TDDBUDDY.COM

Stages of Naming

Importance of Naming

“There are 2 hard problems in computer science: cache invalidation, naming things, and off-by-1 errors.”
- **Phil Karlton**

Naming is critical to software development because it directly affects the readability and therefore how easy code is to understand; code that is easy to read and understand is easier to change.

Initial Development

Change in the context of initial development might seem irrelevant because it is tempting to assume that initial development of a systems only involves writing new code. This however is very rarely the case, code is revisited, modified, updated and/or refactored as initial development proceeds and increases with frequency the further you get into the process. Code that is harder read causes the rate of change to decrease over the lifetime of the application.

Maintenance Development

Now consider what change means in the context of maintenance. Most systems have longer than anticipated life spans and need ongoing maintenance, in the form of bug fixes and adjustments to meet new business requirements, during the course of their life spans. Maintenance requires change, so the easier it is to change code the quicker and cheaper it is to maintain code.

3 Stages of Naming

We advocate for 3 stages of naming: meaningless, specific and meaningful, while others advocate for up to 7 stages[1]. We feel 3 is a usable amount of stages for production, while seven feels academic.

Things can start at any point and move in any direction. For example you can have a meaningless name that is changed to be specific, or you can have a meaningful name that is changed to be meaningless.

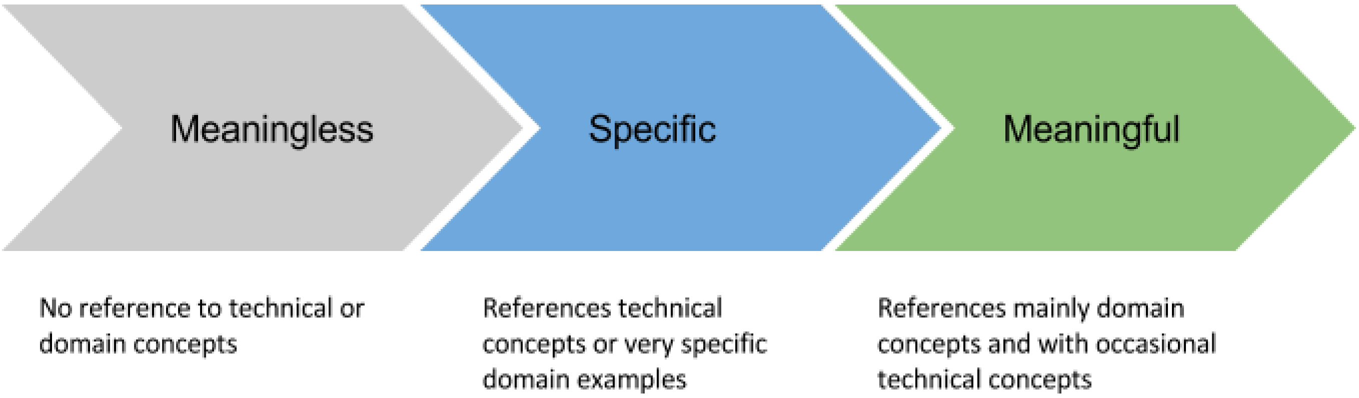
Meaningless
Specific
Meaningful
No reference to technical or domain concepts
References technical concepts or very specific domain examples
References mainly domain concepts and with occasional technical concepts

Figure 1 : 3 Stages of Naming

It is important to be aware of where the names of the things you are working on are so that you can strive to push them towards meaningful as you work. The key here is continuously improving your names as your work, choosing a name is hard and often you only have all the information you need to name something meaningfully after you’ve gained new information through the act of writing the code.

[1] http://arlobelshee.com/good-naming-is-a-process-not-a-single-step/

Don’t spend too much energy on coming up with a meaningful name right at the beginning, this is wasted energy, pick a name based on the information you have, identify what stage it is in, and revisit the name every time you read or modify that section of code, if you have a better name for it then update it, otherwise leave it. If you have to leave that section of code and don’t have meaningful names yet, leave *todo* comments to improve the names later.

Variable and Method Examples

Below are examples of variables and method names as they move from meaningless to specific and finally to meaningful.

Naming Continuum

Meaningless	**Specific**	**Meaningful**
var x;	var respository;	var itemCatalogue;
var cake;	var webservice;	var fetchBtcPrices;
private bool IsStuff();	private bool AreItemsNullOrEmpty();	private bool IsCatalogueEmpty();
public double Bar();	public double AddTwoNumbers();	public double Sum();

Test Name Examples

Starting with a meaningless name of *Method_GivenStuff_ShouldReturnResult* then moving to a specific name of *GetResult_GivenThree_ShouldReturnFizz* and finally ending up at the meaningful name of: *GetResult_GivenMultipleOfThree_ShouldReturnFizz*