

Der Java Event-Mechanismus



Wie erfährt Objekt a, dass Objekt b sich geändert hat?



Möglichkeit I (Polling)



- Objekt a untersucht den Zustand von b.
- a prüft auch, wenn sich b nicht geändert hat.
- Wie merkt a, **dass** sich b geändert hat?
- die Attribute von b müssen a alle bekannt sein.

3

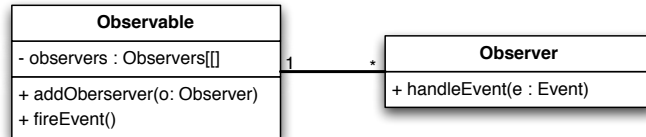
Möglichkeit II (Events)



- Objekt b teilt Objekt a mit, dass sich etwas geändert hat.
- ✓ keine unnötigen Polls (Abfragen)
- ✓ Geheimnisprinzip bleibt erhalten.
- Objekt b muss alle Objekte kennen, die benachrichtigt sein wollen.

4

Das Observer-Muster



5

```
class Observable {
    Observer[] observers;
    ...
```

```
} // Klassendeklaration
```

6

```
class Observable {
    Observer[] observers;
    ...

    public void addObserver(Observer obs) {
        observers[n+1] = obs; //so ähnlich
    }
```

```
} // Klassendeklaration
```

7

```
class Observable {
    Observer[] observers;
    ...

    public void addObserver(Observer obs) {
        observers[n+1] = obs; //so ähnlich
    }

    public void fireEvent() {
        // erzeuge ein neues Event-Objekt
        Event evt = new Event();

        // Benachrichtige alle Observer
        for(int i = 0; i<observers.length(); i++) {
            observers[i].handleEvent(evt);
        }
    } // fireEvent
}
```

```
} // Klassendeklaration
```

8

```

class Observer {

    public void handleEvent(Event e) {
        // Aha, b hat sich geändert.
        // Neue Werte stehen in e.
        // Ich kann darauf reagieren
        // ... und weiß sonst nichts von b
    }

} // Klassendeklaration

```

9

Bezeichnung

- Alle GUI-Elemente sind Observables
- ➔ In Swing kommt der Name Observable **nicht** vor!
- Observerver heißen in Swing Listener

10

Event-Zoo

- Mausclick
- Fenstergröße verändert
- Button gedrückt
- Listenelement ausgewählt
- Maus bewegt
- Taste gedrückt
- Text markiert
- Maustaste gedrückt
- Maustaste losgelassen
- Textfeldeintrag geändert
- Fensterposition geändert
- Maus ist über dem Button
- Maus ist nicht mehr über dem Button
-

11

Event-Zoo

[AWTEvent](#), [BeanContextEvent](#), [CaretEvent](#), [ChangeEvent](#), [ConnectionEvent](#), [DragGestureEvent](#), [DragSourceEvent](#), [DropTargetEvent](#), [FlavorEvent](#), [HandshakeCompletedEvent](#), [HyperlinkEvent](#), [LineEvent](#), [ListDataEvent](#), [ListSelectionEvent](#), [MenuEvent](#), [NamingEvent](#), [NamingExceptionEvent](#), [NodeChangeEvent](#), [Notification](#), [PopupMenuEvent](#), [PreferenceChangeEvent](#), [PrintEvent](#), [PropertyChangeEvent](#), [RowSetEvent](#), [RowSorterEvent](#), [SSLSessionBindingEvent](#), [StatementEvent](#), [TableColumnModelEvent](#), [TableModelEvent](#), [TreeExpansionEvent](#), [TreeModelEvent](#), [TreeSelectionEvent](#), [UndoableEditEvent](#), [UnsolicitedNotificationEvent](#)

```

java.lang.Object
├── java.util.EventObject
│   └── java.awt.AWTEvent
│       ├── java.awt.event.ComponentEvent
│       └── java.awt.event.InputEvent
│           └── java.awt.event.MouseEvent

```

Method Summary

void	mouseClicked (MouseEvent e)	Invoked when the mouse button has been clicked (pressed and released) on a component.
void	mouseEntered (MouseEvent e)	Invoked when the mouse enters a component.
void	mouseExited (MouseEvent e)	Invoked when the mouse exits a component.
void	mousePressed (MouseEvent e)	Invoked when a mouse button has been pressed on a component.
void	mouseReleased (MouseEvent e)	Invoked when a mouse button has been released on a component.

12

Granularität von Events

- Benutzer macht einen Mausklick
- ➔ Events:
 - ▶ mouseClicked(MouseEvent e)
 - ▶ mousePressed(MouseEvent e)
 - ▶ mouseEntered(MouseEvent e)
 - ▶ mouseExited(MouseEvent e)
 - ▶ mouseReleased(MouseEvent e)

13

Fragen?

Wie reagiere ich auf Mauseingaben

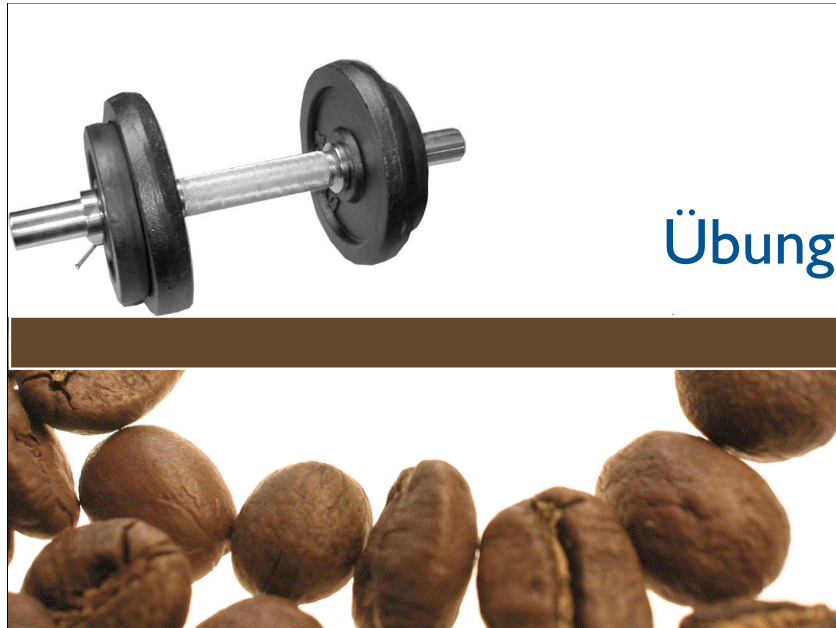
- Die Klasse muss zum `MouseListener` werden.
 - `public class DingsBums implements MouseListener`
- Wir müssen auf unsere eigenen Events hören:
 - `this.addMouseListener(this); // im Konstruktor`
- `MouseListener` haben bestimmte Methoden:
 - ▶ `mouseClicked(MouseEvent e)`
 - ▶ `mousePressed(MouseEvent e)`
 - ▶ `mouseEntered(MouseEvent e)`

15

Wenn man Methoden nicht braucht...

- ... lässt man sie leer
 - `public void mouseEntered(MouseEvent e) {
}`

16



Übung

- Erweitern Sie das einfache Zeichenfenster so, dass
- Sie auf Mausektionen reagieren können.
- Als erste Reaktion auf das Klicken mit der Maus: Schreiben Sie „Huch, ich bin geklickt worden!“ auf die Konsole

