

# **BitNami DjangoStack 1.4-0**

## **Quick Start Guide**

---

# BitNami DjangoStack 1.4-0

---

Release 1.4-0 2012-03-26

Copyright © 2012 BitNami

<http://bitnami.org>

All rights reserved.

This product and its documentation are protected by copyright. The information in this document is provided on an "as is" basis, without warranty. Although every precaution has been taken in the preparation of this document, the authors will not have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the information contained in this work.

Trademark names may appear in this document. All registered and unregistered trademarks in this document are the sole property of their respective owners.

# Acknowledgements

---

BitNami DjangoStack is based on a number of open source components:

The Django Project, developed by Lawrence Journal-World.

<http://www.djangoproject.com/>.

The Apache HTTP server, developed by The Apache Software Foundation.

<http://www.apache.org/>.

SQLite. A library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine.

<http://www.sqlite.org>

Python. Python is a dynamic object-oriented programming language that can be used for many kinds of software development.

<http://www.python.org/>.

MySQL. The world's leading open source database.

<http://www.mysql.com>

PostgreSQL. The world's most advanced open source database.

<http://www.postgresql.org>.

pysqlite. A DB-API 2.0 database interface for Python and SQLite.

<http://initd.org/tracker/pysqlite>.

MySQLdb. The Python interface to MySQL database.

<http://sourceforge.net/projects/mysql-python/>.

psycopg. The most popular PostgreSQL adapter for the Python. <http://initd.org/psycopg/>.

Mod\_wsgi. Mod\_wsgi is an Apache module that provides a WSGI compliant interface for hosting Python based web applications under Apache. <http://code.google.com/p/modwsgi/>.

The zlib data-compression library.

<http://www.zlib.net>

The libiconv library for multiple character encoding.

[www.gnu.org/software/libiconv/](http://www.gnu.org/software/libiconv/)

The expat XML parser.

<http://expat.sourceforge.net/>.

The neon HTTP and WebDAV client library.

<http://www.webdav.org/neon/>.

You can find the individual licenses for the above projects as part of the installation.

# BitNami DjangoStack Overview

---

The BitNami DjangoStack is an installer that greatly simplifies the installation of Django and runtime dependencies. It includes ready-to-run versions of Django, Apache, MySQL, PostgreSQL, and Python. DjangoStack is distributed for free under the Apache 2.0 license. Please see the appendix for the specific licenses of all Open Source components included.

## Components

BitNami DjangoStack includes **Django 1.4-0**, **Apache 2.2.22**, **SQLite 3.7.3**, **Python 2.6.5**, **MySQL 5.5.16**, **PostgreSQL 9.1.1** and other required libraries.

**Django** Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design. Django focuses on automating as much as possible and adhering to the DRY principle.

<http://www.djangoproject.com>.

**Apache** is the most popular HTTP server on the Internet. It provides a secure, efficient and extensible web platform. It is maintained by the Apache Software Foundation. You can find more information about Apache at

<http://www.apache.org>.

**SQLite** is a in-process library that implements a self-contained, serverless, zero-configuration, transactional SQL database engine. The code for SQLite is in the public domain and is thus free for use for any purpose, commercial or private. SQLite is currently found in more applications than we can count, including several high-profile projects.

<http://www.sqlite.org>.

**Python** is a dynamic object-oriented programming language that can be used for many kinds of software development. It offers strong support for integration with other languages and tools, comes with extensive standard libraries, and can be learned in a few days. Many Python programmers report substantial productivity gains and feel the language encourages the development of higher quality, more maintainable code. <http://www.python.org/>.

**MySQL** is the world's most popular open source database. It is a relational database management system that combines speed, reliability and ease of use. It is developed and maintained by MySQL AB. You can find more information about MySQL at <http://www.mysql.com>.

**PostgreSQL** is a powerful, open source relational database system. It has more than 15 years of active development and a proven architecture that has earned it a strong reputation for reliability, data integrity, and correctness. You can find more information about PostgreSQL at <http://www.postgresql.com>.

## Requirements

To run BitNami DjangoStack you will need:

- Intel x86 or compatible processor
- Minimum of 512 MB RAM
- Minimum of 100 MB hard drive space
- An x86 or x64 Linux operating system or  
A 32-bit Windows operating system such as Windows 2000, XP, Vista or Windows Server 2003,

Windows 7, Windows Server 2008 or an OS X operating System.

- TCP/IP protocol support

# Installation Guide

---

This section describes where to download BitNami DjangoStack and the different installation modes that are available.

## Downloading BitNami DjangoStack

You can download the BitNami DjangoStack binary file from <http://bitnami.org>. It will be named `bitnami-djangostack-1.4-0-windows-installer.exe` for Windows. On Linux, it will be named `bitnami-djangostack-1.4-0-linux-installer.bin` or `bitnami-djangostack-1.4-0-linux-x64installer.bin`. The same binary file will work on any Linux distribution. On OSX, it will be named `bitnami-djangostack-1.4-0-osx-x86-installer.app.zip`.

Once you have downloaded the file on Linux, make sure it has read and executable permissions: From your Desktop environment, right-click on the file, select "Properties" and then set the appropriate permissions. Alternatively, you can issue the following shell command:

```
$ chmod 755 bitnami-djangostack-1.4-0-linux-installer.bin
```

## Installing BitNami DjangoStack

You can install BitNami DjangoStack in graphical and text modes. By default, the graphical mode will be used.

### Graphical Mode

To begin the installation process, double-click on the file from your Desktop environment (bear in mind uncompressing it if you are using OSX) or invoke it directly from the command line with:

```
$ ./bitnami-djangostack-1.4-0-linux-installer.bin on Linux
$ ./bitnami-djangostack-1.4-0-linux-x64-installer.bin on Linux 64 bits
$ open ./bitnami-djangostack-1.4-0-osx-x86-installer.app on OSX x86
```

You will be greeted by the 'Welcome' screen. The next step is to select the installation directory. The default installation path will be a folder on your home directory if you are running the installer as a regular user, or `/opt/djangostack-1.4-0`, if you are running the installation as root. If the destination directory does not exist, it will be created as part of the installation.

The default listening ports are 80 for Apache on Windows, 8080 for Apache on Linux and OSX, 3306 for MySQL and 5432 for PostgreSQL. If those ports are already in use by other applications, you will be prompted for alternate ports to use.

After selecting the installation directory you will be asked for the password to the initial MySQL root and postgres accounts. This password cannot be empty.

The next screen will ask you if you want to create an initial project within the installer. If you select 'no', please skip the next paragraph.

In order to create a project, the installer will prompt you for a project name and an available database engine. It is strongly recommended using MySQL or PostgreSQL for a real deployment.

You are now ready to begin the installation, which will start when you press 'Next'. Once the installation process has been completed, you will see the 'Installation Finished' page. You can launch the browser at this point.

If you receive an error message during installation, please refer to the Troubleshooting section.

The rest of this guide assumes that you installed BitNami DjangoStack in `/home/user/djangostack-1.4-0` and that you use port 8080 for Apache.

## Text Mode

This installation mode is designed for remote installation or installation on servers without X-Window support. It is started by default when a graphical environment is not available or by issuing one of the following commands, depending on your operating system:

```
$ ./bitnami-djangostack-1.4-0-linux-installer.bin --mode text on Linux
$ ./bitnami-djangostack-1.4-0-linux-x64-installer.bin --mode text on Linux 64 bits
$ ./bitnami-djangostack-1.4-0-osx-x86-installer.app/Contents/MacOS/installbuilder.sh --mode text on OSX x86
```

You will be greeted by the 'Welcome' message. The next step is to select the installation directory. The default installation path will be a folder on your home directory if you are running the installer as a regular user, or `/opt/djangostack-1.4-0`, if you are running the installation as root. If the destination directory does not exist, it will be created.

The default listening ports are 8080 for Apache, 3306 for MySQL and 5432 for PostgreSQL. If those ports are already in use by other applications, you will be prompted for alternate ports to use. Remember that if you plan to run both applications as a regular user you should select port numbers above 1024.

After selecting the installation directory you will be asked for the password to the initial MySQL root and postgres accounts. This password cannot be empty.

The next screen will ask you if you want to create an initial project within the installer. If you select 'no', please skip the next paragraph.

In order to create a project, the installer will prompt you for a project name and an available database engine. It is strongly recommend using MySQL or PostgreSQL for a real deployment.

You are now ready to begin the installation process, which will start when you press 'Enter'. Once the installation process has been completed, you will see the 'Installation Finished' message.

## Directory Structure

The installation process will create several subfolders under the main installation directory:

- `apache2/`: Apache Web server.
- `sqlite/`: SQLite Database.
- `python`: Python interpreter.
- `mysql`: MySQL files.
- `postgresql`: PostgreSQL files.
- `common/`: Common libraries.
- `scripts/`: Script with environment vars.

- `licenses/`: Component licenses.
- `apps/django/`: Django Application files.

# Uninstalling BitNami DjangoStack

---

As part of the installation, an uninstall program will be created in the installation folder and a link placed in the Start Menu on Windows. The uninstallation can also be performed in graphical, text and unattended modes. You can run the uninstaller by double-clicking on the uninstall application or through the command line:

```
$ /home/user/djangostack-1.4-0/uninstall on Linux
```

```
$ open /Applications/djangostack-1.4-0/uninstall.app on OSX
```

# Django

---

Developed and used over two years by a fast-moving online-news operation, Django was designed to handle two challenges: the intensive deadlines of a newsroom and the stringent requirements of the experienced Web developers who wrote it. It lets you build high-performing, elegant Web applications quickly. Django focuses on automating as much as possible and adhering to the DRY principle.

## Starting DjangoStack

Django needs a project before it can be started. There is an option in the own installer to setup an initial project. Projects are stored in `/home/user/djangostack-1.4-0/apps/django/django_projects` on Linux, `/Applications/djangostack-1.4-0/apps/django/django_projects` on OSX, `C:\Documents and Settings\user\BitNami DjangoStack Projects` on Windows 2000 or XP and `C:\Users\user\BitNami DjangoStack Projects` on Windows Vista or 7. If you didn't setup the project at installation time, consult the section "Creating a Django Project". If you created a project during installation time, your project will be in the folder `django_projects` or `BitNami DjangoStack Projects`.

You can start your Django project in two different ways. One is using the standalone server in Django (no Apache required), which is started using a BitNami console (consult the section "Django Environment") from the command line inside your project folder with the following line:

```
$ python manage.py runserver host
```

Where `host` must be a valid hostname. Optionally you may not specify a host and your server will be started on port 8000 at localhost (127.0.0.1:8000).

The Django standalone server is most suitable for testing purposes and is not advised to be used for production. The way to launch your Django project for production purposes is to use the Apache webserver with the `mod_wsgi` module. Use the BitNami Application Manager, a simple graphical interface included in the stack, to start and stop the BitNami servers. It is located in the same installation directory.

To start the utility, double click the file named 'manager-linux', 'manager-windows' or 'manager-osx' from your file browser.

Alternatively, you can also start and stop the services manually, as explained below. There is a script in the installation directory which controls the status of the bundled application servers. The syntax to use it is the following:

```
$ /home/user/djangostack-1.4-0/ctlscript.sh start and
$ /home/user/djangostack-1.4-0/ctlscript.sh stop on Linux
$ /Applications/djangostack-1.4-0/ctlscript.sh start and
$ /Applications/djangostack-1.4-0/ctlscript.sh stop on OSX
```

If no errors are found, you will see a message similar to:

```
Syntax OK
/home/user/djangostack-1.4-0/apache2/scripts/ctl.sh : httpd started at port 8080
/home/user/djangostack-1.4-0/postgresql/scripts/ctl.sh : postgresql started at port 5432
/home/user/djangostack-1.4-0/mysql/scripts/ctl.sh : mysql started at port 3306 on Linux
```

```
Syntax OK
/Applications/djangostack-1.4-0/apache2/scripts/ctl.sh : httpd started at port 8080
/Applications/djangostack-1.4-0/postgresql/scripts/ctl.sh : postgresql started at port 5432
/Applications/djangostack-1.4-0/mysql/scripts/ctl.sh : mysql started at port 3306 on OSX
```

That will start Apache Postgresql and MySQL servers. Once started, you can open your browser and access the following URL:

`http://127.0.0.1:80` on Windows

`http://127.0.0.1:8080` on Linux and OSX

# Apache Web server

---

This section describes how to start Apache for the first time and gives a basic introduction to the Apache configuration and log files.

## Starting Apache

On Windows, you can start BitNami DjangoStack Services using the shortcuts created in the Start Menu, under Programs -> BitNami -> BitNami Service

You can start Apache from the command line by issuing:

```
$ ./ctlscript.sh start apache on Linux
$ ./ctlscript.sh start apache on OSX
```

Be sure you are in the main directory of the DjangoStack installation

If no errors are found, you will see a message similar to:

```
Syntax OK
django-1.4-0/apache2/scripts/ctl.sh : httpd started
```

This indicates that the server is up and running. You can test so by opening a browser and accessing the following URL `http://127.0.0.1:8080`, which will take you to the test page.

If you receive an error message, the server cannot start or you cannot see the test page, please refer to the Troubleshooting section.

## Stopping Apache

On Windows, you can stop BitNami DjangoStack Services using the shortcuts created in the Start Menu, under Programs -> BitNami -> BitNami Service

You can stop Apache from the command line issuing:

```
$ ./scripts/ctl.sh stop apache on Linux
$ ./scripts/ctl.sh stop apache on OSX
```

Be sure you are in the main directory of the DjangoStack installation

After a moment you should see a message similar to:

```
Syntax OK
/home/user/djangostack-1.4-0/apache2/scripts/ctl.sh : httpd stopped
```

## Apache Basic Configuration

The main Apache configuration file is called `httpd.conf` which you can find at `C:\Program Files\BitNami DjangoStack\apache2\conf\httpd.conf` on Windows  
`/home/user/djangostack-1.4-0/apache2/conf/httpd.conf` on Linux  
`/Applications/djangostack-1.4-0/apache2/conf/httpd.conf` on OSX.

This configuration file calls Django's Apache configuration file located at

C:\Documents and Settings\user\BitNami DjangoStack Projects\apps\django\conf\django.conf or  
C:\Users\user\BitNami DjangoStack Projects\apps\django\conf\django.conf on Windows  
/home/user/djangostack-1.4-0/apps/django/conf/django.conf on Linux  
/Applications/djangostack-1.4-0/apps/django/conf/django.conf on OSX.

Since the configuration of Apache strongly depends on your project, consult the Django Official website for more information on [how to set up Django on Apache with mod\\_wsgi](#).

Once Apache starts, it will create two log files, the `access_log` and the `error_log`. You can find both files at

C:\Program Files\BitNami DjangoStack\apache2\logs on Windows  
/home/user/djangostack-1.4-0/apache2/logs on Linux  
/Applications/djangostack-1.4-0/apache2/logs on OSX

The `access_log` file is used to track client requests. When a client requests a document from the server, Apache records several parameters associated with the request in this file, such as: the IP address of the client, the document requested, the HTTP status code, and the current time.

The `error_log` file is used to record important events. This file includes error messages, startup messages, and any other significant events in the life cycle of the server. This is the first place to look when you run into a problem when using Apache.

If you already have a web page and you want to serve its content with Apache, you can do it simply by copying your files to the default document root directory:

C:\Program Files\BitNami DjangoStack\apache2\htdocs on Windows  
/home/user/djangostack-1.4-0/apache2/htdocs/ on Linux  
/Applications/djangostack-1.4-0/apache2/htdocs/ on OSX

With the default configuration, Apache will wait for requests in the port 8080 on Linux and OSX or 80 on Windows. You can change that by editing the `httpd.conf` file and modifying the value specified in the `port` directive.

You can find more information about Apache in the technical documentation that is located at

C:\Program Files\BitNami DjangoStack\apache2>manual on Windows  
/home/user/djangostack-1.4-0/apache2/manual on Linux  
/Applications/djangostack-1.4-0/apache2/manual on OSX

# Django

---

This section gives an introduction on how to use Django. You can find the Django official tutorial and documentation at <http://www.djangoproject.com/>

## Django Environment

BitNami DjangoStack ships with a BitNami console to run any command included in the Stack. This is a console where you can readily use Django as well as the bundled components of DjangoStack like MySQL or Python. To use this console, open a terminal window and use the following commands:

```
/home/user/djangostack-1.4-0/use_djangostack on Linux, where user is the name of your user account.  
/Applications/djangostack-1.4-0/use_djangostack on OSX
```

Once inside the console you can test if everything went well by using the following commands:

```
$ which python
```

And you may see something like:

```
/home/user/djangostack-1.4-0/python/bin/python on Linux  
/Applications/djangostack-1.4-0/python/bin/python on OSX
```

where user is the name of your user account, and indicates the `python` binary of BitNami DjangoStack will be used instead of other pre-installed versions of Python.

On Windows, there is a shortcut in Start -> BitNami DjangoStack -> Use Bitnami DjangoStack to run the BitNami console as well.

## Creating a Django Project

To create a new Django project, start the BitNami console as described above, and once inside it, use the following command to create a new project inside projects folder:

```
$ cd "C:\Documents and Settings\user\BitNami DjangoStack Projects" on Windows XP  
$ cd "C:\Users\user\BitNami DjangoStack Projects" on Windows Vista or 7  
$ cd /home/user/djangostack-1.4-0/apps/django/django_projects on Linux  
$ cd /Applications/djangostack-1.4-0/apps/django/django_projects on OSX  
$ django-admin.py startproject newProject
```

## Create an application

Once your project is created, you can start developing your applications. To create an application of the project `newProject` use the following command to create an application `newApp`:

```
$ cd newProject  
$ python manage.py startapp newApp
```

After the project is created and an application started, you may wanted to configure your project settings. Inside the `newProject` folder you can edit `settings.py` to properly set the installer settings. Please consult the [official Django tutorial](#)

to a more in depth explanation of how to use Django

## Create or update the database

After properly setting up the project settings you can create the database or update it if you already have created it by using the command (inside the project's folder):

```
$ python manage.py syncdb
```

## Configure your Django project on Apache

As we saw in the section "Apache Basic Configuration", you must configure some files in order to start using the new Django project on Apache server. The steps to do that are:

Create a WSGI application script file which will be loaded by Apache. You can use the `django.wsgi` file located at `C:\Documents and Settings\user\BitNami DjangoStack Projects\apps\django\scripts\django.wsgi` or `C:\Users\user\BitNami DjangoStack Projects\apps\django\scripts\django.wsgi` on Windows  
`/home/user/djangostack-1.4-0/apps/django/scripts/django.wsgi` on Linux  
`/Applications/djangostack-1.4-0/apps/django/scripts/django.wsgi` on OSX  
as base and edit the path to point to your project folder and change the module name to your project name. The final result should be similar to:

```
...
sys.path.append('/home/user/djangostack-1.4-0/apps/django/django_projects/newProject')
os.environ['DJANGO_SETTINGS_MODULE'] = 'newProject.settings'
...
```

Create an Apache configuration file. You can use the `django.conf` file located at `C:\Documents and Settings\user\BitNami DjangoStack Projects\apps\django\conf\django.conf` or `C:\Users\user\BitNami DjangoStack Projects\apps\django\conf\django.conf` on Windows  
`/home/user/djangostack-1.4-0/apps/django/conf/django.conf` on Linux  
`/Applications/djangostack-1.4-0/apps/django/conf/django.conf` on OSX  
as base and edit the `WSGIScriptAlias` directive to specify your URL mount point and the absolute pathname to the WSGI application script file. For example:

```
WSGIScriptAlias /newProject "/home/user/djangostack-1.4-0/apps/django/scripts/newProject.wsgi"
```

Finally edit the `http.conf` file to include the Apache configuration file you have been created.

Restart the server and browse the following URL `http://127.0.0.1/newProject` on Windows or `http://127.0.0.1:8080/newProject` on Linux and OSX to access to your new created project.

# Troubleshooting

---

This section describes some of the most common problems you may find when installing BitNami DjangoStack.

## Installer

### Installer Payload Error

You may get the following error while trying to run the installer from the command line:

```
Installer payload initialization failed. This is likely due to an incomplete or corrupt
downloaded file.
```

The installer binary is not complete, likely because the file was not downloaded correctly. You will need to download the file and repeat the installation process.

## Apache

If you find any problem starting Apache, the first place you should look at is the error log file that will be created at

`c:\Program Files\BitNami DjangoStack\apache2\logs\error_log` on Windows

`/home/user/djangostack-1.4-0/apache2/logs/error_log` on Linux

`/Applications/djangostack-1.4-0/apache2/logs/error_log` on OSX

There you will likely find useful information to determine what the problem may be. For issues not covered in this Quick Start guide, please refer to the Apache documentation included as part of the installation, which is located at `/home/user/djangostack-1.4-0/apache2/htdocs/manual`.

### Test page is not working

If the server seems to start correctly (i.e. you can see the `./apachectl start: httpd started` message) but you cannot see the test page when you type `http://127.0.0.1:8080/` in your browser, it may be that there is already a server running in that port.

### Search

`/home/user/djangostack-1.4-0/apache2/logs/error_log` on Linux

`/Applications/djangostack-1.4-0/apache2/logs/error_log` on OSX

for a message similar to this:

```
[Mon Oct 9 19:52:10 2007] [crit] (98)Address already in use: make_sock: could not bind to port
8080
```

This means that the port 8080 is already being used by another program. You can either stop the program that is using that port or edit the `httpd.conf` configuration file and change the port Apache will listen for requests in.

### Cannot bind to port 80

If you change the default listening port for Apache and get the following error in `error_log`:

```
[Mon Oct 9 20:09:50 2007] [crit] (13)Permission denied: make_sock: could not bind to port 80
```

you do not have enough permissions to bind to that port.

On Unix, to be able to bind to a port below 1024, you need to be a privileged user. Log in as root or issue the 'su' command and try to start the server again.

## MySQL

If you encounter any problems starting MySQL, the first place to look in is the "Problems and Common Errors" section of the MySQL manual, which you will find at <http://dev.mysql.com/doc/>

The following are some common problems:

### **Access denied when trying to connect to MySQL.**

If you get an `Access Denied` message while trying to connect to MySQL, make sure you are using the correct username and password.

### **"Can't connect to server" message.**

Make sure the MySQL daemon is up and running.

## PostgreSQL

If you encounter any problems starting PostgreSQL, the first place to look in is the "Problems and Common Errors" section of the PostgreSQL manual, which you will find at <http://www.postgresql.org/docs/>

The following are some common problems:

### **Access denied when trying to connect to PostgreSQL.**

If you get an `Access Denied` message while trying to connect to PostgreSQL, make sure you are using the correct username and password.

### **"Can't connect to server" message.**

Make sure the PostgreSQL daemon is up and running.