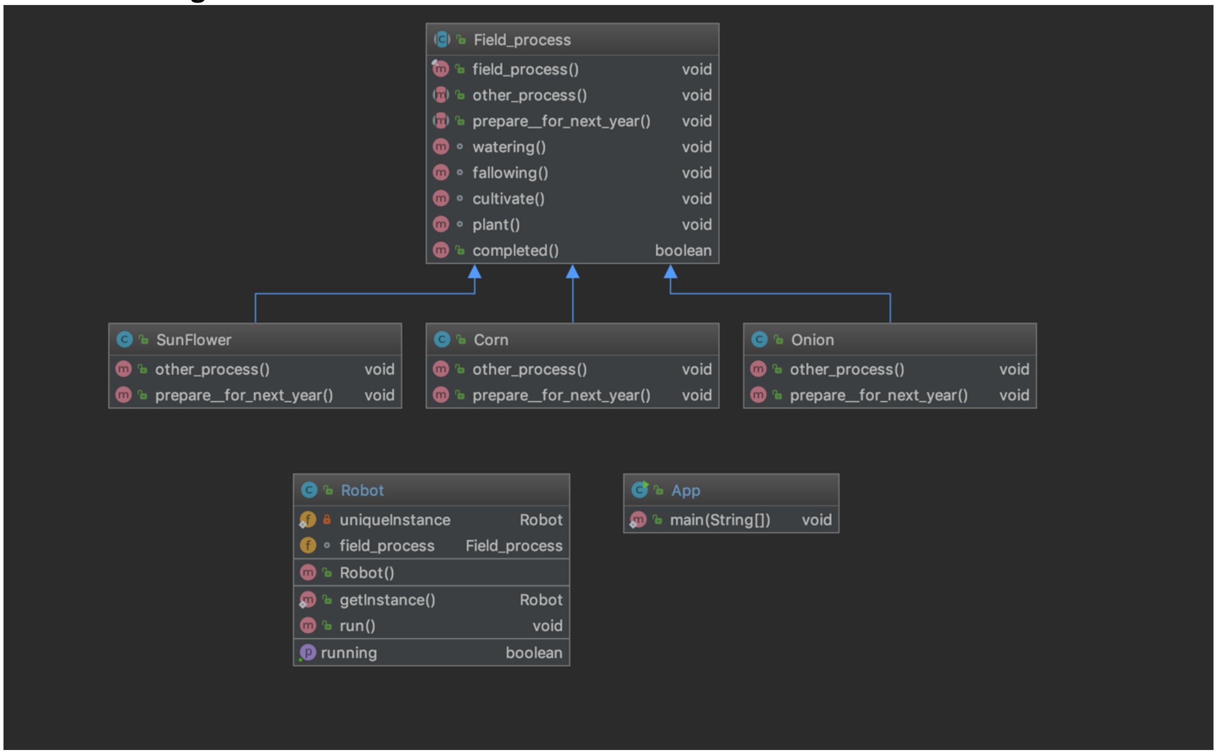
1) Statement ofWork

We have farm. The farm's owners working with a robot. The robot helps farmer's plan. Every farmer should have one robot. The robot specified farmer's plans. Farmer's fields should be specified by farmer. The robot and fields connected each other. The robot's real job is starting a to do list. In to list include watering, cultivate, other process, fallowing, prepare for next year and plant the fields. These processes should be respectively. Firstly to the degree of field's situation watering, then cultivate the field, then if it is necessary something like disinfection should be other processes, then again watering and fallowing, for next year there are two processes also there are ; prepare for next year in that time defined the planting plani and last process is plant. When robot is running these processes will start doing, and when field's processes will be done robot will be stopping.

2) Explanation on Utilized Design Patterns

I choose Singleton Pattern and Template Pattern. The robot is a singleton object I need to create this one time. And I need also to use Template Pattern. My working shold be respectively, sequent method's and these methods shouldn't be change and I have two methods also but these methods can change in other class so this pattern was useful for me.

3) UML Class Diagram



4) Research

The application has Robot class. This class include Singleton object. I created this object for being global Access point. When application is starting only single robot object should be there so we just one object created. I adapt singleton object and other classes' methods.
There is Field_process class. In ths class one final method; final void field_process, in this method other methods defined respectively, we don't want to change the methods place so it is final. Also in this class has two abstract methods; other_process and prepare_for_next_year methods, these method can be change by other class, and there are four class also, this class can not change by other class, because these are constant works, they are using each field's working. There is one boolean method to check the process complated or not if all process completed this method is stop the robot.
There are three field class, these are extends field_process class. Because of the extends, they implement two abstract method. Each method depends on field's situation and according to field these change by farmer.
When all process is completed or not is check by completed method, if it is true robot will stop it's work.