LAPORAN PRATIKUM CODELAB PBO 2E

MODUL 6

Nama: Muhammad Syakhish Al Hanif
NIM: 202410370110189
Kelas: PBO 2E
Github:
CreamyButDreamy/PBO_2E_Syakhish_189

CODELAB:

1. **Struktur Program dan Penjelasan**

```
10
11  public class TebakAngka extends Application {  new *
12
13      private int angkaRahasia;  3 usages
14      private int jumlahTebakan;  3 usages
15      private final Random random = new Random();  1 usage
16
```

Baris “public class TebakAngka extends Application” menyatakan bahwa “TebakAngka” adalah kelas utama dalam aplikasi ini. Dengan menggunakan “extends Application”, berarti kelas “TebakAngka” mewarisi semua fitur dasar dari kelas “Application” milik JavaFX.

Atribut-atribut berikut adalah inti dari logika permainan:

- “private int angkaRahasia;” : Variabel integer ini akan menyimpan angka acak yang harus ditebak oleh pemain.
- “private int jumlahTebakan;” : Variabel integer untuk menghitung dan menyimpan berapa kali pemain telah melakukan tebakan.
- “private final Random random = new Random();”: Objek dari kelas `java.util.Random`. Kata kunci `final` berarti referensi `random` tidak dapat diubah setelah diinisialisasi (objek `Random` itu sendiri tetap bisa digunakan untuk menghasilkan angka acak). Objek ini digunakan untuk menghasilkan `angkaRahasia`.

2. **Atribut untuk Komponen Antarmuka Pengguna (GUI)**

```
16
17      // Komponen GUI
18      private Label labelFeedback;  7 usages
19      private Label labelJumlahTebakan;  4 usages
20      private TextField inputTebakan;  6 usages
21      private Button tombolTebak;  6 usages
22
```

Variabel-variabel ini dideklarasikan sebagai atribut kelas (instance variables) karena mereka merepresentasikan komponen GUI yang perlu diakses dan dimanipulasi dari berbagai metode di dalam kelas `TebakAngka`.

- “private Label labelFeedback;”: Objek `Label` untuk menampilkan pesan umpan balik kepada pengguna (misalnya, "Kekecilan!", "Benar!").
- “private Label labelJumlahTebakan;”: Objek `Label` lain untuk menampilkan jumlah percobaan yang telah dilakukan.
- “private TextField inputTebakan;”: Objek `TextField` yang memungkinkan pengguna mengetikkan angka tebakan mereka.
- “private Button tombolTebak;”: Objek `Button` yang akan diklik pengguna untuk mengirimkan tebakan mereka.

3. **Metode `start(Stage stage)`: Titik Masuk dan Inisialisasi UI**

```
@Override  new *
public void start(Stage stage) {
    generateAngkaRahasia();

    // Label judul
    Label labelJudul = new Label( s: "Tebak angka 0 - 100");
    labelJudul.setStyle("-fx-font-size: 16px; -fx-font-weight: bold;");

    // Input dan tombol dalam satu baris (HBox)
    inputTebakan = new TextField();
    inputTebakan.setPromptText("Masukkan angka");
    inputTebakan.setMaxWidth(120);

    tombolTebak = new Button( s: "Coba Tebak!");
```

Langkah-langkah awal di dalam `start()`:

- “generateAngkaRahasia();”: Memanggil metode internal untuk menghasilkan angka rahasia pertama dan mereset jumlah tebakan. Ini memastikan permainan siap dimainkan saat aplikasi dimulai.
- “Label labelJudul = new Label("Tebak angka 0 - 100");”: Membuat objek `Label` baru untuk menampilkan judul permainan. Teks "Tebak angka 0 - 100" akan ditampilkan.
- “labelJudul.setStyle("-fx-font-size: 16px; -fx-font-weight: bold;");”: Mengatur gaya CSS inline untuk `labelJudul`. Ini mengubah ukuran font menjadi 16 piksel dan membuatnya tebal.
- “inputTebakan = new TextField();”: Membuat objek `TextField` baru. Ini akan menjadi tempat pengguna memasukkan tebakan mereka.
- “inputTebakan.setPromptText("Masukkan angka");”: Mengatur teks placeholder (prompt text) yang muncul di `inputTebakan` saat kosong, memberikan petunjuk kepada pengguna.
- “inputTebakan.setMaxWidth(120);”: Membatasi lebar maksimum dari `inputTebakan` menjadi 120 piksel, agar tidak terlalu lebar di layar.
- “tombolTebak = new Button("Coba Tebak!");”: Membuat objek `Button` baru dengan teks "Coba Tebak!" yang akan ditampilkan pada tombol.

4. **Pengaturan Event Handler pada Tombol (`setOnAction`)**

```
tombolTebak.setOnAction( ActionEvent e -> prosesTebakan());
```

`setOnAction` pada `tombolTebak` menetapkan bahwa metode `prosesTebakan()` akan dieksekusi setiap kali tombol diklik. Ini menggunakan lambda expression `e -> prosesTebakan()` untuk menangani event klik.

5. **Pengaturan Layout Komponen (`HBox` dan `VBox`)**

```
38
39          HBox inputArea = new HBox( v: 10, inputTebakan, tombolTebak);
40          inputArea.setAlignment(Pos.CENTER);
41
42          // Label feedback dan jumlah tebakan
43          labelFeedback = new Label( s: "Silakan masukkan tebakan.");
44          labelJumlahTebakan = new Label( s: "Jumlah percobaan: 0");
45
46          // Layout utama
47          VBox layout = new VBox( v: 15, labelJudul, inputArea, labelFeedback, labelJumlahTebakan);
48          layout.setPadding(new Insets( v: 20));
49          layout.setAlignment(Pos.CENTER);
```

- "HBox inputArea = new HBox(10, inputTebakan, tombolTebak);": Membuat objek `HBox` (Horizontal Box). Angka `10` adalah spasi (dalam piksel) antar komponen di dalam `HBox` ini (`inputTebakan` dan `tombolTebak`).
- "inputArea.setAlignment(Pos.CENTER);": Mengatur agar konten di dalam `inputArea` (yaitu, TextField dan Button) diposisikan di tengah secara horizontal dalam `HBox` tersebut.
- "labelFeedback = new Label("Silakan masukkan tebakan.");" dan "labelJumlahTebakan = new Label("Jumlah percobaan: 0");": Membuat dua objek `Label` lagi yang akan digunakan untuk menampilkan umpan balik dan jumlah percobaan. Ini diinisialisasi dengan teks awal.
- "VBox layout = new VBox(15, labelJudul, inputArea, labelFeedback, labelJumlahTebakan);": Membuat objek `VBox` (Vertical Box). Angka `15` adalah spasi antar komponen di dalam `VBox` ini. Komponen yang ditambahkan adalah `labelJudul`, `inputArea` (HBox yang kita buat sebelumnya), `labelFeedback`, dan `labelJumlahTebakan`.
- "layout.setPadding(new Insets(20));": Mengatur padding (jarak antara tepi `VBox` dengan konten di dalamnya) sebesar 20 piksel di semua sisi (atas, kanan, bawah, kiri).
- "layout.setAlignment(Pos.CENTER);": Mengatur agar seluruh konten di dalam `VBox` (`layout`) diposisikan di tengah secara vertikal dan horizontal dalam area yang tersedia untuk `VBox` tersebut.

6. **Pembuatan `Scene` dan Menampilkan `Stage`**

```
        Scene scene = new Scene(layout,  v: 350,  v1: 200);
        stage.setScene(scene);
        stage.setTitle("Tebak Angka");
        stage.show();
    }
```

- "Scene scene = new Scene(layout, 350, 200);": Membuat objek `Scene`. Parameter pertama (`layout`) adalah *root node* dari scene graph – dalam kasus ini, `VBox` yang berisi semua komponen kita. Parameter kedua (`350`) dan ketiga (`200`) adalah lebar dan tinggi awal dari `Scene` dalam piksel.
- "stage.setScene(scene);": Mengatur `Scene` yang baru dibuat ini ke `Stage` (jendela utama aplikasi). Setiap `Stage` harus memiliki sebuah `Scene` untuk menampilkan konten.
- "stage.setTitle("Tebak Angka");": Mengatur teks yang akan muncul di title bar jendela aplikasi.
- "stage.show();": Ini membuat `Stage` (dan semua isinya) terlihat oleh pengguna.

7. **Metode `prosesTebakan()`: Logika Awal untuk "Main Lagi"**

```
private void prosesTebakan() {  1 usage  new *
    if (tombolTebak.getText().equals("Main Lagi")) {
        generateAngkaRahasia();
        inputTebakan.clear();
        tombolTebak.setText("Coba Tebak!");
        labelFeedback.setText("Silakan masukkan tebakan.");
        labelJumlahTebakan.setText("Jumlah percobaan: 0");
        return;
    }
```

Jika teks tombol adalah "Main Lagi" (setelah tebakan benar), metode ini mereset permainan: memanggil `generateAngkaRahasia()`, mengosongkan input, dan mereset teks tombol serta label. Pernyataan `return` mengakhiri eksekusi metode.

8. **Metode `prosesTebakan()`: Pengambilan dan Validasi Input**

```
try {
    int tebakan = Integer.parseInt(inputTebakan.getText());
    jumlahTebakan++;
} catch (NumberFormatException e) {
    labelFeedback.setText("Masukkan angka yang valid!");
}
}
```

Program mencoba mengubah input pengguna menjadi integer menggunakan `Integer.parseInt()`. Jika berhasil, `jumlahTebakan` ditambah. Jika gagal (misalnya input bukan angka), `NumberFormatException` ditangkap dan pesan error ditampilkan di `labelFeedback`

9. **Metode `prosesTebakan()`: Logika Perbandingan dan Umpan Balik**

```
    if (tebakan < angkaRahasia) {
        labelFeedback.setText("Kekecilan!");
    } else if (tebakan > angkaRahasia) {
        labelFeedback.setText("Kebesaran!");
    } else {
        labelFeedback.setText("Tebakan anda benar!");
        tombolTebak.setText("Mainkan Lagi");
    }

    labelJumlahTebakan.setText("Jumlah percobaan: " + jumlahTebakan);
```

Tebakan pengguna dibandingkan dengan `angkaRahasia`. `labelFeedback` diperbarui sesuai hasilnya ("Kekecilan!", "Kebesaran!", atau "Tebakan anda benar!"). Jika benar, teks tombol diubah menjadi "Main Lagi". `labelJumlahTebakan` selalu diperbarui.

10. Metode `generateAngkaRahasia()`

```
    private void generateAngkaRahasia() {  2 usages  new *
        angkaRahasia = random.nextInt( bound: 100) + 1;
        jumlahTebakan = 0;
    }

    public static void main(String[] args) {  new *
        launch(args);
    }
}
```

- Metode ini menghasilkan angka acak baru antara 1 dan 100 (inklusif) menggunakan `random.nextInt(100) + 1` dan menyimpannya di `angkaRahasia`. `jumlahTebakan` juga direset menjadi 0 untuk permainan baru.
- berfungsi sebagai pintu masuk utama untuk menjalankan aplikasi Java. Perintah launch(args); di dalamnya secara khusus memulai aplikasi JavaFX Anda, menyiapkan semua yang dibutuhkan untuk menampilkan antarmuka pengguna grafis.