PBDTGetAPPL Identify the application that can open a file with a given creator

#include <Files.h> **Finder Interface**

OSErr	**PBDTGetAPPL(***paramBlock, async***);**	
DTPBPtr	*paramBlock* ;	pointer to a DTP Param Block
Boolean	*async* ;	0 = await completion; 1 = immediate return

Parameter block

→	12	ioCompletion	long	completion routine
←	16	ioResult	short	result code
→	18	ioNamePtr	long	pointer to application's name
→	24	ioDTRefNum	short	database reference number
→	26	ioIndex	short	index into application list
←	28	ioTagInfo	long	application's creation date
→	52	ioFileCreator	long	application's signature
←	100	ioAPPLParID	long	application's parent directory

For an application in the database specified in ioDTRefNum with the signature specified in ioFileCreator, **PBDTGetAPPL** returns the filename in ioNamePtr, the parent directory ID in ioAPPLParID, and the creation date in ioTagInfo. A single call, with ioIndex set to 0, finds the application file with the most recent creation date. If you want to retrieve all copies of the appli-cation with the given signature, start with ioIndex set to 1 and increment until ioResult returns afpItemNotFound; when called multiple times in this fashion, **PBDTGetAPPL** returns the application's copies, including the file with the most recent creation date, in arbitrary order.

Returns: an Error code. It will be one of the following:

noErr	(0)	No error
ioErr	(-36)	I/O error
rfNumErr	(-51)	Reference number invalid
extFSErr	(-58)	External file system-file system identifier is nonzero
afpItemNotFound	(-5012)	Information not found

Note: There is a second, asynchronous, version of this function. It does not take a second parameter; instead, it adds the suffix "Async" to the name of the routine.

Similarly, the third (synchronous) version of the routine does not take a second parameter; instead, it adds the suffix "Sync" to the name of the routine.

Note, however, that the second and third versions of these routines do not use the glue code that the first versions use and are therefore more efficient.