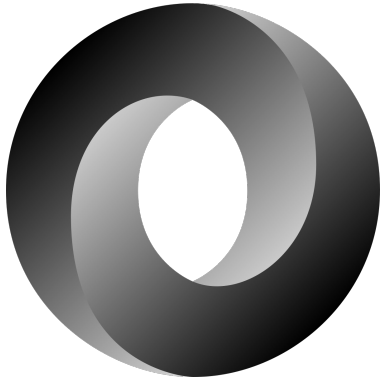


Unidad 7

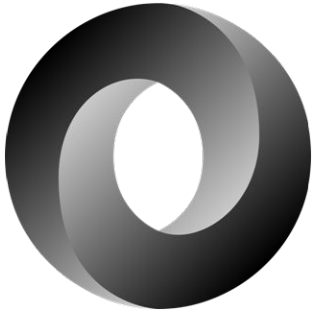
JSON, el lenguaje ligero de JavaScript

JSON, el lenguaje ligero

7.1. Introducción a JSON
7.2. Estructura y sintaxis del formato JSON
7.3. Aplicaciones de JSON
7.4. Herramientas de edición y consulta de documentos JSON

7.1. Introducción a JSON

- **JSON es el acrónimo de JavaScript Object Notation.**
- **Es un lenguaje de marcado muy sencillo apropiado, sobre todo, para el intercambio de datos**
- **Origen: representación de objetos JavaScript.**



Características de JSON:

- **Sencillo de construir y procesar.**
- **Sintaxis simple.**
- **No dispone de etiquetas.**
- **Muy versátil.**

Características de JSON:

- **JSON permite representar estructuras de datos jerárquicas.**
- **Es admitido en múltiples entornos: configuración de aplicaciones, intercambio de datos y en aplicaciones web.**
- **Utilizado en algunas bases de datos no relacionales, como Firebase o MongoDB.**

Estructura del formato JSON

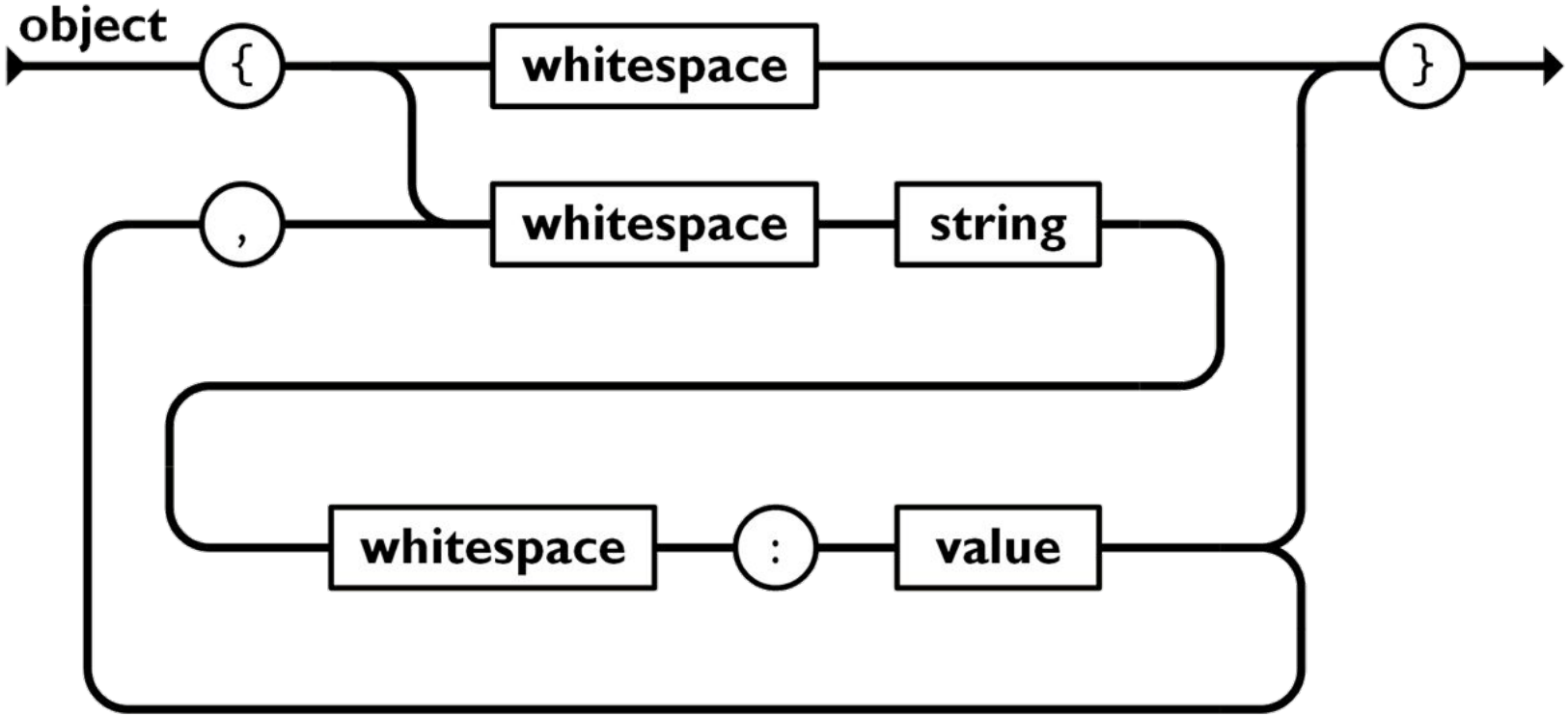
- Los ficheros se almacenan con la extensión .json.
- Este lenguaje no permite incluir comentarios.
- JSON dispone de dos tipos de elementos:
  - - Objetos.
  - - Listas o arrays.

Sintaxis de los objetos JSON:

- **Están delimitados por llaves.**
- **Contienen una lista de pares de clave-valor separados por comas.**
- **Los nombres de las claves se separan de los valores por dos puntos.**
- **Los nombres de las claves se escriben entre comillas dobles.**

```
{"titulo":"Los Ojos de Julia"}
```

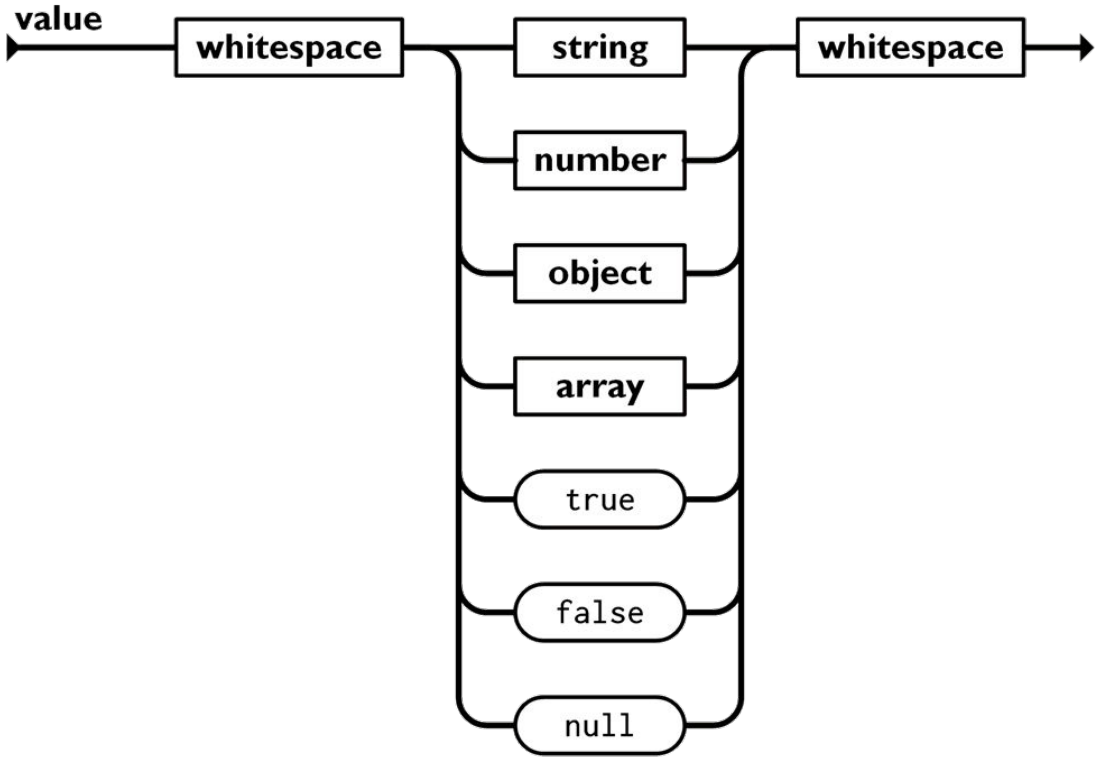
Sintaxis de los objetos JSON:



Valores de los objetos JSON

- **Cadenas de caracteres. Se escriben entre comillas dobles.**
- **Números, aceptando enteros y decimales.**
- **Valores lógicos (true y false)**
- **El valor null.**
- **Objetos.**
- **Arrays de objetos.**

Valores de los objetos JSON



Ejemplos

Un único par tipo-valor, siendo el valor una cadena de caracteres

```
{"titulo": "Los Ojos de Julia"}
```

Un único par tipo-valor, siendo el valor un número entero

```
 {"anyo":2010}
```

Un único par tipo-valor, siendo el valor un número decimal

```
{"calificacion":6.1}
```

Un único par tipo-valor, siendo el valor de tipo lógico

```
{"estreno-cine":true}
```

Dos pares tipo-valor


```
{
"titulo":"Los Ojos de Julia",
"anyo":2010
}
```


Tres pares tipo-valor, siendo el último de ellos un objeto

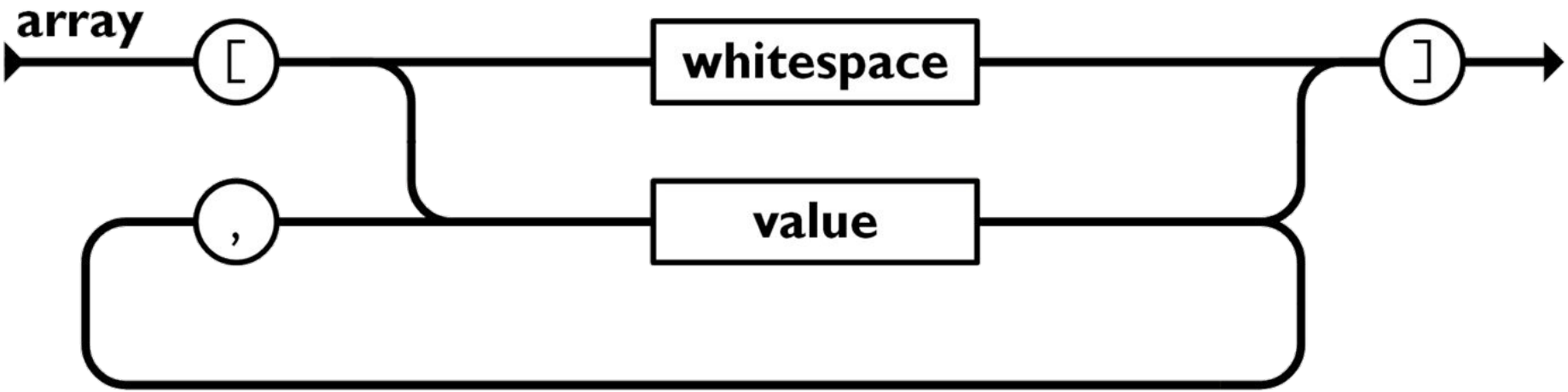

```
{ "titulo":"Los Ojos de Julia",
"anyo":2010,
"director":{
"nombre":"Guillem Morales",
"nacionalidad":"Español"
}
 }
```

Sintaxis de las listas o arrays de objetos JSON:

- **Están delimitadas por corchetes.**
- **Los elementos están separados por comas.**

```
{
   “titulo” : “Nosferatu, vampiro de la noche”,
   “titulo-original” : “Nosferatu: Phantom der Nacht”,
   “director” : “Werner Herzog”,
   “reparto” : [“Klaus Kinski”,”Bruno Ganz”,”Jacques Dufilho”]
}
```

Sintaxis de las listas o arrays de objetos JSON:



Ejemplo de lista

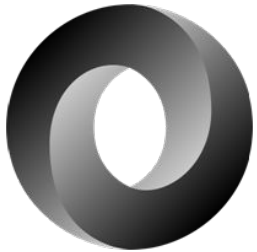

```
{ "titulo": "Nosferatu, vampiro de la noche",
"titulo-original": "Nosferatu: Phantom der Nacht",
"director": "Werner Herzog",
 "reparto": ["Klaus Kinski","Bruno Ganz","Jacques Dufilho"]
}
```

```
{ "titulo": "Los Ojos de Julia",
"anyo": 2010,
"director": {
"nombre": "Guillem Morales",
"nacionalidad": "Español" },
 "reparto": [
{ "nombre": "Belén Rueda",
 "personaje": "Julia" },
 { "nombre": "Lluis Homar",
"personaje": "Isaac" },
{ "nombre": "Francesc Orella",
"personaje": "Inspector Dimas" }
 ]
 }
```


Un documento JSON se forma con un objeto que contiene objetos o listas de objetos, creando una estructura jerárquica en forma de árbol.

Diferencias con XML:

- **Más ligero.**
- **Menos completo.**
- **No dispone de mecanismos de validación.**
- **No dispone de las mismas herramientas.**
- **Son lenguajes complementarios, no rivales.**



¿XML o JSON?

¿Cuándo elegir XML?

•Entornos más estrictos: restricciones de fiabilidad.

•Necesidad de herramientas.

•Condiciones técnicas o funcionales.

¿Cuándo elegir XML?

•Entornos más estrictos: restricciones de fiabilidad.

•Necesidad de herramientas.

•Condiciones técnicas o funcionales.

Ejemplo XML y JSON

```xml
<Clientes>
    <Cliente ID="283">
        <Nombre>Juan Carlos Crespin</Nombre>
        <Direccion verificada="si">
            <Calle>Avda Perú</Calle>
            <Numero>235</Numero>
            <Ciudad>Bahía Blanca</Ciudad>
            <Provincia>Buenos
Aires</Provincia>

<CodigoPostal>8000</CodigoPostal>
            <Pais>Argentina</Pais>
        </Direccion>
    </Cliente>
</Clientes>
```

```json
{
    "Clientes": {
        "Cliente": {
        "-ID": "283",
        "Nombre": "Juan Carlos Crespin",
        "Direccion": {
            "-verificada": "si",
            "Calle": "Avda Perú",
            "Numero": "235",
            "Ciudad": "Bahía Blanca",
            "Provincia": "Buenos Aires",
            "CodigoPostal": "8000",
            "Pais": "Argentina"
            }
        }
    }
}
```

Ejemplo XML y JSON

```xml
<persona>
    <nombre>juan</nombre>
    <edad>22</edad>
    <estudios>
        <estudio>primario</estudio>
        <estudio>secundario</estudio>
    </estudios>
</persona>
```

```json
{
    "persona":{
        "nombre":"juan",
        "edad":22,

"estudios":["primario","secundario"]
    }
}
```

Ejemplo XML y JSON

```xml
<empleados>
    <empleado>
        <nombre>Jorge</nombre>
        <apellido>Mesa</apellido>
        <edad>28</edad>
    </empleado>
    <empleado>
        <nombre>Ana</nombre>
        <apellido>Sánchez</apellido>
        <edad>21</edad>
    </empleado>
    <empleado>
        <nombre>Pedro</nombre>
        <apellido>Lee</apellido>
        <edad>44</edad>
    </empleado>
</empleados>
```

```json
{
    "empleados":[
            {
            "nombre": "Jorge",
            "apellido": "Mesa",
            "edad": 28
            },
            {
            "nombre": "Ana",
            "apellido": "Sánchez",
            "edad": 21
            },
            {
            "nombre": "Pedro",
            "apellido": "Lee",
            "edad": 44
            }
        ]
}
```

Ejemplo XML y JSON

```xml
<libros>
    <libro id="1" year="1999">
    <autores>
        <nombre>Ricardo Pérez</nombre>
        <nombre>Marta Gómez</nombre>
    </autores>
    </libro>
</libros>
```

```json
{
    "libros": {
        "libro": {
            "-year": "1999",
            "-id": "1",
            "autores": {
            "nombre": [
                "Ricardo Pérez",
                "Marta Gómez"
                ]
            }
        }
    }
}
```

7.3. Aplicaciones de JSON

JSON se utiliza en múltiples contextos:

- Configuración de aplicaciones.
- Intercambio de datos entre aplicaciones convencionales.
- Intercambio de datos entre aplicaciones web.
- Almacenamiento.

7.3. Aplicaciones de JSON

JSON se utiliza en varios contextos distintos. Esto es posible gracias a que está formado por texto plano. Además utiliza tipos de datos muy básicos y universales, facilitando la existencia de procesadores en cualquier lenguaje de programación.

7.3.1 Configuración de aplicaciones

Consiste en dar soporte al mantenimiento de la configuración de aplicaciones. Por ejemplo, supongamos una aplicación que permite configurar los siguientes parámetros:

- Color de fondo.

- Tamaño de texto.

- Color de texto.

- Idioma.

- Pantalla de inicio.

7.3.1 Configuración de aplicaciones

La estructura de fichero JSON sería:

```
{
  "usuarios": [
    {
      "identificador": "jhipolito",
      "configuracion": {
        "color-fondo": "#AA99AA",
        "tamaño-texto": 12,
        "color-texto": "#441238",
        "idioma": "es",
        "pantalla-inicio": "listado-clientes"
      }
    },
    {
      "identificador": "alanchas",
      "configuracion": {
        "color-fondo": "#8911F8",
        "tamaño-texto": 14,
        "color-texto": "#311103",
        "idioma": "it",
        "pantalla-inicio": "listado-deudores"
      }
    }
  ]
}
```

7.3.2 Intercambio de datos y servicios web

- Es el uso más habitual de los ficheros JSON.
- En muchos de los casos de intercambio de datos, JSON puede utilizarse como formato para representar los datos, garantizando la interoperabilidad: todos los lenguajes y todos los sistemas operativos son capaces de procesar ficheros de texto planos.
- Por otra parte, las comunicaciones en tiempo real entre sistemas requieren formatos de datos compatibles y representaciones susceptibles de ser enviadas por procedimientos telemáticos como sockets, RPC o servicios web -todos ellos mecanismos de comunicación entre sistemas-.

7.3.2 Intercambio de datos y servicios web

- En algunos sistemas, se utiliza JSON para almacenar datos.

- Por ejemplo, Firebase Realtime Database de Google en tiempo real alojada en la nube, almacena sus datos en formato JSON, lo que facilita el tratamiento desde los programas clientes de la base de datos. Proporciona la posibilidad de importar y exportar ficheros JSON.

7.4. Herramientas de edición y consulta de documentos JSON

Algunas herramientas de edición y visualización de documentos JSON:

- Editores de escritorio.
- Editores en línea.
- Navegadores.