TDD can set you free

Mark Reed

Introduction

What is the coolest thing about TDD?

- The freedoms it affords you.

I want to focus on some of the fundamental benefits of TDD

These are potentially life changing (professionally at least)

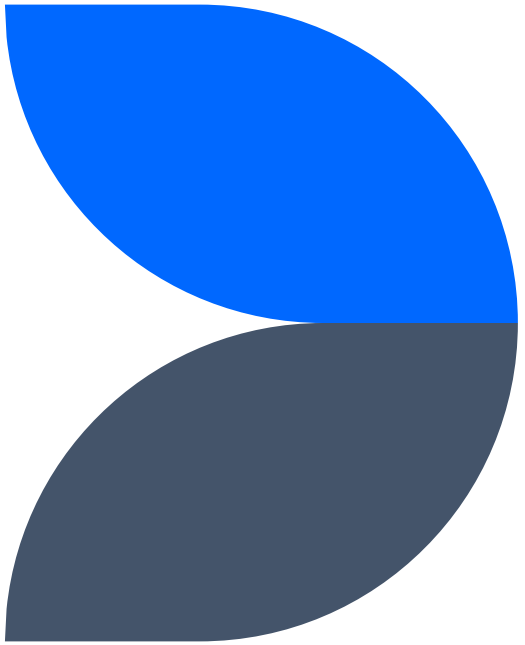
It all stems from the safety and security of knowing you can have confidence in your tests

I plan to talk through my own experiences, working for different companies as a developer in contrast with FDB and why TDD is the key to a more fulfilling experience as a developer

> If you're happy slamming some code together that more or less works and you're happy never looking at the result again, TDD is not for you.
>
> *Kent Beck*

Slamming Code Together

The “quick and dirty” school of software development



My First Job in Software Development

Small startup company, desperate to secure sales and grow

Tried to operate on low overheads and only employed a small team of graduate level developers

Directors brought no software development experience

The development process was largely undefined, we just took verbal or written requirements from the directors and implemented them to tight timescales

Quality was stated as a priority, but the methods and pressures did not allow it

> If you’re not doing test-driven development, you’re doing debug-later development.
>
> *James Grenning*

My Second Job in Software Development

Bespoke software house with ISO certified development processes and a far more professional approach

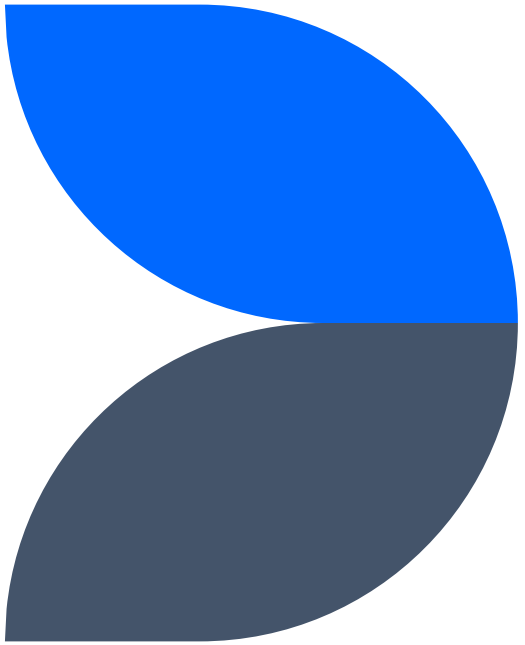
Testing was very much in the “QA department” domain

Developer testing was a quick manual once through of a feature before shipping to QA

Time taken and the predictability of time taken were the main concerns, all work completed was on a clock against tasks with limited time constraints, any overruns became very problematic

Test Driven Development

Confidence in your solution is “baked in”



The Safety Net

One of the primary purposes of writing tests iteratively for each behavior, before implementing the code is to inspire confidence in your solution

This confidence evidence based and is powerful

You don’t think your solution is fit for purpose; you *know* it is fit for all the purposes for which you have written tests

> Worried that TDD will slow down your programmers? Don't. They probably need slowing down.
>
> *J. B. Rainsberger*

Experimentation

Confidence in your tests and tests based on behavior rather than implementation = freedom to improve the design safely

Now you can be creative

You are free to carry out extensive and wide-ranging refactoring, you can make your code something you can really take pride in

You can speculate and try things out or see how they look and revert when you are not convinced

This process never really ends

> Any fool can write code that a computer can understand. Good programmers write code that humans can understand.
>
> *Martin Fowler*

Code as communication

Where does documentation go to die?

- Confluence

Behavioral tests provide living documentation

The test directs you specifically to the relevant code in the implementation

Summary

TDD breeds confidence in our automated tests covering the necessary behaviors of our software

Having confidence in your tests and your tests focused on behavior means you can refactor or even rewrite in the knowledge that once you are green you have returned to your safe place

Understanding the behavior modelled by the tests gives context to what the code is trying to achieve and enhances our ability to interpret it on a human level as well as reducing the need for other documentation

All this results in a less stressful and more fulfilling experience as a software developer, certainly when compared to slamming it together.

Thankyou

Mark Reed

mark.reed@fdbhealth.com