Universitas Pamulang
Sistem Informasi

MOBILE PROGRAMMING

PERTEMUAN 12

Yudisti Prayigo Permana, S.Kom., M.Kom.

Universitas Pamulang
Sistem Informasi

TOPIK PEMBAHASAN

- Pengenalan OptionMenu 01
- Pengenalan ContextMenu 02
- Pengenalan Latihan 03
- Form Login 04
- Register Form 05
- Tugas 06

Universitas Pamulang
Sistem Informasi

Apa itu OptionMenu?

OptionMenu merupakan menu utama yang digunakan dalam sebuah Activity. Ini menyediakan pilihan - pilihan seperti “Settings”, “Help”, “About”, “Profile”, “Logout” dan lain - lain. Menu ini ditampilkan saat pengguna mengetuk tombol menu (di perangkat lama) atau ikon overflow (tiga titik vertikal) di ActionBar/Toolbar.

Contoh

```xml
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/action_settings"
        android:title="Settings"
        android:orderInCategory="100"
        android:showAsAction="never" />
    <item
        android:id="@+id/action_about"
        android:title="About"
        android:orderInCategory="101"
        android:showAsAction="never" />
</menu>
```

```kotlin
class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
    }

    override fun onCreateOptionsMenu(menu: Menu?): Boolean {
        menuInflater.inflate(R.menu.menu_main, menu)
        return true
    }

    override fun onOptionsItemSelected(item: MenuItem): Boolean {
        return when (item.itemId) {
            R.id.action_settings -> {
                Toast.makeText(this, "Settings diklik", Toast.LENGTH_SHORT).show()
                true
            }
            R.id.action_about -> {
                Toast.makeText(this, "Tentang aplikasi", Toast.LENGTH_SHORT).show()
                true
            }
            else -> super.onOptionsItemSelected(item)
        }
    }
}
```

Universitas Pamulang
Sistem Informasi

Apa itu ContextMenu?

ContextMenu merupakan menu konsekstual yang muncul saat pengguna menekan lama (long press) pada suatu elemen (biasanya View, seperti ListView, TextView, dll). Menu ini memberikan opsi tindakan untuk yang terkait langsung dengan elemen tersebut.

Struktur ContextMenu

Untuk menggunakan ContextMenu, perlu :

- Mendaftarkan view agar bisa memiliki ContextMenu
- Membuat dan menampilkan menu-nya
- Menangani aksi saat item menu diklik

Contoh

```xml
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp">

    <TextView
        android:id="@+id/myTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Tekan lama saya untuk menu konteks"
        android:textSize="18sp" />
</LinearLayout>

```

```kotlin
class MainActivity : AppCompatActivity() {

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val textView = findViewById<TextView>(R.id.myTextView)
        registerForContextMenu(textView)
    }

    override fun onCreateContextMenu(
        menu: ContextMenu?,
        v: View?,
        menuInfo: ContextMenu.ContextMenuInfo?
    ) {
        super.onCreateContextMenu(menu, v, menuInfo)
        menuInflater.inflate(R.menu.context_menu, menu)
    }

    override fun onContextItemSelected(item: MenuItem): Boolean {
        return when (item.itemId) {
            R.id.edit -> {
                Toast.makeText(this, "Edit diklik", Toast.LENGTH_SHORT).show()
                true
            }
            R.id.delete -> {
                Toast.makeText(this, "Delete diklik", Toast.LENGTH_SHORT).show()
                true
            }
            else -> super.onContextItemSelected(item)
        }
    }
}

```

ContextMenu

Dalam Flutter, ContextMenu tidak ada secara langsung seperti di Android Native. Namun, bisa mengimplementasikan fungsionalitas serupa (menu yang muncul saat pengguna melakukan long press atau tap tertentu pada widget) menggunakan beberapa cara :

- showMenu() - Untuk menampilkan menu manual di lokasi tertentu
- PopupMenuButton - Untuk menu yang terikat ke ikon
- GestureDetector atau InkWell + showMenu() - Untuk menangani long press dan memunculkan menu
- ContextMenu dari Cupertino (khusus iOS style)

Android VS Flutter

Fitur	Andriod		Flutter	
	OptionMenu	ContextMenu	OptionMenu	ContextMenu
Tujuan	Menampilkan opsi umum (global actions)	Menampilkan opsi konsekstual	Menampilkan opsi umum di AppBar/ikon	Menampilkan opsi berdasarkan elemen tertentu
Pemicu	Tekan ikon tiga titik di AppBar	Tekan lama (long press)	Tekan ikon di AppBar atau Button	Long press / tap elemen tertetu
Implementasi Visual	AppBar -> Overflow Menu	Menu pop-up saat long press View	AppBar -> PopupMenuButton	showMenu() manual pada posisi long press
XML/Dart	Menggunakan XML + Java/Kotlin	XML menu + Java/Kotlin + register	Gunakan widget PopupMenuButton di Dart	Gunakan GestureDetector dan showMenu() di Dart
Cocok Untuk	Navigasi umum, pengaturan, favorit	Aksi spesifik item (hapus, ubah, dsb)	Pengaturan, filter, sort, logout	Aksi per item seperti edit, hapus, share
Support iOS look	Tidak	Tidak	Tidak langsung, bisa custom	Ya, CupertinoContextMenu tersedia untuk iOS

Android vs Flutter

```kotlin
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    val textView = findViewById<TextView>(R.id.myTextView)
    registerForContextMenu(textView)
}

override fun onCreateContextMenu(menu: ContextMenu?, v: View?, menuInfo: ContextMenu.ContextMenuInfo?) {
    super.onCreateContextMenu(menu, v, menuInfo)
    menuInflater.inflate(R.menu.context_menu, menu)
}

override fun onContextItemSelected(item: MenuItem): Boolean {
    return when (item.itemId) {
        R.id.edit -> { /* handle edit */ true }
        R.id.delete -> { /* handle delete */ true }
        else -> super.onContextItemSelected(item)
    }
}

```

```dart
AppBar(
  title: Text("OptionMenu Example"),
  actions: [
    PopupMenuButton<String>(
      onSelected: (value) {
        if (value == 'settings') {
          // Handle Settings
        }
      },
      itemBuilder: (context) => [
        PopupMenuItem(value: 'settings', child: Text('Settings')),
        PopupMenuItem(value: 'logout', child: Text('Logout')),
      ],
    )
  ],
),

```

SOURCE CODE

main.dart

```dart
import 'package:flutter/material.dart';

void main() => runApp(const TodoApp());

class TodoApp extends StatelessWidget {
  const TodoApp({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      title: 'Mobile Programming',
      theme: ThemeData(
        colorScheme: ColorScheme.fromSeed(seedColor: Colors.blue),
        useMaterial3: true,
      ),
      home: const TodoListScreen(),
    );
  }
}
```

```dart
enum TodoMenuAction {
  add,
  clearAll,
  showCompleted,
  settings,
  about,
}

enum TodoItemAction {
  complete,
  edit,
  delete,
  priority,
}

class TodoListScreen extends StatefulWidget {
  const TodoListScreen({super.key});

  @override
  State<TodoListScreen> createState() => _TodoListScreenState();
}
```

SOURCE CODE

```dart
class _TodoListScreenState extends State<TodoListScreen> {
  final List<TodoItem> _todos = [];
  bool _showCompleted = true;
  final TextEditingController _addController =
TextEditingController();
  final List<PopupMenuEntry<TodoMenuAction>> _optionMenuItems = [
    const PopupMenuItem<TodoMenuAction>(
      value: TodoMenuAction.add,
      child: ListTile(
        leading: Icon(Icons.add),
        title: Text('Tambah Task'),
      ),
    ),
    const PopupMenuItem<TodoMenuAction>(
      value: TodoMenuAction.clearAll,
      child: ListTile(
        leading: Icon(Icons.delete_forever),
        title: Text('Hapus Semua'),
      ),
    ),
    const PopupMenuItem<TodoMenuAction>(
      value: TodoMenuAction.showCompleted,
      child: ListTile(
        leading: Icon(Icons.check_circle),
        title: Text('Toggle Selesai'),
      ),
    ),
    const PopupMenuItem<TodoMenuAction>(
      value: TodoMenuAction.about,
      child: ListTile(
        leading: Icon(Icons.info),
        title: Text('Tentang Aplikasi'),
      ),
    ),
  ];
```

```dart
final List<PopupMenuEntry<TodoItemAction>> _contextMenuItems = [
    const PopupMenuItem<TodoItemAction>(
      value: TodoItemAction.complete,
      child: ListTile(
        leading: Icon(Icons.check),
        title: Text('Tandai Selesai'),
      ),
    ),
    const PopupMenuItem<TodoItemAction>(
      value: TodoItemAction.edit,
      child: ListTile(
        leading: Icon(Icons.edit),
        title: Text('Edit'),
      ),
    ),
    const PopupMenuItem<TodoItemAction>(
      value: TodoItemAction.priority,
      child: ListTile(
        leading: Icon(Icons.flag),
        title: Text('Prioritas'),
      ),
    ),
    const PopupMenuItem<TodoItemAction>(
      value: TodoItemAction.delete,
      child: ListTile(
        leading: Icon(Icons.delete),
        title: Text('Hapus'),
      ),
    ),
  ];
```

```dart
@override
Widget build(BuildContext context) {
  final displayedTodos = _showCompleted
      ? _todos
      : _todos.where((todo) => !todo.isCompleted).toList();

  return Scaffold(
    appBar: AppBar(
      title: const Text('Pertemuan 12'),
      backgroundColor: Colors.blue,
      actions: [
        IconButton(
          icon: const Icon(Icons.add),
          onPressed: _showAddDialog,
        ),
        PopupMenuButton<TodoMenuAction>(
          onSelected: _handleOptionMenuAction,
          itemBuilder: (context) => _optionMenuItems,
        ),
      ],
    ),
    body: displayedTodos.isEmpty
        ? Center(
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              children: [
                const Icon(Icons.assignment, size: 64, color: Colors.grey),
                const SizedBox(height: 16),
                Text(
                  'Tidak ada task',
                  style: Theme.of(context).textTheme.headlineSmall,
                ),
                const SizedBox(height: 8),
                const Text('Tekan tombol + untuk menambahkan task baru'),
              ],
            ),
          )
```

SOURCE CODE

```
void _showContextMenu(BuildContext context, int index) {
  final RenderBox overlay =
      Overlay.of(context).context.findRenderObject() as RenderBox;

  showMenu<TodoItemAction>(
    context: context,
    position: RelativeRect.fromRect(
      Rect.fromPoints(
        const Offset(0, 0),
        const Offset(0, 0),
      ),
      Offset.zero & overlay.size,
    ),
    items: _contextMenuItems,
  ).then((action) {
    if (action != null) {
      _handleContextMenuAction(action, index);
    }
  });
}

void _handleOptionMenuAction(TodoMenuAction action) {
  switch (action) {
    case TodoMenuAction.add:
      _showAddDialog();
      break;
    case TodoMenuAction.clearAll:
      _showClearAllDialog();
      break;
    case TodoMenuAction.showCompleted:
      setState(() {
        _showCompleted = !_showCompleted;
      });
      break;
    case TodoMenuAction.settings:
      break;
    case TodoMenuAction.about:
      _showAboutDialog();
      break;
  }
}
```

```
void _handleContextMenuAction(TodoItemAction action, int index) {
  final todo = _todos[index];
  switch (action) {
    case TodoItemAction.complete:
      setState(() {
        todo.isCompleted = !todo.isCompleted;
      });
      break;
    case TodoItemAction.edit:
      _showEditDialog(index);
      break;
    case TodoItemAction.priority:
      _showPriorityDialog(index);
      break;
    case TodoItemAction.delete:
      _showDeleteDialog(index);
      break;
  }
}

void _showAddDialog() {
  _addController.clear();
  showDialog(
    context: context,
    builder: (context) {
      return AlertDialog(
        title: const Text('Tambah Task Baru'),
        content: TextField(
          controller: _addController,
          autofocus: true,
          decoration: const InputDecoration(
            hintText: 'Masukkan task baru',
            border: OutlineInputBorder(),
          ),
        ),
        actions: [
          TextButton(
            onPressed: () => Navigator.pop(context),
            child: const Text('Batal'),
          ),
          TextButton(
            onPressed: () {
              if (_addController.text.trim().isNotEmpty) {
                setState(() {
                  _todos.add(TodoItem(_addController.text));
                });
                Navigator.pop(context);
              }
            },
            child: const Text('Tambah'),
          ),
        ],
      );
    },
  );
}
```

```
void _showEditDialog(int index) {
  final controller = TextEditingController(text:
_todos[index].title);
  showDialog(
    context: context,
    builder: (context) {
      return AlertDialog(
        title: const Text('Edit Task'),
        content: TextField(
          controller: controller,
          autofocus: true,
          decoration: const InputDecoration(
            border: OutlineInputBorder(),
          ),
        ),
        actions: [
          TextButton(
            onPressed: () => Navigator.pop(context),
            child: const Text('Batal'),
          ),
          TextButton(
            onPressed: () {
              if (controller.text.trim().isNotEmpty) {
                setState(() {
                  _todos[index].title = controller.text;
                });
                Navigator.pop(context);
              }
            },
            child: const Text('Simpan'),
          ),
        ],
      );
    },
  );
}
```

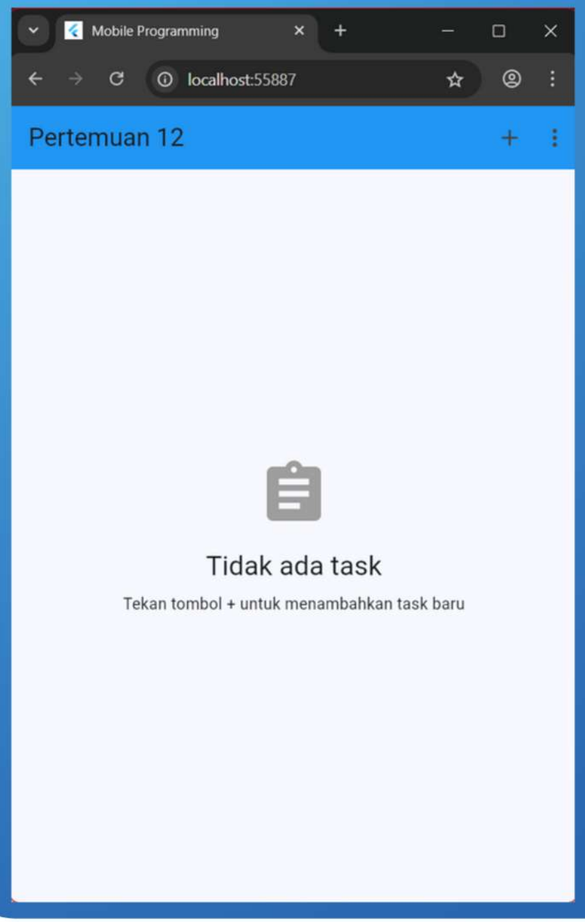
SOURCE CODE

```
void _showPriorityDialog(int index) {
  showDialog(
    context: context,
    builder: (context) {
      return AlertDialog(
        title: const Text('Set Prioritas'),
        content: Column(
          mainAxisSize: MainAxisSize.min,
          children: [
            ListTile(
              title: const Text('Tinggi'),
              leading: const Icon(Icons.flag, color: Colors.red),
              onTap: () {
                setState(() {
                  _todos[index].priority = Priority.high;
                });
                Navigator.pop(context);
              },
            ),
            ListTile(
              title: const Text('Sedang'),
              leading: const Icon(Icons.flag, color: Colors.orange),
              onTap: () {
                setState(() {
                  _todos[index].priority = Priority.medium;
                });
                Navigator.pop(context);
              },
            ),
            ListTile(
              title: const Text('Rendah'),
              leading: const Icon(Icons.flag, color: Colors.green),
              onTap: () {
                setState(() {
                  _todos[index].priority = Priority.low;
                });
                Navigator.pop(context);
              },
            ),
          ],
        ),
      );
    },
  );
}
```

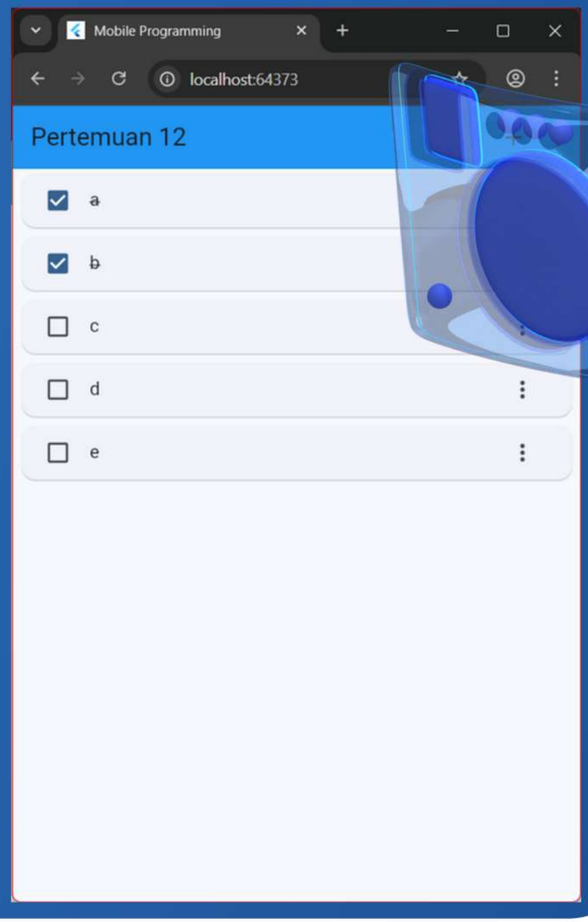
```
void _showDeleteDialog(int index) {
  showDialog(
    context: context,
    builder: (context) {
      return AlertDialog(
        title: const Text('Hapus Task?'),
        content: Text('Yakin ingin menghapus "${_todos[index].title}"?'),
        actions: [
          TextButton(
            onPressed: () => Navigator.pop(context),
            child: const Text('Batal'),
          ),
          TextButton(
            onPressed: () {
              setState(() {
                _todos.removeAt(index);
              });
              Navigator.pop(context);
            },
            child: const Text('Hapus', style: TextStyle(color: Colors.red)),
          ),
        ],
      );
    },
  );
}

void _showClearAllDialog() {
  showDialog(
    context: context,
    builder: (context) {
      return AlertDialog(
        title: const Text('Hapus Semua Task?'),
        content: const Text('Yakin ingin menghapus semua task?'),
        actions: [
          TextButton(
            onPressed: () => Navigator.pop(context),
            child: const Text('Batal'),
          ),
          TextButton(
            onPressed: () {
              setState(() {
                _todos.clear();
              });
              Navigator.pop(context);
            },
            child: const Text('Hapus Semua', style: TextStyle(color:
Colors.red))),
          ],
        );
    },
  );
}
```

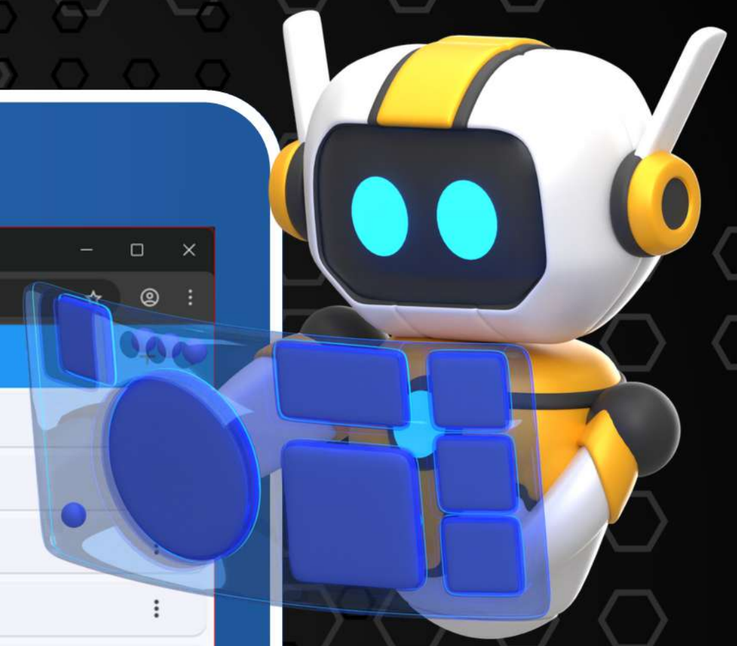
```
void _showAboutDialog() {
  showDialog(
    context: context,
    builder: (context) {
      return AlertDialog(
        title: const Text('Tentang Aplikasi'),
        content: const Column(
          mainAxisSize: MainAxisSize.min,
          children: [
            Text('Mobile Programming', style: TextStyle(fontSize: 20)),
            SizedBox(height: 16),
            Text('Versi 1.0.0'),
            Text('Dibuat dengan Flutter'),
          ],
        ),
        actions: [
          TextButton(
            onPressed: () => Navigator.pop(context),
            child: const Text('Tutup'),
          ),
        ],
      );
    },
  );
}

class TodoItem {
  String title;
  bool isCompleted;
  Priority priority;
  DateTime? dueDate;

  TodoItem(this.title, {this.isCompleted = false, this.priority = Priority.low});
}

enum Priority {
  low,
  medium,
  high,
}
```

HASIL


Mobile Programming
localhost:55887
Pertemuan 12
Tidak ada task
Tekan tombol + untuk menambahkan task baru


Mobile Programming
localhost:64373
Pertemuan 12
a
b
c
d
e


Mobile Programming
localhost:64373
Pertemuan 12
a
b
c
d
e



Universitas Pamulang
Sistem Informasi

TUGAS

Lakukan instalasi ke perangkat ponsel kalian dan dokumentasikan

Batas pengumpulan 28 Juni 2025

Universitas Pamulang
Sistem Informasi

Thank You!