Project #2

1 프로젝트 소개 및 요구사항

본 프로젝트에서는, 서울시의 공공데이터를 지도상에서 3차원으로 시각화하는 웹페이지를 구현하는 것을 목표로 한다.

개인별로 적어도 다음과 같은 두 개의 웹페이지를 구현하여 제출한다.

1.1 서울시의 "동별" 공공데이터 시각화.

- "`population`" 예제를 수정하여 구현한다.
- 서울시 공공데이터 중 적절한 동별 통계데이터를 검색하여 이를 지도상에 3차원으로 시각화한다.
- 데이터는 "서울시 열린데이터 광장" (`https://data.seoul.go.kr`)에서 "동별" 키워드로 검색하면 얻을 수 있다.
- 여러 데이터를 조합하여 시각화하는 등 다양한 방법으로 확장해 보자.

1.2 서울시의 "위치" 기반 공공데이터 시각화

- "`wifi`" 예제를 수정하여 구현한다.
- 서울시 공공데이터 중 적절한 위치데이터를 검색하여 이를 지도상에 3차원으로 시각화한다.
- 데이터는 "서울시 열린데이터 광장" (`https://data.seoul.go.kr`)에서 "위치" 키워드로 검색하면 얻을 수 있다.
- 단, 위치정보가 위도+경도 형식의 좌표계(WGS1984)로 되어있는 것을 사용한다. (기타 형식인 경우 본인이 좌표계를 변환하도록 한다.)
- (모든 데이터를 검토해 보지는 않았지만) 특별히 좌표계가 언급되어 있지 않은 데이터는 WGS1984형식을 되어 있는 것으로 보인다.
- 다양한 방법으로 확장해 보자.

2 제출 파일

각 시각화 웹페이지 별로 다음 파일들을 제출한다. 제출 시, 각 웹페이지별로 폴더를 만들어 각각 필요한 파일들을 넣은 후 모두 한번에 압축하여 하나의 zip파일로 제출한다.

- `app.js`: main file. 대부분의 이 파일을 수정하여 구현하게 된다.
- `index.html`
- `package.json`: 필요한 node.js package에 대한 정보를 담은 파일. 추가적인 JS 라이브러리를 사용할 경우 수정할 수 있지만, node.js에 대해서 잘 모르는 경우 수정하지 말고 제출하도록 한다.
- `webpack.config.js`: webpack 설정파일. webpack에 대해서 잘 모르는 경우 수정하지 말고 제출하도록 한다.
- 필요한 공공데이터 파일 및 geojson 파일 등
- readme.txt: 앱에서 시각화하는 데이터에 대한 간략한 설명, 확장했을 시 어떠한 방법으로 확장하였는지에 대한 간략한 설명 등

다음 파일들은 제출해서는 안된다.

- `package-lock.json`: "`npm install`"을 수행하면 생성되는 파일
- `node_modules` 밑에 있는 파일들

채점 과정

1. "`npm install`"을 수행하여 package 빌드
2. "`npm start`"를 수행하여 app 실행
3. `readme.txt`를 참조하여 평가

3 서울시 데이터 시각화 예제

3.1 공통 요구환경

아래 두 예제는 모두 Mapbox API를 사용하고, node.js를 필요로 한다. 따라서 다음과 같이 환경설정을 해주어야 한다.

3.1.1 Mapbox Access Token 설정

Mapbox API를 사용하기 위해서는 access token 정보를 다음과 같이 설정해야 한다.

1. Go to `https://www.mapbox.com` and create an account.
2. Sign-in and go to the account page (`https://account.mapbox.com`)

3. Copy the "default public token"
4. Set the environment variable
   (1) 설정 – 시스템 – 정보 – 고급 시스템 설정 – 환경 변수...
   (2) XXX에 대한 사용자 변수 – 새로 만들기...
   (3) 변수 이름: `MapboxAccessToken`
   (4) 변수 값: 〈복사한 access token〉

3.1.2 Node.js 설치

1. Go to `https://www.nodejs.org`
2. Download an install Node.js LTS (not the "current" version) with default options

3.2 "서울시 주민등록인구 (동별) 통계" 데이터 시각화 예제

- 서울시의 동별 주민등록 인구데이터를 행정구역 경계데이터(geojson)와 결합하여 지도상에 3차원으로 시각화한다.
- 두 개의 파일을 비동기적으로 읽기 위해 `Promise.all` method를 사용하였다.
- deck.gl의 `GeoJsonLayer`를 사용하여 시각화하였다.
- deck.gl의 GeoJsonLayer (Polygons) 예제를 기반으로 구현하였다.
- 각 동 영역의 높이는 해당 동의 전체인구를, 각 동의 색깔은 가구별 인구수를 나타낸다.
- 두 파일 (행정구역 경계 geojson파일과 통계데이터) 간 동이름이 일치하지 않아 이를 수정하는 과정을 거친다. 즉, 여러 동이 합쳐있는 경우 (예: "종로 1.2.3.4가동") period(.)와 midpoint(·)이 혼용되어 있어, 이를 모두 midpoint로 수정한다. 본인이 다른 데이터를 사용할 경우에도 이러한 점에 유의한다.
- 행정구역 geojson 데이터는 대한민국 전체의 행정구역 경계데이터를 포함하고 있다. 본 프로젝트에서는 서울시의 행정구역만 필요하므로, geojson 파일을 읽은 후 서울시 데이터만 필터링하여 사용한다. 미리 geojson 파일을 수정하는 방법이 더 표율적이나, 본 예제에서는 필터링을 하는 예시를 보여주기 위해 이러한 방식으로 구현하였다.
- 주민등록 인구 통계데이터의 확장자는 txt이나, 파일은 csv (comma-separated values) 형식으로 작성되었다. 본 예제에서는 papaparse 라이브러리를 사용하여 이를 parsing하였다.
- 마우스 커서를 각 동 위에 올리면 자세한 정보가 tooltip으로 보여진다.
- 사용기술: deck.gl, Mapbox API, React framework, geojson 파일포맷, csv 파일 포맷 등

3.2.1 빌드 및 실행

1. Create a folder ("`population`") and copy the required files under it.
   - `app.js`
   - `index.html`
   - `package.json`
   - `webpack.config.js`
2. Open a terminal (e.g., "Windows PowerShell" for Windows10)
3. Go to the "`population`" foler.
4. Install the required JS packages by executing
   ```
   >> npm install
   ```
5. After a while, all the required packages are installed under the "`node_modules`" folder.
6. Download "`HangJeongDong_ver20200701.geojson`" in the "population" folder. (Right click -> "Save as...")
   (a) Go to `https://github.com/vuski/admdongkor/tree/master/ver20200701`
   (b) Click "`HangJeongDong_ver20200701.geojson`"
   (c) Click "Download" button.
   (d) Right click -> "Save as..."
7. 서울시 동별 인구데이터 생성
   (a) Go to `https://data.seoul.go.kr/dataList/10043/S/2/datasetView.do`
   (b) Click the "내려받기(TXT)" button.
   (c) Change the file name to "`stat_population_Seoul.txt`"
8. In the terminal, run the example by executing
   ```
   >> npm start
   ```

3.3 "서울시 공공와이파이 위치정보" 시각화 예제

- 서울시 공공와이파이 위치정보를 3차원 히스토그램 형태로 지도상에 시각화 한다.
- deck.gl의 `HexagonLayer`를 사용하여 구현하였다.
- deck.gl의 HexagonLayer 예제에 기반하여 구현하였다.
- csv 형식 혹은 json 형식의 데이터를 입력으로 받도록 구현하였다.
- csv 형식의 경우, 파일의 인코딩이 UTF-8 형식으로 되어 있는지 확인하고, 그렇지 않은 경우 변환을 하여야 한다.

3.3.1 빌드 및 실행

1. Create a folder (“`wifi`”) and copy the required files under it.
   - `app.js`
   - `index.html`
   - `package.json`
   - `webpack.config.js`
2. Open a terminal.
3. go to the “`wifi`” foler.
4. Install the required JS packages by executing

   ```
   >> npm install
   ```
5. After a while, all the required packages are installed under the “`node_modules`” folder.
6. 공공와이파이 위치 데이터 생성
   1) Go to `https://data.seoul.go.kr/dataList/OA-1218/S/1/datasetView.do`
   2) Click the ”내려받기(CSV)” button.
   3) Change the file name to “`locs_wifi_Seoul.txt`”
7. Convert the encoding of the file.
   1) Open “`locs_wifi_Seoul.txt`” with Notepad app.
   2) File -> Save As...
   3) encoding: UTF-8
   4) file name: `locs_wifi_Seoul-UTF8.txt`
8. In the terminal, run the example by executing

   ```
   >> npm start
   ```
9. Instead of CSV format, you can use JSON format by modifying the “`renderToDOM` function in “`app.js`” file.

4 디버깅

위와 같은 방법으로 앱을 실행하여 브라우저에서 소스를 볼 경우, 매우 길고 알아보기 힘든 소스가 화면에 출력되어 디버깅할 수가 없다. Google Chrome의 경우 다음과 같이 디버깅할 수 있다.

1. More Tools – Developer Tools – “Sources” tab

2. In the tree pane, double click "`app.js`" under the folder top − App − unnamed folder − .
3. This is the file you modified. You can put break points in it, etc.