Locating a Port

Before initiating a session, you can use either the **PPCBrowser** function or the **IPCListPorts** function to locate a port to communicate with.

Use the **PPCBrowser** function to display the program linking dialog box on the user's screen.

Note: Because this function displays a dialog box on the user's screen, you must not call the **PPCBrowser** function from an application that is running in the background.

In the program linking dialog box, the user chooses the computer, zone, and application. The zone list is not displayed if there is no network connection.

As shortcuts for the user, the program linking dialog box supports standard keyboard equivalents. Pressing Command-period or the Esc (Escape) key selects Cancel-pressing Enter or Return selects the OK button.

Each list is sorted in alphabetical order. As in the Chooser, the current list is indicated by a thick outline around its border. The program linking dialog box supports keyboard navigation and use of the arrow keys to select items from the current list. Pressing Tab or clicking the rectangle of another list switches the current list. Pressing Shift-Tab reverses the order in which the lists are selected. In addition, double-clicking an application name in the Programs list of the program linking dialog box is equivalent to clicking the OK button.

The **PPCBrowser** allows users to browse for PPC ports.

```
err = PPCBrowser  (prompt, applListLabel, defaultSpecified,
                   &theLocation, &thePortInfo, portFilter,
                   theLocNBPType);
```

If the defaultSpecified parameter is TRUE, the **PPCBrowser** function tries to select the PPC port specified by the parameters theLocation and thePortInfo when the program linking dialog box first appears. If the default cannot be found, the **PPCBrowser** selects the first PPC port in the list.

An application can open multiple ports as long as each port name is unique within a particular computer. Unique ports can have duplicate name fields but different types. For example, you can designate "make memo" as the application's name string and "word processor" as its type string. You can also designate a separate port as "make memo" (the application's name string) and "text only" (its type string).

In such a case, the **PPCBrowser** function does a secondary sort based on the port type. Ports with a type selector of ppcByCreatorAndType are displayed before ppcByString ports, and types are sorted alphabetically within each type selector.

The **PPCBrowser** function uses the **IPCListPorts** function to obtain the list of existing ports on a particular computer within a particular zone. The portFilter parameter of the **PPCBrowser** function allows you to filter the list of PPC ports before it displays them in the program linking dialog box. If this parameter is NIL, the names of all the existing PPC ports returned by the **IPCListPorts** function are displayed. If the portFilter field is not NIL, it

must contain a pointer to a port filter function that you create.

The following program illustrates how you use a sample port filter function. In this listing, the MyBrowserPortFilter function returns TRUE for ports with the port type string "Example".

```
// Using a port filter function
// Assumes inclusion of MacHeaders
#include <PPCToolBox.h>

// contains prototype for strcmp
#include <string.h>

// Prototype your filter procedure like this prior to using it
pascal Boolean MyBrowserPortFilter(LocationNameRec,PortInfoRec);

pascal Boolean MyBrowserPortFilter(LocationNameRec theLocationNameRec,
                                   PortInfoRec thePortInfoRec)
{
    // If port type selector for this port is a string
    // check for the right port, otherwise return FALSE(don't display port)
    if (thePortInfoRec.name.portKindSelector == ppcByString)
        // If string matches port we are looking for, display it
        if  ( !strcmp ((char *)thePortInfoRec.name.u.portTypeStr,
                       (char *)"\pExample" ) )
        return TRUE;
    else
        return FALSE;
    else
        return FALSE;
}
```

The **PPCBrowser** function calls your filter function once for each port on the selected computer. Your function should return TRUE for each port you want to display in the program linking dialog box, and FALSE for each port that you do not want to display. Do not modify the data in the filter function parameters theLocationNameRec and thePortInfoRec.

The **PPCBrowser** function returns the selected port name in the parameter thePortInfo. The **IPCListPorts** function returns the port names in the area of memory pointed to by the bufferPtr field of the **IPCListPorts** parameter block. Both functions specify each port name in a port information record. See also the description of the PortInfoRec.

If the authRequired field is TRUE, the port requires authentication before a session can begin. You should use the **StartSecureSession** function to initiate a session with this port. If this field returns FALSE, you can use either the **PPCStart** function or the **StartSecureSession** function to initiate a session. The name field of the port information record specifies an available port name.

The following program illustrates how you use the **PPCBrowser** function to display the program linking dialog box in order to obtain the location and name of a port chosen by the user. In this listing, the **PPCBrowser** function builds

lists of zones (shown in the AppleTalk Zones list of the program linking dialog box), objects (shown in the Macintoshes list), and ports (shown in the Programs list). In this example, the **PPCBrowser** function next tries to default to object "Moof" in the "Twilight" zone. If it matches the object and zone, it also tries to default to the port "Inside Macintosh" with the port type "Example".

Note that the data in the records LocationNameRec and PortInfoRec is used to match the names in the program linking dialog box. The data has nothing to do with the NBP type used by **NBPLookup** or the filtered PPC ports that show up in the program linking dialog box. **NBPLookup** uses the NBP type supplied in theLocNBPType. The PPC port names are filtered using the MyBrowserPortFilter function shown in the previous listing.

```
// Browsing through dictionary service ports
// Assuming inclusion of MacHeaders
#include <PPCToolBox.h>
#include <Script.h>

// Contains prototype for strcpy
#include <String.h>

// Prototype routine like this prior to calling it
OSErr MyPPCBrowser(LocationNameRec *,PortInfoRec *);

// Filter proc used.  See previous listing for an example.
pascal Boolean MyBrowserPortFilter(LocationNameRec,PortInfoRec);

OSErr MyPPCBrowser(LocationNameRec *theLocationNameRec,
                   PortInfoRec *thePortInfoRec)
{
    Str255    prompt;          // Prompt to display in dialogue
    Str255    applListLabel;   // Title for list of applications
    Boolean   defaultSpecified;
    Str32  theLocNBPType;

    strcpy ((char *)prompt, "Choose an example to link to:");
    CtoPstr(prompt);
    strcpy((char *)applListLabel,"Examples");
    CtoPstr(applListLabel);

    defaultSpecified = TRUE;

    theLocationNameRec->locationKindSelector = ppcNBPLocation;
    strcpy((char *)theLocationNameRec->u.nbpEntity.objStr,"Moof");
    CtoPstr(theLocationNameRec->u.nbpEntity.objStr);
    // typeStr is ignored
    strcpy((char *)theLocationNameRec->u.nbpEntity.zoneStr,"Twilight");
    CtoPstr(theLocationNameRec->u.nbpEntity.zoneStr);

    thePortInfoRec->name.nameScript = smRoman;
    strcpy ((char *)thePortInfoRec->name.name,"Inside Macintosh");
    CtoPstr(thePortInfoRec->name.name);
    thePortInfoRec->name.portKindSelector = ppcByString;
    strcpy ((char *)thePortInfoRec->name.u.portTypeStr,"Example");
```

```
        CtoPstr(thePortInfoRec->name.u.portTypeStr);

        // when building the list of objects (Macintoshes), show only
        // those with the NBP type "PPC Example"

        strcpy ((char *)theLocNBPType ,"PPC Example"); // match this NBP type
        CtoPstr(theLocNBPType);

        return PPCBrowser(prompt, applListLabel, defaultSpecified,
                theLocationNameRec, thePortInfoRec,
                (PPCFilterProcPtr)MyBrowserPortFilter, theLocNBPType);
        // Note that casting of MyBrowserPortFilter was necessary to pass
        //  compilation
}
```