Université de Montpellier

IUT de Béziers

Département MMI

SAE301

Benoît DARTIES /Nicolas Maurin

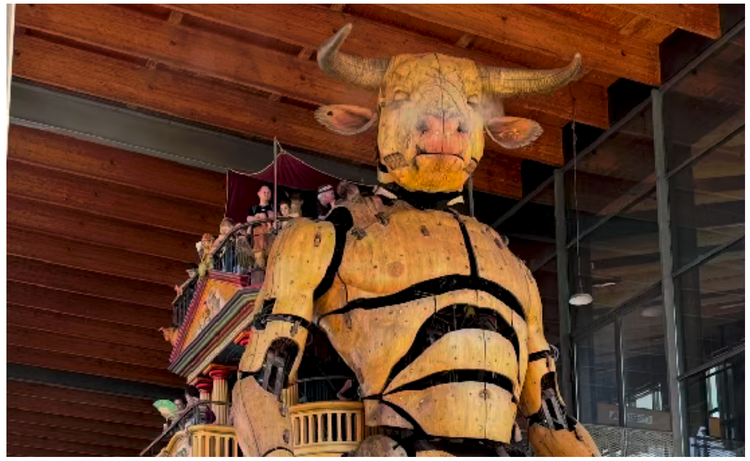
Année Universitaire 2025 - 2026

Objectif de la SAE

Réaliser un jeu video jouable sur borne d'arcade

- À réaliser sur **3** semaines (non consécutives)
- Groupe de **3** étudiants : **un chef de projet** nommé par le groupe
- Technologie : Phaser
- Sujet à choisir parmi **3** thèmes imposés

Thèmes SAE



UNIVERS STEAMPUNK

RETOUR EN 1900

DONGEONS ET DRAGONS

Réaliser un jeu complet, fonctionnel et agréable à jouer avec Phaser

Contraintes de réalisation : grande liberté créative

- Histoire et scénario originaux
- Visuellement agréable, ludique
- Bonne expérience utilisateur : prise en main intuitive / guidée
- Mécanismes de jeu de votre choix
  - platformer "classique" 👍🏻
  - jeu vu de haut 🤔
  - Shoot'em Up 👾
  - 3D 🤯
  - MetroidVania ❤️
  - roguelikes
  - ...

Réaliser un jeu complet, fonctionnel et agréable à jouer avec Phaser

Contraintes techniques :

- Utilisation de Phaser - avec ou sans calques de tuiles -
- Code construit à partir d'un template communiqué sur Moodle
  - Vous pouvez choisir d'utiliser les mécanismes présents sur les TP de R312. Ou pas.
  - Les assets (video, son, spritesheet, tileset...) peuvent etre récupérés sur internet, créés par vous-même, récupérés d'un travail précédent :

Utilisation de l'IA :

- Utilisation **Intelligente et raisonnée**
  - <u>Doit vous permettre d'aller beaucoup plus vite, pas de vous remplacer</u>
  - Sur du code : comprendre le code retourné, savoir l'expliquer
  - Sur des Assets (images, video, sons...) : open bar

Déroulé général

Montée en compétences

Test des démos par les étudiants : feedback argumentés

Evaluation par les pairs + enseignants

Semaine 1
Prototypage

Définition du jeu, organisation et mécanismes envisagés, création des assets.

Attendu : prototype sommaire avec mécanismes de jeu

Identification des connaissances non maitrisées à acquérir

Semaine 2
Démo jouable

Réalisation d'une démo jouable : 1er niveau, assets temporaires, tests d'intégration sur la borne d'arcade, tests UX ...

Semaine 3
Finalisation

Prise en compte des retours
Finalisation du jeu et des livrables, finalisation de l'intégration

Organisation générale :

Enseignants présents

- Nicolas Maurin / Benoit Darties
- Rôle de **facilitateur** : réponses aux questions / soucis de code
- Mini-cours en cas de difficulté

Présence obligatoire sur l'IUT !

- Salles mises à disposition sur l'emploi du temps

- **Débrief global chaque début de semaine + éventuelles consignes supplémentaires**

Livrables attendus

Livrables à déposer sur Moodle chaque jour :

- Rapport de suivi journaliers : à déposer par le **chef de projet uniquement** chaque soir à **20h00 max**
- 3 paragraphes : Ce que vous avez fait ce jour, ce que vous comptez faire la prochaine fois, et remarque éventuelles

Livrables à déposer sur Moodle en fin de projet :

- **Note technique en PDF** : 5-6 pages environ (ou plus), présentant ce que vous avez réaliser, les points forts de votre jeu, les améliorations possibles, des éléments techniques, les prompts IA utilisés
- **Archive zip contenant l'intégralité de votre projet : respecte l'organisation des fichiers / templates**
  - Jeu construit à partir du template donné, contenu dans un répertoire principal
    - Le répertoire principal doit contenir :
      - Un fichier **index.js** : point de départ de votre jeu
      - Un fichier **demo.mp4** : 10 secondes de jeu, qui vont tourner en boucle pour la présentation du jeu
      - Un fichier **presentation.png** : image "jaquette" de 800x450 pixels
      - Un fichier game.json rempli avec les données de votre jeu : format imposé (diapo suivante)

Exemple

```
{
    "title": "SUPER JEU",
    "description": "Super jeu de la mort qui tue.",
    "genre" : "action / aventure",
    "authors": ["Benoit Darties", "Nicolas Maurin"]
}
```

3 notes

Technicité et compréhension coef 1	**Notation par les enseignants** - Mode : Démo en fin de SAE - Critères évalués : fonctionnalités, mécanismes techniques, propreté du code, réponses aux demande d'explication de code
Expérience de jeu coef 1	**Notation par les étudiants** - Mode : Test sur la borne après la SAE - Critères évalués :compréhension, design, jouabilité, musiques, histoire, UX - **Notes harmonisées avec la moyenne / l'écart type de la note technique**
Gestion de projet coef 1	**Notation par les enseignants modulée avec évaluation par les pairs** **Critères évalués :** - Réalisations du groupe : Comptes rendus journaliers, rapport final - Réalisation individuelles : feedback+ évaluation des jeux, et qualité de ces derniers. - Auto-évaluation des membres de son groupe + **Bonus 3 points du Chef**

Une note sur l'affichage et les assets

Taille et résolution :

- Affichage de la borne : 16/9eme
- Résolution du jeu : **1280 x 720 pixels (HD / 16/9eme, équivalent switch portable, PS3 ou Xbox 360)**
  - Une image plein écran ne devrait pas dépasser **200Ko !**
  - Un son : **16 ko / seconde**
  - Une texture de sprite / une frame d'un spritesheet : ~ **10-20 Ko**

Intégration des assets :

- Chemins des répertoires : relatifs seulement! Pas de répertoire absolu
  - `this.load.image("img_ciel", "./assets/sky.png");` **OK**
  - `this.load.image("img_ciel", "/src/assets/sky.png");` **PAS OK**

Utilisation d'assets trouvés sur internet

- **OK si conditions d'exploitation conformes! (Libre droits, copyright avec mention des auteurs, utilisation non commerciale ...)**

Le moindre asset non libre de droit : ZERO

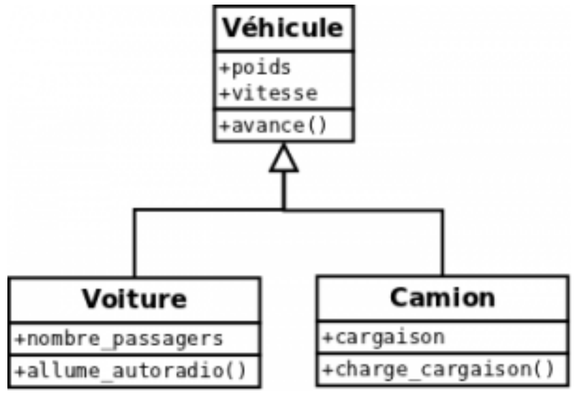
La moindre image pompée à plus de 30% sur une image non libre de droit : ZERO

Le moindre problème avec une image / son ne respectant pas le droit à l'image : ZERO

Mapping des boutons

En cours de finalisation, communiqué dans la journée via le template de code

Comment faire le meilleur jeu possible : les 3 règles d'or



Montée en compétences

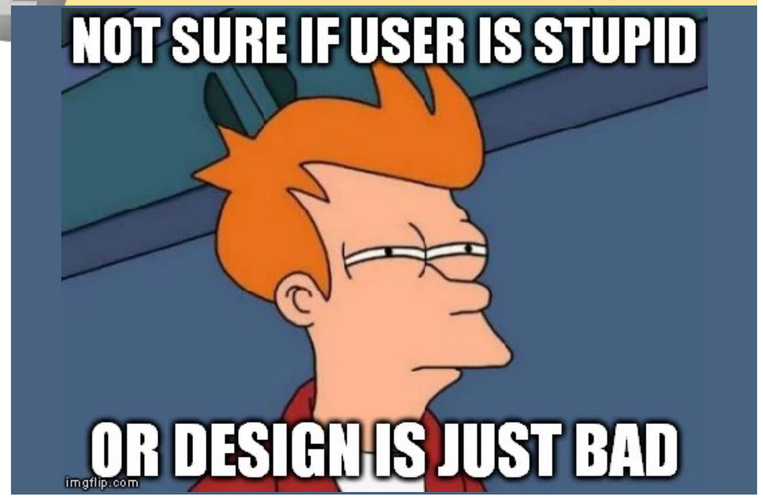
Tutoriels mis à disposition :

- Tuto 1 : Création d'un premier jeu
- Tuto 2 : création d'une carte de jeu
- Tuto 3 : ajout d'une fonction de tir
- Tuto 4: création de plusieurs niveaux
- + plein de petit tutos annexes
- + documentation en ligne :
  - https://newdocs.phaser.io/docs/3.60.0
  - darties.fr
  - http://labs.phaser.io/

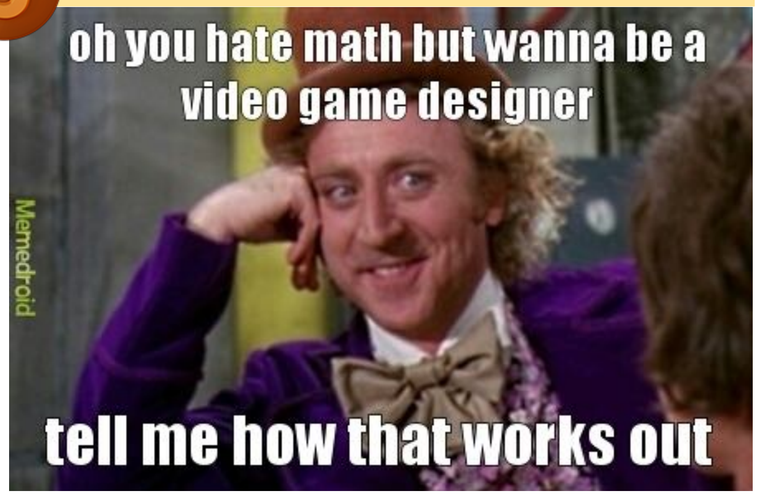
Les 3 règles d'or

1. Définir précisément le jeu, son périmètre, ses limites

2. Placer l'expérience utilisateur au centre du jeu



3. Mettre des maths dans le jeu



Les 3 règles d'or

Liste pas du tout exhaustive!

1 Définir précisément le jeu, son périmètre, ses limites

Objectif de votre jeu

- De quoi parle globalement votre jeu
- Quel est l'objectif : finalité? Raccord au scénario.
- Système de score?

Réflexion sur la durée de vie

- Combien de niveaux? Conditions de passage
- Points de vie? Nombre de vies? Mécanisme de perte / gain de vie.

Présence d'ennemis?

- Apparition? Déplacement? Caractéristiques vitales ? Tirs ? À quelle fréquence?
- Comment s'en débarrasser. Armes ?
- Respawn ? Fréquence?

Liste de tous les mécanismes que vous souhaitez implémenter

- Ex : Présence d'item ? Effets et durée? Obtention ? Placement sur la carte? Loot?

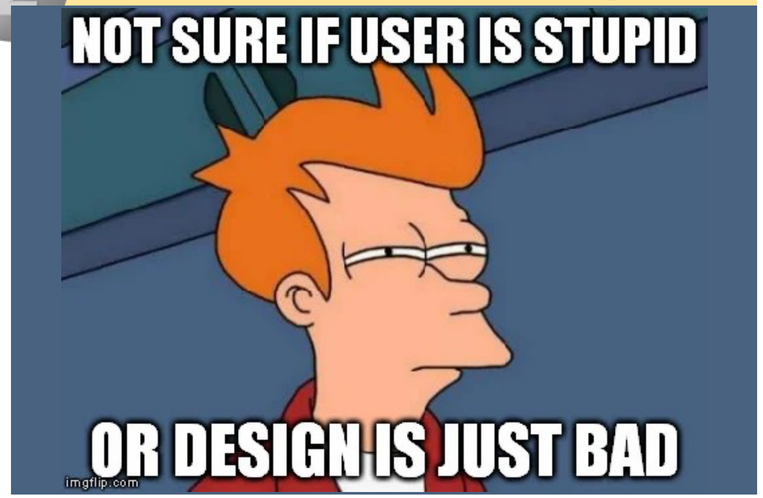
Level design!!!

- Réalisation d'un croquis de toutes les cartes, écrans de jeu, détailler les transitions entre les écrans...
- **Erreur courante : se lancer directement dans le code, et rajouter les fonctionnalités au fur et à mesure.**

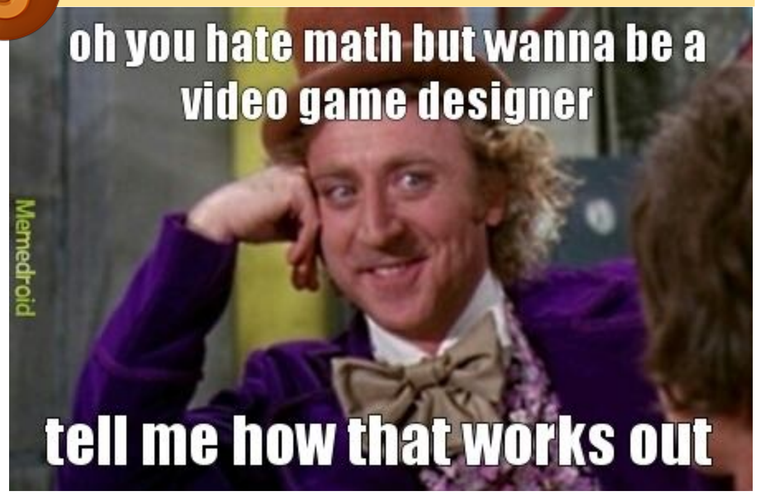
Les 3 règles d'or

1. Définir précisément le jeu, son périmètre, ses limites

2. Placer l'expérience utilisateur au centre du jeu

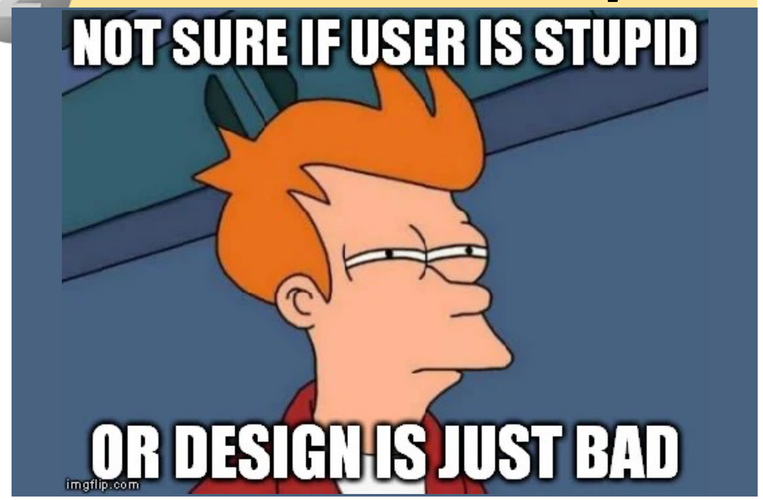


3. Mettre des maths dans le jeu



Comment réaliser un bon jeu

Placer l'expérience utilisateur au centre du jeu



L'utilisateur ne connait pas votre jeu

- Préciser les règles, l'objectif dans le jeu, les fonctionnalités.
- **Accompagnement**
- **Commandes de jeu?**
- Présence d'Ecrans de jeu ? Aide?

Pourquoi l'utilisateur aurait envie de jouer à votre jeu

- Pourquoi c'est un bon jeu?
- Public ciblé?

A-t'on envie de continuer à jouer au bout de 3 minutes?

- A-t'on envie de **recommencer** votre jeu une fois mort?
- Chaque nouvelle partie est-elle différente de la précédente?
- Comprend-t'on pourquoi on est mort (ex : Die and Retry?)
- Identifier les points de friction utilisateur

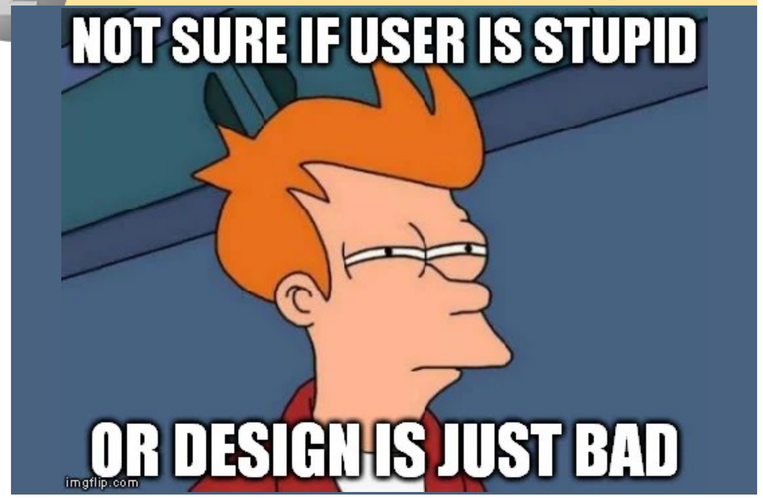
Travailler l'immersion

- Esthétique, sons, ambiance
- Enchainement entre les écrans

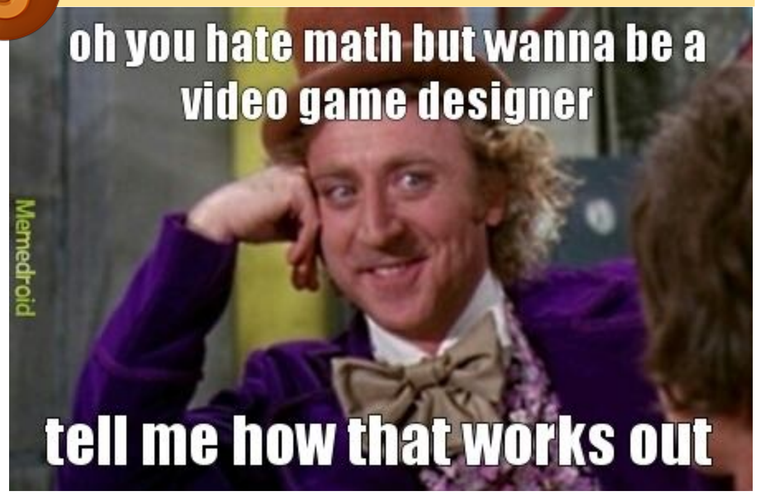
Les 3 règles d'or

1. Définir précisément le jeu, son périmètre, ses limites

2. Placer l'expérience utilisateur au centre du jeu



3. Mettre des maths dans le jeu



Comment réaliser un bon jeu

3 Mettre des maths dans le jeu

Absolument tout est une histoire de chiffres, de vecteurs, de fonctions mathématiques, de timers et de fréquences

- Paramètres physiques : Vitesse de déplacement, de saut ...
- Cadence et vitesse de tir
- Points de vie des monstres
- **Difficulté progressiv**e : Augmenter progressivement la difficulté en modifiant les attributs des ennemis ou du jeu.
- **Zones d'influence** : Définition des périmètres dans lesquels les ennemis détectent le joueur ou modifient leur comportement.

Mettez de l'aléatoire!

- **Répétabilité** : L'aléatoire augmente la rejouabilité en modifiant des éléments clés du jeu à chaque partie, rendant chaque session unique.
- **Adaptation** : Il force le joueur à s'adapter constamment, créant des défis nouveaux à chaque partie.
- Surprise et imprévisibilité : Cela ajoute un effet de surprise qui rend l'expérience plus engageante et immersive.

doit être utilisé de manière équilibrée pour éviter la frustration, tout en apportant de la diversité et du dynamisme dans le gameplay

Les 3 règles d'or

Apparition d'ennemis et de monstres :

- Fréquence d'apparition : Temps aléatoire entre les vagues ou les apparitions de monstres, gardant le joueur sur ses gardes.
- Position d'apparition : Les monstres peuvent apparaître à différents endroits à chaque partie, rendant l'exploration moins prévisible.
- Type d'ennemi : Choisir aléatoirement entre plusieurs types d'ennemis pour augmenter la variété des combats.

Objets et récompenses (Loot) :

- Items dropés : Les objets laissés par les ennemis vaincus peuvent varier en fonction de probabilités définies (armes, potions, etc.).
- Rareté des items : Les chances d'obtenir un objet rare (basé sur des pourcentages) peuvent rendre le jeu plus excitant.
- Quantité d'items : La quantité de ressources trouvées (or, munitions, etc.) peut être aléatoire pour forcer le joueur à s’adapter.

Les 3 règles d'or

Niveaux et environnement :

- **Génération procédurale des niveaux** : générés de manière procédurale (roguelikes, par exemple) pour que chaque partie soit différente.
- Position des pièges et obstacles : La disposition des pièges peut changer à chaque tentative pour rendre la progression moins linéaire.

Caractéristiques des personnages et ennemis :

- **Points de vie et statistiques** : Variation légère des points de vie, de la vitesse ou de la puissance des ennemis à chaque combat.
- **Compétences des ennemis** : Choisir aléatoirement parmi plusieurs compétences ou attaques pour les ennemis afin de rendre leurs mouvements moins prévisibles.
- **Équipement des personnages** : Certains jeux peuvent inclure des éléments aléatoires dans les équipements de départ ou les attributs des personnages.

Ultimes conseils

Trouvez le bon équilibre technicité / temps disponible

Ne confondez pas simplicité du jeu, et simplicité de la réalisation

Faites tester vos prototypes, acceptez la critique : le client est roi