

Blocks World For Teams using GOAL

Maaïke Harbers, Wouter Pasman, Birna van Riemsdijk
November 2013

Introduction

This document describes how to install Blocks World For Teams (BW4T) for use with GOAL. BW4T is a *client-server* system. The *server* is responsible for the administration, simulation and visualization of the virtual world: it keeps track of robots, rooms, blocks, connected GOAL agents, etc. The server uses Repast, software to simulate virtual environments, to do part of this administration. The *client* is GOAL, which runs a multi-agent system (MAS) and connects to the server. The agents in the GOAL client get percepts from the server, and send actions to the server. Client and server can run on a different computer. This document describes how to install the BW4T server, configure it, place the BW4T client in GOAL, configure the MAS file, and run the system.

We use the following names to refer to directories of BW4T:

- <GOAL> refers to the directory where you installed GOAL.
- <SERVER> refers to <GOAL>/environments/BW4T2/BW4TServer/.

System requirements

To use BW4T you need Java JDK 6 or higher. The BW4T2 environment has been tested on Windows 7 and OSX.

Installing the server

1. Download and install Repast2.0 BETA from our mirror ii.tudelft.nl/repast.
2. Install Repast in the default installation directory which is:
on Windows **C:\RepastSymphony-2.0-beta**, on OSX **/Applications/Repast-Symphony-2.0.0-beta** and on Linux **~/repast-2.0.0-beta** (~ refers to your home directory). If you choose to install it in another directory, see Advanced run settings (Server) below (or make a link from the default install location).

Custom server installation

If you use a non-default install directory for Repast, you need to fix the server startup script as follows.

1. Go to the <SERVER> directory and open the **BW4TServer.bat** (windows), **BW4TServer.sh.command** (OSX) or **BW4TServer.sh** (Linux) file in a text editor.
2. Change the Repast variable to the directory that you installed Repast in.
3. Save your changes

Running the Server

Run the server before running the client, as otherwise the client cannot connect to the server. Start the server by going to the <SERVER> directory and running the **BW4TServer.bat** file (Windows), **BW4TServer.sh.command** (OSX) or **BW4TServer.sh** (Linux). This should open the Server window (Figure 1). Note, this takes some time.

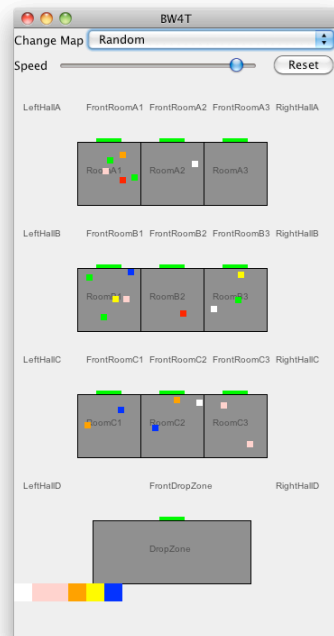


Illustration 1. Server window showing the actual state overview window of the Environment.

Advanced run settings (Server)

The default settings for the server can be changed in its startup script: **BW4TServer.bat** (Windows), **BW4TServer.sh.command** (OSX) or **BW4TServer.sh** (Linux). To do so, go to the <SERVER> directory and open the file with the startup script in a text editor. The following settings can be changed:

1. Serverip: the ip address that the server listens on (default: localhost).
2. Serverport: the port that the server listens on (default: 8000).
3. Map: the default map that is loaded initially (default: maps/Random) (see section “Loading and creating new maps” below).

If you change the serverip and/or serverport, change the client settings correspondingly (see below). You need to close and restart the server to use these new values.

Running the Client

Before running the client, make sure that the server is running (see above).

1. Start GOAL.
2. Open the bw4t2.mas2g located in <GOAL>/GOALagents/BW4T2
3. Run the bw4t2.mas2g file in GOAL to create the agents in correspondence with the specification in the mas2g file.
4. Start (un-pause) the agents by again pressing the run button in GOAL.

Note that the server shows only connected entities, so the bots only appear after the GOAL MAS has been started (See Illustration 2).

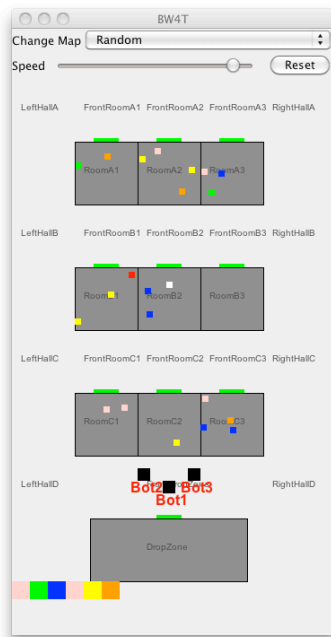


Illustration 2: Bots are shown only after the entities have been connected to GOAL agents.

Advanced run settings (Client)

The default settings for the client can be changed in the `bw4t2.mas2g` file (GOAL). The following settings can be changed.

- **Serverip**: the ip address that the server listens on (default: localhost).
- **Serverport**: the port that the server listens on (default:8000).
- **Agentcount** : the amount of agents (also specified in the launchpolicy section, see below), that the client should load. If the agentcount is higher than the amount of entities in the map then they won't be loaded. (default: 0).
- **Humancount**: the amount of human players that should be loaded. If the humancount is higher than the amount of entities in the map then they won't be loaded. (default: 0).
- **Clientip** : the ip address that the client listens on (default: localhost).
- **Clientport** : the port that the client listens on (default: 2000).
- **Launchgui**: whether to launch a separate GUI for each bot (controlled by an agent or human) can be set to true or false. This GUI shows the environment from the perspective of the bot. (default:false).
- **Map**. The map name to be loaded. If you specify a map, the server will reset to load the new map, which disconnects all entities.

The number of agents is specified at two places in the `mas2g` file. First, the `agentcount` and `humancount` specify the number of entities of the corresponding type that should be created in the environment. Second, the `launchpolicy` specifies which and how many agents should be connected


to these entities. Make sure that the agentcount and humancount in the initialization parameters are in line with the launch policy section in the mas2g file. Furthermore, the number of agents should not exceed the maximum number of agents defined on the map (see section “Loading and creating new maps” below).

If you use BW4T from a batch runner, you may want to reset the server after each run. This is done by specifying a map in the mas file init parameters.

Restarting, pausing and resuming the system

To pause the system, select the entire MAS in GOAL, and press the pause button. To resume, select the MAS in GOAL and press run. You can also run and pause environment or agents individually but of course you can get errors if the agents run while the environment is paused so this is not recommended.

To restart the system, do the following

1. In GOAL, kill the MAS by selecting the MAS and pressing .
2. In Repast, Press **Reset**, or choose a new map from the Change Map menu.
3. Run the MAS in GOAL as described above.

Loading and creating new maps

Using the map editor

The Map Editor is a tool for editing maps for the BW4T server (Illustration 5). Double click the jar file <GOAL>/environments/BW4TServer/mapeditor.jar to start the map editor.

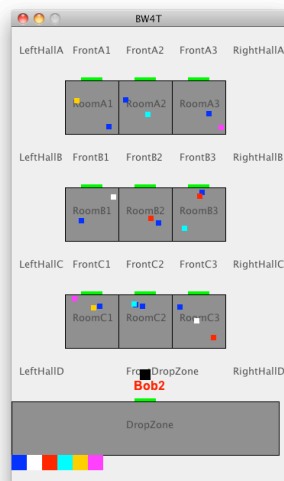


Illustration 3: A typical map as visualized by the server

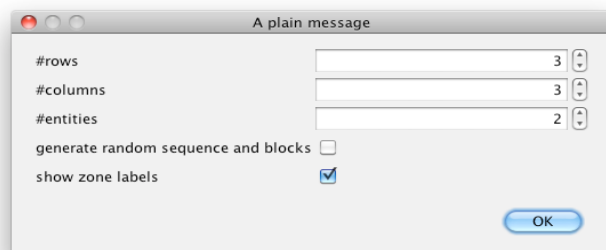


Illustration 4: The map size dialog

After starting up, the map size dialog (Illustration 4) appears, which can be used as follows. Enter the number of rows.

1. Enter the number of columns.
2. Enter the number of entities. This indicates the maximum number of bots that this environment can create. These can be either java-, goal- or humanbots. The entities will be

named BotX where X runs from 1 to the number of entities you requested. They are all placed initially in front of the drop zone.

3. Choose whether you want random goal sequence and block generation. If you check this box, the server will add a random goal sequence with $\frac{2}{3} * N$ blocks where N is the number of rooms on your map. It will also add $2.5 * N$ blocks of random color to the rooms. In the random option, you can not edit any details on the map.
4. Choose whether you want the zone labels to be visible in the server render window.

If you did not select 'generate random sequence and blocks', you proceed to the map editor page (Illustration 3).

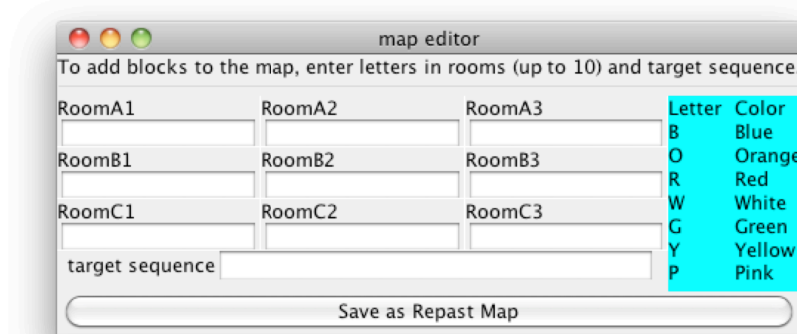


Illustration 5: Map Editor

In the Map editor GUI (Illustration 3) you can add blocks to rooms and to the target sequence. To add blocks to rooms, click in the text box under the room label and type one or more letters, where each letter represents a block of a particular color. To add blocks to the target sequence, type the letters of the colors you want to add in the target sequence text box. Do not type spaces between letters as every character is associated with one block. When you're done, press save.

Manual Editing

It is also possible to manually edit a map as it is a plain text XML file.

1. Copy an existing map file in the <SERVER>/maps folder to a new file.
2. Open the copied map with a text editor. You can then edit the colors in the rooms. Each <blocks>COL</blocks> line inside a <zones> of type ROOM adds another block to the room. The currently available colors are BLUE, ORANGE, RED, WHITE, GREEN, YELLOW AND PINK. A room has place for at most 10 blocks.
3. You can change the number of rooms by adding or removing <zones> of type ROOM and position the rooms correctly on the map by editing the <x> <y> <width> and <height> inside the room's <boundingbox>. Also you need to add a <door> properly positioned on the border of the room.
4. You can edit the goal sequence by adding <sequence> items to the map, with colors as mentioned above.
5. You can prevent multiple entities to enter corridor zones by putting true in the item <oneBotPerCorridorZone> in the map.
6. You can add entities to the environment by creating more <entities> items. Make sure they have a unique <name> and that their start position is in a different <zone> if you have

- `<oneBotPerCorridorZone>` set to `true`¹.
7. You can let the server pick a random sequence of a given length by setting the `<randomSequence>` to a positive value. These random blocks are added to the existing sequence items and the random blocks are placed randomly on the map.
 8. You can let the server pick random extra blocks to be placed in the rooms on the map by putting a positive number in the `<randomBlocks>` in the map. Note that this addition is on top of all blocks that are already placed in the map; so normally you leave the room zones empty when using this option.
 9. You can set per-zone visibility of the zone namelabel in the server map renderer, using a `<renderOptions> <labelVisible> false </labelVisible> </renderOptions>` block to disable visibility.
 10. Save the map in the `<SERVER>/maps` directory.
 11. Edit the map initialization parameter for the server to your new map file. See the section on customizing the server settings. If you now start the server and your new map should be loaded.

Human Interface

BW4T bots can also be controlled by humans instead of agents. Figure 6 shows a screenshot of the user interface (GUI) that enables humans to control a bot. Whereas the Server interface shows the positions of all bots and all blocks (Figure 7), the human GUI only shows the bot's own position and the blocks in the room where the bot is in.

The human interface has, from top to bottom, the following four areas:

1. the button area showing bot buttons having the names of the other bots and an 'all' button
2. the map area showing the rooms, hallways and drop zone,
3. a row of blocks showing the current state of the sequence
4. a chat area showing exchanged text messages.

¹ If you use `oneBotPerCorridorZone`, you should use `attach` all entities in the map to an agent. If you do not, the unused entities remain invisible inside the `frontdropzone`, blocking access to the `dropzone` for the other entities.

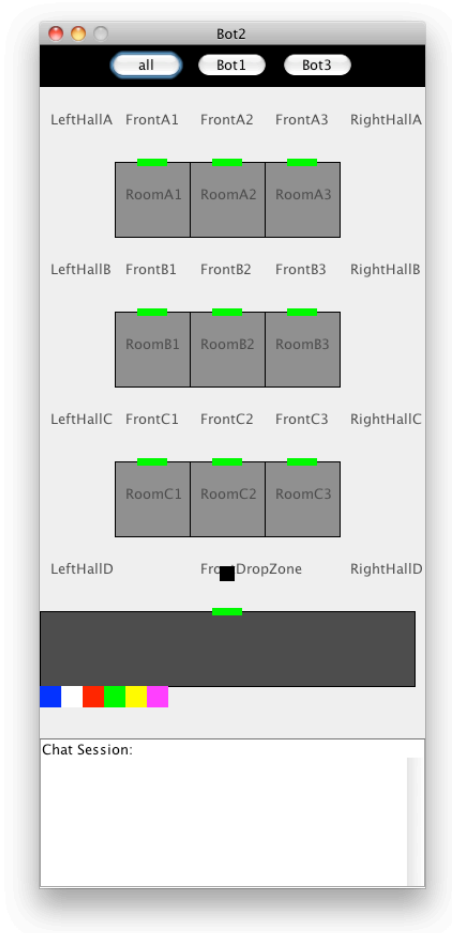


Illustration 6: Human User Interface

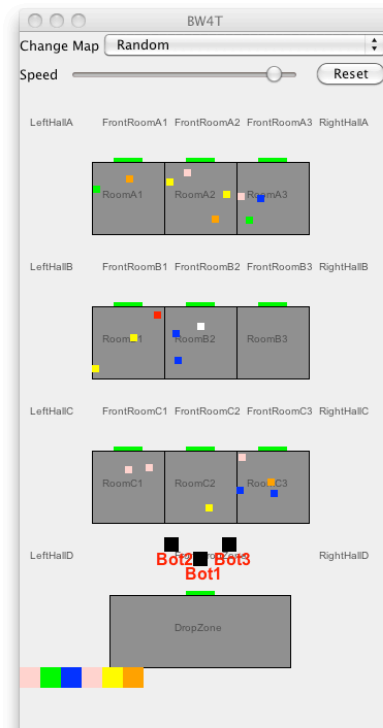


Illustration 7: Server Interface

Enable Human Interface(s)

The simple way to start a humanbot is to load and run the provided `bw4thuman.mas2g`.

To add a humanbot manually to an already existing MAS, you need to enable one or more human interfaces, as follows:

- Modify the **humancount** parameter in the `mas2g` file. Check the section 'advanced run settings' for details.
- Set the **launchgui** parameter in the `mas2g` file true. Check the section 'advanced run settings' for details.
- Ensure that the `humanbot.goal`, `message.mod2g` and `messageTranslation.pl` files are in your project folder.
- Add the line `"humanbot.goal"` to the `agentfiles` section in your `mas`.
- Add the line `"when [type=human,max=1]@env do launch *:humanbot"` to the `launchpolicy` section of the `mas2g` file. Note that in order for the `humanbot.goal` program to work correctly, your agents should get the same name as the EIS entities. The `">*` in this `mas` line ensures this.

Usage

To use the interface, the user clicks with the left (occasionally the right) mouse button in the GUI. Depending on where the user clicks, different menus appear. The user then picks the appropriate action from the menu to execute that action. Below the possibilities are explained.

Click on bot buttons

By clicking on a bot button, the user can send a request or question to that bot, or to all bots if the 'all' button is used. Typical requests are to go to a room or to find a particular color. Typical questions are how far the bot is from completing the request.

Click on room or dropzone

By clicking on a room, the user can order his bot to go to that room, tell everybody something about that room or ask something about that room.

Click on blocks

By clicking on a block (particularly, those below the drop zone), the user can tell everybody something about that block, or ask others for information about that block.

Click on hallway

By clicking on a hallway, the user can point to an exact (X,Y) location to go to. Also it is possible to tell all others roughly where one is in the hallway.

Click on chat area

Clicking on the chat area gives a number of standard answers: yes, no, don't know, ok, etc. It is not possible to use free text because the non-human agents can only process these pre-specified messages. The specification document gives more details on how messages are processed by non-human agents.

Running on multiple computers

If you want to run BW4T on multiple computers you should designate one of these computer as the server. On this computer you can start the server as described in the first section of this guide. You must use RMI messaging (check the GOAL Run menu) to allow other GOAL runtimes to connect². You need to check a few things in the MAS that you use here:

- specify a map such that the environment resets when you start up the MAS
- make sure that the map that is used has enough entities to accommodate all agents in all computers that want to connect
- make sure that the entities get the proper type, by specifying the proper agentcount and humancount.

The other computers will then function as client. Create a MAS file for each of these, and configure

² If you do not do this, the system may seem to work properly but the agents running in the various GOAL instances will not be able to communicate with the GOAL send action and the humanbots will not work properly.

this MAS as follows (see also `bw4thuman.mas2g` in the GOALagents directory of GOAL):

- set the **serverip** and **serverport** initialization parameters to the ones that the server is listening on (default for the server is localhost and port 8000).
- Set the **humancount** and **agentcount** parameter on each client to reflect how many human or agent players that client should load.
- Use `humanbot.goal` for human agents
- use `env = "BW4T2/BW4TClient.jar"` in the environment section. Do not connect to an already running environment in another MAS. This is because BW4TClient creates GUIs for humanbots, on the machine where it is running.
- Check that the launchpolicy picks up the proper entity type, so use 'human' if you want to attach to human entities etc.

Distributed Human GUIs

This section describes how to run a set of distributed human GUIs such that they all communicate through GOAL. Before running a set of distributed Human GUIs with GOAL, make sure that the server is installed on one machine. GOAL itself is not necessary to start the server, but it is convenient to also install GOAL there so that you have the server startup scripts. Furthermore, make sure that the server map can contain a sufficient number of entities.

To run a set of distributed Human GUIs with GOAL, do the following:

1. Start the server
2. For each machine where you want to have a human GUI:
 1. Start GOAL
 2. open the MAS file of the `bw4thuman.mas2g`
 3. fix the `serverip` to correctly point to the machine ip number where the server runs
 4. select **Run RMI (Distributed)** in the GOAL Run menu.
 5. Start the MAS
 6. on the prompt “Enter middleware host”,
 - enter localhost for the first GOAL machine,
 - or enter `rmi://<ip-number>` where `<ip-number>` is the ip number of the first GOAL machine.

Programming a BW4T agent

To program your own BW4T agent, use the same actions as specified in the demorobot. You can choose to change the pre- and post conditions of each action.

Percepts are retrieved automatically by GOAL, see the percept specifications for what percepts you can expect.

In order for messages received by other GOAL agents to appear on the GUI of your agent, you should add the following lines to your GOAL agent’s code:

1. Add the following line to your knowledge base:
`#import “messageTranslation”.`
2. Add the following line at the end of your goal file.
`#import “message.mod2g”.`

3. Add the following line to the start of your event module:
`if bel(true) then message.`
 This will make sure that the message module that was just imported will be run first when entering the event module.
4. Your own message handling code should be performed after this line and should delete any messages after handling them. Otherwise they will be continuously posted to the GUI as they are not deleted by the message module. If you don't do any message handling yet you can add the following line below the one provided in step 3 to delete all received messages:
`forall bel(received(Agt,Msg)) do delete(received(Agt,Msg)).`

Testing your agent

In order to test your agent you should edit the `bw4t2.mas2g` file as follows.

- Add your agent to the agentfiles list.
- Add a line to the launchpolicy (copy the line of the demorobot and replace 'demorobot' with the name of your agent).

Make sure that you set GOAL to launch the desired number of your agent. Also make sure that the initialization parameter of agentcount is not set to 0 as then only humanbots will be loaded.

Note that besides in the `bw4t2.mas2g` file, the number of agents is also specified map that you use. This number determines the maximum amount of bots that can appear on the map. If the maximum number of agents to be launched as specified in the `bw4t2.mas2g` file is *bigger* than the number of agents specified in the map, some of the agents will not appear on the map and not be part of the team.

Log file

Repast logs the following for each run into a file. The filename is "BW4TXXXX.txt" where XXX is a large number to make the filename unique. The file is saved in the <SERVER> directory. It contains:

1. sequence: goal sequence (which block colors are to be dropped)
2. room: initial blocks per room
3. action: log of each action of a bot, with timestamp in ms since january 1, 1970
4. sequencecomplete: total time to complete task. Begin time is determined by first incoming action. End time is determined by the last block of the sequence dropped.
5. agentsummary: for each agent:
 - the bot type
 - # correct drops in dropzone
 - # incorrect drops in dropzone
 - total time of standing still
 - #messages to other agents
 - #rooms entered

We will write info as soon as possible to the log file, so that you at least have some log info even when the system is killed before the end is reached (sequence completed).