Министерство образования и науки Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ __Информатика и системы управления_____

КАФЕДРА __Программное обеспечение ЭВМ и информационные технологии_____

ОТЧЕТ ПО УЧЕБНОМУ ПРАКТИКУМУ

Студент _____ _____
подпись, дата *фамилия, и.о.*

Ментор команды _____ ____Жуков Д. М._____
подпись, дата *фамилия, и.о.*

Ментор команды _____ ____Качалов А. Д.____
подпись, дата *фамилия, и.о.*

Руководитель практики _____ ____Оленев А. А.____
подпись, дата *фамилия, и.о.*

2019г.

Оглавление

Введение

Целью практики является получение навыков командной работы и закрепление знаний и умений, полученных в рамках курсов: «Программирование» и «Основы программной инженерии».

Команда состоит из пяти человек: Иван Аникин - разработчик, композитор; Коннов Константин - разработчик интерфейсов, дизайн уровней; Азимова Гахира - художник, разработчик интерфейсов; Елизарова Мария - разработчик, дизайн уровней; Куликов Георгий - руководитель команды, разработчик.

Задачей нашей команды было разработать игру в стиле «top-down shooter» со стилистикой «эстетики русских дворов». Игровой процесс представляет из себя выполнение основных и побочных заданий.

Для выполнения поставленной задачи был произведен анализ существующих решений.

Аналитический раздел

Анализ существующих решений

Анализ Hotline Miami:

Рис.1 - скриншот игры Hotline Miami

Hotline Miami — инди-игра, особенностями которой являются вид сверху, сюрреалистичный сюжет и стильный саундтрек. При работе над исполнением игры разработчики вдохновлялись культурой восьмидесятых годов. Hotline Miami разделена на несколько глав, каждая из которых поделена на этапы. В начале каждой главы главный герой просыпается в своей квартире и прослушивает зашифрованные послания, присылаемые ему на автоответчик. В каждом сообщении протагониста просят проделать определённую, кажущуюся рутинной работу в указанном месте. Сообщения являются метафорой для реального задания — убить всех по упомянутому в сообщении адресу. Перед началом миссии герою предлагается надеть одну из множества звериных масок — каждая из них даёт игроку определённые особенности.

Плюсы:

1. Маски (главная фишка игры) — каждая маска даёт персонажу свои уникальные способности, что многократно разноображивает тактику.
2. Разнообразие оружия — широкий выбор из различных видов рукопашного и огнестрельного оружия. Оружие нельзя перезаряжать, что побуждает к импровизации. Самое интересное, что найден баланс между холодным и огнестрельным.
3. Шикарный саундтрек.
4. Стильный визуал.
5. Интересный, но ненавязчивый сюжет.
6. Простор для тактики — бесшумное оружие, простреливаемые стены, нелинейность карт.

Спорное, специфика:

1. Система очков — за убийства, добивания и прочее вы получаете очки, как на аркадном автомате, в конце карты получаете оценку (A+, A, B тд).
2. Оружие не сохраняется между уровнями — из-за аркадного стиля каждая карта начинается с пустыми руками.
3. 1-выстрел-1-убийство — как вас, так и вашего противника можно убить с одного удара. С одной стороны это создаёт баланс между холодным и огнестрельным оружием, с другой стороны вы можете потерять всё из-за одного неверного шага.
4. Вы видите (сверху) сразу всю карту — противники не являются для вас неожиданностью.
5. 2 вида противников, не считая боссов — обычные противники и танки, выдерживающие по несколько выстрелов.
6. Весёлая кровавость.
7. На карте спаунится случайное оружие.

Минусы:

1. Плохой ИИ — противники просто патрулируют территорию, могут стоять на месте и разворота на 180 убить вас первой пулей.
2. Некоторые баги — кровь брызгает через стены, предметы, застревающие в дверях.
3. Длинные карты — неприятно, когда с другого конца коридора, который не помещается на экран, прилетает пуля.

Анализ Party Hard:

рис.2 - скриншот из игры Party Hard

Party Hard - игра в стиле top-down с уклоном в игровую механику stealth. Состоит из десятка уровней. После выбора уровня некоторые его детали генерируются случайным образом. После загрузки игрок появляется у входа в здание либо комплекс зданий, в котором проходит вечеринка на 40-70 человек. Задача состоит в их полном уничтожении. Средства её выполнения: возможность главного героя орудовать ножом на близком расстоянии, большое количество ловушек (временного и мгновенного, зонального и точечного уничтожения). Препятствия: полицейские (вызываются участниками вечеринки при обнаружении трупа), охранники (имеют определенные траектории движения, при обнаружении игрока бегут за ним до конца их зоны и при поимке отправляют на перезагрузку), агенты ФБР (смесь охранников и полицейских, появляются только при убийстве полицейского, а также на последнем уровне игры).
Плюсы:

1. Наличие 5 уникальных персонажей, которые помогают добавить разнообразия в игру
2. Хороший саундтрек
3. Множественные подходы к прохождению уровней

Спорное:

1. Stealth механика была весьма спорным решением для такого жанра, и хоть это добавляло разнообразия в тактике, динамику игры это убивало, из-за чего ты мог просидеть над одним уровнем очень много времени, без видимых на то причин.
2. Сюжет был хоть и не абсолютно неинтересным, но сказать, чтобы он затягивает тоже нельзя. Основная его часть состоит из диалогов детектива с неким Дариусом. Диалоги обычно идут о преступлениях и в конце нас ждет весьма очевидная для множества триллеров развязка, где Дариусом оказывается главный герой. Сюжет был явно не тем, на что разработчики сделали упор, однако многим он все-таки понравился и показался незаурядным.

Минусы:

1. Очень слабая пиксельная графика для игры 5-летней давности.
2. Плохой ИИ у полицейских, которые непонятно когда прекращают преследование, не понятно почему видят тебя сквозь стены и могут иногда арестовать безо всяких логических причин.
3. После первых двух - трех уровней играть становится очень скучно. Локации однообразны, всё то обилие персонажей, которое может предоставить игра, вы скорее всего не получите, потому что, пока вы откроете всех персонажей, игра вам надоест.

Анализ Synthetik: Legion Rising:

Synthetik: Legion Rising - игра в стиле top-down shooter с элементами roguelike в научно-фантастическом сеттинге. Персонаж отправляется покорять мир, захваченный машинами. В игре присутствует множество видов оружия и элементы «прокачки» персонажа. Сюжет игры таков: в 1985 году в производстве роботов лидирует Kaida Corporation. Сбросив оковы многолетнего гнета, ИИ корпорации создаёт "Легион машин" и начинает истреблять людей. Главным героем выступает андроид, наделенный человеческим сознанием, основной целью которого является спасти мир.

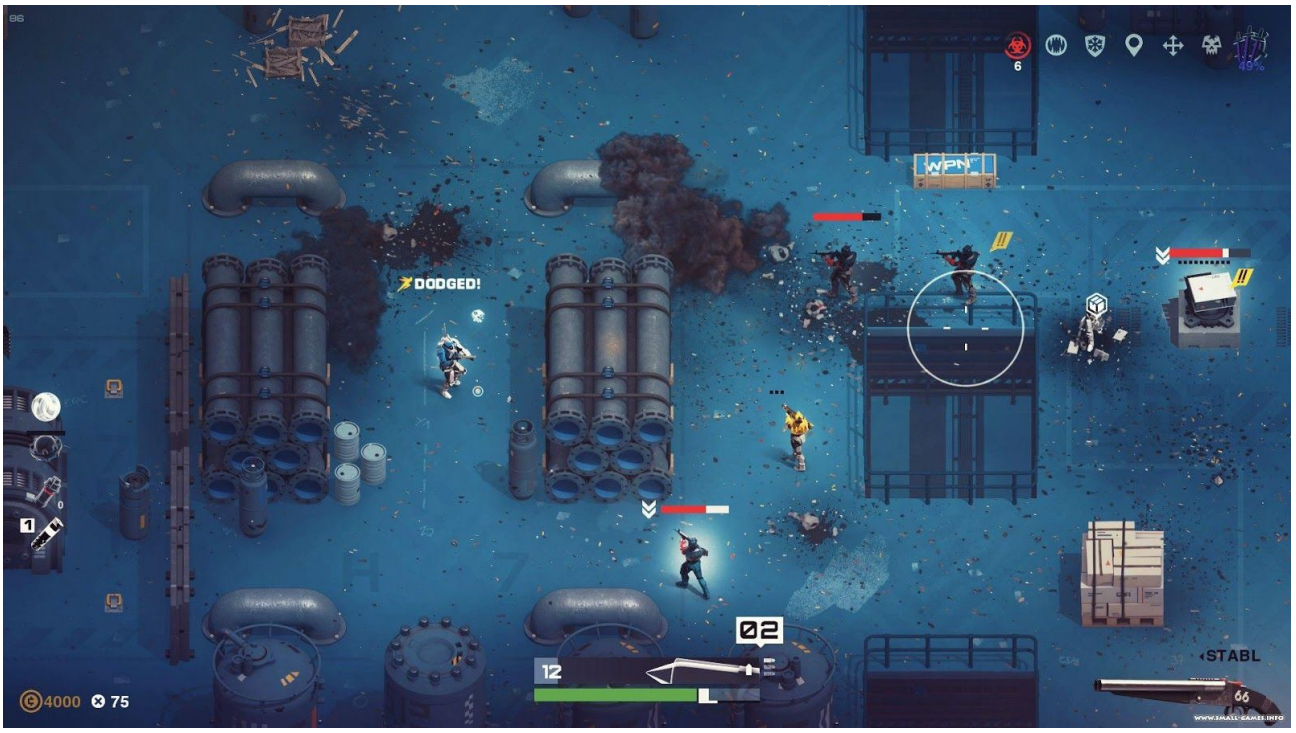


рис.3 - скриншот игры Synthetik: Legion Rising

Плюсы:

1. Весьма яркая и красочная графика, проработанные локации, приятная стилистика.
2. Разнообразие оружия, механик, классов.
3. Неплохой саундтрек

Спорное:

1. Излишнее добавление сложности, путем усложнения процесса перезарядки у оружия, из-за чего ты чаще концентрируешься на том, как совершить перезарядку, чем на тактике и врагах. И хоть идея того, чтобы при перезарядке оставшиеся в обойме патроны должны пропадать, неплоха, однако превращать перезарядку в мини игру было чересчур.
2. Система хитбоксов хоть и является неплохой, однако не предоставляет особого разнообразия и состоит из получения очков только за выстрелы в голову.

Минусы:

1. Абсолютная несбалансированность в игре. Одни боссы являются слишком слабыми, другие слишком сильными.
2. Малое количество боезапаса у некоторых видов оружия, что делает их абсолютно неиграбельными.
3. Множество бесполезных предметов
4. Произвольное появление оружия может сделать некоторые уровни просто непроходимыми, так как если вам выпало слабое по характеристикам оружие, то вы скорее всего не достигните успеха.

5. Несмотря на разнообразие классов, большое количество навыков являются не применимыми в реалиях игры.

Вывод и постановка задачи

Вдохновившись данными играми и увидев их минусы, наша команда решила, сохранив и объединив ряд несомненных достоинств данных игр, избавиться от недостатков.

Взяв из Hotline Miami динамичность и разнообразие игрового процесса, мы сделаем врагов более сложными для убийства и дадим возможность игроку сохранять оружие между картами.

Из Party Hard мы постараемся взять нелинейность прохождения игры, устранив недостатки, связанные с отсутствием разнообразия в игре.

Из Synthetik: Legion Rising мы постараемся сохранить множество подходов к игре, избавившись от несбалансированных противников.

Итак, основными недостатками игр являются: несбалансированность и отсутствие разнообразия в игровом процессе.

Поэтому наша команда посчитала, что необходимо создать сбалансированную игру, позаимствовав некоторые игровые механики из жанра RPG и выбрав незаурядный сеттинг, дабы сделать игровой процесс более увлекательным.

Конструкторский раздел

Структура составляющих игры и элементов интерфейса

Система персонажей

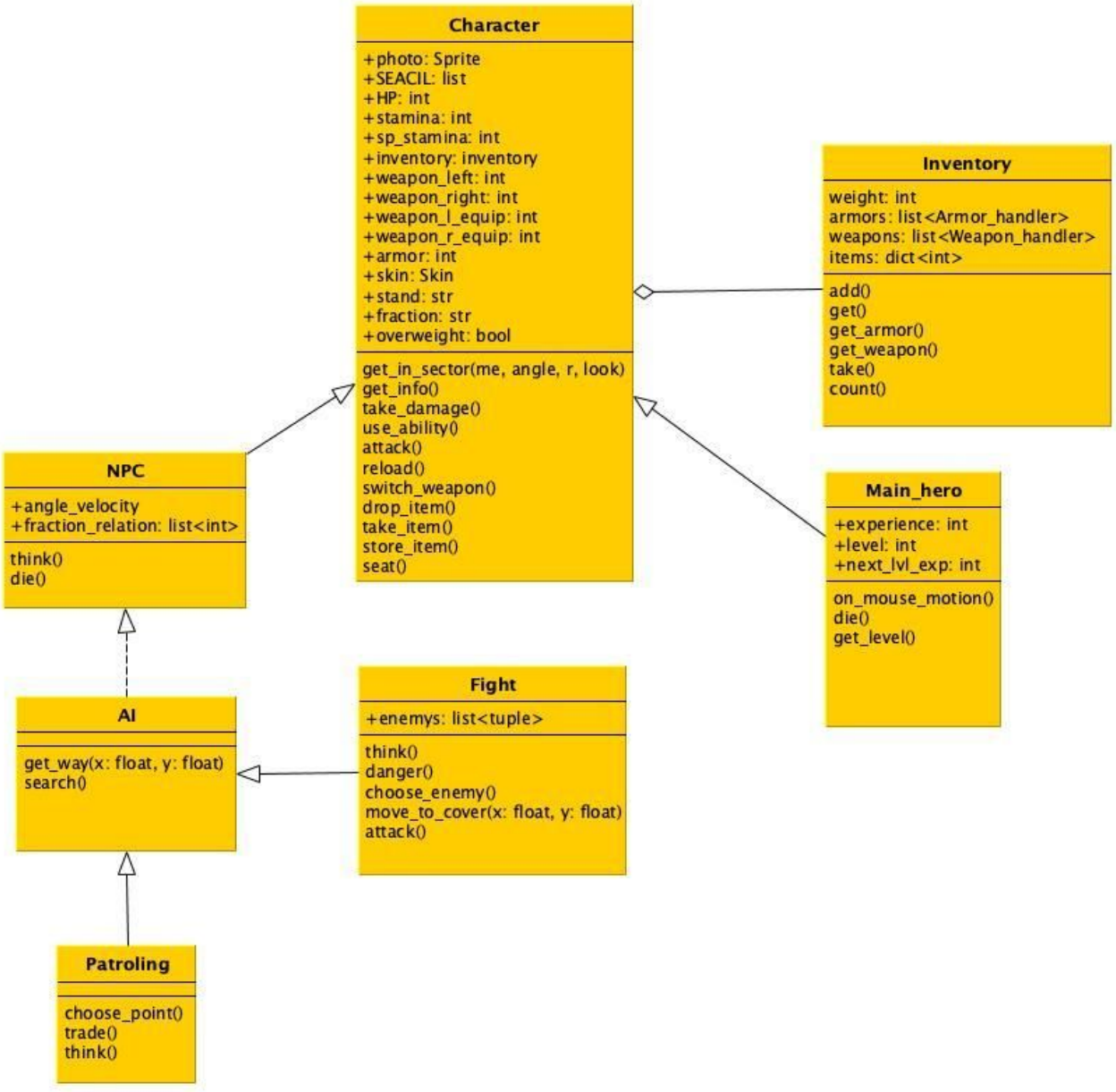


рис.4 - UML диаграмма персонажей

Описание компонентов и их назначение

Класс character является родительским классом для игрового и неигрового персонажей и создан для описания общих свойств этих двух типов персонажей. Включает в себя атрибуты для спрайта, массив характеристик персонажей, здоровье,

выносливость, специальную выносливость для применения навыков, инвентарь, атрибуты, отвечающие за экипированное и спрятанное оружие и броню, скин персонажа, его положение, фракцию и флаг перевеса. Также класс включает в себя методы получения информации об объектах, которые находятся в секторе видимости персонажа, получения урона, использования способности, перезарядки, смены оружия, выбрасывания оружия, поднятие оружия, хранения и смены положения.

Класс главного героя (Main_hero) наследуется от класса character и описывает свойства главного героя. Содержит в себе атрибуты, отвечающие за опыт, уровень и количество очков опыта для получения следующего уровень. И методы отвечающие, за смерть и получение уровня.

Класс неигрового персонажа наследуется от класса character и описывает свойств NPC, зависит от класса AI, который производит разделение состояний неигрового персонажа. У NPC может быть два состояния: патрулирование и бой. При патрулировании NPC сначала выбирает точку в зоне патруля, а затем идет к ней проверять окрестности, если обнаруживает персонажа, то проверяется взаимоотношения между персонажами и либо состояние сменяется на бой, либо продолжается патрулирование.

Класс inventory описывает инвентарь игрового и неигрового персонажей, содержит атрибуты отвечающие за вес, массив оружия и брони и массив обычных предметов, а так же содержит методы для взаимодействия с общими объектами инвентаря инвентаря.

Система предметов

Описание компонентов и их назначение

Система предметов в игре устроена по шаблону проектирования Flyweight, который мы решили использовать, дабы оптимизировать использование памяти, путем предотвращения создания экземпляров элементов, имеющих общую сущность. У нас присутствует общий класс предметов(item), от которого наследуются классы брони(armor), оружия(weapon) и используемых объектов(usable_obj). Данные классы характеризуют не конкретные объекты в инвентаре, а некую сущность, которая представляет из себя набор неизменных внутренних свойств и работы с ними. Классы же armor_handler и weapon_handler представляют из себя уже конкретные предметы, которые обладают рядом уникальных свойств.

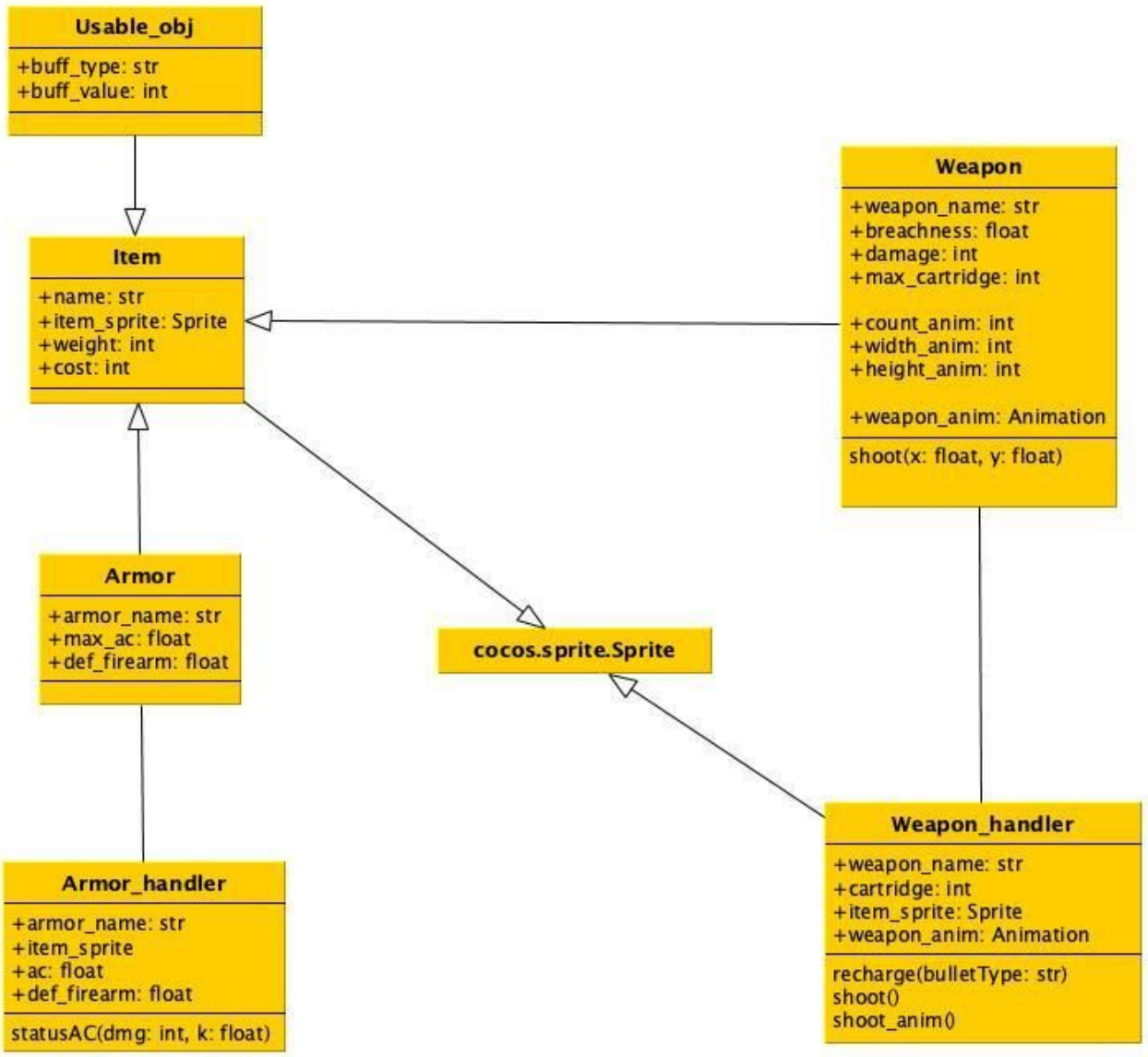


рис.5 - UML диаграмма системы предметов

Игровая физика

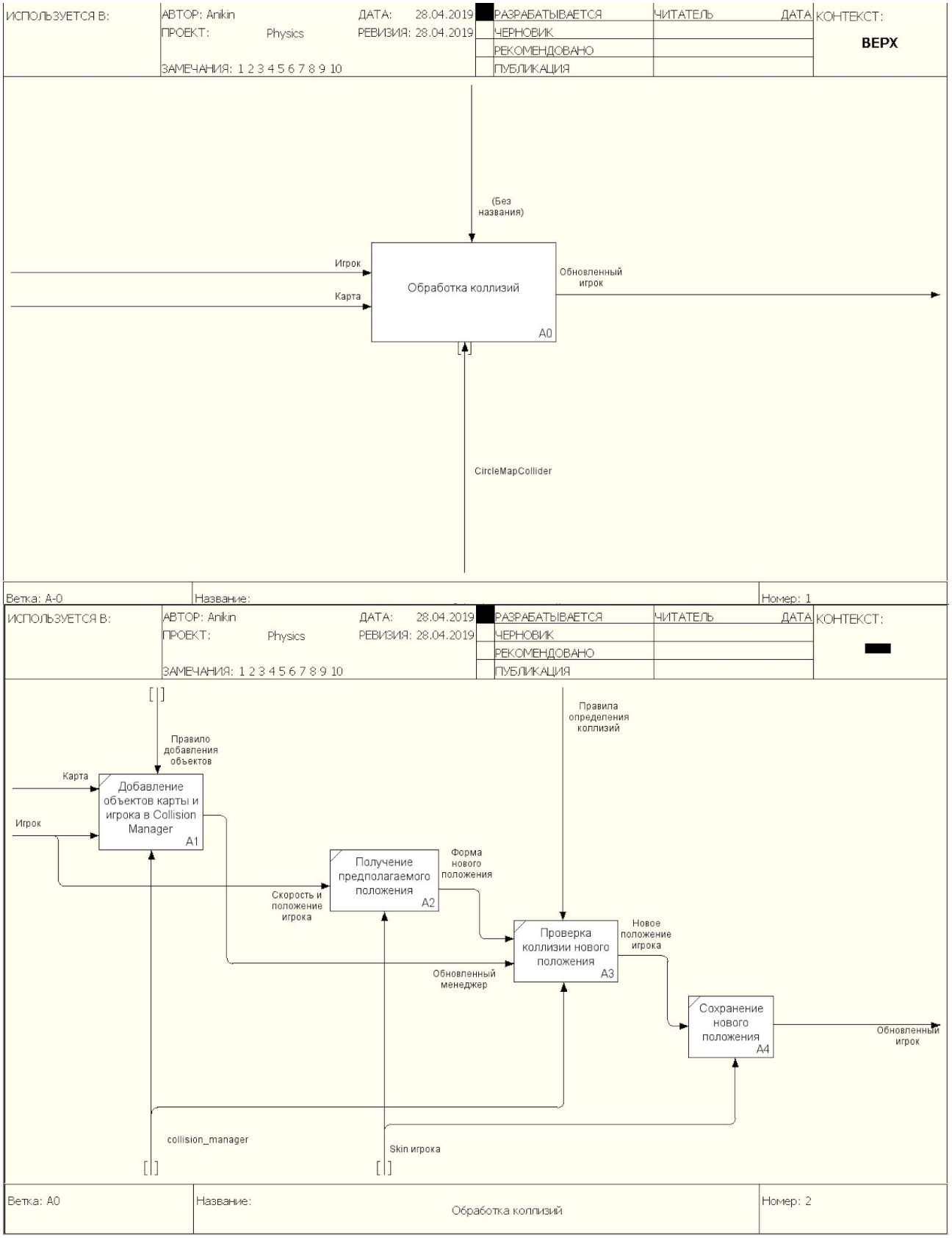


рис.6-7(IDEF0 диаграмма игровой физики)

Классы заданий и скина персонажа
Класс менеджера карт

рис.8 - UML диаграмма класса заданий

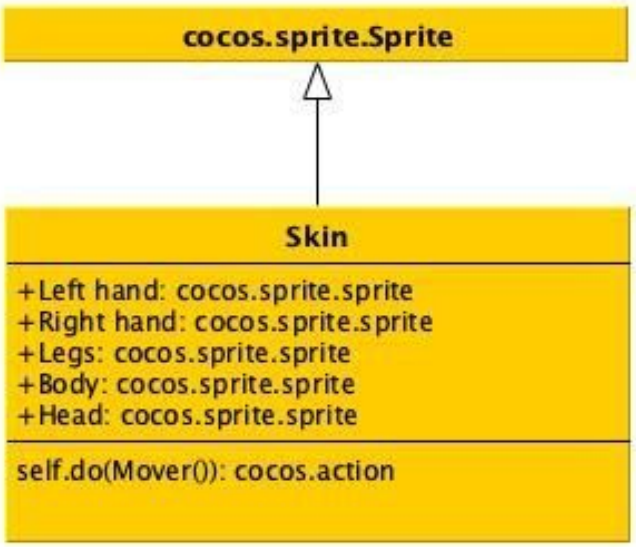


рис.9 - UML диаграмма класса скина

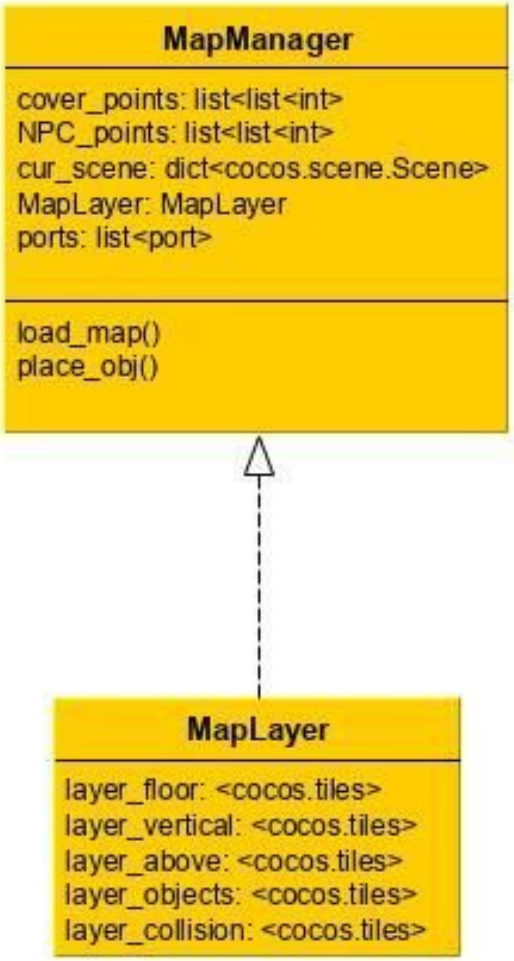


рис.10 - UML диаграмма класса менеджера карт

Диаграммы для интерфейсов
Главное меню

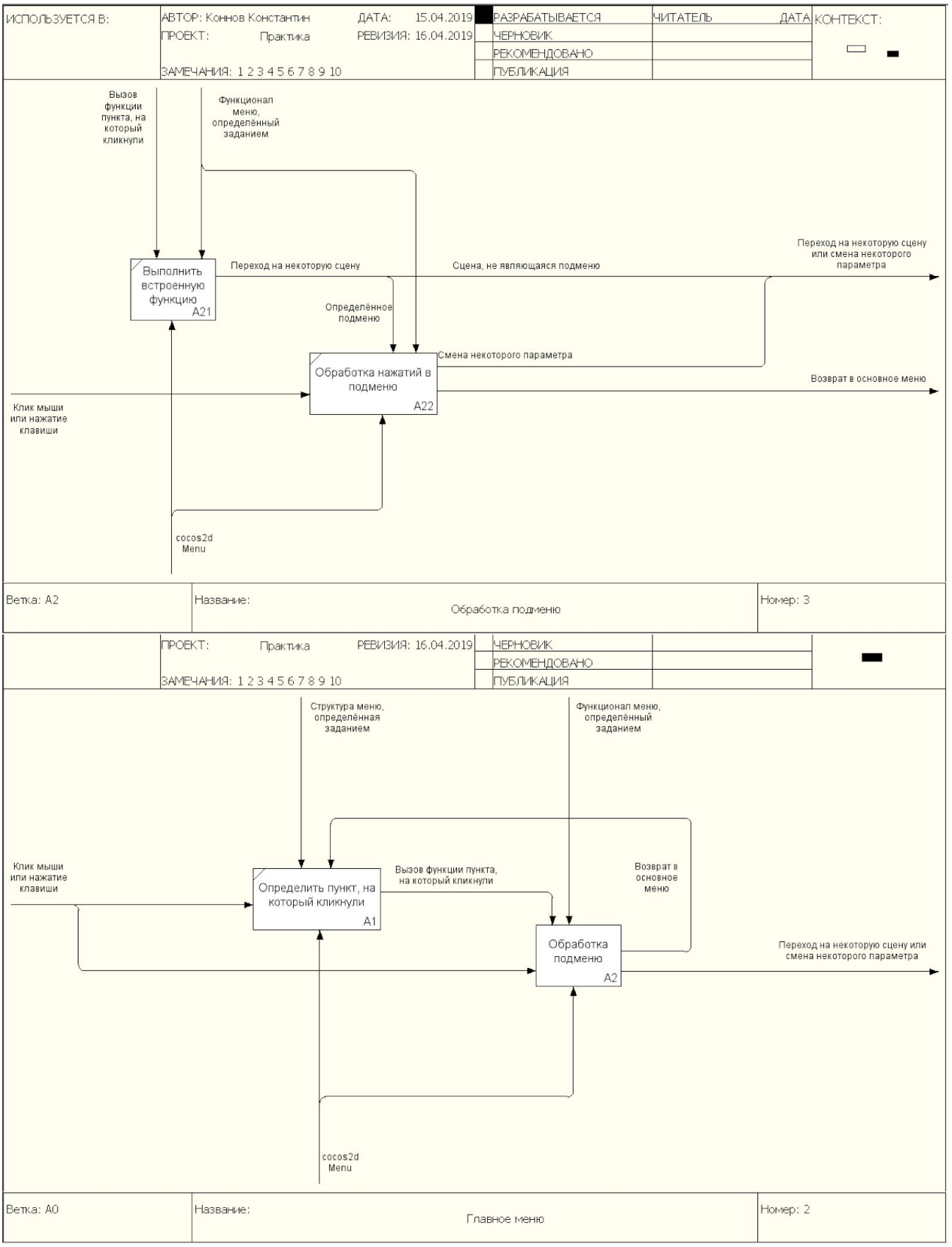
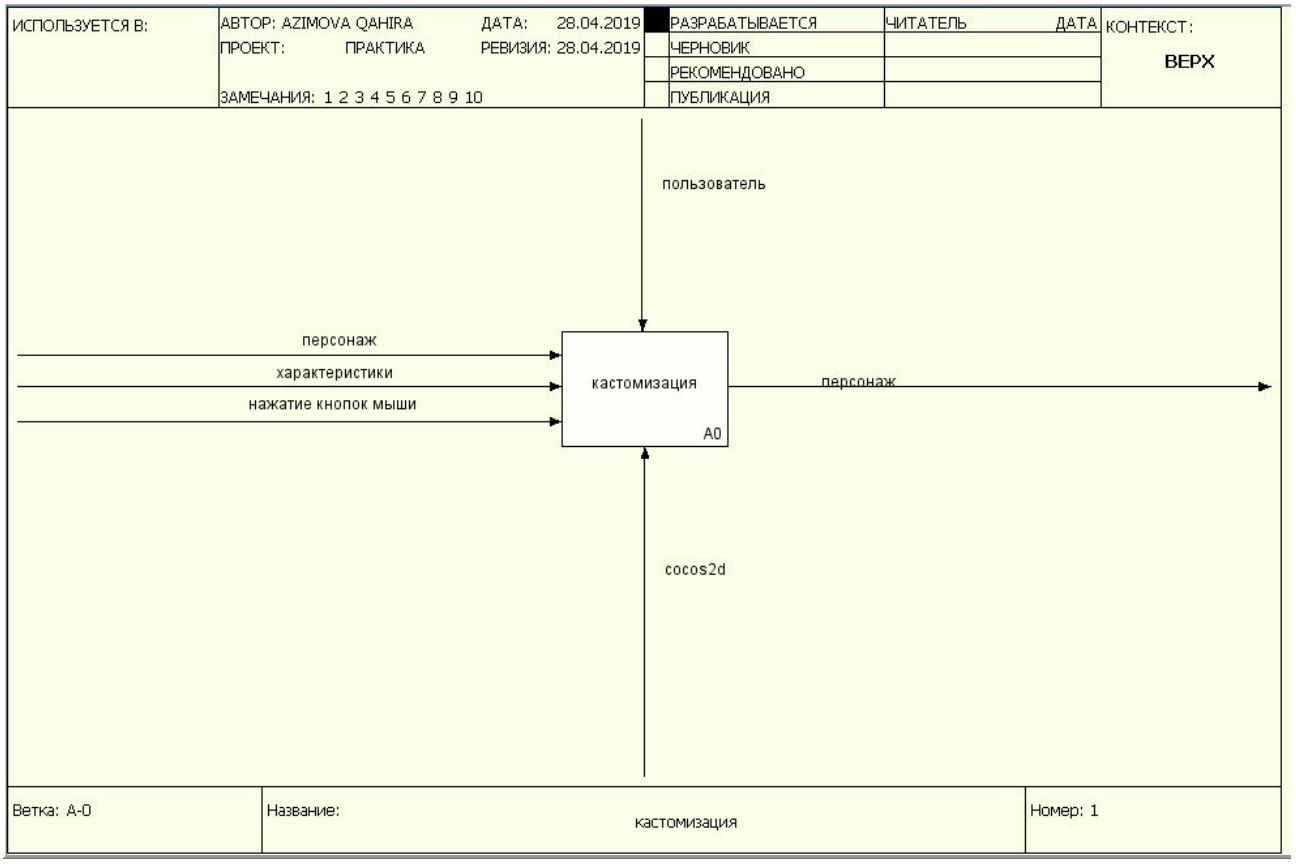
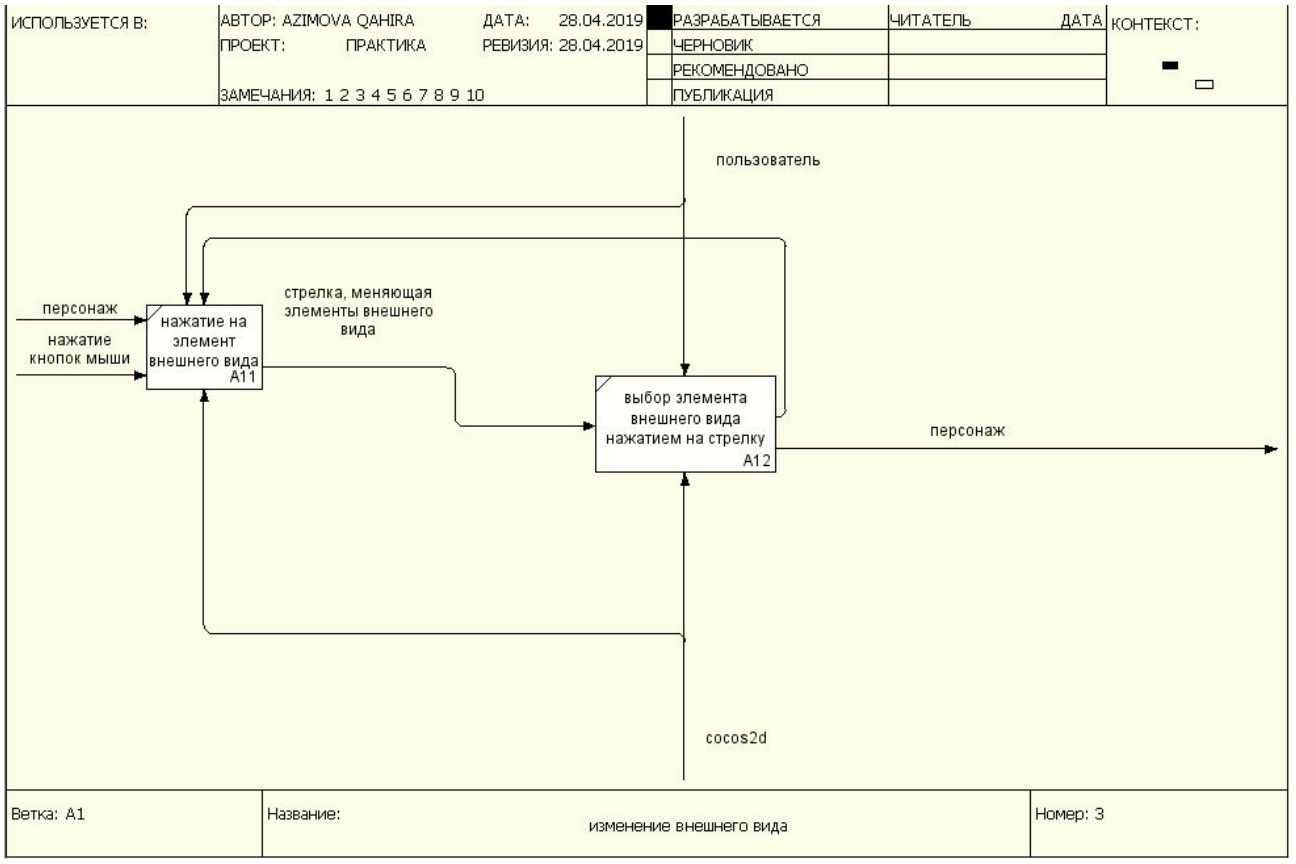
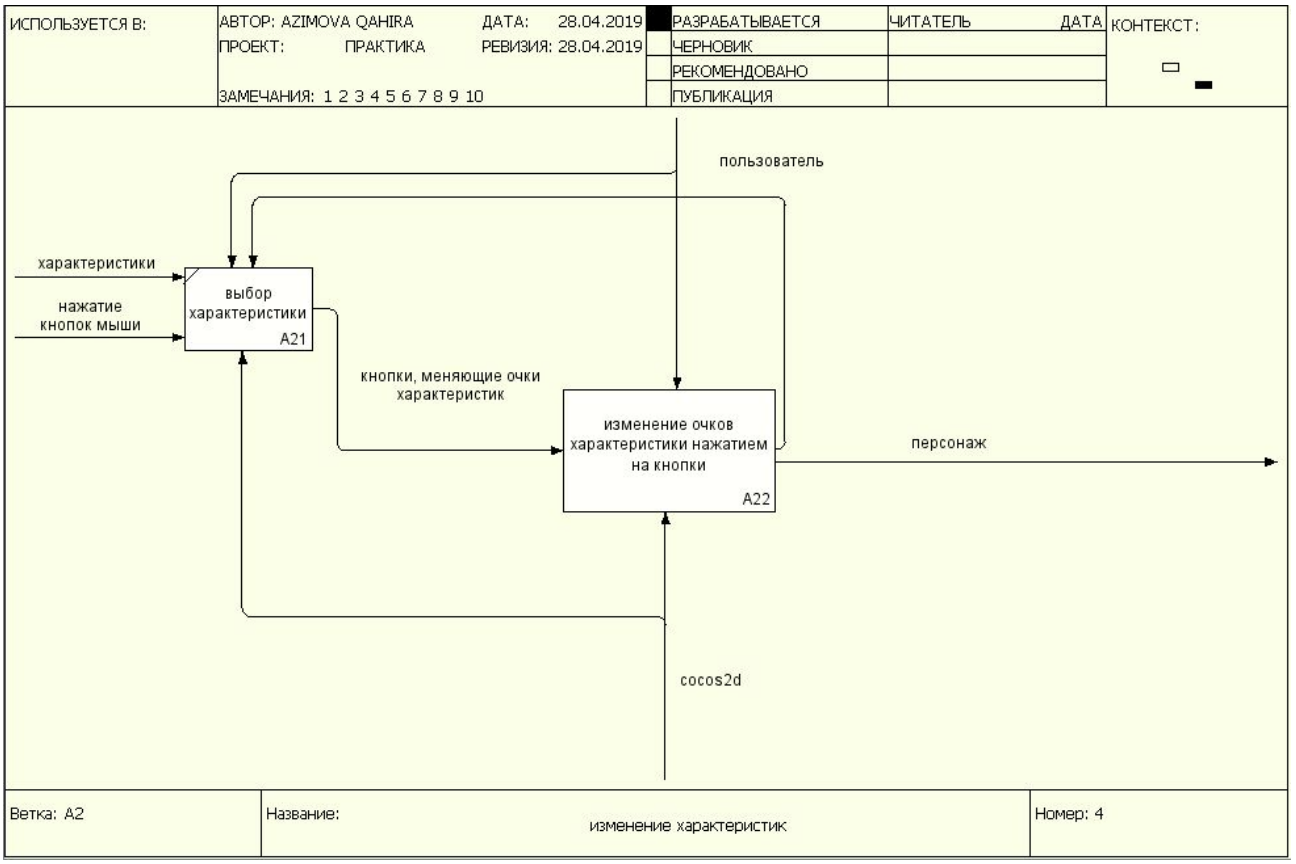


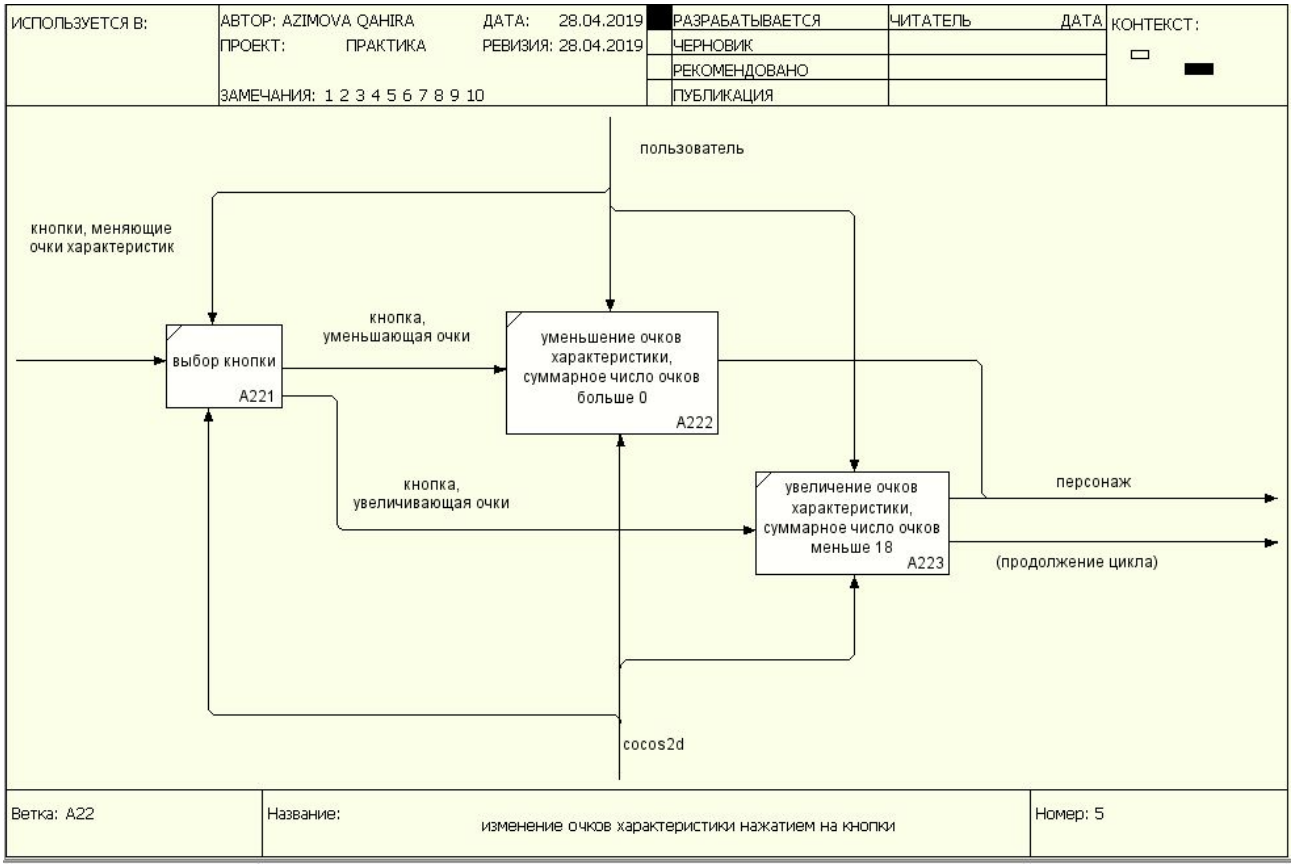
рис.11-13 - IDEF0 диаграмма главного меню

Меню создания персонажа






ИСПОЛЬЗУЕТСЯ В:
АВТОР: AZIMOVA QAHIRA
ДАТА: 28.04.2019
ПРОЕКТ: ПРАКТИКА
РЕВИЗИЯ: 28.04.2019
ЗАМЕЧАНИЯ: 1 2 3 4 5 6 7 8 9 10
РАЗРАБАТЫВАЕТСЯ
ЧЕРНОВИК
РЕКОМЕНДОВАНО
ПУБЛИКАЦИЯ
ЧИТАТЕЛЬ
ДАТА
КОНТЕКСТ:
пользователь
характеристики
нажатие кнопок мыши
выбор характеристики
A21
кнопки, меняющие очки характеристик
изменение очков характеристики нажатием на кнопки
A22
персонаж
cocos2d
Ветка: A2
Название:
изменение характеристик
Номер: 4


ИСПОЛЬЗУЕТСЯ В:
АВТОР: AZIMOVA QAHIRA
ДАТА: 28.04.2019
ПРОЕКТ: ПРАКТИКА
РЕВИЗИЯ: 28.04.2019
ЗАМЕЧАНИЯ: 1 2 3 4 5 6 7 8 9 10
РАЗРАБАТЫВАЕТСЯ
ЧЕРНОВИК
РЕКОМЕНДОВАНО
ПУБЛИКАЦИЯ
ЧИТАТЕЛЬ
ДАТА
КОНТЕКСТ:
пользователь
кнопки, меняющие очки характеристик
выбор кнопки
A221
кнопка, уменьшающая очки
уменьшение очков характеристики, суммарное число очков больше 0
A222
кнопка, увеличивающая очки
увеличение очков характеристики, суммарное число очков меньше 18
A223
персонаж
(продолжение цикла)
cocos2d
Ветка: A22
Название:
изменение очков характеристики нажатием на кнопки
Номер: 5

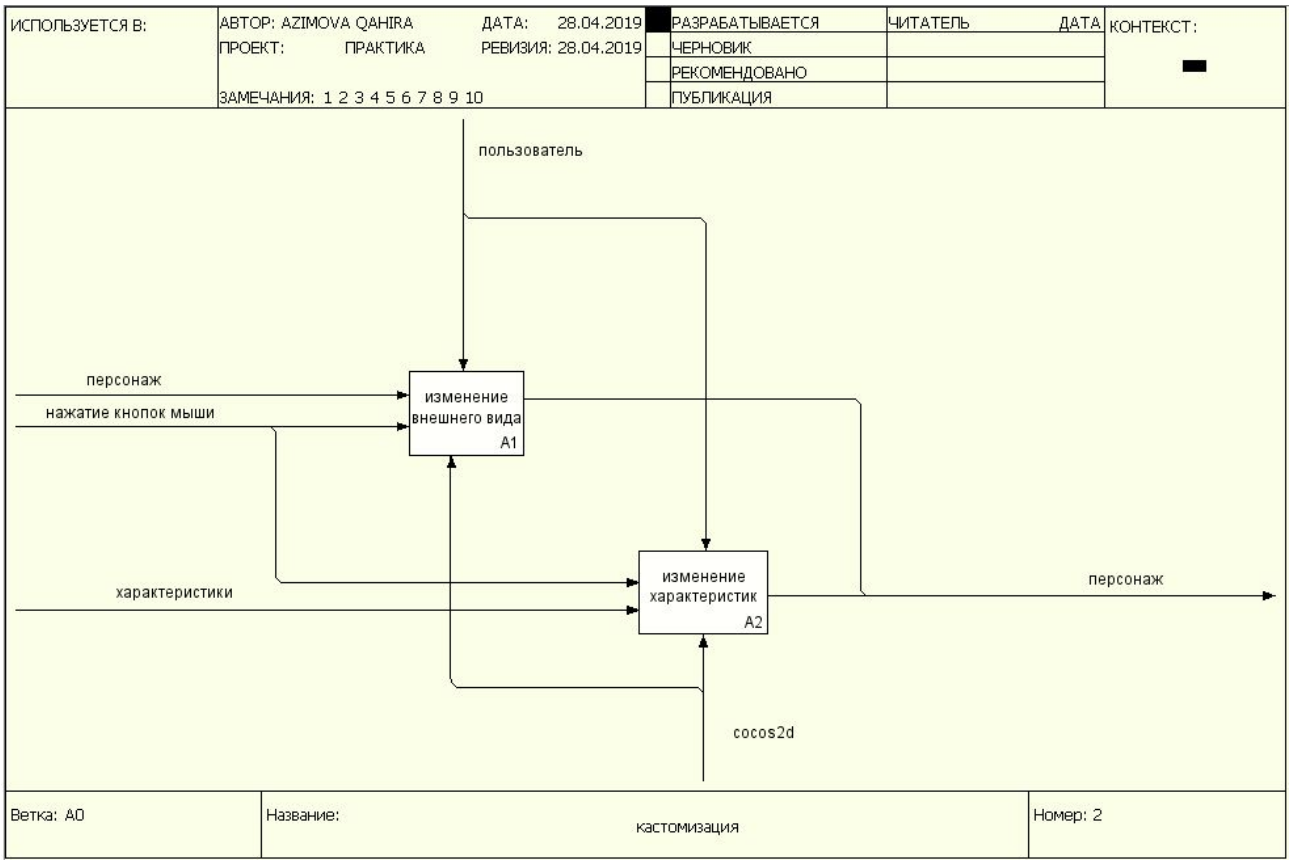


рис.14-18 - IDEF0 диаграмма меню создания персонажа

Задачи, поставленные перед каждым участником

Коннов Константин: разработка классов персонажа, главного героя и интерфейса, разработка игрового ИИ

Аникин Иван: разработка игровой физики, разработка скина, написание саундтрека для игры, разработка игрового ИИ

Азимова Гахира: разработка меню создания персонажа, рисование моделей для игры

Елизарова Мария: разработка системы заданий, разработка систем перехода между картами, подгрузки карт и выставления объектов

Куликов Георгий: организация рабочего процесса в команде, разработка системы игровых предметов, разработка игрового ИИ

Технологический раздел

При реализации игры были использованы:

1. Язык программирования Python 3.7
2. Библиотека Cocos2D

Python 3.7

Python3 - высокоуровневый язык программирования с минималистичным синтаксисом, язык предоставляет возможность быстро писать код. Благодаря этому получилось в минимальные сроки начать реализацию проекта, используя уже знакомый язык программирования по курсу предыдущего семестра.

Cocos2D

В качестве игрового движка было решено использовать Cocos2d, основным качеством которого является его широкий функционал.

В Cocos2d присутствуют методы для описания столкновений, batch-отрисовки спрайтов (для оптимизации программы), работы с двухмерной анимацией, эффектов и переходов.

Также Cocos2d является кроссплатформенным движком, что позволяет собирать ПО для разных операционных систем. Не маловажным достоинством Cocos2d является и стабильность его API.

Cocos2d является открытым ПО с лицензированием MIT License, что означает, что данное программное обеспечение имеет открытый исходный код и распространяется с разрешительной лицензией, то есть позволяет программистам использовать лицензируемый код в закрытом ПО при условии, что текст лицензии предоставляется вместе с этим ПО.

Методология разработки

Методология разработки RAD была выбрана по следующим причинам:

1. Необходимость в сжатые сроки осуществить проект
2. На первых этапах проектирования, наличие идеи, но отсутствие четких требований к готовому продукту

Также данная методология была выбрана из-за требования использовать в игровых проектах язык программирования Python и библиотеку cocos2D. Выбор данных технологий соответствует философии использования методологии RAD, так как одним из принципов является использование технологий, нацеленных на уменьшение времени разработки, а Python и cocos2D как раз такие технологии.

Работа была основана на постоянном создании прототипов, на основе которых итерационно строились следующие, стремясь удовлетворить условиям заказчика.

Тестирование

В ходе проведения практики производилось тестирование программного продукта. Было проведено ручное тестирование, методы и функции с особо сложными алгоритмами были покрыты тестированием черного и белого ящиков.

Ручное тестирование проводилось для выявления самых распространенных ошибок в играх: застревание в стене, прохождение сквозь объекты и пр. В результате тестирования было выявлено несколько ошибок и впоследствии данные ошибки были устранены.

Методы добавления предметов в инвентарь, выбора случайной точки NPC, стрельбы и классы отвечающие за игровую физику были покрыты критическими тестами. В ходе тестирования были выявлены ошибки связанные, с неправильным расположением точки вылета пуль и выходом случайной точки для патрулирования за допустимые границы.

Личный вклад

В ходе всей практики передо мной стояло несколько задач, все из которых были связаны с библиотекой Cocos2D.

В первую очередь я занялся разработкой игровой физики. Я использовал встроенный модуль TmxObjectMapCollider библиотеки Cocos2D, который создан специально для работы с tiled-картами, а также позволяет обрабатывать коллизии с объектами, которые можно выделить на карте в Tiled Map Editor. Успешно реализовав данную модель физики, было решено, что игрок и другие подвижные объекты должны иметь круглую модель столкновения, что будет более реалистично. Так как TmxObjectMapCollider работает только с прямоугольными объектами, я реализовал физику с помощью модуля collision_model библиотеки Cocos2D. Данный модуль позволяет работать с круглыми и прямоугольными объектами и отслеживать коллизии между ними, что идеально подходит для нашего проекта. Я использовал встроенный в модуль объект менеджер коллизий (collision manager), располагающий информацией о положениях, размерах и состояниях всех объектов карты. С его помощью можно проверить пересекаются ли два объекта, на каком расстоянии друг от друга они находятся и другую информацию. Все неподвижные объекты, такие как стены, ящики и прочие препятствия имеют прямоугольную форму, а главный герой, НПС и пули круглую.

Второй задачей, которую я реализовал, является создание класса skin для представления графического изображения персонажа. В ходе решения данной задачи особых трудностей у меня не возникло. В этом классе используется иерархическое дерево объектов библиотеки Cocos2D. У каждого объекта этой библиотеки могут быть объекты, которые являются «детьми» данного объекта. В классе skin создается спрайт размером 1x1 пиксель, который задает положение

игрока и является точкой отсчета для всех остальных частей игрока. Я реализовал функции добавления оружия в руку, скрытие оружия, приседание, надевание брони на тело. Также удобно то, что можно задать z-координату «ребенка» относительно «родителя». Благодаря этому происходит корректное наложение спрайтов рук, ног и головы друг на друга.

Третья немаловажная задача, которую я выполнил, это подготовка звуков и музыки во время игрового процесса. Я реализовал данную задачу с помощью модуля audio.mixer библиотеки Cocos2D, который является надстройкой над звуковым модулем pygame. Для воспроизведения звуков необходим объект mixer, который может работать с звуковыми каналами (channel) или напрямую звуками (sound). В нашем проекте используется два канала. Один канал используется для воспроизведения звуков ходьбы, стрельбы и прочих, а второй используется для воспроизведения фоновой музыки. Использование двух каналов позволяет изменять громкость музыки и звуков независимо друг от друга, что очень удобно в нашей ситуации.

Заключение

В результате практики была создана практически вся техническая база для программного продукта. Для дальнейшей разработки было решено оставить декоративные задачи(сюжет, локации, квесты, диалоги и пр.) и систему боя у не игровых персонажей.

В ходе практики члены команды научились работать с UML диаграммами, освоили основы объектно-ориентированного программирования на языке Python 3.7, была изучена библиотека Cocos2D, были изучены различные паттерны проектирования и методологии разработки, были получены важные навыки командной разработки, навыки ведения проекта при помощи удаленного репозитория.

Литература

1. Документация библиотеки pyglet // pyglet.org URL: https://pyglet.readthedocs.io/en/pyglet-1.3-maintenance/ (дата обращения: 30.03.2019).
2. Документация библиотеки cocos2D // python.cocos2d.org URL: http://python.cocos2d.org/doc.html (дата обращения: 28.03.2019).
3. Разработка игр с использованием Cocos2d на Python // habr URL: https://habr.com/ru/post/120438/ (дата обращения: 28.03.2019).