

HOW TO DIAGNOSE (ALMOST) ANYTHING

Modularity 

Correctness 

Real-world engineering 

CS-214 W02 SE – 2025-09-17

Clément Pit-Claudé

Quick announcements

degrees

... was due last Friday!

Don't var

(won't work on the midterm)

First exam lab

Tonight, in BC07/08!

Training and exam VMs

Now available!

First debrief

Out on Ed!

New help sessions

Mondays 1–3 PM

The plan for CS-214.SE

- Make you (better) software engineers
- Show you how to figure things out

Versioning and collaboration 

3 lectures

Writing correct code 

4 lectures

Building apps from scratch 

3 lectures

+ unguided lab

Refactoring & code evolution 

Callbacks

EXERCISE

Think of a bug that frustrated you in a recent lab.
Share with your neighbor.

Today

Debugging:
Principles + case studies

- Identify and describe **unwanted behaviors**
- Produce useful and complete **defect reports**
- Pinpoint the **root cause** of an issue in a software system

THE 2025 CS-214 GUIDE TO DEBUGGING: ON ONE SLIDE

Process

Triage

1. Check that there is a problem
2. Reproduce the issue
3. Decide whether it's your problem
4. Write it up

Diagnose and fix

1. Learn about the system
2. Simplify, minimize, and isolate
3. Observe the defect
4. Guess and verify
5. Fix and confirm the fix
6. Prevent regressions

Techniques

- Keep notes
- Change one thing at a time
- Apply the scientific method
- Instrument
- Divide and conquer
- Ask for help

Pitfalls

- Random mutation
- Staring aimlessly
- Wasting time
- Assuming a bug went away
- Fixing effects, not causes
- Losing data

A CASE STUDY



EC 44 M2
Genève-Aéroport
Morges Nyon (

Step 1 (of 2)

Triage

Check

that there is a problem

Reproduce

the issue

Decide

whether it's your problem

Write up

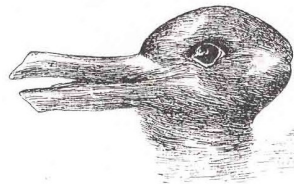
a detailed bug report

TRIAGE 1 & 2: CONFIRM AND REPRODUCE

1. Confirm that there is a problem

Know what you're looking for

- State what the issue is.
(May be refined later)
→ *“Coursier exits silently without installing Scala”*
- Compare to the spec, the documentation, the requirements.
→ *“find -type f should not return directories”*
- Is it obvious why the bad case is different from a good case?
→ *“System will identify animal in picture”*



2. Reproduce the issue

Gather information

- Does it happen every time?
→ *“Only on Tuesdays”*
- Does it happen for every input?
→ *“Only in test 3”*
→ *“Only when clicking repeatedly”*
- Does it depend on system config?
→ *“Only with SBT version XYZ”*
- Are there any diagnostics?
→ Error messages, logs, dialogs
- Did it work at one point?
→ Previous versions, commit history

TRIAGE 3 & 4: DECIDE AND WRITE UP

3. Decide whether it's your pb

Not all bugs are worth fixing

- Do I need to fix it?
 - Closed-source SW, lack of expertise/interest
- Is it worth fixing?
 - Workaround may be sufficient
- Does it need to be fixed *now*?
 - Lab must be released to students at 4PM
- Do I know where to complain?
 - May require diagnosis

4. Write it up

Don't waste your work

- Check previous reports
 - Browse bugs DB
 - Check known issues
 - **Browse previous questions!**
- Find where to report
 - Email, bug tracker, contact form
- Check reporting guidelines
 - Is there a security policy?
- Write clearly and completely
 - Pick a good title
 - Include steps taken, results observed, expectations
 - Include system details, any customization

Number: CS006551005

Date: 04.10.2023 15:50:10 CEST

Description: Incorrect display on train to Geneva in Lausanne:
"Genève-Aéroport" as "Genwve-Aщroport"

Hi,

My name is Clément, I work at EPFL.

On September 3rd, in Lausanne, at 11:12 PM, I noticed an incorrect display on my train door.

It appears that "Genève" and "Aéroport" were not displayed correctly:
"Genève-Aéroport" was displayed as "Genwve-Aщroport".

I have attached a picture.

Some speculation: it could be that the text "Genève-Aéroport" was encoded using Latin-1, and decoded using ISO-8859-5 or one of its variants.

Thank you for your time,
Clément Pit-Claudé.

HOW TO ASK A CS-214 ED QUESTION

What does 'A pure expression does nothing in statement position' mean?



Anonymous

5 hours ago in Lab 2



PIN



STAR



WATCH

VIEWS



Hi,

When I replace the contents of `playground.worksheet.sc` with the following code, VSCode underlines the first 0 in red. The corresponding error message is *"A pure expression does nothing in statement position; you may be omitting necessary parentheses"*

```
def f(): (Int) =  
  if true then 0  
  else if true  
    val r = f()  
    (r)  
  else 0
```

What does the error message mean, and how can I fix my code?

Comment Edit Delete Endorse ...

- Helpful title
- Good punctuation
- Clear problem
- Has expected behavior
- Has reproduction steps
- Properly formatted code
- Relevant info present

ED POST CHECKLIST

☑ Is the title clear?

- ✗ *“Lab 2”* → ✓ *“Broken link to lab 2 on Moodle”*
- ✗ *“Git”* → ✓ *“Git error: ‘corrupt loose object’”*
- ✗ *“infinite loop”* → ✓ *“Infinite loop in minMax”*

☑ Is the description precise?

- ✗ *“It doesn’t work”* → ✓ *“When given input x, it produces value y [...]”*
- ✗ *“It’s weird”* → ✓ *“[...] it produces value y, which is unexpected because [...]”*
- ✗ *“In the lecture [...]”* → ✓ *“On slide n of the lecture on x, [...]”*

☑ Is the description usable?

- ✗ Code screenshot → ✓ Code block
- ✗ Code snippet → ✓ Self-contained example

☑ Are attempts at a solution documented?

- ✗ *“I tried everything”* → ✓ *“Here are the things I tried: [...]”*

Step 2 (of 2)

Diagnose & fix

Learn

about the system

Observe

the defect

Simplify

and minimize

Guess

and verify

Fix

and confirm the fix

Prevent

regressions

DIAGNOSIS 1&2: LEARN AND OBSERVE

1. Learn about the system

Know what to expect

- Consult the docs/manual
 - Read the lab write-up in detail
- Search for relevant resources
 - Known issues, error guides
 - Documentation on error messages
- Know what you don't know
 - *"What happens if %PATH% contains special characters?"*
 - *"Does 1 until n include n?"*
 - *"I'm not sure how VSCode starts the JVM to run Metals"*
- Know the relevant tools
 - JTAG, perf, strace, objdump, a multimeter, ...
- Skim through the code
 - Find entry point, seemingly relevant functions

2. Observe the issue

Learn more about the problem

- Exercise different angles
 - Vary the inputs
 - Look upstream and downstream (consequences)
- Read error messages
- Add logging / tracing
 - Print relevant variables and rerun
 - Turn on errors and warnings
- Use the right tools
 - But don't get distracted
- Use the VCS history
 - Especially for regressions
- Read the code

DIAGNOSIS 3&4: SIMPLIFY AND GUESS

3. Simplify and minimize the issue

Find the relevant subsystem

- Simplify the inputs
 - Shorten the test case
 - Find a short and simple input
- Simplify the system
 - Remove unnecessary code
 - Reproduce the problem in isolation
 - Check a different machine/config
- Slow things down
 - Introduce delays or pauses
- Determine the failure
 - Set random seeds
 - Stop other processes
- Automate the failure
 - Have tests not just a user story
 - Use simplest possible tests (unit tests if possible)
 - Run tests on CLI, not in UI with clicks

4. Guess and verify

Be a scientist!

- Formulate a hypothesis
 - *“I forgot to clamp the speed”*
 - *“The acceleration may be applied twice”*
 - *“Hard drive delays are causing our latency issues”*
 - *“Values get corrupted in step 5”*
- Design an experiment
 - Transform or observe the system to test your hypothesis.
 - Add logging, breakpoints, assertions, prints,
- Narrow down issue
 - Divide bug into smaller bugs using informed guesses and hypothesis testing
 - Move failing tests up until they pass
 - Look for the root cause

A SIMPLE MODEL: *decode* ◦ *encode* ≠ *id*?

```
import java.nio.charset.Character
```

```
val good = "Genève-Aéroport"
```

```
val bad  = "Genшve-Aщроport"
```

```
@main def findPairs =
```

```
  Charset.availableCharsets().forEach((name1, c1) =>
```

```
    Charset.availableCharsets().forEach((name2, c2) =>
```

```
      if c1.canEncode then
```

```
        val roundtrip = c2.decode(c1.encode(good))
```

```
        if roundtrip.toString == bad then
```

```
          println(f"$name1 → $name2"))
```

DIAGNOSIS 5 & 6: FIX AND PREVENT

5. Fix and confirm the fix

Change relevant system

- Decide whether to fix the problem
 - Is the problem easily fixable?
 - Is a workaround preferable?
 - Revisit “*Do I have to fix it?*”
- Apply the changes
- Revert other changes
 - Use your VCS to undo unrelated changes
 - Confirm the fix on a clean system
- Confirm the fix
 - Re-run all tests
 - Confirm that other behavior isn’t affected

6. Prevent regressions

Protect future developers

- Document the resolution
 - Write a detailed commit message
 - Update the report to document the root cause, the fix or workaround, and any required follow-ups
- Look for similar instances
 - Fix all bugs in that family, not just one
- Add missing tests
 - Prevent future regressions

TRIAGE & DIAGNOSIS TECHNIQUES (DO THIS...)

✓ Keep notes

- Write every change down:
“Ran cs_setup.exe from cmd in C:\”
- Write your observations
“The checksum is correct after the download”
- Work on a Git branch
- Keep decision trees, guesses, mind maps

✓ Change one thing at a time

- Needed to establish cause / effect relationships
Avoid: *“I added a lock and a cooldown period and an automated retry and a generic try-catch block and the error is gone”*

✓ Apply the scientific method

- Write down a conjecture (potential cause)
“The function is not exploring all directories”
- Design an experiment
Add `println` to every call to find
- Reject or accept
“All directories are printed” ⇒ reject

✓ Instrument

- Add `println` calls
- Write to logs

✓ Divide and conquer

- Design tests to pinpoint the source
“Bug happens in UI but not on CLI”
- Inspect intermediate states
“System is stable until cohesion force kicks in”
- Test components in isolation
Move from integration tests to unit tests

✓ Ask for help

- Find a system expert
- Write up problem online
- Look for troubleshooting guides
- Post on Ed

PITFALLS (... DON'T DO THIS)

X Random mutation

→ Don't change things "until it works"

X Staring aimlessly

→ "I spent 8 hours looking at this code"

X Guessing without verifying

→ Fixing a bug takes time. Don't fix correct code!

→ Guessing well takes practice.

X Wasting time

→ Experiments are not all equally easy

→ Look out for signs of going down a rabbit hole

→ Rewriting from scratch is costly and ineffective

→ Sometimes a quick check can eliminate many possible hypotheses

→ Reinstalling the OS is a last resort

X Assuming bug went away

→ Bugs don't disappear

→ ... and they reappear at the worst time

X Focusing on the wrong thing

→ "I just added cohesion and it fails, so the error must be there"

X Fixing effects, not causes

→ "I will just multiply all speeds by two"

→ "I just need to reinstall SBT every time I log in"

→ "I rewrote all the code and now it works"

X Losing data

→ Take a disk image

→ Save the buggy code somewhere

→ Keep the logs / core dumps / ...

CONCLUSION:

USE THE CS-214 DEBUGGING GUIDE!

After fixing a bug:

- Go through this guide
- Reflect on what went right and wrong

When tackling a new bug:

- Pull up this guide
- Follow the steps in order

When asking a question on Ed:

- Pick a helpful title
- Describe which steps you took and where you got stuck, in detail

SUGGESTED READING

- [How do I ask a good question? - Help Center - Stack Overflow](#)
- [Bug Reporting Guide | Contributing to Scala's OSS Ecosystem](#)
- [Understanding Bug Reporting \(GNU Emacs Manual\)](#)
- [Checklist for bug reports \(GNU Emacs Manual\)](#)
- [Submitting Bugs and Suggestions · microsoft/vscode Wiki · GitHub](#)
- *Debugging: The 9 Indispensable Rules for Finding Even the Most Elusive Software and Hardware Problems (David J. Agans)*
- [How to Debug – Embedded in Academia](#)

From: SBB Customer Service <customerservice@sbb.ch>
To: clement.pit-claudel@epfl.ch
Subject: AW: CS006551005 - SBB - Customer information on the display
Date: Mon, 09 Oct 2023 05:58:03 -0700

Sehr geehrte Damen und Herren

* Der Fehler lag hier vermutlich im «KIS» System, dem Software System welche die Ansage und die Beschriftung koordiniert.

mit freundlichen Grüßen