Simple XML Reference

You can use Simple XML source code to build dashboards and forms. Use the dashboard source code editor to access and edit Simple XML. In some cases, Simple XML provides customization options not available in the dashboard user interface editor.

Overview

Elements

Simple XML source code has parent and child elements, starting with a <dashboard> or <form> root element. Use elements to structure a dashboard or form. For example, you can use a <row> element to organize one or more <panel> child elements in a dashboard. A <panel> can contain visualization elements such as a <table> or a <map>.

Attributes and options

Most Simple XML elements have attributes and options that you can use to configure dashboard appearance and behavior.

In visualization elements, you can use options to configure formatting, interactivity, and other components, such as legend placement or the colors to use in a chart.

Attributes and options have types, like number or text. Some attributes and options have default settings and additional requirements, such as an accepted value range.

Using the Simple XML Reference

Get reference information about Simple XML elements for building dashboards and forms.

<table>
<thead>
<tr>
<th>Build dashboards or forms. Configure structure and layout.</th>
<th>Add and configure visualizations</th>
<th>Work with searches to drive dashboard content</th>
<th>Add interactivity</th>
</tr>
</thead>
<tbody>
<tr>
<td>• dashboard or form&lt;br&gt;• row&lt;br&gt;• panel</td>
<td>• Working with visualization elements&lt;br&gt;• event&lt;br&gt;• table&lt;br&gt;• chart&lt;br&gt;• map&lt;br&gt;• single value&lt;br&gt;• html</td>
<td>• search</td>
<td>• fieldset (form) &lt;br&gt;• input (form) &lt;br&gt;• drilldown &lt;br&gt;• Predefined drilldown tokens &lt;br&gt;• eval, link, set, unset</td>
</tr>
</tbody>
</table>

dashboard or form

The <dashboard> or <form> root element appears at the top of dashboard Simple XML source code. A <form> root element indicates a dashboard that contains one or more inputs.

Dashboard element hierarchy

Dashboard definitions in Simple XML use the following parent and child element hierarchy.

<dashboard>
Form element hierarchy

Form definitions in Simple XML use the following parent and child element hierarchy. Unlike dashboards, forms have a <fieldset> element that contains <input> elements.

<form>
  <init> (0..1)
  <label> (0..1)
  <description> (0..1)
  <search> (0..1)
  <fieldset> (1)
    <input> (1..n)
    <row> (1..n)
    <panel> (0..n)
    <search> (0..n)
    <chart> | <event> | <html> | <map> | <single> | <table> | <viz> (1..n)
    <search> (0..n, for each visualization element)
</form>

Attributes

Use the following attributes to configure <dashboard> and <form> root elements.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hideChrome</td>
<td>boolean</td>
<td>false</td>
<td>Hide the Splunk bar, app bar, and footer.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For this and all of the following hide&lt;element_name&gt; attributes, if they are specified as a URL query string parameter without a value, they are handled as &quot;true&quot;. For example, both of the following query strings are handled as &quot;true&quot;. <code>&lt;dashboard_url&gt;?hideChrome</code> `&lt;dashboard_url&gt;?hideChrome=&quot;true&quot;</td>
</tr>
<tr>
<td>hideAppBar</td>
<td>boolean</td>
<td>false</td>
<td>Hide the bar listing available apps.</td>
</tr>
<tr>
<td>hideEdit</td>
<td>boolean</td>
<td>false</td>
<td>Hide the dashboard editing interface. If enabled, use the Settings &gt; User interface &gt; Views or the Dashboards page to edit the dashboard.</td>
</tr>
<tr>
<td>hideExport</td>
<td>boolean</td>
<td>false</td>
<td>Set to true to hide the dashboard Export menu.</td>
</tr>
<tr>
<td>hideFilters</td>
<td>boolean</td>
<td>false</td>
<td>Hide &lt;form&gt; inputs to increase panel display space. Not applicable to &lt;dashboard&gt;.</td>
</tr>
<tr>
<td>hideSplunkBar</td>
<td>boolean</td>
<td>false</td>
<td>Hide the bar that provides a link to the home page and settings pages.</td>
</tr>
<tr>
<td>hideTitle</td>
<td>boolean</td>
<td>false</td>
<td>Hide the text defined in the &lt;label&gt; and &lt;description&gt; elements.</td>
</tr>
<tr>
<td>isDashboard</td>
<td>boolean</td>
<td>true</td>
<td>For internal use.</td>
</tr>
<tr>
<td>isVisible</td>
<td>boolean</td>
<td>true</td>
<td>Indicates if the dashboard or form appears in the app Dashboards listing page and navigation menus.</td>
</tr>
<tr>
<td>onunloadCancelJobs</td>
<td>boolean</td>
<td></td>
<td>Indicate whether to cancel search jobs when a user navigates away from the dashboard or form.</td>
</tr>
<tr>
<td>refresh</td>
<td>integer</td>
<td>0</td>
<td>Specify a refresh interval, in seconds. The dashboard or form reloads after the interval.</td>
</tr>
</tbody>
</table>
### Name | Type | Default | Description
--- | --- | --- | ---
script | string |  | Comma-separated list of custom `.js` files to load. The files must be in a folder or subfolder of the `appserver/static` directory. `$SPLUNK_HOME/etc/apps/<app_name>/appserver/static/`  
To reference custom `.js` files from another app, specify the the app name when referencing the file. For example, use the following reference.  

```html
<dashboard script="myApp:myScript.js">
```

stylesheet | string |  | Comma-separated list of custom stylesheets to use for the dashboard. The stylesheet files must be in a folder or subfolder of the following directory.  
$SPLUNK_HOME/etc/apps/<app_name>/appserver/static/  
To reference a custom css file from another app, specify the the app name when referencing the file. For example, use the following reference.  

```html
<dashboard stylesheet="myApp:myStyles.css">
```

theme | string | light | Change the dashboard or form theme to `dark` or `light`. The dashboard or form will render in light mode if no theme is set, or if the theme is set to `light`.  
You must save the dashboard and refresh your browser to see theme changes.  
To change a dashboard's theme for viewing purposes only, without saving the change, you can specify the theme as a URL query string parameter. For example, the following parameter will render a dashboard in dark theme. `<dashboard_url>?theme=dark` Note that this overrides the saved state of the dashboard. You must delete the URL query string and refresh the page to view the actual saved dashboard theme.

---

**Attribute examples**

Use the `hideEdit` attribute to hide the editing interface from a dashboard.

```html
<dashboard hideEdit="true">
...
</dashboard>
```
Use the `refresh` attribute to reload a form after 30 seconds.

```html
<form refresh="30">
...
</form>
```

**Child elements**

A `<dashboard>` or `<form>` root element can contain one or more of the following child elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| `<description>` | Optional text description for a dashboard or a form.  
If a `<label>` is also used, the description appears below the label in a smaller font. | `<dashboard>`

```html
<description>
The following panels show monthly
```

```html
<dashboard>

```html
<label>Sales this month</label>
```
<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| <fieldset> | Used within a `<form>` to organize form inputs. Not applicable to `<dashboard>`. `<fieldset>` has two attributes:  
  - `<autoRun>`, which defaults to false.  
  - `<submitButton>`, which defaults to true.  
  If `<submitButton>` is set to false, then two things happen:  
    - `<autoRun>` is ignored, and behaves as if it was set to true.  
    - The attribute `<searchWhenChanged>` for the element `<input>` is ignored and behaves as if it was set to true. | `<form>`  
  `<label>Events by sourcetype</label>`  
  `<fieldset>`  
  `<input type="text" token="series">`  
  `<label>Enter a source type</label>`  
  `<default></default>`  
  `<initialValue>splunkd</initialValue>`  
  `</input>`  
  `</fieldset>`  
  `</form>` |
| <init> | Element containing token updates to make on page load. For example, set a token value to show specific search results when a dashboard opens. See Token usage in dashboards for more information on setting tokens on page load. | `<dashboard>`  
  `<init>`  
  `<set token="selected_sourcetype">mongod</set>`  
  `</init>`  
  `</dashboard>` |
| <label> | Optional text label for a dashboard, form, or form input. | `<dashboard>`  
  `<label>Recent login activity</label>`  
  `</dashboard>` |
| <row> | Organizes `<panel>` elements in a dashboard or form. See row for more details. | `<dashboard>`  
  `<label>Events by sourcetype</label>`  
  `...`  
  `</dashboard>` |
| <search> | Global search element in a dashboard or form. A dashboard or form can contain one or more global `<search>` elements. If a dashboard or form contains a global search, use a post-process search within a `<panel>` element to show results in the panel. See search for more details. | |
fieldset (form)

A <fieldset> contains <form> inputs, such as a checkbox or radio buttons.

Element hierarchy

<form>
    <fieldset autoRun="[Boolean]" submitButton="[Boolean]">
        <html> (0..n)
            <input type="[input type]" token="[search token]"> (1..n)
            <default> (0..1)
            <fieldForLabel> (0..1)
            <fieldForValue> (0..1)
            <initialValue> (0..1)
            <label> (0..1)
            <prefix> (0..1)
            <search> (0..1)
            <selectFirstChoice> (0..1)
            <suffix> (0..1)
    </fieldset>
</form>

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>autoRun</td>
<td>Boolean</td>
<td>False</td>
<td>Indicates whether to run the search when the page loads.</td>
</tr>
<tr>
<td>submitButton</td>
<td>Boolean</td>
<td>True</td>
<td>Indicates whether to display a Submit button.</td>
</tr>
</tbody>
</table>

If submitButton is set to false, then two things happen:

- autoRun is ignored, and behaves as it was set to true.
- The attribute searchWhenChanged for the element <input> is ignored and behaves as if it was set to true.

Child elements

A <fieldset> contains one or more <input> elements.

Example

```html
<fieldset autoRun="true" submitButton="false">
    <input type="text" token="series">
        <label>sourcetype</label>
        <default></default>
        <initialValue>splunkd</initialValue>
        <suffix>*</suffix>
    </input>
</fieldset>
```

input (form)

Input elements define the user interaction interface for a form. Selection elements, such as radio buttons or a checkbox, let users click or select choices. You can use a search to populate selection elements dynamically. Other inputs, such as time or text, let users type content or change a search time range.

Use attributes to configure user inputs and use child elements to define input content. Some child elements are shared across several input types. Others are specific to the input type.
Input attributes

Use these attributes in any form input.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>depends</td>
<td>Comma-separated list</td>
<td>All tokens in the list must be defined for the input to render. These tokens might be used in an input or drilldown context in the form.</td>
</tr>
<tr>
<td>id</td>
<td>Text (minimum two characters)</td>
<td>Input identifier. Only alphanumeric and underscore characters are valid. The id cannot begin with a number or the underscore character.</td>
</tr>
<tr>
<td>rejects</td>
<td>Comma-separated list</td>
<td>Prevent input rendering if one or more tokens in this list are defined. These tokens might be used in an input or drilldown context in the form.</td>
</tr>
<tr>
<td>searchWhenChanged</td>
<td>Boolean</td>
<td>Specifies to run the search when the selection changes. Defaults to false.</td>
</tr>
<tr>
<td>token</td>
<td>Token name</td>
<td>Token representing the user selected value from this input.</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>One of the following input types.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- checkbox</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- dropdown</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- link</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- multiselect</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- radio</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- text</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- time</td>
</tr>
</tbody>
</table>

Shared input child elements

Use these child elements in `<input>` elements. There are a few usage exceptions.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
<th>Usage exceptions</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;change&gt;</code></td>
<td>See change</td>
<td>Not available for multiselect inputs.</td>
</tr>
<tr>
<td><code>&lt;condition&gt;</code></td>
<td>See condition (input)</td>
<td>Not available for multiselect inputs.</td>
</tr>
<tr>
<td><code>&lt;default&gt;</code></td>
<td>Default value for the input. If the user does not select or input a value, this value is used. Overrides the initialValue.</td>
<td></td>
</tr>
<tr>
<td><code>&lt;earliest&gt;</code></td>
<td>The earliest and latest times defining the <code>&lt;search&gt;</code> time range. Use relative time modifiers, as described in Specify time modifiers in your search, or use UNIX epoch time format for absolute time.</td>
<td>Not available for text or time inputs.</td>
</tr>
<tr>
<td><code>&lt;latest&gt;</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>&lt;fieldForLabel&gt;</code> and <code>&lt;fieldForValue&gt;</code></td>
<td>In dynamically populated selection inputs, these elements indicate the search result fields to use for the label and value of each choice. Use with the <code>&lt;search&gt;</code> element that generates results for populating the input choices.</td>
<td>Not available for text or time inputs.</td>
</tr>
<tr>
<td><code>&lt;initialValue&gt;</code></td>
<td>Initial value for the input. This value is used before the user selects or inputs a value.</td>
<td></td>
</tr>
<tr>
<td><code>&lt;label&gt;</code></td>
<td>Label for the input.</td>
<td></td>
</tr>
<tr>
<td><code>&lt;prefix&gt;</code></td>
<td>Prefix to prepend to the selected value. For example, you can add a quotation mark to the beginning of a string value and use <code>&lt;suffix&gt;</code> to add a closing quotation mark after the string value. The prefix can be a string or regular expression.</td>
<td></td>
</tr>
<tr>
<td><code>&lt;search&gt;</code></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Name</td>
<td>Description</td>
<td>Usage exceptions</td>
</tr>
<tr>
<td>------------</td>
<td>------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>Search that dynamically populates choices for the input. Use the ref attribute of the &lt;search&gt; element to reference a search from a report. See &lt;search&gt;.</td>
<td>Not available in text inputs.</td>
</tr>
<tr>
<td>&lt;suffix&gt;</td>
<td>Suffix to append to the selected value. The suffix can be a string or regular expression.</td>
<td></td>
</tr>
</tbody>
</table>

**checkbox child elements**

In addition to shared input child elements, you can use the following child elements in a `<checkbox>`.

<table>
<thead>
<tr>
<th>element</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;delimiter&gt;</td>
<td>text</td>
<td></td>
<td>A string that will be placed between each selected value. Typically, you specify &quot; OR &quot; or &quot; AND &quot; using upper case – do not specify the quote marks, but specify a space character before and after the text.</td>
</tr>
<tr>
<td>&lt;valuePrefix&gt;</td>
<td>text</td>
<td></td>
<td>String prefixed to the value of the input element. Can be a regular expression.</td>
</tr>
<tr>
<td>&lt;valueSuffix&gt;</td>
<td>text</td>
<td></td>
<td>String appended to the value of the input element. Can be a regular expression.</td>
</tr>
</tbody>
</table>

**dropdown child elements**

In addition to shared input child elements, you can use the following child elements in a `<dropdown>`.

<table>
<thead>
<tr>
<th>element</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;allowCustomValues&gt;</td>
<td>boolean</td>
<td>false</td>
<td>If true, enables the selection of custom values typed into the text field for the input.</td>
</tr>
<tr>
<td>&lt;choice value='value'&gt;</td>
<td>text</td>
<td></td>
<td>Required. Specifies the value to use for the choice.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Specifies choices for a radio or dropdown element. &lt;choice&gt; Is the label to use for the specified value.</td>
</tr>
<tr>
<td>&lt;selectFirstChoice&gt;</td>
<td>boolean</td>
<td>false</td>
<td>Indicates if the first item listed is the default item for the input. If a value for &lt;default&gt; is present, &lt;selectFirstChoice&gt; is ignored.</td>
</tr>
<tr>
<td>&lt;showClearButton&gt;</td>
<td>boolean</td>
<td>true</td>
<td>Indicates if the clear button for the dropdown is present. When present, the user clicks the clear button to change the choice to the default value for the dropdown.</td>
</tr>
</tbody>
</table>

**link child elements**

In addition to shared input child elements, you can use the following child elements in a `<link>`.

<table>
<thead>
<tr>
<th>element</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;choice value='value'&gt;</td>
<td>text</td>
<td></td>
<td>Required. Specifies the value to use for the choice.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Specifies choices for the link input element. &lt;choice&gt; Is the label to use for the specified value.</td>
</tr>
<tr>
<td>&lt;selectFirstChoice&gt;</td>
<td>boolean</td>
<td>false</td>
<td>Indicates if the first item listed is the default item for the input. Overrides any value for &lt;initialValue&gt;. If a value for &lt;default&gt; is present, &lt;selectFirstChoice&gt; is ignored.</td>
</tr>
</tbody>
</table>
multiselect child elements

In addition to shared input child elements, you can use the following child elements in a `<multiselect>`.

<table>
<thead>
<tr>
<th>element</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;allowCustomValues&gt;</code></td>
<td>boolean</td>
<td>false</td>
<td>If true, enables the selection of custom values typed into the text field for the input.</td>
</tr>
<tr>
<td><code>&lt;delimiter&gt;</code></td>
<td>text</td>
<td></td>
<td>A string that will be placed between each selected value. Typically, you specify &quot; OR &quot; or &quot; AND &quot; using upper case – do not specify the quote marks, but specify a space character before and after the text.</td>
</tr>
<tr>
<td><code>&lt;valuePrefix&gt;</code></td>
<td>text</td>
<td></td>
<td>String prefixed to the value of the input element. Can be a regular expression.</td>
</tr>
<tr>
<td><code>&lt;valueSuffix&gt;</code></td>
<td>text</td>
<td></td>
<td>String appended to the value of the input element. Can be a regular expression.</td>
</tr>
</tbody>
</table>

radio child elements

In addition to shared input child elements, you can use the following child elements in a `<radio>`.

<table>
<thead>
<tr>
<th>element</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;choice value=[value]&gt;</code></td>
<td>text</td>
<td></td>
<td>Required. Represents a radio button choice value.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Specifies choices for a radio or dropdown element. <code>&lt;choice&gt;</code> is the label for the specified value.</td>
</tr>
<tr>
<td><code>&lt;selectFirstChoice&gt;</code></td>
<td>boolean</td>
<td>false</td>
<td>Indicates if the first item listed is the default selected item for the input. If a value for <code>&lt;default&gt;</code> is present, <code>&lt;selectFirstChoice&gt;</code> is ignored.</td>
</tr>
</tbody>
</table>

text child elements

See shared input child elements for available elements and exceptions.

time child elements

See shared input child elements for available elements and exceptions.

change (form input)

Use a `<change>` element to define how a form responds to user input. When a user makes a selection or types in an input, the actions configured in the `<change>` element for that input happen.

The `<change>` element is available in the following `<input>` types.

- `<checkbox>`
- `<dropdown>`
- `<radio>`
- `<text>`
- `<time>`

Use tokens to define responses to user input

Inside a `<change>` element, you can define actions such as token setting or other token updates that help you manage
dynamic behavior or display in the form.

**Conditional responses**

Use one or more `<condition>` elements inside a `<change>` element to define conditional responses to specific user input. See `<condition>` for more information.

**Element hierarchy**

```
<input>
  <change>
    ( <set> | <unset> | <link> | <eval> ) (1..n)
    <condition> (0..n)
    ( <set> | <unset> | <link> | <eval> ) (1..n)
```

**Attributes**

None

**Child elements**

You can use the following child elements in a `<change>` element.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;set&gt;</code></td>
<td>Set or update a token value.</td>
</tr>
<tr>
<td><code>&lt;unset&gt;</code></td>
<td>Unset a previously set token value.</td>
</tr>
<tr>
<td><code>&lt;link&gt;</code></td>
<td>Link to a target destination, such as another dashboard, form, or an external website.</td>
</tr>
<tr>
<td><code>&lt;eval&gt;</code></td>
<td>Filter or format a token value.</td>
</tr>
</tbody>
</table>

**Example**

Use the `<change>` element to capture the selected label and value from an input.

```
<form>
  <label>Use tokens with input choices to capture input labels and values</label>
  <init>
    <set token="date_label">Last 7 Days</set>
    <set token="earliest_tok">-7d@d</set>
  </init>
  <fieldset submitButton="false">
    <input type="radio" token="period_tok">
    <label>Select a time range</label>
    <choice value="-24h@h">Last 24 Hours</choice>
    <choice value="-7d@d">Last 7 Days</choice>
    <choice value="-30d@d">Last 30 Days</choice>
    <default>Last 7 Days</default>
  </fieldset>
  <change>
    <!-- use predefined input tokens to set -->
    <!-- tokens for the selected label and value -->
    <set token="date_label">$label$</set>
    <set token="earliest_tok">$value$</set>
  </change>
</form>
```
Conditional Inputs

Use tokens with input choices to capture input labels and values

Select a time range
- Last 24 Hours
- Last 7 Days
- Last 30 Days

Conditional Inputs

Source Type by Last 7 Days

Events

Time period

10
condition (form input)

Use a `<condition>` inside a form input `<change>` element to define a conditional response to user input.

The `<condition>` element is not available for multiselect inputs.

For information on using a `<condition>` element as part of a drilldown, see `<condition>` (drilldown).

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>text</td>
<td>*</td>
<td>Specifies the input label element to which the condition applies. Applies the condition to all input <code>&lt;label&gt;</code> elements.</td>
</tr>
<tr>
<td>match=</td>
<td>text</td>
<td></td>
<td>Specifies a condition to evaluate for a match. For example, you can use `&lt;condition match=&quot;'job.resultCount' == 0&quot;&gt; to specify a condition to apply when a search returns no results.</td>
</tr>
<tr>
<td>value</td>
<td>text</td>
<td>*</td>
<td>Specifies the input value element to which the condition applies. Applies the condition to all input <code>&lt;value&gt;</code> elements.</td>
</tr>
</tbody>
</table>

Child elements

Use the following child elements in a `<condition>` to define conditional token setting or other behavior. See eval, link, set, and unset for more details.

<table>
<thead>
<tr>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;set&gt;</td>
<td>Set a token value.</td>
</tr>
<tr>
<td>&lt;unset&gt;</td>
<td>Unset a previously set token value.</td>
</tr>
<tr>
<td>&lt;link&gt;</td>
<td>Link to a target destination, such as another dashboard, form, or an external website.</td>
</tr>
<tr>
<td>&lt;eval&gt;</td>
<td>Process or format a token value.</td>
</tr>
</tbody>
</table>

Element hierarchy

```
<change>
  <condition>
    { <set> | <unset> | <link> | <eval> } (1..n)
```

Example

Use conditional inputs to select preset time ranges for a search.

The token for the selected choice appears in the title for the chart. The conditional token for the selected value drives the data for the chart.
<label>Use tokens with conditional input choices</label>
<fieldset submitButton="false">
<input type="radio" token="period_tok">
<label>Select a time range</label>
<choice value="-24h@h">Last 24 Hours</choice>
<choice value="-7d@h">Last 7 Days</choice>
<choice value="-30d@h">Last 30 Days</choice>
<default>Last 7 Days</default>

<!-- set condition based on the label defined by <choice> -->
<!-- Within each condition, specify a custom label for display -->
<!-- Capture the selected value in the token, earliest_tok -->
<change>
<condition label="Last 24 Hours">
<set token="date_label">Yesterday</set>
<set token="earliest_tok">$value$</set>
</condition>
<condition label="Last 7 Days">
<set token="date_label">Last week</set>
<set token="earliest_tok">$value$</set>
</condition>
<condition label="Last 30 Days">
<set token="date_label">Last month</set>
<set token="earliest_tok">$value$</set>
</condition>
</change>
</input>
</fieldset>
</form>

<!-- Display selected label in the title -->
title>$date_label$</title>

<search>
<query>index = _internal | timechart count by sourcetype</query>
<!-- use the value of earliest_tok -->
<!-- to set the time range -->
<earliest>$earliest_tok$</earliest>
<latest>now/latest</latest>
</search>

<option name="charting.axisY.scale">log</option>
<option name="charting.axisTitleX.text">Time periods</option>
<option name="charting.axisTitleY.text">Events</option>
</chart>
</row>
</form>
row

A `row` contains one or more `<panel>` child elements in a dashboard or form. Use a `row` to configure panel grouping and layout.

Attributes

Use the following attributes within the `<row>` element.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>depends</td>
<td>Comma-separated list of tokens</td>
<td>All tokens from the list of tokens must be defined to render this row in a dashboard. However, undefined tokens do not prevent the searches in the row from running in the background.</td>
<td></td>
</tr>
<tr>
<td>rejects</td>
<td>Comma-separated list of tokens</td>
<td>All tokens from this list must be defined to prevent this row from rendering in a dashboard.</td>
<td></td>
</tr>
<tr>
<td>id</td>
<td>text</td>
<td>Row identifier. Only alphanumeric and underscore characters are valid. The <code>id</code> cannot begin with a number or the underscore character. The following terms are reserved for internal use and cannot be used for an <code>id</code>.</td>
<td></td>
</tr>
</tbody>
</table>

- `dashboard`
- `search`
- `default`
- `submitted`
- `footer`
- `url`
- `header`

Child element

A `<row>` normally contains one or more `<panel>` elements. It's also possible to have `<row>` that contains an `<html>` element. However a single `<row>` cannot contain both a `<panel>` and an `<html>`.
**Panel**

Inside a `<row>`, use a `<panel>` to contain one or more visualizations. You can use panels to manage visualization grouping and layout. You can also show or hide panels dynamically.

**Panel alignment**

Two or more visualization elements in a panel align vertically, except for single value visualizations. Two or more single value visualizations in the same panel are grouped horizontally.

**Panel types**

There are two available `<panel>` types in Simple XML.

<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
<th>Requirements</th>
<th>Editing options</th>
<th>Limitations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inline</td>
<td>Contains one or more visualization elements.</td>
<td>N/A</td>
<td>Create and update an inline panel using the dashboard editing user interface or Simple XML source code.</td>
<td>N/A</td>
</tr>
<tr>
<td>Reference</td>
<td>Shows the contents of a prebuilt panel</td>
<td>Requires a <code>ref</code> attribute to indicate the prebuilt panel to display. Optionally, add an <code>app</code> reference.</td>
<td>Reference panels cannot be edited using the panel editor. Reference panels do not recognize <code>&lt;panel&gt;</code> child elements specified in Simple XML.</td>
<td></td>
</tr>
</tbody>
</table>

For more information on prebuilt panels, see Dashboard panels and Create and add a panel by reference.

**Element hierarchy**

**Inline panel**

```xml
<row>
    <title> (0..1)
    <search> (0..n)
    <chart> | <event> | <html> | <map> | <single> | <table> | <viz> (1..n)
</row>
```

**Reference panel**

```xml
<panel ref="[panel name]" [app="[app name]"]> (0..n)
    <!-- Other <panel> child elements ignored -->
</panel>
```

**Attributes**

Use the following attributes within the `<panel>` element.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ref</td>
<td>Text</td>
<td></td>
<td>Required for reference panels only. References the name of a prebuilt panel. Prebuilt panel names appear in Settings &gt; User Interface &gt; Panels.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>---------------------------</td>
<td>-------------------------------------------------------------------------</td>
<td>----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>app</td>
<td>Text</td>
<td>Defaults to the app containing the dashboard.</td>
<td>Optional for reference panels. References the name of the app containing the prebuilt panel. Panel app names appear in Settings &gt; User Interface &gt; Panels.</td>
</tr>
<tr>
<td>depends</td>
<td>Comma-separated list of tokens</td>
<td></td>
<td>All tokens in this list must be defined to render this panel in a dashboard.</td>
</tr>
<tr>
<td>id</td>
<td>text</td>
<td></td>
<td>Panel identifier. Only alphanumeric and underscore characters are valid. The id cannot begin with a number or the underscore character. The following terms are reserved for internal use and cannot be used for an id. <em>dashboard</em> <em>search</em> <em>default</em> <em>submitted</em> <em>footer</em> <em>url</em> <em>header</em></td>
</tr>
<tr>
<td>rejects</td>
<td>Comma-separated list of tokens</td>
<td></td>
<td>All tokens from this list must be defined to prevent this panel from rendering in a dashboard.</td>
</tr>
</tbody>
</table>

**Child elements**

Use `<panel>` child elements to configure inline panels. Reference panels ignore `<panel>` child elements.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| `<title>`        | Optional `<title>` panel title. A panel title can provide context for a group of visualizations. Each visualization element in the panel can also have its own title. | `<form> ...
<row>
<title>Video game sales</title>
<table>
</table>
</panel>
</row>
</form>` |
| `<search>`       | Base search for a panel. Use a base panel `<search>` element with a post-process search in a panel visualization. | `<form> ...
<row>
<panel>
<title>Video game sales</title>
<table>
</table>
</panel>
</row>
</form>` |
| visualization element | A panel can contain one or more of the following visualization types. | `<chart> | <event> | <html> | <map> | <single> | <table> | <viz>` |
Panel example

This `panel` element shows single value visualizations and chart groupings.

```xml
<dashboard>
  <label>Dashboard Panel Example</label>
  <description></description>
  <row>
    <panel>
      <chart>
        <title>Chart grouping</title>
        <search>
          <query>
            index=_internal source="*splunkd.log"
            ( log_level=ERROR OR log_level=WARN* OR log_level=FATAL OR log_level=Critical )
            | stats count as log_events
            | rangemap field=log_events low=1-100 elevated=101-300 default=severe
          </query>
          <earliest>-7d@h</earliest>
          <latest>now</latest>
        </search>
        <option name="charting.chart">radialGauge</option>
      </chart>
      <chart>
        <search>
          <query>
            index=_internal source="*splunkd.log"
            ( log_level=ERROR OR log_level=WARN* OR log_level=FATAL OR log_level=Critical )
            | stats count as log_events
            | rangemap field=log_events low=1-100 elevated=101-300 default=severe
          </query>
          <earliest>-7d@h</earliest>
          <latest>now</latest>
        </search>
        <option name="charting.chart">markerGauge</option>
      </chart>
    </panel>
  </row>
  <row>
    <panel>
      <single>
        <title>Single value grouping</title>
        <search>
          <query>
            index=_internal source="*splunkd.log"
            ( log_level=ERROR OR log_level=WARN* OR log_level=FATAL OR log_level=Critical )
            | stats count as log_events
            | rangemap field=log_events low=1-100 elevated=101-300 default=severe
          </query>
          <earliest>-7d@h</earliest>
          <latest>now</latest>
        </search>
        <option name="beforeLabel">Found</option>
        <option name="afterLabel">errors</option>
      </single>
    </panel>
  </row>
</dashboard>
```
Working with visualization elements

Use one or more of the following elements to add a visualization to a <panel>.

<table>
<thead>
<tr>
<th>Element</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;event&gt;</td>
<td>Event list</td>
</tr>
<tr>
<td>&lt;table&gt;</td>
<td>Statistics table</td>
</tr>
<tr>
<td>&lt;chart&gt;</td>
<td>Line, bar, area, bubble, and scatter charts</td>
</tr>
<tr>
<td>&lt;map&gt;</td>
<td>Cluster maps and Choropleth maps</td>
</tr>
<tr>
<td>&lt;single&gt;</td>
<td>Single value visualization</td>
</tr>
<tr>
<td>&lt;html&gt;</td>
<td>Custom html content</td>
</tr>
<tr>
<td>&lt;viz&gt;</td>
<td>Custom visualization. See Custom visualizations in Simple XML in Developing Views and Apps for Splunk Web.</td>
</tr>
</tbody>
</table>

Using visualization attributes and options

Like other Simple XML elements, visualizations use attributes for identification and for managing display. There are several shared attributes that you can use in any visualization type. Some visualizations have additional attributes available.

Each visualization has several available <option> named properties for configuration. Shared options can be used in all visualizations. Other <option> properties are specific to one visualization type.

Option syntax

Use the following option tagging syntax within the visualization element.

```xml
<option name="[option_name]">[option_value]</option>
```

For example, use the following option to show the Export button in a panel.

```xml
<option name="link.exportResults.visible">true</option>
```

Shared attributes

Use the following attributes to identify and control display for any visualization element.
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>depends</td>
<td>Comma-separated list of tokens that must be set for the panel, row, or visualization to display.</td>
<td>All tokens from the list of tokens must be defined to render this visualization. For example, <code>&lt;chart depends=&quot;$show1$&quot;&gt;</code> indicates that the chart is shown only when the <code>$show1$</code> token is set. Hiding an element does not stop any search elements from running in the background.</td>
</tr>
<tr>
<td>rejects</td>
<td>Comma-separated list of tokens that must be set for this visualization to be hidden.</td>
<td>All tokens from this list must be defined to prevent this visualization from rendering. For example, <code>&lt;table rejects=&quot;$one$, $two$&quot;&gt;</code> indicates that the table is hidden when either the <code>$one$</code> or <code>$two$</code> token is set.</td>
</tr>
</tbody>
</table>
| id | text | Visualization identifier. Only alphanumeric and underscore characters are valid. The id cannot begin with a number or the underscore character. The following terms are reserved for internal use and cannot be used for an id.  
  • dashboard  
  • search  
  • default  
  • submitted  
  • footer  
  • url  
  • header |

**Shared options**

**Trellis layout options**

Trellis layout is available for any visualization type except cluster maps and tables. For more information, see Use trellis layout to split visualizations.

<table>
<thead>
<tr>
<th>Option name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trellis.enabled</td>
<td>Boolean</td>
<td>0</td>
<td>Enable or disable trellis layout. Defaults to 0 (disabled).</td>
</tr>
</tbody>
</table>
| trellis.scales.shared | Boolean | 1 | Indicate whether to share scale for axes in line, area, column, and bar charts or value ranges in Choropleth map segments. Use one of the following values.  
  • 1: Shared scales  
  • 0: Independent scales |
| trellis.size | String | medium | Configure the visualization segment size. Segment size affects panel display density for the split visualization. Use one of the following values.  
  • small  
  • medium  
  • large |
| trellis.splitBy | Result field name | N/A | Indicate the search result field or aggregation name to use for splitting the visualization. Segments appear for each value in this field. |
Search, Inspect, Refresh, and Export options

Use the following `<option>` settings to configure Open in Search behavior and manage other panel user interface elements.

<table>
<thead>
<tr>
<th>Option name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>link.exportResults.visible</td>
<td>Boolean</td>
<td>Defaults to the value of link.visible</td>
<td>Show the Export button at the bottom of the panel.</td>
</tr>
<tr>
<td>link.inspectSearch.visible</td>
<td>Boolean</td>
<td>Defaults to the value of link.visible</td>
<td>Show the Inspect button at the bottom of the panel.</td>
</tr>
<tr>
<td>link.openPivot.visible</td>
<td>Boolean</td>
<td>Defaults to the value of link.visible</td>
<td>Show the Open in Pivot button at the bottom of the panel.</td>
</tr>
<tr>
<td>link.openSearch.search</td>
<td>search</td>
<td>Defaults to the panel search</td>
<td>Specify a search to open when a user clicks the Open in Search button.</td>
</tr>
<tr>
<td>link.openSearch.searchEarliestTime</td>
<td>time modifier</td>
<td>Defaults to the panel search earliest time.</td>
<td>The earliest time to use for the alternative search specified by link.openSearch.search.</td>
</tr>
<tr>
<td>link.openSearch.searchLatestTime</td>
<td>time modifier</td>
<td>Defaults to the panel search latest time.</td>
<td>The latest time to use for the alternative search specified by link.openSearch.search.</td>
</tr>
<tr>
<td>link.openSearch.text</td>
<td>text</td>
<td>Open in Search</td>
<td>Specify a custom label for the Open in Search button.</td>
</tr>
<tr>
<td>link.openSearch.viewTarget</td>
<td>View name</td>
<td>search</td>
<td>Specify a target view to use when a user clicks the Open in Search button.</td>
</tr>
<tr>
<td>link.openSearch.visible</td>
<td>Boolean</td>
<td>Defaults to the value of link.visible</td>
<td>Show the Open in Search button at the bottom of the panel.</td>
</tr>
<tr>
<td>link.visible</td>
<td>Boolean</td>
<td>true</td>
<td>Show link buttons at the bottom of the panel.</td>
</tr>
<tr>
<td>refresh.time.visible</td>
<td>Boolean</td>
<td>true</td>
<td>Display the refresh time indicator in the panel.</td>
</tr>
<tr>
<td>refresh.link.visible</td>
<td>Boolean</td>
<td>true</td>
<td>Display the refresh link in the panel.</td>
</tr>
</tbody>
</table>

**event**

Use the `event` element to add an events list to a `<panel>`.
Attributes

You can use any of the shared attributes for visualizations in an `<event>` element.

Child elements

You can use the following child element in an `<event>` list.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| `<fields>` | Comma separated list | Restrict search results to these fields. The order of the listed fields determines the order of fields in the event list. | `<dashboard>`
  `<label>Fields Example</label>`
  `<row>`
  `<panel>`
  `<event>`
    `<search>`
      `<query>`
        `index=_internal | timechart count by sourcetype`<br>`<earliest>-7d@d</earliest>`<br>`<latest>now</latest>`
      `</query>`
    `</search>`
    `<fields>_time, splunkd, splunk_web_access, splunk_web_service</fields>`
  `</event>`
  `</panel>`
  `</row>`
`</dashboard>` |

Options

In addition to the shared options for visualizations, you can use the following options to configure an event list.

<table>
<thead>
<tr>
<th>Option name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Integer</td>
<td></td>
<td>The maximum number of rows to display.</td>
</tr>
<tr>
<td>Option name</td>
<td>Type</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>----------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>list.drilldown</td>
<td>One of the following values.</td>
<td>full</td>
<td>Specify drilldown enabling for the events list.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>full</strong>: Enables the entire entry for drilldown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>inner</strong>: Enables inner elements of the event listing for drilldown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>outer</strong>: Enables outer elements of the event listing for drilldown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>none</strong>: Disables drilldown.</td>
</tr>
<tr>
<td>list.wrap</td>
<td>Boolean</td>
<td>true</td>
<td>Indicates whether to wrap the contents of the events list.</td>
</tr>
<tr>
<td>maxLines</td>
<td>Integer</td>
<td></td>
<td>The maximum number of lines to display for each result/event.</td>
</tr>
<tr>
<td>raw.drilldown</td>
<td>One of the following values.</td>
<td>full</td>
<td>Specify drilldown enabling in the raw event listing.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>full</strong>: Enables the entire entry for drilldown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>inner</strong>: Enables inner elements of the event listing for drilldown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>outer</strong>: Enables outer elements of the event listing for drilldown.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>none</strong>: Disables drilldown.</td>
</tr>
<tr>
<td>rowNumbers</td>
<td>Boolean</td>
<td>false</td>
<td>Indicates whether to display row numbers.</td>
</tr>
<tr>
<td>showPager</td>
<td>Boolean</td>
<td>true</td>
<td>Toggle pagination on or off.</td>
</tr>
<tr>
<td>table.sortColumn</td>
<td>text</td>
<td></td>
<td>Specifies the column on which to sort for the table.</td>
</tr>
<tr>
<td>table.sortDirection</td>
<td>One of the following values.</td>
<td>asc</td>
<td>Indicates the sort direction for items in the table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>asc</strong>:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>desc</strong>:</td>
</tr>
<tr>
<td>table.drilldown</td>
<td>One of the following values.</td>
<td>all</td>
<td>Indicates whether drilldown functionality is enabled for the table.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>all</strong>: Drilldown is enabled.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>none</strong>: Drilldown is disabled.</td>
</tr>
<tr>
<td>table.wrap</td>
<td>Boolean</td>
<td>true</td>
<td>Indicates whether text in the table wraps.</td>
</tr>
<tr>
<td>type</td>
<td>One of the following values.</td>
<td>list</td>
<td>Indicates the format for displaying events.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>list</strong>:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>raw</strong>:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td><strong>table</strong>:</td>
</tr>
</tbody>
</table>

**Deprecated event options**

| Property | Type | Default | Description       | Alternatives |
|----------|------|---------|-------------------|--------------|--------------|

21
<table>
<thead>
<tr>
<th><strong>drilldown</strong></th>
<th><strong>String</strong></th>
<th><strong>all</strong></th>
<th><strong>Enable or disable drilldown types.</strong></th>
<th><strong>Use typed drilldown options (list.drilldown, table.drilldown, raw.drilldown) or the &lt;drilldown&gt; element to configure drilldown behavior.</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>entityName</strong></td>
<td><strong>String</strong></td>
<td><strong>events</strong></td>
<td><strong>Show events or results in the list.</strong></td>
<td><strong>Use type to specify content to show.</strong></td>
</tr>
<tr>
<td><strong>segmentation</strong></td>
<td><strong>String</strong></td>
<td><strong>none</strong></td>
<td><strong>Specify which event elements are clickable for drilldown.</strong></td>
<td><strong>Use list.drilldown or raw.drilldown to specify which elements in each event are clickable.</strong></td>
</tr>
<tr>
<td><strong>softWrap</strong></td>
<td><strong>Boolean</strong></td>
<td><strong>Configure event wrapping.</strong></td>
<td><strong>Use list.wrap or table.wrap to configure event wrapping.</strong></td>
<td></td>
</tr>
</tbody>
</table>

**Example**

```xml
<dashboard>
  . . .
  <row>
  <panel>
    <title>Events in the last 24 hours</title>
    <event>
      <search>
        <query>index=_internal</query>
        <earliest>-24h@h</earliest>
        <latest>now</latest>
        <sampleRatio>1</sampleRatio>
      </search>
      <option name="count">20</option>
      <option name="list.drilldown">full</option>
      <option name="list.wrap">1</option>
      <option name="maxLines">5</option>
      <option name="raw.drilldown">full</option>
      <option name="rawNumbers">0</option>
      <option name="table.drilldown">all</option>
      <option name="table.sortDirection">asc</option>
      <option name="table.wrap">1</option>
      <option name="type">list</option>
    </event>
  </panel>
  . . .
</row>
</dashboard>
```

**table**

The `<table>` element defines a statistics table visualization.
Attributes

You can use any of the shared attributes in a `<table>`.

Child elements

You can use the following child element in a `<table>`.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
</table>
| `<fields>` | Comma separated list | Restrict search results to these fields. The order of the listed fields determines table column order. | `<dashboard>`  
  `<label>Fields Example</label>`  
  `<row>`  
  `<panel>`  
  `<table>`  
  `<search>`  
  `<query>`  
  `index=_internal | timechart count by sourcetype`  
  `</query>`  
  `<earliest>-7d@d</earliest>`  
  `<latest>now</latest>`  
  `</search>`  
  `<fields>_time, splunkd, splunk_web_access, splunk_web_service</fields>`  
  `</table>`  
  `</panel>`  
  `</row>`  
  `</dashboard>` |

Options

In addition to the shared options for visualizations, you can use the following options to configure a table.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>count</td>
<td>Integer</td>
<td>10</td>
<td>The maximum number of rows to display.</td>
</tr>
<tr>
<td>Name</td>
<td>Type</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>-------------------------------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>dataOverlayMode</td>
<td>One of the following values.</td>
<td>none</td>
<td>Indicates which type of overlay to display.</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>none</td>
<td></td>
</tr>
<tr>
<td></td>
<td>heatmap</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>highlow</td>
<td></td>
<td></td>
</tr>
<tr>
<td>drilldown</td>
<td>One of the following values.</td>
<td>cell</td>
<td>Enables drilldown on row or cell level, or disables drilldown.</td>
</tr>
<tr>
<td></td>
<td>cell</td>
<td>cell</td>
<td>Enable drilldown on table cells.</td>
</tr>
<tr>
<td></td>
<td>row</td>
<td>row</td>
<td>Enable drilldown on table rows.</td>
</tr>
<tr>
<td></td>
<td>none</td>
<td>none</td>
<td>Disable drilldown.</td>
</tr>
<tr>
<td>rowNumbers</td>
<td>Boolean</td>
<td>false</td>
<td>Toggle display of row numbers.</td>
</tr>
<tr>
<td>showPager</td>
<td>Boolean</td>
<td>true</td>
<td>Toggle pagination on or off.</td>
</tr>
<tr>
<td>totalsRow</td>
<td>Boolean</td>
<td>false</td>
<td>Set to true to add a column totals row to the table.</td>
</tr>
<tr>
<td>percentagesRow</td>
<td>Boolean</td>
<td>false</td>
<td>Set to true to add a column percentages row to the table.</td>
</tr>
<tr>
<td>wrap</td>
<td>Boolean</td>
<td>true</td>
<td>Enable wrapping of text in the results table.</td>
</tr>
</tbody>
</table>

**Column formatting**

You can use color and other formatting in table columns to show context or highlight values. Table column formatting uses a different syntax from other visualization options. To learn about formatting table columns in Simple XML, see Table column Simple XML.

**Example**

Example of a table panel using an inline search, displaying five rows, and disabling row numbers:

```xml
<dashboard>
  <label>Dashboard with Table</label>
  <row>
    <panel>
      <table>
        <title>Top source types in the last 24 hours</title>
        <search>
          <query>
            index=_internal group=per_sourcetype_thruput
              | chart sum(kb) by series | sort -sum(kb)
          </query>
          <earliest>-24h</earliest>
          <latest>now</latest>
        </search>
        <option name="count">5</option>
        <option name="rowNumbers">0</option>
      </table>
    </panel>
  </row>
</dashboard>
```

**Table sparklines**

Table sparklines show recent behavior patterns for each table row. You can format a sparkline in Simple XML.
Single value sparklines work differently from table sparklines. See single value for more details.

Configuring sparkline formatting in Simple XML does not change sparkline appearance in PDF. Sparklines appear without additional formatting in PDF.

Generating a sparkline

To use the formatting options described here, your search must generate a sparkline in the visualization. To create a sparkline, use the `chart` or `stats` command `sparkline( )` function. See Add sparklines to search results in the Search Manual for more details.

**Format syntax and attributes**
The `<format>` element contains formatting rules for sparklines. Include the "sparkline" type attribute and the field that the sparkline is tracking.

```xml
<format type="sparkline" field="[field name]">
  ...
</format>
```

Put `<option>` child elements inside the `<format>` to configure sparkline appearance.

**Format element attributes**
Use the following attributes to configure the `<format>` element.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>type</td>
<td>String</td>
<td>Required. Use &quot;sparkline&quot;.</td>
</tr>
<tr>
<td>field</td>
<td>Search result field that the sparkline is tracking</td>
<td>Required. Use the same field that the <code>sparkline( )</code> command in your search is using to generate the sparkline. For example, you might use this search to generate sparklines for the <code>trend</code> field. ...</td>
</tr>
</tbody>
</table>

Use `trend` as the `field` value to format the sparklines.

```xml
<format type="sparkline" field="trend">
  ...
</format>
```

**Format options**

Use `<option>` elements inside the `<format>` element to define sparkline appearance. Some options are common to any sparkline. Other options are specific to the sparkline chart type.

25
### Common sparkline options

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>chartRangeMax</td>
<td>Number</td>
<td>n/a</td>
<td>Specify an alternate maximum sparkline range value.</td>
</tr>
<tr>
<td>chartRangeMin</td>
<td>Number</td>
<td>n/a</td>
<td>Specify an alternate minimum sparkline range value.</td>
</tr>
<tr>
<td>height</td>
<td>CSS style</td>
<td>auto</td>
<td>Chart height. Use any valid CSS width. For example, 1.5em or 20px.</td>
</tr>
<tr>
<td>tooltipPrefix</td>
<td>text</td>
<td></td>
<td>Text to place before each field displayed in a tooltip.</td>
</tr>
<tr>
<td>tooltipSuffix</td>
<td>text</td>
<td></td>
<td>Text to append to each field displayed in a tooltip.</td>
</tr>
<tr>
<td>type</td>
<td>(bar</td>
<td>discrete</td>
<td>line)</td>
</tr>
</tbody>
</table>

### Sparkline bar chart options

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>barSpacing</td>
<td>Number</td>
<td></td>
<td>Space between each bar, in pixels.</td>
</tr>
<tr>
<td>barWidth</td>
<td>Number</td>
<td></td>
<td>Width of each bar, in pixels.</td>
</tr>
<tr>
<td>colorMap</td>
<td>See description</td>
<td></td>
<td>Map field values to selected colors. For example, if you want all values of -2 to appear yellow, use colorMap: {'-2': '#ff0'}. You can also use a value array instead of a mapping to specify a color for each bar. For example if your chart has the values 1, 3, and 1, you can use colorMap=['red', &quot;green&quot;, &quot;blue&quot;].</td>
</tr>
</tbody>
</table>

### Sparkline discrete chart options

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>lineColor</td>
<td>CSS style</td>
<td></td>
<td>Used by line and discrete charts to specify the color of the line drawn as a CSS values string.</td>
</tr>
<tr>
<td>lineHeight</td>
<td>Number</td>
<td>30% of graph height</td>
<td>Height of each line, in pixels.</td>
</tr>
<tr>
<td>thresholdColor</td>
<td>CSS color</td>
<td></td>
<td>CSS color to use in combination with thresholdValue.</td>
</tr>
<tr>
<td>thresholdValue</td>
<td>CSS color</td>
<td></td>
<td>Draw values less than this threshold using thresholdColor instead of lineColor.</td>
</tr>
</tbody>
</table>

### Sparkline line chart options

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fillColor</td>
<td>CSS color</td>
<td></td>
<td>Specify the color to fill the area under the graph as a CSS value. Set to false to disable fill.</td>
</tr>
<tr>
<td>highlightLineColor</td>
<td>CSS color</td>
<td>#f22</td>
<td>CSS color for the vertical line that appears through a value when moused over. Set to null to disable.</td>
</tr>
<tr>
<td>highlightSpotColor</td>
<td>CSS color</td>
<td>#f5f</td>
<td>Color for the spot that appears on a value when moused over.</td>
</tr>
<tr>
<td>Property</td>
<td>Type</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------</td>
<td>---------</td>
<td>---------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>lineColor</td>
<td>CSS style</td>
<td></td>
<td>Set to null to disable.</td>
</tr>
<tr>
<td>lineWidth</td>
<td>Number</td>
<td>1</td>
<td>line width, in pixels.</td>
</tr>
<tr>
<td>maxSpotColor</td>
<td>CSS color</td>
<td></td>
<td>CSS color of the marker displayed for the maximum value.</td>
</tr>
<tr>
<td>minSpotColor</td>
<td>CSS color</td>
<td></td>
<td>CSS color of the marker displayed for the minimum value.</td>
</tr>
<tr>
<td>normalRangeMax</td>
<td>Number (see description)</td>
<td></td>
<td>Use with normalRangeMin to define threshold values for a normal or expected value range. Bars appear in the sparkline to show which values are within and outside of this range.</td>
</tr>
<tr>
<td>normalRangeMin</td>
<td>Number</td>
<td></td>
<td>Use with normalRangeMax to define threshold values for a normal or expected value range. Bars appear in the sparkline to show which values are within and outside of this range.</td>
</tr>
<tr>
<td>spotColor</td>
<td>CSS color</td>
<td></td>
<td>CSS color of the final value marker.</td>
</tr>
<tr>
<td>spotRadius</td>
<td>Number</td>
<td>1.5</td>
<td>Radius, in pixels, of all spot markers.</td>
</tr>
<tr>
<td>valueSpots</td>
<td>See description</td>
<td></td>
<td>Points and colors for drawing value spots. Accepts a range. For example, to render green spots on all values less than 50 and red on values higher than 50, use {':49': 'green', '50': 'red'}.</td>
</tr>
<tr>
<td>width</td>
<td>CSS style</td>
<td>auto</td>
<td>Chart width. Specify any valid CSS width (for example, 1.5em or 20px).</td>
</tr>
</tbody>
</table>

**Example**

This Simple XML source code generates a table with a bar chart sparkline.

```xml
<dashboard>
  <label>Sparkline Example</label>
  <row>
    <panel>
      <table>
        <title>Basic Sparkline Bar w/ Color Map</title>
        <!-- Set span for each sparkline datapoint to be 1 hour -->
        <search>
          <query>
            index=_internal | chart count sparkline(count, 1h) as trend by sourcetype | sort -count
          </query>
          <earliest>-24h@h</earliest>
          <latest>now</latest>
        </search>
        <option name="count">3</option>
        <!-- Set sparkline options here; make sure that field matches field name of the search results -->
        <format type="sparkline" field="trend">
          <option name="type">bar</option>
        </format>
      </table>
    </panel>
  </row>
</dashboard>
```
chart

Use the `<chart>` element to add column, bar, line, area, bubble, or scatter charts to a `<panel>`. You can also use the `<chart>` element to add a gauge visualization to a `<panel>`.

The search generating a chart can be an inline search or a saved report. A saved report can contain chart format settings. These settings are used by default, but you can use options and attributes in your Simple XML source code to override them.

For more information on saving reports, see Create and edit reports.

Attributes

Use any of the shared attributes for visualizations in a chart.

Options

In addition to the shared options for visualizations, you can use the following options for general chart configuration. See the Chart Configuration Reference for a complete list of chart options.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>charting.chart</td>
<td>area</td>
<td>bar</td>
<td>column</td>
</tr>
</tbody>
</table>
### Charting Options

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>fillerGauge</td>
<td>line</td>
<td></td>
<td>the chart type.</td>
</tr>
<tr>
<td>markerGauge</td>
<td>pie</td>
<td></td>
<td></td>
</tr>
<tr>
<td>radialGauge</td>
<td>scatter</td>
<td></td>
<td></td>
</tr>
<tr>
<td>charting.legend.placement</td>
<td>top</td>
<td>left</td>
<td>bottom</td>
</tr>
<tr>
<td>charting.&lt;option_name&gt;</td>
<td></td>
<td></td>
<td>Formatting options supported for the chart. See the Custom Chart Reference to review available options for each chart type.</td>
</tr>
<tr>
<td>height</td>
<td>Number between 100-10000</td>
<td>250</td>
<td>Chart height, in pixels</td>
</tr>
</tbody>
</table>

### Example

```html
<form>
  ...
  <row>
    <panel>
      <title>Chart example</title>
      <chart>
        <title>Top sourcetypes in the last week</title>
        <search>
          <query>index=_internal | timechart sum(kb) by series | head 5</query>
          <earliest>-4h@m</earliest>
          <latest>now</latest>
        </search>
        <option name="height">200</option>
        <option name="charting.chart">line</option>
        <option name="charting.axisY.scale">log</option>
        <option name="charting.chart.nullValueMode">connect</option>
      </chart>
    </panel>
  </row>
  ...
</form>
```

### Map

Use the `<map>` element to add a Choropleth or cluster map to a `<panel>`. 
Attributes

In addition to the shared attributes for visualizations, you can use the following attributes in a `<map>` element.

**General map options**

In addition to the shared options for visualizations, you can use the following options to configure Choropleth and cluster maps.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>drilldown</td>
<td>all</td>
<td>none</td>
<td>all</td>
</tr>
<tr>
<td>mapping.fieldColors</td>
<td>One or more &lt;field&gt; : &lt;color_hexvalue&gt; mappings.</td>
<td></td>
<td>A comma-separated list of field names mapped to hexadecimal color values (0xRRGGBB) to define colors for specific series.</td>
</tr>
<tr>
<td>mapping.map.center</td>
<td>(lat, long)</td>
<td></td>
<td>The initial center point of the map. Latitude values can range from -85 to 85, with values outside of this range being clipped. Longitude values can range from -180 to 180, with values outside of this range being wrapped to fall within it.</td>
</tr>
<tr>
<td>mapping.map.scrollZoom</td>
<td>Boolean</td>
<td>false</td>
<td>Indicates whether the map zooms when a user scrolls the map.</td>
</tr>
<tr>
<td>mapping.map.panning</td>
<td>Boolean</td>
<td>true</td>
<td>Indicates whether the map pans when dragged.</td>
</tr>
<tr>
<td>mapping.map.zoom</td>
<td>Number</td>
<td>2</td>
<td>The initial zoom level of the map.</td>
</tr>
<tr>
<td>mapping.map.fitBounds</td>
<td>(south-lat, west-long, north-lat, east-long)</td>
<td></td>
<td>The initial bounds to fit within the map view area. Latitude values can range from -85 to 85, with values outside of this range being clipped. Longitude values can range from -180 to 180, with values outside of this range being wrapped to fall within it. Values assigned to this property effectively override any values assigned to the center or zoom properties.</td>
</tr>
<tr>
<td>mapping.seriesColors</td>
<td>List</td>
<td>Default*</td>
<td>Comma-separated list of hexadecimal color values (0xRRGGBB). The list indicates sampling colors for series with no specific fieldColors color assigned to them.</td>
</tr>
<tr>
<td>mapping.showTiles</td>
<td>boolean</td>
<td>true</td>
<td>Specify whether to show or hide map tiles.</td>
</tr>
</tbody>
</table>
### Mapping tileLayer properties

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapping.tileLayer.tileOpacity</td>
<td>text</td>
<td>1</td>
<td>Specifies the opacity of the tiles. Values can range from 0 (transparent) to 1 (opaque).</td>
</tr>
<tr>
<td>mapping.tileLayer.url</td>
<td>URL template</td>
<td>See description</td>
<td>The URL to use for requesting tiles, based on the following template. http://(s).tile.openstreetmap.org/(z)/(x)/(y).png</td>
</tr>
<tr>
<td>mapping.tileLayer.subdomains</td>
<td>[string,...]</td>
<td>[a,b,c]</td>
<td>List of subdomains over which to distribute tile requests. More subdomains allows more tiles to be requested simultaneously. See example below.</td>
</tr>
<tr>
<td>mapping.tileLayer.minZoom</td>
<td>Integer</td>
<td>0</td>
<td>The minimum zoom level of the tileset.</td>
</tr>
<tr>
<td>mapping.tileLayer.maxZoom</td>
<td>Integer</td>
<td>7</td>
<td>The maximum zoom level of the tileset. Use any non-negative integer to specify the maximum zoom level.</td>
</tr>
<tr>
<td>mapping.tileLayer.invertY</td>
<td>Boolean</td>
<td>False</td>
<td>Whether to invert the y coordinate for tile requests. TMS servers use inverse y-axis numbering.</td>
</tr>
<tr>
<td>mapping.tileLayer.attribution</td>
<td>String</td>
<td>See description</td>
<td>A copyright attribution to be displayed in the bottom right corner of the map. The default value: Map data (c) 2012 OpenStreetMap contributors, CC-BY-SA. See example below.</td>
</tr>
</tbody>
</table>

### Choropleth map options

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapping.choroplethLayer.colorBins</td>
<td>integer</td>
<td>5</td>
<td>Specifies the number of color bins to use.</td>
</tr>
<tr>
<td>mapping.choroplethLayer.colorMode</td>
<td>“sequential”</td>
<td>‘auto’</td>
<td>Specifies the color mode to use for the choropleth shapes. Possible modes are 'sequential', 'divergent', or 'categorical'.</td>
</tr>
<tr>
<td>mapping.choroplethLayer.maximumColor</td>
<td>text</td>
<td>DB5800</td>
<td>Specifies the color to use for the highest value shapes.</td>
</tr>
<tr>
<td>mapping.choroplethLayer.minimumColor</td>
<td>text</td>
<td>2F25BA</td>
<td>Only used when the color mode is divergent. The color to use for the lowest value shapes.</td>
</tr>
<tr>
<td>mapping.choroplethLayer.neutralPoint</td>
<td>text</td>
<td>0</td>
<td>Only used when the color mode is divergent. The value where the color palette should switch from using the minimum color to the maximum color.</td>
</tr>
<tr>
<td>mapping.choroplethLayer.shapeOpacity</td>
<td>text</td>
<td>0.75</td>
<td>Specifies the opacity of the shapes. Values can range from 0 (transparent) to 1 (opaque).</td>
</tr>
<tr>
<td>mapping.choroplethLayer.showBorder</td>
<td>Boolean</td>
<td>true</td>
<td>Specifies whether to show borders around each shape.</td>
</tr>
<tr>
<td>mapping.showTiles</td>
<td>Boolean</td>
<td>true</td>
<td>Determines whether the map tiles are shown.</td>
</tr>
</tbody>
</table>

31
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapping.tileLayer.tileOpacity</td>
<td>text</td>
<td>1</td>
<td>Specifies the opacity of the tiles. Values can range from 0 (transparent) to 1 (opaque).</td>
</tr>
</tbody>
</table>

### Cluster map options

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mapping.data.maxClusters</td>
<td>Integer</td>
<td>100</td>
<td>The maximum number of clusters to render in a cluster map. Caution: Setting this option to a large number of clusters can significantly degrade performance. Use a value below 1000.</td>
</tr>
<tr>
<td>mapping.markerLayer.markerOpacity</td>
<td>Number</td>
<td>0.8</td>
<td>Opacity of cluster map markers. Values can range from 0 (transparent) to 1 (opaque).</td>
</tr>
<tr>
<td>mapping.markerLayer.markerMinSize</td>
<td>Number</td>
<td>10</td>
<td>The minimum size of cluster map markers, in pixels.</td>
</tr>
<tr>
<td>mapping.markerLayer.markerMaxSize</td>
<td>Number</td>
<td>50</td>
<td>The maximum size of cluster map markers, in pixels.</td>
</tr>
</tbody>
</table>

### Examples

The following examples show you how to use different map configurations.

**mapping.data.maxClusters**

The following example sets the maximum number of clusters to 250.

```xml
<map>
  <option name="mapping.data.maxClusters">250</option>
</map>
```

**mapping.fieldColors and mapping.seriesColors**

The following example configures the "foo" and "bar" fields to be red (0xFF0000) and green (0x00FF00), respectively, and configures all other fields to be blue (0x0000FF).

```xml
<map>
  <option name="mapping.fieldColors">{foo:0xFF0000,bar:0x00FF00}</option>
  <option name="mapping.seriesColors">[0x0000FF]</option>
</map>
```

**mapping.map.fitBounds example**

The following example initializes the map view to a boundary around San Francisco.

```xml
<map>
  <option name="mapping.map.fitBounds">
```

32
mapping.tileLayer.*

The following example configures the client to request tiles from openstreetmap.org (this is the default configuration).

```
<map>
  <option name="mapping.tileLayer.url">http://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png</option>
  <option name="mapping.tileLayer.subdomains">[a,b,c]</option>
  <option name="mapping.tileLayer.maxZoom">18</option>
  <option name="mapping.tileLayer.attribution">
    Map data (c) 2012 OpenStreetMap contributors, CC-BY-SA.
  </option>
</map>
```

map example using foursquare data

This example assumes you are indexing foursquare data as source foursquare.

```
<map>
  <title>Roma</title>
  <search>
    sourcetype=foursquare
      | geostats latfield=checkin.geolat longfield=checkin.geolong count by checkin.user.gender
  </search>
  <option name="mapping.data.maxClusters">500</option>
  <option name="mapping.markerLayer.markerMaxSize">20</option>
  <option name="mapping.map.fitBounds">(41.3,12.7,41.5,12.8)</option>
  <option name="mapping.seriesColors">[0x0060DD]</option>
  <option name="mapping.map.zoom">4</option>
</map>
```

Complete Choropleth map example

```
<dashboard>...
  <row>
    <panel>
      <title>choropleth map</title>
      <map>
        <search>
          <query>source="my_mapping_data_source" | iplocation clientip | lookup geo_us_states latitude as lat longitude as lon | stats count by featureId | geom geo_us_states</query>
          <earliest>-24h@h</earliest>
          <latest>now</latest>
          <sampleRatio>1</sampleRatio>
        </search>
        <option name="drilldown">all</option>
        <option name="mapping.choroplethLayer.colorBins">5</option>
        <option name="mapping.choroplethLayer.colorMode">auto</option>
        <option name="mapping.choroplethLayer.maximumColor">0xFFB600</option>
        <option name="mapping.choroplethLayer.minimumColor">0x2F25BA</option>
        <option name="mapping.choroplethLayer.neutralPoint">0</option>
        <option name="mapping.choroplethLayer.shapeOpacity">0.75</option>
        <option name="mapping.choroplethLayer.showBorder">1</option>
        <option name="mapping.data.maxClusters">100</option>
        <option name="mapping.legend.placement">bottomright</option>
      </map>
    </panel>
  </row>
```
The `<single>` element defines a single value visualization in a `<panel>`.

![Single Value Visualization](image)

### Attributes

You can use any of the shared attributes for visualizations in a `<single>` value element.

### Options

In addition to the shared options for visualizations, you can use the following options to configure a single value visualization.

Use the following options to configure a `<single>` value visualization.

**Note:** Configurations made in the **Format** menu can override Simple XML settings.

<table>
<thead>
<tr>
<th>Option name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>colorBy</td>
<td>One of the following values: <code>value</code>, <code>trend</code></td>
<td>value</td>
<td>Indicate whether to show color changes in the single value or the trend indicator. To use this option, the <code>useColors</code> option must be set to <code>true</code>. If you use <code>value</code>, the value range determines the color used. Color by trend is only available if the visualization search uses the <code>timechart</code> command. If you use <code>trend</code>, the indicator appears in black (no</td>
</tr>
<tr>
<td>Option name</td>
<td>Type</td>
<td>Default</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>------------------</td>
<td>---------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>colorMode</td>
<td>One of the following values.</td>
<td>none</td>
<td>Specify whether the single value or the background shows the value range color. Use <code>block</code> to show color in the background with white text. Use <code>none</code> to show color in the single value with a white background.</td>
</tr>
<tr>
<td>drilldown</td>
<td>One of the following values.</td>
<td>none</td>
<td>Use <code>all</code> to enable drilldown on the singe value. Use <code>none</code> to disable drilldown.</td>
</tr>
<tr>
<td>field</td>
<td>field name</td>
<td>Defaults to the first field returned</td>
<td>Indicate which field's single value to display in the visualization.</td>
</tr>
<tr>
<td>numberPrecision</td>
<td>string</td>
<td>0</td>
<td>Specify how many decimal places to display. For decimal precision, indicate the number of places using 0. followed by up to four zeroes. For example, 0.0 or 0.00.</td>
</tr>
<tr>
<td>rangeColors</td>
<td>String array</td>
<td>Defaults to standard range colors: red, orange, yellow, blue, and green. If there are more rangeColors values than ranges, excess color values are ignored. If there are more rangeValues values than rangeColors hex colors, the single value appears dark grey if it falls within a range without a color mapped to it.</td>
<td></td>
</tr>
<tr>
<td>rangeValues</td>
<td>Number array</td>
<td>none</td>
<td>Use an array of numbers to specify ranges for the single value. Ranges determine single value color if you are using color by value. Use the rangeColors option to specify colors that map to rangeValues values.</td>
</tr>
<tr>
<td>showSparkline</td>
<td>boolean</td>
<td>true</td>
<td>Indicate whether to show the single value sparkline, if one is available. A sparkline is only available for searches that include the timechart search command.</td>
</tr>
<tr>
<td>showTrendIndicator</td>
<td>boolean</td>
<td>true</td>
<td>Show or hide the single value trend indicator. A trend indicator is only available when the visualization query includes the timechart command.</td>
</tr>
<tr>
<td>trendColorInterpretation</td>
<td>One of the following values.</td>
<td>standard</td>
<td>Specifies whether a field value greater than 0 is shown as a positive (standard) or negative (inverse) trend.</td>
</tr>
<tr>
<td>trendDisplayMode</td>
<td>One of the following values.</td>
<td>absolute</td>
<td>Specifies whether the delta value is displayed as a percentage or an absolute count.</td>
</tr>
</tbody>
</table>

change, green (positive), or red (negative). Use the trendColorInterpretation option to configure positive and negative value interpretation.
<table>
<thead>
<tr>
<th>Option name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>trendInterval</td>
<td>SPL time modifier</td>
<td>auto</td>
<td>Indicate the start time for calculating the trend delta. The trend is evaluated from the start time to the most recent data point. Use the search syntax for time modifiers to indicate the range.</td>
</tr>
<tr>
<td>underLabel</td>
<td>String</td>
<td></td>
<td>Caption for the visualization.</td>
</tr>
<tr>
<td>useColors</td>
<td>Boolean</td>
<td>false</td>
<td>Enable or disable color in the visualization. Set to true for value and trend color options to be available.</td>
</tr>
<tr>
<td>useThousandSeparators</td>
<td>boolean</td>
<td>true</td>
<td>Specifies whether the numeric value includes thousand separator characters. For example, 1,000 includes a thousand separator.</td>
</tr>
<tr>
<td>unit</td>
<td>Brief text label</td>
<td>none</td>
<td>Specify a label to show next to the single value. The unit text should be brief. Typically, an abbreviation, such as MB, or a symbol, such as $, can be used.</td>
</tr>
<tr>
<td>unitPosition</td>
<td>One of the following values.</td>
<td>after</td>
<td>Specify whether the unit label should appear before or after the single value.</td>
</tr>
</tbody>
</table>

**Deprecated options**

<table>
<thead>
<tr>
<th>Property</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>additionalClass</td>
<td>CSS class name</td>
<td></td>
<td>CSS class name to add to the result container.</td>
</tr>
<tr>
<td>afterLabel</td>
<td>String</td>
<td></td>
<td>Visualization label</td>
</tr>
<tr>
<td>beforeLabel</td>
<td>String</td>
<td></td>
<td>Visualization label</td>
</tr>
<tr>
<td>classField</td>
<td>String</td>
<td></td>
<td>Adds the value of the classField of the first result as an additional CSS class to the result container.</td>
</tr>
<tr>
<td>linkFields</td>
<td>String</td>
<td>result</td>
<td>Indicates the visualization element clickable for drilldown. The single value itself is clickable for drilldown. Use the &lt;drilldown&gt; element to configure drilldown behavior.</td>
</tr>
<tr>
<td>linkSearch</td>
<td>Search string</td>
<td></td>
<td>Search whose results are clickable for drilldown. The single value itself is clickable for drilldown. Use the &lt;drilldown&gt; element to configure drilldown behavior.</td>
</tr>
<tr>
<td>linkView</td>
<td>Dashboard or form</td>
<td></td>
<td>Dashboard or form used with linkSearch search for drilldown. Use the &lt;drilldown&gt; element to configure drilldown behavior, including linking to a dashboard or form.</td>
</tr>
<tr>
<td>refresh.auto.interval</td>
<td>integer</td>
<td>0</td>
<td>Specifies a visualization refresh interval. To disable panel refresh, specify 0 or a negative integer. Configure dashboard or form refresh behavior using the refresh attribute or use the &lt;search&gt; element &lt;refresh&gt; child element to configure panel refresh behavior.</td>
</tr>
</tbody>
</table>
Example

```xml
<dashboard>
  <panel depends="$show_single_value$">
    <title>Event count for $selected_sourcetype$</title>
    <single>
      <search>
        <query>index=_internal sourcetype=$selected_sourcetype$ | stats count</query>
      </search>
      <option name="colorMode">block</option>
      <option name="drilldown">all</option>
      <option name="rangeColors">["0x65a637","0x6db7c6","0xf58f39","0xd93f3c"]</option>
      <option name="rangeValues">[0,30,100]</option>
      <option name="underLabel">events</option>
      <option name="useColors">1</option>
    </single>
  </panel>
  ...
</dashboard>
```

### html

Show inline HTML content.

**Attributes**

In addition to the shared attributes for visualizations, you can use the following attributes in an `<html>` element.

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>encoded</td>
<td>boolean</td>
<td>false</td>
<td>For internal use only.</td>
</tr>
<tr>
<td>src</td>
<td>String</td>
<td></td>
<td>Put the contents of an HTML or image file into the <code>&lt;html&gt;</code> panel. Directory locations and Simple XML syntax are different for HTML files and image files.</td>
</tr>
</tbody>
</table>

**HTML file from the same app context**

```xml
<html src="<file_name>.html">
</html>
```

**HTML file from a different app context**

```xml
<html src="<other_app_name>:<file_name>.html">
</html>
```

**Image file**

```xml
<html>
<img src="/static/app/<app_name>/images/<file_name>.png">
</html>
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>tokens</td>
<td>boolean</td>
<td>true</td>
<td>If false, disables token replacement for the <code>&lt;html&gt;</code> panel.</td>
</tr>
</tbody>
</table>

See the following instructions for details on where to save HTML and image files and how to reference files from different app contexts.

Use an HTML file in a dashboard panel

**Steps**

1. Put the HTML file in the following directory. `$SPLUNK_HOME/etc/apps/<appname>/appserver/static`
2. In the `<html>` panel, use this syntax to indicate a file from the current app context.
   ```html
   <html src="<file_name>.html">
   </html>
   ```
   If you are specifying an HTML file from another app context, use this syntax.
   ```html
   <html src="<other_app_name>:<file_name>.html">
   </html>
   ```

Use an image file in a dashboard panel

**Steps**

1. Put the image file in the following directory. `$SPLUNK_HOME/etc/apps/<appname>/appserver/static/images`
   If an `/images` directory does not already exist, create one and put the file in it.
2. Verify that the image file path is accessible by testing the following URL.
   ```text
   http://<host>:<port>/static/app/<app_name>/images/<image>
   ```
   For example, use this URL to verify that the `my_image.png` file is accessible.
   ```text
   http://localhost:8000/static/app/search/images/my_image.png
   ```
3. In the `<html>` panel, use this syntax to indicate a file from the current app context.
   ```html
   <html>
   <img src="/static/app/search/images/<file_name>.png">
   </img>
   </html>
   ```

**Options**

There are no child options for the `<html>` element.

**Example**

```xml
<dashboard>
  <label>test_db</label>
  <row>
    <panel>
```

38
search

Use the `<search>` element to define a search in a dashboard, form, or panel. You can also use a `<search>` to define input choices dynamically.

Search types

The `<search>` element can define any of the following search types.

<table>
<thead>
<tr>
<th>Search type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inline</td>
<td>Define a search directly in the dashboard, form, or panel. Use the <code>&lt;query&gt;</code> element to specify the search string. You can add time range modifiers and other options to configure the search.</td>
</tr>
<tr>
<td>Report reference</td>
<td>Use a report search and visualization in a panel. Add the <code>ref</code> attribute and the report name to the <code>&lt;search&gt;</code> element. You can configure the visualization from the referenced report and add custom time range modifiers to the search. You cannot modify the search string in Simple XML. Edit the report to adjust the search string. When you edit a report search, panels with references to the report update automatically.</td>
</tr>
<tr>
<td>Populating search for input choices</td>
<td>Use <code>&lt;search&gt;</code> as a child element of a form input to populate choices for checkbox, dropdown, multiselect, or radio inputs. Indicate <code>&lt;fieldForLabel&gt;</code> and <code>&lt;fieldForValue&gt;</code> search result fields to populate the input choice labels and choice values. Do not use real-time searches for populating searches. Input choice labels and values do not update as real-time search results generate.</td>
</tr>
</tbody>
</table>
| Base search            | A base search generates transformed results for post-process searches to modify. A base search can be global, defined at the `<dashboard>` or `<form>` level. You can also use a base search at the `<panel>` level in Simple XML. Make sure that all base searches meet the following requirements.  
  1. Use transforming commands in the base search to generate transformed results. Non-transforming base searches can cause performance and timeout issues.  
  2. The base search must have an `<id>` attribute for post-process searches to reference. |
| Post-process search    | A post-process search accepts transformed results from a base search and performs additional search processing on these results. Use the `base` and `id` attributes to reference the base search id for a post-process search. Specify `<earliest>` and `<latest>` elements in the base search. A post-process search ignores time modifiers defined within the post-process `<search>` element. For more information on post-process searches, see Post-process searches in this manual. |
Element hierarchy

You can put a `<search>` element into any of the following parent elements.

- `<form>`
- `<dashboard>`
- `<panel>`
- `<chart>`
- `<event>`
- `<map>`
- `<single>`
- `<table>`

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>app</td>
<td>text</td>
<td></td>
<td>The name of an app. Use the <code>app</code> attribute with the <code>ref</code> attribute to reference a report that is not in the current app.</td>
</tr>
<tr>
<td>base</td>
<td>text</td>
<td></td>
<td>In a post-process search, references a base search in this dashboard, form, or panel. Use <code>base</code> with the <code>id</code> attribute to indicate the base search that the post-process search should use.</td>
</tr>
<tr>
<td>depends</td>
<td>List of one or more tokens</td>
<td></td>
<td>All of the tokens listed must be defined for the search to dispatch. If the search does not dispatch, the panel does not render.</td>
</tr>
<tr>
<td>id</td>
<td>Text (minimum two characters)</td>
<td></td>
<td>Search identifier. In a post-process search, reference the base search using this identifier. Make sure that search identifiers follow these requirements.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Use only alphanumeric or underscore characters. - The <code>id</code> cannot begin with a number or the underscore character. - The following terms are reserved for internal use and cannot be used for an <code>id</code>.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>♦ dashboard</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>♦ default</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>♦ footer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>♦ header</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>♦ search</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>♦ submitted</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>♦ url</td>
</tr>
<tr>
<td>ref</td>
<td>text</td>
<td></td>
<td>Reference to a report. Adds the report search and visualization to the dashboard or form. If you are referencing a report in another app, use the <code>app</code> attribute to specify the app.</td>
</tr>
<tr>
<td>rejects</td>
<td>List of one or more tokens</td>
<td></td>
<td>Stop the search from dispatching if one or more tokens in this list are defined. If the search does not dispatch, the panel does not render.</td>
</tr>
</tbody>
</table>

Child elements

Use the following child elements within a `<search>`.

<table>
<thead>
<tr>
<th>Element</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;cache&gt;</code></td>
<td>For saved searches, use one of the following values.</td>
<td>scheduled</td>
<td>- <code>true</code>: Always use the results from a preexisting saved search job when possible. - <code>false</code>: Never use results from preexisting saved search jobs. - <code>scheduled</code>: Reuse any previously run scheduled saved search jobs.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Element | Type | Default | Description
--- | --- | --- | ---
| | • scheduled • [integer] | • [integer]: The number of seconds indicating the maximum saved search job results age. Only results that are newer than this number of seconds are used. | 
| <cancelled> | N/A | N/A | Execute actions when a search is cancelled. |
| <done> | N/A | N/A | Execute actions based on finished search events. Includes job properties and first result row. |
| <error> | N/A | N/A | Execute actions when there is a search error event, such as an invalid query. |
| <earliest> and <latest> | text | Optional time expressions that specify the earliest and latest time parameters for a search. Post-process searches ignore child <earliest> and <latest> elements. Instead, the <earliest> and <latest> elements from the base search are used. You can specify the time as relative time or absolute time. For relative time, use relative time modifiers, as described in Specify relative time ranges in your search in the Search Manual. For absolute time, specify the time in UNIX epoch time format. |
| <progress> | | Execute an action on search progress events. Access job properties and the first results row. |
| <query> | text | Search string. |
| <refresh> | Integer or relative time expression | No refresh | Indicate a delay or interval time for inline or saved searches. This setting does not apply to post-process searches, which refresh automatically when their base search refreshes. Integers are handled as seconds. Use SPL syntax for relative time expressions. For example, 1h5m or 5m. Use the <refreshType> setting to specify refresh behavior in relation to search completion or dispatch. You can use the <refresh.display> setting in a visualization to specify a refresh progress indicator. |
| <refreshType> | interval or delay | delay | Indicate the starting time for counting down to a refresh. Use delay to start counting when the search is done. Use interval to count down when the search is dispatched. If the runtime of the search is longer than the configured time, the search job is cancelled and a new job is dispatched. |
| <sampleRatio> | number | Event sampling ratio. To learn more, see Event sampling with reports and dashboard panels in the Search Manual. |

Examples

Base search from inline search

```
<search id=[base ID]>
  <query>[search string]</query> (1)
```

41
<earliest> (0..1)  
<latest> (0..1)

**Base search from report**

<search id=[base ID] [ref=[report name]]>
  <earliest> (0..1)  
  <latest> (0..1)

**Post-process search**

<search base=[base ID]> (0..n)  
  <query>[post-process search string]</query> (1)

**Dashboard with base search and two post-process searches**

<dashboard>
  <label>Dashboard with post-process search</label>
  <description></description>
  <!-- Example uses stats transforming command -->  
  <!-- This limits events passed to post-process search -->
  <search id="baseSearch">
    <query>
      index=_internal source=*splunkd.log | stats count by component, log_level
    </query>
    <earliest>-30d</earliest>
    <latest>now</latest>
  </search>
  <row>
    <panel>
      <chart>
        <title>Post-process 1: Event count by log level</title>
        <!-- post-process search -->
        <search base="baseSearch">
          <query>
            stats sum(count) AS count by log_level
          </query>
        </search>
      </chart>
    </panel>
    <panel>
      <chart>
        <title>Post-process 2: Error count by component</title>
        <!-- post-process search -->
        <search base="baseSearch">
          <query>
            search log_level=error | stats sum(count) AS count by component
          </query>
        </search>
        <option name="charting.chart">bar</option>
      </chart>
    </panel>
  </row>
</dashboard>
Dashboard with empty post-process search

<dashboard>
  <label>Dashboard with empty post-process search</label>
  <description></description>
  <!-- Example uses stats transforming command -->
  <!-- This limits events passed to post-process search -->
  <search id="baseSearch">
    <query>index=_internal source=*splunkd.log | stats count by component, log_level</query>
    <earliest>-30d</earliest>
    <latest>now</latest>
  </search>
  <row>
    <panel>
      <chart>
        <title>Count by component, log level (from post-process search)</title>
        <!-- post-process search -->
        <search base="baseSearch">
          <query>stats sum(count) AS count by log_level</query>
        </search>
        <option name="charting.axisY.scale">log</option>
      </chart>
    </panel>
    <panel>
      <chart>
        <title>Count by component (from base search)</title>
        <!-- empty post-process search -->
        <search base="baseSearch" />
        <option name="charting.chart">bar</option>
      </chart>
    </panel>
  </row>
</dashboard>
selection (area, column, and line charts)

Use a `<selection>` element in area, column, or line charts to configure pan and zoom time window parameters or to set other token values.

See Chart controls for more details on working with chart pan and zoom.

Parent elements

```xml
<chart>
<option name="charting.chart">area</option> | <option name="charting.chart">column</option> | <option name="charting.chart">line</option>
```

Use predefined tokens to capture the earliest and latest time of the time window and the earliest and latest values within that time window for a field.

Example

```xml
<selection>
  <set token="selection.earliest">$start$</set>
  <set token="selection.latest">$end$</set>
  <set token="start.[fieldname]">$start.[fieldname]$</set>
  <set token="end.[fieldname]">$end.[fieldname]$</set>
</selection>
```

You can also `<link>` to a target dashboard, form, or external website in a `<selection>` element.

Attributes

None

Example

A selection on the left chart zooms into the right chart with details for the selected area.

```xml
<dashboard>
  <label>Pan and Zoom</label>
  <row>
    <panel>
      <chart>
        <title>Pan and Zoom (All source types)</title>
```
<search>
  <query>
    index=_internal | timechart count by sourcetype
  </query>
  <earliest>-7d@h</earliest>
  <latest>now</latest>
</search>
<option name="charting.axisX.scale">linear</option>
<option name="charting.axisY.scale">log</option>
<option name="charting.chart">line</option>
</selection>
<option name="charting.axisTitleX.text">Last 7 Days</option>
</chart>
</panel>
<panel>
<chart>
<title>Pan and Zoom (Web access source type)</title>
<search>
  <query>
    index=_internal sourcetype=splunk_web_access
    | timechart count by sourcetype
  </query>
  <earliest>$selection.earliest$</earliest>
  <latest>$selection.latest$</latest>
</search>
<option name="charting.chart">column</option>
<option name="charting.legend.placement">none</option>
<option name="charting.legend.masterLegend">null</option>
<option name="charting.axisX.scale">linear</option>
<option name="charting.axisY.scale">log</option>
<option name="charting.axisTitleX.text">Selected Time Range</option>
</chart>
</panel>
<html>
<h3>Token values for the splunk_web_access selection</h3>
<table border="0" cellpadding="12" cellspacing="0">
  <tr>
    <td>
      <p><b>Time range (epoch time)</b></p>
      <p>
        $$selection.earliest$$: $selection.earliest$<br/>
        $$selection.latest$$: $selection.latest$
      </p>
    </td>
    <td>
      <p><b>Count at the begining and end of time range.</b></p>
      <p>
        $$start.splunk_web_access$$: $start.splunk_web_access$<br/>
        $$end.splunk_web_access$$: $end.splunk_web_access$
      </p>
    </td>
  </tr>
</table>
</html>
</panel>
drilldown

Drilldown lets you build dynamic responses to user input or interactions. Use the following elements to configure drilldown behavior.

See Use drilldown for dashboard interactivity for details on using drilldown in dashboards and forms.

Use the `<drilldown>` element to start implementing drilldown behavior. The `<drilldown>` element contains child elements that define the response to a user click.

**Drilldown actions**
Insie a `<drilldown>` element, use `<set>`, `<unset>`, or `<eval>` to update token values as part of building dynamic behavior or display in the current dashboard or form. Use `<link>` to open a search, another dashboard, or an external website. You can pass token values to a linked target on a user click.

**Drilldown conditions**
Define conditional responses to different user actions by adding a `<condition>` element within the `<drilldown>`.

**Use actions or conditions as drilldown child elements**
You cannot combine actions and conditions as direct child elements of `<drilldown>`. If you are defining conditional drilldown behavior, you cannot include action elements directly inside the `<drilldown>` element. If you are defining actions directly inside the `<drilldown>` element, you cannot add `<condition>` elements directly inside the drilldown.

**Attributes**
None

**Element hierarchy**
Put a `<drilldown>` element inside any of the following visualization parent elements.

- `<chart>`
- `<event>`
Child elements

Use the following child elements inside a `<drilldown>` to configure the actions that happen when users click on the element where drilldown is enabled.

<drilldown>
  ( <link> | <set>  | <unset> ) (1..n) | <condition> (1..n)
</drilldown>

Examples

Pass a value to a form

<table>
  <search>index=_internal</search>
  <!-- Pass the clicked row's 'count'-column value -->
  <!-- to populate a destination form's 'foo' token. -->
  <drilldown>
    <link>
      /app/search/simple_xml_form?form.foo=$row.count$
    </link>
  </drilldown>
</table>

Pass parameters to a form

<chart>
  <search>
    index=_internal | chart count by sourcetype
  </search>
  <option name="charting.chart">column</option>
  <![CDATA[
    <![CDATA[
      /app/search/simple_xml_form?form.foo=$click.value2$&earliest=$earliest$&latest=$latest$
    ]]]>
  </link>
</drilldown>
</chart>

Pass a value from a chart to a website

<chart>
  <search>
    index=_internal | chart count by sourcetype
  </search>
  <option name="charting.chart">column</option>
  <![CDATA[
    <![CDATA[
      /app/search/simple_xml_form?form.foo=$click.value2$&earliest=$earliest$&latest=$latest$
    ]]]>
  </link>
</drilldown>
</chart>
condition (drilldown)

Define conditional responses to user clicks on specific fields in the dashboard or form. If the `<condition>` element is not present, drilldown actions apply to clicks on all fields.

See `<condition>` (input) for details on using a `<condition>` in a form input.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>text</td>
<td>*</td>
<td>Specifies the search field on which to implement the drilldown, or to set or unset a token.</td>
</tr>
</tbody>
</table>

Child elements

See eval, link, set, and unset.

Example

See the example for `<set>` for using the `<condition>` tag to set a token for in-page drilldown.

eval, link, set, and unset

Managing token values

Update token values to create dynamic display and behavior in dashboards and forms. Use eval, link, set, and unset with `<drilldown>` or `<change>` elements to define the response to user interactions.

Element hierarchy

```
<input>
  <change>
    ( <set> | <unset> | <link> | <eval> ) (1..n)
      <condition> (0..n)
      ( <set> | <unset> | <link> | <eval> ) (1..n)
  </change>
</input>

<drilldown>
  ( <set> | <unset> | <link> | <eval> ) (1..n)
    <condition> (0..n)
      ( <set> | <unset> | <link> | <eval> ) (1..n)
  </drilldown>
```

eval

Filter or format token values. See Custom logic for dashboard eval expressions for more information.
Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>token</td>
<td>text</td>
<td>None</td>
<td>Token whose value is generated by the <code>&lt;eval&gt;</code> expression.</td>
</tr>
</tbody>
</table>

In an `<eval>` expression, you can use either `$...$` delimiters or single quote delimiters for tokens. For example, both of the following options are valid:

- `$my_token$
- `'my_token'`

Example

This example uses `<eval>` with the `<progress>` search event handler to compute and display job duration in the dashboard.

```xml
<dashboard stylesheet="eval_tokens.css">
  <label>Eval Tokens</label>
  <row>
    <panel>
      <title search="search_logic">
        <query>index=_internal | top sourcetype</query>
        <earliest>0</earliest>
        <latest>now</latest>
        <progress>
          <eval token="duration">tostring tonumber($job.runDuration$),"duration")</eval>
        </progress>
      </title>
    </panel>
  </row>
</dashboard>
```

![Dashboard screenshot](image)
### Link

Link to a dashboard, form, or external website in response to a drilldown click or form input.

### Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>target</td>
<td>Text</td>
<td></td>
<td>Indicate the browser window where the drilldown target opens. Corresponds to the target attribute of the <code>&lt;a&gt;</code> HTTP tag.</td>
</tr>
</tbody>
</table>

- `_blank`: Open the target in a new window.
- `_self`: Open the target in the current window.
- `<optional_string>`: Use an arbitrary string to open the target in a new window. Each subsequent time that a user click invokes this drilldown, the target opens in the same window.

### Deprecated attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>Alternatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>field</td>
<td>Field name</td>
<td><code>&lt;drilldown&gt;</code> only Specifies which values to capture in a table from the specified column or row. Cannot be specified together with the series attribute.</td>
<td>Specify fields with the <code>&lt;condition&gt;</code> tag.</td>
</tr>
<tr>
<td>series</td>
<td>Series name</td>
<td><code>&lt;drilldown&gt;</code> only Specifies which values to capture in a chart from the specified series. Cannot be specified together with the field attribute.</td>
<td>Specify series with the <code>&lt;condition&gt;</code> tag.</td>
</tr>
</tbody>
</table>

### Link path syntax

The `<link>` element contains a path to the target and any token values that you are passing from the source to the target. Use one of the following syntax options.

<table>
<thead>
<tr>
<th>Target and behavior</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Link to a dashboard in your Splunk deployment.</td>
<td>Use a relative path that includes the dashboard or form id.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;link&gt;</code></td>
</tr>
<tr>
<td></td>
<td>[relative path]/[dashboard or form id]</td>
</tr>
<tr>
<td></td>
<td>&lt;/link&gt;</td>
</tr>
<tr>
<td>Link to a form in your Splunk deployment. Show customized content in the form by passing a token value captured from the source. Use the token value to populate a form input.</td>
<td>Add a ? symbol after the relative path. Set tokens in the target to values passed from the source. This example sets a token in a target form to a value from the source. Prefix tokens in the target form with form., as shown here.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;link&gt;</code></td>
</tr>
<tr>
<td></td>
<td>[relative path]/[dashboard or form id]?form.[target_token_name]=[$source_value$]</td>
</tr>
<tr>
<td></td>
<td>&lt;/link&gt;</td>
</tr>
<tr>
<td>Pass the <code>&lt;earliest&gt;</code> and <code>&lt;latest&gt;</code> time range modifiers from the source search to a search in the target.</td>
<td>Add <code>&amp;earliest=$earliest$&amp;latest=$latest$</code> to the target path and token values. Use the <code>&lt;![CDATA[ ... ]]&gt;</code> wrapper to make sure that the &amp; symbol is interpreted correctly.</td>
</tr>
<tr>
<td></td>
<td><code>&lt;link&gt;</code></td>
</tr>
<tr>
<td></td>
<td>&lt;![CDATA[[relative path]/[dashboard or form]]]&gt;</td>
</tr>
</tbody>
</table>
Target and behavior | Syntax
---|---
Use a URL and query argument to pass a value to a target web page. | `<link>[target_URL]?q={$source_value$}`

See also Token usage in dashboards to review available token filters.

**Example**

Use `<link>` with conditional inputs to open a new page.

```
<form>
  . . .
  <fieldset>
    <input type="dropdown" token="openNewPageToken">
      <label></label>
      <default> Select a page to open </default>
      <choice value=""> Select a page to open </choice>
      <choice value="manager_page"> Buttercup Games dashboard </choice>
      <choice value="splk_page"> Splunk home page </choice>
    <change>
      <condition value="manager_page">
        <link target="_blank">
          /app/search/buttercup_games
        </link>
      </condition>
      <condition value="splk_page">
        <link target="_blank">
          http://splunk.com
        </link>
      </condition>
    </change>
  </input>
  . . .
</fieldset>
</form>
```

**set**

Set or update token values that other dashboard or form elements can consume. You can set tokens as part of creating dynamic behavior or display, such as passing a user selected value from a form input to a search or managing panel hide/show behavior.

**Element hierarchy**

**Form input**

```
<change>
  <condition> (optional)
    <set>
```
Drilldown

<drilldown>
    <condition> (optional)
    <set>

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>token</td>
<td>Token name</td>
<td>Required. Use the token whose value you are setting or updating.</td>
<td>&lt;set token=&quot;[token_name]&quot;&gt;sourcetype=$click.value</td>
</tr>
<tr>
<td>prefix</td>
<td>text</td>
<td>String to add to the beginning of the token value.</td>
<td>&lt;set token=&quot;[token_name]&quot;&gt;prefix=&quot;sourcetype=&quot;&quot; suffix=&quot;&quot;&quot;&gt;$click.value$&lt;/set&gt;</td>
</tr>
<tr>
<td>suffix</td>
<td>text</td>
<td>String to append to the value of the token.</td>
<td>&lt;set token=&quot;[token_name]&quot;&gt;prefix=&quot;sourcetype=&quot;&quot; suffix=&quot;&quot;&quot;&gt;$click.value$&lt;/set&gt;</td>
</tr>
</tbody>
</table>

Example

A click on the table sets a token which is consumed by the search of the chart visualization.

<dashboard>
    <label>In-page Drilldown</label>
    <row>
        <panel>
            <table>
                <title>Set sourcetype token on click</title>
                <search>
                    <query>
                        index=_internal | stats count by sourcetype
                    </query>
                    <earliest>-1h</earliest>
                    <latest>now</latest>
                </search>
                <drilldown>
                    <condition field="sourcetype"/>
                    <set token="sourcetype">$click.value2$</set>
                </drilldown>
            </table>
            <chart>
                <title>Chart for $sourcetype$</title>
                <search>
                    <query>
                        index=_internal sourcetype=$sourcetype$ | timechart count by sourcetype
                    </query>
                    <earliest>-1h</earliest>
                    <latest>now</latest>
                </search>
            </chart>
        </panel>
    </row>
</dashboard>
unset

Use `<unset>` to clear a previously set token value. When you unset a token, its value is empty until it is set again. Unsetting token values can help you with implementing dynamic display or other behavior in dashboards and forms.

Attributes

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>token</td>
<td>Token name</td>
<td>Required. Use the token whose value you are unsetting.</td>
</tr>
</tbody>
</table>

Example

Use `<set>` and `<unset>` with `depends` and `rejects` in dashboard panels to change the visualization that appears on different user clicks.

```xml
<dashboard>
  <label>Example for <set> and <unset></label>
  <row>
    <panel>
      <table>
        <title>Set sourcetype token</title>
        <search>
          <query>
            index=_internal | stats count by sourcetype
          </query>
          <earliest>-1h</earliest>
          <latest>now</latest>
        </search>
        <drilldown>
          <!-- For the sourcetype field clicked: -->
          <!-- Set token to display a chart -->
          <!-- Unset token to display a table -->
          <condition field="sourcetype">
            <set token="sourcetype">$row.sourcetype$</set>
            <set token="showChart">foo</set>
            <unset token="showTable"></unset>
          </condition>
          <!-- For any other field clicked: -->
          <!-- Set token to display a table -->
          <!-- Unset token to display a chart -->
          <condition field="*">
            <set token="sourcetype">$row.sourcetype$</set>
            <set token="showTable">foo</set>
            <unset token="showChart"></unset>
          </condition>
        </drilldown>
      </table>
    </panel>
    <!-- Hide the html panel when either token is present -->
    <!-- Click in the original table to set either token -->
    <panel>
      <html rejects="$showTable$, $showChart$">
        <h2>Details</h2>
      </html>
    </panel>
  </row>
</dashboard>
```
Predefined drilldown tokens

Predefined tokens capture information when a user clicks different visualization elements. Depending on the visualization type, you can use different tokens to capture a click location or related data.

**chart**

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$click.name$</td>
<td>X-axis field or category name for the clicked location. Not available if the user clicks the chart legend.</td>
</tr>
<tr>
<td>$click.value$</td>
<td>X-axis field or category value for the clicked location. Not available if the user clicks the chart legend.</td>
</tr>
<tr>
<td>$click.name2$</td>
<td>Y-axis field or series name for the clicked location. Not available if the user clicks the chart legend.</td>
</tr>
<tr>
<td>$click.value2$</td>
<td>Y-axis field or series value for the clicked location. Not available if the user clicks the chart legend.</td>
</tr>
<tr>
<td>$row.&lt;fieldname&gt;$</td>
<td>Access any y-axis field value corresponding to the clicked location x-axis. Not available if the user clicks the chart legend.</td>
</tr>
<tr>
<td>$row.&lt;x-axis-name&gt;$</td>
<td>Access any x-axis field value corresponding to the clicked location. Not available if the user clicks the chart legend.</td>
</tr>
<tr>
<td>Token</td>
<td>Description</td>
</tr>
<tr>
<td>------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>$earliest$</td>
<td>Earliest time for the clicked chart segment. If not applicable, uses the earliest time for the search.</td>
</tr>
<tr>
<td>$latest$</td>
<td>Latest time for the clicked chart segment. If not applicable, uses the latest time for the search.</td>
</tr>
</tbody>
</table>

**event**

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$click.name$</td>
<td>Field name for the clicked element in the event list. If a field name is not available for the clicked location, the $click.name$ value defaults as follows.</td>
</tr>
<tr>
<td></td>
<td>- Clicking an element in the event: _raw</td>
</tr>
<tr>
<td></td>
<td>- Clicking the event timestamp: _time</td>
</tr>
<tr>
<td></td>
<td>- Clicking a tag: uses the tag name. For example, if host is tagged, uses host.</td>
</tr>
<tr>
<td>$click.value$</td>
<td>Field value for the clicked element in the event list.</td>
</tr>
<tr>
<td>$click.name2$</td>
<td>Identical to $click.name$.</td>
</tr>
<tr>
<td>$click.value2$</td>
<td>Identical to $click.value$.</td>
</tr>
<tr>
<td>$row.&lt;fieldname&gt;$</td>
<td>Access any field value in the clicked event. For example, to access the host field value, use $row.host$.</td>
</tr>
<tr>
<td>$earliest$</td>
<td>Earliest time for the clicked event. Equivalent to the _time field value. Defaults to the earliest search time.</td>
</tr>
<tr>
<td>$latest$</td>
<td>Latest time for the clicked event. Equivalent to one second after the _time field value. Defaults to the latest search time.</td>
</tr>
</tbody>
</table>

**map**

Predefined tokens are available for cluster and Choropleth maps. Some tokens are only available in cluster maps.

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$click.name$</td>
<td>Field name for the clicked location. If multiple fields are associated with the location, uses the first field.</td>
</tr>
<tr>
<td>$click.value$</td>
<td>Field value for the clicked location. If multiple fields are associated with the location, uses the first field.</td>
</tr>
<tr>
<td>$click.name2$</td>
<td>Same as $click.name$.</td>
</tr>
<tr>
<td>$click.value2$</td>
<td>Same as $click.value$.</td>
</tr>
<tr>
<td>$row.&lt;fieldname&gt;$</td>
<td>Access field values related to the clicked location. Check the Statistics tab for available fields.</td>
</tr>
<tr>
<td>$earliest$</td>
<td>Earliest time for the search generating the map.</td>
</tr>
<tr>
<td>$latest$</td>
<td>Latest time for the search generating the map.</td>
</tr>
<tr>
<td>$click.lat.name$</td>
<td>For cluster maps: latitude field name for the clicked location.</td>
</tr>
<tr>
<td>$click.lat.value$</td>
<td>For cluster maps: latitude field value for the clicked location.</td>
</tr>
<tr>
<td>$click.lon.name$</td>
<td>For cluster maps: longitude field name for the clicked location.</td>
</tr>
<tr>
<td>$click.lon.value$</td>
<td>For cluster maps: longitude field value for the clicked location.</td>
</tr>
<tr>
<td>$click.bounds.&lt;orientation&gt;$</td>
<td>For cluster maps: south, west, north, or east outer boundary for the clicked location. For example, use $click.bounds.east$ to get the eastern outer boundary.</td>
</tr>
</tbody>
</table>
### single value

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$click.name$</td>
<td>Name of the field that the single value represents.</td>
</tr>
<tr>
<td></td>
<td><strong>In a <code>&lt;condition&gt;</code> element configuring drilldown for a single value,</strong> field corresponds to $click.name$.</td>
</tr>
<tr>
<td>$click.value$</td>
<td>Field value that the single value represents.</td>
</tr>
<tr>
<td>$click.name2$</td>
<td>Same as $click.name$.</td>
</tr>
<tr>
<td>$click.value2$</td>
<td>Same as $click.value$.</td>
</tr>
<tr>
<td>$row.&lt;fieldname&gt;$</td>
<td>Access any field value from the Statistics table row for the single value.</td>
</tr>
<tr>
<td>$earliest$</td>
<td>Earliest time for the search that generates the single value.</td>
</tr>
<tr>
<td>$latest$</td>
<td>Latest time for the search that generates the single value.</td>
</tr>
</tbody>
</table>

### table

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$click.name$</td>
<td>Leftmost field (column) name in the table. Typically, <code>_time</code> is leftmost in a table that includes this field and where columns are ordered by default.</td>
</tr>
<tr>
<td>$click.value$</td>
<td>Leftmost field (column) value in the clicked table row.</td>
</tr>
<tr>
<td>$click.name2$</td>
<td>Clicked field (column) name. In a <code>&lt;condition&gt;</code> element configuring drilldown in a table, field corresponds to $click.name2$.</td>
</tr>
<tr>
<td>$click.value2$</td>
<td>Clicked field (column) value. Use this token to capture the specific table cell value that users click.</td>
</tr>
<tr>
<td>$row.&lt;fieldname&gt;$</td>
<td>Access any field (column) value from the clicked table row.</td>
</tr>
<tr>
<td>$earliest$</td>
<td>Earliest time for the clicked table row. If not applicable, uses the earliest time of the search.</td>
</tr>
<tr>
<td>$latest$</td>
<td>Latest time for the clicked table row. If not applicable, uses the latest time of the search.</td>
</tr>
</tbody>
</table>

### trellis

<table>
<thead>
<tr>
<th>Token</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$trellis.name$</td>
<td>Split field name</td>
</tr>
<tr>
<td>$trellis.value$</td>
<td>Split field value</td>
</tr>
</tbody>
</table>

### Deprecations and removals

Check the Deprecated features list in the *Release Notes* for additional information on deprecated or removed elements.