各种变换组件

旋转变换 RotationTransition

可容纳一个子组件，并使其进行旋转动画，需要提供动画器 turns ，拥有 alignment 属性。

缩放变换 ScaleTransition

可容纳一个子组件，并使其进行缩放动画，需要提供动画器 scale ，拥有 alignment 属性。

尺寸变换 SizeTransition

可容纳一个子组件，并使其进行尺寸动画，需要提供动画器 sizeFactor ，可指定尺寸变化轴及轴向的 axisAlignment。

位置变换 PositionedTransition

只能用于 Stack 中，可容纳一个子组件，让其在两个矩形间进行位置动画，需要提供动画器 rect 。

对齐变换 AlignTransition

AnimatedWidget的子类，使用 Alignment 类型的动画器让子组件在两个 Alignment 对象之间进行过渡动画。

滑动变换 SlideTransition

AnimatedWidget的子类，使用 Offset 类型的动画器让子组件在两个 Offset 对象之间进行过渡动画。

装饰变换 DecoratedBoxTransition

AnimatedWidget的子类，使用 Decorated 类型的动画器让子组件在两个 Decorated 对象之间进行过渡动画。

文本样式变换 DefaultTextStyleTransition

AnimatedWidget的子类，使用 TextStyle 类型的动画器让子组件在两个 TextStyle 对象之间进行过渡动画。

矩形位置变换 RelativePositionedTransition

AnimatedWidget的子类，使用 Rect 类型的动画器让子组件在两个 Rect 对象之间进行过渡动画。

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

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

  @override
  State<TransitionEffectWidget> createState() =>
_TransitionEffectWidgetState();
}

class _TransitionEffectWidgetState extends State<TransitionEffectWidget>
    with TickerProviderStateMixin {
  // 旋转变化参数
  late AnimationController _rotationCtrl;

  // 缩放变换参数
  late AnimationController _scaleCtrl;

  // 尺寸变换参数
```

```dart
  late AnimationController _sizeCtrl;

  // 位置变换参数
  late AnimationController _positionedCtrl;

  // 对齐变换参数
  late AnimationController _alignCtrl;

  // 滑动变换参数
  late AnimationController _slideCtrl;

  // 装饰变换参数
  late AnimationController _decoratedCtrl;

  // 文本样式变换参数
  late AnimationController _textCtrl;

  // 矩形位置变换参数
  late AnimationController _relativePosCtrl;

  @override
  void initState() {
    _rotationCtrl =
        AnimationController(vsync: this, duration: const Duration(seconds:
1));
    _rotationCtrl.forward();

    _scaleCtrl =
        AnimationController(vsync: this, duration: const Duration(seconds:
1));
    _scaleCtrl.forward();

    _sizeCtrl =
        AnimationController(vsync: this, duration: const Duration(seconds:
1));
    _sizeCtrl.forward();

    _positionedCtrl =
        AnimationController(vsync: this, duration: const Duration(seconds:
1));
    _positionedCtrl.forward();

    _alignCtrl =
```

```
        AnimationController(vsync: this, duration: const Duration(seconds:
1));
    _alignCtrl.forward();

    _slideCtrl =
        AnimationController(vsync: this, duration: const Duration(seconds:
1));
    _slideCtrl.forward();

    _decoratedCtrl =
        AnimationController(vsync: this, duration: const Duration(seconds:
1));
    _decoratedCtrl.forward();

    _textCtrl =
        AnimationController(vsync: this, duration: const Duration(seconds:
2));
    _textCtrl.forward();

    _relativePosCtrl =
        AnimationController(vsync: this, duration: const Duration(seconds:
2));
    _relativePosCtrl.forward();

    super.initState();
  }

  @override
  void dispose() {
    _rotationCtrl.dispose();
    _scaleCtrl.dispose();
    _sizeCtrl.dispose();
    _positionedCtrl.dispose();
    _alignCtrl.dispose();
    _slideCtrl.dispose();
    _decoratedCtrl.dispose();
    _textCtrl.dispose();
    _relativePosCtrl.dispose();
    super.dispose();
  }

  @override
  Widget build(BuildContext context) {
```

```
    return Scaffold(
      appBar: AppBar(
        title: const Text('各种变换组件'),
      ),
      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(
                '旋转变换 RotationTransition',
                style: titleStyle,
              ),
              Container(
                margin: const EdgeInsets.symmetric(
                  vertical: 10.0,
                ),
                child: const Text(
                  '可容纳一个子组件，并使其进行旋转动画，需要提供动画器 turns ，拥有
alignment 属性。',
                  style: descStyle,
                ),
              ),
              GestureDetector(
                onTap: () {
                  setState(() {
                    _rotationCtrl.reset();
                    _rotationCtrl.forward();
                  });
                },
                child: Container(
                  color: Colors.grey.withAlpha(22),
                  width: 100,
                  height: 100,
                  child: RotationTransition(
                    // 采用线性动画
                    turns: CurvedAnimation(
                      parent: _rotationCtrl,
                      curve: Curves.linear,
                    ),
```

```
                    // 子组件
                    child: const Icon(
                      Icons.android,
                      color: Colors.green,
                      size: 60,
                    ),
                  ),
                ),
              ),
              const Text(
                '缩放变换 ScaleTransition',
                style: titleStyle,
              ),
              Container(
                margin: const EdgeInsets.symmetric(
                  vertical: 10.0,
                ),
                child: const Text(
                  '可容纳一个子组件，并使其进行缩放动画，需要提供动画器 scale ，拥有 alignment 属性。',
                  style: descStyle,
                ),
              ),
              GestureDetector(
                onTap: () {
                  setState(() {
                    _scaleCtrl.reset();
                    _scaleCtrl.forward();
                  });
                },
                child: Container(
                  color: Colors.grey.withAlpha(22),
                  width: 100,
                  height: 100,
                  child: ScaleTransition(
                    scale: CurvedAnimation(
                      parent: _scaleCtrl,
                      curve: Curves.linear,
                    ),
                    child: const Icon(
                      Icons.favorite,
                      color: Colors.red,
                      size: 60,
```

```
            ),
          ),
        ),
      ),
      const Text(
        '尺寸变换 SizeTransition',
        style: titleStyle,
      ),
      Container(
        margin: const EdgeInsets.symmetric(
          vertical: 10.0,
        ),
        child: const Text(
          '可容纳一个子组件，并使其进行尺寸动画，需要提供动画器 sizeFactor，可指定尺寸变化轴及轴向的 axisAlignment。',
          style: descStyle,
        ),
      ),
      GestureDetector(
        onTap: () {
          setState(() {
            _sizeCtrl.reset();
            _sizeCtrl.forward();
          });
        },
        child: Wrap(
          runSpacing: 20,
          children: <Widget>[
            SizeTransition(
              // 水平方向变换
              axis: Axis.horizontal,
              sizeFactor: CurvedAnimation(
                parent: _sizeCtrl,
                curve: Curves.linear,
              ),
              child: Container(
                width: MediaQuery.of(context).size.width,
                color: Colors.orange,
                child: const Icon(
                  Icons.android,
                  color: Colors.green,
                  size: 80,
                ),
```

```
                ),
              ),
              SizeTransition(
                // 垂直方向变换
                axis: Axis.vertical,
                sizeFactor: CurvedAnimation(
                  parent: _sizeCtrl,
                  curve: Curves.linear,
                ),
                child: Container(
                  width: MediaQuery.of(context).size.width,
                  color: Colors.orange,
                  child: const Icon(
                    Icons.android,
                    color: Colors.green,
                    size: 80,
                  ),
                ),
              ),
            ],
          ),
        ),
        const Text(
          '位置变换 PositionedTransition',
          style: titleStyle,
        ),
        Container(
          margin: const EdgeInsets.symmetric(
            vertical: 10.0,
          ),
          child: const Text(
            '只能用于 Stack 中，可容纳一个子组件，让其在两个矩形间进行位置动画，需要提供动画器 rect 。',
            style: descStyle,
          ),
        ),
        GestureDetector(
          onTap: () {
            setState(() {
              _positionedCtrl.reset();
              _positionedCtrl.forward();
            });
          },
```

```
                child: Container(
                  color: Colors.grey.withAlpha(33),
                  width: 200,
                  height: 100,
                  child: Stack(
                    children: <Widget>[
                      PositionedTransition(
                        rect: RelativeRectTween(
                          begin: const RelativeRect.fromLTRB(0, 50, 150,
100),
                          end: const RelativeRect.fromLTRB(60, 0, 150,
-50),
                        ).animate(_positionedCtrl),
                        child: const Icon(
                          Icons.android,
                          color: Colors.green,
                          size: 60,
                        ),
                      ),
                    ],
                  ),
                ),
              ),
              const Text(
                '对齐变换 AlignTransition',
                style: titleStyle,
              ),
              Container(
                margin: const EdgeInsets.symmetric(
                  vertical: 10.0,
                ),
                child: const Text(
                  'AnimatedWidget的子类，使用 Alignment 类型的动画器让子组件在两个
Alignment 对象之间进行过渡动画。',
                  style: descStyle,
                ),
              ),
              GestureDetector(
                onTap: () {
                  setState(() {
                    _alignCtrl.reset();
                    _alignCtrl.forward();
                  });
```

```
                },
                child: Container(
                  width: MediaQuery.of(context).size.width,
                  color: Colors.grey.withAlpha(33),
                  height: 100,
                  child: AlignTransition(
                    // 开始左上，结束右下
                    alignment: AlignmentTween(
                      begin: Alignment.topLeft,
                      end: Alignment.bottomRight,
                    ).animate(_alignCtrl),
                    child: const Icon(
                      Icons.android,
                      color: Colors.green,
                      size: 60,
                    ),
                  ),
                ),
              ),
              const Text(
                '滑动变换 SlideTransition',
                style: titleStyle,
              ),
              Container(
                margin: const EdgeInsets.symmetric(
                  vertical: 10.0,
                ),
                child: const Text(
                  'AnimatedWidget的子类，使用 Offset 类型的动画器让子组件在两个
Offset 对象之间进行过渡动画。',
                  style: descStyle,
                ),
              ),
              GestureDetector(
                onTap: () {
                  setState(
                    () {
                      _slideCtrl.reset();
                      _slideCtrl.forward();
                    },
                  );
                },
                child: Container(
```

```
                width: MediaQuery.of(context).size.width,
                color: Colors.grey.withAlpha(33),
                height: 100,
                alignment: Alignment.topLeft,
                child: SlideTransition(
                  // x轴方向
                  textDirection: TextDirection.ltr,
                  position: Tween<Offset>(
                    begin: Offset.zero,
                    end: const Offset(3.3, 0.5),
                  ).animate(_slideCtrl),
                  child: const Icon(
                    Icons.android,
                    color: Colors.green,
                    size: 60,
                  ),
                ),
              ),
            ),
            const Text(
              '装饰变换 DecoratedBoxTransition',
              style: titleStyle,
            ),
            Container(
              margin: const EdgeInsets.symmetric(
                vertical: 10.0,
              ),
              child: const Text(
                'AnimatedWidget的子类，使用 Decorated 类型的动画器让子组件在两个 Decorated 对象之间进行过渡动画。',
                style: descStyle,
              ),
            ),
            GestureDetector(
              onTap: () {
                setState(() {
                  _decoratedCtrl.reset();
                  _decoratedCtrl.forward();
                });
              },
              child: SizedBox(
                width: 200,
                height: 100,
```

```
      child: DecoratedBoxTransition(
        position: DecorationPosition.background,
        decoration: DecorationTween(
          begin: const BoxDecoration(
              color: Colors.greenAccent,
              borderRadius: BorderRadius.all(
                Radius.circular(50),
              ),
              boxShadow: [
                BoxShadow(
                    offset: Offset(1, 1),
                    color: Colors.purple,
                    blurRadius: 3,
                    spreadRadius: 1)
              ]),
          end: const BoxDecoration(
              color: Colors.orange,
              borderRadius: BorderRadius.all(
                Radius.circular(10),
              ),
              boxShadow: [
                BoxShadow(
                    offset: Offset(1, 1),
                    color: Colors.blue,
                    blurRadius: 1,
                    spreadRadius: 0)
              ]),
        ).animate(_decoratedCtrl),
        child: const Icon(
          Icons.android,
          color: Colors.white,
          size: 60,
        ),
      ),
    ),
  ),
),
const Text(
  '文本样式变换 DefaultTextStyleTransition',
  style: titleStyle,
),
Container(
  margin: const EdgeInsets.symmetric(
    vertical: 10.0,
```

```
            ),
            child: const Text(
              'AnimatedWidget的子类，使用 TextStyle 类型的动画器让子组件在两个
TextStyle 对象之间进行过渡动画。',
              style: descStyle,
            ),
          ),
          GestureDetector(
            onTap: () {
              setState(() {
                _textCtrl.reset();
                _textCtrl.forward();
              });
            },
            child: Container(
              alignment: Alignment.center,
              width: 260,
              height: 100,
              color: Colors.grey.withAlpha(55),
              child: DefaultTextStyleTransition(
                textAlign: TextAlign.start,
                softWrap: true,
                maxLines: 1,
                overflow: TextOverflow.ellipsis,
                style: TextStyleTween(
                  begin: const TextStyle(
                      color: Colors.blue,
                      fontSize: 50,
                      shadows: [
                        Shadow(
                          offset: Offset(1, 1),
                          color: Colors.black,
                          blurRadius: 3,
                        ),
                      ]),
                  end: const TextStyle(
                      color: Colors.white,
                      fontSize: 20,
                      shadows: [
                        Shadow(
                          offset: Offset(1, 1),
                          color: Colors.purple,
                          blurRadius: 3,
```

```
                    ),
                  ]),
              ).animate(_textCtrl),
              child: const Text('走进flutter'),
            ),
          ),
        ),
        const Text(
          '矩形位置变换 RelativePositionedTransition',
          style: titleStyle,
        ),
        Container(
          margin: const EdgeInsets.symmetric(
            vertical: 10.0,
          ),
          child: const Text(
            'AnimatedWidget的子类，使用 Rect 类型的动画器让子组件在两个 Rect 对象之间进行过渡动画。',
            style: descStyle,
          ),
        ),
        GestureDetector(
          onTap: () {
            setState(() {
              _relativePosCtrl.reset();
              _relativePosCtrl.forward();
            });
          },
          child: Container(
            color: Colors.grey.withAlpha(33),
            width: 200,
            height: 100,
            child: Stack(
              children: <Widget>[
                RelativePositionedTransition(
                  size: const Size(200, 100),
                  rect: RectTween(
                    begin: const Rect.fromLTRB(0, 0, 50, 50),
                    end: const Rect.fromLTRB(0, 0, 50, 50)
                        .translate(100, 50),
                  ).animate(_relativePosCtrl),
                  child: const Icon(
                    Icons.android,
```

```
                        color: Colors.green,
                        size: 50,
                      ),
                    ),
                  ],
                ),
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```