不要再重复造轮子了，这款开源工具类库贼好使！

原创 梦想de星空 macrozheng 2020-09-02 09:02

收录于合集
#mall学习教程（参考篇）
27个

> Hutool是一个小而全的Java工具类库，它帮助我们简化每一行代码，避免重复造轮子。如果你有需要用到某些工具类的时候，不妨在Hutool里面找找。本文总结了平时常用的16个工具类，希望对大家有所帮助！

安装

Hutool的安装非常简单，Maven项目中只需在 `pom.xml` 添加以下依赖即可。

```
<dependency>
    <groupId>cn.hutool</groupId>
    <artifactId>hutool-all</artifactId>
    <version>5.4.0</version>
</dependency>
```

常用工具类

> 使用一个工具方法代替一段复杂代码，避免 复制粘贴 代码，可以极大的提高我们的开发效率，下面介绍下我常用的工具方法！

Convert

类型转换工具类，用于各种类型数据的转换。平时我们转换类型经常会面临类型转换失败的问题，要写 `try catch` 代码，有了它，就不用写了！

```
//转换为字符串
int a = 1;
```

```
String aStr = Convert.toStr(a);
//转换为指定类型数组
String[] b = {"1", "2", "3", "4"};
Integer[] bArr = Convert.toIntArray(b);
//转换为日期对象
String dateStr = "2017-05-06";
Date date = Convert.toDate(dateStr);
//转换为列表
String[] strArr = {"a", "b", "c", "d"};
List<String> strList = Convert.toList(String.class, strArr);
```

DateUtil

日期时间工具类，定义了一些常用的日期时间操作方法。JDK自带的Date和Calendar对象真心不好用，有了它操作日期时间就简单多了！

```
//Date、long、Calendar之间的相互转换
//当前时间
Date date = DateUtil.date();
//Calendar转Date
date = DateUtil.date(Calendar.getInstance());
//时间戳转Date
date = DateUtil.date(System.currentTimeMillis());
//自动识别格式转换
String dateStr = "2017-03-01";
date = DateUtil.parse(dateStr);
//自定义格式化转换
date = DateUtil.parse(dateStr, "yyyy-MM-dd");
//格式化输出日期
String format = DateUtil.format(date, "yyyy-MM-dd");
//获得年的部分
int year = DateUtil.year(date);
//获得月份，从0开始计数
int month = DateUtil.month(date);
//获取某天的开始、结束时间
Date beginOfDay = DateUtil.beginOfDay(date);
Date endOfDay = DateUtil.endOfDay(date);
//计算偏移后的日期时间
Date newDate = DateUtil.offset(date, DateField.DAY_OF_MONTH, 2);
//计算日期时间之间的偏移量
long betweenDay = DateUtil.between(date, newDate, DateUnit.DAY);
```

JSONUtil

JSON解析工具类，可用于对象与JSON之间的互相转化。

```
PmsBrand brand = new PmsBrand();
brand.setId(1L);
brand.setName("小米");
brand.setShowStatus(1);
//对象转化为JSON字符串
String jsonStr = JSONUtil.parse(brand).toString();
LOGGER.info("jsonUtil parse:{}", jsonStr);
//JSON字符串转化为对象
PmsBrand brandBean = JSONUtil.toBean(jsonStr, PmsBrand.class);
LOGGER.info("jsonUtil toBean:{}", brandBean);
List<PmsBrand> brandList = new ArrayList<>();
brandList.add(brand);
String jsonListStr = JSONUtil.parse(brandList).toString();
//JSON字符串转化为列表
brandList = JSONUtil.toList(new JSONArray(jsonListStr), PmsBrand.class);
LOGGER.info("jsonUtil toList:{}", brandList);
```

StrUtil

字符串工具类，定义了一些常用的字符串操作方法。 `StrUtil` 比 `StringUtil` 名称更短，用起来也更方便！

```
//判断是否为空字符串
String str = "test";
StrUtil.isEmpty(str);
StrUtil.isNotEmpty(str);
//去除字符串的前后缀
StrUtil.removeSuffix("a.jpg", ".jpg");
StrUtil.removePrefix("a.jpg", "a.");
//格式化字符串
String template = "这只是个占位符:{}";
String str2 = StrUtil.format(template, "我是占位符");
LOGGER.info("/strUtil format:{}", str2);
```

ClassPathResource

ClassPath单一资源访问类，可以获取classPath下的文件，在Tomcat等容器下，classPath一般是WEB-INF/classes。

```
//获取定义在src/main/resources文件夹中的配置文件
ClassPathResource resource = new ClassPathResource("generator.properties");
Properties properties = new Properties();
properties.load(resource.getStream());
LOGGER.info("/classPath:{}", properties);
```

ReflectUtil

Java反射工具类，可用于反射获取类的方法及创建对象。

```
//获取某个类的所有方法
Method[] methods = ReflectUtil.getMethods(PmsBrand.class);
//获取某个类的指定方法
Method method = ReflectUtil.getMethod(PmsBrand.class, "getId");
//使用反射来创建对象
PmsBrand pmsBrand = ReflectUtil.newInstance(PmsBrand.class);
//反射执行对象的方法
ReflectUtil.invoke(pmsBrand, "setId", 1);
```

NumberUtil

数字处理工具类，可用于各种类型数字的加减乘除操作及类型判断。

```
double n1 = 1.234;
double n2 = 1.234;
double result;
//对float、double、BigDecimal做加减乘除操作
result = NumberUtil.add(n1, n2);
result = NumberUtil.sub(n1, n2);
result = NumberUtil.mul(n1, n2);
result = NumberUtil.div(n1, n2);
//保留两位小数
BigDecimal roundNum = NumberUtil.round(n1, 2);
String n3 = "1.234";
//判断是否为数字、整数、浮点数
NumberUtil.isNumber(n3);
```

```
NumberUtil.isInteger(n3);
NumberUtil.isDouble(n3);
```

BeanUtil

JavaBean工具类，可用于Map与JavaBean对象的互相转换以及对象属性的拷贝。

```
PmsBrand brand = new PmsBrand();
brand.setId(1L);
brand.setName("小米");
brand.setShowStatus(0);
//Bean转Map
Map<String, Object> map = BeanUtil.beanToMap(brand);
LOGGER.info("beanUtil bean to map:{}", map);
//Map转Bean
PmsBrand mapBrand = BeanUtil.mapToBean(map, PmsBrand.class, false);
LOGGER.info("beanUtil map to bean:{}", mapBrand);
//Bean属性拷贝
PmsBrand copyBrand = new PmsBrand();
BeanUtil.copyProperties(brand, copyBrand);
LOGGER.info("beanUtil copy properties:{}", copyBrand);
```

CollUtil

集合操作的工具类，定义了一些常用的集合操作。

```
//数组转换为列表
String[] array = new String[]{"a", "b", "c", "d", "e"};
List<String> list = CollUtil.newArrayList(array);
//join：数组转字符串时添加连接符号
String joinStr = CollUtil.join(list, ",");
LOGGER.info("collUtil join:{}", joinStr);
//将以连接符号分隔的字符串再转换为列表
List<String> splitList = StrUtil.split(joinStr, ',');
LOGGER.info("collUtil split:{}", splitList);
//创建新的Map、Set、List
HashMap<Object, Object> newMap = CollUtil.newHashMap();
HashSet<Object> newHashSet = CollUtil.newHashSet();
ArrayList<Object> newList = CollUtil.newArrayList();
```

```
//判断列表是否为空
CollUtil.isEmpty(list);
```

MapUtil

Map操作工具类，可用于创建Map对象及判断Map是否为空。

```
//将多个键值对加入到Map中
Map<Object, Object> map = MapUtil.of(new String[][]{
    {"key1", "value1"},
    {"key2", "value2"},
    {"key3", "value3"}
});
//判断Map是否为空
MapUtil.isEmpty(map);
MapUtil.isNotEmpty(map);
```

AnnotationUtil

注解工具类，可用于获取注解与注解中指定的值。

```
//获取指定类、方法、字段、构造器上的注解列表
Annotation[] annotationList = AnnotationUtil.getAnnotations(HutoolController.class, false);
LOGGER.info("annotationUtil annotations:{}", annotationList);
//获取指定类型注解
Api api = AnnotationUtil.getAnnotation(HutoolController.class, Api.class);
LOGGER.info("annotationUtil api value:{}", api.description());
//获取指定类型注解的值
Object annotationValue = AnnotationUtil.getAnnotationValue(HutoolController.class, RequestMapping
```

SecureUtil

加密解密工具类，可用于MD5加密。

```
//MD5加密
String str = "123456";
```

```
String md5Str = SecureUtil.md5(str);
LOGGER.info("secureUtil md5:{}", md5Str);
```

CaptchaUtil

验证码工具类，可用于生成图形验证码。

```
//生成验证码图片
LineCaptcha lineCaptcha = CaptchaUtil.createLineCaptcha(200, 100);
try {
    request.getSession().setAttribute("CAPTCHA_KEY", lineCaptcha.getCode());
    response.setContentType("image/png");//告诉浏览器输出内容为图片
    response.setHeader("Pragma", "No-cache");//禁止浏览器缓存
    response.setHeader("Cache-Control", "no-cache");
    response.setDateHeader("Expire", 0);
    lineCaptcha.write(response.getOutputStream());
} catch (IOException e) {
    e.printStackTrace();
}
```

Validator

字段验证器，可以对不同格式的字符串进行验证，比如邮箱、手机号、IP等格式。

```
//判断是否为邮箱地址
boolean result = Validator.isEmail("macro@qq.com");
LOGGER.info("Validator isEmail:{}", result);
//判断是否为手机号码
result = Validator.isMobile("18911111111");
LOGGER.info("Validator isMobile:{}", result);
//判断是否为IPV4地址
result = Validator.isIpv4("192.168.3.101");
LOGGER.info("Validator isIpv4:{}", result);
//判断是否为汉字
result = Validator.isChinese("你好");
LOGGER.info("Validator isChinese:{}", result);
//判断是否为身份证号码（18位中国）
result = Validator.isCitizenId("123456");
```

```
LOGGER.info("Validator isCitizenId:{}", result);
//判断是否为URL
result = Validator.isUrl("http://www.baidu.com");
LOGGER.info("Validator isUrl:{}", result);
//判断是否为生日
result = Validator.isBirthday("2020-02-01");
LOGGER.info("Validator isBirthday:{}", result);
```

DigestUtil

摘要算法工具类，支持MD5、SHA-256、Bcrypt等算法。

```
String password = "123456";
//计算MD5摘要值，并转为16进制字符串
String result = DigestUtil.md5Hex(password);
LOGGER.info("DigestUtil md5Hex:{}", result);
//计算SHA-256摘要值，并转为16进制字符串
result = DigestUtil.sha256Hex(password);
LOGGER.info("DigestUtil sha256Hex:{}", result);
//生成Bcrypt加密后的密文，并校验
String hashPwd = DigestUtil.bcrypt(password);
boolean check = DigestUtil.bcryptCheck(password,hashPwd);
LOGGER.info("DigestUtil bcryptCheck:{}", check);
```

HttpUtil

Http请求工具类，可以发起GET/POST等请求。

```
String response = HttpUtil.get("http://localhost:8080/hutool/covert");
LOGGER.info("HttpUtil get:{}", response);
```

其他工具类

Hutool中的工具类还有很多，可以参考：https://www.hutool.cn/

项目源码地址

https://github.com/macrozheng/mall-learning/tree/master/mall-tiny-hutool

推荐阅读

- 还在手动部署**SpringBoot**应用？试试这个自动化插件！
- 为什么我要从 **Windows** 切换到 **Linux?**
- 一键生成数据库文档，堪称数据库界的**Swagger**，有点厉害**!**
- 面对成百上千台服务器产生的日志，试试这款轻量级日志搬运神器！
- 居然有人想白嫖我的日志，赶紧开启安全保护压压惊！
- **Docker**不香吗，为啥还要**K8s**?
- **mall-swarm** 微服务电商项目发布重大更新，打造**Spring Cloud**最佳实践！
- 为什么阿里巴巴禁止使用**Apache BeanUtils**进行属性拷贝？
- **Mall** 电商实战项目发布重大更新，全面支持**SpringBoot 2.3.0** ！
- 我的**Github**开源项目，从**0**到**20000 Star**！

欢迎关注，点个在看

收录于合集 #mall学习教程（参考篇） 27

上一篇
使用Docker Compose部署SpringBoot应用

下一篇
Nginx的这些妙用，你肯定有不知道的！

阅读原文

喜欢此内容的人还喜欢

项目中到底该不该用Lombok?
macrozheng