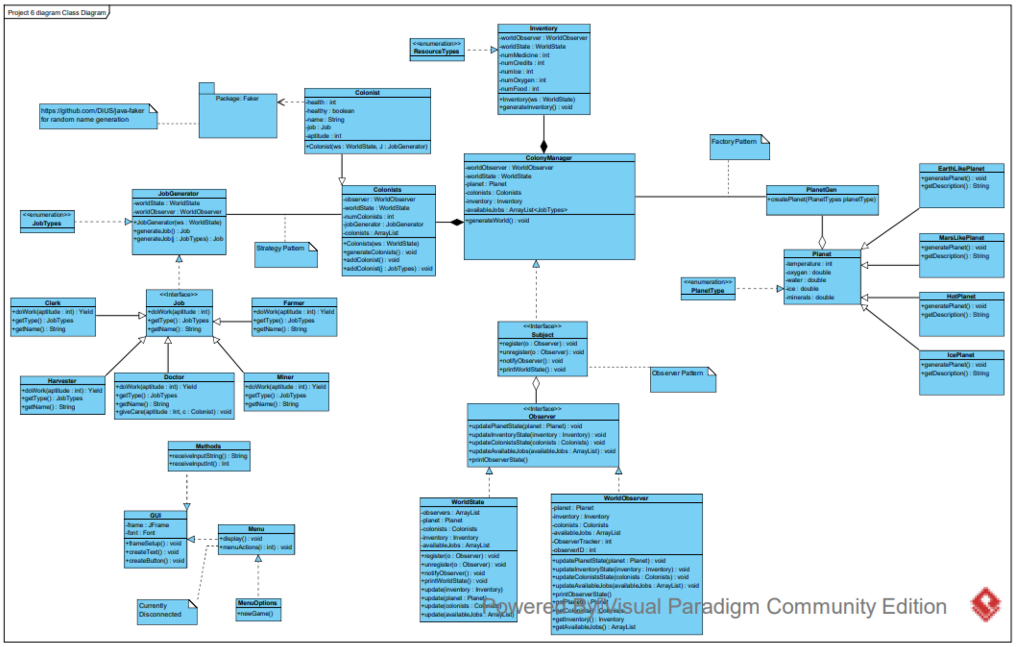
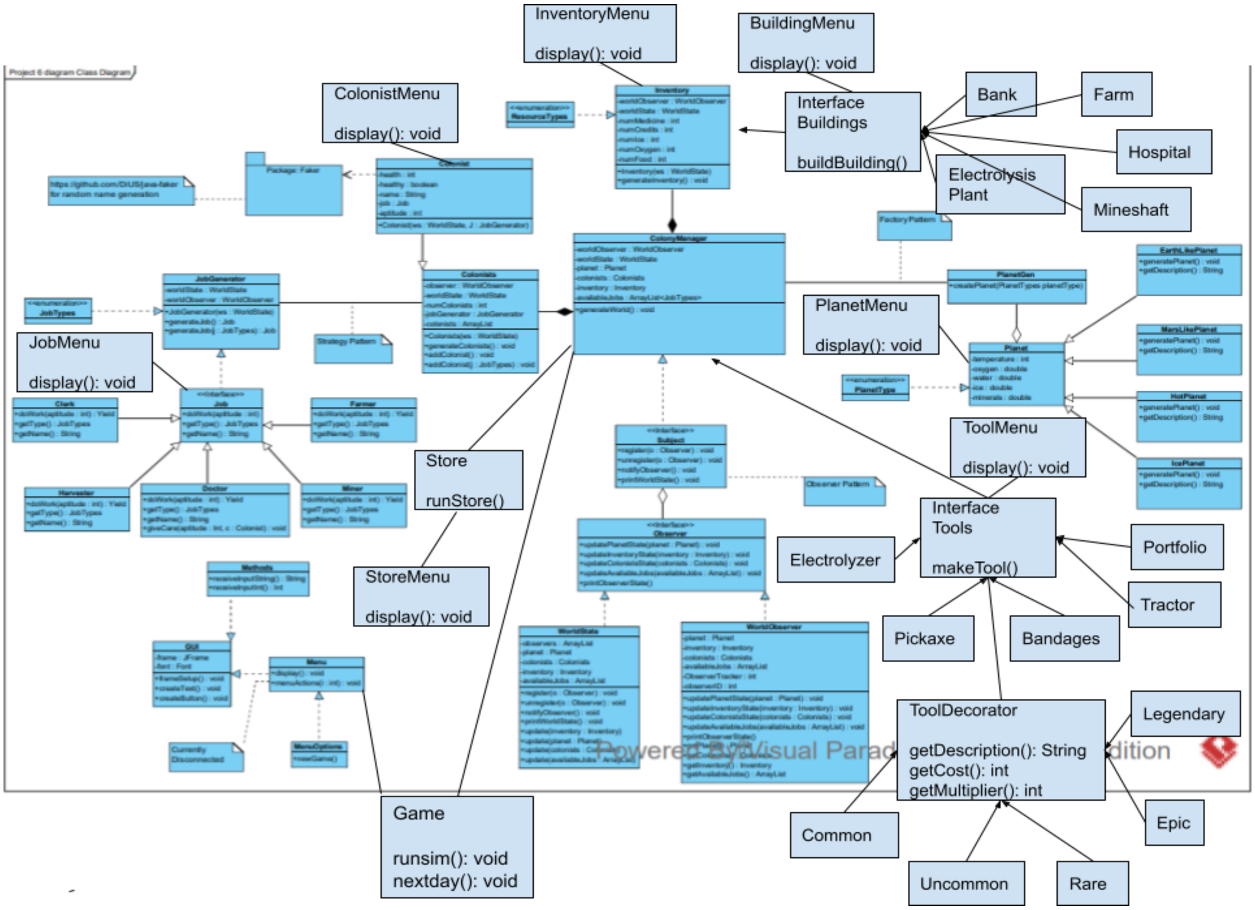
Tyler Paik and Joseph Davis

Project 7 Report

Planet Colony Game

Since Project 6, a lot has changed, and we have had to completely rework our scope over the week. We originally had plans to make a frame GUI for it to be played on, but we had to scale back to a console print game. The different parts of the game, such as the buildings, tools, colonists, planets, menus, and more are all completely finished, but they aren't tied together. We have used patterns in situations where they were useful, and no new ones have been added since project 6. Jobs, Buildings, and Tools are implemented with Strategy Patterns, Tool Efficiencies are implemented with a Decorator Pattern, World State is handled with an Observer Pattern, and Planets are created with a Factory Pattern.


Project 6 diagram Class Diagram
<<enumeration>>
ResourceTypes
Inventory
-worldObserver : WorldObserver
-worldState : WorldState
-numMedicine : int
-numCredits : int
-numIce : int
-numOxygen : int
-numFood : int
+Inventory(ws : WorldState)
+generateInventory() : void
https://github.com/DiUS/java-faker
for random name generation
Package: Faker
Colonist
-health : int
-healthy : boolean
-name : String
-job : Job
-aptitude : int
+Colonist(ws : WorldState, J : JobGenerator)
Factory Pattern
ColonyManager
-worldObserver : WorldObserver
-worldState : WorldState
-planet : Planet
-colonists : Colonists
-inventory : Inventory
-availableJobs : ArrayList<JobTypes>
+generateWorld() : void
PlanetGen
+createPlanet(PlanetTypes planetType)
EarthLikePlanet
+generatePlanet() : void
+getDescription() : String
<<enumeration>>
JobTypes
JobGenerator
-worldState : WorldState
-worldObserver : WorldObserver
+JobGenerator(ws : WorldState)
+generateJob() : Job
+generateJob(j : JobTypes) : Job
Strategy Pattern
Colonists
-observer : WorldObserver
-worldState : WorldState
-numColonists : int
-jobGenerator : JobGenerator
-colonists : ArrayList
+Colonists(ws : WorldState)
+generateColonists() : void
+addColonist() : void
+addColonist(j : JobTypes) : void
MarsLikePlanet
+generatePlanet() : void
+getDescription() : String
Planet
-temperature : int
-oxygen : double
-water : double
-ice : double
-minerals : double
<<enumeration>>
PlanetType
HotPlanet
+generatePlanet() : void
+getDescription() : String
Clerk
+doWork(aptitude : int) : Yield
+getType() : JobTypes
+getName() : String
<<Interface>>
Job
+doWork(aptitude : int)
+getType() : JobTypes
+getName() : String
Farmer
+doWork(aptitude : int) : Yield
+getType() : JobTypes
+getName() : String
<<Interface>>
Subject
+register(o : Observer) : void
+unregister(o : Observer) : void
+notifyObserver() : void
+printWorldState() : void
Observer Pattern
IcePlanet
+generatePlanet() : void
+getDescription() : String
Harvester
+doWork(aptitude : int) : Yield
+getType() : JobTypes
+getName() : String
Doctor
+doWork(aptitude : int) : Yield
+getType() : JobTypes
+getName() : String
+giveCare(aptitude : Int, c : Colonist) : void
Miner
+doWork(aptitude : int) : Yield
+getType() : JobTypes
+getName() : String
<<Interface>>
Observer
+updatePlanetState(planet : Planet) : void
+updateInventoryState(inventory : Inventory) : void
+updateColonistsState(colonists : Colonists) : void
+updateAvailableJobs(availableJobs : ArrayList) : void
+printObserverState()
Methods
+receiveInputString() : String
+receiveInputInt() : Int
GUI
-frame : JFrame
-font : Font
+frameSetup() : void
+createText() : void
+createButton() : void
Menu
+display() : void
+menuActions(i : int) : void
Currently
Disconnected
MenuOptions
+newGame()
WorldState
-observers : ArrayList
-planet : Planet
-colonists : Colonists
-inventory : Inventory
-availableJobs : ArrayList
+register(o : Observer) : void
+unregister(o : Observer) : void
+notifyObserver() : void
+printWorldState() : void
+update(inventory : Inventory)
+update(planet : Planet)
+update(colonists : Colonists)
+update(availableJobs : ArrayList)
WorldObserver
-planet : Planet
-inventory : Inventory
-colonists : Colonists
-availableJobs : ArrayList
-ObserverTracker : int
-observerID : int
+updatePlanetState(planet : Planet) : void
+updateInventoryState(inventory : Inventory) : void
+updateColonistsState(colonists : Colonists) : void
+updateAvailableJobs(availableJobs : ArrayList) : void
+printObserverState()
+getPlanet() : Planet
+getColonists() : Colonists
+getInventory() : Inventory
+getAvailableJobs() : ArrayList
Powered By Visual Paradigm Community Edition



As I'm sure you can tell, Joseph was the one to do the UML diagrams, and I don't have the software he used, so I appended a google drawing to the old one, as we only built on it. I removed some details for readability, I know you aren't supposed to but it was impossible to fit with it. Also I don't have access to the project 5 UML diagram, so this is comparing 6 and 7. The main differences were the connection of buildings and tools to the main game, organizational changes to add Game.java, making the store, creating the ToolDecorator for different efficiencies of tool, and making menus for the game display.

The only third party code we used was faker for random name generation. We have linked the original source https://github.com/DiUS/java-faker. We did have a third party source for our GUI, but we didn't implement it. tutorial from https://www.youtube.com/watch?v=bn8MDLsubOQ&ab_channel=RyiSnow

In this project, one main element was pattern recognition, finding places where patterns would be beneficial and figuring out how to cleanly implement them was something we experienced. A problem we encountered was refactoring when we realized we wanted something done a different way after, and having to completely change everything, like when we started the save file code and had to redo the way we retrieved the private variables to account for the need to put it on a text file. A thing I think we did well was creating our initial architecture, as we didn't end up removing or changing much, just adding to it as needed.