

Install JBoss Wildfly on Ubuntu 18.04

WildFly, formerly known as **JBoss AS**, or simply **JBoss**, is a **Java EE certified application server** authored by JBoss, now developed by Red Hat.

Wildfly is a production-ready, cross-platform and open-source application server that is flexible, lightweight and is based on pluggable subsystems that can be added or removed as needed.

Some of the key features of Wildfly are:

- **Incredible Web Performance and Scalability**
- **Efficient Memory Management**
- **Slim and Customizable Runtime**
- **Powerful Server Administration**
- **Unified Configuration & Management**
- **Domain & Standalone Operational Modes**
- **Concurrent and Fast Classloading**
- **Full Java EE 8 Support**

In this tutorial, we will go through the process of setting up the Wildfly server on Ubuntu 18.04. So let us begin.

Step 1 — Install JDK

Wildfly requires Java 8 or later versions to work. You can check and verify that Java is installed with the following command.

```
$ java -version
```

If java is not installed, you will see “***java: command not found***”. Run below commands to install Java.

```
$ sudo apt-get update
```

```
$ sudo apt-get install default-jdk -y
```

After installation, check if java is installed correctly by executing below command

```
$ java -version
```

```
openjdk version "11.0.3" 2019-04-16
OpenJDK Runtime Environment (build 11.0.3+7-Ubuntu-1ubuntu218.04.1)
OpenJDK 64-Bit Server VM (build 11.0.3+7-Ubuntu-1ubuntu218.04.1, mixed mode, sharing)
```

If Java is installed, the output should look similar to above depending upon what is the latest version of java at that time.

Step 2 — Download and Extract Wildfly Server

Check Wildfly [downloads](#) page for latest releases before downloading. For this tutorial, we will download **Wildfly 16.0.0.Final (Java EE Full & Web Distribution)**.

We are going to install Wildfly to **/opt/** directory, so we will download the Wildfly package to that location.

Change directory to **/opt/** and download Wildfly to that directory.

```
$ cd /opt
```

```
$ sudo wget https://download.jboss.org/wildfly/16.0.0.Final/wildfly-16.0.0.Final.tar.gz
```

Extract the tar package and rename the extracted directory to ***wildfly***. This will be Wildfly's installation directory

```
$ sudo tar -xvzf wildfly-16.0.0.Final.tar.gz
```

```
$ sudo mv wildfly-16.0.0.Final /opt/wildfly
```

Step 3 — Create User and Group for Wildfly

We should not run Wildfly under the root user for security reasons. Let's create a group ***wildfly*** and add a user ***wildfly*** to it.

Additionally, the home directory of ***wildfly*** user will be the Wildfly's installation directory i.e. ***/opt/wildfly/***.

```
$ sudo groupadd wildfly
```

```
$ sudo useradd -r -g wildfly -d /opt/wildfly -s /sbin/nologin wildfly
```

Step 4 — Change Permission and Ownership of the Wildfly Installation Directory

Next, we will modify ownership and permission of ***/opt/wildfly/*** directory. We will also give executable permissions to ***/opt/wildfly/bin/*** directory. While under ***/opt/*** directory, run the following commands:

```
$ sudo chown -R wildfly: wildfly  
  
$ sudo chmod o+x /opt/wildfly/bin/
```

Step 5 — Creating a SystemD Service File for Wildfly

Create a configuration directory for Wildfly under */etc/* directory by the name ***wildfly***.

```
$ cd /etc/  
  
$ sudo mkdir wildfly
```

Copy Wildfly configuration file */opt/wildfly/docs/contrib/scripts/systemd/wildfly.conf* to */etc/wildfly/* directory

```
$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.conf /etc/wildfly/
```

Next, copy Wildfly launch script (***launch.sh***) under */opt/wildfly/docs/contrib/scripts/systemd/* to */opt/wildfly/bin/* directory

```
$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/launch.sh /opt/wildfly/bin/
```

We need to make ***wildfly*** user as the owner of this script so that it can execute it:

```
$ sudo chown wildfly: /opt/wildfly/bin/launch.sh
```

Now, copy service definition file (**wildfly.service**)

under **/opt/wildfly/docs/contrib/scripts/systemd/** to **/etc/systemd/system/** directory

```
$ sudo cp /opt/wildfly/docs/contrib/scripts/systemd/wildfly.service /etc/systemd/system/
```

Open **wildfly.service** in an editor

```
$ sudo nano /etc/systemd/system/wildfly.service
```

Make the changes marked as bold or you can simply copy/paste the below content as it is.

```
[Unit]
```

```
Description=The WildFly Application Server
```

```
After=syslog.target network.target
```

```
Before=httpd.service
```

```
[Service]
```

```
Environment=LAUNCH_JBOSS_IN_BACKGROUND=1
```

```
EnvironmentFile=-/etc/wildfly/wildfly.conf
```

```
User=wildfly
```

```
Group=wildfly
```

```
LimitNOFILE=102642
```

```
PIDFile=/var/run/wildfly/wildfly.pid
```

```
ExecStart=/opt/wildfly/bin/launch.sh $WILDFLY_MODE $WILDFLY_CONFIG $WILDFLY_BIND
```

```
StandardOutput=null
```

```
[Install]
```

```
WantedBy=multi-user.target
```

Save and exit the file.

Reload **systemd** manager configuration and enable **wildfly** service on system startup

```
$ sudo systemctl daemon-reload
```

```
$ sudo systemctl enable wildfly
```

To start **wildfly** system service:

```
$ sudo systemctl start wildfly
```

Once the service is started, we can check the status by running below command:

```
$ sudo systemctl status wildfly
```

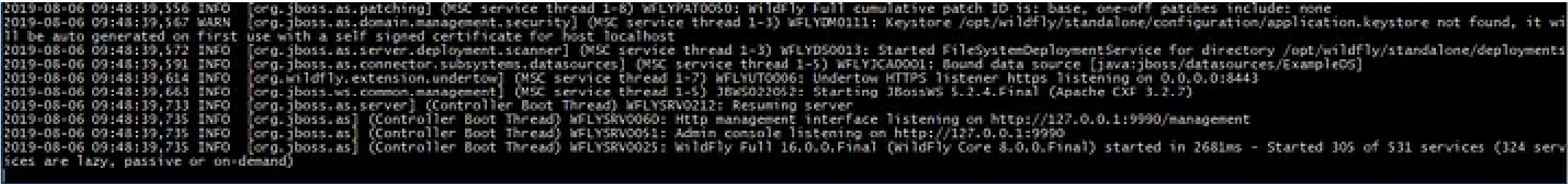
If the service started successfully, we should see something like below:

```
● wildfly.service - The WildFly Application Server
   Loaded: loaded (/etc/systemd/system/wildfly.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2019-08-06 09:48:36 EDT; 1s ago
     Main PID: 11918 (launch.sh)
    CGroup: /system.slice/wildfly.service
            └─11918 /bin/bash /opt/wildfly/bin/launch.sh standalone standalone.xml 0.0.0.0
            └─11919 /bin/sh /opt/wildfly/bin/standalone.sh -c standalone.xml -b 0.0.0.0
            └─11996 java -D[Standalone] -server -Xms64m -Xmx512m -XX:MetaspaceSize=96M -XX:MaxMetaspaceSize=256m -Djava.net.preferIPv4Stack=true -Djboss.modules.system.pkgs=org.jboss.byteman ...
```

The **Active** status, as highlighted, above verifies that the service is up and running.

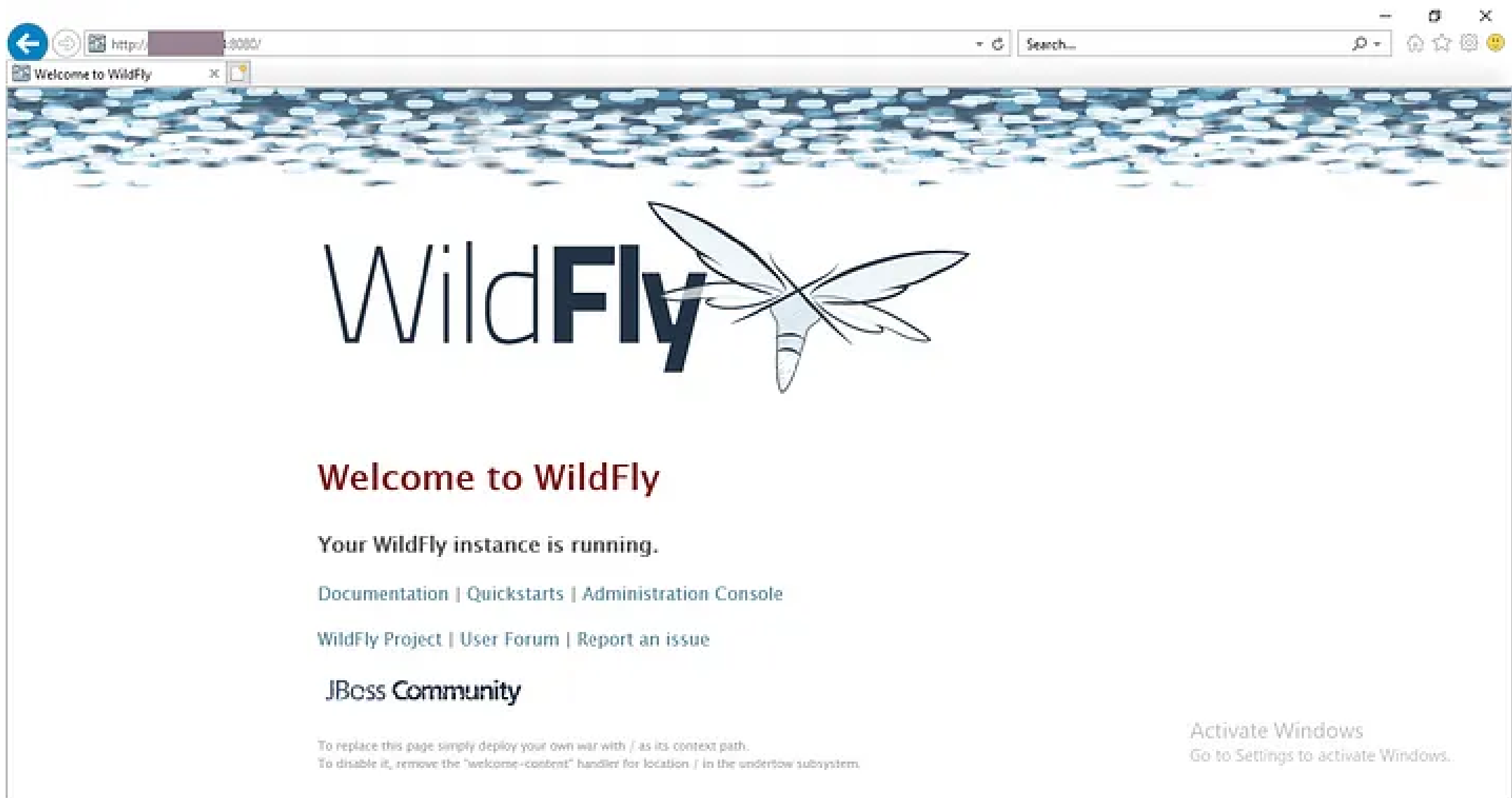
We can also tail the Wildfly server logs with below command:

```
$ sudo tail -f /opt/wildfly/standalone/log/server.log
```



Now access Wildfly server at:

<http://<instance-public-ip>:8080/>



Step 8 — Configure Wildfly Management Console

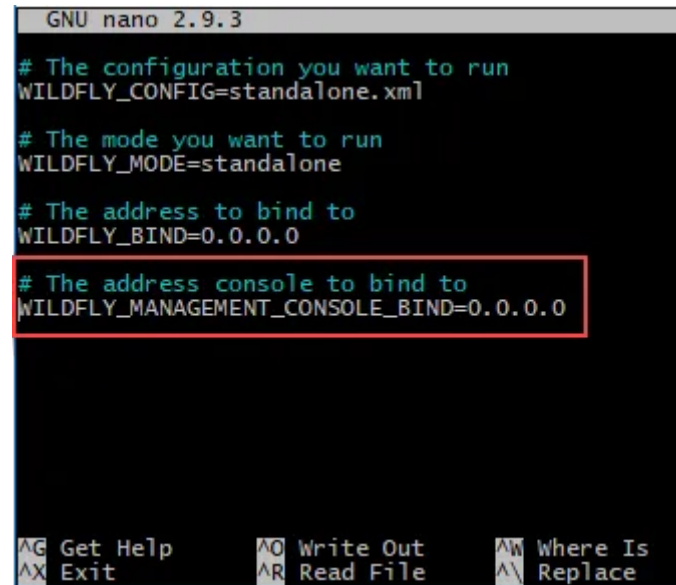
The Wildfly management console allows us to manage different aspects of the Wildfly server. e.g. configuring subsystems, server monitoring, managing deployments or access control.

By default, the management console is not made accessible remotely. To make it accessible remotely, we have to make small changes in 3 files. So let's start

Open **wildfly.conf** file under **/etc/wildfly/** directory

```
$ sudo nano /etc/wildfly/wildfly.conf
```

Add a line at the end as shown below:



```
GNU nano 2.9.3

# The configuration you want to run
WILDFLY_CONFIG=standalone.xml

# The mode you want to run
WILDFLY_MODE=standalone

# The address to bind to
WILDFLY_BIND=0.0.0.0

# The address console to bind to
WILDFLY_MANAGEMENT_CONSOLE_BIND=0.0.0.0

^G Get Help      ^O Write Out     ^W Where Is
^X Exit          ^R Read File     ^\ Replace
```

Save and exit the file.

Now open **launch.sh** in **/opt/wildfly/bin/** directory and change its contents as shown below:

```
$ sudo nano /opt/wildfly/bin/launch.sh
```

```
#!/bin/bash

if [ "x$WILDFLY_HOME" = "x" ]; then
    WILDFLY_HOME="/opt/wildfly"
fi

if [[ "$1" == "domain" ]]; then
    $WILDFLY_HOME/bin/domain.sh -c $2 -b $3 -bmanagement $4
else
    $WILDFLY_HOME/bin/standalone.sh -c $2 -b $3 -bmanagement $4
fi
```

Save and exit the file

Finally, open Wildfly's system service definition file (*wildfly.service*) under */etc/systemd/system/* and make the changes as shown below:

```
$ sudo nano /etc/systemd/system/wildfly.service
```

```
[Unit]
Description=The WildFly Application Server
After=syslog.target network.target
Before=httpd.service

[Service]
Environment=LAUNCH_JBOSS_IN_BACKGROUND=1
EnvironmentFile=-/etc/wildfly/wildfly.conf
User=wildfly
Group=wildfly
LimitNOFILE=102642
PIDFile=/var/run/wildfly/wildfly.pid
ExecStart=/opt/wildfly/bin/launch.sh $WILDFLY_MODE $WILDFLY_CONFIG $WILDFLY_BIND $WILDFLY_MANAGEMENT_CONSOLE_BIND
StandardOutput=null

[Install]
WantedBy=multi-user.target
```

Wrote 17 lines

Get Help WriteOut Read File Prev Page
Exit Justify Where Is Next Page

Save and exit the file

Since we have changed the service unit file, we have to inform the ***systemd manager***

```
$ sudo systemctl daemon-reload
```

Now restart the ***wildfly*** service

```
$ sudo systemctl restart wildfly
```

Once restarted, Access the Wildfly management console at:

<http://<instance-public-ip>:9990>



We can successfully access the management console but as shown above, we need a management user to login.

We can use the ***add-user.sh*** script, packaged with Wildfly server distribution, to create a management user. Run the script with below command:

```
$ sudo /opt/wildfly/bin/add-user.sh
```

```
What type of user do you wish to add?
a) Management User (mgmt-users.properties)
b) Application User (application-users.properties)
(a): a

Enter the details of the new user to add.
Using realm 'ManagementRealm' as discovered from the existing property files.
Username : 
Password recommendations are listed below. To modify these restrictions edit the add-user.properties configuration file.
- The password should be different from the username
- The password should not be one of the following restricted values {root, admin, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s), 1 digit(s), 1 non-alphanumeric symbol(s)
Password : wFLYDM0102: Password should have at least 1 non-alphanumeric symbol.
Are you sure you want to use the password entered yes/no? yes
Re-enter Password :
What groups do you want this user to belong to? (Please enter a comma separated list, or leave blank for none)[ ]:
About to add user 'kashat' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user  to file '/opt/wildfly/standalone/configuration/mgmt-users.properties'
Added user  to file '/opt/wildfly/domain/configuration/mgmt-users.properties'
Added user  with groups  to file '/opt/wildfly/standalone/configuration/mgmt-groups.properties'
Added user  with groups  to file '/opt/wildfly/domain/configuration/mgmt-groups.properties'
Is this new user going to be used for one AS process to connect to another AS process?
e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.
yes/no? yes
To represent the user add the following to the server-identities definition <secret value=  />
```

Once prompted, select to add a management user and provide your desired **username** and **password**.

In the last prompt, for enabling remote access for this user write **yes** or **y**.

After providing the required information, the script will verify the user creation as shown above.

We need to restart Wildfly server so our user is picked up during boot.

```
$ sudo systemctl restart wildfly
```

Now if we access the management console again, it will prompt for HTTP basic authorization.

Provide the management user credentials that we created above and click ‘ok’:



Activate Windows
Go to Settings to activate Windows.

Once successfully logged in, we land inside the management console

The screenshot shows the WildFly HAL Management Console in a web browser. The browser's address bar displays `http://localhost:9990/console/index.html`. The console's header includes the WildFly logo and a navigation menu with links to **Homepage**, **Deployments**, **Configuration**, **Runtime**, **Patching**, and **Access Control**. The main content area is titled "WildFly Application Server" and features four primary sections:

- Deployments** (Add and manage deployments): Includes a "Deploy an Application" link and a "Start" button. The instructions are: "Deploy an application to the server", "1. Use the 'Add Deployment' wizard to deploy the application", and "2. Enable the deployment".
- Configuration** (Configure subsystem settings): Includes a "Create a Datasource" link and a "Start" button. The instructions are: "Define a datasource to be used by deployed applications. The proper JDBC driver must be deployed and registered.", "1. Select the Datasources subsystem", "2. Add a Non-XA or XA datasource", and "3. Use the 'Create Datasource' wizard to configure the datasource settings".
- Runtime** (Monitor server status): Includes a "Monitor the Server" link and a "Start" button. The instructions are: "View runtime information such as server status, JVM status, and server log files.", "1. Select the server", and "2. View log files or JVM usage".
- Access Control** (Manage user and group permissions for management operations): Includes an "Assign User Roles" link and a "Start" button. The instructions are: "Assign roles to users or groups to determine access to system resources.", "1. Add a new user or group", and "2. Assign one or more roles to that user or group".

At the bottom of the console, there is a "Patching" section and a "Need Help?" link. A Windows watermark in the bottom right corner reads "Activate Windows Go to Settings to activate Windows." The footer of the console shows the version "3.12.Final", a "Tools" menu, and a "Settings" link.

We have just done basic setup of *wildfly* server and enabled/configured remote access to administration and management console. This concludes our tutorial.

P.S. As a bonus, I have included some helpful resources below to get you started with Java EE application development. To test your Java EE application, you can deploy it on the Wildfly installation that you did using this tutorial.

Thanks for reading. Please feel free to comment.