<# Guía de Creación de Wizards

Introducción

Un wizard es una secuencia de pasos interactivos que guían al usuario a través de un proceso específico. En esta plataforma, los wizards están compuestos por múltiples elementos que trabajan en conjunto para crear experiencias de usuario dinámicas y personalizables.

📑 Índice

Estructura General de un Wizard

Un wizard está compuesto por los siguientes elementos principales:

- **Nombre**: Identificador único para invocar el wizard desde el frontend usando la *API key* correspondiente
- **Descripción**: Información sobre qué trata el wizard
- **Paso inicial**: Define cuál es el primer paso que se ejecutará
- **Pasos (Steps)**: Secuencia de pasos que componen el wizard
- **Componentes**: Elementos interactivos dentro de cada paso
- **Eventos**: Comportamientos que se ejecutan en respuesta a acciones del usuario
- **Bloques**: Estructura de lógica que define cuándo y cómo ejecutar acciones en respuesta a eventos del wizard
- **Contextos**: Fuentes de datos para las condiciones y acciones
- **Acciones**: Operaciones que se realizan cuando se cumplen las condiciones

Convenciones de Nomenclatura

⚠ **IMPORTANTE**: Todos los nombres deben seguir estas reglas estrictas:

- **Sin caracteres especiales**: No usar símbolos como `@`, `#`, `$`, `%`, etc.
- **Sin espacios**: Usar camelCase o snake_case
- **Únicos**: No repetir nombres dentro del mismo scope, es decir, no pueden existir dos pasos con el mismo nombre o dos componentes con el mismo nombre dentro de un paso
- **Descriptivos**: Usar nombres que indiquen claramente la función

Ejemplos válidos: `userName`, `step1`, `submitButton` Ejemplos inválidos: `user name`, `step-1`, `submit@button`

Componentes

Tipos de Componentes

Campos (Fields)

Los campos son componentes de entrada de datos que permiten al usuario introducir información.

Propiedades comunes del campo

Propiedad	Descripción	Ejemplo
`type`	Tipo de campo (requerido)	`"TEXT"`
`placeholder`	Texto de ayuda	`"Ingrese su nombre"`
`save`	Guardar valor en BD	`true`
`className`	Clases CSS externas	`"w-full p-2"`
`variantName`	Nombre de variante	`"primary-field"`
`disabled`	Está deshabilitado	`true` o `false`
`visible`	Es visible	`true` o `false`

> El placeholder no existe en todos los campos, uno de ellos es el *CHECKBOX*

Tipos de Campos Disponibles

TEXT

Campo de texto simple. Ejemplo:

```
{
  "order": 0,
  "name": "header",
  "componentType": "TEXT",
  "component": {
    "value": "Ejemplo del campo de archivos"
  },
  "className": "col-span-2"
}
```

NUMBER

Campo numérico. Ejemplo:

```
{
  "order": 25,
  "name": "capital_suscripto",
  "componentType": "FIELD",
  "component": {
    "type": "NUMBER",
    "save": false
  },
  "className": "col-span-1"
}
```

DATE

Campo de selección de fecha

Propiedad	Opcional	Descripción	Ejemplo
`format`	X	Formato de la fecha	`"dd-MM-yyyy"`
`closeOnScroll`	X	Cerrar el selector cuando se hace scroll	`true` o `false`
`minDate`	X	Fecha mínima para seleccionar	`"2026-01-14T03:00:00.000Z"`
`maxDate`	X	Fecha máxima para seleccionar	`"2026-01-20T03:00:00.000Z"`
`excludeDates`	X	Fechas excluidas para seleccionar	`["2026-01-06T03:00:00.000Z"]`
`excludeDateIntervals`	X	Intervalos excluidos de fechas para seleccionar	`[{"start": "2026-01-01", "end": "2026-01-05"} ]`
`includeDates`	X	Fechas incluidas para seleccionar	`["2026-01-10","2026-01-11","2026-01-12"]`
`includeDateIntervals`	X	Intervalos de fechas incluidas para seleccionar	`[{"start": "2026-01-01", "end": "2026-01-05"} ]`
`isClearable`	X	Se puede borrar el valor seleccionado	`true` o `false`
`withPortal`	X	Se abre un portal en lugar de un selector de fechas	`true` o `false`
`shouldCloseOnSelect`	X	Se debería cerrar al seleccionar una fecha	`true` o `false`
`filterDate`	X	Función que siga criterio de filtrado	`(date) => { return 0 < date.getDay() && date.getDay() < 6}`
`readOnly`	X	Es de solo lectura	`true` o `false`

Ejemplo:

```
{
  "order": 75,
  "name": "fecha_transferencia",
  "componentType": "FIELD",
  "component": {
    "type": "DATE",
    "placeholder": "DD/MM/YYYY",
    "save": false
  },
  "className": "col-span-1"
}
```

SELECT

Campo de lista desplegable

Propiedad	Opcional	Descripción	Ejemplo
`options`		Arreglo de opciones	`[ {'label': 'Si', 'value': true}, {'label': 'No', 'value': false} ]`
`isMultiple`	X	Soporta multiples opciones	`true` o `false`

Ejemplo:

```
{
  "order": 70,
  "name": "tipo_transferencia",
  "componentType": "FIELD",
  "component": {
    "type": "SELECT",
    "save": false,
    "placeholder": "Seleccioná un tipo",
    "options": [
      {
        "label": "Plazo fijo",
        "value": "plazo_fijo"
      },
      {
        "label": "Depósito",
        "value": "deposito"
      }
    ]
  }
}
```

CHECKBOX

Campo de casilla de verificación. Ejemplo:

```
{
  "order": 15,
  "name": "es_administrador",
  "componentType": "FIELD",
  "component": {
    "type": "CHECKBOX",
    "save": false
  },
  "className": "col-span-1"
}
```

TEXTAREA

Campo de área de texto multilínea. Ejemplo:

```
{
  "order": 25,
  "name": "capital_suscripto",
  "componentType": "FIELD",
  "component": {
    "type": "TEXTAREA",
    "save": false
  },
  "className": "col-span-1"
}
```

FILE

Campo de archivos

Propiedad	Opcional	Descripción	Ejemplo
`restrictions`	X	Restricciones respecto a la cantidad, tamaño (en bytes) y tipo de archivo	`{ "maxFileSize": 512, "maxTotalFileSize": 1024, "maxNumberOfFiles": 3, "minNumberOfFiles": 1, "allowedFileTypes": [".pdf"] }`
`uploadStatus`	N/A	Contadores computados del estado de los archivos	`{"pending": 1, "uploading": 2, "completed": 3, "failed": 0, "total": 6}`

Propiedad	Opcional	Descripción	Ejemplo
`validationStatus`	N/A	Estado de validaciones computado	`{"hasMinFiles": false,"hasMaxFiles": true, "totalSizeValid": true, "allFilesValid": true, "isValid": false }`
`isUploading`	N/A	Valor computado de si se está subiendo al menos un archivo	`true` o `false`
`isComplete`	N/A	Valor computado de si se ha completado la subida de todos los archivos	`true` o `false`
`canProceed`	N/A	Valor computado de si se puede proceder porque se han subido todos los archivos y no hay errores de ningún tipo	`true` o `false`
`hasErrors`	N/A	Valor computado de si se tiene errores	`true` o `false`

Ejemplo:

```
{
  "order": 5,
  "name": "files_uploader",
  "componentType": "FIELD",
  "component": {
    "type": "FILE",
    "save": true,
    "restrictions": {
      "maxNumberOfFiles": 3,
      "minNumberOfFiles": 2,
      "allowedFileTypes": [".pdf"]
    }
  },
  "className": "col-span-2"
}
```

REPEATABLEGROUP

Campo de grupo de componentes repetible

Propiedad	Opcional	Descripción	Ejemplo
`components`		Arreglo de componentes	`[...<aquí componentes>...]`
`componentsClassName`	X	Clases CSS externas para el grupo de componentes	`"grid grid-cols-2 gap-2 border border-gray-300 rounded-lg p-3"`
`minBlocksQuantity`	X	Cantidad mínima de grupos de componentes	`1`
`maxBlocksQuantity`	X	Cantidad máxima de grupos de componentes	`10`
`blocksQuantity`	N/A	Valor computado de la cantidad de grupos de componentes	`2`
`blocks`	N/A	Arreglo computado de bloques de componentes	`[...<aquí grupos de componentes>...]`

Ejemplo:

```
{
  "order": 20,
  "name": "bloques",
  "componentType": "FIELD",
  "component": {
    "type": "REPEATABLE_GROUP",
    "blocksQuantity": 5,
    "minBlocksQuantity": 1,
    "maxBlocksQuantity": 5,
    "className": "flex flex-col gap-2 w-full",
    "componentsClassName": "grid grid-cols-2 gap-2 border border-gray-300 rounded-lg p-3",
    "components": [
      {
        "order": 5,
        "name": "text_example",
        "componentType": "FIELD",
        "component": {
          "type": "TEXT",
          "save": false,
          "placeholder": "Ingrese el nombre"
```

```
        },
        "className": "col-span-2",
        "top": {
          "className": "flex justify-between mb-2",
          "components": [
            {
              "order": 5,
              "name": "header",
              "componentType": "TEXT",
              "component": {
                "value": "Nombre N°{{index}}",
                "contexts": [
                  {
                    "key": "index",
                    "type": "GROUP_INDEX",
                    "expression": "$+1"
                  }
                ]
              },
              "className": "col-span-2"
            },
            {
              "order": 10,
              "name": "eliminar_bloque",
              "componentType": "BUTTON",
              "component": {
                "label": "Eliminar",
                "variantName": "default-button-outline"
              },
              "events": {
                "ON_CLICK": [
                  {
                    "order": 5,
                    "type": "ACTIONS",
                    "actions": [
                      {
                        "type": "REMOVE_GROUP",
                        "order": 5,
                        "groupName": "bloques",
                        "groupIndex": {
                          "key": "index",
                          "type": "GROUP_INDEX"
                        }
                      }
                    ]
                  }
                ]
              }
            }
          ]
        }
      }
    ]
  },
  "className": "col-span-2"
}
```

MAP

Campo de mapa

Propiedad	Opcional	Descripción	Ejemplo
`center`		Es la ubicación sobre la que se mostrará el mapa	`{ "type": "LAT_LNG", "value": [-32.88968899303711, -68.84452462724614] }`
`maxMarkers`	X	Cantidad máxima de marcadores permitida	`1`
`zoom`	X	Valor de zoom inicial	`1`
`scrollWheelZoom`	X	Hacer zoom usando scroll	`true` o `false`

Ejemplo:

```
{
  "order": 70,
  "name": "mapa",
  "componentType": "FIELD",
```

```
  "component": {
    "type": "MAP",
    "maxMarkers": 1,
    "center": {
      "type": "LAT_LNG",
      "value": [-32.88968899303711, -68.84452462724614]
    },
    "save": false
  },
  "className": "col-span-2 w-full"
}
```

OTP

Campo de OTP

Propiedad	Opcional	Descripción	Ejemplo
`numInputs`		Cantidad de inputs	`6`

Ejemplo:

```
{
  "order": 20,
  "name": "codigo_verificacion",
  "componentType": "FIELD",
  "component": {
    "type": "OTP",
    "save": false,
    "numInputs": 6
  },
  "className": "col-span-2"
}
```

VIDEO_CAMERA

Campo de cámara de video. Ejemplo:

```
{
  "order": 20,
  "name": "video",
  "componentType": "FIELD",
  "component": {
    "type": "VIDEO_CAMERA",
    "save": false
  },
  "className": "col-span-2"
}
```

Botones (Buttons)

Los botones permiten al usuario realizar acciones específicas.

Propiedades de Botones

Propiedad	Opcional	Descripción	Ejemplo
`label`	X	Template del botón	`"Enviar {{variable}}"`
`labelContexts`	X	Contextos del template del botón	`[{"key": "variable", "type": "VALUE", "value": "test"}]`
`labelClassName`	X	Texto del botón	`"text-bold"`
`icon`	X	Nombre del ícono (lucide-icons). **Deprecado**, se recomienda insertar iconos en template del label	`"send"`
`className`	X	Clases CSS	`"bg-blue-500 text-white"`
`variantName`	X	Nombre de variante	`"primary-button"`
`disabled`	N/A	Está cargando	`true` o `false`
`loading`	N/A	Está cargando	`true` o `false`

Propiedad	Opcional	Descripción	Ejemplo
`visible`	N/A	Es visible	`true` o `false`

> Tanto disabled, loading como visible son propiedades dinámicas que existen durante el renderizado del componente

Ejemplo:

```
{
  "order": 95,
  "name": "verificar",
  "componentType": "BUTTON",
  "component": {
    "label": "Verificar",
    "variantName": "mxm-next-button"
  },
  "className": "col-span-1 flex justify-end"
}
```

Variantes de Botones

Propiedad	Descripción
default-button	Acción primaria del sistema
alternative-button	Acción secundaria o complementaria
dark-button	Acción fuerte o neutral en contextos oscuros
light-button	Acción neutra y discreta
green-button	Confirmación positiva o éxito
red-button	Acción destructiva o peligrosa
yellow-button	Acción de advertencia o intermedia
purple-button	Acción especial o diferenciada

> Las variantes de botones tiene sus respectivas versiones outline. Ejemplo: `purple-button-outline`

Etiquetas (Labels)

Las etiquetas muestran texto estático o dinámico.

Propiedades de Etiquetas

Propiedad	Opcional	Descripción	Ejemplo
`value`		Template de la etiqueta	`"Nombre del usuario {{index}}"`
`contexts`	X	Contextos para el template de la etiqueta	`[{"key": "index", "type": "VALUE", "value": 1}]`
`className`	X	Clases CSS	`"text-lg font-bold"`
`variantName`	X	Nombre de variante	`"title-label"`
`visible`	X	Es visible	`true` o `false`

Ejemplo:

```
{
  "order": 5,
  "name": "etiqueta_instancia_tramite",
  "componentType": "LABEL",
  "component": {
    "value": "Trámite",
    "variantName": "mxm-label"
  }
}
```

Textos (Texts)

Similar a las etiquetas pero con soporte completo para contextos dinámicos.

Propiedades de Textos

Propiedad	Opcional	Descripción	Ejemplo
`value`		Texto base	`"Bienvenido {{userName}}"`
`contexts`	X	Contextos para texto dinámico	`[{"key": "userName", "type": "VALUE", "value": "Octavio"}]`
`className`	X	Clases CSS	`"text-gray-600"`
`variantName`	X	Nombre de variante	`"info-text"`
`visible`	X	Es visible	`true` o `false`

Ejemplo:

```
{
  "order": 5,
  "name": "titulo",
  "componentType": "TEXT",
  "component": {
    "value": "Atención: Cancelación del trámite {{current_date}}",
    "contexts": [
      {
        "key": "current_date",
        "type": "CURRENT_DATE"
      }
    ]
    "variantName": "mxm-title"
  },
  "className": "col-span-2"
}
```

Alertas (Alerts)

Muestran mensajes informativos, de advertencia o error.

Propiedades de Alertas

Propiedad	Opcional	Descripción	Ejemplo
`value`		Template de la alerta	`"El campo debería tener {{cant_archivos}} archivos"`
`contexts`	X	Contextos para el template	`[{"key": "cant_archivos", "type": "VALUE", "value": 3]`
`className`	X	Clases CSS	`"bg-red-100 text-red-800"`
`variantName`	X	Nombre de variante	`"error-alert"`
`visible`	X	Es visible	`true` o `false`

Variantes de Alertas

Propiedad	Descripción
`info-alert`	Informar al usuario sobre un estado neutro o informativo
`success-alert`	Confirmar que una acción se completó correctamente
`warning-alert`	Advertir sobre una situación que requiere atención, pero no es un error
`error-alert`	Informar sobre un fallo o problema que impide completar una acción
`default-alert`	Mostrar mensajes generales sin connotación específica

Ejemplo:

```
{
  "order": 5,
  "name": "error-top",
  "componentType": "ALERT",
  "component": {
    "value": "Servicio no disponible en este momento. Por favor, inténtelo más tarde. {{current_date}}",
    "contexts": [
      {
        "key": "current_date",
        "type": "CURRENT_DATE"
      }
    ]
  }
}
```

Spinners

Indicadores de carga o procesamiento.

Propiedades de Spinners

Propiedad	Opcional	Descripción	Ejemplo
`className`	X	Clases CSS	`"text-blue-500"`
`variantName`	X	Nombre de variante	`"loading-spinner"`
`size`	X	Tamaño en píxeles	`50`
`strokeWidth`	X	Grosor de la línea	`5`
`visible`	X	Es visible	`true` o `false`

Ejemplo:

```
{
  "order": 5,
  "name": "loader",
  "componentType": "SPINNER",
  "component": {
    "className": "text-purple-800/100",
    "size": 50,
    "strokeWidth": 2
  },
  "className": "flex items-center justify-center"
}
```

Componentes Anidados

Los componentes pueden tener componentes hijos en cuatro posiciones:

- **TOP**: Arriba del componente
- **BOTTOM**: Debajo del componente
- **LEFT**: A la izquierda del componente
- **RIGHT**: A la derecha del componente

> ⚠ **Limitación**: Los componentes anidados solo pueden tener una profundidad de 1 nivel. Los hijos no pueden tener más hijos.

> 📝 **Nota**: El arreglo de hijos pueden ser estilizados con la propiedad `className`

Ejemplo de Componente con Hijos

```
{
  "order": 10,
  "name": "nuevo_tramite",
  "componentType": "FIELD",
  "component": {
    "type": "SELECT",
    "save": false,
    "placeholder": "Elegí qué hacer",
    "options": []
  },
  "className": "col-span-2",
  "top": {
    "className": "m-2" // <- útil para utilizarlo con grid o flex
    "components": [
      {
        "order": 5,
        "name": "etiqueta_tipo_entidad_juridica",
        "componentType": "LABEL",
        "component": {
          "value": "¿Qué desea hacer?",
          "variantName": "mxm-label"
        }
      }
    ]
  }
}
```

Eventos

Un evento en JavaScript es una señal del sistema que anuncia que algo relevante ocurrió, y que permite que el código reaccione a ese cambio sin tener que estar preguntando todo el tiempo. Cada componente tiene una serie de eventos de acuerdo a su naturaleza.

Descripción de Eventos

Evento	Cuándo se Dispara
`ON_CLICK`	Al hacer clic en el componente
`ON_CHANGE`	Al modificar el valor de un campo
`ON_FOCUS`	Al hacer foco en el componente
`ON_BLUR`	Al salir del componente
`ON_MOUNTED`	Al cargar el componente en la página
`ON_UNMOUNTED`	Al quitar el componente de la página
`ON_CALENDAR_OPEN`	Al abrir el selector de fecha/hora
`ON_CALENDAR_CLOSE`	Al cerrar el selector de fecha/hora
`ON_WINDOW_FOCUS`	Al enfocar la ventana del navegador
`ON_WINDOW_BLUR`	Al desenfocar la ventana del navegador
`ON_WINDOW_BEFORE_UNLOAD`	Antes de cerrar o recargar la página
`RESTRICTION_FAILED`	A la falla de validaciones de restricciones de subida del archivo

Eventos Soportados por Componente

Evento	DateField	CheckboxField	TextAreaField	TextField	NumberField	SelectField	FileField	RepeatableGroupField
`ON_CHANGE`	✓	✓	✓	✓	✓	✓	✓	✓
`ON_FOCUS`	✓	✓	✓	✓	✓	✓		
`ON_BLUR`	✓	✓	✓	✓	✓	✓		
`ON_MOUNTED`	✓	✓	✓	✓	✓	✓	✓	✓
`ON_UNMOUNTED`	✓	✓	✓	✓	✓	✓	✓	✓
`ON_CLICK`								
`ON_CALENDAR_CLOSE`	✓							
`ON_CALENDAR_OPEN`	✓							
`ON_WINDOW_FOCUS`								
`ON_WINDOW_BLUR`								
`ON_WINDOW_BEFORE_UNLOAD`								
`RESTRICTION_FAILED`							✓	

Acciones Remotas

Las acciones remotas permiten realizar peticiones HTTP a servicios externos.

Propiedades de Acciones Remotas

Propiedad	Tipo	Descripción	Ejemplo
`name`	string	Nombre único de la acción	`"getUserData"`
`contentType`	CONTENT_TYPE	Tipo de contenido	`"JSON"`
`method`	HTTP_METHOD	Método HTTP	`"GET"`
`url`	string	URL del servicio	`"https://api.example.com/users"`
`body`	string	Cuerpo de la petición	`"{\"name\": \"{{userName}}\"}"`
`params`	string	Parámetros de URL	`"{\"id\": \"{{userId}}\"}"`
`headers`	string	Cabeceras HTTP	`"{\"Authorization\": \"Bearer {{token}}\"}"`
`expression`	string	Expresión para procesar respuesta	`"{{data.user.name}}"`
`certificateName`	string	Nombre del certificado	`"my-certificate"`

Tipos de Contenido (CONTENT_TYPE)

Tipo	Descripción	Uso
JSON	application/json	APIs REST estándar
URL_ENCODING	application/x-www-form-urlencoded	Formularios HTML
FORM_DATA	multipart/form-data	Subida de archivos
TEXT	text/plain	Texto plano

Métodos HTTP Soportados

Método	Descripción
GET	Obtener datos
POST	Crear/enviar datos
PUT	Actualizar datos
DELETE	Eliminar datos
PATCH	Actualizar parcialmente

Ejemplo de Acción Remota

```
{
  "name": "getUserProfile",
  "contentType": "JSON",
  "method": "GET",
  "url": "https://api.example.com/users/{{userId}}",
  "headers": "{\"Authorization\": \"Bearer {{authToken}}\"}",
  "expression": "{{data.profile}}"
}
```

Interpolación con LiquidJS

Las acciones remotas soportan interpolación usando la sintaxis de LiquidJS. Quien invoque esta acción remota deberá pasar los contextos necesarios que permitan reemplazar las variables del template. Ejemplo para llamar a este caso particular

```
{
  "name": "createUser",
  "contentType": "JSON",
  "method": "POST",
  "url": "{{domain}}/users",
  "body": "{\"name\": \"{{userName}}\", \"email\": \"{{userEmail}}\"}",
  "headers": "{\"Content-Type\": \"application/json\", \"X-API-Key\": \"{{apiKey}}\"}"
}
```

Expresiones JSONata

Las expresiones JSONata permiten navegar y manipular datos JSON de forma intuitiva. Recomendamos encarecidamente que pruebe el playground https://try.jsonata.org/

Ejemplos introductorios

Los siguientes ejemplos los podrá pegar directamente enn el playground para obtener ejemplos concretos para probar.

- Acceder a una propiedad

```
Account.'Account Name'
```

- Concatenar

```
'Hello ' & Account.'Account Name'
```

- Filtrar por condición

```
Account.Order[OrderID='order103']
```

> **IMPORTANTE**: Observe que no devuelve un arreglo con los elementos que satisfacen la condición, si solamente uno lo hace devuelve uno solo por lo que tendrá que incrustar la expresión entre corchetes: *[<expresión>]*. Esto es importante porque puede tener problemas al querer setear las opciones de un componente de tipo SELECT con algo que no es un arreglo.

- Obtener fecha actual

`$now()`

- Castear tipos de datos

```
{ 'producto_como_cadena': $string(Account.Order[0].Product[0].ProductID), 'sku_como_numero': $number(Account.Order[0].Product[0].SKU) }
```

- Dividir cadena

```
$split($now(),"T")[0]
```

- Formatear fecha

```
$fromMillis($toMillis($now()),'[D01]-[M01]-[Y0001]')
```

- Expresiones regulares

```
$match('hola paroz',/paroz/)
```

- Existe propiedad en objeto

```
$exists(Account2)
```

- Longitud de una cadena

```
$length(Account.'Account Name')
```

- Cantidad elementos de un arreglo

```
$count(Account.Order)
```

- Unir elementos del arreglo

```
$join(Account.Order.OrderID,", ")
```

- Map `$map(Account.Order, function($v, $i) { {'index': $i, 'order': $v.OrderID} })`
- Agregación

```
$sum(Account.Order.Product.(Price * Quantity))
```

- Formatear números

```
$formatNumber(Account.Order[0].Product[0].Price*100, '$#,###.00')
```

Recomendación:

- Si el documento es un arreglo debe usar `*` y si es un objeto usar `$` para referenciar al documento completo. En la mayoria de los casos puede usar indistintamente uno u otro pero recomendamos usar uno u otro de acuerdo al caso.

Ejemplo 1:

```
{
  "a": 1,
  "b": 2,
  "c": 3
}
```

Expresión	Resultado
`$`	`{"a": 1, "b": 2, "c": 3}`
`$[0]`	(**no usar**)
`*`	`[1,2,3]` (**evitar**)
`*[0]`	`1` (**evitar**)

Ejemplo 2:

```
[1, 2, 3]
```

Expresión	Resultado
`$`	`[1, 2, 3]` (**evitar**)
`$[0]`	`1` (**evitar**)
`*`	`[1, 2, 3]`

Expresión	Resultado
`*[0]`	`1`

Contextos

Los contextos son fuentes de datos que permiten acceder a información para condiciones y acciones. Pueden ser anidados para crear expresiones complejas.

Tipos de Contextos

CURRENT_STEP_VALUES

Valores del paso actual.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`encodingType`	X	Tipo de condificación (solo para FILES)	`"BASE64"`, `"BINARY"`
`expression`	X	Expresión de JSONata	`"options[value='{{value}}']"`
`contexts`	X	Arreglo de contextos	`[{ "key": "value", "type": "TEMPLATE", "template": "null"}]`

Ejemplo:

```
{
  "key": "nuevo_tramite",
  "type": "CURRENT_STEP_VALUES",
  "expression": "nuevo_tramite.value"
}
```

OTHER_STEP_VALUE

Valores de otro paso.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`stepName`		Nombre del paso	`"pasoA"`
`encodingType`	X	Tipo de codificación (solo para *FILES*)	`"BASE64"`, `"BINARY"`
`expression`	X	Expresión de JSONata	`"options[value='{{value}}']"`
`contexts`	X	Arreglo de contextos	`[{ "key": "value", "type": "TEMPLATE", "template": "null"}]`

Ejemplo:

```
{
  "key": "businessName",
  "type": "OTHER_STEP_VALUE",
  "stepName": "crear_homonimia",
  "expression": "homonimia"
}
```

EXTERNAL_METADATA

Metadatos externos. Es un objeto que le pasa el instanciador del componente de la libreria.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`expression`	X	Expresión de JSONata	`"options[value='{{value}}']"`
`contexts`	X	Arreglo de contextos	`[{ "key": "value", "type": "TEMPLATE", "template": "null"}]`

Ejemplo:

```
{
  "key": "procedureId",
  "type": "EXTERNAL_METADATA",
```

```
  "expression": "tramiteId"
}
```

VALUE

Valor estático, puede ser primitivo o un objeto. **Deprecado, usar TEMPLATE**.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`value`		Valor	`null`, `"Hola Mundo"`, `1.3`, `true`, `["Hola", "Mundo"]`, `{"1":"Hola", "2":true }`
`expression`	X	Expresión de JSONata	`"options[value='{{value}}']"`
`contexts`	X	Arreglo de contextos	`[{ "key": "value", "type": "TEMPLATE", "template": "null"}]`

Ejemplo:

```
{
  "key": "paso",
  "type": "VALUE",
  "value": "esperar_pago"
}
```

> Algunos ejemplos de **value** pueden ser `"ejemplo"`, `true`, `34`, `{ "value": true }`

LOCAL

Contexto local del wizard. Es una variable instanciada por el usuario que puede ser un valor primitivo u objeto.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`contextName`		Template del nombre del contexto	`"user.{{index}}"`
`nameContexts`	X	Arreglo de contextos del template	`[{ "key": "value", "type": "TEMPLATE", "template": "null"}]`
`expression`	X	Expresión de JSONata	`"options[value='{{value}}']"`
`contexts`	X	Arreglo de contextos	`[{ "key": "value", "type": "TEMPLATE", "template": "null"}]`

Ejemplo:

```
{
  "key": "url_pagar",
  "type": "LOCAL",
  "contextName": "datos_pagar",
  "expression": "data.url"
}
```

REMOTE_ACTION

Resultado de acción remota. Permite realizar peticiones HTTP.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`actionName`		Nombre de la acción	`"obtenerUsuarios"`
`expression`	X	Expresión de JSONata	`"options[value='{{value}}']"`
`contexts`	X	Arreglo de contextos	`[{ "key": "value", "type": "TEMPLATE", "template": "null"}]`

> El atributo `contexts` define tanto los contextos necesarios para la ejecución de la acción remota como aquellos requeridos para la evaluación de la expresión `expression`, en caso de ser utilizados.

Ejemplo:

```
{
  "key": "salario_minimo",
```

```
    "type": "REMOTE_ACTION",
    "actionName": "ObtenerSMVM",
    "contexts": [
      {
        "key": "keycloakToken",
        "type": "EXTERNAL_METADATA",
        "expression": "token"
      }
    ]
  }
```

> Las acciones remotas tiene sus limitaciones. La limitación más importante es que solamente soporta el protocolo HTTP 1.1

NAVIGATION_HISTORY

Historial de navegación.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`expression`	X	Expresión de JSONata	`"*[{{step_number}}]"`
`contexts`	X	Arreglo de contextos	`[{ "key": "step_number", "type": "TEMPLATE", "template": "-2"}]`

Ejemplo:

```
{
  "key": "penultimo_paso",
  "type": "NAVIGATION_HISTORY",
  "expression": "*[-2]"
}
```

> El contexto de arriba se queda con el nombre del penúltimo paso. Sin expresión devuelve el arreglo de nombres de pasos ordenados del más antiguo hasta el paso actual

COMPONENT_VALUES

Estado del componente. Ejemplo:

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`componentName`		Nombre del componente	`"dni_usuario"`
`expression`	X	Expresión de JSONata	`"paises[value='{{value}}']"`
`contexts`	X	Arreglo de contextos	`[{ "key": "value", "type": "TEMPLATE", "template": "Argentina"}]`

Ejemplo:

```
{
  "key": "boton_deshabilitado",
  "type": "COMPONENT_VALUES",
  "componentName": "siguiente",
  "expression": "disabled"
}
```

> El contexto de arriba devuelve un booleano que indica si el botón está o no deshabilitado

TEMPLATE

Expresión de jsonata. Quizás es el contexto más importante porque abarca incluso a otro tipo de contextos (sustituto directo del contexto *VALUE*, *CURRENT_DATE*). Todos los contextos permiten anidamiento pero este es especialmente útil para condicionales, se profundizará dicha utilidad mas adelante. Ejemplo:

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`template`		Expresión de JSONata	`"paises[value='{{value}}']"`
`contexts`	X	Arreglo de contextos	`[{ "key": "value", "type": "TEMPLATE", "template": "Argentina"}]`

Ejemplo:

```
{
  "key": "description",
  "type": "TEMPLATE",
  "template": "'COMERCIAL'",
}
```

FOREACH

Valor iterado actual. Cuando se utiliza un bloque iterable *FOREACH* todos los bloques hijos tiene acceso a este contexto, contiene el valor iterado y el índice.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`forEachName`		Nombre del iterador	`"loop1"`
`expression`	X	Expresión de JSONata	`"value.paises[value='{{value}}']"`
`contexts`	X	Arreglo de contextos	`[{ "key": "value", "type": "TEMPLATE", "template": "Argentina"}]`

Ejemplo:

```
{
  "key": "index",
  "type": "FOREACH",
  "forEachName": "loop1", // Nombre del iterador
  "expression": "index" // Si quisiera ingresar al valor iterado usaría value
}
```

CURRENT_DATE

Fecha y hora actual del sistema. **Deprecado, usar TEMPLATE**. Ejemplo:

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`expression`	X	Expresión de JSONata	`"$fromMillis($toMillis($now()),'{{formato}}')"`
`contexts`	X	Arreglo de contextos	`[{ "key": "formato", "type": "TEMPLATE", "template": "'[D01]-[M01]-[Y0001]'"}]`

Ejemplo:

```
{
  "key": "updatedAt",
  "type": "CURRENT_DATE"
}
```

GROUP_INDEX

Número de indice del grupo actual.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`expression`	X	Expresión de JSONata	`"$ + {{base}}"`
`contexts`	X	Arreglo de contextos	`[{ "key": "base", "type": "TEMPLATE", "template": "1"}]`

Ejemplo:

```
{
  "key": "index",
  "type": "GROUP_INDEX",
  "expression": "$+1"
}
```

IS_MOBILE

Booleano que indica si se está ejecutando en un dispositivo móvil.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`expression`	X	Expresión de JSONata	`"$ and {{tiene_permiso}}"`
`contexts`	X	Arreglo de contextos	`[{ "key": "tiene_permiso", "type": "TEMPLATE", "template": "false"}]`

Ejemplo:

```
{
  "key": "es_mobile",
  "type": "IS_MOBILE"
}
```

WIZARD_DRAFT_ID

UUID de la instancia de wizard creada. Ejemplo:

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`expression`	X	Expresión de JSONata	`"'pfb-' & $ & '-' & '{{cuil}}'"`
`contexts`	X	Arreglo de contextos	`[{ "key": "cuil", "type": "TEMPLATE", "template": "XX-YYYYYYYY-ZZ"}]`

Ejemplo:

```
{
  "key": "wizardDraftId",
  "type": "WIZARD_DRAFT_ID"
}
```

SEARCH_PARAMS

Objeto de parámetros del URL.

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`expression`	X	Expresión de JSONata	`"q"`
`contexts`	X	Arreglo de contextos	`[{ "key": "cuil", "type": "TEMPLATE", "template": "XX-YYYYYYYY-ZZ"}]`

Ejemplo:

```
{
  "key": "turnId",
  "type": "SEARCH_PARAMS",
  "expression": "q"
}
```

> Si el URL fuese `https://google.com?q=hola` entonces el contexto devolvería `hola`

Contextos anidados

Los contextos anidados son contextos que dependen de otros contextos para evaluar una expresion de jsonata. El ejemplo típico es el caso de las acciones remotas, en la mayoria de casos se necesitan pasar variables a la petición para poder ejecutarla

Ejemplo 1: Accion remota

```
{
  "key": "datos_socio",
  "type": "REMOTE_ACTION",
  "actionName": "BuscarDatosSocio",
```

```
  "contexts": [
    {
      "key": "cuil",
      "type": "EXTERNAL_METADATA",
      "expression": "cuil"
    }
  ]
}
```

Ejemplo 2:

Las expresiones de jsonata pueden recibir variables que serían otros contextos tantos como se quieran.

```
{
  "key": "description",
  "type": "TEMPLATE",
  "expression": "100 + {{capital}}",
  "contexts": [
    {
      "key": "capital",
      "type": "VALUE",
      "value": 100
    }
  ]
}
```

Condicionales

Los condicionales determinan cuándo se ejecutan las acciones y se construyen con los contextos, en particular con el TEMPLATE como raíz.

Acciones

Las acciones se ejecutan cuando se cumplen las condiciones. Todas las acciones tienen un `order` para definir la secuencia de ejecución.

Tipos de Acciones

SET_VALUE

Establecer valor en componente.

Propiedades

Propiedad	Descripción	Ejemplo
`targetName`	Nombre del campo	`"user.{{index}}"`
`targetContexts`	Arreglo de contextos del nombre del campo	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`
`context`	Valor de un contexto que se quiere establecer	`{ "key": "value", "type": "TEMPLATE", "template": "null"}`

Ejemplo:

```
{
  "type": "SET_VALUE",
  "order": 25,
  "targetName": "subentidad",
  "context": {
    "key": "value",
    "type": "TEMPLATE",
    "template": "null"
  }
}
```

UPDATE_COMPONENT

Actualizar propiedad de componente.

Propiedades

Propiedad	Descripción	Ejemplo
`targetName`	Nombre del componente	`"user.{{index}}"`

Propiedad	Descripción	Ejemplo
`targetContexts`	Arreglo de contextos del nombre del componente	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`
`targetProp`	Propiedad del componete no computable	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`
`context`	Valor de un contexto que se quiere establecer	`{ "key": "value", "type": "TEMPLATE", "template": "null"}`

> Las propiedades computadas no deberían bajo ninguna cirscunstancia ser actualizadas forzosamente

Ejemplo:

```
{
  "type": "UPDATE_COMPONENT",
  "order": 10,
  "targetName": "subentidad",
  "targetProp": "visible",
  "context": {
    "key": "visible",
    "type": "TEMPLATE",
    "template": "false"
  }
}
```

GO_TO_STEP

Navegar a otro paso.

Propiedades

Propiedad	Descripción	Ejemplo
`stepName`	Valor de un contexto que se quiere establecer	`{ "key": "value", "type": "TEMPLATE", "template": "null"}`

Ejemplo:

```
{
  "type": "GO_TO_STEP",
  "order": 10,
  "stepName": {
    "key": "paso",
    "type": "TEMPLATE",
    "template": "'switchA'"
  }
}
```

FINISH_WIZARD

Al instanciar el componente de la librería, puede definirse un callback opcional onCompleted, el cual se ejecuta automáticamente cuando se dispara esta acción.
Ejemplo:

```
{
  "order": 5,
  "type": "FINISH_WIZARD"
}
```

INIT_STEP_VALUES

Cuando se regresa a un paso previamente visitado, los valores pueden mantenerse inicializados exactamente como estaban antes de abandonarlo. Ejemplo:

```
{
  "order": 1,
  "type": "INIT_STEP_VALUES"
}
```

ADD_ERROR

Agregar un error a la bolsa de errores de un campo.

Propiedades

Propiedad	Descripción	Ejemplo
`fieldName`	Nombre del campo	`"user.{{index}}"`
`fieldContexts`	Arreglo de contextos del nombre del campo	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`
`errorMessage`	Template del mensaje de error	`"No deberia ser inferior a {{cant_minima}}"`
`context`	Valor de un contexto que se quiere establecer	`{ "key": "cant_minima", "type": "TEMPLATE", "template": "1"}`

Ejemplo:

```
{
  "type": "ADD_ERROR",
  "order": 5,
  "fieldName": "capital_suscripto",
  "errorMessage": "Este campo debe ser mayor o igual a ${{dos_salarios_minimos}}",
  "contexts": [
    {
      "key": "dos_salarios_minimos",
      "type": "LOCAL",
      "contextName": "salario_minimo",
      "expression": "$formatNumber($number(data.results.monto)*2, '#,###.00')"
    }
  ]
}
```

> Los mensajes de errores son reactivos, es decir, si el mensaje de error tiene contextos que pueden cambiar el mensaje de error también lo hará.

CLEAR_ERRORS

Limpiar errores de campo

Propiedades

Propiedad	Descripción	Ejemplo
`fieldName`	Nombre del campo	`"user.{{index}}"`
`fieldContexts`	Arreglo de contextos del nombre del campo	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`

Ejemplo:

```
{
  "type": "CLEAR_ERRORS",
  "order": 1,
  "fieldName": "capital_suscripto"
}
```

SET_LOCAL_CONTEXT

Crear o sobrescribir un contexto local

Propiedades

Propiedad	Descripción	Ejemplo
`contextName`	Nombre del contexto	`"user.{{index}}"`
`nameContexts`	Arreglo de contextos del nombre del contexto	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`
`context`	Valor de un contexto que se quiere establecer	`{ "key": "cant_minima", "type": "TEMPLATE", "template": "1"}`

Ejemplo:

```
{
  "order": 5,
  "type": "SET_LOCAL_CONTEXT",
  "contextName": "socios.{{index}}",
  "nameContexts": [
    {
      "key": "index",
      "type": "GROUP_INDEX"
    }
  ],
```

```
  "context": {
    "key": "socio",
    "type": "REMOTE_ACTION",
    "actionName": "BuscarSocio",
    "contexts": [
      {
        "key": "cuil",
        "type": "CURRENT_STEP_VALUES",
        "expression": "cuil",
      }
    ]
  }
}
```

> Los contextos locales admiten nombres dinámicos, lo que los convierte en una opción ideal para almacenar resultados de acciones remotas, así como para iterar y persistir valores asociados a estructuras repetibles.

REDIRECT

Redirigir a URL

Propiedades

Propiedad	Descripción	Ejemplo
`url`	Template del url	`"https://example.com/user?id={{index}}"`
`contexts`	Arreglo de contextos del nombre del contexto	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`

Ejemplo:

```
{
  "order": 5,
  "type": "REDIRECT",
  "contextName": "https://example.com/user?id={{index}}",
  "contexts": [
    {
      "key": "index",
      "type": "GROUP_INDEX"
    }
  ]
}
```

OPEN_WINDOW

Abrir pestaña nueva

Propiedades

Propiedad	Descripción	Ejemplo
`url`	Template del url	`"https://example.com/user?id={{index}}"`
`contexts`	Arreglo de contextos del nombre del contexto	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`

Ejemplo:

```
{
  "url": "{{url_pagar}}",
  "type": "OPEN_WINDOW",
  "order": 5,
  "contexts": [
    {
      "key": "url_pagar",
      "type": "LOCAL",
      "contextName": "datos_pagar",
      "expression": "data.url"
    }
  ]
}
```

SCHEDULE_TASK

Programar tarea

Propiedades

Propiedad	Descripción	Ejemplo
`name`	Nombre de la tarea programada	`"verificar_pago"`
`delays`	Arreglo de delays que debe esperar antes de ejecutar otra vez los bloques	`[0,10,20,50,100,100]`
`blocks`	Arreglo de bloques	`-`

Ejemplo:

```
{
  "order": 5,
  "type": "CONDITIONAL",
  "conditions": true,
  "actions": [
    {
      "order": 5,
      "name": "esperar_pago",
      "delays": [
        0,
        5000,
        10000,
        15000,
        20000,
        20000,
        20000
      ],
      "type": "SCHEDULE_TASK",
      "blocks": [
        {
          "order": 5,
          "type": "ACTIONS",
          "actions": [
            {
              "type": "CONSOLE_LOG",
              "order": 5,
              "message": "Verificando estado del pago..."
            }
          ]
        }
      ]
    }
  ]
}
```

> Esta acción es especialmente útil para implementar polling controlado, ya que permite ejecutar bloques de acciones en una secuencia finita de reintentos, definidos explícitamente mediante una lista limitada de delays.

> Una tarea no admite ejecuciones simultáneas; al reprogramarla, se cancela la ejecución anterior y se reinicia.

KILL_TASK

Permite cancelar una tarea

Propiedades

Propiedad	Descripción	Ejemplo
`name`	Nombre de la tarea programada	`"verificar_pago"`

Ejemplo:

```
{
  "order": 5,
  "type": "KILL_TASK",
  "name": "esperar_pago"
}
```

CONFIRM_LEAVE_PAGE

Confirmar salida de página

```
{
  "order": 5,
  "type": "CONFIRM_LEAVE_PAGE"
}
```

> Esta acción está diseñada exclusivamente para ser utilizada dentro del evento ON_WINDOW_BEFORE_UNLOAD, donde el navegador permite interceptar la salida de la página.

CONSOLE_LOG

Imprime por la consola (ideal para debugear contextos). Solamente se imprimen en modo debugger (al instanciar la libreria de componentes)

Propiedades

Propiedad	Descripción	Ejemplo
`message`	Template de lo que imprimirá por consola	`"El indice es {{index}}"`
`contexts`	Arreglo de contextos	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`

Ejemplo:

```
{
  "type": "CONSOLE_LOG",
  "order": 5,
  "message": "Propiedades del uploader: {{fileUploaderProps | prettyjson}}",
  "contexts": [
    {
      "key": "fileUploaderProps",
      "type": "COMPONENT_VALUES",
      "componentName": "files_uploader",
      "expression": "restrictions"
    }
  ]
}
```

> Cuando se imprimen objetos complejos, es recomendable aplicar un formatter (por ejemplo, prettyjson) para garantizar una salida legible y facilitar el proceso de depuración.

EXECUTE_FORM

Ejecutar un formulario

Propiedades

Propiedad	Descripción	Ejemplo
`target`	Dónde abrir la URL	`"_self"`, `"_blank"`, `"_parent"`, `"_top"`
`method`	Método HTTP	`"GET"`, `"POST"`, `"PUT"`, `"DELETE"`, `"PATCH"`
`url`	Template de la URL	`{ "key": "url", "type": "TEMPLATE", "template": "https://example.com"`
`inputs`	Arreglo de contextos	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`

Ejemplo:

```
{
  "order": 1,
  "type": "EXECUTE_FORM",
  "target": "_self",
  "method": "POST",
  "url": {
    "key": "url",
    "type": "LOCAL",
    "contextName": "resGenerarLink",
    "expression": "data.url"
  },
  "inputs": [
    {
      "key": "operacion",
      "type": "LOCAL",
      "contextName": "resGenerarLink",
      "expression": "data.checkout.operacion"
    }
```

```
    ]
  }
```

> Esta acción es especialmente útil para realizar redirecciones mediante el envío de parámetros al servidor usando un formulario HTML tradicional (POST), permitiendo que el servidor procese la información y devuelva una nueva página con el HTML completamente renderizado, de forma similar al comportamiento clásico de aplicaciones PHP.

ADD_GROUP

Agregar un grupo de componentes dentro del componente *REPEATABLEGROUP* en la propiedad `blocks`

Propiedades

Propiedad	Descripción	Ejemplo
`groupName`	Nombre del *REPEATABLEGROUP*	`"usuarios"`
`quantity`	Cantidad de grupos de componetes a agregar	`2`

Ejemplo:

```
{
  "type": "ADD_GROUP",
  "order": 5,
  "groupName": "grupos_componentes"
}
```

REMOVE_GROUP

Eliminar un grupo de componentes dentro del componente *REPEATABLEGROUP* en la propiedad `blocks`

Propiedades

Propiedad	Descripción	Ejemplo
`groupName`	Nombre del *REPEATABLEGROUP*	`"usuarios"`
`groupIndex`	Indice del grupo a borrar	`2`

Ejemplo:

```
{
  "type": "REMOVE_GROUP",
  "order": 5,
  "groupName": "grupos_componentes",
  "groupIndex": {
    "key": "index",
    "type": "GROUP_INDEX"
  }
}
```

TAKE_PHOTO

Capturar una foto del componente *VIDEO_CAMERA*

Propiedades

Propiedad	Descripción	Ejemplo
`fieldName`	Nombre del campo	`"camara.{{index}}"`
`fieldNameContexts`	Arreglo de contextos del nombre del campo	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`
`width`	Ancho de la imagen en pixeles	`720`
`height`	Altua de la imagen en pixeles	`720`

Ejemplo:

```
{
  "order": 5,
  "type": "TAKE_PHOTO",
  "fieldName": "video",
```

```
    "width": 720
  }
```

POST_MESSAGE_WEB_VIEW

Emitir un mensaje al webview (ideal para notificar cosas en aplicaciones móviles)

Propiedades

Propiedad	Descripción	Ejemplo
`parameter`	Contexto	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`

Ejemplo:

```
{
  "type": "POST_MESSAGE_WEB_VIEW",
  "order": 5,
  "parameter": {
    "key": "parameter",
    "type": "TEMPLATE",
    "template": "{'channel': 'REQUEST_CAMERA_PERMISSION'}"
  }
}
```

> En este ejemplo, se notifica al WebView para que gestione la solicitud de permisos de cámara ante el sistema operativo.

SET_SEARCH_PARAMS

Setear parametros a la URL (debe ser un seteo del objeto completo porque sobrescribe todos)

Propiedades

Propiedad	Descripción	Ejemplo
`contexts`	Arreglo de contextos	`[{ "key": "index", "type": "TEMPLATE", "template": "1"}]`

Ejemplo:

```
{
  "type": "SET_SEARCH_PARAMS",
  "order": 15,
  "contexts": [
    {
      "key": "id",
      "type": "LOCAL",
      "contextName": "sas_id"
    }
  ]
}
```

Bloques

Los bloques son los disparados cuando se ejecuta un evento.

Tipos de bloques

CONDITIONAL

Permite evaluar si se cumple un condición, en caso de hacerlo se ejecutan las *actions* y el *then* que es un arreglo de bloques ordenados

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`orden`		Número de orden del bloque	`1`
`then`		Bloques que se ejecutan por condición **verdadera**	`[...<aquí van bloques>...]`
`else`	X	Bloques que se ejecutan por condición **falsa**	`[...<aquí van bloques>...]`
`actions`	X	Acciones que se ejecutan por condición **verdadera** (**DEPRECADO**, usar bloque *ACTION*)	`[...<aquí van acciones>...]`

Ejemplo:

```
{
  "order": 10,
  "type": "CONDITIONAL",
  "conditions": {
    "context": // aquí un contexto que devuelva true o false
  },
  "actions": [
    // aquí acciones
  ],
  "then": [
    // aquí van bloques
  ],
  "else": [
    // aquí van bloques
  ]
}
```

FOREACH

Propiedades

Propiedad	Opcional	Descripción	Ejemplo
`orden`		Número de orden del bloque	`1`
`context`		Contexto que devuelve un arreglo o número	`{ "key": "index", "type": "TEMPLATE", "template": "1"}`
`blocks`		Bloques que se ejecutan	`[...<aquí van bloques>...]`
`actions`	X	Acciones que se ejecutan (**DEPRECADO**, usar bloque *ACTION*)	`[...<aquí van acciones>...]`

Ejemplo:

```
{
  "order": 15,
  "type": "FOREACH",
  "name": "loop1",
  "context": // aquí contexto,
  "blocks": [
    // aquí van bloques
  ]
}
```

> El contexto del iterador debe ser un arreglo u número de veces que se quiere iterar.

ACTION

Propiedad	Opcional	Descripción	Ejemplo
`orden`		Número de orden del bloque	`1`
`actions`		Acciones que se ejecutan	`[...<aquí van acciones>...]`

```
{
  "order": 10,
  "type": "ACTION",
  "actions": [
    {
      "type": "CLEAR_ERRORS",
      "order": 5,
      "fieldName": "fecha_transferencia"
    }
  ]
}
```

Certificados

Los certificados se utilizan para autenticación en acciones remotas que requieren SSL/TLS.

Tipos de Certificados

Tipo	Descripción	Propiedades
PEM	Certificado en formato PEM	`certificate`, `key`, `headers`
PFX	Certificado en formato PFX	`pfx`, `passphrase`, `headers`

Ejemplo de Certificado PEM

```
{
  "name": "my-certificate",
  "type": "PEM",
  "certificate": "-----BEGIN CERTIFICATE-----...",
  "key": "-----BEGIN PRIVATE KEY------...",
  "headers": "{\"X-Custom-Header\": \"value\"}"
}
```

> El campo headers se procesa como un template y hereda el contexto de ejecución de la acción remota, permitiendo el reemplazo automático de variables dinámicas.

Variantes

Las variantes permiten reutilizar estilos de Tailwind CSS en diferentes componentes.

Propiedades de Variantes

Propiedad	Tipo	Descripción	Ejemplo
`name`	string	Nombre único de la variante	`"primary-button"`
`className`	string	Clases de Tailwind CSS	`"bg-blue-500 text-white hover:bg-blue-600"`
`componentType`	COMPONENT_TYPE	Tipo de componente	`"BUTTON"`

Ejemplo de Variante

```
{
  "name": "primary-button",
  "className": "bg-blue-500 text-white px-4 py-2 rounded hover:bg-blue-600 focus:outline-none focus:ring-2 focus:ring-
blue-500",
  "componentType": "BUTTON"
}
```

Uso de Variantes

```
{
  "name": "submitButton",
  "order": 1,
  "componentType": "BUTTON",
  "component": {
    "label": "Enviar",
    "variantName": "primary-button",
    "className": "w-full" // Se combina con la variante
  }
}
```

> Las variantes están pensadas principalmente para componente *TEXT*, *LABEL* , *BUTTON* y *ALERT* aunque está previsto que se puedan aplicar a componentes de campos.

Estructura Completa de un Wizard

Puede consultar la carpeta *postman-collections* para obtener ejemplos en postman

Consideraciones Técnicas

Liquidjs

- Al crear templates que inyectan datos dinámicos en un contexto, es fundamental conocer la naturaleza y el origen de esos datos, ya que una inserción incorrecta puede provocar problemas de escape de caracteres. Para evitarlo, se debe aplicar el filtro de escape correspondiente, por ejemplo: `{{ value | escape }}`, asegurando que el contenido se renderice de forma segura.

Tailwind CSS

- Para estilizar cualquier `className` se debe usar Tailwind CSS.

Almacenamiento

- Los archivos que se cargan durante un wizard se guardan en un directorio temporal del servidor de Minio por lo que será borrado finalizado el tiempo máximo de duración de la sesión que se encuentra en las variables de entorno.

Recursos Adicionales

- Documentación JSONata
- Documentación LiquidJS
- Tailwind CSS
- Lucide Icons

Esta documentación cubre todos los aspectos necesarios para crear wizards complejos y funcionales en la plataforma. Para casos específicos o dudas adicionales, consultar con el equipo de desarrollo.