Cat Mouse

High Distinction Project for HIT2302 Object Oriented Programming

Generated by Doxygen 1.8.5

Sat Nov 9 2013 22:18:05

Contents

1 C++ Coupled Implementation

Version

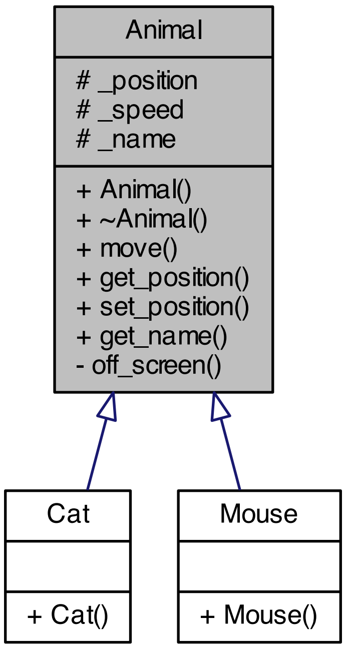
3

2 Class Documentation

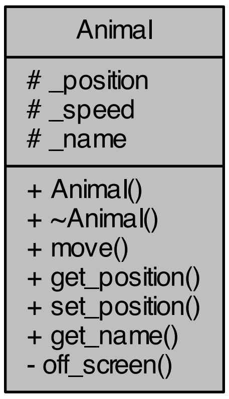
2.1 Animal Class Reference

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

Inheritance diagram for Animal:



Collaboration diagram for Animal:



Public Member Functions

- **Animal** ()
- virtual ∼**Animal** ()=0
- void **move** (**dirs** dir)
- point2d ∗const **get_position** ()
- void **set_position** (point2d ∗newPos)
- string const **get_name** ()

Protected Attributes

- point2d ∗ **_position**
- const int **_speed** = 3
- string **_name**

Private Member Functions

- void **off_screen** ()

2.1.1 Detailed Description

Defines an abstract, base class for a playable ‘thing’ on the screen which can move around etc.

Defines class for the general ‘game’ of the cat and mice.

Author

Alex Cummaudo

Date

16 Oct 2013

Author

Alex Cummaudo

Date

2 Nov 2013

2.1.2 Constructor & Destructor Documentation

2.1.2.1 Animal::Animal ( )

Default constructor for initialising _position and _speed for all new Animals.

2.1.2.2 Animal::∼Animal ( ) `[pure virtual]`

Destructor reliquishes resources created in this class.

2.1.3 Member Function Documentation

2.1.3.1 void Animal::move (dirs *dir*)

Move implementation for a **Animal** (p. 2) to move an animal in a direction at its speed.

Parameters

dir	Direction the animal is told to move in (alters x and y axis position of poisition accordingly)

2.1.3.2 point2d∗ const Animal::get_position ( )

Read property to get the position of the **Animal** (p. 2)

Returns

Position of the animal

2.1.3.3 void Animal::set_position (point2d ∗ *newPos*)

Write property to set the position of the **Animal** (p. 2)

Parameters

newPos	New position to set the animal at

2.1.3.4 string const Animal::get_name ( )

Readonly property to get the name of the animal

Returns

Name of the animal

2.1.3.5 void Animal::off_screen ( ) `[private]`

Off screen check that prevents any **Animal** (p. 2) from going outside the borders of the screen.

2.1.4 Member Data Documentation

2.1.4.1 point2d∗ Animal::_position [protected]

Centrepoint position of the animal.

2.1.4.2 const int Animal::_speed = 3 [protected]

Speed at which animals move at, set to a value of 3.

2.1.4.3 string Animal::_name [protected]

Name of animals, overriden by children (i.e. ‘Cat’ or ‘Mouse’)

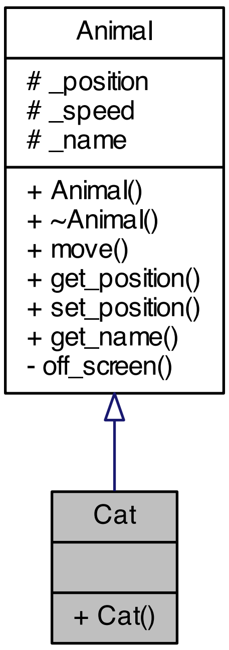
The documentation for this class was generated from the following files:

- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**Animal.-h**
- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**Animal.-cpp**

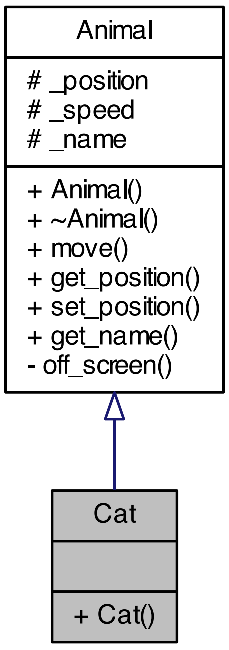
2.2 Cat Class Reference

```
#include <CatMouse.hpp>
```

Inheritance diagram for Cat:



Collaboration diagram for Cat:



Public Member Functions

- **Cat** ()

Additional Inherited Members

2.2.1 Detailed Description

Defines an class for a playable chaser (i.e. the chasing cat)

Author

Alex Cummaudo

Date

18 Oct 2013

2.2.2 Constructor & Destructor Documentation

2.2.2.1 Cat::Cat ( )

The default constructor for the cat constructs parent and sets position on lefthand-side of screen.

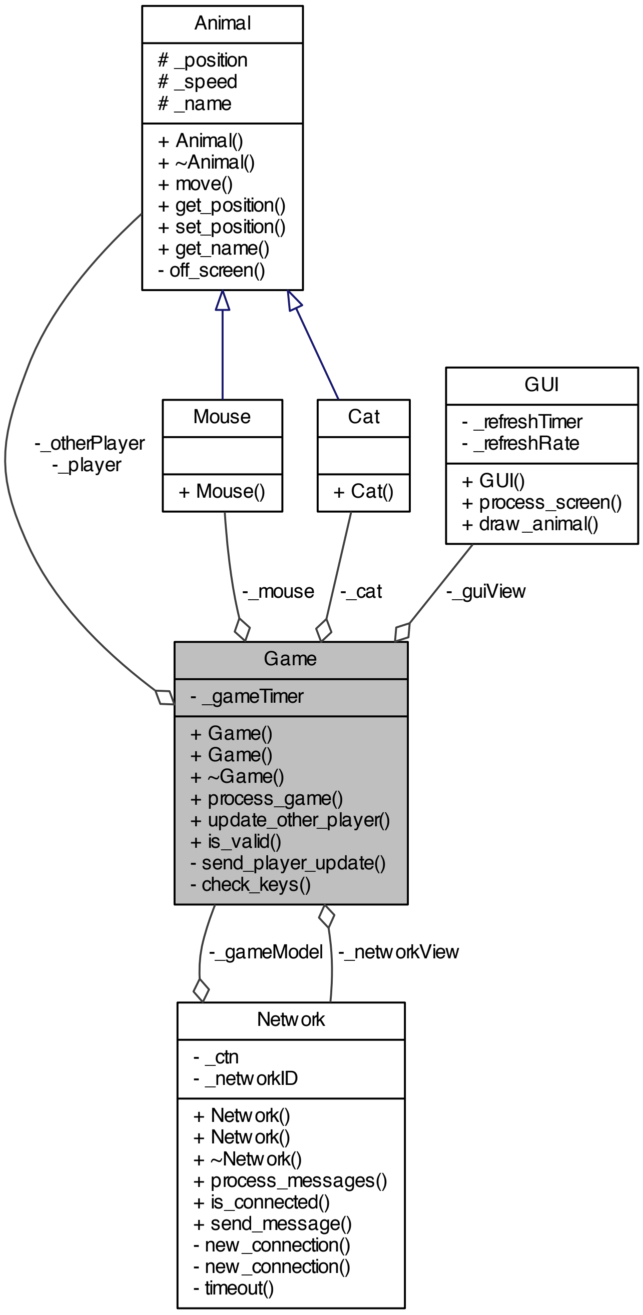
The documentation for this class was generated from the following file:

- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**Cat-Mouse.hpp**

2.3 Game Class Reference

```
#include <Game.h>
```

Collaboration diagram for Game:



Public Member Functions

- **Game** ()
- **Game** (string ipAddr)

- ∼**Game** ()
- void **process_game** ()
- void **update_other_player** (float x, float y)
- bool **is_valid** ()

Private Member Functions

- void **send_player_update** ()
- void **check_keys** ()

Private Attributes

- **Animal** ∗ **_player**
- **Animal** ∗ **_otherPlayer**
- **Cat** ∗ **_cat**
- **Mouse** ∗ **_mouse**
- timer **_gameTimer**
- **Network** ∗ **_networkView**
- **GUI** ∗ **_guiView**

2.3.1 Detailed Description

Defines class for the general 'game' of the cat and mice.

Author

Alex Cummaudo

Date

2 Nov 2013

2.3.2 Constructor & Destructor Documentation

2.3.2.1 Game::Game ( )

On construction of a game, this overloaded version of the constructor will intitalise the network as a host, and make the player a cat.

2.3.2.2 Game::Game (string *ipAddr*)

On construction of a game, this overloaded version of the constructor will intitalise the network as a client, and make the player a mouse.

Parameters

ipAddr	Host to connect to

2.3.2.3 Game::∼Game ( )

Destructor reliquishes resources created in this class.

2.3.3 Member Function Documentation

2.3.3.1 void Game::process_game ( )

Called to process the game.

2.3.3.2 void Game::update_other_player (float *x,* float *y*)

Updates the other player's position.

Note

This is called from CMNetwork

2.3.3.3 bool Game::is_valid ( )

The is valid ensures that the game is still valid, only where it has a network connection (essentially, returns the state of the connection as true/false)

Returns

The validity of the game (whether or not is connected)

2.3.3.4 void Game::send_player_update ( ) `[private]`

Sends a message over the network to update the current player's position.

2.3.3.5 void Game::check_keys ( ) `[private]`

Method called to update the players position and send it to the remote model.

2.3.4 Member Data Documentation

2.3.4.1 Animal∗ Game::_player `[private]`

Player of the game (person controlling the game)

2.3.4.2 Animal∗ Game::_otherPlayer `[private]`

Other player in the game (other person controlling enemy)

2.3.4.3 Cat∗ Game::_cat `[private]`

Cat (p. 5) (chaser) of the game.

2.3.4.4 Mouse∗ Game::_mouse `[private]`

Mouse (p. 12) (chasee) of the game.

2.3.4.5 timer Game::_gameTimer `[private]`

Ingame-timer to keep track of how long game has gone for.

2.3.4.6 Network∗ Game::_networkView `[private]`

Couple the network view to the game model.

2.3.4.7 GUI∗ Game::_guiView `[private]`

Couple the **GUI** (p. 10) view to the game model only for HUD.

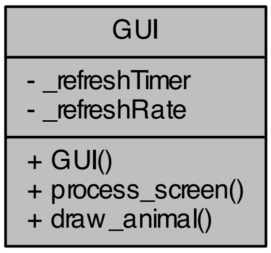
The documentation for this class was generated from the following files:

- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**Game.-h**
- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**Game.-cpp**

2.4 GUI Class Reference

`#include <GUI.h>`

Collaboration diagram for GUI:



Public Member Functions

- **GUI** (float refRate)
- void **process_screen** ()
- void **draw_animal** (**Animal** ∗animal)

Private Attributes

- timer **_refreshTimer**
- float **_refreshRate**

2.4.1 Detailed Description

Provides **GUI** (p. 10) View for the game to display the game on in a graphics window.

Author

Alex Cummaudo

Date

2 Nov 2013

2.4.2 Constructor & Destructor Documentation

2.4.2.1 GUI::GUI (float *refRate*)

Constructor for **GUI** (p. 10) view creates a graphics window for SwinGame.

2.4.3 Member Function Documentation

2.4.3.1 void GUI::process_screen ( )

Clears and refreshes the screen by the time given in reset timer.

2.4.3.2 void GUI::draw_animal (Animal * *animal*)

Draws the given animal to the screen.

Parameters

animal	**Animal** (p. 2) to draw to the screen

2.4.4 Member Data Documentation

2.4.4.1 timer GUI::_refreshTimer `[private]`

Timer used to refresh the screen at the by clearing the screen and resetting at refreshRate given

2.4.4.2 float GUI::_refreshRate `[private]`

Seconds to refresh the screen at.

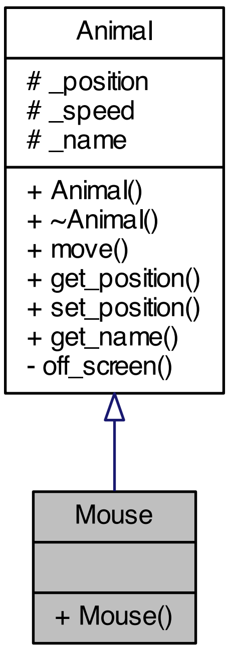
The documentation for this class was generated from the following files:

- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**G-UI.h**
- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**G-UI.cpp**

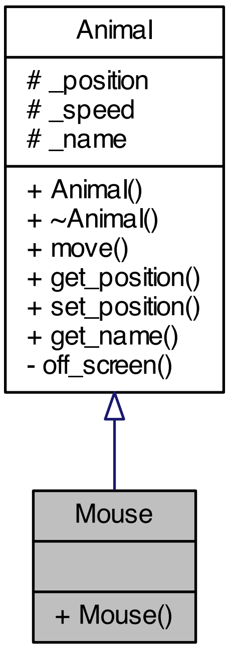
2.5 Mouse Class Reference

`#include <CatMouse.hpp>`

Inheritance diagram for Mouse:



Collaboration diagram for Mouse:



Public Member Functions

- **Mouse** ()

Additional Inherited Members

2.5.1 Detailed Description

Defines an class for a playable chasee (i.e. the hunted mouse)

Author

Alex Cummaudo

Date

18 Oct 2013

2.5.2 Constructor & Destructor Documentation

2.5.2.1 Mouse::Mouse ( )

The default constructor for the mouse constructs parent and sets position on righthand-side of screen.

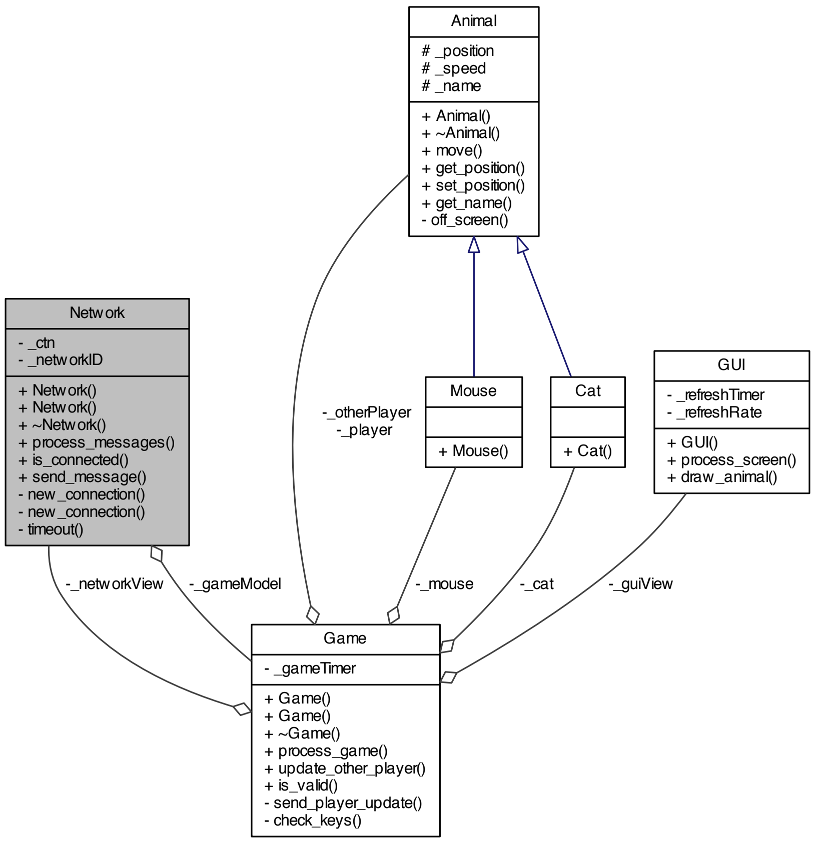
The documentation for this class was generated from the following file:

- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**Cat-Mouse.hpp**

2.6 Network Class Reference

`#include <Network.h>`

Collaboration diagram for Network:



Public Member Functions

- **Network** (**Game** ∗gameModel)
- **Network** (**Game** ∗gameModel, string ipAddr)
- ∼**Network** ()
- void **process_messages** ()
- bool **is_connected** ()
- void **send_message** (string msg)

Private Member Functions

- connection **new_connection** ()
- connection **new_connection** (string ipAddr)
- void **timeout** (string msgPrompt, string msgSucc, string msgFail, int timeoutSecs, function< void(void)> countdownBody, function< bool(void)> breakCondition)

Private Attributes

- connection **_ctn**

- string **_networkID**
- **Game** ∗ **_gameModel**

2.6.1 Detailed Description

Packages up data recieved from the controller and passes it to a given network.

Author

Alex Cummaudo

Date

2 Nov 2013

2.6.2 Constructor & Destructor Documentation

2.6.2.1 Network::Network (Game ∗ *gameModel*)

Constructor initiates a connection as a host.

Parameters

gameModel	The game model to link to and update

2.6.2.2 Network::Network (Game ∗ *gameModel,* string *ipAddr*)

Constructor initiates a connection to a given ip address (as a client)

Parameters

gameModel	The game model to link to and update
ipAddr	The IP Address of the host this client will connect to

2.6.2.3 Network::∼Network ( )

Destructor closes all connections and announces Goodbye message.

2.6.3 Member Function Documentation

2.6.3.1 void Network::process_messages ( )

Process messages that are being recieved (i.e. incoming network string to an outgoing event)

Note

Any incoming messages that suggests changes to the model will do so directly here.

Note

Messages recieved in the format: key:value,key:value| etc. Hence we want to parse the msg back into its event kind

2.6.3.2 bool Network::is_connected ( )

Delcare is connected property

Returns

Boolean whether or not the network is connected

2.6.3.3 void Network::send_message (string *msg*)

Sends a string over the network by packaging it and sending it over the network as a string (i.e. an outgoing event to a network string)

Parameters

msg	String to send over the network

2.6.3.4 connection Network::new_connection ( ) [private]

Initiates the connection as a host, returning true or false on a success or error.

Returns

A new host connection to work with

Force break condition to be true

2.6.3.5 connection Network::new_connection (string *ip_addr*) [private]

Initiates the connection as a client, returning true or false on a success or error.

Parameters

ip_addr	IP Address that this client should connect to

Returns

A new client connection to work with

2.6.3.6 void Network::timeout (string *msgPrompt,* string *msgSucc,* string *msgFail,* int *timeoutSecs,* function< void(void)> *countdownBody,* function< bool(void)> *breakCondition*) [private]

The timeout connection; runs the passed function success on a success, and error function on error; allow passing of two functors so that lambda expressions can be passed in as parameters to the timeout.

Parameters

msgPrompt	Prompt message announced when timeout begins (i.e., why we're having a timeout).
msgSucc	Message announced when timeout did not run out and the break condition was met
msgFail	Message announced when timeout did ran out of timeoutSecs and the break condition was never met
timeoutSecs	How long to run timeout for
countdownBody	Function to run every second on timeout
breakCondition	Function that returns a bool to check whether or not the timeout should break

2.6.4 Member Data Documentation

2.6.4.1 connection Network::_ctn [private]

Network (p. 14) connection controller between client and host.

2.6.4.2 string Network::_networkID [private]

Defines a unique address of this machine.

2.6.4.3 Game* Network::_gameModel [private]

Defines the coupling between game and network.

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

- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**Network.-h**

- /Users/Alex/Dropbox/Swinburne/HIT2302 - OOP/Projects/Cat and Mouse/#3_CatMouse_C++_Coupled/src/**Network.-cpp**