JESUÏTES El Clot
Escola del Clot

M06. Desenvolupament Web en Entorn Client

UF4. Comunicació asíncrona client-servidor

4.1 Node.JS
4.2 AJAX
4.3 MongoDB

(RA1)

curs 2017-2018

Sergi Grau

Descripció.

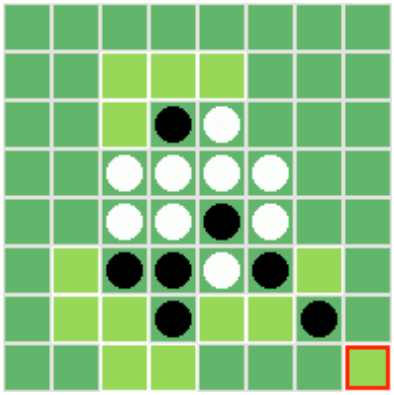
Es vol desenvolupar una aplicació web que permeti jugar a un joc de torns multijugador: Othello.

Podeu trobar les regles del joc a https://ca.wikipedia.org/wiki/Othello_(joc)

Per a fer-ho s'utilitzarà Node.JS. Les comunicacions seran asíncrones utilitzant la tecnologia AJAX.

Per a fer més senzill el joc, suposarem que només poden jugar 2 jugadors. Les dades que s'envien són la fitxa que mou un jugador i la posició. Cal tenir contrades en tot moment les fitxes que li queden a un jugador.

Es tracta d'un joc multijugador, on NO es pot jugar contra la màquina.



Criteris d'avaluació

La puntuació màxima del projecte és un 10.

La puntuació mínima per a superar-lo es de 5 sobre 10.

Rúbrica d'avaluació

La rúbrica d'avaluació del projecte és la següent:

Criteris	4	3	2	1
Sistema d'autenticació (10%)	El sistema de login està implementat, i poden jugar-se diverses partides a la vegada.	El sistema de login està implementat però només poden jugar 2 jugadors. Hi ha logout.	El sistema de login està implementat però només poden jugar 2 jugadors. No hi ha logout.	El sistema de login no està implementat
Lògica del joc en NodeJS(40%)	La lògica és correcta. És veu l'efecte d'encertar de guanyar un moviment, i de posar les fitxes. S'utilitza encaminament.	La lògica és correcta. És veu l'efecte d'encertar de guanyar un moviment, i de posar les fitxes. No s'utilitza encaminament.	La lògica és correcta però molt bàsica. No s'utilitza encaminament.	La lògica és incorrecta i no permet jugar al joc de manera completa.
POO (15%)	S'ha creat la classe Partida i Jugador i s'han implementat mètodes per a les diverses operacions dels moviments. Es fan les verificacions necessàries i associades a la lògica del joc.	S'ha creat la classe Partida i Jugador i s'han implementat mètodes per a les diverses operacions dels moviments, però sense verificar la lògica del koc.	S'ha creat la classe Partida i Jugador.	No s'ha creat cap classe.
AJAX (25%)	S'ha utilitzat XHR nivell 2 amb JSON. La programació del costat client incorpora ES5 i diversos objectes prototipus per a representar la lògica de la partida.	S'ha utilitzat XHR nivell 2. S'utilitzen objectes JSON	S'ha utilitzat XHR nivell 1. No s'utilitzen objectes JSON.	No s'ha utilitzat XHR
MongoDB (10%)	S'ha utilitzat	S'ha utilitzat	S'ha utilitzat	No s'ha utilitzat persistència de

	persistència de dades amb MongoDB, inclou el desament de les puntuacions, consulta de les mateixes i enregistraments dels jugadors. El tractament es fa mitjançant un patró DAO.	persistència de dades amb MongoDB, inclou el desament de les puntuacions, consulta de les mateixes i enregistraments dels jugadors.	persistència de dades amb MongoDB, però és molt bàsica.	dades.

Funcionalitats a implementar.

- El Backend de l'aplicació ha d'estar desenvolupat amb Node.JS
- El Frontend de l'aplicació cal desenvolupar-ho amb JS (és opcional utilitzar ES6).
- Les dades del Frontend s'envien mitjançant AJAX i objectes JSON
- cal informació en el backend a la base de dades NoSQL MongoDB.

Característiques del disseny.

- Cal utilitzar el paradigma de POO.
- El disseny d'interfície web és lliure.
- Les comunicacions amb AJAX utilitzen JSON.
- Només es pot utilitzar Node.JS per a resoldre la part del backend.

Cal lliurar-la abans del dia especificat al l'apartat de lliurament de treballs de la NET.

S'ha d'entregar en fitxer zip o tar, amb el format

COGNOM_NOM_PRACTICA_ENTORN_CLIENT_UF4.zip

- Aquest fitxer contindrà un fitxer en format **PDF** descrivint les característiques de la implementació realitzada.
- Aquest fitxer zip contindrà tot el codi font, biblioteques emprades i el contingut estàtic de la aplicació web.