Notre jeu inclut tous les aspects imposés par le cahier des charges, dont :

La Balle :

Nous avons mis en place différents types de balles :

- ClassicBall : une balle ronde qui n'a rien de spécial.
- MagnetBall : une balle qui peut être chargée positivement ou négativement.
- GravityBall : une balle qui a une force qui l'attire vers le bas.
- HyperBall : une balle qui gagne en vitesse tout au long de la partie (non utilisée en jeu en raison de sa non-conformité).

Toutes les classes des balles étendent la classe Ball (dans le module physics), qui elle-même étend la classe Entity.
Effets physiques sur la balle dans le jeu :

- Les rotations
- Collision entre 2 balles
- Collision de la balles avec une figure (toute les formes possible de la raquette)
- Gravité
- Magnet

La Raquette :

De la même manière que la balle, il existe plusieurs types de raquettes :

- ClassicRacket : raquette rectangulaire avec la capacité de sauter.
- MagnetRacket : raquette qui peut être chargée positivement ou négativement pour interagir avec les balles à champ électromagnétique.
- DegradeRacket : raquette en forme de triangle pour des collisions différentes de la ClassicRacket.

Raquettes codées mais non utilisées en jeu :

- DiamondRacket : raquette en forme de losange.
- CircleRacket : raquette en forme d'ellipse.
- YNotFixeRacket : raquette rectangulaire qui peut aussi monter et descendre.

Toutes les classes des raquettes étendent la classe Racket (dans le module physics), qui elle-même étend la classe Entity.

Les Briques :

Un seul type de briques existe. Plusieurs modes de jeu sont créés autour d'autres fonctionnalités des briques (cf Règles de jeu).

Moteur Physique :

La partie la plus importante de notre projet réside dans la mise en place d'un moteur physique (dont la plupart des propriétés des balles et des raquettes ont été intégrées dans le jeu).

Notre moteur physique implémente de la manière réaliste des phénomènes comme le vent ou bien la gravité, tout en ayant une liberté dans la manipulation de la simulation (modifier les données des paramètres, bouger les entités /les changer)
En dehors du jeu, une interface a été mise en place pour tester différents phénomènes :

- La collision entre deux balles.
- La collision entre une balle et une figure (ensemble de segments, dans notre jeu : rectangle, triangle ou losange).
- Des tests pré faits sont présents pour montrer des événements rares ou compliqués à reproduire.

Difficultés rencontrées lors de l'implémentation du moteur :

- Collision entre une balle et une autre entité (balle ou figure) sous différents aspects :
    - Insertion d'une entité dans une autre.
    - Gestion et renvoi réalistes de l'entité avec des formules mathématiques et physiques (ex. Pythagore, effet Magnus).
    - Sortie des entités du cadre de la simulation (surtout à cause de vitesses excessives).
    - Variances de mise à jour des positions des entités rendant les collisions difficilement gérables.

Ajouts en dehors des contraintes du cahier des charges:

La Console :

Permet au joueur d'interagir avec les données du jeu et de les modifier selon son souhait.
La ConsoleView envoie la saisie de l'utilisateur à Console pour traitement :

- Si c'est un message ordinaire (ne commence pas par un /), il l'envoie directement dans la file d'attente pour l'affichage (affiché ensuite par ConsoleView).
- Si c'est une commande (commence par un /), elle la traite.

Traitement : Le traitement se déroule en plusieurs phases :

- Découpage : la saisie est découpée en tableau de String, les espaces sont les délimiteurs, avec tolérance et gestion des erreurs de frappe (espaces en trop).
- Analyse : traite et envoie à d'autres analyseurs si besoin, similaire à un traitement Token dans un compilateur de code.
- Résultat : affiche la commande retenue et agit en conséquence.

Avec la console sont fournis plusieurs commandes qui servent chacun un but différents tel que : une commande pour sortir du jeu et faire une sauvegarde automatique, **/get …** qui permet d'afficher différentes informations tel que ; l'inventaire, le thème choisi, le pseudo ….ect , **/set …** qui permet de changer ces mêmes informations, ou meme d'ouvrir directement la demo ou le moteur physique

Différentes règles de jeu :

Dans la classe GameRules qui comporte l'ensemble des règles additionnelles au jeu classique, appliquées au Game.
Règles :

- **Temps limité (limitedTime) :** Si le joueur ne casse pas toutes les briques avant le temps imparti, il perd la partie.
- **Rebonds limités (limitedBounces) :** Si le joueur atteint la limite des rebonds sans casser toutes les briques, il perd la partie.
- **Briques avec switch de position aléatoire (randomSwitchBricks) :** La position de chaque brique est téléportée aléatoirement à chaque rebond sur la raquette.
- **Couleur (colorRestricted) :** Les briques et la balle ont une couleur. La balle peut casser la brique si leur couleur est identique. La couleur de la balle change en fonction de la brique touchée.
- **Transparence/Ghost (transparent) :** Les briques peuvent devenir transparentes, laissant la balle passer à travers. La transparence dure 1 tour.
- **Incassable (unbreakable) :** Les briques deviennent incassables pour 1 tour.
- **Infini (infinite) :** Les briques apparaissent de haut en bas. Si la brique touche la raquette, le joueur a perdu. Le but est de tenir le plus longtemps possible.

Système de sauvegarde :

Permet de sauvegarder la monnaie, le thème choisi, les niveaux débloqués, les skins/textures, les touches de contrôle de la raquette, les niveaux sonores, et l'inventaire. Avec ceci est fourni un logiciel Repair Software qui sert à : Réparer automatiquement les sauvegardes, Enlever la sauvegarde par défaut du jeu, Sauvegarder les données dans un fichier, Supprimer les sauvegardes corrompues, Vérifier la validité d'une sauvegarde…
Avec ça est créé une classe PlayerData qui encapsule les données brutes de l'utilisateur (c'est aussi là que sont recopiées les données de la Sauvegarde), les données sont initialisées avec des valeurs par défaut si aucune Sauvegarde n'est établie. Les données sont : le pseudo, le niveau d'expérience, la monnaie, son rôle (admin ou joueur), sa progression (StageProgress: la classe englobant la progression du joueur, ses données sur chaque niveau (StageLevel)) ainsi que son inventaire (Inventaire), et qui sert à récupérer les données sauvegardé dans le jeu.

Options :

Rajoute de la personnalisation :

- Afficher les FPS

- Avoir une traînée de particules
- Modifier le choix des touches de contrôle de la raquette
- Régler les niveaux sonores
- Choisir le thème (des thèmes adapté à différents types de daltonisme tel que : Achromatopsie , Deutéranopie, Protanopie, Tritanopie.)

Système de monnaie :

Gain de monnaie:
(Base : Score de la partie)
Gain = (Base + Bonus) * (Multiplicateurs)
Calcul en fonction de :
Bonus de gain de monnaie : (+)

- Certaines règles (ex. LimitedTime prend en compte le temps restant)
- Vies restantes (+15/vie)
- Difficulté (niveau, chapitre)

Multiplicateur de gain de monnaie : (x)

- Premier clear d'un niveau (x2)
- Finir un niveau déjà clear (x1)
- Certaines règles (entre x1.05 et x1.5 /règle)

La monnaie gagnée peut être dépensée dans la Boutique du jeu.

Tutoriel :

Afin de mieux comprendre le jeu dans le menu du jeu il y a disposition un bouton “tutoriel” qui va ouvrir un google docs qui permet d’expliquer le jeux mais pas que:

- F5: permet d’actualiser la page Google docs ou d'y retourner
- F11: mettre le jeu en plein écran
- ESCAPE: pour revenir au menu
- F4: ouvrir le rapport
- F3: ouvrir le dépôt git du projet
- F2: ouvrir une page Google

Mode infini :

Un mode de jeu où les briques apparaissent continuellement du haut de l'écran et descendent progressivement vers la raquette. Le but est de survivre aussi longtemps que possible en évitant que les briques ne dépassent la raquette.

Personnalisation de niveau :

Permet de créer son propre niveau en ajustant la taille de la balle , sa vitesse, le nombre de colonnes et de lignes des briques, le nombre de vies, et d’activer ou de désactiver toutes les différentes règles citées précédemment.

Chapitre de niveaux :

Le jeu possède 4 chapitres de 9 niveaux chacun. Chacun des niveaux est différent.
Les niveaux varient avec différentes contraintes (cf. les règles de jeu).
Les chapitre varient avec une configuration de balle et de raquette différent:

- chapitre 1: ClassicRacket et ClassicBall
- chapitre 2: MagnetRacketet et MagnetBall
- chapitre 3: DegradeRacket et ClassicBall
- chapitre 4: ClassicRacket et GravityBall

Diagrammes de classes :

Dans ce diagrammes on a pris la décision de mettre uniquement les classes les plus importantes (rien de tout ce qui est interface graphiques pour éviter de le surcharge)

Pour la partie graphique : on a utilisé le modèle MVC, le modèle étant les classes représenté dans ce diagrammes :