


Introduction



Programmation de Jeux

Différents domaines de programmations de jeux :

- **Gameplay** : Règles du jeu
- **Moteur** : Coeur de jeu mutualisé entre projets
- **3D** : Rendu graphique et effets visuels (shader)
- **UI** : Interface graphique et navigation
- **Outils** : Aides à la conception et au design
- **Réseau** : Interactions clients / serveurs, web, ...
- **IA** : Analyses de situations et prises de décision

Programmation Gameplay

Programmation gameplay = construire une boîte de jeu et écrire un livret de règles.

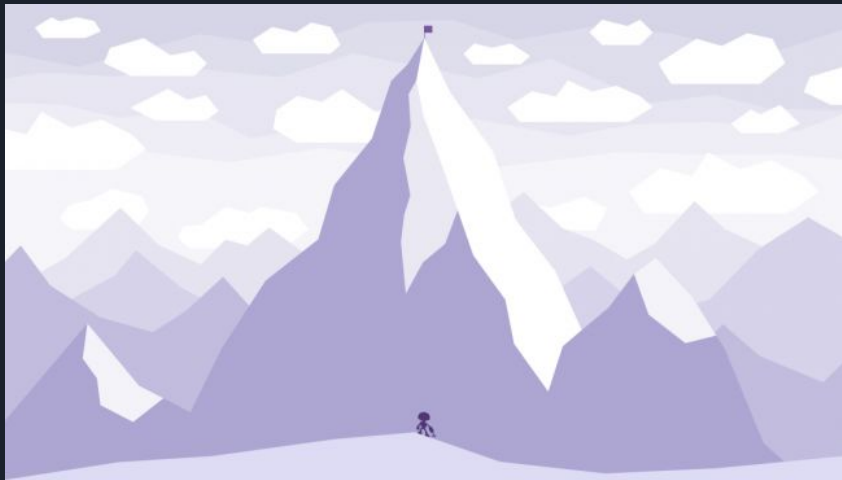


Méthodologie

"**Diviser chacune des difficultés** en autant de parcelles qu'il serait requis **pour les mieux résoudre.**" - Descartes



Pour Gravir une Montagne...



**Il ne faut pas regarder le
sommet...**



mais atteindre le prochain Git !

Quand On Architecture Bien ...



On peut isoler un élément...



sans tirer tout le plat de spaghetti !



Méthodologie

Avant de se lancer dans la programmation, toujours commencer par identifier :

- Les **éléments** de jeu et leurs **données**
- Le **déroulement** d'une partie
- Les **règles** de jeu

Plus cette analyse est précise, plus la programmation sera facile !



Méthodologie

Comme en game-design, on dissocie le **core-gameplay** des **mécaniques satellites**.

On commencera toujours par implémenter le core-gameplay avant d'ajouter de nouvelles mécaniques.

Exercice d'Analyse



Outils

Le minimum pour programmer :

- Texte et Compilateur (langages compilés) :



- Texte et Machine virtuelle (langages interprétés) :





Outils : IDE

Un environnement de développement intégré (IDE) comprend entre autre :

- Un éditeur de code avec **coloration syntaxique**,
- Un outil de **gestion de projet**,
- Un **compilateur intégré** avec configuration simplifiée,
- Un **détecteur d'erreurs**,
- Un outil de **débogage**,
- L'**auto-complétion**,
- Le **refactoring**,
- **La magie !**



Outils : IDE

Quelques IDE pour le C++ :

- **Visual Studio**
- Eclipse
- Code::Blocks
- CodeLite
- Dev-C++



C++

Pourquoi le C++ ?

Adapté au jeu vidéo :

- Compilé donc performant,
- Robuste,
- Complet,

Éprouvé et populaire :

- Large communauté,
- Nombreuses bibliothèques,
- Bien documenté.