Retrospective - second iteration

Syntax Highlighters

March 22, 2018

What worked well?

Communication: We've greatly improved our communication from the first assignment, and discussions are now held very frequently on our Discord server. This change has probably happened in part because there are more concrete tasks to work with, and also because we have a very clear heuristic for determining whether something someone has done is good enough - if it's not working, it isn't. This lends itself well to discussion of implementation details and small things. Partly to blame is probably also the increased enthusiasm from actually getting to work with programming.

Attendance: We still have a pretty good attendance, given the size of the team and our different schedules - we were lucky to find a second meeting time that worked for everyone. Considering how we've also sometimes lacked members due to various health or travel related issues, we've handled it pretty well. The team members are very diligent in notifying the rest of the team of their absence ahead of time, if they for some reason cannot be there.

Increased base experience/skill: As every member is different in terms of their knowledge of programming and experience with different tools, we've all learned from each other. In particular, many members seem to have gotten a significantly greater understanding of the Git version control system, largely because it's an integral part of the project, so that a certain level of mastery of this tool is required. The fact that we've had a ”support team” which has been able to provide instructions remotely has probably helped the people having git issues to learn the effects of the different commands, as they can see them take effect after typing them in. We believe that we have also improved in programming by seeing how different people solve different issues, and in teamwork, simply through learning by doing.

Role distribution: While we initially intended to have a looser role assignment, with everyone stepping up as required, we found that in this project we naturally drifted into certain areas of expertise, to the point where it would almost always be easier for the person already working on a topic to add a certain functionality, rather than anyone else jumping in and having to read up on that part of the project before being able to start writing the code. The main areas of expertise were UI (mostly handled by Loc and Robin G), AI (impressively implemented solely by Stian), tests (largely done by Sverre and Robin E), and the rest of the project centered around the game logic and its structural foundations, which the rest of us scrambled for.

The advantage of this was that if one person could immerse themselves in a topic and leave the rest of the project to the others, they could work much more efficiently as they were familiar with the code they had at their disposal. However, there were also some disadvantages - see below.

What didn't work so well?

Task assignment/inflexibility: Although we were significantly better at distributing tasks than the last assignment, we still found it difficult, particularly as the project dragged on and people became entrenched in their own areas of code. We've traded the flexibility of breadth for the increased depth of a narrow focus, but this can also lead to inefficiency, particularly when people are waiting on implementations of functionality which are outside their area of expertise, which halts their progress.

Focus during group meetings: It's still hard to hold a conversation involving the entire group for long. However, we feel that although holding everyone's attention for two hours straight is too much of a challenge, it's also less important during this iteration, since there is less large-scale planning and more particular implementation details which can be brought up directly with the people involved. Meetings are now largely an arena where one can try to bring things up with the group, or otherwise branch off in subgroups to make plans, discuss particulars or receive guidance about something which another member has expertise in. We feel that this has worked out somewhat okay.

Design conflicts: A number of times a certain thing happened: A member would write an initial piece of code for something - later, for some reason or another, another member would have to review that code, and for reasons of their own decide that this would be better done in a different way. They would rewrite and refactor, and this would in turn invalidate the original author's understanding of a certain part of the code. Such a thing can be largely demotivating and foster ill will, and it's not certain that there will ever be proper agreement between everyone about how something should be done. This has been an issue in particular because the API has been organic, ever-changing and ever-growing, because as we found new challenges to tackle, our approach to solving the original challenges evolved. This is not necessarily a bad thing - as requirements change, we fall or prosper by how we adapt - but it's something which is worth a mention, as it has been quite a challenge to us during this iteration.

Testing: For a while, it's been difficult to test anything, in particular because the final version of the API has not been clear until the late stages of the project. We have tried to write tests as best as we can, but mostly they've been written in hindsight, which, although better than no tests, is not really in the spirit of test-driven development. This has, however, been very soothing to know that for the tests that have been written, if a refactor doesn't break them, then at least that part of the code still works as it's supposed to.

Contribution on (and off) the record: Throughout this project, a lot of the task distribution has been done informally, often through Discord messages, and without bothering to create a proper Gitlab issue, especially if the problem was minor and easily fixable. This, however, means that a lot of the commit messages don't have a clear issue to refer to, and as such, the structure of them may vary. Another big problem was that we didn't really have any clear commit writing policy until a bit into the project, so during the early stages there are some nonsense commit messages which we now cannot fix without completely rewriting history, which is something we'd prefer to avoid. We've greatly improved on this, however, and now have contribution guidelines on the record.

Related to this is also how, since a lot of the discussion has been informal on our off-record communication channel, the issues don't always include a lot of information on the problem. This is a problem, since the more detailed the information in an issue is, the easier it is to work towards solving it. Issues are also great for documenting the process we've gone through. We hope to improve on this during future iterations, but it's difficult to come up with any strategy that amounts to more than "encourage members to follow best practices, and hope they do".

Issues with libGDX: Due to lobbying from certain members, we decided to use libGDX for our UI. The results have been mixed, as certain issues led to problems with running the program on Mac OS for a long time. Luckily, we managed to identify the cause of this issue, introduced by upgrading to a different version of libGDX, and were able to eliminate it, to the detriment of none.

The future

It's difficult to say for certain what the future will bring, and anything we say here might become null and void in the face of the next assignment. However, we will most sincerely attempt to address the issues described above. Notably, it may be a good idea to consider the breadth vs. depth tradeoff again - the advantage of breadth being flexibility in workload distribution, and of depth the ability to focus one's effort on doing a particular part of the project better, in addition to it being the method we actually have the most experience in, since we have already done it for one iteration. It would also be good to make sure that everyone is on top of the contribution guidelines, and to document our progress better in terms of Gitlab issues, something we've clearly been slacking off on.

It's also possible that we'll consider a role change for the next iteration, as some people have expressed a wish to work with different parts of the project. However, we will have to wait with deciding that until we see what the next assignment has in store for us.