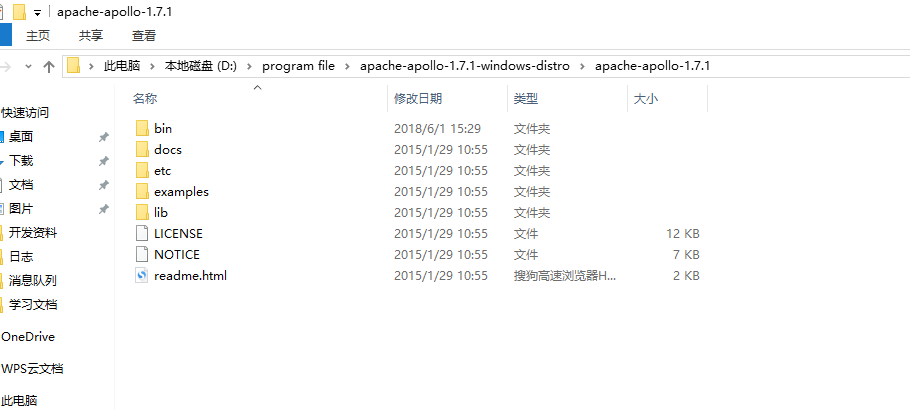
**mqtt服务器搭建**

**一．搭建环境**

首先下载mqtt代理服务器，我们到Apollo官网下载

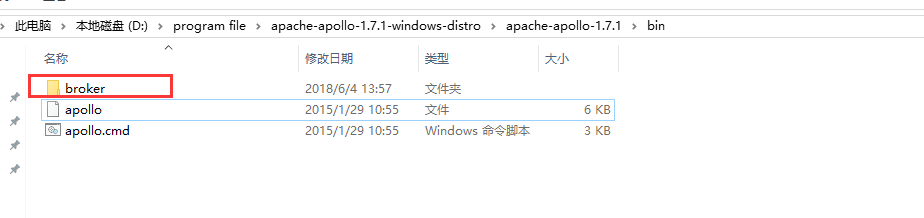
[http://activemq.apache.org/apollo/download.html](http://activemq.apache.org/apollo/download.html" \t "https://blog.csdn.net/oneil2016/article/details/_blank)

下载以后解压缩如下：

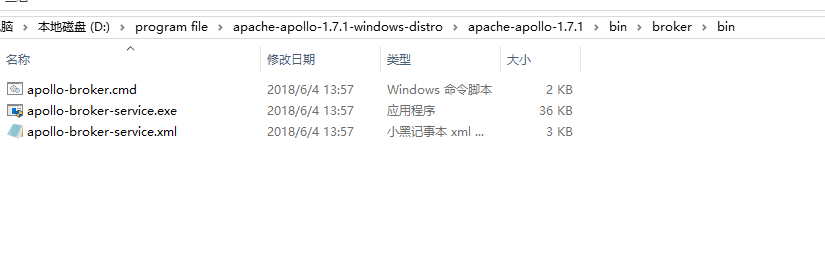


进入到bin 目录下执行命令： apollo.cmd create broker

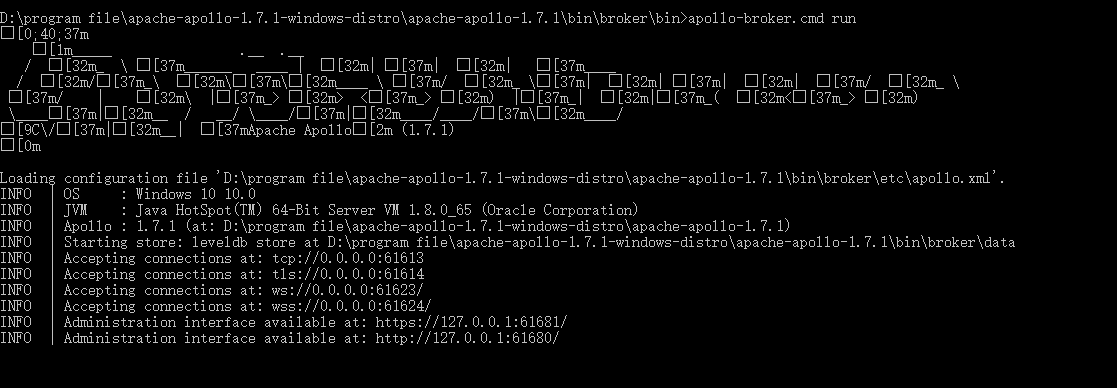
其中create broker 参数表示创建一个代理。执行完命令以后我们发现bin目录下多了一个文件夹



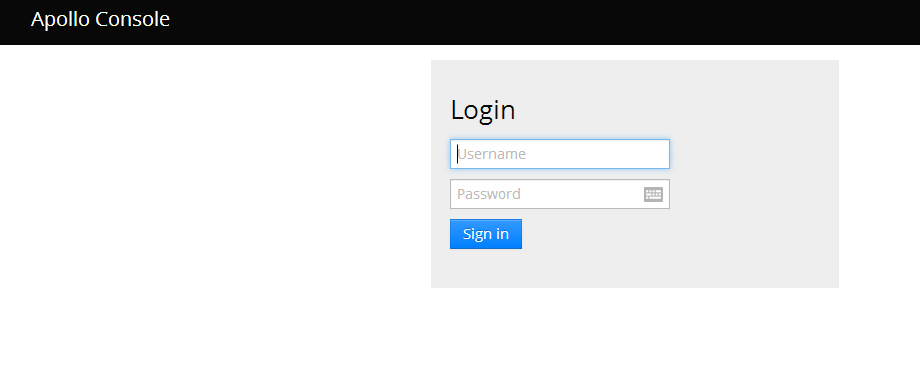
进入broker下面的bin目录



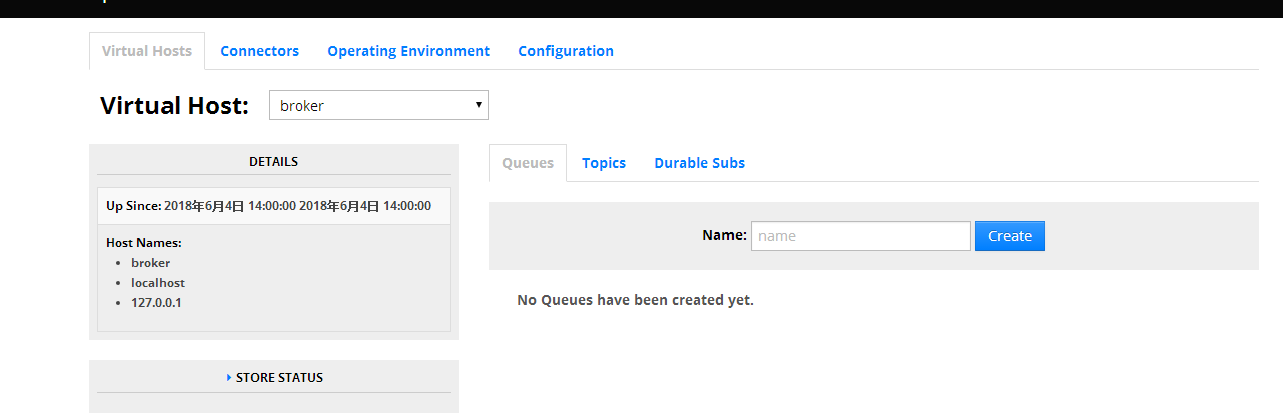
我们启动broker 执行命令：apollo-broker.cmd run



执行结果如上，我们在浏览器中输入http://127.0.0.1:61680/console/index.html 进入登录界面



输入默认用户名admin和密码password，看到如下界面说明我们安装成功。



二.编写测试代码

1.首先引入Jar包

<**dependency**>  
 <**groupId**>org.eclipse.paho</**groupId**>  
 <**artifactId**>org.eclipse.paho.client.mqttv3</**artifactId**>  
 <**version**>1.2.0</**version**>  
</**dependency**>

服务端代码：  
**import** org.eclipse.paho.client.mqttv3.\*;  
**import** org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;  
**import** java.io.UnsupportedEncodingException;  
  
**public class** Server {  
 **public static final** String ***HOST*** = **"tcp://127.0.0.1:61613"**;  
 **public static final** String ***TOPIC*** = **"toclient/124"**;  
 **public static final** String ***TOPIC125*** = **"toclient/125"**;  
 **private static final** String ***clientid*** = **"server"**;  
 **private** MqttClient **client**;  
 **private** MqttTopic **topic**;  
 **private** MqttTopic **topic125**;  
 **private** String **userName** = **"admin"**;  
 **private** String **passWord** = **"password"**;  
 **private** MqttMessage **message**;  
  
 **public** Server() **throws** MqttException {  
 *// MemoryPersistence设置clientid的保存形式，默认为以内存保存* **client** = **new** MqttClient(***HOST***, ***clientid***, **new** MemoryPersistence());  
 connect();  
 }  
  
 **private void** connect() {  
 MqttConnectOptions options = **new** MqttConnectOptions();  
 options.setCleanSession(**false**);  
 options.setUserName(**userName**);  
 options.setPassword(**passWord**.toCharArray());  
 *// 设置超时时间* options.setConnectionTimeout(10);  
 *// 设置会话心跳时间* options.setKeepAliveInterval(20);  
 **try** {  
 **client**.setCallback(**new** PushCallback());  
 **client**.connect(options);  
 **topic** = **client**.getTopic(***TOPIC***);  
 **topic125** = **client**.getTopic(***TOPIC125***);  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
  
 **public void** publish(MqttTopic topic, MqttMessage message) **throws** MqttPersistenceException,  
 MqttException {  
 MqttDeliveryToken token = topic.publish(message);  
 token.waitForCompletion();  
 System.***out***.println(**"message is published completely! "** + token.isComplete());  
 }  
  
 **public static void** main(String[] args) **throws** MqttException, UnsupportedEncodingException {  
  
 Server server = **new** Server();  
 server.**message** = **new** MqttMessage();  
 server.**message**.setQos(2);  
 server.**message**.setRetained(**true**);  
 server.**message**.setPayload(**"给客户端124推送的信息"**.getBytes());  
 server.publish(server.**topic**, server.**message**);  
 server.**message** = **new** MqttMessage();  
 server.**message**.setQos(2);  
 server.**message**.setRetained(**true**);  
 server.**message**.setPayload(**"给客户端125推送的信息"**.getBytes());  
 server.publish(server.**topic125**, server.**message**);  
 System.***out***.println(server.**message**.isRetained() + **"------ratained状态"**);  
 }  
}

回调类：  
**import** org.eclipse.paho.client.mqttv3.IMqttDeliveryToken;  
**import** org.eclipse.paho.client.mqttv3.MqttCallback;  
**import** org.eclipse.paho.client.mqttv3.MqttMessage;  
  
**public class** PushCallback **implements** MqttCallback {  
 **public void** connectionLost(Throwable cause) {  
 *// 连接丢失后，一般在这里面进行重连* System.***out***.println(**"连接断开，可以做重连"**);  
 }  
  
 **public void** deliveryComplete(IMqttDeliveryToken token) {  
*// System.out.println("deliveryComplete---------" + token.isComplete());* }  
  
 **public void** messageArrived(String topic, MqttMessage message) **throws** Exception {  
 *// subscribe后得到的消息会执行到这里面* System.***out***.println(**this**.hashCode() + **"接收消息主题 : "** + topic);  
 System.***out***.println(**this**.hashCode() + **"接收消息Qos : "** + message.getQos());  
 System.***out***.println(**this**.hashCode() + **"接收消息内容 : "** + **new** String(message.getPayload()));  
 }  
}

客户端代码：  
**import** org.eclipse.paho.client.mqttv3.MqttClient;  
**import** org.eclipse.paho.client.mqttv3.MqttConnectOptions;  
**import** org.eclipse.paho.client.mqttv3.MqttException;  
**import** org.eclipse.paho.client.mqttv3.MqttTopic;  
**import** org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;  
  
**public class** Client {  
 **public static final** String ***HOST*** = **"tcp://127.0.0.1:61613"**;  
 **public static final** String ***TOPIC*** = **"toclient/124"**;  
 **public static final** String ***TOPIC125*** = **"toclient/125"**;  
 **private static final** String ***clientid*** = **"client"**;  
 **private** MqttClient **client**;  
 **private** MqttConnectOptions **options**;  
 **private** String **userName** = **"admin"**;  
 **private** String **passWord** = **"password"**;  
  
 **private void** start() {  
 **try** {  
  
 *// host为主机名，clientid即连接MQTT的客户端ID，一般以唯一标识符表示，MemoryPersistence设置clientid的保存形式，默认为以内存保存* **client** = **new** MqttClient(***HOST***, ***clientid***, **new** MemoryPersistence());  
 *// MQTT的连接设置* **options** = **new** MqttConnectOptions();  
 *// 设置是否清空session,这里如果设置为false表示服务器会保留客户端的连接记录，这里设置为true表示每次连接到服务器都以新的身份连接* **options**.setCleanSession(**true**);  
 *// 设置连接的用户名* **options**.setUserName(**userName**);  
 *// 设置连接的密码* **options**.setPassword(**passWord**.toCharArray());  
 *// 设置超时时间 单位为秒* **options**.setConnectionTimeout(10);  
 *// 设置会话心跳时间 单位为秒 服务器会每隔1.5\*20秒的时间向客户端发送个消息判断客户端是否在线，但这个方法并没有重连的机制* **options**.setKeepAliveInterval(20);  
 *// 设置回调* **client**.setCallback(**new** PushCallback());  
 MqttTopic topic = **client**.getTopic(***TOPIC***);  
 *//setWill方法，如果项目中需要知道客户端是否掉线可以调用该方法。设置最终端口的通知消息* **options**.setWill(topic, **"close"**.getBytes(), 2, **true**);  
  
 **client**.connect(**options**);  
 *//订阅消息* **int**[] Qos = {1, 1};  
 String[] topic1 = {***TOPIC***, ***TOPIC125***};  
 **client**.subscribe(topic1, Qos);  
  
 } **catch** (Exception e) {  
 e.printStackTrace();  
 }  
 }  
  
 **public static void** main(String[] args) **throws** MqttException {  
 Client client = **new** Client();  
 client.start();  
 }  
}

**说明** ：我们连接broker的代码中都有一个clienId,这个clientId是标明客户身份的字符串，如果两个客户端都使用同一个clientId去连接broker,那么后面的那个会将前面的那个连接挤掉线，即broker会直接断掉与前一个clientId的连接，再与后面的一个clientId建立连接。