**工业软件后端 - 工具类测试用例文档**

**文档信息**

**项目名称：**工业软件后端系统  
**功能模块：**工具类测试  
**文档版本：**v1.0  
**创建日期：**2025-08-07  
**测试工具：**JUnit 5

**一、测试概述**

**1.1 测试目标**

验证核心工具类的正确性和稳定性，包括JWT工具、密码工具、文件大小转换工具、分布式锁工具等。

**二、测试用例详细设计**

**2.1 JWT工具类测试**

**测试用例 TC001: JWT Token生成测试**

**用例ID：**TC001  
**测试方法：**JwtUtils.generateToken()  
**测试类型：**功能测试  
**优先级：**高

**测试数据**

**输入参数：**用户信息Map

Map claims = new HashMap<>(); claims.put("id", 1L); claims.put("name", "testuser");

**预期结果**

**返回值：**有效的JWT Token字符串  
**验证点：**Token格式正确，包含用户信息

**测试用例 TC002: JWT Token解析测试**

**用例ID：**TC002  
**测试方法：**JwtUtils.parseToken()  
**测试类型：**功能测试  
**优先级：**高

**测试数据**

**输入参数：**有效的JWT Token

**预期结果**

**返回值：**Claims对象  
**验证点：**正确解析用户ID和用户名

**2.2 密码工具类测试**

**测试用例 TC003: 密码加密测试**

**用例ID：**TC003  
**测试方法：**PasswordUtil.encodePassword()  
**测试类型：**功能测试  
**优先级：**高

**测试数据**

**输入参数：**原始密码字符串

String rawPassword = "password123";

**预期结果**

**返回值：**加密后的密码字符串  
**验证点：**加密结果不为空，与原始密码不同

**测试用例 TC004: 密码验证测试**

**用例ID：**TC004  
**测试方法：**PasswordUtil.matches()  
**测试类型：**功能测试  
**优先级：**高

**测试数据**

**输入参数：**原始密码和加密密码

**预期结果**

**返回值：**true（密码匹配）  
**验证点：**正确密码返回true，错误密码返回false

**2.3 文件大小转换工具类测试**

**测试用例 TC005: 文件大小转换测试**

**用例ID：**TC005  
**测试方法：**TransFileSizeUtil.transFileSize()  
**测试类型：**功能测试  
**优先级：**中

**测试数据**

**测试用例：**

// 字节级别 transFileSize(1023L) -> "1023 B" // KB级别 transFileSize(1024L) -> "1.00 KB" // MB级别 transFileSize(1024 \* 1024L) -> "1.00 MB" // GB级别 transFileSize(1024 \* 1024 \* 1024L) -> "1.00 GB" // 边界值 transFileSize(null) -> "0 B" transFileSize(-1L) -> "0 B"

**预期结果**

**返回值：**格式化后的文件大小字符串  
**验证点：**正确转换不同大小的文件，处理边界值

**2.4 分布式锁工具类测试**

**测试用例 TC006: 分布式锁获取测试**

**用例ID：**TC006  
**测试方法：**DistributedLockUtil.executeWithLock()  
**测试类型：**功能测试  
**优先级：**高

**测试数据**

**输入参数：**锁键、过期时间、业务操作

String lockKey = "test:lock:1"; long expireTime = 10; TimeUnit timeUnit = TimeUnit.SECONDS; Supplier action = () -> "业务执行成功";

**预期结果**

**返回值：**业务操作结果  
**验证点：**成功获取锁，执行业务逻辑，自动释放锁

**测试用例 TC007: 分布式锁冲突测试**

**用例ID：**TC007  
**测试方法：**DistributedLockUtil.tryExecuteWithLock()  
**测试类型：**异常测试  
**优先级：**中

**测试数据**

**测试场景：**并发获取同一把锁

**预期结果**

**返回值：**默认值  
**验证点：**获取锁失败时返回默认值，不抛出异常

**三、测试数据准备**

**3.1 JWT测试数据**

// 测试用户信息 Map testClaims = { "id": 1L, "name": "testuser", "email": "test@example.com" } // 测试Token String testToken = "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9..."

**3.2 密码测试数据**

// 测试密码 String testPassword = "password123"; String wrongPassword = "wrongpassword"; // 预期加密结果（BCrypt每次结果不同） String expectedPattern = "^\\$2a\\$\\d{1,2}\\$[./A-Za-z0-9]{53}$";

**3.3 文件大小测试数据**

// 测试文件大小 Long[] testSizes = { 0L, 1023L, 1024L, 1025L, 1024 \* 1024L, 1024 \* 1024 \* 1024L, null, -1L }

**3.4 分布式锁测试数据**

// 测试锁键 String[] testLockKeys = { "user:lock:1", "order:lock:123", "payment:lock:456" } // 测试过期时间 long[] testExpireTimes = {5, 10, 30, 60};

**四、测试环境配置**

**4.1 依赖配置**

// Maven依赖 org.junit.jupiter junit-jupiter test org.mockito mockito-core test org.springframework.boot spring-boot-starter-test test

**4.2 Redis配置**

// 测试环境Redis配置 spring: redis: host: localhost port: 6379 database: 1 # 使用不同的数据库避免影响生产数据

**4.3 测试注解**

// 常用测试注解 @Test - 标记测试方法 @BeforeEach - 每个测试方法前执行 @AfterEach - 每个测试方法后执行 @Mock - 创建Mock对象 @InjectMocks - 注入Mock对象 @ExtendWith(MockitoExtension.class) - 启用Mockito扩展