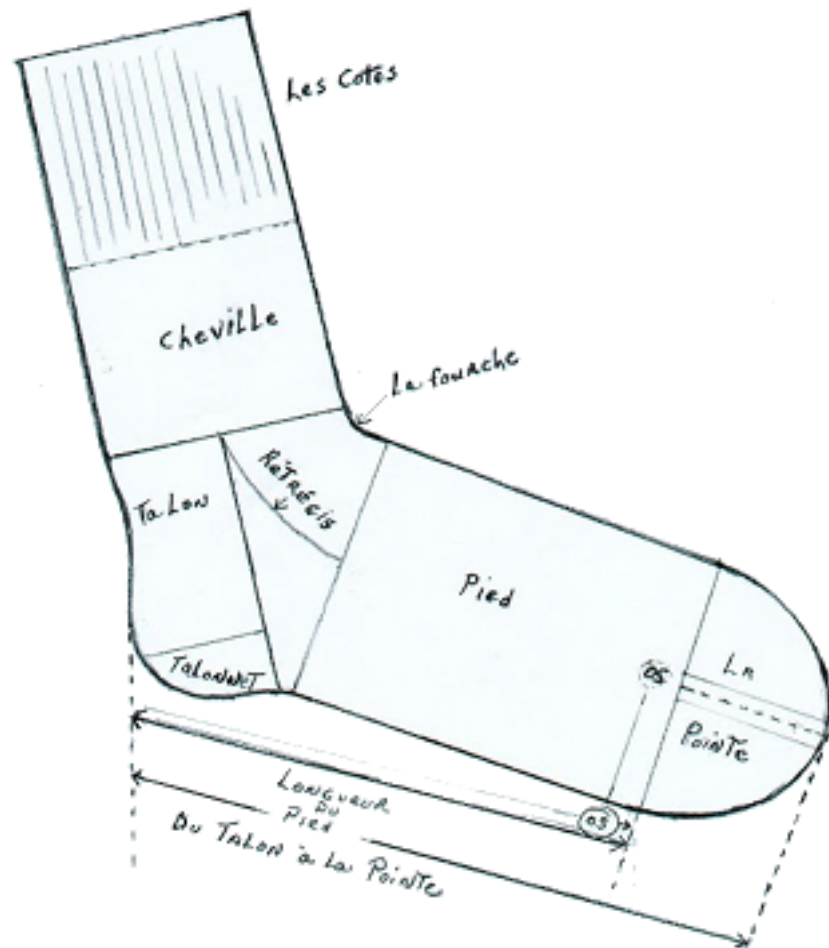# Chaussette Documentation

Release 0.3

Tarek Ziade

July 06, 2012

CONTENTS

Chaussette is a WSGI server you can use to run your Python WSGI applications.

The particularity of **Chaussette** is that it can either bind a socket on a port like any other server does **or** run against **already opened sockets**.

That makes **Chaussette** the best companion to run a WSGI ord Djando stack in Circus.

USAGE

Chaussette provides a console script you can launch against a WSGI application, like any WSGI server out there:

```
$ chaussette myapp
Application is <function myapp at 0x104d97668>
Serving on localhost:8080
Using <class chaussette.backend._wsgiref.ChaussetteServer at 0x104e58d50> as a backend
```

Chaussette has a specific mode to run against an existing open socket. This can only be used when Chaussette is forked from another process that has created the socket.

```
$ chaussette --fd 12 myapp
Application is <function myapp at 0x104d97668>
Serving on fd://12
Using <class chaussette.backend._wsgiref.ChaussetteServer at 0x104e58d50> as a backend
```

To get more options, just run *chaussette –help*.

RUNNING A DJANGO APPLICATION

Chaussette will let you run a Django project. You just need to use to provide the path to the project in the **application**, prefixed with **django:**

You can optionally provide the name of the settings module with **–django-settings**, when not provided, **Chaussette** will try to find it in the project directory and pass it for you.

Here's an example:

```
$ bin/chaussette django:path/to/mysite --backend gevent --django-settings mysite.settings
Application is <django.core.handlers.wsgi.WSGIHandler object at 0x10ec3f350>
Serving on localhost:8080
Using <class 'chaussette.backend._gevent.Server'> as a backend
```

RUNNING IN CIRCUS

The typical use case is to run Chaussette processes in Circus, which takes care of binding the sockets and spawning Chaussette processes.

To run your WSGI application using Circus, define a *socket* section in your configuration file, then add a Chaussette watcher.

Minimal example:

```
[circus]
endpoint = tcp://127.0.0.1:5555
pubsub_endpoint = tcp://127.0.0.1:5556
stats_endpoint = tcp://127.0.0.1:5557

[watcher:web]
cmd = chaussette --fd ${socket:web} --backend meinheld server.app
use_sockets = True
numprocesses = 5

[socket:web]
host = 0.0.0.0
port = 8000
```

When Circus runs, it binds a socket on the *8000* port and passes the file descriptor value to the Chaussette process, by replacing *${socket:web}* by the file number value.

BACKENDS

Chaussette is just a bit of glue code on the top of existing WSGI servers, and is organized around **back ends**.

By default Chaussette uses a pure Python implementation based on **wsgiref**, but it also provides more efficient back ends:

- **gevent** – based on Gevent's *pywsgi* server
- **fastgevent** – based on Gevent's *wsgi* server – faster but does not support streaming.
- **meinheld** – based on Meinheld's fast C server
- **waitress** – based on Pyramid's waitress pure Python web server

You can select your backend by using the **–backend** option and providing its name.

For some backends, you need to make sure the corresponding libraries are installed:

- **gevent** and **fastgevent**: *pip install gevent*
- **meinheld** : *pip install meinheld*
- **waitress** : *pip install waitress*

If you want to add your favorite WSGI Server as a backend to Chaussette, send me an e-mail !

If you curious about how each on of those backends performs, you can read:

- http://blog.ziade.org/2012/06/28/wgsi-web-servers-bench/
- http://blog.ziade.org/2012/07/03/wsgi-web-servers-bench-part-2/

RATIONALE AND DESIGN

Most WGSI servers out there provide advanced features to scale your web applications, like multi-threading or multi-processing. Depending on the project, the *process management* features, like respawning processes that die, or adding new ones on the fly, are not always very advanced.

On the other hand, Circus provides very advanced features to manage your processes, and is able to manage sockets as well.

The goal of *Chaussette* is to delegate process and socket management to Circus and just focus on serving requests.

Using a pre-fork model, Circus binds sockets and forks Chaussette processes that are able to accept connections on those sockets, as child processes.

For more information about this design, read :

- http://blog.ziade.org/2012/06/12/shared-sockets-in-circus.

- http://circus.readthedocs.org/en/latest/sockets/

USEFUL LINKS

- Repository : https://github.com/tarekziade/chaussette