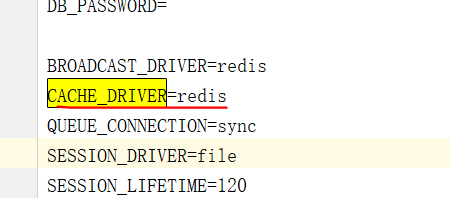
## 一．Redis

### 安装

composer require predis/predis

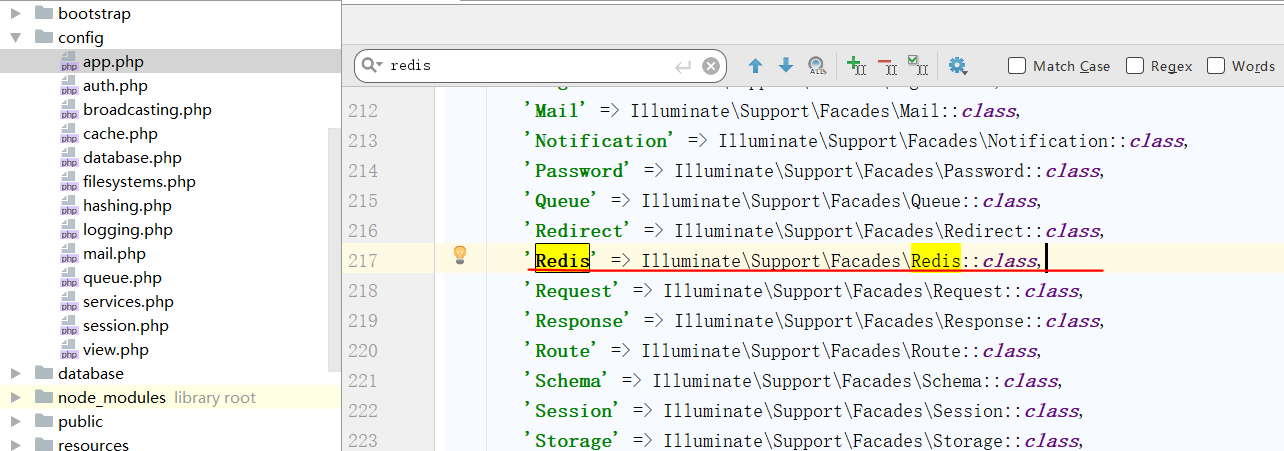
### 配置驱动信息

Laravel默认驱动是file我们改成redis



### 3.

并注释 config/app.php 中的 alias 中 Redis 一行防止 Redis 命名空间冲突



## 二．事件

官方文档：

<https://xueyuanjun.com/post/19508.html#bkmrk-%E4%BA%8B%E4%BB%B6%E7%9B%91%E5%90%AC%E5%99%A8%E9%98%9F%E5%88%97>

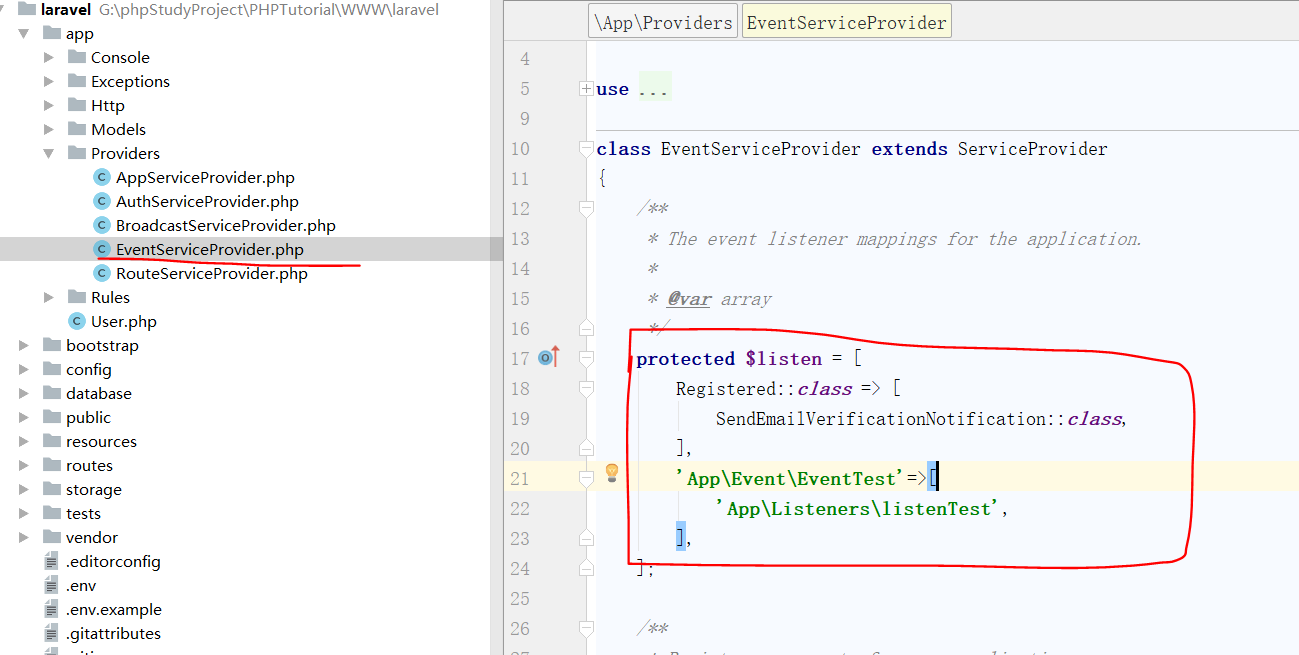
<https://learnku.com/laravel/t/28421>

<https://www.cnblogs.com/fps2tao/p/9681455.html>

场景：用户注册的时候发短信，邮件给用户

流程：

Controller-🡪触发事件类->事件类触发监听器



Php artisan event:generate 自动生成

**定义事件**

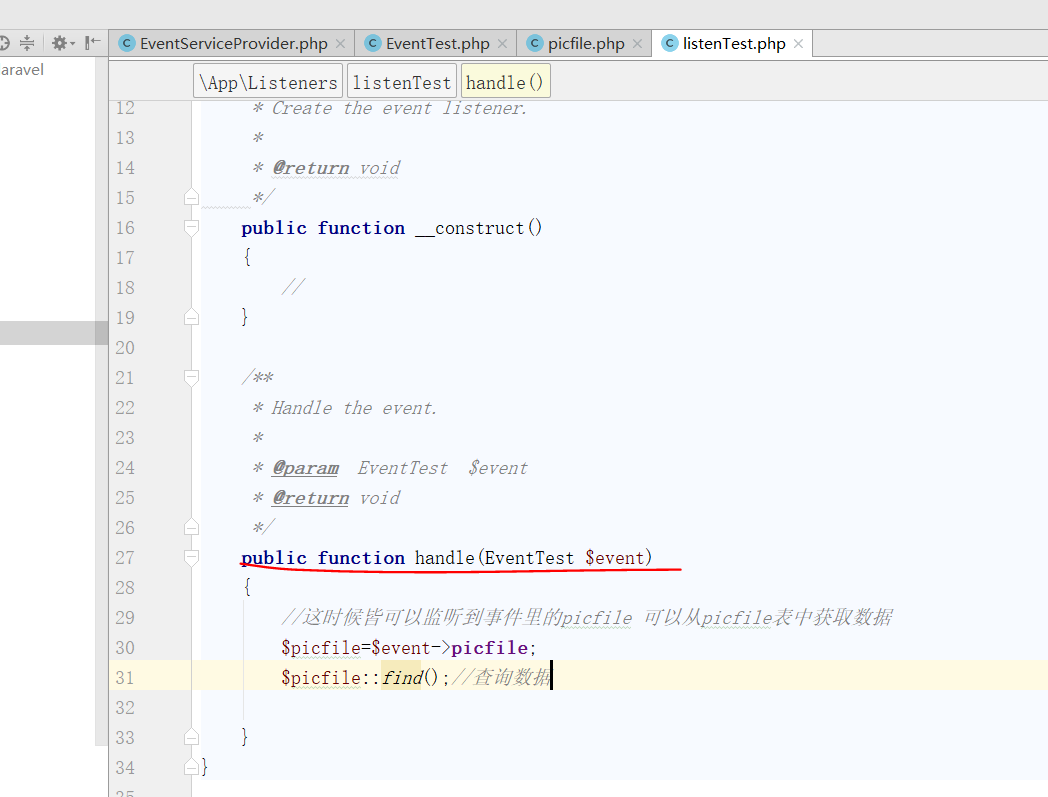


**定义监听器**

监听器在 handle 方法中接收事件实例

**停止事件继续往下传播**

有时候，你希望停止事件被传播到其它监听器，你可以通过从监听器的 handle 方法中返回 false 来实现

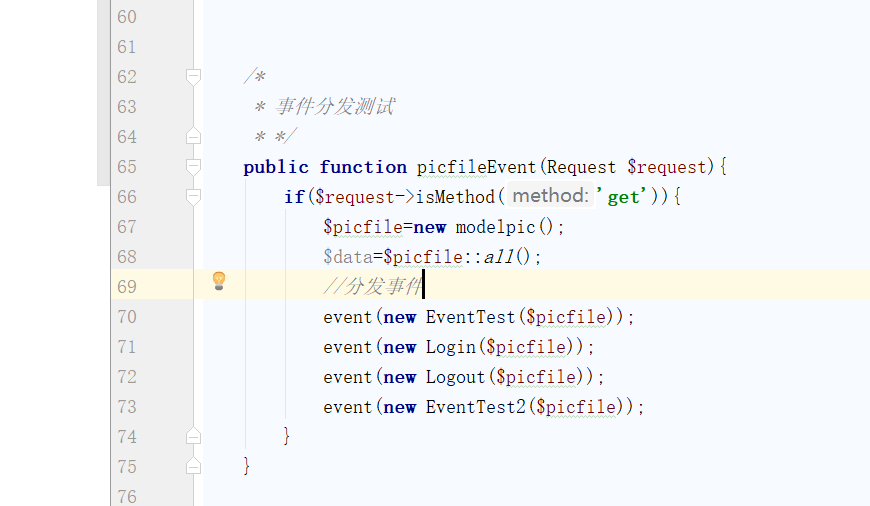


**事件监听器队列**

操作之前要配置好队列

**事件分发**

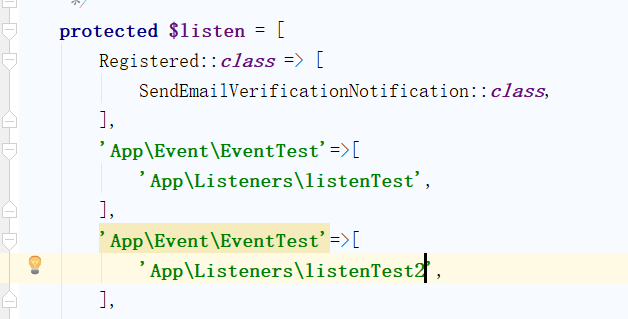
即什么时候触发监听器



**事件订阅者**

很多时候同一事件会被不同监听器监听

像这样



这样会让我们多生成许多监听器功能差不多

因此我们是有事件订阅者，用来在一个监听器中监听器中区分不同事件

**定义订阅类：**

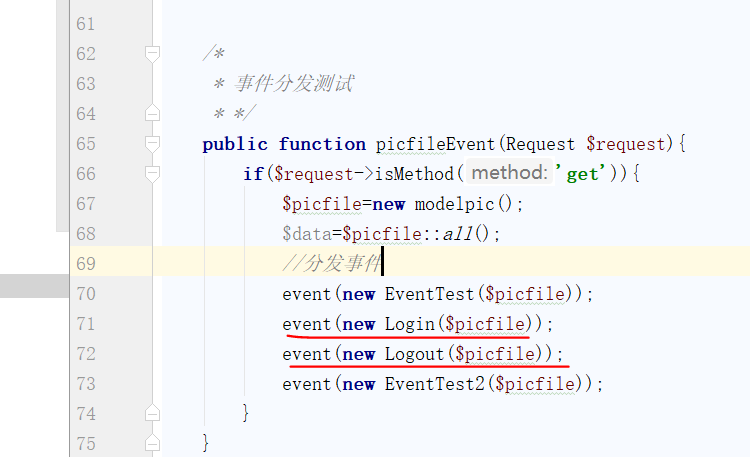
**<?php**/\*\*  
 \* Created by PhpStorm.  
 \* User: SWESWE  
 \* Date: 2020/2/19  
 \* Time: 10:40  
 \*/  
  
**namespace** App\Listeners;  
  
  
**class** EventSubscriber  
{  
 /\*  
 \* 事件订阅者  
 \* \*/  
 /\*\*  
 \* 处理用户登录事件.  
 \* **@translator** laravelacademy.org  
 \*/  
 **public function** onUserLogin($event) {  
 **echo"事件订阅者登录"**;  
 **echo'<br>'**;  
  
 }  
 /\*\*  
 \* 处理用户退出事件.  
 \*/  
 **public function** onUserLogout($event) {  
 **echo"事件订阅者登出"**;  
 **echo'<br>'**;  
 }  
  
 /\*\*  
 \* 为订阅者注册监听器.  
 \*直接用方法监听事件不再需要新建一个监听器类来监听  
 \* **@param** \*/  
 **public function** subscribe($events)  
 {  
 $events->listen(  
 **'App\Event\Login'**,  
 **'App\Listeners\EventSubscriber@onUserLogin'** );  
  
 $events->listen(  
 **'App\Event\Logout'**,  
 **'App\Listeners\EventSubscriber@onUserLogout'** );  
 }



**注册订阅者**

  
  
}

**分发事件：**



## 广播

官方文档：

<https://xueyuanjun.com/post/19505.html>

<https://qianjinyike.com/laravel-%E5%86%85%E7%BD%AE%E8%81%8A%E5%A4%A9%E5%AE%A4/>

应用场景

实时聊天

用户不用刷新页面就可以更新其他用户的评论

**广播配置文件**

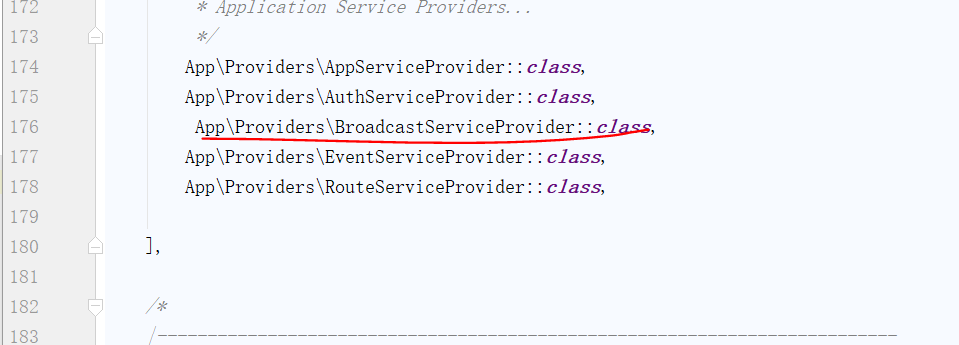
**流程参考：**

<https://qianjinyike.com/laravel-%E5%85%AC%E6%9C%89%E5%B9%BF%E6%92%AD/>

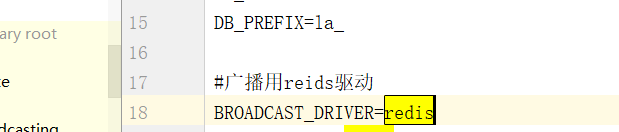
config/broadcasting.php

Laravel 开箱支持多种广播驱动：[Pusher](https://pusher.com/)、[Redis](https://xueyuanjun.com/post/19525.html)以及一个服务于本地开发和调试的 log 驱动。此外，还提供了一个 null 驱动用于完全禁止事件广播

### 1.去掉注释 config/app.php provider 中的BroadcastServiceProvider



2.在 .env 中配置驱动 BROADCAST\_DRIVER=redis



### 安装redis

安装广播之前要安装redis

参考：菜鸟教程windwos 下安装redis

<https://www.runoob.com/redis/redis-install.html>

运行命令：

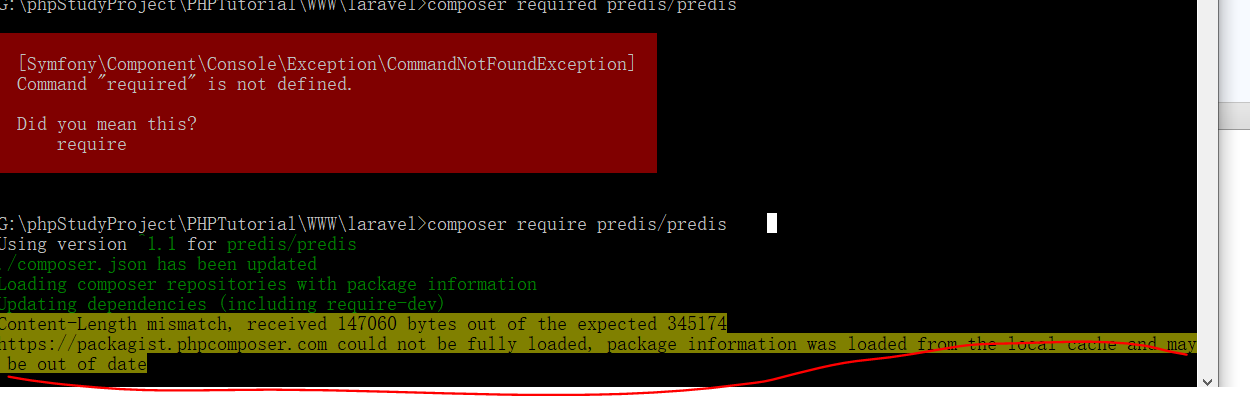
切到redis目录下

redis-server.exe redis.windows.conf

windows安装redis桌面应用 redis desktop manager

composer required predis/predis//composer安装有问题安装不了

出现下面问题

  
运行composer config -g repo.packagist composer <https://packagist.phpcomposer.com>后再次运行上面命令

在开始介绍广播事件之前，还需要配置并运行一个[队列监听器](https://xueyuanjun.com/post/19516.html)。所有事件广播都通过队列任

### **初始化 websocket 客户端**

import Echo from "laravel-echo"

window.io = require('socket.io-client');

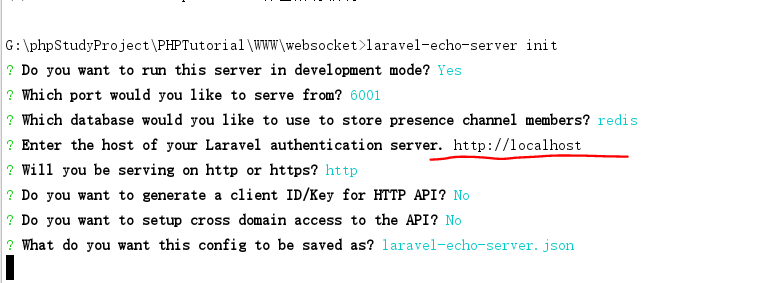
window.Echo = new Echo({

broadcaster: 'socket.io',

host: window.location.hostname + ':6001'

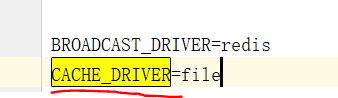
});

创建事件



画红线的要配成自己的域名

配置公有频道：

画线的可以配redis也可以配file,但BROADCAST\_DRIVER

要配成redis驱动

npm install --save socket.io-client | echo 'websocket 客户端'

npm install --save laravel-echo | echo 'websocket 客户端封装'

npm install -g laravel-echo-server | echo 'websocket 服务端'

npm install | echo '安装所有其他依赖'

下面三个严格执行

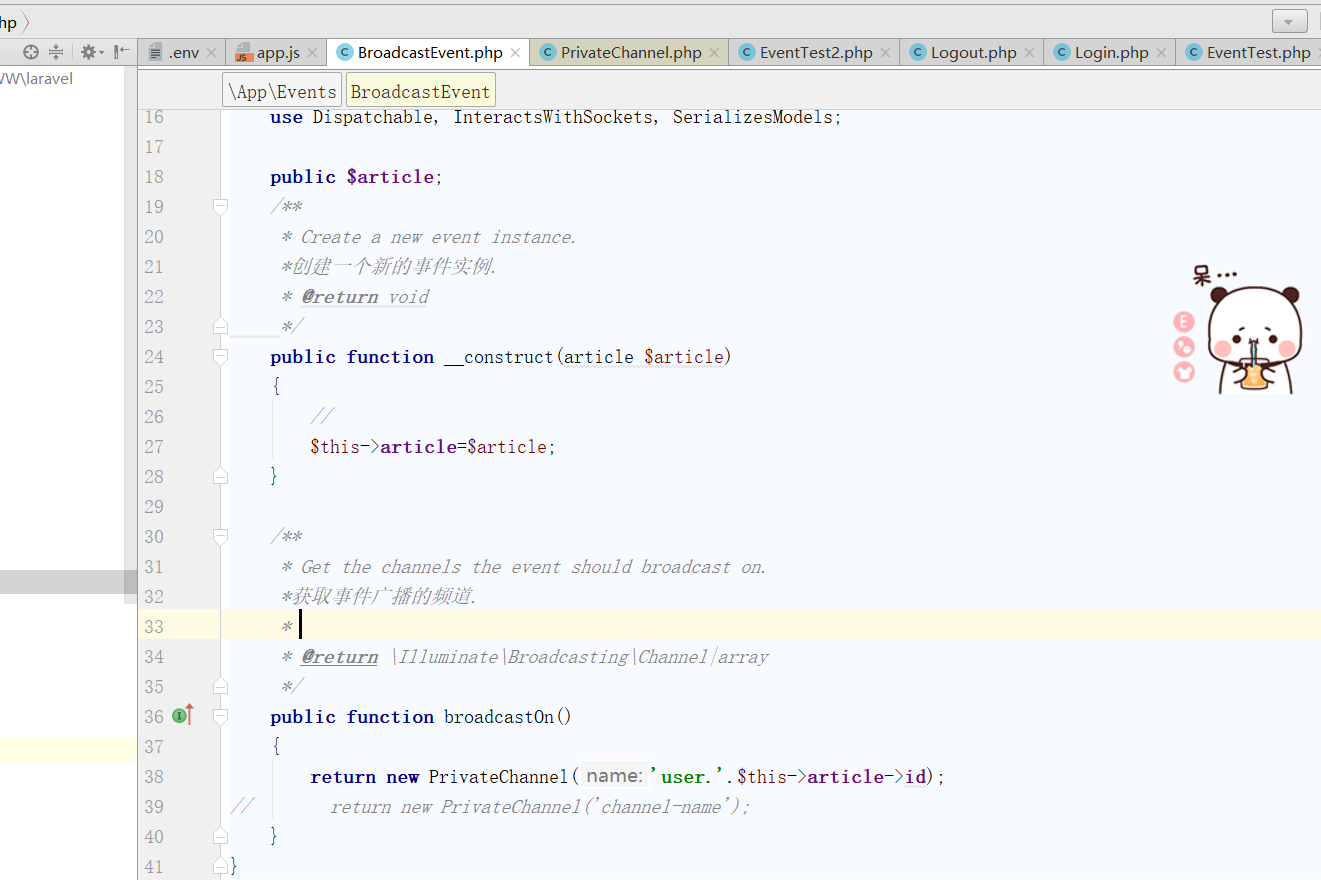
npm run watch | echo '监控文件变化编译前端资源'

laravel-echo-server init | echo '初始化 websocket 服务端'

laravel-echo-server start | echo '启动 websocket 服务端'

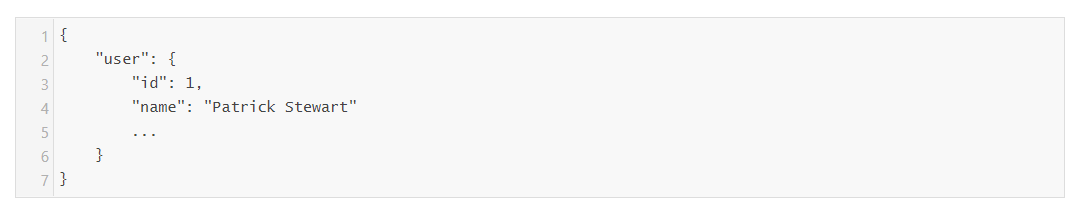
务来完成以便应用的响应时间不受影响。

### 3.定义广播事件

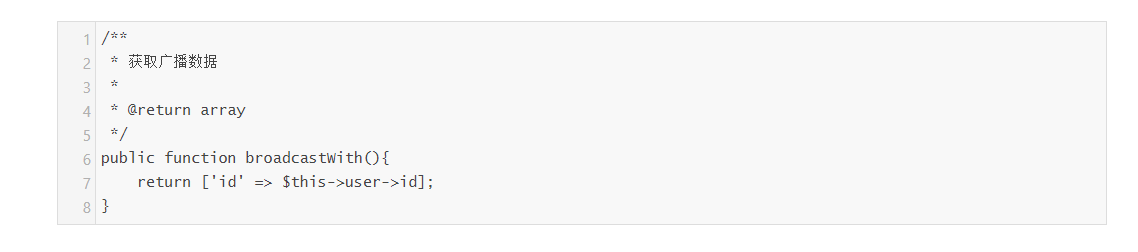


#### 广播数据

如果某个事件被广播，其所有的 public 属性都会按照事件负载（payload）自动序列化和广播，从而允许你从 JavaScript 中访问所有 public 数据，举个例子，如果你的事件有一个单独的包含 Eloquent 模型的 $user 属性，广播负载定义如下：

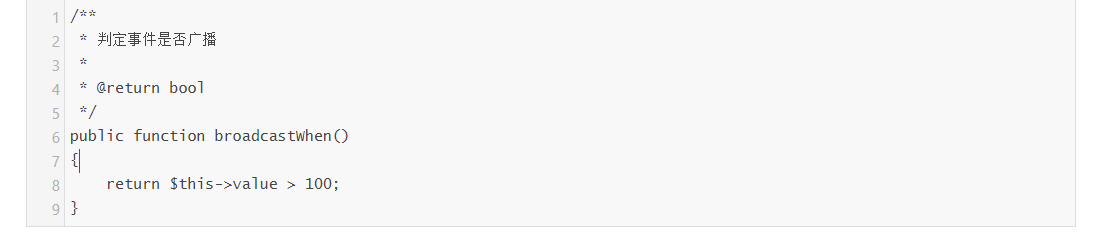


不过，如果你希望对广播负载有更加细粒度的控制，可以添加 broadcastWith 方法到事件，该方法会返回你想要通过事件广播的数组数据：



#### 广播条件

有时候你想要在指定条件为 true 的前提下才广播事件，可以通过添加 broadcastWhen 方法到事件类来定义这些条件：



Controller—分发事件—》

事件广播类中绑定频道

在routes/channels.php 中授权频道

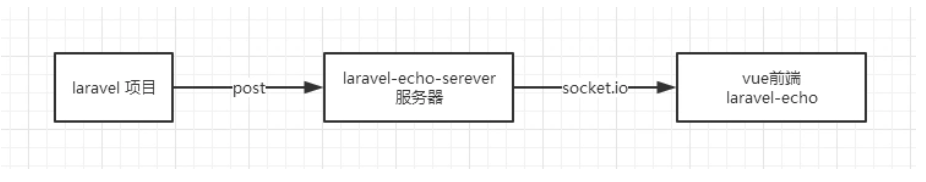
Broadcast::channel('order.{$orderId}', function ($user, $orderId) {

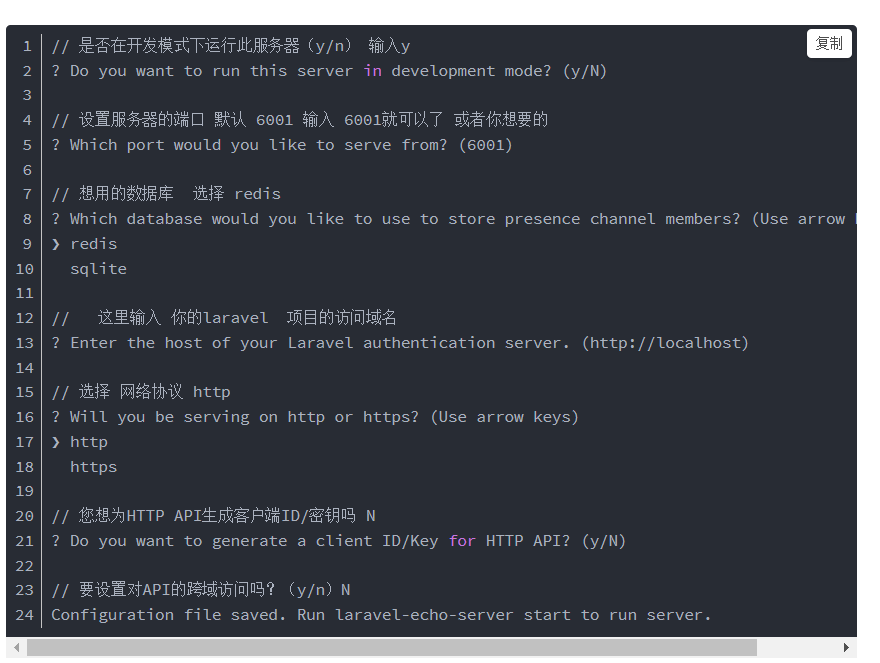
return $user->id === Order::findOrNew($orderId)->user\_id;

});

使用 Laravel Echo 来监听事件广播

**整体架构：**





**Phpstorm快捷键**

CTRL + SHIFT + N 文件名查找

调试数据库中的数据。通过使用 Laravel artisan 内建的 php artisan [tinker](https://link.jianshu.com/?t=http://laravelacademy.org/tags/tinker)

## 三．广播

### 3.1公共频道

#### 配置：

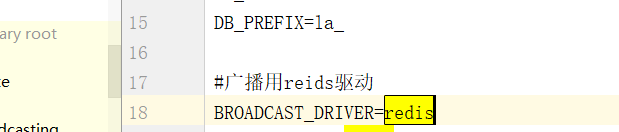
##### 配置广播驱动

//安装redis

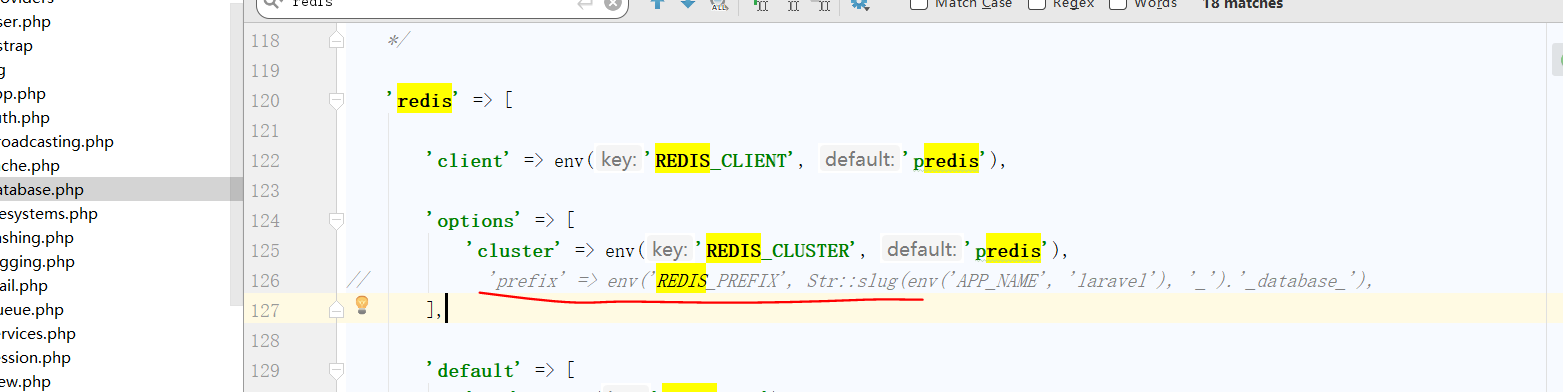
#### composer require predis/predis

必须安装

在 .env 中配置驱动 BROADCAST\_DRIVER=redis



取消redis前缀 config/database



否则会在频道前加上前缀



##### 2．安装

###### 2.1安装redis

安装广播之前要安装redis

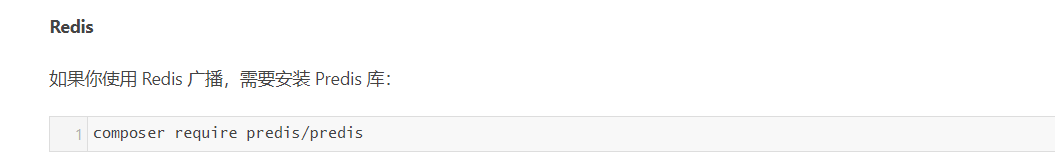
参考：菜鸟教程windwos 下安装redis

<https://www.runoob.com/redis/redis-install.html>

运行命令：

切到redis目录下

redis-server.exe redis.windows.conf

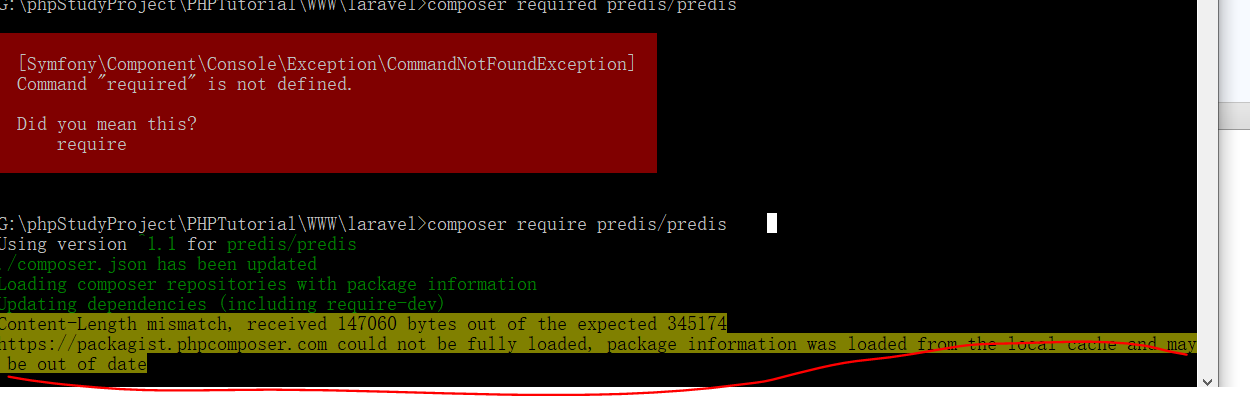


windows安装redis桌面应用 redis desktop manager

composer dumpautoload

composer require predis/predis//composer安装有问题安装不了

出现下面问题

  
运行composer config -g repo.packagist composer <https://packagist.phpcomposer.com>后再次运行上面命令composer update

composer config -l

###### 2.2npm安装

npm install --save socket.io-client | echo 'websocket 客户端'

npm install --save laravel-echo | echo 'websocket 客户端封装'

npm install -g laravel-echo-server | echo 'websocket 服务端'

npm install | echo '安装所有其他依赖'

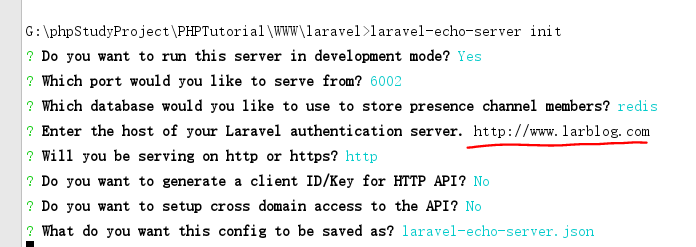
###### 2.3运行

npm run watch | echo '监控文件变化编译前端资源'

laravel-echo-server init | echo '初始化 websocket 服务端'

laravel-echo-server start | echo '启动 websocket 服务端'

运行laravel-echo-server init 的时候



画线部分一般为自己域名

3.定义事件

Php artisan make:event free

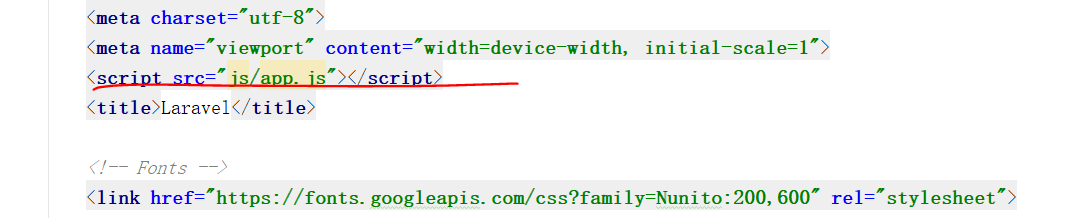
**<?php  
  
namespace** App\Events;  
  
**use** Illuminate\Broadcasting\Channel;  
**use** Illuminate\Queue\SerializesModels;  
**use** Illuminate\Broadcasting\PrivateChannel;  
**use** Illuminate\Broadcasting\PresenceChannel;  
**use** Illuminate\Foundation\Events\Dispatchable;  
**use** Illuminate\Broadcasting\InteractsWithSockets;  
**use** Illuminate\Contracts\Broadcasting\ShouldBroadcast;  
  
**class** free **implements** ShouldBroadcast *// 1. 事件是要广播出去的,*这边一定要**继承**接口

patchable, InteractsWithSockets, SerializesModels;  
 **public $msg**;  
 */\*\*  
 \* Create a new event instance.  
 \*  
 \** ***@return*** *void  
 \*/* **public function** \_\_construct($msg) *// 2. 广播出去的内容* {  
 *//* $this->**msg**=$msg;  
 }  
  
 */\*\*  
 \* Get the channels the event should broadcast on.  
 \*  
 \** ***@return*** *\Illuminate\Broadcasting\Channel|array  
 \*/* **public function** broadcastOn() *// 3. 对哪些频道进行广播* {  
*// return new PrivateChannel('channel-name');  
// return [new Channel('countryside'), new Channel('a')];* **return new** Channel(**'countryside'**);  
 }  
  
}

##### 定义客户端监听广播



并在welcome页面中引入



注销redis前缀否则会在频道前加上前缀



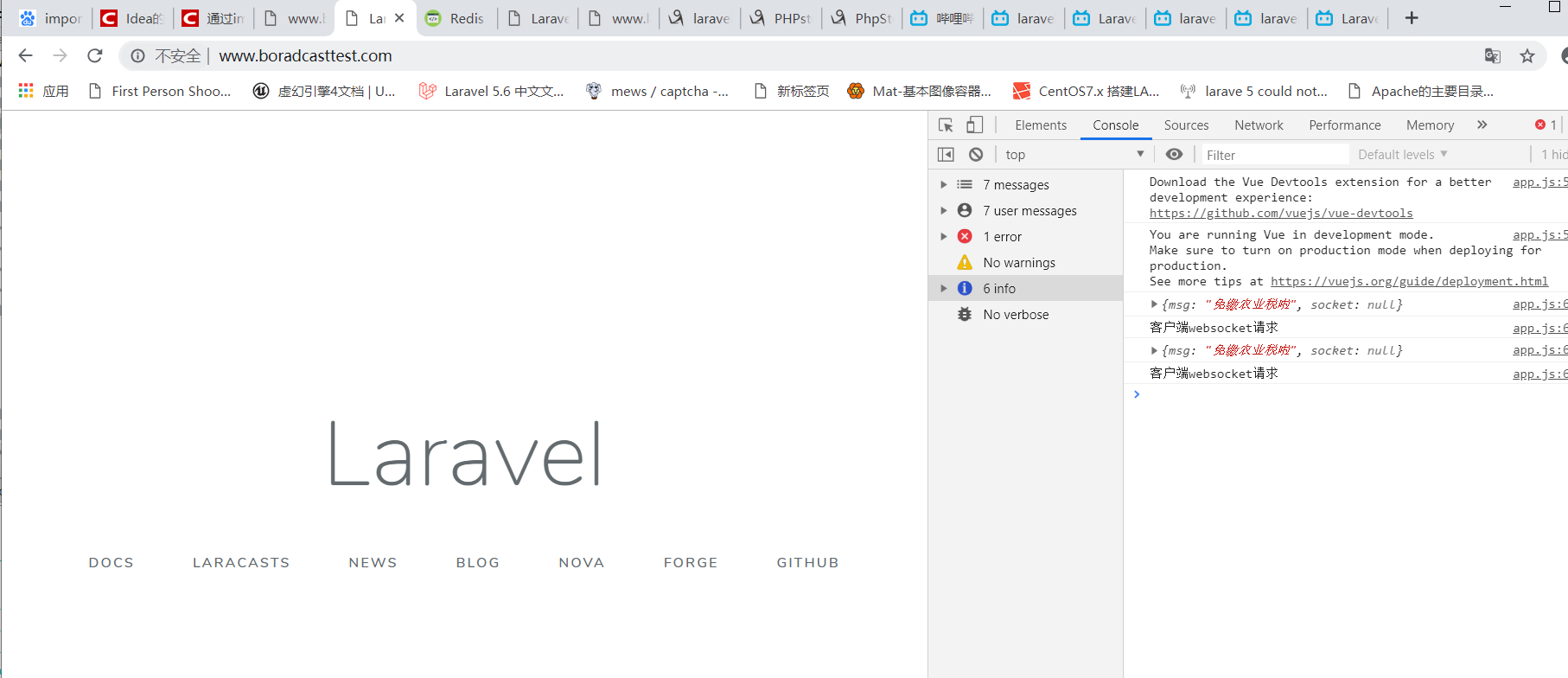
##### 定义路由

Route::*get*(**'/event'**, **function** () {  
 event(**new** \App\Events\free(**'免缴农业税啦'**));  
});

##### 测试

访问<http://www.boradcasttest.com/event>路由

再访问<http://www.boradcasttest.com> 发现控制台打印出消息

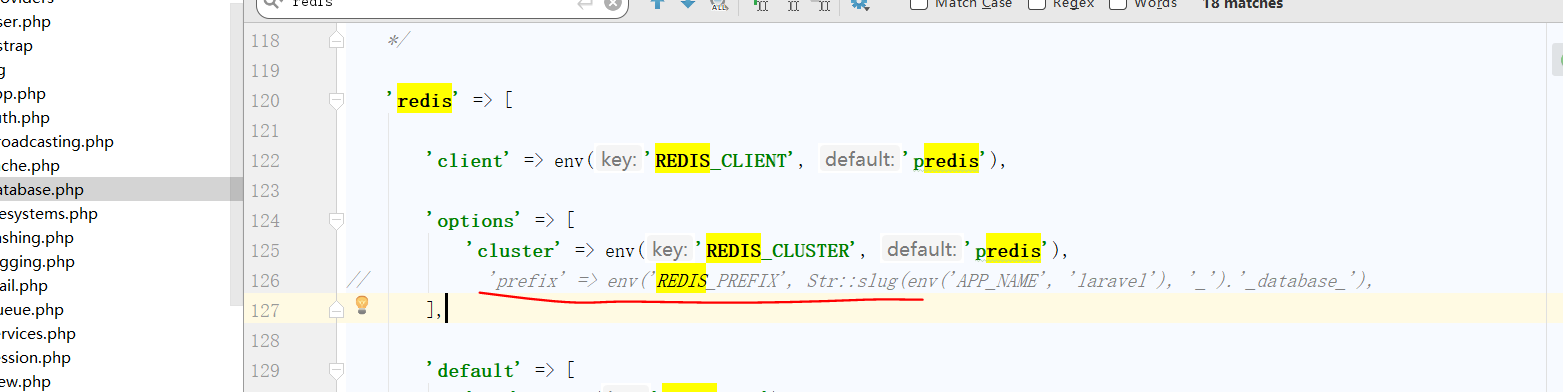


### 3.2私有频道

#### 1. 配置

在config/app/php中取消BroadcastServiceProvider注释，判断用户是否有权限监听我的的频道，用来注册广播授权路由和闭包在 .env 中配置驱动 BROADCAST\_DRIVER=redis

取消redis前缀

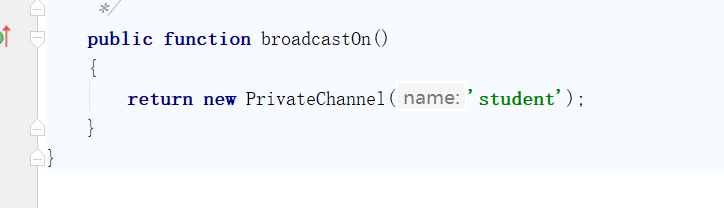


否则会在频道前加上前缀



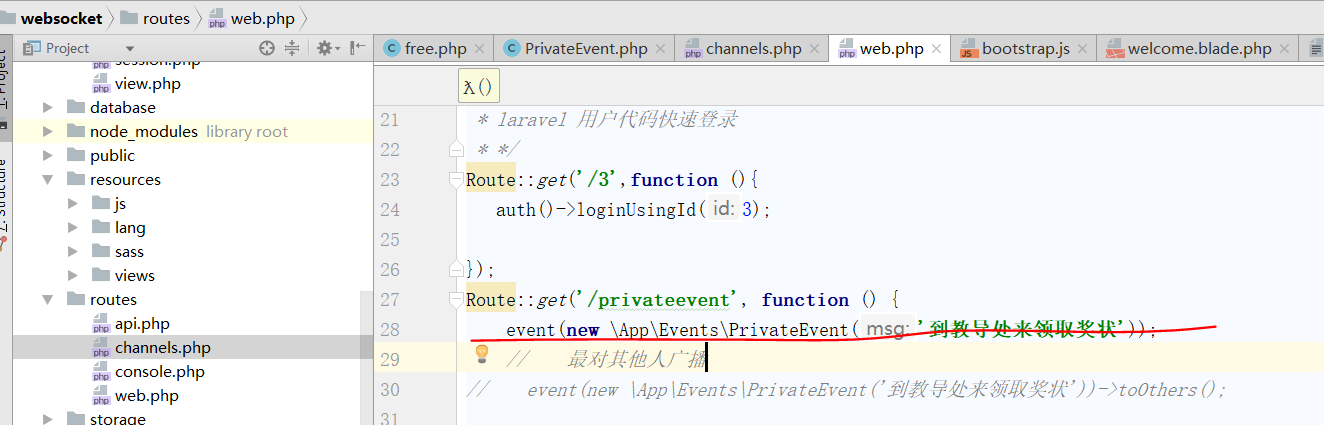
#### 安装 同公有频道一致

#### 定义事件

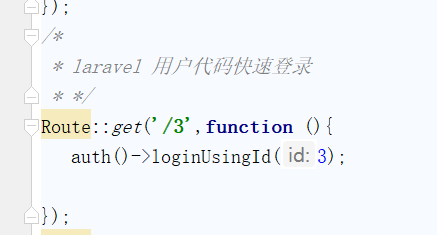


#### 4.设置频道权限

Route/channel



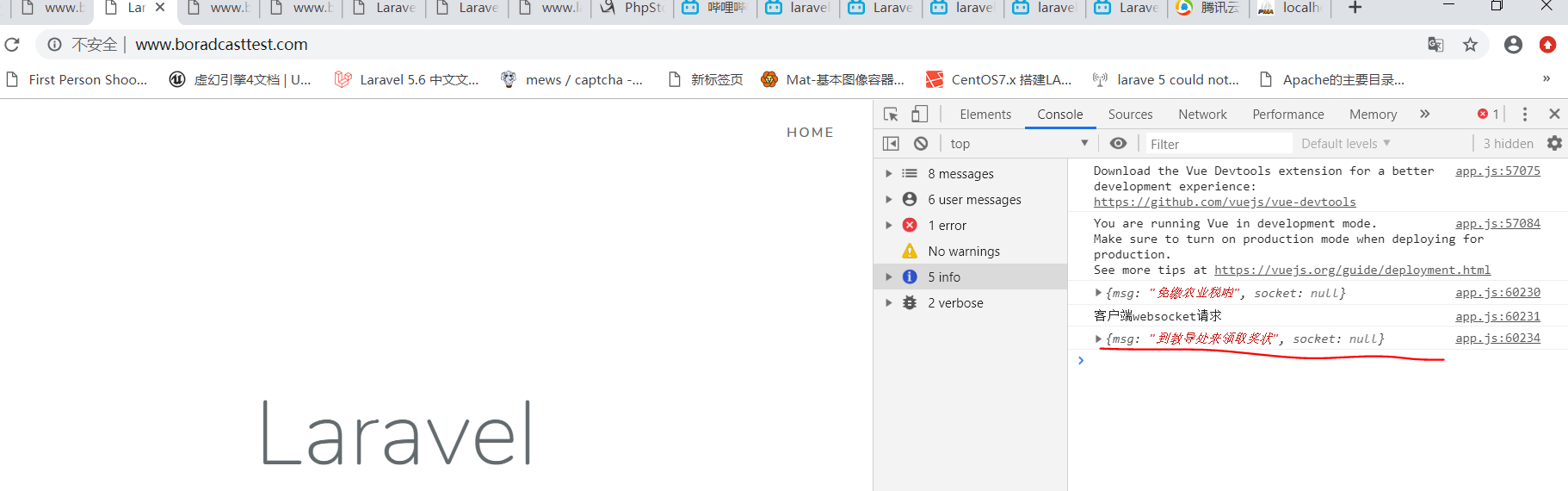
#### 5.模拟用户登录



#### 6.测试

输入<http://www.boradcasttest.com/3>

<http://www.boradcasttest.com/privateevent>



### 3.3实时聊天

<https://learnku.com/articles/36574>

composer config --global cache-files-maxsize 5000MiB

## 四．Laravel数据库操作

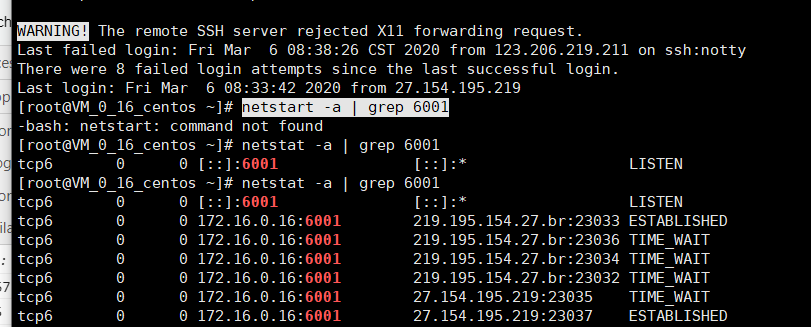
### 3.1查询构建器

## Linux广播环境搭建：

Laravel广播在服务器上搭建要开启相应的端口 6001



netstart -a | grep 6001 查看端口是否开启

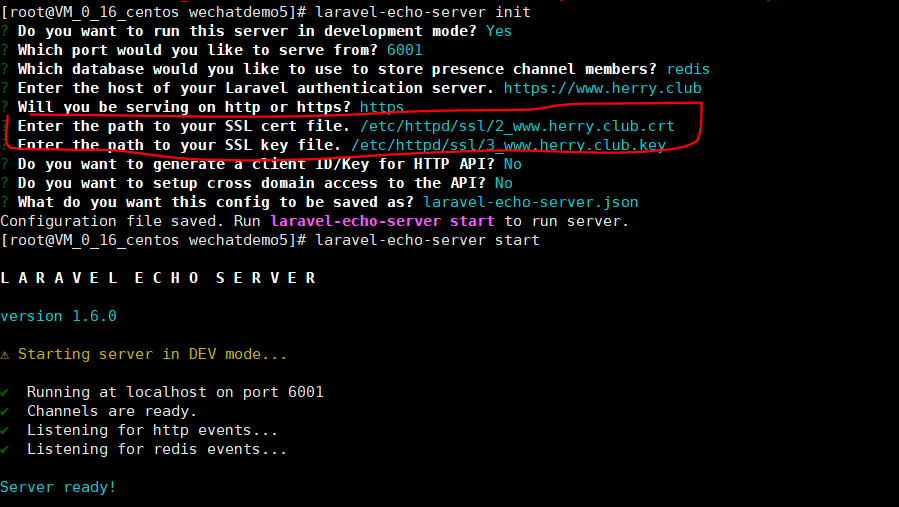


安装 redis ，node.js ,composer,其他安装和windows一样

Npm run watch 要在项目路径下启用

Laravel-echo-server init :

选择https要配置证路径



服务器上 要让laravel-echo-server 一直运行

参考：

<https://segmentfault.com/a/1190000019969184?utm_source=tag-newest>

<https://www.cnblogs.com/quzq/p/11192993.html>

<https://blog.csdn.net/donggege214/article/details/80264811>

### 安装supervisor

yum install supervisor

### 设置开机自启动

编辑服务文件

# vim /usr/lib/systemd/system/supervisord.service

1

内容如下

[Unit]

Description=Supervisor daemon

[Service]

Type=forking

PIDFile=/var/run/supervisord.pid

ExecStart=/bin/supervisord -c /etc/supervisord.conf

ExecStop=/bin/supervisorctl shutdown

ExecReload=/bin/supervisorctl reload

KillMode=process

Restart=on-failure

RestartSec=42s

[Install]

WantedBy=multi-user.target

保存退出

Esc:x

1

启动服务

# systemctl enable supervisord

### 配置一个进程

[program:laravel-echo-server]

process\_name=%(program\_name)s

directory=/home/www/wechatdemo5

command=laravel-echo-server start

autostart=true

autorestart=true

user=root

numprocs=1

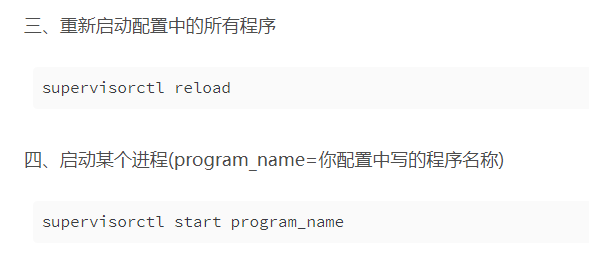
redirect\_stderr=true

stdout\_logfile=/var/log/supervisor/laravel-echo-server.log

### 启动：

supervisord -c /etc/supervisord.conf

或者输入supervisord      启动



基本命令<https://www.jianshu.com/p/ece62206d210>

监听某个端口

lsof -i:9001