SUPER MARIO на PYTHON

Выполнила:
Шаронова Вероника

Введение

Проект Super Mario был создан из-за некоторых аспектов:

- Относительная простота выполнения
- Личный интерес к данной игре
- Практика в создании собственной игры

Описание реализации

Структура классов в приложении следующая:

- Класс Animation отвечает за анимацию объектов. В конструкторе класса указываются изображения, используемые для анимации, а также другие параметры, такие как текущий индекс изображения и задержка между сменой изображений. Методы класса позволяют обновлять анимацию и изменять текущее изображение.

- Класс Camera используется для управления камерой в приложении. В конструкторе класса задается начальная позиция камеры и объект, за которым следует камера. Метод move() перемещает камеру в зависимости от положения объекта.

- Класс Collider предназначен для обработки столкновений объектов в приложении. В конструкторе класса указываются объект и уровень, на котором происходят столкновения. Методы класса позволяют проверять столкновения по осям X и Y.

- Класс Dashboard отвечает за отображение данных на экране, таких как счет, количество монет и время. Конструктор класса принимает пусть к файлу со шрифтом, его размер и экран. Метод update() обновляет отображаемые данные.

Описание реализации

Структура классов в приложении следующая:

- Класс CreatureCollider используется для обработки столкновений между объектами. В конструкторе класса указывается объект, для которого выполняется проверка столкновений. Метод check() проверяет столкновение с целевым объектом.

- Класс CollisionState представляет состояние столкновения и содержит информацию о том, произошло ли столкновение и было ли оно сверху.

- Класс "Font" наследуется от класса "Spritesheet" и отвечает за загрузку шрифтов. Он инициализирует массив символов, определяет функцию "loadFont", которая загружает изображения символов шрифта, используя метод "image_at" из класса "Spritesheet". Загруженные шрифты хранятся в словаре "font".

- Класс "Blur" отвечает за применение гауссового размытия к поверхности. Он инициализирует размер ядра размытия и имеет метод "filter", который принимает поверхность, позицию, ширину и высоту и возвращает новую размытую поверхность.

Описание реализации

Структура классов в приложении следующая:

- Класс "Input" отвечает за обработку ввода с клавиатуры и мыши. Он инициализирует координаты мыши и объект сущности, проверяет ввод с клавиатуры (нажатие клавиш WASD, пробела и клавиш SHIFT), обрабатывает ввод с мыши (нажатие левой и правой кнопки), а также обрабатывает события завершения приложения и перезапуска.

- Класс "Level" отвечает за загрузку уровня и его отрисовку. Он инициализирует объекты спрайтов, звук, экран и уровень, а также список сущностей. Он имеет методы для загрузки уровня из файла JSON, загрузки сущностей, слоев и объектов из данных уровня, а также обновления сущностей и отрисовки уровня.

- Класс Vec2D представляет двумерный вектор с координатами x и y. Использованы следующие библиотеки: json, sys, os, pygame.

Описание реализации

Структура классов в приложении следующая:

- Класс Menu отвечает за отображение и взаимодействие с главным меню игры.
  - В конструкторе инициализируются переменные и загружаются настройки из файла "settings.json".
  - Метод update обновляет состояние меню и отрисовывает его на экране, в зависимости от текущего режима.
  - Методы drawDot, drawMenu, drawMenuBackground, drawSettings отвечают за отрисовку элементов меню.
  - Методы loadSettings и saveSettings загружают и сохраняют настройки в файле "settings.json".
  - Методы chooseLevel, drawLevelChooser, loadLevelNames используются для выбора уровня.
  - Метод checkInput обрабатывает пользовательский ввод.
- Класс Pause отображает паузу в игре.
  - Метод update обновляет состояние паузы и отрисовывает ее на экране.
  - Методы drawDot и createBackgroundBlur отвечают за отрисовку элементов паузы и создание размытого фона.

Описание реализации

Структура классов в приложении следующая:

- Класс Sound отвечает за управление звуковыми эффектами и музыкой в приложении. Он инициализирует звуковые каналы и загружает звуковые файлы.

- Класс Sprite представляет спрайт и его свойства, такие как изображение, коллизии, анимация и необходимость перерисовки фона.

- Класс Sprites отвечает за загрузку спрайтов из файлов JSON. Он использует классы Spritesheet и Sprite для создания объектов спрайтов из изображений.

- Класс Spriteheet представляет лист спрайтов, загружает изображение и предоставляет методы для извлечения отдельных спрайтов из листа.

- Класс Tile представляет тайл и его свойства, такие как спрайт и прямоугольник.

Описание реализации

Структура классов в приложении следующая:

- Класс Coin описывает объект монеты (Coin) в игре. У него есть атрибуты screen, spriteCollection, animation, type. Метод update обновляет анимацию объекта и отображает его на экране.

- Класс CoinBox описывает коробку с монетой (CoinBox) в игре. У него есть атрибуты screen, spriteCollection, animation, type и др. Метод update обновляет анимацию коробки, а также отображает на экране пустую коробку и спавнит монеты из коробки.

- Класс CoinBrick описывает кирпичную коробку с монетой (CoinBrick) в игре. У него есть атрибуты screen, spriteCollection, image, type и др. Метод update обновляет изображение кирпичной коробки, а также отображает на экране пустую коробку и спавнит монеты из коробки.

- Класс CreatureBase является базовым классом для всех игровых объектов и содержит общую логику для всех объектов. Он содержит атрибуты vel, rect, gravity и другие, а также методы для применения гравитации, обновления особых черт объекта и т.д.

Описание реализации

Структура классов в приложении следующая:

- Класс Goomba описывает объект гумбы (Goomba) в игре. У него есть атрибуты screen, spriteColl, animation, type и др. Метод update обновляет позицию и анимацию гумбы, проверяет столкновения с другими объектами и выполняет соответствующие действия при столкновениях.

- Классе "Item" реализует создание и отображение монетки, которая двигается вверх и вниз и исчезает после определенного времени.

- Класс "Koopa" реализует движение и анимация творения "Koopa", а также обработка его столкновения с другими объектами.

- Класс "Mario" реализует управление персонажем "Mario", его движение, столкновение с другими объектами и взаимодействие с ними.

Описание реализации

Структура классов в приложении следующая:

- Класс "RedMushroom" реализует движение и отображение объекта "RedMushroom", а также обработка его столкновений.

- Класс "RandomBox" отображает объект "RandomBox", его анимацию и обработку запуска рандомного предмета при взаимодействии.

- Класс bounceTrait реализует функциональность отскока объекта и обновление его состояния.

Описание реализации

Структура классов в приложении следующая:

- Класс GoTrait реализует движение объекта влево и вправо с анимацией. Включает различные параметры для скорости и ускорения.

- Класс JumpTrait реализует функциональность прыжка объекта с учетом гравитации и высоты прыжка.

- Класс LeftRightWalkTrait реализует движение объекта влево и вправо по заданному уровню. Использует класс Collider для обнаружения столкновений.

Технологии

В приложении используются следующие технологии:

- pygame (версия 2.0.0.dev10) (библиотека для разработки игр на языке Python);
- scipy (версия 1.4.1) (библиотека для научных вычислений);
- json (библиотека для работы с данными в формате JSON);
- random для случайного выбора направления движения;
- collider для обнаружения столкновений;

Выводы

В начале работы над проектом были поставлены задачи:

- Закрепиить полученные знания;
- Улучшить практических навыков в работе с pygame;
- Создать собственное приложение;

В конце написания проекта я выполнила все поставленные задачи, а значит цель достигнута

Перспективы

Данной работе необходимы доработки:

- Добавить уровни (на данный момент их 2)
- Изменить слежение камеры за объетком (создать зону, в которой персонажможет двигаться без камеры)
- В целом подчистить код, проверить на наличие логических или других ошибок

Пример текущего состояния