DESIGN PATTERNS

The following design patterns have been indicated as comments in the project, the JavaScript file specifically:

1. **THE MODULE PATTERN**: The Module pattern is used to further *emulate* the concept of classes in such a way that we're able to include both public/private methods and variables inside a single object, thus shielding particular parts from the global scope. What this results in is a reduction in the likelihood of our function names conflicting with other functions defined in additional scripts on the page.

2. **THE SINGLETON PATTERN**: The Singleton pattern is thus known because it restricts instantiation of a class to a single object. Classically, the Singleton pattern can be implemented by creating a class with a method that creates a new instance of the class if one doesn't exist. In the event of an instance already existing, it simply returns a reference to that object.

3. **THE COMMAND PATTERN**: The Command pattern aims to encapsulate method invocation, requests or operations into a single object and gives us the ability to both parameterize and pass method calls around that can be executed at our discretion. In addition, it enables us to decouple objects invoking the action from the objects which implement them, giving us a greater degree of overall flexibility in swapping out concrete classes (objects).