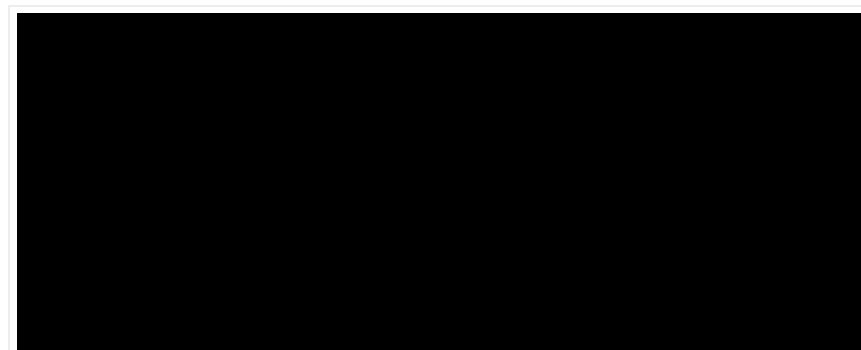


10 most common Java programming mistakes revealed by Big Data

What's the most common programming mistake beginners make? Perhaps that they always confuse equality (==) with assignment (=), or & with &&? Or perhaps that they always use the wrong separators in for-loops (for (int i = 0, i < 5, i++))...?

To answer this question, data scientists recently looked at the mistakes from over 250,000 Java programming novices from all over the world. Using a massive amount of data (source code from 37 million compilations, to be exact), they revealed the most common errors that students make when they first learn Java, and how long it typically takes them to learn from their mistakes. The results are surprising.



The Blackbox data collection project

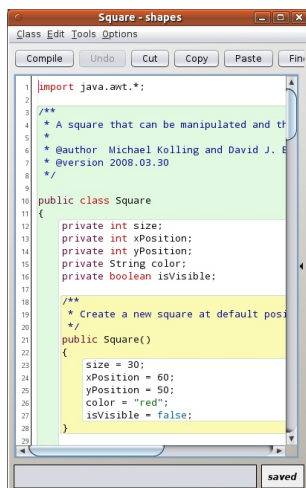
Learning a new programming language is challenging, because you have to learn how to express complex thoughts with a strict, formal grammar. Naturally many programming novices make mistakes. In order to shed light on which mistakes are most commonly made by programming novices, researchers from the University of Kent, UK recently looked at Java code collected from roughly 265,000 students around the world in the Blackbox data set.

The [Blackbox data collection project](#) is built around BlueJ, a free Java IDE specifically designed for beginners. BlueJ attempts to reduce the barriers to success for novices, for example by highlighting where code blocks begin and end (scope highlighting), which can help students visually scanning the code and spotting misplaced curly braces. Another feature is that objects can be inspected while the program is running, such that the contents of fields can be displayed in order to aid understanding and debugging. BlueJ comes with a textbook, teacher support, and is already part of the [Raspbian Linux](#) distribution.



Blackbox then acts as an opt-in extension to BlueJ that collects various anonymous data about how the software is being used. For example, it tracks which functions are executed and when, what errors are encountered and how frequently.

The data set is enormous. During the 2013-2014 academic school year alone, a total of 37,158,094 compilation events were collected, of which 19,476,087 were successful and 17,682,007 were unsuccessful.



While most of us might cringe at the thought of having to parse such a large amount of data, the challenge was just right for Amjad Altadmri and Neil Brown from the University of Kent, UK. They looked at every single one of the 46,448,212 source files involved in the 37 million compilation events mentioned above, and tracked changes in the file over time. At each compilation they checked the source file for one of 18 possible categorical mistakes, which they labeled A through R, and used these data to build a ranking of the most common errors. They also calculated the time it took students to fix the bug by looking forward in time to find the next compilation where the mistake was no longer present.

Beginners' 10 most common Java programming mistakes

Altadmri and Brown (2014) were surprised to find that what Java experts and teachers thought to be the most common mistakes that beginners made were merely *common misconceptions* based on anecdotal evidence. This might be a dangerous trend, they argue, because it is often experts and teachers that write Java books in which these misconceptions are reflected and nurtured.

However, if you ask the data, a clear ranking of the top 10 errors emerges (the letters A through R correspond to the 18 error classes included in the study):

- C. Unbalanced parentheses, curly braces, brackets, and quotation marks, or using these different symbols interchangeably, such as in: `while (a == 0]`.
- I. Invoking methods with wrong arguments or argument types, such as in: `list.get("abc")`.
- O. Control flow can reach end of non-void method without returning, such as in:

```
public int foo(int x)
{
    if (x < 0)
        return 0;
    x += 1;
}
```

- A. Confusing the assignment operator (=) with the comparison operator (==), such as in: `if (a = b)`.
- N. Ignoring or discarding the return value of a method with non-void return type, such as in: `myObject.toString();`.
- B. Use of `==` instead of `.equals` to compare strings.
- M. Trying to invoke a non-static method as if it was static, such as in: `MyClass.toString();`.
- R. Class claims to implement an interface, but does not implement all the required methods, such as in: `class Y implements ActionListener { }`.
- P. Invoking the types of parameters when invoking a method, such as in: `myObject.foo(int x, String s);`.
- E. Incorrect semicolon in `if` statements or `for` and `while` loops, such as in: `if (a==b); return 6;`.

There you have it. The most common error (C) is in fact to misplace or forget parentheses, curly braces, brackets, and quotation marks—the very thing BlueJ is trying to counteract with scope highlighting. This finding might be an indication that errors of type C would be even more prevalent in an IDE that does not provide any kind of debugging help.

Surprisingly, these mistakes did not make the Top Ten:

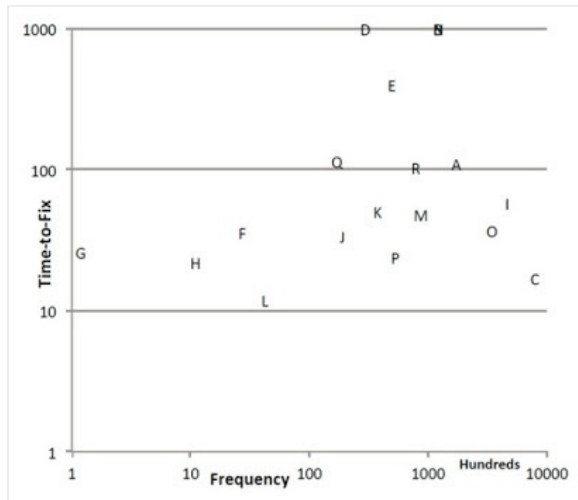
- D. Confusing "short-circuit" evaluators (`&&` and `||`) with conventional logical operators (`&` and `|`).
- J. Forgetting parentheses after a method call, such as in: `myObject.toString;`.
- Q. Using incompatible types between method return and type of variable that the value is assigned to, such as in: `int x = myObject.toString();`.
- F. Wrong separators in `for` loops (using commas instead of semi-colons), such as in: `for (int i=0, i < 6, i++)`.
- H. Using keywords as methods or variable names, such as in: `int new;`.

How long it takes students to learn from their mistakes

Furthermore, when Altadmri and Brown looked at how long it took students to


notice and fix the bug, it became clear that the most common mistake (error C) was also among the quickest to spot. Other bugs were notoriously hard to find, a sentiment most programmers might find easy to relate to. Among these mistakes were the confusing of `&&` or `||` with `&` or `|` (error D), the use of `==` instead of `.equals` to compare strings (error B), and ignoring or discarding the return value of a method with non-void return type (error N). These went unnoticed for more than 1,000 seconds (after which file changes were no longer tracked) or were never found at all.

The median time-to-fix (in seconds) for all error types is shown in the figure below (adapted from Altadmri & Brown (2015)). The two errors that overlap in the top right are errors B and N.



References:

- A. Altadmri & N.C.C. Brown, (2015). 37 Million Compilations: Investigating Novice Programming Mistakes in Large-Scale Student Data. Proceedings of the 46th ACM Technical Symposium on Computer Science Education (SIGCSE '15), 522-527, doi:10.1145/2676723.2677258.
- N.C.C. Brown & A. Altadmri, (2014). Investigating novice programming mistakes: educator beliefs vs. student data. Proceedings of the tenth annual conference on International computing education research (ICER '14), 43-50, doi:10.1145/2632320.2632343.

Posted by Michael Beyeler at 7:30 AM 
beginners , big data , data science , java , programming , top10