

Improvement of the hedgehog controller

Amalia Sirbescu*, Katharina Stelzer, Daniel Wieser, Christoph Hofer, Florian Baumgartner, Lukas Miehl, Jan Fenyö

HTL (school for higher technical education)
Department of Computer Science
2700 Wiener Neustadt, Austria

*E-Mail of the corresponding author: sirbescu.amalia@student.htlwn.ac.at

One of our robots we have chosen to work with for the PRIA Open® competition is the hedgehog. During our work we realized that the hedgehog controller sometimes has some saving errors. For example, after improving the program, which was just written, the hedgehog controller overwrites the actual program. In this work we want to present different solutions on how to deal with these kinds of and other problems. The simple backup copy, the more advanced backup copy, how to manually generate a *.pyc file from a *.py file, the public-keys-procedure and cross-compiling (a method, which works better for C/C++ programming). Each and every solution has its advantages and disadvantages and needs specific circumstances, so that it can work.

1. SIMPLE BACKUP COPY

The first solution that probably comes to everybody's minds on how to deal with saving errors is to make a backup copy. Every time before closing the program, all the program has to be copied and inserted into an empty text document. Of course, this way is pretty impractical. That one time the work has been forgotten to be copied, the controller will erase it, or only parts of it, and it will have to be written again. The advantage, if this happens, is that the new program may be better than the last. But, if, for example, more than one programs are being worked on, this way of securing can get confusing and repetitive.

2. ADVANCED BACKUP COPY

Especially, if more programs are being worked on, this solution will be easier to work with. Instead of pasting the program on a simple text document, this solution shows how to copy whole files with programs to the hedgehog. First of all, it needs to be made sure there is a FTP-Client-Software installed

on the computer. We are using WinSCP, a free FTP-Client-Software for Windows (the following pictures are based on WinSCP).

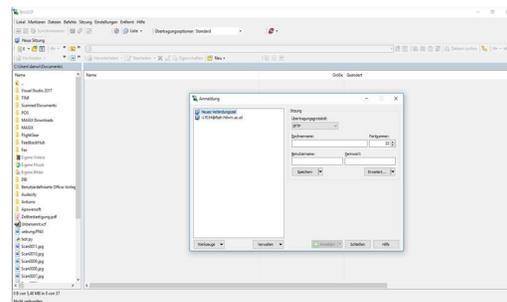


Figure 1 Starting Page WinSCP

After starting the FTP-Client-Software, a connection between the software and the hedgehog has to be developed.

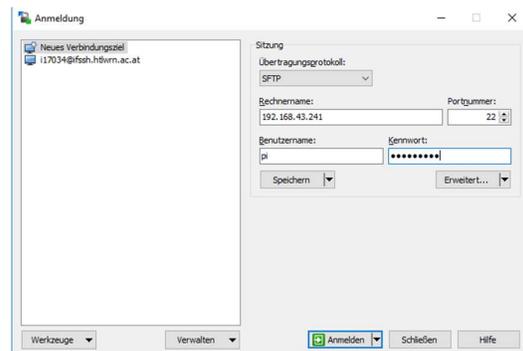


Figure 2 Connecting with hedgehog

(As you may notice in the pictures above, our WinSCP software language is German, but this shall not have any impact on your understanding.) The host name field must be filled with the IP-address of the hedgehog. The user name is “pi” and the associated password is “raspberrry”. Finally, after clicking on the “login” button, a connection with the hedgehog will be built.



Figure 3 Hedgehog program structure

Afterwards the hedgehog's program structure will be displayed on the desktop. The programs, which were written before on the Web-IDE, will be in the folder "hedgehog-programs".

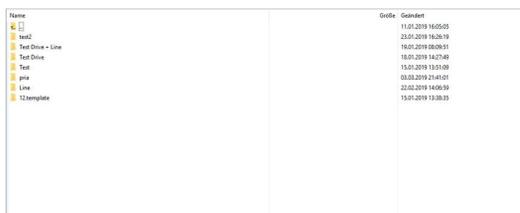


Figure 4 Example: our programs in "hedgehog-programs"

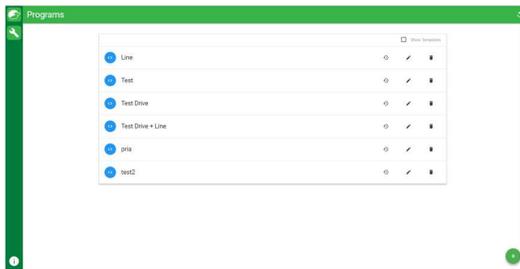


Figure 5 Example: our programs in Web-IDE

Now the programs can be written on the PC and the programs are being copied in the directory.

3. MANUALLY GENERATE *.PYC FILE FROM A *.PY FILES

If the operating system is Linux, "compileall" can be used, if wished. The following command will go recursively into sub directories and make *.pyc files for all the python files it finds. The "compileall" module is part of the python standard library, so nothing extra has to be installed for using it. This works the same way for python2 and python3.

`"python -m compileall"`

With the following command a program, which in our case just prints out the number "123", will be copied on the hedgehog.

`"scp program.py pi@ip-address"`



Figure 6 Copy on hedgehog

After inserting it in the hedgehog, the next step is to switch to the hedgehog by typing in the command below.

`"ssh pi@ip-address"`

And the according password, which still is "raspberrry".

The following command has to be used to execute the program on the hedgehog:

`"python xxx.pyc"`

(In python3 a subdirectory "_pycache_" will be created, where the *.pyc files can be found and started from.)

4. AUTHENTICATION WITH PUBLIC-KEYS

The public-keys-procedure is used for simpler working, since the password does not need to be typed in every time. Asymmetric cryptography (encryption) will be used to authenticate the user. The user has a public key, which can be found in the folder "~/.ssh/authorized_keys" of the target system. The private key, however, is placed in the "id_rsa" folder in a directory called "~/.ssh" on the local system. When using the public-key-method to log in on an SSH-server, the server will send a random generated challenge to the client. The server then encrypts the data block with the client's public key. If the client is able to encrypt its cipher with the associated private key, the identity of the user will be confirmed. In the directory "~/.ssh" the key will be created with the following command:

`"ssh-keygen -t rsa -b 4096"`

With a simple click on the "Enter"- button, the name can be left on standard.



Figure 7 Creating the key in the ssh-directory

be made possible and they allow the user to edit the programs comfortably with the desired IDE on the local PC or Laptop. Especially when using public-key authentication, working will become very convenient, as any program can be started from the PC easily with just one command.