TP 1 : Programmation procédurale JavaScript

Exercice 1 : Calculs de grandeurs de figures

Nous souhaitons calculer le périmètre et la surface d'un rectangle dont les dimensions seront fournies par l'utilisateur. Les fonctions JS à utiliser seront :

- Alert
- Prompt

- ✓ Récupérer et importez le projet « **insee-client-js** » dans Eclipse
- ✓ Coder une fonction qui prend 2 paramètres et retourne la surface
- ✓ Code une fonction qui prend 2 paramètres et retourne le périmètre
- ✓ Modifier la fonction « **main ()** » afin qu'elle puisse demander à l'utilisateur de rentrer la longueur et la largeur et qu'elle puisse successivement afficher la surface et le périmètre. Une démo des affichages attendus sera fournie en TP

Exercice 2 : Manipulation des tableaux

Nous souhaitons afficher les éléments d'un tableau selon l'indice fourni par un utilisateur. Cette action sera répétée autant de fois jusqu'à ce que l'utilisateur saisisse le nombre « -1 » pour sortir. Travail à faire :

- ✓ Coder une fonction qui initialisera le tableau ci dessous :

```
var tab = new Array("Hollande","ZOME","Sarkozy","LE PEN","LE MAIRE","VALLS");
```

- ✓ Faire pointer l'action du bouton sur cette nouvelle fonction. Une démo du déroulement sera fournie

Exercice 3 : Manipulation du formulaire en JavaScript

L'objectif est de pouvoir interagir avec un formulaire.
Nous allons ensemble coder une calculatrice. Travail à faire :

- ✓ Dessiner le formulaire tel que vu dans la démo
- ✓ Coder une fonction JavaScript qui va récupérer les valeurs saisies et mettre à jour dynamiquement le champ résultat.

Exercice 4 : JavaScript et les évènements 1:

L'objectif de cette partie est d'étudier les évènements en JS.

Nous allons construire un formulaire ayant des champs qui feront l'objet de vérifications par un script JS. Pour cela nous allons dessiner un formulaire contenant les champs suivants :

- Un champ « **nom** » : Ce champ doit être obligatoire
- Un champ « **annee_naissance** » : Ce champ s'il est renseigné ne doit accepter que des nombres numériques
- Un champ **email** : Si ce champ est renseigné l'email doit être valide avant de perdre le focus
- Un champ **code postal** : le code Postal doit être un nombre à 5 chiffres s'il est renseigné.

Lors de la soumission, s'il y'a erreur afficher en rouge en bas de page les erreurs du formulaires. S'il n'ya pas d'erreur transmettre le formulaire à un script php.

TP 2 : Programmation Orienté Objet en JavaScript

Nous allons créer nos propres classes dans cette partie.

L'objectif de cette partie est de manipuler quelques notions principales liées à l'héritage. Vous modéliserez un cas concret de la vie professionnelle d'une entreprise.

Cas d'étude : Une SSII souhaite concevoir un logiciel permettant de gérer son personnel, ce logiciel aura pour fonction principale de calculer le salaire annuel d'un membre du personnel, les différentes entités métier à gérer sont :

- Un membre du **personnel** est caractérisé par :
  - Un nom
  - Une année d'embauche
  - Un salaire mensuel

- Un **développeur** (qui est évidemment un membre du personnel) caractérisé par :
  - Un nombre d'Application auxquels il a participé
  - Une note moyenne issue de la bonne qualité de codage

- Un **commercial** (qui est également membre du personnel) caractérisé par :

  - Un nombre de clients gérés
  - Un chiffre d'affaires moyenne sur une année

TAF 1 : Modèle de Classes :

- ✓ Dessiner sur papier un diagramme de classe du système d'information
- ✓ Créer un nouveau fichier « classes.js » dans le répertoire « scripts »
- ✓ Y Coder les classes du diagramme dans le package ci dessus.
- ✓ Penser aux constructeurs avec paramètres

Nous devons désormais doter notre projet du calcul salaire annuel. Le salaire annuel est tout simplement le salaire mensuel x 12 pour tout membre du personnel. Toutefois l'entreprise verse une prime aux commerciaux en fonction du chiffre d'affaires annuel réalisé. Le salaire d'un commercial est en fait son salaire annuel auquel on rajoute un certain pourcentage du chiffre d'affaire réalisé.

TAF 2 : Calcul du salaire

- ✓ Implémenter ce calcul de salaire annuel de manière générale pour tout membre du personnel et de manière spécifique pour les commerciaux.
- ✓ Créer un formulaire qui permettra de simuler le calcul des différents salaires en fonction du type de salarié et du pourcentage.
- ✓ Une librairie « **utils.js** » vous a été fournie pour la lecture et l'écriture dans les composants

TP3 : JQuery et les Web Storage

L'objectif de cette partie est à plusieurs niveaux :

- Manipuler quelques instructions en JQuery
- Manipuler les différents de stockage en JavaScript
- Générer dynamiquement des composants Html via JavaScript
- Manipuler les apis de persistance de données en JavaScript

Importation du projet :

- ✓ Importez le projet de type Maven nommé « produit-jersey-web » sous Eclipse
- ✓ Une librairie « **mainStorage.js** » vous a été fournie pour manipuler la partie dynamique (création, Modification, suppression), visualisez son contenu.
- ✓ Un formulaire ainsi qu'un css vous ont été fournis, visualisez les également.
- ✓ Une classe « Client » dans le fichier « **Client.js** » vous a également fournie car nous travaillerons en Objet.

Dans un premier temps nous persisterons par onglet.
Aucune modification n'est à apporter au fichier html pour la question, tout doit se faire dans les 2 fichiers JavaScript !!!

TAF 1 : Création de client

- ✓ Commencer modifier la JS pour qu'elle gère la réinitialisation du formulaire
- ✓ Modifier ensuite la JS afin qu'une action sur le bouton « save » persiste et rafraichisse le tableau situé sur la droite.

TAF 2 : Modification de Client

- ✓ Modifiez la JS afin qu'un client sur une ligne charge le formulaire
- ✓ Faites le nécessaire pour que les modifications soient persistantes et rafraichies dans le tableau

TAF 3 : Suppression d'un client

- ✓ Branchez l'action écoutant un double clic sur chaque ligne du tableau
- ✓ Modifiez la méthode de suppression réelle dans la session
- ✓ Pensez également à rafraichir le tableau

TAF 4 : Persistance pour tout le navigateur

- ✓ Adapter votre projet afin que la persistance soit valable quel que soit l'onglet du navigateur

TP4 : Mise en place d'une architecture REST

L'objectif de cette partie est de pouvoir :

- ✓ Manipuler les Objets JSON (Consommation, Transfert)
- ✓ Faire des appels REST avec Ajax

De manière résumée nous allons refaire les mêmes actions que celles du TP précédent, mais cette fois les appels se feront un serveur exposant du REST.

TAF 1 : Initialisation de la base de données

Pour commencer nous allons installer notre base de données :

- ✓ Lancer le serveur WAMP
- ✓ Lancez « phpmyadmin »
- ✓ Créer une base de données nommée « **ges_client** »
- ✓ Positionnez-vous sur la base de données et importez le fichier « src/test/resources/ges_client.sql »

TAF 2 : Lancement de notre appli

- ✓ Lancez votre serveur en exécutant le « main » de la classe « `JerseyProductApp` »
- ✓ `Visualisez le contenu du fichier de paramétrage de notre webapp (port,nom de la webapp, logging)`
- ✓ `Visualisez les services exposées dans la classe « ClientRestResource »`
- ✓ `Testez votre application en rentrant cette url : http://localhost:8082/eproduit-web/ dans votre navigateur`

TAF 3 : Actions CRUD

- ✓ Remplissez les TODO 1 à 3 pour la création, modification et suppression d'un utilisateur conforme à la démonstration

TAF 4 : En bonus

- ✓ Rajouter l'action permettant d'effectuer une recherche sur le nom

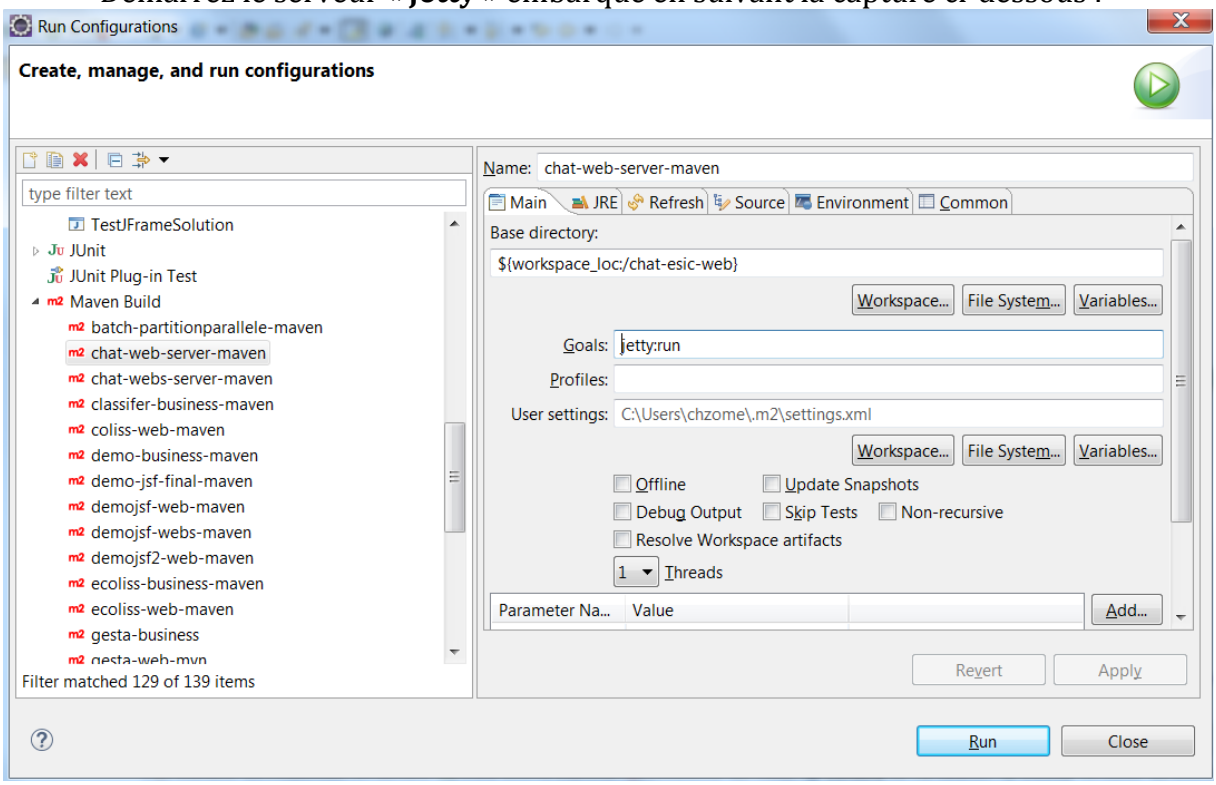
TP 5 : Mise en place du « Chat ESIC »

Dans cette partie nous mettrons en place un serveur de chat en utilisant les notions ci-dessous :

- Web Sockets

TAF 1 : Initialisation du projet

- ✓ Importez le projet « **chat-esic-web** » et visualisez son contenu
- ✓ Démarrez le serveur « **jetty** » embarqué en suivant la capture ci-dessous :



- ✓ Assurez-vous que le port défini dans le pom.xml, et appelé dans le « chat.js » ne sont pas déjà occupés, si oui mettez en un autre.
- ✓ Cliquez sur « Run » et vérifiez votre console (port, web-app)
- ✓ Testez cette URL ➔ http://localhost:8091/chat-web/?loginParam=ZOME

TAF 2 : Echange de messages entre le client et serveur

- ✓ Complétez les « TODO » définis dans « chat.js » afin de reproduire l'attendu présenté en démo.
- ✓ Rajouter l'instruction pour quitter le chat