**เสียดาย… โปรแกรมเมอร์ไม่ได้อ่าน**

หากคุณคือโปรแกรมเมอร์ แล้วยังไม่ได้อ่านหนังสือเล่มนี้ ก็นับว่าน่าเสียดายอย่างยิ่ง เพราะหนังสือ “The Clean Coder” เปรียบเหมือน documentation สำหรับอาชีพโปรแกรมเมอร์ ที่จะช่วยให้คุณก้าวข้ามจาก “โปรแกรมเมอร์ที่ไม่มีความเป็นมืออาชีพ (Unprofessional Programmer)” ไปสู่ “โปรแกรมเมอร์มืออาชีพ (Professional Programmer)”

โปรแกรมเมอร์แบบไหนที่เรียกว่ามืออาชีพ?

หนังสือ The Clean Coder เริ่มต้นด้วยการอธิบายความหมายของคำว่า “มืออาชีพ” โดยลุงบ๊อบ (Robert C. Martin) แยกมืออาชีพกับความไม่เป็นมืออาชีพด้วย “ความรับผิดชอบ (Responsibility)”

ความรับผิดชอบ (Responsibility)

โปรแกรมเมอร์มืออาชีพ ควรที่จะรู้จักทั้ง “รับผิด” และ “รับชอบ” ในการทำงานของตน ซึ่งคำว่า “ความรับผิดชอบ (responsibility)” หมายถึง การรับผิดชอบต่อภาระหน้าที่ตามตำแหน่งงานที่ได้รับมาในการปฏิบัติงานเพื่อบรรลุเป้าหมาย ไม่เพียงแต่รับผิดชอบในเรื่องงานเท่านั้น ยังต้องรู้จักรับผิดชอบใน “อาชีพ” ของตนเองอีกด้วย

⇢ รอบรู้ในสายงานของตัวเอง (Know Your Field)

โปรแกรมเมอร์มืออาชีพยังต้องรอบรู้ในสายงานของตัวเองเป็นอย่างดี ลองถามตัวคุณเองดูว่า คุณรู้และสามารถอธิบายเรื่องเหล่านี้ได้หรือไม่

Design Pattern: คุณรู้จัก Model–View–Controller (MVC) pattern หรือเปล่า? Abstract factory เคยได้ยินหรือเปล่า แล้ว Template method, State, Dependency Injection, Delegate และ Singleton ฯลฯ

หากสงสัยว่าทำไมต้องเข้าใจเรื่อง Design Pattern แนะนำให้อ่านจากบทความ [“Design Pattern คือ ปัญหา หรือ วิธีการแก้ปัญหา”](http://www.somkiat.cc/design-pattern-is-a-problem-or-solution/) ที่

[UP1](https://medium.com/u/c5753ed98fb7?source=post_page-----c7976105acca-----------------------------------)

 (พี่ปุ๋ย) เขียนได้ครับ

Design principles: ซึ่งในปัจจุบันมีหลักการในการออกแบบซอฟต์แวร์มากมาย เช่น DRY (Don’t repeat yourself), KISS (Keep it simple, stupid) หรือแม้แต่ SOLID ที่ลุงบ๊อบคิดค้นขึ้น

Method: คุณควรเข้าใจถึงความหมายและวิธีการของ Extreme Programming (XP), Scrum, Lean, Kanban, Waterfall, Structured Analysis และ Structured Design

Disciplines (หลักปฎิบัติ): คุณควรฝึกปฎิบัติในเรื่องของ TDD (Test Driven Development), Object-Oriented design, Structured Programming, Continuous Integration และ Pair Programming

อื่นๆ เช่น คุณควรรู้วิธีการใช้ UML, DFDs, Structure Charts, Petri Nets, State Transition Diagrams and Tables, Flow charts, และ Decision tables

ไม่ทำอะไรที่ก่อเกิดความเสียหาย (Do no harm)

สิ่งที่ก่อให้เกิดความเสียหายในโปรแกรมที่เราเขียนขึ้นมานั่นก็คือ “บั๊ก (Bug)” ดังนั้นการเป็นโปรแกรมเมอร์มืออาชีพจะต้องไม่สร้างโปรแกรมที่มีบั๊ก!!!

😱😱😱😱….

ใช่แล้วครับ ฟังไม่ผิด… เราจะต้องสร้างโปรแกรมที่ไม่มีบั๊ก แต่เป็นเรื่องยากที่จะเขียนโปรแกรมที่ซับซ้อนไม่ให้มีบั๊กเลย

เมื่อเกิดบั๊กขึ้นมา โปรแกรมเมอร์มืออาชีพจะต้องรู้จัก “รับผิดชอบ” ต่อสิ่งที่ตัวเองสร้างขึ้นมา

โปรแกรมเมอร์ทั่วไป เมื่อเขียนโค้ดขึ้นมาแล้ว ก็จะผลักภาระให้ QA ตรวจสอบหาบั๊กและรอรายงานกลับ แต่โปรแกรมเมอร์มืออาชีพจะต้องตรวจสอบโค้ดของตัวเอง จนมั่นใจได้ว่า โค้ดนั้นทำงานได้อย่างถูกต้อง จึงส่งให้ QA ตรวจสอบอีกครั้ง

⇢ ต้องรู้ว่าโค้ดสามารถทำงานได้อย่างถูกต้อง (You Musk Know It Work)

แล้วเราจะรู้ได้อย่างไร ว่าโค้ดที่เราเขียนขึ้นมานั้น ทำงานได้อย่างถูกต้อง?

คำตอบก็คือ… ทดสอบมัน แล้วก็ทดสอบมันอีกครั้ง ทดสอบซ้าย ทดสอบขวา ทดสอบหน้า ทดสอบหลัง ทดสอบตั้งแต่ 7–11 เปิด ยัน 7–11 ปิด !!!

จนมั่นใจได้ว่า โค้ดของเราทำงานได้อย่างถูกต้อง

ซึ่งลุงบ๊อบ ให้ความสำคัญกับการทำ TDD ค่อนข้างมาก และการปรับปรุงโค้ดให้มี coverage สูงสุดเท่าที่จะทำได้ และใช้ Acceptance Test เพื่อตรวจสอบความถูกต้องกับทางฝั่ง Business

Acceptance test are not unit test. Unit tests are written by programmers for programmers. Acceptance tests are written by the business for the business (even when you, the developer, end up writing them)

ฝึกฝน, ฝึกฝน และก็ฝึกฝน (Practice)

สิ่งที่โปรแกรมเมอร์มืออาชีพ จะขาดไม่ได้เลยก็คือการฝึกฝนทักษะในสายงานอาชีพของตัวเองอยู่เสมอ เนื่องจากในสายงานนี้เทคโนโลยีและองค์ความรู้มีการพัฒนาปรับปรุง และถูกคิดค้นขึ้นมาใหม่ รวมถึงติดตามข่าวสารในวงการอยู่เสมอๆ

การจัดการ (Management)

และอีกองค์ประกอบสำคัญในการเป็นโปรแกรมเมอร์มืออาชีพ ก็คือ “ความสามารถในการจัดการ”

Time Management: เช่น การประเมินเวลางาน (Task estimation), การจัดลำดับความสำคัญของงาน, การบริหารเวลาเมื่อต้องมีการประชุม

Teams and Projects Management: การจัดการทีมและการทำงานร่วมกันเป็นทีม, การจัดการ project

Pressure Management: การรับมือกับสภาวะที่มีความกดดัน, commitment, การส่งมอบงานให้ทันตามกำหนดเวลา

หนังสือเล่มนี้ เหมาะสมอย่างยิ่งที่จะให้เป็นวิชา Programmer 100: Ethics for Programmer (จรรยาบรรณโปรแกรมเมอร์) หรือคนที่เป็นโปรแกรมเมอร์อยู่แล้วและอยากจะพัฒนาให้ตนเองเป็นโปรแกรมเมอร์ที่เรียกได้ว่า “มืออาชีพ”