Michael Jonathan Setiawan

58170468

Pertemuan 13 - Java GUI

*Graphical User Interface* (GUI) merupakan antar-muka grafis yang memfasilitasi interaksi antara pengguna dan program aplikasi secara lebih baik. Pengguna dapat berinteraksi dengan berbagai elemen grafis seperti *label*, *text field*, *button*, *dialog box*, *menu*, dan sebagainya.

Komponen yang digunakan untuk membangun GUI adalah *swing*. Komponen ini didefinisikan di dalam paket javax.swing. *Swing* diturunkan dari *Abstract Windowing Toolkit* (API) dari dalam paket java.awt

API untuk aplikasi GUI di Java:

1. *Abstract Window Toolkit* (AWT)

*Library* dan komponen GUI (java.awt) yang pertama kali diperkenalkan oleh Java dan Sun tidak merekomendasi lagi penggunaan komponennya.

1. *Java Foundation Class* (*Swing* atau JFC)

*Library* dan komponen GUI (javax.swing) yang terbaru dari Java dan yang direkomendasikan Sun untuk pemrograman Java. Komponennya sebagian besar turunan AWT dan lebih lengkap daripada AWT.

Java menyediakan dua *graphical library* yaitu AWT dan *Swing*.

Contoh: import java.awt.\*; import java.awt.event.\*; import javax.swing.

Fitur *swing*:

Komponen GUI lengkap: *button*, *listbox*, *combobox*, *textarea*, dsb.

*Pluggable Look-and-Feel*: tampilan GUI dapat diubah sesuai dengan kehendak (tidak perlu mengikuti *native* sistem operasi).

*Data Transfer* antar komponen: *drag and drop*, *copy and paste*.

*Internationalization*: proses desain aplikasi yang memungkinkan aplikasi dijalankan sesuai dengan preferensi tanpa rekompilasi.

*Localization*: proses translasi teks ke bahasa lokal dan menambahkan komponen lokal.

Komponen dasar *swing*:

1. *Top-Level container*: kontainer dasar dimana komponen lainnya diletakkan (JFrame, JDialog, dan Applet).
2. *Intermediate container*: kontainer perantara dimana komponen lainnya diletakkan (JPanel, JScrollPane, JTabbedPane, JToolbar, dan JSplitPane).
3. *Atomic component*: komponen yang memiliki fungsi spesifik dan menerima interaksi langsung dari user (JButton, JLabel, JTextField, JTextArea, dsb).
4. *Layout manager*: mengatur tata letak dan posisi komponen dalam kontainer (BorderLayout, BoxLayout, FlowLayout, GridBagLayout, dan GridLayout).
5. *Event handling*: menangani *event* yang dilakukan *user* (klik mouse, ketik keyboard, perbesar frame, dsb).

Aplikasi GUI Java

1. Konvensional (*full script*)

Merupakan cara yang digunakan *programmer* untuk membuat aplikasi dengan menulis *source code* untuk pembuatan objek *swing* yang akan digunakan seperti membuat *label*, *text field*, dll.

1. *Palette*

Pembuatan *form* dengan merancang masing-masing objek *pallets* yang dibutuhkan ke dalam *form* yang sudah disediakan oleh Netbeans.

*Top-Level container* (JFrame, JDialog, dan Applet)

1. JFrame

Kontruktor *class* adalah JFrame () 🡪 JFrame (String title).

Jika *frame* ditutup, gunakan *DO\_NOTHING\_ON\_CLOSE*, *HIDE\_ON\_CLOSE*, *DISPOSE\_ON\_CLOSE*, atau *EXIT\_ON\_CLOSE*.

1. JDialog

Berbeda dengan JFrame, JDialog tidak dibuat berdiri sendiri melainkan dibuat bersama-sama dengan *frame* sebagai *parent*-nya. Jika *parent*-nya ditutup, maka *dialog* akan dihapus dari memori. Kemunculan *dialog* akan membuat semua *input* terhadap *frame parent*-nya terblokir sampai *dialog* ditutup.

Tiga jenis tombol jika ingin ditampilkan pada kotak *dialog*:

1. JOptionPane.OK\_CANCEL\_OPTION
2. JOptionPane.YES\_NO\_OPTION
3. JOptionPane.YES\_NO\_CANCEL\_OPTION

Empat gambar *icon* yang digunakan untuk memanggil menggunakan perintah:

1. JOptionPane.QUESTION\_MESSAGE
2. JOptionPane.INFORMATION\_MESSAGE
3. JOptionPane.WARNING\_MESSAGE
4. JOptionPane.ERROR\_MESSAGE

*Intermediate container* (JPanel, JScrollPane, JTabbedPane, JToolbar, dan JSplitPane)

*Intermediate container* digunakan sebagai tempat untuk meletakkan komponen lain di dalamnya.

1. JPanel

Digunakan untuk mengelompokkan beberapa komponen lainnya dengan menggunakan *layout manager* tertentu.

Kontruktor *class* adalah:

JPanel ()

JPanel (boolean isDoubleBuffered)

JPanel (LayoutManager layout)

JPanel (LayoutManager layout, boolean isDoubleBuffered)

1. JScrollPane

Digunakan untuk menampilkan komponen yang ukurannya lebih besar dari daeah tampilannya.

Kontruktor *class* adalah:

JScrollPane ()

JScrollPane (Component view)

JScrollPane (Component view, int vsbPolicy, int hsbPolicy)

JScrollPane (int vsbPolicy, int hsbPolicy)

*Atomic component* (JButton, JLabel, JTextField, JTextArea, dsb).

1. JLabel

Digunakan untuk menampilkan teks atau gambar statis.

Kontruktor *class* adalah:

JLabel ()

JLabel (Icon image)

JLabel (Icon image, int horizontalAligment), atau

JLabel ()

JLabel (String text)

JLabel (String text, Icon icon, int horizontalAligment)

JLabel (String text, int horizontalAligment)

1. JTextField

Digunakan untuk menampilkan teks yang dapat di-*edit­* oleh ­*user*.

Kontruktor *class* adalah:

JTextField ()

JTextField (Document doc, String text, int coloumns)

JTextField (int columns)

JTextField (String text)

JTextField (String text, int columns)

*Method* yang digunakan untuk mengambil teks di dalam *text field*:

void setText (String)

String getText()

*Layout manager* (BorderLayout, BoxLayout, FlowLayout, GridBagLayout, dan GridLayout)

*Layout manager* digunakan untuk mengatur tampilan dari setiap komponen yang diletakkan pada *container*.

Untuk mengatur *layout manager*, gunakan *method* berikut:

void setLayout (LayoutManager mgr)

\*mgr: objek *layout manager* yang ingin digunakan.

Untuk mendapatkan *content pane* dari *top*-*level container*, gunakan *method* berikut:

container getContentPane()

1. BorderLayout

Merupakan *default manager* untuk ContentPane. *Layout* ini menempatkan komponen dengan pendekatan arah mata angin (*north*, *south*, *west*, *centre*, dan *east*).

*Layout* meletakkan komponen berurutan ke kanan-kiri (sumbu x) dan ke atas-bawah (sumbu y).

1. FlowLayout

*Layout* ini digunakan oleh JPanel, dimana hanya meletakkan komponen yang ada berurutan dari kiri ke kanan dan jika diperlukan akan berpindah baris.

1. GridLayout

*Layout* ini pada dasarnya akan meletakkan komponen yang ada ke dalam baris dan kolom yang telah ditentukan.

*Event handling* (klik mouse, ketik keyboard, perbesar frame, dsb)

*Event* merupakan suatu peristiwa atau kejadian yang dilakukan oleh *user* terhadap *user-interface*.

Contoh *event* yang dihasilkan oleh komponen *swing*:

1. ActionListener: mengklik tombol pada form, menekan enter pada waktu mengetik di *text box* atau memilih menu.
2. WindowListener: menutup sebuah *frame*.
3. MouseListener: mengklik tombol *mouse* selagi kursor *mouse* berada di atas sebuah komponen.
4. MouseMotionListener: menggerakkan kursor *mouse* di atas sebuah komponen.
5. ComponentListener: menampilkan suatu komponen.
6. FocusListener: suatu komponen mendapatkan fokus untuk *input*.