**« Backlog » de sprint #004**

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 1er avril au 15 avril 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 #23** | |
| Acteurs | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux que mon vaisseau se déplace en consommant du carburant. |
| Description | 1. Créer la propriété du carburant maximal et du carburant restant.    1. Qui et temps       1. J.B.       2. 30 min    2. Préconditions       1. Aucune    3. Règles d’affaires       1. Créer une propriété pour le carburant maximum.       2. Créer une propriété pour le carburant restant.       3. Créer une constante pour le carburant maximum par défaut qui vaudra 10.       4. Créer une constante pour le carburant restant qui sera la même chose que le carburant par défaut.    4. Règles d’affaires alternatives    5. Tests d’acceptation    6. Post-conditions 2. Créer une barre de progression qui affichera la quantité de carburant restante.    1. Qui et temps       1. J.B.       2. 30 min    2. Préconditions       1. Les propriétés du carburant sont créées.    3. Règles d’affaires       1. Créer la barre de progression avec SceneBuilder.       2. Utiliser le data binding pour connecter la propriété du carburant restant et le progrès de la barre de progression.       3. Utiliser le data binding pour connecter la propriété du maximum de carburant et la grandeur de la barre de progression.    4. Règles d’affaires alternatives    5. Tests d’acceptation    6. Post-conditions 3. Rendre le carburant modifiable dans l’éditeur de niveau.    1. Qui et temps       1. J.B.       2. 1h    2. Préconditions       1. Les propriétés du carburant sont créées.    3. Règles d’affaires       1. Créer un label et une textBox pour le carburant maximum et les placer dans la vbox exclusivement visible quand un vaisseau est sélectionné.       2. Créer un label et une textBox pour le carburant restant et les placer dans la vbox exclusivement visible quand un vaisseau est sélectionné.       3. Placer des écouteurs sur les textBox.    4. Règles d’affaires alternatives    5. Tests d’acceptation    6. Post-conditions 4. Faire en sorte que le vaisseau dépense son carburant.    1. Qui et temps       1. J.B.       2. 30 min    2. Préconditions       1. Les propriétés du carburant sont créées.    3. Règles d’affaires       1. Faire diminuer le carburant du vaisseau dans le moteur physique. Faire en sorte que le carburant du vaisseau diminue d’un litre par seconde.       2. Vérifier qu’il reste du carburant dans le vaisseau quand l’utilisateur appuis sur espace pour avancer.    4. Règles d’affaires alternatives    5. Tests d’acceptation 5. Post-conditions |
| Tests d’acceptation | Créer un vaisseau, modifier son carburant maximum et restant, sauvegarder le niveau et vérifier qu’il apparait bien dans le jeu. S’assurer que le vaisseau dépense du carburant quand les moteurs sont allumés. |
| Complexité | 1 |
| Effort | 2 |
| Commentaires |  |
| **Scénario #24** | |
| Acteurs | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux que mon vaisseau perde de la solidité après les collisions jusqu’à ce qu’elle tombe à 0. |
| Description | 1. Créer la propriété de la solidité.    1. Qui et temps       1. J.B.       2. 30 min    2. Préconditions       1. Aucune    3. Règles d’affaires       1. Créer une propriété pour la solidité.       2. Créer une constante pour la solidité par défaut qui sera 1.    4. Règles d’affaires alternatives    5. Tests d’acceptation    6. Post-conditions 2. Créer une barre de progression qui affichera la solidité restante.    1. Qui et temps       1. J.B.       2. 30 min    2. Préconditions       1. La propriété de la solidité est créée.    3. Règles d’affaires       1. Créer la barre de progression avec SceneBuilder.       2. Utiliser le data binding pour connecter la propriété de la solidité et le progrès de la barre de progression.       3. Mettre la valeur maximale de la barre de progression à 1.    4. Règles d’affaires alternatives    5. Tests d’acceptation    6. Post-conditions 3. Faire en sorte que le vaisseau perde de la solidité lors d’une collision.    1. Qui et temps       1. J.B.       2. 30 min    2. Préconditions       1. La propriété de la solidité est créée.    3. Règles d’affaires       1. Faire diminuer la solidité lors d’une collision avec un autre vaisseau.       2. Mettre la solidité à 0 lorsque le vaisseau entre en collision avec une autre planète.    4. Règles d’affaires alternatives    5. Tests d’acceptation 4. Post-conditions |
| Tests d’acceptation | S’assurer que la partie se termine lorsque le vaisseau entre en collision avec une planète et que la santé diminue lorsqu’il entre en collision avec un autre vaisseau. |
| Complexité | 1 |
| Effort | 1 |
| Commentaires |  |
| **Scénario #25** | |
| Acteurs | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux que mon vaisseau ait une vitesse initiale et que la direction soit décidée par le joueur. |
| Description | 1. Obtenir le vecteur initial du joueur.   * 1. Qui et temps      1. É.G.      2. 60 min.   2. Préconditions      1. Il faut pouvoir charger un niveau.   3. Règles d’affaires      1. Mettre l’horloge interne en pause lors du chargement du niveau.      2. Écouter la souris jusqu’à ce que le joueur clique.      3. Lors du clique, obtenir le point dans l’espace (méthode Camera::localToGlobal()) et calculer un vecteur de direction.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. La physique s’arrête au chargement du jeu.      2. Le vecteur de direction est correctement calculé.   6. Post-conditions      1. Le vecteur direction voulue par l’utilisateur est obtenu.   2. Appliquer la vitesse initiale au vaisseau   * 1. Qui et temps      1. É.G.      2. 20 min.   2. Préconditions      1. Avoir le vecteur de direction.   3. Règles d’affaires      1. Obtenir le vecteur de direction.      2. Ajouter la vitesse de départ (obtenue dans la classe niveau) au vaisseau.      3. Démarrer l’horloge interne du jeu.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. Vérifier que la vitesse est ajoutée.   6. Post-conditions      1. L’utilisateur peut ajouter une vitesse initiale au vaisseau. |
| Tests d’acceptation | Charger un tableau, puis, à l’aide de la souris, choisir un angle de départ. Vérifier que l’angle est respecté et que le vaisseau obtient une vitesse initiale. |
| Complexité | 3 |
| Effort | 2 |
| Commentaires |  |
| **Scénario #27** | |
| Acteurs | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux que les objectifs soient visibles. |
| Description | 1. Créer l’objet dessinable   * 1. Qui et temps      1. É.G.      2. 60 min.   2. Préconditions      1. Il faut que le niveau ait un objectif.   3. Règles d’affaires      1. Créer une image contenant un portal.      2. Créer une classe héritant de l’interface Dessinable.      3. Associé un ObjectifRayon à cet objet.      4. Dans la méthode getNoeud, retourné un Group contenant un ImageView de l’image précédemment créée.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. Vérifier que la nouvelle classe retourne bien le nœud désiré.   6. Post-conditions      1. Une classe dessinable retourne un Group contenant un portal.   2. Ajouter le portal dans la scène lors du chargement d’un niveau.   * 1. Qui et temps      1. É.G.      2. 30 min.   2. Préconditions      1. Avoir une classe représentant l’objectif.   3. Règles d’affaires      1. Obtenir un objet représentant l’objectif (créé précédemment).      2. Lors du chargement d’un niveau, ajouter la classe représentant l’objectif.      3. Démarrer l’horloge interne du jeu.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. Vérifier que l’objectif est bien affiché.   6. Post-conditions  1. Le portal de fin de niveau apparaît à l’écran. |
| Tests d’acceptation | Chargé un niveau de test et vérifier que le portal de fin de niveau apparaît. |
| Complexité | 3 |
| Effort | 2 |
| Commentaires |  |
| **Scénario #28** | |
| Acteurs | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux pouvoir modifier la description et le titre dans l’éditeur et les voir dans l’écran de sélection de niveau. |
| Description | 1. Passer le paramètre description du niveau dans la fenêtre de choix.   * 1. Qui et temps      1. J.S.      2. 60 min.   2. Préconditions   3. Règles d’affaires      1. Avoir un éditeur de niveau fonctionnel      2. Avoir un paramètre description dans la classe niveau.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. Le message de description du niveau s’affiche quand on clique sur son numéro. Il vient de la classe niveau.   6. Post-conditions      1. On peut maintenant modifier les descriptions des niveaux depuis leurs fichiers textes sauvegardés. Chaque niveau customisé sera aussi capable d’avoir sa description.   2. Passer le paramètre titre du niveau dans la fenêtre de choix.   * 1. Qui et temps      1. J.S.      2. 60 min.   2. Préconditions   3. Règles d’affaires      1. Avoir un éditeur de niveau fonctionnel      2. Avoir un paramètre titre dans la classe niveau.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. Le titre du niveau s’affiche quand on clique sur son numéro. Il vient de la classe niveau.   6. Post-conditions      1. On peut maintenant modifier les titres des niveaux depuis leurs fichiers textes sauvegardés. Chaque niveau customisé sera aussi capable d’avoir son titre personnalisé. |
| Tests d’acceptation | Cliquer sur le bouton du niveau et regarder si son titre et sa description s’affiche. |
| Complexité | 2 |
| Effort | 2 |
| Commentaires |  |
| **Scénario #29** | |
| Acteurs | Utilisateur |
| Scénario | En tant qu’utilisateur, je veux qu’il y ait des niveaux déjà fait dans l’interface niveau. |
| Description | 1. Créer dix niveaux différents et les sauvegarder avec le projet dans un package.   * 1. Qui et temps      1. J.S.      2. Indéterminé, dépend de l’imagination et de la création des niveaux. Mais cela peut être long.   2. Préconditions      1. Avoir accès à un éditeur de niveau fonctionnel qui travaille avec les objectifs. Sans les objectifs, nous ne pouvons pas faire de niveaux fonctionnels.   3. Règles d’affaires      1. Créer environ 10 niveaux différents avec différents degrés de difficulté.      2. Être capable de jouer sur les niveaux créés.   4. Règles d’affaires alternatives   5. Tests d’acceptation      1. Le jeu enchaine les différents niveaux et progresse.   6. Post-conditions  1. Nous avons maintenant un jeu jouable. |
| Tests d’acceptation | Le jeu enchaine les niveaux un après l’autre. Quand on finit le 1er niveau, on tombe au niveau 2, etc. |
| Complexité | 2 |
| Effort | 5 |
| Commentaires | Ceci peut être très long, car il faut s’assurer que chaque niveau créé soit assez difficile. Plusieurs paramètres sont à réviser. |