

Introduction à la Programmation Objet :

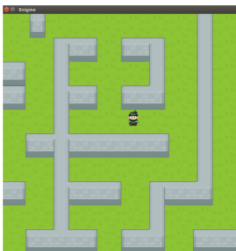
Mini-projet 2

Laboratoire d'Intelligence Artificielle
Faculté I&C

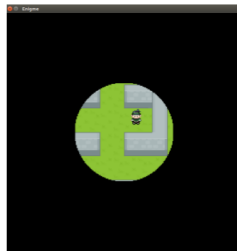
«Mini-projet» 2 : les origines



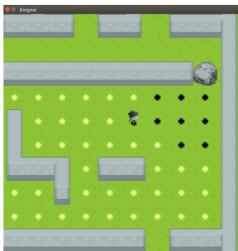
(a)



(b)



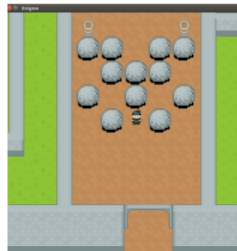
(c)



(d)



(e)



(f)

«Mini-projet» 2 : au fil du temps

La maquette a permis de réaliser de nombreuses instances de jeux :

- ▶ ZeldIC
- ▶ IC Super Pacman
- ▶ ICWars
- ▶ IC Rogue
- ▶ ICMon
- ▶ ...

et cette année ... ICoop :)

Objectifs : appréhender les avantages et limitations de l'approche orientée-objet

- ▶ en se plaçant à des niveaux d'abstraction suffisamment généraux pour permettre des réutilisation intéressantes
- ▶ en intégrant du code existant

Mini-projet 2 : pourquoi des jeux ?

- ▶ Terrain d'expérimentation très fertile pour l'orienté objet
- ▶ Un nombre infini d'évolutions possibles
- ▶ Chaque évolution sous-tend souvent des problématiques de conception intéressantes
- ☞ Le but est moins le résultat obtenu que ce que l'on apprend sur le chemin pour y parvenir :)

Consignes de base

Une archive est fournie avec déjà beaucoup de code et des ressources.

- ▶ Le code de la maquette (`game-engine`) ne doit pas être modifié
- ▶ Ne pas utiliser de paquetages non standards
- ▶ Utilisez un dépôt privé si vous utilisez `git`
- ▶ L'ajout de ressources est possible, mais le portail limite l'archive à 4MB
 - ☞ Contactez l'équipe du cours pour une dérogation
- ▶ Rendu final ouvert
 - ☞ fichiers `HELP` et `CONCEPTION` à prévoir
- ▶ Concours
 - ☞ Petit descriptif à fournir

Consignes méthodologiques

- ▶ Regardez la vidéo sur le schéma des interactions avant le TP de vendredi.
- ▶ Commencez d'emblée le mini-projet **même si vous n'avez pas fini le tutoriel**.
- ▶ Fixez vous des objectifs raisonnables en fonction de votre niveau.
- ▶ Lisez l'intégralité des consignes précédant une tâche avant de vous lancer dans le codage.
- ▶ Sauvegardez régulièrement.
- ▶ Aidez vous du debugger pour comprendre les situations de «crash».
- ▶ Si vous travaillez avec git, poussez sur le dépôt toute tâche fonctionnelle avant de passer à la suivante.
- ▶ Tenez-vous informé.es des «précisions et correctifs».

Délai : **Jeudi 19.12 13 :00**

- ▶ Protocole de rendu analogue au mini-projet 1
 - ▶ Utilitaire fourni pour constituer une archive `.zip`
 - ▶ Soumission sur Moodle
 - ▶ Une soumission par groupe
- ☞ le protocole exact de soumission sera donné en temps voulu
- ▶ Portail de rendu ouvert 10 jours avant la date limite
 - ☞ **Essayez de soumettre dès l'ouverture, même si le projet n'est de loin pas achevé**



Attention ! : vous devez rendre des projets qui compilent. **Seuls les projets compilables seront corrigés.**

Le portail génère un feedback indiquant si le projet compile ou pas.

- ▶ Évaluation fonctionnelle : les détails de mise en oeuvre sont peu spécifiés et peu importants (documentation dans le HELP de choix alternatifs)
- ▶ Les bugs marginaux ou difficile à reproduire ne sont pas pris en considération
- ▶ Évaluation qualitative :
 - ▶ Bonne conception OO :
 - ▶ Appliquez le principe de «code to an interface not to an implementation».
 - ▶ Évitez les failles d'encapsulation.
 - ▶ Évitez la prolifération d'attributs protégés.
 - ▶ Utilisez le `abstract` à bon escient.
 - ▶ Établissez des liens qui font sens entre les classes.
 - ▶ Utilisez l'héritage pour éviter des duplications inutiles de code.
 - ▶ Critères qualitatifs usuels : lisibilité, modularisation, conventions de nommage, cohérence du style.

- ▶ Étape 1 : application assez directe du tutoriel
 - ☞ Application du schéma des interaction du tutoriel III
 - ☞ Parer à l'absence du «multiple dispatch»
 - ☞ Découverte d'un premier «patron de conception»Plus complexe mais très guidé
- ▶ Étape 2 : Premières collaborations entre les deux personnages

Les étapes sont jalonnées de tâches intermédiaires pour **avancer de façon robuste petit pas par petit pas.**

Le code des graphismes et animations est fourni.

Étapes (suite)

- ▶ Étape 3 : Adversaires et batailles
 - ▶ Concept d'«automate à états finis» (logique des combats)
 - ▶ Aspects orthogonaux (parallélisables) : différents adversaires
- ▶ Étape 4 : le 1/2 point pour arriver au 6
 - ☞ Beaucoup moins guidée et va vous «challenger» un peu sur l'encapsulation
- ▶ Étape 5 : 20 points d'extensions à choix
 - ☞ libre et très peu guidé

Particularité cette année : les ajouts sont assez modulaires
(augmente la possibilité de travailler en parallèle)

- ▶ Les assistant.es n'ont pas pour tâche de résoudre vos erreurs d'exécution ;
- ▶ **Il ne faut pas les contacter en privé** (par exemple sur Telegram) : le seul canal de discussion doit être le forum Ed.
- ▶ Ils/elles peuvent vous aider sur les fautes de compilation ;
- ▶ et vous aider à développer des stratégies pour débuser les sources des erreurs d'exécution par vous même ;
- ▶ il y a une équipe limitée en appui : **privilégiez le forum Ed pendant la semaine !**

- ▶ **Ne pas publier votre code** (utilisez un post privé s'il est nécessaire de montrer votre code et ne le faites qu'à notre demande).
- ▶ Avant de montrer du code, essayer de circonscrire les conditions d'occurrence du bug, en nous indiquant par exemple, ce que vous observez dans le debugger.
- ▶ Donnez des **titres parlants** à vos posts.
- ▶ **Cherchez si la question n'a pas déjà été posée.**

Bon codage !