Java编程测试1M内存可用来缓存多少对象 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/144/2021_2022_Java_E7_BC_96_E7_A8_8B_c104_144737.htm 为了提高系统的响应性能，一般都会采用缓存技术来实现，如果用象ehcache、oscache这样的开源的cache工具来实现，一般都需要由开发人员来设置maxElementsInMemory这个值，但这个值在设置的时候大家都是怎么去设置的呢？凭想像还是随便写一个值呢？这个值设的过大嘛有可能会造成outofmemory，设的过小嘛又浪费服务器巨大的内存，为了能够更好的设置这个值，我写了个测试程序来估算1M内存能够缓存多少个对象，代码如下： public void testSpike(){ print("最大的内存为：" Runtime.getRuntime().maxMemory()/1024). print("总的内存为：" Runtime.getRuntime().totalMemory()/1024). print("==========="). long currMemory=Runtime.getRuntime().freeMemory(). print("目前可用的内存为：" currMemory/1024). print("==========="). Map cache=new HashMap(). for (int i = 0. i MockBean bean=new MockBean(). bean.setId(i). bean.setName("jerry" i). bean.setValue(i "jerry"). cache.put(String.valueOf(i), bean). long tempMemory=Runtime.getRuntime().freeMemory(). if((currMemory-tempMemory)/1024==1024){ print("此时可用的内存为：" tempMemory/1024). print("此时缓存了：" i "个对象"). break. } } print("===========").

cache.clear(). long tempMemory=Runtime.getRuntime().freeMemory(). print("目前可用的内存为：" tempMemory/1024). print("消耗的内存为：" (currMemory-tempMemory)/1024). print("==========================================="). Runtime.getRuntime().gc(). tempMemory=Runtime.getRuntime().freeMemory(). print("目前可用的内存为：" tempMemory/1024). print("消耗的内存为：" (currMemory-tempMemory)/1024). } private void print(String msg){ System.out.println(msg). } 在我机器上运行的结果是1M内存可缓存大概4479个对象，同时可以看到，在cache.clear后内存并没有变化，因为gc是没那么及时的，这个时候显式的调用gc则会发现可用的内存量甚至比最初都多。 当然，这里只是个简单的测试，这里测试的也只是缓存一个非常简单的bean对象，缓存的对象消耗的内存大小还需要根据这个对象中具体的内容而定，比如当缓存的是blob类型的字段的时候，可想而知，这个时候消耗的内存量绝对是不同的。 这里只是建议大家在对系统性能做优化时最好根据需要缓存的内容做一个估算，设置好应用所需要的jvm的内存值，以便充分利用服务器的硬件资源。 