Simulating a drop, part two

devblogs.microsoft.com/oldnewthing/20080725-00

July 25, 2008

Raymond Chen

Last time, we wrote a tiny program to simulate dropping a file on another file, but we discovered that it didn't work for dropping a file onto `Mail Recipient.MAPIMail`. The reason, as you no doubt instantly recognized, is that the `MAPIMail` handler creates a worker thread, and we're exiting the process before the worker thread has finished its work.

And as you no doubt know by now, the solution is to use the `SHSetInstanceExplorer` function. Let's bring back the `ProcessReference` class and use it to solve our process lifetime problem.

```
int __cdecl wmain(int argc, WCHAR **argv)
{
 ProcessReference ref;
 …
```

Compile this program and run it with the command line

```
fakedrop c:\autoexec.bat "%USERPROFILE%\SendTo\Mail Recipient.MAPIMail"
```

to watch our process reference save the day.

Oh wait, it didn't help. What's going on?

Run this under the debugger and you'll see an interesting exception:

```
(918.110): Unknown exception - code 80010012 (first chance)
```

A little hunting through `winerror.h` reveals what this exception means:

```
//
// MessageId: RPC_E_SERVER_DIED_DNE
//
// MessageText:
//
//  The callee (server [not server application]) is not available and
//  disappeared; all connections are invalid. The call did not execute.
//
#define RPC_E_SERVER_DIED_DNE            _HRESULT_TYPEDEF_(0x80010012L)
```

Huh? What's this RPC stuff? Oh wait, COM uses RPC. That should be a clue.

Notice that our `ProcessReference`'s destructor runs after we have already uninitialized COM. When we invoked the `IDropTarget::Drop` method on the `MAPIMail` drop target, it kicked off a background thread to do the work, and in order to access the parameters from the background thread, it had to do a bit of marshalling, with the help of the functions with ridiculously long names `CoMarshalInterThreadInterfaceInStream` and the only slightly less ridiculous `CoGetInterfaceAndReleaseStream`. But since we tear down COM immediately, when the background thread gets around to asking, "Okay, and what was that file name?" the thread that has the answer (the main thread) has already shut down COM. Hence the "server died" error.

To fix this, we need to fix the scope of the process reference object:

```
if (argc == 3 && SUCCEEDED(CoInitialize(NULL))) {
 ProcessReference ref;
 …
```

Compile this program and run it with the same command line and… it still doesn't work! But this time you definitely know why: The destructor is running at the wrong time.

I leave it as an exercise to fix the destructor timing problem. To see whether you got it right, run the `fakedrop` command line again. When you successfully get an email message, then you know you've got it.

Raymond Chen

Follow