Comp 324/424 - Client-side Web Design - Slides

Spring Semester 2018 - week 11

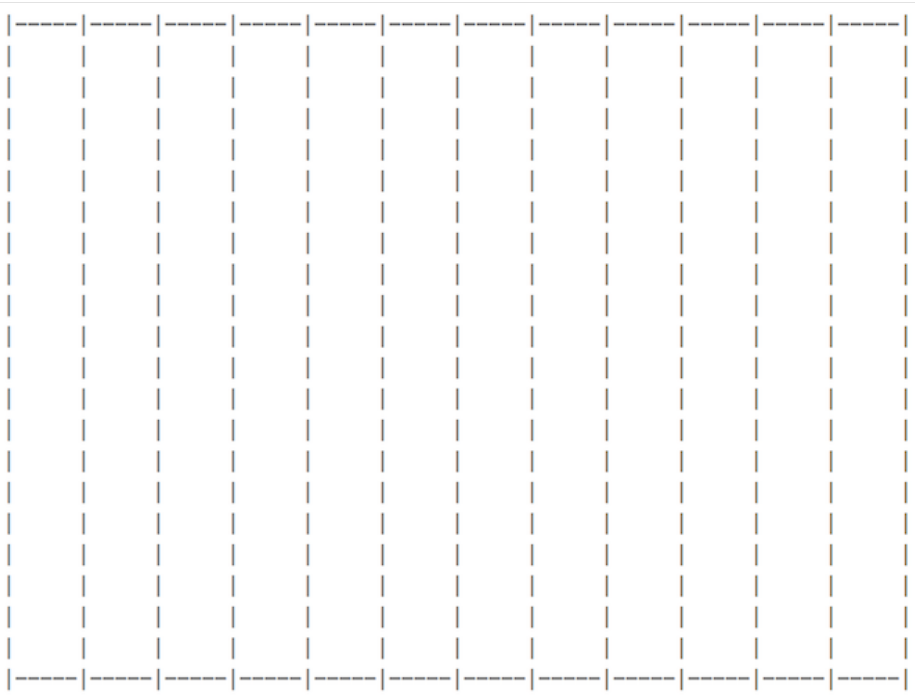
Dr Nick Hayward

CSS grid layout - part 1

intro

- grid designs for page layout, components...
  - *increasingly popular over the last few years*
  - *useful for creating responsive designs*
- quick and easy to layout a scaffolding framework for our structured content
- create boxes for our content
  - *then position them within our grid layout*
- content can be stacked in a horizontal and vertical manner
  - *creating most efficient layout for needs of a given application*
- another benefit of CSS grids is that they are framework and project agnostic
  - *thereby enabling easy transfer from one to another*
- concept is based upon a set number of columns per page with a width of 100%
- columns will increase and decrease relative to the size of the browser window
- also set break points in our styles
  - *helps to customise a layout relative to screen sizes, devices, aspect ratios...*
  - *helps us differentiate between desktop and mobile viewers*

Image - Grid Layout



Grid Layout - Columns and rows

CSS grid layout - part 2

grid.css

- build a grid based upon 12 columns
  - *other options with fewer columns as well*
- tend to keep our grid CSS separate from the rest of the site
  - *maintain a CSS file just for the grid layout*
- helps abstract the layout from the remaining styles
  - *makes it easier to reuse the grid styles with another site or application*
- add a link to this new stylesheet in the `head` element of our pages

```
<link rel="stylesheet" type="text/css" href="assets/styles/grid.css">
```

or

```
<link rel="stylesheet" href="assets/styles/grid.css">
```

- ensure padding and borders are included in total widths and heights for an element
  - *reset `box-sizing` property to include the `border-box`*
  - *resetting box model to ensure padding and borders are included*

```
* {
box-sizing: border-box;
}
```

CSS grid layout - example - part 3

grid.css

- set some widths for our columns, 12 in total
  - *each representing a proportion of the available width of a page*
  - *from a 12th to the full width of the page*

```
.col-1 {width: 8.33%;}
.col-2 {width: 16.66%;}
.col-3 {width: 25%;}
.col-4 {width: 33.33%;}
.col-5 {width: 41.66%;}
.col-6 {width: 50%;}
.col-7 {width: 58.33%;}
.col-8 {width: 66.66%;}
.col-9 {width: 75%;}
.col-10 {width: 83.33%;}
.col-11 {width: 91.66%;}
.col-12 {width: 100%;}
```

- classes allow us to set a column span for a given element
  - *from 1 to 12 in terms of the number of grid columns an element may span*

CSS grid layout - example - part 4

grid.css

- then set some further styling for each abstracted `col-` class

```
[class*="col-"] {
  position: relative;
  float:left;
  padding: 20px;
  border: 1px solid #333;
}
```

- create columns by wrapping our content elements into rows
- each row always needs 12 columns

```
<div class="row">
  <div class="col-6">left column</div>
  <div class="col-6">right column</div>
</div>
```

CSS grid layout - example - part 5

grid.css

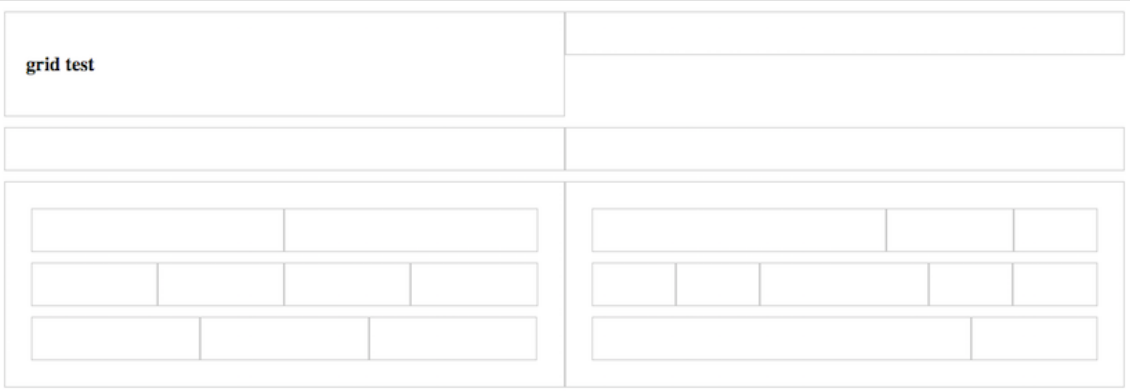
- due to the initial CSS of float left, each column is floated to the left
- columns are interpreted by subsequent elements in the hierarchy as non-existent
  - *initial placement will reflect this design*
- prevent this issue in layout, add the following CSS to grid stylesheet

```
.row:before, .row:after {
  content: "";
  clear: both;
  display: block;
}
```

- benefit of the clearfix, `clear: both`
  - *make row stretch to include columns it contains*
  - *without the need for additional markup*

DEMO - Grid Layout 1 - no gutters

Image - Grid Layout I



Grid Layout - No Gutters

CSS grid layout - example - part 6

grid.css

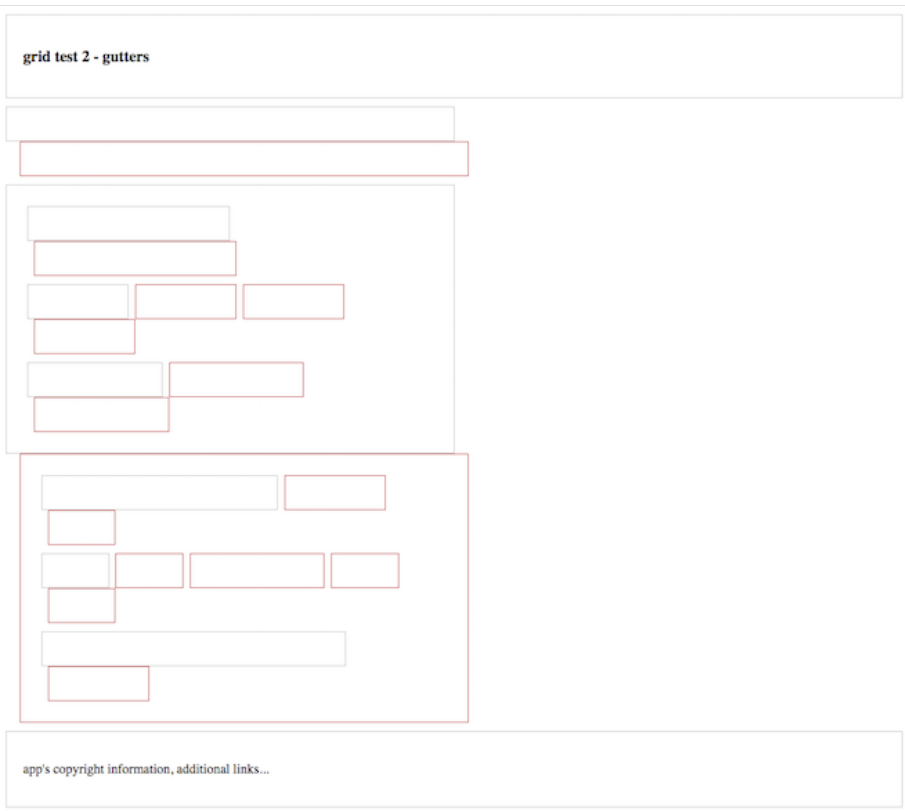
- add gutters to our grid to help create a sense of space and division in the content
- simplest way to add a gutter to the current grid css is to use padding
  - *rows can use padding, for example*

```
.row {
  padding: 5px;
}
```

- issue with simply adding padding to the columns
  - *margins are left in place, next to each other*
  - *column borders next to each with no external column gutter*
- fix this issue by targeting columns that are a sibling to a preceding column
- means we do not need to modify the first column, only subsequent siblings

```
[class*="col-"] + [class*="col-"] {
  margin-left: 1.6%;
}
```

Image - Grid Layout 2



Grid Layout - Gutters Overflow

CSS grid layout - part 7

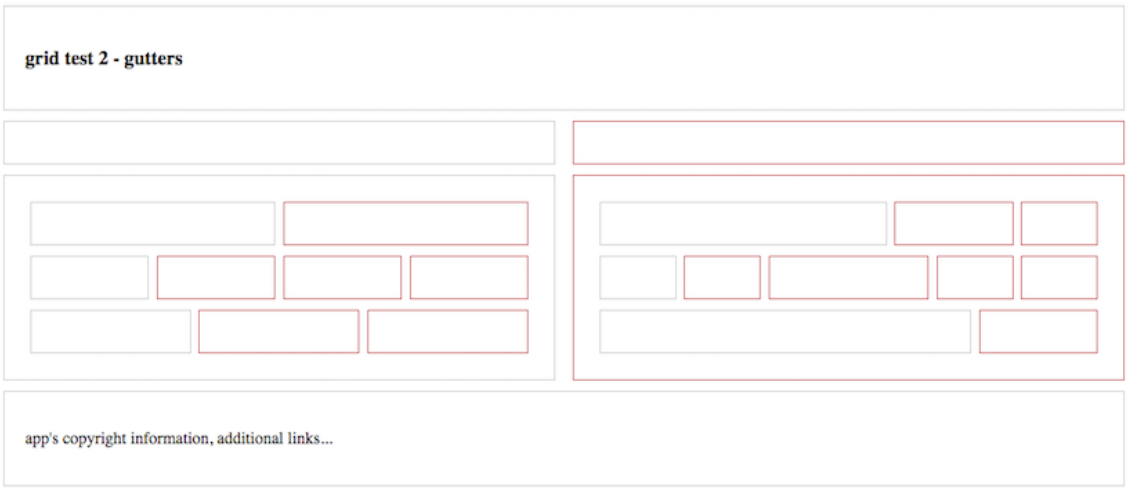
grid.css

- to fix this issue we recalculate permitted % widths for our columns in the CSS
  - *we now have % widths as follows*

```
.col-1 {width: 6.86%;}
.col-2 {width: 15.33%;}
.col-3 {width: 23.8%;}
.col-4 {width: 32.26%;}
.col-5 {width: 40.73%;}
.col-6 {width: 49.2%;}
.col-7 {width: 57.66%;}
.col-8 {width: 66.13%;}
.col-9 {width: 74.6%;}
.col-10 {width: 83.06%;}
.col-11 {width: 91.53%;}
.col-12 {width: 100%;}
```

- DEMO - Grid Layout 2 - gutters

Image - Grid Layout 3



Grid Layout - Gutters

CSS grid layout - part 8

media queries

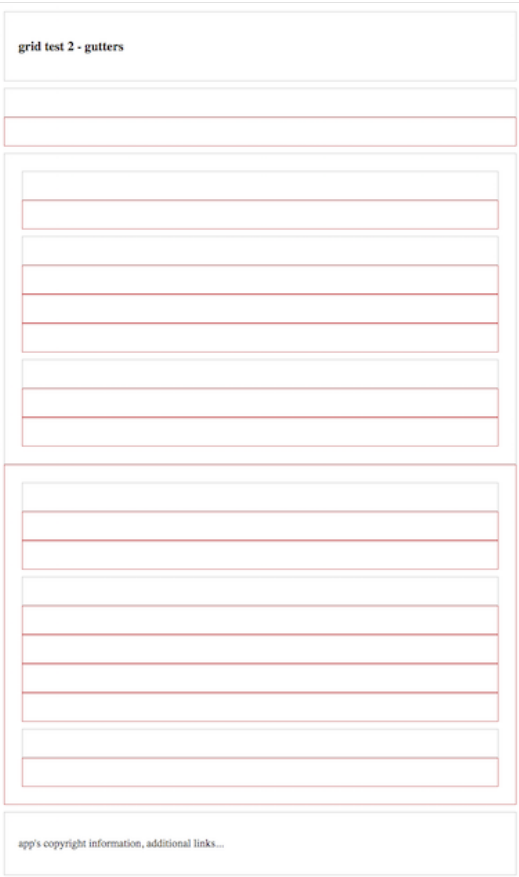
- often need to consider a mobile-first approach
- introduction of CSS3, we can now add **media queries**
- modify specified rulesets relative to a given condition
  - *eg: screen size for a desktop, tablet, and phone device*
- media queries allow us to specify a breakpoint in the width of the viewport
  - *will then trigger a different style for our application*
- could be a simple change in styles
  - *such as colour, font etc*
- could be a modification in the grid layout
  - *effective widths for our columns per screen size etc...*

```
@media only screen and (max-width: 900px) {
  [class*="col-"] {
  width: 100%;
  }
}
```

- gutters need to be removed
  - *specifying widths of 100% for our columns*

```
[class*="col-"] + [class*="col-"] {
  margin-left:0;
}
```

Image - Grid Layout 4



Grid Layout - Media Queries

HTML5, CSS, & JS - example - part 1

add grid layout

- update the layout of our Travel Notes application to include a grid layout
- apply this grid layout to the overall application
  - *organisation and presentation of the notes*
- remove the centred, fixed width for the `body` in our style.css stylesheet
- removes centre styling, results in content spanning full width of browser window
- add the grid layout to help us control this layout

```
<link rel="stylesheet" type="text/css" href="assets/styles/grid.css">
```

- then modify content categories, child elements to use new grid css

```
<!-- document header -->
<header>
  <div class="row">
    <div class="col-5">
      <h3>travel notes</h3>
      <h5>record notes from various places visited...</h5>
    </div>
    <div class="col-7"></div>
  </div>
</header>
```

Image - HTML5, CSS, & JS - grid layout

Grid Layout - Updated Header

HTML5, CSS, & JS - example - part 2

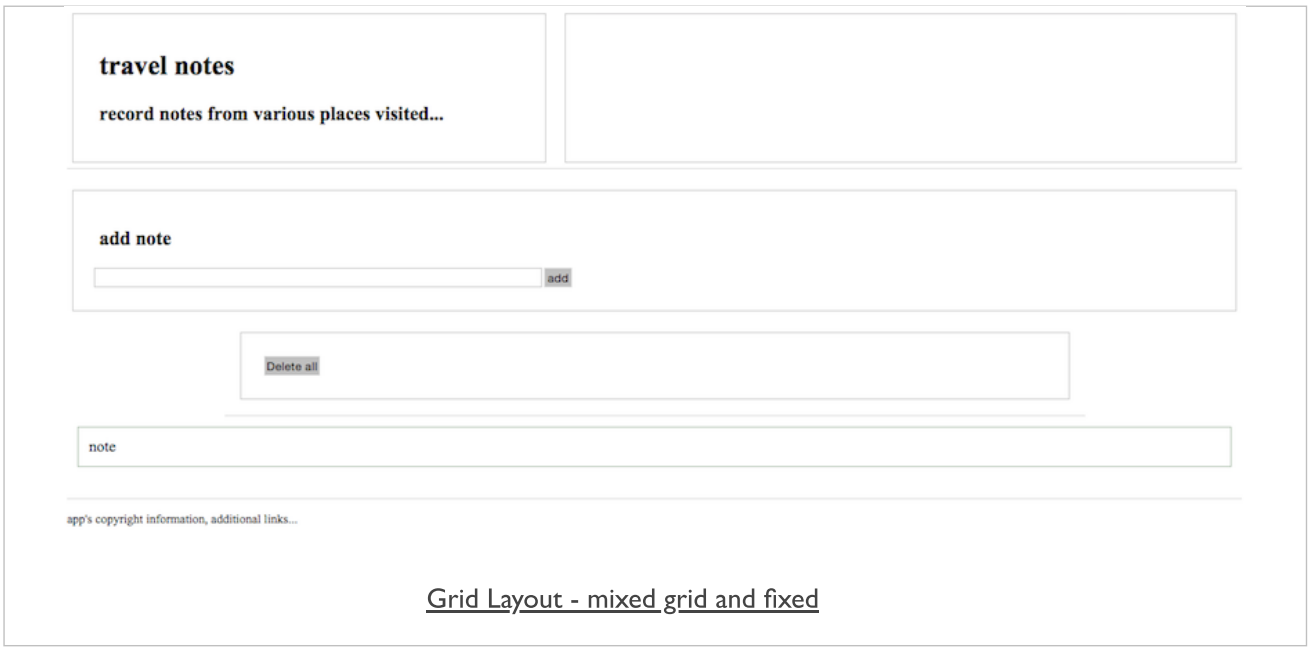
add grid layout

- update our main content to position the `note-input` and `note-controls`

```
<!-- note input -->
<section class="note-input">
  <div class="row">
    <div class="col-12">
      <h5>add note</h5>
      <input><button>add</button>
    </div>
  <div>
</section>
<!-- note controls for delete... -->
<section class="note-controls">
  <div class="row">
    <div class="col-12">
      <button id="notes-delete">Delete all</button>
    </div>
  </div>
</section>
```

- still need to amend style.css to remove additional fixed styling

Image - HTML5, CSS, & JS - grid layout 2



Grid Layout - mixed grid and fixed

HTML5, CSS, & JS - example - part 3

add grid layout

- fix mixed rendering by removing width, margin, and padding for `.note-controls`

```
/* note controls */
.note-controls {
  border-bottom: 1px solid #dedede;
  display: none;
}
```

- continue to update Travel Notes app
  - *modify output for notes*
  - *add further options for users*

DEMO - Travel Notes - grid layout with media queries

HTML5, CSS, & JS - example - part 4

add flex to grid layout

- an additional option to consider - flex layouts
  - *a recent W3 working draft*
  - *aims to provide efficient way to align and proportion content*
- known as **Flexbox Layout**
  - *idea is to apportion width and height for content*
  - *proportions relative to container even when their size is unknown or dynamic*
- flex layout could, in theory, replace a full grid layout
  - *considered more a complement to overall grid structure*
- defined flex container expands items to fill the container's available space
  - *can also shrink them to prevent any possible overflow*
- think of a flex layout as supporting multiple directions
  - *direction agnostic*
- many properties available for **flex**
  - *focus upon styling flex container and any flex items*

HTML5, CSS, & JS - example - part 5

add flex to grid layout

- we might specify CSS properties for a flex container

```
.flex-container {
display: flex; /* defines container as flex */
flex-direction: row; /* defines positioning of items added to container */
flex-wrap: wrap; /* defines whether to wrap items to another line */
justify-content: flex-start; /* defines start point and distribution of items */
}
```

- allows us to position our container starting at the left
  - *items contained in a row*
  - *contained items wrapping to additional lines if necessary*
- many additional options available for each property
- also add rulesets for specific styling of items within a flex container
- we could add properties to a flex item such as
  - *specify the order of the flex items*
  - *whether a particular item can grow or shrink relative to content*
  - *default size of an item before any remaining space is distributed*
  - *individual alignment for a given item...*

HTML5, CSS, & JS - example - part 6

add flex to notes

- flex container and items useful for organising and positioning our notes
- due to uncertainty about content, size, and general note requirements
  - *flex positioning and styling removes the need for assumptions or fixed sizes*
- we can start to modify the styling and rendering of our notes using flex

```
/* flex item */
.flex-item {
  flex-basis: 300px; /* default size before extra */
  flex-grow: 1; /* all items will be equal */
}
```

- gives us a default smallest size for each note
- then the ability for each note to grow to fill the row as required
- also work with responsive layouts
  - *due to the minimum size and the option to grow for each item*
  - *and wrap flex items per flex container*
- modify and update styles as we develop travel notes app

DEMO - Travel Notes - grid layout with flex notes

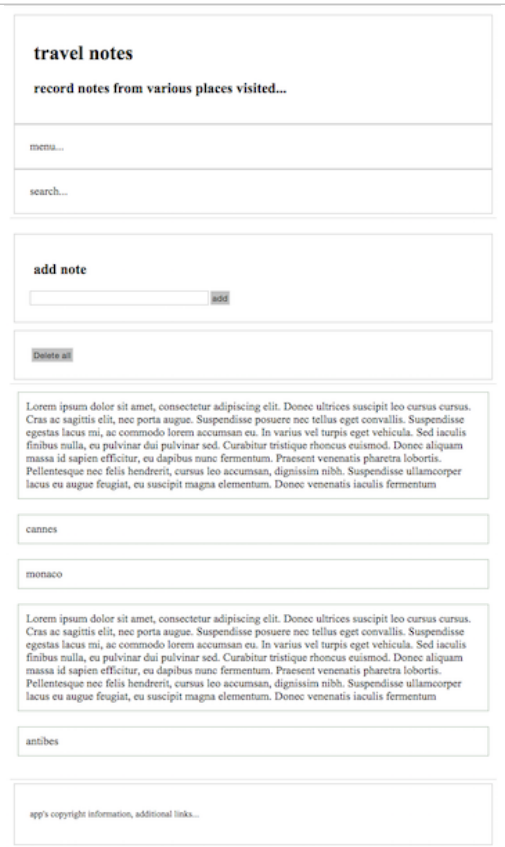
Image - HTML5, CSS, & JS - Flex Notes

Grid Layout - flex notes

Image - HTML5, CSS, & JS - Flex Notes 2

Grid Layout - flex notes - medium

Image - HTML5, CSS, & JS - Flex Notes 3



Grid Layout - flex notes - small

add flex to notes

Notes with Flex and Media Queries

HTML5, CSS, & JS - example - part 1

add AJAX and JSON - load notes from json

- update our **travel notes** app to allow us to load some test persistent notes from a local JSON file
- initial JSON is as follows

```
{
  "travelNotes": [{
    "created": "2015-10-12T00:00:00Z",
    "note": "a note from Cannes..."
  }, {
    "created": "2015-10-13T00:00:00Z",
    "note": "a holiday note from Nice..."
  }, {
    "created": "2015-10-14T00:00:00Z",
    "note": "an autumn note from Antibes..."
  }]
}
```

HTML5, CSS, & JS - example - part 2

add AJAX and JSON - load notes from json

- add option to load notes from JSON as app initially loads
  - *use deferred promise pattern*
  - *checks source JSON as it loads via the promise*
  - *then outputs the end result*
- start with the following update

```
//get the notes JSON
function getNotes() {
  //.get returns an object derived from a Deferred object - do not need explicit deferred object
  var $deferredNotesRequest = $.getJSON (
    "docs/json/notes.json",
    {format: "json"}
  );
  return $deferredNotesRequest;
}
```

HTML5, CSS, & JS - example - part 3

add AJAX and JSON - load notes from json

- help us better manage logic of our notes from app's loading and execution
  - *create two separate JS files*
- our updated structure might be as follows

```
...
|- assets
   |- scripts
      |- travel.js
      |- notes.js
...
```

- we can extend this further, as needed by app features and data

HTML5, CSS, & JS - example - part 4

add AJAX and JSON - load notes from json

- add our `.when()` function to the app's loader
  - *`.when()` function accepts a deferred object*
  - *in our case a limited promise*
- then allows us to chain additional deferred functions
  - *including required `.done()` function*
- for returned data, use standard response object to get `travelNotes`
  - *then iterate over the array for each property*
  - *for each iteration, we can simply call our `createNote` function*
  - *builds and renders required notes to the app's DOM*

```
//use deferred object from getJson
$.when(getNotes()).done(function(response) {
  //get travelNotes object
  var $travelNotes = response.travelNotes
  //process travelNotes array
  $travelNotes.forEach(function(item) {
    //check each property
    if (item !== null) {
      //get note
      var note = item.note;
      //create each note for rendering
      createNote(note);
    }
  });//end foreach
});
```

HTML5, CSS, & JS - example - part 5

add AJAX and JSON - load notes from json

- simple problem - existing `createNote()` function does not accept a parameter
- need to update the logic of that function to accept and handle a parameter
- also requires a quick update to any functions and calls to the `createNote()`
  - *event handlers for creating a new note using the `add` button and keypress within the input field*

```
//manage input field and new note output
function createNote(data) {
  ...
  //conditional check for data
  if (data !== "") {
  //set content for note
  $note.html(data);
  ...
  }
}
```

HTML5, CSS, & JS - example - part 6

add AJAX and JSON - load notes from json

- update our event handlers for the note input button and input field keypress as follows,

```
//handle user event for `add` button click
$(".note-input button").on("click", function(e) {
  var $note_data = getNoteInput();
  //call note builder function
  createNote($note_data);
});
```

```
//handle user event for keyboard press
$(".note-input input").on("keypress", function(e) {
  //check code for keyboard press
  if (e.keyCode === 13) {
    var $note_data = getNoteInput();
    //call note builder function
    createNote($note_data);
  }
});
```

- our notes now load by default as the app starts
- note input button and keypress work as expected
- DEMO - travel notes & JSON

Working with APIs - part 1

remote api options - Flickr

- **Travel Notes** app loads data from a local JSON file
- add option to load different types of data using remote APIs
  - *Flickr API for images, tags...*
- basics and principles are similar to the patterns we've already seen and tested
- test a sample JSON return from the Flickr API
- JSON return - useful properties for app
  - *title*
  - *link*
  - *media (direct url for image - where available)*
  - *description*
  - ...
- public feed for searching public photos, videos, groups, recent activity...
- Flickr API Public Feed - Cannes and France

Working with APIs - part 2

working with Flickr API

- query Flickr's public feed for photos
  - *we can use our now familiar pattern for requesting JSON*

```
//get the Flickr public feed JSON for images
function getImages() {
  //.get returns an object derived from a Deferred object - do not need explicit deferred object
  var $deferredNotesRequest = $.getJSON (
  "http://api.flickr.com/services/feeds/photos_public.gne?jsoncallback=?",
    { tags: "cannes,france,boules",
      tagmode: "all",
      format: "json"
    });
  return $deferredNotesRequest;
}
```

- need to make a few specific modifications to the request
  - *JSONP to avoid browser security restrictions*

Working with APIs - part 3

working with Flickr API

- Flickr's public feed includes options
  - *eg: a specific user ID for photos, various tags, how tags are interpreted by the search...*
- use our `.when()` function to load and render some test images from Flickr

```
$.when(getImages()).done(function(response) {
  console.log("done..."+response);
  //use jQuery's generic iterative function for the response...
  $.each( response.items, function( i, item ) {
    buildImage(item.media.m);
    //limit test images to 8
    if ( i === 7 ) {
      return false;
    }
  });
});
```

- DEMO - AJAX and JSON - Flickr api

HTML5, CSS, & JS - example - part 7

working with Flickr API - update travel notes

- add option to Travel Notes app to allow a user to view images from Flickr
- need to update app's HTML, CSS, and JS
- modify how our notes, and associated options, are rendered to our users
- add a search option for photos on Flickr
- render our images to match the notes
- app's structure still reflects three primary content categories
  - *`header`, `main`, and `footer` with slight modifications to the `main` category*
- `main` content category updated to create two distinct rows for initial content
  - *contain defined semantic containers*
- row containing `.note-input` and Flickr search option `.contextual-choice`
  - *then split this row into two columns of 6*

HTML5, CSS, & JS - example - part 8

working with Flickr API - update travel notes HTML

- updated HTML for `.note-input` and Flickr search `.contextual-choice`

```
<div class="row">
  <!-- note input -->
  <section class="note-input col-6">
    <h5>add note</h5>
    <input><button>add</button>
  </section>
  <!-- contextual choice -->
  <section class="contextual-choice col-6">
    <h5>search flickr</h5>
    <input><button>search</button>
  </section>
</div>
```

HTML5, CSS, & JS - example - part 9

working with Flickr API - update travel notes HTML

- update the HTML for rendering the images
  - *add alongside our notes*
- create another row for these containers
  - *add two section containers for `.note-output` and `.contextual-output`*
- make `.note-output` slightly larger to show primary app focus

```
<div class="row">
  <!-- note output -->
  <section class="note-output col-7 flex-container">
  </section>
  <!-- contextual output -->
  <section class="contextual-output col-5 flex-container">
  </section>
</div>
```

HTML5, CSS, & JS - example - part 10

working with Flickr API - update travel notes JS

- add further functionality to **Travel Notes** app
- split our JS logic into three files to help with oranisation
  - *a main loader file, `travel.js`,*
  - *and a file each for notes and contextual options*
- updated app structure for JS

```
...

|- assets
   |- scripts
      |- contextual.js
      |- notes.js
      |- travel.js
...
```

- underlying logic for the notes will remain the same
  - *move loading of default notes to the `travel.js` main loader file*
- updates for searching, returning, and rendering images from Flickr
  - *added to the `contextual.js` file*

HTML5, CSS, & JS - example - part 11

working with Flickr API - update travel notes JS

- test Flickr API in our app using some set data for image tags
  - *respond to the user clicking on the search button*
  - *submit our query to Flickr*
  - *process the returned JSON for the images*
  - *render them for viewing*
- request and process our images using the familiar pattern

```
//get the Flickr public feed JSON for images
function getImages(data) {
  var img_tags = data;
  //.get returns an object derived from a Deferred object - do not need explicit deferred object
  var $deferredNotesRequest = $.getJSON (
    "http://api.flickr.com/services/feeds/photos_public.gne?jsoncallback=?",
    { tags: img_tags,
      tagmode: "all",
      format: "json"
    });
  return $deferredNotesRequest;
}
```

HTML5, CSS, & JS - example - part 12

working with Flickr API - update travel notes JS

- returned data using standard deferred promise object
  - *add a new function to handle the processing of the images*

```
function processImages(data) {
  $.when(getImages($img_data)).done(function(response) {
    //use jQuery's generic iterative function for the response...
    $.each( response.items, function( i, item ) {
      createImage(item.media.m);
      //limit test images to 4
      if ( i === 3 ) {
        return false;
      }
    });
  });
}
```

- using deferred promise object with `.when()` function chained to `.done()` function
- add jQuery's generic iterative function to help us process the response
  - *instead of standard JavaScript `.forEach()` option*
- loop through each value, and pass the image to our new function, `createImage()`
  - *ready for rendering to our app's DOM*
  - *limit number of images for testing*

HTML5, CSS, & JS - example - part 13

working with Flickr API - update travel notes JS

```
//manage new image output
function createImage(data) {
  //create each image element
  var img = $('<img class="flex-img">');
  //add image
  img.attr("src", data);
  //append to DOM
  $(".contextual-output").append(img);
}
```

- `.createImage()` function accepts a parameter for image data
- then process ready for rendering to the app's DOM
- image is added to a new `img` element with a new class of `.flex-img`
  - *creates a flex item for rendering*
- added to the new `.contextual-output` section
- rendered images displayed as thumbnails for the user
  - *complementary to the existing notes*

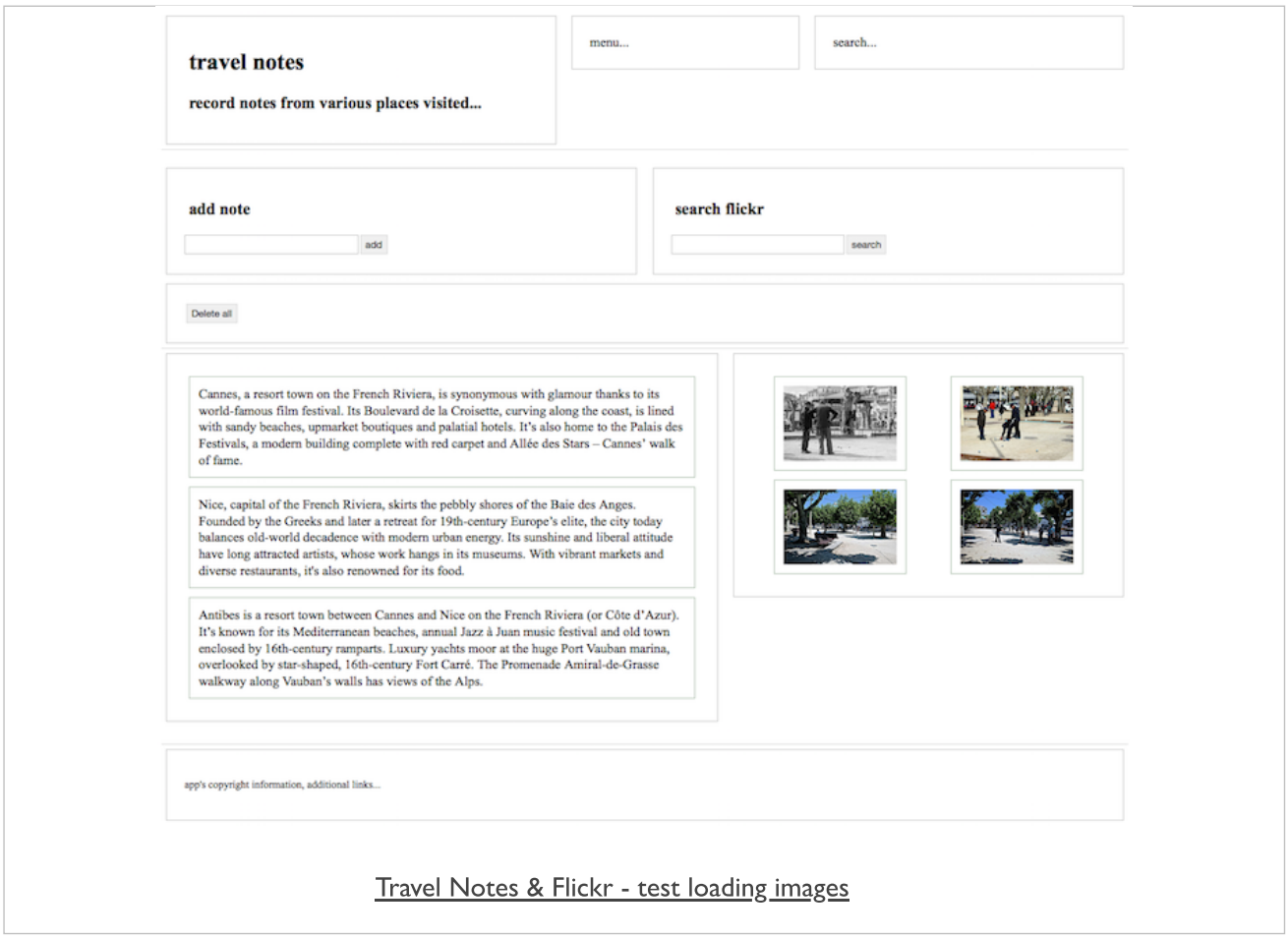
HTML5, CSS, & JS - example - part 14

working with Flickr API - update travel notes JS

- to add images to the app
  - *a user can enter their requested tags in the search field*
  - *then click on the `search` button to return any available images*
- event handler for this `search` button click uses the requested tags
  - *passes them as a parameter to the `processImages()` function*

```
//handle user event for image `search` button click
$(".contextual-choice button").on("click", function(e) {
  //test tags for testing image search
  $img_data = "cannes,france,boules"
  //process images
  processImages($img_data);
});
```

Image - HTML5, CSS, & JS - Travel Notes & Flickr



Travel Notes & Flickr - test loading images

HTML5, CSS, & JS - example - part 15

working with Flickr API - update travel notes CSS

- need to update and modify existing CSS
  - *helps with correct rendering of the thumbnail images*
- CSS additions are initially modest
  - *reflects integration with existing app, grid, and flex layouts*
- add new ruleset for image rendering in the `.contextual-output` section

```
/* contextual output images */
.contextual-output img {
  margin: 5px;
  padding: 5px;
  border: 1px solid #b1c4b1;
}
```

- update `.flex-container` class to change `justify-content` property to value of `space-around`
- add new ruleset for a `.flex-img` class.

```
/* flex image */
.flex-img {
  flex-basis: 150px;
  flex-grow:0;
}
```

- specify size of a thumbnail image
  - *initially restrict their ability to grow relative to flex*

HTML5, CSS, & JS - example - part 16

working with Flickr API - update travel notes JS

- we can now request, process, and render images from Flickr to Travel Notes app
  - *still need to accept and process search queries from search input field.*
- add option to check search input field
  - *then submit query to Flickr for images*

```
//get input value for image search
function getImageInput() {
  //define img value
  var img_val = "";
  //define input field
  var $img_tags = $(".contextual-choice input");
  if ($img_tags.val() !== "") {
    img_val = $img_tags.val();
    return img_val;
  } else {
    return img_val;
  }
}
```

HTML5, CSS, & JS - example - part 17

working with Flickr API - update travel notes JS

- use `getImageInput()` function with a modified `processImages()` function

```
//process image production, loading, and pass to rendering
function processImages() {
  //check img visibility for contextual-output - clear existing images
  if (checkVisible($(".contextual-output img")) === false) {
    //empty existing images
    $(".contextual-output").empty();
  }
  //get data from image search input field
  var $img_data = getImageInput();
  //use image data to get images, and pass for rendering
  $.when(getImages($img_data)).done(function(response) {
    console.log("done..."+response);
    //use jQuery's generic iterative function for the response...
    $.each( response.items, function( i, item ) {
      createImage(item.media.m);
      //limit test images to 4
      if ( i === 3 ) {
        return false;
      }
    });
  });
}
```

HTML5, CSS, & JS - example - part 18

working with Flickr API - update travel notes JS

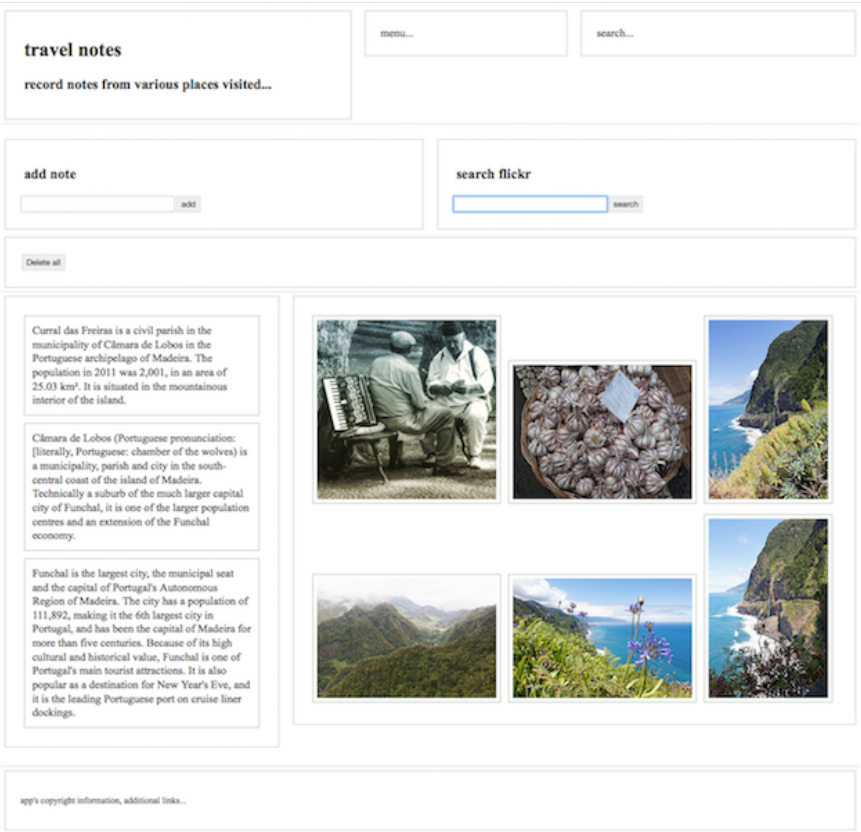
- updated `processImages()` function then called within event handlers
  - *for the search button and a keypress in the search input field*

```
//handle user event for image search button click
$(".contextual-choice button").on("click", function(e) {
  //process images
  processImages();
});

//handle user event for keyboard press
$(".contextual-choice input").on("keypress", function(e) {
  //check code for keyboard press
  if (e.keyCode === 13) {
    //process images
    processImages();
  }
});
```

- DEMO - travel notes & Flickr

Image - HTML5, CSS, & JS - Travel Notes & Flickr



Travel Notes & Flickr - different layout

Demos

Travel notes app - series 3

- DEMO 1 - Travel notes - grid layout with media queries
- DEMO 2 - Travel notes - demo2

Travel notes app - series 4

- DEMO 1 - Travel Notes & JSON
- DEMO 2 - Travel Notes & Flickr
- DEMO 3 - Travel Notes & Flickr - error checking

Grids

- Grids 1 - Grid 1 with no gutters
- Grids 2 - Grid 2 with gutters

References

- Flickr API
  - *Public feeds*
  - *Public feed - public photos & video*
- jQuery
  - *jQuery*
  - *jQuery API*
  - *jQuery - deferred*
  - *jQuery - .getJSON()*
  - *jQuery - JSONP*
  - *jQuery - promise*
- MDN
  - *MDN - JS - Iterators and Generators*
  - *MDN - JS Objects*
  - *MDN - JS - Using Promises*
- W3
  - *W3 - CSS Flexible Box Layout Module 1*
  - *W3 - JS Object*
  - *W3 - JS Performance*