Loving Legacy Code

Kathleen Dollard

Principal Program Manager, .NET Team

Microsoft

@KathleenDollard

kathleen.dollard@microsoft.com

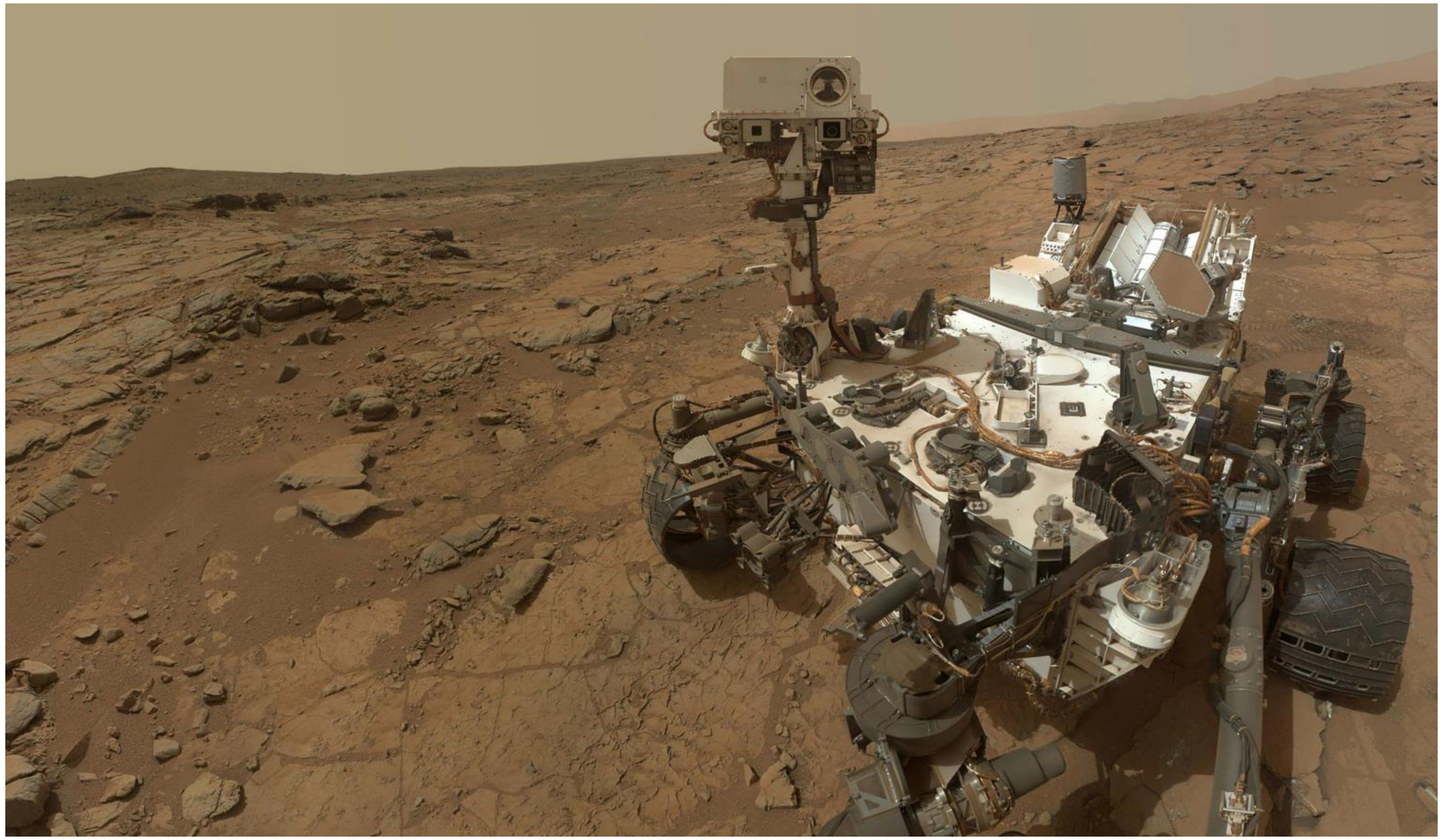
Legacy code is…

Code you approach from the point of view
of making changes
instead of designing from a blank sheet

All code that is in production

So how much legacy code is there?

- Your bank
- Your social media
- Your payroll
- Your health records
- Your car



Do you want this job?

- Old code base written for machines that are now obsolete
- Hardware is 15 years old
- Planned lifecycle - 3 months
- Very slow deployment cycle for any code changes
- When memory started failing due to age, someone added code to ignore memory faults
- No one plans to upgrade
- Even the bandwidth is tiny

What’s the one thing

we can say with confidence

about legacy code?

What's the one thing
we can say with confidence
about legacy code?

It works!!

Skills

- Test
- Debug
- Refactor

And attitude…

Learn

Teach

Enjoy

Don’t be a jerk

Respect legacy code

- Think long term
- Be gentle
- It’s really hard to replace, rewrites fail
- Leave it better than you found it, every time

Things you should never do, Part 1

Joel Spolsky, https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

“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…”

Things you should never do, Part 1

Joel Spolsky, https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

“…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 to install the thing on a computer that didn’t have Internet Explorer. Another one fixes that bug that occurs in low memory conditions. Another one fixes that bug that occurred when the file is on 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.”

Things you should never do, Part 1

Joel Spolsky, https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

"…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 to install the thing on a computer that didn't have Internet Explorer. Another one fixes that bug that occurs in low memory conditions. Another one fixes that bug that occurred when the file is on 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."

Things you should never do, Part 1

Joel Spolsky, https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

“…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 to install the thing on a computer that didn’t have Internet Explorer. Another one fixes that bug that occurs in low memory conditions. Another one fixes that bug that occurred when the file is on 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.”

Things you should never do, Part 1

Joel Spolsky, https://www.joelonsoftware.com/2000/04/06/things-you-should-never-do-part-i/

“…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 to install the thing on a computer that didn’t have Internet Explorer. Another one fixes that **bug** that occurs in low memory conditions. Another one fixes that **bug** that occurred when the file is on 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.”

What can we
say with confidence
about legacy code?

It works!!

What can we
say with confidence
about legacy code?

It works!!

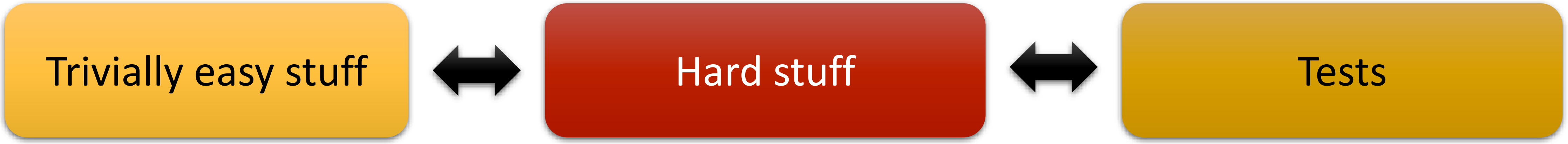
No one knows everything it does

Order of needs

1. Access to code and other artifacts
   - It will not get easier
2. Ability to build and deploy
   - Automated build
3. Source control
4. Regression test plan
   - Even specific manual test plan
5. Health tracking/tracing/telemetry
6. Bug tracking

Testing

Architecture for testing



Approval tests
Llewellyn Falco, http://approvaltests.com

- Write a scenario with output
- Run and have a human (you) approve output
- Use as regression test

But my code is so bad, I *have* to rewrite…

- Netscape 6
- Borland Arago and CA Clipper (dBase for Windows)
- Borland QuatroPro

…and so on

If your code is bad, fix it. One step at a time.

(from Joel Spolsky’s article)

Back to Joel

- First, there are architectural problems…These problems can be solved, one at a time, by carefully moving code, refactoring, changing interfaces...Even fairly major architectural changes can be done without *throwing away the code*.
- A second reason programmers think that their code is a mess is that it is inefficient…
- Third, the code may be doggone ugly…

Refactoring

Demo

A few tools…

- Analyzers, light bulb and Ctl-period
- Code Style
  - Tools/Options/Text Editor/[language]/Code Style
    - Export/import settings
  - .editor config
  - IntelliCode
- Code Analysis/FxCop
- Find clones (Enterprise only)

What’s the worst thing lurking in your code?

Zombie code

Kevlin Henny https://www.infoq.com/news/2017/02/dead-code

Re: Knight CapitalGroup

This bankruptcy-defining event arose from a perfect storm. In anticipation of a new NYSE system, to be launched on the 1st of August, they had deployed updates to their servers. They updated their servers manually and, unbeknown to them, one of the deployments failed, leaving the old version running. To take advantage of the new NYSE system, they recycled an old flag, a flag that was no longer used but had now been repurposed to mean something different. Although it hadn't been used in eight years, the old version of the code still had a dependency on the old flag.

The code had been dead for years, but was awakened by a change to the flag's value. The zombie apocalypse arrived and the rest is bankruptcy.

Zombie code

Kevlin Henny https://www.infoq.com/news/2017/02/dead-code

Re: Knight CapitalGroup – 450 million in 45 minutes

This bankruptcy-defining event arose from a perfect storm. In anticipation of a new NYSE system, to be launched on the 1st of August, they had deployed updates to their servers. They **updated their servers** manually and, unbeknown to them, **one of the deployments failed, leaving the old version running**. To take advantage of the new NYSE system, they recycled an old flag, a flag that was no longer used but had now been repurposed to mean something different. Although it hadn’t been used in eight years, the old version of the code still had a dependency on the old flag.

The code had been dead for years, but was awakened by a change to the flag’s value. The zombie apocalypse arrived and the rest is bankruptcy.

Zombie code

Kevlin Henny https://www.infoq.com/news/2017/02/dead-code

Re: Knight CapitalGroup – 450 million in 45 minutes

This bankruptcy-defining event arose from a perfect storm. In anticipation of a new NYSE system, to be launched on the 1st of August, they had deployed updates to their servers. They **updated their servers** manually and, unbeknown to them, **one of the deployments failed, leaving the old version running**. To take advantage of the new NYSE system, they **recycled an old flag**, a flag that was no longer used but had now been repurposed to mean something different. **Although it hadn't been used in eight years, the old version of the code still had a dependency on the old flag**.

The code had been dead for years, but was awakened by a change to the flag's value. The zombie apocalypse arrived and the rest is bankruptcy.

Zombie code

Kevlin Henny https://www.infoq.com/news/2017/02/dead-code

Re: Knight CapitalGroup – 450 million in 45 minutes

This bankruptcy-defining event arose from a perfect storm. In anticipation of a new NYSE system, to be launched on the 1st of August, they had deployed updates to their servers. They **updated their servers** manually and, unbeknown to them, **one of the deployments failed, leaving the old version running**. To take advantage of the new NYSE system, they **recycled an old flag**, a flag that was no longer used but had now been repurposed to mean something different. **Although it hadn’t been used in eight years, the old version of the code still had a dependency on the old flag**.

The code had been dead for years, but was awakened by a change to the flag’s value. The zombie apocalypse arrived and the rest is bankruptcy.

What else?

By Carol M. Highsmith - Library of CongressCatalog: http://lccn.loc.gov/2011630866Image download: https://cdn.loc.gov/master/pnp/highsm/12600/12672a.tifOriginal url: http://hdl.loc.gov/loc.pnp/highsm.12672, Public Domain, https://commons.wikimedia.org/w/index.php?curid=52220198

Contraption Code

James Gleick https://www.around.com/ariane.html

“But in this case, the programmers had decided that this particular velocity figure would never be large enough to cause trouble. After all, it never had been before. Unluckily, Ariane 5 was a faster rocket than Ariane 4. One extra absurdity: the calculation containing the bug, which shut down the guidance system, which confused the on-board computer, which forced the rocket off course, actually served no purpose once the rocket was in the air. Its only function was to align the system before launch. So it should have been turned off. But engineers chose long ago, in an earlier version of the Ariane, to leave this function running for the first 40 seconds of flight -- a "special feature" meant to make it easy to restart the system in the event of a brief hold in the countdown.”

Contraption Code

James Gleick https://www.around.com/ariane.html

“But in this case, the programmers had **decided that this particular velocity figure would never be large enough to cause trouble**. After all, it never had been before. Unluckily, Ariane 5 was a faster rocket than Ariane 4. One extra absurdity: the calculation containing the bug, which shut down the guidance system, which confused the on-board computer, which forced the rocket off course, actually served no purpose once the rocket was in the air. Its only function was to align the system before launch. So it should have been turned off. But engineers chose long ago, in an earlier version of the Ariane, to leave this function running for the first 40 seconds of flight -- a "special feature" meant to make it easy to restart the system in the event of a brief hold in the countdown.”

Contraption Code

James Gleick https://www.around.com/ariane.html

“But in this case, the programmers had **decided that this particular velocity figure would never be large enough to cause trouble**. After all, it never had been before. Unluckily, **Ariane 5 was a faster rocket** than Ariane 4. One extra absurdity: the calculation containing the bug, which shut down the guidance system, which confused the on-board computer, which forced the rocket off course, actually served no purpose once the rocket was in the air. Its only function was to align the system before launch. So it should have been turned off. But engineers chose long ago, in an earlier version of the Ariane, to leave this function running for the first 40 seconds of flight -- a "special feature" meant to make it easy to restart the system in the event of a brief hold in the countdown.”

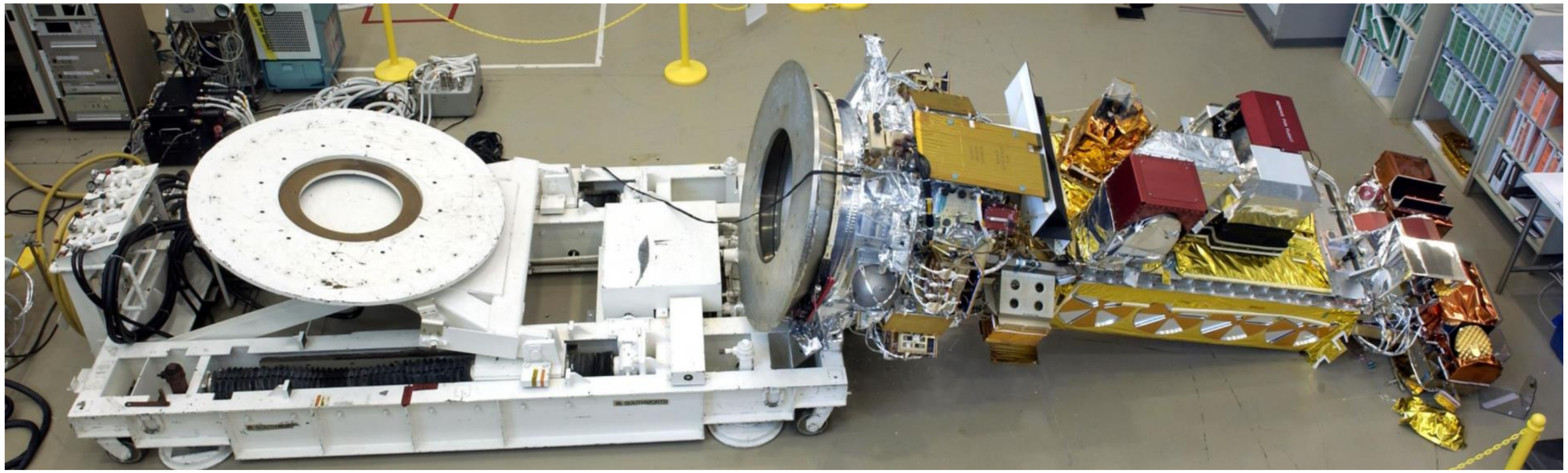
Contraption Code

James Gleick https://www.around.com/ariane.html

“But in this case, the programmers had **decided that this particular velocity figure would never be large enough to cause trouble**. After all, it never had been before. Unluckily, **Ariane 5 was a faster rocket** than Ariane 4. One extra absurdity: the calculation containing the bug, which shut down the guidance system, which confused the on-board computer, which forced the rocket off course, **actually served no purpose once the rocket was in the air**. Its only function was to align the system before launch. **So it should have been turned off**. But engineers chose long ago, in an earlier version of the Ariane, to leave this function running for the first 40 seconds of flight -- **a "special feature" meant to make it easy to restart the system in the event of a brief hold in the countdown.**”

Solution is not to rewrite the code

- Respect the code and the coders
- Add basics that support stability, like testing and tracing
- Fix bugs thoughtfully
- Be vigilant on dead, spurious, unused and redundant code



$135M

Rewriting code
and not shutting down the previous system

hides a ton of zombie code,

results in hybrid contraptions, and

risks pulling the bolts out

Should you ever rewrite code?

Keep current

- Constant evolution extends lifetime
- Move to modern tech makes sense
- Architecture separates concerns

Static support

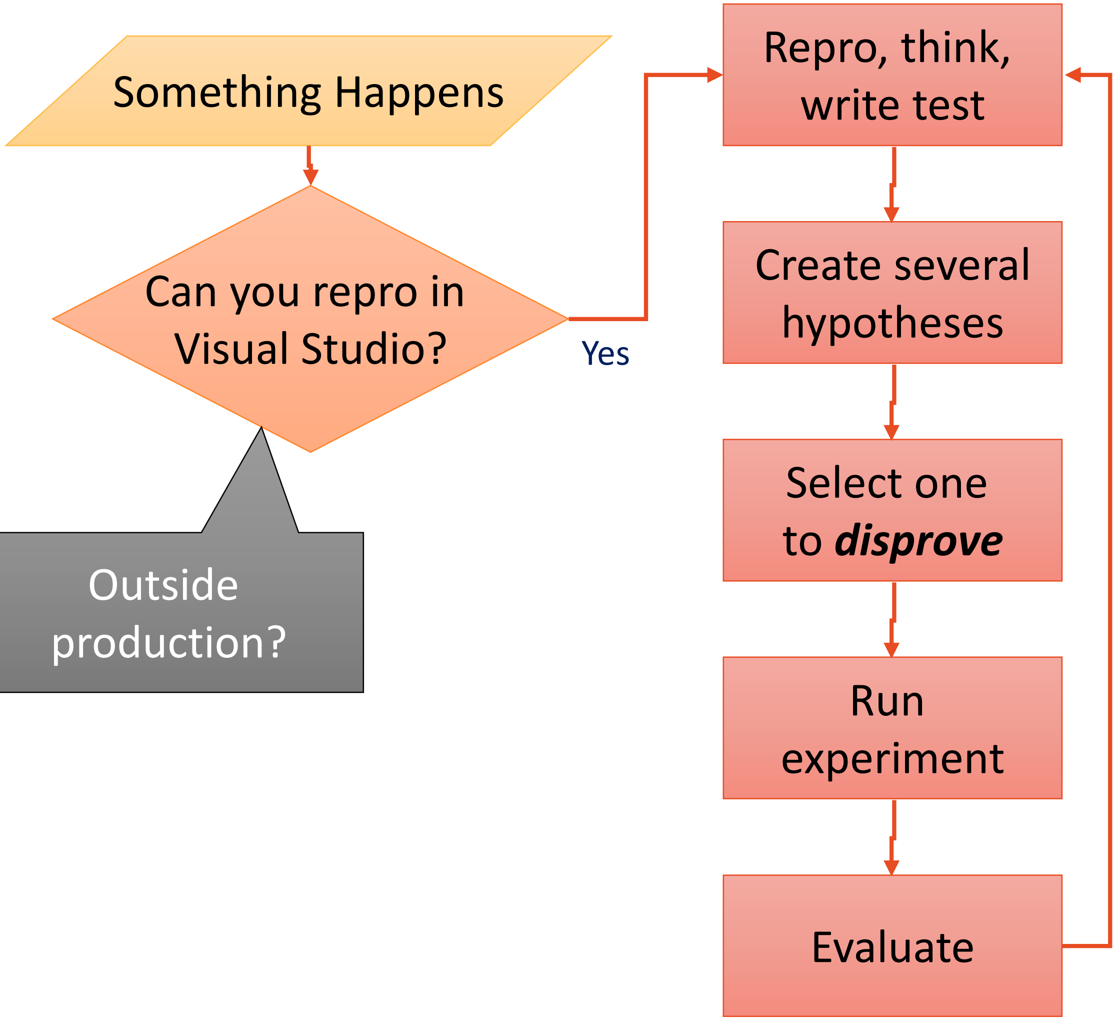
- Little change now or expected
- Technology is stable, if old
- Focus refactoring on separating concerns

Plan to abandon

- Technology is out of support
- Large percent of app affected by business change (ie. GDPR)
- Try to save business logic

Debugging

A Debugging Strategy - *Scientific Method*



Learn

Teach

Enjoy

Don’t be a jerk

Learn *from your working code*

Teach

Enjoy

Don’t be a jerk

Learn *from your working code*

Teach *specific skills for legacy code*

Enjoy

Don’t be a jerk

Learn *from your working code*

Teach *specific skills for legacy code*

Enjoy *consistently making code better*

Don’t be a jerk

Learn *from your working code*

Teach *specific skills for legacy code*

Enjoy *consistently making code better*

Don’t be a jerk – *value working code*

Skills

- Test
- Debug
- Refactor

What can we
say with confidence
about legacy code?

It works!!

No one knows everything it does

What can we
say with confidence
about legacy code?

It works!!

No one knows everything it does

You can make it better

What can we say with confidence about legacy code?

It works!!

No one knows everything it does

You can make it better

Thank you!

@KathleenDollard11111

Kathleen.Dollard@Microsoft.com