

Sistema de generación de etiquetas: explicación funcional del JSON

Este documento explica cómo interpretar el JSON devuelto por el servicio de *Generador de Etiquetas* que se utiliza para la impresión de etiquetas por artículo. Está diseñado para que cualquier persona pueda comprender:

- La estructura global del JSON
- La función de cada bloque
- Cómo se calculan los datos variables
- Cómo intervienen los *assets* estáticos
- Cómo se generan los *items*
- Cómo se resuelven los valores para cada etiqueta de cada artículo

Un *item* se define como la combinación de:

- Label
- Modelo / Calidad
- Color
- Talla

Cada combinación representa la unidad mínima imprimible.

1. POInformation: información del pedido

El bloque `POInformation` contiene la información básica del artículo incluida en el pedido.

Ejemplo simplificado:

```
"POInformation": { "productionOrderNumber": 80001, "campaign": "V25", "orderQty": 25000, "brand": "ZARA", "section": "CABALLERO", "productType": "ROPA", "model": 3332, "quality": 405, "colors": [...] }
```

1.1 Campos clave

model / quality

Identifican el artículo base del pedido.

Se utilizan cuando un componente variable tiene un `groupKey = MODEL_QUALITY`.

colors

Array con los colores y sus tallas correspondientes.

Ejemplo:

```
{ "color": 712, "sizes": [ { "size": 5, "qty": 1860 }, { "size": 3, "qty": 4560 }, { "size": 4, "qty": 3580 } ] }
```

Cada combinación color-talla generará uno o varios *items* dependiendo del número de etiquetas asignadas.

2. supplier: datos del proveedor

El nodo `supplier` contiene información descriptiva del proveedor.

Este bloque es informativo y no interviene en la resolución de datos de impresión.

3. componentValues: valores de los componentes variables

Este bloque contiene todos los valores variables que pueden aparecer en las etiquetas.

Ejemplo de estructura:

```
{ "groupKey": "MODEL_QUALITY", "name": "Quality", "type": "string", "valueMap": { "3332/405": "405" } }
```

3.1 Significado de campos

- **name:** nombre del componente.

- **type**: tipo de dato (string, number, image_url...).
- **groupKey**: determina cómo se agrupa el valor.
- **valueMap**: diccionario clave-valor que define el valor final para cada combinación aplicable.

3.2 Tipos de groupKey

groupKey	Clave en valueMap	Ejemplo clave
MODEL_QUALITY	"model/quality"	"3332/405"
COLOR	"color"	"712"
SIZE	"size"	"4"
COLOR_SIZE	"color/size"	"712/5"

3.3 Ejemplo interpretado

Si el componente `QR_product` tiene:

```
"groupKey": "COLOR_SIZE", "valueMap": { "401/4": "http://inditex.com/imagen.png" }
```

Significa:

Para cualquier item con color 401 y talla 4, el valor del componente `QR_product` será esa URL.

4. labels: definición de las etiquetas del pedido

El array `labels` indica qué etiquetas deben imprimirse para cada item del pedido.

Ejemplo:

```
{ "reference": "HPZCALL004", "components": [ ... ], "assets": [ ... ], "childrenLabels": [ { "reference": "BLUE_LABEL", ... }, { "reference": "RED_LABEL", ... } ] }
```

4.1 Campos principales

reference

Nombre identificativo de la etiqueta.

components

Lista de componentes variables que se deben resolver para esta etiqueta. Estos nombres deben existir en `componentValues`.

childrenLabels

Etiquetas hijas.

Si se imprime la etiqueta padre, también deben imprimirse todas sus hijas.

5. assets: recursos estáticos asociados a la etiqueta

El nodo `assets` contiene recursos que pertenecen a la etiqueta y que **no dependen del color, talla, modelo o calidad** del artículo.

Son elementos estáticos, propios de la etiqueta en sí.

5.1 Ejemplos típicos de assets estáticos

- Iconos fijos (por ejemplo, el icono del RFID)
- Logos específicos de la etiqueta
- Gráficos o imágenes que forman parte de la maquetación
- Archivos que no cambian entre items

Ejemplo:

```
"assets": [ { "name": "RFID_ICON", "type": "image_url", "url": "https://inditex.com/static/rfid_icon.png" } ]
```

5.2 Relación entre labels y assets

Una etiqueta puede declarar uno o varios assets.

Si una etiqueta usa un asset:

1. Debe aparecer en su lista `assets`.
2. El asset no necesita resolución mediante `componentValues`.
3. El asset es el mismo para todos los items asociados a esa etiqueta.

Durante la generación de items:

- Las columnas de assets deben incluir el valor estático correspondiente.
-

6. Definición formal de un item

Un *item* se genera para cada combinación de:

1. Label
2. Model / Quality
3. Color
4. Size

Ejemplo con:

- 1 modelo
- 3 colores
- 3 tallas por color
- 3 etiquetas (1 padre + 2 hijas)

Total items = $1 \times 3 \times 3 \times 3 = 27$ items.

Cada item puede tener:

- Componentes variables resueltos dinámicamente.
 - Assets estáticos copiados desde la definición de la etiqueta.
-

7. Resolución de valores para cada item

7.1 Componentes variables

Para cada componente:

1. Verificar si la etiqueta requiere ese componente.
2. Localizar el componente en `componentValues` por su campo `name`.
3. Según el `groupBy`, generar la clave de búsqueda:
 - `MODEL_QUALITY` "model/quality"
 - `COLOR` "color"
 - `SIZE` "size"
 - `COLOR_SIZE` "color/size"
4. Buscar el valor en `valueMap`:
 - Si existe usarlo.
 - Si no valor vacío.

7.2 Assets estáticos

Para cada asset:

1. Comprobar si la etiqueta incluye el asset en su nodo `assets`.
2. Si lo incluye usar el valor estático definido (URL, nombre, contenido).

Los assets nunca se resuelven por combinación color-talla.
Son independientes del item.
