

Student Invaders

ILLUSTRATION !!!!!
Echéance 1

Table des matières

1	Analyse préliminaire	3
1.1	Introduction	3
1.2	Objectifs.....	3
1.3	Planification initiale	5
2	Analyse / Conception.....	5
2.1	Concept	5
2.2	Stratégie de test.....	7
2.3	Risques techniques	7
2.4	Planification	7
2.5	Dossier de conception	8
3	Réalisation.....	17
3.1	Dossier de réalisation	19
3.2	Description des tests effectués	20
3.3	Erreurs restantes	20
3.4	Liste des documents fournis	21
4	Conclusions	21
5	Annexes.....	22
5.1	Résumé du rapport du TPI / version succincte de la documentation	22
5.2	Sources – Bibliographie.....	22
5.3	Journal de travail	22
5.4	Manuel d'Installation	22
5.5	Manuel d'Utilisation.....	22
5.6	Archives du projet.....	22

NOTE L'INTENTION DES UTILISATEURS DE CE CANEVAS:

Toutes les parties en italiques sont là pour aider à comprendre ce qu'il faut mettre dans cette partie du document. Elles n'ont donc aucune raison d'être dans le document final.

De plus, en fonction du type de projet, il est tout à fait possible que certains chapitres ou paragraphes n'aient aucun sens. Dans ce cas il est recommandé de les retirer du document pour éviter de l'alourdir inutilement.

1 Analyse préliminaire

1.1 Introduction

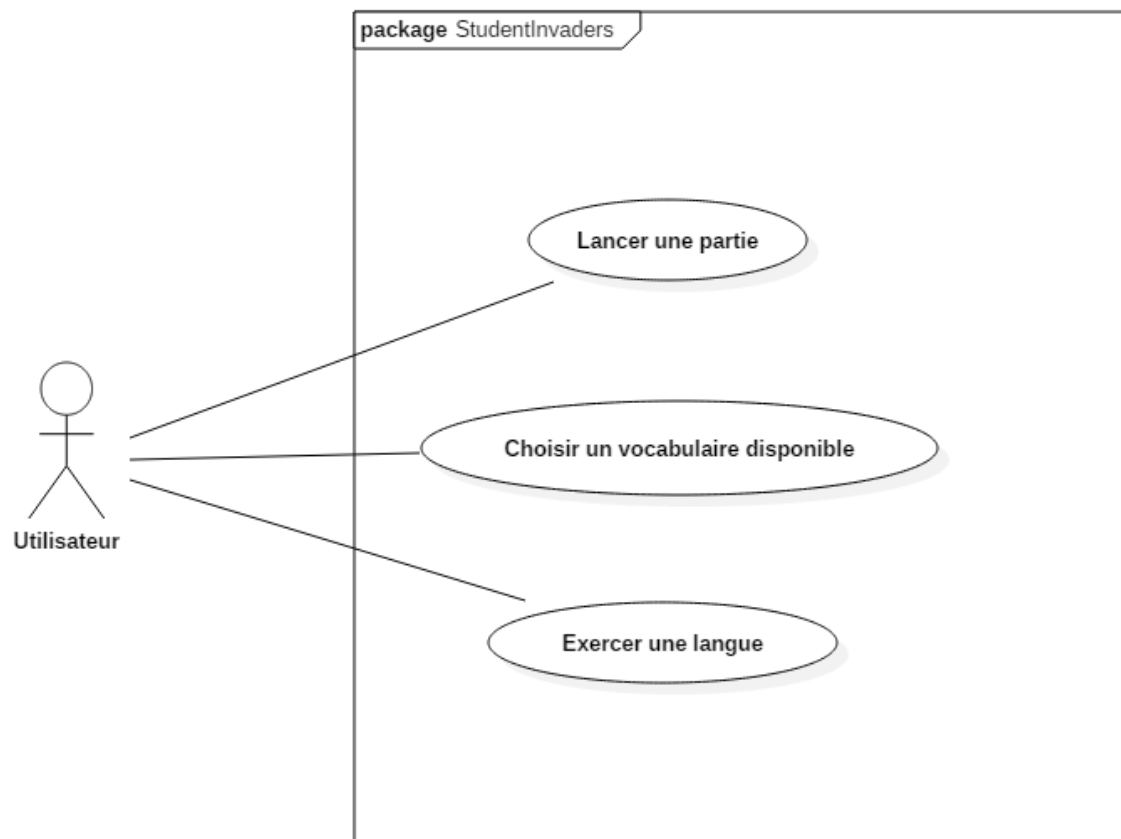
Student Invaders est un jeu didactique d'apprentissage de vocabulaire sur la plateforme Android d'une manière d'un Space Invaders. Dans cette application, le joueur contrôle le professeur. Il a pour but d'envoyer les mots traduits aux élèves avec un avion en papier. Ce projet est une manière assez intuitive d'apprendre et d'exercer une langue. Personnellement, j'apprendrai plus du côté de la programmation mais le jeu peut apporter un côté interactif aux cours. Ce projet reprend le même langage de programmation que mon projet de pré-TPI, AndroidSnake. J'ai donc les bases pour commencer ce projet.

1.2 Objectifs

Les objectifs définis pour ce projet sont les suivants :

- Le joueur peut sélectionner la langue du côté du prof et de l'élève qu'il souhaite exercer ainsi que le vocabulaire.
- Le professeur peut se déplacer latéralement.
- Les élèves avancent aussi latéralement. Cependant, ils avancent en avant quand un élève qui est à l'extrême gauche ou l'extrême droite atteint un bord.
- Un élève part en pause dès qu'il a eu 3 mots traduits correctement.
- Si un élève reçoit une fausse traduction, il avance en avant (uniquement l'élève ayant reçu l'avion en papier avance et non la ligne).
- Si le professeur a envoyé tous les élèves en pause, un écran « Fin de partie » avec le score.
- Si l'élève atteint le prof, un écran « Game Over » s'affiche.

Les use-cases que j'ai défini sont :



1.3 Planification initiale

Durant ce TPI, j'évoluerai avec la méthode Agile. Elle a été imposée par le chef de projet. La méthode Agile consiste à mettre en place différents sprint durant lesquels certaines tâches seront effectuées. À chaque fin de sprint, le mandataire doit fournir une démo au chef de projet pour ainsi reporter différents problèmes dans le sprint suivant. La validation est faite par le chef de projet ainsi que les deux experts. L'outil utilisé pour réaliser la planification est Trello. Les détails de chaque sprint seront dessus. Il est accessible via le lien suivant : <https://trello.com/b/KeV2VVap/student-invaders>

Voici un tableau de la planification initiale :

Sprint 1	Mise en place de l'interface	Au 17 Mai
Sprint 2	Mise en place du Prof	Au 24 Mai
Sprint 3	Mise en place des étudiants	Au 29 Mai
Sprint 4	Menu, Scoring, Début, Fin et Pause	Au 1 ^{er} Juin
Sprint 5	Tests et correction des bugs	Au 5 Juin
Rendu	Rendu final	Au 7 Juin

2 Analyse / Conception

2.1 Concept

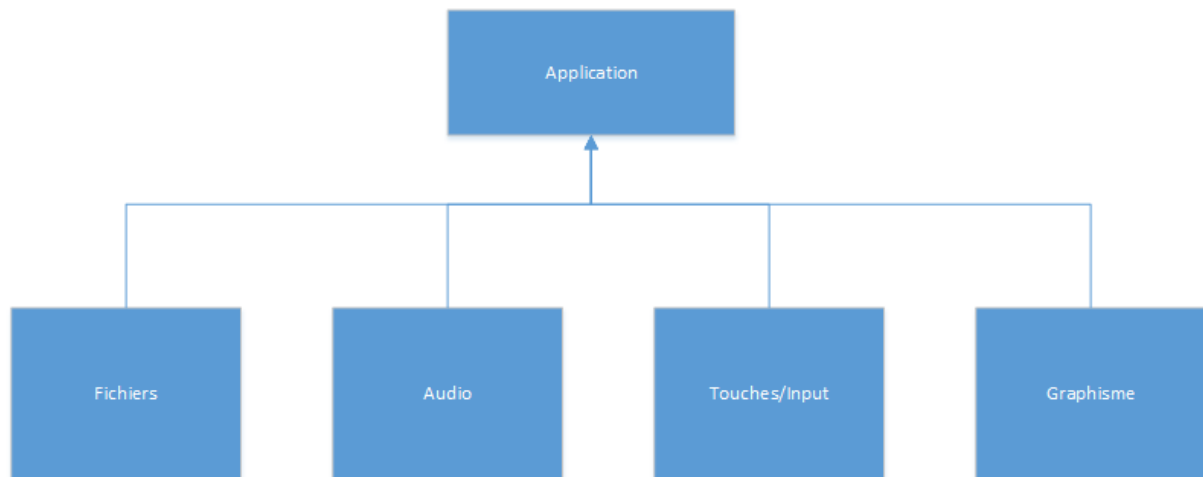
2.1.1 Vue d'ensemble

L'architecture Android se compose ainsi :



Sur ce graphique, notre application sera dans la partie « Applications » et le framework LibGDX dans « Application Framework ».

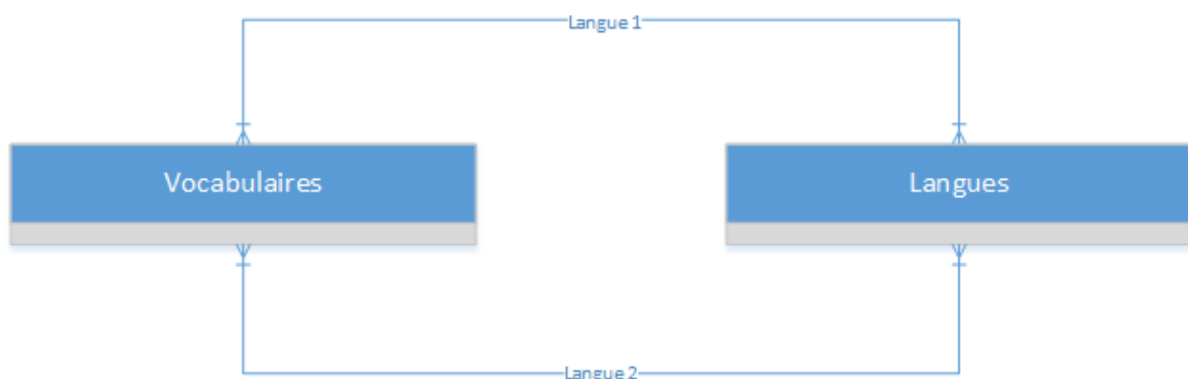
Concernant l'architecture du framework LibGDX :



L'application est le framework. Elle s'occupe de gérer les cycles du jeu ainsi que les boucles. Dans ce bloc contient différents modules utilisables :

- Le module des fichiers : Lecture/Ecriture des ressources ainsi que des assets.
- Le module de l'audio : Jouer de la musique, les effets sonores.
- Le module des touches ou Input/Output : Tous les événements concernant les clics de souris, l'appui sur une touche du clavier ou le toucher sur un écran mobile.
- Le module Graphisme : Dessiner les sprites/images ou ~~les~~ tout autre graphique sur l'écran.

2.1.2 MCD



Le MCD ne concerne que les vocabulaires et les langues. Chaque vocabulaire à deux langues et chaque langue a un ou plusieurs vocabulaires. Deux traits distincts ont été dessinés pour montrer qu'il y a une langue 1 qui concerne le prof et une langue 2 qui concerne celui de l'élève.

2.2 Stratégie de test

Il y aura 3 types de tests pour ce projet. Le test de robustesse, le test de performance et le test fonctionnel. L'ordre dans lequel ils seront effectués sera :

1. Test fonctionnel.
 - a. Je commence par ce test pour m'assurer que tout fonctionne correctement et que mon application ne plante pas durant les essais.
2. Test de performance.
 - a. Il s'agit de vérifier si mon application a des ralentissements quand il traite les données. Cela peut avoir un impact sur l'expérience de jeu.
3. Test de robustesse.
 - a. Au final, nous allons tester s'il y a une gestion des erreurs suivant des cas anormaux.

Dans les tests fonctionnels, ~~Il y aura~~ des tests unitaires qui seront effectués tout au long du projet. Les tests systèmes auront aussi lieu pour tester la connexion entre le webservice et le jeu. Ces essais se dérouleront à chaque fin de sprint.

Pour réaliser ces tests, il faut se munir d'une tablette Android ou d'un émulateur Android sur ordinateur. Un téléphone est aussi possible pour l'utilisation mais l'application n'est pas optimisée pour ce dernier.

Comme données de test, il faudra utiliser des données réelles pour les langues disponibles ainsi que les mots traduits. Ces données sont fournies par Mr. Carrel dans une base de donnée mis à disposition. Même si je peux effectuer les tests moi-même, je vais demander à mes camarades qui ont du temps pour essayer l'application et découvrir les différents problèmes/bugs.

2.3 Risques techniques

~~Quelques~~ Un des risques techniques possible est l'implémentation du web service. En effet, je n'ai jamais appris à implémenter un web service sous le langage Java. Cependant, différents professeurs peuvent me venir en aide si nécessaire. Différents tutoriels sont aussi disponibles pour l'implémentation.

- *risques techniques (complexité, manque de compétences, ...).*

Décrire aussi quelles solutions ont été appliquées pour réduire les risques (priorités, formation, actions, ...).

Echéance 3

2.4 Planification

Révision de la planification initiale du projet :

- *planning indiquant les dates de début et de fin du projet ainsi que le découpage connu des diverses phases.*
- *partage des tâches en cas de travail à plusieurs.*

*Il s'agit en principe de la planification **définitive du projet**. Elle peut être ensuite affinée (découpage des tâches). Si les délais doivent être ensuite modifiés, le responsable de projet doit être avisé, et les raisons doivent être expliquées dans l'historique.*

=résumé du Trello

<i>Sprint</i>	<i>Démo</i>	<i>Résultat</i>
<i>No 1</i>	<i>17 Mai</i>	<i>Manque l'affichage des vocabulaires. Nouvelle classe à créer pour utiliser les données. Deux tâches sont reportées au sprint suivant.</i>
<i>No 2</i>	<i>24 Mai</i>	<i>L'envoi de l'avion et la prise des mots ne sont pas encore en place. Une démo est à faire pour ces deux objectifs le 29 Mai.</i>
<i><u>No 2</u></i>	<i><u>29 Mai</u></i>	

Echéance 2
Echéance 3
Echéance 4
Echéance 5

2.5 Dossier de conception

Fournir tous les document de conception:

- ~~le choix du matériel HW~~
- ~~le choix des systèmes d'exploitation pour la réalisation et l'utilisation~~
- ~~le choix des outils logiciels pour la réalisation et l'utilisation~~
- ~~site web: réaliser les maquettes avec un logiciel, décrire toutes les animations sur papier, définir les mots-clés, choisir une formule d'hébergement, définir la méthode de mise à jour,...~~
- ~~bases de données: décrire le modèle relationnel, le contenu détaillé des tables (caractéristiques de chaque champs) et les requêtes.~~
- ~~programmation et scripts: organigramme, architecture du programme, découpage modulaire, entrées-sorties des modules, pseudo-code / structogramme...~~

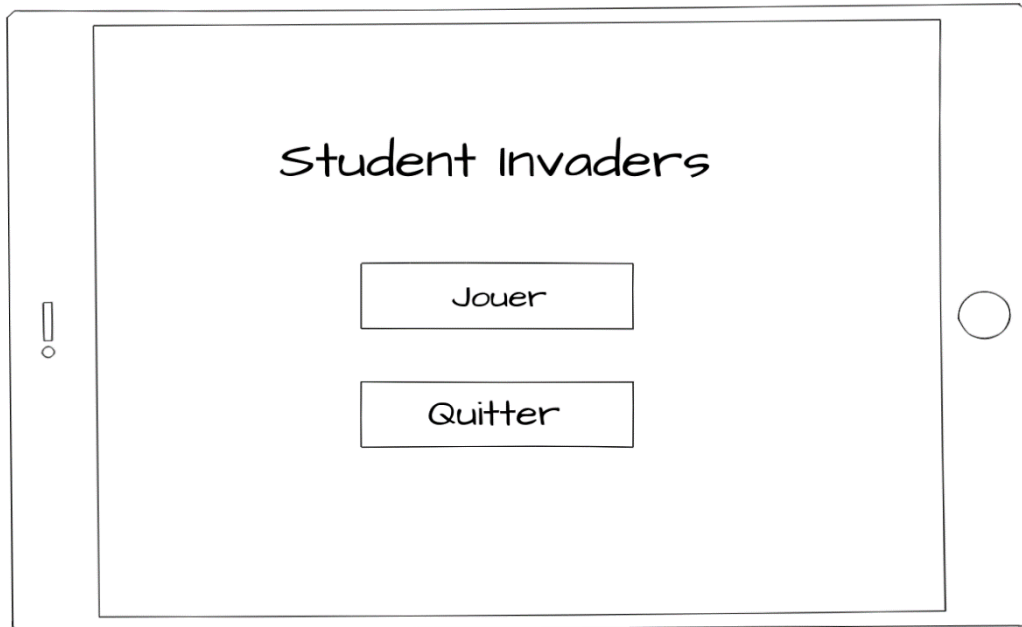
~~Le dossier de conception devrait permettre de sous-traiter la réalisation du projet !~~

Pour ce projet, J'ai utilisé comme système d'exploitation Windows 7. J'ai ensuite choisis le logiciel Android Studio. Il a été choisi car il est simple d'utilisation avec les raccourcis et que c'est un classique. Le langage de programmation utilisé pour le jeu est Java. J'ai aussi utilisé le framework LibGDX.

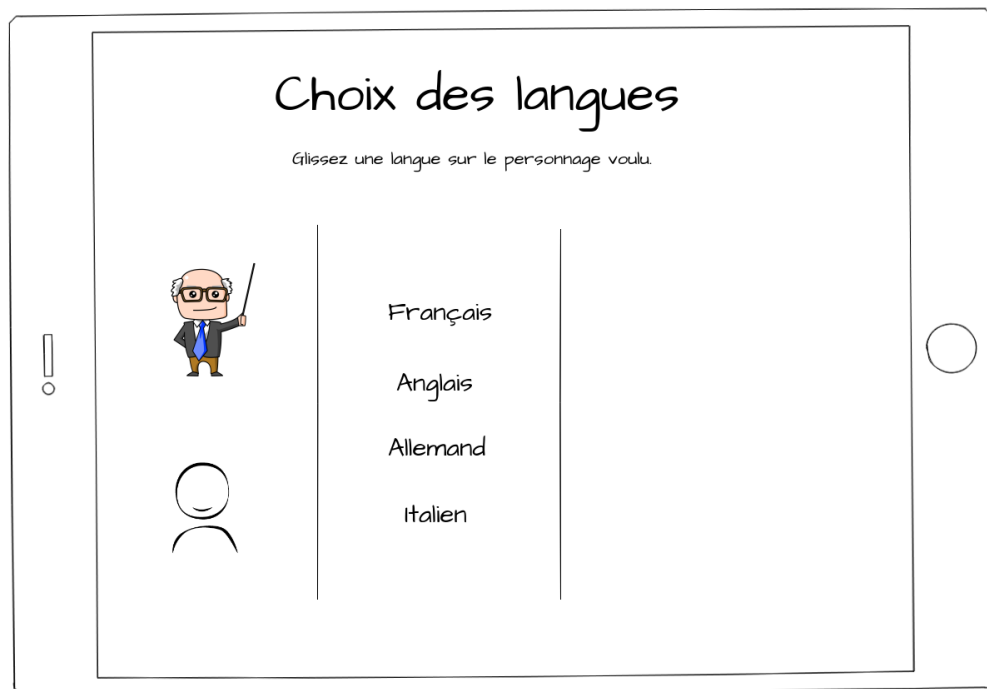
J'ai emprunté une tablette Samsung Galaxy Tab 2 sous Android 4.2.2 pour effectuer mes tests directement sur un appareil mobile.

2.5.1 Maquettes / Use cases / Scénarios

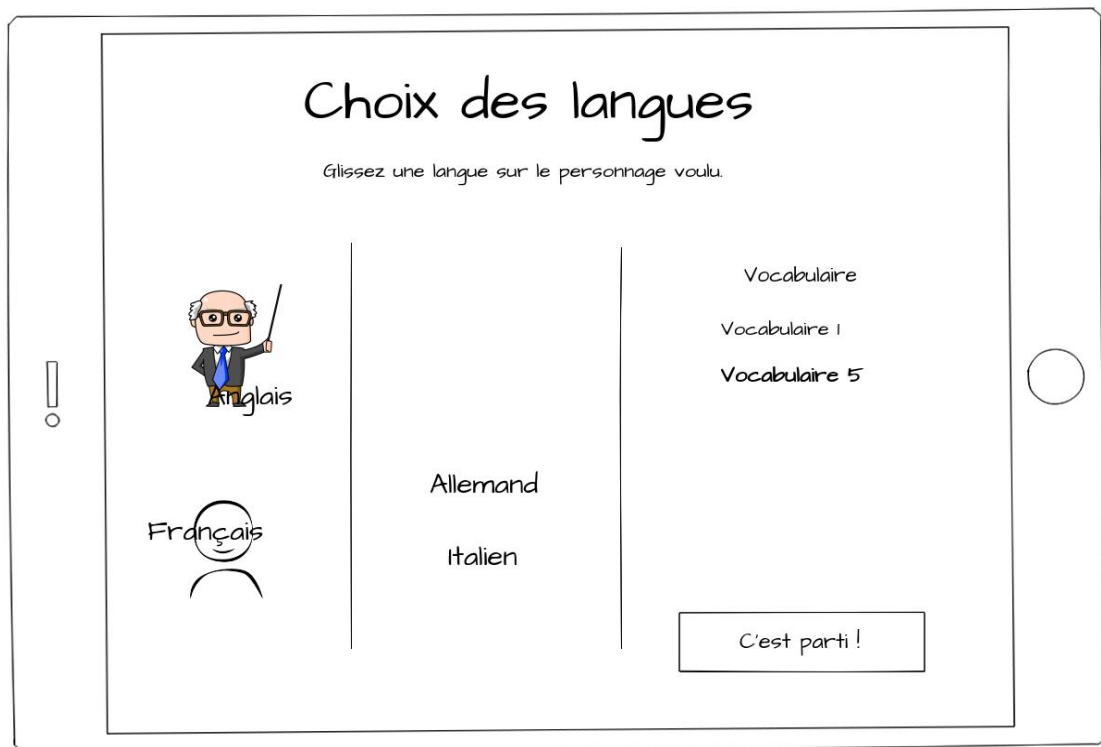
1. Le menu :



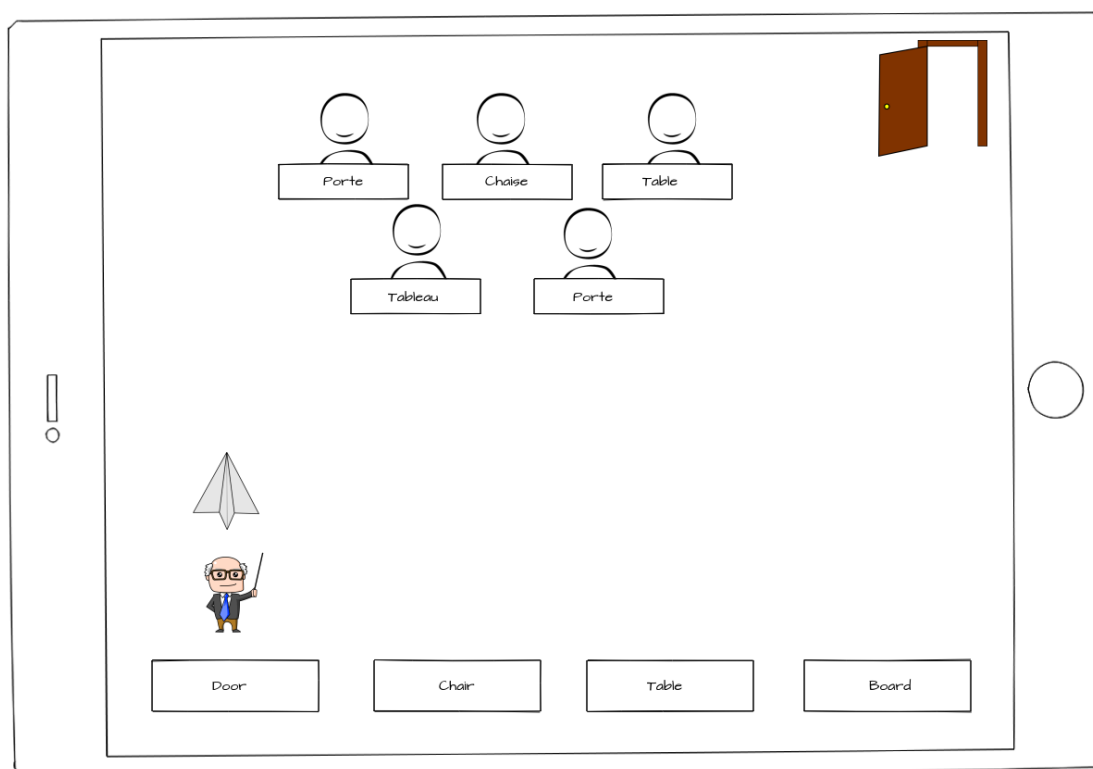
2. La sélection des langues :



Langues choisies ainsi que le vocabulaire :



3. La partie jeu :



Scénarios :

1. Menu

1.1. Lancement d'une partie

Action	Condition particulière	Réponse
Le joueur appuie sur le bouton « Jouer »		Le jeu passe sur l'écran « Choix des langues ».

1.2. Quitter le jeu

Action	Condition particulière	Réponse
Le joueur appuie sur le bouton « Quitter »		Retour à l'écran d'accueil de l'appareil.

2. Sélection des langues

2.1. Le joueur choisit le français pour le professeur et l'anglais pour l'élève

Action	Condition particulière	Réponse
Le joueur appuie sur le bouton « Jouer »		Le jeu passe sur l'écran « Choix des langues ».

Le joueur glisse avec son doigt le texte « Français » sur le professeur		Un message s'affiche en indiquant que le français a été sélectionné pour le professeur.
Le joueur glisse avec son doigt le texte « Anglais » sur l'élève		Un message s'affiche en indiquant que l'anglais a été sélectionné pour l'élève.
Le joueur clique sur le vocabulaire souhaité		Un bouton « C'est parti » s'affiche.

2.2. Le joueur souhaite changer de langue pour le professeur

Action	Condition particulière	Réponse
Le joueur appuie sur le bouton « Jouer »		Le jeu passe sur l'écran « Choix des langues ».
Le joueur glisse avec son doigt le texte « Français » sur le professeur		Un message s'affiche en indiquant que le français a été sélectionné pour le professeur.
Le joueur glisse avec son doigt le texte « Anglais » sur l'élève		Un message s'affiche en indiquant que l'anglais a été sélectionné pour l'élève.
Le joueur glisse le texte « Français » à côté et met le texte « Allemand » sur le professeur.		Un message s'affiche en indiquant que le l'allemand a été sélectionné pour le professeur.
Le joueur clique sur le vocabulaire souhaité		Un bouton « C'est parti » s'affiche.

2.3. Le joueur souhaite changer de langue pour l'élève

Action	Condition particulière	Réponse
Le joueur appuie sur le bouton « Jouer »		Le jeu passe sur l'écran « Choix des langues ».
Le joueur glisse avec son doigt le texte « Français » sur le professeur		Un message s'affiche en indiquant que le français a été sélectionné pour le professeur.
Le joueur glisse avec son doigt le texte « Anglais » sur l'élève		Un message s'affiche en indiquant que l'anglais a été sélectionné pour l'élève.
Le joueur glisse le texte « Anglais » à côté et met le texte « Allemand » sur le professeur.		Un message s'affiche en indiquant que le l'allemand a été sélectionné pour le l'élève.
Le joueur clique sur le vocabulaire souhaité		Un bouton « C'est parti » s'affiche.

2.4. Pas de vocabulaire pour la langue sélectionnée

Action	Condition particulière	Réponse
Le joueur appuie sur le bouton « Jouer »		Le jeu passe sur l'écran « Choix des langues ».
Le joueur glisse avec son doigt le texte « Français » sur le professeur		Un message s'affiche en indiquant que le français a été sélectionné pour le professeur.
Le joueur glisse avec son doigt le texte « Anglais » sur l'élève		Un message s'affiche en indiquant que l'anglais a été sélectionné pour l'élève.
-	Pas de vocabulaire retourné par le webservice.	Un message s'affiche indiquant qu'aucun vocabulaire n'est disponible pour ces deux langues sélectionnées.

2.5. Le joueur change de vocabulaire

Action	Condition particulière	Réponse
Le joueur appuie sur le bouton « Jouer »		Le jeu passe sur l'écran « Choix des langues ».
Le joueur glisse avec son doigt le texte « Français » sur le professeur		Un message s'affiche en indiquant que le français a été sélectionné pour le professeur.
Le joueur glisse avec son doigt le texte « Anglais » sur l'élève		Un message s'affiche en indiquant que l'anglais a été sélectionné pour l'élève.
Le joueur appuie sur « Vocabulaire 1 »	Des vocabulaires sont disponibles pour les langues choisies.	Un bouton « C'est parti » s'affiche.
Le joueur appuie sur « Vocabulaire 5 »	Le vocabulaire 1 doit être sélectionné	Changement de vocabulaire. Le bouton « C'est parti » est toujours présent.

3. La partie jeu

3.1. Prof :

3.1.1. Le prof se déplace de gauche à droite

Action	Condition particulière	Réponse
Le joueur appuie sur une partie précise à gauche de l'écran.	L'endroit appuyé doit être au même niveau que le prof.	Le prof bouge petit à petit vers sa gauche.
Le joueur appuie sur une partie précise à droite de l'écran.	L'endroit appuyé doit être au même niveau que le prof.	Le prof bouge petit à petit vers sa droite.

3.1.2. Le prof choisit un mot

Action	Condition particulière	Réponse
Le joueur appuie sur une partie précise à gauche de l'écran.	L'endroit appuyé est au même niveau que le prof.	Le prof bouge petit à petit vers sa gauche.
Le joueur appuie sur le mot en dessous.	Le prof n'est pas au-dessus du rectangle.	Le prof continue son déplacement.
Le joueur appuie sur le mot en dessous.	Le prof est au-dessus du rectangle.	L'avion en papier se forme en dessus du professeur et le professeur se tourne.
Le joueur appuie sur une partie précise à droite de l'écran.		Le prof bouge petit à petit vers sa droite avec l'avion en dessus de lui.

3.1.3. Le prof lance un avion en papier

Ce scénario a lieu une fois que le prof a choisi un mot.

Action	Condition particulière	Réponse
Le joueur appuie sur une partie précise à gauche de l'écran.	L'endroit appuyé est au même niveau que le prof.	Le prof bouge petit à petit vers sa gauche.
Le joueur appuie sur une partie précise à droite de l'écran.		Le prof bouge petit à petit vers sa droite avec l'avion en dessus de lui.

Action	Condition particulière	Réponse
Le joueur appuie sur une partie de l'écran au-dessus du prof		L'avion part en ligne droite.
	L'avion atteint le bord de l'écran.	L'avion disparaît et le même mot réapparaît en bas de l'écran.

3.1.4. Le prof a envoyé tous les élèves en pause

Action	Condition particulière	Réponse
Le joueur joue sa partie.		Le prof lance des avions en papier, les élèves avancent.
-	Tous les élèves ont été envoyés en pause.	Un écran « Bien joué ! » s'affiche en indiquant le score du joueur ainsi que la possibilité de rejouer ou de changer la langue.

3.1.5. L'élève atteint le prof

Action	Condition particulière	Réponse
Le joueur joue sa partie.		Le prof lance des avions en papier, les élèves avancent.
-	L'élève atteint le prof.	Un écran Game Over s'affiche en proposant de rejouer ou de retourner au menu.

Echéance 2

Donner un identifiant à chaque maquette et chaque scénario. Ce n'est pas nécessairement un chiffre, mais ça reste court et unique.

Format de scénario exemple

Action	Condition particulière	Réponse
Va à l'URL www.wv.ch	Pas connecté	Page « Accueil anonyme »
Clic sur 'se connecter'		Page « Login »

2.5.2 MLD

mwb ou diagramme de classe

Echéance 3

2.5.3 Particularité 1 – Les blocs pour les mots du professeur

Arrivé à la moitié du projet, un problème s'imposait au moment d'afficher les mots pour le professeur. En effet, durant la programmation, je me suis posé la question suivante : « Comment faire pour afficher ces boîtes qui sont dans un tableau ? Dois-je afficher ces blocs en continu même s'ils seront cachés car ils dépassent l'écran ou dois-je en mettre 4 et dès qu'un mot a été traduit, on le supprime pour le remplacer par le suivant ? ». J'ai donc posé la question à mon chef de projet et il m'a conseillé de partir avec la première méthode car elle est plus simple. Pour cela, j'ai dû modifier le code en ajoutant une nouvelle classe qui est « TeacherWords ». Cette classe permet de connaître les mots que le prof possède ainsi que de savoir si un mot a été appris.

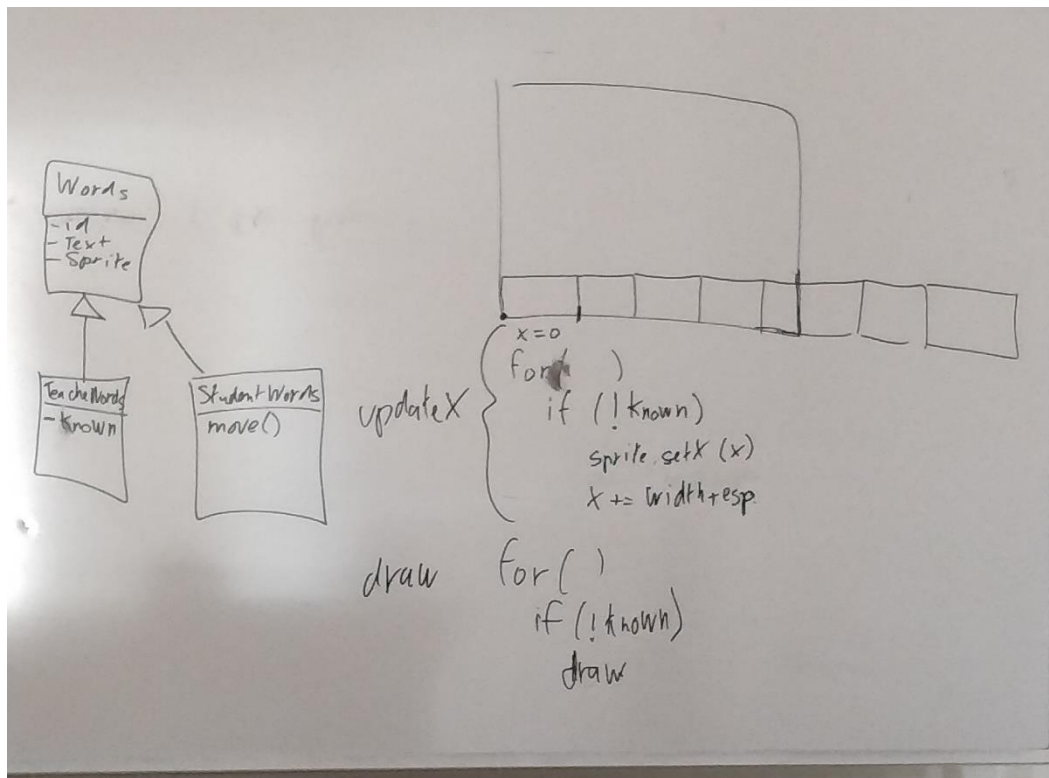


Figure 1. Photo de la discussion avec Mr. Carrel

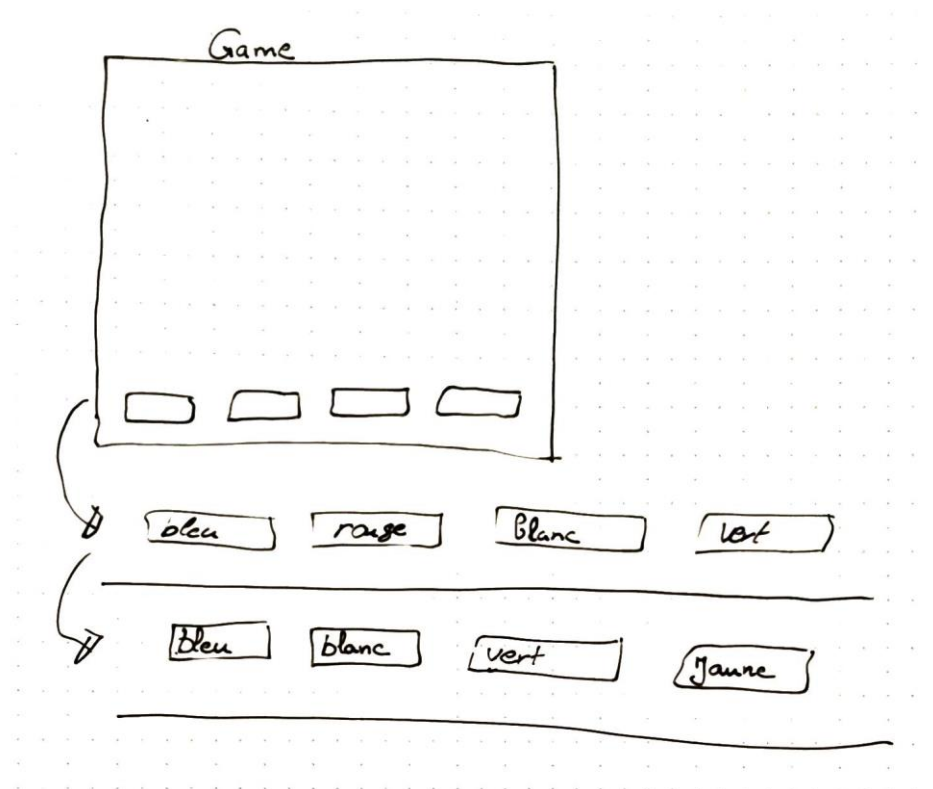


Figure 2. Dessin clair pour l'explication

Pour reprendre la figure 2, voici comment sera exécuté la méthode d'affichage des mots. Les premiers mots affichés, par exemple, seront « Bleu, Rouge, Blanc et Vert ». Si le joueur appuie sur « Rouge », le bloc disparaît à la fin de l'envoi de l'avion. Après cela, les blocs vont se décaler en reprenant l'ancienne position du bloc visible. Avec cette position, on additionne la largeur du bloc et un espace pour espacer les mots. De cette manière, nous évitons de modifier des valeurs dans un tableau et ainsi éviter des bugs qui pourraient intervenir (l'utilisation d'un index récupéré).

2.5.4 (Particularité 2)

Echéance 4







3 Réalisation

3.1 Dossier de réalisation

Pour ce projet, j'ai utilisé la plateforme Github. Il est accessible au lien suivant : <https://github.com/Seni-J/Student-Invaders> ~~Décrire la réalisation "physique" de votre projet~~

~~• les répertoires où le logiciel est installé~~
~~la liste de tous.~~ Cela permet un suivi constant au chef de projet à chaque mise à jour de l'application.

Le répertoire se présente ainsi :

 Code	Ajout des boxes "dynamiques" + Doc
 Documentation	MaJ dossier
 Sprites/Inkscape	les mots peuvent être pris et un avion peut être lancé.
 .gitattributes	Initial commit
 .gitignore	Ajout de la documentation
 README.md	Mise à jour du README

- Le dossier **Code** contient le code source du projet. On y retrouve donc les fichiers ~~et une rapide description de leur contenu (classe Java).~~
- Le dossier **Documentation** contient le cahier des ~~noms qui parlent !)~~ charges, le journal de travail, la documentation, les maquettes, les différents diagrammes ainsi que les anciennes versions de la documentation.
 - ~~• les versions des systèmes d'exploitation et des outils logiciels~~
- Le dossier **Sprites/Inkscape** contient les différents sprites que j'ai créé tout au long du projet.
 - ~~• Le fichier **README.md** contient le manuel d'installation ainsi qu'une description exacte du matériel~~
 - ~~• le numéro de version de votre produit !~~
 - ~~• programmation et scripts: librairies externes, dictionnaire des données, reconstruction du logiciel - cible à partir des sources~~ projet.

NOTE : Evitez d'inclure les listings des sources, à moins que vous ne désiriez en expliquer une partie vous paraissant importante. Dans ce cas n'incluez que cette partie... Les logiciels utilisés ainsi que les versions :

<u>Nom</u>	<u>Version</u>
<u>Android Studio</u>	<u>3.0.1</u>
<u>LibGDX</u>	<u>1.6.1</u>
<u>GitHub Desktop</u>	<u>1.2.0</u>
<u>Inkscape</u>	<u>0.92.2</u>
<u>Samsung Galaxy Tab 2</u>	<u>4.2.2</u>
<u>Mockflow (web)</u>	-
<u>Trello (web)</u>	-

=référence sur le repo Git + description arborescence

=Explication d'éventuelle « spécialité » d'implémentation

Echéance 3

3.2 Description des tests effectués

Pour chaque partie testée de votre projet, il faut décrire:

- les conditions exactes de chaque test
- les preuves de test (papier ou fichier)
- tests sans preuve: fournir au moins une description

= Tableau scénario / date. Exemple :

Scénario	10.5	15.5	22.5	22.5
1.3 Créer utilisateur	Dév → OK	CdP → OK	CdP → OK	Dév → OK
1.4 Modifier utilisateur	Dév → OK	CdP → KO	CdP → OK	Dév → OK
1.5 Suppression utilisateur	Dév → KO		CdP → OK	Dév → OK
2.1 Démarrage simulation			CdP → OK	Dév → OK
2.2 Publier les résultats			CdP → OK	Dév → OK

Echéance 4

Echéance 5

3.3 Erreurs restantes

S'il reste encore des erreurs:

- Description détaillée
- Conséquences sur l'utilisation du produit
- Actions envisagées ou possibles

Echéance 5

3.4 Liste des documents fournis

Lister les documents fournis au client avec votre produit, en indiquant les numéros de versions

- *le rapport de projet*
- *le manuel d'Installation (en annexe)*
- *le manuel d'Utilisation avec des exemples graphiques (en annexe)*
- *autres...*

Echéance 5

4 Conclusions

Développez en tous cas les points suivants:

- *Objectifs atteints / non-atteints*
- *Points positifs / négatifs*
- *Difficultés particulières*
- *Suites possibles pour le projet (évolutions & améliorations)*

Echéance 5**5 Annexes****5.1 Résumé du rapport du TPI / version succincte de la documentation****5.2 Sources – Bibliographie**

Liste des livres utilisés (Titre, auteur, date), des sites Internet (URL) consultés, des articles (Revue, date, titre, auteur)... Et de toutes les aides externes (noms)

5.3 Journal de bord

Date	Événement
17.05	Revue du Sprint 1. L'affichage du vocabulaire n'est pas fait, il est reporté au sprint suivant.
22.05	Visite de Mr. Tièche pour voir l'avancement du projet.
24.05	Revue du Sprint 2. L'envoi des avions et la prise des mots ne sont pas fonctionnels. Les scénarios ainsi que le MCD/Vue d'ensemble ont été revus et validés. Une deuxième démo a lieu le 29 Mai pour montrer l'envoi des avions ainsi que la prise des mots.

Référence au journal de travail externe. Inclus ici seulement si c'est exigé par l'expert.

Echéance 1
Echéance 2
Echéance 3
Echéance 4
Echéance 5

5.4 Manuel d'Installation

Important !

Echéance 4 (Readme dans Git)

5.5 Manuel d'Utilisation

Pas important (pour XCL). Ou plutôt : pas prioritaire

5.6 Archives du projet

Media, ... dans une fourre en plastique

