Сборка

Никаких внешних библиотек и систем сборок не использовал, так что сборка:

1. Открыть проект в студии.
2. Build -> Build Solution

Писал и собирал в Visual Studio 2010.

Кружочки

Запуская Кружочки игрок попадает в стартовое меню, где может либо сразу же завершить программу, нажав правую кнопку мыши, либо начать игру, нажав левую.

В игре можно отстреливать кружочки, кликая на них левой кнопкой мыши, либо открыть главное меню, нажав правую кнопку (игра встанет на паузу).

В главном меню можно либо вернуться в игру (левая кнопка), либо выйти в ОС (правая кнопка).

Структура кода

Логически впроект можно разбить на три части:

1. Framework Core.

Это интерфейсы общих компонент, необходимых для построения игры, и структура Event - представление событий, которое мы используем вместо нативных событий (требование кросс-платформенности). Все они находятся в неймспейсе kruz. Я называю это `фреймворком` потому, что логически это он.

Заголовочные файлы:

1. Config.h

Макроопределения для дебажного вывода.

2. Event.h

Вышеупомянутые события.

3. IEventManager.h

Интерфейс менеджера событий, который должен получать платформо-зависимые события, преобразовывать в наши, а их рассылать зарегистрированным обработчикам событий.

4. IEventHandler.h

Интерфейс обработчика событий, который будучи зарегистрированным в менеджере событий, будет оные события получать, и, вероятно, как-то обрабатывать.

5. IGameState.h

Интерфейс игрового состояния - "Интро", "Стартовое меню", "Собственно, игра", "Основное меню" и т.д.

6. IGameStateManager.h

Интерфейс менеджера игровых состояний. Должен осуществлять переключение между зарегистрированными игровыми состояниями.

7. IGfxManager.h

Интерфейс графического менеджера, который должен уметь рисовать наши кружочки, счетчик и надписи для меню.

8. IRoot.h

Интерфейс игрового ядра, которое которое должно управлять игровым циклом и всеми нашими менеджерами.

9. Kruzhochki.h

Включает все вышеописанные файлы, т.е. для того, чтобы использовать фреймворк, достаточно сделать #include "Kruzhochki.h".

2. Framework Implementations

Это как зависимые, так и независимые от платформы реализации интерфейсов из Framework Core.
Они также находятся в неймспейсе kruz.

Заголовочные файлы:

1. CWinapiRoot.h

Winapi-реализация ядра. Также инкапсулирует в себе Windows-окно и платформо-зависимую работу с OpenGL.

2. CWinapiEventManager.h

Winapi-реализация менеджера событий. Получает от ядра (только CWinapiRoot) Windows-события и на их основе формирует и рассылает наши игровые события.

3. CGameStateManager.h

Платформо-независимая реализация менеджера состояний.

4. COpenglGfxManager.h

Платформо-независимая реализация менеджера графики на OpenGL.

3. Game Logics

Классы, собственно, игры, построенные на классах `фреймворка`.

Заголовочные файлы:

1. Kruzhochek.h

Класс кружочка, который знает свои координаты центра, радиус, скорость и количество очков, которые игрок получит за его убийство. Скорость и ценность расчитываются исходя из величины

радиуса. Кружочек также умеет падать вниз со своей постоянной скоростью скоростью и рисовать себя с помощью IGfxManager.

2. KruzhochkiState.h

Основное игровое состояние игры "Кружочки": кружочки падают, юзер их чпокает, очки насчитываются.
Инкапсулерует, помимо прочего, нашу сцену (Я решил не выделять самостоятельный менеджер сцены, в виду её простоты - std::list<Kruzhochek*>).

3. IntroState.h

Стартового состояние: название игры и меню (начать игру/выйти).

4. MainMenuState.h

Состояние главного меню (вернуться в игру/выйти). Переход в это состояние происходит из основного игрового состояния, при этом, игра не прерывается, а встает на паузу и продолжается, если пользователь вернется в игру.