

CS 2033

Multimedia & Communications II

LECTURE 9 – MODERN WEBSITE FEATURES

Modern features

2

- ▶ Websites can be modernized with cool effects and features.
- ▶ What are common features of professional modern websites?
- ▶ Think about both functionality and aesthetics as we look at examples of modern sites.

Modern features

3

▶ <http://www.thebeet.ca/>

▶ <https://www.skyzone.com/ca-toronto>

▶ <https://nasaprospect.com/>

Modern features

4

- ▶ Parallax effect
- ▶ Scrollfire
- ▶ Accessibility
- ▶ Responsive
- ▶ And others

Modern features

5

▶ Parallax effect

- ▶ Elements moving at different rates to create the illusion of depth.

▶ Scrollfire

- ▶ Elements transition in as user approaches them.
- ▶ Track section in navigation bar as you scroll.

Modern features

6

▶ **Accessibility**

- ▶ Allowing people with disabilities or impairments to use the website.

▶ **Responsive**

- ▶ Display nicely on any platform, device, and screen size.

Parallax

7

- ▶ **Parallax** is a great aesthetic feature that creates the illusion of depth (3D) in the website.
- ▶ Produced by different rates of motion as we scroll up and down.
- ▶ For example, text moves at a different rate than the background image behind it.

Parallax

8

- ▶ By default, everything moves at the same rate as the user's scrolling.
- ▶ We want to change that!
- ▶ There are different kinds of parallax.
- ▶ The simplest is no movement on the background while the foreground moves with the scrolling.
- ▶ This simple form uses just CSS.

Parallax

9

- ▶ Add an image to an element using the CSS background-image style.
- ▶ The background-attachment style indicates whether the image scrolls.
- ▶

```
.myclass {  
    width:100%; height:300px;  
    background-image: url('sky.jpg');  
    background-attachment: fixed;  
}
```

Parallax

10

- ▶ We can enhance this by including motion on both the background and foreground elements.
- ▶ This requires JS so that we can access the elements and change their positions dynamically.
- ▶ This form of parallax is more elegant than the pure CSS form.

Parallax

11

- ▶ There's another event listener that helps with this!
 - ▶ **onscroll** – triggers as the user scrolls up or down the site
- ▶ Note there are many ways to scroll with different sized jumps:
 - ▶ Mousewheel
 - ▶ Up/Down keys, PgUp/PgDn keys
 - ▶ Clicking or dragging the scrollbar

Parallax

12

- ▶ Add onscroll listener to the body.
- ▶ Retrieve the current scroll position in pixels using `window.scrollY`.
 - ▶ This value starts at 0 from the top and increases per pixel scrolled.
- ▶ Now we can use this value to calculate the elements' positions.
- ▶ How do we calculate this?

Parallax

13

- ▶ Linear equations work well.
 - ▶ $y = mx + b$
 - ▶ x: scroll position (independent)
 - ▶ y: image position (dependent)
 - ▶ m: amount of change (slope)
 - ▶ b: position when x=0 (y-intercept)
- ▶ m and b are the parameters.

Parallax

14

- ▶ Let's start with b . Remember b is the position when $x=0$ (no scrolling).
- ▶ Thus, b must be where we want the element to start by default.
- ▶ i.e. if we use CSS to position an element at 50px from the top, then b in our linear equation must also be 50px.

Parallax

15

- ▶ The m parameter is the rate of movement of our element.
- ▶ $m=1$ means no movement.
- ▶ $m<1$ means up movement.
- ▶ $m>1$ means down movement.
- ▶ Values are typically around 0-1.
- ▶ Play with it until it looks good!

Parallax

16

- ▶ For that type of parallax, we used JS to change the CSS margin-top or top or other position property.
- ▶ An alternative option is to change the background-position of the background layer/element.
- ▶ This property doesn't change the position of any element, but rather where the image begins.

Parallax

17

► Samples:

► CSS Parallax

► Enhanced JS Parallax

► Jumpy (Bad) Parallax



<https://www.csd.uwo.ca/courses/CS2033b/samples/lec9/>

Scrollfire

18

- ▶ Scrollfire is another feature based on scrolling, as its name suggests.
- ▶ This is when elements appear or change as you scroll into specific ranges.
- ▶ Common form is applying an entry transition on an element that triggers as you scroll near it.

Scrollfire

19

- ▶ For example, have an image in your website that starts hidden and appears as you approach it.
- ▶ In some cases, it may cause an existing image to grow or slide over when you enter a specific range.
- ▶ Sample:
<https://www.csd.uwo.ca/courses/CS2033b/samples/lec9/>

Scrollfire

20

- ▶ Navigation bars use derivatives of this to change their placement or "stickyness" depending on the scroll.
- ▶ Some also highlight the current section name as you scroll through.
- ▶ Example:
 - ▶ Highlight Section Example

Scrollfire

21

- ▶ Like parallax, we start with an onscroll listener and retrieve the scroll position in the event handler.
- ▶ Rather than calculating a position, use conditionals to check the range of the scroll.
- ▶ Change classes or individual styles depending on the range.

Scrollfire

22

- ▶ Making the element visible isn't that appealing on its own.
- ▶ Often there is a smooth transition, like a fade-in or slide-in, which looks nicer than a sudden appearance.
 - ▶ Remember the transition property we learned previously.

Accessibility

23

- ▶ Websites are expected to be accessible by everyone, including those with disabilities.
- ▶ For example, those with colour blindness or limited hand control should be able to use websites.
- ▶ Standards have been put in place to ensure sites are fully accessible.

Accessibility

24

- ▶ Blind users depend on screen readers to read the content of websites aloud.
- ▶ Deaf users depend on text transcriptions of audio/video.
- ▶ Other disabilities may mean other tools or appliances are required.

Accessibility

25

► Text content

- Have no or minimal text in images.
- Screen readers read text but cannot read text inside images.
- Keep the text organized and use proper spelling and grammar.
- Have fonts clear to read or an option to change font size.

AAAAA

Accessibility

26

▶ **Alt text**

- ▶ Alternate text is what shows up when an image doesn't load.
- ▶ Screen readers also read this.
- ▶ This is good practice in general – but now even more so!

Accessibility

27

► Clear colours

- Colour blindness is common, affecting nearly 5% of people.
- Use crisp colours with high contrast (i.e. black text on white background or vice versa).
- Be careful with colour-coding the text in your website.

Accessibility

28

► Overlaid text

- Text overlaid on an image must be entirely high contrast.
- This is difficult when images have many varying colours/shades.
- Use outline or shadows or an intermediate coloured panel to split the background and foreground.

Accessibility

29

▶ Audio

- ▶ If you have audio, transcribe it to text and provide the text in a file.

▶ Video

- ▶ If you have videos, provide closed captioning at least as an available option for deaf viewers.

Responsive

30

- ▶ 20 years ago, there wasn't much variety in computer screen sizes.
- ▶ It is the opposite today. Screens come in a wide range of sizes from small phones to TV screens hooked up to computers.
- ▶ Websites need to look good across all different screens and platforms.

Responsive

31

- ▶ This is **responsive** web design.
- ▶ Creating a website now takes more thought and effort than it did 20 years ago.
- ▶ However, technologies have improved to help with this process.
- ▶ Before we look at them, let's talk about common screen sizes.

Responsive

32



Responsive

33

- ▶ Suggested screen size ranges (not every device conforms to these ranges)
 - ▶ Phones: $\leq 640\text{px}$
 - ▶ Tablets: 641px to 1007px
 - ▶ Monitors: $\geq 1008\text{px}$
- ▶ This is a good guide for responsive web design.

Responsive

34

- ▶ Now we know the ranges, but how do we design a site around them?
- ▶ Different CSS rule-sets!
- ▶ We have a banner 1200px wide. It can be full size on a monitor but smaller (i.e. 300px) on mobile.
- ▶ % values adapt to window but it's not always feasible or preferable to use a %-based size.

Responsive

35

- ▶ In addition to small screens, we have to consider users on a large screen but a small browser window.
- ▶ This isn't a major issue but we need to be aware of it for responsiveness.
- ▶ For the most part, this can be handled the same way screen sizes are handled.

Responsive

36

- ▶ A basic approach is to use JS to get the browser size and load the CSS accordingly.
- ▶ `var w = window.innerWidth;`
- ▶ `if (w < 320) {
 // Load mobile CSS
} else if (w < 800) {
 // Load tablet CSS
} ...`

Responsive

37

- ▶ To load a CSS file from within the JS conditionals, use the HTML line to load a CSS file within the JS function `document.write()`
- ▶ Be careful with your quotations!

```
<script type="text/javascript">
  var w = window.innerWidth;

  if (w <= 640) {
    // Load mobile stylesheet.
    document.write('<link rel="stylesheet" type="text/css" href="mob-styles.css">');
  } else if (w <= 1007) {
    // Load tablet stylesheet.
    document.write('<link rel="stylesheet" type="text/css" href="tab-styles.css">');
  } else {
    // Load desktop stylesheet.
    document.write('<link rel="stylesheet" type="text/css" href="dsk-styles.css">');
  }
</script>
```

Responsive

38

- ▶ This way is a bit clunky.
- ▶ Some potential problems with loading stylesheets this way.
 - ▶ **Disabled JavaScript** – would result in no styles or only default styles loaded
 - ▶ **Resizing browser** – loads files at initial window size
 - ▶ **A lot of code** – consider separate pages (similar to internal CSS issues)

Responsive

39

- ▶ CSS includes **media queries** which allow us to specify which files load depending on the screen size (or other properties) without using JS.
- ▶ Add media queries as attributes within the HTML `<link>` tag for loading the CSS files.

Responsive

40

- ▶ `<link rel="stylesheet" href="styles.css" media="(max-width: 640px)">`
- ▶ Set the min-width and/or max-width that apply to this stylesheet.
- ▶ You may also specify the screen's orientation (**landscape** for wider or **portrait** for taller).

Responsive

41

- ▶ The CSS media queries also provide a third way for us to make our sites responsive.
- ▶ In this approach, we load the CSS files the way we did before, without any media queries in the HTML.
- ▶ Instead, media queries are within the stylesheet(s).

Responsive

42

- ▶ These media queries contain rule-sets that apply to the specified screen properties.

- ▶ `@media (min-width: 641px) {`
 `p, .title {`
 `color:red;`
 `width:300px;`
 `}`
`}`

Responsive

43

- ▶ Both media query options are better than the JavaScript option.
- ▶ The option in the HTML forces you to keep separate, organized files.
- ▶ The option in the stylesheets allows for more freedom with files. This is both a blessing and a curse.
 - ▶ Too much freedom can result in disorganized code.

Responsive

44

- ▶ Now we know the methods of creating different rule-sets for the different screen size ranges.
- ▶ The next step is figuring out how the site should actually look on each of the screens.
- ▶ Which elements can stay the same and which have to change?

Responsive

45

▶ Column structure

- ▶ 1 column is best on small screens
- ▶ 2 columns could fit on tablets
- ▶ Resize your columns accordingly
- ▶ If you have 3 columns across a normal screen, make them wider on mobiles so they become single columns.

Responsive

46

► Navigation menu

- Navigation menus should become vertical on smaller screens.
- These often get added to an expandable list that can be toggled open and closed.
- Use JS to switch between open and closed classes on clicking a button.
- Example

Responsive

47

► Fonts

- Titles or headers with very large font might not fit on a mobile screen.
- Make those font sizes smaller and/or change the unit type.
- Units **vw** and **vh** are relative to window size (similar to %) so they may be helpful for responsiveness.

Responsive

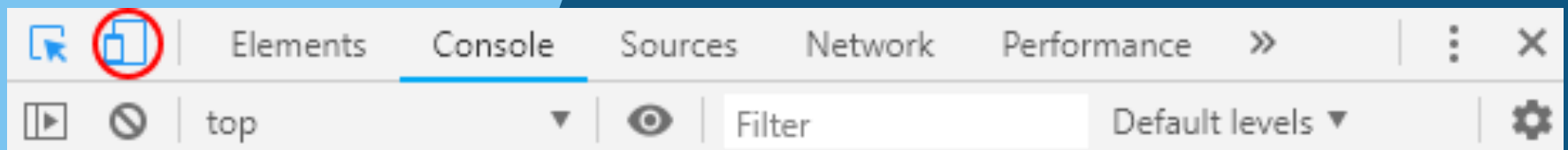
48

- ▶ Test the site on various screen sizes.
- ▶ If your site is on a live server, open it from different devices if available.
- ▶ If it's only stored on your computer or you don't have access to other devices, there are some ways to test it on your own computer.

Responsive

49

- ▶ Resize your browser.
 - ▶ Simulate a smaller screen by resizing your browser big and small.
- ▶ Use the Chrome screen emulator.
 - ▶ Click the three dots, More tools, Developer Tools, then click the little icon of the phone and tablet.



Responsive

50

- ▶ Chrome screen emulator (cont'd)
 - ▶ This toggles the device mode in which you can select a specific device or free responsive mode.
 - ▶ Several device specs are provided so you can test on those sizes.
 - ▶ The "Responsive" mode allows you to resize the emulator freely.

Responsive

51

- ▶ Use other emulators.
 - ▶ <https://bluetree.ai/screenfly/>
 - ▶ This only works for online sites but it also has many device emulators.

Responsive

52

- ▶ Don't just test the site at a couple stationary sizes.
- ▶ Resize the window back and forth and make sure the site looks fine at all sizes as you resize.
- ▶ Tip: it may help to change the background colour for each device during testing phases (remove it when finished).