Колледж Автономной некоммерческой образовательной организации высшего образования

«Научно-технологический университет «Сириус»

**Доклад по дисциплине “Введение в специальность”**

***«Разработка Unity»***

Выполнил:

студент 1 курса, группы 1.9.7.3

Лебедев Михаил

Проверил:

Старший преподаватель

Тенигин А. А.

г. Сочи, 2022

Оглавление

[1 Что такое Unity? 3](#_Toc122040276)

[2 Архитектура 4](#_Toc122040277)

[2.1 Инверсия управления 4](#_Toc122040278)

[2.2 Модель+Представление+Контроллер 5](#_Toc122040279)

[2.3 Передача сообщений 5](#_Toc122040280)

[2.4 Модульное тестирование 6](#_Toc122040281)

[3 Интерфейс Программы 7](#_Toc122040282)

[4 Примеры использования Unity 8](#_Toc122040283)

[5 Характеристики Unity 9](#_Toc122040284)

[6 Концепции использование Unity программистами. 10](#_Toc122040285)

[6.1 Сцены 10](#_Toc122040286)

[6.2 Игровые объекты 10](#_Toc122040287)

[6.3 Компоненты 10](#_Toc122040288)

[6.4 Активы 11](#_Toc122040289)

[6.5 Вложенные сборные блоки 11](#_Toc122040290)

[6.6 Серелизация и десериализация 11](#_Toc122040291)

[7 Преимущества и недостатки Unity 13](#_Toc122040292)

# Что такое Unity?

Unity – это современное платформенное ядро, образующее систему для разработки двух и трехмерных игр или различных сцен в приложениях, разработанный компанией Unity Techinologies упоминаются не только игры, так как есть обучающие симуляторы, приложения экстренного реагирования и приложения ориентированные на бизнес, которые разрабатывались с помощью Unity.

Unity позволяет оперировать позволяет оперировать не только кодом, но и визуальными компонентами с последующими их экспортом на любую распространённую платформу. В Unity проделана потрясающая работа, благодаря которой обеспечивается кроссплатформенная поддержка.

Unity 1 был представлен на сцене Всемирной конференции разработчиков Apple в 2005 году Дэвидом Хелгасоном, Николасом Фрэнсисом и Иоахимом Анте.[1]

Созданные с помощью Unity приложения работают под операционными системами Windows, OS X, Windows, Phone, Android, Apple, iOS, Linux, а также на игровых приставках Wii, PlayStation 3, PlayStation 4, Xbox 360, Xbox One и MotionParallax3D.

# Архитектура

Основными целями этой архитектуры являются:

* Поддерживаемость
* Расширяемость
* Тестируемость

Эти три цели нелегко достичь в движке, который в первую очередь нацелен на быстрое прототипирование[[1]](#footnote-1). Среди разработчиков игр распространено мнение, что эти принципы больше подходят для бизнес-решений, чем для игр, и я категорически с этим не согласен. Обращая свой взгляд на решения в этой области, можем обнаружить, что существуют полезные инструменты, которые можем применить и к играм[2]. Самыми популярными являются:

* Инверсия управления
* Интерфейс передачи сообщений
* Модель+Представление+Контроллер
* Модульное тестирование[7]

### Инверсия управления

Внедрение зависимостей — это подход к реализации инверсии управления. Внедрение зависимостей используется как строитель для генерации нашего управления, проверки необходимых зависимостей и их автоматического предоставления. Управление не заботит то, какой конкретно класс, реализующий требуемый интерфейс, используется, пока таковой имеется в наличии. Обычно для избавлении от негативного влияния рефлексий на производительность ядра, используется функция «запекания» рефлексий[2]

### Модель+Представление+Контроллер

Суть этой архитектуры — разбиение кода на отдельные уровни. Базовые скрипты Unity обитают на уровне представления, как предполагается, защищает остальную часть архитектуры от затрудняющих модульное тестирование элементов Unity. Этот уровень имеет доступ только к уровню контроллера. Представление создают серверные ресурсы и использует для использования типичных элементов захвата и перетаскивания мыши. Здесь не должно быть никакой игровой логики, только чистая визуализация данных.

Уровень контроллера содержит бизнес-логику и выполняет всю тяжелую работу. Этот код должен быть тестируемым, он не зависит от специфики уровня представления Unity. Но все же этот уровень не определяет способ хранения данных на уровне модели, он только контролирует изменения на нем[7].

Модель содержит фактические данные, ну или в каком-либо бэкенде. Обычно модели — это старые добрые, хорошо нам известные типы данных.

Поскольку представление не должно запрашивать информацию об изменении данных, для его уведомления используем передачу сообщений.

Решение о том, считывает ли представление данные прямо из модели или через контроллер, не является какой-нибудь утвержденной критикой. Единственное правило – изменения происходят только через уровень контроллера. Считывание значений может происходить прямо из модели[2].

### Передача сообщений

Вышеупомянутая архитектура полагается на соответствующих уведомлениях, чтобы уровень представления мог реагировать на события.

Важно отметить, что сигналы должны быть легковесными и не содержать данных — для этого используем остальные уровни MVC[2].

### Модульное тестирование

Благодаря всем вышеперечисленным ограничениям и механизмам теперь можем покрыть модульными тестами почти всю игровую логику.

Такого рода тесты обходятся дешево в выполнении и сохраняют целостность игровой логики, потому что прогоняем их еще до создания нового тестового билда[7].

# Интерфейс Программы

Интерфейс в Unity удобен и интуитивно понятен, начинающий пользователь способен изучить основной функционал среды и приступить к разработке за относительно небольшой временной интервал.

Главное окно редактора состоит из нескольких вкладок, называемых Видами. В Unity. есть несколько типов видов - все они предназначены для конкретных целей:

* Окно сцен используется для выделения и позиционирования элементов сцен, камеры и прочих объектов.
* Консоль обеспечивает вывод сообщений, предупреждений, ошибок.
* Окно анимацй используется для анимации объектов в сцене.
* Профайлер используется для обзора производительности и поиска «узких» мест приложения.
* Окно активного сервера, в случае его использования, дает возможность контролировать версии проекта.
* Окно текстур, используя встроенные в Unity инструменты, позволяет настраивать карты освещения.
* Окно рендеринга объектов используется для настройки параметров производительности приложения.
* Окно инспектора отображает детальную информацию о текущем выбранном объекте, включая все прикреплённые компоненты и их свойства[7].

Как правило, разработанные в Unity игры состоят из множества игровых объектов, которые содержат полигональную сетку (совокупность вершин, рёбер и граней, которые определяют форму многогранного объекта в трехмерной компьютерной графике и объёмном моделировании), скрипты, звуки или другие графические элементы, вроде источников света[4].

# Примеры использования Unity

Unity можно использовать для реализации как крупных высококачественные проекты с высоким бюджетом и соответствующим уровнем качества проектов[6], так и для небольших 2D игр под мобильные платформы, при этом поддержка 2D в Unity появилась относительно недавно. В настоящий момент на базе Unity активно создаются браузерные игры. Большинство разработчиков используют ядро Unity для моделирования виртуальных объектов, но применение Unity на этом не ограничивается [8]. Так, например, данное ядро отлично себя зарекомендовала для решения задач визуализации объектов реального мира[5].

К подобным сферам применения можно отнести виртуальные туры по музеям, памятникам и другим культурным объектам. Визуализация объектов строительства дает возможность предварительно понять и оценить объект с различных точек зрения, застройщику и потенциальному покупателю [6].

# Характеристики Unity

Для разработки в Unity интегрирован игровой движок, в котором можете протестировать свою игру не выходя из редактора.

Unity поддерживает импорт огромного количества различных форматов, что позволяет разработчику игры конструировать сами модели в более удобном приложении, а Unity использовать по прямому назначению разработки продукта.

Unity3d является актуальной платформой, с помощью которой можете создавать свои собственные приложения и экспортировать их на различные устройства.[9]

# Концепции использование Unity программистами.

Выбор главного материала, которые познакомят вас с концепциями:

* С#
* ООП

Выбор может оказаться слишком сложный, так как учебные материалы найти не очень уж и просто, для ознакомления с концепциями. Также есть расширенные руководства, в которых предоставлено дедуктивное определение основных концепций.

#### Сцены

Про сцены я уже писал ранее в пункте 2.2

#### Игровые объекты

Игровой объект является одним из основных строительных блоков игры.

Игровые объекты могут представлять как физические вещи, которые вы видите в игре, например: игрока, землю, дерево, местность, огни, оружие, пулю, взрыв, так и метафизические объекты, например: менеджер инвентаря, многопользовательский контроллер в вашей игре.

У каждого игрового объекта есть положение и поворот. Для метафизических объектов это не имеет значения.

Игровые объекты могут располагаться друг под другом. Положение и поворот каждого объекта относительно его родительского объекта. Объект непосредственно в сцене относительно координат «мирового пространства».

#### Компоненты

Эту тему я уже затрагивал ранее в пункте 3. Но сейчас углублюсь в нее глубже.

Каждый игровой объект состоит из компонентов.

Компонент реализует четко определенный набор поведения для выполнения игрового объекта. Все, что делает объект тем, чем он является, будет происходить из компонентов.

Хотя вы можете написать большие и сложные Компоненты, которые соответствуют 1:1 объекту, обычно логику разбивают на упрощенные части.

#### Активы

Активы - это ресурсы на диске, из которых состоит ваш игровой проект. К ним относятся сетки (модели), текстуры, спрайты, звуки и другие ресурсы.

При сериализации[[2]](#footnote-2) на диск ваши сцены представляются как активы, состоящие из игровых объектов внутри них.

Активы также могут представлять менее осязаемые вещи, такие как карты управления вводом, настройки графики, строковые базы данных

#### Вложенные сборные блоки

Игровые объекты, их компоненты и их входные параметры существуют как отдельные экземпляры в сцене. Такие объекты можно превратить в префаб[[3]](#footnote-3), который фактически является объектом в форме актива.

Экземпляры префаба в сцене могут иметь локальные модификации, которые отличают его (например, если объект дерева является префабом, у вас могут быть экземпляры дерева разной высоты). Все экземпляры префаба наследуют и переопределяют данные из своих префабов.

#### Серелизация и десериализация

Ресурсы, сцены и объекты вашего проекта сохраняются на диске. При редактировании игры эти объекты загружаются в память и сохраняются обратно на диск с помощью системы сериализации Unity. При тестировании игры объекты и сцены в памяти загружаются через одну и ту же систему сериализации. Эта система также сопоставляет ресурсы в вашем скомпилированном пакете с загруженными или выгруженными объектами сцены в памяти. Поток сериализации или десериализации движок Unity загружает ресурсы с диска в память и отвечает за сохранение состояния ваших отредактированных объектов и компонентов обратно в их сцены.

# Преимущества и недостатки Unity

Преимущества[10]:

* Функциональный графический редактор – предусматривает создание локаций, моделей, расстановку объектов сцены с функцией тестирования результатов в реальном времени
* Кроссплатформенная среда разработки позволяет адаптировать проект для любой платформы путём внесения минимальных изменений. Минимизирует стоимость проекта
* Интегрированная среда разработки – программный комплекс для работы с движком, имея базовые знаниями программирования
* Поддержка плагинов – возможность установки компонентов, расширяющих возможности, повышающих удобство, скорость работы со средой
* Модульность – позволяет конструировать пакеты компонентов в одной игровой сцене
* Продвинутая методика создания объектов: вместо наследования путём размещения в древе они объединяются в функциональные блоки, что упрощает этап прототипирования
* Использование высокоуровневого языка C# с рядом готовых решений упрощает подключение программистов к разработке
* Качество поддержки – Unity сайт содержит описание всех функций движка, примеры их применения с пояснениями. Служба поддержки оперативно отвечает на вопросы
* Бесплатная эксплуатация для применения в личных и образовательных целях
* Сообщество – продвинутые пользователи всегда подскажут, помогут советом. На тематических форумах и блогах полно информации по Unity

Недостатки[11]:

* Закрытость кода. Невозможность получения исходных кодов движка даже по лицензии
* Невозможность дополнения физики движка сторонними возможностями.

Список использованных источников:

1. Представление Unity // Unity URL: https://unity.com/ru/how-to/how-architect-code-your-project-scales (дата обращения: 13.12.2022).
2. Архитектура // Habr URL: https://habr.com/ru/company/otus/blog/530056/ (дата обращения: 13.12.2022).
3. Прототипирование // Википедия URL: https://goo.su/ybKCh (дата обращения: 13.12.2022).
4. Интерфейс программы Unity // Docs.unity3d URL: https://docs.unity3d.com/Manual/UIB-interface-overview.html (дата обращения: 13.12.2022).
5. Unity3D - видеоуроки, уроки, статьи и документация к игровому движку // Unity3dForge URL: http://unity3dforge.com/ (дата обращения: 14.12.2022).
6. Полезный инструмент для ускорения процесса разработки приложений // HabraHabr URL: http://habrahabr.ru/company/ximad/blog/252525/ (дата обращения: 14.12.2022).
7. Unity - Руководство // Docs.unity3d URL: http://docs.unity3d.com/ru/current/Manual/UnityMan ualRestructured.html (дата обращения: 14.12.2022).
8. Unity3D для реальной реальности // Pvsm URL: http://www.pvsm.ru/unity3d/30451 (дата обращения: 15.12.2022).
9. Характеристики Unity // Unity URL: https://unity.com/ru/products/unity-platform (дата обращения: 15.12.2022).
10. Преимущества Unity // Instant URL: https://itstan.ru/programmirovanie/igrovoy-dvizhok-unity-pochemu-ego-vybirayut.html (дата обращения: 15.12.2022).
11. Недостатки Unity // Gamesmaker URL: https://gamesmaker.ru/3d-game-engines/unity3d/osobennosti-besplatnogo-krossplatformennogo-dvizhka-unity3d/ (дата обращения: 15.12.2022).

1. Прототипирование — быстрая «черновая» реализация базовой функциональности будущего продукта/изделия, для анализа работы системы в целом[3] [↑](#footnote-ref-1)
2. Эта тема рассмотрена в пункте [↑](#footnote-ref-2)
3. Префаб – это шаблон для объекта в игровом движке [↑](#footnote-ref-3)