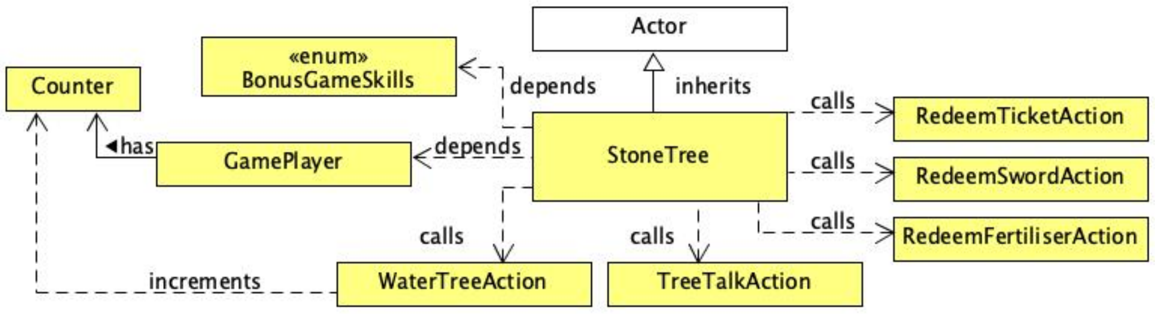
Bonus marks documentation
Background

1. There’s a magical stonetree added to the bonus game which is an entity that adds stones into the player’s stone counter. It also has the ability to talk on every turn.
2. In order for the tree to create a stone, the player has to collect water in a bucket and water the tree.
3. Then, there is a 50% chance that a stone will be added into the player’s stone counter.
4. When the player has these number of stones they can redeem the following from the stone tree:
    - 2 stones = Redeem a ticket for intra-world teleportation. Can teleport back and forth from the locked room to get the rocket plans without the key once.
    - 3 stones = Fertilizer to level up the tree to spawn two stones after watering.
    - 4 stones = A weapon with higher damage to attack the enemies and mini bosses.

New classes added
<u>Class StoneTree</u>
```
public class StoneTree extends
Actor
```



Based on the class diagram above, the StoneTree class inherits from Actor. Its getAllowableActions method depends on the GamePlayer’s inventory and the BonusGameSkills enum. Then it calls the WaterTreeAction, RedeemFertiliserAction, RedeemTicketAction and RedeemFertiliserAction. The StoneTree class uses the TreeTalkAction in its playTurn method. GamePlayer has an Counter attribute, stoneCounter. The WaterTreeAction increments player’s stone counter to give the impression that stones are added to player’s inventory.

This class represents the stone tree entity in the bonus game. In its overridden playTurn method, it calls for the TreeTalkAction on every turn to give it the ability to talk. Then, in its overridden getAllowableActions method,

1. If the player has a filled bucket of water, it adds a newly instantiated WaterTreeAction object into the List of Actions so that the player can water the tree.
2. If the stone tree has not been fertilised, it adds a newly instantiated RedeemFertiliserAction object into the List of Actions so that the player can fertilise the tree to create two stones.
3. Newly instantiated RedeemTicketAction and RedeemSwordAction are always added into the List of Actions. Hence, the player can redeem tickets for intra-world teleportation or redeem a sword.

<u>Class TreeTalkAction</u>
```
public class TreeTalkAction extends TalkAction
```

This class represents the Action for the tree to talk. In its overridden execute method, it randomly returns a String literal that represents the tree’s dialogue.

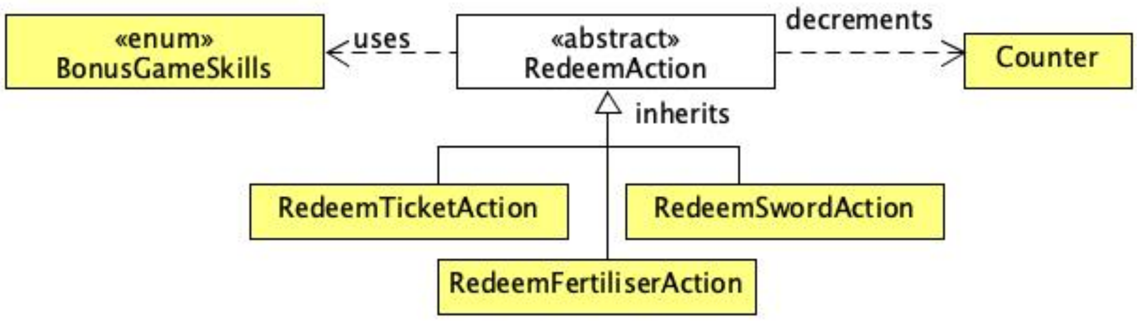
Class WaterTreeAction
`public class WaterTreeAction extends Action`

This class represents the Action of watering the stone tree. In its overridden execute method, there is a 50% chance that player's stoneCounter will be increment by 1. If the stone tree is fertilised, there is a 50% chance that player's stoneCounter will be increment by 2. The bucket will be emptied after executing this Action.

Class FillBucketAction
`public class FillBucketAction extends Action`

This class represents the Action of filling the empty bucket. It is called in the Water class' allowableActions method if the player has a bucket in their inventory so that the player has an option to fill their empty bucket if they're adjacent to the water terrain.

Class RedeemAction
`public abstract class RedeemAction extends Action`



Based on the class diagram above, the RedeemFertiliserAction, RedeemTicketAction and RedeemFertiliserAction inherit from RedeemAction which is an abstract class. The RedeemAction class uses the Counter class to check if the Actor has sufficient stones to redeem the item and decrements the counter if the Actor has sufficient stones and chooses to redeem an item. The RedeemAction class also uses the BonusGameSkills enum as the key of the HashMap attribute.

This class provides methods for its subclasses to redeem items. It has been declared as abstract so that its subclass have to implement its superclass' methods. It has a HashMap which stores the BonusGameSkills (of the Item) and its stone value. Then, it has methods addItemValue which its subclass calls to add the item and its stone value. It also has a method checkSufficientStones which compares the stone "price" of the item to be redeemed and the amount of stones in the player's stone counter. Lastly, it has a method redeem which reduces the player's stone counter by the stone value of the item.

Class RedeemTicketAction
`public class RedeemTicketAction extends RedeemAction`

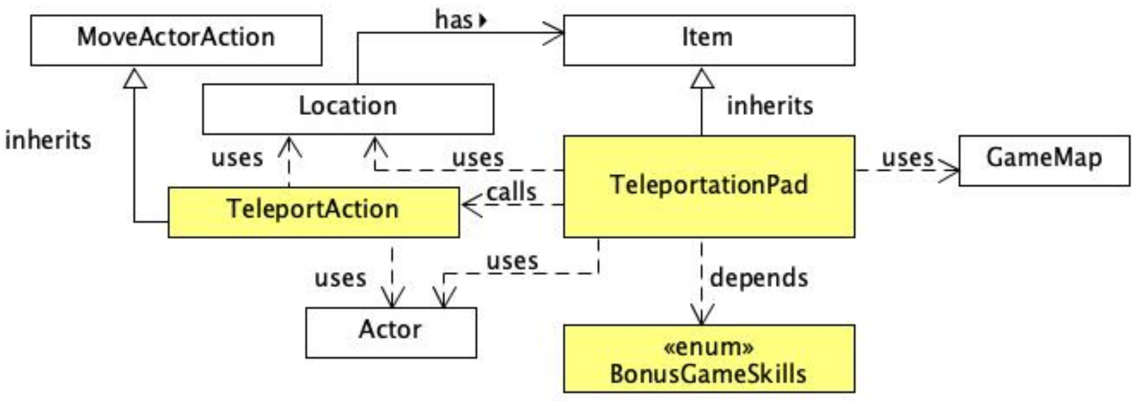
This class is responsible for the Action of redeeming a ticket for intra-world teleportation. If the player has sufficient amount of stones, two tickets will be added into the player's inventory for a trip to and from the locked room with the rocket plans.

Class RedeemFertiliserAction
`public class RedeemFertiliserAction extends RedeemAction`

This class is responsible for the Action of adding fertiliser to the stone tree. If the player has sufficient amount of stones, the stone tree will be fertilised. Thus the stone tree will be adding two stones into the player's stone Counter after being watered from next round onwards.

Class RedeemSwordAction
```
public class RedeemSwordAction extends RedeemAction
```

This class is responsible for the Action of redeeming a sword. If the player has sufficient amount of stones, the WeaponItem sword will be dropped in the Location of the player and the player has to pick it up to be added into their inventory.



Based on the class diagram above, the TeleportationPad inherits from Item and is instantiated as a newFurnitureItem. It depends on the BonusGameSkills enum and calls the TeleportAction if the Actor has BonusGameSkills.TICKET. It also uses the GameMap class in its addTeleportationPadToMap method. The TeleportAction inherits from MoveActorAction and uses the Location class to move the Actor. It also has an Item attribute which is the ticket that is to be removed from the Actor's inventory.

Class TeleportationPad
```
public class TeleportationPad extends Item
```

This class represents the teleportation pad that the actor can stand on to teleport. If the actor has a ticket in their inventory with BonusGameSkills.TICKET, its getAllowableActions method adds the TeleportAction into a List of Actions that can be performed by the actor when they're on the teleportation pad.

Class TeleportAction
```
public class TeleportAction extends Action
```

This class is responsible for the Action of teleporting the actor. When this class is called, it moves the player from the teleportation pad out of the locked room or into the locked room. Then, it removes a ticket from the Actor for each trip.

Enum BonusGameSkills
```
public enum BonusGameSkills
```

This enum represents skills that a bonus game item may possess.

Additions to existing classes
Class GamePlayer

This is an existing class. A Counter attribute has been added to keep track of the number of stones that the player has.

Class Application

The Item bucket with BonusGameSkills.BUCKETISEMPTY, two TeleportationPad objects and a StoneTree object are instantiated on the Earth map.