코드 분석

2021년 11월 30일

2021년 2학기 소프트웨어시스템실습

담당교수: 최성종

이름 이미리내

dkfs1324@office.uos.ac.kr

서울시립대학교 전자전기컴퓨터공학부

차 례

[1. 코드 분석 1](#_Toc54003187)

[1.1. 예제 분석 1](#_Toc54003188)~5

1.2. 지난주 실습 분석…………………………………………………….6~8

1. 코드 분석
   1. 예제 분석
2. GeoJSON 로드

-[GeoJSON](https://navermaps.github.io/maps.js.ncp/docs/global.html" \l "GeoJSON) 형식의 데이터를 로드 하려면 addGeoJson 메서드를 사용한다. 기본적으로 모든 [Map](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Map.html) 객체는 data 객체를 포함한다. 따라서, map.data.addGeoJson 메서드를 사용한다.

-$ajax 부분에서는 원하는 url에서 원하는 데이터타입으로 불러온 경우 성공했을 때 startDataLayer 함수에 내용을 전달함.

-전달한 내용을 geojson파일에 저장하는 형식으로 진행됨.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. GeoJSON 데이터 스타일 지정

-StyleOptions 객체 리터럴로 스타일 지정

: [StyleOptions](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Data.html#~StyleOptions) 객체 리터럴에서 설정한 스타일은 모든 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체에 적용된다. 단, 각 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체의 기하 구조(geometry) 타입에 따라 적용되는 범위는 다르다. 예를 들어, icon 스타일은 [Marker](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Marker.html) 오버레이 타입인 Point, MultiPoint 기하 구조에만 영향을 준다.

-setStyle이라는 메소드를 사용하여 실행 시 pin으로 구성되어 있는 스트로크와 내부의 초기 색을 결정해준다.

텍스트이(가) 표시된 사진

자동 생성된 설명

-StylingFunction 함수로 스타일 지정

: [StylingFunction](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Data.html#~StylingFunction) 함수를 이용하면 각 형상 정보([Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체)의 속성이나 특성에 따라 다른 스타일을 지정할 수 있다.

-color라는 변수 지정 후, red로 저장했으며, feature라는 객체와 getProperty라는 메소드를 이용하여 계속 빨간 색으로 유지 되고, 후에 다시 돌아오는 색도 빨간 색이 되도록 설정 하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 지정된 스타일 재정의

- 앞서 설명한대로 setStyle 메서드는 [Data](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Data.html) 레이어의 모든 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체에 영향을 미친다(이를 전역 스타일이라고 함). 특정 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체에 특별한 스타일을 적용하고자 할 때는 overrideStyle 메서드를 사용한다. 이 메서드는 전역 스타일에 추가로 적용된다. 재정의한 스타일을 이전 스타일로 다시 복원하려면 revertStyle 메서드를 사용합니다. 이 메서드는 특정 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체에 사용할 수 있지만, 인수를 생략하면 모든 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체의 스타일을 복원한다.

-addListener라는 메소드를 사용하여 마우스 위치에 따라 function(e)를 활용한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. KML 로드

- [KML](https://navermaps.github.io/maps.js.ncp/docs/global.html#KML) 형식의 데이터를 로드 하려면 addKml 메서드를 사용한다. 기본적으로 모든 [Map](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Map.html) 객체는 data 객체를 포함한다. 따라서, map.data.addKml 메서드를 사용한다.

-map 안에 string을 이용하여 지도를 불러왔을 때의 초기 상황을 설정하고 있다.

-그래서 마지막 addKml 메소드를 사용하여 데이터를 로드 하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. GPX 로드

- [GPX](https://navermaps.github.io/maps.js.ncp/docs/global.html#GPX) 형식의 데이터를 로드 하려면 addGpx 메서드를 사용한다. 기본적으로 모든 [Map](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Map.html) 객체는 data 객체를 포함한다. 따라서, map.data.addGpx 메서드를 사용한다.

-위와 같이 KML이 아니라 GPX를 로드 하기 위해 addGPX 메소드를 이용하였다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 데이터 레이어 이벤트 활용

- [Data](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Data.html) 레이어는 각 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체의 UI 인터랙션과 속성 변경 등에 대한 이벤트와 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체의 추가/삭제에 대한 이벤트를 발생시킨다. 이벤트에 관한 자세한 내용은 [Data](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Data.html) 레이어의 Events 내용을 참고한다. 밑의 코드는 마우스 포인터가 기하 구조의 경계에 들어올 때와 벗어날 때 각 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체의 스타일을 변경하고, 특정 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체 클릭 시 해당 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체의 속성값을 이용해 스타일을 반영하며, 더블 클릭 시 [Feature](https://navermaps.github.io/maps.js.ncp/docs/naver.maps.Feature.html) 객체의 좌표 경계를 이용해 지도를 이동시킨다.

텍스트이(가) 표시된 사진

자동 생성된 설명

1.2. 지난주 실습 분석

1) XML 파일 로드

-myFunction이라는 함수를 만들어서 해당 함수에 테이블에 지역 이름과 문자 내용을 넣는 구문이다.

-document라는 객체를 이용하여 table을 출력할 수 있도록 하였음.

-getElementById라는 메소드를 사용하여 메소드의 파라미터에 선택하고자 하는 태그의 id 속성값을 문자열로 전달해 줌.

-innerHTML은 프로퍼티로 엘리먼트 내에 포함된 HTML 또는 XML 마크업을 가져오거나 설정함.

텍스트이(가) 표시된 사진

자동 생성된 설명

2) 클래스 Map

-오브젝트 타입의 value를 갖는 클래스.

-전달받은 내용을 다음에 해당하는 조건에 맞추어 표현함.

텍스트이(가) 표시된 사진

자동 생성된 설명

3) 이벤트 실행

-Event라는 프로퍼티를 통해 이벤트를 실행하는 데, Once라는 조건이 붙으면서 실행된다.

-Once로 인해 지도가 로드 된 이후 일회성으로 이벤트가 진행된다.

텍스트이(가) 표시된 사진

자동 생성된 설명

4) 마우스가 지도 위에 올라간 경우, 테이블 로드

-document 객체에서는 getElementById라는 메소드를 사용할 수 있다. getElementById는 HTML 요소를 id 속성을 이용해 찾을 때 사용한다.

텍스트이(가) 표시된 사진

자동 생성된 설명