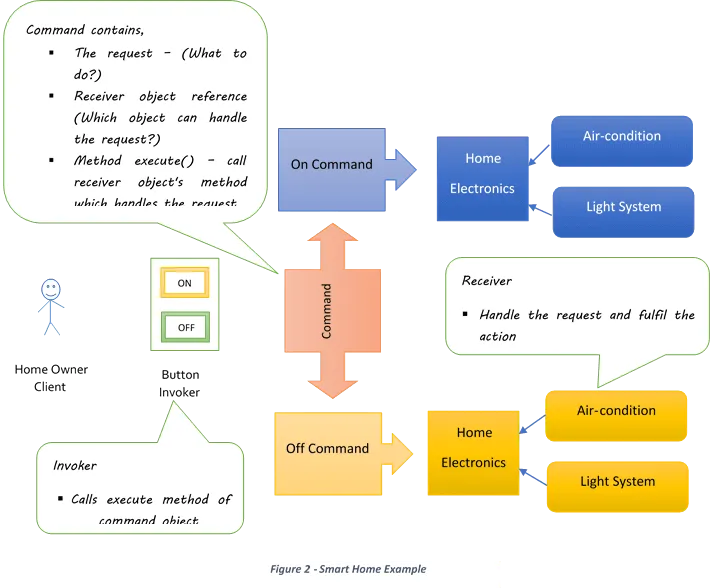
Exercise:

Let’s assume a smart home electrical unit system that controls by a universal remote. There are two kinds of systems, which are operated by the universal remote. Those two are the living room light system and Air Conditioner.  We will include those two sub-systems into a super system as ‘Home Electronics’. Universal remote has two buttons as ‘On’ and ‘Off’. When the homeowner press ‘On’ both the light system and air-conditioned activates.

Let’s apply the command pattern to this scenario:

* Command: Command interface to initiate the execution and abstract the concrete commands.
* OnCommand: Concrete command class that implements the ‘On’ operations for the ‘HomeElectronics’
* OffCommand: Concrete command class that implements the ‘Off’ operations for the ‘HomeElectronics’
* UniversalRemote: UniversalRemote class will maintain all the receivers and will provide a method to identify the currently active receiver
* HomeElectronics: Parent class for ‘LightSystem’ and ‘Airconditioner’ home electronic systems
* LightSystemReceiver: Act as a ‘Receiver’ object for the scenario, contains actual method implementations to execute the required functions
* AirconditionReceiver: Act as the ‘Receiver’ object for the scenario, contains actual method implementations to execute the required functions
* ButtonInvoker: This class act as the invoker for any action
* HomeOwnerClient: The client who needs the service of receivers

High level diagram:



Add the missing interfaces, libraries and check errors to the Command exercise in this repository to get the complete solution and see how this pattern works.

Check the complete solution at:

[Command Pattern with Java examples - Java Code Gists (javagists.com)](https://www.javagists.com/command-pattern-java-examples)