**« Backlog » de sprint #002**

Produit : Astéria

Conçu par : EquBolduc

## Équipe :

|  |  |  |
| --- | --- | --- |
| **Nom** | **Initiales** | **Responsabilité** |
| Émile Grégoire | É. G. | Chef d’équipe |
| Jonathan Samson | J. S. | Responsable des livrables |
| Simon-Pierre Deschênes | S.-P. D. | Responsable de la qualité |
| Jérémie Bolduc | J. B. | Responsable des réunions |

## Échéancier:

Du 25 février au 11 mars 2015.

## Légende :

* Vert, indique que ces items sont réalisés.
* Jaune, indique que ces items sont en cours de réalisation.
* Rouge, problème ou questionnement important qui demande une rencontre d’équipe.
* Aucune couleur indique que ces items ne sont pas encore faits ou commencés, **on peut toujours les enrichir mais il faut le consentement de toute l’équipe**.

## « Backlog » de sprint

|  |  |
| --- | --- |
| **Scénario #8** | |
| Acteur | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux pouvoir mettre le jeu en pause afin de pouvoir réfléchir. |
| Description | . Mettre l’horloge interne sur pause.   * 1. Qui et temps      1. J.B.      2. 30 min.   2. Préconditions   3. Règles d’affaires      1. Arrêter d’appeler la méthode «update»   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. La physique s’arrête dans le jeu   6. Post-conditions   2. Créer un menu pause.   * 1. Qui et temps      1. J.B.      2. 1 h.   2. Préconditions   3. Règles d’affaires      1. Ajouter un pane dans la vueMenu.      2. Ajouter des boutons dans le menu      3. Afficher le pane quand le jeu est en pause.   4. Règles d’affaires alternatives   5. Tests d’acceptation   6. Post-conditions   3. Mettre le jeu en pause.   * 1. Qui et temps      1. J.B.      2. 30 min.   2. Préconditions   3. Règles d’affaires      1. Créer un écouteur sur la touche «p»      2. Mettre le jeu sur pause.   4. Règles d’affaires alternatives   5. Tests d’acceptation   Post-conditions |
| Tests d’acceptation | Mettre le jeu sur pause et vérifier que le moteur physique est arrêté. |
| Complexité | 2 |
| Effort | 2 |
| Commentaires |  |
| **Scénario #9** | |
| Acteur | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux que le vaisseau ne puisse pas passer au travers des planètes. |
| Description | Récupérer tous les corps.   * 1. Qui et temps      1. S.-P. D.      2. 20 min.   2. Préconditions      1. Le moteur physique doit être fonctionnel.      2. Une liste de tous les corps physiques est disponible dans le contrôleur principal.   3. Règles d’affaires      1. À chaque frame, récupérer tous les corps physiques à partir du contrôleur principal.      2. Vérifier qu’ils héritent de l’interface Corps.   4. Règles d’affaires alternatives      1. Si aucun corps n’existe, mettre fin aux calculs.   5. Tests d’acceptation      1. Imprimer dans un syso tous les corps récupérés et vérifier manuellement.   6. Post-conditions      1. Les corps à calculer sont déterminés.   Vérifier s’il y a des collisions.   * 1. Qui et temps      1. S.-P. D.      2. 30 min.   2. Préconditions      1. Tous les corps pouvant entrer en collision doivent avoir été recueillis.   3. Règles d’affaires      1. Vérifier que tous les corps ne sont pas en collision en se fiant à leur position et à leur grosseur.   4. Règles d’affaires alternatives      1. Si aucun corps n’existe, mettre fin aux calculs.   5. Tests d’acceptation      1. Imprimer dans un syso tous les corps ayant entré en collision et vérifier manuellement grâce à leur position et à leur grosseur.   6. Post-conditions   Faire réagir les corps aux collisions.   * 1. Qui et temps      1. S.-P. D.      2. 30 min.   2. Préconditions      1. Les collisions doivent avoir été détectées.   3. Règles d’affaires      1. Faire baisser le niveau de santé du vaisseau.      2. Donner une vitesse en sens opposé aux objets ayant entré en collision.   4. Règles d’affaires alternatives      1. S’il n’y a aucune collision, mettre fin aux calculs.   5. Tests d’acceptation      1. Mettre plusieurs corps à proximité et vérifier que les collisions provoquent bien un changement de direction.   6. Post-conditions |
| Tests d’acceptation | Mettre plusieurs corps à proximité et vérifier que les collisions ont bien lieu et s’effectuent normalement. |
| Complexité | 3 |
| Effort | 3 |
| Commentaires |  |

|  |  |
| --- | --- |
| **Scénario #11** | |
| Acteur | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux que la caméra suive mon vaisseau. |
| Description | Créer une classe Caméra permettant de sauvegarder l’état de la caméra.   * 1. Qui et temps      1. É. G.      2. 1h   2. Préconditions   3. Règles d’affaires      1. Créer les méthodes pour déplacer la caméra.      2. Créer les méthodes pour zoomer.      3. Créer des méthodes qui transforment des coordonnées de l’écran en coordonnées dans l’espace.      4. Implémenter une transition vers les différentes positions.   4. Règles d’affaires alternatives      1. S’assurer que les positions et facteur d’agrandissement sont valides.   5. Tests d’acceptation      1. Créer des tests unitaires.   6. Post-conditions      1. Une classe caméra permettant de changer la vue est disponible.   Utiliser la caméra dans la vue.   * 1. Qui et temps      1. É. G.      2. 45min.   2. Préconditions      1. La classe caméra doit exister.   3. Règles d’affaires      1. Créer une instance de caméra.      2. Regrouper tous les objets visuels dans un groupe.      3. À chaque frame, transformer le groupe selon ce que l’objet Caméra retourne.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. Vérifier visuellement que les changements ont lieu.   6. Post-conditions      1. La vue répond aux changements de la caméra.   Contrôler la caméra à l’aide de la souris.   * 1. Qui et temps      1. É. G.      2. 30min.   2. Préconditions      1. La vue doit répondre aux changements de la caméra.   3. Règles d’affaires      1. Ajouter des écouteurs sur la souris.      2. Selon la position du clique, déplacer la caméra.      3. Selon les mouvements de la molette, zoomer la caméra.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. Vérifier visuellement que la vue répond à la souris.   6. Post-conditions      1. La vue répond aux changements de caméra faits par la souris. |
| Tests d’acceptation | Vérifier visuellement que la caméra se déplace |
| Complexité | 3 |
| Effort | 3 |
| Commentaires |  |

|  |  |
| --- | --- |
| **Scénario #12** | |
| Acteur | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux diriger mon vaisseau avec le clavier. |
| Description | Tourner le vaisseau.   * 1. Qui et temps      1. J.B.      2. 30 min.   2. Préconditions      1. Le vaisseau doit s’afficher      2. La physique doit être fonctionnelle   3. Règles d’affaires      1. Créé les écouteurs des touches «a», «d», «gauche», «droite».      2. Créé les méthodes « Tourner » dans la classe vaisseau.      3. Appeler les méthodes « Tourner » dans les écouteurs.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. S’assurer que le vaisseau tourne dans le jeu   6. Post-conditions   Appliquer une force avec les réacteurs.   * 1. Qui et temps      1. J.B.      2. 30 min.   2. Préconditions      1. Le vaisseau doit s’afficher      2. La physique doit être fonctionnelle   3. Règles d’affaires      1. Créé un écouteur sur la touche «espace».      2. Compléter la méthode force extérieure.      3. Appeler la méthode force extérieure dans l’écouteur.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. S’assurer que le vaisseau est contrôlable dans le jeu   6. Post-conditions |
| Tests d’acceptation | S’assurer que le vaisseau tourne et qu’il est contrôlable dans le jeu |
| Complexité | 1 |
| Effort | 2 |
| Commentaires |  |

|  |  |
| --- | --- |
|  |  |

|  |  |
| --- | --- |
| **Scénario #10** | |
| Acteur | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux pouvoir créer mes propres niveaux. |
| Description | Faire le fichier FXML   * 1. Qui et temps      1. Jonathan Samson      2. 1 heure   2. Préconditions   1.2.1 Avoir une vue existante pour pouvoir éditer le fichier fxml.   * 1. Règles d’affaires   1.3.1 Ajouter une barre qui est présente dans la fenêtre principale du jeu.  1.3.2 Faire en sorte que cette barre d’outils ne prenne pas trop d’espace. (ergonomie)   * 1. Règles d’affaires alternatives   2. Tests d’acceptation   1.5.1 Présence de la barre d’outils intégrée au fichier FXML.   * 1. Post-conditions   1.6.1 Nous pouvons maintenant travailler avec nos objets spatiaux sur mesure.  Ajouter des objets spatiaux dans l’espace de jeu.   * 1. Qui et temps      1. Jonathan Samson et Simon-Pierre D.      2. 4 heures   2. Préconditions   1.2.1 Avoir une barre d’outils fonctionnelle avec laquelle on peut voir les différents objets disponibles.   * 1. Règles d’affaires   1.3.1 Sélectionner un objet disponible dans la barre d’outils.  1.3.2 Placer l’objet dans l’espace de jeu.  1.3.3 Faire en sorte que ces objets réagissent bien avec la vue du jeu (zoom)   * 1. Règles d’affaires alternatives   2. Tests d’acceptation   1.11.1 Les objets ajoutés sont inclus dans les calculs de la physique sur le vaisseau.   * 1. Post-conditions   2. 1.6.1 Nous pouvons maintenant jouer avec de nouveaux niveaux. |
| Tests d’acceptation | Être en mesure d’ajouter des planètes dans l’espace de jeu et qu’elles fonctionnent dans les lois de la physique. Notes : Ces nouvelles planètes ne seront pas customizables, ce sera dans un prochain sprint. |
| Complexité | 5 |
| Effort | 4 |
| Commentaires |  |