本模块是一个基于poll的socket服务器，由以下类组成：

1. sock\_service
2. sock\_module
3. listen\_meta
4. connect\_meta
5. readEvent
6. writeEvent

sock\_service：持有并维护多个sock\_module，主循环中轮询执行sock\_module中的prepare和process，及对多有待检测的socket执行poll操作。

sock\_module：每个sock\_module维护一个监听套接字监听一个port，并创建维护该监听套接字接受的所有链接，每个sock\_module中维护：

1. 一个监听套接字，监听一个端口
2. 一个链接套接字链表，维护监听套接字创建的所有链接
3. 一个线程池，负责执行读写事件

链接套对象有八种状态（其他状态可以陆续添加）：

有读请求，不可读 0

有读请求，可读 1

无读请求，可读 2

无读请求，不可读 3

有写请求，不可写 4

有写请求，可写 5

无写请求，可写 6

无写请求，不可写 7

其中sock\_module的prepare和process操作，prepare操作负责将监听套接字和链接套接字链表中需要poll的socket加入到sock\_service的数组中（哪些套接字需要被poll可以有不同的实现），process操作负责将poll过的socket设置链接对象的状态

本例中只用到四个状态，socket交互采用一请求一回复的模式。

有读请求，不可读 状态0，当处于这个状态时，监听pollin。

有读请求，可读 状态1，当处于这个状态时，不监听，创建readEvent。

有写请求，不可写 状态4，当处于这个状态时，监听pollout。

有写请求，可写 状态5，当处于这个状态时，不监听，创建writeEvent。

状态转换：

1. 链接对象刚建立时，设置为状态0。
2. prepare操作中发现链接对象状态为0时，会为其socket设置侦测pollin标志。
3. sockService的mainloop中进行poll操作。
4. Process操作中看到状态0，且侦测pollin可读时，将状态修改为1，并则创建readEvent添加到线程池中。
5. readEvent中处理完读操作，将链接对象状态设置为4。
6. prepare操作中发现链接对象状态为4时，会为其socket设置侦测pollout标志。
7. sockService的mainloop中进行poll操作。
8. Process操作中看到状态4，且侦测pollout可写时，将状态修改为5，并则创建writeEvent添加到线程池中。
9. writeEvent中完成写event后，设置链接对象状态为0

依次循环直到socket关闭或出错。