Become a Better Developer With Debugging

Dustin Younse
Software Engineer, Acquia
@milsyobtaf

Rob Ristroph
Technical Architect, Acquia
@robgr

The Outline

1. What is a Bug
2. What is Debugging
3. Why it’s Important
4. “Scientific Method” Approach
5. Toolbox
6. Other tricks
7. More reading

What Is A Bug?

Your mental model of the code and it’s actual behaviour don’t match.

Usually you typed code that you thought did one thing and in fact it did another.

Most of the bugs you work on are your own.

Pobody's Nerfect

Glitch 🎏
@glitch

Follow

My code once took down a client's Intranet mid-demo. Tell us your coding screw-ups, so we can show new coders #weallmakemistakes 🤦‍♀️

> **Glitch** 🎏 @glitch
> This Junior Dev nuked the prod db on 1st day of new job:
> reddit.com/r/cscareerques…

Retweets	Likes
330	617

5:02 AM - 13 Jun 2017

335 330 617

Zach Holman
@holman

Follow

Replying to @glitch

i shipped github to production in development mode, that was neat

Retweets	Likes
9	85

6:46 AM - 13 Jun 2017

2 9 85

jennosaur js 🏔🏔🏔

@jennschiffer

Follow

i miiiight have also taken down the china nba site for a few min while switching the nets from nj to brooklyn (long after they moved fyi)

Likes
26

6:24 AM - 13 Jun 2017

1 26

K Cravens
@CakeRavens

Follow

Replying to @glitch

My first client patch caused all their W2's to print with the "deceased" box checked. It was a nunnery. I KILLED AN ENTIRE NUNNERY.

Retweets 30 **Likes** 155

5:10 AM - 13 Jun 2017

3 30 155

Dustin Younse
@milsyobtaf

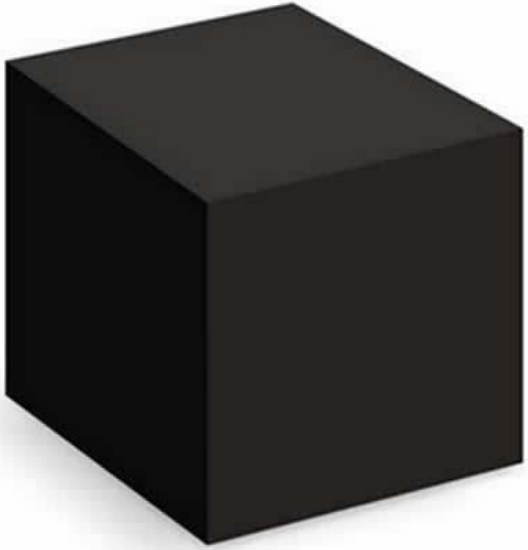
Follow

Replying to @glitch

I took down a university library on the 1st day of the semester with a botched FTP transfer of .htaccess. Better than Finals Week I suppose.

10:21 AM - 14 Jun 2017

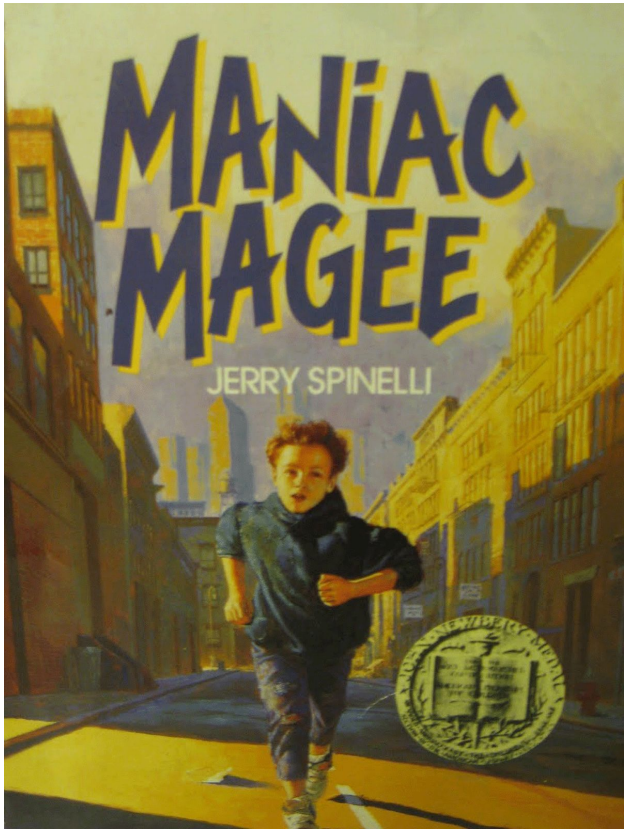
Debugging vs Troubleshooting



Basic Mental Model of Drupal

Advanced Mental Model of Drupal

A Brief Digression

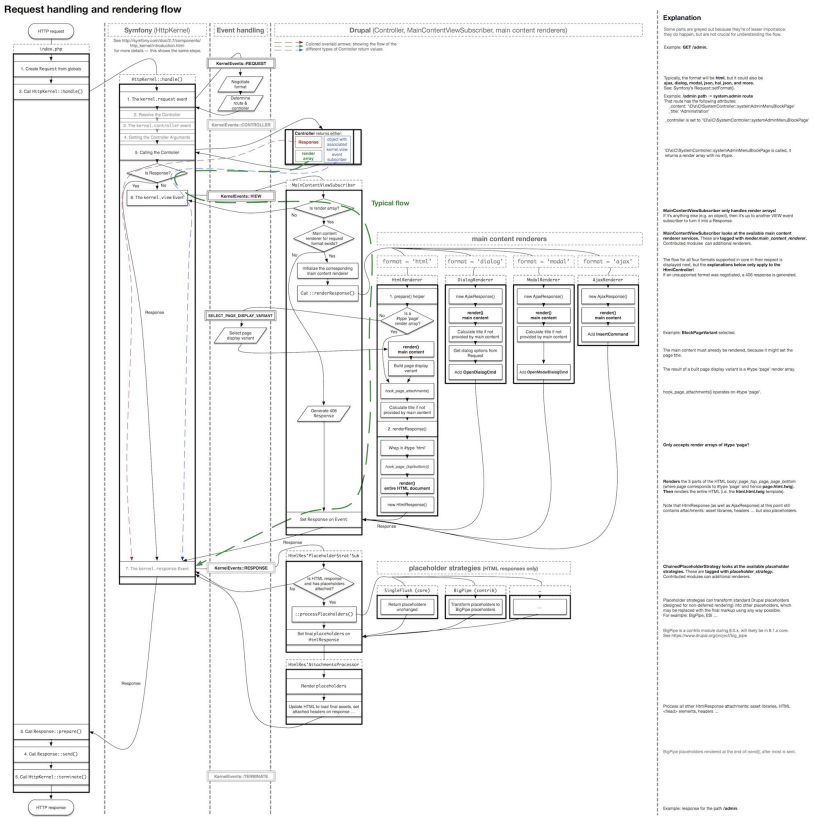


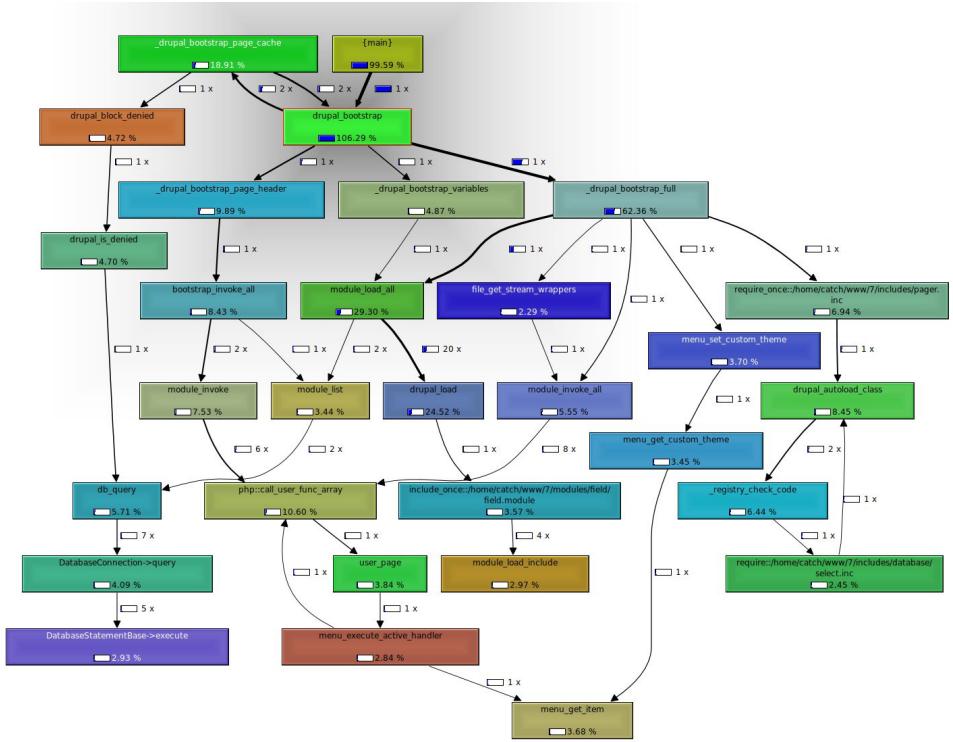
A Book About A Boy Genius (At Untangling Knots)

The Fearsome Cobble's Knot

A Basic Mental Model of Drupal

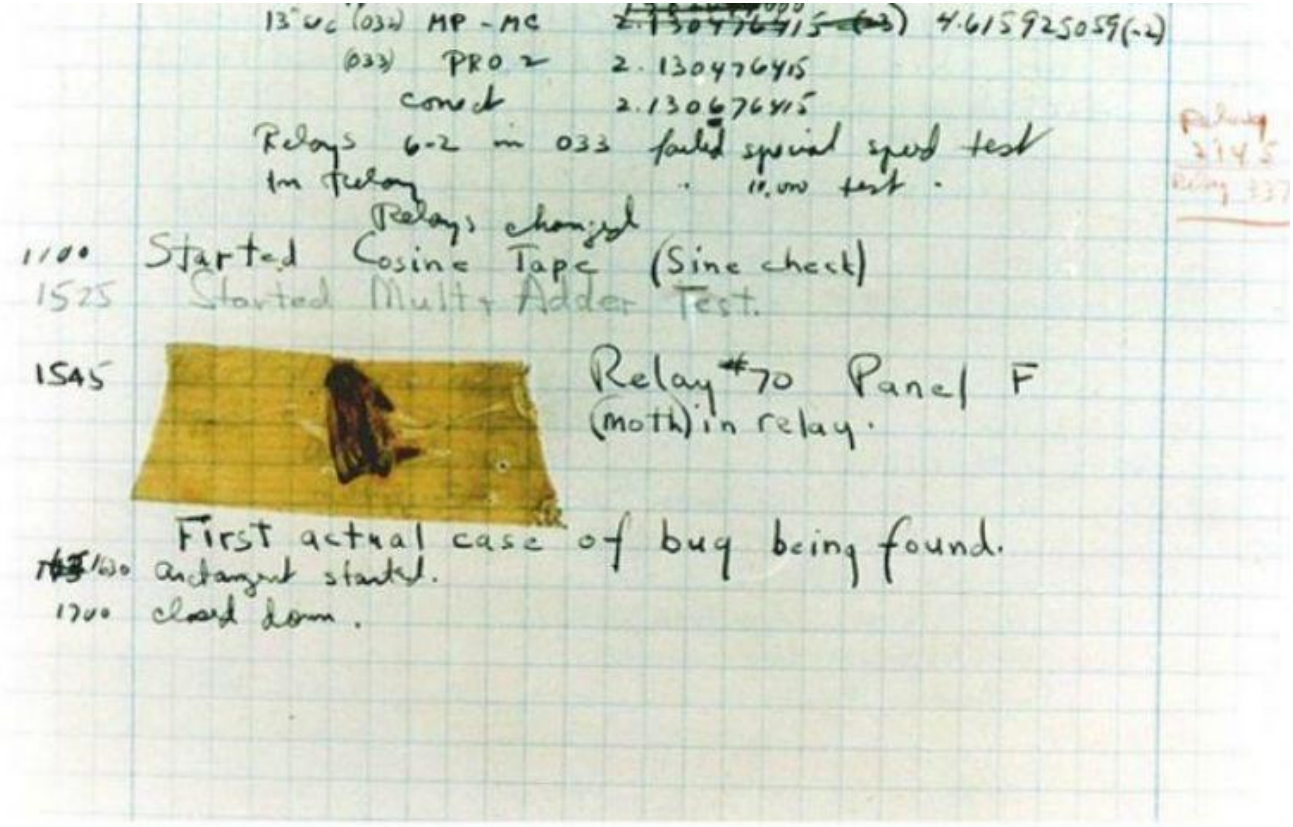
Back to Drupal





An Advanced Mental Model of Drupal

A Divergence On Origin



The First Computer Bug

So...

What *is* Debugging?

Debugging Is:
The Process of Making Your Mental Model Match Reality

Understanding WHY the bug happened is different from fixing it.

Why is Debugging Important?

You spend more time *debugging* than you do *programming*.

Furthermore the time debugging is much harder to estimate.

Why is Debugging Important?

> “As soon as we started programming, we found to our surprise that it wasn't as easy to get programs right as we had thought. Debugging had to be discovered. I can remember the exact instant when I realized that a large part of my life from then on was going to be spent in finding mistakes in my own programs.”

– Maurice Wilkes, 1949, *on developing the first stored program computer*

Why is Debugging Important?

- You do it more than you realize.
- It’s the source of much uncertainty in estimating and delivery.
- As a distinct thought process / skill, it is possible to become good and more efficient at it.

The Scientific Method of Debugging

1. Observe - Collect data, as much as possible
2. Make a testable Hypothesis
    a. Change to your mental model
3. Collect data from the test
4. Adjust understanding of the model
5. goto 1

What Does This Look Like In Real Life?

Something Is Broken!!!

Relax.

DHH
@dhh

Unless you're making software for rockets, self-driving cars, or pacemakers, ease off on the "mission critical" bullshit, yeah?

RETWEETS 353 LIKES 968

7:22 AM - 15 Feb 2017

50 353 968

Remember Cobble's Knot

What Exactly Is Broken?

- Is something not showing up?
  - New content - is it published? Front end cache?
  - Old content - permissions set properly, or changed?
- Is something showing up that shouldn't?
  - Raw html or javascript in a wysiwyg field?
- A more complex behavior - workbench or etc - can we state exactly the steps to cause the bug, and why it's not what we expect?

Non-technical members of your team have huge impact collecting data at this stage.

Replicate The Bug

- User reports matter
- Worst case is making changes, waiting to see if the customer reports the problem is still there
- Replication can be tedious, but extremely valuable
- Observe and think about your user's operating procedure
- Without being able to replicate the bug, *you can't debug*.

Sometimes figuring out how to replicate the bug is 99% of fixing it.

Work From The Bottom Up

- Log files
  - Know where they are on your systems / environments
- Multitail
  - Linux / Mac utility to easily view logs, with more options
- Contextual information - browsers, environments, users

Vacuum up as much information as possible in the first stage.

Where Is It Broken?

- Custom module
- Theme template.php / .template file
- Configuration in database

Potential tests - disable modules, switch themes, re-install clean without live data.

Divide-and-conquer by narrowing down where the mental model breaks.

Debugging As Scientific Method

1. Change ONE thing at a time
2. Test that change
3. Repeat - undoing the change if it gave no information

Better debuggers are generally better at thinking of clever changes and tests.

- “Cheap” tests first (clear caches, etc)
- Test for common problems first
- A good test should narrow the problem scope by eliminating *something*

git Is Your Friend

Save your progress as you work

- Re-create your Features / Export your Config
- Quickly undo unhelpful changes
- Helps to make Rabbit Holes manageable

Better debuggers generally take notes and keep a log.

git diff Is Your Friend

- Remove debug statements
  - You only commit print_r(‘Butts’); to master ONCE
- Ensure you only changed as much as necessary

The less code you change, the fewer bugs you might create.

git blame Is Your Friend

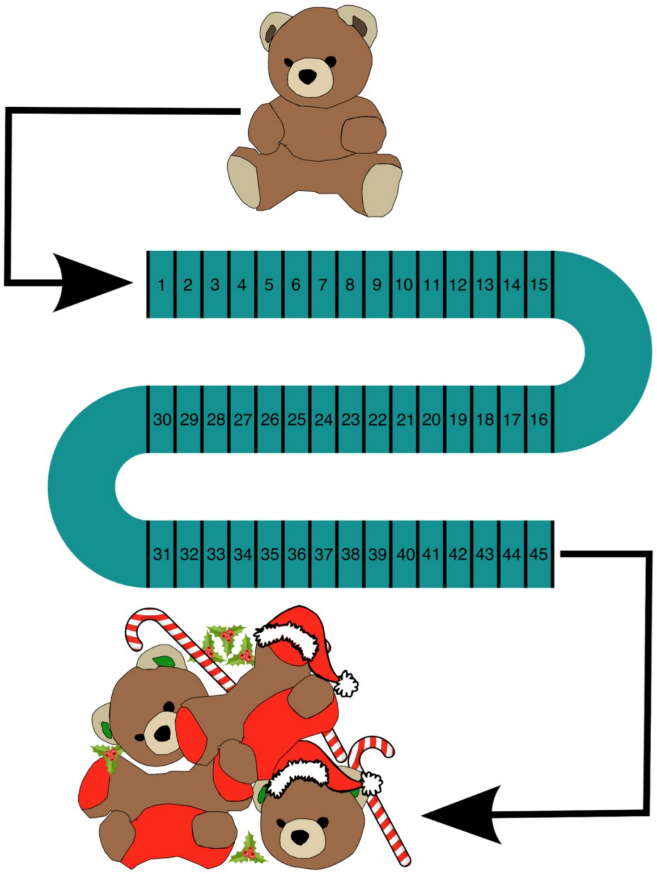
- Who wrote (or at least committed) the offending code
- Should *NOT* be a witch hunt
- *Should* be a chance to understand the context of the code
  - Re-reading the old Jira tickets or other requirements can cause you to re-assess everything

You can use “git annotate” in politically sensitive situations.

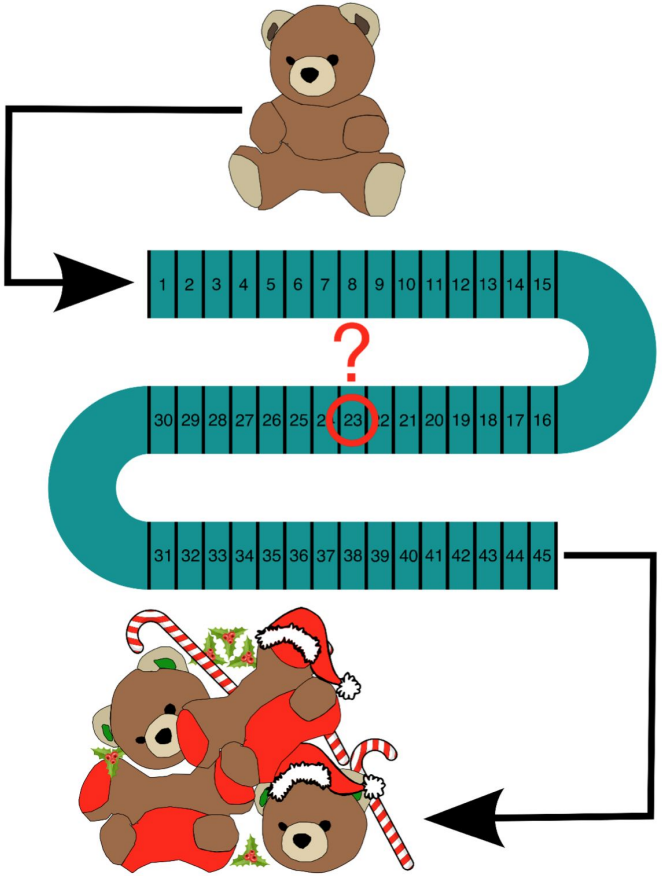
git bisect Is Your Friend

- Let's you run quick and dirty A/B style tests in your failing code
- Iterates through known good and known bad states to find the issue

You can only use "git bisect" if you have multiple granular commits.



git bisect Is Your Friend



git bisect Is Your Friend

Make The Future Easier

- Watchdog (D7)
- \Drupal::Logger() (D8)
- syslog module
  - https://sumologic.com
  - https://loggly.com
- Write a test!
  - PHPUnit
  - Behat

Thoughtful instrumentation of your code as it's written the first time can massively pay off later.

Unit test vs. Integration test

“Interaction” Bugs are the Hardest

The hardest bugs are those that only appear when two “bug free” components interact.

- Module weights, order of hook operations
  - Systematically disable modules, change weights
- Theme / module interactions
- External service requests

If your problem resists divide-and-conquer, maybe it’s not in one component or the other, but in how they connect.

Performance Related Debugging

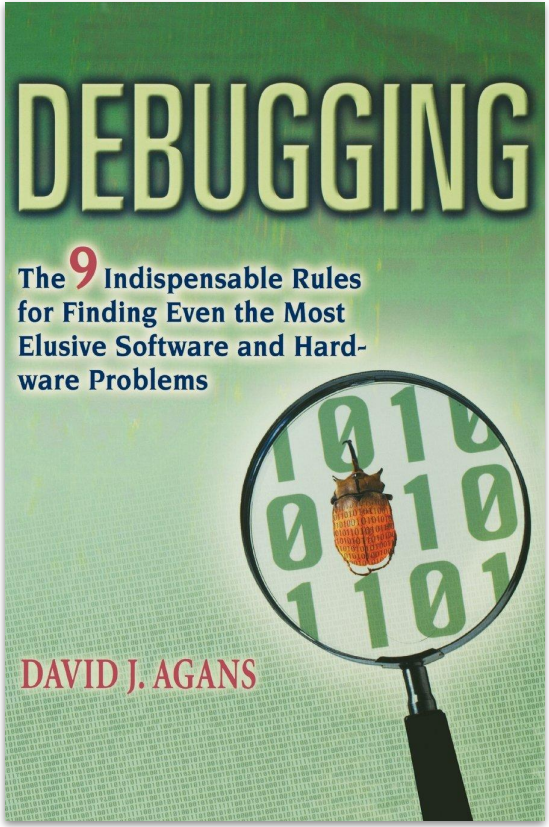
Just like other debugging

- Replicate the problem! Otherwise you flail at random
- Apache bench (ab), wget spiders, load generators
- Add headers, log statements, to indicate cache hits /misses
- Different logs often apply - mysql or system logs

Further Reading

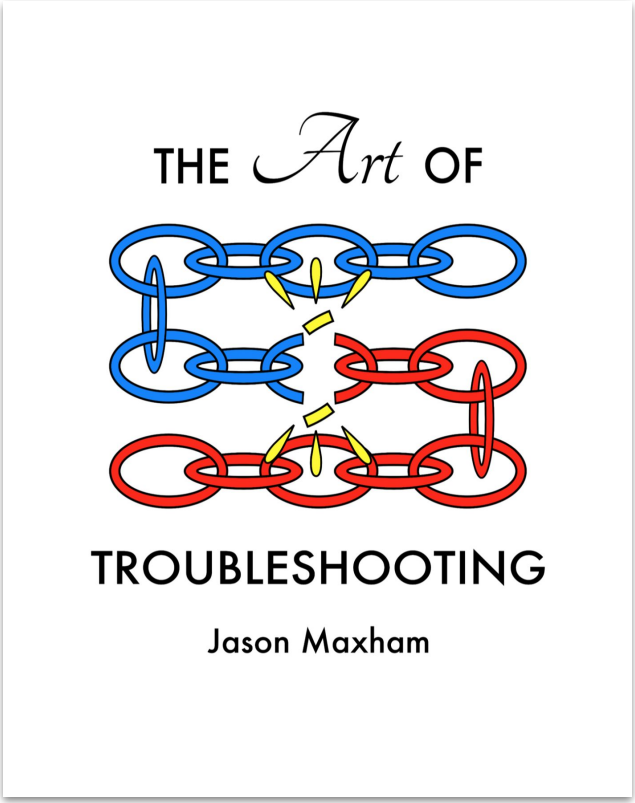
Debugging: The Nine Indispensable Rules

1. Understand the System
2. Make it Fail
3. Quit Thinking and Look
4. Divide and Conquer
5. Change One Thing at a Time
6. Keep an Audit Trail
7. Check the Plug
8. Get a Fresh View
9. If You Didn't Fix It, It Ain't Fixed



The Art of Troubleshooting

1. Strategies
2. Virtues
3. Cleaning Up



Conclusions

- Thinking strategically is more important than applying fancy tools
- The hardest bugs are “Interaction” bugs

Finally...

Debugging can be hard to tell someone how to do, but it can be learned if you persist and think about it. Level up!

Reference Links

- The First Bug
  - https://en.wikipedia.org/wiki/Software_bug#Etymology
- Debugging: The Nine Indispensable Rules, by David J. Agans
  - http://www.debuggingrules.com
- The Art of Troubleshooting, by Jason Maxham
  - https://artoftroubleshooting.com
- Maniac Magee, by Jerry Spinelli
  - https://www.worldcat.org/oclc/20422223

Become a Better Developer With Debugging

Dustin Younse
Software Engineer, Acquia
@milsyobtaf

Rob Ristroph
Technical Architect, Acquia
@robgr