UNIT-3:

Understanding Sudoku Game:

Understanding Sudoku Game

Sudoku is a popular logic-based number puzzle played on a 9x9 grid. The grid is divided into nine 3x3 smaller boxes. The goal is to fill every empty cell with a number from 1 to 9.

Rules of Sudoku

1. Each row must contain all digits from 1 to 9 without repetition.
2. Each column must contain all digits from 1 to 9 without repetition.
3. Each 3x3 box must also contain all digits from 1 to 9 without repetition.

How to Play

- The puzzle starts with some numbers already filled in.
- Using logical reasoning, you fill in the missing numbers.
- The challenge is to complete the grid so that all rows, columns, and boxes follow the rules.
- No guessing is needed if the puzzle is valid and solvable.

Purpose of Sudoku

- Improves logical thinking and problem-solving skills.
- Enhances concentration and patience.
- Provides a fun and engaging mental exercise.

Designing By Declaration:

Designing by declaration is a programming approach where you define **what** you want rather than **how** to do it. Instead of writing detailed step-by-step instructions, you declare the desired outcome or structure, and the system figures out the process to achieve it.

Procedural Design

- **Definition:** Focuses on **how** to perform tasks by writing sequences of instructions or procedures.
- **Approach:** Step-by-step, telling the computer exactly what to do and how to do it.
- **Characteristics:**
    - Uses functions, loops, and conditional statements.
    - Programmer controls the flow of execution explicitly.
    - Example languages: C, Java (imperative parts), Python (imperative style).

- **Example:** Writing code to calculate the sum of numbers by iterating through each element.

Declarative Design

- **Definition:** Focuses on **what** the desired result is, without specifying how to achieve it.
- **Approach:** You declare the outcome or rules, and the system decides how to implement them.
- **Characteristics:**
    - More about describing properties, constraints, or structure.
    - The underlying system manages control flow.
    - Example languages: SQL, HTML, XML, functional programming languages.
- **Example:** Writing a SQL query to select all users over 18 without specifying how to traverse the database.

Creating the Opening Screen

The **opening screen** is the first screen users see when they launch an app. It usually shows the app logo, name, or a brief introduction before moving to the main content.

1. Splash Screen

The splash screen is the first screen shown when an app launches. It usually displays the app logo or branding briefly while the app loads necessary resources in the background. It improves user experience by avoiding a blank screen and reinforcing brand identity.

2. Introduction Navigation

Introduction navigation guides new users through key features of the app using a series of screens or slides. This helps users understand how to use the app effectively. It often includes “Next” or “Skip” buttons to navigate through the intro.

3. Call to Action (CTA)

A Call to Action is a prompt that encourages users to take a specific action, such as “Sign Up,” “Get Started,” or “Learn More.” CTAs should be clear, visible, and compelling to guide user behavior.

4. Visual Design

Visual design focuses on the app’s look and feel, including colors, typography, layout, icons, and images. Good visual design improves usability, conveys brand personality, and creates an appealing user experience.

5. Functionality

Functionality refers to how the app works—its features, navigation, responsiveness, and performance. A functional app meets user needs smoothly and without bugs or crashes.

6. Loading Indicators

Loading indicators (like spinners or progress bars) inform users that the app is processing data or loading content. They reduce user frustration by showing that the app is working and hasn't frozen.

What is an About Box?

An **About Box** is a dialog or screen that provides information about the app, such as the app name, version, developer info, copyright, and sometimes links or contact details.

Steps to Implement an About Box

1. **Design the Layout:**
   Create a simple layout with text views for the app name, version, developer name, and any other details you want to show.
2. **Create the About Box UI:**
   - In desktop apps, this might be a modal dialog box.
   - In mobile apps, it can be a separate activity, fragment, or dialog.
3. **Trigger the About Box:**
   Add an option in the menu or a button (e.g., in Settings) that users can tap to open the About Box.
4. **Show the About Box:**
   When the user selects the About option, display the About Box UI with the relevant info.
5. Define an XML layout (about_box.xml) with app info.
6. Create an `AboutActivity` or use a dialog.
7. Launch it from a menu item:

```
// Launch About Activity

Intent intent = new Intent(this, AboutActivity.class);
startActivity(intent);
```

1. **Create layout `activity_about.xml`:**

```
<TextView
        android:id="@+id/about_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="@string/about_text"
        android:textSize="16sp" />
```

2. **Add string resource in `res/values/strings.xml`:**

```
<string name="about_text">
   Sudoku App\nVersion 1.0\nDeveloped by Your Name\n© 2025
</string>
```

3. Create `AboutActivity.java`:

```
public class AboutActivity extends AppCompatActivity {
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.activity_about);

      TextView aboutText = findViewById(R.id.about_text);
```