# Mastère Hybride
## Programmation de Jeux Vidéo

tech université technologique

tech université technologique

# Mastère Hybride

## Programmation de Jeux Vidéo

Modalité: **Hybride (En ligne + Formation Pratique)**
Durée: **12 mois**
Diplôme: **TECH Université Technologique**
Heures de cours: **1.620 h.**
Accès au site web: **www.techtitute.com/fr/jeux-video/mastere-hybride/mastere-hybride-programmation-jeux-video**

# Sommaire

01

# Présentation

Le secteur de la création de jeux vidéo est plus que consolidé et attire de plus en plus de jeunes. L'un des piliers sur lesquels repose un grand titre de l'industrie du jeu vidéo est la programmation. Le procédé qui crée les instructions de base et dicte son fonctionnement général est essentiel pour que l'histoire et le développement du jeu fonctionnent parfaitement. Cette formation d'apprentissage Hybride offre aux professionnels du jeu vidéo une spécialisation qui leur permettra de faire progresser leur carrière dans le secteur du *Gaming*. Il s'agit d'un diplôme 100% en ligne, adapté à vos besoins et qui comprend un stage dans un studio de création et de développement de jeux vidéo. Tout cela vous permettra d'avancer dans votre carrière professionnelle auprès des meilleurs experts.


*Devenez l'un des meilleurs programmeurs de l'industrie du jeu vidéo grâce à ce Mastère Hybride"*

L'industrie des jeux vidéo a un grand potentiel. La demande croissante et les exigences des *Gamers* eux-mêmes ont conduit ce secteur à une course à la perfection de ses titres. Le haut niveau de qualité de chacune des créations est soutenu par une équipe de professionnels de la programmation possédant d'excellentes qualifications.

Ce Mastère Hybride en Programmation de Jeux Vidéo répond aux besoins actuels du marché, qui exige des professionnels de plus en plus spécialisés et très impliqués dans les créations. La créativité joue un rôle important, mais sans connaissances solides, il ne serait pas possible d'obtenir des jeux vidéo de haut niveau.

C'est pourquoi ce diplôme offre aux étudiants une connaissance exhaustive des fondamentaux de la programmation et du génie logiciel, approfondit la structure des données et des algorithmes, ainsi que l'enseignement de la programmation orientée aux objets et des spécifications des moteurs. Ce programme aborde également la programmation en temps réel afin d'offrir au professionnel du jeu vidéo un Mastère Hybride complet.

Afin d'atteindre l'objectif de progresser dans la carrière professionnelle de la Programmation de Jeux Vidéo, les étudiants disposeront d'un corps enseignant expert dans ce domaine, qui les guidera et les encadrera à tout moment. En outre, le contenu interactif avec des résumés vidéo, des études de cas et des lectures supplémentaires complétera le programme complet mis à disposition par TECH dans ce diplôme 100% en ligne avec des stages en entreprise.

Ainsi, les étudiants pourront compléter ce programme académique par un séjour intensif de trois semaines en face-à-face dans un studio de programmation de jeux vidéo de premier plan. Un environnement professionnel idéal, où vous pourrez voir in situ les méthodes de travail, les programmes et les techniques les plus sophistiqués pour la création de titres de haute qualité. Une opportunité unique que seule TECH, la plus grande université numérique au monde, peut vous offrir.

Ce **Mastère Hybride en Programmation de Jeux Vidéo** contient le programme académique le plus complet et le plus actuel du marché. Les principales caractéristiques sont les suivantes:

- Développement de plus de 100 études de cas de Programmation de Jeux Vidéo présentées par des professionnels de la programmation et des professeurs d'université ayant une grande expérience dans l'industrie du jeu vidéo
- Son contenu graphique, schématique et éminemment pratique, qui vise à fournir des informations scientifiques et d'assistance sur les disciplines médicales indispensables à la pratique professionnelle
- Le développement d'études de cas présentées par des experts en programmation et développement de jeux vidéo
- Les contenus graphiques, schématiques et éminemment pratiques avec lesquels ils sont conçus fournissent des informations scientifiques et sanitaires essentielles à la pratique professionnelle
- Les exercices pratiques où effectuer le processus d'auto-évaluation pour améliorer l'apprentissage
- Il met l'accent sur des méthodologies innovantes
- Cours théoriques, questions à l'expert, forums de discussion sur des sujets controversés et travail de réflexion individuel
- La possibilité d'accéder aux contenus depuis n'importe quel appareil fixe ou portable doté d'une connexion internet
- Tout cela sera complété par des cours théoriques, des questions à l'expert, des forums de discussion sur des sujets controversés et un travail de réflexion individuel
- Disponibilité des contenus à partir de tout appareil fixe ou portable doté d'une connexion internet
- Vous pourrez également effectuer un stage dans l'un des meilleurs studio de production de jeux vidéo

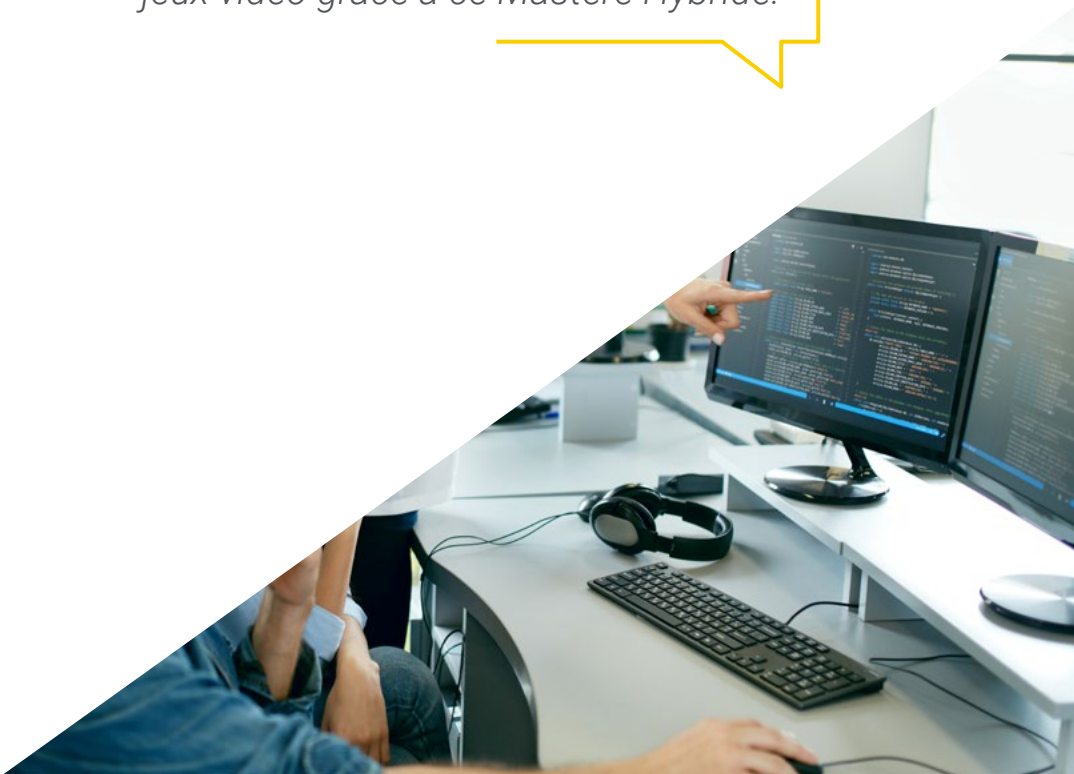
*Ce programme 100% en ligne vous donnera l'occasion de faire un stage dans un studio et de vous tester auprès des meilleurs programmeurs"*

*Ce Mastère Hybride vous permettra de devenir une référence dans le domaine de la programmation de jeux vidéo. Inscrivez-vous maintenant.*

Dans cette proposition de Mastère, de nature professionnalisante et de modalité d'apprentissage Hybride, le programme vise à mettre à jour les professionnels du jeu vidéo qui exercent leurs fonctions dans de grands studios créatifs, et qui exigent un haut niveau de qualification. Les contenus sont basés sur les dernières preuves scientifiques et orientés de manière didactique pour intégrer les connaissances théoriques dans la pratique de l'industrie du *gaming,* et les éléments théoriques-pratiques faciliteront la mise à jour des connaissances et permettront la prise de décision dans la programmation de jeux vidéo.

Grâce à son contenu multimédia développé avec les dernières technologies éducatives, il permettra au professionnel du jeu vidéo d'apprendre de manière située et contextuelle, c'est-à-dire dans un environnement simulé qui fournira un apprentissage immersif programmé pour s'entraîner dans des situations réelles. La conception de ce programme est axée sur l'apprentissage par les problèmes, grâce auquel vous devrez essayer de résoudre les différentes situations de pratique professionnelle qui se présenteront tout au long du programme. Pour ce faire, l'étudiant sera assisté d'un innovant système de vidéos interactives, créé par des experts reconnus.

*Développez des applications efficacement appliquées aux moteurs de jeux vidéo grâce à ce Mastère Hybride.*



02

# Pourquoi suivre ce Mastère Hybride?

La programmation requiert de solides connaissances théoriques, mais c'est sans aucun doute la pratique quotidienne qui parfait la maîtrise du professionnel dans le secteur des jeux vidéo. C'est pourquoi TECH a créé ce diplôme, qui combine parfaitement le programme en ligne le plus avancé dans des domaines tels que la conception d'algorithmes, les systèmes intelligents et la programmation en temps réel avec un séjour pratique dans un studio de programmation de premier plan. De cette manière, les étudiants obtiendront une vision beaucoup plus complète de la Programmation de Jeux Vidéo, des programmes utilisés et des techniques les plus récentes. Le tout, guidé à tout moment par les meilleurs experts du domaine.

*TECH vous donne l'occasion de suivre une Formation Pratique efficace et intensive dans le domaine de la Programmation de Jeux Vidéo"*

### 1. Actualisation des technologies les plus récentes

Ces dernières années, la technologie a révolutionné le domaine de la Programmation de Jeux Vidéo, favorisant la création de titres d'une plus grande qualité et d'un plus grand réalisme. Pour cette raison, et afin de rapprocher les étudiants de cette technologie, TECH a créé ce Mastère Hybride, où le professionnel apprendra à connaître les moteurs de jeux vidéo, les défis technologiques actuels et les logiciels pour la création d'ontologies. De cette manière, vous obtiendrez une mise à jour des dernières technologies disponibles.

### 2. Exploiter l'expertise des meilleurs spécialistes

Dans ce parcours académique, les étudiants seront toujours guidés par les meilleurs spécialistes de la Programmation de Jeux Vidéo. Ainsi, le diplômé disposera d'une excellente équipe d'enseignants possédant une vaste expérience dans le secteur, qui l'accompagnera pendant la phase théorique, tandis que, pendant le séjour pratique, il sera entouré de véritables experts, qui font partie de l'équipe du studio où il effectuera son stage.

### 3. Accéder à des environnements de premier ordre

TECH procède à une sélection rigoureuse de tous les studios et entreprises où se déroulent les stages. Les étudiants ont ainsi la garantie d'accéder à un environnement professionnel de haut niveau dans le domaine de la Programmation de Jeux Vidéo. Ils pourront ainsi se rendre compte par eux-mêmes de ce qu'est le travail quotidien d'un programmeur spécialisé, ainsi que des techniques et méthodes utilisées pour réaliser des titres de qualité.



### 4. Combiner la meilleure théorie avec la pratique la plus avancée

Avec ce diplôme, TECH s'adapte au travail quotidien des programmeurs et adopte donc une approche théorique-pratique, loin des longues heures d'étude, pour se concentrer sur les concepts clés. Cette institution académique offre ainsi un modèle d'apprentissage innovant qui permet aux étudiants d'acquérir les connaissances nécessaires pour prendre les devants dans la Programmation de Jeux Vidéo de haute qualité.

### 5. Élargir les frontières de la connaissance

TECH offre la possibilité de réaliser cette Formation Pratique dans des studios d'envergure internationale. Le spécialiste pourra ainsi élargir ses frontières et se rapprocher des meilleurs professionnels exerçant dans des studios de premier ordre sur différents continents. Une opportunité unique que seul TECH, la plus grande université numérique au monde, pouvait offrir.

*Vous serez en immersion totale dans le centre de votre choix”*

03

# Objectifs

La conception du programme de ce Mastère Hybride permettra aux étudiants d'acquérir les compétences nécessaires pour progresser dans la programmation, domaine hautement compétitif qui nécessite du personnel qualifié. Par conséquent, l'objectif principal de ce cours est de s'assurer que les étudiants deviennent d'excellents développeurs de jeux vidéo en utilisant tous les outils que ce cours fournit. Pour ce faire, ce diplôme propose un contenu multimédia actualisé avec des lectures supplémentaires et un système d'apprentissage, le *Relearning,* basé sur la réitération du contenu, qui facilitera l'ancrage des concepts.

*Vous voulez créer un jeu multijoueur? Apprenez à transformer votre idée en un véritable langage de programmation pour ce type de jeux vidéo"*

## Objectif général

- S'assurer que les étudiants sont familiarisés avec les différents langages et méthodes de programmation appliqués aux jeux vidéo. À cette fin, le processus de production d'un titre et son intégration dans les différentes étapes seront étudiés en profondeur. De même, le professionnel du jeu vidéo apprendra les bases de la conception de jeux vidéo et les principales connaissances théoriques. Par ailleurs, à l'issue de ce cours, les étudiants seront capables de comprendre le rôle de la programmation et de développer des jeux vidéo web et multijoueurs

*Ce programme vous permettra de progresser professionnellement. Vous serez en mesure de maîtriser toutes les structures de données et les algorithmes grâce à ce Mastère Hybride"*

## Objectifs spécifiques

### Module 1. Principes fondamentaux de la programmation

- Comprendre la structure de base d'un ordinateur, les logiciels et les langages de programmation à usage général
- Analyser les éléments essentiels d'un programme informatique, tels que les différents types de données, les opérateurs, les expressions, les instructions, les entrées/sorties et les instructions de contrôle
- interpréter des algorithmes, qui constituent la base nécessaire au développement de programmes informatiques

### Module 2. Structure des données et algorithmes

- Apprendre les principales stratégies de conception d'algorithmes, ainsi que les différentes méthodes et mesures de calcul d'algorithmes
- Distinguer le fonctionnement des algorithmes, leur stratégie et des exemples de leur utilisation dans les principaux problèmes connus
- Comprendre la technique de *Backtracking et ses utilisations principales*

### Module 3. Programmation orientée objet

- Apprendre les différents modèles de conception pour les problèmes orientés objet
- Comprendre l'importance de la documentation et des tests dans le développement de logiciels
- Gérer l'utilisation des threads et de la synchronisation, ainsi que la résolution des problèmes courants de la programmation concurrente

### Module 4. Consoles et appareils de jeux vidéo

- Connaître le fonctionnement de base des principaux périphériques d'entrée et de sortie
- Comprendre les principales implications de conception des différentes plateformes
- Étudier la structure, l'organisation, le fonctionnement et l'interconnexion des dispositifs et des systèmes
- Comprendre le rôle du système d'exploitation et des kits de développement pour les appareils mobiles et les plateformes de jeux vidéo

### Module 5. Ingénierie logicielle

- Distinguer les bases de l'ingénierie logicielle, ainsi que le processus logiciel et les différents modèles de développement, y compris les technologies agiles
- Reconnaître l'ingénierie des exigences, leur développement, leur élaboration, leur négociation et leur validation afin de comprendre les principales normes relatives à la qualité des logiciels et à la gestion de projet

### Module 6. Moteurs de jeux vidéo

- Découvrez le fonctionnement et l'architecture d'un moteur de jeu vidéo
- Comprendre les caractéristiques de base des moteurs de jeu existants
- Applications de programmation correctement et efficacement appliquées aux moteurs de jeux vidéo
- Choisir le paradigme et les langages de programmation les plus appropriés pour la programmation d'applications appliquées aux moteurs de jeux vidéo

### Module 7. Systèmes intelligents

- Établir les concepts liés à la théorie et à l'architecture des agents et à leur processus de raisonnement
- Assimiler la théorie et la pratique des concepts d'information et de connaissance, ainsi que les différentes manières de représenter la connaissance
- Comprendre le fonctionnement des raisonneurs sémantiques, des systèmes à base de connaissances et des systèmes experts

### Module 8. Programmation en temps réel

- Analyser les principales caractéristiques d'un langage de programmation en temps réel qui le différencient d'un langage de programmation traditionnel
- Comprendre les concepts de base des systèmes informatiques
- Acquérir la capacité d'appliquer les principaux principes fondamentaux et techniques de la programmation en temps réel

### Module 9. Design et développement de jeux web

- Être capable de concevoir des jeux et des applications web interactives avec la documentation correspondante
- Évaluer les principales caractéristiques des jeux et des applications web interactives afin de communiquer de manière professionnelle et correcte

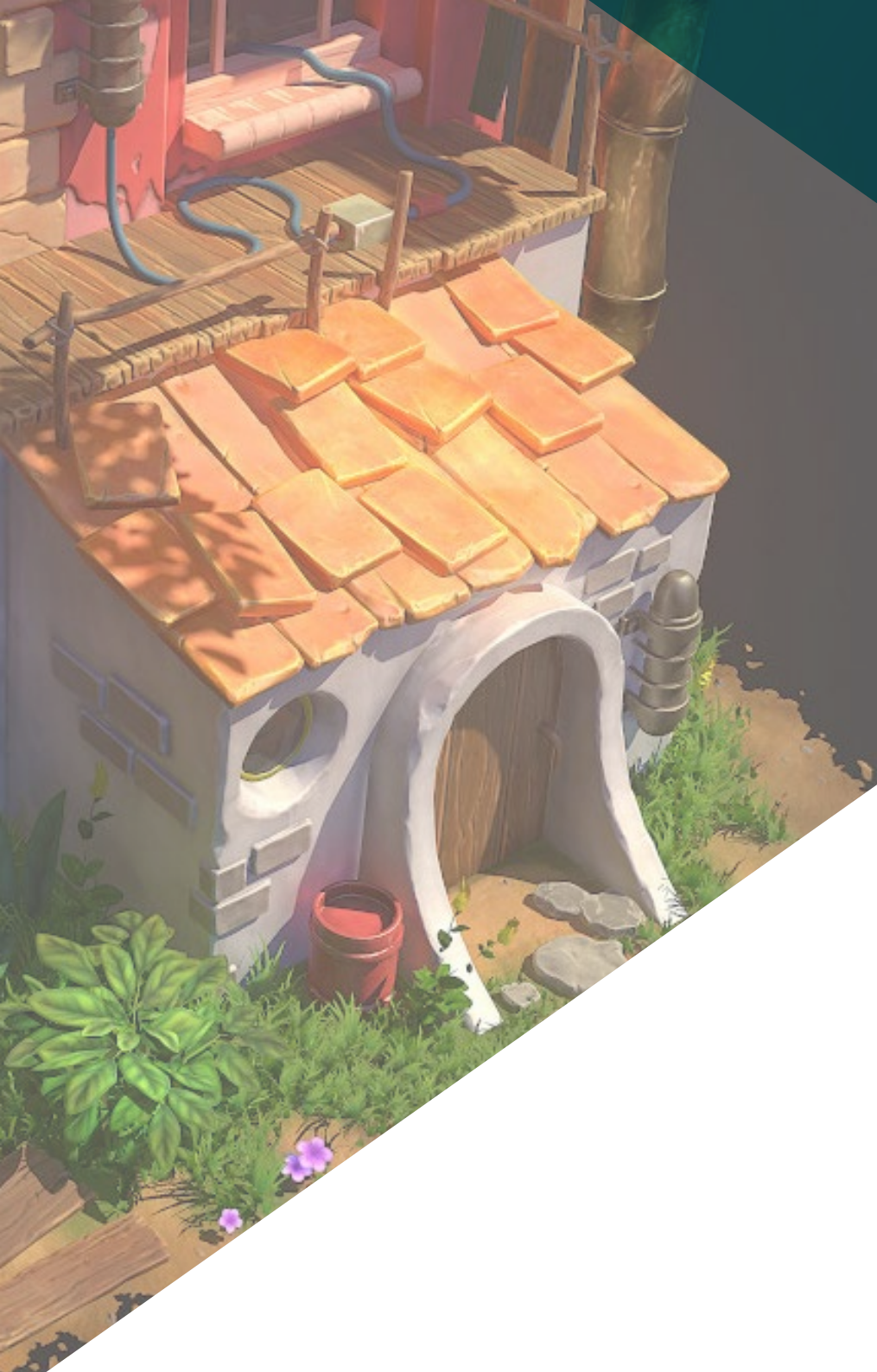
### Module 10. Réseaux et systèmes multi-joueurs

- Décrire l'architecture du protocole de contrôle de transmission/protocole Internet (TCP/IP) et le fonctionnement de base des réseaux sans fil
- Analyser la sécurité appliquée aux jeux vidéo
- Acquérir la capacité de développer des jeux en ligne multi-joueurs

04

# Compétences

Les professionnels du jeu vidéo qui suivent ce Mastère Hybride acquièrent une série de compétences qui leur permettent d'intégrer tout type de studio de développement de jeux vidéo. Ainsi, à l'issue de cette formation, les étudiants auront les compétences nécessaires pour programmer dans les différents langages utilisés dans l'industrie du *gaming* et acquerront les compétences essentielles pour mener un projet du début à la fin sur différentes plateformes et moteurs de jeux vidéo.

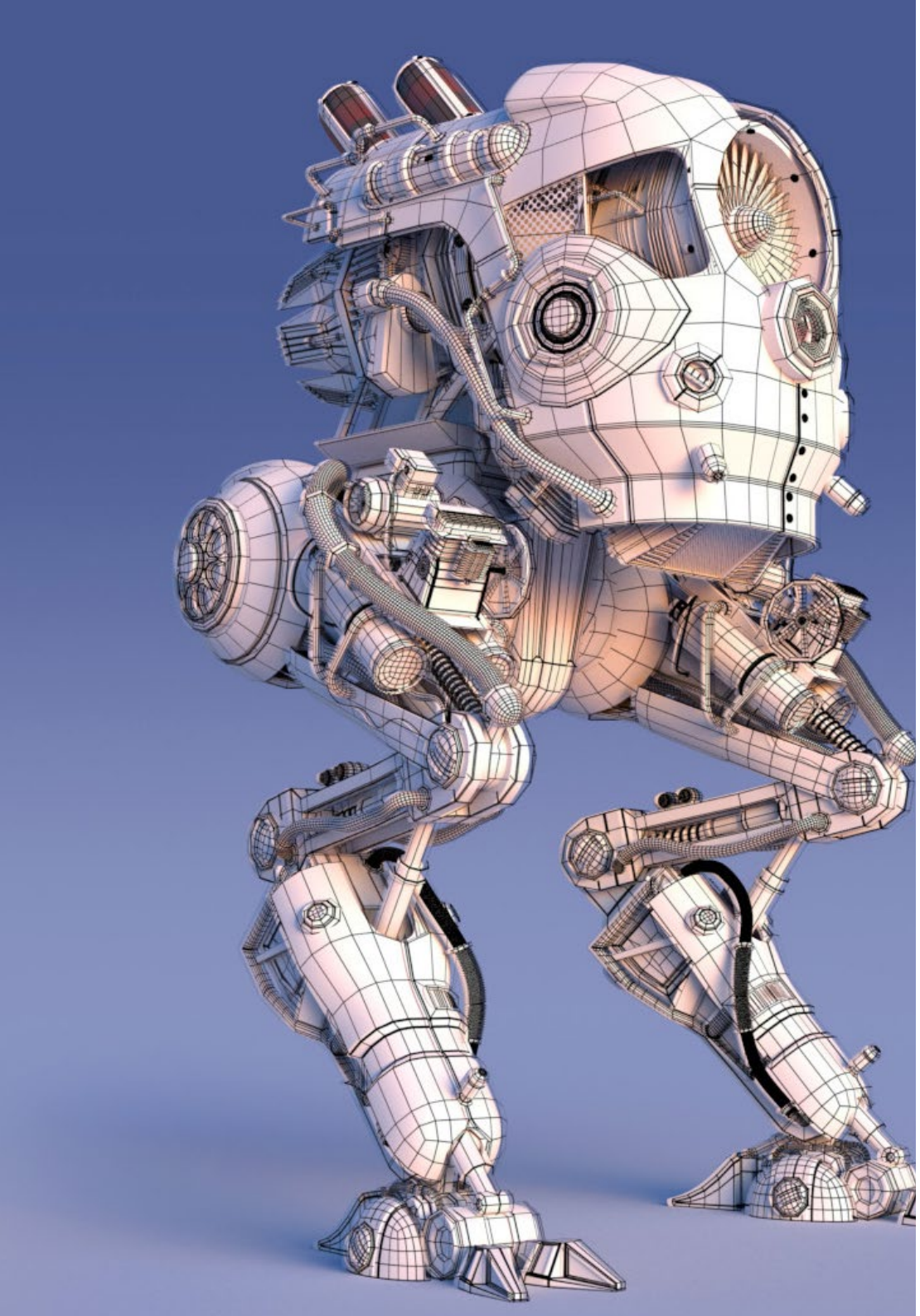


*Acquérir les compétences nécessaires pour programmer des jeux vidéo pour Xbox One, PS4 et Wii U-Switch"*

## Compétences générales

- Concevoir toutes les phases d'un jeu vidéo, de l'idée initiale au lancement final
- Se spécialiser en tant que programmeur de jeux vidéo
- Se plonger dans toutes les parties du développement, de l'architecture initiale à la programmation du personnage du joueur, en passant par tous les éléments impliqués dans le processus de jeu
- Obtenir une vision globale du projet, en étant capable d'apporter des solutions aux différents problèmes et défis qui se posent dans la conception d'un jeu vidéo

*Avant de programmer, il faut connaître l'expérience du joueur et analyser la jouabilité du jeu vidéo. Avec ce Mastère Hybride, vous apprendrez à atteindre la réussite"*



## Compétences spécifiques

- Connaître les logiciels nécessaires pour être un développeur professionnel de jeux vidéo
- Comprendre l'expérience du joueur et savoir analyser la jouabilité du jeu vidéo
- Comprendre toutes les procédures théoriques et pratiques impliquées dans le processus de Programmation de Jeux Vidéo
- Maîtriser les langages de programmation les plus utiles dans le monde du jeu vidéo
- Intégrer la programmation apprise aux différents types de consoles et de plateformes
- Programmer des jeux vidéo web et multijoueurs
- Assimiler le concept de moteur de jeu vidéo afin de pouvoir programmer correctement
- Appliquer les connaissances du génie logiciel à la programmation de jeux vidéo

05

# Plan d’étude

Ce Mastère Hybride en Programmation de Jeux Vidéo offre aux étudiants un contenu complet et varié en développement de jeux vidéo structuré en 10 modules spécialisés. Les enseignants qui composent ce diplôme pourront approfondir les fondamentaux de la programmation tout en acquérant des connaissances actualisées et orientées vers la pratique. Les étudiants reçoivent ainsi un enseignement de qualité, avec des contenus innovants au format multimédia, téléchargeables et consultables à tout moment. La formation théorique 100% en ligne vous donnera la liberté de répartir la charge d'enseignement en fonction de vos besoins.

*Un plan d'étude conçu pour vous offrir le contenu le plus avancé et le plus actuel sur la Programmation de Jeux Vidéo"*

## **Module 1.** Principes fondamentaux de la programmation

1.1. Introduction à la programmation
- 1.1.1. Structure de base d'un ordinateur
- 1.1.2. Software
- 1.1.3. Langages de programmation
- 1.1.4. Cycle de vie d'une application logicielle

1.2. Conception d'algorithmes
- 1.2.1. Résolution de problèmes
- 1.2.2. Techniques descriptives
- 1.2.3. Éléments et structure d'un algorithme

1.3. Éléments d'un programme
- 1.3.1. Origine et caractéristiques du langage C++
- 1.3.2. L'environnement de développement
- 1.3.3. Concept du programme
- 1.3.4. Types de données fondamentales
- 1.3.5. Opérateurs
- 1.3.6. Expressions
- 1.3.7. Phrases
- 1.3.8. Entrée et sortie de données

1.4. Déclarations de contrôle
- 1.4.1. Phrases
- 1.4.2. Branches
- 1.4.3. Boucles

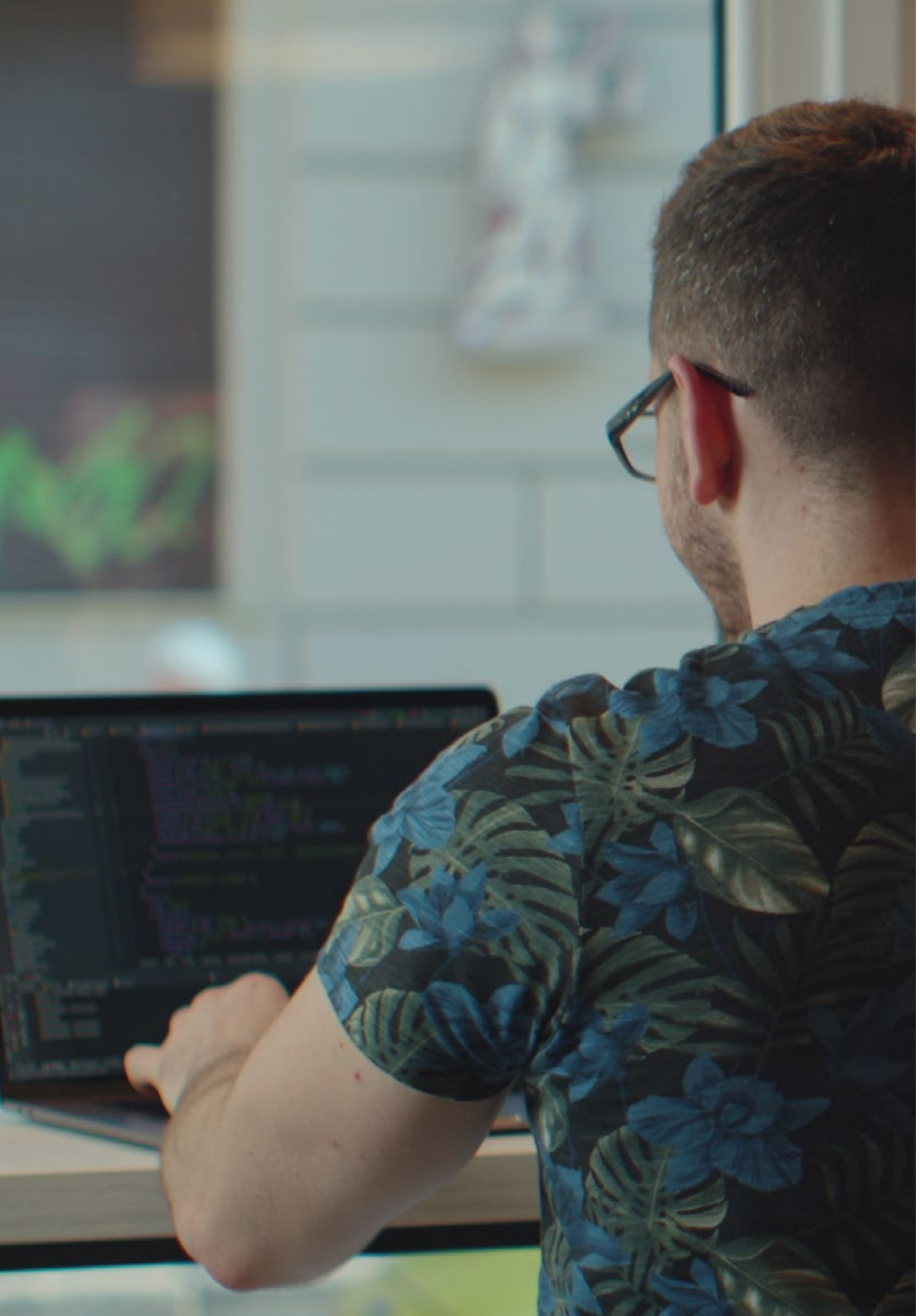
1.5. Abstraction et modularité: fonctions
- 1.5.1. Conception modulaire
- 1.5.2. Concept de fonction et d'utilité
- 1.5.3. Définition d'une fonction
- 1.5.4. Flux d'exécution dans un appel de fonction
- 1.5.5. Prototypes d'une fonction
- 1.5.6. Retour des résultats
- 1.5.7. Appel d'une fonction: paramètres
- 1.5.8. Passage de paramètres par référence et par valeur
- 1.5.9. Identifiant du champ d'application

1.6. Structures de données statiques
- 1.6.1. *Arrays*
- 1.6.2. Les matrices. Polyèdres
- 1.6.3. Recherche et tri
- 1.6.4. Cordes Fonctions E/S pour les chaînes de caractères
- 1.6.5. Structures Jonctions
- 1.6.6. Nouveaux types de données

1.7. Structures de données dynamiques: pointeurs
- 1.7.1. Concept. Définition du pointeur
- 1.7.2. Opérateurs et opérations avec des pointeurs
- 1.7.3. *Tableaux* de pointeurs
- 1.7.4. Pointeurs et *Tableaux*
- 1.7.5. Pointeurs vers des chaînes de caractères
- 1.7.6. Pointeurs vers des structures
- 1.7.7. Indications multiples
- 1.7.8. Pointeurs vers les fonctions
- 1.7.9. Passage de fonctions, de structures et de *tableaux* comme paramètres de fonction

1.8. Fichiers
- 1.8.1. Concepts de base
- 1.8.2. Opérations avec des fichiers
- 1.8.3. Types de fichiers
- 1.8.4. Organisation des fichiers
- 1.8.5. Introduction aux fichiers C++
- 1.8.6. Traitement des fichiers

1.9. Récursion
- 1.9.1. Définition de la récursion
- 1.9.2. Types de récursions
- 1.9.3. Avantages et inconvénients
- 1.9.4. Considérations
- 1.9.5. Conversion récursive-interative
- 1.9.6. La pile de récursion





1.10. Tests et documentation
- 1.10.1. Test du programme
- 1.10.2. Test boîte blanche
- 1.10.3. Tests en boîte noire
- 1.10.4. Outils de test
- 1.10.5. Documentation du logiciel

## **Module 2.** Structure des données et algorithmes

2.1. Introduction aux stratégies de conception d'algorithmes
- 2.1.1. Récursion
- 2.1.2. Diviser pour mieux régner
- 2.1.3. Autres stratégies

2.2. Efficacité et analyse des algorithmes
- 2.2.1. Mesures d'efficacité
- 2.2.2. Taille de l'entrée de mesure
- 2.2.3. Mesure du temps d'exécution
- 2.2.4. Pire, meilleur et moyen cas
- 2.2.5. Notation asymptotique
- 2.2.6. Critères d'analyse mathématique des algorithmes non récursifs
- 2.2.7. Analyse mathématique des algorithmes récursifs
- 2.2.8. Analyse empirique des algorithmes

2.3. Algorithmes de tri
- 2.3.1. Concept de tri
- 2.3.2. Triage des bulles
- 2.3.3. Tri par sélection
- 2.3.4. Triage par insertion
- 2.3.5. Tri par mélange (Merge_Sort)
- 2.3.6. Tri rapide (Quick_Sort)

2.4. Algorithmes avec arbres
- 2.4.1. Concept d'arbre
- 2.4.2. Arbres binaires
- 2.4.3. Allées d'arbres
- 2.4.4. Représentation des expressions
- 2.4.5. Arbres binaires ordonnés
- 2.4.6. Arbres binaires équilibrés

2.5. Algorithmes avec *Heaps*
- 2.5.1. Les *heaps*
- 2.5.2. L'algorithme *Heapsort*
- 2.5.3. Files d'attente prioritaires

2.6. Algorithmes graphiques
- 2.6.1. Représentation
- 2.6.2. Voyage en largeur
- 2.6.3. Profondeur de déplacement
- 2.6.4. Disposition topologique

2.7. Algorithmes *Greedy*
- 2.7.1. La stratégie *Greedy*
- 2.7.2. Éléments de la stratégie *Greedy*
- 2.7.3. Change de devises
- 2.7.4. Le problème du voyageur
- 2.7.5. Problème de sac à dos

2.8. Recherche de chemins minimaux
- 2.8.1. Le problème du chemin minimal
- 2.8.2. Arcs et cycles négatifs
- 2.8.3. Algorithme de Dijkstra

2.9. Algorithmes *greedy* sur les graphes
- 2.9.1. L'arbre à chevauchement minimal
- 2.9.2. L'algorithme de Prim
- 2.9.3. L'algorithme de Kruskal
- 2.9.4. Analyse de la complexité

2.10. *Backtracking*
- 2.10.1. Le *Backtracking*
- 2.10.2. Techniques alternatives

## **Module 3.** Programmation orientée aux objets

3.1. Introduction à la programmation orientée objet
- 3.1.1. Introduction à la programmation orientée objet
- 3.1.2. Conception de la classe
- 3.1.3. Introduction à UML pour la modélisation des problèmes

3.2. Relations entre les classes
- 3.2.1. Abstraction et héritage
- 3.2.2. Concepts d'héritage avancés
- 3.2.3. Polymorphisme
- 3.2.4. Composition et agrégation

3.3. Introduction aux patrons de conception pour les problèmes orientés objet
- 3.3.1. Que sont les patrons de conception?
- 3.3.2. Modèle *Factory*
- 3.3.3. Modèle *Singleton*
- 3.3.4. Modèle *Observer*
- 3.3.5. Modèle *Composite*

3.4. Exceptions
- 3.4.1. Quelles sont les exceptions?
- 3.4.2. Capture et traitement des exceptions
- 3.4.3. Lancer d'exceptions
- 3.4.4. Création d'exceptions

3.5. Interfaces utilisateur
- 3.5.1. Introduction à Qt
- 3.5.2. Positionnement
- 3.5.3. Que sont les événements?
- 3.5.4. Événements: définition et saisie
- 3.5.5. Développement d'interfaces utilisateurs

3.6. Introduction à la programmation concurrente
- 3.6.1. Introduction à la programmation concurrente
- 3.6.2. Le concept de processus et de fil conducteur
- 3.6.3. Interaction entre processus ou threads
- 3.6.4. Threads en C++
- 3.6.5. Avantages et inconvénients de la programmation concurrente

3.7. Gestion et synchronisation des threads
- 3.7.1. Cycle de vie du fil
- 3.7.2. La classe *Thread*
- 3.7.3. Programmation du fil
- 3.7.4. Groupes de fils
- 3.7.5. Fils démoniaques
- 3.7.6. Synchronisation
- 3.7.7. Mécanismes de verrouillage
- 3.7.8. Mécanismes de communication
- 3.7.9. Moniteurs

3.8. Problèmes courants de la programmation concurrente
- 3.8.1. Le problème du producteur-consommateur
- 3.8.2. Le problème des lecteurs et des écrivains
- 3.8.3. Le problème du dîner des philosophes

3.9. Documentation et test des logiciels
- 3.9.1. Pourquoi est-il important de documenter les logiciels?
- 3.9.2. Documentation sur la conception
- 3.9.3. Utilisation des outils de documentation

3.10. Tests de logiciels
- 3.10.1. Introduction aux tests logiciels
- 3.10.2. Types de tests
- 3.10.3. Tests unitaires
- 3.10.4. Test d'intégration
- 3.10.5. Test de validation
- 3.10.6. Test du système

## **Module 4.** Consoles et appareils de jeux vidéo

4.1. Histoire de la programmation dans jeux vidéo
- 4.1.1. Période Atari (1977-1985)
- 4.1.2. Période NES et SNES (1985-1995)
- 4.1.3. Période PlayStation/PlayStation 2 (1995-2005)
- 4.1.4. Période Xbox 360, PS3 et Wii (2005-2013)
- 4.1.5. Période Xbox One, PS4 et Wii U- Switch (2013-présent)
- 4.1.6. Le futur

4.2. Histoire de la programmation dans jeux vidéo
- 4.2.1. Introduction
- 4.2.2. Le contexte social
- 4.2.3. Diagramme structurel
- 4.2.4. Futur

4.3. Adaptation aux temps modernes
- 4.3.1. Jeux basés sur le mouvement
- 4.3.2. Réalité virtuelle
- 4.3.3. Réalité augmentée
- 4.3.4. Réalité mixte

4.4. Unity: *Scripting* I et exemples
- 4.4.1. Qu'est-ce qu'un *script*?
- 4.4.2. Notre premier *script*
- 4.4.3. Ajout d'un *script*
- 4.4.4. Ouverture d'un *script*
- 4.4.5. *MonoBehaviour*
- 4.4.6. *Debugging*

4.5. Unity: *Scripting* II et exemples
- 4.5.1. Entrée du clavier et de la souris
- 4.5.2. *Raycast*
- 4.5.3. Installation
- 4.5.4. Variables
- 4.5.5. Variables publiques et sérialisées

4.6. Unity: *Scripting* III et exemples
- 4.6.1. Obtention des composants
- 4.6.2. Modifier les composants
- 4.6.3. Essais
- 4.6.4. Objets multiples
- 4.6.5. *Colliders et Triggers*
- 4.6.6. Quaternions

4.7. Périphériques
- 4.7.1. Évolution et classification
- 4.7.2. Périphériques et interfaces
- 4.7.3. Périphériques actuels
- 4.7.4. Futur proche

4.8. Jeux vidéo: perspectives d'avenir
- 4.8.1. Jeux en ligne
- 4.8.2. Absence de contrôleurs
- 4.8.3. Réalité immersive
- 4.8.4. Autres alternatives

4.9. Architecture
- 4.9.1. Exigences particulières pour les jeux vidéo
- 4.9.2. Évolution de l'architecture
- 4.9.3. Architecture actuelle
- 4.9.4. Différences entre les architectures

4.10. Les kits de développement et leur évolution
- 4.10.1. Introduction
- 4.10.2. Kits de développement de troisième génération
- 4.10.3. Kits de développement de quatrième génération
- 4.10.4. Kits de développement de cinquième génération
- 4.10.5. Kits de développement de sixième génération

## **Module 5.** Ingénierie logicielle

5.1. Introduction au génie logiciel et à la modélisation
- 5.1.1. La nature des logiciels
- 5.1.2. La nature unique des *WebApps*
- 5.1.3. Ingénierie logicielle
- 5.1.4. Le processus logiciel
- 5.1.5. La pratique du génie logiciel
- 5.1.6. Mythes sur les logiciels
- 5.1.7. Comment tout commence?
- 5.1.8. Concepts orientés objet
- 5.1.9. Introduction à UML

5.2. Le processus logiciel
- 5.2.1. Un modèle général de processus
- 5.2.2. Modèles de processus prescriptifs
- 5.2.3. Modèles de processus spécialisés
- 5.2.4. Le processus unifié
- 5.2.5. Modèles de processus personnels et d'équipe
- 5.2.6. Qu'est-ce que l'agilité?
- 5.2.7. Qu'est-ce qu'un processus agile?
- 5.2.8. *Scrum*
- 5.2.9. Boîte à outils du processus Agile

5.3. Principes guidant la pratique du génie logiciel
- 5.3.1. Principes guidant le processus
- 5.3.2. Principes guidant la pratique
- 5.3.3. Principes de la communication
- 5.3.4. Principes de planification
- 5.3.5. Principes de modélisation
- 5.3.6. Principes de construction
- 5.3.7. Principes de déploiement

5.4. Comprendre les besoins
- 5.4.1. Ingénierie des exigences
- 5.4.2. Établir la base
- 5.4.3. Détermination des besoins
- 5.4.4. Développer des cas d'utilisation
- 5.4.5. Élaboration du modèle d'exigences
- 5.4.6. Négociation des exigences
- 5.4.7. Validation des exigences

5.5. Modélisation des exigences: scénarios, classes d'information et d'analyse
- 5.5.1. Analyse des besoins
- 5.5.2. Modélisation basée sur des scénarios
- 5.5.3. Modèles UML fournissant le cas d'utilisation
- 5.5.4. Concepts de modélisation des données
- 5.5.5. Modélisation basée sur les classes
- 5.5.6. Diagrammes de classes

5.6. Modélisation des exigences: flux, comportement et modèles
- 5.6.1. Stratégies de modélisation des exigences
- 5.6.2. Modélisation orientée flux
- 5.6.3. Diagrammes d'état
- 5.6.4. Création d'un modèle comportemental
- 5.6.5. Diagrammes de séquence
- 5.6.6. Diagrammes de communication
- 5.6.7. Modèles pour la modélisation des exigences

5.7. Concepts de design
- 5.7.1. La conception dans le contexte du génie logiciel
- 5.7.2. Le processus de conception
- 5.7.3. Concepts de design
- 5.7.4. Concepts de conception orientée objet
- 5.7.5. Le modèle de conception

5.8. Architecture de conception
- 5.8.1. Architecture logicielle
- 5.8.2. Genres architecturaux
- 5.8.3. Styles architecturaux
- 5.8.4. Conception architecturale
- 5.8.5. Évolution des conceptions alternatives de l'architecture
- 5.8.6. Cartographie de l'architecture à l'aide du flux de données

5.9. Conception au niveau des composants et des modèles
- 5.9.1. Qu'est-ce qu'un composant?
- 5.9.2. Conception de composants basée sur les classes
- 5.9.3. Réalisation de la conception au niveau des composants
- 5.9.4. Conception traditionnelle des composants
- 5.9.5. Développement basé sur les composants
- 5.9.6. Modèles de conception
- 5.9.7. Conception de logiciels basée sur des modèles
- 5.9.8. Modèles architecturaux
- 5.9.9. Patrons de conception au niveau des composants
- 5.9.10. Modèles de conception d'interface utilisateur

5.10. Qualité des logiciels et gestion de projet
- 5.10.1. Qualité
- 5.10.2. Qualité des logiciels
- 5.10.3. Le dilemme de la qualité des logiciels
- 5.10.4. Atteindre la qualité des logiciels
- 5.10.5. Assurance qualité des logiciels
- 5.10.6. Le spectre de la gestion
- 5.10.7. Personnel
- 5.10.8. Le produit
- 5.10.9. Le processus
- 5.10.10. Le projet
- 5.10.11. Principes et pratiques

## **Module 6.** Moteurs de jeux vidéo

6.1. Jeux vidéo et TIC
- 6.1.1. Introduction
- 6.1.2. Opportunités
- 6.1.3. Défis
- 6.1.4. Conclusions

6.2. Histoire des moteurs de jeux
- 6.2.1. Introduction
- 6.2.2. L'ère Atari
- 6.2.3. L'ère des années 80
- 6.2.4. Les premiers moteurs. L'ère des années 90
- 6.2.5. Moteurs actuels

6.3. Moteurs de jeux vidéo
- 6.3.1. Types de moteurs
- 6.3.2. Parties d'un moteur de jeu vidéo
- 6.3.3. Moteurs actuels
- 6.3.4. Sélection d'un moteur pour notre projet

6.4. *Motor Game Maker*
- 6.4.1. Introduction
- 6.4.2. Conception de scénarios
- 6.4.3. *Sprites* et animations
- 6.4.4. Collisions
- 6.4.5. *Scripting* dans GML

6.5. Motor Unreal Engine 4: Introduction
- 6.5.1. Qu'est-ce qu'Unreal Engine 4? Quelle est sa philosophie?
- 6.5.2. Matériaux
- 6.5.3. UI
- 6.5.4. Animations
- 6.5.5. Systèmes de particules
- 6.5.6. Intelligence artificielle
- 6.5.7. FPS

6.6. Motor Unreal Engine 4: Visual *Scripting*
- 6.6.1. Philosophie des *Blueprints* et le *Visual Scripting*
- 6.6.2. *Debugging*
- 6.6.3. Types de variables
- 6.6.4. Contrôle des flux basiques

6.7. Motor Unity 5
- 6.7.1. Programmation en C# et Visual Studio
- 6.7.2. Création de *Prefabs*
- 6.7.3. Utilisation de *Gizmos* pour contrôler le jeu vidéo
- 6.7.4. Moteur adaptatif: 2D et 3D

6.8. Moteur Godot
- 6.8.1. La philosophie de conception de Godot
- 6.8.2. Conception et composition orientées objet
- 6.8.3. Paquet tout-en-un
- 6.8.4. Logiciels libres et communautaires

6.9. Motor RPG Maker
- 6.9.1. Philosophie de RPG Maker
- 6.9.2. En prenant comme référence
- 6.9.3. Créer un jeu avec de la personnalité
- 6.9.4. Des jeux commerciaux réussis

6.10. Motor Source 2
- 6.10.1. Philosophie de la Source 2
- 6.10.2. Source et Source 2: Évolution
- 6.10.3. Utilisation de la communauté: contenus audiovisuels et jeux vidéo
- 6.10.4. L'avenir du moteur Source 2
- 6.10.5. *Mods* et jeux réussis

## **Module 7.** Systèmes intelligents

7.1. Théorie des agents
- 7.1.1. Histoire du concept
- 7.1.2. Définition d'agent
- 7.1.3. Les agents en Intelligence Artificielle
- 7.1.4. Les agents en ingénierie logicielle

7.2. Architectures des agents
- 7.2.1. Le processus de raisonnement d'un agent
- 7.2.2. Agents réactifs
- 7.2.3. Agents déductifs
- 7.2.4. Agents hybrides
- 7.2.5. Comparaison

7.3. Information et connaissance
- 7.3.1. Distinction entre données, informations et connaissances
- 7.3.2. Évaluation de la qualité des données
- 7.3.3. Méthode de capture des données
- 7.3.4. Méthodes d'acquisition des informations
- 7.3.5. Méthodes d'acquisition des connaissances

7.4. Représentation des connaissances
- 7.4.1. L'importance de la représentation de la connaissance
- 7.4.2. Définition de la représentation des connaissances à travers leurs rôles
- 7.4.3. Caractéristiques de la représentation de la connaissance

7.5. Ontologies
- 7.5.1. Introduction aux Métadonnées
- 7.5.2. Concept philosophique d'ontologie
- 7.5.3. Concept informatique d'ontologie
- 7.5.4. Ontologies de domaine et ontologies de niveau supérieur
- 7.5.5. Comment construire une ontologie?

7.6. Langages ontologiques et logiciels pour la création d'ontologies
- 7.6.1. Triplés RDF, Turtle et N3
- 7.6.2. RDF Schema
- 7.6.3. OWL
- 7.6.4. SPARQL
- 7.6.5. Introduction aux différents outils de création d'ontologies
- 7.6.6. Installation et utilisation de Protégé

7.7. Le web sémantique
- 7.7.1. L'état actuel et futur du web sémantique
- 7.7.2. Applications du web sémantique

7.8. Autres modèles de représentation des connaissances
- 7.8.1. Vocabulaire
- 7.8.2. Vision globale
- 7.8.3. Taxonomie
- 7.8.4. Thésaurus
- 7.8.5. Folksonomies
- 7.8.6. Comparaison
- 7.8.7. Cartes mentales

7.9. Évaluation et intégration des représentations des connaissances
- 7.9.1. Logique d'ordre zéro
- 7.9.2. Logique de premier ordre
- 7.9.3. Logique descriptive
- 7.9.4. Relations entre les différents types de logique
- 7.9.5. Prolog: programmation basée sur la logique du premier ordre

7.10. Raisonneurs sémantiques, systèmes à base de connaissances et systèmes experts
- 7.10.1. Concept de raisonneur
- 7.10.2. Applications d'un raisonneur
- 7.10.3. Systèmes basés sur la connaissance
- 7.10.4. MYCIN, histoire des systèmes experts
- 7.10.5. Éléments et Architecture des Systèmes Experts
- 7.10.6. Création de Systèmes Experts

## **Module 8.** Programmation en temps réel

8.1. Concepts de base de la programmation concurrente
- 8.1.1. Concepts fondamentaux
- 8.1.2. Concurrence
- 8.1.3. Avantages de la concurrence
- 8.1.4. Concurrence et matériel

8.2. Structures de base du support de la concurrence en Java
- 8.2.1. Concurrence en Java
- 8.2.2. Créer des *Threads*
- 8.2.3. Méthodes
- 8.2.4. Synchronisation

8.3. *Threads*, cycle de vie, priorités, interruptions, états, exécuteurs
- 8.3.1. *Threads*
- 8.3.2. Cycle de vie
- 8.3.3. Priorités
- 8.3.4. Interruptions
- 8.3.5. États
- 8.3.6. Exécuteurs testamentaires

8.4. Exclusion mutuelle
- 8.4.1. Qu'est-ce que l'exclusion mutuelle?
- 8.4.2. L'algorithme de Dekker
- 8.4.3. L'algorithme de Peterson
- 8.4.4. Exclusion mutuelle en Java

8.5. Dépendances à l'égard des États
- 8.5.1. Injection de dépendances
- 8.5.2. Mise en œuvre du modèle en Java
- 8.5.3. Façons d'injecter des dépendances
- 8.5.4. Exemple

8.6. Modèles de conception
- 8.6.1. Introduction
- 8.6.2. Modèles de création
- 8.6.3. Modèles de structure
- 8.6.4. Modèles de comportement

8.7. Utilisation des librairies Java
- 8.7.1. Que sont les bibliothèques Java?
- 8.7.2. Mockito-all, Mockito-core
- 8.7.3. Goyave
- 8.7.4. Commons-io
- 8.7.5. Commons-lang, Commons-lang3

8.8. Programmation de *Shaders*
- 8.8.1. Pipeline et trame 3D
- 8.8.2. Vertex Shading
- 8.8.3. Pixel Shading: illumination I
- 8.8.4. Pixel Shading: illumination II
- 8.8.5. Post- Effect

8.9. Programmation en temps réel
- 8.9.1. Introduction
- 8.9.2. Traitement des interruptions
- 8.9.3. Synchronisation et communication interprocessus
- 8.9.4. Systèmes d'ordonnancement en temps réel

8.10. Planification en temps réel
- 8.10.1. Concepts
- 8.10.2. Modèle de référence des systèmes en temps réel
- 8.10.3. Politiques de planification
- 8.10.4. Ordonnancements cycliques
- 8.10.5. Planificateurs avec des propriétés statiques
- 8.10.6. Des planificateurs aux propriétés dynamiques

## **Module 9.** Design et développement de jeux web

9.1. Origines et normes du web
- 9.1.1. Origines d'internet
- 9.1.2. Création du *World Wide Web*
- 9.1.3. Émergence des normes web
- 9.1.4. L'essor des normes web

9.2. HTTP et structure client-serveur
- 9.2.1. Rôle client-serveur
- 9.2.2. Communication client-serveur
- 9.2.3. Histoire récente
- 9.2.4. Informatique centralisée

9.3. Programmation web: introduction
- 9.3.1. Concepts de base
- 9.3.2. Configuration d'un serveur web
- 9.3.3. Les bases du HTML5
- 9.3.4. Formulaires HTML

9.4. Introduction au HTML et exemples
- 9.4.1. Histoire du HTML5
- 9.4.2. Éléments du HTML5
- 9.4.3. APIS
- 9.4.4. CCS3

9.5. Modèle d'objet de document
- 9.5.1. Qu'est-ce que le Document Object Model?
- 9.5.2. Utilisation de *DOCTYPE*
- 9.5.3. L'importance de la validation du HTML
- 9.5.4. Accès aux éléments
- 9.5.5. Création d'éléments et de textes
- 9.5.6. Utilisation d'innerHTML
- 9.5.7. Suppression d'un élément de texte ou d'un nœud
- 9.5.8. Lecture et écriture des attributs d'un élément
- 9.5.9. Manipulation des styles d'éléments
- 9.5.10. Joindre plusieurs fichiers à la fois

9.6. Introduction au CSS et exemples
- 9.6.1. Syntaxe CSS3
- 9.6.2. Feuilles de style
- 9.6.3. Tags
- 9.6.4. Sélecteurs
- 9.6.5. Conception Web avec CSS

9.7. Introduction à JavaScript et exemples
- 9.7.1. Qu'est-ce que JavaScript?
- 9.7.2. Brève histoire de la langue
- 9.7.3. Versions de JavaScript
- 9.7.4. Affichage d'une boîte de dialogue
- 9.7.5. Syntaxe JavaScript
- 9.7.6. Comprendre les *Scripts*
- 9.7.7. Espaces
- 9.7.8. Commentaires
- 9.7.9. Fonctions
- 9.7.10. JavaScript interne et externe

9.8. Fonctions JavaScript
- 9.8.1. Déclarations de fonctions
- 9.8.2. Expressions de fonctions
- 9.8.3. Fonctions d'appel
- 9.8.4. Récursion
- 9.8.5. Fonctions imbriquées et fermetures
- 9.8.6. Préservation des variables
- 9.8.7. Fonctions imbriquées
- 9.8.8. Conflits de noms
- 9.8.9. Fermetures
- 9.8.10. Paramètres d'une fonction

9.9. PlayCanvas pour le développement de jeux web
- 9.9.1. Qu'est-ce que PlayCanvas?
- 9.9.2. Configuration du projet
- 9.9.3. Création d'un objet
- 9.9.4. Ajout de la physique
- 9.9.5. Ajout d'un modèle
- 9.9.6. Modification des paramètres de gravité et de scène
- 9.9.7. Exécution de *Scripts*
- 9.9.8. Contrôle de la caméra

9.10. Phaser pour le développement de jeux web
- 9.10.1. Qu'est-ce que Phaser?
- 9.10.2. Chargement des ressources
- 9.10.3 Construire le monde
- 9.10.4. Plateformes
- 9.10.5. Le joueur
- 9.10.6. Ajout de la physique
- 9.10.7. Utilisation du clavier
- 9.10.8. Ramassage *pickups*
- 9.10.9. Points et notation
- 9.10.10. Bombes rebondissantes

## **Module 10.** Réseaux et systèmes multijoueurs

10.1. Histoire et évolution des jeux multijoueurs
- 10.1.1. Années 1970: premiers jeux multijoueurs
- 10.1.2. Les années 90: Duke Nukem, Doom, Quake
- 10.1.3. L'essor des jeux vidéo multijoueurs
- 10.1.4. Multijoueur local et en ligne
- 10.1.5. Jeux de société

10.2. Modèles commerciaux multijoueurs
- 10.2.1. Origine et fonctionnement des modèles d'entreprise émergents
- 10.2.2. Services de vente en ligne
- 10.2.3. Jouer gratuitement
- 10.2.4. Micropaiements
- 10.2.5. Publicité
- 10.2.6. Abonnement avec paiements mensuels
- 10.2.7. Pay-per-play
- 10.2.8. Essayez avant d'acheter

10.3. Jeux locaux et jeux en réseau
- 10.3.1. Les jeux locaux: pour commencer
- 10.3.2. Jeux de société: Nintendo et la convivialité en famille
- 10.3.3. Les jeux en réseau: pour commencer
- 10.3.4. Évolution des jeux en réseau

10.4. Modèle OSI: Couches I
- 10.4.1. Modèle OSI: Introduction
- 10.4.2. Couche physique
- 10.4.3. Couche liaison de données
- 10.4.4. Couche réseau

10.5. Modèle OSI: couche II
- 10.5.1. Couche de transport
- 10.5.2. Couche session
- 10.5.3. Couche de présentation
- 10.5.4. Couche d'application

10.6. Réseaux informatiques et Internet
- 10.6.1. Qu'est-ce qu'un réseau informatique?
- 10.6.2. Software
- 10.6.3. Hardware
- 10.6.4. Serveurs
- 10.6.5. Stockage en réseau
- 10.6.6. Protocoles de réseau

10.7. Réseaux mobiles et sans fil
- 10.7.1. Réseau mobile
- 10.7.2. Réseau sans fil
- 10.7.3. Fonctionnement des réseaux mobiles
- 10.7.4. Technologie numérique

10.8. Sécurité
- 10.8.1. Sécurité personnelle
- 10.8.2. *Hacks* et *Cheats* dans les jeux vidéo
- 10.8.3. Sécurité anti-fraude
- 10.8.4. Analyse des systèmes de sécurité anti-triche

10.9. Systèmes multi-joueurs: serveurs
- 10.9.1. Hébergement de serveurs
- 10.9.2. Jeux vidéo MMO
- 10.9.3. Serveurs dédiés aux jeux vidéo
- 10.9.4. *LAN Parties*

10.10. Conception et programmation de jeux multi-joueurs
- 10.10.1. Principes fondamentaux de la conception de jeux multijoueurs Unreal
- 10.10.2. Principes fondamentaux de la conception de jeux vidéo multijoueurs dans Unity
- 10.10.3. Comment rendre un jeu multijoueur amusant?
- 10.10.4. Plus qu'un joystick: Innovation dans les joystick multijoueurs

*Grâce à la méthode de Relearning, basée sur la répétition des concepts clés, vous pourrez réduire les longues heures d'étude"*

06

# Stages Pratiques

Une fois que le professionnel du jeu vidéo a terminé la formation théorique en ligne, cette qualification comprend un stage dans un studio de création et de développement de jeux vidéo. Les étudiants seront ainsi en contact direct avec les équipes de développeurs du secteur et pourront démontrer tout ce qu'ils ont appris. Pendant cette phase d'apprentissage, les étudiants auront un tuteur qui les accompagnera jusqu'à la fin de la formation.

*Asseyez-vous à côté de grands développeurs et travaillez en équipe pendant la phase pratique de cette formation"*

La période de Formation Pratique de ce programme de Programmation de Jeux Vidéo dure 3 semaines. Les étudiants se rendront au studio de création et de développement de jeux vidéo du lundi au vendredi, 8 heures par jour, pour suivre l'enseignement pratique auprès d'un spécialiste.

Cette phase pratique permettra aux étudiants de déployer toutes leurs connaissances en programmation acquises dans ce Mastère Hybride. Avec d'autres professionnels de cette entreprise, vous pourrez vous coordonner avec le reste de l'équipe et apprendre les principaux outils et logiciels de programmation. Vous vivrez ainsi une expérience d'apprentissage proche de la réalité de l'industrie du jeu vidéo.

Au cours de son séjour dans ce studio, le professionnel ne se contentera pas d'utiliser le langage de programmation habituel, mais il sera également en mesure de concevoir des jeux vidéo multijoueurs. Une catégorie de jeux vidéo qui a le vent en poupe et qui fascine les *Gamers*.

La partie pratique sera réalisée avec la participation active de l'étudiant qui réalisera les activités et les procédures de chaque domaine de compétence (apprendre à apprendre et apprendre à faire), avec l'accompagnement et les conseils des enseignants et autres collègues formateurs qui faciliteront le travail en équipe et l'intégration multidisciplinaire en tant que compétences transversales pour la pratique de la Programmation de Jeux Vidéo (apprendre à être et apprendre à être en relation).

*Formez-vous dans un studio qui vous donnera l'occasion de démontrer votre potentiel en tant que programmeur de jeux vidéo"*



Les procédures décrites ci-dessous constitueront la base de la partie pratique de la formation et leur mise en œuvre sera en fonction de la disponibilité et de la charge de travail du studio, les activités proposées étant les suivantes:

| Module | Activité pratique | Quantité |
|---|---|---|
| Création du Structure de données et algorythmes | Réaliser la technique de *Backtracking*, pour la création de données et d'algorithmes | 1 |
| | Participer à l'analyse des algorithmes en vue de gains d'efficacité | |
| | Effectuer des tâches de mesure des données d'entrée et du temps d'exécution | |
| | Créer des algorithmes de tri avec des arbres, des *Heaps*, des graphes et avec *Greedy* | |
| Programmation orientée aux objets | Utiliser les *Modèles Factory, Singleton, Observer et Composite dans la création d'objets* | 1 |
| | Créer, capturer et gérer les exceptions dans la création d'objets | |
| | Effectuer une programmation concurrente | |
| | Utiliser des mécanismes de verrouillage et de communication | |
| | Créer de la documentation et des tests de logiciels | |
| Programmation en temps réel | Créer et synchroniser *Threads* | 1 |
| | Programmer *Shaders* | |
| | Mettre en œuvre le modèle en Java et utiliser les bibliothèques Java | |
| | Créer des post-effets | |
| | Traiter interruptions, synchronisation et communication inter-processus | |
| Conception et développement de jeux web | Programmer le web avec des formulaires HTML | 1 |
| | Utiliser le DOCTYPE et innerHTML pour développer des jeux web | |
| | Utiliser PlayCanvas pour développer des jeux en ligne | |
| | Mettre en place un projet de conception et de développement de jeux en ligne | |

## Assurance responsabilité civile

La principale préoccupation de cette institution est de garantir la sécurité des stagiaires et des autres collaborateurs nécessaires aux processus de formation pratique dans l'entreprise. Parmi les mesures destinées à atteindre cet objectif figure la réponse à tout incident pouvant survenir au cours de la formation d'apprentissage.

Pour ce faire, cette université s'engage à souscrire une assurance Responsabilité Civile pour couvrir toute éventualité pouvant survenir pendant le séjour au centre de stage.

Cette police d'assurance couvrant la Responsabilité Civile des stagiaires doit être complète et doit être souscrite avant le début de la période de Formation Pratique. Ainsi, le professionnel n'a pas à se préoccuper des imprévus et bénéficiera d'une couverture jusqu'à la fin du stage pratique dans le centre.

## Conditions générales de la Formation pratique

Les conditions générales de la Convention de Stage pour le programme sont les suivantes:

1.TUTEUR: Pendant le Mastère Hybride, l'étudiant se verra attribuer deux tuteurs qui l'accompagneront tout au long du processus, en résolvant tous les doutes et toutes les questions qui peuvent se poser. D'une part, il y aura un tuteur professionnel appartenant au centre de placement qui aura pour mission de guider et de soutenir l'étudiant à tout moment. D'autre part, un tuteur académique sera également assigné à l'étudiant, et aura pour mission de coordonner et d'aider l'étudiant tout au long du processus, en résolvant ses doutes et en lui facilitant tout ce dont il peut avoir besoin. De cette manière, le professionnel sera accompagné à tout moment et pourra consulter les doutes qui pourraient surgir, tant sur le plan pratique que sur le plan académique.

2. DURÉE: Le programme de formation pratique se déroulera sur trois semaines continues, réparties en journées de 8 heures, cinq jours par semaine. Les jours de présence et l'emploi du temps relèvent de la responsabilité du centre, qui en informe dûment et préalablement le professionnel, et suffisamment à l'avance pour faciliter son organisation.

3. ABSENCE: En cas de non présentation à la date de début du Mastère Hybride, l'étudiant perdra le droit au stage sans possibilité de remboursement ou de changement de dates. Une absence de plus de deux jours au stage, sans raison médicale justifiée, entraînera l'annulation du stage et, par conséquent, la résiliation automatique du contrat. Tout problème survenant au cours du séjour doit être signalé d'urgence au tuteur académique.

4. CERTIFICATION: Les étudiants qui achèvent avec succès le Mastère Hybride recevront un certificat accréditant le séjour pratique dans le centre en question.

5. RELATION DE TRAVAIL: Le Mastère Hybride ne constituera en aucun cas une relation de travail de quelque nature que ce soit.

6. PRÉREQUIS: Certains centres peuvent être amener à exiger des réferences académiques pour suivre le Mastère Hybride. Dans ce cas, il sera nécessaire de le présenter au département de formations de TECH afin de confirmer l'affectation du centre choisi.

7. NON INCLUS: Le mastère Hybride n'inclus auncun autre élément non mentionné dans les présentes conditions. Par conséquent, il ne comprend pas l'hébergement, le transport vers la ville où le stage a lieu, les visas ou tout autre avantage non décrit.

Toutefois, les étudiants peuvent consulter leur tuteur académique en cas de doutes ou de recommandations à cet égard. Ce dernier lui fournira toutes les informations nécessaires pour faciliter les démarches.

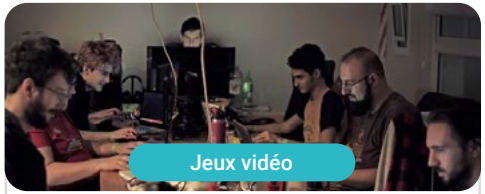
07

# Où suivre la Formation Pratique?

Ce Mastère Hybride comprend un séjour pratique dans un studio de référence où les étudiants mettront en pratique tout ce qu'ils ont appris sur la Programmation de Jeux Vidéo. TECH offre ainsi aux étudiants la possibilité d'apporter ce diplôme aux professionnels du jeu vidéo qui souhaitent acquérir une formation de qualité en phase avec les exigences actuelles du secteur.

*Atteignez vos objectifs en mettant en œuvre toutes vos connaissances dans un studio de référence dans le secteur des jeux vidéo"*

Les étudiants peuvent suivre la partie pratique de ce Mastère Hybride dans le studio suivant:



Jeux vidéo

### Startreming Games

Pays
Argentine

Ville
**Mendoza**

Adresse: Ruta 160 esquina Buenos Aires 88, San Rafael, Mendoza

Studio de développement de Jeux Vidéo indépendants avec Home Office

**Formations pratiques connexes:**
- Programmation de jeux
- Modélisation 3D Hard Surface

**Programmation de Jeux Vidéo**

*Profitez de l'occasion pour vous informer sur les derniers développements dans ce domaine afin de les appliquer à votre pratique quotidienne"*

08

# Méthodologie

Ce programme de formation offre une manière différente d'apprendre. Notre méthodologie est développée à travers un mode d'apprentissage cyclique: ***el Relearning***.

Ce système d'enseignement est utilisé, par exemple, dans les écoles de médecine les plus prestigieuses du monde et a été considéré comme l'un des plus efficaces par des publications de premier plan telles que le ***New England Journal of Medicine.***

*Découvrez le Relearning, un système qui abandonne l'apprentissage linéaire conventionnel pour vous emmener à travers des systèmes d'enseignement cycliques: une façon d'apprendre qui s'est avérée extrêmement efficace, en particulier dans les matières qui nécessitent une mémorisation"*

## À TECH, nous utilisons la méthode des cas

Notre programme propose une méthode révolutionnaire de développement des compétences et des connaissances. Notre objectif est de renforcer les compétences dans un contexte changeant, compétitif et exigeant.

*Avec TECH, vous pourrez découvrir une façon d'apprendre qui fait avancer les fondations des universités traditionnelles du monde entier"*


*Vous bénéficierez d'un système d'apprentissage basé sur la répétition, avec un enseignement naturel et progressif sur l'ensemble du cursus.*

*L'étudiant apprendra, par le biais d'activités collaboratives et de cas réels, à résoudre des situations complexes dans des environnements commerciaux réels.*

## Une méthode d'apprentissage innovante et différente

Cette formation TECH est un programme d'enseignement intensif, créé de toutes pièces, qui propose les défis et les décisions les plus exigeants dans ce domaine, tant au niveau national qu'international. Grâce à cette méthodologie, l'épanouissement personnel et professionnel est stimulé, faisant ainsi un pas décisif vers la réussite. La méthode des cas, technique qui constitue la base de ce contenu, permet de suivre la réalité économique, sociale et professionnelle la plus actuelle.

> *Notre programme vous prépare à relever de nouveaux défis dans des environnements incertains et à réussir votre carrière"*

La méthode du cas a été le système d'apprentissage le plus utilisé dans les meilleures écoles de commerce du monde depuis qu'elles existent. Développée en 1912 pour que les étudiants en Droit n'apprennent pas seulement le droit sur la base d'un contenu théorique, la méthode des cas consiste à leur présenter des situations réelles complexes afin qu'ils prennent des décisions éclairées et des jugements de valeur sur la manière de les résoudre. Elle a été établie comme méthode d'enseignement standard à Harvard en 1924.

Face à une situation donnée, que doit faire un professionnel? C'est la question à laquelle nous vous confrontons dans la méthode des cas, une méthode d'apprentissage orientée vers l'action. Pendant 4 ans, vous serez confronté à de multiples cas réels. Vous devrez intégrer toutes vos connaissances, faire des recherches, argumenter et défendre vos idées et vos décisions.

## Relearning Methodology

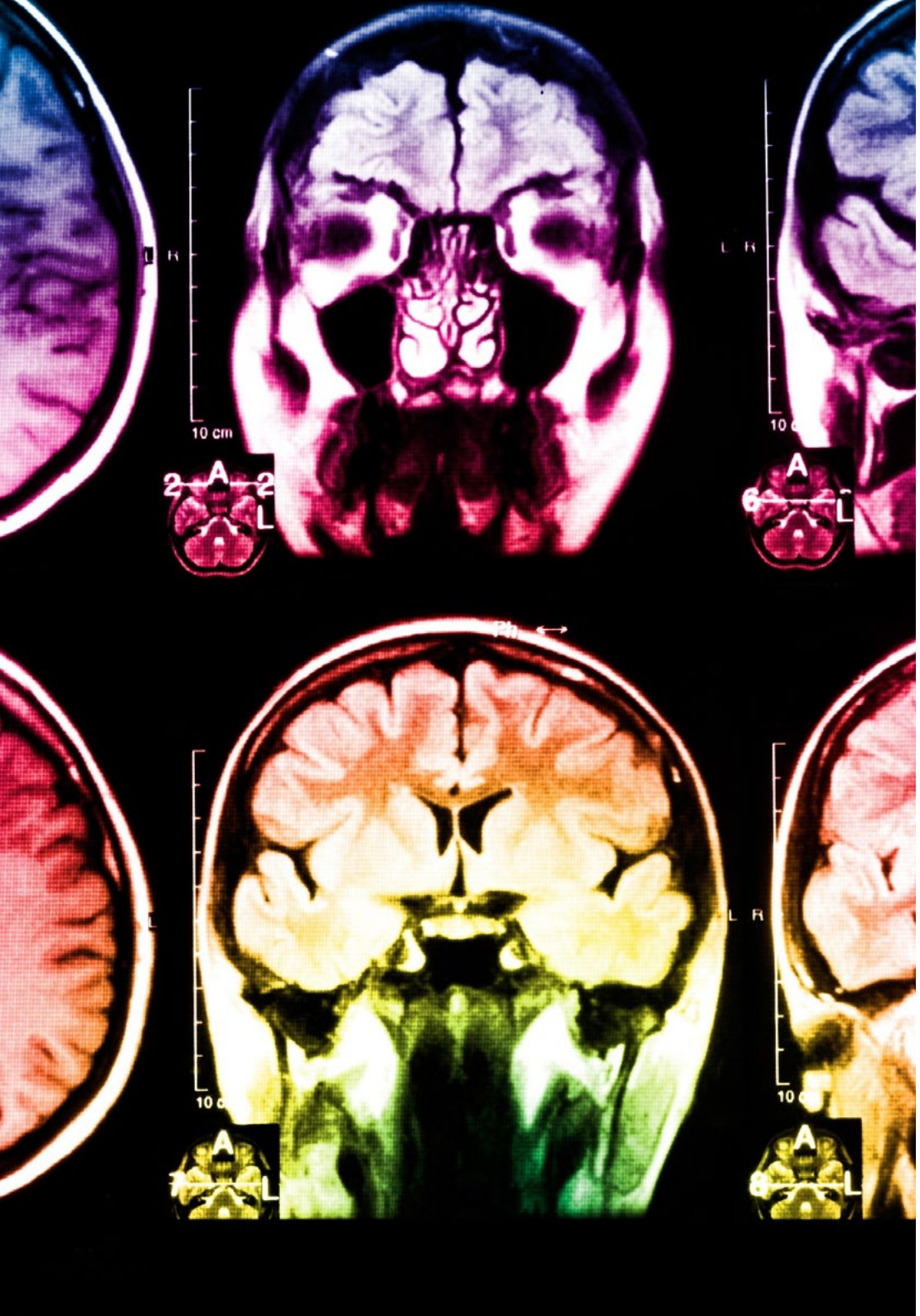
TECH combine efficacement la méthodologie des études de cas avec un système d'apprentissage 100% en ligne basé sur la répétition, qui associe 8 éléments didactiques différents dans chaque leçon.

Nous enrichissons l'Étude de Cas avec la meilleure méthode d'enseignement 100% en ligne: le Relearning.

*En 2019, nous avons obtenu les meilleurs résultats d'apprentissage de toutes les universités en ligne du monde.*

À TECH, vous apprendrez avec une méthodologie de pointe conçue pour former les managers du futur. Cette méthode, à la pointe de la pédagogie mondiale, est appelée Relearning.

Notre université est actuellement université  hispanophone à posséder la licence l'autorisant à utiliser la méthode d'apprentissage Relearning. En 2019, nous avons réussi à améliorer les niveaux de satisfaction globale de nos étudiants (qualité de l'enseignement, qualité des supports, structure des cours, objectifs...) par rapport aux indicateurs de la meilleure université en ligne.



Dans notre programme, l'apprentissage n'est pas un processus linéaire mais il se déroule en spirale (nous apprenons, désapprenons, oublions et réapprenons). Par conséquent, ils combinent chacun de ces éléments de manière concentrique. Grâce à cette méthodologie, nous avons formé plus de 650.000 diplômés universitaires avec un succès sans précédent et ce dans toutes les spécialités aussi divers que la biochimie, la génétique, la chirurgie, le droit international, les compétences en matière de gestion, les sciences du sport, la philosophie, le droit, l'ingénierie, le journalisme, l'histoire ou les marchés et instruments financiers. Tout cela dans un environnement très exigeant, avec un corps étudiant universitaire au profil socio-économique élevé et dont l'âge moyen est de 43,5 ans.

*Le Relearning vous permettra d'apprendre plus facilement et de manière plus productive tout en développant un esprit critique, en défendant des arguments et en contrastant des opinions: une équation directe vers le succès.*

À partir des dernières preuves scientifiques dans le domaine des neurosciences, non seulement nous savons comment organiser les informations, les idées, les images et les souvenirs, mais nous savons aussi que le lieu et le contexte dans lesquels nous avons appris quelque chose sont fondamentaux pour notre capacité à nous en souvenir et à le stocker dans l'hippocampe, pour le conserver dans notre mémoire à long terme.

De cette façon, et dans ce que l'on appelle Neurocognitive context-dependent e-learning les différents éléments de notre programme sont liés au contexte dans lequel le participant développe sa pratique professionnelle.

**Dans ce programme, vous aurez accès aux meilleurs supports pédagogiques élaborés spécialement pour vous:**

### Support d'étude

Tous les contenus didactiques sont créés par les spécialistes qui enseignent les cours. Ils ont été conçus en exclusivité pour la formation afin que le développement didactique soit vraiment spécifique et concret.

Ces contenus sont ensuite appliqués au format audiovisuel, pour créer la méthode de travail TECH en ligne. Tout cela, élaboré avec les dernières techniques afin d'offrir des éléments de haute qualité dans chacun des supports qui sont mis à la disposition de l'apprenant.

### Cours magistraux

Il existe de nombreux faits scientifiques prouvant l'utilité de l'observation par un tiers expert.

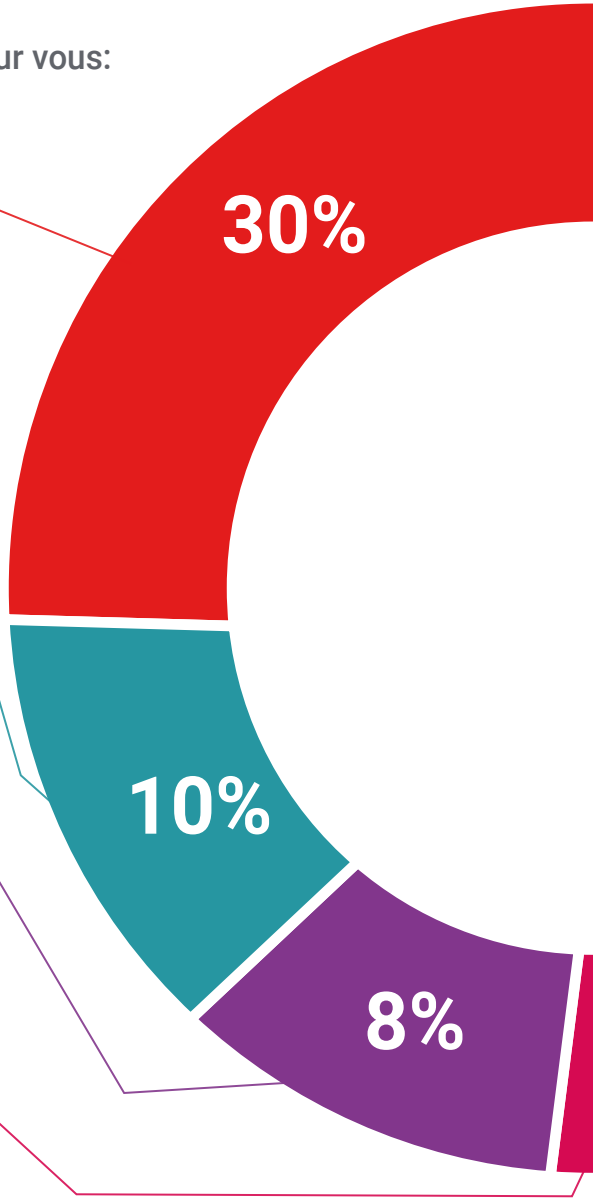
La méthode "Learning from an Expert" renforce les connaissances et la mémoire, et génère de la confiance pour les futures décisions difficiles.

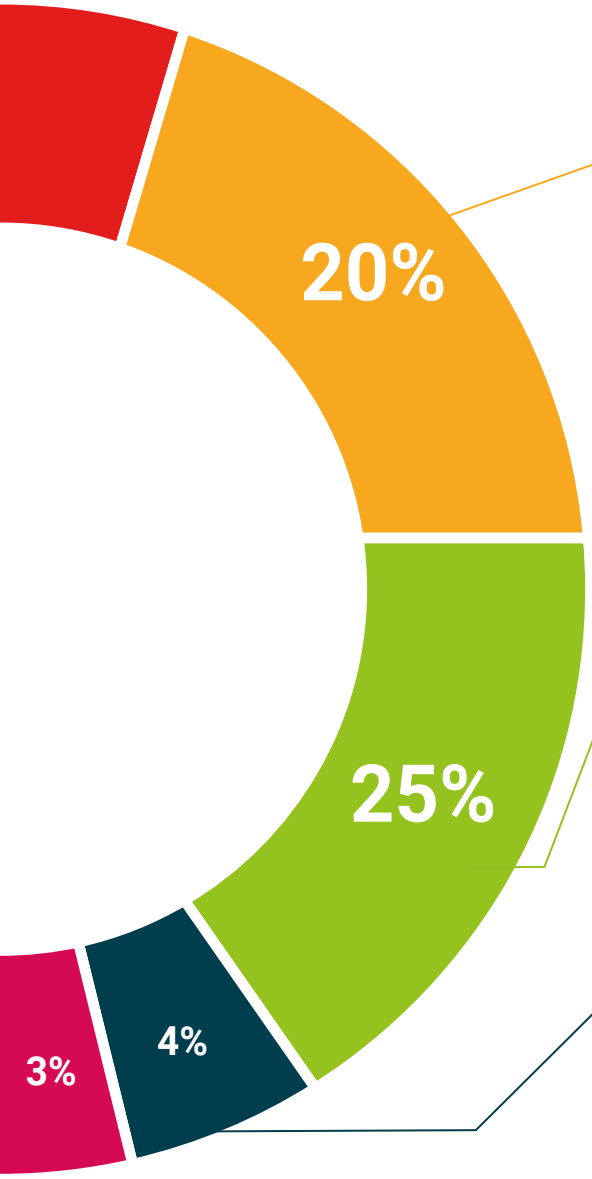
### Pratique des aptitudes et des compétences

Vous réaliserez des activités de développement des compétences et des compétences spécifiques dans chaque domaine thématique. Pratiques et dynamiques pour acquérir et développer les compétences et aptitudes qu'un spécialiste doit développer dans le cadre de la mondialisation dans laquelle nous vivons.

### Bibliographie complémentaire

Articles récents, documents de consensus, guides internationaux et autres supports. Dans la bibliothèque virtuelle de TECH, l'étudiant aura accès à tout ce dont il a besoin pour compléter sa formation.






### Case Studies

Ils réaliseront une sélection des meilleures études de cas choisies spécifiquement pour ce diplôme. Des cas présentés, analysés et tutorés par les meilleurs spécialistes de la scène internationale.

### Résumés interactifs

Nous présentons les contenus de manière attrayante et dynamique dans des dossiers multimédias comprenant des fichiers audios, des vidéos, des images, des diagrammes et des cartes conceptuelles afin de consolider les connaissances.

Ce système unique de formation à la présentation de contenus multimédias a été récompensé par Microsoft en tant que "European Success Story".

### Testing & Retesting

Nous évaluons et réévaluons périodiquement vos connaissances tout au long du programme, par le biais d'activités et d'exercices d'évaluation et d'auto-évaluation: vous pouvez ainsi constater vos avancées et savoir si vous avez atteint vos objectifs.

09

# Diplôme

Le Mastère Hybride en Programmation de Jeux Vidéo garantit, en plus de la formation la plus rigoureuse et actualisée, l'accès à un diplôme de Mastère Hybride délivré par TECH Université Technologique.

*Terminez ce programme avec succès et recevez votre diplôme sans avoir à vous soucier des déplacements ou des formalités administratives”*

Ce **Mastère Hybride en Programmation de Jeux Vidéo** contient le programme scientifique le plus complet et le plus actuel du marché.

Après avoir réussi les évaluation, l'étudiant recevra par courrier postal avec accusé de réception le diplôme de **Mastère Hybride**, qui accréditera la réussite des évaluations et l'acquisition des compétences du programme.

En complément du diplôme, vous pourrez obtenir un certificat de qualification, ainsi qu'une attestation du contenu du programme. Pour ce faire, vous devrez contacter votre conseiller académique, qui vous fournira toutes les informations nécessaires.

Titre: **Mastère Hybride en Programmation de Jeux Vidéo**

Modalité: **Hybride (En ligne + Formation Pratique)**

Durée: **12 mois**

Diplôme: **TECH Université Technologique**

Heures officielles: **1.620 h.**


tech université technologique

Délivre le présent

DIPLÔME

a

Mme/M. _____, avec nº d'identification _____

Pour avoir finalisé et accrédité avec succès le programme de

MASTÈRE HYBRIDE

en

Programmation de Jeux Vidéo

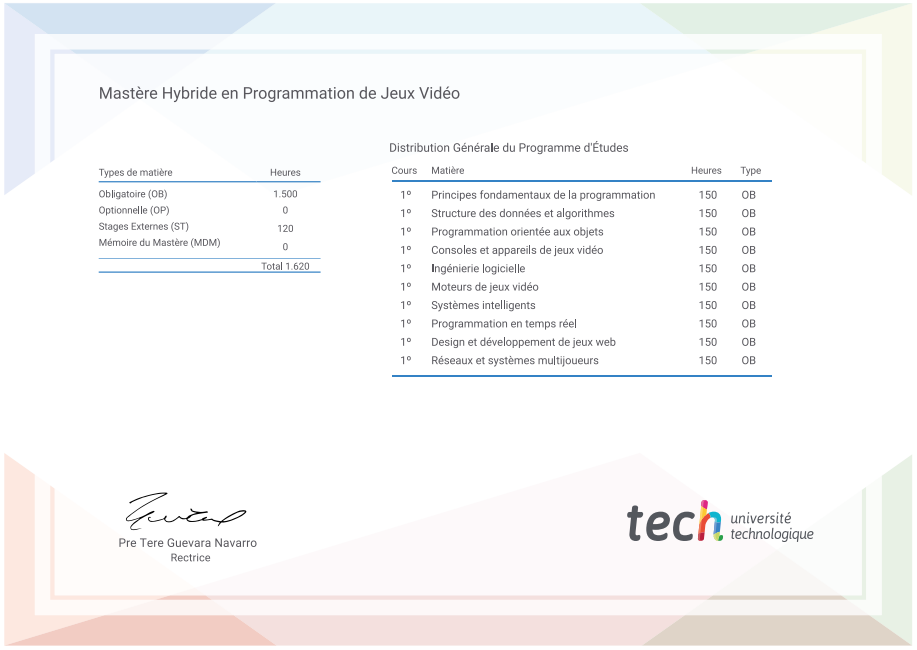
Il s'agit d'un diplôme spécialisé octroyé par cette Université d'une durée de 1.620 heures, débutant le dd/mm/aaaa et finalisant le dd/mm/aaaa.

TECH est une Institution Privée d'Enseignement Supérieur reconnue par le Ministère de l'Enseignement Public depuis le 28 juin 2018.

Fait le 17 juin 2020

Pre Tere Guevara Navarro
Rectrice

Code Unique TECH: AFWOR23S techtitute.com/diplômes


Mastère Hybride en Programmation de Jeux Vidéo

| Types de matière | Heures |
|---|---|
| Obligatoire (OB) | 1.500 |
| Optionnelle (OP) | 0 |
| Stages Externes (ST) | 120 |
| Mémoire du Mastère (MDM) | 0 |
| | Total 1.620 |

Distribution Générale du Programme d'Études

| Cours | Matière | Heures | Type |
|---|---|---|---|
| 1º | Principes fondamentaux de la programmation | 150 | OB |
| 1º | Structure des données et algorithmes | 150 | OB |
| 1º | Programmation orientée aux objets | 150 | OB |
| 1º | Consoles et appareils de jeux vidéo | 150 | OB |
| 1º | Ingénierie logicielle | 150 | OB |
| 1º | Moteurs de jeux vidéo | 150 | OB |
| 1º | Systèmes intelligents | 150 | OB |
| 1º | Programmation en temps réel | 150 | OB |
| 1º | Design et développement de jeux web | 150 | OB |
| 1º | Réseaux et systèmes multijoueurs | 150 | OB |

Pre Tere Guevara Navarro
Rectrice

tech université technologique

*Si l'étudiant souhaite que son diplôme version papier possède l'Apostille de La Haye, TECH EDUCATION fera les démarches nécessaires pour son obtention moyennant un coût supplémentaire.


tech université technologique

# Mastère Hybride
## Programmation de Jeux Vidéo

Modalité: **Hybride (En ligne + Formation Pratique)**
Durée: **12 mois**
Diplôme: **TECH Université Technologique**
Heures officielles: **1.620 h.**

# Mastère Hybride
## Programmation de Jeux Vidéo

tech université technologique