```
HotSpot初始化阶段在src\share\vm\runtime\virtualspace.cpp的如下代码中生成ReservedSpace
C++代码  收藏代码
ReservedSpace::ReservedSpace(size_t size, size_t alignment,
                             bool large,
                             bool executable) {
  initialize(size, alignment, large, NULL, 0, executable);
}

这种方式生成的地址是随机的，导致加上如下三个参数时
Java代码  收藏代码
+PrintStubCode
+PrintAssembly
+PrintInterpreter

生成的汇编代码的地址也是随机的，不方便调试，
后来发现只要把前面的代码改成如下:
C++代码  收藏代码
ReservedSpace::ReservedSpace(size_t size, size_t alignment,
                             bool large,
                             bool executable) {
  //0x01cd0000是requested_address
  initialize(size, alignment, large, (char *)0x01cd0000, 0, executable);
}

这样生成的汇编代码的地址每次都一样了，
比如StubRoutines::call_stub的汇编代码固定如下:
Java代码  收藏代码
//...
;; loop:
0x01cd0425: mov    -0x4(%edx,%ecx,4),%eax
0x01cd0429: mov    %eax,(%esp,%ebx,4)
0x01cd042c: inc    %ebx
0x01cd042d: dec    %ecx
0x01cd042e: jne    0x01cd0425
;; parameters_done:
0x01cd0430: mov    0x14(%ebp),%ebx
0x01cd0433: mov    0x18(%ebp),%eax
0x01cd0436: mov    %esp,%esi
;; call Java function
0x01cd0438: call   *%eax
//...
;; call_stub_return_address:


调试时，只要按地址打断点就能精确地到达你想要的那条汇编，
比如0x01cd0438对应的肯定是call   *%eax，从这里转到method entry point的汇编。

只不过有一点点缺陷，我用的是windows，
发现没有正确释放requested_address 对应的空间，
导致不是每次重启HotSpot时都成功，多试几次又好了。

我甚至在ReservedSpace::initialize中加了如下代码:
Java代码  收藏代码
void initialize(size_t size, size_t alignment, bool large,
                  char* requested_address,
                  const size_t noaccess_prefix,
                  bool executable) {
  //...
    if (requested_address != 0) {
      unsigned int count = 10;

      while(count) {
        base = os::attempt_reserve_memory_at(size, requested_address);
        if (failed_to_reserve_as_requested(base, requested_address, size, false)) {
          // OS ignored requested address. Try different address.
          base = NULL;
          os::release_memory(requested_address, size);
          count--;
        } else {
          break;
        }
      }
    } else {
      base = os::reserve_memory(size, NULL, alignment);
    }

  //...
}
```