Ryan Yoon (rdy23)
CS 2300
Project 2

Rationale

I implemented and developed an online catalog for the characters of the recently released mobile game Fire Emblem Heroes. The online catalog neatly displays the properties of each character and allows users to search the catalog for characters satisfying user-defined conditions or add custom characters of their own.

When using the search function, a user can specify zero or more of the properties listed. If a user does not specify any of the properties and runs the search, the catalog will simply display all of the characters since no restrictions were specified. If a user specifies a single property, then any character that matches that specification will be displayed. For example, if a user specifies 'Name: Roy', only those characters with name Roy will appear below (in this case, only the single character Roy is displayed since names are unique). However, if a user specifies 'Gender: Male', many characters will be displayed since there are many characters with gender male (Roy and Leo will be displayed, but not Lilina since her gender is female). If two or more properties are specified, only those characters that meet all of the specified properties are listed, i.e., searching multiple fields is an "and" search. For example, if a user specifies 'Attribute: Red, Class: Sword', only those characters that satisfy both properties would be listed. For the above example, Alfonse would be displayed, but not Leo because his class is Mage. I chose to make searching multiple fields an "and" search because it is more practical for users trying to find a single character in a large database like this one.

Text inputs for the name and title when using the search function are limited to 40 characters and can only contain letters and white spaces, i.e., no numbers or special characters. I check this by comparing the text inputs to a regular expression that matches zero or more only lowercase letters, uppercase letters, and white spaces. I prohibit special characters, numbers, and strings exceeding 40 characters because they do not make sense in the context of the character properties. Searches are case insensitive, so a search for 'Name: roy' would result in the character Roy. Leading and trailing whitespaces are also trimmed before any search or validation is performed. The hp, atk, spd, def, and res fields only allow numbers ranging from 0 to 99, and I restrict the input to that range using html attributes.

When using the add function, a user must specify all of the properties listed before the input is validated. This was done by adding the required attribute to the input elements and setting default values to the select elements. The name and title inputs only allow letters and white spaces, and the number inputs only allow numbers ranging from 0 to 99 (the way this was done is similar to search). An additional restriction is placed on the name and title inputs that prohibits duplicates. In other words, if a character already exists in the database with either the input name or the input title, the character will not be added. This is achieved by traversing each character in the database and comparing his or her name and title to the input name and title.

For the WOW factor, I defined a Hero class with specific properties that describe a character in the database. I used the Hero class to organize the data so that it was more efficient and convenient to transfer information about any character. Using the Hero class allowed me to package all of the information pertaining to a single character and ship it to be used in other functions. For example, displaying search results was very simple since it just involved traversing through an array of Hero objects and displaying each one. I also used JQuery to create a smooth transition upon pressing the "View The Heroes Below!" or "Return To Top!" buttons. I also used JQuery to fade in the "Return To Top!" button once the page had scrolled past a certain breakpoint, and fade out once the page had scrolled back up past a certain breakpoint.