*Juris Civilis Development Diary*

| Date of Session | Time Spent | Development Segment | Notes |
| --- | --- | --- | --- |
| 14/10/2021 | 20 Minutes | Meeting | Discussed what documentation was needed and what steps were required for starting the project. |
| 21/10/2021 | 30 Minutes | Meeting | Discussed progress on project plan and any steps forward needed |
| 04/11/2021 | 2 Hours | Analysis | Began writeup of the project literature review, specifically the review of “Dwarf Fortress” and its use of world generation and world simulation |
| 04/11/2021 | 1 Hour | System Design | Began development of some basic tools for the project to work from – specifically some basic camera controls (movement and zooming speed), as well as some work on mouse clicking interactions and the spawning of basic 2D circles as a response to clicking interactions. |
| 06/11/2021 | 3 Hours | Analysis | Added multiple segments to the literature review, including references to Europa Universalis 4, Civilization 5 and a past project of mine with relevance to the topic area. |
| 06/11/2021 | 3 Hours | System Design | Added more to the aforementioned tools, including the basic ability to generate polygons around a selected point, and application of properties to these polygons which can be accessed by clicking on them. This system has some flaws, notably with the hit detection of the polygons, but these should be mitigated as the project progresses. |
| 13/11/2021 | 4 Hours | System Design | Began to use the research collated from the literature review stages to document the design of the world generation procedure – specifically the discussion of the geographical features of the world and values such as rainfall, elevation, and temperature. Began discussion of province mechanics. |
| 14/11/2021 | 7 Hours | World Generation – Objective 1 | Attempted to create a prototype for the diamond-square algorithm to be imported into the main system. This unfortunately was largely a failure but can be built from in the future. |
| 15/11/2021 | 7 Hours | World Generation – Objective 1 | Attempted to implement a Perlin noise algorithm, to limited success. The algorithm still needs work to be used in the project. |
| 18/11/2021 | 20 minutes | Meeting | Discussed progress with literature review and world generation methods, confirming progress on these topics has been acceptable thus far |
| 18/11/2021 | 4 Hours 20 minutes | World Generation – Objective 1 | Added the first working terrain generation algorithm, using a reworked Perlin noise algorithm and fractal Brownian motion. This algorithm is able to differentiate land and sea as well as mountainous regions. The source code for running the other Perlin functions is still present in the code for later use in other functions such as temperature. |
| 24/11/2021 | 6 Hours | World Generation – Objective 1 | Started the unity side of the project and ported over the existing terrain generation algorithm into the unity system – then added the ability to perform additional Perlin generation algorithms with variable settings to represent other features such as temperature. As of yet the algorithm is largely inefficient and poorly implemented, but changes will be done to improve the performance of this system. |
| 25/11/2021 | 5 Hours | World Generation – Objective 1 | Added some code to make a basic equator – this code is not yet perfected as it does not properly blend with its surroundings, and needs some reworking to ensure it blends properly with the Perlin temperature system. |
| 28/11/2021 | 1 Hour 50 minutes | World Generation – Objective 1 | Reworked the equator system to blend better with the surroundings, also removed a significant amount of the randomness of the system to provide a more consistently positive result. |
| 29/11/2021 | 1 Hour | World Generation – Objective 1 | Added some small cosmetic changes – including some basic threading incorporation to stop “not responding” issues when the program takes time to generate stages. As of yet this means that the software simply tells the user what stage they are on – but there may be merit to implementing a system to draw the map after each stage, though this could cause some slowdown issues. |
| 6/12/2021 | 2 Hours 10 minutes | Progress Report | Began the writeup of the progress report, demonstrating what has been achieved and how the project is on track to meet its goals. |
| 7/12/2021 | 1 Hour 30 minutes | Progress Report | Added additional details to the progress report to better catalogue what has been achieved thus far |
| 18/12/2021 | 5 Hours | World Generation – Objective 1 | Added the basics of the biomes system – splitting the land into different types so that environmental factors can be better modelled. As of yet the feature is not successful, but the code is there to permit the spawning of biomes, it just needs refining. |
| 19/12/2021 | 2 Hours 10 Minutes | World Generation – Objective 1 | Implemented the biome system properly – allowing for the generation of a world with different climates. As of this stage, there should be sufficient progress to move onto the next goals as well as update the dissertation document. |
| 20/12/2021 | 2 Hours | World Generation – Objective 1 | Attempted to clean up the code for the existing system, as the memory usage was unacceptably high. During the course of this, I discovered the primary source of the memory issues came from the getting of deciles in the system. Due to prior coding faults, the median algorithm was unintentionally appending all of the property data into a single set and finding the same deciles for each value – this was changed to use the original intention of the data using its own deciles per property.  Unfortunately, this solution still suffers from the main problem impacting memory usage – the fact that the deciles function stores up to 4x4000x6000 integer values and sorts them. This uses the majority of the memory space available to the system, and improvements to this algorithm could improve the performance of the system tremendously. The current proposed solutions are thus:   * Switch from using specific medians to using the upper bounds and lower bounds of the set to calculate approximate deciles. This procedure would be the most memory efficient but would also have the least accurate decile data, as high ranges could distort the values significantly. * Use the medians of medians algorithm. This algorithm is designed to provide the median of an unsorted set without first sorting the set – meaning potentially less data needs to be stored while not significantly sacrificing the accuracy of data.   Both of these solutions must be considered as they each present the ability to reduce memory costs at the expense of accuracy, but which fits the intentions of the program best is not yet clear. |
| 21/12/2021 | 2 Hours 40 Minutes | World Generation – Objective 1 | Reworked various systems to improve memory-usage. First and foremost, this included removing the storing of integer values for each tile – instead deferring to using the enumerables that are generated from the integers. By removing the unnecessary storage of the integer property values, each pixel in the map now takes up 16 bytes less. Additionally, the deciles system has been reworked to use the upper bounds and lower bounds as previously discussed, removing the need for the declaration of a new list for calculating deciles.  As a result of these changes, the memory usage of the features currently implemented has dropped by a full third. While the system overall still uses a heavy amount of memory, lowering this load by such a significant amount helps immensely. |
| 23/12/2021 | 4 Hours 30 Minutes | World Generation – Objective 2 | Began to implement the chunk system to allow for future generation of provinces, as well as implemented chunk-based map displaying which operates infinitely faster than the previous pixel-by-pixel map drawing. |
| 26/12/2021 | 3 Hours | World Generation – Objective 2 | Added the ability for the chunk system to create two equal right angled triangles using either top left->bottom right or top right->bottom left splitting. Each of these triangles functions as a chunk as before, but will be vital in creating the provinces system.  The orientation of the triangles is random per tile – using a custom linear congruential generator (with parameters from glibc) to permit for a pseudo random stream of data. This must be used as the randomisation procedure included in c# by default uses system clock data to generate a result, therefore any processes completed within the same timeframe will receive the same value – by using the system clock random as a seed for the following modulo function this restriction is removed.  As a result of these procedures, the memory usage has once again spiked, but at this stage of development there is minimal need for the pixel set to remain in memory, therefore it may be possible to save this data to a file and discard it. |
| 28/12/2021 | 2 Hours 30 Minutes | World Generation – Objective 2 | Began the procedure of dividing chunks into provinces – the final process in the generation of the map itself. After the provincial generation is complete the map can be saved and the other factors for each province can be generated. |
| 29/12/2021 | 4 Hours 20 Minutes | World Generation – Objective 2 | Added the basic generation of provinces with a slow algorithm. When perfected this should allow the saving of the data generated so far and then generation of the other factors needed for the objective to reach completion. As of yet it is too slow to run as intended, taking sometimes over half an hour to finish generating. Additionally, due to the low speed of the generation, very little has been achieved in terms of testing these procedures. |
| 30/12/2021 | 3 Hours 15 Minutes | World Generation – Objective 2 | Reworked the province system once again so that each chunk now keeps a record of its adjacent chunks – allowing for the province generator, that previously took over half an hour, to process in almost a minute and a half. The next stage of development is to add the ability to render these provinces on the map, the basics of which have been implemented. As of yet there is little indication on if the province system has been successful in the structures it generates, but when the province renderer is complete adjustments can be made where necessary. |
| 31/12/2021 | 4 Hours 15 Minutes | World Generation – Objective 2 | Added province rendering to show all the generated province boundaries. Early into this procedure issues were found in which the provinces were displaying as overlapping sets, but this was identified as a rotation issue. Rotating the provinces by 180 degrees x and 180 degrees y created a fully mapped provincial system. As of now, there are still oceans included within the province definitions – as well as some small provinces that could be put to better use by combining them with other nearby provinces. |
| 01/01/2022 | 2 Hours 30 Minutes | Dissertation writeup | Added new segments on the dissertation writing about the progress achieved for the first stage of development, as well as what changes were made to the design. |
| 02/01/2022 | 3 Hours | World Generation- Objective 2 | Began implementing the ability to save the map data to a file – so that obsolete data can be removed from memory when conducting the next parts of the generation procedure |
| 11/01/2022 | 2 Hours 30 Minutes | World Generation – Objective 2 | Reworked the map writing algorithm to write the map geography to a png file – allowing it to be easily loaded as a material later. Also added some basic province adjacency storing code for later usage in the simulation, as well as the initial aspects of province data writing. The subsequent updates should focus on improving province saving and appending new province data. |
| 12/01/2022 | 3 Hours | World Generation – Objective 2 | Fully implemented province saving using xml files and a png for the map geography. This implementation means that new properties can be appended easily and that a lot of unneeded data can be removed from memory. Also began to add some new UI elements to replace the existing UI – including adding a new menu screen for generating a map. |
| 13/01/2022 | 4 Hours | Simulation – Objective 3 | Began to implement some basic viewing tools, including a new UI element designed to allow switching between map modes. This should lead the way to allowing a user to select provinces so their data can be displayed. |
| 14/01/2022 | 3 Hours 20 minutes | Simulation – Objective 3 | Added the map modes for showing elevation, temperature, rainfall and flora, as well as the storing and saving of these datapoints for provinces. Also began to work on implementing the single province viewer, but ran into issues with mesh hitboxes. |
| 17/01/2022 | 2 Hours 20 Minutes | World Generation – Objective 2 | Simultaneously to researching ways to fix the mesh hitbox issues, implemented the province name generator functionality – providing a unique string name to each city on the map. This algorithm uses data provided by **Geonames** under the creative commons license, though the algorithm is of my own design. |
| 18/01/2022 | 3 Hours 20 Minutes | Simulation – Objective 3 | Fixed the mesh collider by rendering the vertices of a triangle in the correct order. Also added the ability for the province data screen to show information pertaining to a province when a user clicks on the mesh. |
| 19/01/2022 | 3 Hours 40 Minutes | World Generation – Objective 2 | Added the culture generation system and mapmode, providing a new culture value for every province on the map. This may later be expanded to have both culture groups and cultures, but this isn’t decided as of yet. The cultures generated have their own names and are saved in the simulation files. |
| 20/01/2022 | 4 Hours | Simulation – Objective 3 | Reworked the ability to select provinces, making selected provinces pop out more, allowing for easier selection. Also reworked the existing map loading mechanism to be independent of the map generation script – allowing this script to be reused for the simulation. |
| 23/01/2022 | 5 Hours | Simulation – Objective 3 | Fixed some speed issues relating to the updating and loading of meshes (Caused by repeat calls to the mesh vertices length property, which creates a duplicate array to find the length of) as well as updated other code to allow for better performance |