Front-End Web Development

Unit 10: Dynamic HTML

Course Outline

Course Learning Outcomes

- Competently write HTML and CSS code
- Create web page layouts according to requirements using styles
- Add interactivity to a web page with JavaScript
- Access and display third-party data on the web page
- Leverage Bootstrap and Static Site Generator

Assessment

- Final Project - 100% of the grade

  - Design and Build functioning Website using HTML5, CSS (including Bootstrap), JavaScript (browser only)

  ✓ Code will be managed in GitHub
  ✓ Website will be deployed to GitHub Pages
  ✓ All code to follow best practice and be documented

  - Details and How-To-Guide are available on the course page under the section called Assessments

10. Dynamic HTML

In This Unit

Title
Walkthrough of Carousel, Lightbox
Walkthrough of loading table data from JSON
Walkthrough of building a grid of images from JSON

Walkthrough of Carousel, Lightbox

Carousels, aka Sliders

Sliders are a set of frames, wherein each frame can be traversed respectively. Frames in Sliders can be images, videos, or even HTML elements (as in the case of testimonials or reviews)

Background

- Carousels allow multiple pieces of content (frames) to occupy a single, coveted space, e.g. top of homepage or section
- Good navigation and content can help make it effective
- A static hero or integrating content in the UI may be a better solution, it depends on the context
- offers some indication (or navigation) that there is more than one piece of featured content, or frame, within the carousel
- contains images and a small amount of text in each frame

Resources

- https://www.nngroup.com/articles/designing-effective-carousels/
- https://www.smashingmagazine.com/2022/04/designing-better-carousel-ux/

Carousel - Basic Example

1. Create the Basic Layout of the Image Slider using HTML Code
2. Add Prev and Next Buttons
3. Add the Required Images and Text to the Slider
4. Activate the two Buttons using JavaScript Code
5. Automate the movement of the frames (optional)
6. Add navigation dots
7. Wide variety of libraries available but can be restrictive

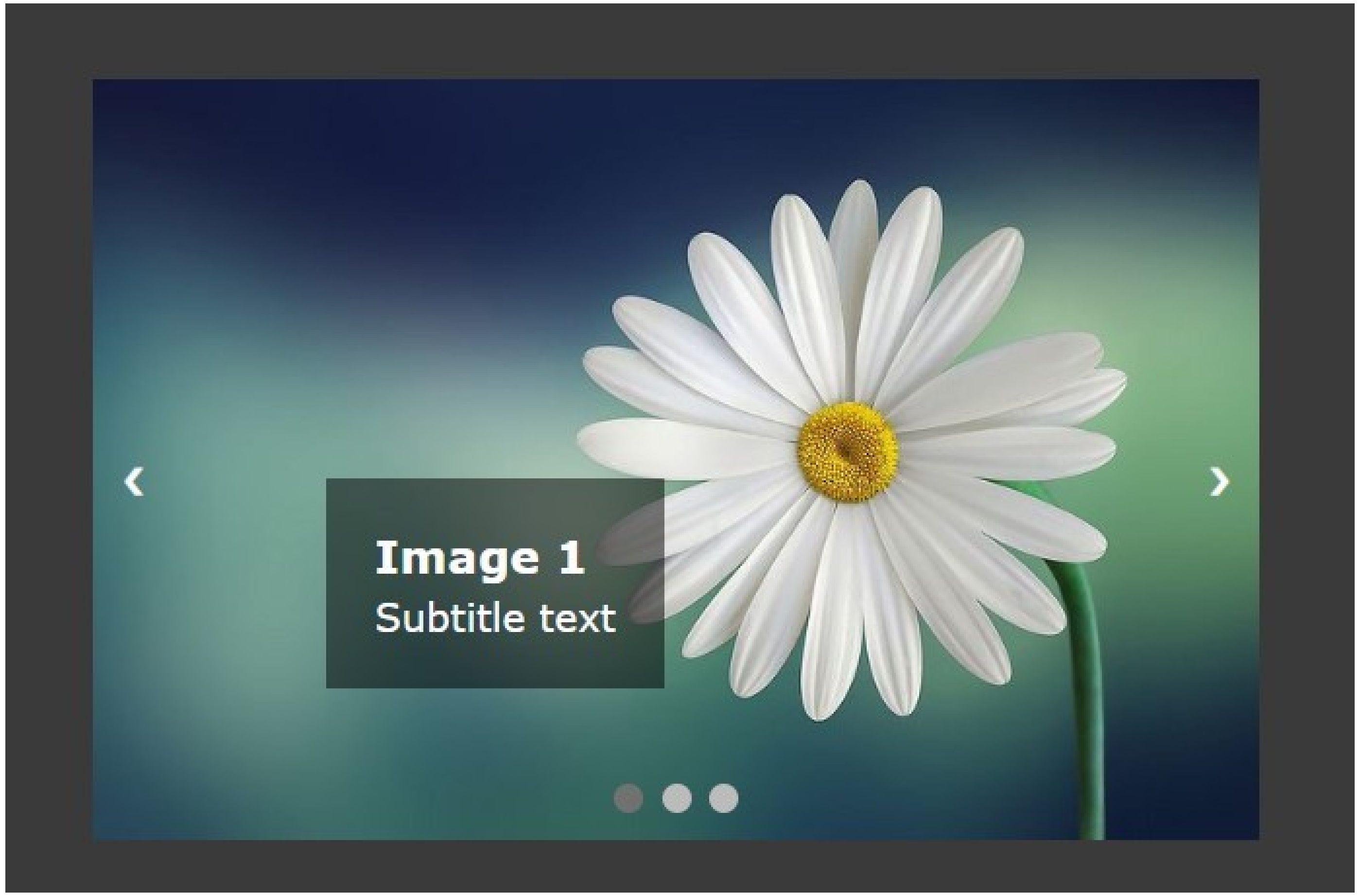
Carousel - UX Checklist

As usual, here's a general checklist of a few important guidelines to consider when designing better carousels:

- Choose the sequence of slides carefully. Most important slides always come first, 5 slides (max)
- Limit the height of the slides to 45-50% of the screen's height (max)
- Slides shouldn't rotate too quickly (min delay of 5–7s).
- Try to avoid auto-rotation on mobile. Always pause auto-rotation on hover, stop on interaction
- Don't rely on dragging the carousel alone. Make sure the slides are keyboard-accessible.

  Always support swipe gestures on mobile
- Always indicate a slice of the upcoming slide
- Always show at which slide a user currently is
- Consider replacing progress dots with labels, thumbnails, key highlights
- On desktop, group prev/next steps and display them above the carousel
- On mobile, group prev/next steps and display them below the carousel

Demo



HTML

```html
<section class="component-carousel">
  <div id="carousel-container" class="carousel-container">
    <div class="slider-carousel">
      <div class="slide main">
        <img src="https://cdn.pixabay.com/photo/2015/04/19/08/32/marguerite-729510__480.jpg" alt="flower 1" />
        <div class="image-text">
          <h3>Image 1</h3>
          <p>Subtitle text</p>
        </div>
      </div>
      <div class="slide">
        <img src="https://cdn.pixabay.com/photo/2014/02/27/16/10/tree-276014_960_720.jpg" alt="flower 2" />
        <div class="image-text">Image 2</div>
      </div>
      <div class="slide">
        <img src="https://cdn.pixabay.com/photo/2015/04/23/22/00/tree-736885_960_720.jpg" alt="flower 3" />
        <div class="image-text">Image 3</div>
      </div>
      <div class="navigation-buttons">
        <a class="previous">❮</a>
        <a class="next">❯</a>
      </div>
    </div>
    <!-- Navigation Dots-->
    <div class="navigation-dot-container">
      <span class="navigation-dot"></span>
      <span class="navigation-dot"></span>
      <span class="navigation-dot"></span>
    </div>
  </div>
</section>
```

JavaScript - 1

```javascript
(() => {
  // globals
  const componentCarousel = document.querySelector('.component-carousel');
  const navigationDots = componentCarousel.querySelectorAll('.navigation-dot');
  const navigationButtons = componentCarousel.querySelectorAll('.navigation-buttons  > a')
  const slides = document.getElementsByClassName("slide");

  let currentIndex = 0;

  //Initiate moving of slides
  function showSlides(n) {
    let i;
    currentIndex = n;
    for (let i=0; i<slides.length; i++) {
      slides[i].style.display = "none";
    }
    for (let i=0; i<navigationDots.length; i++) {
      navigationDots[i].className = navigationDots[i].className.replace(" active", "");
    }
    slides[currentIndex].style.display = "block";
    navigationDots[currentIndex].className += " active";
  }

  // named function expression
  const incrementIndex = (increment) => Math.abs((currentIndex + increment) % slides.length);
```

JavaScript - 2

```
  function initCarousel() {
    showSlides(currentIndex);
    // add event handlers for navigation buttons
    navigationButtons.forEach(button => {
      button.addEventListener('click', (event) => {
        event.preventDefault();
        if (event.target.className === 'next') {
          currentIndex = incrementIndex(1);
        }
        if (event.target.className === 'previous') {
          currentIndex = incrementIndex(-1);
        }
        showSlides(currentIndex);
      });
    });
    // add event handlers for navigation dots
    navigationDots.forEach(dot => {
      dot.addEventListener('click', (event) => {
        // find index of current dot
        const parent = event.target.parentNode;
        const index = [].indexOf.call(parent.children, event.target);
        showSlides(index);
      });
    });
    // automate movement of the slides
    /*setInterval(() => {
      showSlides(incrementIndex(1));
    }, "2000"); */
  }

  window.addEventListener("load", (event => {
    initCarousel();
  }));

})();
```

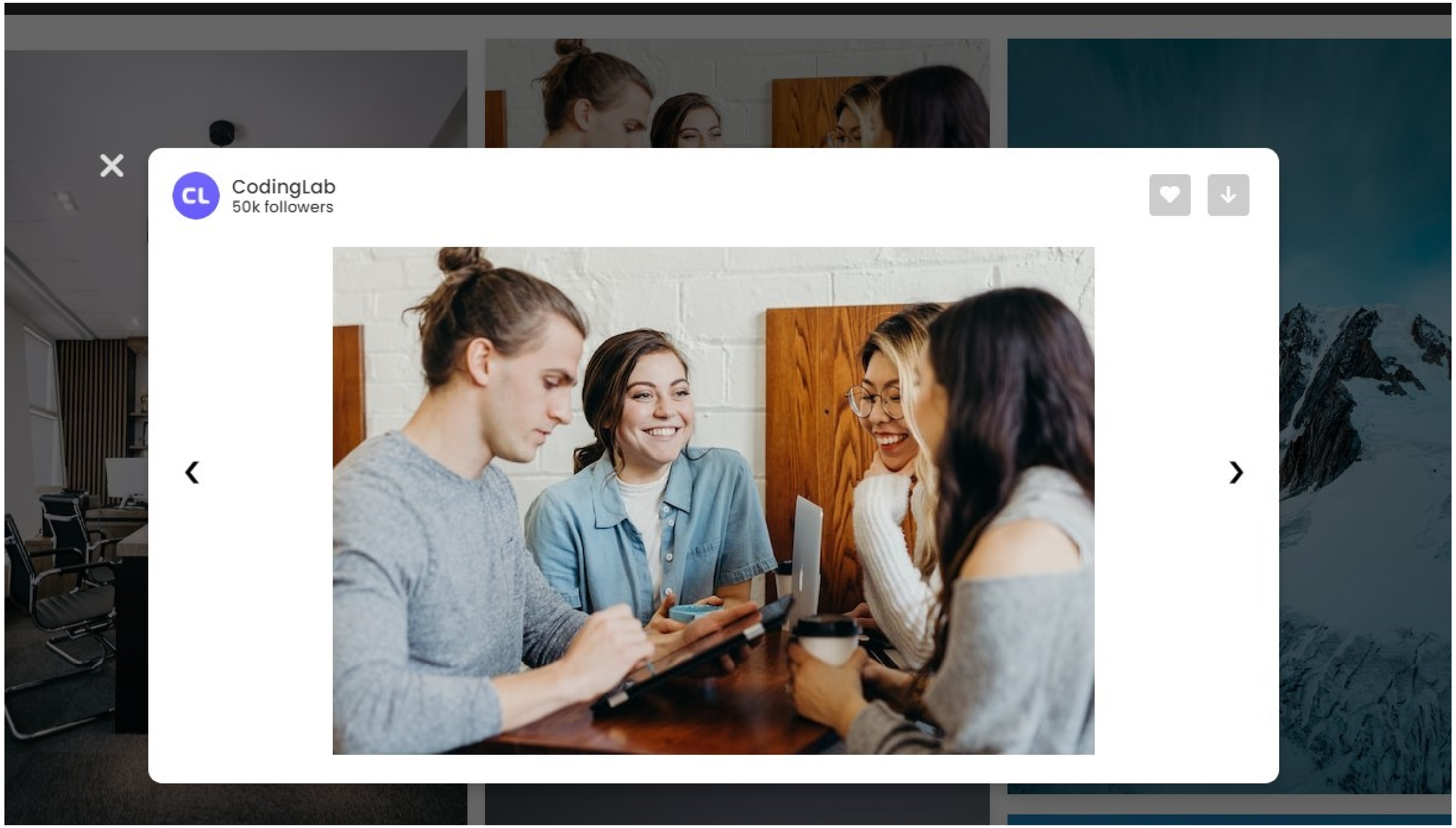
Lightboxes

A lightbox refers to a user interface element or overlay that is used to display images, videos, or other multimedia content in a focused and visually appealing manner. When activated, the lightbox typically dims the background content and highlights the displayed media, providing a distraction-free viewing experience

Lightboxes are commonly used in web design for several reasons:

- **Enhanced user experience:** Lightboxes provide an immersive and interactive way to showcase media content without redirecting users to a separate page. This creates a seamless and engaging user experience.
- **Visual appeal:** Lightboxes often feature a sleek and visually appealing design, with customizable elements such as transitions, overlays, and captions. They help present media content in an aesthetically pleasing and attention-grabbing manner.
- **Focus on media content:** By dimming the background or hiding other elements on the page, lightboxes draw the viewer's attention solely to the displayed media. This allows for a more focused and uninterrupted viewing experience.
- **Accessibility:** Lightboxes can enhance accessibility by providing keyboard navigation options, alternative text descriptions, and other features that accommodate users with disabilities. They can also be designed to be responsive and compatible with various devices and screen sizes.
- https://www.nngroup.com/articles/overuse-of-overlays/

Demo



HTML

```html
<div class="image-gallery">
  <header>Masonry Image Gallery</header>
  <div class="image-container">
    <div class="image-box">
      <img class="gImg" src="images/img1.jpg" alt="" />
      <div class="logo_icons">
        <a href="#">
          <img class="logoImg" src="images/logo.png" alt="" />
          <div class="text_content">
            <span class="name">CodingLab</span>
            <span class="followers">50k followers</span>
          </div>
        </a>
        <div class="icons">
          <i class="fas fa-heart"></i>
          <i class="fas fa-arrow-down"></i>
        </div>
      </div>
    </div>
    <div class="image-box">
      <img class="gImg" src="images/img2.jpg" alt="" />
      <div class="logo_icons">
```

JavaScript - 1

```
(() => {
  // globals
  let currImage = 0;
  let body  =  document.querySelector("body"),
      lightBox = document.querySelector(".lightBox"),
      images = document.querySelectorAll(".gImg"),
      showImg = lightBox.querySelector(".showImg img"),
      close = lightBox.querySelector(".close");

  function lightbox() {
    for (const [index, image] of images.entries()) {
      // https://medium.com/@_DandyLyons/how-to-use-a-js-for-of-loop-with-an-index-a4675ed22351
      image.addEventListener("click", () => {
        showImg.src = image.src;
        currImage = index;
        lightBox.style.display = "block";
        body.style.overflow = "hidden";
        close.onclick = ()=>{
          lightBox.style.display = "none";
          body.style.overflow = "visible";
        };
      });
    }
```

JavaScript - 2

```
      // add next/prev links to lightbox
      lightBox.querySelector('.lightBox_content .showImg').insertAdjacentHTML("afterend",
      `<div class="navigation-buttons">
        <a class="previous">❮</a>
        <a class="next">❯</a>
      </div>`);

      lightBox.querySelector('.navigation-buttons .previous').addEventListener("click", (event => {
        if (currImage !== 0) {
          currImage -= 1;
          showImg.src = images[currImage].src;
        }
      }));
      lightBox.querySelector('.navigation-buttons .next').addEventListener("click", (event => {
        if (currImage !== (images.length-1)) {
          currImage += 1;
          showImg.src = images[currImage].src;
        }

      }));

    }
```

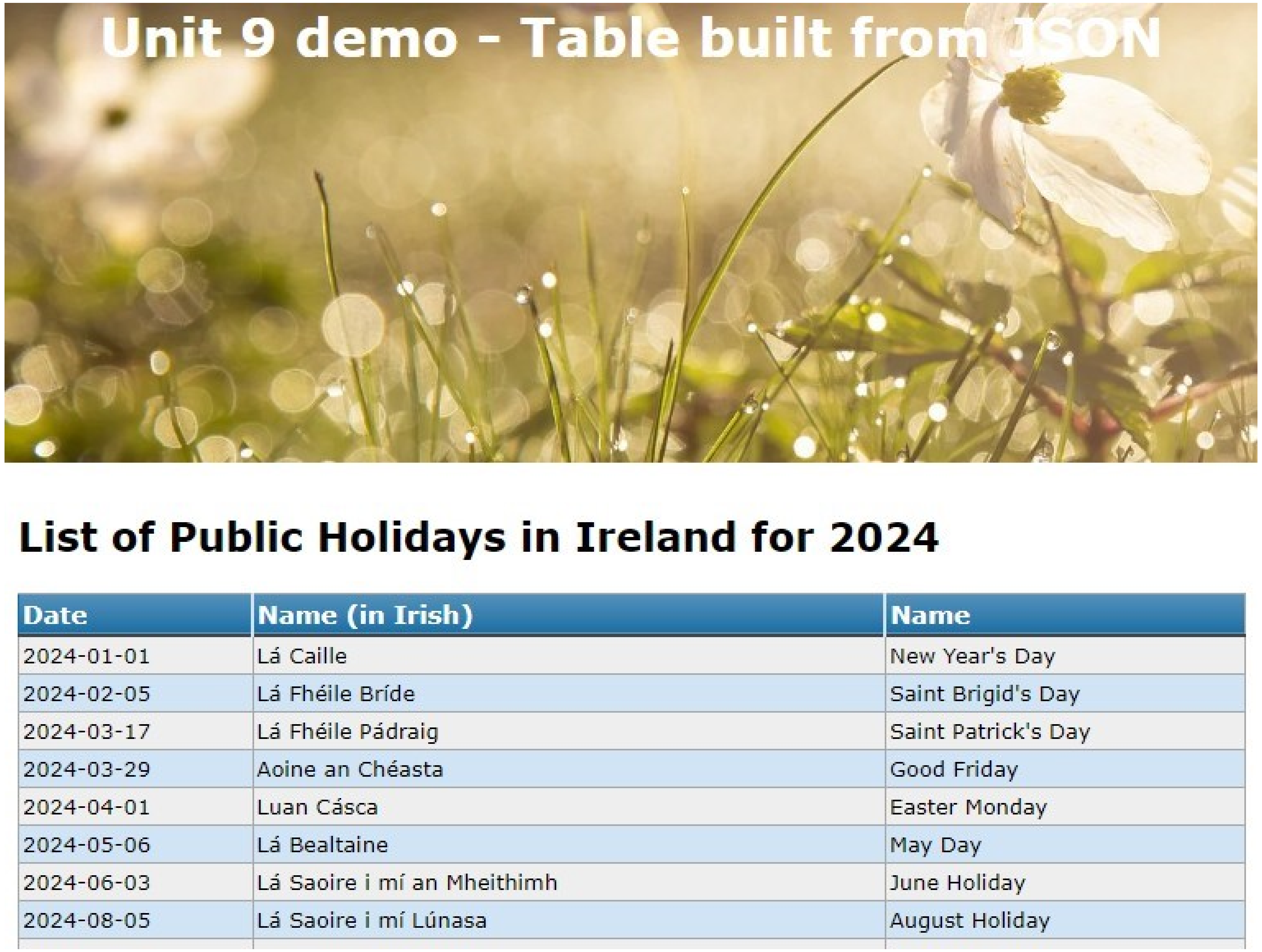
Activity

Breakout

- Join a breakout room
- Download the unit 10 exercises code from Moodle
- Follow the instructions and complete the exercises
- You have 35 minutes
- Lecturer will visit each room in turn, etc...
- Will start next topic on the hour

Walkthrough of loading table data from JSON

Table Data (via JSON)



Unit 9 demo - Table built from JSON

List of Public Holidays in Ireland for 2024

Date	Name (in Irish)	Name
2024-01-01	Lá Caille	New Year's Day
2024-02-05	Lá Fhéile Bríde	Saint Brigid's Day
2024-03-17	Lá Fhéile Pádraig	Saint Patrick's Day
2024-03-29	Aoine an Chéasta	Good Friday
2024-04-01	Luan Cásca	Easter Monday
2024-05-06	Lá Bealtaine	May Day
2024-06-03	Lá Saoire i mí an Mheithimh	June Holiday
2024-08-05	Lá Saoire i mí Lúnasa	August Holiday

HTML

Problem: Load a list of Irish public holidays (JSON) and display them in a table using JavaScript

- load JSON

```
<script>
    // generated by https://date.nager.at/Api
    // https://date.nager.at/api/v3/PublicHolidays/2024/IE
    let data = [{"date":"2024-01-01","localName":"Lá Caille",
    "name":"New Year's Day","countryCode":"IE","fixed":false,
    "global":true,"counties":null,"launchYear":null,"types":["Public"]},
    {"date":"2024-02-05","localName":"Lá Fhéile Bríde","name":"Saint Brigid's Day",
    "countryCode":"IE","fixed":false,"global":true,"counties":null,"launchYear":null,
    "types":["Public"]},{"date":"2024-03-17","localName":"Lá Fhéile Pádraig",
    "name":"Saint Patrick's Day","countryCode":"IE","fixed":false,"global":true,...
</script>
```

JavaScript - Load the table data JSON

```
function init() {
    try {
        // load data as a JS object
        console.log(data);
        renderTable(data); // render table
    } catch(err) {
        console.error(err);
        contentContainer.innerHTML = '<h2>Error</h2><p>No public holidays to display.</p><p>' + err + '</p>';
    }
}

window.addEventListener("load", (event => {
    init();
}));
```

JavaScript- Generate the HTML

```
function renderTable(publicHolidays){
    let hols = publicHolidays;
    let outputHtml = '<div class="table-container">';

    outputHtml +='<h2>List of Public Holidays in Ireland for
    2024</h2>'; outputHtml += '<table class="blueTable">';
    outputHtml += '  <thead>';
    outputHtml += '    <tr>';
    outputHtml += '      <th>Date</th>';
    outputHtml += '      <th>Name (in Irish)</th>';
    outputHtml += '      <th>Name</th>';
    outputHtml += '    </tr>';
    outputHtml += '  </thead>';
    outputHtml += '  <tbody>';

    for (let i in hols) {
        outputHtml += '<tr>';
        outputHtml += '  <td>' + hols[i].date + '</td>';
        outputHtml += '  <td>' + hols[i].localName + '</td>';
        outputHtml += '  <td>' + hols[i].name + '</td>';
        outputHtml += '</tr>';
    }
    outputHtml += '  </tbody>';
    outputHtml += '</table>';
    outputHtml += '</div>';

    contentContainer.innerHTML = outputHtml;
}
```

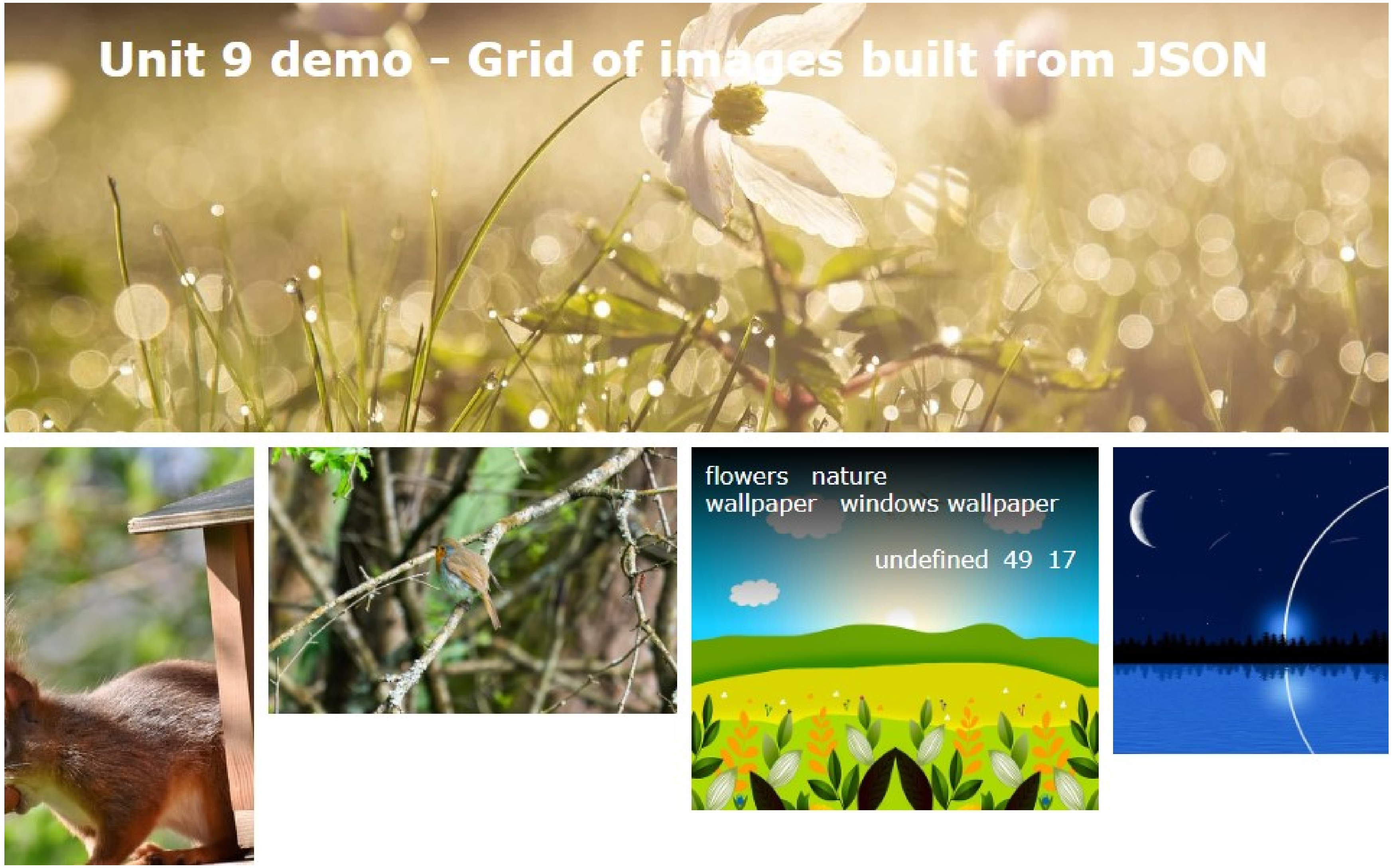
Activity

Breakout

- Join a breakout room
- Continue working on the unit exercises
- You have 35 minutes
- Lecturer will visit each room in turn, etc...
- Will start next topic on the hour

Walkthrough of building a grid of images from JSON

Grid of Images (via JSON)



Unit 9 demo - Grid of images built from JSON

flowers nature
wallpaper windows wallpaper

undefined 49 17

HTML

Problem: Load a list of images and associated metadata (JSON) and display them in a grid format using JavaScript

- number of ways to load JSON

```html
<!-- option 1 - JS object -->
<script>
    let dataOption1 = {...};
</script>
<!-- option 2 - JS string -->
<script>
    let dataStringOption2 = '{...}';
</script>
<!-- option 3 - embedded JSON -->
<script id="dataStringOption3" type="application/json">
  {...}
</script>
<!-- option 4 - using AJAX/Fetch -->
<!-- will be covered in unit 12 -->
```

Option 4 is the preferred option but the others may be suitable depending on the situation

JavaScript - Load the image JSON

```
function init() {
    try {
        // option 1 - load data as a JB object
        console.log(dataOption1);
        renderImages(dataOption1); // render images
        // option 2 - load data as a string variable
        const dataOption2 = JSON.parse(dataStringOption2);
        console.log(dataOption2);
        // option 3 - embed data in html
        const dataOption3 = JSON.parse(document.getElementById('dataStringOption3').text);
        console.log(dataOption3)
    } catch(err) {
        console.error(err);
        contentContainer.innerHTML = '<h2>Error</h2><p>No images to display.</p><p>' + err + '</p>';
    }
}

window.addEventListener("load", (event => {
    init();
}));
```

JavaScript- Generate the HTML

```
function renderImages(images){
    let imagesHtml ='';
    imagesHtml = '<div class="image-container">';
    for (let i in images.hits) {
        imagesHtml += '<div class="image">’;
        imagesHtml += '<a title="Click to see a large version of this image" href="' + images.hits[i].largeImageURL + '">';
        imagesHtml += '<img src="' + images.hits[i].webformatURL + '">’;
        imagesHtml += '</a>’;
        imagesHtml += '<div class="image-info">'; imagesHtml += '<div class="tags">’;
        let tagsArray = images.hits[i].tags.split(',’);
        for (j in tagsArray) {
             let linkQuery = encodeURI(tagsArray[j].trim() );
             let linkText = tagsArray[j].trim();
             imagesHtml += '<a href="#" onclick="loadImages(\'' +      linkQuery + '\')">' + linkText + '</a>’;
        }
         imagesHtml += '</div>';
         imagesHtml += '<div class="right">’;
          imagesHtml += '<span class="favorites">' + images.hits[i].favorites + '</span>’;
          imagesHtml += '<span class="likes">' + images.hits[i].likes + '</span>’;
         imagesHtml += '<span class="comments">' + images.hits[i].comments + '</span>’;
          imagesHtml += '</div></div></div>';
     }
     imagesHtml += '</div>';

     contentContainer.innerHTML = imagesHtml;
}
```

PROFESSIONAL ACADEMY

Activity

Breakout

- Join a breakout room
- Continue working on the unit exercises
- You have 35 minutes
- Lecturer will visit each room in turn, etc...

Summary

Completed This Week

- Walkthrough of Carousel, Lightbox
- Walkthrough of loading table data from JSON
- Walkthrough of building a grid of images from JSON

For Next Week

- Complete the remaining exercises for unit 9 before next class
- Review the slides and examples for unit 10