To Do App (using Python)

This code appears to be a Python script for creating a simple To-Do list application using a graphical user interface (GUI) framework/library called "flet."

Flet's built-in components and features effectively to create a functional and user-friendly to-do app.

1. ***Importing Necessary Modules:***

```
import flet
from flet import (
    Checkbox,
    Column,
    FloatingActionButton,
    IconButton,
    OutlinedButton,
    Page,
    Row,
    Tab,
    Tabs,
    Text,
    TextField,
    UserControl,
    colors,
    icons,
)
```

This section imports various modules and controls from the "flet" library, which is assumed to be a GUI framework. These modules provide the building blocks for creating the user interface of the To-Do list application.

2. ***Defining the Task Class:***

The Task class is responsible for representing individual tasks within the To-Do list. Let's break down its key components:

task_name: This attribute stores the name or description of the task.

completed: A boolean attribute indicating whether the task is completed or not.

task_status_change and task_delete: These are callback functions passed to the Task class, which are called when the task's status changes or when it is deleted.

The Task class has several methods:

build: This method creates the visual representation of a task, including a checkbox for completion status and buttons for editing and deleting the task. ***edit_clicked, save_clicked, status_changed, and delete_clicked are event handlers that respond to user interactions.***

3. ***Defining the TodoApp Class:***

The TodoApp class is the main application window for the To-Do list. Here are its primary components:

new_task: This is a text input field where users can enter new tasks.

tasks: A container (Column) to hold the individual Task instances.

filter: A set of tabs for filtering tasks by status (all, active, completed).

items_left: A text element displaying the count of active tasks.

The TodoApp class also has methods for adding, updating, and deleting tasks, as well as handling tab changes and clearing completed tasks. The update method is responsible for dynamically updating the UI based on user interactions.

4. ***Building the GUI:***

In both the Task and TodoApp classes, the build method is used to create the graphical user interface by defining the structure and arrangement of various controls, including Rows, Columns, TextFields, Buttons, and more.

5. ***Event Handling:***

Event handling methods in the Task and TodoApp classes respond to user interactions such as clicking buttons, changing checkbox states, selecting tabs, etc.

For example, when a user clicks the "Edit" button for a task, the edit_clicked method is called to allow the user to edit the task name.

6. ***Updating the UI:***

The update method is called to dynamically update the UI elements in response to changes in the To-Do list, such as adding or deleting tasks or changing their completion status.

It also controls the visibility of tasks based on the selected filter tab.

7. ***Main Function:***

The main function sets up the initial state of the application. It sets the page title, alignment, and scroll behavior. Then, it creates an instance of the TodoApp class and adds it to the application's page.

8. Running the Application:

The last line of code uses flet.app(target=main) to start the application. It executes the main function, initializing and displaying the To-Do list application.