ADDO

ALL DAY DEVOPS

NOVEMBER 6, 2019

David Maillet

Delivery Flexibility: Pipeline as Code with Jenkins & Groovy

ADDO
ALL DAY DEVOPS

Who Am I

- IT Director, American Tire Distributors
- Background
  - DevOps, CI/CD
  - Performance Engineering
  - Software Architect
  - Java Developer

David Maillet

American Tire Distributors

~80,000 customers

~200 suppliers

AMERICAN TIRE DISTRIBUTORS®

- 140 distribution centers
- 5,000 employees
- $4.5B in revenue (2017)

ADDO
ALL DAY DEVOPS

Continuous Delivery Pipeline

Problem & Solution

- **Goal**:
  Quality attributes (maintainability, security, reliability) with the CI/CD platform

- **Solution**:
  Jenkins Multibranch Pipeline as code

Multibranch Pipelines

Implement different pipelines in different branches

Master:	**Pipeline v.1**
Develop:	**Pipeline v.2**
Feature-XYX:	**Pipeline v.3**

Technologies Used in This Sample

- Linux VM
- Jenkins
  - Pipeline plug-in
  - Blue Ocean plug-in
- GitHub source code repository
- Git

Multibranch Pipelines

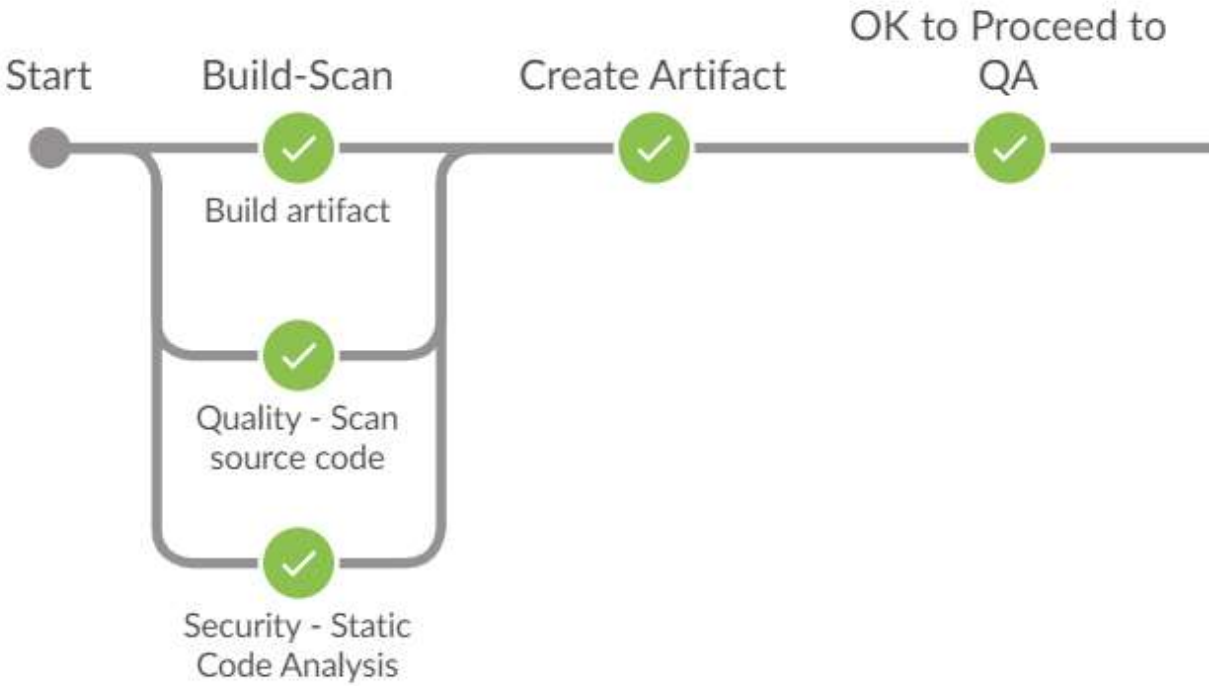
Pipeline defined in code – **Jenkinsfile**:

- Checked into source control with application code
- Text file

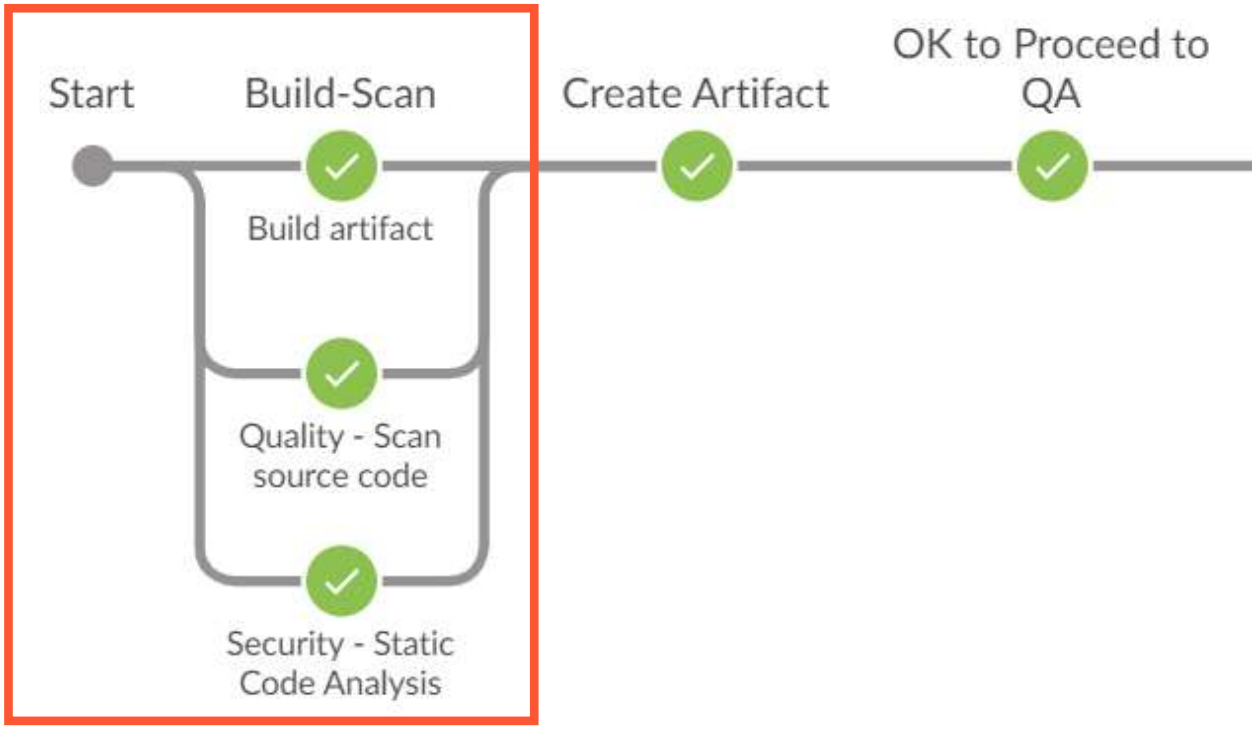
Benefits:

- Code review on the Pipeline
- Audit trail for the Pipeline
- Single source of truth for the Pipeline

Sample Pipeline

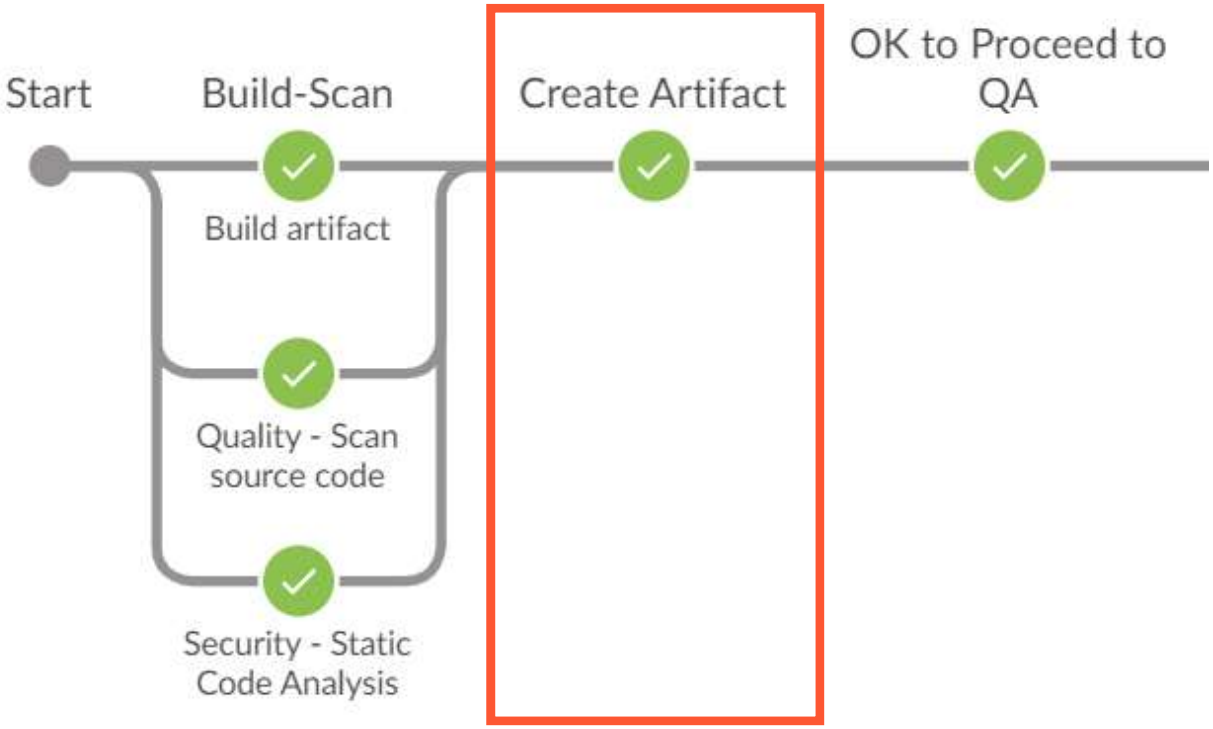


Sample Pipeline – Parallel Steps



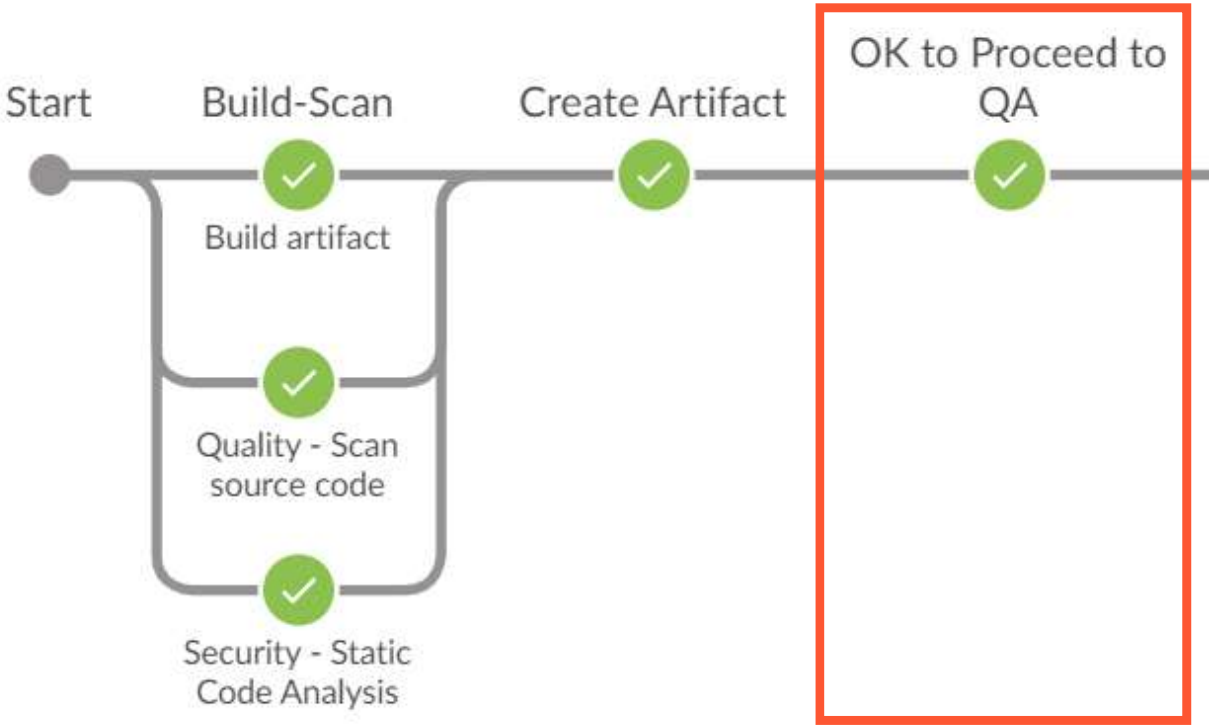
```
node {
  stage('Build-Scan’) {
     parallel(
       'Build artifact’: {
           echo "Build step"
           sh 'mvn clean verify'
        },
        'Quality - Scan source code’: {
           echo "Scan step"
           sh 'echo “Scan report shows success" > scan.txt'
        },
        'Security - Static Code Analysis’: {
           echo "Static Code Analysis step"
           sh 'echo “Security success" > security-scan.txt'
        }
     )
  }
```

Sample Pipeline – Linear Steps



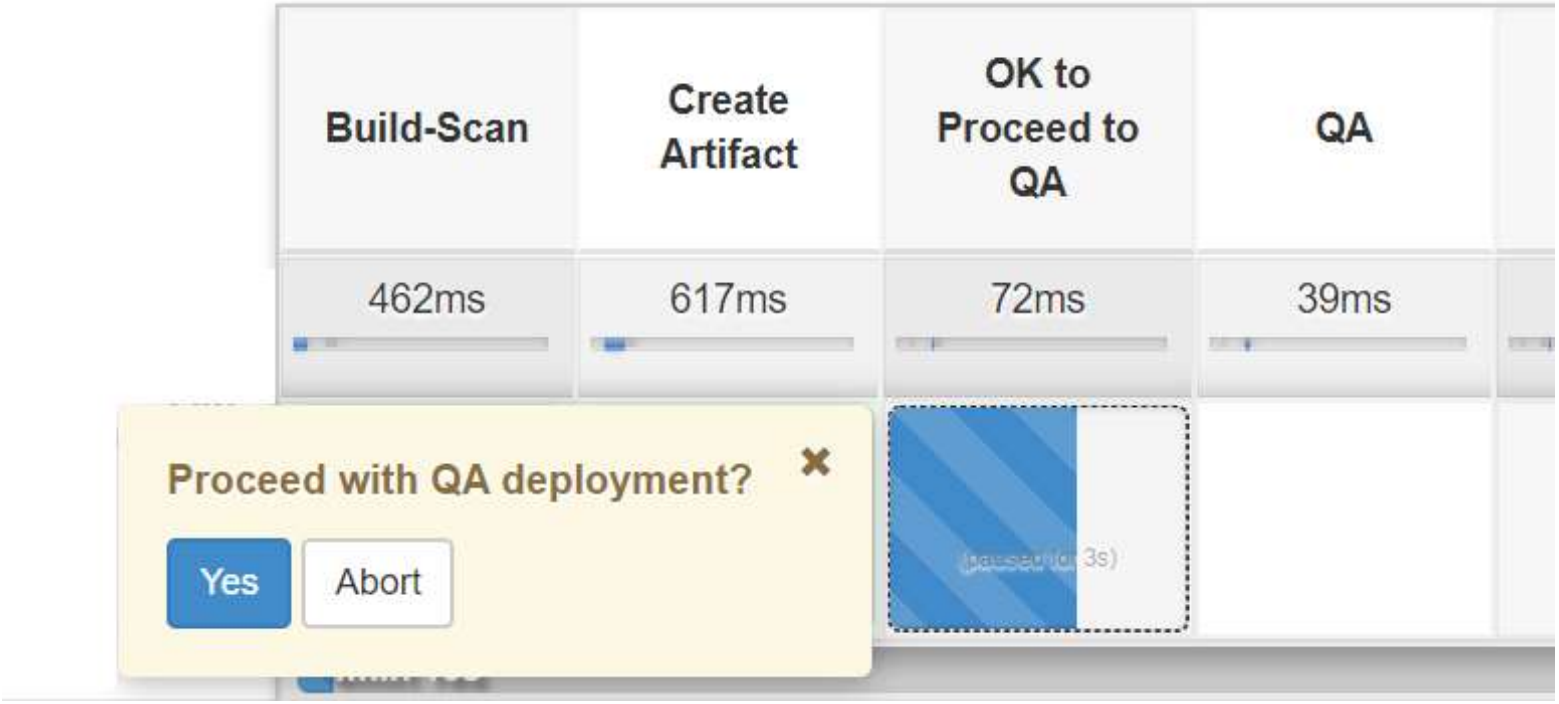
```
40   stage('Create Artifact’) {
41      echo "Creating artifact named: ${ARTIFACT_NAME}“
42      sh "tar -cvzf ${ARTIFACT_NAME} build.txt“
43      dir("${ARTIFACT_PATH}") {
44         sh "mv ../${ARTIFACT_NAME} .“
45      }
46   }
```

Sample Pipeline – Pause for Human



```
47   stage('OK to Proceed to QA’) {
48      input message: 'Proceed with QA deployment?', ok: 'Yes’
49   }
```

Support for Gates



Audit Approvals

```
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (OK to Proceed to QA)
[Pipeline] input
Proceed with QA deployment?
Yes or Abort
Approved by David Maillet
[Pipeline] }
[Pipeline] // stage
```

Conditional Logic

For example:

If using the Master branch

    Do something

Otherwise

    Do something else

```
stage('QA') {
    echo 'Deploy to QA’
    if (env.BRANCH_NAME == 'master') {
        echo 'Execute Full QA testing’
    } else {
        echo 'Execute Quick smoke test’
    }
}
```

Modularize for Maintainability

Define functions for commonly reused features.

Enables long-term faster modifications.

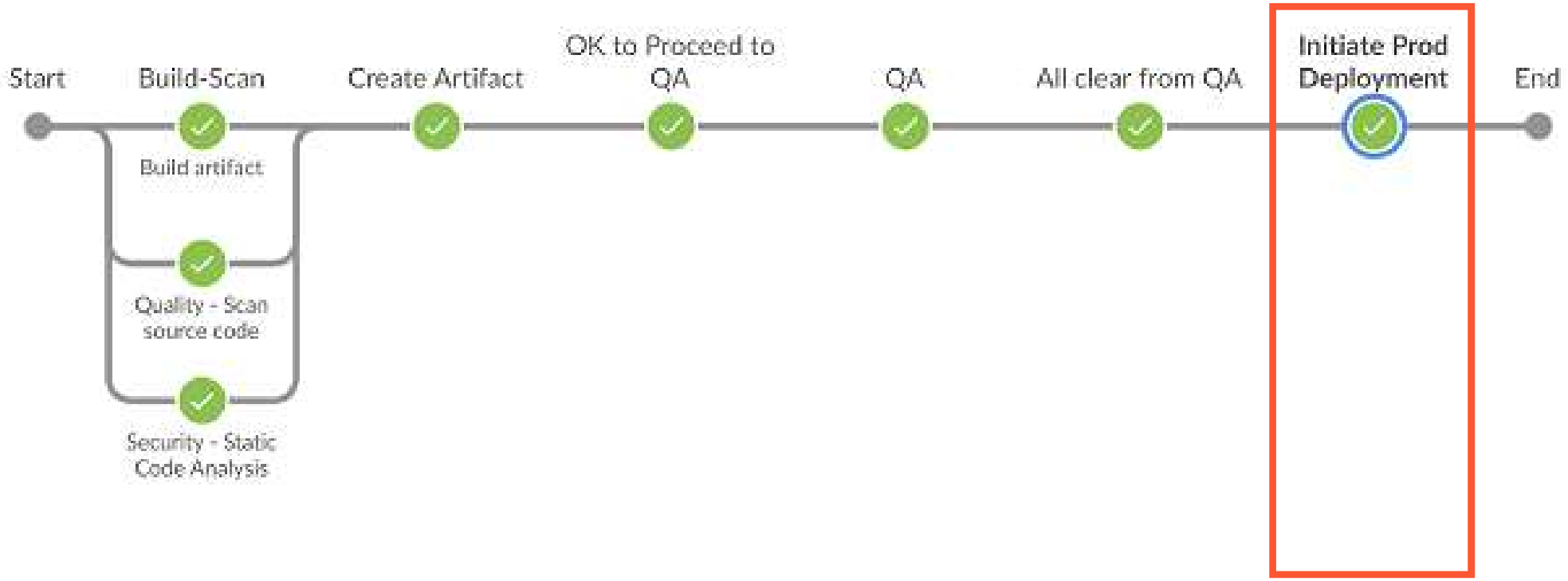
Our example:

- Pipeline sends message to communication channel

```
47   stage('OK to Proceed to QA’) {
48       notifyTeam("ECommerce",
                    "${ARTIFACT_NAME} is ready for QA")
49       input message: 'Proceed with QA deployment?', ok: 'Yes'
50   }
```

```
21  // *** Define functions ***
22
23  def notifyTeam(channel, message) {
24    echo "Message for ${channel}: ${message}“
25    build job: "Operations/Notification",
26          propagate: false, wait: false,
27          parameters:
28           [string(name: 'channel', value: "${channel}"),
29            string(name: 'text', value: "${message}")]
30  }
```

Call Another Pipeline



```
stage('Initiate Prod Deployment') {
   build job: "Operations/Prod-Deploy",
    propagate: true, wait: true, parameters:
      [string(name: 'artifact', value: "${ARTIFACT_NAME}"),
       string(name: 'branch', value: "${env.Branch_Name}"),
       string(name: 'pipelineBuildNumber’,
            value: "${env.BUILD_ID}")]
}
```

Pipeline Job

Multiple Branches

```
$ git branch
* develop
  master
```

Multiple Branches – Jenkins Multiple Jobs

Folder name: My Pipeline
Sample CI/CD

Branches (2) Pull Requests (0)

S	W	Name ↓	Last Success	Last Failure	Last Duration
		develop	24 min - #19	36 min - #17	4 min 28 sec
		master	1 mo 0 days - #2	1 mo 0 days - #1	3.1 sec

Create a New Branch

```
$ git checkout -b feature-123
Switched to a new branch 'feature-123'

$ git push --set-upstream origin feature-123
Total 0 (delta 0), reused 0 (delta 0)
* [new branch]      feature-123 -> feature-123
Branch 'feature-123' set up to track remote branch 'feature-123' from 'origin'.

$ git branch
  develop
* feature-123
  master
```

Jenkins Auto-created Job

Folder name: My Pipeline
Sample CI/CD

Branches (3) Pull Requests (0)

S	W	Name ↓	Last Success	Last Failure	Last Duration
		develop	1 hr 16 min - #19	1 hr 28 min - #17	4 min 28 sec
		feature-123	N/A	N/A	N/A
		master	1 mo 1 day - #2	1 mo 1 day - #1	3.1 sec

```
23  node {
24    stage('Build-Scan') {
25       parallel(
26         'Build artifact': {
27             echo "Build step"
28             sh 'echo "The build file" > build.txt'
29          },
30          'Quality - Scan source code': {
31             echo "Scan step"
32             sh 'echo "Scan report shows success" > scan.txt'
33          },
34          'Security - Static Code Analysis': {
35             echo "Static Code Analysis step"
36             sh 'echo "Security success" > security-scan.txt'
37          }
38       )
39    }
```

Add Security Code Scan

```
'Security - Static Code Analysis': {
    build job: "Security/Code-Scan", propagate: true, wait: true,
          parameters:
             [string(name: 'branch',
                     value:"${env.Branch_Name}"),
              string(name: 'pipelineBuildNumber',
                     value: "${env.BUILD_ID}")]
}
```


Each branch has its own Pipeline

Folder name: My Pipeline
Sample CI/CD

Branches (3) Pull Requests (0)

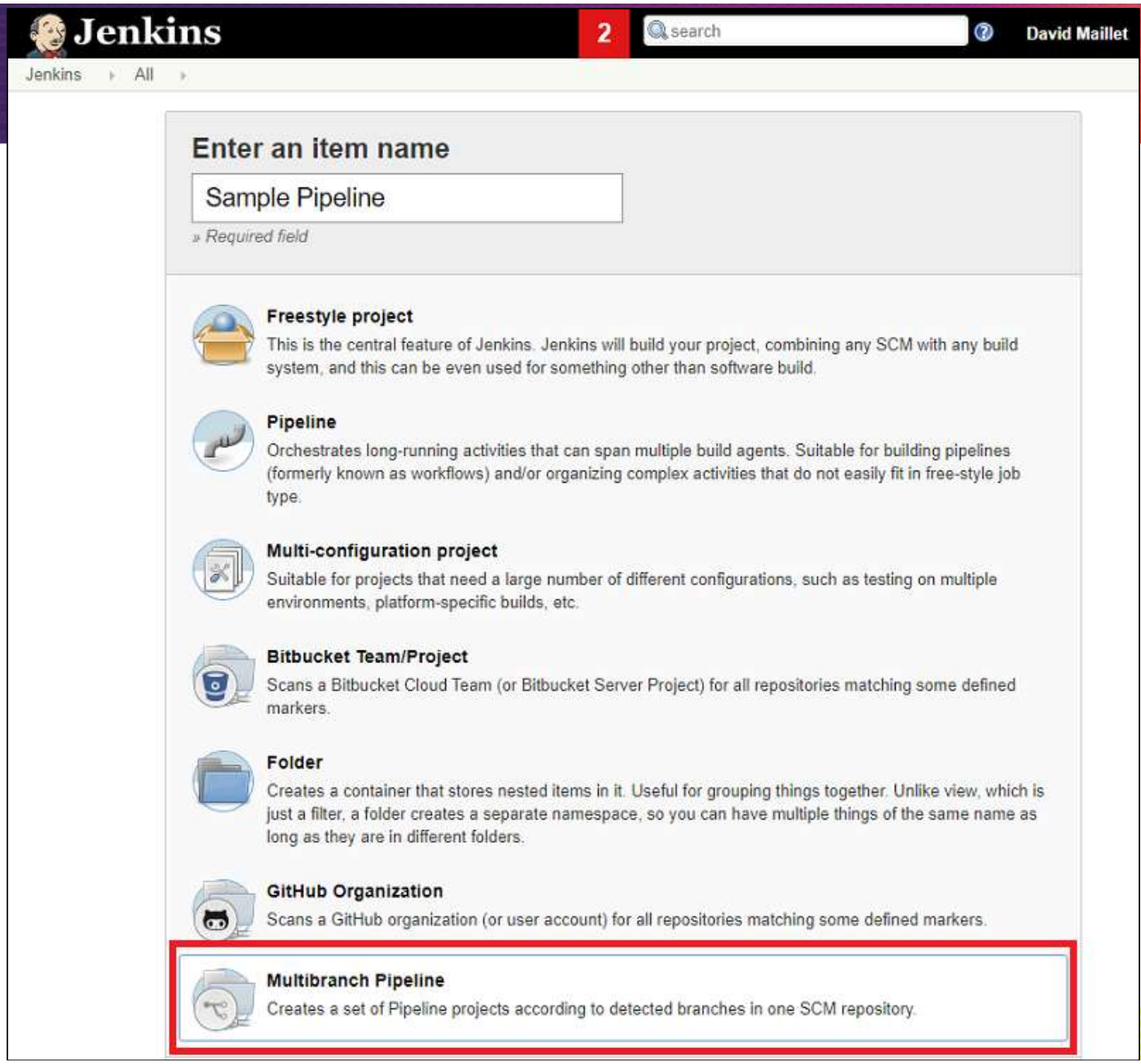
S	W	Name ↓	Last Success	Last Failure	Last Duration
		develop	2 days 8 hr - #19	2 days 8 hr - #17	4 min 28 sec
		feature-123	1 min 1 sec - #4	2 days 6 hr - #2	50 sec
		master	1 mo 3 days - #2	1 mo 3 days - #1	3.1 sec

Setup Jenkins

- Create Linux sandbox VM
- Open firewall for port 8080 (for Jenkins)
- Install Jenkins
- Configure Jenkins
  with Pipeline and Blue Ocean plug-in's

ADDO
ALL DAY DEVOPS

Create a new Multibranch Pipeline



Configure the Pipeline

Branch Sources

GitHub

Credentials

/****** (github)

Add

User

Repository HTTPS URL

Repository HTTPS URL

https://github.com/ /sample.git

Configure the Multibranch Pipeline

Build Configuration

Mode: by Jenkinsfile

Script Path: Jenkinsfile

Summary

- Everything is code
- Different pipelines for different branches
- Modularize for:
  - Maintainability
  - Reuse
  - Access control
- Pipeline code sits with application code
- Self-service

ADDO
ALL DAY DEVOPS

SPONSORS

Sponsorship packages for All Day DevOps are available. If your organization is interested, please contact us for details.

DIAMOND SPONSORS

sonatype

StateFarm

GOLD SPONSORS

vmware

Fannie Mae

NORTHROP GRUMMAN

COMMUNITY ADVOCATES AND VIEWING PARTY SPONSORS

XebiaLabs Enterprise DevOps

cloudbees

Carnegie Mellon University Software Engineering Institute

TIGERA

CONTINO

NowSecure

MEDIA SPONSORS

DZone A DEVADA MEDIA PROPERTY

TechBeacon

DevOps.com

MediaOps

ITSP MAGAZINE

the cyberwire

ADDO
ALL DAY DEVOPS

SUPPORTERS

ADDO
ALL DAY DEVOPS

Contact Info

David Maillet

IT Director

- DevOps, CI/CD
- Performance Engineering
- Quality Engineering
- Robotic Process Automation

Sample File:

- https://github.com/dmaillet63/sample

Contact:

- **Email**: dmaillet@atd-us.com
- **Twitter**: @dmaillet
- **Office**: (704) 912-2384
- **LinkedIn**: https://www.linkedin.com/in/davidmaillet/
- **Website**: https://www.atd-us.com/