Flutter第7天--字体图标+综合小案例+Android代码交互

转载

weixin_33713707 于 2018-12-22 08:46:16 发布 135 收藏
文章标签： 移动开发 java javascript ViewUI
原文链接：https://juejin.im/post/5c1df995e51d451611220186
版权

Flutter七日游第七天：2018-12-22 天气：雨-阴

零、前言

> 最后一天了,今天将把前六天的零碎知识整合起来,以及未涉及的零碎知识
> 最后会附上源码，在github,我按天分包稍微整理了一下，顺手Star一下吧
> 顺便提一下：Dart语法的相关测试在`test`包的`base`里(怕你们找不到)
> 与Android代码交互后感觉flutter还是蛮可以的，可惜没条件玩ios,不然岂不是可以通杀
> (给我七天或许可以把ios跑一圈,以后有钱再说吧)

留图镇楼：分类效果和查询效果

-	-

一、字体图标的相关问题

1.字体图标：

> 字体图标放大不变形，又能改变颜色，主要根据`.ttf`的字体，
> 然后图标算一个文字，根据unicode来对应图标,就可以用了。
> `Icon(Icons.android)`也许你经常用，但内置图标有限，只能测试玩玩
> 实际上用还是需要自定义才行，前端的时候有字体图标，Flutter应该也行

```
//比如我们经常怎样用：
Icon(Icons.comment)
复制代码
```

2.进入源码看看：

> 貌似都是静态常量，核心在unicode，如`0xe577`,还有就是字体(`MaterialIcons`)

```
///  * [design.google.com/icons](https://design.google.com/icons/)
class Icons {
  Icons._();

  // Generated code: do not hand-edit.
  // See https://github.com/flutter/flutter/wiki/Updating-Material-Design-Fonts
  // BEGIN GENERATED

  /// <i class="material-icons md-36">360</i> &#x2014; material icon named "360".
  static const IconData threesixty = IconData(0xe577, fontFamily: 'MaterialIcons');

  /// <i class="material-icons md-36">3d_rotation</i> &#x2014; material icon named "3d rotation".
  static const IconData threed_rotation = IconData(0xe84d, fontFamily: 'MaterialIcons');

  /// <i class="material-icons md-36">4k</i> &#x2014; material icon named "4k".
  static const IconData four_k = IconData(0xe072, fontFamily: 'MaterialIcons');
复制代码
```

3.怎么才能自定义字体图标

> 玩前端的应该都知道:还是进入阿里图标神库:iconfont

4.根据Flutter内置的类，我写了一个自动代码生成器

> 虽然直接也能用，不够要记住图标的unicode码，算了，还是跟Flutter看齐吧
> 注意：为了简单使用：拷贝到的位置，命名，请务必和下面保持一致!保持一致!
> 把两个文件拷贝到对应处，`icon_by_toly.dart`写好(在下面)，右键运行就自动生成`iconfont.dart`了

> 代码生成器：`icon_by_toly.dart`

```
import 'dart:io';

main() {
  var result = """import 'package:flutter/widgets.dart';
//Power By 张风捷特烈---

class TolyIcon {

    TolyIcon._();
""";
  var file = File.fromUri(Uri.parse("${Uri.base}iconfont./iconfont.css"));
  var read = file.readAsStringSync();

  var split = read.split(".icon-");
  split.forEach((str) {
    if (str.contains("before")) {
      var split = str.split(":");
      result += "static const IconData " +
          split[0].replaceAll("-", "_") +
          " = const IconData(" +
          split[2].replaceAll("\"\\", "0x").split("\"")[0] +
          ", fontFamily: \"TolyIcon\");\n";
    }
  });
  result+="}";
  var fileOut = File.fromUri(Uri.parse("${Uri.base}lib./iconfont.dart"));
  fileOut.writeAsStringSync(result);
}
```
复制代码

使用：将下面拷贝到`pubspec.yaml`的flutter标签下：

```
  fonts:
    - family: TolyIcon
      fonts:
        - asset: iconfont/iconfont.ttf
```
复制代码

```
Icon(TolyIcon.icon_spring_boot)//颜色可自行处理
```
复制代码

> 友情提示：下载之前最好把图标名字改一下，不然之后找起来费劲
> 如果实在不想该，可以点击这里查看名字和图标的对应情况

二、综合小案例

1.初始代码：主页面：`android_stack.dart`

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

class AndroidPage extends StatefulWidget {
  @override
  _AndroidPageState createState() => _AndroidPageState();
}

class _AndroidPageState extends State<AndroidPage>
    with SingleTickerProviderStateMixin {


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

  @override
  Widget build(BuildContext context) {

    var scaffold = Scaffold(
      appBar: AppBar(
        title: Text("张风捷特烈"),
      ),
      body: Container(),
      floatingActionButton: FloatingActionButton(
        onPressed: () {

        },
        tooltip: 'Increment',
        child: Icon(Icons.add),
      ),
    );

    return scaffold;
  }
}
复制代码
```

2.拼接底部条

2.1:常量的准备：(为了方便使用或修改)

```
class ItemBean {
  Color color;
  IconData iconId;
  String info;
  ItemBean(this.color, this.iconId, this.info);
}

//底部栏图标信息
var iconLi=[
  ItemBean(Color(0xff8FC552),TolyIcon.android,"Android"),
  ItemBean(Color(0xff6BFB00),TolyIcon.icon_spring_boot,"SpringBoot"),
  ItemBean(Color(0xff63DAFF),TolyIcon.react,"React"),
  ItemBean(Color(0xffF3D861),TolyIcon.biji,"编程随笔"),
  ItemBean(Color(0xff5CEBF2),TolyIcon.daima,"系列文章")
];
```
复制代码

2.2:底部栏：

```
//成员变量
int _curIndex = 0;

//底部栏
var bottomNavigationBar = BottomNavigationBar(
  items: iconLi.map((item) {
    return BottomNavigationBarItem(
        title: Text(
          item.info,
          style: TextStyle(fontSize: 12, color: Colors.black),
        ),
        icon: Icon(
          item.iconId,
          color: item.color,
        ),
        backgroundColor: Color(0xffffffff));
  }).toList(),
  currentIndex: _curIndex,
  onTap: _onTapBNB,
);
```
复制代码

2.3：底部栏点击监听：`_onTapBNB`

```
//底部栏点击监听
  void _onTapBNB(int position) {
    _curIndex = position;
    setState(() {});
  }
```
复制代码

3:页面条目：

> 第五天写了几个条目，现在拿来用(详细分析见第五天，这里不废话了)

静态填充	左侧滑栏

3.1：左侧滑栏:`left_draw.dart`

```
class LeftDrawPage extends StatefulWidget {
  @override
  _LeftDrawPageState createState() => _LeftDrawPageState();
}

class _LeftDrawPageState extends State<LeftDrawPage>
    with SingleTickerProviderStateMixin {
  @override
  Widget build(BuildContext context) {
//左边头像
    var headImg3 = Image.asset(
      "images/icon_90.png",
      width: 50,
      height: 50,
    );
//中间的信息
    var center3 = Column(
      mainAxisAlignment: MainAxisAlignment.center,
      crossAxisAlignment: CrossAxisAlignment.start,
      children: <Widget>[
        Text(
          "张风捷特烈",
          style: bigStyle,
        ),
        Row(
          children: <Widget>[
            Icon(Icons.next_week, size: 15),
            pd(Text("创世神 | 无"), l: 5)
          ],
        ),
        Row(
          children: <Widget>[
            Icon(Icons.keyboard, size: 15),
            pd(Text("海的彼岸有我未曾见证的风采"), l: 5)
          ],
        ),
      ],
    );
    var rowLine3 = Row(
      children: <Widget>[
        pda(headImg3, 5),
        Expanded(child: pda(center3, 5)),
      ],
    );
    var test3 = Card(
        child: Container(
            height: 95,
            color: Colors.white,
            padding: EdgeInsets.all(5),
            child: rowLine3));
    return Drawer(
        elevation: 5,
        child: Container(
            padding: EdgeInsets.only(top: 50),
            alignment: AlignmentDirectional.topCenter,
            color: Color(0xff99C6F9),
            child: test3));
  }
}
```
复制代码

3.2：列表静态填充:`home_list.dart`

```
class HomeListPage extends StatefulWidget {
  @override
  _HomeListPageState createState() => _HomeListPageState();
}

class _HomeListPageState extends State<HomeListPage> {
  @override
  Widget build(BuildContext context) {
    return ListView.builder(
      itemCount: 10,
      itemBuilder: (BuildContext context, int index) {
        return renderItem(index);
      },
    );
  }

  renderItem(int index) {
    -----------------测试4--------------------------------
    var line1_4 = Row(
      children: <Widget>[
        Image.asset("images/icon_90.png", width: 20, height: 20),
        Expanded(
          child: pd(Text("张风捷特烈"), l: 5),
        ),
        Text(
          "Flutter/Dart",
          style: infoStyle,
        )
      ],
    );

    var center_right = Column(
      mainAxisSize: MainAxisSize.min,
      children: <Widget>[
        Text(
          "Flutter第4天--基础控件(下)+Flex布局详解",
          style: littelStyle,
          maxLines: 2,
        ),
        pd(
            Text(
              "1.2：优雅地查看：图片的适应模式--BoxFit1.3：优雅地查看：颜色混合模式--colorBlendMode",
              [illegible]
              style: infoStyle,
              maxLines: 2,
              overflow: TextOverflow.ellipsis,
            ),
            t: 5),
      ],
    );

//中间的信息
    var center4 = Row(
      children: <Widget>[
        Expanded(child: pda(center_right, 5)),
        Image.asset(
          "images/wy_300x200.jpg",
          width: 80,
          height: 80,
          fit: BoxFit.fitHeight,
        )
      ],
    );

    var end4 = Row(
      children: <Widget>[
        Icon(
          Icons.grade,
          color: Colors.green,
          size: 20,
        ),
        Text(
          "1000W",
          style: infoStyle,
        ),
        pd(Icon(Icons.tag_faces, color: Colors.lightBlueAccent, size: 20),
            l: 15, r: 5),
        Text("2000W", style: infoStyle),
      ],
    );

    var item4 = Column(
      children: <Widget>[line1_4, Expanded(child: center4), end4],
    );

    var aCard = Card(
        child: Container(
            height: 160,
            color: Colors.white,
            padding: EdgeInsets.all(10),
            child: item4));

    return aCard;
  }
}
```
复制代码

4.动态数据获取：

昨天已经把http获取数据的内容将过了，并且把服务端的数据解析了
今天就是使用这些数据，来填充静态界面，api接口介绍和NoteBean昨天已完成
封装一个获取数据的方法：简单说下用法:
style是类型：Android是A ;SpringBoot是SB ; React 是Re ; 笔记是 Note
offset和num 联合使用可以达到分页效果， 比如offset=24,num=12,就是一页12条数据的第3页

```
import 'dart:convert';

import 'package:http/http.dart' as client;
import 'package:toly/pager/day7/bean.dart';

const BASE_URL = 'http://192.168.43.60:8089';//api接口的域名自己改一下
const API = '/api/android/note/';

Future<List<NoteBean>> get({style = '', offset = 0, num = 1}) async {
  var dataLi = <NoteBean>[];
  var url = BASE_URL + API + style + "/" + "$offset" + "/" + "$num";
  try {
    final response = await client.get(url);
    if (response.statusCode == 200) {
      var result = ResultBean.fromJson(json.decode(response.body));
      List data = result.data;
      print(NoteBean.fromJson(data[0]).type);
      for (int i = 0; i < data.length; i++) {
        dataLi.add(NoteBean.fromJson(data[i]));
      }
      return dataLi;
    }
  } catch (e) {
    print(e);
  }
}
```
复制代码

5.用一个数据来进行填充测试：

主页面：android_stack.dart,initState的时候获取数据，并更新状态

```
//定义一个成员变量
  List<NoteBean> _notes = [];

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

  get(num: 1).then((beanLi) {
    _notes = beanLi;
    setState(() {});
  });
}
```
复制代码

> 列表界面：`home_list.dart`:接收主界面传来的_notes，并渲染数据

```
class HomeListPage extends StatefulWidget {
  List<NoteBean> _notes;
  HomeListPage(List<NoteBean> notes) {
    _notes = notes;
  }
  @override
  _HomeListPageState createState() => _HomeListPageState();
}

class _HomeListPageState extends State<HomeListPage> {
  @override
  Widget build(BuildContext context) {
    var notes = widget._notes;

    return ListView.builder(
      itemCount: notes.length,
      itemBuilder: (BuildContext context, int index) {
        return renderItem(notes[index]);
      },
    );
  }
    //渲染条目
  renderItem(NoteBean note) {
    var line1_4 = Row(
      children: <Widget>[
        Image.asset("images/icon_90.png", width: 20, height: 20),
        Expanded( child: pd(Text("张风捷特烈"), l: 5),),
        Text( note.type,style: infoStyle,)
        ],
    );
    var center_right = Column(
      mainAxisSize: MainAxisSize.min,
      children: <Widget>[Text(note.name,style: littelStyle,maxLines: 2,),
        pd(Text( note.info, style: infoStyle, maxLines: 2,
              overflow: TextOverflow.ellipsis, ), t: 5),
      ],
    );

//中间的信息
```

```
    var center4 = Row(
      children: <Widget>[
        Expanded(child: pda(center_right, 5)),
        Image.network( note.imgUrl,
          width: 80, height: 80, fit: BoxFit.fitHeight )
      ],
    );

    var end4 = Row(
      children: <Widget>[
        Icon( Icons.grade, color: Colors.green, size: 20, ),
        Text( "1000W", style: infoStyle,),
        pd(Icon(Icons.tag_faces, color: Colors.lightBlueAccent, size: 20),
            l: 15, r: 5),
        Text("2000W", style: infoStyle),
      ],
    );

    var item4 = Column(
      children: <Widget>[line1_4, Expanded(child: center4), end4],
    );

    var aCard = Card(
        child: Container(  height: 160,color: Colors.white,
            padding: EdgeInsets.all(10), child: item4));
    return aCard;
  }
}
```
复制代码

> 现在万事俱备，东风也到了,`num`小小动一下：`num=30`

--	--

> 也许你感觉还未开始，但确实已经结束了...

6.底部导航栏的切换：(下面两个图一样的，为了撑场面...)

> 刚才是数据没有分类型，现在点击底部导航，按范围进行展示
> `get(style: "area/A", num: 30)//`这样就是展示又有安卓类的文章

-	-

> android_stack.dart添加成员变量
> 这里我默认加载完，做分页的话，再添加个_count的成员变量就行了

```
String style = "area/A";

//页面打开，默认加载安卓页
  @override
  void initState() {
    super.initState();
    get(style: style, num: 1000).then((beanLi) {
      _notes = beanLi;
      setState(() {});
    });
  }

  //底部栏点击监听---动态改变范围
  void _onTapBNB(int position) {
    switch (position) {
      case 0:
        style = "area/A";
        break;
      case 1:
        style = "area/SB";
        break;
      case 2:
        style = "area/Re";
        break;
      case 3:
        style = "area/Note";
        break;
      case 4:
        style = "area/A";
        break;
    }
    _curIndex = position;
    get(style: style, num: 1000).then((beanLi) {
      _notes = beanLi;
      setState(() {});
    });
  }
复制代码
```

7.底部栏和搜索功能

> 底部栏用法详情在第四篇

--	--

> 底部栏:这里把事件写在里面了，你也可以抽成方法
> 或者有些控件太长，你也可以抽出来做变量

```
var searchSheet = BottomSheet(
    onClosing: () {},
    builder: (context) => (Card(
        color: Color.fromARGB(255, 214, 242, 251),
        child: Wrap(
          children: <Widget>[
            Center(child: pdhv(TextField(
              onChanged: (v) {style = "name/" + v;}), h: 60)),
            Center(child: pdhv( GestureDetector(child:
            Image.asset("images/icon_90.png",width: 50,height: 50 ),
                      onTap: () {
                        get(style: style, num: 1000).then((beanLi) {
                          _notes = beanLi;
                          setState(() {});
                        });
                      },
                    ),
                    v: 10)),
          ],
        ))));

//点击按钮弹出：
 var scContext; //先声明一下Scaffold的context
    var scaffold = Scaffold(
        appBar: AppBar(
          title: Text("张风捷特烈"),
        ),
        body: Builder(builder: (context) {
          scContext = context;
          return HomeListPage(_notes);
        }),
        floatingActionButton: FloatingActionButton(
          onPressed: () {
            Scaffold.of(scContext).showBottomSheet(searchSheet.builder);
          },
          //下面不用修改，略...
```

复制代码

> Ok,小案例就这样

三、Android代码交互

1.最简单的无参无返回函数调用：两对应

> 不得不说：前六天不能弹吐司真是不好受，原生交互肯定先拿他开刀

1.1：Android代码

```
public class MainActivity extends FlutterActivity {
    private static final String CHANNEL = "www.toly1994.com/test.名字随意起";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        GeneratedPluginRegistrant.registerWith(this);

        new MethodChannel(getFlutterView(), CHANNEL).setMethodCallHandler(
                new MethodChannel.MethodCallHandler() {
                    @Override
                    public void onMethodCall(MethodCall methodCall, MethodChannel.Result result) {
                        if (methodCall.method.equals("showToast")) {
                            showToast("Hello Flutter,I am in Android");
                        } else {
                            result.notImplemented();
                        }
                    }
                }
        );
    }
    /**
     * 显示吐司
     *
     * @param msg 信息
     */
    public void showToast(String msg) {
        Toast toast = Toast.makeText(this, msg, Toast.LENGTH_SHORT);
        toast.show();
    }
}
```
复制代码

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

const platform = const MethodChannel("www.toly1994.com/test.名字随意起");

var toastTest = Center(
  child: RaisedButton(
    onPressed: () {
      platform.invokeMethod("showToast");
    },
    child: new Text("点击弹吐司"),
  ),
);
```

复制代码

2.Flutter中传参，调用Android含参方法:三对应

2.1：Android代码

```
public class MainActivity extends FlutterActivity {
    private static final String CHANNEL = "www.toly1994.com/test.名字随意起";

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        GeneratedPluginRegistrant.registerWith(this);

        new MethodChannel(getFlutterView(), CHANNEL).setMethodCallHandler(
                new MethodChannel.MethodCallHandler() {
                    @Override
                    public void onMethodCall(MethodCall methodCall, MethodChannel.Result result) {
                        if (methodCall.method.equals("showToast")) {
                            //解析参数
                            String msg = methodCall.argument("msg");
                            showToast(msg);
                        } else {
                            result.notImplemented();
                        }
                    }
                }
        );
    }

    /**
     * 显示吐司
     *
     * @param msg 信息
     */
    public void showToast(String msg) {
        Toast toast = Toast.makeText(this, msg, Toast.LENGTH_SHORT);
        toast.show();
    }
}
复制代码
```

2.2:Flutter代码：

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

const platform = const MethodChannel("www.toly1994.com/test.名字随意起");

var toastTest = Center(
  child: RaisedButton(
    onPressed: () {
      platform.invokeMethod("showToast",{"msg":"Flutter大爷我赏你一口吐司"});
    },
    child: new Text("点击弹吐司"),
  ),
);
```
复制代码

2.3：加返回值的方法调用：

> 举什么例子，我想了一会，就来个MD5码吧

```
//Activity添加判断分支
if (methodCall.method.equals("getMD5")) {
    String arg = methodCall.argument("arg");
    String md5 = getMD5(arg);
    result.success(md5);
}

    /**
     * 获取一个字符串的Md5值
     *
     * @param content 内容
     * @return Md5值
     */
    public String getMD5(String content) {
        content = content + "芝麻开门";
        try {
            MessageDigest digest = MessageDigest.getInstance("MD5");
            digest.update(content.getBytes());
            return getHashString(digest);
        } catch (NoSuchAlgorithmException e) {
            e.printStackTrace();
        }
        return null;
    }

    private static String getHashString(MessageDigest digest) {
        StringBuilder builder = new StringBuilder();
        for (byte b : digest.digest()) {
            builder.append(Integer.toHexString((b >> 4) & 0xf));
            builder.append(Integer.toHexString(b & 0xf));
        }
        return builder.toString();
    }
```

复制代码

2.2:Flutter代码：

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

const platform = const MethodChannel("www.toly1994.com/test.名字随意起");

var toastTest = Center(
  child: RaisedButton(
    onPressed: () {

      var result= platform.invokeMethod("getMD5",{"arg":"https://www.jianshu.com/p/9bac0072d1a0"});
      result.then((str){
        platform.invokeMethod("showToast",{"msg":str});
      });

    },
    child: new Text("点击弹吐司"),
  ),
);
```
复制代码

> 基本上也就这三种情况

[彩蛋]--以前Mark的一个小点：Card的shape

> 有人说学习的时候一个问题会牵扯到很多其他的问题，问题一多就无从下手
> 我只说一个字："栈"：来最后一波学习看源码的方法了,走起

1.shape是什么:形状

> 现在的问题栈

> 可见默认圆角是4的RoundedRectangleBorder

```
---->[shape属性]----
/// The default shape is a [RoundedRectangleBorder] with a circular corner
/// radius of 4.0.
final ShapeBorder shape;

---->[RoundedRectangleBorder]----
 const RoundedRectangleBorder({
    this.side = BorderSide.none,
    this.borderRadius = BorderRadius.zero,

//可见有两个属性：BorderSide和BorderRadius

---->[BorderSide]----
class BorderSide {
  /// Creates the side of a border.
  ///
  /// By default, the border is 1.0 logical pixels wide and solid black.
  const BorderSide({
    this.color = const Color(0xFF000000),
    this.width = 1.0,
    this.style = BorderStyle.solid,

---->[BorderRadius]----
class BorderRadius extends BorderRadiusGeometry {
  /// Creates a border radius where all radii are [radius].
  const BorderRadius.all(Radius radius) : this.only(
    topLeft: radius,
    topRight: radius,
    bottomLeft: radius,
    bottomRight: radius,
  );

  /// Creates a border radius where all radii are [Radius.circular(radius)].
  BorderRadius.circular(double radius) : this.all(
    Radius.circular(radius),
  );

---->[Radius]------
class Radius {
  /// Constructs a circular radius. [x] and [y] will have the same radius value.
  const Radius.circular(double radius) : this.elliptical(radius, radius);

  /// Constructs an elliptical radius with the given radii.
  const Radius.elliptical(this.x, this.y);

  /// The radius value on the horizontal axis.
  final double x;

  /// The radius value on the vertical axis.
  final double y;
```

复制代码

> 一个shape牵扯出这么多类，有人可能就栈溢出了，还是使用默认的吧，等一下，且听我分析
> 当Radius入问题栈之后，看一下也就是两个值，就出栈了，BorderRadius跟着也出栈了
> BorderSide三个字段，看一下，出栈了，现在栈顶是`RoundedRectangleBorder`你还不会吗?

2. RoundedRectangleBorder改变圆角大小+边线

```
var card_shape = Card(
//    shape: CircleBorder(side: BorderSide(width: 1)),
    shape: RoundedRectangleBorder(
      side:BorderSide(color: Color(0xffD516F5),width: 5) ,
        borderRadius: BorderRadius.all(Radius.circular(20))),
    clipBehavior: Clip.antiAlias,
    child: Container(
      width: 100,
      height: 100,
      color: Color(0xffCDECF6),
      child: Center(child:Text(
        "捷",
        style: TextStyle(color: Colors.black,fontSize: 40),
      ) ,),
    ));
```

复制代码

> 那弹栈过后问题跑哪里?
> 我想应该是临时知识库吧,你解决的问题中获取的知识，经验会累积
> 可能长久不用知识库里的知识会漏掉，但印象有的，下一次再入栈，解决起来会更快
> 在的知识库里扎根的知识，那当你遇到时，就不是问题，直接弹栈，这样想学习是不是也挺好玩的

> 大神级的Coder知识库丰富，问题都不是问题，也许偶尔入栈一两个，但栈很深(感觉挺浪费哈)
> 新手就是栈比较浅，问题多，所以容易`StackOver`,所以修炼你容忍问题的能力(栈深)很有必要
> 像我这样不深不浅的刚刚好，会碰到问题，也能一点点解决，一点一点踏上封神之路
> 但所有的大神也都是从新手这样过来的，解决问题的能力也不是与生俱来，祝你慢慢弹栈，收获多多。

3.接下来看`ShapeBorder`在栈顶，我们去瞅瞅

> BorderSide现在已经化敌为友，CircleBorder岂不是秒出栈，并且俘获`CircleBorder`一枚
> 而且BorderSide强化+1，知识就是这样积累的，难道还有别的方法吗?除非记忆拷贝...

> 转一转当CD背景感觉挺不错

```
var card_shape = Card(
    shape: CircleBorder(side: BorderSide(width: 15,color: Color(0xffF9DFA7))),
    clipBehavior: Clip.antiAlias,
    child: Container(
      width: 100,
      height: 100,
      color: Color(0xffCDECF6),
      child: Center(child:Text(
        "捷",
        style: TextStyle(color: Colors.black,fontSize: 40),
      ) ,),
    ));
复制代码
```

4.前方高能，非战斗人员带好零食

> 其实觉得shape好玩，是在粗略看源码时,看到了canvas,才mark的
> 自定义ShapeBorder走起：画具在手，天下我有

```
var card_shape = Card(
    shape: StarBorder(),
//    shape: CircleBorder(side: BorderSide(width: 15,color: Color(0xffF9DFA7))),
//    shape: RoundedRectangleBorder(
//      side:BorderSide(color: Color(0xffD516F5),width: 5) ,
//        borderRadius: BorderRadius.all(Radius.circular(20))),
    clipBehavior: Clip.hardEdge,
    child: Container(
      width: 100,
      height: 100,
      color: Color(0xffCDECF6),
      child: Center(
        child: Text(
          "捷",
          style: TextStyle(color: Colors.black, fontSize: 40),
        ),
      ),
    ));

class StarBorder extends ShapeBorder {
  @override
  // TODO: implement dimensions
  EdgeInsetsGeometry get dimensions => null;

  @override
  Path getInnerPath(Rect rect, {TextDirection textDirection}) {
    // TODO: implement getInnerPath
    return null;
  }

  @override
  Path getOuterPath(Rect rect, {TextDirection textDirection}) {
    print(rect.right);
    return regularPolygonPath(10, 50,x: rect.height/2,y: rect.width/2);
  }

  @override
  void paint(Canvas canvas, Rect rect, {TextDirection textDirection}) {
    canvas.translate(50, 50);
//    canvas.drawPath(nStarPath(5, 40, 20), new Paint());
  }

  @override
  ShapeBorder scale(double t) {
    // TODO: implement scale
    return null;
  }
}
复制代码
```

路径封装(稍微优化了一下)

```
/**
 * n角星路径
 *
 * @param num 几角星
 * @param R   外接圆半径
 * @param r   内接圆半径
 * @return n角星路径
 */
Path nStarPath(int num, double R, double r, {x = 0, y = 0}) {
  Path path = new Path();
  double perDeg = 360 / num; //尖角的度数
  double degA = perDeg / 2 / 2;
  double degB = 360 / (num - 1) / 2 - degA / 2 + degA;

  path.moveTo(cos(_rad(degA)) * R+x, (-sin(_rad(degA)) * R+y));
  for (int i = 0; i < num; i++) {
    path.lineTo(
        cos(_rad(degA + perDeg * i)) * R+x, -sin(_rad(degA + perDeg * i)) * R+y);
    path.lineTo(
        cos(_rad(degB + perDeg * i)) * r+x, -sin(_rad(degB + perDeg * i)) * r+y);
  }
  path.close();
  return path;
}

/**
 * 画正n角星的路径:
 *
 * @param num 角数
 * @param R   外接圆半径
 * @return 画正n角星的路径
 */
```

```
Path regularStarPath(int num, double R,{x = 0, y = 0}) {
  double degA, degB;
  if (num % 2 == 1) {
    //奇数和偶数角区别对待
    degA = 360 / num / 2 / 2;
    degB = 180 - degA - 360 / num / 2;
  } else {
    degA = 360 / num / 2;
    degB = 180 - degA - 360 / num / 2;
  }
  double r = R * sin(_rad(degA)) / sin(_rad(degB));
  return nStarPath(num, R, r,x: x,y:y);
}

/**
 * 画正n边形的路径
 *
 * @param num 边数
 * @param R   外接圆半径
 * @return 画正n边形的路径
 */
Path regularPolygonPath(int num, double R,{x = 0, y = 0}) {
  double r = R * cos(_rad(360 / num / 2)); //!!一点解决
  return nStarPath(num, R, r,x: x,y:y);
}

/**
 * 角度制化为弧度制
 *
 * @param deg 角度
 * @return 弧度
 */
double _rad(double deg) {
  return deg * pi / 180;
}
```
复制代码

> 师傅领进门，修行在个人，我已经把功力传给你了,能否修成正果，就看各自造化。
> 事了拂衣去，深藏功与名,Ok,Flutter七日游，完捷散花，自认为没有烂尾，耶!

后记：捷文规范

1.本文成长记录及勘误表

项目源码	日期	备注
V0.1-github	2018-12-22	Flutter第7天--字体图标+综合小案例+Android代码交互

2.更多关于我

笔名	QQ	微信	爱好
张风捷特烈	1981462002	zdl1994328	语言
我的github	我的简书	我的掘金	个人网站

我的github	我的简书	我的掘金	个人网站
笔名	**QQ**	**微信**	爱好

3.声明

> 1----本文由张风捷特烈原创,转载请注明
> 2----欢迎广大编程爱好者共同交流
> 3----个人能力有限，如有不正之处欢迎大家批评指证，必定虚心改正
> 4----看到这里，我在此感谢你的喜欢与支持