{<L3/>}
INFORMATIQUE

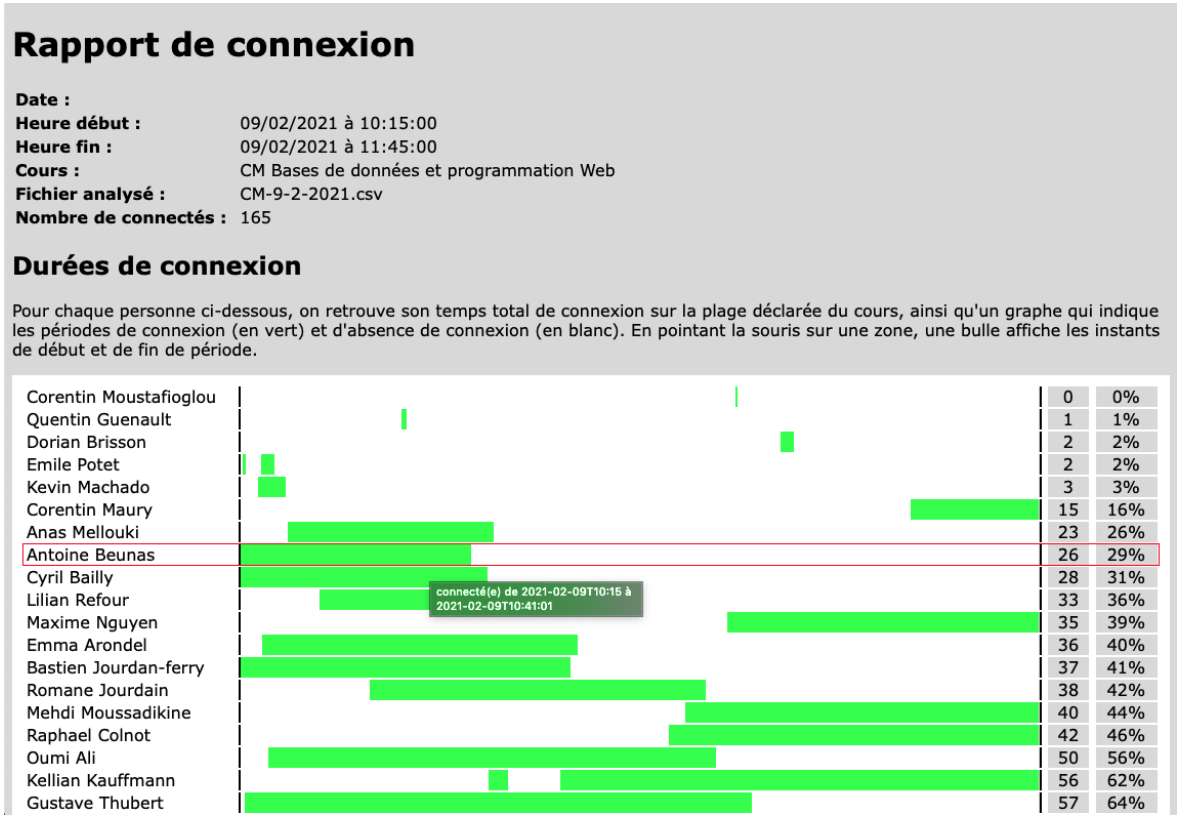
Design Patterns

Projet – 2021

Le projet consiste à faire du refactoring, c'est à dire à réorganiser le code d'un logiciel. Cette réorganisation est guidée par différents besoins, énoncés ci-dessous. Il s'agit cette année de travailler sur un (petit) outil capable d'extraire d'un fichier texte des informations de façon à les visualiser « proprement » en HTML.

Contexte

La plateforme de visioconférence Microsoft TEAMS permet de savoir, lorsqu'on est propriétaire d'une réunion, qui s'est connecté, et quand. On cherche donc à extraire d'un fichier de ce type toutes les informations de façon à faire un planning de connexion. Le code qui est donné réalise cette transformation et affiche, dans la console, le script HTML du résultat. Visualisé par un navigateur cela donne par exemple :



On y voit le pseudo TEAMS, des périodes de connexion (vert) ou d'absence (blanc), le nombre de minutes, et le pourcentage que cela représente par rapport à la durée de la séance. Ce premier prototype a été développé en moins de 6h. Il est donc largement incomplet, un peu fragile sur certains aspects, et il est temps de procéder à un toilettage de l'architecture afin de permettre certaines évolutions ou caractéristiques. Elles sont mentionnées ci-dessous, certaines relèvent de l'utilisation de design patterns, d'autres peut-être pas.

1 – La sortie actuelle est en HTML, sur la console. C'est pas hyper pratique. Il faut trouver ici un moyen d'utiliser un objet chargé, par exemple, d'enregistrer cela dans un fichier HTML. Une autre possibilité serait d'ouvrir un fichier type Excel, et d'aller écrire les informations dans les bonnes cases. Un autre exemple d'objets pourrait être réservé au débogage, en écrivant les détails des structures de données sur la console. **Il y a donc plusieurs possibilités de traitement, qui doivent pouvoir être échangées facilement**.

2 – Les parcours des collections doivent se faire avec un **Iterator (TP5)**

3 – Les données sont, par défaut, triées par durée de connexion croissante. Il sera possible d'avoir d'autres options, par ex. par nom d'utilisateur, ou par identifiant.

4 – Les données peuvent être anonymisées, ie. Les noms, voir les id, ne s'affichent pas. Cela peut se faire à deux niveaux : les données ne sont pas générées, ou bien elles ne sont pas affichées (neutralisation au niveau du CSS). On peut aussi avoir « sans planning » (juste la liste des connectés).

4 – Autant que possible, les créations d'objets devraient reposer sur des **Factory (TP4)** (il y a plusieurs patterns dans cette famille).

5 – Il n'y a pas, pour le moment, à proprement parler, d'interface utilisateur. Bien que celle-ci soit minimale, elle doit permettre, **selon un MVC**, de saisir : le fichier à traiter (en drag n drop), l'heure de début du cours, l'heure de fin (on va considérer que le cours n'est pas à cheval sur 2 jours), l'objet de la réunion. Une fois validé, le fichier est traité. Voir également le scénario type en dernière page.

6 – On espère, d'ici 1 mois, avoir accès à un objet qui donne de façon exacte, pour un pseudo TEAMS, l'identifiant de l'utilisateur, son nom et son prénom. Actuellement il n'y a pas de règle précise pour cela, on n'est donc pas capable, à coup sûr, de connaître le nom d'une personne (alors qu'il suffirait, par exemple, de noter le nom en majuscules et le prénom en minuscules, pour que le problème ne se pose pas).

Pour chacun des points ci-dessus :

* sur papier : choix argumenté de traiter par design pattern ou non. Si oui, diagramme de classes avant (limité à ce qu'il est nécessaire de considérer), choix argumenté du design pattern (si le DP n'est pas imposé), diagramme de classes après. Diagramme de séquences si le fonctionnement n'est pas évident.

* dans code : en cas de DP, mise en place, commenter pour indiquer la présence du DP en faisant référence à votre documentation (idéal, la doc est dans le javadoc, mais c'est plus long…), et vérifier que ça fonctionne.

Modalités :

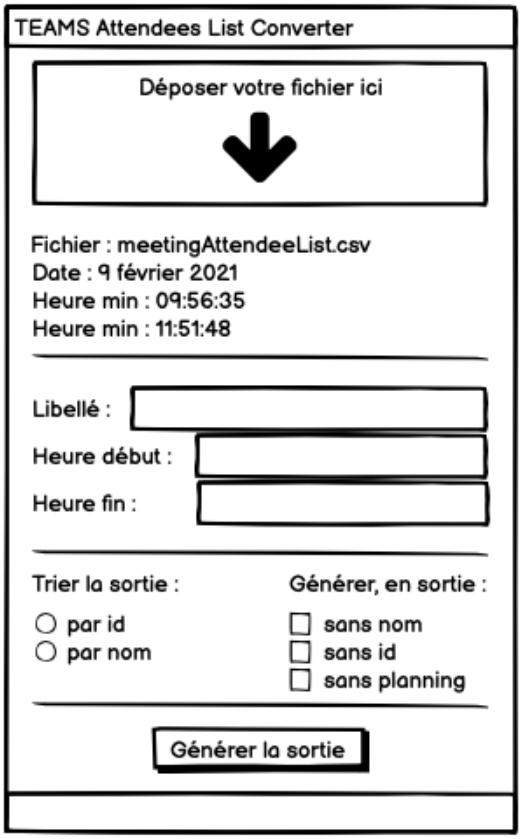
- Groupes de 4 personnes minimum, 6 personnes maximum
- Début : 20 février 2021
- Fin : 30 avril 2021 – Remise hors délai : groupe considéré comme ABI.
- Pas de tests unitaires demandés, mais versioning avec git.
- Remettre le tout sous git (enregistrer l'adresse de clonage http sous Celene), la documentation est dans un dossier bien identifié.
- A la racine du git, un readme.md identifie les auteurs.
- Licence : GPL 3

Scénario utilisateur

Le scénario débute lorsque l'utilisateur exécute le logiciel

1- La fenêtre d'accueil du logiciel apparaît, faisant comprendre à l'utilisateur qu'on attend de lui qu'il choisisse le fichier à traiter (en drag n drop ou en désignation via un sélecteur de fichiers)
2- L'utilisateur désigne le fichier
3- La fenêtre, rappelant le fichier choisi, montre la date et l'amplitude horaire
4- L'utilisateur doit remplir l'objet de la réunion (par ex. CM n°3 de design patterns – L3 Info)
5- L'utilisateur doit définir l'heure de début de la séance, et l'heure de fin (parfois les séances débordent)
6- L'utilisateur règle les options (par ex. sortie HTML anonymisée mais avec identifiant, triée par n°id
7- L'utilisateur valide la fenêtre
8- Un sélecteur de fichiers demande où et comment appeler le fichier de sortie
9- La fenêtre rend compte du résultat : fichiers créés, et où et permet de quitter ou recommencer

Exemple de maquette (non contractuelle ;-))



[réalisé avec balsamiq mockup (https://balsamiq.com/wireframes/)]

Cette vue présente le logiciel, après avoir déposé le fichier. Il a déjà été analysé, et attend la saisie du titre de la réunion (libellé), et les horaires officiels.