Tom Preston-Werner home

Fork me on GitHub

Readme Driven Development

23 August 2010 - San Francisco

I hear a lot of talk these days about TDD and BDD and Extreme Programming and SCRUM and stand up meetings and all kinds of methodologies and techniques for developing better software, but it's all irrelevant unless the software we're building meets the needs of those that are using it. Let me put that another way. A perfect implementation of the wrong specification is worthless. By the same principle a beautifully crafted library with no documentation is also damn near worthless. If your software solves the wrong problem or nobody can figure out how to use it, there's something very bad going on.

Fine. So how do we solve this problem? It's easier than you think, and it's important enough to warrant its very own paragraph.

Write your Readme first.

First. As in, before you write any code or tests or behaviors or stories or ANYTHING. I know, I know, we're programmers, dammit, not tech writers! But that's where you're wrong. Writing a Readme is absolutely essential to writing good software. Until you've written about your software, you have no idea what you'll be coding. Between The Great Backlash Against Waterfall Design and The Supreme Acceptance of Agile Development, something was lost. Don't get me wrong, waterfall design takes things way too far. Huge systems specified in minute detail end up being the WRONG systems specified in minute detail. We were right to strike it down. But what took its place is too far in the other direction. Now we have projects with short, badly written, or entirely missing documentation. Some projects don't even have a Readme!

This is not acceptable. There must be some middle ground between reams of technical specifications and no specifications at all. And in fact there is. That middle ground is the humble Readme.

It's important to distinguish Readme Driven Development from Documentation Driven Development. RDD could be considered a subset or limited version of DDD. By restricting your design documentation to a single file that is intended to be read as an introduction to your software, RDD keeps you safe from DDD-turned-waterfall syndrome by punishing you for lengthy or overprecise specification. At the same time, it rewards you for keeping libraries small and modularized. These simple reinforcements go a long way towards driving your project in the right direction without a lot of process to ensure you do the right thing.

By writing your Readme first you give yourself some pretty significant advantages:

- Most importantly, you're giving yourself a chance to think through the project without the overhead of having to change code every time you change your mind about how something should be organized or what should be included in the Public API. Remember that feeling when you first started writing automated code tests and realized that you caught all kinds of errors that would have otherwise snuck into your codebase? That's the exact same feeling you'll have if you write the Readme for your project before you write the actual code.

- As a byproduct of writing a Readme in order to know what you need to implement, you'll have a very nice piece of documentation sitting in front of you. You'll also find that it's much easier to write this document at the beginning of the project when your excitement and motivation are at their highest. Retroactively writing a Readme is an absolute drag [illegible]'re sure to miss all kinds of important details when you do so.

- If you're working with a team of developers you get even more mileage out of your Readme. If everyone else on the team has access to this information before you've completed the project, then they can confidently start work on other projects that will interface with your code. Without any sort of defined interface, you have to code in serial or face reimplementing large portions of code.

- It's a lot simpler to have a discussion based on something written down. It's easy to talk endlessly and in circles about a problem if nothing is ever put to text. The simple act of writing down a proposed solution means everyone has a concrete idea that can be argued about and iterated upon.

Consider the process of writing the Readme for your project as the true act of creation. This is where all your brilliant ideas should be expressed. This document should stand on its own as a testament to your creativity and expressiveness. The Readme should be the single most important document in your codebase; writing it first is the proper thing to do.

--

Discuss this post on Hacker News

Related Posts

- 21 Apr 2014 » Farewell GitHub, Hello Immersive Computing
- 22 Nov 2011 » Open Source (Almost) Everything
- 03 May 2011 » Rejected Bio from The Setup

Tom Preston-Werner
Cofounder of GitHub
tom@mojombo.com

github.com/mojombo
twitter.com/mojombo
flickr.com/photos/mojombo