AVD Extended Workshop

Intro into Ansible, Ansible AVD, Git and VSCode for new and existing AVD users

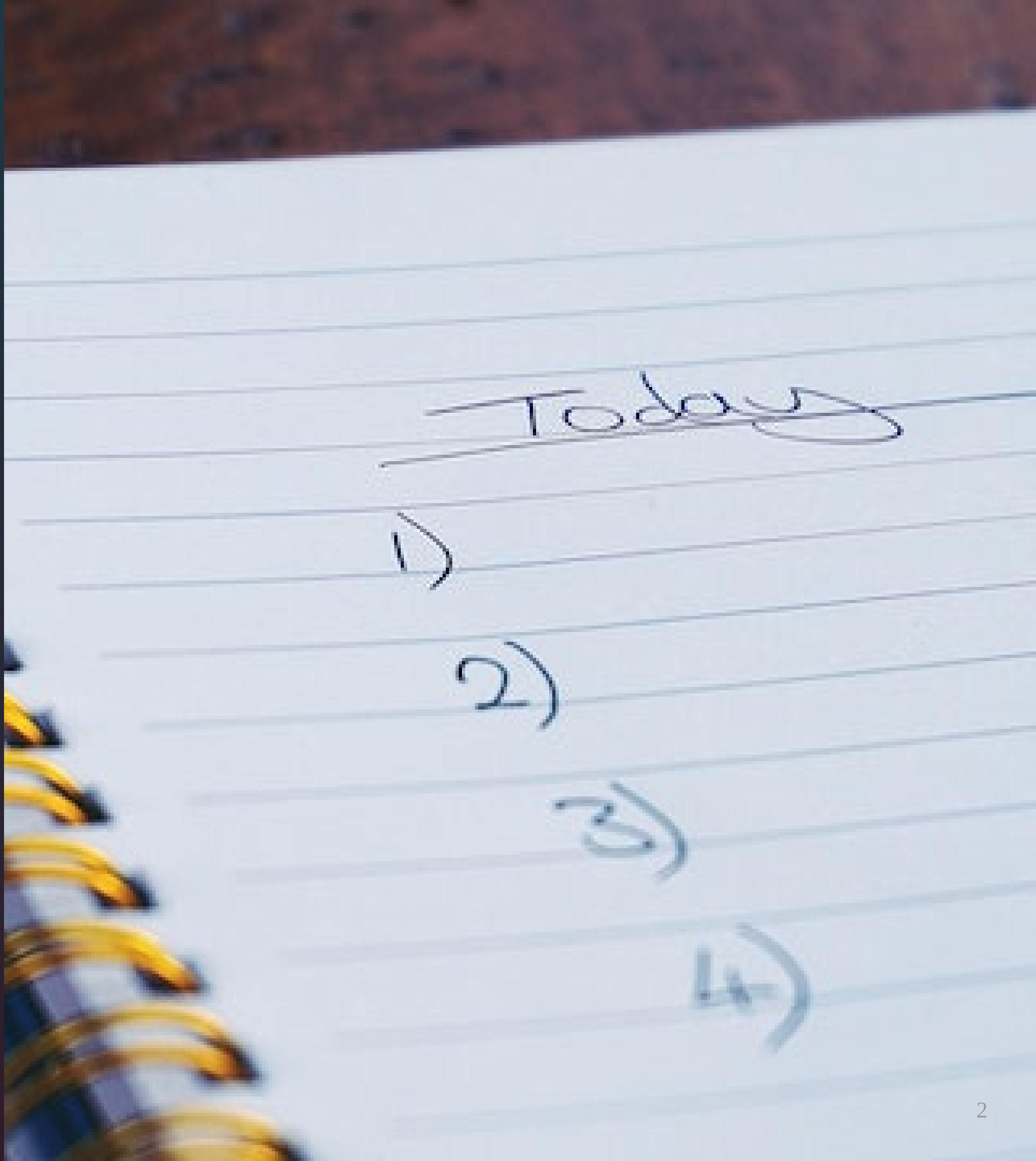
What is this Workshop about?

Topics:

- Git basics
- VSCode basics
- Ansible basics
- Containers basics and using container to build Ansible AVD environment
- Building simple L3LS network with Ansible AVD

Structure:

- This workshop is split into 3 sections. Each section takes around 2 hours to complete. That can be done as a full day workshop or split into 3 separate sessions.
- Make a break when you see a slide with a coffee cup ☕
- Ask questions at any time!



What is NOT covered in this Workshop?

- This workshop is not a deep dive into each and every topic. It is covering some advanced concepts, but you may need additional documentation and training to understand every topic in details.
  For additional information please refer to the following resources:
  - Ansible AVD Documentation
  - VSCode Documentation
  - Git Documentation - Pro Git book is a good start
  - Container Trainings by @jpetazzo:
    - Github repository
    - Training materials
- We are not going to use Arista CloudVision Portal (CVP) in this workshop. It provides a lot of advantages, but is not essential to understand the concepts covered in this workshop.
- If you will not find something you expect in this workshop, there can be 2 reasons:
  - It is not covered in this workshop
  - It is waiting for your contribution to this repository! 🤝

Requirements

- You **MUST** have a Github account ❗
  Register here.

References

- If you are not using ATD, the functionality of this repository will rely on many amazing open source projects:
    - ContainerLab
    - VSCode
    - DevContainers
    - Marp
    - Excalidraw VSCode
- This repository is also relying on following free/commercial Github features:
    - Github Actions
    - Github Pages
    - Github Codespaces
- All photos are taken from Pexels and Unsplash. Excellent free stock photos resources. It's not possible to reference every author individually, but their work is highly appreciated.

Introducing The Tools

`Section 1.1`

- The bird view on the tools we are going to use in this workshop.
- No details, they will come in a later sections. Just and overview.

What is Git?

- **In Short**:

  > Git is a distributed version control system that tracks changes to a set of files and enables collaborative work.

- **Fun Fact**:

  > Git was created by Linus Torvalds in 2005 to develop Linux kernel.

What is GitHub?

- GitHub is a Git repository hosting platform.
- Allows to coordinate multiple local copies of the same repository and more.

Before We Start

`Section 1.2`

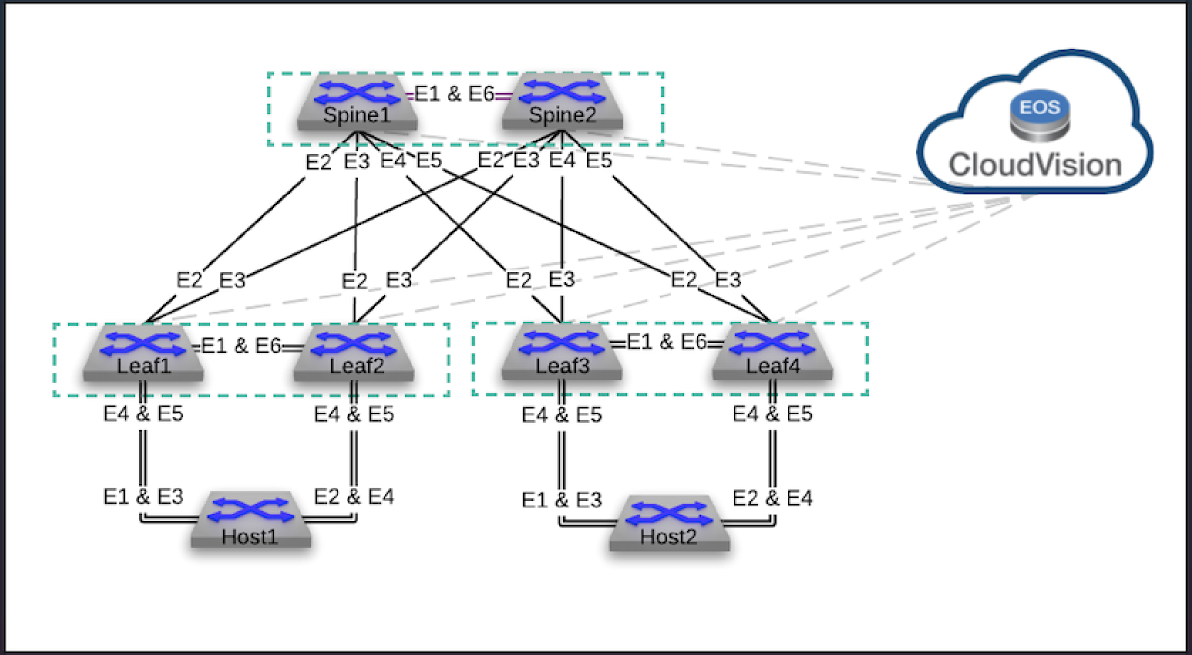
- How to get your lab environment up and running

How to use this Workshop?

- To try all practical examples you need to have access to the lab environment. There are 3 possible options:
    - Use Github Codespaces. This is the preferred option, but double check that you understand all the costs and free tier limits.
    - Use Arista Test Drive - Single DC topology. Please ask your Arista SE to create an ATD lab for you.
    - Build your own lab environment: Ubuntu LTS + Docker + ContainerLab. This option is not described in detail, but the devcontainer used to build Codespaces environment will work on any machine with Docker installed. Please contact your SE if you need help.

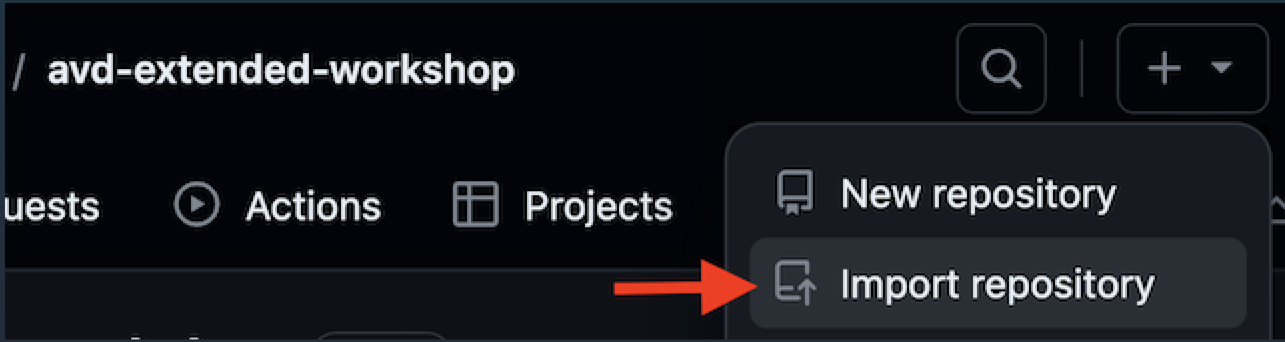
Lab Topology

- This workshop is using Arista Test Drive Single DC topology.
- To match minimize resources and fit default Codespaces 4-core machine, the topology was reduced by removing leaf3, leaf4, host1 and host2.
- Feel free to adjust Ansible inventory and group variables if you are using ATD lab and would prefer to activate them all. But it's not essential for this workshop.
- CVP is not used as it's not required for this workshop.



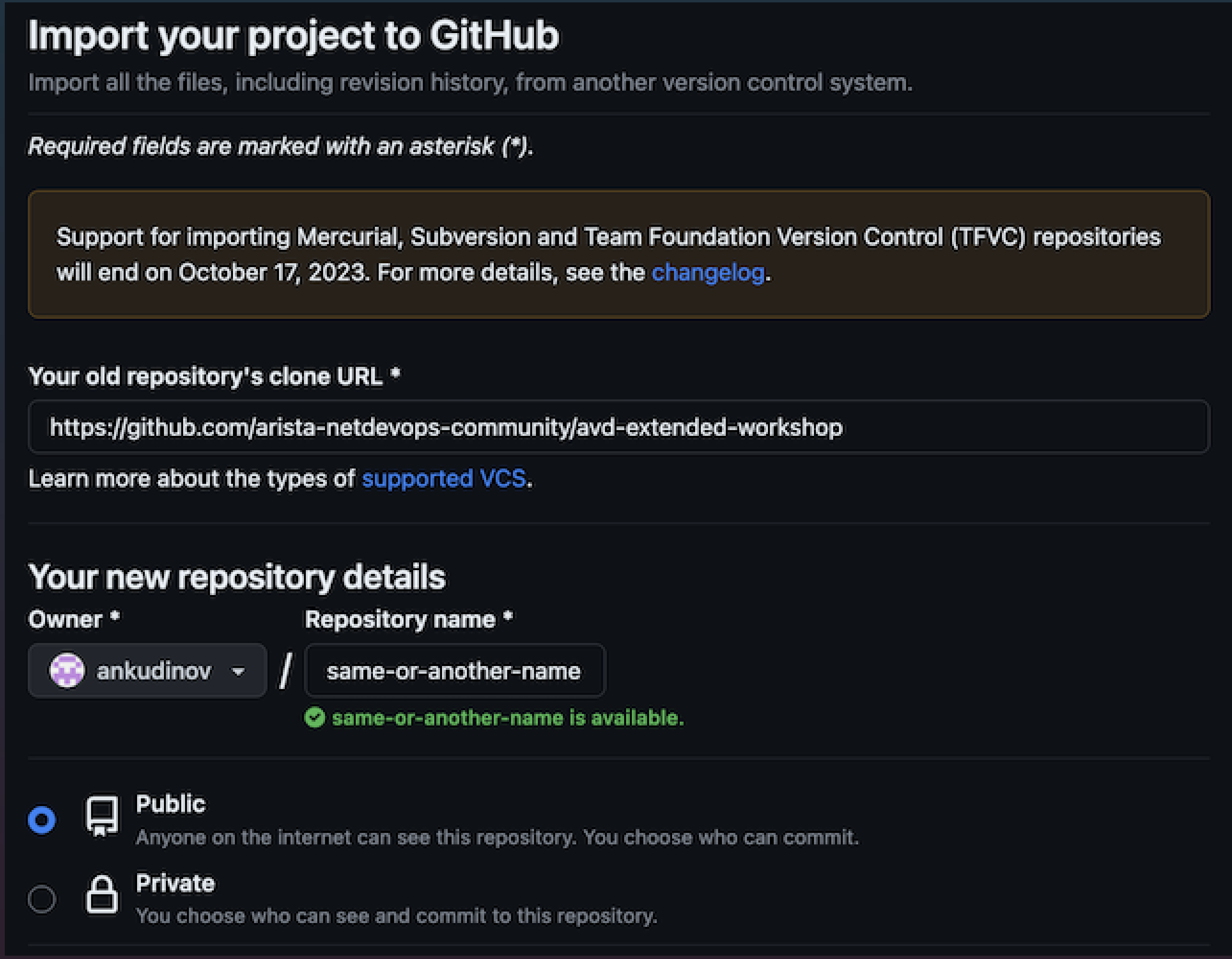
Github Repository Import

- Create a copy of this repository on your Github account. That will allow you to make any changes without impacting the original repository.
- Alternatively you can fork this repository, but in this case you must **NOT** (❗) open any pull requests to the original repository.
- To make a copy of the repository click + button in the top right corner of the Github page and select `Import repository` option.



Github Repository Import, Step 2

- Enter the following URL in `Your Old Repository's Clone URL` field:
  - `https://github.com/arista-netdevops-community/avd-extended-workshop`
- Use your own account in `Owner` field and `avd-extended-workshop` or another name in the `Repository Name` field.
- Create `Public` repository. That will simplify interaction with this repo and allow use of Github free features.
- Wait until the import is completed.
- Your clone will now be referenced as `<your-copy-of-this-repository>` in this workshop.



Import your project to GitHub

Import all the files, including revision history, from another version control system.

Required fields are marked with an asterisk (*).

Support for importing Mercurial, Subversion and Team Foundation Version Control (TFVC) repositories will end on October 17, 2023. For more details, see the changelog.

Your old repository's clone URL *

https://github.com/arista-netdevops-community/avd-extended-workshop

Learn more about the types of supported VCS.

Your new repository details

Owner *: ankudinov / Repository name *: same-or-another-name

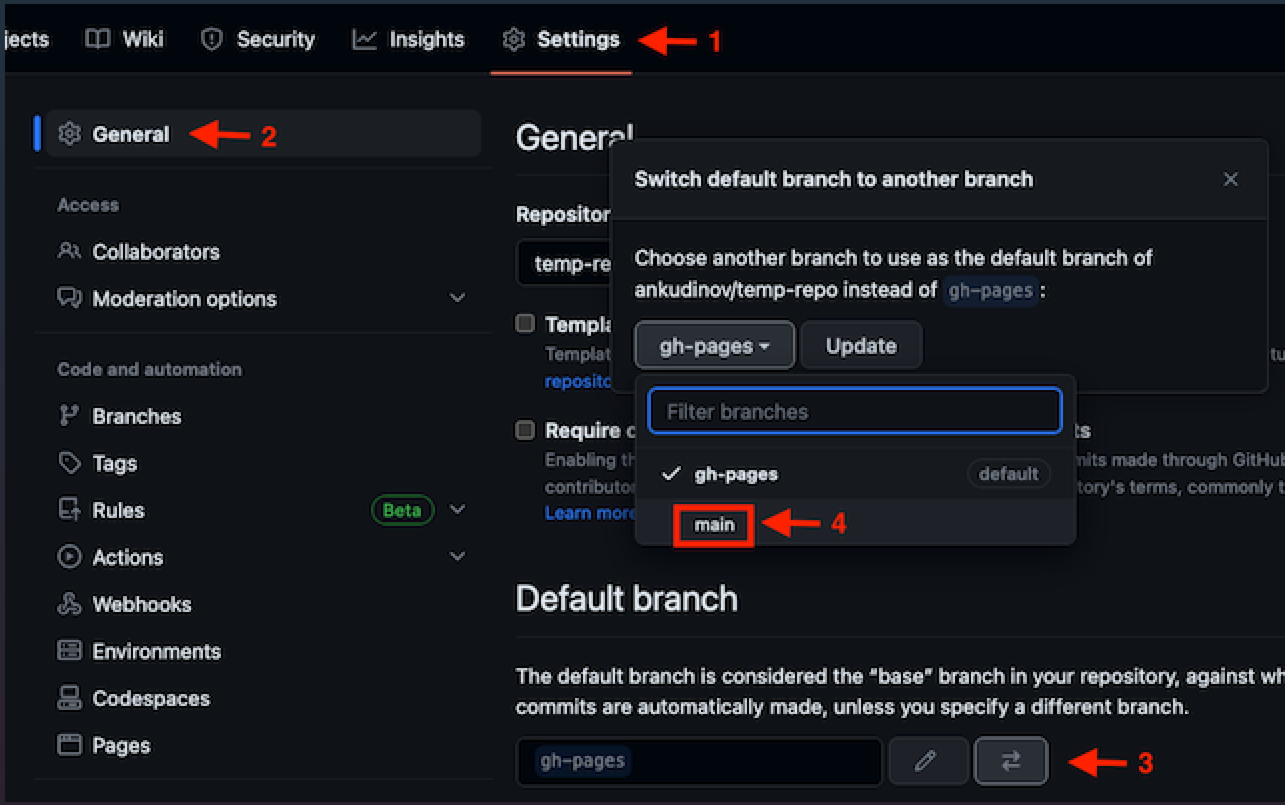
same-or-another-name is available.

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private You choose who can see and commit to this repository.

Github Repository Import, Step 3

- Confirm that `main` is the default Git branch after the import.
- Click `Settings` tab in the top right corner of the Github page.
- Click `General` on the left panel.
- Scroll down to `Default branch` section, click `Switch to another branch` button and select `main` branch.
- All set! 🎉



Choose another branch to use as the default branch of ankudinov/temp-repo instead of `gh-pages`:

The default branch is considered the "base" branch in your repository, against wh commits are automatically made, unless you specify a different branch.

How to Setup ATD Environment

`Section 1.3`

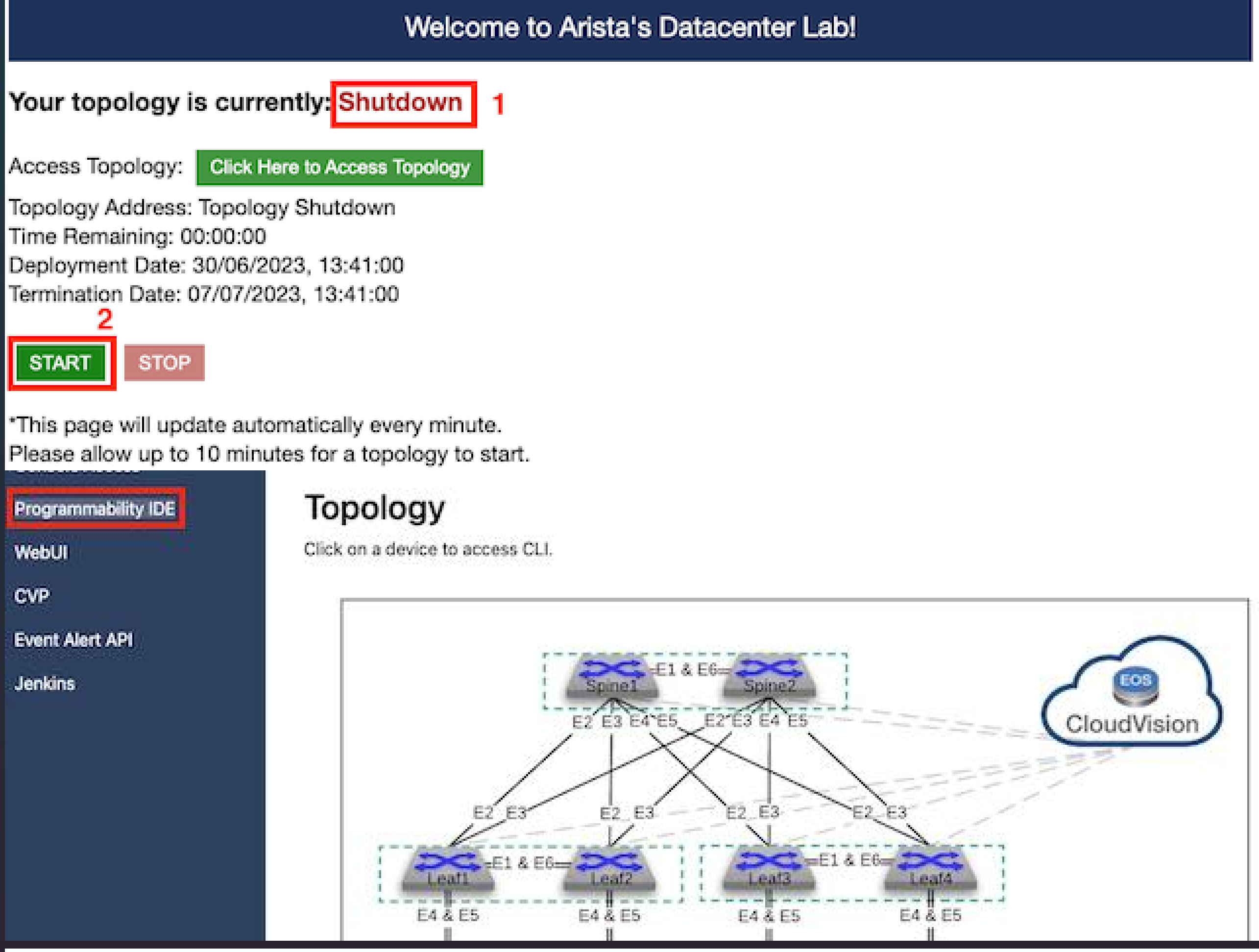
> - skip practice this section if you are using Codespaces
> - still read the slides as they explain AVD installation process

How to setup Ansible AVD in Arista Test Drive environment?

- We could use a script to setup required Ansible collections and tools in Arista Test Drive environment, but it's a good opportunity to discuss what are the requirements but installing them manually.
- For details please check AVD documentation `Installation > Collection Installation` section.

Open Programmability IDE

- Use the lab token provided by Arista representative to access the lab environment.
- Check the status of the lab environment. If it's `Shutdown` - click `Start` button.
- Click `Programmability IDE` button to open VSCode in the browser:
    - To access `Programmability IDE` use the password listed on the starting Web page.
    - The VSCode functionality in the Web browser is provided by ATD Code server container
- Click `Yes, I trust the authors` button to continue. 🕵️
- Open new terminal in VSCode: `Top Left Corner (3 parallel lines) > Terminal > New Terminal`


Welcome to Arista's Datacenter Lab!

Your topology is currently: Shutdown 1

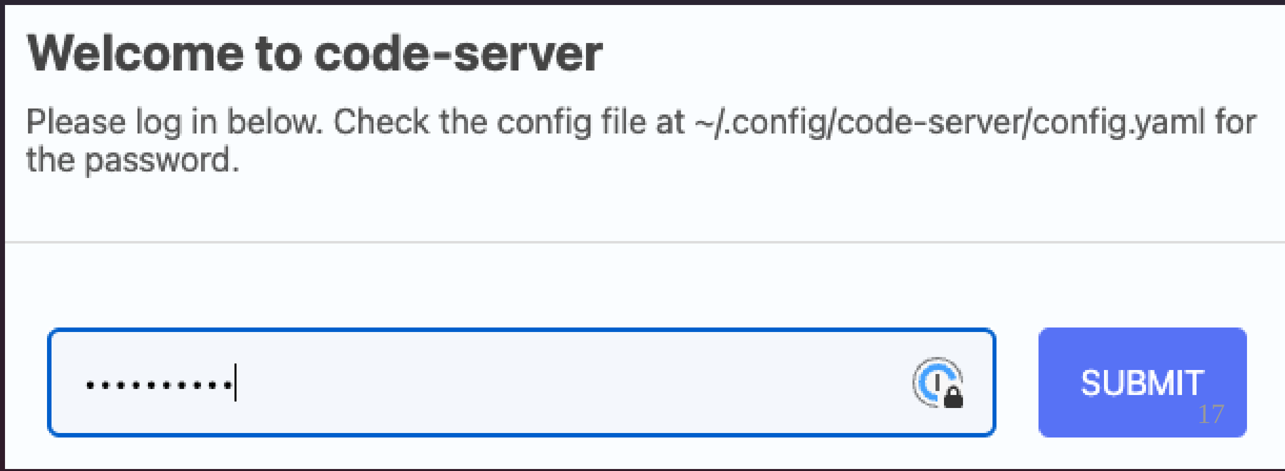
Access Topology: Click Here to Access Topology

Topology Address: Topology Shutdown
Time Remaining: 00:00:00
Deployment Date: 30/06/2023, 13:41:00
Termination Date: 07/07/2023, 13:41:00

2

START STOP

*This page will update automatically every minute.
Please allow up to 10 minutes for a topology to start.


Welcome to code-server

Please log in below. Check the config file at ~/.config/code-server/config.yaml for the password.

Install Ansible AVD

```
# 1. Update package index files
sudo apt-get update

# 2. Install iputils as life is hard without ping
sudo apt-get install -y --no-install-recommends iputils-ping

# 3. Add .local/bin in home folder to PATH
export PATH=$PATH:/home/coder/.local/bin

# 4. Upgrade pip and install Ansible core
#    If you get errors, ignore. This bug will be fixed soon.
pip install --upgrade pip
pip3 install "ansible-core>=2.13.1,<2.14.0"

# 5. Install Arista AVD collection
ansible-galaxy collection install arista.avd:==4.1.0

# 6. Install AVD collection requirements
pip3 install -r /home/coder/.ansible/collections/ansible_collections/arista/avd/requirements.txt
```

For additional details check Arista Ansible AVD Collection installation docs.

Ansible Installation Warnings

- Double check that the path to Ansible collection is correct. Normally it is expected to be in `/home/coder/.ansible/`
- You `PATH` environment variable must be set correctly!
- Never install Ansible as root user!
- Watch out for environments with a long history, conflicting Python installations etc.
- Containers make it simple! Use containers! 🐳

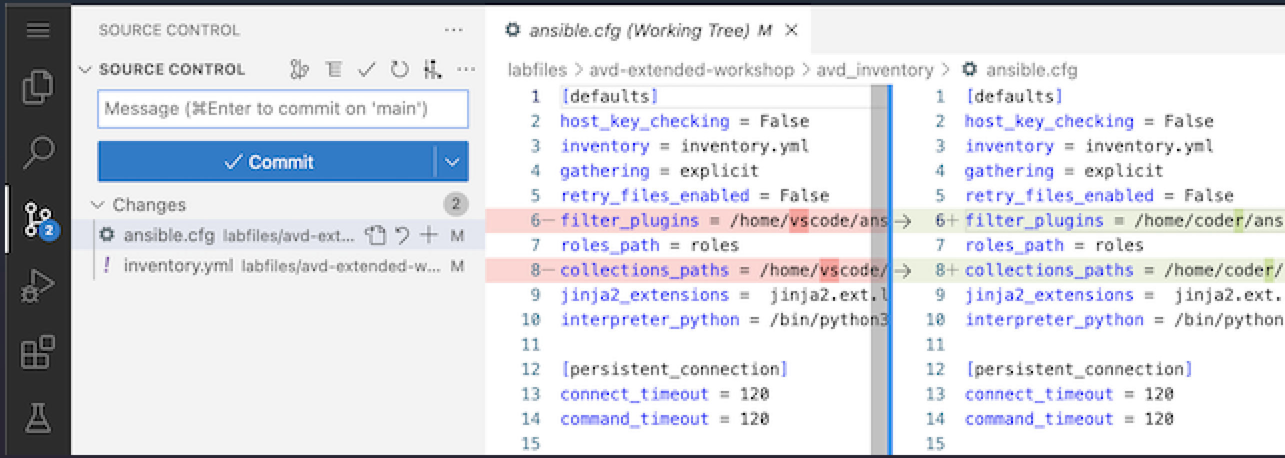
  > The Codespaces environment is based on a container with all requirements installed.

Setup Ansible AVD Repository

```
# 1. install yq to adjust AVD yaml files - https://github.com/mikefarah/yq
#    you can certainly edit yaml files manually, but there would be no fun 👎
export VERSION="v4.34.1" BINARY="yq_linux_amd64"
sudo wget https://github.com/mikefarah/yq/releases/download/$VERSION/$BINARY -O /usr/bin/yq \
    && sudo chmod +x /usr/bin/yq
# 2. Clone your copy of this repository
cd labfiles
git clone https://github.com/<gh-handle>/<your-copy-of-this-repository>.git avd-extended-workshop
# 3. switch to the repository directory
cd avd-extended-workshop
# 4. confirm that you are working with the `main` branch
#    if not, type following command to change the branch
git checkout main
#    you should see the following prompt
▫  avd-extended-workshop git:(main)
# 5. update ansible.cfg to match ATD container user
cp extras/ansible-avd.cfg avd_inventory/ansible.cfg
# 6. set Ansible password to your AVD environment password
yq -i '.all.vars.ansible_password = "<your-password>"' avd_inventory/inventory.yml
```

Commit Changes to Git

- Click VSCode `Source Control` icon in the left panel.
- Click `+` button to stage all changes. Alternatively you can accept VSCode suggestion to do that automatically every time by selecting `Always` option.
- Enter a *meaningful* commit message in the `Message` field.
- Click `Commit` button.


SOURCE CONTROL
SOURCE CONTROL
Message (⌘Enter to commit on 'main')
✓ Commit
Changes 2
ansible.cfg labfiles/avd-ext... M
inventory.yml labfiles/avd-extended-w... M
ansible.cfg (Working Tree) M
labfiles > avd-extended-workshop > avd_inventory > ansible.cfg

```
[defaults]
host_key_checking = False
inventory = inventory.yml
gathering = explicit
retry_files_enabled = False
filter_plugins = /home/vscode/ans
roles_path = roles
collections_paths = /home/vscode/
jinja2_extensions =  jinja2.ext.l
interpreter_python = /bin/python3

[persistent_connection]
connect_timeout = 120
command_timeout = 120

```

```
[defaults]
host_key_checking = False
inventory = inventory.yml
gathering = explicit
retry_files_enabled = False
filter_plugins = /home/coder/ans
roles_path = roles
collections_paths = /home/coder/
jinja2_extensions =  jinja2.ext.
interpreter_python = /bin/python

[persistent_connection]
connect_timeout = 120
command_timeout = 120

```

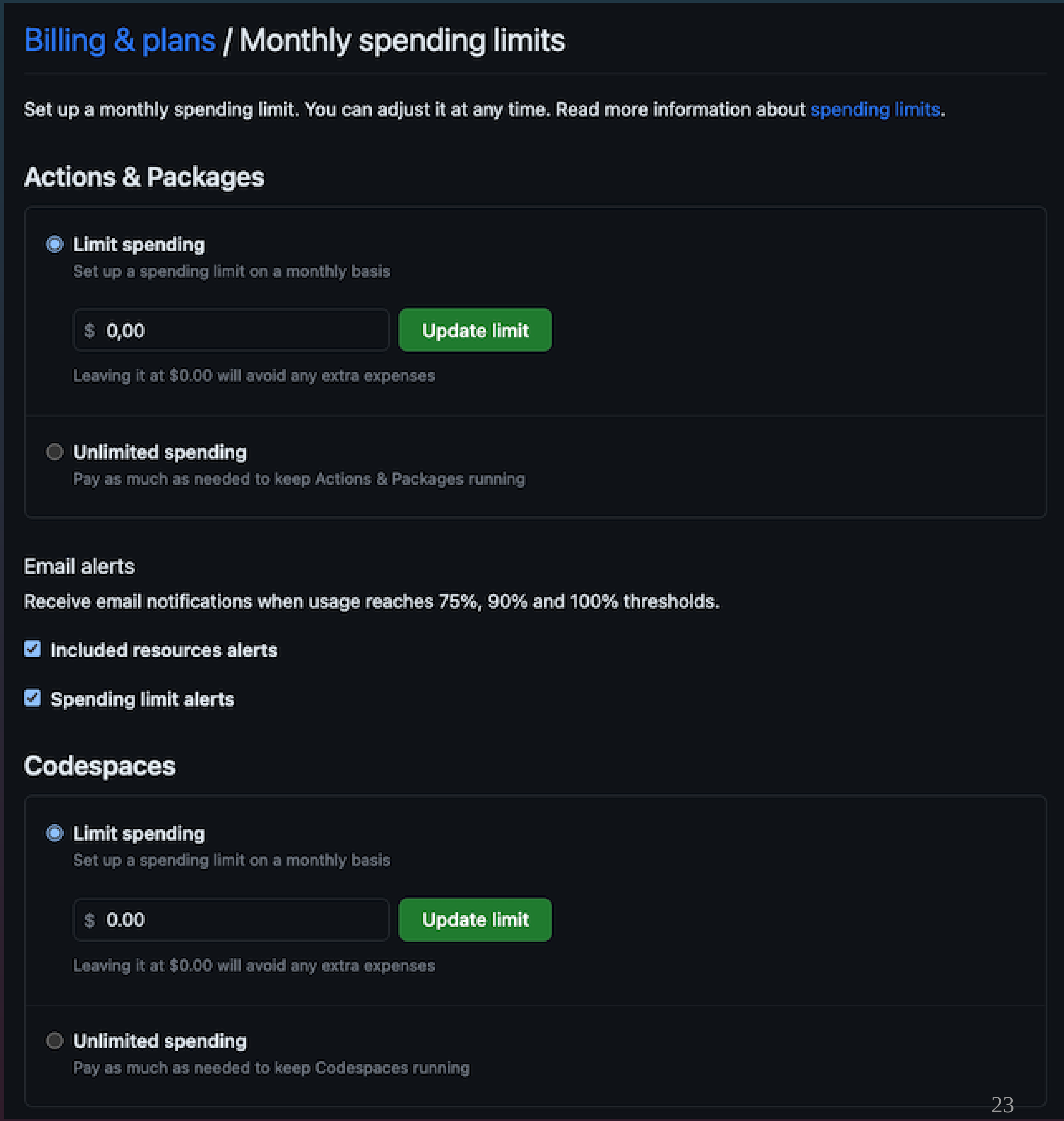
Prepare Github Codespaces Environment

`Section 1.4`

- you can skip this section if you are using ATD lab
- still read the slides as they explain how to use Codespaces

Before You Create Codespaces Environment

- Codespaces is a paid feature. Please check Github Codespaces pricing
- It has a free tier for personal accounts:
  - 120 core-hours per month -> will be 30 hours on a 4-core machine
  - 15 GB storage per month -> this will be a bottleneck for the workshop container image
- The free tier is enough to complete this workshop, but don't forget to delete the Codespaces environment after the workshop.
- Check `your account > Settings > Billing and plans > Spending limits` to make sure that if you exceed the limit, there will be no charges. The default limit of `0.00` will avoid any extra expenses.



Billing & plans / Monthly spending limits

Set up a monthly spending limit. You can adjust it at any time. Read more information about spending limits.

Actions & Packages

[● Limit spending] Set up a spending limit on a monthly basis

$ 0,00

Leaving it at $0.00 will avoid any extra expenses

○ Unlimited spending — Pay as much as needed to keep Actions & Packages running

Email alerts

Receive email notifications when usage reaches 75%, 90% and 100% thresholds.

[✓] Included resources alerts

[✓] Spending limit alerts

Codespaces

[● Limit spending] Set up a spending limit on a monthly basis

$ 0.00

Leaving it at $0.00 will avoid any extra expenses

○ Unlimited spending — Pay as much as needed to keep Codespaces running

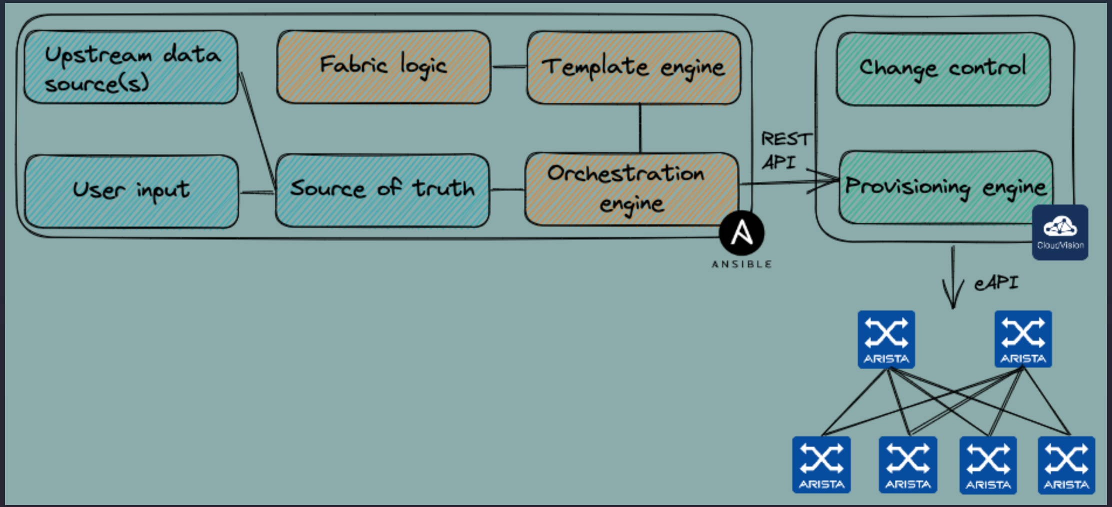
Run First AVD Playbooks

`Section 1.5`

- Just build an EVPN network with Ansible AVD and enjoy the result!

Typical Ansible AVD Automation Workflow

- Collect user input from various data sources and aggregate in a single source of truth. For ex. git repository.
- Generate low level variables from abstracted input data using sophisticated fabric logic
- Parse Jinja2 templates to produce plain text configs
- Push plain text configs via CloudVision Portal as change-control "proxy" or directly to devices via eAPI.



Run Ansible AVD Playbooks

```
# 1. switch to AVD inventory directory
cd ~/project/labfiles/avd-extended-workshop/avd_inventory
# 2. run ansible-playbook to generate configs
#    wait until the playbook will finish execution and check the configs in avd_inventory/intended/configs
ansible-playbook playbooks/atd-fabric-build.yml
# 3. run ansible-playbook to push configs to devices
ansible-playbook playbooks/atd-fabric-provision-eapi.yml
# 4. Done! 🎉 Click on on any lab switch and check `show bgp evpn summary`
```

Playbook execution example:

```
▫  avd_inventory git:(main) ▫ ansible-playbook playbooks/atd-fabric-provision-eapi.yml

PLAY [Deploy Configs] ***********************************************************************************************************************************************************************************************

TASK [arista.avd.eos_config_deploy_eapi : Verify Requirements] ******************************************************************************************************************************************************
AVD version 4.1.0
Use -v for details.
ok: [spine1 -> localhost]

TASK [arista.avd.eos_config_deploy_eapi : Create required output directories if not present] ************************************************************************************************************************
changed: [spine1 -> localhost] => (item=/home/coder/project/labfiles/avd-extended-workshop/avd_inventory/config_backup)
ok: [spine1 -> localhost] => (item=/home/coder/project/labfiles/avd-extended-workshop/avd_inventory/config_backup)

TASK [arista.avd.eos_config_deploy_eapi : Replace configuration with intended configuration] ************************************************************************************************************************
[WARNING]: To ensure idempotency and correct diff the input configuration lines should be similar to how they appear if present in the running configuration on device including the indentation
changed: [spine1]
changed: [spine2]
changed: [leaf1]
changed: [leaf2]

RUNNING HANDLER [arista.avd.eos_config_deploy_eapi : Backup running config] *****************************************************************************************************************************************
changed: [spine1]
changed: [spine2]
changed: [leaf1]
changed: [leaf2]

PLAY RECAP **********************************************************************************************************************************************************************************************************
leaf1                      : ok=2    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
leaf2                      : ok=2    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
spine1                     : ok=4    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
spine2                     : ok=2    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Useful eAPI Troubleshooting Trick

If you are facing any issues when to push configs or collect any data using eAPI, test access with the following command:

```
curl --user <login>:<password> --data "show version" --insecure https://<switch-mgmt-ip>:443/command-api --verbose
```

Try it now! 🔨

With `--verbose` it can tell you a lot.