Architecture et Programmation du web
Mise en oeuvre d'un serveur Node.js et de clients JS gérant des websockets
Synchronisation des occupations / déplacements de n joueurs sur un damier

-

Pierre Pompidor

Etape 1 : créez un clients-serveur

Reportez-vous aux codes donnés dans le support de cours et ci-après pour :
- créer un serveur basé sur la plateforme **Node.js** ;
- et un client (qui sera au moins instanciés deux fois).

se synchronisant grâce à des **websockets**.

Créez le serveur :

```
var nomsJoueurs = [];
var nbJoueursConnectes = 0;
var app = require('http').createServer(function(req, res){});
app.listen(8888);
var io = require("socket.io").listen(app);

io.sockets.on('connection', function (socket) {
    socket.on('etat', function(message) {  //renvoie les joueurs déjà connectés ... });

    socket.on('rejoindre',function(message) { //un nouveau joueur se connecte ... });

    socket.on('quitter',function(message) { //un joueur quitte la partie ... });
});
```

Vous pouvez utiliser **io.emit()** (émission du message à tous) ou **socket.broadcast.emit()** (émission aux autres clients que celui venant d'appeler le serveur) pour émettre des messages.

Lancez le serveur : `node serveur.js`

Créez le client et instanciez-le au moins deux fois :

Le client doit permettre à un joueur de rejoindre ou de quitter la partie (reprendre l'exemple donné dans le support de cours). Gérez au moins deux joueurs en chargeant le client dans (au moins) deux instances de navigateurs positionnées l'une à côté de l'autre.

Etape 2 : affichage d'un damier sur le client

Intégrez dans le client un damier en nid d'abeilles (par exemple de 100 hexagones) → réutilisez le code du TP précédent.

Etape 3 : synchronisation des déplacements

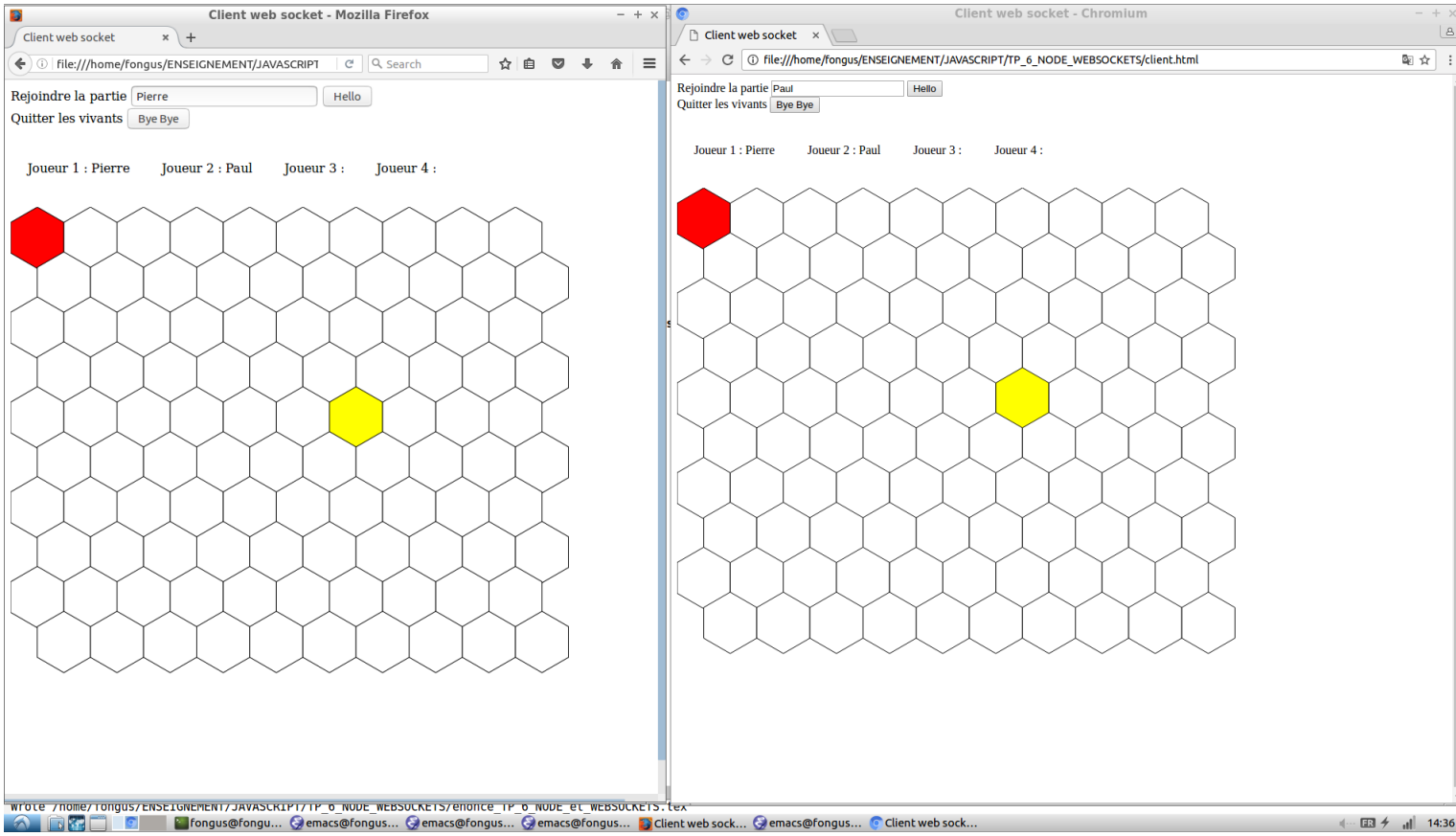
Sur le client, lors d'un clic sur une case du damier, envoyez un message au serveur lui indiquant :
- le numéro du joueur ayant occupé une nouvelle case ;
- l'id de cette nouvelle case.

Voici un fragment de code à attacher à un hexagone :

```
  ...
  .on("click", function(d) {
    var id = d3.select(this).attr('id');
    d3.select(this).attr('fill', couleursJoueurs[joueurLocal]);
    ... // envoyez le message au serveur indiquant la case occupée
  });
```

Sur le serveur, renvoi d'un déplacement à tous les autres joueurs

```
    socket.on('deplacement',function(message) { // un joueur s'est déplacé sur le damier
      ... // utilisez socket.broadcast.emit()
    });
```



Si vos interfaces clientes sont synchronisées, vous avez développé les codes de base vous permettant de synchroniser des déplacements ou occupations de pions sur un damier...