

**NAME**

sysv\_signal – signal handling with System V semantics

**LIBRARY**

Standard C library (*libc*, *-lc*)

**SYNOPSIS**

```
#define _GNU_SOURCE      /* See feature_test_macros(7) */
#include <signal.h>

typedef void (*sighandler_t)(int);

sighandler_t sysv_signal(int signum, sighandler_t handler);
```

**DESCRIPTION**

The **sysv\_signal()** function takes the same arguments, and performs the same task, as **signal(2)**.

However **sysv\_signal()** provides the System V unreliable signal semantics, that is: a) the disposition of the signal is reset to the default when the handler is invoked; b) delivery of further instances of the signal is not blocked while the signal handler is executing; and c) if the handler interrupts (certain) blocking system calls, then the system call is not automatically restarted.

**RETURN VALUE**

The **sysv\_signal()** function returns the previous value of the signal handler, or **SIG\_ERR** on error.

**ERRORS**

As for **signal(2)**.

**ATTRIBUTES**

For an explanation of the terms used in this section, see **attributes(7)**.

Interface	Attribute	Value
<b>sysv_signal()</b>	Thread safety	MT-Safe

**STANDARDS**

This function is nonstandard.

**NOTES**

Use of **sysv\_signal()** should be avoided; use **sigaction(2)** instead.

On older Linux systems, **sysv\_signal()** and **signal(2)** were equivalent. But on newer systems, **signal(2)** provides reliable signal semantics; see **signal(2)** for details.

The use of *sighandler\_t* is a GNU extension; this type is defined only if the **\_GNU\_SOURCE** feature test macro is defined.

**SEE ALSO**

**sigaction(2)**, **signal(2)**, **bsd\_signal(3)**, **signal(7)**