[Workerman3.x版本手册](http://doc3.workerman.net/)

Workerman是一个高性能的socket服务器通讯框架，用于快速开发各种网络应用，包括tcp的、udp的、长连接、短连接应用。   
[在线手册 »](http://doc3.workerman.net/)

[GatewayWorker手册](http://www.workerman.net/gatewaydoc/)

GatewayWorker是基于Workerman开发的一套**TCP长连接**的应用框架，  
实现了单发、群发、广播等接口，内置了mysql类库，  
GatewayWorker分为Gateway进程和Worker进程，天然支持分布式部署，能够支持庞大的连接数（百万甚至千万连接级别的应用）。  
可用于开发IM聊天应用、移动通讯、游戏后台、物联网、智能家居后台等等。

# 工作原理

1、Register、Gateway、BusinessWorker进程启动

2、Gateway、BusinessWorker进程启动后向Register服务进程发起长连接注册自己

3、Register服务收到Gateway的注册后，把所有Gateway的通讯地址保存在内存中

4、Register服务收到BusinessWorker的注册后，把内存中所有的Gateway的通讯地址发给BusinessWorker

5、BusinessWorker进程得到所有的Gateway内部通讯地址后尝试连接Gateway

6、如果运行过程中有新的Gateway服务注册到Register（一般是分布式部署加机器），则将新的Gateway内部通讯地址列表将广播给所有BusinessWorker，BusinessWorker收到后建立连接

7、如果有Gateway下线，则Register服务会收到通知，会将对应的内部通讯地址删除，然后广播新的内部通讯地址列表给所有BusinessWorker，BusinessWorker不再连接下线的Gateway

8、至此Gateway与BusinessWorker通过Register已经建立起长连接

9、客户端的事件及数据全部由Gateway转发给BusinessWorker处理，BusinessWorker默认调用Events.php中的onConnect onMessage onClose处理业务逻辑。

10、BusinessWorker的业务逻辑入口全部在Events.php中，包括onWorkerStart进程启动事件(进程事件)、onConnect连接事件(客户端事件)、onMessage消息事件（客户端事件）、onClose连接关闭事件（客户端事件）、onWorkerStop进程退出事件（进程事件）

## 重要的事情说三遍

业务开发只需要关注 Applications/项目/Events.php一个文件即可。  
业务开发只需要关注 Applications/项目/Events.php一个文件即可。  
业务开发只需要关注 Applications/项目/Events.php一个文件即可。

开放的端口及协议在start\_gateway.php中更改。参见[Gateway类的使用](http://www.workerman.net/gatewaydoc/getting-started/gateway-worker-development/gateway.html)一章。

## 注意

1、服务端启动成功，但是无法通讯，请检查服务器防火墙。

2、客户端与服务端要能保持正常通讯，需要保证客户端与服务端的通讯协议是一致的。比如服务端是websocket协议，客户端也要使用websocket协议来通讯(js里用ws=new WebSocket('ws://ip:port');来连接)，而浏览器地址栏(http协议)直接访问websocket服务肯定是不通的。

3、长连接应用切记需要开启应用层心跳(GatewayWorker提供了设置，[参见心跳检测](http://www.workerman.net/gatewaydoc/getting-started/gateway-worker-development/heartbeat.html))，心跳间隔20-30秒最佳，为了避免长连接应为长时间不通讯被节点防火墙断开。

## 目录结构

.

├── Applications // 这里是所有开发者应用项目

│ └── YourApp // 其中一个项目目录，目录名可以自定义

│ ├── Events.php // 开发者只需要关注这个文件

│ ├── start\_gateway.php // gateway进程启动脚本，包括端口号等设置

│ ├── start\_businessworker.php // businessWorker进程启动脚本

│ └── start\_register.php // 注册服务启动脚本

│

├── start.php // 全局启动脚本，此脚本会依次加载Applications/项目/start\_\*.php启动脚本

│

└── vendor // GatewayWorker框架和Workerman框架源码目录，此目录开发者不用关心

## start\_gateway.php

start\_gateway.php为gateway进程启动脚本，主要定义了客户端连接的端口号、协议等信息，具体参见 Gateway类的使用一节。

客户端连接的就是start\_gateway.php中初始化的Gateway端口。

## start\_businessworker.php

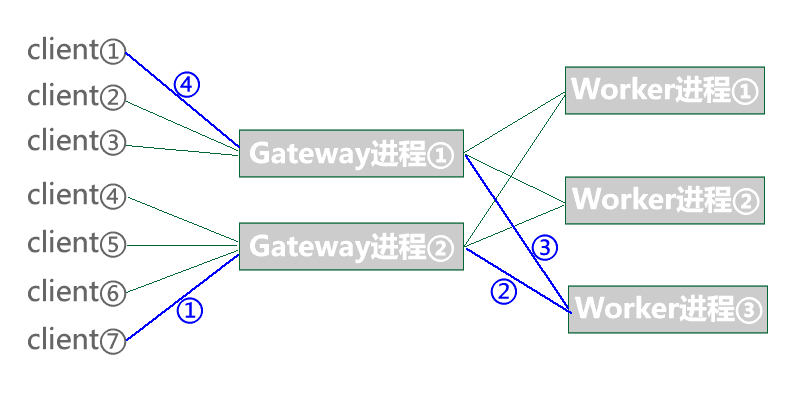
start\_businessworker.php为businessWorker进程启动脚本，也即是调用Events.php的业务处理进程，具体参见 BusinessWorker类的使用一节。

## start\_register.php

start\_register.php为注册服务启动脚本，用于协调GatewayWorker集群内部Gateway与Worker的通信，参见Register类使用一节。

注意：客户端不要连接Register服务端口，客户端应该连接Gateway端口

### Gateway/Worker 的进程模型



**特点：**

从图上我们可以看出Gateway负责接收客户端的连接以及连接上的数据，然后Worker接收Gateway发来的数据做处理，然后再经由Gateway把结果转发给其它客户端。每个客户端都有很多的路由到达另外一个客户端，例如client⑦与client①可以经由蓝色路径完成数据通讯

**优点：**

1、可以方便的实现客户端之间的通讯

2、Gateway与Worker之间是基于socket长连接通讯，也就是说Gateway、Worker可以部署在不同的服务器上，非常容易实现分布式部署，扩容服务器

3、Gateway进程只负责网络IO，业务实现都在Worker进程上，可以reload Worker进程，实现在不影响用户的情况下完成代码热更新。

**适用范围：**

适用于客户端与客户端需要实时通讯的项目。