PAC 2

Llenguatges de desenvolupament front-end

Universitat Oberta de Catalunya

Informació rellevant:

- Data límit de lliurament: 10 d'abril.
- Pes a la nota de FC: 15%.

Contingut

Informació docent

Presentació

Aquesta pràctica se centra a conèixer les millores en el llenguatge de JavaScript al llarg dels darrers anys basats en l'estàndard ECMAScript, proporcionant fonaments del llenguatge Web que és utilitzat per tots els frameworks d'avui dia.

Objectius

Els objectius que es volen assolir amb el desenvolupament d'aquesta PAC són:

- **Desenvolupar** codi JavaScript **usant les característiques d'ES6-ES12.**
- **Utilitzar el patró darquitectura MVC** en el desenvolupament duna aplicació web.

Enunciat

Aquesta PAC conté 3 exercicis avaluables. Has de lliurar la solució dels 3 exercicis avaluables (veure l'últim apartat).

Com que les activitats estan encadenades (i.e. per fer-ne una s'ha d'haver comprès l'anterior), **és altament recomanable fer les tasques i els exercicis en l'ordre en què apareixen en aquest enunciat.**

Abans de continuar has de:

- Haver llegit els recursos teòrics disponibles a la PAC.
- Crea una carpeta PEC2 i inicialitza git en aquesta ruta. Igual que a la PAC1, has de crear un fitxer README.md a l'arrel de la carpeta que contingui que contindrà almenys aquesta informació:

- Login UOC
- Nom i cognoms de l'alumne.
- Breu descripció del que s'ha realitzat en aquesta PAC, dificultats, millores realitzades, si cal tenir alguna cosa en compte a l'hora de corregir/executar la pràctica o qualsevol aspecte que vulgueu destacar.

Exercici 1 – Callbacks/Promeses/async-await (2 punts)

Abans de continuar has de:

Descarregar el fitxer **`PEC2_Ej1.zip`** adjunt a la PAC i descomprimir-ho dins de la carpeta PAC2.

Observa el contingut del fitxer ejer1.js:

```
const findOne = (list, { key, value }, { onSuccess, onError }) => {
    setTimeout(() => {
        const element = list.find(element => element[key] === value);
        element ? onSuccess(element) : onError({ msg: 'ERROR: Element Not Found' });
    }, 2000);
};

const onSuccess = ({ name }) => console.log(`user: ${name}`);
const onError = ({ msg }) => console.log(msg);

const users = [
  {
    name: 'Carlos',
    rol: 'Teacher',
  },
  {
    name: 'Ana',
    rol: 'Boss',
  },
];

console.log('findOne success');
findOne(users, { key: 'name', value: 'Carlos' }, { onSuccess, onError });

console.log('findOne error');
findOne(users, { key: 'name', value: 'Fermin' }, { onSuccess, onError });

/*
findOne success
findOne error
  //wait 2 seconds
user: Carlos
ERROR: Element Not Found
*/
```

a. El codi anterior fa ús de *callbacks* per realitzar una tasca després de fer la cerca en un *array* (`find`). Dins la carpeta PEC2_Ej1, crea un fitxer `ejer1-a.js` i documenta al codi, que s'està realitzant a cada línia, posant èmfasi en el *callback* i el valor mostrat per la pantalla. (0.5 punts)

b. Copia el fitxer resultat de l'exercici anterior a un fitxer `ejer1-b.js`. Després modifica el codi anterior per eliminar **els** *callbacks* i, al seu lloc, fer ús de promeses (cal crear-les a la funció `findOne`) i consumir-les al codi principal (amb les paraules reservades `then` **i** `catch`). Documenta al codi, línia a línia, quins canvis has realitzat. (0.5 punts)

c. Copia el fitxer resultat de l'exercici anterior a un fitxer `ejer1-c.js`. Després modifica el codi anterior per fer ús de `async/await` (tingues en compte que hauràs de separar en una funció l'ús de la funció `async`. És a dir, podràs crear una funció extra si és necessari). Explica línia a línia a línia, dins del codi mateix, què has modificat. (0.5 punts)

d. Copia el fitxer resultat de l'exercici anterior a un fitxer `ejer1-d.js`. Escriu i documenta al propi codi, com seria usant promeses i `async/await` la versió paral·lela, és a dir, que s'executin totes les trucades a la funció `findOne` en paral·lel, sense necessitat d'esperar que conclogui la primera per executar-se la segona. (0.5 punts)

Durant el desenvolupament de l'exercici executa les ordres `git` necessàries per afegir aquesta nova carpeta, fitxers i els canvis que hi facis al repositori local `git` de la PAC.

Exercici 2 – Arquitectura MVC usant VanillaJS (4 punts)

En aquest exercici construirem, usant totes les novetats de JavaScript, una aplicació SPA (*Single Page Application*) sense utilitzar cap framework, el que es coneix com programar usant VanillaJS o JavaScript pur. És a dir, VanillaJS és la referència per crear aplicacions basades en JavaScript que no fan ús de cap framework (p.ex. Angular o Vue). Això sí, seguirem l'arquitectura de MVC (Model-Vista-Controlador). Aquesta pràctica us permetrà conèixer l'arquitectura que segueixen frameworks com Angular, però utilitzant com a llenguatge base JavaScript.

En primer lloc, hem de comprendre com es construeix una aplicació usant el patró d'arquitectura MVC utilitzant VanillaJS. Per això, us facilitem un projecte ja acabat anomenat `Ejer2-1-TODO` que consisteix en una aplicació SPA que realitza les 4 operacions elementals CRUD (Create, Read, Update i Delete) sobre una llista de tasques (TO**DO**). Un cop comprès el projecte `Ejer2-1-TODO`, haureu de fer una aplicació similar trucada `Ejer2-2-expense-tracker`.

Abans de continuar heu de:

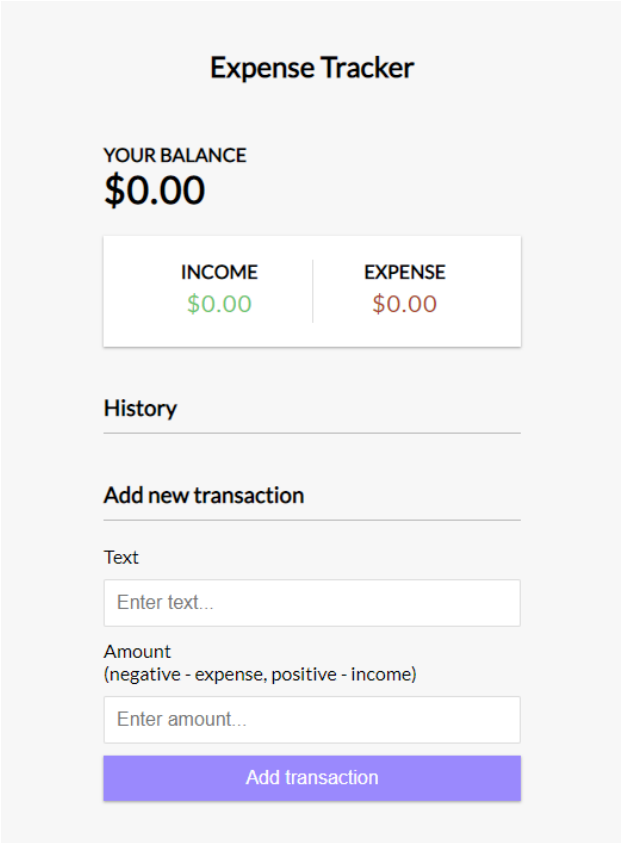
- Descarregar el fitxer **`PEC2_Ej2.zip`** adjunt a la PAC, que conté els dos projectes **`Ejer2-1-TODO`** i **`Ejer2-2-expense-tracker`**.

- Llegir el recurs **`P02.PAC2_Exemple_Arquitectura_MVC.pdf`** que explica com està construït el projecte **`Ejer2-1-TODO`**.

Un cop vist l'exemple d'una aplicació completa usant el patró MVC programada a VanillaJS, heu de crear una aplicació MVC de control de despeses. Per això heu de seguir els passos del codi anterior per crear un CRUD per al control de despeses. És a dir, es realitzaran les operacions de crear, esborrar, editar i mostrar un registre de despeses.

Al codi facilitat per comprendre el MVC es realitzen totes les operacions sobre objectes TO**DO**, en aquest cas, el domini del problema ha canviat per ser registres de despeses.

El codi font inicial de la pràctica (`Ejer2-2-expense-tracker`) genera la següent pàgina:



La informació de les despeses que volem gestionar és la següent:

- Text de la despesa
- Quantitat de la despesa

Abans de començar amb el desenvolupament de `Ejer2-2-expense-tracker`

Crea una carpeta `PEC2_Ej2`, dins d'aquesta carpeta crea un fitxer text que tingui com a nom `PEC2_Solucion_Ejercicio_2a.md` i respon a la següent pregunta sobre `Ejer2-1-TODO`.

a. (0.50 punts) Observa que s'han creat funcions *handle* al fitxer controlador (`todo.controller.js`), les quals són passades com a paràmetre. Això és degut al problema amb el canvi de context (`this`) que existeix a JavaScript. Ara mateix si no tens molt clar que està succeint, revisa què fan les **"**fat-arrow**"** d'ES6 sobre l'objecte `this`, i prova a canviar el codi per comprendre què està passant quan es modifica la línia següent:

```
this.view.bindAddTodo(this.handleAddTodo);
```

Per aquesta:

```
this.view.bindAddTodo(this.service.addTodo);
```

Respon, en un document text al directori de lliurament a la següent pregunta:

Per què és el valor de `this` és `undefined`?

A continuació, el que es demana és que dins de la carpeta `PEC2_Ej2`, creus una subcarpeta `Ejer2-2-expense-tracker` on aniràs modificant el codi i estructura del projecte per convertir un simple projecte amb html/css/js en una estructura MVC fent ús del potencial d'ES6 i seguint el codi d'exemple (TODO) que s'ha proporcionat.

En primer lloc, heu de generar l'estructura de directoris del nostre projecte (models-vistes-serveis-controladors) i anar pensant on ubicar el contingut dels diferents fitxers.

b. (0,50 punts) Construeix les classes relatives a models, controladors, serveis, vistes i llançador de l'aplicació des d'on aniràs desenvolupant l'aplicació. En aquest punt només heu de crear l'estructura de fitxers que modelen el nostre problema. És a dir, organitzar les classes relatives a models (`expense.model.js`), controladors (`expense.controller.js`), serveis (`expense.service.js`) i llançadora (`app.js`).

c. (0.50 punts) Construeix la classe model (anèmic) que sigui necessària per a aquesta aplicació.

d. (1 punt) Construeix la classe servei que és l'encarregada de fer totes les operacions sobre una estructura de dades (on s'emmagatzemarà la informació de totes les despeses).

e. (0,50 punts) Construeix la classe vista que controlarà totes les operacions relatives a la vista.

f. (0,50 punts) Construeix el controlador que és l'encarregat de posar en comunicació la vista amb el servei, en aquest projecte.

g. (0,50 punts) El projecte lliurat no té implementat l'actualització d'una despesa concreta. Afegeix aquesta nova funcionalitat.

Durant el desenvolupament de l'exercici executa les ordres `git` necessàries per afegir aquesta nova carpeta, fitxers i els canvis que hi facis al repositori local `git` de la PAC.

Exercici 3 – Arrays (4 punts)

Utilitzant els mètodes natius d'*array* como son *filter, some, map, reduce, …* has de completar els fitxers `core.js` que se subministren al projecte `PEC2_Ej3.zip` inclòs a la PAC.

Abans de continuar has de:

Descarregar fitxer **`PEC2_Ej3.zip`** adjunt a la PAC

Per fer l'exercici només heu de modificar els fitxers `core.js`, **no s'han de modificar els fitxers de tests `core.spec.js`.**

L'objectiu és aconseguir que l'execució dels tests de les proves definides doni com a resultat els `tests passed`.

Instruccions:

- Instal·lar motxa amb l'ordre `npm install -g mocha`.
- Completar el codi del fitxer `core.js` de cada subcarpeta
- Executar cadascun dels tests amb la instrucció `mocha <nom prova>` per exemple `mocha a_map`

Has de completar el codi del fitxer `core.js` i aconseguir que els tests donin com a resultat `passed` per a totes les carpetes subministrades al zip.

a. map – 0.5 punts
b. filter – 0.5 punts
c. reduce – 0.5 punts
d. every – 0.5 punts
e. some – 0.5 punts
f. zoo (exercici complet) – 1.5 punts

Si necessiteu més informació, els següents enllaços disposen d'abundant documentació sobre els mètodes nadius d'Array:

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array

https://www.w3schools.com/js/js_arrays.asp

Durant el desenvolupament de l'exercici executa les ordres `git` necessàries per afegir aquesta nova carpeta, fitxers i els canvis que hi facis al repositori local `git` de la PAC.

Format i data de lliurament

Has de lliurar un fitxer *.zip, el nom del qual ha de seguir aquest patró: `loginUOC_PEC2.zip`. Per exemple: `dgarciaso_PEC2.zip`. Aquest fitxer comprimit ha d'incloure els elements següents:

La **carpeta PEC2**, i al seu interior:

- Un fitxer README.md a l'arrel de la carpeta amb la informació indicada.
- La carpeta oculta `.git` amb el contingut del repositori local `git`.
- Una carpeta `PEC2_Ej1` amb el codi completat i comentat seguint les peticions/especificacions de l'Exercici 1. Com el que se sol·licita és modificar el codi `ejer1.js`, heu de lliurar un `ejer1.js` modificat per a cada apartat dels sol·licitats. És a dir, heu de lliurar quatre fitxers: `ejer1-a.js, ejer1-b.js, ejer1-c.js y ejer1-d.js`. No cal lliurar ni l'original `ejer1.js`, ni el `index.html`, però si és molt important que tot el codi que es lliuri aquest comentat amb els canvis realitzats i el motiu.
- Una carpeta `PEC2_Ej2:`
  - Amb el fitxer `PEC2_Solucion_Ejercicio_2a.md` i responeu a la pregunta formulada a l'apartat A de l'Exercici 2.
  - Una subcarpeta `Ejer2-2-expense-tracker` amb el codi completat seguint les peticions i especificacions de l'Exercici 2
  - No cal lliurar la carpeta exemple Ejer2-1-TODO
- Una carpeta `PEC2_Ej3` amb el codi completat seguint les peticions i les especificacions de l'Exercici 3.

Penalitzacions

- Lliurament en un altre format que no sigui l'especificat (ex. en `zip`): -0.75 punts
- Comprimir fitxers dins del `zip`: -1 punts.
- Per a cada exercici/apartat on no es respecti la nomenclatura exacta de les carpetes o fitxers indicats (símbols, minúscules, majúscules, etc.): -0.75 punts.
- La no entrega del repositori local `git` -3 punts

L'últim dia per lliurar aquesta PAC és el 10 d'abril fins a les 23:59. Qualsevol PAC lliurada més tard serà penalitzada.