

Academia

JSON Path

Anonymous · 08/10/2026

Tabla de contenidos

JSON Path	3
<i>Ventajas y desventajas</i>	4
<i>Selector JSONPath</i>	5
<i>Combinaciones con JSONPath</i>	8
<i>¿Por qué usar selectores?</i>	11

Academia » Leer datos de JSON con Selectores » JSON Path

JSON Path

Anonymous · 08/10/2026

JSON Path

Última actualización el 08/10/2026

Academia » Leer datos de JSON con Selectores » JSON Path » Ventajas y desventajas

Ventajas y desventajas

Ventajas y desventajas

Anonymous · 08/10/2026

Ventajas y desventajas

Es verdad, es natural pensar “¿para qué creamos selectores si existe JSONPath?”. Tanto la implementación personalizada con selectores como el uso de JSONPath tienen sus ventajas y desventajas.

Principalmente las ventajas de usar los selectores son:

1. Tenemos control total de la funcionalidad y la podemos adaptar según sea necesario para nuestra plataforma BillMySales.
2. Es muy simple escribir un selector para casos específicos.
3. Nos encanta aprender ~~y reinventar la rueda.~~

Algunas ventajas de usar JSONPath son:

1. Es un estándar de la industria, por lo que muchos usuarios técnicos ya lo conocerán.
2. Tiene más características y funcionalidades para manejar los datos de un JSON.

Nuestros selectores personalizados han sido probados minuciosamente y están funcionando bien para las necesidades específicas de BillMySales. Sin embargo, como nos gusta siempre estar mejorando, hemos añadido compatibilidad con JSONPath. De esta forma, puedes escribir un selector tanto con nuestro formato como con el estándar de la industria.

Última actualización el 08/10/2026

Academia » Leer datos de JSON con Selectores » JSON Path » Selector JSONPath

Selector JSONPath

Selector JSONPath

Anonymous · 08/10/2026

Selector JSONPath

Consideraremos el siguiente JSON como base para todos los ejemplos de esta sección. Este JSON representa datos típicos en un contexto de facturación electrónica (no asociada a ningún país específico):

```
{
  "cliente": {
    "nombre": "Juan Pérez",
    "es_vip": false
  },
  "total": 150000,
  "items": [
    {"nombre": "Producto A", "precio": 50000},
    {"nombre": "Producto B", "precio": 100000}
  ],
  "factura": {
    "detalles": [
      {"producto": "Producto A", "cantidad": 1},
      {"producto": "Producto B", "cantidad": 2}
    ]
  },
  "productos": [
    {"id": "A100", "nombre": "Producto A", "precio": 25000},
    {"id": "B200", "nombre": "Producto B", "precio": 50000}
  ],
  "ventas": [
    {"producto": "Producto 1", "cantidad": 3},
    {"producto": "Producto 3", "cantidad": 6},
    {"producto": "Producto 5", "cantidad": 10}
  ],
  "historial_compras": [
    {"codigo": "C101", "detalle": {"fecha": "2023-01-15"}},
    {"codigo": "C102", "detalle": {"fecha": "2023-03-15"}},
    {"codigo": "C103", "detalle": {"fecha": "2023-05-20"}}
  ],
  "numeros_de_serie": ["NS100", "NS101", "NS102"]
}
```

Ejemplos

1. Acceso Directo a una Clave en Primer Nivel

- **Selector JSONPath:** \$.total
- **Descripción:** Extrae el valor total de la factura.
- **Resultado:** 150000

2. Acceso a un Elemento Anidado

- **Selector JSONPath:** \$.cliente.nombre
- **Descripción:** Obtiene el nombre del cliente de la factura.
- **Resultado:** "Juan Pérez"

3. Acceso a un Elemento de un Arreglo por Índice

- **Selector JSONPath:** \$.items[0].precio
- **Descripción:** Recupera el precio del primer artículo en la factura.
- **Resultado:** 50000

4. Acceso a un Elemento Anidado Dentro de un Arreglo

- **Selector JSONPath:** \$.factura.detalles[1].cantidad
- **Descripción:** Accede a la cantidad del segundo artículo en el detalle de la factura.
- **Resultado:** 2

5. Filtrar Elementos de un Arreglo (por valor existente)

- **Selector JSONPath:** \$.productos[?(@.id == 'A100')].precio
- **Descripción:** Busca y obtiene el precio de un producto específico, identificado por su ID.
- **Resultado:** 25000

6. Filtrar Elementos de un Arreglo (resultado vacío)

- **Selector JSONPath:** \$.productos[?(@.id == 'A999')]
- **Descripción:** Intenta encontrar un producto con un ID que no existe en la lista, esperando un resultado vacío.
- **Resultado:** None

7. Acceso a Todos los Elementos de un Arreglo

- **Selector JSONPath:** \$.numeros_de_serie[*]
- **Descripción:** Obtiene todos los números de serie de los productos.
- **Resultado:** ["NS100", "NS101", "NS102"]

8. Obtener Elementos Basados en una Condición Compleja

- **Selector JSONPath:** \$.ventas[?(@.cantidad > 5)].producto
- **Descripción:** Selecciona los nombres de los productos que se vendieron en cantidades mayores a cinco.
- **Resultado:** ["Producto 3", "Producto 5"]

9. Obtener un Elemento Anidado en un Diccionario dentro de un Arreglo

- **Selector JSONPath:** \$.historial_compras[?(@.codigo == 'C102')].detalle.fecha
- **Descripción:** Encuentra la fecha de una compra específica en el historial de compras, usando su código.
- **Resultado:** "2023-03-15"

10. Acceso Directo a un Elemento Booleano

- **Selector JSONPath:** \$.cliente.es_vip
- **Descripción:** Verifica si el cliente es marcado como VIP.
- **Resultado:** false

Estos ejemplos demuestran la versatilidad y potencia de JSONPath para acceder y manipular datos en un JSON, especialmente útil en escenarios de facturación electrónica, permitiendo desde el acceso a datos simples hasta la extracción basada en condiciones complejas.

Última actualización el 08/10/2026

Combinaciones con JSONPath

Combinaciones con JSONPath

Anonymous · 08/10/2026

Combinaciones con JSONPath

Consideraremos el siguiente JSON como base para todos los ejemplos de esta sección. Este JSON representa datos típicos en un contexto de facturación electrónica (no asociada a ningún país específico):

```
{
  "factura_id": "FAC-00123",
  "cliente": {
    "nombre": "Empresa XYZ",
    "rut": "76.123.456-7"
  },
  "fecha_emision": "2023-10-05",
  "detalles": [
    {"producto": "Laptop Pro", "cantidad": 2, "precio_unitario": 1200000},
    {"producto": "Monitor 24\"", "cantidad": 3, "precio_unitario": 240000}
  ],
  "estado_pago": "pendiente",
  "montos": {
    "neto": 3120000,
    "iva": 592800,
    "total": 3712800
  },
  "historial_pagos": [],
  "es_exportacion": false
}
```

Ejemplos

1. Acceso Directo y Concatenación de Texto

- **Selector:** `"factura_id": "($.factura_id)`
- **Descripción:** Concatena el texto `"factura_id: "` con el ID de la factura.
- **Resultado:** `"factura_id: FAC-00123"`

2. Selección con OR y JSONPath

- **Selector:** `($.estado_pago_inexistente)||($.estado_pago)`

- **Descripción:** Utiliza el operador OR para seleccionar el estado de pago de la factura, proporcionando un valor por defecto si el campo original no existe.
- **Resultado:** "pendiente"

3. Valor por Defecto con OR y JSONPath

- **Selector:** (\$.historial_pagos_inexistente)||"Sin pagos"
- **Descripción:** Ofrece un valor por defecto en caso de que el historial de pagos no esté presente o esté vacío.
- **Resultado:** "Sin pagos"

4. Condición con IF Ternario y JSONPath

- **Selector:** (((\$.estado_pago_inexistente)||(\$.estado_pago)) == "pendiente" ? ("Pendiente") : ("Pagado"))
- **Descripción:** Comprueba si el estado del pago es "pendiente" y muestra "Pendiente" o "Pagado" en consecuencia.
- **Resultado:** "Pendiente"

5. Concatenación con Valor de un Arreglo

- **Selector:** "detalles[0]: "(\$.detalles[0].producto)
- **Descripción:** Concatena la descripción "detalles[0]: " con el nombre del primer producto en los detalles de la factura.
- **Resultado:** "detalles[0]: Laptop Pro"

6. Filtrado de Arreglo y Concatenación

- **Selector:** "Productos con precio > 500000: " + str(\$.detalles[?(@.precio_unitario > 500000)].producto)
- **Descripción:** Concatena una descripción con la lista de productos cuyo precio unitario supera los 500000.
- **Resultado:** "Productos con precio > 500000: ['Laptop Pro']"

7. Acceso a Elemento Anidado y Concatenación

- **Selector:** "Total factura: "(\$.montos.total)
- **Descripción:** Concatena "Total factura: " con el monto total de la factura.
- **Resultado:** "Total factura: 3712800"

8. Filtrado de Arreglo de Diccionarios y Concatenación

- **Selector:** "Cantidad de Laptops: "(\$.detalles[?(@.producto == "Laptop Pro")].cantidad)
- **Descripción:** Concatena "Cantidad de Laptops: " con la cantidad de laptops compradas según los detalles de la factura.
- **Resultado:** "Cantidad de Laptops: 2"

9. Concatenación de un Valor Extraído del Arreglo con Texto

- **Selector:** "Primer producto: "(\$.detalles[0].producto)**
- **Descripción:** Concatena "Primer producto: " con el nombre del primer producto en la factura.
- **Resultado:** "Primer producto: Laptop Pro"

10. Acceder a un Valor Anidado y Concatenar con un Valor de un Arreglo Usando JSONPath

- **Selector:** "Cliente y neto: "(\$.cliente.nombre)" y "(\$.montos.neto)
- **Descripción:** Concatena el nombre del cliente y el monto neto de la factura, separados por " y ".
- **Resultado:** "Cliente y neto: Empresa XYZ y 3120000"

Estos ejemplos demuestran cómo los selectores pueden ser utilizados de manera flexible para extraer y combinar datos de un JSON, especialmente en contextos de facturación electrónica.

Última actualización el 08/10/2026

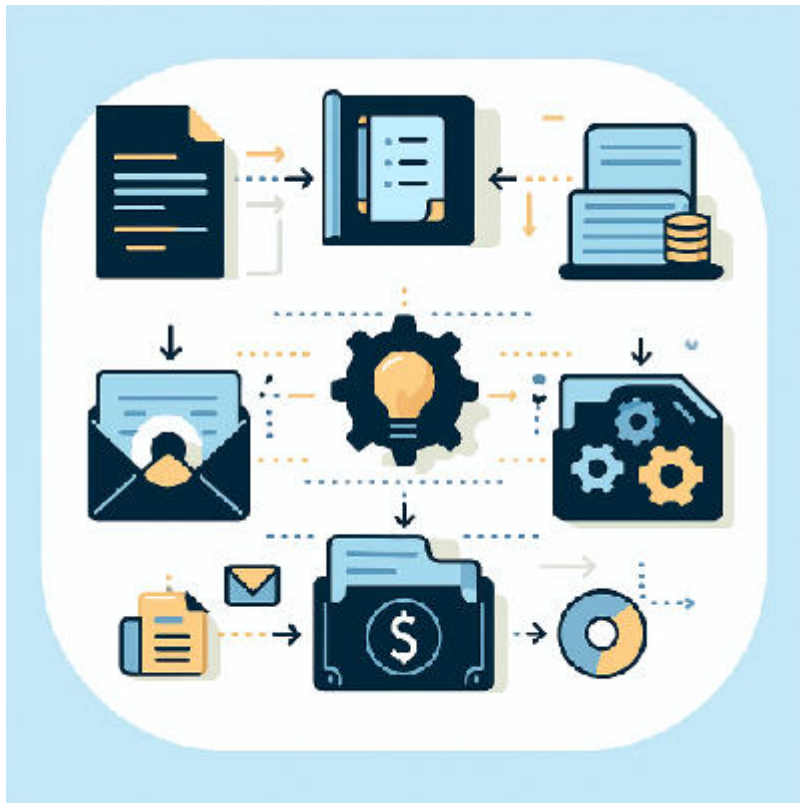
Academia » Leer datos de JSON con Selectores » JSON Path » ¿Por qué usar selectores?

¿Por qué usar selectores?

¿Por qué usar selectores?

Anonymous · 08/10/2026

¿Por qué usar selectores?



En el mundo digital de hoy, la eficiencia y precisión en el manejo de datos son cruciales, especialmente cuando se trata de procesos tan importantes como la facturación electrónica. Aquí es donde entra en juego nuestra innovadora solución que combina la potencia de JSONPath con la flexibilidad de nuestros selectores personalizados.

¿Por qué esta combinación es una revolución en la manipulación de datos JSON? JSONPath por sí solo es una herramienta potente para extraer datos de estructuras JSON, permitiendo acceder a información compleja de manera sencilla y directa. Sin embargo, cuando se enfrenta a necesidades específicas de personalización y manejo de datos en contextos únicos, como la facturación electrónica, su alcance puede resultar limitado.

Aquí es donde nuestros selectores personalizados entran en juego, ofreciendo un nivel de personalización y flexibilidad inigualable. Al combinarlos con JSONPath, se desbloquean posibilidades ilimitadas para manipular, transformar y presentar datos de formas que antes eran imposibles o requerían un esfuerzo considerable de codificación.

Beneficios clave de esta combinación poderosa

- 1. Personalización Avanzada:** Nuestros selectores personalizados permiten ajustes finos en la extracción y presentación de datos, lo que es crucial en procesos de facturación donde cada detalle cuenta.
- 2. Mayor Control y Precisión:** Mientras JSONPath maneja la extracción de datos de manera eficiente, nuestros selectores añaden una capa adicional de control, permitiendo ajustes precisos y condiciones específicas.
- 3. Facilidad de Uso y Aprendizaje:** Aunque JSONPath es simple, la integración con nuestros selectores hace que sea aún más accesible para usuarios no técnicos, permitiendo realizar tareas complejas sin necesidad de conocimientos profundos de programación.
- 4. Soluciones Creativas para Requisitos Únicos:** La combinación de JSONPath con selectores personalizados abre un mundo de soluciones creativas para requisitos específicos que van más allá de la extracción de datos estándar.
- 5. Optimización de Procesos de Negocio:** Esta herramienta no solo mejora la precisión en la facturación electrónica, sino que optimiza los procesos de negocio, ahorrando tiempo y recursos valiosos.

En resumen, al combinar JSONPath con nuestros selectores personalizados, no solo aprovechas las fortalezas de ambos sino que también abres la puerta a un mundo de posibilidades en la gestión de datos. Esta combinación no es simplemente una mejora; es una transformación en la forma en que interactúas y utilizas los datos en tu día a día, especialmente en procesos críticos como la facturación electrónica. ¡Descubre el poder de esta integración y lleva la gestión de tus datos a un nuevo nivel de eficiencia y efectividad!

Última actualización el 08/10/2026