Notes de cours sur le fonctionnement d'express

Nous avons vu dans les séances précédentes que le web2.0 est initialement constitué en couches.

- 1980 Une couche élémentaire permettant le transport d'une information unitaire simple entre deux machines situées n'importe où sur l'Internet. C'est la couche socket. Elle se programme dans n'importe quel langage de programmation qui fournit une interface de programmation. (cf séance d2)
- 1995 Une couche de communication de type requête / réponse permettant l'envois de documents complexes entre un client et un serveur. Le protocole de cette couche est http, qui se résume à l'envoi d'une requête sous la forme : Enveloppe, ligne vide, Corps, à laquelle un serveur compatible renvoie une réponse sous la même forme.
  - Les requêtes qu'un client peut émettre sont aux nombre de quatre. GET, HEAD, PUT, POST. Avec un très large majorité de requêtes GET. L'écriture d'un client s'est fait dans la séance d3.
  - La réponse d'un serveur consiste la plupart du temps à fabriquer un document au format html qui sera présenté par le client. Le document peut contenir des instructions demandant au client l'aller rechercher de nouvelles ressources vers de nouveau sites web pour pouvoir compléter l'affichage de la page. Ainsi le rendu d'une page web dans un navigateur provient de l'accumulation d'informations obtenues sous la forme de différentes requêtes http.
- 2010 Mise en place d'un server Web simple et efficace. Si les serveurs web Apache existent dès le début des années 95, ce n'est qu'assez récemment qu'on a obtenu des versions très simplifiées de ces systèmes. RubyOnRail qui a été utilisé comme serveur de Twitter pendant quelques années en est un exemple typique. Nous étudions en cour le serveur javascript express.
  - Express : http://express.org, appartient à cette catégorie de serveurs web, qui se programme de manière réactive et événementielle en très peu de lignes de codes. Faire fonctionner un serveur node, se fait de la manière suivante :
    - Installation de l'intrepréteurJavaScript nodejs http://nodejs.org
      - On suppose qu'il est déjà installé sur nos postes de travail. Sinon se référer au site de référence pour faire l'installation.
    - Installation pour le serveur node du système express
      - > npm install express
    - Ecriture de la fonction répondant aux requêtes d'un client. Par exemple.

```
1  var express = require('express'),
2      app = express();
3
4  app.use('/public', express.static(__dirname+'/public/'));
5
6  app.get('/', function(req, rep) {
7    rep.send('Hello');
8  });
9
10 app.get('/calcul', function(req, rep) {
11   rep.send('Une réponse');
12  });
13
14 app.listen(3000);
```

monsite.js

Le lancement de ce programme, sous la forme node ./monsite.js permet :

- l14 : d'attendre des demandes de connexions sur le port 3000 de la part de navigateus Web. http://localhost:3000 ou http://tc-info403.insa-lyon.fr:3000
- l4, 6 et 10 : d'associer des actions aux requêtes suivantes :
  - GET / pour la ligne 6
  - GET /public pour la ligne 4
  - GET /calcul pour la ligne 10
- Pour chaque association entre une requête GET d'un client et une action serveur, la second partie de ces fonction indique le traitement à effectuer.
  - Affichage de la page web stockée sur le disque pour la ligne 4
  - Exécution d'une fonction spécifique pour les lignes 6 et 10.

/* Exercice : réaliser une fonction renvoyant à un client le résultat de la suite de fibonnacci pour une valeur fixe donnée*/
/* Exercice : réaliser cette fonction permettant au client web d'indiquer la valeur du rang de calcul de la fonction */

Il est possible de lancer directement express afin qu'une structure type d'un serveur web soit générée. L'exécution de la commande express monsiteweb, générera un 'skelette' de site complet en fonctionnement. La génération produit la structuration 'classique' de site webs, séparant les aspects de calcul et de présentation, ainsi que de codes s'exécutant du coté serveur et de codes s'exécutant du coté client.

Exemples de sites webs typiques :
Au dessus d'express on peut trouver des programmes plus complets que notre exemple monsite.js, qui fonctionnent de manière similaires à certains services standard. Nous en étudions deux exemples :
Les sites de Wiki collaboratif,
Les sites de Chat interactifs.

Un exemple de Wiki sous express est disponible par le service github.
Pour le mettre en place voici les actions :

1) Récupérer le code source du système de gestion de wiki.
   - git clone https://github.com/sfrenot/wiki-challenge.git -b Upgrade-Express-2.5-3.0
2) Installer les modules de dépendance
   - npm install express@3
   - npm install markdown
   - npm install jade
3) Lancer le systeme
   - node ./app.js
4) Accéder au wiki et le manipuler

/* Exercice : tester et monter trois pages web sur un thème quelconque sur le site du wiki que vous venez de créer*/
/* Exercice : modifiez votre wiki afin que la zone de saisie soit sur plusieurs lignes et plus large. Le but ici est de fournir les bons paramètres à la génération de la textarea du formulaire html */