**Game**

Klasa Game jest główną klasą gry i nie jest przypisana do żadnego konkretnego folderu. Pełni kluczową rolę w funkcjonowaniu gry. Jest odpowiedzialna za dostarczanie najważniejszych bibliotek, takich jak SDL2, wszystkim innym klasom w grze. Współpracuje również z plikiem Main.cpp, które razem odpowiadają za działanie gry

Klasa Game tworzy nieskończoną pętlę gry, która kontroluje przebieg rozgrywki. Odpowiada za inicjalizację i zarządzanie GameLevelManager’em, który jest odpowiedzialny za tworzenie i zarządzanie poziomami gry.

Klasa Game posiada atrybuty, które odpowiadają za wymiary aplikacji i gry. Mogą to być na przykład szerokość i wysokość okna gry, rozdzielczość ekranu itp.

Jednak najważniejsze atrybuty klasy Game to static SDL\_Renderer\* mainGameRender oraz static SDL\_Event mainGameEvent. Są to obiekty biblioteki SDL2, które są niezbędne do renderowania grafiki i obsługi zdarzeń w grze. Atrybuty te są statyczne, co oznacza, że są dostępne dla wszystkich instancji klasy Game i mogą być używane przez inne klasy w celu renderowania grafiki i obsługi zdarzeń.

**Folder SDL2\_Managers:**

Folder SDL2\_Managers zawiera klasy, które dostarczają niezbędne narzędzia do zarządzania czcionkami, obrazami i tekstem w grze. Dzięki nim można w sposób intuicyjny i wygodny renderować napisy i obrazy na ekranie, dodając w ten sposób estetyczny i funkcjonalny aspekt do pikselowej gry 2D.  
Oto opis tych klas:

**FontTextureManager**

Klasa FontTextureManager to prawdziwy magik czcionek w grze. Jest ona odpowiedzialna za załadowanie odpowiedniej czcionki z pliku i przygotowanie tekstury, która może być wykorzystana do wyświetlania tekstu na ekranie gry. FontTextureManager umożliwia łatwe manipulowanie rozmiarem, kolorem i stylem tekstu, dając Ci kontrolę nad wyglądem napisów w grze. Ta klasa jest niezastąpiona, gdy chcesz wyświetlić informacje dla gracza, takie jak wyniki, wskazówki czy dialogi postaci.

**ImageTextureManager**

Klasa ImageTextureManager to niezawodny obrazowy pomocnik, który umożliwia łatwe ładowanie i wyświetlanie obrazów na ekranie gry. Załaduje obraz z odpowiedniej ścieżki, a następnie przygotuje go do renderowania na ekranie. Dzięki ImageTextureManagerowi możesz wyświetlać tła, postacie, przedmioty i wiele innych elementów graficznych w swojej grze. Oferuje on również funkcje skalowania, przycinania i manipulowania wyglądem obrazu, aby dostosować go do potrzeb gry.

**TextObject**

Klasa TextObject to magiczny obiekt tekstowy, który reprezentuje napis, który można wyświetlić na ekranie gry. Ten obiekt korzysta z FontTextureManager i ImageTextureManager, aby przetworzyć tekst na teksturę i wyświetlić go w odpowiednim miejscu na ekranie. Dzięki TextObject możesz tworzyć dynamiczne napisy, takie jak wyniki, punktacja czy informacje o zdrowiu postaci. Możesz kontrolować styl, kolor i rozmiar tekstu, aby dostosować go do wyglądu gry i tworzyć wspaniałe efekty wizualne.

**Folder MapUtils**

Folder SDL2\_ MapUtils zawiera klasy, które dostarczają narzędzi i funkcji potrzebnych do zarządzania mapami w grze. Dzięki nim możesz tworzyć mapy za pomocą plików tekstowych, generować losowe poziomy gry i dostosowywać położenie bohatera oraz elementów w grze. To umożliwia tworzenie różnorodnych i ekscytujących poziomów, które stanowią kluczową część rozgrywki i zapewniają graczom interesujące wyzwania do pokonania.  
Oto opis tych klas:

**\_VirtualMap**

Klasa \_VirtualMap jest interfejsem dla wszystkich map w grze. Wszystkie klasy dziedziczące po \_VirtualMap będą miały dostęp do ścieżek do niezbędnych grafik potrzebnych do zbudowania mapy. Klasa ta umożliwia również konstrukcję mapy z pliku tekstowego, zapisywanie mapy do pliku tekstowego oraz generowanie losowo wygenerowanej mapy. Dzięki temu interfejsowi klasy dziedziczące będą miały wspólne funkcjonalności i będą mogły operować na mapach w sposób spójny i elastyczny.

**BackgroundMap**

Klasa BackgroundMap zajmuje się konkretną konstrukcją poziomów gry. Korzysta ona w pełni z interfejsu \_VirtualMap. Posiada informacje dotyczące położenia środka mapy, położenia bohatera gracza, przesunięcia bohatera względem środka mapy i wiele innych. Dzięki tym informacjom BackgroundMap może dokładnie określić, jak zbudować poziom gry na podstawie dostępnych danych. Może tworzyć różnorodne elementy mapy, takie jak tła, platformy, przeszkody, przedmioty do zebrania i wiele więcej, aby zapewnić bogate i zróżnicowane środowisko gry.

**Folder LevelUtils**  
Folder LevelUtils zawiera klasy, które są odpowiedzialne za tworzenie i zarządzanie poziomami gry. Umożliwiają tworzenie dynamicznych i zróżnicowanych poziomów, które dostarczają różnorodności i wyzwań dla graczy.  
Oto opis tych klas:

**GameLevel**

Klasa GameLevel reprezentuje konkretny poziom gry. Posiada unikalne ID, które identyfikuje dany poziom, oraz wektory zawierające wszystkie GameObjects (obiekty gry). GameObjects to obiekty takie jak przeciwnicy, przedmioty i obiekty interaktywne. Obiekty gry generowane są w losowej ilości i na losowych pozycjach, co sprawia, że każda rozgrywka jest inna i bardziej zróżnicowana.

**GameLevelManager**

Klasa GameLevelManager posiada wektor zbudowany z obiektów klasy GameLevel. Jest odpowiedzialna za zarządzanie poziomami gry. Ta klasa obsługuje zderzenia między GameObjects, co oznacza, że zajmuje się wywoływaniem odpowiednich akcji w zależności od interakcji między obiektami. Na przykład może kontrolować walkę z przeciwnikami, zbieranie monet czy przechodzenie przez portale. GameLevelManager pełni kluczową rolę w kontrolowaniu przebiegu gry i zarządzaniu poziomami.

**Folder GameObjects**

Folder GameObjects zawiera obiekty gry, które współdziałają z postacią gracza. Jest to kluczowy folder, który ułatwia rozwój i rozbudowę gry poprzez dostarczanie interfejsów i wartości generowanych losowo, co urozmaica każdą rozgrywkę.

Zawiera w sobie podfoldery SpecEnemyObjects, SpecInteractiveObjects i SpecItemObjects, które z kolei zawierają podklasy dziedziczące po odpowiednich interfejsach (kolejno \_EnemyObject, \_InteractiveObject, \_ItemObject). Każdy podfolder zawiera konkretne typy obiektów, takie jak różni przeciwnicy, interaktywne obiekty i przedmioty. Te klasy specjalizowane mogą dostosowywać zachowanie i właściwości obiektów do konkretnych potrzeb gry, tworząc różnorodność i unikalność wśród elementów rozgrywki.

Zawiera również w sobie podfolder HeroUtils, który zawiera klasy powiązane z MainHeroObject. HeroBasicClass przechowuje bazowe statystyki bohatera i pomaga w obsłudze kolizji. HeroChoices obsługuje wybory i akcje podejmowane przez gracza, a HeroKeyboardHandler zbiera informacje o naciśniętych przyciskach na klawiaturze, umożliwiając grze reagowanie na działania gracza. Te klasy w HeroUtils są istotne dla sterowania bohaterem gracza i umożliwiają interakcję z innymi obiektami w grze.

Poniżej znajduje się opis poszczególnych klas w tym folderze:

**\_\_VirtualObject**

Klasa \_\_VirtualObject jest interfejsem dla wszystkich obiektów gry. Posiada metody i atrybuty, które umożliwiają komunikację z ImageTextureManager, co pozwala nam na renderowanie obiektów na ekranie gry. Każdy obiekt gry musi posiadać metody Update i Render, które odpowiednio aktualizują stan obiektu i rysują go na ekranie. Dzięki temu interfejsowi, klasy dziedziczące po \_\_VirtualObject mogą definiować różne typy obiektów w grze i dostosowywać ich zachowanie w zależności od potrzeb.

**\_EnemyObject**

Klasa \_EnemyObject jest interfejsem dla wszystkich przeciwników w grze. Posiada metody i atrybuty odpowiedzialne za poruszanie się przeciwników oraz może generować różne parametry dla konkretnych obiektów. Dodatkowo, klasa ta posiada atrybuty umożliwiające rozróżnienie siły przeciwników, co pozwala na wprowadzenie różnych poziomów trudności w grze. Przeciwnicy mogą mieć różne umiejętności, takie jak atakowanie gracza, obrona przed atakami, czy poruszanie się w określonych wzorcach. Dzięki klasie \_EnemyObject, możesz tworzyć różnorodne przeciwników, które stanowią wyzwanie dla gracza i dodają dynamiki do rozgrywki.

**\_InteractiveObject**

Klasa \_InteractiveObject jest interfejsem dla wszystkich obiektów interaktywnych w grze. Posiada metody i atrybuty odpowiedzialne za interakcję z tymi obiektami. Dodatkowo, klasa ta posiada atrybuty umożliwiające rozróżnienie wymaganego wyzwania, czyli jakie statystyki gracza muszą być spełnione, aby móc skorzystać z danego obiektu. Obiekty interaktywne mogą mieć różne funkcje, takie jak otwieranie skrzyń, aktywowanie teleportów, czy wyzwalanie innych zdarzeń specjalnych. Dzięki klasie \_InteractiveObject, możesz tworzyć różnorodne obiekty, z którymi gracz może interakcjonować, co dodaje głębi i możliwości w rozgrywce.

**\_ItemObject**

Klasa \_ItemObject jest interfejsem dla wszystkich przedmiotów, które mogą być automatycznie zebrane przez gracza. Posiada metody i atrybuty odpowiedzialne za efekty zbieranych przedmiotów. Dodatkowo, klasa ta posiada atrybuty umożliwiające określenie siły efektów zebranych przedmiotów, takich jak przywracanie zdrowia, zadawanie obrażeń, czy poprawianie statystyk gracza. Przedmioty mogą mieć różne właściwości i wpływać na rozgrywkę w różnorodny sposób. Dzięki klasie \_ItemObject, możesz tworzyć różne przedmioty, które gracze mogą zbierać, co daje im nagrody i możliwość rozwijania swojej postaci.

**MainHeroObject**

Klasa MainHeroObject reprezentuje głównego bohatera gracza. Skupia w sobie inne pomocnicze klasy z podfolderu HeroUtils, które pomagają w zarządzaniu postacią gracza. Może przechowywać bazowe statystyki bohatera (np. zdrowie, punkty doświadczenia, siłę, zwinność, inteligencję) oraz obsługiwać kolizje z innymi obiektami. Dodatkowo, klasa ta może reagować na wybory i akcje podejmowane przez gracza za pomocą maszyny stanów, co pozwala na różnorodne zachowanie bohatera w zależności od kontekstu gry. MainHeroObject jest kluczową klasą reprezentującą gracza i umożliwiającą sterowanie nim w grze.

**Podfolder SpecEnemyObjects**

Podfolder SpecEnemyObjects zawiera podklasy \_EnemyObject, czyli różne typy przeciwników w grze, które mają unikalne cechy i zachowania w grze. Gracz będzie musiał podejmować odpowiednie działania, takie jak zebranie jabłek, aby poprawić relacje z przeciwnikami lub rozwijać swoje statystyki, aby pokonać ich. To wprowadza element strategii i wyzwań dla gracza, zachęcając go do rozwijania swojej postaci i podejmowania odpowiednich decyzji w rozgrywce.  
Oto opis poszczególnych klas w tym podfolderze:

**EnemyMageObject**

Klasa EnemyMageObject reprezentuje przeciwnika-maga. Na początku gry ta postać nie będzie atakować bohatera gracza. Jednak wraz z zebraniem każdej mikstury oraz otwarciem skrzyni, stosunki z tymi przeciwnikami będą się pogarszać, aż w końcu będą oni atakować gracza i zadawać podwójne obrażenia. Zbieranie jabłek poprawia relacje z nimi. Aby pokonać przeciwników-magów, gracz będzie musiał posiadać odpowiednio wysoką statystykę siły (Strength).

**EnemySentinelObject**

Klasa EnemySentinelObject reprezentuje przeciwnika-strażnika. Na początku gry ta postać nie będzie atakować bohatera gracza. Jednak za każdym razem, gdy bohater przejdzie przez teleport, stosunki z tymi przeciwnikami będą się pogarszać, aż w końcu będą oni atakować gracza i zadawać podwójne obrażenia. Podobnie jak w przypadku przeciwników-magów, zbieranie jabłek poprawia relacje z nimi. Aby pokonać przeciwników-strażników, gracz będzie musiał posiadać odpowiednio wysoką statystykę siły (Strength).

**Podfolder SpecInteractiveObjects**

Podfolder SpecInteractiveObjects zawiera podklasy \_InteractiveObject, czyli różne typy obiektów interaktywnych w grze, które wpływają na rozgrywkę i relacje z przeciwnikami. Gracz będzie musiał rozważnie korzystać z tych obiektów, biorąc pod uwagę swoje statystyki i konsekwencje wyborów, które mogą mieć wpływ na rozwój gry.  
Oto opis poszczególnych klas w tym podfolderze:

**ChestObject**

Klasa ChestObject reprezentuje interaktywny obiekt w postaci skrzyni. Gracz może otworzyć tę skrzynię, używając odpowiedniego przycisku i posiadając odpowiednio wysoką statystykę zwinności (Agility). Po otwarciu skrzyni, gracz otrzymuje wzmocnienie losowej statystyki, jednak relacje z przeciwnikami-magami się pogarszają. Dodatkowo, gracz otrzymuje określoną ilość punktów wyniku (ScorePoints).

**TeleportObject**

Klasa TeleportObject reprezentuje interaktywny obiekt w postaci teleportu. Gracz może podróżować za pomocą tego teleportu, używając odpowiedniego przycisku i posiadając odpowiednio wysoką statystykę inteligencji (Intelligence). Korzystając z teleportu, gracz może przenieść się do kolejnych poziomów gry. Jednak korzystanie z teleportu pogarsza relacje z przeciwnikami-strażnikami.

**Podfolder SpecItemObjects**

Podfolder SpecItemObjects zawiera podklasy \_ItemObject, czyli różne typy przedmiotów do automatycznego zbierania w grze, które można zbierać w grze. Każdy z tych przedmiotów ma unikalne właściwości i wpływ na rozgrywkę, zarówno pod względem zdrowia, relacji z przeciwnikami, jak i punktów wyniku. Gracz będzie musiał decydować, jakie przedmioty zbierać, biorąc pod uwagę ich efekty i konsekwencje w grze.  
Oto opis poszczególnych klas w tym podfolderze:

**AppleItemObject**

Klasa AppleItemObject reprezentuje zbieralny przedmiot w postaci jabłka. Zbieranie jabłek może mieć różne efekty, takie jak przywracanie zdrowia lub zadawanie obrażeń. Jednak najważniejsze jest to, że zbieranie jabłek poprawia relacje z każdym typem przeciwnika. To oznacza, że zebranie jabłek może wpłynąć na postawę przeciwników wobec gracza.

**CoinItemObject**

Klasa CoinItemObject reprezentuje zbieralny przedmiot w postaci monety. Zbieranie monet przynosi określoną ilość punktów wyniku (ScorePoints). Monety są wartościowe i mogą być wykorzystane do zwiększenia wyniku gracza.

**PotionItemObject**

Klasa PotionItemObject reprezentuje zbieralny przedmiot w postaci mikstury. Zbieranie mikstur zawsze przywraca zdrowie gracza. Jednak warto zauważyć, że zbieranie mikstur pogarsza relacje z przeciwnikami-magami. Zbierając mikstury, gracz może odnieść korzyści zdrowotne, ale może również narazić się na nieprzychylność przeciwników.

**Podfolder HeroUtils**

Podfolder HeroUtils zawiera klasy powiązane z klasą MainHeroObject (która reprezentuje głównego gracza) i pomagają w zarządzaniu statystykami, wyborami oraz obsługą klawiatury związanej z głównym bohaterem gracza. Te klasy są istotne dla funkcjonowania bohatera gracza i umożliwiają interakcję z grą w sposób, który odzwierciedla decyzje i działania gracza.  
Oto opis poszczególnych klas w tym podfolderze:

**HeroBasicClass**

Klasa HeroBasicClass reprezentuje bazowe statystyki bohatera gracza. Może zawierać informacje takie jak HeroHealthPoints (punkty zdrowia bohatera), ScorePoints (punkty wyniku), Strength (siła), Agility (zwinność), Intelligence (inteligencja) itp. Ta klasa pomaga w zarządzaniu statystykami bohatera oraz rozpatrywaniu kolizji z innymi obiektami w grze.

**HeroChoices**

Klasa HeroChoices reprezentuje wybory i akcje podejmowane przez gracza jako maszynę stanów. Może zawierać informacje o relacjach gracza z przeciwnikami, takich jak zmiany w tych relacjach w zależności od podjętych akcji. Ta klasa pomaga śledzić i zarządzać interakcjami bohatera gracza z otoczeniem i przeciwnikami.

**HeroKeyboardHandler**

Klasa HeroKeyboardHandler zbiera informacje o naciśniętych przyciskach na klawiaturze przez gracza i przekazuje je dalej w celu odpowiedniego zareagowania w grze. Dzięki tej klasie gra może odczytywać intencje gracza związane z poruszaniem się i wykonywaniem działań przez bohatera. HeroKeyboardHandler pomaga w obsłudze sterowania bohaterem gracza na podstawie akcji gracza na klawiaturze.