

进程基础 (五)

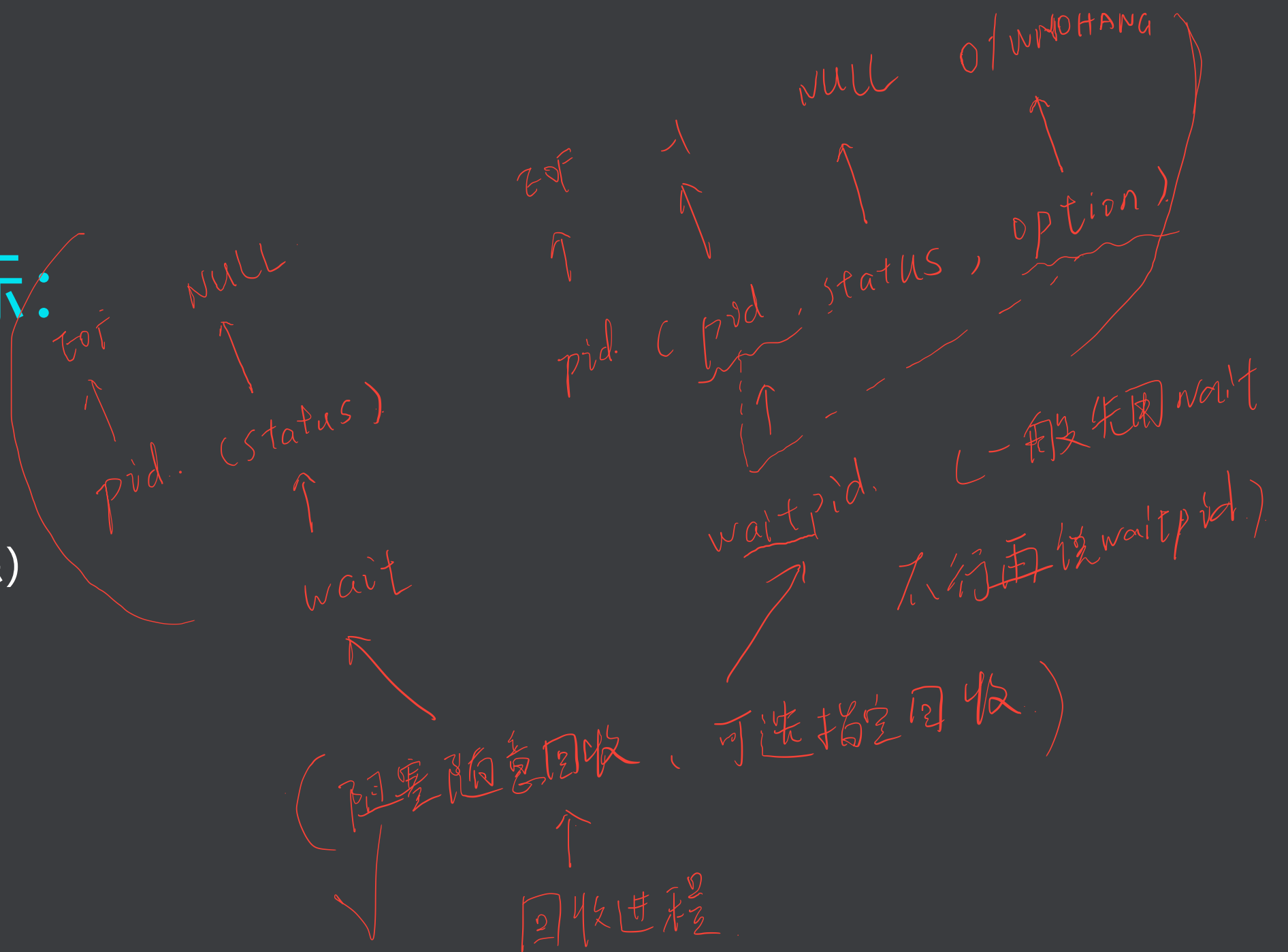
主讲:大海老师

课程目标:

wait (熟练)

waitpid (熟练)

小结



进程回收

子进程结束时由父进程回收

孤儿进程由init进程回收

若没有及时回收会出现僵尸进程

进程回收 – wait

```
#include <sys/wait.h>
```

```
pid_t wait(int *status);
```

↔ status

成功时返回回收的子进程的进程号；失败时返回EOF

若子进程没有结束，父进程一直阻塞

若有多个子进程，哪个先结束就先回收

status 指定保存子进程返回值和结束方式的地址

status为NULL表示直接释放子进程PCB,不接收返回值

进程回收 – wait – 示例

```
int status;  
pid_t pid;  
  
if ((pid = fork()) < 0) {  
    perror( "fork" ); exit(-1);  
}  
else if (pid == 0) {  
    sleep(1); exit(2);  
}  
else {  
    wait(&status); printf( "%x\n" , status);  
}
```

进程返回值和结束方式

子进程通过exit / _exit / return 返回某个值(0-255)

父进程调用wait(&status) 回收

WIFEXITED(status) 判断子进程是否正常结束

WEXITSTATUS(status) 获取子进程返回值

WIFSIGNALED(status) 判断子进程是否被信号结束

WTERMSIG(status) 获取结束子进程的信号类型

(觉到时可man 到这些宏用或来
这儿查).

信号 返回值
信号 结 70
正常结 0



进程回收 – waitpid

```
#include <sys/wait.h>
```

```
pid_t waitpid(pid_t pid, int *status, int option);
```

→ pid status option

- 成功时返回回收的子进程的pid或0；失败时返回EOF
- pid可用于指定回收哪个子进程或任意子进程
- status指定用于保存子进程返回值和结束方式的地址
- option指定回收方式，0 或 WNOHANG

进程回收 – waitpid – 示例

```
waitpid(pid, &status, 0);
```

```
waitpid(pid, &status, WNOHANG);
```

```
waitpid(-1, &status, 0);
```

```
waitpid(-1, &status, WNOHANG);
```


进程小结

wait

waitpid

