

Events API

Edit New Page Jump to bottom

lisongye edited this page on Sep 27, 2016 · 9 revisions

The Sigma Event dispatcher

Multiple objects in Sigma can dispatch events, such as node and edge clicking and hovering. In addition to providing default behavior for some of these actions, they also implement sigma.classes.dispatcher to provide event handler bindings to the user. This interface exposes the following methods:

- bind(events, handler):
 - Binds the function handler to the event or events given, with multiple events separated by spaces. The handler takes a data object whose contents depend on the event being dispatched.
 - o Returns the dispatcher.
- unbind(events, ?handler):
 - Unbinds the function handler from the event or events given, with multiple events separated by spaces. If handler is undefined, all handlers are unbound.
 - o Returns the dispatcher.
- dispatchEvent (events, ?data):
 - Calls all of the handlers bound to the event given (or multiple events separated by spaces) with the given data object as the argument. If data is not provided, an empty object is used.
 - Returns the dispatcher.
- getEvent (event, data):
 - Returns an event object with the given event type set as type, the given data set as data, and the target equal to the dispatcher.
- extend (target, args):
 - Causes the given target object to implement dispatcher.

Sigma renderer events

Sigma renderers dispatch the following events with the provided arguments in the data object. Unless otherwise noted, each data object also contains a property called captor that contains the raw event data forwarded from the captor (described below).

Event	Description
click	Re-dispatches the raw click event (no captor property).
clickNode	Dispatches once for each node clicked, with the node in node.
clickNodes	Dispatches once for each click, with all clicked nodes contained in nodes.
clickEdge	Dispatches once for each edge clicked, with the edge in edge.
clickEdges	Dispatches once for each click, with all clicked edges contained in edges.
clickStage	Dispatches when the background is clicked.
doubleClick	Re-dispatches the raw double click event (no captor property).
doubleClickNode	Dispatches once for each node double clicked, with the node in node.
doubleClickNodes	Dispatches once for each double click, with all clicked nodes contained in nodes.
doubleClickEdge	Dispatches once for each edge double clicked, with the edge in edge.
doubleClickEdges	Dispatches once for each double click, with all clicked edges contained in edges .
doubleClickStage	Dispatches when the background is double clicked.
rightClick	Re-dispatches the raw right click event (no captor property).
rightClickNode	Dispatches once for each node right clicked, with the node in node.
rightClickNodes	Dispatches once for each right click, with all clicked nodes contained in nodes.
rightClickEdge	Dispatches once for each edge right clicked, with the edge in edge.
rightClickEdges	Dispatches once for each right click, with all clicked edges contained in edges.
rightClickStage	Dispatches when the background is right clicked.
overNode	Dispatches when the mouse moves over a node, with the node in node.
overNodes	Dispatches when the mouse moves over a set of nodes, with the nodes in nodes.
overEdge	Dispatches when the mouse moves over an edge, with the edge in edge.
overEdges	Dispatches when the mouse moves over a set of edges, with the edges in edges.
outNode	Dispatches when the mouse moves out of a node, with the node in node.
outNodes	Dispatches when the mouse moves out of a set of nodes, with the nodes in nodes.
outEdge	Dispatches when the mouse moves out of an edge, with the edge in edge.
outEdges	Dispatches when the mouse moves out of a set of edges, with the edges in edges.

Renderers also dispatch a "render" event with no data each time they finish redrawing.

Event binding for these events can either be applied to the entire Sigma instance (which connects it to all renderers):

```
sigInst.bind("clickNode", function () { ... });
```

or to the individual renderer:

```
sigInst.renderers[0].bind("clickNode", function () { ... });
```

Sigma does not fire manipulation events such as panning and zooming - the captors handle this automatically.

Sigma top-level events

In addition to the render event, the sigma instance itself also fires a few general-information events:

Event	Description
kill	Fires when the instance is killed.

Sigma camera events

Sigma cameras fire one event:

Event	Description
coordinatesUpdated	Fires when the camera has moved via <code>goTo()</code> . No data is provided - simply read the properties of the camera to get the new values.

Sigma mouse events

Sigma captors dispatch the following raw mouse events, with data found in the captor property of most top-level events:

- mousemove
- mousedown
- mouseup
- mouseout
- click
- doubleclick
- rightclick

Each event, except for mouseout, will provide a data object with the same event information as the underlying mouse events, including the clientX, clientY, ctrlKey, metaKey, altKey, and shiftKey properties from the DOM Level 2 specification, and also adding cross-browser standardized x and y properties.

Sigma does not redispatch mouse wheel or touch events - the captor objects handle the behavior for those internally. If you wish to change this or any other behavior, you must define a custom captor.

Writing a Custom Captor

The default captors (mouse and touch) should suit most purposes, and Sigma is designed to primarily use those. However, if you wish to support other events (such as network events or platform-specific gesture APIs), you can write a custom captor.

Captor constructors have the following signature:

```
new captor_constructor(target, camera, settings)
```

taking the following parameters:

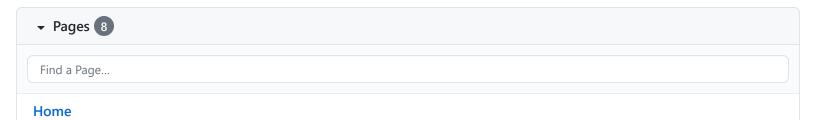
- target: the DOM element that the captor is expected to attach events to
- camera: the Sigma camera for the renderer context
- settings: the Sigma settings function for the captor

To use a custom captor, add its constructor to the configuration object in the property captors:

```
sigInst.addRenderer({
  container: document.getElementById("graph-container"),
  captors: [my_captor_constructor]
});
```

Sigma captors do not implement any methods or properties inherently, but are expected to implement sigma.classes.dispatcher and should dispatch the events mentioned above with the appropriate data. Captors also need to modify the camera for manipulation events such as panning and zooming - they need not forward this to the user.

+ Add a custom footer



Camera API
Events API
Graph API
Public API
Renderers
Settings
Settings: How Settings Work in Sigma
+ Add a custom sidebar

Clone this wiki locally

https://github.com/jacomyal/sigma.js.wiki.git