Item 358

git_comments:

git_commits:

1. **summary:** Merge pull request #915 from Vishwa1311/MIFOSX-1208
   message: Merge pull request #915 from Vishwa1311/MIFOSX-1208 [MIFOSX-1208] Issue with retrieving datatables for offices has been fixed

github_issues:

github_issues_comments:

github_pulls:

1. **title:** FINERACT-986: Update spotbugs to v4.2.0
   body: Update spotbugs to v4.2.0 ## Description Describe the changes made and why they were made. Ignore if these details are present on the associated Jira ticket ## Checklist Please make sure these boxes are checked before submitting your pull request - thanks! - [ ] Commit message starts with the issue number from https://issues.apache.org/jira/projects/FINERACT/. Ex: FINERACT-646 Pockets API. - [ ] Coding conventions at https://cwiki.apache.org/confluence/display/FINERACT/Coding+Conventions have been followed. - [ ] API documentation at fineract-provider/src/main/resources/static/api-docs/apiLive.htm has been updated with details of any API changes. - [ ] Integration tests have been created/updated for verifying the changes made. - [ ] All Integrations tests are passing with the new commits. - [ ] Submission is not a "code dump". (Large changes can be made "in repository" via a branch. Ask on the list.) Our guidelines for code reviews is at https://cwiki.apache.org/confluence/display/FINERACT/Code+Review+Guide

github_pulls_comments:

1. @vorburger @awasum LGTY?
2. **body:** Of you have to use Suppress (it would be better to avoid it), then please create a JIRA and add a comment like this: @SuppressWarning("...") // TODO FINERACT-123 >
   label: code-design
3. > Of you have to use Suppress (it would be better to avoid it), then please create a JIRA and add a comment like this: @SuppressWarning("...") // TODO FINERACT-123 > […](#) Suppress is not helping here. Unfortunately.
4. I tried forcing the function to be able to accept a null parameter, but that just won't do: ``` org.apache.fineract.portfolio.savings.domain.SavingsAccountCharge since first historized release unspecified In class org.apache.fineract.portfolio.savings.domain.SavingsAccountCharge In method org.apache.fineract.portfolio.savings.domain.SavingsAccountCharge.percentageOf(BigDecimal, BigDecimal) Parameter value At SavingsAccountCharge.java:[lines 540-548] value must be non-null but is marked as nullable This parameter is always used in a way that requires it to be non-null, but the parameter is explicitly annotated as being Nullable. Either the use of the parameter or the annotation is wrong. ``` This got me thinking, why would the default value for a decimal be `null` instead of zero. It's a glitch that has worked well so far, probably because we never stumble on the crash. The method is vouched to fail if a null value is passed through.
5. The method is updating the values, in the savings account charge now the null value was there to initialise it , might be the case that it was checking for null values somewhere else while applying the charges and then based on that computing from zero. This could be interesting to note that we can start to remove all the null values where it should be a bigDecimal as zero functionally . What are your thoughts ?
6. Had some other concern, passing a ZERO to the function and performing a ZERO multiplication means the method returns a ZERO. I fear what a bunch of ZEROs everywhere may do to the core system functions.
7. I just created https://issues.apache.org/jira/browse/FINERACT-987 for this.
8. @awasum, @vorburger, @maektwain can we conclude this thread on these changes?

github_pulls_reviews:

1. **body:** Hoping this is not checked for null somewhere in the code base?
   label: code-design
2. Can't tell for sure, Just know it's a nullable field and as null, spotbugs throws an error when the value is passed to `update.percentageOf(BigDecimal, BigDecimal)`. So I just replaced that particular instance where I needed it to be something other than null
3. I think we cannot have this zero, since it would affect the calculation, there is a reason why they were using null, Zero means there is some value provided ie zero, did you test it though?
4. **body:** Hmm, This is troublesome. I thought this was the most subtle way of going around this. I don't think multiplication by null is possible. This is actually some faulty method but I think I'll see how to make spotsbug ignore this just to stay on the safe side
   label: code-design
5. Yes true, might be just initializing it with null and then populating with real values... Could be like that

jira_issues:

1. **summary:** Update spotbugs to v4.2.0
   description: I faced a new issue with spotbugs reporting an error but not providing details {code:java} > A failure occurred while executing com.github.spotbugs.snom.internal.SpotBugsRunnerForWorker$SpotBugsExecutor > Verification failed: SpotBugs execution thrown exception {code} Look at [https://github.com/spotbugs/spotbugs-gradle-plugin/issues/210]

jira_issues_comments:

1. [~kaze] for future issues you create yourself, could I invite you to directly set Fix Version (to 1.4.0) and mark In Progress, and close as Resolved when PR is merged and you consider issue fully done? Thanks! /Cc [~awasum]
2. [~kaze] I had not really fully read your description above here, and independently filed and investigated FINERACT-989 - now duplicated! :D Thanks for this!