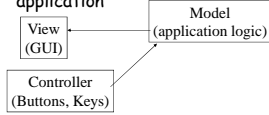


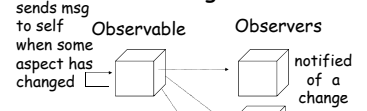
The Observer Pattern

Based on Model View Controller (MVC)

- The earliest design pattern - Smalltalk80
- Goal: separate functionality within an application

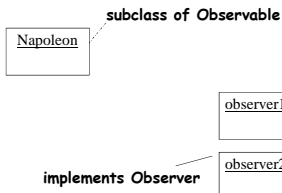


Observer Design Pattern

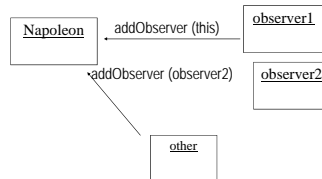


Advantage: Observer does not need to maintain a reference to Observables

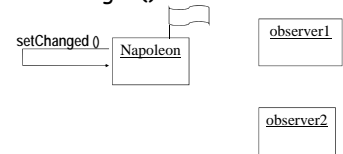
Players



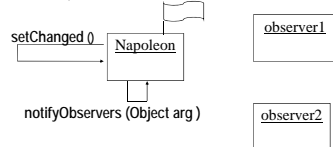
Step 1: Observers register with Observable



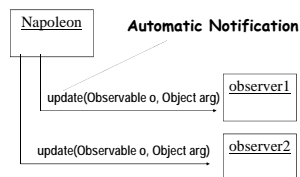
Step 2. Observable sends itself setChanged()



Step 3. Observable sends itself notifyObservers()



Step 4. Notification



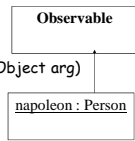
java.util.Observable

- The superclass of all 'observable' objects to be used in the Model View design pattern
- Methods are provided to:
 - void addObserver(anObserver)
 - int countObservers()
 - void deleteObserver (anObserver)
 - void deleteObservers ()

java.util.Observable

• More Methods:

- void setChanged()
- void clearChanged()
- boolean hasChanged()
- void notifyObservers(Object arg)
- void notifyObservers()



Observer must implement update



Observer is an INTERFACE - with one method:
void update (Observable obj, Object arg)

Java.util.Observer

- Interface
- Specifies the update() method that allows an object to 'observe' subclasses of Observable
- Objects that implement the interface may be passed as parameters in:
 - addObserver(Observer o)