## АРХИТЕКТУРА

## СПЕЦИФИКАЦИИ ВСЕХ МОНСТРОВ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Локация | Название | Урон | HP | Эффект |
| Пустошь | **Зомби** | 2 | 5 | После своей атаки, если игрок его атаковал — получает +1 урона |
| Пустошь | **Низший скелет** | 3 | 4 | Нет эффекта |
| Снежные руины | **Драург** | 4 | 8 | 25% шанс нанести двойной урон (2 удара по 4) |
| Снежные руины | **Гуль** | 5 | 10 | 50% шанс парализовать игрока на 1 ход |
| Песчаные пустоши | **Ревенант** | 6 | 14 | Иммунитет к огню |
| Песчаные пустоши | **Мумия** | 7 | 16 | 50% шанс уменьшить урон игрока вдвое на 3 хода |
| Сожжённые джунгли | **Кровопийца** | 8 | 20 | 30% шанс отразить следующую атаку игрока |
| Сожжённые джунгли | **Дух** | 9 | 22 | Иммунитет к льду |
| Кратер | **Рыцарь смерти** | 10 | 50 | 25% шанс нанести тройной урон (30 суммарно) |

## СПЕЦИФИКАЦИИ ВСЕХ ЭФФЕКТОВ

|  |  |
| --- | --- |
| НАЗВАНИЕ | ОПИСАНИЕ |
| ГОРЕНИЕ | Наносит 2 урона в конце хода, 2 хода |
| *ОСЛАБЛЕНИЕ* | Урон ×0.5 на 2 хода |
| ЗАМОРОЗКА | Пропускает 1 ход |
| ПАРАЛИЧ | Пропускает 1 ход |
| ВАМПИРИЗМ | +3 урона игроку, -3 врагу, 2 хода |
| УКЛОНЕНИЕ | 50% шанс избежать урона |
| УСИЛЕНИЕ | +50% к следующей атаке |
| ИММУНИТЕТ | Не воспринимает определённый эффект |

## СПЕЦИФИКАЦИИ ВСЕХ ЗАКЛИНАНИЙ

## Начальные заклинания (даются в начале игры):

### 1. **IGNIS (Огонь)**

**Описание:** Поджигает врага, нанося урон каждый ход (эффект «Горение» на 2 хода).  
**Механика:**

* Наносит 5 урона сразу
* Затем 2 урона в конце следующего и ещё одного хода (итого 9)  
  **Применение:**  
  cast IGNIS — в консоли появляется:  
  “Ты поджигаешь врага. Он начинает гореть.”

### 2. **TERRO (Земля)**

**Описание:** Сковывает врага, снижая его урон на 50% на 2 хода.  
**Механика:**

* Эффект «Замедление» — враг наносит меньше урона  
  **Применение:**  
  cast TERRO —  
  “Ты насылаешь тяжесть земли. Враг замедлен.”

### 3. **AERO (Воздух)**

**Описание:** Шанс уклонения от следующей атаки (50%), усиление следующей атаки (50%)  
**Механика:**

* На 1 ход:
* Если враг атакует, есть шанс полностью избежать урона
* Гарантировано усиление следующей атаки  
  **Применение:**  
  cast AERO —  
  “Порыв ветра окутывает тебя. Ты становишься быстрее.”

### 4. **WATER (Вода)**

**Описание:** Лечит 5 HP.  
**Механика:**

* Простое восстановление здоровья  
  **Применение:**  
  cast WATER —  
  “Прохладная вода заливает раны. Ты восстанавливаешь здоровье.”

## Промежуточные заклинания (получаются в ходе игры):

### 5. **VAMPIRUM (Вампиризм)**

**Описание:** Вампирское заклинание. Вытягивает силу врага — ворует 3 HP врага и лечит тебя на 3 HP.  
**Механика:**

* У врага -3 HP
* Ты получаешь +3 к урону
* Восстанавливаешь 3 HP  
  **Применение:**  
  cast VAMPIRUM —  
  “Ты насылаешь проклятие крови. Жизненная сила врага уходит к тебе.”

### 6. **GLACIA (Лёд)**

**Описание:** Обрушивает ледяной удар, нанося 5 урона и с шансом 50% замораживает врага на 1 ход.  
**Механика:**

* 5 гарантированного урона
* 50% шанс обездвижить врага на следующий ход  
  **Применение:**  
  cast GLACIA —  
  “Ледяные осколки пронзают врага. Он может быть заморожен.”

### 7. **VITA (Жизнь)**

**Описание:** Мгновенно лечит 15 HP и снимает все негативные эффекты.  
**Механика:**

* Сильное исцеление, пригодится в критический момент  
  **Применение:**  
  cast VITA —  
  “Свет наполняет тебя. Боль отступает.”

## Финальное заклинание (по сути только для победы над боссом):

### 8. **UMBRA (Тьма)**

**Описание:** Призывает абсолютную тьму, уничтожающую всё.  
**Механика:**

* Наносит 25 урона врагу
* Похищает 15 твоего здоровья  
  **Применение:**  
  cast UMBRA —  
  “Тьма поглощает всё. Ни один свет не спасёт врага. Но за силу нужно платить.. своей кровью.”

## МЕХАНИКА БОЯ

Вся система боевых заклинаний основана на **уникальной механике набора букв**, требующей от игрока внимания, реакции и запоминания. Это не просто «нажал кнопку — и победил», а полноценная мини-игра в бою.

### Основной алгоритм:

* При каждом вызове заклинания появляется **строка из случайных букв**.
* Чтобы активировать заклинание, игрок должен **выбрать буквы в правильном порядке**, составляя нужное слово.
* Например, для заклинания IGNIS игрок поочерёдно выбирает буквы I → G → N → I → S из строки.

### Условия активации:

* Если последовательность собрана **верно** — заклинание **активируется**.
* Если игрок **ошибся** — заклинание **не срабатывает**, попытку можно повторить.

### Система получения заклинаний:

* Заклинания **выдаются только по сюжету**, при появлении **первого врага** на каждом уровне голосом в голове.
* Использовать более сложные заклинания **раньше времени нельзя**.
* Это делает механику более глубокой и сюжетно обоснованной.

## Исследование: Прототип механики

Мы реализовали **первый прототип** этой системы с помощью **SFML** — библиотеки для создания 2D-приложений.

### Как это работает:

* Запускается **игровое окно**.
* В нём отображается **случайная строка символов**.
* Игроку нужно **вводить буквы заклинания в правильном порядке**, выбирая их из строки.
* После каждого правильного нажатия строка **обновляется** новыми случайными буквами.

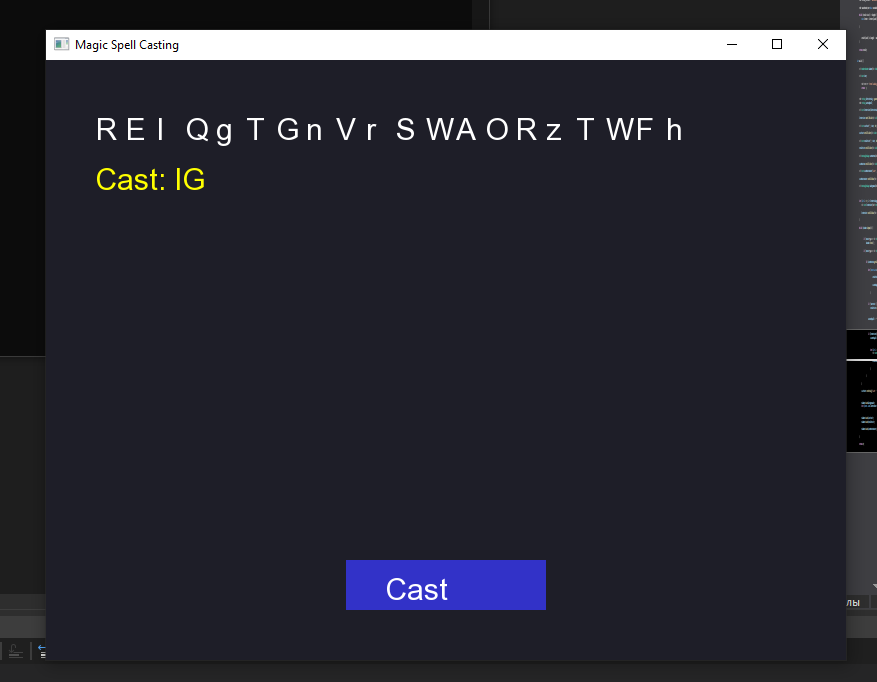
### Что получилось:

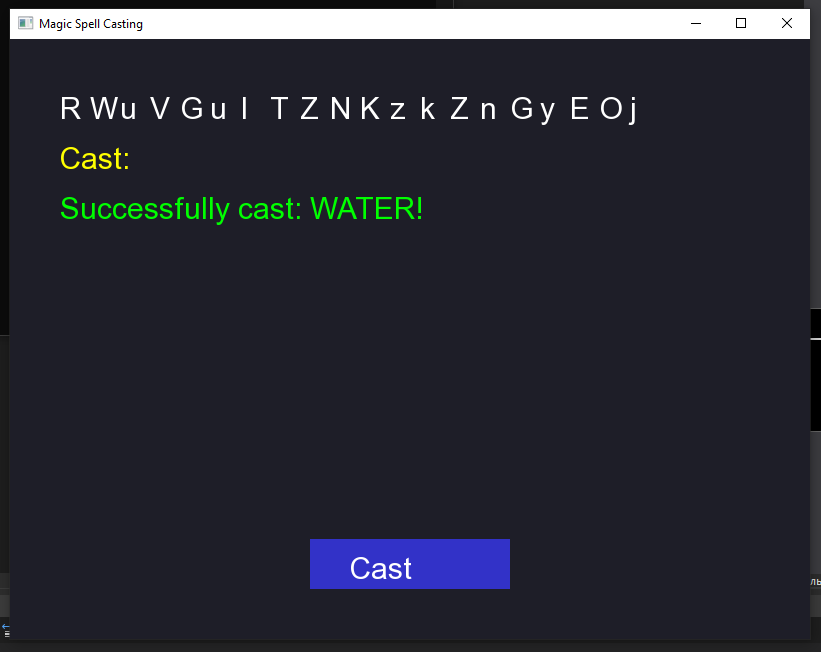
* Механика вышла **живой, интересной и затягивающей**.
* Процесс напоминает **мини-головоломку**, что выделяет бой среди обычных RPG.
* Это не «просто нажал кнопку» — заклинания нужно **помнить или записывать**, что добавляет стратегический элемент.
* В отличие от типичных текстовых RPG, где через 10 минут ты уже действуешь на автомате, здесь каждое заклинание — это **небольшая битва за фокус**.

**Вывод**

Механика получилась по-настоящему увлекательной. Благодаря простоте реализации и глубине восприятия она будет **ключевой фишкой** боевой системы игры.

**Скриншоты прототипа**





**Итоговый код прототипа**

#include <SFML/Graphics.hpp>

#include <SFML/Window.hpp>

#include <iostream>

#include <vector>

#include <ctime>

#include <cstdlib>

#include <unordered\_set>

const int WINDOW\_WIDTH = 800;

const int WINDOW\_HEIGHT = 600;

const std::vector<std::string> SPELLS = { "IGNIS", "WATER", "TERRA", "AERO" };

std::string generateLetterString(int length) {

std::string letters = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz";

std::string result;

std::unordered\_set<char> neededLetters = { 'I', 'G', 'N', 'S', 'W', 'A', 'T', 'E', 'R', 'O' };

while (result.size() < length) {

char letter = letters[rand() % letters.size()];

result += letter;

}

for (char needed : neededLetters) {

result[rand() % length] = needed;

}

return result;

}

int main() {

srand(static\_cast<unsigned int>(time(nullptr)));

sf::RenderWindow window(sf::VideoMode(WINDOW\_WIDTH, WINDOW\_HEIGHT), "Magic Spell Casting");

sf::Font font;

if (!font.loadFromFile("arial.ttf")) {

std::cerr << "Error loading font!" << std::endl;

return -1;

}

std::string letterString = generateLetterString(20);

std::string castedSpell;

sf::Text lettersText(letterString, font, 30);

lettersText.setPosition(50, 50);

lettersText.setFillColor(sf::Color::White);

sf::Text castText("", font, 30);

castText.setPosition(50, 100);

castText.setFillColor(sf::Color::Yellow);

sf::Text resultText("", font, 30);

resultText.setPosition(50, 150);

resultText.setFillColor(sf::Color::Green);

sf::RectangleShape castButton(sf::Vector2f(200, 50));

castButton.setPosition(300, 500);

castButton.setFillColor(sf::Color(50, 50, 200));

sf::Text castButtonText("Cast", font, 30);

castButtonText.setPosition(340, 510);

castButtonText.setFillColor(sf::Color::White);

sf::RectangleShape background(sf::Vector2f(WINDOW\_WIDTH, WINDOW\_HEIGHT));

background.setFillColor(sf::Color(30, 30, 40));

std::vector<sf::Text> letterTexts;

for (int i = 0; i < letterString.size(); ++i) {

sf::Text letterText(std::string(1, letterString[i]), font, 30);

letterText.setPosition(50 + i \* 30, 50);

letterText.setFillColor(sf::Color::White);

letterTexts.push\_back(letterText);

}

while (window.isOpen()) {

sf::Event event;

while (window.pollEvent(event)) {

if (event.type == sf::Event::Closed)

window.close();

if (event.type == sf::Event::MouseButtonPressed && event.mouseButton.button == sf::Mouse::Left) {

sf::Vector2i mousePos = sf::Mouse::getPosition(window);

if (castButton.getGlobalBounds().contains(static\_cast<sf::Vector2f>(mousePos))) {

bool success = false;

for (const auto& spell : SPELLS) {

if (castedSpell == spell) {

resultText.setString("Successfully cast: " + spell + "!");

success = true;

castedSpell = "";

break;

}

}

if (!success) {

resultText.setString("Failed to cast spell.");

}

castedSpell = "";

}

for (int i = 0; i < letterString.size(); ++i) {

if (letterTexts[i].getGlobalBounds().contains(static\_cast<sf::Vector2f>(mousePos))) {

castedSpell += toupper(letterString[i]);

letterString = generateLetterString(20);

letterTexts.clear();

for (int i = 0; i < letterString.size(); ++i) {

sf::Text letterText(std::string(1, letterString[i]), font, 30);

letterText.setPosition(50 + i \* 30, 50);

letterText.setFillColor(sf::Color::White);

letterTexts.push\_back(letterText);

}

}

}

}

}

castText.setString("Cast: " + castedSpell);

window.clear();

window.draw(background);

for (const auto& letterText : letterTexts) {

window.draw(letterText);

}

window.draw(castText);

window.draw(resultText);

window.draw(castButton);

window.draw(castButtonText);

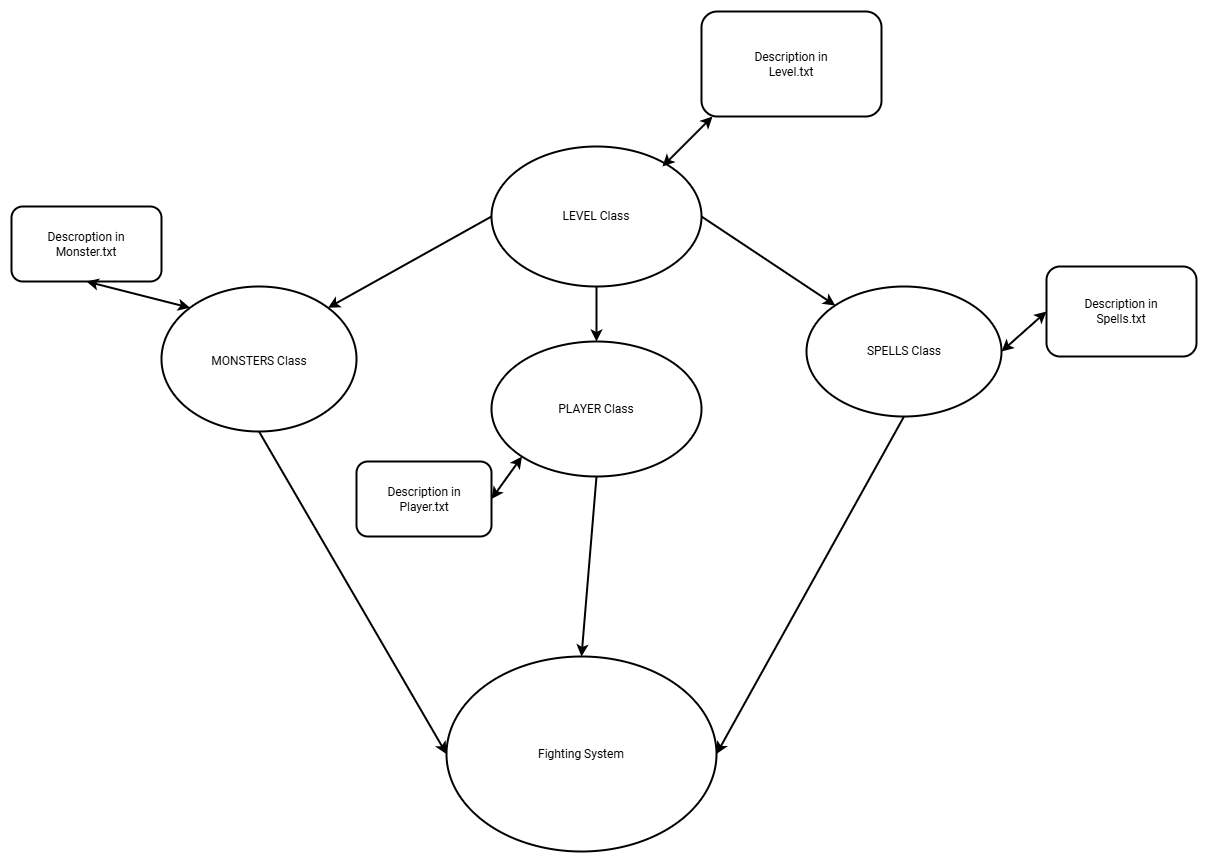
window.display();

}

return 0;

}

## ДИАГРАММА КЛАССОВ



## ФОРМУЛЫ

**Формула атаки героя:**

## int heroDamage = isFrozen || isParalyzed ? 0 : round(baseDamage \* damageModifier);

**Формула атаки монстра:**

## int monsterDamage = isFrozen ? 0 : round(baseDamage \* damageModifier);

## КЛАССЫ

**Класс: Hero**

**Характеристики:**

* int health — здоровье (изначально 15)
* int baseDamage — базовый урон (например 3)
* std::vector<Effect> activeEffects — активные эффекты
* std::vector<Spell> knownSpells — изученные заклинания
* bool isFrozen — заморозка
* bool isParalyzed — паралич
* bool isBurning — горит?
* int burnDuration — сколько ещё ходов горит
* float damageModifier — множитель урона (учитывает дебаффы)
* bool reflectNextHit — отражение (если получен от духа)

**Методы:**

### void castSpell(std::string name, Monster& target)

* Применяет заклинание по имени к цели (монстру).
* Учитывает эффекты, урон, исцеление, иммунитеты.
* Активирует все механики заклинания (включая баффы на героя).

### void attack(Monster& target)

* Атакует монстра с учётом всех активных баффов и дебаффов.
* Если активен усиление атаки — увеличивает урон.
* Если парализован или заморожен — ход пропускается.
* Проверяет отражение урона монстром.

### void applyEffects()

* Применяется в начале хода.
* Обрабатывает все активные эффекты (горение, усиление, уклонение и т.п.).
* Обновляет длительность, снимает закончившиеся.

### void receiveDamage(int amount)

* Уменьшает текущее здоровье на amount.
* Проверяет на смерть (если здоровье ≤ 0).

### void heal(int amount)

* Увеличивает здоровье (не больше максимума).
* Используется заклинаниями WATER, VAMPIRUM, VITA.

**Класс: Monster**

**Характеристики:**

* std::string name — имя монстра (например, "Зомби")
* int baseDamage — базовый урон
* int health — текущий запас здоровья
* std::vector<Effect> activeEffects — активные эффекты, действующие на монстра (горение, замедление и т.д.)
* Effect specialEffect — особый эффект (например, "тройной удар", "паралич")
* float effectChance — шанс применения specialEffect (например, 0.25 для 25%)
* bool immuneToFire — иммунитет к эффекту огня (например, "Ревенант")
* bool immuneToIce — иммунитет к эффекту льда (например, "Дух")
* bool isFrozen — не может действовать в текущем ходу
* bool isParalyzed — шанс пропуска хода
* float damageModifier — множитель урона (например, 0.5 при замедлении)
* bool reflectNextAttack — если активен, следующая атака будет отражена
* bool attackedLastTurn — флаг для логики типа "Зомби" (ответный урон игроку, если был атакован)

**Методы:**

 void attack(Hero& hero)

* Атакует героя с учётом всех модификаторов урона, заморозки и т.д.
* Если isFrozen — пропускает ход.
* Учитывает damageModifier.
* Применяет applySpecialEffect() после атаки.

 void applySpecialEffect(Hero& hero)

* Срабатывает с шансом effectChance.
* Вешает specialEffect на героя или влияет напрямую (например, тройной удар).

 void applyEffects()

* Обрабатывает каждый активный эффект в списке activeEffects.
* Например, горение наносит урон, замедление влияет на damageModifier.

 void receiveDamage(int amount)

* Уменьшает health.
* Ставит флаг attackedLastTurn = true.

 void applyImmunityCheck(Effect e)

* При попытке применить эффект проверяет иммунитеты.
* Например, e.name == "Burning" и immuneToFire == true → эффект не вешается.

 void onTurnStart()

* Сброс attackedLastTurn = false
* Применение applyEffects()

 bool isAlive()

* Возвращает true, если health > 0

## Класс: Spell

**Характеристики:**

* std::string name — название заклинания (например, "IGNIS")
* std::string description — краткое описание действия (например, "Поджигает врага, нанося урон каждый ход")
* std::string effectType — тип эффекта ("Burning", "Slow", "Heal", "Freeze", "Buff", "Vampirism", "Ultimate")
* int instantDamage — мгновенный урон, наносимый при применении
* int duration — длительность эффекта в ходах (если есть, 0 — мгновенный)
* int periodicDamage — урон, который наносится в конце каждого хода при длительном эффекте
* int healAmount — количество восстанавливаемого здоровья
* float chanceToApply — шанс наложения эффекта (например, 0.5 для GLACIA)
* std::string unlockLocation — локация, с которой заклинание становится доступно  
  (например, "Начальная" для IGNIS, "Сожжённые джунгли" для VAMPIRUM)
* bool isSelfTargeted — true, если заклинание применяется на героя (WATER, VITA и т.п.)

**Методы:**

### • void apply(Monster& target, Hero& caster)

* Применяет эффект заклинания к цели (монстру), учитывая иммунитеты.
* Вешает Effect на монстра (горение, заморозка и т.п.) или наносит урон.

### • void applySelf(Hero& hero)

* Используется, если isSelfTargeted == true.
* Применяет эффект на героя (лечение, усиление, снятие эффектов и т.д.)

### • bool isAvailableIn(std::string location)

* Проверяет, может ли быть использовано заклинание в данной локации.
* Сравнивает текущую локацию героя с unlockLocation.