* Pages: Static vs Dynamic (google news is dynamic coz it keeps updating).
* Responsive site means responsive to change in size of browser.
* % makes more responsive as compared to pixels.
* HTML, CSS, JavaScript of any site is accessible.
* <http://codepen.io/>
* Look for errors in the Inspect -> Console.
* For setting background image, refer to Purrfect Match in Bootstrap. We have to set html’s height as 100% so that body takes up the entire screen. Also background-size: cover and background-position: center.
* Padding and margin when used as a percentage, is always based on the width of the parent element.
* If you want your bootstrap styled website to be responsive on mobile then be sure to add the following meta tag to your <head> element, above the <title> tag: <meta name="viewport" content="width=device-width, initial-scale=1">

**VS Code Shortcuts**

* Lorem20 will give 20 lorem ipsum words.
* div.container.text-center will make a div with classes container and text-center.
* Alt+Click for multiple cursors.
* Ctrl+d for copy pasting the current line or the selected lines.

**HTML**

* MDN HTML elements reference to find all the elements that can be used like body, h1, p..
* Title is what we see on google search and when opened, on the tab name.
* Headings, paragraphs are block-level elements. Not inline (unlike bold). So each heading (h1, h2,..) or paragraph take a separate line.
* Em > italic (emphasis is meaningful)
* Strong > bold (strong is meaningful)
* Divs and Spans are generic containers (For Styling)
  + Divs groups things together (for example, group images, texts and have a box around them)
  + But Divs are block level unlike Spans. So span groups inline.
* Attributes – Additional information for HTML elements. MDN HTML Attributes reference for details.
* Anchor tag is inline. Also, when mentioning a site, mention explicitly (including protocol (HTTP)). For example, <http://ww.google.com> instead of [www.google.com](http://www.google.com). Explicit is required because otherwise it would consider it a relative path. Like another html file in the same folder with have a common preceding path.
* Imgur to store images online which can be used in HTML file.
* Tables, use <thead>..</thead> and <tbody>..</tbody> to separate headings and rows of a table so that search engine understands. It doesn’t change anything visually. <tr>..</tr> denotes a row and <th>..</th> represents table heading. Table data <td>..</td> is for columns in a row.
* Form tag is like a container in which we put all the different types of inputs like radio button, drop-downs, check boxes, text boxes.
  + Method means what HTTP method (get/post) and action means where the form sends the data to.
  + Get method is by-default and if action isn’t specified then it refreshes the current \*.html file.
  + Get method (Google) doesn’t store in database, it just searches and gets data back.
  + Post method is like signup page of Facebook which gets stored in Facebook’s database.
  + Type attribute of Input element decides the type of input (MDN input).
  + Labels are important for visually impaired people. It tells what each part of the form corresponds to. Input can be nested inside label or can be separate but have same value of for in label and id in input.
  + Validation- required (blank not allowed) and data type (correct format).
    - Minlength and maxlength attributes to use for the input validation has less browser support than the pattern attribute.
    - Input type = email ensures that email if entered is in correct format.
    - <https://validator.w3.org/> (Markup validation service)
  + For radio input, we can use the same value for the name attribute in the input element so that we select only one of the radio buttons as a choice.
    - Value of value attribute is reflected in the URL on submitting. As name’s value=value’s value (radio) or the input text in the textbox.
* Textarea element is different from input type text. It lets us choose size of textbox.
* Dropdown, selected disabled will act like a placeholder.

**CSS**

* It’s better to separate HTML from CSS. Hence inline styling is a bad idea.
* If there are conflicting styles for the same element (suppose h1), then the one used later will be considered.
* Colors are
  + 6 digits hexadecimal (RGB). Use [www.colorpicker.com](http://www.colorpicker.com)
  + Or represented as rgb(rval, gval, bval) where each ranges from 0 to 255.
  + Rgba(rval, gval, bval, alpha) where alpha (0.0-1.0) for transparency.
* Background can be a color or image (set background-repeat to no if you don’t want the image to be tiled, and set background-size to cover if you want to stretch the image the entire screen).
* Border: width, style, color.
  + border: 5px solid purple;
* CSS Selectors: Element, #id and .class.
  + Same id values can be used only once a page.
  + Class values can be same in a page.
  + Descendant selectors- Select anchor tags inside <li>..</li> only.

li a{

    color: red;

}

* + If we want to select both h1 and h3, we can use a comma in between.
  + Adjacent Selector: Allows us to select siblings. Here, all uls that come after h4 but in the same level, not inside.

h4 + ul{

    border: 4px solid red;

}

* Attribute Selector: All inputs of type text suppose. Or all anchors with link to google.com.
  + - * input[type=”text”]{
        + background: blue;
      * }
* Select nth of type selects suppose every 3rd li. Or select every even number.
* Inheritance and Specificity
  + Inheritance: If a body is set color pink, all elements within it will be pink.
  + Specificity: If a body is set pink, but paragraph within it is set blue, the paragraph wins as it’s more specific. Example, ***Id>class> li>ul>body***
  + Calculator*:* [*https://specificity.keegan.st/*](https://specificity.keegan.st/)
  + Type Selectors < Class, Attribute, Pseudo-Class Selector < ID Selector.
* Fonts availability In the different systems: <https://www.cssfontstack.com/>
  + Font-family, font-size, font-weight, line-height, text-align, text-decoration
  + Put font-family inside “” if the name consists of number.
  + Font-size either in px, em, rem.
    - Size of em value is for relative hence dynamic.
    - Em set 2.0 means twice the size of parent element.
    - If we set body font-size as 40 px, then we can relatively specify the size of h1, p, span (nested inside) and so on by using em.
    - Rem is not decided by parent element unlike em. They are decided by root element on the page.
    - If font-size of the root element is 16px then 1 rem = 16px for all elements. If font-size is not explicitly defined in root element then 1rem will be equal to the default font-size provided by the browser (usually 16px).
  + Font-weight can also increase page load.
  + Google fonts: <https://fonts.google.com/?sort=date>
* Box Model!!: Padding, Margin, Border, and More!
  + Every element has a rectangular box around it.
  + Spacing between elements achieved by margin.
  + Padding: Space between border and element inside it. Can choose between top, left, bottom and right like border.
  + Width: 50% means 50% of parent element. Like p inside body.
  + Blue=content, green=padding, orange=margin
  + Margin: top right bottom left. Or can put two values only which means top-bottom and right-left. Auto value is also acceptable.
* Class=”vertical middle” means two classes separated by space.
* ***FLOAT***
  + Since each div is block level, we can use ***float*** left which would consider no gravity means divs will be pushed up and then left.
  + Float also removes the whitespaces between images for suppose. These whitespaces are not through padding or margin. They are default.
* There should be only one h1 in the page. It should be the biggest thing. Otherwise use h2 or its descendants.
* This command sets width as 700px when possible otherwise 80%.

max-width: 700px;

width: 80%;

* Different hr styles: <https://css-tricks.com/examples/hrs/>

**Bootstrap-3**

* <https://getbootstrap.com/docs/3.3/> for version 3.3.7.
* Helps make good looking *responsive* websites very quickly.
* Navbars (converts to hamburger), Grid System, forms, buttons, jumbotrons,…
* Bootstrap.min.css will load faster than bootstrap.css but both are same.
* Bootstrap.css is more readable than bootstrap.min.css.
* Bootstrap CDN: Link to hosted version of same bootstrap file we downloaded.
* We can always overwrite the bootstrap styles. So add custom styles link after bootstrap link.
* Container class gives nice padding and margin in bootstrap. We can have separate elements with class container, like inside navbar and outside jumbotron.
* Don’t but navbar inside container. Only its contents.
* Bootstrap’s javascript requires jQuery. Hence jQuery link before bootstrap’s javascript CDN.
* Grid System is number one reason why people use Bootstrap!!!
  + Every container in bootstrap can be divided into 12 different columns. Then we can decide how many of 12 units each element can take up.
  + As we resize browser, the units each element takes adjusts.
  + We can nest grids inside grids to further divide it into 12 units.
  + 4 layouts (sizes) available: xs (mobile), sm (tablet), md, lg (Details: <https://getbootstrap.com/docs/3.3/css/#grid> ).
  + If you want each element take 1 unit when large size, and 2 units when medium size:

<div class="col-lg-1 col-md-2">Responsive</div>

* If we set 2 for sm, it will automatically set 2 for md, lg as well. Similarly, for md, it will set up same value for lg. But we can always explicitly set values for larger sizes.
* Nesting syntax:

<div class="col-md-3 col-sm-6 col-xsm-12 pink">

<div class="row">

<div class="col-lg-6 orange">First Half</div>

<div class="col-lg-6 orange">Other Half</div>

</div>

</div>

* **NOTES varying size images**:
  + If you don't mind the images being of varying heights, but want to get rid of the white space bug then see [here](https://www.udemy.com/the-web-developer-bootcamp/learn/v4/questions/1989586).
  + If you're looking for Pinterest-like presentation of thumbnails of varying heights and/or widths, you'll need to use a third-party plugin such as [Masonry](https://masonry.desandro.com/), [Isotope](https://isotope.metafizzy.co/), or [Salvattore](https://salvattore.js.org/).
  + If you want to crop all of the images to the same size then see an example of this [here](https://codepen.io/nax3t/pen/MJwpdb).
* [www.unsplash.com](www.unsplash.com%20) provides a bunch of high-resolution pictures.
* Thumbnail class in bootstrap when our image is too large for a column so it would scale it down. Also, it gives a nice border to our image.
* When navbar is fixed to top, we will have to have the padding for body. Basically, the goal is to push the contents of the body down so it doesn't fall behind the fixed navbar, and we use padding to achieve that.
* Put bootstrap.css inside the css folder and css folder adjacent to font folder and html file so that fonts load successfully on html page.
* Fonts: Bootstrap Glyphicon or <https://fontawesome.com/icons?d=gallery&m=free>

**Bootstrap-4**

* Summary under migration tab in the documentation will detail all the changes.
* Bootstrap 4 supports flexbox! So no more floats!
* Global font-size changed to 16px (used for rem).
* 5 screen sizes available. Extra large is the new one.
* Stable release means don’t expect any breaking changes so you won’t have to rewrite much code. Alpha release means very buggy and liable to change. Beta means a little more polished.
* Bootstrap 4 Typography: Change font-size of root in html. As Bootstrap 4 is based on rem unit of measurement.
* Bootstrap Utilities- Spacing (margin and padding), borders and a lot more! $spacer is 1 rem by default. Can be changed.
* Responsive breakpoints!
  + {property}{sides}-{breakpoint}-{size} for sm, md, lg, xl.
  + {property}{sides}-{size} for xs.
* p-md-5 means apply padding 5 for size medium and **up**. Similar for other sizes.
* Here, padding left 5 at sm size will make padding left 5 for md, lg, xl also. Padding top 5 at md will make padding top 5 for lg and xl also. And so on.. Hence at xl size, we have padding of 5 on all sides.

<button class="btn btn-success p-0 pl-sm-5 pt-md-5 pr-lg-5 pb-xl-5">Button!</button>

* Navbars: We can change when the navbar should collapse and we can can background colors easily using style.
* Display Utility!! We can set an element as block-level, inline, hide and so on.
  + .d-{value} for xs.
  + .d-{breakpoint}-{value} for sm, md, lg, xl.
* **Flexbox!!**
  + Flex direction or main axis: Left to right. (.justify-content-start is default)
  + Cross axis: Top to bottom. (.align-items-stretch is default so they take stretch to the height of longest)
  + These are also responsive. Example, .justify-content-lg-end.
  + We can make left to right as right to left for main axis. As similarly change for cross axis.
  + We can make cross axis as main axis by .flex-column.
  + We can also move each item inside a flex individually by overwriting the default or class common to all items. For example .align-self-start.
  + With nav class, we have the flexbox enabled.
  + **Grid** is also built with flexbox. If not specified the elements will take equal space in 12 or remaining units.
* Pattern Project-
* **Section element** instead of div and **lead class** inside paragraph to make the text stand out and give extra space.
* Use anchor tag with btn class instead of button when not a form as more suitable.
* In bootstrap the classes col-lg-someNumber for example, has some specific padding set to it by default.
* Background image centered horizontally and vertically, cover and no-repeat. The forward slash is required syntax for separating background position values and background size values. This is to eliminate misinterpretation of the values specified. If you're going to specify both, then background-position has to come first, then the forward slash, then the background-size.

    background: url("imgs/header.jpeg") center center / cover no-repeat;

* btn-outline-danger also changes color of text and font inside it to danger color.
* Museum of Candy Project:
  + Aria-label to make the hamburger meaningful. Nothing changes functionally.
  + In Bootstrap **images are made responsive** with .img-fluid which makes max-width: 100%, height: auto so that the image scales with the parent element.
  + Media query to selectively apply styles based on different screen size.
  + Change order of contents inside the flexbox based on screen size using Bootstrap!