Учреждение образования
«Белорусский государственный университет культуры и искусств»

Факультет Культурологии и социокультурной деятельности
Кафедра Информационных технологий в культуре

СОГЛАСОВАНО
Заведующий кафедрой
_____
«___»_____20___ г.

СОГЛАСОВАНО
Декан факультета
_____
«___» _____20___ г.

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС
ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

ИГРОВАЯ КУЛЬТУРА И ДИЗАЙН

для специальности 1–21 04 01 Культурология,
направление специальности 1–21 04 01–02 Культурология (прикладная),
специализации 1–21 04 01–02 04 Информационные системы в культуре

Составитель:
О.М. Кунцевич, старший преподаватель кафедры информационных технологий в культуре учреждения образования «Белорусский государственный университет культуры и искусств»

Рассмотрено и утверждено
на заседании Совета университета
(протокол № 7 от 17.03.2020)

Составитель:
Кунцевич Ольга Михайловна, преподаватель кафедры информационных технологий в культуре учреждения образования «Белорусский государственный университет культуры и искусств»

Рецензенты:

Е.А. Криштаносова, доцент кафедры межкультурных коммуникаций учреждения образования «Белорусский государственный университет культуры и искусств», кандидат культурологии, доцент

В.С. Романчик, профессор кафедры веб-технологий и компьютерного моделирования БГУ, кандидат физико-математических наук, доцент

Рассмотрен и рекомендован к утверждению:
Кафедрой информационные технологии в культуре
(протокол от 21.11.2019 г. №4)

Советом факультета культурологии и социокультурной деятельности
(протокол от «___»_____ 2019 г. № __)

СОДЕРЖАНИЕ

1. ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Сегодня понятие игровой культуры перестало принадлежать только сфере детских игровых форм досуга, став всеобъемлющим в области любых контактов человека разного возраста и социального статуса. Игровая культура становится самостоятельным структурным компонентом культурной деятельности человека. Внедрение игровых технологий и игровых элементов в настоящее время можно проследить в различных сферах человеческой деятельности.

Дисциплина «Игровая культура и дизайн» позволит раскрыть возможности в создании и реализации игрового проекта в сфере культуры. Пользователи игровых технологий должны обладать знаниями в области игровой культуры как объекта культурологического иследования, индустрии компьютерных игр, развитию игровых платформ, основ процессов разработки и реализации идеи, подготовка дизайн-концепта, создание 2-D и 3-D объектов, знание основ программирования, что становится необходимым для менеджера-культуролога и специалиста информационных технологий в культуре. В процессе обучения учитывается уровень владения компьютерными технологиями.

Целью учебно-методического комплекса по дисциплине «Игровая культура и дизайн» является предоставить студенту комплект учебно-методических материалов для изучения дисциплины, способствовать усвоению в полном объеме учебного материала дисциплины через систематизацию, планирование и контроль собственной деятельности, дать рекомендации по подготовке к текущей и итоговой аттестации.

Учебно-методический комплекс включает следующие разделы: пояснительную записку, теоретический, практический, контроля знаний, вспомогательный. Теоретический раздел учебно-методического комплекса содержит материалы, которые знакомят студентов с тематикой лекционных занятий и текстами лекций. Практический раздел содержит тематику лабораторных работ, задания и рекомендации по их выполнению. Раздел контроля знаний включает в себя перечень требований к зачету, критерии оценки результатов учебной деятельности, задания для контролируемой самостоятельной работы студентов, контрольные вопросы по темам, перечень вопросов к зачету. Вспомогательный раздел учебно-методического комплекса содержит учебную программу, учебно-методические каты для дневной и заочной формы обучения, список основной и дополнительной литературы, учебный терминологический словарь.

2. ТЕОРЕТИЧЕСКИЙ РАЗДЕЛ

2.1 Тематика лекционных занятий

Лекция 1. Введение. Игровая культура как социокультурное явление. Дизайн-концепт в разработке компьютерных игр

Лекция 2. Игровая индустрия. Игровая культура в условиях компьютеризации

2.2 Конспект лекций

Лекция 1

Введение. Игровая культура как социокультурное явление. Дизайн-концепт в разработке компьютерных игр.

Курс лекций

Основные вопросы

1. Роль дисциплины «Игровая культура и дизайн» в подготовке специалиста высшей квалификации.
2. Связь с другими дисциплинами специализации.
3. Игровая культура как объект культурологического исследования.
4. Сущность и особенности игровой культуры.
5. Понятие игры в контексте современных научных исследований.
6. Первичная и вторичная игровая культура.
7. Культурологические концепции игровой культуры.
8. Первичный концепт-документ: краткое описание игрового процесса и особенности игры.
9. Создание дизайн-документ («диздок»).
10. Концепция и художественный стиль игры.
11. Дизайн уровней, создание игрового ландшафта.
12. Моделировние объектов (2-D и 3-D графика).

Цель. Расмотреть игровую культуру как социокультурное явление и понятие игры в контексте современных научных исследований. Изучить дизайн-концепт в разработке компьютерных игр.

Введение.

Понятие игровой культуры часто рассматривается как в сфере прикладной культурологии, так и исследователями фундаментальных направлений изучения истории и теории культуры. Данное понятие может рассматриваться с одной стороны, как основная характеристика любого

проявления культуры в целом, так и с другой стороны как явления культуры в частности на протяжении всего развития человечества.

В последнее время игровая культура, пройдя стадии становления понятия, расширения общего смыслового поля через осмысление в различных областях знаний, стала объектом культурологического анализа. Выделение исследователями чувственного компонента в «генетике культуры» как основного фактора обеспечения традиции, а значит, самого способа существования культуры, привело к признанию игрового фактора не только в качестве культуроформирующего, но и культурообразующего. Исследование формирования игровой культуры, в связи с этим, представляется не только как самоценность, но и как важная составная часть изучения культуры как феномена человеческого бытия.

Долгое время игровая культура считалась областью досуговой сферы жизнедеятельности человека и определялась стремлением к заполнению свободного времени, не связываясь с другими видами человеческой деятельности. Сегодня игровая культура всё больше признаётся исследователями как один из основных и характерных факторов человеческого существования, что придаёт исследованиям в данной области особое значение. Результаты исследований формирования игровой культуры могут, таким образом, использоваться в различных областях изучения культуры вообще.

Содержание учебной дисциплины «Игровая культура и дизайн» межпредметно связана с учебными дисциплинами «Веб-дизайн и реклама», «Информационные технологии в культуре», «Компьютерная графика», «Графический дизайн», «Художественное проектирование», «Трехмерное моделирование и анимация».

Игровая культура как социокультурное явление.

В советской науке традиционно проявления игровой культуры изучались в процессе педагогической деятельности. Такой подход во многом способствовал пониманию игровой культуры как важного фактора формирования и развития узкоспециальных личностных образований у детей. Однако ценностно-смысловое значение игровой деятельности человека оставалось не выявленным. Происхождение игр связывается разными исследователями с магико-культовыми ритуалами или с врожденными биологическими потребностями организма и психики человека.

Сегодня сформировалось несколько научных подходов к исследованию игры, в рамках которых также существуют отдельные самостоятельные направления. Феномен игры как определяющего начала (концепта) игровой

культуры представлен в современных исследованияхс различных аспектов и фрагментарно. Отдельные свойства и проявления игровой культуры изучаются, по преимуществу, представителями современной педагогической науки Я. Ю. Манусовой, В.Д. Пономаревым, Е. А. Репринцевой, В. Я. Суртаевым, Л.И. Козловской. Также педагогический аспект, рассматривающий роль игры в воспитании личности, отражен в работах Н.П. Аникеевой, Л.С. Выготского, С.Ф. Занько, С.А. Макаренко, И.Г. Песталоцци, С.Л. Рубинштейна, К.Д.Ушинского. Игра в культурологическом подходе понимается как способ действия и реализации сущностных сил человека, а также как одна из форм существования культуры в целом. Психологический подход связан с пониманием игры, как одного из компонентов человеческой психики. К биологическому подходу исследования игры относятся теории избытка сил В. Вундт и Ч. Спенсера; теории упражнения К. Гросса, В. Штерна, К. Бюллера; общая психоаналитическая теория игры Ф. Бейтендайка, теория восстановления сил М. Лацаруса, в которых игра предстает как объект анализа на различных этапах фило – и онтогенеза. Социологический взгляд на феномен игры обусловлен его функционированием в общественной жизни. Отдельное место в осмыслении феномена игры занимают художественные источники в образной форме представляющие особенности игровой деятельности и характеристики ее субъектов-игроков. В этом контексте исследовательский интерес для нас представляют произведения Г.Гессе, Н. В. Гоголя, Э. Т. А. Гофмана, Ф. М. Достоевского, А. С. Пушкина, С.Цвейга [2].

Интерес к игровой проблематике впервые оформился в философии. «Игра» как самостоятельная категория в системе научных знаний стала признаваться со 2-ой половины XX века в основном в системе философского знания, как феномен культуры. Между тем на неутилитарный, эстетический и сущностно значимый для человеческой жизни характер игры философская мысль обратила свое внимание еще в древности. Так, представления об игре как системе представлений о вечно становящимся мире, присутствуют у Гераклита, уподоблявшего эон «играющему дитя», создающего в игре своеобразную модель Вселенной. Долгое время сохранялось представление о множественности возможных путей развития становящегося мира.

Мысль о том, что мир управляется высшей разумной волей, имеющей сверхъестественное происхождение, формирует представление об игре как форме проявленности высших сил, предписанной свыше. Наиболее отчётливо эта позиция представлена у Платона. У Аристотеля уже явно присутствует тенденция рассмотрения игры в её прикладных возможностях рациональной организации свободного времени человека, способствующей его воспитанию.

Дальнейшее философско-теоретическое осмысление игровой деятельности связано с развитием классической немецкой философии, прежде всего с именем И. Канта. Немецкая классическая философия выдвинула на первый план эстетический аспект игры. В работах И.Канта на первый план выдвигается эстетический аспект игры, в контексте которого впервые появляется постановка проблемы «искусство и игра» [9].

Неотъемлемые свойства сущности игры, как спонтанность существования, свобода, непредзаданность, случайность, не получают теоретического осмысления в рамках рационалистической философии. Существенный шаг в изменении классических представлений о природе случайного и возможного был впервые осуществлён в рамках математической теории вероятностей, предпосылки которой связаны с подсчётом вероятности в азартных играх (середина XVII века). Постепенно случайность и неопределённость из «недостатков нашего разума» превращаются в самостоятельные объекты исследования у Д. Спиноза.

Нидерландский мыслитель и историк культуры Й. Хейзинга, предпринявший фундаментальное исследование игры «Homo ludens» усматривает ее сущность в способности приводить в восторг, доставлять радость; он определяет ее как «добровольное действие либо занятие, совершаемое внутри установленных границ места и времени по добровольно принятым, но абсолютно обязательным правилам с целью, заключенной в нем самом, сопровождаемое чувством напряжения и радости, а также сознанием «иного бытия», нежели «обыденная» жизнь». Игра существует не только в человеческом, но и в животном мире и не поддается логической интерпретации. Человеческая культура возникает и развертывается в игре, носит игровой характер. Игра стала, по мнению философа, формирующим элементом человеческой культуры. Раньше, чем изменять окружающую среду, человек сделал это в собственном воображении, в сфере игры. Правильно подчеркивая символический характер игровой деятельности, Й.Хейзинга обходит главный вопрос культурогенеза [38].

Вопрос о взаимосвязи игры и поэзии является центральной темой рассуждения о связи между игрой и культурой. Для понимания поэзии нужно облечь себя душой ребенка, и мудрость ребенка поставить выше мудрости взрослого. Внешним признаком глубоко психологической связи игры и искусства является то, что во многих языках исполнение на музыкальных инструментах зовется игрой. Игра лежит вне благоразумия практической жизни, вне сферы необходимости или пользы, то же относится к музыкальным формам и музыкальному выражению. Игра стоится по законам, которые не определяются нормами разума, долга и истины. То же справедливо и для музыки. Действенность ее форм и ее функции

определяется нормами, которые никак не соприкасаются ни с логическими понятиями, ни со зрительными или осязаемыми образами.

Роже Кайуа считает, что игра есть форма активности, свободной, изолированной, нечеткой (правила игры предоставляют определенную свободу действия), непродуктивной (в процессе игры не создается никаких новых материальных благ), регламентированной, т.е. протекающей по определенным правилам и фиктивной, т.е. сопровождающейся особым сознанием иной реальности. Игра позволяет создавать параллельную реальность вне контекста обыденной жизни. Игровая деятельность в такой ситуации носит фиктивный характер [18, 34-35].

Р. Кайуа описал четыре типа игр:

– Агон (др. греч. – борьба) – тип игр, которые построены на принципе соревнования, борьбы с противником (спортивные игры, коммерческая конкуренция, система конкурсов и экзаменов).

– Алеа (др. греч. – жребий) – игры, построенные на случайности, удаче, жребии, например: рулетка и кости, карты и скачки, биржевые спекуляции.

– Мимикрия (др. греч. – подражание) – тип игр, основанных на воспроизведении разных типов человеческой деятельности: театр и балет, игры в куклы и шарады, церемониал, униформа.

– Иллинкс (др. греч. – головокружение) – интенсивное, форсированное изменение состояния сознания. Среди развлечений сюда относятся качели, карусели, гигантские шаги [18, 50].

Р. Кайуа проводил параллели в игровой деятельности и использование её в психотерапевтической практике.

В целом педагогика и психология раскрывают понятие игры и подчеркивают ее влияние на становление и развитие ребенка. Игра может выступает как своеобразный мир, в котором растет и развивается ребенок, осуществляет активную познавательную деятельность, приобретает большой объем новых знаний, впитывая в себя богатство культуры – деловые игры, спортивные игры, игра актера и т.п.

Интересно, что игра нашла своих исследователей и среди довольно серьезных направлений науки, таких как математика и логика, воплотившись в создании теории игр и моделировании. С развитием технологий и увеличением потребностью человека, игра переходит в технологический и виртуальный мир. В связи с этим появляется новое направление в игровой культуры, как компьютерная игра, которую можно представить как ключ к виртуальному миру.

Дизайн-концепт в разработке компьютерных игр.

Процесс создания игры – это сложный, многостадийный и многосоставной процесс. Разработка игры состоит из немалого количества четко разделенных между собой этапов. В основе каждой игры лежит базовая идея, которая каким-либо образом отделяет ее от похожих игр данного жанра. В противном случае, итог будет прямолинейным клоном, и это служит достаточно большим препятствием для распространения.

1 этап: Работа с идеей

Базовая идея, или иначе – концепция, – это текстово-графическое описание ключевого замысла проекта. В этот момент осуществляется определение жанровой принадлежности игры, так как на одной идее могут быть созданы игры совершенно разной направленности.

Человек, разрабатывающий правила игр – это геймдизайнер. Для работы данному специалисту необходимы навыки аналитика, психолога, технического писателя и игрока, умение работать в команде. Из дополнительных навыков – любые, применимые в разработке игр: художественный вкус, рисование, 3D-моделирование, минимальное знание математики, физики и программирования.

2 этап: Первичная документация

После формирования идеи (продуманный и структурированный набор тезисов), необходимо все зафиксировать и дополнить деталями.

Концепт-документ – это короткий документ, который относительно детально описывает вашу игру. Целевое назначение этого документа заключается в содержании понятного описания концепции игры, что в свою очередь знакомит членов команды с идеей игры.

Содержание концепт-документа должно быть следующим:

Среда разработки: Мобильная платформа, или ПК, а может быть консоль или соцсети. Выбор платформы также поможет определить среду разработки и выбрать игровой движок, что в свою очередь создаст определенные требования к программистам на проекте. Выбор платформы также определяет рекламную кампанию, и направление для маркетинговых работ.

Жанр, целевая аудитория: Выбор целевой аудитории даст представление о том, как в дальнейшем продвигать проект, о том, какую нишу он займет на рынке игр, и о некоторых игровых механиках игры.

Описание геймплея, список фич: Фича – feature – это игровая механика, особенность игры. Здесь необходимо максимально понятно описать, в чем, собственно, и состоит сам процесс игры.

Описание game loop – игрового цикла – последовательности действий, которые совершает игрок постоянно, каждую игровыю сессию.

Перечислить более подробно основные игровые механики: что еще интересного и веселого игрок сможет делать в игре. Например: прокачка персонажа, или лечение раненых войск, или крафт уникальных предметов. Насколько полным будет это описание – зависит от типа документа.

USP – Unique Selling Points (уникальные особенности игры): это уникальные фичи, которые делают вашу игру особенной и которые должны привлечь игрока, заставить его выделить ваш продукт среди конкурентов в том же жанре. В этот пункт также попадают какие-то инновационные идеи, или необычное сочетание уже известных игровых механик.

Сюжет, история мира: Общее представление о сюжете и о механики, описанные выше обоснованы сюжетно. Можно также указать ключевых персонажей вашей истории с кратким описанием их характеров. Можно упомянуть основные конфликты, интересные сюжетные ходы, чтобы дать наиболее полное представление об игровом мире.

Контент: В начале работы четко описать наполнение игры будет сложно, но в данном разделе необходимо примерно посчитать то, что нужно будет сделать. Это поможет вам понять, какого размера команда вам потребуется для разработки, и возможно протрезветь на тему того, потянете ли вы вообще масштаб, который задумали. Посчитать количество персонажей, квестов, диалогов, бэкграудов, объектов и прочего и записать списком в документ.

Стиль графики с референсами: Если вы сами не рисуете, найдите в сети референсы на то, как вы хотите, чтобы выглядела ваша игра, скриншоты других игр, работы художников, палитры цветов. Добавьте эти картинки с небольшим текстовым описанием: что самое важное на той, или иной картинке.

3 этап: Поиск команды

После формирования идеи, следует поиск команды. Наиболее продуктивный концепт для разработки игры требует не более 4 человек. Необходимы четкие требования к возможным кандидатам: специалисты по геймдизайну, программированию и концепт-арту.

4 этап: Разработка

Составить расписание для рабочего процесса над проектом. Существуют целые системы циклов производств, которые помогут системно подойти к разработке. Обсуждение нюансов идеи и детализирование документации по

проекту. Каждый отдельный пункт написанного ранее концепт-документа и превращать его в отдельный набор документации.

Файл концепт-документа является не просто текстовым документом, а содержит дополнительное оформление. Цели этого оформления:

1. Предоставить читателям дополнительные сведения и вместе с тем персонализировать документ.

2. Более-менее подробно изложить суть игры, так чтобы возможный инвестор или новый член команды понял, с чем он имеет дело, не потратив очень много времени на чтение документа.

3. Добавить документу солидности, оно информативно и лаконично.

Отдельно отмечается такая полезная вещь, как автоматически создаваемое оглавление, содержащее ссылки на указанные в нем разделы. Оно ускоряет доступ к нужным фрагментам документа, хотя концепт должен читаться за один раз, иногда необходимо вернуться и освежить в памяти какие-то моменты. Количество страниц с учетом титульного листа, оглавления и раздела контактов должно быть не больше 6–7 страниц. Пожалуй, такой объем вполне удовлетворителен – документ по-прежнему можно быстро прочесть, при этом он является вполне информативным.

Выводы. Многие исследователи в разные века и в разных направлениях обращались к понятиям игра и игровая культура. Происхождение игр связывается разными исследователями с магико-культовыми ритуалами или с врожденными биологическими потребностями организма и психики человека. Сегодня сформировалось несколько научных подходов к исследованию игры, в рамках которых также существуют отдельные самостоятельные направления. Феномен игры как определяющего начала (концепта) игровой культуры представлен в современных исследованияхс различных аспектов.

Развитие компьютерных, цифровых технологий позволило расширить границы для трансформации игры и игровой культуры в целом. Появляется новый тип «компьютерная игра». Создание самой игры происходит только после написания пакета документов, в которых раскрывается основная идея. Данная работа требует больших усилий и навыков самоорганизации. Координатор или управляющий проектом может сам и не прописывать код или дизайн, но он должен правильно составить план-график работы и проследить за его выполнением.

Ключевые понятия: игра, игровая культура, геймдизайнер, дизайн-концепт, контент, USP, фич, геймплей, game loop.

Лекция 2

Игровая индустрия. Игровая культура в условиях компьютеризации

Курс лекций

Основные вопросы

1. Индустрия компьютерных игр.
2. Экономический аспект.
3. Культурный аспект.
4. Нормативно-правовое регулирование игровой индустрии.
5. Игровые платформы: персональные компьютеры, ноутбуки, игровые консоли, мобильные устройства (сотовые телефоны, смартфоны, планшеты).
6. Програмное обеспечение игровых платформ.
7. Этапы реализации творческой идеи.
8. Издатели (создание) и продвижение компьютерных игр.
9. Игровое сообщество (игроки). Жанры компьютерных игр.

Цель. Ознакомиться с развитием игровой индустрии. Рассмотреть динамику игровой культуры в условиях компьютеризации.

Игровая индустрия

В настоящее время создание и продвижение игр взаимосвязано с развитием компьютерных технологий, которые впоследствии применяются и для других целей. Индустрия компьютерных игр является отдельным сектором экономики, который связан с разработкой, продвижением и продажей компьютерных игр. Игровая индустрия тесно связана с производством центральных процессоров и других компонентов персональных компьютеров. Для эффективного функционирования в сфере игровой индустрии необходимы высококвалифицированные специалисты: программисты, геймдизайнеры, дизайнеры, специалисты по звуку, копирайтеры, 3D- и 2D- художники и др.

В современном мире создание видеоигр является одним из наиболее крупных сегментов индустрии развлечений, который с каждым годом набирает скорость роста. По степени влияния на потребителей и вовлеченности их в интерактивное окружение, предлагаемое видеоиграми, этот сегмент уже давно выделяется среди других видов развлечений.

Разработку игр (или геймдев – GameDev) невозможно рассматривать обособленно от индустрии компьютерных игр в целом. Непосредственно создание игр – это только часть комплексной «экосистемы», обеспечивающей полный жизненный цикл производства, распространения и потребления таких сложных продуктов, как компьютерные игры [34].

В структуре современной игровой индустрии можно выделить следующие уровни: платформы, игровые движки, разработка видеоигр, издание и оперирование, популяризация и потребление.

В 1960-х годах появляются первые игры на персональных компьютерах, которые являются крупной игровой платформой. Это были текстовые приключенческие игры, в которых общение между игроком и компьютером осуществлялось посредством ввода команд через клавиатуру. Дальнейшая эволюция игр тесно связана с ростом производительности и возможностей компьютерного оборудования: популяризация компьютерных мышей (взаимодействие человек-компьютер), появление звуковых карт (возможность воспроизводить полноценные звуки), ежегодное улучшение качества выводимого изображения, и конечно развитие компьютерный сетей, вылившихся в итоге в появление Интернета [34].

Вместе с игровыми аппаратами игры развивались на платформе игровых приставок. И хотя история приставок началась еще в 1972 году вместе с релизом первой в мире домашней игровой приставки Magnavox Odyssey, прорыв в этом направлении начинается в 1977 году, когда в продажу поступает игровая приставка Atari 2600. Благодаря инвестициям Atari популяризация компьютерных и видеоигр выходит на новый уровень. В качестве крупнейших кроме Atari в историю вошли Sega, Nintendo, Sony и Microsoft.

Началом развития игровой индустрии можно определить с выпуском игры Space Invaders в 1978 годукоторая смогла смотивировать крупных инвесторов обратить внимание на игровую индустрию. Эта видеоигра была выпущена в Японии на платформе, под названием аркадные аппараты. Игра стала популярной сперва на территории Японии, а затем быстро нашла свое распространение в Америке. За 2 года было продано 60 000 аппаратов, которые стояли во всех барах, боулинг-клубах и кинотеатрах. Благодаря Space Invaders начался бурное развития аркадных аппаратов, ставших первой ступенью эволюции индустрии игр. В 1979 году это было настоящим прорывом: американский дистрибьютор игры Belly Midway утроил рынок игровых автоматов в США, доведя объёмы продаж до 1,33 млрд. долларов за год. Многие известные компьютерные игры впервые появились именно на игровых аппаратах и уже позднее были портированы на другие платформы. Например, Pac-Man, Street Fighter, Killer Instinct [34].

С эволюцией сетей на смену привычным всем однопользовательским клиентским играм приходит новое поколение игр, дающих возможность играть совместно с другими игроками. Самая первая онлайн игра была сделана в текстовом виде и запущена в сети TelNet в 1978 году. Разработчиками этой игры были Ричард Бартл и Рой Трабшоу. Они назвали

свое творение Multi-user Dungeon, сокращенно MUD. В дальнейшем эта аббревиатура стала использоваться в качестве классификации для многопользовательских онлайн игр, где абсолютно все отображается в виде текста. MUD является истоком всех виртуальных миров.

С появлением интернета в дополнение к привычным всем однопользовательским играм приходят онлайн проекты: клиентские и браузерные. В конце 90-х растущие скорости доступа в интернет делают возможным становление браузера как самостоятельной игровой платформы. Он выступает в роли операционной оболочки для игр, позволяя играть без установки дополнительных программ.

В середине 2000-х наступает экспоненциальный рост социальных сетей, которые становятся новым способом общения между людьми. Большое количество разработчиков игр стали ориентироваться на следующий вид «социальные игры». Это стало следующим шагом в развитии игровой индустрии. Темпы годового роста превышали 200% потому, что это была инновация и ранее рынка даже не существовало. Однако насыщение пришло довольно быстро. На смену низкокачественным игровым продуктам, производимым с бешеной скоростью с целью занять рынок, приходят высококачественные игры с хорошей графикой и серьезным геймплеем [13].

Следующий виток развития игровой индустрии произошел благодаря колоссальному росту популярности мобильных телефонов, которые стали новой платформой общения людей и новой платформой распространения игр. Игры на Java, которые были на мобильных телефонах не составляли серьезной конкуренции ни браузерным, ни клиентским, ни социальным играм. Однако начиная с конца первого десятилетия 2000-х годов появление смартфонов, под которые можно было разрабатывать мощные проекты, открыло для индустрии рынок мобильных игр. Реализация мобильной платформы сейчас происходит через два типа устройств: смартфоны и планшеты. Мобильные игры, в настоящий момент, являются перспективным рынком, способном генерировать максимальные прибыли при низком пороге вхождения [13].

Многие разработчики рассматривают индустрию компьютерных игр лишь как сектор экономики, в котором можно неплохо заработать. Но, на самом деле, создание игр это ещё и инструмент культуры. Для современных дошкольников игры заменяют собой детские сказки, создающие мировоззрение, для подростков они создают персонажей-кумиров, заменяя собой классические книги, и лишь для взрослых людей игры служат только развлечением. При правильном подходе, идеи, заложенные в компьютерные игры, гораздо лучше усваиваются детьми, чем идеи из книг или мультфильмов. Современные дети познают мир во многом через призму

компьютерных игр. Поэтому культурный потенциал компьютерных игр имеет очень большое значение и огромный потенциал.

Компьютерные игры – не просто деятельность: это деятельность конструирования миров. Психологический термин «конструирование миров» появился несколько лет назад. В научный обиход его ввел отечественный психолог А. Г. Асмолов. Конструирование миров – процесс создания образа мира в человеческой психике. Люди, народы, континенты, эпохи и галактики упаковываются в мир души. Чтобы поместиться в человеческую голову, вселенная должна быть свернута в «сконструированный мир».

Как сконструированный мир, любая популярная компьютерная игра имеет собственную физику и свойства пространства, искусственную историю и течение времени, оригинальную философию, этику и мораль. Игры дают возможность игроку активно действовать в сконструированном мире. Чем-то это похоже на карнавал, но только степень свободы «смены масок» в компьютерных играх неизмеримо выше.

Основным условием для существования компьютерных игр является наличие технических устройств, способных создавать управляемые подвижные изображения. На сегодняшний момент такими устройствами являются: персональные компьютеры и ноутбуки, игровые консоли, мобильные устройства (сотовые телефоны, смартфоны, планшеты) [34].

Игровые консоли представляют собой те же компьютеры, но лишь с одной функцией – воспроизведение игр. Узконаправленность этих устройств делает их эффективнее и проще в использовании. В некоторых странах игровые консоли даже популярнее обычных полнофункциональных компьютеров. Индустрия консольных игр накладывает некие ограничения на разработчиков. Для выпуска игры на консоли необходимо получить лицензию от производителя консоли. Благодаря этому, игроки защищены от низкокачественных игр, но страдают от малого общего количества выпущенных игр [34].

Мобильные устройства по техническим характеристикам гораздо слабее стационарных компьютеров, поэтому мобильные игры выглядят проще обычных игр. Но, тем не менее, развитие уровня самих мобильных устройств, открывают возможности и для развития рынка мобильных игр.

Управление техническими устройствами осуществляется с помощью компьютерных программ. Игры являются такими же программами. Изначально компьютерные игры создаются с помощью программного кода, но в последнее время стали появляться особые программы – игровые движки (game engine). Движки содержат в себе множество уже готовых игровых процессов, механик, элементов. Использование игрового движка существенно сокращает время разработки новой игры. При использовании

игрового движка разработчики больше концентрируются не на рутинных работах, а на творческих моментах разработки игры.

Имея готовую игровую платформу и специальные программы, разработчики создают компьютерные игры из своих творческих идей. В разработке компьютерных игр участвуют специалисты самых разных профессий. Это и игровые программисты; и художники, дизайнеры, левелдизайнеры; и композиторы, аранжировщики, актеры [34].

Продажей созданных компьютерных игр занимаются не сами разработчики, а издательские компании. К задачам издательства можно отнести полное материально-техническое обеспечение процесса разработки игры. В практике существует несколько вариантов отношений между разработчиками и издателями: *издатель – правообладатель* (полностью спонсируют процесс разработки, и юридически права на игру принадлежат именно такой компании), *издатель-разработчик* (распределение ответственности и прибыли), *издатель-сервис* (только предоставляет удобный инструмент продажи игры) [26].

Просто выпустить игру в продажу – недостаточно для хорошего результата. О появлении игры ещё нужно сообщить как можно большему числу потенциальных игроков. Этой задачей занимаются различные средства массовой информации, в основном – игровая журналистика: выставки, сайты, журналы, тв-передачи.

Игрок – это одновременно и источник прибыли всей игровой индустрии, и ценитель творчества разработчиков, и потенциальный участник других слоёв игровой индустрии.

Отдельные игроки часто объединяются в целые *игровые сообщества* по интересам. Объединение происходит как на форумах уже популярных общеигровых сайтов, так и наоборот – вокруг спонтанно появившегося сообщества, основанного на отдельной игре или игровой серии, начинают появляться новые сайты, форумы, интернет-группы.

Наиболее опытные игроки превращают своё хобби в профессию, так они попадают в *киберспорт*. По всему миру периодически проводятся различные чемпионаты, на которых киберспортсмены выясняют, кто из них самый лучший. Не каждая компьютерная игра подходит для спортивных соревнований, но их достаточно много, чтобы организовывать большие мероприятия с набором самых разнообразных дисциплин, наподобие Олимпиады. Сейчас киберспорт развился в целую отдельную индустрию, со своей инфраструктурой, финансированием, звёздами и знаменитостями. Умение играть лучше всех может открыть дорогу к славе и богатству.

Многие фанаты игр находят свое отражение в творчестве и не ограничиваются одним потреблением игр: создают собственные сайты,

рисунки, журналы, видеопередачи, косплеи, игровые дополнения, полноценные игры. Это ещё раз доказывает, что игры несут культурную ценность, и находят отклик в сердцах игроков.

Привести точную классификацию компьютерных игр очень сложно. Все зависит от того какой элемент или сегмент берется за основной показатель.

Жанры игр на ПК: РПГ, платформеры, шутеры, стратегии, экономические стратегии, военные стратегии, моба (Moba), королевская битва или Battle Royale.

Мобильные игры: тайм-киллеры (пазлы, головоломки, стрелялки, паркур и др.), симуляторы (гонки, авиа-симуляторы, спортивные игры), карточные игры (настольные игры), текстовые квесты.

Отдельно стоят *браузерные игры, социальные игры и консольные игры*.

Вследствие того что критерии принадлежности игры к тому или иному жанру не определены однозначно, классификация компьютерных игр недостаточно систематизирована, и в разных источниках данные о жанре конкретного проекта могут различаться. Существуют игры, в которых присутствуют элементы нескольких жанров. В этом случае игру причисляют либо к одному из жанров, который является основным, либо к нескольким, больше всего выделяющимся в игре [16].

Ниже приведены классификации игр по различным параметрам: жанрам, количеству игроков, стилистике, платформы.

Классификация по жанрам

1. Action (экшен)
   1.1. 3D-шутеры
   1.2. Шутеры от первого и от третьего лица
   1.3. Тактические шутеры
   1.4. Файтинги
   1.5. Избей их всех (Beat 'em up)
   1.6. Слэшер
   1.7. Платформер
2. Аркада
3. Симуляторы/Менеджеры
   3.1. Технические
   3.2. Спортивные симуляторы
   3.3. Спортивные менеджеры
   3.4. Симулятор строительства и управления
4. Стратегии
   4.1. Стратегии по схеме игрового процесса
      4.1.1. Стратегии в реальном времени

4.1.2. Пошаговые стратегии
4.1.3. Карточные стратегии
4.2.Стратегии по масштабу игрового процесса
4.2.1. Варгеймы
4.2.2. Тактические стратегии
4.2.3. Экономические стратегии
4.2.4. Глобальные стратегии
4.2.5. Симуляторы бога
5. Приключения
5.1.Текстовая приключенческая игра
5.2.Графический квест
5.3.Головоломки
5.4.Приключенческий боевик
5.5.Симулятор свиданий
5.6.Визуальная новелла
6. Музыкальные игры
7. Ролевые игры (RPG)
7.1. Тактическая
7.2. Подземелье
7.3. Японская
7.4. Ролевой боевик
7.5. Hack and slash
7.6. Roguelike
7.7. ZPG
8. Головоломки, логические, пазлы
9. Традиционные и настольные
10. Текстовые
10.1. Игры в псевдографике
10.2. Текстовые симуляторы
10.3. Текстовые RPG
10.4. Текстовые Квест
10.4.1. Текстовые приключения
10.4.2. Графические квесты
10.4.3. Квесты-головоломки
10.4.4. Визуальные романы
10.4.5. Симуляторы ходьбы
10.5. Книга-игра
10.6. Языковая игра
10.7. MUD

Классификация по количеству игроков

- Одиночные
- Многопользовательские
- Многопользовательские на одном компьютере
- Многопользовательские офлайн-игры
- Массовые онлайновые

Классификация по стилистике

- Вестерн
- Киберпанк
- Космические
- Постапокалиптические
- Стимпанк
- Современные
- Фэнтези

Классификация по платформам

- Персональные компьютеры
- Игровые консоли/приставки
- Мобильные телефоны и КПК
- VR (виртуальная реальность)

Индустрия и технологии компьютерных, мобильных и видеоигр стремительно развиваются. Совершенствование технологий открывает захватывающие возможности для разработчиков игр и медиа-студий, позволяющие применять расширенные аналитические средства для дальнейшего улучшения разработки и оптимизации усилий по монетизации игр. Если появятся новые законы, легализующие азартные сетевые игры, то в фокусе расширенных аналитических средств и микросегментации появится новый объект для стимулирования монетизации игроков на основе моделей азартных игр. Чем больше пользователей будут тяготеть к смартфонам и планшетным компьютерам, тем больше бизнес-моделей игр будет двигаться в направлении моделей бесплатного минимума; при этом будут возникать новые модели монетизации игроков, которые потребуют моделирования предрасположенностей и сегментации для лучшего определения ориентиров. Все это становится благодатной почвой для специализированных решений в области игровой аналитики и для аналитиков, специализирующихся на технологиях работы с большими данными для экономически эффективного анализа поведения пользователей игр и потребителей.

Выводы. Жизненый цикл любой игры проходит очень сложный путь: от создания до продвижения, что не является гарантией успеха и популярности. Игры которые завоевали популярность напрямую зависят от главного элемента – своих игроков. Игроки могут объединятся в различные сообщества, которые отличаются по своим направлениям деятельности. Некоторые игры, в силу своей высокой популярности, стали началом развития такого направления как киберспорт.

Развитие компьютерных или цифровых технологий напрямую влияют на совершенствование игровой индустрии, в связи с чем появляются новые жанры игр, в том числе компьютерных и мобильных.

История становления игровых платформ показывает, что с развитием технологий ускоряется процесс появления инновационных платформ для игр. Некоторые из них «взрывают» рынок и вытесняют старые, некоторые не обретают взрывного роста но занимают свою нишу – к примеру Smart TV, умные часы, игры в самолетах, игры на вейпах и др. В настоящее время широкую популярность в игровой индустрии начинает преобретать новая технология взаимодействия человек-компьютер. Это может стать как виртуальная или дополненная реальность в текущей реализации, так и виртуальная реальность с полным погружением.

Ключевые понятия: игровая индустрия, игровые платформы, игровые консоли, виртуальная реальность, дополненная реальность, GameDev, конструирование миров, мобильные игры, социальные игры, компьютерные игры, жанры компьютерных игр, РПГ, платформеры, шутеры, стратегии, моба (Moba), королевская битва или Battle Royale, тайм-киллеры, симуляторы, карточные игры (настольные игры), текстовые квесты, браузерные игры, социальные игры, консольные игры.

.

3. ПРАКТИЧЕСКИЙ РАЗДЕЛ

3.1 Тематика лабораторных работ

Тема 3. Игровая культура в условиях компьютеризации (2 часа)

Лабораторная работа 1. Аналитический обзор компьютерной игры

Тема 4. Дизайн-концепт в разработке компьютерных игр (4 часа)

Лабораторная работа 2. Морфологический анализ элементов компьютерной игры

Лабораторная работа 3. Составление проектной документации для создания компьютерных игр (концепт-документ, дизайн-концепт)

Тема 5. Программная среда разработки 2-D игр (6 часов)

Лабораторная работа 4. Знакомство с визуальной средой программирования Scratch. Создание авторского проекта игры. (Constract)*

Тема 6. Программная среда разработки 3-D игр (6 часов)

Лабораторная работа 5. Знакомство с пакетом для создания трёхмерной графики и анимации Cinema 4D .

Лабораторная работа 6. Знакомство со средой разработки компьютерных игр Unity. Создание 3 - D игр (4 часа).

Тема 7. Создание авторского проекта компьютерной игры (6 часов)

Лабораторная работа 7. Создание авторского проекта компьютерной игры

3.2 Описание лабораторных работ

Тема 3. Игровая культура в условиях компьютеризации

Лабораторная работа 1. Аналитический обзор компьютерной игры

Цель. Приобрести основные знания по аналитическому обзору компьютерной игры, определение игровой культуры в условиях компьютеризации.

Задание 1. Рассмотрение аналитического обзора.

Методические рекомендации по выполнению

1. Ознакомиться с основными параметрами аналитического обзора.

Разработчики мобильных и социальных игр в своей деятельности применяют стратегию, основанную на применении углубленной аналитики игроков для анализа пользователей и принятия решений по его результатам. Аналитика игроков позволяет студиям-разработчикам социальных игр в режиме реального времени выявлять причины, по которым пользователи бросают игру, и определять других игроков с повышенным риском отказа от игры, что дает возможность вырабатывать стратегии удержания игроков, предотвращающие их уход из игры. Аналитические приложения оптимизируют вызванное рекламой взаимодействие и внутриигровые продажи виртуальных товаров, применения вознаграждений. Аналитические инструментальные средства позволяют выявлять мошенническое поведение игрока для его исключения. Мобильные и социальные игры используют аналитику для понимания того, какой контент и какие кампании работают лучше всего, зачастую в сочетании со встроенным A/B-тестированием и улучшением контента. Сегментация игроков по параметрам, связанным с устройствами, платформами, операторами связи, географическим местоположением и демографическими характеристиками, упрощает подготовку более эффективных внутри игровых и неигровых предложений брендов и ориентированных на партнеров предложений в реальном времени.

При создании игры необходимо определить, в каком жанре она будет, это является ключевым для создания и продвижения игрового продукта.

Существует различные варианты классификаций игр по жанрам. Одним из основных факторов, который выделяют во всех играх, является «действие», совершаемое в игре [20]:

– игры информации (ролевые игры) – главное в игре получение какой-либо информации;

– игры действия – главное, это совершение определенных действий;

– игры контроля – основной фактор в игре, это планирование и управление (рис.3.2.1).

Игры информации	Игры действия	Игры контроля
Обучение	Собирание	Забота
Загадки	Уклонение	Создание
Общение	Уничтожение	Контроль
Роль	Соревнование	Тактика
Изучение	Вождение	План

Рисунок 3.2.1 – Классификация компьютерных игр по группам

В связи с этими группами можно выделить 15 универсальных составляющих игрового процесса (гейплея) [20]:

Рисунок 3.2.2 – Схема жанров компьютерных игр

Все игры внутри одной группы похожи между собой основной идеей. Во многих играх присутствует синтез составляющих геймплея. В связи с этим бывает сложно определить точно жанр игры.

Следующим элементом анализа компьютерной игры, является описание сеттинга. *Сеттинг* – это принадлежность игры к сюжетной теме или определенному миру, то есть определение места и времени. Существует несколько наиболее популярных сеттингов по определению мира или эпохи, в котором происходит действие [23]:

Миры:

– Реальный мир;
– Параллельный мир (переход из реального в искаженный мир);
– Альтернативный мир (варианты развития настоящего);
– Фэнтези (сказочный мир с элементами магии);
– Освоение космоса (космические путешествия);
– Геройская мифология (супергерои, боги, сверхсущества, и т. д.);

– Современная мифология (апокалипсис, постапокалипсис, инопланетяне, зомби и т. д.);

– Христианская мифология (ангелы, демоны, ад и рай и т. д.).

Исторические эпохи:

– Зарождение жизни (простейшие организмы; клеточный уровень);

– Доисторические времена (пещерные люди, эра динозавров);

– Средневековье (рыцари, крестовые походы);

– Эпоха колонизации (морские путешествие, открытие новых земель);

– Эпоха индустриализации (стимпанк, XVIII-XIX вв.);

– Прошедшие войны (Первая и Вторая мировая война, война в Афганистане, Вьетнаме и другие);

– Информационная эпоха (киберпанк, антиутопия будущего);

– Эволюция (наличие нескольких эпох);

– Наше время.

Определение *целевой аудитории* является одним из основных факторов, влияющих на развитие игрового контента: возраст, пол, занятость, уровень дохода, социальная группа, игровое устройства, цель игры, личные интересы, образ жизни и т. д.

Список элементов игры – описание составляющих проекта: игровые механики, сущности и их взаимодействии.

Игрой цикл – принцип, согласно которому геймдизайнеры выстраивают главный элемент игровой механики, опыт игрока. Игровой цикл представляет собой действие игрока, результат от этого действия, реакцию игрока и запрос игры на следующее действие. Игрок постоянно совершает действие основного цикла: выполнить задание, получить опыт, силу, навык, чтобы выполнить следующее задание и т. д.

Метагейм (греч. Meta – за пределами, после и анг. Game – игра) или метагейминг, метаигра – использование в игре того, что относится к игроку: информация, рассуждения, комментарии, решения.

Модели подписки, например видеоигры в жанре MMO: World of Warcraft или Star Wars: the Old Republic подразделяются на несколько категорий:

– платный игровой процесс («Pay-to-play») – игроки должны ежемесячно вносить абонентскую плату;

– бесплатный игровой процесс («Free-to-play») – обычно включает предварительную оплату стоимости программного обеспечения без каких-либо дополнительных взносов;

– бесплатный минимум («Freemium») – позволяет игрокам получать доступ к игровому контенту и играть бесплатно, при этом за определенную

плату предлагается возможность получения дополнительного контента и доступа.

Для игр с платным игровым процессом игровая аналитика фокусируется на определении наиболее ценных игроков, стиля их игры, наличия с их стороны пропаганды игры и вовлечения в игровой процесс других игроков, а также определении их индивидуальных игровых особенностей и мотивов как игроков:

– продолжение подписки;
– возвращение к игре после пауз в подписке;
– стимулирование к подписке новых игроков;
– превращение в опытных и влиятельных руководителей игровых гильдий (кланов).

В число элементов игровых данных входят следующие параметры: традиционные ключевые показатели эффективности (KPI) игрового времени на инструментальной панели; время, затраченное на прохождение уровней; характеристики поведения (игра в одиночку или общение с другими игроками), выбор аватара, показатели стиля взаимодействия, пол аватара, поведенческие параметры игровой стратегии, связанные с играми твиты, активность в социальных сетях, язык и многое другое.

Задание 2. Составление аналитического обзора.

Методические рекомендации по выполнению

1. Выбрать тему из списка для аналитического обзора:
– игра для игрового автомата;
– мобильная игра;
– игра из социальной сети;
– экшн игра;
– музыкальная игра;
– стратегия;
– настольная игра;
– РПГ;
– головоломка.

2. Составить письменный аналитический обзор выбранной игры по рассмотренным параметрам по следующему плану:
– классификация игры;
– жанр;
– сеттинг;
– целевая аудитория;
– список элементов игры;
– список игровых циклов;

– метагейм;
– модели подписки.

Тема 4. Дизайн-концепт в разработке компьютерных игр

Лабораторная работа 2. Морфологический анализ* элементов *компьютерной игры

Цель. Приобрести основные знания по морфологическому анализу элементов компьютерной игры.

Задание 1. Рассмотреть технологии морфологического анализа.

Методические рекомендации по выполнению

1. Ознакомиться с основными параметрами морфологического анализа.

Морфологический анализ – это метод систематизации перебора вариантов всех теоретически возможных решений, основанный на анализе структуры объекта.

В 30-х гг. XX Фриц Цвикки (американский астроном швейцарского происхождения), с помощью такого метода, предсказал существование нейтронных звезд, а также предположил наличие во Вселенной так называемых «адских звезд» (черные дыры). Ф. Цвикки выпустил книгу «Морфологическая астрономия», где описал возможности прогнозирования астрономических открытий, с помощью разработанного им морфологического метода. Такой метод стали часто использовать изобретатели, писатели и фантасты.

Суть морфологического анализа заключается в систематическом обзоре всех мыслимых вариантов данного явления или объекта. На практике необходимо выбрать какой-либо объект и составить список его основных характеристик. Далее для каждой из характеристик нужно перечислить все возможные варианты. Комбинирование разных вариантов открывает огромное множество новых, еще нигде не фигурировавших объектов [23]:

2. Рассмотрите пример применения морфологического анализа для объекта «компьютер».

На первом этапе перечислить его характеристики:
– тип блока питания,
– производительность,
– способ управления,
– энергопотребление,
– габаритные размеры и т. д.

Чем больше выделено характеристик на этом этапе, тем большее количество вариантов может быть получено на выходе.

Далее для каждой из характеристик составляем список всех мыслимых и немыслимых вариантов, необязательно относящихся к данному объекту.

Например:

1. блок питания: электрический, паровой, механический, гидравлический, внутреннего сгорания, ядерный и т. д.;

2. производительность (в сравнении с лучшей моделью на данный момент): низкая, средняя, высокая, сверхвысокая и т. д.;

3. способ управления: ручной, голосовой, жестовый, мысленными командами, интуитивный, аппарат сам предугадывает желания;

4. энергопотребление (в сравнении с моделями данного времени): сверхнизкое, низкое, среднее, высокое, сверхвысокое;

5. размеры (в сравнении с современными моделями): сверхмалые, малые, средние, большие, сверхбольшие.

Таким образом, комбинируя полученные варианты, можно представить себе, например:

– управляемый мыслью сверхмалый компьютер с чрезвычайно низким энергопотреблением (как, например, от биотоков самого человека).

– компьютер, размером с планету с фантастической производительностью, использующий для работы энергию внутренних процессов занятого им небесного тела.

Вариантов может быть очень много, остается только выбрать наиболее подходящий. Основное значение имеет первоначальное количество характеристик в первом шаге и оригинальность вариантов их исполнения во втором. Обширная и полная морфологическая таблица позволяет сформировать оригинальное фантастическое допущение [23].

Задание 2. Составить морфологический анализ для заданных объектов.

Методические рекомендации по выполнению

1. Из списка выбрать тему для морфологического анализа.

– Строительство в игре.

– Реализация возможности отдыха игрового персонажа (условия, необходимые предметы, приносимая польза).

– Игровой транспорт.

– Приспособления для перемещения по миру в фэнтезийной RPG.

– Рисование в обучающих играх.

– Создание музыки в детских обучающих играх.

– Боевые классы для кооперативной игры (тип урона, оружие, особенности, слабые стороны, преимущество, взаимодействие с другими

классами).

– Материалы для создания одежды и обмундирования в игре (материал, описание, свойства).

– Система питания (приготовление еды).

– Система развития уровней и получения опыта.

– Механизм охоты в играх с открытым миром (условия, поведение тех, на кого охотится игрок, полезность охоты, возможные трофеи).

2. Составить морфологический анализ по представленному алгоритму. Результат показать преподавателю.

– Точно сформулировать проблему.

– Определить важнейшие элементы.

– Определить варианты исполнения элементов.

– Занести их в таблицу.

– Оценить все имеющиеся в таблице варианты.

– Выбрать оптимальный вариант.

Тема 4. Дизайн-концепт в разработке компьютерных игр

Лабораторная работа 3. Составление проектной документации для создания компьютерных игр (концепт-документ, дизайн-концепт)

Цель. Рассмотреть основную проектную документацию для создания компьютерных игр. Разработать пакет документов для собственного проекта компьютерной игры в сфере культуры.

Задание 1. Рассмотреть основную проектную документацию проекта.

Методические рекомендации по выполнению

1. Ознакомиться с основными документами для создания проекта.

Человек, которые занимается разработкой правил игры, называется геймдизайнер. Для такого специалиста нужны навыки психолога, аналитика, игрока, умение работать в команде и руководить. Не лишним будет и следующие навыки: художественный вкус, рисование, 3 D – моделирование, программирование, знание математики и физики [26].

Компании-разработчики компьютерных игр, в зависимости от своей работы, могут варьировать перечень подготавливаемых документов. Но существует три основных документа:

1. *концепт-документ* – раскрывает основные особенности игры, обычно 2-6 страниц текста, по возможности, разбавленных иллюстрациями, из которого приблизительно видно, какие ресурсы потребуются на разработку;

2. *дизайн-документ* – дополненный деталями разработки и наполнения концепт-документ, после своего утверждения;

3. *документ-предложение* (англ.proposal document) краткое описание игры, без внутренних деталей разработки, объясняющее потенциальному инвестору, почему игра принесет прибыль.

Некоторые компании могут отдельно создавать документ (technical design document), который раскрывает технические требования к игре (объём памяти, используемые утилиты, языки программирования, базы данных).

2. Ознакомиться с содержанием концепт-документа.

Содержание концепт-документа должно отражать основную цель проекта. Концепт-арт – направление в искусстве, предназначенное для того, чтобы визуально передать идею произведения, но не форму или внешние атрибуты. Как правило, создаётся на начальной стадии разработки проекта и предназначается для использования в компьютерных играх, комиксах до создания финальной версии. Также называется «концепт-дизайном» [26].

Введение. Краткая формулировка всей идеи игры в одном или нескольких предложениях. Краткое описание проекта с указанием жанра, аудитории, основных особенностей.

Жанр, сеттинг, аудитория и платформа. Основные сведения обо всех этих составляющих и о позиционировании игры.

По необходимости перечислить платформы, на которых планируется создание игры. Для платформы PC кратко указать минимальные и рекомендуемые системные требования и, если необходимо, дополнительное оборудование (модем). Для составления таких требований можно воспользоваться примерным вариантом таблицы.

Требования	Минимальные	Рекомендуемые
Процессор		
ОЗУ		
Видео карта		

Основные особенности игры. Ключевые особенности (USP–unique selling points), отличающие игру от других игр в этом жанре и ориентированных на туже целевую группу. Ключевые особенности – это не только новые, нигде не используемые, но правильно подобранные самые распространенные в других играх близких по жанру, или их сочетании.

В данном разделе можно указать и примерный объем игры (время прохождения).

*Подробное описание игры. Н*еобходимо донести саму суть игры с точки зрения игрока: список основных элементов (механика), описание игрового цикла, принцип работы.

Сравнение и предпосылки создания. Обоснование игры, общие тенденции игрового рынка в данном направлении. Вопросы лицензирования (если используется).

Контакты. Необходимо указать контактные данные, для дальнейшего сотрудничества: контактные лица, телефоны, E-mail, адрес, сайт.

3. Рассмотреть примерную структуру дизайн-документа.

Дизайн-документ – это план работы от начала и до конца проекта. Данный документ может в процессе создания игры дополняться или уточняться. В начале работы в дизайн-документе должны находиться основные задачи и приблизительные методы решения, а именно [26]:.

Схема игры. Что должен делать игрок, какова конечная цель, что мешает ее достижению.

Интерфейс. Подробно описанная функциональная часть (что можно делать, каким образом – меню, мышь, горячие клавиши, кнопки…).

Игровая механика. Как устроен игровой мир, какие характеристики есть у его объектов, формулы движения, боя и всего остального, ролевая система, физика.

Программные механизмы и алгоритмы. Какими характеристиками будут обладать графический движок, ИИ, сетевой код, интерфейс, редактор карт, звук.

Графика. Сколько и каких вам понадобится моделей, анимаций, двумерной графики, роликов, обоев (фон). Здесь необходимы какие-нибудь наброски, концепт-арт (concept art), по которым можно почувствовать визуальный стиль игры.

Звуки и музыка. Темы, вид и способ отображения звуков, набор звуковых эффектов.

Сюжет. Общая сюжетная канва, план кампаний, основные задания и т.п. – в зависимости от жанра. Каждая из предполагаемых карт должна быть запланирована здесь.

Игровой мир. Основные персонажи, способом добычи и производства.

Сотрудники, зарплаты, сроки и план работы [26].

Задание 2. Подготовить проектную документацию авторского проекта.

Методические рекомендации по выполнению

1. Выбрать тему для создания авторской игры (*из списка*).
2. Выбрать обязательные элементы, которые будут использованы в

проекте *(выбирают 5 карточек с различными изображениями, которые могут быть использованы в качестве персонажей, фона, игровых объектов*).

3. Составить первоначальный концепт-документ авторского проекта по плану (объем не должен превышать 2-х страниц формата А4):

– Введение;

– Сеттинг, целевая аудитория;

– Описание основных действий главного героя;

– Список USP (уникальных, неповторимых элементов, которые привлекут игроков).

4. На основе концепт-документа, составить первоначальный дизайн-документ, используя примерный план [26]:

Содержание:

1.Введение

2.Концепция

2.1.Введение

2.2.Жанр и аудитория

2.3.Основные особенности игры

2.4.Описание игры

2.5.Предпосылки создания

2.6.Платформа

3.Функциональная спецификация

3.1.Принципы игры

3.1.1.Суть игрового процесса

3.1.2.Ход игры и сюжет

3.2.Физическая модель

3.3.Персонаж игрока

3.4.Элементы игры

3.5.«Искусственный интеллект»

3.6.Многопользовательский режим

3.7.Интерфейс пользователя

3.7.1.Блок-схема

3.7.2.Функциональное описание и управление

3.7.3.Объекты интерфейса пользователя

3.8.Графика и видео

3.8.1.Общее описание

3.8.2.Двумерная графика и анимация

3.8.3.Трехмерная графика и анимация

3.8.4.Анимационные вставки

3.9.Звуки и музыка

3.9.1.Общее описание

3.9.2.Звуки, звуковые эффекты

3.9.3.Музыка

3.10.Описание уровней

3.10.1.Общее описание дизайна уровней

3.10.2.Диаграмма взаимного расположения уровней

3.10.3.График введения новых объектов

4.Контакты

5. Составленные концепт-документ и дизайн-документ проекта игры показать преподавателю.

Тема 5. Программная среда разработки 2- D игр (6 часов)

Лабораторная работа 4. Знакомство с визуальной средой программирования Scratch. Создание авторского проекта игры. (Constract)*

Цель. Приобрести основные знания по работе с визуальной средой программирования Scratch (* Constract). Создать авторский проект игры по созданной проектной документации.

Задание 1. Знакомство с визуальной средой программирования Scratch.

Методические рекомендации по выполнению

1. Зарегистрироваться на официальном сайте Scratch.

Scratch – это не только среда для программирования, это еще большое сообщество. Это проект группы Lifelong Kindergarten в MIT Media Lab. Scratch позволяет учиться программированию, развивать творческие способности, создавать игры и интерактивные анимации, а также общаться между собой, изучать и использовать проекты друг друга.

2. Перейдите в среду программирования Scratch по адресу https://scratch.mit.edu/ .

Язык интерфейса сайта может оказаться английским. Чтобы поменять его на русский, надо прокрутить страницу вниз и в выпадающем списке выбрать русский язык.

3. Зарегистрируйтесь на сайте. Нажмите кнопку «Присоединяйся» (рис.3.2.3).

Рисунок 3.2.3 – Вход для регистрации

4. Заполните поля: Псевдоним, пароль, почта и другие.
5. После подтверждения войдите в свою учетную запись.
6. Ознакомьтесь с вкладками меню (рис.3.2.4):

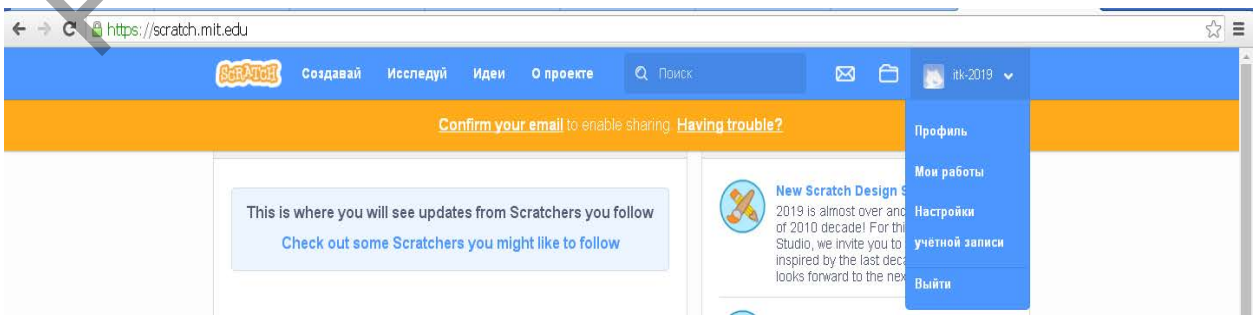

Рисунок 3.2.4 – Вкладки меню личного кабинета

Логотип Scratch в верхнем левом углу – переход на главную страницу сайта.

«Создавай» – откроется среда программирования Scratch.

«Исследуй» – просмотр чужих проектов. Часть из них доступна также с главной страницы. Чтобы посмотреть чужой проект, надо кликнуть по нему. Откроется страница, где слева будет сам проект, справа – его описание, внизу – комментарии. Проект будет в режиме исполнения, то есть вы смотрите готовую работу, а не то, как она запрограммирована и из каких блоков собрана.

«Идеи» – страница с руководством для начала работы.

«О проекте» – страница с основной информацией проекта: создатели и применение, цитаты и исследования.

Значок с вашим именем содержит дополнительное меню, где можно настроить сам профиль или учетную запись, просмотреть все ваши проекты.

7. Ознакомьтесь с вкладками меню профиля (рис. 3.2.5):

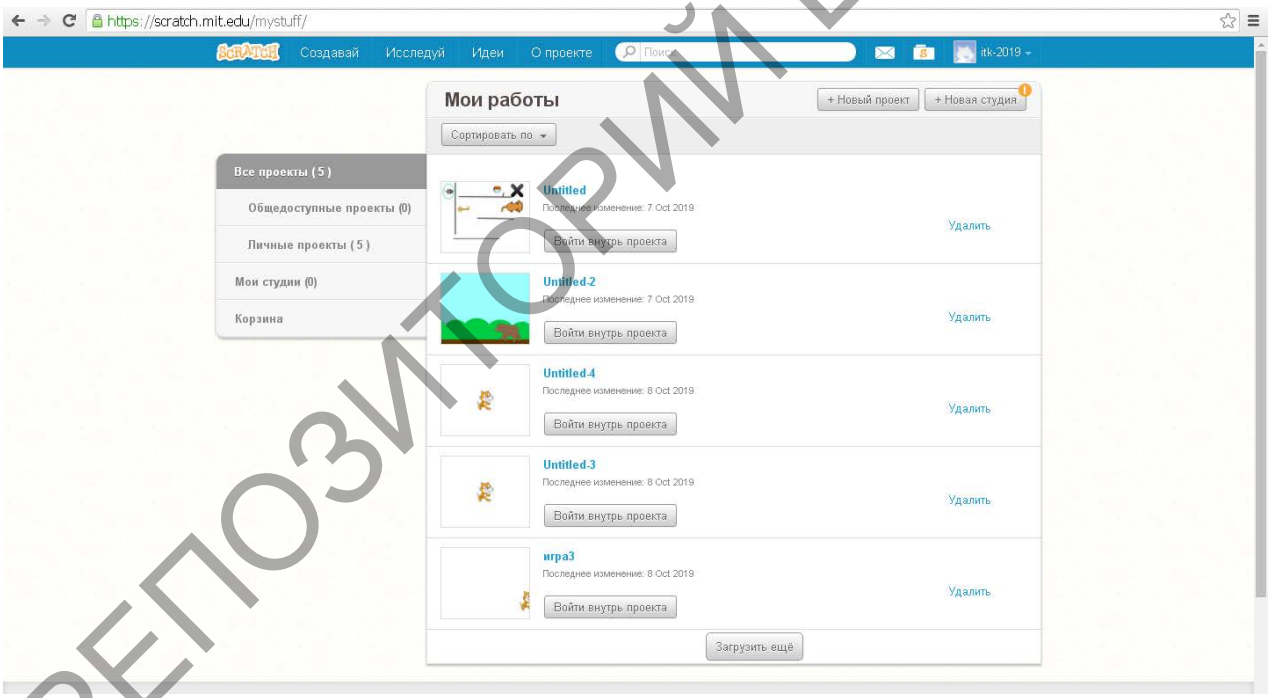

Рисунок 3.2.5 – Вкладки меню профиля

Вкладка «Мои работы» открывает страницу, с которой доступны собственные проекты. На эту страницу также можно перейти, кликнув по изображению папки в меню.

Чтобы начать новый проект, надо нажать кнопку «+ Новый проект» («Создавай»). Для редактирования ранее созданного проекта, нажать на кнопку «Войти внутрь проекта». Нажатие по названию проекта откроет его в режиме исполнения, то есть просмотра.

Задание 2. Изучить интерфейс Scratch.

Методические рекомендации по выполнению

1. Создать новый проект Scratch.

Если редактор открылся не на русском языке, нажмите глобус слева вверху и выберите русский язык (рис.3.2.6).

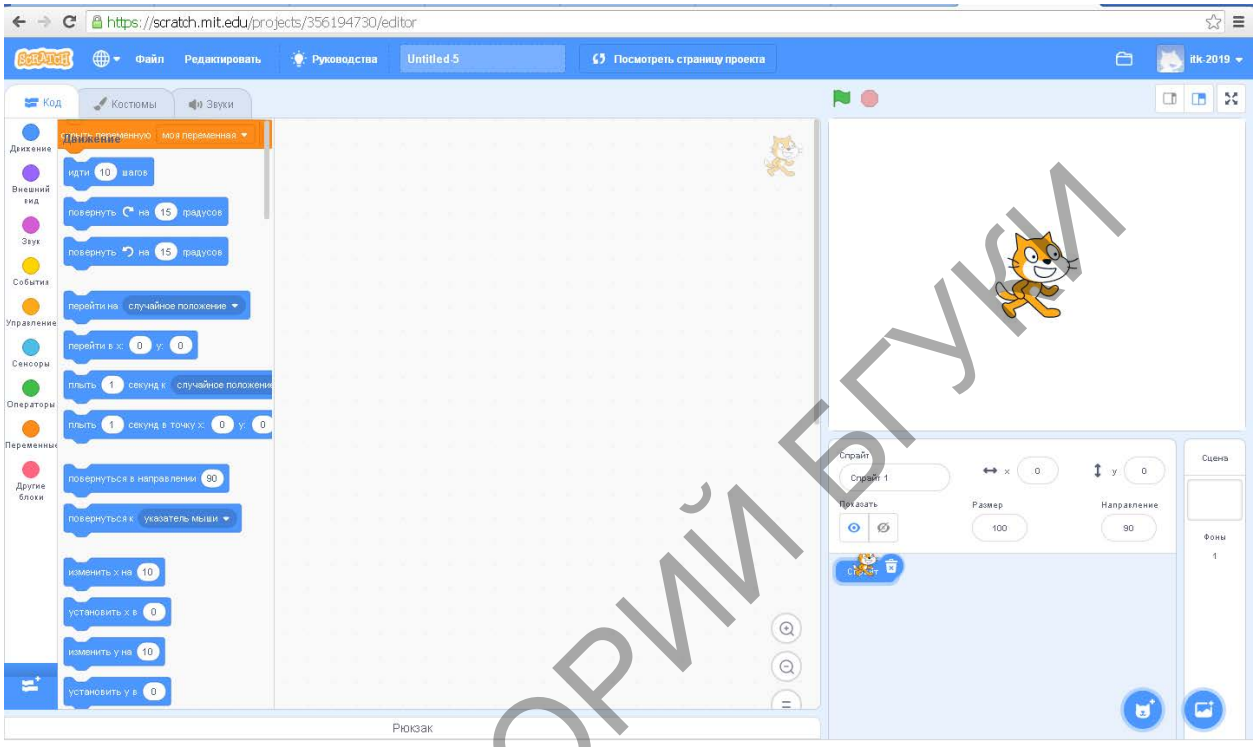

Рисунок 3.2.6 – Вкладки меню профиля

2. По умолчанию проект назван Untitled (Неозаглавленный). Введите имя для своего проекта, оно сохранится автоматически.

3. Если же нажать на кнопку «Опубликовать», ваш проект станет доступен всем для просмотра. Обычно сырые проекты не публикуют, и они остаются доступными только своим создателям.

4. Ознакомьтесь с интерфейсом проекта: Рабочая зона, Зона спрайтов, сцена (игровой холст).

5. В основной области экрана слева расположены кнопки: «Код», «Костюмы», «Звуки».

6. На вкладке «Код» расположены строительные блоки, которые задают поведение объектов (спрайтов). Они разделены по секциям-разделам «Движение», «Внешний вид», «Звук» и так далее. Нажимая на соответствующий цветной кружок, вы быстро перейдете к блокам необходимой секции. Большое пустое поле посередине – это самая важная область – редактор кода. Здесь конструируют код из блоков, которые перетаскивают из левой части.

– Движение (синий) – содержит команды перемещения объектов;

– Внешность (фиолетовый) – команды изменения внешнего вида объекта;

– Звук (малиновый) – команды управления звуком;

– Перо (темно-зеленый) – команды рисования на экране;

– События (желтый) – команды управления, контролирующие операторы;

– Управление (оранжевый) – условные операторы и операторы циклов;

– Сенсоры (голубой) – датчики, команды управления мышью, определяющие расстояние и координаты;

– Числа (ярко-зеленый) – операции с числами, логические операторы, вычисления, команды сравнения;

– Операторы (ярко-зеленый) – операторы для переменных.

7. Изучите работу в зоне спрайтов.

Слово «спрайт» часто используется в программировании игр для обозначения фигурок героев и предметов (рисунки или фотографии), то есть компьютерная графика.

Спрайт – это объект, наделенный программным кодом и как следствие свойствами и способностями. Со спрайтом можно взаимодействовать как с виртуальной сущностью, отдавать ей команды и получать от нее данные.

Кот в Scratch – не единственный объект-спрайт, но он появляется автоматически при создании нового проекта (рис. 3.2.7). Добавить на сцену другие спрайты можно несколькими способами: загрузить картинку с компьютера, нарисовать в самой среде программирования, выбрать из библиотеки. Для всего этого в Scratch предусмотрено специальное меню, которое находится внизу справа на панели спрайтов. Для добавление спрайтов из библиотеки, необходимо нажать на лупу и из открытой библиотеки выбрать подходящего героя (рис. 3.2.8). Все спрайты разделены по группам. Некоторые спрайты включают несколько вариантов костюмов.

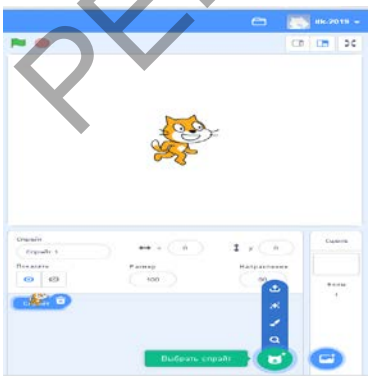

Рисунок 3.2.7 – Спрайт Кот

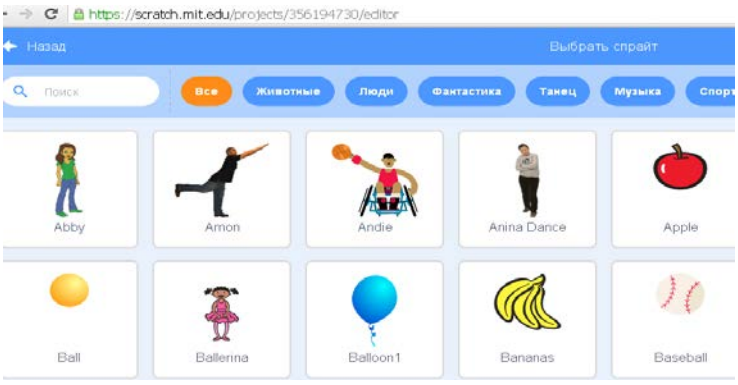

Рисунок 3.2.8 – Библиотека спрайтов

Кликнуть по спрайту, означает добавить спрайт на сцену. На панели спрайтов под свойствами появится его иконка. Рядом с иконкой кота.

Свойства объекта, где вы задаете ему имя, положение, размер и поворот, отображаются только для того спрайта, который выделен.

Для удаления спрайта, необходимо кликнуть по его иконке под свойствами. Спрайт выделиться синим цветом, появится крестик, кликнув по которому можно убрать спрайт из игры.

У каждого спрайта есть свои блоки в программном коде – своя программа, свое предназначение. Когда объект выделен, в верхнем правом углу поля редактора кода появляется подсказка, отображается фигурка выделенного на данный момент спрайта бледным цветом. Когда выделяется другой спрайт, то код предыдущего становится невидимым.

8. Самостоятельно изучите работу с выбором фона (рис. 3.2.9).

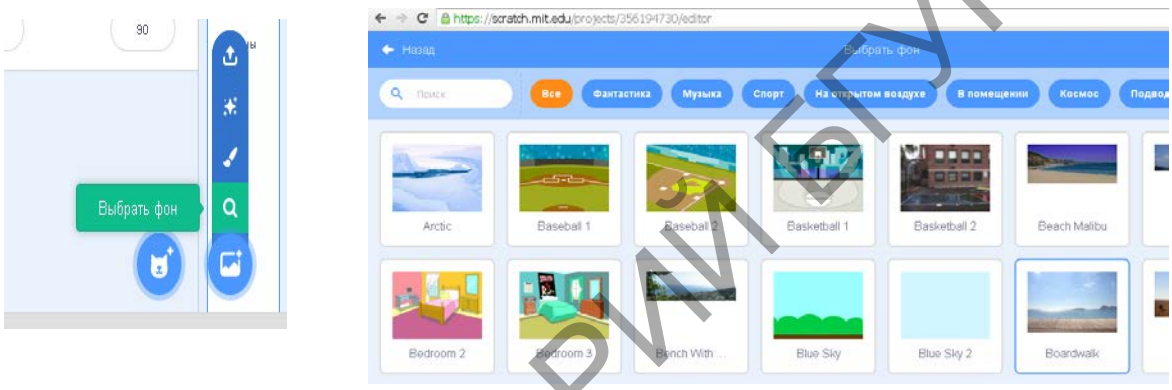

Рисунок 3.2.9 – Работа с фоном сцены

9. Самостоятельно изучите работу на вкладке «Костюмы».

Можно добавлять новые костюмы или редактировать предложенные варианты костюмов спрайтов (рис. 3.2.10).

10. Самостоятельно изучите возможности работы на вкладке «Звуки» (рис. 3.2.11).

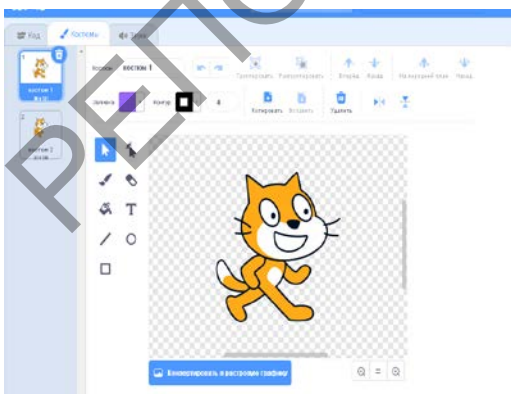

Рисунок 3.2.10 – Работа с костюмами спрайта

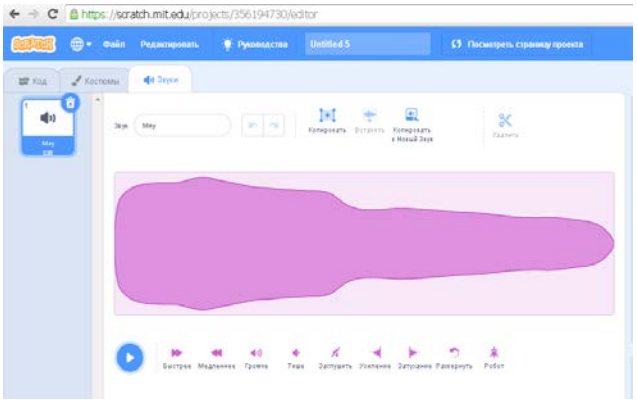

Рисунок 3.2.11 – Работа со звуком

Задание 3. Создать проект в Scratch

Методические рекомендации по выполнению

1. Из секции «Движение» перетащите в редактор кода блок «идти 10 шагов». Щелкните по нему мышью, спрайт кота на сцене будет смещаться на небольшое расстояние. Если кот на холсте далеко ушел от середины, его можно взять здесь мышью и перетащить в новое место холста.

2. Выберите из раздела «События» (желтый цвет) команду «когда флажок нажат».

3. Соедините две команды.

Последовательность блоков важна. Команда, которая расположена выше, будет выполняться раньше, чем команда, которая расположена ниже. Часто конфигурация самих блоков подсказывает, как их надо соединять. Программа всегда начинается с команды «когда флажок нажат».

4. Просмотрите результат. Щелкните по зеленому флажку над холстом.

5. Поменяйте значение шага в команде. Просмотрите результат.

6. Выберите из оранжевого раздела «Управление» блок «повторить 10 раз». Правильно соедините блоки.

Данная команда нужно для того, чтобы создать цикл движения. В реальности мы много раз делаем одно и то же: шаг левой, шаг правой, шаг левой, шаг правой, «мы шагаем в цикле». Циклы – это тоже строительные блоки и одно из важнейших понятий программирования. Необходимо поместить команду «идти 10 шагов» внутрь цикла «повторить 10 раз», а цикл присоединим к блоку «когда флажок нажат».

7. Просмотрите результат. Измените значения.

8. Замените «повторить … раз» на «повторять всегда».

9. Запустите программу. Просмотрите результат.

Кот должен дойти до края экрана, и остановится, но программа должна продолжать работать.

10. Остановите программу – нажать на красный кружок над холстом.

11. Из синего раздела «Движение» добавьте команду «если касается края, оттолкнуться» и поместите его в цикл.

12. Запустите программу. Просмотрите результат.

Кот будет ходить от левого края к правому без конца, до тех пор, пока вы сами не остановите программу. Кот будет ходить странно: влево идет ногами, а как направо – становится на голову.

13. Поменяйте поведение кота по умолчанию с помощью команды «установить способ вращения влево-вправо» из раздела «Движение» и поместите её в цикл.

14. Самостоятельно просмотрите другие варианты этой команды и оцените результат («кругом» или «не вращать»).

15. Составьте самостоятельно программу, согласно которой кот ходит туда-сюда, но через каждые 100 шагов останавливается на 1 секунду и говорит «Мяу». Для этого вам понадобятся две команды: одна из раздела «Управление», а вторая – из раздела «Звук».

16. Результат покажите преподавателю.

Задание 4. Создать проект игры «Собери» в Scratch.

Методические рекомендации по выполнению

1. Ознакомьтесь с планом (этапами) создания игры.
   1) Создание фона – игрового поля.
   2) Расстановка различных предметов (объектов) на игровом поле.
   3) Программирование главного героя.
   4) Программирование собираемых объектов.
   5) Программирование препятствий.
   6) Программирование финиша (финишной кнопки).

2. Выполните первый этап: создание игрового фона (рис. 3.2.12).

– Войдите в меню «Сцена», вкладка «Фоны».

– С помощью инструментов для рисования «прямоугольник» и «линия» создать поле 8х8 клеток.

– При желании можно добавить какой-либо текст (кнопка Т).

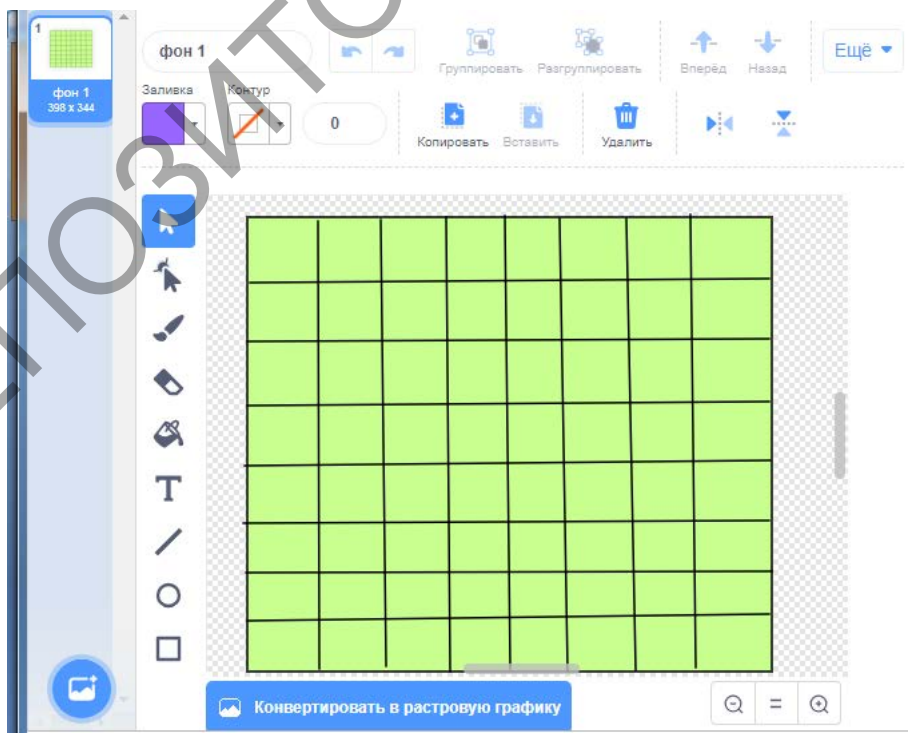

Рисунок 3.2.12 – Создание игрового фона

3. Выполните второй этап: расстановка игровых объектов.

– Добавьте спрайты из библиотеки: главный герой (котенок), яблоки

(то, что надо собирать) и препятствия (рис. 3.2.13).

– Количество объектов задать самостоятельно.

– Размер объектов уменьшить до 50.

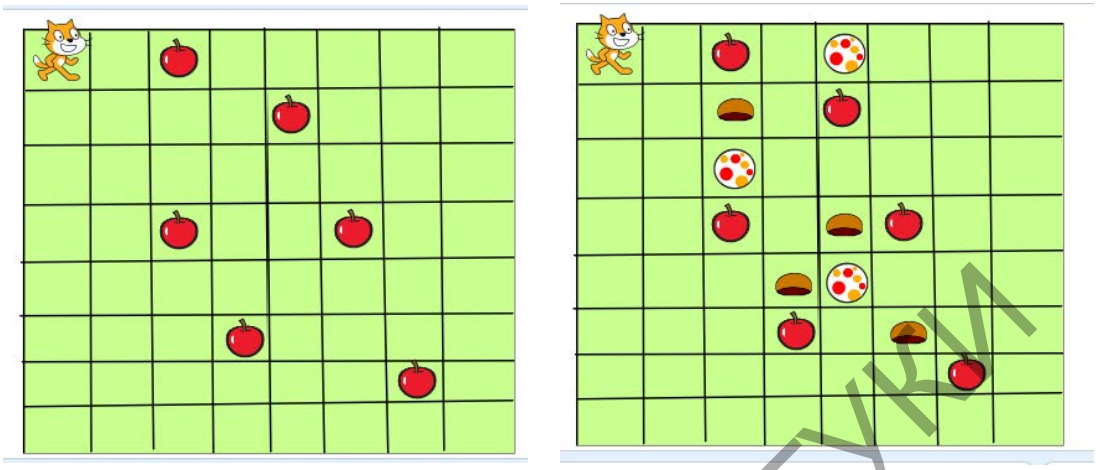

Рисунок 3.2.13 – Расстановка игровых объектов

4. Выполните третий этап: программирование «героя».

– Поместить героя в начальную точку (рис. 3.2.14).

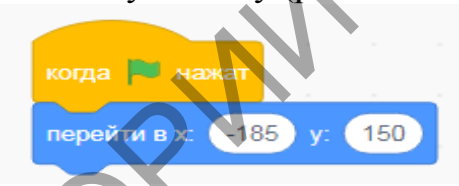

Рисунок 3.2.14 – Начальная точка объекта

– При необходимости задать начальное направление – например, «повернуться в направлении 90».

– Запрограммировать кнопки управления передвижения объекта стрелками с клавиатуры (рис. 3.2.15).

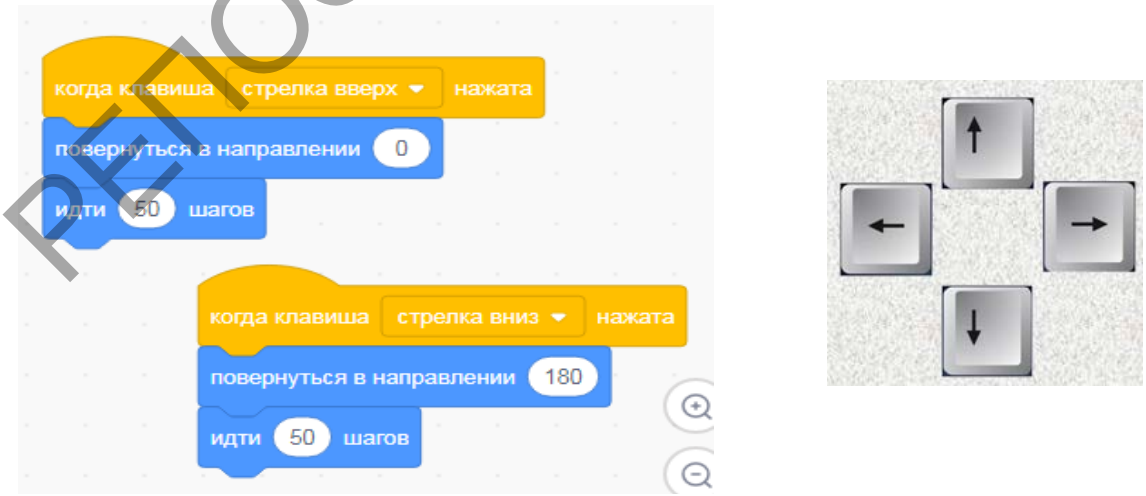

Рисунок 3.2.15 – Программирование кнопок управления

– Таких команд должно быть четыре. Направление задается следующим образом: вверх – 0; вправо – 90; вниз – 180; влево – -90.

– Длина шага определяется размером клеток на игровом поле. При

размере поля 8х8 клеток шаг получается – примерно 50.

– Проверить, как ходит герой, и исправьте длину шага.

5. Выполните четвертый этап: программирование собираемых объектов.

– Все команды и действия запускаются по команде «Когда флажок нажат» и через цикл «Повторять всегда», так как мы не знаем, в какой момент игры наш герой коснется именно этой фишки.

– Задайте реакцию на прохождение героя через ветвление «Если – то» с условием «касается Спрайт1» – команда «Спрятаться».

– Задать движение объектов (любым из 2х способов): 1) «идти 1 шаг», «ждать 1 секунду», «идти -1 шаг»; 2) «изменить размер на 5%», «ждать 1 секунду», «изменить размер на -5%» (рис. 3.2.16).

– Для «восстановления» объекта задайте в начале игры команду: «Когда клавиша пробел нажата» «показаться» (рис. 3.2.17).

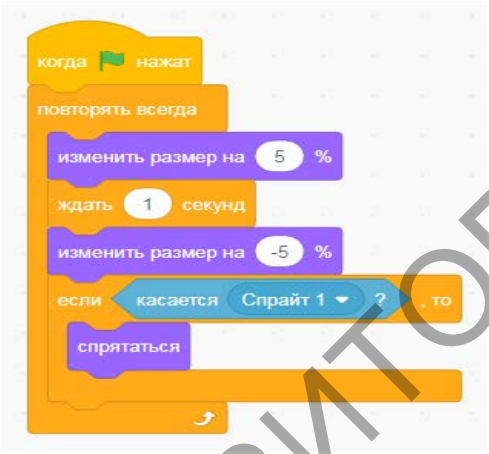

Рисунок 3.2.16 – Программирование движения объекта

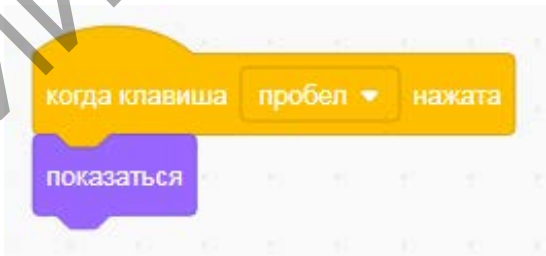

Рисунок 3.2.17 – Программирование восстановления объекта

– Команды задать для каждого собираемого объекта.

6. Выполните пятый этап: программирование препятствий

– Препятствия должны взаимодействовать с героем (аналогично собираемым объектам) (рис. 3.2.18).

– Варианты для взаимодействия: препятствие может менять костюм: «взрываться», «переворачиваться» (рис. 3.2.19), выдавать сообщение «Ты проиграл» или «Здесь ходить нельзя», можно задать команду, которая будет перемещать героя в начало игры (рис. 3.2.20).

– Варианты для взаимодействия препятствий придумать самостоятельно.

– В начале игры необходимо вернуть этим объектам исходный костюм.

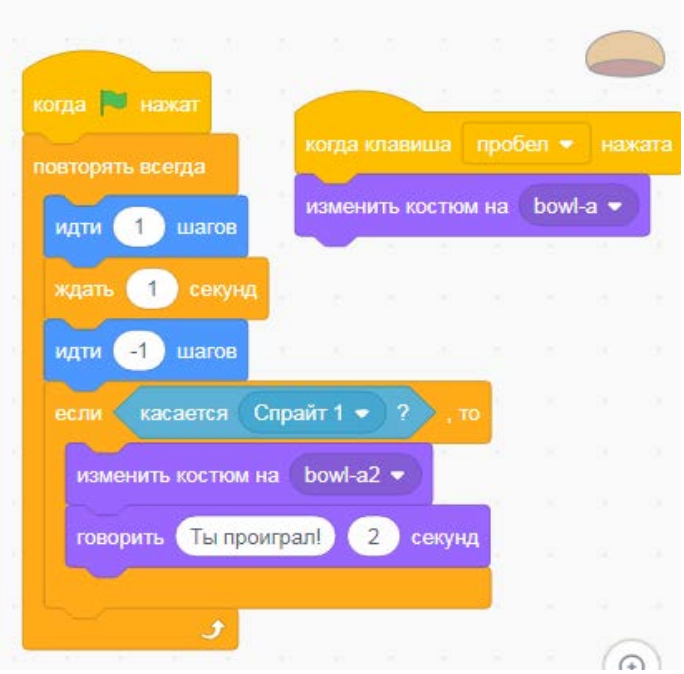

Рисунок 3.2.18 – Программирование движения объекта

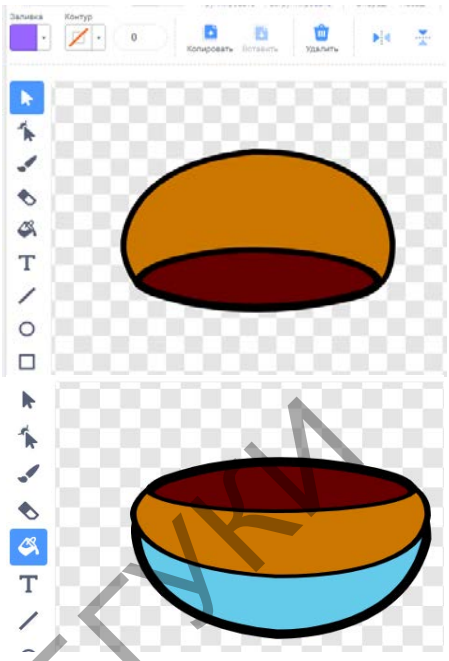

Рисунок 3.2.19 – Костюмы спрайта

Рисунок 3.2.20 – Программирование взаимодействия объекта

7. Выполните шестой этап: программирование кнопки финиш.
   – Создайте два костюма для нового спрайта – кнопка «финиш».
   – Аналогично предыдущим объектам напишите программу действий.
   – Реакция на прохождение «героя»: сменить костюм и сказать «Ты победил!» или «Переходи на второй уровень» (рис. 3.2.21).
8. Самостоятельно создать второй уровень игры.
9. Просмотреть результат и показать преподавателю.

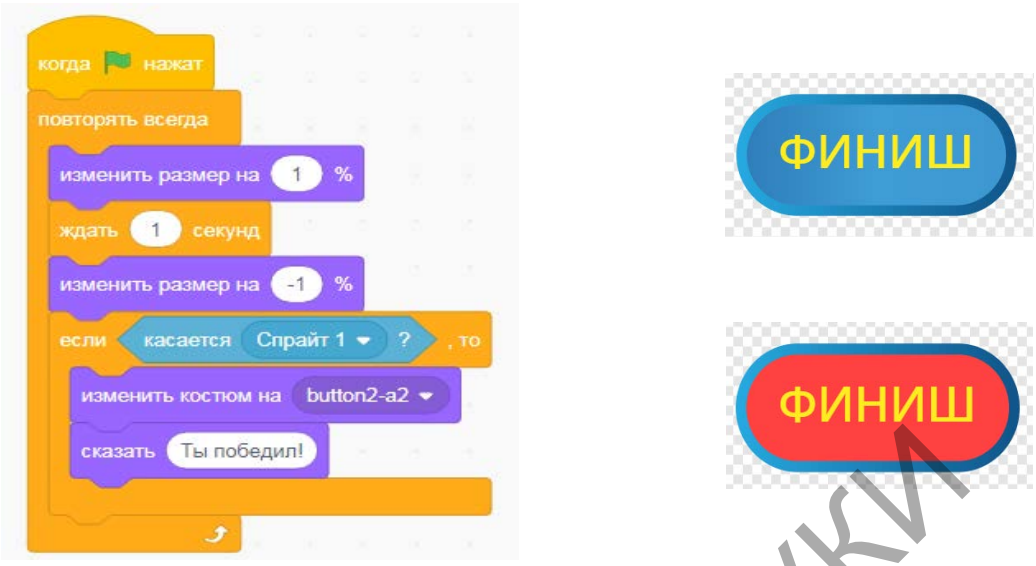

Рисунок 3.2.21 – Программирование объекта – кнопка «финиш»

Задание 5. Создайте авторский проект игры в Scratch.

Методические рекомендации по выполнению

1. Составить план (этапы) создания игры.
2. Создать спрайты или взять из библиотеки.
3. Подготовить фон.
4. Запрограммировать все объекты.
5. Результат показать преподавателю.

Тема 6. Программная среда разработки 3- D игр (6 часов)

Лабораторная работа 5. Знакомство с пакетом для создания трёхмерной графики и анимации Cinema 4D .

Цель. Приобрести основные знания работы для создания трехмерных объектов в пакете для создания трёхмерной графики и анимации Cinema 4D.

Задание 1. Рассмотрение интерфейса программы Cinema 4D.

Методические рекомендации по выполнению

1. Запустите программу Cinema 4D.

При первом входе в программу предъявляется диалоговое окно (окно доступно через главное меню Help – Personalize) для ввода регистрационных данных пользователя и регистрационных номеров каждого из модулей программы (обычно они прилагаются на диске) (рис. 3.2.22). Отложить регистрацию можно щелкнув по кнопке Cancel.

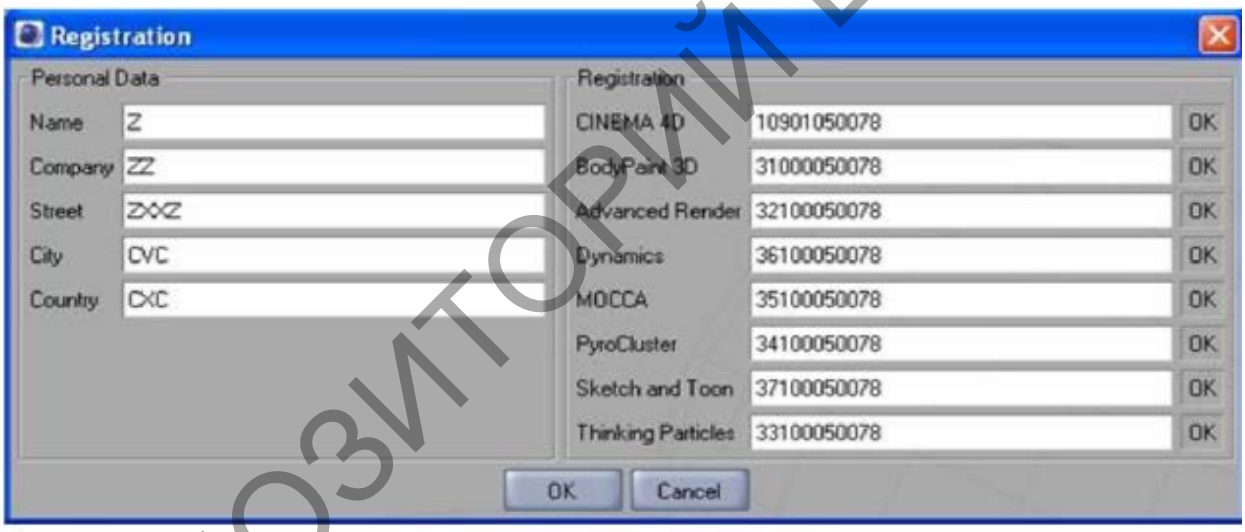

Рисунок 3.2.22 – Ввод регистрационных данных

2. Ознакомьтесь с интерфейсом.

– Cinema 4D имеет свободно настраиваемый интерфейс.

– Начальный вариант интерфейса при разрешении монитора 1024x768 (рис. 3.20.23).

– При меньшем разрешении монитора доступ к не поместившимся кнопкам осуществляется прокруткой панели инструментов (нажать левой или правой кнопкой мыши на разделитель между пиктограммами и протянуть влево/вправо, вверх/вниз).

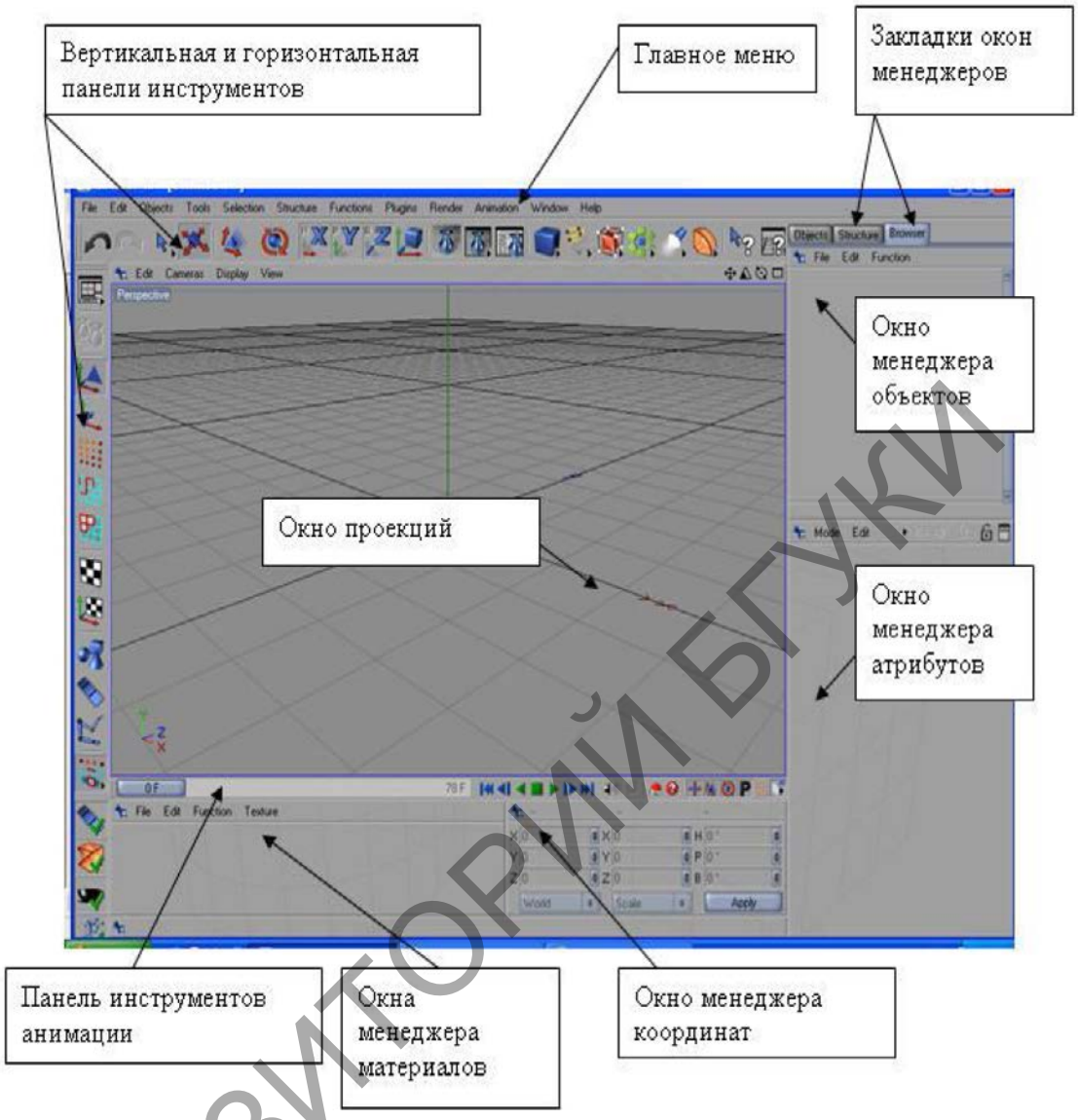

Рисунок 3.2.22 – Ввод регистрационных данных

3. Ознакомьтесь со способами отображения команд.

Чтобы изменить способ отображения команд на панелях инструментов, следует щелкнуть правой кнопкой мыши на панели инструментов и в контекстном меню выбрать один или сразу несколько из следующих пунктов: Icons (отображает значки), Text (отображает надпись) и Vertical (размещает надпись под значком). Настройка вертикальной и горизонтальной панелей инструментов производятся раздельно.

4. Ознакомьтесь с окнами проекции и их параметрами.

Окно проекции позволяет рассматривать создаваемый объект или сцену фронтально, сбоку, сверху и в перспективе. Соответственно этому Cinema 4D имеет 4 типа окон проекции, которые вызываются через меню View – Panels (Вид – Панели) (рис. 3.2.23).

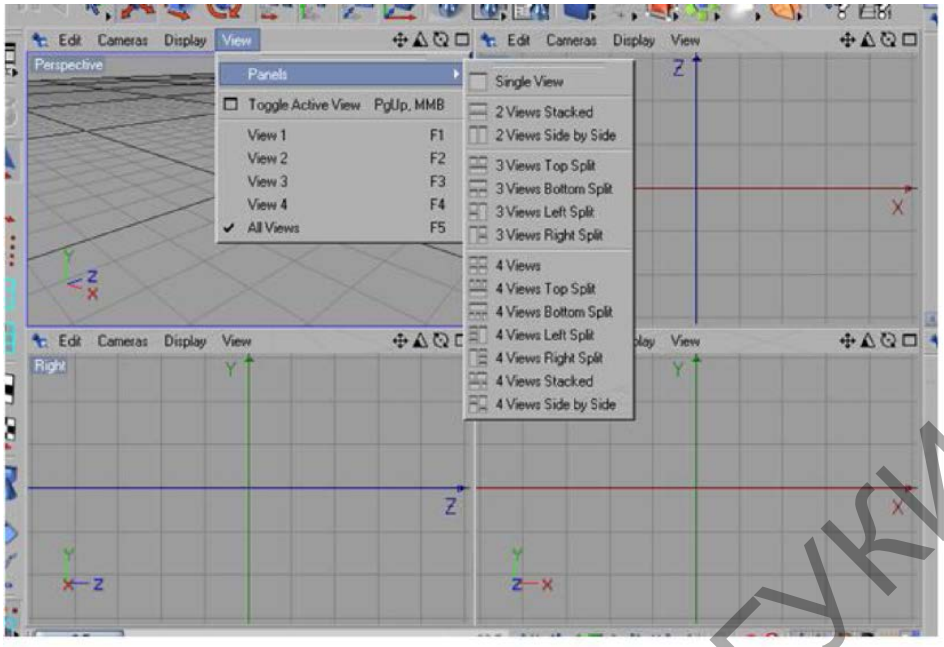

Рисунок 3.2.23 – Просмотр окон проекции

Каждое из окон проекции имеет: координатную сетку, оси глобальной системы координат, фон, заголовок и рамку. Можно выбрать разное количество, расположение и вид окон.

5. Ознакомьтесь с вкладкой меню Edit.

Для удобства работы размер объекта (или объектов) можно привести в соответствие с размером окна. Для этого в меню Edit предусмотрены следующие возможности:

– Frame Selected Elements – размеры выделенного элемента сцены становятся соизмеримы с размерами окна;

– Frame Active Object – размеры активного объекта становятся соизмеримы с размерами окна;

– Frame Scene Without Camera/Light – размеры всех объектов сцены, кроме камер и источников освещения, становятся соизмеримы с размерами окна;

– Frame Scene – размеры всех объектов, включая камеры и источники освещения, становятся соизмеримы с размерами окна;

– Frame Default – установка по умолчанию.

Кроме основных 4-х видов проекции объекта (фронтально, сверху, сбоку и в перспективе) можно использовать дополнительные виды, перечисленные совместно с основными в меню Cameras (камеры). Эти виды становятся доступны благодаря различному расположению камер, через которые ведется наблюдение за объектом. В основном новые виды проекций получаются как разновидности параллельного вида (Parallel), в котором отсутствуют перспективные сближения параллельных линий, создающих эффект глубины (перспективы).

6. Ознакомьтесь с параметрами вкладки меню Preferences.

В настройках программы (*Preferences*) содержаться разные каталоги с параметрами.

Каталог *Import/Export* (импорт/экспорт) содержит настройки для программ, из которых можно импортировать сцены в Cinema 4D и в которые можно импортировать сцены из Cinema 4D. Настройки позволяют производить импорт/экспорт отдельных составляющих ее элементов: текстуры, освещения и т. д.

Каталог *Texture Path* (размещение текстур) содержит 10 полей, в которые могут быть введены пути размещения на диске папок, хранящих текстуры для данной сцены.

Каталог *Units* (единицы измерения) содержит две группы настроек Units и Color Chooser. Группа Units позволяет (рис. 3.2.24):

– Display Units (отображать единицы измерения) выводить название используемых единиц измерения во всех числовых полях программы;

– Basic Units (базовые единицы измерения) выбрать единицу измерения;

– Use HPB System (использовать систему курс-тангаж-крен) заблокировать вращение объекта с помощью мыши (для опытных пользователей);

– Animation Units выбрать анимационную единицу (кадр, секунда, временной код SMPTE) для отсчета времени анимации.

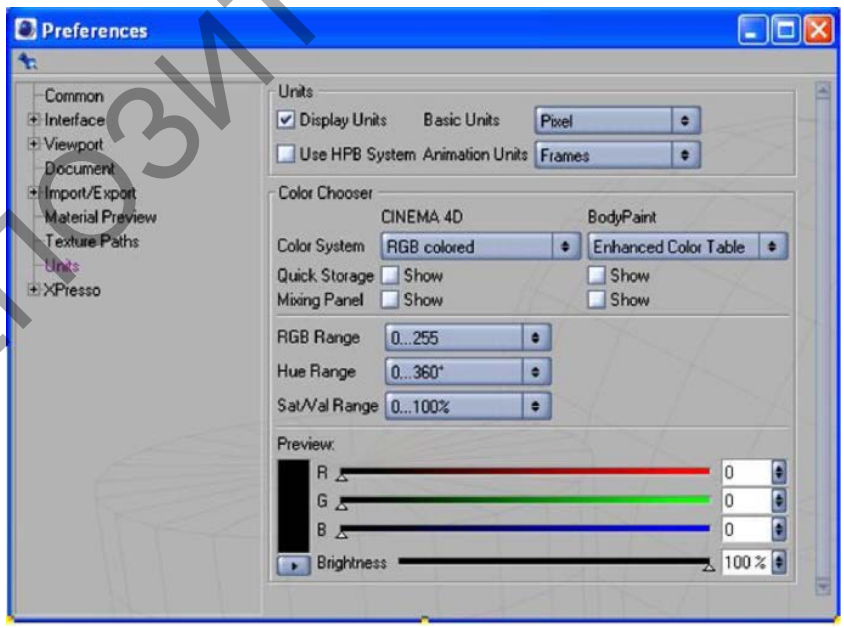

Рисунок 3.2.24 – Окно настройки параметров

Группа *Color Chooser* позволяет выбрать и настроить цветовую систему. RGB (красный – зеленый – синий) или HSV (оттенок – насыщенность – значение).

Каталог *XPresso* содержит настройки цвета для элементов интерфейса окна редактора узлов XPresso.

Для настройки частоты смены кадров при анимации (параметр Frame Rate), задания номера начального кадра (Minimum) и конечного кадра анимации (Maximum), а также задания уровня детализации (Level of Detail) при отображении объекта служит окно *Project Settings* (настройки проекта). Окно вызывается через главное меню Edit – Project Settings.

Флажок Use Render LOD of Editor Rendering позволяет использовать степень детализации, указанную в настройках визуализации для объекта.

Это окно *Render Settings* (настройки визуализации) вызывается через главное меню Render – Render Settings (визуализация – настройки визуализации).

Задание 2. Рассмотрение интерфейса программы Cinema 4D.

Методические рекомендации по выполнению

1. Ознакомьтесь с параметрическими примитивами.

Параметрическими примитивами в программе называются объекты, которые описываются математическими уравнениями и изначально заданы в параметрическом виде. В большей своей части примитивы задают базовые геометрические формы (сферу, цилиндр, конус, куб, пирамиду, тор и др.), но включают также манекен (Figure) для наглядного изучения кинематики, рельеф, плоскость. Всего программа содержит 16 примитивов.

2. Создайте примитив.

– Пиктограмма на верхней горизонтальной панели инструментов Add Cube Object (добавить куб) (рис. 3.2.25).

– Список примитивов через меню Objects – Primitive (рис. 3.2.26).

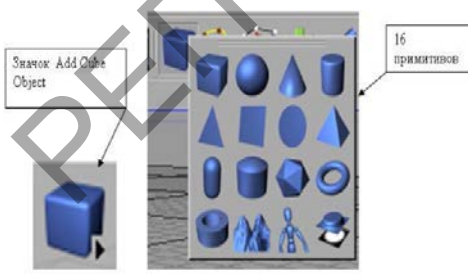

Рисунок 3.2.25 – Примитивы на панели инструментов

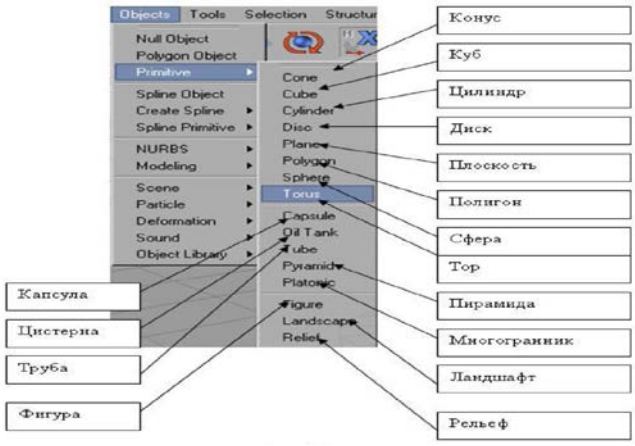

Рисунок 3.2.26 – Список примитивов в меню

3. *Выберите примитив (Torus).*

В соответствии с настройками интерфейса примитив появился в центре окна проекций. В окне менеджера объектов в левом поле появилось имя Torus и значок тора синего цвета, указывающий, что примитив представлен в параметрическом виде (рис. 3.2.27).

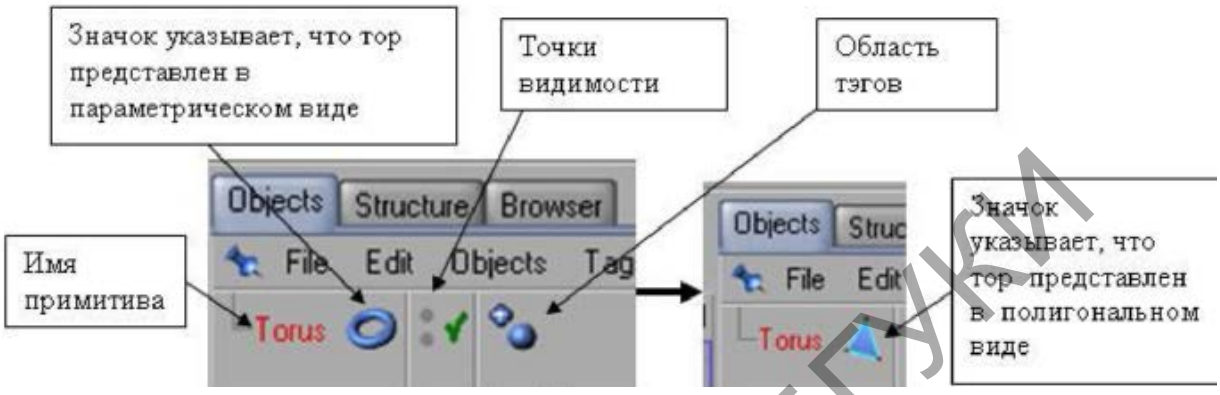

Рисунок 3.2.27 – Отображение примитива в окне менеджера объектов

4. Переведите примитив в полигональный вид.

Выберите в меню Functions – Make Editable или нажмите латинскую клавишу C. Значок изменится на синий треугольник – одинаковый для всех примитивов, представленных в полигональном виде.

5. Рассмотрите настройки видимости объекта.

Две точки позволяют управлять видимостью объекта в окне проекций (верхняя точка) и в окне рендеринга (нижняя точка). Чтобы сделать объект невидимым, точка должна быть красного цвета. Для этого щелкните по ней левой кнопкой мыши. Чтобы объект был видимым, точка должна быть зеленого цвета или дезактивирована. Проверьте действие точек видимости на торе. Чтобы вызвать окно рендеринга, в главном меню (имя Torus должно быть выделено) выберите Render-> Render View.

6. Рассмотрите область тегов объекта.

В правой части указываются значки тэгов – программ, предоставляющих дополнительные свойства объектам. В нашем случае, это тэг черно-белого тонировщика по Фонгу (Phong), который присваивается примитивам по умолчанию. Тонировщик создает впечатление объема тора, распределяя светотень по его поверхности. Источник освещения по умолчанию находится слева сзади от пользователя, смотрящего на экран монитора (вернее, от камеры, через объектив которой мы видим примитив в окне проекций). Тэгов у объекта может быть несколько, и они играют важную роль в работе с объектами.

7. Рассмотрите свойства объекта в окне атрибутов.

Щелкните левой кнопкой мыши по имени объекта *Torus*. В окне менеджера атрибутов будут выведены его свойства (рис. 3.2.28).

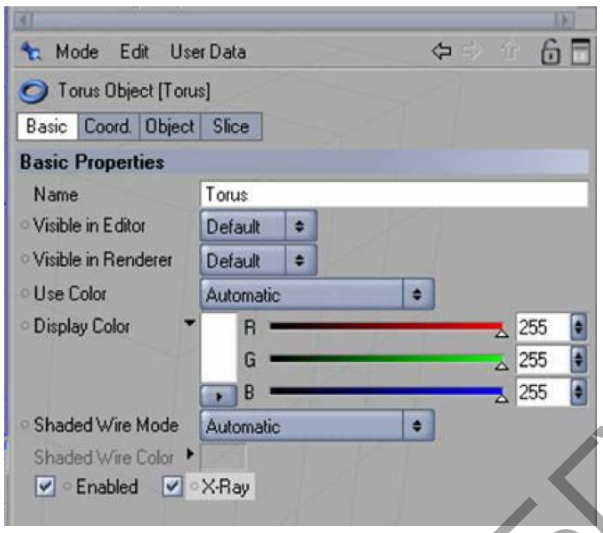

Рисунок 3.2.28 – Свойства объекта в окне менеджера атрибутов

8. Рассмотрите параметры объекта в окне атрибутов.

Представление примитивов в параметрическом виде позволяет легко изменять такие их параметры, как, например, радиус, высоту и т. п. на вкладке Object.

9. Рассмотрите возможности пластического изменения формы.

Для пластического изменения формы поверхности примитива (продавливание поверхности, вытягивание, скручивание и т. п.) к примитиву следует применить специальные инструменты деформации, либо примитив надо преобразовать в полигональный вид и применить к нему инструменты работы с вершинами, ребрами и полигонами.

Выделите имя примитива, а затем выполнить одно из действий:

- нажать латинскую клавишу C;
- выбрать в главном меню Functions – Make Editable.

10. Рассмотрите работу с полигонами объекта.

В этом режиме станут доступны основные структурные элементы примитива:

– вершины (Vertex);

– ребра (грани) (Edges), получаемые при соединении вершин между собой;

– полигоны (Polygons) – треугольники или четырехугольники, получающиеся при соединении трех или четырех граней.

С выделенными элементами объекта можно работать: перемещать их, масштабировать, вращать, то есть, изменить вид объекта, смоделировать из поверхности примитива нужные формы (рис. 3.2.29).

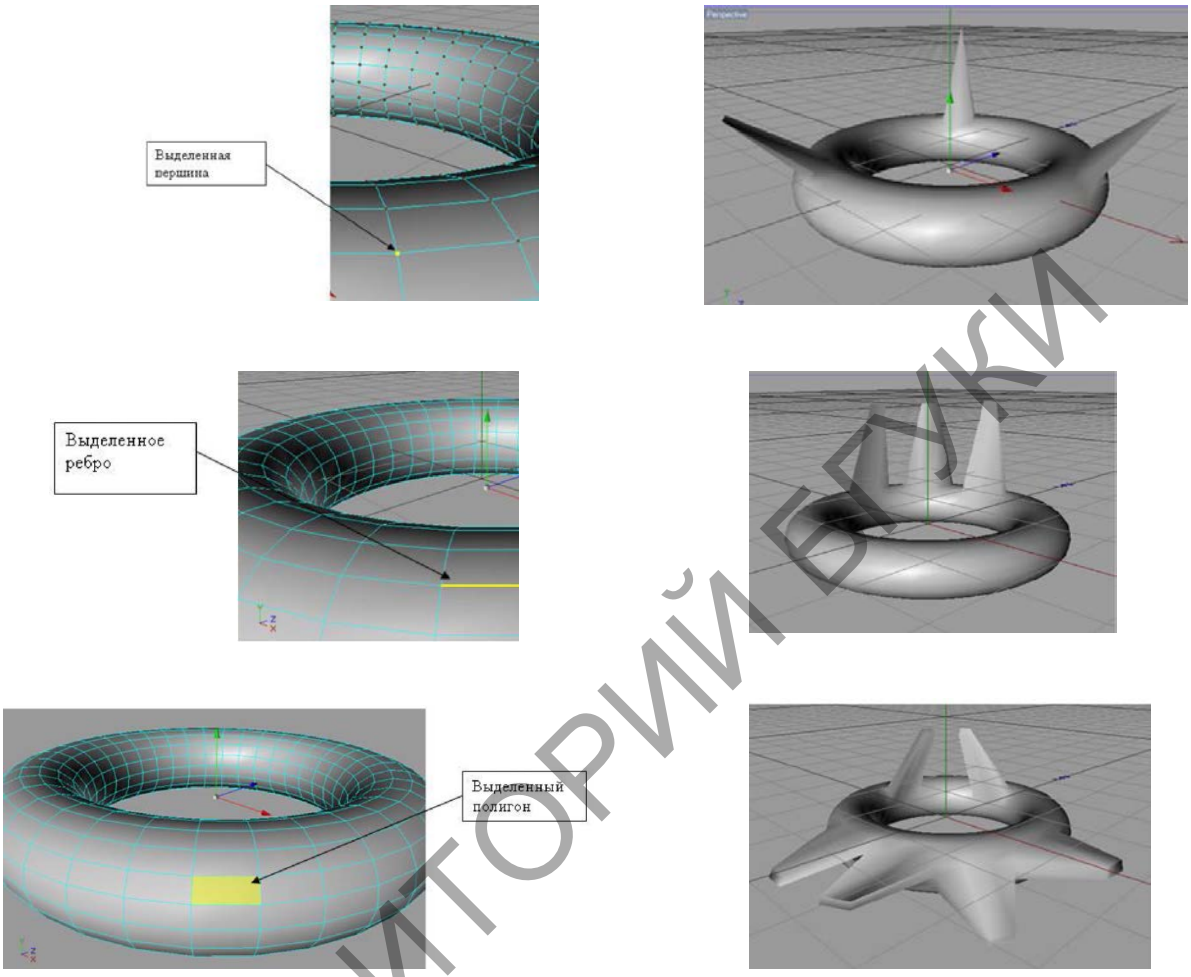

Рисунок 3.2.29 –Перемещение вершин, ребер и полигонов

11. Самостоятельно изучите другие окна с инструментами для работы с объектами (рис. 3.2.30).

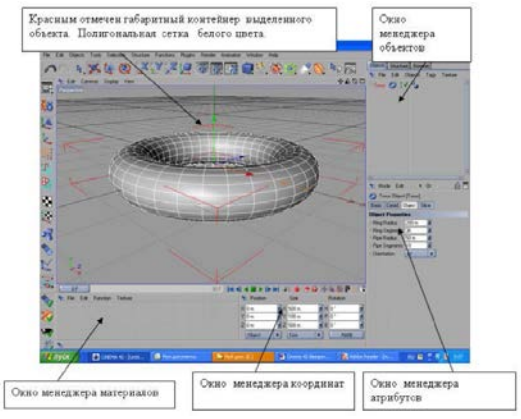

Рисунок 3.2.30 –Инструменты для работы с объектами

12. *Окно менеджеров координат* (рис. 3.2.31).

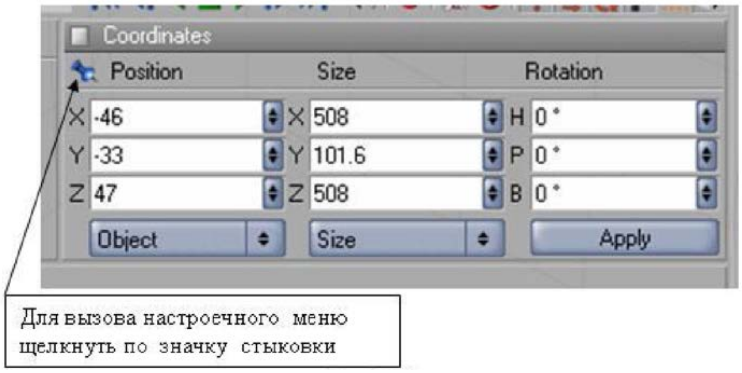

Рисунок 3.2.31 –Окно менеджера координат

13. Выделение объектов.

Для выделения всех объектов сцены служит пункт меню Edit – Select All или сочетание клавиш Ctrl+A. Выделять объекты можно также, выделяя их имена в окне менеджера объектов.

14. Инструменты навигации.

В правом верхнем углу окна проекции расположены пиктограммы инструментов навигации (рис. 3.2.32).

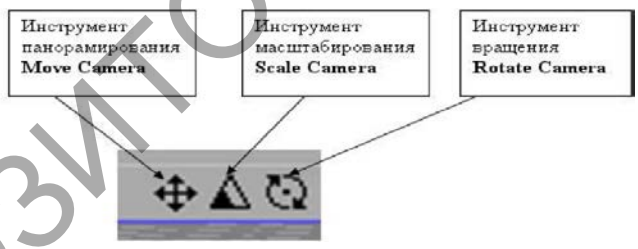

Рисунок 3.2.32 –Инструменты навигации

Для пользования инструментом надо щелкнуть на нем левой кнопкой мыши и, удерживая кнопку, перемещать мышь по пространству окна проекций. Возможны три способа навигации:

– Панорамирование – проекция объекта перемещается в плоскости, параллельной плоскости экрана (удерживать нажатой клавишу 1 и нажав любую кнопку мыши или колесико, двигать указателем мыши);

– Масштабирование – проекция объекта приближается или удаляется (удерживать клавишу 2 + любая кнопка мыши; перемещать указатель мыши влево для уменьшения, вправо для увеличения размера проекции);

– Вращение – вращение объекта или группы объектов вокруг их геометрического центра (удерживать клавишу 3 + любую кнопку мыши или колесико, перемещать указатель мыши по окну проекций).

Перемещать примитив по окну проекций вправо, влево, вверх, вниз при неподвижной координатной сетке: выделить имя примитива в окне менеджера объектов и, удерживать клавишу 4 и левую кнопку мыши, двигать указатель мыши по окну проекций.

Удалять и приближать объект – те же действия + нажатая клавиша 5. Для вращения – нажатая клавиша 6.

Задание 15. Создайте сцену с деревьями в Cinema 4D.

1. Создать куб с параметрами 50x425x50 (рис. 3.2.33).
2. Скопировать куб, уменьшить параметры, развернуть (рис. 3.2.34).
3. Добавить еще ветки, развернуть, разместить вдоль ствола (рис. 3.2.35).
4. Создать сферу, скопировать и расставить по веткам (рис. 3.2.36).
5. Уменьшить полигоны у сфер, отключить тег фонг, сгруппировать сферы в объект «0».
6. Выбрать деформер (Polygon Reduction) (рис. 3.2.37).
7. Применить для группы сфер (рис. 3.2.38).
8. Создать нулевую группу для дерева.
9. Создать цилиндр 20x200, убрать фонг тег, уменьшить циклические сегменты до 7.
10. Создать сферу радиус – 150 (рис. 3.2.39).
11. Выбрать деформер (Taper) (рис. 3.2.40).
12. Применить к сфере (рис. 3.2.41).
13. Подобрать параметры (рис. 3.2.42).
14. Применить деформер изменения полигонов у сферы (рис. 3.2.43).
15. Создать цилиндр 40x450, убрать фонг тег, уменьшить циклические сегменты до 7 (рис. 3.2.44).
16. Создать конус с параметрами 80x250x280x8x8 (рис. 3.2.45).
17. Удалить тег фон, создать две копии конуса, уменьшить параметры, сдвинуть вверх. Объединить в нулевой объект (рис. 3.2.46).
18. Создать ещё два дерева по аналогичному алгоритму (рис. 3.2.48).
19. Выбрать инструмент Enabie Axis (редактирования осей) (рис. 3.2.47).
20. Спозиционировать все объекты относительно пола (вид спереди).
21. Выйти из режима редактирования осей.
22. Создать материалы: зеленого цвета – для лиственных деревьев, темно зеленый – еловые (снять рефлект), коричневый – для стволов. Задать название материалам.
23. Сохранить и показать преподавателю.

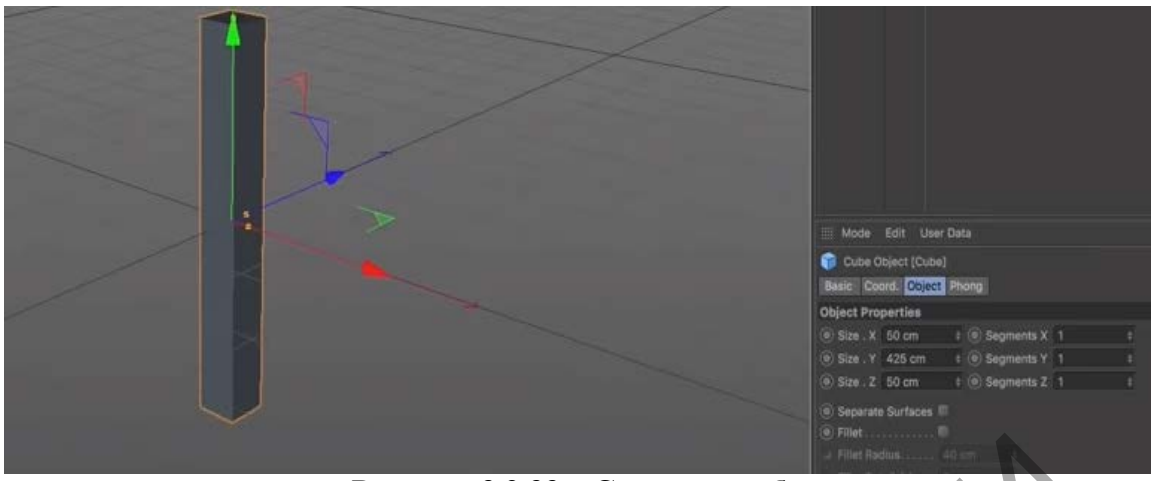

Рисунок 3.2.33 – Создание куба

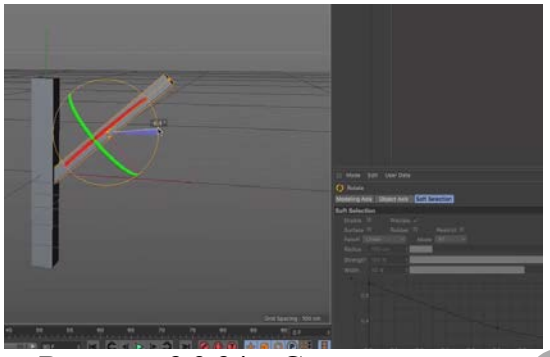

Рисунок 3.2.34 – Создание ветки

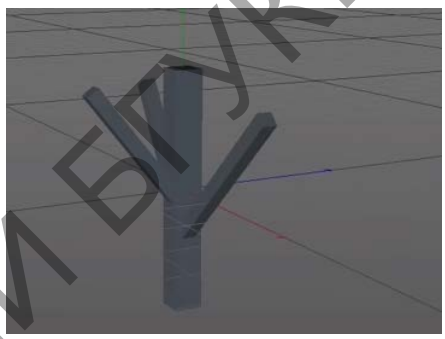

Рисунок 3.2.35 – Размещение веток

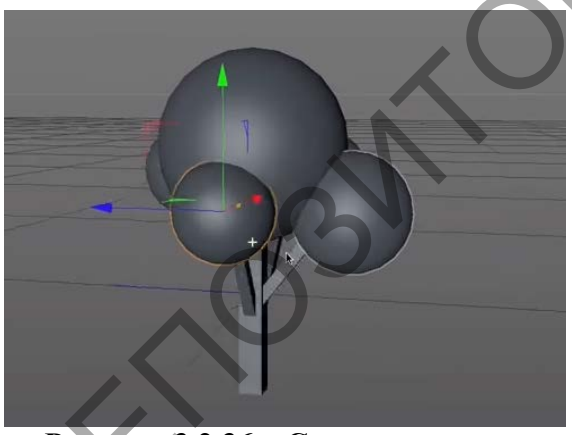

Рисунок 3.2.36 – Создание кроны

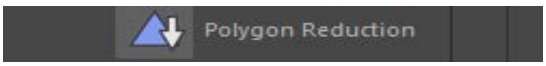

Рисунок 3.2.37 – Деформер (Polygon Reduction)

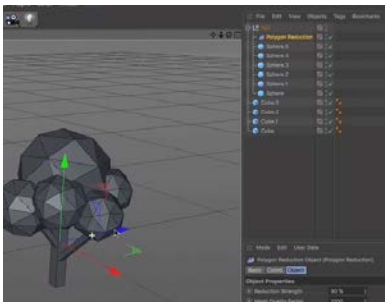

Рисунок 3.2.38 – Применение деформера

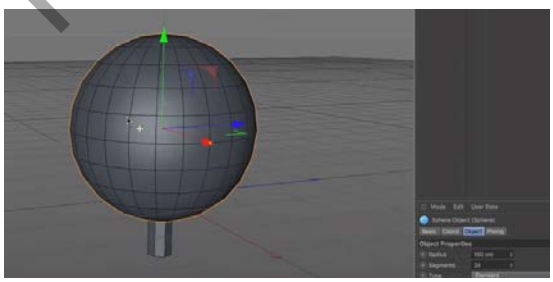

Рисунок 3.2.39 – Создание сферы

Рисунок 3.2.40 – Деформер Taper

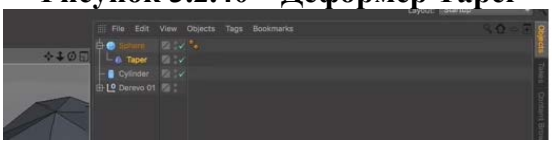

Рисунок 3.2.41 – Назначение деформера

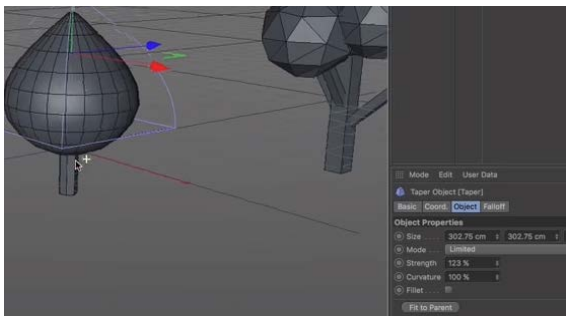

Рисунок 3.2.42 – Подбор параметров деформера

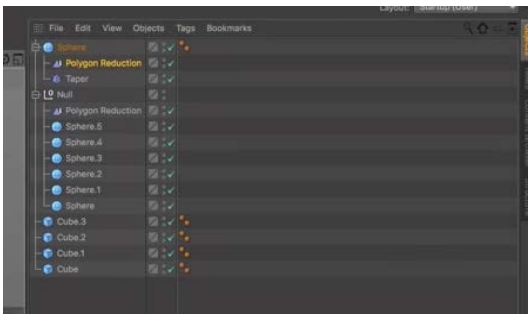

Рисунок 3.2.43 – Назначение деформера полигонов сферы

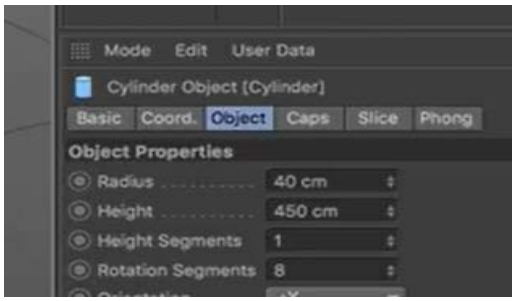

Рисунок 3.2.44 – Создание цилиндра с параметрами

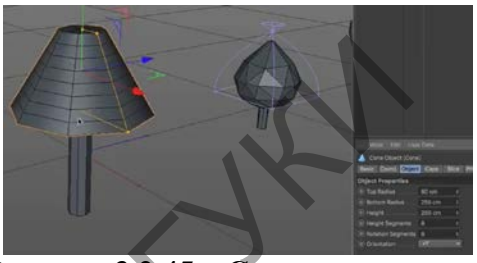

Рисунок 3.2.45 – Создание конуса с параметрами

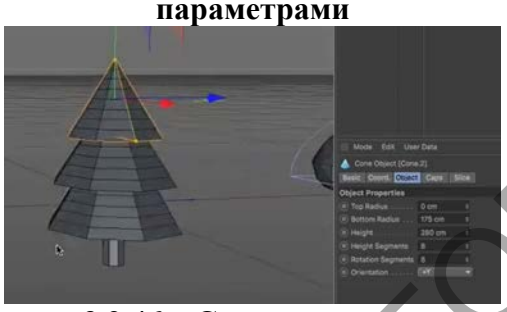

Рисунок 3.2.46 – Создание ели из конусов

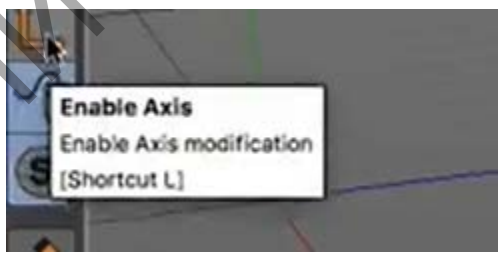

Рисунок 3.2.47 – Редактирование осей

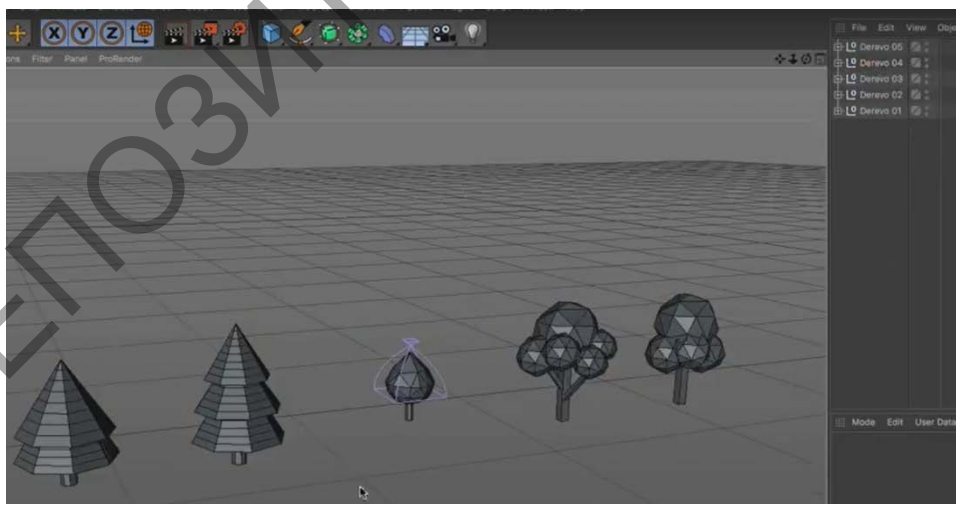

Рисунок 3.2.48 – 3-D модели деревьев в программе Cinema 4D

Тема 6. Программная среда разработки 3- D игр (6 часов)

Лабораторная работа 6. Знакомство со средой разработки компьютерных игр Unity. Создание 3- D игр (4 часа).

Цель. Приобрести основные знания по работе в среде разработки компьютерных игр Unity. Приобрести основные навыки по созданию игр.

Задание 1. Ознакомиться с интерфейсом программы Unity.

Методические рекомендации по выполнению

1. Ознакомиться с общим видом программы.

Рабочее окно Unity разбито на 6 взаимосвязанных областей (рис. 3.2.49). (По умолчанию в Unity включается вид «Default» или «Tall». Если у вас другой вид рабочего окна, то можете переключить его в меню «Layout» в верхнем правом углу окна).

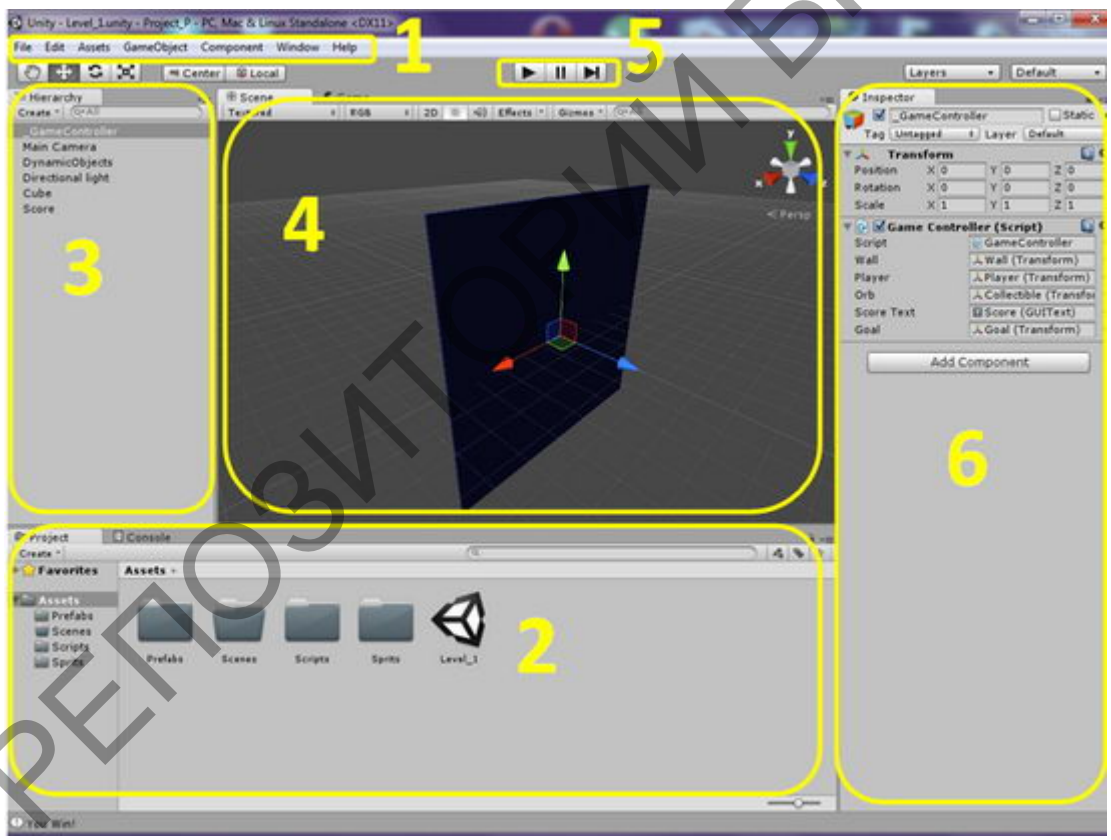

Рисунок 3.2.49 – Рабочие области окна Unity

1. Главное меню (Main menu) – располагаются все команды, доступные в программе. Многие команды продублированы кнопками и меню в рабочих областях.

2. Обзор проекта (Project View) – список, в котором показаны все используемые файлы в игре: файл сцены, файл программного кода, графические и аудиофайлы.

3. Иерархия (Hierarchy) – список, где перечислены все объекты, добавленные на сцену. Здесь можно работать с объектами, копировать их, переименовывать, удалять.

4. Сцена (Scene) – область, где отображается игровой мир или игровая сцена. Здесь можно добавлять новые объекты, перетаскивать их, менять вид.

5. Игра (Game) – область предпросмотра, где видно, как сцена будет выглядеть в игре. Здесь можно настраивать различные настройки экрана и видеорежима.

6. Инспектор (Inspector) – список, состоящий из нескольких различных по виду разделов. Показывает все свойства выбранного объекта: размеры, модели, текстуры, скрипты.

2. Самостоятельно ознакомиться с вкладками Главного меню.
3. Рассмотреть область проекта (Project View) (рис. 3.2.50).

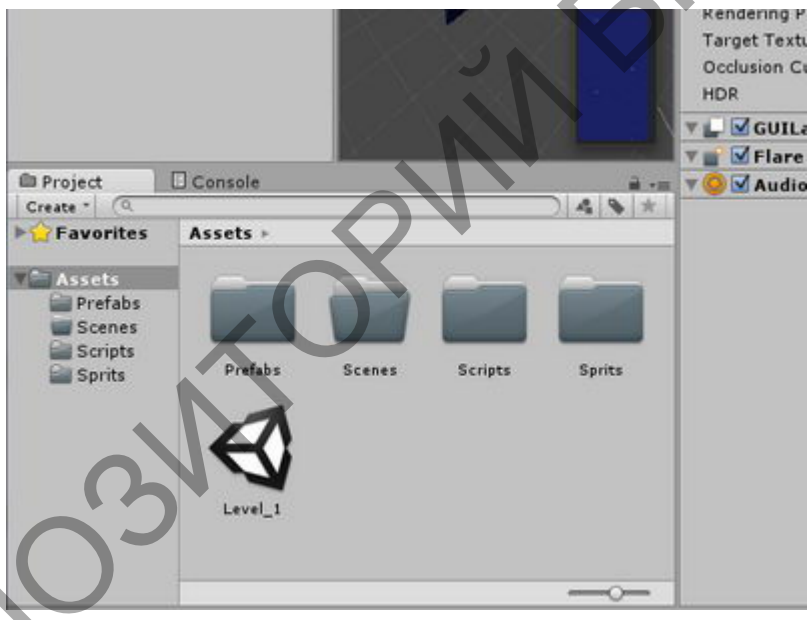

Рисунок 3.2.50 – Рабочая область Проекта

Каждый проект содержит папку «Assets», в которой представлены ресурсы игры: файлы-скрипты, 3D-модели, текстуры, аудиофайлы, префабы (объекты, которые можно клонировать).

Добавить новый ресурс в проект можно двумя способами:

- перетащить файл из Проводника в Project View.

- команда «Assets» > «Import New Assets».

Проекты игры состоят из одного или нескольких файлов сцены. Каждая отдельная сцена – это отдельный уровень игры.

Для создания ресурса используется выпадающее меню «Create» в «Project View» или (ПКМ) – «Create» (скрипты, префабы, папки и прочее).

Переименовать ресурс или папку можно, нажав «F2» или сделав два клика по имени. Если зажать «Alt», то при раскрытии директории будут раскрыты и все поддиректории.

4. Рассмотреть область Иерархии (Hierarchy) (рис. 3.2.51).

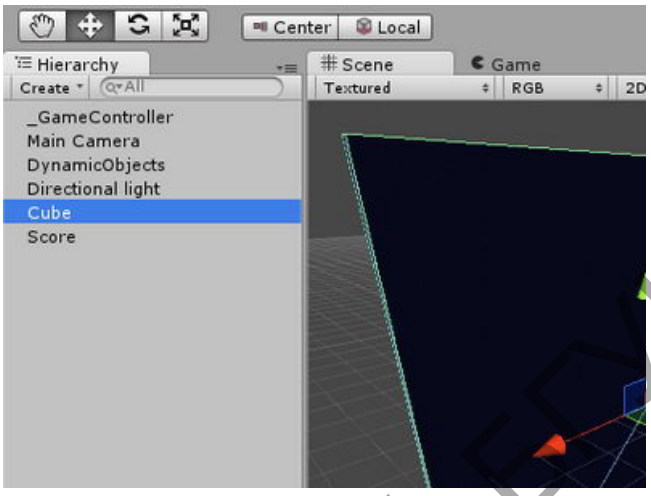

Рисунок 3.2.51 – Рабочая область Иерархия

Иерархия содержит все объекты (GameObject) открытой сцены.

Объектам можно задавать наследование (Parenting) – являться дочерним по отношению к другому. Дочерний объект будет двигаться и вращаться вместе с родителем. Для создания дочерней связи достаточно перетащить объект на «родителя» в Hierarchy.

5. Рассмотреть область Сцена (Scene View) (рис. 3.2.52).

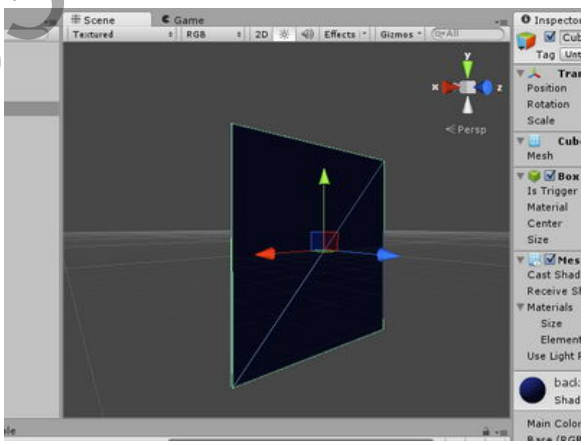

Рисунок 3.2.52 – Рабочая область Сцена

Игровая сцена используется для расстановки объектов на уровне (окружение, персонажи, камеры, системы частиц и прочее). Сцена может являться игровым уровнем, главным меню, заставкой.

Приёмы для перемещения по сцене в 3D режиме:

– Зажатая правая кнопка мыши активирует режим свободного полёта.

– Перемещаться можно клавишами WASD.

– Выбрать объект из списка иерархии и нажать «F». Вид сцены будет центрирован и масштабирован по выбранному объекту.

– При зажатой клавише «Alt», левая клавиша мыши будет крутить камеру вокруг текущей точки опоры.

– При зажатой клавише «Alt», СКМ будет перемещать камеру.

– При зажатой клавише «Alt», правая кнопка мыши будет масштабировать вид сцены.

– Альтернативный режим перемещений — клавиша «Q».

6. Рассмотреть область Игровой вид (Game View) (рис. 3.2.53).

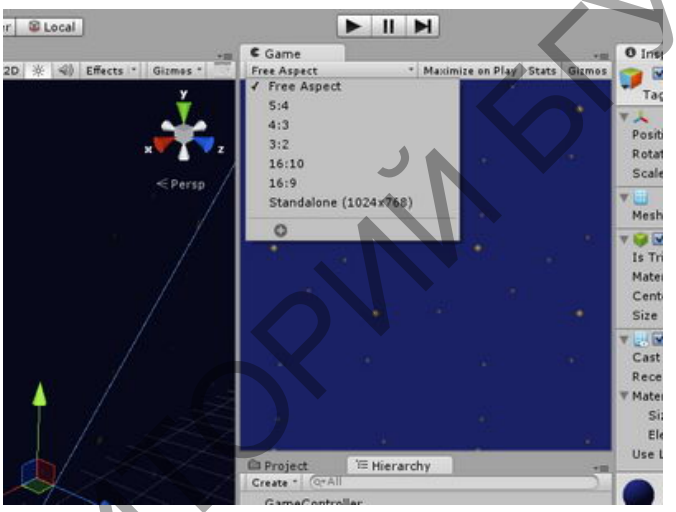

Рисунок 3.2.53 – Рабочая область Игровой вид

Game View — предпросмотр игры (рендер из игровой камеры).

В расположении окон по умолчанию «Игровой вид» отсутствует, для его включения нужно выбрать вкладку «Game» над игровой сценой. В других режима окон «Игровой вид» занимает своё отдельное место.

– Play Mode – три кнопки в верхней части окна, которые отвечают за управление предпросмотром игры: «Play», «Pause» и «Step». Все изменения, произведённые во время предпросмотра, сбрасываются при выходе из него.

– Control Bar – контроль пропорций изображения (Aspect Drop-down). На некоторых дисплеях это соотношение отличается от стандартного 4:3 (например, на широкоформатных мониторах — 16:10).

– Maximize on Play – раскрывает окно предпросмотра на всё окна редактора.

– Gizmos – включает отображение контейнеров гизмо в «Game View».

– Stats – показывает статистку рендеринга (Rendering Statistics), полезную при оптимизации.

7. Рассмотреть область Инспектор (Inspector) (рис. 3.2.54).

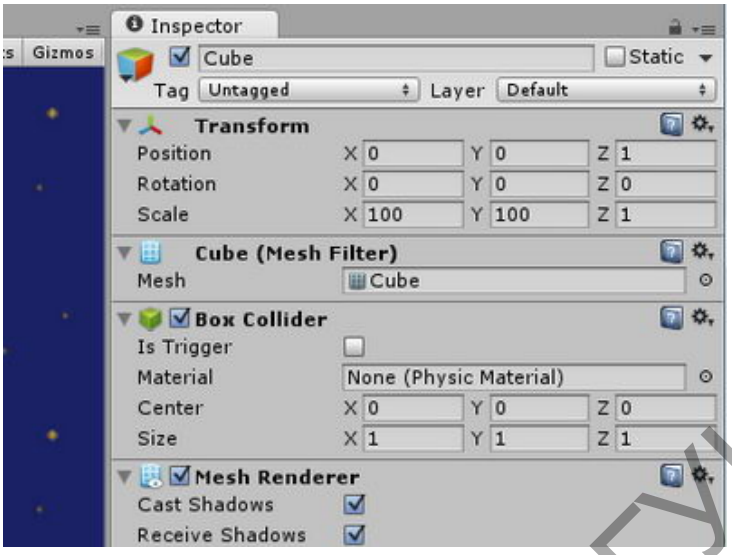

Рисунок 3.2.54 – Рабочая область Инспектор

GameObject – это любой объект в игре со специальными свойствами. *Объект* – это лишь контейнер, который содержит в себе различные компоненты (Components).

Объект может содержать в себе такие типы компонентов: расположение в пространстве (Transform), меши (meshes), скрипты (scripts), звуки, ИС (Lights) и другие элементы.

Status Bar – строка состояния расположена в нижней части окна редактора и отображает ошибки компиляции. Двойное нажатие вызовет окно консоли (Console), в котором отображаются все ошибки.

8. Рассмотреть настройки рабочего окна.

Расположение (Layout) секторов можно настраивать, перетаскивая их за закладки. Если перетащить закладку в область закладок уже существующего окна, то она будет добавлена к присутствующим там закладкам.

Закладки могут открепляться от главного окна редактора и включаться в состав плавающего окна редактора. Плавающее окно может содержать сектора и закладки так же, как и главное окно, но не имеет панели инструментов (Toolbar).

Когда расположение секторов задано, его можно сохранить и загрузить в нужный момент через выпадающее меню Layout (Save и Load).

Правым кликом по закладке можно вызвать меню с дополнительными возможностями: максимизация сектора или добавление новых закладок.

Задание 2. Создать новый проект в Unity.

Методические рекомендации по выполнению

1. Выбрать Файл – Создать проект («File – New Project»).
2. Выбрать папку для сохранения, например: C:/Project_1.
3. Выбирать двухмерную игру «Setup defaults for: 2D».
4. Нажать кнопку «Create» (создать).

Задание 3. Создать иерархию проекта (рис. 3.2.55).

Методические рекомендации по выполнению

1. В окне «Project» нажать кнопку «Create».
2. Из списка выбрать «Folder» (папка). В корневой папке «Assets» появится новая папка.
3. Создать четыре папки: Prefabs (префабы – массивы объектов), Scripts (скрипты – программный код), Sounds (звуки – для звукового сопровождения игровых действий), Sprites (спрайты – изображения для игровых объектов).

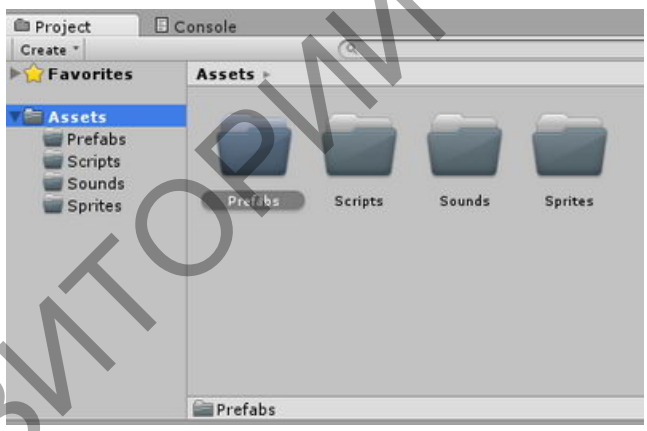

Рисунок 3.2.55 – Настройка проекта

Задание 4. Создать игровую сцену.

Методические рекомендации по выполнению

1. Подготовить игровые объекты – космический корабль, космос, препятствие, выстрел (рис. 3.2.56).

Рисунок 3.2.56 – Изображения, необходимые для создания игры

Создать спрайты можно самостоятельно в любой программе (без фона) или скачать с официального сайта https://unity3d.com/ru.

2. Загрузить объекты в сцену (можно 2 способами).

– Перетащить файл изображения в окно Юнити в зону «Project – Sprites». Выделить загруженный файл, в окне «Inspector» в строчке «Texture Type» выбрать значение «Sprite (2D and UI)». Нажать «Apply» (применить).

– Нажать правой кнопкой мыши по папке «Sprites» («Project»), выбрать «Import New Asset», выбрать файл изображения.

3. Выбрать загруженный файл в зоне «Project», перетащить его в зону «Scene».

4. В зоне «Inspector» нажать правой кнопкой мыши по меню «Transform» (координаты объекта в игровом пространстве), выбрать «Reset Position». Объект встанет ровно по центру мира (X=0, Y=0) (рис. 3.2.57).

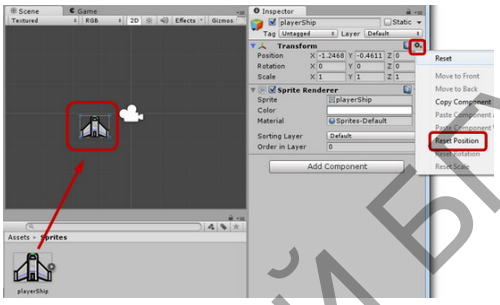

Рисунок 3.2.57 – Назначение координат объекту

5. Загрузить изображение фона (размер – 100 x 100, формат – .png).

6. Изменить расположение объектов (рис. 3.2.58):

– На сцене выбрать объект фона, в окне «Inspector» – свойства «Sprite Renderer» – строка «Sorting Layer» выбрать «Add Sorting Layer».

– Название слоя «Layer 1» переименовать в «Background».

– Создать ещё два слоя: «Foreground» и «GUI».

– Установить для объекта корабля в строчке «Sorting Layer» слой «Foreground».

– Для объекта фона – слой «Background».

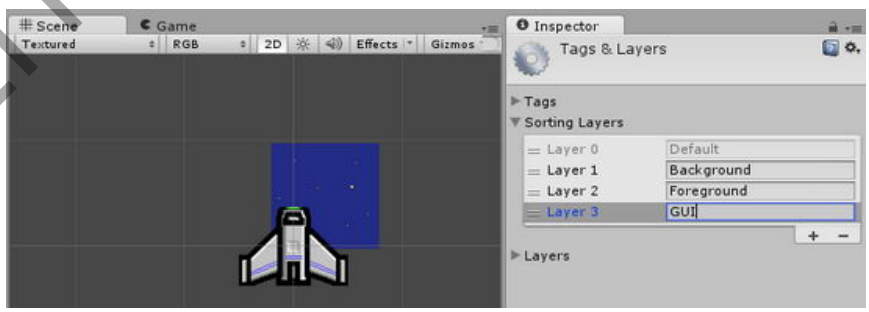

Рисунок 3.2.58 – Назначение слоев объектов

Задание 5. Перевести спрайт в текстуру.

Методические рекомендации по выполнению

1. Превратить спрайт в текстуру.

2. Удалить фон из списка «Hierarchy».

3. В зоне «Project» выбираем изображение фона, в окне «Inspector» меняем его тип «Texture Type» на «Texture».

4. В строчке «Wrap Mode» ставим значение «Repeat». Нажимаем кнопку «Apply».

5. В главном меню выбрать строчку «Game Object | Create Other | Cube».

6. Изменить имя появившегося объекта на «Background».

7. В свойствах «Transform» изменяем расположение «Position: 0, 0, 1» и размер «Scale: 100, 100, 1».

8. Удалить в свойствах куба раздел «Box Collider» (обработчик столкновений). Для этого нажать правой кнопкой мыши на разделе, в появившемся меню выбрать строчку «Remove Component».

9. В зоне «Project» сверху нажать «Create | Material», назвать «BackgroundMaterial».

10. В свойствах «Shader» выбрать «Unlit | Texture». В правой части свойств определить изображение текстуры «Texture box».

11. В свойстве «Tiling» выставить значения x = 25, y = 25 (рис. 3.2.59).

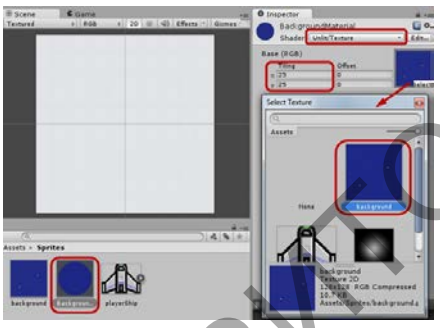

Рисунок 3.2.59 – Добавление текстуры

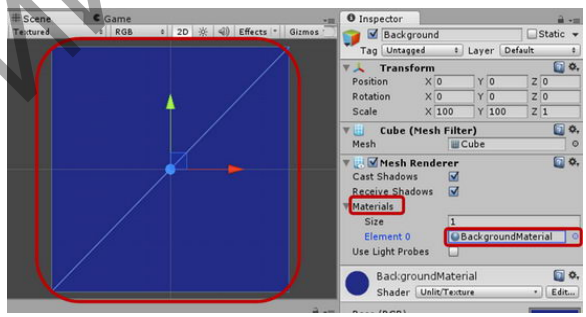

Рисунок 3.2.60 – Назначение материала

12. В зоне «Hierarchy» выбираем объект «Background». В его свойствах, под компонентом «Mesh Renderer» открываем «Materials» и меняем значение «Element 0» на наш материал «BackgroundMaterial».

Задание 6. Написание скриптов.

Методические рекомендации по выполнению

Скрипты – это своеобразные логические команды, которые предписывают объектам, как им себя вести в той или иной ситуации. В Unity можно писать скрипты на следующих языках программирования: C#, Boo, UnityScript. Мы будем использовать язык C#, так как он принадлежит к серии самых популярных языков программирования.

1. Создайте скрипт.

– В области «Проекта» нажать правой кнопкой мыши на папке

«Scripts», выбрать «Create» — «C# Script». Назвать созданный файл — «PlayerScript» (рис. 3.2.61).

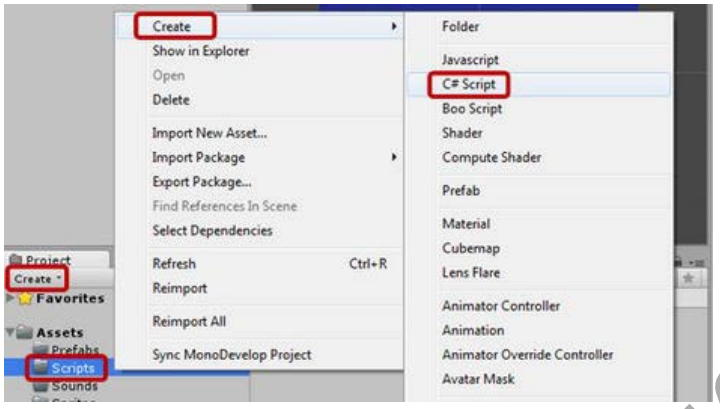

Рисунок 3.2.61 – Создание скрипта

– Двойным щелчком откройте файл скрипта.

2. Ознакомиться с кодом скрипта.

– «MonoDevelop» - дополнительной программа из комплекта Unity, предназначенная для написания скриптов (рис. 3.2.62).

– Часть программного кода создается автоматически.

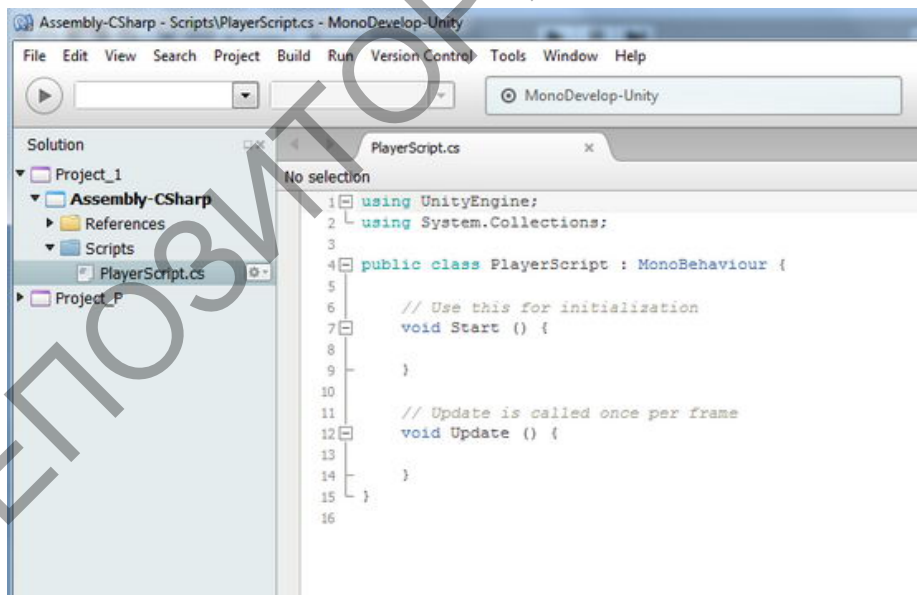

Рисунок 3.2.62 – Окно программы «MonoDevelop»

– Первые две строчки:

using UnityEngine;

using System.Collections.Generic;

Команда «*using*» подключает библиотеки описания классов, которые используют в программном коде.

В библиотеке «UnityEngine» содержатся описания всех стандартных объектов внутри движка Unity (объекты, их свойства, файлы, префабы, система наследования, связи между объектами).

В библиотеке «System.Collections.Generic» содержатся простейшие логические конструкции (классы, списки, перечни, массивы, таблицы, векторы), а так же источники внешних данных для нашей будущей игры (нажатие клавиш клавиатуры, кнопок мыши, свойства экрана).

– Четвертая строчка: public class PlayerScript : MonoBehaviour {

Это заголовок созданного скрипта, который включает название скрипта («PlayerScript») и класс скрипта («MonoBehaviour» – стандартный класс для всех скриптов Unity).

После символа «{» начинается перечень команд внутри скрипта. В самой последней строчке скрипт обязательно должен завершится символом «}».

– Внутри скрипта видим строчки:

```
// Use this for initialization
void Start () {
}
// Update is called once per frame
void Update () {
}
```

Это две пустые стандартные функции. «void» — это команда вызова функции. «Start» и «Update» — названия функций.

«()» – означает что это процедурная функция, для неё не нужны внешние значения, и она не выдаёт результат, а просто выполняет определенные действия.

«{}» – начальные и конечные границы функций, между этими символами должны содержаться строчки функции.

«//» – обозначение комментарий к программному коду.

3. Дополнить код скрипта для управления объектом.

– Дописать подкатегорию «.Generic»: using System.Collections.Generic;

– Удалить функцию «Start».

– Создать переменные значения после строчки «public class PlayerScript : MonoBehaviour {»:

```
// Изменение скорости перемещения героя
public float playerSpeed = 2.0f;
// Текущая скорость перемещения
private float currentSpeed = 0.0f;
// Создание переменных для кнопок
```

```
public List<KeyCode> upButton;
public List<KeyCode> downButton;
public List<KeyCode> leftButton;
public List<KeyCode> rightButton;
// Сохранение последнего перемещения
private Vector3 lastMovement = new Vector3();
```

– Рассмотреть переменные значения:

«*public*» – публичный тип переменной (её смогут изменять другие игровые объекты).

«*float*» – тип значения, хранящегося в переменной.

«*playerSpeed*» – название переменной (можно свое название).

«= 2.0f» – начальное значение, хранящееся в переменной. Формат дробного числа: число с точкой, а в конце буква «f», используется для координат объекта в пространстве.

«*private*» — приватный тип переменной (такую переменную может изменять только сам объект, переменная для внутреннего пользования).

«*List<KeyCode>»* — тип переменной «массив из нескольких значений», в массиве содержатся ссылки на клавиши клавиатуры. «upButton», «downButton»,.. – названия применяемых клавиш.

«*Vector3»* — тип переменной «вектор в трех измерениях». «new Vector3()» — создание пустого вектора (обязательно для инициализации такого типа переменной).

– Сохранить изменения в скрипте (комбинация клавиш «Ctrl + S»).

4. Назначить скрипт объекту.

– В «иерархии» выбрать объект корабля.

– Перетащить файл скрипта в свойства корабля в окно «инспектора». Там появится новое свойство объекта «Player Script (Script)».

– В свойствах каждой переменной-кнопки «Up Button», «Down Button», «Left Button», «Right Button» в строчке «Size» поставить значение «2».

– Назначить для переменной-кнопки «Up Button» в строке «Element 0» «UpArrow» (клавиша со стрелкой вверх на клавиатуре) и «Element 1» – «W». Назначить остальным переменным все кнопки-стрелки и клавиши «W, A, S, D» (рис. 3.2.63).

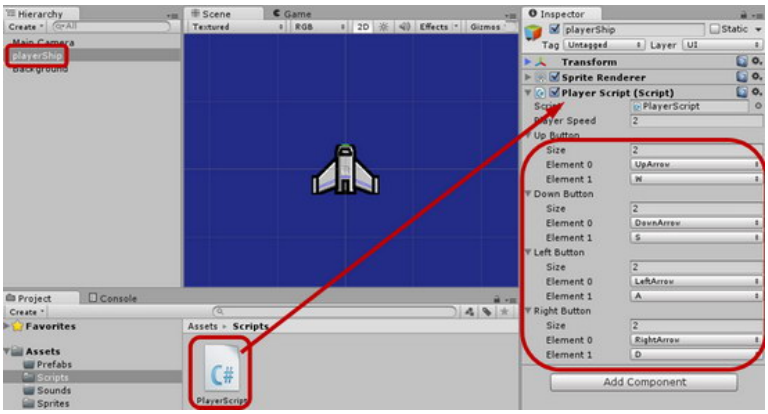

Рисунок 3.2.63 – Назначение управления объектом

Задание 7. Дополнить код скрипта для перемещения объекта.

Методические рекомендации по выполнению

1. Открыть скрипт в «MonoDevelop».
2. Внутри функции «Update» прописать ещё две функции (для многократного повторения на протяжении всей игры):

```
// Update is called once per frame
void Update () {
// Поворот героя к мышке
Rotation();
// Перемещение героя
Movement();
}
```

3. Дополнить функции после символа «{», закрывающего функцию «Update», и перед последним символом «}» следующим кодом:

```
// Поворот героя к мышке
void Rotation() {
    // Показываем игроку, где мышка
    Vector3 worldPos = Input.mousePosition;
    worldPos = Camera.main.ScreenToWorldPoint(worldPos);
    // Сохраняем координаты указателя мыши
    float dx = this.transform.position.x — worldPos.x;
    float dy = this.transform.position.y — worldPos.y;
    // Вычисляем угол между объектами «Корабль» и «Указатель»
    float angle = Mathf.Atan2(dy, dx) * Mathf.Rad2Deg;
    // Трансформируем угол в вектор
    Quaternion rot = Quaternion.Euler(new Vector3(0, 0, angle + 90));
    // Изменяем поворот героя
    this.transform.rotation = rot;
}
```

4. Описать функцию движения корабля «Movement»:

```
// Движение героя к мышке
void Movement() {
  // Необходимое движение
  Vector3 movement = new Vector3();
  // Проверка нажатых клавиш
  movement += MoveIfPressed(upButton, Vector3.up);
  movement += MoveIfPressed(downButton, Vector3.down);
  movement += MoveIfPressed(leftButton, Vector3.left);
  movement += MoveIfPressed(rightButton, Vector3.right);
  // Если нажато несколько кнопок, обрабатываем это
  movement.Normalize();
  // Проверка нажатия кнопки
  if(movement.magnitude > 0)
  {
    // После нажатия двигаемся в этом направлении
    currentSpeed = playerSpeed;
    this.transform.Translate(movement * Time.deltaTime * playerSpeed,
Space.World);
    lastMovement = movement;
  }
  Else
  {
    // Если ничего не нажато
    this.transform.Translate(lastMovement * Time.deltaTime * currentSpeed,
Space.World);
    // Замедление со временем
    currentSpeed *= 0.9f;
  }
}
// Возвращает движение, если нажата кнопка
Vector3 MoveIfPressed(List<KeyCode> keyList, Vector3 Movement) {
  // Проверяем кнопки из списка
  foreach (KeyCode element in keyList)
  {
    if(Input.GetKey (element))
    {
      // Если нажато, покидаем функцию
      return Movement;
    }
```

```
    }
    // Если кнопки не нажаты, то не двигаемся
    return Vector3.zero;
}
```

5. Сохранить файл скрипта.
6. Сохранить сцену (в главном меню нажимаем «File | Save Scene»).
7. Запустить игру – вкладка «Game» – кнопка «Maximize on Play» (игра запустится во всё окно Unity) – нажать клавишу «Play» (для отключения игры снова нажать клавишу «Play») (рис. 3.2.64).

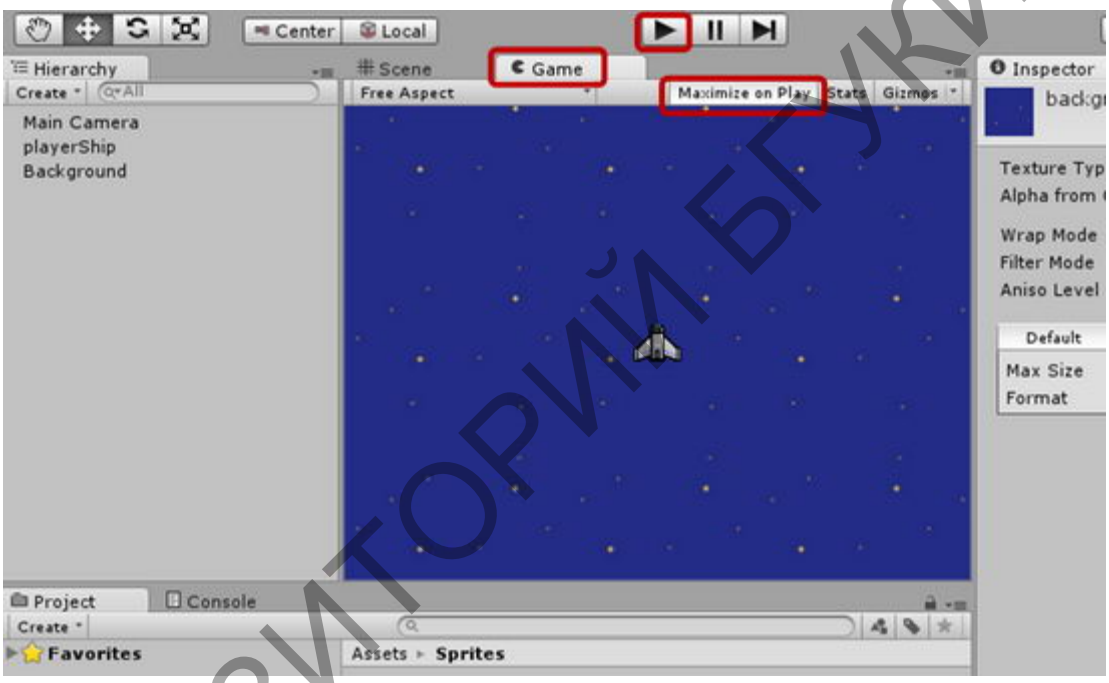

Рисунок 3.2.64 – Запуск игры в Unity

Задание 8. Добавить программный код для лазера.

Методические рекомендации по выполнению

1. Открыть скрипт «PlayerScript», добавить переменные (рис. 3.2.65):

```
// Переменная для лазера
public Transform laser;
// Как далеко от центра корабля будет появляться лазер
public float laserDistance = 0.2f;
// Задержка между выстрелами (кулдаун)
public float timeBetweenFires = 0.3f;
// Счетчик задержки между выстрелами
private float timeTilNextFire = 0.0f;
// Кнопка, которая используется для выстрела
public List<KeyCode> shootButton;
```

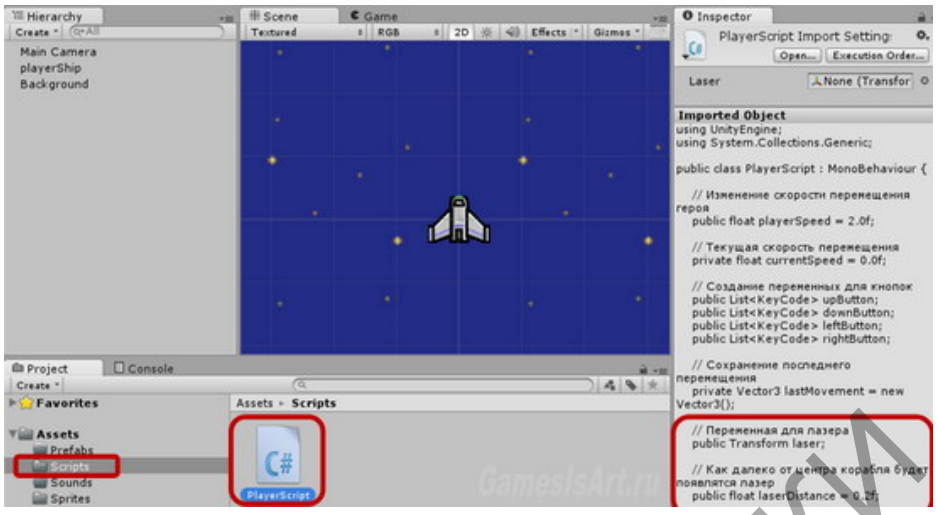

Рисунок 3.2.65 – Добавление переменной для лазера

2. В функции «Update()» кроме строчек «Rotation()» и «Movement()» добавить код обработки для добавленных переменных:

```
foreach (KeyCode element in shootButton)
    {
    if(Input.GetKey(element) && timeTilNextFire < 0)
        {
        timeTilNextFire = timeBetweenFires;
        ShootLaser();
        break;
        }
}
    timeTilNextFire -= Time.deltaTime;
```

Этот фрагмент кода проверяет нажатую клавишу. Если клавиша для выстрела, то проверяет следующее условие. Если задержка между выстрелами уже прошла (кулдаун истёк), то появляется возможность создавать новую пулю и обновить кулдаун. Если условия не выполняются, то уменьшается время кулдауна до следующего выстрела.

3. В конце скрипта до последних закрывающих скобок, расписать функцию создания пули:

```
// Создание лазера
void ShootLaser()
    {
    // Высчитываем позицию корабля
    float posX = this.transform.position.x + (Mathf.Cos ((transform.localEulerAngles.z — 90) * Mathf.Deg2Rad) * -laserDistance);
```

*float posY = this.transform.position.y + (Mathf.Sin ((transform.localEulerAngles.z — 90) * Mathf.Deg2Rad) * -laserDistance);*

// Создаём лазер на этой позиции

Instantiate(laser, new Vector3(posX, posY, 0), this.transform.rotation);

}

4. Сохранить файл скрипта.

5. В инспекторе для новых переменных назначить значения по умолчанию. В строчке «Shoot Button» выбрать «Size» = «2». В появившихся строчках выбрать клавиши «Mouse0» (левая кнопка мыши) и «Space» (пробел) (рис. 3.2.66).

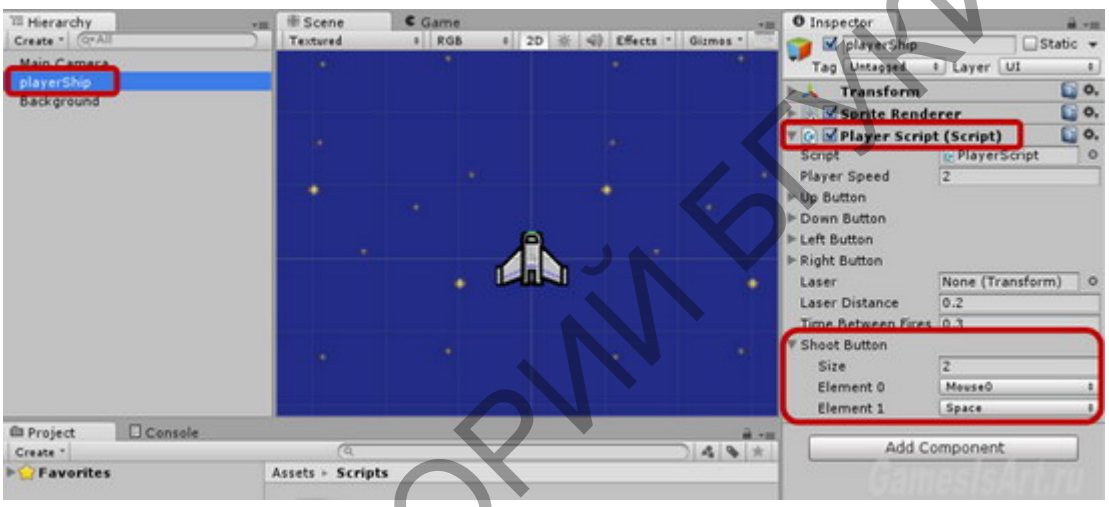

Рисунок 3.2.66 – Добавление функции создания пули

Задание 9. Добавить объект для пули.

Методические рекомендации по выполнению

1. Создайте и добавьте изображение «laser.png» для пули в папку «Sprites».

2. Разместить файл «laser.png» на игровой сцене.

3. Создать новый скрипт для объекта «Лазер» – «LaserScript».

4. Изменить текст скрипта:

using UnityEngine;

using System.Collections;

public class LaserScript : MonoBehaviour {

// Как долго существует лазер

public float lifetime = 2.0f;

// Как быстро движется лазер

public float speed = 5.0f;

// Как много наносит урона лазер при соприкосновении с врагами

public int damage = 1;

// Use this for initialization

```
    void Start () {
        // Уничтожение лазера по окончанию таймера
        Destroy (gameObject, lifetime);
    }
    // Update is called once per frame
    void Update () {
        transform.Translate(Vector3.up * Time.deltaTime * speed);
    }
}
```

5. Присоединить «LaserScript» к объекту «Лазер».

6. Создать для «Лазера» физическое тело –в главном меню выбрать «Component | Physics 2D | Box Collider 2D».

7. В разделе «Box Collider» изменить атрибуты «X = 0.06», «Y = 0.3». Так тело лазера будет чуть толще, чем его изображение, чтобы игроку было легче попадать во врагов (рис. 3.2.67).

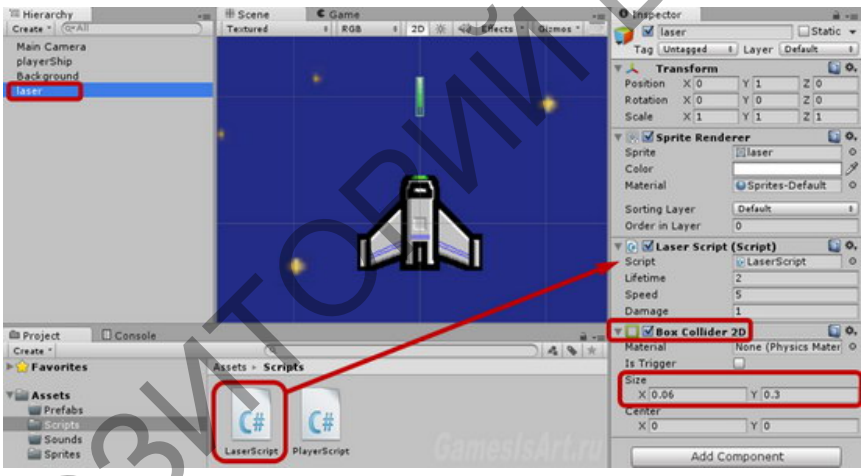

Рисунок 3.2.67 – Изменение параметров «Box Collider»

8. В окне «Project» открыть папку «Assets | Prefabs» и перетащить объект лазера из списка иерархии в папку «Prefabs».

Префаб – это не объект, а некий чертёж объекта, сам он не существует на игровой сцене, но мы можем создавать на сцене множество копий такого объекта-префаба (выстрелы, враги, спецэффекты). Любое изменение свойств в префабе будет действовать и на все созданные копии этого префаба, расставленные на игровой сцене.

9. Удалить объект «Лазер» с игровой сцены. Выделить объект «playerShip», перетащить префаб лазера в строчку свойства «Laser» в разделе «PlayerScript» (рис. 3.2.68).

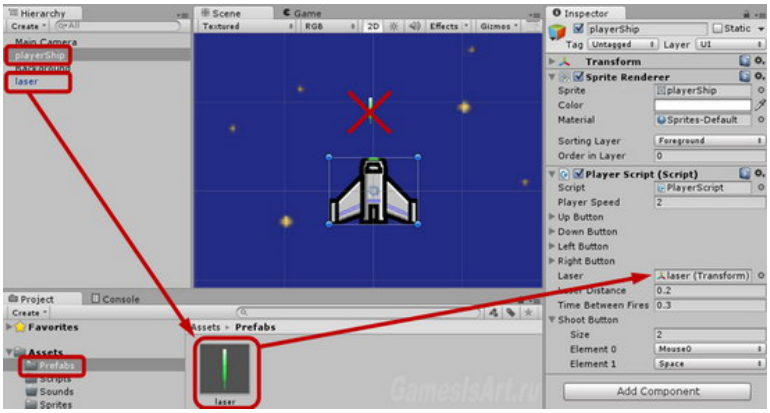

Рисунок 3.2.68 – Расстановка префабов

10. Создать круговой объект-коллайдер вокруг корабля: выбрать в главном меню «Component | Physics 2D | Circle Collider 2D», изменить радиус на «0.4» (рис. 3.2.69).

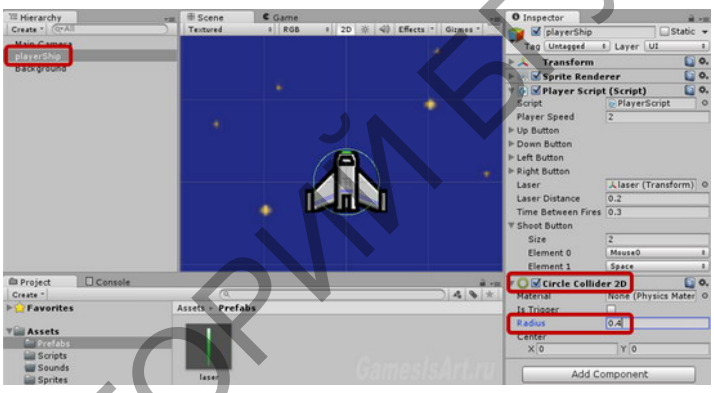

Рисунок 3.2.69 – Создание области столкновения

11. Запустить игру и проверить: по нажатию клавиши «пробел» или «левой кнопки мыши», корабль должен выпускать лазеры по направлению курсора мыши. Выйти из режима тестового запуска игры.

Задание 10. Добавить внешние объекты-припятствия.

Методические рекомендации по выполнению

1. В окне проекта в папку «Sprites» добавить спрайт «enemy.png».
2. Перетащить спрайт на игровую сцену (рис. 3.2.70).

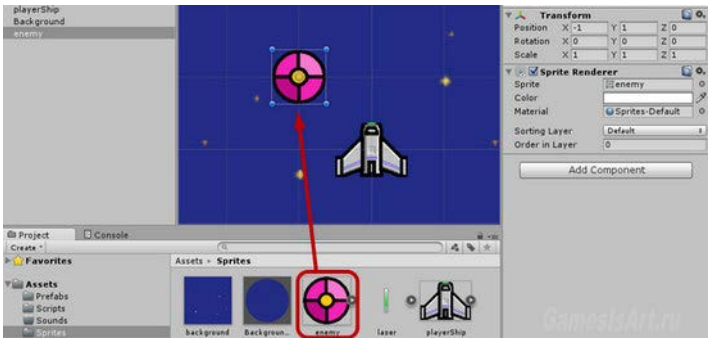

Рисунок 3.2.70 – Создание внешних объектов-припятствий

3. Создать новый скрипт – «MoveTowardsPlayer», дополнить:

```
using UnityEngine;
using System.Collections;
public class MoveTowardsPlayer : MonoBehaviour
  {
  // Переменная для координат объекта player
  private Transform player;
  // Скорость движения врага
  public float speed = 1.5f;
  // Use this for initialization
  void Start ()
  {
    player = GameObject.Find("playerShip").transform;
  }
  // Update is called once per frame
  void Update ()
  {
    Vector3 delta = player.position – transform.position;
    delta.Normalize();
    float moveSpeed = speed * Time.deltaTime;
    transform.position = transform.position + (delta * moveSpeed);
  }
}
```

В этом скрипте в начале игры находим объект «Корабль» и записываем его в переменную «player». Затем, в каждый момент игры скрипт: считывает новое положение корабля, вычисляет разницу между вражеским объектом и кораблём, сохраняет её в переменной «delta»; выставляет скорость врага так, чтобы он двигался в направлении корабля.

4. Перетащить созданный скрипт в свойства нового объекта.
5. Создать в свойствах вражеского объекта новый компонент: в главном меню выбирать «Component | Physics 2D | Circle Collider 2D». В его свойствах в строчке «Radius» ставим «0.350».
6. Запустить игру для проверки.

Задание 11. Создать скрипт для взаимодействия объектов.

Методические рекомендации по выполнению

1. Создать новый скрипт «EnemyScript»:

```
using UnityEngine;
public class EnemyScript : MonoBehaviour
{
  // Сколько раз нужно попасть во врага, чтобы уничтожить его
```

```
    public int health = 2;
    void OnCollisionEnter2D(Collision2D theCollision)
    {
        //Проверяем коллизию с объектом типа «лазер»
        if(theCollision.gameObject.name.Contains("laser"))
        {
            LaserScript laser = theCollision.gameObject.
GetComponent("LaserScript") as LaserScript;
            health -= laser.damage;
           Destroy (theCollision.gameObject);
        }
        if (health <= 0)
        {
            Destroy (this.gameObject);
        }
    }
}
```

2. Сохранить скрипт и присоединить к объекту.

3. Для столкновения объектов (событие «коллизия») создать физическое тело. В компонентах добавляем «Component | Physics 2D | Rigidbody 2D».

4. В компоненте изменить свойство «Gravity Scale» на значение «0» (убрать гравитацию) (рис. 3.2.71).

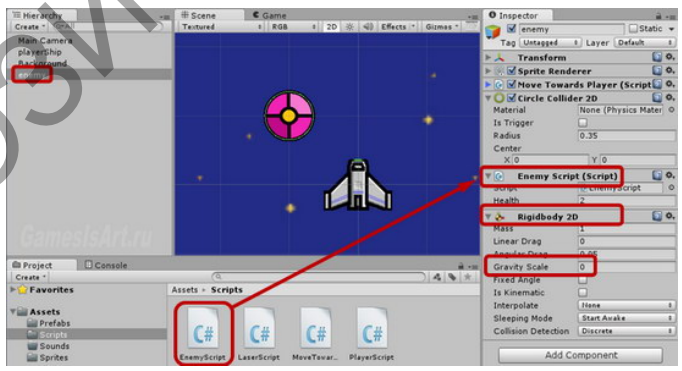

Рисунок 3.2.71 – Изменение параметров гравитации

5. Включить игру для тестирования.

Задание 12. Создать систему событий.

Методические рекомендации по выполнению

Систему событий (скрипт-контроллер) определяет, что будет происходить в игре: запускать игру, создавать волны, отображать игровые очки, и завершать игру, когда закончатся жизни.

1. Создать пустой игровой объект «GameController»: «GameObject |

Create Empty». В свойствах ставим «Position» = (0, 0, 0).

2. В строке «Tag» заменить значение «Untagged» на «GameController» («Tag» — это способ объединить несколько объектов в группу).
3. Создать новый скрипт «GameController».
4. В разделе описания переменных добавить следующие строчки:

```
// Создание переменной «враг»
public Transform enemy;
```

5. Переместить объект «enemy» в папку «Prefabs». Удалить объект «enemy» с игровой сцены.
6. Перетащить префаб в переменную «enemy» в свойствах скрипта.
7. Создать переменные, которые нам понадобятся:

```
// Временные промежутки между событиями, кол-во врагов
public float timeBeforeSpawning = 1.5f;
public float timeBetweenEnemies = 0.25f;
public float timeBeforeWaves = 2.0f;
public int enemiesPerWave = 10;
private int currentNumberOfEnemies = 0;
```

8. Внутри функции «Start» написать функцию SpawnEnemies – чтобы припятствия появлялись не группами по 10 штук.

```
StartCoroutine (SpawnEnemies());
```

Функция Coroutine приостанавливает или продолжает выполнение действий, срабатывающих через какой-то промежуток времени.

9. После функции Update прописать саму функцию:

```
// Появление волн врагов
IEnumerator SpawnEnemies()
{
  // Начальная задержка перед первым появлением врагов
  yield return new WaitForSeconds (timeBeforeSpawning);
  // Когда таймер истекёт, начинаем производить эти действия
  while(true)
  {
    // Не создавать новых врагов, пока не уничтожены старые
    if (currentNumberOfEnemies <= 0)
    {
      float randDirection;
      float randDistance;
      // Создать 10 врагов в случайных местах за экраном
      for (int i = 0; i < enemiesPerWave; i++)
        {
        // Задаём случайные переменные для расстояния и направления
```

```
                randDistance = Random.Range (10, 25);
                randDirection = Random.Range (0, 360);
                // Используем переменные для задания координат появления врага
                float posX = this. transform.position.x + (Mathf.Cos((randDirection) *
Mathf.Deg2Rad) * randDistance);
                float posY = this. transform.position.y + (Mathf.Sin((randDirection) *
Mathf.Deg2Rad) * randDistance);
                // Создаём врага на заданных координатах
                Instantiate (enemy, new Vector3 (posX, posY, 0),
this.transform.rotation);
                currentNumberOfEnemies++;
                yield return new WaitForSeconds (timeBetweenEnemies);
              }
            }
            // Ожидание до следующей проверки
            yield return new WaitForSeconds (timeBeforeWaves);
          }
        }
```

10. Внутри класса «GameController» создать переменную «currentNumberOfEnemies»:

```
// Процедура уменьшения количества врагов в переменной
public void KilledEnemy()
{
  currentNumberOfEnemies--;
}
```

11. Отредактировать скрипт «EnemyScript»: внутри функции «onCollisionEnter2D» после функции «Destroy (this.gameObject)» добавить:

```
GameController controller = GameObject.FindGameObjectWithTag
("GameController").GetComponent("GameController") as GameController;
controller.KilledEnemy();
```

12. Сохранить оба скрипта.
13. Сохранить игровую сцену.
14. Запустить игру.

Задание 13. Добавить спецэффекты в игру. Система частиц.

Методические рекомендации по выполнению

Система частиц – это инструмент для создания спецэффектов, состоит из двух типов объектов: источник частиц и сами частицы. *Частицы* – это небольшие объекты, которые живут в игре строго ограниченное время. Примеры частиц: брызги воды, языки пламени в огне, клубы дыма.

Источники частиц постоянно создают всё новые и новые частицы, взамен старых исчезнувших, они определяют все характеристики этих частиц.

1. В главном меню выбать строчку «GameObject | Particle System» для создания объекта-источника. Изменить имя объекта на «Explosion».

2. Настроить анимацию частиц:

– В свойствах объекта в разделе «Particle System» в строчке «Duration» (продолжительность) выставить значение «1.00».

– В строчке «Start Lifetime» нажать на треугольник справа, выбрать значение «Random Between Two Constants». Снизу выставить значения «0» и «1» (частицы будут появляться с размером от 0 до 1). В строчке «Start Size» выставить «случайное число» от «0» до «5».

– В строчке «Start Color» выбрать вариант «Gradient» (желтый – красный), альфа-канал (200).

– Раздел «Emission» (излучение) изменить строчку «Rate» на значение «200». Это определяет, сколько частиц будет появляться за отрезок времени.

– В разделе «Shape» изменить значение строчки «Shape» на значение «Sphere» (чтобы частицы разлетались по кругу во все стороны) и «Radius» на «0.15». Включить опцию «Random Direction».

– Раздел «Explosion» изменить «Simulation Space» на «World».

– Убрать галочку «Looping».

3. Переместить созданный объект в папку «Prefabs», удалить с игровой сцены.

4. Отредактировать скрипт «EnemyScript», добавить переменную:

```
// Анимация при уничтожении объекта
public Transform explosion;
```

В описании функции CollisionEnter2D после строчки «if (health <=0)» добавить несколько строчек:

```
if (health <= 0)
{
// Срабатывает при уничтожении объекта
if(explosion);
{
GameObject exploder = ((Transform)Instantiate(explosion, this.transform.position, this.transform.rotation)).gameObject;
Destroy(exploder, 2.0f);
}
Destroy (this.gameObject);
```

5. Сохранить изменения в скрипте.

6. Выбрать префаб «enemy», в его свойствах найти «Enemy Script» - «Explosion». Перетащить префаб «Explosion» в найденную переменную

«Explosion».

7. Запустить игру, просмотреть результат (рис. 3.2.72).

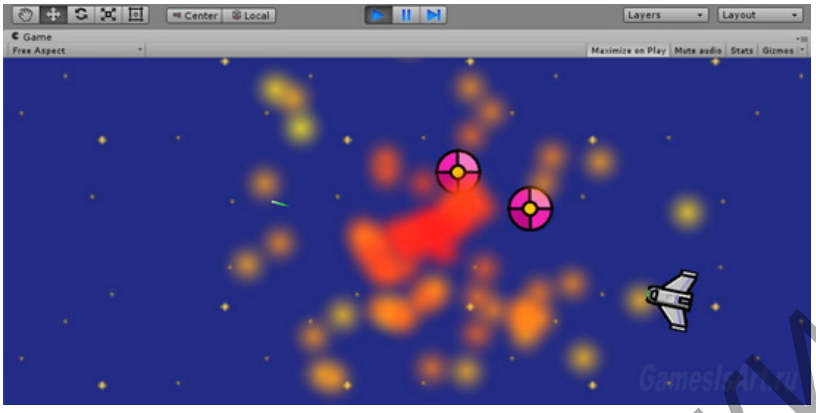

Рисунок 3.2.72 – Визуализация игры

Задание 14. Добавить в игру звуковые эффекты и музыку.

Методические рекомендации по выполнению

1. Ознакомиться с примером добавления звуковых эффектов и музыки в приведенном примере по ссылке: http://gamesisart.ru/game_dev_unity.html.
2. Самостоятельно добавить звуковые эффекты и музыку в игру.

Задание 15. Создать текст на игровом поле.

Методические рекомендации по выполнению

1. Ознакомиться с примером добавления текста (игровые очки, информация о прогрессе) в приведенном примере по ссылке: http://gamesisart.ru/game_dev_unity.html.
2. Самостоятельно добавить текст на игровой экран.

Задание 16. Сохранить и опубликовать игру.

Методические рекомендации по выполнению

1. В главном меню выбрать «File | Build Settings».
2. Выбрать, какие игровые сцены добавить в конечную сборку.
3. Нажать кнопку «Add Current».
4. Выбрать, для какой игровой платформы публикуем игру: компьютер, консоли, мобильные устройства, браузеры. Выбрать «Windows – x86» (рис. 3.2.73).
5. Нажать клавишу «Build And Run».
6. Выбрать название для игры «Game», сохранить.
7. После компиляция появится меню параметров экрана для запуска игры.
8. Выбрать развертку экрана и запустить игру, нажав «Play».

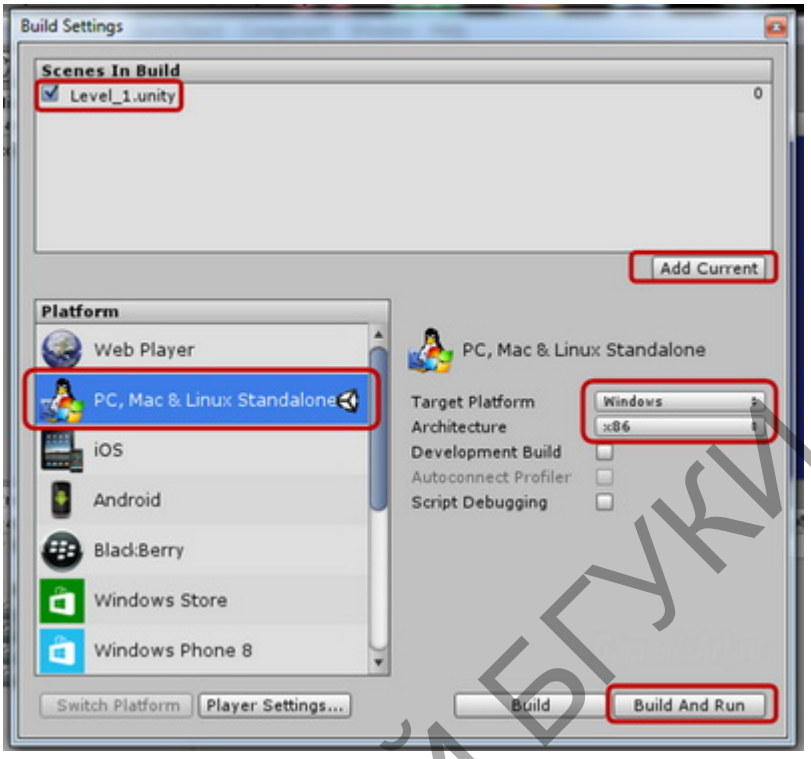

Рисунок 3.2.73 – Параметры для компиляции игры

Источник практического занятия: Игровой движок Unity 3-D. Курс обучения (http://gamesisart.ru/game_dev_unity.html)

Тема 7. Создание авторского проекта компьютерной игры (6 часов)

Лабораторная работа 7. Создание авторского проекта компьютерной игры

Цель. Используя полученные знания подготовить и разработать авторский проект компьютерной игры.

Задание 1. Разработать дизайн-документ игры по выбранной теме.

Задание 2. Подготовить модели-заготовки (2-D, 3-D) для проекта.

Задание 3. Выбрать программную среду для разработки авторского проекта.

Задание 4. Завершить создание авторского проекта.

4. РАЗДЕЛ КОНТРОЛЯ ЗНАНИЙ

4.1 Перечень требований к зачету

Дисциплина изучается в течение одного семестра и предусматривает самостоятельную работу студентов. Для допуска к зачету студент должен выполнить следующие требования:

1. посещение лекционных занятий;
2. выполнение лабораторных работ;
3. выполнение заданий для контролируемой самостоятельной работы.

Форма проведения зачета – устный опрос.

4.2 Критерии оценки результатов учебной деятельности студентов

В целях подготовки к текущей/промежуточной аттестации, студенту следует просмотреть все имеющиеся и рекомендуемые материалы, представленные в печатном или электронном виде. Промежуточная аттестация проводиться с целью оценки качества усвоения студентами всего объёма содержания дисциплины и определения фактически достигнутых знаний, навыков и умений, а также компетенций, сформированных за время аудиторных занятий и самостоятельной работы студента.

Критерии оценивания ответов студентов

Оценка «отлично» (10-8 баллов) / «зачтено». Ответы на поставленные вопросы излагаются логично, последовательно и не требуют дополнительных пояснений. Делаются обоснованные выводы. Демонстрируются глубокие знания в изучаемой области. Студент демонстрирует владение понятийным аппаратом и научным языком по предмету, умение его использовать в постановке и решении научных и профессиональных задач; способность самостоятельно решать сложные проблемы в рамках учебной программы; усвоение основной и дополнительной литературы, рекомендованной учебной программой; активная самостоятельная работа на лабораторных (практических) занятиях, высокий уровень культуры исполнения заданий, грамотное оформление учебной документации.

Оценка «хорошо» / «зачтено» (7-5 баллов). Ответы на поставленные вопросы излагаются систематизировано и последовательно. Материал излагается уверенно. Демонстрируется умение анализировать материал, однако не все выводы носят аргументированный и доказательный характер. Студент демонстрирует активную самостоятельную работу на практических, лабораторных занятиях, высокий уровень культуры исполнения заданий и

оформления учебной документации, периодически участвует в групповых обсуждениях.

Оценка «удовлетворительно» (4 балла) / «зачтено». Допускаются нарушения в последовательности изложения. Имеются упоминания об отдельных базовых нормативно-правовых актах. Демонстрируются поверхностные знания вопроса, с трудом решаются конкретные задачи. Имеются затруднения с выводами. Студент демонстрирует достаточный объем знаний по предмету в рамках образовательного стандарта.

Оценка «неудовлетворительно» (3-1 баллов) / «не зачтено». Материал излагается непоследовательно, сбивчиво, не представляет определенной системы знаний по дисциплине. Не проводится анализ. Выводы отсутствуют. Ответы на дополнительные вопросы отсутствуют. На лабораторных (практических) занятиях студент был пассивен, демонстрировал низкий уровень культуры исполнения заданий и их оформления, отсутствие знаний по предмету в рамках образовательного стандарта или отказ от ответа.

4.3 Задания для контролируемой самостоятельной работы студентов

Самостоятельная работа студентов направлена на совершенствование их умений и навыков по дисциплине «Игровая культура и дизайн». Цель самостоятельной работы студентов – способствование усвоению в полном объеме учебного материала дисциплины через систематизацию, планирование и контроль собственной деятельности.

Вопросы и творческие задания

Задание 1. Подготовить реферат или презентацию по одному из вопросов по теме «Тема 3. Игровая культура в условиях компьютеризации»

1. Игровые платформы: персональные компьютеры, ноутбуки.
2. Игровые платформы: игровые консоли.
3. Игровые платформы: мобильные устройства (сотовые телефоны, смартфоны, планшеты).
4. Програмное обеспечение игровых платформ.
5. Этапы реализации творческой идеи.
6. Издатели (создание) и продвижение компьютерных игр.
7. Игровое сообщество (игроки).
8. Классификации компьютерных игр.
9. Жанры компьютерных игр.

Задание 2. «Тема 4. Дизайн-концепт в разработке компьютерных игр»

Разработать и предоставить проектную документацию для создания компьютерной игры на выбранную самостоятельно тему. Форма отчета – электронная версия (файл).

Задание 3. «Тема 7. Создание авторского проекта компьютерной игры»

Разработать компьютерную игру на основе своей проектной документации. Форма отчета – электронная версия (файл).

4.4 Контрольные вопросы по темам

Введение

1. Роль дисциплины «Игровая культура и дизайн» в подготовке специалиста высшей квалификации.
2. Связь с другими дисциплинами специализации.
3. Цели и задачи курса, предмет и объект изучения.

Тема 1. Игровая культура как социокультурное явление

1. Игровая культура как объект культурологического исследования.
2. Сущность и особенности игровой культуры.
3. Понятие игры в контексте современных научных исследований.
4. Первичная и вторичная игровая культура.
5. Культурологические концепции.

Тема 2. Игровая индустрия

1. Нормативно-правовое регулирование игровой индустрии.
2. Индустрия компьютерных игр.
3. Экономический аспект индустрии компьютерных игр.
4. Культурный аспект индустрии компьютерных игр.

Тема 3. Игровая культура в условиях компьютеризации

1. Игровые платформы: персональные компьютеры, ноутбуки.
2. Игровые платформы: игровые консоли.
3. Игровые платформы: мобильные устройства (сотовые телефоны, смартфоны, планшеты).
4. Програмное обеспечение игровых платформ.
5. Этапы реализации творческой идеи.
6. Издатели (создание) и продвижение компьютерных игр.
7. Игровое сообщество (игроки).
8. Классификация и жанры компьютерных игр.

Тема 4. Дизайн-концепт в разработке компьютерных игр

1. Первичный концепт-документ: краткое описание игрового процесса и особенности игры.
2. Создание дизайн-документ («диздок»).
3. Концепция и художественный стиль игры.
4. Программированием игрового дизайна: сюжетная линия, структура игры и правила.
5. Дизайн уровней, создание игрового ландшафта.
6. Моделировние объектов (2-D и 3-D графика).

Тема 5. Программная среда разработки 2- D игр

1. Обзор программная сред разработки 2- D игр: Indie Game Maker, Sonic Maker, 001 Game Creator, StencylWorks, Construct и т.д.
2. Язык и среда программирования Scratch (scratch.mit.edu).
3. Работа с объектами-спрайтами в Scratch.
4. Способы записи алгоритмов в Scratch.
5. Алгоритмические структуры в Scratch.
6. Визуальные языки программирования.
7. Игровой конструктор – Construct2. Интерфейс программы, набор панелей.
8. Списки событий, построение логических блоков в Construct2.
9. Галерея объектов, встроенные поведения в Construct2.
10. Создание спрайтов, воспроизведение музыки и звуков в Construct2.
11. Эффекты и визуализация в Construct2. Дополнительные функции: физика объектов, система костной анимации, 3-Д функции, встроенные сценарии в Construct2.

Тема 6. Программная среда разработки 3- D игр

1. Обзор программных сред разработки 3- D игр: Neo Axis Engine, Unity, Game Guru, Fps Creator, Kodu Game Lab, Sauerbraten (aka Cube 2) и другие.
2. Платформа разработки многоплатформенной игры и интерактивного контента Unity3d (unity3d.com).
3. Достоинства и недостатки платформы Unity3d.
4. Интерфейс и основные инструменты программы Unity3d.
5. Импортирование объектов и игровых ландшафтов в Unity3d.
6. Наполнение библиотеки проекта Unity3d.
7. Сохранение и экспорт проекта в Unity3d.
8. Языки программирования в Unity3d: C#, UnityScript.

9. Создание и структура файла скрипта.

Тема 7. Создание авторского проекта компьютерной игры

1. Разбор алгоритма игры.
2. Этапы и особенности разработки проекта игры.
3. Определение жанра и сюжета игры: легенда, идея, конвенции.
4. Выбор игровой платформы и программного средства.
5. Игровой дизайн-концепт.
6. Рабочий прототип.
7. Развитие прототипа в конечный продукт.
8. Завершение проекта и запуск игры.

4.5 Перечень вопросов к зачету

1. Игровая культура как объект культурологического исследования.
2. Сущность и особенности игровой культуры.
3. Понятие игры в контексте современных научных исследований.
4. Первичная и вторичная игровая культура.
5. Культурологические концепции.
6. Индустрия компьютерных игр.
7. Экономический аспект индустрии компьютерных игр.
8. Культурный аспект индустрии компьютерных игр.
9. Нормативно-правовое регулирование игровой индустрии.
10. Програмное обеспечение игровых платформ.
11. Игровые платформы: персональные компьютеры, ноутбуки.
12. Игровые платформы: игровые консоли.
13. Игровые платформы: мобильные устройства (сотовые телефоны, смартфоны, планшеты).
14. Этапы реализации творческой идеи.
15. Издатели (создание) и продвижение компьютерных игр.
16. Игровое сообщество (игроки).
17. Классификация и жанры компьютерных игр.
18. Первичный концепт-документ: краткое описание игрового процесса и особенности игры.
19. Создание дизайн-документ («диздок»). Концепция и художественный стиль игры.
20. Программированием игрового дизайна: сюжетная линия, структура игры и правила.

21. Дизайн уровней, создание игрового ландшафта. Моделировние объектов (2-D и 3-D графика).

22. Обзор программная сред разработки 2- D игр: Indie Game Maker, Sonic Maker, 001 Game Creator, StencylWorks, Construct и т.д.

23. Язык и среда программирования Scratch (scratch.mit.edu).

24. Работа с объектами-спрайтами в среде программирования Scratch.

25. Способы записи алгоритмов в среде программирования Scratch. Алгоритмические структуры.

26. Визуальные языки программирования.

27. Игровой конструктор – Construct2. Интерфейс программы, набор панелей.

28. Списки событий, построение логических блоков в Construct2. Галерея объектов, встроенные поведения.

29. Создание спрайтов, воспроизведение музыки и звуков. Эффекты и визуализация.

30. Дополнительные функции: физика объектов, система костной анимации, 3-Д функции, встроенные сценарии.

31. Обзор программных сред разработки 3- D игр: Neo Axis Engine, Unity, Game Guru, Fps Creator, Kodu Game Lab, Sauerbraten (aka Cube 2) и другие.

32. Платформа разработки многоплтформенной игры и интерактивного контента Unity3d (unity3d.com). Достоинства и недостатки.

33. Интерфейс и основные инструменты программы Unity3d (unity3d.com).

34. Импортирование объектов и игровых ландшафтов в. Сохранение и экспорт.

35. Наполнение библиотеки проекта Unity3d (unity3d.com).

36. Языки программирования в Unity3d: C#, UnityScript. Создание и структура файла скрипта.

37. Разбор алгоритма игры. Определение жанра и сюжета игры: легенда, идея, конвенции.

38. Этапы и особенности разработки проекта игры.

39. Выбор игровой платформы и программного средства.

40. Игровой дизайн-концепт.

41. Рабочий прототип игры и его развитие в конечный продукт.

42. Завершение проекта и запуск игры.

5. ВСПОМОГАТЕЛЬНЫЙ РАЗДЕЛ

5.1 Учебная программа

Учебная программа составлена на основе Образовательного стандарта высшего образования ОСВО 1-21 04 01-2013 по специальности 1-21 04 01 Культурология (по направлениям) и учебного плана по специализации 1-21 04 01-02 04 Информационные системы в культуре, рег. №Д-21-1-71/17 уч. от 04.07.2017

СОСТАВИТЕЛЬ:

О.М. Кунцевич, преподаватель кафедры информационных технологий в культуре учреждения образования «Белорусский государственный университет культуры и искусств»

РЕЦЕНЗЕНТЫ:

А.Г. Буравкин, ведущий научный сотрудник отдела совместных программ космических и информационных технологий государственного научного учреждения «Объединенный институт проблем информатики Национальной академии наук Беларуси», кандидат технических наук, доцент

Е.А. Криштаносова, доцент кафедры культурологии учреждения образования «Белорусский государственный университет культуры и искусств», кандидат культурологии, доцент

РЕКОМЕНДОВАНА К УТВЕРЖДЕНИЮ:

кафедрой информационных технологий в культуре учреждения образования «Белорусский государственный университет культуры и искусств» (протокол № 9 от 29.05.2019)

президиумом научно-методического совета учреждения образования «Белорусский государственный университет культуры и искусств» (протокол № .. от 12.06.2019)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Стремительное развитие форм и методов социокультурной деятельности предполагает высокий уровень подготовки квалифицированных специалистов в сфере игровой культуры. Информатизация культуры накладываает серьезный отпечаток на игровую деятельность, в которой активно начинают использоваться веб-технологии. Популярность компьютерных игр предъявляет требования к игровому дизайну и решения профессиональных задач по использованию, разработке, сопровождению и продвижению игрового продукта.

Игра имманентно присуща человеку, актуальна во все эпохи. Современный интерес к проблеме игры обусловлен существенными трансформациями технологического контекста существования человека и формированием новой сферы человеческой деятельности, обладающей значительным культуротворческим потенциалом. Данные трансформации представлены целым комплексом социально-культурных изменений, основная часть которых определяются появлением новых форм коммуникации, рыночным характером производства и внедрением компьютерной техники и новых информационных систем во все формы жизнедеятельности человека.

Игра в социокультурной деятельности может выполнять рекреоционные, образовательные, воспитательные, просветительные и другие функции. Разитие методик и технологий создания и продвижения игрового продукта повлияло на появление новой профессиональной группы людей, делающие игру игрой «геймдизайнеров». Разработчики игр, создавая мир игры, правиласуществования, мотивируют игрока на следующий ход развития персонажа. Разработка игровой среды и дизайна невозможны без теоретических знаний, умений и практических навыков, игровой культуры, разработчиков и игроков.

Целью учебной дисциплины «Игровая культура и дизайн» является формирование у студентов игровой культуры, необходимого объема теоретических знаний и практических умений использования современных информационных компьютерных технологий по созданию игровой среды, овладение навыками практического использования инструментов программы в будущей профессиональной деятельности.

Предметом изучения дисциплины «Игровая культура и дизайн» являются методы и технологии создания, сопровождения и продвижения игрового продукта.

Целевая направленность дисциплины обуславливает решение следующих *задач*:

– формирование знаний в области дизайн-концепта в разработке компьютерных игр;

– формирование системы базовых знаний и навыков работы в конструкторах игр;

– освоение методов и технологий работы на игровых платформах;

– освоение эффективных методов и средств решения творческих задач в области игровой среды на основе совместного использования игровых технологий и векторной компьютерной графики.

В результате изучения дисциплины студенты должны *знать*:

– классификацию компьютерных игр;

– цели и задачи использования современных игровых технологий;

– теоретические основы растровой и векторной компьютерной графики;

– тенденции развития дизайна в игровой культуре;

– принципы и методы создания игровой среды;

– специфику создания игровой среды в сфере культуры;

– освоение эффективных методов и средств решения творческих задач в области игровой культуры.

Должны *уметь*:

– создавать трехмерные модели для игровой среды;

– выполнять построение формы предметов в различных проекциях;

– разработать дизайн-концепт игры в сфере культуры;

– применять основные методы создания игрового сценария.

Должны *владеть*:

– практическими навыками использования игровых платформ;

– основными навыками создания и продвижения компьютерной игры;

– творческим подходом к поставленной задачи в процессе практической деятельности.

Усвоение данной учебной программы должно обеспечивать формирование следующих групп компетенций.

Академические компетенции, которые включают знания и умения по изучаемой дисциплине:

– АК-1. Уметь использовать базовые научно-теоретические знания для решения теоретических и практических задач;

– АК-2. Владеть системным и сравнительным анализом;

– АК-3. Владеть исследовательскими навыками;

– АК-4. Уметь работать самостоятельно;

– АК-5. Быть способным порождать новые идеи (обладать креативностью);

– АК-6. Владеть междисциплинарным подходом к решению проблем;

– АК-7. Иметь навыки, связанные с использованием технических устройств, управлением информацией и работой с компьютером;

– АК-8. Владеть навыками устной и письменной коммуникации;

– АК-10. Владеть методическими знаниями исследовательскими умениями, которые обеспечивают решение задач инновационно-методической и научно-исследовательской деятельности в культурологии.

Социально-личностные компетенции, которые включают культурно-ценностные ориентации, знание идеологических, моральных ценностей общества и государства и умение управлять ими:

– СЛК-2. Быть способным к социальному взаимодействию;

– СЛК-6. Уметь работать в команде;

– СЛК-8. Совершенствовать и развивать свой интелектуальный и общий уровень, добиваться морального и физического совершенствования своей личности;

– СЛК-9. Формировать и аргументировать собственные суждения и профессиональную позицию.

Профессиональные компетенции, которые включают способность рещать задачи, разрабатывать планы и обеспечивать их решение в выбранной сфере профессиональной деятельности:

– ПК-4. Оценивать состояние, тенденции и перспективы развития культуры и искусства;

– ПК-5. Прогнозировать, планировать и организовать инновационно-метадическую и художественно-творческую деятельность в сфере культуры и искусств.

– ПК-7. Заниматься научно-исследовательской деятельностью в культурологии.

– ПК-8. Анализировать и оценивать собранную информацию;

– ПК-9. Организовывать свою работу на научной основе, владеть компьютерными методами сбора, сохранения и обработки информации в сфере профессиональной деятельности;

– ПК-14. Использовать современные методики и технические средства обучения.

Содержание дисциплины «Игровая культура и дизайн» межпредметно связана с учебными дисциплинами «Веб-дизайн и реклама», «Информационные технологии в культуре», «Компьютерная графика», «Графический дизайн», «Художественное проектирование», «Трехмерное моделирование и анимация».

Основными формами учебной дисциплины являются лекции, лабораторные работы и самостоятельное изучение отдельных вопросов. Учебным планом на изучение учебной дисциплины «Игровая культура и

дизайн » всего предусмотрено 62 часа, из них 36 часов – аудиторные занятия. Примерное распределение аудиторных часов по видам занятий: лекции – 4 часа, лабораторные – 32 часа.

Рекомендованная форма контроля – зачет.

СОДЕРЖАНИЕ УЧЕБНОГО МАТЕРИАЛА

Введение

Роль дисциплины «Игровая культура и дизайн» в подготовке специалиста высшей квалификации. Связь с другими дисциплинами специализации. Цели и задачи курса, предмет и объект изучения.

Тема 1. Игровая культура как социокультурное явление

Игровая культура как объект культурологического исследования. Сущность и особенности игровой культуры. Понятие игры в контексте современных научных исследований. Первичная и вторичная игровая культура. Культурологические концепции.

Тема 2. Игровая индустрия

Индустрия компьютерных игр. Экономический аспект. Культурный аспект. Нормативно-правовое регулирование игровой индустрии.

Тема 3. Игровая культура в условиях компьютеризации

Игровые платформы: персональные компьютеры, ноутбуки, игровые консоли, мобильные устройства (сотовые телефоны, смартфоны, планшеты). Програмное обеспечение игровых платформ. Этапы реализации творческой идеи. Издатели (создание) и продвижение компьютерных игр. Игровое сообщество (игроки). Жанры компьютерных игр.

Тема 4. Дизайн-концепт в разработке компьютерных игр

Первичный концепт-документ: краткое описание игрового процесса и особенности игры. Создание дизайн-документ («диздок»). Концепция и художественный стиль игры. Программированием игрового дизайна: сюжетная линия, структура игры и правила. Дизайн уровней, создание игрового ландшафта. Моделировние объектов (2-D и 3-D графика).

Тема 5. Программная среда разработки 2- D игр

Обзор программная сред разработки 2- D игр: Indie Game Maker, Sonic Maker, 001 Game Creator, StencylWorks, Construct и т. д.

Язык и среда программирования Scratch (scratch.mit.edu). Работа с объектами-спрайтами. Способы записи алгоритмов. Алгоритмические структуры. Визуальные языки программирования. Разработка игр по собственному сценарию.

Игровой конструктор – Construct2. Интерфейс программы, набор панелей. Списки событий, построение логических блоков. Галерея объектов,

встроенные поведения. Создание спрайтов, воспроизведение музыки и звуков. Эффекты и визуализация. Дополнительные функции: физика объектов, система костной анимации, 3-Д функции, встроенные сценарии.

Тема 6. Программная среда разработки 3- D игр

Обзор программных сред разработки 3- D игр: Neo Axis Engine, Unity, Game Guru, Fps Creator, Kodu Game Lab, Sauerbraten (aka Cube 2) и другие.

Unity3d (unity3d.com). Платформа разработки многоплтформенной игры и интерактивного контента. Интерфейс и основные инструменты программы. Импортирование объектов и игровых ландшафтов. Наполнение библиотеки проекта. Создание игр по заранее определенному сценарию. Разбор алгоритма игры. Разработка проекта игры по собственному сценарию.

Тема 7. Создание авторского проекта компьютерной игры

Определение жанра и сюжета игры: легенда, идея, конвенции. Выбор игровой платформы и программного средства. Игровой дизайн-концепт. Рабочий прототип. Развитие прототипа в конечный продукт. Завершение.

5.2 Учебно-методические карты учебной дисциплины для дневной и заочной формы получения высшего образования

Для дневной формы обучения

Название раздела, темы	Количество аудиторных часов		Количество часов УСР	Форма контроля знаний
	Лекции	Лабораторные занятия		
Введение	0,5			
Тема 1. Игровая культура как социокультурное явление	1			
Тема 2. Игровая индустрия	1			
Тема 3. Игровая культура в условиях компьютеризации	1	2	2	реферат
Тема 4. Дизайн-концепт в разработке компьютерных игр	0,5	4	2	дизайн-концепт
Тема 5. Программная среда разработки 2- D игр		6		
Тема 6. Программная среда разработки 3- D игр		6		
Тема 7. Создание авторского проекта компьютерной игры		6	4	проект
Всего:	4	24	8	

Для заочной формы обучения

Название раздела, темы	Количество аудиторных часов		Форма контроля знаний
	Лекции	Лабораторные занятия	
Тема 1. Игровая культура как социокультурное явление	1		
Тема 3. Игровая культура в условиях компьютеризации	1		
Тема 4. Дизайн-концепт в разработке компьютерных игр		2	
Тема 5. Программная среда разработки 2- D игр		2	
Тема 6. Программная среда разработки 3- D игр		2	
Тема 7. Создание авторского проекта компьютерной игры		4	проект
Всего:	2	10	

***** Тема 2. «Игровая индустрия» остаются на самостоятельное изучение студентов.

5.3 Список основной литературы

1. *Аликин, В. А.* Игра в философии Роже Кайуа: между инстинктами и культурой // Вестн. Волгогр. гос. ун-та. Сер. 7, Филос. – 2016. № 1 (31). – С. 21–27.

2. *Аликин, В. А.* Феномен игры в обществе: социально-философский анализ : дисс. … канд. филос. наук / В. А. Алики . – Новочеркасск, 2003. – 153 с.

3. *Андресен, Бент. Б.* Мультимедиа в образовании: специализированный учеб. курс: [пер. с англ] / Бент. Б. Андерсен, Катя Ван ДенБринк. – М. : Дрофа, 2007. – 221 с.

4. *Беркут, М.* Топ 5 программ для создания 3д игр без программирования [Электронный ресурс]. – Режим доступа: https://pikabu.ru/story/top_5_programm_dlya_sozdaniya_3d_igr_bez_programmirovaniya_4244461– Дата доступа: 15.05.2019

5. *Берн, Э.* Игры, в которые играют люди. Психология человеческих взаимоотношений ; Люди, которые играют в игры. Психология человеческой судьбы / Э. Берн. – СПб. : Лениздат, 1992. – 400 с.

6. *Боресков, А. В.* Графика трехмерной компьютерной игры на основе OpenGL : практическое пособие / А.В. Боресков. – Москва : Диалог-МИФИ, 2004. – 383 с. – Режим доступа: по подписке. – URL:http://biblioclub.ru/index.php?page=book&id=89378. – Текст : электронный.

7. *Вербина, О. В.* Игра как агент развития современных социальных коммуникаций / О. В Вербина // Наука. Искусство. Культура. – 2016. – № 2 (10) 2016. – С. 152-157 ; То же [Электронный ресурс]. – Режим доступа: https://www.elibrary.ru/item.asp?id=26561655

8. *Волынникова, А. А.* Программирование мини-игры в программе Скретч [Электронный ресурс] / Открытый урок. 1 сентября – Режим доступа: https://urok.1sept.ru/%D1%81%D1%82%D0%B0%D1%82%D1%8C%D0%B8/673699/ – Дата доступа: 10.11.2019

9. *Голиков Д.* Книга юных программистов на Scratch / Д. Голиков, А. Голиков. – Изд. Smashwords, 2013. – 140 с.

10. *Грыгаровіч, Я. Д.* Прыкладная культуралогія / Я.Д.Грыгаровіч, А.І. Смолік. – Мн.: Адукацыя і выхаванне. – 2005. – 215 с.

11. *Григорьева, Л. Ю.* Практики игрофикации в массмедиа: сопряженность персонального и имперсонального / Л. Ю. Григорьева // Гуманитарные исследования в Восточной Сибири и на Дальнем Востоке. – 2013. – № 4. – С. 92–102.

12. Дизайн. История, современность, перспективы: учеб. пособие / В. И. Куманин [и др.] ; под общ. ред. И. В. Голубятникова. – М.: Аванта+, Астрель, 2011. – 224 с.

13. Игровая Индустрия [Электронный ресурс] / Газета "Detroit News" 10/27/18 – Режим доступа: https://aminoapps.com/c/detroitstat/page/blog/detroit-news/x0Yj_zbh2uxX147d6pwGmxB5Qd1qxvYjN6. – Дата доступа: 10.08.2019

14. Игровой движок Unity 3-D. Курс обучения [Электронный ресурс] // GagesIsArt.ru. Компьютерные игры как искусство – Режим доступа: http://gamesisart.ru/game_dev_unity.html. – Дата доступа: 06.08.2019.

15. Жанры игр: классификация, самый полный список [Электронный ресурс] // COREMISSION. Про игры и геймдев. – Режим доступа: https://coremission.net/zhanry-komputernuh-igr. – Дата доступа: 06.10.2019.

16. Жанры компьютерных игр [Электронный ресурс] / КомиВики. Сыктывкарский государственный университет – Режим доступа: http://komiwiki.syktsu.ru/index.php. – Дата доступа: 06.10.2019.

17. *Жилинская, Т.С* Теоретические основы информационных технологий : учебно-методический комплекс / Т.С.Жилинская, П.В.Гляков, Т.И.Песецкая. – Минск : БГУКИ, 2017. – 319 с.

18. *Кайуа, Р.* Игры и люди ; Статьи и эссе по социологии культуры / Р. Кайуа. – М. : ОГИ, 2007. – 504 с.

19. *Каманкина, М. В.* Видеоигры: общая проблематика, страницы истории, опыт интерпретации / Мария Каманкина. – Москва : [б. и.], 2016. – 338 с.

20. Классификация жанров компьютерных игр [Электронный ресурс] // GamesIsArt.ru: Компьютерные игры как искусство – Режим доступа: http://gamesisart.ru/janr.html. – Дата доступа: 15.09.2019.

21. *Коноплева, И.А.* Информационные технологии / И.А. Коноплева, О.А. Хохлова, А.В. Денисов. – М.: Проспект, 2011. – 328 с.

22. *Корпан, Л.* Компьютерная графика и дизайн / Л. Корпан, В.Тозик. – М. : Academia, 2014. – 208 с.

23. Кукин, Д.П. Индустрия компьютерных игр. Лабораторный практикум: пособие / Д.П. Кукин, Т.А. Рак, О.О. Шатилова. – Минск : БГУИР, 2019. – 60 с.

24. *Кунцевич, О.М.* Информационные технологии в культуре [Электронный ресурс]: учебно-методический комплекс для специальности 1–21 04 01 Культурология (по направлениям) / Белорусский государственный университет культуры и искусств, Факультет культурологии и социокультурной деятельности, Кафедра информационных технологий в культуре ; сост.: Т. Д. Орешко [и др.]. – Электронные текстовые данные. – Минск, 2017. – 459 с. : табл. – Заголовок с экрана. – Депонировано в БГУКИ 31.10.2017, № 014631102017. – URL:http://repository.buk.by/123456789/14552

25. *Лесной, Д. С.* Игра [Электронный ресурс] // Megabook. Универсальная энциклопедия Кирилла и Мефодия – Режим доступа: https://megabook.ru/article/%D0%98%D0%B3%D1%80%D0%B0– Дата доступа: 15.09.2019

26. *Луговая, Т.* Проектная документация для создания компьютерных игр [Электронный ресурс] / Тренинг GameHub: проектирование компьютерных игр для обучения – Режим доступа: http://gamehub-cbhe.eu/wp-content /uploads/2018/10/GameHub_Local_Training_ONPU_Project-documentation-games.pdf – Дата доступа: 15.09.2019

27. *Немцова, Т.* Компьютерная графика и Web-дизайн. Практикум по информатике / Т. Немцова, Ю. Назарова. – М. : Форум, 2013. – 288 с.

28. *Нильсен, Я.* Веб-дизайн / Я.Нильсен. – СПб. : Символ, 2015. – 512 с.

29. *Папанек, В*. Дизайн для реального мира / В. Папанек. – М. : Аронов Д., 2017. – 72 с.

30. *Пигулевский, В. О.* Дизайн и культура / В.О. Пигулевский. – Харьков : Гуманитар. Центр, 2014. – 313 с.

31. *Прахов, А.*Blender. 3D-моделирование и анимация: руководство для начинающих (+CD) / А.Прахов. – СПб. : БХВ-Петербург, 2009. – 256 с.

32. Программирование в Scratch [Электронный ресурс] / Лаборатория линуксоида. – Режим доступа: https://younglinux.info/scratch/introduction. – Дата доступа: 15.09.2019.

33. *Розенсон, И.А.* Основы теории дизайна: учебник для вузов / И.А. Розенсон. – СПб. : Питер, 2013. – 252 с.

34. *Сахнов, К., Уточкин, В.* Игровая индустрия: геймдев (GAMEDEV) [Электронный ресурс] / Высшая школа бизнес-информатики – Режим доступа: https://hsbi.hse.ru/articles/igrovaya-industriya-geymdev. – Дата доступа: 10.08.2019.

35. Схема жанров компьютерных игр [Электронный ресурс] – Режим доступа: http://www.gamer.ru/everything/shema-zhanrov-kompyuternyh-igr. – Дата доступа: 10.08.2019.

36. *Сырых, Ю.* Современный веб-дизайн. Настольный и мобильный / Ю. Сырых. – М. : Вильямс, 2014. – 384 с.

37. *Торн, Алан* Искусство создания сценариев в Unity : [рус.]. – СПб. : ДМК, 2016. – 362 с.

38. *Хейзинга, Й.* Homo Ludens : статьи по истории культуры / Й. Хейзинга. – М. : Прогресс-Традиция, 1997. – 416 с.

39. *Хокинг, Джозеф.* Unity – в действии. Мультиплатформенная разработка на C# : [рус.]. – 2-е изд. – СПб. : Питер, 2016. – 336 с.

40. *Чувиков, Д. А.* Разработка игрового виртуального симулятора=Development of the virtual game simulator : монография : [16+] /

Д.А. Чувиков. – Москва : БИБЛИО-ГЛОБУС, 2017. – 164 с. – Режим доступа: по подписке. – URL: http://biblioclub.ru/index.php?page=book&id=498912. – Текст : электронный.

41. *Шапошникова С.* Введение в Scratch [Электронный ресурс] // Единое окно доступа к информационным ресурсам – Режим доступа: http://window.edu.ru/catalog/pdf2txt/056/78056/58975. – Дата доступа: 10.11.2019.

42. CINEMA 4D Практическое руководство [Электронный ресурс] // Программирование с использованием OPENGL. – Режим доступа: http://www.opengl.org.ru/cinema-4d-prakticheskoe-rukovodstvo/cinema-4d-prakticheskoe-rukovodstvo-str3.html. – Дата доступа: 15.09.2019.

43. GameMaker.Studio. Конструкторы игр [Электронный ресурс]. – Режим доступа: http://game-maker.ru/infusions/pro_download_panel/download.php ?catid=31. – Дата доступа: 15.05.2019.

44. Scratch. Информационный портал [Электронный ресурс]. – Режим доступа: https://scratch.mit.edu. – Дата доступа: 10.08.2019.

45. UX-дизайн: идея – эскиз – воплощение / С.Гринберг, Ш.Карпендэйл, Н.Маркардт [и др.]. – СПб..: Питер, 2014. – 272 с.

5.4 Список дополнительной литературы

1. *Аликин, В. А.* Идеи игры в философии постмодернизма / В. А. Аликин // Вестник Вятского государственного гуманитарного университета. – Киров : ВятГГУ, 2014. – № 11. – С. 19–26.

2. *Бонд, Р.* Домашний мультимедийный компьютер на все 100% / Ралф, Бонд ; пер. с англ. Н.Ю.Васильева. – М. : НТ Пресс, 2007. – 352 с.

3. *Евсеев, Д.*Web-дизайн в примерах и задачах / Д.А. Евсеев, В.В. Трофимов. – М.:КноРус, 2016. – 272 с.

4. *Зеньковский, В.* 3D-эффекты при создании презентаций, сайтов и рекламных видеороликов / В. Зеньковский. – СПб.: БХВ-Петербург, 2011. – 512 с.

5. *Жаркова, Л.С.* Деятельность учреждений культуры : учеб.пособие / Л.С. Жаркова. – 3-е изд., испр. и доп. – М.: МГУКИ, 2003. – 225с.

6. История появления аркадных автоматов [Электронный ресурс] / Arkada.by– Режим доступа: https://rion-by.wixsite.com/arkade/blank-ojdkj– Дата доступа: 15.05.2019

7. *Кайуа, Р.* Игры и люди; Статьи и эссе по социологии культуры / Р. Кайуа. – М. : ОГИ, 2007. – 504 с.

8. *Кирсанов, Д.* Веб-дизайн / Д. Кирсанов. – СПб.: Символ, 2015. – 368 с.

9. *Киселев, С.* Веб-дизайн: учеб.пособие / С.В.Киселев, С.В. Алексахин, А.В. Остроух. – М.: Академия, 2014. – 64 с.

10. *Киселев, С.* Средства мультимедиа / С. Киселев. – М.: Academia, 2012. – 64 с.

11. *Крам, Рэнди*Инфографика. Визуальное представление данных / Р. Крам. – Вятка: Аквариум-Принт, 2014. – 127с.

12. *Мидлтон, К.,Херриотт, Л.*Клипарт.Использование готовых изображений в профессиональном дизайне / Крис Мидлтон, Люк Херриотт. – АСТ, 2015. –176 с.

13. *Нагибина, М* Волшебная азбука. Анимация от А до Я / М. Нагибина. – М: Перспектива, 2011. – 148 с.

14. Новые аудиовизуальные технологии : [учебное пособие] / [авт. коллектив: О. В. Грановская и др.] ; отв. ред. К. Э. Разлогов. – Москва : Едиториал УРСС, 2005. – 481, [1] с.

15. Об информации, информатизации и защите информации: ЗаконРесп. Беларусь, 10 нояб. 2008 г., № 455-З // Консультант Плюс: Беларусь. Технология 3000 [Электронный ресурс] / ООО «ЮрСпектр», Нац.центр правовой информ. Респ. Беларусь. – Минск, 2009.

16. *Пилюгина, Е. В*. Современная социальная реальность: панмифологизация, информационные войны и кризис постмодерна / Е. В.

Пилюгина // Вестник Волгоградского государственного университета. Серия 7, Философия. Социология и социальные технологии. – 2014. – № 3 (23). – С. 7–15.

17. *Постнов, К.В.* Компьютерная графика / К.В. Постнов. –М. : МГСУ, 2009. – 249 с.

18. *Соколов, А.В.* Феномен социально-культурной деятельности / А.В. Соколов. – СПб.: СПбГУП, 2003. – 204 с.

19. *Яковлева, Е. С.* 3D-графика и видео в Photoschop CS4 Extended / Е.С. Яковлева. – СПб.: БХВ-Петербург, 2010. – 272 с.

5.5 Учебный терминологический словарь

Агон (др. греч. – борьба) – тип игр, построенных на принципе соревнования, борьбы с противником (спортивные игры, коммерческая конкуренция, система конкурсов и экзаменов).

Алгоритм – набор инструкций, описывающих порядок действий исполнителя для достижения результата решения задачи за конечное число действий. В старой трактовке вместо слова «порядок» использовалось слово «последовательность», но по мере развития параллельности в работе компьютеров слово «последовательность» стали заменять более общим словом «порядок».

Алгоритм – это последовательность математических, логических или вместе взятых операций, отличающихся детерменированностью, массовостью, направленностью и приводящая к решению всех задач данного класса за конечное число шагов".

– *Дискретность* (прерывность, раздельность) – свойство алгоритм, который представляет процесс решения задачи как последовательное выполнение простых (или ранее определенных) шагов. Каждое действие, предусмотренное алгоритмом, исполняется только после того, как закончилось исполнение предыдущего.

– *Определенность* – свойство алгоритма, при котором он должно быть четким, однозначным и не оставлять места для произвола. Благодаря этому свойству выполнение алгоритма носит механический характер и не требует никаких дополнительных указаний или сведений о решаемой задаче.

– *Результативность* (конечность) – свойство алгоритма, которое направлено на результат в решении задачи за конечное число шагов.

– *Массовость* – алгоритм решения задачи разрабатывается в общем виде, то есть, он должен быть применим для некоторого класса задач, различающихся только исходными данными. При этом исходные данные могут выбираться из некоторой области, которая называется областью применимости алгоритма.

– *Линейный алгоритм* – набор команд (указаний), выполняемых последовательно во времени друг за другом.

– *Разветвляющийся* алгоритм – алгоритм, содержащий хотя бы одно условие, в результате проверки которого ЭВМ обеспечивает переход на один из двух возможных шагов.

– *Циклический алгоритм* – алгоритм, предусматривающий многократное повторение одного и того же действия (одних и тех же операций) над новыми исходными данными. К циклическим алгоритмам сводится большинство методов вычислений, перебора вариантов.

Алеа (др. греч. – жребий) – игры, построенные на случайности, удаче, жребии и т. д., а именно, рулетка и кости, карты и скачки, биржевые спекуляции.

Аркада (англ. arcade game, arcade genre – пассаж, крытая галерея магазинов) – жанр компьютерных игр, характеризующийся коротким по времени, но интенсивным игровым процессом. Аркадной считается игра для аркадных игровых автоматов.

Браузерная игра – игра, использующая брайзерный интерфейс и обычно не требующая установки на компьютер дополнительных приложений, кроме самого браузера и иногда плагина для него. Браузерные игры можно разделить на однопользовательские, многопользовательские и массово-многопользовательские.

Визуальный роман (яп. бидзюару нобэру, от англ. visual novel) – жанр компьютерных игр, подвид текстового квеста, в котором зрителю демонстрируется история при помощи вывода на экран текста, статичных (либо анимированных) изображений, а также звукового и/или музыкального сопровождения. Нередко используются и вставки полноценных видеороликов. Степень интерактивности в таких играх обычно низка, и от зрителя лишь изредка требуется сделать определённый выбор, в частности — выбрать вариант ответа в диалоге. Персонажи этих игр обычно выполнены в стиле аниме, который, как и визуальные романы, возник в Японии. На 2012 год этот жанр остаётся популярным прежде всего в Японии, где находятся большинство ведущих разработчиков, но набирает популярность и в других странах.

Виртуальная реальность (ВР, англ. virtual reality, VR, искусственная реальность) – созданный техническими средствами мир, передаваемый человеку через его ощущения: зрение, слух, осязание и другие. Виртуальная реальность имитирует как воздействие, так и реакции на воздействие. Для создания убедительного комплекса ощущений реальности компьютерный синтез свойств и реакций виртуальной реальности производится в реальном времени.

Геймдизайнер – человек, разрабатывающий правила игр, который должен обладать навыками аналитика, психолога, технического писателя и игрока, умение работать в команде.

Геймплей – содержания игрового процесса.

Дизайн-документ («диздок») – дополненный деталями разработки и наполнения концепт-документ, после своего утверждения. Цель дизайн-документа заключается в том, чтобы однозначно описать коммерческие аспекты игры, целевую аудиторию, игровой процесс, графику, дизайн уровней, историю (сюжет), персонажей, пользовательский интерфейс.

Документ-предложение (англ.proposal document) краткое описание игры, без внутренних деталей разработки, объясняющее потенциальному инвестору, почему игра принесет прибыль.

Дополненная реальность (англ. augmented reality, AR – «дополненная реальность») – результат введения в поле восприятия любых сенсорных данных с целью дополнения сведений об окружении и улучшения восприятия информации.

Игра – это вид непродуктивной деятельности, мотив которой заключается не в ее результатах, а в самом процессе. В истории человеческого общества переплеталась с магией, культовым поведением и др.; тесно связана со спортом, военными и другими тренировками, искусством (особенно его исполнительными формами).

Игровой движок (game engine) – особая программа, которая содержит в себе множество уже готовых игровых процессов, механик, элементов. Использование игрового движка существенно сокращает время разработки новой игры.

Игровая индустрия тесно связана с производством центральных процессоров и других компонентов персональных компьютеров.

Игровые консоли представляют собой те же компьютеры, но лишь с одной функцией – воспроизведение игр.

Игровая культура – область досуговой сферы жизнедеятельности человека, которая определяется стремлением к заполнению свободного времени, не связываясь с другими видами человеческой деятельности.

Игровая приставка (*игровая консоль*) – специализированное электронное устройство, предназначенное для видеоигр. Основная задача заключается в запуске и воспроизведение видеоигр. Домашние игровые приставки используют телевизор, проектор или компьютерный монитор в качестве независимого устройства отображения.

Игровой дизайн (также геймдизайн, англ.gamedesign) – процесс создания формы и содержания игрового процесса (геймплея) разрабатываемой игры. Игровой дизайн определяет: набор возможных вариантов, из которых игрок может выбирать во время игры; условия победы и поражения; как игрок контролирует происходящее в игре; как взаимодействует с игровым миром; сложность игры и др.

Игровой цикл (Game loop) – описание последовательности действий, которые совершает игрок постоянно, каждую игровую сессию.

Игровые сообщества – объединение игроков по интересам на форумах популярных общеигровых сайтов, а также вокруг спонтанно появившегося сообщества, основанного на отдельной игре или игровой серии, в следствии чего начинают появляться новые сайты, форумы, интернет-группы.

Игрок – потенциальный участник игровой индустрии.

Избей их всех (англ. *beat 'em up* или beat 'em all, также brawler — «драчун») – жанр компьютерных игр, основной чертой которого является рукопашная схватка главного героя против огромного количества врагов. Как правило, действие таких игр происходит в городском антураже, а сюжет основывается на темах борьбы с преступностью или отмщения, однако встречаются игры, основанные на исторической, научно-фантастической или фентезийной тематике.

Издательские компании занимаются продажей созданных компьютерных игр. К задачам издательства можно отнести полное материально-техническое обеспечение процесса разработки игры.

Иллинкс (др. греч. – головокружение) – тип игры, который связан с интенсивным, форсированным изменением состояния сознания. Среди развлечений сюда относятся качели, карусели, гигантские шаги

Индустрия компьютерных игр – сектор экономики, связанный с разработкой, продвижением и продажей компьютерных игр.

Карточная игра — игра с применением игральных карт, характеризуется случайным начальным состоянием, для определения которого используется набор (колода) карт. Существует также множество наборов карт, созданных под конкретные игры. Процесс определения начального состояния каждого тура игры называется раздачей карт и состоит в раскладывании определенного правилами игры количества карт по определенным местам.

Квест (англ. quest), или приключенческая игра (англ. adventure game) — один из основных жанров компьютерных игр, представляющий собой интерактивную историю с главным героем, управляемым игроком. Важнейшими элементами игры в жанре квеста являются собственно повествование и исследование мира, а ключевую роль в игровом процессе играют решение головоломок и задач, требующих от игрока умственных усилий. Такие характерные для других жанров компьютерных игр элементы, как бои, экономическое планирование и задачи, требующие от игрока скорости реакции и быстрых ответных действий, в квестах сведены к минимуму или вовсе отсутствуют.

Киберспорт (компьютерный спорт или электронный спорт) — командное или индивидуальное соревнование на основе видеоигр. Все киберспортивные дисциплины делятся на несколько основных классов, различаемых свойствами пространств, моделей, игровой задачей и развиваемыми игровыми навыками киберспортсменов.

Книга-игра (англ. gamebook) – литературное произведение, которое позволяет читателю участвовать в формировании сюжета. Чаще всего читателю предлагается стать главным героем книги, и, в зависимости от

принимаемых решений, он перемещается между страницами или главами. Таким образом, книга-игра читается не последовательно, а в той очерёдности, в которой читатель проходит страницы или главы.

Компьютерная игра – компьютерная программа, служащая для организации игрового процесса (геймплея), связи с партнёрами по игре, или сама выступающая в качестве партнёра.

Консольная игра (приставочная игра) (англ. console game) – компьютерная игра, которая предназначена для работы на игровой консоли (игровой приставке). Версия мультиплатформенной компьютерной игры для консоли называется приставочной (консольной) версией игры.

Конструирование миров – процесс создания образа мира в человеческой психике. Психологический термин, который ввел отечественный психолог А. Г. Асмолов.

Контент – информационное содержание сайта (тексты, графическая, звуковая информация и др.), а также книги, газеты, сборника статей, материалов и др.

Концепт-арт – направление в искусстве, предназначенное для того, чтобы визуально передать идею произведения, но не форму или внешние атрибуты. Как правило, создаётся на начальной стадии разработки проекта и предназначается для использования в компьютерных играх, комиксах до создания финальной версии. Также называется «концепт-дизайном».

Концепт-документ – это короткий документ, который относительно детально описывает вашу игру, раскрывает основные особенности игры, обычно 2-6 страниц текста, по возможности, разбавленных иллюстрациями.

Концепция – это текстово-графическое описание ключевого замысла проекта.

Королевская битва (англ. Battle Royale) — один из жанров массовых многопользовательских онлайн-игр, совмещающий в себе элементы симулятора выживания с режимом last man standing. «Королевская битва» сталкивает большое количество управляемых игроками персонажей с минимальным набором снаряжения на ограниченной карте; игроки должны искать на карте оружие и уничтожать противников, пока в игре не останется только один. Характерной особенностью «королевской битвы» является уменьшающаяся по мере игры «безопасная зона»: чем дольше длится матч, тем меньшая часть карты остаётся доступной для игроков. Само название жанра восходит к роману «Королевская битва» японского писателя Косюна Таками и его экранизации.

Мимикрия (др. греч. – подражание). Здесь мы имеем дело с типом игр, основанных на воспроизведении разных типов человеческой деятельности. Театр и балет, игры в куклы и шарады, церемониал, униформа.

Мобильная игра – игровая программа для мобильных устройств: сотовые телефоны, смартфоны, коммуникаторы, КПК и прочих (за исключением ноутбуков).

Музыкальная игра (англ. Rhythm game, rhythm action) – жанр компьютерных игр, где во главу ставится музыкальная составляющая, а от игрока требуется наличие чувства ритма.

Настольная игра – игра, основанная на манипуляции относительно небольшим набором предметов, которые могут целиком разместиться на столе или в руках играющих. В число настольных игр входят игры со специальным полем, карточные игры, кости, солдатики и другие. Игры данной категории, в отличие от спортивных и видеоигр, не требуют активного перемещения игроков, наличия дополнительного технически сложного инвентаря или специальных сооружений, игровых площадок, полей.

Платформер (разг. бродилка; англ. platformer, platform game) – жанр компьютерных игр, в которых основной чертой игрового процесса является прыгание по платформам, лазанье по лестницам, собирание предметов, обычно необходимых для завершения уровня. Реже предметы собираются в «инвентарь» героя и применяются специальной командой (такое поведение более характерно для аркадных головоломок). Сходный жанр компьютерных игр сайд-скроллер.

Проект – это создание чего-то нового для решения проблемы или для достижения определенной цели.

Ролевая игра (РПГ) представляет собой моделирование событий, происходящих в определённом мире в определённое время. Её участники отыгрывают собственных персонажей, руководствуясь при этом характером своей роли и внутренними убеждениями персонажа в рамках игровых реалий. Индивидуальные и коллективные действия игроков составляют сюжет игры. Как правило, существуют правила проведения ролевой игры, где описаны рамки действий игроков, их поведения, моделирования игровых ситуаций. Действия игроков представляют из себя вольную импровизацию в рамках выбранных правил, а также определяют суть игры и её результат.

Симулятор – имитатор (обычно механический или компьютерный), задача которого состоит в имитации управления каким-либо процессом, аппаратом или транспортным средством. Компьютерные игры: симуляторы автомобилей и мотоциклов, самолётов, космических кораблей, танков, поездов, подлодок, различных видов спорта и т. п. Основным принципом симулятора является точное воспроизведение особенностей какой-то тематической области (например: автосимулятор должен максимально точно воспроизводить физические особенности машин).

Скретч (англ. Scratch) – визуальная событийно-ориентированная среда программирования, созданная для детей и подростков. Название произошло от слова scratching – техники, используемой хип-хоп-диджеями.

Социальные игры – это игры, направленные на социальное взаимодействие. Не все игры в социальных сетях на него направлены, поэтому не любую игру там можно назвать социальной. Цель социальной игры – индивидуальный прогресс с помощью коллектива.

Стратегическая игра – жанр компьютерных игр, где игроку для достижения цели необходимо применять стратегическое мышление, и оно противопоставляется быстрым действиям и реакцией, которые, как правило, не обязательны для успеха в таких играх. Стратегические игры бывают абстрактные, настольные (шахматы, шашки, «Монополия»), с симуляцией менеджмента (M.U.L.E., Spaceward Ho!) и другие.

Таймкиллеры – простые (или не очень) игрушки, предназначенные для непродолжительной игры, то есть занять десять-пятнадцать минут между поездками, занятиями или рабочими заданиями. Обычно предлагают элементарные действия — как в Angry Birds или Hopeless.

Текстовые квесты – требуется только выбор вариантов продвижения по сюжету. Например, игрок получает следующее сообщение: «Хозяин отеля предлагает вам поесть», и варианты ответа: «согласиться», «отказаться». Если он выбрал вариант «согласиться», его персонаж отравляется в трапезную комнату. Там после отравления он оказывается в темнице, из которой придётся выбираться. Если же игрок отказался от еды, то его персонаж продолжает свой путь и т. д. Текстовые квесты существуют как в электронном варианте, так и в книжном.

Текстовые симуляторы – это игры, в которых существует экономическая система. Зарабатывая деньги, игрок продвигается по сюжету. Например, если проходить жизнь хозяина компании по созданию компьютерных игр, то игрок должен зарабатывать деньги на новое оборудование, при этом следить, чтобы никто не увольнялся, не умереть с голоду и т. д. Игры представляют собой совмещение симулятора экономики и стратегии.

Текстовые RPG – в играх этого жанра игрок может развивать характеристики персонажа, получать заклинания, броню, оружие и продвигаться по сюжету, выбирая, куда двигаться. Всё проходит в следующем виде – игроку задаётся вопрос: «Вы пришли к развилке трёх дорог в лесу. Куда вы повернёте?» Игрок должен выбрать один из вариантов ответа («налево», «направо», «вперёд»). При столкновении с противником игрок имеет ряд возможных вариантов действия.

Технический дизайн-документ (technicaldesigndocument) – раскрывает технические требования к игре (объём памяти, используемые утилиты, языки программирования, базы данных).

Файтинг (от англ. Fighting – бой, драка, поединок, борьба) – жанр компьютерных игр, имитирующих рукопашный бой малого числа персонажей в пределах ограниченного пространства, называемого ареной. Важной особенностью является нацеленность на соревнование, а не на сотрудничество игроков. Обычно файтинги предоставляют игроку возможность вести бой в режиме «один на один» против компьютерного противника или другого игрока, реже – позволяют сражаться одновременно трём или четырём противникам на одной арене.

Фич (feature) – это игровая механика, особенность игры.

Цикл программы – последовательность команд (серия, тело цикла), которая может выполняться многократно (для новых исходных данных) до удовлетворения некоторого условия.

Шутер (стрелялка, англ. shooter — «стрелок») – жанр компьютерных игр где игрок находится в трёхмерном пространстве и имеет некоторую свободу передвижения. Уровни, как правило, являют собой ограниченный лабиринт, в котором расположены враги, союзники и нейтрально настроенные не игровые персонажи (NPC).

Экшен (action в переводе с англ. – «действие») – жанр компьютерных игр, в котором делается упор на эксплуатацию физических возможностей игрока, в том числе координации глаз и рук и скорости реакции. Жанр представлен во множестве разновидностей от файтингов, шутеров и платформеров, которые считаются наиболее важными для жанра, до MOBA и некоторых стратегий в реальном времени, которые возможно отнести к жанру экшен. Игрок управляет персонажем, который должен найти выход из уровня, собрать предметы, избежать препятствий и сразиться с врагами разными способами

Языковая игра (нем. Sprachspiel) – термин Людвига Витгенштейна, введённый им в «Философских исследованиях» 1945 года для описания языка как системы конвенциональных правил, в которых участвует говорящий. Понятие языковой игры подразумевает плюрализм значений. Концепция языковой игры приходит на смену концепции метаязыка.

MOBA (Multiplayer Online Battle Arena – «многопользовательская онлайновая боевая арена») – жанр компьютерных игр, сочетающий в себе элементы стратегий в реальном времени и компьютерных ролевых игр. В играх жанра MOBA две команды игроков сражаются друг с другом на карте особого вида. Каждый игрок управляет одним персонажем из определенного списка доступных героев, отличающихся характеристиками и

способностями. В течение матча персонажи могут становиться сильнее, получать новые способности и снаряжение, подобно компьютерным ролевым играм. Конечной целью в ходе матча является уничтожение главного здания вражеской команды с помощью как «героев», управляемых игроками, так и «крипов», управляемых компьютером.

MUD (Multi User Dungeon, Dimension или Domain, русский вариант — МПМ (многопользовательский мир), также используются аббревиатуры МУД и МАД) – текстовая многопользовательская компьютерная игра, в которой присутствуют элементы ролевой игры, hack-and-slash, interactive fiction и обязательно чат, как правило разделённый на каналы. Традиционно для передачи сообщений используется протокол telnet, отдельную разновидность составляют JMUD — игры через протокол Jabber (XMPP).

USP (Unique Selling Points) – это уникальные фичи, которые делают игру особенной и которые должны привлечь игрока, включают какие-то инновационные идеи, или необычное сочетание уже известных игровых механик.