EAMFS

재난 문자 필터링 서비스

2021년 12월 10일

2021년 2학기 소프트웨어시스템실습

담당교수: 최성종

이름 태은우

이메일 kstew16@uos.ac.kr

서울시립대학교 전자전기컴퓨터공학부2017440140

차 례

1. 요약 Abstraction

2. 문제와 해결책 Problem & Solution

3. 설계 Design (Structure, Flow, Technology Stack)

4. 구현 Implementation

5. 토론 Discuss

6. 결론 Conclusion

7. 부록 Appendix

# 요약

공공 데이터 포털의 지역별/종합 재난 문자 데이터를 이용하여 지도와 결합한 형식으로 재난문자를 지역별로 분류하여 제공하고, 나아가 원하는 내용을 조금 더 편하게 얻을 수 있도록 설계한 서비스입니다.

# 문제점 및 해결책 제시

문자메시지의 원하는 내용 검색 기능은 있어도 원하지 않는 내용 필터링은 없는 점에서 착안하여 코로나 제외/기간 지정과 동시에 넓은 뷰로 가독성 높게 재난문자를 활용할 수 있는 것이 주요한 특징입니다.

보통 재난 문자는 시끄럽게 울려대지만 정작 필요한 내용은 없고 같은 내용의 문자만 반복되어 오히려 재난 문자만 알림이 꺼져 있는 분들도 많습니다. 따라서 원하는 문자만을 골라 받을 수 있도록 문자 내용 필터링 기능을 추가함과 동시에, 자동으로 재난문자를 업데이트하여 실시간으로 메시지를 수신할 수 있도록 서비스를 구현하였습니다.

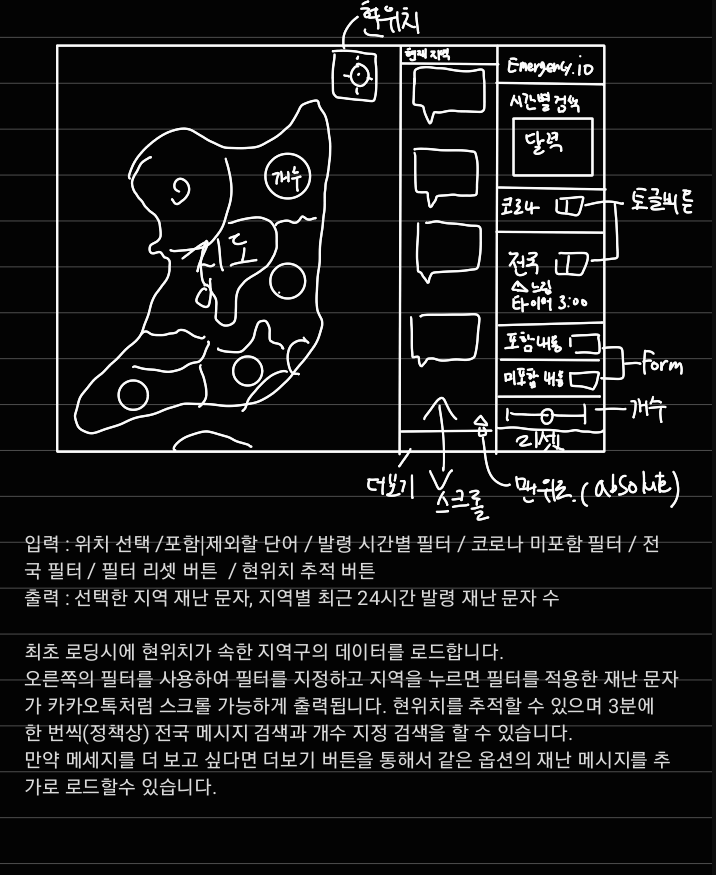
# 설계

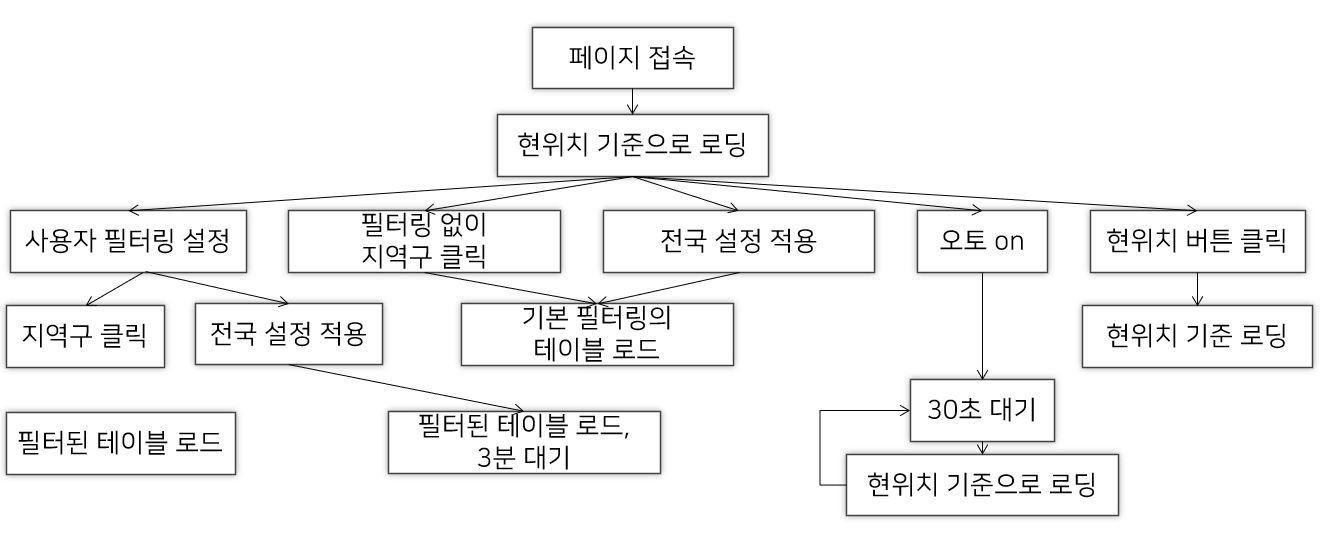
**프로그램은 크게 지도 API파트, 메시지 테이블 파트, 메시지 필터링 파트 등 세 파트로 구분할 수 있습니다.**

먼저 지도 파트는 데이터 레이어를 통해 지역을 선택하는 기능과, 현위치를 추적하여 표시하는 마커와 현위치 추적 버튼, 그리고 현위치에 대해 설정된 필터를 적용하여 자동으로 재난 문자를 갱신하는 갱신 버튼으로 구성되어 있습니다.

테이블 파트는 무한정 스크롤이 가능한 테이블과 테이블의 최상단으로 올라가는 버튼으로 구성되어 있으며, 필터링 파트는 날짜 필터링, 코로나 필터링 토글버튼, 전국 토글버튼, 포함 문자열, 미포함 문자열, 검색 개수와 필터 적용/재설정 버튼으로 구성되어 있습니다.

사용자는 앱을 통해 각 지역을 눌러 지역별 재난문자를 로드할 수 있으며, 전국 토글버튼이 체크된 상태에서 전국 단위의 재난문자를 로드할 수 있습니다. 또 현위치 버튼을 눌러 수동으로 사용자 위치의 재난문자를 로드하거나, 자동 갱신 버튼으로 30초에 한 번씩 재난 문자를 로드할 수 있습니다. 모든 로드는 사용자가 설정한 필터링이 적용된 상태로 이루어지게 됩니다.





최초 계획시에는 각 지역구별 메시지 개수와 메시지 더 보기 기능 등을 구현할 예정이었으나, 지역구별 메시지 개수는 실용성이 너무 없으며, 테이블 끝에서 메시지 더 보기 기능을 지원하는 대신 무한 스크롤이 더 적합할 것이라는 생각이 들어 계획이 수정되었습니다.

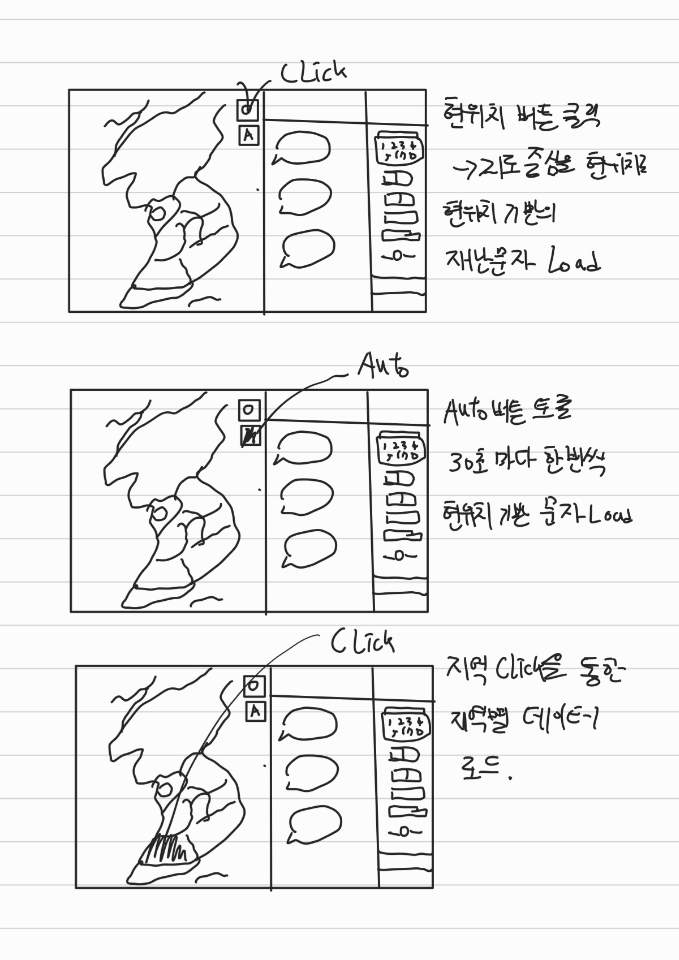


그림 1. 시나리오 1~3, 일반적인 재난문자 로드 방식

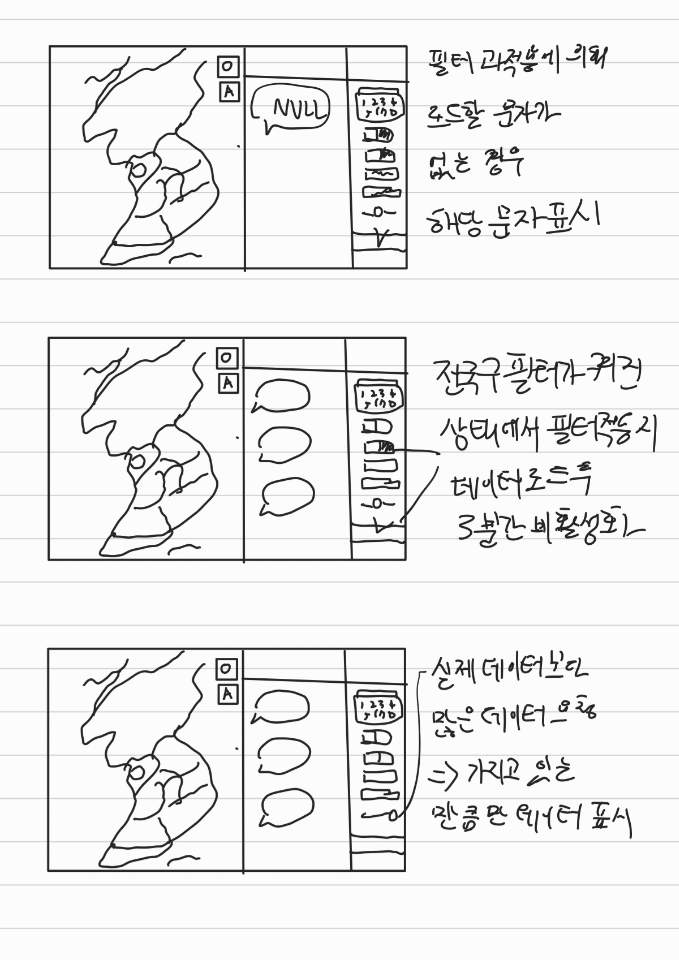


그림 2. 시나리오 4~ 6, 몇 가지 오류나 특별 상황

Ajax를 위해 jquery가, 지도 구현을 위해 navermap API가, 현위치 표현을 위해 Geocoder가 사용됩니다.

# 구현

재난 문자 로딩 부분은 지난 보고서와 상당 부분 유사하기 때문에 설명을 차용하였습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

먼저 map 객체를 선언해 줍니다.

map객체에는 naver.mpas 네임스페이스에 속한 Map() 클래스를 할당합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Map 클래스의 설명입니다.



Map 클래스의 생성자를 통해서 map 객체를 생성할 것입니다. mapDiv에는 Map 클래스의 객체가 할당될 div 태그의 id를 전달하고, mapOptions 매개변수에는 Map 클래스에 정의된 여러 가지 지도 옵션들을 지정할 수 있습니다.

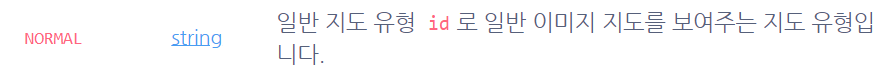


예제 코드에서는 초기 지도를 zoom 하는 수준을 7로 지정했습니다.

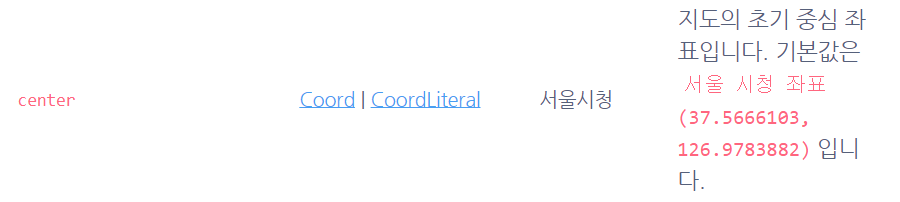
크롬 줌 100% 기준 한반도 전체가 대략적으로 보이는 수준입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명



지도 유형은 일반 이미지 지도이며



초기 중심 좌표값을 현위치 좌표로 설정한

Map 클래스 객체를 생성하여 map 객체에 할당함과 동시에 map이라는 id를 가진 div 에도 할당해 주었습니다. 현위치 획득에 대해선 후의 GeoCoder 파트에서 설명하겠습니다.

이제 생성된 map 객체에 데이터를 로드할 차례입니다. 데이터는 json타입의 데이터를 ajax방식으로 로드하게 되는데, 지도가 최초로 초기화되면 익명 함수를 통하여 ajax를 수행해 줍니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

데이터들을 로드할 경로의 홈 경로가 되는 주소를 HOME\_PATH에 저장하고

Ajax로 데이터를 수신합니다. Ajax를 수행할 url과 받아올 데이터 타입(json)을 스트링으로 전달한 뒤, Ajax가 성공하면 startDataLayer메소드와 현위치를 중심으로 하는 setCenterToCurrent 메소드를 호출하는 것으로 Ajax를 마무리합니다.

여기서 Ajax를 단 한 번 수행해 줄 클래스가 naver.maps 네임스페이스의

Event 클래스입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명  
텍스트이(가) 표시된 사진

자동 생성된 설명

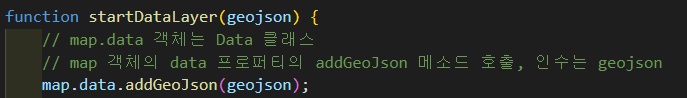
텍스트이(가) 표시된 사진

자동 생성된 설명

Event의 once 객체는 한 번만 이벤트 알림을 받아 핸들러를 호출해 줍니다.

예제 코드에서는 map의 stylemap(네이버 맵 API에서 제공하는 맵 타일) 이 초기화되면 ajax를 수행하는 익명 함수가 호출됩니다. Ajax의 내용에 대해서는 본 페이지 상단에 서술했습니다.

startDataLayer메소드에서는 GeoJson을 통해 지도에 점과 선으로 다각형 영역을 표시하거나, 지도의 스타일을 바꾸는 등의 내용이 포함되어 있습니다.



여기서 Geojson이란 위치정보를 갖는 점을 기반으로 지형을 표현하기 위해 사용하는 데이터 표준 형식입니다. startDataLayer 메소드는 호출될 때 Ajax를 통해 geojson 데이터를 받게 되고, 이것을 map 객체의 data 인스턴스에 추가함으로써 지도에 점, 선 도형등을 표시할 수 있게 됩니다.

여기서 data 인스턴스는 Map 클래스의 Data 인스턴스로 기본적으로 map을 생성할 때 생성된 인스턴스입니다.

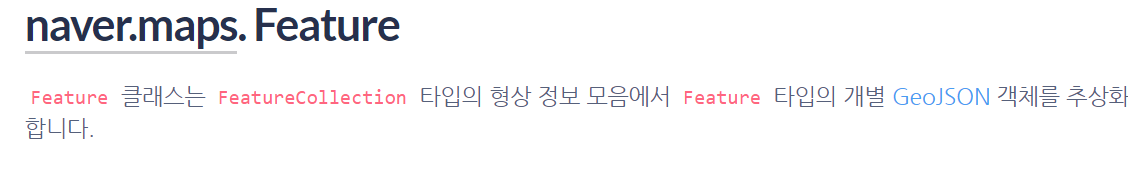
텍스트이(가) 표시된 사진

자동 생성된 설명

텍스트이(가) 표시된 사진

자동 생성된 설명

다음으로 map.data 인스턴스의 setStyle 메소드를 호출해 줍니다. 이때 데이터 레이어에는 Feature 클래스의 feature객체가 포함되어 있습니다



텍스트이(가) 표시된 사진

자동 생성된 설명

이 feature 객체는 GeoJSON 지리 공간 데이터의 형상 정보를 다루는 객체로,

본 코드에서는 익명 함수의 매개변수로 feature를 전달해 줍니다.

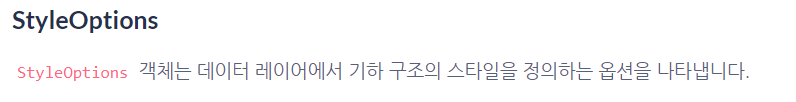
이 익명 함수는 feature의 getProperty() 메소드를 호출합니다.



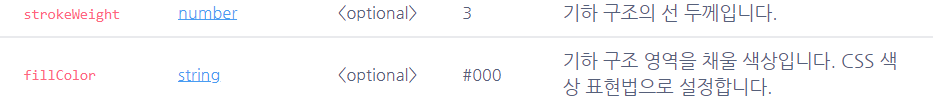
최종적으로 익명 함수는 syleOptions 형의 매개변수를 setStyle 메소드에 전달하게 됩니다.

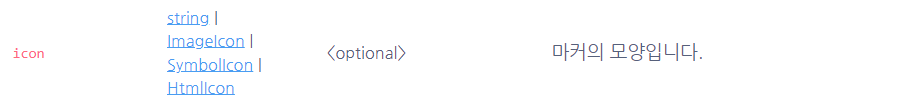
텍스트이(가) 표시된 사진

자동 생성된 설명









다음은 리스너 영역입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

data 레이어에 addListener 메소드로 리스너를 추가해 줍니다.



익명함수를 전달받은 리스너는 객체 위에 마우스가 올라올 때 지역명을 표시하거나, 마우스가 다시 내려갈 때 지역명 툴팁이 사라지게 합니다.

지도이(가) 표시된 사진

자동 생성된 설명

나머지 영역은 문자열을 통해서 원하는 문자열만을 동적 테이블로 구성하는 코드와

텍스트이(가) 표시된 사진

자동 생성된 설명

여타 지연시간 부여 함수, 테이블 최상단으로 이동하는 함수 등 DOM을 통한 간단한 함수들과 실습과정에서 사용한 여러 함수들로 구성되어 있습니다. 해당 내용은 구현에 큰 어려움이 없다고 생각하기 때문에 생략하겠습니다.

마지막으로 GeoCoder로 현위치를 받아오는 부분을 설명하겠습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

기본적으로 현위치는 BOM 중 navigator 객체의 getCurrentPosition 객체를 통하여 얻을 수 있습니다. 해당 메소드를 통해 얻은 경도와 위도를 naverMaps API 형식의 객체로 생성한 뒤 naver.maps.Service.reverseGeocode를 실행합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Naver.maps.Service 객체를 통해서 로드한 Geocoder API는

텍스트이(가) 표시된 사진

자동 생성된 설명

Reversegeocode API를 호출하여 navigator를 통해 수신한 좌표 정보를 주소 정보로 바꾸는데 사용되며, 이 반환인 Result 컨테이너를 통해서 현위치가 속해 있는 지역구의 문자열을 얻을 수 있습니다. 아래에 Result 컨테이너의 데이터 예시를 첨부합니다.

|  |  |
| --- | --- |
| 지도이(가) 표시된 사진  자동 생성된 설명 | 지도이(가) 표시된 사진  자동 생성된 설명 |

텍스트이(가) 표시된 사진

자동 생성된 설명

이제 마지막으로 11주차에 실습한 재난문자 로드 방식까지 사용하면 구현 과정은 끝이 납니다.

이제 필터링 등 세부 구현에 사용한 함수와 기능을 설명하겠습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

필터의 데이터 내용을 불러와 변수로 저장하는 getFormData()와 저장된 필터들을 default 값으로 리셋하는 resetFormData()입니다. 필터을 filter라는 class

로 선언하였기 때문에 인덱스별로 처리하였습니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

자동 메시지 업데이트를 담당하는 toggle 계열 함수입니다. toggleUpdate() 함수는 toggle버튼의 입력에 따라 타이머를 활성화하거나 비활성화하며 버튼의 아이콘을 바꾸고. toggleTimer 함수를 부릅니다. toggleTimer는 변수로 저장된 타이머의 30초 제한에 따라 1초에 한 번씩 setInterval을 이용해서 타이머를 진행합니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

테이블의 스크롤을 최상단으로 올려주는 버튼의 onclickListener인 tableToTop()함수입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Data.or.kr 의 정책에 따라 전국 재난메시지 로드를 3분 간격으로 제한하는 timer()와 시간이 다 되었을 때 전국 메시지 토글 버튼을 다시 활성화하는 activate() 함수입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

마지막으로 input type range의 값을 입력시바다 바꿔주는 showVal 함수입니다.

최종적으로 구현된 서비스의 스크린샷과 유저 시나리오를 첨부합니다.

지도이(가) 표시된 사진

자동 생성된 설명

그림 3. 지역별 재난 문자 로드 – 동대문 필터가 적용된 서울

지도이(가) 표시된 사진

자동 생성된 설명

그림 4. 코로나 필터를 적용한 전국 단위 재난문자 로드 (100개중 필터링)

지도이(가) 표시된 사진

자동 생성된 설명

그림 5. 현위치 자동 갱신 상태에서 안양을 포함하고 확진을 미포함한 문자

소프트웨어 구조는 설계와 같이 지도 파트와 메시지 테이블 파트, 필터링 파트로 구분되어 있으며, 사용자는 필터링을 설정하거나 설정하지 않은 뒤, 현위치나 지역구를 선택해 재난문자를 로드하거나, 자동 로드 버튼을 토글하여 사용자 위치의 재난 문자를 로드할 수 있습니다.

# 토론

기본적으로 메시지를 필터링하고 실시간으로 로드한다는 프로젝트의 기초적인 요구사항은 충분히 충족되었다고 볼 수 있습니다. 그러나 가시성의 측면에서 모바일 등의 다양한 기계에 따른 화면 크기 변화에 충분히 대응하지 못한다는 점에서 개선점이 보입니다.

또 동적 테이블이 자주 사용되는 프로그램인데, DOM 객체를 동적으로 접근하는 데에 Jquery가 유용하다고 들었으나 능숙하지 못해 효율적인 코드 작성이 이루어지지 않았던 점이 개선점이 될 수 있어 보입니다.

# 결론

과제 구현 결과 자동 메시지 로드, 지역별 메시지 로드와 메시지의 태그와 내용에 따른 필터링 등을 성공적으로 구현하였습니다. 다만 “코로나 필터링”기능에서 현재 사용한 방식은 코로나와 관련된 코로나, 확진자, 접종 등의 단어를 필터링하는 방식인데, 이 방식은 연관성이 상대적으로 작은 메시지도 필터링할 가능성이 존재하기 때문에 이를 문맥적이나 다른 방법을 통해 분석하는 알고리즘을 적용한다면 더욱 완성도 높은 프로그램이 될 수 있을 것입니다. 향후 이 프로그램이 모바일에서의 호환성을 갖출 수 있도록 Layout – HTML 구조 개선과 반응형 웹 서비스의 학습을 통하여 개선해 나갈 것입니다.

# 부록

<https://keichee.tistory.com/357>

Animated 토글 버튼 구현 참조

<https://navermaps.github.io/maps.js.ncp/docs/tutorial-Geocoder-Geocoding.html>

Geocoder

<https://navermaps.github.io/maps.js.ncp/docs/index.html>

Naver Maps API

전체 코드를 첨부합니다. Media와 font, 외부 css항목은 kstew16.github.io를 통해 다운로드 하실 수 있습니다.

<!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>Emergency.io</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=gslzy2klbe"></script>

    <!--Geocoder-->

    <!--현위치 좌표-> 주소로 변환용 -->

    <script type="text/javascript" src="https://openapi.map.naver.com/openapi/v3/maps.js?ncpClientId=gslzy2klbe&submodules=geocoder"></script>

    <style>

        .page{

            float: left;

            margin: 0px;

        }

        body{ -ms-overflow-style: none; }

        ::-webkit-scrollbar { display: none; } /\*특정 부분 스크롤바 없애기\*/

        .box{ -ms-overflow-style: none; }

        .box::-webkit-scrollbar{ display:none; }

        @font-face { /\* 나눔고딕 \*/

            font-family: "NanumGothic";

            src: url("font/NanumGothic.ttf") format("truetype");

            font-weight: normal;

        }

        form>hr{

            margin-bottom: 15px;

            margin-top: 15px;

        }

    </style>

    <link rel="stylesheet" href="toggle.css" type="text/css">

</head>

<body style="margin: 0px;">

<div id="map" style="width:55%; height:100vh; margin:0px; display: inline-block;" class="page">

    <button id="currentLocation" onclick="setCenterToCurrent()" style="position: absolute; z-index: 500; left: 93%; top: 2%; padding:0%"><img src="media/position.jpg" alt=""></button>

    <button id="updateButton" style="position: absolute; z-index: 500; left: 93%; top: 10%; padding:0%" onclick="toggleUpdate()"><img src="media/manual.png" alt=""><br><span id="secData" style="font-size: 0.5em;">30 초</span></button>

</div>

<div id="table" style="width: 30%; height: 100vh; overflow: auto; display: inline-block; background-color:#74BFF2;" class="page">

    <button id="tableToTop" onclick="tableToTop()" style="position: absolute; z-index: 500; left: 81%; top: 94%; padding:5px"><img src="media/arrowup.jpg"></button>

</div>

<div id="sideNav" style="width: 15%; height: 100vh; display: inline-block;" class="page;">

    <div style=

    "text-align: center;

    background-color: #E8726F;

    margin: 0%;

    font: bold 1.3em NanumGothic;

    padding-top: 25px;

    padding-bottom:25px;

    color: white;

    ">Emergency.io</div>

    <form style="height: 90%;">

        <legend style="margin:10px;">검색 시작 일자</legend>

        <input type="date" style="margin:10px; width: 90%;" class="filter" value="2021-06-22">

        <hr>

        <div style="width: 50%; display: inline-block; margin-left: 5%;">코로나 제외</div>

        <label class="switch">

            <input type="checkbox" class="filter">

            <span class="slider round"></span>

        </label>

        <hr>

        <div style="width: 50%; display: inline-block; margin-left: 5%;">전국 동시 검색</div>

        <label class="switch">

            <input type="checkbox" id="allSwitch" class="filter">

            <span class="slider round"></span>

        </label>

        <div style="margin-top: 5%; margin-left: 5%;"><img src="media/warning.png" style="margin-left: 3%; width: 1em;">  엄청나게 느리므로 주의, <br>버튼 클릭시 3분간 비활성화</div>

        <br>

        <span style="margin-left: 25%">남은 시간 : </span><span id="timeLeft">180</span>

        <hr>

        <div style="margin-top: 5%; margin-left: 5%;"><img src="media/warning.png" style="margin-left: 3%; width: 1em;">  특수 문자 / , [ , ] 금지  </div>

        <label style="margin-left: 5%;">포함 내용</label>

        <input type="text" style="margin-left: 5%; margin-top: 5%;" placeholder="쉼표(,)로 구분하여 입력" class="filter">

        <hr>

        <div style="margin-top: 5%; margin-left: 5%;"><img src="media/warning.png" style="margin-left: 3%; width: 1em;">  특수 문자 / , [ , ] 금지  </div>

        <label style="margin-left: 5%;">제외 내용</label>

        <input type="text" style="margin-left: 5%; margin-top: 5%;" placeholder="쉼표(,)로 구분하여 입력" class="filter">

        <hr>

        <label style="margin-left: 2%;">검색 개수</label>

        <input type="range" id="range" name="slide" for="searchVal" oninput = "showVal(this.value)" min="1" max="200" class="filter" value="100">

        <span id= "searchVal">100</span>

        <hr>

        <button onclick="getFormData()"

        onmouseleave = "this.style.backgroundColor='#7DDB74'"

        onmouseover = "this.style.backgroundColor='#5DBB54'"

        style="width: 95%;height: 10%; margin-left: 2%; background-color: #7DDB74; border:hidden; font: bold 1.5em NanumGothic;color: white;"

        type="button"

        id="set">적용</button>

        <hr>

        <button onclick="resetFormData()"

        onmouseleave = "this.style.backgroundColor='#E8726F'"

        onmouseover = "this.style.backgroundColor='#C8524F'"

        style="width: 95%;height: 10%;  margin-left: 2%; background-color: #E8726F; border:hidden; font: bold 1.5em NanumGothic;color: white"

        type="button"

        id="reset">재설정</button>

    </form>

</div>

<script>

let formDate = "2021/06/22 00:00:00";

let formNoCorona = false;

let formAllSearch = false;

let formInclude = new Array();

let formExclude = new Array();

let formNumRows = 100;

let autoUpTimerID = null;

let autoTimerClock = 30;

let autoUpdate = false;

function toggleUpdate(){

    let updateButton = document.getElementById("updateButton");

    if(!autoUpdate){

        updateButton.childNodes[0].setAttribute("src","media/auto.png");

        if(autoUpTimerID!=null){

            clearInterval(autoUpTimerID);

            autoTimerClock = 30;

        }

        autoUpTimerID = setInterval(toggleTimer,1000);

        autoTimerClock = 30;

        autoUpdate = true;

    }

    else{

        updateButton.childNodes[0].setAttribute("src","media/manual.png");

        clearInterval(autoUpTimerID);

        autoTimerClock = 30;

        autoUpdate = false;

        document.getElementById("secData").innerHTML = autoTimerClock + " 초"

    }

}

function toggleTimer(){

    autoTimerClock -=1;

    if(autoTimerClock==0){

        setCenterToCurrent();

        autoTimerClock=30;

    }

    document.getElementById("secData").innerHTML = autoTimerClock + " 초"

}

function getFormData(){

    filters = document.getElementsByClassName("filter");

    formDate=(filters[0].value).replace(/-/g,"/")+" 00:00:00"

    formNoCorona = filters[1].checked

    formAllSearch = filters[2].checked

    formInclude = (filters[3].value).toString().split(",");

    if(formInclude[0] == ""){

        formInclude = [];

    }

    formExclude = (filters[4].value).toString().split(",");

    if(formExclude[0] == ""){

        formExclude = [];

    }

    formNumRows = filters[5].value

    if(formAllSearch){

        timer();

    }

}

function resetFormData(){

    // 화면에 있는 요소 다 리셋

    // 전역변수도 리셋

    filters = document.getElementsByClassName("filter");

    filters[0].value="2021/06/22 00:00:00";

    filters[1].checked=false;

    filters[2].checked=false;

    filters[3].value = ""

    filters[4].value = ""

    filters[5].value=100;

    document.getElementById("searchVal").innerHTML=100;

    getFormData();

}

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(36.4303004, 127.817960)

});

function setCenterToCurrent(){

    // 현위치 좌표로 받아오는 부분

    var position = navigator.geolocation.getCurrentPosition(function(pos){

        var latitude = pos.coords.latitude;

        var longitude = pos.coords.longitude;

        var location = new naver.maps.LatLng(latitude,longitude);

        map.setCenter(location);

        // 현위치에 마커 추가

        var marker = new naver.maps.Marker({

            position: location,

            map: map

        });

        naver.maps.Service.reverseGeocode({

        coords: location,

        }, function(status, response) {

            if (status !== naver.maps.Service.Status.OK) {

                return alert('Something wrong!');

            }

            var result = response.v2, // 검색 결과의 컨테이너

            items = result.results;

            var locationAreaName = items[1].region.area1.name;

            // 현위치 기반으로 재난메시지 로드

            getLocationMessage(locationAreaName);

        });

    });

}

// 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();

                        setCenterToCurrent(); // 레이어 다 그렸으면 중앙을 현위치로, 현위치 메시지 로드

                    }

                }

            }(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;

    });

    // 받아온 Geojson파일을 map.data

    regionGeoJson.forEach(function(geojson) {

        map.data.addGeoJson(geojson);

    });

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

        var feature = e.feature;

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

            if(formAllSearch){

            }

            else{

                getLocationMessage(feature.getProperty('area1'));

                getLocationMessage(feature.getProperty('area1'));

            }

        }

    });

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

        // 이름을 받아

        var feature = e.feature,

            regionName = feature.getProperty('area1');

        // 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 sleep(ms) {

    const wakeUpTime = Date.now() + ms;

    while (Date.now() < wakeUpTime) {}

}

function getAllMessage(){

    let xhr = new XMLHttpRequest();

    let url = 'http://apis.data.go.kr/1741000/DisasterMsg3/getDisasterMsg1List'; /\*URL\*/

    let queryParams = '?' + encodeURIComponent('serviceKey') + '='+'oaXeVfWtlmMq/MVsfEXxkMocbaeAD88XfN9GtwY6NWMbMVLiQFrB0/2IbNYCBqyqZJiCUzCe8cIK2Asx3QYHHg=='; /\*Service Key\*/

    queryParams += '&' + encodeURIComponent('pageNo') + '=' + encodeURIComponent('1'); /\*\*/

    queryParams += '&' + encodeURIComponent('numOfRows') + '=' + encodeURIComponent(formNumRows); /\*\*/

    queryParams += '&' + encodeURIComponent('type') + '=' + encodeURIComponent('xml'); /\*\*/

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

    xhr.onreadystatechange = function () {

        if (this.readyState == 4) {

            let xml = this.responseXML;

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

            while(tableDiv.childElementCount>1){

                tableDiv.childNodes[0].remove();

            }

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

            //테이블 스타일 설정

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

            locTable.style.borderCollapse = "collapse";

            locTable.style.backgroundColor = "#74BFF2";

            //locTable.style.border = "1px solid gray"

            locTable.style.borderTop="0px"

            locTable.style.width = "100%"

            locTable.setAttribute("id","tableElement");

            // 헤더 생성

            let header = locTable.createTHead();

            let headRow = header.insertRow();

            //let headLoc = headRow.insertCell(0);

            let headMsg = headRow.insertCell(0);

            headMsg.style="position: sticky; top: 0%;"

            headRow.style.backgroundColor = "#7DDB74";

            headMsg.style.zIndex="3"

            //헤더 스타일 설정

            headMsg.style.backgroundColor = "#7DDB74";

            headMsg.style.textAlign="center"

            headRow.style.fontWeight = "bold"

            headRow.style.fontFamily = "NanumGothic"

            headRow.style.fontSize = "1.3em"

            headRow.style.color = "white"

            headMsg.innerHTML = "<br>요청지역 : "+"전 국"+ "<br><br>";

            //맨위로가기 버튼

            let upButton = document.createElement("BUTTON");

            upButton.style="position: absolute; z-index: 500; left: 81%; top: 94%; padding:5px";

            let upButtonIcon = document.createElement("IMG");

            upButtonIcon.setAttribute("src","media/arrowup.jpg");

            upButton.appendChild(upButtonIcon)

            upButton.onclick=tableToTop;

            locTable.appendChild(upButton);

            //수신한 데이터 사용 준비

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

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

            let dates = xml.getElementsByTagName("create\_date");

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

                let wrapDiv = document.createElement("DIV");

                let imgDiv = document.createElement("DIV");

                let textDiv = document.createElement("DIV");

                let img = document.createElement("IMG");

                img.setAttribute("src","media/speech\_bubble2.png");

                img.style.width="100%"

                let msgContent = msgs[i].childNodes[0].nodeValue;

                let splitPosition = msgContent.indexOf("]");

                textDiv.innerHTML = msgContent.substring(0,splitPosition+1)+"<br><br>"+msgContent.substring(splitPosition+1,);

                wrapDiv.style.position = "relative"

                imgDiv.style = "vertical-align : middle; position:relative; margin-top:30px; z-index:1;"

                textDiv.style = "width:70%; position:absolute; top:15%; left:5%; margin-left:50px; margin-right:50px; z-index:2"

                textDiv.style.fontFamily = "NanumGothic"

                textDiv.style.fontSize="0.9em"

                // 텍스트 거르기 여기서 msgs[i].childNodes[0].nodeValue; 보고 continue 때리면 됨

                if(formNoCorona){

                    formExclude.push("코로나")

                    formExclude.push("선별")

                    formExclude.push("확진")

                    formExclude.push("검사")

                    formExclude.push("접종")

                    formExclude.push("0시")

                    formExclude.push("거리두기")

                    formExclude.push("방역패스")

                }

                let isValid = true;

                if(formInclude.length>0){

                    for(j=0;j<formInclude.length;j++){

                        if((msgContent.toString()).search(formInclude[j])<0){

                            isValid = false;

                        }

                    }

                }

                if(formExclude.length>0){

                    for(j=0;j<formExclude.length;j++){

                        if((msgContent.toString()).search(formExclude[j])>=0){

                            isValid = false;

                        }

                    }

                }

                if(isValid){

                    imgDiv.appendChild(img);

                    wrapDiv.appendChild(textDiv);

                    wrapDiv.appendChild(imgDiv);

                    let row = locTable.insertRow();

                    let cellMsg = row.insertCell(0);

                    cellMsg.appendChild(wrapDiv);

                }

            }

            // 새로 로드할 때 테이블이 겹치지 않게

            // 여기에 맨 위로 버튼 추가해야 함. 그냥 row로 추가해

            while(tableDiv.childElementCount>0){

                tableDiv.childNodes[0].remove()

            }

            if(locTable.rows.length==1){

                let wrapDiv = document.createElement("DIV");

                let imgDiv = document.createElement("DIV");

                let textDiv = document.createElement("DIV");

                let img = document.createElement("IMG");

                img.setAttribute("src","media/speech\_bubble2.png");

                img.style.width="100%"

                wrapDiv.style.position = "relative"

                imgDiv.style = "vertical-align : middle; position:relative; margin-top:30px; z-index:1;"

                textDiv.style = "width:70%; position:absolute; top:15%; left:5%; margin-left:50px; margin-right:50px; z-index:2"

                textDiv.style.fontFamily = "NanumGothic"

                textDiv.style.fontSize="0.9em"

                imgDiv.appendChild(img);

                textDiv.innerHTML = "검색된 내용이 없습니다."

                wrapDiv.appendChild(textDiv);

                wrapDiv.appendChild(imgDiv);

                let row = locTable.insertRow();

                let cellMsg = row.insertCell(0);

                cellMsg.appendChild(wrapDiv);

            }

            tableDiv.appendChild(locTable);

            //tableDiv.appendChild(locTable);

        }

    };

    xhr.send('');

}

function getLocationMessage(location\_name){

    var xhr = new XMLHttpRequest();

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

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

    queryParams += '&' + encodeURIComponent('pageNo') + '=' + encodeURIComponent('1'); /\*\*/

    queryParams += '&' + encodeURIComponent('numOfRows') + '=' + encodeURIComponent(formNumRows); /\*\*/

    queryParams += '&' + encodeURIComponent('type') + '=' + encodeURIComponent('xml'); /\*\*/

    queryParams += '&' + encodeURIComponent('create\_date') + '=' + encodeURIComponent(formDate); /\*\*/

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

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

    //document.write(url+queryParams);

    xhr.onreadystatechange = function () {

        if (this.readyState == 4) {

            let xml = this.responseXML;

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

            while(tableDiv.childElementCount>1){

                tableDiv.childNodes[0].remove();

            }

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

            //테이블 스타일 설정

            locTable.style.borderCollapse = "collapse";

            locTable.style.backgroundColor = "#74BFF2";

            locTable.style.borderTop="0px"

            locTable.style.width = "100%"

            locTable.setAttribute("id","tableElement");

            // 헤더 생성

            let header = locTable.createTHead();

            let headRow = header.insertRow();

            //let headLoc = headRow.insertCell(0);

            let headMsg = headRow.insertCell(0);

            headMsg.style="position: sticky; top: 0%;"

            headRow.style.backgroundColor = "#7DDB74";

            headMsg.style.zIndex="3"

            //헤더 스타일 설정

            headMsg.style.backgroundColor = "#7DDB74";

            headMsg.style.textAlign="center"

            headRow.style.fontWeight = "bold"

            headRow.style.fontFamily = "NanumGothic"

            headRow.style.fontSize = "1.3em"

            headRow.style.color = "white"

            //headLoc.innerHTML = "지역";

            headMsg.innerHTML = "<br>요청지역 : "+location\_name + "<br><br>";

            //맨위로가기 버튼

            let upButton = document.createElement("BUTTON");

            upButton.style="position: absolute; z-index: 500; left: 81%; top: 94%; padding:5px";

            let upButtonIcon = document.createElement("IMG");

            upButtonIcon.setAttribute("src","media/arrowup.jpg");

            upButton.appendChild(upButtonIcon)

            upButton.onclick=tableToTop;

            locTable.appendChild(upButton);

            //수신한 데이터 사용 준비

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

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

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

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

                let wrapDiv = document.createElement("DIV");

                let imgDiv = document.createElement("DIV");

                let textDiv = document.createElement("DIV");

                let img = document.createElement("IMG");

                img.setAttribute("src","media/speech\_bubble2.png");

                img.style.width="100%"

                let msgContent = msgs[i].childNodes[0].nodeValue;

                let splitPosition = msgContent.indexOf("]");

                textDiv.innerHTML = msgContent.substring(0,splitPosition+1)+"<br><br>"+msgContent.substring(splitPosition+1,);

                wrapDiv.style.position = "relative"

                imgDiv.style = "vertical-align : middle; position:relative; margin-top:30px; z-index:1;"

                textDiv.style = "width:70%; position:absolute; top:15%; left:5%; margin-left:50px; margin-right:50px; z-index:2"

                textDiv.style.fontFamily = "NanumGothic"

                textDiv.style.fontSize="0.9em"

                // 텍스트 거르기 여기서 msgs[i].childNodes[0].nodeValue; 보고 continue 때리면 됨

                if(formNoCorona){

                    formExclude.push("코로나")

                    formExclude.push("선별")

                    formExclude.push("확진")

                    formExclude.push("검사")

                    formExclude.push("접종")

                    formExclude.push("0시")

                    formExclude.push("거리두기")

                    formExclude.push("방역패스")

                }

                let isValid = true;

                if(formInclude.length>0){

                    for(j=0;j<formInclude.length;j++){

                        if((msgContent.toString()).search(formInclude[j])<0){

                            isValid = false;

                        }

                    }

                }

                if(formExclude.length>0){

                    for(j=0;j<formExclude.length;j++){

                        if((msgContent.toString()).search(formExclude[j])>=0){

                            isValid = false;

                        }

                    }

                }

                if(isValid){

                    imgDiv.appendChild(img);

                    wrapDiv.appendChild(textDiv);

                    wrapDiv.appendChild(imgDiv);

                    let row = locTable.insertRow();

                    let cellMsg = row.insertCell(0);

                    cellMsg.appendChild(wrapDiv);

                }

            }

            // 새로 로드할 때 테이블이 겹치지 않게

            // 여기에 맨 위로 버튼 추가해야 함. 그냥 row로 추가해

            for(i=0;i<tableDiv.childElementCount;i++){

                tableDiv.removeChild(tableDiv.firstChild);

            }

            if(locTable.rows.length==1){

                let wrapDiv = document.createElement("DIV");

                let imgDiv = document.createElement("DIV");

                let textDiv = document.createElement("DIV");

                let img = document.createElement("IMG");

                img.setAttribute("src","media/speech\_bubble2.png");

                img.style.width="100%"

                wrapDiv.style.position = "relative"

                imgDiv.style = "vertical-align : middle; position:relative; margin-top:30px; z-index:1;"

                textDiv.style = "width:70%; position:absolute; top:15%; left:5%; margin-left:50px; margin-right:50px; z-index:2"

                textDiv.style.fontFamily = "NanumGothic"

                textDiv.style.fontSize="0.9em"

                imgDiv.appendChild(img);

                textDiv.innerHTML = "검색된 내용이 없습니다."

                wrapDiv.appendChild(textDiv);

                wrapDiv.appendChild(imgDiv);

                let row = locTable.insertRow();

                let cellMsg = row.insertCell(0);

                cellMsg.appendChild(wrapDiv);

            }

            tableDiv.appendChild(locTable);

        }

    };

    xhr.send('');

}

function tableToTop(){

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

    table.scrollIntoView(true);

}

function showVal(val){

    valSpan = document.getElementById("searchVal");

    valSpan.innerHTML = val;

}

let interID = null;

function activate(){

    clearInterval(interID);

    let textArea = document.getElementById("timeLeft");

    textArea.innerHTML="180"

    let allSwitch = document.getElementById("allSwitch");

    allSwitch.disabled=false;

}

function timer(){

    let allSwitch = document.getElementById("allSwitch");

    allSwitch.disabled=true;

    if(interID!=null){

        clearInterval(interID);

    }

    interID = setInterval(timer,1000);

    let textArea = document.getElementById("timeLeft");

    passed = parseInt(textArea.innerHTML)-1;

    if(passed>=0){

        textArea.innerHTML = passed;

    }

    else{

        activate();

    }

    if(passed>=179){

        //전국로드자랑!

        getAllMessage();

        alert("약 30초 후에 메시지가 로드됩니다.");

    }

}

</script>

</body>

</html>