Dokumentacja projektu Labirynt

Autor: Igor Macedoński

Wrocław 2022

Spis treści

Wprowadzenie

Napisana przeze mnie gra to konsolowe mini rpg, w którym bohater porusza się po różnych lokalizacjach, walczy z potworami i rozwiązuje przy okazji zadania. Celem gry jest pokonanie Gothmoga, czyli najmocniejszego przeciwnika, który znajduje się w lokalizacji nr 13. Numer lokalizacji odpowiada trudności przeciwników w niej, więc warto (jeżeli nie chce się przegrać) odwiedzać lokalizację w kolejności ich numerów, rozwiązując przy tym zadania i zdobywając wyposażenie. Przed pojedynkami z mocniejszymi potworami warto zrobić zapis gry, ponieważ w przypadku śmierci bohatera gra się kończy i (jeżeli nie zrobiło się wcześniej żadnego zapisu) trzeba zaczynać ją od początku.

Rozdział 1

Spis prostych klas

1.1 Item

Klasa ta służy do reprezentowania przedmiotów w grze. Posiada ona trzy atrybuty:

(1) name - Czyli oczywiście nazwę danego przedmiotu.

(2) type - Typ przedmiotu (weapon, armor, potion lub quest).

(3) value - Wartość przedmiotu; w przypadku przedmiotu typu Quest ta wartość jest None, w przypadku broni wartość to bonus do ataku itd.

1.2 Enemy

Klasa ta służy do reprezentowania przeciwników w grze. Mamy w niej atrybuty takie jak: name, attack, health, defence. Ponadto klasa ta zawiera fundamentalne metody takie jak: get_damage(), apply_damage(), do_attack(). Z klasy tej dziedziczy znacznie bardziej złożona klasa Hero.

1.3 Vertex

Implementacja tej klasy jest taka sama jak na wykładzie, dziedziczy z niej bardziej złożona klasa Place.

1.4 Graph

Implementacja tej klasy jest niemalże taka sama jak na wykładzie. Jedyną różnica jest dodany atrybut visited_places bedący zbiorem przechowującym klucze obiektów typu Place, w których bohater gry już był. Atrybut ten jest istotny w graficznym przedstawieniu mapy w matplotlibie.

1.5 Quest

Klasa ta służy do reprezentowania Questów w grze. Questy są niejako "ruchome", tzn. początkowo są elementem obiektu NPC, następnie (w chwili, gdy bohater gry rozpocznie dialog z NPC) stają się elementem Hero, a gdy zostaną ukończone przestają istnieć. Klasa ta zawiera atrybuty: name, reward, monsters, find_item. Dwa początkowe są dosyć oczywiste. Atrybut monsters jest początkowo równy 3 i w przypadku zabicia odpowiedniego potwora jest zmniejszany. Atrybut find_item jest początkowo równy False, w przypadku znalezienia odpowiedniego przedmiotu zmienia się na True. Ponadto, klasa ta zawiera trzy metody: was_killed(), was_found() oraz is_done(), które są wykorzystywane wewnątrz klasy Hero i służą do sprawdzania czy Quest został już ukończony.

1.6 NPC

Ta klasa reprezentuje obiekty NPC w grze i jest raczej bardzo prosta. Posiada dwa atrybuty: name oraz quest (obiekt typu Quest) oraz dwie metody: text() i set_none(), z których pierwsza służy do wypowiedzenia treści zadania, a druga do sprawienia, że atrybut quest jest ustawiony na None (służy to głównie temu, żeby bohater gry nie mógł wziąć tego samego Questa więcej niż raz).

Rozdział 2

Spis złożonych klas

2.1 Place

Klasa ta dziedziczy z klasy Vertex. Ponadto, jej dodatkowymi atrybutami są:

(1) chest, czyli lista obiektów typu Item.

(2) enemies, czyli lista obiektów typu Enemy.

(3) npc, czyli obiekt typu NPC (w niektórych miejscach gry NPC nie ma, wtedy jest on równy None).

Metody dostępne w tej klasie to: add_npc(npc), add_item(item), add_enemy(enemy), get_enemies() oraz kill_enemy(enemy). Większość z nich jest dosyć oczywista. Mniej oczywiste metody to kill_enemy, która usuwa obiekt typu Enemy z listy enemies w chwili, gdy bohater wygra z nim pojedynek oraz get_enemies(), które zwraca atrybut enemies.

2.2 Hero

Klasa ta reprezentuje bohatera gry. Dziedziczy ona z klasy Enemy. Jej dodatkowe atrybuty to:

(1) equipment - Jest to słownik, którego klucze to "armor" oraz "weapon", wartościami oczywiście jest None lub Item odpowiedniego typu.

(2) inventory - Słownik, kluczem jest atrybut name obiektu Item, a wartością obiekt item.

(3) backpack - Również słownik, klucze to "small potion", "medium potion", "big potion" a wartości to ich liczba, którą bohater zebrał podczas gry.

(4) experience - Doświadczenie, jak wzrośnie do wartości 3 to zwiększa się lvl o 1 w górę.

(5) max_health - Czyli maksymalny poziom zdrowia, w przeciwieństwie do atrybutu health jest on stały i nie zmienia się podczas walki (zwiększa się jedynie gdy wbijemy wyższy lvl).

(6) active_quests - Słownik aktywnych Questów, klucz to nazwa Questa, a wartość to obiekt typu Quest.

Metody w tej klasie to:

(1) increase_health(value) - Zwiększa atrybut health.

(2) use_potion(name) - Używa mikstury odpowiedniego rozmiaru, korzysta z (1).

(3) put_off_armor() / put_off_weapon() - Ściąga ubraną zbroję/broń i wkłada ją do inventory.

(4) put_on(name) - Ubiera zbroję/broń, jeżeli coś już jest ubrane to pierwsze korzysta z (3).

(5) add_quest(quest) - Dodaje do słownika active_quests quest.

(6) get_item(item) - Dodaje do inventory item, w przypadku gdy jego typ to nie "potion". W przeciwnym razie zwiększa liczbę odpowiedniego typu mikstur w backpack.

(7) get_reward() - Sprawdza czy bohater może odebrać nagrodę za questy w active_quests, jeżeli tak to dostaję nagrodę do inventory.

(8) light_strike()/strong_strike() - Symuluje wyprowadzenie lekkiego/silnego ataku, które ma prawdopodobieństwo 80%/40% sukcesu. Silny atak zadaje oczywiście większe obrażenia.

(9) do_attack($strong = False$) - Nadpisana metoda Enemy, żeby bohater wyprowadzał trochę mocniejsze ataki niż przeciwnicy i żeby mógł decydować czy chce wykonać lekki czy silny atak.

(10) get_experience() - Dodaje +1 doświadczenia.

(11) lvl_up() - Zwiększa lvl o 1.

(12) quest_item() - Sprawdza czy bohater nie znalazł już czasem przedmiotu z zadania, jeżeli tak to zmienia atrybut find_item obiektu Quest na True.

(13) check_item(name) - Wypisuje informacje na temat danego przedmiotu jeżeli jest w inventory, jeżeli go nie ma to rzuca wyjątek.

(14) show_equipment() - Jak w nazwie.

(15) drop_item(name) - Usuwa niepotrzebny przedmiot z inventory.

2.3 Game

Klasa ta jest najbardziej złożona i jest czymś w rodzaju silnika gry. Korzysta ona z klas Graph, Hero, Item, NPC, Quest oraz Enemy. Ponadto, wykorzystuje również wbudowane biblioteki takie jak matplotlib, networkx oraz pickle. Jej atrybuty to:

(1) world - Świat gry, obiekt typu Graph.

(2) hero - Bohater gry, obiekt typu Hero.

(3) current_place - Klucz obiektu typu Place, w którym aktualnie znajduje się bohater gry.

(4) view - Obiekt typu networkx, odpowiednik world do reprezentacji graficznej w matplotlib.

(5) pos - Układ labiryntu, obiekt typu networkx, również służący do reprezentacji graficznej świata gry.

(6) list_of_commands - Lista komend dostępnych w grze.

Z racji tego, że klasa ta zawiera sporo metod, niektóre (te bardziej oczywiste) będą opisane bardziej skrótowo. W skład klasy wchodzą następujące metody:

(1) clear_pickles() - Statyczna metoda służąca do czyszczenia plików, do których zapisywane są dane gry.

(2) check_statistics() - Stringowa reprezentacja statystyk bohatera.

(3) check_inventory() - Stringowa reprezentacja przedmiotów w inventory bohatera.

(4) show_enemies() - Wypisanie przeciwników w danym miejscu (tzn. w obecnej lokalizacji).

(5) show_ways() - Wypisanie aktualnego miejsca oraz lokalizacji, do których można się z tego miejsca udać.

(6) go_to(destination) - Przeniesienie bohatera do innej lokalizacji. Metoda sprawdza czy można się do tej lokalizacji udać i czy zabite zostały wszystkie potwory (w przypadku, gdy chcemy udać się do lokalizacji z wyższym numerem).

(7) show_npc() - Wypisuje atrybut name obiektu NPC będącego w obecnej lokalizacji lub stosowny komunikat, gdy w lokalizacji nie ma npc.

(8) talk_to_npc() - Jeżeli w danej lokalizacji jest NPC to korzysta z metody text dla NPC oraz dodaje do listy aktywnych questów bohatera zadanie od NPC i ustawia atrybut quest u NPC na None. W przeciwnym wypadku wyświetla stosowny komunikat.

(9) fight(enemy) - Metoda obsługująca pojedynki w grze. W przypadku walki z "final bossem" gry wyświetla na początku dodatkowy tekst. Metoda ta zarządza wyborem lekkiego lub silnego ataku i aplikuje odpowiednie metody na bohaterze (czyli obiekcie klasy Hero) takie jak do_attack() i get_damage() oraz na przeciwniku (obiekcie typu Enemy).

(10) take_items - Jeżeli w danej lokalizacji nie ma potworów to dodaje przedmioty dostępne w tej lokalizacji do inventory bohatera, w przeciwnym wypadku wyświetla stosowny komunikat. Ponadto po dodaniu przedmiotów sprawdza czy nie został dodany przedmiot z zadania (jeżeli został, to bohater dostaje nagrodę za wykonanie Questa).

(11) help - Wyświetla listę komend dostępnych w grze.

(12) action(command) - Odpowiada za obsługę tego, co gracz wpisuje z klawiatury podczas gry. Jeżeli podana komenda jest poprawna to dokonywane jest działanie opisane w metodach, jeżeli nie to wyświetlany jest stosowny komunikat.

(13) play() - Metoda ta odpowiada za wywołanie gry, początkowo wyświetla komunikat związany z tym czy gracz chce zacząć nową grę czy wczytać zapis gry. Następnie w nieskończonej pętli wywoływana jest metoda (12).

(14) show_map - Metoda wyświetla mapę gry w matplotlibie korzystając oczywiście z metod klasy networkx. W odpowiedni sposób kolorowane są wierzchołki grafu, które już były odwiedzone oraz te, które są jeszcze przez gracza nieodwiedzone.

(15) save_game()/load_game() - Metoda odpowiedzialna za zapisanie/wczytanie gry.

(16) generate_new_game() - Metoda odpowiadająca za utworzenie nowej gry. Tworzy wszystkie instancje Place, Enemy itd.