

Linux平台PHP服务端开发——

第九讲 Swoole与Websocket协议

目录

HTTP协议的特点与不足

Websocket协议

基于Websocket实现推送

1

HTTP协议的特点与不足

HTTP无状态

- HTTP是无状态协议，客户端的每次请求之间没有关系。
- HTTP服务端并不清楚客户端是什么情况，客户端发起请求，服务端就处理请求返回数据。

HTTP连接

- HTTP/1.0使用短连接，数据传输完成就会关闭连接。
- HTTP/1.1默认使用长连接，响应头部信息会有Connection: keep-alive 。连接在数据传输完成后会维持一段时间。

HTTP与服务端推送

- HTTP无法实现服务端推送，请求是由客户端发起的。服务端无法主动向客户端发送数据。

2

Websocket协议

WebSocket协议基本介绍

- WebSocket协议解决了基于Web的服务端推送问题。
- WebSocket基于TCP协议，但是要依靠HTTP协议建立连接。使用HTTP实现握手操作之后，就会建立WebSocket连接。
- 2011年WebSocket成为国际标准。

Websocket协议流程简介

- HTTP请求头部信息：
GET /chat HTTP/1.1
Host: server.example.com
Upgrade: websocket
Connection: Upgrade
Sec-WebSocket-Key: dGhlIHNhbXBsZSBub25jZQ==
Origin: http://example.com
Sec-WebSocket-Protocol: chat, superchat
Sec-WebSocket-Version: 13
- Web服务器在遇到Upgrade : websocket和Connection : Upgrade后会把连接协议升级成基于TCP的Websocket协议。建立连接的过程称为Handshake（握手操作）。简单来讲服务端根据Sec-Websocket-Key字段的值加上一个固定的GUID值并进行sha1与based64编码后返回给客户端。

Swoole编写Websocket服务

- 使用代码示例来理解如何用Swoole实现Websocket服务：

```
class phpWebSocket {
    private $server;
    public function __construct() {
        $this->server = new swoole_websocket_server('localhost',9876);
        $this->server->on('open',function($server, $req) {
            echo "websocket open by " . $req->fd . "\n";
        });
        $this->server->on('message',function($server,$req) {
            $server->push($req->fd,mt_rand(0,1024));
        });
        $this->server->on('close',function($server, $fd) {
            echo $fd . " closed\n";
        });
        $this->server->start();
    }
}
new phpWebSocket();
```

Swoole编写Websocket客户端

- 客户端每隔100ms向服务器发送一条消息：

```
$cli = new swoole_http_client('127.0.0.1',9876);
```

//onMessage事件是必须的

```
$cli->on('message',function($cli,$frame){
```

```
    echo "Received: $frame->data\n";
```

```
    usleep(100000);
```

```
    $cli->push("Send: " . mt_rand(1000,10000));
```

```
});
```

```
$cli->upgrade('/',function($cli){
```

```
    $cli->push( time() );
```

```
});
```

3

基于WebSocket实现推送

Swoole实现每秒推送一条消息

- 使用Swoole的Websocket服务可以实现服务端消息推送。
- 每秒推送一条消息要考虑的问题：
 - Swoole采用事件驱动的异步处理方式，**需要有一个触发事件。**
 - 不能使用循环与sleep的方式，这会把Swoole变成同步模式，并且每次只能处理一个请求，只有等一个连接关闭，才可以有新的请求进行处理。
 - Swoole多进程分配请求导致连接信息不能由进程使用变量进行存储，需要一个公共的缓存空间进行存储。
 - 由于是服务程序，需要创建守护进程。
 - 进程要捕获SIGTERM信号进行后续处理后再退出。
- 实现方案：
 - 使用一个Websocket客户端连接到Websocket服务端，客户端每秒发送一条数据，服务端进行转发。由客户端触发消息推送事件。
 - 使用Memcached缓存服务存储连接信息。

客户端触发推送事件

- 代码示例：

```
<?php
```

```
$cli = new swoole_http_client('127.0.0.1',4567);
```

```
$cli->on('message',function($cli,$frame){
```

```
    //由于websocket客户端必须要onMessage事件，但是回调函数什么也不做  
});
```

```
$cli->upgrade('/push_client/phpswoolewebsocket',function($cli){
```

```
    while(true) {
```

```
        $cli->push(strftime("%Y.%m.%d %H:%M:%S"));
```

```
        sleep(1);
```

```
    }
```

```
});
```

消息推送服务端：onOpen事件

- 代码示例：

```
public function on_open($server, $req) {  
    $tmp_key = $this->conn_head . $req->fd;  
    //简单实现方案，通过检测PATH_INFO信息确定是推送事件触发客户端的连接  
    //PATH_INFO在此处是以key的作用来使用的  
    if ($req->server['path_info'] == '/push_client/phpswoolewebsocket') {  
        $tmp_key = $this->client_index;  
    }  
    //缓存连接  
    $this->mcache->set($tmp_key, $req->fd);  
}
```

消息推送服务端：onMessage事件

- 代码示例：

```
public function on_message($server, $req) {  
    //获取保存的推送客户端连接与当前连接对比，确定是推送客户端发送数据则转发  
    if ($req->fd == $this->mcache->get($this->client_index)) {  
        //start push  
        if(!empty($req->data)) {  
            $keys = $this->mcache->getAllKeys();  
            $this->mcache->getDelayed($keys);  
            $key_vals = $this->mcache->fetchAll();  
            foreach ($key_vals as $kv) {  
                $server->push($kv['value'],$req->data);  
            }  
        }  
    }  
}
```


消息推送服务端：onClose事件和onShutdown事件

- onClose事件代码示例：

```
public function on_close($server,$fd) {  
    $this->mcache->delete($this->conn_head.$fd,0);  
}
```

- onShutdown 事件代码示例（捕获SIGTERM信号并处理）：

```
public function on_shutdown($server) {  
    $this->mcache->deleteMulti($this->mcache->getAllKeys());  
    $this->mcache->quit();  
}
```