UDACITY

PROJECT SPECIFICATION

Build a Portfolio Site

Design

CRITERIA	MEETS SPECIFICATIONS
Required Elements	The page at minimum includes all of the following: • at least 4 images • title text (h1, h2, etc.) • regular (paragraph) text • a logo.
Semantic HTML	HTML5 semantic tags such as `<header>`, `<footer>`, `<article>`, `<section>` etc. are used to add meaning to the code. No `<div>` or `<section>` tags are without a `CSS` class or id. Check out the W3C documentation on HTML Structural Elements to learn more!

CRITERIA	MEETS SPECIFICATIONS
Custom Design	Provide at least one of the following customizations: • Customize images and text. • Customize placement of the elements on the page (grid layout) with `HTML`, `CSS` or both. • Customize `CSS` styles applied at minimum to paragraph and heading elements.
Grid-Based Layout	Page utilizes a grid-based layout with styles making use of the `flexbox` layout or a framework like `Bootstrap`, `Foundation`, etc. If you're using `Bootstrap`: the rows and columns of the grid must be wrapped in an element with a `container` class.

Responsiveness

CRITERIA	MEETS SPECIFICATIONS

CRITERIA	MEETS SPECIFICATIONS
Cross-Device Compatibility	All content is responsive and displays on all display sizes. This includes: • Desktop • Mobile: Google Nexus 5 • Tablet: Apple iPad An image's associated title and text renders next to the image in all viewport sizes. *TIP*: Test responsiveness with Chrome Developer Tools device emulation by right-clicking anywhere on page, selecting 'Inspect Element', clicking the rectangle to the left of the Elements tab, select Apple iPad or Google Nexus 5 from Device drop-down list, and click reload.
Provide All Content	All content is rendered on all three devices. No content should be hidden on mobile devices.
Viewport meta Tag	Viewport `meta` tag is included in `HTML`. (i.e. `<meta name="viewport" …`)
Responsive Images	If a CSS framework is used, classes provided by the CSS framework are used to make images responsive, otherwise media-queries are used to ensure responsiveness of images.

Separation of Concerns

CRITERIA	MEETS SPECIFICATIONS
Styles Separated From HTML	Portfolio completely separates structure (`HTML`) from design/style (`CSS`). There are no `style` attributes present in the body of the `HTML` document. There are no `<style>` elements in the document. *Note*: It is acceptable to include `height` and `width` attributes in `<img>` elements.
File structure	Files are organized with a directory structure that separates files based on functionality. For example: `css/` for stylesheets `img/` for images `js/` for JavaScript files

Code Quality

CRITERIA	MEETS SPECIFICATIONS
HTML Formatting rules	• All code (`HTML` element names, attributes, attribute values) is lowercase (except `text/CDATA`). • Code does not have trailing white spaces. • Indentation is consistent (either all tabs or all 2 spaces or all 4 spaces etc). • Code uses a new line for every block, list or table element and indent every such child element (it's acceptable to put all `<li>` elements in one line). • [Optional] When quoting attribute values, code uses double quotation marks.

CRITERIA	MEETS SPECIFICATIONS
HTML Style Rules	• `HTML` documents use `HTML5` `<!doctype html>`. • Code passes `HTML` and `CSS` validators. *[Optional] Code does not use entity references unless necessary e.g. characters with special meaning in `HTML` (like < and &) as well as control or “invisible” characters (like no-break spaces). • [Optional] Code omits type attributes for style sheets and scripts.
CSS Formatting Rules	• Code does not have trailing white spaces. • Indentation is consistent (either all tabs or all 2 spaces or all 4 spaces etc). • Code indents all block content, that is rules within rules as well as declarations to reflect hierarchy and improve understanding. • Code uses a semicolon after every declaration for consistency and extensibility reasons. • Code always uses a space after a property name's colon, but no space between property and colon, for consistency reasons. • Code always use a single space between the last selector and the opening brace that begins the declaration block. • Code always start a new line for each selector and declaration. • Code always put a blank line (two line breaks) between rules. • [Optional] Code uses double quotation marks for attribute selectors or property values. Do not use quotation marks in `URI` values (`url()`).

CRITERIA	MEETS SPECIFICATIONS
CSS Style Rules	• Code uses meaningful or generic ID and class names that are as short as possible but as long as necessary. • Code does not use element names in conjunction with IDs or classes. • Code uses shorthand properties where possible. • [Optional] Code omits unit specification after 0 values. • [Optional] Code includes leading 0s in decimal values for readability. • [Optional] Code uses 3-character hexadecimal notation where possible. • [Optional] Code separate words in ID and class names by a hyphen. • [Optional] Code avoids user agent detection as well as `CSS` "hacks"—try a different approach first.
General Meta Rules	• `HTML` templates and documents use `UTF-8` encoding. (no `BOM`) i.e. `<meta charset="utf-8">`. *[Optional] Mark todos and action items with `TODO`

Suggestions to Make Your Project Stand Out!

Use srcset in the img elements to provide optimized images to users on all device sizes.

Include additional JavaScript functionality, while maintaining required components. For example: Bootstrap Navbar, Polymer Components.

Student FAQ