各种动画效果

透明动画 AnimatedOpacity

能让子组件进行Opacity(透明度)动画，可指定时长和曲线，有动面结束事件。

边距动画 AnimatedPadding

能让子组件进行 Padding（内边距）动画，可指定时长和曲线，有动面结束事件。

对齐动画 AnimatedAlign

能让子组件进行 Align（对齐）动画，可指定时长和曲线，有动面结束事件。

定位动画 AnimatedPositioned

能让子组件进行 Positioned（定位）动画，可指定时长和曲线，有动面结束事件，只能用于Stack中。

尺寸动画 AnimatedSize

子组件大小发生变化时进行动画渐变，可指定时长、对齐方式、曲线等属性。

方向定位动画 AnimatedPositionedDirectional

能让子组件进行Positioned Directional(方向定位)动画，可指定时长和曲线，有动面结束事件。只能用于Stack之中。

文字样式动画 AnimatedDefaultTextStyle

能让文字组件进行TextStyle样式动画，可指定时长和曲线，有动画结束事件。

物理模型动画 AnimatedPhysicalModel

相关属性变化时具有动画效果的PhysicalModel组件，本质是PhysicalMode l和动画结合的产物。可指定阴影、影深、圆角、动画时长、结束回调等属性。

主题切换动画 AnimatedTheme

主题变化时具有动画效果的组件，本质是Theme组件和动面结合的产物。可指定ThemeData、动面时长、曲线、结束回调等。相当于增强版的Theme组件。

渐变动画构造器 TweenAnimationBuilder

通过渐变器 Tween 对相关属性进行渐变动画，通过 builder 进行局部构建，减少刷新范围。不需要自定义动画器，可指定动画时长、曲线、结束回调。

```
import 'package:flutter/material.dart';
import 'package:flutter_demo/common/style.dart';

class AnimatedEffectWidget extends StatefulWidget {
  const AnimatedEffectWidget({Key? key}) : super(key: key);

  @override
  State<AnimatedEffectWidget> createState() =>
_AnimatedEffectWidgetState();
}

class _AnimatedEffectWidgetState extends State<AnimatedEffectWidget> {
  // 透明动画参数
  double _opacity = 1.0;

  // 边距动画参数
  final EdgeInsets startPadding = const EdgeInsets.all(10);
  final EdgeInsets endPadding = const EdgeInsets.all(30);
  late EdgeInsets _padding;

  // 对齐动画参数
  final Alignment start = const Alignment(0, 0);
  final Alignment end = Alignment.bottomRight;
  late Alignment _alignment;
```

```dart
// 定位动画参数
final startTop = 0.0;
final endTop = 30.0;
var _top = 0.0;

// 定位方向动画参数
final startTop1 = 0.0;
final endTop1 = 30.0;
var _top1 = 0.0;

// 尺寸动画参数
final double starting = 100;
final double ending = 200;
late double _width;

// 文字动画参数
final TextStyle startStyle = const TextStyle(
    color: Colors.blue,
    fontSize: 50,
    shadows: [
      Shadow(offset: Offset(1, 1), color: Colors.black, blurRadius: 3)
    ]);
final TextStyle endStyle = const TextStyle(
    color: Colors.white,
    fontSize: 25,
    shadows: [
      Shadow(offset: Offset(1, 1), color: Colors.purple, blurRadius: 3)
    ]);
late TextStyle _style;

// 物理模块动画参数
bool flag = false;

// 主题切换动画参数
ThemeData startTheme = ThemeData(
  primaryColor: Colors.blue,
  textTheme: const TextTheme(
    headline1: TextStyle(
      color: Colors.white,
      fontSize: 24,
      fontWeight: FontWeight.bold,
    ),
```

```
  ),
);

ThemeData endTheme = ThemeData(
  primaryColor: Colors.red,
  textTheme: const TextTheme(
    headline1: TextStyle(
      color: Colors.black,
      fontSize: 16,
      fontWeight: FontWeight.normal,
    ),
  ),
);

late ThemeData theme;

// 渐变动画构造器参数
Color _value = Colors.red;

@override
void initState() {
  _padding = startPadding;
  _alignment = start;
  _top = startTop;
  _top1 = startTop1;
  _width = starting;
  _style = startStyle;
  theme = startTheme;
  super.initState();
}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      title: const Text('各种动画效果'),
    ),
    body: Container(
      padding: const EdgeInsets.only(
        top: 10,
        left: 10,
        right: 10,
        bottom: 30,
```

```
        ),
        child: SingleChildScrollView(
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.start,
            children: <Widget>[
              const Text(
                '透明动画',
                style: titleStyle,
              ),
              Container(
                margin: const EdgeInsets.symmetric(
                  vertical: 10.0,
                ),
                child: const Text(
                  '能让子组件进行Opacity(透明度)动画，可指定时长和曲线，有动面结束事
件。',
                  style: descStyle,
                ),
              ),
              Switch(
                value: _opacity == 0,
                onChanged: (v) {
                  setState(() {
                    _opacity = v ? 0 : 1.0;
                  });
                },
              ),
              Container(
                color: Colors.grey.withAlpha(77),
                width: 200,
                height: 100,
                child: AnimatedOpacity(
                  // 动画时长
                  duration: const Duration(seconds: 1),
                  // 动画曲线
                  curve: Curves.fastOutSlowIn,
                  // 透明度
                  opacity: _opacity,
                  // 动画结束回调
                  onEnd: () => print('End'),
                  // 子组件
                  child: const Icon(
                    Icons.android,
```

```
                color: Colors.green,
                size: 60,
              ),
            ),
          ),
          const Text(
            '边距动画',
            style: titleStyle,
          ),
          Container(
            margin: const EdgeInsets.symmetric(
              vertical: 10.0,
            ),
            child: const Text(
              '能让子组件进行 Padding（内边距）动画，可指定时长和曲线，有动面结束事件。',
              style: descStyle,
            ),
          ),
          Switch(
            value: _padding == endPadding,
            onChanged: (v) {
              setState(() {
                _padding = v ? endPadding : startPadding;
              });
            },
          ),
          Container(
            color: Colors.grey.withAlpha(77),
            width: 200,
            height: 100,
            child: AnimatedPadding(
              duration: const Duration(seconds: 1),
              curve: Curves.fastOutSlowIn,
              padding: _padding,
              onEnd: () => print('End'),
              child: Container(
                alignment: Alignment.center,
                color: Colors.blue,
                child: const Text(
                  '走进flutter',
                  style: TextStyle(color: Colors.white),
                ),
```

```
          ),
        ),
      ),
      const Text(
        '对齐动画',
        style: titleStyle,
      ),
      Container(
        margin: const EdgeInsets.symmetric(
          vertical: 10.0,
        ),
        child: const Text(
          '能让子组件进行 Align（对齐）动画，可指定时长和曲线，有动面结束事
件。',
          style: descStyle,
        ),
      ),
      Switch(
        value: _alignment == end,
        onChanged: (v) {
          setState(() {
            _alignment = v ? end : start;
          });
        },
      ),
      Container(
        color: Colors.grey.withAlpha(77),
        width: 200,
        height: 100,
        child: AnimatedAlign(
          duration: const Duration(seconds: 1),
          curve: Curves.fastOutSlowIn,
          alignment: _alignment,
          onEnd: () => print('End'),
          child: Container(
            height: 40,
            width: 80,
            alignment: Alignment.center,
            color: Colors.blue,
            child: const Text(
              '走进flutter',
              style: TextStyle(color: Colors.white),
            ),
```

```
            ),
          ),
        ),
        const Text(
          '定位动画',
          style: titleStyle,
        ),
        Container(
          margin: const EdgeInsets.symmetric(
            vertical: 10.0,
          ),
          child: const Text(
            '能让子组件进行 Positioned（定位）动画，可指定时长和曲线，有动面结束事件，只能用于Stack中。',
            style: descStyle,
          ),
        ),
        Switch(
          value: _top == endTop,
          onChanged: (v) {
            setState(() {
              _top = v ? endTop : startTop;
            });
          },
        ),
        Container(
          color: Colors.grey.withAlpha(77),
          width: 200,
          height: 100,
          child: Stack(
            children: <Widget>[
              AnimatedPositioned(
                duration: const Duration(seconds: 1),
                top: _top,
                left: _top * 4,
                child: const Icon(
                  Icons.android,
                  color: Colors.green,
                  size: 50,
                ),
              ),
              AnimatedPositioned(
                duration: const Duration(seconds: 1),
```

```
                top: 50 - _top,
                left: 150 - _top * 4,
                child: const Icon(
                  Icons.android,
                  color: Colors.red,
                  size: 50,
                ),
              )
            ],
          ),
        ),
        const Text(
          '定位方向动画',
          style: titleStyle,
        ),
        Container(
          margin: const EdgeInsets.symmetric(
            vertical: 10.0,
          ),
          child: const Text(
            '能让子组件进行Positioned Directional(方向定位)动画，可指定时长和曲线，有动面结束事件。只能用于Stack之中。',
            style: descStyle,
          ),
        ),
        Switch(
          value: _top1 == endTop1,
          onChanged: (v) {
            setState(() {
              _top1 = v ? endTop1 : startTop1;
            });
          },
        ),
        Container(
          color: Colors.grey.withAlpha(77),
          width: 200,
          height: 100,
          child: Stack(
            children: <Widget>[
              AnimatedPositionedDirectional(
                duration: const Duration(seconds: 1),
                top: _top1,
                start: _top1 * 4,
```

```
                child: const Icon(
                  Icons.android,
                  color: Colors.green,
                  size: 50,
                ),
              ),
              AnimatedPositionedDirectional(
                duration: const Duration(seconds: 1),
                top: 50 - _top1,
                start: 150 - _top1 * 4,
                child: const Icon(
                  Icons.android,
                  color: Colors.red,
                  size: 50,
                ),
              )
            ],
          ),
        ),
        const Text(
          '尺寸动画',
          style: titleStyle,
        ),
        Container(
          margin: const EdgeInsets.symmetric(
            vertical: 10.0,
          ),
          child: const Text(
            '子组件大小发生变化时进行动画渐变，可指定时长、对齐方式、曲线等属
性。',
            style: descStyle,
          ),
        ),
        Switch(
          value: _width == ending,
          onChanged: (v) {
            setState(() {
              _width = v ? ending : starting;
            });
          },
        ),
        Container(
          color: Colors.grey.withAlpha(77),
```

```
            width: 200,
            height: 100,
            alignment: Alignment.center,
            child: AnimatedSize(
              duration: const Duration(seconds: 1),
              curve: Curves.fastOutSlowIn,
              alignment: const Alignment(0, 0),
              child: Container(
                height: 40,
                width: _width,
                alignment: Alignment.center,
                color: Colors.blue,
                child: const Text(
                  '走进flutter',
                  style: TextStyle(color: Colors.white),
                ),
              ),
            ),
          ),
          const Text(
            '文字样式动画',
            style: titleStyle,
          ),
          Container(
            margin: const EdgeInsets.symmetric(
              vertical: 10.0,
            ),
            child: const Text(
              '能让文字组件进行TextStyle样式动画，可指定时长和曲线，有动画结束事
件。',
              style: descStyle,
            ),
          ),
          Switch(
            value: _style == endStyle,
            onChanged: (v) {
              setState(() {
                _style = v ? endStyle : startStyle;
              });
            },
          ),
          Container(
            alignment: Alignment.center,
```

```
                color: Colors.blue.withAlpha(77),
                width: 300,
                height: 100,
                child: AnimatedDefaultTextStyle(
                  // 文字对齐放
                  textAlign: TextAlign.start,
                  // 是否包裹
                  softWrap: true,
                  // 最大行数
                  maxLines: 1,
                  // 溢出模式
                  overflow: TextOverflow.ellipsis,
                  // 动画时长
                  duration: const Duration(seconds: 1),
                  //  动画曲线
                  curve: Curves.fastOutSlowIn,
                  // 文字样式
                  style: _style,
                  // 动画结束回调
                  onEnd: () => print('End'),
                  // 子组件
                  child: const Text(
                    '走进flutter',
                    style: TextStyle(color: Colors.white),
                  ),
                ),
              ),
              const Text(
                '物理模块动画',
                style: titleStyle,
              ),
              Container(
                margin: const EdgeInsets.symmetric(
                  vertical: 10.0,
                ),
                child: const Text(
                  '相关属性变化时具有动画效果的PhysicalModel组件，本质是PhysicalMode l和动画结合的产物。可指定阴影、影深、圆角、动画时长、结束回调等属性。',
                  style: descStyle,
                ),
              ),
              Switch(
                value: flag,
```

```
            onChanged: (v) {
              setState(() {
                flag = v;
              });
            },
          ),
          SizedBox(
            width: 150,
            height: 150,
            child: AnimatedPhysicalModel(
              duration: const Duration(seconds: 2),
              curve: Curves.fastOutSlowIn,
              // 阴影色
              shadowColor: flag ? Colors.orange : Colors.purple,
              // 影深
              elevation: flag ? 10 : 5,
              // 圆角
              borderRadius: BorderRadius.all(
                Radius.circular(flag ? 10 : 75),
              ),
              // 裁剪但不应用抗锯齿
              clipBehavior: Clip.hardEdge,
              // 形状
              shape: BoxShape.rectangle,
              color: Colors.deepPurpleAccent,
              onEnd: () => print('End'),
              child: Image.asset(
                'images/flutter.webp',
                fit: BoxFit.cover,
              ),
            ),
          ),
          const Text(
            '主题切换动画',
            style: titleStyle,
          ),
          Container(
            margin: const EdgeInsets.symmetric(
              vertical: 10.0,
            ),
            child: const Text(
              '主题变化时具有动画效果的组件，本质是Theme组件和动面结合的产物。可指定 ThemeData、动面时长、曲线、结束回调等。相当于增强版的Theme组件。',
```

```
                style: descStyle,
              ),
            ),
            Switch(
              value: theme == endTheme,
              onChanged: (v) {
                setState(() {
                  theme = v ? endTheme : startTheme;
                });
              },
            ),
            AnimatedTheme(
              data: theme,
              duration: const Duration(seconds: 1),
              curve: Curves.easeInOut,
              onEnd: () => print('End'),
              child: const ChildContent(),
            ),
            const Text(
              '渐变动画构造器',
              style: titleStyle,
            ),
            Container(
              margin: const EdgeInsets.symmetric(
                vertical: 10.0,
              ),
              child: const Text(
                '通过渐变器 Tween 对相关属性进行渐变动画，通过 builder 进行局部构建，减少刷新范围。不需要自定义动画器，可指定动画时长、曲线、结束回调。',
                style: descStyle,
              ),
            ),
            SizedBox(
              width: 200,
              height: 100,
              child: TweenAnimationBuilder(
                tween: ColorTween(begin: Colors.blue, end: _value),
                duration: const Duration(milliseconds: 800),
                builder: (BuildContext context, Color? color, Widget?
child) {
                  return GestureDetector(
                    onTap: () {
                      setState(() {
```

```
                        _value =
                            _value == Colors.red ? Colors.blue :
Colors.red;
                      });
                    },
                    child: Container(
                      width: 40,
                      height: 40,
                      decoration: BoxDecoration(
                        color: color,
                        borderRadius: BorderRadius.circular(5),
                      ),
                      child: child,
                    ),
                  );
                },
                child: const Icon(
                  Icons.android_outlined,
                  color: Colors.white,
                  size: 60,
                ),
              ),
            ),
          ],
        ),
      ),
    );
  }
}

class ChildContent extends StatelessWidget {
  const ChildContent({Key? key}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    return Container(
      width: 200,
      height: 80,
      alignment: Alignment.center,
      decoration: BoxDecoration(
        borderRadius: const BorderRadius.all(Radius.circular(5)),
        color: Theme.of(context).primaryColor,
```

```
      ),
      padding: const EdgeInsets.all(10),
      child: Text(
        'Flutter Unit',
        style: Theme.of(context).textTheme.headline1,
      ),
    );
  }
}
```