Aplicaciones Telemáticas

XML y JSON

J. E. López Patiño, F. J. Martínez Zaldívar

Índice

Definición de XML

- XML (eXtensible Markup Language) es un lenguaje para la definición de lenguajes de marcado desarrollado por el W3C *(World Wide Web Consortium)*:

  "El Lenguaje Extensible de Marcado (XML) describe una clase de objetos llamados documentos XML y parcialmente describe el comportamiento de programas de computador que pueden procesarlos"

- Deriva de otro estándar: SGML *(Standard Generalized Markup Language")* ISO 8879:1986. (HTML deriva también de SGML). SGML proviene de GML *(Generalized Markup Language)* ideado por IBM en los 70
- Se propone como estándar de intercambio de información estructurada entre plataformas. Usado en BB. DD., editores de texto, hojas de cálculo, ...

Definición de JSON

- JSON (JavaScript Object Notation)

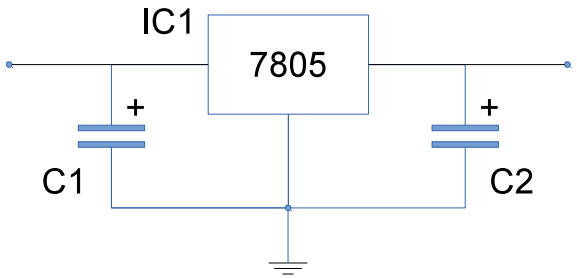
  "Estándar de formato de texto legible por humanos para transmitir objetos de datos consistentes en pares nombre-valor"

- Deriva del lenguaje de programación JavaScript: notación empleada para definir literalmente objetos
- Más ligero que XML: se tiende a trabajar más con JSON.

Características de XML

- Define la sintaxis y los requisitos que deben cumplir los lenguajes de marcado que especifica
- Marcado: señales *añadidas* a la información para ayudar a su procesado
- Extensible: permite definir *nuestras* propias marcas
- General: cualquier clase de objetos de datos
- Descriptivo: describe datos, pero no qué hacer con ellos
- Implica tener la información estructurada
- Perspectiva de documentos XML:
  - Documentos XML bien formados: sintácticamente correctos
  - Documentos XML válidos: semánticamente correctos

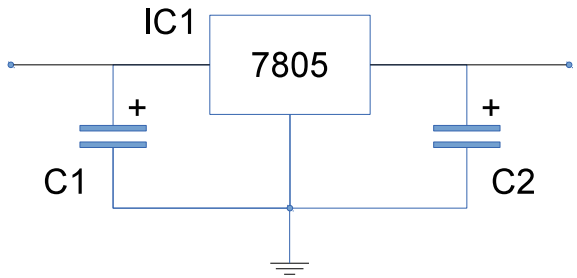
Ejemplo: BOM *(Bill Of Materials)* regulador de tensión



ID	Componente	Tipo	Valor	Tensión	Encapsulado	Disipador
C1	condensador	cerámico	0,47 μF	50 V	0805	-
C2	condensador	electrolítico	0,1 μF	12 V	-	-
IC1	circuito integrado	LM7805	-	-	T0220	aluminio

Ejemplo: BOM regulador de tensión descrito en XML

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



Ejemplo JSON

regulador.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```

regulador.json

```
{ "regulador" :
    {
    "condensadores" : [
      {
        "ID":"C1",
        "tipo":"ceramico",
        "fabricante":"Murata",
        "valor":{"magnitud":0.47, "unidades":"uF"},
        "voltaje":{"magnitud":50, "unidades":"V"},
        "encapsulado":"0805",
        "disipador":false
      },
      {
        "ID":"C2",
        "tipo":"electrolitico",
        "fabricante":"Murata",
        "valor":{"magnitud":0.1, "unidades":"uF"},
        "voltaje":{"magnitud":12, "unidades":"V"},
        "encapsulado":null,
        "disipador":false
      }
    ],
    "circuitos integrados" : [
      {
        "ID":"IC1",
        "tipo":"LM7805",
        "fabricante":"Fairchild Semiconductor",
        "encapsulado":"T0220",
        "disipador":true
      }
    ]
  }
}
```

Índice

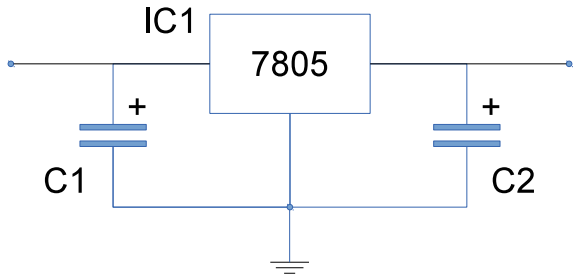
Prólogo y cuerpo

- Prólogo
  - Declaración XML / Instrucciones de procesado (PI: *Processing Instructions*)
  - Declaración DTD (Document Type Declaration)
  - Comentarios
- Documento, cuerpo o elemento raíz
  - Elementos
    - Atributos
    - Datos
    - Otros elementos
    - Comentarios

Índice

Ejemplo con declaración XML y comentario

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



Declaración XML y comentarios

```
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
```

- `version`: versión de XML empleada (1.0 y 1.1 hasta la actualidad)
- `encoding` (opcional): código de caracteres empleados
- `standalone` (opcional): autocontenido o no.
- Orden expuesto de atributos anteriores: obligatorio

Comentario: `<!-- ... -->`
(No se deben incluir los caracteres `--` dentro de un comentario)

Índice

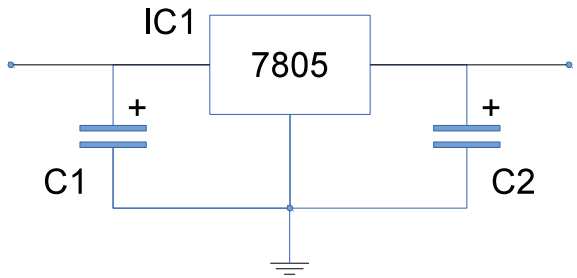
Codificación del contenido

- Diferencia entre:
  - Juego, conjunto o repertorio de caracteres: colección ordenada de caracteres
  - Glifo: dibujo del carácter
  - Código de carácter: identificación unívoca del carácter
  - Codificación del carácter: representación digital mediante secuencia de uno o más octetos.
  - En ocasiones: todos los conceptos anteriores ligados.
  - Ejemplos:
    - UNICODE (1 114 112 caracteres), U+XXXX[XXXX], UTF-8/UTF-16/...
    - ISO-8859-1/ISO-8859-15, ... (255 caracteres)
    - ASCII (128 caracteres)
    - ...

Índice

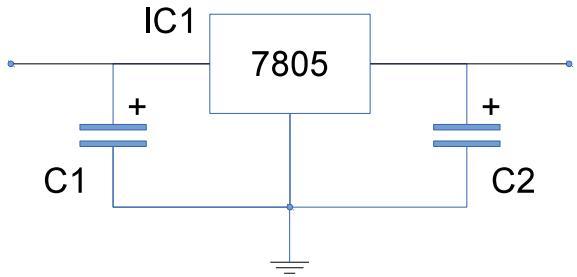
Elemento raíz (documento): sólo uno

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



Un elemento cualquiera

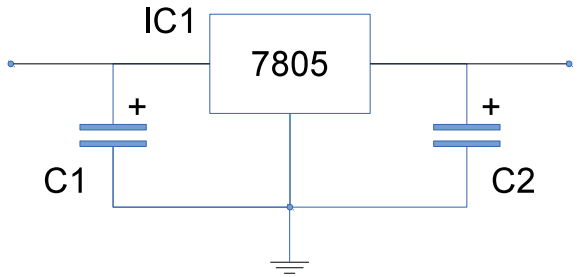
```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



Índice

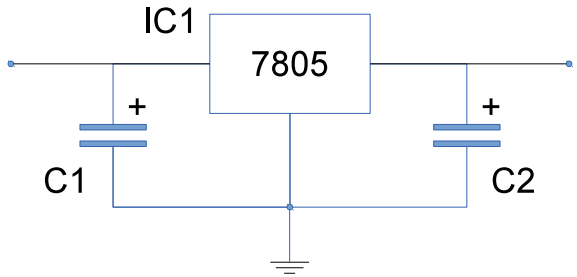
De apertura

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



De cierre (correctamente anidadas)

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



Visualización en navegador

Nombres

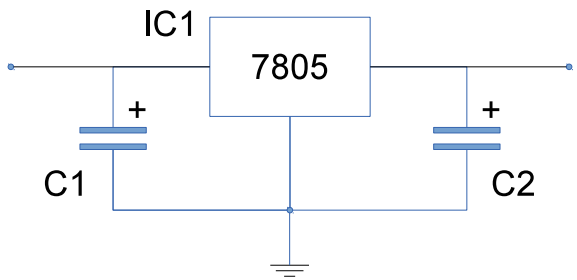
- Formato:

  ⋮
  <etiqueta␣>
  ⋮
  </etiqueta␣>
  ⋮
  <etiqueta␣/>
  ⋮

  donde ␣ indica 0 ó más espacios en blanco
- No debe haber ningún espacio entre < ó </ y `etiqueta`
- Los nombres de las etiquetas en mayúsculas o minúsculas son distintos
- Los nombres de las etiquetas:
  - Pueden contener letras, números y otros caracteres no especiales
  - No pueden empezar con número o carácter de puntuación. Pueden comenzar por letra, guión bajo o carácter de dos puntos (reservadas)
  - No pueden comenzar por las letras `xml` o `XML` o `Xml`, ...
  - No pueden contener espacios

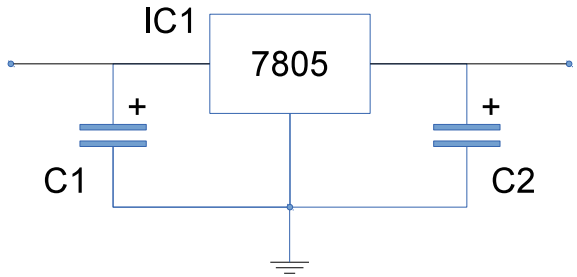
Atributo

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



Valor del atributo

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



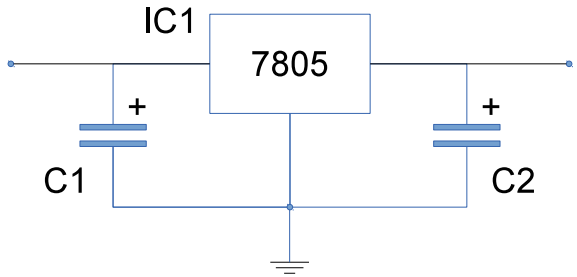
Valor del atributo (cont.)

- Dentro de etiqueta de apertura (o autocierre: `<etiqueta a="b"/>`)
- Obligación de asignación de valor
- Valor debe estar entrecomillado (con comillas simples o dobles —ambas iguales—)

```
<condensador tipo='cerámico' fabricante="Murata's Electronics">
  C1
  <valor unidades="uF">0,47</valor>
  <voltaje unidades="V">50</voltaje>
  <encapsulado>0805</encapsulado>
  <disipador></disipador>
</condensador>
```

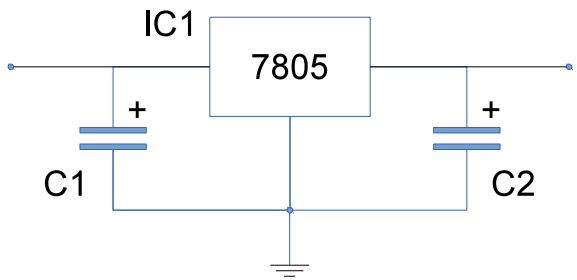
Contenido

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



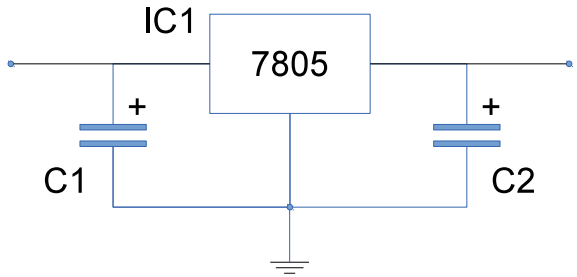
Dato o texto

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



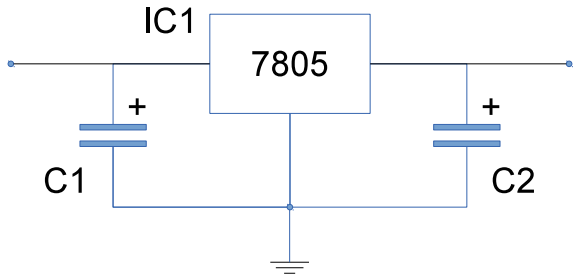
Relaciones entre elementos: padre (parent) e hijo (child)

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



Relaciones entre elementos: hermanos (siblings)

```
<?xml version="1.0" encoding="UTF-8" ?>
<!-- Lista de componentes de un regulador de tensión de 5 V -->
<regulador>
  <condensador tipo="cerámico" fabricante="Murata">
    C1
    <valor unidades="uF">0,47</valor>
    <voltaje unidades="V">50</voltaje>
    <encapsulado>0805</encapsulado>
    <disipador></disipador>
  </condensador>
  <condensador tipo="electrolítico" fabricante="Murata">
    C2
    <valor unidades="uF">0,1</valor>
    <voltaje unidades="V">12</voltaje>
    <encapsulado/>
    <disipador/>
    <!-- Quizá convendría aumentar la capacidad??? -->
  </condensador>
  <circuito_integrado tipo="LM7805" fabricante="Fairchild Semiconductor">
    IC1
    <encapsulado>TO220</encapsulado>
    <disipador>aluminio</disipador>
  </circuito_integrado>
</regulador>
```



Caracteres de *escape*

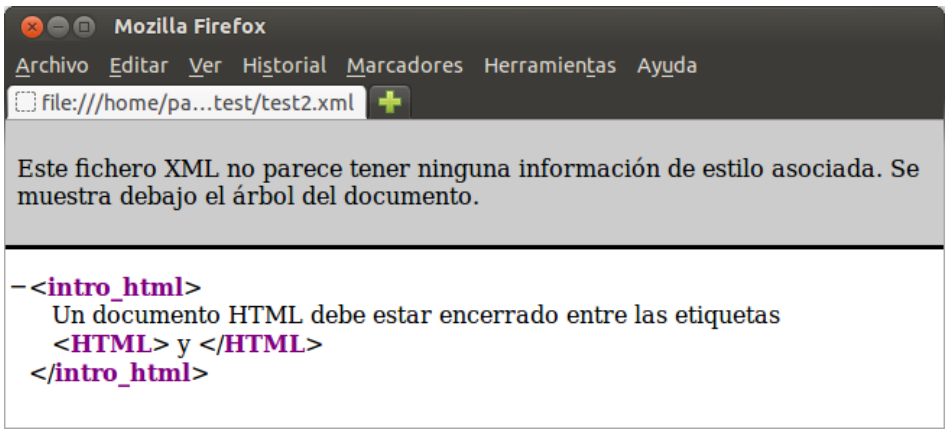
- Si aparecen los caracteres < ó > puede haber ambigüedad o incorrección; solución: secuencias de *escape*:
  - <→ <
  - >→ >
- Otros caracteres de *escape*:
  - & → &
  - ' (comilla simple o apóstrofo) → '
  - "(comillas dobles) → "
- Pueden considerarse `ENTIDADES` predefinidas
- Caracteres Unicode ISO/IEC 10646: forma de expresar cualquier carácter (no necesariamente caracteres de *escape*)

Cáracter	Código	Código (HEX.)	Carácter	Código	Código (hex.)
Á	&#193;	&#xC1;	á	&#225;	&#xe1;
É	&#201;	&#xC9;	é	&#233;	&#xe9;
Í	&#205;	&#xCD;	í	&#237;	&#xed;
Ñ	&#209;	&#xD1;	ñ	&#241;	&#xf1;
Ó	&#211;	&#xD3;	ó	&#243;	&#xf3;
Ú	&#218;	&#xDA;	ú	&#250;	&#xfa;
Ü	&#220;	&#xDC;	ü	&#252;	&#xfc;
¿	&#191;	&#xBF;	¡	&#161;	&#xa1;

Caracteres de *escape*

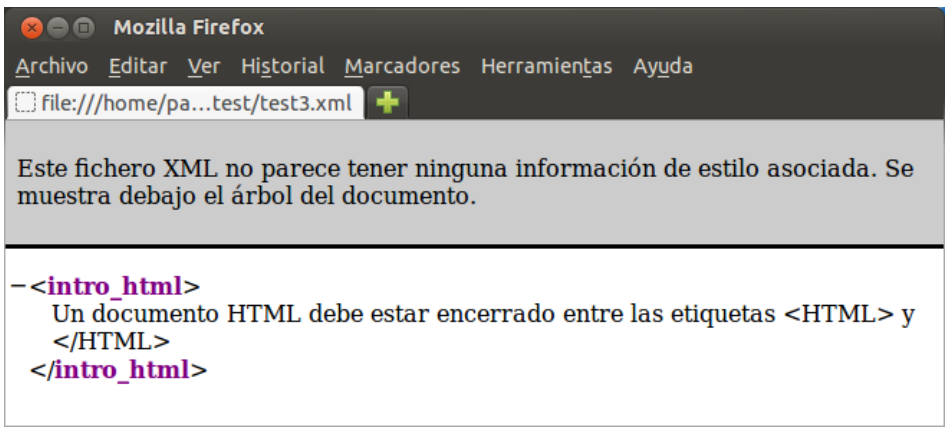
sin_entidad.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<intro_html>
Un documento HTML debe estar encerrado entre las
etiquetas <HTML> y </HTML>
</intro_html>
```



con_entidad.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<intro_html>
Un documento HTML debe estar encerrado entre las
etiquetas &lt;HTML&gt; y &lt;/HTML&gt;
</intro_html>
```



Secciones CDATA (vs. PCDATA)

- CDATA *(Character DATA)*: sección etiquetada

Sección CDATA

```
<![CDATA[  ... ]]>
```

- El contenido interno se interpreta tal y como aparece, sin necesidad de emplear caracteres especiales

Ejemplo sección CDATA

```
<?xml version="1.0" encoding="UTF-8" ?>
<intro_html>
<![CDATA[Un documento HTML debe estar encerrado entre las etiquetas <HTML> y </HTML>]]>
</intro_html>
```

- PCDATA *(Parsed Character Data)*: datos a analizar
  - Se analizan elementos XML
  - Se expanden entidades

Índice

Definición

- Un espacio de nombres es un conjunto de nombres en el que todos los nombres son únicos.
  - Los nombres de los pueblos de una provincia
  - Dominios de internet
  - Palabras reservadas de Java
  - Hijos de una familia
  - ...
- Creación de un espacio de nombres:

```
xmlns:prefijo="identificador_espacio_nombres"
```

  - No se debe interpretar como un atributo
  - Valor es irrelevante, pero debe ser distintivo. En ocasiones es una URI donde aparece algún tipo de descripción
- Se puede definir en la misma etiqueta de uso
- Nombre cualificado:

`prefijo:nombre_local`

Espacio de nombres: ejemplo

- Ejemplo

Espacio de nombres

```
<aatt:alumno
 xmlns:aatt="http://www.etsit.upv.es/aatt"
 xmlns:conocimientos="http://www.etsit.upv.es/aatt/conocimientos"
 xmlns="http://www.etsit.upv.es/aatt/convoc">
    <aatt:id>12</aatt:id>
    <aatt:nombre>Pepe Gutiérrez</aatt:nombre>
    <aatt:lenguaje>Italiano</aatt:lenguaje>
    <aatt:titulo>Introduccion a XML</aatt:titulo>
    <aatt:titulo xmlns:aatt="http://www.etsit.upv.es/aatt/titulos">
      Bachiller
    </aatt:titulo>
    <conocimientos:lenguaje>Java</conocimientos:lenguaje>
    <aatt:nota>9,5</aatt:nota>
  <convocatoria>1</convocatoria>
    <convocatoria xmlns="">7</convocatoria>
</aatt:alumno>
```

Índice

Sintaxis correcta

- El documento sólo debe contener caracteres válidos del juego de caracteres indicado
- Sólo hay un elemento raíz que contiene a los restantes
- Los nombres de los elementos y de sus atributos no tienen espacios (espacios en blanco, tabuladores, saltos de línea, . . .)
- El primer caracter de un nombre de elemento o atributo pude ser:
  - Letra
  - Carácter de dos puntos (:)
  - Carácter de subrayado (_)
- Resto de caracteres: además, números, guiones (-) y puntos (.)
- "<" y "&" sólo empleados como comienzo de marcas; en caso contrarios deben utilizarse con códigos de *escape*
- Etiquetas de apertura, cierre y vacías correctamente anidadas, sin faltar ni sobrar ninguna; además coincidencia en mayúsculas/minúsculas
- Las etiquetas de cierre no tienen atributos
- Una etiqueta no puede tener dos atributos distintos con el mismo nombre
- Los valores de los atributos: entrecomillados
- Los atributos deben tener algún valor (aunque sea vacío: '' ó "")
- No existen referencias en los valores de los atributos

Índice

DTD: Document Type Declaration

- *Gramática* para que un documento XML se considere válido
- Los navegadores no suelen *analizar sintácticamente* un documento XML
- Validador *online*: `http://www.xmlvalidation.com/`
- Forma:

DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE raiz [ ... ]>
<raiz>
   ...
</raiz>
```

- Descripción de elementos y atributos

Ejemplo DTD: biblioteca

biblio.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE biblioteca [
   <!ELEMENT biblioteca (libro)* >
   <!ELEMENT libro ( titulo, autores, fecha, ISBN,
                     editorial ) >
   <!ELEMENT titulo (#PCDATA )>
   <!ELEMENT autores ( autor+ | anonimo )>
   <!ELEMENT autor (nombre_de_pila, apellido+) >
   <!ELEMENT nombre_de_pila (#PCDATA) >
   <!ELEMENT apellido (#PCDATA) >
   <!ELEMENT ISBN ( #PCDATA ) >
   <!ELEMENT editorial ( #PCDATA ) >
   <!ELEMENT fecha (#PCDATA) >
   <!ELEMENT anonimo EMPTY>
   <!ATTLIST biblioteca nombre CDATA #FIXED "UPV">
   <!ATTLIST titulo idioma (castellano | ingles |
           frances | aleman | chino) "castellano">
   <!ATTLIST libro genero ( policiaco | caballeresco
      | historia | terror | picaresco )  #IMPLIED>
]>
...
```

biblio.xml (cont.)

```
...
<biblioteca>
<libro genero="caballeresco">
   <titulo>El ingenioso hidalgo Don Quijote de
           La Mancha</titulo>
   <autores>
      <autor>
         <nombre_de_pila>Miguel de</nombre_de_pila>
         <apellido>Cervantes</apellido>
         <apellido>Saavedra</apellido>
      </autor>
   </autores>
   <fecha>1605</fecha>
   <ISBN>123456789X</ISBN>
   <editorial>Planeta</editorial>

</libro>
<libro genero="picaresco">
   <titulo>La vida de Lazarillo de Tormes y de sus
           fortunas y adversidades</titulo>
   <autores>
      <anonimo/>
   </autores>
   <fecha>1554</fecha>
   <ISBN>123456780X</ISBN>
   <editorial>Juan de Junta, Burgos</editorial>
</libro>
</biblioteca>
```

DTD externo y público

Externo y público

```
<!DOCTYPE raiz PUBLIC "nombre_DTD" "localizacion_DTD">
```

con

- nombre_DTD:

  "prefijo//propietario_del_DTD//descripcion_del_DTD//identificador_del_idioma_ISO_639"

  Prefijos:

Prefijo	Definición
ISO	La DTD es un estándar ISO (por tanto aprobado)
+	La DTD es un estándar no-ISO, pero aprobado
-	La DTD es un estándar no-ISO y no aprobado

- localizacion_DTD: URL

Ejemplo fichero (X)HTML

```
<?xml version="1.0" standalone="no" ?>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN"
  "http://www.w3.org/TR/REC-html40/loose.dtd">
<HTML>
<HEAD>
<TITLE>Fichero HTML típico</TITLE>
</HEAD>
<BODY>
  Estructura típica de un fichero HTML. Sigue la especificación
  HTML 4.0, incluyendo etiquetas obsoletas (de ahí la calificación
  de transicional)
</BODY>
</HTML>
```

Índice

Alrededor de XML...

- XSD: alternativa a DTD
  - El *esquema* está también escrito en XML
- XSL: *(eXtensible Stylesheet Language)*: lenguaje para describir el estilo de documentos HTML
  - XSLT *(*eXtensible Stylesheet Languge Transformations): transformaciones XSL, lenguaje para transformar documentos XML en otros documentos XML (ej.: para visualización en navegador)
- XPath: lenguaje para *navegar* en documentos XML
- XQuery: lenguaje para realizar búsquedas en documentos XML
- XLink: lenguaje que define la manera de crear hiperenlaces en documentos XML
- XPointer: permite a los hiperenlaces apuntar a partes específicas de documentos XML

Índice

Analizadores *parsers* de XML

- Clases de Java para analizar/generar documentos XML.
- Diferentes *filosofías*
  - DOM: Document Object Model: paquetes de `org.w3c.com` en JDK SE
  - SAX: Simple API for XML: paquetes de `org.xml.sax` en JDK SE
  - JDOM: Java Document Object Model (no en JDK SE)
  - StAX: Streaming API for XML (no en JDK SE)
  - DOM4J: DOM for Java (no en JDK SE)

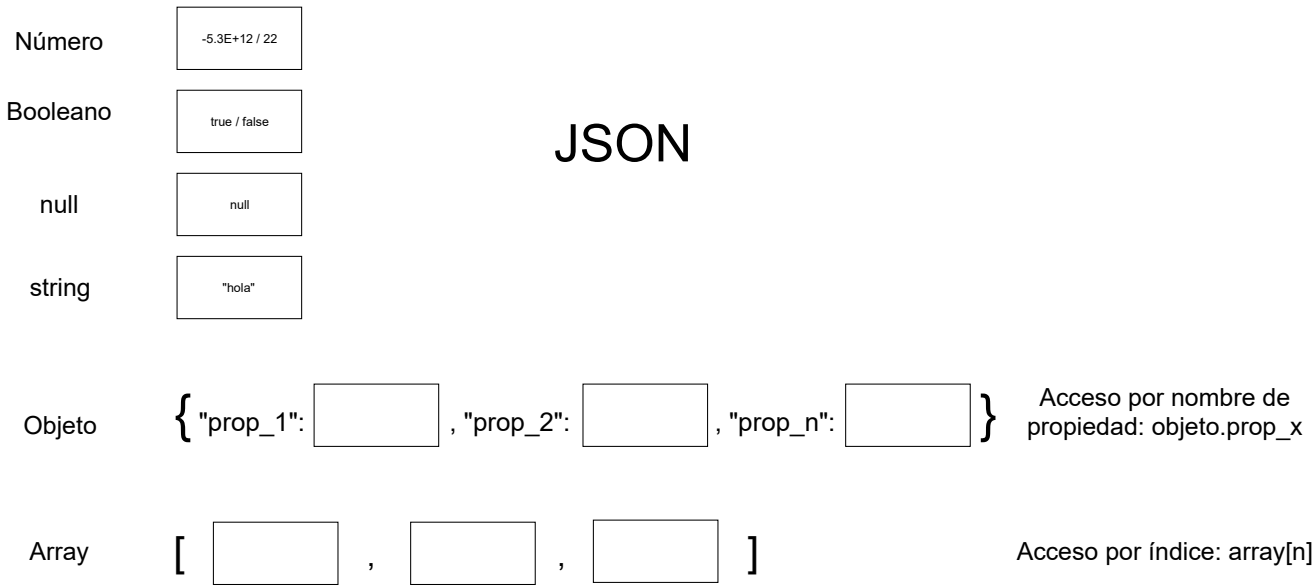
Índice

Índice

Objetos y arrays JSON

- Notación: objeto JSON encerrado entre llaves (objeto) / corchetes (array)
- Propiedades/componentes del objeto/array separadas por comas
- Cada propiedad/componente de un objeto/array consta de:
  - pares `"nombre":valor` / `valor`
- `valor` puede ser
  - Un número (entero o real)
  - String (entre comillas dobles)
  - Un booleano: true o false
  - Un objeto JSON (entre llaves $\{\ldots\}$)
  - Un array (entre corchetes $[\ldots]$)
    - Los elementos de un array se separan por comas
    - Los elementos de un array JSON pueden ser cualquier tipo de valor descrito hasta ahora
  - `null`
- Un validador: `http://jsonviewer.stack.hu/`

Gráficamente



Paquete `javax.json` de Java

- No disponible en JDK SE, sino en JDK EE
- Búsquese (en Google) con criterio: `javax json`
- Un sitio: `http://www.java2s.com`
  - ⇒ `14. Java File Download` ⇒ `j` ⇒ `javax.json`
- 2 clases, 1 excepción y 13 interfaces

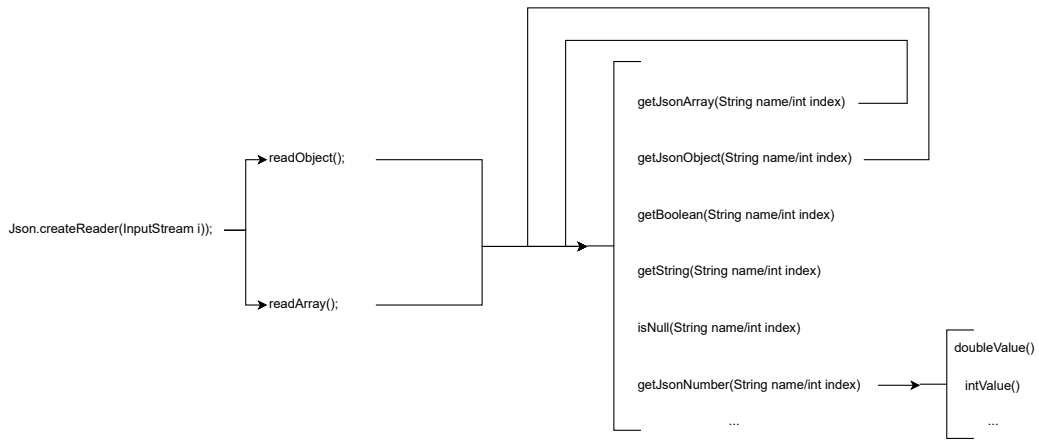
Paquete `javax.json` de Java (cont.)

- Clases:
  - `Json`: clase *factoría* para crear objetos de procesamiento JSON
  - `JsonValue.ValueType`: clase enumerada para indicar tipo de un objeto `JsonValue`
- Interfaces:
  - Componentes:
    - `JsonObject`: objeto JSON
    - `JsonArray`: array JSON
    - `JsonStructure`: superclase de `JsonObject` y `JsonArray`
    - `JsonNumber`: número JSON
    - `JsonString`: string JSON
    - `JsonValue`: valor genérico
  - *Builders*:
    - `JsonArrayBuilder`
    - `JsonObjectBuilder`
  - Interfaz *factoría* para crear *JsonArrayBuilder* o *JsonObjectBuilder*:
    - `JsonBuilderFactory`
  - Lector/escritor de objetos o arrays JSON
    - `JsonReader`
    - `jsonWriter`
  - Interfaces *factoría* para crear *JsonReader*/*JsonWriter*
    - `JsonReaderFactory`
    - `JsonWriterFactory`

Paquete `javax.json` de Java: análisis *(parsing)*

regulador.json

```
{ "regulador" :
    {
    "condensadores" : [
      {
        "ID":"C1",
        "tipo":"ceramico",
        "fabricante":"Murata",
        "valor":{"magnitud":0.47, "unidades":"uF"},
        "voltaje":{"magnitud":50, "unidades":"V"},
        "encapsulado":"0805",
        "disipador":false
      },
      {
        "ID":"C2",
        "tipo":"electrolitico",
        "fabricante":"Murata",
        "valor":{"magnitud":0.1, "unidades":"uF"},
        "voltaje":{"magnitud":12, "unidades":"V"},
        "encapsulado":null,
        "disipador":false
      }
    ],
    "circuitos integrados" : [
      {
        "ID":"IC1",
        "tipo":"LM7805",
        "fabricante":"Fairchild Semiconductor",
        "encapsulado":"T0220",
        "disipador":true
      }
    ]
  }
}
```



Obsérvense sobrecargas y otros métodos en la documentación

Ejemplo: objeto JSON desde String

TestJSON.java

```
import java.io.StringReader;
import javax.json.*;

public class TestJSON {
    public static void main(String[] args) {

        String raizStr =
            "{ \"regulador\": " +
                "{" +
                "\"condensadores\" : [" +
                  "{" +
                    "\"ID\":\"C1\"," +
                    "\"tipo\":\"ceramico\"," +
                    "\"fabricante\":\"Murata\"," +
                    "\"valor\":{\"magnitud\":0.47, \"unidades\":\"uF\"}," +
                    "\"voltaje\":{\"magnitud\":50, \"unidades\":\"V\"}," +
                    "\"encapsulado\":\"0805\","+
                    "\"disipador\":false" +
                  "}," +
                  "{" +
                    "\"ID\":\"C2\"," +
                    "\"tipo\":\"electrolitico\"," +
                    "\"fabricante\":\"Murata\"," +
                    "\"valor\":{\"magnitud\":0.1, \"unidades\":\"uF\"}," +
                    "\"voltaje\":{\"magnitud\":12, \"unidades\":\"V\"}," +
                    "\"encapsulado\":null," +
                    "\"disipador\":false" +
                  "}" +
                "]," +
                "\"circuitos integrados\" : [" +
                  "{" +
                    "\"ID\":\"IC1\"," +
                    "\"tipo\":\"LM7805\"," +
                    "\"fabricante\":\"Fairchild Semiconductor\"," +
                    "\"encapsulado\":\"TO220\"," +
                    "\"disipador\":true +
                  "}" +
                "]" +
              "}" +
            "}" ;

        System.out.println("String raiz:\n" + raizStr + "\n");

        JsonReader lector = Json.createReader(new StringReader(raizStr));

        JsonObject raiz = lector.readObject();

        System.out.println("Objeto JSON regulador:\n" + raiz + "\n");

        JsonObject reguladorObject = reguladorRaiz.getJsonObject("regulador");

        lector.close();

        JsonArray condensadores = reguladorObject.getJsonArray("condensadores");
```

TestJSON.java (cont.)

```
        for (JsonValue condensador : condensadores) {
            JsonObject capacitor = (JsonObject) condensador;

            String ID = capacitor.getString("ID");

            String tipo = capacitor.getString("tipo");

            String fabricante = capacitor.getString("fabricante");

            JsonObject valor = capacitor.getJsonObject("valor");
            double capacidad = valor.getJsonNumber("magnitud").doubleValue();
            String unidades_capacidad = valor.getString("unidades");

            JsonObject voltaje = capacitor.getJsonObject("voltaje");
            double tension = voltaje.getJsonNumber("magnitud").doubleValue();
            String unidades_tension = voltaje.getString("unidades");

            // En todas aquellas propiedades cuyo valor posible sea null,
            // debe emplearse esta sobrecarga para evitar Excepcion
            String encapsulado = capacitor.getString("encapsulado", null);

            boolean disipador = capacitor.getBoolean("disipador");

            System.out.println("ID: " + ID);
            System.out.println("tipo: " + tipo);
            System.out.println("fabricante: " + fabricante);
            System.out.println("valor: " + capacidad + unidades_capacidad);
            System.out.println("voltaje: " + tension + unidades_tension);
            System.out.println("encapsulado: " + encapsulado);
            System.out.println("disipador: " + disipador);
            System.out.println("\n");
        }

        JsonArray CIs = reguladorObject.getJsonArray("circuitos integrados");
        for (JsonValue CI : CIs) {
            JsonObject ci = (JsonObject) CI;
            String ID = ci.getString("ID");
            String tipo = ci.getString("tipo");
            String fabricante = ci.getString("fabricante");
            String encapsulado = ci.getString("encapsulado");
            boolean disipador = ci.getBoolean("disipador");

            System.out.println("ID: " + ID);
            System.out.println("tipo: " + tipo);
            System.out.println("fabricante: " + fabricante);
            System.out.println("encapsulado: " + encapsulado);
            System.out.println("disipador: " + disipador);
            System.out.println("\n");
        }

    } // main

}  // Class
```

Compilación y ejecución

```
javac -cp javax.json-1.0.jar TestJSON.java
java  -cp javax.json-1.0.jar;. TestJSON // java -cp javax.json-1.0.jar:. TestJSON # en UNIX
```

Ejemplo II: creación objeto JSON

TestJSON2.java

```
import javax.json.*;

public class TestJSON2 {
    public static void main(String[] args) {


        JsonBuilderFactory factory = Json.createBuilderFactory(null);
        JsonObject reguladorRaiz = factory.createObjectBuilder()
                .add("regulador", factory.createObjectBuilder()
                    .add("condensadores", factory.createArrayBuilder()
                        .add(factory.createObjectBuilder()
                            .add("ID", "C1")
                            .add("tipo", "ceramico")
                            .add("fabricante", "Murata")
                            .add("valor", factory.createObjectBuilder()
                                .add("magnitud", 0.47)
                                .add("unidades", "uF")
                                )
                            .add("voltaje", factory.createObjectBuilder()
                                .add("magnitud", 50)
                                .add("unidades", "V")
                                )
                            .add("encapsulado", "0805")
                            .add("disipador", false)
                            )
                        .add(factory.createObjectBuilder()
                            .add("ID", "C2")
                            .add("tipo", "electrolítico")
                            .add("fabricante", "Murata")
                            .add("valor", factory.createObjectBuilder()
                                .add("magnitud", 0.1)
                                .add("unidades", "uF")
                                )
                            .add("voltaje", factory.createObjectBuilder()
                                .add("magnitud", 12)
                                .add("unidades", "V")
                                )
                            .add("encapsulado", JsonValue.NULL)
                            .add("disipador", false)
                            )
                        )
                    .add("circuitos integrados", factory.createArrayBuilder()
                        .add(factory.createObjectBuilder()
                            .add("ID", "IC1")
                            .add("tipo", "LM7805")
                            .add("fabricante", "Fairchild Semiconductor")
                            .add("encapsulado", "TO220")
                            .add("disipador", true)
                        )
                    )
                ).build();



        System.out.println("Objeto JSON regulador:\n" + reguladorRaiz + "\n");

        JsonObject reguladorObject = reguladorRaiz.getJsonObject("regulador");


        JsonArray condensadores = reguladorObject.getJsonArray("condensadores");
```

TestJSON2.java (cont.)

```
        for (JsonValue condensador : condensadores) {
            JsonObject capacitor = (JsonObject) condensador;

            String ID = capacitor.getString("ID");

            String tipo = capacitor.getString("tipo");

            String fabricante = capacitor.getString("fabricante");

            JsonObject valor = capacitor.getJsonObject("valor");
            double capacidad = valor.getJsonNumber("magnitud").doubleValue();
            String unidades_capacidad = valor.getString("unidades");

            JsonObject voltaje = capacitor.getJsonObject("voltaje");
            double tension = voltaje.getJsonNumber("magnitud").doubleValue();
            String unidades_tension = voltaje.getString("unidades");

            // En todas aquellas propiedades cuyo valor posible sea null,
            // debe emplearse esta sobrecarga para evitar Excepcion
            String encapsulado = capacitor.getString("encapsulado", null);

            boolean disipador = capacitor.getBoolean("disipador");

            System.out.println("ID: " + ID);
            System.out.println("tipo: " + tipo);
            System.out.println("fabricante: " + fabricante);
            System.out.println("valor: " + capacidad + unidades_capacidad);
            System.out.println("voltaje: " + tension + unidades_tension);
            System.out.println("encapsulado: " + encapsulado);
            System.out.println("disipador: " + disipador);
            System.out.println("\n");
        }

        JsonArray CIs = reguladorObject.getJsonArray("circuitos integrados");
        for (JsonValue CI : CIs) {
            JsonObject ci = (JsonObject) CI;
            String ID = ci.getString("ID");
            String tipo = ci.getString("tipo");
            String fabricante = ci.getString("fabricante");
            String encapsulado = ci.getString("encapsulado");
            String disipador = ci.getBoolean("disipador");

            System.out.println("ID: " + ID);
            System.out.println("tipo: " + tipo);
            System.out.println("fabricante: " + fabricante);
            System.out.println("encapsulado: " + encapsulado);
            System.out.println("disipador: " + disipador);
            System.out.println("\n");
        }


    } // main

}  // Class
```

Índice

Ejemplo servicio web de cambio de divisas

Véase: `https://exchangeratesapi.io/`

Cambio.java

```
import java.io.StringReader;
import javax.json.*;
import java.net.URLEncoder;
import java.net.URL;
import java.util.Scanner;

class Cambio {

  public static void main (String[] args) throws Exception  {

    // URL
    String s = "https://api.exchangeratesapi.io/latest";
    System.out.println("API REST: " + s);

    URL url = new URL(s);

    // lectura respuesta API REST
    Scanner scan = new Scanner(url.openStream());
    String str = new String();
    while (scan.hasNext())       str += scan.nextLine();
    scan.close();
    System.out.println("Respuesta: " + str);

    // JSON
    JsonReader lector = Json.createReader(new StringReader(str));
    JsonObject respuesta = lector.readObject();

    String base = respuesta.getString("base");
    String fecha = respuesta.getString("date");

    System.out.println("base: " + base+ ", fecha: "+fecha);


    JsonObject ratios = respuesta.getJsonObject("rates");

    // Ratio con dolar
    double dolar_ratio = ratios.getJsonNumber("USD").doubleValue();
    System.out.println("Ratio 1 euro: "+dolar_ratio + " dolares");

  }
}
```

Compilación y ejecución

```
javac -cp javax.json-1.0.jar:. Cambio.java
java  -cp javax.json-1.0.jar:. Cambio
```

Ejemplo servicio web de cambio de divisas: alternativa

Alternativa usando otra clase para invocación API-REST: OkHttpClient

Cambio2.java

```
import java.io.StringReader;
import javax.json.*;
//import java.net.URLEncoder;
//import java.net.URL;
import okhttp3.*;

class Cambio2 {

  public static void main (String[] args) throws Exception  {

    /*
    // URL
    String s = "https://api.exchangeratesapi.io/latest";
    System.out.println("API REST: " + s);

    URL url = new URL(s);

    // lectura respuesta API REST
    Scanner scan = new Scanner(url.openStream());
    String str = new String();
    while (scan.hasNext()) {
      str += scan.nextLine();
    }
    scan.close();
    */


    OkHttpClient client = new OkHttpClient();

    Request request = new Request.Builder()
      .url("https://api.exchangeratesapi.io/latest")
      .get()
      .addHeader("cache-control", "no-cache")
      .build();

    Response response = client.newCall(request).execute();
    String str = response.body().string();



    /* Común a partir de aquí */

    System.out.println("Respuesta: " + str);

    // JSON
    JsonReader lector = Json.createReader(new StringReader(str));
    JsonObject respuesta = lector.readObject();

    String base = respuesta.getString("base");
    String fecha = respuesta.getString("date");

    System.out.println("base: " + base+ ", fecha: "+fecha);


    JsonObject ratios = respuesta.getJsonObject("rates");

    // Ratio con dolar
    double dolar_ratio = ratios.getJsonNumber("USD").doubleValue();
    System.out.println("Ratio 1 euro: "+dolar_ratio + " dolares");

  }
}
```

Compilación y ejecución

```
javac javac -cp javax.json-1.0.jar:okhttp-3.13.1.jar Cambio2.java
java -cp javax.json-1.0.jar:okhttp-3.13.1.jar:okio-1.14.1.jar:. Cambio2 // FALTA ALGO!!! Búsquese...
```

Ejemplo servicio web de Google: *Geocoding*

- Ejemplo con servicio web de Google: Geocoding

```
http://maps.google.com/maps/api/geocode/json?sensor=false&address=Camino%20de%20Vera,%20Valencia,%20Spain
```

- Necesidad de *API Key*:
  - Introducir cuenta Google
  - Conseguir API para *Google Maps Geocoding* en `https://console.developers.google.com/`
    Empiezan a pedir VISA...

```
http://maps.google.com/maps/api/geocode/json?sensor=false&address=Camino%20de%20Vera,%20Valencia,%20Spain&key=API_KEY
```

Geolocalizacion.java

```
import java.io.StringReader;
import javax.json.*;
import java.net.URLEncoder;
import java.net.URL;
import java.util.Scanner;

class GeoLocalizacion {

  public static void geocoding(String addr) throws Exception  {

       // build a URL
    String s = "http://maps.google.com/maps/api/geocode/json?" +
          "sensor=false&address=";
    s += URLEncoder.encode(addr, "UTF-8");

    System.out.println("API REST: " + s);
    URL url = new URL(s);

    // read from the URL
    Scanner scan = new Scanner(url.openStream());
    String str = new String();
    while (scan.hasNext()) {
      str += scan.nextLine();
    }
    scan.close();
    System.out.println("Respuesta: " + str);
    // build a JSON object
    JsonReader lector = Json.createReader(new StringReader(str));
    JsonObject obj = lector.readObject();


    //JsonObject obj = new JsonObject(str);

    if (! obj.getString("status").equals("OK"))
    return;

    // get the first result
    JsonObject res = obj.getJsonArray("results").getJsonObject(0);
    System.out.println(res.getString("formatted_address"));
    JsonObject loc = res.getJsonObject("geometry").getJsonObject("location");
    System.out.println("lat: " + loc.getJsonNumber("lat").doubleValue() +
    ", lng: " + loc.getJsonNumber("lng").doubleValue());
    }

  public static void main (String args[]) throws Exception {
    geocoding("Camino de Vera, Valencia, Spain");
  }
}
```