Assignment 2 – Javascript and JSON  
COS318 – Web Programming

The second assignment begins! *The song of time plays in the background. (to help you get in the right mood: https://www.youtube.com/watch?v=jlMWSAcQce4)*  In this assignment, you’ll be creating some more advanced javascript than you did in assignment 1. You’ll also be doing work with the DOM and JSON objects. Your page will accept a generic block of text from a user, then if the inputted text is valid JSON, generate a web form based on the input. Hopefully you will obtain the three triforce pieces along the way as well, but that’s not a required part of the assignment.

1. **(10 Points) Textarea and parse button**
   1. Create a textarea on the page and a button below it. Pressing the button should parse the text in the text area into JSON.
   2. If the text isn’t valid JSON, display an error.
   3. Each time the parse button is clicked, all previous errors and results should be cleared from the page before doing any other processing of the JSON.
2. **(30 Points) JSON key ‘buttons’**
   1. Parse the JSON ‘buttons’ key and use it to create html buttons on the page.
   2. The ‘buttons’ key will be an array of strings.
      1. The ‘buttons’ key is not required. If it is missing, then continue to parse the rest of the JSON as normal.
   3. Validate and display errors if any of the following are true:
      1. ‘buttons’ key isn’t an array
      2. ‘buttons’ array elements aren’t strings
3. **(40 Points) JSON key ‘fields’**
   1. Parse the JSON ‘fields’ key and use it to create html fields on the page.
   2. The ‘fields’ key will be an array of strings or objects.
      1. The ‘fields’ key is not required. If it is missing, then continue to parse the rest of the JSON as normal.
   3. If an element of the array is a string, display a text box labeled with that string.
   4. If an element of the array is an object, read the name and default keys of this object. The name key becomes the label for the input box, and the default key sets the initial value of the input box.
   5. Validate and display errors if any of the following are true:
      1. ‘fields’ key isn’t an array
      2. ‘fields’ array element isn’t an object or a string
      3. ‘fields’ array element was an object, but didn’t contain a name key
4. **(20 Points)** Code style, formatting, completeness, and quality.
   1. **Parse as much as possible!** For example, If the ‘buttons’ key doesn’t exist, the ‘fields’ key should still be parsed and created on the page.
   2. The page should load and work correctly with no external libraries.
   3. You can find JSON that covers all the cases required on the class github page.

Stretch Levels

If you already have a lot of experience with Javascript and JSON or you just really like Zelda, try to complete these stretch levels for a reputation bonus. If you try for the stretch levels, make sure to type it in the comments on Moodle so I don’t miss it.

**Power Level**

Add some CSS to your page to make it look nicer. Background colors, font colors, or anything that looks good.

**Courage Level**

Add another type of data that your page supports, the ‘selects’ key. The ‘selects’ will be an array of arrays, is not required in the JSON, and each one generates a select box with the options in each sub array. Each option must be a string or number. As with the base assignment, all data types should be validated.

**Wisdom Level**

You will need to complete Courage Level first. Select boxes allow for default values. Come up with a way for a user to specify in the JSON a default value that will be selected automatically for them in the select boxes. No matter the format, validate to make sure the data types match as expected. There are no JSON examples for this stretch level on github. You will need to create your own. Be sure to give some JSON examples in the moodle comments that show your JSON structure working and some invalid JSON that shows error messages when the user enters bad data formatting for your default value.

The Rules

1. No inline styles or inline javascript.
2. Error messages must be “in-page” i.e. no pop-ups or alerts.
3. Any resources not created by you (images, javascript libraries, etc.) must be referenced using a CDN or URL, not directly included in your assignment submission.