

## ANIMAL

Sugeneruota Doxygen 1.13.2



<b>1 Hierarchijos Indeksas</b>	<b>1</b>
1.1 Klasių hierarchija	1
<b>2 Klasės Indeksas</b>	<b>3</b>
2.1 Klasės	3
<b>3 Failo Indeksas</b>	<b>5</b>
3.1 Failai	5
<b>4 Klasės Dokumentacija</b>	<b>7</b>
4.1 Animal Klasė	7
4.1.1 Smulkus aprašymas	8
4.1.2 Konstruktoriaus ir Destruktoriaus Dokumentacija	8
4.1.2.1 Animal() [1/2]	8
4.1.2.2 Animal() [2/2]	8
4.1.3 Metodų Dokumentacija	8
4.1.3.1 getID()	8
4.1.3.2 getName()	9
4.1.3.3 getX()	9
4.1.3.4 getY()	9
4.1.3.5 move()	9
4.1.3.6 operator=()	9
4.1.3.7 setMovementStrategy()	10
4.1.4 Draugiškų Ir Susijusių Funkcijų Dokumentacija	10
4.1.4.1 operator<<	10
4.1.4.2 operator>>	10
4.2 AnimalException Klasė	11
4.2.1 Konstruktoriaus ir Destruktoriaus Dokumentacija	12
4.2.1.1 AnimalException()	12
4.2.2 Metodų Dokumentacija	12
4.2.2.1 what()	12
4.3 DiagonalMovement Klasė	13
4.3.1 Smulkus aprašymas	14
4.3.2 Konstruktoriaus ir Destruktoriaus Dokumentacija	14
4.3.2.1 DiagonalMovement()	14
4.3.3 Metodų Dokumentacija	14
4.3.3.1 move()	14
4.3.3.2 typeTag()	15
4.4 MovementStrategy Klasė	15
4.4.1 Smulkus aprašymas	16
4.4.2 Konstruktoriaus ir Destruktoriaus Dokumentacija	16
4.4.2.1 ~MovementStrategy()	16
4.4.3 Metodų Dokumentacija	16

4.4.3.1 deserialize()	16
4.4.3.2 move()	16
4.4.3.3 serialize()	17
4.4.3.4 typeTag()	17
4.5 NormalMovement Klasė	18
4.5.1 Smulkus aprašymas	19
4.5.2 Konstruktoriaus ir Destruktoriaus Dokumentacija	19
4.5.2.1 NormalMovement()	19
4.5.3 Metodų Dokumentacija	19
4.5.3.1 move()	19
4.5.3.2 typeTag()	20
4.6 SpeedBasedMovement Klasė	20
4.6.1 Smulkus aprašymas	21
4.6.2 Konstruktoriaus ir Destruktoriaus Dokumentacija	21
4.6.2.1 SpeedBasedMovement()	21
4.6.3 Metodų Dokumentacija	22
4.6.3.1 deserialize()	22
4.6.3.2 getSpeedMultiplier()	22
4.6.3.3 serialize()	22
4.6.3.4 setSpeedMultiplier()	22
4.6.4 Atributų Dokumentacija	23
4.6.4.1 speedMultiplier_	23
<b>5 Failo Dokumentacija</b>	<b>25</b>
5.1 src/animal/Animal.cpp Failo Nuoroda	25
5.1.1 Smulkus aprašymas	25
5.1.2 Funkcijos Dokumentacija	26
5.1.2.1 operator<<()	26
5.1.2.2 operator>>()	26
5.2 src/animal/Animal.h Failo Nuoroda	27
5.2.1 Smulkus aprašymas	27
5.3 Animal.h	28
5.4 src/animal/exception/AnimalException.cpp Failo Nuoroda	28
5.4.1 Smulkus aprašymas	29
5.5 src/animal/exception/AnimalException.h Failo Nuoroda	29
5.5.1 Smulkus aprašymas	30
5.6 AnimalException.h	31
5.7 src/main.cpp Failo Nuoroda	31
5.7.1 Smulkus aprašymas	32
5.7.2 Funkcijos Dokumentacija	32
5.7.2.1 main()	32
5.8 src/movement/DiagonalMovement.h Failo Nuoroda	32

---

5.8.1 Smulkus aprašymas . . . . .	33
5.9 DiagonalMovement.h . . . . .	33
5.10 src/movement/MovementStrategy.h Failo Nuoroda . . . . .	34
5.10.1 Smulkus aprašymas . . . . .	35
5.11 MovementStrategy.h . . . . .	35
5.12 src/movement/NormalMovement.h Failo Nuoroda . . . . .	36
5.12.1 Smulkus aprašymas . . . . .	37
5.13 NormalMovement.h . . . . .	37
5.14 src/movement/SpeedBasedMovement.h Failo Nuoroda . . . . .	38
5.14.1 Smulkus aprašymas . . . . .	38
5.15 SpeedBasedMovement.h . . . . .	39
<b>Rodyklė</b>	<b>41</b>



## skyrius 1

# Hierarchijos Indeksas

### 1.1 Klasių hierarchija

Šis paveldėjimo sąrašas yra beveik surikiuotas abėcėlės tvarka:

Animal . . . . .	7
std::exception	
AnimalException . . . . .	11
MovementStrategy . . . . .	15
SpeedBasedMovement . . . . .	20
DiagonalMovement . . . . .	13
NormalMovement . . . . .	18





## skyrius 2

# Klasės Indeksas

### 2.1 Klasės

Klasės, struktūros, sąjungos ir sąsajos su trumpais aprašymais:

<a href="#">Animal</a>	Represents an <a href="#">Animal</a> with replaceable movement . . . . .	7
<a href="#">AnimalException</a>	. . . . .	11
<a href="#">DiagonalMovement</a>	Allows diagonal moves at constant cost . . . . .	13
<a href="#">MovementStrategy</a>	Abstract interface for animal movement behavior . . . . .	15
<a href="#">NormalMovement</a>	Simple axis-aligned movement . . . . .	18
<a href="#">SpeedBasedMovement</a>	Base class for strategies that share a speedMultiplier . . . . .	20



## skyrius 3

# Failo Indeksas

### 3.1 Failai

Visų failų sąrašas su trumpais aprašymais:

src/main.cpp . . . . .	31
src/animal/Animal.cpp . . . . .	25
src/animal/Animal.h . . . . .	27
src/animal/exception/AnimalException.cpp . . . . .	28
src/animal/exception/AnimalException.h . . . . .	29
src/movement/DiagonalMovement.h . . . . .	32
src/movement/MovementStrategy.h . . . . .	34
src/movement/NormalMovement.h . . . . .	36
src/movement/SpeedBasedMovement.h . . . . .	38



# skyrius 4

## Klasės Dokumentacija

### 4.1 Animal Klasė

Represents an [Animal](#) with replaceable movement.

```
#include <Animal.h>
```

#### Vieši Metodai

- [Animal](#) (const std::string &name, std::unique\_ptr< [MovementStrategy](#) > strat)  
*Construct a new [Animal](#) object.*
- [Animal](#) (const [Animal](#) &o)  
*Copy preserves runtime strategy.*
- [Animal](#) & [operator=](#) (const [Animal](#) &o)  
*Copy assignment operator for the [Animal](#) class.*
- void [move](#) (int dx, int dy)  
*Move by (dx,dy) via current strategy.*
- void [setMovementStrategy](#) (std::unique\_ptr< [MovementStrategy](#) > strat)  
*Set the Movement Strategy object.*
- int [getID](#) () const  
*get Identifier of the animal*
- int [getX](#) () const  
*get X coordinate of the animal*
- int [getY](#) () const  
*get Y coordinate of the animal*
- std::string [getName](#) () const  
*Get the Name object.*

#### Draugai

- std::ostream & [operator<<](#) (std::ostream &os, const [Animal](#) &a)  
*Binary serialization (writes x, y, name, then strategy tag + data).*
- std::istream & [operator>>](#) (std::istream &is, [Animal](#) &a)  
*Binary deserialization (reads x, y, name, then strategy tag + data).*

### 4.1.1 Smulkus aprašymas

Represents an [Animal](#) with replaceable movement.

### 4.1.2 Konstruktoriaus ir Destruktoriaus Dokumentacija

#### 4.1.2.1 Animal() [1/2]

```
Animal::Animal (
    const std::string & name,
    std::unique_ptr< MovementStrategy > strat)
```

Construct a new [Animal](#) object.

##### Parametrai

<i>name</i>	animal name
<i>strat</i>	initial movement strategy (default = <a href="#">NormalMovement</a> )

#### 4.1.2.2 Animal() [2/2]

```
Animal::Animal (
    const Animal & o)
```

Copy preserves runtime strategy.

##### Parametrai

<i>o</i>	<a href="#">Animal</a> to copy from
----------	-------------------------------------

### 4.1.3 Metodų Dokumentacija

#### 4.1.3.1 getID()

```
int Animal::getID () const
```

get Identifier of the animal

##### Gražina

int

#### 4.1.3.2 getName()

```
std::string Animal::getName () const
```

Get the Name object.

Gražina

std::string

#### 4.1.3.3 getX()

```
int Animal::getX () const
```

get X coordinate of the animal

Gražina

int

#### 4.1.3.4 getY()

```
int Animal::getY () const
```

get Y coordinate of the animal

Gražina

int

#### 4.1.3.5 move()

```
void Animal::move (
    int dx,
    int dy)
```

Move by (dx,dy) via current strategy.

Parametrai

<i>dx</i>	delta x
<i>dy</i>	delta y

#### 4.1.3.6 operator=()

```
Animal & Animal::operator= (
    const Animal & o)
```

Copy assignment operator for the [Animal](#) class.

## Parametrai

<i>o</i>	
----------	--

## Gražina

[Animal](#)&

## 4.1.3.7 setMovementStrategy()

```
void Animal::setMovementStrategy (
    std::unique_ptr< MovementStrategy > strat)
```

Set the Movement Strategy object.

## Parametrai

<i>strat</i>	strategy to set
--------------	-----------------

## 4.1.4 Draugiškų Ir Susijusių Funkcijų Dokumentacija

## 4.1.4.1 operator&lt;&lt;

```
std::ostream & operator<< (
    std::ostream & os,
    const Animal & a) [friend]
```

Binary serialization (writes x, y, name, then strategy tag + data).

## Parametrai

<i>os</i>	output stream
<i>a</i>	animal to write

## Gražina

std::ostream&amp;

## 4.1.4.2 operator&gt;&gt;

```
std::istream & operator>> (
    std::istream & is,
    Animal & a) [friend]
```

Binary deserialization (reads x, y, name, then strategy tag + data).



## Parametrai

<i>is</i>	input stream
<i>a</i>	animal to read into

## Gražina

std::istream&

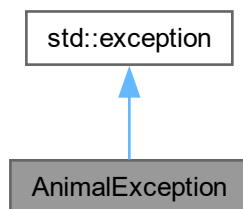
Dokumentacija šiai klasei sugeneruota iš šių failų:

- [src/animal/Animal.h](#)
- [src/animal/Animal.cpp](#)

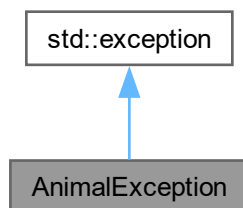
## 4.2 AnimalException Klasė

```
#include <AnimalException.h>
```

Paveldimumo diagrama AnimalException:



Bendradarbiavimo diagrama AnimalException:



## Vieši Metodai

- [AnimalException](#) (const std::string &msg)  
*Construct a new [Animal](#) Exception object.*
- const char \* [what](#) () const noexcept override  
*just a wrapper for std::exception::what()*

## 4.2.1 Konstruktoriaus ir Destruktoriaus Dokumentacija

### 4.2.1.1 AnimalException()

```
AnimalException::AnimalException (  
    const std::string & msg) [explicit]
```

Construct a new [Animal](#) Exception object.

#### Parametrai

<i>msg</i>	message to be displayed
------------	-------------------------

## 4.2.2 Metodų Dokumentacija

### 4.2.2.1 what()

```
const char * AnimalException::what () const [override], [noexcept]
```

just a wrapper for std::exception::what()

#### Gražina

const char\*

Dokumentacija šiai klasei sugeneruota iš šių failų:

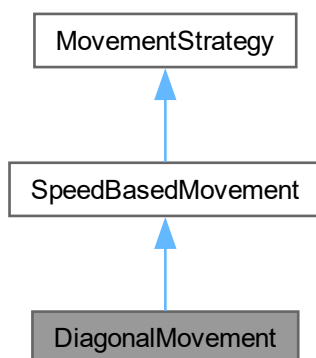
- src/animal/exception/[AnimalException.h](#)
- src/animal/exception/[AnimalException.cpp](#)

## 4.3 DiagonalMovement Klasė

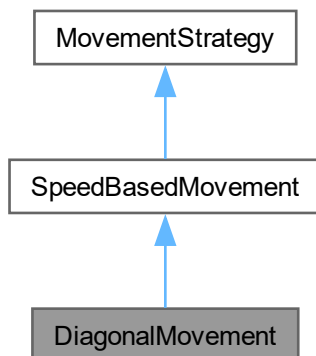
Allows diagonal moves at constant cost.

```
#include <DiagonalMovement.h>
```

Paveldimumo diagrama DiagonalMovement:



Bendradarbiavimo diagrama DiagonalMovement:



### Vieši Metodai

- **DiagonalMovement** (double mult=1.0)  
*Construct a new Diagonal Movement object.*
- void **move** (int &x, int &y, int dx, int dy) override  
*Move the animal by (dx,dy), updating x,y in place.*
- char **typeTag** () const override  
*Return a tag to identify concrete type in the stream.*

## Vieši Metodai inherited from [SpeedBasedMovement](#)

- [SpeedBasedMovement](#) (double mult=1.0)  
*Construct a new Speed Based Movement object.*
- void [serialize](#) (std::ostream &os) const override  
*Serialize strategy-specific data to a binary stream.*
- void [deserialize](#) (std::istream &is) override  
*Deserialize strategy-specific data from a binary stream.*
- double [getSpeedMultiplier](#) () const  
*Get the speed multiplier.*
- void [setSpeedMultiplier](#) (double m)  
*Set the speed multiplier.*

## Vieši Metodai inherited from [MovementStrategy](#)

- virtual [~MovementStrategy](#) ()=default  
*Destroy the Movement Strategy object.*

## Additional Inherited Members

## Apsaugoti Atributai inherited from [SpeedBasedMovement](#)

- double [speedMultiplier\\_](#)

### 4.3.1 Smulkus aprašymas

Allows diagonal moves at constant cost.

### 4.3.2 Konstruktorius ir Destruktorius Dokumentacija

#### 4.3.2.1 DiagonalMovement()

```
DiagonalMovement::DiagonalMovement (
    double mult = 1.0) [inline], [explicit]
```

Construct a new Diagonal Movement object.

#### Parametrai

<i>mult</i>	initial speed multiplier, default = 1.0
-------------	---

### 4.3.3 Metodų Dokumentacija

#### 4.3.3.1 move()

```
void DiagonalMovement::move (
    int & x,
    int & y,
    int dx,
    int dy) [inline], [override], [virtual]
```

Move the animal by (dx,dy), updating x,y in place.

## Parametrai

<i>x</i>	int& x coordinate of the animal
<i>y</i>	int& y coordinate of the animal
<i>dx</i>	int x displacement
<i>dy</i>	int y displacement

Realizuoja [MovementStrategy](#).

## 4.3.3.2 typeTag()

```
char DiagonalMovement::typeTag () const [inline], [override], [virtual]
```

Return a tag to identify concrete type in the stream.

## Gražina

char

Realizuoja [MovementStrategy](#).

Dokumentacija šiai klasei sugeneruota iš šio failo:

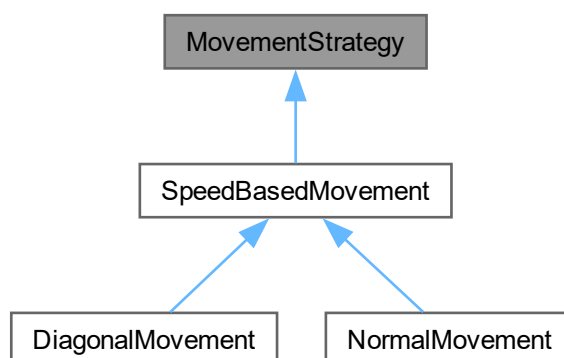
- src/movement/[DiagonalMovement.h](#)

## 4.4 MovementStrategy Klasė

Abstract interface for animal movement behavior.

```
#include <MovementStrategy.h>
```

Paveldimumo diagrama MovementStrategy:



## Vieši Metodai

- virtual `~MovementStrategy()`=default  
*Destroy the Movement Strategy object.*
- virtual void `move` (int &x, int &y, int dx, int dy)=0  
*Move the animal by (dx,dy), updating x,y in place.*
- virtual void `serialize` (std::ostream &os) const =0  
*Serialize strategy-specific data to a binary stream.*
- virtual void `deserialize` (std::istream &is)=0  
*Deserialize strategy-specific data from a binary stream.*
- virtual char `typeTag` () const =0  
*Return a tag to identify concrete type in the stream.*

### 4.4.1 Smulkus aprašymas

Abstract interface for animal movement behavior.

### 4.4.2 Konstruktoriaus ir Destruktoriaus Dokumentacija

#### 4.4.2.1 ~MovementStrategy()

```
virtual MovementStrategy::~MovementStrategy () [virtual], [default]
```

Destroy the Movement Strategy object.

### 4.4.3 Metodų Dokumentacija

#### 4.4.3.1 deserialize()

```
virtual void MovementStrategy::deserialize (
    std::istream & is) [pure virtual]
```

Deserialize strategy-specific data from a binary stream.

Parametrai

<i>is</i>	input stream to read from
-----------	---------------------------

Realizuota [SpeedBasedMovement](#).

#### 4.4.3.2 move()

```
virtual void MovementStrategy::move (
    int & x,
    int & y,
    int dx,
    int dy) [pure virtual]
```

Move the animal by (dx,dy), updating x,y in place.

## Parametrai

<i>x</i>	int& x coordinate of the animal
<i>y</i>	int& y coordinate of the animal
<i>dx</i>	int x displacement
<i>dy</i>	int y displacement

Realizuota [DiagonalMovement](#), ir [NormalMovement](#).

## 4.4.3.3 serialize()

```
virtual void MovementStrategy::serialize (  
    std::ostream & os) const [pure virtual]
```

Serialize strategy-specific data to a binary stream.

## Parametrai

<i>os</i>	output stream to write to
-----------	---------------------------

Realizuota [SpeedBasedMovement](#).

## 4.4.3.4 typeTag()

```
virtual char MovementStrategy::typeTag () const [pure virtual]
```

Return a tag to identify concrete type in the stream.

## Gražina

char

Realizuota [DiagonalMovement](#), ir [NormalMovement](#).

Dokumentacija šiai klasei sugeneruota iš šio failo:

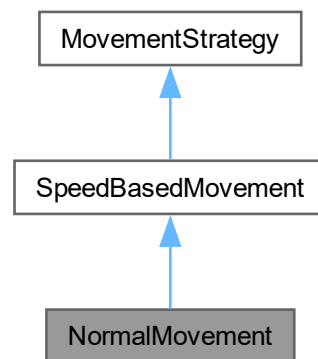
- src/movement/[MovementStrategy.h](#)

## 4.5 NormalMovement Klasė

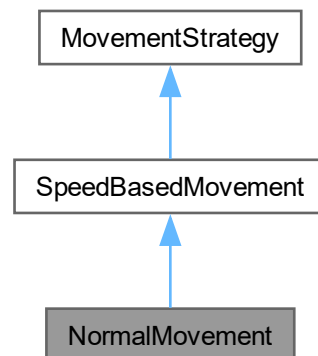
Simple axis-aligned movement.

```
#include <NormalMovement.h>
```

Paveldimumo diagrama NormalMovement:



Bendradarbiavimo diagrama NormalMovement:



### Vieši Metodai

- **NormalMovement** (double mult=1.0)  
*Construct a new Normal Movement object.*
- void **move** (int &x, int &y, int dx, int dy) override  
*Move the animal by (dx,dy), updating x,y in place.*
- char **typeTag** () const override  
*Return a tag to identify concrete type in the stream.*



### Vieši Metodai inherited from SpeedBasedMovement

- `SpeedBasedMovement` (double mult=1.0)  
*Construct a new Speed Based Movement object.*
- void `serialize` (std::ostream &os) const override  
*Serialize strategy-specific data to a binary stream.*
- void `deserialize` (std::istream &is) override  
*Deserialize strategy-specific data from a binary stream.*
- double `getSpeedMultiplier` () const  
*Get the speed multiplier.*
- void `setSpeedMultiplier` (double m)  
*Set the speed multiplier.*

### Vieši Metodai inherited from MovementStrategy

- virtual `~MovementStrategy` ()=default  
*Destroy the Movement Strategy object.*

### Additional Inherited Members

### Apsaugoti Atributai inherited from SpeedBasedMovement

- double `speedMultiplier_`

## 4.5.1 Smulkus aprašymas

Simple axis-aligned movement.

## 4.5.2 Konstruktoriaus ir Destruktoriaus Dokumentacija

### 4.5.2.1 NormalMovement()

```
NormalMovement::NormalMovement (
    double mult = 1.0) [inline], [explicit]
```

Construct a new Normal Movement object.

#### Parametrai

<i>mult</i>	initial speed multiplier, default = 1.0
-------------	---

## 4.5.3 Metodų Dokumentacija

### 4.5.3.1 move()

```
void NormalMovement::move (
    int & x,
    int & y,
    int dx,
    int dy) [inline], [override], [virtual]
```

Move the animal by (dx,dy), updating x,y in place.

**Parametrai**

<i>x</i>	int& x coordinate of the animal
<i>y</i>	int& y coordinate of the animal
<i>dx</i>	int x displacement
<i>dy</i>	int y displacement

Realizuoja [MovementStrategy](#).

**4.5.3.2 typeTag()**

```
char NormalMovement::typeTag () const [inline], [override], [virtual]
```

Return a tag to identify concrete type in the stream.

**Gražina**

char

Realizuoja [MovementStrategy](#).

Dokumentacija šiai klasei sugeneruota iš šio failo:

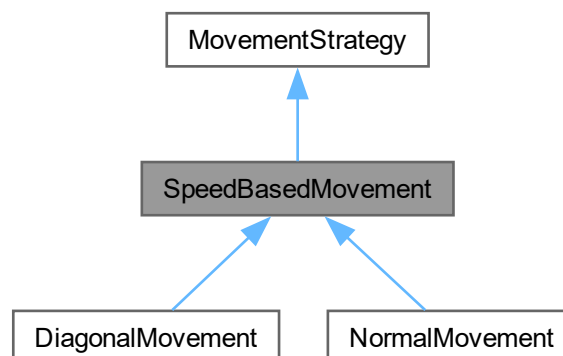
- src/movement/[NormalMovement.h](#)

**4.6 SpeedBasedMovement Klasė**

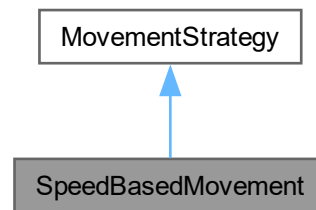
Base class for strategies that share a speedMultiplier.

```
#include <SpeedBasedMovement.h>
```

Paveldimumo diagrama SpeedBasedMovement:



Bendradarbiavimo diagrama SpeedBasedMovement:



### Vieši Metodai

- `SpeedBasedMovement` (double mult=1.0)  
*Construct a new Speed Based Movement object.*
- void `serialize` (std::ostream &os) const override  
*Serialize strategy-specific data to a binary stream.*
- void `deserialize` (std::istream &is) override  
*Deserialize strategy-specific data from a binary stream.*
- double `getSpeedMultiplier` () const  
*Get the speed multiplier.*
- void `setSpeedMultiplier` (double m)  
*Set the speed multiplier.*

### Vieši Metodai inherited from `MovementStrategy`

- virtual `~MovementStrategy` ()=default  
*Destroy the Movement Strategy object.*
- virtual void `move` (int &x, int &y, int dx, int dy)=0  
*Move the animal by (dx,dy), updating x,y in place.*
- virtual char `typeTag` () const =0  
*Return a tag to identify concrete type in the stream.*

### Apsaugoti Atributai

- double `speedMultiplier_`

## 4.6.1 Smulkus aprašymas

Base class for strategies that share a speedMultiplier.

## 4.6.2 Konstruktoriaus ir Destruktoriaus Dokumentacija

### 4.6.2.1 SpeedBasedMovement()

```
SpeedBasedMovement::SpeedBasedMovement (
    double mult = 1.0) [inline], [explicit]
```

Construct a new Speed Based Movement object.

## Parametrai

<i>mult</i>	speed multiplier, default = 1.0
-------------	---------------------------------

### 4.6.3 Metodų Dokumentacija

#### 4.6.3.1 deserialize()

```
void SpeedBasedMovement::deserialize (  
    std::istream & is) [inline], [override], [virtual]
```

Deserialize strategy-specific data from a binary stream.

## Parametrai

<i>is</i>	input stream to read from
-----------	---------------------------

Realizuoja [MovementStrategy](#).

#### 4.6.3.2 getSpeedMultiplier()

```
double SpeedBasedMovement::getSpeedMultiplier () const [inline]
```

Get the speed multiplier.

## Gražina

double

#### 4.6.3.3 serialize()

```
void SpeedBasedMovement::serialize (  
    std::ostream & os) const [inline], [override], [virtual]
```

Serialize strategy-specific data to a binary stream.

## Parametrai

<i>os</i>	output stream to write to
-----------	---------------------------

Realizuoja [MovementStrategy](#).

#### 4.6.3.4 setSpeedMultiplier()

```
void SpeedBasedMovement::setSpeedMultiplier (  
    double m) [inline]
```

Set the speed multiplier.

Parametrai

<i>m</i>	
----------	--

## 4.6.4 Atributų Dokumentacija

### 4.6.4.1 speedMultiplier\_

```
double SpeedBasedMovement::speedMultiplier_ [protected]
```

Dokumentacija šiai klasei sugeneruota iš šio failo:

- src/movement/[SpeedBasedMovement.h](#)



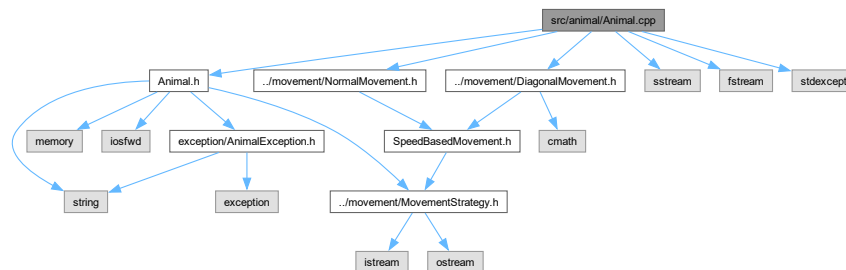
## skyrius 5

# Failo Dokumentacija

### 5.1 src/animal/Animal.cpp Failo Nuoroda

```
#include "Animal.h"
#include "../movement/NormalMovement.h"
#include "../movement/DiagonalMovement.h"
#include <sstream>
#include <fstream>
#include <stdexcept>
```

Įtraukimo priklausomybių diagrama Animal.cpp:



#### Funkcijos

- `std::ostream & operator<< (std::ostream &os, const Animal &a)`
- `std::istream & operator>> (std::istream &is, Animal &a)`

#### 5.1.1 Smulkus aprašymas

##### Autorius

Rokas Braidokas ( [rokasbraidokas@gmail.com](mailto:rokasbraidokas@gmail.com) )

**Versija**

0.1

**Data**

2025-05-06

**Copyright**

Copyright (c) 2025

## 5.1.2 Funkcijos Dokumentacija

### 5.1.2.1 operator<<()

```
std::ostream & operator<< (  
    std::ostream & os,  
    const Animal & a)
```

**Parametrai**

<i>os</i>	output stream
<i>a</i>	animal to write

**Gražina**

std::ostream&amp;

### 5.1.2.2 operator>>()

```
std::istream & operator>> (  
    std::istream & is,  
    Animal & a)
```

**Parametrai**

<i>is</i>	input stream
<i>a</i>	animal to read into

**Gražina**

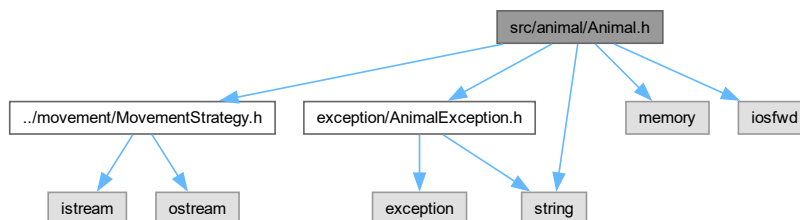
std::istream&amp;



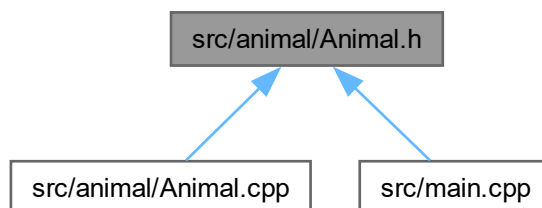
## 5.2 src/animal/Animal.h Failo Nuoroda

```
#include "../movement/MovementStrategy.h"
#include "exception/AnimalException.h"
#include <memory>
#include <string>
#include <iosfwd>
```

Įtraukimo priklausomybių diagrama Animal.h:



Šis grafas rodo, kuris failas tiesiogiai ar netiesiogiai įtraukia šį failą:



### Klasės

- class [Animal](#)

*Represents an [Animal](#) with replaceable movement.*

### 5.2.1 Smulkus aprašymas

#### Autorius

Rokas Braidokas ( [rokasbraidokas@gmail.com](mailto:rokasbraidokas@gmail.com) )

#### Versija

0.1

**Data**

2025-05-06

**Copyright**

Copyright (c) 2025

## 5.3 Animal.h

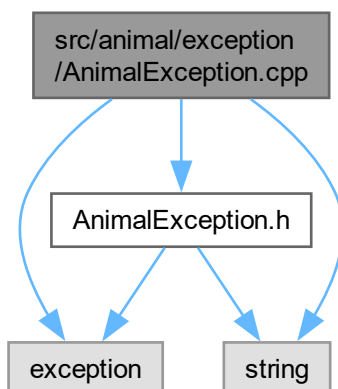
[Eiti į šio failo dokumentaciją.](#)

```
00001
00011 #ifndef ANIMAL_H
00012 #define ANIMAL_H
00013
00014 #include "../movement/MovementStrategy.h"
00015 #include "exception/AnimalException.h"
00016 #include <memory>
00017 #include <string>
00018 #include <iosfwd>
00019
00024 class Animal {
00025     const int ID_;
00026     int x_, y_;
00027     std::string name_;
00028     std::unique_ptr<MovementStrategy> mover_;
00029
00030     static int nextID_;
00031 public:
00038     Animal(const std::string& name,
00039           std::unique_ptr<MovementStrategy> strat);
00040
00046     Animal(const Animal& o);
00053     Animal& operator=(const Animal& o);
00054
00061     void move(int dx, int dy);
00062
00068     void setMovementStrategy(std::unique_ptr<MovementStrategy> strat);
00069
00075     int getID() const;
00081     int getX() const;
00087     int getY() const;
00093     std::string getName() const;
00094
00102     friend std::ostream& operator<<(std::ostream& os, const Animal& a);
00110     friend std::istream& operator>>(std::istream& is, Animal& a);
00111 };
00112
00113 #endif // ANIMAL_H
```

## 5.4 src/animal/exception/AnimalException.cpp Failo Nuoroda

```
#include "AnimalException.h"
#include <string>
#include <exception>
```

Įtraukimo priklausomybių diagrama AnimalException.cpp:



### 5.4.1 Smulkus aprašymas

#### Autorius

Rokas Braidokas ( [rokasbraidokas@gmail.com](mailto:rokasbraidokas@gmail.com) )

#### Versija

0.1

#### Data

2025-05-09

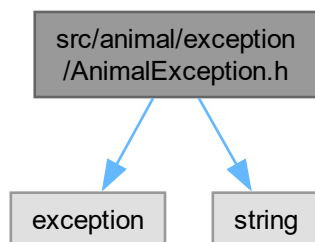
#### Copyright

Copyright (c) 2025

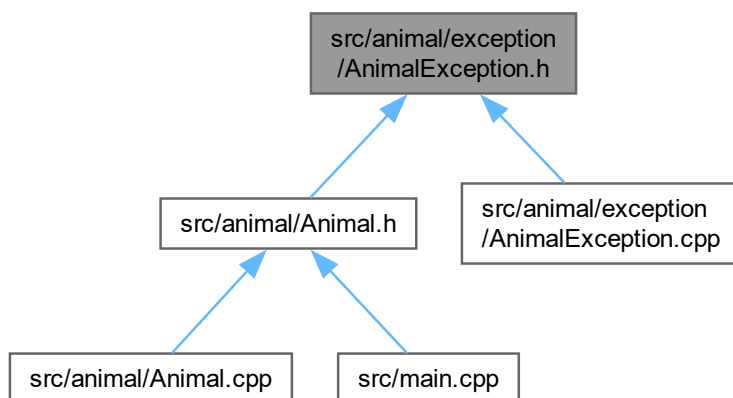
## 5.5 src/animal/exception/AnimalException.h Failo Nuoroda

```
#include <exception>
#include <string>
```

Įtraukimo priklausomybių diagrama AnimalException.h:



Šis grafas rodo, kuris failas tiesiogiai ar netiesiogiai įtraukia šį failą:



## Klasės

- class [AnimalException](#)

### 5.5.1 Smulkus aprašymas

#### Autorius

Rokas Braidokas ( [rokasbraidokas@gmail.com](mailto:rokasbraidokas@gmail.com) )

#### Versija

0.1

## Data

2025-05-09

## Copyright

Copyright (c) 2025

## 5.6 AnimalException.h

[Eiti į šio failo dokumentaciją.](#)

```

00001
00011 #ifndef ANIMALEXCEPTION_H
00012 #define ANIMALEXCEPTION_H
00013
00014 #include <exception>
00015 #include <string>
00016
00017 class AnimalException : public std::exception {
00018 private:
00019     std::string message;
00020 public:
00026     explicit AnimalException(const std::string& msg);
00032     const char* what() const noexcept override;
00033 };
00034
00035 #endif

```

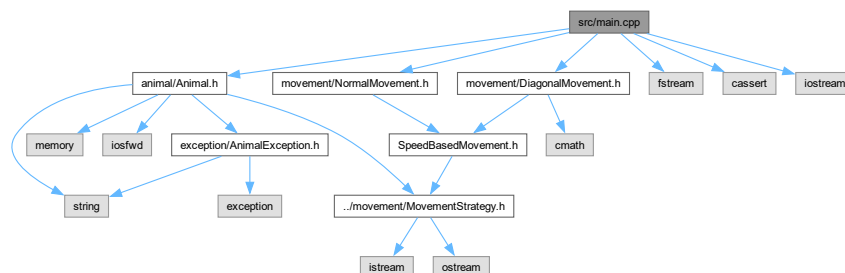
## 5.7 src/main.cpp Failo Nuoroda

```

#include "animal/Animal.h"
#include "movement/NormalMovement.h"
#include "movement/DiagonalMovement.h"
#include <fstream>
#include <cassert>
#include <iostream>

```

Įtraukimo priklausomybių diagrama main.cpp:



## Funkcijos

- int [main](#) ()

### 5.7.1 Smulkus aprašymas

#### Autorius

Rokas Braidokas ( [rokasbraidokas@gmail.com](mailto:rokasbraidokas@gmail.com) )

#### Versija

0.1

#### Data

2025-05-06

#### Copyright

Copyright (c) 2025

### 5.7.2 Funkcijos Dokumentacija

#### 5.7.2.1 main()

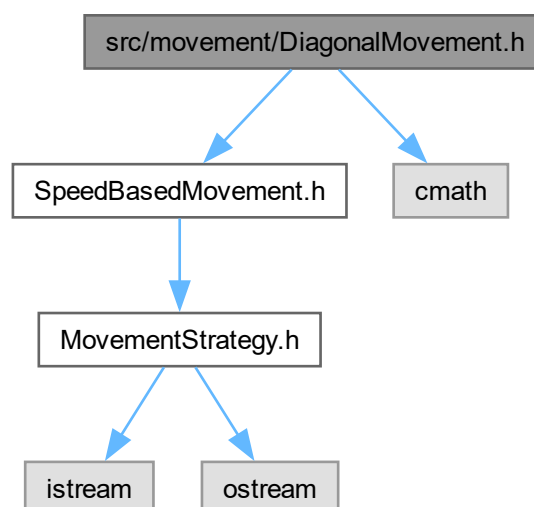
```
int main ()
```

## 5.8 src/movement/DiagonalMovement.h Failo Nuoroda

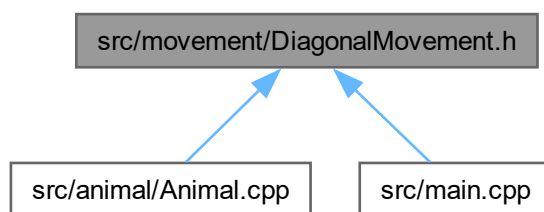
```
#include "SpeedBasedMovement.h"
```

```
#include <cmath>
```

Įtraukimo priklausomybių diagrama DiagonalMovement.h:



Šis grafas rodo, kuris failas tiesiogiai ar netiesiogiai įtraukia šį failą:



### Klasės

- class [DiagonalMovement](#)  
*Allows diagonal moves at constant cost.*

## 5.8.1 Smulkus aprašymas

### Autorius

Rokas Braidokas ( [rokasbraidokas@gmail.com](mailto:rokasbraidokas@gmail.com))

### Versija

0.1

### Data

2025-05-06

### Copyright

Copyright (c) 2025

## 5.9 DiagonalMovement.h

Eiti į šio failo dokumentaciją.

```
00001
00011 #ifndef DIAGONALMOVEMENT_H
00012 #define DIAGONALMOVEMENT_H
00013
00014 #include "SpeedBasedMovement.h"
00015 #include <cmath>
00016
00021 class DiagonalMovement : public SpeedBasedMovement {
00022 public:
00028     explicit DiagonalMovement(double mult = 1.0)
00029         : SpeedBasedMovement(mult)
```

```

00030     {}
00039     void move(int& x, int& y, int dx, int dy) override {
00040         double dist = std::hypot(dx, dy);
00041         if (dist > 0) {
00042             x += static_cast<int>((dx / dist) * speedMultiplier_);
00043             y += static_cast<int>((dy / dist) * speedMultiplier_);
00044         }
00045     }
00046
00052     char typeTag() const override { return 'D'; }
00053 };
00054
00055 #endif // DIAGONALMOVEMENT_H

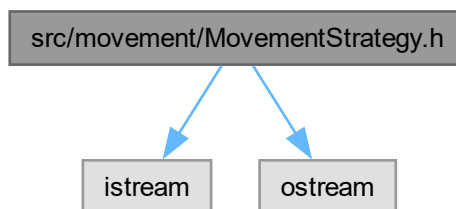
```

## 5.10 src/movement/MovementStrategy.h Failo Nuoroda

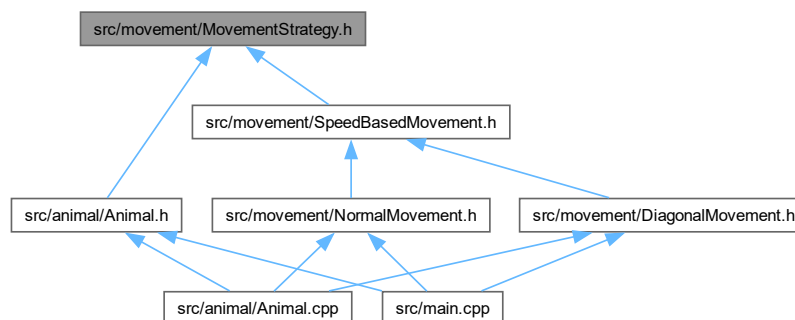
```
#include <istream>
```

```
#include <ostream>
```

Įtraukimo priklausomybių diagrama MovementStrategy.h:



Šis grafas rodo, kuris failas tiesiogiai ar netiesiogiai įtraukia šį failą:



### Klasės

- class [MovementStrategy](#)

*Abstract interface for animal movement behavior.*



### 5.10.1 Smulkus aprašymas

#### Autorius

Rokas Braidokas ( [rokasbraidokas@gmail.com](mailto:rokasbraidokas@gmail.com))

#### Versija

0.1

#### Data

2025-05-06

#### Copyright

Copyright (c) 2025

## 5.11 MovementStrategy.h

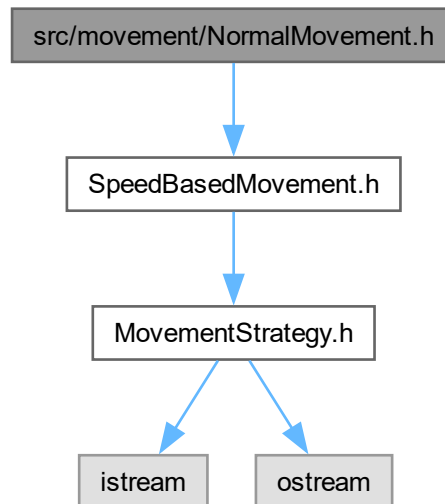
[Eiti į šio failo dokumentaciją.](#)

```
00001
00011 #ifndef MOVEMENTSTRATEGY_H
00012 #define MOVEMENTSTRATEGY_H
00013
00014 #include <istream>
00015 #include <ostream>
00016
00018 class MovementStrategy {
00019 public:
00024     virtual ~MovementStrategy() = default;
00025
00034     virtual void move(int& x, int& y, int dx, int dy) = 0;
00035
00041     virtual void serialize(std::ostream& os) const = 0;
00042
00048     virtual void deserialize(std::istream& is) = 0;
00049
00055     virtual char typeTag() const = 0;
00056 };
00057
00058 #endif // MOVEMENTSTRATEGY_H
```

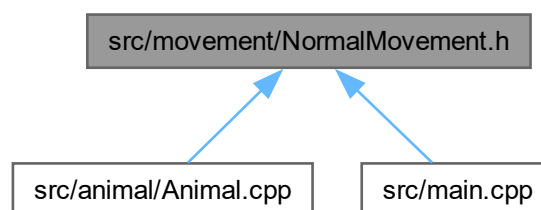
## 5.12 src/movement/NormalMovement.h Failo Nuoroda

```
#include "SpeedBasedMovement.h"
```

Įtraukimo priklausomybių diagrama NormalMovement.h:



Šis grafas rodo, kuris failas tiesiogiai ar netiesiogiai įtraukia šį failą:



### Klasės

- class [NormalMovement](#)  
*Simple axis-aligned movement.*

### 5.12.1 Smulkus aprašymas

#### Autorius

Rokas Braidokas ( [rokasbraidokas@gmail.com](mailto:rokasbraidokas@gmail.com))

#### Versija

0.1

#### Data

2025-05-09

#### Copyright

Copyright (c) 2025

## 5.13 NormalMovement.h

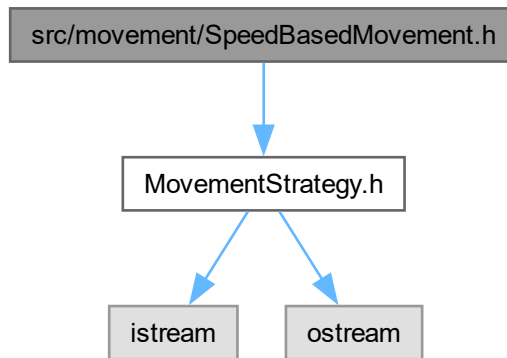
[Eiti į šio failo dokumentaciją.](#)

```
00001
00011 #ifndef NORMALMOVEMENT_H
00012 #define NORMALMOVEMENT_H
00013
00014 #include "SpeedBasedMovement.h"
00015
00020 class NormalMovement : public SpeedBasedMovement {
00021 public:
00027     explicit NormalMovement(double mult = 1.0)
00028         : SpeedBasedMovement(mult)
00029     {}
00038     void move(int& x, int& y, int dx, int dy) override {
00039         x += static_cast<int>(dx * speedMultiplier_);
00040         y += static_cast<int>(dy * speedMultiplier_);
00041     }
00042
00048     char typeTag() const override { return 'N'; }
00049 };
00050
00051 #endif // NORMALMOVEMENT_H
```

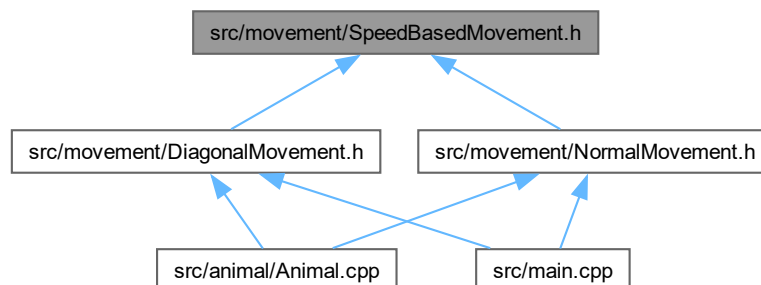
## 5.14 src/movement/SpeedBasedMovement.h Failo Nuoroda

```
#include "MovementStrategy.h"
```

Įtraukimo priklausomybių diagrama SpeedBasedMovement.h:



Šis grafas rodo, kuris failas tiesiogiai ar netiesiogiai įtraukia šį failą:



### Klasės

- class [SpeedBasedMovement](#)  
*Base class for strategies that share a speedMultiplier.*

### 5.14.1 Smulkus aprašymas

#### Autorius

Rokas Braidokas ( [rokasbraidokas@gmail.com](mailto:rokasbraidokas@gmail.com) )

## Versija

0.1

## Data

2025-05-06

## Copyright

Copyright (c) 2025

## 5.15 SpeedBasedMovement.h

[Eiti į šio failo dokumentaciją.](#)

```
00001
00011 #ifndef SPEEDBASEDMOVEMENT_H
00012 #define SPEEDBASEDMOVEMENT_H
00013
00014 #include "MovementStrategy.h"
00015
00020 class SpeedBasedMovement : public MovementStrategy {
00021 protected:
00022     double speedMultiplier_;
00023
00024 public:
00030     explicit SpeedBasedMovement(double mult = 1.0)
00031         : speedMultiplier_(mult)
00032     {}
00033
00039     void serialize(std::ostream& os) const override {
00040         os.write(reinterpret_cast<const char*>(&speedMultiplier_), sizeof(speedMultiplier_));
00041     }
00042
00048     void deserialize(std::istream& is) override {
00049         is.read(reinterpret_cast<char*>(&speedMultiplier_), sizeof(speedMultiplier_));
00050     }
00051
00057     double getSpeedMultiplier() const { return speedMultiplier_; }
00058
00064     void setSpeedMultiplier(double m) { speedMultiplier_ = m; }
00065 };
00066
00067 #endif // SPEEDBASEDMOVEMENT_H
```



# Rodyklë

- ~MovementStrategy
  - MovementStrategy, [16](#)
- Animal, [7](#)
  - Animal, [8](#)
  - getID, [8](#)
  - getName, [8](#)
  - getX, [9](#)
  - getY, [9](#)
  - move, [9](#)
  - operator<<, [10](#)
  - operator>>, [10](#)
  - operator=, [9](#)
  - setMovementStrategy, [10](#)
- Animal.cpp
  - operator<<, [26](#)
  - operator>>, [26](#)
- AnimalException, [11](#)
  - AnimalException, [12](#)
  - what, [12](#)
- deserialize
  - MovementStrategy, [16](#)
  - SpeedBasedMovement, [22](#)
- DiagonalMovement, [13](#)
  - DiagonalMovement, [14](#)
  - move, [14](#)
  - typeTag, [15](#)
- getID
  - Animal, [8](#)
- getName
  - Animal, [8](#)
- getSpeedMultiplier
  - SpeedBasedMovement, [22](#)
- getX
  - Animal, [9](#)
- getY
  - Animal, [9](#)
- main
  - main.cpp, [32](#)
- main.cpp
  - main, [32](#)
- move
  - Animal, [9](#)
  - DiagonalMovement, [14](#)
  - MovementStrategy, [16](#)
  - NormalMovement, [19](#)
- MovementStrategy, [15](#)
- ~MovementStrategy, [16](#)
  - deserialize, [16](#)
  - move, [16](#)
  - serialize, [17](#)
  - typeTag, [17](#)
- NormalMovement, [18](#)
  - move, [19](#)
  - NormalMovement, [19](#)
  - typeTag, [20](#)
- operator<<
  - Animal, [10](#)
  - Animal.cpp, [26](#)
- operator>>
  - Animal, [10](#)
  - Animal.cpp, [26](#)
- operator=
  - Animal, [9](#)
- serialize
  - MovementStrategy, [17](#)
  - SpeedBasedMovement, [22](#)
- setMovementStrategy
  - Animal, [10](#)
- setSpeedMultiplier
  - SpeedBasedMovement, [22](#)
- SpeedBasedMovement, [20](#)
  - deserialize, [22](#)
  - getSpeedMultiplier, [22](#)
  - serialize, [22](#)
  - setSpeedMultiplier, [22](#)
  - SpeedBasedMovement, [21](#)
  - speedMultiplier\_, [23](#)
- speedMultiplier\_
  - SpeedBasedMovement, [23](#)
- src/animal/Animal.cpp, [25](#)
- src/animal/Animal.h, [27](#), [28](#)
- src/animal/exception/AnimalException.cpp, [28](#)
- src/animal/exception/AnimalException.h, [29](#), [31](#)
- src/main.cpp, [31](#)
- src/movement/DiagonalMovement.h, [32](#), [33](#)
- src/movement/MovementStrategy.h, [34](#), [35](#)
- src/movement/NormalMovement.h, [36](#), [37](#)
- src/movement/SpeedBasedMovement.h, [38](#), [39](#)
- typeTag
  - DiagonalMovement, [15](#)
  - MovementStrategy, [17](#)
  - NormalMovement, [20](#)

what

AnimalException, [12](#)