Rapport final projet Applications du Web — Candy box

Alexandre RENOUX - Pierre-Emmanuel NOVAC

7 mai 2016

Le site web a été nommé CraftMine, et est disponible à l'adresse `https ://craftmine.piernov.org/`.

1 Fonctionnalités

1.1 Ce qui a été fait

1.1.1 Mine

Développée comme prévu. Le développement a été très rapide et tient en peu de lignes de code.

1.1.2 Guilde de mineurs

Développée comme prévu. Le développement a été très rapide et tient en peu de lignes de code.

1.1.3 Magasin

Développé en majorité comme prévu. Le développement a été plus long et a demandé la rédaction de bien plus de code. Il fait appelle à plus de fonctions pour lire du XML, gérer le stockage dans la session, générer le HTML, gérer l'inventaire, etc...

1.1.4 Donjon

Développé en partie comme prévu. C'est la fonctionnalité qui a été la plus difficile et la plus longue à implémenter, et cela se vérifie sur la longueur du code.

1.1.5 Enregistrement chargement de partie

Développé comme prévu. Le joueur a aussi la possibilité de télécharger/téléverser un fichier de sauvegarde. La génération et la lecture du XML a posé quelques problèmes au début mais finalement les fonctions sont assez concises. Le téléversement n'a pas été facile à implémenter : on trouve beaucoup d'informations contradictoires et la plupart d'entre elles reposent sur jQuery.

1.1.6 Interface utilisateur

Développée en majorité comme prévu. C'est principalement un travail de recherche pour comprendre comment structuer correctement le code HTML de manière à ce que la bibliothèque CSS Bootstrap applique correctement les styles.

1.2 Ce qui n'a pas été fait

Utilisation des objets du magasin. Empêcher le joueur de récupérer l'or de la mine lorqsu'il est dans le donjon

2 Répartition du travail

Les fonctionnalités liées à la mine et à l'enregistrement/chargement de la partie ainsi que l'interface utilisateur ont été développées par Pierre-Emmanuel, qui avait aussi entamé le développement du magasin.
La guilde de mineurs et le donjon ont été développées par Alexandre. Il a aussi poursuivi le développement du magasin.
Le développement s'est fait grâce à l'outil Git, dont le dépôt est consultable à l'adresse `http ://git.piernov.org/candybox/`. Alexandre développer les fonctionnalités qui lui avaient été attribuées sur sa branche alexichi, tandis que Pierre-Emmanuel développer les différentes fonctionnalités en parallèle sur les branches feat/ correspondantes. Il s'occupait aussi de la fusion des branches pour mettre en commun le code développer séparément.

3 Organisation du code

Les fonctions développées en PHP sont généralement concises et découpées de manière logique. En Javascript en revanche, du fait de la génération de code HTML, des traitement plus lourds et des fonctions moins intuitives, le code est moins clair et plus long.

3.1 Javascript

Les différentes fonctions Javascripts viennent modifier et compléter le contenu de la page dynamiquement. Les scripts ont été séparés dans différents fichiers correspondant chacun à un aspect de l'application. Ainsi les fichiers suivant ont été créés :

3.1.1 js/craftmine.js

Ce fichier contient la déclaration du tableau data qui contient des données à propos du joueur utiles côté client.
La fonction sendRequest permet d'envoyer des requêtes POST au serveur. Elle décode par ailleurs les données reçus sous forme de JSON et affiche les messages d'erreurs/informations le cas échéant.
Les fonctions de la mine y sont aussi présentes.
La fonction initCraftMine est exécutée au chargement de la page par init pour récupérer les données de la sessions précédentes et mettre à jour l'affichage en conséquence. init appelle aussi les fonctions pour charger le bon onglet, pour actualiser la liste des sauvegardes et pour incrémenter l'or de la mine.

3.1.2 js/dungeon.js

Des fonctions pour faire évoluer le personnage dans le donjon.

3.1.3 js/gui.js

Ce fichier contient quelques méthodes pour afficher des messages d'informations/erreurs dans les boîtes appropriées.
changeTab() permet de charger l'onglet correspondant à ce qui a été spécifié dans l'URL. Elle réimplémente une fonctionnalité de la bibliothèque Bootstrap mais permet alors de s'affranchir du chargement de Bootstrap et jQuery, qui seraient démesurées pour cette simple fonctionnalité.

3.1.4 js/guild.js

Deux simples fonctions envoyant une requête au serveur pour créer la guilde de mineurs et recruter un mineur.

3.1.5 js/perso.js

Pour mettre à jour les informations du personnage (expérience, niveau, vie).

3.1.6 js/savegame.js

Les différentes fonctions pour gérer les actions des boutons de l'onglet Save. À noter l'utilisation de window.open() pour lancer le téléchargement de la sauvegarde sur le client. Cette fois-ci, les paramètres sont passés en GET pour plus de simplicité.
uploadSave utilise des fonctionnalités récentes de HTML5/XMLHttpRequest2 pour l'envoi de fichier au serveur.

3.1.7 js/shop.js

Des fonctions pour gérer le magasin et l'inventaire, ainsi que les actions des objets.

3.2 PHP

Le fichier craftmine.php gère les requêtes reçues du client et lance les fonctions correspondantes. Le fichier upload.php est dédié à la réception du fichier de sauvegarde envoyé par le client. Les fonctions sont, comme pour les scripts Javascript, répartis dans différents fichiers.

3.2.1 account.inc

Des fonctions pour gérer débiter/créditer de l'or.

3.2.2 craftmine.inc

Répond à la requête d'initCraftMine pour envoyer les données de la session précédente au chargement de la page. Fait appelle à différentes fonctions des autres fichiers en conséquence.

3.2.3 inc/dungeon.inc

3.2.4 inc/guild.inc

Des fonctions de gestion de la guilde de mineurs

3.2.5 inc/Inventory.inc

Une classe qui décrit l'inventaire du personnage, implémenté comme singleton dont la référence est stockée dans la session.

3.2.6 inc/Item.inc

Une classe qui décrit un objet dans l'inventaire ou dans le magasin.

3.2.7 inc/messages.inc

La liste des messages d'information/erreur envoyés par le serveur ainsi que les fonctions utilisées pour le faire.

3.2.8 inc/mine.inc

Des fonctions pour la gestion de la mine.

3.2.9 inc/Monster.inc

Une classe qui décrit un monstre du donjon.

3.2.10 inc/perso.inc

Des fonctions pour gérer les informations (vie, niveau, expérience) du personnage.

3.2.11 inc/savegame.inc

Un ensemble de fonctions pour gérer chacune des requêtes de gestion des sauvegardes. Les sauvegardes sont enregistrées en XML, des fonctions ont été développées pour générer du XML à partir d'un tableau (en prenant en compte les objets pour lesquels on fait appel à leur méthodes addToXML) et générer un tableau à partir du XML. Elles sont aussi utilisées lors du téléversement d'un fichier de sauvegarde, pour le réécrire et ainsi s'assurer qu'il est valide.

3.2.12 inc/shop.inc

Les différentes fonctions pour les interactions avec le magasin. loadShop lit un fichier XML pour générer les objets.

3.3 Autres

3.3.1 index.xhtml

La page XHTML principale auquel le client accède et que le Javascript modifie dynamiquement. Elle est structurée pour pouvoir utiliser Bootsrap. Elle possède un doctype HTML5 bien que cela ne soit pas strictement nécessaire. À la fin du chargement, elle fait appelle à la méthode init.

3.3.2 .htaccess

Pour forcer le serveur à charger la page index.xhtml lorsque le client accède au répertoire.

3.3.3 css/bootstrap.min.css

Le fichier CSS officiel de Bootstrap pour avoir une interface plus jolie.

3.3.4 css/craftmine.css

Un fichier CSS pour quelques personnalisations supplémentaires de l'interface.

3.3.5 css/Symbola.ttf

Police utilisée pour les icônes des objets, fournit par le serveur au cas où le client ne la possède pas.

3.3.6 data/items.xml

La liste des objets du magasin au format XML.

3.3.7 data/monsters.xml

La liste des monstres du donjon au format XML.

4 Choix techniques

4.1 Javascript

Nous avons utilisé des fonctionnalités plutôt récentes en Javascript comme l'objet classList pour ajouter/supprimer une classe en Javascript, utilisé dans notre cas pour changer l'apparence de l'onglet actif. XMLHttpRequest2 a aussi été utilisé pour le téléversement des fichiers de sauvegarde. De ce fait, les navigateurs ancien ou avec une prise en charge limitée du Javascript ne pourront pas faire fonctionner correctement le site web. Ces navigateurs sont entre autres Internet Explorer version 9 et antérieures, ou Opera Mini. Le site web a été développé sous Firefox, et testé sous Chromium.
Aucune bibliothèque Javascript externe n'est utilisée.

4.2 XHTML

Le XHTML a été utilisé pour sa rigueur et ainsi s'éviter des libertés dans le code HTML. Des navigateurs anciens ne le prennent pas en charge et ne pourront pas du tout afficher la page.

4.3 Bootstrap

La bibliothèque CSS Bootstrap est utilisé pour avoir un rendu visuel correct sans trop d'effort. Elle fournit en effet des styles d'éléments qui conviennent et très peu de propriétés ont dû être ajoutées. Nous n'utilisons pas la bibliothèque Javascript de Bootstrap.

4.4 JSON

Le format de données JSON est utilisé pour transmettre des informations du serveur au client. Celui-ci est en effet très simple à utiliser et des méthodes sont présentes à la fois en Javascript et en PHP nativement pour encoder/décoder des données sous format JSON.

4.5 XML

Les sauvegardes, les objets du magasin et les monstres sont enregistrés au format XML pour être facilement modifiable. Il aurait été intéressant d'enregistrer d'autres données comme les bâtiments sous ce même format et ainsi rendre le site web plus extensible.

4.6 PHP

Le langage PHP est utilisé côté serveur car très adapté au développement de site web. Nous ne l'utilisons en revanche pas pour générer des pages HTML mais simplement pour traiter les requêtes envoyées du client, les données étant retournées au format JSON.
Certaines fonctionnalités comme l'inventaire et ses objets font appel à de la programmation orienté objet. Ceci n'était en aucun cas nécessaire mais a été fait simplement pour expérimenter les possibilités de PHP. Les autres fonctionnalités n'ont pas été développés en programmation orienté objet.

5 Planification

5.1 Planification théorique

- 1ère semaine : développement des fonctionnalités de la mine, de la guilde de mineur et du magasin.
- 2e semaine : développement du donjon et de l'enregistrement de la partie au format XML.
- 3e semaine : amélioration de l'interface utilisateur et finalisation du code.

5.2 Planification effective

5.2.1 Semaine 1

Les fonctionnalités de la mine et de la guilde des mineurs ont été correctement développées.
Le magasin n'a pas été finalisé dans les temps, mais il était en partie fonctionnel.
Le développement de l'interface utilisateur a été attaquée bien qu'à l'origine prévue pour la fin : le développement du magasin est en partie lié à son affichage, et il nous trouvions dommage d'avoir une interface repoussante jusqu'à la fin...

5.2.2 Semaine 2

Le magasin a été finalisé. Le chargement des objets à partir d'un fichier XML a été plus long que prévu.
Les fonctionnalités d'enregistrement au format XML et de chargement de la partie ont été correctement développées, mais les fonctions pour créer le XML à partir du tableau session et charger le XML dans le tableau session ont pris du temps à être réalisées, retardant le développement des autres fonctionnalités.
Du retard a été pris sur le développement du donjon qui n'a été qu'en partie développé et n'est pas encore fonctionnel.

5.2.3 Semaine 3

L'interface utilisateur utilise désormais correctement Bootstrap 3 et semble finalisée.
La possibilité de téléverser un fichier de sauvegarde a été ajoutée.
Le donjon n'est pas totalement terminé mais est fonctionnel.
Le code a été en partie relu, mais n'a pas été testé en profondeur.