Q: How does a server that registers like:

```
[NXConnection registerRoot:self  withName: "myname"];
```

get notification of a client that has disconnected? The delegate method **connection:didConnect:** indicates when the client connects, but not when it disconnects. Specifically, I want the server to know when a client application has been killed or crashed.

A: In order to receive invalidation notification you must select the object that is to be notified and register it for invalidation notification. This object then must conform to the NXSenderIsInvalid protocol. For example, one of your server objects (probably the one that called **registerRoot:withName:**) should conform to the NXSenderIsInvalid protocol. The method you need to implement to conform to this protocol is: **-senderIsInvalid:**. When the client connection dies, **senderIsInvalid:** will be called passing that connection.

You will also need to call **worryAboutPortInvalidation** (NXPort class method)Ðthis spawns a thread that

listens for a Port death and will message **senderIsInvalid:**. The code would look something like:

```
NXConnection *primary = [NXConnection registerRoot: rootObj  withName: "foo"];
[primary setDelegate:self];
[NXPort worryAboutPortInvalidation];
[primary run];
```

Finally, in your **connection:didConnect:** delegate method you should call **registerForInvalidationNotification:**. You should register on the second connection passed to this method.

Note: The above information assumes a non-AppKit program. If you are using the AppKit, this notification mechanism is already in place. Another important distinction is that in the AppKit case, there is not a separate thread, so there is less worrying that needs to be done about multi-threaded access.

See also: The introduction to Distributed Objects which is online in:

GeneralRef/06_DistributedObjects/IntroDistObjects.rtf and the Distributed Objects release notes.

QA877

Valid for 3.0