CS336 - Final Project

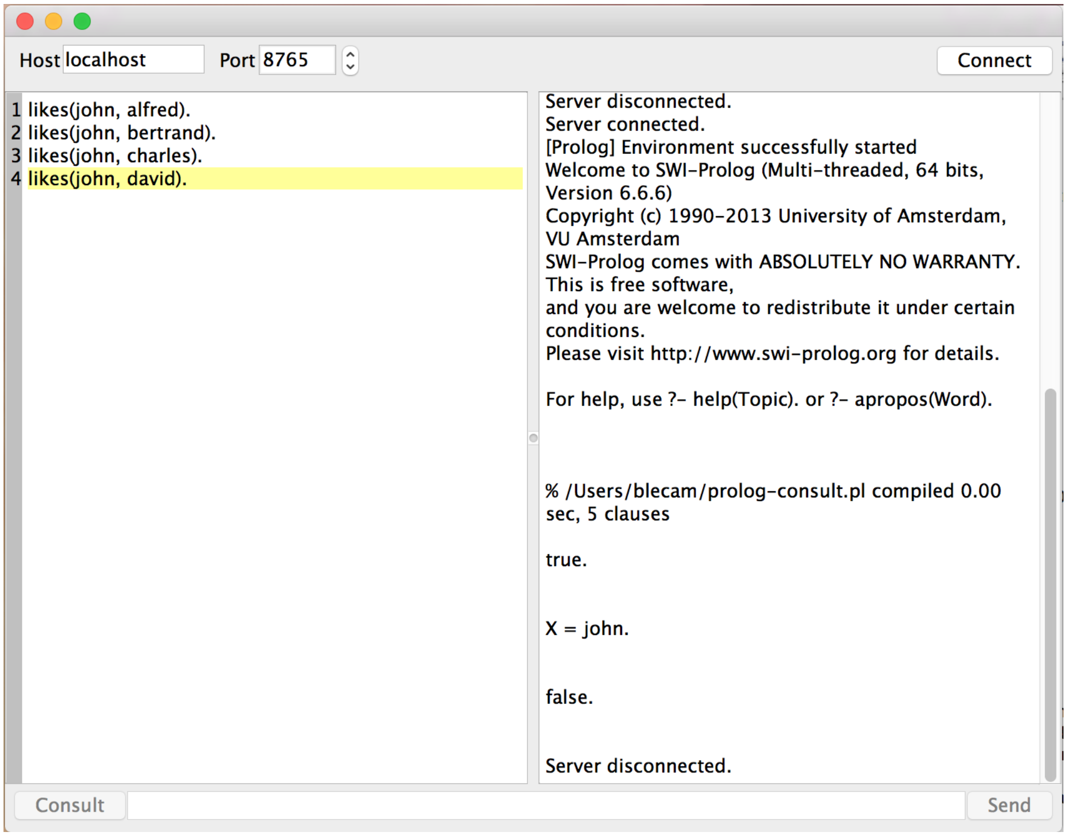
Ben Le Cam and Ben Lipson

December 15th, 2014

- Project Name: Prolog GUI
- Our project was to build a GUI for prolog that would not require Prolog to be installed on the local machine. Therefore, a client-server architecture was necessary.
- The client is the only software that has an interface. Our GUI contains a host and port edit fields as well as "connect" button on top of the page to connect to the server. Then in the middle of the application, we have on the left the Prolog code and on the right the server output. At the bottom of our GUI, we have the "Consult" button and we have a Prolog input field along with a "Send" button.
- We first designed the architecture by laying out our requirements:
  - A code editor
  - No need to have Prolog on a local machine
  - Being able to consult a file, and send commands to Prolog
  - Eventually have the possibility to use different multiple versions of Prolog depending on the server configuration

  Then we started by testing to interpret Prolog in C++. We made it work pretty quickly but we eventually realized that a good long term vision would be to have the possibility to chose between multiple versions of Prolog. From there, we tried calling "gprolog" from Qt and it worked out really well. However, we could not get gprolog output but not to input data. We thus tried with SWI-prolog and we could get it to input our data. After testing the Prolog-C++ communication, we built the server around it by using the TCP protocol. We also
- Ben Le Cam built the back-end server as well as the communication part on the client. Ben Lipson built the code editor and GUI.
- We used Qt, an open source C++ framework to develop our server and client. Qt has a lot of features such as network sockets, threads or GUI classes.
- The code is organized as following on the Client software:
  - client.h and client.cpp: class to handle the communication to the server via TCP sockets.
  - codeeditor.h codeeditor.cpp : class that represents the text editor of the GUI

  and as follows in the Server software:
  - prologserver.h and prologserver.cpp: class that represent a server identity, it has a list of threads.
  - prologthread.h prologthread.cpp : class that is used to handle one client from the server. A thread contains a process of prolog.
- We tested a simple hello world in class, as well as some more complex predicates as seen in the screenshots.
- The code is available on GitHub at Prolog Client et Prolog Server.
- Here are screenshots of the softwares:

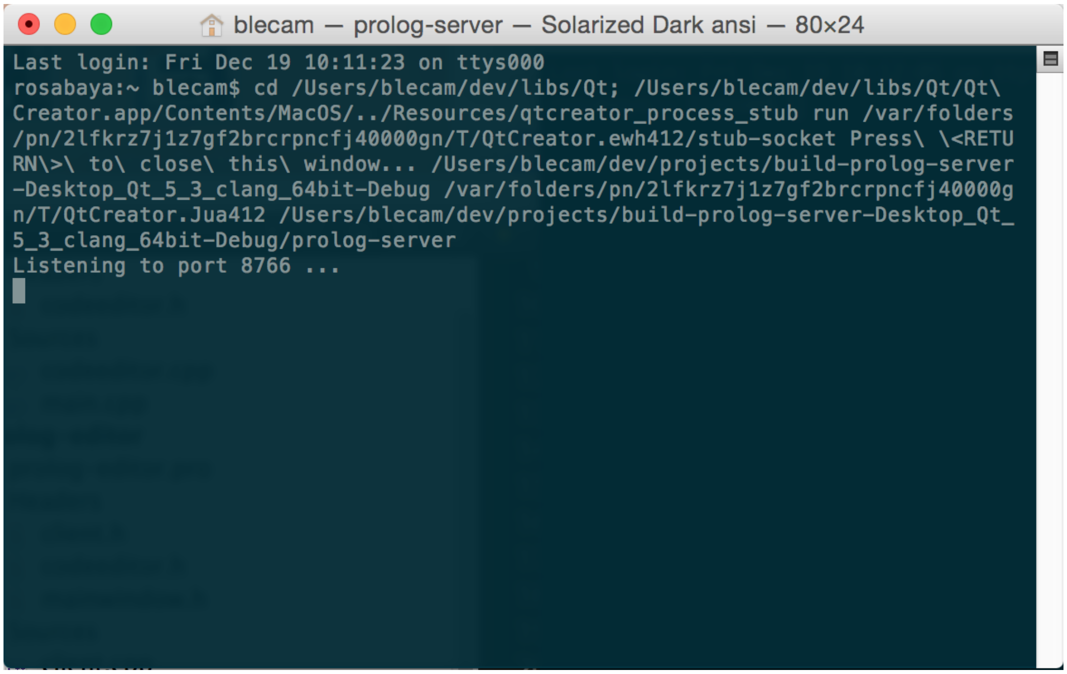


Host localhost Port 8765

```
likes(john, alfred).
likes(john, bertrand).
likes(john, charles).
likes(john, david).
```

```
Server disconnected.
Server connected.
[Prolog] Environment successfully started
Welcome to SWI-Prolog (Multi-threaded, 64 bits,
Version 6.6.6)
Copyright (c) 1990-2013 University of Amsterdam,
VU Amsterdam
SWI-Prolog comes with ABSOLUTELY NO WARRANTY.
This is free software,
and you are welcome to redistribute it under certain
conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).



% /Users/blecam/prolog-consult.pl compiled 0.00
sec, 5 clauses

true.


X = john.


false.


Server disconnected.
```

Figure 1: The client side.



```
Last login: Fri Dec 19 10:11:23 on ttys000
rosabaya:~ blecam$ cd /Users/blecam/dev/libs/Qt; /Users/blecam/dev/libs/Qt/Qt\
Creator.app/Contents/MacOS/../Resources/qtcreator_process_stub run /var/folders
/pn/2lfkrz7j1z7gf2brcrpncfj40000gn/T/QtCreator.ewh412/stub-socket Press\ \<RETU
RN\>\ to\ close\ this\ window... /Users/blecam/dev/projects/build-prolog-server
-Desktop_Qt_5_3_clang_64bit-Debug /var/folders/pn/2lfkrz7j1z7gf2brcrpncfj40000g
n/T/QtCreator.Jua412 /Users/blecam/dev/projects/build-prolog-server-Desktop_Qt_
5_3_clang_64bit-Debug/prolog-server
Listening to port 8766 ...
```

Figure 2: The server side.