APOCALYPSE PROJECT

7: Piotr Łach (256761), Jakub Szpak(252782)

Prowadzący: mgr inż. Bogdan Gulowaty
Termin zajęć: Czwartek, 18:55-20:35
Kierunek: Informatyka Techniczna
Rok 1, Semestr 2

Spis treści

1 Analiza czasownikowo-rzeczownikowa **3**

2 Diagramy aktywności **3**
2.1 Diagram aktywności - symulacja 3
2.2 Diagram aktywności - człowiek 4
2.3 Diagram aktywności - zarażony 5
2.4 Diagram aktywności - ruch 6

3 Diagram przypadków użycia **7**

4 Diagramy klas **8**
4.1 Diagram projektu 8
4.2 Diagram pakietu character 9
4.3 Diagram pakietu game 10
4.4 Diagram pakietu move 11
4.5 Diagram pakietu displayer 11

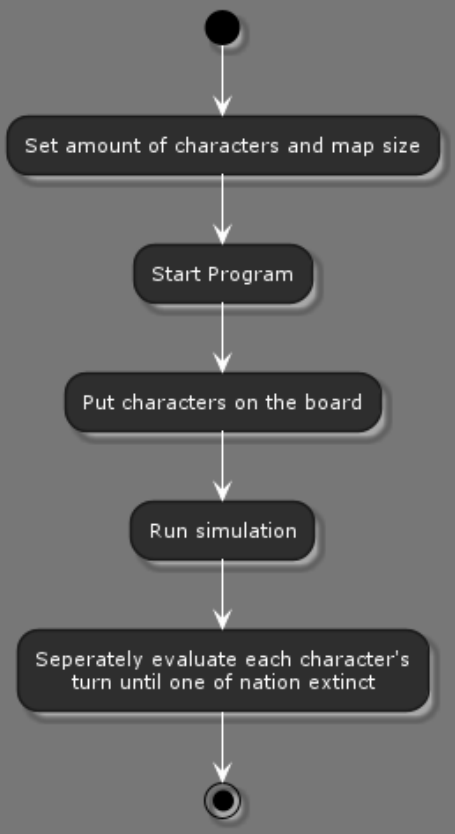
5 Karty CRC **12**

1 Analiza czasownikowo-rzeczownikowa

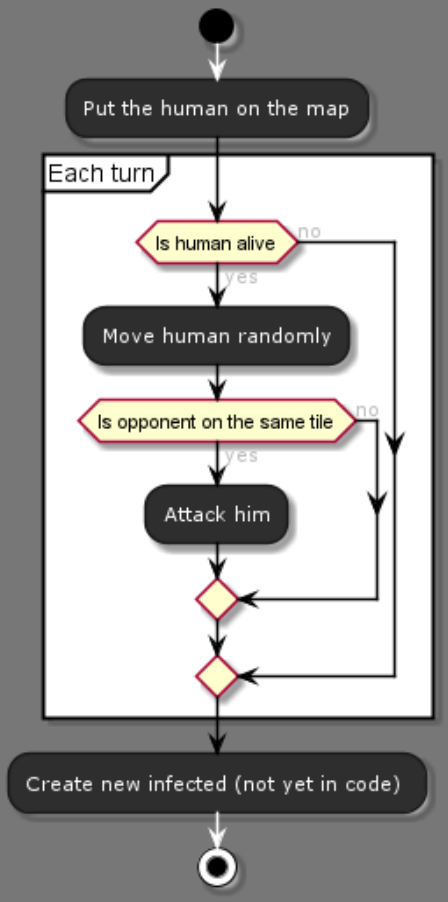
Projektujemy symulację agentową, w której stoczy się walka o przetrwanie ludzi lub zwycięstwo zarażonych. Postacie będą losowo rozmieszczane na dwuwymiarowej planszy o zadanej wielkości. Każda z klas postaci (ludzie, zarażeni) będzie dążyła do eliminacji przeciwnika. Użytkownik dostanie możliwość ustalenia ilości obiektów danej klasy i na podstawie sporządzonych algorytmów system wyświetli wynik symulacji. Podczas trwania programu obiekty będą zmieniały swoje pozycje na mapie w celu jak najskuteczniejszego wykluczenia przeciwnika. Symulacja będzie zawierała również elementy losowe takie jak pozycje początkowe ludzi i zarażonych czy bonusy przyznawane dla poszczególnych klas.

2 Diagramy aktywności

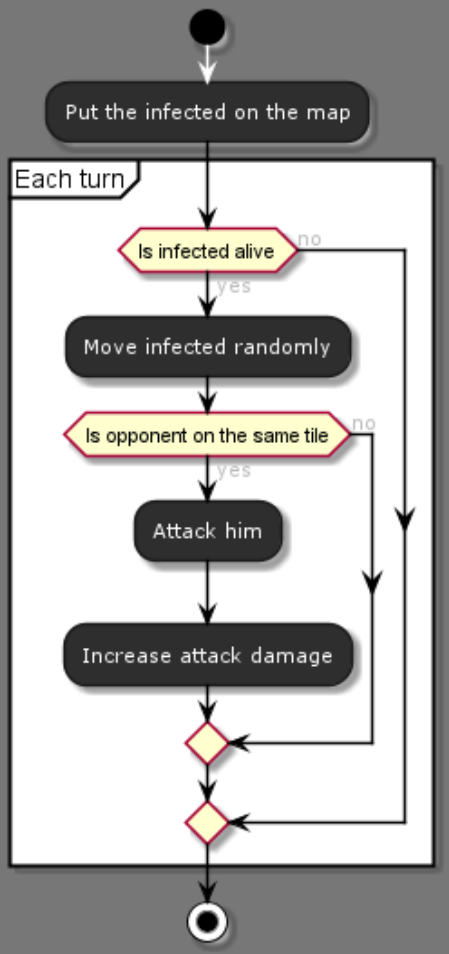
2.1 Diagram aktywności - symulacja



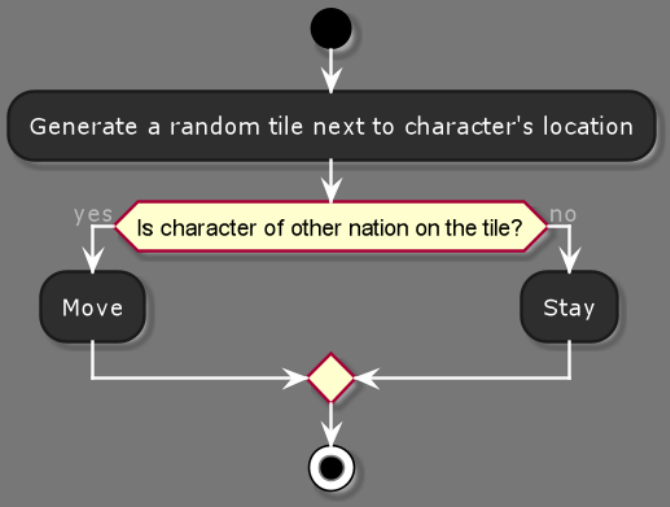
2.2 Diagram aktywności - człowiek



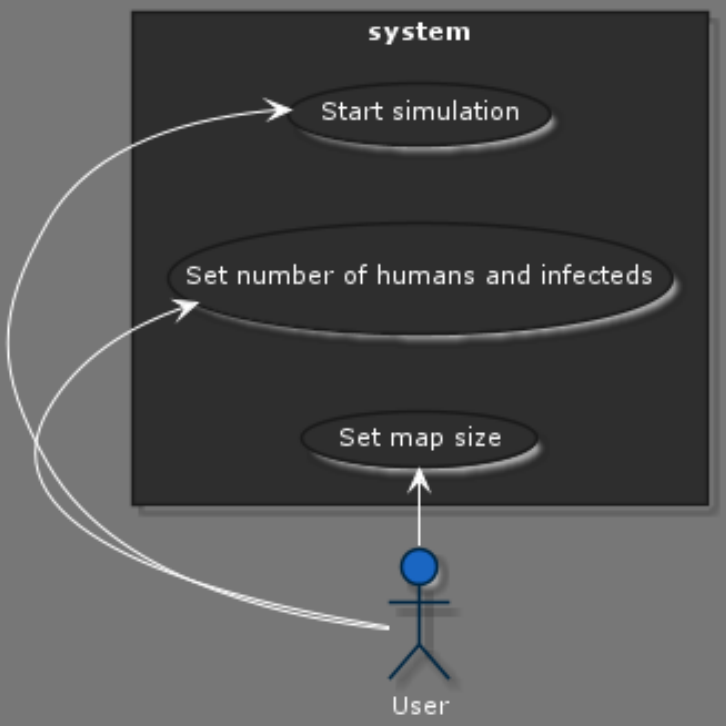
2.3 Diagram aktywności - zarażony



2.4 Diagram aktywności - ruch



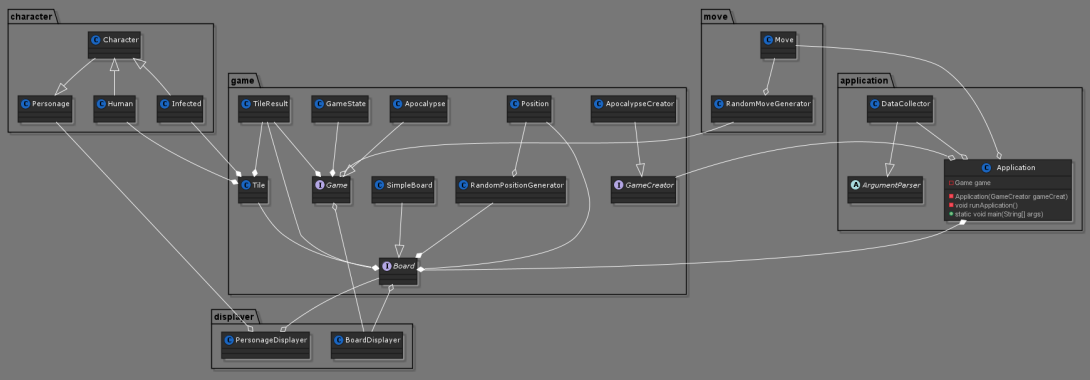
3 Diagram przypadków użycia



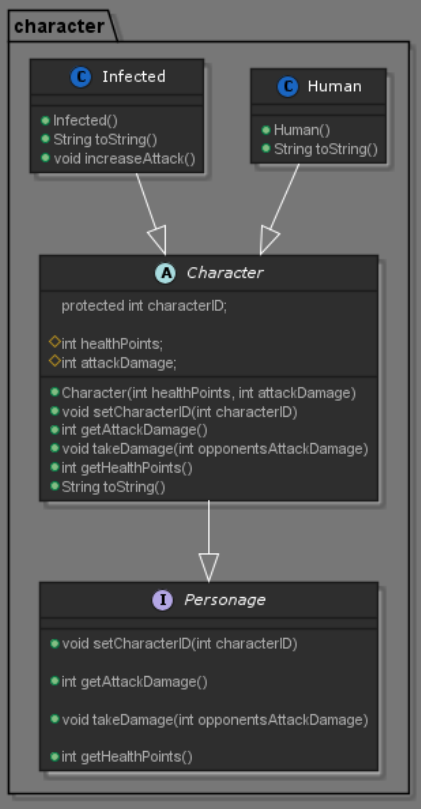
4 Diagramy klas

Diagramy klas bardziej rozszerzonych pakietów są dostępne w repozytorium w folderze diagramsUML w rozszerzeniach .png oraz .plantuml.

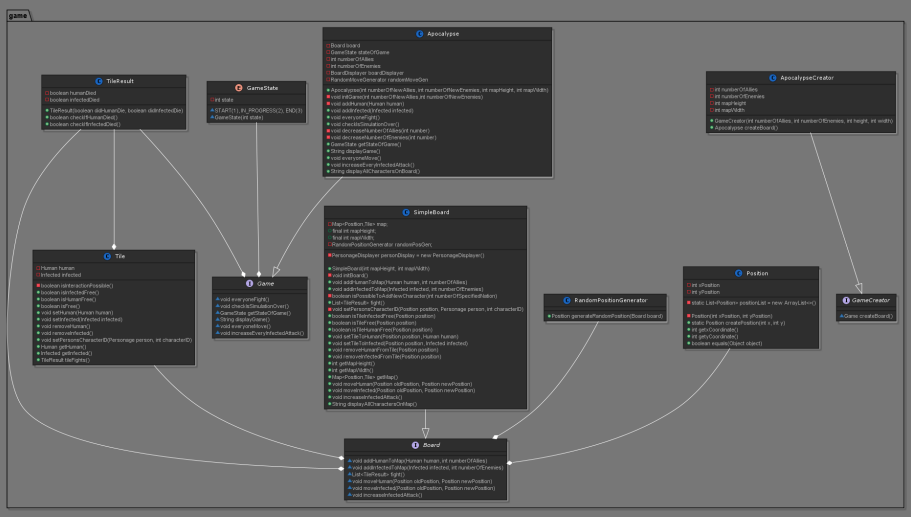
4.1 Diagram projektu



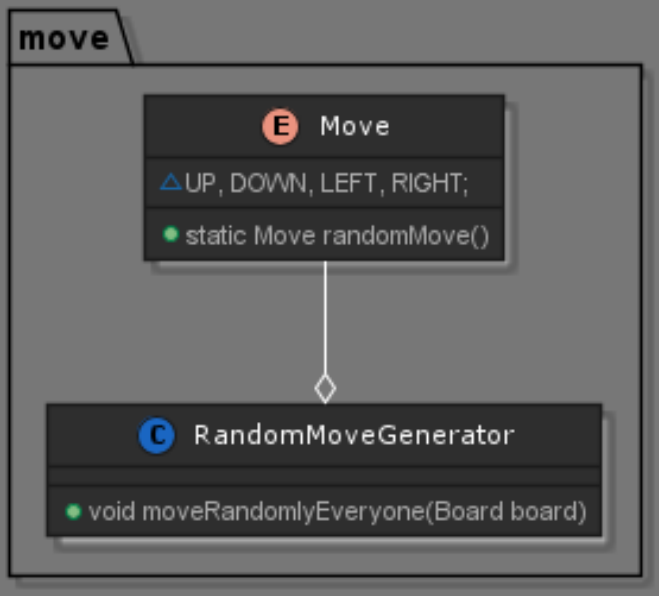
4.2 Diagram pakietu character



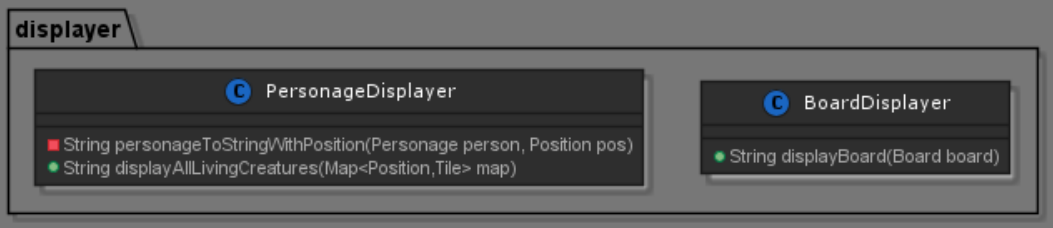
4.3 Diagram pakietu game



4.4 Diagram pakietu move



4.5 Diagram pakietu displayer



5 Karty CRC

Interface **Application**	
• main Class, using GameCreator to create a Game with specified number of allies, enemies, numberOfAllies and numberOfEnemies • runApplication() - doesn't stop working until one of the nations is gone, every round iterates through maps of allies and enemies and moves them around, calls for them to fight and displaysBoard and Characters, also every move every Infected's attack gets increased by 1	• BoardDisplayer • PersonageDisplayer • GameCreator • Game

ApocalypseCreator	Interface - GameCreator
• creates a Game with specified amounts of allies and enemies, checks if they are valid, numbers change in the future as it is possible to add a Infected and Human at any point while the application is running	• Apocalypse

Interface - Board **SimpleBoard**	
• Maps Position to Tiles, contains information about mapSize • initBoard- creates a Map with specified size • addHuman - adds one Human to a random free Tile, sets its ID to current number of allies • addInfected - adds one Infected to a random free Tile, sets its ID to current number of enemies • fight - iterates through every Tile and if it contains a Human and a Infected calls fightOnTile method from Tile, returns a List of TileResults to Game, in a way that it can then subtract amount of Creatures that died • moveInfected - getInfected from oldPosition Tile in Map and put him in newPosition Tile • moveHuman - getHuman from oldPosition Tile in Map and put him in newPosition Tile	• TIle • Human • Infected

Abstract **Move**	
• moveRandomlyEachCharacter - if not in combat, randomly chooses one nearby location to go to for each Character fromRandomMoveGenerator and if there is not already a different Human/Infected standing on it then it moves this Human/Infected to new Tile	• RanomMoveGenerator • Human • Infected • Tile • Position

Interface **RandomMoveGenerator**	
• Enum • randomMove - randomly chooses from one of these values: • UP, DOWN, RIGHT, LEFT	

Apocalypse	Interface - Game
• contains a Board, gathers information about current number of allies and enemies, current GameState • initGame() - adds specified amounts of Human and Infected to Board • addHuman - in case we would like to implement adding a Human in the middle of program runTime • addInfected - in case we would like to implement adding a Infected after the death of a Human • isSimulationOver - checks if numberOfAllies equals 0 or numberOfEnemies equals 0 • everyoneFight - calls method fight in board, counts amounts of Characters that died and extracts them from current numbers	• Board • Human • Infected • BoardDisplayer • RandomMoveGenerator • TileResult

Tile	
• Tile is an abstract object that can contain at most one Human and one Infected, it's not aware of its position • isInteractionPossible - boolean, checks if a Tile contains a Human and a Infected class objects other than null value • isInfectedFree - boolean, checks if the Tile does contain an Infected class object other than a null value • isHumanFree - boolean, checks if the Tile does contain a Human class object other than a null value • isFree - boolean, checks if the Tile does not contain a Human and an Infected class objects other than null values • setInfected, setHuman - set this Tile to contain a specified character • removeInfected, removeHuman - change Human/Infected on this Tile to 0 • fightOnTile - called by fight in Board, damages both characters and if they are dead removes them from the Tile and returns TileResult	• Human • Infected • TileResult

Abstract **BoardDisplayer**	
• displays the board, H stands for Human, I stands for Infected, * stands for both on a Tile and 0 stands for a free Tile (may be changed later)	• Board

Abstract **Character**	Interface - Personage Subclasses: Human, Infected
• Every character has an ID associated with the number of their nation objects on the map at the time of them being spawned on the Board, healthPoints and attackDamage	

Human	Superclasses: Character
• inherits from Character, different values in constructor	

Infected	Superclasses: Character
• inherits from Character, different values in a constructor • increaseAttack - every iteration the attack of Infected increases, they learn as the program is runing	

Position	
• Two Integers coordinate (x , y) class, there cannot be two objects of class Position with same coordinates • list of already declared Positions • setters and getters	

PersonageDisplayer	
• DisplaysPerson stats with its Position on board	• Board • Personage

GameState	
• Enum • stateOfTheGame(START, IN_PROGRESS, END)	

TileResult	
• two booleans, each for result • first if humanDiedDuringATurn • second if infectedDiedDuringATurn	