**今天完成的事情**：

早晨一直在部署Resin，由于开始运行resin.sh不成功，然后又不知道修改什么端口来访问，再后面不知道把war项目包放在那里能加载，所以把resin的配置文件各种修改了，导致了后面虽然理解了配置文件，但是因为配置参数都改错了，所以无法正常加载。在最后各种尝试过后，决定停止服务，删除文件夹，重新来过。

在此记录几个部署的步骤，以及几个遇到的问题：

1. **启动**

找到resin的bin文件夹，使用绝对路径的方式启动resin.sh

既输入命令：sh \*\*\*\*\*/resin-pro-4.0.44/bin/resin.sh start

1. **安装war包，修改resin.xml配置**
2. **首先介绍resin.xml配置**
3. **概念和命名约定**

**1# cluster（集群）**

一个集群包含多个server

**2# environment（环境）**

即应用享有的资源，server，host，web-app是最主要的环境

**3# host**

一个HTTP访问链接的虚拟host主机地址

**4# proxy cache（代理缓存）**

HTTP代理缓存。

收到请求之后，检查本地的缓存来确定是否要向后端服务器发起请求，向后端发起请求并收到回复的情况下，把响应回复又缓存在本地。类似nginx

**5# resource（资源）**

应用可使用的驱动或服务。

包含：JNDI，CDI，JMS队列，习惯性CDI配置。

Resin特有的资源含：安全，身份验证，健康检测和重定向/URL映射系统。

**6# rewrite/dispatch（重定向/调度）**

定向HTTP请求到servlet和相应响应代码。

类似于使用apache的mod\_rewrite进行URL地址的重定向。

**7# server（服务）**

一个Resin JVM实例。一台机器上可以有多个server。

**8# watchdog（监控）**

一个用于监控Resin server的JVM实例，它可以在它认为有必要的时候自动重启server。

**9# web-app （web应用）**

一个HTTP web应用，里面包含了servlet。

1. **默认配置：服务默认配置，集群默认配置，web应用默认配置**

**（server-default,cluster-default, web-app-default）**

想给一组server配一些默认项，比如最大线程数，http端口号时，可以把这些公共的配置放在<server-default>中。

对于cluster，virtual-host或web-app-default，可以用<cluster-default>,<host-default>和<web-app-default>去管理这些公共配置。

1. **host：HTTP虚拟host**

用<host>标签可以配置一个HTTP虚拟host。host的名字用id或host-name属性标识。名字为空的host，如id=””这样的是默认host。

host可以专门用<host>标签配置，也可以用<host-deploy>标签隐式配置。host们共享<host-default>中的公共配置。<host-deploy>中包含一个隐式定义的host的目录，比如说”webapps”目录用于<web-app-deploy>。

<host>标签映射一个HTTP url到一个虚拟host上。该host可以对应单或多个<web-app-default>、<web-app>、<web-app-deploy>中的web-app。

对于cluster来说，<host-deploy>可以当做命令行部署的目的地。所有cluster中的server可以继承拓展命令行部署的host。

因为host可定义环境资源，因此它可以定义共享资源如数据库，缓存，CDI服务和队列。host也可以定义URL映射规则。

1. **Web应用：HTTP/Servlet应用（即静态或动态的应用）**

一个web-app是一个基于http请求的应用，用servlet来处理请求。web-apps可以在WEB-INF/web.xml和WEB-INF/resin-web.xml中配置。一般用my-web-app.war这样的文件部署或者通过命令行以集群的形式部署，也可以把它按一个文件夹的形式放在webapps目录下。

在resin.xml配置文件中，web-app可以用<web-app>配置在<host>中或用<web-app-deploy>标签的形式隐式配置。web-app共享<web-app-default>中的公共配置。<web-app-deploy>指定一个webapps目录。

1. **负载均衡：用于对多个后端服务器负载均衡**

Resin的HTTP负载均衡基于轮询机制分配后端服务器的HTTP流量，在服务器发生故障时可自动切换。负载均衡基于resin的集群，当有添加或删除server的情况时可以自动弹性处理。

负载均衡基于重定向操作，soket参数和超时标准可以在<server>中配置。它可以识别服务器是否有效，超时或过载。

1. **日志：JDKjava.util.logging**

Resin的日志打印基于JDK的java.util.logging。

另外，watchdog会把控制台输出打印到log/jvm-server.log。

JDK的日志打印模块由两个元素构成：log handler和logger。

log handler获得日志信息并处理。Resin进行基本的日志格式化并将日志保存到日志文件中。Java代码中使用JDK logger来发送信息。每一个logger都有一个名字，通常命名为java类的名字，并在日志中打印带着个名字标识的日志。比如"com.caucho.util.ThreadPool"。

通过配置，我们可以设置日志的名称和日志等级。日志等级对所有的log-handler生效，设置之后只有符合等级的日志才会被打印。因此一些handler会收到所有的日志信息，但是logger只会接受info等级的信息。

以下log-handler发送所有日志信息到JVM的控制台输出。它也接受所有info等级的日志和MyBean的finer等级日志。所以控制台输出会包含所有的info和finer日志。

1. **resin.xml配置文件**
2. **default host为host默认配置**

<!-- the default host, matching any host name -->

<host id="" root-directory=".">

<!--webapps can be overridden/extended in the resin.xml-->

<web-app id="/" root-directory="webapps/ROOT"/>

</host>

1. **部署war包**

将项目打包好的war包，放置在webapps下，resin启动时会自动解压

如果想加载war包，在resin.xml配置中的host内配置即可

既修改上文host默认配置中的root-directory，改为war包解压缩后的文件名

1. **修改resin访问端口**

通过对resin.properties文件的修改测试，发现页面的访问端口为app.http属性

默认参数为8080，将app.http修改，就能改变页面访问的端口

app.http : 8082

# app.https : 8443

1. **修改jetty访问端口**

在jetty根目录下，通过对start.ini文件进行修改，页面访问端口属性为：

将属性jetty.http.port属性参数修改，就能改变页面访问的端口

## Connector port to listen on

jetty.http.port=8081

1. **配置DBCP连接池**

DBCP连接池为阿里的连接配置

**具体需要2个依赖包**：

apache.commons.dbcp

apache.commons.pool

**连接池配置为**：

<!-- 使用JDBC数据库连接池 -->

<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource" destroy-method="close">

<!-- destroy-method设置为close使Spring容器关闭同时数据源能够正常关闭，以免造成连接泄露 -->

<property name="driverClassName" value="${jdbc.driver}"/>

<property name="url" value="${jdbc.url}"/>

<property name="username" value="${jdbc.uid}"/>

<property name="password" value="${jdbc.password}"/>

<!-- 初始连接池大小 -->

<property name="initialSize" value="5" /><!-- 在启动连接池初始创建的数据库连接，默认为0 -->

<property name="maxActive" value="15" /><!-- 设置数据库同一时间的最大活跃连接默认为8，负数表示不闲置 -->

<property name="maxIdle" value="10"/><!-- 在连接池空闲时的最大连接数，超过的会被释放，默认为8，负数表示不闲置 -->

<property name="minIdle" value="1" /><!-- 空闲时的最小连接数，低于这个数量会创建新连接，默认为0 -->

</bean>

**明天计划完成的事情**：

明天计划完成Nginx的配置，部署负载均衡和反向代理

有空看看深度思考，完成任务二的学习

**遇到的问题**：

因为没有配置数据库连接池，所以在使用三个服务器同时连接数据库时，导致数据库莫名掉线，无法连接。所以重新部署了数据连接池，想办法让服务器的数据库上线。

**收获**：

学会了jetty、resin、tomcat的部署方法，以及如何修改web项目的加载配置，访问端口

只有实际使用工具时，才能更好的理解工具的作用。

特别是使用工具时遇到了问题，尝试理解工具的配置，构析工具的功能时，才能对工具有更深入的理解，方便以后的使用。

还有就是数据连接池的使用情况，在我服务器因为多个同时连接导致的掉线后，我知道连接池的重要性了。