New issue Jump to bottom

How to detect when a node is dragged? #297

Closed **sheymann** opened this issue on Jun 12, 2014 · 9 comments

sheymann commented on Jun 12, 2014 Contributor

I'm using the dragNodes plugin and I'd like to detect when a node is currently dragged to avoid firing other events such as overNode and outNode.

Any idea how to do it without a dirty hack? (-> comparing x,y at overNode and at outNode events)

Thanks

apitts commented on Jun 13, 2014 Contributor

@sheymann - **@patrickmarabeas** and myself have been looking at some of the issues that you run into with the dragNodes plug-in. You may consider this a 'dirty hack' but the best solution we have come up with so far to the problem you refer to above is to add the below code (we are using angular obviously but simple enough to replace with jquery, etc.):

```
var drag = false;
angular.element(document).bind('mousedown', function(e) {
  drag = false;
  var x = e.clientX;
  var y = e.clientY;
  angular.element(document).bind('mousemove', function(e) {
    if(Math.abs(e.clientX - x) || Math.abs(e.clientY - y) > 1) drag = true;
  });
});
angular.element(document).bind('mouseup', function() {
  angular.element(document).unbind('mousemove');
  angular.element('body').removeClass('disableSelect');
});
```

We previously had it working without the distance metric (i.e. just on mousemove) but found we needed to add that distance metric after some recent changes to sigma.

Then, in overNode, outNode, etc. we include:

```
    s.bind('overNode', function(e) {
      if(!drag) {
      ---insert code-----
      }
```

sheymann commented on Jun 13, 2014 Contributor Author

Thank you! I've adapted your code to remove dependencies and to provide an easy integration into sigma. Do you think that it would be interesting to submit it as a plugin?

```
;(function(undefined) {
  'use strict';

  if (typeof sigma === 'undefined') {
    throw 'sigma is not declared';
  }

  sigma.utils.pkg('sigma.events');

  /**
   * Dispatch 'drag' and 'drop' events by dealing with mouse events.
   *
   * @param {object} renderer The renderer to listen.
   */
  sigma.events.drag = function(renderer) {
    sigma.classes.dispatcher.extend(this);

    var _self = this,
        _drag = false,
        _x = 0,
        _y = 0;

    // Set _drag to true if the mouse position has changed.
    var detectDrag = function(e) {
      //console.log('mousemove');
      if(Math.abs(e.clientX - _x) || Math.abs(e.clientY - _y) > 1) {
        _drag = true;
        console.log('drag');
        _self.dispatchEvent('drag');
      }
    };

    // Initialize the mouse position and attach the 'mousemove' event
    // to detect dragging.
    renderer.container.addEventListener('mousedown', function(e) {
      console.log('mousedown');
      _drag = false;
      _x = e.clientX;
      _y = e.clientY;
      renderer.container.addEventListener('mousemove', detectDrag);
    });

    renderer.container.addEventListener('mouseup', function() {
        // 'mouseup' event is called at the end of the call stack
        // so that 'mousemove' is called before.
        setTimeout(function() {
          console.log('mouseup');
          if (_drag) {
            _self.dispatchEvent('drop');
          }
          _drag = false;
          renderer.container.removeEventListener('mousemove', detectDrag);
        }, 1);
      });
    };
  }).call(this);
```

To use it:

```
var drag = false;

var _dragListener = new sigma.events.drag(sigma_instance.renderers[0]);
_dragListener.bind('drag', function(e) {
    drag = true;
});
_dragListener.bind('drop', function(e) {
    drag = false;
});
```

apitts commented on Jun 13, 2014 Contributor

That looks great **@sheymann**! I for one would certainly welcome a pull request for that plugin.

sheymann commented on Jun 13, 2014 Contributor Author

Submitted! #301

patrickmarabeas commented on Jul 6, 2014 Contributor

@sheymann The mouse event listeners need to be placed on `window` rather than `renderer.container` so you don't run into issues dropping nodes behind elements (like UI).

sheymann commented on Jul 6, 2014 Contributor Author

@patrickmarabeas I don't understand the issue, can you provide a screenshot?

Event listeners should be attached to renderer containers following the sigma event model.

patrickmarabeas commented on Jul 6, 2014 Contributor

Which requires the mouse to be lifted within the render container. If I mouseup, irregardless of where this occurs - on top of a UI element that rests above the canvas, or away from the browser window all together, I expect the drag event to be concluded.

sheymann commented on Jul 6, 2014 Contributor Author

I disagree because the drag target matters, and this plugin only makes sense when we stop dragging inside the renderer, i.e. a node is moved from one position to another in the 2d space.

patrickmarabeas commented on Jul 7, 2014 Contributor

OK, well - the event still needs to occur if the drop event occurs on a UI element blocking the canvas... and if these elements need to be interacted with, they can't simply be ignored with use of the CSS `pointer-events` property for instance...

sheymann closed this on Sep 17, 2014

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked pull requests

Successfully merging a pull request may close this issue.

None yet

3 participants