

09 Filtros

Tipos de filtros

Filtros en Ansible

Los *filtros* en Ansible Playbook son herramientas que permiten procesar, transformar, extraer o manipular datos durante la ejecución de tareas en los playbooks, facilitando la gestión avanzada de variables y resultados. Se inspiran en la funcionalidad de los filtros de Jinja2 y se aplican usando el símbolo de tubería (| pipe), lo que indica que el dato recibido será transformado por el filtro especificado. En ansible destacamos los siguientes filtros:

- **default**: Proporciona un valor por defecto si la variable no está definida.
 - {{ usuario | default('admin') }}
- **int, float, string**: Realiza conversión de tipos de datos.
 - {{ puerto | int }}
- **length**: Devuelve la cantidad de elementos de una lista.
 - {{ items | length }}
- **replace**: Sustituye una cadena por otra dentro de una variable.
 - {{ cadena | replace(' ', '_') }}
- **split**: Divide una cadena en una lista usando un delimitador.
 - {{ texto | split(',') }}
- **join**: Une los elementos de una lista en una cadena con un separador.
 - {{ lista | join(';') }}
- **sort**: Ordena los elementos de una lista.
 - {{ lista | sort }}
- **unique**: Elimina elementos duplicados de una lista.
 - {{ lista | unique }}
- **map**: Aplica una función a cada elemento de una lista.
 - {{ lista | map('upper') | list }}
- **min, max**: Devuelve el valor mínimo o máximo de una lista.
 - {{ lista | min }} o {{ lista | max }}
- **regex_search, regex_replace**: Busca o sustituye patrones mediante expresiones regulares.
 - {{ cadena | regex_search('patrón') }}
- **dict2items** y **items2dict**: Convierte un diccionario en una lista de items y viceversa.
 - {{ mi_diccionario | dict2items }}
- **type_debug**: Muestra el tipo de dato de una variable.
 - {{ variable | type_debug }}
- **shuffle**: Mezcla aleatoriamente los elementos de una lista.
 - {{ lista | shuffle }}
- **hash**: Calcula el hash de una cadena (por ejemplo SHA1).
 - {{ password | hash('sha1') }}
- **ternary**: Devuelve diferentes valores para verdadero, falso o nulo.
 - {{ valor | ternary('sí', 'no', 'nulo') }}
- **permutations** y **combinations**: Genera permutaciones o combinaciones de una lista.
 - {{ [1,2,3] | permutations(2) }}

```
---
- name: Ejemplos de filtros
  hosts: srv1
  vars:
    cadena: "Soy una cadena"
    numero: 10
    verdad: true
    lista:
      - Martín
      - Alex
      - Carlos
    lista1: ['Martín','Alex','Carlos']
    diccionario:
      nombre: Alex
      edad: 27

  tasks:
    - name: Averiguar el tipo de una variable
      ansible.builtin.debug:
        var: lista | type_debug
```

Conversión

Permite convertir de un tipo a otro. Puedes convertir una cadena en un entero con `| int`, en booleano con `| bool`, o en una lista con `| split(',')`, facilitando así la adaptación de variables al tipo correcto para cada operación.

```
---
- name: Ejemplos de filtros
  hosts: srv1
  vars:
    cadena: "Soy una cadena"
    cadena: "100"
    numero: 10
    verdad: true
    lista:
      - Martín
      - Alex
      - Carlos
    lista1: ['Martín','Alex','Carlos']
    diccionario:
      nombre: Alex
      edad: 27
    debug_enabled: no

  tasks:
    - name: Muestra la variable como patilla jinja2
      ansible.builtin.debug:
        msg: 'El número es: {{ numero }}'
```

```

- name: Muestra la variable
  ansible.builtin.debug:
    var: numero

- name: Convertir a cadena
  ansible.builtin.debug:
    var: numero | string

- name: Convertir a entero
  ansible.builtin.debug:
    var: cadena | int

- name: Convertir a entero 2
  ansible.builtin.debug:
    var: cadena2 | int

- name: Convertir a entero
  ansible.builtin.debug:
    var: cadena | bool

- name: Imprimir todas las Ansible Facts
  ansible.builtin.debug:
    var: ansible_facts
  when: debug_enabled

- name: Visualizar las máquinas con una versión de distribución superior a la 10
  ansible.builtin.debug:
    msg: "{{ansible_facts.distribution_version}}"
  when: ansible_distribution_version | int > 10

```

Nota: No hay ninguna diferencia real entre **ansible_facts.distribution_version** y **ansible_distribution_version**; ambos hacen referencia al mismo valor y son intercambiables en la práctica. La variable **ansible_distribution_version** es un alias o acceso directo a la clave correspondiente dentro de **ansible_facts**, usada para facilitar la escritura en los playbooks.

Cadenas

Las cadenas de texto (strings) se usan para representar datos de texto y se manejan dentro de tareas, variables y plantillas con la sintaxis de YAML y Jinja2. Como es lógico a estas cadenas se les pueden aplicar sus correspondientes filtros.

```

---
- name: Ejemplos varios de filtros con CADENAS
  hosts: srv1
  vars:
    cadena: "Esto es una cadena"

  tasks:
    - name: Mayúsculas
      ansible.builtin.debug:
        var: cadena | upper

```

```
- name: Minúsculas
  ansible.builtin.debug:
    var: cadena | lower

- name: Reemplazar
  ansible.builtin.debug:
    var: cadena | replace("e","*")

- name: Reemplazar con mensaje
  ansible.builtin.debug:
    msg: 'La cadena reemplazada es: {{cadena | lower | replace("e","*")}}'

- name: Longitud de cadena
  ansible.builtin.debug:
    var: cadena | length
```

Números

En Ansible podemos trabajar con múltiples operaciones sobre número como calcular la potencia, redondear, calcular su raíz cuadrada, calcular un número aleatorio, entre otras.

```
---
- name: Ejemplos varios de filtros con NUMEROS
  hosts: srv1
  vars:
    numero: 10.40

  tasks:
    - name: Potencia
      ansible.builtin.debug:
        var: numero | pow(2)

    - name: Raiz cuadrada
      ansible.builtin.debug:
        var: numero | root()

    - name: Redondeo
      ansible.builtin.debug:
        var: numero | round()

    - name: Número aleatorio desde cero hasta el número indicado
      ansible.builtin.debug:
        var: numero | int | random
```

Listas

Tenemos diferentes formas de trabajar con listas pudiendo juntarlas, convertir una cadena a lista, tener el número menor y mayor o la palabra menor o mayor (en orden alfabético).

```
---
- name: Ejemplos varios de filtros con LISTAS
  hosts: srv1
  vars:
    lista_numero:
      - 2
      - 10
      - 9
      - 1
    lista_cadena:
      - Pedro
      - Zhais
      - Charo
      - Antonio
    cadena: "Esto es una cadena"

  tasks:
    - name: Valor menor numero
      ansible.builtin.debug:
        msg: "{{ lista_numero | min }} --- {{ lista_numero | max }}"

    - name: Valor menor cadena
      ansible.builtin.debug:
        msg: "{{ lista_cadena | min }} --- {{ lista_cadena | max }}"

    - name: Unir elementos de una lista
      ansible.builtin.debug:
        var: lista_cadena | join(',')

    - name: Convertir cadena en lista
      ansible.builtin.debug:
        var: cadena | split()
```

Ejercicios para repasar los conceptos anteriores

Se propone la [Tarea 5](#) para repasar los conceptos vistos hasta este punto.