参考链接：

https://www.ruanyifeng.com/blog/2016/03/systemd-tutorial-part-two.html

一，起单独服务

准备工作：游戏微服务api-money

```
[appuser@LOCAL-192-168-81-15 api-money]$ ls
api-money  api-money@.service  conf.ini  log  start_group
[appuser@LOCAL-192-168-81-15 api-money]$ pwd
/data/gameserver/gamehall/api-money
```

1. 新增api-money@.service 配置文件

2. 将api-money@.service文件拷贝至 /etc/systemd/system/ 下

```
[appuser@LOCAL-192-168-81-15 api-money]$ sudo cp api-money@.service /etc/systemd/system
```

3. 更新system配置文件

```
[appuser@LOCAL-192-168-81-15 api-money]$ sudo systemctl daemon-reload
```

4. 设置服务开机启动

```
[appuser@LOCAL-192-168-81-15 api-money]$ sudo systemctl enable api-money@1
Created symlink from /etc/systemd/system/domino.target.wants/api-money@1.service to /etc/systemd/system/api-m
oney@.service.
```

5. 启动服务

```
[appuser@LOCAL-192-168-81-15 api-money]$ sudo systemctl start api-money@1.service
Job for api-money@1.service failed because the control process exited with error code. See "systemctl status a
[appuser@LOCAL-192-168-81-15 api-money]$ journalctl -xe
Hint: You are currently not seeing messages from other users and the system.
      Users in the 'systemd-journal' group can see all messages. Pass -q to
      turn off this notice.
No journal files were opened due to insufficient permissions.
[appuser@LOCAL-192-168-81-15 api-money]$ sudo journalctl -xe
Jan 28 10:25:49 LOCAL-192-168-81-15 kill[6771]: -V, --version  output version information and exit
Jan 28 10:25:49 LOCAL-192-168-81-15 kill[6771]: For more details see kill(1).
Jan 28 10:25:49 LOCAL-192-168-81-15 systemd[1]: Failed to start api-money@1.service money  service of domino s
-- Subject: Unit api-money@1.service has failed
```

启动失败：使用sudo journalctl -xe 查看启动失败原因

6. 启动服务成功后查看服务状态

```
[admin@LOCAL-192-168-81-15 api-money]# systemctl start api-money@1
[admin@LOCAL-192-168-81-15 api-money]# systemctl status api-money@1.service
● api-money@1.service - api-money@1.service money  service of domino service group
   Loaded: loaded (/etc/systemd/system/api-money@.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2021-01-28 14:35:50 CST; 8s ago
 Main PID: 24159 (api-money)
   CGroup: /system.slice/system-api\x2dmoney.slice/api-money@1.service
           └─24159 /data/gameserver/gamehall/api-money/api-money -i 1

Jan 28 14:35:50 LOCAL-192-168-81-15 systemd[1]: Started api-money@1.service money  service of domino service group.
Jan 28 14:35:50 LOCAL-192-168-81-15 bash[24159]: My App begin to Init ...
Jan 28 14:35:50 LOCAL-192-168-81-15 bash[24159]: loadFrameConf finished ...
Jan 28 14:35:50 LOCAL-192-168-81-15 bash[24159]: initRPC finished ...
Jan 28 14:35:50 LOCAL-192-168-81-15 bash[24159]: 2021/01/28 14:35:50 Connected to 192.168.81.15:2181
Jan 28 14:35:50 LOCAL-192-168-81-15 bash[24159]: 2021/01/28 14:35:50 Authenticated: id=72067437328864631, timeout=4000
Jan 28 14:35:50 LOCAL-192-168-81-15 bash[24159]: 2021/01/28 14:35:50 Re-submitting `0` credentials after reconnect
```

kill掉进程后，会自动拉起服务

```
[admin@LOCAL-192-168-81-15 api-money]# ps -ef | grep api-money
appuser  24159     1  0 14:35 ?        00:00:00 /data/gameserver/gamehall/api-money/api-money -i 1
admin    24238 19336  0 14:36 pts/20   00:00:00 grep --color=auto api-money
[admin@LOCAL-192-168-81-15 api-money]# kill -9 24159
[admin@LOCAL-192-168-81-15 api-money]# ps -ef | grep api-money
appuser  24244     1  6 14:37 ?        00:00:00 /data/gameserver/gamehall/api-money/api-money -i 1
admin    24274 19336  0 14:37 pts/20   00:00:00 grep --color=auto api-money
[admin@LOCAL-192-168-81-15 api-money]#
```

注意：

api-money@1 api-money@.service 中的@符号是为了起多组实例，如果服务是单个实例服务，可以不用加@符号。

```
[appuser@LOCAL-192-168-81-15 api-money]$ sudo systemctl enable api-money@
Failed to execute operation: Unit name api-money@.service is missing the instance name.
```

二、将多个服务加入组内

1. 在api-money.service中加入Parfof标记

```
[Unit]
Description=%n access  service of domino service group
Documentation=
PartOf=domino.target
;Start and stop with domino.targt.
```

```
[Install]
WantedBy=domino.target
```

2. 增加domino.target文件，增加组的定义，放置在 /etc/systemd/system/ 下

```
stem_start | api-store@.service | domino.target | access@.service
[Unit]
Description=domino daemon service group
Requires=basic.target
Conflicts=rescue.service rescue.target
After=basic.target rescue.service rescue.target
AllowIsolate=no
```

在执行enable操作时，会将服务配置放在/etc/systemd/system目录下的domino.target.wants子目录下，可以直接操作一整组服务。（xxx.target.wants是enable后自动生成的）

```
te          dbus-org.freedesktop.nm-dispatcher
vice        default.target
            default.target.wants
ice         domino.target
rvice       domino.target.wants
ervice      dtsvr@.service
 service    getty.target.wants
```

3. 对整组进行操作：

```
[admin@LOCAL-192-168-81-15 api-money]# ls
api-money  api-money.service  conf.ini  log  start_group
[admin@LOCAL-192-168-81-15 api-money]# systemctl start gtjtest.target
[admin@LOCAL-192-168-81-15 api-money]# systemctl status api-money.service
● api-money.service - api-money.service money  service of domino service group
   Loaded: loaded (/etc/systemd/system/api-money.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2021-01-28 15:08:53 CST; 4s ago
 Main PID: 27173 (api-money)
   CGroup: /system.slice/api-money.service
           └─27173 /data/gameserver/gamehall/api-money/api-money -i 1
```

systemctl start targetname

三、name.service配置文件的书写规则

unit：启动顺序与依赖关系

service：启动行为

install:安装配置，开机启动

[Unit]区块通常是配置文件的第一个区块，用来定义 Unit 的元数据，以及配置与其他 Unit 的关系。它的主要字段如下。

Description：简短描述

Documentation：文档地址

Requires：当前 Unit 依赖的其他 Unit，如果它们没有运行，当前 Unit 会启动失败

Wants：与当前 Unit 配合的其他 Unit，如果它们没有运行，当前 Unit 不会启动失败

BindsTo：与Requires类似，它指定的 Unit 如果退出，会导致当前 Unit 停止运行

Before：如果该字段指定的 Unit 也要启动，那么必须在当前 Unit 之后启动

After：如果该字段指定的 Unit 也要启动，那么必须在当前 Unit 之前启动

Conflicts：这里指定的 Unit 不能与当前 Unit 同时运行

Condition...：当前 Unit 运行必须满足的条件，否则不会运行

Assert...：当前 Unit 运行必须满足的条件，否则会报启动失败

[Install]通常是配置文件的最后一个区块，用来定义如何启动，以及是否开机启动。它的主要字段如下。

WantedBy：它的值是一个或多个 Target，当前 Unit 激活时（enable）符号链接会放入/etc/systemd/system目录下面以 Target 名 + .wants后缀构成的子目录中

RequiredBy：它的值是一个或多个 Target，当前 Unit 激活时，符号链接会放入/etc/systemd/system目录下面以 Target 名 + .required后缀构成的子目录中

Alias：当前 Unit 可用于启动的别名

Also：当前 Unit 激活（enable）时，会被同时激活的其他 Unit

[Service]区块用来 Service 的配置，只有 Service 类型的 Unit 才有这个区块。它的主要字段如下。

Type：定义启动时的进程行为。它有以下几种值。

Type=simple：默认值，执行ExecStart指定的命令，启动主进程

Type=forking：以 fork 方式从父进程创建子进程，创建后父进程会立即退出

Type=oneshot：一次性进程，Systemd 会等当前服务退出，再继续往下执行

Type=dbus：当前服务通过D-Bus启动

Type=notify：当前服务启动完毕，会通知Systemd，再继续往下执行

Type=idle：若有其他任务执行完毕，当前服务才会运行

ExecStart：启动当前服务的命令

ExecStartPre：启动当前服务之前执行的命令

ExecStartPost：启动当前服务之后执行的命令

ExecReload：重启当前服务时执行的命令

ExecStop：停止当前服务时执行的命令

ExecStopPost：停止当其服务之后执行的命令

RestartSec：自动重启当前服务间隔的秒数

Restart：定义何种情况 Systemd 会自动重启当前服务，可能的值包括always（总是重启）、on-success、on-failure、on-abnormal、on-abort、on-watchdog

TimeoutSec：定义 Systemd 停止当前服务之前等待的秒数

Environment：指定环境变量

四、 日志管理

使用 *journalctl* 查看日志

```
# 查看所有日志（默认情况下 ，只保存本次启动的日志）
$ sudo journalctl

# 查看内核日志（不显示应用日志）
$ sudo journalctl -k

# 查看系统本次启动的日志
$ sudo journalctl -b
$ sudo journalctl -b -0

# 查看上一次启动的日志（需更改设置）
$ sudo journalctl -b -1

# 查看指定时间的日志
$ sudo journalctl --since="2012-10-30 18:17:16"
$ sudo journalctl --since "20 min ago"
$ sudo journalctl --since yesterday
$ sudo journalctl --since "2015-01-10" --until "2015-01-11 03:00"
$ sudo journalctl --since 09:00 --until "1 hour ago"

# 显示尾部的最新10行日志
$ sudo journalctl -n

# 显示尾部指定行数的日志
$ sudo journalctl -n 20

# 实时滚动显示最新日志
$ sudo journalctl -f
```

```
# 查看指定服务的日志
$ sudo journalctl /usr/lib/systemd/systemd

# 查看指定进程的日志
$ sudo journalctl _PID=1

# 查看某个路径的脚本的日志
$ sudo journalctl /usr/bin/bash

# 查看指定用户的日志
$ sudo journalctl _UID=33 --since today

# 查看某个 Unit 的日志
$ sudo journalctl -u nginx.service
$ sudo journalctl -u nginx.service --since today

# 实时滚动显示某个 Unit 的最新日志
$ sudo journalctl -u nginx.service -f

# 合并显示多个 Unit 的日志
$ journalctl -u nginx.service -u php-fpm.service --since today

# 查看指定优先级（及其以上级别）的日志，共有8级
# 0: emerg
# 1: alert
# 2: crit
# 3: err
```

```
# 4: warning
# 5: notice
# 6: info
# 7: debug
$ sudo journalctl -p err -b

# 日志默认分页输出，--no-pager 改为正常的标准输出
$ sudo journalctl --no-pager

# 以 JSON 格式（单行）输出
$ sudo journalctl -b -u nginx.service -o json

# 以 JSON 格式（多行）输出，可读性更好
$ sudo journalctl -b -u nginx.serviceqq
 -o json-pretty

# 显示日志占据的硬盘空间
$ sudo journalctl --disk-usage

# 指定日志文件占据的最大空间
$ sudo journalctl --vacuum-size=1G

# 指定日志文件保存多久
$ sudo journalctl --vacuum-time=1years
```