Sprint 2 Report

Brett Raible
Nicholas Paquette
Maren Sandner
Maike Keune

In the Sprint 2 report we will talk about our progress during the second Sprint of the computer science project.

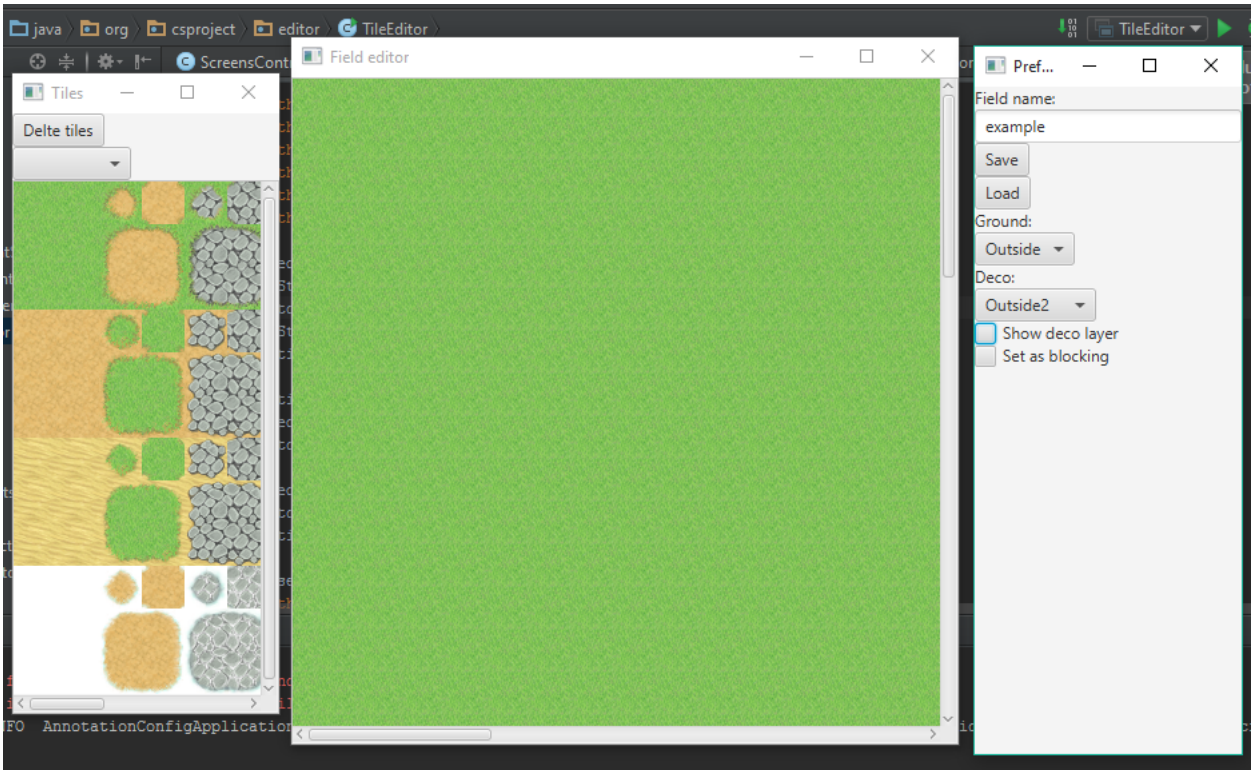
Progress reflection
Our goals for the second Sprint were to implement a random dungeon generator, a random world generator, and a working GUI. Also, we wanted to implement a money system, magic, items, a “fog of war” effect, and loading/saving.

To create the random dungeon, we used an algorithm that creates the impression of seemingly random pathways. This algorithm works by randomly splitting the field into squares with different sizes. It does this until a parameterized threshold is reached. Afterwards, it connects pairs of squares with a path.

Another new implementation is our field editor with which we are able to build maps, rooms and towns. So far the editor is fairly limited, and we will need to expand it in the following weeks. For example, we plan to add tile chunks to the editor so it will be easier to add decorations to a town, room, or map. The editor is able to put start points and teleport points to the map, which we will explain a bit more of in the little editor tutorial below.

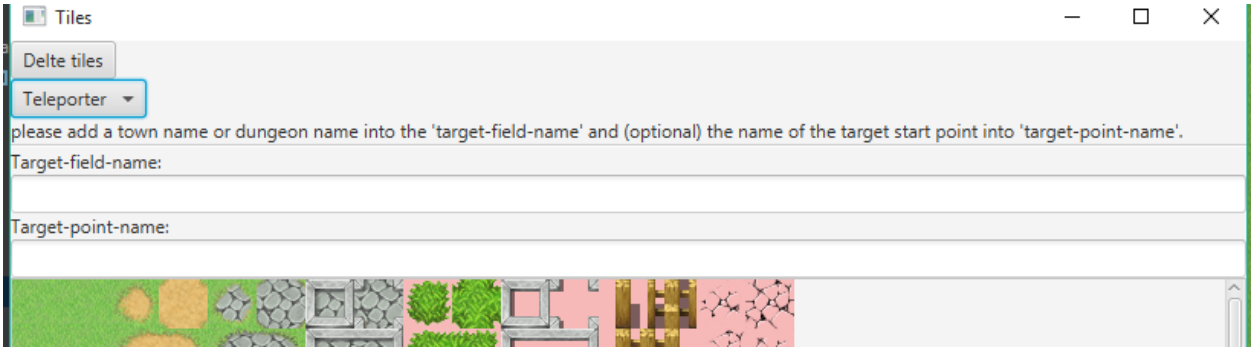
Finally, we developed camera movement so the whole map has to be explored by walking the character around. We decided that creating a fog of war would be too much to handle in the short amount of time we had. The camera movement stops when the end of the visible field is reached.

Editor tutorial

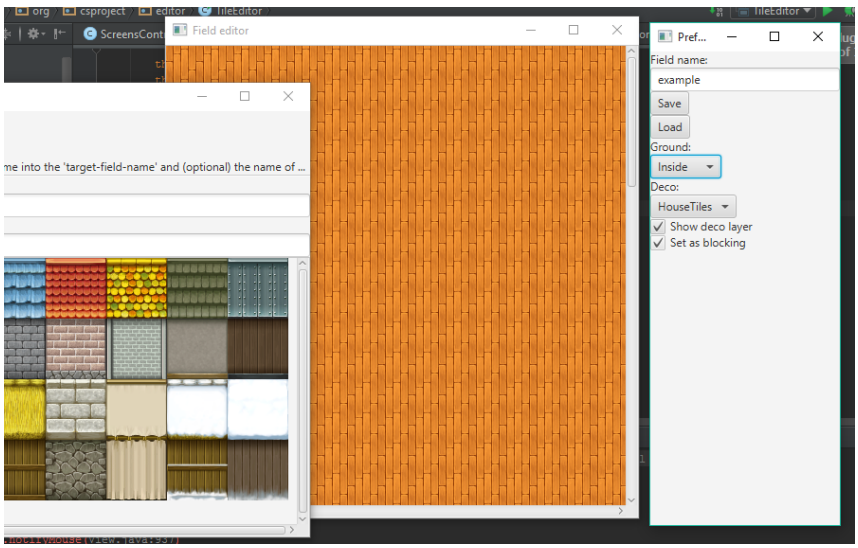


First, we will give an overview of the editor. The editor consists of three windows which can be closed separately. It can be started separately with its own main class: TileEditor.java.

The central window shows the field as the game would display it. The left window shows placeable tiles, and has the ability to delete tiles from the field and add new startPoints and teleportPoints with it. The right window allows you to load files or save the current field with a name of your choice. These files will be saved to c:fields on your hard drive. So you have to copy and paste it into the project in intellijIDEA. You can choose the tileset image of the ground-layer or deco-layer for creating your field. Currently there is a very limited selection of tile-sets to choose from, and we hope to expand more on this in the future. Clicking the checkbox allows you to use the deco-layer, which allows you to add decorations such as rocks or trees. The “set as blocking” checkbox will be checked automatically, so decorations will be non walkable. You can uncheck it if you want decorations to be walkable.



Now we will explain start points and teleport points. Start points are set to define where you want your character to be set on the field. Each start point can have its own name. Teleport points are used to transport characters between fields. If the character enters a town or a dungeon, for example, you must add a teleportPoint on the town-tile or dungeon-entrance-tile. As seen in the picture above, names can be entered for the “target-field-name”, which must be a json-file. Also you can give the targetPoint a name, which is convenient because you can always take a look into our json files. Teleport points are a necessity for being able to transition to areas other than the world map, whether it be a town or a dungeon. The town itself has startPoints where the character will be teleported to upon entering.

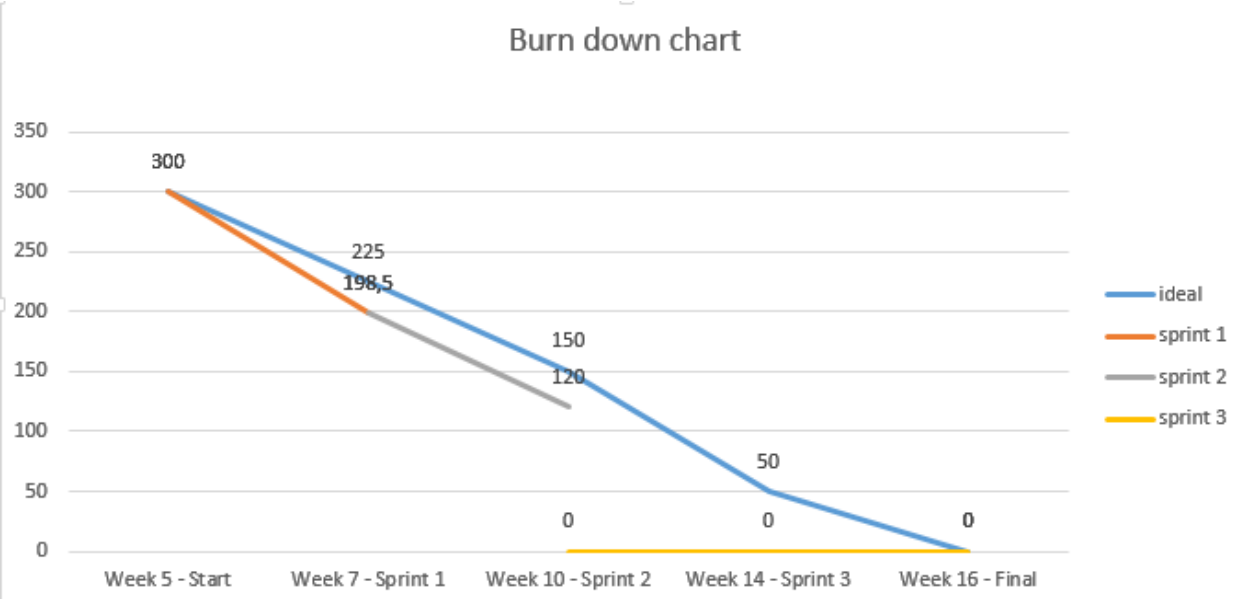


This is another view of the editor. This shows how the inside looks on default. You may change that by clicking other ground tiles on the field.

Problems encountered

We planned to implement a fog of war, instead of that we developed a camera movement so we can explore dungeons, towns and maps. This provides an effect similar to a fog of war, which we believe would be too much of work to have it implemented. Perhaps we can attempt to implement a fog of war if there is enough time in the last Sprint. At the moment we have too many remaining tasks to focus on a mainly aesthetic feature. We also were not able to communicate as much as we had hoped at the end of sprint 1.

Burn down chart



As seen in the chart we worked more hours than our estimated time. The work hours are more evenly spread this time than in the last sprint. But we think we will have to work even more in the next sprint because we have a week less than planned (because of the delayed sprint 2 meeting).

Projected Progress

All of the things we said would get done this sprint (the random world generator and the static town) got done. The static town, unfortunately, due to a new implementation of the JSON files must be remade. File saving and loading is implemented by pressing ‘c’ in game to save, and saves a short text file which can be loaded and parsed to load the game. The random world generator is currently working, but still lacks some features and is thus not implemented yet. We are on track with our backlog and no changes need to be made except that we need to work on the random world generator a little bit more to get it fully working.

Conclusion

At the beginning of the computer science project we created a plan for making our game and thought about what we would need to develop it. However as we continue working on the game new problems emerge that were not expected and need to be solved or worked around. Some ideas like tile chunks allow for convenient ways to create towns and dungeons, or the editor which allows for the handmade creation of fields, make it a lot easier in the future to build new maps, rooms, and towns. There was a bit of lack of communication this sprint. We had no skype meetings, though we communicated through skype chat. We plan to fix this during the next sprint, by attempting to implement a plan of weekly assignments. We plan to create a chart that lists all of the tasks for this sprint that we need to implement, as well as optional tasks to implement, and to assign people to each task. We still feel we are on track for completion of our project.