https://www.cnblogs.com/mushroom/p/4435997.html

[Redis系列(二)-Hredis客户端设计及开源](https://www.cnblogs.com/mushroom/p/4435997.html)

接上篇[c#实现redis客户端(一)](http://www.cnblogs.com/mushroom/p/4217541.html)，重新整理些了下。

**阅读目录：**

1. [项目说明](https://www.cnblogs.com/mushroom/p/4435997.html#three)
2. [Hredis设计图](https://www.cnblogs.com/mushroom/p/4435997.html#two)
3. [单元测试场景](https://www.cnblogs.com/mushroom/p/4435997.html#one)
4. [总结](https://www.cnblogs.com/mushroom/p/4435997.html#four)

**项目说明**

**背景**：因为有地方要用，而又没找到对sentinel良好支持的Net客户端，所以就简单重写了个。

**目标：**尽可能的简单，轻量级，不进行过度的封装，使用方便。

**代码说明：**

        1. 与Redis服务端的Socket通信、协议格式封装。在RedisBaseClient里

        2. 只对Set、Get封装，暴露出Send接口。在RedisCommand里面添加自己想要的支持。

var info = rcClient.Send(RedisCommand.INFO);

       3.  RedisBaseClient是通信层。 如果扩展其他用途继承即可，比如RedisPubSub:RedisBaseClient，RedisSentinel:RedisBaseClient

       4.  供上层良好调用的话，可以做成partial类扩展redisclient。比如RedisClient.String

       5.  订阅的监听使用while的，可在触发事件里面做阻塞。

       6.  PoolRedisClient池的实现使用ConcurrentStack，仅达到了复用socket连接的目的。

       7.  支持socket重连，做法是关闭旧连接，重新建立新socket。

       8.  多个命令使用管道实现。见set实现

**后续思路：**

  一：PoolRedisClient池里面连接的释放问题？

           1. 使用using

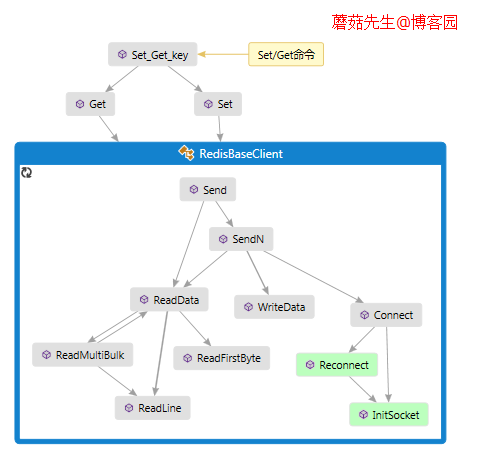
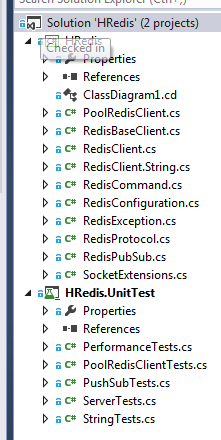
           2. 不用使用using，会自动检测并回收 。

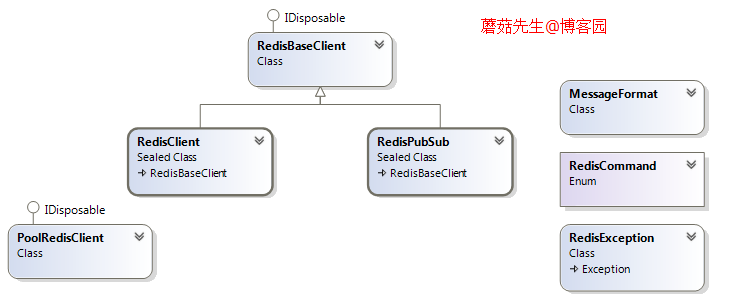
           不做成自动检测的话，就会出现连接无法释放的问题，总会有人忘记释放的，所以要优化成1+2结合的方式。

     二：client池和socket池分离，socket单独做一个池？ 还在考虑中。

**Hredis设计图**

命令执行流程图、解决方案图、类图。

**  **



**单元测试场景**

**一. Info命令通信、密码配置。**

[复制代码](javascript:void(0);)

[TestMethod, TestCategory("Server")]

public void Redis\_PassWord()

{

using (var rcClient = new RedisClient(new RedisConfiguration()

{

Host = ip,

Port = 6381,

PassWord = "123465"

}))

{

var info = rcClient.Send(RedisCommand.INFO);

Debug.Write(info.ToString());

}

}

[复制代码](javascript:void(0);)

**二. 普通订阅，及模式匹配订阅。这里订阅的是Sentinel事件信息。**

[复制代码](javascript:void(0);)

[TestMethod, TestCategory("PushSub")]

public void Subscribe\_Sentinel\_Test()

{

using (RedisPubSub rsc = new RedisPubSub("127.0.0.1", 20001))

{

rsc.SubscriptionReceived += rsc\_SubscriptionReceived;

//rsc.Subscribe("+sdown");

}

}

[复制代码](javascript:void(0);)

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif View Code

**三.  client连接池、Parallel并发模拟。**

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif View Code

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif View Code

**三.  超时重连、多线程并发超时重连。**

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif View Code

[复制代码](javascript:void(0);)

[TestMethod, TestCategory("poolRedisclient")]

public void Thread\_PoolClient\_Test()

{

PoolRedisClient prc = new PoolRedisClient(new PoolConfiguration()

{

Host = ip,

Port = port

});

Parallel.For(0, 1000, new ParallelOptions() {MaxDegreeOfParallelism = 100}, (index, item) =>

{

var t = new Thread(() =>

{

Thread.Sleep(1000);

object info2;

using (var client = prc.GetClient())

{

client.Set("Parallel\_PoolClient\_Test" + index, "Parallel\_PoolClient\_Test");

Thread.Sleep(15000);

info2 = client.Get("Parallel\_PoolClient\_Test" + index);

}

Assert.AreEqual(info2.ToString(), "Parallel\_PoolClient\_Test");

});

t.Start();

});

Thread.Sleep(20000);

prc.Dispose();

}

[复制代码](javascript:void(0);)

**四. String类型添加、过期时间添加。**

[复制代码](javascript:void(0);)

[TestMethod, TestCategory("String")]

public void Set\_Get\_key()

{

using (var rcClient = new RedisClient(ip, port))

{

rcClient.Set("Set\_Get\_key", "Set\_Get\_key");

var info2 = rcClient.Get("Set\_Get\_key");

Assert.AreEqual(info2.ToString(), "Set\_Get\_key");

}

}

[复制代码](javascript:void(0);)

https://images.cnblogs.com/OutliningIndicators/ContractedBlock.gif View Code

**总结**

开源地址：<https://github.com/mushroomsir/HRedis>  有需要的同学，可以参考下。

Hredis后续会跟实际需求来写，如果有更好的实现思路，欢迎一起交流。