systemd Cheat Sheet

systemd is a suite of basic building blocks for a Linux system. It provides a system and service manager that runs as PID 1 and starts the rest of the system. It also provides a logging daemon, tracks logged-in users and running containers and virtual machines, maintains mount and automount points, and more.

Services	
List all available services	`systemctl list-unit-files --type service`
Start service **sshd**	`systemctl start sshd`
Stop service **sshd**	`systemctl stop sshd`
Show status of service **sshd**	`systemctl status sshd`
Start **sshd** now and at system startup	`systemctl enable --now sshd`
At system startup, start (enable) **sshd**	`systemctl enable sshd`
At system startup, do not start **sshd**	`systemctl disable sshd`
Show whether service **sshd** is enabled	`systemctl is-enabled sshd`
Prevent service from starting (mask)	`systemctl mask sshd`
Unmask service, allowing it to be started	`systemctl unmask sshd`

Targets (runlevels)	
List all available targets	`systemctl list-unit-files --type target`
Boot to a graphical desktop	`systemctl set-default graphical`
Boot to a text console	`systemctl set-default multi-user`
Show default boot target	`systemctl get-default`
Show dependencies of a target	`systemctl list-dependencies graphical`

systemd Cheat Sheet

Logs		
View all system logs	`journalctl`	
View system logs from most recent	`--catalog --pager-end`	`-xe`
View logs for unit **foo**	`--unit foo`	`-u`
View logs since boot NUMBER (default: current)	`--boot`	`-b`
List boot numbers	`--list-boots`	

Introspection	
Show contents of unit file	`systemctl cat sshd`
Show unit file settings	`systemctl show sshd`
Show whether unit is active	`systemctl is-active sshd`
Show whether unit has failed	`systemctl is-failed sshd`
Edit unit file configuration	`sudo systemctl edit sshd`
Restart daemon	`sudo systemctl daemon-reload`

Power	
Power down the system	`systemctl poweroff`
Reboot, inserting the message "foo" into the logs	`systemctl --message="foo" reboot`
Reboot container or virtual machine **foo**	`systemctl --machine=foo reboot`
Halt remote host **example** as user **tux**	`systemctl --host=tux@example halt`