BlocksWorldForTeam using with Java

Maaike Harbers, Wouter Pasman, Birna van Riemsdijk
November, 2013

Introduction

This document describes how to install Blocks World For Teams (BW4T) for use with Java. 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 Java agents, etc. The server uses Repast, software to simulate virtual environments, to do part of this administration. The *client* is Java, which runs a multi-agent system (MAS) and connects to the server. The agents in the Java 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, configure the MAS file, and creating and testing your own Java agent.
We use the following names to refer to directories of BW4T:

- <SERVER> refers to the directory where you installed the server scripts.
- <CLIENT> refers to the directory where you installed the client scripts.

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:\RepastSimphony-2.0-beta** and on OSX **/Applications/Repast-Simphony-2.0.0-beta**. If you choose to install it in another directory, see Custom server installation 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) or **BW4TServer.sh.command** (OSX) file in a text editor.
2. Change the Repast variable to the directory that you installed Repast in.
3. Save your changes

Install server run scripts

1. Download and extract the BW4T2Server.zip to any directory on your hard drive. We refer to this directory as the <SERVER> directory.
2. On OSX set execute rights for .sh files as follows:

   a) open a terminal

b) go to the unzipped folder
c) execute **chmod 777 BW4TServer.sh.command**
d) close the terminal

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) or **BW4TServer.sh.command** (OSX). This should open the Server window (Figure 1). Note, this takes some time.

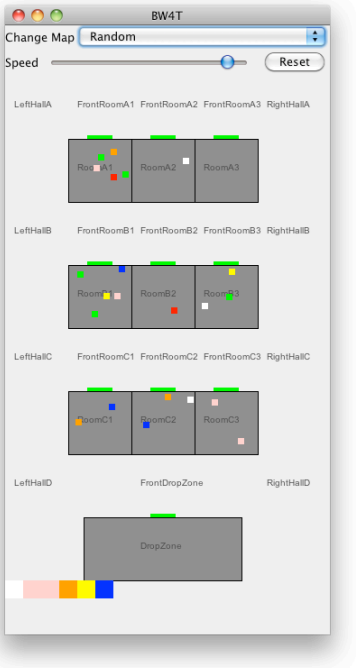

Illustration 1. Server window showing the actual state overview window of the Environment.

Advanced run settings (Server)

The default settings for the server can be changed in its startup script: **BW4TServer.bat** (Windows) or **BW4TServer.sh.command** (OSX). 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.

Installing the Client

1. Download and unzip the BW4TClient.zip to a convienient directory. We refer to this directory as <CLIENT>.
2. On OSX set execute rights for .sh files as follows:
    a. Open a terminal
    b. Go to the <CLIENT> directory
    c. execute **chmod 777 BW4TClient.sh.command**
    d. close the terminal

Running the Client

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

1. Go to the <CLIENT> directory.
2. Run the BW4TClient.bat file (Windows) or BW4TClient.sh.command (OSX) in the environments folder to create agents.

Advanced run settings (Client)

The following default settings for the client 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).

To change these settings, type *-setting value* in the command line, e.g. *-serverip localhost*, or *-agentcount 3*

Running different agent classes

To run different agents, each having a different class, you need to start up two separate clients in the launch script. Do this as follows

1. Go to the <CLIENT> directory
2. copy the startup script (BW4TClient**.****) to a file script2.**.
3. Open script2.** in a text editor.
4. Change the agentclass in the second script to use your other agent class
5. Check that the agentcount in the two startup scripts add up to the available number of entities in the map.
6. Run the BW4TClient.** file
7. Run the script2.** file

You can make more copies of the startup script to run more different classes.

Creating an agent

When creating an agent, make sure to add the BW4TClient.jar to your build path, as this contains the classes necessary to create a custom agent.
Your agent should extend the BW4TAgent in the nl.tudelft.bw4t.agent package. The BW4TAgent already contains methods to perform the available actions in the environment. See the TestAgent included <CLIENT>/src/ for an example of an agent that visits all places.
You should override the run() method from the BW4TAgent class, this method will be called when your agent has been connected to an entity on the server. This method should therefore perform the reasoning process of your agent. Your agent will stop when this method is exited.
Percepts can be retrieved by calling the getPercepts() method contained in the BW4TAgent class, this will return a list of percepts. Each percept in the list has a name which can be retrieved by getName(). The parameters of the percept are retrieved by calling getParameters, with each parameter being either a Numeral or Identifier. The int or long value of a Numeral can be gotten by calling getValue().intValue() or getValue().longValue(). The String value of an Identifier can be gotten by just calling getValue(). See the percept specification for what parameters to expect for each type of percept.
Please see the Specifications document and the javadoc in <CLIENT> for more details on the java interfaces. The javadoc also explains how to translate BW4TMessages to strings and back.

Testing your agent

Set your development environment to compile your agent into the folder that contains the **BW4TClient.jar** file. This is to prevent you having to copy the compiled files there each time, although you can still choose to do that.

The typical command to compile the java program (run from the BW4TClient directory) is

```
javac -d . -classpath BW4TClient.jar src/nl/tudelft/bw4t/agent/TestAgent.java
```

The -d option is to tell the java compiler to compile into the current directory. The BW4TClient.jar has "." in its classpath so it will find your classes if they are in the same directory as BW4TClient.jar.

Edit the **BW4TClient.bat** (Windows) or **BW4TClient.sh.command** (OSX) file to

- make sure you have the –agentclass parameter following the classname of your agent ("nl.tudelft.bw4t.agent.TestAgent"), see Advanced run settings (Client).
- Make sure the parameter -agentcount is not set to 0, as then only human players will be loaded.

After this you can run the server and then the client and your agent should be loaded.

Note that besides in the **BW4TClient.bat** (Windows) or **BW4TClient.sh.command** (OSX) file, the number of agents is also specified map that you use. This number determines the amount of bots that appear on the map. If the agentcount in the **BW4TClient** file is *smaller* than the number of agents specified in the map, some of the bots that appear on the map will not be used (they are

visible but do not display any behavior). If the agentcount in the **BW4TClient** 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.

Loading and creating new maps

Using the map editor

The Map Editor is a tool for editing maps for the BW4T server (Illustration 3). Double click the jar file <GOAL>/environments/BW4TServer/mapeditor.jar to start the map editor.

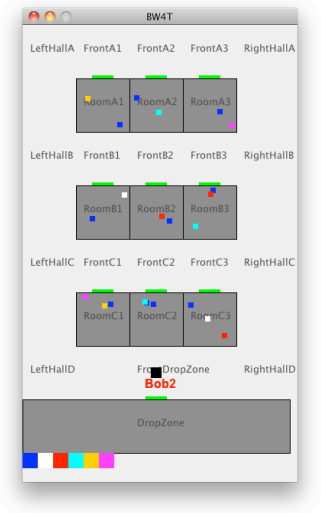

Illustration 2: A typical map as visualized by the server

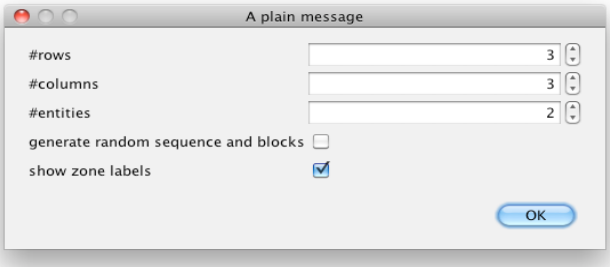

Illustration 3: The map size dialog

After starting up, the map size dialog (Illustration 2) 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 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 4).

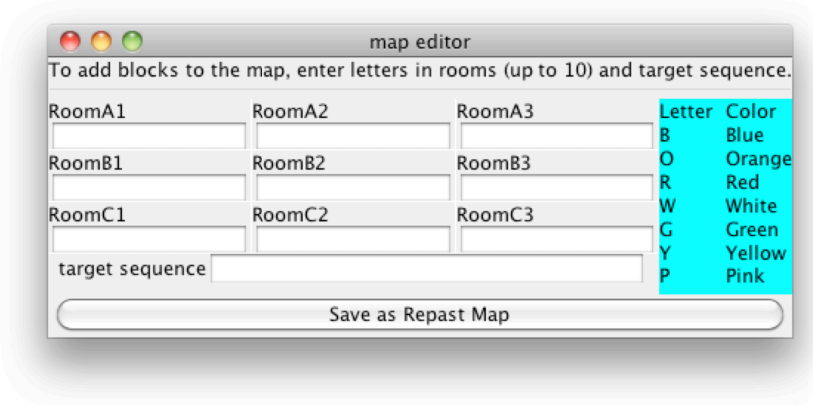

Illustration 4: Map Editor

In the Map editor GUI (Illustration 4) 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 <oneBotPerCorrridorZone> 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

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.

<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.

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 connect[2]. 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 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.

Human Interface

BW4T bots can also be controlled by humans instead of agents. Figure 5 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 6), 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.

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.

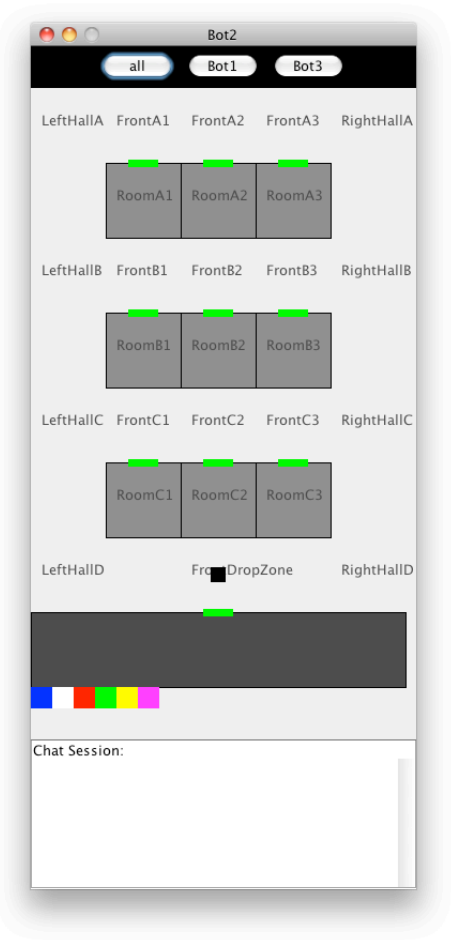

Illustration 5: Human User Interface

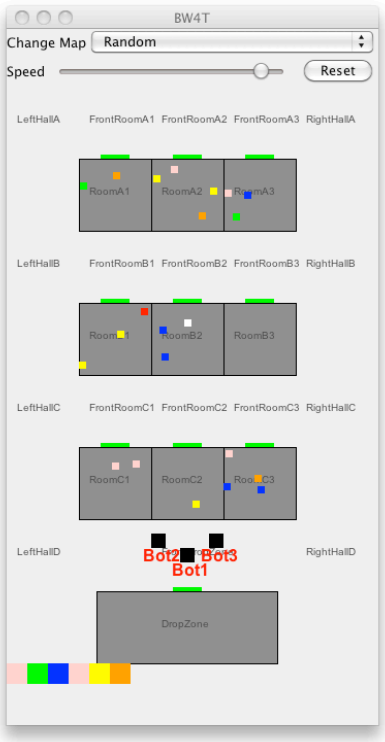

Illustration 6: Server Interface

Enable Human Interface(s)

To enable one or more human interfaces, you need to modify the **humancount** parameter in the startup script. Check the advanced run settings of the Java instructions document for details.

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.