תוכנה 1

תבנית העיצוב Observer

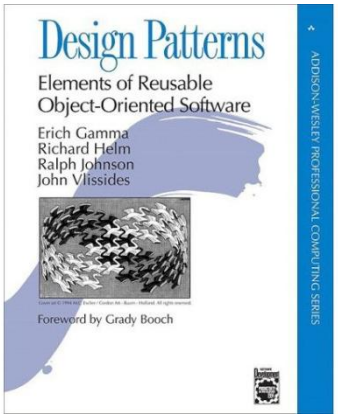
שחר מעוז

בית הספר למדעי המחשב

אוניברסיטת תל אביב

תבניות עיצוב (Design Patterns)

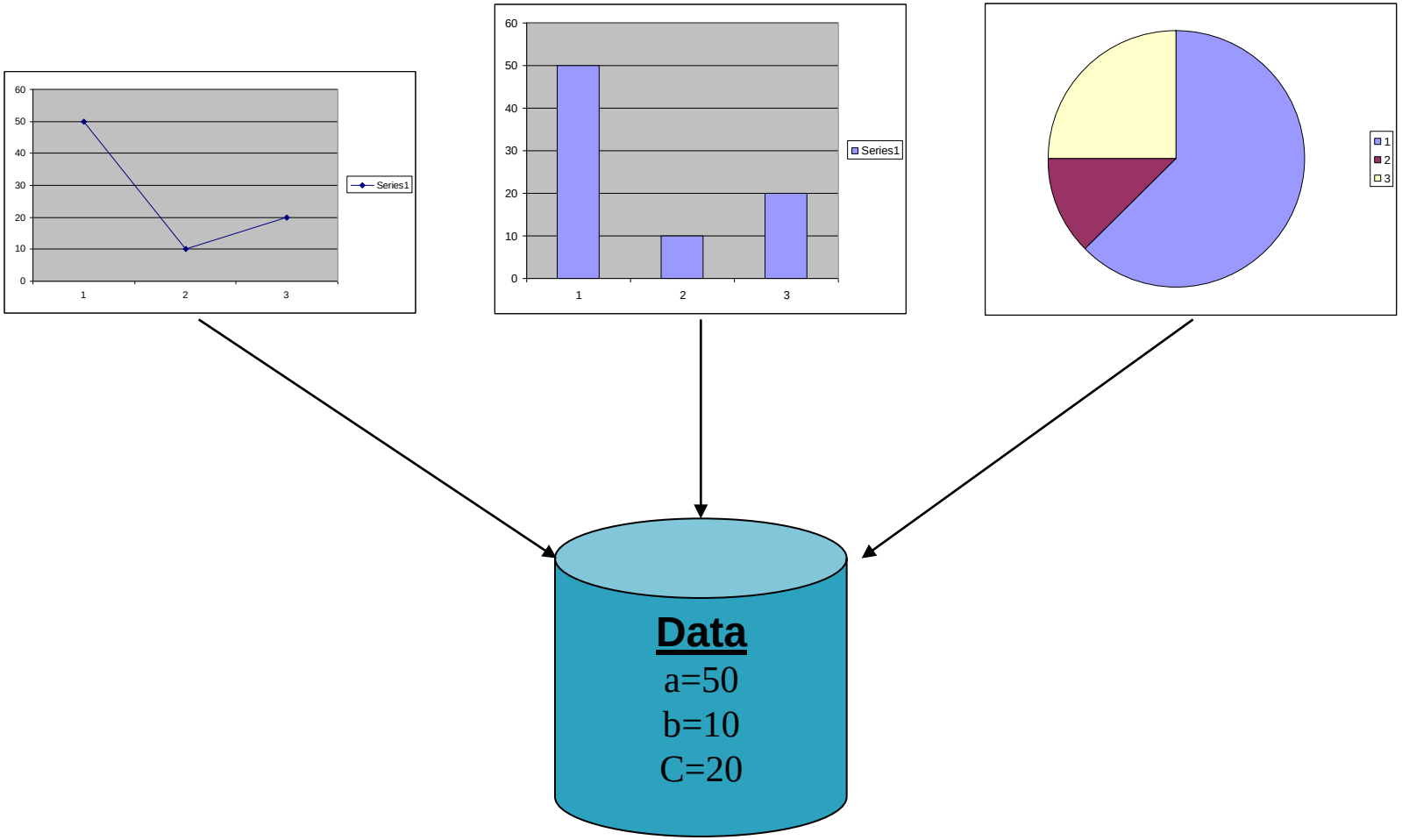
- פתרונות כלליים לבעיות עיצוב שחוזרות על עצמן
- מגדירים שפה כללית יותר לדיון על עיצוב התכנית
  - Factory, Singleton, Observer במקום "המחלקה A יורשת מהמחלקה B"



- **ספר:** *Design Patterns: Elements of Reusable Object-Oriented Software*
- מידע רב בנושא קיים ברשת

- בקורס ראינו כבר מספר תבניות, למשל Factory, Bridge, Adapter, Template method

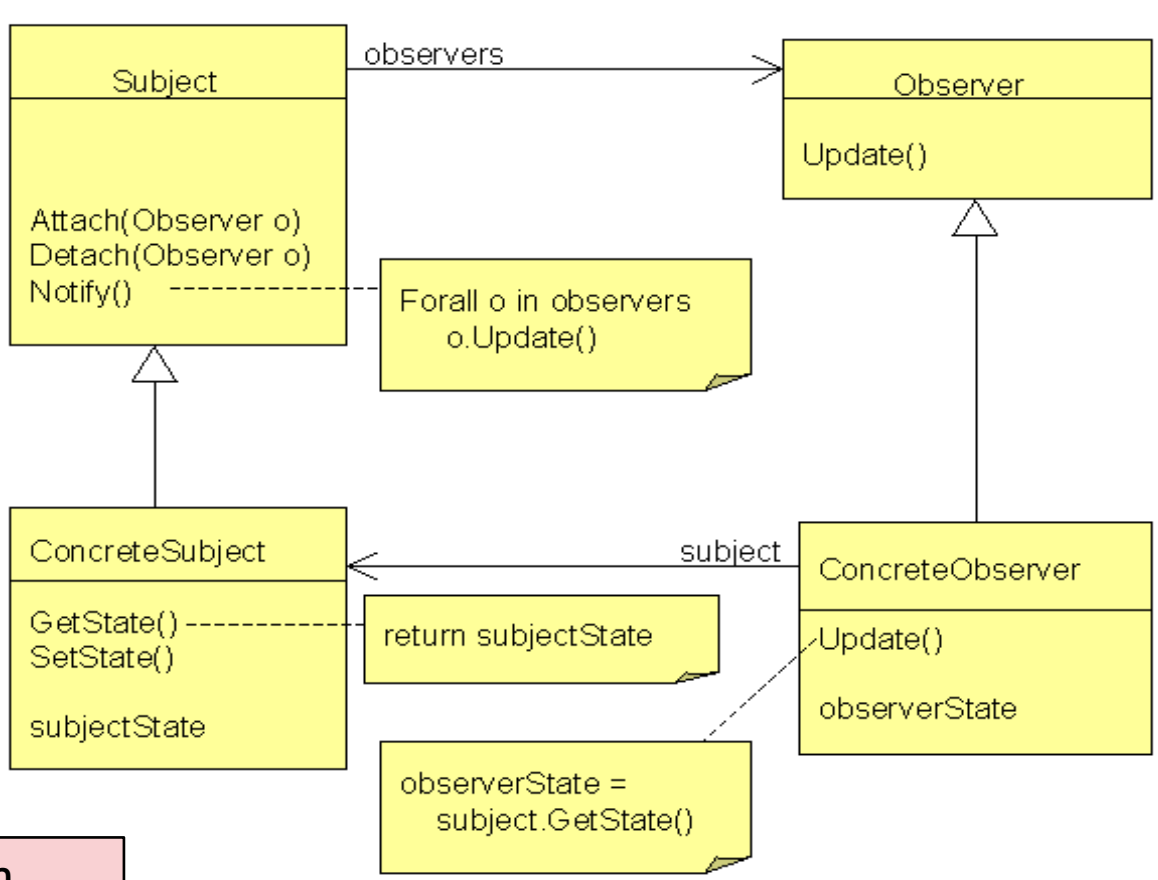
Different Views



Different Views (cont.)

- When the data changes, all views should change
  - Views dependant on data
- Views may vary, more added in the future
- Data store implementation may changes
- We want:
  - Separate the data aspect from the view one
  - Notify views upon change in data

תבנית העיצוב Observer



ניתן לתכנן את
המחלקות גם אחרת!

Observer בג'אווה

- ג'אווה מספקת לנו מנשק Observer, ומחלקה Observable
- נממש את Observer
- כדי ליצור subject, נכתוב מחלקה שיורשת מ-Observable. כבר נתונים לנו
  - הוספה והסרה של Observers
  - מסירת ההודעה ל- Observers הרשומים

Observer

java.util

Interface Observer

`public interface Observer`

A class can implement the `Observer` interface when it wants to be informed of changes in observable objects.

Since:

JDK1.0

See Also:

`Observable`

Method Summary

Methods

Modifier and Type	Method and Description
`void`	`update(Observable o, Object arg)` This method is called whenever the observed object is changed.

Observable

Method Summary

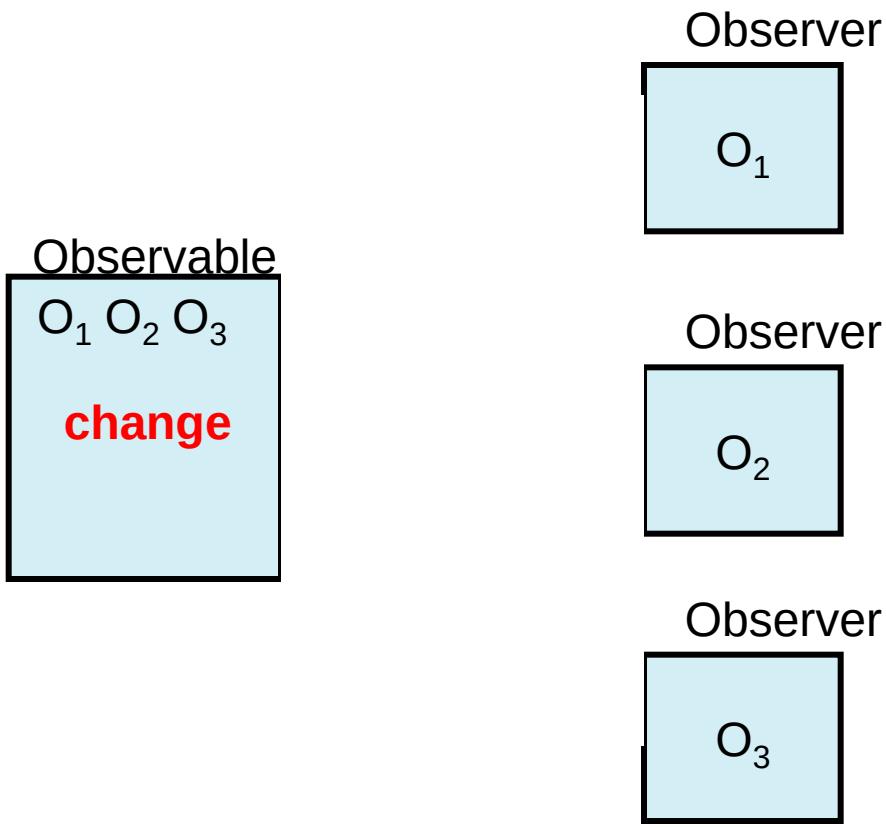
Methods

Modifier and Type	Method and Description
void	addObserver(Observer o) Adds an observer to the set of observers for this object, provided that it is not the same as some observer already in the set.
protected void	clearChanged() Indicates that this object has no longer changed, or that it has already notified all of its observers of its most recent change, so that the hasChanged method will now return false.
int	countObservers() Returns the number of observers of this Observable object.
void	deleteObserver(Observer o) Deletes an observer from the set of observers of this object.
void	deleteObservers() Clears the observer list so that this object no longer has any observers.
boolean	hasChanged() Tests if this object has changed.
void	notifyObservers() If this object has changed, as indicated by the hasChanged method, then notify all of its observers and then call the clearChanged method to indicate that this object has no longer changed.
void	notifyObservers(Object arg) If this object has changed, as indicated by the hasChanged method, then notify all of its observers and then call the clearChanged method to indicate that this object has no longer changed.
protected void	setChanged() Marks this Observable object as having been changed; the hasChanged method will now return true.

Methods inherited from class java.lang.Object

clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Observable and Observer



Example Code - Subject

```
public class IntegerDataBag extends Observable
         implements Iterable<Integer>  {

    private ArrayList<Integer> list = new ArrayList<Integer>();

    public void add( Integer i ) {
        list.add(i);
        setChanged();
        notifyObservers();
    }

    public Iterator<Integer> iterator() {
        return list.iterator();
    }

    public Integer remove( int index ) {
        if( index < list.size() ) {
            Integer i = list.remove( index );
            setChanged();
            notifyObservers();
            return i;
        }
        return null;
    }
}
```

Example Code - Observer

```
public class IntegerAdder implements Observer {

    private IntegerDataBag bag;

    public IntegerAdder( IntegerDataBag bag ) {
        this.bag = bag;
        bag.addObserver( this );
    }

    public void update(Observable o, Object arg) {
        if (o == bag) {
            println("The contents of the IntegerDataBag have changed.");
            int sum = 0;
            for (Integer i : bag) {
                sum += i;
            }
            println("The new sum of the integers is: " + sum);
        }
    }

    ...
}
```