Bug Log – Bug 1

# Hypothesis 1 – What I’m Checking

The error is in the user interface somewhere (not sure what I’m checking for yet) – Wrong Book ID Scanned/passed through by UI

# Test 1 – How I’m Checking

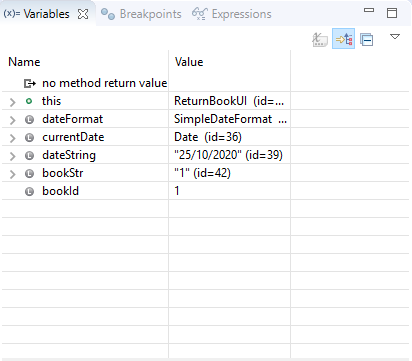
# Checking bookID & bookStr match 1

# Result 1 – Was I correct?

# False - bookId and bookStr were both 1

# Conclusion 1 – Further Comments On/Based On Result

The UI is not the issue



# Hypothesis 2

# The issue is in the bookScanned method – the bookID results in the wrong book being fetched from the library to go into currentLoan.

# Test 2

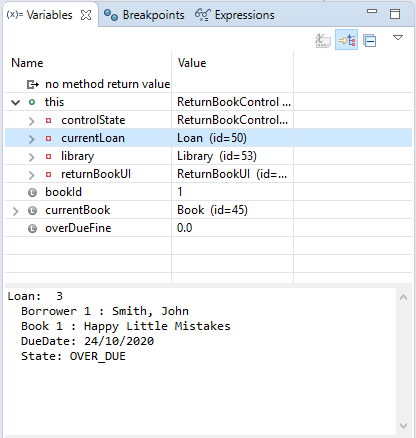
# Check bookID in bookScanned + currentLoan

# Result 2

# False - bookID is correct and the correct book is fetched and assigned to currentLoan.

# Conclusion 2

Neither bookID nor currentLoan have any bearing on the issue.



# Hypothesis 3

# Overdue Fine remains 0 after being calculated

# Test 3

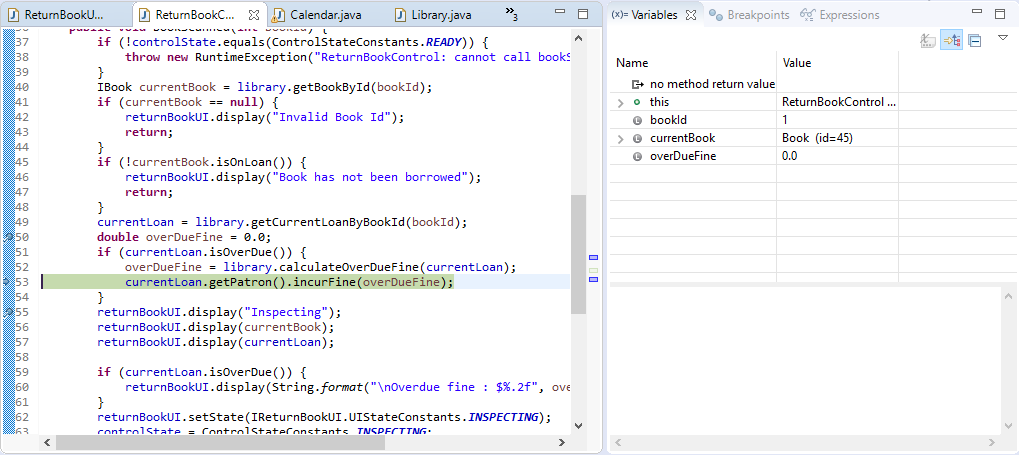
# Check the value of Overdue fine in bookScanned

# Result 3

# True

# Conclusion 3

The loan being overdue triggers the calculation but something must be funky with the calculation itself.



# Hypothesis 4

# The issue is in calculateOverDueFine in library. The fine is incorrectly calculated resulting in 0.

# Test 4

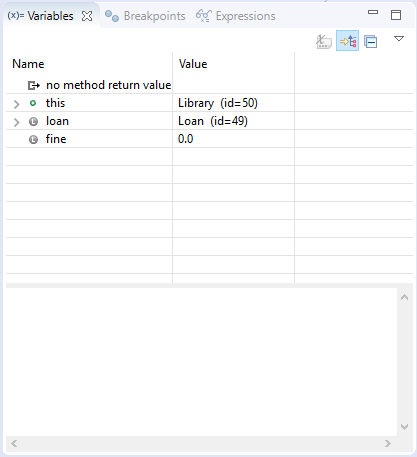
# Check fine in calculateOverDueFine

# Result 4

# False – fine calculation makes sense anything times by 0 equals 0

# Conclusion 4

It’s not the value of fine that is the issue but something else that contributes to the calculation of fine.



# Hypothesis 5

# DaysOverDue is incorrectly calculated.

# Test 5

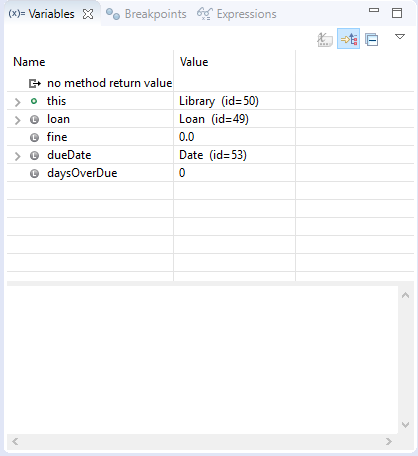
# Check DaysOverDue in calculateOverDueFine.

# Result 5

# True but it isn’t the origin of this issue. (Mostly called this one as I saw it was 0 when looking for fine’s value in the last test)

# Conclusion 5

Must be a value feeding into the calculation of DaysOverDue that is the issue.



# Hypothesis 6

# The wrong duedate is assigned from the loan

# Test 6

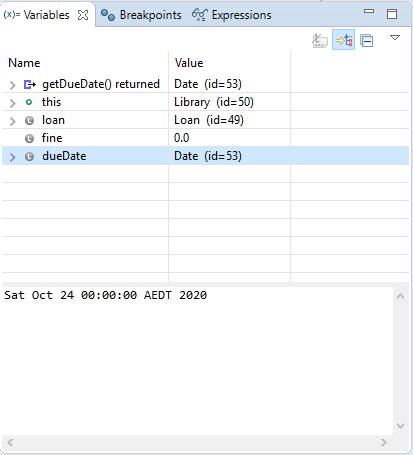
# Check dueDate in calculateOverDueFine

# Result 6

# False – dueDate is correct

# Conclusion 6

Must be another value that feeds into the fine calculation



# Hypothesis 7

# The issue is in the getDaysDifference method somewhere. The returned value diffDays is wrong.

# Test 7

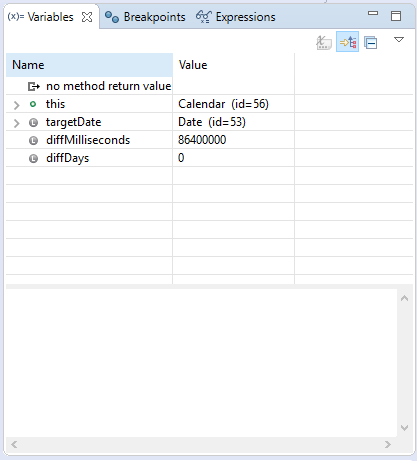
# Check diffDays in getDaysDifference.

# Result 7

# True, it is 0.

# Conclusion 7

The issue must be further back in the calculation for diffDays



# Hypothesis 8

# The issue must be with diffMilliseconds.

# Test 8

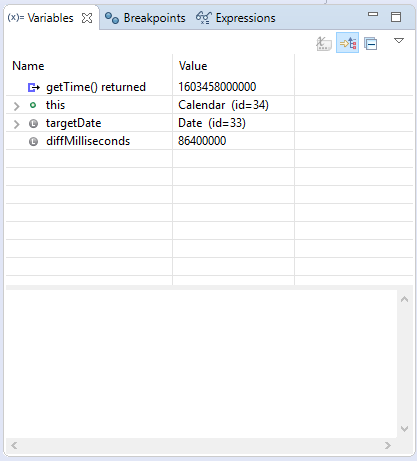
# Check diffMilliseconds in getDaysDifference

# Result 8

# It has a value, unsure if right value, not sure if this is a pass or fail

# Conclusion 8

Not sure where to go from here. Regroup, start again maybe? Working theory at this point, for some reason the value for days overdue is getting rounded down to 0, maybe?



# Hypothesis 9

# After staring at the code for ages, I googled milliseconds in a day. So now assuming the constant MILLIS\_PER\_DAY is wrong, that it’s set to 2 days worth of milliseconds instead of 1. And that by halving the value the calculation for diffDays would result in 1 / 1 which would result in 1, thereby fixing the 0.

# Test 9

# Check diffDays after changing MILLIS\_PER\_DAY?

Only because I can quite literally check MILLIS\_PER\_DAY in the code?

# Result 9

# True, diffDays does in fact return 1 now

# Conclusion 9

I can’t see anything else in the code that would be a likely culprit. Everything else checks out except this one thing and I wouldn’t have even realised had I not thought to google milliseconds in a day. I might be wrong but I’m pretty confident I’m right.

