智能开发助手

最佳实践

文档版本	01
发布日期	2024-07-16

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 CodeArts Snap 在 UT 用例编写方面效率提升

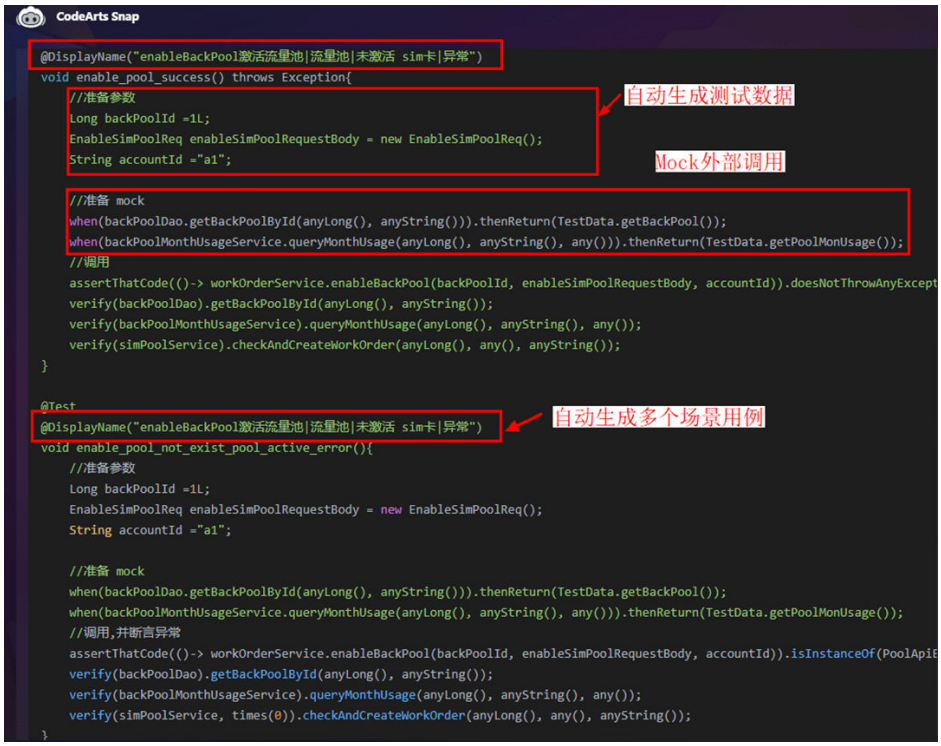
1.1 CodeArts Snap UT 生成案例 - 快速生成 UT 用例

业务痛点

1. 编写单元测试用例较耗时。
2. 单元测试相关的开发框架有一定的学习、使用成本。
3. 补齐历史遗留代码UT用例工作量较大。

生成效果

1. 自动生成多场景用例。
2. 自动生成测试用例数据。
3. 自动Mock外部调用。



使用效果

利用CodeArts Snap生成UT用例224个，平均每个用例生成耗时仅十秒左右，提升UT编写效率20%~30%。

2 CodeArts Snap 在代码续写方面效率提升

2.1 CodeArts Snap 代码生成案例 - 通用逻辑生成

一些常见的算法，比如正则表达式、时间处理函数等算法，具有业务逻辑简单，但是研发人员编写较为复杂的特性（往往是因为复杂的编码规则，需要人员查阅对应的资料）。

生成正则表达式/字符串处理函数

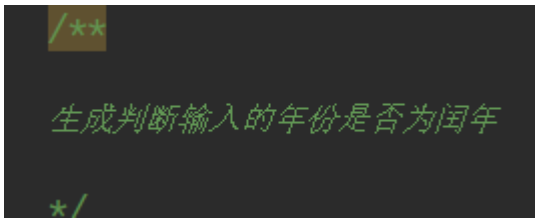
此案例函数逻辑清晰，但是正则的编写对研发人员来说往往需要翻阅资料，较为耗时。

```
/**

使用正则表达式，提取输入的字符串中所有的数字并返回为数组

*/
private static String[] extractDigits(String inputMsg){
```

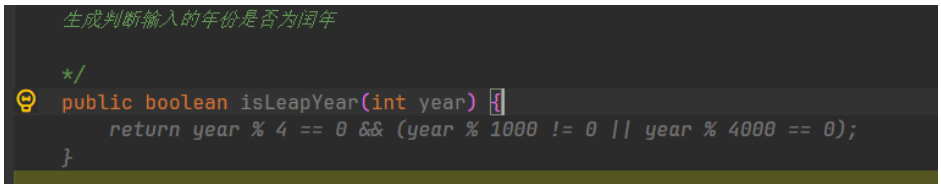
使用代码生成直接生成对应的处理函数，开发人员在此基础上微调即可满足业务要求。

```
50      /**
51      获取字符串中所有的email地址，并返回为数组
52      */
53      private String[] extractEmail(String inputMsg){
54          Next (Alt+]) Previous (Alt+[) Accept (Tab)
55          String[] words = inputMsg.split("\\s+");
            List<String> emailList = new ArrayList<>();
            for(String word : words){
                if(word.matches("[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\\.[a-zA-Z]{2,}")) {
                    emailList.add(word);
                }
            }
            return emailList.toArray(new String[0]);
56      }
57  }
```


生成日期处理函数



使用代码生成，直接根据上述描述生成代码。



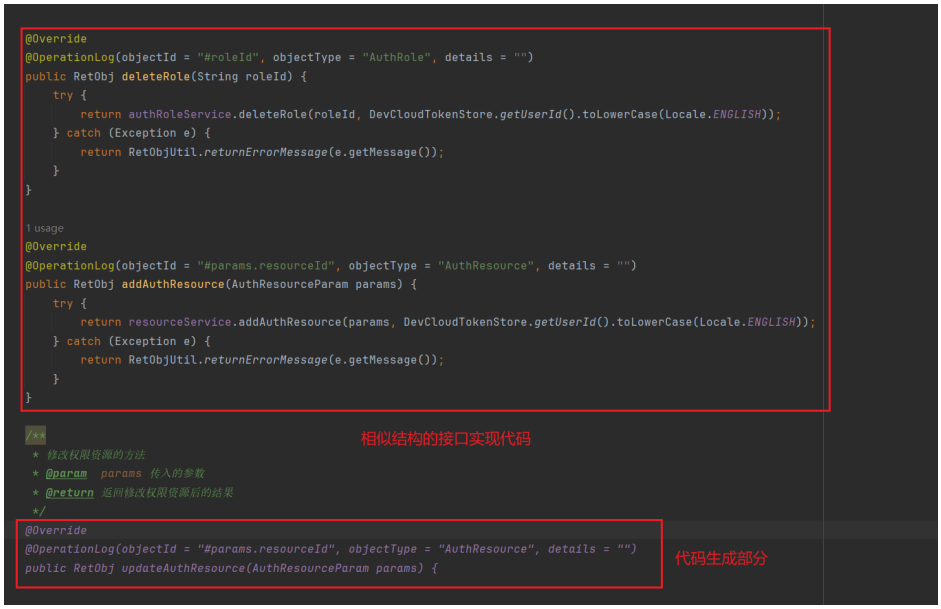
案例总结

可以使用代码生成，快速生成常见的基础算法，让开发人员专注于复杂逻辑处理上。

2.2 CodeArts Snap 代码生成案例 - 快速进行仿写

API Controller层的开发基本上和复杂业务逻辑进行了解耦。同时一个业务内的API相似度很高，可以直接使用代码生成，依赖现有的接口去扩展业务接口。

根据注释生成代码



案例总结

上述项目文件中，已经有结构清晰的上文代码，研发人员在续写的时候，可以通过注释的方式，生成类似结构的代码，完整生成结果如下：

```
/**
 * 修改权限资源的方法
 * @param  params 传入的参数
 * @return 返回修改权限资源后的结果
 */
no usages
@Override
@OperationLog(objectId = "#params.resourceId", objectType = "AuthResource", details = "")
public RetObj updateAuthResource(AuthResourceParam params) {
    try {
        return resourceService.updateAuthResource(params, DevCloudTokenStore.getUserId().toLowerCase(Locale.ENGLISH));
    } catch (Exception e) {
        return RetObjUtil.returnErrorMessage(e.getMessage());
    }
}
```

2.3 CodeArts Snap 代码生成案例 - 数据库接口编写

MyBatis作为常见的数据库框架，经常涉及到大量的接口类生成，并且很多情况下这些接口类都具有类似的格式，因此在上文的基础上生成新的业务接口也是比较常见的代码生成场景。

根据注释生成数据库接口代码

```
// 获取L0的IdList
2 usages
List<ToolAllListCkBillEntity> getL0IdList(@Param("type") String type, @Param("start_time") String startTime,
        @Param("end_time") String endTime);

// 获取L1的IdList
List<ToolAllListCkBillEntity> getL1IdList(@Param("type") String type, @Param("start_time") String startTime,
        @Param("end_time") String endTime);
```

对于类似的业务逻辑，可以直接生成对应的接口。

案例总结

可以通过代码生成能力，快速学习到已有代码的行文风格，并在此基础上快速扩展代码，提升编码效率。

3 CodeArts Snap 在知识搜索方面效率提升

3.1 CodeArts Snap 知识搜索案例 - 未知依赖查询

缺失代码依赖，或者缺少软件包信息，不用去网站搜索，在IDE内部完成查询。

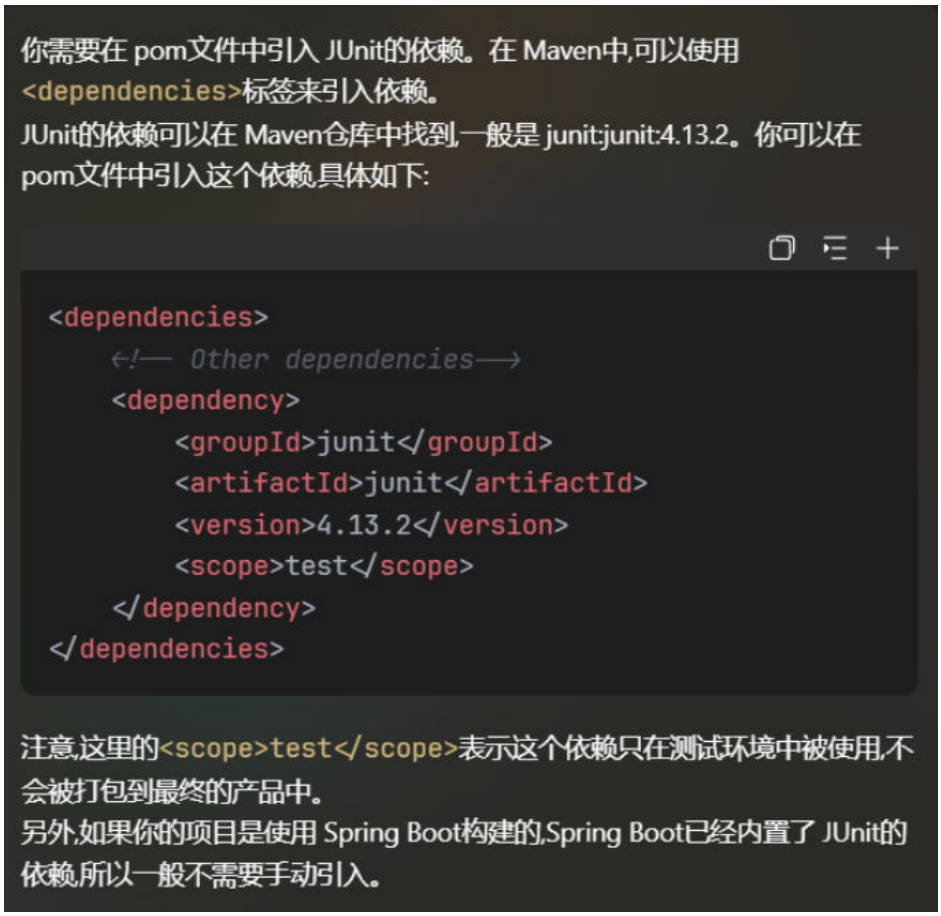
对 CodeArts Snap 提问

我的代码中引入了如下API，请问我需要在pom文件中引入什么依赖：

import org.junit.Before;

import org.junit.Test;

CodeArts Snap 响应结果



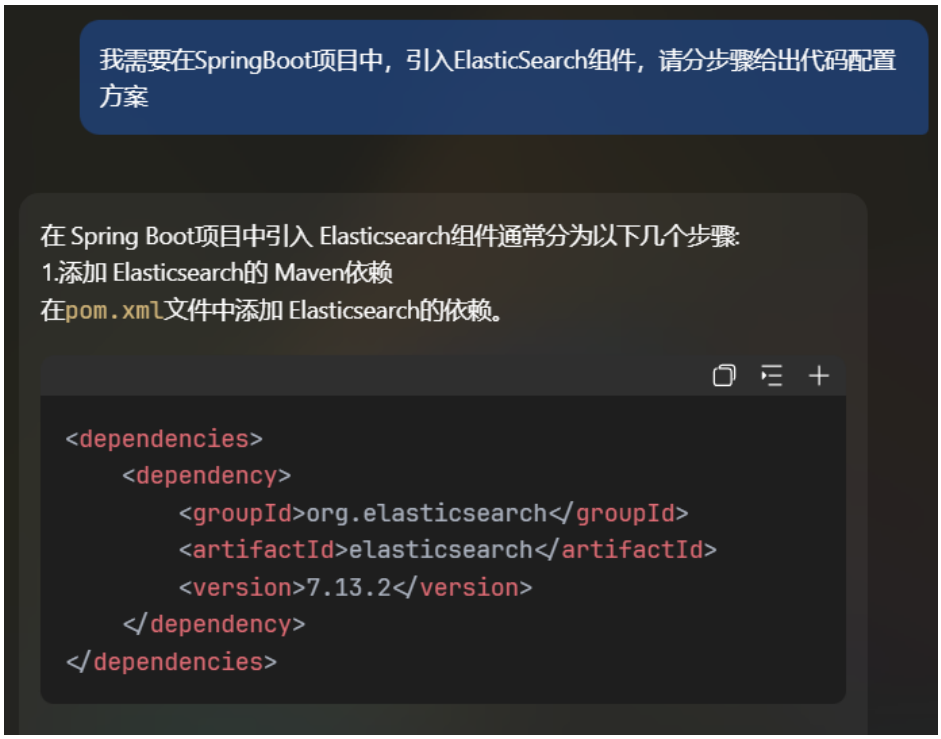
3.2 CodeArts Snap 知识搜索案例 - 直接引入组件

对于常见的三方组件，可以使用CodeArts Snap提示常见的引入方法和代码模板。

对 CodeArts Snap 提问

我需要在SpringBoot项目中，引入ElasticSearch组件，请分步骤给出代码配置方案。

CodeArts Snap 响应结果



4.在 Spring Boot的应用中使用 Elasticsearch
我们可以通过@Autowired注解来注入 Elasticsearch的客户端,然后使用它来进行索引的创建、查询、删除等操作。

```
import org.elasticsearch.action.index.IndexResponse;
import org.elasticsearch.client.Response.Result;
import org.springframework.beans.factory.annotation.Aut
import org.springframework.stereotype.Component;

import java.util.Arrays;

@Component
public class ElasticsearchService{

    @Autowired
    private Client client;

    //创建索引
    public void createIndex(String indexName){
        IndexRequest request = new IndexRequest(indexNa
        client.index(request, RequestOptions.DEFAULT);
    }

    //删除索引
    public void deleteIndex(String indexName){
        IndexRequest request = new IndexRequest(indexNa
        client.delete(request, RequestOptions.DEFAULT);
    }

    //创建类型
    public void createType(String indexName, String typ
        Map<String, Object> settingsMap = new HashMap<>
        settingsMap.put("index.number_of_shards",1);
        settingsMap.put("index.number_of_replicas",0);
```