

# **Arranging Components on a User Interface**

---

**Pertemuan 14  
Pemrograman Berorientasi Obyek  
Oleh  
Tita Karlita**

# Tampilan Dasar User Interface di Java

---

- Graphical user interface sangat rentan thp perubahan ukuran window.
- Java bisa diimplementasikan di berbagai macam platform.
- Untuk mengontrol layout interface digunakan layout manager.

# Topik hari ini:

---

- Bagaimana cara menggunakan layout manager untuk menyusun komponen di interface.
  - Flow Layout
  - Grid Layout
  - Border Layout
  - Card Layout
  - Grid Bag Layout
- Bagaimana menggunakan berbagai macam layout manager yg berbeda untuk bekerja dalam satu interface.

# Bagaimana cara menggunakan layout manager?

---

- Layout manager hrs dideklarasikan terlebih dahulu sblm komponen ditambahkan ke kontainer.
- Default layout manager untuk panel adalah FlowLayout
- Default layout manager untuk frame, window, dan applet adalah BorderLayout
- Utk membuat layout manager, hrs dibuat instace dgn statement:

```
FlowLayout flo = new FlowLayout();
```

- Agar bisa digunakan maka hrs menambahkan method:

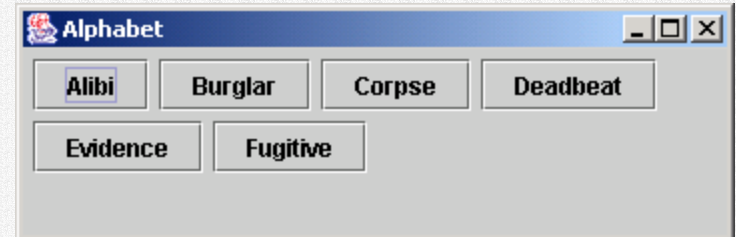
```
setLayout();
```



# Flow Layout

---

- The simplest
- Cara peletakan komponen : dimulai dari kiri ke kanan sampai area penuh dan selanjutnya menuju ke baris dibawahnya.
- Gunakan class variable `FlowLayout.LEFT`, `FlowLayout.CENTER`. atau `FlowLayout.RIGHT` utk mengubah alignment komponen.
- Default : `FlowLayout.CENTER`.
- Contoh mengeset alignment:  
`FlowLayout righty = new FlowLayout(FlowLayout.RIGHT);`



# Flow Layout

---

- Konstruktor FlowLayout:

`FlowLayout(int, int, int);`

- **Alignment** :

- 0 = FlowLayout.LEFT

- 1 = FlowLayout.CENTER

- 2 = FlowLayout.RIGT

- **Horizontal** gap between components in pixels

- **Vertical** gap between components in pixels

- Contoh program:

# Grid Layout



- Menyusun komponen dalam grid of rows and columns
- Tiap cell mempunyai ukuran yang sama
- Ukuran komponen dlm GridLayout di expand untuk memenuhi space yg tersedia utk tiap komponen dlm tiap cell.
- Cara peletakan komponen : dimulai dari baris teratas grid sebelah kiri dilanjutkan ke kanan. Bila baris teratas penuh, penambahan komponen akan diletakkan pada baris selanjutnya dimulai dari sebelah kiri.

# Grid Layout

---

- Cara menggunakan kelas grid layout:  
`GridLayout gr = new GridLayout(10,30,40,40);`
  
- Kontruktor :  
`GridLayout(int, int);`  
`GridLayout(int, int, int, int);`
  - jumlah baris
  - jumlah kolom
  - horizontal gap, dlm pixel
  - vertical gap, dlm pixel
  
- Default gap = 0 pixel
  
- Contoh program:



# Border Layout

- Area dibagi menjadi 5 bagian yaitu : north, south, east, west, and center.
- Komponen di bag north, south, east, dan west akan mengambil area sebanyak yg diperlukan, sisanya akan diberikan ke center.



# Border Layout

---

- Penambahan komponen dilakukan dgn memanggil method `add()`;  
`add(String, component);`

String = lokasi dr border layout tempat meletakkan komponen.

Ada 5 pilihan = north, south, east, west, dan center.

component = komponent yg akan diletakkan ke kontainer.

- Konstuktur:  
`BorderLayout();` --- no gap between any component  
`BorderLayout(int, int);` -- horizontal and vertical gap

- Contoh program:

# Mixing Layout Managers

---

- Menambahkan small container ke main container (frame atau applet) dan menambahkan masing2 layout manager ke tiap small container tsb.
- Small container ini bisa berupa panel dr kelas JPanel.
- Karakteristik panel:
  1. Panel hrs diisi dgn komponen, sblm diletakkan di larger container.
  2. Tiap panel punya layout manager sendiri.

# Card Layout

---

- Menyembunyikan bbrp komponen from view.
- Sekelompok container atau komponen ditampilkan pd suatu waktu tertentu.
- Diperlukan triger agar perubahan terjadi.
- Tiap kontainer dlm satu group disebut *card*.
- Pd umumnya digunakan panel utk tiap card.
- Cara membuat: komponen ditambahkan ke panel, kmdn panel ditambahkan ke kontainer sekaligus mengeset layout manager-nya



# Card Layout

---

- Cara menggunakan kelas card layout:  
*CardLayout cc = new CardLayout();*
- Lakukan set layout dgn memanggil method:  
*setLayout(cc);*
- Contoh penggunaan method add:  
*add("Option Card", options);*  
argumen1 : nama dr card  
argumen2 : nama komponen

# Card Layout

---

- Penggunaan method show:  
*cc.show(this, "Fact Card");*

argumen1: kontainer tempat semua  
card ditambahkan

argumen2: nama card

- Contoh program:

# Grid Bag Layout

---

- Grid Bag Layout vs Grid Layout
  - satu komponen bisa menempati lebih dari satu cell
  - proporsi/ukuran antar baris dan kolom bisa berbeda
  - komponen dlm grid cell dpt disusun dlm cr yang beda
- To create grid bag layout digunakan kelas *GridBagLayout* dan sebuah helper class yang disebut *GridBagConstraints*.

# Grid Bag Layout

- GridBagConstraints digunakan untuk mendefinisikan properti2 tiap komponen yang diletakkan dlm cell, meliputi: Posisi, alignment dll
- Langkah pembuatan grid bag layout dan constraints:
  - Create GridBagLayout object dan mendefinisikannya sebagai current lay out manager
  - Create new instance dr GridBagConstraints
  - Setting up konstrain utk tiap komponen
  - Telling the lay out manager about component and its constraints.
  - Adding komponen ke dalam kontainer



# Designing the Grid

---

- Akan dibuat aplikasi sbb:



A screenshot of a Windows-style dialog box titled "Username and Password". The dialog box has a blue title bar with a small icon on the left and standard minimize, maximize, and close buttons on the right. The main area is light gray and contains two text input fields. The first field is labeled "Name:" and the second is labeled "Password:". Below the input fields is a single "OK" button.

# Designing the Grid

---



The image shows a classic Windows-style dialog box titled "Username and Password". The title bar is blue and contains a small icon on the left and three control buttons (minimize, maximize, close) on the right. The dialog body is divided into three horizontal sections by black lines. The first section has a label "Name:" on the left and an empty text input field on the right. The second section has a label "Password:" on the left and an empty text input field on the right. The third section is a solid gray area containing a single "OK" button centered horizontally.

Username and Password	
Name:	<input type="text"/>
Password:	<input type="text"/>
OK	

# Creating the Grid

---

- Initial layout manager : GridBagLayout dan membuat constraint object:

```
public NamePass() {  
    super("Username and Password");  
    setSize(290,110);  
    GridBagLayout gridbag = new GridBagLayout();  
    GridBagConstraints constraints = new GridBagConstraints();  
    JPanel pane = new JPanel();  
    pane.setLayout(gridbag);  
  
    setContentPane(pane);  
    constraints.fill = GridBagConstraints.NONE;  
}
```

# Creating the Grid

- Cara implementasi desain: Kita buat konstrain untuk tiap komponent dengan menggunakan helper method yaitu *buildConstraints()*.
- *Method buildConstraints():*

```
void buildConstraints(GridBagConstraints gbc, int gx, int gy, int gw,  
int gh, int wx, int wy) {
```

```
    gbc.gridx = gx;  
    gbc.gridy = gy;  
    gbc.gridwidth = gw;  
    gbc.gridheight = gh;  
    gbc.weightx = wx;  
    gbc.weighty = wy;  
}
```



# Creating the Grid

---

- Argumen dlm helper method `buildConstraint()`:

```
buildConstraints(constraints, int gridx, int gridy, int gridwidth , int gridheight  
, int weightx , int weighty);
```

- The first 2 integer are : `gridx` dan `gridy` = merupakan koordinat cell.  
Bila terdapat span multiple cell mk digunakan koordinat cell top-left corner.
- The second 2 integer are: `gridwidth` dan `gridheight` = merupakan jumlah baris dan atau kolom yang di span.  
`gridwidth` utk kolom dan `gridheight` utk baris
- Last 2 integer are: `weightx` dan `weighty` = digunakan untuk menentukan proporsi (ukuran) dr baris dan kolom.

# Creating the Grid

---

- Menambahkan label ke dalam layout:

*// Name label*

```
buildConstraints(constraints, 0, 0, 1, 1, 100, 100);  
JLabel label1 = new JLabel("Name:");  
gridbag.setConstraints(label1, constraints);  
add(label1);
```

- buildConstraints utk komponen lain:

```
buildConstraints(constraints, 1, 0, 1, 1, 100, 100); // Name text field  
buildConstraints(constraints, 0, 1, 1, 1, 100, 100); // password label  
buildConstraints(constraints, 1, 1, 1, 1, 100, 100); // password text field  
buildConstraints(constraints, 0, 2, 2, 1, 100, 100); // OK Button
```

# Determining the Proportions

---

- Digunakan konstraint *weightx* dan *weighty*.

- Sebelum setting

```
buildConstraints(constraints, 0, 0, 1, 1, 100, 100); // Name label  
buildConstraints(constraints, 1, 0, 1, 1, 100, 100); // Name text field  
buildConstraints(constraints, 0, 1, 1, 1, 100, 100); // password label  
buildConstraints(constraints, 1, 1, 1, 1, 100, 100); // password text field  
buildConstraints(constraints, 0, 2, 2, 1, 100, 100); // OK Button
```

- Sesudah setting

```
buildConstraints(constraints, 0, 0, 1, 1, 10, 40); // Name label  
buildConstraints(constraints, 1, 0, 1, 1, 90, 0); // Name text field  
buildConstraints(constraints, 0, 1, 1, 1, 0, 40); // password label  
buildConstraints(constraints, 1, 1, 1, 1, 0, 0); // password text field  
buildConstraints(constraints, 0, 2, 2, 1, 0, 20); // OK Button
```



# Creating the Grid

---

- Setelah membangun konstrain selanjutnya attach them to an object dgn menggunakan method `setConstraint()`

// Name label

```
buildConstraints(constraints, 0, 0, 1, 1, 100, 100);  
JLabel label1 = new JLabel("Name:");  
gridbag.setConstraints(label1, constraints);  
add(label1);
```

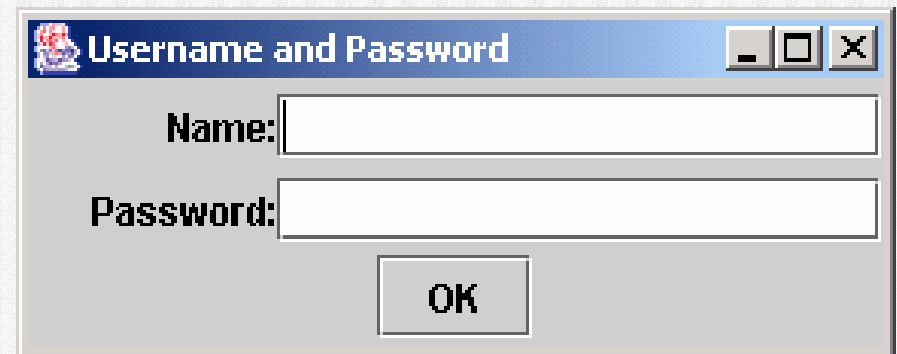


# Adding and Arranging the Components

- Gunakan constraints *fill* dan *anchor* utk mengeset tampilan komponen didalam cell.



A screenshot of a Windows-style dialog box titled "Username and Password". The dialog has a blue title bar with a small icon on the left and standard window controls (minimize, maximize, close) on the right. The main area is light gray and contains two labels: "Name:" and "Password:". To the right of each label is a vertical line representing an empty text input field. At the bottom center of the dialog is a button labeled "OK".



A screenshot of a Windows-style dialog box titled "Username and Password", identical in layout to the first one. However, the text input fields for "Name:" and "Password:" are now filled with white space, indicating they have been populated. The "OK" button remains at the bottom center.

# Fill constraints

- Digunakan utk komponen yang tampilannya memenuhi cell.
- Nilai default = NONE
- Terdapat 4 macam nilai:
  - **GridBagConstraints.BOTH**  
stretch component to fill the cell in both direction
  - **GridBagConstraints.NONE**  
the component display in its smallest size
  - **GridBagConstraints.HORIZONTAL**  
stretch component to fill the cell in the horizontal direction
  - **GridBagConstraints.VERTICAL**  
stretch component to fill the cell in the vertical direction

# Fill constraints

---

- name label  
`constraints.fill = GridBagConstraints.NONE;`
- name text field  
`constraints.fill = GridBagConstraints.HORIZONTAL;`
- password label  
`constraints.fill = GridBagConstraints.NONE;`
- password text field  
`constraints.fill = GridBagConstraints.HORIZONTAL;`
- ok button  
`constraints.fill = GridBagConstraints.NONE;`

# Anchor constraints

---

- Digunakan utk komponen yang tampilannya tidak memenuhi cell (alignment).
- Nilai default = CENTER
- Terdapat 8 macam nilai
  - GridBagConstraints.NORTH
  - GridBagConstraints.NORTHEAST
  - GridBagConstraints.EAST
  - GridBagConstraints.SOUTHEAST
  - GridBagConstraints.SOUTH
  - GridBagConstraints.SOUTHWEST
  - GridBagConstraints.WEST
  - GridBagConstraints.NORTHWEST



# Anchor constraints

---

- name label  
`constraints.anchor = GridBagConstraints.EAST;`
- name text field  
-
- password label  
`constraints.anchor = GridBagConstraints.EAST;`
- password text field  
-
- ok button  
`constraints.anchor = GridBagConstraints.CENTER;`

# Cell Padding and Insets

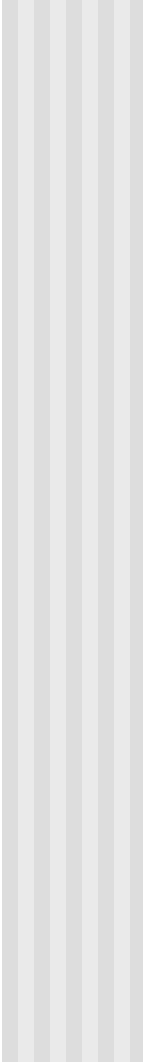
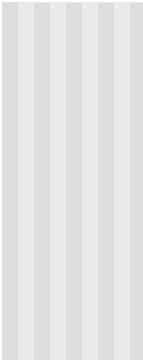
---

- Cell padding digunakan utk mengeset lebar atau tinggi suatu komponen.
- Digunakan constraints : *ipadx* dan *ipady*.
- Setting utk lebar dan tinggi komponent at least = nilai minimum komponen + (*ipadx* or *ipady* \* 2) pixels.
- Insets digunakan untuk mengeset lebar atau tinggi area diluar komponen.
- Digunakan konstrain insets.
- Konstruktor : *insets(int top, int left, int bottom, int right)*



---

■ FINISH





- 
- . BorderLayout either stacks its components on top of each other (with the first component at the top) or places them in a tight row from left to right -- your choice. You might think of it as a full-featured version of [FlowLayout](#). Here is an applet that demonstrates using BorderLayout to display a centered column of components:

- 
- One big difference between BoxLayout and the existing AWT layout managers is that BoxLayout respects each component's maximum size and X/Y alignment. We'll discuss that later.