1.idea中出现乱码:

a.settings-->File Encodings 改为UTF-8

b.打开idea的工作目录，将idea64.exe.vmoptions和idea.exe.vmoptions在最后追加 -Dfile.encoding=UTF-8

c.配置tomcat的页面中:VMOption配置:-Dfile.encoding=UTF-8

2.热部署问题

Update:更新操作:很多时候无效

Frame:idea失去焦点触发

推荐选项:

Update:任意

Frame:update classes and resources

idea: 热部署，如果Run启动，仅jsp有效

如果是debug启动，jsp和java等均有效

3.jar包问题

1.java项目,直接在src下就可以了

2.web项目:

a.jar包本身只在运行时有效，只需要复制到web-Content/lib/中即可

b.jar包要在各个运行时有效,要复制到web-Content/lib/和src下才可以

3.JNDI:java命名与目录接口

jndi:将某一个对象和资源(对象)，以配置文件(tomcat/conf/context.xml)的形式写入;

<Environment name="jndiName" value="jndiValue" type="java.lang.String"/>

在jsp中

```
Context ctx=new InitialContext();
String testJndi=
(String)ctx.lookup("java:comp/evn/jndiName");
out.print(testJndi);
```

4.连接池(database connection pool)

怎么用?

不用连接池

```
Class.forName("");
```

```
Connection conn=DriverManager.getConnection();
```

用连接池的核心：将连接的指向改了,现在指向的是数据源,而不是数据库

```
Class.forName("oracle.jdbc.driver.OracleDriver");
Context ctx=new InitialContext();
DataSource ds=(DataSource) ctx.lookup("java:comp/evn/student");
conn=ds.getConnection();
```

常见连接池:Tomcat-dbcp　dbcp　c3p0　druid

可以使用数据源(javax.sql.DataSource)管理连接池

DataSource是所有sql数据源的上级类

BasicDataSource是所有dbcp类型的数据源的上级类

1.Tomcat-dbcp ：

a.类型jndi,在context.xml中配置

```
<Resource
    name="student"
    auth="Container"
    type="javax.sql.DataSource"
    driverClassName="oracle.jdbc.driver.OracleDriver"
    username="jiang"
    password="bjsxt"
    maxActive="100"
    maxIdle="40"
    minIdle="10"
    maxWait="5000"
   defaultAutoCommit="false"
   url="jdbc:oracle:thin:@localhost:1521:orcl"/>
```

b.在web.xml中配置

```
<resource-ref>
   <res-ref-name>student</res-ref-name>
   <res-auth>Container</res-auth>
   <res-type>javax.sql.DataSource</res-type>
</resource-ref>
```

c.获取

```
Class.forName("oracle.jdbc.driver.OracleDriver");
Context ctx=new InitialContext();
DataSource ds=(DataSource) ctx.lookup("java:comp/evn/student");
conn=ds.getConnection();
```

属性	简介
name	指定Resource的JNDI名字
auth	指定Resource的管理者，共有两个可选值：Container和Application。 Container：由容器来创建Resource。 Application：由Web应用来创建和管理Resource
Type	指定Resource的类型
maxActive	指定连接池中，处于活动状态的数据库连接的最大数量；如果值为0，表示不受限制
maxIdle	指定连接池中，处于空闲状态的数据库连接的最大数量；如果值为0，表示不受限制
maxWait	指定连接池中，连接处于空闲状态的最长时间(单位是毫秒)，如果超出此最长时间将会抛出异常。如果值为-1，表示允许无限制等待。
username	指定访问数据库的用户名
password	指定访问数据库的密码
driverClassName	指定连接数据库的驱动程序的类名
url	指定连接数据库的URL

2. dbcp连接池

引入jar包

commons-dbcp-1.4.jar 以及依赖jar包commons-pool.jar

BasicDataSource,BasicDataSourceFactory

BasicDataSource方式:

```
public static DataSource getDataSourceWithDBCP(){
    BasicDataSource dbcp=new BasicDataSource();
    dbcp.setDriverClassName("oracle.jdbc.driver.OracleDriver");
    dbcp.setUrl("jdbc:oracle:thin:@localhost:1521:orcl");
    dbcp.setUsername("bjsxt");
    dbcp.setPassword("jiang");
    dbcp.setInitialSize(20);
    dbcp.setMaxActive(10);
    dbcp.setMaxIdle(20);
    dbcp.setMaxWait(5000);
    return dbcp;
}
```

BasicDataSourceFactory方式:

方法	简介
… void setDriverClassName(String driverClassName)	设置连接数据库的驱动名
… void void setUrl(String url)	设置连接数据库的 URL
… void setUsername(String username)	设置数据库的用户名
… void setPassword(String password)	设置数据库的密码
… void setInitialSize(int initialSize)	设置初始化时，连接池中的连接数量
… void setMaxActive(int maxActive)	设置连接池中，处于活动状态的数据库连接的最大数量
… void setMinIdle(int minIdle)	设置连接池中，处于空闲状态的数据库连接的最小数量
… Collection getConnection () throws SQLException	从连接池中获取一个数据库连接

3.c3p0连接池:ComboPooledDataSource

导入jar包:c3p0.jar,c3p0-oracle-thin-extras.jar（oracle兼容jar包）

两种方式

硬编码

配置文件

->合二为一，通过ComboPooledDataSource的构造方法参数区分:如果无参,硬编码;有参,通过配置文件

a.无参，硬编码

```
ComboPooledDataSource c3p0=new ComboPooledDataSource();
c3p0.setDriverClass("oracle.jdbc.driver.OracleDriver");
c3p0.setJdbcUrl("jdbc:oracle:thin:@localhost:1521:orcl");
c3p0.setUser("jiang");
c3p0.setPassword("bjsxt");
return c3p0;
```

b.有参，配置文件(c3p0-config.xml)

```
<?xml varsion="4.0" encoding="UTF-8"?>
<c3p0-config>
  <!--默认-->
  <default-config>
    <property name="user"></property>
    <property name="password"></property>
    <property name="driverClass"></property>
    <property name="jdbcUrl"></property>
    <property name="maxIdel"></property>
    <property name="checkoutTimeout"></property>

  </default-config>
```

```
    <!--自定义一个-->
    <named-config name="jiang">


    </named-config>

    <named-config name="jiang">


    </named-config>
</c3p0-config>
//指定一个,没有的参数从default获取
ComboPooledDataSource c3p0=new ComboPooledDataSource("jiang");
//也可以直接使用defaulr
ComboPooledDataSource c3p01=new ComboPooledDataSource();
return c3p0;
```

方法	简称
public ComboPooledDataSource() public ComboPooledDataSource(String configName)	构造方法，用于创建 ComboPooledDataSource 对象。
public void setDriverClass(String driverClass) throws PropertyVetoException	设置连接数据库的驱动名
public void setJdbcUrl(String jdbcUrl)	设置连接数据库的 URL
public void setUser(String user)	设置数据库的用户名
public void setPassword(String password)	设置数据库的密码
public void setMaxPoolSize(int maxPoolSize)	设置连接池的最大连接数目
public void setMinPoolSize(int minPoolSize)	设置连接池的最小连接数目
public void setInitialPoolSize(int initialPoolSize)	设置初始化时，连接池中的连接数量
public Connection getConnection() throws SQLException	从连接池中获取一个数据库连接。该方法是由 ComboPooledDataSource 的父类 AbstractPoolBackedDataSource 提供。