

---

# **Master Class OL3 Documentation**

***Version 0.1***

**Éric Lemoine**

04 June 2013



<b>1</b>	<b>Exercices élémentaires</b>	<b>3</b>
1.1	Exercice élémentaire 1 . . . . .	4
1.2	Exercice élémentaire 2 . . . . .	4
1.3	Exercice élémentaire 3 . . . . .	4
1.4	Exercice élémentaire 4 . . . . .	5
1.5	Exercice élémentaire 5 . . . . .	5
1.6	Exercice élémentaire 6 . . . . .	6
1.7	Exercice élémentaire 7 . . . . .	6
1.8	Exercice élémentaire 8 . . . . .	6
1.9	Exercice élémentaire 9 . . . . .	7
1.10	Exercice élémentaire 10 . . . . .	7
<b>2</b>	<b>Corrections des exercices élémentaires</b>	<b>9</b>
2.1	Correction exercice élémentaire 1 . . . . .	9
2.2	Correction exercice élémentaire 2 . . . . .	9
2.3	Correction exercice élémentaire 3 . . . . .	10
2.4	Correction exercice élémentaire 4 . . . . .	10
2.5	Correction exercice élémentaire 5 . . . . .	11
2.6	Correction exercice élémentaire 6 . . . . .	12
2.7	Correction exercice élémentaire 7 . . . . .	13
2.8	Correction exercice élémentaire 8 . . . . .	14
2.9	Correction exercice élémentaire 9 . . . . .	15
2.10	Correction exercice élémentaire 10 . . . . .	16



Ce document contient les énoncés et corrections des exercices de la *master class* OpenLayers 3 donnée aux [Rencontres SIG-la-lettre 2013](#). Il est disponible en ligne à <http://erilem.net/master-class-ol3/exercices/>. Une version PDF peut être téléchargée à <http://erilem.net/master-class-ol3/exercices/exercices.pdf>.

Exercices :



---

## Exercices élémentaires

---

Tous les exercices de cette partie se font à partir de la page HTML suivante :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">
    <title>Exercice élémentaire</title>
    <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">
    <style>
      #map {
        width: 600px;
        height: 400px;
      }
    </style>
  </head>
  <body>
    <div id="map"></div>
    <script src="http://ol3js.org/en/master/build/ol.js"></script>
    <script>
      var map = new ol.Map({
        target: 'map',
        view: new ol.View2D({
          zoom: 0,
          center: [0, 0]
        }),
        layers: [
          new ol.layer.TileLayer({
            source: new ol.source.OSM()
          })
        ]
      });
    </script>
  </body>
</html>
```

---

**Note :** Pensez à lire les **indices** !

---

## 1.1 Exercice élémentaire 1

*Afficher une carte simple*

1. Copier le code HTML donné ci-dessus et le coller dans un fichier du nom de votre choix, `exercice-elementaire-1.html` par exemple.
2. Placer ce fichier dans un répertoire rendu accessible par HTTP par le serveur web.
3. Ouvrir la page web dans un navigateur, en utilisant `http` comme protocole d'accès à la page.
4. Vérifier que l'exemple fonctionne correctement.
5. Utiliser les *interactions* et les *controls* de la carte.

(Pas de correction pour cet exercice.)

## 1.2 Exercice élémentaire 2

*Choisir le renderer de la carte*

OL3 peut utiliser trois technologies web différentes pour l'affichage de la carte : WebGL, Canvas, et DOM. Si rien n'est précisé par l'utilisateur la librairie OL3 choisit WebGL si le navigateur prend en charge WebGL. Sinon OL3 choisit Canvas, et enfin, si Canvas n'est pas pris en charge, c'est le *renderer* DOM qui est utilisé.

1. Copier le code HTML de base dans un nouveau fichier, `exercice-elementaire-2.html` par exemple.
2. Faire en sorte que le *renderer* Canvas soit utilisé par la carte. Il faut pour ceci positionner la propriété `renderer` dans l'objet d'options passé au constructeur `ol.Map`. Et la valeur pour cette propriété doit être `ol.RendererHint.CANVAS`.
3. Faire maintenant en sorte que ce soit le *renderer* DOM qui soit utilisé.
4. Utiliser maintenant la valeur `ol.RendererHints.createFromQueryData()` pour que le *renderer* puisse être spécifié dans l'URL par le paramètre `renderer`. Exemple : `exercice-elementaire-2.html?renderer=canvas`.

*Correction exercice élémentaire 2*

---

**Indice :** Pour vérifier quel *renderer* est utilisé vous pouvez effectuer un clic droit sur la carte et sélectionner "Inspecter l'élément". Si c'est le *renderer* DOM qui est utilisé l'élément inspecté sera une balise `<img>`, sinon ce sera une balise `<canvas>`. Cette technique ne permet pas de distinguer Canvas de WebGL.

---

## 1.3 Exercice élémentaire 3

*Initialiser la vue de la carte*

1. Copier le code HTML de base dans un nouveau fichier, `exercice-elementaire-3.html` par exemple.
2. Modifier l'objet d'options passés au constructeur `ol.View2D` pour que la vue soit initialisée avec un niveau de zoom égal à 17.
3. Modifier à nouveau l'objet d'options pour que la vue soit initialisée avec un centre égal à `[288074.8449901076, 6247982.515792289]`.
4. Modifier à nouveau l'objet d'options pour que la vue soit pivotée (rotation) d'un angle de 45 dans le sens des aiguilles d'une montre.

*Correction exercice élémentaire 3*



## 1.4 Exercice élémentaire 4

### *Changer la vue de la carte*

1. Copier le code HTML de base dans un nouveau fichier, `exercice-elementaire-4.html` par exemple.
2. Ouvrir la nouvelle page HTML et ouvrir les outils de développement du navigateur. Vous pouvez utiliser la touche F12 comme raccourci.
3. Saisir dans la console des outils de développement la commande JavaScript permettant d'obtenir le centre actuel de la vue.
4. Toujours dans la console, agir sur la vue pour changer son centre à `[288074.8449901076, 6247982.515792289]`.
5. Obtenir dans la console la résolution actuelle de la vue.
6. Agir sur la vue pour passer à une résolution 131072.0 fois plus petite que la résolution actuelle.
7. Agir sur la vue pour faire pivoter la carte de 45 degré à l'est.
8. Ajouter les lignes JavaScript saisies précédemment dans la console dans le code JavaScript de la page HTML, juste après la création de la carte. Recharger la page dans le navigateur et vérifier que la vue est correcte.

### **Indice :**

- La variable `map` définie dans le code JavaScript de la page est globale. Elle est donc directement accessible dans la console.
- Pour obtenir l'objet `vue` (`ol.View2D`) avec lequel la carte a été configurée il faut utiliser `map.getView()`.
- Les objets de type `ol.View2D` fournissent des fonctions *getter* pour accéder aux états de la vue. Exemple : `view.getCenter()`.
- De la même façon ils fournissent des *setters* pour changer les états de la vue. Exemple : `view.setRotation(45)`.

### *Correction exercice élémentaire 4*

## 1.5 Exercice élémentaire 5

### *Aller un peu plus loin avec la vue*

1. Copier le code HTML de base dans un nouveau fichier, `exercice-elementaire-5.html` par exemple.
2. Modifier le code JavaScript de la page afin que l'objet `ol.View2D` soit créé à l'extérieur de la définition de l'objet des options passé à la carte. L'objet `ol.View2D` créé sera référencé par une variable nommée `view`. Et c'est cette référence qui devra être passée à la carte par l'intermédiaire de l'option `view`.
3. Après la création de la carte, agir sur la vue (référéncée par la variable `view`) pour recentrer celle-ci sur l'étendue `[287716.5464200208, 288433.14356019435, 6247743.650078897, 6248221.38150568]`.
4. Changer le code pour que la vue n'ait pas d'état initial, et pour qu'elle ne soit réellement définie que lorsque qu'elle est recentrée sur l'étendue spécifiée.
5. Tenter de comprendre pourquoi la fonction `fitExtent` a besoin des dimensions de la carte en pixels pour faire ce qui lui est demandé.

### **Indice :**

- C'est la fonction `fitExtent` de `ol.View2D` qui doit être utilisée pour ça.
- La fonction `getSize` de `ol.Map` doit aussi être utilisée pour cette exercice.
- Pensez à jeter un oeil à la doc de l'API : <http://ol3js.org/en/master/apidoc/>.

### *Correction exercice élémentaire 5*

## 1.6 Exercice élémentaire 6

*Utiliser une autre source de donnée de type OpenStreetMap*

1. Copier le code HTML de base dans un nouveau fichier, `exercice-elementaire-6.html` par exemple.
2. Changer les options de la vue pour centrer la carte sur l'ENSG. (Options utilisées dans *Exercice élémentaire 3* par exemple.)
3. Passer au constructeur `ol.source.OSM` un objet d'options contenant une propriété `url` dont la valeur est `http://{a-c}.tile3.opencyclemap.org/landscape/{z}/{x}/{y}.png`. Voir <http://www.thunderforest.com/landscape/> pour obtenir des informations sur cette source de donnée. Et voir `OSMOptions` dans la doc de l'API pour connaître toutes les options qui peuvent être passées à `ol.source.OSM`.
4. Ajouter une attribution à la source pour que les conditions d'utilisation des tuiles OpenCycleMap soient respectées. Une attribution du type `<a href="http://www.opencyclemap.org">OpenCycleMap</a>` est en accord avec ces conditions.

---

**Indice :** Regarder l'exemple <http://ol3js.org/en/master/examples/localized-openstreetmap.html> pour inspiration.

---

*Correction exercice élémentaire 6*

## 1.7 Exercice élémentaire 7

*Changer les paramètres d'affichage d'une couche*

1. Copier le code HTML de base dans un nouveau fichier, `exercice-elementaire-7.html` par exemple.
2. Dans l'objet d'options passé au constructeur `ol.layer.TileLayer` ajouter des propriétés `opacity`, `hue`, `saturation`, `brightness`, et `contrast`.
3. Dans la console, récupérer une référence sur la couche et changer sa visibilité.

---

**Indice :**

- `ol.Map` fournit une fonction `getLayers`. Cette fonction retourne un objet de type `ol.Collection`, qui lui-même fournit des méthodes pour accéder aux différents objets de la collection. Voir la doc de l'API pour plus d'informations.
- 

*Correction exercice élémentaire 7*

## 1.8 Exercice élémentaire 8

*Manipuler les projections*

1. Copier le code HTML de base dans un nouveau fichier, `exercice-elementaire-8.html` par exemple.
2. Dans une console, afficher le centre de la vue.
3. Le système de coordonnées (projection) de ce centre est "EPSG:3857" (connu sous le nom de Spherical Mercator). Convertir les coordonnées du centre de "EPSG:3857" à "EPSG:4326" (longitude/latitude WGS84).
4. Les coordonnées GPS de l'ENSG sont latitude : 48,8413379, longitude : 2,5878203. Modifier le code JavaScript de la page pour que la vue soit centrée sur ces coordonnées à l'état initial. Changer aussi le zoom à 17 pour un meilleur résultat.

---

**Indice :**

- La fonction `ol.proj.transform` est à utiliser pour transformer des coordonnées d'un système de coordonnées à un autre. Voir la doc de l'API.
- 

*Correction exercice élémentaire 8*

## 1.9 Exercice élémentaire 9

### *Manipuler les controls*

1. Copier le code HTML de base dans un nouveau fichier, `exercice-elementaire-9.html` par exemple.
2. Modifier le code JavaScript de la page de telle façon qu'une échelle graphique soit ajoutée à la carte.
3. Modifier la configuration du *control* pour que les unités de mesure anglo-saxonnes (inches, feets, miles) soient utilisées plutôt que les unités métriques.

---

### **Indice :**

- C'est le *control* `ol.control.ScaleLine` qui permet d'ajouter une échelle graphique sur la carte.
  - La fonction `ol.control.defaults` permet d'obtenir un tableau contenant les *controls* par défaut, et d'autres choisis de manière spécifique.
- 

### *Correction exercice élémentaire 9*

## 1.10 Exercice élémentaire 10

### *Manipuler les interactions*

1. Copier le code HTML de base dans un nouveau fichier, `exercice-elementaire-10.html` par exemple.
2. Modifier le code JavaScript de la page pour ajouter une interaction de type `ol.interaction.DragRotateAndZoom` à la carte. (C'est la touche `SHIFT` qui active cette interaction.)

### *Correction exercice élémentaire 10*

Corrections :



---

## Corrections des exercices élémentaires

---

### 2.1 Correction exercice élémentaire 1

#### *Exercice élémentaire 1*

Aucune correction pour cet exercice.

### 2.2 Correction exercice élémentaire 2

#### *Exercice élémentaire 2*

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">
    <title>Exercice élémentaire</title>
    <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">
    <style>
      #map {
        width: 600px;
        height: 400px;
      }
    </style>
  </head>
  <body>
    <div id="map"></div>
    <script src="http://ol3js.org/en/master/build/ol.js"></script>
    <script>
      var map = new ol.Map({
        target: 'map',
        view: new ol.View2D({
          zoom: 0,
          center: [0, 0]
        }),
        layers: [
          new ol.layer.TileLayer({
            source: new ol.source.OSM()
          })
        ],
        renderer: ol.RendererHint.CANVAS
      });
    </script>
  </body>
</html>
```

```
    });  
  </script>  
</body>  
</html>
```

Pour que `render` DOM soit utilisé il faut passer la valeur `ol.RendererHint.DOM` pour l'option `render`. Et pour que le `render` puisse être spécifié dans l'URL c'est la valeur retournée par la fonction `ol.RendererHints.createFromQueryData` qu'il faut passer.

## 2.3 Correction exercice élémentaire 3

### *Exercice élémentaire 3*

```
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="utf-8">  
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">  
    <title>Exercice élémentaire</title>  
    <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">  
    <style>  
      #map {  
        width: 600px;  
        height: 400px;  
      }  
    </style>  
  </head>  
  <body>  
    <div id="map"></div>  
    <script src="http://ol3js.org/en/master/build/ol.js"></script>  
    <script>  
      var map = new ol.Map({  
        target: 'map',  
        view: new ol.View2D({  
          zoom: 17,  
          center: [288074.8449901076, 6247982.515792289],  
          rotation: 45  
        }),  
        layers: [  
          new ol.layer.TileLayer({  
            source: new ol.source.OSM()  
          })  
        ]  
      });  
    </script>  
  </body>  
</html>
```

## 2.4 Correction exercice élémentaire 4

### *Exercice élémentaire 4*

```

> view = map.getView()
> view.getCenter()
> view.setCenter([288074.8449901076, 6247982.515792289])
> view.getResolution()
> view.setResolution(view.getResolution() / 131072.0)
> view.setRotation(45)

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">
    <title>Exercice élémentaire</title>
    <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">
    <style>
      #map {
        width: 600px;
        height: 400px;
      }
    </style>
  </head>
  <body>
    <div id="map"></div>
    <script src="http://ol3js.org/en/master/build/ol.js"></script>
    <script>
      var map = new ol.Map({
        target: 'map',
        view: new ol.View2D({
          zoom: 0,
          center: [0, 0]
        }),
        layers: [
          new ol.layer.TileLayer({
            source: new ol.source.OSM()
          })
        ]
      });
      view = map.getView()
      view.setCenter([288074.8449901076, 6247982.515792289])
      view.setResolution(view.getResolution() / 131072.0)
      view.setRotation(45)
    </script>
  </body>
</html>

```

## 2.5 Correction exercice élémentaire 5

### Exercice élémentaire 5

```

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">
    <title>Exercice élémentaire</title>
    <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">

```

```
<style>
  #map {
    width: 600px;
    height: 400px;
  }
</style>
</head>
<body>
  <div id="map"></div>
  <script src="http://ol3js.org/en/master/build/ol.js"></script>
  <script>
    var view = new ol.View2D();
    var map = new ol.Map({
      target: 'map',
      view: view,
      layers: [
        new ol.layer.TileLayer({
          source: new ol.source.OSM()
        })
      ]
    });

    var extent = [287716.5464200208, 288433.14356019435,
      6247743.650078897, 6248221.38150568];
    view.fitExtent(extent, map.getSize());
  </script>
</body>
</html>
```

La fonction `fitExtent` a besoin de connaître les dimensions de la carte en pixels pour être capable de recentrer la vue sur une étendue donnée. La vue est en effet complètement déterminée par un centre, une résolution et une rotation, et elle n'a aucune connaissance des dimensions du rectangle d'affichage dans la page. Dans ces conditions, si on ne lui passe pas des dimensions (largeur et hauteur en pixels), il ne lui est pas possible de recentrer la vue sur une étendue géographique.

## 2.6 Correction exercice élémentaire 6

### *Exercice élémentaire 6*

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">
    <title>Exercice élémentaire</title>
    <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">
    <style>
      #map {
        width: 600px;
        height: 400px;
      }
    </style>
  </head>
  <body>
    <div id="map"></div>
    <script src="http://ol3js.org/en/master/build/ol.js"></script>
```



```

<script>
var map = new ol.Map({
  target: 'map',
  view: new ol.View2D({
    zoom: 17,
    center: [288074.8449901076, 6247982.515792289]
  }),
  layers: [
    new ol.layer.TileLayer({
      source: new ol.source.OSM({
        attributions: [
          new ol.Attribution(
            '<a href="http://www.opencyclemap.org/">OpenCycleMap</a>' ),
            ol.source.OSM.DATA_ATTRIBUTION
          ],
          url: 'http://{a-c}.tile3.opencyclemap.org/landscape/{z}/{x}/{y}.png',
        })
      })
    ],
    renderer: ol.RendererHint.CANVAS
  });
</script>
</body>
</html>

```

## 2.7 Correction exercice élémentaire 7

### Exercice élémentaire 7

```

<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">
  <title>Exercice élémentaire</title>
  <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">
  <style>
    #map {
      width: 600px;
      height: 400px;
    }
  </style>
</head>
<body>
  <div id="map"></div>
  <script src="http://ol3js.org/en/master/build/ol.js"></script>
  <script>
var map = new ol.Map({
  target: 'map',
  view: new ol.View2D({
    zoom: 0,
    center: [0, 0]
  }),
  layers: [
    new ol.layer.TileLayer({
      source: new ol.source.OSM(),

```

```
        brightness: 0.1,  
        contrast: 1.625,  
        hue: -1.25,  
        opacity: 0.5,  
        saturation: 5  
    })  
  ]  
});  
</script>  
</body>  
</html>
```

Il est important de noter ici que les paramètres d’affichage (`opacity`, etc.) sont des paramètres de l’objet `Layer`, et non de l’objet `Source`. La source ne concerne que les données elles mêmes, et pas la façon dont ces données sont affichées.

Pour changer la visibilité de la couche dans la console :

```
> layerCollection = map.getLayers()  
> osmLayer = layerCollection.getAt(0)  
> osmLayer.setVisible(!osmLayer.getVisible())  
> osmLayer.setVisible(!osmLayer.getVisible())
```

`osmLayer.getVisible()` retourne un booléen (`true` ou `false`). `osmLayer.setVisible(!osmLayer.getVisible())` permet donc d’inverser la visibilité.

## 2.8 Correction exercice élémentaire 8

### *Exercice élémentaire 8*

Dans la console :

```
> ol.proj.transform(map.getView().getCenter(), 'EPSG:3857', 'EPSG:4326')
```

```
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="utf-8">  
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">  
    <title>Exercice élémentaire</title>  
    <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">  
    <style>  
      #map {  
        width: 600px;  
        height: 400px;  
      }  
    </style>  
  </head>  
  <body>  
    <div id="map"></div>  
    <script src="http://ol3js.org/en/master/build/ol.js"></script>  
    <script>  
      var center = ol.proj.transform(  
        [2.5878203, 48.8413379], 'EPSG:4326', 'EPSG:3857');  
      var map = new ol.Map({  
        target: 'map',  
        view: new ol.View2D({
```

```

        zoom: 17,
        center: center
    )),
    layers: [
        new ol.layer.TileLayer({
            source: new ol.source.OSM()
        })
    ]
    });
</script>
</body>
</html>

```

## 2.9 Correction exercice élémentaire 9

### Exercice élémentaire 9

Si rien concernant les *controls* n'est spécifié dans l'objet d'options passé au constructeur `ol.Map` alors `ol.Map` crée trois *controls* (les *controls* par défaut) : `ol.control.Attribution`, `ol.control.Logo`, et `ol.control.Zoom`.

```

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">
    <title>Exercice élémentaire</title>
    <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">
    <style>
      #map {
        width: 600px;
        height: 400px;
      }
    </style>
  </head>
  <body>
    <div id="map"></div>
    <script src="http://ol3js.org/en/master/build/ol.js"></script>
    <script>
      var map = new ol.Map({
        target: 'map',
        view: new ol.View2D({
          zoom: 0,
          center: [0, 0]
        }),
        layers: [
          new ol.layer.TileLayer({
            source: new ol.source.OSM()
          })
        ],
        controls: ol.control.defaults({}, [
          new ol.control.ScaleLine({
            units: ol.control.ScaleLineUnits.IMPERIAL
          })
        ])
      });
    </script>
  </body>
</html>

```

```
    </script>
  </body>
</html>
```

## 2.10 Correction exercice élémentaire 10

### *Exercice élémentaire 10*

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="initial-scale=1.0, user-scalable=no, width=device-width">
    <title>Exercice élémentaire</title>
    <link rel="stylesheet" href="http://ol3js.org/en/master/build/ol.css" type="text/css">
    <style>
      #map {
        width: 600px;
        height: 400px;
      }
    </style>
  </head>
  <body>
    <div id="map"></div>
    <script src="http://ol3js.org/en/master/build/ol.js"></script>
    <script>
      var map = new ol.Map({
        target: 'map',
        view: new ol.View2D({
          zoom: 0,
          center: [0, 0]
        }),
        layers: [
          new ol.layer.TileLayer({
            source: new ol.source.OSM()
          })
        ],
        interactions: ol.interaction.defaults({}, [
          new ol.interaction.DragRotateAndZoom()
        ])
      });
    </script>
  </body>
</html>
```