Coding Guide

Code	Definition	Example
way to reproduce	Emphasize that GFIs descriptions should clarify how to reproduce the issue.	1) We wrote up exactly what the problem was and pinpointed the exact code that needed to be changed. (D1); 2) The description may clearly explain where the issue is. (D15)
detail description	Just indicate GFIs should have a detail description without details provided.	1) very simple changes that we pretty much spelled out in the description. (D2) 2) Good first issues usually have a detailed descriptions. (D23)
way to fix	Emphasize that GFIs descriptions should clarify the way to fix the issue.	Issue description had also already suggestions about how this could get fixed. (D6)
clear description	Just indicate GFIs should have a clear description without details provided.	1) The issue to be addressed was described clearly in the issue. (D7); 2) The issues with clear descriptions can be identified as good first issues. (D3)
scope of understanding	The cope of understanding is limited.	the scope of understanding necessary. E.g. I'm more likely to label extension code (which no other libraries depend on) as GFI and not runtime code (which a lot of extensions depend on). (D1)
not involve multiple file/function	The implementation does not involve multiple files/functions.	1) This issues is just a minor fix to a single file and do not need deep understanding of the whole project. (D24); 2) The change that is isolated to a single file is more likely to be solved by newcomers. (D22)
not involve wide-spread APIs	The implementation does not involve wide-spread APIs.	it doesn't change wide-spread APIs that require chasing callers; (D4)
unit-testable change	The implementation does not involve unit-testable.	ideally, it's unit-testable. (D4)
minimal impact	The change should have little impact to other part of the software.	1) Difficult also can be, if adding feature breaks something else. (D5)
self-contained change	Just indicate GFIs should be self-contained without details provided.	Small, more or less self-contained tasks where the team already has a vague idea of how the fix might look like. (D14)
basic knowledge needed	The implementation only needs basic knowledge.	This issue requires some basic understanding of Java (understanding a stack trace and reading through some

		existing code to fix an exception arising in a particular situation). (D2)
no internal implementation knowledge needed	The implementation does not involve internal implementation knowledge.	it didn't require internal implementation knowledge, it was just about documenting a thing better. (D6)
no additional interpretation needed	The issues is straightforward, so no additional interpretation is needed.	Issues don't require additional interpretation or understanding to tackle the issue. (D12)
repetitive task	The implementation is a repetitive task, e.g., similar to another feature.	Repetitive task - once you've done one, doing the rest should be quite straightforward. (D19)
limited lines of code	The implementation involves limited lines of code.	The amount of code needed to be written is minimal (a few lines at most).(D2)
limited time	Solving the issue does not require much time.	As for what we use to judge if it's suitable for newcomers, usually it's the amount of time it would take to implement it (usually < 3h). (D5)
limited investigation	Do not need deep investigation before solving an issue.	This issues is just a minor fix to a single file and do not need deep understanding of the whole project. (D24)
rough idea	Project members should have a rough idea on how to solve when labelling a GFI.	I treat the issues as good first issues that whether I have a rough idea of the implementation so I can provide support for newcomers. (D2)
available support	Whether can provide available support is an important factor when identifying GFIs.	I can support newcomers if they get stuck and lead the decisions. (D4)
visible final impact	Solving this issue is expected to have a visible final impact.	It has a visible final impact on users. (D8)
medium-high value	GFIs should have medium-high value.	Issues with medium-high value: it can provide a chance to learn about and practice with a new feature.(D12)
interesting change	The change of GFIs should be interesting.	Could this provide an interesting challenge? Does would solving this help someone new learn a little bit about our codebase? (D20)
low urgency	The priority of GFIs should be low.	I usually don't labelled the issues with high priority as GFIs.(D17)