International Journal of Advancements in Technology

Das et al., Int J Adv Technol 2016, 7:1
DOI: 10.4172/0976-4860.1000150

Research Article Open Access

Windows Based Graphical Objective C IDE

Suvadeep Das*, Gaurav Singh and Balraj Kumar

Lovely Professional University, Jalandhar, Punjab, India

Abstract

On the growing demand of mobile application development, the learning and knowledge of the basic languages used in the development becomes a necessity. For the Android development, we need the basics of Java technology, for Windows phone application, the Dot Net and C# technology is promoted and for iOS development, the basics of Objective C is the main prerequisite. For most of the Windows based system users, it becomes difficult to practice the Objective C programs, since Windows provide no specific tool for it. Based upon this issue, by learning and doing a tremendous research on the working of different available Objective C IDEs and the methodologies used by them, we have come up with a Windows based User-friendly Objective C IDE that provides a cross platform advantage with graphical features.

Keywords: Objective C; GCC; GNUStep; OBJCC IDE; Xcodel; Windows

Introduction

In the early 1970s, Mr. Dennis Ritchie at AT&T Bell labs developed C Programming language but to utter shock, this programming language didn't gain any popularity until the late 1970s. The main reason found behind the failure in making this language popular was the unavailability of C compilers for commercial-use outside the AT&T Bell labs. Soon after the development of UNIX operating system, which was written almost in C language, has made the language popular. Later in early 1980s Cox came up with a new language based on SmallTalk-80 language and the language was named as Objective C.

Inheriting the basic features of C language, Objective C is the addition of OOP concepts in it, including extensions to the functionality of the language by supporting object creation and manipulation. Objective C was further adopted by NeXT with the licensing agreement of the language in 1988 and hence used as their main language. NeXT, again, developed libraries to use it with their main language for the development of operating system known as NeXTSTEP, further with which iOS and MacOS X were derived. Objective C includes object-oriented features to C language, bringing an entirely new syntax (Figure 1).

In the year 1992, Free Software Foundation (FSF)'s GNU, the open source software foundation included support for Objective C language development environment and in the year 1994 NeXT and Sun-Microsystems released a standardize specification for NeXTSTEP known as OPENSTEP. The Free Software Foundation implemented OPENSTEP as GNUStep and later they developed a Linux version which included the Linux kernel and GNUStep development environment and named it LinuxSTEP.

Till Date Achievable Objective C IDEs (Integrated Development Environments)

Xcode IDE for Mac OS

In 2003, Apple Inc. was first to develop a suite of software development tools named as Xcode. Xcode is an Integrated Development Environment with which softwares for MacOS X and iOS are developed. Xcode is capable of supporting number of languages like C, C++, Objective C, Objective C++, Java, AppleScript, Python, Ruby, Rez and Swift, together with the involvement of variety of programming models such as Cocoa, Carbon and Java in it, for user to have multiple framework options. Objective C being the primary language used in Xcode for the development of iOS and MacOS X applications. With the help of Clang Compiler, used by Xcode, the programs written in Objective C, C and few other languages were converted into executable code.

GNU Compiler Collection (GCC) using GNUStep libraries

The GNU Compiler Collection (GCC) is a compiler system developed by GNU Project. It supports various Programming languages. The Free Software Foundation (FSF) gives license to GCC under General Public License (GNU GPL). GCC, initially was written as the compiler for GNU Operating System. GNU Step which is an open implementation of OPENSTEP, is used as the backbone for running Objective C language. With the help of GCC compiler, we can compile and run Objective C codes in windows using the following command in DOS command:

The file is created in notepad and saved with ".m" extension. It is basically a command line based compiler in which everything needs to be done using Command Prompt tool (Figure 2).

Tutorials point's coding ground online cloud IDE

Tutorials Point's Coding Ground [1], earlier known as *compileonline.com,* combined more than hundreds of online IDE of popular languages. Its Open source IDE is fully based on cloud where you can edit, compile, execute and share your projects. Keeping GNU libraries as their backbone of the Objective C online IDE, they provide much functionality which is normally present in IDE offline software like create new projects, create directories, and create new files. We can also download and share the projects that were created using it.

Objectives

Primary purpose of Objective C language is to develop applications for iOS and MacOS X, hence, in Microsoft windows operating system there is just a scarce development of this language. Looking at the limited popular IDEs available for Objective C, we can make out that in windows the compiler of objective c is not as featured as that of Xcode in MacOS X. For Microsoft windows, we have a GCC compiler, which is a CLI (command line interface) compiler with limited scope of functionalities. There is a lack of a good environment for development

***Corresponding author:** Suvadeep Das, Lovely Professional University, Jalandhar, Punjab, India, Tel: +91 9056613339; E-mail: suvadeepdas91@gmail.com

Received November 24, 2015; **Accepted** November 27, 2015; **Published** December 15, 2015

Citation: Das S, Singh G, Kumar B (2016) Windows Based Graphical Objective C IDE. Int J Adv Technol 7: 150. doi:10.4172/0976-4860.1000150

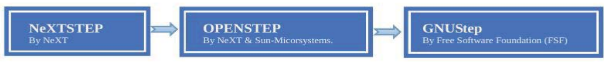
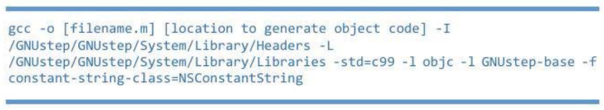

NeXTSTEP
By NeXT
OPENSTEP
By NeXT & Sun-Micorsystems.
GNUStep
By Free Software Foundation (FSF)

Figure 1: Lineage of GNUstep.



```
gcc -o [filename.m] [location to generate object code] -I
/GNUstep/GNUstep/System/Library/Headers -L
/GNUstep/GNUstep/System/Library/Libraries -std=c99 -l objc -l GNUstep-base -f
constant-string-class=NSConstantString
```

Figure 2: GNU compiler collection.

of Objective C programs.

The main objectives of our research are as follows:

1. Analyze the available IDEs and the research work done on Objective C in previous years so as to understand the working of IDE and the necessary libraries that are required for the compilation of the code in Microsoft Windows.

2. Implementation of a GUI based IDE with all the basic functionalities which are required to successfully develop and debug Objective C codes under Microsoft windows environment.

Literature Review

Brandon et al. [2] discussed about a case study in which Glasgow University took help from NeXTSTEP to upgrade their computers. IN their paper they have described about implementation and porting of NeXT MusicKit, they also have cloned NeXT SoundKit – The SndKit on various number of platforms, both old and new. They have outlined some of the strengths and weaknesses of the kit and demonstrated several applications which had made the transition from NextSTEP to WebObjects/NT and MacOS X.

Muqri et al. [3] exposed the technology students to the fascinating comparison between Objective C language which is used for developing applications in iPhone and Java language which is used for developing apps for Android phones. This paper explains about how the learning and teaching module is progressive in learning of students and also demonstrated that developing smart phone applications are not just limited to fun and gaming applications but to develop apps for students through which they can inform and educate themselves about the topics which are traditionally not covered by conventional software courses.

Gupta et al. [4] have done research on various languages she studied the history and influence of one language to another. In their paper they have discussed that in the beginning C language was developed and the purpose was to give programmers a coding friendly environment. The C language had some drawbacks and Objective C was developed to overcome the drawbacks. Objective C also had some issues and then C++ had arrived. It had features of C and Objective C both.

Gulhane et al. [5] have presented a code converter application for cross platform development. They have developed an online examination system that will reduce evaluation time and give the results in seconds. In this paper they discussed about developing an application which can run on both android and iPhone using their code converter tool. The can be used for other languages also which demands conversion. They have talked about android and iPhone only in this paper. By using their tool, code has to be developed only once reducing the code development time and energy by half.

Chisnall et al. [6] presented with an interoperable framework for allowing code written in C (potentially with inline assembly), Objective-C, Smalltalk, and higher-level domain-specific languages to coexist with very low cognitive or performance overhead. Our implementation shares an underlying object model, in interpreted, JIT -compiled and statically compiled code among all languages, allowing a single object to have method implemented in any of the supported languages. We also describe several techniques that we have used to improve the performance of late-bound dynamic languages.

Chisnall et al. [7] discussed a new Objective C runtime with 10% decrease in the already existing GNU runtime library which is of 11,688 lines. The newer library is 15% closer to the GNU runtime Library. The new library supports various functionalities that are present in GNU runtime, he had described them in the research article. Garbage collection is not yet supported in the proposed runtime. He had released the code for the runtime under 3-clause BSD license.

Problem Definition

In this section, we describe several challenging problems based on our experiences in Objective C learning. These areas are focused on the lack of development of objective c compiler in any other platforms Mac OS. Since 2007, Objective C is used for developing iOS device applications. It becomes usual to have a Mac system in-order to learn Objective C most of the time. However, there is a free implementation of OPENSTEP framework present for windows named as GNUStep but it becomes bit difficult for a normal user to work on it. We need to use a text editor to write the code and later on with command line, we compile and execute it by writing a huge command every time. When the share of apple's smartphone grew to 42.8 percent from 39.2 percent in a year as per cnet.com, the popularity of the objective c language arouse even more. Developers have been using Objective C to build mobile apps since March 6th, 2008 with iPhone OS. However more than 60% of the IT persons are using Microsoft Windows, where it comes the need to have a cross platform environment to develop Objective C codes in Microsoft windows which can enable the developers to learn Objective C without having a Macintosh system.

Motivation

The motivating factor for the development of GUI based Objective C IDE is the lack of a good compiler for the development of Objective C programs in Microsoft's Windows environment. As we were learning Objective C in our university that has a MAC Lab and Xcode installed to it but it was very difficult for us to develop codes in our Microsoft Windows installed laptop systems as 99% of the students had Window based systems. Therefore, we had only two options for practicing and compiling the Objective C codes, either to use GNUStep command line compiler or Tutorial Point's Codeground Online IDE. Codeground requires a stable and fast internet connection. Along with these problems, we came up with an idea of developing a GUI based IDE for Microsoft Windows which can compile and execute the Objective C codes similar to the IDEs available for executing C or C++ languages.

Proposed Solution

An ideal method to solve this cross platform dilemma is the ability to design and develop a GUI based IDE that runs on Microsoft windows based operating systems. Looking at the Coding Ground by Tutorial's point that have used GCC's GNUStep libraries for the compile and execution of the Objective C codes, our OBJCC (Objective C Compiler) compiler is also using the same libraries. By installing following three files in windows operating system, we can execute and compile Objective C codes using command line tool Microsoft's Command Prompt [8]:

GNUStep core

It consists of GNUStep-make, GNUStep-base, GNUStep-gui & GNUStep-back. In order to compile Objective C code, GNUStep-make and GNUStep-base is required. With core, we can be able to open shell window and then execute commands.

GNUStep devel

This will give us GCC with Objective C and GNUStep libraries which allow us to develop codes, compile and execute them.

GNUStep Msys system

This gives us the core UNIX like tools which are required for performing operations in windows with GNUStep. After having all the three files installed in Microsoft windows based operating system, we can now compile and execute Objective C codes. After this, the objective was to develop GUI based IDE. OBJCC IDE is developed using .NET technology using C# programming language.

The IDE includes the following features:

A code editor with expression highlighting feature and line number (Figure 3a).

An output window at the bottom to show any message while execution (Figure 3b).

Menu options for creating new file, saving, compiling and for execution (Figure 3c).

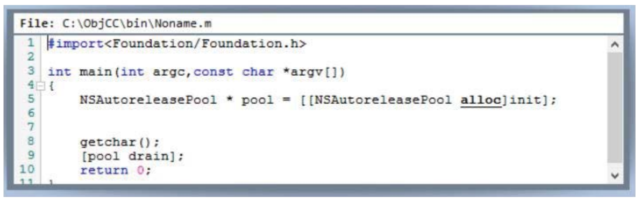
Right click menu with some basic options (Figure 3d).

Basic Editing options are also included like Quick Find, Find and Replace etc (Figure 3e).

How OBJCC IDE works?

OBJCC (Objective C Compiler) IDE is using GCC compiler as its backend and following is a flow chart representing a general process flow in GCC's GNUStep compiler (Figure 4a):

GNUStep Core, Devel and Msys System are the three essential libraries in which all the classes and objects are present to provide a runtime environment for developing and compiling Objective C programs.

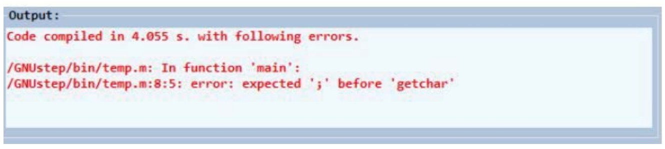


```
File: C:\ObjCC\bin\Noname.m
#import<Foundation/Foundation.h>

int main(int argc,const char *argv[])
{
    NSAutoreleasePool * pool = [[NSAutoreleasePool alloc]init];


    getchar();
    [pool drain];
    return 0;
```

Figure 3a: Expression highlighting.



```
Output:
Code compiled in 4.055 s. with following errors.

/GNUstep/bin/temp.m: In function 'main':
/GNUstep/bin/temp.m:8:5: error: expected ';' before 'getchar'
```

Figure 3b: Code output area.

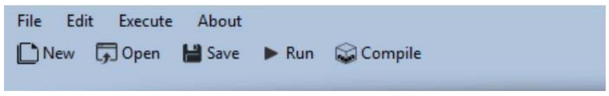
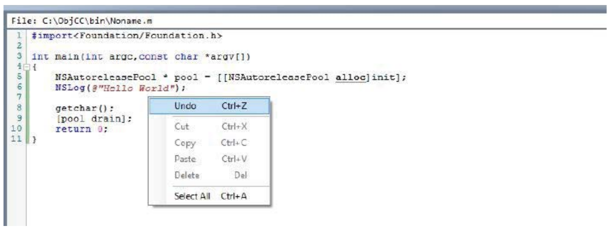


Figure 3c: Menu options.



```
File: C:\ObjCC\bin\Noname.m
#import<Foundation/Foundation.h>

int main(int argc,const char *argv[])
{
    NSAutoreleasePool * pool = [[NSAutoreleasePool alloc]init];
    NSLog(@"Hello World");

    getchar();
    [pool drain];
    return 0;
}
```

Figure 3d: Right click menu.

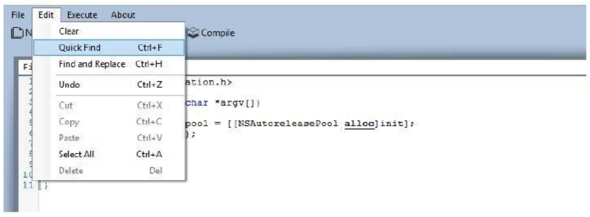


Figure 3e: Editing options.

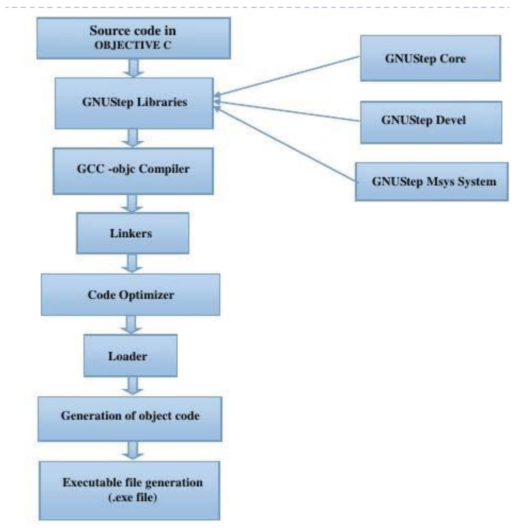


Figure 4a: GNU compiler collection execution flow.

OBJCC uses the above GCC compiler process as its back-end, by having a GUI based interface. In flow chart it can be represented in the Figure 4b:

In OBJCC IDE, for executing an Objective C code, first task is to create a new file and after creating a new file in a code editor Objective C code is to be developed. The code editor has all the basic functionalities which are required for writing a well formatted code. It automatically formats the code and highlight the functions and predefined keywords, etc. After successfully writing a code, our next task is to compile the

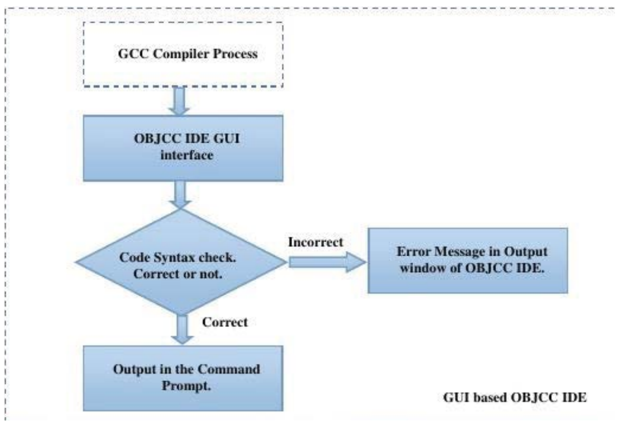

GCC Compiler Process
OBJCC IDE GUI interface
Code Syntax check. Correct or not.
Incorrect
Error Message in Output window of OBJCC IDE.
Correct
Output in the Command Prompt.
GUI based OBJCC IDE

Figure 4b: OBJCC extended process flow.

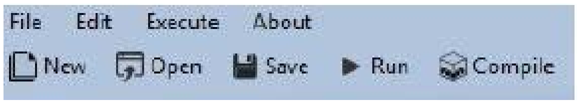
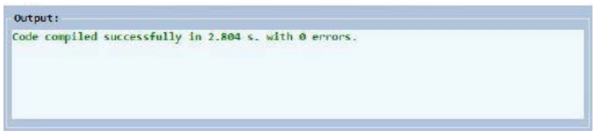
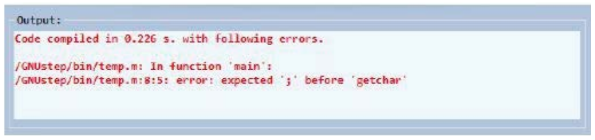


Figure 4c: Menu options.



```
Output:
Code compiled successfully in 2.804 s. with 0 errors.
```

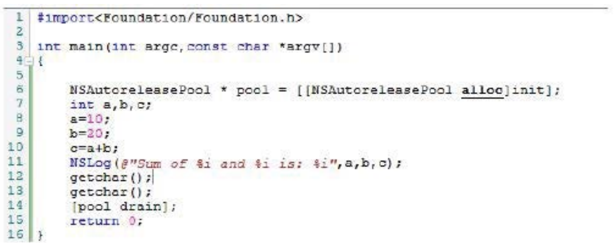
Figure 4d: Code output area.



```
Output:
Code compiled in 0.226 s. with following errors.

/GNUstep/bin/temp.m: In function 'main':
/GNUstep/bin/temp.m:8:5: error: expected ';' before 'getchar'
```

Figure 4e: Code output area.



```
#import<Foundation/Foundation.h>

int main(int argc,const char *argv[])
{

    NSAutoreleasePool * pool = [[NSAutoreleasePool alloc]init];
    int a,b,c;
    a=10;
    b=20;
    c=a+b;
    NSLog(@"Sum of %i and %i is: %i",a,b,c);
    getchar();
    getchar();
    [pool drain];
    return 0;
}
```

Figure 4f: XCode editor.

code. For compiling the code, we have a compile button provided in the top menu bar or we can use shortcut also Alt+F9 (Figure 4c).

For compiling, the code will be sent to the GCC compiler back-end and the GCC compiler with the help of GNUStep libraries compiles the code and returns the message. The message is then redirected to the output window.

If the code does not contain any syntax error then output will be shown as follow (Figure 4d):

And if the code has some syntactical errors then the errors will show in the output window with the line number, as follows (Figure 4e):

After compilation of the code, our next task is to execute or run the code. Execution of the written Objective C code is done by clicking on Run button present on the top menu. If we directly click Run button without prior clicking on the compile, then too the Objective C code will directly compile first and then it will show the output. The output will be shown in a Command Prompt. The format of the output will be same as shown in Xcode.

Following is a sample code executed in Xcode and OBJCC IDE, output of the program is shows as follows:

Code to be executed (Figure 4f):

The output of the above program in OBJCC IDE is as follows (Figure 4g):

The output of the program in MacOS X's Xcode is as follows (Figure 4h):

As the output is similar in both the IDEs, this made it possible to use OBJCC IDE as an alternate for Xcode in Microsoft Windows based operating systems.

Comparison

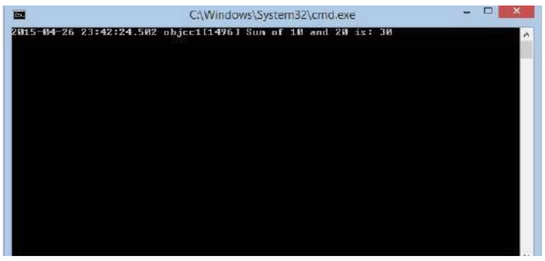
In this section, the table below represents the clear comparison between all the available Objective C Compiler. After few more researches, I came up with the comparison table between the existing compilers as shown below in Table 1. As mentioned before, the OBJCC IDE is able to run in windows working environment along with some additional feature and GUI interface. It also violates the necessity of any online platform for the working of objective c programs.

However, we haven't yet included the multi-file support feature in our OBJCC IDE, but would be considered in future developments.

Future Scope

OBJCC IDE meets the basic requirements for successfully compilation and execution of the Objective C codes in Microsoft Windows based operating system. There are some additional features that can be added later on to the OBJCC IDE to make the development more user friendly and efficient like adding multi-file support by which we could be able to develop multi-tier programs. Multiple file support will enable us to develop header files, properties, modular functions in different files and call them in a file through referencing.

The other features those can be added to the OBJCC IDE are the Intellisense feature. With Intellisense feature, we don't have to remember all the syntax; it will automatically give suggestions while



```
2015-04-26 23:42:24.502 objcc1[1496] Sum of 10 and 20 is: 30
```

Figure 4g: Output in OBJCC IDE.

	OBJCC IDE	**XCODE IDE**	**GCC CLI compiler**	**Tutorials points coding ground**
Windows support	✓	✗	✓	✓
Offline	✓	✓	✓	✗
GUI Interface	✓	✓	✗	✓
Function	✓	✓	✗	✓

Table 1: IDE comparison.

writing the codes. So, in future work there can be a planning to extend the features of OBJCC IDE by including the above features.

Conclusion

In Microsoft window based operating systems, there was a lack of GUI based IDE for compiling and execution of Objective C programs. We overcame with this issue by successfully implementing OBJCC GUI based IDE that is developed using .NET technology and C# language, which have enabled us to write and compile Objective C codes in Microsoft Windows. The IDE supports most of the features which a basic IDE supports and the output produced by the proposed IDE is similar to Xcode in MacOS. In the proposed IDE, there is no need to write a line for compilation like in GCC compiler, there are a Run button and a Compile button available to do this just by clicking. Finally, the proposed model can replace GCC compiler by a GUI based IDE which is more user friendly.

References

1. http://www.tutorialspoint.com/codingground.htm
2. Brandon S, Leigh MS (2000) Next steps from NeXTSTEP: MusicKit and SoundKit in a new world.
3. Muqri MR, James RL (2012) Objective-C versus Java for Smart Phone Applications. American Society for Engineering Education
4. Gupta S, Puneet B (2013) Comparative study of C, Objective C, C++ programming language. Int J Eng Comp Sci 2: 202-206.
5. Gulhane A, Shailesh M, Shaunak S, Chinmay G, Rakhi B (2014) Code converter for android and ios. IJARCSSE 4: 1152-1156.
6. Chisnall D (2014) Smalltalk in a C world. Sci Comput Program 96: 4-16.
7. Chisnall D (2009) A Modern Objective-C Runtime. J Object Tech 8: 221-240.
8. http://www.gnustep.org/windows/installer.html