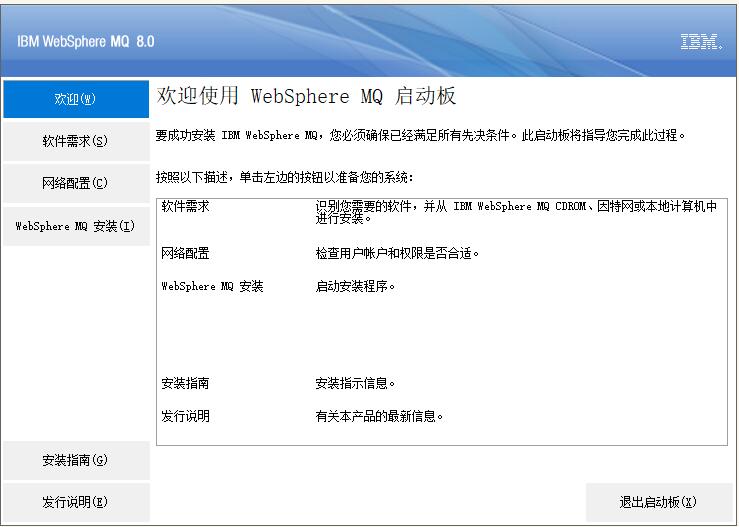
# 1.IMB BQ8.0在windows10下的安装

1)在官方网站<https://developer.ibm.com/messaging/mq-downloads/下载8.0版本的MQ>，解压软件包，双击Setup.exe安装程序，进入WebSphere MQ 启动板



2)点击“软件需求”，能看到你的机器是否符合安装条件（7版本好像装不了win10）



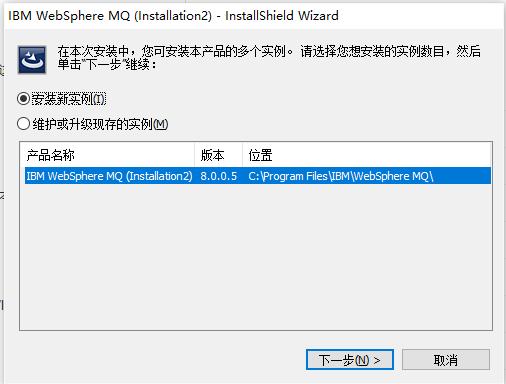
3)下一步“网络配置”，如果需要配置域用户标识，就点击是，由于目前没有用到，这里选择的是“否”；



4)最后一步点击“WebSphere MQ 安装”，启动”IBM WebSphereMQ安装程序”；



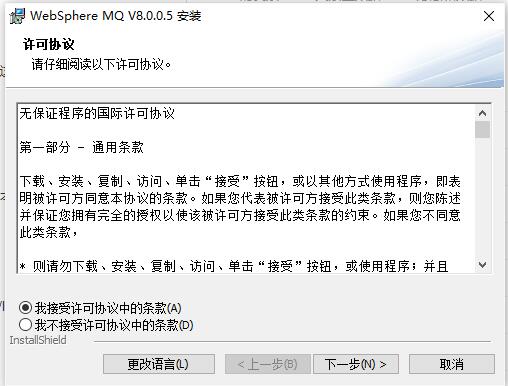
5)选择 “安装新实例”



6)进入安装界面



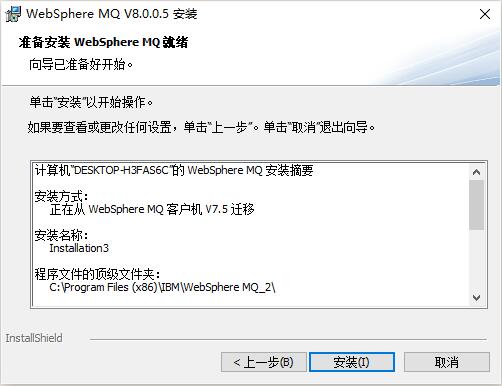
7)接受协议，下一步



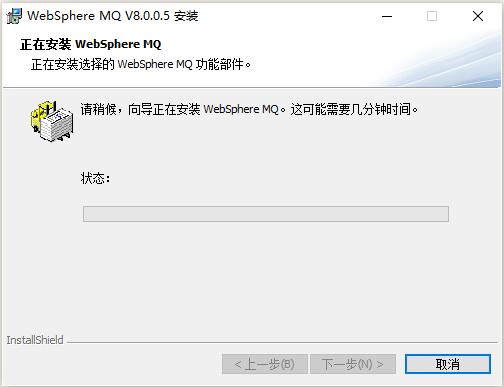
8)安装类型，选择“典型”，下一步（由于我已经安装好一个实例，这里没有“典型选项”）



9)直接点击安装



10)安装过程，可能需要等几分钟



11)完成安装，进入“WebSphere MQ向导”



12)后面直接下一步几连，完成向导安装。

# 2.配置MQ

**统一说明：**

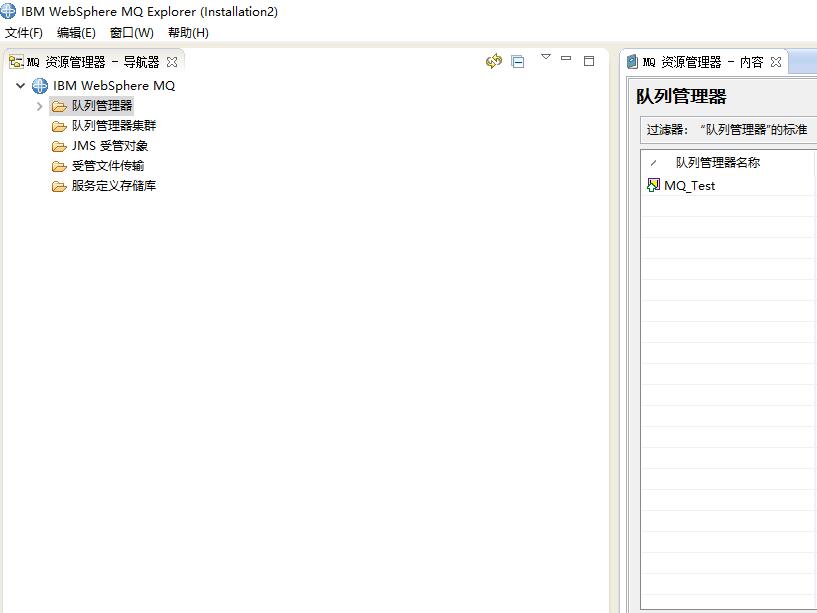
**队列管理器名字为：”MQ\_Test”**

**使用的端口为：”1414”**

**本地队列名称为：”LOCALQUEUE”**

**服务器连接通道名称为：”SERVERCONN”**

1）进入IBM Explorer界面



2）创建队列管理器

点击“队列管理器”->“新建”->“队列管理器”->填写队列管理器名称

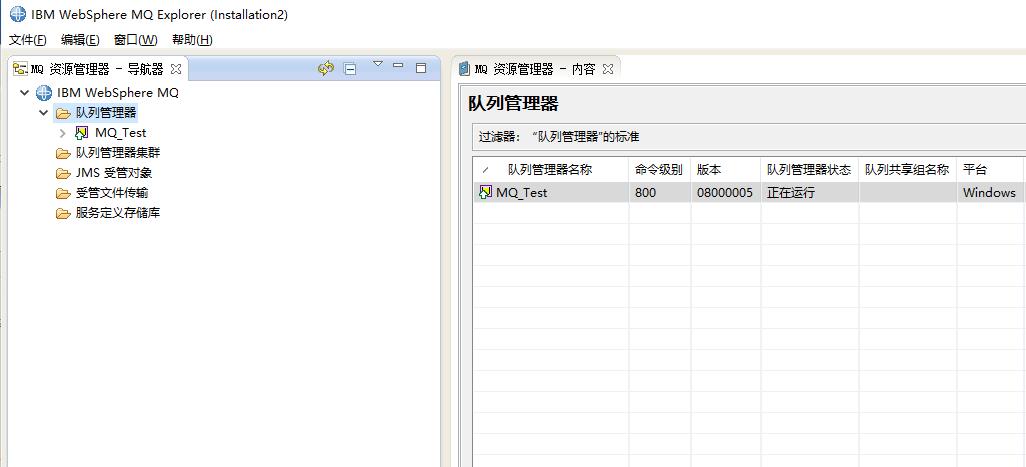
一直点击下一步，直到这个界面：



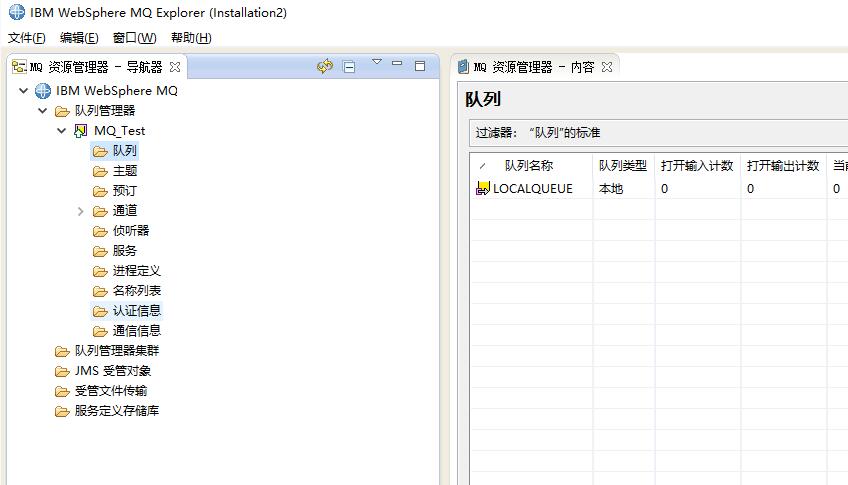


将监听端口设置为1414

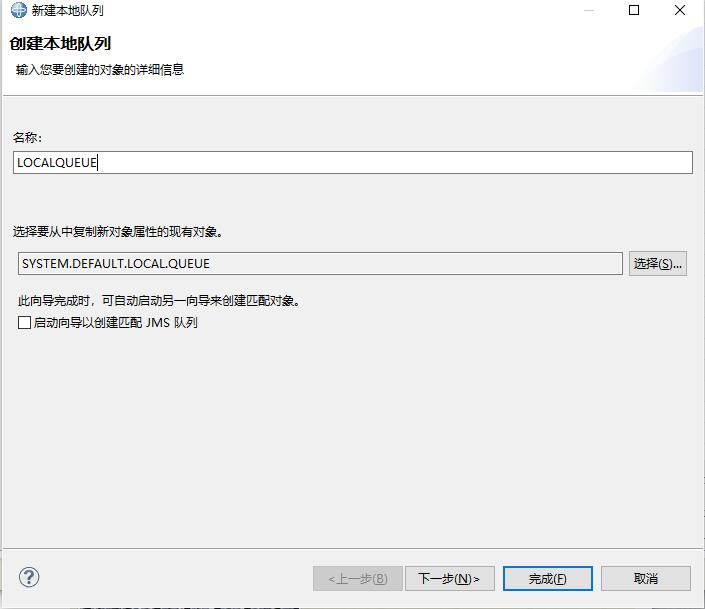
3）队列管理器创建完成



4）展开队列管理器MQ\_Test

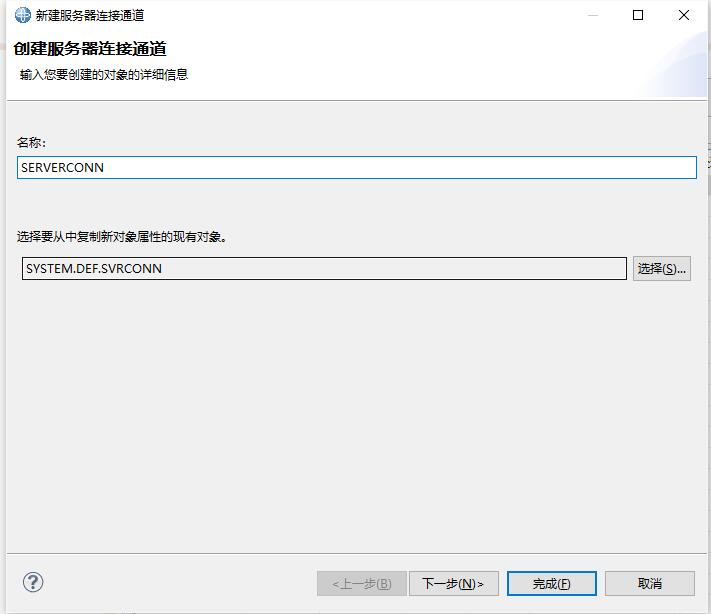


5）点击队列->“新建”->“本地队列”->输入本地队列名称”LOCALQUEUE”->“完成”



6）新建服务器连接通道

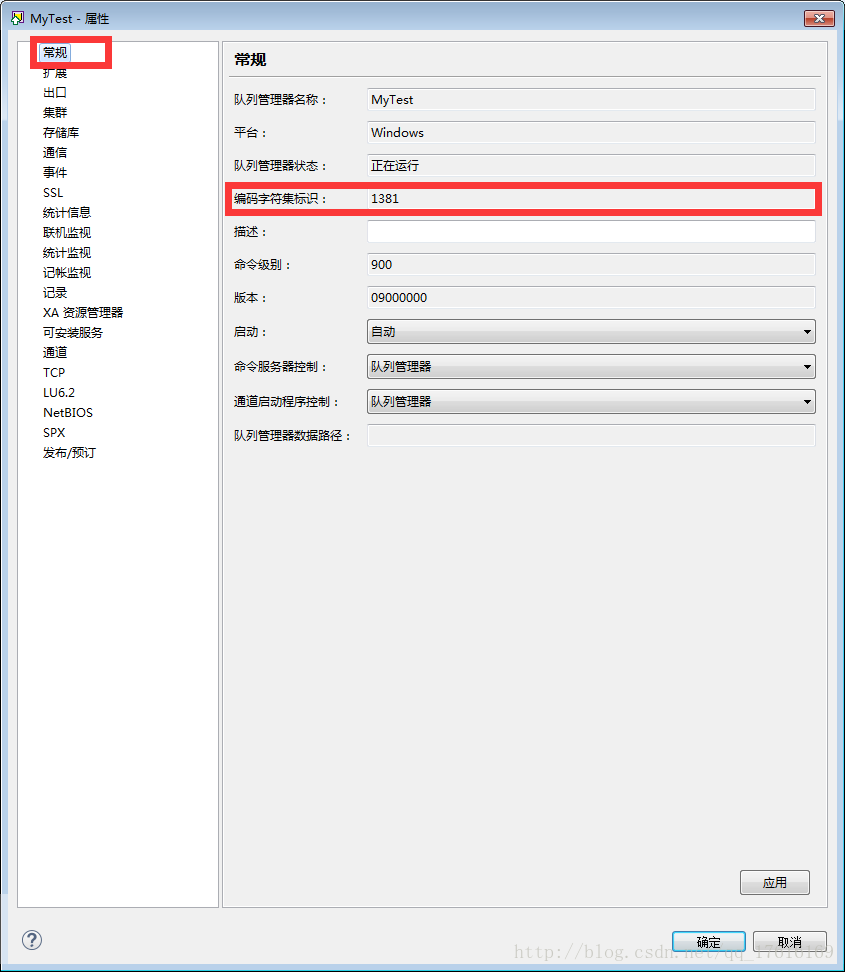
点击“通道”->“服务器连接通道”->输入名称“SERVERCONN”->“完成”



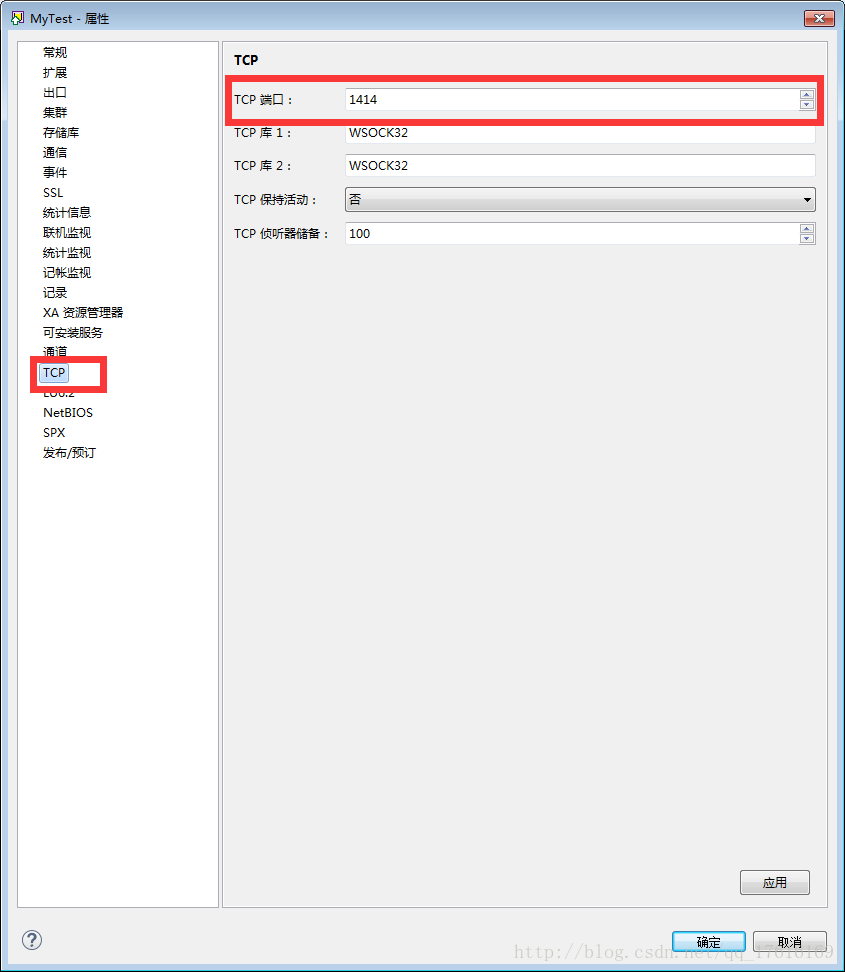
7）确认编码字符集表示和确认端口号

右键“MQ\_Test” -> “属性”，进入MQ\_Test属性界面

在“常规”中确认“编码字符集标识”是否为1381



点击”TCP”确认“TCP端口”是否为1414



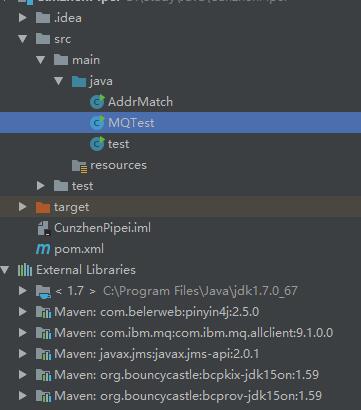
# 3.创建项目

## 3.1项目简介

本项目为Maven结构项目

实现用java连接IBM MQ,并能发送消息到对列和接收消息队列。

## 3.2项目结构



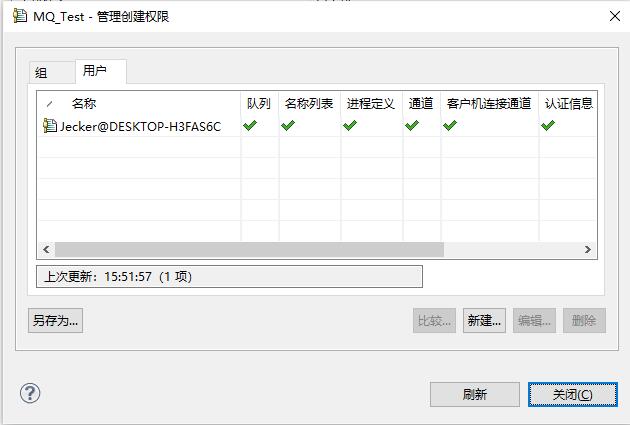
3.2.1 Maven依赖

<dependency>  
 <groupId>com.ibm.mq</groupId>  
 <artifactId>com.ibm.mq.allclient</artifactId>  
 <version>9.1.0.0</version>  
 </dependency>  
</dependencies>

## 3.3MQTest.java

import com.ibm.mq.\*;  
import com.ibm.msg.client.wmq.compat.base.internal.MQC;  
  
public class MQTest {  
 static MQQueueManager *qMgr*;  
 static int *CCSID* = 1381;  
 static String *queueString* = "LOCALQUEUE";  
  
 public static void connect() throws MQException {  
 MQEnvironment.*hostname* = "localhost"; *//队列服务器IP* MQEnvironment.*channel* = "SERVERCONN";*//通道名称* MQEnvironment.*port* = 1414; *//队列管理器端口* MQEnvironment.*CCSID* = *CCSID*; *//编码字符集标识* MQEnvironment.*userID* = "Jecker"; *//访问用户名称* MQEnvironment.*password* = "\*\*\*\*\*\*"; *//访问用户密码  
 qMgr* = new MQQueueManager("MQ\_Test"); *// "MQ\_Test"：队列管理器名字* }  
  
 public static void sendMsg(String msgStr) {  
 int openOptions = MQC.*MQOO\_INPUT\_AS\_Q\_DEF* | MQC.*MQOO\_OUTPUT* | MQC.*MQOO\_INQUIRE*;  
 MQQueue queue = null;  
 try {  
 *//建立Q1通道的连接* queue = *qMgr*.accessQueue(*queueString*, openOptions, null, null, null);  
 MQMessage msg = new MQMessage();  
 msg.format = MQC.*MQFMT\_STRING*;  
 msg.characterSet = *CCSID*;  
 msg.encoding = *CCSID*;  
 *//msg.writeObject(msgStr); //将消息写入消息对象* msg.writeString(msgStr);  
 MQPutMessageOptions pmo = new MQPutMessageOptions();  
 msg.expiry = -1; *//设置消息用不过期* queue.put(msg ,pmo);  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 if (queue != null) {  
 try {  
 queue.close();  
 } catch (MQException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 }  
  
  
 public static void receiveMsg() {  
 int openOptions = MQC.*MQOO\_INPUT\_AS\_Q\_DEF* | MQC.*MQOO\_OUTPUT* | MQC.*MQOO\_INQUIRE*;  
 MQQueue queue = null;  
 try {  
 queue = *qMgr*.accessQueue(*queueString*, openOptions, null, null, null);  
 System.*out*.println("该队列当前的深度为:" + queue.getCurrentDepth());  
 System.*out*.println("===========================");  
 int depth = queue.getCurrentDepth();  
 *//将队列里的消息读出来* while (depth-- > 0){  
 MQMessage msg = new MQMessage(); *//要读取的队列的消息* MQGetMessageOptions gmo = new MQGetMessageOptions();  
 queue.get(msg, gmo);  
 System.*out*.println("消息的大小为：" + msg.getDataLength());  
 System.*out*.println("消息的内容：\n" + msg.readStringOfByteLength(msg.getDataLength()));  
 System.*out*.println("---------------------------");  
 }  
 } catch (Exception e) {  
 e.printStackTrace();  
 } finally {  
 if (queue != null) {  
 try {  
 queue.close();  
 } catch (MQException e) {  
 e.printStackTrace();  
 }  
 }  
 }  
 }  
 public static void main(String[] args) {  
 try {  
 *connect*();  
 *sendMsg*("消息测试");  
 *receiveMsg*();  
 } catch (MQException e) {  
 e.printStackTrace();  
 }  
 }  
}

备注：如果不知道访问用户名，可右键“MQ\_Test”->”对象权限”->”管理对象权限”->用户”，查看该队列管理器开放的访问用户



## 3.4运行结果

