

[Home](#) > [Laravel](#) >

LARAVEL

Laravel Queues Tutorial With Example From Scratch



By Krunal — Last updated Feb 22, 2018



Laravel Queues Tutorial With Example From Scratch is today's leading topic. Laravel 5.5 queues provide a unified API across a variety of different queue backends, such as Beanstalk, Amazon SQS, Redis, or even a relational database. Queues allow you to defer the processing of a time-consuming task, such as sending an email. Delaying these time-consuming tasks drastically speeds up web requests to your application.

The queue configuration file is stored in `config/queue.php`. In this file, you will find connection configurations for each of the queue drivers that are included with the framework, which consists of a database, [Beanstalkd](#), [Amazon SQS](#), [Redis](#), and synchronous driver that will execute jobs immediately (for local use). A `null` queue driver is also included which simply discards queued jobs.

Content Overview [\[hide\]](#)

- 1 [Laravel Queue Send Email Example](#)
- 2 [Step 1: Configure the Laravel](#)
- 3 [Step 2: Create a route for sending mail.](#)
- 4 [Step 3: Configure Queue.](#)
- 5 [Step 4: Create a job](#)

Laravel Queue Send Email Example

Sending an email is a time-consuming task, so what we will do is, we will create a job that's the only task is to send an email to the user in the background. So let us get started.



WEB HOSTING

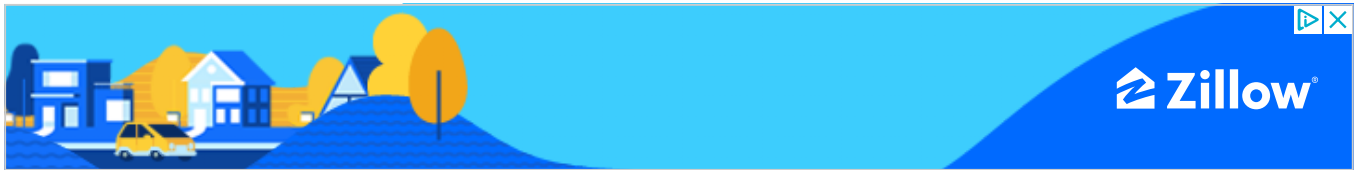
\$3.95/mo

Get Started Now

bluehost

Step 1: Configure the Laravel

Install it by the following command.



```
composer create-project laravel/laravel --prefer-dist emailqueue
```

Now, in the `.env` file, configure the database and email server.

For email sending, I have used <https://mailtrap.io>. So if you want that use, go to the website and signup, and you will see your username and password, grab it and put in the `.env` file, and you are good to go.

Also, we need to change the **queue driver to the database**. By default laravel uses `sync`.

```
APP_NAME=Laravel
APP_ENV=local
APP_KEY=base64:N/+b/PAQXgtG8OiYu3TKyudhicgYsGYfb551yrqkjmQ=
APP_DEBUG=true
APP_LOG_LEVEL=debug
APP_URL=http://localhost

DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
DB_DATABASE=queue
DB_USERNAME=root
DB_PASSWORD=

BROADCAST_DRIVER=log
CACHE_DRIVER=file
SESSION_DRIVER=file
SESSION_LIFETIME=120
```

```
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_DRIVER=smtp
MAIL_HOST=smtp.mailtrap.io
MAIL_PORT=2525
MAIL_USERNAME=
MAIL_PASSWORD=
MAIL_ENCRYPTION=tls

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
```

Step 2: Create a route for sending mail.

We will create one route to send an email and also create one controller called **EmailController**.

```
php artisan make:controller EmailController
```

In **routes >> web.php** file, add the following code.

```
Route::get('email', 'EmailController@sendEmail');
```

Okay, now we need to create one mailable class, so in the terminal, type the following command.

```
php artisan make:mail SendMailable
```

So, it will create this file inside **App\Mail\SendMailable.php**.

Now, write the **sendMail** function inside the **MailController.php** file.



```
namespace App\Http\Controllers;

use Illuminate\Http\Request;
use Illuminate\Support\Facades\Mail;
use App\Mail\SendMailable;

class EmailController extends Controller
{
    public function sendEmail()
    {
        Mail::to('mail@appdividend.com')->send(new SendMailable());
        echo 'email sent';
    }
}
```

Also, out **SendMailable.php** file looks like this.

```
<?php

namespace App\Mail;

use Illuminate\Bus\Queueable;
use Illuminate\Mail\Mailable;
use Illuminate\Queue\SerializesModels;
use Illuminate\Contracts\Queue\ShouldQueue;

class SendMailable extends Mailable
{
    use Queueable, SerializesModels;

    /**
     * Create a new message instance.
     *
     * @return void
     */
    public function __construct()
    {
        //
    }

    /**
     * Build the message.
     *
     * @return $this
     */
    public function build()
    {
        return $this->view('welcome');
    }
}
```

Now, start the server with the following command.

Switch to the browser and hit this URL: <http://localhost:8000/email>

You will see, there is a delay and then we can get the mail. So here is the solution and that is a **queue**.

Step 3: Configure Queue.

We can use queue with the **Database**, **Redis**, and other drivers as mentioned in [Laravel 5.5 Queues](#).

We will use Database and so we need to create a migration for Jobs table.

```
php artisan queue:table
```

It will create a migration and then we need to migrate the database.

```
php artisan migrate
```

It will create the **jobs** table in the database.

Step 4: Create a job

Now, we will need to create the job that purpose is to send the actual email to the user.

```
php artisan make:job SendEmailJob
```

Now, the send email function will reside in the job file. So that job file looks like this.

```
<?php

namespace App\Jobs;

use Illuminate\Bus\Queueable;
```

```
use Illuminate\Foundation\Bus\Dispatchable;
use Illuminate\Support\Facades\Mail;
use App\Mail\SendMailable;

class SendEmailJob implements ShouldQueue
{
    use Dispatchable, InteractsWithQueue, Queueable, SerializesModels;

    /**
     * Create a new job instance.
     *
     * @return void
     */
    public function __construct()
    {
        //
    }

    /**
     * Execute the job.
     *
     * @return void
     */
    public function handle()
    {
        Mail::to('mail@appdividend.com')->send(new SendMailable());
    }
}
```

In **EmailController.php** file, we will need to trigger this job.

So we need to **dispatch** the newly created **job** from the **EmailController**. So that code will look like this.

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Jobs\SendEmailJob;

class EmailController extends Controller
{
    public function sendEmail()
    {
        dispatch(new SendEmailJob());

        echo 'email sent';
    }
}
```

Now, again hit that email URL.

```
public function sendEmail()  
{  
    $emailJob = (new SendEmailJob())->delay(Carbon::now()->addSeconds(3));  
    dispatch($emailJob);  
  
    echo 'email sent';  
}
```

Now, hit the URL again.

This time there is only 3 seconds delay, or we can say no delay, but wait, if you switch to **mailtrap** then there is no mail either. So how we can send real mail also.



Go to the database and see the **jobs** table. There is one entry for the job. So that means, the process of that job is not started. So Laravel, there is a concept called **Queue Worker**. We need to run that **Queue Worker**.

Switch to the terminal and type the following command to start the queue worker.

```
php artisan queue:work
```

In the terminal, it will say like this.

```
[2017-12-21 13:58:14] Processing: App\Jobs\SendEmailJob  
[2017-12-21 13:58:19] Processed: App\Jobs\SendEmailJob
```

Now, go to the mailtrap, and yes we will see that mail sent.

Again refresh this URL.

It will say the email sent and after 5-6 seconds mail also sent to the mailtrap.

We can create as many jobs as we want but make sure that the queue worker is started and all the configurations are correctly set.



Laravel 5.5

Laravel Queues

Queues



Krunal - 575 Posts - 176 Comments



Krunal Lathiya is From India, and he is an Information Technology Engineer. By profession, he is the latest web and mobile technology adapter, freelance developer, Machine Learning, Artificial Intelligence enthusiast, and primary Author of this blog.

← PREV POST

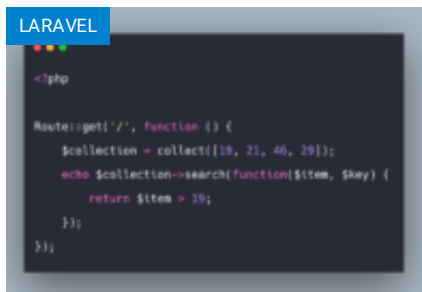
NEXT POST →

[Laravel Gates Tutorial Example From Scratch](#)

[Simple Nodejs Authentication System Using Passport](#)

YOU MIGHT ALSO LIKE

More From Author



Laravel Collections Search Method Tutorial With Example



Laravel Collections Filter Method Tutorial With Example



How To Create Multilingual Website using Laravel Localization

< PREV

NEXT >

9 COMMENTS



Daniel Barde Says

📅 1 year ago

Thanks, finally got this to work with beanstalkd.

↩ Reply



Nirav Says

📅 1 year ago

Code work like charm but how do i send mails to multiple user ??

↩ Reply



Jeff Says

📅 11 months ago

I think you are missing the use Carbon\Carbon; line in the EmailController.php file. Otherwise great post, thanks!

↩ Reply



Amadou Diallo Says

📅 10 months ago

Good work

I think that I wrote Carbon\Carbon because his app name is Carbone

↩ Reply



Lopo Says

📅 5 months ago

Great post.

If I wanted to save the email sent to the senders account I need to save it with IMAP. SwiftMailer don't to that and I found a workaround but I don't know in what place I should add it.

The code follows:

```
$mailbox = "{".$mail->Host.":143/imap/novalidate-cert}INBOX.Sent";  
$imapStream = imap_open($mailbox, $mail->Username, $mail->Password);  
imap_append($imapStream, $mailbox, "To: ".$destinatario."\r\n".$mail->getSentMIMEMessage(), "\\Seen");  
imap_close($imapStream);
```

↩ Reply



Andy Says

📅 5 months ago

Be sure to change app/config/queue.php to

```
"` 'default' => env('QUEUE_CONNECTION', 'database'),"
```

↩ Reply



Amit Says

📅 4 months ago

hey give me ur code successfully run code

↩ Reply



Mehul Says

📅 2 months ago

Thanks for this awesome tutorial!

↩ Reply



Pooja Says

📅 2 weeks ago

there is no entry in jobs table
and php artisan queue:work is also not working

↩ Reply

LEAVE A REPLY

Your email address will not be published.

Your Comment

Your Name *

Your Email *

Your Website

☐ Save my name, email, and website in this browser for the next time I comment.

POST COMMENT

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)



The advertisement features a laptop screen displaying an email campaign report. The report includes a 'Email Results' section with a '20% OFF' discount code. The 'Email Results' table shows:

Metric	Value	Goal
Email Opens	2851	100%
Clicks	1713	100%

Below the table, there is a 'Try It Free' button and the Constant Contact logo. The background of the ad shows a group of people in a retail setting.

ezeoic [report this ad](#)

Like This Page

Categories

- Angular
- AWS
- Blockchain
- C++
- Cloud Computing
- Dart
- Flutter
- Go
- Java
- Javascript
- Laravel
- Machine Learning
- MongoDB
- Node.js
- PHP
- Python

Bring your team together with Slack, the collaboration hub for work.

HIDE AD • AD VIA BUYSSELLADS

- SQL
- Tools
- VueJS



ezoic

report this ad

Tags

- C
- Go
- SQL
- Vue
- Java
- express
- React Native
- mongodb
- Angular 7
- React
- Laravel 5.5
- React.js
- Vue.js
- Laravel 5.6
- PHP
- Node.js
- Angular
- Laravel
- JavaScript
- Python

AdChoices

Job Vacancy

Python Tutorial

Stay With Us



840

Likes



307

Followers



30

Subscribers



407

Followers



26

Followers



162

Followers



Linkedin

Follow us




Posts

Posts

New simplicity
brings new power
New boards in Jira

Try it free

ATLASSIAN

 Jira

 ezoic

[report this ad](#)



 ezoic

[report this ad](#)