AccessParam structure

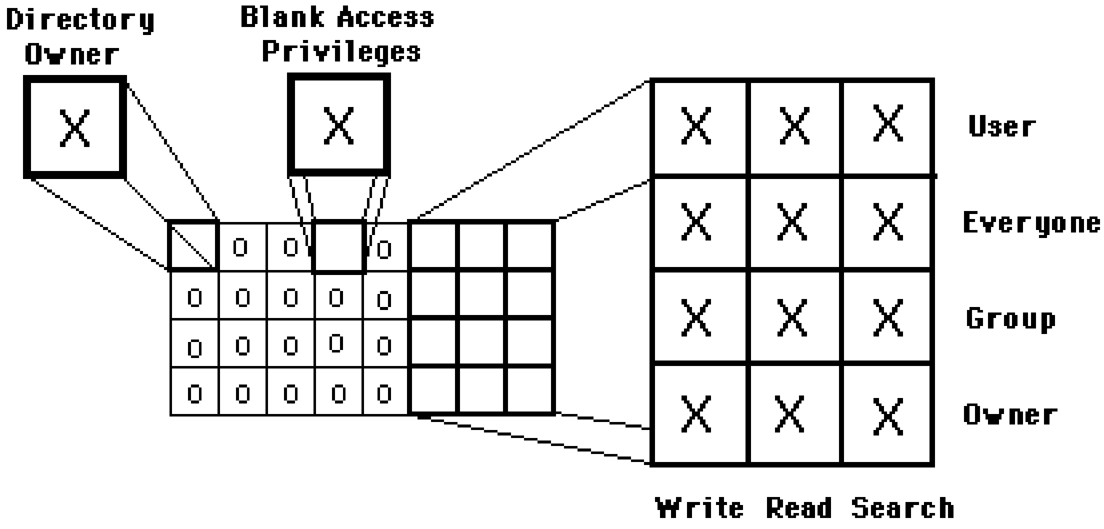
```
#include <Files.h>

typedef struct AccessParam{          Size   Offset   Description
  ParamBlockHeader                   24       0      common fields of ParamBlock types
  short            filler3;          2       24
  short            ioDenyModes;      2       26      access rights data
  short            filler4 ;         2       28
  char             filler5;          1       30
  char             ioACUser;         1       31      access rights for directory only
  long             filler6;          4       32
  long             ioACOwnerID;      4       36      owner ID
  long             ioACGroupID;      4       40      group ID
  long             ioACAccess;       4       44      access rights
} AccessParam;                       48
```

Notes: The **AccessParam** structure is used in shared environments to define the extent of access privileges for specific files, volumes and directories.

The most common way to use this structure is to allocate a union that is an aggregate and create and initialize a pointer to the desired data type. See HParamBlockRec for an example.

The ioACAccess field consists of a long integer with the format: uueeggoo. uu represents the user's rights, ee represents everyone's rights, gg represents the group's rights and oo represents the owner's rights. It can be represented as:



From left to right,

bit 7 is set if the user is not the directory's owner

bits 6-5 are reserved and are returned set to 0 bit 4 is set if folder has same access privileges as its parent

bit 3 is reserved and is returned set to 0

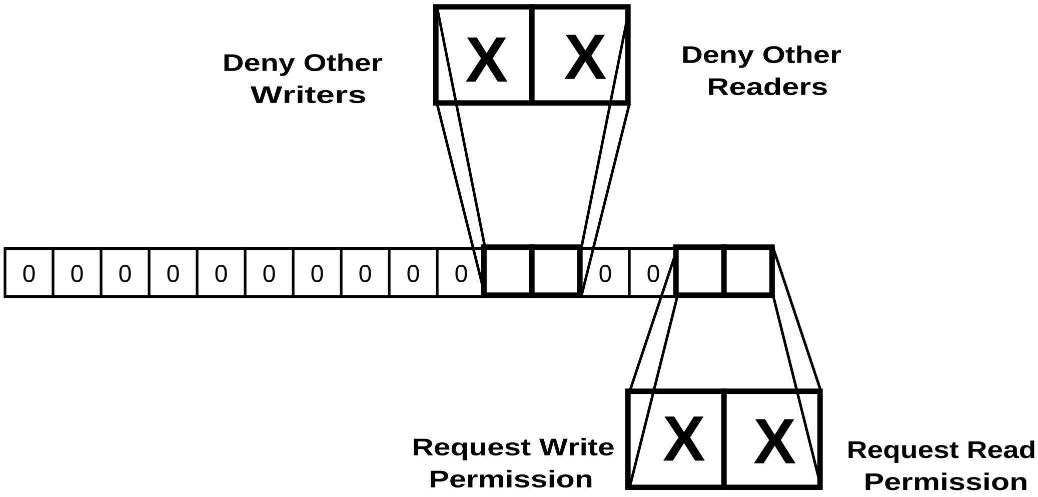
bit 2 is set if the user does not have Write privileges

bit 1 is set if the user does not have Read privileges

bit 0 is set if the user does not have Search privileges

User's rights are the logical OR of the rights for Everyone, Group and Owner. The User bit and the Owner bit are returned only from the GetDirAccess call and are never passed by the SetDirAccess routine. Changing a folder's owner requires changing the OwnerID field of the SetDirAccess routine.

The ioDenyMode integer contains permissions information and looks like:



From left to right,

bits 15 to 6 are reserved and set to zero

bit 5 is set to deny other writers

bit 4 is set to deny other readers

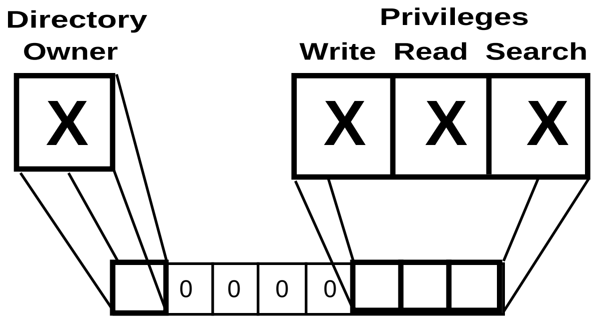
bits 3 and 2 are reserved and set to zero

bit 1 is set to request Write permission

bit 0 is set to request Read permission

The ioDenyMode field is used in calls PBHOpenDeny and PBHopenRFDeny.

The ioACUser field is a SignedByte in a format that looks like:



From left to right:

bit 7 is set if the user is not the directory's owner

bits 6 to 3 are reserved and set to 0

bit 2 is set if the user does not have Write privileges

bit 1 is set if the user does not have Read privileges

bit 0 is set if the user does not have Search privileges

The ioACUser field provides information in the above format with regard to the user's access privileges for a directory whose volume supports access controls. It is used by the **PBGetCatInfo** function.