JAVA 1

Space Pig Fighter

Authors : Vincent Reynaert, Nicolas Sobczak

Contents

Chapter 1

Game presentation

1.1 What is it ?

"Space Pig Fighter" is a game that is played in the terminal by 2 players. Each player is a space pig and have to beat the other one.

A game happens in 2 phases. The first one is a spacecraft battle. The second one is a melee battle. Each spacecraft has several caracteristics. Each pig has several caracteristics and some weapon.

1.2 Rules

1.2.1 Animal class

Here are the concept we chose :

Animal class	Life	Force	Resistance	Special attack
Bear	mid	mid	big	damageAnnulation
Chicken	low	big	mid	triple attack
Duck	big	mid	low	fly
Pig	mid	low	big	moreDamage
Tiger	mid	big	low	paralyze foe which can’t attack next turn

Here are the exact values we chose :

Animal class	Life (hp)	Force	Resistance	Special attack
Bear	1000	110	40	damageAnnulation: nn
Chicken	800	130	20	triple attack: nn
Duck	1200	110	0	fly: nn
Pig	1000	90	40	moreDamage: nn
Tiger	1000	130	0	paralyze foe which can’t attack next turn: nn

1.2.2 Animal specialAttack

Bear - damageAnnulation

Pig - moreDamage

Tiger - paralyze foe which can't attack next turn

Chicken - tripleAttack, 1 turn to recharge after

Duck - fly, dodge attack

1.2.3 Meteorites malus

Size	Malus
small	-20 hp
medium	-50 hp
big	-100 hp

1.2.4 Stuff choice

You have 2 skill points to share between offensif and defensif stuff. You may choose to boost your attack at the expense of the your defense or to boost your defense at the expense of the your attack. Unless you prefer to choose a well balanced build.

Build	Attack points	Defense points
Offensive	2	0
Well balanced	1	1
Defensive	0	2

Here are the bonus value of each stuff :

Build	Offensive stuff	Stats bonus	Defensive stuff	Stats bonus
Offensive	Axe	40	None	00
Well balanced	Sword	20	Helmet	20
Defensive	None	00	Shield	40

1.3 How it is thought/programmed

Each player plays when it is its turn.

Chapter 2

Story: what happens when you launch the game

To launch the game, you need to run the Main class from the spacePigFighter package. What happens next ? That is what is described below.

2.1 Start the game

Game welcome players.

- player 1 is invited to choose his animal, enter animal's pseudo and color (pink by default), his spacecraft's color (gray by default).
- player 2 is invited to choose his animal, enter animal's pseudo and color (pink by default), his spacecraft's color (gray by default).

2.2 Part 1 of the game

- launch part1 of the game: space battle. You have to find the right location of the other player's spacecraft by entering a position. Each player try to guess turn by turn. You have to be careful, avoid meteorites ! Otherwise your pig's life will decrease.
- when a player find the other one's spacecraft, he climbs aboard and it's time for part 2 of the game.

2.3 Part 2 of the game

Players are welcomed to choose a stuff build in order to fight the other player.

1 turn happens in 3 steps:

1- Player 1 choose an action for his animal to do (choose between normal attack, special action and scream)

2- Player 2 choose an action for his animal to do (choose between normal attack, special action and scream)

3- Resolution

Game is over when a animal has no life point left. Since the resolution happens after both player's action, the result can be a draw.

2.4 End

At the end of the game, a file is created with the game summary written in it. If the file already exists, it is overwritten.

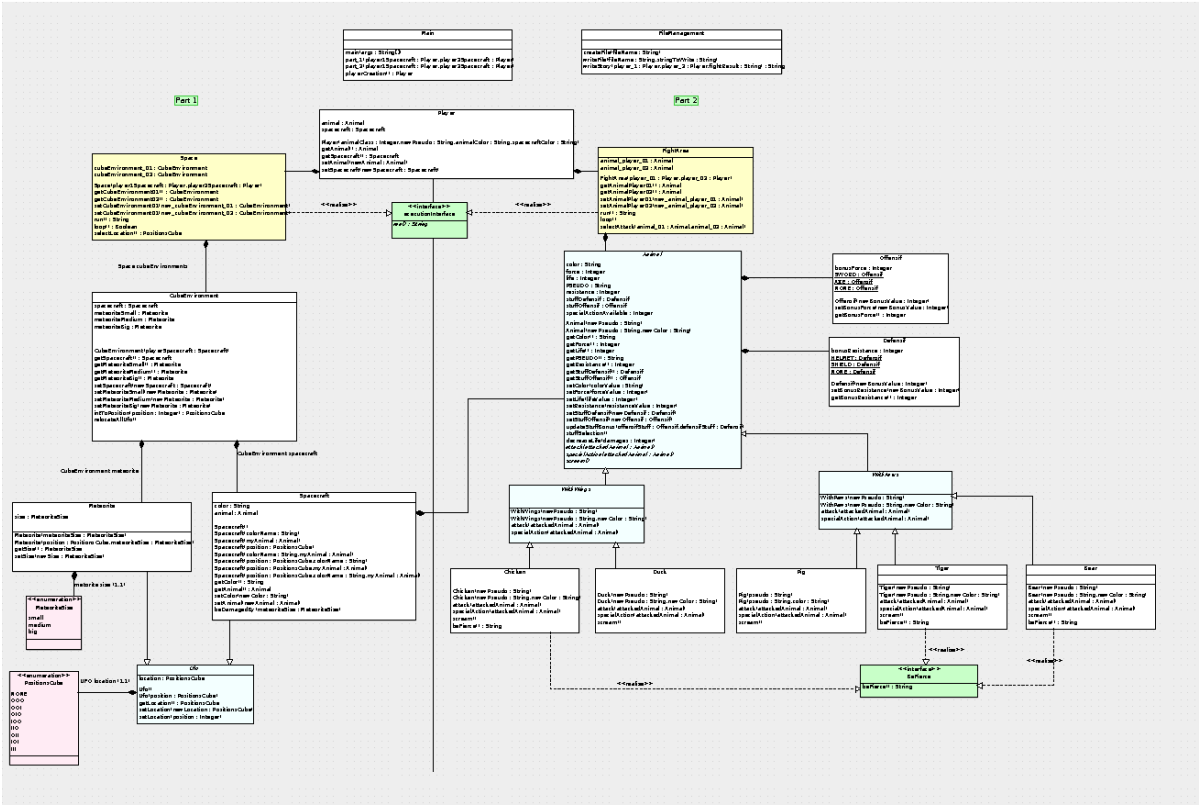
Chapter 3

Development part

Each player plays when it is its turn.

3.1 UML

Here is the global UML diagram of the program:



Since you can't see anything on this screenshot, there bigger screenshots below.

Blue classes are abstract classes.
Green classes are interface.
Pink classes are enumeration.
Purple classes are exception.
Yellow classes are the two main classes from the 2 different parts of the game.

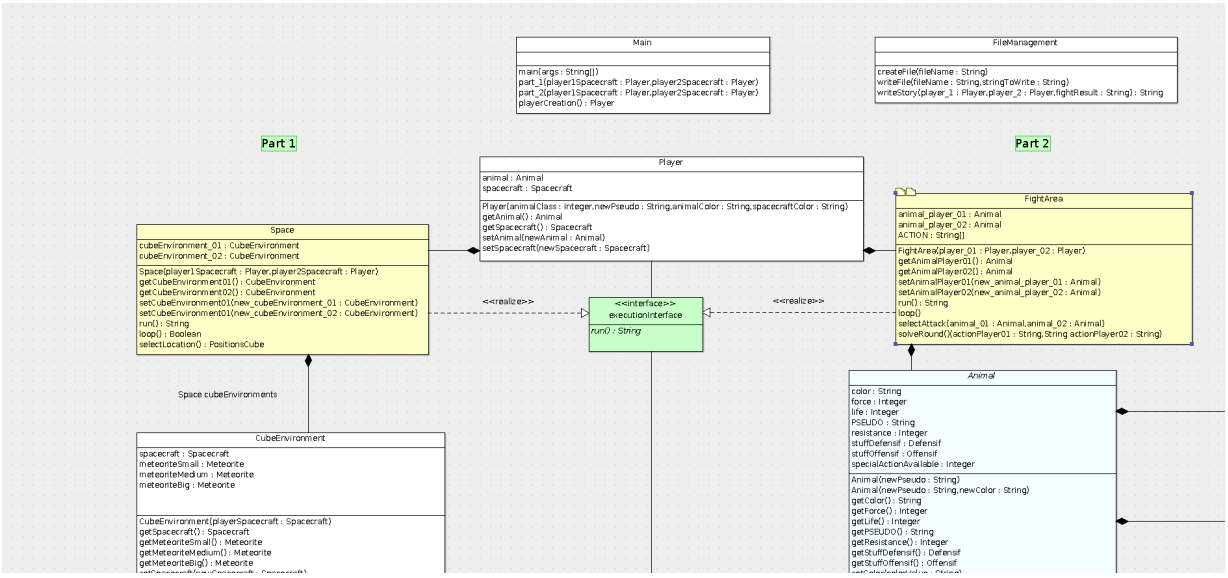


Figure 3.1 left screenshot 1

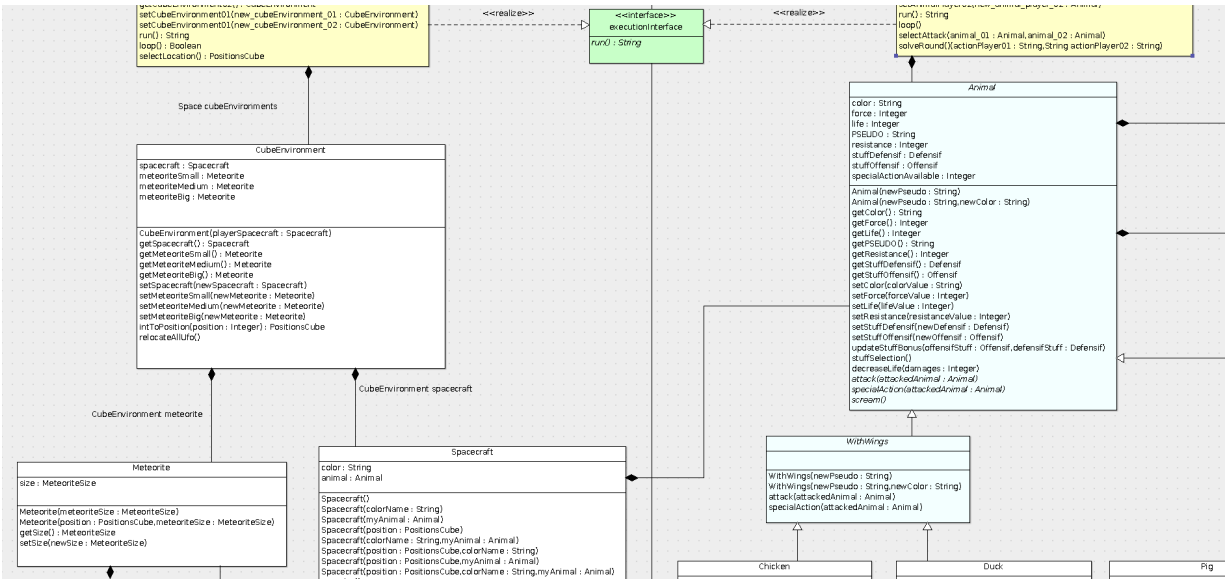


Figure 3.2 left screenshot 2

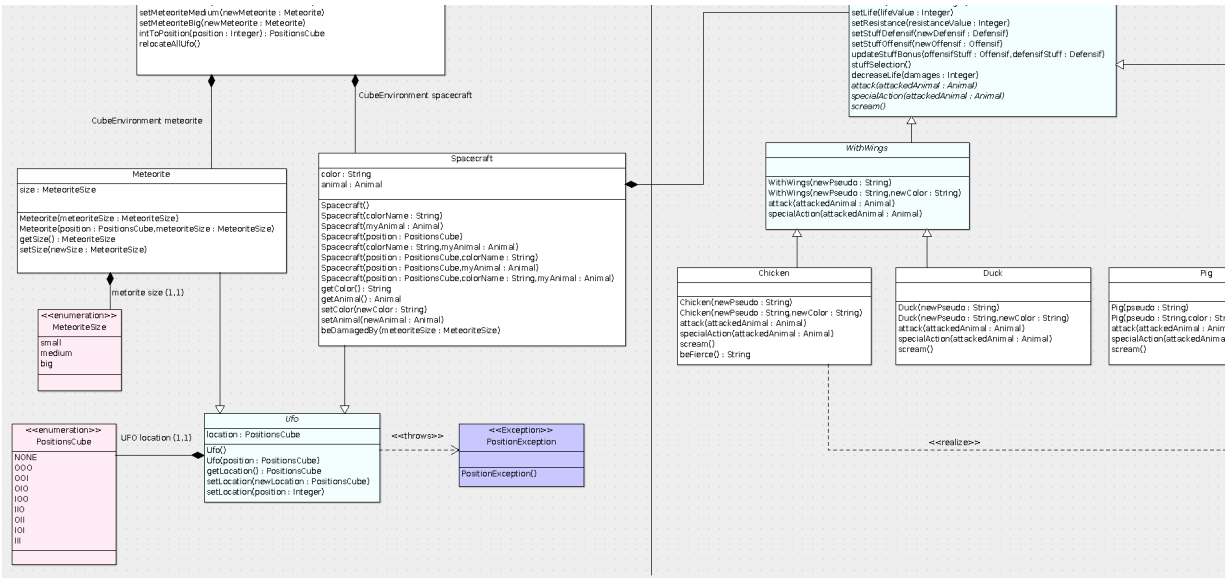


Figure 3.3 left screenshot 3

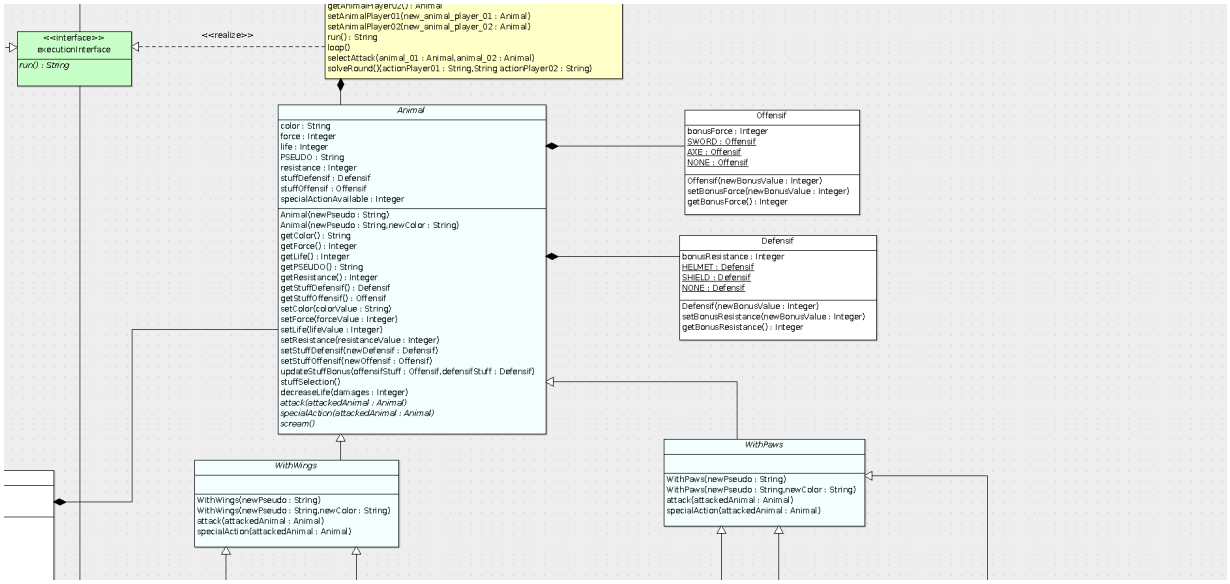


Figure 3.4 right screenshot 1

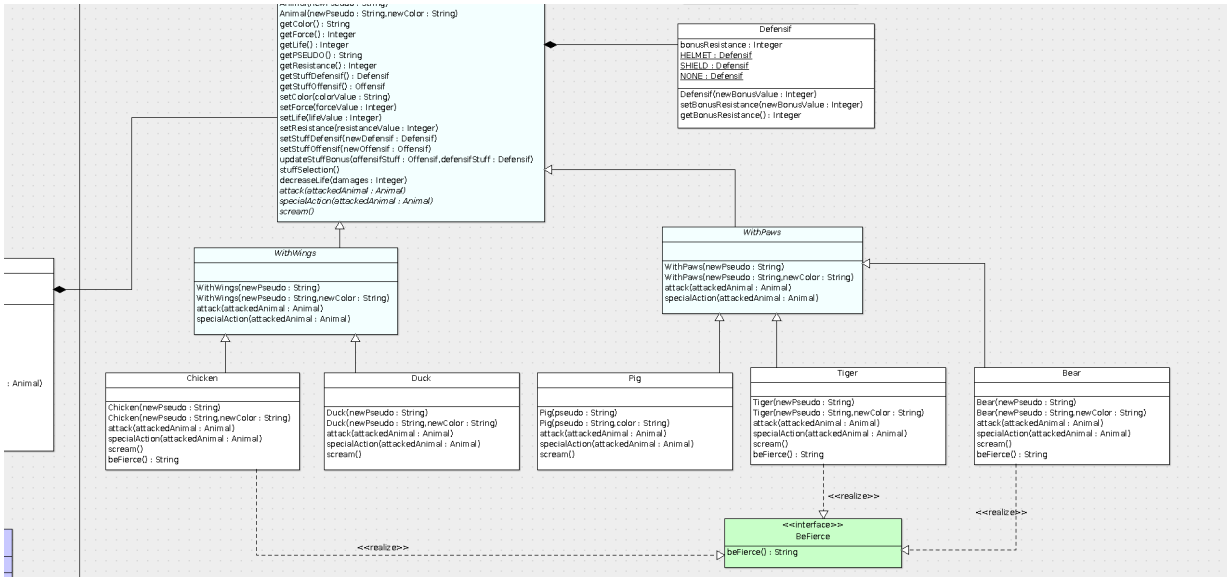


Figure 3.5 right screenshot 2

3.2 Organisational part: package description

We created package to organize our project. The main package which contains the main classes is called spacePigFighterPackage.

3.2.1 spacePigFighterPackage

This is the main package. It contains the following classes:

- Main
- Space
- FightArea
- ExecutionInterface

3.2.2 fileManagementPackage

This package contains all classes needed to interact with files. It contains the following class:

- FileManagement

3.2.3 playerPackage

This package contains all classes needed to create player. It contains the following class:

- Player

3.2.4 cubeEnvironment

This package contains all classes needed to create space environment. It contains the following class:

- CubeEnvironment

3.2.5 spaceObjects

This package contains all classes needed to manage space objects. It contains the following classes:

- UFO
- PositionsCube
- Meteorite
- MeteoriteSize
- Spacecraft
- PositionException

3.2.6 animalPackage

This package contains all classes needed to manage animal. It does not contain stuff classes. It contains the following classes:

- Animal
- WithPaws
- WithWings
- Bear
- Chicken
- Duck
- Pig
- Tiger
- BeFierce

3.2.7 stuff

This package contains all stuff classes. It contains the following classes:

- Offensif
- Defensif

3.3 Technical part: class description

This part contains a brief description of all project classes.

3.3.1 Main

This class contains the main functions:

- main : main function that calls all the following functions.
- playerCreation : function that create the 2 players.
- part_1 : function that runs game part 1.
- part_2 : function that runs game part 2.

3.3.2 FileManagement

This class contains all useful functions to save the game story in a file. We chose to put them in a class in order not to overload the Main class.

3.3.3 Player

We created a Player class that keeps all information about each player. That is to say that a player contains a spacecraft and its animal. It is from this class that we can access all information at any time and everywhere in our code.

3.3.4 The 2 main classes of the game

We created 1 class for each part of the game. It is from these 2 classes that each part is run. They both implements the executionInterface interface.

3.3.4.1 executionInterface interface:

This interface has only one function: *run()*. We decided to create this interface in order to create a name convention for the function which runs each part of the game. By doing this, the Main class won't change, it will always call the *run()* function of each class even if each class change.

3.3.4.2 Space class:

It is composed by 2 CubeEnvironments created thanks to the 2 Players. It has 3 main functions :

- run() : main function from the interface, it runs all game part 1.
- loop() : it runs the main loop while no spacecraft has been found, each player select a location en try to guess spacecraft postition.
- selectLocation(): it return the position selected by a player.

3.3.4.3 FightArea class:

It is composed by 2 Animals created thanks to the 2 Players and a list of special actions. It has 4 main functions :

- run() : main function from the interface, it runs all game part 2.
- loop() : it runs the main loop while no dead animal has been found, each player select an action to do.
- selectAttack(): it allows a player to select an action for its animal to do.
- solveRound(): this function manage special actions.

3.3.5 Part1

3.3.5.1 CubeEnvironment class

We thought the space environment in a particular way. Indeed, we assimilate it to 2 cubes, 1 for each player. That's why the Space class is composed of 2 CubeEnvironment. Each cube is composed of a spacecraft and 3 meteorites. They can be located to 8 different positions that correspond to each corner of the cube.

During the 1st part of the game, each player try to find the location of the other one's spacecraft. Of course he has to avoid meteorites that decrease the life. Once one player find the other one, part 2 of the game is started.

3.3.5.2 UFO class

It is an abstract class. It was created in order to manage position of both meteorites en spacecrafts. That's why Meteorite class and Spacecraft class both extends UFO abstract class.

To manage location, an UFO has an attribute *location*. We also created function which make us be able to manage location. Constructor was overloaded in order to create a UFO default position (000) or take the position in parameter.

3.3.5.3 PositionsCube enumeration

This enumeration enumerates all available positions in a cube. These positions match each corner of the cube. They are coordinates.

3.3.5.4 meteorites

There are 3 meteorites in each cube. A Meteorite has size which can be one from the MeteoriteSize enumeration. The size impact the amount of life to withdraw to an animal if a player collides a meteorite.

3.3.5.5 MeteoriteSize

This enumeration enumerates all existing meteorite size.

3.3.5.6 spacecraft

There is one spacecraft in each cube. Spacecraft class has a color and an Animal. The spacecraft can be damaged by a meteorite. A damaged spacecraft means its animal life decreases.

3.3.6 Part2

3.3.6.1 Animal class

It is an abstract class.

3.3.6.2 WithWings class

It is an abstract class which extends animal class. It overrides *attack()* function to caracterize it by the way the animal attack (with paws or with wings).

3.3.6.3 WithPaws class

It is an abstract class which extends animal class. It overrides *attack()* function to caracterize it by the way the animal attack (with paws or with wings).

3.3.6.4 Bear class

Bear is an animal with paws. That's why it extends WithPaws abstract class. It overrides *attack()*, *specialAction()* and *scream()* functions. Since Bear is a fierce animal, it implements BeFierce interface and overrides *beFierce()* function.

3.3.6.5 Chicken class

Chicken is an animal with paws. That's why it extends WithWings abstract class. It overrides *attack()*, *specialAction()* and *scream()* functions. Since Chicken is a fierce animal, it implements BeFierce interface and overrides *beFierce()* function.

3.3.6.6 Duck class

Duck is an animal with paws. That's why it extends WithWings abstract class. It overrides *attack()*, *specialAction()* and *scream()* functions.

3.3.6.7 Pig class

Pig is an animal with paws. That's why it extends WithPaws abstract class. It overrides *attack()*, *specialAction()* and *scream()* functions.

3.3.6.8 Tiger class

Tiger is an animal with paws. That's why it extends WithPaws abstract class. It overrides *attack()*, *specialAction()* and *scream()* functions. Since Tiger is a fierce animal, it implements BeFierce interface and overrides *beFierce()* function.

3.3.6.9 BeFierce interface

This interface was created to caracterize scream of some animals that are said to be fierce. It contains 1 function, *beFierce()* function.

3.3.6.10 Offensif class

Each animal has an offensive stuff which gives it a force bonus. Offensif class is here to do that. It has force bonus value and constants that defines existing offensive stuff.

3.3.6.11 Defensif class

Each animal has an defensive stuff which gives it a force bonus. Defensif class is here to do that. It has force bonus value and constants that defines existing defensive stuff.

3.3.7 Set the game

- set Player class for each player.
- set Space class with 2 CubeEnvironment (1 for each player). Each CubeEnvironment is set with 3 meteorites and 1 spacecraft.
- set FightArea class with 2 pigs. Each pig is initialized with stuff selected by the player.

3.4 Encountered difficulties

3.4.1 Special action

Special actions are very different. So we had to think our code so that it would be able to welcome each special action. We had to modify our code a little bit and to add the *solveRound()* function from FightArea.

3.4.2 Exception

We created an exception. We had difficultie because it was the first time and we didn't undertand exception very well. We no longer do !

Chapter 4

Conclusion

We think our project cover lots of different aspects of java language. Besides we enjoyed doing this game. That's why we may add a graphical interface in the future.

Chapter 5

Class Index

5.1 Class List

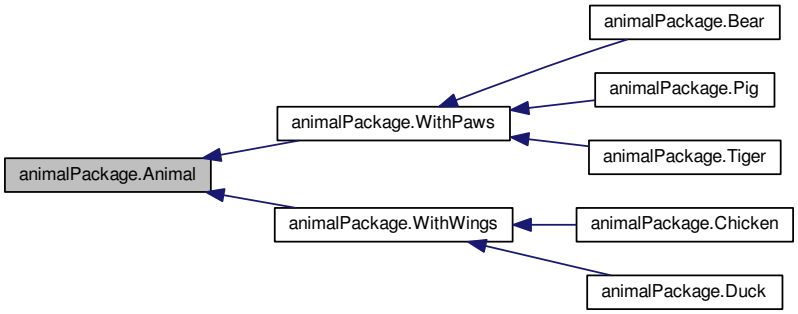
Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 6

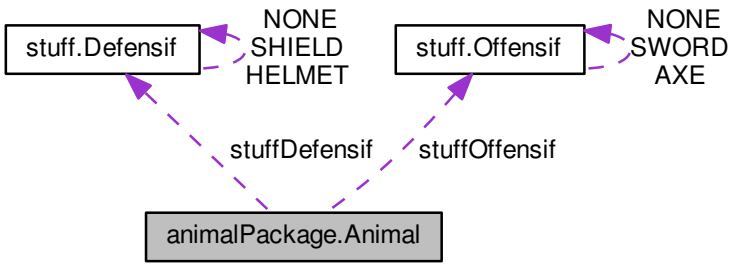
Class Documentation

6.1 animalPackage.Animal Class Reference

Inheritance diagram for animalPackage.Animal:



Collaboration diagram for animalPackage.Animal:



Public Member Functions

- Animal (String newPseudo)
- Animal (String newPseudo, String newColor)
- String getColor ()
- Integer getForce ()
- Integer getLife ()
- String getPSEUDO ()
- Integer getResistance ()
- Defensif getStuffDefensif ()
- Offensif getStuffOffensif ()
- Integer getSpecialActionAvailable ()
- Boolean getAbleToAct ()
- void setColor (String colorValue)
- void setForce (Integer forceValue)
- void setResistance (Integer resistanceValue)
- void setLife (Integer lifeValue)
- void setStuffDefensif (Defensif newDefensif)
- void setStuffOffensif (Offensif newOffensif)
- void setSpecialActionAvailable (int newSpecialActionAvailable)
- void setAbleToAct (Boolean abilityToAct)
- void updateStuffBonus (Offensif offensifStuff, Defensif defensifStuff)
- void stuffSelection ()
- void decreaseLife (Integer damages)
- void increaseLife (Integer bonus)
- abstract void attack (Animal attackedAnimal)
- abstract String specialAction (Animal attackedAnimal)
- abstract void scream ()

Protected Attributes

- String **color**
- Integer **life**
- Integer **force**
- Integer **resistance**
- Defensif **stuffDefensif**
- Offensif **stuffOffensif**
- Integer **specialActionAvailable**
- Boolean **ableToAct**

6.1.1 Detailed Description

===== Abstract Class Animal =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.01, 10/2016

6.1.2 Constructor & Destructor Documentation

6.1.2.1 animalPackage.Animal.Animal (String *newPseudo*)

Constructor

Parameters

1	String = animal's Pseudo

6.1.2.2 animalPackage.Animal.Animal (String *newPseudo,* String *newColor*)

Constructor

Parameters

1	String = animal's Pseudo
1	String = animal's color

6.1.3 Member Function Documentation

6.1.3.1 abstract void animalPackage.Animal.attack (Animal *attackedAnimal*) [abstract]

attack : abstract function which executes a normal attack

Parameters

Animal	attackedAnimal

6.1.3.2 void animalPackage.Animal.decreaseLife (Integer *damages*)

Decrease animal's life

Parameters

1	Integer = damages

6.1.3.3 Boolean animalPackage.Animal.getAbleToAct ( )

Get animal's ableToAct

Returns

1 Boolean = animal's ability to act

6.1.3.4 String animalPackage.Animal.getColor ( )

Get animal's color

Returns

1 String = animal's color value

6.1.3.5 Integer animalPackage.Animal.getForce ( )

Get animal's force

Returns

1 int = animal's force value

6.1.3.6 Integer animalPackage.Animal.getLife ( )

Get animal's life

Returns

1 int = animal's life value

6.1.3.7 String animalPackage.Animal.getPSEUDO ( )

Get animal's pseudo

Returns

1 String = animal's pseudo value

6.1.3.8 Integer animalPackage.Animal.getResistance ( )

Get animal's resistance

Returns

1 int = animal's resistance value

6.1.3.9 Integer animalPackage.Animal.getSpecialActionAvailable ( )

Get animal's specialActionAvailable

Returns

1 int = animal's specialActionAvailable

6.1.3.10 Defensif animalPackage.Animal.getStuffDefensif ( )

Get animal's defensif stuff

Returns

1 Defensif = animal's defensif stuff

6.1.3.11 Offensif animalPackage.Animal.getStuffOffensif ( )

Get animal's offensif stuff

Returns

1 Offensif = animal's offensif stuff

6.1.3.12 void animalPackage.Animal.increaseLife (Integer *bonus*)

Increase animal's life

Parameters

1	Integer = bonus

6.1.3.13 abstract void animalPackage.Animal.scream ( ) [abstract]

scream : function which makes the animal scream

6.1.3.14 void animalPackage.Animal.setAbleToAct (Boolean *abilityToAct*)

Set animal's ableToAct

1 Boolean = animal's ability to act

6.1.3.15 void animalPackage.Animal.setColor (String *colorValue*)

Set animal's color

Parameters

1	String = animal's color value

6.1.3.16 void animalPackage.Animal.setForce (Integer *forceValue*)

Set animal's force

Parameters

1	int = animal's force value

6.1.3.17 void animalPackage.Animal.setLife (Integer *lifeValue*)

Set animal's life

Parameters

1	int = animal's life value

6.1.3.18 void animalPackage.Animal.setResistance (Integer *resistanceValue*)

Set animal's resistance

Parameters

1	int = animal's resistance value

6.1.3.19 void animalPackage.Animal.setSpecialActionAvailable (int *newSpecialActionAvailable*)

Get animal's specialActionAvailable

Parameters

1	int = animal's newSpecialActionAvailable

6.1.3.20 void animalPackage.Animal.setStuffDefensif (**Defensif** *newDefensif*)

Set animal's defensif stuff

Parameters

1	Defensif = animal's defensif stuff

6.1.3.21 void animalPackage.Animal.setStuffOffensif (**Offensif** *newOffensif*)

Set animal's offensif stuff

Parameters

1	Offensif = animal's offensif stuff

6.1.3.22 abstract String animalPackage.Animal.specialAction (**Animal** *attackedAnimal*) [abstract]

attack : abstract function which executes a special attack

Parameters

Animal	attackedAnimal

6.1.3.23 void animalPackage.Animal.stuffSelection ( )

stuffSelection

6.1.3.24 void animalPackage.Animal.updateStuffBonus (**Offensif** *offensifStuff*, **Defensif** *defensifStuff*)

Apply animal's stuff bonus

Parameters

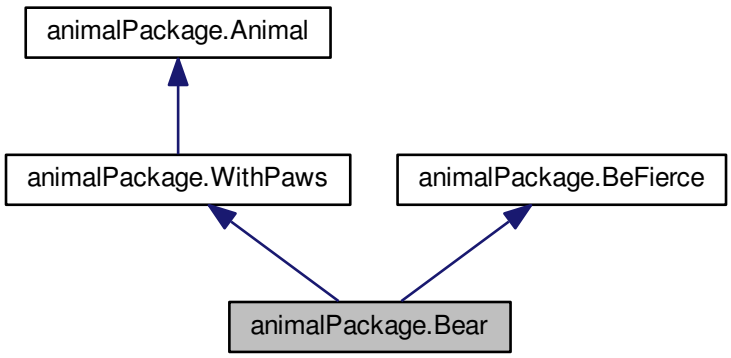
1	Offensif offensifStuff
1	Defensif defensifStuff

The documentation for this class was generated from the following file:

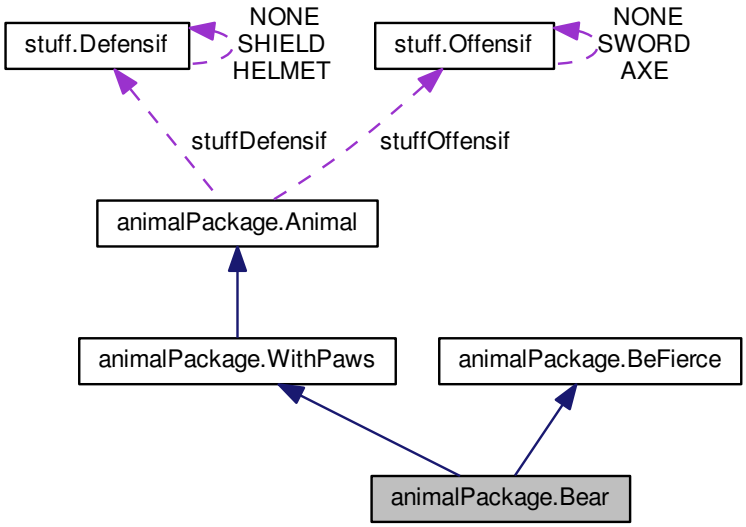
- src/animalPackage/Animal.java

6.2 animalPackage.Bear Class Reference

Inheritance diagram for animalPackage.Bear:



Collaboration diagram for animalPackage.Bear:



Public Member Functions

- Bear (String newPseudo)
- Bear (String newPseudo, String newColor)
- void attack (Animal attackedAnimal)
- String specialAction (Animal attackedAnimal)
- void scream ()
- String beFierce ()

Additional Inherited Members

6.2.1 Detailed Description

===== Class Bear =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.2.2 Constructor & Destructor Documentation

6.2.2.1 animalPackage.Bear.Bear (String *newPseudo*)

Constructor

Parameters

1	String = bear's Pseudo

6.2.2.2 animalPackage.Bear.Bear (String *newPseudo,* String *newColor*)

Constructor

Parameters

1	String = bear's Pseudo
1	String = bear's color

6.2.3 Member Function Documentation

6.2.3.1 void animalPackage.Bear.attack (Animal *attackedAnimal*)

attack : function which executes a basic attack

Parameters

Animal	attackedAnimal

6.2.3.2 String animalPackage.Bear.beFierce ( )

beFierce : function which return an adjective to describe behavior

Returns

1 String = an adjective

Implements animalPackage.BeFierce.

6.2.3.3 void animalPackage.Bear.scream ( )

scream : function which makes the animal scream

6.2.3.4 String animalPackage.Bear.specialAction (Animal *attackedAnimal*)

specialAction : function which executes a special attack For the bear it is damageAnnulation

Parameters

Animal	attackedAnimal

Returns

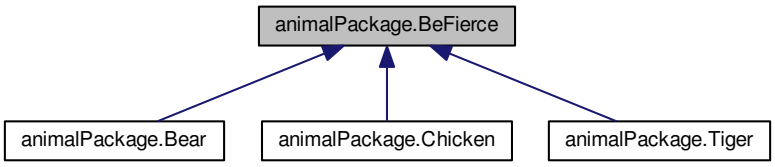
String

The documentation for this class was generated from the following file:

- src/animalPackage/Bear.java

6.3 animalPackage.BeFierce Interface Reference

Inheritance diagram for animalPackage.BeFierce:



Public Member Functions

- String beFierce ()

6.3.1 Detailed Description

===== interface BeFierce =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.01, 11/2016

6.3.2 Member Function Documentation

6.3.2.1 String animalPackage.BeFierce.beFierce ( )

beFierce : function which return an adjective to describe behavior

Returns

1 String = an adjective

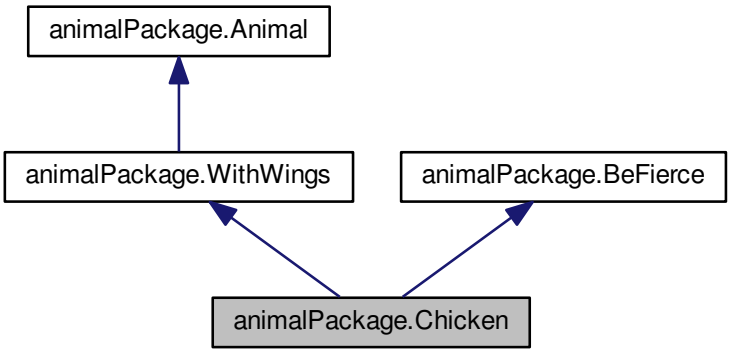
Implemented in animalPackage.Bear, animalPackage.Chicken, and animalPackage.Tiger.

The documentation for this interface was generated from the following file:

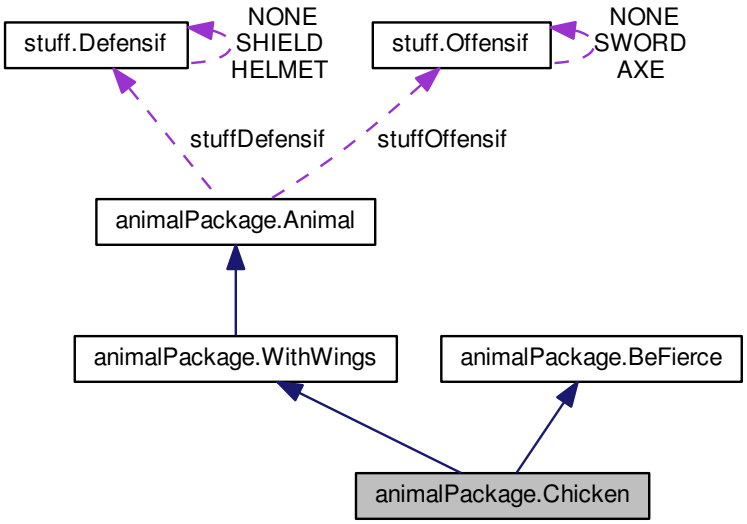
- src/animalPackage/BeFierce.java

6.4 animalPackage.Chicken Class Reference

Inheritance diagram for animalPackage.Chicken:



Collaboration diagram for animalPackage.Chicken:



Public Member Functions

- Chicken (String newPseudo)
- Chicken (String newPseudo, String newColor)
- void attack (Animal attackedAnimal)
- String specialAction (Animal attackedAnimal)
- void scream ()
- String beFierce ()

Additional Inherited Members

6.4.1 Detailed Description

===== Class Chicken =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.4.2 Constructor & Destructor Documentation

6.4.2.1 animalPackage.Chicken.Chicken (String *newPseudo*)

Constructor

Parameters

1	String = chicken's Pseudo

6.4.2.2 animalPackage.Chicken.Chicken (String *newPseudo,* String *newColor*)

Constructor

Parameters

1	String = chicken's Pseudo
1	String = chicken's color

6.4.3 Member Function Documentation

6.4.3.1 void animalPackage.Chicken.attack (Animal *attackedAnimal*)

attack : function which executes a basic attack

Parameters

Animal	attackedAnimal

Returns

String

6.4.3.2 String animalPackage.Chicken.beFierce ( )

beFierce : function which return an adjective to describe behavior

Returns

1 String = an adjective

Implements animalPackage.BeFierce.

6.4.3.3 void animalPackage.Chicken.scream ( )

scream : function which makes the animal scream

6.4.3.4 String animalPackage.Chicken.specialAction (Animal *attackedAnimal*)

specialAction : function which executes a special attack

Parameters

Animal	attackedAnimal

The documentation for this class was generated from the following file:

- src/animalPackage/Chicken.java

6.5 cubeEnvironment.CubeEnvironment Class Reference

Public Member Functions

- CubeEnvironment ()
- CubeEnvironment (Player playerI)
- Spacecraft getSpacecraft ()
- Meteorite getMeteoriteSmall ()
- Meteorite getMeteoriteMedium ()
- Meteorite getMeteoriteBig ()
- void setSpacecraft (Spacecraft newSpacecraft)
- void setMeteoriteSmall (Meteorite newMeteorite)
- void setMeteoriteMedium (Meteorite newMeteorite)
- void setMeteoriteBig (Meteorite newMeteorite)
- PositionsCube **intToPosition** (int position) throws PositionException
- void **relocateAllUfo** ()

6.5.1 Detailed Description

===== Class CubeEnvironment =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.5.2 Constructor & Destructor Documentation

6.5.2.1 cubeEnvironment.CubeEnvironment.CubeEnvironment ( )

Constructor

6.5.2.2 cubeEnvironment.CubeEnvironment.CubeEnvironment (Player *playerI*)

Constuctor

Parameters

1	Player = playerI

6.5.3 Member Function Documentation

6.5.3.1 Meteorite cubeEnvironment.CubeEnvironment.getMeteoriteBig ( )

Get CubeEnvironment meteoriteBig

Returns

1 Meteorite = meteoriteBig

6.5.3.2 Meteorite cubeEnvironment.CubeEnvironment.getMeteoriteMedium ( )

Get CubeEnvironment meteoriteMedium

Returns

1 Meteorite = meteoriteMedium

6.5.3.3 Meteorite cubeEnvironment.CubeEnvironment.getMeteoriteSmall ( )

Get CubeEnvironment meteoriteSmall

Returns

1 Meteorite = meteoriteSmall

6.5.3.4 Spacecraft cubeEnvironment.CubeEnvironment.getSpacecraft ( )

Get CubeEnvironment spacecraft

Returns

1 Spacecraft = spacecraft

6.5.3.5 void cubeEnvironment.CubeEnvironment.setMeteoriteBig (Meteorite *newMeteorite*)

Set CubeEnvironment meteoriteBig

Parameters

1	Meteorite = newMeteorite

6.5.3.6 void cubeEnvironment.CubeEnvironment.setMeteoriteMedium (Meteorite *newMeteorite*)

Set CubeEnvironment meteoriteMedium

Parameters

1	Meteorite = newMeteorite

6.5.3.7 void cubeEnvironment.CubeEnvironment.setMeteoriteSmall (Meteorite *newMeteorite*)

Set CubeEnvironment meteoriteSmall

Parameters

1	Meteorite = newMeteorite

6.5.3.8 void cubeEnvironment.CubeEnvironment.setSpacecraft (Spacecraft *newSpacecraft*)

Set CubeEnvironment spacecraft

Parameters

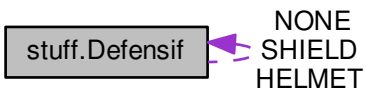
1	Spacecraft = newSpacecraft

The documentation for this class was generated from the following file:

- src/cubeEnvironment/CubeEnvironment.java

6.6 stuff.Defensif Class Reference

Collaboration diagram for stuff.Defensif:



Public Member Functions

- Defensif (Integer newBonusValue)
- void setBonusResistance (Integer newBonusValue)
- Integer getBonusResistance ()

Static Public Attributes

- static final Defensif HELMET = new Defensif(5)
- static final Defensif SHIELD = new Defensif(10)
- static final Defensif NONE = new Defensif(0)

6.6.1 Detailed Description

===== Class Defensif =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.02, 11/2016

6.6.2 Constructor & Destructor Documentation

6.6.2.1 stuff.Defensif.Defensif (Integer *newBonusValue*)

Constructor

Parameters

int	newBonusValue

6.6.3 Member Function Documentation

6.6.3.1 Integer stuff.Defensif.getBonusResistance ( )

Get the bonusResistance value

Returns

int bonusResistance

6.6.3.2 void stuff.Defensif.setBonusResistance (Integer *newBonusValue*)

Set the bonusResistance

Parameters

int	newBonusValue

6.6.4 Member Data Documentation

6.6.4.1 final Defensif stuff.Defensif.HELMET = new Defensif(5) [static]

Increases the resistance of 5

6.6.4.2 final Defensif stuff.Defensif.NONE = new Defensif(0) [static]

Doesn't increase the resistance

6.6.4.3 final Defensif stuff.Defensif.SHIELD = new Defensif(10) [static]

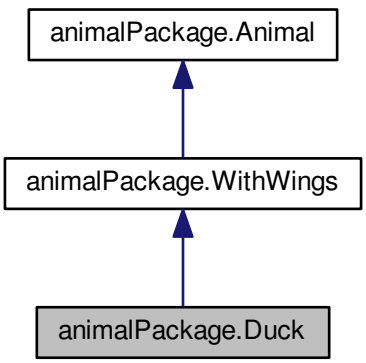
Increases the resistance of 10

The documentation for this class was generated from the following file:

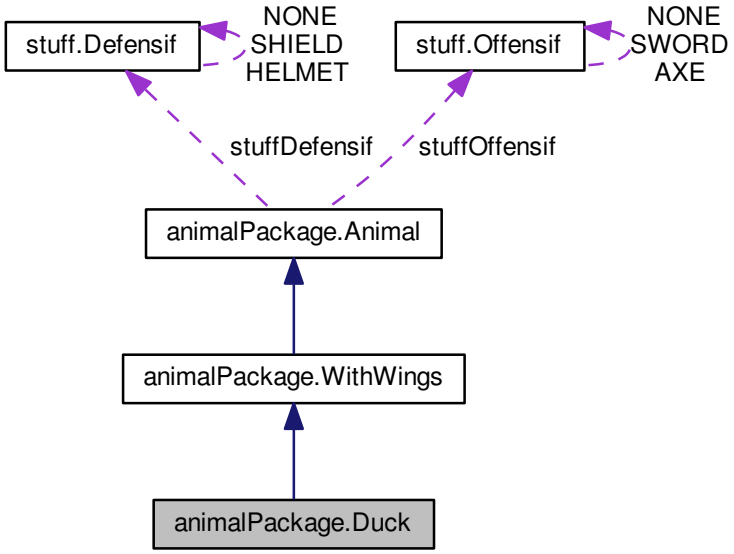
- src/stuff/Defensif.java

6.7 animalPackage.Duck Class Reference

Inheritance diagram for animalPackage.Duck:



Collaboration diagram for animalPackage.Duck:



Public Member Functions

- Duck (String newPseudo)
- Duck (String newPseudo, String newColor)
- void attack (Animal attackedAnimal)
- String specialAction (Animal attackedAnimal)
- void scream ()

Additional Inherited Members

6.7.1 Detailed Description

===== Class Duck =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.7.2 Constructor & Destructor Documentation

6.7.2.1 animalPackage.Duck.Duck (String *newPseudo*)

Constructor

Parameters

1	String = duck's Pseudo

6.7.2.2 animalPackage.Duck.Duck (String *newPseudo,* String *newColor*)

Constructor

Parameters

1	String = duck's Pseudo
1	String = duck's color

6.7.3 Member Function Documentation

6.7.3.1 void animalPackage.Duck.attack (Animal *attackedAnimal*)

attack : function which executes a basic attack

Parameters

Animal	attackedAnimal

6.7.3.2 void animalPackage.Duck.scream ( )

scream : function which makes the animal scream

6.7.3.3 String animalPackage.Duck.specialAction (Animal *attackedAnimal*)

specialAction : function which executes a special attack

Parameters

Animal	attackedAnimal

Returns

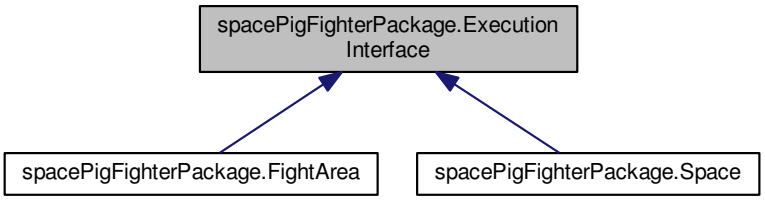
String

The documentation for this class was generated from the following file:

- src/animalPackage/Duck.java

6.8 spacePigFighterPackage.ExecutionInterface Interface Reference

Inheritance diagram for spacePigFighterPackage.ExecutionInterface:



Public Member Functions

- String **run** ()

6.8.1 Detailed Description

===== interface ExecutionInterface =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

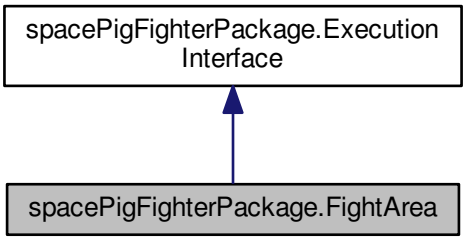
1.01, 11/2016

The documentation for this interface was generated from the following file:

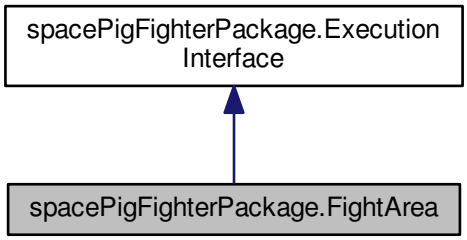
- src/spacePigFighterPackage/ExecutionInterface.java

6.9 spacePigFighterPackage.FightArea Class Reference

Inheritance diagram for spacePigFighterPackage.FightArea:



Collaboration diagram for spacePigFighterPackage.FightArea:



Public Member Functions

- FightArea (Player player_01, Player player_02)
- Animal getAnimalPlayer01 ()
- Animal getAnimalPlayer02 ()
- void setAnimalPlayer01 (Animal new_animal_player_01)
- void setAnimalPlayer02 (Animal new_animal_player_02)
- String run ()

6.9.1 Detailed Description

===== Class FightArea =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.05, 11/2016

6.9.2 Constructor & Destructor Documentation

6.9.2.1 spacePigFighterPackage.FightArea.FightArea (Player *player_01,* Player *player_02*)

Constructor

Parameters

1	Player = player_01
1	Player = player_02

6.9.3 Member Function Documentation

6.9.3.1 Animal spacePigFighterPackage.FightArea.getAnimalPlayer01 ( )

Get FighteArea animal_player_01

Returns

Animal animal_player_01

6.9.3.2 Animal spacePigFighterPackage.FightArea.getAnimalPlayer02 ( )

Get FighteArea animal_player_02

Returns

Animal animal_player_02

6.9.3.3 String spacePigFighterPackage.FightArea.run ( )

run() : function which gives the result

Implements spacePigFighterPackage.ExecutionInterface.

6.9.3.4 void spacePigFighterPackage.FightArea.setAnimalPlayer01 (Animal *new_animal_player_01*)

Set FighteArea animal_player_01

Parameters

Animal	new_animal_player_01

6.9.3.5 void spacePigFighterPackage.FightArea.setAnimalPlayer02 (Animal *new_animal_player_02*)

Set FighteArea animal_player_02

Parameters

Animal	new_animal_player_02

The documentation for this class was generated from the following file:

- src/spacePigFighterPackage/FightArea.java

6.10 fileManagementPackage.FileManagement Class Reference

Static Public Member Functions

- static void createFile (String fileName)
- static void writeFile (String fileName, String stringToWrite)
- static String writeStory (Player player_1, Player player_2, String fightResult)

6.10.1 Detailed Description

===== Class FileManagement =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.10.2 Member Function Documentation

6.10.2.1 static void fileManagementPackage.FileManagement.createFile (String *fileName*) [static]

createFile function that create a file

Parameters

1	String fileName

6.10.2.2 static void fileManagementPackage.FileManagement.writeFile (String *fileName,* String *stringToWrite*) [static]

writeFile function

Parameters

1	String fileName
1	String stringToWrite

6.10.2.3 static String fileManagementPackage.FileManagement.writeStory (Player *player_1,* Player *player_2,* String *fightResult*) [static]

writeStory function which writes the fight story

Parameters

2	Player player_1 and player_2
1	String fightResult : the result of the fightArea fight

The documentation for this class was generated from the following file:

- src/fileManagementPackage/FileManagement.java

6.11 spacePigFighterPackage.Main Class Reference

Static Public Member Functions

- static Player playerCreation ()
- static String part_1 (Player player_1, Player player_2)
- static String part_2 (Player player_1, Player player_2)
- static void main (String[ ] args)

6.11.1 Detailed Description

===== Class Main =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.01, 10/2016

6.11.2 Member Function Documentation

6.11.2.1 static void spacePigFighterPackage.Main.main (String[ ] *args*) [static]

main function

Parameters

1	String[] = args

6.11.2.2 static String spacePigFighterPackage.Main.part_1 (Player *player_1,* Player *player_2*) [static]

Game part 1 function

Parameters

1	Player = player_1
1	Player = player_2

6.11.2.3 static String spacePigFighterPackage.Main.part_2 (Player *player_1,* Player *player_2*) [static]

Game part 2 function

Parameters

1	Player = player_1
1	Player = player_2

6.11.2.4 static Player spacePigFighterPackage.Main.playerCreation ( ) [static]

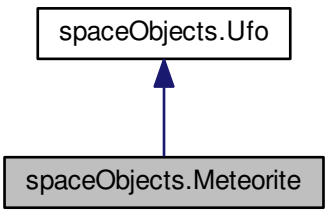
playerCreation function

The documentation for this class was generated from the following file:

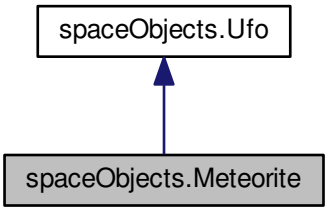
- src/spacePigFighterPackage/Main.java

6.12 spaceObjects.Meteorite Class Reference

Inheritance diagram for spaceObjects.Meteorite:



Collaboration diagram for spaceObjects.Meteorite:



Public Member Functions

- Meteorite (MeteoriteSize meteoriteSize)
- Meteorite (PositionsCube position, MeteoriteSize meteoriteSize)
- MeteoriteSize getSize ()
- void setSize (MeteoriteSize newSize)

6.12.1 Detailed Description

===== Class Meteorite =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.02, 11/2016

6.12.2 Constructor & Destructor Documentation

6.12.2.1 spaceObjects.Meteorite.Meteorite (MeteoriteSize *meteoriteSize*)

Constructor where size is necessary selected by the player

Parameters

1	MeteoriteSize = meteoriteSize

6.12.2.2 spaceObjects.Meteorite.Meteorite (PositionsCube *position,* MeteoriteSize *meteoriteSize*)

Constructor with selected position and size

Parameters

1	PositionsCube = position
1	MeteoriteSize = meteoriteSize

6.12.3 Member Function Documentation

6.12.3.1 MeteoriteSize spaceObjects.Meteorite.getSize ( )

Get the meteorite size

Returns

1 MeteoriteSize = size

6.12.3.2 void spaceObjects.Meteorite.setSize (MeteoriteSize *newSize*)

Set a new size to the meteorite

Parameters

1	MeteoriteSize = newSize

The documentation for this class was generated from the following file:

- src/spaceObjects/Meteorite.java

6.13 spaceObjects.MeteoriteSize Enum Reference

Public Attributes

- **SMALL**
- **MEDIUM**

6.13.1 Detailed Description

===== Enumeration MeteoriteSize =====

enumeration of available meteorite sizes

Author

Vincent Reynaert, Nicolas Sobczak

Version

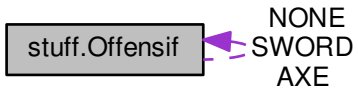
1.01, 10/2016

The documentation for this enum was generated from the following file:

- src/spaceObjects/MeteoriteSize.java

6.14 stuff.Offensif Class Reference

Collaboration diagram for stuff.Offensif:



Public Member Functions

- Offensif (Integer newBonusValue)
- void setBonusForce (Integer newBonusValue)
- Integer getBonusForce ()

Static Public Attributes

- static final Offensif SWORD = new Offensif(5)
- static final Offensif AXE = new Offensif(10)
- static final Offensif NONE = new Offensif(0)

6.14.1 Detailed Description

===== Class Offensif =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.02, 11/2016

6.14.2 Constructor & Destructor Documentation

6.14.2.1 stuff.Offensif.Offensif (Integer *newBonusValue*)

Constructor

Parameters

int	newBonusValue

6.14.3 Member Function Documentation

6.14.3.1 Integer stuff.Offensif.getBonusForce ( )

Get the bonusForce value

Returns

int bonusForce

6.14.3.2 void stuff.Offensif.setBonusForce (Integer *newBonusValue*)

Set the bonusForce

Parameters

int	newBonusValue

6.14.4 Member Data Documentation

6.14.4.1 final Offensif stuff.Offensif.AXE = new Offensif(10) [static]

Increases the force of 10

6.14.4.2 final Offensif stuff.Offensif.NONE = new Offensif(0) [static]

Doesn't increase the resistance

6.14.4.3 final Offensif stuff.Offensif.SWORD = new Offensif(5) [static]

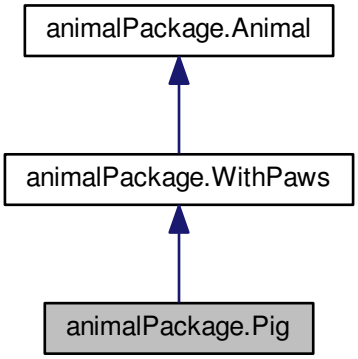
Increases the force of 5

The documentation for this class was generated from the following file:

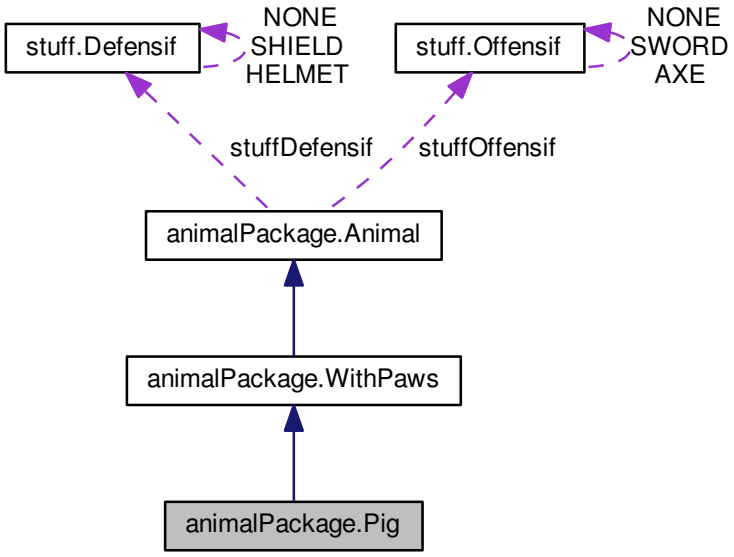
- src/stuff/Offensif.java

6.15 animalPackage.Pig Class Reference

Inheritance diagram for animalPackage.Pig:



Collaboration diagram for animalPackage.Pig:



Public Member Functions

- Pig (String newPseudo)
- Pig (String newPseudo, String newColor)
- void attack (Animal attackedAnimal)
- String specialAction (Animal attackedAnimal)
- void scream ()

Additional Inherited Members

6.15.1 Detailed Description

===== Class Pig =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.15.2 Constructor & Destructor Documentation

6.15.2.1 animalPackage.Pig.Pig (String *newPseudo*)

Constructor

Parameters

1	String = pig's Pseudo

6.15.2.2 animalPackage.Pig.Pig (String *newPseudo,* String *newColor*)

Constructor

Parameters

1	String = pig's Pseudo
1	String = pig's color

6.15.3 Member Function Documentation

6.15.3.1 void animalPackage.Pig.attack (Animal *attackedAnimal*)

attack : function which executes a basic attack

Parameters

Animal	attackedAnimal

6.15.3.2 void animalPackage.Pig.scream ( )

scream : function which makes the animal scream

6.15.3.3 String animalPackage.Pig.specialAction (Animal *attackedAnimal*)

specialAction : function which executes a special attack

Parameters

Animal	attackedAnimal

Returns

String

The documentation for this class was generated from the following file:

- src/animalPackage/Pig.java

6.16 playerPackage.Player Class Reference

Public Member Functions

- Player (int animalClass, String newPseudo, String animalColor, String spacecraftColor)
- Animal getAnimal ()
- Spacecraft getSpacecraft ()
- void setAnimal (Animal newAnimal)
- void setSpacecraft (Spacecraft newSpacecraft)

6.16.1 Detailed Description

===== Class Player =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.01, 10/2016

6.16.2 Constructor & Destructor Documentation

6.16.2.1 playerPackage.Player.Player (int *animalClass,* String *newPseudo,* String *animalColor,* String *spacecraftColor*)

Constructor with animal

Parameters

1	int animalClass
1	String newPseudo
1	String animalColor
1	String spacecraftColor

6.16.3 Member Function Documentation

6.16.3.1 Animal playerPackage.Player.getAnimal ( )

Get player's animal

Returns

1 Animal = player's animal

6.16.3.2 Spacecraft playerPackage.Player.getSpacecraft ( )

Get player's spacecraft

Returns

1 Spacecraft = player's spacecraft

6.16.3.3 void playerPackage.Player.setAnimal (Animal *newAnimal*)

Set player's animal

Parameters

1	Animal = newAnimal

6.16.3.4 void playerPackage.Player.setSpacecraft (Spacecraft *newSpacecraft*)

Set player's spacecraft

Parameters

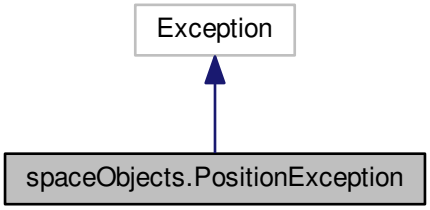
1	Spacecraft = newSpacecraft

The documentation for this class was generated from the following file:

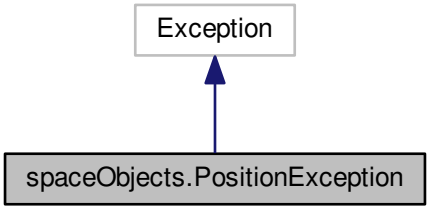
- src/playerPackage/Player.java

6.17 spaceObjects.PositionException Class Reference

Inheritance diagram for spaceObjects.PositionException:



Collaboration diagram for spaceObjects.PositionException:



6.17.1 Detailed Description

===== Class PositionException =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.01, 11/2016

The documentation for this class was generated from the following file:

- src/spaceObjects/PositionException.java

6.18 spaceObjects.PositionsCube Enum Reference

Public Attributes

- **NONE**
- **OOO**
- **OOI**
- **OIO**
- **IOO**
- **IIO**
- **OII**
- **IOI**

6.18.1 Detailed Description

===== Enumeration MeteoriteSize =====

enumeration of possible positions for the Spacecraft in the CubeEnvironment O = 0 and I = 1

Author

Vincent Reynaert, Nicolas Sobczak

Version

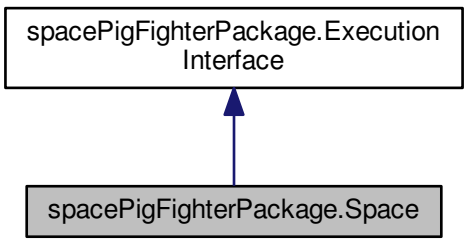
1.01, 10/2016

The documentation for this enum was generated from the following file:

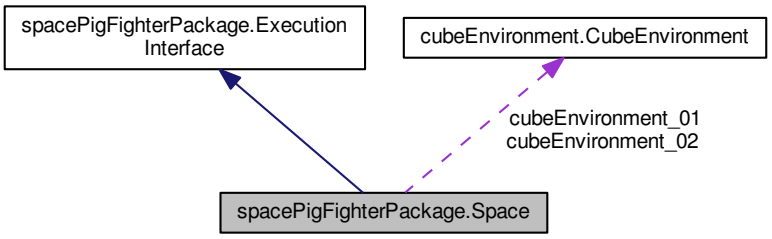
- src/spaceObjects/PositionsCube.java

6.19 spacePigFighterPackage.Space Class Reference

Inheritance diagram for spacePigFighterPackage.Space:



Collaboration diagram for spacePigFighterPackage.Space:



Public Member Functions

- Space (Player player_1, Player player_2)
- CubeEnvironment getCubeEnvironment01 ()
- CubeEnvironment getCubeEnvironment02 ()
- void setCubeEnvironment01 (CubeEnvironment new_cubeEnvironment_01)
- void ssetCubeEnvironment02 (CubeEnvironment new_cubeEnvironment_02)
- String run ()

Public Attributes

- CubeEnvironment **cubeEnvironment_01**
- CubeEnvironment **cubeEnvironment_02**

6.19.1 Detailed Description

===== Class Space =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.19.2 Constructor & Destructor Documentation

6.19.2.1 spacePigFighterPackage.Space.Space (Player *player_1,* Player *player_2*)

Constructor

Parameters

1	Player = player_1
1	Player = player_2

6.19.3 Member Function Documentation

6.19.3.1 CubeEnvironment spacePigFighterPackage.Space.getCubeEnvironment01 ( )

Get Space cubeEnvironment_01

Returns

CubeEnvironment cubeEnvironment_01

6.19.3.2 CubeEnvironment spacePigFighterPackage.Space.getCubeEnvironment02 ( )

Get Space cubeEnvironment_02

Returns

CubeEnvironment cubeEnvironment_02

6.19.3.3 String spacePigFighterPackage.Space.run ( )

run()

Implements spacePigFighterPackage.ExecutionInterface.

6.19.3.4 void spacePigFighterPackage.Space.setCubeEnvironment01 (CubeEnvironment *new_cubeEnvironment_01*)

Set Space cubeEnvironment_01

Parameters

CubeEnvironment	cubeEnvironment_01

6.19.3.5 void spacePigFighterPackage.Space.ssetCubeEnvironment02 (CubeEnvironment *new_cubeEnvironment_02*)

Set Space cubeEnvironment_02

Parameters

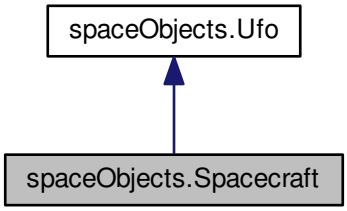
CubeEnvironment	cubeEnvironment_02

The documentation for this class was generated from the following file:

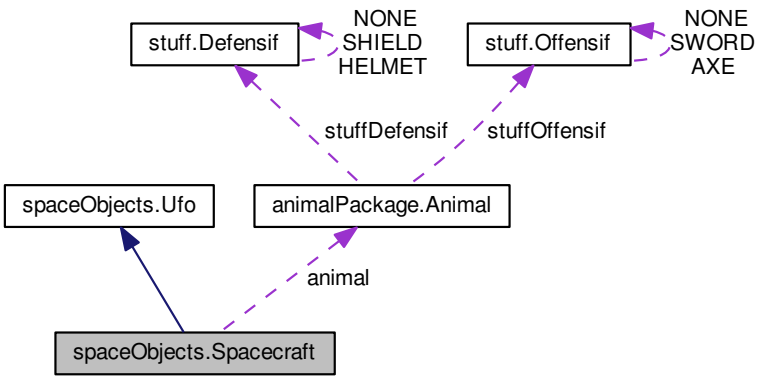
- src/spacePigFighterPackage/Space.java

6.20 spaceObjects.Spacecraft Class Reference

Inheritance diagram for spaceObjects.Spacecraft:



Collaboration diagram for spaceObjects.Spacecraft:



Public Member Functions

- Spacecraft ()
- Spacecraft (String colorName)
- Spacecraft (Animal myAnimal)
- Spacecraft (PositionsCube position)
- Spacecraft (String colorName, Animal myAnimal)
- Spacecraft (PositionsCube position, String colorName)
- Spacecraft (PositionsCube position, Animal myAnimal)
- Spacecraft (PositionsCube position, String colorName, Animal myAnimal)
- String getColor ()
- Animal getAnimal ()
- void setColor (String newColor)
- void setAnimal (Animal newAnimal)
- void beDamagedBy (MeteoriteSize meteoriteSize)

Public Attributes

- Animal **animal**

6.20.1 Detailed Description

===== Class Spacecraft =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.20.2 Constructor & Destructor Documentation

6.20.2.1 spaceObjects.Spacecraft.Spacecraft ( )

Constructor by default we have a Pig unnamed and a Spacecraft grey colored at the position OOO

6.20.2.2 spaceObjects.Spacecraft.Spacecraft (String *colorName*)

Constructor with selected color

Parameters

1	String = colorName

6.20.2.3 spaceObjects.Spacecraft.Spacecraft (Animal *myAnimal*)

Constructor with selected animal

Parameters

1	Animal = myAnimal

6.20.2.4 spaceObjects.Spacecraft.Spacecraft (PositionsCube *position*)

Constructor with selected location

Parameters

1	PositionsCube = position

6.20.2.5 spaceObjects.Spacecraft.Spacecraft (String *colorName,* Animal *myAnimal*)

Constructor with selected color and animal

Parameters

1	String = colorName
1	Animal = myAnimal

6.20.2.6 spaceObjects.Spacecraft.Spacecraft (PositionsCube *position,* String *colorName*)

Constructor with selected location and color

Parameters

1	PositionsCube = position
1	String = colorName

6.20.2.7 spaceObjects.Spacecraft.Spacecraft (PositionsCube *position,* Animal *myAnimal*)

Constructor with selected location and animal

Parameters

1	PositionsCube = position
1	Animal = myAnimal

6.20.2.8 spaceObjects.Spacecraft.Spacecraft (PositionsCube *position,* String *colorName,* Animal *myAnimal*)

Constructor. with selected location, color and animal

Parameters

1	PositionsCube = position
1	String = colorName
1	Animal = myAnimal

6.20.3 Member Function Documentation

6.20.3.1 void spaceObjects.Spacecraft.beDamagedBy (MeteoriteSize *meteoriteSize*)

The Animal will receive damages proportional to the meteoriteSize

Parameters

MeteoriteSize	= meteoriteSize

6.20.3.2 Animal spaceObjects.Spacecraft.getAnimal ( )

Get Spacecraft animal

Returns

1 Animal = animal

6.20.3.3 String spaceObjects.Spacecraft.getColor ( )

Get Spacecraft color

Returns

1 String = color

6.20.3.4 void spaceObjects.Spacecraft.setAnimal (Animal *newAnimal*)

Set Spacecraft animal

Parameters

1	Animal = newAnimal

6.20.3.5 void spaceObjects.Spacecraft.setColor (String *newColor*)

Set Spacecraft color

Parameters

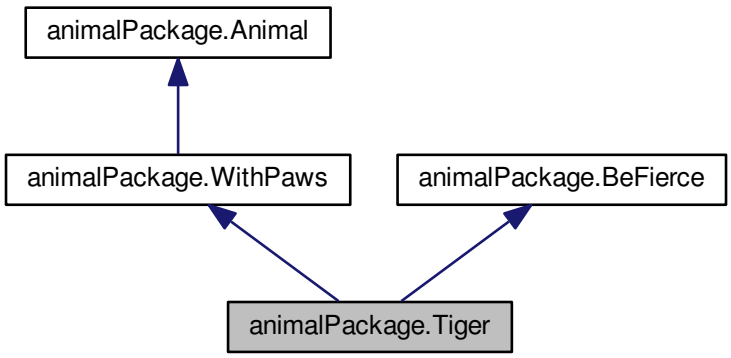
1	String = newColor

The documentation for this class was generated from the following file:

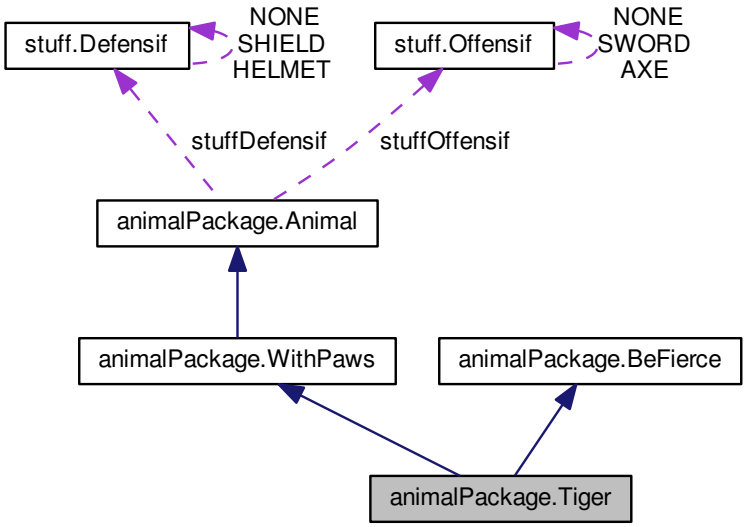
- src/spaceObjects/Spacecraft.java

6.21 animalPackage.Tiger Class Reference

Inheritance diagram for animalPackage.Tiger:



Collaboration diagram for animalPackage.Tiger:



Public Member Functions

- Tiger (String newPseudo)
- Tiger (String newPseudo, String newColor)
- void attack (Animal attackedAnimal)
- String specialAction (Animal attackedAnimal)
- void scream ()
- String beFierce ()

Additional Inherited Members

6.21.1 Detailed Description

===== Class Tiger =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.21.2 Constructor & Destructor Documentation

6.21.2.1 animalPackage.Tiger.Tiger (String *newPseudo*)

Constructor

Parameters

1	String = tiger's Pseudo

6.21.2.2 animalPackage.Tiger.Tiger (String *newPseudo,* String *newColor*)

Constructor

Parameters

1	String = tiger's Pseudo
1	String = tiger's color

6.21.3 Member Function Documentation

6.21.3.1 void animalPackage.Tiger.attack (Animal *attackedAnimal*)

attack : function which executes a basic attack

Parameters

Animal	attackedAnimal

6.21.3.2 String animalPackage.Tiger.beFierce ( )

beFierce : function which return an adjective to describe behavior

Returns

1 String = an adjective

Implements animalPackage.BeFierce.

6.21.3.3 void animalPackage.Tiger.scream ( )

scream : function which makes the animal scream

6.21.3.4 String animalPackage.Tiger.specialAction (Animal *attackedAnimal*)

specialAction : function which executes a special attack

Parameters

Animal	attackedAnimal

Returns

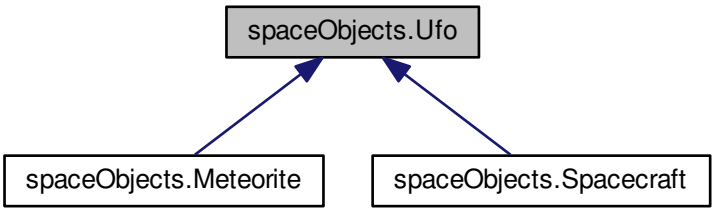
String

The documentation for this class was generated from the following file:

- src/animalPackage/Tiger.java

6.22 spaceObjects.Ufo Class Reference

Inheritance diagram for spaceObjects.Ufo:



Public Member Functions

- Ufo ()
- Ufo (PositionsCube position)
- PositionsCube getLocation ()
- void setLocation (PositionsCube position)
- void setLocation (int position) throws PositionException

6.22.1 Detailed Description

===== Class Ufo =====

useful to have position in the cube

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.02, 10/2016

6.22.2 Constructor & Destructor Documentation

6.22.2.1 spaceObjects.Ufo.Ufo ( )

Constructor. Set location by default to (0,0,0)

6.22.2.2 spaceObjects.Ufo.Ufo (PositionsCube *position*)

Constructor. with selected position

Parameters

1	PositionsCube = position

6.22.3 Member Function Documentation

6.22.3.1 PositionsCube spaceObjects.Ufo.getLocation ( )

Get the Ufo location

Returns

1 Positions = location

6.22.3.2 void spaceObjects.Ufo.setLocation (PositionsCube *position*)

Set the Ufo location

Parameters

1	PostionsCube = position

6.22.3.3 void spaceObjects.Ufo.setLocation (int *position*) throws PositionException

Set the Ufo location

Parameters

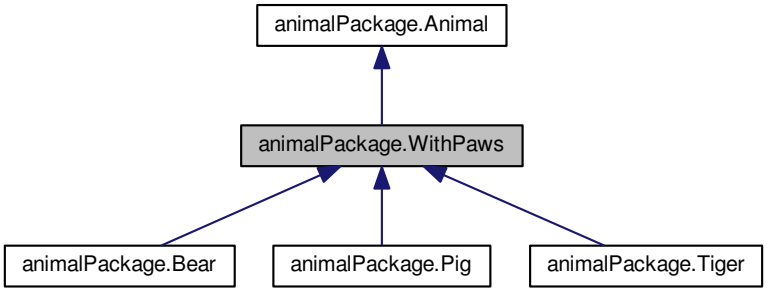
1	int = position

The documentation for this class was generated from the following file:

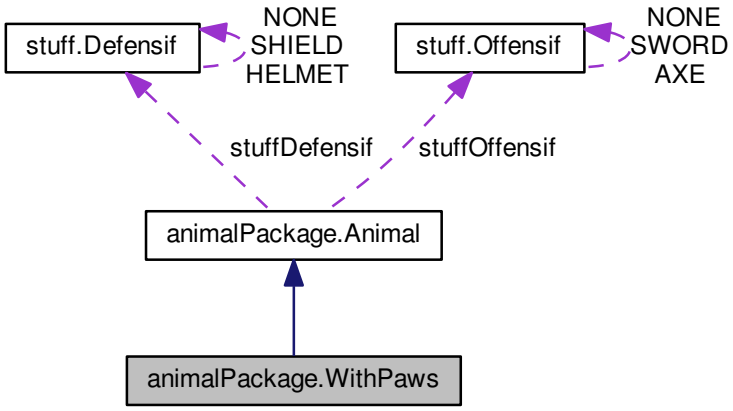
- src/spaceObjects/Ufo.java

6.23 animalPackage.WithPaws Class Reference

Inheritance diagram for animalPackage.WithPaws:



Collaboration diagram for animalPackage.WithPaws:



Public Member Functions

- WithPaws (String newPseudo)
- WithPaws (String newPseudo, String newColor)
- void attack (Animal attackedAnimal)
- String specialAction (Animal attackedAnimal)

Additional Inherited Members

6.23.1 Detailed Description

===== Abstract Class WithPaws =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.23.2 Constructor & Destructor Documentation

6.23.2.1 animalPackage.WithPaws.WithPaws (String *newPseudo*)

Constructor

Parameters

1	String = animal's Pseudo

6.23.2.2 animalPackage.WithPaws.WithPaws (String *newPseudo,* String *newColor*)

Constructor

Parameters

1	String = animal's Pseudo
1	String = animal's color

6.23.3 Member Function Documentation

6.23.3.1 void animalPackage.WithPaws.attack (Animal *attackedAnimal*)

attack : function which executes a basic attack

Parameters

Animal	attackedAnimal

6.23.3.2 String animalPackage.WithPaws.specialAction (Animal *attackedAnimal*)

attack : function which executes a special attack

Parameters

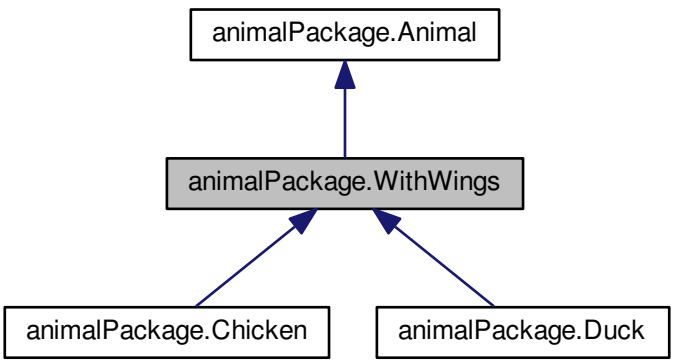
Animal	attackedAnimal

The documentation for this class was generated from the following file:

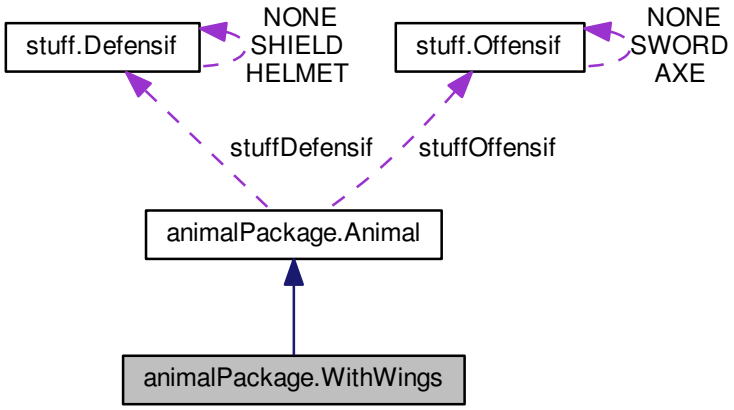
- src/animalPackage/WithPaws.java

6.24 animalPackage.WithWings Class Reference

Inheritance diagram for animalPackage.WithWings:



Collaboration diagram for animalPackage.WithWings:



Public Member Functions

- WithWings (String newPseudo)
- WithWings (String newPseudo, String newColor)
- void attack (Animal attackedAnimal)
- String specialAction (Animal attackedAnimal)

Additional Inherited Members

6.24.1 Detailed Description

===== Class WithWings =====

Author

Vincent Reynaert, Nicolas Sobczak

Version

1.03, 11/2016

6.24.2 Constructor & Destructor Documentation

6.24.2.1 animalPackage.WithWings.WithWings (String *newPseudo*)

Constructor

Parameters

1	String = animal's Pseudo

6.24.2.2 animalPackage.WithWings.WithWings (String *newPseudo,* String *newColor*)

Constructor

Parameters

1	String = animal's Pseudo
1	String = animal's color

6.24.3 Member Function Documentation

6.24.3.1 void animalPackage.WithWings.attack (Animal *attackedAnimal*)

attack : function which executes a basic attack

Parameters

Animal	attackedAnimal

6.24.3.2 String animalPackage.WithWings.specialAction (Animal *attackedAnimal*)

attack : function which executes a special attack

Parameters

Animal	attackedAnimal

The documentation for this class was generated from the following file:

- src/animalPackage/WithWings.java

Index