

# Alternatives to systemd

This page describes the various init systems which are available as alternatives to systemd

**Init** is the first process started during system boot. It is a daemon process that continues running until the system is shut down. Init is the direct or indirect ancestor of all other processes, and automatically adopts all orphaned processes. It is started by the kernel using a hard-coded filename; if the kernel is unable to start it, panic will result. Init is typically assigned process identifier 1.

The *init scripts* (aka *rc* runtime configuration scripts) are launched by the init process to guarantee basic functionality on system start and shutdown. This includes (un)mounting of file systems and launching of daemons. A *service manager* takes this one step further by providing active control over launched processes, or [process supervision](#). An example is to monitor for crashes and restart processes accordingly.

These components combine to the *init system*. Some init systems incorporate the service manager in the init process or have init scripts in close relation to them. Below, such init systems are referred to as *integrated*, although entries in different categories may explicitly depend on each other.

A nice (but still non-comprehensive) overview of init systems can be found in [this](#) blog entry, titled “A history of modern init systems (1992-2015)”

## init systems

System	Description	Latest release
<a href="#">BusyBox</a> init	quite small init without runlevels, solely signal driven IPC (no fifos, sockets, SysV IPC)	2019-10-25
<a href="#">dinit</a>	dependency based C++ init with process supervision, roll-back, and socket activation	2020-01-01
<a href="#">finit</a>	fast, event based, modular, extensible (C hooks/plugins) init with SysV runlevels, proper daemon supervision and logging, and optional builtin getty and inetd	2018-01-23
<a href="#">nosh</a>	exec chaining, dependency based <a href="#">daemontools family</a> C++ init and supervision suite with reliable logging, (console) virtual terminal management, systemd unit file compatibility; can also be used as service manager under a different init	2019-03-17
<a href="#">openrc-init</a>	init system of <a href="#">OpenRC</a> , shipped since version 0.25	2019-08-20
<a href="#">pies</a>	dependency based GNU init daemon and super server with SysV runlevels, daemon supervision and logging, inetd functionality, socket activation, fine grained per service access control, understands SysV inittab, inetd.conf, and MeTA1 config files	2019-07-02
<a href="#">procd</a> (2)	OpenWrt init and process/service management daemon with ubus integration	2018-07-30
<a href="#">runit-init</a>	init from <a href="#">runit</a>	2014-08-10
<a href="#">sinit</a>	Simple init initially based on Rich Felker’s minimal init	2018-03-26
<a href="#">sninit</a>	Small init implementation with SysV init like (sub)runlevels	2017-12-29
<a href="#">SysV</a> init (2)	Traditional System V init. New <a href="#">release: v2.96 on July 07 2019</a>	2019-09-11

System	Description	Latest release
<a href="#">ToyBox</a> init (2)	similar to/(almost) compatible with BusyBox init (same config syntax/file)	2019-10-28
<a href="#">ToyBox</a> oneit (2)	very simple init launcher (just (re)spawns a single child process and reacts to incoming signals)	2019-10-28
<a href="#">ueld</a>	simple configuration, solely signal driven (like BSD and Busy/ToyBox init)	2018-07-09
<a href="#">uinit</a>	Smallest init possible	2017-05-16

## process supervisors (overview) and service managers

service manager: a **suite of programs** which start and stop services, both long-running daemons and one-time initialization scripts, in the proper order according to a dependency tree

Program	Description	Latest release
<a href="#">s6</a>	a collection of system tools built around <a href="#">s6</a> and <a href="#">s6-rc</a> created to make the implementation and manipulation of service files easier	2019-12-16
<a href="#">Circus</a> (docs)	runs, controls and monitors processes and sockets; written in <a href="#">Python</a> ; uses <a href="#">ZeroMQ</a>	2019-12-27
<a href="#">daemontools</a> encore	an enhanced version of daemontools ( <a href="#">github repository</a> )	2018-10-14
<a href="#">GNU Shepherd</a>	service manager and init daemon written in <a href="#">Guile</a> (formerly known as GNU dmd)	2019-05-11
<a href="#">MiniBase</a>	small userspace utilities for Linux; includes a process supervisor that can be used as stage 2 of process #1 via exec chaining akin to <a href="#">s6-svscan</a>	2019-11-14
<a href="#">OpenRC</a> (2, 3)	dependency based service management suite; includes a process supervisor and a simple init alternative	2018-08-06
<a href="#">runit</a>	process supervision suite in the line of daemontools; includes a simple <a href="#">init (2) replacement</a>	2014-08-10
<a href="#">s6</a>	small process supervision suite for UNIX ala daemontools, runit, and perp; can be used as stage 2 of process #1 via <a href="#">exec chaining</a>	2019-10-21
<a href="#">s6-rc</a>	service management suite for <a href="#">s6</a>	2019-21-21
<a href="#">Supervisor</a>	process control daemon written in Python	2018-02-15
<a href="#">ssm</a>	simple service manager bash scripts for Linux	2018-03-06
<a href="#">supervise-daemon</a> from <a href="#">OpenRC</a>	superise-daemon introduced in OpenRC .0.21	2019-08-20
<a href="#">Supervision</a> Framework	OpenRC-like service management suite for daemontools(-encore), runit, and s6	2018-09-01
<a href="#">systemctl-replacement</a>	drop-in reading systemd service descriptions, with docker-oriented init function, written in Python	2019-10-24

## udev alternatives for automatic Linux device file creation and management

[udev](#) (userspace /dev) is a device manager for the Linux kernel.

In April 2012, udev's source code was merged into the systemd source tree.[\[1\]](#)

Although udev can still be compiled for usage without systemd, Lennart Poettering said that they will not polish udevd outside of systemd[\[2\]](#), adding:

*"Yes, udev on non-systemd systems is in our eyes a dead end, in case you haven't noticed it yet. I am looking forward to the day when we can drop that support entirely."* So to be on the safe side, you probably want to use an alternative to udev:

DevMan	Description	Last release
<a href="#">eudev (2, 3)</a>	Gentoo's maintained fork of udev	2017-11-22
<a href="#">mdev</a>	the plug-and-play manager built into BusyBox	2018-07-31
<a href="#">mdevd (github)</a>	nonforking Linux kernel uevent handler daemon; uses the mdev config file format	2018-08-14
<a href="#">nldev (2)</a>	Suckless netlink frontend for mdev; replacements for udevd and udevadm (see also <a href="#">nlmon</a> )	2018-08-03
<a href="#">smdev</a>	mostly mdev-compatible Suckless program to manage device nodes	2015-04-12

For more see [this](#) StackExchange question.

Related helper tool: [udevil \(2\)](#) (un)mounts removable devices without a password, shows device info, and monitors device changes. It can also mount ISO files, nfs://, smb://, ftp:, ssh:// and WebDAV URLs, and tmpfs/ramfs filesystems. Uses Glib and libudev (i. e. requires (e)udev) and can work completely without systemd, consolekit, policykit, dbus, udisks, gvfs & fuse (although it can coexist with any of these).

## abandoned/defunct/discontinued/dormant/halted projects

[Listed here](#)

## See also

- [Gentoo wiki page: Comparison of init systems](#)
- [A history of modern init systems \(1992-2015\)](#)
- [Understanding \(SysV\) init](#)
- [init systems overview](#)
- [init system features and benefits](#)
- [Demystifying the init system](#) (init written in [Ruby](#))
- Process supervision: Problem [apparently](#) solved
- How [process supervision](#) and [service management](#) differ (according to Laurent Bercot, author of s6(-rc))
- An [overview](#) of daemontools family supervision suites

From:

<https://without-systemd.frama.wiki/> - **Without Systemd**

Permanent link:

[https://without-systemd.frama.wiki/alternatives\\_to\\_systemd](https://without-systemd.frama.wiki/alternatives_to_systemd)

Last update: **2020/01/16 21:16**

