**使用Apache Curator实现服务的注册和发现**

原创 2016年07月01日 17:24:14

* 标签：
* [curator](http://so.csdn.net/so/search/s.do?q=curator&t=blog) /
* [服务的注册和发现](http://so.csdn.net/so/search/s.do?q=%E6%9C%8D%E5%8A%A1%E7%9A%84%E6%B3%A8%E5%86%8C%E5%92%8C%E5%8F%91%E7%8E%B0&t=blog)
* 1966

使用zookeeper可以实现服务的注册和发现，而Curator是对zookeeper进行的一层封装，自然也封装了一套实现服务的注册和发现，本文就介绍如何使用Curator实现服务的注册和发现

首先要安装zookeeper，我这里安装的是：zookeeper-3.4.6

curator 的依赖版本如下：

**[html]** [view plain](http://blog.csdn.net/mn960mn/article/details/51803128) [copy](http://blog.csdn.net/mn960mn/article/details/51803128)

1. **<dependency>**
2. **<groupId>**org.apache.curator**</groupId>**
3. **<artifactId>**curator-x-discovery-server**</artifactId>**
4. **<version>**2.9.1**</version>**
5. **</dependency>**

做服务的注册和发现，自然要有服务端和客户端

服务端代码如下：

服务端一：

**[java]** [view plain](http://blog.csdn.net/mn960mn/article/details/51803128) [copy](http://blog.csdn.net/mn960mn/article/details/51803128)

1. **package** com.mall.zk.service;
3. /\*\*
4. \* 服务的附加信息
5. \*/
6. **public** **class** ServiceDetail {
7. //服务注册的根路径
8. **public** **static** **final** String REGISTER\_ROOT\_PATH = "/mall";
10. **private** String desc;
11. **private** **int** weight;
13. **public** ServiceDetail() {}
14. **public** ServiceDetail(String desc, **int** weight) {
15. **this**.desc = desc;
16. **this**.weight = weight;
17. }
18. **public** String getDesc() {
19. **return** desc;
20. }
21. **public** **void** setDesc(String desc) {
22. **this**.desc = desc;
23. }
24. **public** **int** getWeight() {
25. **return** weight;
26. }
27. **public** **void** setWeight(**int** weight) {
28. **this**.weight = weight;
29. }
30. @Override
31. **public** String toString() {
32. **return** "ServiceDetail [desc=" + desc + ", weight=" + weight + "]";
33. }
34. }

**[java]** [view plain](http://blog.csdn.net/mn960mn/article/details/51803128) [copy](http://blog.csdn.net/mn960mn/article/details/51803128)

1. **package** com.mall.zk.server;
3. **import** java.util.concurrent.TimeUnit;
5. **import** org.apache.curator.framework.CuratorFramework;
6. **import** org.apache.curator.framework.CuratorFrameworkFactory;
7. **import** org.apache.curator.retry.ExponentialBackoffRetry;
8. **import** org.apache.curator.x.discovery.ServiceDiscovery;
9. **import** org.apache.curator.x.discovery.ServiceDiscoveryBuilder;
10. **import** org.apache.curator.x.discovery.ServiceInstance;
11. **import** org.apache.curator.x.discovery.ServiceInstanceBuilder;
12. **import** org.apache.curator.x.discovery.details.JsonInstanceSerializer;
14. **import** com.mall.zk.service.ServiceDetail;
16. **public** **class** AppServer {
17. **public** **static** **void** main(String[] args) **throws** Exception {
18. CuratorFramework client = CuratorFrameworkFactory.newClient("192.168.1.101:2181", **new** ExponentialBackoffRetry(1000, 3));
19. client.start();
20. client.blockUntilConnected();
22. /\*\*
23. \* 指定服务的 地址，端口，名称
24. \*/
25. ServiceInstanceBuilder<ServiceDetail> sib = ServiceInstance.builder();
26. sib.address("192.168.1.100");
27. sib.port(8855);
28. sib.name("tomcat");
29. sib.payload(**new** ServiceDetail("主站web程序", 1));
31. ServiceInstance<ServiceDetail> instance = sib.build();
33. ServiceDiscovery<ServiceDetail> serviceDiscovery = ServiceDiscoveryBuilder.builder(ServiceDetail.**class**)
34. .client(client)
35. .serializer(**new** JsonInstanceSerializer<ServiceDetail>(ServiceDetail.**class**))
36. .basePath(ServiceDetail.REGISTER\_ROOT\_PATH)
37. .build();
38. //服务注册
39. serviceDiscovery.registerService(instance);
40. serviceDiscovery.start();
42. TimeUnit.SECONDS.sleep(70);
44. serviceDiscovery.close();
45. client.close();
46. }
47. }

服务端二：

**[java]** [view plain](http://blog.csdn.net/mn960mn/article/details/51803128) [copy](http://blog.csdn.net/mn960mn/article/details/51803128)

1. **package** com.mall.zk.server;
3. **import** java.util.concurrent.TimeUnit;
5. **import** org.apache.curator.framework.CuratorFramework;
6. **import** org.apache.curator.framework.CuratorFrameworkFactory;
7. **import** org.apache.curator.retry.ExponentialBackoffRetry;
8. **import** org.apache.curator.x.discovery.ServiceDiscovery;
9. **import** org.apache.curator.x.discovery.ServiceDiscoveryBuilder;
10. **import** org.apache.curator.x.discovery.ServiceInstance;
11. **import** org.apache.curator.x.discovery.ServiceInstanceBuilder;
12. **import** org.apache.curator.x.discovery.details.JsonInstanceSerializer;
14. **import** com.mall.zk.service.ServiceDetail;
16. **public** **class** AppServer2 {
17. **public** **static** **void** main(String[] args) **throws** Exception {
18. CuratorFramework client = CuratorFrameworkFactory.newClient("192.168.1.101:2181", **new** ExponentialBackoffRetry(1000, 3));
19. client.start();
20. client.blockUntilConnected();
22. /\*\*
23. \* 指定服务的 地址，端口，名称
24. \*/
25. ServiceInstanceBuilder<ServiceDetail> sib = ServiceInstance.builder();
26. sib.address("192.168.1.100");
27. sib.port(8866);
28. sib.name("tomcat");
29. sib.payload(**new** ServiceDetail("主站web程序", 2));
31. ServiceInstance<ServiceDetail> instance = sib.build();
33. ServiceDiscovery<ServiceDetail> serviceDiscovery = ServiceDiscoveryBuilder.builder(ServiceDetail.**class**)
34. .client(client)
35. .serializer(**new** JsonInstanceSerializer<ServiceDetail>(ServiceDetail.**class**))
36. .basePath(ServiceDetail.REGISTER\_ROOT\_PATH)
37. .build();
38. //服务注册
39. serviceDiscovery.registerService(instance);
40. serviceDiscovery.start();
42. TimeUnit.SECONDS.sleep(70);
44. serviceDiscovery.close();
45. client.close();
46. }
47. }

上面两个服务端代码，几乎一样，只是服务端口，权重不一样。然后运行上面两个main方法

然后，去zk里面查看

[zk: localhost:2181(CONNECTED) 54] ls /mall/tomcat  
[a8657594-ef32-4efc-b2b0-3077d105e486, b5b55bbc-ce04-4b70-9413-1ec3c54023b8]  
可以看到，有2个临时节点，也就意味着有两个服务

客户端代码：

**[java]** [view plain](http://blog.csdn.net/mn960mn/article/details/51803128) [copy](http://blog.csdn.net/mn960mn/article/details/51803128)

1. **package** com.mall.zk.client;
3. **import** java.util.Collection;
5. **import** org.apache.curator.framework.CuratorFramework;
6. **import** org.apache.curator.framework.CuratorFrameworkFactory;
7. **import** org.apache.curator.retry.ExponentialBackoffRetry;
8. **import** org.apache.curator.x.discovery.ServiceDiscovery;
9. **import** org.apache.curator.x.discovery.ServiceDiscoveryBuilder;
10. **import** org.apache.curator.x.discovery.ServiceInstance;
12. **import** com.mall.zk.service.ServiceDetail;
14. **public** **class** AppClient {
15. **public** **static** **void** main(String[] args) **throws** Exception{
16. CuratorFramework client = CuratorFrameworkFactory.newClient("192.168.1.101:2181", **new** ExponentialBackoffRetry(1000, 3));
17. client.start();
18. client.blockUntilConnected();
20. ServiceDiscovery<ServiceDetail> serviceDiscovery = ServiceDiscoveryBuilder.builder(ServiceDetail.**class**)
21. .client(client)
22. .basePath(ServiceDetail.REGISTER\_ROOT\_PATH)
23. .build();
24. serviceDiscovery.start();
26. //根据名称获取服务
27. Collection<ServiceInstance<ServiceDetail>> services = serviceDiscovery.queryForInstances("tomcat");
28. **for**(ServiceInstance<ServiceDetail> service : services) {
29. System.out.println(service.getPayload());
30. System.out.println(service.getAddress() + "\t" + service.getPort());
31. System.out.println("---------------------");
32. }
34. serviceDiscovery.close();
35. client.close();
36. }
37. }

运行客户端，输出：

**[plain]** [view plain](http://blog.csdn.net/mn960mn/article/details/51803128) [copy](http://blog.csdn.net/mn960mn/article/details/51803128)

1. ServiceDetail [desc=主站web程序, weight=2]
2. 192.168.1.100   8866
3. ---------------------
4. ServiceDetail [desc=主站web程序, weight=1]
5. 192.168.1.100   8855
6. ---------------------

版权声明：本文为博主原创文章，未经博主允许不得转载。