terraform validate   ##terraform aplly dememeden once denir ve main.tf dosyasinda hatalari gosterir

- Go to the terminal and run `terraform validate`. It validates the Terraform files syntactically correct and internally consistent.

terraform fmt   ##validate e benzer main.tf dosyasinda bosluk vb hatalri duzeltir cok onemli hatalr degildsir sistemin calismasina engel degildir

**### terraform console**

- Go to the terminal and run `terraform console`.This command provides an interactive command-line console for evaluating and experimenting with expressions. This is useful for testing interpolations before using them in configurations, and for interacting with any values currently saved in state. You can see the attributes of resources in tfstate file and check built in functions before you write in your configuration file.

- Lets create a file under the terraform-aws directory and name it `cloud` and paste `hello devops engineers`.

```bash

echo "hello devops" > cloud   #terraform-aws klasoru altinda bu cloud adli #dosya olusuturuz bu dosyayi terraform console komutunu calisitridigimizda #cagirmak icin kullandirk ornek olarak

```

Run the following commands.

```bash

terraform console   ##terraformda yeni bir consola acmaya yarar #asagidaki #komutlari terraform console dedikten sonra yeni bir consaole acildiktan sonra #maintf icindeki kaynaklari tek tek cagirmak icin kullanilir

> aws\_instance.tf-ec2

> aws\_instance.tf-ec2.private\_ip

> min (1,2,3)

> lower("HELLO")

> file("${path.module}/cloud")

> aws\_s3\_bucket.tf-s3

> aws\_s3\_bucket.tf-s3.bucket

> exit or (ctrl+c)

```

terraform show   ##bu komut terraform.tfstate dosyasindaki bilgiler gibi ##bilgiler maintf icindekileri ayrintili gosterir

**### graph command.**

- Go to the terminal and run `terraform graph`. It creates a visual graph of Terraform resources. The output of "terraform graph" command is in the DOT format, which can easily be converted to an image by making use of dot provided by GraphViz.

- Copy the output and paste it to the `https://dreampuf.github.io/GraphvizOnline`. Then display it. If you want to display this output in your local, you can download graphviz (`sudo yum install graphviz`) and take a `graph.svg` with the command `terraform graph | dot -Tsvg > graph.svg`.

```bash

terraform graph  ##maintf kaynaklarini bu komut girdiktan sonra cikan cevabi ##ctrlc ile alip graph sitesinde ctlv  dedigimizde maintf deki kaynakalrin ##algoritma diyagramii gosterir

**### output command. ##cikti komutu**

- Output values make information about your infrastructure available on the command line, and can expose information for other Terraform configurations to use.

- Now add the followings to the `main.tf` file.  Then run the commands `terraform apply or terraform refresh` and `terraform output`. `terraform output` command is used for reading an output from a state file. It reads an output variable from a Terraform state file and prints the value. With no additional arguments, output will display all the outputs for the (parent) root module.  If NAME is not specified, all outputs are printed.

```go   ###aws

output "tf\_example\_public\_ip" {

  value = aws\_instance.tf-ec2.public\_ip

}

output "tf\_example\_s3\_meta" {

  value = aws\_s3\_bucket.tf-s3.region

}

```

```bash

terraform apply

terraform output  ##maintf daki kaynaklardan sonucunu gormekistegimiz kaynagi ##yazariz

terraform output -json

terraform output tf\_example\_public\_ip

**### terraform apply -refresh-only command.**

- The `terraform apply -refresh-only` command is used to update the state file with the real-world infrastructure. This can be used to detect any drift from the last-known state, and to update the state file. First, check the current state of your resources with `terraform state list`. Then go to the AWS console and delete your S3 bucket `sam-tf-test-bucket-addwhateveryouwant`. Display the state list again and refresh the state. Run the following commands.

```bash

$ terraform state list  #3var olan main.tf deki kaynaklari  listeler

aws\_instance.tf-example-ec2  ##tf-example-ec2 adinda kaynak var

aws\_s3\_bucket.tf-example-s3   ##tf-example-s3 adinda s3 bucket var

$ terraform apply -refresh-only  ##bu komut terraform main.tf dosyasinda girdigimiz kaynak listesi guncler cunku eger birisi sizin haberiniz olmadan aws kayanklarindan birini silerse aws deki bu durumu terraform makinesi gormez o sadece kendi main.tf dosyasindaki gorur.hatta bu gormek icin terraform state list dedigimizde hala eski kaynaklarin var olarak listeler bize .bunun icin refresh komutunu girdigimizde main.tf de sanki o kaynakta silinmis gibi terraform state list komutunu girince kontrol etmek icinaws deki olusan kaynaklari gormek icin bu sefer silinen kaynagin silinmis oldugunu goruruz listete o kaynak yoktur.

$ terraform state list  ##yeni durumdaki hali burda gosteriryo refresh sonrasi

aws\_instance.tf-example-ec2

```

- Now, you can see the differences between files `terraform.tfstate` and `terraform.tfstate.backup`. From tfstate file S3 bucket is deleted but in backup file you can see the S3 bucket.

- Run terraform apply -auto-approve and create S3 bucket again.

```bash

 terraform apply -refresh=false  ##refresh yapma demektir

- Terraform loads variables in the following order:

  - Any -var and -var-file options on the command line, in the order they are provided.

  - Any \*.auto.tfvars or \*.auto.tfvars.json files, processed in lexical order of their filenames.

  - The terraform.tfvars.json file, if present.

  - The terraform.tfvars file, if present.

  - Environment variables

-komut satırında -var kullanıldıysa önce onu dikkate alır. yoksa auto.tfvars ya da auto.tfvars.json dosyalarını dikkate alır.

onlar yoksa terraform.tfvars.json dikkate alınır. O yoksa terraform.tfvars dikkate alınır. Bunlardan sonra da environment variables dikkate alınır.