Ping All Hosts (Basic Connectivity),

```
- hosts: all
  tasks:
    - name: Check connectivity
      ping:
```

Install Package (Idempotency),

```
- hosts: web
  become: yes
  tasks:
    - name: Install nginx
      yum:
        name: nginx
        state: present
```

Start and Enable Service,

```
- hosts: web
  become: yes
  tasks:
    - name: Start nginx
      service:
        name: nginx
        state: started
        enabled: yes
```

Copy File,

```
- hosts: web
  tasks:
    - name: Copy config file
      copy:
        src: app.conf
        dest: /tmp/app.conf
```

Using Variables,

```
- hosts: localhost
  vars:
    message: "Hello Huidrom"
  tasks:
    - debug:
        msg: "{{ message }}"
```

Conditional Execution (when),

```
- hosts: all
  tasks:
    - name: Run only on RedHat
      debug:
        msg: "This is RHEL"
      when: ansible_os_family == "RedHat"
```

Loop Example,

```
- hosts: localhost
  tasks:
    - name: Install multiple packages
      yum:
        name: "{{ item }}"
        state: present
      loop:
        - git
        - curl
        - wget
```

Register Output,

```
- hosts: localhost
  tasks:
    - name: Get uptime
      command: uptime
      register: result

    - debug:
        var: result.stdout
```

changed_when,

```
- hosts: localhost
  tasks:
    - name: Custom change condition
      command: echo "Hello"
      register: output
      changed_when: false
```

failed_when,

```
- hosts: localhost
  tasks:
    - name: Fail if file missing
      command: ls /tmp/testfile
      register: result
      failed_when: result.rc != 0
```

Block / Rescue / Always,

```
- hosts: localhost
  tasks:
    - block:
        - command: ls /wrongpath
      rescue:
        - debug:
            msg: "Recovered from failure"
      always:
        - debug:
            msg: "Always runs"
```

Handler Example

```
- hosts: web
  tasks:
    - name: Update config
      copy:
        src: app.conf
        dest: /etc/app.conf
      notify: restart app

  handlers:
    - name: restart app
      service:
        name: nginx
        state: restarted
```

run_once

```
- hosts: web
  tasks:
    - name: Run only once
      command: echo "Database migration"
      run_once: true
```

delegate_to

```
- hosts: web
  tasks:
    - name: Backup DB from controller
      command: pg_dump mydb
      delegate_to: db01
```

Retry Until

```
- hosts: localhost
  tasks:
    - name: Wait for service
      command: curl http://localhost
      register: result
      retries: 5
      delay: 5
      until: result.rc == 0
```

Using Template (Jinja2)

```
- hosts: web
  tasks:
    - name: Deploy config template
      template:
        src: app.conf.j2
        dest: /etc/app.conf
```

Serial (Rolling Update)

```
- hosts: web
  serial: 2
  tasks:
    - name: Restart service gradually
      service:
        name: nginx
        state: restarted
```

connection: local

```
- hosts: localhost
  connection: local
  tasks:
    - name: Run locally
      command: echo "Running on control node"
```

Include Tasks (Modularization)

main.yml

```
- hosts: localhost
  tasks:
    - include_tasks: tasks.yml
```

tasks.yml

```
- debug:
    msg: "Included task file"
```

corporealsoul
corporealsoul
corporealsoul.github.io

1. The Basic Playbook (Structure)

```
- name: Hello World
  hosts: all
  tasks:
    - name: Print a message
      debug:
        msg: "Hello Ansible"
```

2. Variables (Vars)

```
- name: Variable usage
  hosts: all
  vars:
    pkg_name: vim
  tasks:
    - name: Show variable
      debug:
        msg: "Target package is {{ pkg_name }}"
```

3. Inventory Variables (Host/Group Vars)

```
- name: Inventory data
  hosts: webservers
  tasks:
    - name: Access group variable
      debug:
        msg: "Connecting to {{ inventory_hostname }}"
```

4. Privilege Escalation (Become)

```
- name: Root task
  hosts: all
  become: yes
  tasks:
    - name: Install Nginx
      apt:
        name: nginx
        state: present
```

5. Idempotency (State Management)

```
- name: Ensure directory exists
  hosts: all
  tasks:
    - name: Create folder
      file:
        path: /tmp/ansible_test
        state: directory
```

6. Handlers (Event-Driven)

```
- name: Handler demo
  hosts: all
  tasks:
    - name: Change config
      copy: src=test.conf dest=/etc/test.conf
      notify: Restart Service
  handlers:
    - name: Restart Service
      service: name=ssh state=restarted
```

7. Loops (Iterating)

```
- name: Loop demo
  hosts: all
  tasks:
    - name: Create users
      user: name={{ item }} state=present
      loop: ['alice', 'bob', 'charlie']
```

8. Conditionals (When)

```
- name: Conditional task
  hosts: all
  tasks:
    - name: Shut down Debian only
      command: /sbin/shutdown -h now
      when: ansible_os_family == "Debian"
```

9. Gathering Facts (Setup)

```
- name: Use Facts
  hosts: all
  tasks:
    - name: Show OS
      debug:
        msg: "OS is {{ ansible_distribution }}"
```

10. Register (Capturing Output)

```
- name: Register demo
  hosts: all
  tasks:
    - name: Check uptime
      shell: uptime
      register: uptime_result
    - name: Print result
      debug:
        var: uptime_result.stdout
```

11. Templates (Jinja2)

```
- name: Template demo
  hosts: all
  tasks:
    - name: Deploy config
      template:
        src: config.j2
        dest: /etc/myconfig.conf
```

12. Error Handling (Ignore Errors)

```
- name: Error demo
  hosts: all
  tasks:
    - name: Run risky command
      command: /bin/false
      ignore_errors: yes
```

13. Blocks (Grouping Tasks)

```
- name: Block demo
  hosts: all
  tasks:
    - block:
        - name: Task 1
          command: uptime
      rescue:
        - name: Recovery task
          debug: msg="Task 1 failed!"
```

14. Local Actions (Delegate To)

```
- name: Delegate demo
  hosts: webservers
  tasks:
    - name: Record deployment locally
      shell: echo "Deployed to {{ inventory_hostname }}" >> log.txt
      delegate_to: localhost
```

15. Tags (Selective Execution)

```
- name: Tag demo
  hosts: all
  tasks:
    - name: Setup DB
      debug: msg="Database setup"
      tags: database
```

16. Roles (Reusability)

```
- name: Role demo
  hosts: webservers
  roles:
    - common
    - webserver
```

17. Vault (Encryption)

```
- name: Vault demo
  hosts: all
  vars_files:
    - secrets.yml
  tasks:
    - name: Use secret
      debug: var=db_password
```

18. Prompting (Interactivity)

```
- name: Prompt demo
  hosts: all
  vars_prompt:
    - name: username
      prompt: "Enter username"
  tasks:
    - name: Greet user
      debug: msg="Hello {{ username }}"
```

19. Wait For (Orchestration)

```
- name: Wait demo
  hosts: all
  tasks:
    - name: Wait for port 80
      wait_for:
        port: 80
        state: started
```

20. Stat (File Checking)

```
    - name: Stat demo
      hosts: all
      tasks:
        - name: Check if file exists
          stat:
            path: /etc/passwd
          register: p
        - name: Report
          debug: msg="It exists!"
          when: p.stat.exists
```

The "All-in-One" Ansible Master Playbook - 01

```
---
# 1. PLAY HEADER: Defines scope and global settings
- name: Master Playbook Demonstrating All Core Concepts
  hosts: webservers
  become: yes                    # PRIVILEGE ESCALATION (sudo)
  gather_facts: yes              # GATHERING FACTS (setup module)
  serial: 2                      # ROLLING UPDATE (Orchestration)

  # 2. VARIABLES: Static and Prompted
  vars:
    app_port: 8080               # STATIC VAR
    doc_root: "/var/www/master"

  vars_prompt:                   # INTERACTIVE PROMPT
    - name: "user_password"
      prompt: "Enter the secret password"
      private: yes

  vars_files:                    # EXTERNAL VARS (can be Vault-encrypted)
    - secrets.yml

  # 3. PRE_TASKS: Runs before any roles or main tasks
  pre_tasks:
    - name: Check Connectivity
      ping:

  # 4. ROLES: Reusable automation units
  roles:
    - common_setup
```

corporealsoul
corporealsoul
corporealsoul.github.io

```
# 5. TASKS: The core logic
tasks:
  - name: 1. BLOCK: Grouping tasks with Error Handling
    block:
      - name: Install Nginx (Idempotency)
        package:
          name: nginx
          state: present
        register: nginx_install   # REGISTER: Captures output for later use

      - name: Conditional Task (When)
        debug:
          msg: "System is Debian-based"
        when: ansible_os_family == "Debian"

    rescue:                       # ERROR HANDLING: Runs if block fails
      - name: Rescue failed installation
        debug:
          msg: "Installation failed, rolling back..."

  - name: 2. LOOPS: Creating multiple directories
    file:
      path: "{{ item }}"
      state: directory
      mode: '0755'
    loop:                         # LOOPING
      - "/var/www/assets"
      - "/var/www/logs"

  - name: 3. TEMPLATING: Deploying a Jinja2 config
    template:
      src: server.conf.j2
      dest: /etc/nginx/conf.d/server.conf
    notify: Restart Nginx         # HANDLERS: Triggered by change

  - name: 4. DELEGATION: Record deployment on Control Node
    shell: "echo 'Deployed to {{ inventory_hostname }}' >> /tmp/deploy.log"
    delegate_to: localhost        # LOCAL ACTION / DELEGATION
```

```
    - name: 5. ASYNC: Run a long-running background task
      command: /opt/scripts/heavy_task.sh
      async: 300                    # Run for up to 5 mins
      poll: 0                       # Don't wait for it (fire and forget)

    - name: 6. TAGS: A task that only runs if requested
      debug:
        msg: "This is a maintenance task"
      tags: [ never, maintenance ]  # TAGS: Use --tags "maintenance"

  # 6. HANDLERS: Event-driven tasks
  handlers:
    - name: Restart Nginx
      service:
        name: nginx
        state: restarted

  # 7. POST_TASKS: Runs after everything else
  post_tasks:
    - name: Verify Service Port
      wait_for:                     # ORCHESTRATION: Wait for logic
        port: "{{ app_port }}"
        state: started
        timeout: 30
---
```

The "All-in-One" Ansible Master Playbook - 02

inventory.ini

```
[web]
web1 ansible_host=192.168.1.10
web2 ansible_host=192.168.1.11

[db]
db1 ansible_host=192.168.1.20
```

vars/secrets.yml (Example Vault File)

```
db_password: supersecret
```

deploy.yml (MAIN PLAYBOOK)

```
---
- name: Complete Production Deployment
  hosts: web
  become: yes
  serial: 1
  vars:
    app_name: myapp
    app_port: 8080
  vars_files:
    - vars/secrets.yml

  pre_tasks:
    - name: Gather minimal facts
      setup:
        gather_subset:
          - network
```

```
tasks:

  - name: Install required packages
    yum:
      name: "{{ item }}"
      state: present
    loop:
      - git
      - nginx
    tags: install

  - name: Copy application config
    template:
      src: app.conf.j2
      dest: /etc/nginx/conf.d/{{ app_name }}.conf
    notify: restart nginx
    tags: config

  - name: Check if app directory exists
    stat:
      path: /var/www/{{ app_name }}
    register: app_dir

  - name: Clone application repository
    git:
      repo: "https://github.com/example/app.git"
      dest: /var/www/{{ app_name }}
    when: not app_dir.stat.exists
    tags: deploy

  - name: Run database migration (run once)
    command: /var/www/{{ app_name }}/migrate.sh
    run_once: true
    delegate_to: db1
    register: migration
    changed_when: "'No changes' not in migration.stdout"
    failed_when: migration.rc != 0
```

```
  - name: Wait for application port
    wait_for:
      port: "{{ app_port }}"
      timeout: 30
    register: wait_result
    retries: 5
    delay: 5
    until: wait_result is succeeded

  - name: Long running background task
    command: sleep 60
    async: 120
    poll: 10

  - name: Example block with error handling
    block:
      - name: Try risky operation
        command: /bin/false

    rescue:
      - name: Handle failure
        debug:
          msg: "Operation failed, handled in rescue."

    always:
      - name: Always run cleanup
        debug:
          msg: "Cleanup complete."

handlers:
  - name: restart nginx
    service:
      name: nginx
      state: restarted
```