Farming Simulator (Prototype)

A Resource management game where you can grow resources to build upon and expand your farm. (*Crops are referred as resources for this project for management purposes*)

How To Play

- Play Main Menu Scene
- Click Start Game Button
- Ready to play

Gameplay Loop

- Natural resources are generated by nature at specific intervals (i.e., water).
- Based on natural resources you can grow more man grown resources, which again can help you to grow different type of resources.
- Manage your resources carefully and expand upon.

Developers Approach

First thing I did is, getting the basic requirements down (i.e., 3 lands which can be harvested simultaneously, Main Menu, Sound System). Once I got rough idea of what type of classes I will be needing, I made rough sketch of all the main managers and subclasses. Till this point, game used bare minimum, basic cubes provided by unity as a visual place holder.

After this, I got down to code the system, made changes on the fly on the architecture, as per requirements. Did not added any complex design patterns where it was not needed for e.g., Player Controller because player controller was not the focus of this project, instead I made very generic easy and fast to understand script using Character Controller.

Added Visual elements and models to game using Unity's free assets. Also, added some juice to the game for e.g., whenever resource is ready to harvest a little scaling effect takes place on respective land. Added background music according to the feel of the game.

Design Architecture

List of scripts

- GameManager.cs (Singleton)
- LevelManager.cs (Singleton)
- SoundManager.cs (Singleton)
- LandController.cs (MVC) (Land.cs and LandView.cs serving as Model and View respectively)
- Land.cs
- LandView.cs
- Inventory.cs
- Resource.cs (Scriptable Object)
- ResourceController.cs
- NaturalResource.cs
- PlayerController.cs
- MessageDisplay.cs

Overview of each script
(Main Logic is in LandController.cs and it's MVC scripts)

GameManager.cs – Responsible for maintaining and managing overall state of the game. Has states like Start, Initialize, Playing, Paused, Resume. Upon any change in state of the game, an Event is fired, which helps other components to act accordingly.

LevelManager.cs – Responsible for loading and changing of scenes as per requirements.

SoundManager.cs – Uses Unity's Sound Mixer to simultaneously playing multiple sounds. Sound Mixer has two tracks for now (Background and UI). What clip needs to played on what action, that is passed to Sound Manager and in turn sound manager plays that specific clip on desired track.

Inventory.cs – Stores current amount of resources present, which are accessed by classes depended on this

LandController.cs (MVC and Object pooling) (View – LandView.cs) (Model – Land.cs)
Responsible for managing and updating particular land info on user. For.eg. how many crops are ready to harvest, how many speeds are there, what type of resource is needed to grow more crops.

Resource.cs (Scriptable object) – Holds data like, Type of resource, Name of resource, prefab gameobject which is Instantiated.

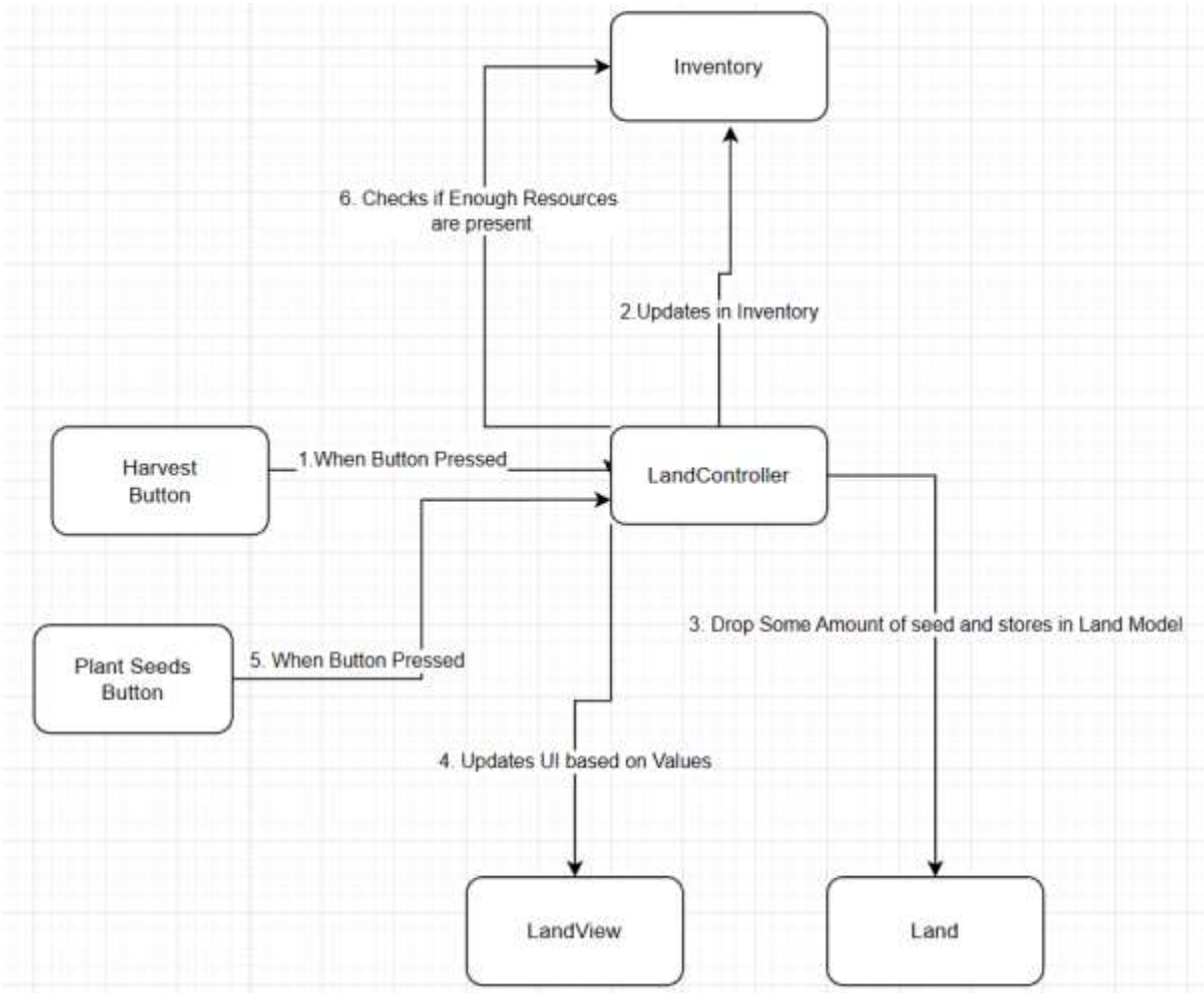
ResourceController.cs – Manages growing of resource over the period and when ready to harvest, let's his LandController know.

NaturalResource.cs – Grows specific resource naturally at regular interval, which are again, updated in inventory at runtime.

PlayerController.cs – Movement of player is handled by this.

MessageDisplay.cs – Any popup as per the requirements is displayed by this class.

Working of Landcontroller and MVC



Conclusion

Here by, I completed the assignment as per the requirements. Used required design patterns and implemented single responsibility principal. Adding new types of land and new type to both natural and man-made resources can be done with few little steps and code base is fairly expandable as per the requirements.

Thanks.