**Terraform workspaces refer to separate instances of state data inside the same Terraform working directory.**

**Here are some of the key commands related to Terraform workspaces:**

**NOTE –**

Terraform workspaces are like separate folders for your infrastructure.

They let you manage different sets of resources (like cloud servers) using the same Terraform configuration, but keep them independent.

Think of it like having separate playbooks for development, staging, and production environments.

So, you can use one Terraform codebase to create different versions of your infrastructure without accidentally messing things up in another environment. Simple and neat!

terraform workspace new <name> **: Creates a new workspace**

terraform workspace select <name> **: Selects an existing workspace**

terraform workspace list **: Lists all available workspaces**

terraform workspace show **: Shows the currently selected workspace**

terraform workspace delete <name> **: Deletes an existing workspace**

Remember, for a given working directory, you can only select one workspace at a time¹.

Most Terraform commands only interact with the currently selected workspace¹.

This includes provisioning and state manipulation¹..

**Sure, here are examples of Terraform workspace commands for workspaces named `dev`, `qa`, and `uat`:**

**- Creating a new workspace:**

**- For `dev`:** terraform workspace new dev

**- For `qa`:** terraform workspace new qa

**- For `uat`:** terraform workspace new uat

**- Selecting an existing workspace:**

**- For `dev`:** terraform workspace select dev

**- For `qa`:** terraform workspace select qa

**- For `uat`:** terraform workspace select uat

**- Listing all available workspaces:**

terraform workspace list

- This will list all workspaces including `dev`, `qa`, and `uat`.

**- Showing the currently selected workspace:**

terraform workspace show

- This will show the currently selected workspace, which could be `dev`, `qa`, or `uat` depending on the last `select` command.

**- Deleting an existing workspace:**

**- For `dev`:** terraform workspace delete dev

**- For `qa`:** terraform workspace delete qa

**- For `uat`:** terraform workspace delete uat

Please replace `dev`, `qa`, and `uat` with your actual workspace names. Also, ensure you're in the correct directory where your Terraform configuration files are located before running these commands. Remember, most Terraform commands only interact with the currently selected workspace. This includes provisioning and state manipulation. So, be careful while selecting and deleting workspaces.

**If you have separate `.tfvars` files for different environments, you can specify the appropriate file when running the `terraform apply` command using the `-var-file` option.**

Here's how you can do it for `dev`, `qa`, and `uat` environments:

**- For `dev` environment:**

**1. Select the `dev` workspace:** terraform workspace select dev

**2. Apply the configuration:** terraform apply -var-file=dev.tfvars

**- For `qa` environment:**

**1. Select the `qa` workspace:** terraform workspace select qa

**2. Apply the configuration:** terraform apply -var-file=qa.tfvars

**- For `uat` environment:**

**1. Select the `uat` workspace:** terraform workspace select uat

**2. Apply the configuration:** terraform apply -var-file=uat.tfvars

This way, Terraform will use the variables defined in the specified `.tfvars` file for the current workspace. Please replace `dev.tfvars`, `qa.tfvars`, and `uat.tfvars` with your actual `.tfvars` file paths if they're not in the same directory as your Terraform configuration files. Also, ensure you're in the correct directory where your Terraform configuration files are located before running these commands. Remember, most Terraform commands only interact with the currently selected workspace. This includes provisioning and state manipulation. So, be careful while selecting workspaces.