Enumerating all the programs that can open a particular file extension

devblogs.microsoft.com/oldnewthing/20151130-00

November 30, 2015

Raymond Chen

Today's Little Program enumerates all the applications which are registered for a particular file extension and lets the user choose one. We then open a file with that chosen program.

As always, Little Programs do little to no error checking.

```
#include <windows.h>
#include <ole2.h>
#include <shlobj.h>
#include <atlbase.h>
#include <atlalloc.h>
#include <vector>
#include <iostream>

std::vector<CComPtr<IAssocHandler>> LoadHandlers(
  PCWSTR extension,
  ASSOC_FILTER filter)
{
  std::vector<CComPtr<IAssocHandler>> handlers;
  CComPtr<IEnumAssocHandlers> enumerator;
  SHAssocEnumHandlers(extension, filter, &enumerator);
  for (CComPtr<IAssocHandler> handler;
       enumerator->Next(1, &handler, nullptr) == S_OK;
       handler.Release()) {
       handlers.push_back(handler);
  }
  return handlers;
}
```

The `LoadHandlers` function shows off the meat of the program: We use `SHAssocEnum-Handlers` to enumerate all the handlers for a particular extension. The results get saved into a vector.

```
auto
ChooseHandler(
  const std::vector<CComPtr<IAssocHandler>>& handlers,
  bool allowChooseMore) -> decltype(handlers.size())
{
  decltype(handlers.size()) i;
  for (i = 0; i < handlers.size(); i++) {
    CComHeapPtr<wchar_t> name;
    handlers[i]->GetUIName(&name);
    std::wcout << i << L": " << static_cast<PCWSTR>(name)
                             << std::endl;
  }
  if (allowChooseMore) {
    std::wcout << i << L": Show more handlers" << std::endl;
    i++;
  }

  decltype(handlers.size()) selection;
  std::wcin >> selection;
  if (std::wcin.fail()) selection = i + 1;
  return selection;
}
```

The `ChooseHandler` function prints the vector of handlers (and optionally adds a "Show more handlers" option). It collects the user's reply and returns it, using `handlers.size()` to represent the "Show more handlers" option, if available. If the user's input is invalid, we return a value that is out of range. (I'm assuming you don't have four billion handlers.)

```
int __cdecl main(int, char**)
{
  CCoInitialize init;
  ProcessReference ref;

  auto handlers = LoadHandlers(L".txt", ASSOC_FILTER_RECOMMENDED);
  auto selection = ChooseHandler(handlers, true);
  if (selection == handlers.size()) {
    handlers = LoadHandlers(L".txt", ASSOC_FILTER_NONE);
    selection = ChooseHandler(handlers, false);
  }

  if (selection < handlers.size()) {
    CComPtr<IDataObject> dobj;
    GetUIObjectOfFile(nullptr, L"C:\\windows\\win.ini",
                      IID_PPV_ARGS(&dobj));
    handlers[selection]->Invoke(dobj);
  }
  return 0;
}
```

And here's the main function that ties everything together.

After some initial throat-clearing, it loads up the recommended handlers for the `.txt` file extension and lets the user choose from among them.

If the user says “Show more handlers”, then we load up all handlers and try again.

We then take the user’s selection and open `WIN.INI` with that program.

Exercise: What is the purpose of the `ProcessReference` object?

Raymond Chen

Follow