AWT komponedndid

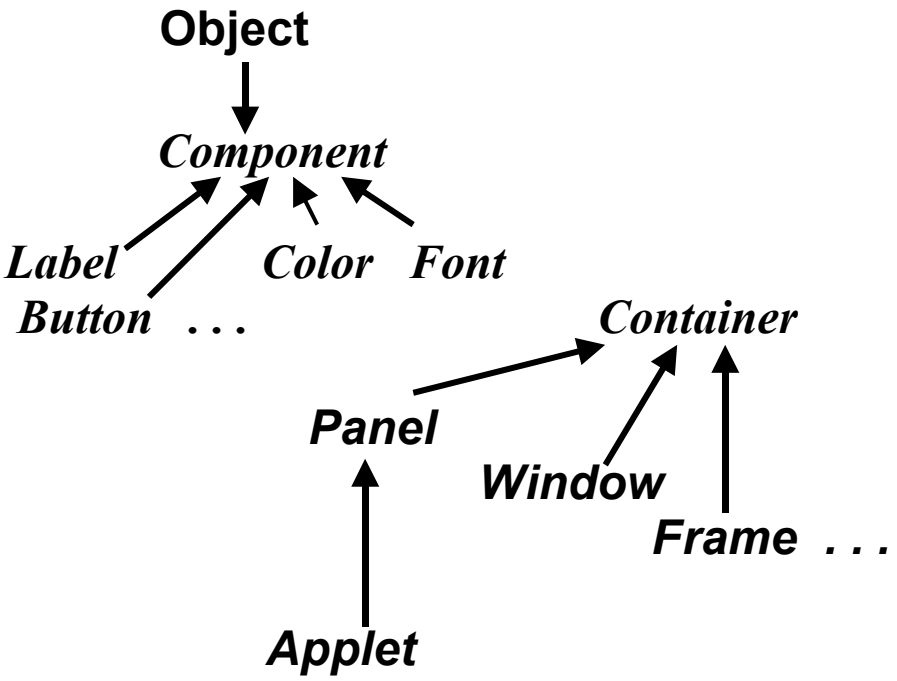
- Komponent on kuvatav objekt
- Komponendid sisaldavad

— nuppe
— tekstivälju ja tekstipiirkondi
— aknaid

- Iga komponendi klassil on:

— konstruktorid
— meetodid
—klassimuutujad
—sündmused

AWT klassihierahia



Sildid — klass Label

- Komponent kujutab endast lihtsat tekstirida
- Sildi tekitamine:

```
Label l0 = new Label ("Nimi:");

// Keskele joondatud silt:
Label l1 = new Label ("Pealkiri",
Label.CENTER);

// Paremale joondatud silt:
Label l2 = new Label ("Position",
Label.RIGHT);
```

- Lisame sildi appleti:

```
public void init ()
{
Label silt = new Label ("Kalender");
add (silt);
}
```

- määrame sildi teksti:

```
silt.setText ("Tere !!!");
```

- määrame fondi:

```
silt.setFont (new Font
("Courier",Font.BOLD, 9));
```

- määrame sildi värvi:

```
// sildi tekst sinine
silt.setForeground (Color.blue);
// sildi taust valge
lab.setBackground(Color.white);
```

Nupud

- Nupp on komponent , mida saab juhtida nupule vajutamisega
- Tekitame nupu:

```
Button nupp = new Button ("Vajuta !");
```

- Lisame apletti

```
public void init ()
{
Button b = new Button ("Stop");
add (b);
}
```

Tegevustega seotud sündmused

- Nupu komponent reageerib ja töötleb hiiresündmusi.
- Kui nuppu vajutatakse ,siis nupp saadab välja sündmuse, mis käivitab sündmuste kuulaja liidese vastava meetodi.

- Nupud suhlevad tegevuste sündmustega läbi oma konteineri või `ActionListener` liidese:

```
public class Nupuapplet extends
Applet
{
Button b1 = new Button("Vajuta !");

public void init() {

b1.addActionListener(new B1L());

add(b1);
}
class B1L implements ActionListener {
public void actionPerformed(
ActionEvent e) {
getAppletContext().

showStatus("!!!!");
}
}
// ...
```

Teksti komponendid

- `TextField` —— väli , mis sisaldab teksirida
- `TextField` objekti saab redigeerida
- Tekitame tekstivälja

```
TextField tf = new TextField (80);

// argument määrab ära tekstirea
```

```
// pikkuse
```

- Teksti paigutamine tekstivälja:

```
tf.setText ("XXXX -- YYYY");
```

- määrame tekstifondi:

```
tf.setFont (new Font ("Courier",
Font.BOLD, 9));
```

- tekstivälja teksti salvestamine stringi

```
String text = tf.getText ();
```

- teksti värv:

```
tf.setForeground(Color.blue);
```

- taustavärv:

```
tf.setBackground(Color.green);
```

- valitud teksti salvestamine stringi:

```
String selection=tf.getSelectedText();
```

- tekstivälja fokusseerimine:

```
tf.requestFocus();
```

- kogu tekst tekstiväljast:

```
tf.selectAll ();
```

- Tekstiväli genereerib tegevuslikke sündmusi (action events) , juhul kui kasutaja on valinud tekstivälja sisestuseks ning vajutab klahvile ENTER.

```
// ...
class TL implements ActionListener {
public void actionPerformed(
ActionEvent e) {
//
// töötlus
}
}
```

Ainult lugemisrežiimis **kasutatavad** tekstiväljad

```
TextField Loetav =
new TextField ("Algväärtus", 40);
readOnly.setEditable (false);
```

- Tahame teada ,kas tekstivälja tohib redigeerida:

```
boolean isEditable = tf.isEditable ();
```

Tekstiaken — klass `TextArea`

- Tekstiaknas on võimalik kuvada mitut tekstirida. Tekstiaknal on kerimisribad.

- Loome **tekstiakna:**

```
int rows = 20;
int columns = 40;
TextArea ta = new TextArea (rows,
columns);
String text =loadTextFromSomewhere ();
ta.setText (text);
```

Tekstiaknad ei genereeri tegevuslikke sündmusi.

Listid

- Listid on kerimisribadega tekstiloendid:

- Loomine:

```
List list = new List ();
list.add ("valge");
list.add ("sinine");
list.add ("punane”);
```

- Neljaelemendilise listi loomine

```
List list2 = new List (4, true);
```

- Valitud elemendi indeksi määramine:

```
int index = list.getSelectedIndex ();
```

—või väljavalimine:

```
String str = list.getSelectedItem ();
```

- Element listist välja:

```
String str = list.getItem (index);
```

- Listist eemaldamine:

```
list.remove (index);
```

Listi sündmused

- elemendi valimise korral genereeritakse `ItemEvent.SELECTED`
- loobumise korral `ItemEvent.DESELECTED`
- topeltklikk listi elemendile genereerib tegevusliku sündmuse (action event) , mida tööteldatakse ActionListener liidese realiseerimisega.
- `ItemEvent.SELECTED` või `ItemEvent.DESELECTED` sündmusi töödeldakse `ItemListener` **liidese realiseerimisega.**

```
public void itemStateChanged (ItemEvent evt)
{
if (evt.getStateChange() == ItemEvent.SELECTED)
{
int index = myList.getSelectedIndex ();
/* ...töötlus... */
}
}
```

Ripploendid

Ripploendid on tekstimenüüd , mis sisaldavad

erinevaid valikuvariante. Mitteaktiivses olekus kuvatakse hüpikloendist ainult viimativalitud element.

- Loomine

```
Choice os = new Choice ();

  os.add("Windows NT");
  os.add("Windows 2000");
  os.add("Solaris");
```

- Elemendi valimine ripploendist `Choice`:

```
int index = choice.getSelectedIndex();
```

  — või:

```
String str = choice.getSelectedItem();
```

- Elemendi lugemine ripploendist:

```
String str = choice.getItem (index);
```

- Elemendi valimisel hüpikloendist genereeritakse sündmus `ItemEvent`.

Märkekast

- Märkekast `Checkbox` laseb kasutajal valida ühe võimaluse või elemendi. Märkekast kujutab kastikest ja silti. Võimaluse valimisel ilmub kastikesse linnnuke.

- Märkekasti loomine:

```
Checkbox cb = new Checkbox
("Gravitatsioon");

cb = new Checkbox ("Kiirus", null,
true);
```

- Valimise korral genereerib märkekast sündmuse `ItemEvent`.

Märkekastide grupid

- Märkekastide grupp on variantide kogum , milledest korraga saab valida ainult ühe võimaluse.

- Erinevad võimalused esitatakse märkekastide grupis raadionuppudena.

- Märkekastide grupi loomine:

```
CheckboxGroup group = new
CheckboxGroup ();
add(new Checkbox("Ruutu",group,
true));
add(new Checkbox("Ärtu", group,
false));
add(new Checkbox("Poti", group,
false));
add (new Checkbox ("Risti", group,
false));
```

- Valitud variandi muutmine:

```
group.setSelectedCheckbox (box1);
```

- Valitud variandi lugemine:

```
Checkbox checked =
group.getSelectedCheckbox ();
```

Kerimisribad

- Kerimisribasid kasutatakse väärtuste valimiseks mingist katkematust vahemikust.

- Kerimisribade atribuudid :

— orientatioon (horisontaalne või vertikaalne)
— väärtus (positsioon)
— dokumendi korraga nähtav osa (lehekülg)

— minimaalne väärtus
— maksimaalne väärtus

- Kerimisriba loomine:

```
Scrollbar sb =
new Scrollbar (Scrollbar.HORIZONTAL,
0 /* algpositsioon */,
1 /* nähtav positsioon */,
0 /* miinimum */,
127 /* maksimum */);
```

- Alternatiivne variant:

```
Scrollbar sb = new Scrollbar
(Scrollbar.VERTICAL);
sb.setValues (0 /* algpositsioon */,
1 /* dokumendi nähtav osa */,
0 /* miinimum */,
127 /* maksimum */);
```

- Krollimisinkremendi määramine, kui kasutaja klikib noolel:

```
sb.setUnitIncrement(2);/*vaikimisi 1*/
```

- Krollimisinkremendi määramine , kui kasutaja klikib kerimisribal:

```
sb.setBlockIncrement(20);/*vaikimisi
10*/
```

- Jooksva väärtuse lugemine:

```
int value = sb.getValue ();
```

Kerimisriba sündmused:

- Kui kasutaja muudab kerimisribal positsiooni, siis kerimisriba genereerib sündmuse `AdjustmentEvent` koos järgmiste variantidega:

```
AdjustmentEvent.UNIT_INCREMENT
AdjustmentEvent.UNIT_deCREMENT
AdjustmentEvent.BLOCK_INCREMENT
AdjustmentEvent.BLOCK_DECREMENT
AdjustmentEvent.TRACK
```

- Sündmuste töötlus:

```
public void adjustmentValueChanged
(AdjustmentEvent evt){
switch (evt.getAdjustmentType ())
{
case AdjustmentEvent.UNIT_INCREMENT:
case AdjustmentEvent.UNIT_DECREMENT:
case AdjustmentEvent.BLOCK_INCREMENT:
case AdjustmentEvent.BLOCK_DECREMENT:
case AdjustmentEvent.TRACK:
```

```
Scrollbar sb =
(Scrollbar)evt.getAdjustable();
int orientation =sb.getOrientation();
int value = sb.getValue ();
  ...
} }
```