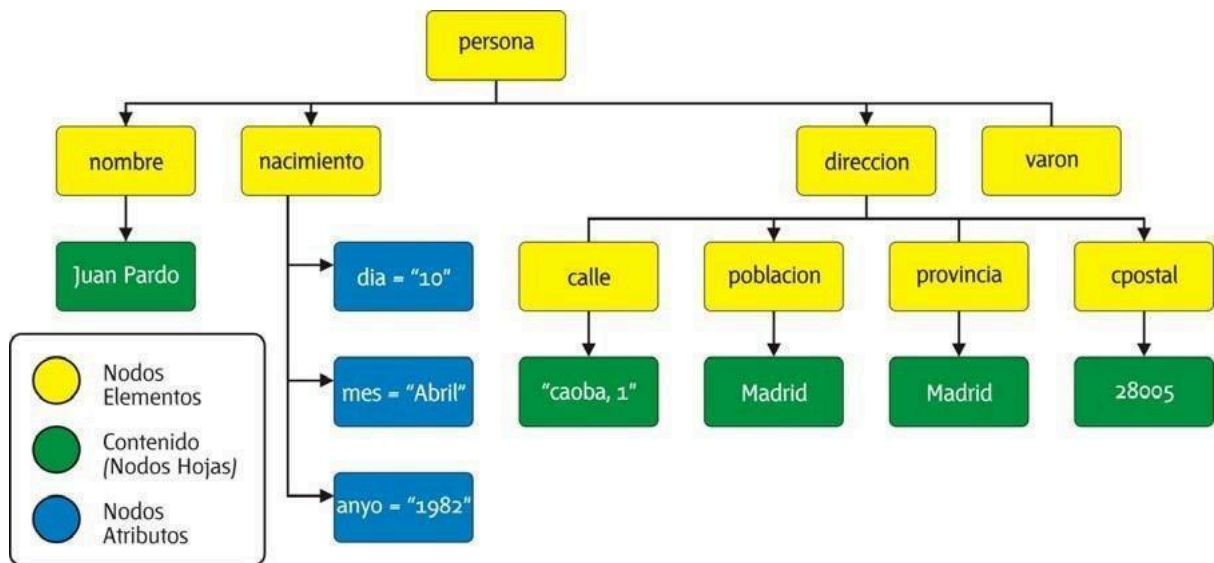


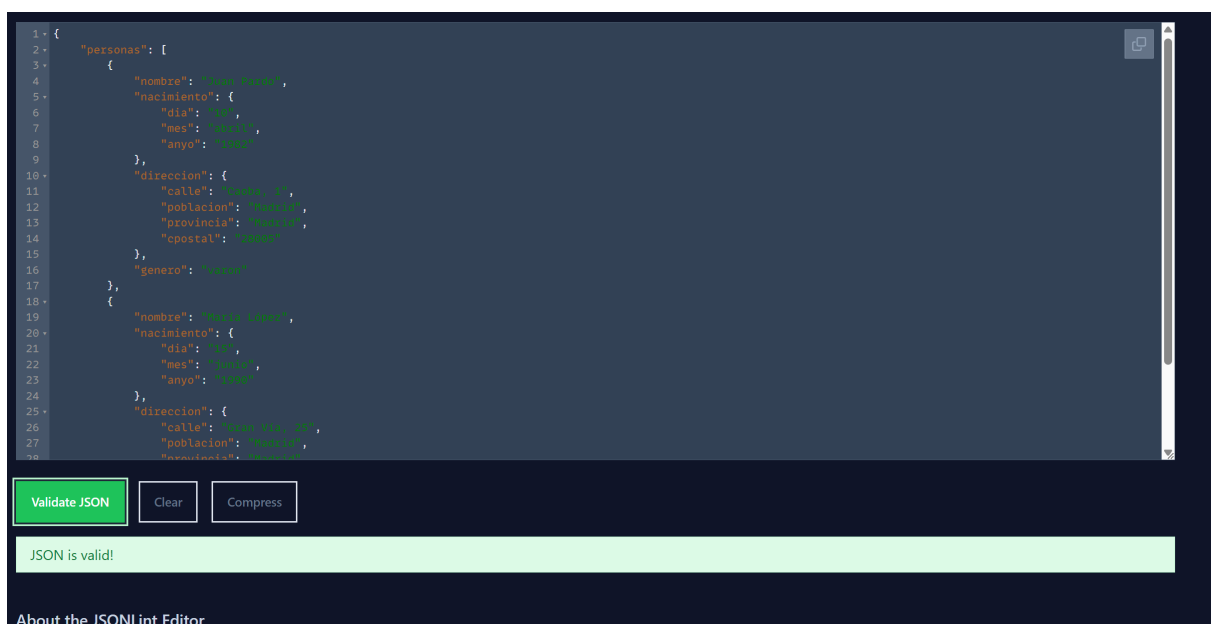
### Ejercicio 1

Hacer una JSON que pueda contener una lista de personas, cada persona está formada por los siguientes elementos. **Hacer un ejemplo con 2 personas**



```
{  
  "personas": [  
    {  
      "nombre": "Juan Pardo",  
      "nacimiento": {  
        "dia": "10",  
        "mes": "abril",  
        "año": "1982"  
      },  
      "direccion": {  
        "calle": "Caoba, 1",  
        "poblacion": "Madrid",  
        "provincia": "Madrid",  
        "cpostal": "28005"  
      },  
      "genero": "varon"  
    },  
  ],  
}
```

```
{  
  
  "nombre": "María López",  
  
  "nacimiento": {  
  
    "dia": "15",  
  
    "mes": "junio",  
  
    "anyo": "1990"  
  
  },  
  
  "direccion": {  
  
    "calle": "Gran Vía, 25",  
  
    "poblacion": "Madrid",  
  
    "provincia": "Madrid",  
  
    "cpostal": "28013"  
  
  },  
  
  "genero": "mujer"  
  
}  
  
]  
  
}
```



The screenshot displays the JSONLint Editor interface. The main text area contains a JSON array with two objects. The first object has properties: "nombre" (Juan Pardo), "nacimiento" (day: 08, mes: Abril, anyo: 1990), "direccion" (calle: Gran Vía, poblacion: Madrid, provincia: Madrid, cpostal: 28001), and "genero" (hombre). The second object has properties: "nombre" (María López), "nacimiento" (day: 15, mes: junio, anyo: 1990), "direccion" (calle: Gran Vía, 25, poblacion: Madrid, provincia: Madrid, cpostal: 28013), and "genero" (mujer). Below the text area are three buttons: "Validate JSON" (highlighted in green), "Clear", and "Compress". A green message bar at the bottom states "JSON is valid!". At the very bottom, there is a link "About the JSONLint Editor".

```
1 {  
2   "personas": [  
3     {  
4       "nombre": "Juan Pardo",  
5       "nacimiento": {  
6         "dia": "08",  
7         "mes": "Abril",  
8         "anyo": "1990"  
9       },  
10      "direccion": {  
11        "calle": "Gran Vía",  
12        "poblacion": "Madrid",  
13        "provincia": "Madrid",  
14        "cpostal": "28001"  
15      },  
16      "genero": "hombre"  
17    },  
18    {  
19      "nombre": "María López",  
20      "nacimiento": {  
21        "dia": "15",  
22        "mes": "junio",  
23        "anyo": "1990"  
24      },  
25      "direccion": {  
26        "calle": "Gran Vía, 25",  
27        "poblacion": "Madrid",  
28        "provincia": "Madrid",  
29        "cpostal": "28013"  
30      },  
31      "genero": "mujer"  
32    }  
33  ]  
34 }  
35
```

Validate JSON Clear Compress

JSON is valid!

[About the JSONLint Editor](#)

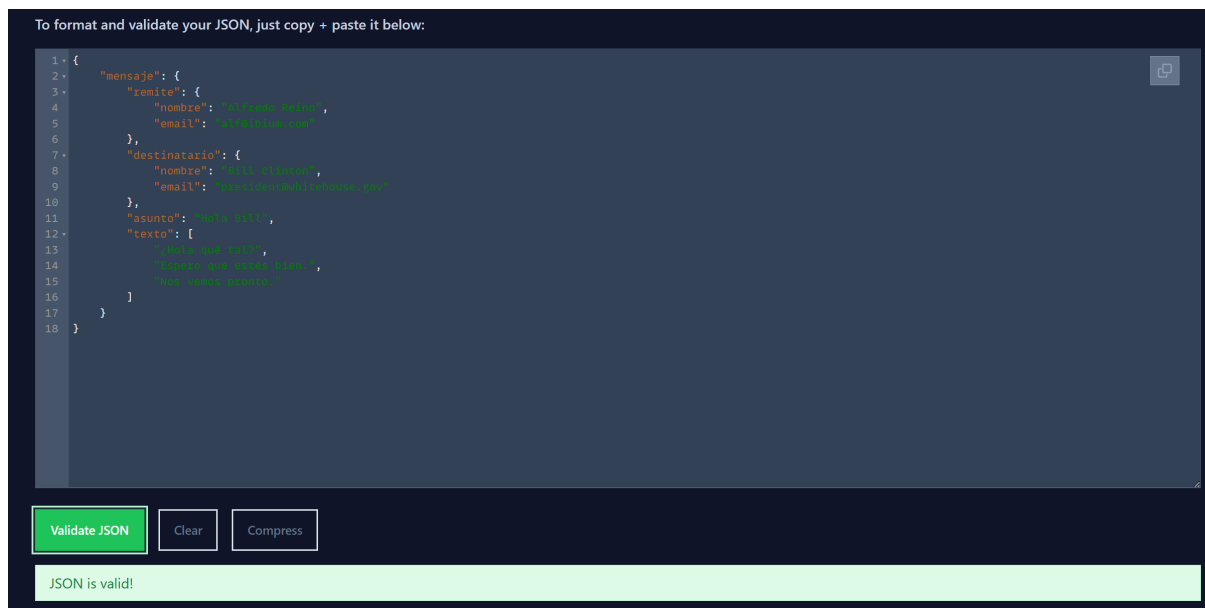
## Ejercicio 2

Una aplicación de mensajería quiere contener cada mensaje en un JSON. Cada mensaje tiene estos campos



Ha un JSON con un mensaje de ejemplo

```
{
  "mensaje": {
    "remite": {
      "nombre": "Alfredo Reino",
      "email": "alf@ibium.com"
    },
    "destinatario": {
      "nombre": "Bill Clinton",
      "email": "president@whitehouse.gov"
    },
    "asunto": "Hola Bill",
    "texto": [
      "¿Hola qué tal?",
      "Espero que estés bien.",
      "Nos vemos pronto."
    ]
  }
}
```



### Ejercicio 3 ( Pedidos )

Genera el JSON necesario para una app de compras que debe enviar la información de **cada pedido** al servidor en un archivo JSON. Para cada pedido


se debe generar un archivo JSON que contendrá:


- Un número de pedido, la fecha ( AAAA-MM-DD ) y la hora del pedido ( hh:mm:ss )
- Un elemento usuario que contendrá, el nombre de usuario, el nombre completo y la dirección de envío.
- La dirección de envío a su vez es un elemento que está compuesto por la calle, la población y el código postal.
- A continuación irá una lista de artículos. Para cada artículo se incluirá el código de artículo, una descripción y el precio.

Haz un ejemplo de archivo para **un pedido de tres artículos**

```
{
  "pedido": {
    "numeroPedido": "A12345",
    "fecha": "2025-05-05",
    "hora": "16:45:30",
    "usuario": {
      "nombreUsuario": "juan_garcia",
      "nombreCompleto": "Juan García López",
      "direccionEnvio": {
        "calle": "Calle del Río",
```

```
    "poblacion": "Toledo",
    "codigoPostal": "45001"
  },
  "articulos": [
    {
      "codigoArticulo": "P001",
      "descripcion": "Aceite de oliva virgen extra 1L",
      "precio": 6.50
    },
    {
      "codigoArticulo": "P002",
      "descripcion": "Vinagre balsámico 250ml",
      "precio": 3.75
    },
    {
      "codigoArticulo": "P003",
      "descripcion": "Pack de especias surtidas",
      "precio": 4.20
    }
  ]
}
```


**JSONLint**  
 Validator and Formatter



Mejoramos las tarifas  
SIMYO

[Learn more](#)

To format and validate your JSON, just copy + paste it below:

```

1 {
2   "pedido": {
3     "numeroPedido": "012345",
4     "fecha": "2023-05-05",
5     "hora": "15:45:30",
6     "usuario": {
7       "nombreUsuario": "Juan Garcia",
8       "nombreCompleto": "Juan Garcia Lopez",
9       "direccionEnvio": {
10        "calle": "Calle del Rio",
11        "poblacion": "Madrid",
12        "codigoPostal": "28001"
13      }
14    },
15    "articulos": [
16      {
17        "codigoArticulo": "0001",
18        "descripcion": "Botella de agua mineral 1L",
19        "precio": 1.5
20      },
21      {
22        "codigoArticulo": "0002",
23        "descripcion": "Panqueques deshidratados 250g",
24        "precio": 2.5
25      },
26      {
27        "codigoArticulo": "0003",
28        "descripcion": "Barra de chocolate 100g"
  
```

Validate JSON
 Clear
 Compress

JSON is valid!

#### Ejercicio 4 ( Concesionarios)

Se desea modelar en JSON la información relativa a distintos concesionarios de vehículos de una marca. La lista de concesionarios debe tener al menos uno, no puede estar vacía. Deberá poder recoger la información de contacto de cada concesionario ( nombre, dirección, localidad y uno o más teléfonos ) así como la lista de los vehículos que tiene a la venta. La lista de vehículos puede estar vacía, o contener 1 o varios vehículos diferentes. Cada vehículo se identifica

mediante su matrícula y pueden ser del tipo “turismo” o “furgoneta”, siendo el

valor por defecto “turismo”. Del vehículo se recoge el modelo, cilindrada, tipo de combustible ( sólo puede ser Gasolina

o Diesel ), color, y puede contener un elemento “asegurado” que es un booleano

**Hacer un ejemplo con 2 concesionarios con 2 y 3 vehículos respectivamente**

```

{
  "concesionarios": [
    {
      "nombre": "AutoMadrid",
      "direccion": "Calle Alcalá 123",
      "localidad": "Madrid",
      "telefonos": ["911234567", "919876543"],
      "vehiculos": [
        {
          "matricula": "1234ABC",
  
```

```
"tipo": "turismo",
"modelo": "Seat Ibiza",
"cilindrada": 1200,
"combustible": "Gasolina",
"color": "Rojo",
"asegurado": true
},
{
  "matricula": "5678DEF",
  "tipo": "furgoneta",
  "modelo": "Ford Transit",
  "cilindrada": 2000,
  "combustible": "Diesel",
  "color": "Blanco"
}
],
{
  "nombre": "MotorSur",
  "direccion": "Av. Andalucía 56",
  "localidad": "Sevilla",
  "telefonos": ["954112233"],
  "vehiculos": [
    {
      "matricula": "4321GHI",
      "modelo": "Renault Clio",
      "cilindrada": 1100,
      "combustible": "Gasolina",
      "color": "Negro"
    },
    {
      "matricula": "8765JKL",
      "tipo": "furgoneta",
      "modelo": "Citroën Jumpy",
      "cilindrada": 1900,
      "combustible": "Diesel",
```

```

        "color": "Gris",
        "asegurado": false
    },
    {
        "matricula": "1122MNO",
        "modelo": "Peugeot 208",
        "cilindrada": 1300,
        "combustible": "Gasolina",
        "color": "Azul"
    }
]
}
]
}

```



### Ejemplo 5 ( Videoclub )

Una cadena de videoclubs quiere emplear una base de datos para almacenar información referente a las facturas que se hacen a los clientes. Esta información es la siguiente:

- En un mismo documento se puede guardar información de varias facturas.
- Cada factura está formada por dos tipos de información: datos de cliente y datos del ticket de factura propiamente dichos.



- De los datos del cliente se desea guardar: su nombre, su primer y segundo apellidos, DNI y teléfono (uno o varios). Además, como características del cliente, se desea conocer el identificador de cliente.
- En cada factura habrá alquileres, compras o los dos. Incluye la forma de pago y el importe total.
- Los alquileres se realizan de películas. El alquiler de películas lleva asociada una fecha de devolución que es común a todas las películas alquiladas en la misma factura.
- De cada película se quiere conocer su título, género, duración y los nombres y apellidos de dos actores que participan en ella. Existen dos atributos que definen a las películas: idPelícula y valoración.
- Además, opcionalmente se puede guardar para cada película el nombre de un archivo con la imagen de la carátula en formato jpg.
- Con respecto a las compras, puedes ser de DVDs o cintas de video pero no los dos a la vez.
- De los DVDs interesa el título del DVD, la fecha de salida al mercado y si viene o no con extras ( aunque es opcional )
- De las cintas de video se guardará el título, el formato (VHS, por ejemplo) y si está rebobinada o no.

**Hacer un ejemplo con dos facturas, una para un cliente por el alquiler de dos películas y otro de la compra de un DVD**

```
{
  "facturas": [
    {
      "cliente": {
        "idCliente": "C001",
        "nombre": "Mario",
        "apellido1": "Sánchez",
        "apellido2": "López",
        "dni": "12345678A",
        "telefonos": ["600123456", "911223344"]
      },
      "alquileres": {
        "fechaDevolucion": "2025-05-10",
        "peliculas": [
          {
            "idPelícula": "P001",
            "titulo": "La Aventura Espacial",
```

```
"genero": "Ciencia Ficción",
"duracion": 120,
"actores": [
  {
    "nombre": "Carlos",
    "apellido": "Martínez"
  },
  {
    "nombre": "Lucía",
    "apellido": "Gómez"
  }
],
"valoracion": 4.5,
"caratula": "la_aventura_espacial.jpg"
},
{
  "idPelicula": "P002",
  "titulo": "Amor en París",
  "genero": "Romántica",
  "duracion": 95,
  "actores": [
    {
      "nombre": "Javier",
      "apellido": "Ruiz"
    },
    {
      "nombre": "Elena",
      "apellido": "Fernández"
    }
  ],
  "valoracion": 4.0
}
]
},
"formaPago": "Tarjeta",
"importeTotal": 7.50
```

```
},
{
  "cliente": {
    "idCliente": "C002",
    "nombre": "Laura",
    "apellido1": "Pérez",
    "apellido2": "Díaz",
    "dni": "87654321B",
    "telefonos": ["655443322"]
  },
  "compras": {
    "tipo": "DVD",
    "dvd": {
      "titulo": "Matrix Revolutions",
      "fechaSalida": "2003-11-05",
      "extras": true
    }
  },
  "formaPago": "Efectivo",
  "importeTotal": 12.00
}
]
```

To format and validate your JSON, just copy + paste it below:

```
1 {
2   "facturas": [
3     {
4       "cliente": {
5         "idCliente": "0001",
6         "nombre": "Mario",
7         "apellido1": "Sánchez",
8         "apellido2": "López",
9         "dni": "123456789",
10        "telefonos": [
11          "987654321",
12          "987654321"
13        ]
14      },
15      "alquileres": {
16        "fechaDevolucion": "2025-05-10",
17        "películas": [
18          {
19            "idPelícula": "0001",
20            "titulo": "La Aventura Espacial",
21            "genero": "Ciencia Ficción",
22            "duracion": 120,
23            "actores": [
24              {
25                "nombre": "Carlos",
26                "apellido": "Martínez"
27              },
28            ]
29          }
30        ]
31      }
32    ]
33  }
```

Validate JSON

Clear

Compress

JSON is valid!