PREUVES

DOCUMENTATION :

* Je sais décrire le contexte de mon application, pour que n'importe qui soit capable de comprendre à quoi elle sert. [sur 2 points]

Voir dans le dossier.

* Je sais faire un diagramme de cas d'utilisation pour mettre en avant les différentes fonctionnalités de mon application. [sur 5 points]

Voir dans le dossier.

* Je sais concevoir un diagramme UML de qualité représentant mon application. [sur 7 points]

Voir dans le dossier.

*Je sais décrire un diagramme UML en mettant en valeur et en justifiant les éléments essentiels. [sur 6 points]

Voir dans le dossier

Programmation :

* Je sais utiliser les Intent pour faire communiquer deux activités. [sur 1 point]

Preuve :

- Modele/Activity/MainActivity -> ligne 79

* Je sais développer en utilisant le SDK le plus bas possible. [sur 1 point]

Preuve :

Nous développons notre application avec le SDK 15 qui supporte 100% des téléphones

* Je sais distinguer mes ressources en utilisant les qualifier. [sur 1 point]

Preuve :

Toutes nos strings sont déclarés dans le string.xml (res/values/strings.xml).

Nos drawables sont placés dans le répertoire drawable-xhdpi : res/drawable (optimisé pour les nexus)

* Je sais modifier le manifeste de l'application en fonction de mes besoins. [sur 1 point]

Preuve :

- manifests/AndroidManifest.xml : Toutes les activités sont déclarées (avec leurs activités parentes) dans le manifeste et on demande les permissions Bluetooth.

* Je sais faire des vues xml en utilisant layouts et composants adéquats. [sur 1 point]

Preuve:

- layout/bt_select_layout.xml : Utilisation d'un linear layout en vertical afin d'afficher tous nos éléments verticalement, les uns en dessous des autres. La ListView affiche les appareils auquels le joueur peut se connecter.
- layout/game_over.xml : Utilisation d'un linear layout en vertical afin d'afficher tous nos éléments verticalement, les uns en dessous des autres. Un TextView et un ImageButton sont placé à l'intérieur pour afficher que la partie est terminée et retourner au menu principal.

* Je sais coder proprement mes activités, en m'assurant qu'elles ne font que relayer les évènements. [sur 1 point]

Preuve :

- Activity/MainActivity. Cette activité est celle qui se lance lorsque nous démarrons l'application. Elle ne fait que relayer les événements de clic sur les boutons pour lancer le jeu par exemple.
- Activity/GameActivity. Cette activité est celle qui se lance lorsque le joueur clique sur « jouer ». Lorsqu'elle reçoit l'événement « un joueur est mort », elle lance l'activité « GameOver ».
- Activity/GameOverActivity. Cette activité se lance lorqu'un joueur est mort. Lorsque le joueur clique sur le bouton « Home », elle lance l'activité « MainActivity » afin de retourner au menu principal du jeu.

* Je sais coder une application en ayant un véritable métier. [sur 2 points]

Preuve :

- Package Object qui définit tous les éléments graphiques du jeu ainsi que le joueur qui est lui aussi un élément graphique. Voir diagramme de classe dans l'autre dossier.

* Je sais parfaitement séparer vue et modèle. [sur 1 point]

Preuve :

La GameView représente la vue de notre jeu. Elle possède une Grid qui va nous permettre de positionner chaque objet. La classe GameView va alors relayer les événements liés à l'interaction avec l'utilisateur (comme les clics par exemple) au GameEngine qui lui gère la mise à jour des élément graphiques en fonction du modèle.

Voir diagramme de classe.

* Je maîtrise le cycle de vie de mon application. [sur 1 point]

Preuve :

- Modele/saveGameState : Quand notre vue est détruite elle se sauvegarde avec notre singleton SaveGameState puis se restaure dans le onCreate. Nous avons utilisé la persistance légère pour gérer le cycle de vie.
- Utils/BluetoothManager/BluetoothActivity : Notre activité bluetooth utilise également une persistance légère pour se sauvegarder.

* Je sais utiliser le findViewById à bon escient. [sur 1 point]

Preuve :

- Utils/BluetoothManager/BluetoothActivity : Lignes 65, 66, 73 : Nos findViewById sont tout le temps sauvegardé dans une variable afin d'éviter de rapeller la methode à chaque fois

* Je sais gérer les permissions dynamiques de mon application. [sur 1 point]

Preuve :

Pour demander par exemple l'accès à la localisation bluetooth pour la découverte de téléphone.

- Utils/BluetoothManager/BluetoothActivity : Ligne 166 à 169 : Pour demander par exemple l'accès à la localisation bluetooth pour la découverte de téléphone.

* Je sais gérer la persistance légère de mon application. [sur 1 point]

Preuve :

- Modele/SaveGameState : On a utilisé un singleton qui sauvegarde et restaure l'état de jeu en fonction du cycle de vie de l'activité

* Je sais gérer la persistance profonde de mon application. [sur 1 point]

Nous n'avons pas eu le temps de traiter cette partie. Nous aurions pu faire un classement de joueur (avec enregistrement avec persistance profonde).

* Je sais afficher une collection de données. [sur 1 point]

Preuve :

- Res/layout/bt_select_layout et Utils/BluetoothActivity : Lorsqu'on cherche des appareils dans le mode 2 joueurs, les appareils à proximité s'affiche sous forme de collection de données dans une listView :

* Je sais coder mon propre adaptateur. [sur 2 points]

Voir la classe Utils.MonAdapter.java qui est un adapteur personnel qui affiche les données dans un TextView avec un LayoutInflater.

Nous n'avons pas trouver l'utilité de faire un fragment pour notre jeu.

* Je maîtrise l'utilisation de Git. [sur 1 point]

Preuve :

- Voir branche, commit, merge

Application :

* Mon application présente un intérêt à être publié sur le store. [sur 5 points]

Preuve :

- Notre application est un jeu pour se divertir de type Bomberman. Il peut être joué à la fois tout seul mais aussi en multijoueur avec des amis. Idéale pour passer une bonne soirée entre amis !

* Mon application fonctionne de manière à être utilisée par le public. [sur 5 points]

Preuve :

- Les différentes commandes sont affichées lorsqu'on clique sur le bouton Instructions sur l'activité principale. L'application est le plus ergonomique possible et simple d'utilisation pour que n'importe quel utilisateur puisse jouer. Nous avons voulu faire une application le plus user-friendly possible.

* Mon application utilise des contraintes spécifiées lors du choix du projet. [sur 5 points]

Preuve :

Notre application utilise la technologie Bluetooth.

* Mon application utilise les contraintes à bon escient. [sur 5 points]

Preuve :

- Grâce au bluetooth, 2 joueurs peuvent jouer l'un contre l'autre en se déplaçant et en posant des bombes tout en essayant de pulvériser son adversaire. Toutes les données de la partie transitent via le serveur bluetooth entre les 2 appareils. Un des téléphones est le serveur l'autre le client.