Dokumentation

Aufgabe 1

Die 3 html Seiten erstellt:

- Grundgerüst eingefügt
- Die 3 Seiten miteinander vernetzt durch Navigationsleiste und Namen vergeben
- Navigationsleiste und Überschriften jeweils gestylet.

Aufgabe 2

Im head der page-1.html Datei, die JS- und CSS-Bibliotheken einbinden:

```
<link rel="stylesheet" href="/libs/ol-v6.4.3/ol.css">

<style>
  #map{
    width: 100%;
    height: 500px;
  }
</style>

<script src='/libs/ol-v6.4.3/ol.js' type='text/javascript'></script>
```

Open-Layer Karte mit einem OSM Hintergrundlayer erzeugt:

```
<h2>GeojSON-Map</h2>

<div id="map" class="map"></div>

<script type="text/javascript">
const map = new ol.Map({
    target: 'map',
    view: new ol.View({


center: ol.proj.fromLonLat([10.450196, 51.769200]),
zoom: 5.5,


      projection: "EPSG:3857"
    }),
     layers: [
       new ol.layer.Tile({
        source: new ol.source.OSM()
      })]
```

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

Bundesländer als Vector-Layer einbinden:

```
const Bundeslaender = new ol.layer.Vector({
  source: new ol.source.Vector({
    url: '/data/bundeslaender.geojson',
    projection: 'EPSG: 3857',
    format: new ol.format.GeoJSON()
      }),
      visible: true,
      title: 'Bundeslaender'
})

map.addLayer(Bundeslaender);
```

Wind-Turbinen als Vector-Layer einbinden:

```
const WindTurbines = new ol.layer.Vector({
  source: new ol.source.Vector({
    url: '/data/numberOfWindTurbinesByLand.geojson',
    projection: 'EPSG: 3857',
    format: new ol.format.GeoJSON()
      }),
      visible: true,
      title: 'WindTurbines'
})

map.addLayer(WindTurbines);
```

Style-Funktion, welche die Fläche der dargestellten Kreise proportional zur Anzahl (value) skaliert:

```
let min_radius = 1;
let min_value = 16;


function calculateRadius(value){
  if (min_value > value) {
    min_value = value;
  }
  let radius = min_radius = Math.sqrt(value / min_value)
  if (min_radius > radius) {
      min_radius = radius;
  }
```

```
return radius
}

const WindTurbines = new ol.layer.Vector({
  source: new ol.source.Vector({
    url: '/data/numberOfWindTurbinesByLand.geojson',
    projection: 'EPSG: 3857',
    format: new ol.format.GeoJSON()
  }),
  visible: true,
  title: 'WindTurbines',
  style: function (feature){
     return new ol.style.Style({
        image: new ol.style.Circle({
          radius: calculateRadius(feature.getProperties().value),
          fill: new ol.style.Fill({
            color:"red"
          })
        })
     })
  }
})


map.addLayer(WindTurbines);
```

Aufgabe 3

Open-Layers Bibliotheken im head der page-2.html Datei einbinden (CSS + JS)

```
<link rel="stylesheet" href="/libs/ol-v6.4.3/ol.css">

<style>
  #map2{
    width: 100%;
    height: 500px;
  }
</style>

<script src='/libs/ol-v6.4.3/ol.js' type='text/javascript'></script>
```

Im body der page-2.html Datei: div Container erstellt und Überschrift eingefügt und diese gestylet:

```
<div id="map2" class="map2"></div>

<script type="text/javascript">
```

```
</script>
```

Layer „osmlanduse:osm_lulc_combined_osm4eo“ von OSMLanduse als Tile-Layer mit XYZ-Source eingebunden:

```
<script type="text/javascript">

const map = new ol.Map({
    target: 'map2',
    view: new ol.View({

center: ol.proj.fromLonLat([20.170900, 50.090984]),

zoom: 5,

    }),
});

const OSMLanduse = new ol.layer.Tile({
  source: new ol.source.XYZ({
     tileUrlFunction:  function(coordinate) {
      return
'https://maps.heigit.org/osmlanduse/tiles/osmlanduse:osm_lulc_combined_osm4eo/web
mercator/' + coordinate[0] + '/' +
      coordinate[1] + '/' + coordinate[2] + '.png';
      }
  })
})

map.addLayer(OSMLanduse);


</script>
```

Aufgabe 4

Open-Layers Bibliotheken im head der page-2.html Datei einbinden (CSS + JS)

```
<link rel="stylesheet" href="/libs/ol-v6.4.3/ol.css">

<style>
  #map3{
    width: 100%;
    height: 500px;
  }
</style>
```

```
<script src='/libs/ol-v6.4.3/ol.js' type='text/javascript'></script>
```

Im body der page-2.html Datei: div Container erstellt und Überschrift eingefügt und diese gestylet:

```
<h2>Karte mit Daten einer lokalen CSV Datei</h2>


<div id="map3" class="map3"></div>

<script type="text/javascript">

</script>
```

OpenLayers Karte mit einem OSM Hintergrundlayer:

```
<h2>GeojSON-Map</h2>

<div id="map3" class="map3"></div>

<script type="text/javascript">
let basemap = new ol.Map({
    target: 'map',
    view: new ol.View({
      center: ol.proj.fromLonLat([10.450196, 51.769200]),
      zoom: 5.5,
      projection: "EPSG:3857"
    }),
     layers: [
       new ol.layer.Tile({
        source: new ol.source.OSM()
      })]
});
</script>
```

HTML Element <input> vom Typ „file“ im body der page-3.html Datei eingefügt, um dem Nutzer die Möglichkeit zu geben eine lokale Datei (CSV) von seinem Rechner auszuwählen:

```
<input type="file" onchange="readfile(this.files)">
```

CSV Datei des Nutzers (Landeshauptstädte) mit der Bibliothek PapaParse geparset:

Im head:

```
<script type='text/javascript' src='/libs/papa-v5.0.2/papaparse.min.js'></script>
```

Im body:

```
<script type="text/javascript">
```

```
function readfile(filelist) {
  let csvFile = fileList[0];

  Papa.parse(csvFile, {
    header: false,
    skipEmptyLines: true,
    error: onError,
    complete: onComplete
  })
}
</script>
```

Aufgabe 4 e) und f): Habe versucht eine for-Schleife zu generieren, um die einzelnen Zeilen mit den Koordinaten der csv-Datei einzulesen und daraus die Geometrien zu erstellen. Daraufhin folgen die Schritte zur Erstellung und des Stylings der Geometrien. Letztendlich wurde ein vector-layer erstellt und in die basemap integriert:

```
<script type="text/javascript">

function readFile(fileList) {
  let csvFile = fileList[0];

   Papa.parse(csvFile, {
    header: false,
    skipEmptyLines: true,
    error: onError,
    complete: onComplete
  })
}

function onComplete(result) {
  let data = result.data;

  let coordinates = [];

  let numbers = 16;

  for (let i = 0; i < numbers; i++){

    let row = data[i,];

    let xcoordinate[i] = row[1]
    let ycoordinate[i] = row[2]
  }
```

```
}


let coordinates = ol.proj.fromLonLat([xcoordinate, ycoordinate])

let vectorGeom = new ol.geom.Point(coordinates);

let vectorFeature = new ol.Feature({
    name: "Staedte",
    geometry: vectorGeom
  });

let vectorSource = new ol.source.Vector({
  features: [vectorFeature]
});

let vectorStyle = new ol.style.Style({
  image: new ol.style.Circle({
    fill: new ol.style.Fill({color: "rgb(255, 0, 0"}),
    radius: 6
  })
});

let vectorLayer = new ol.layer.Vector({
  source: vectorSource,
  style: vectorStyle
});

basemap.addLayer(vectorLayer);


function onError(error) {
  alert(JSON.stringify(error, null,2));
}

</script>
```