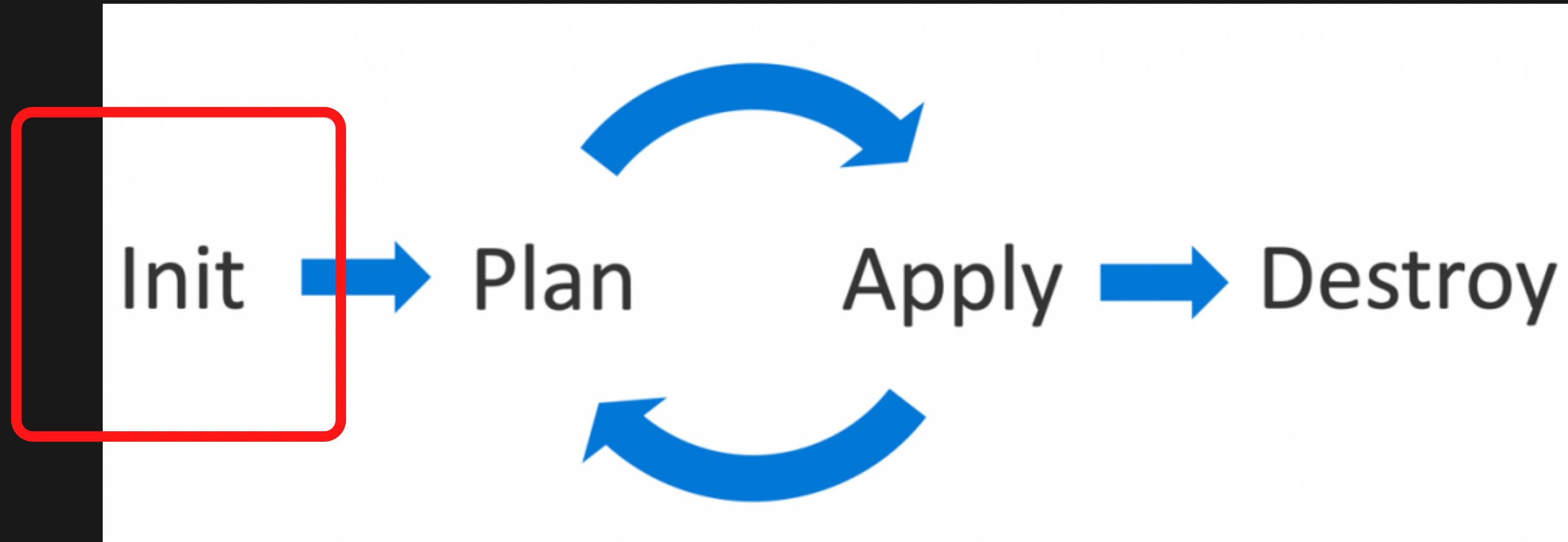


Terraform Init

Terraform Init



Terraform Init

- **terraform init**
- Usually ran as the first command in the workflow
- Initializes the working directory
 - Pulls in any recent Provider changes (more on Providers later)
 - Configures any State backends (more on States & Backends later)
 - Downloads any missing Provider & Module binaries/references (more on Modules later)
- Safe command to run - run multiple times
- When to run the command?
 - On a brand new Terraform project
 - When pulling any recent changes from Github / source control
 - When first working on a Terraform project
- Always run the command when first working on a project, for that day

Terraform Init

- Several things happen in the background
- Creates `.terraform` folder
 - Acts as local cache - stores Provider & Module binaries and references
 - Helps improve Terraform performance
 - Do not push this folder into your repository
- Creates a `.terraform.lock.hcl` file
 - Helps Terraform manage the Provider version
 - Locks the Provider version to the latest acceptable version
 - If the version specifies 1.0.0 - Terraform will always download the 1.0.0 Provider version
 - Push this into your repository
- They are automatically managed by Terraform

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

commands will detect it and remind you to do so if necessary.
PS C:\Users\tekke\Desktop\levelup_with_terraform\workflow_basics> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 3.0"...
- Installing hashicorp/aws v3.63.0...
- Installed hashicorp/aws v3.63.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\tekke\Desktop\levelup_with_terraform\workflow_basics> █
```

Terraform Init

- Downloads defined Providers
 - Stores the Provider binary into the `.terraform` folder
 - In our example, we are using the AWS Provider
- Initializing the backend
 - Backend is a central storage used to hold your state files
 - Consults your config to see where your State is stored
 - Pulls in state if using remote state, or uses the local state file
- Terraform reads your top level global configuration, to see where your State is stored, what Provider version you're using, etc.

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL

commands will detect it and remind you to do so if necessary.
PS C:\Users\tekke\Desktop\levelup_with_terraform\workflow_basics> terraform init

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching "~> 3.0"...
- Installing hashicorp/aws v3.63.0...
- Installed hashicorp/aws v3.63.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\Users\tekke\Desktop\levelup_with_terraform\workflow_basics> █
```

Terraform Init

- You have to run **terraform init** whenever there is a change made to your top level terraform configuration
 - Example 1 - If you changed the Provider version, you need to run **terraform init** to re-download the updated Provider version
 - Example 2 - If you changed your State to use a different S3 bucket backend, need to run **terraform init** to re-initialize your State backend
- Same thing applies to modules
 - If you added a new Module, changed an existing Module, removed a Module, need to run **terraform init** to download the new Module binary
 - TLDR - run **terraform init** whenever a change is made to your Modules and Providers
 - BUT - If you are using local modules, you do not need to run terraform
 - Terraform stores a local reference for local Modules
 - This is awesome! Means you can keep making changes to your local Modules, without running **terraform init** everytime