JOURNAL OF OBJECT TECHNOLOGY

Online at http://www.jot.fm. Published by ETH Zurich, Chair of Software Engineering ©**JOT, 2010**

Vol. 9, No. 2, March-April 2010

Professional Developers Practice their Kata to Stay Sharp

Dave Thomas

1 CRAFTSMANSHIP AND KATA

One of the most interesting facets of the Software Craftsmanship movement is the return to *practice makes perfect* exemplified by KataCasts [1] and others. In contrast to Learn to Program in 10 Easy Lessons, there is an increased recognition that expertise is only achieved through substantive experience[1]. Programming Kata are small exercises that ensure that developers acquire and *maintain* the skills, idioms and techniques for effective programming.

2 YELLOW BELT – KNOW YOUR TOOL CHAIN KEYBOARD KATA

Flying without a mouse

While all modern operating systems, editors and IDEs provide menus and point and click operation, using them this way is an order of magnitude slower than using the keyboard shortcuts. This is why handing your favourite IDE to an Emacs user is like handing a classical guitar to a piano player. It doesn't matter if you can read music; it now becomes a serious cognitive and mechanical load to play even a few notes. Subconscious keystrokes become time consuming searches through pull downs, shortcut lists etc. Emacs et al also support user written macros that can compress tedious activities into a couple of keystrokes to do everything from reformatting your code to sending an email. Unfortunately, far too many developers don't use them. It should be part of every professional developer's craft to be an effective user of their tools. Exercises are needed to acquire and instinctively use the right shortcuts without the need to use the mouse or a shortcut reference card. The Kata can also ensure that the developers can use the IDE features for TDD by having one or more refactoring exercises.

[1] 10,000 hour rule, Outliers: The Story of Success

Dave Thomas, "Professional Developers Practice their Kata to Stay Sharp", in *Journal of Object Technology*, vol. 9 no. 2, March - April 2010, pp. 23 - 25 http://www.jot.fm/issues/issue_2010_03/column3/

You can't TDD without Shortcuts

Test First and *Refactoring* are arguably only possible by leveraging shortcuts. Using the mouse and menus makes it look like a developer is working in slow motion! Imagine if when you get an error in a new test you could not use a shortcut to define a new class or method. It is no surprise that those who don't use two hands and keyboard shortcuts find TDD takes a lot longer and often express great frustration or give up. Pairing between one handed and two handed developers is also very frustrating. It is surprising that most TDD books and courses hardly mention the importance of knowing the IDE shortcuts.

Developers should be required to do regular exercises to keep their yellow belt skills. The Kata should consist of a set of regular exercises that take the developer through the shortcuts needed to create, test and refactor code. They should also include exercises to interact with other tools such as Confluence and JIRA, which are key parts of the developer work flow. Developers should be able to complete the kata in roughly the same amount of time every time they do it.

3 GREEN BELT – KNOW YOUR LANGUAGE AND LIBRARY IDIOMS

Learning a programming language consists of learning the language syntax, including declarations, definitions and control structures. This is, however, the easy part. In order to be effective programming in a language, one needs to understand the styles, idioms and patterns in which elements of the language are composed. This requires the careful study of the data types and operations and, for an OO language, the libraries (APIs) that one intends to use. Developers program through use of *idioms*, putting whole functions together through combinations of common phrases. It has long been the tradition in the Lisp and APL communities to require newbies to first explore and develop familiarity and competence with the idioms of the language through small kata-like exercises [2, 3].

Through these exercises new programmers learn the common phrases and constructions used to compose programs. Indeed many of the OO design patterns are common idioms in functional programming. Once the idioms are known there is little need to search the language reference manual or library API for the meaning of an operation; instead, developers can reflexively select the appropriate idioms. They also seldom make newbie errors of passing the wrong argument types or getting an unexpected return result. This is in sharp contrast to the way many junior developers lean on their IDE and auto-completion[2] to provide APIs, casts and even correct syntax.

[2] All too often auto complete is used without thinking, resulting in code that is created and introducing accidental errors, even the use of the wrong API. Clearly this affordance is also very useful but you need to read the code that is written for you to make sure it is what you really intended.

4 PRACTICE MAKES PROFICIENT

Regular practice of various Kata improves the productivity of developers first by using finger exercises to enable the subconscious two-handed use of their IDE and second by building proficiency with the common idioms used in the language. Additional short exercises reinforce more advanced techniques and pairing, while individual challenges and reviews allow a developer to proceed to higher level belts.

REFERENCES

[1] Corey Haines, http://www.coreyhaines.com/, http://katas.softwarecraftsmanship.org/

[2] Alan J. Perlis, Spencer Rugaber; Programming with idioms in APL, ACM SIGAPL APL Quote Quad, June 1979, Pages: 232 - 235

[3] Daniel P. Friedman, Matthias Felleisen; The Little Schemer - 4th Edition

About the author

Dave Thomas is cofounder/chairman of Bedarra Research Labs (www.bedarra.com), www.Online-Learning.com and the Open Augment Consortium (www.openaugment.org) and a founding director of the Agile Alliance (www.agilealliance.com). He is an adjunct research professor at Carleton University, Canada and the University of Queensland, Australia. Dave is the founder and past CEO of Object Technology International (www.oti.com) creator of the Eclipse IDE Platform, IBM VisualAge for Smalltalk, for Java, and MicroEdition for embedded systems. Contact him at dave@bedarra.com or www.davethomas.net.