**SUMMARY**

A concurrency bug in Eclipse 3.1 JDT DEBUG

**DETAILS**

Some details can also be found at: <https://bugs.eclipse.org/bugs/show_bug.cgi?id=78534>

This bug is due to a data race.

fJavaMonitorThreads: HashMap

|  |  |
| --- | --- |
| Thread1 (ThreadMonitorManager.java) | Thread2 () |
| protected JavaMonitorThread getJavaMonitorThread(IJavaThread thread) {  JavaMonitorThread javaMonitorThread= (JavaMonitorThread) fJavaMonitorThreads.get(thread);  if (javaMonitorThread == null) {  javaMonitorThread= new JavaMonitorThread(thread);  fJavaMonitorThreads.put(thread, javaMonitorThread);  DebugPlugin.getDefault().asyncExec(new DetectDeadlock());  } | private void handleThreadTerminate(IJavaThread thread) {  // remove this thread  fJavaMonitorThreads.remove(thread);  } |
| public V put(K key, V value) {  if (key == null)  return putForNullKey(value);  int hash = hash(key.hashCode());  int i = indexFor(hash, table.length);  for (Entry<K,V> e = table[i]; e != null; e = e.next) {  Object k;  if (e.hash == hash && ((k = e.key) == key || key.equals(k))) {  V oldValue = e.value;  e.value = value;  e.recordAccess(this);  return oldValue;  }  }    modCount++;  addEntry(hash, key, value, i);  return null;  } | final Entry<K,V> More ...removeEntryForKey(Object key) {  int hash = (key == null) ? 0 : hash(key.hashCode());  int i = indexFor(hash, table.length);  Entry<K,V> prev = table[i];  Entry<K,V> e = prev;    while (e != null) {  Entry<K,V> next = e.next;  Object k;  if (e.hash == hash &&  ((k = e.key) == key || (key != null && key.equals(k)))) {  modCount++;  size--;  if (prev == e)  table[i] = next;  else  prev.next = next;  e.recordRemoval(this);  return e;  }  prev = e;  e = next;  }    return e;  } |
| 0-lock | 0-lock |