* 学习目标

* 能够使用C3P0数据库连接池

  * 导入jar，配置文件，使用某个数据源

  * 参数了解清楚一点

* 能够使用Druid数据库连接池

  * 导入jar，配置文件，使用某个数据源

  * 参数了解清楚一点

* 能够了解常见加密算法

    * 单向加密：MD5，SHA

    * 对称加密：DES

    * 非对称加密：RSA

* 能够给Druid数据源配置文件密码加密

      * ConfigTools:对私钥进行加密，对公钥进行解密

      * 配置加密密码

        * filters

        * connectionProperties

---

* 回顾

  * 代理模式

    * 用户，代理，目标对象（被代理）

    * 提供额外的功能

  * 分类：静态代理，动态代理（JDK代理（接口代理），子类代理（CGLib代理））

  * 静态代理

    * 代理对象给目标对象额外增加功能

    * 代理对象和目标对象有共同接口

    * 代理对象持有目标对象引用

    * 运行前，提交准备好的（手工创建代理类或者有其他编译器自动创建代理类）

* 静态代理类：导致代理类很多，增加工作量，提供维护难度

* 动态代理--JDK代理

* 在内存中生成代理类

* API

```
Object proxy=
Proxy.newInstanceProxy(target.getClass().getClassLoader().target.getClass().getInterfaces(),
                new InvocationHandler(){
                  @override
                   public Object invoke(Object obj,Method method,Object[] args){
                         // 前后或者出现异常的时候，额外添加的功能
                         method.invoke(target,args);
                    }
           });
```

* 动态代理--子类代理

* 在内存中生成目标对象子类

* API

* 引入jar：cglib，asm

```
Enhancer enhancer=new Enhancer();
enhancer.setSuperClass(target.getClass());
enhancer.setCallback(
  new MethodInterceptor(){
     @override
     public Object intercept(Object obj,Method method,Object[] args,MethodProxy proxy){
                         method.invoke(target,args);
      }
    }
```

);

Object proxy=enhancer.create();

* 自定义连接池

* 池：LinkedList

* Connection:close--->关闭数据库链接--->放到池

* 动态代理：代理是Connection的子类

* 能够使用C3P0数据库连接池

* C3P0概述

* C3P0是一个开源的JDBC连接池，它实现了数据源和JNDI绑定，支持JDBC3规范和JDBC2的标准扩展。目前使用它的开源项目有Hibernate、Spring等

```
* 使用C3P0
 * 导入jar包
    * c3p0-0.9.5.4.jar
    * c3p0-oracle-thin-extras-0.9.5.4.jar
    * mchange-commons-java-0.2.15.jar
 * copy配置文件c3p0-config.xml
 <?xml version="1.0" encoding="UTF-8"?>
<c3p0-config>
    <!-- C3P0的缺省(默认)配置，
       如果在代码中“ComboPooledDataSource ds =
       new ComboPooledDataSource();”
         这样写就表示使用的是C3P0的缺省(默认)配置信息来创建数据源 -->
    <default-config>
        <property name="driverClass">oracle.jdbc.driver.OracleDriver</property>
        <property name="jdbcUrl">jdbc:oracle:thin:@192.168.1.121:1521/orcl</pro
        <property name="user">scott</property>
        <property name="password">tiger</property>
        <property name="acquireIncrement">5</property>
        <property name="initialPoolSize">10</property>
        <property name="minPoolSize">5</property>
        <property name="maxPoolSize">20</property>
    </default-config>

    <!-- C3P0的命名配置，
     如果在代码中“ComboPooledDataSource ds =
```

```
     new ComboPooledDataSource("MySQL");
     ”这样写就表示使用的是name是MySQL的配置信息来创建数据源 -->
    <named-config name="MySQL">
        <property name="driverClass">com.mysql.jdbc.Driver</property>
        <property name="jdbcUrl">jdbc:mysql://127.0.0.1:3306/test</property>
        <property name="user">root</property>
        <property name="password">root</property>
        <property name="acquireIncrement">5</property>
        <property name="initialPoolSize">10</property>
        <property name="minPoolSize">5</property>
        <property name="maxPoolSize">20</property>
    </named-config>
</c3p0-config>

 * 链接工具类
 public class ConnectionUtilsC3P0 {
    public static DataSource dataSource=new ComboPooledDataSource();
    // 获得链接
    public static Connection getConn() {
        try {
            return dataSource.getConnection();
        } catch (SQLException e) {
            e.printStackTrace();
        }
        return null;
    }

    // 释放资源
    public static void close(Connection con,Statement st,ResultSet rs) {
        if(rs!=null) {
            try {
                rs.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        if(st!=null) {
            try {
                st.close();
            } catch (SQLException e) {
```

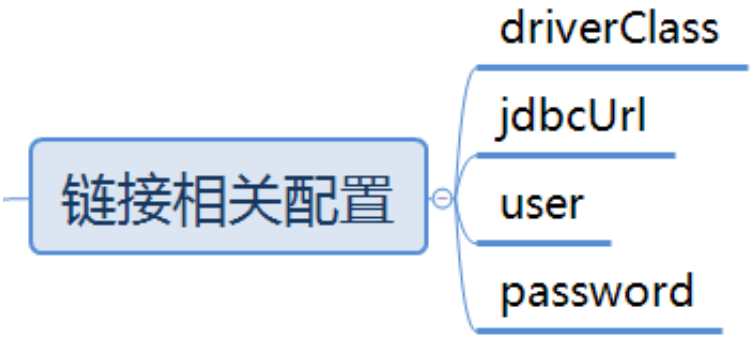
```
                e.printStackTrace();
            }
        }
        if(con!=null) {
            try {
                con.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }

    // 关闭连接
    public static void closeConn(Connection conn) {
        close(conn, null, null);
    }

    // 关闭SQL执行对象
    public static void closeSt(Statement st) {
        close(null, st, null);
    }

    //关闭结果集
    public static void closeRs(ResultSet rs) {
        close(null, null, rs);
    }

}
 * UserDaoImpl更改获取链接方式
   * Connection conn = ConnectionUtilsC3P0.getConn();
 * 单元测试
 @Test
    public void test1() throws SQLException {
         UserDao userDao=new  UserDaoImpl();
         User user=new User();
         user.setName("刘备");
         user.setPassword(MD5Utils.md5("123"));
         userDao.add(user);
    }
 * 假如需要日志的显示
```

```
106     * 添加log4j的jar
107     * log4j.properties 配置文件
108
```

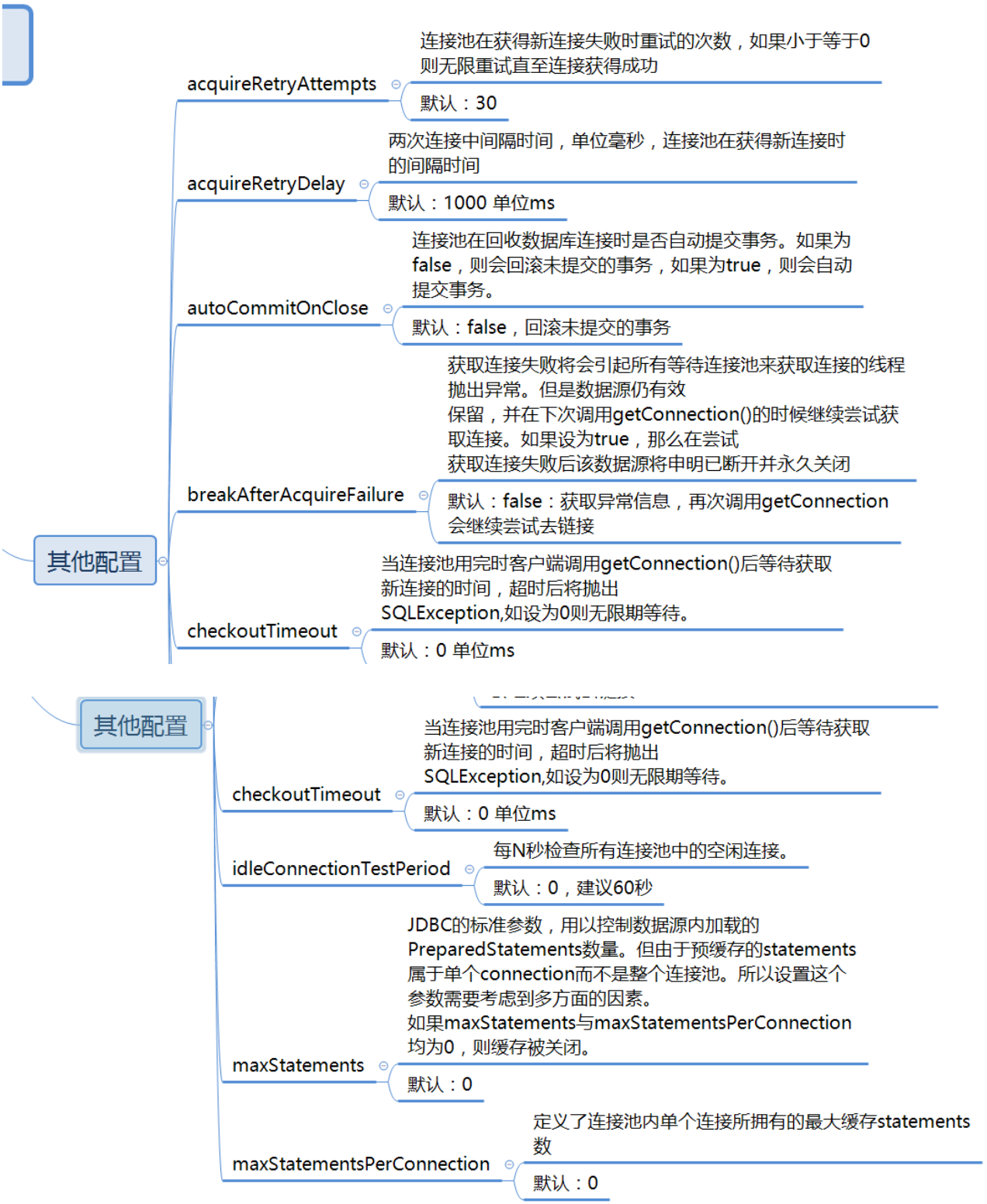
* c3p0 常见配置参数

  * 基础配置

* 数据库链接配置



* 其他配置



* 能够使用Druid数据库连接池

  * Druid 概述

    * 阿里巴巴数据库事业部出品，为监控而生的数据库连接池。

      * 例如阿里云，也是用druid

    * Druid 是一个 JDBC 组件库，包含数据库连接池、SQL Parser 等组件,

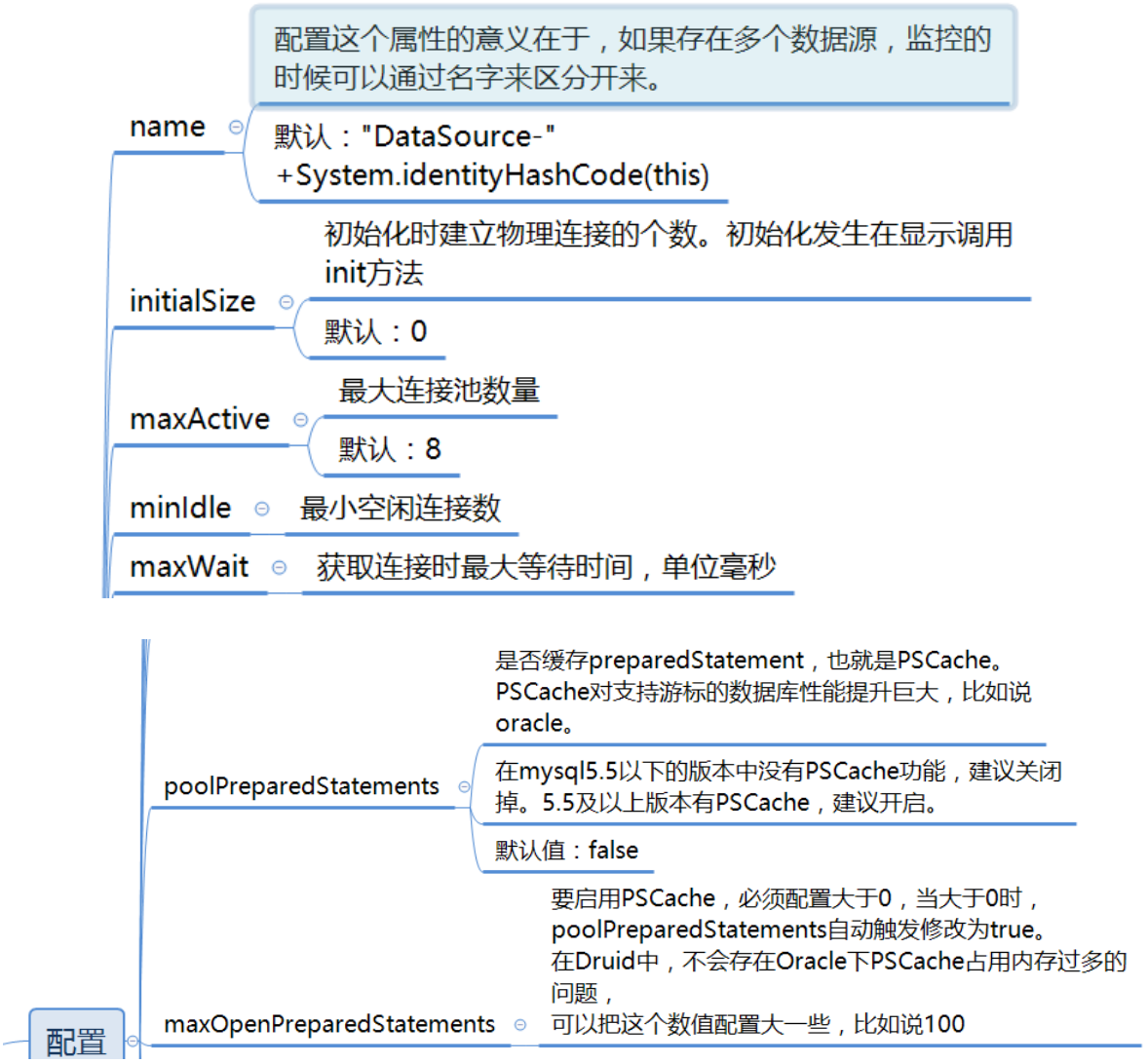
* 被大量业务和技术产品使用或集成，经历过最严苛线上业务场景考验，是你值得信赖的技术产品。

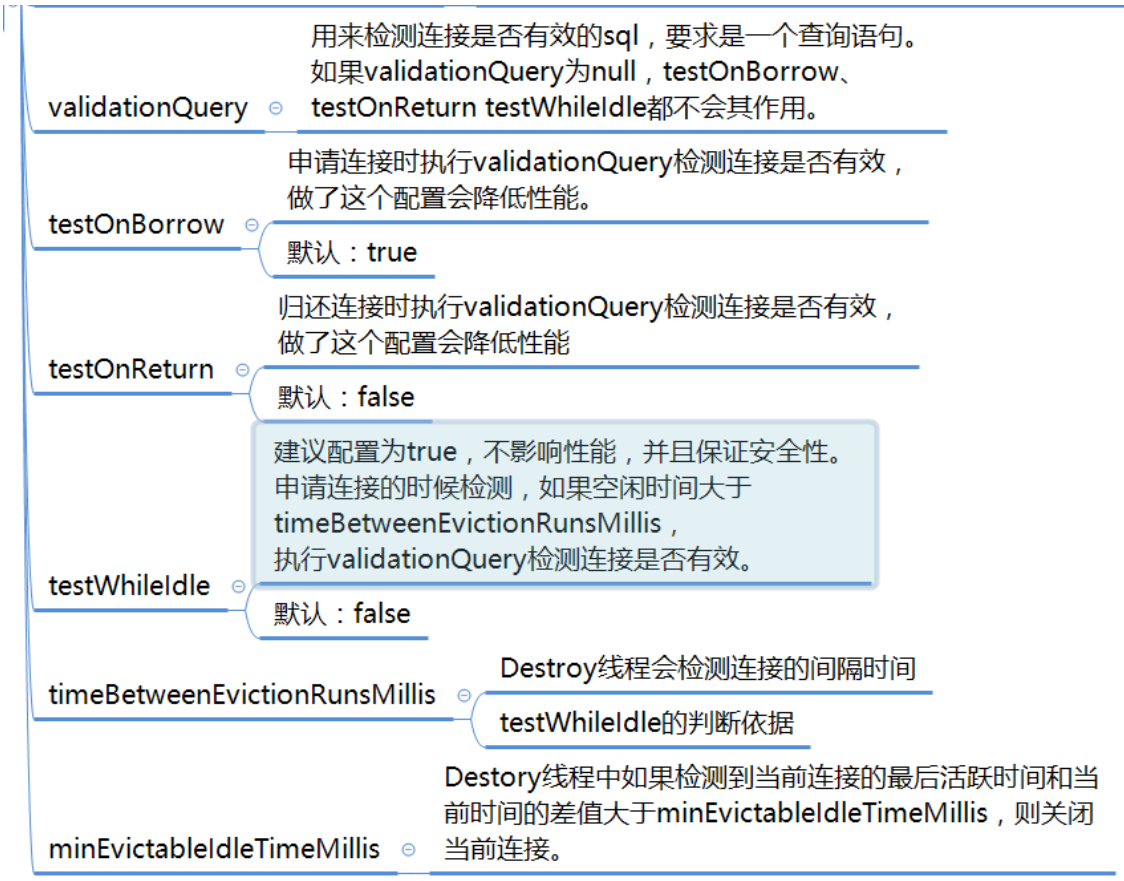
```
* 使用Druid
 * 导入jar包
    * druid-1.0.16.jar
 * copy配置文件druid.properties
   driverClassName=oracle.jdbc.driver.OracleDriver
   url=jdbc:oracle:thin:@192.168.1.121:1521/orcl
   username=scott
   password=tiger
   initialSize=5
   timeBetweenEvictionRunsMillis=60000
   minEvictableIdleTimeMillis=300000
   validationQuery=SELECT 1 from dual

 * ConnectionUtilsDruid 编写
 public class ConnectionUtilsDruid {
    public static DruidDataSource dataSource;
    static {
    InputStream is = ConnectionUtilsDruid.class.getClassLoader().getResourceAsS
    Properties props=new Properties();
        try {
            props.load(is);
            dataSource=(DruidDataSource) DruidDataSourceFactory.createDataSourc
        } catch (IOException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }

    }
    // 获得链接
    public static Connection getConn() {
        try {
            return dataSource.getConnection();
        } catch (SQLException e) {
            e.printStackTrace();
        }
```

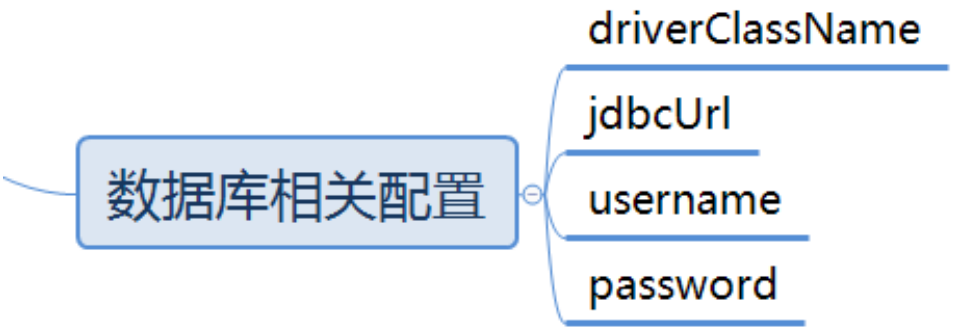
```
        return null;
    }

    // 释放资源
    public static void close(Connection con,Statement st,ResultSet rs) {
        if(rs!=null) {
            try {
                rs.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        if(st!=null) {
            try {
                st.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
        if(con!=null) {
            try {
                con.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }

    // 关闭连接
    public static void closeConn(Connection conn) {
        close(conn, null, null);
    }

    // 关闭SQL执行对象
    public static void closeSt(Statement st) {
        close(null, st, null);
    }

    //关闭结果集
    public static void closeRs(ResultSet rs) {
```

```
77          close(null, null, rs);
78      }
79
80  }
81   * UserDaoImpl更改获取链接方式
82     * Connection conn = ConnectionUtilsDruid.getConn();
83   * 使用单元测试
84   * 假如需要日志的显示
85      * 添加log4j的jar
86      * log4j.properties 配置文件
```

* Druid数据库连接池配置







* 能够了解常见加密算法

  * 参考Java安全技术研究_change.pptx

```
* 加密分类
  * 单向加密：MD5，SHA
    * MD5Util
    * SHAUtil
  * 对称加密：DES
    * DesUtil
  * 非对称加密：RSA
```

```
    * RSAUtil

* MD5Util测试
  @Test
  public void test1() {
        String md5=MD5Util.md5("123");
        System.out.println(md5);
  }
  结果:
  202cb962ac59075b964b07152d234b70

* SHAUtil测试
    @Test
    public void test2() {
        String sha = SHAUtil.SHAEncode("123");
        System.out.println(sha);
    }
 * 结果
    40bd001563085fc35165329ea1ff5c5ecbdbbeef

* DesUtil 测试
@Test
    public void test3() throws Exception {
        // 加密
        String key="abc123de";
        String data="亮哥教育";
        String encrypt = DesUtil.encrypt(data, key);
        System.out.println(encrypt);

        // 解密
        String decrypt = DesUtil.decrypt(encrypt, key);
        System.out.println(decrypt);
    }
结果:
9mRRlpoW4FARsL3gI1hMbw==
亮哥教育

* RSAUtil测试
 @Test
    public void test5() throws Exception {
```

```
        // 产生一对密钥
        Map<String, Object> genKeyPair = RSAUtil.genKeyPair();
        // 获取公钥
        String publicKey = RSAUtil.getPublicKey(genKeyPair);
        // 获取私钥
        String privateKey = RSAUtil.getPrivateKey(genKeyPair);
        System.out.println("publicKey:");
        System.out.println(publicKey);
        System.out.println("privateKey:");
        System.out.println(privateKey);
        // 用公钥加密，私钥解密
        byte[] data="亮哥教育".getBytes();
        byte[] encryptByPublicKey = RSAUtil.encryptByPublicKey(data, publicKey)
        System.out.println("公钥加密后的数据:");
        System.out.println(new String(encryptByPublicKey));
        byte[] decryptByPrivateKey = RSAUtil.decryptByPrivateKey(encryptByPubli
        System.out.println("私钥解密后的数据:");
        System.out.println(new String(decryptByPrivateKey));
        System.out.println("----------------------------------------");
        // 用私钥加密，公钥解密
        byte[] encryptByPrivateKey = RSAUtil.encryptByPrivateKey(data, privateK
        System.out.println("私钥加密后的数据:");
        System.out.println(new String(encryptByPrivateKey));
        byte[] decryptByPublicKey = RSAUtil.decryptByPublicKey(encryptByPrivate
        System.out.println("公钥解密后的数据:");
        System.out.println(new String(decryptByPublicKey));
    }
 * 结果
publicKey:
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDbEU8AU//S1mZPLWNoksgKBsJlepEqHtro/Wjd
QY42c2T3rLXJ1J4YFCtWfks8hEyTGxSZ2BAJJH2UvjN690ZYNq+Brmn41QFLNXMAWpDE1jqZtRgc
r7jvjfkLRmH+g5M3FD6C7gV0yCNcAg7cHTPyPufSEoqUNg0aMEvTJGpy3wIDAQAB
privateKey:
MIICdwIBADANBgkqhkiG9w0BAQEFAASCAmEwggJdAgEAAoGBANsRTwBT/9LWZk8tY2iSyAoGwmV6
kSoe2uj9aN1BjjZzZPestcnUnhgUK1Z+SzyETJMbFJnYEAkkfZS+M3r3Rlg2r4GuafjVAUs1cwBa
kMTWOpm1GByvuO+N+QtGYf6DkzcUPoLuBXTII1wCDtwdM/I+59ISipQ2DRowS9MkanLfAgMBAAEC
gYBBh4fuPTJFS0UHYjheS6ny9dqoqGVMCDbLgyIfLUDxIMuPGua2HRe9dCsSkGzJCXoududOrcb+
NSArokmb+uPbabH4jJJU+k6g1U0fJ77/WyRhic5rO55aecU9mvg84ZScdFUoE3mO+c4np/RlAjMv
q43+pHi0AyRUVHgSIBIoQQJBAO48YoGfHxUMaFtZ5zmdeW7s/ZRVN1eGXgFVEk5E8NmwRljyDqNu
i/2ipiiNn01Zqra9cdTtDApBUsV2Z4fCbTkCQQDrZwa/ldAGonjD33pTRqwPqFT01TduHX2lgfno
```

```
GxIHDYlzxNlN2+D21bqF7eymlcaiXYlr2eiLltICrGHyXnjXAkEArte+a+c+RpJenHy0yIIcNx7T
VGPeQ+wnXc7zqmHKadmHJ4wu1h8xuqn+TFD0Ey61rXwH6P84EOa0OpuVi/tTcQJBALJVlb40j6lH
ErhJKFHIW3dtLhdhJCIOw25MXnfhcjHXpocG3R1n8zDsT+dAjSTSmO3OcNo89jgx5R3TXEySF3cC
QGEQWk2MVXOTRZUQCUNAURAzl4h5yfny7oUEgcfSvSt4kCCWFCPjkaxtsYflsK0iOK18tW9rZiJP
dH8bYDIEZM8=
公钥加密后的数据:
••袍•蠥•?,侏齣&•u?8?•鋿检掷掟U/ W?•吉•跕%0,筵mY?•?•?2洟&,?••??{撏?••?••+咧釭f•□ \
私钥解密后的数据:
亮哥教育
----------------------------------------
私钥加密后的数据:
[K?•蔬'r]•苣U•NI••?"o?•鳩E亮>$穗桩Ka薆渺*w冥•J□+;C•□2g*?•傒桲藣塔o跟A耿g敕O>?儤 ﾊ
公钥解密后的数据:
亮哥教育
```

* 能够给Druid数据源配置文件密码加密

```
 * Druid 加密的工具类ConfigTools测试
   * 注意文件的编码：UTF-8，否则无法加密中文
   * 提供了私钥加密，公钥解密，没有提供公钥加密，私钥解密的方法
@Test
    public void test6() throws Exception {
        String[] genKeyPair2 = ConfigTools.genKeyPair(1024);
        // 获取私钥
        String privateKey = genKeyPair2[0];
        // 获取公钥
        String publicKey = genKeyPair2[1];
        System.out.println("publicKey:");
        System.out.println(publicKey);
        System.out.println("privateKey:");
        System.out.println(privateKey);
        // 用私钥加密，公钥解密
        String data="tiger";
        String encrypt = ConfigTools.encrypt(privateKey, data);
        System.out.println("私钥加密后的数据:");
        System.out.println(encrypt);
```

```
        String decrypt=ConfigTools.decrypt(publicKey, encrypt);
        System.out.println("公钥解密后的数据:");
        System.out.println(decrypt);
    }
结果:
publicKey:
MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCfIAncXk6he3vSKS42mnLqgoWWOEvM6K4jxHKm2kA
privateKey:
MIICdgIBADANBgkqhkiG9w0BAQEFAASCAmAwggJcAgEAAoGBAJ8gCdxeTqF7e9IpLjaacuqChZY4S8z
私钥加密后的数据:
a3XspqUwY5bOYa4yoYJAK89m/enZfv4MerDoLM7Zpapw9XwbzNHnd0EGxBl2r7piuInSNfG39q5sDg+
公钥解密后的数据:
tiger

* 温馨提示:
    * 记得保存好自己的私钥
* 修改配置文件
driverClassName=oracle.jdbc.driver.OracleDriver
url=jdbc:oracle:thin:@192.168.1.121:1521/orcl
username=scott
password=a3XspqUwY5bOYa4yoYJAK89m/enZfv4MerDoLM7Zpapw9XwbzNHnd0EGxBl2r7piuInSNf
#url=jdbc:mysql://localhost:3306/test
#username=root
#password=root
initialSize=5
timeBetweenEvictionRunsMillis=60000
minEvictableIdleTimeMillis=300000
validationQuery=SELECT 1 from dual
filters=config
connectionProperties=config.decrypt=true;config.decrypt.key=MIGfMA0GCSqGSIb3DQE

* 测试
  * 可以使用

```