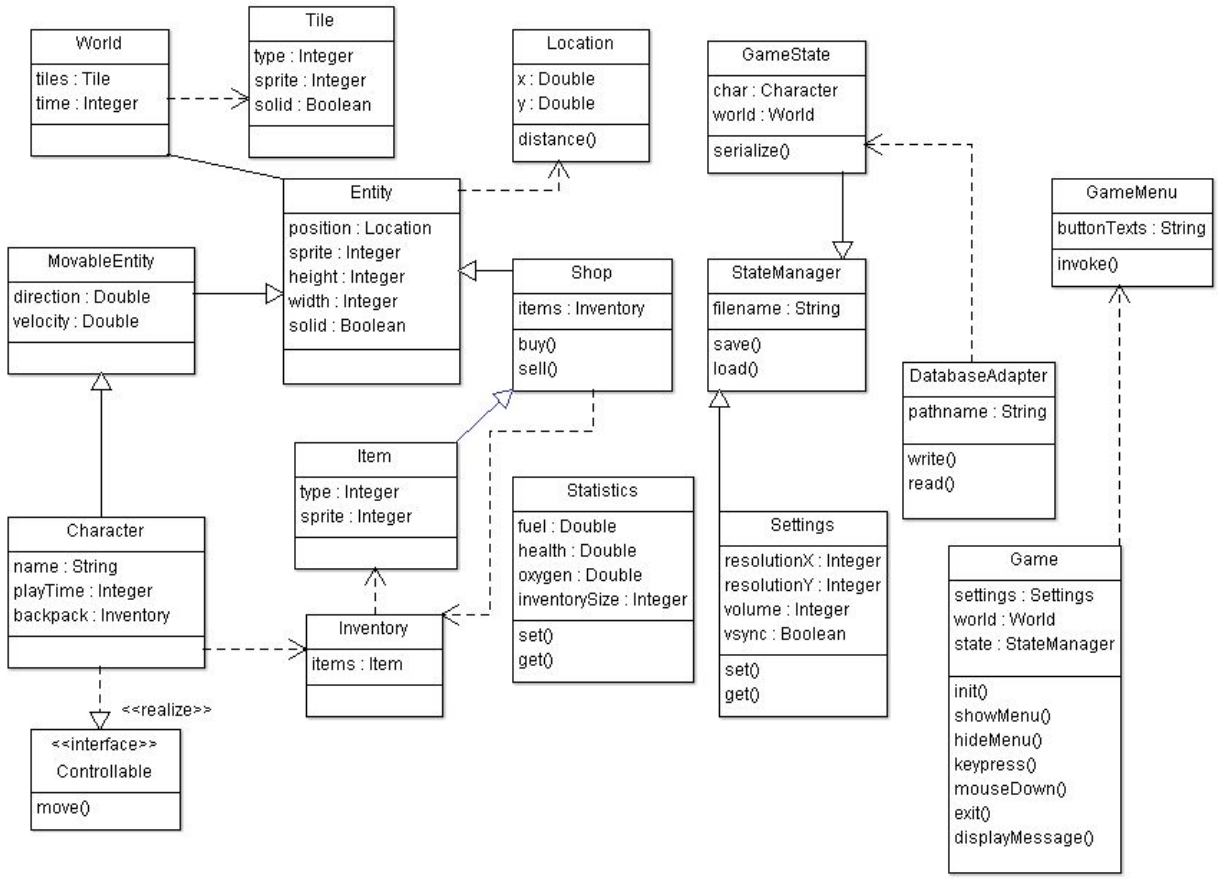
AstroExplorer Part 3:

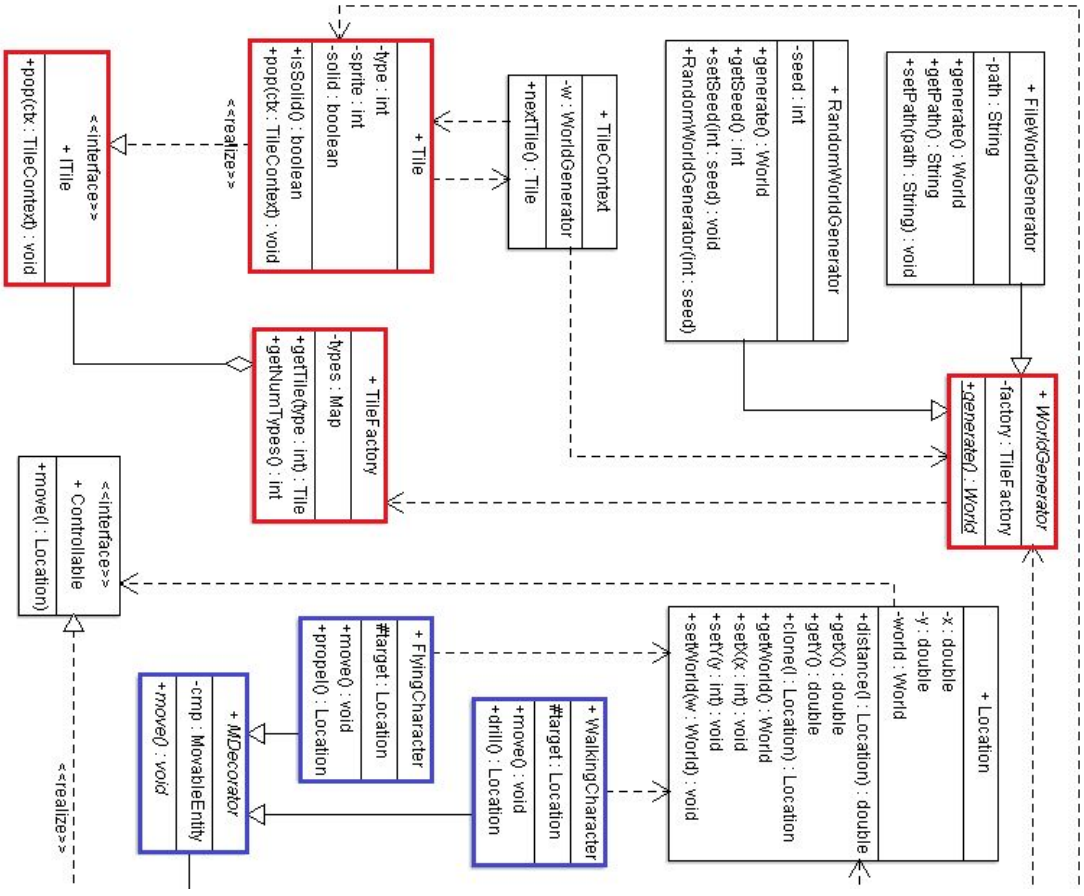
Team:
-David Kleckner
-Evan Su
-Michael Xiao
-David Munson

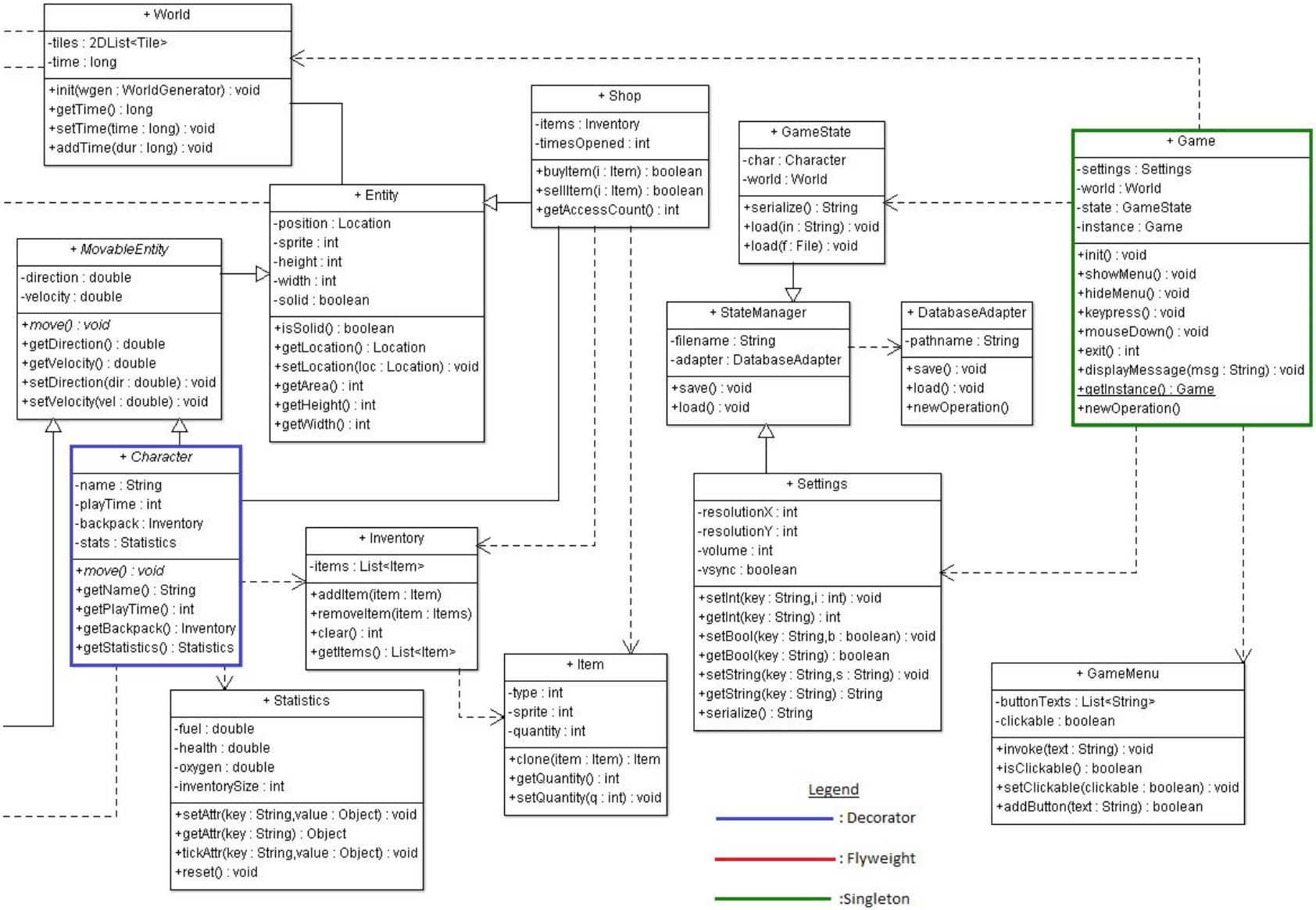
Title: AstroExplorer: Voyage Home

Part 2 Class Diagram:



Part 3 Class Diagram:




+ World
-tiles : 2DList<Tile>
-time : long
+init(wgen : WorldGenerator) : void
+getTime() : long
+setTime(time : long) : void
+addTime(dur : long) : void
+ Shop
-items : Inventory
-timesOpened : int
+buyItem(i : Item) : boolean
+sellItem(i : Item) : boolean
+getAccessCount() : int
+ GameState
-char : Character
-world : World
+serialize() : String
+load(in : String) : void
+load(f : File) : void
+ Game
-settings : Settings
-world : World
-state : GameState
-instance : Game
+init() : void
+showMenu() : void
+hideMenu() : void
+keypress() : void
+mouseDown() : void
+exit() : int
+displayMessage(msg : String) : void
+getInstance() : Game
+newOperation()
+ Entity
-position : Location
-sprite : int
-height : int
-width : int
-solid : boolean
+isSolid() : boolean
+getLocation() : Location
+setLocation(loc : Location) : void
+getArea() : int
+getHeight() : int
+getWidth() : int
+ MovableEntity
-direction : double
-velocity : double
+move() : void
+getDirection() : double
+getVelocity() : double
+setDirection(dir : double) : void
+setVelocity(vel : double) : void
+ StateManager
-filename : String
-adapter : DatabaseAdapter
+save() : void
+load() : void
+ DatabaseAdapter
-pathname : String
+save() : void
+load() : void
+newOperation()
+ Character
-name : String
-playTime : int
-backpack : Inventory
-stats : Statistics
+move() : void
+getName() : String
+getPlayTime() : int
+getBackpack() : Inventory
+getStatistics() : Statistics
+ Settings
-resolutionX : int
-resolutionY : int
-volume : int
-vsync : boolean
+setInt(key : String,i : int) : void
+getInt(key : String) : int
+setBool(key : String,b : boolean) : void
+getBool(key : String) : boolean
+setString(key : String,s : String) : void
+getString(key : String) : String
+serialize() : String
+ Inventory
-items : List<Item>
+addItem(item : Item)
+removeItem(item : Items)
+clear() : int
+getItems() : List<Item>
+ Item
-type : int
-sprite : int
-quantity : int
+clone(item : Item) : Item
+getQuantity() : int
+setQuantity(q : int) : void
+ Statistics
-fuel : double
-health : double
-oxygen : double
-inventorySize : int
+setAttr(key : String,value : Object) : void
+getAttr(key : String) : Object
+tickAttr(key : String,value : Object) : void
+reset() : void
+ GameMenu
-buttonTexts : List<String>
-clickable : boolean
+invoke(text : String) : void
+isClickable() : boolean
+setClickable(clickable : boolean) : void
+addButton(text : String) : boolean
Legend
: Decorator
: Flyweight
:Singleton

Applied Changes:

- `Game` -> singleton
- `Character` -> decorator
- `Tile` -> flyweight

Decorator Pattern: The implementation of the `Character` class and `Movable` interface have been modified by applying the decorator design pattern. In doing so, we are now able to dynamically add and remove behaviors to objects that implement the `Movable` specification without generating a large class hierarchy for different type of movements.

Participants:

- Component: `Movable`
- ConcreteComponent: `Character`
- Decorator: `MDecorator`
- ConcreteDecorator: `FlyingCharacter`, `WalkingCharacter`

Flyweight: We have modified the implementation of `Tile` objects to utilize the flyweight pattern. Since the generation of a player environment often involves recycling `Tile` types, we have specified a flyweight factory for their generation. This permits us to aggregate a large number of `Tile` objects during a world generation event, while making use of redundant objects to minimize memory use.

Participants:

- Flyweight: `ITile`
- ConcreteFlyweight: `Tile`
- UnsharedConcreteFlyweight (flyweights that
- FlyweightFactory: `TileFactory`
- Client: `WorldGenerator`

Intrinsic State:

- `type`
- `sprite`
- `solid`

Extrinsic State:

- `TileContext`

Singleton: We have applied the singleton pattern to the `Game` class, because it is only possible for a singular game session to be running at any point in time. Any attempts to obtain a reference to a `Game` will return the game session that is currently active, as to reduce any complexity that may arise from the existence of multiple `Game` objects.

Participants:

- Singleton: `Game`