Nexus — Elemento: Data Container

¿Qué es?

Data Container es un componente de tipo Data pensado para optimizar el rendimiento de un tablero. Su función es realizar una única llamada a una base de datos o API, para que otros componentes (por ejemplo, tablas) tomen la información desde ahí en lugar de que cada componente haga su propia llamada por separado. Esto evita llamadas repetidas e innecesarias a la misma fuente de datos.

Si la información es dinámica, se puede programar un botón de “refresh” (mediante Events & Actions) para que Data Container vuelva a consultar la base de datos y actualice la información que consumen los demás componentes.

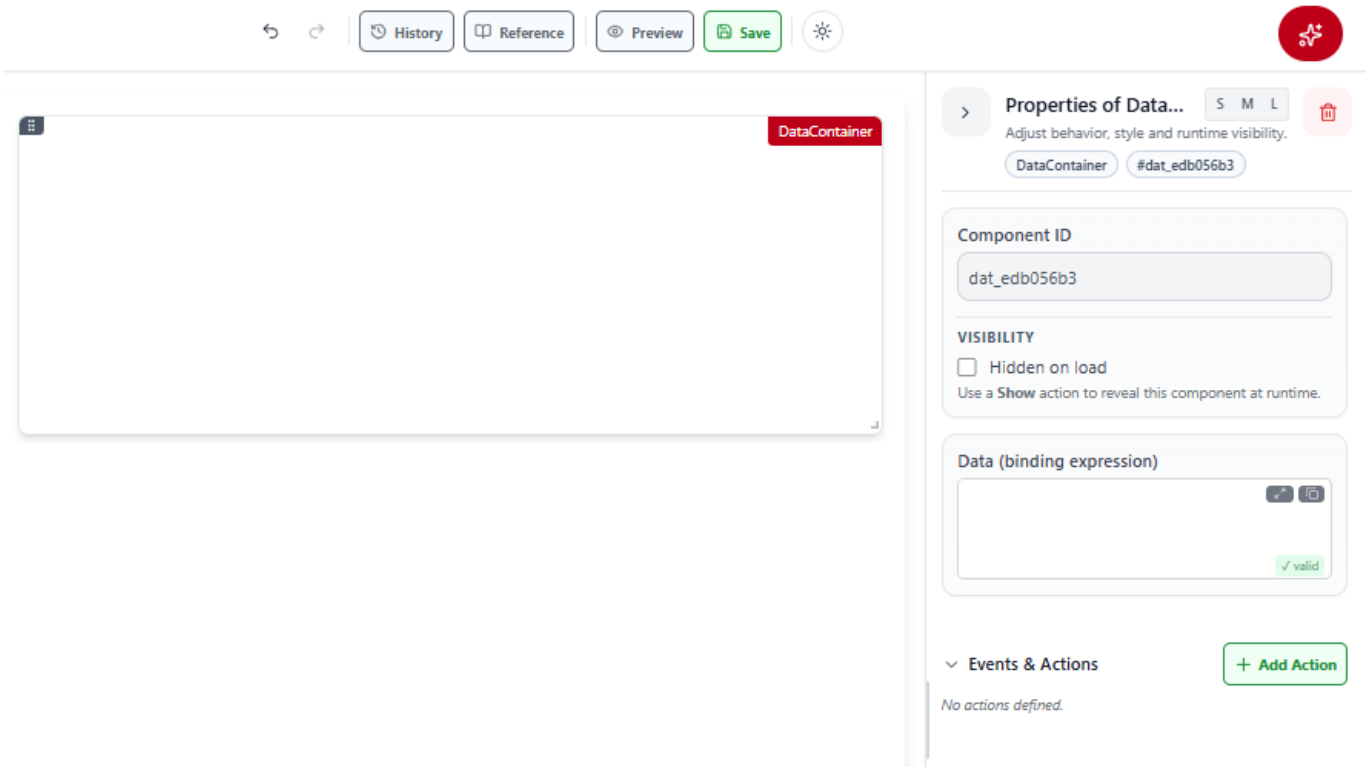
El propio Data Container no se muestra en el Preview, ya que es un contenedor sin representación visual. Esto no significa que el botón de refresh tampoco se vea: ese botón es un componente aparte, así que se muestra con normalidad si el usuario lo arma en el tablero; al presionarlo, la información se actualiza correctamente.

Propiedades

Al seleccionar Data Container en el lienzo, el panel derecho muestra “Properties of DataContainer”.

- **Component ID:** identificador único del componente, autogenerado (ej. dat_edb056b3).
- **Visibility → Hidden on load:** casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo Show.
- **Selector S / M / L:** ajusta el tamaño de visualización del panel de propiedades.
- **Data (binding expression):** campo de código donde se define la expresión de binding con los datos que va a almacenar/exponer el contenedor (por ejemplo, una llamada a una API o base de datos). Cuenta con el editor de código expandido para más comodidad y valida el contenido en tiempo real (marca “✓ valid”).
- **Events & Actions:** sistema para definir comportamientos interactivos, como la acción de refresh. Ver documentación aparte: Nexus — Events & Actions .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del

lienzo.



Ejemplo de uso

Un caso típico es un tablero con varias tablas y gráficos que necesitan los mismos datos de ventas. En vez de que cada componente llame a la base de datos por su cuenta, se agrega un Data Container que hace esa llamada una sola vez, y las tablas y gráficos toman la información desde ahí.

Si esos datos cambian con frecuencia, se puede agregar un botón en el tablero y configurarle, desde Events & Actions, una acción que refresque el Data Container. Así, cada vez que el usuario presiona ese botón, se vuelve a consultar la base de datos y todos los componentes conectados se actualizan automáticamente, sin necesidad de recargar toda la pantalla.

Nexus — Elemento: Calendar

¿Qué es?

Calendar es un componente de tipo Data. Muestra un calendario interactivo con distintas vistas (Mes, Semana, Día) donde se pueden visualizar eventos con título, rango de fecha/hora y color propio. Al arrastrarlo al lienzo viene con un ejemplo precargado (mes de septiembre de 2026, con 3 eventos: “Sprint planning”, “Design review” y “Demo day”).

Propiedades

Al seleccionar Calendar en el lienzo, el panel derecho muestra “Properties of Calendar”, organizado en tres pestañas: Content, Style y Behavior.

- **Component ID:** identificador único del componente, autogenerado (ej. cal_732f41bc).
- **Visibility → Hidden on load:** casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo Show.
- **Selector S / M / L:** ajusta el tamaño de visualización del panel de propiedades.

Pestaña Content:

- **Data (array with start/end/title):** campo de código donde se definen los eventos en formato JSON. Cada evento se compone de:
  - title: título del evento (ej. “Sprint planning”).start: fecha y hora de inicio, en formato AAAA-MM-DDTHH:mm:ss (ej. “2026-03-25T10:00:00”). También admite solo fecha sin hora (ej. “2026-03-28”) para eventos de todo el día.end: fecha y hora de fin, mismo formato que start.color: color del evento en el calendario, en formato hexadecimal (ej. “#2563eb“).

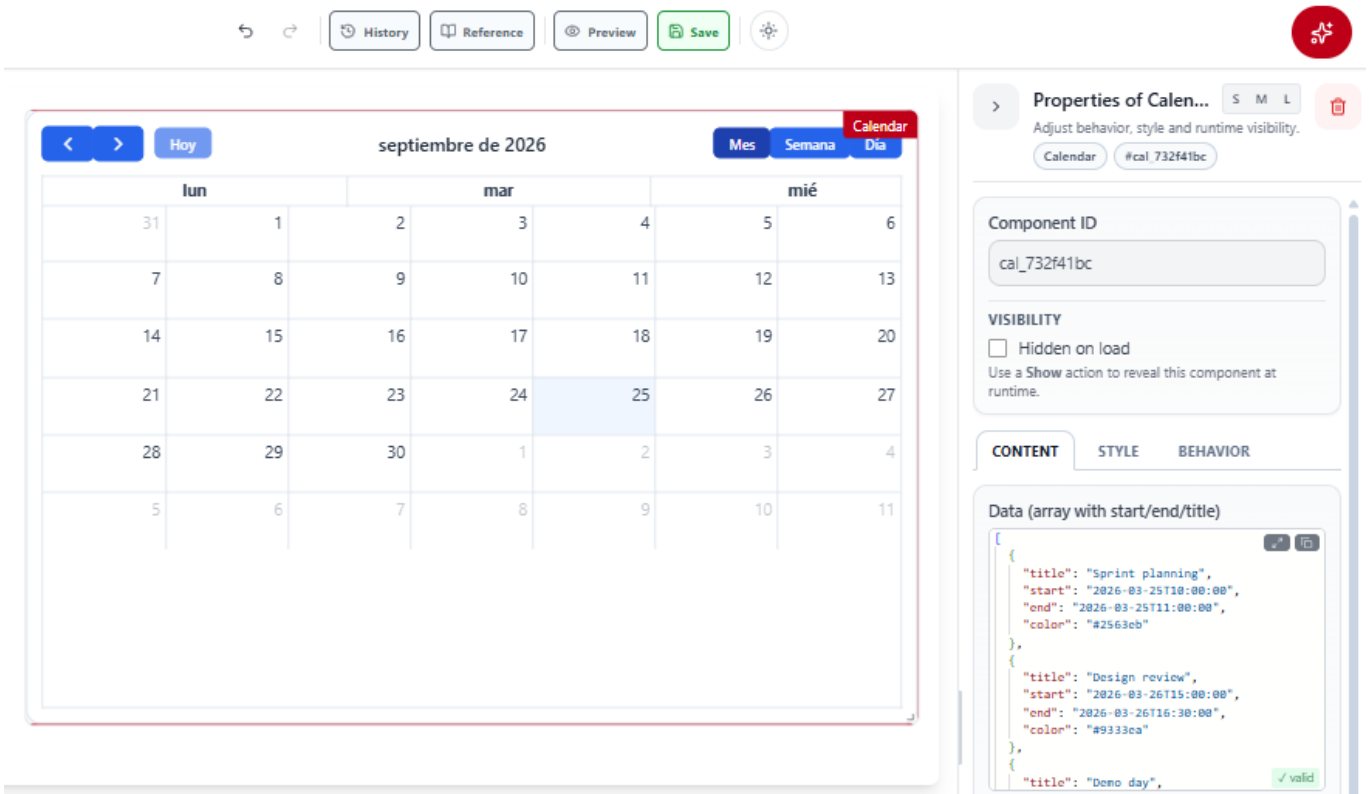
  Ejemplo de estructura:

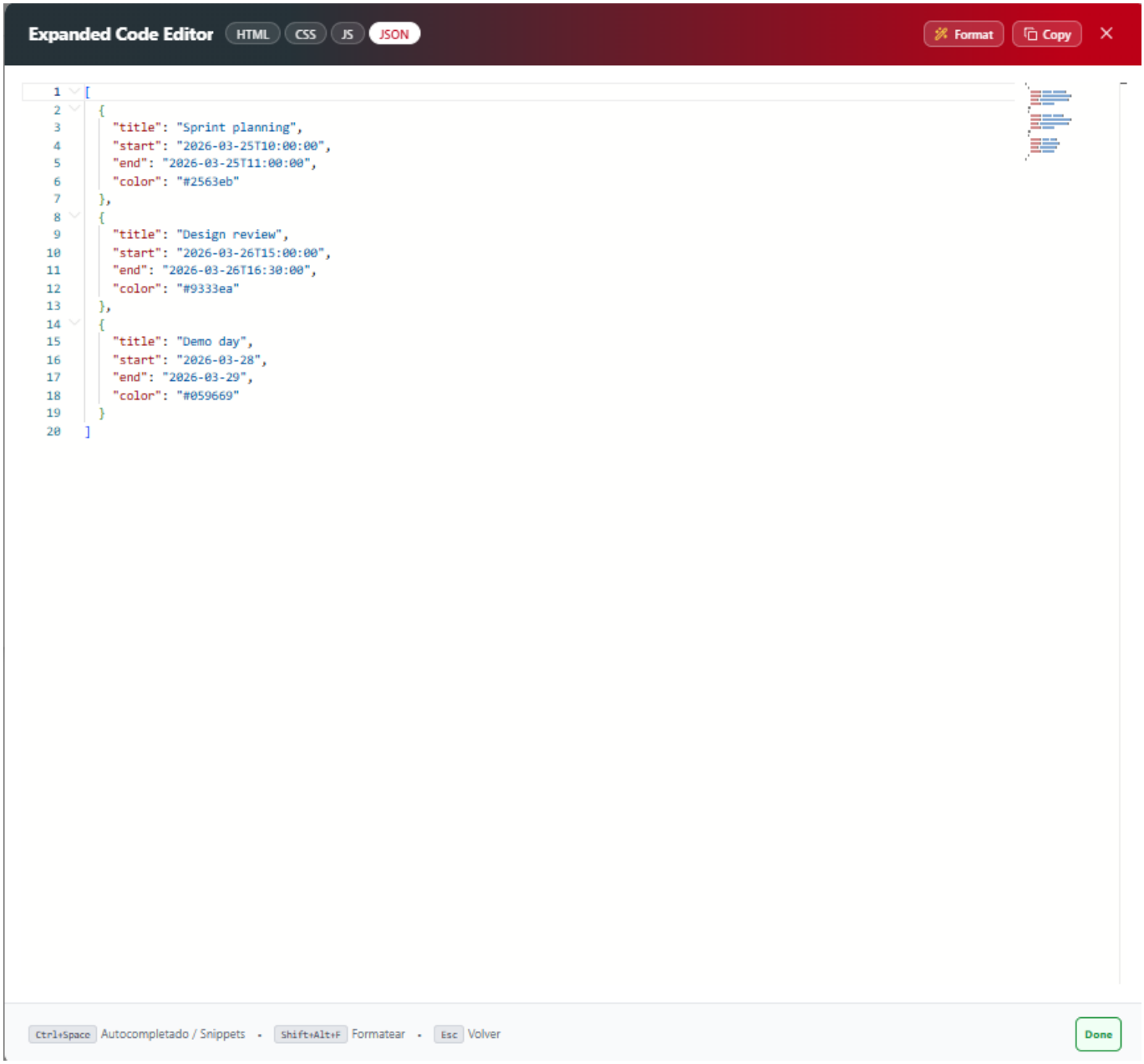
```
[
  {
```

```
    "title": "Sprint planning",
    "start": "2026-03-25T10:00:00",
    "end": "2026-03-25T11:00:00",
    "color": "#2563eb"
  },
  {
    "title": "Design review",
    "start": "2026-03-26T15:00:00",
    "end": "2026-03-26T16:30:00",
    "color": "#9333ea"
  },
  {
    "title": "Demo day",
    "start": "2026-03-28",
    "end": "2026-03-29",
    "color": "#059669"
  }
]
```

Cuenta con el editor de código expandido para más comodidad, con pestañas HTML/CSS/JS/JSON y botones Format y Copy. El editor valida el JSON en tiempo real (marca “✓ valid”).

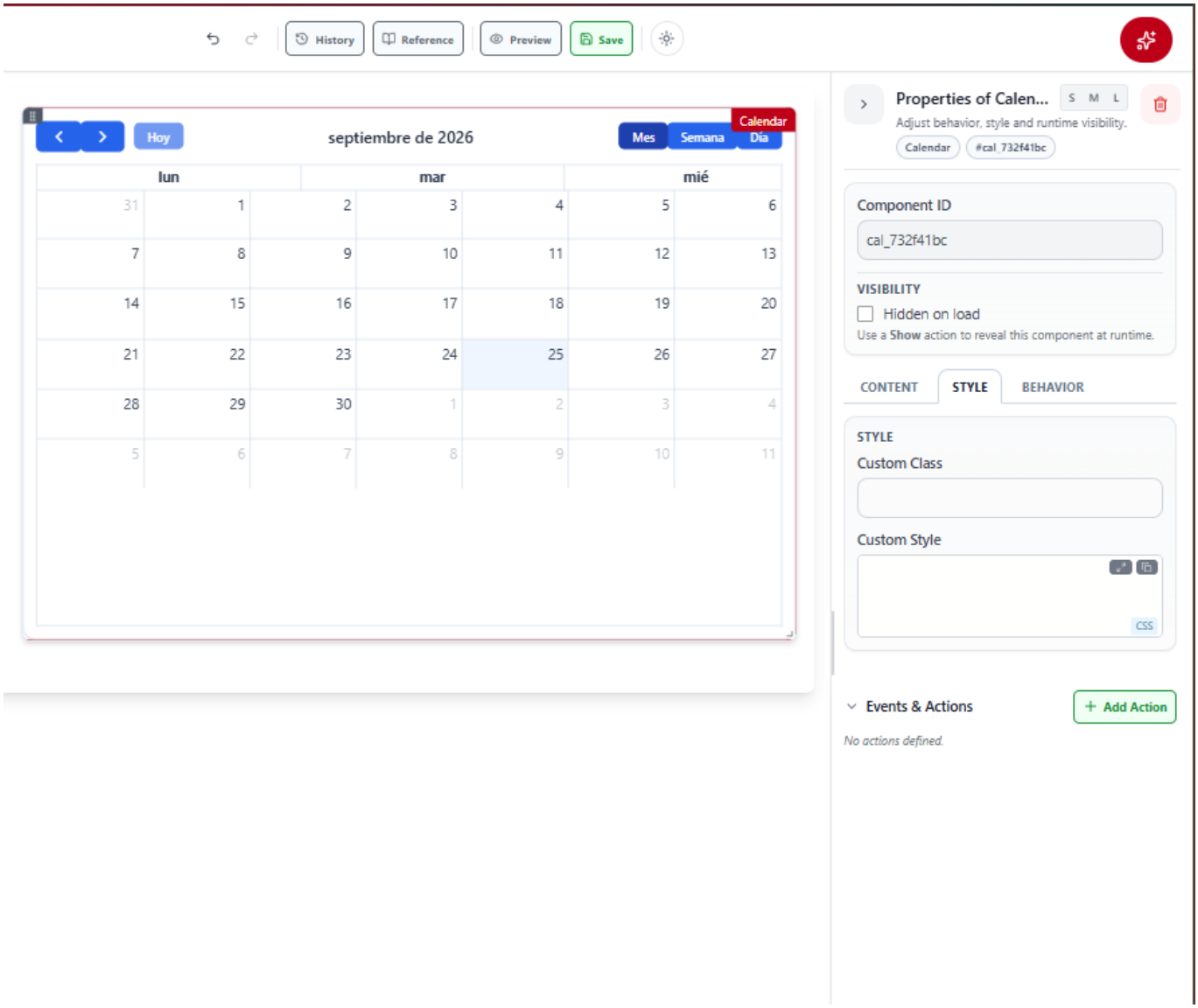
- **Event Start Field:** nombre de la propiedad del JSON que se usa como fecha/hora de inicio (por defecto start). Permite adaptar el componente a datos con otro nombre de campo.
- **Event End Field:** nombre de la propiedad del JSON que se usa como fecha/hora de fin (por defecto end).
- **Title Field:** nombre de la propiedad del JSON que se usa como título del evento (por defecto title).
- **Color Field (optional):** nombre de la propiedad del JSON que se usa como color del evento (por defecto color). Es opcional; si no se define, el componente asigna un color por defecto a los eventos.


History
Reference
Preview
Save
Calendar
Hoy
septiembre de 2026
Mes
Semana
Día
lun
mar
mié
31 1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 1 2 3 4
5 6 7 8 9 10 11
Properties of Calen...
S M L
Adjust behavior, style and runtime visibility.
Calendar
#cal_732f41bc
Component ID
cal_732f41bc
VISIBILITY
Hidden on load
Use a Show action to reveal this component at runtime.
CONTENT
STYLE
BEHAVIOR
Data (array with start/end/title)
[
{
"title": "Sprint planning",
"start": "2026-03-25T10:00:00",
"end": "2026-03-25T11:00:00",
"color": "#2563eb"
},
{
"title": "Design review",
"start": "2026-03-26T15:00:00",
"end": "2026-03-26T16:30:00",
"color": "#9333ea"
},
{
"title": "Demo day",
✓ valid
Event Start Field
start
Event End Field
end
Title Field
title
Color Field (optional)
color
Events & Actions
+ Add Action
No actions defined.



Pestaña Style:

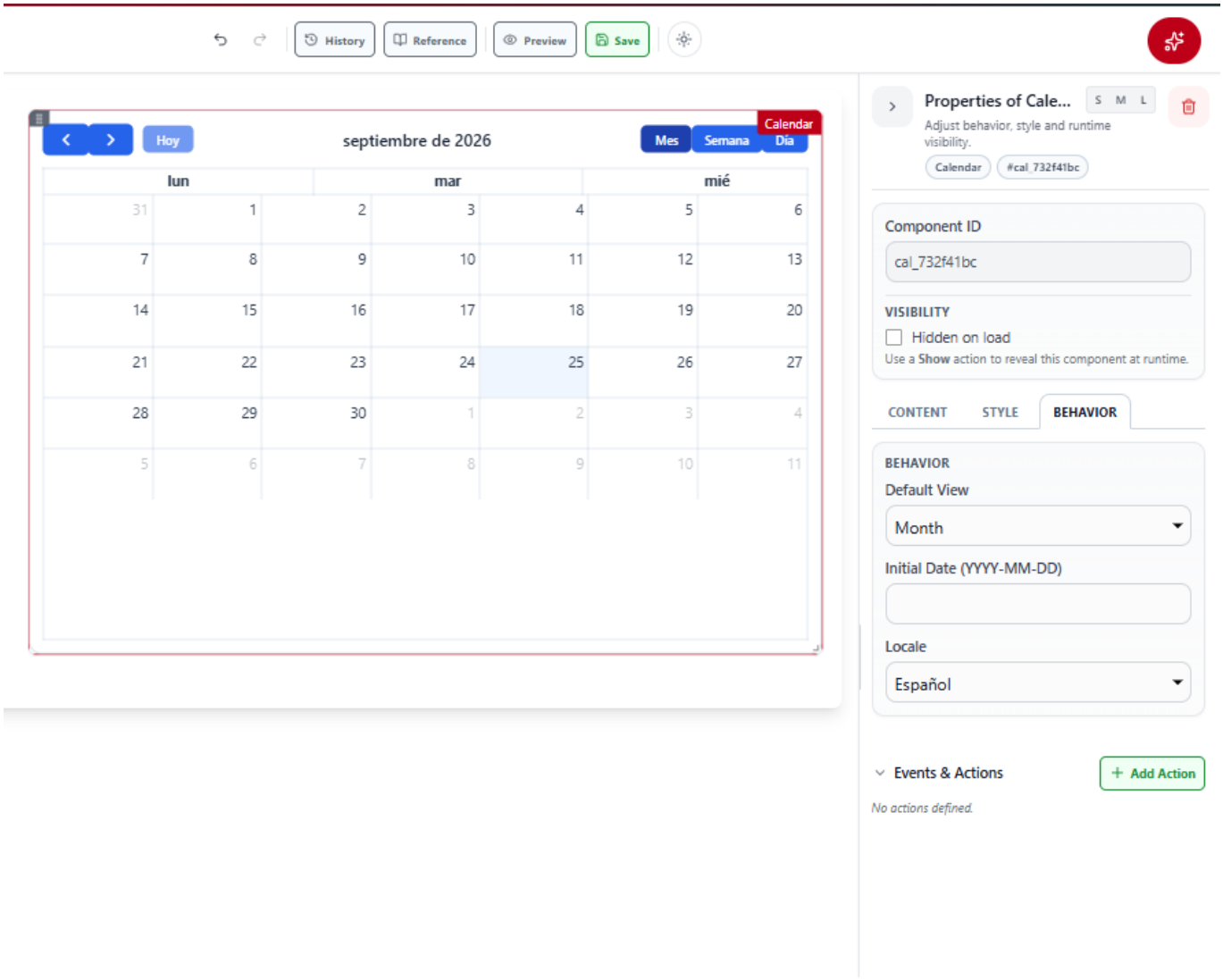
- **Custom Class:** clase CSS personalizada.
- **Custom Style:** CSS personalizado aplicado directamente a este componente.



Pestaña Behavior:

- **Default View:** menú desplegable con la vista inicial del calendario al cargar. Opciones disponibles:
  - Month
  - (a confirmar el resto de opciones – ver duda abajo)

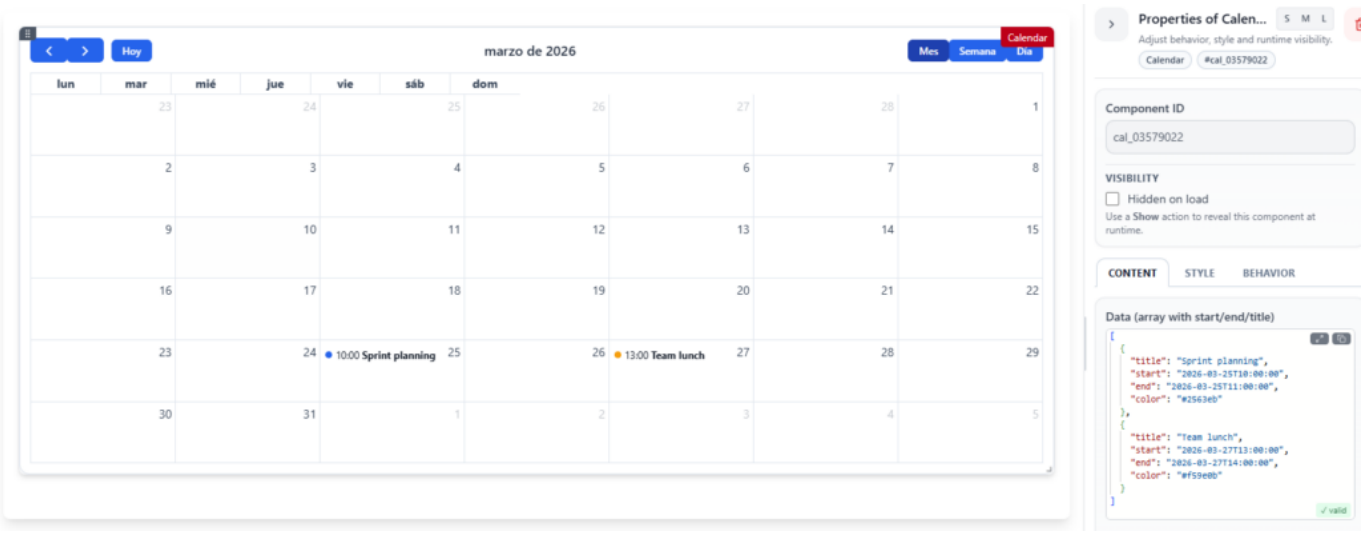
- **Initial Date (YYYY-MM-DD):** fecha con la que se abre el calendario por defecto. Si se deja vacío, se abre en la fecha actual.
- **Locale:** menú desplegable con el idioma/localización del calendario (formato de nombres de mes, días de la semana, etc.). Valor por defecto visto: “Español”.
- **Events & Actions:** sistema para definir comportamientos interactivos. Ver documentación aparte: Nexus – Events & Actions .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del lienzo.



Ejemplo de uso

Para agregar un nuevo evento al calendario, se suma un nuevo objeto dentro del array, separado del anterior por una coma. El array completo, con sus corchetes de apertura y cierre, quedaría así:

```
[
  {
    "title": "Sprint planning",
    "start": "2026-03-25T10:00:00",
    "end": "2026-03-25T11:00:00",
    "color": "#2563eb"
  },
  {
    "title": "Team lunch",
    "start": "2026-03-27T13:00:00",
    "end": "2026-03-27T14:00:00",
    "color": "#f59e0b"
  }
]
```

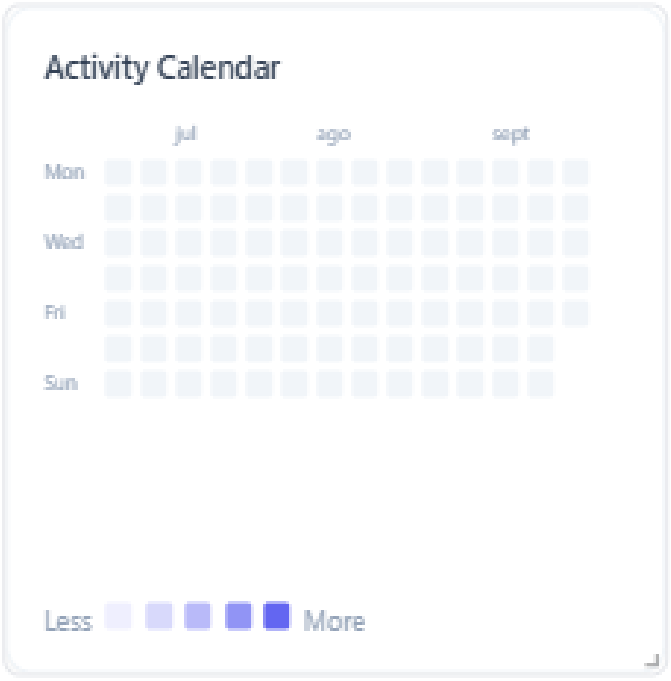


Con Event Start Field, Event End Field, Title Field y Color Field se puede conectar el calendario a datos que ya vienen de otra fuente (por ejemplo una API) sin necesidad de renombrar sus propiedades. Con Default View e Initial Date se puede definir cómo se ve el calendario apenas carga la pantalla, por ejemplo abriéndolo directamente en la vista semanal de una fecha puntual.

Nexus – Elemento: Heatmap Calendar

¿Qué es?

Heatmap Calendar es un componente de tipo Data. Muestra un calendario tipo mapa de calor (similar al de contribuciones de GitHub), donde cada día se representa como una celda cuya intensidad de color varía según un valor asociado. Al arrastrarlo al lienzo viene con un ejemplo precargado (“Activity Calendar”, con datos de actividad diaria distribuidos entre diciembre y marzo).



Propiedades

Al seleccionar Heatmap Calendar en el lienzo, el panel derecho muestra “Properties of HeatmapCalendar”, organizado en tres pestañas: Content, Style y Behavior.

- **Component ID:** identificador único del componente, autogenerado (ej. hmp_4d2d519a).
- **Visibility → Hidden on load:** casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo Show.
- **Selector S / M / L:** ajusta el tamaño de visualización del panel de propiedades.

Pestaña Content:

- **Title:** título del calendario (ej. “Activity Calendar”).
- **Data (array {date, value}):** campo de código donde se definen los registros en formato JSON. Cada registro se compone de:
  - date: fecha del registro, en formato AAAA-MM-DD (ej. “2025-12-18”).value: valor numérico asociado a esa fecha, que determina la intensidad del color en la celda (ej. 8).

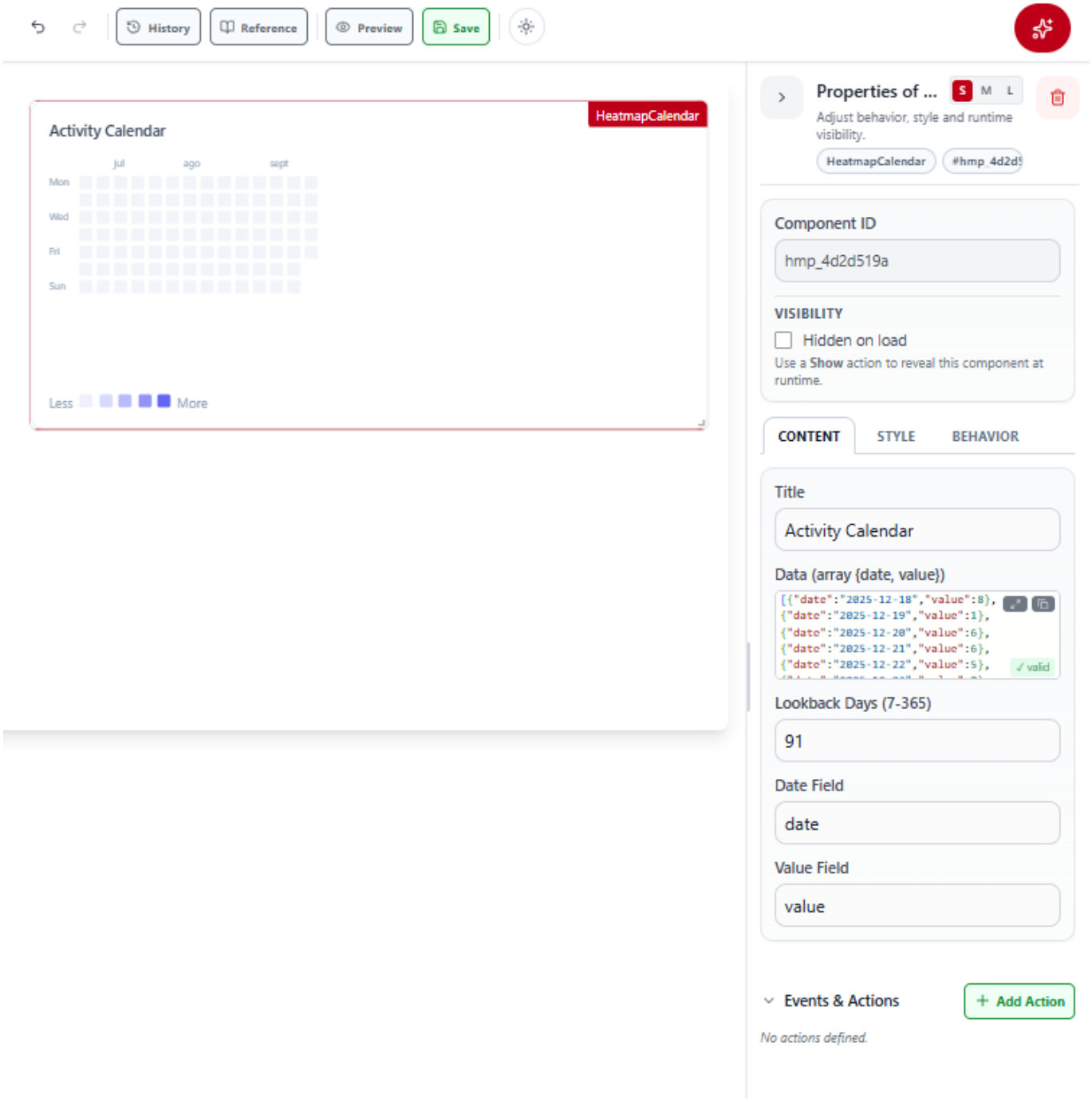
  Ejemplo de estructura:

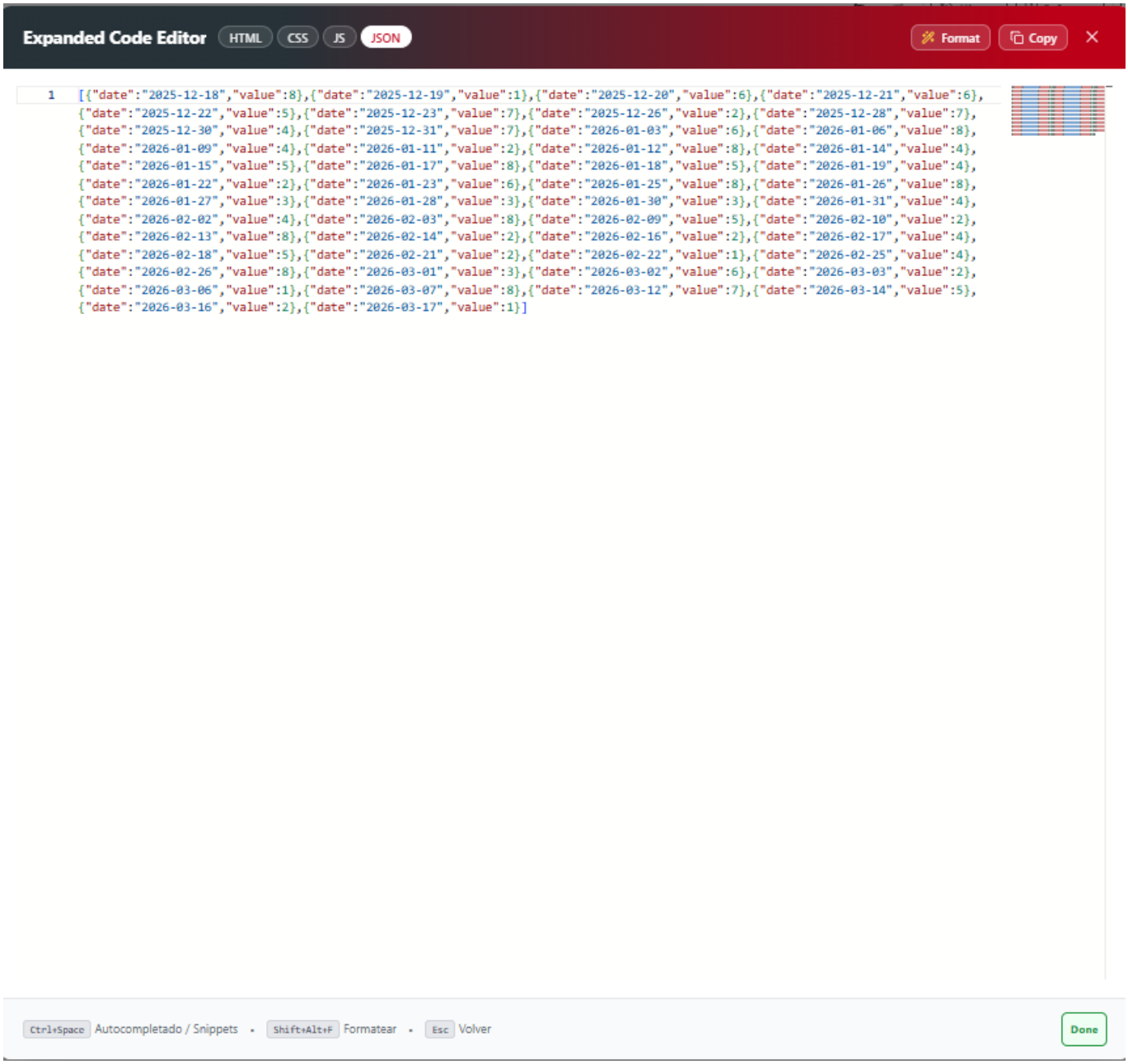
```
[
```

```
    {
      "date": "2025-12-18",
      "value": 8
    },
    {
      "date": "2025-12-19",
      "value": 1
    },
    {
      "date": "2025-12-20",
      "value": 6
    }
  ]
```

Cuenta con el editor de código expandido para más comodidad, con pestañas HTML/CSS/JS/JSON y botones Format y Copy. El editor valida el JSON en tiempo real (marca “✓ valid”).

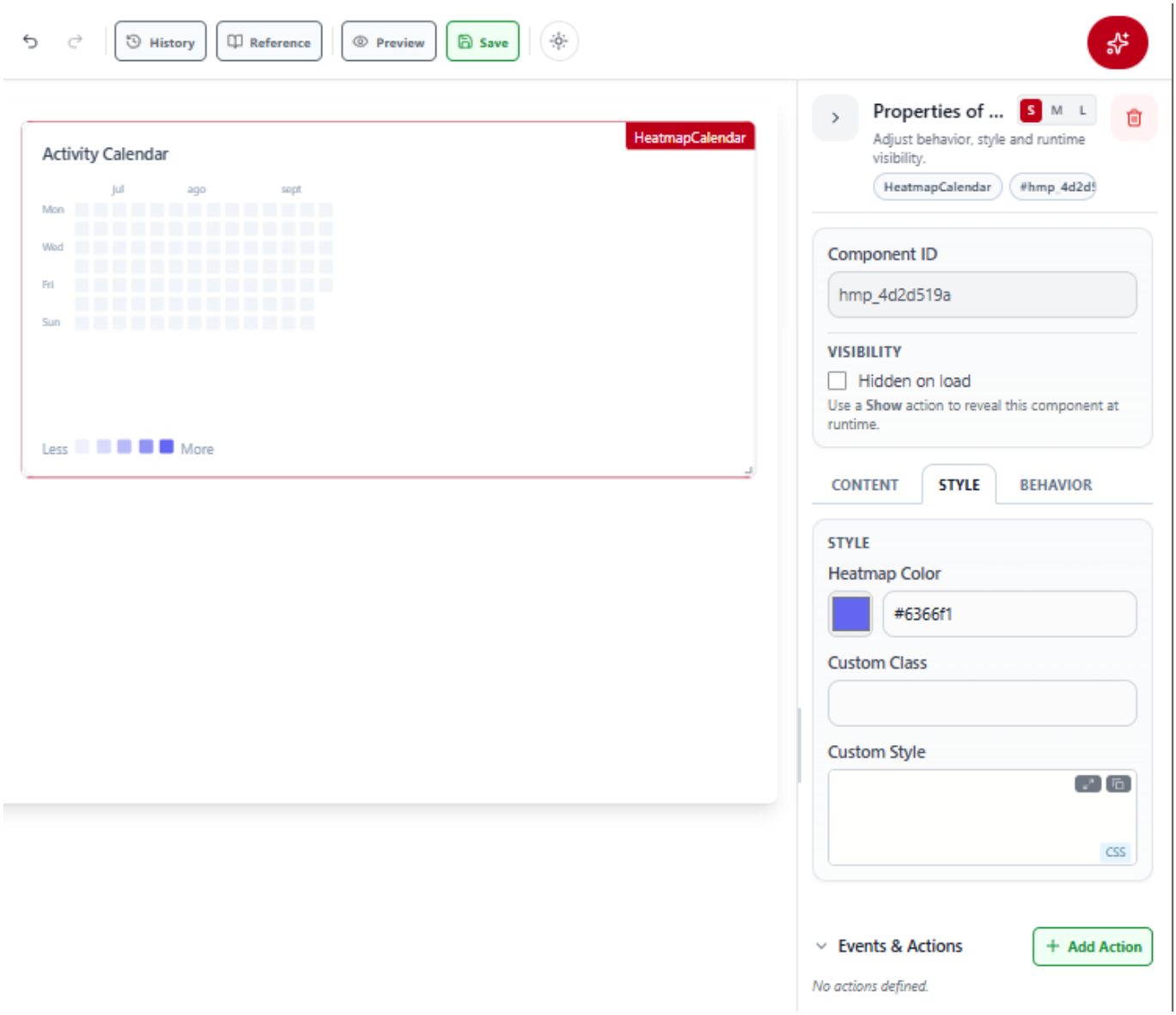
- **Lookback Days (7-365):** cantidad de días hacia atrás que se muestran en el calendario, contados desde la fecha más reciente de los datos (ej. 91).
- **Date Field:** nombre de la propiedad del JSON que se usa como fecha (por defecto date). Permite adaptar el componente a datos con otro nombre de campo.
- **Value Field:** nombre de la propiedad del JSON que se usa como valor (por defecto value). Permite adaptar el componente a datos con otro nombre de campo.


History
Reference
Preview
Save
HeatmapCalendar
Activity Calendar
jul
ago
sept
Mon
Wed
Fri
Sun
Less
More
Properties of ...
S M L
Adjust behavior, style and runtime visibility.
HeatmapCalendar
#hmp_4d2d5
Component ID
hmp_4d2d519a
VISIBILITY
Hidden on load
Use a Show action to reveal this component at runtime.
CONTENT
STYLE
BEHAVIOR
Title
Activity Calendar
Data (array {date, value})
[{"date":"2025-12-18","value":8},
{"date":"2025-12-19","value":1},
{"date":"2025-12-20","value":6},
{"date":"2025-12-21","value":6},
{"date":"2025-12-22","value":5},
✓ valid
Lookback Days (7-365)
91
Date Field
date
Value Field
value
Events & Actions
+ Add Action
No actions defined.



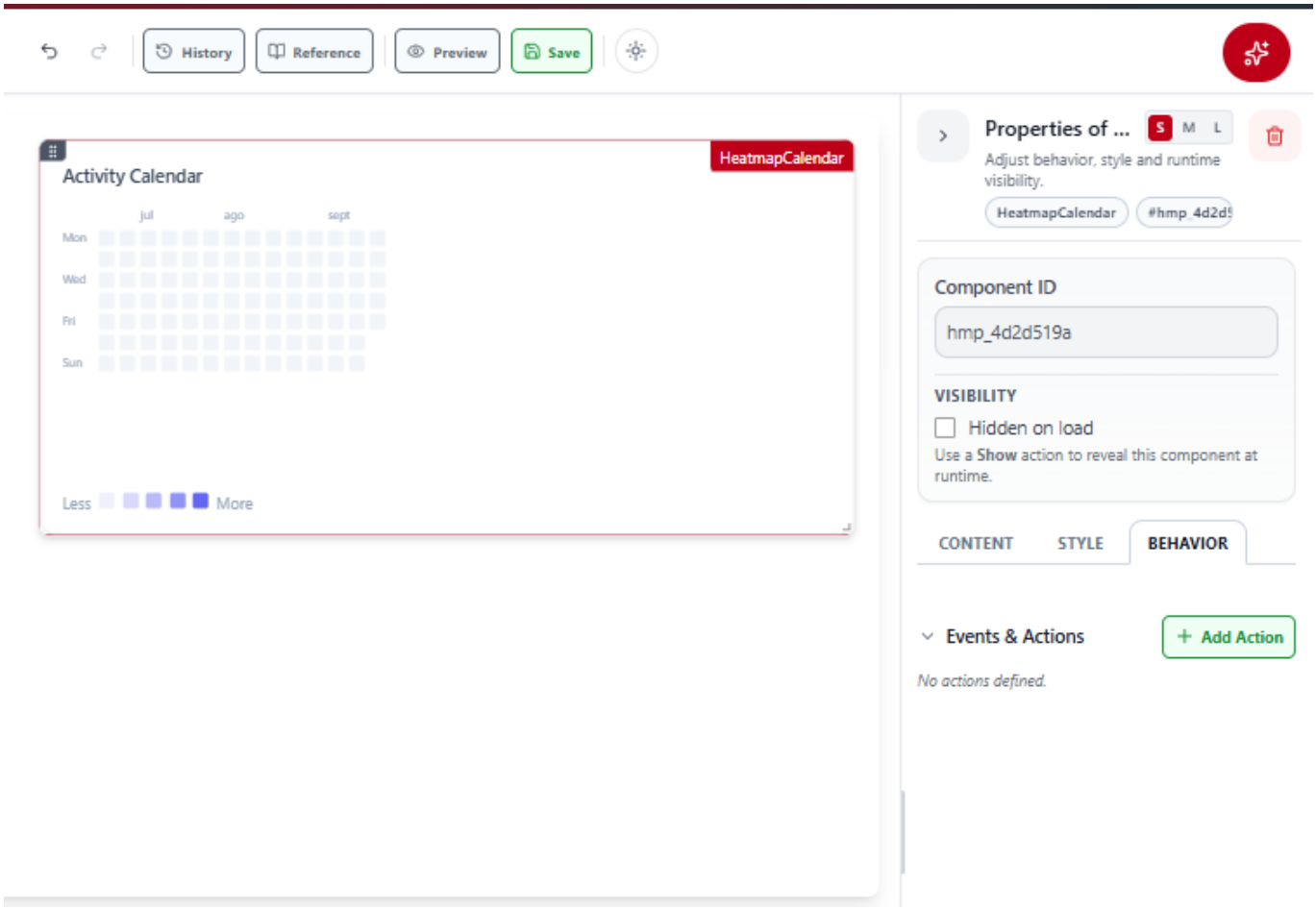
Pestaña Style:

- **Heatmap Color:** selector de color base para las celdas con mayor intensidad (color picker + campo de texto con código hexadecimal, ej. #6366f1).
- **Custom Class:** clase CSS personalizada.
- **Custom Style:** CSS personalizado aplicado directamente a este componente.



Pestaña Behavior:

- **Events & Actions:** sistema para definir comportamientos interactivos. Ver documentación aparte: [Nexus — Events & Actions](#) .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del lienzo.



Ejemplo de uso

Para agregar un nuevo registro al calendario, se suma un nuevo objeto dentro del array, separado del anterior por una coma. El array completo, con sus corchetes de apertura y cierre, quedaría así:

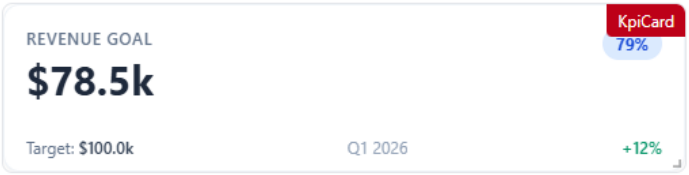
```
[
  {
    "date": "2025-12-18",
    "value": 8
  },
  {
    "date": "2026-03-18",
    "value": 5
  }
]
```

La escala de color (referencia “Less” a “More” que se ve debajo del calendario) se calcula automáticamente en función del rango de valores presentes en los datos, y se puede personalizar el tono base desde Heatmap Color. Date Field y Value Field son útiles cuando el JSON de origen proviene de otra fuente (por ejemplo una API externa) que use nombres de propiedad distintos a date y value, evitando así tener que renombrar los campos manualmente.

Nexus — Elemento: KPI Card

¿Qué es?

KPI Card es un componente de tipo Data. Muestra un indicador clave de rendimiento con su valor actual, una barra de progreso hacia un objetivo, y datos complementarios como el porcentaje de avance, el período correspondiente y una tendencia. Al arrastrarlo al lienzo viene con un ejemplo precargado (“Revenue Goal”, valor actual $78.5k sobre un objetivo de $100.0k, período Q1 2026, con tendencia +12%).



Propiedades

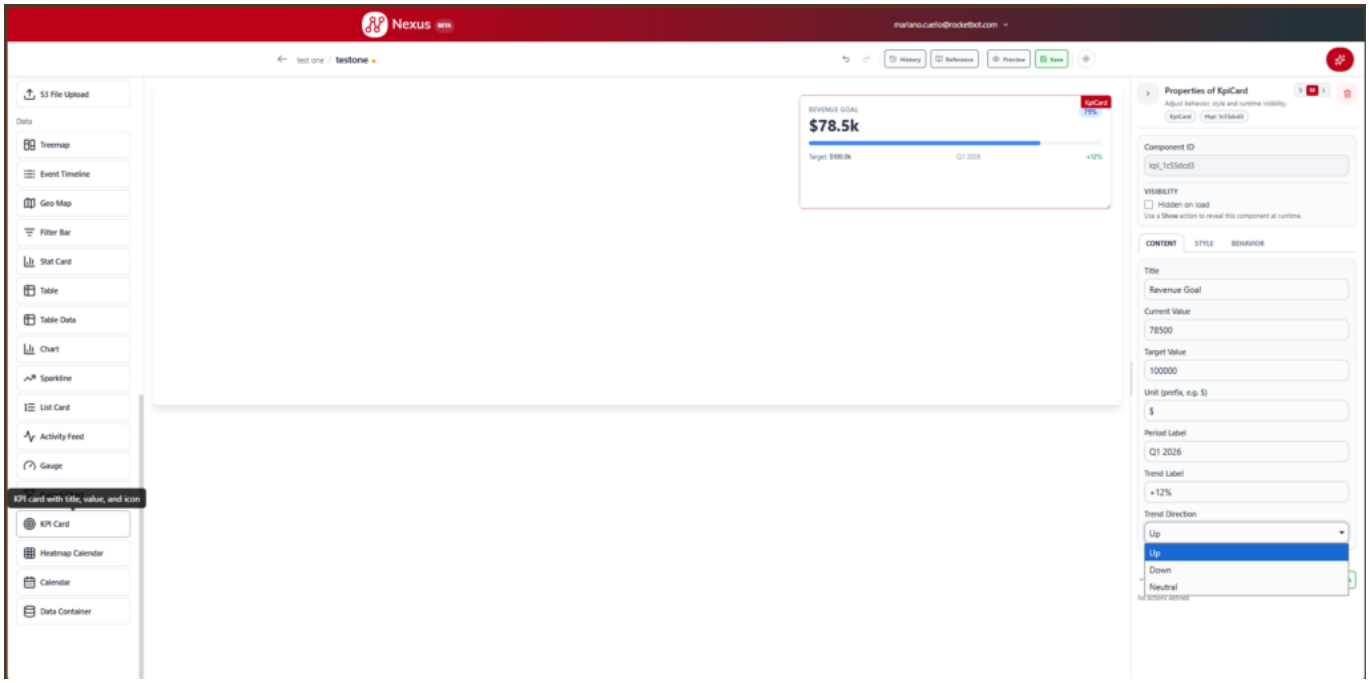
Al seleccionar KPI Card en el lienzo, el panel derecho muestra “Properties of KpiCard”, organizado en tres pestañas: Content, Style y Behavior.

- **Component ID:** identificador único del componente, autogenerado (ej. kpi_1c55dcd3).
- **Visibility → Hidden on load:** casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo Show.
- **Selector S / M / L:** ajusta el tamaño de visualización del panel de propiedades.

Pestaña Content:

- **Title:** título de la tarjeta (ej. “Revenue Goal”).
- **Current Value:** valor actual del indicador, en formato numérico simple (ej. 78500; se muestra abreviado en el componente como “$78.5k”).
- **Target Value:** valor objetivo a alcanzar (ej. 100000).
- **Unit (prefix, e.g. $):** símbolo o texto que se antepone a los valores mostrados (ej. “$”).

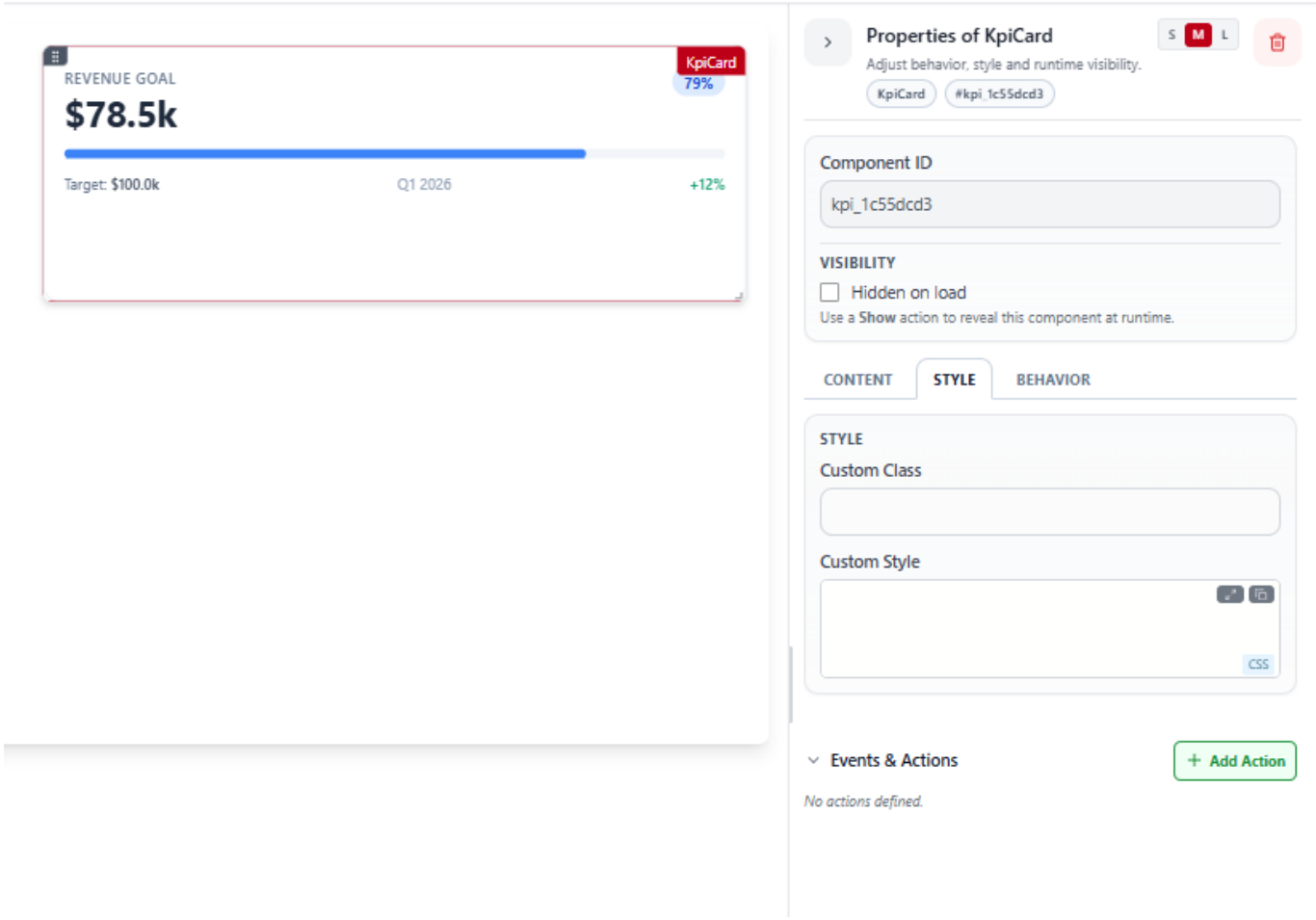
- **Period Label:** etiqueta del período que representa el KPI (ej. “Q1 2026”).
- **Trend Label:** texto que indica la variación respecto a un período anterior (ej. “+12%”).
- **Trend Direction:** menú desplegable que determina el color con que se muestra el Trend Label (verde para positivo, rojo para negativo, gris/neutro para sin cambios). Opciones disponibles:
    - Up
    - Down
    - Neutral



Además, en la esquina superior derecha del componente se muestra automáticamente el porcentaje de avance (Current Value / Target Value), calculado sin necesidad de configuración adicional (ej. “79%”).

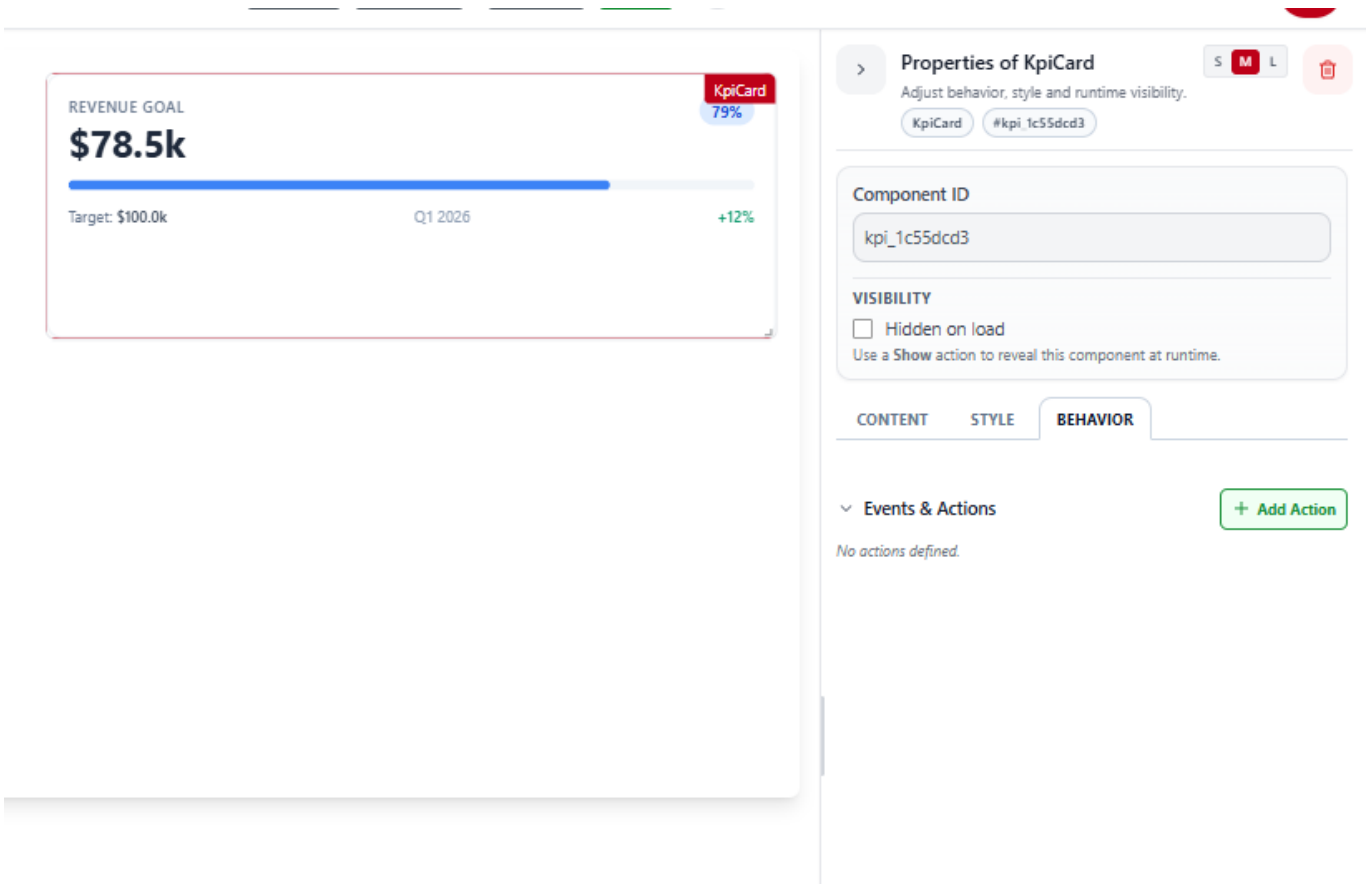
Pestaña Style:

- **Custom Class:** clase CSS personalizada.
- **Custom Style:** CSS personalizado aplicado directamente a este componente.



Pestaña Behavior:

- **Events & Actions:** sistema para definir comportamientos interactivos. Ver documentación aparte: [Nexus — Events & Actions]() .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del lienzo.



Ejemplo de uso

Para adaptar KPI Card a otra métrica, por ejemplo un objetivo de tickets resueltos en soporte:

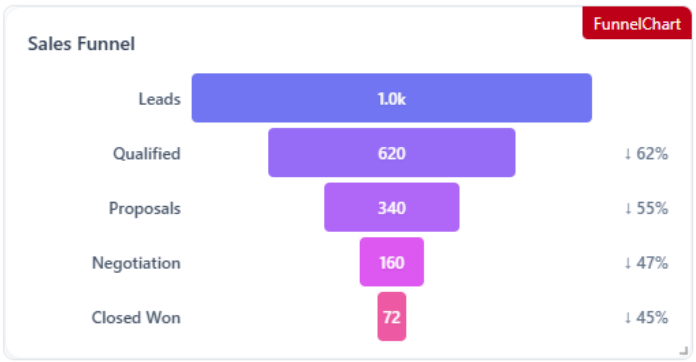
- Title: Tickets Resolved
- Current Value: 340
- Target Value: 500
- Unit (prefix): (vacío, ya que no es un valor monetario)
- Period Label: September
- Trend Label: -8%
- Trend Direction: Down

La barra de progreso se calcula automáticamente a partir de Current Value y Target Value, por lo que no requiere configuración manual. Trend Direction es puramente visual (colorea el Trend Label) y es independiente del signo que se escriba en el texto, por lo que ambos deben coincidir manualmente para que el indicador sea coherente.

Nexus — Elemento: Funnel Chart

¿Qué es?

Funnel Chart es un componente de tipo Data. Muestra un embudo de conversión con distintas etapas, donde cada una se representa como una barra cuyo ancho es proporcional a su valor respecto a la primera etapa. Al arrastrarlo al lienzo viene con un ejemplo precargado (“Sales Funnel”, con 5 etapas: Leads, Qualified, Proposals, Negotiation y Closed Won).



Propiedades

Al seleccionar Funnel Chart en el lienzo, el panel derecho muestra “Properties of FunnelChart”, organizado en tres pestañas: Content, Style y Behavior.

- **Component ID:** identificador único del componente, autogenerado (ej. fnl_f31ab12c).
- **Visibility → Hidden on load:** casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo Show.
- **Selector S / M / L:** ajusta el tamaño de visualización del panel de propiedades.

Pestaña Content:

- **Title:** título del embudo (ej. “Sales Funnel”).
- **Stages (JSON):** campo de código donde se definen las etapas del embudo en formato JSON. Cada etapa se compone de:

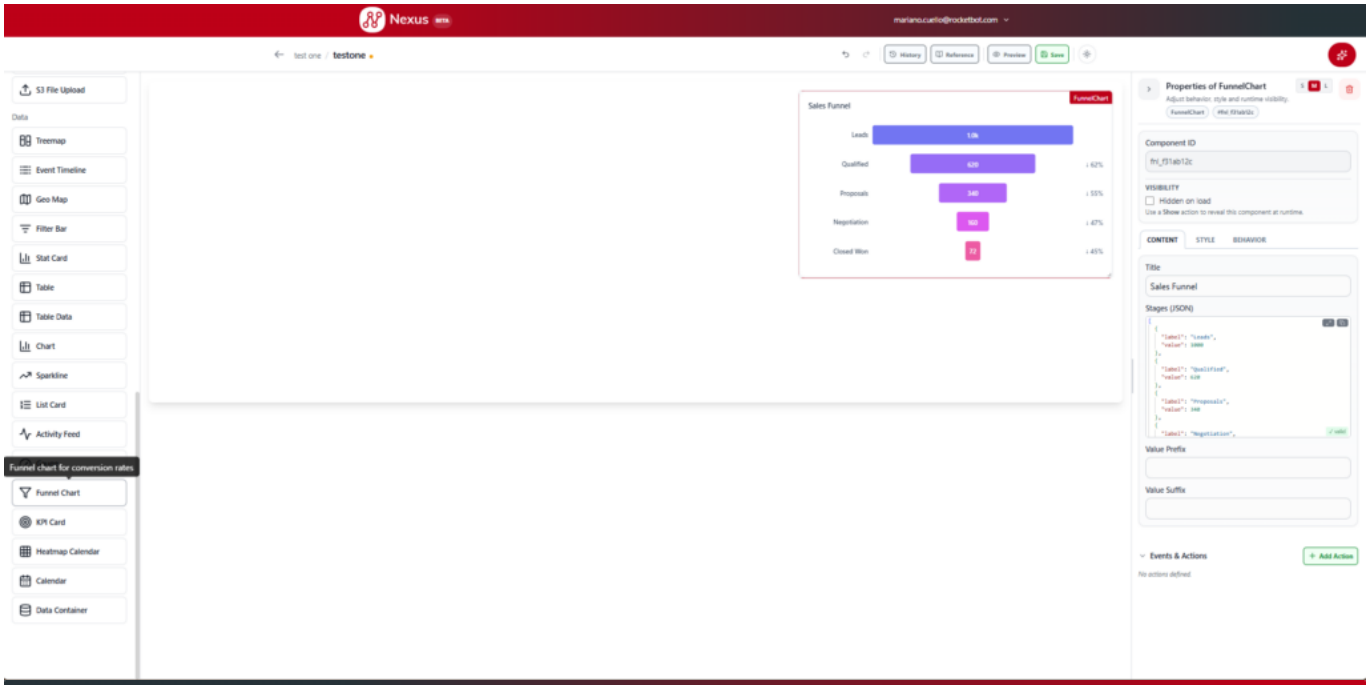
- label: nombre de la etapa (ej. “Leads”).value: valor numérico asociado (ej. 1000).

Ejemplo de estructura:

```
[
  {
    "label": "Leads",
    "value": 1000
  },
  {
    "label": "Qualified",
    "value": 620
  },
  {
    "label": "Proposals",
    "value": 340
  },
  {
    "label": "Negotiation",
    "value": 160
  },
  {
    "label": "Closed Won",
    "value": 72
  }
]
```

Cuenta con el editor de código expandido para más comodidad, con pestañas HTML/CSS/JS/JSON y botones Format y Copy. El editor valida el JSON en tiempo real (marca “✓ valid”).

- **Value Prefix:** texto que se antepone al valor mostrado en cada etapa.
- **Value Suffix:** texto que se agrega después del valor mostrado en cada etapa.



Expanded Code Editor HTML CSS JS JSON Format Copy

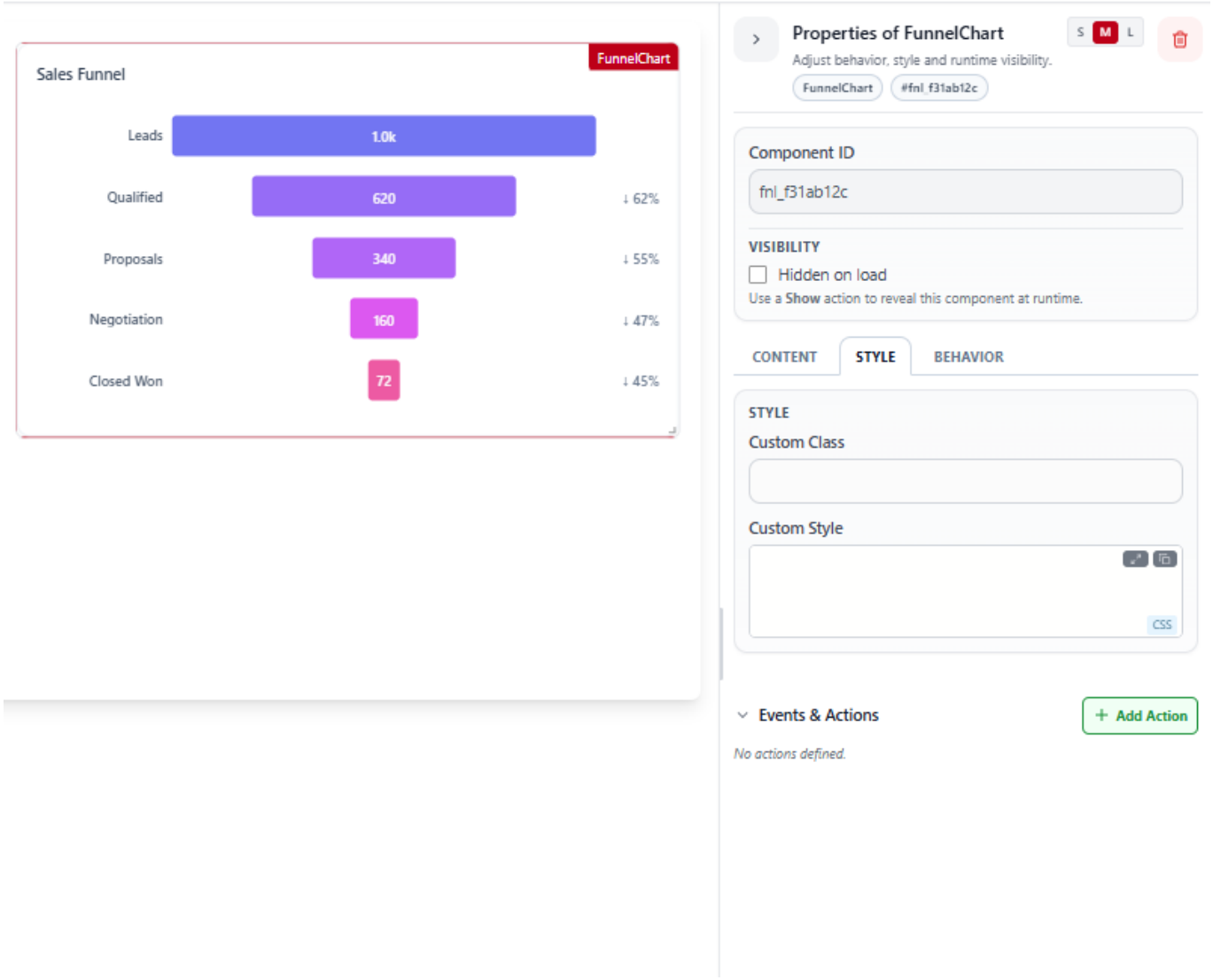
```json
[
  {
    "label": "Leads",
    "value": 1000
  },
  {
    "label": "Qualified",
    "value": 620
  },
  {
    "label": "Proposals",
    "value": 340
  },
  {
    "label": "Negotiation",
    "value": 160
  },
  {
    "label": "Closed Won",
    "value": 72
  }
]
```

Ctrl+Space Autocompletado / Snippets · Shift+Alt+F Formatear · Esc Volver

Done

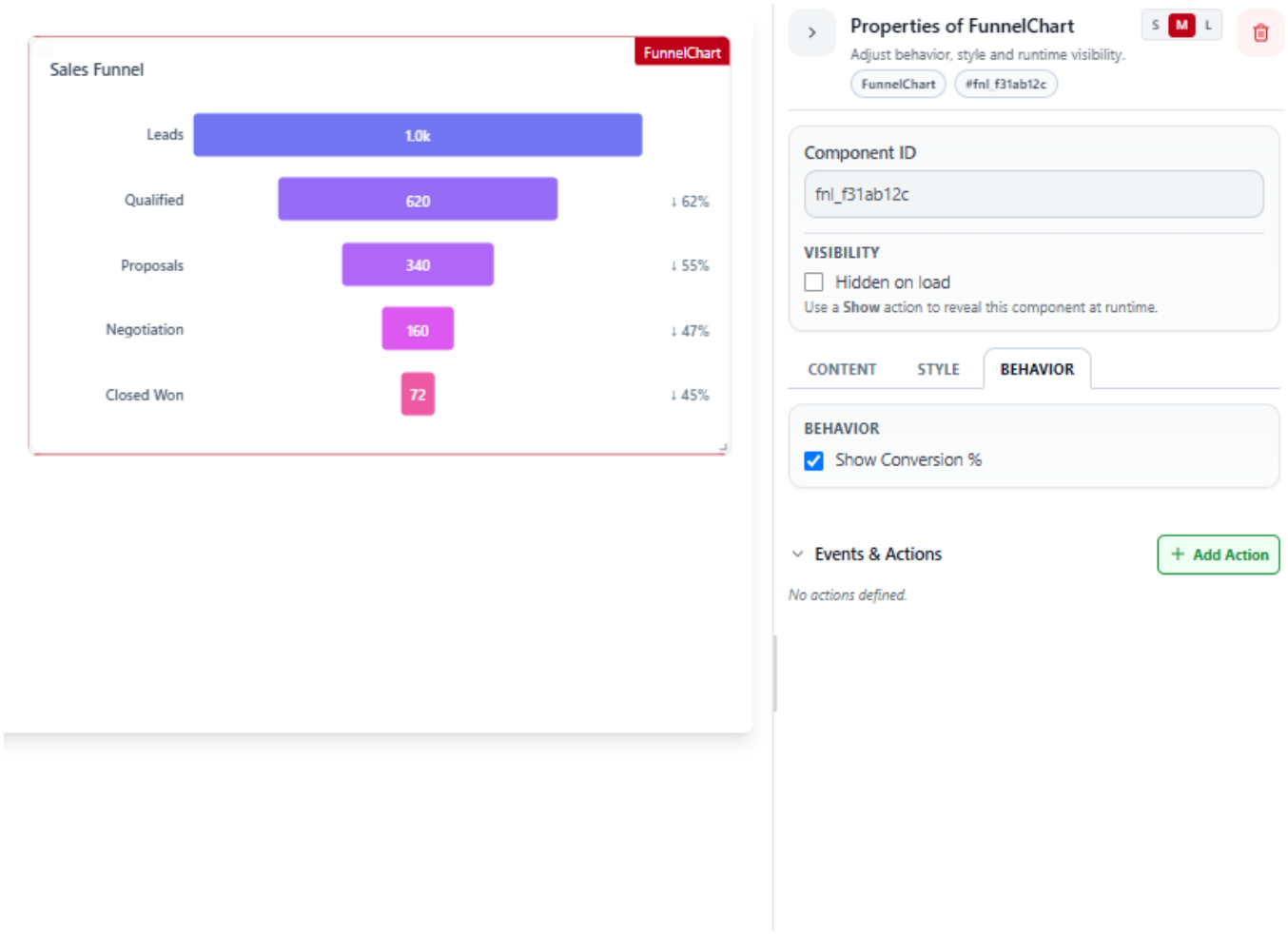
Pestaña Style:

- **Custom Class:** clase CSS personalizada.
- **Custom Style:** CSS personalizado aplicado directamente a este componente.



Pestaña Behavior:

- **Show Conversion %:** casilla que muestra u oculta, junto a cada etapa (a excepción de la primera), el porcentaje de caída respecto a la etapa anterior (ej. "↓ 62%").
- **Events & Actions:** sistema para definir comportamientos interactivos. Ver documentación aparte: Nexus — Events & Actions .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del lienzo.



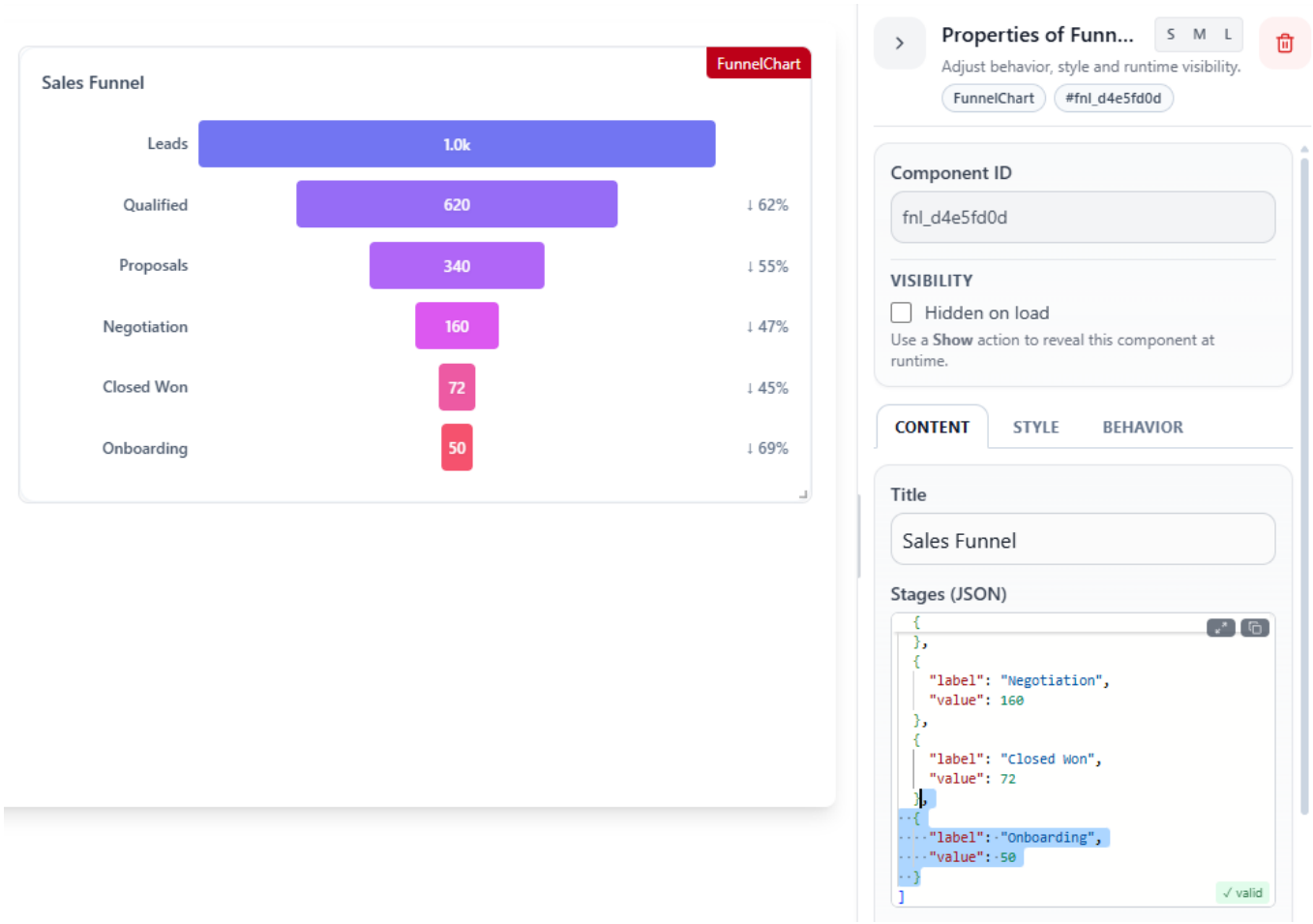
Ejemplo de uso

Para agregar una nueva etapa al embudo, hay que ubicarse dentro del array en Stages (JSON), agregar una coma después del último objeto de cierre }, y pegar el nuevo bloque a continuación:

```
[
 {
  "label": "Onboarding",
  "value": 50
 }
]
```

> ⚠ Importante: la coma va pegada a la llave } de la etapa anterior, no al principio del bloque nuevo.

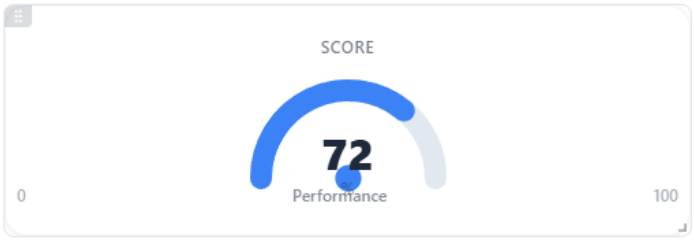
Con Show Conversion % se puede visualizar rápidamente en qué etapa del embudo se produce la mayor pérdida de conversión, útil para detectar cuellos de botella en un proceso de ventas u onboarding. Con Value Prefix y Value Suffix se puede dar formato a los valores (por ejemplo, agregando “$” como prefijo si el embudo representa montos en lugar de cantidades).



Nexus — Elemento: Gauge

¿Qué es?

Gauge es un componente de tipo Data. Muestra un medidor circular (tipo velocímetro) que representa un valor dentro de un rango definido, con una etiqueta y una unidad de medida. Al arrastrarlo al lienzo viene con un ejemplo precargado (“Score”, valor 72 sobre un rango de 0 a 100, con la etiqueta “Performance”).



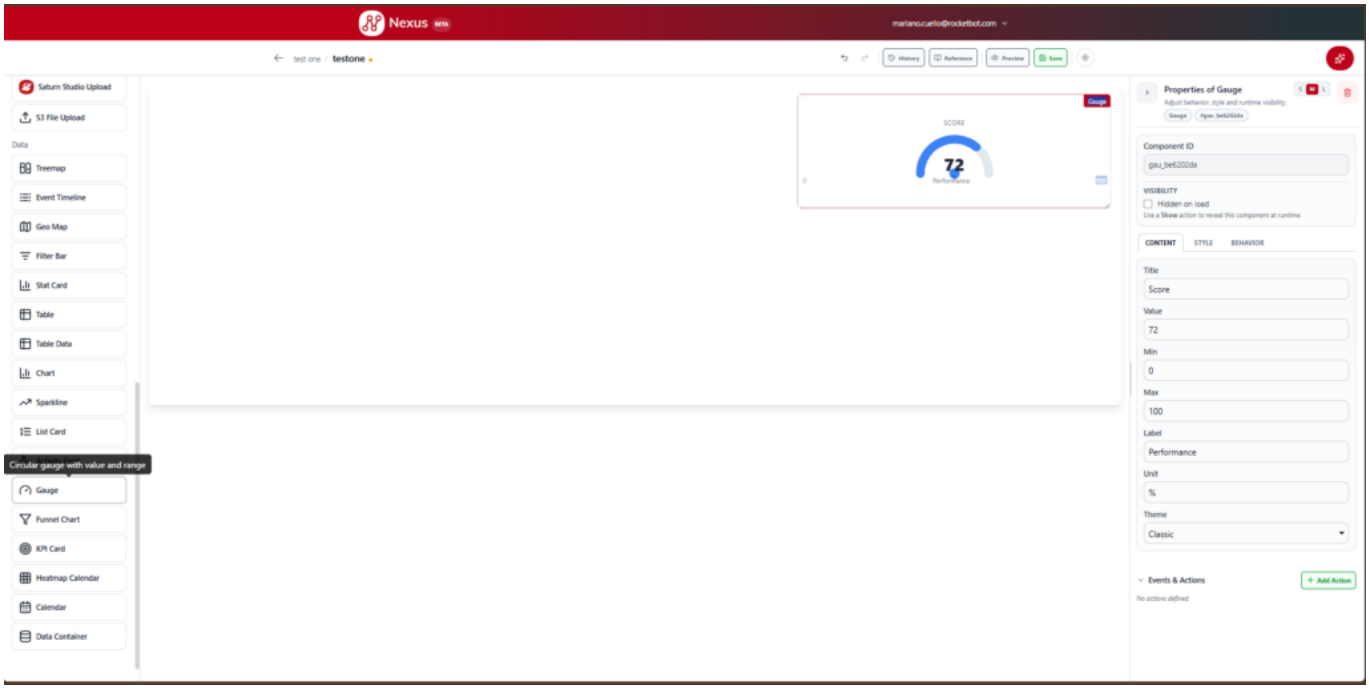
Propiedades

Al seleccionar Gauge en el lienzo, el panel derecho muestra “Properties of Gauge”, organizado en tres pestañas: Content, Style y Behavior.

- **Component ID:** identificador único del componente, autogenerado (ej. gau_be6202da).
- **Visibility → Hidden on load:** casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo Show.
- **Selector S / M / L:** ajusta el tamaño de visualización del panel de propiedades.

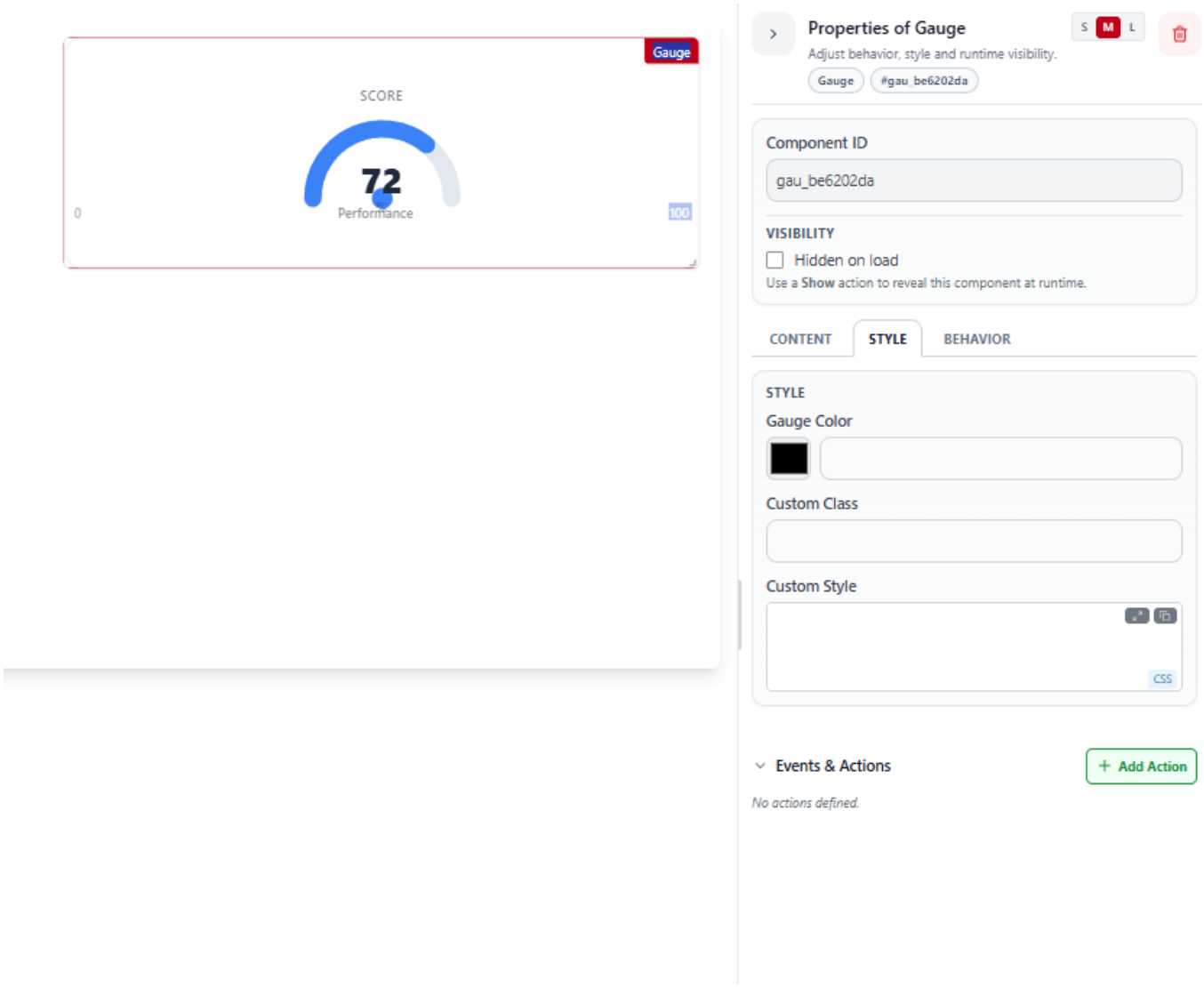
Pestaña Content:

- **Title:** título del medidor (ej. “Score”).
- **Value:** valor actual que se representa en el medidor (ej. 72).
- **Min:** valor mínimo del rango (ej. 0).
- **Max:** valor máximo del rango (ej. 100).
- **Label:** etiqueta descriptiva que se muestra debajo del valor (ej. “Performance”).
- **Unit:** unidad de medida que acompaña al valor (ej. “%”).
- **Theme:** menú desplegable con el estilo visual del medidor. Opciones disponibles:
  - Classic
  - Dark
  - Gradient



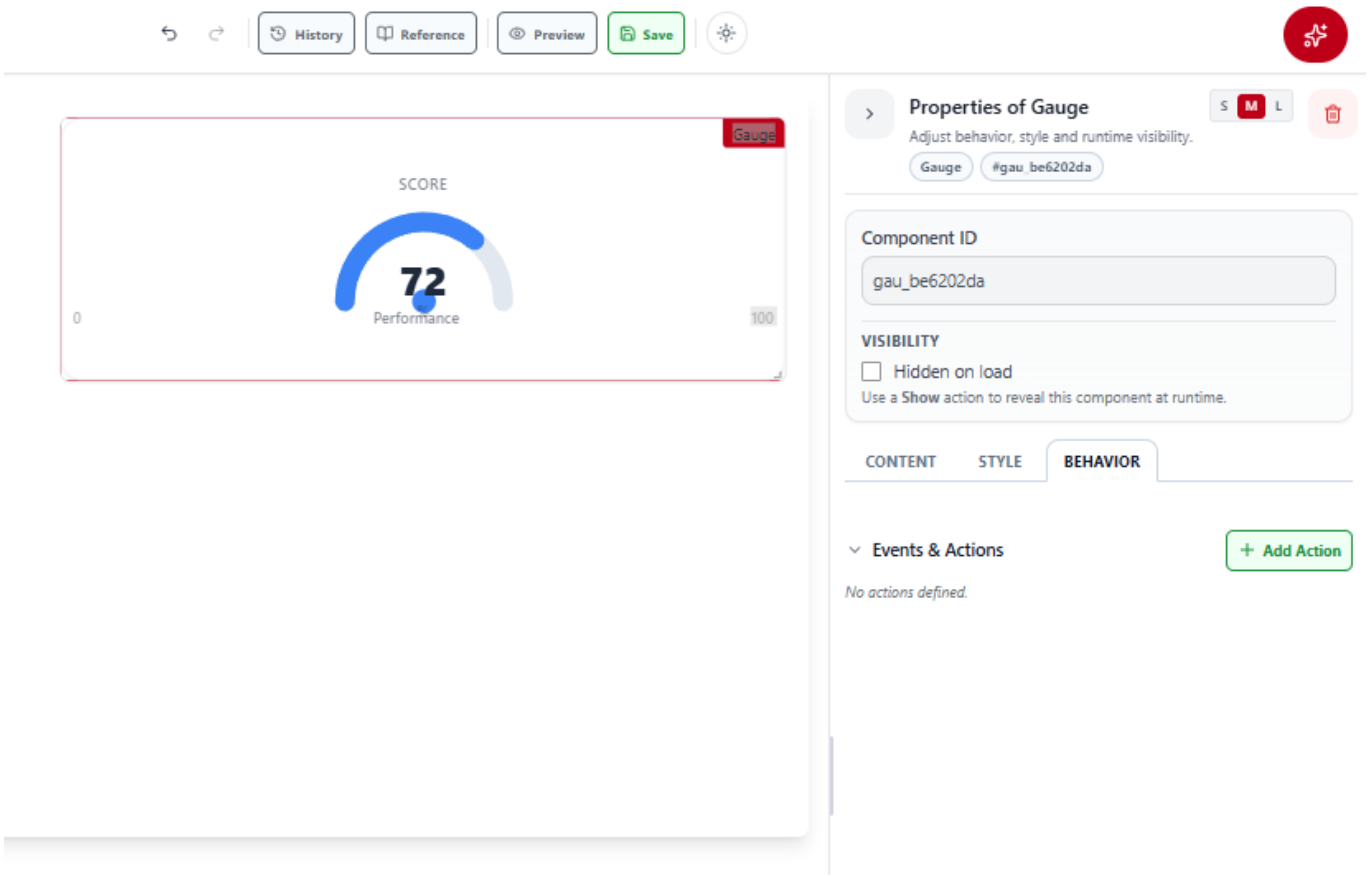
Pestaña Style:

- **Gauge Color:** selector de color para el arco del medidor (color picker + campo de texto para código de color).
- **Custom Class:** clase CSS personalizada.
- **Custom Style:** CSS personalizado aplicado directamente a este componente.



Pestaña Behavior:

- **Events & Actions:** sistema para definir comportamientos interactivos. Ver documentación aparte: [Nexus — Events & Actions]() .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del lienzo.

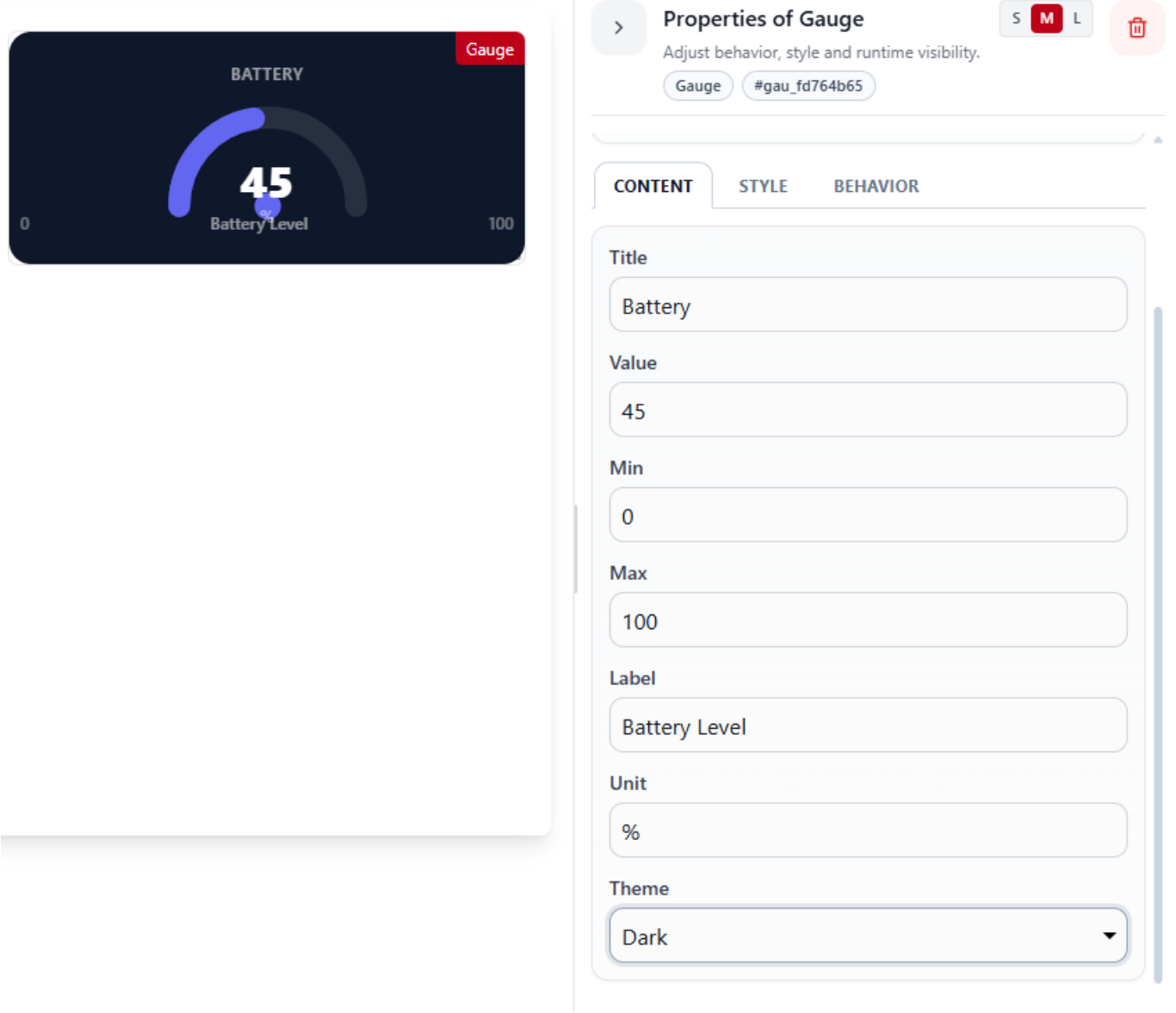


Ejemplo de uso

Para usar Gauge como indicador de una métrica propia, basta con reemplazar los valores de Value, Min, Max, Label y Unit según el dato que se quiera representar. Por ejemplo, para mostrar el nivel de batería de un dispositivo:

- Title: Battery
- Value: 45
- Min: 0
- Max: 100
- Label: Battery Level
- Unit: %

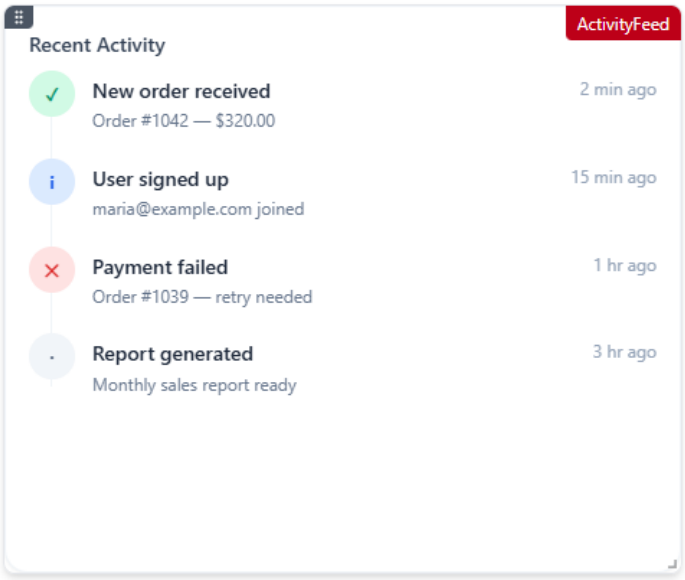
Con Theme se puede adaptar el estilo visual del medidor al diseño general del tablero (claro, oscuro o degradado), y con Gauge Color se puede personalizar el color del arco independientemente del tema elegido.



Nexus — Elemento: Activity Feed

¿Qué es?

Activity Feed es un componente de tipo Data. Muestra un listado cronológico de eventos o notificaciones, cada uno con un ícono de estado, título, descripción y una marca de tiempo. Al arrastrarlo al lienzo viene con un ejemplo precargado (“Recent Activity”, con 4 eventos de muestra: nuevo pedido, usuario registrado, pago fallido y reporte generado).



Propiedades

Al seleccionar Activity Feed en el lienzo, el panel derecho muestra “Properties of ActivityFeed”, organizado en tres pestañas: Content, Style y Behavior.

- **Component ID:** identificador único del componente, autogenerado (ej. act_c5b67137).
- **Visibility → Hidden on load:** casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo Show.
- **Selector S / M / L:** ajusta el tamaño de visualización del panel de propiedades (no afecta al componente en sí).
- **Ícono de IA** (círculo rojo con destello, arriba a la derecha): abre el asistente de inteligencia artificial de Nexus para ayudar en la edición.

Pestaña Content:

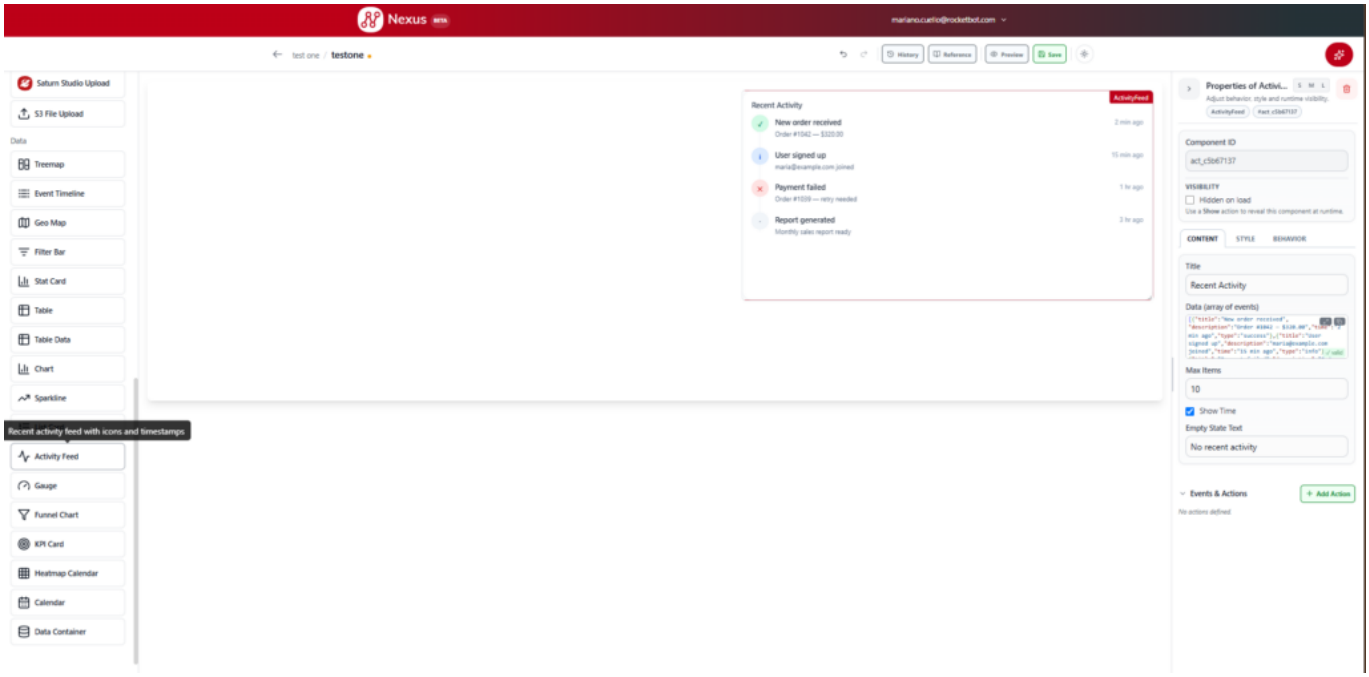
- **Title:** título del listado (ej. “Recent Activity”).
- **Data (array of events):** campo de código donde se definen los eventos en formato JSON. Cada evento se compone de:
  - title: título del evento (ej. “New order received”).description: texto secundario debajo del título (ej. “Order #1042 — $320.00”).time: texto de tiempo relativo (ej. “2 min ago”).type: determina el ícono y color asociado. Valores vistos: "success" (✓ verde), "info" (azul), "danger" (× rojo), "default" (guion gris). Este campo es de texto libre: se puede modificar sin restricción a esos cuatro valores.

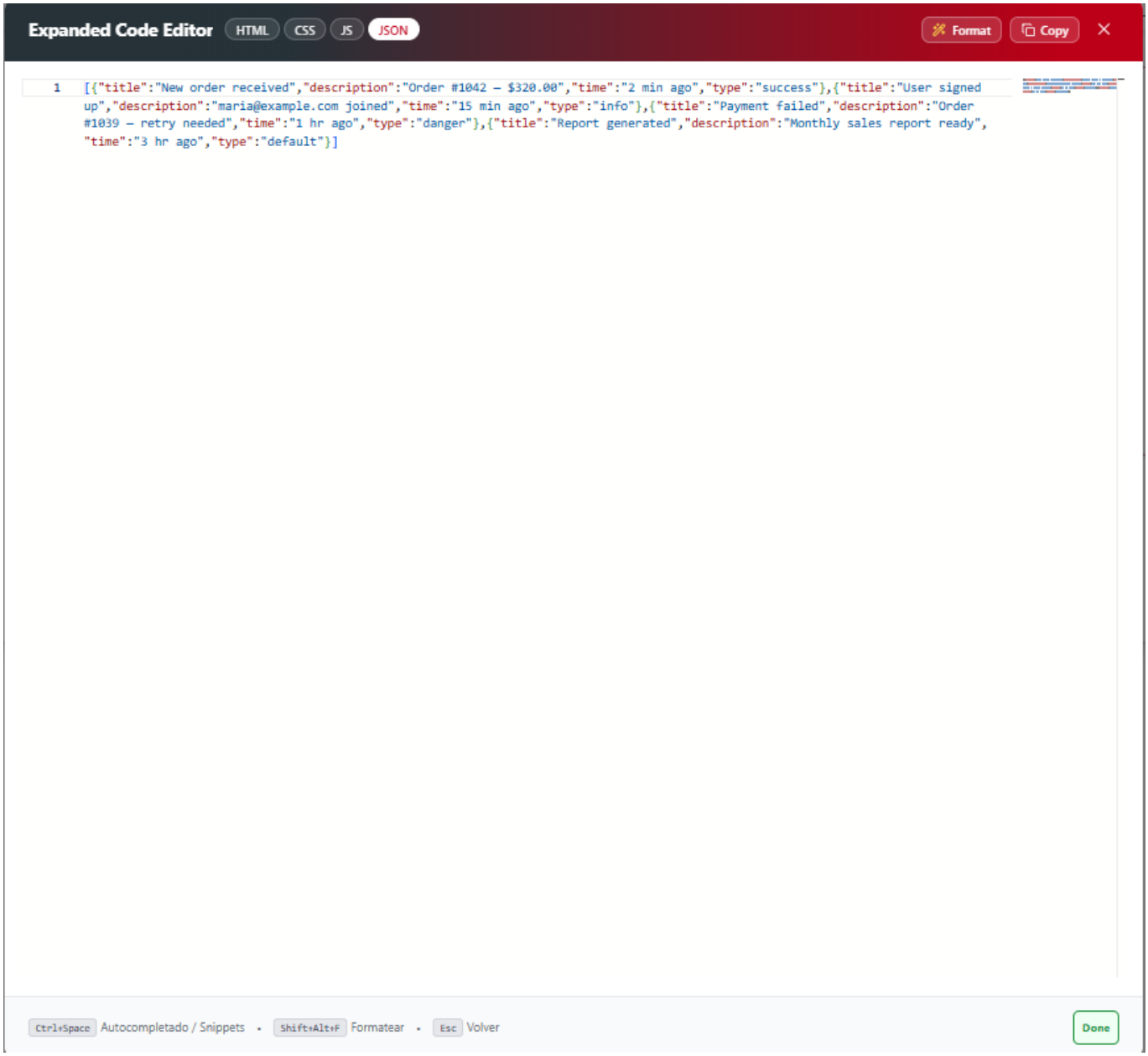
  Ejemplo de estructura:

```
[
  {
    "title": "New order received",
    "description": "Order #1042 — $320.00",
    "time": "2 min ago",
    "type": "success"
  },
  {
    "title": "User signed up",
    "description": "maria@example.com joined",
    "time": "15 min ago",
    "type": "info"
  }
]
```

Cuenta con el editor de código expandido para más comodidad, con pestañas HTML/CSS/JS/JSON y botones Format y Copy. El editor valida el JSON en tiempo real (marca "✓ valid").

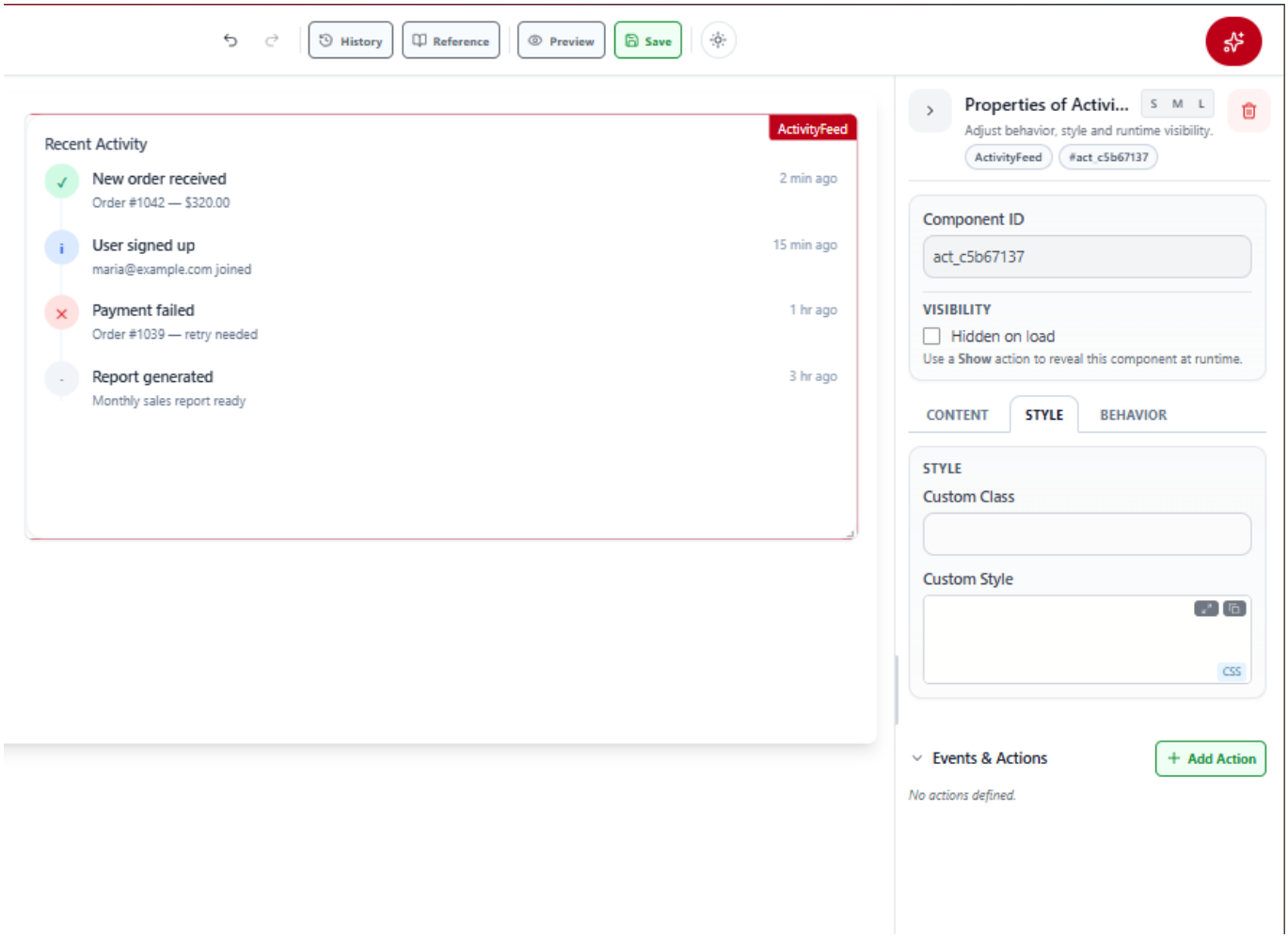
- **Max Items:** cantidad máxima de eventos que se muestran (valor por defecto: 10).
- **Show Time:** casilla que muestra u oculta la marca de tiempo de cada evento.
- **Empty State Text:** texto que se muestra cuando no hay eventos (ej. "No recent activity").





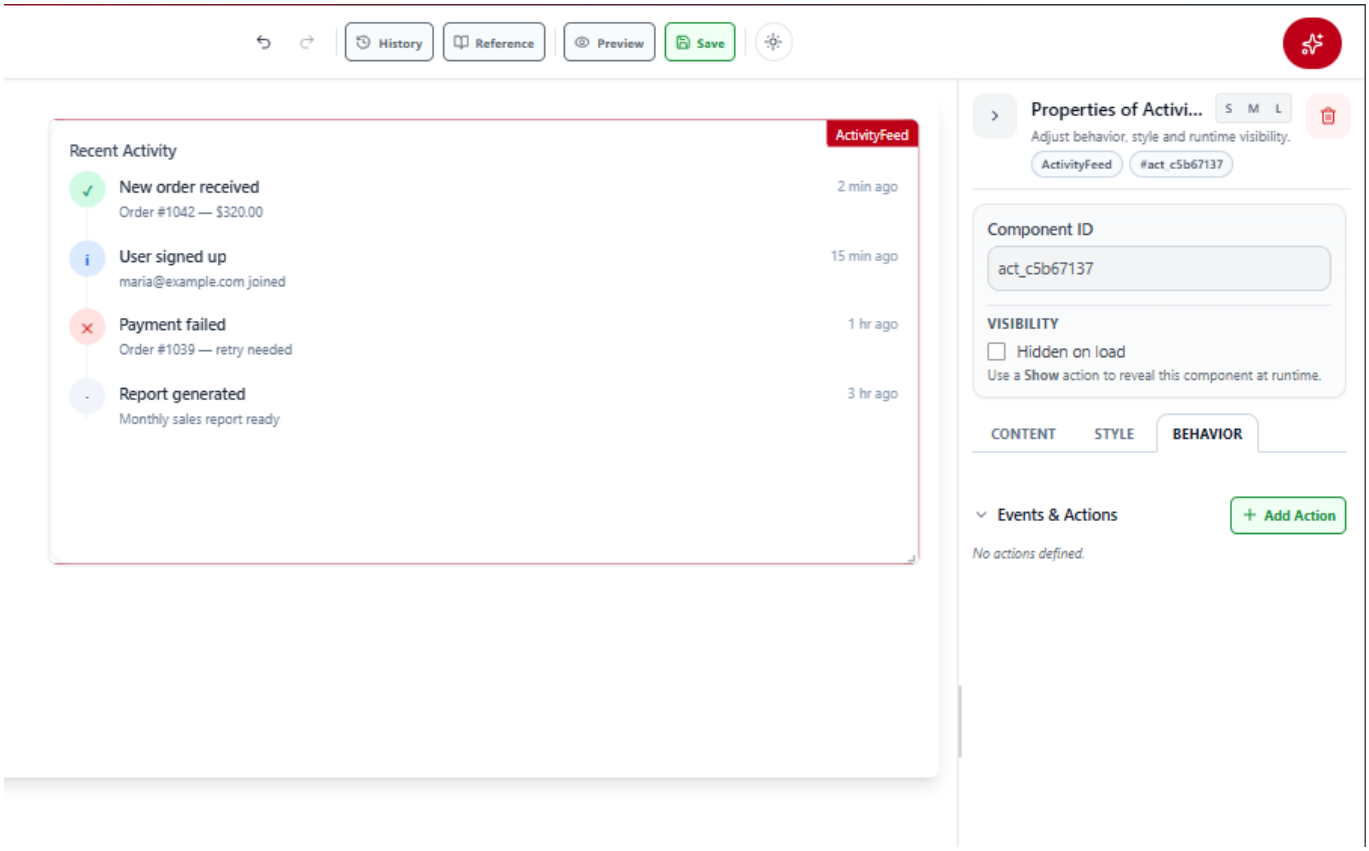
Pestaña Style:

- **Custom Class:** clase CSS personalizada.
- **Custom Style:** CSS personalizado aplicado directamente a este componente.



Pestaña Behavior:

- **Events & Actions:** sistema para definir comportamientos interactivos. Ver documentación aparte: [Nexus — Events & Actions]() .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del lienzo.

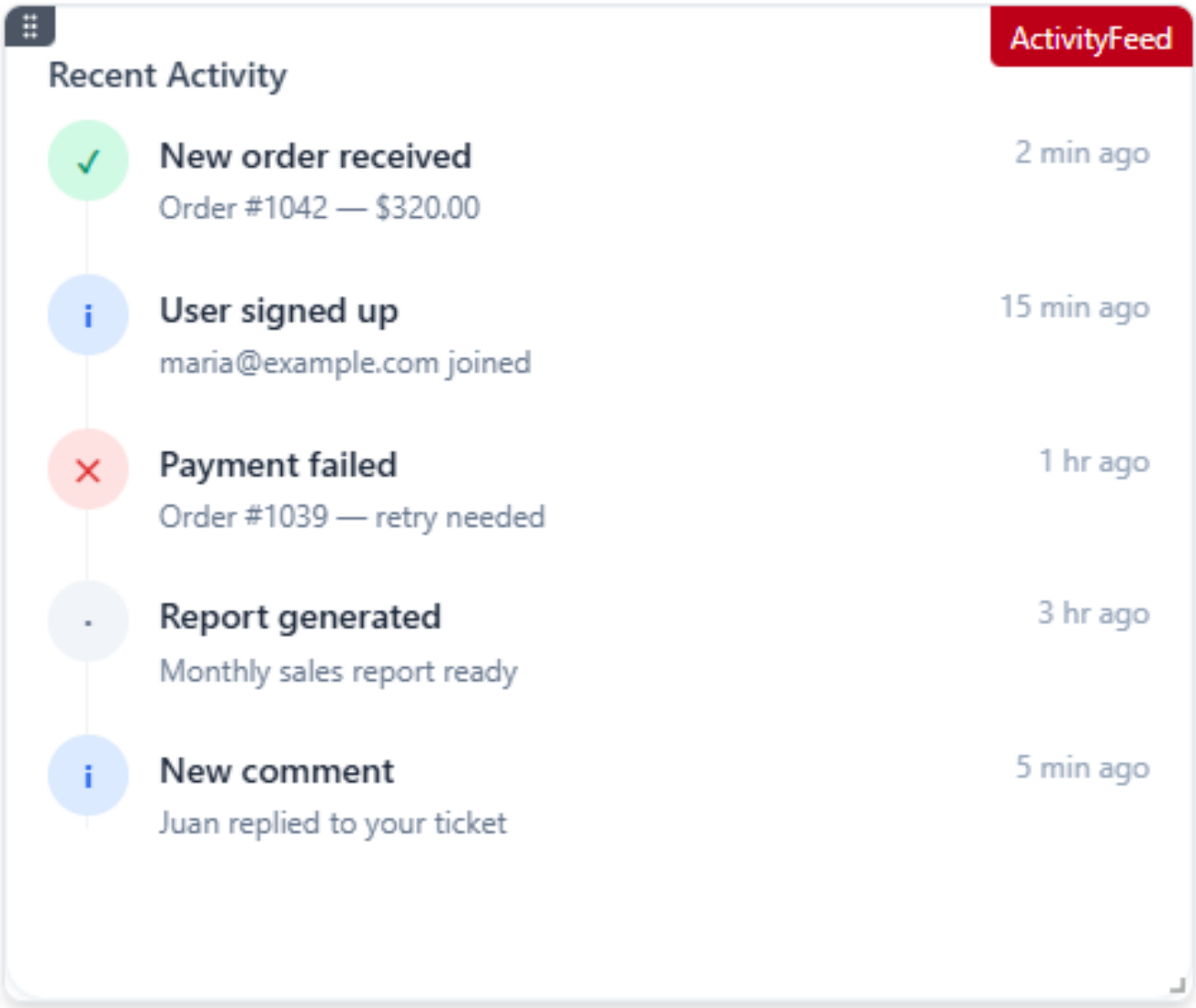


Ejemplo de uso

Para agregar un nuevo evento al listado, hay que ubicarse dentro del array en Data, agregar una coma después del último objeto de cierre }, y pegar el nuevo bloque a continuación:

```
[
  {
  "title": "New comment",
  "description": "Juan replied to your ticket",
  "time": "5 min ago",
  "type": "info"
  }
]
```

> ⚠ Importante: la coma va pegada a la llave } del evento anterior, no al principio del bloque nuevo.

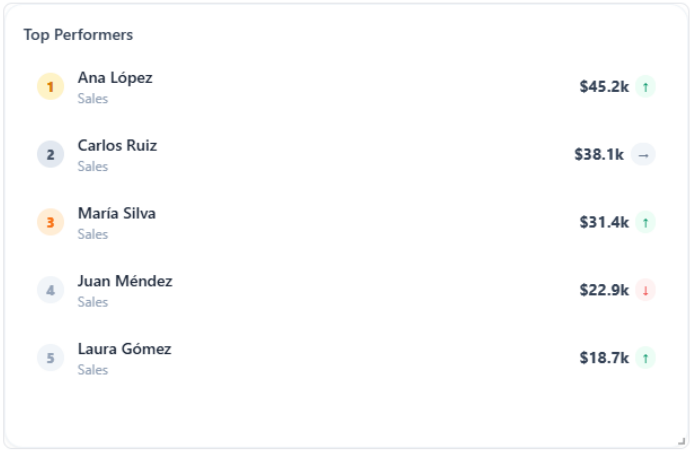


Con type se indica visualmente la naturaleza del evento (éxito, información, error o neutro) mediante un ícono y color diferenciado, aunque al ser un campo libre también se pueden usar otros valores según la necesidad. Con Show Time se puede optar por ocultar la marca de tiempo si no es relevante para el caso de uso.

Nexus — Elemento: List Card

¿Qué es?

List Card es un componente de tipo **Data**. Muestra una lista de ítems tipo ranking, cada uno con nombre, subtítulo, valor y una tendencia (flecha de subida, bajada o neutra). Al arrastrarlo al lienzo viene con un ejemplo precargado (“Top Performers”, 5 vendedores con sus ventas).



Propiedades

Al seleccionar List Card en el lienzo, el panel derecho muestra **“Properties of ListCard”**, organizado en tres pestañas: **Content**, **Style** y **Behavior**.

- **Component ID**: identificador único del componente, autogenerado (ej. lst_420dde73).
- **Visibility → Hidden on load**: casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo **Show** (ver Events & Actions).

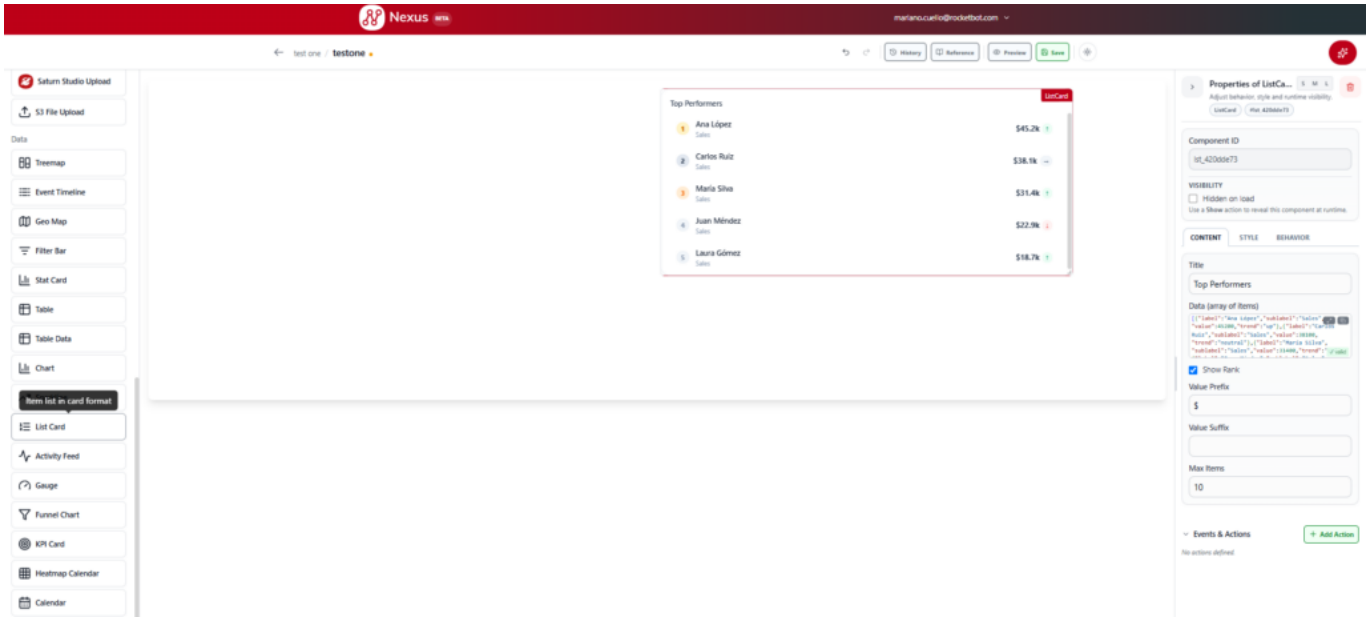
Pestaña Content:

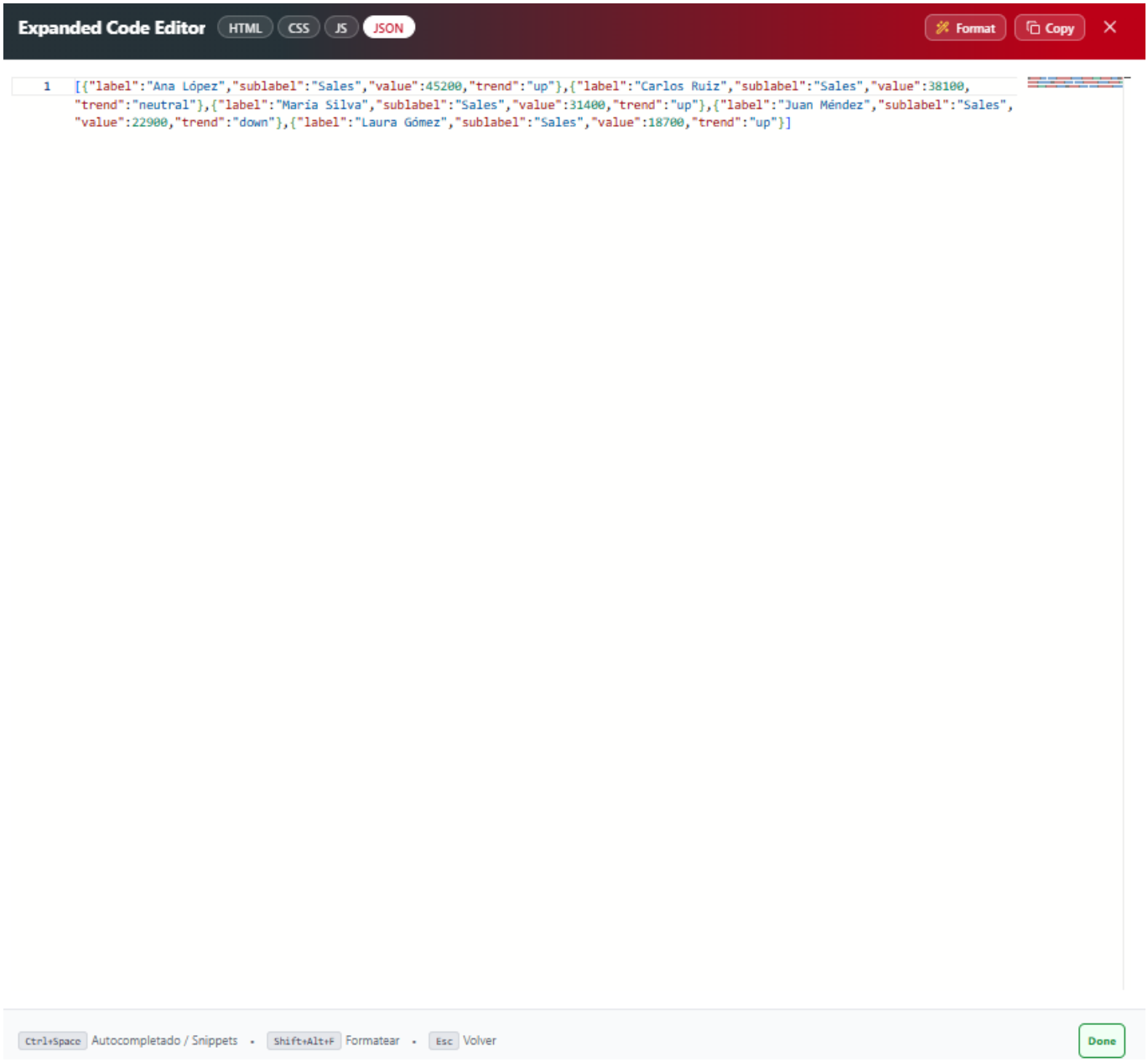
- **Title**: título de la lista (ej. “Top Performers”).
- **Data (array of items)**: campo de código donde se definen los ítems en formato JSON. Cada ítem se compone de:
  - label: nombre del ítem (ej. “Ana López”).
  - sublabel: texto secundario debajo del nombre (ej. “Sales”).
  - value: valor numérico asociado (ej. 45200).
  - trend: dirección de la tendencia. Valores vistos: "up", "down", "neutral".

  Cuenta con el editor de código expandido para más comodidad. El editor valida el JSON en tiempo real (marca “✓ valid” cuando el formato es correcto).

- **Show Rank**: casilla que muestra u oculta el número de posición/ranking junto a cada ítem.
- **Value Prefix**: texto que se antepone al valor mostrado (ej. $).

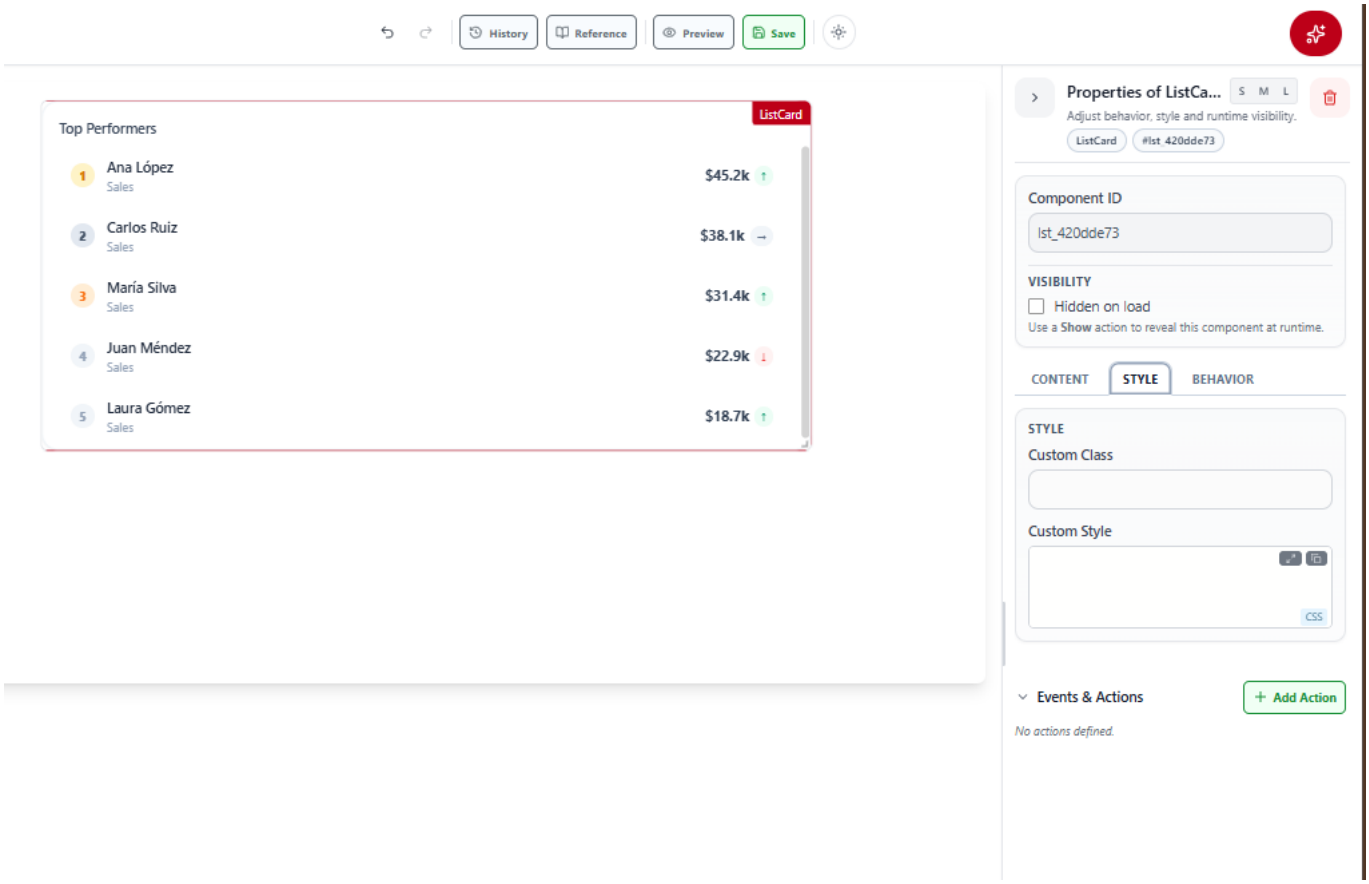
- **Value Suffix**: texto que se agrega después del valor mostrado.
- **Max Items**: cantidad máxima de ítems que se muestran en la lista (valor por defecto: 10).





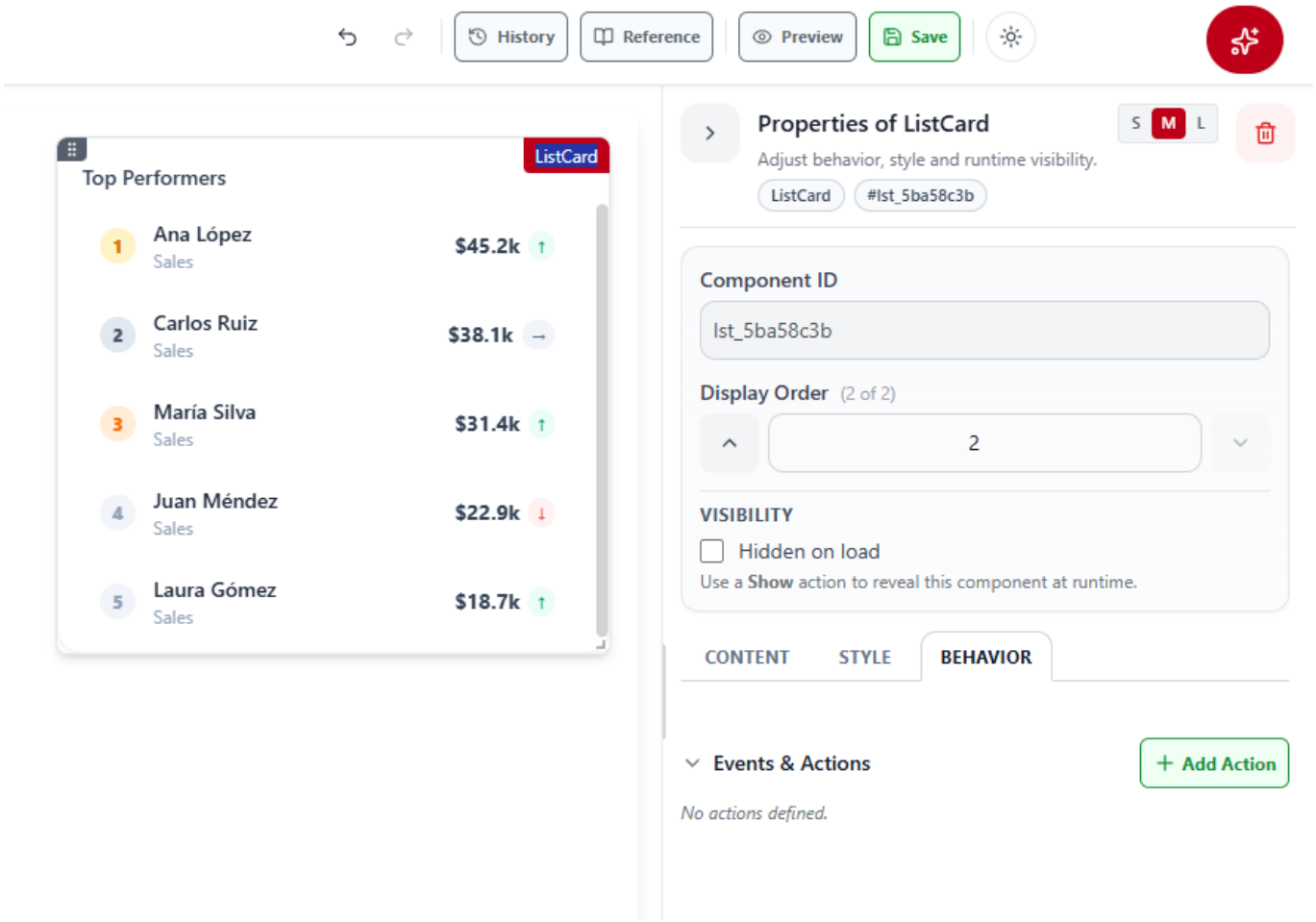
Pestaña Style:

- **Custom Class**: clase CSS personalizada.
- **Custom Style**: CSS personalizado aplicado directamente a este componente.



Pestaña Behavior:

- **Events & Actions**: sistema para definir comportamientos interactivos. Ver documentación aparte: **Nexus — Events & Actions** .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del lienzo.

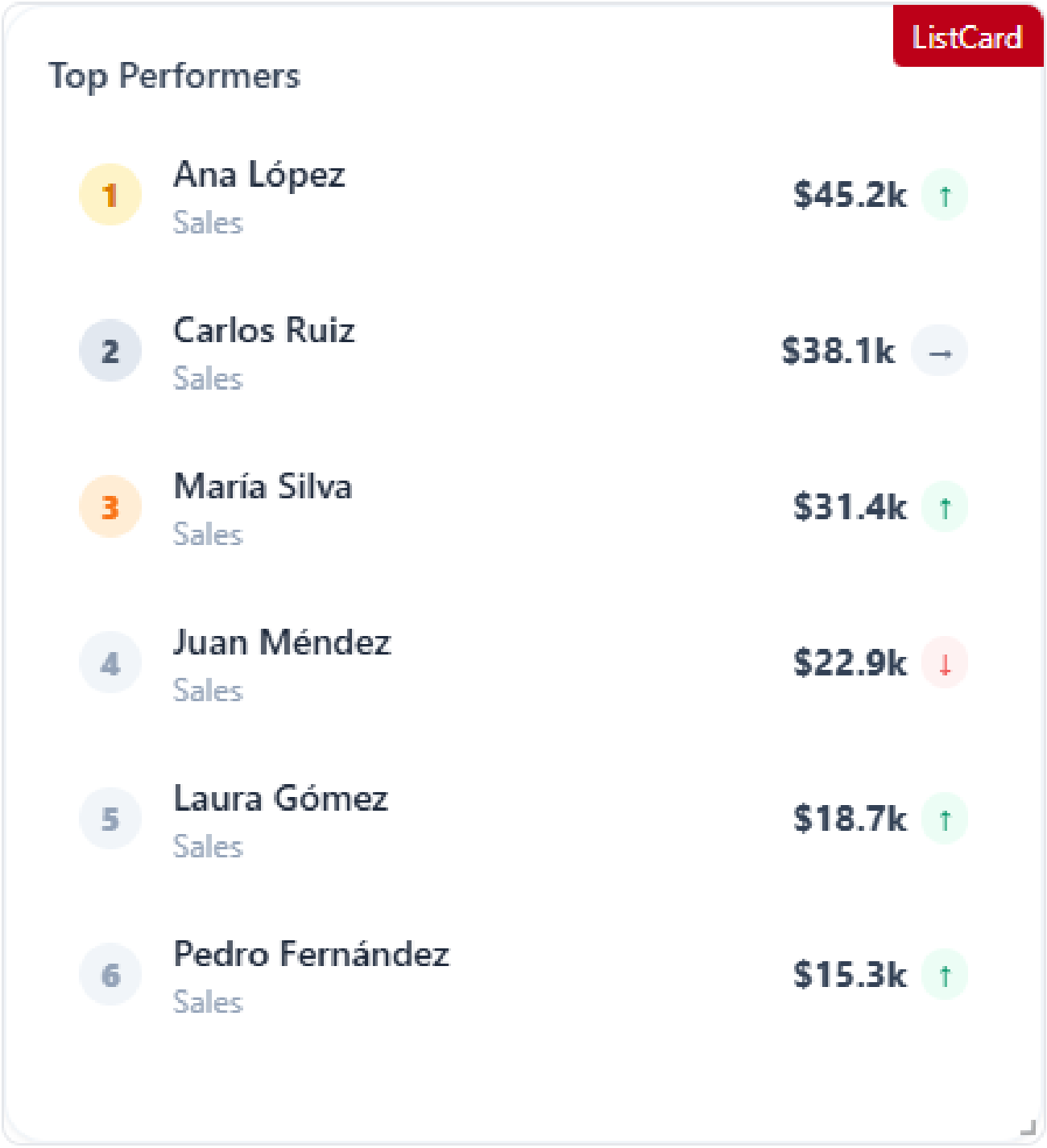


Ejemplo de uso

Para agregar un nuevo ítem a la lista, hay que ubicarse dentro del array en **Data**, agregar una coma después del último objeto }, y pegar el nuevo bloque a continuación:

```
[{"label":"Pedro Fernández","sublabel":"Sales","value":15300,"trend":"up"}]
```

> ⚠ Importante: la coma va pegada a la llave } del ítem anterior, no al principio del bloque nuevo.



Con trend se indica visualmente si ese valor mejoró (up), empeoró (down) o se mantuvo estable (neutral) respecto a un período anterior. Con **Value Prefix** y **Value Suffix** se da formato al valor sin modificar el JSON (por ejemplo, $ como prefijo para montos en dólares). Con **Show Rank** se puede optar por mostrar u ocultar la numeración de posición, útil si la lista no representa necesariamente un ranking sino solo un listado de ítems con valores.

Nexus — Elemento: Sparkline

¿Qué es?

Sparkline es un componente de tipo **Data**. Muestra un mini gráfico de tendencia, minimalista y sin ejes visibles, ideal para mostrar la evolución de un valor en poco espacio. Al arrastrarlo al lienzo viene con un ejemplo precargado (gráfico de línea, “Last 10 days”).

Propiedades

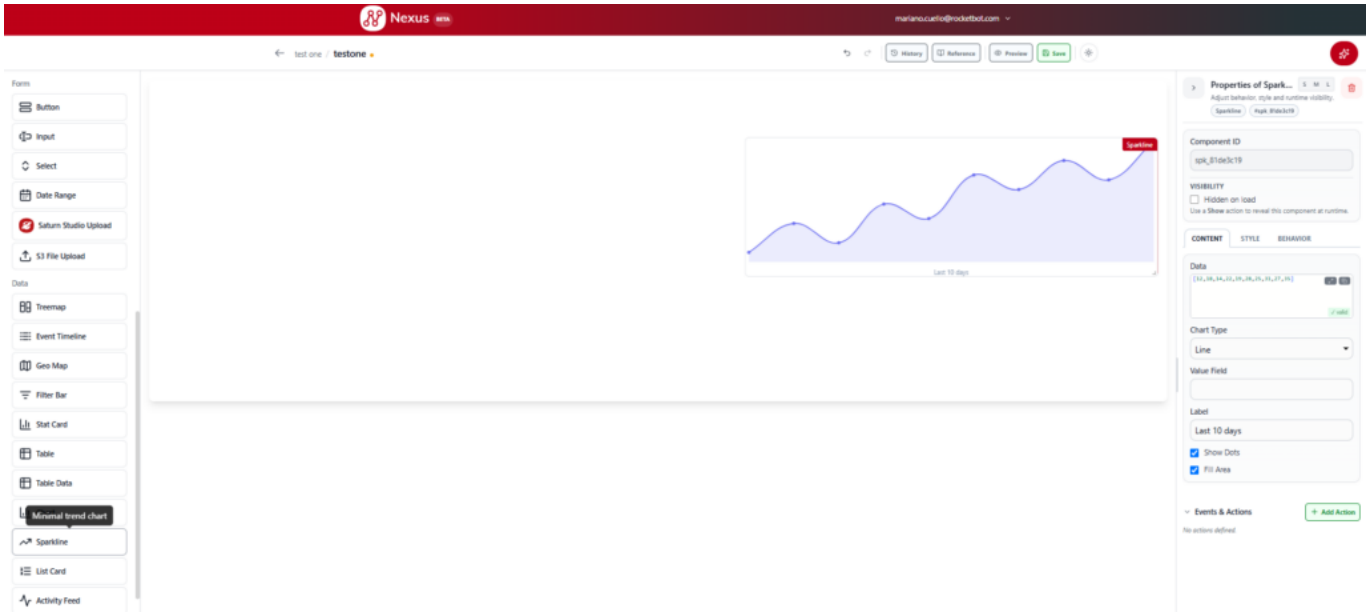
Al seleccionar Sparkline en el lienzo, el panel derecho muestra **“Properties of Sparkline”**, organizado en tres pestañas: **Content**, **Style** y **Behavior**.

- **Component ID**: identificador único del componente, autogenerado (ej. spk_81de3c19).
- **Visibility → Hidden on load**: casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo **Show** (ver Events & Actions).

Pestaña Content:

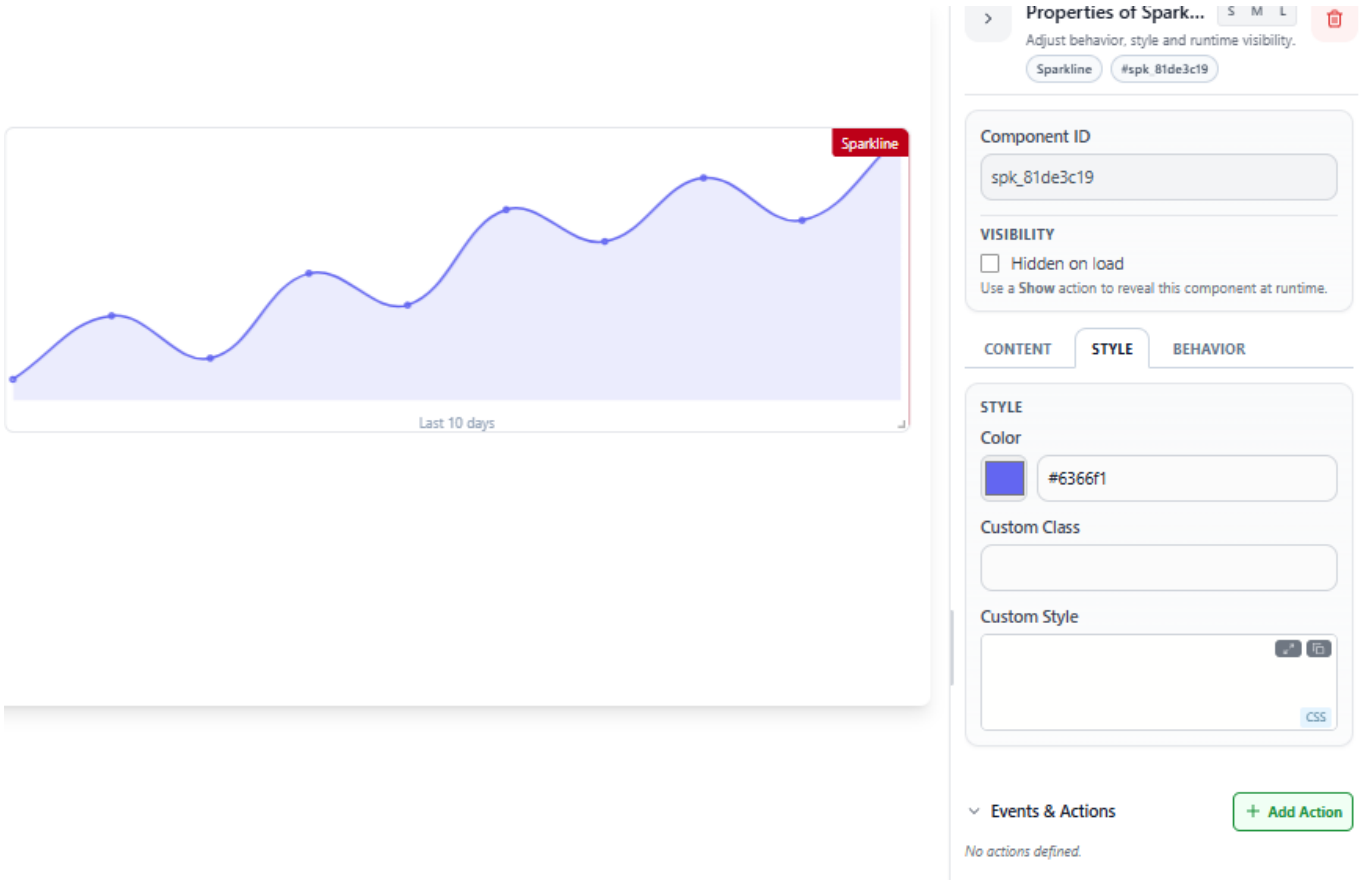
- **Data**: array simple de números que representa los puntos del gráfico (ej. [12,18,14,22,19,28,25,31,27,35]).
- **Chart Type**: menú desplegable que define el tipo de gráfico. Opciones: **Line**, **Bar**.
- **Value Field**: nombre del campo a usar como valor, cuando los datos provienen de una estructura más compleja (objetos) en vez de un array simple de números (vacío por defecto).
- **Label**: texto descriptivo debajo del gráfico (ej. “Last 10 days”).
- **Show Dots**: casilla que muestra u oculta los puntos sobre la línea del

gráfico.
- **Fill Area**: casilla que rellena de color el área debajo de la línea del gráfico.



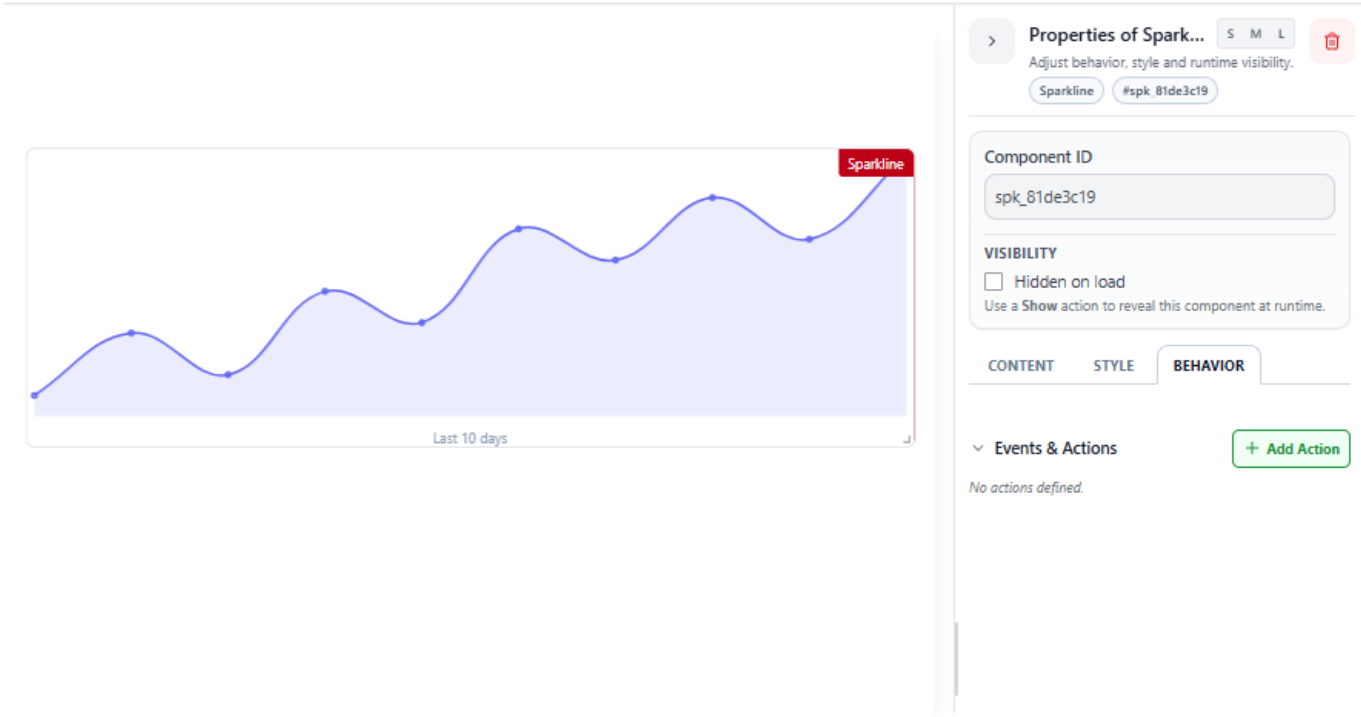
Pestaña Style:

- **Color**: color de la línea/barras del gráfico, con selector visual y campo hexadecimal (ej. #6366f1).
- **Custom Class**: clase CSS personalizada.
- **Custom Style**: CSS personalizado aplicado directamente a este componente.



Pestaña Behavior:

- **Events & Actions**: sistema para definir comportamientos interactivos. Ver documentación aparte: [Nexus — Events & Actions]() .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del lienzo.



Ejemplo de uso

Sparkline es ideal para acompañar una métrica (por ejemplo, dentro de una **Stat Card** o al lado de un **KPI Card**) mostrando su tendencia reciente sin ocupar el espacio de un gráfico completo. Con **Data** se carga la serie de valores directamente como array de números (ej. [12,18,14,22,19,28,25,31,27,35]), y con **Chart Type** se elige si se representa como línea continua o como barras. Activando **Fill Area** y **Show Dots** se le da más presencia visual al gráfico, útil cuando el Sparkline es un elemento protagonista y no solo un acompañamiento discreto.

Nexus — Elemento: Chart

¿Qué es?

Chart es un componente de tipo **Data**. Muestra un gráfico de datos con múltiples tipos de visualización disponibles. Al arrastrarlo al lienzo viene con un ejemplo precargado (“Monthly Revenue”, gráfico de barras con datos de Enero a Mayo).

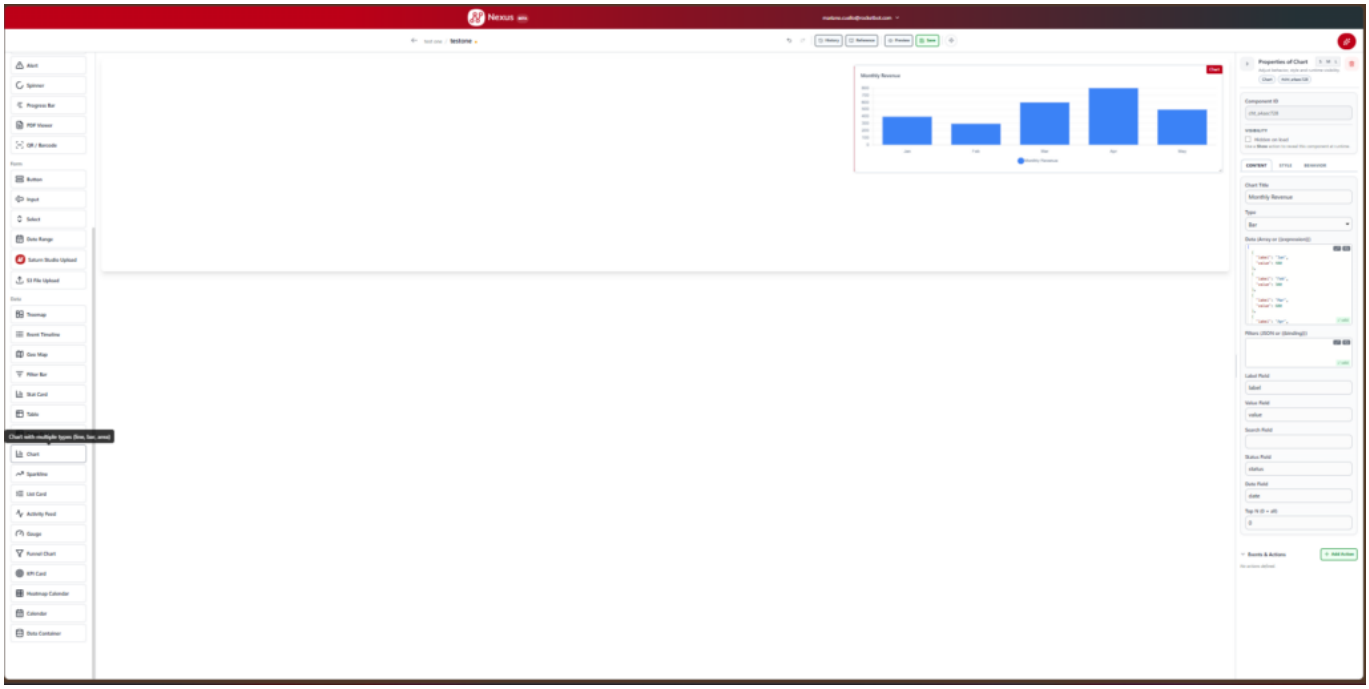
Propiedades

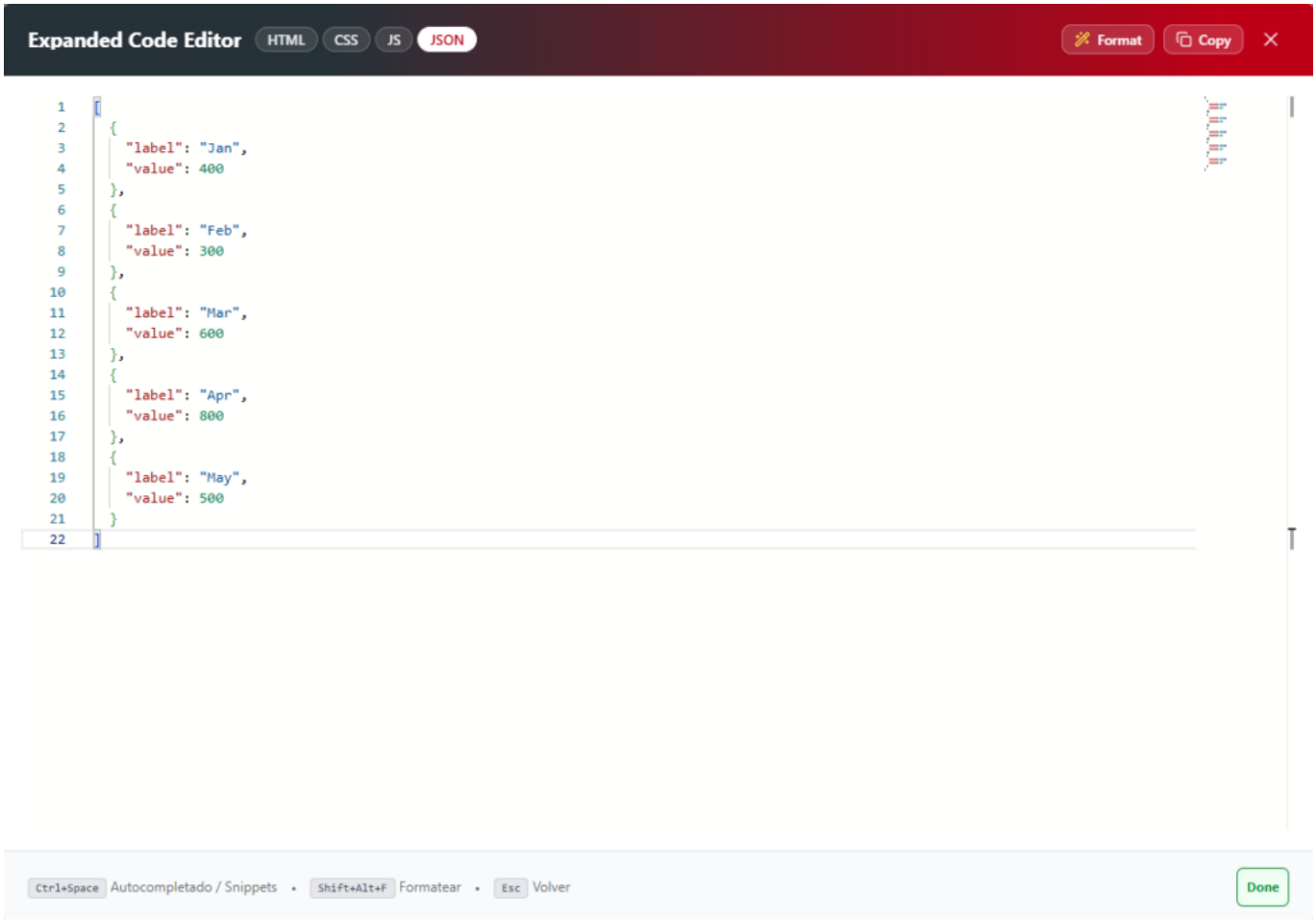
Al seleccionar Chart en el lienzo, el panel derecho muestra **“Properties of Chart”**, organizado en tres pestañas: **Content**, **Style** y **Behavior**.

- **Component ID**: identificador único del componente, autogenerado (ej. cht_a4aec728).
- **Visibility → Hidden on load**: casilla para que el componente esté oculto al cargar la pantalla. Puede mostrarse luego mediante una acción de tipo **Show** (ver Events & Actions).

Pestaña Content:

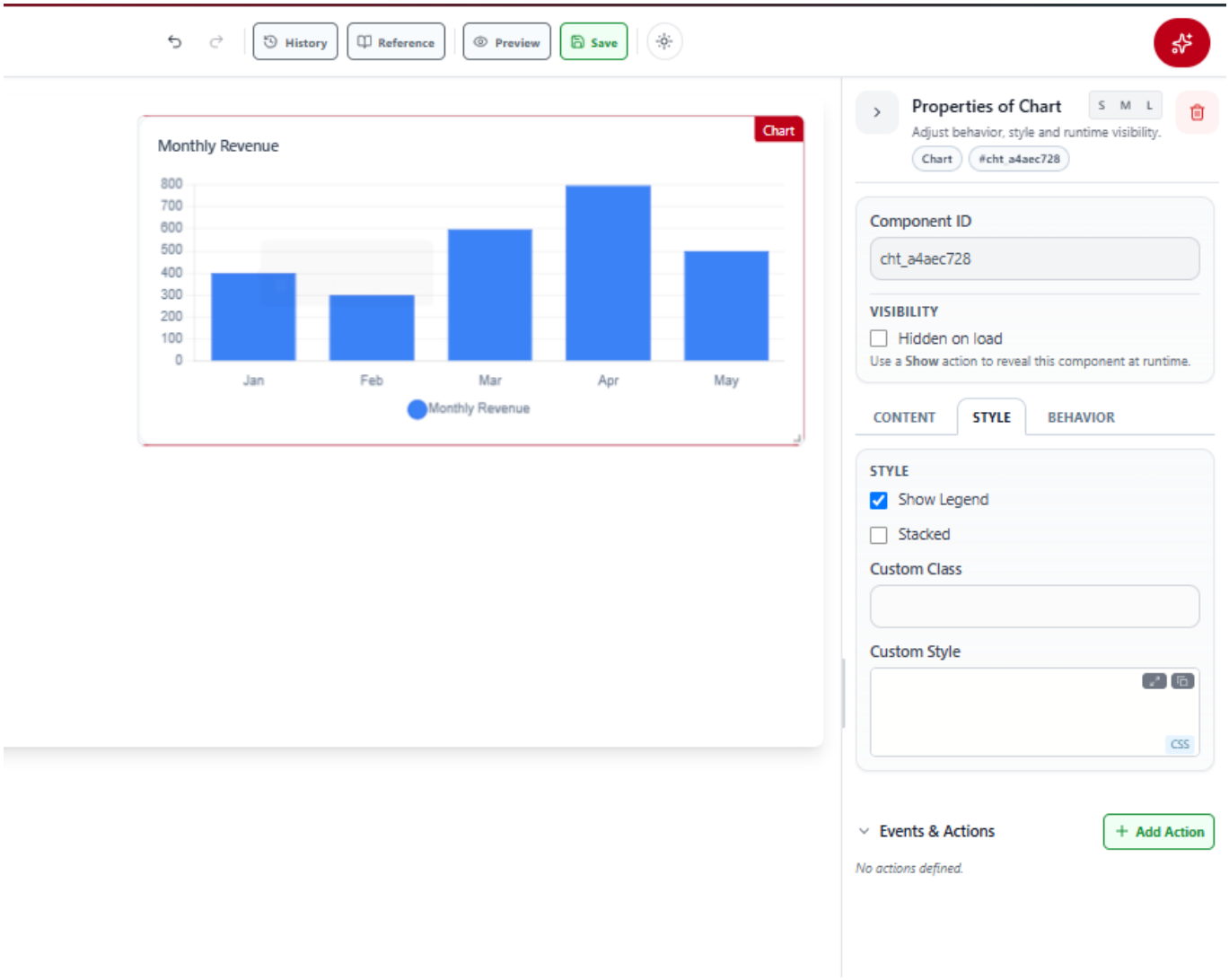
- **Chart Title**: título del gráfico (ej. “Monthly Revenue”).
- **Type**: menú desplegable que define el tipo de gráfico. Opciones: **Bar**, **Line**, **Pie**, **Doughnut**.
- **Data (Array or {{expression}})**: campo de código donde se cargan los datos del gráfico en formato JSON (array de objetos label / value), o alternativamente mediante una expresión/binding dinámico ({{...}}). Cuenta con el editor de código expandido.
- **Filters (JSON or {{binding}})**: campo para aplicar filtros a los datos mostrados, en JSON o mediante binding.
- **Label Field**: nombre del campo que se usa como etiqueta de cada dato (por defecto: label).
- **Value Field**: nombre del campo que se usa como valor de cada dato (por defecto: value).
- **Search Field**: nombre del campo usado para búsqueda/filtrado por texto (vacío por defecto).
- **Status Field**: nombre del campo usado para filtrar por estado (por defecto: status).
- **Date Field**: nombre del campo usado para filtrar por fecha (por defecto: date).
- **Top N (0 = all)**: limita el gráfico a mostrar solo los N valores más altos (0 = mostrar todos).





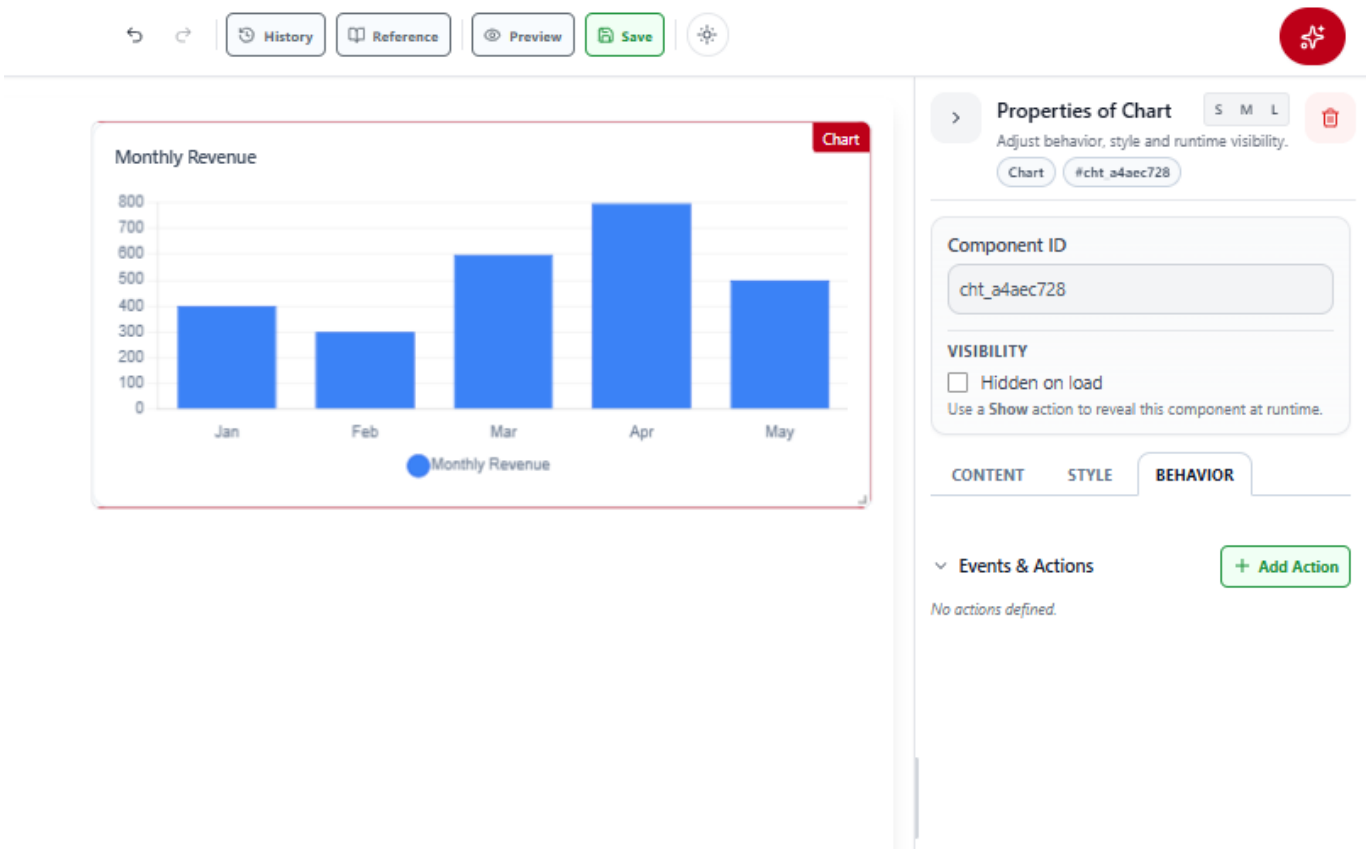
Pestaña Style:

- **Show Legend**: casilla que muestra u oculta la leyenda del gráfico.
- **Stacked**: casilla que apila las series de datos unas sobre otras (relevante en gráficos de barras/líneas con múltiples series).
- **Custom Class**: clase CSS personalizada.
- **Custom Style**: CSS personalizado aplicado directamente a este componente.



Pestaña Behavior:

- **Events & Actions**: sistema para definir comportamientos interactivos. Ver documentación aparte: [Nexus — Events & Actions]() .
- **Delete Component** (ícono de papelera, arriba): elimina el componente del lienzo.

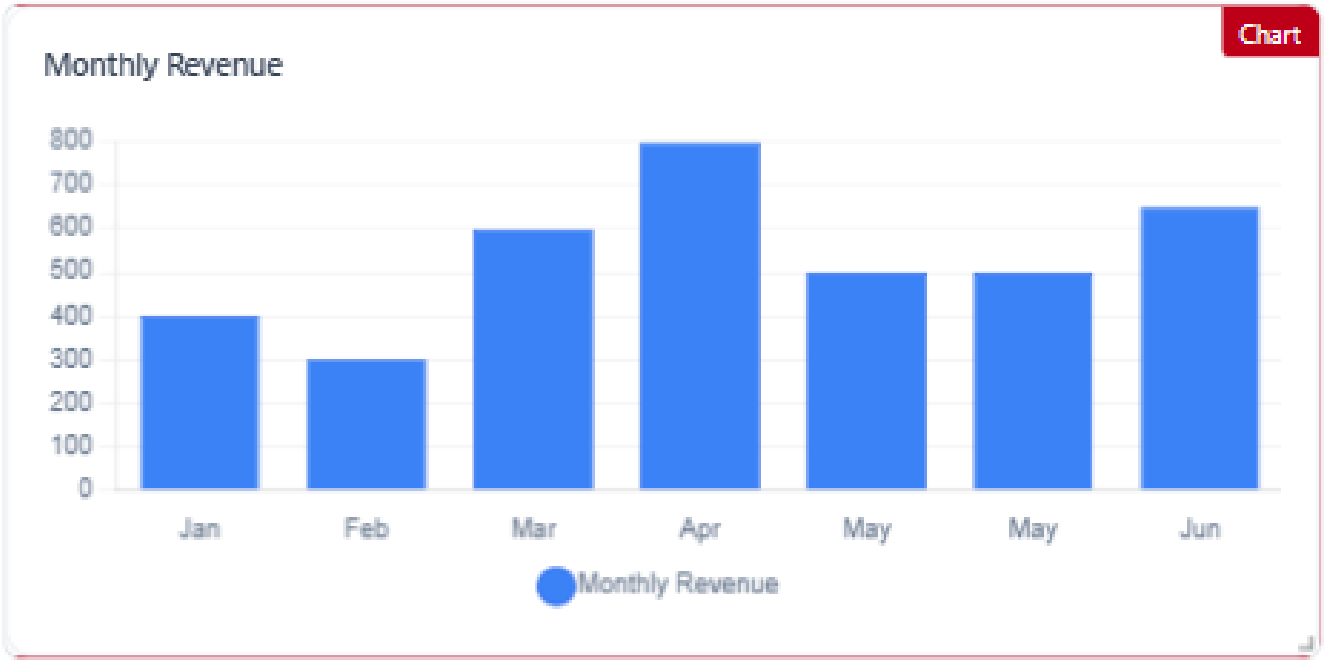


Ejemplo de uso

Para agregar un nuevo dato al gráfico, hay que ubicarse en la última llave } del bloque anterior dentro de **Data**, agregarle una coma, y pegar el nuevo bloque a continuación:

```
[
 {
    "label": "May",
    "value": 500
  },
  {
    "label": "Jun",
    "value": 650
  }
]
```

> ⚠ Importante: la coma va pegada a la llave } del bloque anterior, no al principio del bloque nuevo.



Con **Type** se elige la mejor forma de representar los datos: **Bar** o **Line** para tendencias en el tiempo, y **Pie** o **Doughnut** para mostrar proporciones dentro de un total. Los campos **Label Field** y **Value Field** permiten reutilizar el mismo componente con datos que usen claves distintas a label/value (por ejemplo, datos que vengan de una query con nombres de columna diferentes), sin tener que transformar el JSON. Con **Filters**, **Search Field**, **Status Field** y **Date Field**, el gráfico puede conectarse a un **Filter Bar** para actualizarse dinámicamente según lo que el usuario filtre.