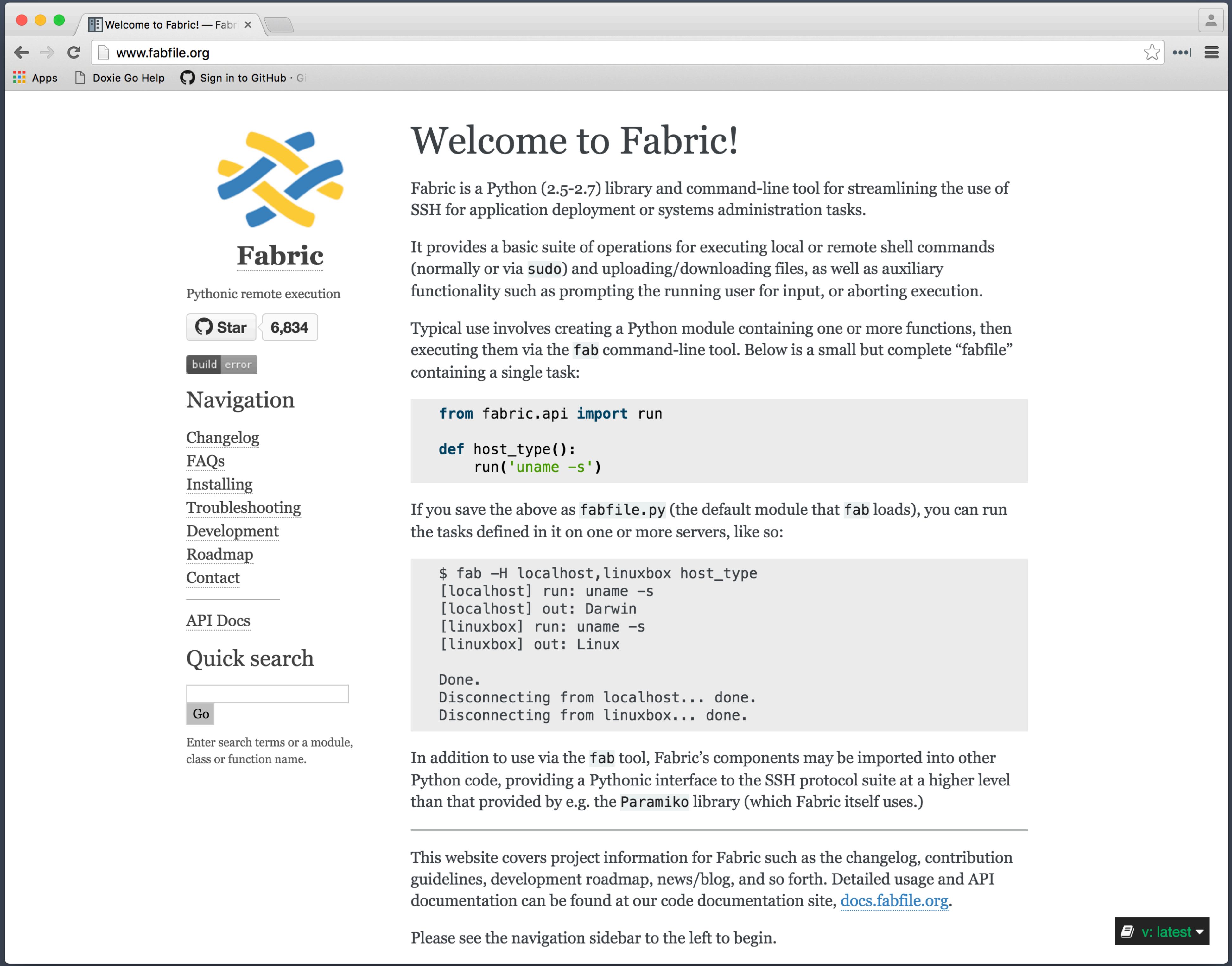
FABRIC AND PUPPET

RICH BURROUGHS

WHAT IS FABRIC?



Fabric

Pythonic remote execution

Navigation

- Changelog
- FAQs
- Installing
- Troubleshooting
- Development
- Roadmap
- Contact

API Docs

Quick search

Enter search terms or a module, class or function name.

Welcome to Fabric!

Fabric is a Python (2.5-2.7) library and command-line tool for streamlining the use of SSH for application deployment or systems administration tasks.

It provides a basic suite of operations for executing local or remote shell commands (normally or via `sudo`) and uploading/downloading files, as well as auxiliary functionality such as prompting the running user for input, or aborting execution.

Typical use involves creating a Python module containing one or more functions, then executing them via the `fab` command-line tool. Below is a small but complete “fabfile” containing a single task:

```
from fabric.api import run

def host_type():
    run('uname -s')
```

If you save the above as `fabfile.py` (the default module that `fab` loads), you can run the tasks defined in it on one or more servers, like so:

```
$ fab -H localhost,linuxbox host_type
[localhost] run: uname -s
[localhost] out: Darwin
[linuxbox] run: uname -s
[linuxbox] out: Linux

Done.
Disconnecting from localhost... done.
Disconnecting from linuxbox... done.
```

In addition to use via the `fab` tool, Fabric’s components may be imported into other Python code, providing a Pythonic interface to the SSH protocol suite at a higher level than that provided by e.g. the `Paramiko` library (which Fabric itself uses.)

This website covers project information for Fabric such as the changelog, contribution guidelines, development roadmap, news/blog, and so forth. Detailed usage and API documentation can be found at our code documentation site, docs.fabfile.org.

Please see the navigation sidebar to the left to begin.

> FABRIC IS A PYTHON....LIBRARY AND COMMAND-LINE TOOL FOR STREAMLINING THE USE OF SSH FOR APPLICATION DEPLOYMENT OR SYSTEMS ADMINISTRATION TASKS.
>
> – FABFILE.ORG

LIBRARY AND COMMAND-LINE TOOL

APPLICATION DEPLOYMENT OR SYSTEMS ADMINISTRATION TASKS

ORCHESTRATION + REMOTE EXECUTION

SIMILAR TO ANSIBLE AND CAPISTRANO

WHY FABRIC?

> I SUPPOSE IT IS TEMPTING, IF THE ONLY TOOL YOU HAVE IS A HAMMER, TO TREAT EVERYTHING AS IF IT WERE A NAIL.

— ABRAHAM MASLOW

WHY NOT MCOLLECTIVE?

FABRIC FUNCTIONS AKA OPERATIONS

RUN SUDO LOCAL GET PUT REBOOT

```
def deploy():
    local('git push')
    run('git pull')
    sudo('puppet agent -t')
```

FUNCTIONS ARE BUILDING BLOCKS

TASKS

```
@task
def foo():
    sudo('foo')
```

FABFILES

ENV

FAB COMMAND

```
fab -H host.example.com task
```

AUTHENTICATION

```
fab -I -H host.example.com task
```

ROLES

```
def get_roles(*roles):
    return lambda:[y for x in roles for y in env.roledefs[x]]

env.roledefs = {
    'web': ['web1', 'web2', 'web3'],
    'db': ['db1', 'db2'],
    'amq': ['mq1', 'mq2']
    'all': get_roles('web', 'db', 'amq')
}
```

PARAMETERS

```python
@task
def agent_run(environment='production'):
    sudo("/usr/local/bin/puppet agent -t --environment=%s" % environment)
```

```
fab -I -R web agent_run:environment=rich
```

AD HOC COMMANDS

```
fab -I -R all -- '/bin/ls /'
```

DEMO TIME

CONCLUSIONS