MSNMago

Krzysztof Olczyk

January 2008

Contents

List of Figures

1 Aim of project

Aim of a project was to built an interactive command-based bot for Microsoft MSN Messenger. Main utilization of a bot is to provide remote controlling capabilities through the messenger service. It is going to provide means to interact remotely with system shells, to exchange files and insepct system by obtaining a screenshots.

2 Project specification

2.1 Running msnmago

The prerequisite necessary to run **msnmago** is to have separate MSN or Windows Live account which bot is going to operate on. Having registered at Microsoft you may run msnmago to login at account myaccount@windowslive.com with password 1234, executing the following command:

```
msnmago.py myaccount@windowslive.com 1234
```

Alternatively, you may create file **login.conf** in msnmago's directory that consists of two lines: first being a login and second being a password. For more details, see 2.5.

2.2 Available commands

Msnmago is controlled by commands sent as Messenger messages. Most of the commands support two invocation models. In first model, the command consists of the line consisting of a single word (terminated by space) identifying the command and rest of line being the data for a command.
Example:

```
shell ls
```

In the second model, the command consists only of a single word identifying the command. After having executed such a command, it is *gaing the control* over msnmago and every message sent is used to "feed" command with data. In order to return back to the main state of msnmago the string special string **go back** must be sent.

For the complete list of commands, see figure 1.

2.3 More on 'shell' command

Shell command lets the user remotely execute and control shell commands. If a shell command is given as a argument to **shell** command, it is executed and all data it produces on standard output and standard error streams is sent to a client as a message.

If **shell** command is sent without parameters, the shell mode is entered. In this mode any shell command can be executed, including

Command	Description	Data format	Invocation Model[1]
info	Displays msnmago version info	n/a	n/a
echo	Echoes the data sent	plain text to echo	both
shell	Executes shell command. See section 2.4 for more details.	command to execute	both
screenshot	Request the screenshot to be sent back as JPEG file	n/a	n/a
downloads	Downloads the file from download repository. See sectopm 2.4 for more details.	See section 2.4	n/a
broadcast	Broadcast message to all authorized clients of msnmago	textual message to be sent	both
die	Terminates msnmago	n/a	n/a
set	Sets given parameter to the given value.	set *param_name* to *value*	n/a
allow	Allows the user given by the parameter to control the bot.	email of contact	1st
disable	Disables the specific command to the specific user.	disable *command* to *username*	1st
enable	Reenables the specific command to the specific user. Stronger than **disable**	enable *command* to *username*	1st

Figure 1: List of msnmago commands

those expecting data on standard input. If no command is running - the message sent is interpreted as a shell command to be executed. After having executed command that does not quit immediately and expect data on standard input, every following message will be sent to command's standard input pipe.
Note: The new line at the end of message is stripped when sending it to commands input pipe. You may force the new-line character at any part of message typing **NL**. In the same manner you may order sending the *escape* character typing **ESC**.

There are several tokens that has special meaning while in the shell mode[2]:

- **APPEND NL** - lets msnmago append new-line at the end of any message sent (the same as typing NL at the end of all the following messages.
- **DO NOT APPEND NL** - switches off appending of new-line.
- **ABANDON** - abandons the currently executing process and waits for new shell command to execute.

2.4 More on 'downloads' command

The **download** command lets the user interacting with msnmago download the file from the repository, ie. content of the subdirectory **downloads** located in the directory where msnmago resists.

After typing **downloads**, msnmago turns into the file downloads mode. In this mode you may type one of the following commands:

- **list files** or **ls** - List all files in the current scope
- **list dirs** - List all dirs in the current scope
- **get *file*** - Downloads file *file* from current directory
- **cd *dir*** - Enters the directory *dir*
- **up** or **cd ..** - Enters the parent directory

Note: Msnmago possesses a feature to upload any file. It is not a part of download command and my be done any time by simply sending the file in MSN Messenger. File is accepted automatically and placed in the users subdirectory in directory **uploads** in msnmago's path.

2.5 Configuration file

Msnmago may be configured by providing two configuration files:

[2]Notice that all the shell mode special verbs are typed in capitals

msnmago.conf This file is in fact a script of commands to be executed just after loging into the MSN service. In fact, it may contain any command that may be sent to msnmago by user. However, the most useful commands include:

- **set** - command to specify bot's display name and status, eg.:

```
set name to ''Mr robot''
set status to online
```

- **allow** - the vital command of the configuration. Lets the user choose which contacts are allowed to interact with it and which are not, eg.:

```
allow mydude1@hotmail.com
allow sbelse@windowslive.com
```

- **disable** and **enable** - controls which commands are available to which users, eg.:

```
disable shell to dude@hotmail.com
disable * to loser@hotmail.com
enable info to loser@hotmail.com
```

The commands in **msnmago.conf** file are separated by semicolons. Any data between characters # and ; are considered comments.

login.conf Simple file containing just logon information, namely the username (Microsoft Passport ID) constituting the first line of a file and Base64 encrypted password as a second line.

2.6 Defining custom commands

Msnmago features extensibility of its command set. User can define his/her own commands being either a macro of msnmago's commands, python code or realized by an external application. In order to define new command, the command **def** is used.

Macro Command being a macro of msnmago commands is defined in the following manner:

```
def macro_name as macro
@
        # commands being a body of macro;
        # separated by semicolons go here;
@
```

Any occurence of #1, #2, #3, etc. is replaced by the respective argument passed to the command.

For instance, the macro called *myinfo* may be defined to print string *My version is:* followed by the output of **info** command:

```
def myinfo as macro
@
        echo My version is:;
        info;
@
```

Python implementation In order to create a command implemented in python the following code snippet must be adapted:

```
def macro_name as pythonic
@
         # python code goes here
@
```

While executing the python code the following two implicit global function are available:

- **sendmsg(s)** - sends a message given as a string argument *s* to the user who executed the command
- **do(s)** - executes msnmago's command given in a string *s*

As well, global tuple **args** is present containing the arguments passed to the command.

Sample code implementing command returning current date at computer's clock would look like following:

```
def today as pythonic
@
         from datetime import date
         sendmsg("Today is %s :>" % str(date.today()))
@
```

Realized by external application The command may be also defined to be realized by an external application. In this case, all parameters are passed to standard input and standard output is sent back as a message to user.

The format of definition is:

```
def command_name as application ''applicationname''
```

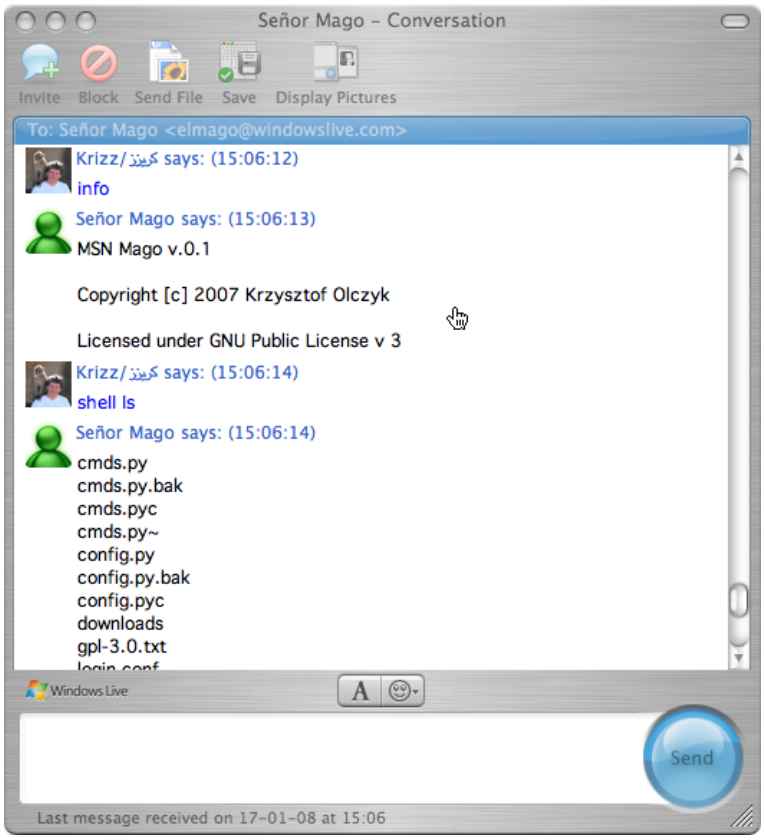

Figure 2: Msnmago - interacting with a bot