Parcial Primer Corte – Desarrollo Web

1. Pensamos que habíamos terminado con nuestro código, pero apareció este raro error. Parece que la función \_displayAvatar () recibe el primer argumento incorrecto, ya que recibe el último elemento en los nombres de usuario. Modifique el siguiente código para dar solución:

function getUsersAvatars(userNames.length, cb){

var url = "/userAvatars/";

for(var index in userNames){

\_fetchAvatar(url + userNames[index], function(avatarUrl){

\_displayAvatar(userNames[index], avatarUrl);

});

}

}

1. ¿Qué se imprime en consola del siguiente código ?

let name = "Jerry Only";

name = "Glenn Danzig";

console.log( name );

Glen Danzig

1. ¿Qué se imprime en consola del siguiente código ?

let topic = "The latest news in JavaScript";

let topic = "The latest news in ES2015";

console.log( topic );

Uncaught SyntaxError: Identifier ‘topic’ has already been declared.

No pueden haber 2 lets con el mismo nombre.

1. ¿Qué se imprime en consola del siguiente código ?

let message = "web forum";

function printInCaps(value){

let message = value.toUpperCase();

return message;

}

printInCaps("profiles");

console.log( message );

web forum

1. Señala y corrige el error del siguiente código:

function loadProfiles(userNames){

const MAX\_USERS;

const MAX\_USERS = 15;

if(userNames.length > MAX\_USERS){

return false;

}

for(let i=0; i < userNames.length; i++){

\_fetchProfile(userNames[i], function(data){

\_addToSidebar(userNames[i], data);

});

}

}

1. ¿Qué se imprime en consola del siguiente código ?

function hideReplies(topicId){

let previewText;

if(\_isFixedTopic(topicId)){

const TEXT\_SEPARATOR = '%%';

previewText = \_fetchLongPreview(topicId, TEXT\_SEPARATOR);

}else{

previewText = \_fetchPreview(topicId);

}

console.log( TEXT\_SEPARATOR );

}

hideReplies(42);

El output sería: %%

1. La función displayTopicsPreview () provoca un error en la primera línea cuando se llama sin argumentos. . Modifique el siguiente código para dar solución:

function displayTopicsPreview( topics = [] ){

var message = "There are currently " + topics.length;

\_displayPreviewMessage(topics, message);

}

1. Modifique el siguiente código para usar parámetros con nombre y con valores predeterminados:

function loadProfiles(userNames = [], options = {}) {

let profilesClass = options.profilesClass || ".user-profile";

let reverseSort = options.reverseSort || false;

if (reverseSort) {

userNames = \_reverse(userNames);

}

\_loadProfilesToSideBar(userNames, profilesClass);

}

1. De acuerdo a la siguiente función:

function fetchReplies(topicId, { displayClass, includeAvatar }){

//...

}

Seleccione las posibles funciones sin que se genere error:

a.

fetchReplies(12, {

displayClass: "topic-replies"

});

b.

let options = {

displayClass: "topic-replies",

includeAvatar: true

};

fetchReplies(12, options);

c.

fetchReplies(12);

1. De acuerdo a la siguiente función:

function setPageThread(name, {popular, expires, activeClass} = {}){

// ...

}

Seleccione las posibles funciones sin que se genere error:

a.

setPageThread(“ES2015”, {

popular: true

});

b.

setPageThread(“ES2015”, {});

c.

setPageThread(“ES2015”);

1. Cuando se invoca la función getActiveUsers(), el segundo argumento es un callback. Dicho callback, tiene la variable data como un parámetro.

Se puede pasar los datos de data.userNames como argumentos a la función appendUserNames(). Sin embargo, esta dato es un vector mientras que la función appendUserNames() esperar argumentos individuales

Realice las modificaciones pertinentes para que funciones el código:

getActiveUsers(15, function(data){

let userNameDivs = appendUserNames( );

appendToSidebar(".side-bar", userNameDivs);

});

function getActiveUsers(topicId, cb){

\_fetchTopicInfo("/topics/" + id, function(data){

cb(data);

});

}

1. Reescriba la siguiente función en formato arrow =>:

let printName = (value)=>{

console.log( value );

}

1. Se está teniendo un bug debido al alcance de las variables, encuentre el error y modifíquelo:

function ActiveUsersComponent(target, topicId){

this.targetElement = target;

this.topicId = topicId;

}

ActiveUsersComponent.prototype.render = function(){

getActiveUsers(this.topicId, function(data){

let userNameDivs = appendUserNames(...data.userNames);

appendToSidebar(this.targetElement, userNameDivs);

});

};

1. Complete el siguiente código para que se imprima “This topic has 21 a total of replies” siguiendo las nuevas funcionalidades de ES2015:

let replyCount = 21;

var message = ‘This topic has $(replyCount) a total of replies’;

console.log(message);

1. Dados los siguientes objetos:

let defaults = {

container: ".main",

isActiveClass: ".is-active"

};

let options1 = {

container: ".main-container",

isActiveClass: ".is-active-element"

};

let options2 = {

isActiveClass: ".is-active-content"

};

Cuál es el resultado del siguiente código:

Let let no puede ser mayúscula settings = Object.assign({}, defaults, options1, options2);

Se reproduce la función normalmente

1. Dada la siguiente función:

function spinner(target, options = {}) {

let defaults = {

message: "Please wait",

spinningSpeed: 5,

cssClass: ".is-spinning"

};

let settings = Object.assign({} , defaults , options);

console.log(`Message: ${settings.message}`);

console.log(`spinningSpeed: ${settings.spinningSpeed}`);

console.log(`cssClass: ${settings.cssClass}`);

}

Cual es resultado en consola de invocarla de la siguiente manera:

spinner(targetElement, {

cssClass: ".is-fast-spinning",

spinningSpeed: 8

});

Message: Please Wait

SpinningSpeed: 8

cssClass: .is-fast-spinning