Table Of Content

IStats 2
pl.projekt.game.item ..... 2 AbstractItem 3
Armor 5
ArmorTest 6
Axe 7
Dagger 8
Hammer 10
Jewelery 11
Maze 13
Shield 14
Sword 15
pl.projekt.game.material 17
AbstractMaterials 17
Diamond 18
Iron 19
Stone 20
Wood 21
pl.projekt.game.mob 23
AbstractMonster 23
Dwarf 26
Elf 28
Minotaur 30
MinotaurTest 33
Orc 33
pl.projekt.simulation 36
Board 36
IRandom 38
SimulationApp 39
Index 41

Interface IStats

< Methods >

public interface **IStats**

Interfejs po którym dziedziczą moby oraz przedmioty.

Methods

addArmour

```
public void addArmour(double armour)
```

---

addAttack

```
public void addAttack(double attack)
```

---

addHP

```
public void addHP(double hp)
```

Package pl.projekt.game.item

Class Summary

AbstractItem

Klasa abstrakcyjna po której dziedziczą wszystie przedmioty.

Armor

Przedmiot dostępny dla każdego moba.Zwiększa znaczaco ilość punktów defensywy oraz nieznacznie zdrowie.

ArmorTest

Axe

Przedmiot dostępny tylko dla Orka.Zwiększa znacznie atak,ale zmniejsza nieznacznie defensywę.

Dagger

Przedmiot dostęny dla wszystkich.Nieznacznie zwiększa atak.

Hammer

Przedmiot dostępny wyłącznie dla Krasnoluda.

Jewelery

Przedmiot dostepny dla wszystkich mobów.Nieznacznie zwiększa wszystkie statystyki.

Maze

Przedmiot dostępny wyłącznie dla Minotaura.Zwiększa on bardzo znacznie atak,za to zmiejsza nieznacznie hp oraz defensywę.

Shield

Przedmiot dostęny dla wszystkich.Zwiększa nieznacznie zarówno hp,jak i defensywę.

Sword

Przedmiot dostępny wyłącznie dla Elfa.

---

pl.projekt.game.item

Class AbstractItem

```
java.lang.Object
    |
    +--pl.projekt.game.item.AbstractItem
```

All Implemented Interfaces:

IStats

Direct Known Subclasses:

Armor, Axe, Dagger, Hammer, Jewelery, Maze, Shield, Sword

< Constructors > < Methods >

---

public abstract class **AbstractItem**
extends java.lang.Object
implements IStats

Klasa abstrakcyjna po której dziedziczą wszystkie przedmioty.

Constructors

AbstractItem

```
public AbstractItem()
```

Methods

getArmourPoints

public int **getArmourPoints**()

---

getDamagePoints

public int **getDamagePoints**()

---

getHpPoints

public int **getHpPoints**()

---

getMob1

public AbstractMonster **getMob1**()

---

setArmourPoints

public void **setArmourPoints**(int armourPoints)

---

setDamagePoints

public void **setDamagePoints**(int damagePoints)

---

setHpPoints

public void **setHpPoints**(int hpPoints)

---

setMob1

```
public void setMob1(AbstractMonster mob1)
```

---

pl.projekt.game.item

Class Armor

```
java.lang.Object
    |
    +--AbstractItem
        |
        +--pl.projekt.game.item.Armor
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Armor**
extends AbstractItem

Przedmiot dostępny dla każdego moba.Zwieksza znaczaco ilość punktów defensywy oraz nieznacznie zdrowie.

Constructors

Armor

```
public Armor(double damagePoints,
             double       armourPoints,
             double hpPoints,
             AbstractMonster mob1)
```

Armor

```
public Armor(AbstractMonster mob1)
```

Methods

addArmour

```
public void addArmour(double armour)
```

metoda dodaje 2% aktualnych punktów armour moba,punkty armour Itemu oraz parametr armour do punktów armour moba który posiada Item **Parameters:**

armour - przyjmuje double

---

addAttack

```
public void addAttack(double attack)
```

---

addHP

```
public void addHP(double hp)
```

metoda dodaje 5% aktualnych punktów hp moba,punkty hp Itemu oraz parametr hp do punktów hp moba który posiada Item **Parameters:**

hp - przyjmuje double

---

pl.projekt.game.item

Class ArmorTest

```
java.lang.Object
    |
    +--pl.projekt.game.item.ArmorTest
```

< Constructors > < Methods >

---

public class **ArmorTest**
extends java.lang.Object

Constructors

ArmorTest

```
public ArmorTest()
```

Methods

addArmour

```
public void addArmour()
```

---

addAttack

```
public void addAttack()
```

---

addHP

```
public void addHP()
```

---

pl.projekt.game.item

Class Axe

```
java.lang.Object
    |
    +--AbstractItem
        |
        +--pl.projekt.game.item.Axe
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Axe**
extends AbstractItem

Przedmiot dostępny tylko dla Orka.Zwiększa znacznie atak,ale zmniejsza nieznacznie defensywę.

Constructors

Axe

```
public  Axe(double damagePoints,
            double armourPoints,
            double hpPoints,
            AbstractMonster
            mob1)
```

Axe

```
public  Axe(AbstractMonster mob1)
```

metoda odejmuje 3% aktualnych punktów armour moba,punkty armour Itemu oraz parametr armour do punktów armour moba który posiada Item **Parameters:**

armour - przyjmuje double

Methods

addArmour

```
public void addArmour(double armour)
```

---

addAttack

```
public void addAttack(double attack)
```

metoda dodaje 7% aktualnych punktów ataku moba,punkty Damage Itemu oraz parametr attack do punktów ataku moba który posiada Item **Parameters:**

attack - przyjmuje double

---

addHP

```
public void addHP(double hp)
```

---

pl.projekt.game.item

Class Dagger

```
java.lang.Object
    |
    +--AbstractItem
```

```
        |
        +--pl.projekt.game.item.Dagger
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Dagger**
extends AbstractItem

Przedmiot dostępny dla wszystkich.Nieznacznie zwiększa atak.

Constructors

Dagger

```
public Dagger(double damagePoints,
              double       armourPoints,
              double hpPoints,
              AbstractMonster mob1)
```

Dagger

```
public Dagger(AbstractMonster mob1)
```

Methods

addArmour

```
public void addArmour(double armour)
```

---

addAttack

```
public void addAttack(double attack)
```

metoda dodaje 3% aktualnych punktów ataku moba,punkty Damage Itemu oraz parametr attack do punktów Ataku moba który posiada Item **Parameters:**

attack - przyjmuje double

addHP

```
public void addHP(double hp)
```

---

pl.projekt.game.item

Class Hammer

```
java.lang.Object
    |
    +--AbstractItem
        |
        +--pl.projekt.game.item.Hammer
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Hammer**
extends AbstractItem

Przedmiot dostępny wyłącznie dla Krasnoluda. Zwiększa nieznacznie atak oraz życie.

Constructors

Hammer

```
public Hammer(double damagePoints,
              double       armourPoints,
              double hpPoints,
              AbstractMonster mob1)
```

Hammer

```
public Hammer(AbstractMonster mob1)
```

Methods

addArmour

`public void **addArmour**(double armour)`

metoda dodaje punkty armour Itemu oraz parametr armour do punktów armour moba który posiada Item

Parameters:

armour - przyjmuje double

---

addAttack

`public void **addAttack**(double attack)`

metoda dodaje 4% aktualnych punktów ataku moba,punkty ataku Itemu oraz parametr attack do punktów Ataku moba który posiada Item **Parameters:**

attack - przyjmuje double

---

addHP

`public void **addHP**(double hp)`

metoda dodaje 2% aktualnych punktów hp moba oraz parametr hp do punktów hp moba który posiada Item

Parameters:

hp - przyjmuje double

---

pl.projekt.game.item

Class Jewelery

```
java.lang.Object
    |
    +--AbstractItem
        |
        +--pl.projekt.game.item.Jewelery
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

public class **Jewelery**
extends AbstractItem

Przedmiot dostępny dla wszystkich mobów.Nieznacznie zwiększa wszystkie statystyki.

Constructors

Jewelery

```
public Jewelery(double damagePoints,
                double armourPoints,
                double hpPoints,
                AbstractMonster mob1)
```

Jewelery

```
public Jewelery(AbstractMonster mob1)
```

Methods

addArmour

```
public void addArmour(double armour)
```

metoda dodaje 2% aktualnych punktów armour moba,punkty armour Itemu oraz parametr armour do punktów armour moba który posiada Item **Parameters:**

armour - przyjmuje double

---

addAttack

```
public void addAttack(double attack)
```

metoda dodaje 2% aktualnych punktów ataku moba,punkty ataku Itemu oraz parametr attack do punktów ataku moba który posiada Item **Parameters:**

attack - przyjmuje double

---

addHP

```
public void addHP(double hp)
```

metoda dodaje 2% aktualnych punktów hp moba,punkty hp Itemu oraz parametr hp do punktów hp moba który posiada Item **Parameters:**

hp - przyjmuje double

---

pl.projekt.game.item

Class Maze

```
java.lang.Object
    |
    +--AbstractItem
        |
        +--pl.projekt.game.item.Maze
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Maze**
extends AbstractItem

Przedmiot dostępny wyłącznie dla Minotaura.Zwiększa on bardzo znacznie atak,za to zmiajsza nieznacznie hp oraz defensywę.

Constructors

Maze

```
public Maze(double damagePoints,
            double       armourPoints,
            double hpPoints,
            AbstractMonster mob1)
```

Maze

```
public Maze(AbstractMonster mob1)
```

Methods

addArmour

```
public void addArmour(double armour)
```

metoda odejmuje 3% aktualnych punktów armour moba,punkty armour Itemu oraz parametr armour do punktów armour moba który posiada Item **Parameters:**

armour - przyjmuje armour

---

addAttack

```
public void addAttack(double attack)
```

metoda dodaje 10% aktualnych punktów ataku moba,punkty ataku Itemu oraz parametr attack do punktów ataku moba który posiada Item **Parameters:**

attack - przyjmuje double

---

addHP

```
public void addHP(double hp)
```

metoda odejmuje 3% aktualnych punktów hp moba,punkty hp Itemu oraz parametr hp do punktów hp moba który posiada Item **Parameters:**

hp - przyjmuje double

---

pl.projekt.game.item

Class Shield

```
java.lang.Object
    |
    +--AbstractItem
        |
        +--pl.projekt.game.item.Shield
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Shield**
extends AbstractItem

Przedmiot dostępny dla wszystkich.Zwiększa nieznacznie zarówno hp,jak i defensywę.

Constructors

Shield

```
public Shield(double armourPoints,
              double damagePoints,
              double hpPoints,
              AbstractMonster mob1)
```

Shield

```
public Shield(AbstractMonster mob1)
```

Methods

addArmour

```
public void addArmour(double armour)
```

metoda dodaje 4% aktualnych punktów armour moba,punkty armour Itemu oraz parametr armour do punktów armour moba który posiada Item **Parameters:**

armour - przyjmuje double

---

addAttack

```
public void addAttack(double attack)
```

---

addHP

```
public void addHP(double hp)
```

metoda dodaje 2% aktualnych punktów hp moba,punkty hp Itemu oraz parametr hp do punktów hp moba który posiada Item **Parameters:**

hp - przyjmuje double

---

pl.projekt.game.item

Class Sword

```
java.lang.Object
    |
    +--AbstractItem
        |
        +--pl.projekt.game.item.Sword
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Sword**
extends AbstractItem

Przedmiot dostępny wyłącznie dla Elfa. Zwiększa znacznie atak moba.

Constructors

Sword

```
public Sword(double armourPoints,
             double       damagePoints,
             double hpPoints,
             AbstractMonster mob1)
```

Sword

```
public Sword(AbstractMonster mob1)
```

Methods

addArmour

```
public void addArmour(double armour)
```

---

addAttack

```
public void addAttack(double attack)
```

metoda dodaje 5% aktualnych punktów ataku moba,punkty ataku Itemu oraz parametr attack do punktów ataku moba który posiada Item **Parameters:**

attack - przyjmuje double

---

addHP

```
public void addHP(double hp)
```

Package pl.projekt.game.material

Class Summary

AbstractMaterials

Klasa abstrakcyjna po której dziedziczą wszystkie materiały.

Diamond

Materiał, dostępny dla każdego moba.

Iron

Materiał który może zostać zebrany przez Elfa oraz Minotaura.

Stone

Materiał który może być zebrany przez Krasnoluda,Minotura oraz Orka.

Wood

Materiał, dostępny dla każdego moba.Można dzięki niemu stworzył miemal każdy przedmiot w symulacji.

Class AbstractMaterials

```
java.lang.Object
    |
    +--pl.projekt.game.material.AbstractMaterials
```

Direct Known Subclasses:

Diamond, Iron, Stone, Wood

< Constructors > < Methods >

---

public abstract class **AbstractMaterials**
extends java.lang.Object

Klasa abstrakcyjna po której dziedziczą wszystkie materiały. Znajduje się tutaj metoda która sprawdza czy materiał nie jest za ciężki dla moba.

Constructors

AbstractMaterials

```
public AbstractMaterials()
```

Methods

getWeight

```
public abstract int getWeight()
```

---

isNotToHeavy

```
public boolean isNotToHeavy(int weight, int
                            maxWeight)
```

Metoda sprawdza czy w ekwipunku mob nie ma za dużo przedmiotow 1 rodzaju

Parameters:

weight - waga przedmiotu
maxWeight - maksymalna waga która mob moze przenieść

Returns:

zwraca prawdę lub fałsz w zależnosi czy mob moze podniesc wiecej czy nie

pl.projekt.game.material

Class Diamond

```
java.lang.Object
    |
    +--AbstractMaterials
        |
        +--pl.projekt.game.material.Diamond
```

< Constructors > < Methods >

public class **Diamond**
extends AbstractMaterials

Materiał dostępny dla każdego moba. Potrzebny do wytworzenia Biżuterii.

Constructors

Diamond

```
public Diamond()
```

Diamond

```
public Diamond(int weight)
```

Methods

getWeight

```
public int getWeight()
```

Overrides:

getWeight in class AbstractMaterials

pl.projekt.game.material

Class Iron

```
java.lang.Object
    |
    +--AbstractMaterials
        |
        +--pl.projekt.game.material.Iron
```

< Constructors > < Methods >

---

public class **Iron** extends
AbstractMaterials

Materiał który może zostać zebrany przez Elfa oraz Minotaura. Służy do stworzenia Buzdyganu oraz Miecza.

Constructors

Iron

```
public Iron()
```

Iron

```
public Iron(int weight)
```

Methods

getWeight

```
public int getWeight()
```

Overrides:

getWeight in class AbstractMaterials

---

pl.projekt.game.material

Class Stone

```
java.lang.Object
```

```
    |
    +--AbstractMaterials
        |
        +--pl.projekt.game.material.Stone
```

< Constructors > < Methods >

---

public class **Stone**
extends AbstractMaterials

Materiał który może być zebrany przez Krasnoluda,Minotura oraz Orka. Służy do stworzenia Młota,Buzdyganu oraz Topora.

Constructors

Stone

```
public Stone()
```

Stone

```
public Stone(int weight)
```

Methods

getWeight

```
public int getWeight()
```

Overrides:

getWeight in class AbstractMaterials

---

pl.projekt.game.material

Class Wood

```
java.lang.Object
```

```
    |
    +--AbstractMaterials
        |
        +--pl.projekt.game.material.Wood
```

< Constructors > < Methods >

---

public class **Wood**
extends AbstractMaterials

Materiał dostępny dla każdego moba.Można dzięki niemu stworzył niemal każdy przedmiot w symulacji.

Constructors

Wood

```
public Wood()
```

Wood

```
public Wood(int weight)
```

Methods

getWeight

```
public int getWeight()
```

Overrides:

getWeight in class AbstractMaterials

Package pl.projekt.game.mob

Class Summary

AbstractMonster

Klasa abstrakcyjna po której dziedziczy każdy mob.Dzięki niej potrafią tworzyyć sztylet,zbroje czy tarczę oraz zbierać diamenty i drewno.Znajdują się w niej metody na zbieranie materiałów,tworzenie przedmiotów oraz walkę i łączenie się mobów.

Dwarf

Krasnolud jest jednym z 4 ras pojawiających się w symulacji.

Elf

Elf jest jednym z 4 ras pojawiających się w symulacji.

Minotaur

Minotaur jest jednym z 4 ras pojawiających się w symulacji.

MinotaurTest

Orc

Ork jest jednym z 4 ras pojawiających się w symulacji.

---

pl.projekt.game.mob

Class AbstractMonster

```
java.lang.Object
    |
    +--pl.projekt.game.mob.AbstractMonster
```

All Implemented Interfaces:

IStats

Direct Known Subclasses:

Dwarf, Elf, Minotaur, Orc

< Constructors > < Methods >

---

public abstract class **AbstractMonster**
extends java.lang.Object
implements IStats

Klasa abstrakcyjna po której dziedziczy każdy mob.Dzięki niej potrafi tworzyć sztylet,zbroje czy tarczę oraz zbierał diamenty i drewno.Znajdują się w niej metody na zbieranie materiałów,tworzenie przedmiotów oraz walkę i łączenie się mobów.

Constructors

AbstractMonster

```
public AbstractMonster()
```

Methods

addArmour

```
public void addArmour(double Armr)
```

---

addAttack

```
public void addAttack(double Attack)
```

---

addHP

```
public void addHP(double HP)
```

---

collectMaterial

```
public void collectMaterial(AbstractMonster monster1,
                            AbstractMaterials materials)
```

Metoda sprawdzwa czy materiał może zostać zebrany przez moba oraz zmiejsza posiadane przez moba wolne miejsce na materiały **Parameters:**

monster1 - mob który zbiera materiał
materials - zbierany materiał

---

craftNewItem

```
public void craftNewItem(AbstractMonster monster)
```

metoda w której mob wybiera jaki item ma stworzył

Parameters:

monster - mob który tworzy item

---

fight

```
public AbstractMonster fight(AbstractMonster firstM,
                             AbstractMonster secondM)
```

Metoda wykonuje walke 2 mobow róznych klas a nastepnie zwraca wygranego

Parameters:

firstM - 1 mob
secondM - 2 mob

Returns:

zwraca wygranego moba

---

getAttack

```
public double getAttack()
```

---

getDefence

```
public double getDefence()
```

---

getEquipment

```
public java.util.ArrayList getEquipment()
```

---

getHealth

```
public double getHealth()
```

getWoodnmb

```
public int getWoodnmb()
```

merge

```
public AbstractMonster merge(AbstractMonster
          monster1, AbstractMonster monster2)
```

Metoda która w parametrach przyjmuje 2 moby tej samej klasy a następnie łaczy je w 1 nowego moba o sumie parametrow **Parameters:**

monster1 - 1 mob
monster2 - 2 mob **Returns:**
zwraca nowego moba

---

pl.projekt.game.mob

Class Dwarf

```
java.lang.Object
    |
    +--AbstractMonster
        |
        +--pl.projekt.game.mob.Dwarf
```

All Implemented Interfaces:
IStats

< Constructors > < Methods >

---

public class **Dwarf**
extends AbstractMonster

Krasnolud jest jednym z 4 ras pojawiających się w symulacji. Charakteryzuje się wysoką obroną,średnim życiem i niskim atakiem. Posiada możliwoscią tworzenia Młota.

Constructors

Dwarf

```
public Dwarf()
```

Dwarf

```
public  Dwarf(double  healthPoints,
              double defencePoints,
              double attacPoints)
```

Methods

addArmour

```
public void addArmour(double armr)
```

Overrides:

addArmour in class AbstractMonster

---

addAttack

```
public void addAttack(double attack)
```

Overrides:

addAttack in class AbstractMonster

---

addHP

```
public void addHP(double HP)
```

Overrides:

addHP in class AbstractMonster

---

collectStone

```
public void collectStone()
```

metoda zwiększa ilość posiadanego przez moba kamienia o 1

---

createHammer

```
public void createHammer()
```

metoda tworzy oraz dodaje do Ekwipunku moba jedną sztukę Młota oraz odejmuje od ilości kawałków drewna oraz kamienia cenę za stworzenie Młota(odpowiednio 2 i 1)

---

getAttack

`public double` **`getAttack`**`()`

Overrides:

getAttack in class AbstractMonster

---

getDefence

`public double` **`getDefence`**`()`

Overrides:

getDefence in class AbstractMonster

---

getHealth

`public double` **`getHealth`**`()`

Overrides:

getHealth in class AbstractMonster

---

pl.projekt.game.mob

Class Elf

```
java.lang.Object
    |
    +--AbstractMonster
        |
        +--pl.projekt.game.mob.Elf
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Elf** extends
AbstractMonster

Elf jest jednym z 4 ras pojawiających się w symulacji. Charakteryzuje się standardowymi statystykami oraz możliwoscią stworzenia Miecza.

Constructors

Elf

```
public Elf()
```

Elf

```
public  Elf(double  healthPoints,
            double defencePoints,
            double attacPoints)
```

Methods

addArmour

```
public void addArmour(double armr)
```

Overrides:

addArmour in class AbstractMonster

---

addAttack

```
public void addAttack(double attack)
```

Overrides:

addAttack in class AbstractMonster

---

addHP

```
public void addHP(double HP)
```

Overrides:

addHP in class AbstractMonster

---

collectIron `public void`

`collectIron`()

Zwiększa ilość posiadanego przez moba żelaza o 1

createSword

`public void` **`createSword`**()

metoda tworzy oraz dodaje do Ekwipunku moba jedną sztukę Miecza oraz odejmuje od ilości żelaza oraz kawałków drewna cenę za stworzenie Miecza (Odpowiednio 2 i 1)

---

getAttack

`public double` **`getAttack`**()

Overrides:

getAttack in class AbstractMonster

---

getDefence

`public double` **`getDefence`**()

Overrides:

getDefence in class AbstractMonster

---

getHealth

`public double` **`getHealth`**()

Overrides:

getHealth in class AbstractMonster

---

pl.projekt.game.mob

Class Minotaur

`java.lang.Object`
`    |`

```
    +--AbstractMonster
         |
         +--pl.projekt.game.mob.Minotaur
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Minotaur**

extends AbstractMonster

Minotaur jest jednym z 4 ras pojawiających się w symulacji. Charakteryzuje się wysokim atakiem oraz niskim życiem oraz obroną.Potrafi wytorzyć Buzdygan.

Constructors

Minotaur

```
public Minotaur()
```

Minotaur

```
public  Minotaur(double  healthPoints,
                 double defencePoints,
                 double attacPoints)
```

Methods

addArmour

```
public void addArmour(double armr)
```

Overrides:

addArmour in class AbstractMonster

---

addAttack

`public void addAttack(double attack)`

Overrides:

addAttack in class AbstractMonster

---

addHP

`public void addHP(double HP)`

Overrides:

addHP in class AbstractMonster

collectIron `public void`

`collectIron`()

Zwiększa ilość posiadanego przez moba żelaza o 1

---

collectStone

`public void` **`collectStone`**()

metoda zwiększa ilość posiadanego przez moba kamienia o 1

---

createMaze

`public void` **`createMaze`**()

metoda tworzy oraz dodaje do Ekwipunku moba jedną sztukę Buzdyganu oraz odejmuje od ilości żelaza,kawałków drewna oraz kamienia cenę za stworzenie Buzdyganu(odpowiednio 1,1 i 2)

---

getAttack

`public double` **`getAttack`**()

Overrides:

getAttack in class AbstractMonster

---

getDefence

`public double` **`getDefence`**()

Overrides:

getDefence in class AbstractMonster

---

getHealth

`public double` **`getHealth`**()

Overrides:

---

getHealth in class AbstractMonster

pl.projekt.game.mob

Class MinotaurTest

```
java.lang.Object
    |
    +--pl.projekt.game.mob.MinotaurTest
```

< Constructors > < Methods >

---

public class **MinotaurTest**
extends java.lang.Object

Constructors

MinotaurTest

```
public MinotaurTest()
```

Methods

collectWood

```
public void collectWood()
```

---

createJewelery

```
public void createJewelery()
```

---

pl.projekt.game.mob

Class Orc

```
java.lang.Object
```

```
    |
    +--AbstractMonster
        |
        +--pl.projekt.game.mob.Orc
```

All Implemented Interfaces:

IStats

< Constructors > < Methods >

---

public class **Orc** extends
AbstractMonster

Ork jest jednym z 4 ras pojawiających się w symulacji. Charakteryzuje się dużym życiem oraz średnim atakiem oraz defensywą.Potrafi tworzyć Topór.

Constructors

Orc

```
public Orc()
```

Orc

```
public  Orc(double  healthPoints,
            double defencePoints,
            double attacPoints)
```

Methods

addArmour

```
public void addArmour(double armr)
```

Overrides:

addArmour in class AbstractMonster

---

addAttack

```
public void addAttack(double attack)
```

Overrides:

addAttack in class AbstractMonster

addHP

`public void **addHP**(double HP)`

Overrides:

addHP in class AbstractMonster

---

collectStone

`public void **collectStone**()` metoda zwiÄ™ksza iloÅ›Ä‡

posiadanego przez moba kamienia o 1

---

createAxe

`public void **createAxe**()`

metoda tworzy oraz dodaje do Ekwipunku moba jedną sztuke Toporu oraz odejmuje od ilości kawałków drewna oraz kamienia cenę za stworzenie Toporu(odpowiednio 2 i 2)

---

getAttack

`public double **getAttack**()`

Overrides:

getAttack in class AbstractMonster

---

getDefence

`public double **getDefence**()`

Overrides:

getDefence in class AbstractMonster

---

getHealth

`public double **getHealth**()`

Overrides:

getHealth in class AbstractMonster

Package pl.projekt.simulation

Interface Summary

IRandom

Class Summary

Board

Klasa, na której obiekcie pojawiają się materiały oraz moby walczą/łączą się

SimulationApp

Glowna klasa programu posiada metodę main oraz obsluguje inne klasy

---

pl.projekt.simulation

Class Board

```
java.lang.Object
    |
    +--pl.projekt.simulation.Board
```

All Implemented Interfaces:

IRandom

< Constructors > < Methods >

---

public class **Board**
extends java.lang.Object
implements IRandom

Klasa, na której obiekcie pojawiają się materiały oraz moby walczą/łączą się

Constructors

Board

```
public Board(int size,
             int mobs)
```

Methods

getInfo

`public java.lang.String` **`getInfo`**`()`

Metoda tworzy nowego stringa z informacjami o ilości mobów po rundzie

Returns:

zwraca String z informacjami o mobach

---

getPositionX

`public int` **`getPositionX`**`()`

Metoda losuje nowa kordynate X

Returns:

zwraca kordynate X

---

getPositionY

`public int` **`getPositionY`**`()`

Metoda losuje nowa kordynate Y

Returns:

zwraca kordynate Y

---

move

`public void` **`move`**`()`

Metoda uzywajac metody lookForMobs szuka miejsc na ktorych wystepuja moby,nastepnie uzywajac w petli metod getPositionX i getPositionY losuje nowe kordynaty dla wybranego moba,nastepnie sprawdza czy nowe kordynaty sa puste czy wystepuje na nich jakis mob/materiał:- jesli na na nowych kordynatach znajduje sie materiał używa metody collectMaterial. -jesli na nowych kordynatach znajduje sie mob takiej samej klasy używa metody merge. -jesli na nowych kordynatach znajduje sie mob innej klasy uzywa metody fight

---

onlyOneSpeciesLeft

`public boolean` **`onlyOneSpeciesLeft`**`()`

Metoda sprawdza czy na planszy znajduje się tylko jeden gatunek moba

Returns:

zwraca prawdę jeśli znajduje się tylko 1 a fałsz jeśli kilka

---

placeOnTheBoard

`public void` **`placeOnTheBoard`**`()`

Metoda tworzy nowa plansze a nastepnie ustawia na niej moby i materialy

---

setMaterialPosition

`public void` **`setMaterialPosition`**`()`

Metoda tworzy nowe materiały i ustawia je w randomowych miejscach na planszy

---

setMobPosition

`public void` **`setMobPosition`**`()`

Metoda tworzy ilosc mobów podanych przez użytkownika a następnie przy pomocy ustawia w randomowych miejscach na planszy

---

pl.projekt.simulation

Interface IRandom

< Methods >

---

public interface **IRandom**

Methods

getPositionX

`public int` **`getPositionX`**`()`

---

getPositionY

`public int` **`getPositionY`**`()`

move

```
public void move()
```

setMaterialPosition

```
public void setMaterialPosition()
```

setMobPosition

```
public void setMobPosition()
```

---

pl.projekt.simulation

Class SimulationApp

```
java.lang.Object
    |
    +--pl.projekt.simulation.SimulationApp
```

< Constructors > < Methods >

---

public class **SimulationApp**
extends java.lang.Object

Glowna klasa programu posiada metodę main oraz obsługuje inne klasy

Version:

v0.9.1

Author:

Jakub Gliwa, Kacper Ziejło

Constructors

SimulationApp

```
public SimulationApp()
```

Methods

main

```
public static void main(java.lang.String[] args)
```

INDEX

A

addArmour ... 2

addArmour ... 5

addArmour ... 6

addArmour ... 8

addArmour ... 9

addArmour ... 10

addArmour ... 12

addArmour ... 13

addArmour ... 15

addArmour ... 16

addArmour ... 24

addArmour ... 27

addArmour ... 29

addArmour ... 31

addArmour ... 34

addAttack ... 2

addAttack ... 6

addAttack ... 7

addAttack ... 8

addAttack ... 9

addAttack ... 11

Dagger

addAttack ... 12

Dagger

addAttack ... 14

Dagger

addAttack ... 15

Diamond

addAttack ... 16

Diamond

addAttack ... 24

Diamond

B

Board ... 36

Board ... 36

C

collectIron ... 29

collectIron ... 32

collectMaterial ... 24

collectStone ... 27

collectStone ... 32

collectStone ... 35

collectWood ... 33

craftNewItem ... 24

createAxe ... 35

createHammer ... 27

createJewelery ... 33

createMaze ... 32

createSword ... 30

D

... 8

... 9

... 9

... 18

... 18

... 19

addAttack ... 27
Dwarf
addAttack ... 29
Dwarf
addAttack ... 31
Dwarf
addAttack ... 34
addHP ... 2
addHP ... 6
addHP ... 7
addHP ... 8
addHP ... 9
addHP ... 11
addHP ... 12
addHP ... 14
addHP ... 15
addHP ... 16
addHP ... 24
addHP ... 27
addHP ... 29
addHP ... 31
addHP ... 35
AbstractItem ... 3
AbstractItem ... 3
AbstractMaterials ... 17
AbstractMaterials ... 17
AbstractMonster ... 23
AbstractMonster ... 24
Armor ... 5
Armor ... 5
Armor ... 5
ArmorTest ... 6
ArmorTest ... 7
Axe ... 7
... 26
... 26
... 26

E

Elf ... 28
Elf ... 29
Elf ... 29

F

fight ... 25

Axe 8

Axe ... 7

G

getArmourPoints ... 4

getAttack ... 25

getAttack ... 28

getAttack ... 30

getAttack ... 32

getAttack ... 35

getDamagePoints ... 4

getDefence ... 25

getDefence ... 28

getDefence ... 30

getDefence ... 32

getDefence ... 35

getEquipment ... 25

getHealth ... 25

getHealth ... 28

getHealth ... 30

getHealth ... 32

getHealth ... 35

getHpPoints ... 4

getInfo ... 37

getMob1 ... 4

getPositionX ... 37

placeOnTheBoard

getPositionX ... 38

getPositionY ... 37

getPositionY ... 38

getWeight ... 18

getWeight ... 19

getWeight ... 20

M

main ... 40

merge ... 26

move ... 37

move ... 39

Maze ... 13

Maze ... 13

Maze ... 13

Minotaur ... 30

Minotaur ... 31

Minotaur ... 31

MinotaurTest ... 33

MinotaurTest ... 33

O

onlyOneSpeciesLeft ... 37

Orc ... 33

Orc ... 34

Orc ... 34

P

... 38

S

setArmourPoints ... 4

setDamagePoints ... 4

setHpPoints ... 4

getWeight ... 21
getWeight ... 22
getWoodnmb ... 25
setMobPosition ... 38

H

Shield ... 14
Hammer ... 10
Shield
Hammer ... 10
Shield
Hammer ... 10
SimulationApp
SimulationApp ... 39

I

isNotToHeavy ... 18
IRandom ... 38
Iron ... 19
Iron ... 19
Iron ... 19
IStats ... 2

J

Jewelery ... 11
Jewelery ... 11
Jewelery ... 12
setMaterialPosition ... 38
setMaterialPosition ... 39
setMob1 ... 4
setMobPosition ... 39
... 14
... 15
... 39
Stone ... 20
Stone ... 20
Stone ... 20
Sword ... 15
Sword ... 16
Sword ... 16

W

Wood ... 21
Wood ... 21
Wood ... 21