Workshop: Advanced JSXGraph

Vol. 1

Alfred Wassermann

UNIVERSITÄT BAYREUTH

12-11-2020

Contents

Preliminaries

Include JSXGraph

- JSXGraph skeleton page:

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>JSXGraph template</title>
    <meta content="text/html; charset=utf-8" http-equiv="Content-Type">
    <link href="https://cdn.jsdelivr.net/npm/jsxgraph@1.1.0/distrib/
        jsxgraph.css" rel="stylesheet" type="text/css" />
    <script src="https://cdn.jsdelivr.net/npm/jsxgraph@1.1.0/distrib/
        jsxgraphcore.js" type="text/javascript" charset="UTF-8"></script>

    <!-- The next line is optional: MathJax -->
    <script src="https://cdn.jsdelivr.net/npm/mathjax@3/es5/tex-chtml.js"
        id="MathJax-script" async></script>
  </head>
  <body>

  <div id="jxgbox" class="jxgbox" style="width:500px; height:200px;"></div
      >

  <script>
    var board = JXG.JSXGraph.initBoard('jxgbox', {boundingbox: [-5, 2, 5,
        -2]});
  </script>

  </body>
</html>
```

- See JSXGraph handbook (in development): https://ipesek.github.io/jsxgraphbook/

Images

JSXGraph image element

```
const board = JXG.JSXGraph.initBoard('jxgbox', {
    boundingbox: [-5, 5, 5, -5], axis:true
});

var im = board.create('image', ['https://jsxgraph.org/distrib/images/
   uccellino.jpg', [-3, -3], [2,2]]);
```

- Any URL can be included
- Uses HTML image element: all image formats supprted by the browser can be used.
- Example at https://jsfiddle.net/yve4L7fr/1/
- Attention: some images from third party servers may be blocked because of CORS, see https://developer.mozilla.org/en-US/docs/Web/HTML/CORS_enabled_image

Use data URIs

- There are multiple online services to create dataURIs
- Linux bash script to create data URI `data_uri.sh`:

```
#!/bin/sh
mimetype=$(file -bN --mime-type "$1")
content=$(base64 -w0 < "$1")
# --break0 on Mac
echo "url('data:$mimetype;base64,$content')"
```

- Example: https://jsfiddle.net/pk6ynb2d/
- Pros:
  - Construction is self-contained
  - Faster for small images
- Cons:
  - Source code hard to maintain

Animations

Animations of JSXGraph point elements can be realized with the methods `moveTo`, `visit` and `moveAlong`.

1) `moveTo([x,y], msec, [options])`

- Options:

```
    {
        effect: `<>`, `--`
        callback: function
    }
```

- If the time is omitted it is set to 0.

- Examples:
  - https://jsfiddle.net/x167uwnv/1/
  - https://jsxgraph.org/wiki/index.php/Animation_II
  - Random moves: https://jsxgraph.org/wiki/index.php/Animation_III
  -

2) `visit([x,y], msec, [options])`

- Options:

```
{
    repeat: Number,
    effect: `<>`, `--`
    callback: function
}
```

- Examples:
  - https://jsxgraph.org/wiki/index.php/Animations

3) `moveAlong(points, msec, options)`

- Options:

```
{
    effect: `<>`, `--`
    callback: function,
    interpolate: Boolean
}
```

- Example: https://jsfiddle.net/8saok9hr/2/
- Interpolation path: see https://jsxgraph.org/wiki/index.php/Interpolation:_Neville%27s_algorithm, compare also lagrange interpolation in https://jsxgraph.org/wiki/index.php/Lagrange_interpolation.

4) Of course, also the JavaScript methods `setTimeout`, `setInterval`, `requestAnimationFrame` can be used to craft animations:

- https://jsxgraph.org/wiki/index.php/User:Cleonis/sandbox/requestAnimationFrame
- https://jsxgraph.org/wiki/index.php/User:Cleonis/sandbox/setInterval

Events

In JSXGraph, event handling is similar to that of JavaScript. Of course, “regular” JavaScript events fired by the browser can be catched in JSXGraph, too.

Board events

The JSXGraph `board` element fires the following events:

- `down`, `mousedown`, `pendown`, `pointerdown`, `touchstart`
- `up`, `mouseup`, `pointerup`, `touchend`
- `move`, `mousemove`, `penmove`, `pointermove`, `touchmove`
- `hit`:

```
    var c = board.create('circle', [[1, 1], 2]);
     board.on('hit', function(evt, el) {
        console.log("Hit element", el);
     });
```

- `update`: the board update method has been called
- `boundingbox`: The bounding box of the board has changed
- Events of the selection rectangle (see API docs)
- It is recommended to use the generic events `down`, `move`, `up` instead of the events specific for pointer types. A device that supports pointer events does not trigger touch events, while most of these devices additional trigger mouse events.
- Since iOS 13, also Apple has switched to pointer events. That means **all** major browsers use **pointer events**.
- In the event object, the pointer type is stored (`mouse`, `pen`, `touch`):

```
board.on('down', function(evt) {
    console.log(evt.pointerType);
});
```

- Example: https://jsfiddle.net/zmf4swao/1/

Element events

1) The `down`, `move`, `up` event categories also exist for elements:

```
var p1 = board.create('point', [1,2]);
var p2 = board.create('point', [-1,2]);
```

```
var listen = function(evt) {
    console.log(this.name, this.X(), this.Y());
};
p1.on('down', listen);
p2.on('down', listen);
```

2) Additional events for elements are:

- `drag`, `mousedrag`, `pendrag`, `touchdrag`
- `over`, `mouseover`, `out`, `mouseout`

Example: https://jsfiddle.net/zmf4swao/1/:

```
var p1 = board.create('point', [1,2]);
var p2 = board.create('point', [-1,2]);

var listen = function(evt) {
    console.log(this.name, this.X(), this.Y());
};
p1.on('down', listen);
p2.on('down', listen);

// Keep an element in a certain area:
p1.on('drag', function(evt) {
    var x = this.X();
  if (x < 0) {
    this.moveTo([0, this.Y()]);
  }
});
```

- Using the drag event, points can be restricted to certain areas, similar than `glider` elements are restricted to path elements.

3) Events triggered on attribute changes:

- `attribute`: Notify every time any attribute is changed
- `attribute:[attribute]`: Notify every time a specific attribute is changed, eg. `attribute:strokecolor`, see https://jsxgraph.org/docs/symbols/JXG.GeometryElement.html#event:attribute. *Attention:* the attribute name is given in lower case.

Discussion and suggestion of further topics

- Highlighting off for specific elements?
- Combine a group of elements for highlighting?

- Dragging a group of points: `group`
- How is a construction interpreted? (timeouts?)

Next webinar

> The next webinar will be **Wednesday, December 16th, 2020 at 4 pm CET**