Animations:-

- **In Flutter, an `Animation` object knows nothing about what is onscreen. An `Animation` is an abstract class that understands its current value and its state (completed or dismissed). One of the more commonly used animation types is `Animation<double>`.An `Animation` object sequentially generates interpolated numbers between two values over a certain duration.**

1. CurvedAnimation
2. TweenAnimation
3. FooAnimation
4. Hero Animation
5. Lottie Animation

1.Curved Animation:-

- **CurvedAnimation is useful when you want to apply a non-linear Curve to an animation object, especially if you want different curves when the animation is going forward vs when it is going backward.**
- **Depending on the given curve, the output of the CurvedAnimation could have a wider range than its input. For example, elastic curves such as Curves.elasticIn will significantly overshoot or undershoot the default range of 0.0 to 1.0.**

Example:-

```
import 'package:example/flip_drawer.dart';
import 'package:example/page.dart';
import 'package:example/shared.dart';
import 'package:example/slide_drawer.dart';
import 'package:flutter/material.dart';

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

class App extends StatefulWidget {
  static _AppState? of(BuildContext context) =>
      context.findAncestorStateOfType<_AppState>();

  @override
  _AppState createState() => _AppState();
}

class _AppState extends State<App> {
  String _title = 'Curved Animation';
  Key _key = UniqueKey();
```

```
  restart() {
    setState(() {
      _key = UniqueKey();
    });
  }

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      key: _key,
      title: 'Curved Animation Controller Demo',
      theme: ThemeData(
        primarySwatch: Colors.teal,
        visualDensity: VisualDensity.adaptivePlatformDensity,
      ),
      home: isFlipDrawer
          ? FlipDrawer(title: _title, drawer: MenuDrawer(), child: HomePage())
          : SlideDrawer(drawer: MenuDrawer(), child: HomePage(title: _title)),
    );
  }
}

class MenuDrawer extends StatelessWidget {
  BoxDecoration get _gradient => BoxDecoration(
        gradient: LinearGradient(
          begin: Alignment.topLeft,
          end: Alignment.bottomRight,
          stops: [0.0, 1.0],
          colors: [
            Color(0xFF43CEA2),
            Color(0xFF1D6DBD),
          ],
        ),
      );

  BoxDecoration get _color => BoxDecoration(
        color: Colors.teal[500],
      );

  @override
  Widget build(BuildContext context) {
    return Material(
      shadowColor: Colors.transparent,
      borderOnForeground: false,
      child: Container(
        decoration: isSlideDrawer ? _gradient : _color,
        child: SafeArea(
          child: Theme(
            data: ThemeData(brightness: Brightness.dark),
            child: Column(
              mainAxisAlignment: MainAxisAlignment.center,
              crossAxisAlignment: CrossAxisAlignment.start,
              mainAxisSize: MainAxisSize.max,
              children: [
```

```
            if (!isSlideDrawer)
              ListTile(
                leading: Icon(Icons.adjust),
                title: Text('Slide Drawer'),
                onTap: () {
                  type = DrawerType.slide;
                  App.of(context)!.restart();
                },
              ),
            if (!isFlipDrawer)
              ListTile(
                leading: Icon(Icons.adjust),
                title: Text('Flip Drawer'),
                onTap: () {
                  type = DrawerType.flip;
                  App.of(context)!.restart();
                },
              ),
            ListTile(
              leading: Icon(Icons.rss_feed),
              title: Text('News'),
            ),
            ListTile(
              leading: Icon(Icons.favorite_border),
              title: Text('Favourites'),
            ),
            ListTile(
              leading: Icon(Icons.map),
              title: Text('Map'),
            ),
            ListTile(
              leading: Icon(Icons.settings),
              title: Text('Settings'),
            ),
            ListTile(
              leading: Icon(Icons.person_outline),
              title: Text('Profile'),
            ),
          ],
        ),
      ),
    ),
  );
 }
}
```

2.Tween Animation:-

Tween is useful if you want to interpolate across a range.

To use a Tween object with an animation, call the Tween object's animate method and pass it the Animation object that you want to modify.

Example:-

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


void main() => runApp(MyApp());


class MyApp extends StatefulWidget {
 _MyApp createState() => _MyApp();
}


class _MyApp extends State<MyApp> with SingleTickerProviderStateMixin
{
 Animation<double> animation;
 AnimationController controller;


 @override
 void initState() {
   super.initState();


   controller =
       AnimationController(vsync: this, duration: Duration(seconds:
2));
   animation = Tween<double>(begin: 0, end: 300).animate(controller)
     ..addListener(() {
       setState(() {});
```

```
      })
      ..addStatusListener((status) {
        if (status == AnimationStatus.completed) {
          controller.reverse();
        } else if (status == AnimationStatus.dismissed) {
          controller.forward();
        }
      });

    controller.forward();
  }

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: Center(
        child: Container(
     color: Colors.white,
     height: animation.value,
     width: animation.value,
    ),
   ),
  );
}

@override
void dispose() {
 controller.dispose();
 super.dispose();
```

```
}
}
```

3.Hero Animation:-

When a PageRoute is pushed or popped with the Navigator, the entire screen's content is replaced. An old route disappears and a new route appears. If there's a common visual feature on both routes then it can be helpful for orienting the user for the feature to physically move from one page to the other during the routes' transition.

Such an animation is called a *hero animation*. The hero widgets "fly" in the Navigator's overlay during the transition and while they're in-flight they're, by default, not shown in their original locations in the old and new routes.

Example:-

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


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


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

  @override
  Widget build(BuildContext context) {
    return const MaterialApp(
      home: HeroExample(),
    );
  }
}
```

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

  @override
  Widget build(BuildContext context) {
    return Scaffold(
      appBar: AppBar(title: const Text('Hero Sample')),
      body: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: <Widget>[
          const SizedBox(height: 20.0),
          ListTile(
            leading: const Hero(
              tag: 'hero-rectangle',
              child: BoxWidget(size: Size(50.0, 50.0)),
            ),
            onTap: () => _gotoDetailsPage(context),
            title: const Text(
              'Tap on the icon to view hero animation transition.',
            ),
          ),
        ],
      ),
    );
  }

void _gotoDetailsPage(BuildContext context) {
  Navigator.of(context).push(MaterialPageRoute<void>(
    builder: (BuildContext context) => Scaffold(
      appBar: AppBar(
        title: const Text('Second Page'),
      ),
      body: const Center(
        child: Hero(
          tag: 'hero-rectangle',
          child: BoxWidget(size: Size(200.0, 200.0)),
        ),
      ),
```

```
    ),
  ));
}
```

```
}

class BoxWidget extends StatelessWidget {
  const BoxWidget({super.key, required this.size});

  final Size size;

  @override
  Widget build(BuildContext context) {
    return Container(
      width: size.width,
      height: size.height,
      color: Colors.blue,
    );
  }
}
```

4.Lottie Animation:-

Lottie is a mobile library for Android and iOS that parses Adobe After Effects animations exported as json with Bodymovin and renders them natively on mobile!

This repository is an unofficial conversion of the Lottie-android library in pure Dart.

Example:-

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

import 'package:lottie/lottie.dart';


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


class MyApp extends StatelessWidget {

    const MyApp({super.key});

    @override
    Widget build(BuildContext context) {
      return MaterialApp(
        home: Scaffold(
          body: ListView(
            children: [
              // Load a Lottie file from your assets
```

```
            Lottie.asset('assets/LottieLogo1.json'),

            // Load a Lottie file from a remote url
            Lottie.network(
              'https://raw.githubusercontent.com/xvrh/lottie-
        flutter/master/example/assets/Mobilo/A.json'),

            // Load an animation and its images from a zip file
            Lottie.asset('assets/lottiefiles/angel.zip'), ], )),);}}
```

5.Foo Animation:-

They're called AnimatedFoo widgets, where Foo is the animated property.

Most of them are animated versions of the widgets you already know and use, like Container/AnimatedContainer, Padding/AnimatedPadding, Positioned/AnimatedPositioned etc.

Example:-

```
AnimatedPositioned(
 top: selectedItemIndex * itemHeight,
 left: 0,
 right: 0,
 duration: const Duration(milliseconds: 200),
 curve: Curves.easeInOut,
 child: //...
),
//...
AnimatedContainer(
 duration: const Duration(milliseconds: 200),
 curve: Curves.easeInOut,
 decoration: BoxDecoration(
   color: selectedItemIndex == i ? yellow : pink,
   border: Border.all(
     color: selectedItemIndex == i
         ? Colors.white
         : Colors.transparent,
     width: 2,
```

```
  ),
),
child: AnimatedDefaultTextStyle(
  duration: const Duration(milliseconds: 200),
  style: TextStyle(
    color: selectedItemIndex == i
        ? Colors.black
        : Colors.white,
  ),
  child: const Text('Featured!'),),),
```

29.Animated Container:-

- In Flutter a container is a simple widget with well-defined properties like height, width, and color, etc. **The AnimatedContainer widget is a simple container widget with animations**.
- These types of widgets can be animated by altering the values of their properties which are the same as the Container widget.
- Example:-

```
import 'dart:math';

import 'package:flutter/material.dart';

void main() => runApp(AnimatedContainerApp());

class AnimatedContainerApp extends StatefulWidget {
@override
```

```
_AnimatedContainerAppState createState() =>
_AnimatedContainerAppState();
}

class _AnimatedContainerAppState extends
State<AnimatedContainerApp> {

double _width = 70;
double _height = 70;
Color _color = Colors.green;
BorderRadiusGeometry _borderRadius = BorderRadius.circular(10);

@override
Widget build(BuildContext context) {
        return MaterialApp(
        home: Scaffold(
                appBar: AppBar(
                title: Text('GeeksForGeeks'),
                backgroundColor: Colors.green,
                ),
                body: Center(
                child: AnimatedContainer(
                        width: _width,
                        height: _height,
                        decoration: BoxDecoration(
                        color: _color,
                        borderRadius: _borderRadius,
                        ),
                        duration: Duration(seconds: 1),
                        curve: Curves.fastOutSlowIn,
                ),
                ),
                floatingActionButton: FloatingActionButton(
                child: Icon(Icons.play_arrow),
                backgroundColor: Colors.green,
                onPressed: () {
                        setState(() {
                        // random generator
                        final random = Random();

                        // random dimension generator
                        _width = random.nextInt(500).toDouble();
                        _height = random.nextInt(500).toDouble();

                        // random color generator
                        _color = Color.fromRGBO(
```

```
                        random.nextInt(300),
                        random.nextInt(300),
                        random.nextInt(300),
                        1,
                    );

                    // random radius generator
                    _borderRadius =

        BorderRadius.circular(random.nextInt(100).toDouble());
                    });
                },
                ),
        ),
        );
}}
```

30.Animated Opacity:-

- The **AnimatedOpacity** makes its child mostly transparent. This class colors its child into a middle buffer and afterward consolidates the child once again into the scene mostly transparent.
- For values of opacity other than 0.0 and 1.0, this class is moderately costly as it needs shading the child into a halfway support. For the value 0.0, the child is just not colored by any means. For the value 1.0, the child is colored without a moderate buffer.
- Example:-

```
import 'package:flutter/cupertino.dart';
import 'package:flutter/material.dart';
class OpacityDemo extends StatefulWidget {
```

```
 @override
 OpacityDemoState createState() => OpacityDemoState();
}
class OpacityDemoState extends State<OpacityDemo> {
 var _opacity = 0.0;
 var _width = 230.0;
 @override
 Widget build(BuildContext context) {
   return Scaffold(
     backgroundColor: Color(0xffffffff),
     appBar: AppBar(
       backgroundColor: Colors.cyan[300],
       title: Text("Flutter AnimatedOpacity Demo"),
       automaticallyImplyLeading: false,
     ),
     body: Center(
       child: GestureDetector(
         onTap: () {
           setState(() {
             _opacity = _opacity == 0.0 ? 1 : 0.0;
           });
         },
         child: Container(
           alignment: Alignment.center,
           height: MediaQuery.of(context).size.height
*0.08,
           width: _width,
           decoration: BoxDecoration(
             borderRadius: BorderRadius.circular(20.0),
             color: Colors.cyan[400],
           ),
           child: AnimatedOpacity(
             duration: Duration(milliseconds: 700),
             curve: Curves.bounceIn,
             opacity: _opacity,
             child: Row(
               mainAxisAlignment:
MainAxisAlignment.spaceBetween,
               children: [
                 Image.asset("assets/devs.jpg",
                   scale: 10,
                   fit: BoxFit.contain,
                 ),
                 Padding(
                   padding: const
EdgeInsets.only(right:30.0),
                   child: Text(
```

```
                    'Flutter Devs',
                    style: TextStyle(color: Colors.white,
                        fontSize: 20.0)
                    ,
                ),
            ),
        ],
    ),
  ),
 ),
),
),
);
}
}
```