Comparaison des technologies

Pour pouvoir réaliser notre projet, le clone du jeu The Binding of Isaac, nous avons le choix à la fois entre de nombreux langages de programmation, de Framework ou même d’IDE.

Traditionnellement pour ce genre de projet les langages de programmation de haut niveau sont utilisés. Parmi tous ceux disponibles nous pouvons citer le Java, le C#, le C++, Python Ruby et encore bien d’autre.

Pour le développement de l’application, notre client souhaite que le tout soit codé en Java. Si cela correspond à notre premier critère, nous pouvons cependant nous demander s’il cela reste cohérent et adéquat à notre projet.

Grâce à notre cursus, nous connaissons et avons l’habitude de manipuler du Java, du C# et du C++. Nous allons donc comparer ces trois langages entre eux et laisser de côté les autres. En effet, apprendre un nouveau langage pourrait nous prendre trop de temps et être contre-productif dans la réalisation du projet alors que ceux que nous connaissons pourrait être suffisant.

Une première chose qui les différentie est leur date de création : C++ a été créé dans les 1983, 1999 pour Java et 2001 pour C#. Les trois sont encore mis-à-jour C++ est donc le plus ancien et pourrait bénéficier d’une documentation plus complète au vu de son âge, mais pourrait souffrir d’une syntaxe vieillissante. Concernant Java et C#, ils sont plus récent mais ils sont cependant plus utilisé et possède une communauté plus présente et active.

Une autre différence entre ces trois langages est que les projets en Java et C# sont indépendants de la plateforme sur laquelle ils se trouvent. Cela veut dire que nous pouvons coder une seule application et elle pourra s’exécuter sur n’importe quelle machine qu’importe son système d’exploitation. En revanche, il faut que l’ordinateur possède un interpréteur de ce langage qui va s’occupe de le traduire en langage que la machine pourra directement comprendre.

Prenons l’exemple de Java. Un fois le code écrit, il est compilé pour pouvoir être exécuté. Le compilateur converti le Java en bytecode. Ce byte code peut alors être exécuté sur n’importe quel système exploitation si ce dernier possède un Java Runtime Environment (JRE) compatible. Il est a noté, malgré tout, que Java s’installe plus facilement sur différent support puisqu’il est implémenté de base dans les systèmes sous Android.

Une autre différence importante est sur le temps d’exécution. Pour les projets les plus conséquents, utiliser C++ présente un grand avantage puisqu’il s’exécute plus vite que C# et Java su fait qu’il ne passe pas par une machine virtuelle. Cependant pour la taille de notre jeu, même si nous utilisons Java qui est plus lent, cela ne nous posera pas de soucis, ne nous impactant pas.

Finalement, nous pouvons voir une tendance se dessiner avec nos choix. Java et C# semble assez semblable, comparé à C++. Cela vient de leur date de création, très proche, mais aussi de la raison de la création de C# qui a été lancé pour concurrencer Java en proposant une version amélioré qui lui-même devait être une amélioration de C++.

Nous nous orientons plus favorablement vers Java ou C#. Si C# semble être plus privilégié pour réaliser des projets vidéos ludiques, Java peut être intéressant à choisir puisque, dans une situation réelle, il pourrait être plus facile de recruter des personnes pour travailler sur notre projet. En effet, Java est gratuit et plus facilement accessible que C# qui est détenu par Microsoft.

Maintenant que nous savons que Java est adaptée à ce que nous souhaitons faire, nous pouvons nous demander si nous allons utiliser un Framwork, une bibliothèque regroupant de nombreuses fonctions pré-codées, très utile pour éviter de recoder des méthodes et pour potentiellement développer plus vite notre projet.

Pour Java, de nombreux Framexork sont adaptés pour la création d’un jeu comme par exemple LibGDX ou Slick2d. Cependant notre client ne souhaite pas avoir recours à ces framework, cependant est-ce dérangeant ?

Pas vraiment, puisque si un framework peut faire gagner du temps, apprendre à le maîtriser et à savoir l’utiliser peut couter un temps précieux. Dans notre cas, puisque nous n’avons jamais utilisé ces Framework auparavant nous risquons de ne pas savoir les utiliser correctement et de ne pas être aussi efficace que si nous codions toutes nos fonctions. Nous n’en n’utiliserons donc pas.

Finalement, en ce qui concerne les IDE, nous avons le choix entre plusieurs IDE, Eclipse, IntelliJ IDEA Netbean.

Si IntelliJ est l’IDE le plus prisé par les développeurs Java, le fait qu’il soit payant est un frein pour que nous puissions l’utiliser.

Nous avons donc NetBean, que nous connaissons bien puisque nous l’avons toujours utilisé, nous savons donc comment l’utiliser.

Tout comme Eclipse, il possède des outils de de débogage et de test. Toutefois, Netbeans est plus moderne dans son apparence et plus intuitif.

Nous utiliserons donc NetBeans pour coder notre projet en Java, sans Framework.