23/06/2025, 09:29 gallery.js

data-vis\gallery.js

```
function Gallery() {
 1
 2
 3
        this.visuals = [];
 4
        this.selectedVisual = null;
 5
        var self = this;
 6
 7
 8
        // Add a new visualisation to the navigation bar.
 9
        this.addVisual = function(vis) {
10
11
          // Check that the vis object has an id and name.
12
          if (!vis.hasOwnProperty('id')
              && !vis.hasOwnProperty('name')) {
13
            alert('Make sure your visualisation has an id and name!');
14
15
          }
16
17
          // Check that the vis object has a unique id.
          if (this.findVisIndex(vis.id) != null) {
18
19
            alert(`Vis '${vis.name}' has a duplicate id: '${vis.id}'`);
20
          }
21
22
          this.visuals.push(vis);
23
24
25
          // Create menu item.
26
          var menuItem = createElement('li', vis.name);
27
          menuItem.addClass('menu-item');
28
          menuItem.id(vis.id);
29
30
          menuItem.mouseOver(function(e)
31
32
              var el = select('#' + e.srcElement.id);
33
              el.addClass("hover");
34
35
          })
36
          menuItem.mouseOut(function(e)
37
38
              var el = select('#' + e.srcElement.id);
39
              el.removeClass("hover");
40
          })
41
42
43
          menuItem.mouseClicked(function(e)
44
          {
45
              //remove selected class from any other menu-items
46
47
              var menuItems = selectAll('.menu-item');
48
              for(var i = 0; i < menuItems.length; i++)</pre>
49
50
                  menuItems[i].removeClass('selected');
```

```
23/06/2025, 09:29
                                                           gallery.js
   52
                  }
   53
   54
                  var el = select('#' + e.srcElement.id);
   55
                  el.addClass('selected');
   56
   57
                  self.selectVisual(e.srcElement.id);
   58
   59
             })
   60
   61
             var visMenu = select('.visuals-menu');
   62
             visMenu.child(menuItem);
   63
   64
             // Preload data if necessary.
   65
             if (vis.hasOwnProperty('preload')) {
   66
               vis.preload();
   67
             }
   68
   69
           };
   70
   71
           this.findVisIndex = function(visId) {
   72
             // Search through the visualisations looking for one with the id
   73
             // matching visId.
   74
             for (var i = 0; i < this.visuals.length; i++) {</pre>
               if (this.visuals[i].id == visId) {
   75
                  return i;
   76
   77
               }
             }
   78
   79
   80
             // Visualisation not found.
             return null;
   81
           };
   82
   83
   84
           this.selectVisual = function(visId){
   85
             var visIndex = this.findVisIndex(visId);
   86
   87
             if (visIndex != null) {
   88
               // If the current visualisation has a deselect method run it.
   89
               if (this.selectedVisual != null
   90
                    && this.selectedVisual.hasOwnProperty('destroy')) {
   91
                 this.selectedVisual.destroy();
               }
   92
               // Select the visualisation in the gallery.
   93
   94
               this.selectedVisual = this.visuals[visIndex];
   95
   96
               // Initialise visualisation if necessary.
   97
               if (this.selectedVisual.hasOwnProperty('setup')) {
   98
                  this.selectedVisual.setup();
   99
               }
  100
  101
               // Enable animation in case it has been paused by the current
  102
               // visualisation.
  103
               loop();
  104
             }
  105
           };
```

localhost:53597/181b8330-52b9-495f-9d9c-e880f6f41fd4/