

Asking for a Shared Memory Segment - shmget()

The system call that requests a shared memory segment is **shmget()**. It is defined as follows:

```
shm_id = shmget(
    key_t    k,          /* the key for the segment */
    int      size,       /* the size of the segment */
    int      flag);      /* create/use flag */
```

In the above definition, **k** is of type **key_t** or **IPC_PRIVATE**. It is the numeric key to be assigned to the returned shared memory segment. **size** is the size of the requested shared memory. The purpose of **flag** is to specify the way that the shared memory will be used. For our purpose, only the following two values are important:

1. **IPC_CREAT | 0666** for a server (*i.e.*, creating and granting read and write access to the server)
2. **0666** for any client (*i.e.*, granting read and write access to the client)

Note that due to Unix's tradition, **IPC_CREAT** is *correct* and **IPC_CREATE** is *not!!!*

If **shmget()** can successfully get the requested shared memory, its function value is a non-negative integer, the shared memory ID; otherwise, the function value is negative. The following is a server example of requesting a private shared memory of four integers:

```
#include <sys/types.h>
#include <sys/ipc.h>
#include <sys/shm.h>
#include <stdio.h>

.....
int      shm_id;          /* shared memory ID */
.....
shm_id = shmget(IPC_PRIVATE, 4*sizeof(int), IPC_CREAT | 0666);
if (shm_id < 0) {
    printf("shmget error\n");
    exit(1);
}

/* now the shared memory ID is stored in shm_id */
```

If a client wants to use a shared memory created with **IPC_PRIVATE**, it must be a child process of the server, created **after** the parent has obtained the shared memory, so that the private key value can be passed to the child when it is created. For a client, changing **IPC_CREAT | 0666** to **0666** works fine. **A warning to novice C programmers:** don't change **0666** to **666**. The leading **0** of an integer indicates that the integer is an octal number. Thus, **0666** is 110110110 in binary. If the leading zero is removed, the integer becomes six hundred sixty six with a binary representation 1111011010.

Server and clients can have a parent/client relationship or run as separate and unrelated processes. In the former case, if a shared memory is requested and

attached prior to forking the child client process, then the server may want to use **IPC_PRIVATE** since the child receives an identical copy of the server's address space which includes the attached shared memory. However, if the server and clients are separate processes, using **IPC_PRIVATE** is unwise since the clients will not be able to request the same shared memory segment with a unique and unknown key.

Suppose process 1, a server, uses **shmget()** to request a shared memory segment successfully. That shared memory segment exists somewhere in the memory, but is not yet part of the address space of process 1 (shown with dashed line below). Similarly, if process 2 requests the same shared memory segment with the same key value, process 2 will be granted the right to use the shared memory segment; but it is not yet part of the address space of process 2. To make a requested shared memory segment part of the address space of a process, use **shmat()**.

