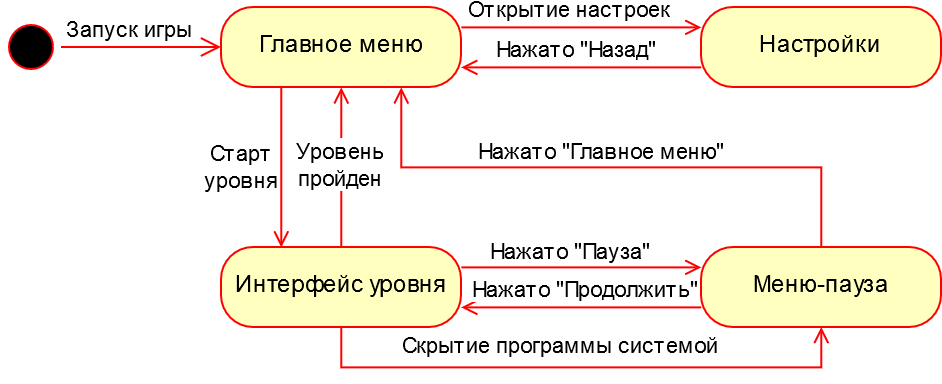
Данный документ создан с целью детализации полученного ТЗ и прототипирования во избежание лишних трат ресурсов разработки и создания четкого видения результата.

Для удобной и расширяемой среды с реализацией всего функционала сперва необходимо очертить требования и саму структуру игры.

Согласно данному ТЗ, диаграмма переходов между окнами строится следующим образом:



*Диаграмма состояний UI игры*

В игровых **настройках** требуется только включение/выключение фоновой музыки на всех окнах (глобальная переменная).

Также следуя ТЗ, выделяются 2 игровых комнаты:

* комната меню (*главное меню* и *настройки*);
* комната уровня (сам игровой процесс, а также *интерфейс уровня*, *меню-пауза*).

**Комната меню** состоит исключительно из UI, которое будет создано по предоставленной вёрстке, и методов его обработки. Большая детализация не требуется.

**Комната уровня** имеет минимальный UI и набор методов для реализации игровой логики. Игра строиться на реалистичных столкновениях фигурок, с чем прекрасно справиться **встроенная физика Unity** (*Rigidbody*, *Collider*). Все фигурки могут иметь определенный цвет, следовательно нужен **метод покраски** (*MaterialPainter*). Сам уровень создается при помощи **рандомного генератора** (*IWallGenerator*, *RegularWallGenerator*) и имеет большой набор элементов. Так как каждый элемент имеет физику и коллайдеры Unity (что довольно ресурсоемко), необходимо создать **менеджер карты** (*MapManager*), который бы включал/выключал физику по мере прохождения, а также **сборщик мусора** (метод *MapManager*), удаляющий уже не активные объекты. Также не помешает **менеджер игры** (*GameManager*), который будет запускать уровень и следить за ходом игрового процесса. Условия победы для гибкости и дальнейших изменений лучше выделить в отдельного **судью** (*GameMode*), который контролирует действия и столкновения персонажа. Сам же персонаж управляем игроком при помощи **контроллера** (*PlayerController*). Все это время камера должна следить за персонажем под определённым углом при помощи **алгоритма движения** (*CamMotion*).

Складывая все вышеперечисленное воедино, вырисовывается диаграмма всех элементов логики и их взаимодействие:

