LaunchParamBlockRec structure

#include <Processes.h>

typedef struct **LaunchParamBlockRec**

		Size	Offset	Description
unsigned long	reserved1;	4	0	reserved
unsigned short	reserved2;	2	4	reserved
unsigned short	launchBlockID;	2	6	extended block
unsigned long	launchEPBLength;	4	8	length of block
unsigned short	launchFileFlags;	2	12	Finder flags of application
LaunchFlags	launchControlFlags ;	2	14	launch options
FSSpecPtr	launchAppSpec;	4	16	location of application file
ProcessSerialNumber	launchProcessSN;	8	20	returned psn
unsigned long	launchPreferredSize;	4	28	returned pref size
unsigned long	launchMinimumSize;	4	32	returned min size
unsigned long	launchAvailableSize ;	4	36	returned available size
AppParametersPtr	launchAppParameters;	4	40	high level event

} **LaunchParamBlockRec**; 44

typedef LaunchParamBlockRec ***LaunchPBPtr**;

Notes: In the *launchEPBLength* field, specify the constant extendedBlockLen to indicate the length of the remaining fields in the **LaunchParamBlockRec**. For compatibility, you should always specify the length value in this field.

The *launchFileFlags* field contains the Finder flags for the application file. (See the **Finder Interface** for a description of the Finder flags.) The **LaunchApplication** function sets this field for you if you set the bit defined by the launchNoFileFlags constant in the *launchControlFlags* field. Otherwise, you must get the Finder flags from the application file and set this field yourself (by using **GetFInfo** , for example).

In the *launchControlFlags* field, you specify various options that control how the specified application is launched. See **Launching Other Applications**, for information on the launch control flags.

You specify the application to launch using the *launchAppSpec* field of the **LaunchParamBlockRec**. You specify a pointer to a file system specification record (**FSSpec**) in this field. See the **File Manager** for a complete description of the file system specification record.

LaunchApplication sets the initial default volume of the application to the parent directory of the application file.

If **LaunchApplication** successfully launched the application, **LaunchApplication** returns in the *launchProcessSN* field a process serial number that you can use in **Process Manager** routines to refer to this application.

The *launchPreferredSize* and *launchMinimumSize* fields of the **LaunchParamBlockRec** are returned by **LaunchApplication** and are based on their corresponding values in the 'SIZE' resource. These values may be greater than those specified in the application's 'SIZE' resource

because the returned sizes include any adjustments to the size of the application's stack. See the **Event Manager** for information on how the size of the application stack is adjusted. **LaunchApplication** always returns values for these fields whether or not the launch was successful. These values are 0 if an error occurred-for example, if the application file could not be found.

The *launchAvailableSize* field is returned by **LaunchApplication** only when the memFullErr result code is returned. This value indicates the largest partition size currently available for allocation.

The *launchAppParameters* field specifies the first high-level event sent to an application. If you set this field to NIL, the **LaunchApplication** function automatically creates and sends an Open Application event to the launched application. (See the **Apple Event Manager** for a description of this event.) To send a particular high-level event to the launched application, you can specify a pointer to an application parameters record. The application parameters record is defined by the data type **AppParameters** .

You specify the high-level event in the fields *theMsgEvent*, *eventRefCon*, and *messageLength*. The body of the message immediately follows the *messageLength* field. The **Event Manager** describes how to use high-level events.