

MakeABot

CSE 331 – Data Structures

Written by:

Daniel Dostal
William Harle
Brandon McGirr

Abstract

The MakeABot program provides an easy-to-use GUI interface to create an AIMBot. The MakeABot program also make use of data structures to allow for storing, accessing, and sorting of user data and responses. Finally, the MakeABot program fully integrates with AIM to allow for logging of conversations and for the MakeABot to converse with other users.

Key terms

AIM – Acronym for America On-Line Instant Messenger, the messenger program our program utilizes.

Bot – In this context, term used to describe an automated user for AIM.

GUI – Graphical User Interface.

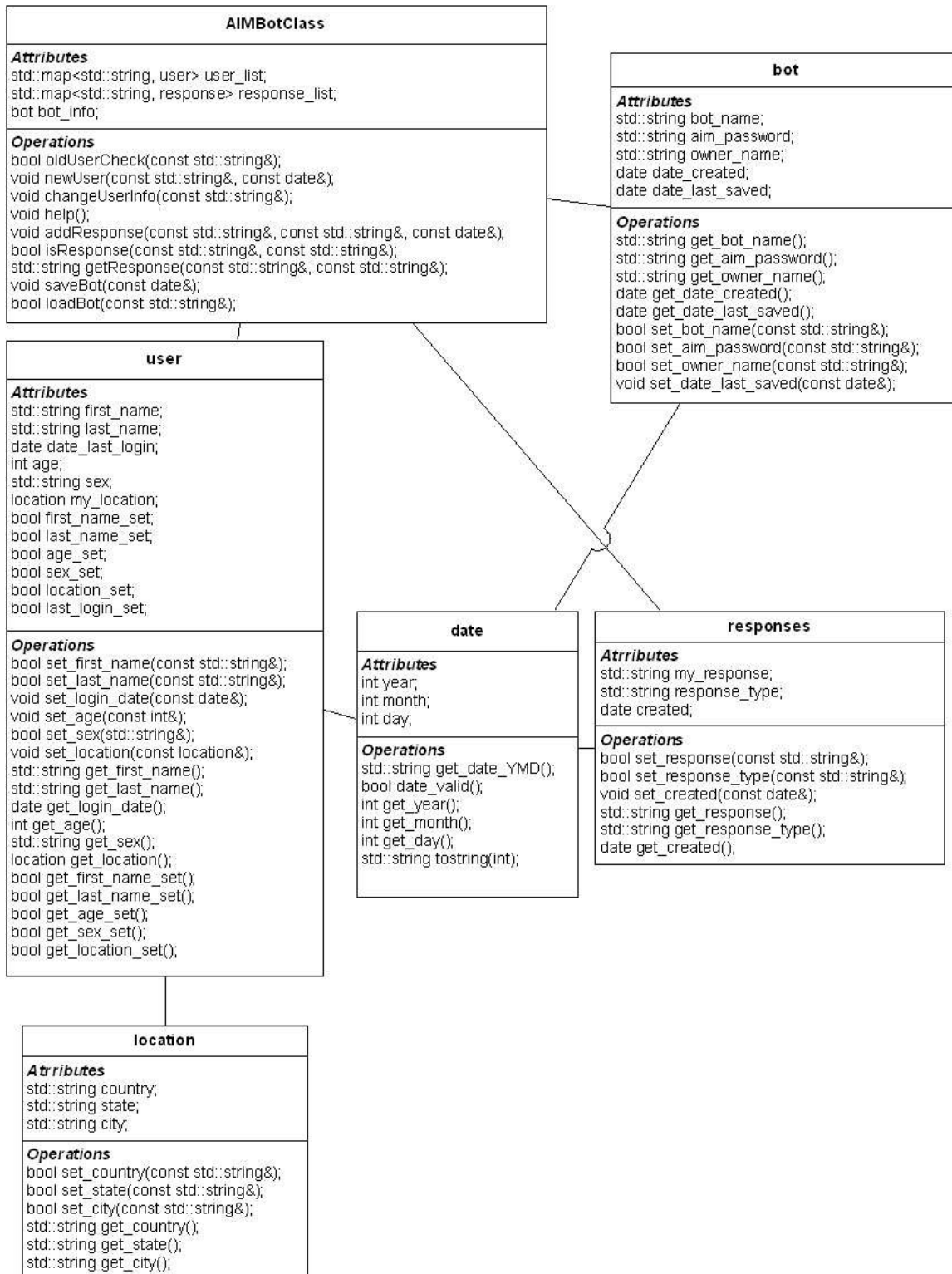
General Description of Project

Our project grew as the result of a desire for an easy to use and easy to expand AIMBot creation program. All the programs that currently create AIMBots are very hard to expand and build upon, and require the user to know a scripting language such as PERL. We wanted to make a Bot creator program that required no previous programming knowledge and was easy to manipulate and change. This required us to create our own GUI, which we did by using Windows forms and to use the source code from the open-source AIM program called gAIM. Our references included the data structures textbook we used in class, How to Program C++ by Deitel, and Teach Yourself .NET Windows Forms by Chris Payne. For websources, we used <http://cppreference.com> and <http://cplusplus.com>.

Program Functionality

Our desired program functionality includes saving and loading operations for AIMBot configurations as well as for user information. We also want our program to be able to log onto the AIM service and be able to log conversations between the user and the Bot while allowing the Bot the ability to respond to the user. The desired program platform is on any Windows based system that runs on the WIN32 architecture such as Windows 95, Windows 98, Windows 2000, and Windows XP. Our potential users include AIM users and those that wish to create AIMBots of their own. The GUI used for our program is Windows Forms, and the core language used is C++.

UML



Data Structures and Algorithms

The key data structure that we used was maps. Maps were used for the sorting and storing of user lists and responses. We decided to go with maps because we found that using keyed entry is extremely useful. Also, maps provide us with a simple binary search which takes advantage of its pair structure.

We chose the map over any other stl data structure (in particular vectors and lists) because of the nice binary search algorithm that the find() function provides. The program is much more lookup based and not insertion based, so a list would not be nearly as useful as a vector or map. The index function of vectors is only useful when things are based on an integer index, but we use strings as identifiers. Thus the vector was not useful. The clear choice is the map for its fast lookup based on a string as a key.

To conduct error handling, we used a lot of boolean values to determine if something worked properly. For instance, if a name with either the /#/ or /###/ delimiters is entered, then the set function will return false and not enter the name. Otherwise this would make the loadBot() function work improperly.

Conclusions

We learned a lot from working on this program, both pertaining to the program in specific and to software development in general. .NET Forms is a powerful WIN32 API development tool. The underlying code integrated fairly easily. Maps and C++ strings are used throughout and we have a much greater understanding of these two data structures. However, the real lessons learned were about software development. The planning phase needed a great deal more attention. If we had investigated the scope of our project more thoroughly, we would have determined that we should have scaled it down a lot. We also would have been able to more effectively divvy up the responsibilities between the team members. As it stands too much relied on the actions of one member and the other two members were frantically trying to work on the parts that the third member had not had time to work on.

References

H.M. Deitel and Deitel, P.J. *How to Program C++ 4th Edition*. New Jersey: Prentice Hall, 2003.

Kohl, Nate. *C/C++ Reference*. <http://cppreference.com>.

Payne, Chris. *Sams Teach Yourself .NET Windows Forms in 21 Days*. Indiana: Sams, 2002.

The C++ Resources Network. *CPlusPlus Resources*. <http://cplusplus.com>.

William Ford and Topp, William. *Data Structures with C++ Using STL 2nd Edition*. New Jersey: Prentice Hall, 2002.

Biographies



William H. Harle Jr. is a Computer Science Engineering major at the University of Notre Dame, where he is currently enrolled in his junior year. When he isn't spending his time programming his life away, Bill likes playing guitar, listening to music, watching movies, and falling in love.



Brandon McGirr is a Junior Computer Science major at Notre Dame. When not posing with stabbed produce or programming, he finds himself developing his 3D modeling skills, writing scripts and animating for cartoons, and in the ultimate act of time-sinking, acting.



Daniel Dostal is a junior Computer Engineer major at Notre Dame. When not mastering the art of vhdl, one can find him playing various old school video games, or listening to his extensive, yet refined music collection. He is also known as a Jesus look-a-like.