

# NetHunt - Developer Guide

NetHunt is an extension of the game FlatHunt, which is well-known among the IT students of the ETH Zurich. In the original game FlatHunt a mode “versus” already exists, but you have to play it on one computer and the one, who plays the hunters always has to close his eyes when it’s the Estate Agent’s turn. To improve this mode we decided to write an extension “NetHunt”, with which it should be possible to play the mode versus over a network.

To realize this project, we had to write a little application for the server and add some classes to FlatHunt.

In the following few pages the new classes and its most important features are explained:

## Nethunt Server:

The rootclass of the new application nethunt\_server is called **START\_SERVER**. In its feature make the code *create\_server.make\_server(port\_number)* is executed, which creates a new server with the port number *port\_number*. Server is an instance of the class *OUR\_SERVER*.

The **class OUR\_SERVER**, which opens a server and handles the communication with the clients, inherits from classes *NETWORK\_SERVER* and *STORABLE*. It has several features:

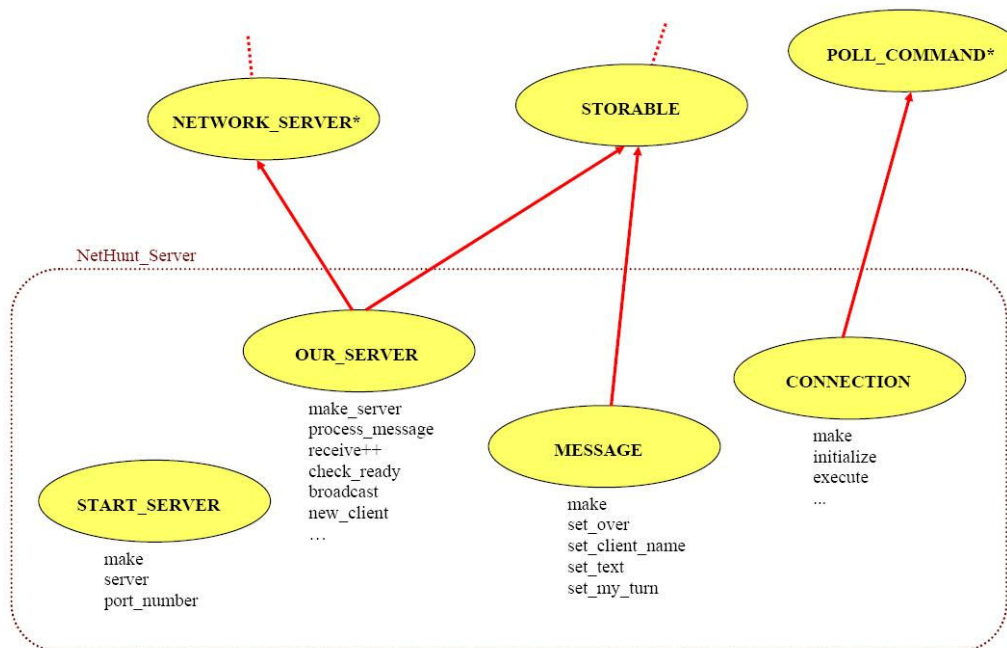
- *make\_server (port: INTEGER)* is its initialization feature with the argument *port* of type *INTEGER*.
- *new\_client* creates a new connection and adds it to *connections*, a linked list (*connections: LINKED\_LIST [CONNECTION]*), in which the connections are stored.
- *process\_message* handles the received message of the client “Estate Agent” or “Flat Hunter”
- *check\_ready* checks if the Estate Agent and the Hunter are connected with the server and then calls feature broadcast if so.
- *broadcast* sends the current message to the client
- *message\_in*, *message\_out*, *received* are all of the type *MESSAGE*.

**Class *CONNECTION*** is to handle the connection with the client. Its important features are the attributes *is\_waiting*: *BOOLEAN* and *client\_name*: *STRING*.

**Class *MESSAGE*** inherits from *STORABLE* and consists of the attributes

- *over*, a *BOOLEAN*,
- *my\_turn*, also a *BOOLEAN* which denotes if it is the turn of the current player,
- *client\_name*, a *STRING* for the name of the client,
- *text*, a *STRING* for some text

and the features *set\_over*, *set\_my\_turn*, *set\_client\_name*, *set\_text* to set the values of the analogical attributes.



## NetHunt (FlatHunt)

For the mode versus we didn't have to change many classes. We only had to change the class *MAIN\_CONTROLLER* in which the whole activity is controlled and to add the classes *MULTIPLAYER\_DIALOG*, *CONNECTION*, *MESSAGE*, *MESSAGE\_PLACE* and *OUR\_CLIENT*.

### MAIN\_CONTROLLER:

In this existing class of FlatHunt we added the feature *open\_multiplayer\_dialog*, which opens a window to choose the multiplayer options.

In the new feature *establish\_connection* the clients "Flat Hunter" and "Estate Agent" are created. We didn't only have to add new features, but also to change already existing ones. The most important one is *process\_click*, because it is where all the turns are controlled. In this feature, if the mode "versus" is selected, we had to control that only one player can make a turn and then the location is sent to the server, which it sends it to the other client, which can make the next turn. This sending of the location doesn't function in our project, but we don't know why not.

### MULTIPLAYER\_DIALOG:

The class *MULTIPLAYER\_DIALOG* creates a window, in which you can choose if you want to be the "Estate Agent" or the "Flat Hunter" and in which you have to enter the IP of the server and the correct port.

### CONNECTION:

In this class the connection is handled. It has the same features as the class *CONNECTION* in the "NetHunt Server".

### MESSAGE:

The class *MESSAGE* has the same implementation like the appropriate one in the "NetHunt Server".

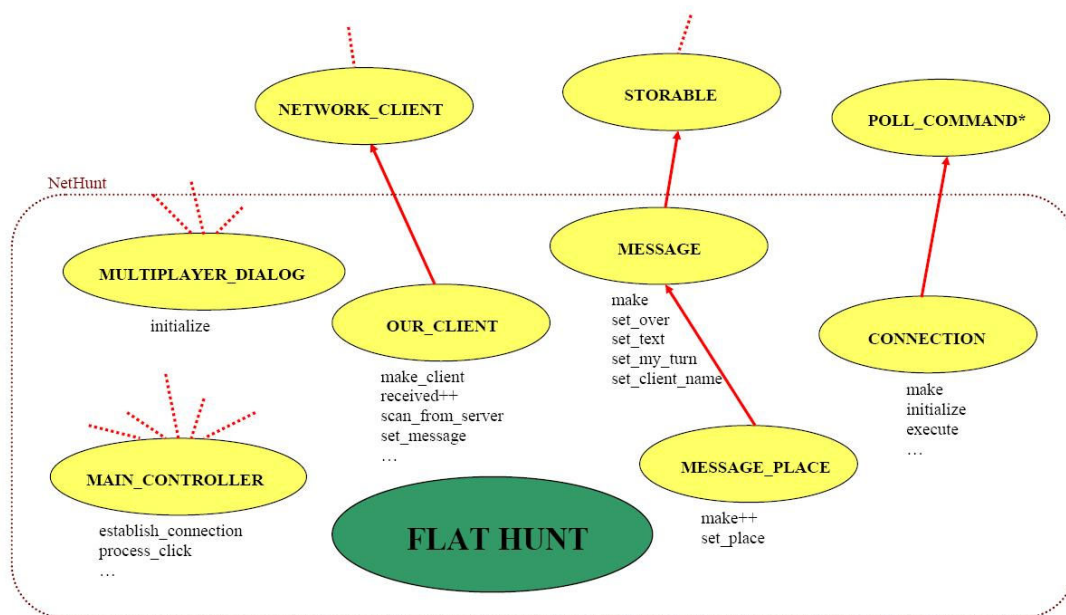
### MESSAGE\_PLACE:

The class *MESSAGE\_PLACE* inherits from the class *MESSAGE*. It has one more attribute: *place*, which is of the type *PLACE* and the feature *set\_place* (*value: PLACE*), which assigns *value* to *place*.

**OUR\_CLIENT:**

The class *OUR\_CLIENT* inherits from the class *NETWORK\_CLIENT*. Its features are:

- *make\_client* (*host*: *STRING*; *port*: *INTEGER*), its initialization feature; *host* is the IP of the server we want to connect with and *port* is the port
- *message\_in*, *message\_out*, *received*, all of the type *MESSAGE*
- *set\_message*, with which we can assign a value to *message\_out*
- *processing*, which is composed of a loop in which the feature *scan\_from\_server* is called
- *scan\_from\_server*, which waits till the server sends something and then receives the message and stores it in *message\_in*.



We regret that the program doesn't work like it should, but we couldn't solve the problem with sending the place. Nevertheless we are thankful for what we learned with this project: to improve our programming, to learn to understand and then extend an already existing program and to manage the work together in our group.