Reactor、Worker、TaskWorker的关系

三种角色分别的职责是：

Reactor线程

- 负责维护客户端TCP连接、处理网络IO、处理协议、收发数据
- 完全是**异步非阻塞**的模式
- 全部为C代码，除Start/Shudown事件回调外，不执行任何PHP代码
- 将TCP客户端发来的数据缓冲、拼接、拆分成完整的一个请求数据包
- Reactor以多线程的方式运行

Worker进程

- 接受由Reactor线程投递的请求数据包，并执行PHP回调函数处理数据
- 生成响应数据并发给Reactor线程，由Reactor线程发送给TCP客户端
- 可以是异步非阻塞模式，也可以是同步阻塞模式
- Worker以多进程的方式运行

TaskWorker进程

- 接受由Worker进程通过swoole_server->task/taskwait方法投递的任务
- 处理任务，并将结果数据返回（使用swoole_server->finish）给Worker进程
- 完全是**同步阻塞**模式
- TaskWorker以多进程的方式运行

关系

可以理解为Reactor就是nginx，Worker就是php-fpm。Reactor线程异步并行地处理网络请求，然后再转发给Worker进程中去处理。Reactor和Worker间通过UnixSocket进行通信。

在php-fpm的应用中，经常会将一个任务异步投递到Redis等队列中，并在后台启动一些php进程异步地处理这些任务。Swoole提供的TaskWorker是一套更完整的方案，将任务的投递、队列、php任务处理进程管理合为一体。通过底层提供的API可以非常简单地实现异步任务的处理。另外TaskWorker还可以在任务执行完成后，再返回一个结果反馈到Worker。

Swoole的Reactor、Worker、TaskWorker之间可以紧密的结合起来，提供更高级的使用方式。

一个更通俗的比喻，假设Server就是一个工厂，那Reactor就是销售，接受客户订单。而Worker就是工人，当销售接到订单后，Worker去工作生产出客户要的东西。而TaskWorker可以理解为行政人员，可以帮助Worker干些杂事，让Worker专心工作。

> 底层会为Worker进程、TaskWorker进程分配一个唯一的ID
> 不同的Worker和TaskWorker进程之间可以通过sendMessage接口进行通信