



I will make you
a better C# developer
2018 edition

Kathleen Dollard

Microsoft

kdollard@microsoft.com

Twitter: @KathleenDollard

<https://github.com/KathleenDollard/Slides>

Follow



Miguel

Follow

Followed others



Chris "V"

Follow



Find people
Import your

Connect other address

Trends for you

#Radja
1,202 Tweets

#RoyalWedding
616K Tweets

#SPC18
6,644 Tweets

SharePoint
6,745 Tweets

Mark Zuckerberg
8,188 Tweets

#EGP28

#Swashfields

Rome
31.2K Tweets



Persian Rose

@PersianRose1

Follow



Branch manager and assistant branch manager



Tweet



ged under tes...



ged under tes...



can try a try a



Picard Tips @PicardTips · Jan 11



Picard management tip: Question authority, and permit others to question yours without fear of punishment.



8



611



1.6K



Demo



Picard Tips @PicardTips · Feb 7



Picard management tip: If you succeed too much and become cocky, recognize it, admit it, and stop it.



8



277



1.1K



You're an **intermediate to advanced** C# programmer excited about attending a conference filled with cutting edge talks from amazing speakers. It's important to stay up to date, but you keep thinking of your existing code assets and how much time you spend on that code. Sometimes you feel that by the time you finish a new application, the code is already out of date.

This workshop builds your capacity as a **keeper of code** - caring for and evolving existing code assets. First, you'll take a step back to learn how to assess your assets and whether it will be easy or hard to evolve them. You'll consider long-range planning to maintain the value of your code assets – especially where to apply gradual change to reduce disruption and minimize risk. A step-wise process offers benefit at every point along the way.

You need to keep code safe, and that means tests to ensure your code continues to work as expected. You'll gain insight and learn techniques for creating tests for existing code assets and improving the effectiveness and the ease of maintaining your tests.

You need to keep code predictable. You'll learn the most common pitfalls in .NET types that can affect accuracy and performance.

You need to keep code understandable. That means shrinking it, organizing it and clarifying intent. You'll learn techniques to minimize boring redundant code so that special case, interesting and error prone code stands out. Those techniques will include generic hierarchies and higher-order functions. You'll also get better at refactoring and isolating technology dependent code.

You need to prepare code for the future. The .NET Standard is the declaration of a set of APIs that current and future frameworks will implement, and you'll learn to evaluate your code's compatibility with .NET Standard. This compatibility also lets you move parts of your app to other platforms and operating systems. Your code will be ready for Xamarin, UWP, and .NET Core and to run on Windows, Linux and MacOS.

You'll see an application evolve from a mess to a well-structured application ready to support additional platforms. Moving your application isn't just about code, so you'll also get **tips for inspiring your coworkers and building management buy in**. During this workshop, you'll learn a lot about code and you'll also understand how to apply these ideas to the ongoing evolution of your most important code assets. You'll leave ready to take the right sized steps for each set of code assets.

What is legacy code?



| What is
| legacy code?

“Code not under
test”

Michael Feathers, *Working
Effectively with Legacy Code*

| What is
| legacy code?

“Code that has
been checked in”

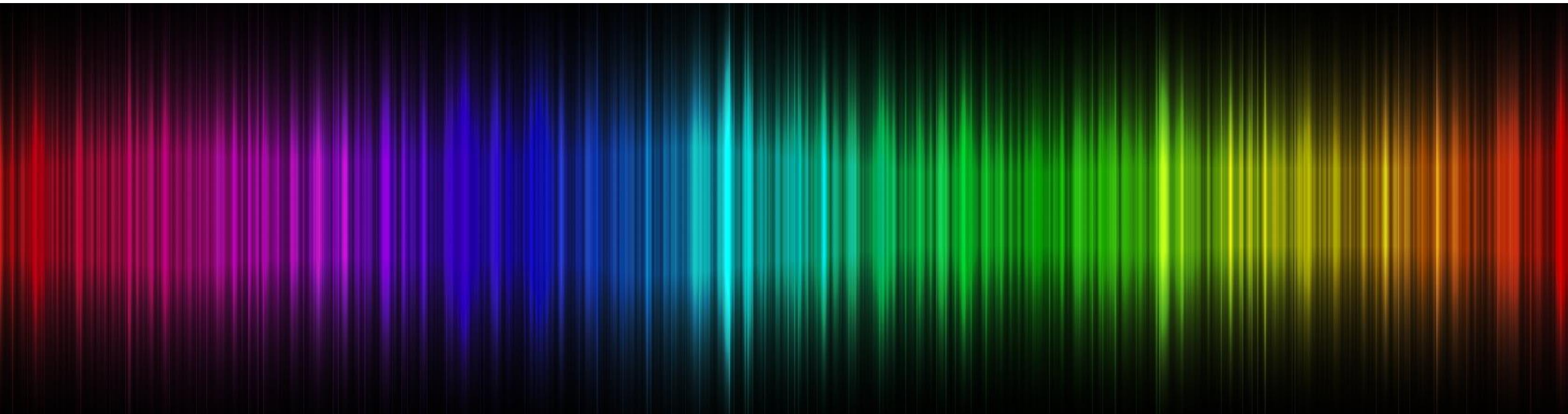
Kathleen Dollard

| What is
| legacy code?

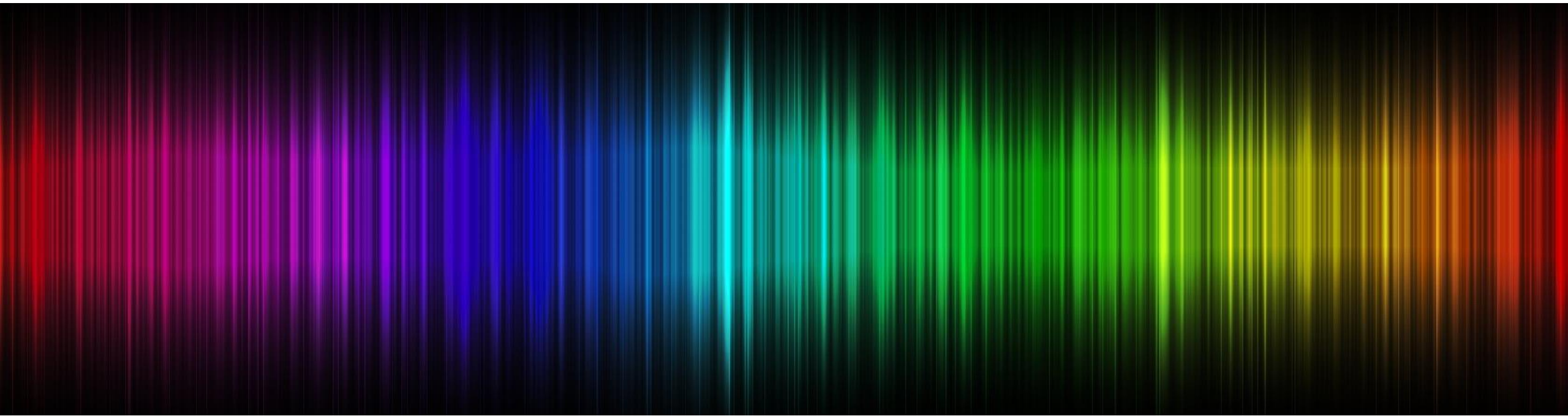
“Code that you
personally do not
understand”

Kathleen Dollard


Spectrum of how much we know about



Spectrum of how much we know about
the problems the code solves





A person in a red shirt stands on the edge of a layered rock cliff, looking out over a vast canyon. The sun is low on the horizon, casting a warm glow over the scene. A river winds through the bottom of the canyon. The sky is a clear gradient from blue to orange.

How do we understand
something so complex?

Tests



Tests matter because...

We clearly understand those scenarios.

We can ensure, even guarantee,
that changes we make to the code do not break
the exact scenario described by the tests



Is it easier to fix or rewrite code?

Netscape 6.0 is finally going into its first public beta. There never was a version 5.0. The last major release, version 4.0, was released almost three years ago. Three years is an *awfully* long time in the Internet world. During this time, Netscape sat by, helplessly, as their market share plummeted.

It's a bit smarmy of me to criticize them for waiting so long between releases. They didn't do it *on purpose*, now, did they?

<https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/>

Netscape 6.0 is finally going into its first public beta. There never was a version 5.0. The last major release, version 4.0, was released almost three years ago. Three years is an *awfully* long time in the Internet world. During this time, Netscape sat by, helplessly, as their market share plummeted.

It's a bit smarmy of me to criticize them for waiting so long between releases. They didn't do it *on purpose*, now, did they?

Well, yes. They did.

They did it by making the **single worst strategic mistake** that any software company can make:

They decided to rewrite the code from scratch.

<https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/>

We're programmers.

Programmers are, in their hearts, architects, and the first thing they want to do when they get to a site is to bulldoze the place flat and build something grand.

We're not excited by incremental renovation: tinkering, improving, planting flower beds.

Maybe
Sometimes

There's a subtle reason that programmers always want to throw away the code and start over.

The reason is that they think the old code is a mess.

And here is the interesting observation: *they are probably wrong.*

The reason that they think the old code is a mess is because of a cardinal, fundamental law of programming:

It's harder to read code than to write it.

This is why code reuse is so hard.

<https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/>

Sure,
But mostly programmers
think they can do a
BETTER job.

Why?

Why do programmers
think they can rewrite?

The idea that new code is better than old is patently absurd.

Old code has been *used*.

It has been *tested*.

Lots of bugs have been found, and they've been *fixed*.

There's nothing wrong with it.

It doesn't acquire bugs just by sitting around on your hard drive. ...

Back to that two page function.

Yes, I know, it's just a simple function to display a window,
but it has grown little hairs and stuff on it and nobody knows why.

Well, I'll tell you why: those are bug fixes.

One of them fixes that bug that Nancy had when she tried

Another one fixes that bug that occurs in low memory conditions.

Another one fixes a floppy disk and the user yanks out the disk in the middle.

That LoadLibrary call is ugly but it makes the code work on old versions of Windows 95.

Do we believe



Our project is different
It's so awful that we can't
possibly rewrite it
No one understands it
We are smarter than we
were last year
It will be painful to rewrite



Picard Tips @PicardTips · Feb 9



Picard schooling tip: Don't be too tentative. Have experiences. Be dastardly. Get into trouble.



12



210



758



NOT
Rewriting.
—tweaking tech.

ASP.NET MVC Music Store Tutorial

Version 3.0b

Jon Galloway - Microsoft

4/28/2011

Moving MusicStore Forward

- App last updated in 2011
- Find *smallest* step you think reasonable
- ***Open and run in Visual Studio 2017 Update 8 Preview***

Moving MusicStore Forward

- Requested migration, assuring it would leave a log file
- ***Sure***
- Dire warnings about WebPages no longer being cool
- ***What could possibly go wrong?***

Moving MusicStore Forward

- Tried to build, received errors on
 - System.Web.Helpers
 - System.Web.Mvc
 - System.Web.WebPages
- Warning icons on several things in the Dependency node of Solution Explorer
- ***Searched MVC 3 migration***
- ***Found docs page and went through steps***
 - ***(this was to MVC 4)***
- ***To update NuGet, I chose MVC 5***
- ***What could possibly go wrong?***

Moving MusicStore Forward

- Could not install Razor 3.2.6 because the app targeted .NET 4.0
- ***Targeted .NET 4.6.1***
 - ***Might as well jump forward on that***
- ***Reinstalled MVC 5 via NuGet***
- ***Closed and reopened VS because of a warning message***

Moving MusicStore Forward

- Four compiler errors about an ambiguity in Compare and ErrorMessage
- ***Removed using statement for System.Web.Mvc from the model class (what was it doing there anyway?)***
- ***YEAH!!! Compile works***

Moving MusicStore Forward

- Error on run
- WebPages 2.0 was specified and 3.0 found
- OK, I knew that MVC 5 decision would cause troubles
- ***Changed WebPages 2.0 to 3.0***

Moving MusicStore Forward

- Code ran to point of data access
- “...Provider not installed...”
- ***Panic sets in***
- ***Read Readme.txt***
 - ***(Finally)***
- ***Glance through PDF***
 - ***Great, it's a CE database***
- ***Tried to create a new connection string***
- ***Tried to create a new connection string***
- ***Tried to create a new connection string***
- ***Tried to create a new connection string***
- ***Tried to create a new connection string***

Moving MusicStore Forward

- There is a broken connection in Server Explorer
- Could not create a new SQL Client database
 - Is this a Community problem, can't be. Dunno.
- ***Finally modified the broken connection to work***
- ***Copied the connection string***

Moving MusicStore Forward

- “Database does not contain model metadata...Ensure that IncludeMdatataCon...”
- ***Searched for message***
- ***Apparently this ancient EF didn't do modern migrations***
- ***One suggestion is to delete and recreate database***
- ***Panic sets in***
- ***Decide to commit***

Moving MusicStore Forward

- db_lock fail on a deep name
 - .git/.../.../.../.../.../...

- ***Search for error***
- ***Add .vs to .gitignore***
 - *Just .vs* didn't work*
 - *Panic*
 - *.vs and .vs* worked*

----- *back to database*

- ***Instead of deleting, just go to package manager and type***

> update-database

Moving MusicStore Forward

- “...cmdlet not found”
 - ***Check NuGet and find 5 packages that need updates***
 - ***Update***
----- *back to database*
 - ***Try again***
- > update-database**

Moving MusicStore Forward

- “No migration configuration type found in the assembly...In Visual Studio you can use Enable-Migrations from Package Manager console...”
- ***OK, yeah, I can do that***
 - > **enable-migrations**
 - > **update-database**

Moving MusicStore Forward

- “...Unable to update...pending changes....or enable auto migrations. Set `DbMigrationConfiguration.AutomaticMigrationsEnabled` to true.

- ***Something about migrations?***

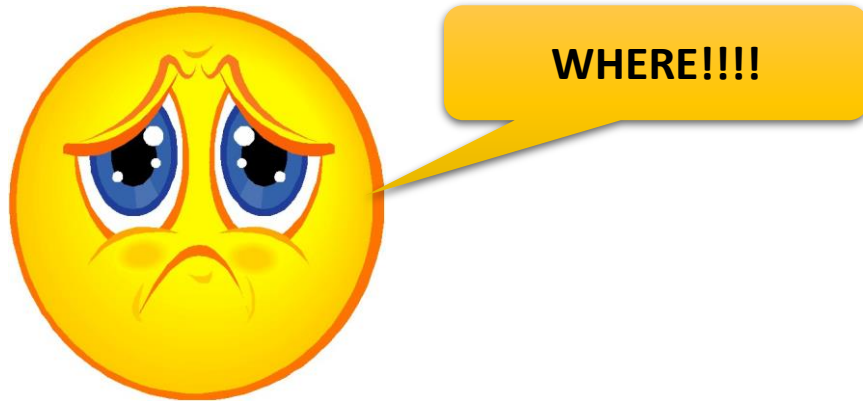
> `add-migration`

Name :



Moving MusicStore Forward

- “...Unable to update...pending changes....or enable auto migrations. Set `DbMigrationConfiguration.AutomaticMigrationsEnabled` to true.



- ***Something about migrations?***

> add-migration

Name: (this could have been a very bad word)

Moving MusicStore Forward

- “...Unable to update...pending changes....or enable auto migrations. Set `DbMigrationConfiguration.AutomaticMigrationsEnabled` to true.

- ***Something about migrations?***
 - > `add-migration`
Name: MvcMusicStoreini
- ***Try to run again...***





ASP.NET MVC MUSIC STORE

[Home](#) | [Store](#) | [Cart \(0\)](#) | [Admin](#)



Fresh off the grill

built with **ASP.NET MVC 3**

Yes, it's going to be painful



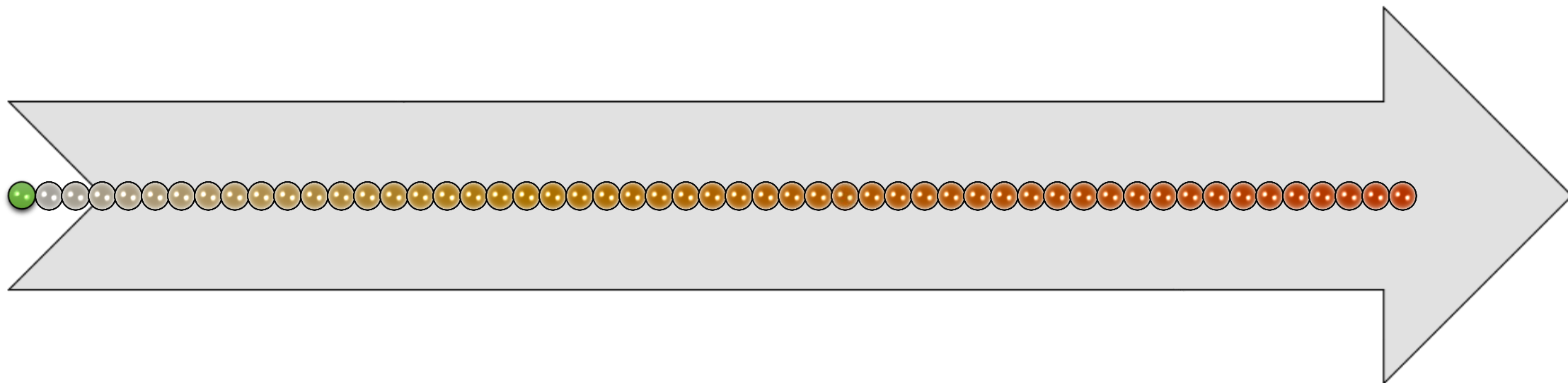
- <https://twitter.com/RichRogersIoT/status/998288346286768128>
- Size you can handle
- https://twitter.com/IJohnson_TNF/status/998566264288305154
 - **Source control**

You are the Keeper of Your Code

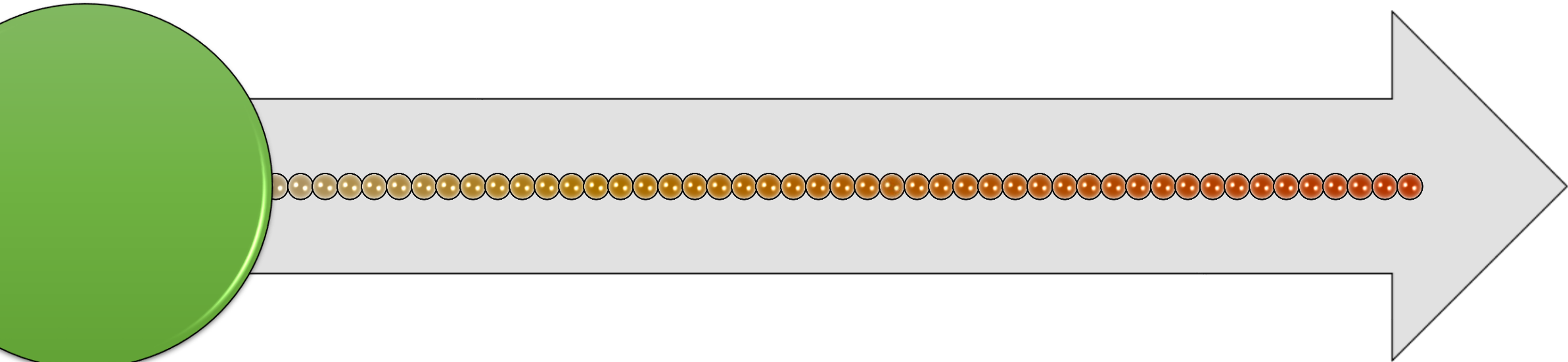
Make it
incrementally
better



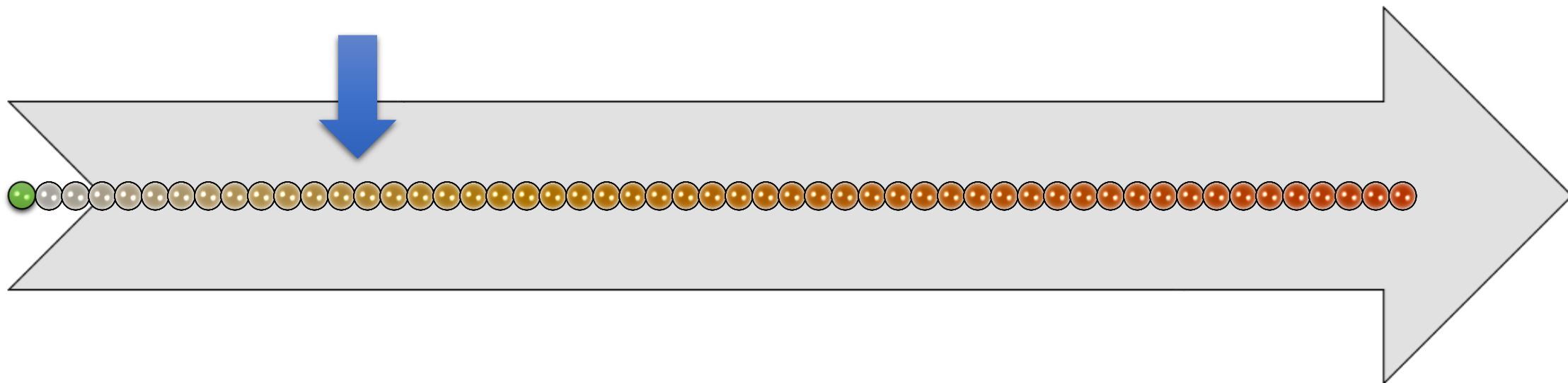
Your app



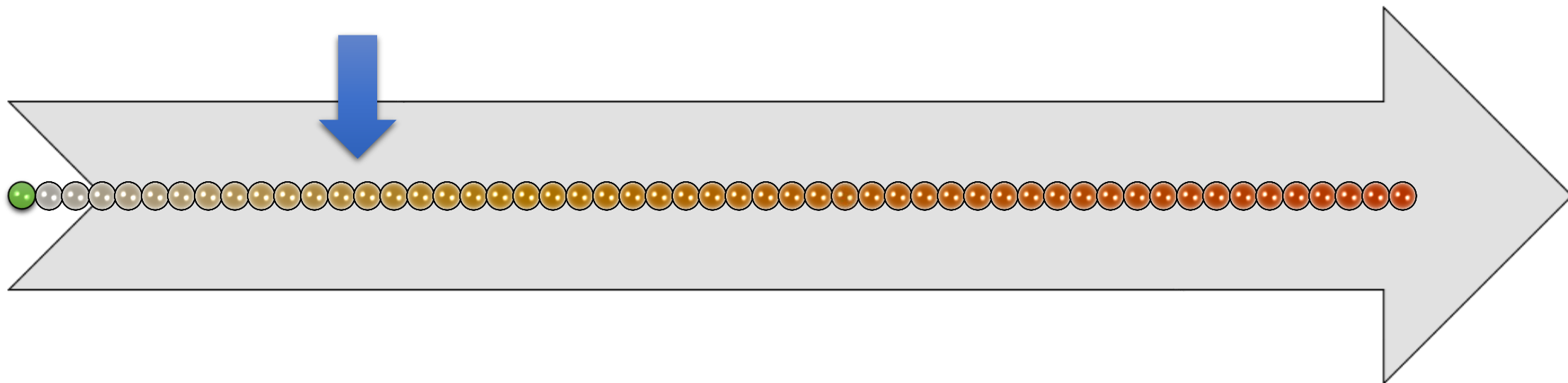
Your app

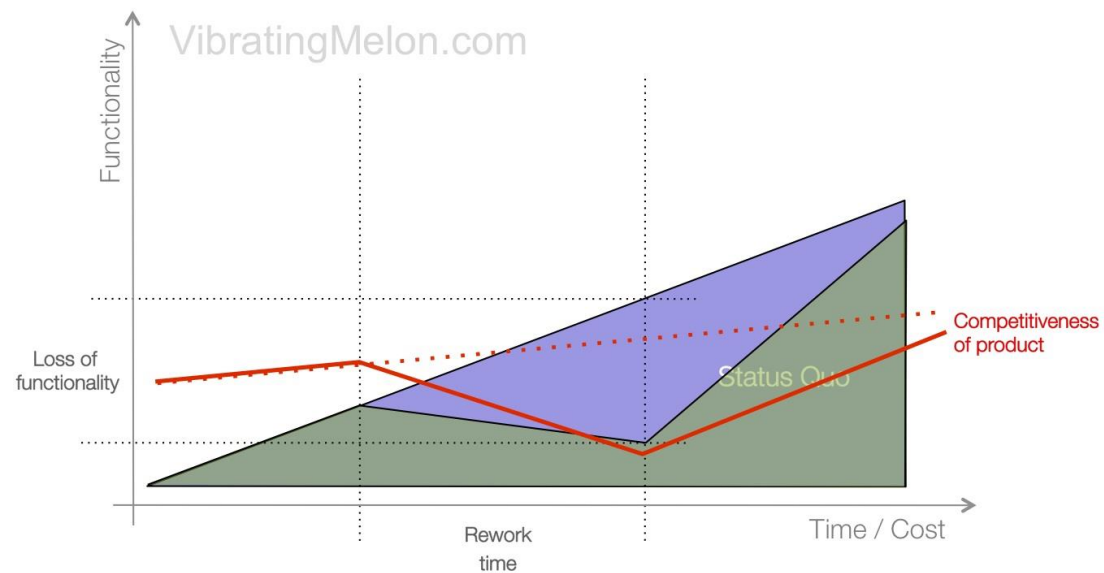
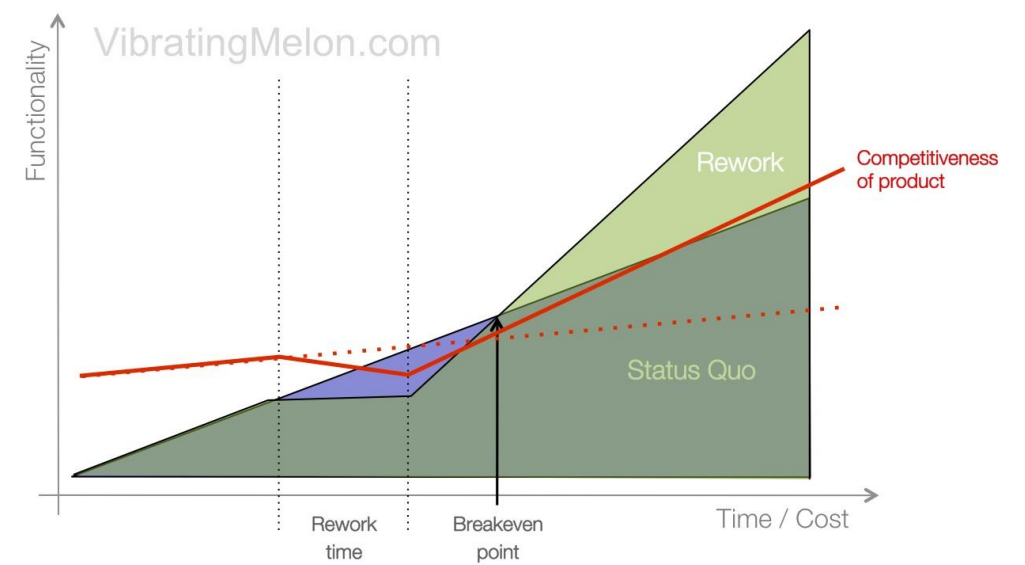
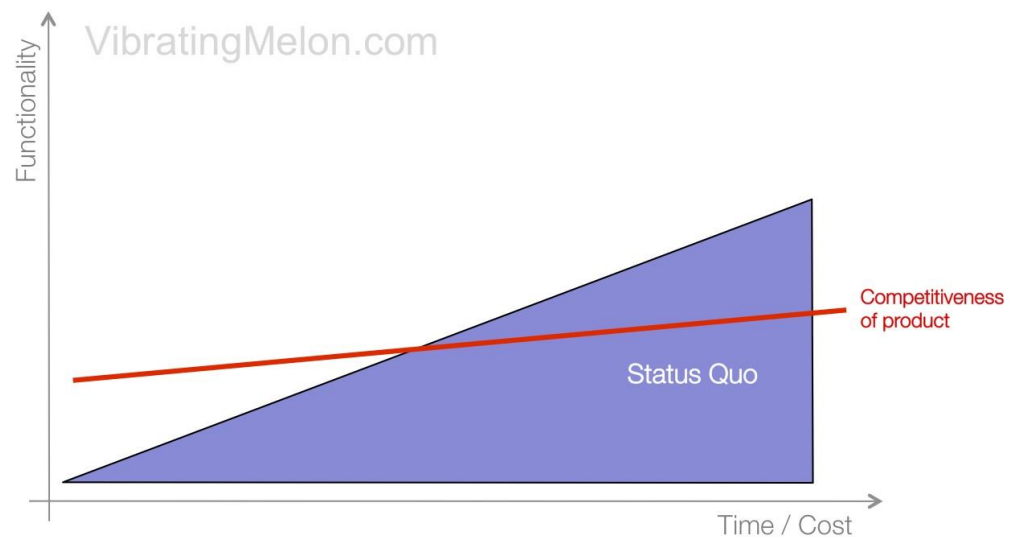


Your app



Your app

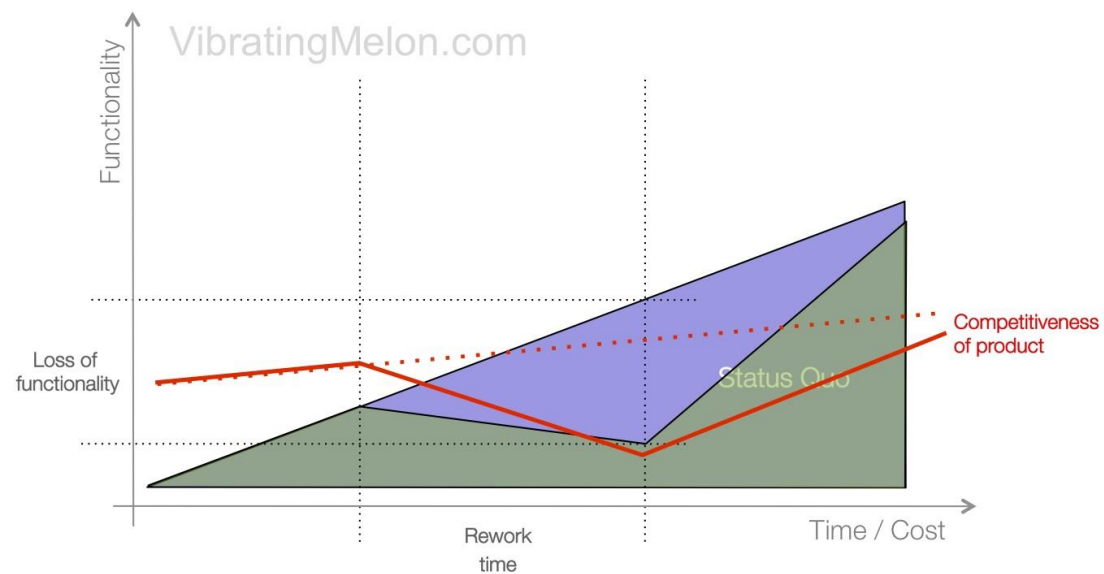
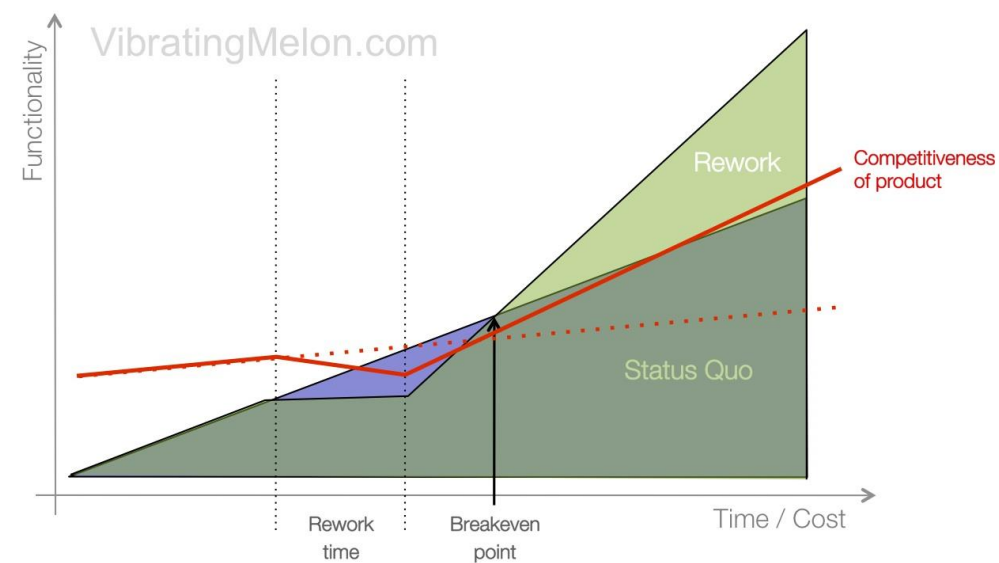
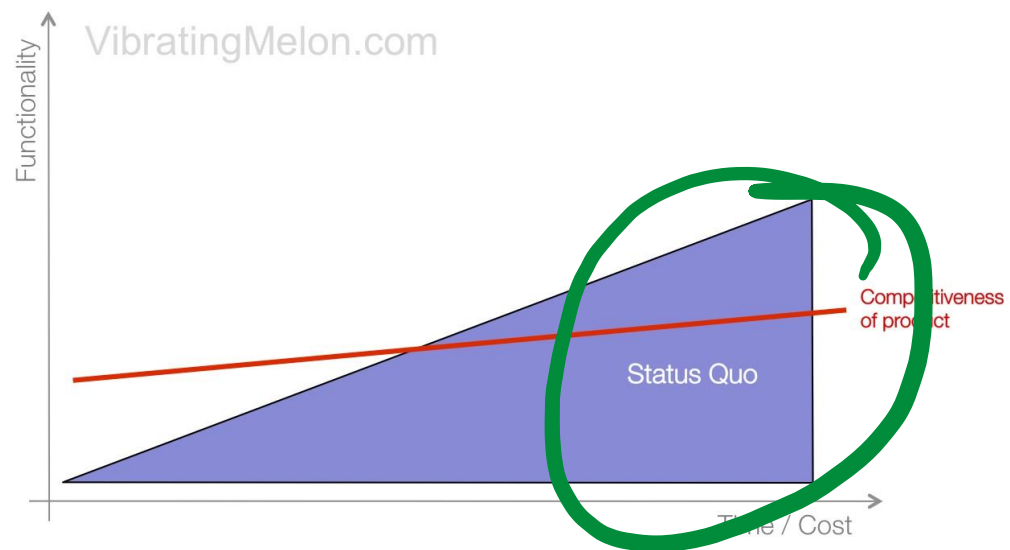




<https://vibratingmelon.com/2011/06/10/why-you-should-almost-never-rewrite-code-a-graphical-guide/>

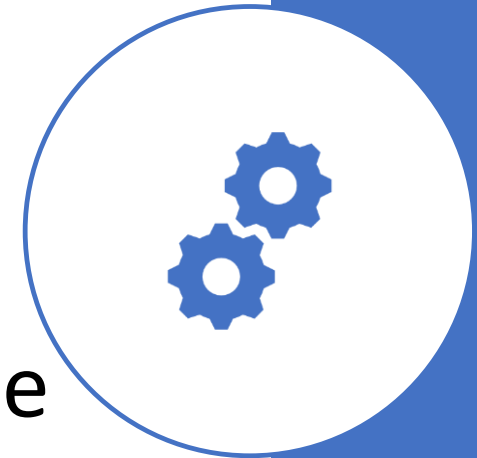
- Your project needs ongoing maintenance
 - Nuisance expense
- If you rewrite your project
 - It will still need ongoing maintenance
 - It will still be a nuisance expense

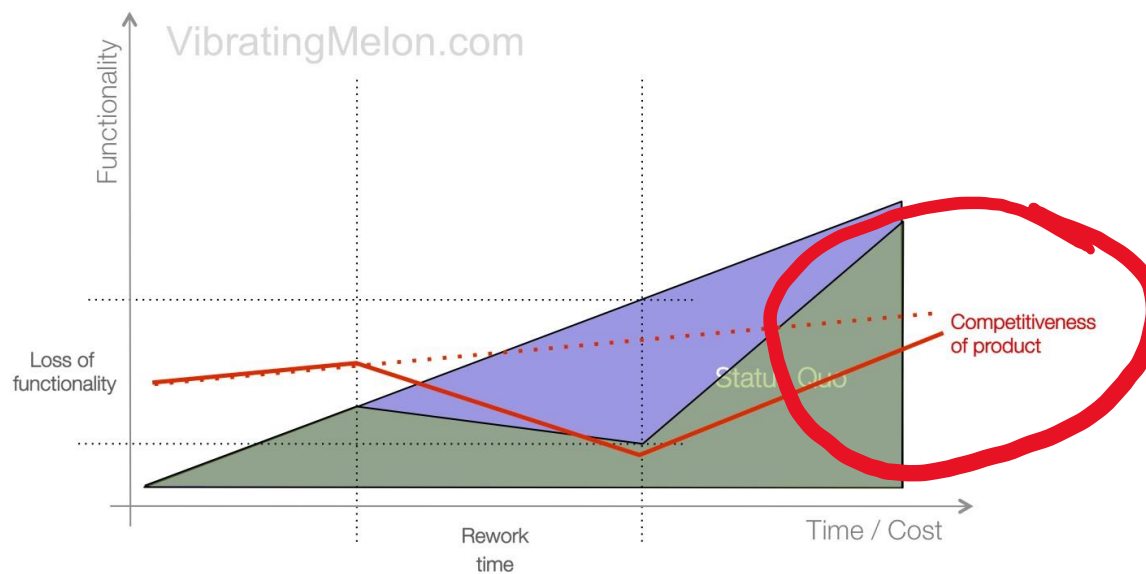
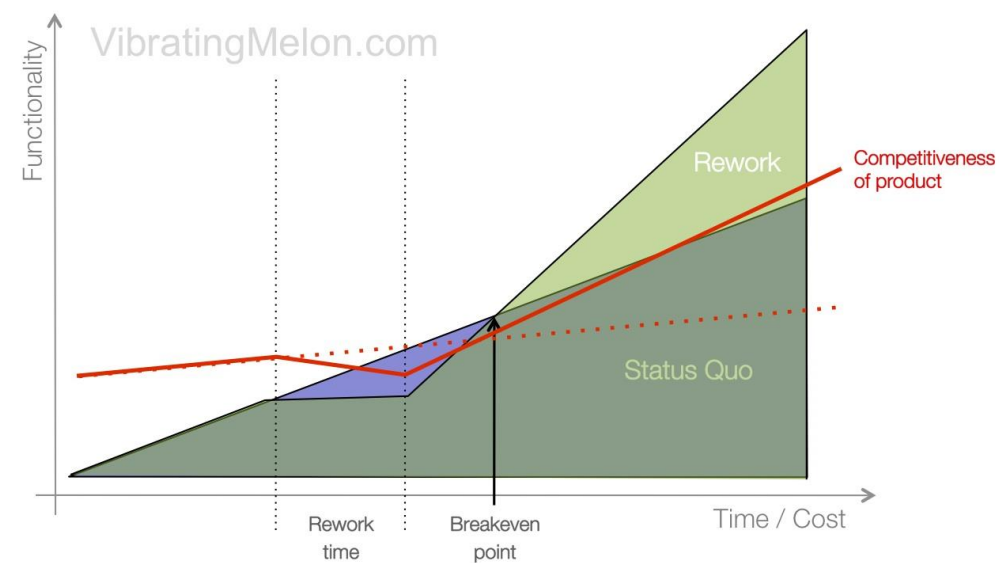
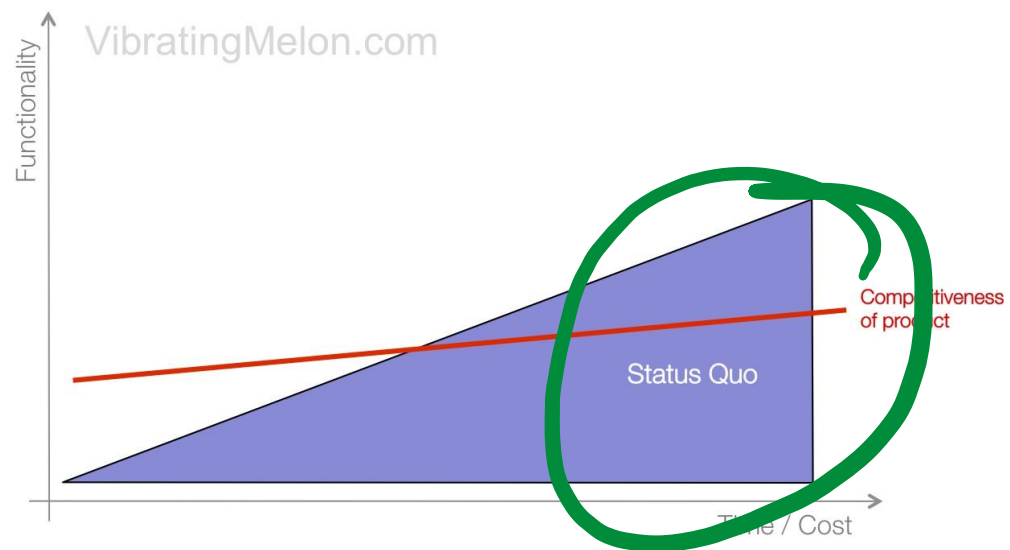




<https://vibratingmelon.com/2011/06/10/why-you-should-almost-never-rewrite-code-a-graphical-guide/>

- Your project needs ongoing maintenance
 - Nuisance expense
- Rewrite your project
 - It will still need ongoing maintenance
 - It will still be a nuisance expense
- The rewritten project will not have all the functionality
 - You can rewrite, but can you turn off the old project?





<https://vibratingmelon.com/2011/06/10/why-you-should-almost-never-rewrite-code-a-graphical-guide/>



Picard Tips @PicardTips · Feb 27



Picard strategy tip: Pick your battles. Fight injustices before annoyances.



4



392



1.1K



Build applications to bolt on new parts



Demo

Puzzles