легчает разработку, уменьшает количество ошибок и помогает лучше распределить ресурсы проекта.

Внедрение прототипа в дизайн позволяет более точно воплотить в жизнь то чего именно хочет от нас заказчик. Простота создания, оперативность внесения изменений, интерактивность, эстетичность внешнего вида и даже частичное тестирование юзабилити – основные преимущества создания прототипа. При разработке прототипа заказчик видит пример и соглашается или не соглашается с направлением и дизайном, которые будут реализованы на его проекте, или указывает какие-то замечания или пожелания. Такая система активного использования прототипа и постоянного общения с заказчиком позволяет значительно уменьшить число конфликтных ситуаций, сократить число затраченных ресурсов и времени. Таким образом можно сделать вывод о том, что прототипирование в дизайне это отличное средство сэкономить ресурсы, время, а добиться хорошего результата при разработке проекта.

Из всего вышеперечисленного можно сделать вывод о том, что прототипирование необходимо сегодня для успешной реализации проекта. Это выгодное средство достижения лучшего и качественного результата.

УДК 004.921

Студ. А.А. Гребенчук
Науч. рук. ассист. И. А. Миронов
(кафедра информатики и веб-дизайна, БГТУ)

СОЗДАНИЕ ИНТЕРАКТИВНЫХ 3D МОДЕЛЕЙ В СРЕДЕ UNITY

С развитием GameDevelopment’а требования к игровому контенту, в частности создания *3D* игр, и программам, которые их создают, увеличиваются с каждым месяцем, задавая более высокие темпы развития игровой индустрии. На данный момент существует много платформ для создания различного игрового, обучающего контента, но некоторые из них не доступны рядовому пользователю, так как предоставляются только в платном варианте, а некоторые сложны для пользователя не знакомого с программированием. Но существуют и исключения, такие как Unity *3D*, Torque *2D/3D*, CryEngine 3, UDK (Unreal Development Kit).

Наш выбор пал на Unity *3D* из-за некоторых его достоинств по сравнению с другими платформами:

– Unity – бесплатный кроссплатформенный 3D-движок;

– собственная IDE, котоаяе включает в себя редактор сцен, редактор игровых объектов, редактор скриптов, deferred освещение, встроенный редактор шейдеров, стандартный набор постпроцессинговых эффектов, генератор деревьев и террейнов;

– возможности для скриптинга – в отличие от UDK, в котором писать можно только на встроенном языке, в Unity доступны три языка: JavaScript, C#, и диалект Python под названием Boo;

– кроссплатформенность – как уже упоминалось выше, поддерживаются Windows, Mac OS, Wii, iPhone, iPod, iPad, Android, PS3 и XBox 360;

– производительность и масштабируемость – с большей частью рутины движок справляется сам, и справляется замечательно, примером будет служить система Level Of Detail (сокр. LOD), суть которой заключается в том, что на дальнем расстоянии от игрока высоко детализированные модели заменяются на менее детализированные, и наоборот, а также систему Occlusion culling, суть которой в том, что у объектов, не попадающих в поле зрения камеры не визуализируется геометрия и коллизия, что снижает нагрузку на центральный процессор и позволяет оптимизировать проект.

Так же стоит отметить, что Unity является интуитивно понятной средой для разработки, так что работать и обучаться в ней будет не так тяжело.

Создание *3D* моделей в Unity начинается с создания примитивов. Примитивами в Unity называются элементарные объекты, которые создаются непосредственно в Unity, такие как: *куб*, *цилиндр*, *сфера*, *плоскость*, *квад*, *капсула*. Они используются для того, чтобы задать основу модели (к примеру создание земли), но также позволяют быстро создать прототипы сложных объектов в целях тестирования.

После создания примитивов, на них используются различные компоненты для обработки физики, называющиеся *collider*.

Чтобы физическое поведение было правдоподобным, объект в игре нужно правильно ускорить и задействовать столкновения, гравитацию и другие силы. Встроенные в Unity физические движки обеспечивают предоставляют компоненты для обработки физики. С помощью настройки всего нескольких параметров, можно создать объекты, которые ведут себя пассивно реалистично (т.е., они будут перемещены в результате столкновений и падений, но не начнут двигаться сами по себе). Управляя физикой из скриптов, можно придать объекту динамику автомобиля, механизма или даже подвижного куска ткани.

Коллайдер, в зависимости от его назначения и свойств, может принимать разные формы, такие как:

– *Boxcollider* – базовый кубический примитив столкновений.

– *Capsule* –состоит из двух полусфер, соединённых между собой цилиндром. Это такая же форма, как и у примитива капсула (*Capsule*).

– *Meshcollider* – использует меш ассет для создания под него коллайдера на основе этого меша. Это гораздо удобнее, чем использовать примитивы для определения столкновений сложных мешей. Меш коллайдеры, которые помечены как *Convex* могут взаимодействовать (сталкиваться) с другими меш коллайдерами.

– *Wheelcollider* – это специальный вид коллайдера для наземного транспорта. В него встроены: определение столкновений, физика колеса и основанная на скольжении шин модель трения. Его можно использовать и не только для колёс, но он был специально создан для транспорта с колёсами.

– *Physics Material* – используется для настройки эффектов трения и отскакивания объектов при столкновениях.

– *Rigidbody* (Твердое тело) – твёрдые тела позволяют игровым объектам взаимодействовать с помощью физики. Для реалистичного перемещения твёрдых тел, на последние воздействуют сила вращения и другие силы. Любой игровой объект должен содержать в себе твёрдое тело, чтобы быть подверженным гравитации, действовать согласно назначенным путём скриптинга силам, или взаимодействовать с другими объектами через физический движок *NVIDIAPhysX*.

– *Charactercontroller* –компонент в основном используется для управления от третьего или первого лица, где не требуется физика *Rigidbody*.

За примитивами и коллайдерами следуют *GameObject* – постоянные объекты на сцене Unity, которые играют важную роль в конструкции мира:

– системы частиц – имитируют жидкие субстанции наподобие разных жидкостей, облаков и чего-нибудь, связанного с огнём, путём генерации и анимации в сцене;

– камеры – являются устройствами, которые захватывают и отображают мир игроку. Путем настройки и манипулирования камерами, можно сделать презентацию своей игры поистине уникальной. В проекте возможно использовать неограниченное количество камер в сцене, настраивать рендеринг камерами в любом порядке, на любом месте экрана, либо только в определенных частях экрана;

– *GUI* текст – отображает текст любого шрифта, который размещается в координатах экрана;

– *GUI* текстура – файлы изображений или видео, которые накладываются на объект или оборачивают вокруг. Т.к. они очень важны, у них есть множество свойств;

– *3D* текст – генерирует *3D* геометрию, которая представляет из себя строки текста;

– точечный свет, направленный свет, освещение территории, источник света, имитирующий солнце – источники света определяющие цвет и атмосферу *3D* окружения проекта;

– деревья – полноценные *3D* объекты, которые растут из поверхности;

– *Terrain*(земля) – стандартная поверхность, имеющая обильное количество настроек.

Выше были перечислены основные компоненты, которые используются при создании непосредственно в среде Unity, остальные, более сложные объекты, возможно создать из примитивов, сильно изменяя их настройки, либо собственно созданную в другой программе *3D* моделирования.

Подводя итоги исследования среды Unity следует отметить, что данная программная среда является подходящей не только для начинающего пользователя но и для опытного разработчика при создании различного уровня сложности интерактивных *3D*моделей.

УДК 004.78:025.4.036

Студ. А. М. Колодкевич
Науч. рук. доц. А.А. Дятко
(кафедра информатики и веб-дизайна, БГТУ)

КЛИЕНТ – СЕРВЕРНОЕ ПРИЛОЖЕНИЕ ДЛЯ УДАЛЕННОГО ТЕСТИРОВАНИЯ ЗНАНИЙ

Тесты – это краткие и ограниченные во времени испытания, предназначенные для определения уровня знаний человека.

На сегодняшний момент времени тестирование является одной из самых наипопулярнейших форм для контроля знаний. Так с развитием информационных технологий стали популярны и компьютерные тесты, ведь применение программного обеспечения для создания и проверки тестов позволяет значительно облегчить процесс контроля знаний. В настоящее время это особенно актуально в связи с популярностью различных форм дистанционного обучения, в частности, массовых открытых онлайн курсов. К плюсам систем компьютерного тестирования можно отнести:

– возможность массовой оценки знаний;
– легкость создания и управления тестами;
– простота назначения тестов;