

使用-XX:+PrintFieldLayout 选项输出字段布局的结果如下：

```
com.classloading$ContendedTest1: field layout
  // @140 表示字段在类中的地址偏移量，通过对对象头 12 字节加上填充的 128 字节得到
  @140 --- instance fields start ---
  @140 "plainField1" Ljava.lang.Object;
  @144 "plainField2" Ljava.lang.Object;
  @148 "plainField3" Ljava.lang.Object;
  @152 "plainField4" Ljava.lang.Object;
  // 字段内存的末尾也需要填充 128 字节，152 加上 8 字节再加上 128 字节得到 288 字节
  @288 --- instance fields end ---
  @288 --- instance ends ---
```

@Contended 注解将使整个字段块的两端都被填充。注意，这里使用了 128 字节的填充数来避免伪共享，这个数是大多数硬件缓存行的 2 倍。

2. 在字段上应用@Contended注解

【实例 5-2】 有以下代码：

```
package com.classloading;

public static class ContendedTest2 {
    @Contended
    private Object contendedField1;
    private Object plainField1;
    private Object plainField2;
    private Object plainField3;
    private Object plainField4;
}
```

在字段上应用@Contended 注解将导致该字段从连续的字段内存空间中分离出来。最终的内存布局如下：

```
com.classloading$ContendedTest2: field layout
  @ 12 --- instance fields start ---
  @ 12 "plainField1" Ljava.lang.Object;
  @ 16 "plainField2" Ljava.lang.Object;
  @ 20 "plainField3" Ljava.lang.Object;
  @ 24 "plainField4" Ljava.lang.Object;
  // 与普通的字段之间进行填充，填充 128 字节
  @156 "contendedField1" Ljava.lang.Object; (contended, group = 0)
  // 字段内存的末尾也需要填充 128 字节
  @288 --- instance fields end ---
  @288 --- instance ends ---
```

3. 在多个字段上应用@Contended注解

【实例 5-3】 有以下代码：

```
package com.classloading;

public static class ContendedTest3 {
    @Contended
```