河北师范大学 软件学院
Software College of Hebei Normal University

Linux平台PHP服务端开发——

第八讲 异步IO与Swoole扩展

目录

异步IO与Swoole介绍

Swoole扩展介绍与安装

使用Swoole编写异步程序

1

异步IO与Swoole介绍

同步的过程与问题

- 以烧水为例：同步过程就是一直等待水烧开，沏到壶里。中间不做其他事情。
- 对于不需要即时返回结果的操作，同步过程加大了网络延迟，降低了网站整体的响应速度。
- 同步过程在很多任务上，效率很低。

异步的概念

- 异步过程：还以烧水为例，烧水的过程中，去做其他事情，等水烧开了发出提示，这时候你可以终端手头的工作先处理水烧开了这个事件。

- 异步提高的是整体的效率，避免不必要的资源浪费。

- 再考虑一个问题：银行办理业务过去都是同步的过程，一个窗口只能等客户业务完成才处理下一个客户的业务。而如果是异步的方式，当客户在填表格等操作时，这时候业务员是空闲的，这时候可以先处理下一个客户的业务，如果上一个客户表格填写完成就发出一个信号告诉业务员响应这个请求。所以现在银行办理业务都是要求客户提前填写表格，提高业务处理效率。

深入理解异步

- 异步是靠多个同步的操作组合起来的一种模式。
- 一些异步IO扩展库使用的是多进程多线程模式实现的。这种方式就是多线程同步模型实现异步。
- 对任何任务拆分来说总会到达一个不可分解，必须是同步的操作的最小任务。
- 异步在编程上的形式往往使用事件驱动以及回调函数的形式来处理。

2

Swoole安装与基本使用

Swoole基本介绍

- Swoole是PHP的异步、并行、高性能网络通信引擎，使用纯C语言编写。Swoole底层内置了异步非阻塞、多线程的网络IO服务器。PHP程序员仅需处理事件回调即可，无需关心底层。
- Swoole内置了Http/WebSocket服务器端/客户端、Http2.0服务器端/客户端。
- Swoole是以PHP扩展的形式集成到PHP的。
- 1.8.7+版本已完全支持PHP7，2.0.12+版本不再支持PHP5。
- Swoole是开源的，授权协议是Apache2.0。企业和个人开发者均可免费使用Swoole的代码，并且在Swoole之上所作的修改可用于商业产品，无需开源（注：必须保留原作者的版权声明）。
- Swoole的处理过程很像NodeJS，使用回调函数的形式，采用异步操作。但不同的是，Swoole既支持异步，也支持同步。

编译安装Swoole

- 国内git镜像：https://gitee.com/swoole/swoole
- 安装：

  下载swoole源代码后，使用PHP扩展编译的步骤操作即可。

  phpize

  ./configure –with-php-config=PHP_INSTALL_DIR/bin/php-config

  make install
- 安装完成后，在php.ini中加入：extension=swoole.so；注意，在PHP7.2中，不需要加入.so
- 然后重启web服务： webrun restart。

代码示例

```php
<?php

$ws = new swoole_http_server('localhost', 3456);

$ws->on('request',function($request, $response){
    $response->write('Swoole is cool');
});

$ws->start();

```

Swoole的处理模式

- Swoole使用异步模式进行请求处理，这和同步方式的PHP程序是不同的。 Swoole使用事件绑定回调函数的方式，当一个事件发生就会执行相对应的回调函数，使用事件驱动。
- 而LNMP这种模式下，PHP程序生存周期通常很短，执行完毕后就会结束。而使用Swoole编写的程序是一直运行的。Swoole本身就是Web服务器，可以直接提供服务。
- Swoole提供的主要模块与事件：

功能模块	swoole_server	支持TCP，UDP，TCP6，UDP6等协议
	swoole_http_server	继承swoole_server实现HTTP，HTTP2，HTTPS协议
	websocket	实现websocket协议，可用于推送服务
事件	onStart; onRecieve; onClose; onConnect; onShutdown……	名称可看出大概含义
	onRequest	swoole_http_server支持的事件
	onMessage; onHandshake	websocket服务需要用到的事件

把Swoole程序变成守护进程

- 之前的演示代码是在终端直接执行，如果shell结束运行，则程序也退出。这时候需要把Swoole的程序变成守护进程。
- 两种方式：
  - 1. 使用PCNTL+POSIX扩展编写守护进程的方式。
  - 2. 使用Swoole的配置选项：

```
$sw_http_server->set([
        'daemonize' => 1
]);
```

- 完整代码参考示例文件。

3

Swoole静态文件服务器

静态文件服务器

- 返回静态文件，不做动态处理。
- 可用于静态网站。
- 大规模网站，会单独有一个静态文件分发服务器。对于变动不频繁的页面部署在一个单独的服务上，减轻整体系统的负载压力。

使用Swoole如何实现

- 使用Swoole实现静态文件服务器很简单，Swoole提供了document_root配置选项。
- 通过两个选项组合即可实现：

```
$sw_http_serv->set([
    'document_root' => '/var/www/swoole_static',
    'enable_static_handler' => true
]);
```

- document_root配置静态文件路径，enable_static_handler设置允许静态文件请求处理。
- 这两个选项设置后，如果在设置的路径存在请求的文件则直接返回文件数据。
- 如果存在文件，就不会再触发onRequest事件。

文件未发现

- 如果文件未发现，则会触发onRequest事件处理请求。
- 可以在onRequest事件回调函数中给出错误提示：

```
$sw_http_serv->on('request',function($request,$response){
    $response->write("Error: file not found");
});
```

写成守护进程

- 作为一个静态文件服务程序，运行时要在后台运行，否则shell退出，就会随之退出。

- 使用配置创建守护进程：

```
$sw_http_serv->set([
    'document_root' => '/var/www/swoole_static',
    'enable_static_handler' => true,
    'daemonize' => 1
]);
```

- 最后一行加入‘daemonize’=>1即可。