unity

Цели экзамена

Сертификационный экзамен
для Unity-программистов

Роль

Unity-программисты занимаются разработкой интерактивного контента с использованием Unity. Работая с другими членами команды разработчиков (например, с аудиоспециалистами и художниками), Unity-программисты воплощают идею приложения с помощью редактора Unity, а также внедряют визуальные и звуковые ресурсы, созданные другими специалистами. Unity-программист — специалист широкого профиля, способный решать сложные вопросы программирования и отвечающий за выполнение широкого спектра технических задач, включая внедрение графических ресурсов, программирование пользовательского интерфейса, реализацию пользовательского взаимодействия и игровых правил, разработку логики состояний приложения, моделирование физики, отладку кода и оптимизацию производительности.

Сертификационный экзамен для Unity-программистов предназначен для профессиональной сертификации программистов начального и среднего уровня, а также выпускников высших учебных заведений, желающих устроиться на работу в различных отраслях. Для работодателя успешная сдача данного экзамена означает, что соискатель:

- знаком с программированием в контексте профессиональной разработки ПО, а также создания и поддержки приложений на базе платформы Unity;
- способен работать в рамках технических процессов, обладает логическим мышлением, находчив;
- может самостоятельно выполнять задачи по программированию типового или среднего уровня и решать более сложные технические задачи под руководством старших специалистов.

Подходящие должности:

- программист игрового процесса;
- инженер-программист;
- разработчик ПО;
- Unity-разработчик;
- разработчик мобильных приложений.

Требования

Программа сертификации создана для выпускников специальностей, связанных с программированием игр и информатикой; лиц, самостоятельно освоивших область знаний, равноценную двум годам обучения в вузе или имеющих аналогичный опыт работы программистом; профессионалов начального или среднего уровня с опытом использования Unity в работе. Желающие сдать экзамен должны иметь опыт разработки приложений с использованием Unity (самостоятельно или в коллективе), в результате которой был создан готовый прототип или техническая демоверсия.

Требования к опыту:

- от 2 лет практического опыта в разработке видеоигр или трехмерных интерактивных приложений с использованием Unity;
- от 2 лет практического опыта программирования на C#;
- участие в полном цикле разработки ПО, от концепции до готового продукта;
- понимание роли профессиональных приложений для Unity-разработки (разработки игр и приложений для интерактивных развлечений, а также визуализации дизайна);
- понимание основ рабочего процесса по созданию графических/трехмерных ресурсов и анимации в Unity, включая создание персонажей и окружения;
- понимание процессов профессиональной разработки ПО в команде, включая unit-тестирование и контроль версий;
- знание сервисов Unity, предназначенных для совместной работы, монетизации, работы с выпущенным продуктом и создания многопользовательского режима;
- понимание ключевых математических действий, используемых в разработке трехмерного интерактивного контента, включая линейные алгебраические и матричные вычисления.

Примечание: программа сертификации разработана для Unity 2017.3.

Ключевые навыки

Ключевые навыки в данной области направлены на совместную реализацию проекта от концепции до выпуска, а также на поддержку.

Программирование ключевых взаимодействий

- Реализация и настройка поведения и физики игрового объекта.
- Реализация и настройка управления и ввода.
- Реализация и настройка режимов камеры и ее движения.

Работа в рамках процессов разработки графики

- Понимание сущности материалов, текстур и шейдеров, написание скриптов, взаимодействующих с API рендеринга Unity.
- Понимание сущности освещения, написание скриптов, взаимодействующих с API освещения Unity.
- Понимание сущности двумерной и трехмерной анимации, написание скриптов, взаимодействующих с API анимации Unity.
- Понимание сущности систем частиц и эффектов, написание скриптов, взаимодействующих с API систем частиц Unity.

Разработка систем приложения

- Чтение скриптов, отвечающих за работу интерфейса приложения, например, скриптов меню, навигации по интерфейсу и настроек приложения.
- Чтение скриптов, отвечающих за работу пользовательских настроек, например, скриптов создания персонажей, снаряжения, магазинов, встроенных покупок.
- Анализ скриптов, отвечающих за прогресс пользователя в игре, например, скриптов ведения счета, присвоения уровней, а также скриптов внутриигровой экономики, использующих такие технологии как Unity Analytics и PlayerPrefs.
- Анализ скриптов для двумерных элементов, накладываемых на экран, например, HUD, мини-карты и рекламы.
- Определение скриптов, отвечающих за чтение и запись данных приложения и пользователя.
- Распознавание и оценка влияния сетевых и многопользовательских функций.

Программирование сцен и окружения

- Определение скриптов для работы со звуковыми ресурсами.
- Определение методов для создания, удаления и управления копиями объектов GameObject.
- Определение скриптов поиска пути с помощью системы навигации Unity.

Оптимизация и мультиплатформенность

- Оценка ошибок и проблем производительности с помощью соответствующих инструментов, например, профайлера Unity.
- Определение методов оптимизации под определенные платформы и/или конфигурации оборудования.
- Определение возможностей и методов оптимизации пользовательского интерфейса для XR-платформ.

Профессиональная командная работа

- Знание понятий, связанных с системами управления версиями, использование соответствующих технологий, например, Unity Collaborate.
- Понимание тестирования и его влияния на процесс разработки, включая знания профайлера Unity и традиционных методов отладки и тестирования.
- Знание методов структурирования скриптов для обеспечения модульности, читаемости и пригодности для многократного использования.

Темы сертификацион-ного экзамена

Программирование ключевых взаимодействий

- Реализация поведения игровых объектов и их взаимодействия как друг с другом, так и с окружением.
- Методы реализации управления и ввода.
- Методы реализации режимов вида камеры и ее движения.

Работа в рамках процессов разработки графики

- Знание материалов, текстур и шейдеров: API рендеринга Unity.
- Знания в области освещения: API освещения Unity.
- Знания в области двумерной и трехмерной анимации: API анимации Unity.
- Знания в области систем частиц: API частиц Unity.

Разработка систем приложения

- Работа интерфейса приложения, например, скриптов меню, навигации по интерфейсу и настроек приложения.
- Работа пользовательских настроек, например, скриптов создания персонажей, снаряжения, магазинов, встроенных покупок.
- Реализация прогресса пользователя в игре, например, скриптов ведения счета, присвоения уровней, а также скриптов внутриигровой экономики, использующих такие инструменты как Unity Analytics.
- Реализация двумерных элементов, накладываемых на экран, например, HUD, мини-карты и рекламы.
- Чтение и запись данных приложения и пользователя.
- Понимание значения и влияния сетевых и многопользовательских функций.

Программирование сцен и окружения

- Определение скриптов для работы со звуковыми ресурсами.
- Определение методов для создания, удаления и управления копиями объектов GameObject.
- Определение скриптов поиска пути с помощью системы навигации Unity.

Оптимизация и мультиплатформенность

- Оценка ошибок и проблем производительности с помощью соответствующих инструментов, например, профайлера Unity.
- Определение методов оптимизации под определенные платформы и/или конфигурации оборудования.
- Определение возможностей и методов оптимизации пользовательского интерфейса для XR-платформ.

Работа в составе команд разработчиков

- Управление версиями: влияние и использование инструментов, например, Unity Collaborate.
- Тестирование и его влияние на процесс разработки ПО.
- Распознавание методов структурирования скриптов для обеспечения модульности, читаемости и пригодности для многократного использования.

Примеры вопросов

Вопрос 1

Перед программистом стоит задача реализовать систему меню пользовательского интерфейса. Каждое меню состоит из объекта UI Panel, а также одного или нескольких объектов UI Button. Все эти объекты вложены в объект UI Canvas. Система меню пользовательского интерфейса будет создана в отдельной сцене, которая в игре будет загружаться отдельно.

Все панели и кнопки должны быть оформлены в одном стиле (цвет, текстура, тип перехода для кнопки и пр.), но арт-директор еще не приняла окончательное решение по этому вопросу. Она планирует работать над ним параллельно с программистом, разрабатывающим интерфейс. Вносимые изменения должны отражаться на всех вновь создаваемых и существующих объектах в сцене.

Как наиболее оптимально создать систему меню, чтобы арт-директор могла работать над внешним видом и дизайном меню самостоятельно, но параллельно с программистом?

A Создать подклассы для `UI.Button` и `UI.Panel`, а значения, определяющие внешний вид, задавать программно.

B Создать новые материалы для кнопок и панели и назначить их всем кнопкам и панелям в сцене.

C Создать префабы для кнопок и панели, над которыми будет работать арт-директор.

D Написать скрипт, который будет проводить поиск по заданному арт-директором тексту и заменять значения в файле сцены.

Вопрос 2

Герой трехмерного бесконечного раннера движется по параллельным путям сортировочной железнодорожной станции. Герой все время движется вперед, его задача — уворачиваться от приближающихся поездов, перепрыгивая их или уходя на соседние пути.
Каждый новый поезд добавляется на путь за всеми остальными. Из-за разницы в скорости движения поездов в сторону персонажа (которая может быть равна и нулю), поезда иногда накладываются друг на друга, что необходимо исправить.

Какой из способов наиболее эффективен для предотвращения наложения поездов друг на друга на одном пути?

A При создании поезда определять место его появления, которое не приведет к наложению, учитывая скорости движения нового и предыдущего поезда на данном пути. Также определять точку, в которой поезд, мимо которого прошел герой, будет исчезать.

B Во время движения поезда строить луч от его лобовой части. Каждый поезд, пересекшийся с лучом, перемещать вперед, увеличивая скорость его движения.

C При создании поезда добавлять к нему компонент Rigidbody, а затем использовать силы для передвижения поездов.

D При создании поезда на пути использовать компонент BoxCast с длиной, пропорциональной скорости поезда, чтобы предотвратить столкновение нового поезда с предыдущим до того, как поезда уйдут за кадр.

Вопрос 3

Программист работает над темной жуткой комнатой, и перед ним стоит задача создать горящий факел, отбрасывающий мерцающие призрачные тени на стены, пол и потолок. Программист реализовал следующие функции в `MonoBehaviour`, добавленного к факелу:

```
void Start()
{
    Light light = GetComponent<Light>();
    light.lightMapBakeType = LightMapBakeType.Mixed;
    light.type = LightType.Area;
    light.shadows = LightShadows.Soft;
    light.range = 5f;
}
void Update()
{
    GetComponent<Light>().intensity = Mathf.PerlinNoise(Time.time, 0);
}
```

При запуске факел не отбрасывает теней и не излучает света. В редакторе Unity источнику света заданы стандартные значения.

Что следует изменить, чтобы факел работал правильно?

A Переменной `light.lightBakeType` присвоить значение `LightmapBakeType.Realtime`.

B Переменной `light.range` присвоить значение `10`.

C Переменной `light.shadows` присвоить значение `LightShadows.Hard`.

D Переменной `light.type` присвоить значение `LightType.Point`.

Вопрос 4

Программист работает над симулятором шахтера, в котором игрок может копать землю в поисках минералов. В одном из мест пользователь может создать туннель, пересекающий существующую систему пещер. В дизайн-документе указано, что в существующих и новых тоннелях у звука должен быть эффект реверберации. Перед программистом стоит задача обеспечить нахождение персонажа пользователя в ближайшей области ReverbZone в любой момент времени.

Что нужно сделать с AudioReverbZone, чтобы выполнить указанное выше требование?

A Увеличить значение параметра Reflections в соответствии с размерами новой области.

B Увеличить значения maxDistance обеих зон ReverbZone, чтобы они соприкасались с новой смежной областью.

C Усилить реверберацию в соответствии с размерами новой области.

D Увеличить значение decayTime у новой области.

Вопрос 5

Во время написания функции загрузки программист получил ошибку компиляции:

```
error CS1624: The body of `CustomAnalytics.LevelLoading()' cannot be
an iterator block because `void' is not an iterator interface type
```

```
void LevelLoading(){
    AsyncOperation async = SceneManager.LoadSceneAsync("Level_01");
    while (!async.isDone)
    {
        yield return null;
    }
}
```

Что следует предпринять для исправления ошибки?

A Заменить `yield return null` на `yield return WaitForSeconds(0)`.

B Заменить `void LevelLoading()` на `IEnumerator LevelLoading`.

C Заменить `SceneManager.LoadSceneAsync("Level_01")` на `Application.LoadLevelAdditiveAsync("Level_01")`.

D Заменить `while (!async.isDone)` на `while (!async.allowSceneActivation)`.

Вопрос 6

Система ввода автосимулятора предусматривает использование горизонтальной оси для рулевого управления. При тестировании обнаружилось, что с некоторыми моделями контроллеров руль поворачивается даже при положении джойстика в центре.

Какие изменения следует внести в систему ввода для устранения данной неполадки?

A Увеличить показатель Gravity.

B Задать параметру Snap значение true.

C Увеличить значение Deadzone.

D Уменьшить значение Sensitivity.

Вопрос 7

Действие приключенческой игры происходит на чужой планете, где перед игроком стоит задача уничтожать различные формы жизни. С каждым убийством счет игрока увеличивается. Дизайн-документ предусматривает привязку счета к учетной записи пользователя, чтобы в новом сеансе или при игре на другом устройстве счет сохранялся.

Какой из способов является оптимальным для сохранения счета?

A Использовать `DontDestroyOnLoad()` для объекта GameObject, хранящего данные счета и загружать данные на сервер перед выходом из приложения.

B Сохранять счет в `PlayerPrefs` при каждом изменении счета и загружать его на сервер перед выходом из приложения.

C Использовать постоянное значение для хранения данных счета, чтобы его можно было восстановить в следующем сеансе.

D Сериализовать данные для постоянной синхронизации счета с сервером.

Правильные ответы: C, A, D, B, B, C, D.