Evaluation of the impact of using an IDE to learn an object-oriented programming language

Thomas Gay - 74672

20 septembre 2018

Introduction

According to the *TIOBE index for September 2018*, five of the top ten programming languages are object-oriented. The job website Indeed.com provides *The 7 Most In-Demand Programming Languages of 2018* in which four are object-oriented and the two first ones are Java and Python. This sector is increasing and offers many opportunities. Faced with this strong job offer, many people want to learn object-oriented programming languages. Before writing the first lines of code, people have to set up their working environment and chose a tool to help them to learn. This paper is trying to quantify the impact of such a tool, in particular integrated development environments (IDEs).

Scope

In this experiment, two type of tools are compared. When people want to teach or learn an object-oriented programming language they have to choose an environment, i.e. a simple text editor or an integrated development environment (IDE). The first one is very simple to install and use but it doesn't provide any help. IDE is a bit more complicated to use because of the number of features, it takes times to control them but they provide more help and it is easier to find errors during the programming process.

Objective. The purpose of the experiment is to compare the two tools and find which one is more efficient to learn an object-oriented programming language like Java or C++. Learning a programming language means to have basic skills to be able to implement short application in a reasonable time.

Object of study. The object of study is persons without experience in programming. Their ability to learn software sciences and programming and their determination are objects of this experiment.

Quality focus. The main parameters studied in the experiment are the time to create and implement an application, the number of errors, and the code optimization through the number of lines of code (LOC). About the last criteria, lines of code are relevant to know in how many steps

Context. The experiment is focused on the comparison between text editor and IDE to learn an object-oriented programming language. The experiment includes 40 persons who haven't skills in programming. They are volunteers to participate. That means all participants have palatability to programming and with this lack of randomization of subjects the experiment is not fully into control. Participants have followed the same course of object-oriented programming. Four environments (two text editors and two IDEs), all under Linux Ubuntu, were used during the experiment: Gedit, GNU Emacs, IntelliJ IDEA Community and Eclipse Oxygen for Java. In both categories we use two different tools to prevent variation from tool efficiency or inefficiency. Then they were divided into groups to learn the basis of each tool.
Of course, it is an off-line experiment, in that way the process of learning and evaluating subjects is under control.

Experimental design

The experiment attempted to answer the following questions:

1. Do IDE reduce the time of implementing programming tasks? This question is, probably, the most important one to compare IDE and text editor. The goal is to learn faster an object-oriented programming language and create software and applications. With which one of these tools is it faster?
2. Do IDE increase the correctness of the delivered solution? The correctness is about the number of errors in a solution and the degree of the errors. Are they blocking the software execution?
3. Do IDE increase the quality of the solution submitted? This quality can be measure with the number of lines of code which provide hint about the code optimization.

Variables

Independent variables. This experiment has two independent variables. The first one is the use of documentation for IDE features and, the second one is the use of documentation of Java.

Dependent variable. The experiment has four dependent variables:

- T: Time to implement a short application excluding environments setting.
- LOC : Number of lines of code.

The correctness of a solution is measured by:

- C: Number of submissions of a solution with a fault.
- C': Number of submissions with a blocking error.

Hypotheses

With this experiment, the null hypothesis says that learn object-oriented programming language is not better with an integrated development environment than a text editor. On the contrary, the alternative hypothesis express the fact that to learn with an IDE is better in term of implementing time, number of errors and code optimization.

Table 1: Tested Hypotheses

Dependent variables	Null hypothesis (H_0)	Alternative hypothesis (H_1)
Time to implement application	T(IDE)≥T(TE)	T(IDE)<T(TE)
Num of submissions of solution with a fault	C(IDE)≥C(TE)	C(IDE)<C(TE)
Num of submissions of solution with a blocking error	C'(IDE)≥C'(TE)	C'(IDE)<C(TE)
Num of line of code	LOC(IDE)≥LOC(TE)	LOC(IDE)<LOC(TE)