OpenCertif
Centre de formation et de certification professionnelle

SYLLABUS OFFICIEL

Examen UCU Programmer

Programmation de jeux vidéo avec Unity et C#

Certification : Unity Certified User — Programmer

Niveau : Foundation / Entry-level | Public : Étudiants / Débutants en développement Unity

1. Présentation de la certification

L'examen **Unity Certified User : Programmer (UCU Programmer)** valide les compétences fondamentales en **programmation de jeux vidéo avec Unity et le langage C#**. Cette certification atteste de votre capacité à utiliser les composants Unity pour créer des objets interactifs, programmer des comportements et logiques de jeu en C#, gérer les entrées utilisateur, les collisions et les événements, contrôler animations, sons et effets visuels, et construire un jeu fonctionnel sur plusieurs plateformes.

La réussite de cet examen unique conduit à l'obtention de la certification **Unity Certified User — Programmer**, reconnue internationalement et délivrée par Unity Technologies via Certiport (Pearson VUE). Elle constitue une première étape idéale pour les étudiants et débutants souhaitant démarrer une carrière dans le développement de jeux vidéo, d'applications mobiles, d'AR/VR et d'expériences interactives temps réel (gaming, entertainment, automobile, XR, AEC).

Informations clés

Code de l'examen	UCU Programmer (Unity Certified User — Programmer)
Intitulé officiel	Unity Certified User : Programmer
Certification obtenue	Unity Certified User — Programmer
Éditeur officiel	Unity Technologies
Centre de test	Certiport (Pearson VUE)
Niveau	Foundation — niveau débutant / entry-level
Version Unity ciblée	Unity 6 (version 2026)
Durée de l'examen	Environ 50 minutes
Nombre de questions	Environ 40 questions (QCM, appariement, sélection d'image, cas pratiques)
Score requis	500 sur 700 (échelle officielle Unity : 200 à 700 points)
Prérequis recommandé	Au moins 150 heures de pratique Unity (recommandation officielle Unity)
Langue de l'examen	Anglais

Âge minimum recommandé	14 ans et plus
Validité de la certification	3 ans à partir de la date de réussite (stackable, attribuée une seule fois)
Politique de reprise	Délai d’attente de 24 heures avant la 1re reprise (voucher retake à utiliser sous 60 jours)
Modalité	En centre agréé Certiport (CATC) — OpenCertif est un centre Certiport autorisé
Badge numérique	Badge officiel Credly délivré automatiquement après réussite

2. Profil du candidat

En tant que candidat à l'examen UCU Programmer, vous développez et validez des compétences fondamentales en programmation de jeux vidéo avec Unity. Vous êtes capable de :

- Naviguer dans l'interface Unity Editor et utiliser ses principales fenêtres.
- Écrire et lire du code C# au sein de scripts Unity (MonoBehaviour).
- Utiliser les composants Unity (Transform, Rigidbody, Collider, Animator, etc.).
- Créer des mécaniques de gameplay et logiques de jeu interactives.
- Gérer les entrées utilisateur (clavier, souris, manette, tactile).
- Implémenter des collisions, triggers et événements physiques.
- Contrôler les animations, sons et effets visuels par code.
- Déboguer du code et interpréter les messages de la console Unity.
- Construire et déployer un jeu fonctionnel sur plusieurs plateformes (Standalone, Mobile, WebGL).

L'examen évalue spécifiquement quatre familles de compétences techniques essentielles à tout développeur Unity débutant :

- Débogage, résolution de problèmes et interprétation de la Scripting API.
- Création de code C# fonctionnel et conforme aux conventions Unity.
- Évaluation et analyse de code existant (lecture, compréhension, anticipation de comportement).
- Navigation et utilisation efficace de l'interface Unity Editor.

3. Prérequis et public cible OpenCertif

Aucun prérequis académique formel n'est exigé pour passer l'examen UCU Programmer. Unity Technologies recommande toutefois aux candidats de disposer d'une expérience pratique préalable :

- **Au moins 150 heures de pratique Unity** (recommandation officielle Unity).
- Notions de base en programmation (variables, conditions, boucles, fonctions).
- Compréhension du langage C# (syntaxe, types, classes, méthodes).
- Familiarité avec l'environnement Unity Editor.
- Connaissance des concepts fondamentaux du développement 3D (GameObjects, Transform, scènes).
- Anglais niveau scolaire suffisant pour comprendre les questions de l'examen.

Public cible OpenCertif

- Étudiants en informatique, multimédia, game design ou animation 3D.
- Lycéens et collégiens (à partir de 14 ans) intéressés par le développement de jeux.
- Auto-didactes et passionnés souhaitant valider leurs compétences Unity / C#.
- Professionnels en reconversion vers le développement de jeux vidéo.

- Candidats aux métiers de Unity Developer, Mobile App Developer, Gameplay Programmer.
- Futurs candidats aux certifications Unity de niveau supérieur (Associate, Professional).

4. Domaines de compétences mesurées

L'examen est structuré autour de 4 grands domaines de compétences. Le tableau ci-dessous indique le poids relatif de chaque domaine dans l'évaluation finale (version 2026 — aligné sur Unity 6). Les pondérations sont des estimations issues du guide officiel Unity / Certiport.

Domaine	Intitulé	Pondération
1	Débogage, résolution de problèmes et interprétation de l'API	25 — 30 %
2	Création de code C# dans Unity	25 — 30 %
3	Évaluation et analyse de code	20 — 25 %
4	Navigation dans l'interface Unity Editor	20 — 25 %

*Remarque : l'examen UCU Programmer dure environ 50 minutes pour 40 questions, soit environ 1 minute 15 par question. La gestion du temps est essentielle. Le score requis pour valider est de **500 sur 700** (sur une échelle officielle Unity de 200 à 700 points).*

5. Détail des compétences mesurées

Cette section détaille de manière exhaustive l'ensemble des compétences couvertes par l'examen UCU Programmer, en s'appuyant sur les Objective Domains publiés par Certiport et Unity Technologies (version 2026 — aligné sur Unity 6).

1 Débogage, résolution de problèmes et interprétation de l'API — 25 — 30 %

1.1 Interpréter les messages de la console et les logs de débogage

- Identifier le code source qui a généré un message Debug.Log(), Debug.LogWarning() ou Debug.LogError().
- Distinguer warnings, errors et exceptions dans la console Unity.
- Comprendre la stack trace d'une exception et localiser la ligne fautive.
- Identifier les erreurs de compilation C# courantes (syntaxe, types, références manquantes).
- Interpréter les NullReferenceException, MissingComponentException, IndexOutOfRangeException.

1.2 Résoudre des problèmes de code

- Identifier la cause d'un bug à partir d'un extrait de code et de son comportement observé.
- Proposer une correction pour un code qui ne compile pas ou qui plante à l'exécution.
- Reconnaître les erreurs logiques (mauvais opérateur, condition inversée, boucle infinie).
- Identifier les problèmes de références non assignées dans l'Inspector.

1.3 Interpréter la Scripting API Unity

- Lire et comprendre une signature de méthode dans la documentation Unity.
- Identifier la classe Unity à utiliser pour une tâche donnée (Transform, GameObject, Component, Mathf, Random, Time, Input, etc.).
- Distinguer méthodes statiques et méthodes d'instance.
- Comprendre les paramètres et valeurs de retour des méthodes courantes : Instantiate, Destroy, GetComponent, Find, Vector3.Lerp, etc.
- Reconnaître les surcharges (overloads) d'une méthode Unity.

2 Création de code C# dans Unity — 25 — 30 %

2.1 Syntaxe C# et fondamentaux

- Déclarer des variables et choisir le type approprié (int, float, bool, string, Vector3, GameObject, etc.).
- Utiliser les structures de contrôle : if / else, switch, for, while, foreach.
- Écrire des méthodes avec paramètres et valeur de retour.
- Comprendre la portée (public, private, protected) et les modificateurs (static, readonly).
- Utiliser les collections de base : tableaux, List, Dictionary.
- Manipuler des chaînes de caractères et faire de la concaténation.

2.2 Programmation MonoBehaviour

- Créer un script C# qui hérite de MonoBehaviour.
- Utiliser les méthodes du cycle de vie : Awake(), Start(), Update(), FixedUpdate(), LateUpdate(), OnEnable(), OnDisable().
- Comprendre la différence entre Update et FixedUpdate (rendu vs physique).
- Déclarer des champs sérialisables visibles dans l'Inspector ([SerializeField], public).
- Utiliser [Header], [Tooltip], [Range], [HideInInspector] pour structurer l'Inspector.

2.3 Manipulation des GameObjects et composants

- Accéder à un composant : GetComponent(), TryGetComponent().
- Créer et détruire des GameObjects au runtime : Instantiate(), Destroy().
- Manipuler la position, rotation et échelle via Transform.
- Gérer la hiérarchie des GameObjects (parent / enfant, transform.SetParent).
- Activer / désactiver des GameObjects (SetActive) et des composants (.enabled).

2.4 Entrées utilisateur, physique et événements

- Lire les entrées clavier, souris et tactiles via Input (ou le nouveau Input System).
- Appliquer une force à un Rigidbody (AddForce, AddTorque) avec les différents ForceMode.
- Gérer les collisions et triggers (OnCollisionEnter, OnTriggerEnter, OnCollisionExit, etc.).
- Effectuer un raycast (Physics.Raycast) pour détecter des objets dans l'espace.
- Utiliser Time.deltaTime pour rendre les déplacements indépendants du framerate.

2.5 Animation, son, UI et persistance

- Piloter un Animator par code (SetTrigger, SetBool, SetFloat, SetInteger).
- Jouer des sons via AudioSource.Play() et AudioSource.PlayOneShot().
- Modifier dynamiquement des éléments UI : Canvas, Button, Text / TextMeshPro, Slider, Image.
- Charger des scènes via SceneManager.LoadScene().
- Sauvegarder des données simples via PlayerPrefs (entre sessions).

3 Évaluation et analyse de code — 20 — 25 %

3.1 Lire et comprendre un code Unity existant

- Prévoir le comportement d'un script en lisant son code source.
- Identifier l'ordre d'exécution des méthodes Unity dans une frame.
- Tracer manuellement les valeurs de variables au fil de l'exécution.
- Comprendre l'effet d'une modification d'un paramètre dans l'Inspector sur le comportement runtime.

3.2 Analyser le comportement attendu vs observé

- Comparer le code écrit avec le comportement attendu d'une mécanique de jeu.
- Identifier pourquoi un GameObject ne se déplace pas, ne tombe pas, ou ne répond pas aux entrées.
- Distinguer un problème de code, de configuration de composant, ou de hiérarchie de scène.
- Anticiper l'impact d'un changement d'ordre d'exécution sur le rendu visuel et la physique.

3.3 Optimisation et bonnes pratiques de base

- Identifier les appels coûteux à éviter dans Update (Find, GetComponent répété, Instantiate massif).
- Recommander la mise en cache de références dans Start ou Awake.
- Comprendre l'impact de Time.deltaTime sur la fluidité et la cohérence du gameplay.
- Identifier les configurations qui ralentissent un projet (collisions inutiles, lights non bakées).

4 Navigation dans l’interface Unity Editor — 20 — 25 %

4.1 Fenêtres principales de l’Editor

- Identifier et utiliser les fenêtres Hierarchy, Scene, Game, Inspector, Project, Console.
- Comprendre le rôle de la barre d’outils (Play, Pause, Step).
- Utiliser la fenêtre Animator, Animation, et Timeline.
- Accéder aux Preferences et Project Settings.

4.2 Gestion des assets et du projet

- Importer des assets (modèles 3D, textures, audio, scripts).
- Créer des Prefabs et comprendre leur instanciation.
- Utiliser le Package Manager pour ajouter des packages Unity (TextMeshPro, Input System, etc.).
- Organiser les assets dans des dossiers et utiliser les Tags et Layers.

4.3 Configuration des composants et de la scène

- Ajouter un composant à un GameObject via l’Inspector ou par code.
- Paramétrer un Rigidbody (mass, drag, useGravity, isKinematic).
- Configurer un Collider (Box, Sphere, Capsule, Mesh) et activer isTrigger.
- Configurer un Camera, un Light et un Canvas UI.
- Configurer le Build Settings pour cibler une plateforme (PC, Mac, Linux, Android, iOS, WebGL).

4.4 Construction et déploiement

- Construire un build standalone d’un projet Unity.
- Comprendre les différences entre les plateformes de build.
- Identifier les paramètres essentiels du Player Settings (nom, icône, résolution).
- Tester un build avant publication.

6. Modalités pédagogiques OpenCertif

OpenCertif accompagne les candidats au UCU Programmer à travers un parcours blended-learning complet, combinant ressources e-learning interactives, projets pratiques en Unity Editor, C#, Unity Scripting API, GameObjects, Rigidbody, Animator et Canvas UI et accompagnement tutoré.

Format de la formation

Durée recommandée	150 à 200 heures de pratique Unity (minimum recommandé par Unity — OpenCertif structure ce parcours sur 60 à 80 heures de formation tutorée complétées par 90 à 120 heures de projet personnel)
Modalité	100 % distanciel asynchrone, ou blended (distanciel + classes virtuelles)
Support pédagogique	Unity Certified User Courseware officiel (GMetrix) + ressources OpenCertif (modules Rise 360, scénarios immersifs)
Plateforme LMS	lmsopencertif.fr (Moodle) — accès 24/7 pendant 12 mois
Encadrement	Tutorat asynchrone par expert Unity certifié + classes virtuelles bimensuelles
Pratique requise	Au moins 150 heures de pratique Unity (recommandation officielle Unity Technologies)
Évaluations	Quiz formatifs par module, 3 projets pratiques Unity, examens blancs CertPREP
Certification finale	Passage de l’examen UCU Programmer en centre OpenCertif (CATC Certiport)

Parcours d’apprentissage proposé

- **Module 1 :** Découverte de l’écosystème Unity et du métier de game programmer.
- **Module 2 :** Navigation dans l’Unity Editor — Hierarchy, Scene, Inspector, Project, Console.
- **Module 3 :** Fondamentaux C# — variables, types, opérateurs, structures de contrôle.
- **Module 4 :** Programmation orientée objet en C# — classes, méthodes, héritage.
- **Module 5 :** Cycle de vie MonoBehaviour — Awake, Start, Update, FixedUpdate.
- **Module 6 :** Manipulation des GameObjects — Transform, Instantiate, Destroy, hiérarchie.
- **Module 7 :** Composants et Inspector — GetComponent, SerializeField, attributs.

- **Module 8 :** Entrées utilisateur — Input legacy et nouveau Input System.
- **Module 9 :** Physique Unity — Rigidbody, Collider, Triggers, Raycast, ForceMode.
- **Module 10 :** Animation — Animator Controller, transitions, paramètres, Blend Trees.
- **Module 11 :** Audio — AudioSource, AudioListener, PlayOneShot, mixage simple.
- **Module 12 :** UI — Canvas, RectTransform, Button, TextMeshPro, anchors.
- **Module 13 :** Gestion de scènes — SceneManager, persistance avec PlayerPrefs.
- **Module 14 :** Débogage — console, breakpoints, interprétation des exceptions.
- **Module 15 :** Optimisation — caching, Time.deltaTime, anti-patterns courants.
- **Module 16 :** Build et déploiement — Player Settings, multi-plateformes, Standalone, WebGL.
- **Module 17 :** Mini-projet 1 — jeu 2D à défilement (Side Scroller).
- **Module 18 :** Mini-projet 2 — jeu 3D simple à la 3e personne.
- **Module 19 :** Mini-projet 3 — prototype interactif (UI, audio, scènes multiples).
- **Module 20 :** Examen blanc et préparation finale CertPREP.

7. Ressources d’étude officielles

En complément du parcours OpenCertif, les ressources officielles Unity Technologies et Certiport suivantes sont fortement recommandées :

- Unity Learn (learn.unity.com) — plateforme officielle d’apprentissage Unity Technologies.
- Unity Manual (docs.unity3d.com/Manual) — documentation utilisateur complète.
- Unity Scripting API (docs.unity3d.com/ScriptReference) — référence complète de l’API C#.
- Unity Certified User Courseware (GMetrix) — 12 modules vidéo officiels + 3 projets pratiques.
- CertPREP Practice Tests — examens blancs Certiport alignés sur les objectifs UCU Programmer.
- Microsoft Learn — fondamentaux C# (learn.microsoft.com/dotnet/csharp).
- Unity Discussions / Unity Forum — communauté officielle.
- Page Certiport officielle : certiport.pearsonvue.com/Certifications/Unity/Certified-User.
- Page officielle Unity Certification : unity.com/products/unity-certifications.
- Chaine YouTube officielle Unity — tutoriels et live coding.
- Badge officiel délivré via Credly (credly.com).
- Pages OpenCertif dédiées : opencertif.fr/unity et opencertif.fr/unity-user-programmer.

8. Modalités de passage de l’examen

Inscription	Via OpenCertif ou directement auprès d’un centre Certiport
Centre d’examen	OpenCertif — Centre Certiport Authorized Testing Center (CATC) / Pearson VUE
Mode de passage	En centre uniquement (Unity n’autorise pas l’examen OnVUE à distance pour les certifications UCU — présence sur site requise)
Pièce d’identité	1 pièce d’identité avec photo obligatoire le jour de l’examen (pour les mineurs : autorisation parentale et CNI / passeport)
Aménagements	Demande possible auprès de Certiport (temps additionnel, assistance technique)
Résultat	Score communiqué immédiatement à la fin de l’examen (échelle 200-700, seuil de réussite 500)

Validité de la certification	3 ans à partir de la date de réussite — attribuée une seule fois (stackable, pas de renouvellement payant requis)
Politique de reprise	Délai d’attente de 24 heures avant la 1re reprise. Voucher retake à utiliser sous 60 jours après l’échec.
Badge numérique	Badge officiel délivré via Credly et intégrable à LinkedIn, CV, portfolio, sites de recrutement

9. Contact et inscription

Pour toute information complémentaire, demande de devis ou inscription à la formation préparatoire au UCU Programmer, l'équipe OpenCertif reste à votre disposition. OpenCertif est un Centre Certiport Authorized Testing Center (CATC) habilité à délivrer les certifications Unity Certified User.

OpenCertif
Centre de formation et de certification

app.opencertif.fr
lmsopencertif.fr

Centre agréé Certiport / Pearson VUE
Certifié Qualiopi — Actions de formation

10. Mentions légales et version

Ce syllabus est établi par OpenCertif sur la base des Objective Domains officiels publiés par Certiport pour la certification UCU Programmer, dans sa version applicable (version 2026 — aligné sur Unity 6). Les compétences mesurées, les pondérations et les objectifs présentés reflètent fidèlement la structure de l'examen telle que publiée par Unity Technologies via Certiport.

Version du syllabus : 2026.05 — Édition mai 2026
Source officielle Certiport : certiport.pearsonvue.com/Certifications/Unity/Certified-User/Certify
Source officielle Unity : unity.com/products/unity-certifications/user-programmer
Page OpenCertif : opencertif.fr/unity-user-programmer