

# Automating All The Things

Sebastian Feldmann



# Automation

"Automate until the job is boring as hell"

# Automation

- System Configuration
- Software Deployment
- Software Preparation
- Putting it all together



ANSIBLE

*“App deployment, configuration management and orchestration  
all from one system. Powerful automation that you can learn quickly.”*

# About Ansible

- Written in Python
- Using SSH
- No agents
- YAML

# Ansible Modules

- Package management (apt, yum...)
- Command execution (command, shell)
- Service management (start, stop...)
- File handling (copy, template)
- SCM (git, Bazaar, Subversion)

# Ansible

- Inventory
- Playbook
- Role
- Tasks

# Ansible Examples

- System Management
- Software deployment



# System Management

# Inventory

```
ops/  
+- inv/  
  +- integration/  
  +- local/  
  +- production/
```

best practice inventory setup

# Inventory

```
[webserver]  
192.168.1.111
```

```
[dbserver]  
192.168.1.222
```

ops/inv/integration/hosts

# Playbook

```
- hosts: webservers
  sudo: true
  roles:
    - apache
```

ops/install.webserver.yml

# Role

```
ops/  
+- roles/  
| +- apache/  
|   +- files/  
|   +- handlers/  
|   +- tasks/  
|   +- templates/  
+- install.webserver.yml
```

best practice role setup

# Tasks

```
- name: install apache
  action: apt
    pkg=apache2
    state=present
    force=yes

- name: copy mpm prefork conf
  copy: dest=/etc/apache2/mods-available/mpm_prefork.conf
        src=mpm_prefork.conf
  notify: restart apache
  tags: apache
```

ops/roles/apache/tasks/main.yml

# Handler

```
- name: restart apache  
  action: service name=apache2 state=restarted
```

ops/roles/apache/handlers/main.yml

# Recap

- Create inventories
- Install playbook
- Apache role
- Install tasks



# Execute Ansible

```
$ ansible-playbook -i inv/integration/hosts install.webserver.yml
```

execution example

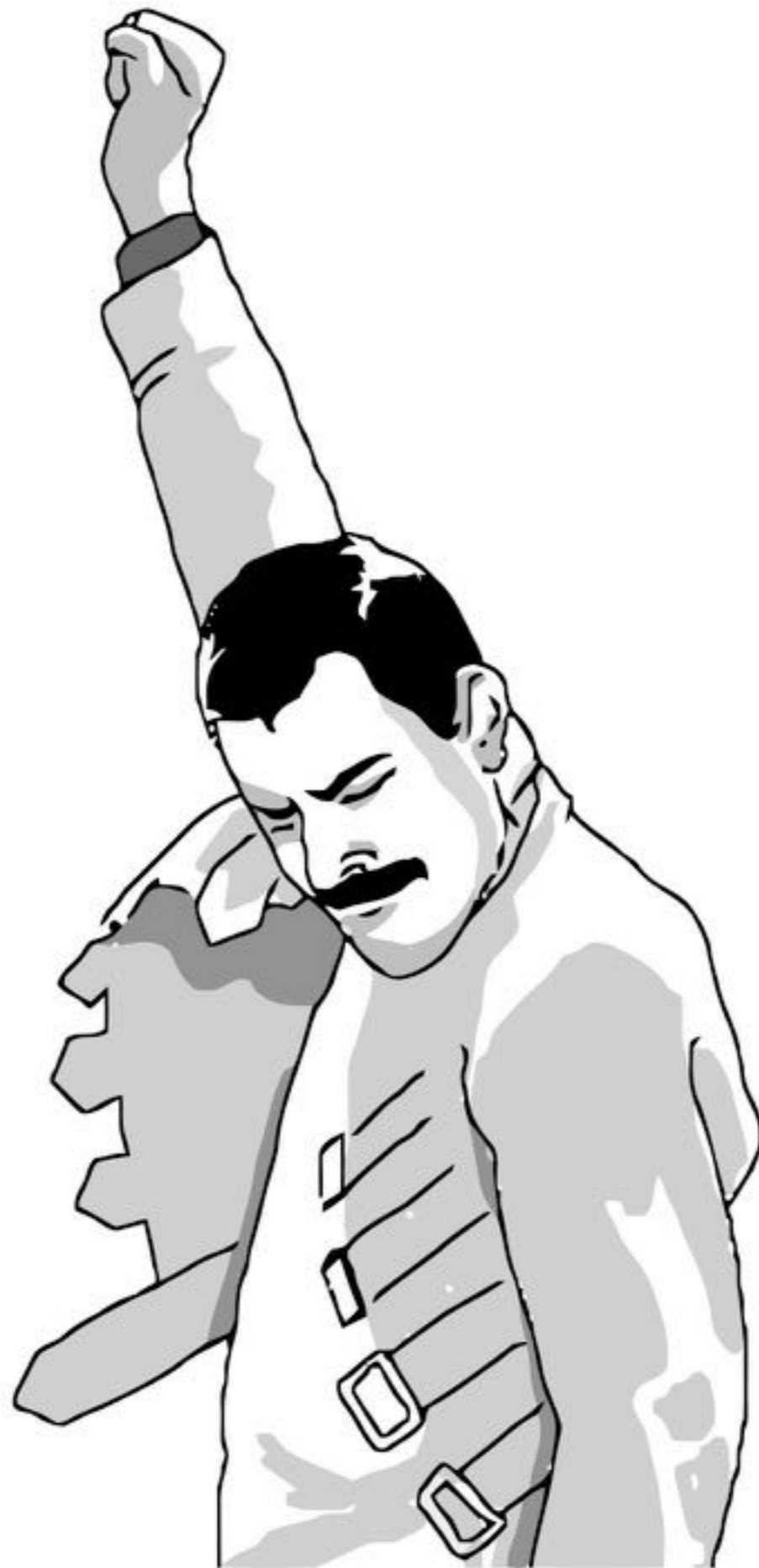
# Execute Ansible

```
$ ansible-playbook -i inv/integration/hosts  
--check install.webserver.yml
```

dry run with --check

```
$ ansible-playbook -i inv/integration/hosts  
--limit 192.168.1.111 install.webserver.yml
```

restrict execution with --limit



# Software Deployment

# Software Deployment

- Copy app to server
- Activate deployed app
- Allow quick rollbacks

# Half Way There

- Enhance the inventory
- New playbook
- New role

# Inventory

```
ops/  
+- inv/  
    +- integration/  
        +- group_vars/  
        +- host_vars/
```

add variables

# Variables

```
system:  
  environment: integration  
  
projectx:  
  domain: www.projectx.int
```

ops/inv/integration/group\_vars/webserver



# Playbook

```
- hosts: webservers
  sudo: true
  roles:
    - projectx
```

ops/deploy.projectx.yml

# Role

```
ops/  
+- roles/  
|   +- projectx/  
|     +- files/  
|     +- handler/  
|     +- tasks/  
|     +- templates/  
+- deploy.projectx.yml
```

best practice role setup

# Tasks

```
# create the directory for the current app version
- name: Create version directory
  file: path=/var/www/projectx/{{ version }}
        state=directory

# copy the prepared app to your webservers
- name: Copy files to server
  synchronize: src={{ app }}
                dest=/var/www/projectx/{{ version }}
                perms=yes
                recursive=yes
                delete=yes
                owner=no
                group=no
```

ops/roles/projectx/main.yml

# Tasks

```
# deploy the vhost configuration for our project
- name: Deploy vhost configuration
  action: template
    src=projectx.conf
    dest=/etc/apache2/sites-available/projectx.conf
  notify: restart apache
  tags: rollback
```

ops/roles/projectx/main.yml

# Tasks

```
# create a symlink to activate the vhost
# could be done with "command: a2ensite projectx.conf"
# but this would always trigger an apache restart
# even if nothing changes
- name: Activate vhost configuration
  file: dest=/etc/apache2/sites-enabled/010-projectx.conf
       src=/etc/apache2/sites-available/projectx.conf
       state=link
  notify: restart apache
```

ops/roles/projectx/tasks/main.yml

# Template

```
<VirtualHost *:80>
  ServerName {{ projectx.domain }}
  DocumentRoot /var/www/my-project/{{ version }}/htdocs

  setenv APP.ENVIRONMENT {{ system.environment }}

  <Directory /var/www/projectx/{{ version }}/htdocs>
    AllowOverride All
    Require all granted
  </Directory>
</VirtualHost>
```

ops/roles/projectx/templates/projectx.conf

# Recap

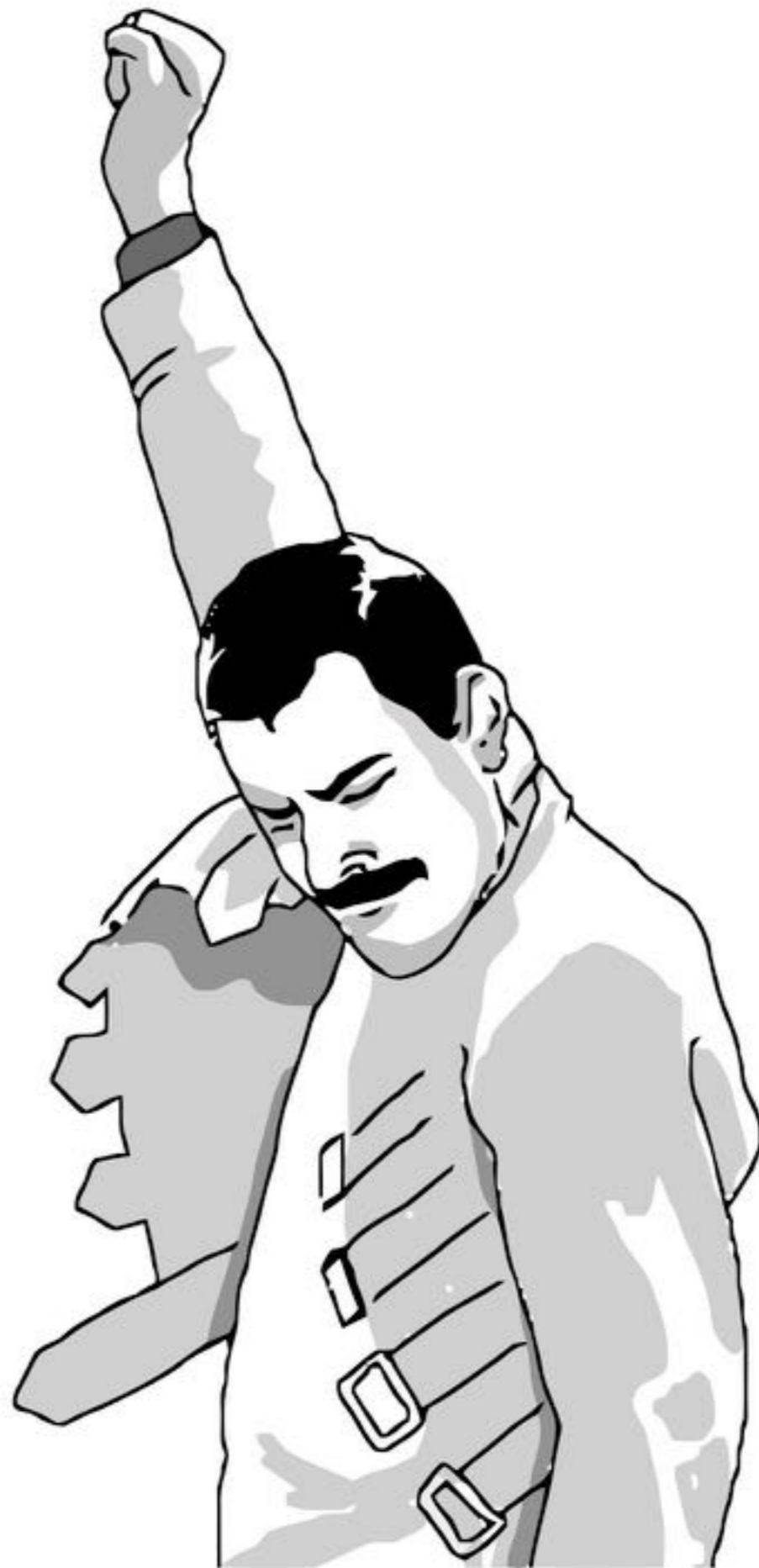
- Inventory variables
- Deploy playbook
- Project role
- Deployment tasks

# Execute Ansible

```
$ ansible-playbook -i inv/integration/hosts  
  --extra-vars "version=1.0.1 app=/some/local/dir/app/"  
  deploy.projectx.yml
```

deployment execution





# Software Preparation



"Ant can be used to pilot any type of process which can be described in terms of targets and tasks"

# Ant

- Written in Java
- XML configuration
- Targets (tasks)
- Bundled with Java

# Ant Configuration

```
<?xml version="1.0" encoding="UTF-8"?>
<project name="projectx" default="build">
  <target name="build" depends="target1, target2, target3" />
  <target name="target1">...</target>
  <target name="target2">...</target>
  ...
</project>
```

build.xml

# Composer

```
<target name="composer-install">  
  <exec executable="composer" failonerror="true">  
    <arg value="install" />  
    <arg value="--no-interaction" />  
  </exec>  
</target>
```

build.xml

# PHPUnit

```
<target name="phpunit" description="Run unit tests">  
  <exec executable="phpunit" failonerror="true"></exec>  
</target>
```

build.xml

# Ant Tasks

- Minify Assets
- Cache warmup (Templates, DIContainer...)
- ...



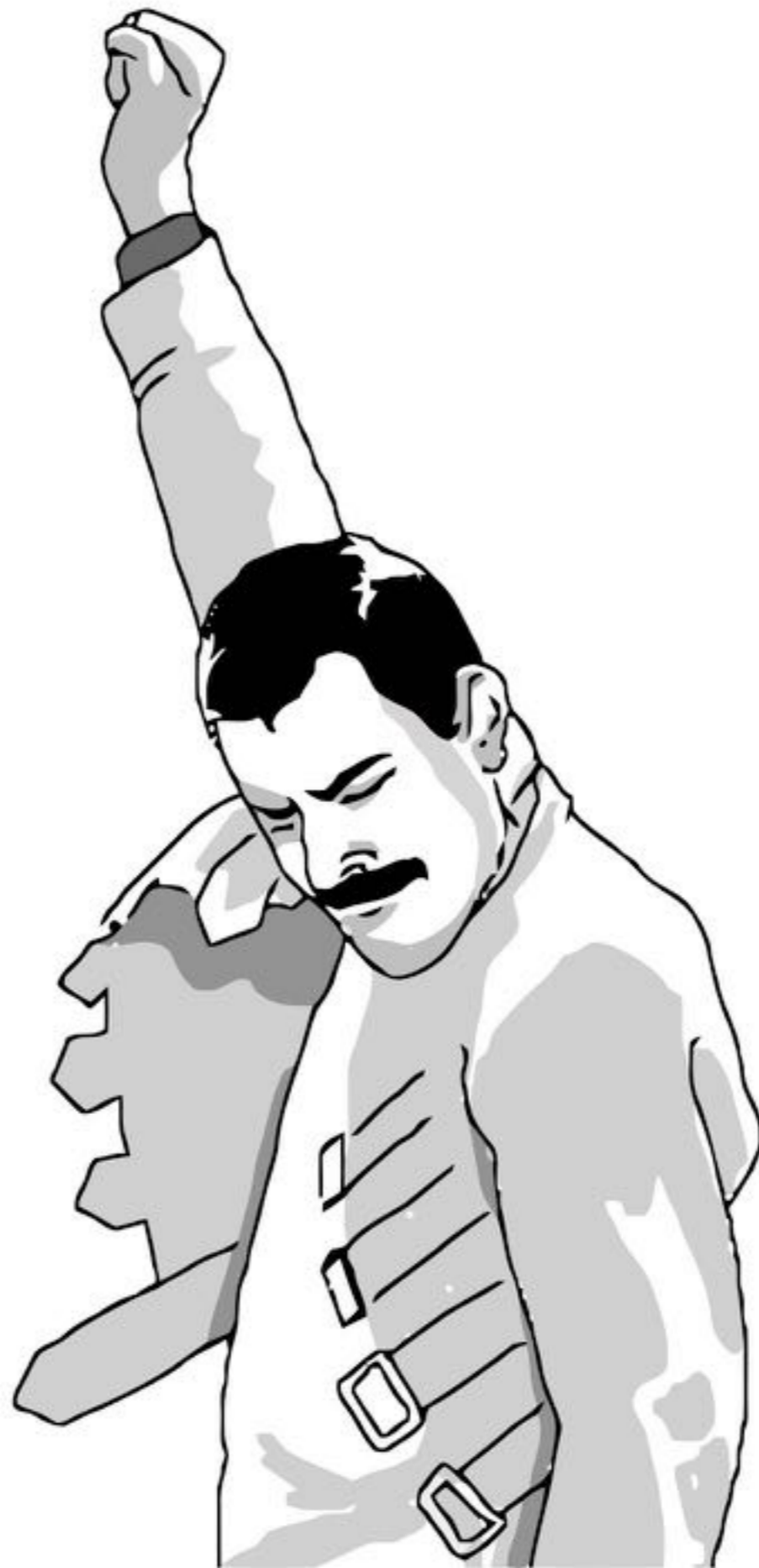
# Ant Execution

```
$ ant
```

execute default target

```
$ ant phpunit
```

execute specific target



And finally...

Putting It All Together



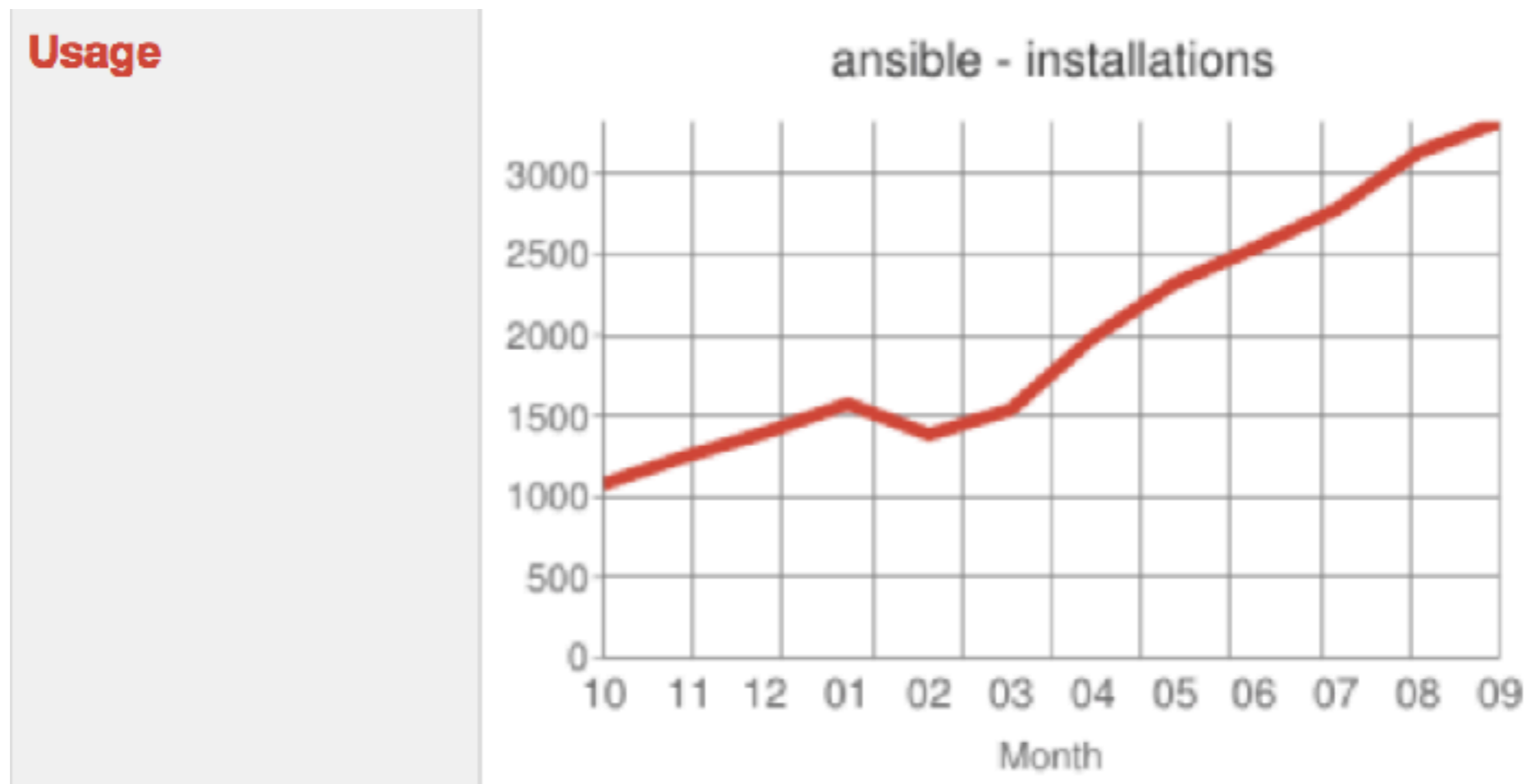
# Jenkins

*“Build great things at any scale”*

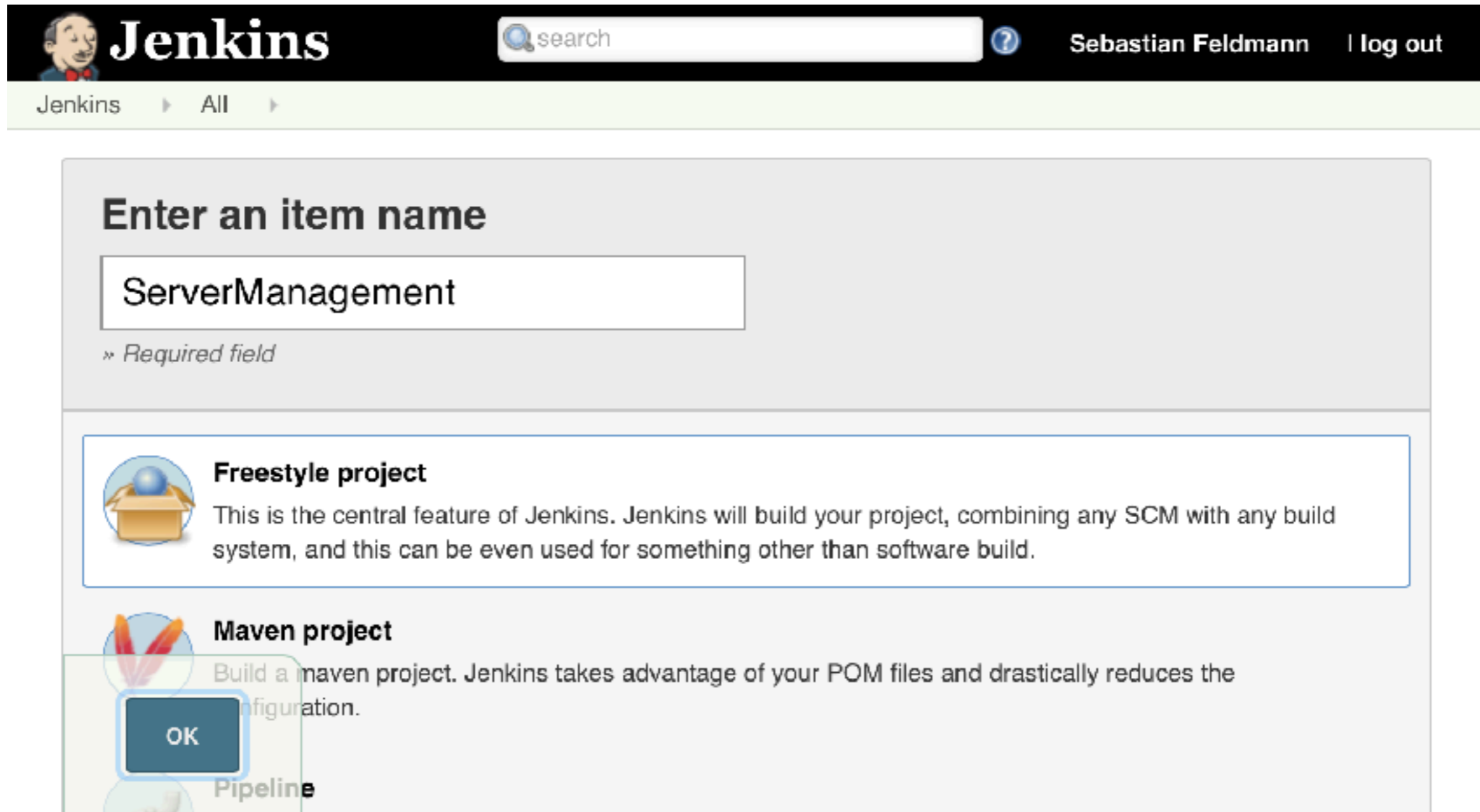
# Jenkins

- Written in Java
- Build server
- Projects / Jobs
- Jetty (build in webserver)

# Ansible Plugin



# System Management



The screenshot shows the Jenkins web interface. At the top, there is a navigation bar with the Jenkins logo, a search bar containing the text 'search', and the user name 'Sebastian Feldmann' with a 'log out' link. Below the navigation bar, there is a breadcrumb trail: 'Jenkins > All >'. The main content area is titled 'Enter an item name' and contains a text input field with the text 'ServerManagement'. Below the input field, there is a small red asterisk and the text '» Required field'. Below the input field, there are three project type options: 'Freestyle project', 'Maven project', and 'Pipeline'. The 'Freestyle project' option is highlighted with a blue border and contains a description: 'This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.' The 'Maven project' option is partially visible and contains the text 'Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.' The 'Pipeline' option is partially visible and contains the text 'Pipeline'. An 'OK' button is visible in the bottom left corner of the 'Maven project' option.

**Jenkins** search Sebastian Feldmann | log out

Jenkins > All >

**Enter an item name**

ServerManagement

» Required field

**Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**Maven project**  
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

**Pipeline**

OK



# Parameter Setup

The image shows two overlapping 'Choice Parameter' configuration windows in a graphical user interface. The top window is titled 'Choice Parameter' and has a red close button and a help icon. It contains the following fields:

- Name:** Inventory
- Choices:** integration, production

The bottom window is also titled 'Choice Parameter' and has a red close button and a help icon. It contains the following fields:

- Name:** Playbook
- Choices:** install.webserver, install.dbserver
- Description:** Choose playbook to execute

On the left side of the top window, there is a vertical sidebar with labels: 'Name', 'Choices', and 'Description'. The 'Description' label is partially obscured by the bottom window.

# SCM Config

**Source Code Management**

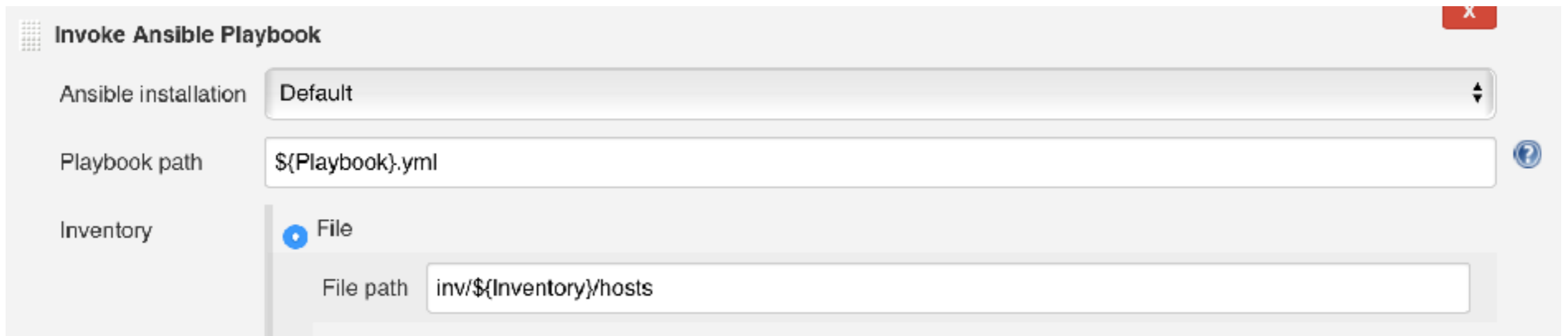
None  
 CVS  
 CVS Projectset  
 Git

Repositories

Repository URL  

use some SCM to manage your Ansible project

# Plugin Config



The screenshot shows a configuration window titled "Invoke Ansible Playbook". It contains the following fields:

- Ansible installation:** A dropdown menu with "Default" selected.
- Playbook path:** A text input field containing the template `$(Playbook).yml`.
- Inventory:** A section with a radio button selected for "File".
- File path:** A text input field containing the template `inv/$(Inventory)/hosts`.

The use of `$(...)` in the paths demonstrates parameter usage.

parameter usage

# Job Execution

The screenshot displays the Jenkins web interface. At the top, there is a black navigation bar with the Jenkins logo on the left, a search bar in the center, and the user name 'Sebastian Feldmann' and a 'log out' link on the right. Below this is a light green breadcrumb trail showing 'Jenkins' and 'SystemManagement'. On the left side, there is a vertical sidebar with several menu items, each with an icon: 'Back to Dashboard' (green arrow), 'Status' (magnifying glass), 'Changes' (notepad), 'Workspace' (folder), 'Build with Parameters' (play button), 'Delete Project' (red prohibition sign), 'Configure' (gear), and 'Move' (hand with arrow). The main content area is titled 'Project SystemManagement' and contains the text 'This build requires parameters:'. Below this text are two dropdown menus: 'Inventory' with 'integration' selected and 'Playbook' with 'install.webserver' selected. Each dropdown has a small blue arrow icon. Below the dropdowns is a dark blue 'Build' button.

Jenkins  Sebastian Feldmann | log out

Jenkins > SystemManagement

- Back to Dashboard
- Status
- Changes
- Workspace
- Build with Parameters
- Delete Project
- Configure
- Move

## Project SystemManagement

This build requires parameters:

Inventory

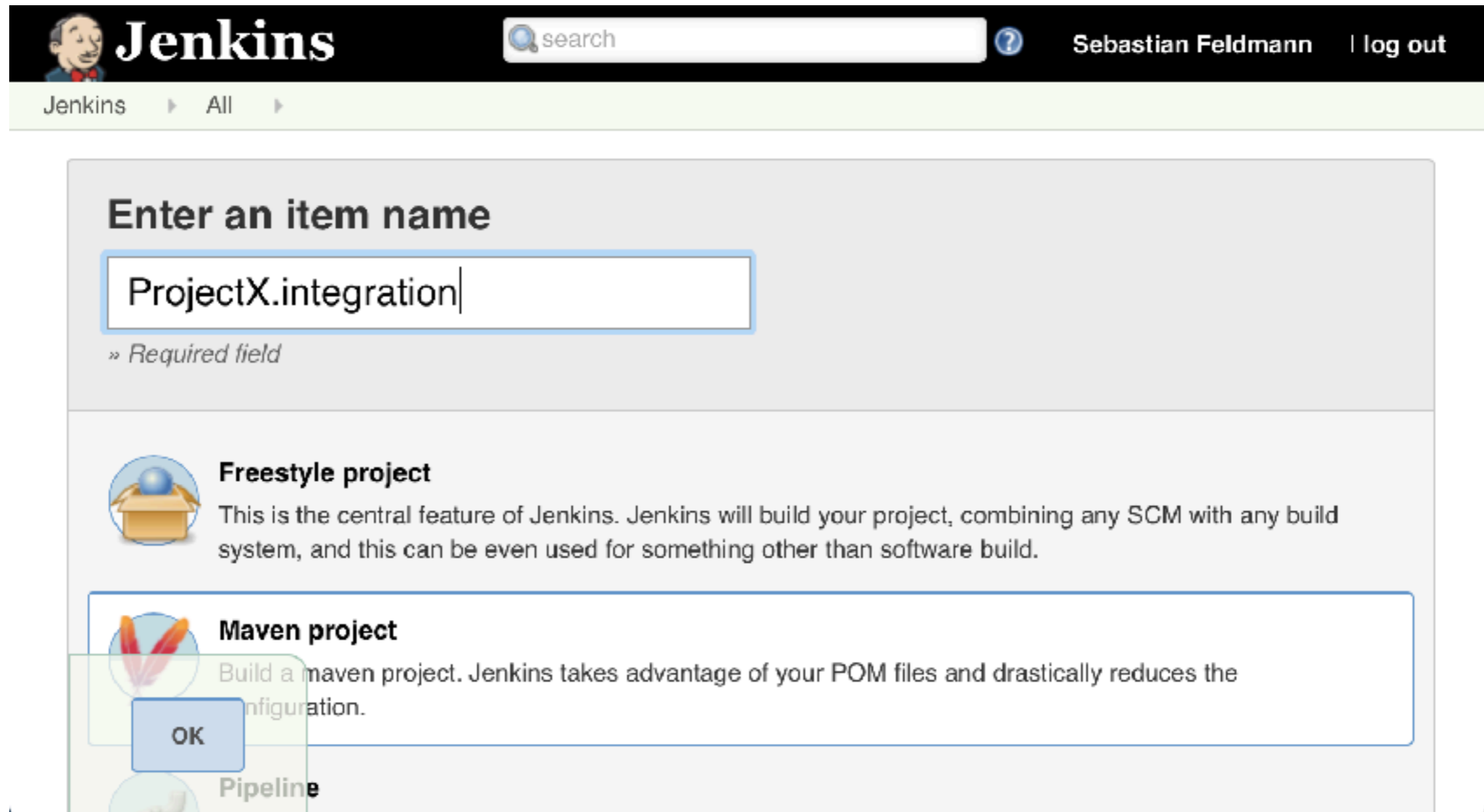
Choose inventory to execute

Playbook

Choose playbook to execute

**Build**

# Software Deployment



The screenshot shows the Jenkins web interface. At the top, there is a black header with the Jenkins logo (a cartoon man) and the word "Jenkins" in white. To the right of the logo is a search bar with the text "search" and a magnifying glass icon. Further right, the user's name "Sebastian Feldmann" and a "log out" link are visible. Below the header is a light green breadcrumb trail: "Jenkins > All >".

The main content area is a light gray box. At the top of this box is the heading "Enter an item name". Below this heading is a text input field containing the text "ProjectX.integration". Below the input field is a small red asterisk and the text "» Required field".

Below the input field are three project type options, each with an icon and a description:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Maven project**: Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration. (This option is highlighted with a blue border and a blue "OK" button is overlaid on it.)
- Pipeline**: (Partially visible at the bottom)

# Parameter Setup

String Parameter X ?

Name	<input type="text" value="version"/>	?
Default Value	<input type="text"/>	?
Description	<input type="text" value="Version you want to deploy"/>	?

[Safe HTML] [Preview](#)

# SCM Config

**Source Code Management**


None

CVS

CVS Projectset

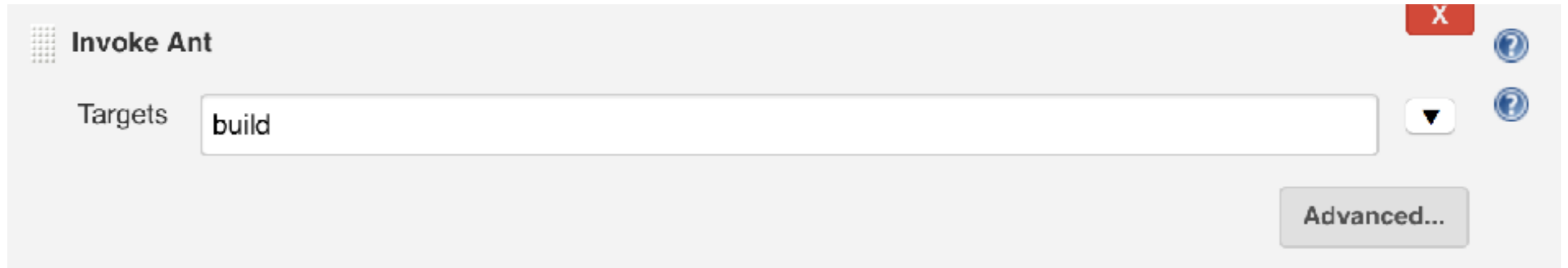
Git

Repositories

Repository URL  

configure projectx SCM

# Ant



Ant support out of the box



# Plugin Config



The screenshot shows a configuration window titled "Invoke Ansible Playbook". It contains three main sections: "Ansible installation" with a dropdown menu set to "Default"; "Playbook path" with a text input field containing "deploy.projectx.yml"; and "Inventory" with a radio button selected for "File" and a sub-section "File path" with a text input field containing "inv/integration/hosts".

Ansible installation	Default
Playbook path	deploy.projectx.yml
Inventory	<input checked="" type="radio"/> File
	File path: inv/integration/hosts

setup ansible playbook and inventory

# Ansible Extra Variables

Extra Variables


Key	version
Value	\$(version)
Hidden variable in build log	<input type="checkbox"/>

Key	app
Value	\$(WORKSPACE)
Hidden variable in build log	<input type="checkbox"/>









Add Extra Variable

setup version and application directory

# Job Execution

 **Jenkins**  Sebastian Feldmann | log out

Jenkins > ProjectX.integration

-  [Back to Dashboard](#)
-  [Status](#)
-  [Changes](#)
-  [Workspace](#)
-  [Build with Parameters](#)
-  [Delete Project](#)
-  [Configure](#)
-  [Move](#)

## Project ProjectX.integration

This build requires parameters:

version

Version you want to deploy

**Build**

# Jenkins Log

```
Buildfile: /var/lib/jenkins/jobs/projectx.integration/workspace/build.xml
...
composer-install:
  [exec] Loading composer repositories with package information
  [exec] Nothing to install or update
  [exec] Generating autoload files
...
[workspace] $ ansible-playbook deploy.projectx.yml -i inv/integration/hosts

PLAYBOOK: deploy.projectx.yml *****

PLAY [webserver] *****

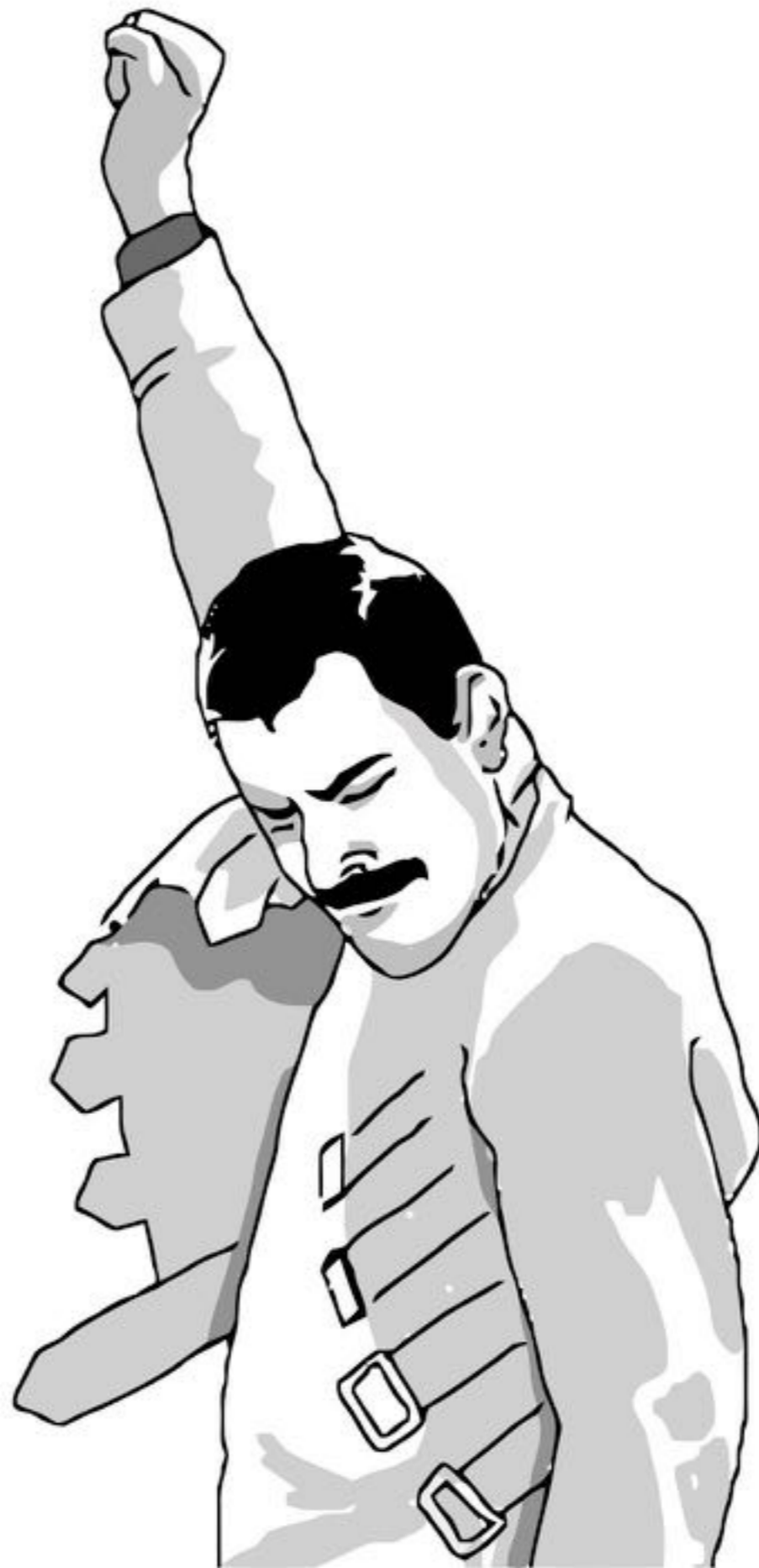
TASK [install vhost] *****
ok: [webserver]
...

BUILD SUCCESSFUL
```

# Recap

- System Management Build
- Software Deployment Build
- Execute everything with a single click

Is it getting boring yet?



thank you



# Sebastian Feldmann

phpbu

<https://phpbu.de>



User Group  
Munich



**CHECK24**



@movetodevnull



sebastianfeldmann

# Q & A



09:00 - 10:00



FORUM 7



**Mastering git - Insights & Tips & Tricks**

**Sebastian Feldmann**, *CHECK24*