📦 Ansible Custom Inventory – Explained

An **Ansible inventory** is a list of hosts (servers, nodes, devices) that Ansible manages. A **custom inventory** allows you to define your infrastructure beyond the default `hosts` file, using your own file (INI, YAML, or dynamic script).

✅ Types of Custom Inventory

1. Static Inventory (INI Format)

A traditional inventory file using INI syntax.

Example: `inventory.ini`

```
[webservers]
web1.example.com
web2.example.com

[dbservers]
db1.example.com ansible_user=admin ansible_port=2222
```

2. Static Inventory (YAML Format)

YAML is more structured and easier to read for complex groups.

Example: `inventory.yml`

```
all:
  children:
    webservers:
      hosts:
        web1.example.com:
        web2.example.com:
    dbservers:
      hosts:
        db1.example.com:
          ansible_user: admin
          ansible_port: 2222
```

🔁 Dynamic Inventory

For cloud or dynamic environments, inventory can be generated on-the-fly via a script or plugin.

Example: AWS EC2 Dynamic Inventory

ansible-inventory -i ec2.py --list

Or use official dynamic inventory plugins like:

[defaults]
inventory = plugins/aws_ec2.yaml

🔧 Using a Custom Inventory File

When running Ansible commands or playbooks, specify your custom inventory file:

ansible all -i inventory.ini -m ping
ansible-playbook -i inventory.yml site.yml

Or define it in your custom `ansible.cfg`:

[defaults]
inventory = ./my_inventory.ini

🧠 Inventory Variables You Can Use

You can assign **host-level**, **group-level**, or **global** variables.

Example in INI:

[web]
web1 ansible_host=192.168.1.10 ansible_user=ubuntu

[web:vars]
ansible_ssh_private_key_file=~/.ssh/web.pem

🧪 Summary

Type	Format	Use Case
Static	INI	Simple environments
Static	YAML	Structured, readable inventories
Dynamic	Script	Cloud-based, auto-discovery
