1.JMeter Java Sampler介绍  
　　setupTest做些初始化的工作，每个线程只执行一次  
　　teardownTest做些清理工作，每个线程只执行一次  
　　runTest具体的测试执行工作，每个并发每次循环都将执行一次  
　　SampleResult记录测试结果，result.sampleStart()一个事务开始，result.sampleEnd()一个事务结束  
　　main方法用于调试  
01  package  com . xxx . yyy . perf;  
02  
03  import  org.apache.jmeter.config.Argument;  
04  import  org.apache.jmeter.config.Arguments;  
05  import  org.apache.jmeter.protocol.java.sampler.AbstractJavaSamplerClient;  
06  import  org.apache.jmeter.protocol.java.sampler.JavaSamplerContext;  
07  import  org.apache.jmeter.samplers.SampleResult;  
08  import  org.springframework.context.ApplicationContext;  
09  import  org.springframework.context.support.ClassPathXmlApplicationContext;  
10  
11  import  com.xxx.udb.client.PersonService;  
12  import  com.xxx.udb.client.result.PersonResult;  
13  
14  /\*\*  
15  \*  
16  \* @author flynewton  
17  \*/  
18  public  class  GetPersonByLongId  extends  AbstractJavaSamplerClient  {  
19      private  static  final  ApplicationContext  ctx  =  new ClassPathXmlApplicationContext(  
20              "spring-udb.xml");  
21      private  static  PersonService  personService  =  null;  
22      private  static  final  String  loginId  =  "00000sb";  
23      private  static  final  String  siteId  =  "CN";  
24  
25  
26      @Override  
27      public  void  setupTest( JavaSamplerContext  context)  {  
28          super . setupTest( context);  
29          personService  = ( PersonService)  ctx . getBean( "personServiceClient");  
30      }  
31  
32      @Override  
33      public  void  teardownTest( JavaSamplerContext  context)  {  
34          super . teardownTest( context);  
35      }  
36  
37      public  SampleResult  runTest( JavaSamplerContext  arg0)  {  
38          SampleResult  result  =  new  SampleResult();  
39          PersonResult  personResult  =  null;  
40  
41          result . setSampleLabel( "result");  
42  
43          try  {  
44              result . sampleStart();  
45              personResult  =  personService . getPersonByLongId( siteId ，  loginId);  
46              result . sampleEnd();  
47          }  catch ( Throwable  t)  {  
48              this . getLogger (). error( "Exception:"  +  t);  
49              return  null;  
50          }  
51  
52          if ( personResult  !=  null  &&  personResult . getCode()  ==  0)  {  
53              result . setSamplerData( personResult . toString());  
54              result . setSuccessful( true);  
55          }  else  {  
56              result . setSuccessful( false);  
57          }  
58  
59          return  result;  
60      }  
61  
62      static  void  printResult( SampleResult  res)  {  
63          System . out . println( "test is success:"  +  res . isSuccessful()  +  " used:"  
64                  + ( res . getEndTime()  -  res . getStartTime())  +  "ms "  +  " result:"  
65                  +  res . getSampleLabel()  +  ":"  +  res . getSamplerData());  
66      }  
67  
68      public  static  void  main( String []  args)  {  
69          GetPersonByLongId  service  =  new  GetPersonByLongId();  
70          JavaSamplerContext  context  =  new  JavaSamplerContext( null);  
71          service . setupTest( context);  
72          SampleResult  res1  =  service . runTest( context);  
73          printResult( res1);  
74          service . teardownTest( context);  
75          System . exit( 0);  
76      }  
77  
78  }

　　2.多接口性能测试  
　　按照上面的方法进行测试，每个需要测试的接口和场景都需要写一个这样的测试类，对于有些具有相似初始化，清理等  
　　工作，只有具体的那行事务代码不一样的情况，会有很多重复的工作。如下：  
　　可以考虑采用反射的方式来解决这个问题，尤其是业务非常类似的接口和场景。当然，反射会带来压力机的性能消耗，  
　　但是这个可以通过调整JMeter的JVM参数和增加JMeter实例来解决。  
　　1)利用反射后的结构如下：  
　　2)AbstractServiceClient是一个抽象基类  
package  com . xxx . yyy . perf . base;  
import  java.lang.reflect.Method;  
import  org.apache.jmeter.protocol.java.sampler.AbstractJavaSamplerClient;  
import  org.apache.jmeter.protocol.java.sampler.JavaSamplerContext;  
import  org.apache.jmeter.samplers.SampleResult;  
/\*\*  
\* Comment of AbstractServiceClient  
\*  @author flynewton  
\*/  
public  abstract  class  AbstractServiceClient  extends  AbstractJavaSamplerClient {  
public  Object  invokeTest( String  testName ，  JavaSamplerContext  context ，SampleResult  sample)  {  
Method []  methods  =  this . getClass (). getMethods();  
for ( Method  m  :  methods)  {  
if ( m . getName (). equalsIgnoreCase( testName))  {  
try  {  
return  m . invoke( this ，  context ， sample);  
}  catch ( Throwable  t)  {  
this . getLogger (). error( "execute method:"  +  testName  +  " falied" ，  t);  
}  
}  
}  
return  null;  
}  
}

　　3)GetPersonTest为某一类具有相似业务场景的测试类  
　　其中 getDefaultParameters方法设置默认参数：test传入具体要执行的测试方法；info传入日志级别，利用main方法进行调试时设置为true，真正进行性能测试的时候设置为false  
package  com . xxx . yyy . perf;  
import  org.apache.jmeter.config.Argument;  
import  org.apache.jmeter.config.Arguments;  
import  org.apache.jmeter.protocol.java.sampler.JavaSamplerContext;  
import  org.apache.jmeter.samplers.SampleResult;  
import  org.springframework.context.ApplicationContext;  
import  org.springframework.context.support.ClassPathXmlApplicationContext;  
import  com.xxx.udb.client.PersonService;  
import  com.xxx.udb.client.result.MultiPersonResult;  
import  com.xxx.udb.client.result.PersonResult;  
import  com.xxx.udb.perf.base.AbstractServiceClient;  
/\*\*  
\* Test get person  
\* @author flynewton  
\*/  
public  class  GetPersonTest  extends  AbstractServiceClient  {  
private  static  final  ApplicationContext  ctx  =  new ClassPathXmlApplicationContext(  
"spring-udb.xml");  
private  static  PersonService  personService  =  null;  
private  static  String  loginIdCache  =  "00000sb";  
private  static  String  loginIdWithoutCache  =  "01haytham";  
private  static  String  siteIdCache  =  "CN";  
private  static  String  siteIdWithoutCache  =  "IN";  
private  static  String  alimailCn  =  "gchg2008@zzgcchen.cn";  
private  static  String  alimailUdb  =  "000001@lvcuinm.com.cn";  
private  static  String  uidCache  =  "300113578";  
private  static  String  uidWithoutCache  =  "in5002997";  
private  static  String  emailCache  =  "00000sb@alibaba-test.com";  
private  static  String  emailWithoutCache  =  "5002997@alibaba-inc.com";  
@Override  
public  void  setupTest( JavaSamplerContext  context)  {  
super . setupTest( context);  
personService  = ( PersonService)  ctx . getBean( "personServiceClient");  
}  
public  Arguments  getDefaultParameters()  {  
Arguments  params  =  new  Arguments();  
params . addArgument( "test" ，  "testMethod");  
params . addArgument( "info" ，  "false");  
return  params;  
}  
@Override  
public  void  teardownTest( JavaSamplerContext  context)  {  
super . teardownTest( context);  
}  
public  Object  testGetPersonByLongIdViaCache( JavaSamplerContext  arg0 ，  
SampleResult  sample)  {  
PersonResult  personResult  =  null;  
try  {  
sample . sampleStart();  
personResult  =  personService . getPersonByLongId( siteIdCache ， loginIdCache);  
sample . sampleEnd();  
}  catch ( Throwable  t)  {  
this . getLogger (). error( "Exception:"  +  t);  
return  null;  
}  
return  personResult;  
}  
public  Object  testGetPersonByLongIdWithoutCache( JavaSamplerContext  arg0 ，  
SampleResult  sample)  {  
PersonResult  personResult  =  null;  
try  {  
sample . sampleStart();  
personResult  =  personService . getPersonByLongId( siteIdWithoutCache ， loginIdWithoutCache);  
sample . sampleEnd();  
}  catch ( Throwable  t)  {  
this . getLogger (). error( "Exception:"  +  t);  
return  null;  
}  
return  personResult;  
}  
public  Object  testGetPersonByAlimailCN( JavaSamplerContext  arg0 ，  
SampleResult  sample)  {  
PersonResult  personResult  =  null;  
try  {  
sample . sampleStart();