NOTE: This Technical Note has been retired. Please see the Technical Notes page for current documentation.

Technical Note NW28
Constructing a Business Card DSSpec

This Technical Note describes how to create an AOCE catalog services specification structure (DSSpec) for an AOCE business card catalog item, given just a file system specification record (FSSpec) for the item.

[Sep 01 1994]

Introduction

The AOCE catalog services specification structure, defined by the `DSSpec` data type (see *Inside Macintosh :AOCE Application Interfaces* 2-36), is used throughout AOCE for performing various tasks such as getting and setting attributes in a record, specifying message addresses, etc. Many of the AOCE software routines defined in *Inside Macintosh :AOCE Application Interfaces* will give you a `DSSpec` for a given catalog item. For example, the Standard Catalog package contains a Catalog-Browsing panel, which allows the user to select records in catalogs. Once a selection is made, the selected item is returned as a `DSSpec` structure.

However, there are instances where you will be given just an `FSSpec` for a catalog item, and you will need to construct a `DSSpec` structure for that same item. For example, if your application uses the Drag Manager to handle dragging HFS objects onto your application's windows, it's possible that you will receive a drag of an HFS object that represents an AOCE catalog item (like a business card). In this case, all you are given for the object is an `FSSpec` structure. If you wanted to use some of the routines defined in the AOCE Catalog Manager on that item, you would need to pass a `RecordID` or `DSSpec` structure describing that item.

Back to top

Building a DSSpec

The following describes how to construct a `DSSpec` (from just a `FSSpec`) for a business card. Basically, business cards are degenerate personal catalogs containing a distinguished record. To find the distinguished record, we look for a pseudonym with a known type. By enumerating that pseudonym you can find the distinguished record. The process is as follows:

- Open the business card with `DirOpenPersonalDirectory` to get the reference number.
- Use `DirEnumerateGet` to get the distinguished record's `CID` (the `CID` and reference number are enough to access the record).

To make a fully specified `packedDSSpec`:

- Call `DirGetNameAndType` passing the `CID` gotten above, to get the distinguished record's type (use the file name as the record's name. If it is not the same, it is OK to set the record's name to the file name).
- Call `DirMakePersonalDirectoryRLI`, using the business card reference number to get the `RLI`.
- Create the `packedDSSpec` using the `OCEPackedDSSpecSize` and `OCEPackedDSSpec` routines.

Code Example

Here is the code, written in MPW C. You call the `ConstructPackedDSSpec` routine, passing a valid `FSSpec` along with a pointer to an empty `PackedDSSpecPtr` type variable. The code then returns a pointer to a valid `PackedDSSpec` structure in the `packedDSSpecPtr` parameter:

```
  /* global data */
#define  kEnumBufferSize  (4096)  /* Working data buffer */


/*****************************************************************
 *    ConstructPackedDSSpec
 *
 *     main code that builds a packedDSSpec struct from an FSSpec
 *     for a Business Card.
 *****************************************************************/

OSErr ConstructPackedDSSpec( FSSpecPtr        fsspecPtr,
                             PackedDSSpecPtr  *packedDSSpecPtr )
{
OSErr            err;
unsigned short   packedSpecSize;
DSSpec           dsspec;
RecordID         rid;
RString          name,type;
PackedRLI        packedRLI;
LocalRecordID    localRID;
CreationID       cid;

      name.dataLength = kRStringMaxBytes;
      type.dataLength = kRStringMaxBytes;
      OCENewLocalRecordID(&name,&type,&cid,&localRID);
      OCENewRecordID(&packedRLI,&localRID,&rid);

      err = GetRecordID(fsspecPtr,
                        &rid,
                        OCEGetIndRecordType(kBusinessCardRecTypeNum));
      if (err == noErr)
      {
         dsspec.extensionType = kOCEentnDSSpec;
         dsspec.extensionSize = 0;
         dsspec.extensionValue = NULL;
         dsspec.entitySpecifier = &rid;
         packedSpecSize = OCEPackedDSSpecSize(&dsspec);
         *packedDSSpecPtr = (PackedDSSpecPtr)NewPtr(packedSpecSize);
         err = MemError();
         if (err == noErr)
         {
            err = OCEPackDSSpec(&dsspec, *packedDSSpecPtr, packedSpecSize);
         }
      }

    return (err);
}


/*****************************************************************
 *    GetRecordID
 *
 *
 *****************************************************************/

OSErr GetRecordID( FSSpecPtr     fsspec,
                   RecordIDPtr   ridPtr,
                   RStringPtr    recordType)
```

```
{
OSErr              err;
DirParamBlock      pb;


    pb.openPersonalDirectoryPB.fsSpec = fsspec;
    pb.openPersonalDirectoryPB.accessRequested = fsRdPerm;
     /* get reference number for Business Card */
    err = DirOpenPersonalDirectory(&pb);
    if (err == noErr)
    {
       err = DoEnumerateGet( pb.openPersonalDirectoryPB.dsRefNum,
                             (long)&ridPtr->local,
                             recordType);
       if (err == noErr)
       {
          pb.getNameAndTypePB.identity = 0;
          pb.getNameAndTypePB.aRecord = ridPtr;
          err = DirGetNameAndType (&pb, false);
          if (err == noErr)
          {
             pb.makePersonalDirectoryRLIPB.fromFSSpec = fsspec;
             pb.makePersonalDirectoryRLIPB.pRLIBufferSize = kRLIMaxBytes;
             pb.makePersonalDirectoryRLIPB.pRLI = ridPtr->rli;
             err = DirMakePersonalDirectoryRLI(&pb);
           }
        }
        DirClosePersonalDirectory(&pb);
     }

 return (err);
}



/*****************************************************************
 *    DoEnumerateGet
 *
 *
 *****************************************************************/

OSErr DoEnumerateGet(short        dsRefNum,
                     long         clientData,
                     RStringPtr   recordType)
{
OSErr          err;
RStringPtr     typeList[1];
DirParamBlock  dirParamBlock;
Ptr            buffer;

#define        GET   (dirParamBlock.enumerateGetPB)
#define        PARSE (dirParamBlock.enumerateParsePB)


   buffer = NewPtr(kEnumBufferSize);
   if (buffer == NULL)
      err = MemError();
   else
   {
      GET.dsRefNum = dsRefNum;
      GET.identity = 0;
      GET.aRLI = NULL;    /* ignored if non-zero dsRefNum is passed */
      GET.startingPoint = 0;
      GET.sortBy = kSortByType;
      GET.sortDirection = kSortForwards;
      typeList[0] = recordType;
      GET.typesList = &typeList;
      GET.typeCount = 1;
      GET.enumFlags = kEnumPseudonymMask;
      GET.includeStartingPoint = true;
      GET.matchNameHow = kMatchAll;
      GET.matchTypeHow = kExactMatch;
      GET.getBuffer = buffer;
      GET.getBufferSize = kEnumBufferSize;
      err = DirEnumerateGet (&dirParamBlock, false);

      if (err == noErr || err == kOCEMoreData)
      {
         PARSE.clientData = clientData;
         PARSE.eachEnumSpec = myForEachDirEnumSpec;

         err = DirEnumerateParse (&dirParamBlock, false);
      }
   }

   return (err);
#undef GET
#undef PARSE

}

/*****************************************************************
 *    myForEachDirEnumSpec
 *
 *
 *****************************************************************/

static pascal Boolean myForEachDirEnumSpec (long clientData,
                                            const DirEnumSpec *enumSpec)
{
OSErr err;

   if ((enumSpec->enumFlag & kEnumPseudonymMask))
    {
          err = OCECopyLocalRecordID(&enumSpec->u.recordIdentifier,
                                (LocalRecordIDPtr)clientData);

          return (true);  /* stop, we found it */
    }
    else
          return (false);  /* get next one */
```

References:

Inside Macintosh :*AOCE Application Interfaces*

Inside Macintosh :*Files*

Drag Manager Programmer's Guide

Back to top

Downloadables

	Acrobat version of this Note (K)	Download

Back to top

Technical Notes by Date | Number | Technology | Title
Developer Documentation | Technical Q&As | Development Kits | Sample Code