

**NAME**

getpagesize - get memory page size

**SYNOPSIS**

```
#include <unistd.h>
```

```
int getpagesize(void);
```

Feature Test Macro Requirements for glibc (see [feature\\_test\\_macros\(7\)](#)):

```
getpagesize():
```

Since glibc 2.12:

```
    _BSD_SOURCE ||  
    !(_POSIX_C_SOURCE >= 200112L || _XOPEN_SOURCE >= 600)
```

Before glibc 2.12:

```
    _BSD_SOURCE || _XOPEN_SOURCE >= 500 ||  
    _XOPEN_SOURCE && _XOPEN_SOURCE_EXTENDED
```

**DESCRIPTION**

The function `getpagesize()` returns the number of bytes in a memory page, where page is a fixed-length block, the unit for memory allocation and file mapping performed by [mmap\(2\)](#).

**CONFORMING TO**

SVr4, 4.4BSD, SUSv2. In SUSv2 the `getpagesize()` call is labeled LEGACY, and in POSIX.1-2001 it has been dropped; HP-UX does not have this call.

**NOTES**

Portable applications should employ `sysconf(_SC_PAGESIZE)` instead of `getpagesize()`:

```
#include <unistd.h>  
long sz = sysconf(_SC_PAGESIZE);
```

(Most systems allow the synonym `_SC_PAGE_SIZE` for `_SC_PAGESIZE`.)

Whether `getpagesize()` is present as a Linux system call depends on the architecture. If it is, it returns the kernel symbol `PAGE_SIZE`, whose value depends on the architecture and machine model. Generally, one uses binaries that are dependent on the architecture but not on the machine model, in order to have a single binary distribution per architecture. This means that a user program should not find `PAGE_SIZE` at compile time from a header file, but use an actual system call, at least for those architectures (like sun4) where this dependency exists. Here glibc 2.0 fails because its `getpagesize()` returns a statically derived value, and does not use a system call. Things are OK in glibc 2.1.

**SEE ALSO**

[mmap\(2\)](#), [sysconf\(3\)](#)

**COLOPHON**

This page is part of release 3.74 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at <http://www.kernel.org/doc/man-pages/>.