David Cardona
DGMD E-28 – Developing Single-Page Web Applications
Assignment 3 – Object-Oriented Programming
Spring 2022

Web Links

- Portal with all links

https://dcardonab.github.io/DGMDE-28_SPA/assignments/A3/assignment3.html

- Task 1 – Associative Array

https://dcardonab.github.io/DGMDE-28_SPA/assignments/A3/t1-1_associative.html

- Task 1 – Simple Object

https://dcardonab.github.io/DGMDE-28_SPA/assignments/A3/t1-2_simple-object.html

- Task 1 – Function Object

https://dcardonab.github.io/DGMDE-28_SPA/assignments/A3/t1-3_function-object.html

- Task 1 – Class

https://dcardonab.github.io/DGMDE-28_SPA/assignments/A3/t1-4_class.html

- Task 2 – Blackjack

https://dcardonab.github.io/DGMDE-28_SPA/assignments/A3/t2_blackjack.html

What are a) the most interesting and b) the most surprising Javascript constructs we have learned in the past few weeks.

a) To me, the most interesting Javascript constructs include the variety of data structures and approaches that are available for developers to implement any given task. Like Python, it is possible to condense a lot of information into a single like of code, taking advantage of techniques such as array deconstruction, methods, and implicit conversion of data types via the '+' operators. Getting accustomed to these approaches requires a lot of practice, as they can lead to mistakes that may be difficult to detect. However, fluency with these constructs may give developers an arsenal of techniques to efficiently complete tasks, each approach valid and preferrable based on the specifics of the task at hand.

b) In terms of the most surprising constructs, I am surprised by the similarities between associative arrays and simple objects. Although these constructs display the varied approaches of Javascript, they may be a source of confusion to new Javascript developers. On a different note, I am fascinated by arrow functions. I like using them for inline-like function declarations, and especially for declaring anonymous functions. Familiarity with them can simplify syntax and enhance readability. Nevertheless, I do not think that every function should be an arrow function. I think the combination of traditional function declarations and arrow functions offers a hierarchical structure where functions that contain main bulks of logic are declared using the *function* keyword, and helper functions can take the form of an arrow function.