Aminata DIAGNE _____ Oumou DEMBELE
2022-2023

Projet Tux Letter Game

Sommaire

Introduction

Le principe du jeu est de créer un jeu utilisant un environnement 3D, permettant l'apprentissage ludique de l'orthographe en cherchant les lettres d'un mot avec un personnage, tout cela dans un temps imparti. Le jeu est configuré grâce à une page xhtml et toutes ces données sont stockées via des fichiers xml bien formés et valides. A terme, le jeu se présente sous la forme de 2 applications : une application serveur fournissant les mots d'un dictionnaire, et une application cliente, le jeu lui-même.

- **Objectif du jeu :** L'objectif de ce projet à été de développer le jeu en 3D autour de différentes technologies comme ***Java*** (pour l'environnement 3D, les classes métiers etc...), ***XML*** (pour stocker certains paramètres du jeu et des données de jeu), ***XSLT*** (pour l'affichage***)***…

Durant une partie vous incarnez un personnage (Tux) qui doit aller récupérer toutes les lettres d'un mot sur un plateau tout en prenant compte du temps qui est elle limitée (selon le niveau de difficulté).Il existe différents niveaux de difficultés (de 1 à 5) qui se solde par une contrainte de temps de plus en plus court et un mot plus long dans le plateau de jeu.Le joueur a aussi la possibilité d'enregistrer son score dans un base de donnée en XML et rejouer plus tard avec le même "compte".

- **Création du jeux:**

Dans un premier temps, nous structurons les données du jeu à travers le langage ***xml***.Pour ce faire on créera d'abord des fichiers xml bien formés avant de les contraindre en ***xsd*** .Avec l'utilisation de ***xsl*** ,on s'assurera ensuite de l'affichage des données en ***html***.Les différents fichiers concernés sont:

- dico qui contient l'ensemble des mots.
- profil qui contient les informations personnelles du joueur ainsi que ses parties jouées.
- plateau qui contient les dimensions et la modélisation du plateau

Ensuite, en utilisant le langage Java allons concevoir le jeu , en faisant l'extraction de données avec ***DOM*** et ***SAX*** des documents xml déjà définis .

Étapes de conception:

❖ *Modélisation des données*

L'ensemble des données sont implémenté dans les fichiers (contenu dans leur arborescence respectifs dans le répertoire ***Data***)ci dessous:

- Un dictionnaire ***dico.xml*** contenant un nombre fini de mots. Chaque mot de ce dictionnaire possède un attribut niveau.
- Le plateau ***plateau.xml*** contenant les dimensions utilisées dans la suite pour définir l'environnement graphique et les fichier avec le contenu visuel de cet environnement.

L'ensemble des documents xml sont bien formés et sont contraints par leurs ***xsd*** respectifs.
Pour l'affichage de dico.xml ,on a implémenté deux version de dico.xsl :

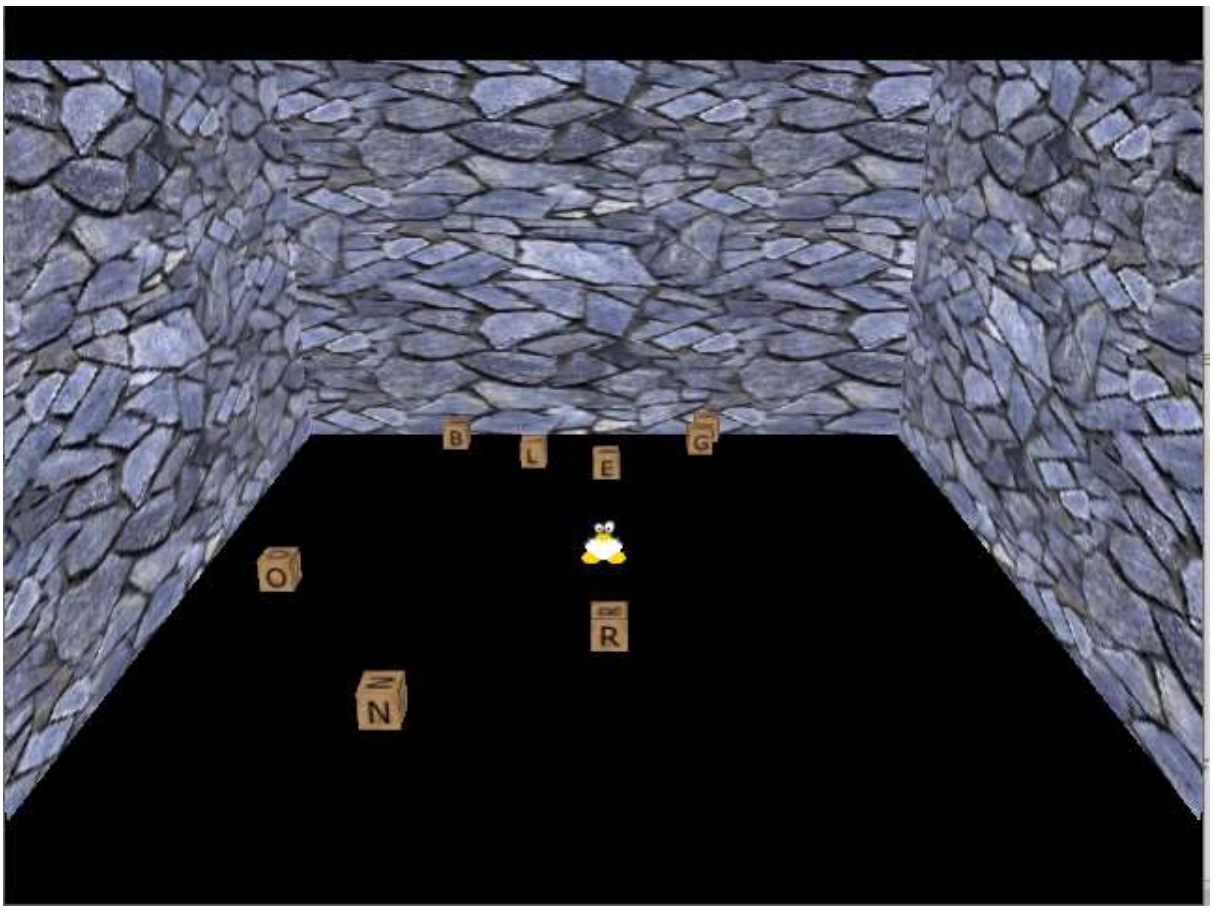
- ☐ ***dico.xsl*** qui se charge de trier le dictionnaire par ordre alphabétique avant de faire l'affichage
- ☐ ***dico_ameliore.xsl*** qui fait le trie alphabétique mais par niveau avant de faire l'affichage

❖ ***Jeu & règles du jeux***

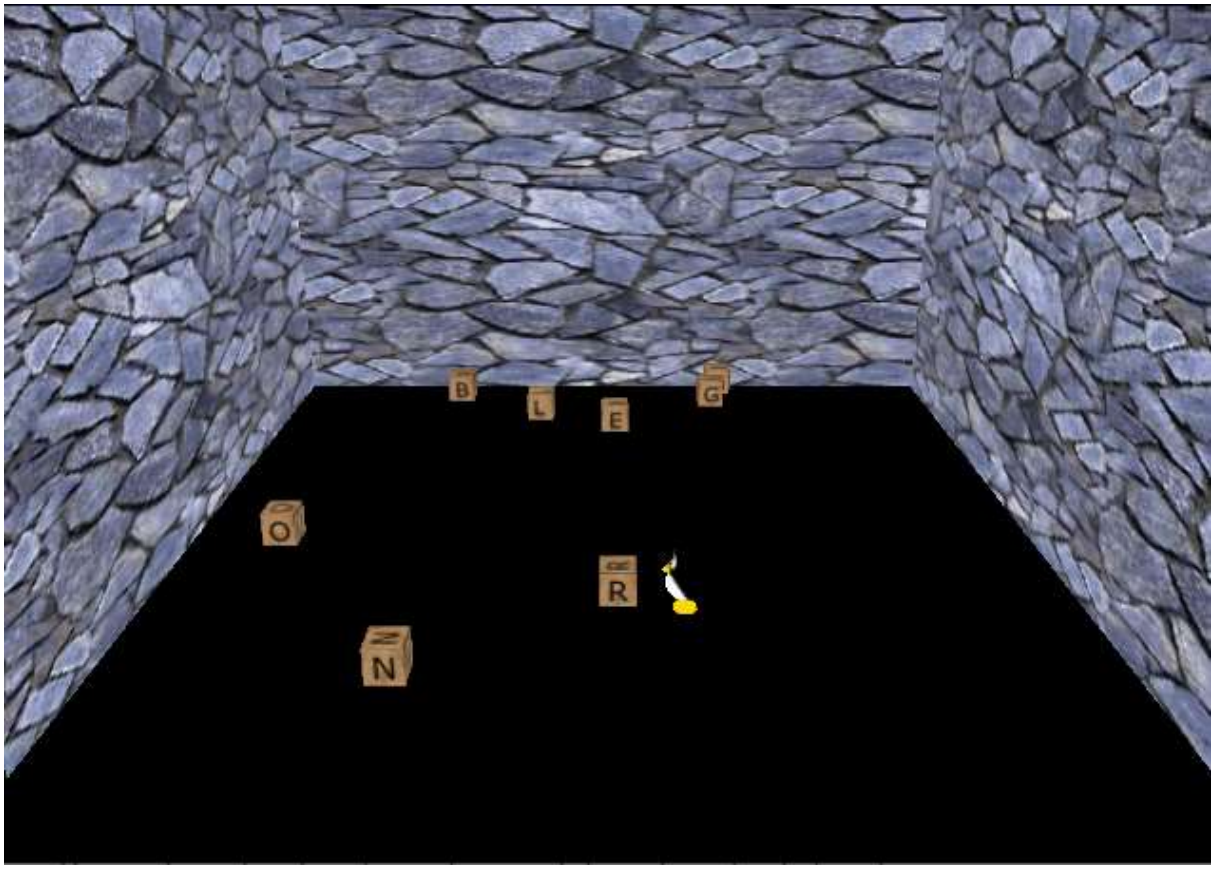
La définition et l'implémentation du jeu se traduiront dans cette partie du projet par l'utilisation du langage ***java*** ,et la création de classes.
Le jeu en lui-même est contenu dans le paquetage ***Game*** qui admets egalement les classes

- ***Tux*** qui modélise le personnage du jeu ,
- ***Room*** qui modélise l'environnement de jeu,
- ***Dico*** qui admet l'ensemble des méthodes pour l'usage et l'extraction des mots contenu dans ***dico.xml***
- ***Profil*** qui modélise le profil du joueur
- ***Chronomètre*** qui se charge du chronométrage du jeu
- ***Letter*** qui se charge de l'affichage des lettres du mot dans l'environnement

Pour exécuter une partie, il suffit de ***RUN*** le fichier ***LanceurDeJeu.java***.
On verra après le lancement l'affichage avec Tux au milieux du plateau et des lettres dispersés dans l'environnement de jeu comme ci dessous:


B
L
E
G
O
R
N
N

Tux pourra avancer et cherchera ainsi à former le mot en récupérant les lettres une par une:



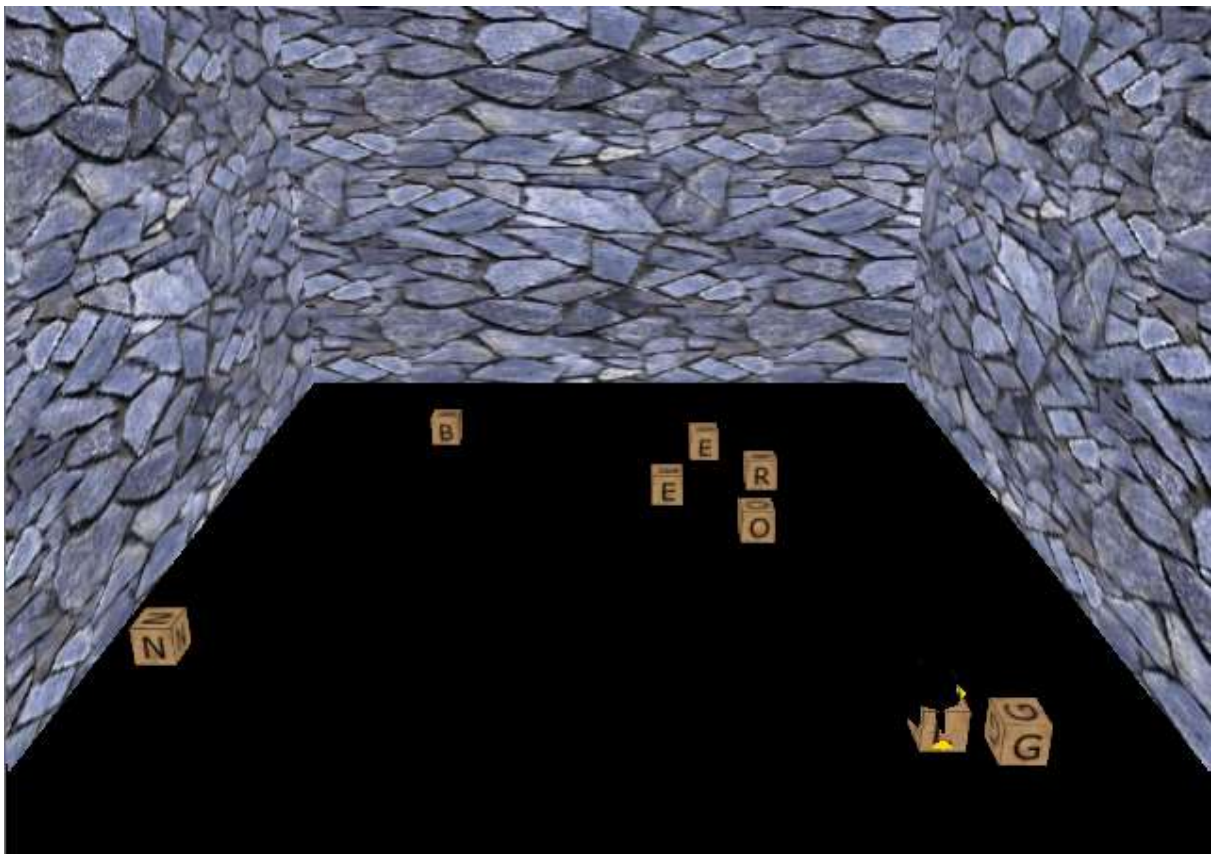
fonctionnement du jeu :Les règles du jeu sont similaires au jeu initialement proposé, les lettres seront disposées aléatoirement dans l'environnement tux de la partie, le joueur doit donc rassembler les lettres du mot en temps imparti. Il devra donc récupérer chaque lettre dans le bon ordre pour reconstituer le mot.

❖ *Extraction de données (DOM & Sax)*

Pour extraire les données notamment dans dico afin de les placer sur l'environnement ,on a parser le document avec DOM en raison de son utilisation simple.
En effet, pour le fichier dico.xml, nous avons codé une méthode ***lireDictionnaireDOM*** pour récupérer les mots du dictionnaire afin de les ajouter dans les ArrayListes Niveau selon leur niveau de difficulté.

Difficultés:

Nous n'avons pas pu réaliser l'ensemble des travaux obligatoires demandés notamment l'usage de ***Sax*** pour parser un document.
Aussi ***Tux*** et l'ensemble des règles sont définis dans l'environnement de jeux mais restent à implémenter correctement son application dans l'environnement.En effet ,***Tux*** n'arrive pas a enlever la lettre lors d'une collision comme voici :



Néanmoins les contraintes à savoir que ***Tux*** ne peux sortir des limites de l'environnement sont appliquées.
La suite du projet n'est pas faite suite à ce blocage mais allons néanmoins continuer sa conception pour notre apprentissage personnel.

Conclusion:

Pour conclure sur ce projet, nous avons beaucoup apprécié passer du temps dessus car cela nous a permis d'assimiler les différentes notions en XML et Java vu en cours.Ce projet nous a également permis appliquer ces notions concrètement et nous sommes plutôt satisfait de l'avancée de notre travail même si on l'a pas globalement fini.
Toutefois, si nous avions eu un peu plus de temps, nous aurions voulu aller un plus loin dans ce projet.