Assignment 8

06/08/2022 Enkhtsetseg Unurbayar

Id: 110883

Week3 Day 3

1. Consider the following definition of an add () function to increment a counter variable:

Question 6

*var* add = (*function* (){

*var* counter = 0;

return *function*(){

return counter +=1;

}

})();

console.log(add());

*let* count = (*function*(){

*let* counter = 0;

*let* add = *function*(){

return (counter +=1);

}

*let* reset = *function*(){

return (counter = 0);

};

return{

add: add,

reset: reset,

};

})();

console.log(count.add());

console.log(count.reset());

1. In the definition of add () shown in question 6, identify the "free" variable. In the context of a function closure, what is a "free" variable?

Counter

The function "add" includes the anonymous function and local variable "counter". So, the anonymous function uses the variable "counter" of their outer function and returns the calculated value. It means the variable "counter" is a free variable.

1. To add () function defined in question 6 always adds 1 to the counter each time it is called. Write a definition of a function make\_adder (inc), whose return value is an add function with increment value inc (instead of 1). Here is an example of using this function:

//Question 8

*let* make\_adder = *function*(*inc*){

*let* counter = 0;

*let* add = *function*(*value*){

return (counter += *value*);

};

*let* reset = *function*(){

return (counter = 0);

};

return {

add : *function*(){

return add(*inc*);

},

reset: reset,

};

};

*let* add5 = make\_adder(5);

console.log(add5.add());

console.log(add5.add());

console.log(add5.add ()); // final counter value is 15

*let* add7 = make\_adder(7);

console.log(add7.add());

console.log(add7.add());

console.log(add7.add()); // final counter value is 21

1. Suppose you are given a file of JavaScript code containing a list of many function and variable declarations. All of these function and variable names will be added to the Global JavaScript namespace. What simple modification to the JavaScript file can remove all the names from the Global namespace?

I need to move all of these functions and variables to function scope or block scope. Using the IIFE pattern, I can decide this situation.

1. Using the Revealing Module Pattern, write a JavaScript definition of a Module that creates an Employee object with the following fields and methods:

//Question 10

*let* employee = (*function* () {

*let* name, age, salary;

*let* getName = *function* () {

return name;

};

// private

*let* setName = *function* (*newName*) {

name = *newName*;

};

*let* getAge = *function* () {

return age;

};

// private

*let* setAge = *function* (*newAge*) {

age = *newAge*;

};

*let* getSalary = *function* () {

return salary;

};

// private

*let* setSalary = *function* (*newSalary*) {

salary = *newSalary*;

};

*let* incrementAge = *function* () {

setAge(getAge() + 1);

return getAge();

};

*let* increaseSalary = *function* (*percentage*) {

*let* tmpSalary = getSalary();

setSalary(tmpSalary + tmpSalary \* (*percentage* / 100));

return getSalary();

};

*let* toString = *function* () {

return getName() + " " + getAge() + " " + getSalary();

};

return {

setName: setName,

setAge: setAge,

setSalary: setSalary,

incrementAge: incrementAge,

increaseSalary: increaseSalary,

toString: toString,

};

})();

1. Write a few JavaScript instructions to extend the Module of Question 10 to have a public address field and public methods setAddress(newAddress) and getAddress().

//question 11

employee.address = " ";

employee.setAddress = (*newAddress*) *=>* {

address = *newAddress*;

};

employee.getAddress = () *=>* {

return address;

};

employee.setAddress("Fairfield, IA 52557");

console.log(employee.getAddress());