Bahasa Pemrograman

(Pemrograman Visual dengan Java)

Modul Praktikum

# Topik

Penggunaan java Swing Menu (menu bar dan popup menu), JDialog (swing window pada aplikasi dengan menggunakan Netbeans IDE.

# Tujuan

Setelah menyelesaikan praktikum ini, mahasiswa diharapkan mampu:

1. Membuat aplikasi GUI dengan menggunakan JmenuBar, JMenu dengan Netbeans IDE.
2. Membuat aplikasi GUI dengan menggunakan JPopupMenu dengan Netbeans IDE.
3. Membuat aplikasi GUI dengan menggunakan JDialog dengan Netbeans IDE
4. Membuat aplikasi GUI dengan menggunakan Menu Bar untuk memanggil JDialog dengan Netbeans IDE.
5. Membuat aplikasi GUI dengan menggunakan Popup Menu untuk memanggil JDialog dengan Netbeans IDE.

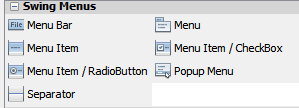
# Alat dan Bahan

* Komputer dengan **Java Development Kit (JDK)** dan Netbeans IDE sudah terinstall
* Informasi download dan instalasi JDK dan Netbeans dapat diperoleh di; https://netbeans.org/downloads/index.html

# Rangkuman/Teori

## [Menjelaskan cara membuat menu pada aplikasi dengan Netbeans IDE]

Untuk membuat menu pada aplikasi berbasis java biasanya digunakan widget-widget pada swing menu sebagai berikut:



Gambar 1. Widget swing menus

Pembuatan menu pada aplikasi bisa dibuat dengan melakukan kombinasi dari widget-widget swing menu seperti dapat dilihat pada gambar 1.

Terdapat 2 jenis menu yang dapat dibuat melalui Swing Menu, yaitu:

* Jenis Menu Bar

Menu Bar merupakan object menu yang melekat pada window aplikasi biasanya berada di bagian atas pada window aplikasi, pada Menu Bar biasanya ditambahkan Menu atau Menu Item sebagai isi atau sub-menu dari Menu bar.

* Jenis Popup Menu

Popup Menu merupakan object menu yang secara dinamic popup (tampil) pada posisi tertentu di dalam window aplikasi, pada Popup Menu biasanya ditambahkan Menu atau Menu Item sebagai isi atau sub-menu dari Popup Menu.

### Membuat menu dengan Menu Bar

Untuk membuat menu dengan Menu Bar:

1. Masukkan Menu Bar ke Frame, (saran: letakkan di sisi atas Frame)



Gambar 2. Menu bar pada frame

1. Ubah variabel name (pada tab code properties) sesuai kegunaan (misal: variable name: mnuBar)
2. Ubah Text tiap-tiap JMenu (*right-click* pada tulisan di menu bar 🡪 pilih Edit Text) sesuai kegunaan. Misal perubahannya seperti berikut:



Gambar 3. Menu bar setelah di *edit text*

1. Ubah variable name untuk tiap-tiap JMenu sesuai teks yang tampil pada Menu Bar (untuk kemudahan mengingat saat membuat kode program), misal: **mnuAplikasi** untuk menu Aplikasi, dan **mnuBantuan** untuk menu Bantuan.
2. Tambahkan Menu Item ke JMenu untuk tiap-tiap JMenu sesuai kebutuhan aplikasi, misalkan seperti berikut:

|  |  |
| --- | --- |
|  |  |

Gambar 4. Menu bar beserta menu item

1. Ubah Text tiap-tiap JMenuItem (*right-click* pada tulisan di menu item 🡪 pilih Edit Text) sesuai kegunaan. Misal perubahannya seperti berikut:

|  |  |
| --- | --- |
|  |  |

Gambar 5. Menu item setelah di *edit text*

1. Ubah variable name untuk tiap-tiap JMenuItem sesuai teks yang tampil pada Menu Item (untuk kemudahan mengingat saat membuat kode program), misal: mnuItemAplikasi1 untuk menu item Aplikasi 1, mnuItemAplikasi2 untuk menu item Aplikasi 2 dan seterusnya.
2. Tambahkan shortcut jika diperlukan.

Untuk shortcut terdapat dua jenis:

* 1. **Accelerator**: Shortcut yang bisa langsung tanpa membuka menu (biasanya dengan kombinasi fungsi Ctrl).

Untuk membuatnya: click **menu item** yang ingin dibuatkan shortcutnya, kemudian pada **tab properties** 🡪 Accelerator: Ctrl+*CharYangDiinginkan*

Misal: Accelerator: Ctrl+x , untuk menu item Exit

* 1. **Mnemonic**: Shortcut yang berfungsi setelah menu dibuka (biasanya dengan langsung menekan tombol sesuai karakter yang di-*underline*)

Untuk membuatnya: click menu item yang ingin dibuatkan shortcutnya, kemudian pada tab properties 🡪 Mnemonic: *CharYangDiinginkan*

Misal: Mnemonic: x , untuk menu item Exit

Misal seperti berikut:

|  |  |
| --- | --- |
|  |  |

Gambar 6. Design dan tampilan aplikasi Menu item dengan shortcut

1. Untuk memberikan instruksi pada menu item:

*Click* menu Item 🡪 *Right-click* menu item 🡪 Events 🡪 Action 🡪 actionPerformed

kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal kodenya seperti berikut:

private void **mnuItemExitActionPerformed**(java.awt.event.ActionEvent evt) {

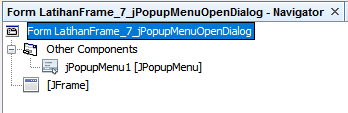
*System.exit(0)*; //instruksi untuk exit window aplikasi

}

### Membuat menu dengan Popup Menu

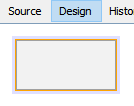
Untuk membuat menu dengan Popup Menu:

1. Masukkan Popup Menu ke Frame (jPopupMenu tidak akan terlihat)
2. Untuk mengakses dan melakukan setting terhadap JpopupMenu gunakan ***navigator*** class (karena JPopupMenu tidak terlihat pada Jframe) dan bahkan di navigator terlihat bahwa JPopupMenu masuk ke ***other*** ***components*** (tidak masuk ke JFrame).



Gambar 7. JPopupMenu di dalam *other* components pada *navigator*

1. Untuk melihat PopupMenu, *double-click* pada jPopupMenu1 sehingga tulisan jPopupMenu1 menjadi bold.
2. Ubah variabel name (pada tab code properties) sesuai kegunaan (misal: **variable name: mnuPopup**)
3. Ubah ukuran Popup Menu (***preferred size*** dan ***Minimum Size***) ***sehingga*** layak dilihat (karena default-nya [0,0] sangat kecil), misal ***preferred size:* [100, 50]** dan ***Minimum Size:*** **[100, 50]**



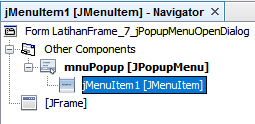
Gambar 8. Popup Menu terlihat dengan ukuran [100, 50]

1. Masukkan JMenuItem ke Popup Menu (JMenuItem tidak akan terlihat), untuk itu gunakan **Navigator class**.



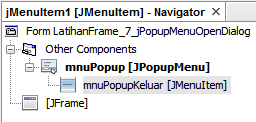
Gambar 9. jMenuItem pada Other Components

1. Pada Navigator, *click & drag* jMenuItem1 ke mnuPopup [JpopupMenu] sehingga jMenuItem sekarang berada di dalam mnuPopup [JpopupMenu] .



Gambar 10. jMenuItem berada dalam mnuPopup [JpopupMenu]

1. *Click* pada jMenuItem1 kemudian pada tab *code* ubah variable name sesuai fungsi dari menu item, misal; ***Variable Name:* mnuPopupKeluar**.



Gambar 11. JMenuItem dengan nama **mnuPopupKeluar** di dalam mnuPopup [JpopupMenu]

1. Pada mnuPopupKeluar Ubah Teksnya (pada tab properties🡪Text) sesuaikan dengan teks yang akan tampil pada popup menu, misal: ***Text*: Exit**
2. Ulangi langkah 6 – 9 untuk menambahkan menu item lainnya.
3. Untuk memberikan instruksi pada menu item (misal pada **mnuPopupKeluar**):

*Click* mnuPopupKeluar (pada navigator class) 🡪 *Right-click* mnuPopupKeluar 🡪 Events 🡪 Action 🡪 actionPerformed

Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

private void mnuPopupKeluarActionPerformed(java.awt.event.ActionEvent evt) {

System.exit(0); //instruksi untuk exit window aplikasi

}

1. Untuk memberikan kode agar Popup Menu bisa tampil saat pengguna *right-click* pada frame

*Right-click* pada *Jframe* 🡪 Events 🡪 Mouse 🡪 MouseClicked

Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

private void formMouseClicked(java.awt.event.MouseEvent evt) {

if(evt.getButton()==MouseEvent.BUTTON3) {

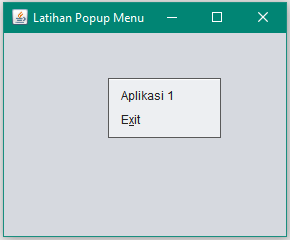
memeriksa apakah right-click

mnuPopup.show(this, evt.getX(), evt.getY());

menampilkan popup menu dengan koordinat sesuai dengan posisi pointer

}

}



Gambar 12. Popup Menu tampil saat di *right-click* di dalam Frame

## [Menjelaskan cara membuat dialog pada aplikasi dengan Netbeans IDE]

Untuk membuat dialog pada aplikasi berbasis java biasanya digunakan widget jDialog pada swing windows.

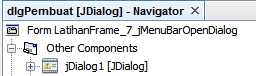


Gambar 13. jDialog pada swing windows.

### Membuat dialog dengan jDialog

Untuk membuat dialog dengan Menggunakan jDialog:

1. Masukkan jDialog ke Frame (jDialog tidak akan terlihat).
2. Untuk mengakses dan melakukan setting terhadap jDialog gunakan ***navigator*** class, karena jDialog tidak terlihat dan bahkan di navigator terlihat bahwa jDialog masuk ke ***other*** ***components*** (tidak masuk ke jFrame).



Gambar 14. jDialog di dalam *other* components pada *navigator*

1. Untuk melihat dialog, *double-click* pada jDialog1 sehingga tulisan jDialog1 menjadi bold.
2. Ubah variabel name (pada tab code properties) sesuai kegunaan (misal: **variable name: dlgPembuat** , untuk membuat dialog tentang pembuat)
3. Ubah title (pada tab properties) sesuai judul dari dialog yang kita buat (misal: **Title: Tentang Pembuat**)
4. Ubah ukuran dialog (***preferred size*** dan ***Minimum Size***) ***sehingga*** layak dilihat (karena default-nya [0,0] sangat kecil), misal ***preferred size:* [290, 220]** dan ***Minimum Size:*** **[290, 220]**
5. Tambahkan widget button pada dialog, kemudian ubah **variable name: btnOk** dan **Text: OK**.

Untuk membuat instruksi pada btnOk:

*right-click* btnOk 🡪 events 🡪 Action 🡪 actionPerformed

Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

private void btnOkActionPerformed(java.awt.event.ActionEvent evt) {

dlgPembuat.dispose(); //instruksi untuk menutup dlgPembuat

}

1. Untuk mengaktifkan dlgPembuat dari menu (Bantuan🡪Tentang Pembuat);



Gambar 15. Menu item Tentang Pembuat

*Click* kemudian *right-click* menu item Tentang Pembuat 🡪 events 🡪 Action 🡪 actionPerformed

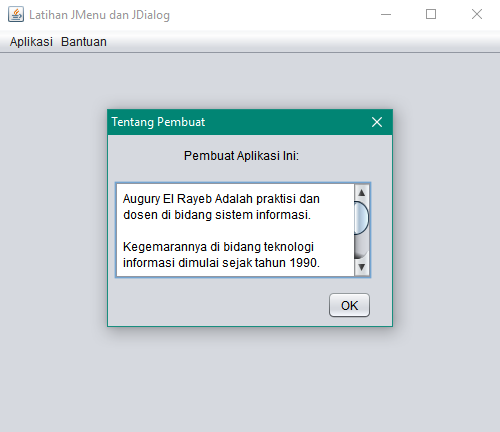
Kemudian tulis instruksi yang diinginkan pada area kode tersebut, misal seperti berikut:

private void mnuItemPembuatActionPerformed(java.awt.event.ActionEvent evt) {

dlgPembuat.setLocationRelativeTo(this); //instruksi agar posisi dialog didalam window

dlgPembuat.setVisible(true); //instruksi menampilkan dlgPembuat

}



Gambar 16. Tampilan dlgPembuat di dalam window aplikasi

# Analisa Hasil Praktek

Berdasarkan praktek yang telah Anda lakukan, buat kesimpulan:

# Tugas

1. Buat aplikasi perhitungan bangun yang akan memiliki dialog untuk fungsi sebagai berikut:
   1. Program yang akan melakukan perhitungan luas segitiga, dan menampilkan hasilnya.
   2. Program yang akan melakukan perhitungan luas lingkaran, dan menampilkan hasilnya.
   3. Program yang akan melakukan perhitungan luas persegi, dan menampilkan hasilnya.
   4. Program menampilkan tentang pembuat aplikasi.

# Appendix 1: Form Data Praktek

Nama : ……………………….

NIM : ……………………….

Tgl : ……………………….

|  |  |
| --- | --- |
| **Penggalan *Source code*** | **Keluaran / *Output*** |
|  |  |
|  |  |
|  |  |