UNESCO-IHE
Institute for Water Education

Upande

SNIEAU

MERPMEDER

Port'Ability
PORTEUR DE SAVOIR-FAIRE

UN GUIDE POUR OPENLAYERS V3

Objectifs

Dans cet exercice, vous allez apprendre à :

- Créer une carte web avec HTML5 et JavaScript (OpenLayers v3)
- Ajouter du contenu spatial (cartes et services de cartes web) à un site web avec Open Layers 3
- Ajouter des contrôles pour manipuler le contenu spatial d'un site web.

Introduction

Les applications cartographiques web sont des sites web avec un contenu cartographique. Une application cartographique web est un site web – cela signifie qu'il est écrit dans un langage interprétable par un navigateur web. Ce langage est appelé HyperText Markup Language (HTML). Le langage JavaScript est utilisé pour autoriser l'utilisateur à interagir avec le site web.

Les applications cartographiques web sont généralement destinées à un groupe d'utilisateurs spécifiques qui détermine la sélection de données spatiales lui-même – c'est-à-dire quel type de données spatiales et quel format sera utilisé. Nous allons utiliser la librairie OpenLayers JavaScript pour mettre le contenu spatial sur l'application cartographique web.

Concepts de base

HTML 5

Qu'est-ce que HTML?
HTML est un langage de balise utilisé pour décrire les documents web (pages web).

- HTML veut dire Hyper Text Markup Language
- Un language de balise (MarkUp language) détermine un ensemble de balises
- Les documents HTML sont décrits par les balises HTML
- Chaque balise HTML tag décrit un contenu différent du document

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<link rel="stylesheet" href="URL" type="text/css">
<style> </style>
<title></title>
<iframe src="URL"></iframe>
</body>
</html>
```

Exemple expliqués

- La balise <!DOCTYPE html> précise que le document est en HTML5
- Le texte entre les balises <html> et </html> décrit le document HTML
- Le texte entre les balises <head> et </head> donne des informations sur le document
- Le lien (link) autorise la référence à une autre page web et librairie source : style et script
- Le texte entre les balises <title> et </title> correspond au titre du document
- Le texte entre les balises <body> et </body> décrit le contenu visible de la page
- Une iframe est utilisée pour afficher une page Web dans une page web

Carte

La composante de base de OpenLayers 3 est la carte (ol.Map). Elle est transmise à un conteneur cible (il s'agit d'une balise div sur la page web qui contient la carte). Toutes les propriétés de la carte peuvent soit être configurées au moment de la création, soit en utilisant une méthode de paramétrage, comme e.g. setTarget()

```
<div id="map" style="width: 100%, height: 400px"></div>
<script>
  var map = new ol.Map({target: 'map'});
</script>
```

View

ol.Map ne prend pas en charge les choses comme le centre, le niveau de zoom et la projection de la carte. Mais ces propriétés sont des propriétés de l'instance ol.View.

```
 map.setView(new ol.View({
    center: [0, 0],
    zoom: 2
  }));
```

Une ol.View a aussi une projection. La projection détermine le système de coordonnées du centre et des unités pour le calcul de résolution cartographiques. Si non-spécifiée, la projection par défaut est Sphérical Mercator (EPSG:3857), avec des mètres comme unités cartographiques.

Source

Pour recevoir les données d'une couche à distance, OpenLayers 3 utilise les sous-classes ol.source.Source. Celles-ci sont disponibles pour des services de mosaïques de cartes gratuits et commerciaux comme OpenStreetMap ou Bing, pour des sources OGC comme WMS ou WMTS, et pour des données vecteur en formats GeoJSON ou KML.

```
var osmSource = new ol.source.OSM();
```

Couche

Une couche est une représentation visuelle de données provenant d'une source. OpenLayers 3 a trois types de couches élémentaires:

- ol.layer.Tile
- ol.layer.Image
- ol.layer.Vector.

ol.layer.Tile est pour les sources de couches qui fournissent des images pré-rendus, des images en mosaïque dans des grilles qui sont organisées par niveau de zoom pour des résolutions spécifiques.
ol.layer.Image est pour les images produites par les serveurs qui sont disponibles pour les extensions et les résolutions arbitraires.
ol.layer.Vector est pour les données vecteur qui sont des images produites par le serveur et qui sont disponibles pour les extensions et les résolutions arbitraires produites par le client

```
var osmLayer = new ol.layer.Tile({source: osmSource});
 map.addLayer(osmLayer);
```

Mettre tout ensemble

Les extraits ci-dessus peuvent être consolidés dans une configuration de carte autonome avec des vues et des couches:

```
<div id="map" style="width: 100%, height: 400px"></div>
<script>
  new ol.Map({
    layers: [
      new ol.layer.Tile({source: new ol.source.OSM()})
    ],
    view: new ol.View({
      center: [0, 0],
      zoom: 2
    }),
    target: 'map'
  });
</script>
```

Prérequis

- Télécharger les fichiers de l'exercice du **Download the SNIEAU Training: OpenLayers v3 Tutorial document**
- Créer un fichier principal OLTutorial

- Créer des sous-fichiers dans OLTutorial: **MapIframe, Task1, Task2, Task3, Task4 & SimpeWebMap**

Publier une carte Iframe

Listing : **MapIframe/iframe.html**

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<link rel="stylesheet" href="http://openlayers.org/en/v3.8.2/css/ol.css" type="text/css">
<style>
#map {
height: 400px;
width: 400px;
}
</style>
<title>SNIEAU Iframe</title>
<iframe src="URL"></iframe>
</body>
</html>
```

URL : Remplacer par une carte publiée dans l'instance de GeoNode
Exemple:

```
<iframe style="border: none;" height="400" width="600"
src="http://192.81.212.100/maps/22/embed"></iframe>
```

Exercice : MapIframe

- Publier une carte GeoNode et remplacer l'**URL.**
- Changer la largeur : **width** et la hauteur : **height** pour l'iframe

Tâche 1

Créer un nouveau fichier et enregistrez-le comme **map.html** sur votre disque dur (emplacement au choix). Copier le contenu du listing 1 (ci-dessous) dans le nouveau fichier, sauvegardez-le et ouvrez le .html dans un navigateur web.

Listing 1:**Task1/map.html**

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<link rel="stylesheet" href="http://openlayers.org/en/v3.8.2/css/ol.css" type="text/css">
<style>
#map {
height: 400px;
width: 400px;
}
</style>
<title>Example OpenLayers3 page</title>
<script src="http://openlayers.org/en/v3.8.2/build/ol.js" ype="text/javascript"></script>
</head>
<body>
<h1>My Map</h1>
<div id="map"></div>
<script type="text/javascript">
var map = new ol.Map({
target: 'map',
layers: [
new ol.layer.Tile({
source: new ol.source.OSM({layer: 'osm'})
})
],
view: new ol.View({
center: [254031,6254016],
zoom: 16
})
});
</script>
</body>
</html>
```

Pour intégrer une carte dans une page web, nous avons besoin de 3 choses :

1. Intégrer la librairie de OpenLayers, qui va nous permettre de mettre du contenu spatial sur une page web :
   Cette ligne intègre la librairie OpenLayers à la page web :

```
<script src="http://openlayers.org/en/v3.8.2/build/ol.js" type="text/javascript"></script>
```

   La première chose est d'intégrer la bibliothèque JavaScript. Techniquement, vous pouvez mettre la référence à une bibliothèque JavaScript n'importe où dans le document .html.

2. Créer un fichier HTML qui sera le conteneur de la carte :
   Définir d'abord le style de la feuille pour notre carte. Nous allons utiliser la feuille de style (CSS) de la cascade standard (standard cascading) définie dans OpenLayers. Nous allons inclure le lien vers ce css dans la balise <head> de notre document :

```
<link rel="stylesheet" href="http://openlayers.org/en/v3.8.2/css/ol.css" type="text/css">
```

   La balise HTML contenant la carte est créée avec . Dans cette balise les propriétés de la carte telles que largeur, hauteur et être controlées par le css. Notre carte a une hauteur de 400 pixels de haut et 400 pixels de large, et cette définition peut aussi être intégrée dans la balise <head> du document .html

```
<style>
#map {
height: 400px;
width: 400px;
}
</style>
```

NOTE : Pour que la carte apparaisse en plein écran, la largeur doit être 100%.

3. Créer une carte simple avec JavaScript en écrivant un script spécifiant les détails de la carte :
   L'étape suivante pour générer notre carte est d'inclure un code d'initialisation. Dans notre cas, nous avons inclus un élément <script> en haut de notre document <body> pour cela.

```
<h1>My Map</h1>
<div id="map"></div>
<script type="text/javascript">
var map = new ol.Map({
target: 'map',
layers: [
new ol.layer.Tile({
source: new ol.source.OSM({layer: 'sat'})
})
],
view: new ol.View({
center: [254031,6254016],
zoom: 16
```

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

Avec du code JavaScript, un objet carte est créé avec une couche à partir d'une tuile OSM en zoomant sur le Bénin.
Regardons notre script d'un peu plus près.

La ligne suivante :

```
var map = new ol.Map({ ... });
```

Crée un objet carte dans OpenLayers. A elle seule, cette commande ne fait rien car il n'y a pas de couches ou d'interactions qui s'y rattachent.

Nous attachons l'objet carte à l'élément , avec l'objet carte a besoin d'un argument 'target'. La valeur est l'identifiant (id) de l'élément : target="map"

La couche, layers: [...] il faut utiliser un tableau pour définir la liste des couches disponibles dans la carte. La première et la seule couche pour notre exemple est une couche tuile :

```
layers: [new ol.layer.Tile({
source: new ol.source.OSM({layer: 'sat'})
})
]
```

Les couches dans OpenLayers 3 sont définies avec un type (Image, Tuile ou Vecteur) qui contient la source. La source est le protocole utilisé pour obtenir les tuiles de la carte. Vous pouvez consulter la liste des sources de couches disponibles ici :
http://openlayers.org/en/v3.8.2/apidoc/ol.source.html

La partie suivante de l'objet carte est la vue (View). La vue permet de spécifier le centre, la résolution et la rotation de la carte. La façon la plus simple de définir une vue est de définir un point central et un niveau de zoom. Remarquez que le niveau de zoom 0 correspond au zoom arrière.

```
view: new ol.View({
center: [0,0],
zoom: 16
})
```

Question?
Quelles sont les coordonnées de Contonou, Benin ? en coordonnées décimales
Exemple : 6.6010193782859 N, 2.241625076858 E

Remarquez que les coordonnées du centre doivent être renseignées dans le système de référence spatiale appelée pseudo-Mercator (EPSG: 3857), qui est la référence d'OpenLayers.

4. Ajout de services de cartographie Web externes à une carte
   Maintenant, nous allons apprendre comment ajouter plusieurs couches à la carte. Dans les exercices précédents, vous avez publié vos propres services de cartographie Web, maintenant nous allons voir comment ajouter ces services à notre carte.
   Pour cet exercice, nous allons créer une nouvelle page web.

Exercice 1:Task1/map.html

- Récupérez les coordonnées du Bénin et le **centre** de la carte. Utilisez map.html
- Augmentez la largeur et la hauteur à plein écran, exemple **width: 100%; height: 100%**

Tâche 2:

Créez un nouveau fichier et enregistrez-le comme map2.html sur votre disque dur. Copier le contenu du listing 2 (ci-dessous) dans le nouveau fichier

Listing 2:**Task2/map.html**

```
<!doctype html>
<!-- Simple exemple de Web Mapping utilisant HTML et OpenLayers 3/JavaScript-->
<html>
<head>
<meta charset="UTF-8">
<!-- definition des styles -->
<link rel="stylesheet" href="http://openlayers.org/en/v3.8.2/css/ol.css" type="text/css">
<style>
.map {
height: 500px;
width: 500px;
}
</style>
<!-- JavaScript libraries used in the application -->
<script src="http://openlayers.org/en/v3.8.2/build/ol.js" type="text/javascript"></script>
<!-- My own OpenLayers 3/JavaScript code -->
<script src="layers.js" type="text/javascript"></script>
<title>OpenLayers 3 Example</title>
</head>
<body onload="init()"> <!-- init() is a JavaScript function defined in a separate file -->
<h2>My Map of Benin</h2>
<div id="map" class="map" style="float:left;"></div>
</body>
</html>
```

Remarquez que l'on intègre le script layers.js dans la section <head> (trouvez la ligne **<script src="layers.js" type="text/javascript"></script>** dans le <head> du document).

Exercice 2:

- Changer le titre pour représenter le nom des ressources et des services publics à afficher
- Changer le style de la carte pour "float:right;"

```
<div id="map" class="map" style="float:right;"></div>
```

- Changer les dimensions de la carte : largeur (width) et hauteur (height)

Tâche 3 :

Créez un nouveau fichier et enregistrez-le comme **layers.js** dans le même dossier que le fichier **Task2/map.html**.
Dans cette tâche, nous allons construire notre layers.js pas à pas. L'ajout d'une couche se réalise en plusieurs étapes :

- Définition de la source de données de la couche
- Définition du comportement initial de la couche
- Ajout de la couche sur la carte

Tâche 3.1 :
Définir la source de données de la couche :
Dans notre exemple, nous allons définir OpenStreetMap comme source de données pour la couche. Ajouter OSM's 'osm' data comme source pour votre couche. Le code pour cela est le suivant :

```
var source1 = new ol.source.OSM({layer: 'osm'});
var osmlayer = new ol.layer.Tile({source: source1});
```

Tâche 3.2 :
Construire la fonction init():
Ajouter le code suivant à votre fichier .js pour définir le comportement initial ('initial behavior') de la carte sur la page web :

```
function init() {
var view = new ol.View({
center: ol.proj.transform([100.50, 13.96], 'EPSG:4326', 'EPSG:3857'),
zoom: 5
});
var map = new ol.Map({
target: 'map',
controls: ol.control.defaults().extend([
new ol.control.ScaleLine({
units: 'metric'
})
]),
view: view,
});
map.addLayer(osmlayer);
}
```

Enfin nous allons écrire un script pour ajouter la couche à la table. Cela peut se faire en ajoutant le script **map.addLayer(osmlayer);** à la fonction **init().**
Le code complet annoté pour la création d'une carte simple contenant une couche (OpenStreetMap) centrée sur le Bénin est le suivant :

Listing 3.2: **Task3/layers.js**

```
// definition de la source de la couche
var source1 = new ol.source.OSM({layer: 'osm'});
var osmlayer = new ol.layer.TileWMS({source: source1});
// init() function referenced in the <body onload=init()... of the application's HTML code

function init() {
var view = new ol.View({
center: ol.proj.transform([100.5, 13.96], 'EPSG:4326', 'EPSG:3857'),
zoom: 5
});
var map = new ol.Map({ // map as a composition of the view and controls
target: 'map', // link to the HTML object in which the map should be displayed
controls: ol.control.defaults().extend([
new ol.control.ScaleLine({
units: 'metric'
})
]),
view: view,
});
// adding layers to the map
map.addLayer(osmlayer);
}
```

Nous pouvons maintenant ajouter d'autres couches - une fois que nous avons zoomé sur le Bénin.

Tâche 3.4 :
Créer une nouvelle couche avec les villes du Bénin :
• l'URL du service est : http://192.81.212.100/geoserver/**<your workspace>**/wms,
NOTE: vous pouvez changer l'URL pour pointer sur votre instance geonode locale, ou **http://www.snieau.bj/**

• le nom de la couche est : **<your workspace>**:**<layer>**
Exemple **geonode:benin_cities**
• le type de server est : **geoserver**
Ajouter le code suivant à votre fichier .js (Note : remplacer '<your workspace>' par le nom de votre propre espace de travail sur le GeoServer où vous publiez vos services web :

```
var source2 = new ol.source.TileWMS({
url: 'http://192.81.212.100/geoserver/<your workspace>/wms',
params: {'LAYERS': '<your workspace>:benin_cities'},
serverType: 'geoserver',
});
var benin_cities = new ol.layer.Tile({
extent:[-1.46821054810656,6.3603727406018,2.94001664084976,11.1303658233031],
source: source2
});
```

NOTE : L'extension définie pour la couche est une extension spatiale dans le système de référence spatiale que vous avez défini pour votre web carte sur GeoServer.
Vous pouvez voir ces valeurs dans votre document WMS. Et biensûr, n'oubliez pas d'ajouter votre nouvelle couche à la carte.

Tâche 3.5:
Intégrer le script : **map.addLayer(benin_cities);** dans la définition de votre fonction init() (après avoir ajouté la couche OSM dans la carte).

Système de référence spatiale dans OpenLayers 3

Le système de référence spatiale par défaut de OpenLayers 3 (SRS) est le Pseudo-Mercator WGS84 (EPSG:3857) (aussi connu comme Spherical Mercator). C'est un système de projection de coordonnées utilisé pour le rendu des cartes dans Google Maps, OpenStreetMap,

OpenLayers 3 propose 2 méthodes pour changer le système de référence : http://epsg.io/3857

1. Ol.proj.transform pour la transformation des coordonnées
2. Ol.proj.transformExtent pour la transformation des extensions spatiales

Tâche 3.6:
Réécrivez votre définition de la couche comme suit:

```
extent:ol.proj.transformExtent([-
1.46821054810656,6.3603727406018,2.94001664084976,11.1303658233031], 'EPSG:4326', 'EPSG:3857',
```

Le script suivant complète notre définition de la couche forêt :

```
var source2 = new ol.source.TileWMS({
url: 'http://localhost:8080/geoserver/<your workspace>/wms',
params: {'LAYERS': '<your workspace>:benin_cities'},
serverType: 'geoserver',
});
var benin_cities = new ol.layer.Tile({
extent: ol.proj.transformExtent ([-
1.46821054810656,6.3603727406018,2.94001664084976,11.1303658233031],'EPSG:4326', 'EPSG:3857'),
source: source2
});
```

Rafraichissez votre carte, vous pouvez maintenant voire une couche des villes en haut de la couche OpenStreetMap.

Exercice 3.6:
- Ajouter une (ou plusieurs) couche WMS Note: extension de la couche
- Ajouter une couche raster: **geonode_mnt_benin**
- Ajouter 2 couches : **raster et vecteur**
- Interchanger les couches WMS : raster et vecteur

Tâche 4 :

Nous allons voir comment ajouter une légende à la carte

Ajouter une légende à la carte
OpenLayers 3 fournit des controles pour interagir avec un contenu spatial dynamique, comme la commande zoom, qui s'affiche par défaut avec un objet 'carte'. L'échelle peut être ajoutée manuellement.
Fortunately, our layers are Web Map Services which, are able to provide their own legend — the **GetLegendGraphic service** operation helps doing this, it returns an image with the layer's legend.

(NOTE: remplacer <your workspace> avec le nom de votre propre espace de travail : **geonode)**

```
<div id="legend" style ="width :250 px; height :200 px;"> <!--placeholder for legend-->
<img src=
"http://192.81.212.100:8080/geoserver/geonode/wms?service=WMS&version=1.1.0&request=GetLegendG
raphic&layer=<your workspace>:benin_cities&format=image/png&width=20&height=20"><br/>
</div > <!-- GetLegendGraphic request to the WMS-->
```

Exercice 4.0:

- Ajouter une légende pour geonode_mnt_benin
- Ajouter une autre couche geonode et sa légende

Tâche 4.1:
OL3 OL3 Maintenant nous avons 2 couches dans notre carte, mais nous n'avons pas décidé laquelle nous souhaitons afficher. Nous allons le faire

Paramétrer la visibilité des couches :
Dans votre fichier **Task4/layer_visibility.js** (à la fin du fichier) écrivez une nouvelle fonction pour définir l'état initial de l'affichage de la couche — le code pour cette fonction est disponible ici :

```
function layerVis (value) { // function invoking the initial status of the layers' visibility
if (value == "osmlayer" ) {
osmlayer.setVisible(document.getElementById("1").checked);
}
else if (value == "benin_cities") {
benin_citieslayer.setVisible(document.getElementById("2").checked);
}
}
```

Dans votre code **Task4/map.html** créez un nouvel espace réservé pour un commutateur de visibilité des couches — Exemple de code ici :

```
<div style="padding:10px;">
<h2> Layer visibility switcher: </h2> <!-- a checkbox activating a function 'layerVis' defined
in a separate .js file (in our case, in the 'layers.js' file) -->
<input id="1" value="osmlayer" type="checkbox" onchange="layerVis(this.value)" checked/>
OSM-OpenStreetMap<br>
<input id="2" value="benin_cities" type="checkbox" onchange="layerVis(this.value)" checked/>
Benin Cities<br>
</div>
```

Exercice 4.1:

- Ajouter une autre couche vecteur, par exemple: **benin_rivers** au **commutateur de visibilité**

Tâche 4.2:

Notre carte est presque prête, nous voulons inclure un autre bel outil améliorera l'utilisation de la carte. Cet outil fournira des informations complémentaires sur les données spatiales affichées dans la carte, il utilisera l'opération **GetFeatureInfo,** tous les services de cartographie web disposent de cet outil.

OpenLayers 3 supporte cette opération, et nous allons voir ici comment l'utiliser:

- Nous avons besoin d'inclure une fonction dans notre fichier .js pour autoriser les requêtes sur les informations d'entité relative à notre couche.
- Nous avons aussi besoin de créer un espace dans notre fichier .html pour afficher cette information.

Ecrivez une nouvelle fonction dans votre fichier **Task4/layer_info.js** pour récupérer les information d'entité pour une couche. Ce code fera partie de la fonction init() — insérer le code juste après la définition de la vue (view):

```
map.on('singleclick', function(evt) { // a click on a map allowing displaying non-spatial
attributes of the feature document.getElementById('info').innerHTML ='';
var viewResolution = /** @type {number} */ (view.getResolution());
var url = source2.getGetFeatureInfoUrl(evt.coordinate, viewResolution, 'EPSG:3857',
{'INFO_FORMAT':'text/html'}
);
if (url) {
document.getElementById('info').innerHTML =
'<iframe with="300px" height="100px" src="' + url + '"></iframe>';
}
}
);
```

Créer un espace dans votre fichier .html pour afficher les informations d'entité — le code est ici :

```
<h5> Click at the map to see additional information... </h5>
<div id="info">
<!-- Attributes of features will be displayed here -->
</div>
```

Exercice 4.2: Finalisation

1. Vérifiez le dossier “**OLTutorial/SimpleWebMap”**
2. Ouvrir dans un navigateur le fichier **map.html**
3. Ouvrir **map.html** en utilisant le bloc-notes (ou un autre éditeur de texte)
    a. Ajouter votre propre carte à l'onglet menu en créant votre propre fichier .html et en remplacer le par "#"

```
<li>
        <a href="#">Map 2</a>
</li>
<li>
        <a href="#">Map 3</a>
</li>
```

4. Utiliser votre propre jeu de données :
    a. Charger un jeu de données
    b. Ajouter un style au jeu de données
    c. Publiez la carte
    d. Utilisez une carte iframe et WMS pour ajouter des couches à votre **SimpleWebMap/map.html**

RESUME

Dans cet exercice, nous avons créé une application cartographique web simple avec HTML5 et OpenLayers 3, maintenant nous sommes capables de :

- Créer un site web simple avec HTML et JavaScript ;
- Ajouter un contenu cartographique (cartes et services cartographiques web) à un site web avec OpenLayers 3 ;
- Ajouter des commandes pour manipuler les données spatiales dans un site web.

QUESTIONS

REFERENCE

1. http://openlayers.org/en/v3.1.1/doc/quickstart.html
2. http://openlayers.org/ol3-workshop/basics/map.html