DataLayer

2021년 11월 30일

2021년 2학기 소프트웨어시스템실습

담당교수: 최성종

임현준

jwalkin0420@office.uos.ac.kr

서울시립대학교 전자전기컴퓨터공학부

차 례

[1. 코드 분석](#_Toc54003187)

[1.1. Map 객체 생성](#_Toc54003188)

1.2. JSON 데이터를 Ajax로 로드

1.3. startDataLayer

1.4. Listener

2. 추가 설명

2.1. Ajax

2.2. JSON

2.3. GeoJSON

3. 실습 코드

# 코드 분석

## Map 객체 생성

텍스트이(가) 표시된 사진

자동 생성된 설명첫번째로 Naver Map api를 사용하기 위하여 위와 같이 객체를 선언한다. Naver Map의 map class에는 지도의 형식, 초기 위치, 확대 레벨 등을 포함한다.

## JSON 데이터를 Ajax로 로드

텍스트이(가) 표시된 사진

자동 생성된 설명json 타입의 데이터를 ajax 방식으로 로드한다. 데이터들을 로드하는 경로가 되는 주소를 위에 나타나 있는 HOME\_PATH에 저장하고 ajax로 수신하여 데이터레이어를 실행한다.

## startDataLayer

텍스트이(가) 표시된 사진

자동 생성된 설명

위에서 나타난 startDataLayer에서 GeoJSON 데이터를 등록하여 지도의 스타일을 설정할 수 있고 지도의 영역 또한 표시할 수 있다. 여기서 setStyle은 초기의 설정을 지정한다.

## Listener

텍스트이(가) 표시된 사진

자동 생성된 설명위와 같이 addListener를 사용하여 이벤트가 발생하게 될 경우, 함수를 실행시킨다. 이 때, mapEvent라는 객체를 사용하여 이벤트를 실행시키게 된다. 위의 경우에는 click, double click, mouse over, mouse out로 4가지 경우가 있으며 첫번째, click이 포함된 listener는 객체가 클릭되는 경우 isColorful을 참으로 받아들여 객체가 가지고 있던 color property를 표시하게 된다.

두번째, double click이 포함된 listener는 더블 클릭한 객체의 bound로 지도가 이동, 확대된다.

세번째, mouse over가 포함된 listener는 마우스 포인터가 객체 위로 올라왔을 때 외곽선 굵기가 8로 바뀌어 두꺼워지게 되며 만약 올라가진 위치에 어떠한 아이콘이 존재하면 해당 아이콘이 바뀌게 된다.

네번째, mouse out이 포함된 listener는 마우스 포인터가 객체 밖으로 이동했을 때 위에서 실행됐던 동작이 원래대로 다시 돌아오게 된다.

# 추가 설명

## Ajax

Ajax은 Asynchronous JavaScript And XML의 약자로 데이터를 이동하고 화면을 구성할 때 웹 화면을 갱신하지 않고 필요한 데이터를 서버로 보내고 가져오는 방법이다. 화면 갱신이 없기 때문에 매우 편리하고 빠르게 작업이 가능하다. Ajax는 비동기식으로 데이터를 전송하며 서버로 데이터를 요청하고 응답을 기다리는 동안 웹은 자신의 다른 업무를 진행하고 응답이 오면 그 후 작업을 진행한다. Ajax 통신에서 데이터를 전송하는 형식은 크게 CSV, JSON, XML 형식으로 나눌 수 있다. 다음은 JavaScript의 간단한 구현 예제이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

## JSON

JSON은 JavaScript의 객체 형태로 데이터를 전송하는 형식이다. 가독성이 좋고 용량이 적지만 데이터 양이 많으면 분석 속도가 떨어진다.  “[속성-값” 쌍](https://ko.wikipedia.org/w/index.php?title=%EC%86%8D%EC%84%B1-%EA%B0%92_%EC%8C%8D&action=edit&redlink=1) 또는 "키-값” 쌍으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 [개방형 표준](https://ko.wikipedia.org/wiki/%EA%B0%9C%EB%B0%A9%ED%98%95_%ED%91%9C%EC%A4%80) 포맷이다. 비동기 브라우저/서버 통신 ([AJAX](https://ko.wikipedia.org/wiki/Ajax))을 위해, 넓게는 [XML](https://ko.wikipedia.org/wiki/XML)([AJAX](https://ko.wikipedia.org/wiki/Ajax)가 사용)을 대체하는 주요 데이터 포맷이다. 다음은 JSON의 형태이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

## GeoJSON

GeoJSON은 위치정보를 갖는 점을 기반으로 체계적으로 지형을 표현하기 위해 설계된 개방형 공개 표준 형식이다. 이것은 [JSON](https://ko.wikipedia.org/wiki/JSON)을 사용하는 [파일 포맷](https://ko.wikipedia.org/wiki/%ED%8C%8C%EC%9D%BC_%ED%8F%AC%EB%A7%B7)이다. [지리좌표계](https://ko.wikipedia.org/wiki/%EC%A7%80%EB%A6%AC%EC%A2%8C%ED%91%9C%EA%B3%84)의 점을 기반으로 [Geocoding](https://ko.wikipedia.org/wiki/Geocoding)된 지형지물(주소 및 위치), 라인스트링(거리, 고속도로 및 경계등 정보를 담고 있는 문자열) 또는 [폴리라인](https://ko.wikipedia.org/wiki/%EC%84%A0%EB%B6%84), 다각형(국가, 도시, 토지) 및 이러한 유형의 여러 부분으로 구성된 모음을 특징으로 한다. 다음은 GeoJSON의 예제이다.



# 실습 코드

<!DOCTYPE html>

<html>

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0, user-scalable=no">

    <title>간단한 지도 표시하기</title>

    <!-- jquery -->

    <script src="https://ajax.googleapis.com/ajax/libs/jquery/3.6.0/jquery.min.js"></script>

    <!-- naver map -->

    <script type="text/javascript" src="https://openapi.map.naver.com/openapi/v3/maps.js?ncpClientId=sg704x31py"></script>

</head>

<body>

<div id="map" style="width:100%;height:400px;"></div>

<div id="table"></div>

<script>

var HOME\_PATH = "https://navermaps.github.io/maps.js/docs",

    urlPrefix = HOME\_PATH +'/data/region',

    urlSuffix = '.json',

    regionGeoJson = [],

    loadCount = 0;

var map = new naver.maps.Map(document.getElementById('map'), {

    zoom: 7,

    mapTypeId: 'normal',

    center: new naver.maps.LatLng(36.4203004, 128.317960)

});

// 지도가 로드된 다음에 일회성으로 실행되는 이벤트

naver.maps.Event.once(map, 'init\_stylemap', function () {

    for (var i = 1; i < 18; i++) {

        var keyword = i +'';

        if (keyword.length === 1) {

            keyword = '0'+ keyword;

        }

        // 01, 02, 03 생성

        $.ajax({

            url: urlPrefix + keyword + urlSuffix,

            success: function(idx) {

                return function(geojson) {

                    regionGeoJson[idx] = geojson;

                    loadCount++;

                    // 17개 다 받아오면 그리기 시작

                    if (loadCount === 17) {

                        startDataLayer();

                    }

                }

            }(i - 1)

        });

    }

});

var tooltip = $('<div style="position:absolute;z-index:1000;padding:5px 10px;background-color:#fff;border:solid 2px #000;font-size:14px;pointer-events:none;display:none;"></div>');

tooltip.appendTo(map.getPanes().floatPane);

function startDataLayer() {

    map.data.setStyle(function(feature) {

        // 기본 라인

        var styleOptions = {

            fillColor: '#ff0000',

            fillOpacity: 0.0001,

            strokeColor: '#ff0000',

            strokeWeight: 2,

            strokeOpacity: 0.4

        };

        // 마우스 올라갔을 떄 스타일

        if (feature.getProperty('focus')) {

            styleOptions.fillOpacity = 0.6;

            styleOptions.fillColor = '#0f0';

            styleOptions.strokeColor = '#0f0';

            styleOptions.strokeWeight = 4;

            styleOptions.strokeOpacity = 1;

        }

        return styleOptions;

    });

    // 제가 받아온 Geojson파일을 map.data

    regionGeoJson.forEach(function(geojson) {

        map.data.addGeoJson(geojson);

    });

    map.data.addListener('click', function(e) {

        var feature = e.feature;

        // 여기서 다른 선택된 지역 있는지 지워줘야 됩니당

        // 지워주는 방법은

        // map.data.\_features 를 foreach 하면서 focus 끄시면 됩니다.

        if (feature.getProperty('focus') !== true) {

            document.getElementById("table").innerHTML = feature.getProperty('area1');

            getLocationMessage(feature.getProperty('area1'))

        } else {

            feature.setProperty('focus', false);

        }

    });

    map.data.addListener('mouseover', function(e) {

        // 이름을 받아

        var feature = e.feature,

            regionName = feature.getProperty('area1');

            // 강원도

        // 지역명 뜨는거

        tooltip.css({

            display: '',

            left: e.offset.x,

            top: e.offset.y

        }).text(regionName);

        map.data.overrideStyle(feature, {

            fillOpacity: 0.6,

            strokeWeight: 4,

            strokeOpacity: 1

        });

    });

    map.data.addListener('mouseout', function(e) {

        // 툴팁 사라지게 하고

        tooltip.hide().empty();

        // 스타일 원상복귀시키는

        map.data.revertStyle();

    });

}

function getLocationMessage(location\_name){

    var xhr = new XMLHttpRequest();

    var url = 'http://apis.data.go.kr/1741000/DisasterMsg4/getDisasterMsg2List';

    var queryParams = '?' + encodeURIComponent('serviceKey') + '='+'oaXeVfWtlmMq%2FMVsfEXxkMocbaeAD88XfN9GtwY6NWMbMVLiQFrB0%2F2IbNYCBqyqZJiCUzCe8cIK2Asx3QYHHg%3D%3D'; /\*Service Key\*/

    queryParams += '&' + encodeURIComponent('pageNo') + '=' + encodeURIComponent('1');

    queryParams += '&' + encodeURIComponent('numOfRows') + '=' + encodeURIComponent('10');

    queryParams += '&' + encodeURIComponent('type') + '=' + encodeURIComponent('xml');

    queryParams += '&' + encodeURIComponent('create\_date') + '=' + encodeURIComponent('2021/06/22 00:00:00');

    queryParams += '&' + encodeURIComponent('location\_name') + '=' + encodeURIComponent(location\_name);

    xhr.open('GET', url + queryParams);

    xhr.onreadystatechange = function () {

        if (this.readyState == 4) {

            let xml = this.responseXML;

            let tableDiv = document.getElementById("table");

            let locTable = document.createElement("table");

            locTable.style.border = "1px solid";

            locTable.style.borderCollapse = "collapse";

            let header = locTable.createTHead();

            let headRow = header.insertRow();

            let headLoc = headRow.insertCell(0);

            let headMsg = headRow.insertCell(1);

            headLoc.style.border = "1px solid"

            headLoc.style.backgroundColor = "lightgray"

            headMsg.style.border = "1px solid"

            headMsg.style.backgroundColor = "lightgray"

            headLoc.innerHTML = "지역";

            headMsg.innerHTML = "메시지";

            let rows = xml.getElementsByTagName("row");

            let locationNames = xml.getElementsByTagName("location\_name");

            let msgs = xml.getElementsByTagName("msg");

            for(i=0;i<rows.length;i++){

                let row = locTable.insertRow();

                let cellLoc = row.insertCell(0);

                let cellMsg = row.insertCell(1);

                cellLoc.style.border = "1px solid"

                cellMsg.style.border = "1px solid"

                if(i%2==1){row.style.backgroundColor = "lightGray"}

                cellLoc.innerHTML = locationNames[i].childNodes[0].nodeValue;

                cellMsg.innerHTML = msgs[i].childNodes[0].nodeValue;

            }

            tableDiv.appendChild(locTable);

        }

    };

    xhr.send('');

}

</script>

</body>

</html>

사용한 클라스 : naver.maps.Map / naver.maps.LatLng / naver.maps.Event