Cégep du Vieux Montréal

2022-09-02

Une image contenant équipement électronique, circuit

Description générée automatiquement

Jeu du carré rouge

Documentation du projet

Réalisé par

Théo Luquet & Justin Mayer

Présenté à

Godefroy Borduas

Dans le cadre du cours

C31 - GÉNIE LOGICIEL I : CONCEPTION ET GESTION | 420\_IN - A22

Table des matières

[Méthodes SCRUM](#_Toc113194806)

[Analyse et modélisation](#_Toc113194807)

[Implantation par sprints successifs](#_Toc113194808)

[Analyse textuelle](#_Toc113194809)

[Cas d’usage et Scénarios d’utilisation, ainsi que les rôles](#_Toc113194810)

[Maquettes des interfaces graphiques](#_Toc113194811)

[CRC](#_Toc113194812)

[Modèle de données](#_Toc113194813)

[Tables et/ou fichiers de données](#_Toc113194814)

[Planning global](#_Toc113194815)

[Sprint 1](#_Toc113194816)

# Méthodes SCRUM

## Analyse et modélisation

|  |  |
| --- | --- |
| **Obtention d’un mandat** | Un mandat provient d’un **client** (externe, interne ou vous-même) |
| **Analyse textuelle** | Son but est d’obliger une lecture formelle du document |
| **Identification des cas d'usage et des rôles** | Ce que logiciel doit permettre de faire, du point de vue du client |
| **Scénarios d’utilisation** | La modélisation spécifiant séquentiellement dans le temps toutes les étapes pour réaliser un cas d’usage |
| **Maquettes graphiques** | Le design visuel des diverses pages-écrans (fenêtres) par lesquelles l’usager manipulera le logiciel |
| **Définition des classes et méthodes** | L’identification des classes permettant d’instantier les objets chargés d’effectuer le traitement dans le programme |
| **Modélisation des données** | La définition des tables de données (ou autres modèles au besoin) dans lesquelles les informations devront être conservées |
| **Planification globale**  *Incluant le découpage en sprints* | C’est le cadre temporel dans lequel on précise l’ordre de réalisation des classes, base de données, et autres composantes faisant partie du logiciel à créer |

## Implantation par sprints successifs

Un sprint est une période courte de production d’où doit sortir une partie du logiciel, le livrable.

|  |  |
| --- | --- |
| **Planification détaillée** | À partir de la section du sprint courant dans la Planification globale, on attribue les tâches à des membres de l’équipe devant les réaliser.  Chaque membre est responsable de préciser ses tâches en sous-tâches afin d’évaluer le temps requis à leur réalisation |
| **SCRUMs quotidiens** | Les scrums sont des réunions d’équipe quotidiennes où chaque membre doit indiquer les tâches complétées, les tâches à réaliser, et les problèmes rencontrés.  Cette réunion doit se faire avec toute l’équipe afin d’assurer que tous comprennent comment le projet avance |
| **Debriefing / Ajustement** | Réunion de fin de sprint permettant d’évaluer comment le projet avance ou retarde, afin d’apporter les ajustements requis au prochain sprint |
| **Livraison partielle** | Le logiciel doit être utilisable pour faire partie d’une livraison |

# Analyse textuelle

L'analyse textuelle a pour but de forcer une lecture optimale du mandat du client. L'exercice consiste essentiellement à identifier les mots et expressions pertinents au logiciel à réaliser.

Comme la plupart des mandats sont écris en langage ordinaire, on se concentre sur les noms des choses, les verbes qui dénotent des actions de ces choses, et sur les adjectifs qui les qualifient.

On ne répète pas inutilement les mots rencontrés plusieurs fois, on ne fait **pas de phrases**, les colonnes n'ont pas de lien l'une avec l'autre. On peut identifier des "expressions", tels "adresse de courriel", ou "envoyer courriel".

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Noms (objets)** | **Verbes (actions)** | **Adjectifs (attributs)** |
| **Explicites**  *(dans le texte)* | Carré  Cadre  Rectangles  Pion  Démarrage  Pointeur  Souris  Espace  Bordure  Vitesse  Contact  Fin  Éléments  Clic  Partie  Session  Informations  Menu  Logiciel  Résultats  Nom  Fichier, disque  Scores  Aire de jeu  Limite | Représente  Empêcher  Touche  Suivre  Déborder  Se déplacent  Rebondissent  Tenir  Affiche  Replace  Redémarrer  Débute  Appartient  Cumule  Réinitialiser  Sauvegarder  Requérir  Effacer  Modifier | Rouge  Bordé de noir  Bleus  Positionnés  Blanc  Noire  Constante  Diagonale  Extérieur  Le plus longtemps  Mort  Différentes (parties)  Déroulant  Fournissant  Anciens  Enfoncée  Variables  Large  Supérieur / Inférieur  Gauche / Droit  Au centre  Accélérant  Progressivement |
| **Implicites**  *(requis par ce qui est explicite, mais jamais mentionné)* | Collision |  | Performant |
| **Supplémentaires**  *(suggestions pour le mandataire de choses possibles mais peut-être ignorées)* | Difficulté  Angles  Powerups | Diffèrent  Ralentir  Stopper | Facile – Moyen – Difficile  Progressif  Invisible  Insensible (à la bordure) |

# Cas d’usage et Scénarios d’utilisation, ainsi que les rôles

Identification des cas d'usage, indication des rôles concernés par ceux-ci (à qui ils vont s'appliquer : président, gestionnaire, vendeur, commis, etc), et précision des détails d'implantation de la dynamique de chaque cas d'usage dans un scénario d'utilisation, à l’égard des commandes de l'humain et du traitement opéré par l'ordinateur en réponse.

**Une seule entrée par ligne** et **jamais** dans les deux colonnes sur la même ligne; du haut vers le bas.

C'est la **représentation temporelles des actions** (soit de l'humain, soit de l'ordinateur) qui compte.

Un cas d'usage peut référer à un autre cas d'usage; ce dernier doit alors expliquer en détail cette partie de la tâche (par exemple la tâche d'identification de l'utilisateur (login).

Utiliser le IF pour des sections conditionnelles, et le FOR pour des boucles.

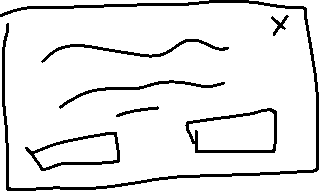
|  |  |  |  |
| --- | --- | --- | --- |
| **RÔLE**  *Profil d’utilisateur* | **CAS D’USAGE**  *Tâche complète* | **HUMAIN**  *Manipulations humaines dans l’interface* | **MACHINE**  *Opérations effectuées par l’ordinateur* |
| Joueur | **Lancer le logiciel** | Démarre le logiciel |  |
|  |  |  | Affiche l’aire de jeu à l’arrêt  Affiche un champ de texte pour le nom du joueur  Affiche le bouton pour débuter une nouvelle session (Option 2)  Affiche le bouton pour afficher les anciens scores (Option 3)  Affiche le bouton pour quitter (Option 4) |
| Joueur | **Lancer une partie (Option 1)** | Maintient le clic sur le carré rouge |  |
|  |  |  | Les rectangles bleus entrent en mouvement  Le carré rouge suit le curseur de la souris |
| Joueur | **Jouer** | Bouge la souris en  maintenant le clic sur le carré rouge |  |
|  |  |  | Les rectangles bleus rebondissent uniquement sur le bord extérieur  La partie se finit si le carré rouge touche un rectangle bleu ou le bord intérieur |
| Joueur | **Fin de partie** |  | Affiche le score du joueur |
| Joueur | **Débuter une nouvelle session (Option 2)** | Clique sur le bouton dédié |  |
|  |  |  | Enregistre la session et le nom du joueur (si précisé préalablement)  Réinitialise le logiciel |
| Joueur | **Afficher les scores (Option 3)** | Clique sur le bouton dédié |  |
|  |  |  | Affiche les anciens scores |
| Joueur | **Quitter (Option 3)** | Clique sur le bouton dédié |  |
|  |  |  | Enregistre la session et le nom du joueur (si précisé préalablement)  Termine le processus du logiciel |

# Maquettes des interfaces graphiques

Maquettes graphiques représentant les pages-écrans du logiciel prévu. Cela permet d'identifier des ensembles fonctionnels de widgets offerts à l'utilisateur.

Regroupement conceptuel, plutôt qu'agencement esthétique, pour l'instant. On parle donc d’**ergonomie** du logiciel.

# CRC



Identification des classes d'objets à programmer qui vont remplir les attentes du client, telles qu'identifiées par les tâches inscrites dans les scénarios d'utilisation provenant de l'analyse textuelle, selon les cas d'usage. Ces tâches deviennent ici des méthodes d'objet (*des fonctions*) : un objet est responsable des tâches qu'il peut effectuer. La collaboration est un lien vers des objets qui seront en relation avec le présent, et donc qui nous orientera vers son propriétaire pour peaufiner le code de cette relation.

|  |  |
| --- | --- |
| **Classe : Logiciel** | **Collaboration** |
| Propriétaire : |
| **Responsabilités :**   * Initialiser la Vue ainsi que le Jeu | * Vue * Jeu |

|  |  |
| --- | --- |
| **Classe : Vue** | **Collaboration** |
| Propriétaire : |
| **Responsabilités :**   * Afficher les menus * Afficher le jeu * Afficher les scores | * Jeu |

|  |  |
| --- | --- |
| **Classe : Jeu** | **Collaboration** |
| Propriétaire : Justin & Théo |
| **Responsabilités :**   * Créer l'aire de jeu * Placer les rectangles bleu et le carré rouge * Gérer les éléments de jeu selon leurs règles : Déplacement du carré rouge selon le curseur de la souris Déplacement des rectangles bleus Collision des rectangles bleus contre la bordure extérieure Collision du carré rouge contre la bordure intérieure Collision du carré rouge contre les rectangles * Capter l'input du joueur pour les menus ainsi que lors du jeu * Actualiser l’état de jeu * Stocker le score de la partie en cours * Stocker les parties dans la session | * Logiciel * Vue * Carré Rouge * Rectangle Bleu * Partie * Sauvegarde |

|  |  |
| --- | --- |
| **Classe : Carré Rouge** | **Collaboration** |
| Propriétaire : |
| **Responsabilités :**   * Stocker les dimensions de l’élément * Stocker la position de l'élément * Stocker état de l'élément (vivant ou mort) | * Jeu |

|  |  |
| --- | --- |
| **Classe : Rectangle Bleu** | **Collaboration** |
| Propriétaire : |
| **Responsabilités :**   * Stocker les dimensions de l’élément * Stocker la position de l'élément | * Jeu |

|  |  |
| --- | --- |
| **Classe : Partie** | **Collaboration** |
| Propriétaire : |
| **Responsabilités :**   * Stocker le nom du joueur * Stocker le score final d’une partie * Stocker la date de la partie | * Jeu |

|  |  |
| --- | --- |
| **Classe : Scores** | **Collaboration** |
| Propriétaire : |
| **Responsabilités :**   * Enregistrement des parties dans le fichier CSV * Lecture des parties dans le fichier CSV | * Jeu |

# Modèle de données

On établit notre diagramme de données. Fichiers plats, BD et SGBD, Big Data, etc. On indique comment nos données sont préservées.

## Tables et/ou fichiers de données

Fichier CSV (comma-separated values) :

|  |  |  |
| --- | --- | --- |
| **nom,** | **score,** | **date** |
|  |  |  |

# Planning global

Cette section sert à préciser les scénarios d’utilisation qui seront développés lors des sprints : vous devez également les ordonner afin de produire les choses plus importantes le plus vite possible (afin notamment de réaligner le tir au besoin).

La liste des tâches doit représenter l'ensemble des éléments à programmer tel qu'indiqué dans les CRCs et la définition des modèles de données.

Placer les lignes de démarcation des sprints, par couleurs ou autre.

**Membres de l’équipe :**

|  |  |  |  |
| --- | --- | --- | --- |
| **Fonctionnalité** | **Priorité** | **Temps prévu** | **Sprint visé** |
| Créer l'aire de jeu | Haute | 30min | 1 |
| Afficher le jeu | Haute | 30min | 1 |
| Classe Carré Rouge | Haute | 5min | 1 |
| Classe Rectangle Bleu | Haute | 5min | 1 |
| Placer les rectangles bleu et le carré rouge | Haute | 20min | 1 |
| Capter l'input du joueur pour les menus ainsi que lors du jeu | Haute | 30min | 1 |
| Déplacement du carré rouge selon le curseur de la souris | Haute | 30min | 1 |
| Déplacement des rectangles bleus | Haute | 20min | 1 |
| Actualiser l’état de jeu | Haute | 30min | 1 |
| Collision des rectangles bleus contre la bordure extérieure | Haute | 20min | 1 |
| Collision du carré rouge contre la bordure intérieure | Haute | 20min | 1 |
| Collision du carré rouge contre les rectangles | Haute | 20min | 1 |
| Stocker le score de la partie en cours | Moyen | 5min | 2 |
| Afficher les menus | Moyen | 20min | 2 |
| Classe Partie | Moyen | 5min | 2 |
| Stocker les parties dans la session | Moyen | 20min | 2 |
| Enregistrement les parties dans le fichier CSV | Moyen | 10min | 2 |
| Lecture des parties dans le fichier CSV | Moyen | 5min | 2 |
| Afficher les scores | Moyen | 30min | 2 |
| Classe Logiciel | Moyen | 1min | 2 |

# Sprint 1

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Planification détaillée** | | | **Gestion des scrums** | | | | | | | | |
| **Étapes**  **(fonctionnalités haut niveau)**  *Litéralement faire un copier-coller à partir de la planification globale* | **Sous-étapes**  *Découper l'étape en sous-tâches pour mieux indiquer le moment du livrable* | **Responsable**  *Un membre de l’équipe* | **Semaine 1** | | | **Semaine 2** | | | **Semaine 3** | | |
| *Prévu* | *Fait* | *Problème* | *Prévu* | *Fait* | *Problème* | *Prévu* | *Fait* | *Problème* |
| Créer l'aire de jeu | * Créer la bordure extérieure * Créer la bordure intérieure |  | 09/09/22 |  |  |  |  |  |  |  |  |
| Afficher le jeu |  |  | 09/09/22 |  |  |  |  |  |  |  |  |
| Classe Carré Rouge | * Donner des dimensions * Donner une position |  | 09/09/22 |  |  |  |  |  |  |  |  |
| Classe Rectangle Bleu | * Donner des dimensions * Donner une position |  | 09/09/22 |  |  |  |  |  |  |  |  |
| Placer les rectangles bleu et le carré rouge |  |  | 09/09/22 |  |  |  |  |  |  |  |  |
| Capter l’input du joueur pour les menus ainsi que lors du jeu | * Récupérer l’input * L’appliquer au jeu |  |  |  |  |  |  |  |  |  |  |
| Déplacement du carré rouge selon le curseur de la souris | * Relier le carré rouge au curseur |  |  |  |  |  |  |  |  |  |  |
| Déplacement des rectangles bleus | * Définir la vélocité * Définir la direction |  |  |  |  |  |  |  |  |  |  |
| Actualiser l’état de jeu | * Créer une boucle * Définir l’intervalle |  |  |  |  |  |  |  |  |  |  |
| Collision des rectangles bleus contre la bordure extérieure | * Détection de la collision * Changement de direction |  |  |  |  |  |  |  |  |  |  |
| Collision du carré rouge contre la bordure intérieure | * Détection de la collision * Changement de l’état |  |  |  |  |  |  |  |  |  |  |
| Collision du carré rouge contre les rectangles | * Détection de la collision * Changement de l’état |  |  |  |  |  |  |  |  |  |  |