Design document

Общие сведения о системе

Описание

Игра жанра `Rogue-like`. Это консольная однопользовательская игра с сохранением. Основная цель — добраться до последнего уровня игры.

Главный персонаж — герой с исходными параметрами (здоровье, скорость, сила). В процессе игры пользователь управляет героем посредством клавиатуры. За одно нажатие совершается одно действие или перемещение на одну клетку карты.

В каждой клетке может находится препятствие, враг или предмет. При попытке переместиться в клетку с врагом происходит поочередная атака.

При перемещении в клетку с предметом предмет добавляется в инвентарь игрока. Игрок может использовать предметы из инвентаря и с их помощью улучшать параметры героя.

Каждый уровень генерируется случайным образом и состоит из карты с комнатами, соединенных проходами. Переход от одного уровня к другому осуществляется по специальному проходу.

Если у героя заканчивается здоровье, он умирает без возможности сохранения. Игра заканчивается и начать её можно только сначала.

Функциональные требования

- Генерация уровней;
- Наличие меню;
- Возможность сохранения прогресса;
- Возможность загрузки уровня;
- Различные виды врагов и предметов;
- Возможность улучшения характеристик героя;
- Наличие инструкции к игре.

Ограничения

- Пошаговость. За одну команду осуществляется один ход.
- Наличие временного приоритета у героя и различных видов врагов при атаке.
- Процедурная генерация. Процедурно генерируются уровни, враги и предметы.
- Размер карты уровня не ограничивается пространством.
- Графика реализована с помощью текстовых символов на экране.
- Интерфейс и управление.
- Наличие набора свойств у героя, предметов, врагов и местности, которые влияют на владельца и его окружение.
- Необратимая смерть героя. Игрок должен начинать игру с начала и не может загрузить последнее сохранение.

Architectural drivers

Расширяемость	Герою, врагам и предметам присваиваются различные типы параметров. Гибкая и расширяемая система настроек позволит не фиксировать изначально каждый тип отдельно и позволит добавлять новые параметры.
Переносимость	Возможность запуска с различных устройств без загрузки дополнительного программного обеспечения.
Производительность	Быстрое отображение команды пользователя. Среднее время отображение изменений на экран — 0,1 секунда.
Удобство использования	Интуитивная доступность и простота в управлении игрой, наличие инструкции.
Готовность	Осуществления процесса игры с минимальным набором исходных параметров.

Роли и случаи использования

Основные действующие лица и их главные цели:

- Пользователь. Осуществляет процесс игры с помощью команд.
- Разработчики. Реализуют и сопровождают игру.

Описание типичного пользователя:

- Пользователь является человеком возрастом от 10 и более лет, вне зависимости от пола, национальности и сферы деятельности.
- Для взаимодействия с игрой опыт не является обязательным.
- Основная идея использования игры пользователем — обеспечить досуг, без определенного ограничения по времени процесса игры.
- Необходимо, чтобы основные ожидания пользователя от жанра оправдались. Обеспечить играбельность и доступность процесса.

Композиция

(`Game Components Diagram` в конце документа)

Внутри система устроена по принципу `Model-View-Controller`.

Команды пользователя обрабатывает `KeyboardHandler`, являющийся частью блока `Controller`. Результат `KeyboardHandler` превращается в высокоуровневую команду при помощи фабрики `CommandFactory` и передаётся в `Model`.

Вся логика игры реализована в блоке `Model`, где основным классом является `GameState`.

Результат выполнения команды на экран пользователю отображает класс `UI`, являющийся частью блока `View`.

Логическая структура

(`Game Class Diagram` в конце документа)

Система состоит из трёх крупных классов: `UI`, `Controller` и `Model`.

Главная функция `UI` — `print_map` — печатает карту на экран пользователю.

Главная функция `Controller` — `run` — обрабатывает команду пользователя, трансформируя её в высокоуровненый класс `Command`, и передаёт её в `Model`.

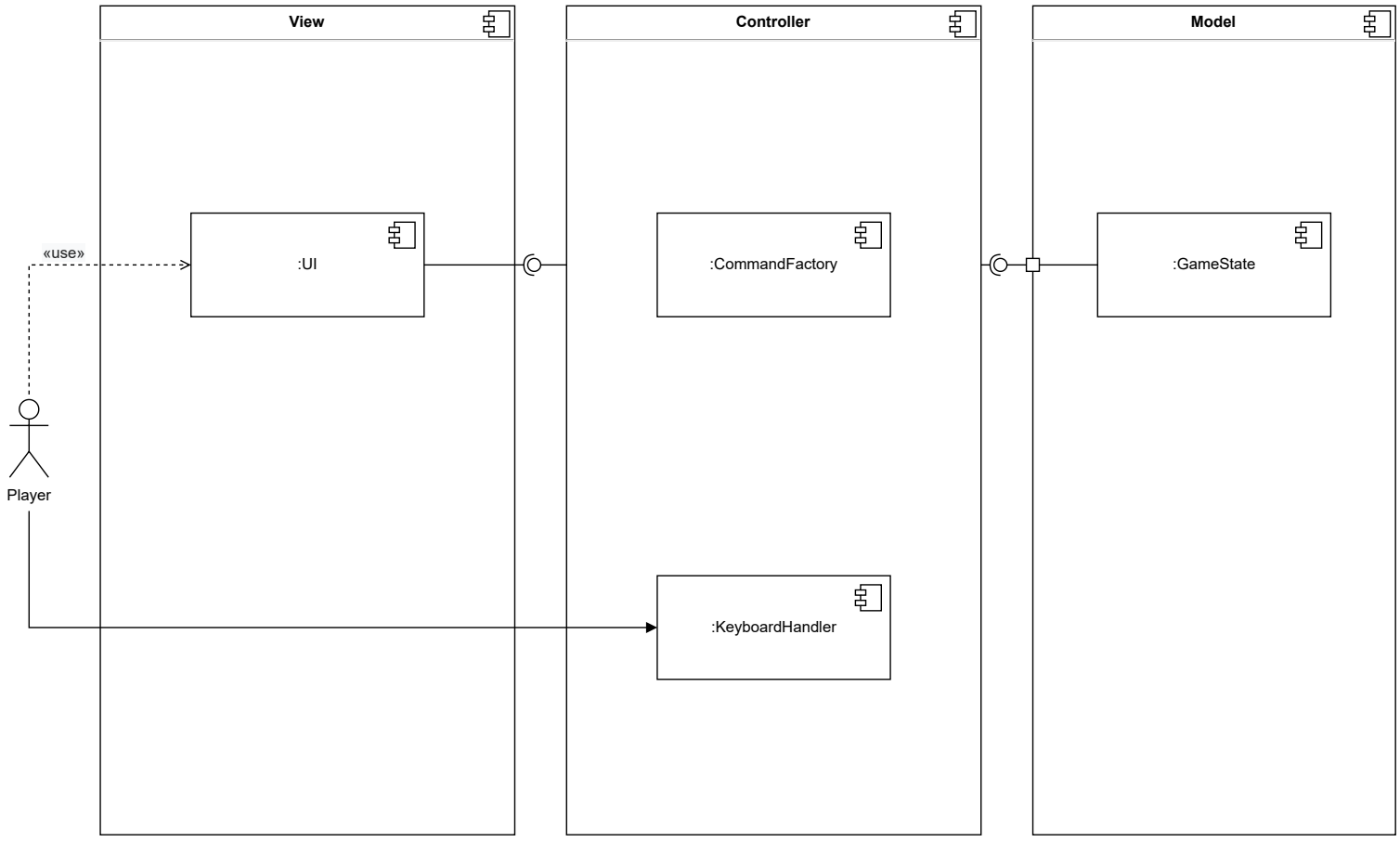
Главная функция `Model` — `process_command` — обрабатывает команду, изменяя состояние игры, и возвращает карту для отрисовки `UI`.

Взаимодействия

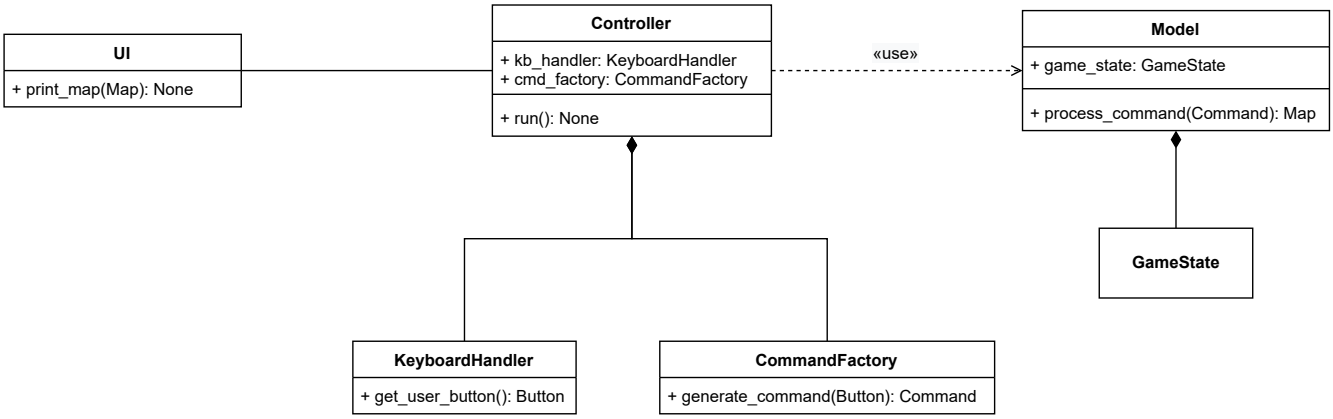
(`Game Sequence Diagram` в конце документа)

Команда пользователя обрабатывается `KeyboardHandler`, затем трансформируется `CommandFactory` в высокоуровневый класс. Результат передаётся в `Model`, который обрабатывает команду, изменяет состояние игры и возвращает карту `UI` для отрисовки пользователю.

Game Components Diagram



Game Class Diagram



Game Sequence Diagram