НЕКЛАССИЧЕСКАЯ ИГРА ЗМЕЙКА НА OPENGL

Панкевич Д. С., Кривоносова Т. М.
Кафедра вычислительных методов и программирования, Белорусский государственный университет информатики и радиоэлектороники
Минск, Республика Беларусь
E-mail: pankevich.daniil@yandex.ru

Рассматривается пример развития классического концепта игры «Змейка» в оригинальную идею, раскрывая потенциал заложенных в игру основных механик.

I. Классическая игра «Змейка»

Игрок управляет существом, напоминающим змею, которое ползает по плоскости, собирая еду, избегая столкновения с собственным хвостом. Каждый раз, когда змея съедает кусок пищи, она становится длиннее, что постепенно усложняет игру. Игрок управляет направлением движения головы змеи, а хвост змеи движется следом. В некоторых версиях игры на поле присутствуют дополнительные препятствия, разные бонусы. Есть версии игры с несколькими уровнями или с одним уровнем, где необходимо набрать как можно больше очков.

II. Разработка своей версии игры

Разрабатываемая версия игры будет иметь следующие особенности:

- На игровом поле может находится несколько единиц еды одновременно;
- От длины Змейки зависит её скорость;
- В игре присутствует несколько уровней;
- У змейки есть несколько жизней на текущий уровень;
- При столкновении с телом у игрока отнимается одна жизнь и та часть тела, которая была «отрезана» от тела, становится стеной;
- При столкновении со стеной или когда у Змейки не останется жизней уровень начнётся заново.

III. Выбор платформы

Выбор платформы я остановил на программном интерфейсе OpenGL и языке программирования C++. При выборе я отталкивался от того, что этого будет достаточно для воплощения в жизнь своего концепта.

IV. Основная структура

Игровое поле - двумерный массив объектов перечисления(enum). На игровом поле могут располагаться следующие объекты(в скобках указан цвет): стена(серый), еда(желтый, пустота(черный), тело змейки(белый) и голова змейки(зависит от количества жизней). Каждый из этих объектов - именованная константа из перечисления. Каждый кадр по определённому алгоритму массив с объектами отображается на экране в виде сетки с расположенными на ней квадратами разного размера и цвета, в зависимости от свойства объектов.

V. Структура Змейки

Змейка - двусвязная очередь объектов, хранящих в себе две целочисленных координаты. Эта очередь каждый игровой тик пересчитывается и вносит изменения в игровое поле. При пересчёте голова перемещается на одну клетку вперёд в направлении текущего движения, а все остальные элементы становятся на место следующих. Координаты последнего элемента змейки на игровом поле превращаются в объект пустоты.

VI. Управление

У змейки есть две переменные, хранящие в себе одно из четырёх направлений. Одна переменна хранит текущее направление – то направление, куда змейка движется, а другая переменная хранит пользовательское направление – то направление, которое хочет задать игрок. Если пользовательское направление не является противоположным текущему – то оно становится подтверждённым и змейка меняет направление своего текущего движения. В ином случае змейка не реагирует на изменение пользовательского направления и продолжает движение в сторону текущего.

Захват управления осуществляется путём считывания состояния старшего бита адреса клавиш клавиатуры.

Стрелки на клавиатуре – задать пользовательское направление движения;

Spacebar – поставить игру на паузу.

Escape – выход из игры.

Цифры от 1 до 5 – позволяют переключаться между пройденными уровнями.

VII. Цикл программы

Каждый игровой тик выполняется набор определённых действий:

1. Обработка нажатий клавиш. На этом этапе происходит обработка состояния каждой из задействованных в игре клавиш, и в зависимости от их текущего и прошлого состояния им присваивается одно из трёх новых:

нажатие, удержание, или отжатие клавиши.
2. Обновление игровых состояний. На этом этапе происходит пересчёт игрового поля по соответсвующим правилам расположенных на нём объектов.
3. Отрисовка экрана. На этом этапе происходит отрисовка игрового поля в виде сетки с разноцветными, разными по размеру клетками.

VIII. Описание реализации механик

У змейки всего 3 жизни на уровень, которые можно восстановить, собирая еду. За каждые 17 единиц еды восстанавливается одна жизнь.

При врезании в сегмент тела у змейки отнимается одна жизнь и начиниается поиск по очереди элемента с координатами столкновения. Когда нужный элемент найден - происходит удаление всех элементов от хвоста до того сегмента, с которым столкнулась голова, при этом координаты удаленных сегментов на поле заменяются на объекты стен. Также отбирается столько еды, сколько было удалено сегментов тела. То бишь, у змейки отпадает часть от хвоста до того сегмента, с которым столкнулась голова, а отпавшая часть превращается в стену и остаётся лежать на игровом поле, как препятствие.

Если у змейки не останется жизней, то уровень начинается заново. Индикатор количества жизней - цвет головы змейки. Всего три цвета: синий(3 жизни), фиолетовый(2 жизни) и розовый(1 жизнь).

При столкновении с краем игрового поля голову змейки переносит на противоположный по текущей координате столкновения конец этого поля.

При врезании в стену змейка умирает и, с небольшой задержкой на кат-сцену, уровень начнётся заново. Каждый уровень представляет собой функцию, которая заполняет массив игрового поля объектами перечисления в определённой последовательности. Для хранения текущего уровня используется указатель на функцию заполнения.

Скорость змейки линейно зависит от количества собранной еды. При увеличении скорости змейки вокруг её тела образуется красная обводка, и чем больше скорость - тем она лучше видна. Чтобы перейти на следующий уровень игроку необходимо набрать определённое количество еды, для каждого уровня оно своё.

При врезании змейки в объект еды он удаляется с игрового поля, счётчик количества собранной еды увеличивается на единицу и у змейки добавляется один элемент в конец очереди. Координатам этого элемента присваиваются координаты предыдущего элемента очереди. После того, как объект еды был удалён, составляется карта пустых клеток игрового поля, затем случайным образом из неё выбирается одна клетка, и по координатам этой клетки на карте на соответсвующих координатах игрового поля создаётся объект еда.

IX. Описание дизайна уровней

В игре есть несколько заготовленных уровней, которые ставят перед игроком определённые испытания:

1. Игроку даётся пустое поле, обнесённое по периметру стенами. Тут игрок понимает, можно ли врезаться в стены, в себя, и что от набранной массы меняется скорость змейки;
2. Уровень построен таким образом, что игроку обязательно необходимо привыкнуть и применять новую механику с врезанием в стены, чтобы пройти уровень;
3. Смешанный тип уровня. Игрок может долгое время находится на одной половине уровня, пока на нём есть еда, а затем перебраться на другую половину уровня используя стены;
4. На уровне нет стен и есть очень большое количество еды. Длина змейки, необходимая для прохождения уровня, в несколько раз выше предыдущих уровней, и игрок разовьёт на змейке максимальную скорость, что вызовет у него очередное испытание, даже не имея статичных препятствий на уровне, т.к. само тело змейки и есть главное препятствие;
5. Уровень со случайной генерацией стен у краёв экрана и свободным пространством ближе к центру. Добавляет разнообразия привычным симметричным уровням.

Отталкиваясь от классического концепта можно прийти к чему-то оригинальному, и это довольно частая практика в мире игровой индустрии. Не смотря на простоту исходной идеи, она имеет безграничные возможности по расширению и добавлению новых механик, правил, фичей, игровых режимов и многого другого. Идя от меньшего к большему – мы создаем не только что-то особенное, но и в целом продвигаем вперёд игровую индустрию.

Список литературы

1. The OpenGL Graphics System: A Specification (Version 4.5 (Core Profile) - August 11, 2014).
2. Шилдт, Герберт C++: базовый курс / Герберт Шилдт – М.: Вильямс, 2008. - 624 с.
3. Лафоре Р. Приложение Е // Лафоре Р. Объектно-ориентировочное программирование на с++. —- Санкт-Петербург: Питер, 2004. —- С. 836—843
4. Скотт Мейерс. Эффективное использование STL = Effective STL. —- Питер, 2002. —- С. 224. —- (Библиотека программиста). -— ISBN 5-94723-382-7.
5. Дэвид Р. Мюссер, Жилмер Дж. Дердж, Атул Сейни. C++ и STL: справочное руководство = STL Tutorial and Reference Guide: C++ Programming with the Standard Template. -— 2-е издание. —- М.: «Вильямс», 2010. —- С. 432. —- (серия C++ in Depth). – ISBN 5-89818-027-3.