|  |
| --- |
| Harvey Nash |
| Unit test exercises | |
| Frontend | |

|  |
| --- |
| 6/16/2016 |

EXCERCISES 1

Create a constructor function Person which has the following functions:

* setFirstName /getFirstName
* setLastName /getLastName
* setGender/getGender: The default value is male
* getFullName (has prefix like: “Mr. FirstName LastName” or “Ms. FirstName LastName”)

Using Jasmine to write the unit tests for the Person and using Karma to run.

Expect:

* The unit test file has name: [ClassName].spec.js
* Have at least one or more Suites.
* Setup and Teardown to define and reset the class instance.
* Has a Spec to check your methods which have been defined in the class.
* Writing the Specs to expect theirs return.

EXCERCISES 2

Create a constructor function Car(maximumNumberOfPassengers) which has the following functions:

* addPassenger(Person) :
  + The parameter has to be an instance the Person in exercise 1.
  + To throw an error with “” if the current number of passengers > maximumNumberOfPassengers
* getNumberOfPassenger: Get the current number of passengers
* getNumberOfMen: Using getGender() of the Person to get the number of men in this car

Using Jasmine to write the unit tests for the Person and using Karma to run.

Expect:

* Setup as the exercise 1
* Has a Spec to check your methods which have been defined in the class
* Writing the Specs to check the correctness of Car’s functions
* Using a spy to track whether getGender is called or not

EXCERCISES 3

Implement constructor function Person and Student to pass their unit tests in excercise\_03 folder

Expect:

* Understand their unit test
* Using Focusing specs feature of Jasmine to run a specific spec and solving one by one.

EXCERCISES 4

Implement Factory and complete its unit tests in exercise\_04 folder.

Expect:

* Understand the description of specs
* Writing the Specs as their description