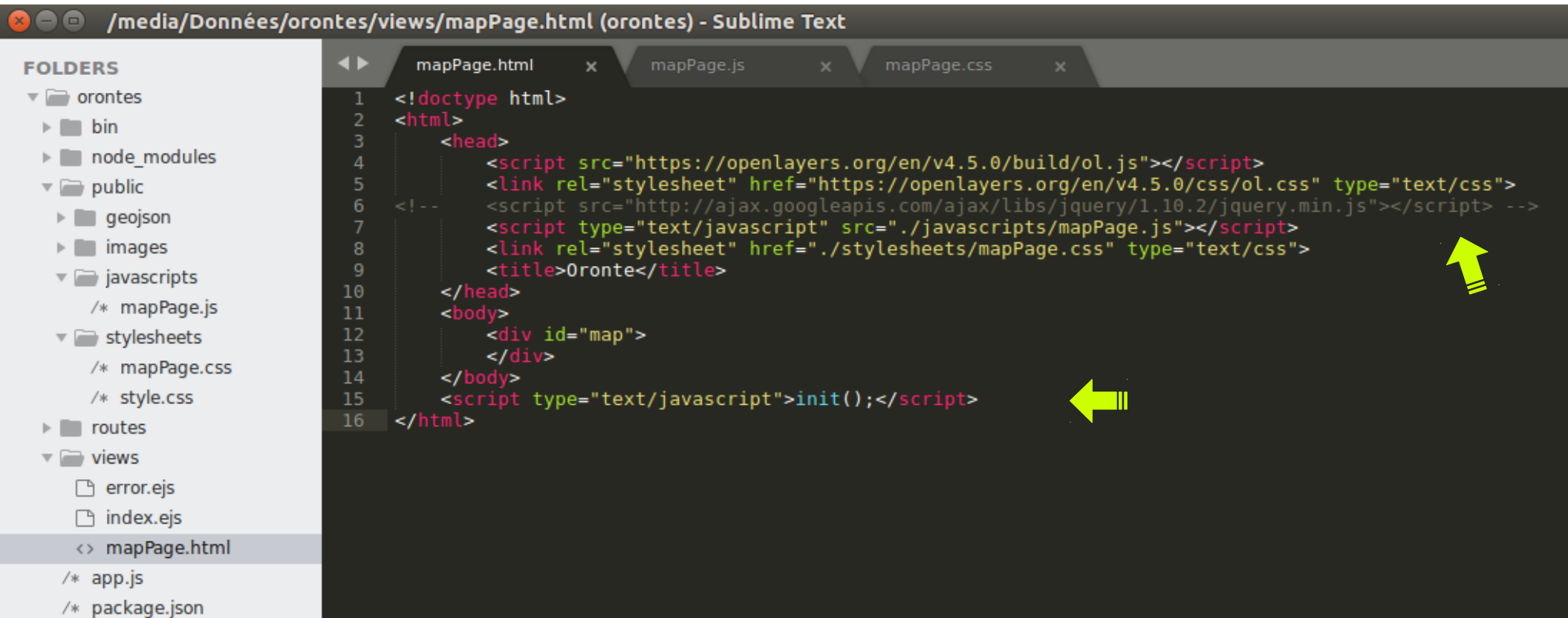
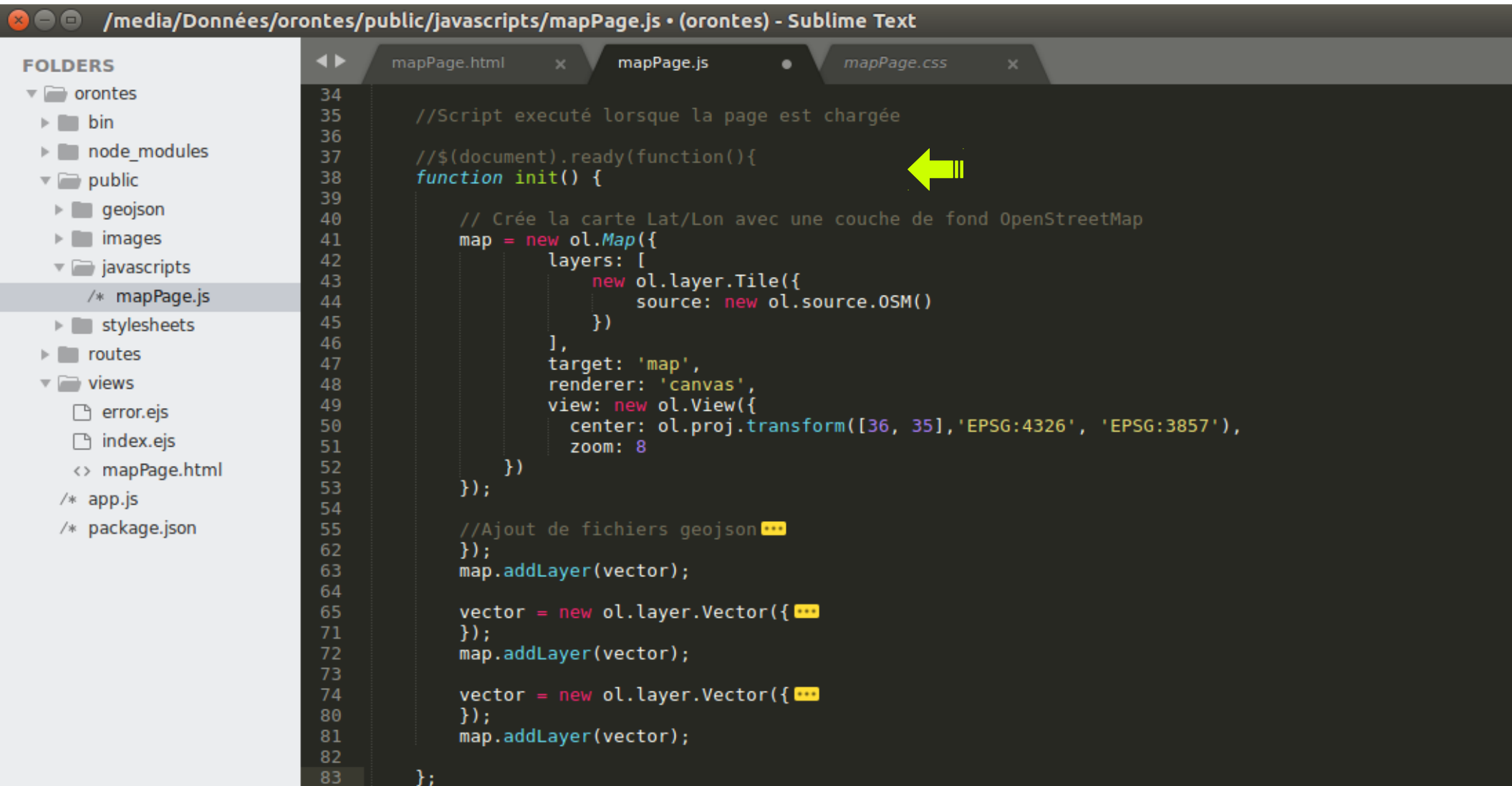


Script init() au lieu de \$(document).ready ...



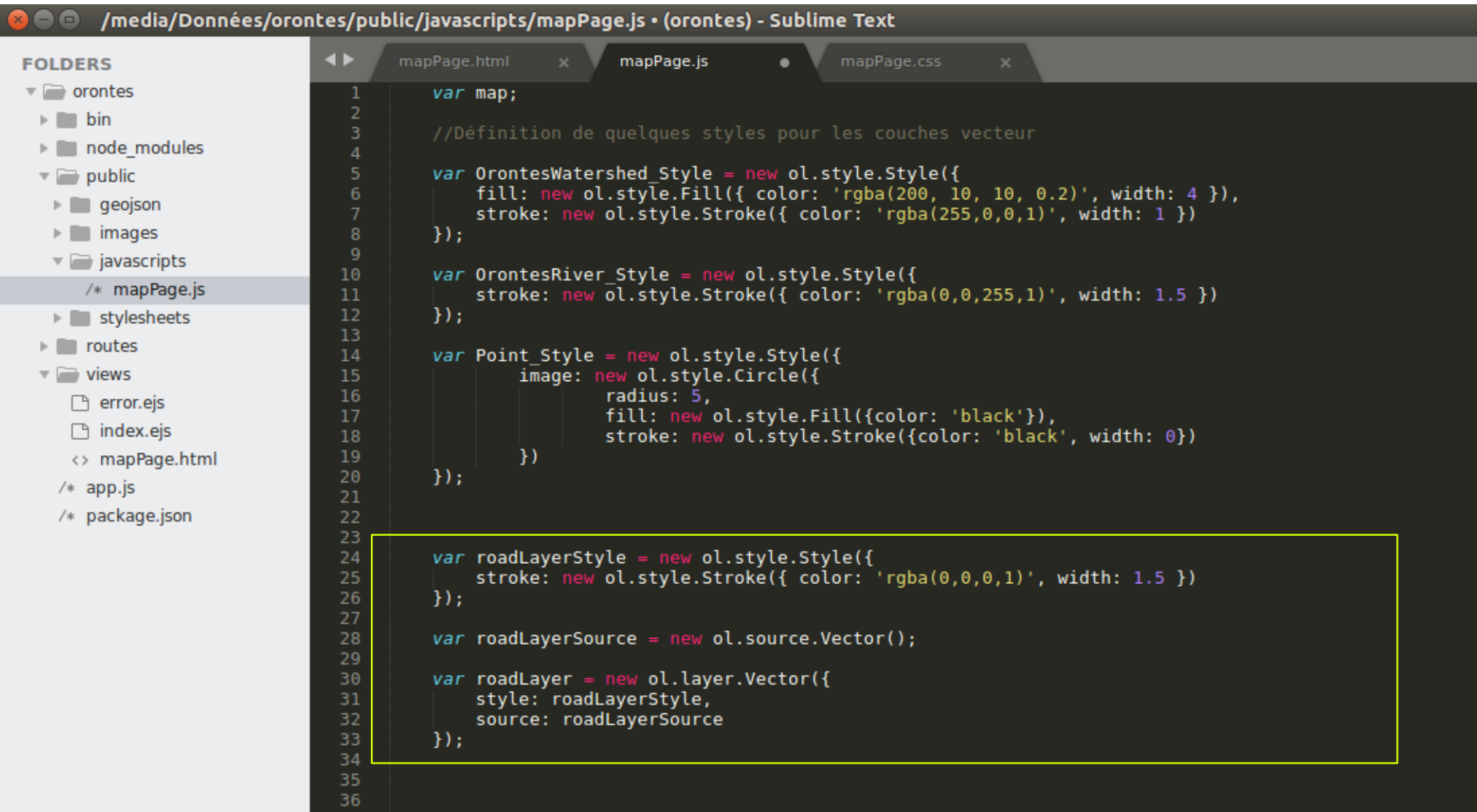
```
1 <!doctype html>
2 <html>
3   <head>
4     <script src="https://openlayers.org/en/v4.5.0/build/ol.js"></script>
5     <link rel="stylesheet" href="https://openlayers.org/en/v4.5.0/css/ol.css" type="text/css">
6     <!-- <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.10.2/jquery.min.js"></script> -->
7     <script type="text/javascript" src="./javascripts/mapPage.js"></script>
8     <link rel="stylesheet" href="./stylesheets/mapPage.css" type="text/css">
9     <title>Oronte</title>
10  </head>
11  <body>
12    <div id="map">
13    </div>
14  </body>
15  <script type="text/javascript">init();</script>
16 </html>
```

Script init() au lieu de \$(document).ready ...



```
34
35 //Script executé lorsque la page est chargée
36
37 //$(document).ready(function(){
38 function init() {
39
40     // Crée la carte Lat/Lon avec une couche de fond OpenStreetMap
41     map = new ol.Map({
42         layers: [
43             new ol.layer.Tile({
44                 source: new ol.source.OSM()
45             })
46         ],
47         target: 'map',
48         renderer: 'canvas',
49         view: new ol.View({
50             center: ol.proj.transform([36, 35], 'EPSG:4326', 'EPSG:3857'),
51             zoom: 8
52         })
53     });
54
55     //Ajout de fichiers geojson ***
56     });
57     map.addLayer(vector);
58
59     vector = new ol.layer.Vector({ ***
60     });
61     map.addLayer(vector);
62
63     vector = new ol.layer.Vector({ ***
64     });
65     map.addLayer(vector);
66
67     };
68 }
```

Définition de la couche des routes, avec un style et une source de données...  
encore indéfinie



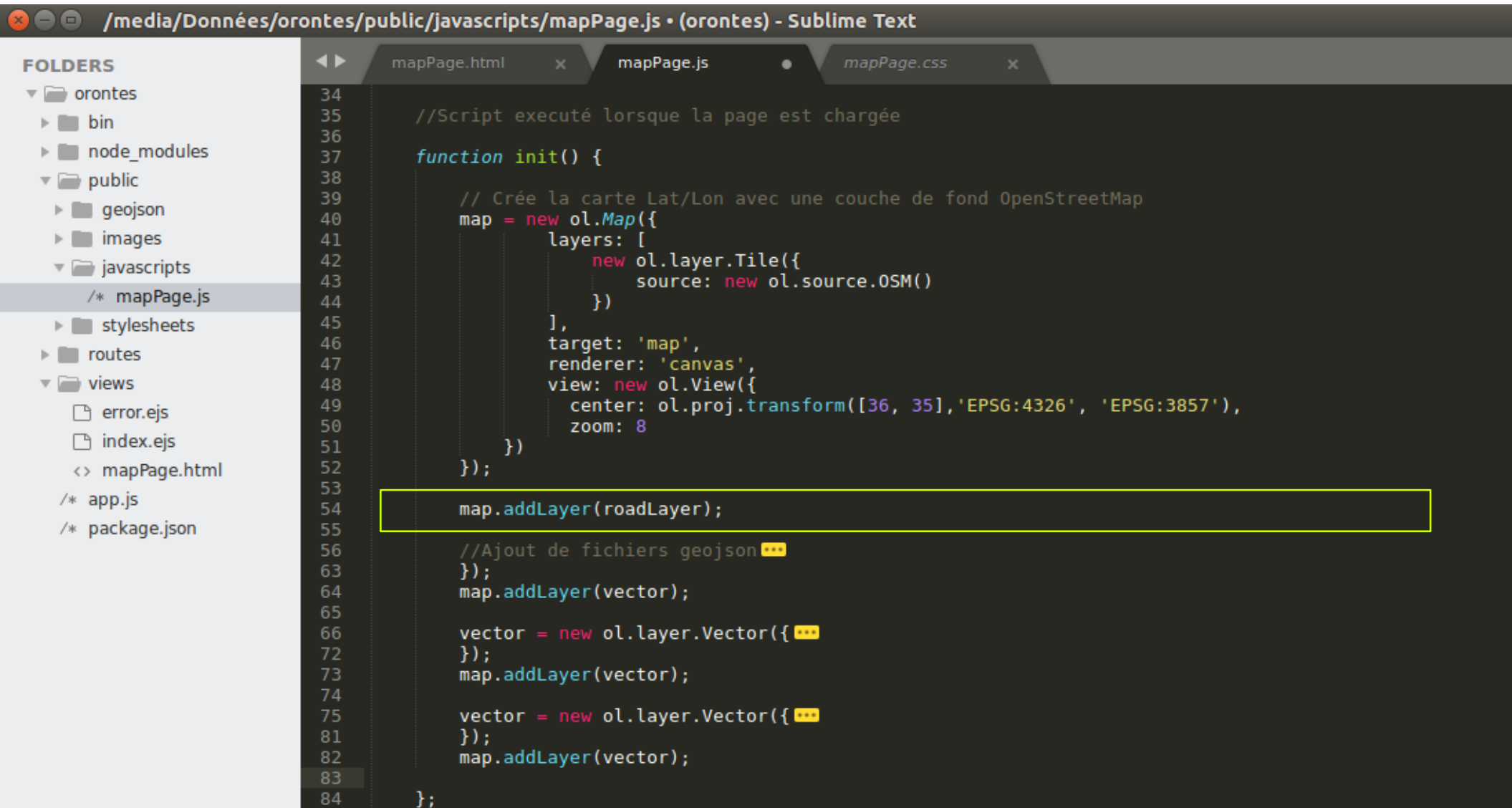
The screenshot shows a Sublime Text editor window titled "/media/Données/orontes/public/javascripts/mapPage.js • (orontes) - Sublime Text". The left sidebar displays a file explorer with the following structure:

- orontes
  - bin
  - node\_modules
  - public
    - geojson
    - images
    - javascripts
      - /\* mapPage.js
    - stylesheets
  - routes
  - views
    - error.ejs
    - index.ejs
    - mapPage.html
  - /\* app.js
  - /\* package.json

The main editor area shows the content of mapPage.js, with line numbers 1 through 36. The code defines several map styles and a layer for roads. The following code block, which defines the road layer, is highlighted with a yellow box:

```
24 var roadLayerStyle = new ol.style.Style({
25   stroke: new ol.style.Stroke({ color: 'rgba(0,0,0,1)', width: 1.5 })
26 });
27
28 var roadLayerSource = new ol.source.Vector();
29
30 var roadLayer = new ol.layer.Vector({
31   style: roadLayerStyle,
32   source: roadLayerSource
33 });
```

## Ajout de la couche des routes à la carte...



The screenshot shows a Sublime Text editor window titled "/media/Données/orontes/public/javascripts/mapPage.js • (orontes) - Sublime Text". The left sidebar displays a file explorer with the following structure:

- orontes
  - bin
  - node\_modules
  - public
    - geojson
    - images
    - javascripts
      - /\* mapPage.js
    - stylesheets
  - routes
  - views
    - error.ejs
    - index.ejs
    - <> mapPage.html
  - /\* app.js
  - /\* package.json

The main editor area shows the content of mapPage.js, with line numbers 34 through 84. The code is as follows:

```
34
35 //Script executé lorsque la page est chargée
36
37 function init() {
38
39     // Crée la carte Lat/Lon avec une couche de fond OpenStreetMap
40     map = new ol.Map({
41         layers: [
42             new ol.layer.Tile({
43                 source: new ol.source.OSM()
44             })
45         ],
46         target: 'map',
47         renderer: 'canvas',
48         view: new ol.View({
49             center: ol.proj.transform([36, 35], 'EPSG:4326', 'EPSG:3857'),
50             zoom: 8
51         })
52     });
53
54     map.addLayer(roadLayer);
55
56     //Ajout de fichiers geojson ***
63     });
64     map.addLayer(vector);
65
66     vector = new ol.layer.Vector({ ***
72     });
73     map.addLayer(vector);
74
75     vector = new ol.layer.Vector({ ***
81     });
82     map.addLayer(vector);
83
84     };
```


A yellow rectangular box highlights the line `map.addLayer(roadLayer);` at line 54.

# Les interactions d'openlayers

OpenLayers v4.5.0 API - Namespace: interaction - Google Chrome

Oronte x OpenLayers v4.5.0 / x

← → ↻ ⓘ openlayers.org/en/latest/apidoc/ol.interaction.html

 **OpenLayers**

Search Documentation

ol.interaction

Methods  
defaults

ol

ol.AssertionError

ol.Attribution

ol.CanvasMap

ol.Collection

ol.Collection.Event

ol.DeviceOrientation

ol.Disposable

ol.Feature

ol.Geolocation

ol.Graticule

ol.Image

ol.ImageBase

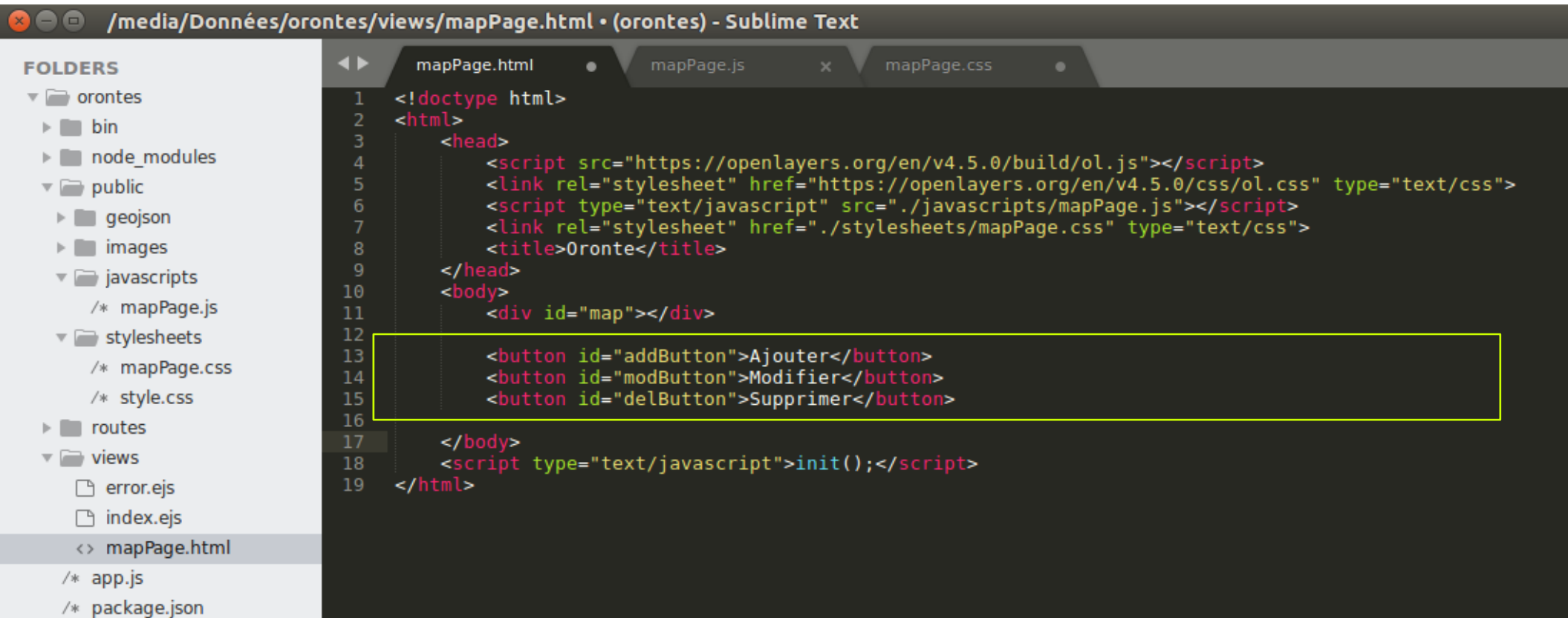
## ol.interaction

```
import interaction from 'ol/interaction';
```

### Classes

- DoubleClickZoom
- DragAndDrop
- DragBox
- DragPan
- DragRotate
- DragRotateAndZoom
- DragZoom
- Draw**
- Extent
- Interaction
- KeyboardPan
- KeyboardZoom
- Modify**
- MouseWheelZoom
- PinchRotate
- PinchZoom
- Pointer
- Select**
- Snap**
- Translate

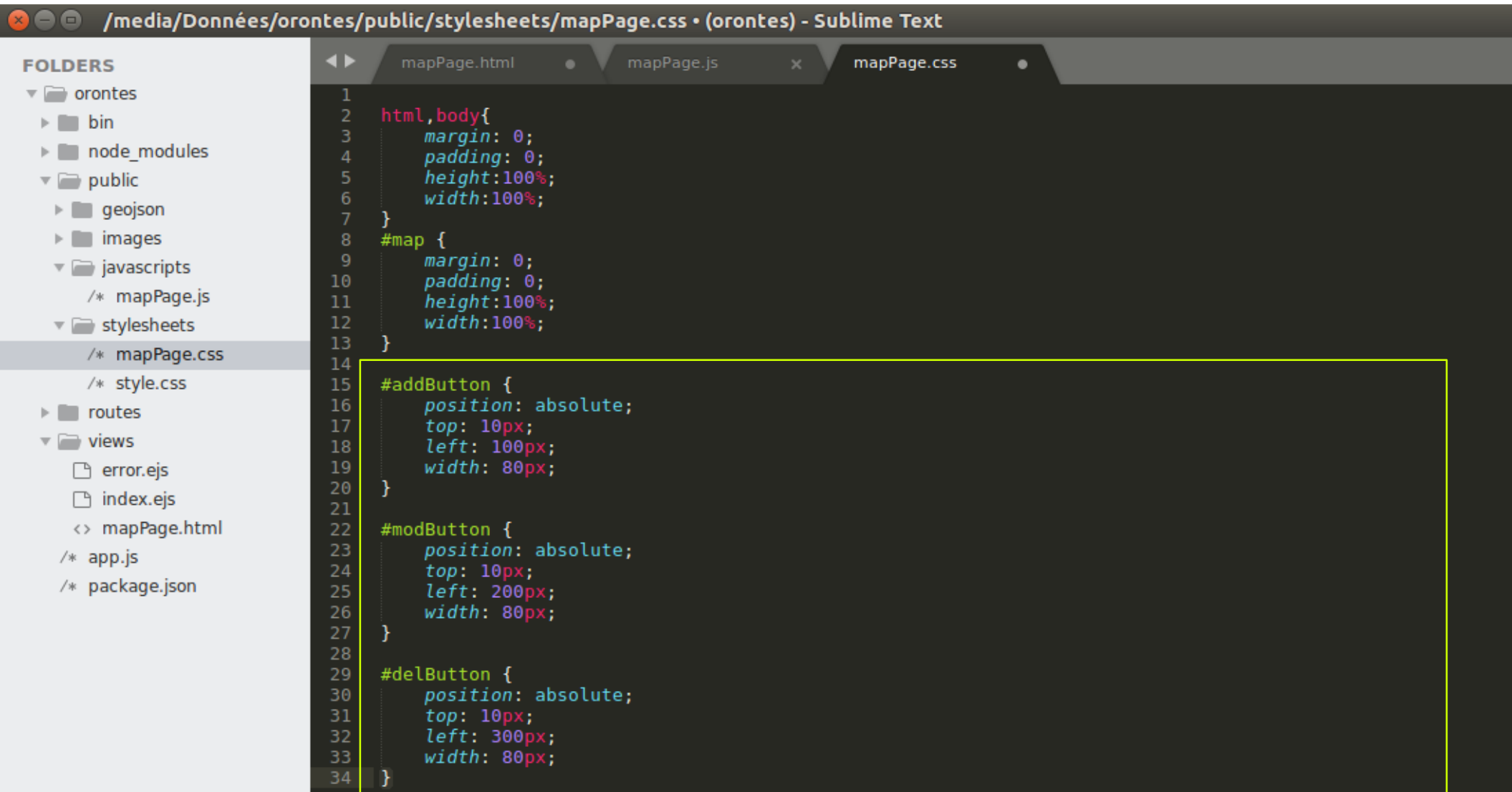
Ajout de boutons pour passer en mode  
création/modification/suppression d'objets...



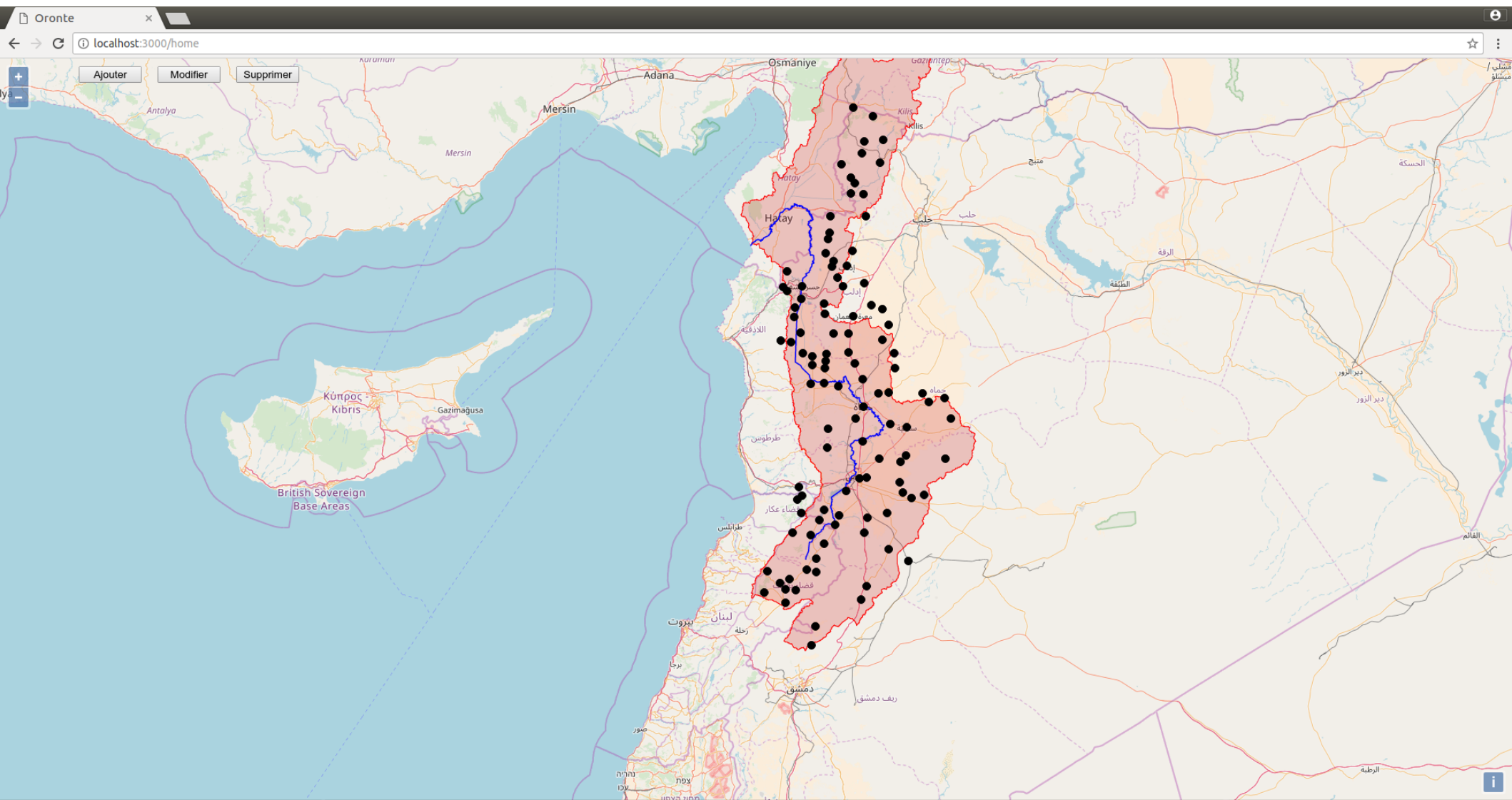
The screenshot shows the Sublime Text editor with the file `/media/Données/orontes/views/mapPage.html` open. The left sidebar displays the project structure, including folders like `orontes`, `bin`, `node_modules`, `public`, `geojson`, `images`, `javascripts`, `stylesheets`, `routes`, and `views`. The `views` folder is expanded, showing `error.ejs`, `index.ejs`, and `mapPage.html`. The main editor area shows the content of `mapPage.html`, which includes a DOCTYPE declaration, a head section with links to OpenLayers CSS and JavaScript, and a body section with a map container and three buttons: `Ajouter`, `Modifier`, and `Supprimer`. The buttons are highlighted with a yellow box.

```
1 <!doctype html>
2 <html>
3   <head>
4     <script src="https://openlayers.org/en/v4.5.0/build/ol.js"></script>
5     <link rel="stylesheet" href="https://openlayers.org/en/v4.5.0/css/ol.css" type="text/css">
6     <script type="text/javascript" src="./javascripts/mapPage.js"></script>
7     <link rel="stylesheet" href="./stylesheets/mapPage.css" type="text/css">
8     <title>Oronte</title>
9   </head>
10  <body>
11    <div id="map"></div>
12
13    <button id="addButton">Ajouter</button>
14    <button id="modButton">Modifier</button>
15    <button id="delButton">Supprimer</button>
16
17  </body>
18  <script type="text/javascript">init();</script>
19 </html>
```

## Définition de styles pour les nouveaux boutons...

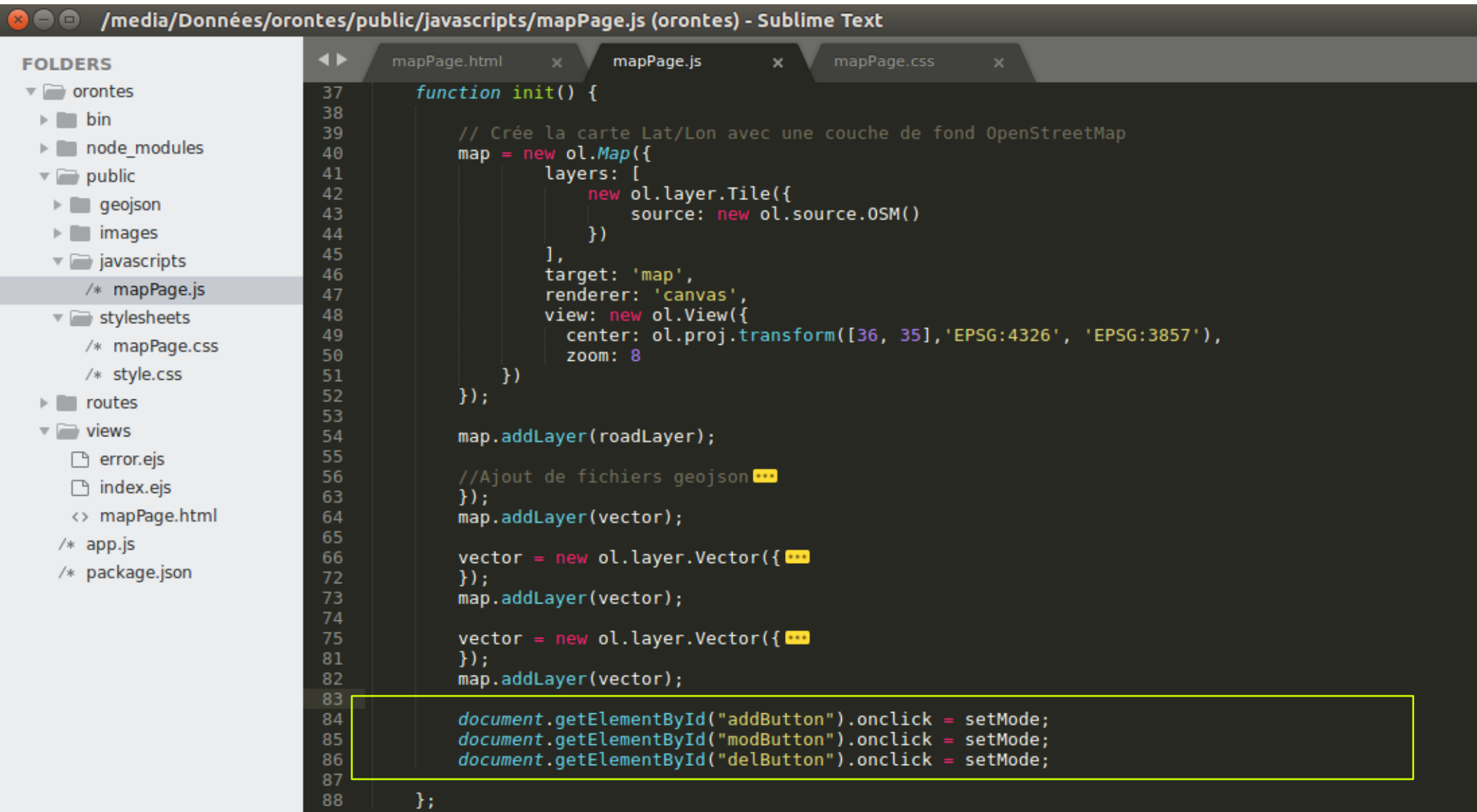


```
1
2  html,body{
3      margin: 0;
4      padding: 0;
5      height:100%;
6      width:100%;
7  }
8  #map {
9      margin: 0;
10     padding: 0;
11     height:100%;
12     width:100%;
13 }
14
15 #addButton {
16     position: absolute;
17     top: 10px;
18     left: 100px;
19     width: 80px;
20 }
21
22 #modButton {
23     position: absolute;
24     top: 10px;
25     left: 200px;
26     width: 80px;
27 }
28
29 #delButton {
30     position: absolute;
31     top: 10px;
32     left: 300px;
33     width: 80px;
34 }
```



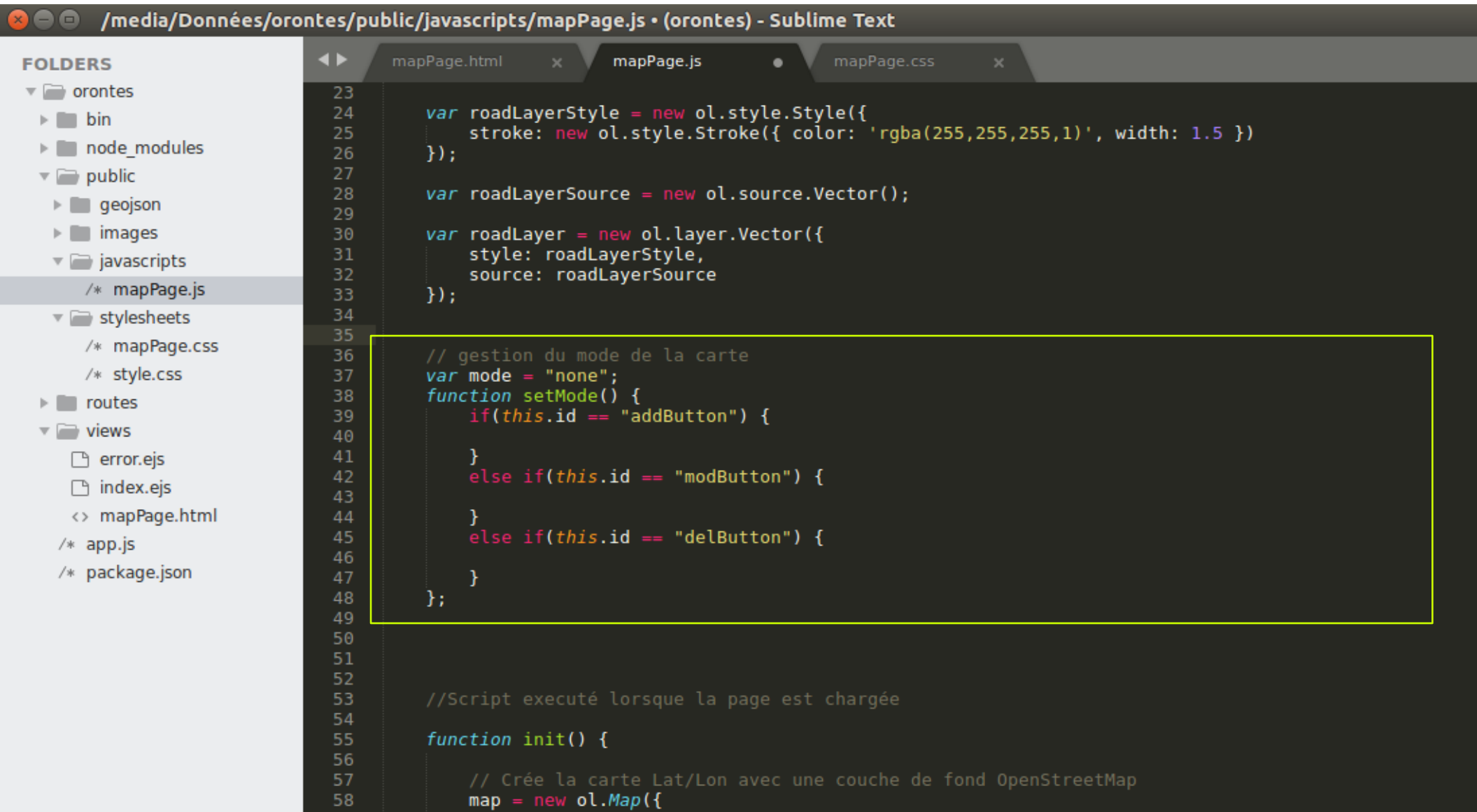


Ajout d'évènements click pour les nouveaux boutons...



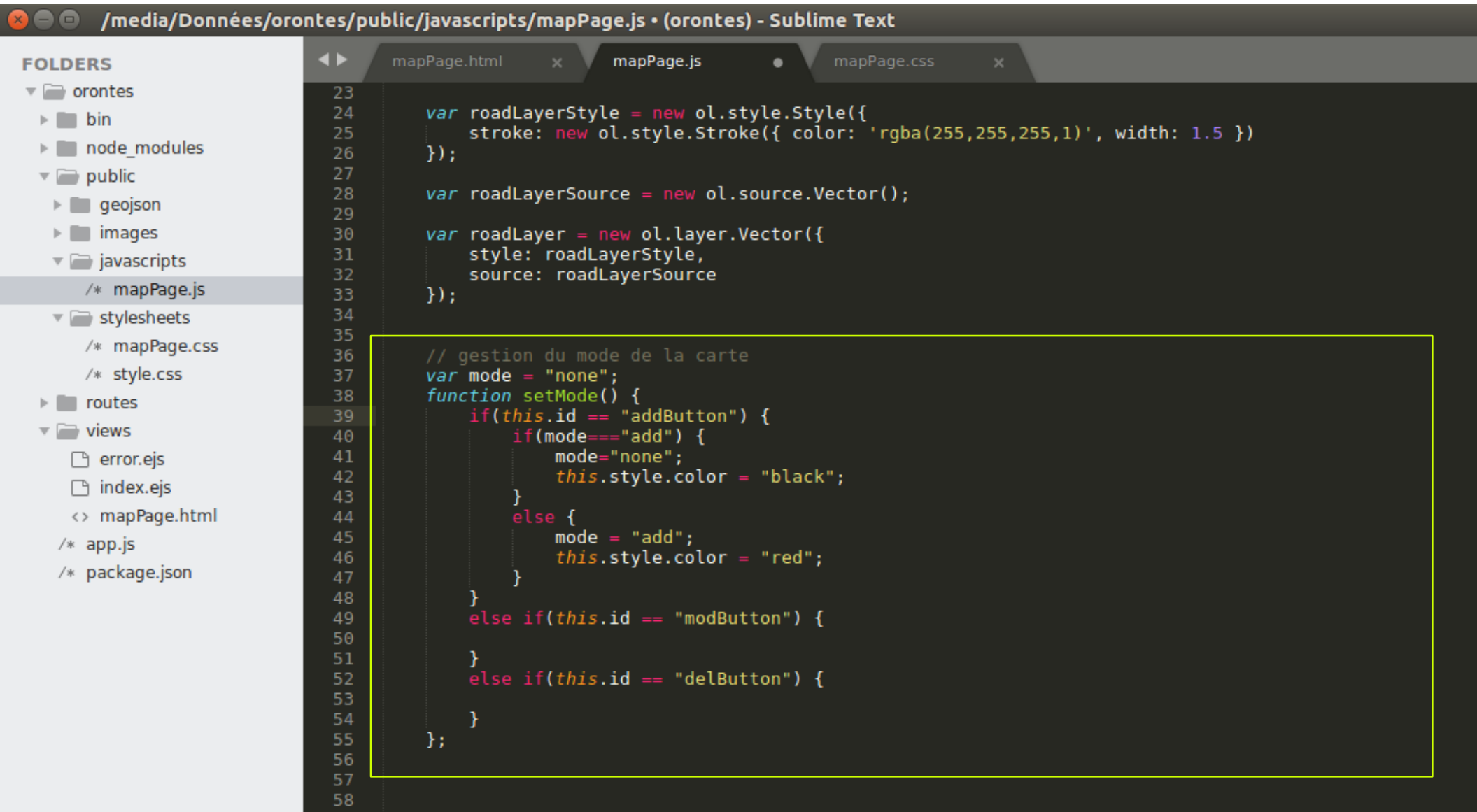
```
37  function init() {
38
39      // Crée la carte Lat/Lon avec une couche de fond OpenStreetMap
40      map = new ol.Map({
41          layers: [
42              new ol.layer.Tile({
43                  source: new ol.source.OSM()
44              })
45          ],
46          target: 'map',
47          renderer: 'canvas',
48          view: new ol.View({
49              center: ol.proj.transform([36, 35], 'EPSG:4326', 'EPSG:3857'),
50              zoom: 8
51          })
52      });
53
54      map.addLayer(roadLayer);
55
56      //Ajout de fichiers geojson ***
57      });
58      map.addLayer(vector);
59
60      vector = new ol.layer.Vector({ ***
61      });
62      map.addLayer(vector);
63
64      vector = new ol.layer.Vector({ ***
65      });
66      map.addLayer(vector);
67
68      document.getElementById("addButton").onclick = setMode;
69      document.getElementById("modButton").onclick = setMode;
70      document.getElementById("delButton").onclick = setMode;
71
72  };
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
```

Ajout du code pour ces évènements...



```
23
24     var roadLayerStyle = new ol.style.Style({
25         stroke: new ol.style.Stroke({ color: 'rgba(255,255,255,1)', width: 1.5 })
26     });
27
28     var roadLayerSource = new ol.source.Vector();
29
30     var roadLayer = new ol.layer.Vector({
31         style: roadLayerStyle,
32         source: roadLayerSource
33     });
34
35
36     // gestion du mode de la carte
37     var mode = "none";
38     function setMode() {
39         if(this.id == "addButton") {
40
41         }
42         else if(this.id == "modButton") {
43
44         }
45         else if(this.id == "delButton") {
46
47         }
48     };
49
50
51
52
53     //Script executé lorsque la page est chargée
54
55     function init() {
56
57         // Crée la carte Lat/Lon avec une couche de fond OpenStreetMap
58         map = new ol.Map({
```

Changement de couleur du bouton lorsque mode correspondant est activé/désactivé...



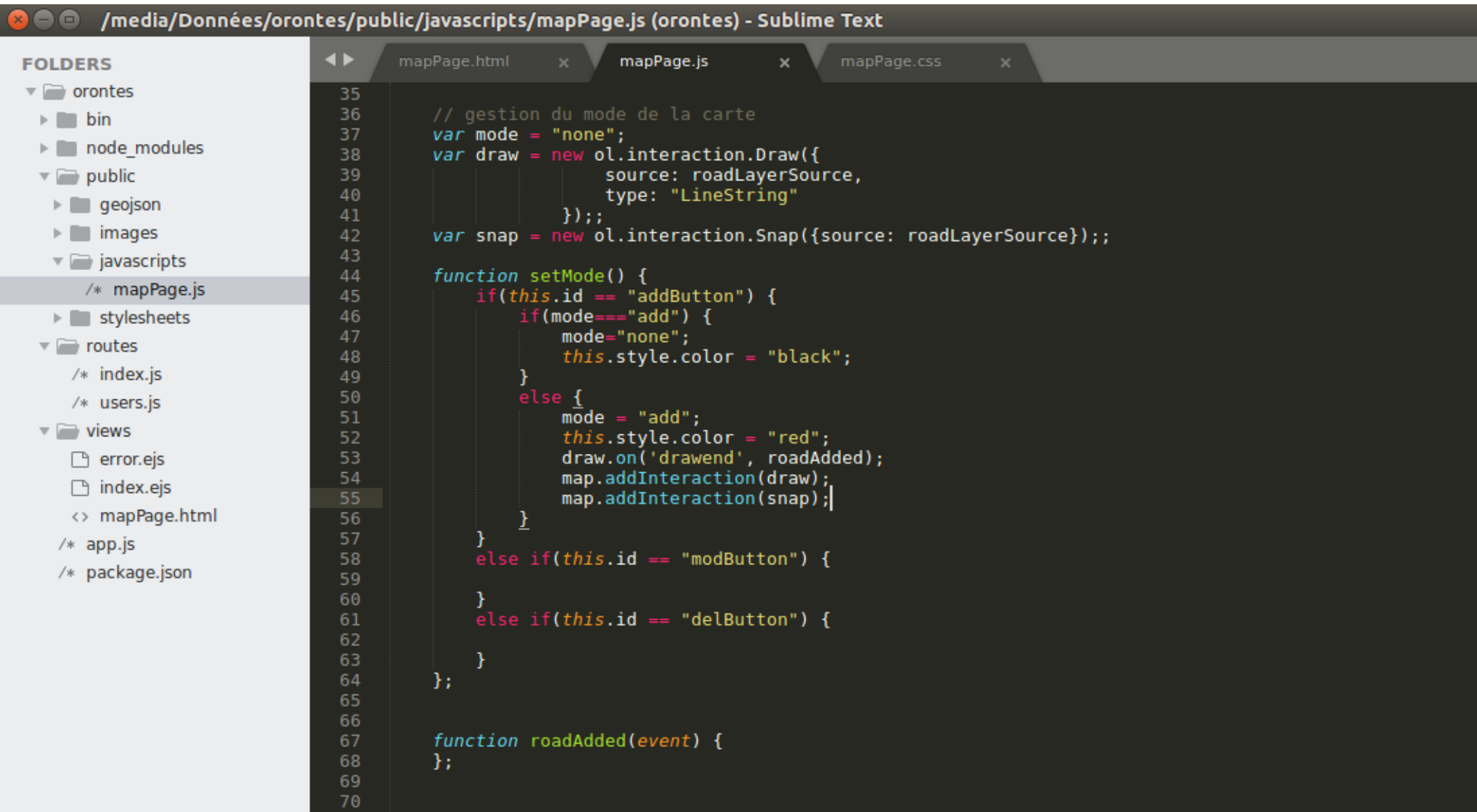
The screenshot shows a Sublime Text editor window titled "/media/Données/orontes/public/javascripts/mapPage.js • (orontes) - Sublime Text". The left sidebar displays a "FOLDERS" panel with the following structure:

- orontes
  - bin
  - node\_modules
  - public
    - geojson
    - images
    - javascripts
      - /\* mapPage.js
    - stylesheets
      - /\* mapPage.css
      - /\* style.css
  - routes
  - views
    - error.ejs
    - index.ejs
    - <> mapPage.html
  - /\* app.js
  - /\* package.json

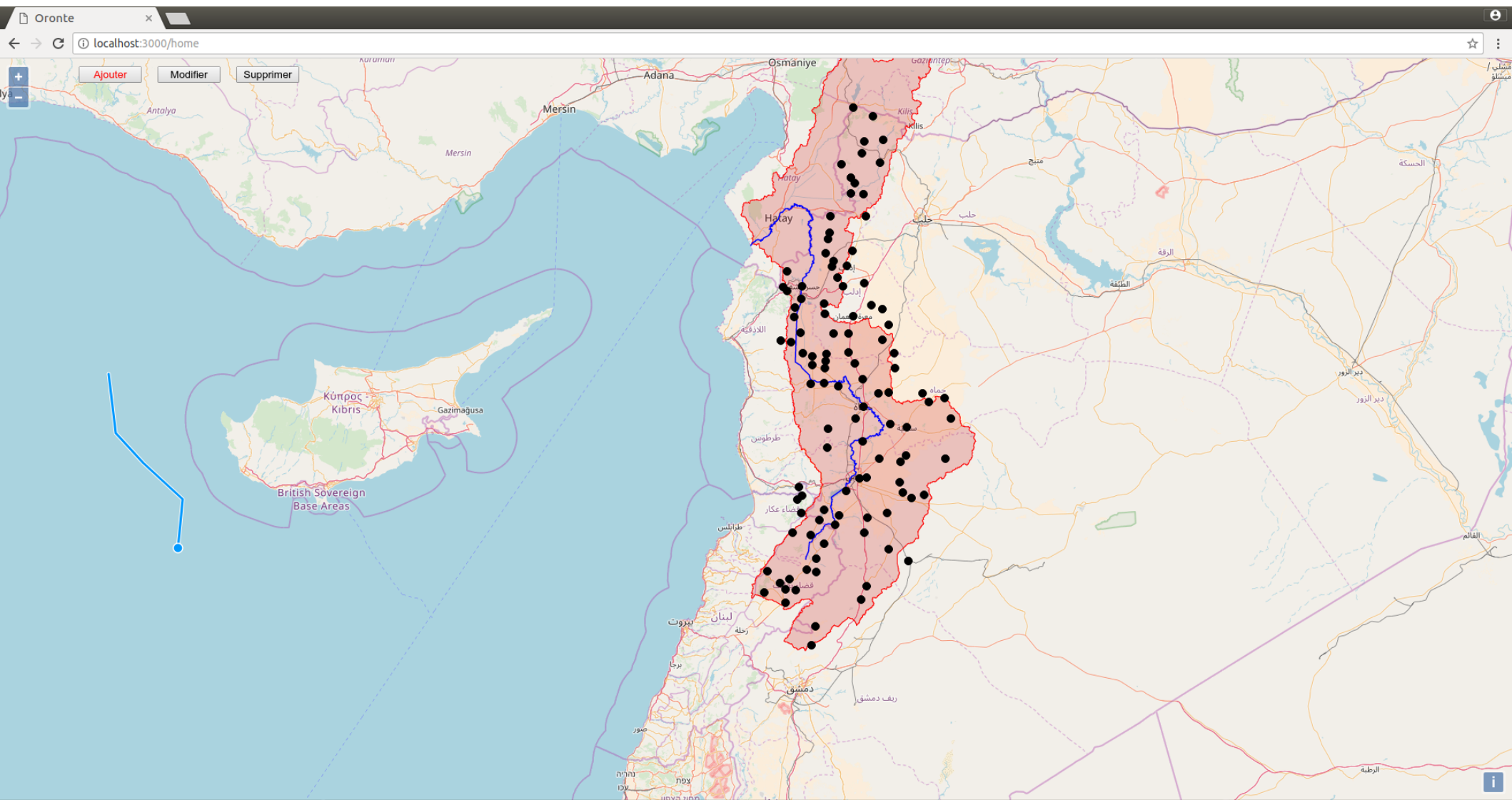
The main editor area shows the content of mapPage.js. The code is as follows:

```
23
24     var roadLayerStyle = new ol.style.Style({
25         stroke: new ol.style.Stroke({ color: 'rgba(255,255,255,1)', width: 1.5 })
26     });
27
28     var roadLayerSource = new ol.source.Vector();
29
30     var roadLayer = new ol.layer.Vector({
31         style: roadLayerStyle,
32         source: roadLayerSource
33     });
34
35
36     // gestion du mode de la carte
37     var mode = "none";
38     function setMode() {
39         if(this.id == "addButton") {
40             if(mode=="add") {
41                 mode="none";
42                 this.style.color = "black";
43             }
44             else {
45                 mode = "add";
46                 this.style.color = "red";
47             }
48         }
49         else if(this.id == "modButton") {
50
51         }
52         else if(this.id == "delButton") {
53
54         }
55     };
56
57
58
```

Ajout de l'interaction Draw... et de la fonction qui sera exécutée à la fin du dessin



```
35
36 // gestion du mode de la carte
37 var mode = "none";
38 var draw = new ol.interaction.Draw({
39     source: roadLayerSource,
40     type: "LineString"
41 });
42 var snap = new ol.interaction.Snap({source: roadLayerSource});
43
44 function setMode() {
45     if(this.id == "addButton") {
46         if(mode=="add") {
47             mode="none";
48             this.style.color = "black";
49         }
50         else {
51             mode = "add";
52             this.style.color = "red";
53             draw.on('drawend', roadAdded);
54             map.addInteraction(draw);
55             map.addInteraction(snap);
56         }
57     }
58     else if(this.id == "modButton") {
59     }
60     else if(this.id == "delButton") {
61     }
62 }
63
64 };
65
66
67 function roadAdded(event) {
68 }
69
70
```



# Recherche des noms des évènements provoqués par l'interaction "Modify"...

OpenLayers v4.5.0 API - X

openlayers.org/en/latest/apidoc/ol.interaction.Modify.html

Search Documentation

ol.interaction.Modify

Methods

- handleEvent
- changed
- dispatchEvent
- get
- getActive
- getKeys
- getMap
- getProperties
- getRevision
- on
- once
- removePoint
- set
- setActive
- setProperties
- un
- unset

Fires

- change
- change:active
- modifyend
- modifystart
- propertychange

ol

ol.AssertionError

ol.Attribution

ol.CanvasMap

ol.Collection

ol.Collection.Event

ol.DeviceOrientation

## ol.interaction.Modify

```
import Modify from 'ol/interaction/modify';
```

Interaction for modifying feature geometries. To modify features that have been added to an existing source, construct the modify interaction with the `source` option. If you want to modify features in a collection (for example, the collection used by a select interaction), construct the interaction with the `features` option. The interaction must be constructed with either a `source` or `features` option.

By default, the interaction will allow deletion of vertices when the `alt` key is pressed. To configure the interaction with a different condition for deletion, use the `deleteCondition` option.

**new ol.interaction.Modify (options)**

Name	Type	Description
<b>options</b>	Options.	
<b>condition</b>	<code>ol.EventsConditionType</code>   undefined	A function that takes an <code>ol.MapBrowserEvent</code> and returns a boolean to indicate whether that event will be considered to add or move a vertex to the sketch. Default is <code>ol.events.condition.primaryAction</code> .
<b>deleteCondition</b>	<code>ol.EventsConditionType</code>   undefined	A function that takes an <code>ol.MapBrowserEvent</code> and returns a boolean to indicate whether that event should be handled. By default, <code>ol.events.condition.singleClick</code> with <code>ol.events.condition.altKeyOnly</code> results in vertex deletion.
<b>insertVertexCondition</b>	<code>ol.EventsConditionType</code>   undefined	
<b>pixelTolerance</b>	number	
<b>style</b>	<code>ol.Style</code>   <code>ol.Styles</code>	
<b>source</b>	<code>ol.Source</code>	
<b>features</b>	<code>ol.Collection</code>	
<b>wrapX</b>	boolean	

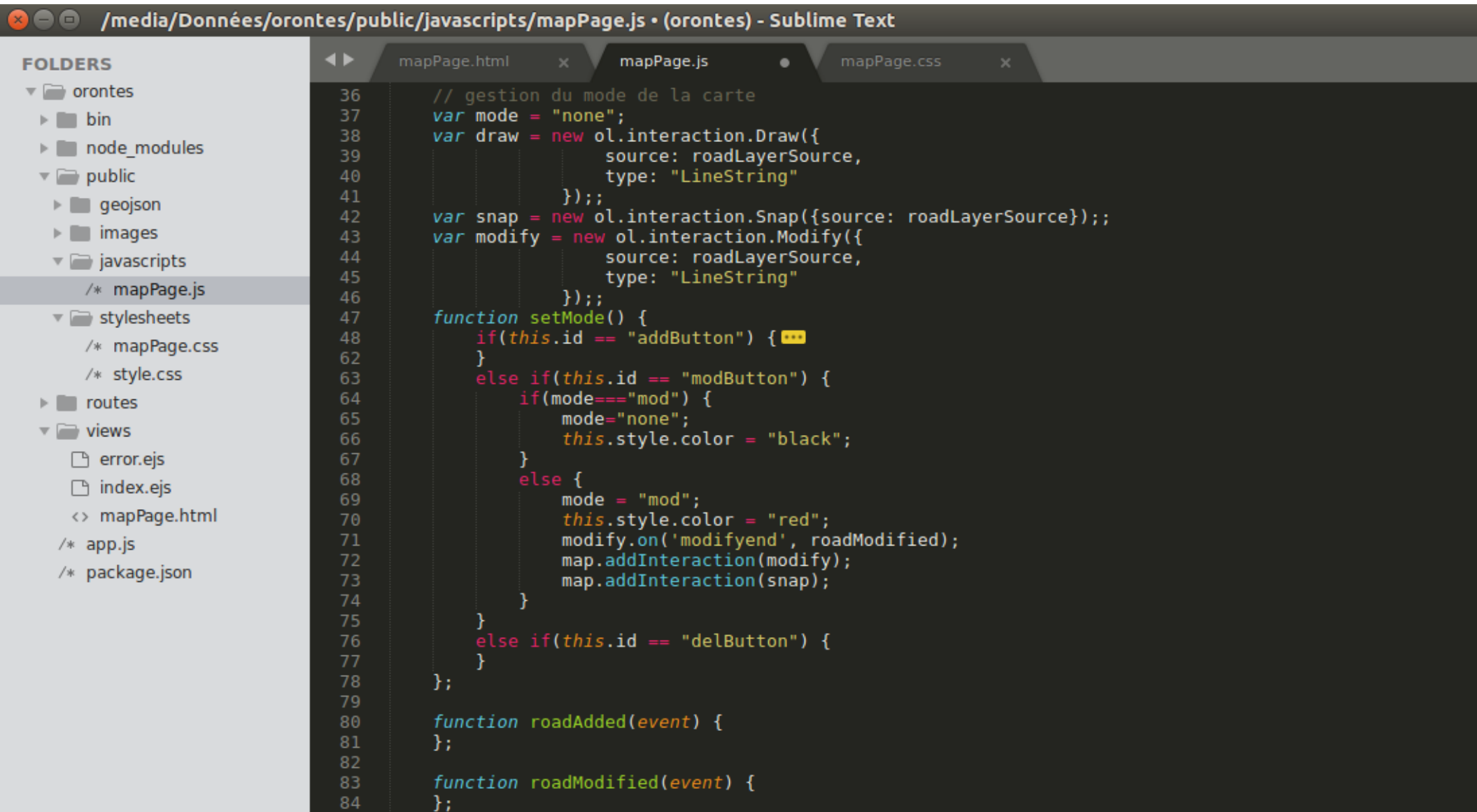
**Fires:**

- `change` (`ol.events.Event`) - Generic change event. Triggered when the revision counter is increased.
- `change:active` (`ol.Object.Event`)
- `modifyend` (`ol.interaction.Modify.Event`) - Triggered upon feature modification end
- `modifystart` (`ol.interaction.Modify.Event`) - Triggered upon feature modification start
- `propertychange` (`ol.Object.Event`) - Triggered when a property is changed.

src/ol/interaction/modify.js, line 45

openlayers.org/en/latest/apidoc/ol.interaction.Modify.Event.html#event:modifyend

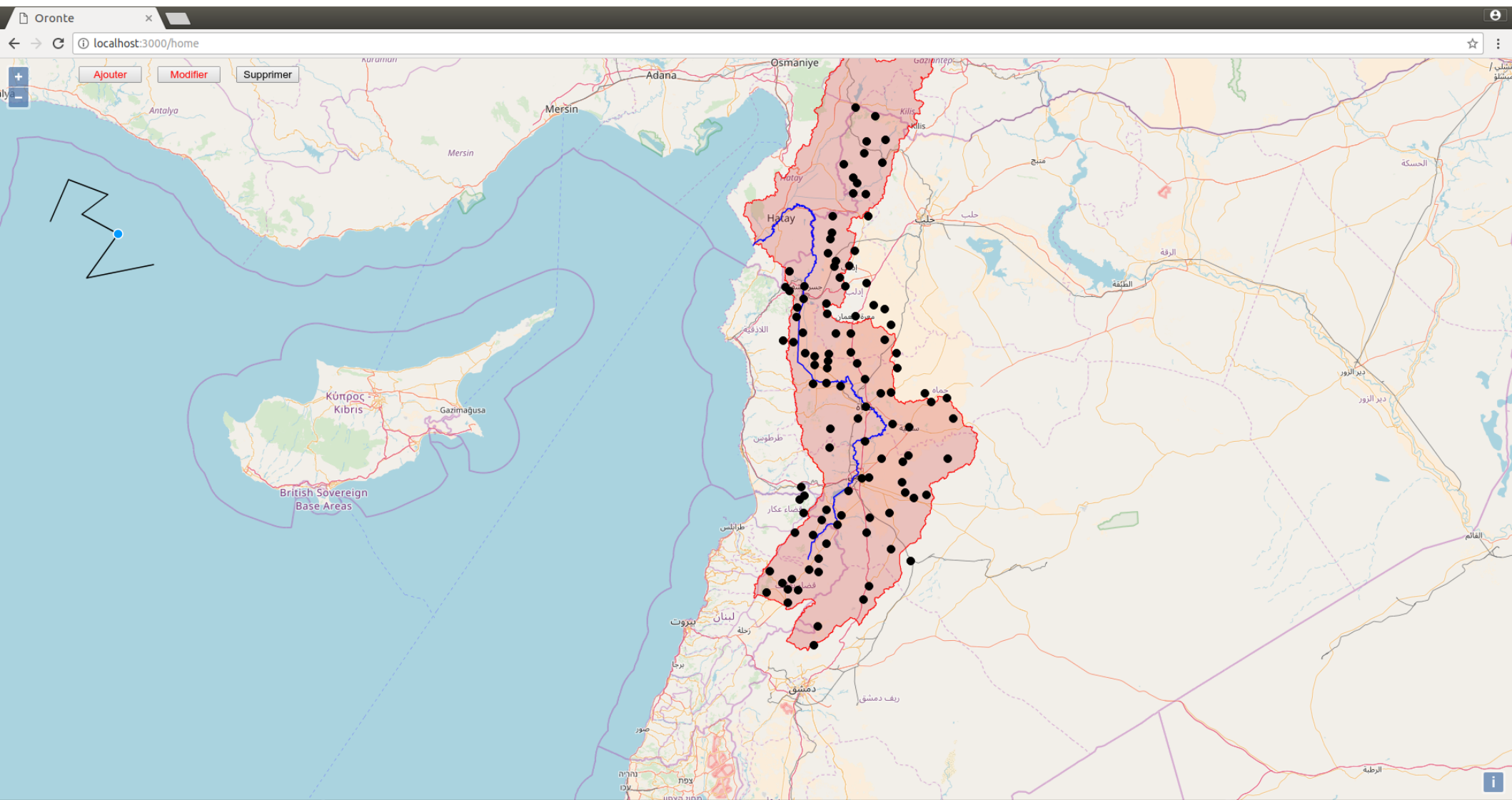
Ajout du code pour le mode “modifier”...



```
36 // gestion du mode de la carte
37 var mode = "none";
38 var draw = new ol.interaction.Draw({
39     source: roadLayerSource,
40     type: "LineString"
41 });
42 var snap = new ol.interaction.Snap({source: roadLayerSource});
43 var modify = new ol.interaction.Modify({
44     source: roadLayerSource,
45     type: "LineString"
46 });
47 function setMode() {
48     if(this.id == "addButton") {
49     }
50     else if(this.id == "modButton") {
51         if(mode=="mod") {
52             mode="none";
53             this.style.color = "black";
54         }
55         else {
56             mode = "mod";
57             this.style.color = "red";
58             modify.on('modifyend', roadModified);
59             map.addInteraction(modify);
60             map.addInteraction(snap);
61         }
62     }
63     else if(this.id == "delButton") {
64     }
65 }
66
67 function roadAdded(event) {
68 }
69
70 function roadModified(event) {
71 }
```

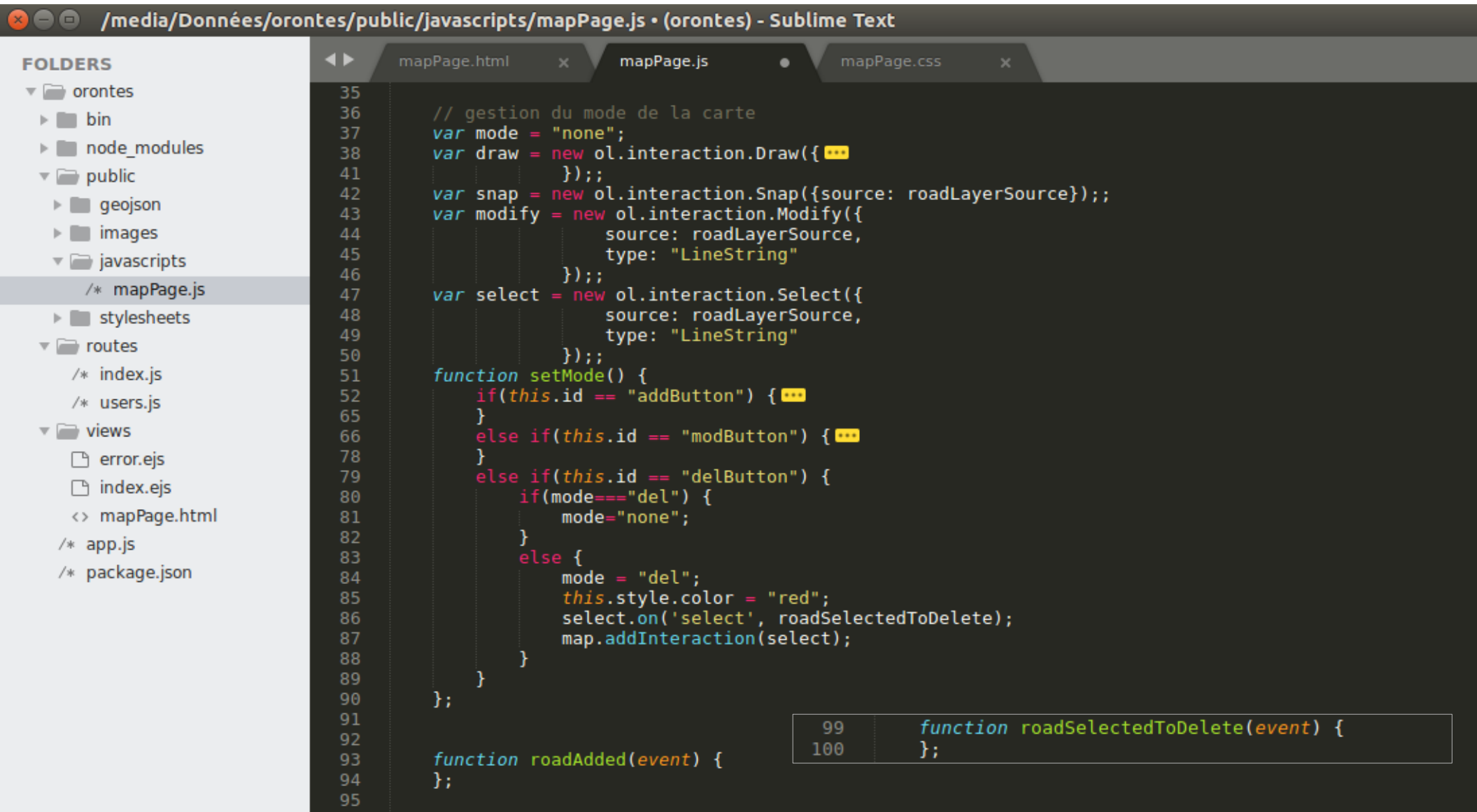


...et ça marche, sauf que le bouton ajouter, utilisé pour créer l'élément qui est en cours de modification, est resté rouge





... reste plus qu'à ajouter le code pour le bouton supprimer



The screenshot shows a Sublime Text editor window titled "/media/Données/orontes/public/javascripts/mapPage.js • (orontes) - Sublime Text". The left sidebar displays a file explorer with the following structure:

- orontes
  - bin
  - node\_modules
  - public
    - geojson
    - images
    - javascripts
      - /\* mapPage.js
    - stylesheets
  - routes
    - /\* index.js
    - /\* users.js
  - views
    - error.ejs
    - index.ejs
    - <> mapPage.html
  - /\* app.js
  - /\* package.json

The main editor area shows the following JavaScript code:

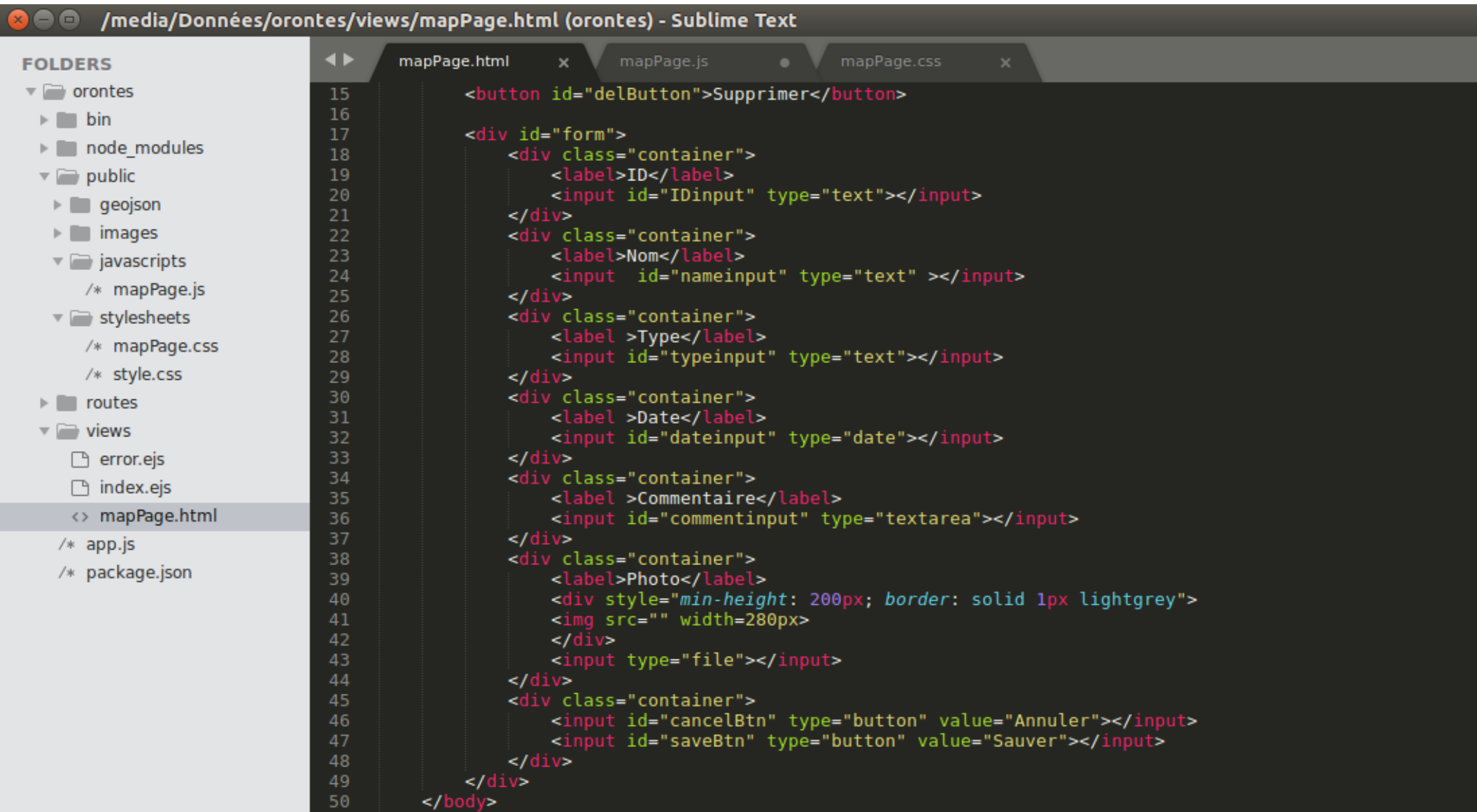
```
35
36 // gestion du mode de la carte
37 var mode = "none";
38 var draw = new ol.interaction.Draw({ ***
41     });
42 var snap = new ol.interaction.Snap({source: roadLayerSource});
43 var modify = new ol.interaction.Modify({
44     source: roadLayerSource,
45     type: "LineString"
46 });
47 var select = new ol.interaction.Select({
48     source: roadLayerSource,
49     type: "LineString"
50 });
51 function setMode() {
52     if(this.id == "addButton") { ***
65     }
66     else if(this.id == "modButton") { ***
78     }
79     else if(this.id == "delButton") {
80         if(mode=="del") {
81             mode="none";
82         }
83         else {
84             mode = "del";
85             this.style.color = "red";
86             select.on('select', roadSelectedToDelete);
87             map.addInteraction(select);
88         }
89     }
90 };
91
92
93 function roadAdded(event) {
94 };
95
```

At the bottom right, a function definition is partially visible:

```
99 function roadSelectedToDelete(event) {
100     };

```

Lorsqu'un nouvel objet a été dessiné, il faut pouvoir documenter ses attributs  
>> formulaire des saisie

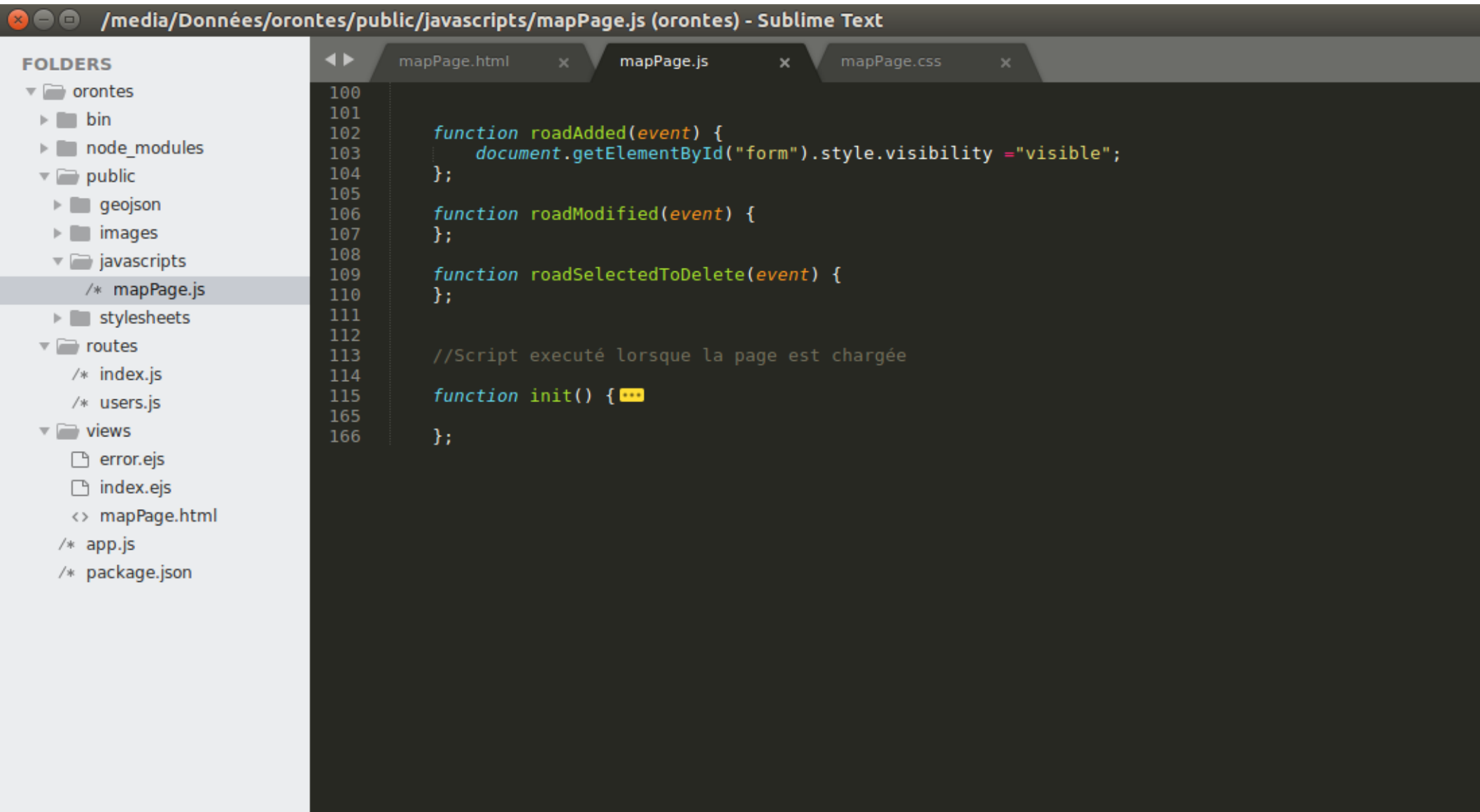


```
15 <button id="delButton">Supprimer</button>
16
17 <div id="form">
18   <div class="container">
19     <label>ID</label>
20     <input id="IDinput" type="text"></input>
21   </div>
22   <div class="container">
23     <label>Nom</label>
24     <input id="nameinput" type="text" ></input>
25   </div>
26   <div class="container">
27     <label >Type</label>
28     <input id="typeinput" type="text"></input>
29   </div>
30   <div class="container">
31     <label >Date</label>
32     <input id="dateinput" type="date"></input>
33   </div>
34   <div class="container">
35     <label >Commentaire</label>
36     <input id="commentinput" type="textarea"></input>
37   </div>
38   <div class="container">
39     <label>Photo</label>
40     <div style="min-height: 200px; border: solid 1px lightgrey">
41       <img src="" width=280px>
42     </div>
43     <input type="file"></input>
44   </div>
45   <div class="container">
46     <input id="cancelBtn" type="button" value="Annuler"></input>
47     <input id="saveBtn" type="button" value="Sauver"></input>
48   </div>
49 </div>
50 </body>
```

... qui nécessite quelques éléments de style

```
28
29 #delButton {
30     position: absolute;
31     top: 10px;
32     left: 300px;
33     width: 80px;
34 }
35 #form {
36     position: absolute;
37     top: 0;
38     right: 0;
39     height: 100%;
40     width: 300px;
41     background: whitesmoke;
42     visibility: collapse;
43 }
44 .container {
45     margin: 10px;
46     padding: 10px;
47 }
48 label {
49     display: block;
50     margin-bottom: 10px;
51 }
52 input[type=button] {
53     width: 100px;
54 }
```

Il faut ensuite rendre ce formulaire visible lorsque le dessin est terminé



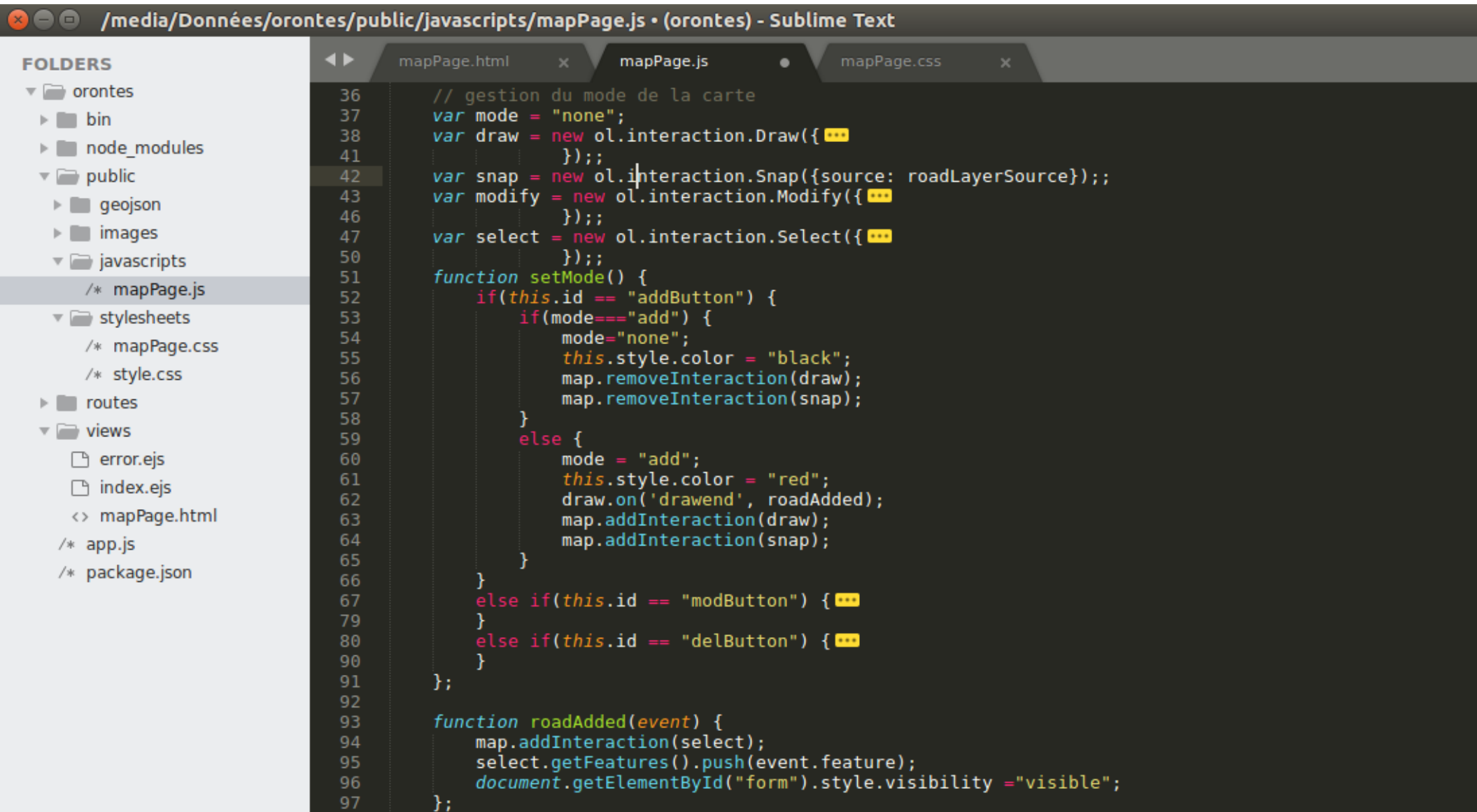
The screenshot shows a Sublime Text editor window titled `/media/Données/orontes/public/javascripts/mapPage.js (orontes) - Sublime Text`. The left sidebar displays a 'FOLDERS' panel with the following structure:

- orontes
  - bin
  - node\_modules
  - public
    - geojson
    - images
    - javascripts
      - `/* mapPage.js`
    - stylesheets
  - routes
    - `/* index.js`
    - `/* users.js`
  - views
    - error.ejs
    - index.ejs
    - `<> mapPage.html`
  - `/* app.js`
  - `/* package.json`

The main editor area shows the content of `mapPage.js` with line numbers 100 to 166. The code is as follows:

```
100
101
102     function roadAdded(event) {
103         document.getElementById("form").style.visibility ="visible";
104     };
105
106     function roadModified(event) {
107     };
108
109     function roadSelectedToDelete(event) {
110     };
111
112
113     //Script executé lorsque la page est chargée
114
115     function init() { ...
165
166     };
```

Et faire un peu de ménage...



The screenshot shows a Sublime Text editor window titled `/media/Données/orontes/public/javascripts/mapPage.js • (orontes) - Sublime Text`. The left sidebar displays a file explorer with the following structure:

- orontes
  - bin
  - node\_modules
  - public
    - geojson
    - images
    - javascripts
      - `/* mapPage.js`
    - stylesheets
      - `/* mapPage.css`
      - `/* style.css`
  - routes
  - views
    - `error.ejs`
    - `index.ejs`
    - `<> mapPage.html`
  - `/* app.js`
  - `/* package.json`

The main editor area shows the content of `mapPage.js` with the following code:

```
36 // gestion du mode de la carte
37 var mode = "none";
38 var draw = new ol.interaction.Draw({ ***
41 });
42 var snap = new ol.interaction.Snap({source: roadLayerSource});
43 var modify = new ol.interaction.Modify({ ***
46 });
47 var select = new ol.interaction.Select({ ***
50 });
51 function setMode() {
52   if(this.id == "addButton") {
53     if(mode=="add") {
54       mode="none";
55       this.style.color = "black";
56       map.removeInteraction(draw);
57       map.removeInteraction(snap);
58     }
59     else {
60       mode = "add";
61       this.style.color = "red";
62       draw.on('drawend', roadAdded);
63       map.addInteraction(draw);
64       map.addInteraction(snap);
65     }
66   }
67   else if(this.id == "modButton") { ***
79 }
80   else if(this.id == "delButton") { ***
90 }
91 };
92
93 function roadAdded(event) {
94   map.addInteraction(select);
95   select.getFeatures().push(event.feature);
96   document.getElementById("form").style.visibility ="visible";
97 };
```

