0 A.D. Feature Implementation - 1

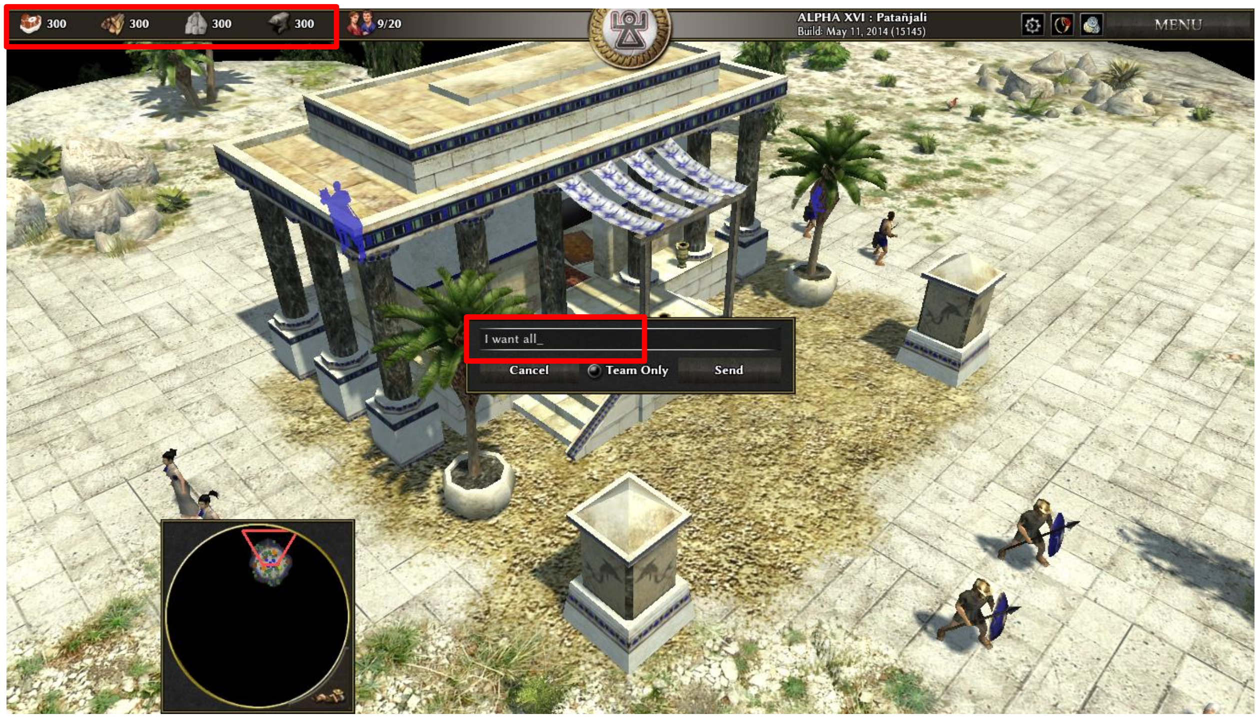
Introduction

0 A.D. has exciting features as most of the other Real Time Strategy game have. Few interesting features are:

- The civilizations are unique in their appearances which include building, units.
- The Combat experience increases when the soldiers fight in groups and go up the ranks.
- Various Unit Formations can be implemented during the battle formations which increase the armor.

Problem Statement

In the game 0 A.D. to increase the army size, we need to gather food, wood, stone and metal. Acquiring those will take a lot of time which will eventually make us bored with strategy games. So we use cheat codes to complete the game sooner or acquire the resources to strengthen the army. As of now we have different cheats to increase food, wood, stone and metal. But we need to enter different cheat codes for each to increase it. So I want to add a cheat code "I want all" which will increase all four of them in a single go.



I want all_

Figure 1: Demonstrating how cheat codes can be entered in 0 A.D. game. Adding the cheat code will increase the wood, food, metal and stone.

Diagrams

Program dependency graph represents the dependencies of several object/statements towards each other. The bellows image depicts the program dependency graph for adding the cheat code.

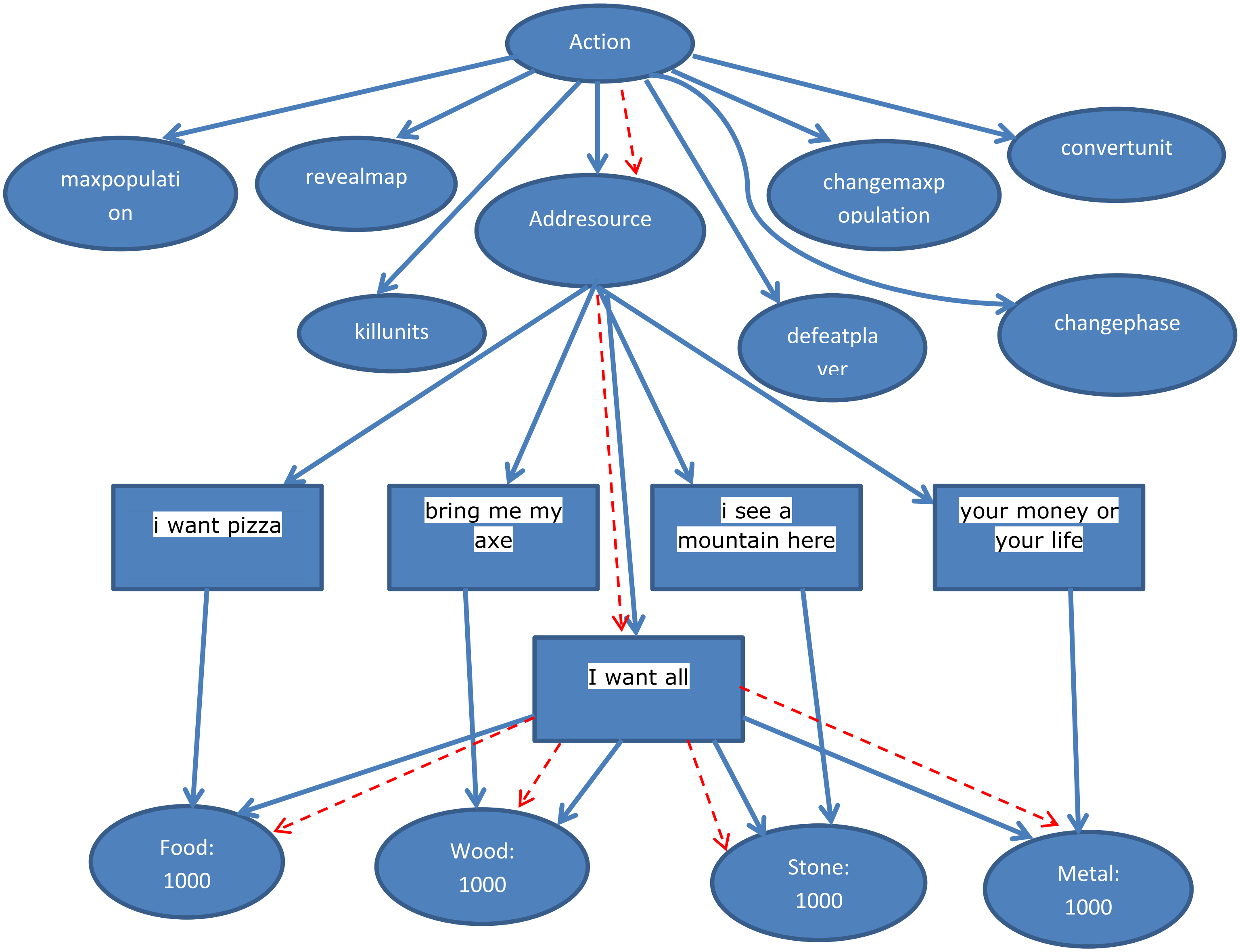


Figure 2: Illustrates the Program dependency graph for executing cheats in 0 A.D. The red dotted line illustrates the flow to execute the cheat “I want all”.

The action acts like a switch case which has many cases where in each case corresponds to a different cheat in the game. The “Addresource” case will handle the food, wood, stone and metal. The cheat pointing to that particular resource will increase that resource by 1000.

Analysis Used

I have put in the game and began playing to pick up how the game can be bettered. Since the game is still in Alpha phase it lacks in a few. I usually use cheat codes to increase the resources in the game. Like typing in “i want pizza” will increase the food by 1000 and similarly typing in “bring me my axe” will increase the wood by 1000. So instead of typing each one separately I want to add a new cheat code “I want all” which will increase food, wood, stone and metal by 1000. I have used is static program analysis, where in I have analyzed the source code and gone through how the cheats are implemented in this game. Adding few extra lines like a case with a condition “I want all” and pointing it to food, wood, stone and metal will increase all four resources. This will add excitement to the game and will keep people playing for longer lengths of time.

Change Plan

Below are steps if carried out will met the objective described above.

1.) Go through the cheats implemented to increase the food, wood, metal and stone.
2.) Add the condition “I want all” in Addresource block.
3.) Inside the “I want all” block call the four blocks food, wood, stone and metal sequentially.
4.) Compile the code in your local system and start the game.
5.) Enter the cheat code “I want all” to check it works.
6.) Deploy the code on to the server and make other users to test the cheat.

0 A.D. Feature Implementation - 2

Introduction

0 A.D. has exciting features as most of the other Real Time Strategy game like

- The civilizations are unique in their appearances which include building, units.
- The Combat experience increases when the soldiers fight in groups and go up the ranks.
- Various Unit Formations can be implemented during the battle formations which increase the armor.

Problem Statement

In 0 A.D. we can garrison soldiers in ships; hence we can transport the soldiers from one place to another. For few civilizations in the game we have war dogs, which can be used in battle. So for the current version of the game we cannot garrison the war dogs in ships.

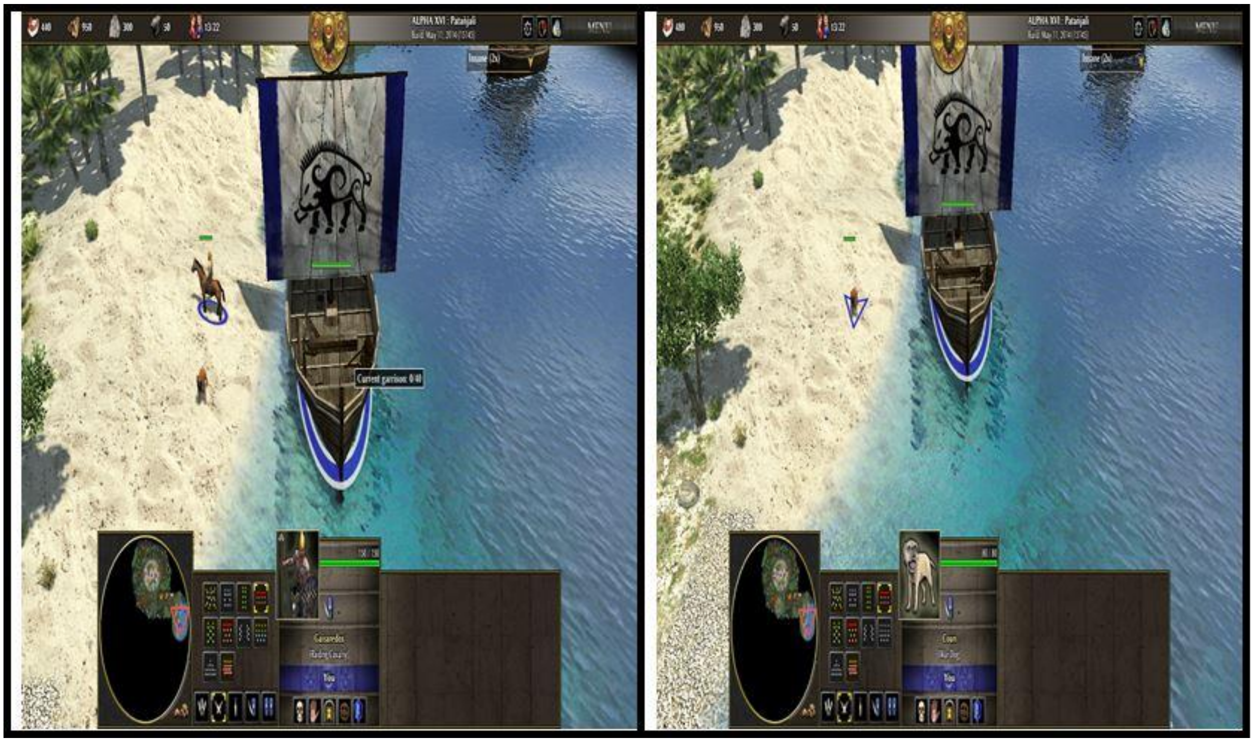


Figure 3: Demonstrating the soldier can garrison in the ship and dog cannot garrison in the ship. (1) The soldier and the dog beside the ship. (2) The soldier is garrison inside the ship but the dog cannot garrison inside the ship.

If the war dogs can also be garrison inside the ship it will increase the team's strength during battles and can also transport from one place to another.

Diagrams

The below diagram depicts the class diagram for bireme ship (ancient warship).

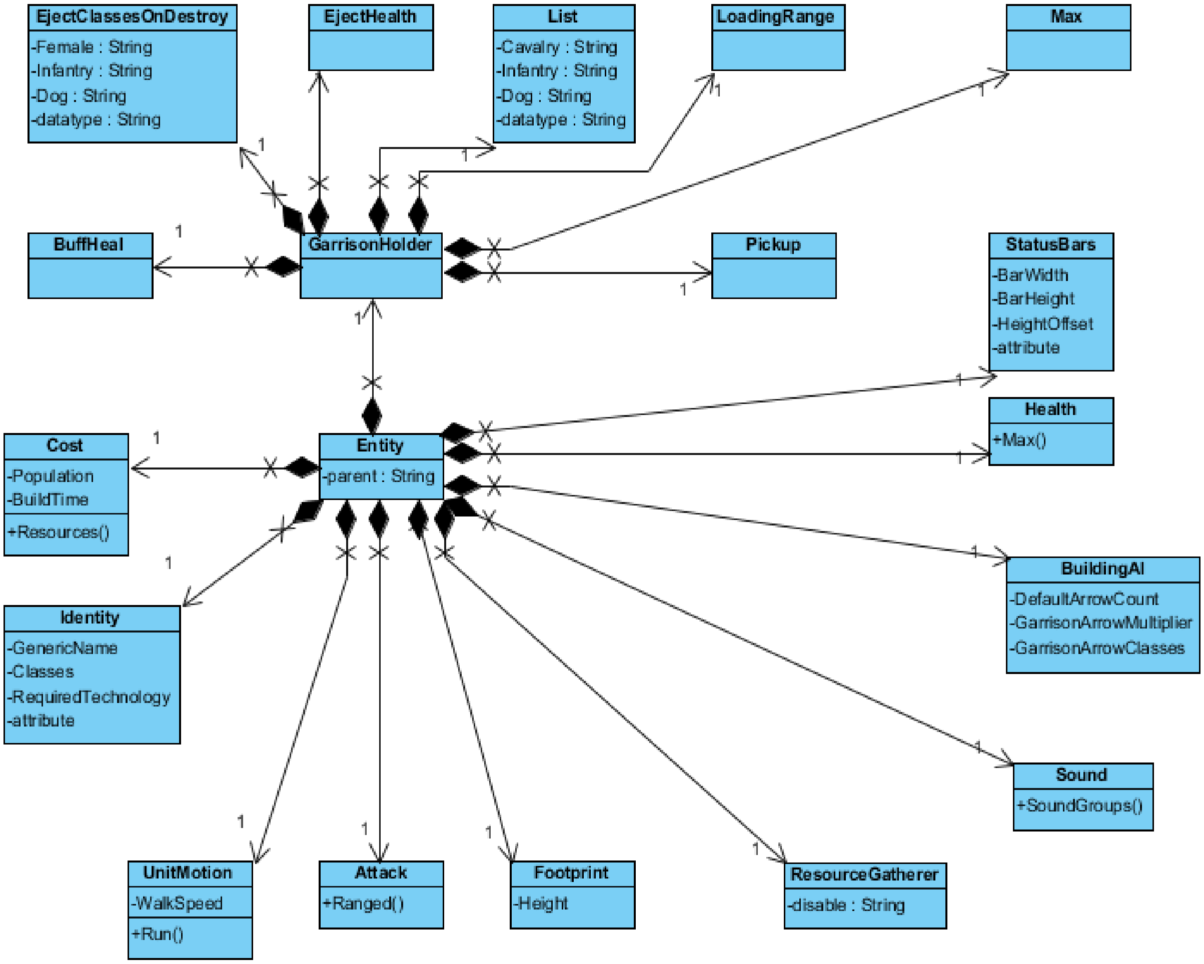


Figure 4: Illustrates the Class diagram for a war ship. Adding the dog to EjectClassesOnDestroy and List will carry the dog and drop it.

The above class diagram depicts a war ship and its features in 0 A.D. Each ship should have all the features and have a strong dependency so the relationship between the classes is a composition. Also each ship as exactly one feature like sound, cost, health, attack etc. so it is 1 to 1 relationship. The garrisonholder class will have few features like transport, ejecting the soldier's garrison inside the ship.

Analysis Used

I have put in the game and began playing to pick up how the game can be bettered. Since the game is still in Alpha phase it lacks in a few. I have faced scenario where I need to garrison the war dog into the ship so I can transport from one place to another, but I was not able to garrison it._ I have used in static program analysis, where in I have analyzed the source code and gone through how soldiers are being transported. They are two things the transport ship does, load the soldiers and drop the soldiers. Adding the war dog to the list of things the transport ship can load and drop will full fill the change. This gives the game a good feature and as dogs run faster so it is easy to explore the map. The garrisonholder class has a list of items which a ship can garrison.

Change Plan

Below are the changes that should be done to meet the objective:

1. List out all the ships which should be implemented with this feature of transport.
2. Considering this to be implemented only for war ship.
3. Check the list of the objects the ship is carrying presently.
4. Add the Dog object to the list and even to eject.
5. Adding this to garrisonholder class list will full fill this feasibility.
6. Compile the code on your local system and start the game.
7. Build a dock and construct a war ship.
8. Try to garrison the dog into the ship and also eject the dog from the ship.
9. Implement the same change to different kinds of ships.
10. Deploy the code to the server and make other users to test.

0 A.D. Improvement of Current Design

Introduction

0 A.D. has exciting features as most of the other Real Time Strategy game like

- The civilizations are unique in their appearances which include building, units.
- The Combat experience increases when the soldiers fight in groups and go up the ranks.
- Various Unit Formations can be implemented during the battle formations which increase the armor.

Problem Statement

In 0 A.D. to build our army we need to acquire wood, food, rock and metal. Currently the objects rock and metal are difficult to differentiate. In the image below the color difference between rock and metal is very less. I would like to increase the color of the metal to make it little brighter so that it will be easy for a player to differentiate between metal and rock.

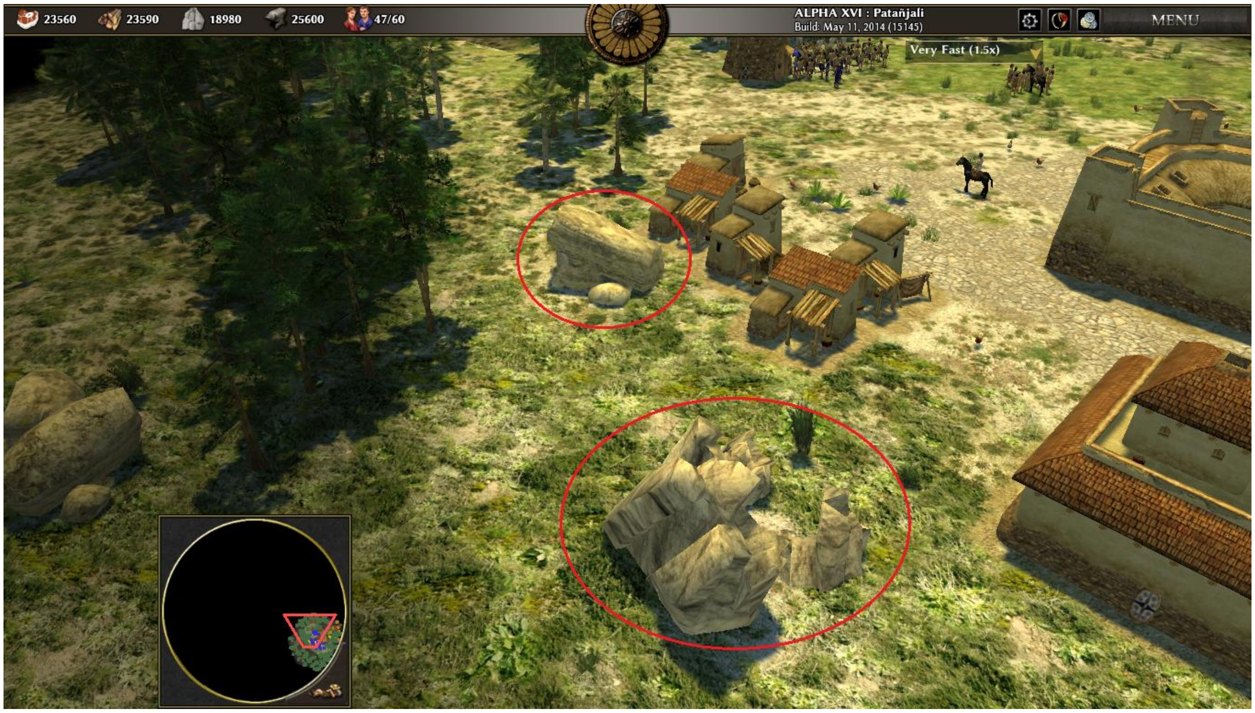


Figure 5: Demonstrating the objects rock and metal in the 0 A.D. game.

Analysis Used

As I was playing the game I came to a point where the color differentiation between metal and rock was very less. I started to go through the files that actually visualize the metal and how the rendering and shading is done. I have analyzed the source code and each scenario has different metal shades. The metal objects are actually drawn in using DirectDraw Surface (DDS) format. These file formats actually provide the graphical look to the objects. So re-drawing the images with a little more color to the metal will be enough. Then we can replace the existing files in the source code.

Change Plan

Below are the steps which will give the list of things that should be done to meet the objective:

1. Identify all the scenarios which have the color differentiation issue.
2. Implement this change only to alpine scenario; we need to redraw the animated files (DDS).
3. Once we have drawn the animated file we need to replace the new files with the existing files.
4. Then run the game and play in alpine scenario and check if the color difference between metal and stone is good.
5. Once this is successfully tested, we need to implement this in other scenarios. After testing the changes in other scenarios.
6. Deploy the code to the server and make other users to test.