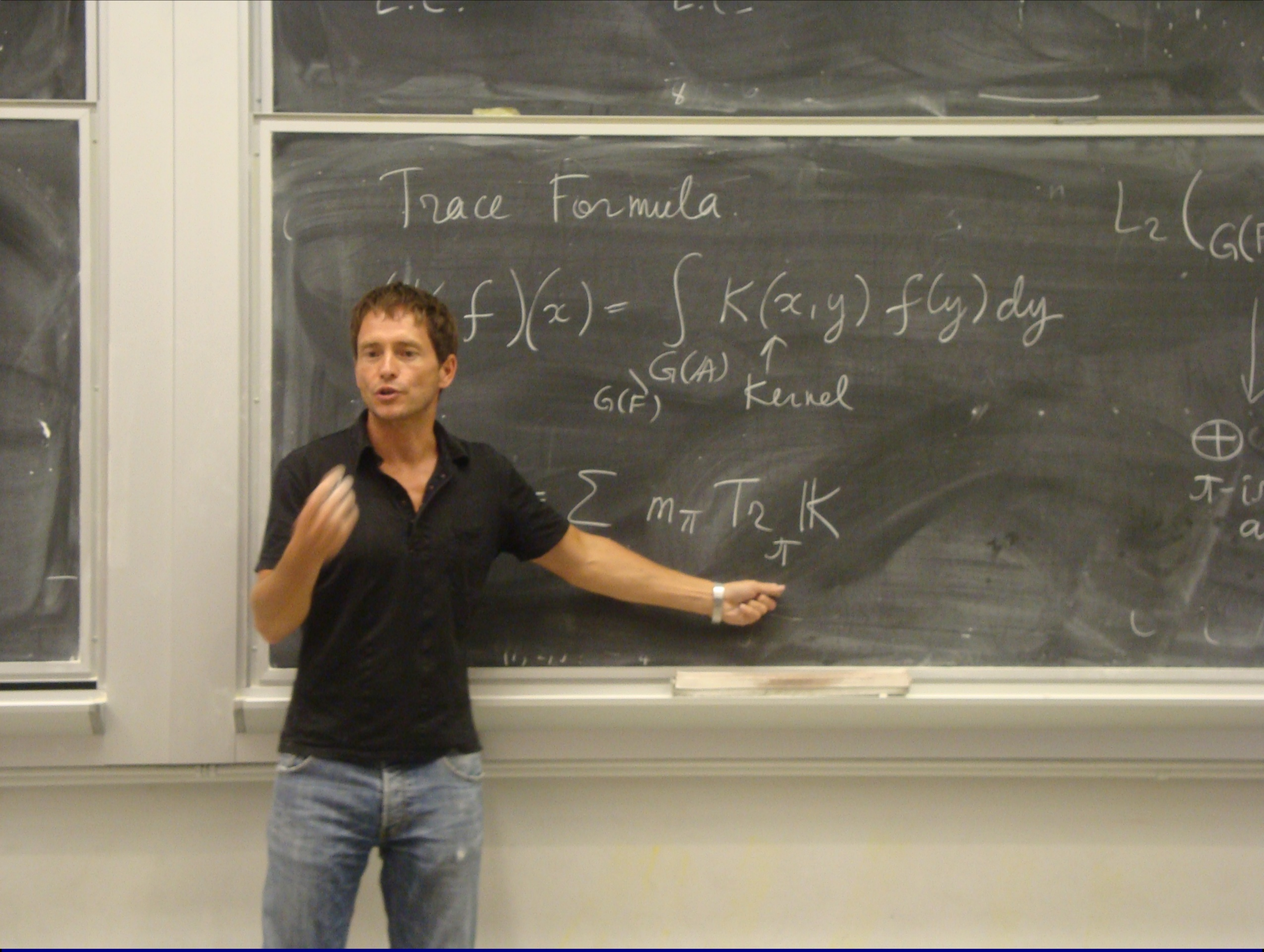

Java Bootcamp for Testers

Student Manual

Version 1.0

Document Information

Course Title:	Java Bootcamp for Testers
Author:	Rod Davison
e-mail:	rod@exgnosis.ca
Version:	1.0
Release Date:	2017-07-10
ProTech ID:	PT3944

"The primary objective of copyright is not 'to reward the labor of authors, but to promote the Progress of Science and useful Arts.' To this end, copyright assures authors the right to their original expression, but encourages others to build freely upon the ideas and information conveyed by a work. This result is neither unfair nor unfortunate. It is the means by which copyright advances the progress of science and art."
Justice Sandra Day O'Connor

Preface to the Student

This is a sort of experimental course. Originally I had been asked to recommend one of the Introduction to Java Programming courses for some testers. I had just taught an introduction to Ruby programming to a class full of testers and the one universal, ubiquitous, unanimous and unchanging comment they all had was "I don't to learn how to program. I will never need to program. What I need to know is enough Ruby/Java/C++/Python so that I can participate in code reviews and work with developers to perform QA on their code."

That was the genesis of this course, to teach Java to testers but not with the objective of turning them into programmers, but teaching them the language in a way that they could understand and critically examine Java code from the perspective of a tester or quality professional. This means that while you are learning basic Java programming, you are learning it from a rather unique perspective. The sorts of things that I liked to emphasize teaching to developers ("Ohh.. look what I can do with this really cool bit of code..") was really of no value to the students and had to be replaced with a different way of looking at code ("Now here are some typical sorts of mistakes that Java programmers make when using arrays.").

The analogy that I find tends to be somewhat appropriate here is that of authors and editors. Programmers are like authors – they are the creators of something. They have to understand the structure of a novel, how to do character development and use language to paint a picture for their readers. But authors tend to be too close to their words, the cannot objectively look at their work in terms of the quality of what they have created – that is the role of the editor. The editor has to also know about the structure of a novel, character development and the rest but not with the intent of writing a book but rather to do a critical analysis of an authors work.

Almost every author says that they can't see how any changes to their creation could make it better, and then the editor goes to work looking for those places where an objective eye can make the author's work even better. When authors read their book after a good editor has finished, they almost always agree that it is a much better book.

In that sense, we are teaching you in this course to be sort of like the editors – to be critical of the Java programs that the developers create so that you can help them write better code and build better applications.. We also want to provide a sense of where to focus your testing efforts based on where programmers tend to make mistakes writing Java code.

Anyway, consider this a work in progress so do feel free to work with the instructor to improvise to meet the needs you want this class to meet.

The Materials

What I did not want to do for this class is create yet another Java 101 set of courseware since there are already so many wonderful tutorials and reference materials available for free out there on that Internet thingy.

Textbooks.

I have a firm belief that you should have some sort of tangible takeaway from the class that will be useful to you later on to help you continue on in your Java learning experience. You won't learn Java in five days, just like you can't learn Italian or playing the flute in five days. You need to practice the concepts, ideas and techniques you will be taught in class to really learn Java – just like you have to practice speaking Italian or playing the flute in order to really learn it.

To implement that belief in a practical manner, I have provided three textbooks for the course:

1. Thinking in Java. This is Bruce Eccles 3rd edition which I like because it is the most object oriented presentation of Java. The downside to this book is that it is older so it doesn't cover the more recent versions of Java, however for the core programming basics I think it is great. The 4th edition is more recent but unfortunately it was released under a commercial license so we would not legitimately be able to offer the pdf for free, which an important consideration. One potential problem you may find with this book is that it is a bit more advanced and might be a bit too much of a plunge into the deep end of programming if you don't have a programming backgrouind.
2. Introduction to Programming Using Java by David Eck. A bit dated as well since it only deals with Java 7 but again for this class that is good enough. This book is a less challenging than the Eccles book but it does have some more advanced programming material that you might find interesting if you want to get into some more advanced coding.
3. Think Java: How to Think Like a Computer Scientist. This is most basic book and probably the one most useful to you in the course if you are a novice programmers since it focuses on the sort of basic programming we will be dealing with during class. A nice feature of this book is that the pdf has color highlights which make it easier to use in a presentation setting.

The last two books are distributed under a creative commons license so we are free to give the electronic versions away but both publishers charge for printed copies so I would prefer not to test legal waters or general ethical principles by providing by competitive printed copies.

The Student Manual.

The student manual contains supporting and ancillary material that I use that is not in any of the provided textbooks although there are overlaps at various points. For example, the portion on the Object Oriented paradigm is sort of a unique thing that I don't see in the books that we are using.

The student manual also tends to focus more on the sorts of places in Java code programmers where programmers tend to make errors since that is one of the objectives of the class.

The Testing Methods Book

This class assumes that you have a good working knowledge of testing and software quality assurance (which is why we call it “Java Bootcamp for Testers”). However, if you do need to do any brushing up on some aspects of testing, then hopefully this manual will be useful.

The TDD and ATDD Materials

Several of the topics that we want to cover in the latter part of the course are the modern techniques of test driven development and acceptance test driven development. These materials are adapted from courses I teach on the subject. There are places where there is some overlap with these and other materials.

Examples.

I’ve tried to provide the code used in any examples so that you can play with them yourself. The examples are not complex because I have tried to strip them down to make the essential points. I have noticed in some materials that in order to demonstrate a simple concept the author will use a rather complex example (“to see how inheritance works, we will build a multi-protocol Internet browser..”) which I personally love but I think has the effect of losing the concept in the complexity of all the other stuff in the example.

As well, I encourage you to follow along with instructor and do it themselves in the tried and true “monkey see, monkey do” philosophy of education. There are some people who just do not really “get it” until they actually get to do something themselves. It’s sort of like learning a new language, you may understand how the grammar works theoretically but it is when you try and speak the language that you discover that there is a difference between theoretical understanding and practical working knowledge.

As well, trying to follow along also gives you some continuous feedback into how well they are doing and the opportunity to correct little things they are doing wrong as you go rather than having to deal with all of it all at once during the lab.

Labs.

There are labs provided to be done at the end in each module. The labs are not the usual where you just write code to do something, although there are some that do that, but rather to start with some code and experiment with it in various ways.

All of the source code is available in the labs directory so that you don’t actually have to write any code from scratch. However there are a number of exercises in the three textbooks that are quite good for practicing more comprehensive programming problems.

> *“The mediocre teacher tells. The good teacher explains. The superior teacher demonstrates. The great teacher inspires.”*
>
> William Arthur Ward