Milestone 1: Front-End Project

Resume with Validating Contact Form

Deadline: Monday, July 20th 2020, 9:00 - Presentations Will Commence

[**GitHub Classroom Repository**](https://classroom.github.com/a/k2t_NI1e)

TECHCareers measures your progress with a series of assignments, quizzes and milestone projects. Milestone projects are larger projects that are expected to be completed to a professional standard of quality. Typically this means spending much more time on polishing a project after it is ‘finished’. Most other assignments will be satisfied by you making your app work, but this milestone project is meant to be a professional portfolio piece and you can expect this kind of project to be carefully inspected by employers who are looking through your Github account. Pay close attention to the Rubric for this assignment. You will notice that we are inspecting this project for things like spelling, and accessibility. We will be running each of your source files through validator(s). The list of requirements in the rubric is much longer than in previous assignments but you should notice that most of the items on the rubric focus on quality assurance rather than focusing on adding more features to this project.  
It is our **strong suggestion** that you plan on finishing your project at least **a day and a half early** so that you can **double-check** the rubric and make sure that your code is perfect. You will likely need to double-check because by the time you reach the end of the list you may have broken something from earlier in the list, so make sure that the full project is fully validated and tested before the deadline.

Time must be tracked on a timesheet with your name on it [in this folder](https://drive.google.com/drive/u/0/folders/1myRiBixLgIdLWErfY1RO76LURo7M225H). the template can be found in the parent folder if you do not yet have one. **Make a copy** of the template, do not edit or move the original document.

As a milestone project, you will be expected to present the finished product to the class.

# Resume

This assignment is for you to make a resume website that has a ‘Contact Me’ form. The resume must contain your own personal information regarding your Goals/Objectives, your Work Experience, your Education, and any other relevant information about your career that you want future employers to know about. For this assignment you are not allowed to put your contact information (phone, mailing address, etc) on your webpage as this can leave you vulnerable to scammers. The [Contact Me](https://www.w3schools.com/howto/howto_css_contact_form.asp) form will be a form that allows users to write an email to you using a mailto link as the form’s action (in the example we have linked the form’s action is made for php. You will have to research how to do this using mailto on your own). Your contact form will also have to validate user input using JavaScript before it open’s the user’s email client to send the form.

# Mandatory Requirements (See Rubric for Details):

* Resume is complete
  + Objective content is included
  + Work Experience content is included
  + Education content is included
  + Resume has a link to your Github account homepage
  + All content is spell checked
  + All content is grammar checked
* Contact form is complete
  + Form has an input for the sender’s email address
  + Form has an input for the subject of the email
  + Form has an input for the body of the email
  + Form will not allow any of the following ‘swear’ words into the email (fake swear words courtesy of Battlestar Galactica):
    - ["feldercarb", "frack", "skinjob", "vulgacarb"]
    - If the user does enter one of these words then an error message will appear reminding the user that you are a professional and you will not tolerate unprofessional language.
* Website includes a mobile, tablet and desktop layout implemented with a media query
  + CSS is in mobile-first format

# Challenge Goals:

Challenge goals are optional and are meant for students who would like a more difficult challenge. You are permitted to add any feature you like so long as you have the mandatory requirements completed.

* You can add buttons to show/hide sections of your resume based on what the employers want to see. For instance - I could have a series of buttons labelled “Tech”, “Forestry”, “Odd Jobs”, then on each job item in my resume I could have class = “Tech” or class = “Tech OddJobs” (notice OddJobs is one word now). Then the javascript handler for each button would toggle the visibility of each job item based on the class name. You can research how to make a collapsible as well.
* You can use FontAwesome to add icons for social media websites BUT your social media account must be appropriate to show to potential employers. For instance - I don’t mind employers seeing my Twitter, but I would never let an employer see my Facebook.
* You can add a portfolio page or section to show visual content. You may not be an artist or a photographer, but screenshots of your own applications can be a nice touch for people who are not technologically inclined enough to look at your github. People like Human Resource managers may really appreciate this touch.

# 

# Hints

* Use the [HTML validator](https://validator.w3.org/) to check your HTML. **Errors found by this validator may take a considerable amount of time to fix if you leave it all to the end. I recommend validating your code frequently.**
* Use the [Accessibility validator](https://achecker.ca/checker/index.php) to make sure your code works for users who have disabilities. **Errors found by this validator may take a considerable amount of time to fix if you leave it all to the end. I recommend validating your code frequently.**
* Use the [CSS validator](https://jigsaw.w3.org/css-validator/#validate_by_input) to check your CSS. **Errors found by this validator may take a considerable amount of time to fix if you leave it all to the end. I recommend validating your code frequently.**
* Your mobile first design should not need a css media query for the mobile version. It should only need media queries for the desktop version.
* C.A.R.P (Contrast, Alignment, Repetition, and Proximity) tips can be found [here](https://vanseodesign.com/web-design/basic-design-principles/) and in other places all over the web.
* Aaron’s recommended HTML/CSS workflow for this project goes:
  + Write your resume in MS Word
  + Decide how many pages you want and build a **sitemap**
  + For each page draw 5 wireframes and pick your favourite
    - Draw wireframes for mobile and desktop
    - Use pencil and paper so you can sketch out ideas quickly
  + For each page you will want to work in the following order (largest to smallest)
    - Layout
    - Background
    - Typography
    - Visual elements (images, buttons, flourishes, etc.)
    - One-off elements and CSS bugs

# Rubric

You will be evaluated on the following points. You must complete all 4 of mandatory sections to pass:

|  |  |
| --- | --- |
| **Requirements** | **Points** |
| **Mandatory Section**  **Functional / General Requirements (11):**   * Assignment submitted on time. * Resume is free of spelling and grammar errors. * Contact form will not permit any Battlestar Galactica swears: ["feldercarb", "frack", "skinjob", "vulgacarb"] * Resume web page has a github link. * Repository on Github Repository has consistent commits (not just a few commits the day before it is due).   + Commit messages describe the code changes within the commit. * Repository has a README.md.   + README.md has a Trello link and Trello is public.   + README.md has a citations summary. * Any “borrowed” code has in-line comment citations. * Commented out code has been removed.   **CSS and Design Requirements (13):**   * No W3C CSS validator errors. * Reset script is linked properly so as to not overwrite any custom styles. * CSS selectors are only as specific as they need to be. * Hex codes or rgb/rgba values are used for colours. Not named colours ('red', 'blue', 'chartreuse'). * Absolute and relative positioning are not used (top, bottom, left, right properties). * Responsive layout for mobile devices is included, and is the base styleset (no media query). * Responsive layout for tablets is included. * Responsive layout for desktop computers is included. * Website demonstrates good use of Contrast. * Website demonstrates good use of Alignment. * Website demonstrates good use of Repetition. * Website demonstrates good use of Proximity. * CSS is properly indented.   **HTML5 Requirements (15):**   * No W3C HTML validator errors. * No WCAG validator accessibility errors. * Semantic HTML5 used wherever possible. * Divs and spans are only used where necessary for styling purposes. * Code does not contain inline style attributes. * Code does not contain inline JavaScript event listeners. * Tags and attribute names are lowercase. * Classes and IDs are camelcase. * Tags are closed and nested properly. * Images have been optimized by [ImageOptim](http://imageoptim.com/) (or similar tool). * Form elements are paired with labels elements containing the for attribute. * Alt attributes exist on all <img> elements. * Uses a valid <title> element with a valid text node. * Uses description metadata. * HTML is properly indented.   **JavaScript Requirements (22):**   * No console errors while running the site AND no errors when validating JS at <https://esprima.org/demo/validate.html> * Code avoids hardcoded values where possible, using constants values instead. * Unused or unreachable code is not present. * No console.log()’s are present (unless you have strong justification for why you need it). * Variables and constants are in camelcase, functions and methods are in pascalcase. * Variables, constants, methods and functions are named semantically. * Global variables (var) are not used, only let and const are used.   + All variables are declared before being used.   + Constants are used over variables when possible. * Code is well-structured (one entry point, one exit point per code block), no returns (other than at the end of a method), breaks or continues are used. * JavaScript does not add inline CSS styles. * When selecting elements with JavaScript, if a reference to a more direct parent exists, that is used for the selection rather than the document node (myList.querySelector() vs document.querySelector). * Javascript named methods contain a docstring comment describing inputs, outputs and purpose. * Code that will run at the same point in all branches of a decision is removed from the decision. * For and for-of/for-in loops are used over while loops where appropriate (counter-controlled, array iteration). * Form errors are displayed in an unordered list so the user can see multiple errors at once. * The form will only submit if the form validates, otherwise the default submit behaviour will be prevented. * ES6 techniques are used over ES5 techniques (for-of over array.forEach, arrow functions over anonymous functions, etc). * Variables and constants are scoped appropriately. * Variables and constants are declared at the beginning of their parent code block. * The script is deferred. * JavaScript is properly indented. | 61 |
| **Challenge / Optional Section**   * A unique/unanticipated feature not mentioned in this document has been added. * Content can be filtered, either with buttons, collapsibles, or another method. * FontAwesome icons have been added. * A visual portfolio has been added. | 4 |
| Total: | 65 |

# 

# Citation Guide

Whenever you borrow code, the following information must be included:

* Comments to indicate both where the borrowed code begins and ends.
* A source linking to where you found the code.
* Your reason for adding the code to your assignment/project instead of writing it out yourself
* How it works. Explain to us how the code is supposed to work, include links to documentation/articles you read to help you understand.
* A small demonstration to prove you understand how the code works.