NEXTSTEP

Title: EOFaults and Infinite Loops

Entry Number:
Last Updated: 19 Apr 1995
Document Revision:

Keywords: EOF, EOFault, infinite loop

Question

Q: I am creating EOFault instances, and I find that sometimes when I trigger the fault, my process hangs. The backtrace looks like this:

```
#712 0x5006c1a in _objc_msgForward ()
#713 0x5006c4c in objc_msgSendv ()
#714 0x7817756 in -[NSInvocation invokeWithTarget:] ()
#715 0xe024a56 in -[_EOObjectFault forwardInvocation:] ()
#716 0x7816d92 in -[NSObject(ForwardInvocation) forward::] ()
#717 0x5006c1a in _objc_msgForward ()
#718 0x5006c4c in objc_msgSendv ()
#719 0x7817756 in -[NSInvocation invokeWithTarget:] ()
#720 0xe024a56 in -[_EOObjectFault forwardInvocation:] ()
#721 0x7816d92 in -[NSObject(ForwardInvocation) forward::] ()
#722 0x5006c1a in _objc_msgForward ()
#723 0x5006c4c in objc_msgSendv ()
#724 0x7817756 in -[NSInvocation invokeWithTarget:] ()
#725 0xe024a56 in -[_EOObjectFault forwardInvocation:] ()
#726 0x7816d92 in -[NSObject(ForwardInvocation) forward::] ()
#727 0x5006c1a in _objc_msgForward ()
#728 0x5006c4c in objc_msgSendv ()
#729 0x7817756 in -[NSInvocation invokeWithTarget:] ()
#730 0xe024a56 in -[_EOObjectFault forwardInvocation:] ()
#731 0x7816d92 in -[NSObject(ForwardInvocation) forward::] ()
#732 0x5006c1a in _objc_msgForward ()
#733 0x490a in -[AppDelegate createFaultForObject:] (self=0x185e38, _cmd=0xa54b, sender=0x1324c4) at
AppDelegate.m:48
```

what's up?

Answer

A: If you create a fault for an object that is already in the uniquing tables, when the object "faults" or is triggered you will get an infinite loop as seen here. The solution is to check and see if the object is in the uniquing tables before creating the fault, and if it is, return the object itself rather than creating a fault.

There are two uniquing tables, one in the EODatabase and one in the EODatabaseContext. The EODatabaseContext has its own table for objects that are inserted and before the transaction is committed. So for the period of time after the insert and before the commit, you need to go to the EODatabaseContext's uniquing tables. You access the tables with the **objectForPrimaryKey:entity:** method.

A convenient category on EOFault can be written that encapsulates this behavior:

```
+ safeObjectFaultWithPrimaryKey:(NSDictionary *)keyDict
    entity:(EOEntity *)entity
    databaseChannel:(EODatabaseChannel *)channel
    zone:(NSZone *)zone
{
    //  check first to see if this object has already been fetched or inserted
    id value = [[channel databaseContext] objectForPrimaryKey:keyDict entity:entity];

    //  return an EOFault only if the object has not been uniqued
    if ( !value )
    {
       value = [EOFault objectFaultWithPrimaryKey:keyDict entity:entity
                   databaseChannel:channel zone:zone];
    }

    return value;
}
```

When you want to create an EOFault instance, use this category in place of EOFault's **+objectFaultWithPrimaryKey:entity:databaseChannel:zone:**.

Valid for: EOF 1.0, EOF 1.1, NEXTSTEP 3.2 Developer, NEXTSTEP 3.3 Developer