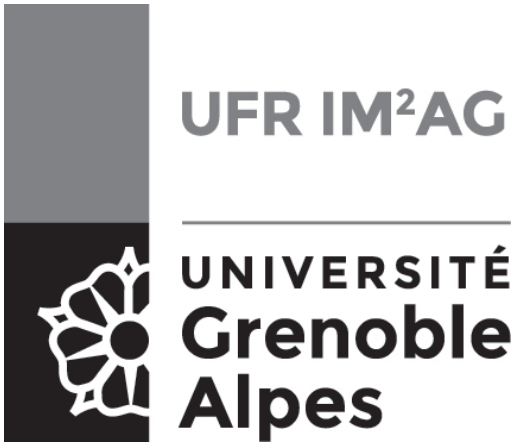

GBIN6U03
Projet logiciel (et social)

Objectif

Réalisation d'un logiciel de jeu

- 1 mois à temps complet
- Réinvestissement des acquis
- Travail en équipe (6 étudiants)
- Autonomie (gestion de projet)
- Taille moyenne (6 HM)

Un jeu

Motivant Cohérent par rapport à la formation

- On retrouve de l'algo (IA) dans les jeux proposés
- Possibilité de jeu en réseau (Système)
- La programmation de l'ensemble requiert un effort de design/spécification, test, intégration, suivi.
- Occasion de découvrir l'IHM

Niveau adapté

Forme

Jeux judicieusement choisis

- Qui amènent à travailler sur les thèmes des UE
- Dont la difficulté de réalisation est du même ordre

Peu de contraintes

- En java 8, avec toute sa bibliothèque standard
- Indépendant de la résolution
- Pas de squelette donné

Support limité

- Support enseignant dans le périmètre des UEs

Déroulement

	Lundi	Mardi	Mercredi	Jeudi	Vendredi
1				Présentation Amphi IA Préprojet (gaufre)	
2			Bilan gaufre Amphi jeux Amphi IHM		Audits IHM1
3	Amphi threads Amphi réseau				
4	Ateliers joueur	Audits code			
5		Audits IHM2		Amphi Soutenances	
6			Rendu	Soutenances	Soutenances Demo+pot

Support additionnel

- Tuteurs (1 par groupe) une fois le jeu choisi
- Hotline : concombre.masque@imag.fr

A rendre

Code source + binaires fonctionnels

- Doit fonctionner en Java runtime environment 8 se
- A rendre dans la salle de soutenance (F108) :
  le 30/05/2018 à 15h15
- A disposition du jury durant la soutenance

2 dossiers de validation

- Un pour l'IHM et un pour l'IA, deux exemplaires chacun
-
- A rendre au début de la soutenance
- Description et justification de votre réalisation
- Synthèse des évaluations effectuées

Evaluation

1/4 Présentation orale

- Clarté, respect du temps, réponse aux questions

1/4 IHM

- Qualité, progression au fil des audits, validation

1/4 IA

- Qualité, validation

1/4 Qualité technique globale

- Robustesse, fonctionnalités

Le préprojet

Gaufre empoisonnée

Objectifs

- Tester le projet sur une petite échelle
- Réfléchir à l'architecture logicielle de votre jeu
- Appréhender le travail au sein de votre groupe

Mise en œuvre

- Deux jours de travail (vendredi et lundi)
- Revue de la part des enseignants mercredi matin

Non noté

Démo

Contraintes de réalisation

IHM :

- Claire : état du jeu, tour, score, fonctionnalités
- Ergonomique : organisation, utilisation

Niveaux pour l'IA

- Aléatoire
- coup gagnant/perdant
- Minimax

Fonctionnalités

- Jeu à deux joueurs sur une même fenêtre
- Historique : annuler / refaire sans limite
- Sauvegarder / charger avec historique
- Nouvelle partie
- Coup aléatoire parmi les meilleurs coups

Conseils d'organisation

Conduite du développement

Identification des besoins

- Triés par priorité
- Temps de développement estimé

Architecture globale

- Arbitre (Moteur de jeu)
- Interface (IHM)
- Joueur (IA)

Spécification

- Relation entre les modules
- Interfaces

Cycle de développement

Privilégier les cycles courts

- Spécification
- Implémentation
- Test

Implémenter plus que nécessaire

- Code de test plus volumineux que le code testé
- Outils, code temporaire
  - Editeur de plateaux de jeu
  - Joueur qui joue une séquence fixe pour tester l'Arbitre
  - Traces d'exécution

Rôles

Correspondent à des tâches globales

- Suivre l'avancement du projet
- Valider chacune des parties au fur et à mesure
- Intégrer l'ensemble des contributions

Typiquement un responsable mais

- peut impliquer plusieurs personnes
- le responsable délègue et collecte les résultats

Constituent les fondations du projet

- Projet fonctionnel
- Réalisation dans les temps

Le rôle de rapporteur

Tenir un journal

- Pistes envisagées / essayées
- Avancée dans les tâches
- Respect du planning

Communiquer régulièrement

- Au niveau de l'ensemble du groupe
- Avec le tuteur

Le rôle de testeur

Tester le code à plusieurs niveaux

- Tests unitaires au sein d'un module
- Tests d'intégration entre modules
- Tests de réponse aux besoins

Conserver et automatiser les tests

- Non régression
- Identification plus aisée des bugs

Tester et faire tester régulièrement

Le rôle d'intégrateur

Intégrer à intervalle réguliers

- Pas de divergence par rapport aux spécifications
- Pas de bug d'intégration

Conserver les versions successives du logiciel

- Facilite la recherche de bug
- Permet de revenir en arrière en cas de fausse piste
- Versions stables utilisées pour les évaluations

Evaluations auprès d'utilisateurs

Définir le public visé : age, activité, expertise, ...
Programmer des évaluations régulières

- Sur les versions successives de votre logiciel
- En conservant inchangée une partie du public

Garder une trace écrite de toute évaluation

- Public testé
- Résultat (votre avis et celui du sujet testé)
- Retour sur l'identification des besoins

L'évaluation de l'IHM

Clarté

- Compréhension des fonctionnalités
- Identification des possibilités

Ergonomie

- Facilité de navigation
- Scénario d'usage typique favorisé

Réponse aux besoins

L'évaluation de l'IA

Bien fondé des niveaux de difficulté

- L'IA difficile bat la moyenne qui bat la facile
- Tester les deux scenarii pour les jeux asymétriques

Comportement contre l'humain

- Adéquation difficulté / expertise du sujet testé
- Absence de comportement pathologique

Temps de calcul approprié

QUESTIONS ?