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

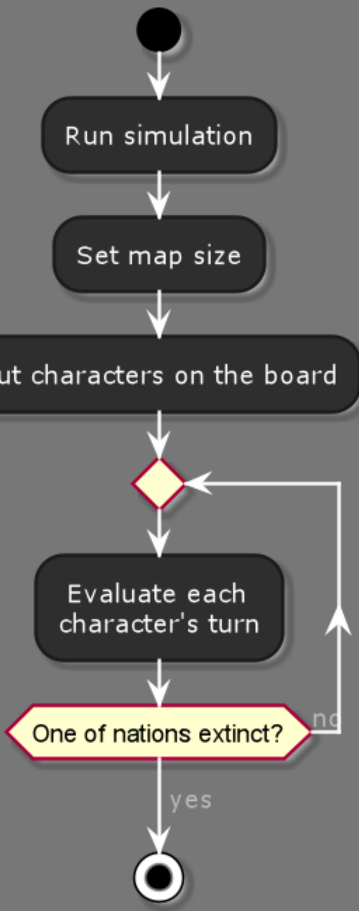
Spis treści

1 Analiza czasownikowo-rzeczownikowa

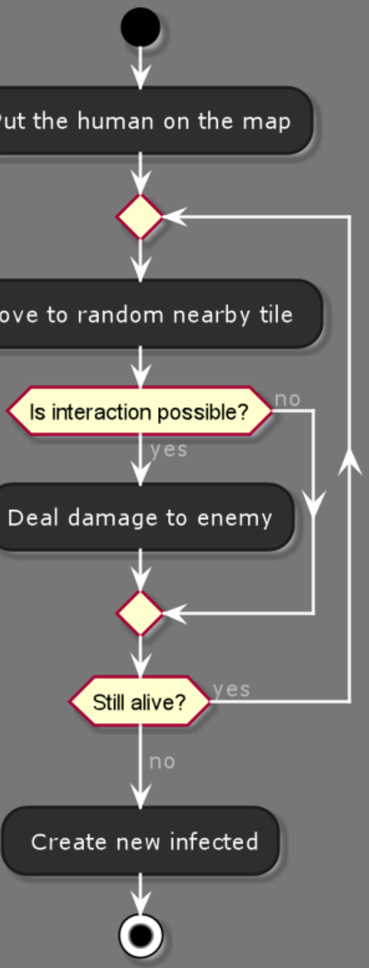
Projektujemy symulację agentową, w której stoczy się walka o przetrwanie ludzi lub zwycięstwo zarażonych. Postacie będą losowo rozmieszczane na dwuwymiarowej planszy o zadanej wielkości. Każda z klas postaci (ludzie, zarażeni) będzie dążyła do eliminacji przeciwnika. Użytkownik dostanie możliwość ustalenia 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

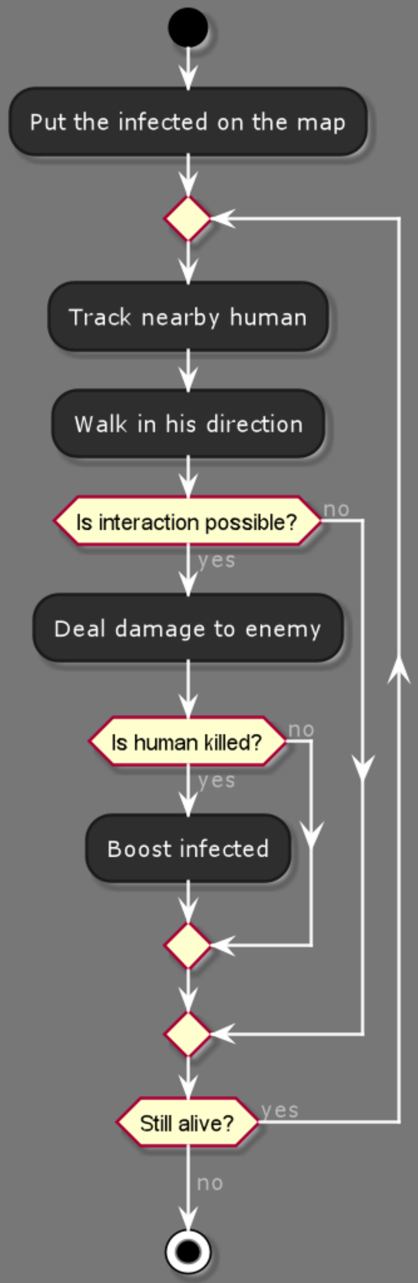
2.1 Diagram aktywności - symulacja



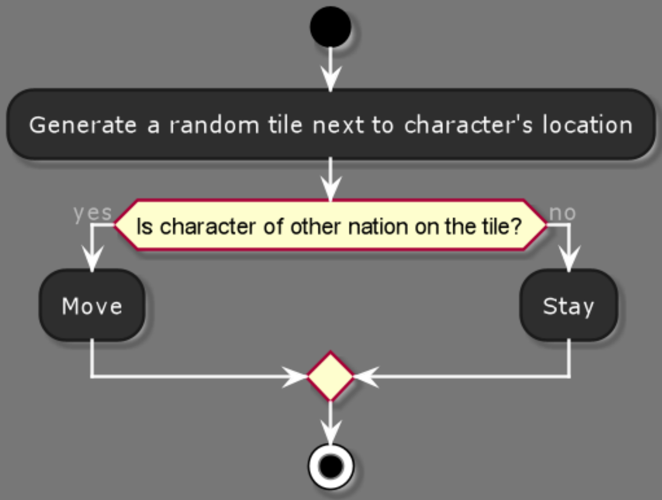
2.2 Diagram aktywności - człowiek



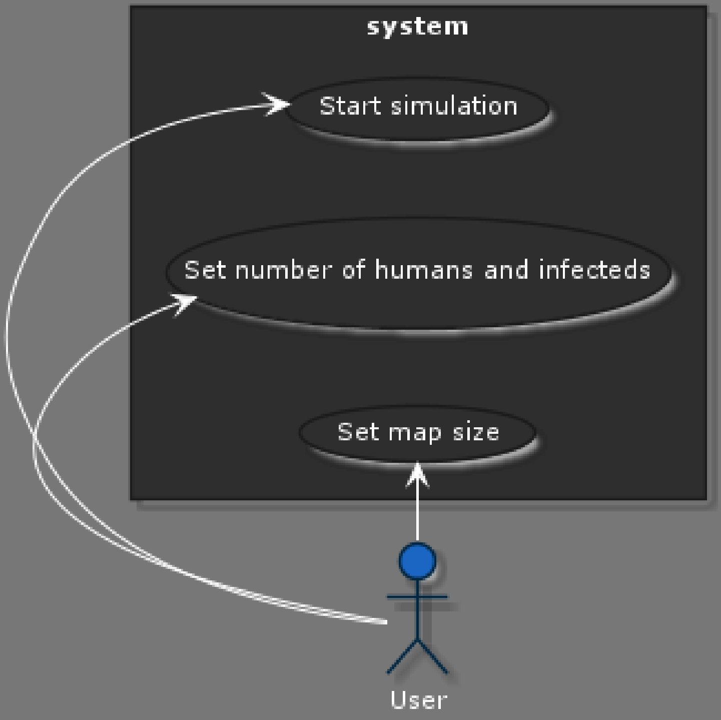
2.3 Diagram aktywności - zarażony



2.4 Diagram aktywności - ruch

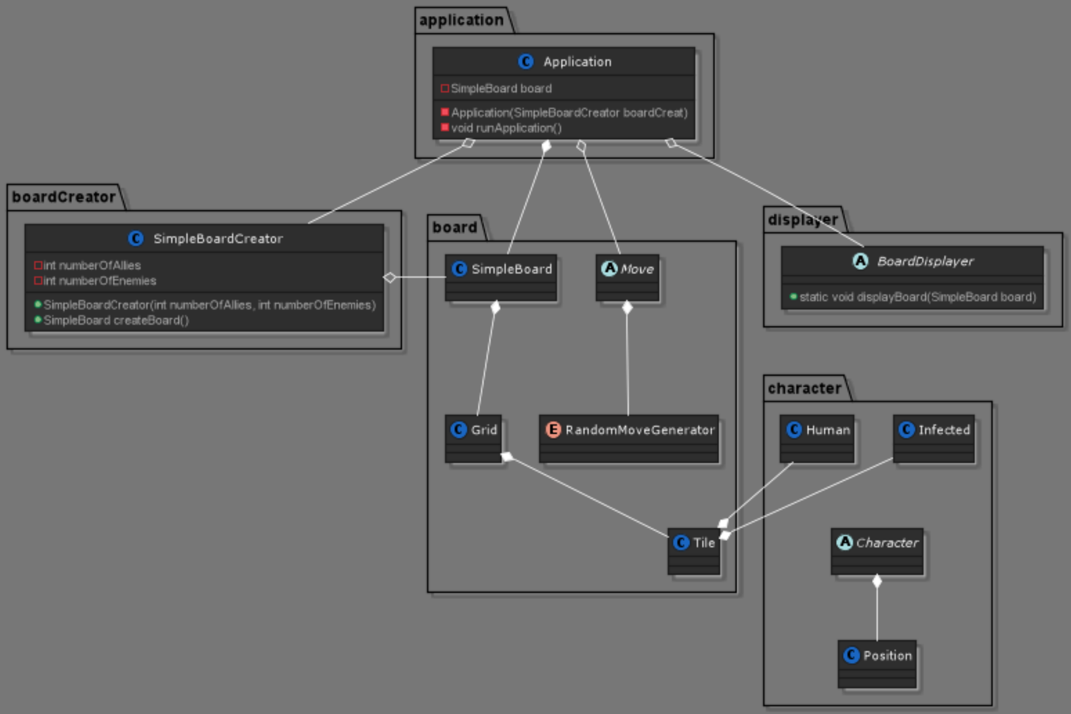

Generate a random tile next to character's location
Is character of other nation on the tile?
yes
no
Move
Stay

3 Diagram przypadków użycia

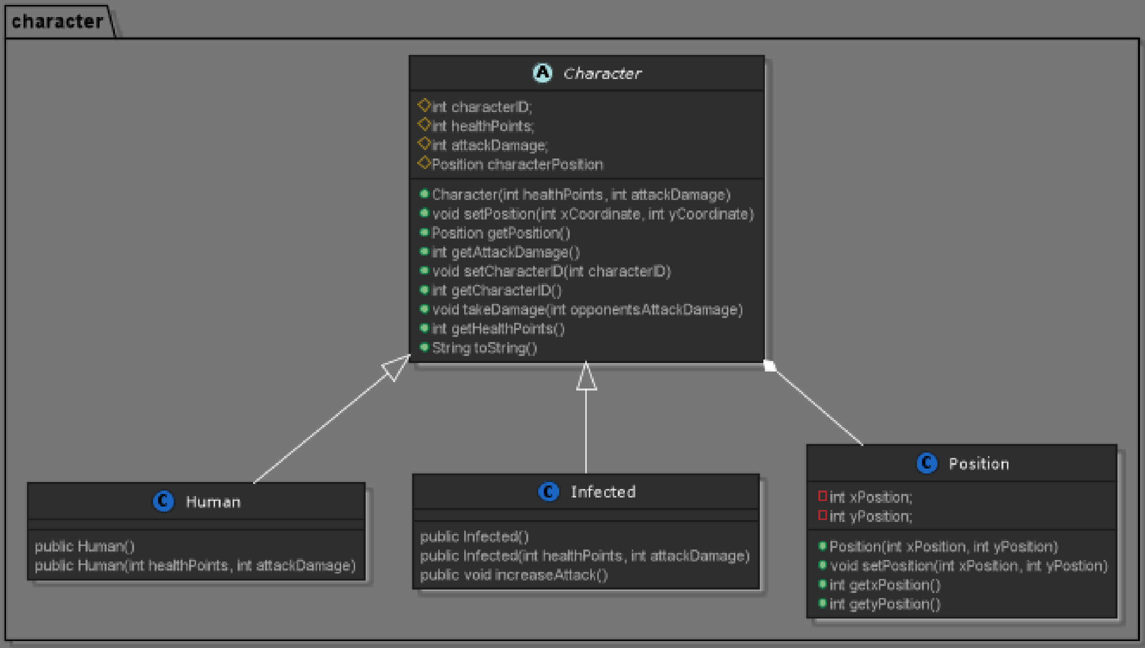


4 Diagramy klas

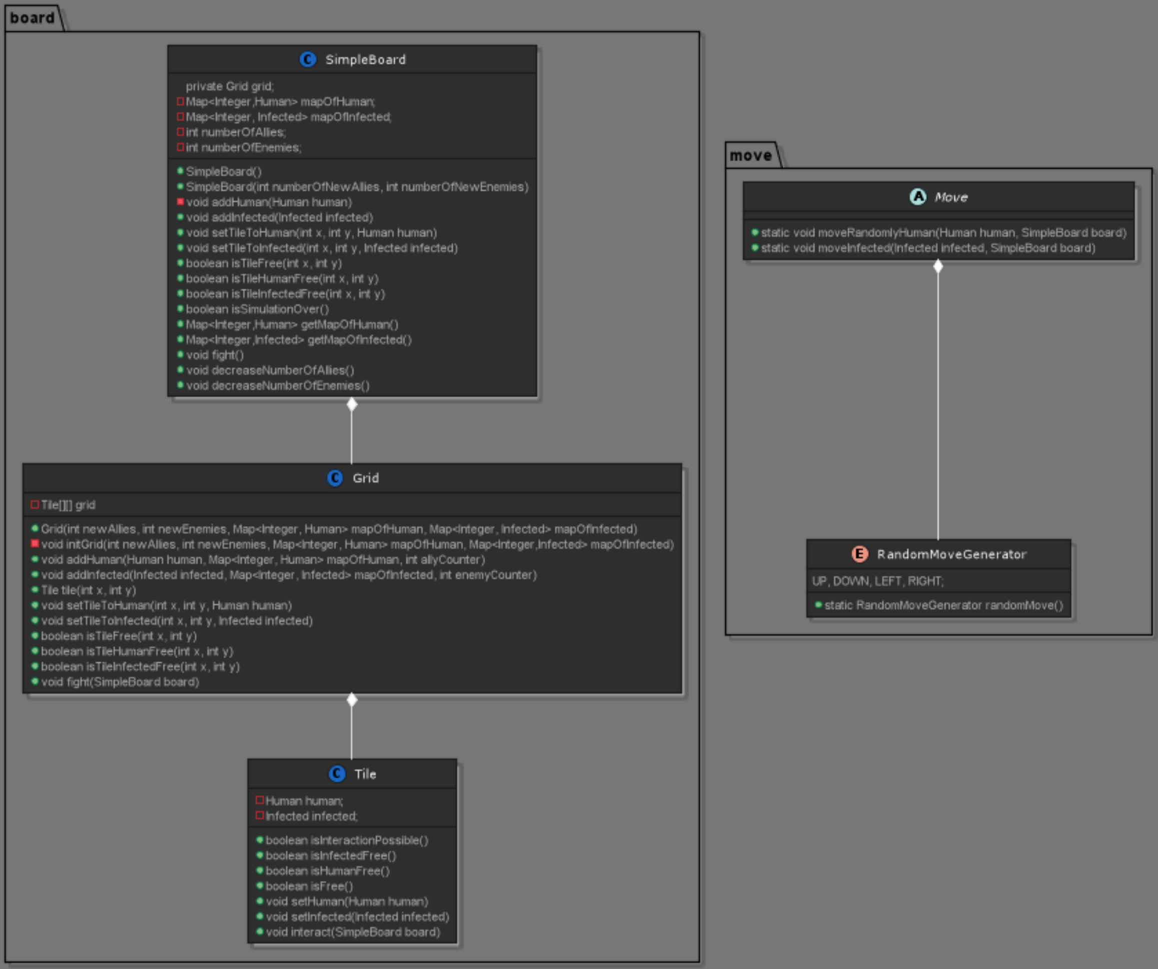
4.1 Diagram projektu



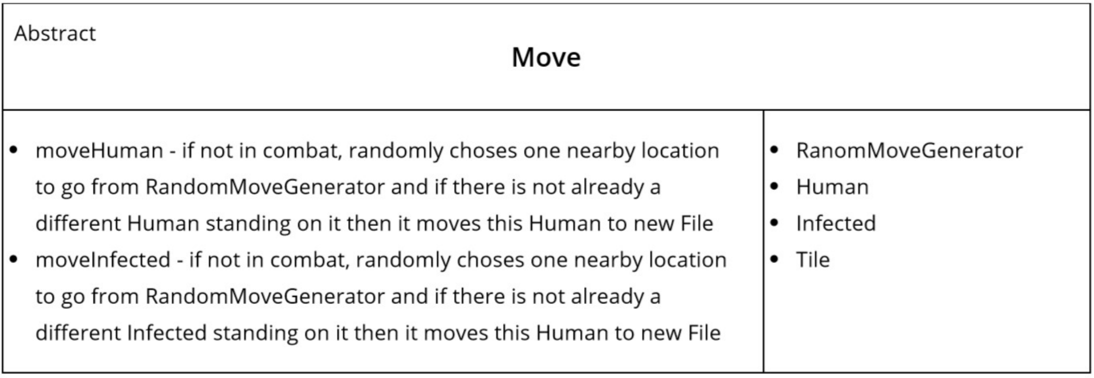
4.2 Diagram pakietu character


character
Character
int characterID;
int healthPoints;
int attackDamage;
Position characterPosition
Character(int healthPoints, int attackDamage)
void setPosition(int xCoordinate, int yCoordinate)
Position getPosition()
int getAttackDamage()
void setCharacterID(int characterID)
int getCharacterID()
void takeDamage(int opponentsAttackDamage)
int getHealthPoints()
String toString()
Human
public Human()
public Human(int healthPoints, int attackDamage)
Infected
public Infected()
public Infected(int healthPoints, int attackDamage)
public void increaseAttack()
Position
int xPosition;
int yPosition;
Position(int xPosition, int yPosition)
void setPosition(int xPosition, int yPosition)
int getxPosition()
int getyPosition()

4.3 Diagramy pakietów board i move


board
SimpleBoard
private Grid grid;
Map<Integer,Human> mapOfHuman;
Map<Integer, Infected> mapOfInfected;
int numberOfAllies;
int numberOfEnemies;
SimpleBoard()
SimpleBoard(int numberOfNewAllies, int numberOfNewEnemies)
void addHuman(Human human)
void addInfected(Infected infected)
void setTileToHuman(int x, int y, Human human)
void setTileToInfected(int x, int y, Infected infected)
boolean isTileFree(int x, int y)
boolean isTileHumanFree(int x, int y)
boolean isTileInfectedFree(int x, int y)
boolean isSimulationOver()
Map<Integer,Human> getMapOfHuman()
Map<Integer,Infected> getMapOfInfected()
void fight()
void decreaseNumberOfAllies()
void decreaseNumberOfEnemies()
Grid
Tile[][] grid
Grid(int newAllies, int newEnemies, Map<Integer, Human> mapOfHuman, Map<Integer, Infected> mapOfInfected)
void initGrid(int newAllies, int newEnemies, Map<Integer, Human> mapOfHuman, Map<Integer,Infected> mapOfInfected)
void addHuman(Human human, Map<Integer, Human> mapOfHuman, int allyCounter)
void addInfected(Infected infected, Map<Integer, Infected> mapOfInfected, int enemyCounter)
Tile tile(int x, int y)
void setTileToHuman(int x, int y, Human human)
void setTileToInfected(int x, int y, Infected infected)
boolean isTileFree(int x, int y)
boolean isTileHumanFree(int x, int y)
boolean isTileInfectedFree(int x, int y)
void fight(SimpleBoard board)
Tile
Human human;
Infected infected;
boolean isInteractionPossible()
boolean isInfectedFree()
boolean isHumanFree()
boolean isFree()
void setHuman(Human human)
void setInfected(Infected infected)
void interact(SimpleBoard board)
move
Move
static void moveRandomlyHuman(Human human, SimpleBoard board)
static void moveInfected(Infected infected, SimpleBoard board)
RandomMoveGenerator
UP, DOWN, LEFT, RIGHT;
static RandomMoveGenerator randomMove()

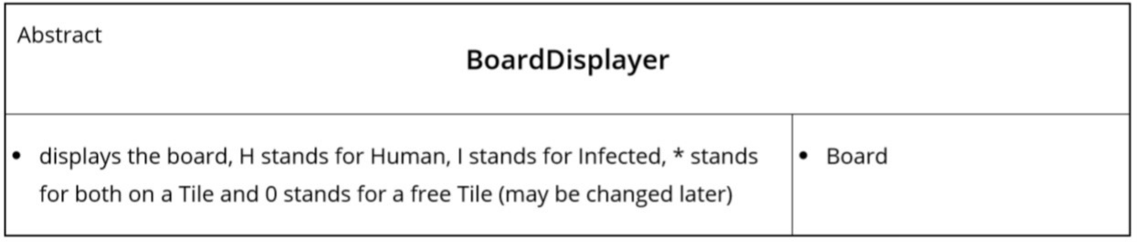
5 Karty CRC



Abstract

Move

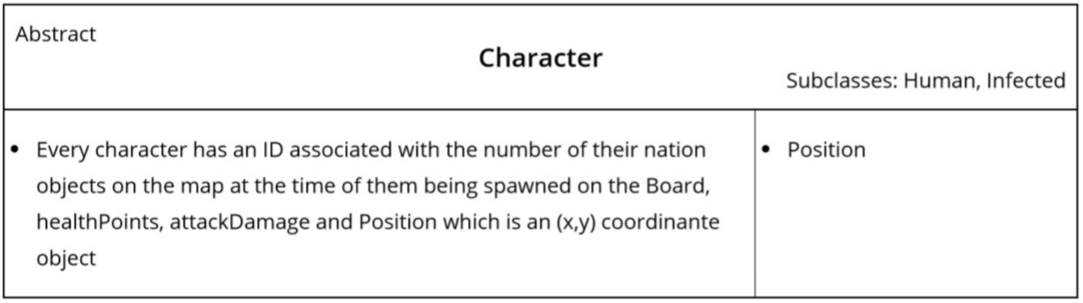
• moveHuman - if not in combat, randomly choses one nearby location to go from RandomMoveGenerator and if there is not already a different Human standing on it then it moves this Human to new File • moveInfected - if not in combat, randomly choses one nearby location to go from RandomMoveGenerator and if there is not already a different Infected standing on it then it moves this Human to new File	• RanomMoveGenerator • Human • Infected • Tile



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

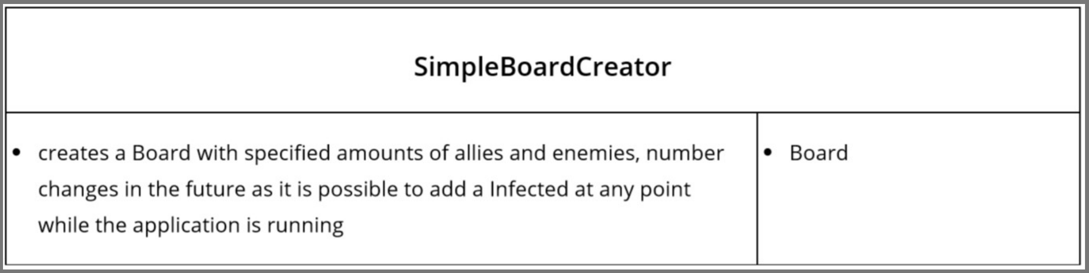


Abstract

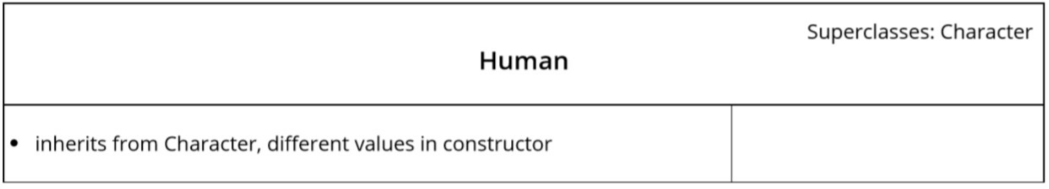
Character

Subclasses: Human, Infected

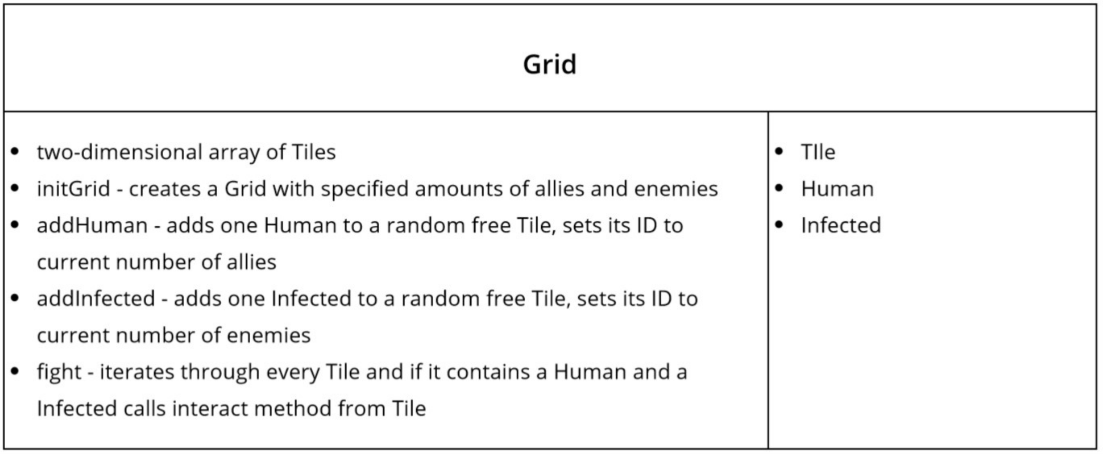
• Every character has an ID associated with the number of their nation objects on the map at the time of them being spawned on the Board, healthPoints, attackDamage and Position which is an (x,y) coordinante object	• Position



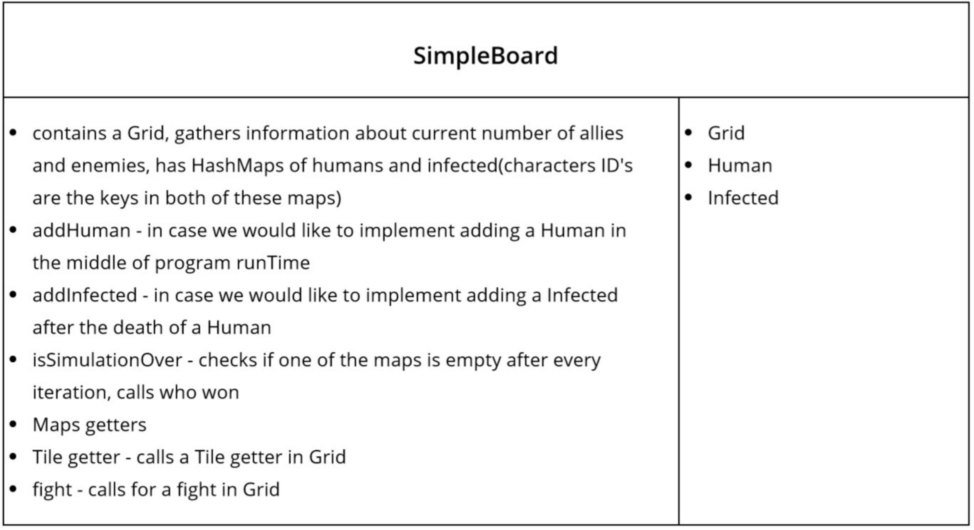
SimpleBoardCreator	
• creates a Board with specified amounts of allies and enemies, number changes in the future as it is possible to add a Infected at any point while the application is running	• Board



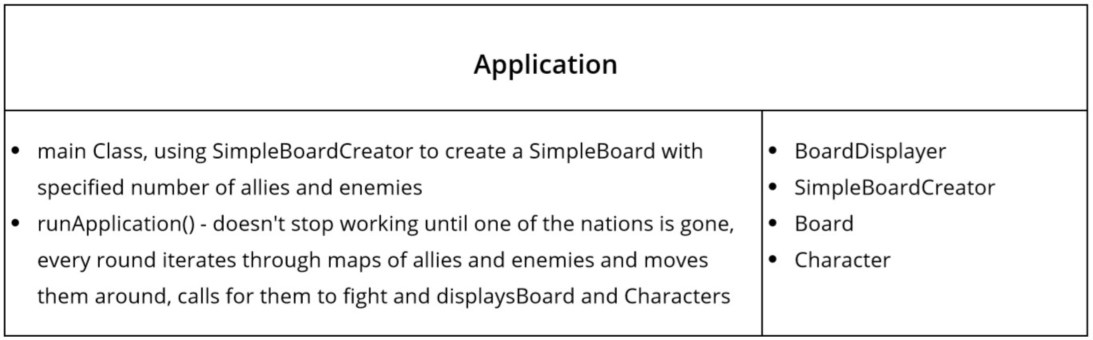
Human	Superclasses: Character
• inherits from Character, different values in constructor	



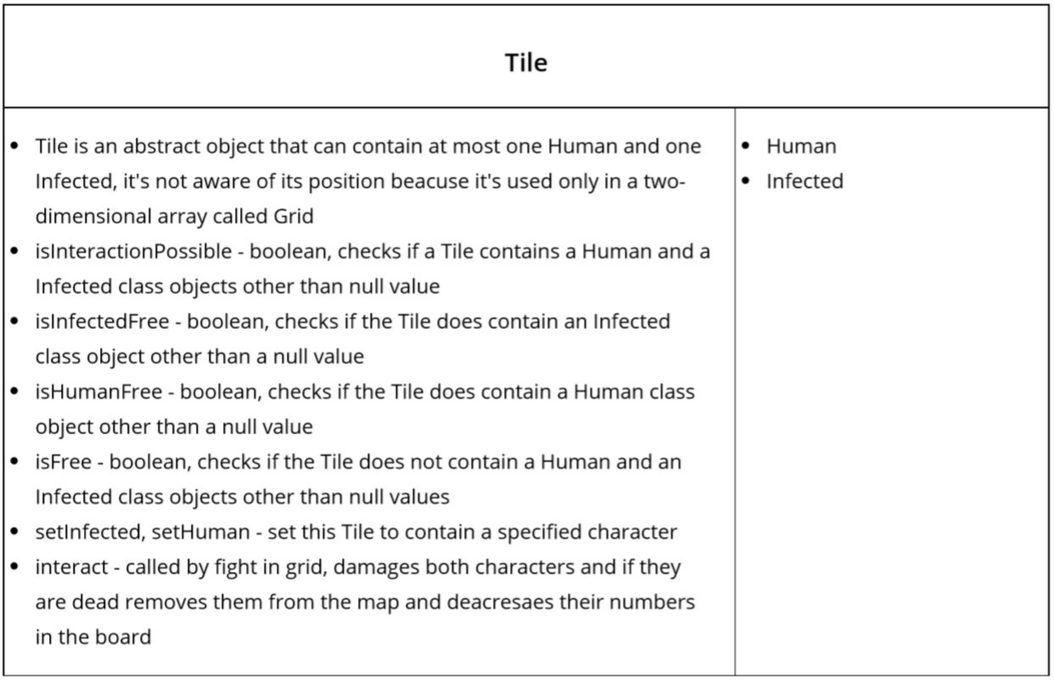
Grid	
• two-dimensional array of Tiles • initGrid - creates a Grid with specified amounts of allies and enemies • addHuman - adds one Human to a random free Tile, sets its ID to current number of allies • addInfected - adds one Infected to a random free Tile, sets its ID to current number of enemies • fight - iterates through every Tile and if it contains a Human and a Infected calls interact method from Tile	• TIle • Human • Infected



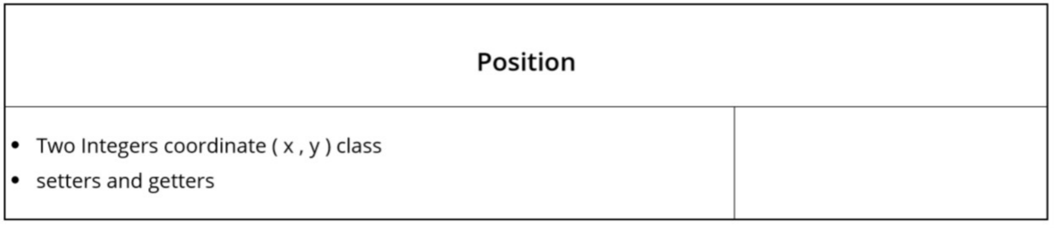
SimpleBoard	
• contains a Grid, gathers information about current number of allies and enemies, has HashMaps of humans and infected(characters ID's are the keys in both of these maps) • addHuman - in case we would like to implement adding a Human in the middle of program runTime • addInfected - in case we would like to implement adding a Infected after the death of a Human • isSimulationOver - checks if one of the maps is empty after every iteration, calls who won • Maps getters • Tile getter - calls a Tile getter in Grid • fight - calls for a fight in Grid	• Grid • Human • Infected



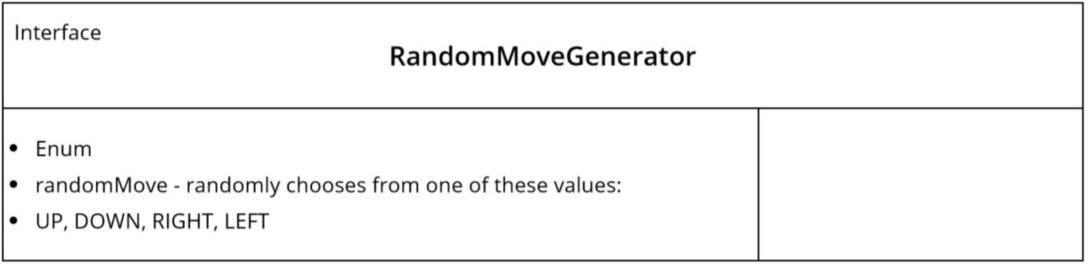
Application	
• main Class, using SimpleBoardCreator to create a SimpleBoard with specified number of allies and enemies • runApplication() - doesn't stop working until one of the nations is gone, every round iterates through maps of allies and enemies and moves them around, calls for them to fight and displaysBoard and Characters	• BoardDisplayer • SimpleBoardCreator • Board • Character



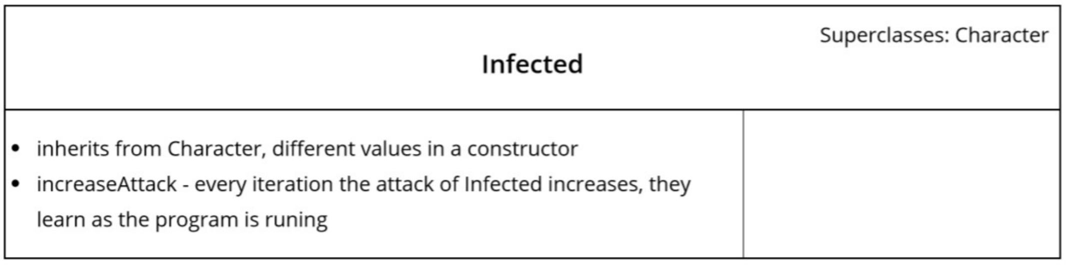
Tile	
• Tile is an abstract object that can contain at most one Human and one Infected, it's not aware of its position beacuse it's used only in a two-dimensional array called Grid • isInteractionPossible - boolean, checks if a Tile contains a Human and a Infected class objects other than null value • 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 • interact - called by fight in grid, damages both characters and if they are dead removes them from the map and deacresaes their numbers in the board	• Human • Infected



Position	
• Two Integers coordinate (x , y) class • setters and getters	



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



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	