Remote Control API Design for Home Automation or Bust, Inc.,

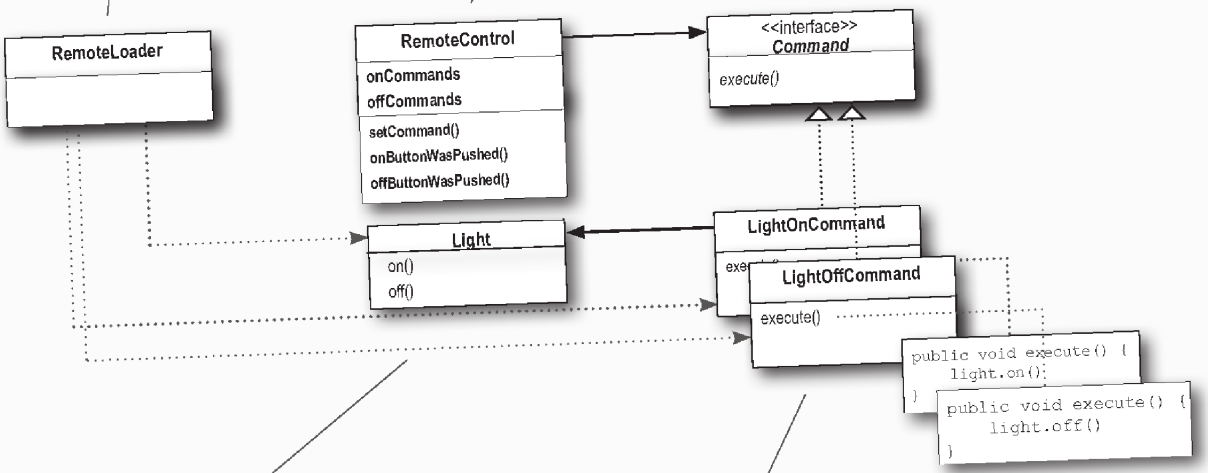
We are pleased to present you with the following design and application programming interface for your Home Automation Remote Control. Our primary design goal was to keep the remote control code as simple as possible so that it doesn't require changes as new vendor classes are produced. To this end we have employed the Command Pattern to logically decouple the RemoteControl class from the Vendor Classes. We believe this will reduce the cost of producing the remote as well as drastically reduce your ongoing maintenance costs.

The following class diagram provides an overview of our design:

The RemoteLoader creates a number of Command Objects that are loaded into the slots of the Remote Control. Each command object encapsulates a request of a home automation device.

The RemoteControl manages a set of Command objects, one per button. When a button is pressed, the corresponding ButtonWasPushed() method is called, which invokes the execute() method on the command. That is the full extent of the remote's knowledge of the classes it's invoking as the Command object decouples the remote from the classes doing the actual home-automation work.

All RemoteControl commands implement the Command interface, which consists of one method: execute(). Commands encapsulate a set of actions on a specific vendor class. The remote invokes these actions by calling the execute() method.



The Vendor Classes are used to perform the actual home-automation work of controlling devices. Here, we are using the Light class as an example.

Using the Command Interface, each action that can be invoked by pressing a button on the remote is implemented with a simple Command object. The Command Object holds a reference to an object that is an instance of a Vendor Class and implements an execute method that calls one or more methods on that object. Here we show two such classes that turn a light on and off, respectively.