# The Three Laws of TDD

By now everyone knows that TDD asks us to write unit tests first, before we write production code. But that rule is just the tip of the iceberg. Consider the following three laws:[1]

1. *Professionalism and Test-Driven Development*, Robert C. Martin, Object Mentor, IEEE Software, May/June 2007 (Vol. 24, No. 3) pp. 32–36

http://doi.ieeecomputersociety.org/10.1109/MS.2007.85

**First Law** You may not write production code until you have written a failing unit test.

**Second Law** You may not write more of a unit test than is sufficient to fail, and not compiling is failing.

**Third Law** You may not write more production code than is sufficient to pass the currently failing test.

These three laws lock you into a cycle that is perhaps thirty seconds long. The tests and the production code are written *together*, with the tests just a few seconds ahead of the production code.

If we work this way, we will write dozens of tests every day, hundreds of tests every month, and thousands of tests every year. If we work this way, those tests will cover virtually all of our production code. The sheer bulk of those tests, which can rival the size of the production code itself, can present a daunting management problem.