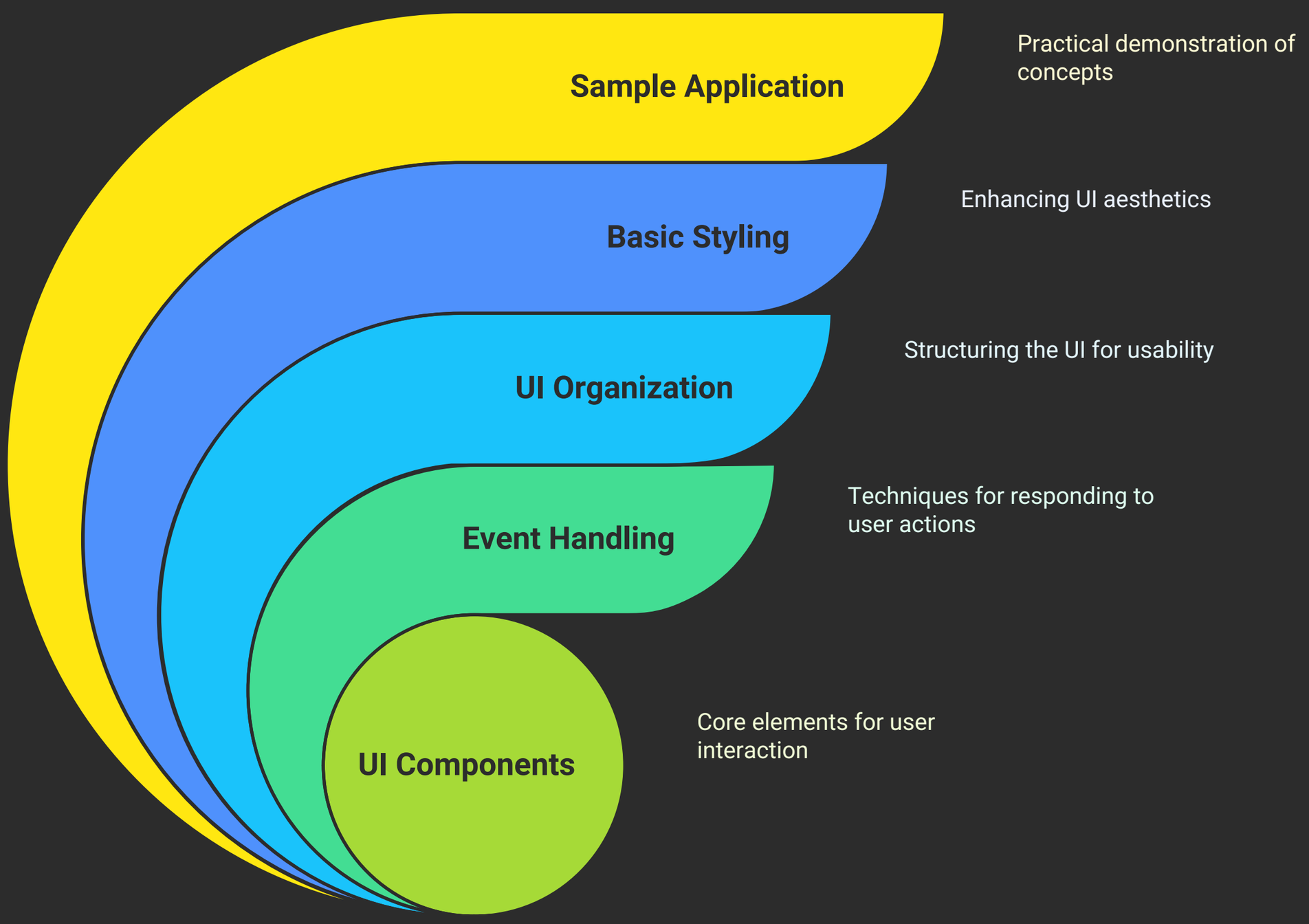
Lecture #3 - Android UI Components and Event Handling

This document provides an overview of fundamental Android UI components and event handling techniques, as introduced in Lecture #3. It details a sample application that showcases a simple user registration form, highlighting various input fields, data collection methods, and user feedback mechanisms. The lecture emphasizes the importance of understanding UI organization, event handling, and basic styling in Android development.



Overview

This lecture introduces fundamental Android UI components and event handling techniques. The sample application demonstrates a simple form with multiple input fields, data collection, and user feedback mechanisms.

Topics Covered

- EditText fields for user input
- Button implementation and click handling
- Toast messages for user feedback
- View binding and **findViewById**
- LinearLayout for UI organization
- Basic styling with XML attributes
- Input validation and data processing

Sample Application: User Registration Form

The application demonstrates a simple registration form with the following features:

- Input fields for first name, last name, address, and phone number
- Submit button that collects and processes form data
- Toast notification displaying the submitted information
- Form clearing after submission
- Basic UI styling with colors and spacing

Key Code Concepts

View Binding and Component Initialization

```
FirstName = findViewById(R.id.first_name);
LastNam = findViewById(R.id.last_name);
Address = findViewById(R.id.address);
Phone = findViewById(R.id.phone);
Submit = findViewById(R.id.btn);
```

Event Handling with OnClickListener

```
Submit.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        String lname, fname, address, phone;
        fname = FirstName.getText().toString();
        lname = LastNam.getText().toString();
        address = Address.getText().toString();
        phone = Phone.getText().toString();

        String data = "First Name : "+fname+", \nLast Name : "+lname+", \n
Address :  "+address+", \nPhone Number : "+phone+";";

        Toast.makeText(MainActivity.this, data, Toast.LENGTH_LONG).show();

        // Clear form fields after submission
        FirstName.setText("");
        LastNam.setText("");
        Address.setText("");
        Phone.setText("");
    }
});
```

XML Layout Structure

The layout uses **LinearLayout** with vertical orientation to organize input fields and button:

```
<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:padding="20px"
    android:orientation="vertical">

    <!-- UI components -->

</LinearLayout>
```

Implementation Notes

- The app follows a simple single-activity architecture.
- Form data is collected but not persistently stored.
- Input validation is minimal in this implementation.
- UI is styled with basic colors and text formatting.

Key Takeaways

- Understanding the relationship between XML layouts and Java code.
- Implementation of event listeners for user interaction.
- Proper gathering and processing of user input.

- Providing feedback to users with Toast messages.
- Basic form handling techniques.

Running the Application

1. Open the project in Android Studio.
2. Run the application on an emulator or physical device.
3. Enter information in the form fields.
4. Click the Submit button to see the toast message with collected data.