



10時より開始します

*Cisco Community Expert Series Community Live*

【Top Out Human Capital】

ネットワークエンジニア向け「Git の使い方」

飯山 克志 (Katsushi Iiyama)  
Top Out Human Capital, Inc.  
October 26th, 2022



# ご参加ありがとうございます



## Download the Presentation!

本日の資料はこちらからダウンロードいただけます

<https://community.cisco.com/t5/-/-/ec-p/4683277>

今すぐ登録

資料のダウンロード

# 音声ブロードキャストについて

[音声ブロードキャスト (Audio Broadcast) ] ウィンドウが自動的に表示され、コンピュータのスピーカーから音声がかかります。

[音声ブロードキャスト (Audio Broadcast) ] ウィンドウが表示されない場合は、[ 通話(Communicate) ] メニューから [音声ブロードキャスト (Audio Broadcast) ] を選択します。

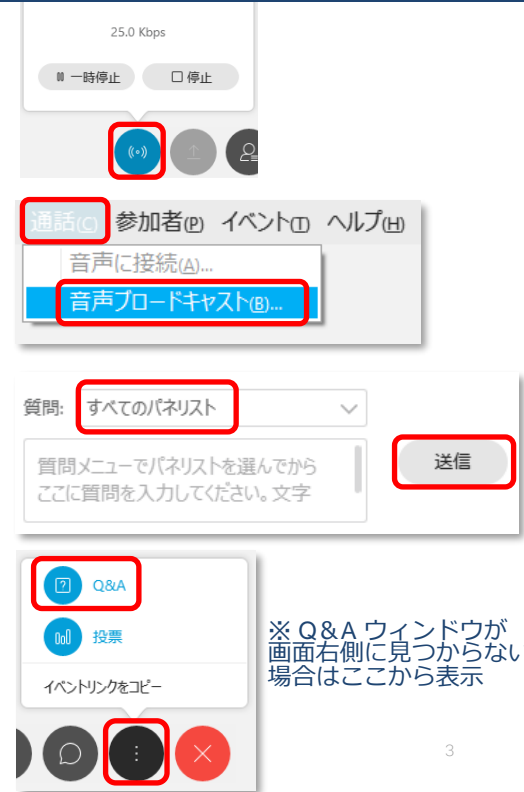
イベントが開始されると自動的に音声がかかります。

音声接続に関する詳細はこちらをご参照ください。

解決しない場合は、Q&A ウィンドウより

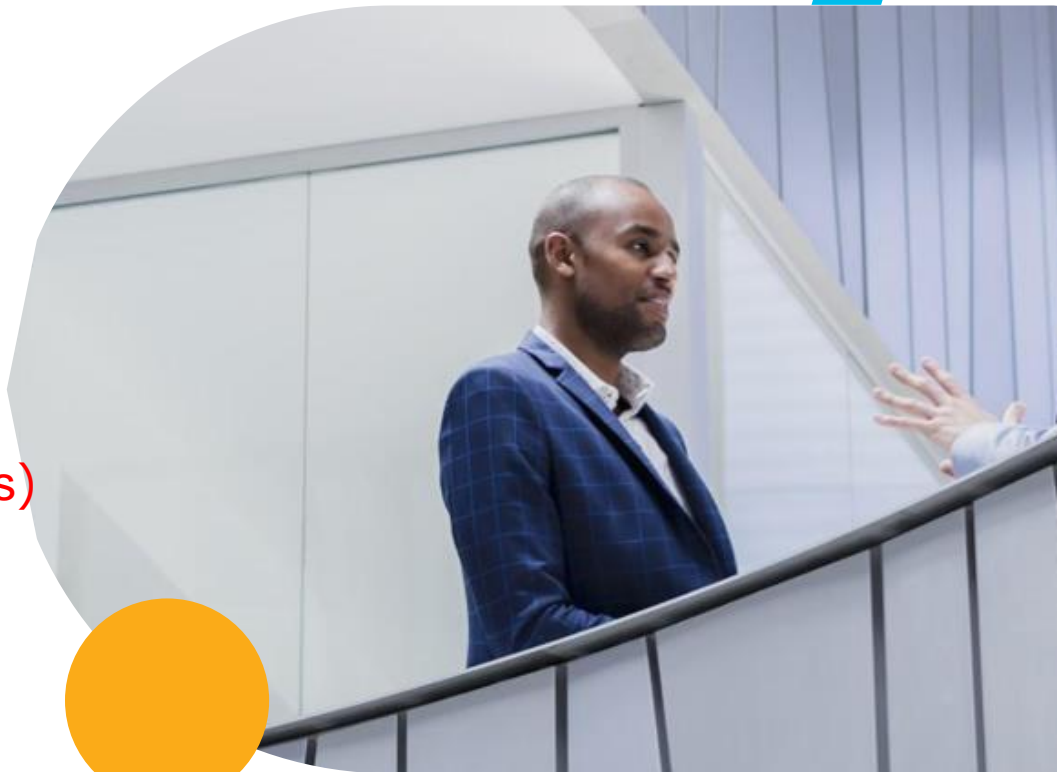
[すべてのパネリスト (All Panelists)] 宛にお知らせください。

<https://community.cisco.com/t5/-/-/ta-p/3129991>



# ご質問方法

Community Live中のご質問は、  
画面右側の Q&A ウィンドウから  
**すべてのパネリスト (All Panelists)**  
宛に送信してください



# 本日のエキスパートご紹介



Cisco Insider  
Champion



Download the Presentation!

本日の資料をダウンロードしてお使いください

<https://community.cisco.com/t5/-/-/ec-p/4683277>



飯山 克志 (Katsushi Iiyama)

Top Out Human Capital 株式会社 CTO

# 投票質問 1

Gitを使ったことがありますか？（単一選択）

- A. 普段から使っている
- B. 使ったことがある
- C. 使ったことがない

# 投票質問 2

Gitを使ったことがある方にご質問です。Gitはどのようにして使っていますか？（単一選択）

- A. 日々の業務において、チームで使用
- B. 個人で使用
- C. 両方で使用

# 投票質問 3

セミナーの中でPythonの話が出てきます。  
Pythonについてどれくらいご存知ですか？（単一選択）

- A. 日々の業務に使っている
- B. sampleを見ながら組むことができる
- C. プログラムを組んだことがない





*Cisco Community Expert Series Community Live*

【Top Out Human Capital】

# ネットワークエンジニア向け「Git の使い方」

飯山 克志 (Katsushi Iiyama)  
Top Out Human Capital, Inc.  
October 26th, 2022



# 本日の目次

- ・ Gitの簡単な紹介
  - ・ Gitでできること
- ・ Gitの仕組み
- ・ Git使用方法
  - ・ コマンド紹介
- ・ デモ1
  - ・ Gitコマンドを使っでの操作
- ・ デモ2
  - ・ Pythonを使っでの操作



# Top Out Human Capital 株式会社

## 会社紹介とご案内



# 日本で唯一がここにある!



- Cisco 認定ラーニングパートナー
- DevOps Institute認定教育パートナー
- Pythonエンジニア育成推進協会認定スクール
- NetApp 認定ラーニングパートナー
- EC-Council 認定トレーニングセンター
- CompTIAトレーニングパートナー
- Citrix認定ラーニングセンター
- HPE Aruba認定トレーニングセンター
- Acronis 認定教育パートナー
- Huawei認定ラーニングパートナー
- F5 認定トレーニングセンター
- IoT関連トレーニング
- Alibaba Cloud ハンズオントレーニング
- PMI認定トレーニングパートナー
- Gigamon 認定トレーニング
- RedHat 認定トレーニング
- ビジネストレーニング (ITILなど)

- (日本で唯一、Collaboration, Security関連コースを実施)
- (日本総代理店、DevOps・SREエンジニアの育成コースを実施)
- (Pythonによる運用&監視の自動化コースを提供)
- (日本で唯一の認定ラーニングパートナー)
- (Cybersecurity、全てのEC-Council認定コースを唯一実施)
- (日本で唯一、Cybersecurity資格に対応する全コースを実施)
- (日本で唯一、Citrixコースすべてを実施)
- (日本で唯一、Wireless LAN、Switch、Clear Pass)
- (日本で唯一、Acronis #CyberFitトレーニングを実施)
- (日本で唯一、5G、Wireless LAN 他)
- (BIG-IPシリーズ)
- (IoTハッキング・セキュリティ、IoT実践ハンズオン)
- (1日で学べるハンズオン)



AUTHORIZED TRAINING CENTER



AUTHORIZED TRAINING CENTER (-) Alibaba Cloud

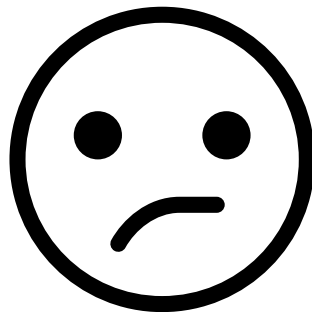


始める前に



# こんな経験ありませんか？

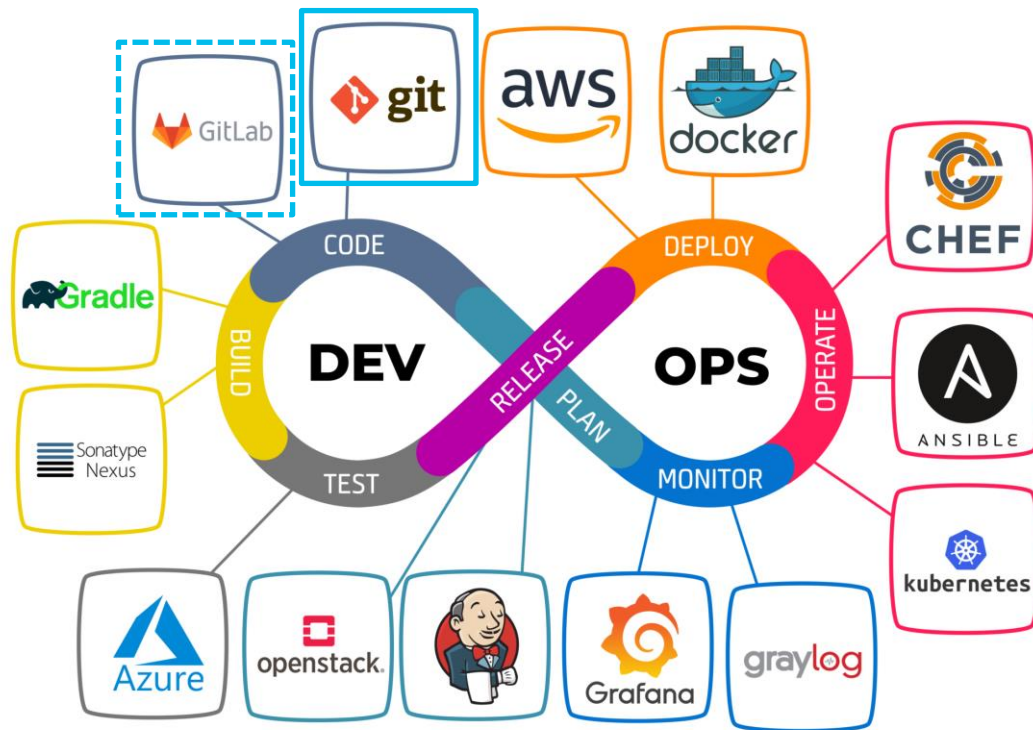
- ・ 設定ファイルを上書きしてしまった
- ・ 先週は動いていたのに、今日試したら動かなかった
- ・ 過去の構成と比較をするが、変更点を探すのに非常に時間がかかる
- ・ DXとかDevOpsについて調べろと言われてるけど・・・



今はどうしているの？

# DevOps

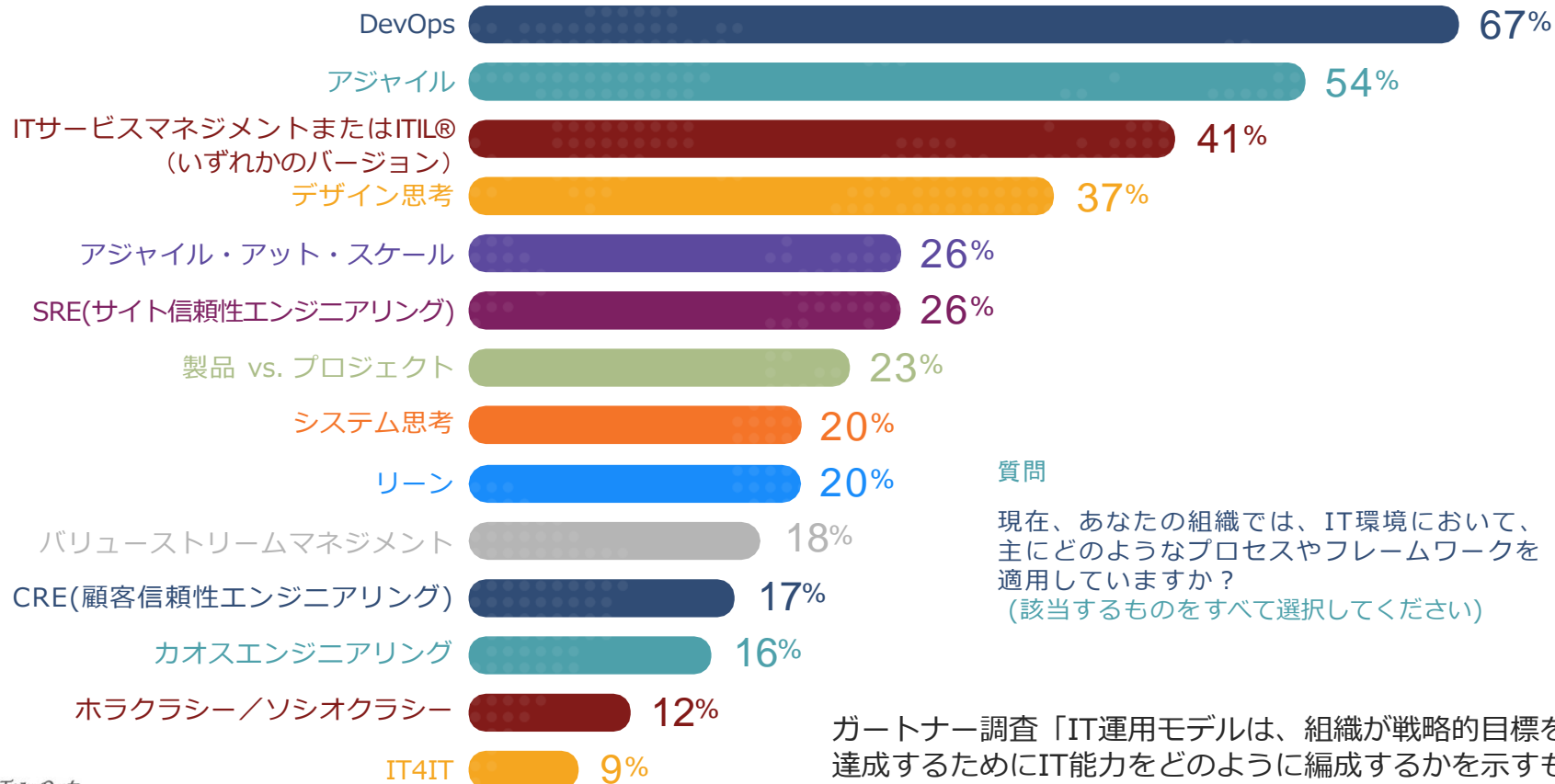
- 開発(Dev) と運用(Ops) を組み合わせた合成語
- 開発担当者と運用担当者が連携して協力する開発手法
- ソフトウェアを迅速にビルドおよびテストする文化と環境により、確実なリリースを、以前よりも迅速に高い頻度で可能とする組織体制の構築を目指している。



→インフラの設計を「開発(Dev)」と捉えてみませんか？  
日頃の「運用(Ops)」で使えるものがたくさんあります！



# APACにおけるプロセスやフレームワークを活用したIT運用モデル



## 質問

現在、あなたの組織では、IT環境において、主にどのようなプロセスやフレームワークを適用していますか？

(該当するものをすべて選択してください)

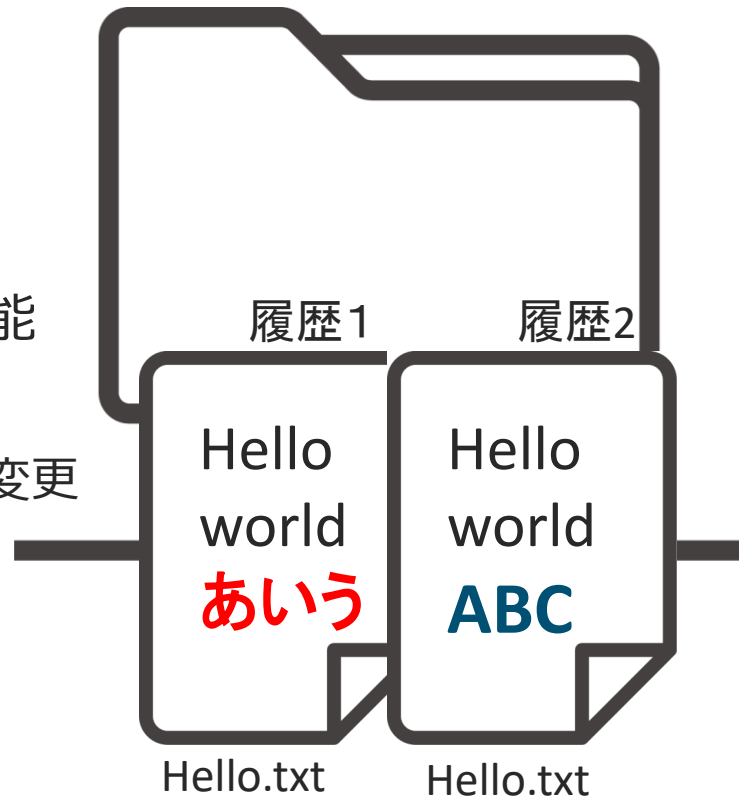
ガートナー調査「IT運用モデルは、組織が戦略的目標を達成するためにIT能力をどのように編成するかを示すもの」

# *Git*の簡単な紹介



# Gitでできること

- 分散型バージョン管理システム
- ソースコードなどの変更履歴の保存が可能
- 履歴情報
  - いつ、誰が、どのファイルを、どのように変更
  - 履歴にメッセージを残すことができる
- 編集したファイルを過去の状態に戻せる
- 編集箇所の差分が確認できる



# Gitでできること

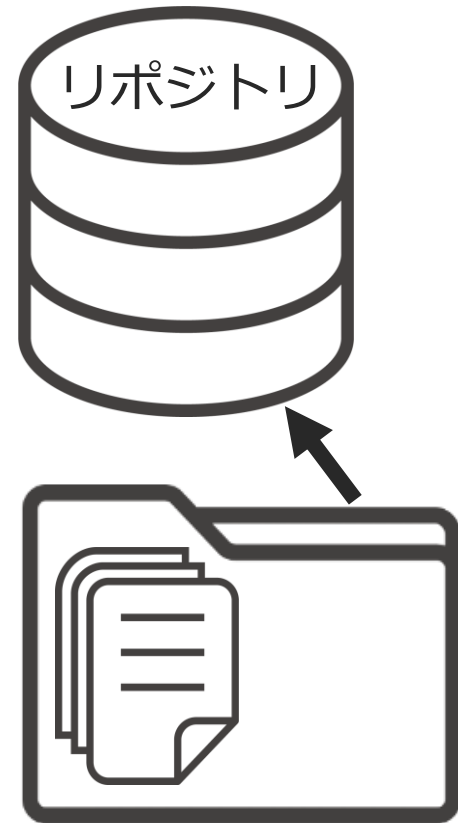
- コンフリクト検知
  - 複数人での修正を1つに統合できる
- 管理できるデータ
  - **設定ファイル、パラメータシート、Visio、図面**、プログラム、テキストデータ、画像データ、Excelデータ、etc
- 共有環境（サーバ）と作業環境（ローカル環境）が分かれているが同じ環境を持つことができる
  - リモートリポジトリ
  - ローカルリポジトリ

# *Git*の仕組み

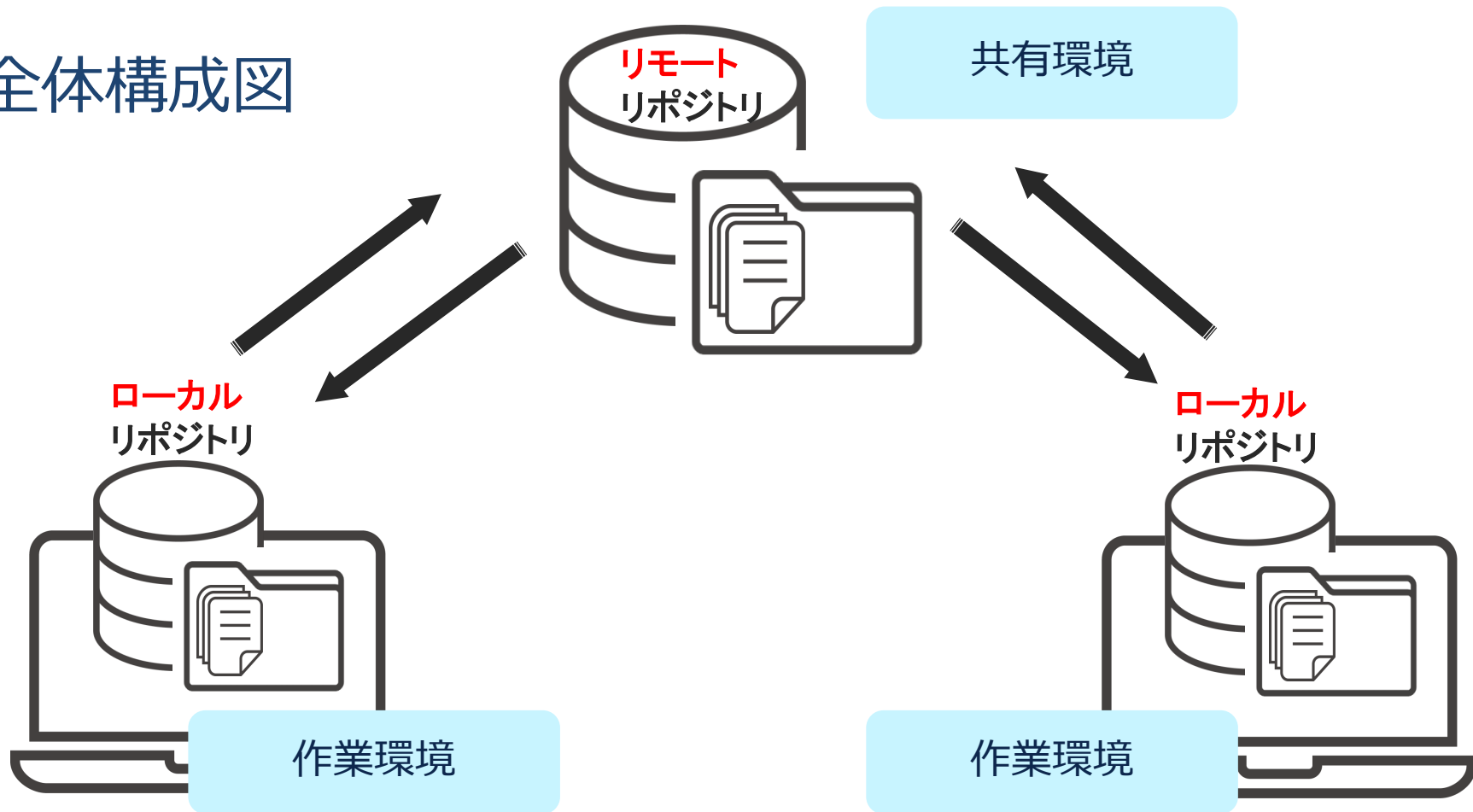


# リポジトリ (repository)

- リポジトリ (repository)
  - ファイルの変更履歴が入っている場所
- **リモート**リポジトリ
  - 複数人で共有するためのリポジトリ
- **ローカル**リポジトリ
  - ユーザごと手元のマシンで編集できるリポジトリ

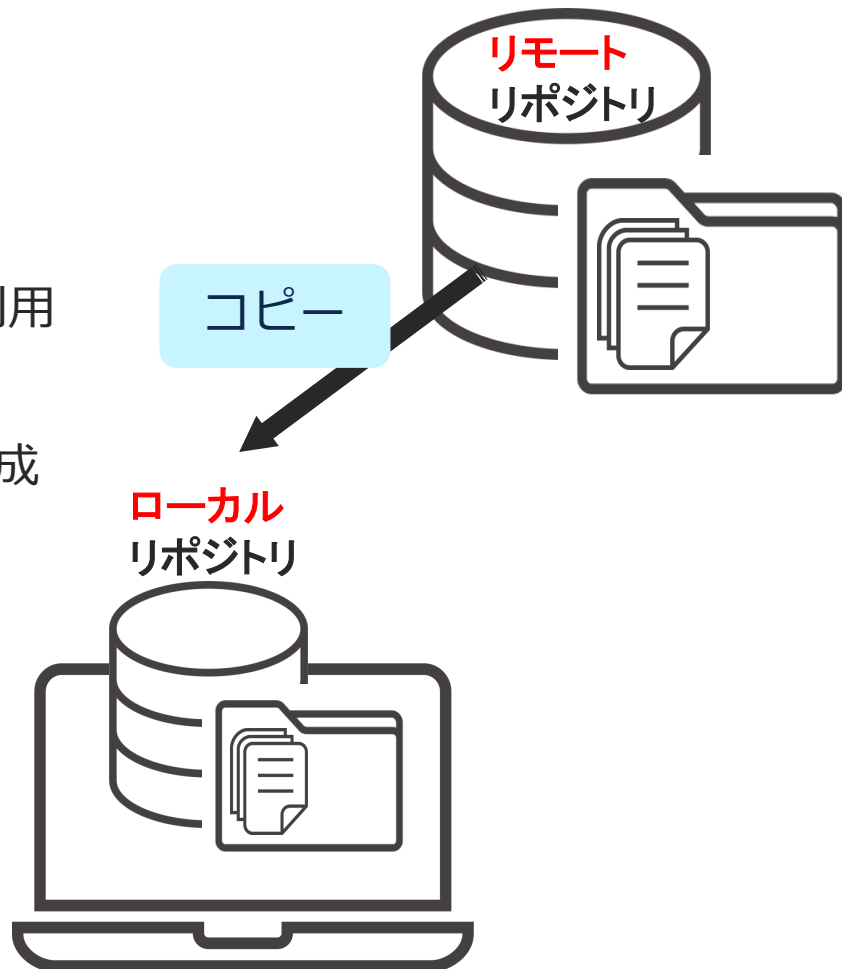


# 全体構成図



# リポジトリの用意

- ・ リモートリポジトリ
  - ・ GitHub、GitLabなどのWebサービスを利用
- ・ ローカルリポジトリ
  - ・ ローカルPC内に新しくリポジトリを作成
  - ・ リモートリポジトリをコピーしてくる





# コミット

- コミット
  - ファイルやディレクトリの追加・変更履歴をローカルリポジトリに記録すること
- コミット時に作成される**変更履歴情報**
  - リビジョン番号
  - いつ
  - 誰が
  - どのファイル
  - どんな内容
  - メッセージの追加

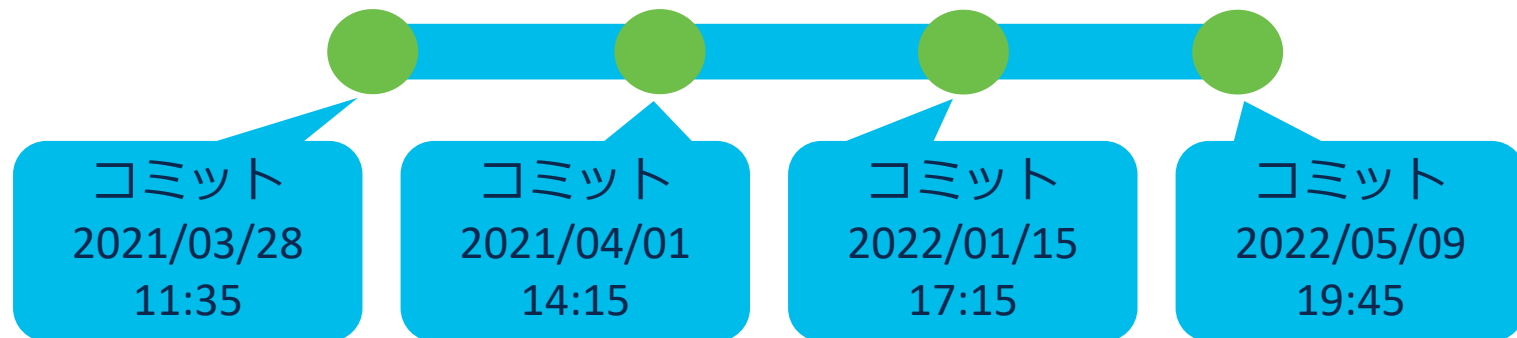
The screenshot shows a Git commit page for a commit with hash `fe78cdf8` authored by `NWAdmin` one week ago. The commit message is `Change Tokyo router config`. The page shows a diff for the file `router_Tokyo`, which has 5 additions and 2 deletions. The diff content includes configuration details for a router interface, such as `interface Loopback22` and `ip address 22.22.22.22 255.255.255.255`.

Annotations in Japanese:

- いつ・誰が**: Points to the commit hash and author information.
- 追加メッセージ**: Points to the commit message.
- リビジョン番号**: Points to the commit hash.
- 対象ファイル名**: Points to the filename `router_Tokyo`.
- 追加内容**: Points to the diff content showing the added interface configuration.
- コミット情報**: Points to the commit hash and author information.

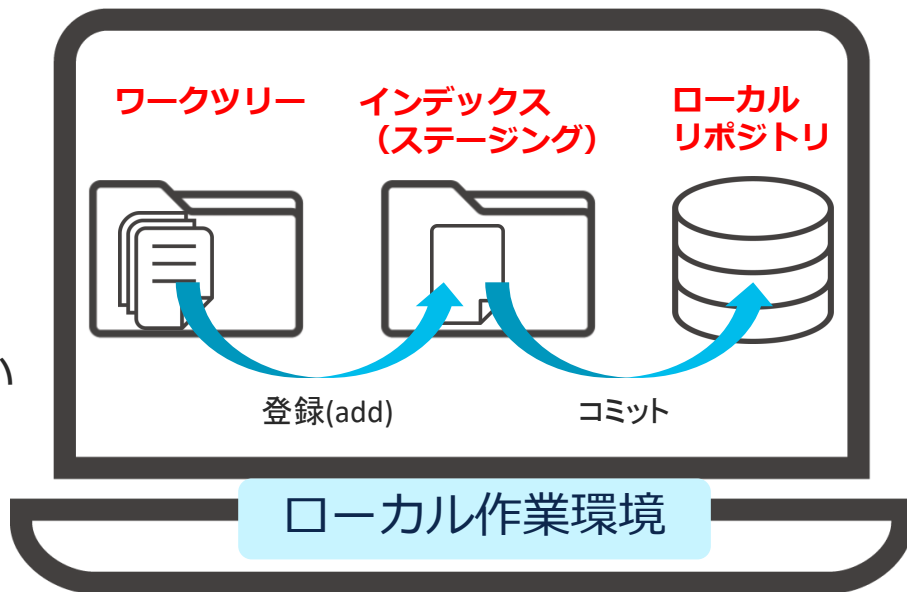
# コミット

- ・ 時系列順にリポジトリに保存
- ・ 履歴を辿ることが可能
- ・ 過去の状態に戻すことが可能



# ワークツリーとインデックス (ステージング)

- ワークツリー
  - Git管理下に置かれたユーザーが作業を行なっているフォルダ
- インデックス
  - ステージングとも呼ばれる
  - リポジトリにコミットする前の準備をするための場所
  - インデックスに登録されていないファイルはコミットされない



# ファイルの流れ確認

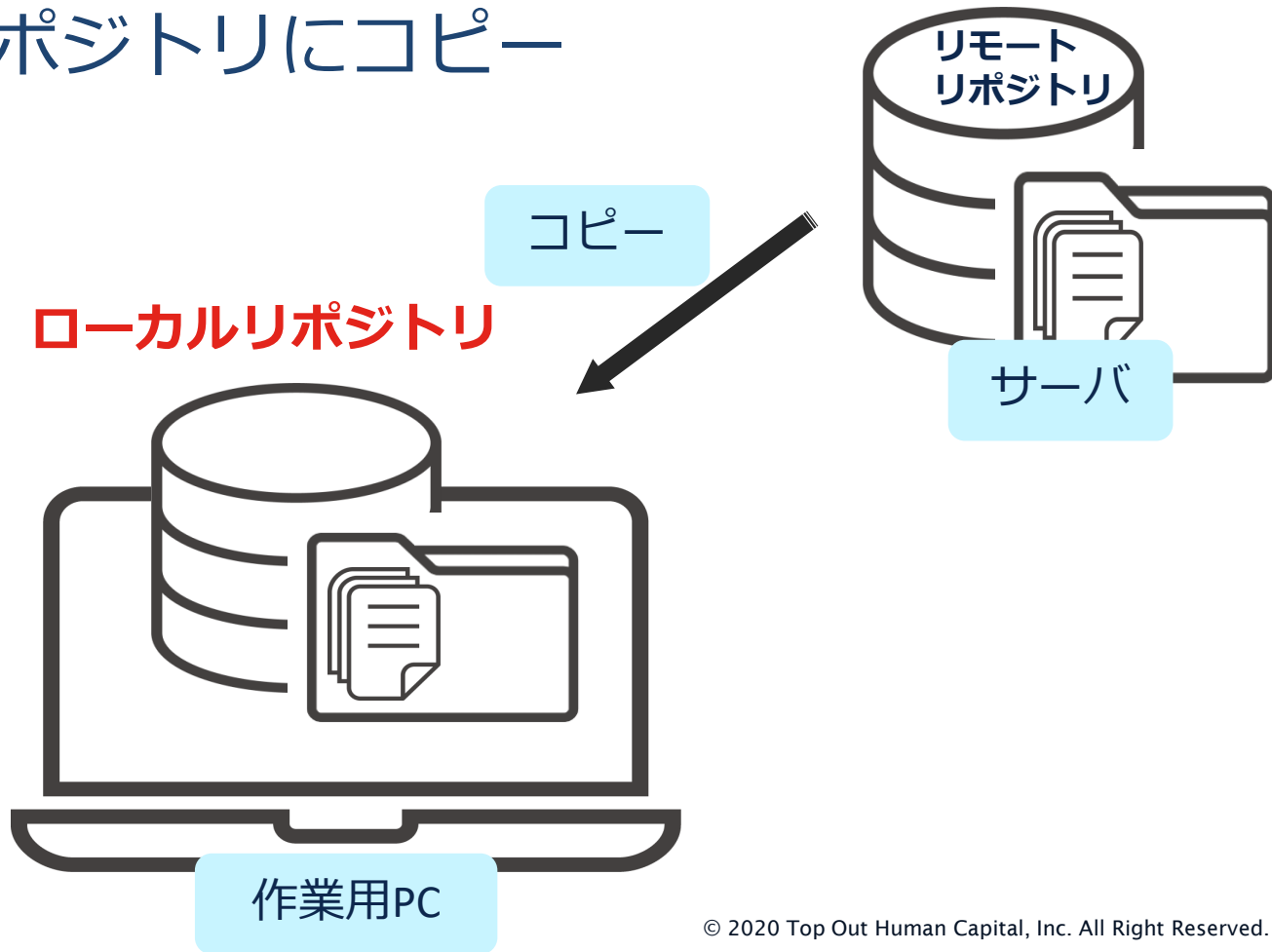


# リモートリポジトリの用意

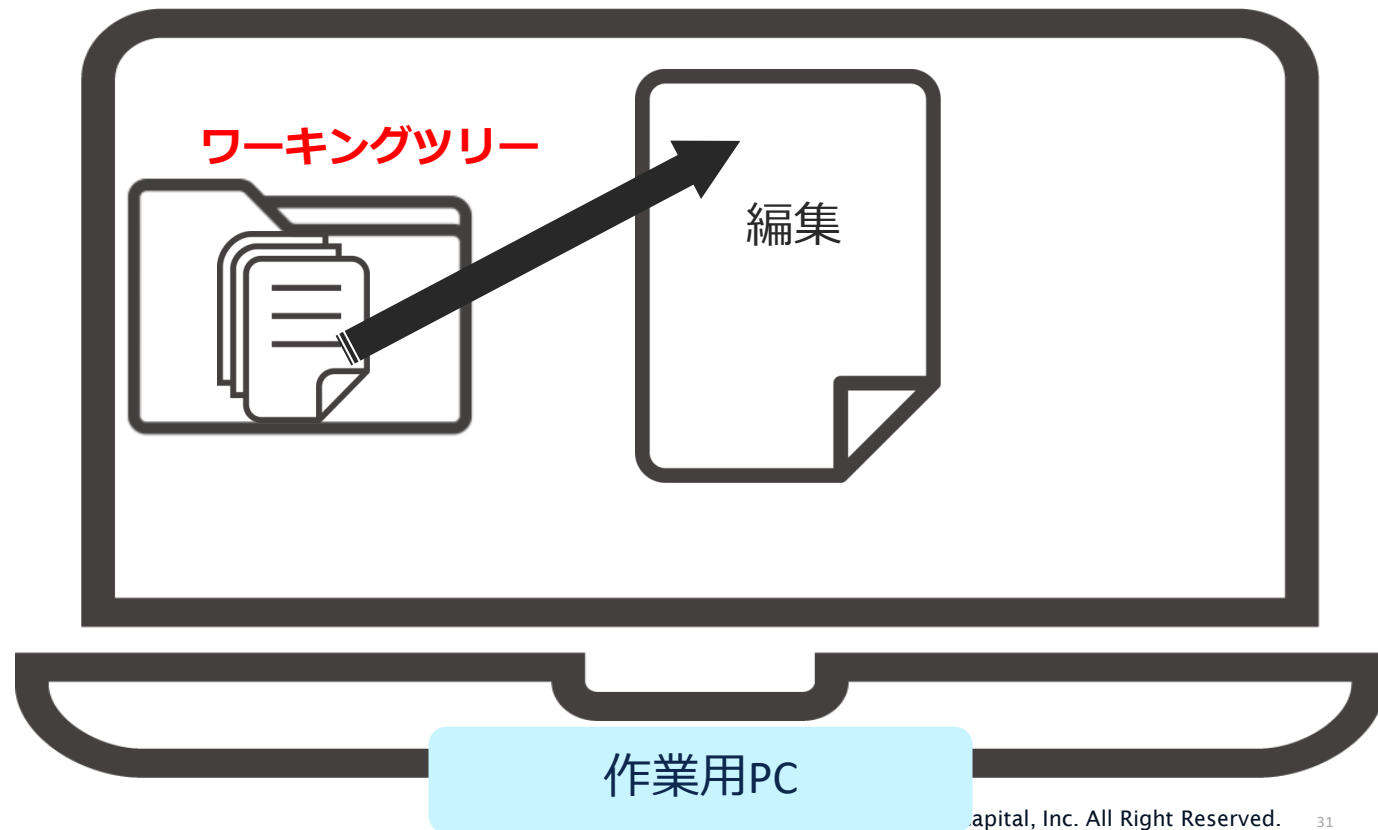
- Git webサービスを利用
- GitHub
- GitLab
- Bitbucket
- など



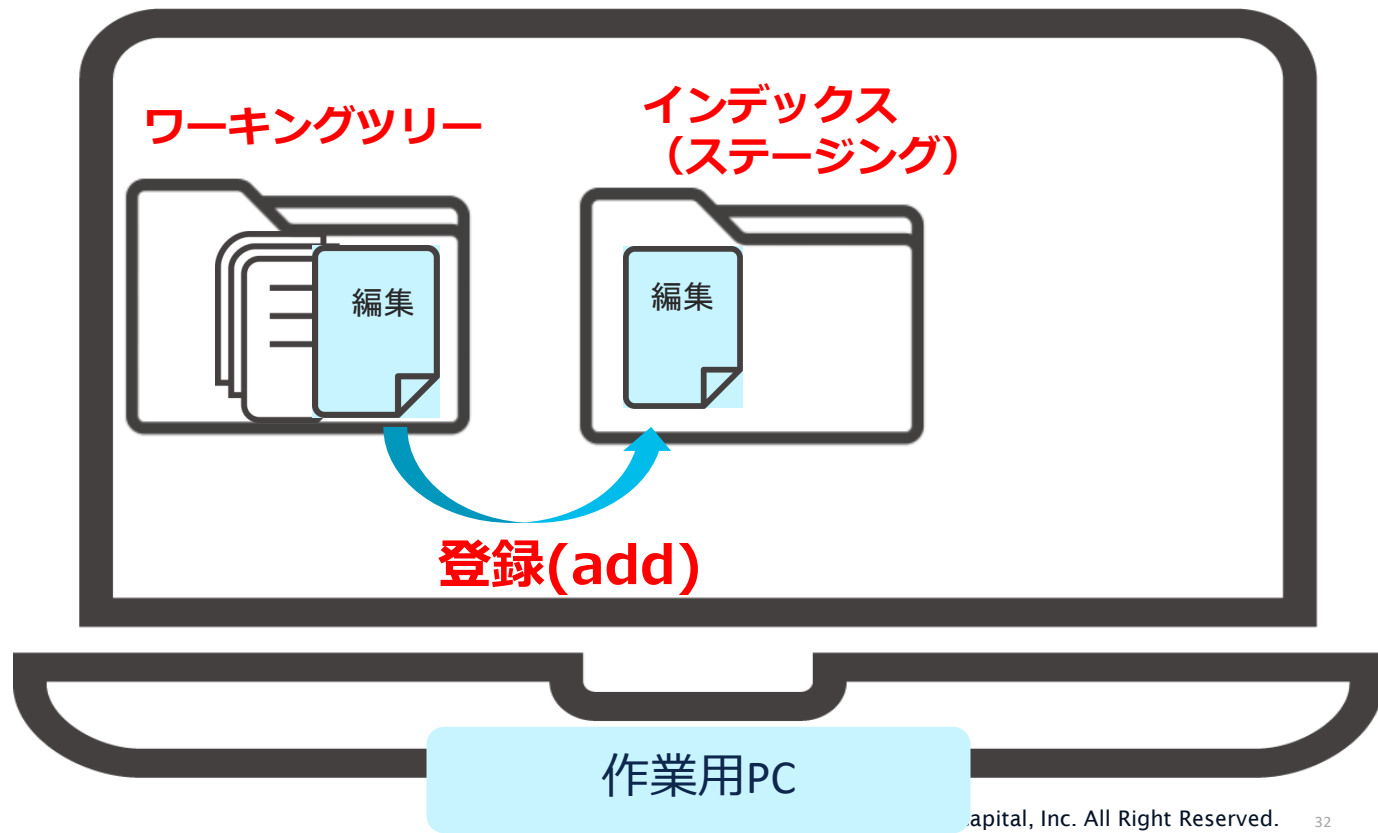
# ローカルリポジトリにコピー



# ファイルの編集

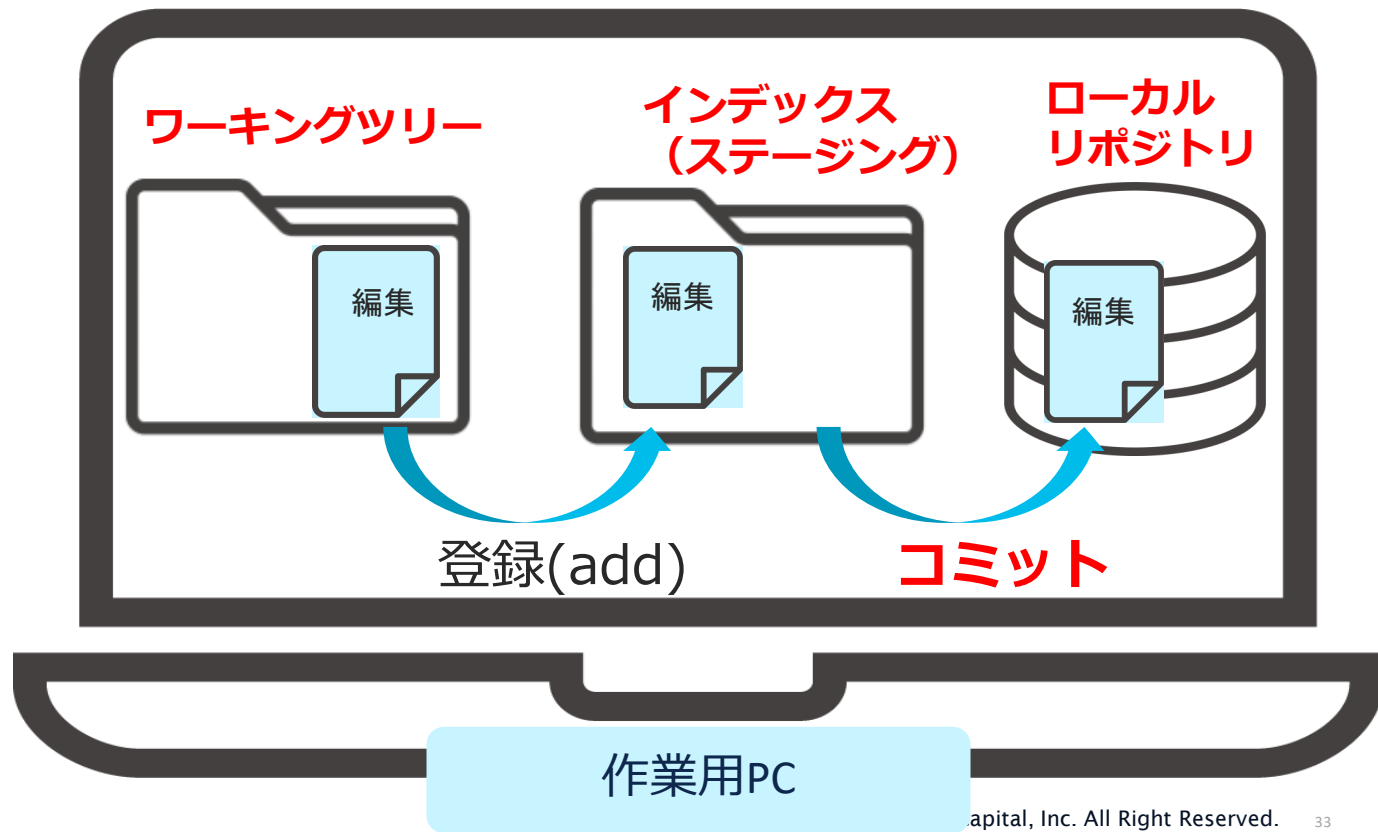


# インデックスに登録

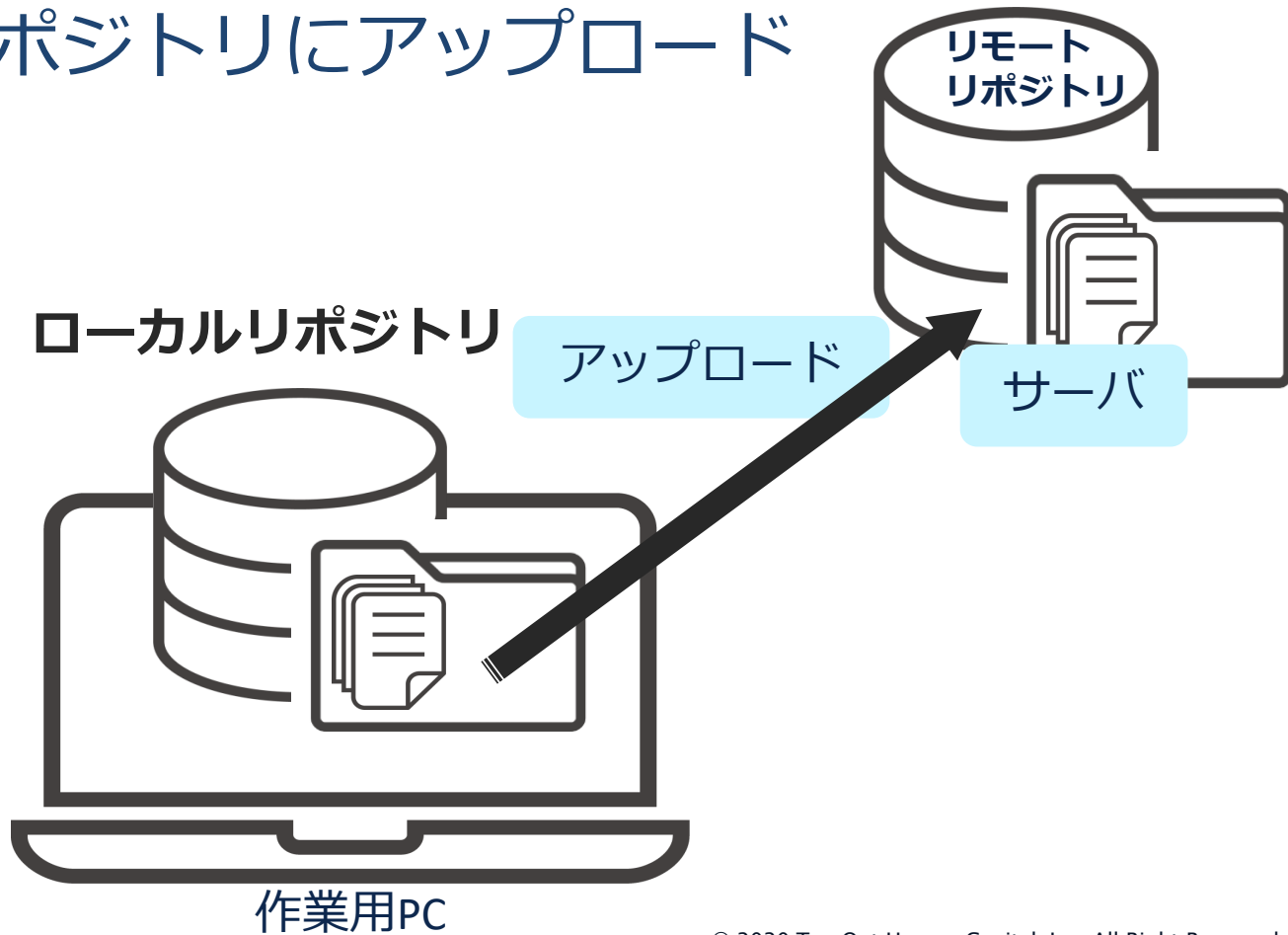




# インデックスに登録



# リモートリポジトリにアップロード



# *Git*の使用方法 コマンドの紹介



# Gitのインストール

- Linux
  - コマンド (`$ sudo yum install git-all`、`$ sudo apt-get install git-all`など)
  - 他Linuxディストリビューション用インストール手順  
<http://git-scm.com/download/linux>
- Mac OS
  - Mavericks (10.9)以降ならば、Gitをターミナルから実行可能
  - Mac用インストールサイト <http://git-scm.com/download/mac>
- Windows
  - Git for Windowsプロジェクト (Gitとは違うプロジェクト)
  - Windows用インストールサイト <http://git-scm.com/download/win>

# Gitの設定

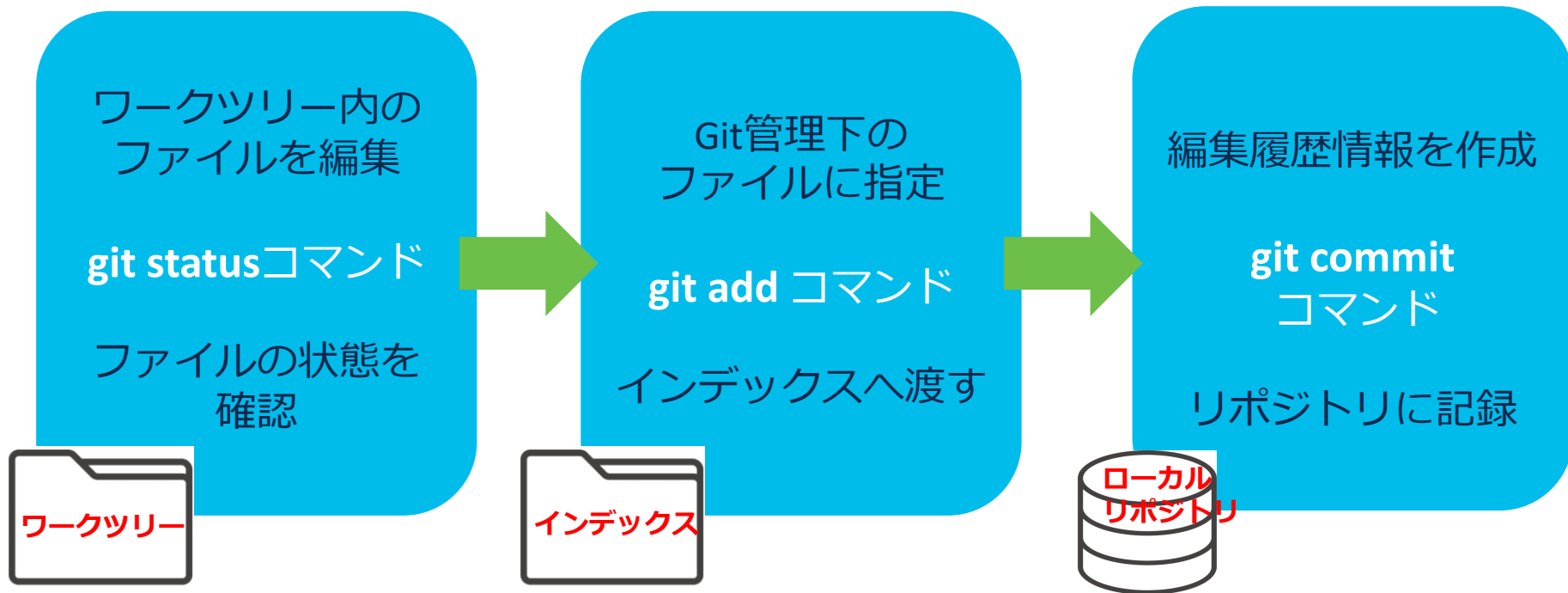
- ユーザ設定
  - コミット時の変更履歴に使用される（ユーザ名など）
- git configコマンドを使用
  - ユーザ名登録：**git config --global user.name "NWadmin"**
  - メール登録：**git config --global user.mail "nwadmin@dev.local"**
    - --globalオプションをつけると、該当ユーザの全リポジトリの設定
- 特定のユーザに対する設定ファイル
  - ~/.gitconfig または ~/.config/git/config

```
cat ~/.gitconfig
[user]
[user]
        name = Nwadmin
        email = nwadmin@dev.local
```

# Git リポジトリの取得

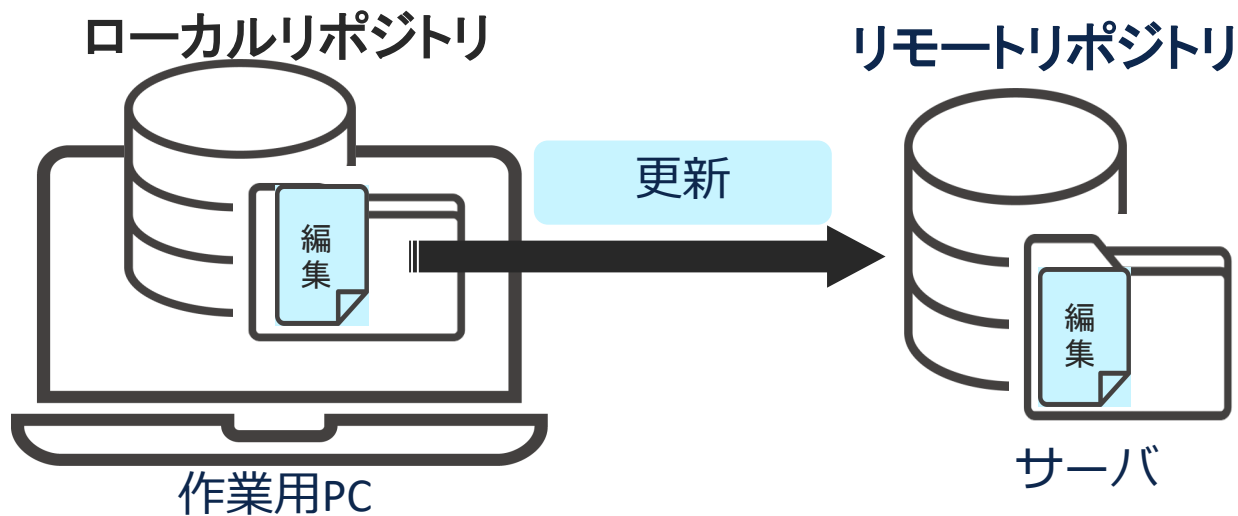
- 既存のプロジェクト（ディレクトリ）をGit にインポートする方法
  1. 既存のプロジェクトがあるディレクトリに移動
  2. `$ git init` コマンドを使用
  3. Git管理対象にしたいファイルをコミット処理する
  
- 既存のGitリポジトリを別のサーバからクローンする方法
  1. GitライブラリのURLを確認する（またはSSHの接続先）
  2. `$ git clone [URL]` コマンドを使用

# ファイルをコミットする



# リモートリポジトリへプッシュする

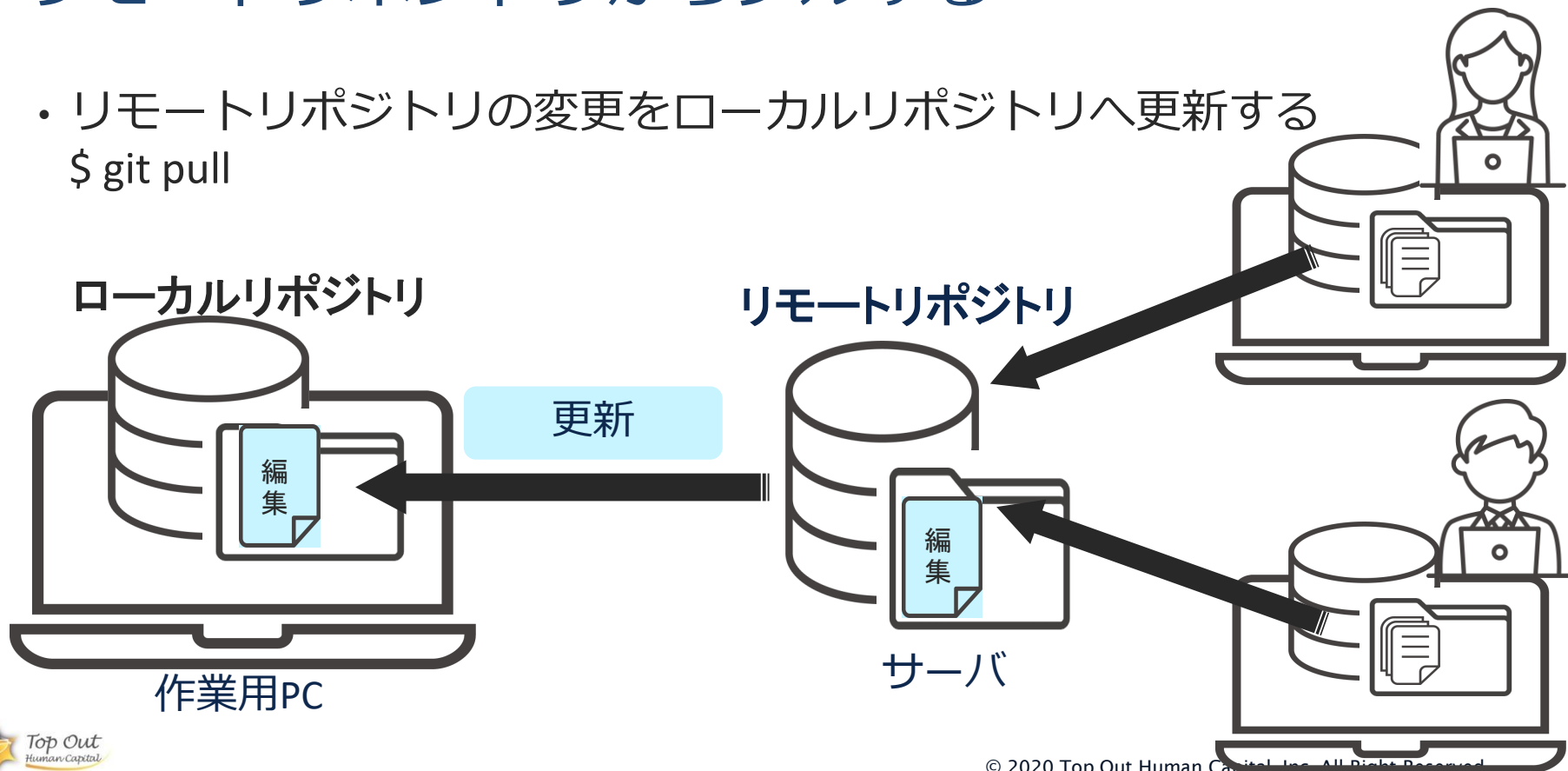
- ローカルリポジトリとリモートリポジトリを同じ状態にする  
\$ git push



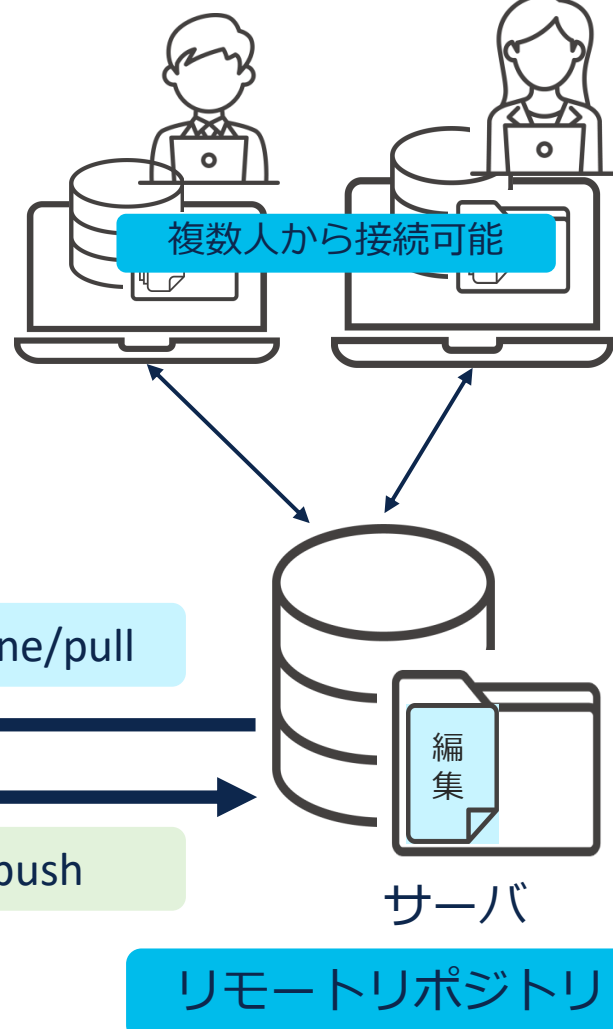
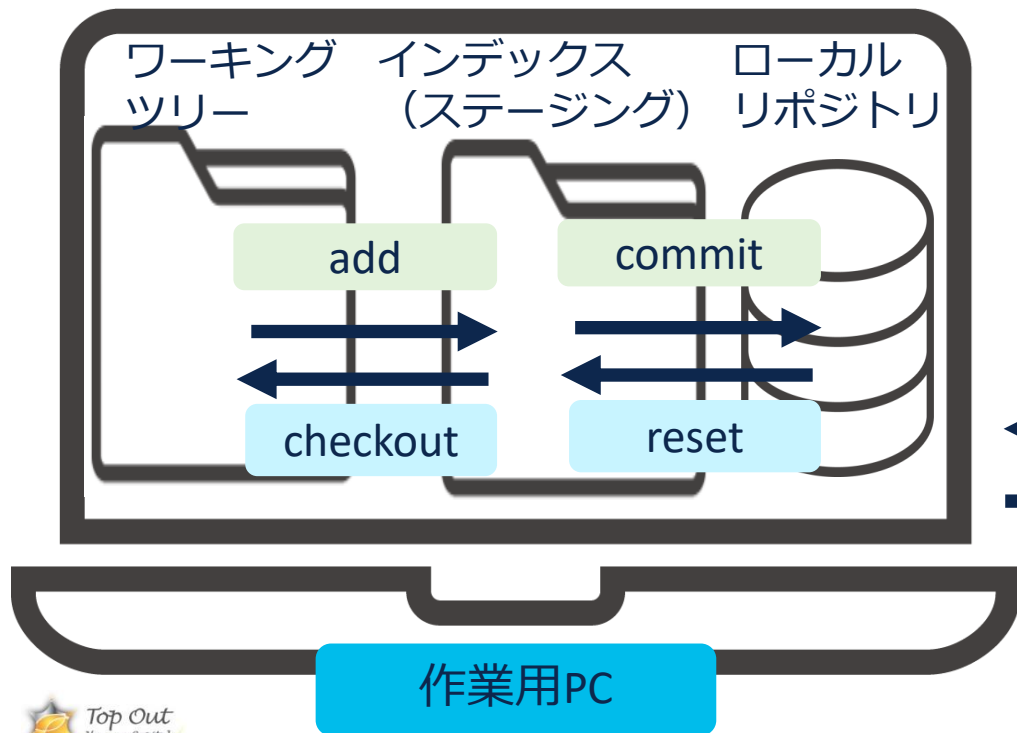


# リモートリポジトリからプルする

- ・ リモートリポジトリの変更をローカルリポジトリへ更新する  
\$ git pull



# Gitコマンドのまとめ



デモ 1

*Git* コマンドを使っての操作

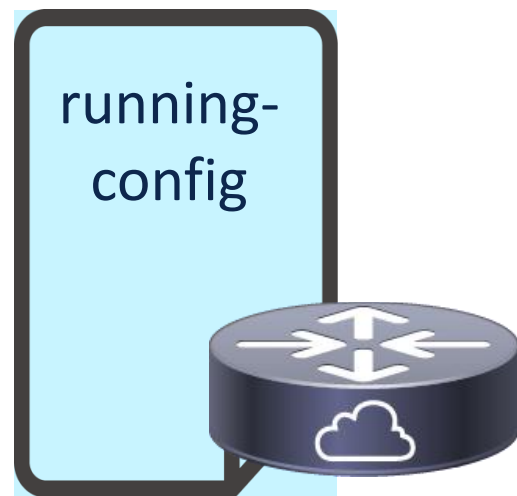


# ルータの設定ファイルの管理

- ルータの設定ファイルはどのように管理していますか？
  - TFTPサーバをたててバックアップしている
  - ルータのflashを活用している
  - 部署で用意されているファイルサーバへ保存
  - startup-configのみに保存 . . .



- Version管理ができない
- 差分がわからず、トラブルシューティングに時間が . . .



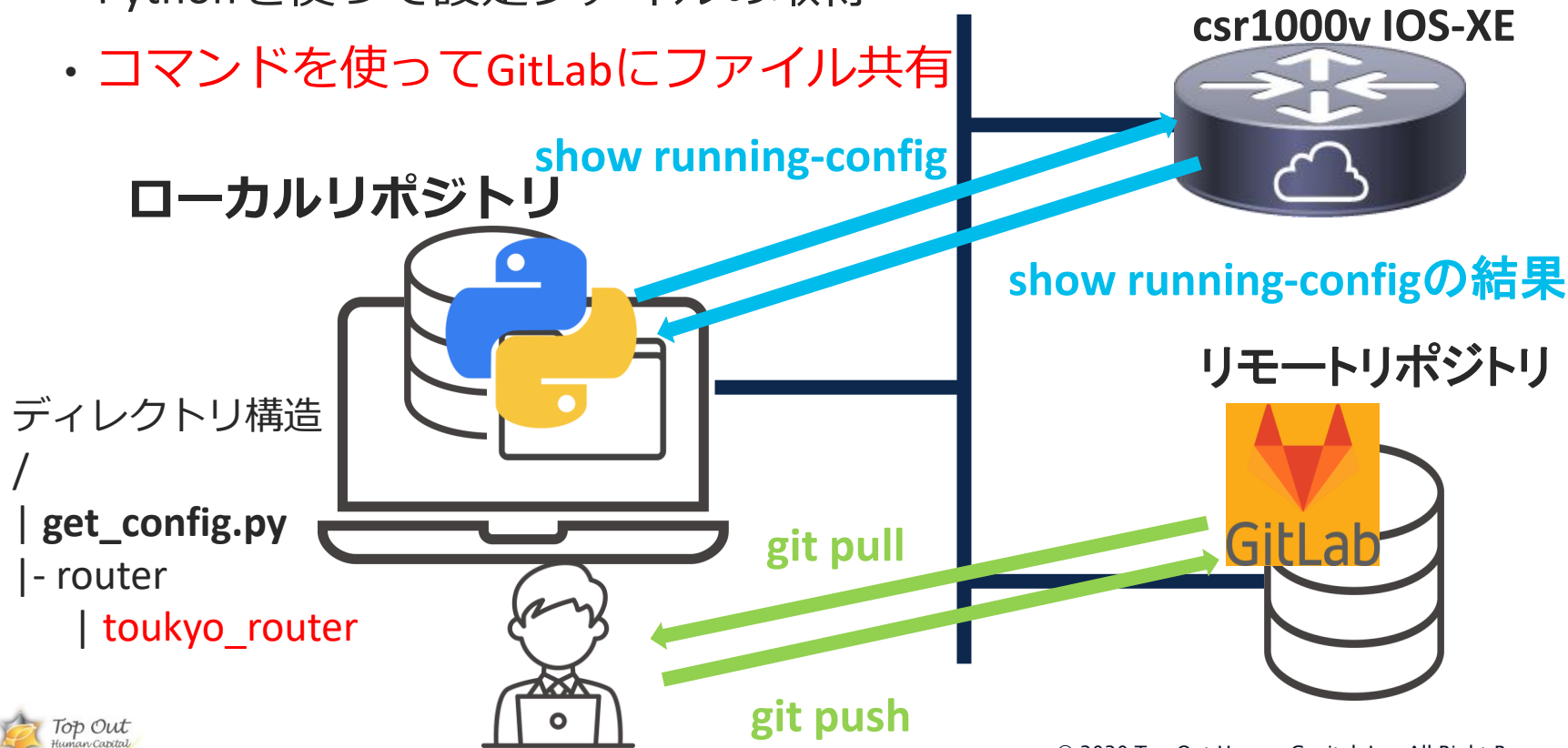
# ルータの設定ファイルの管理

- Gitを使って管理してみましよう！！
  - 変更履歴の管理ができます
  - 差分の確認がしやすい
  - 設定ファイルを共有して確認ができます



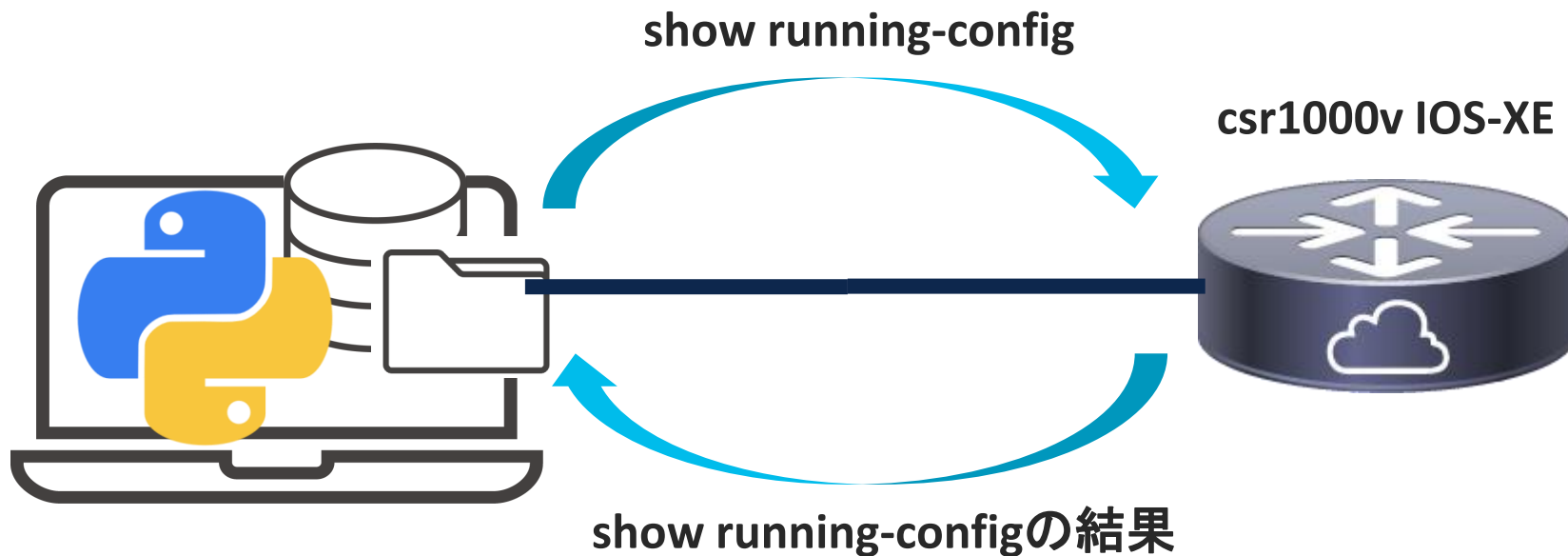
# GitLabを置き、設定ファイルを管理

- Pythonを使って設定ファイルの取得
- コマンドを使ってGitLabにファイル共有



# 1. ルータから設定情報を収集(1)

- Pythonのライブラリ（Netmiko）を使用



# Netmikoとは(1)

- Paramikoをネットワーク機器(Cisco IOSなど)に対応させたライブラリ
- コマンドを通じて**情報の取得**や**設定の変更**を行うことができる
- 長所 : 学習コストが低いため、簡単な自動化には便利
- 短所 : 出力をパースする必要があり、複雑なシステムでは開発効率が低い



## Netmikoとは(2)

- 設定確認メソッド: `send_command`(コマンドの文字列)

```
import netmiko

#netmikoを使ってルータに接続し、コマンドを発行する
session = netmiko.ConnectHandler(device_type='cisco_ios', host='192.168.0.1',
                                  username='admin', password='python123', secret='python123')
session.enable()
output = session.send_command('show running-config')

#取得してきたrunning-configの内容をファイルに書き込む
#明記されているディレクトリに「router_Tokyo」というファイル名で保存される
with open('/home/topout/router/router_Tokyo', mode='w') as f:
    f.write(output)
```

# ネットワーク機器向けのライブラリ比較

	Expect Pexpect	Paramiko	Netmiko	Napalm	Requests
コンソール	✓				
ssh	✓	✓			
telnet	✓				
ネットワーク機器			✓	✓	
Webサーバ					✓
REST API					✓

# ルータから設定情報を収集

1. pythonを実行して、ルータの設定ファイル(running-config)を取得  
**\$ python3 get\_config.py**
2. routerディレクトリに「router\_Tokyo」というファイルができてい  
るか確認  
**\$ ls router/router\_Tokyo**
3. ファイルの中身を確認  
show running-configの結果が取得できている  
**\$ cat router/router\_Tokyo**

# ルータの設定ファイルを共有(1)

1. `git status`コマンドを使って、ワークツリー内のファイルの状態を確認する（`router_Tokyo`ファイルがGit管理下にまだ入っていない）

```
$ cd router
$ git status
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    router_Tokyo

Nothing added to commit but untracked files present (use "git add" to track)
```

## ルータの設定ファイルを共有(2)

2. ルータの設定ファイルが入っている「router\_Tokyo」ファイルを、Git管理下のファイルに指定する（インデックスに追加）

```
$ git add router_Tokyo
```

3. 編集履歴情報を作成する（リポジトリに記録する）

```
$ git commit -m "Tokyo router config"  
[main 2cfad73] Tokyo router config 2252  
1 file changed, 257 insertions(+)  
create mode 100644 router_Tokyo
```

## ルータの設定ファイルを共有(3)

4. `git status`コマンドを使って、ワークツリー内のファイルの状態を確認

```
$ git status
# On branch main
# Your branch is ahead of 'origin/main' by 1 commit.
#   (use "git push" to publish your local commits)
#
Nothing to commit, working directory clean
```

# ルータの設定ファイルを共有(4)

## 5. リモートリポジトリへプッシュする

```
$ git push
```

```
Username for 'http://gitlab.dev.local': root
```

```
Password for 'http://root@gitlab.dev.local':
```

```
Counting objects: 3, done.
```

```
Delta compression using up to 2 threads.
```

```
Compressing objects: 100% (3/3) done.
```












```
Writing objects: 100% (3/3), 3.52 KiB | 0 bytes/s done.
```

```
Total 3 (delta 1), reused 0 (delta 0)
```

```
To http://gitlab.dev.local/root/router.git
```

```
20f269e..2cfad73 main -> main
```

# リモートリポジトリの確認

Name	Last commit	Last update
 hostname test4	hostname test4	6 days ago
 README.md	Initial commit	1 week ago
 hostname ROUTER	test1	6 days ago
 hostname test1	hostname test1	
 hostname test2	hostname test2	
 hostname test3	hostname test3	
 hostname test4	hostname test4	
 hostname test4show ip int bri	hostname test4show ip int bri	6 days ago
 router	init commit	51 minutes ago
 router_Tokyo	Tokyo router config	11 minutes ago
 show run   include hostname	test1	6 days ago

commit時に登録したメッセージ  
クリックすると変更履歴情報の  
確認ができる






# リモートリポジトリの確認 (Inline表示)

Showing 1 **changed file** with 253 **additions** and 0 **deletions**

Hide whitespace changes

Inline

Side-by-side

▼  **router\_Tokyo**  0 → 100644 +253 -0  View file @9b5e946f

```
1 + Building configuration...
2 +
3 + Current configuration : 6467 bytes
4 + !
5 + ! Last configuration change at 11:57:14 JST Thu Oct 13 2022 by admin
6 + !
7 + version 16.12
8 + service timestamps debug datetime msec
9 + service timestamps log datetime msec
10 + service call-home
11 + platform qfp utilization monitor load 80
12 + platform punt-keepalive disable-kernel-core
13 + platform console virtual
14 + !
15 + hostname Tokyo
16 + !
```

# リモートリポジトリの確認 (Side-by-side表示)

Showing 1 changed file ▾ with 253 additions and 0 deletions

Hide whitespace changes

Inline

Side-by-side

▼ router\_Tokyo 0 → 100644 +253 -0 View file @9b5e946f

```
1 + Building configuration...
2 +
3 + Current configuration : 6467 bytes
4 + !
5 + ! Last configuration change at 11:57:14
   JST Thu Oct 13 2022 by admin
6 + !
7 + version 16.12
8 + service timestamps debug datetime msec
9 + service timestamps log datetime msec
10 + service call-home
11 + platform qfp utilization monitor load 80
12 + platform punt-keepalive disable-kernel-
   core
```

# ルータの設定変更

- ルータの設定を変える
  - ループバックインターフェイス22を追加
  - 追加したインターフェイスにアドレス（22.22.22.22/32）を設定

# コンフィグの差分をGitで管理

```
$ python3 get config.py
$ cd router
$ git add router Tokyo
$ git commit -m "Change Tokyo router config"
[main d56ba56] Change Tokyo router config
 1 file changed, 5 insertions(+), 2 deletions(-)
$ git push
Username for 'http://gitlab.dev.local': root
Password for 'http://root@gitlab.dev.local':
Counting objects: 5, done.
Delta compression using up to 2 threads.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 341 bytes | 0 bytes/s,
done.
Total 3 (delta 2), reused 0 (delta 0)
To http://gialab.dev.local/root/router_config.git
 2cfad73..d56ba56  main -> main
```

1. ルータの設定を変更



2. 再度pythonを使って  
ルータのコンフィグを  
取得



3. 取得した設定ファイルを  
Gitで管理し、リモート  
リポジトリへ共有

# コンフィグの差分をGitで確認 (Inlineで確認)

The screenshot shows a Git commit page for a commit with hash `fe78cdf8` authored by `NWadmin`. The commit message is `Change Tokyo router config`. The diff view shows changes to the file `router_Tokyo`. The diff is displayed in inline mode, showing the current configuration (6530 bytes) and the last configuration change (6467 bytes) at 11:57:14 JST Thu Oct 13 2022 by `admin`. The diff shows the addition of interface configuration for `Loopback22` with IP address `22.22.22.22`.

Annotations in the image:

- いつ・誰が** (When and by whom): Points to the commit hash and author.
- 追加メッセージ** (Additional message): Points to the commit message.
- 対象ファイル名** (Target filename): Points to the file name `router_Tokyo`.
- リビジョン番号** (Revision number): Points to the commit hash `@fe78cdf8`.
- 追加内容** (Additional content): Points to the added lines in the diff.

# コンフィグの差分をGitで確認 (Side-by-side表示)

Commit fe78cdf8 authored 1 week ago by NWAdmin

Change Tokyo router config

parent 9b5e946f main

No related merge requests found

Changes 1

Showing 1 changed file with 5 additions and 2 deletions

Hide whitespace changes Inline Side-by-side

router\_Tokyo View file @fe78cdf8

1	Building configuration...	1	Building configuration...
2		2	
3	- Current configuration : 6467 bytes	3	+ Current configuration : 6530 bytes
4	!	4	!
5	- ! Last configuration change at 11:57:14 JST Thu Oct 13 2022 by admin	5	+ ! Last configuration change at 12:07:41 JST Thu Oct 13 2022 by admin
6	!	6	!
7	version 16.12	7	version 16.12
8	service timestamps debug datetime msec	8	service timestamps debug datetime msec
...	@@ -184,6 +184,9 @@ redundancy	...	@@ -184,6 +184,9 @@ redundancy
184	!	184	!
185	!	185	!
186	!	186	!
		187	+ interface Loopback22
		188	+ ip address 22.22.22.22 255.255.255.255
187	interface Loopback99	190	interface Loopback99
188	ip address 98.98.98.98 255.255.255.255	191	ip address 98.98.98.98 255.255.255.255

追加メッセージ

対象ファイル名

リビジョン番号

左側  
元ファイル名

追加内容

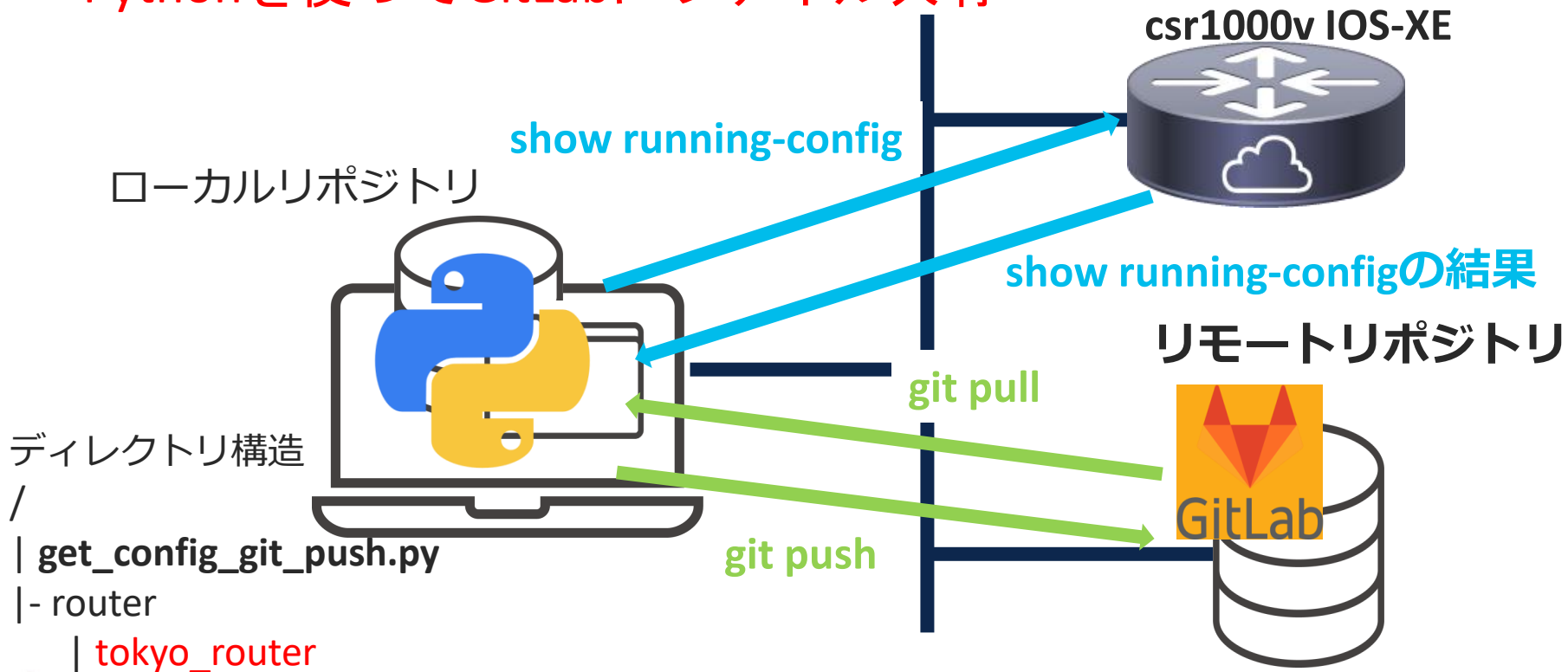
デモ 2

*Python* を使った操作



# GitLabを置き、設定ファイルを管理

- Pythonを使って設定ファイルを取得
- Pythonを使ってGitLabにファイル共有





# ルータの設定変更

- ルータの設定を変える
  - ループバックインターフェイス22のアドレスを (1.1.1.1/32) へ変更
  - ディスクリプションの追加 (Change address)

# ルータの設定ファイルを共有する

1. pythonライブラリ (gitpython) を使用する  
\$ pip install gitpython (インストールされていない場合)
2. pythonプログラムを使って、Gitコマンドの操作をする  
\$ python3 get\_config\_git\_push.py

```
$ python3 get_config_git_push.py
Username for 'http://gitlab.dev.local': root
Password for 'http://root@gitlab.dev.local':
```

# Pythonプログラム get\_config\_git\_push.py

```
import netmiko
import os
import git
```

#netmikoを使ってルータに接続し、コマンドを発行

```
session = netmiko.ConnectHandler(device_type='cisco_ios',
                                  host='192.168.0.1', username='admin',
                                  password='python123', secret='python123')
```

```
session.enable()
```

```
output = session.send_command('show run')
```

#取得してきたrunning-configの内容をファイルに書き込む

```
with open('/home/topout/router/router_Tokyo', mode='w') as f:
    f.write(output)
```

# Pythonプログラム get\_config\_git\_push.py (続き)

```
#gitのディレクトリに移動
os.chdir('/home/topout/router')
#gitライブラリのインスタンス化
repo = git.Repo()
#gitモジュールをインスタンス化
o = repo.remotes.origin
#リモートリポジトリの内容の反映 (リポートリポジトリとの同期。最新状態にするため)
o.pull()
#インデックスに追加
repo.git.add("router_Tokyo")
#コミット処理 (メッセージ追加)
repo.git.commit('.', '-m', 'change config')
#リモートリポジトリへ対象ファイルをアップロード (プッシュ) する
origin = repo.remote(name='origin')
origin.push()
```

手動でコマンドを実行していた作業を  
Pythonのライブラリを使用して、gitを実行

# リモートリポジトリでの確認 (Inlineで確認)

Showing 1 changed file ▾ with 4 additions and 3 deletions

Hide whitespace changes

Inline

Side-by-side

▼ router\_Tokyo +4 -3 View file @908577d7

```
1 1 Building configuration...
2 2
3 - Current configuration : 6530 bytes
3 + Current configuration : 6554 bytes
4 4 !
5 - ! Last configuration change at 12:07:41 JST Thu Oct 13 2022 by admin
5 + ! Last configuration change at 13:59:19 JST Thu Oct 13 2022 by admin
6 6 !
7 7 version 16.12
8 8 service timestamps debug datetime msec
... @@ -185,7 +185,8 @@ redundancy
185 185 !
186 186 !
187 187 interface Loopback22
188 - ip address 22.22.22.22 255.255.255.255
188 + description Change address
189 + ip address 1.1.1.1 255.255.255.255
189 190 .
190 191 interface Loopback99
191 192 ip address 98.98.98.98 255.255.255.255
... ..
```

変更内容

# リモートリポジトリでの確認 (Side-by-side表示)

Showing 1 changed file with 4 additions and 3 deletions

Hide whitespace changes

Inline

Side-by-side

▼ router_Tokyo		+4 -3	View file @908577d7
1	Building configuration...	1	Building configuration...
2		2	
3	- Current configuration : 6530 bytes	3	+ Current configuration : 6554 bytes
4	!	4	!
5	- ! Last configuration change at 12:07:41 JST Thu Oct 13 2022 by admin	5	+ ! Last configuration change at 13:59:19 JST Thu Oct 13 2022 by admin
6	!	6	!
7	version 16.12	7	version 16.12
8	service timestamps debug datetime msec	8	service timestamps debug datetime msec
...	@@ -185,7 +185,8 @@ redundancy	...	@@ -185,7 +185,8 @@ redundancy
185	!	185	!
186	!	186	!
187	interface Loopback22	187	interface Loopback22
188	- ip address 22.22.22.22 255.255.255.255	188	+ description Change address
		189	+ ip address 1.1.1.1 255.255.255.255
189	!	190	!
190	interface Loopback99	191	interface Loopback99
191	ip address 98.98.98.98 255.255.255.255	192	ip address 98.98.98.98 255.255.255.255

変更内容



# *Git*のwebサービス



# GitとGit webサービスの違い

- Gitとは
  - ファイルをいつ、誰が、どのように、どこを編集したかを把握することができるバージョン管理システムのこと
- Gitのwebサービスとは
  - Gitの仕組みと連携したWebサービス
    - ファイル共有をしやすくしたWebサービス
    - GitHub、GitLab、Bitbucketなど

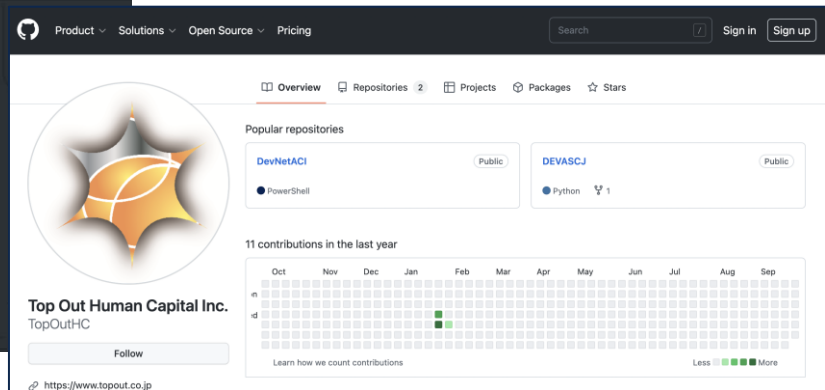




# GitHub



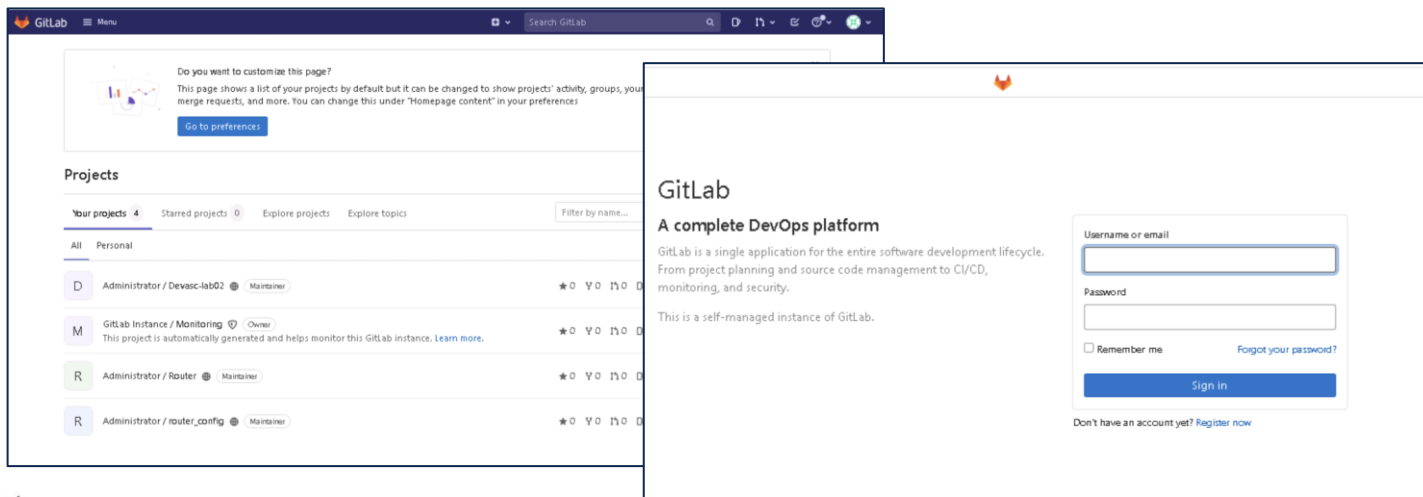
- 最も利用されているGitのサービス
  - クラウド上にて展開され、無料で手軽にプログラムコードを共有可能
- Microsoft社の製品（商用ライセンスが適応）
  - 無料で利用できるフリープランの機能拡充あり



# GitLab



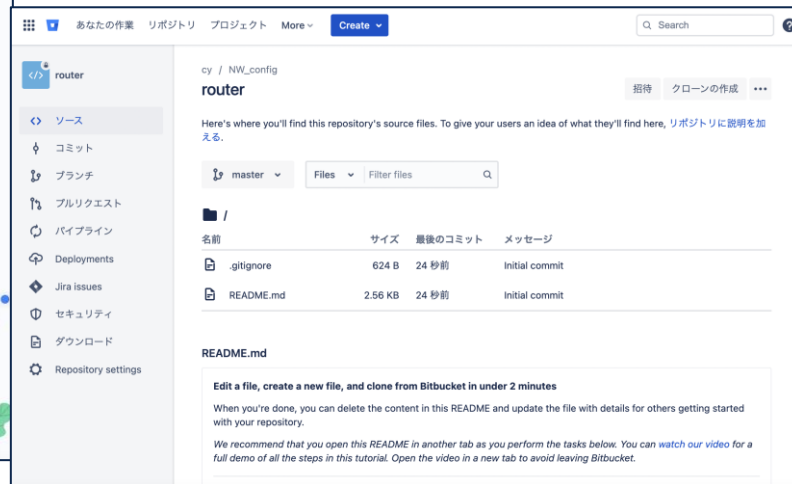
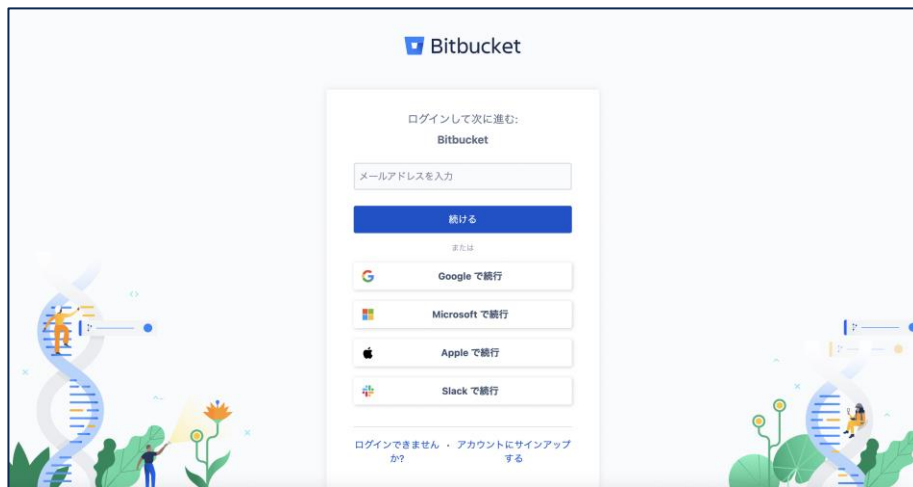
- GitHubと同等の機能を持つ
- OSSライセンスで提供
- オンプレミスで構築すると、無料で自社サーバ内にGit構築可能



# Bitbucket



- ATlassian(アトラシアン)社にて提供
  - 無料プランも提供されている
- ATlassianのDevOpsサービスに含まれている



# *Git*の活用



# ネットワーク管理

- コンフィグの一元管理
  - 複数台、複数の機種種のルータなどのNW機器が存在している時
- コンフィグのバックアップ
- 設定の更新のログを残しておきたい時
- 復元処理をしたい時
- パラメータシートとの管理
- 構成図などの図面管理



# NW機器との連携

- guestshellとの連携
  - guestshell上でgit cloneをしてプログラム追加
    - On-box-pythonを利用した、EEMおよびCisco Teamsとの連携など可能



# NW機器との連携 (EEM、guestshell連携)



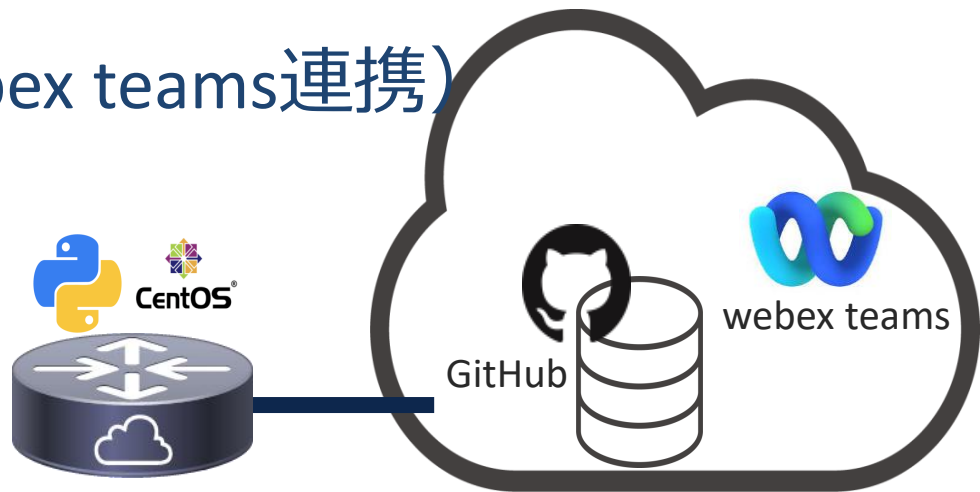
## guestshellでのGitHubからのclone

```
[guestshell@guestshell ~]$ git clone https://github.com/TopOutHC/DEVASCJ.git
Cloning into DEVASCJ'...
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), done.
```

## EEMの設定例 (action 2.0でguestshellからpythonプログラムを実行)

```
ROUTER# conf t
ROUTER(config) # event manager applet config-diff-to-spark
ROUTER(config-applet)# event 1.0 cli command "enable"
ROUTER(config-applet)# action 1.0 cli command "guestshell run python /home/guestshell/DEVASCJ/notice.py"
ROUTER(config-applet)# action 3.0 regexp "ERROR" "$_cli_result"
ROUTER(config-applet)# action 4.0 if $_regexp_result eq "1"
ROUTER(config-applet)# action 5.0 syslog msg "$_cli_result"
ROUTER(config-applet)# action 6.0 end
ROUTER(config-applet)# end
```

# NW機器との連携 (EEM、guestshell、webex teams連携)



```
DevNet Studentbot 18:44
Configuration differences between the running config and the last backup:
1 | !Contextual Config Diffs:
2 | +interface Loopback33
3 | +description TEST interface
4 | +no ip address
```



まとめ

# Gitを知ることができましたか？



- ・ 設定ファイルを上書きしてしまった
  - ・ Gitを使って、設定ファイルの履歴管理ができる！
- ・ 先週は動いていたのに、今日試したら動かなかった
  - ・ 今度もし、動かないという苦情が来ても、何とかかなりそうだ！
- ・ 過去の構成と比較をするが、変更点を探すのに非常に時間がかかる
  - ・ 今まで苦労していた作業が短縮できそう。その分、もっとGitを学習しよう！
- ・ DXやDevOpsについて調べると言われてるけど・・・
  - ・ GitもDevOpsの一部なんだ。他にもっと知ってみたい！

ご紹介

# 関連コースの紹介

- ◆ DevNet関連コース <https://www.topout.co.jp/cisco>
  - DevNet Associate対応コース
    - DEVASC (Developing Applications and Automating Workflows using Cisco Core Platforms)
  - DevNet Specialist / Professional対応コース
    - DEVCOR (Developing Applications Using Cisco Platforms and APIs)
- 前提条件のPythonの習得に <https://www.topout.co.jp/python>
  - Python 初級編・中級編・ネットワーク編・自動化編・サーバ編
- DevOpsを知ってもっと作業を円滑に <https://www.topout.co.jp/devops>
  - DevOpsや、SRE (サイト信頼性エンジニアリング)など幅広く

# DevOps Institute CEO 初来日記念イベント



「SKILup Festival Japan  
- Site Reliability Engineering (SRE) DAY -」

2022年 **11**月 **9**日 (水)

**13:30~16:30**  
(受付 13:00~)

会場：ベルサール八重洲

ご参加の皆様にはもれなく  
**Amazonギフト券1,000円分**  
を進呈します！

ご清聴ありがとうございました

# Thank You

# Q&A

画面右側の Q&A ウィンドウから、  
すべてのパネリスト (All Panelists) 宛  
に送信してください。





# 次回の オンラインセミナー予定



## Catalyst 9800 ワイヤレス コントローラ - 概要とトラブルシューティング

2022 年 11 月 16 日 (水) 10:00 - 11:30

### 李 海峰 (Haifeng Li)

シスコシステムズ

グローバル カスタマー エクスペリエンス センター

テクニカル コンサルティング エンジニア

登録受付中

<https://community.cisco.com/t5/e-/-/ec-p/4696498>



# ユーザーガイド ビデオのご案内



シスコ コミュニティでは、新たに下記のビデオを公開しました

- ▶ [シスコ コミュニティの参加方法およびプロフィールの作成](#)
- ▶ [コンテンツを検索する、エキスパートに質問する、質問へ回答する、感謝を伝える](#)
- ▶ [コンテンツの購読設定と通知の管理](#)
- ▶ [メンバー表彰プログラムとプロジェクト ギャラリー](#)

# スポットライト アワード



シスコ コミュニティでは、毎月アワード受賞者を紹介しています。

9月の受賞者: **cja56910tf** さん

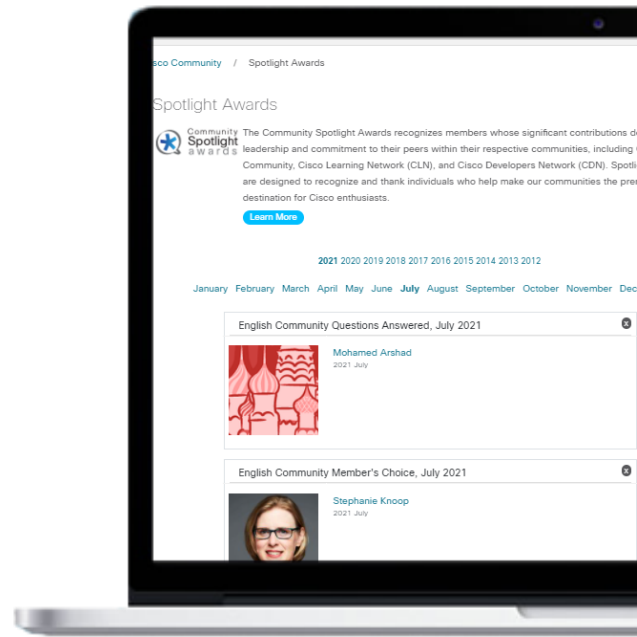
みなさまからのアワードへの推薦もお待ちしています！

このボタンから推薦できます



推薦したいメンバーがいます

Spotlight Award に推薦する



# ニュースレターの購読と シスコ コミュニティガイド



このラベルを購読設定

事務局ニュース

- メール版：アンケートで「メール送信可」にチェック
- Web 版：コミュニティサイトより設定可能

設定方法はこちら

<https://community.cisco.com/t5/-/-/ba-p/4424603#toc-hld--1269189899>

ついに公開！  
シスココミュニティの歩き方  
2020年2月版  
利用方法や小技集など  
All in One 役立ちガイド  
詳しくはこちら



ガイドもご活用ください

<https://community.cisco.com/t5/-/-/ta-p/4021064>



ご参加ありがとうございました。

Community Liveと Cisco Communityの  
各アンケートにも ぜひ ご協力ください。

