Innovation and Integrative Research Center Journal

ISSN: 2584-1491 |www.iircj.org

Volume-1|Issue-3| Nov-2023|Page 11- 17

Enhancing User Experience with Customized Progress Indicators in Flutter

[1]*Rahul Chandrakar*

Assistant Professor,

Department of Information Technology

Pt. Ramsakha Upadhyay College, Gudhiyari,

Raipur, Chhattisgarh, India.

rahulchandrakar52@gmail.com

Abstract:

Progress indicators play a crucial role in modern user interfaces, providing visual feedback to users about ongoing processes and enhancing user experience. In Flutter, a popular cross-platform framework for mobile app development, developers can utilize various types of progress indicators, including circular and linear indicators, to indicate the progress of tasks. This research paper explores the implementation and customization of circular progress indicators in Flutter, focusing on both indeterminate and determinate types. Additionally, it discusses the significance of progress indicators in app development and provides practical guidance on integrating and customizing them effectively.

Keyword: Flutter, Progress Indicator, Dart, Mobile Application.

Introduction:

In today's fast-paced digital world, users expect applications to provide immediate feedback and seamless experiences. Progress indicators are essential UI elements that inform users about ongoing processes, such as loading content, downloading files, or processing data. These indicators reassure users that their actions are being processed and help manage expectations regarding wait times.

Flutter, developed by Google, has gained popularity among developers for its ability to build beautiful and high-performance mobile applications for Android, iOS, web, and desktop platforms from a single codebase. One of the key components of Flutter's UI toolkit is the progress indicator widget, which comes in various forms, including circular and linear types.

This research paper focuses on the Circular Progress Indicator in Flutter, exploring its types, customization options, and implementation techniques. By understanding how to effectively utilize and tailor circular

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 | www.iircj.org
Volume-1|Issue-3| Nov-2023|Page 11- 17

progress indicators, developers can create visually appealing and user-friendly applications.

Circular Progress Indicator:

The *CircularProgressIndicator* widget in Flutter is designed to display progress along a circular path. It serves as a visual representation of ongoing tasks, spinning to indicate activity and completion status. There are two main types of circular progress indicators: indeterminate and determinate.

Indeterminate Circular Progress Indicator:

The indeterminate circular progress indicator does not display a specific value at any given time. Instead, it continuously spins to signify that progress is being made without indicating how much work remains. In Flutter, creating an indeterminate progress bar involves setting the value property to null.

```
CircularProgressIndicator();
```

Determinate Circular Progress Indicator:

In contrast, the determinate circular progress indicator provides a specific value at each instance, indicating the progress completed. The value property of the CircularProgressIndicator widget ranges from 0.0 to 1.0, where 0.0 represents the start of progress, and 1.0 signifies completion.

```
CircularProgressIndicator(
  value: 0.7,
);
```

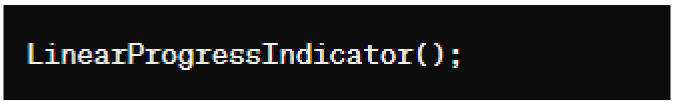
Customization options for circular progress indicators include properties such as backgroundColor, valueColor, and strokeWidth. These properties allow developers to adjust the appearance of the indicator to match the app's design and branding.

Linear Progress Indicator:

In addition to circular progress indicators, Flutter also offers linear progress indicators for displaying progress in a linear direction or along a line. Similar to circular indicators, linear indicators can be either indeterminate or determinate.

Indeterminate Linear Progress Indicator:

The indeterminate linear progress indicator, like its circular counterpart, does not display a specific value at any given time. It signifies ongoing progress without indicating the remaining work.



```
LinearProgressIndicator();
```

Determinate Linear Progress Indicator:

Innovation and Integrative Research Center Journal
ISSN: 2584-1491 |www.iircj.org
Volume-1|Issue-3| Nov-2023|Page 11- 17

On the other hand, the determinate linear progress indicator provides a specific value at each instance, indicating the progress completed.

Similar to circular progress indicators, linear progress indicators can be customized using properties such as backgroundColor, valueColor, and minHeight.

Implementing Progress Indicators in Flutter:

To implement progress indicators in a Flutter application, developers can follow a straightforward process:

1. Import the material.dart package to utilize Flutter's progress indicator widgets.
2. Create a new MaterialApp widget as the root of the application.
3. Define a StatelessWidget to display the progress indicators.
4. Run the application to see the progress indicators in action.

By following these steps, developers can quickly integrate both circular and linear progress indicators into their Flutter applications.

Customizing Progress Indicators:

Flutter provides various customization options for progress indicators, allowing developers to tailor the appearance of indicators to suit their application's design. Common customization options include changing the background color, value color, stroke width, and minimum height.

Changing the Background Color:

Developers can adjust the background color of progress indicators using the backgroundColor property.

For Circular:

```
CircularProgressIndicator(
  backgroundColor: Colors.redAccent,
);
```

For Linear:

```
LinearProgressIndicator(
  backgroundColor: Colors.redAccent,
);
```

Changing the Value Color:

The value color represents the color of the progress indicator's value. Developers can modify the value color using the valueColor property.

For Circular:

```
CircularProgressIndicator(
  valueColor:
AlwaysStoppedAnimation(Colors.green),
);
```

For Linear:

```
LinearProgressIndicator(
  valueColor: AlwaysStoppedAnimation(Colors.green),
);
```

Changing the Stroke Width:

The strokeWidth property defines the width of the line that draws the circle in circular progress indicators. Developers can adjust the stroke width to achieve the desired appearance.

```
CircularProgressIndicator(
  strokeWidth: 10,
);
```

Changing the Minimum Height:

In linear progress indicators, the minHeight property specifies the minimum height of the line that draws the indicator.

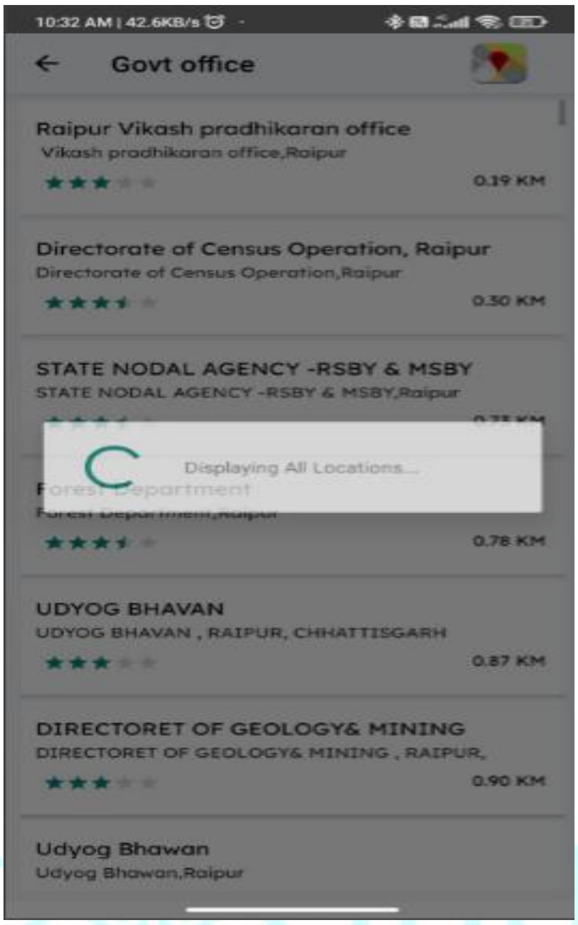
```
LinearProgressIndicator(
  minHeight: 20,
);
```

By customizing these properties, developers can create progress indicators that align with their application's visual identity and enhance the user experience.

Putting it all together:

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

void main() {
  runApp(MyApp());
}

class MyApp extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: ProgressIndicatorsDemo(),
    );
  }
}

class ProgressIndicatorsDemo extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(
        title: Text('Progress Indicators Demo'),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            CircularProgressIndicatorDemo(),
            SizedBox(height: 20),
            LinearProgressIndicatorDemo(),
          ],
        ),
      ),
    );
  }
}

class CircularProgressIndicatorDemo extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        Text(
          'Circular Progress Indicator',
          style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
        ),
        SizedBox(height: 10),
        CircularProgressIndicator(),
        SizedBox(height: 10),
        Text('Indeterminate'),
        SizedBox(height: 10),
        CircularProgressIndicator(
          value: 0.5,
          backgroundColor: Colors.grey,
          valueColor: AlwaysStoppedAnimation<Color>(Colors.blue),
          strokeWidth: 6,
        ),
        SizedBox(height: 10),
        Text('Determinate'),
      ],);}}

class LinearProgressIndicatorDemo extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Column(
      children: [
        Text(
          'Linear Progress Indicator',
          style: TextStyle(fontSize: 20, fontWeight: FontWeight.bold),
        ),
        SizedBox(height: 10),
        LinearProgressIndicator(),
        SizedBox(height: 10),
        Text('Indeterminate'),
        SizedBox(height: 10),
        LinearProgressIndicator(
          value: 0.7,
          backgroundColor: Colors.grey,
          valueColor: AlwaysStoppedAnimation<Color>(Colors.green),
          minHeight: 10,
        ),
        SizedBox(height: 10),
        Text('Determinate'),
      ],);}}
```

Output:


10:32 AM | 42.6KB/s

Govt office

Raipur Vikash pradhikaran office
Vikash pradhikaran office,Raipur
★★★☆☆ 0.19 KM

Directorate of Census Operation, Raipur
Directorate of Census Operation,Raipur
★★★½☆ 0.30 KM

STATE NODAL AGENCY -RSBY & MSBY
STATE NODAL AGENCY -RSBY & MSBY,Raipur
0.73 KM

Displaying All Locations...

Forest Department
Forest Department,Raipur
★★★½☆ 0.78 KM

UDYOG BHAVAN
UDYOG BHAVAN , RAIPUR, CHHATTISGARH
★★★☆☆ 0.87 KM

DIRECTORET OF GEOLOGY& MINING
DIRECTORET OF GEOLOGY& MINING , RAIPUR,
★★★☆☆ 0.90 KM

Udyog Bhawan
Udyog Bhawan,Raipur

Conclusion:

Progress indicators play a vital role in modern application development, providing users with visual feedback about ongoing processes. In Flutter, developers have access to a versatile set of progress indicator widgets, including circular and linear types, which can be easily integrated and customized.

In this research paper, we explored the Circular Progress Indicator in Flutter, discussing its types, customization options, and implementation techniques. By understanding the significance of progress indicators and mastering their implementation, developers can create applications that offer a seamless and engaging user experience.

By following the practical guidance provided in this paper, developers can leverage Flutter's progress indicator widgets to enhance the visual appeal and functionality of their applications. Experimenting with different customization options allows developers to create unique and visually appealing progress indicators that align with their application's design guidelines.

In conclusion, progress indicators are essential elements in Flutter app development, and by incorporating them effectively, developers can create applications that delight users and drive engagement.

References:

1. Watanabe, Y., Suzuki, S., Sugihara, M., & Sueoka, Y. (2002). An experimental study of paper flutter. *Journal of fluids and Structures*, *16*(4), 529-542.

2. Garrick, I. E., and Wilmer H. Reed III. "Historical development of aircraft flutter." *Journal of Aircraft* 18, no. 11 (1981): 897-912.

3. Windmill, Eric. *Flutter in action*. Simon and Schuster, 2020.

4. Watanabe, Y., Isogai, K., Suzuki, S. and Sugihara, M., 2002. A theoretical study of paper flutter. *Journal of fluids and structures*, *16*(4), pp.543-560.