

# Lecture 9: Software Quality in Practice

17-313: Foundations of Software Engineering

Rohan Padhye, Michael Hilton, Chris Timperley, and Daye Nam

# Administrivia

- Missing lecture policy (see Syllabus on website)
- HW2: In-person presentation during this week's recitation
- HW3A: plan due on Thursday
  - Implementation due October 6th
  - Reflection due October 13th
- Midterm is on October 11th
  - review will take place in the week before the midterm

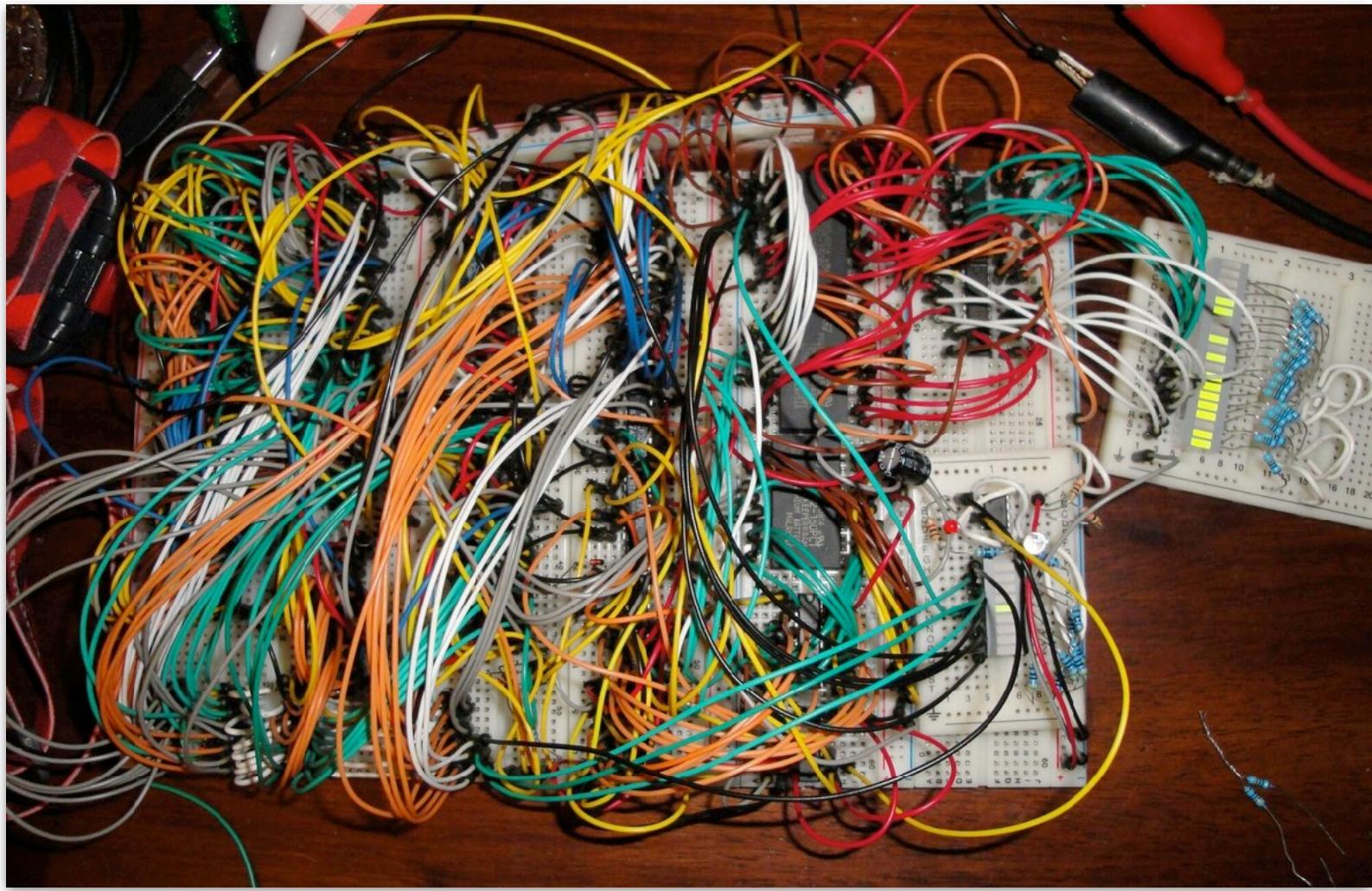
# Learning Goals

- Understand the concepts of software quality and technical debt
- Reflect on personal experiences of technical debt
- Learn best practices for proactively ensuring quality
- Design an explicit QA process for your project
- Learn techniques for reactively dealing with quality problems

# Software Quality



# Internal Quality



- Is the code well structured?
- Is the code understandable?
- How well tested is the code?

# External Quality



- Does the software crash?
- Does it meet the requirements?
- Is the UI well designed?

# How would you rate Teedy?

Screenshot of the GitHub repository for `sismics/docs`. The repository has 1,074 commits, 2 branches, and 11 tags. A pull request titled "jendib #647: fix doc" is shown as the latest commit. The repository page includes sections for About, Releases, and a sidebar with a "teedy.io" badge.

**About**

Lightweight document management system packed with all the features you can expect from big expensive solutions

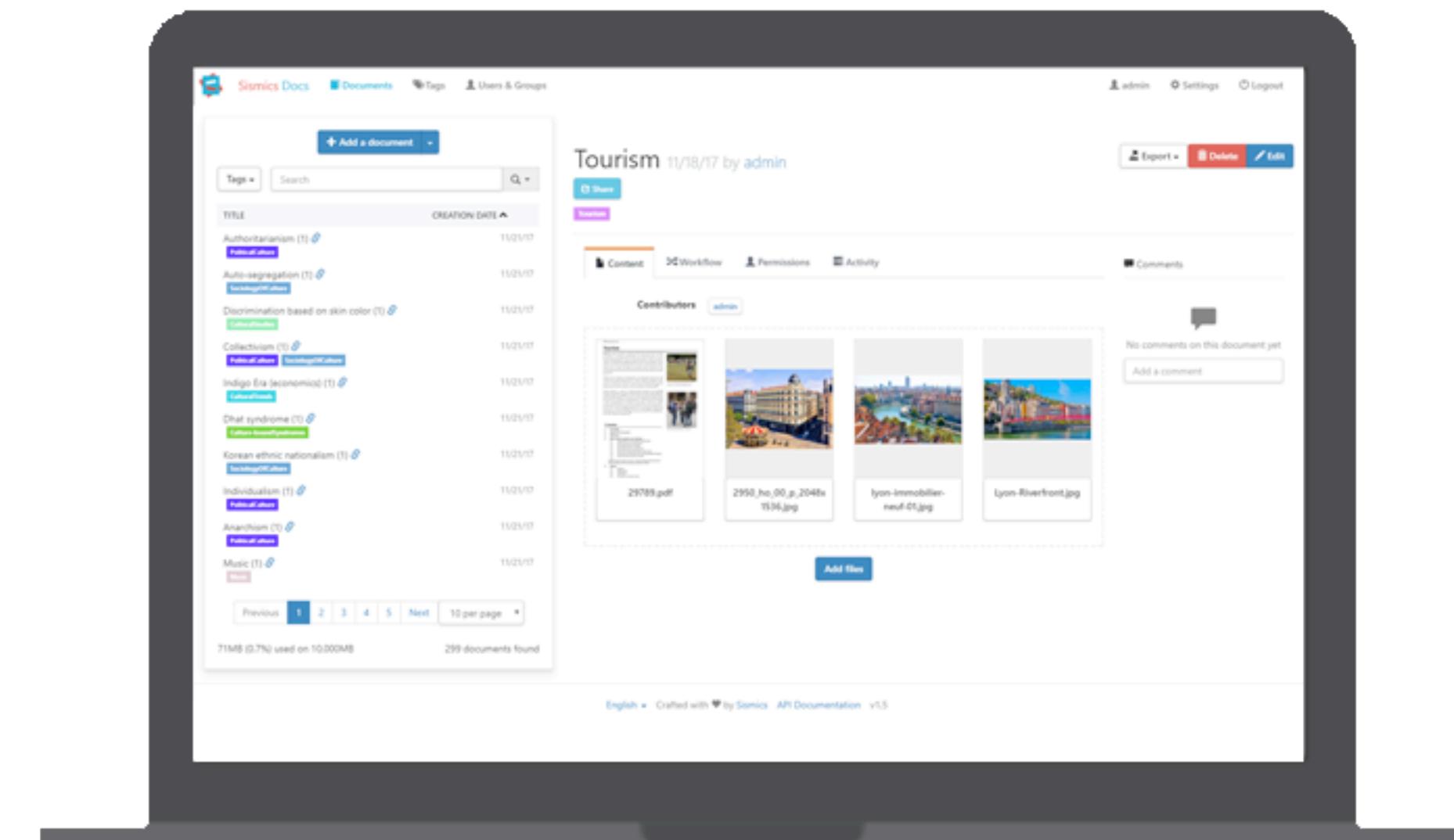
**teedy.io**

javascript, java, docker, open-source, enterprise, workflow, cloud, ocr, sharing, file-sharing, document, self-hosting, dms

**Releases** 11

v1.10 (Latest) on Jan 2

teedy

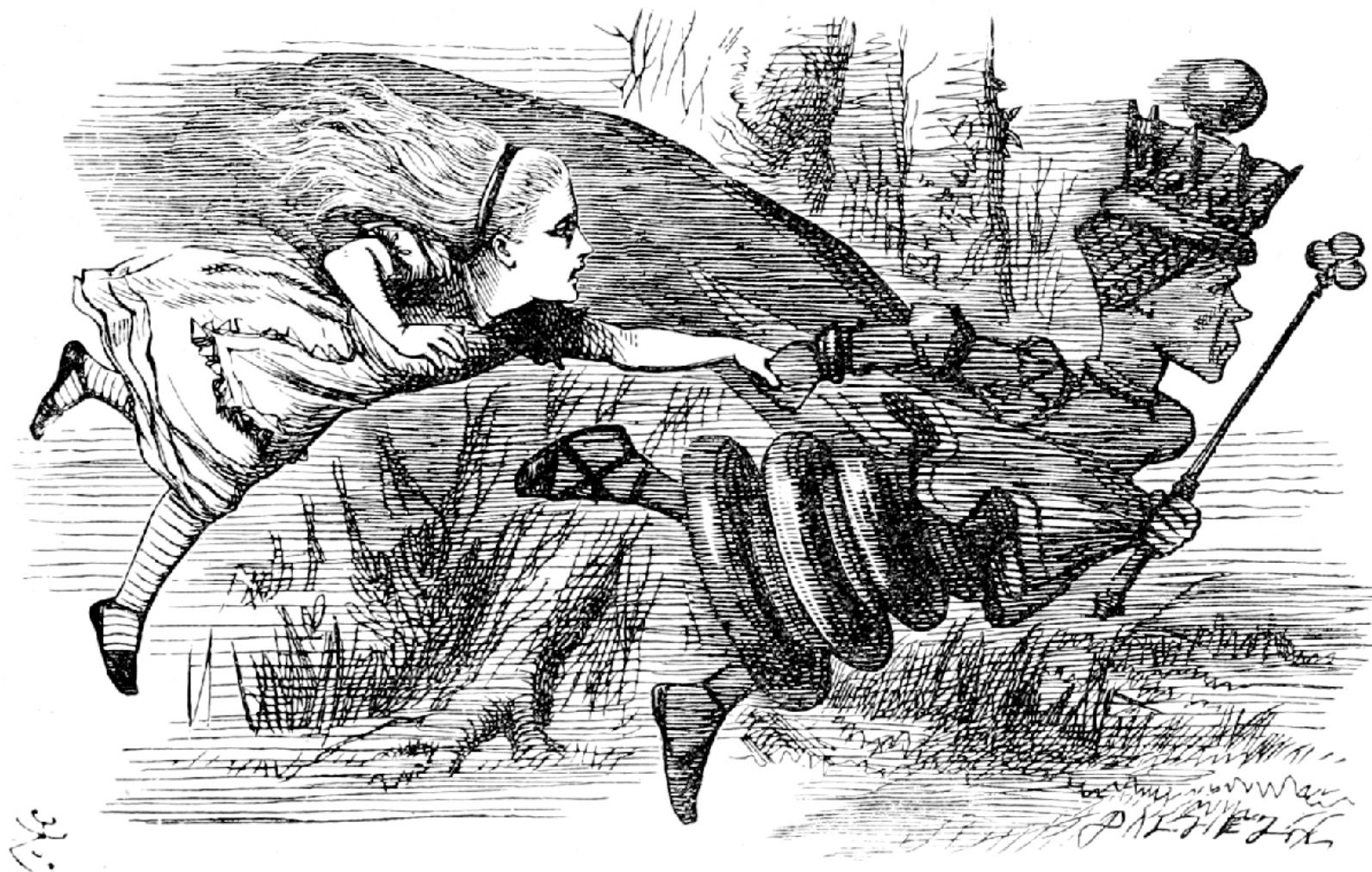


**How did we get here?**

# Software entropy

“As an evolving program is continually changed, its complexity, reflecting deteriorating structure, increases unless work is done to maintain or reduce it.”

*Meir Manny Lehman*



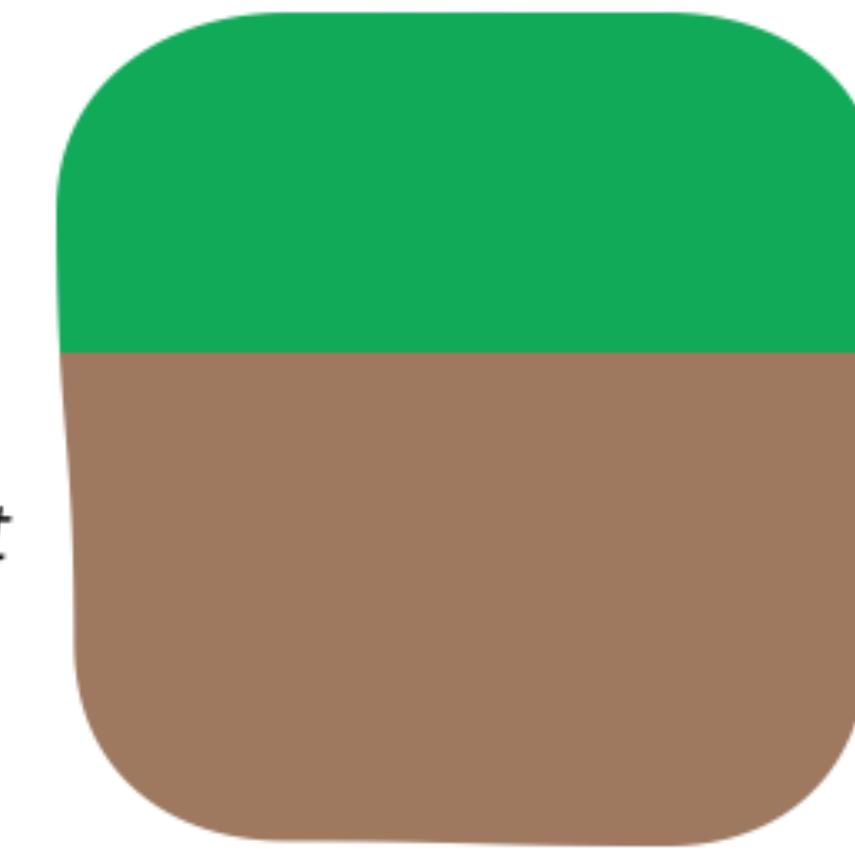
“Now, here, you see, it takes all the running you can do just to keep in the same place. If you want to get somewhere else, you must run at least twice as fast!”

*Through the Looking Glass*

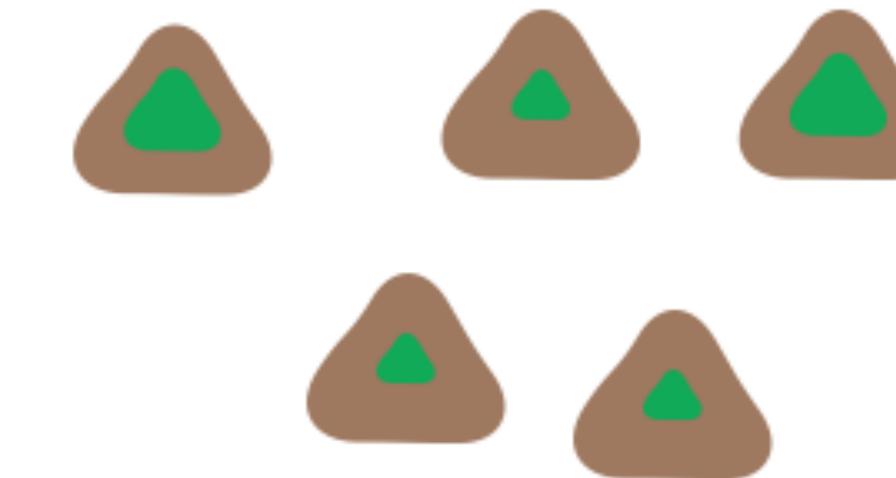
# Technical debt

*Any software system has a certain amount of **essential** complexity required to do its job...*

*... but most systems contain **cruft** that makes it harder to understand.*



*Cruft causes changes to take **more effort***



*The technical debt metaphor treats the cruft as a debt, whose *interest payments* are the extra effort these changes require.*

<https://martinfowler.com/bliki/TechnicalDebt.html>

# A better analogy: Pollution

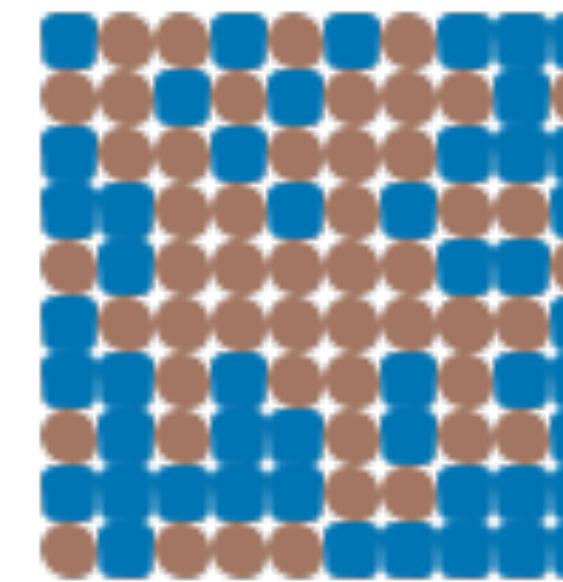


AQI Basics for Ozone and Particle Pollution			
Daily AQI Color	Levels of Concern	Values of Index	Description of Air Quality
Green	Good	0 to 50	Air quality is satisfactory, and air pollution poses little or no risk.
Yellow	Moderate	51 to 100	Air quality is acceptable. However, there may be a risk for some people, particularly those who are unusually sensitive to air pollution.
Orange	Unhealthy for Sensitive Groups	101 to 150	Members of sensitive groups may experience health effects. The general public is less likely to be affected.
Red	Unhealthy	151 to 200	Some members of the general public may experience health effects; members of sensitive groups may experience more serious health effects.
Purple	Very Unhealthy	201 to 300	Health alert: The risk of health effects is increased for everyone.
Maroon	Hazardous	301 and higher	Health warning of emergency conditions: everyone is more likely to be affected.

<https://www.airnow.gov/aqi/aqi-basics>

# Internal quality makes it easier to add features

*If we compare one system with a lot of cruft...*

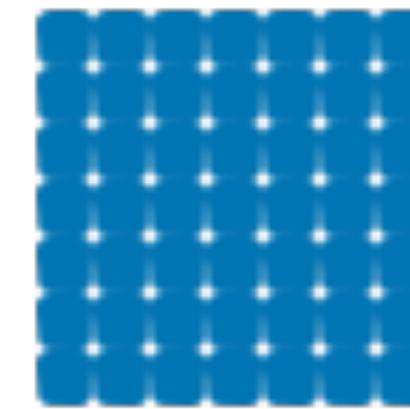


*the cruft means new features take longer to build*



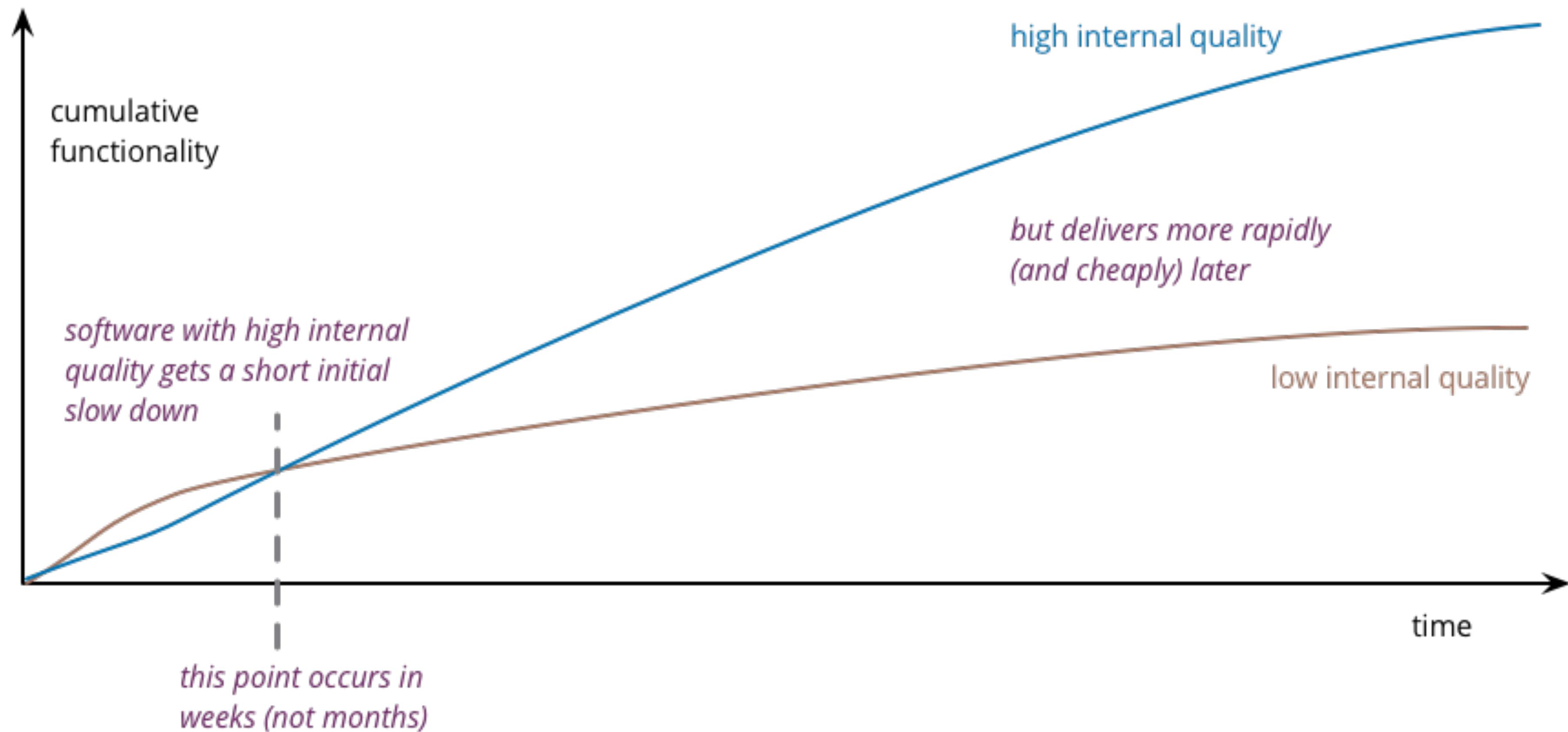
*this extra time and effort is the cost of the cruft, paid with each new feature*

*...to an equivalent one without*



*free of cruft, features can be added more quickly*

# High internal quality is an investment



# What causes technical debt?

# What causes technical debt?

- Tightly-coupled components
- Poorly-specified requirements
- Business pressure
- Lack of process
- Lack of documentation
- Lack of automated testing
- Lack of knowledge
- Lack of ownership
- Delayed refactoring
- Multiple, long-lived development branches
- ...

# Not all technical debt is the same

	Reckless	Prudent
Deliberate	<i>"We don't have time for design"</i>	<i>"We must ship now and deal with consequences (later)"</i>
Inadvertent	<i>"What's layering?"</i>	<i>"Now we know how we should have done it"</i>

<https://martinfowler.com/bliki/TechnicalDebtQuadrant.html>

# EVERYONE CREATES TECHNICAL DEBT



# Bad: Too much technical debt

- Bad code can be demoralizing
- Conversations with the client become awkward
- Team infighting
- Atrophied skills
- Turnover and attrition
- ...



**When should we reduce technical debt?**

# Dealing with technical debt: Fixing broken windows



# Alternative: Putting out fires is expensive!



# Analogy: Cleaning your dryer



# **How can we avoid technical debt?**

# Reflection: Homework 2

The screenshot shows a course website for 17-313 Foundations of Software Engineering. The main navigation bar includes links for Assignments, Recitations, Calendar, Schedule, Staff, and Syllabus. The Assignments section is expanded, showing HW01 Hello Teedy, HW02 Learning the Codebase (which is selected), and HW03 Teamwork. The HW02 page contains a search bar, a breadcrumb trail (Assignments / HW02 Learning the Codebase), and the title "17-313: FOUNDATIONS OF SOFTWARE ENGINEERING". Below the title is a section titled "HOMEWORK 2: LEARNING THE CODEBASE". Under "LEARNING GOALS", there is a bulleted list of tasks related to learning the codebase and collaborating with teammates. The "SPECIFICATION" section describes the project requirements, mentioning the use of an existing document management system called Teedy.

17-313  
Foundations of  
**SOFTWARE  
ENGINEERING**

ASSIGNMENTS

HW01 HELLO TEEDY

**HW02 LEARNING  
THE CODEBASE**

HW03 TEAMWORK

RECITATIONS

CALENDAR

SCHEDULE

STAFF

SYLLABUS

Search 17-313

Assignments / HW02 Learning the Codebase

## 17-313: FOUNDATIONS OF SOFTWARE ENGINEERING

### HOMEWORK 2: LEARNING THE CODEBASE

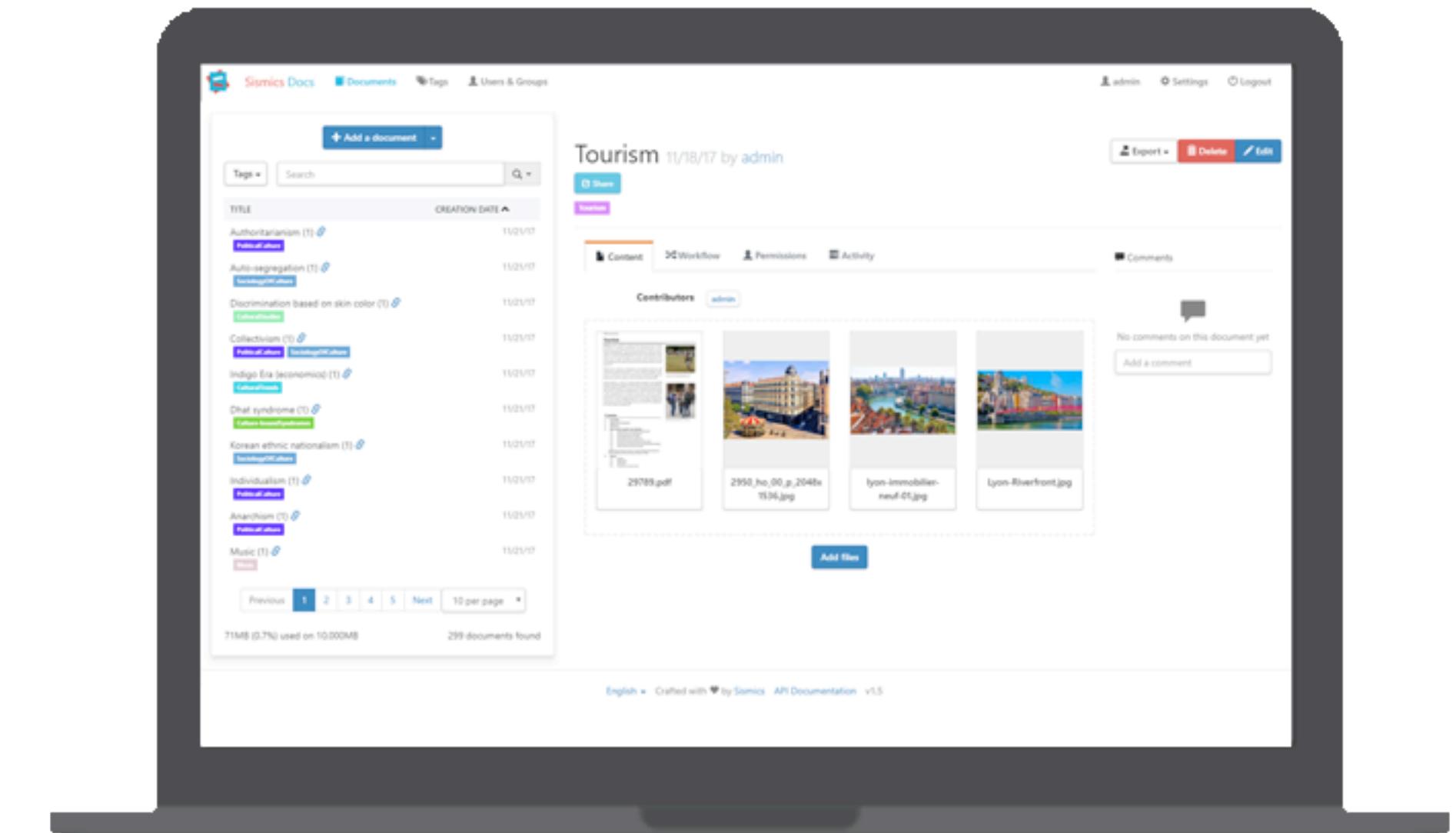
#### LEARNING GOALS

- Learn about the design and implementation of an existing software project that you will build upon this semester
- Collaborate with your teammates to design and implement a small feature addition or modification to that software project
- Practice version control and development best practices within the context of a group assignment

#### SPECIFICATION

Now that you have onboarded to 17-313, setup your development environment, and met your team, upper management has tasked you with building a graduate student admissions system over the course of the semester (described below). Rather than creating a new admissions system from scratch, management has decided that you and your team will repurpose and adapt an existing document management system, Teedy. Beyond being a pretty good document management platform, Teedy provides complex features that you expect will be useful, like tagging, workflows, and fine-grained permissions and ACLs (access control lists).

teedy



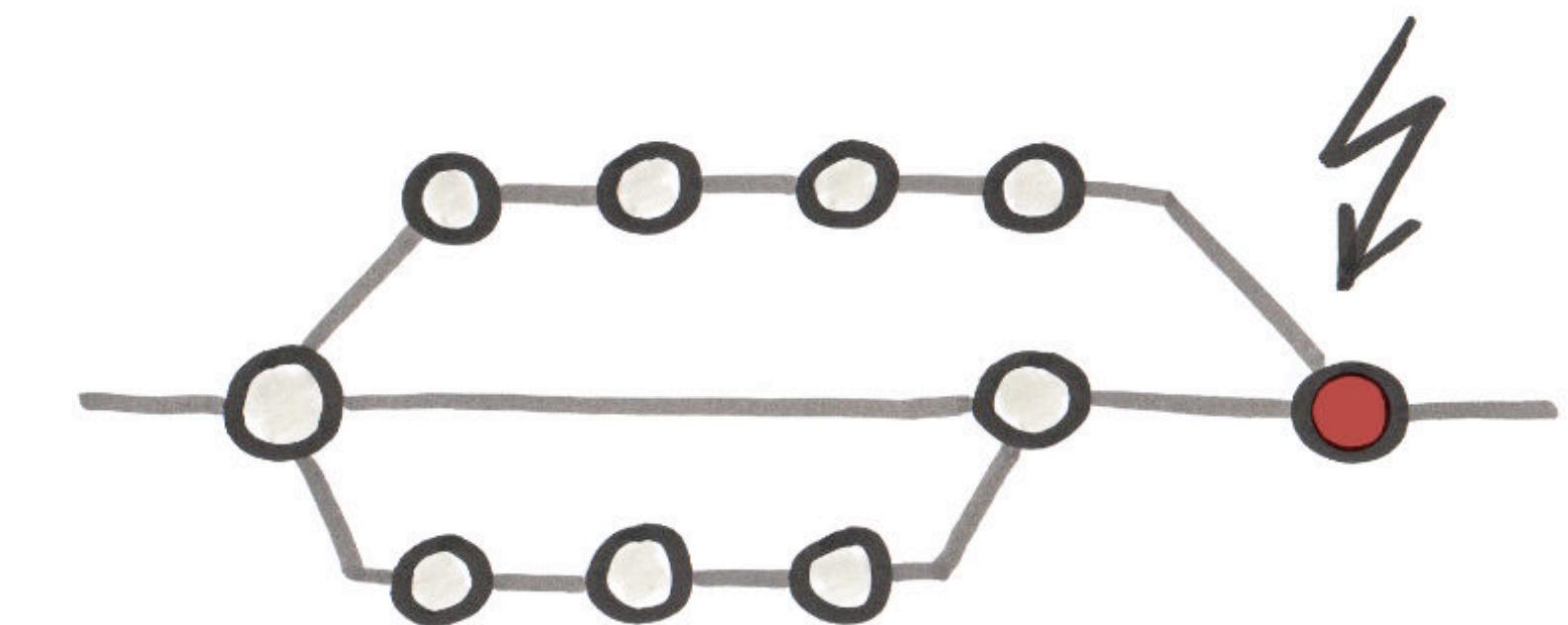
# Common Anti-Patterns

- Not having a QA process! Or no-one follows it



# Common Anti-Patterns

- Not having a QA process! Or no-one follows it
- Bad version control practices
  - Everyone commits to the main branch
  - Long-lived feature branches
  - Huge PRs



COMMENT	DATE
O CREATED MAIN LOOP & TIMING CONTROL	14 HOURS AGO
O ENABLED CONFIG FILE PARSING	9 HOURS AGO
O MISC BUGFIXES	5 HOURS AGO
O CODE ADDITIONS/EDITS	4 HOURS AGO
O MORE CODE	4 HOURS AGO
O HERE HAVE CODE	4 HOURS AGO
O AAAAAAAA	3 HOURS AGO
O ADKFJSLKDFJSOKLFJ	3 HOURS AGO
O MY HANDS ARE TYPING WORDS	2 HOURS AGO
O HAAAAAAAAANDS	2 HOURS AGO

AS A PROJECT DRAGS ON, MY GIT COMMIT MESSAGES GET LESS AND LESS INFORMATIVE.

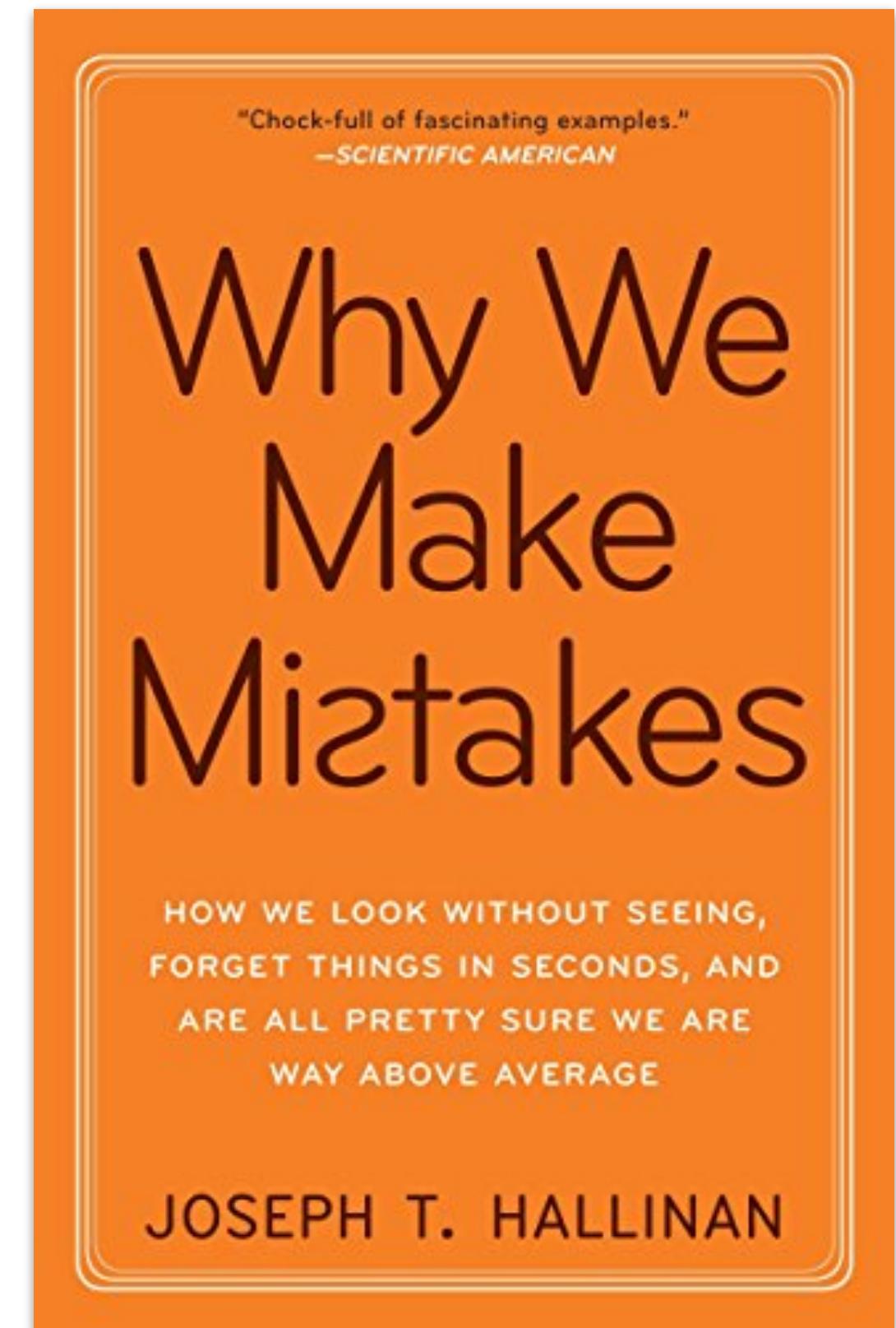
# Common Anti-Patterns

- Not having a QA process! Or no-one follows it
- Bad version control practices
- Slow and encumbering QA processes
  - changes take forever to get merged
  - time could be better spent on new features



# Common Anti-Patterns

- Not having a QA process! Or no-one follows it
- Bad version control practices
- Slow and encumbering QA processes
- Reliance on repetitive manual labor
  - focused on superficial problems rather than structural ones
  - results may vary (e.g., manual testing)
  - **mistakes will happen!**



# Case Study: Knight Capital

## Knightmare: A DevOps Cautionary Tale

• D7 • DevOps • April 17, 2014 • 6 Minutes

I was speaking at a conference last year on the topics of DevOps, Configuration as Code, and Continuous Delivery and used the following story to demonstrate the importance making deployments fully automated and repeatable as part of a DevOps/Continuous Delivery initiative. Since that conference I have been asked by several people to share the story through my blog. This story is true – this really happened. This is my telling of the story based on what I have read (I was not involved in this).

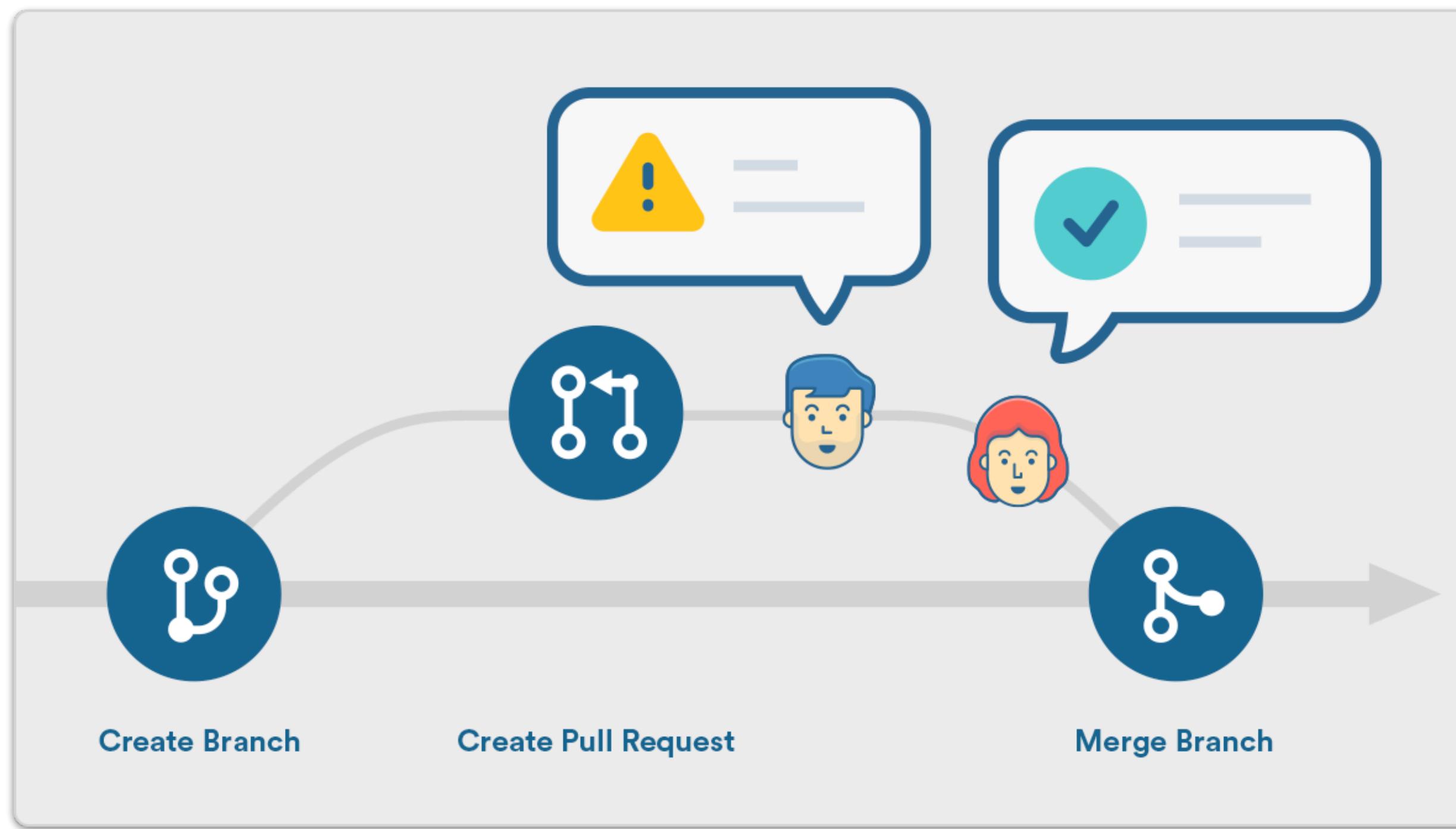


This is the story of how a company with nearly \$400 million in assets went bankrupt in 45-

In laymen's terms, Knight Capital Group realized a \$460 million loss in 45-minutes. Remember, Knight only has \$365 million in cash and equivalents. **In 45-minutes Knight went from being the largest trader in US equities and a major market maker in the NYSE and NASDAQ to bankrupt.**

# Good: Trunk-Based Development

Small changes are easier to develop, review, and integrate



# Tip: Squash and Merge

## Every commit on main should represent a meaningful change

Added coverage workflow to GitHub actions #3

The screenshot shows a GitHub pull request interface. At the top, it says "Added coverage workflow to GitHub actions #3". Below this, there's a purple "Merged" button, followed by the text "ChrisTimperley merged 24 commits into `main` from `add_coverage` 23 days ago". The main area shows a list of 24 commits from Aug 24, 2022, and one commit from Sep 3, 2022. The commits include "add coverage and run via actions", "configure multi-modules", "Autogenerated JaCoCo coverage badge", "Update README.md", "merge main changes", "Merge branch 'add\_coverage' of github.com:CMU-313/Teedy into add\_cove...", and "Autogenerated JaCoCo coverage badge". A prominent green box highlights the merge commit on Sep 3, 2022, with the message "All checks have passed" and "This branch has no conflicts with the base branch". It also features a large green "Squash and merge" button.

Merged ChrisTimperley merged 24 commits into `main` from `add_coverage` 23 days ago

Conversation 15 Commits 24 Checks 0 Files changed 4 +66 -60

Commits on Aug 24, 2022

- add coverage and run via actions
- configure multi-modules
- Autogenerated JaCoCo coverage badge
- Update README.md
- merge main changes
- Merge branch 'add\_coverage' of github.com:CMU-313/Teedy into add\_cove...
- Autogenerated JaCoCo coverage badge

Commits on Sep 3, 2022

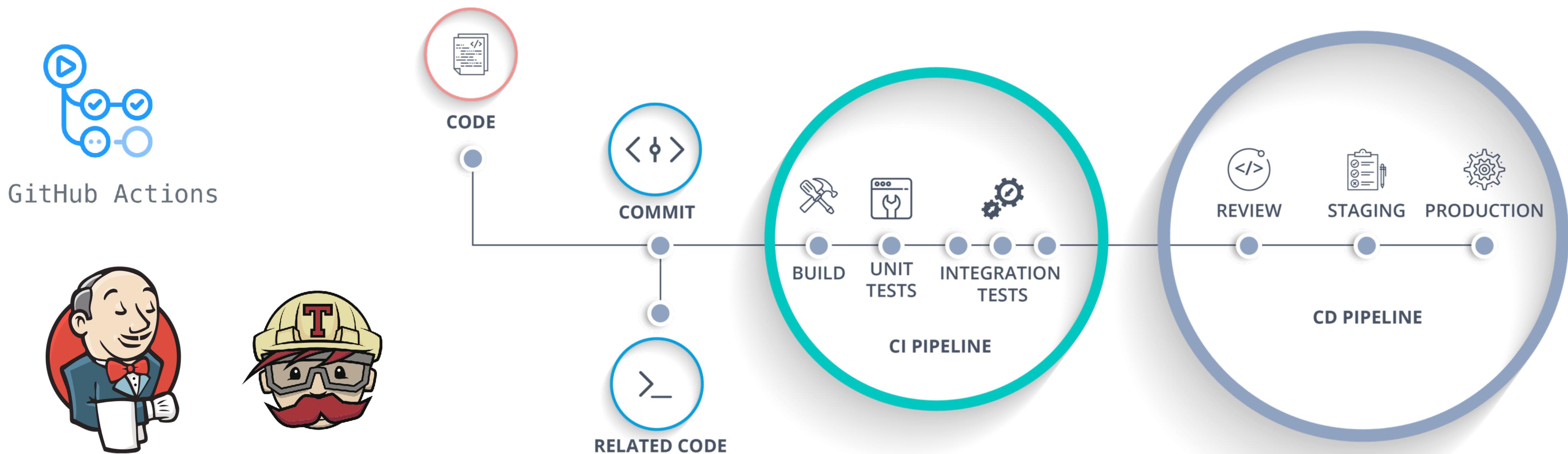
- Added coverage workflow to GitHub actions (#3)

All checks have passed  
This branch has no conflicts with the base branch

Squash and merge

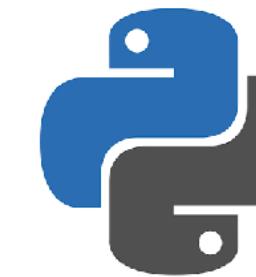
# Good: Continuous Integration and Deployment

Find and fix mistakes as early as possible when they're easier to address



# Good: Automate as much as possible!

Humans are expensive; compute time is cheap



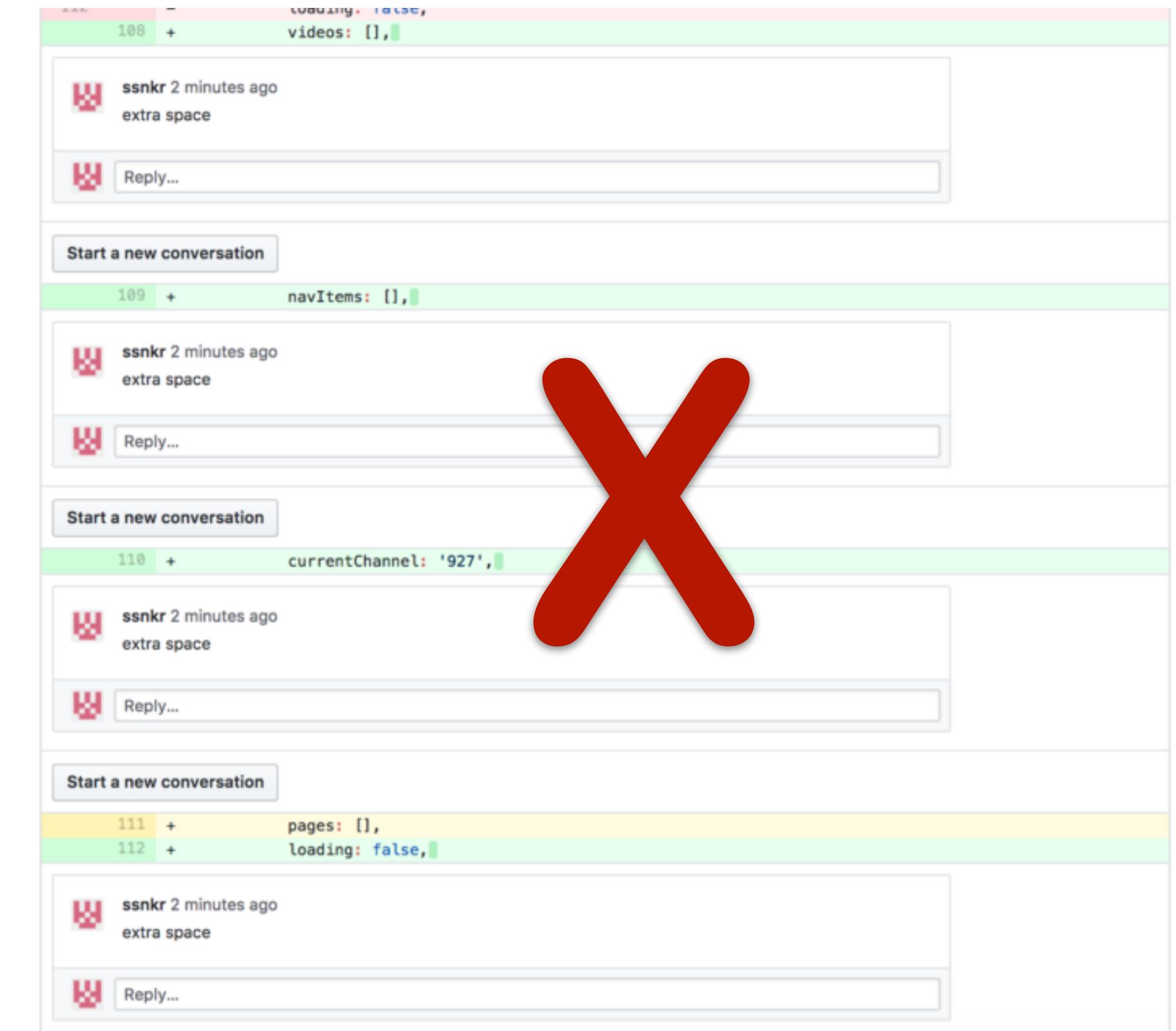
: my[py]



snyk

# Good: Let humans review the hard stuff!

- **Good:** Is this code understandable? Is this going to be a problem to maintain? Is this going to constrain us in the future? Is there a more elegant / simpler / more efficient solution?
- **Bad:** Reviewers shouldn't run the code or tests or make any changes
- **Bad:** Reviewing superficial changes



# Good: Write automated tests

Manual testing is unreliable and unsustainable

## Automated Testing

- + Reproducible
- + Some upfront effort
- + Zero marginal effort
- + Runs on every commit
- + Finds regressions!

## Manual Testing

- Unreproducible
- Low upfront effort
- High marginal effort
- Runs when you remember
- Unsustainable



# Tips for Effective CI/CD

- Your CI pipeline should run quickly



# Tips for Effective CI/CD

- Your CI pipeline should run quickly
- Your CI pipeline should run on every change
  - Only bother the reviewer once all checks pass

The screenshot shows a GitHub pull request interface. On the left, there's a green icon with a gear and a checkmark. Next to it, a summary box says "All checks have passed" with "1 successful check" and a link to "Tox lint checking / build (pull\_request) Successful in 1m — build". To the right, a vertical timeline shows two sets of commits. The first set, under "Commits on Sep 8, 2022", includes a commit by ChrisTimperley and another by ChrisTimperley. The second set, under "Commits on Sep 3, 2022", includes a commit by MichaelHilton.

- Commits on Sep 8, 2022
  - Updated docs-web config to save docs to data/docs directory (#314) Verified [4da4167](#) [🔗](#) [🔗](#)
  - added .DS\_Store to .gitignore Verified [eca3503](#) [🔗](#) [🔗](#)
- Commits on Sep 3, 2022
  - Added coverage workflow to GitHub actions (#3) Verified [e3a72ad](#) [🔗](#) [🔗](#)

# Tips for Effective CI/CD

- Your CI pipeline should run quickly
- Your CI pipeline should run on every change
- Your main branch should always be green
- Otherwise CI loses its value!

Tools: extract\_features.py: correct define name for AP\_RPM\_ENABLED



peterbarker committed 5 days ago

AP\_Mission: prevent use of uninitialised stack data 2



peterbarker committed 5 days ago

AP\_HAL\_ChibiOS: disable DMA on I2C on bdshot boards to free up DMA ch...



andyp1per authored and tridge committed 6 days ago

SITL: Fixed rounding lat/lng issue when running JSBSim SITL



ShivKhanna authored and tridge committed 6 days ago

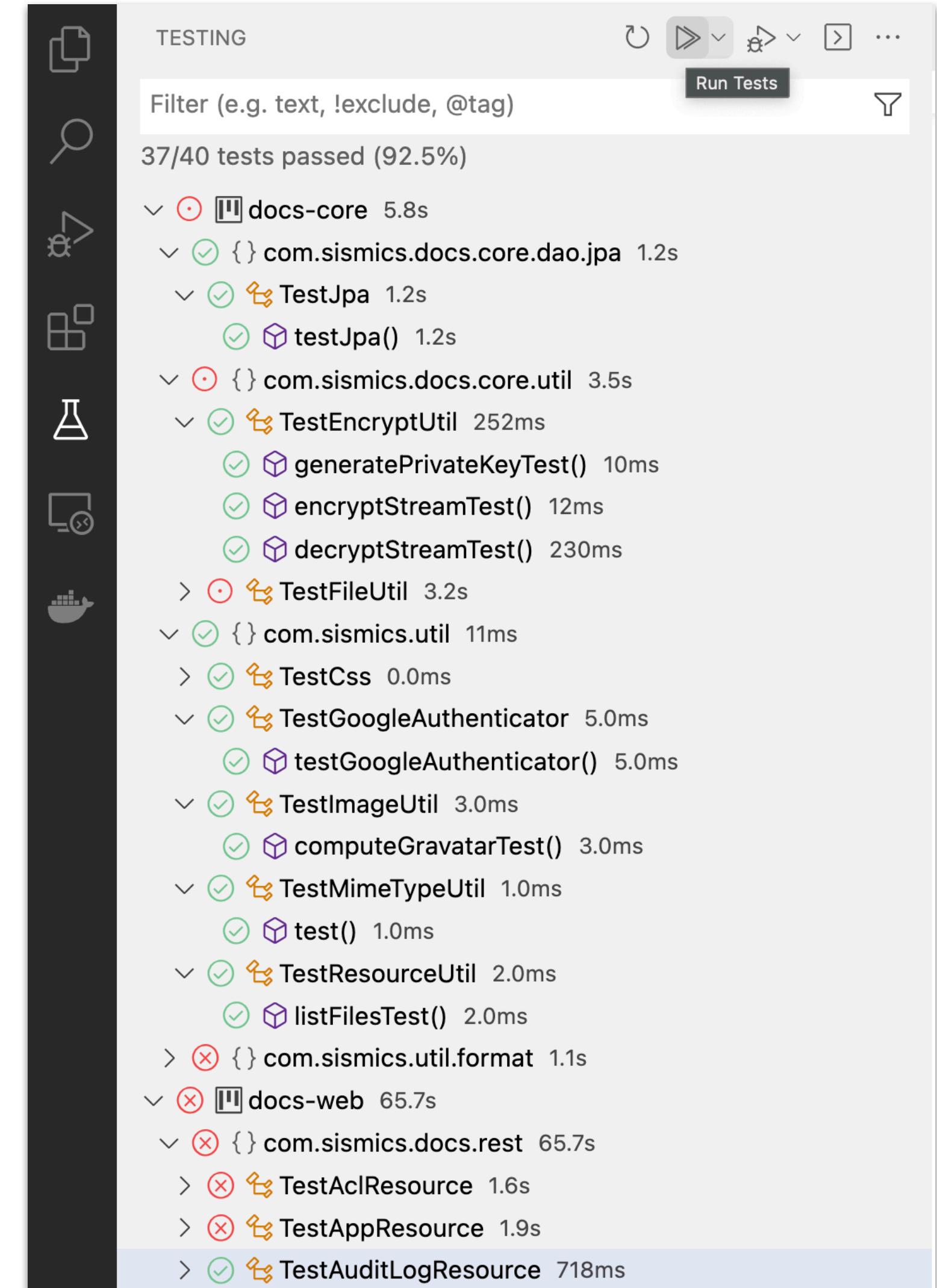
AP\_HAL\_ChibiOS: define skyviper short board names



yuri-rage authored and tridge committed 6 days ago

# Tips for Effective CI/CD

- Your CI pipeline should run quickly
- Your CI pipeline should run on every change
- Your main branch should always be green
- Your CI shouldn't replace your local environment
  - you don't have to re-run every test



# Activity: Let's design a good QA process

- Work in teams of two or three
- Upload to Gradescope

**When it all goes wrong...**



# Diagnose the problem with a checklist

- Have you recompiled the project?
- Have you pushed/pulled all of the changes?
- Have you tried destroying the database?
- Can you reproduce the problem on another machine?
- Can you find a previous version of the project that worked?
- Does the query sent by the frontend match the backend's expectations?
  - Are names spelled exactly the same? Beware of stringly types!
- Does the database contain your changes?
- ...

# Collect evidence and report the issue

## Log files, stack traces, terminal outputs, examples, screenshots

youkidearitai commented 5 hours ago

Description

The following code:

```
<?php
var_dump(mb_strpos("AあいうえおBBC", "?", 0, "ISO-2022-JP"));
var_dump(mb_stripos("AあいうえおBBC", "?", 0, "ISO-2022-JP"));
var_dump(mb_strpos("AあいうえおBBC", "?", 0, "SJIS"));
var_dump(mb_stripos("AあいうえおBBC", "?", 0, "SJIS"));
```

Resulted in this output:

```
int(1)
int(5)
int(1)
int(5)
```

But I expected this output instead:

```
bool(false)
bool(false)
bool(false)
bool(false)
```

note: mb\_stripos is affect mb\_substitute\_character, but mb\_strpos is not affect it is (3v4l: <https://3v4l.org/872nB>)

PHP Version

PHP 8.1.x

Operating System

Ubuntu 20.04 on WSL2

The screenshot shows the Apache Log4j Chainsaw interface. The main window displays log entries from the 'chainsaw' logger at level INFO and WARN. A search bar at the top right shows the query 'msg ~= 'Connect to' || msg ~= securitymanager'. The log entries include:

ID	TIMESTAMP	MARKER	LEVEL	LOGGER	MESSAGE
1	12:07:07		INFO	LogUI	SecurityManager is now: null
2	12:07:07		WARN	help.HelpManager	Could not find any local JavaDocs, you might want to consider running 'ant javadoc'. The re... website.
8	12:07:07		ERROR	receivers.ReceiversHelper	Failed to locate Receiver class:org.apache.log4j.net.JMSReceiver, looks like a dependent cla... OSIntegration.init() called
14	12:07:08		INFO	osx.OSXIntegration	OSXIntegration.init() called
15	12:07:08		INFO	messages.MessageCenter	Zeroconf started!
16	12:07:08		WARN	zeroconf.ZeroConfPlugin	menu 'Connect to' was NOT added because the 'Help' menu could not be located
17	12:07:08	Zeroconf available - supports log4j2 via Advertiser	INFO	messages.MessageCenter	Looks like ZeroConf is available... WooHoo!
54	12:07:47		WARN	LoggerNameTreePanel	Should warn user, depth=6
65	12:10:20		INFO	LogPanel	clearTableExpressionRule invalid - ignoring:

The bottom right shows an 'Analyze Stacktrace' dialog with a stack trace:

```
W: java.lang.Throwable: stack dump
W:     at java.lang.Thread.dumpStack(Thread.java:490)
W:     at com.google.samples.apps.topeka.activity.SignInActivity.onCreate(SignInActivity.j...
W:     at android.app.Activity.performCreate(Activity.java:6237)
W:     at android.app.Instrumentation.callActivityOnCreate(Instrumentation.java:1107)
W:     at android.app.ActivityThread.performLaunchActivity(ActivityThread.java:2369)
W:     at android.app.ActivityThread.handleLaunchActivity(ActivityThread.java:2476)
W:     at android.app.ActivityThread.-wrap11(ActivityThread.java)
W:     at android.app.ActivityThread$H.handleMessage(ActivityThread.java:1344)
W:     at android.os.Handler.dispatchMessage(Handler.java:102)
W:     at android.os.Looper.loop(Looper.java:148)
W:     at android.app.ActivityThread.main(ActivityThread.java:5417)
W:     at java.lang.reflect.Method.invoke(Method)
W:     at com.android.internal.os.ZygoteInit$MethodAndArgsCaller.run(ZygoteInit.java:726)
W:     at com.android.internal.os.ZygoteInit.main(ZygoteInit.java:616)
```

Buttons in the dialog include 'OK', 'Normalize', and 'Cancel'.

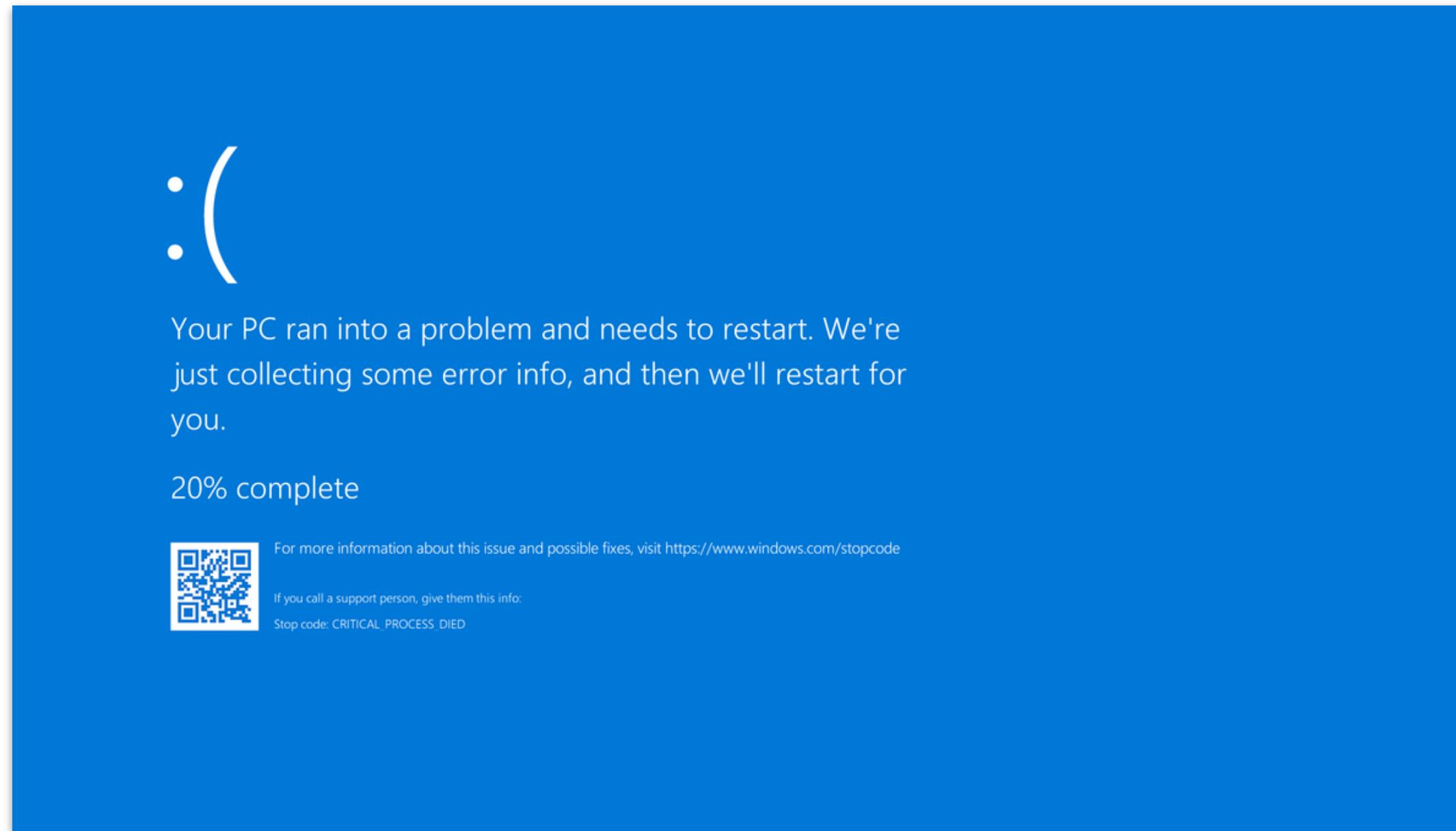
# Write a test to reliably reproduce the failure

## Now your teammates can help out, too!

```
/**  
 * Test related to https://github.com/sismics/docs/issues/373.  
 */  
  
@Test  
public void testIssue373() throws Exception {  
    PdfFormatHandler formatHandler = new PdfFormatHandler();  
    String content = formatHandler.extractContent(  
        language: "deu",  
        Paths.get(ClassLoader.getSystemResource("file/issue373.pdf").toURI())  
    );  
    Assert.assertTrue(content.contains("Aufrechterhaltung"));  
    Assert.assertTrue(content.contains("Außentemperatur"));  
    Assert.assertTrue(content.contains("Grundumsatzmessungen"));  
    Assert.assertTrue(content.contains("ermitteln"));  
}
```

```
1 + --TEST--  
2 + Bug #75917  SplFileObject::seek broken with CSV flags  
3 + --FILE--  
4 + <?php  
5 + $expected = [  
6 +     ['john', 'doe', 'john.doe@example.com', '0123456789'],  
7 +     ['jane', 'doe', 'jane.doe@example.com'],  
8 + ];  
9 +  
10 + $tmp = new SplTempFileObject();  
11 + foreach ($expected as $row) {  
12 +     $tmp->fputcsv($row);  
13 + }  
14 + $tmp->setFlags(0);  
15 + $tmp->seek(23);  
16 + var_dump($tmp->current());  
17 +  
18 + $tmp->setFlags(SplFileObject::READ_CSV |  
+     SplFileObject::SKIP_EMPTY);  
19 + $tmp->seek(23);  
20 + var_dump($tmp->current());  
21 + ?>  
22 + --EXPECT--  
23 + bool(false)  
24 + bool(false)
```

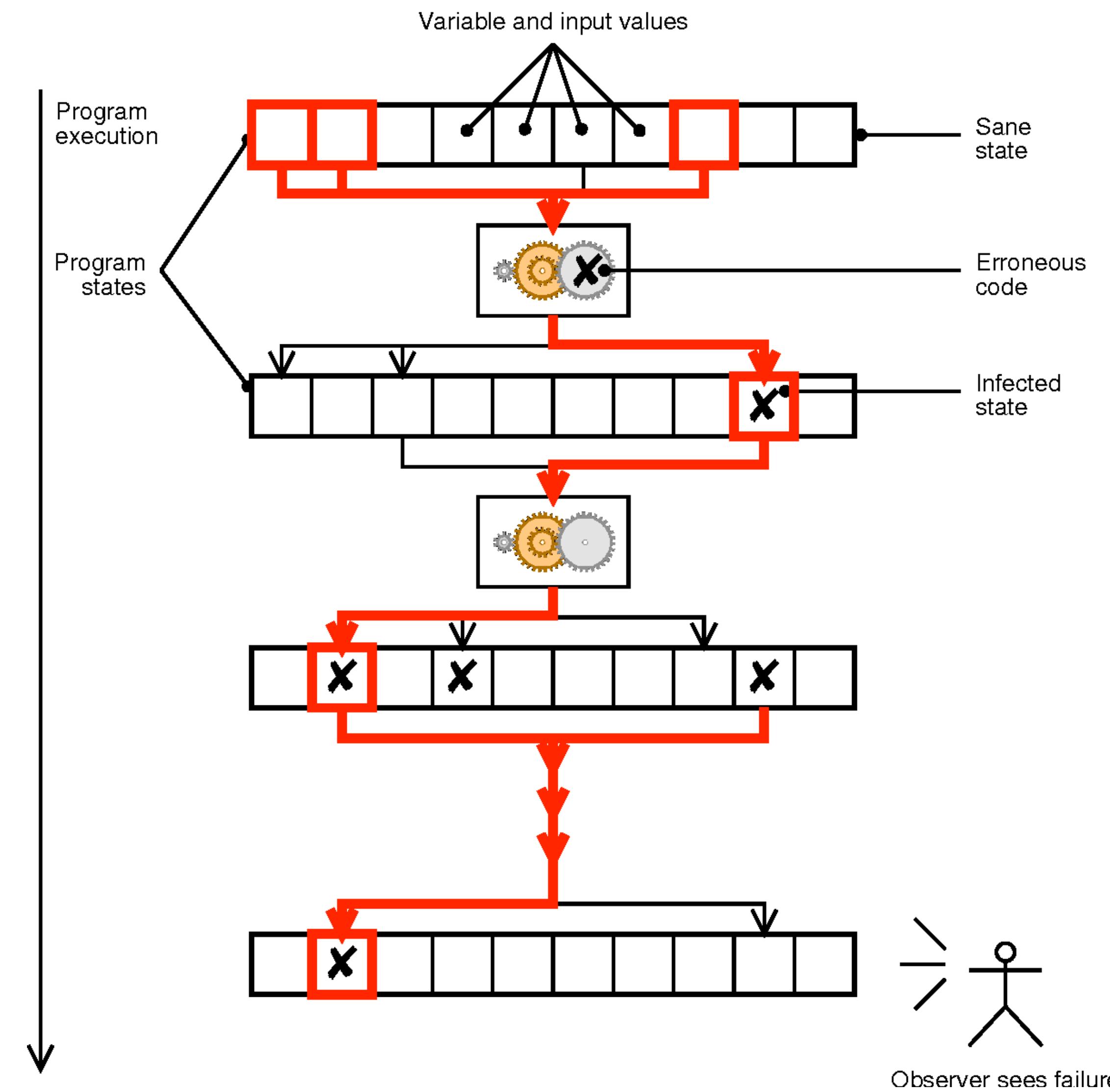
# Problem: What code caused the bug?



# Starting Point: Use the Stack Trace

```
96         // Add a file
97     String file1Id = clientUtil.addFileToDocument(FILE_EINSTEIN_ROOSEVELT LETTER_PNG, javax.ws.rs.InternalServerErrorException: HTTP 500 Internal Error at org.glassfish.jersey.client.JerseyInvocation$Builder.put(JerseyInvocation.java:329)
at com.sismics.docs.rest.util.ClientUtil.addFileToDocument(ClientUtil.java:212)
at com.sismics.docs.rest.TestDocumentResource.testDocumentResource(TestDocumentResource.java:97)
```

# Defects cause program state to be infected



# Locate the defect: Git History

Defects tend to be located in recently changed files

Raw   Blame  

Teedy / docs-web / src / main / java / com / sismics / docs / rest / resource /  
BaseResource.java

Newer Older

100644 | 124 lines (110 sloc) | 3.3 KB

Raw   Normal view   History

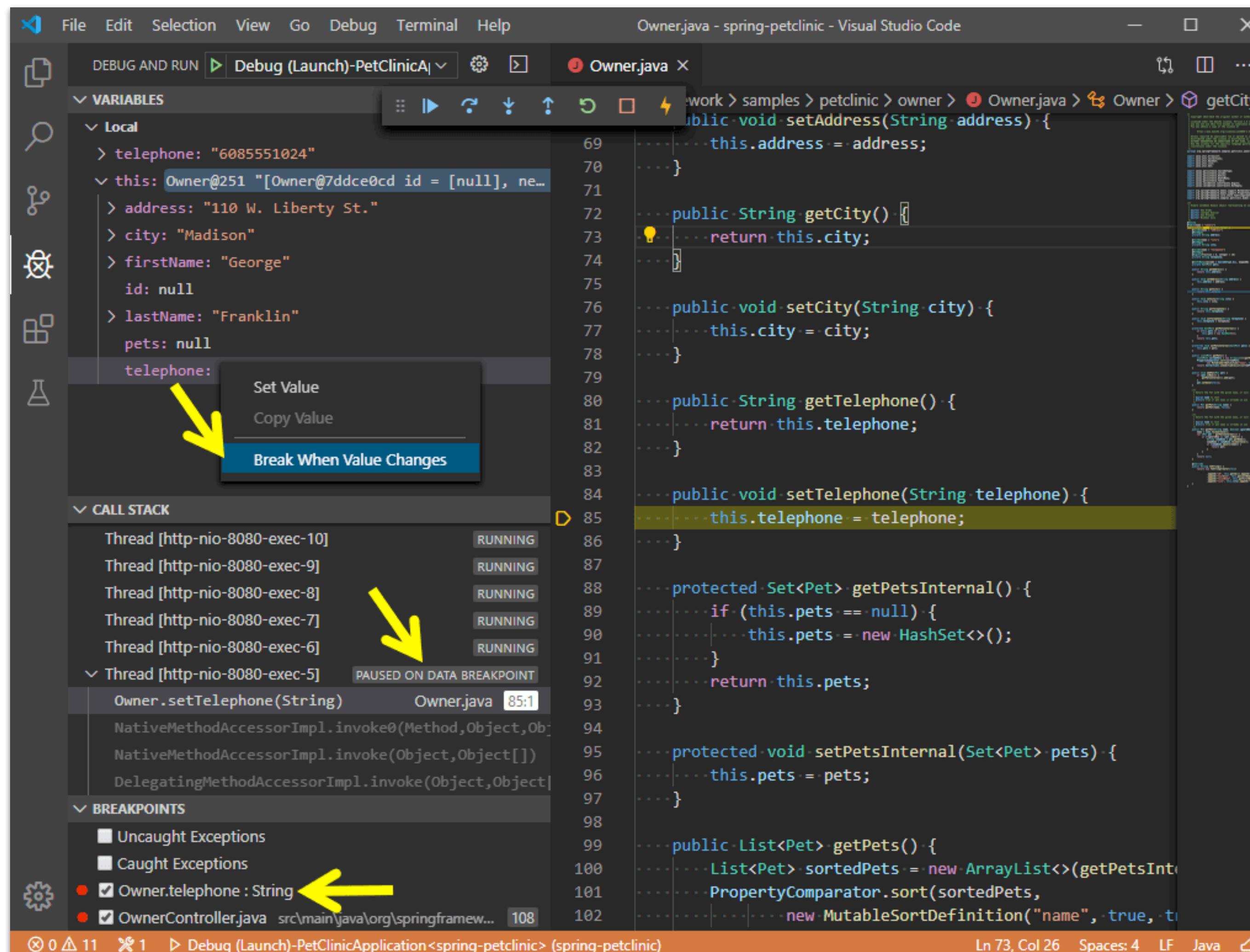
Initial commit	9 years ago	1 package com.sismics.docs.rest.resource; 2
#18: ACL check for groups	7 years ago	3 import com.google.common.collect.Lists;
Initial commit	9 years ago	4 import com.sismics.docs.rest.constant.BaseFunction; 5 import com.sismics.rest.exception.ForbiddenClientException; 6 import com.sismics.security.IPrincipal; 7 import com.sismics.security.UserPrincipal;
Closes #106: Header base authentication	6 years ago	8 import com.sismics.util.filter.SecurityFilter; 9 10 import javax.servlet.http.HttpServletRequest;
upgrade to java 11 + upgrade libraries	2 years ago	11 import javax.ws.rs.Consumes;
Download zip of files not in same document (#591)	5 months ago	12 import javax.ws.rs.Produces;
Closes #106: Header base authentication	6 years ago	13 import javax.ws.rs.QueryParam; 14 import javax.ws.rs.core.Context;
upgrade to java 11 + upgrade libraries	2 years ago	15 import javax.ws.rs.core.MediaType;
Closes #106: Header base authentication	6 years ago	16 import java.security.Principal; 17 import java.util.List; 18 import java.util.Set;

main Teedy / docs-web / src / main / java / com / sismics / docs / rest / resource / Go to file Add file ...

jendib keep filename in temporary file 5e7f060 on May 16

AclResource.java	Closes #313: remove administrators from ACL targets search	3 years ago
AppResource.java	Closes #632: validate POST /app/config_inbox and update documentation	5 months ago
AuditLogResource.java	Closes #257: admin users can see all logs	4 years ago
BaseResource.java	Download zip of files not in same document (#591)	5 months ago
CommentResource.java	Refactor documents and files indexing	5 years ago
DocsMessageBodyWriter.java	Download zip of files not in same document (#591)	5 months ago
DocumentResource.java	Add doc for search syntax (#634)	5 months ago
FileResource.java	keep filename in temporary file	4 months ago
GroupResource.java	Allow the . (dot) and @ (at) character in usernames (#637)	5 months ago
MetadataResource.java	#300: custom metadata fields: UI read/write	3 years ago
RouteModelResource.java	Closes #252: route model permissions	4 years ago
RouteResource.java	Fixed sending workflow emails to previous assignee (#422)	2 years ago
ShareResource.java	Closes #509: guest users cannot share and unshare	2 years ago
TagResource.java	Closes #379: spaces and colons not allowed in tag name	3 years ago
ThemeResource.java	sismics docs -> teedy	4 years ago
UserResource.java	rename	5 months ago
VocabularyResource.java	#300: custom metadata fields: API admin	3 years ago
WebhookResource.java	Closes #243: webhooks UI	4 years ago

# Locate the defect: Interactive Debugging



See: <https://github.com/CMU-313/Teedy/wiki/Using-the-Debugger-via-VSCode>

# Locate the defect: Assertions

## Reduce the distance between the fault and failure

```
public static int findMax(int v[ ]) {  
    int max = Integer.MIN_VALUE;  
  
    // Precondition: v[ ] is not empty  
    assert v.length > 0 : "v[ ] is empty";  
    // Precondition: max <= v[i] for every i  
    for (int i = 0; i < v.length; i++)  
        assert max <= v[i] : "Found value < MIN_VALUE";  
  
    // Locate the real maximum value  
    for (int i = 0; i < v.length; i++)  
        if (v[i] > max)  
            max = v[i];  
  
    // Postcondition: max >= v[i] for every i  
    for (int i = 0; i < v.length; i++)  
        assert max >= v[i] : "Found value > MIN_VALUE";  
    return max;  
}
```

Detect illegal program states

# Write smaller unit tests as failure hypotheses

## Go from large integration tests to small unit tests

```
/**  
 * Exhaustive test of the user resource.  
 *  
 * @author jtremeaux  
 */  
  
public class TestUserResource extends BaseJerseyTest {  
    /**  
     * Test the user resource.  
     */  
    @Test  
    public void testUserResource() {  
        // Check anonymous user information  
        JsonObject json = target().path(path: "/user").request()  
            .acceptLanguage(Locale.US)  
            .get(responseType: JsonObject.class);  
        Assert.assertTrue(json.getBoolean("is_default_password"));  
  
        // Create alice user  
        clientUtil.createUser(username: "alice");  
  
        // Login admin  
        String adminToken = clientUtil.login(username: "admin", password: "admin", remember: false);  
    }  
}
```



```
/**  
 * Test of {@link PdfFormatHandler}  
 *  
 * @author bgamard  
 */  
  
public class TestPdfFormatHandler {  
    /**  
     * Test related to https://github.com/sismics/docs/issues/373.  
     */  
    @Test  
    public void testIssue373() throws Exception {  
        PdfFormatHandler formatHandler = new PdfFormatHandler();  
        String content = formatHandler.extractContent(language: "deu", Paths.get("src/test/resources/issue373.pdf"));  
        Assert.assertTrue(content.contains("Aufrechterhaltung"));  
        Assert.assertTrue(content.contains("Außentemperatur"));  
        Assert.assertTrue(content.contains("Grundumsatzmessungen"));  
        Assert.assertTrue(content.contains("ermitteln"));  
    }  
}
```

# Write a patch and use your tests to validate

## If your tests pass, you've fixed the bug \*

[3.6] bpo-29104: Fixed parsing backslashes in f-strings. (GH-490) #1812

Merged serhiy-storchaka merged 1 commit into python:3.6 from serhiy-storchaka:backport-0cd7a3f-3.6 on May 25, 2017

The screenshot shows a GitHub pull request interface. At the top, it says "[3.6] bpo-29104: Fixed parsing backslashes in f-strings. (GH-490) #1812". Below that, a purple button indicates the commit is "Merged". The main area displays a diff between the Python 3.6 branch and a backport branch. The diff shows changes in the file `Lib/test/test_fstring.py`. The changes are color-coded: red for deleted code and green for added code. The code itself is written in Python, with many lines containing assertions like `self.assertEqual` and `self.assertWarns`.

```
diff --git a/Lib/test/test_fstring.py b/Lib/test/test_fstring.py
@@ -361,6 +361,20 @@ def test_backslashes_in_string_part(self):
    self.assertEqual(f'2\x203', '2 3')
    self.assertEqual(f'\x203', ' 3')
    self.assertEqual(f'\\x203', '\\x203')
+   with self.assertWarns(DeprecationWarning): # invalid
+       value = eval(r"f'\{6*7}'")
+   self.assertEqual(value, '\u42')
+   self.assertEqual(f'\\{6*7}', '\u42')
+   self.assertEqual(fr'\{6*7}', '\u42')
+
+   AMPERSAND = 'spam'
+   # Get the right unicode character (&), or pick up
+   # local variable
+   # depending on the number of backslashes.
+   self.assertEqual(f'\N{AMPERSAND}', '&')
+   self.assertEqual(f'\\N{AMPERSAND}', '\u0042')
+   self.assertEqual(fr'\N{AMPERSAND}', '\u0042')
+   self.assertEqual(f'\\\\N{AMPERSAND}', '\\&')
+
364     def test_misformed_unicode_character_name(self):
365         # These test are needed because unicode names are
366         # differently inside f-strings.
367
368     def test_misformed_unicode_character_name(self):
369         # These test are needed because unicode names are
370         # differently inside f-strings.
```

# Takeaways

- Technical debt is a fact of life
- It's easier to preemptively avoid issues and maintain quality than to accumulate technical debt and address it later
- But you can't avoid every issue! We can use techniques to diagnose, debug, and fix those issues that do occur