsp;
```

Render webpage

Reset code

Your webpage

REACH OUT

PARTICIPATION
ACTIVITY

10.7.3: Desktop design.



- 1) The tablet media query is true on a desktop browser that is 900 pixels wide.
☐ True
☐ False
- 2) The desktop media query is true on a desktop browser that is 900 pixels wide.
☐ True
☐ False
- 3) The desktop media query is true on a desktop browser that is 700 pixels wide.
☐ True
☐ False
- 4) The desktop media query appears above the tablet media query.
☐ True
☐ False



Supporting multiple DPRs

In the final webpage, 3 images at various resolutions are created for each band member. The `srcset` attribute is used to download the member images appropriate for the device's DPR.

Figure 10.7.4: Member images for various DPRs.




		
150px width for 1x	300px width for 2x	450px width for 3x

Figure 10.7.5: Images using srcset attribute.

```


...

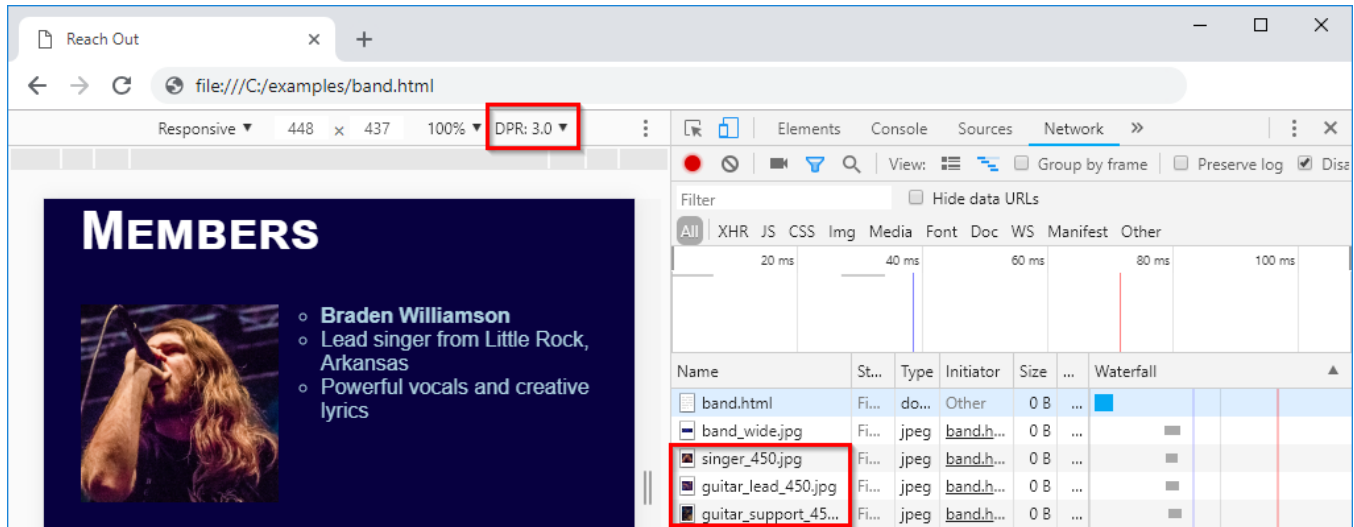
...


```

The final rendered webpage downloads the member image according to the device's DPR. The figure below shows the Chrome DevTools emulating a 3 DPR device to view the band webpage. The Network tab shows the 450px wide images are downloaded. If the DPR is changed to 2 and

the page is reloaded, the 300px wide images are downloaded. And if the DPR is changed to 1, the 150px wide images are downloaded.

Figure 10.7.6: Chrome DevTools emulating a 3 DPR device.



Complete webpage.

The rendered webpage does not look any different than before, but the correct band member image is downloaded according to the device's DPR.

Images from [Wikimedia.org](https://www.wikimedia.org/)

index.html

styles.css

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1">
6   <title>Reach Out</title>
7   <link rel="stylesheet" href="styles.css">
8 </head>
9 <body>
10  <header>
11    <h1>Reach Out</h1>
12    
14    <div class="nav">
15      <a href="#members">Members</a> &nbsp;
16      <a href="#concerts">Concerts</a> &nbsp;
```

Render webpage

Reset code

Your webpage

REACH OUT

PARTICIPATION
ACTIVITY

10.7.4: Final version.



1) The guitar_support_300.jpg image is downloaded when the device's DPR is 2.

- ☐ True
☐ False

2) The Pixel 2 with DPR 2.6 downloads the guitar_support_300.jpg image.

- ☐ True
☐ False

10.8 LAB: Media queries for a vacation website



This section's content is not available for print.

10.9 LAB: Responsive images for a vacation website



This section's content is not available for print.

10.10 LAB: Bootstrap for a vacation website

The provided Mediterranean Vacations webpage needs Bootstrap to make the webpage responsive.

Download Bootstrap (1 point)

Add `<link>` and `<script>` tags to `index.html` so the webpage downloads the necessary Bootstrap-related files:

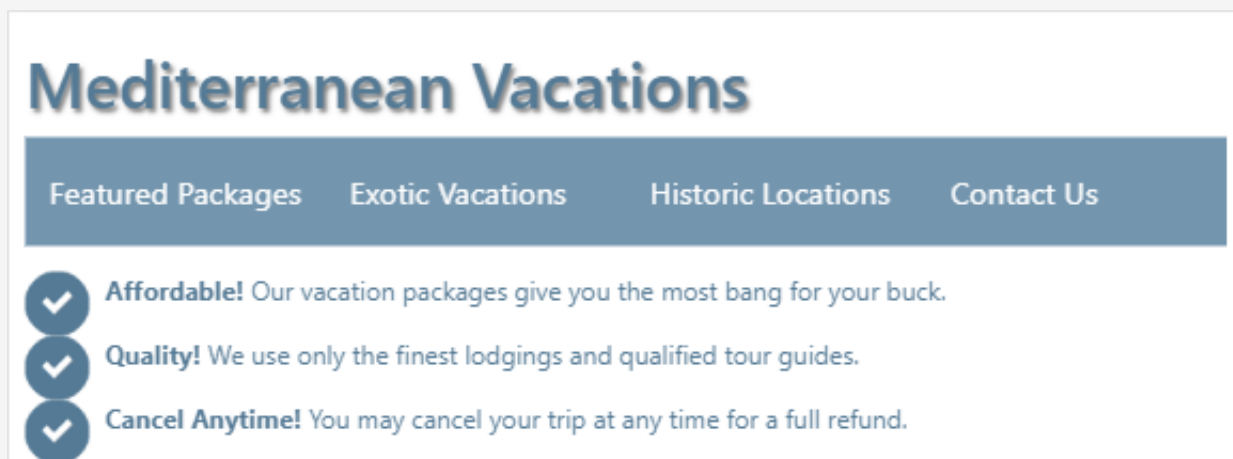
1. `https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css`
`integrity="sha384-1BmE4kWBq78iYhFIdvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"`
`crossorigin="anonymous"`
2. `https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js`
`integrity="sha384-QJHtvGhmr9XOIpI6YVutG+2QOK9T+ZnN4kzFN1RtK3zEFEIsxhlmWl5/YESvpZ13"`
`crossorigin="anonymous"`

Nav container (2 points)

Add the necessary Bootstrap classes to `<nav>` and the nav's child elements so the nav acts as a fluid container with one row. Each `` should use the `col-md` class to form one equally-spaced row when the viewport is at least 768px wide.

Do not add or remove any HTML elements.

The screenshot below the navigation links on a single row when the viewport is at least 768px wide.

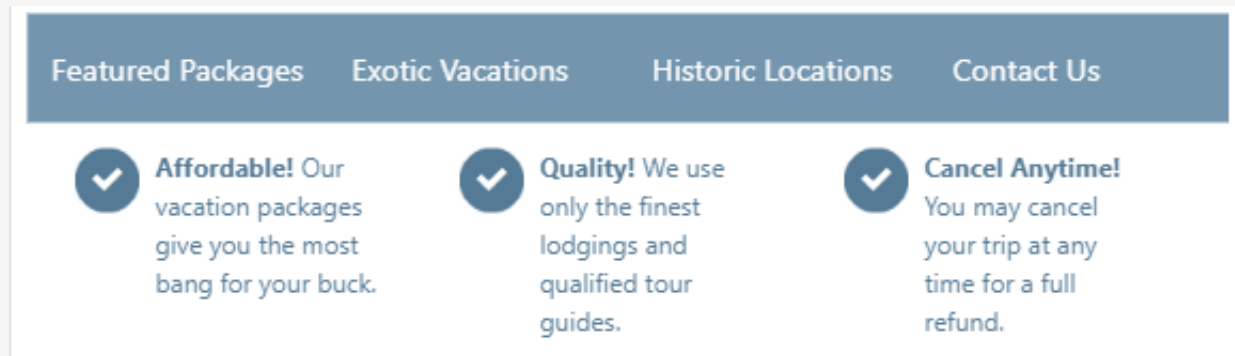


Benefits container (2 points)

Add the necessary Bootstrap classes to the `<div>` with ID `benefits` and the div's child elements so the div acts as a responsive fixed-width container that changes at various breakpoints. The inner divs should use the `col-sm` class to form one equally-spaced row when the viewport is at least 576px wide.

Do not add or remove any HTML elements.

The screenshot below shows the three benefits on a single row when the viewport is at least 576px wide.



Section div containers (2 points)

Each `<section>` has a child `<div>` that has a fluid container with one row. Use the `col-lg` class to create two equally-spaced columns so the photo is on the left and the text description is on the right when the viewport is at least 992px wide.

Do not add or remove any HTML elements.

The screenshot below shows the vacation photos with descriptions on the right when the viewport is at least 992px wide.

Featured Packages



Egypt

Egypt is a land of enduring beauty and rich culture. Explore the capital city of **Cairo**, the **Great Pyramids of Giza**, and the iconic **Great Sphinx**!

Packages include airfare, lodging, and several guided tours!

Exotic Vacations



Greece

Enjoy the scenic, peaceful Greek island of **Santorini**. Take a cruise, soak in the sun, and explore the idyllic villages that sit atop the cliffs!

Packages include airfare, overnight cruises, and lodging!

Responsive images (1 point)

Add the `rounded` class to the photos to give the photos rounded corners.

Add the `img-fluid` class to the photos so the photos stretch to fill the column when the viewport is at least 992px wide.

The screenshot below shows the rounded corners. The images grow in size as the viewport is made wider.

Featured Packages



Egypt

Egypt is a land of enduring beauty and rich culture. Explore the capital city of **Cairo**, the **Great Pyramids of Giza**, and the iconic **Great Sphinx**!

Packages include airfare, lodging, and several guided tours!

Exotic Vacations



Greece

Enjoy the scenic, peaceful Greek island of **Santorini**. Take a cruise, soak in the sun, and explore the idyllic villages that sit atop the cliffs!

Packages include airfare, overnight cruises, and lodging!

Show an Alert (2 points)

A Bootstrap Modal with ID `contact-modal` is defined in the section with ID `contact`. Clicking the "Send a message" button displays the Modal, and clicking either of the Modal's buttons closes the Modal.

Add an Alert with ID `conf-alert` immediately below the "Send a message" button. The Alert should use the following classes: `alert`, `alert-success`, and `collapse`. The Alert should read: "Thank you! We will contact you soon!"

The JavaScript code in `script.js` hides the "Send a message" button and shows the Alert when the Modal's "Send message" button is clicked. No JavaScript code modifications are required.

The screenshot below shows the Modal that is displayed when clicking the "Send a message" button. After the user types their information and click "Send message", the Modal disappears, and the Alert should appear in place of the "Send a message" button.

Contact Us

Our team of travel experts is readily available to make your

[Send a message](#)

Or call us at [1-800-111-2222](tel:1-800-111-2222).

Contact Us

Our team of travel experts is readily available to make your

Thank you! We will contact you soon!

Or call us at [1-800-111-2222](tel:1-800-111-2222).

Contact Us

Name?

Barbra Smith

Email?

barb@gmail.com

I'm interested in...

Booking a vacation.

Close

[Send message](#)

Photos by Frank McCown used with permission.

550544.4142762.qx3zqy7

LAB ACTIVITY

10.10.1: LAB: Bootstrap for a vacation website

8 / 10

Submission Instructions

Downloadable files

`index.html` , `styles.css` , `script.js` ,

`checkmark.png` , `israel_600.jpg` , `greece_600.jpg`

[Download](#)

, and `egypt_600.jpg`

Upload your files below by dragging and dropping into the area or choosing a file on your hard drive.

index.html

Drag file here
or

[Choose on hard drive.](#)

[Submit for grading](#)Coding trail of your work [What is this?](#)4/13 **S3,8** min:3

Latest submission - 11:44 PM PDT on 04/13/24

Total score: 8 / 10

☐ Only show failing tests[Download this submission](#)

1:QUnit test ^

1 / 1

Test Bootstrap libraries

- ✓ Two <link> tags exist
- ✓ <link> downloads styles.css
- ✓ <link> downloads bootstrap.min.css
- ✓ <script> with src
"https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.min.js" exists

2:QUnit test ^

2 / 2

Test nav container

- ✓ <nav> has class "container-fluid"
- ✓ has class "row"
- ✓ All tags have class "col-md"

3:QUnit test ^

2 / 2

Test benefits container

- ✓ <div> with id "benefits" has class "container"
- ✓ <div> with id "benefits" has child <div> with class "row"

✓ All <div> columns have class "col-sm"

4:QUnit test ^

2 / 2

Test section div containers

✓ All section <div> containers have class "container-fluid"

✓ All section <div> containers have child <div> with class "row"

✓ All <div> columns have class "col-lg"

5:QUnit test ^

1 / 1

Test responsive images

✓ All vacation tags have class "rounded"

✓ All vacation tags have class "img-fluid"

6:QUnit test ^

0 / 2

Test Modal and Alert

✓ "Send a message" button is initially visible

✗ Alert with id conf-alert exists

Your value

Expected value

✗ Test aborted.

Previous submissions

11:41 PM on 4/13/24

3 / 10

[View](#) v