소프트웨어시스템실습

2021년 11월 30일

2021년 2학기 소프트웨어시스템실습

담당교수: 최성종

김민호

2017440018@office.uos.ac.kr

서울시립대학교 전자전기컴퓨터공학부

차 례

[1. 네이버 데이터 레이어 예제 1](#_Toc89191157)

[1.1. 헤드(3~20) 2](#_Toc89191158)

[1.2. 바디 시작 및 맵 생성(21~40) 2](#_Toc89191159)

[1.3. startDataLayer 함수(41~81) 5](#_Toc89191160)

[2. 광역지자체별 재난문자 현황 8](#_Toc89191161)

[2.1. 헤드: 기본적인 정보 및 api 추출(3~22) 9](#_Toc89191162)

[2.2. 재난 문자 받아 오기(24~69) 10](#_Toc89191163)

[2.3. 바디 시작 및 지도 생성(71~113) 12](#_Toc89191164)

[2.4. 지도에 데이터 레이어 출력(115~142) 14](#_Toc89191165)

[2.5. 이벤트 리스너 등록(144~180) 15](#_Toc89191166)

네이버 데이터 레이어 예제



전체적인 흐름으로는, 처음에 HTML 파일을 열면 위와 같은 지도가 나타난다. 데이터 레이어인 NAVER의 각 글자와 아래 물결모양의 선들, 지도의 파란색 핀들이 지도 위로 생성된다. 각 데이터 레이어들은 마우스를 올리거나 빠져나오고, 클릭 또는 더블클릭 할 때 마다 이벤트를 발생한다. 마우스를 올리면 글자나 선이 굵어지고, 클릭하면 색이 변하며, 더블 클릭하면 해당 데이터 레이어를 중심으로 화면이 확대되면서 이동한다.

헤드(3~20)

텍스트이(가) 표시된 사진

자동 생성된 설명

4~9줄까지는 emmet으로 인해 자동 생성된 부분으로, html파일의 정보를 나타낸다.

10줄은 페이지의 타이틀이다.

12~16줄은 Naver api를 사용하기 위한 부분이고, 18~19줄은 jQuery를 이용하기 위한 부분이다.

바디 시작 및 맵 생성(21~40)



22줄에서는 지도를 불러와 화면에 표시할 문단인 div를 만들어 주었다. 폭은 화면의 50%를 차지하고, 높이는 800px이다.

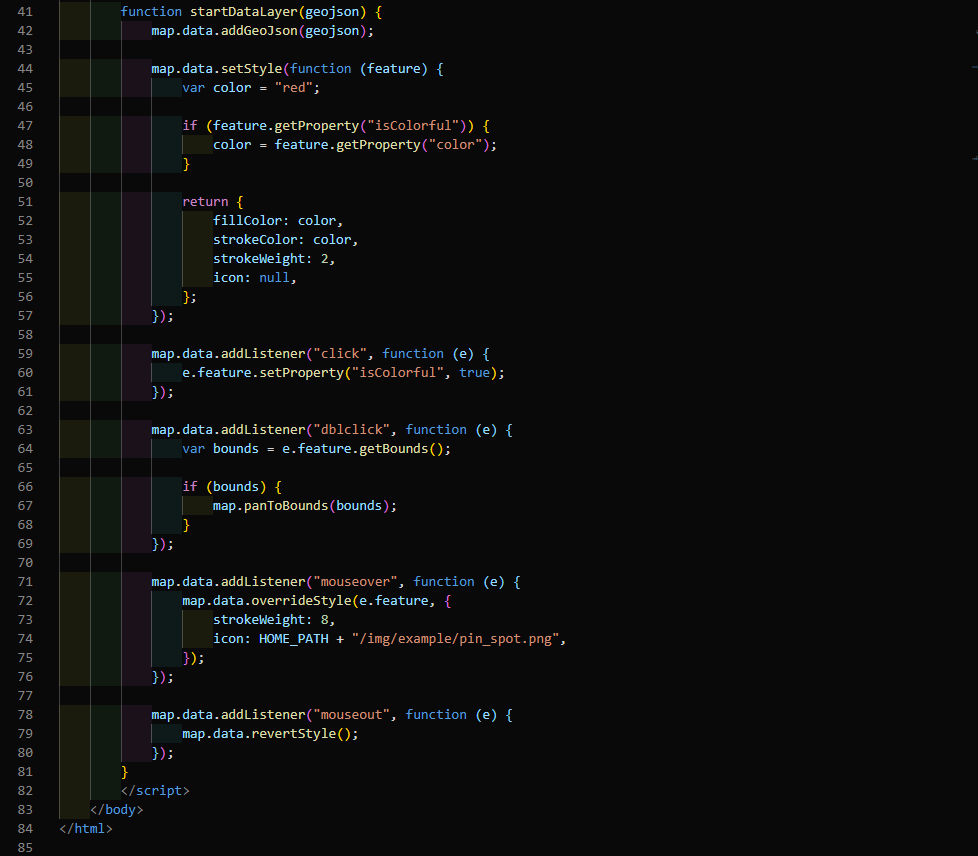
25줄은 변수 HOME\_PATH를 선언한다. 뒤에 나올 함수들이 네이버 사이트로부터 여러 정보를 받아오는데, 불러 올 주소를 저장하는 변수이다.

27~31줄은 실제 불러오게 될 지도 객체를 나타낸다. naver.maps라는 Name-space에서 Map 클래스를 가져와 map이라는 객체를 선언한다. Map 클래스는 애플리케이션에서 지도 인스턴스를 정의한다. 이 객체를 생성함으로써 DOM요소에 지도를 삽입할 수 있다. Map 객체 생성시 인자는 2개이다. 첫 번째는 mapDiv로, 지도를 삽입할 HTML요소 또는 HTML요소의 id이다. 위에서 만들어 놓은 id가 map인 div에 지도를 삽입할 것이므로 이 element를 넣어준다. 두 번째 인자는 mapOptions이다. naver.maps에 있는 MapOptions 객체이다. 지도를 정의하는 옵션을 나타낸다. background, baseTileOpacity, bounds, center, zoom, mapTypeId 등의 프로퍼티들이 있다. 이 중 코드에 사용된 zoom은 지도의 초기 줌 레벨, mapTypeId는 지도의 초기 지도 유형 id, center는 지도의 초기 중심 좌표를 의미한다. 여기서 center를 표현하기 위해 naver.maps의 LatLng 클래스를 사용하였다. LatLng클래스는 위/경도 좌표를 정의하는 클래스로, MapOption클래스의 center 프로퍼티는 LatLng클래스 형식이 필요하다.

33~39줄은 대상 객체에서 한 번만 이벤트 알림을 받아 핸들러를 호출하는 리스너를 등록하는 함수 Event.once()이다. 첫 번째 인자 target은 이벤트가 발생하는 대상 객체이다. 두 번째 인자 eventName은 이벤트의 이름이다. target인 map은 스타일맵이 초기화되면 init\_stylemap이라는 이벤트를 발생시킨다. 세 번째 인자 listener는 이벤트 리스너이다. 이 코드에서는 이벤트가 발생하면 ajax를 요청한다. HOME\_PATH + "/data/naver.json"라는 url에 data-Type으로 json이 넘어오기를 요청한다. ajax 요청이 성공하면 startDataLayer 함수를 실행하게 된다. 즉, map 객체에서 스타일맵이 초기화되면 이벤트가 발생하여 ajax를 네이버에 요청한다. 그 결과로 json이 오면, startDataLayer함수가 실행되는 내용이다.

이때, 전달되는 파일은 GeoJson파일이다. 길이가 9인 Featurecollection으로, 다각형, 선, 다중 선, 점, 다중 점에 대한 정보를 담고 있다.

startDataLayer 함수(41~81)



44줄에서는 전달받은 GeoJson 객체를 map 객체에 추가한다. map.data는 Data 클래스의 객체이고 Data클래스의 addGeoJson()함수는 GeoJson 형식의 지리 공간 데이터를 추가하는 메소드이다. 추가할 GeoJson객체가 인자로 들어오면 Data 객체의 features에 GeoJson 객체들이 추가된다.

46~59줄의 setStyle() 메소드는 GeoJson 지리 공간 데이터의 표현 스타일을 설정한다. 데이터를 표현하는 스타일은 StyleOptions 객체 리터럴 또는 스타일을 처리하는 StylingFunction 함수를 이용해 설정할 수 있다. 이 코드의 경우 StylingFunction 함수를 이용하여 반환값으로 StyleOptions 객체 리터럴을 반환하고 있다.

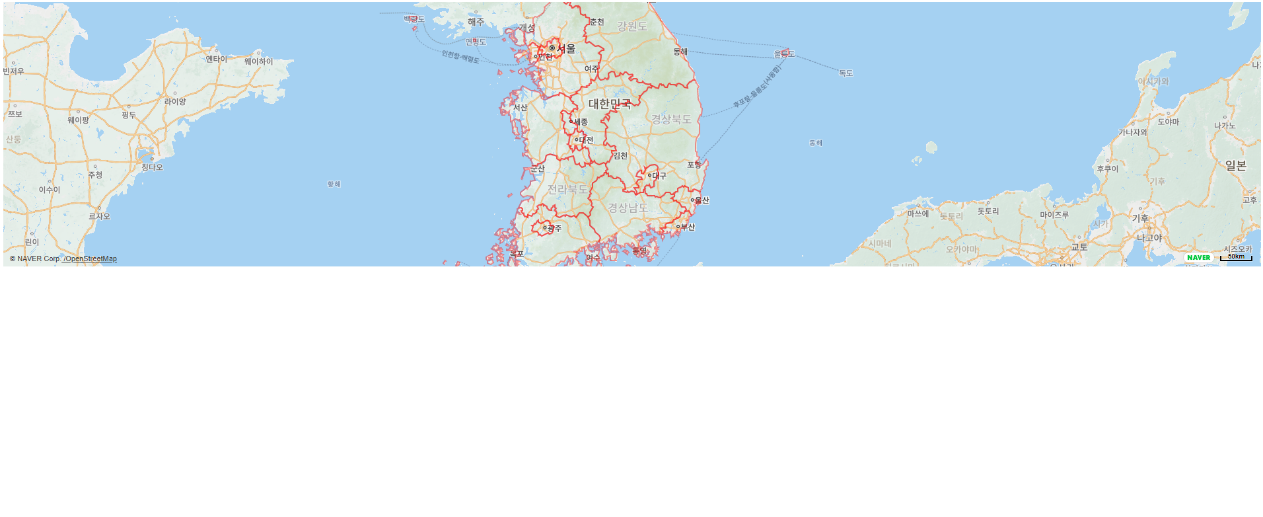
61~63줄의 addListener() 메소드는 Data객체가 상속하는 KVO 클래스로부터 선언되었다. 현재 객체의 이벤트 알림을 받아 핸들러를 호출하는 리스너를 등록하는 함수이다. 첫 번째 인자로 이벤트 이름을 받는데, 이 경우 click이므로 객체를 클릭했을 때 이벤트가 발생한다. 두 번째 인자로 이벤트 리스너를 등록한다. Data 클래스의 프로퍼티 중 feature 프로퍼티는 Feature 클래스의 객체이다. Feature 클래스는 FeatureCollection 타입의 형상 정보 모음에서 Feature 타입의 개별 GeoJSON 객체를 추상화한다. 이 클래스의 setProperty() 메소드는 지정한 이름의 프로퍼티 값을 설정하는 메소드이다. 이 코드의 경우, isColorful 값을 true로 바꾸는 것이다.

65~71줄의 addListener() 함수는 객체를 더블 클릭했을 때의 이벤트를 등록하고 있다. feature의 getBounds()메소드는 Feature 객체의 좌표 경계를 반환하는 메소드이다. 이를 변수 bounds에 저장한다. 이후 bounds값이 반환되면 map 객체의 panToBounds() 메소드를 수행한다. panToBounds() 메소드는 지정한 좌표 경계를 포함하는 위치로 지도를 부드럽게 이동하는 메소드이다. 즉, 객체를 더블 클릭하면 해당 객체의 좌표 경계를 받아와 지도를 그 경계를 포함하는 위치로 부드럽게 이동시킨다.

73~78줄의 addListener() 함수는 객체에 마우스가 올라갔을 때의 이벤트를 등록하고 있다. data 객체의 overrideStyle() 메소드는 적용된 데이터 표현 스타일을 재정의하는 메소드이다. 첫 번째 인자인 e.features의 스타일을 두 번째 인자인 StyleOptions 객체를 이용해 재정의한다. 이 코드의 경우, strokeWeight값을 8로 주어 다각형이나 선들이 굵어지게 되고, icon값에 이미지 파일을 넣어 원래의 파란색 화살표가 아니라 검정색 화살표가 나오도록 하게 된다.

80~82줄의 addListener() 함수는 객체에서 마우스가 빠져나갔을 때의 이벤트를 등록하고 있다. data 객체의 revertStyle() 메소드는 재정의된 데이터 표현 스타일을 기존에 setStyle 메서드를 이용해 지정한 스타일로 복원하는 메소드이다. 73~78줄에서 객체에 마우스가 올라가면 스타일을 재정의하였는데, 이를 복원하는 작업을 하게 된다.

광역지자체별 재난문자 현황



지도이(가) 표시된 사진

자동 생성된 설명

지도이(가) 표시된 사진

자동 생성된 설명

전체적인 흐름으로는, 우선 지도만 화면 상단에 뜬다. 그리고 광역 지자체를 경계로 하는 빨간색 구분선들이 지도 위에 나타나게 된다. 지자체 중 한 곳에 마우스를 올리면 해당 지역이 빨간색으로 칠해지며 지역 이름이 마우스 옆에 나온다. 지자체를 클릭하면 해당 지역이 초록색으로 칠해지며 지도 아래에 해당 지역에서 발령된 최근 10개의 재난 문자를 표로 출력한다. 다른 지역을 클릭하면 원래 칠해진 초록색이 다른 지역으로 옮겨지고, 해당 지역의 재난 문자가 표에 나온다.

헤드: 기본적인 정보 및 api 추출(3~22)

텍스트이(가) 표시된 사진

자동 생성된 설명

4~6줄까지는 emmet으로 인해 자동 생성된 부분으로, html파일의 정보를 나타낸다.

7줄은 페이지의 타이틀이다.

8~16줄은 문서에서 사용할 표에 대한 스타일을 지정한 부분이다.

18~19줄은 Naver api를 사용하기 위한 부분이고, 21~22줄은 jQuery를 이용하기 위한 부분이다.

재난 문자 받아 오기(24~69)

텍스트이(가) 표시된 사진

자동 생성된 설명

공공데이터포탈에서 재난 문자를 받아 오기 위한 스크립트가 있는 부분이다.

26~27줄은 현재 시간을 포매팅하는 부분이다. 공공데이터포탈에 데이터를 요청할 때 시간을 넣어줘야 하는데, 현재 시간을 넣기 위해 Date 클래스를 이용하여 특정 포맷으로 포매팅하였다.

29~36줄은 공공데이터포탈에 요청을 주기 위한 변수를 지정하는 부분이다. 공공데이터포탈 접근을 위한 servicekey, 페이지 수 pageNo, 요청할 메시지 수 numOfRows, 받게 될 데이터 타입 type, 요청 시간 create\_time, 요청할 지역 location\_name을 쿼리에 넣는다. 이때, location\_name은 후에 이 loadTable을 호출할 때 인자로 받게 된다.

28~42줄에서는 readystate가 변화할 때 함수를 실행하는 콜백함수를 지정하고 있다. readyState가 4, 즉 요청이 완료되어 응답이 준비되었고 status가 200, 즉 요청이 정상적으로 반환된 경우 myFunction()을 실행한다.

44~45줄에서는 open() 메소드를 통해 ajax 요청을 GET 형식으로 지정한 뒤 send() 메소드를 이용해 서버에 요청을 보내는 부분이다.

47~68줄은 서버에서 요청이 오면 수행하는 함수로, 지도 아래 표를 그리는 부분이다.

바디 시작 및 지도 생성(71~113)

텍스트이(가) 표시된 사진

자동 생성된 설명

72줄의 div는 지도가 표현될 div 객체로, 폭은 화면의 100%이고 높이는 400px이다.

73줄은 재난문자 내용이 출력 될 table 객체이다.

76~80줄은 다음에 나올 코드 들에서 서버에 요청을 보낼 때 사용할 여러 변수들을 미리 지정하는 부분이다.

82~86줄은 Map 클래스를 이용하여 객체 map을 선언하는 부분이다. 형식은 앞에서 분석한 데이터 레이어 코드의 형식과 동일하다.

89~113줄은 지도가 로드 된 다음 일회성으로 실행되는 이벤트로, 역시 데이터 레이어 코드에서도 나왔던 내용이다. 이벤트 리스너 함수에서는 i를 1부터 17까지 증가시키면서 총 17회 반복한다. 92~95줄은 keyword를 이용하여 숫자를 문자열 데이터로 변환한다. 한 자리 숫자라면 앞에 0을 붙여서 길이가 2인 문자열이 되도록 한다. 이후 98~111줄에서는 ajax 요청을 보낸다. url은 앞에서 지정한 변수에 keyword를 붙여 만들어진다. 요청이 완수되면 101~108줄의 함수를 실행한다. 이 함수에서는 요청의 결과로 얻은 GeoJson을 리스트 regionGeoJson에 넣고, 카운트를 증가시킨다. 카운트가 17이 되면, 즉 반복이 모두 이루어져 마지막 반복에서는 starDataLayer() 함수를 실행시킨다.

지도에 데이터 레이어 출력(115~142)

텍스트이(가) 표시된 사진

자동 생성된 설명

115줄의 tooltip은 화면에 마우스를 올렸을 때 생기는 지역명이 담긴 div에 대한 정보를 담는 jQuery이다.

117줄은 tooltip에 내용을 추가하는 부분이다. map 객체의 getPanes() 메소드는 지도의 창 요소의 집합을 객체로 반환하여 MapPanes 클래스 객체를 반환한다. MapPanes 클래스는 오버레이를 지도 위에 렌더링 할 수 있는 DOM요소를 포함하며, 지도에 있는 여러 레이어의 스택 순서를 나타낸다. 이 중 floatPane은 모든 오버레이보다 위에 위치하는 정보 창 요소를 포함하는 것이다. 즉, 가장 위에 나타나게 한다.

119줄부터는 107줄로부터 실행되는 startDataLayer() 함수이다. 모든 ajax 요청이 완료되어 GeoJson을 받으면 실행된다.

120~138줄은 지도의 스타일을 설정하는 setStyle() 메소드이다. 초기값을 설정하고, 프로퍼티의 값 중 focus가 true일 때 변하게 될 스타일 또한 설정해 준다.

140~142줄은 모든 GeoJson을 하나씩 map 객체의 data 객체에 넣어주고 있다. forEach를 이용하면 각 요소에 대해 함수를 수행할 수 있다.

이벤트 리스너 등록(144~180)

텍스트이(가) 표시된 사진

자동 생성된 설명

144~157줄은 객체를 클릭했을 때의 이벤트 리스너를 등록한다. 우선, data 객체의 프로퍼티인 feature 객체를 별도의 변수 feature에 담는다. 객체가 클릭되었을 때의 액션은 두 가지이다. 첫 번째로 원래 객체에 포커스가 없었을 경우이다. 이 경우 우선 모든 features 객체에 접근하여 포커스를 해제시킨다. 또한 revertStyle() 메소드를 이용하여 data 객체의 스타일을 120줄에서 setStyle() 메소드로 설정한 값으로 복원한다. 이후 loadTable() 함수를 실행한다. 인자로는 해당 객체의 지역 이름이 전달된다. 이 지역 이름은 location‌\_name으로 전달되어 공공데이터포탈에 요청을 보내는 데 사용될 것이다. 그리고 지금 선택한 지역의 포커스를 true로 변경한다. 그러면 129~135줄에서 설정한 스타일이 적용되어 초록색 배경이 될 것이다. 두 번째로 원래 객체에 포커스가 있었을 경우, 클릭하게 되면 포커스를 해제함으로써 초록색 바탕이 없어지게 될 것이다.

159~161줄에서는 객체에 마우스가 올라 왔을 때의 이벤트 리스너를 등록한다. 우선, feature 객체를 별도의 변수 feature에 담는다. 그리고 해당 객체의 지역 이름을 regionName에 담는다. 그리고 앞서 만들었던 div객체의 jQuery인 tooltip에 css() 메소드로 css를 적용하면서 지역 이름을 text() 메소드를 이용하여 넣어준다. css() 메소드의 인자를 보면 마우스의 현재 위치에 div 객체를 위치시키는 것을 볼 수 있다. 이후 data 객체의 overrideStyle() 메소드를 이용하여 스타일을 재정의한다. 이를 통해 바탕의 투명도가 조절되어 색이 칠해지고, 선의 두께가 굵어진다.

176~179줄에서는 객체에서 마우스가 빠져나왔을 때의 이벤트 리스너를 등록한다. tooltip에 hide() 메소드를 이용하여 지역 이름을 보이지 않도록 수정하고, data 객체의 revertStyle() 메소드를 이용하여 원래의 스타일로 복원한다.