



検索 開発 CI/CD ストーリー CLOUD ブログについて プライバシーポリシー ホーム[※]

Jenkins と Git を連携させ、ビルド・リリースのためのCI/CD環境を構築する

BY TRACPATH • 2020-12-22 • DEVELOPMENT

Jenkins ▸ すべて ▸

Enter an item name

» Required field

フリースタイル・プロジェクトのビルド

もっとも汎用性の高いJenkinsの中核機能です。任意のSCMからソースコードをチェックアウトし、任意のビルドシステムでプロジェクトがビルドできます。往々にして、ソフトウェアのビルド以外にも様々な仕事の自動化に利用することができます。

パイプライン

複数のビルドスレーブにまたがる長時間に渡る処理を編成します。(以前は、ワークフローとして知られていた)パイプラインの構築に適しており、フリースタイルジョブでは上手く扱えない複雑な処理を容易に編成することができます。

マルチ構成プロジェクトのビルド

複数の環境でのテストや、プラットフォームごとのビルドなどといった、多数の異なる構成が必要なプロジェクトに適しています。

GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.

GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.

OK

Multibranch Pipeline

この記事シェアする:



はじめに
 本記事の対象となる方
 レッスン1. Gitプロジェクトを準備する
 レッスン2. ジョブをつくる
 tracpath側に公開鍵が登録されていない場合
 レッスン3. Gitに入っているファイルを扱う

チーム開発のノウハウ資料ダウンロード

資料請求

tracpathのサービス内容とチーム開発に必要なノウハウを紹介した限定資料

TWITTERでフォロー

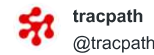
@tracpathさんをフォロー



レッスン4. コミットをポーリングする
レッスン5. ビルドをジョブ化するステップ
おわりに

TWITTER

@tracpathさんのツイート

tracpath
@tracpathGit Extensions 3.5 がリリース [tracpath.com/vcs-news/2021/...](https://tracpath.com/vcs-news/2021/)**Git Extensions 3.5 がリリース**概要 2021年4月26日、Git Extensions 3.5 ...
tracpath.com

2021年4月30日



埋め込む

Twitterで表示

はじめに

メジャーなバージョン管理ツールであるGitと、CI/CDツールの代表格であるJenkins。この2つはいずれも無料で使えることから、様々な開発現場において広く利用されています。特にビルド・リリースという定型作業においては非常に有効で、開発効率の向上に貢献します。これまで単純に自動化作業のみでJenkinsを使っていた方でも、Gitとの組み合わせた使い方を学べば、より便利に使っていただけることかと思います。

本記事の対象となる方

- ✓ 既にJenkinsの環境がある
- ✓ Jenkinsで簡単なジョブの作成をしたことがある
- ✓ Gitの基本的な使い方を知っている(クローン、コミット、プッシュなど)
- ✓ GitでSSH接続の設定を行ったことがある(クライアントの設定だけで十分です)

レッスン1. Gitプロジェクトを準備する

まずはJenkinsで扱うGitプロジェクトを準備します。ホスティングサービスは問いませんが、今回はtracpathを使います。

[→tracpathの登録方法及び、リポジトリ作成方法のページへ](#)

新しいリポジトリを作り、テストとして下記のコマンドを入れたバッチファイル(test.batという名前にします)を準備し、プッシュしておいてください。

```
test.bat
```

```
@echo off  
echo Hello world
```

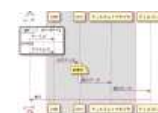
最近の投稿

**Python実践演習 ～ライブラリの使い方～**

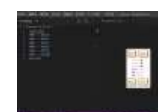
2021-02-12

**機械学習、ディープラーニングを始めた方、Pythonの基礎**

2021-02-10

**PlantUMLの実践応用 (シーケンス図)**

2020-12-25

**PlantUMLの基礎**

2020-12-24

**Jenkins と Git を連携させ、ビルド・リリースのためのCI/CD環境を構築する**

2020-12-22

DEVOPS 導入 : TRACPATH



また、SSH接続のURLを後で使います。tracpathのダッシュボードからURL
左側のプルダウンでSSHを選び、このページを開いたままにしておくか、
URLをどこかに控えておきましょう。

デフォルトリポジトリ

リポジトリ URL: SSH git@git-practice.tracpath.com

ジャンプ: ▼ リビジョン: ▼

名称: test.html.txt



プロジェクト管理

バージョン管理

バグ管理

インシデント管理

DevOps に一歩近づく、開発環境
その全てを、1つのサービスで。
ソフトウェア開発プロジェクト支援ツール
tracpath (トラックパス)

今すぐ無料で試してみる >

60秒で
登録完了!

レッスン2. ジョブをつくる

では、続いて新規ジョブを作ります。Jenkinsのダッシュボードにログインしたら、左側のツリーから「新規ジョブ作成」を選択します。

Jenkins

検索

Administrator

ログアウト

新規ジョブ作成

開発者

ビルド履歴

Jenkinsの管理

My Views

Lockable Resources

New View

ビルドキュー

ビルド待ち

ビルド実行状態

1 待機中

2 待機中

Jenkinsへようこそ!

Create an agent or configure a cloud to set up distributed builds. Learn more.

新しいジョブを作成してください。

ジョブの作成画面になるので、ジョブ名を入れます。入力したら「フリースタイル・プロジェクトのビルド」を選択します。選択したら、画面下部にあ

人気の記事



WindowsにJobSchedulerをインストールする手順を解説

ON 2015-10-15

01



IPAのウェブアプリケーション・セキュリティ要件を確認する（セキュリティ要件確認支援ツール）

ON 2015-07-31

02



商用利用OK！オープンソースの日本語フリーフォントをご紹介します

ON 2015-07-09

03



スマホやタブレットに最適なグラフ描画ライブラリの紹介(オープンソース)

ON 2015-08-07

04



オバマ大統領「プログラミングを学ぶことは自分のためだけではない。国の未来がかかっているのだ。」

ON 2015-05-28

u5

https://tracpath.com/works/development/jenkins_and_git/

3/12

る「OK」を押してください。

Jenkins > すべて

Enter an item name

Test_Job

Required field

フリースタイル・プロジェクトのビルド

もっとも汎用性の高いJenkinsの中核機能です。任意のSCMからソースコードをチェックアウトし、任意のビルドシステムでプロジェクトがビルドできます。往々にして、ソフトウェアのビルド以外にも様々な仕事の自動化に利用することができます。

パイプライン

複数のビルドスレーブにまたがる長時間に渡る処理を編成します。(以前は、ワークフローとして知られていた)パイプラインの構築に適しており、フリースタイルジョブでは上手く扱えない複雑な処理を容易に編成することができます。

マルチ構成プロジェクトのビルド

複数の環境でのテストや、プラットフォームごとのビルドなどといった、多数の異なる構成が必要なプロジェクトに適しています。

GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.

GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.

OK

Multibranch Pipeline

tracpath

DevOps 時代のソフトウェア開発プロジェクトとは？

ソフトウェア開発プロジェクト支援ツール「tracpath (トラックパス)」で、数多くのプロジェクトをサポートしてきた株式会社オープングループが、これからの「DevOps 時代」にふさわしい、開発についてご紹介致します。

相談してみる >

～採用情報～

こちらのサイトを運営している株式会社オープングループでは、技術者の求人を募集しています。

開発者のためのバージョン管理・プロジェクト管理ツールtracpathと一緒に作りませんか？

採用情報の詳細はこちら

ジョブの作成が完了すると、引き続きジョブの設定画面に移行します。ここではジョブの動作内容を細かく決めるメニューが並んでいます。

Jenkins

検索

Administrator ログアウト

Jenkins > TestJob

General ソースコード管理 ビルド・トリガ ビルド環境 ビルド ビルド後の処理

説明

[HTMLをエスケープ] プレビュー

☐ GitHub project

☐ This build requires lockable resources

☐ Throttle builds

☐ ビルドのパラメータ化

☐ 古いビルドの破棄

☐ ビルド無効化

☐ ビルドを並行実行

高度な設定...

ソースコード管理

保存 Apply

設定画面を下にスクロールしていくと、「ソースコード管理」というカテゴリがありますので、「Git」を選んでください。選ぶと、リポジトリのURLの



入力欄が出てきます。

ここに先ほどtracpathで取得したURLを入れます。今回はSSH接続でGitを使ってみます。GitにはHTTPSとSSHの2つの接続方法がありますが、一般的にはセキュリティが高いSSHが良く使われます。

SSH接続を使うためには、設定の途中で秘密鍵をJenkinsのクライアントPCに登録しなければならないので、場合により管理者権限が必要になってきます。こちらは管理者に相談してみてください。

JenkinsのクライアントPCにSSH秘密鍵が登録されており、tracpathで作ったリポジトリがCloneできる環境になっていれば、Jenkins側では認証情報の指定は特に必要ないので、URLだけでクローン出来ます。

URLを入力したら、「保存」を押してジョブの設定を完了します。

tracpath側に公開鍵が登録されていない場合

tracpathのダッシュボードから、一般設定 -> SSH鍵 に移動します。





下記のエリアに公開鍵の中身を貼り付けて、「追加」を押します。コメントは空欄でOKです。

SSH 公開鍵の追加

公開鍵:

ssh-rsa, ssh-ed25519, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, ecdsa-sha2-nistp521, ssh-dss で始まる文字列を貼り付けてください

コメント:

追加

これで公開鍵の登録は完了です。

では、ジョブを実行してみましょう。ジョブの実行が成功したら、コンソール出力を見てみてください。



今回はチェックアウトのみなので、下記のようにコンソール出力に最新のコミットメッセージが出力されれば成功です。

```
Checking out Revision e6304b152277e1420490166bd8d910fc9ca96455 (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f e6304b152277e1420490166bd8d910fc9ca96455 # timeout=10
Commit message: "Merge branch 'feature_B'"
> git.exe rev-list --no-walk e6304b152277e1420490166bd8d910fc9ca96455 # timeout=10
Finished: SUCCESS
```

レッスン 3. Gitに入っているファイルを扱う



では、続いてチェックアウトしたファイルを使用する方法を学びましょう。
今回はあらかじめ入れておいたバッチファイルを実行するジョブにしてみます。

ジョブの設定画面に入り、設定画面中ほどにあるビルドというカテゴリに、「ビルド手順の追加」というボタンがありますので、そこから「Windowsバッチコマンドの実行」を選んでください。



バッチコマンドを入力するエリアが出てくるので、下記のコマンドを入れてみましょう。

```
%WORKSPACE%\%test.bat
```

下記のようになればOKです。そのまま保存して設定画面を閉じてください。



では、さっそく実行してみましょう。実行が成功したら、コンソール出力を確認してみてください。test.batに入力していたHello worldが出力されてい



れば成功です。

● コンソール出力

```

ユーザーAdministratorが実行
Running as SYSTEM
ビルドします。 ワークスペース: C:\Users\¥.jenkins\workspace¥Test_Job
The recommended git tool is: NONE
No credentials specified
> git.exe rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
> git.exe config remote.origin.url git@git-practice.tracpath.com # timeout=10
Fetching upstream changes from git@git-practice.tracpath.com
> git.exe --version # timeout=10
> git --version # 'git version 2.18.0.windows.1'
> git.exe fetch --tags --progress -- git@git-practice.tracpath.com +refs/heads/*:refs/remotes/origin/* # timeout=10
> git.exe rev-parse "refs/remotes/origin/master^{commit}" # timeout=10
Checking out Revision a94f6b57fe2a6e17f0009b484307453a2f64bebe (refs/remotes/origin/master)
> git.exe config core.sparsecheckout # timeout=10
> git.exe checkout -f a94f6b57fe2a6e17f0009b484307453a2f64bebe # timeout=10
Commit message: "File replace"
> git.exe rev-list --no-walk e6304b152277e1420490166bd8d910fc9ca96455 # timeout=10
[Test_Job] $ cmd /c call C:\Users\¥.jenkins\workspace¥Test_Job\O:\Users\¥.jenkins\workspace¥Test_Job¥Test.bat
C:\Users\¥.jenkins\workspace¥Test_Job>C:\Users\¥.jenkins\workspace¥Test_Job¥Test.bat
Hello world
Finished: SUCCESS

```

レッスン4. コミットをポーリングする

毎回ビルドを手動で行っているのは非効率なので、自動でビルドを開始できるような方法を考えます。かといって単にスケジュールで実行すると、変更が無い場合でもビルドされてしまうので、無駄なPCリソースを食ったり、不要な成果物が増えたりする原因となります。そこで、Gitでリポジトリに変更が加わったことを定期的にチェックし、変更があればビルドするという方法を学んでいきたいと思います。これをポーリングと呼びます。

一見難しそうな内容ですが、設定は簡単にできます。設定画面を開き、ビルド・トリガのセクションを開き、「SCMをポーリング」を選んでください。

ビルド・トリガ

- ☐ リモートからビルド (例: スクリプトから)
- ☐ GitHub hook trigger for GITScm polling
- ☒ SCMをポーリング

スケジュール

No schedules so will only run due to SCM changes if triggered by a post-commit hook

- ☐ post-commitフックを無視
- ☐ 他プロジェクトの後にビルド
- ☐ 定期的に行

スケジュールという欄が出てくるので、ここにスケジュールを入力します。

ここに定期実行を設定するコマンドを入力しますが、コマンドのフォーマットは下記のようにになっています。(各文字列の間に半角スペース)
これは主にUNIXで用いられる、cronと呼ばれる書き方になります。



分 時間 日 月 曜日

- ・ 特定しない場合は * を入れる
- ・ ランダムの場合は H を入れる（複数のジョブを動かしていると、前のジョブの実行が終わらない場合もあるので、実行時間に幅を持たせるために使用を推奨）

これを踏まえて、いくつか例を挙げます。これを参考にスケジュールを作ってみてください。

毎日9時に実行

H 9 * * *

平日毎日15時に実行

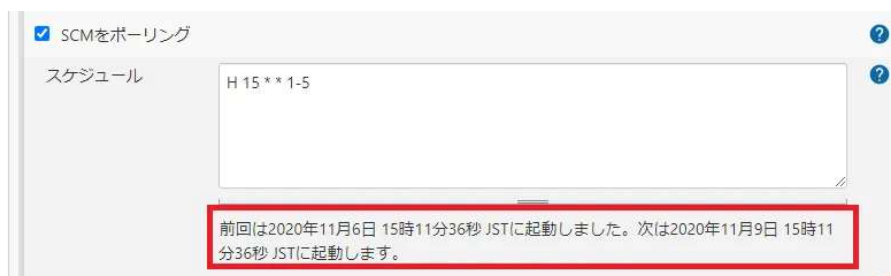
H 15 * * 1-5

土日の10時に実行

H 10 * * 0,6

cronは少々癖があるフォーマットになりますが、詳しい内容はヘルプがあります。（英語のみ）また、jenkins cron等で検索をかければ、日本語のサイトなども見つかりますので、参考にしてください。

入力したものが意図する内容になっているかは、入力後にテキストボックスからフォーカスを外すと、すぐ下に実行される日付や時間が例として出てきますので、そちらを確認してみてください。



保存したら、完了です。正しく起動したかを見たい場合、もし変更が無ければビルドが行われなため、ログに残りませんので、その際にはジョブの左



側のツリーから、「Gitのポーリングログ」を押して確認します。



変更が無ければ、下記のように「No changes」と表示されます。

Gitのポーリングログ

```
Started on 2020/11/09 14:11:00
Using strategy: Default
[poll] Last Built Revision: Revision e6304b152277e1420490166bd8d910fc9ca96455 (refs/remotes/origin/master)
The recommended git tool is: NONE
No credentials specified
> git.exe --version # timeout=10
> git --version # 'git version 2.18.0.windows.1'
> git.exe ls-remote -h -- git@git-practice.tracpath.com
Found 3 remote heads on git@git-practice.tracpath.com
[poll] Latest remote head revision on refs/heads/master is: e6304b152277e1420490166bd8d910fc9ca96455 - already built by E
Done. Took 1.1 秒
No changes
```

レッスン5. ビルドをジョブ化するステップ

今回はバッチファイルの実行方法を例にとってみました。実際の現場ではプロジェクトの環境に応じたコンパイラ等を動かしてファームウェアをビルドすることになるかと思います。

その場合は、下記のステップでジョブ化を検討してみてください。

1. コンパイラのコマンドライン実行仕様を確認する（ヘルプファイル等）
2. 手動でコマンドプロンプトからビルドを試す
3. バッチファイルを作成して、ビルドを試す
4. バッチファイルをJenkinsのクライアントPCに置く



5. ジョブの設定画面で、バッチファイルを起動するコマンドを入れる
→起動オプション等はバッチファイルに埋め込まず、Jenkinsの設定側に入力した方が変更が楽です
6. 手動実行して正しく実行することを確認
7. ポーリングスケジュールを設定し、運用開始

コマンドプロンプトから実行できるコンパイラであれば、どのような言語でも上記方法でジョブ化が可能です。幅広く応用が利きますので、是非お試しください。

おわりに

今回はJenkinsとGitを組み合わせる方法を使ってみました。バージョン管理ツールと組み合わせることで使うことがJenkinsの神髄と言えます。定期的なビルドを正確に行えるので、ファームウェアリリースにかけていた時間をほかの作業に充てることができます。また、今回学んだことを応用すれば、ファームの作成だけでなく、定期的にビルド作業を行って、ビルドエラーが無いかをチェックするといった使い方もできます。プロジェクトの運用方法に沿って、有効活用していきましょう。

最後までお読みいただき、ありがとうございました。

この記事シェアする:



関連

[DevOpsにはCIツールが必要不可欠！Jenkinsの導入を考えてみよう](#)
2016-07-28
Development

[【Ruby版】CIツール導入ガイド 第5回 Linux 環境に Jenkins を導入し CI サーバを構築](#)
2018-12-05
DevOps

[Jenkinsの評価](#)
2016-05-20
Development


CI/CD GIT JENKINS

← Jenkins Pipeline : スクリプト言語
「Groovy」によるデバッグ手法


PlantUMLの基礎 →




You Might Also Like




Python実践演習 〜ライブラリの使い方〜



機械学習、ディープラーニングを始めたい方、Pythonの基礎



PlantUMLの実践応用（シーケンス図）



PlantUMLの基礎

No Comments

コメントを残す

コメントを入力してください。

このサイトはスパムを低減するために Akismet を使っています。 [コメントデータの処理方法の詳細はこちらをご覧ください。](#)

Git/Mercurial/Subversionのクラウド型ホスティングサービス

[詳細を見る](#)

© 2019 TRACPATH.COM ALL RIGHTS RESERVED.

