

청주시 정책지도 기능 분석 및 HTML5 기반 재구현 제안

1. 기존 청주시 정책지도 기능 분석

청주시 지도모아 공간정보시스템에는 “공유 정책지도” 기능이 포함되어 있습니다 ¹. 이 기능은 분야별(주제별)로 분류된 여러 정책 관련 주제도를 지도 상에 중첩하여 보여주는 역할을 합니다. 예를 들어 **보건/복지** 분야에는 “여성 안심 택배함 위치”, “산후조리원 위치”, “약국 위치” 등의 주제도 레이어가 제공되고 ² ³, **안전** 분야에는 “안전상비의약품 판매위치” 등이 포함되는 식입니다. 사용자는 원하는 항목 옆의 체크박스를 클릭하여 해당 주제도를 지도에 표시하거나 숨길 수 있습니다 ⁴. 주제도는 전체, 문화/관광, 보건/복지, 안전, 경제, 환경/교통, 행정 등으로 상위 분류되어 있으며, 화면 상단에 카테고리 탭 버튼으로 제공됩니다 ⁵. 카테고리를 전환하면 해당 분야의 주제도 목록이 갱신되어 표시됩니다.

UI/UX 구성: 정책지도 인터페이스는 좌측 패널에 주제도 목록이, 우측에 지도가 나타나는 형태입니다. 목록에서는 각 주제도의 이름과 체크박스가 나열되고, 주제도 이름이 길 경우 말줄임처리 되며 툴팁이나 상세정보 보기가 가능합니다. 예를 들어 “청주놀이터지도(놀이)”와 같은 주제도도 목록에 표시됩니다 ⁶. 사용자가 체크박스를 클릭하면 **해당 레이어의 ON/OFF 토글** 이벤트가 발생하고, JavaScript를 통해 해당 주제도의 지도 레이어를 추가 또는 제거합니다 ⁷. 또한 검색창을 통해 주제도 이름으로 필터링하거나 초기화 버튼으로 검색어를 지울 수 있는 기능도 갖추고 있습니다 ⁸.

데이터 표시 방식: 사용자가 주제도 체크박스를 켜면, 프론트엔드에서 `displayPolicyMap(...)` 함수를 호출하여 해당 주제도의 데이터를 지도에 로드합니다 ⁹. 이 함수는 내부적으로 AJAX 호출을 통해 해당 주제도의 공간데이터를 서버로부터 가져온 뒤 지도에 그리는 절차를 거칩니다 ¹⁰. 실제 청주시 시스템에서는 `/geocd/selectPolicyAtrb.do` 와 같은 endpoint로 주제도의 속성정보와 도형정보를 요청하고, 응답받은 JSON 데이터를 파싱하여 **점 객체(POG003)**는 지도 마커로, **폴리곤/폴라인 등 도형(POG002)**은 벡터 레이어로 추가하는 식입니다 ¹¹ ¹². 필요에 따라 **Shapefile** 데이터를 읽어들이는 기능도 있으며 (시스템 코드에 `shp.js` 라이브러리를 포함 ¹³), 주제도에 따라 WMS 지도나 다른 주제도를 재귀적으로 포함(POG005), 특정 배경지도로 전환(POG006)하는 기능도 지원하도록 설계되어 있습니다 ¹⁴ ¹⁵. 다만 이러한 고급 기능 일부는 실제 서비스에서 미사용되었거나 주석 처리되어 있습니다. **체크박스를 끌 때는** 해당 레이어를 지도에서 제거하거나 투명도로 숨기는 처리를 하여 여러 주제도를 겹쳐 볼 수 있게 합니다 ¹⁶ ¹⁷. 또한 주제도 항목을 클릭하면 해당 주제도의 상세 설명이나 부가 정보를 팝업으로 제공하기도 합니다 (예: 주제도의 정의, 데이터 출처, 업데이트 일자 등) ¹⁸.

기술 스택: 청주시 지도모아는 **카카오 지도 API**와 **OpenLayers**를 함께 활용하고 있습니다. HTML 소스에 Kakao Maps JavaScript SDK가 포함되어 있으며 ¹⁹, OpenLayers 4.6.5 라이브러리도 로드하여 지도 기능을 구현합니다 ²⁰. 카카오 지도 API는 배경지도(일반지도, 스카이라이프 등)와 주소 검색/좌표계 변환 등을 제공하고, OpenLayers는 다양한 공간 데이터 소스 처리와 지도 상호작용을 담당하는 것으로 보입니다. 이러한 구성으로 사용자는 카카오 지도 스타일의 배경지도를 보면서도, OpenLayers를 통해 구현된 다양한 공간정보 레이어(항공사진 비교, 부동산정보, 주제도 등)를 동시에 활용할 수 있습니다. **반응형 디자인** 측면에서도 meta viewport 태그를 사용하여 모바일 환경에 대응하고 있고 ²¹, UI 요소 크기나 레이아웃도 PC와 모바일 모두에서 사용 가능하도록 개선된 것으로 알려졌습니다 ²² ²³. 실제로 좌측 패널을 숨기는 on/off 버튼을 제공하여 작은 화면에서는 지도를 크게 볼 수 있게 하는 등 모바일 UX를 고려한 설계가 적용되었습니다 ²⁴. 2025년 개선된 지도모아 시스템에서는 PC와 스마트폰 모두에서 **동일한 서비스 이용 경험**을 제공하도록 UI를 개편한 바 있습니다 ²².

요약하면, **청주시 정책지도** 기능은 다양한 부서의 정책적 관심 대상 위치 정보를 한 지도에 모아 제공하는 서비스입니다. 다수의 주제도 레이어를 카테고리별로 묶어 관리하고, 사용자가 필요에 따라 선택적으로 켜서 볼 수 있도록 함으로써, 행정지도 포털 내에서 **테마별 공간정보 대시보드** 역할을 하고 있습니다. 이러한 편리한 기능을 본 과제에서는 웹서

버 없이 로컬 HTML/JS만으로 동작하는 형태로 재구현하고자 합니다. 아래에서는 이를 위한 구체적인 기술 선택과 구현 방안을 제안합니다.

2. HTML5 기반 재구현을 위한 고려사항 및 구현 방안

청주시 정책지도와 유사한 기능을 독립된 HTML5 웹지도 어플리케이션으로 만들기 위해 몇 가지 핵심 요소를 고려해야 합니다:

- (1) 지도 라이브러리 및 기술 스택 선택
- (2) 배경 지도(베이스맵) 공급 방식 - 온라인 vs 오프라인
- (3) 정책지도 데이터 관리 방식 - 정적 파일 vs 외부연동
- (4) Excel (.xls) 파일 연동 방식
- (5) UI/UX 설계 - 반응형 디자인 및 터치 최적화
- (6) 단계별 구현 범위와 향후 고도화 방향

각 항목별로 상세히 살펴보겠습니다.

2-1. 웹 지도 API/라이브러리 선택 (Leaflet vs OpenLayers 등 비교)

정책지도 웹 어플리케이션의 핵심은 지도를 표시하고 다수의 레이어(마커, 도형)를 관리하는 것입니다. 이를 위해 오픈 소스 자바스크립트 지도 라이브러리를 선택하는 것이 바람직합니다. 후보로는 Leaflet, OpenLayers, MapLibre GL(JS) 등이 있으며, 또한 Kakao 지도 API나 VWorld 지도 API 같은 국내 지도 API도 고려될 수 있습니다. 아래 표는 주요 대안들의 특성과 장단점을 비교한 것입니다:

구분	Leaflet	OpenLayers	Kakao Maps API (참고)
장점	- 경량이며 배우기 쉬움. 초기 구현 용이 ²⁵ - 모바일 대응 및 기본 인터랙션 지원 우수 - 풍부한 플러그인 생태계 (클러스터링, 열지도 등) ²⁶ ²⁷	- 풍부한 GIS 기능과 유연성 (다양한 투영법, WMS/WFS 등 지원) ²⁸ ²⁹ - 복잡한 지도 어플 구현에 유리 (제약 적고 강력) ³⁰ ³¹	- 국내 지도의 높은 품질(도로명, POI 정보 풍부) - 주소 검색/길찾기 등 부가 API 제공 - 기존 청주시 시스템과 시각적 일관성
단점	- 고급 GIS 기능 부족 (GeoJSON 외 포맷은 플러그인 의존) ²⁸ - 좌표계 변환 등은 수동 구현 필요 (EPSG:5186 등은 직접 처리 필요) - 방대한 데이터(수천 개 객체) 처리 시 성능 열세	- 초기 러닝커브 높음 (간단한 지도 띄우기도 설정 많음) ³² - 문서/예제가 방대하여 입문자에 어려움 ³³ ³⁴ - 파일 크기 비교적 큼 (경량화 부족)	- 오프라인 불가 (항상 인터넷 필요) - 도메인 기반 API키 필요 (파일:// 동작 불확실) - 커스터마이징 제한 (타일 이미지 임의 저장 불가 등)
오프라인 사용	✓ (라이브러리 자체를 로컬 포함 가능 ³⁵ ; OSM 등 오픈타일 저장하여 사용 가능) ※ 타일 캐시/저장 별도 구현 필요	✓ (라이브러리 자체 로컬 사용 가능 ³⁵ ; 다양한 오프라인 데이터 포맷 지원) ※ 대용량 데이터 파일 로드 시 성능 고려	✗ (타일 서버 접근 필수. 지도 타일을 미리 내려받아 저장 불가능 - 약관상 제한)
모바일/반응형	✓ (터치 이벤트 기본 지원, 핀치 줌 등 제공) ✓ (SVG/CSS 기반 마커로 고해상도 화면 최적화 용이)	✓ (멀티터치 지원, 벡터렌더링 지원) △ (UI 직접 구현 필요, 별도 모바일 최적화 API 적음)	✓ (모바일 웹 SDK 제공, 성능 최적화) △ (웹뷰 등 특정 환경 제약 가능성)

구분	Leaflet	OpenLayers	Kakao Maps API (참고)
학습 곡선 및 커 뮤니 티	매우 쉬움 - 지도 지식 없어도 Quick Start로 구현 ³⁶ . 사용자 커뮤니티 가장 활발 ²⁶ .	중간~높음 - GIS 전문 지식 요함 (투영법, 좌표체계 등). 커뮤니티 규모 Leaflet보다 작음 ³⁷ .	중간 - 한글 문서와 예제 제공. 다만 커뮤니티는 제한적 (주로 Q&A는 공식포럼).

위 비교를 토대로, **1단계 구현**에서는 **Leaflet**을 사용하는 방안을 제안합니다. Leaflet은 “복잡한 GIS 애플리케이션이 아닌 지도 시각화가 주 목적일 때” 적합하며 ²⁹ , 초기 개발이 빠르고 iPhone Safari 등 모바일 호환성이 검증되어 있습니다. 현재 필요한 기능(마커 및 간단한 도형 표시, GeoJSON 처리 등)은 Leaflet의 기본 기능과 몇 가지 플러그인만으로도 충분히 구현 가능합니다. 반면 OpenLayers는 향후 **고도화 단계**에서 고려할 수 있습니다. 예를 들어 **다양한 투영 좌표계**(한국 TM 좌표 등)나 **대형 파일**(SHP, KML) 직접 로드, WMS 서비스 연계 등 고급 기능이 필요해지면 OpenLayers의 도입을 검토합니다. 초기에는 Leaflet으로 신속히 개발하고, 추후 요구 사항 증가 시 OpenLayers로 마이그레이션하거나 보안 기능만 부분 도입하는 전략이 효율적입니다.

참고: MapLibre GL JS는 오프라인용 **벡터 타일** 지도를 구현할 때 유용하지만, 초기 설정이 복잡하고 3D 가속 등을 사용하므로 모바일 브라우저 호환성에 유의해야 합니다. 현재 요구 범위에서는 굳이 벡터 지도까지는 필요 없어 보입니다. 또한 기존 시스템과 시각적 일관성을 중시한다면, Kakao 지도 API를 고려할 수 있으나 이 경우 **항상 인터넷 연결이 필요**하고, 로컬 파일로 실행할 때 API Key 인증 문제가 발생할 수 있습니다. 따라서 **오프라인 동작**을 보장하려면 Leaflet/OpenLayers + 오픈소스 지도타일 조합이 현실적입니다.

2-2. 배경 지도 제공 방식 - 온라인 타일 vs 오프라인 타일

정책지도 시스템의 **배경지도(베이스맵)**는 사용자의 공간 인지에 필수적입니다. 일반적으로는 인터넷을 통해 공용 지도 타일(예: OpenStreetMap, VWorld, Kakao 등)을 불러오면 간편하지만, 본 시스템의 요건 중 “웹서버 없이 로컬 HTML 파일로 작동” 및 “출장/현장 등 네트워크 여건 불확실한 환경”을 고려하면 **오프라인 지도 타일 제공**도 대비해야 합니다. 고려 가능한 전략은 다음과 같습니다:

- **온라인 지도 타일 활용:** 초기 구현 단계에서는 인터넷 연결이 있을 경우 **공개 지도 타일 서비스**를 사용할 수 있습니다. 예를 들어 Leaflet의 기본 예제처럼 OpenStreetMap 타일 서버 (`{s}.tile.osm.org`)를 불러오거나, 국토지리정보원의 VWorld API를 활용하여 배경지도를 표시할 수 있습니다. 이 방식은 간편하지만, 네트워크가 끊기면 지도 영역이 로드되지 않는 한계가 있습니다. 따라서 **오프라인 모드**를 지원하려면 별도 대비가 필요합니다.
- **오프라인 타일 패키지 생성:** 대상 지역(청주시 및 인근)의 지도 타일 이미지를 미리 내려받아 **로컬 자원으로 포함**시키는 방법입니다. 예컨대 TileMill 등을 사용해 특정 축척과 범위의 지도를 렌더링하고 Mbutil로 슬라이싱하여 PNG 타일 세트를 만들 수 있습니다 ³⁸ . 생성된 타일들은 프로젝트 폴더 내 `tiles/z/x/y.png` 형태로 저장해두고, Leaflet/OL에서 해당 경로를 타일 레이어 소스로 지정하면 됩니다 ³⁹ . 이 방식은 초기 작업량이 있지만 일단 구축하면 **완전 오프라인 동작**이 가능합니다. 다만, **데이터 용량**을 고려해야 합니다 - 청주시 전역의 다중 줌레벨 타일은 수백 MB 이상 될 수 있어, 필요한 범위와 축척을 선별적으로 캐시해야 합니다.
- **타일 캐싱 (런타임):** 또 다른 보완책으로, **브라우저 로컬스토리지**나 IndexedDB에 지도 타일을 캐싱하는 기능을 활용할 수 있습니다. OpenLayers의 과거 예시 중 HTML5 Cache Storage를 이용해 **사용자가 본 타일을 자동으로 저장**하고 오프라인 시 재사용하는 기능이 있었습니다 ⁴⁰ . Leaflet에는 기본 제공되지 않지만, Service Worker나 별도의 플러그인을 통해 비슷한 캐시 전략을 구현할 수 있습니다. 예를 들어 사용자가 사전에 특정 영역을 둘러봐서 타일이 로드되면, 이후 네트워크 없이도 그 타일은 저장된 버전을 보여줄 수 있습니다. 이 접근은 사전 캐시 없이 처음부터 완전 오프라인인 경우에는 소용이 없으므로, 근본적인 오프라인 지원을 위해서는 타일 패키지를 제공하는 편이 안전합니다.

제한: 현 단계에서는 **혼합 접근**을 권장합니다. 우선 **기본 배경지도**로는 온라인 시 **OpenStreetMap 표준 타일**을 사용합니다 (사용이 자유롭고 API Key 불필요). 동시에, 청주 지역의 주요 축척(예: 12~17레벨) 타일을 선별 다운로드하여 **로컬 타일 세트**로 포함시켜 둡니다. 애플리케이션 로드 시 인터넷 연결 유무를 확인하여, **온라인 가능 시**에는 실시간 타일을 쓰고 **오프라인 시**에는 내장된 타일 세트를 불러오는 방식으로 동작하게 할 수 있습니다. Leaflet의 `L.tileLayer`를 사용할 경우, URL을 동적으로 변경하거나, 또는 [기존 OSM 타일 레이어 객체](#)와 [기본 로컬 타일 레이어 객체](#) 두 개를 만들어 `addLayer` / `removeLayer`로 전환하는 방법을 쓸 수 있습니다.

타일 세트의 경로 예시:

```
// 온라인 타일 레이어
var osmLayer = L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', { ... });

// 오프라인 타일 레이어 (로컬 파일, xyz 폴더 구조)
var localLayer = L.tileLayer('tiles/{z}/{x}/{y}.png', { ... });
```

위에서 `tiles/` 디렉토리에 오프라인용 타일 이미지들이 저장되어 있다고 가정합니다. 이때 **보안상 제한**으로, `file://` 환경에서 브라우저가 로컬 파일 경로 접근을 막을 수 있으므로, 폴더 구조와 경로 설정에 유의해야 합니다. 일반적으로 동일한 경로상의 파일은 로드 가능하므로, 사용자에게 HTML과 `tiles` 폴더를 동일 디렉토리에 두고 실행하도록 안내하면 문제없습니다³⁹. 만약 브라우저 정책으로 문제가 발생하면, `href="tiles/..."` 형식으로 ``를 동적으로 삽입하거나 Base64 인라인 등 우회도 고려할 수 있지만, 번거롭기 때문에 가급적 표준적인 폴더 구조를 지키는 것이 좋습니다.

추가로, 향후 **PWA(Progressive Web App)** 기술을 도입하면 Service Worker가 네트워크 요청을 가로채 **타일에 대한 캐시 우선 로직**을 구현할 수 있습니다. 이를 통해 온라인 타일을 조회하다가 자동으로 캐싱하고, 오프라인 시 캐시를 사용하는 스마트한 동작도 가능해집니다. 초기 버전에서는 우선 수동적인 오프라인 지원(내장 타일)으로 충분하며, 고도화 단계에서 PWA 캐싱을 검토하면 됩니다.

2-3. 정책지도 데이터 관리 및 로드 방식

청주시 정책지도에 포함된 각 **주제도 레이어 데이터**는 주로 “지점의 위치 목록”(예: 시설 위치 목록) 또는 영역 폴리곤 등으로 구성됩니다. 원 시스템에서는 이러한 데이터를 서버 DB에서 조회하여 JSON으로 보내주지만¹⁰, 우리의 로컬 구현에서는 **서버 없이도 데이터가 제공**되어야 합니다. 이를 위한 방안은 두 가지입니다:

- **정적 파일 내장:** 각 주제도별 데이터를 **미리 확보하여** 애플리케이션에 포함시키는 것입니다. 예를 들어 여성 안심택배함 위치 레이어의 모든 좌표와 속성정보를 담은 `safebox.geojson` 파일을 만들어 두고, 필요 시 이 파일을 불러와 지도에 그리는 방식입니다. 불러오는 방법은 `<script>` 태그로 해당 JSON을 전역 변수로 불러들이거나, 혹은 AJAX로 로컬 경로를 읽는 것입니다. 하지만 `file://` 환경에서는 AJAX(`XMLHttpRequest` 또는 `fetch`)로 로컬 파일을 읽는 것이 동일 출처 제약에 걸릴 수 있습니다. 해결책으로 **데이터를 통합**하는 방법이 있습니다. 즉, 애플리케이션의 main JS에 모든 주제도의 데이터를 변수로 직접 포함하거나, 빌드 타임에 하나의 JSON으로 병합하는 것입니다. 예를 들어 `policyData = { "여성안심택배함": [...], "약국": [...] }` 식으로 초기 로딩시에 갖고 있게 할 수 있습니다. 이렇게 하면 서버 요청 없이 즉시 데이터 사용이 가능합니다.
- **사용자 업로드/연동:** 내부적으로 데이터를 유지보수하기 위해 Excel이나 별도 파일에 저장해놓고, 앱 실행 시 **사용자가 해당 파일을 선택하여 불러오는** 방법도 고려할 수 있습니다. 그러나 정책지도에 포함된 데이터는 비교적 정적인 공공정보(시설 위치 등)로 보이므로, 최신성을 위해 실시간 업데이트가 필요한 경우가 아니라면 정적 내장이 편리합니다. 만약 데이터 갱신 주기가 있다면, 주기적으로 새로운 JSON 파일을 생성하여 앱에 반영하는 워크플로우를 갖추면 됩니다 (예: 분기별로 담당자가 Excel을 JSON으로 변환하여 앱 패키지를 업데이트).

권장 방식: 초기 구현에서는 **주요 주제도 데이터를 앱에 내장**하는 것을 권합니다. 이를 위해 관련 부서로부터 해당 위치 목록과 좌표 데이터를 확보해야 합니다. 다행히 청주시 정책지도에 쓰인 데이터라면 시 자체에 DB나 Excel로 존재할 가능성이 높습니다. 좌표계는 가급적 **WGS84 경위도(EPGS:4326)**로 변환하여 저장해 두면 Leaflet 등에서 추가 변환 없이 활용할 수 있습니다. (국내 TM좌표(EPGS:5174 등)인 경우 proj4를 이용한 변환이 필요할 수 있음.) 각 항목에는 이름, 주소, 전화번호 등 부가 속성이 있을 텐데, 이들도 함께 저장해 두면 지도에서 마커 클릭 시 팝업으로 정보를 보여줄 수 있습니다.

예를 들어, 약국 위치 레이어를 위한 `pharmacies.js` (또는 .json) 파일을 만들어 다음과 같이 구성할 수 있습니다:

```
var PHARMACY_DATA = [
  { "name": "○○약국", "addr": "청주시 상당구 ...", "lat": 36.642, "lng": 127.489 },
  { "name": "□□약국", "addr": "청주시 흥덕구 ...", "lat": 36.628, "lng": 127.457 },
  // ... 이하 생략 ...
];
```

그리고 본 애플리케이션에서 해당 변수를 순회하며 Leaflet의 `L.marker([lat, lng])` 로 마커를 생성하고, `bindPopup(name/addr)` 으로 팝업을 달아주는 식입니다. 이렇게 하여도 수백 개 규모의 마커 표시에는 문제가 없으며, 필요하면 **Marker Cluster** 플러그인 등을 적용해 가시성을 높일 수 있습니다 (예: 약국처럼 개수가 많은 경우 군집 표시).

혹은 GeoJSON 포맷으로 준비해 `L.geoJSON(geojsonData)` 로 한꺼번에 레이어를 그리는 방법도 있습니다. GeoJSON은 표준 포맷이라 유지보수에 용이하고, Leaflet/OL 모두 쉽게 처리할 수 있으므로 적절합니다.

만약 **데이터의 양이 매우 방대**하여 모든 데이터를 한 번에 불러오는 것이 부담된다면, **수요 기반 동적 로드**도 고려해볼 수 있습니다. 예를 들어 특정 주제도의 체크박스를 켜줄 때만 해당 JSON 파일을 `<script>` 로 로드하거나, `fetch()` 로 불러오게 구현하는 것입니다. 이때 `file://` 에서 fetch가 동작하지 않을 수 있으므로, `<script>` 태그 동적 삽입 방법이 안전합니다. (예: `document.head.appendChild(document.createElement('script')).src='pharmacies.js';`) 이렇게 하면 로컬 파일을 불러올 수 있습니다³⁹. 초기에는 데이터량이 많지 않다면(예컨대 레이어 하나당 수백~수천 개 POI 이하) 그냥 모두 로드해도 무방합니다.

마지막으로, 각 주제도 데이터에는 **갱신 일자**나 **출처** 등의 메타정보도 있을 수 있습니다. 이러한 정보는 UI상에 표시해 주는 것이 좋습니다. 원 시스템도 주제도 상세정보 팝업에서 이를 보여줬을 것으로 추정되므로, 우리도 예를 들어 주제도명을 클릭하면 간략한 설명(예: "여성 안심택배함: 여성범죄 예방을 위해 설치한 무인택배 보관함 위치, 2023.5 갱신")을 alert이나 modal로 띄우는 기능을 넣을 수 있습니다. 이는 필수 기능은 아니지만 데이터 이해를 돕기 때문에 고려하면 좋습니다.

2-4. Excel (.xslm) 파일 연동 기능 구현

요구사항의 중요한 부분 중 하나는 “**.xslm 파일 업로드 또는 연결을 통해 출장지/점검지 리스트 기반으로 해당 위치를 지도에서 확인**”하는 기능입니다. 이는 아마 내부 행정 부서에서 **점검 대상 목록**을 매번 Excel 매크로 파일로 관리하고 있기 때문으로 보입니다. 따라서, 사용자가 그 파일을 가지고 현장에서 지도를 보고자 하는 시나리오입니다.

이를 구현하기 위해서는 다음을 고려해야 합니다:

① **XLSM 파일의 처리:** XLSM은 매크로가 포함된 Excel 파일 형식이지만, 파일 구조는 기본적으로 XLSX와 유사한 **ZIP 압축된 XML 구조**입니다. 브라우저 상에서 이 바이너리 포맷을 해석하려면 전용 라이브러리가 필요합니다. 일반적으로 많이 쓰이는 **SheetJS (xlsx 라이브러리)**를 활용할 수 있습니다. SheetJS는 XLSX는 물론 XLSM까지 파싱을 지원하며, 브라우저 환경에서도 동작합니다⁴¹. 이 라이브러리를 HTML에 포함하면, 사용자가 파일 `<input>` 으

로 .xslm을 선택했을 때 JavaScript로 그 내용을 읽어 JSON 객체로 변환할 수 있습니다. 매크로(VBA)는 브라우저에서 실행되지 않지만, 우리는 오직 **데이터 시트 내용**만 필요하므로 상관 없습니다. (SheetJS는 매크로를 제거한 워크북 데이터만 반환합니다.)

② **주소 혹은 좌표 추출:** Excel 내 리스트에는 보통 현장명, 주소, 담당자 등이 들어있을 것입니다. 우리가 지도에 표시하려면 **각 항목의 위치**가 필요합니다. 이상적으로는 Excel에 **위경도 좌표** 컬럼이 있다면 바로 쓰면 되지만, 없고 주소만 있다면 **주소->좌표 변환(지오코딩)** 작업이 필요합니다. 오프라인 환경에서 지오코딩을 수행하는 것은 매우 어렵기 때문에, 두 가지 대안을 고려해야 합니다: - **사전 좌표 포함 전략:** 부서에서 Excel을 생성할 때 미리 매크로나 GIS툴을 통해 주소를 위도/경도로 변환해 놓도록 합니다. 예를 들어 Excel 매크로에서 Kakao 주소 검색 API를 호출해 좌표를 채운다든지, 수동으로 좌표를 입력해 둘 수도 있습니다. 이렇게 하면 현장에서는 추가 변환 없이 바로 지도에 마커를 찍을 수 있습니다. - **온라인 지오코딩 API 활용:** 만약 현장에서 인터넷 연결이 가능한 상황이라면, 브라우저에서 Kakao 혹은 VWorld의 REST API를 호출해 주소를 좌표로 변환할 수 있습니다. Kakao의 경우 JS SDK를 통해 `places.searchAddress` 기능을 사용할 수도 있으나, 로컬 파일에서 Kakao API를 쓰는 건 앞서 언급한 제약(API Key, 도메인)이 있습니다. 차라리 REST API (`dapi.kakao.com/v2/local/search/address.json`)를 `fetch`로 호출하는 편이지만, 이 역시 Kakao는 **Referer 제한**이 있어 `file://` 요청을 거부할 수 있습니다. 해결법으로 프록시 서버 없이 안전하게 하려면 **VWorld 주소 API**가 나올 수 있습니다 (공공기관용으로 키 발급 가능하고, 리퍼러 제한 비교적 유연). 하지만 베스트는 가능하면 **오프라인 상황에도 대비**하도록 Excel에 좌표를 넣어두는 것입니다.

③ **구현 방법:** 사용자 인터페이스 측면에서는 “파일 불러오기” 버튼을 제공하고, 이를 누르면 `<input type="file" accept=".xslm,.xlsx">`를 트리거하여 사용자가 파일을 선택하게 합니다. 파일 선택 후 스크립트에서 SheetJS의 `XLSX.read()`를 사용하여 내용을 파싱하고, 특정 시트 (예: 첫 번째 시트)나 특정 셀 범위를 읽어 목록을 얻습니다. 필요한 컬럼 (예: 장소명, 위도, 경도)을 추출하여 자바스크립트 배열로 만들고, 이를 기반으로 지도에 마커를 생성합니다. 이 때 **특정 스타일**로 표시하여 다른 정책지도 레이어와 구분하거나, 번호를 매겨 순서를 나타낼 수도 있습니다. (예: 1번 현장, 2번 현장 등.)

지도가 로드된 상태에서 사용자가 파일을 업로드하면 곧바로 마커 레이어가 추가되어 **업로드한 목록의 위치들을 모두 지도에 강조**하는 흐름입니다. 추가로, 목록이 매우 많을 경우를 대비해 **지도 영역 내에만 표시**하거나, **클러스터** 기능을 적용하는 것도 생각해볼 수 있습니다.

④ **매크로 연계:** “연동”이라는 표현에서 추측해볼 수 있는 것은, Excel 매크로와 브라우저 간의 연계를 뜻할 수도 있습니다. 예컨대 PC에서 Excel VBA로 Internet Explorer 컨트롤을 열어 지도를 띄우고, VBA에서 좌표를 넘겨주는 방식도 과거에는 활용되었습니다. 그러나 iPhone Safari 등 크로스 플랫폼을 요구하는 상황에서는 이런 방식은 통하지 않습니다. 대신 **Excel -> CSV 내보내기** 등의 방법으로 데이터를 교환하거나, 우리의 웹앱을 개선하여 **Excel 내용을 실시간으로 반영**하는 방향으로 가야 합니다. 현실적으로 브라우저가 Excel 파일 변화를 실시간 감지하긴 어려우므로, 사용자가 변경 시 다시 업로드하는 것으로 충분할 것입니다.



요약 제안: 초기 버전에서는 **파일 수동 업로드** 방식으로 구현합니다. 사용자가 .xslm (또는 .xlsx) 파일을 업로드하면 해당 내용의 **좌표 목록을 읽어 지도에 마커를 표시**합니다. 이를 위해 SheetJS 라이브러리를 포함하고 ⁴¹, 주소 컬럼에 좌표가 없다면 제한적으로 VWorld 주소 API 등을 호출하여 좌표를 얻는 식으로 구현합니다. 추후에는 Excel 매크로 수준에서 좌표를 채워놓도록 가이드하거나, 나아가 **모바일 앱(PWA)**에서 엑셀 파일을 열람/전달하는 기능까지 고민해볼 수 있습니다. 예컨대 PWA로 설치된 지도 앱이 특정 경로의 파일(JSON 등)과 동기화되어 최신 리스트를 자동 표시한다면 더욱 편리해질 것입니다 (고도화 방향에서 추가 설명).

2-5. UI/UX 설계 - 반응형 디자인 및 터치 최적화

사용 환경이 PC 웹브라우저(Chrome/Edge/Whale 등)부터 iPhone의 Safari 모바일 브라우저까지 다양하므로, **반응형 웹디자인**으로 구현해야 합니다. 구체적인 UX 설계 제안은 다음과 같습니다:

- **레이아웃:** 기본 레이아웃은 **좌측 사이드바 + 우측 지도** 형태를 유지하되, 모바일 화면에서는 사이드바를 자동으로 숨기거나 플로팅 패널로 겹쳐 보여주기 방식이 좋습니다. 예를 들어 화면 너비가 600px 미만이면 사이드 메

뉴를 아이콘 버튼(≡ 메뉴 버튼)으로 축소하고, 누르면 전체 화면을 덮는 모달 또는 드롭다운 형태로 주제도 목록을 표시합니다. 화면이 넓을 때는 항상 왼쪽에 패널이 떠있는 형태로 합니다. CSS 미디어쿼리(`@media`)를 사용하여 이러한 레이아웃 변화를 줄 수 있습니다.

- **반응형 요소:** 각 주제도 리스트 항목은 모바일에서는 한 줄에 표시되도록 하고 (원 시스템처럼 `
`로 줄바꿈하지 않도록), 글꼴 크기와 패딩을 조절해 터치하기 쉽게 만듭니다. 일반적으로 **터치 대상 영역은 40px 이상**이 권장되므로 체크박스나 레이블을 포함하는 `label` 높이를 크게 하고 여백을 둡니다. 또한 iOS 사파리에서 tap 했을 때 하이라이트가 남지 않도록 `-webkit-tap-highlight-color: transparent;` 등을 설정하면 미려합니다.
- **터치 제스처:** 지도 상에서는 **드래그 이동, 핀치 줌** 등이 기본 동작으로 지원돼야 합니다. Leaflet나 OpenLayers 모두 멀티터치를 지원하므로 기본 설정을 활용하면 되고, 필요시 `gestureHandling` 플러그인을 통해 두 손가락으로만 지도 이동이 가능하게 하는 등 UX를 개선할 수 있습니다 (지도 줌을 과도하게 뺄 때 실수 터치를 방지하는 등). 또한 **더블탭** 시 확대/축소 동작이 플랫폼마다 다를 수 있으니, 라이브러리 기본 설정을 따르거나 필요에 따라 비활성화합니다.
- **색상 및 테마:** 청주시 지도모아의 시그니처 색상이 있다면 (#7fbe26 등의 포인트 색상 ⁴²) 그대로 가져와 체크박스 선택 색 등 UI 테마에 적용합니다. 이는 사용자들이 친숙함을 느끼게 하고, 기존 시스템과의 연속성을 줍니다.
- **아이콘:** 주제도 종류별로 아이콘을 달리하여 지도 마커에 표시하면 이해를 돕습니다. 예를 들어 약국은  아이콘, 관광지는  아이콘 등. Leaflet에서는 커스텀 마커 이미지를 지정하거나, 폰트어셋 등의 아이콘 폰트를 사용할 수 있습니다. 단, **아이콘 파일을 로컬에 포함**해야 오프라인에서도 표시되므로, SVG로 경량 저장하는 것이 좋습니다.
- **상세 정보 표시:** 사용자가 지도상의 마커를 탭하면 해당 지점의 상세 정보를 팝업으로 표시합니다. 앞서 데이터에 포함된 속성 (명칭, 주소 등)을 예쁘게 보여주면 되고, 필요하면 “길찾기” 버튼을 넣어 눌렀을 때 카카오맵 앱이나 T맵 등을 호출하는 것도 고려 가능합니다 (예: `kakaoapi://route?sp=...` 스킴 활용). 내부 업무용이라면 굳이 길찾기까지는 아니더라도, 최소한 주소 복사하기 정도 기능은 편의를 줄 수 있습니다.
- **성능 최적화:** 모바일 기기는 PC보다 성능이 낮으므로, 초기 로딩 시 너무 많은 요소를 한꺼번에 띄우지 않도록 합니다. 처음부터 모든 레이어를 다 그리지 말고, 체크된 항목만 그리기가 기본이 되게 합니다. (원 시스템도 모든 체크박스가 기본 off이고 사용자가 선택 시 로드하는 방식이었습니다 ⁴³.) 또한 메모리 누수나 반복 연산을 줄여 배터리 소모를 최소화해야 합니다. 예를 들어 업로드한 Excel 마커들은 사용자가 지우기 전까지 남아있도록 하지만, 새 파일을 불러오면 이전 마커 레이어는 제거하여 중복 표시를 방지하는 등 관리가 필요합니다.
- **접근성:** 내부 직원용이지만, 그래도 색약자를 위한 대비 항상 모드, 키보드 내비게이션 등의 접근성도 염두에 두면 좋습니다. 예를 들어 체크박스에 `aria-label`를 달아 스크린리더가 읽을 수 있게 하거나, 중요 알림은 충분한 색상 대비를 주는 식입니다.

전반적으로 **PC와 모바일 간 UX 편차를 최소화**하면서도, 각각의 장점을 살리는 방향으로 설계합니다. iPhone Safari의 특이사항 (예: iOS에서는 파일 다운로드/저장이 Android보다 제약이 있음, standalone 모드 이슈 등)도 테스트하며 조정합니다. 기본적으로 Safari/Webkit 최신 버전 기준으로 개발하고, Edge/Chrome 등은 데스크톱 크롬 기준으로 문제 없도록 하면 큰 어려움은 없을 것입니다.

3. 우선 구현 기능과 단계별 고도화 방향

마지막으로, 프로젝트를 **단계별로 진행하는 방안**을 정리하면 다음과 같습니다:

3-1. 1단계 - 핵심 기능 우선 구현

목표: 기존 정책지도의 필수 핵심 기능을 HTML/JS로 재구현하여, **동일한 UI/UX의 기본 동작**을 확보한다.

- **✓ 지도 기본 구현:** Leaflet 기반 지도를 화면에 표시. OSM 배경지도 연동 및 청주시 지역 중심으로 초기 센터 설정. 줌 레벨 제한 등 필요시 설정.
- **✓ 주제도 목록 UI:** 사이드바 또는 메뉴에 카테고리 탭과 주제도 리스트 구현. 기존 시스템과 동일한 한글 레이블 사용 (예: "보건복지", "경제" 등).
- **✓ 다중 레이어 토글:** 체크박스 on/off에 따라 해당 레이어 추가/삭제. 초기에 예시로 1~2개 주제도 (샘플 데이터) 연결하여 마커가 보이는지 확인.
- **✓ 샘플 데이터 내장:** 예를 들어 "약국 위치", "산후조리원 위치" 등의 좌표 데이터 일부를 하드코딩 또는 JSON으로 포함해 기능 테스트.
- **✓ .XLSX/.XLSM 업로드:** 파일 input을 통해 파일 선택 → SheetJS로 데이터 파싱 → 좌표 목록 → 지도 마커 표시 구현. (주소 → 좌표는 우선 좌표가 포함된 샘플 파일로 가정하거나, 인터넷 연결 상태에서 API 호출로 임시 구현.)
- **✓ 반응형 디자인:** CSS 미디어쿼리로 사이드바 토글, 폰트 크기 조절 등 기본적인 대응. iPhone Safari에서 터치 및 확대 동작 테스트.
- **✓ 팝업 및 정보창:** 마커 클릭시 이름 정도 표시되는 팝업 구현. 주제도 목록의 항목 클릭시 해당 레이어의 모든 마커를 강조하거나 지도를 해당 범위로 줌인하는 등의 피드백 제공.
- **✓ 오프라인 실행 테스트:** 개발 완료 후, 실제로 네트워크 차단된 상태에서 PC와 iPhone에서 파일을 열어 핵심 기능 (내장 데이터 표시, 업로드 데이터 표시)이 돌아가는지 검증. (지도 타일은 이 단계에서는 온라인일 때만 보이겠지만, 에러가 나더라도 앱이 죽지 않도록 예외 처리 필요.)

위 항목들이 완료되면, 비록 모든 주제도의 실제 데이터를 다 넣지는 않았더라도, **동일한 UX 흐름**을 갖춘 프로토타입이 완성될 것입니다. 이를 관련 부서의 시연과 피드백을 받아 개선합니다.

3-2. 2단계 - 데이터 확충 및 사용자 피드백 반영

목표: 모든 주제도 데이터를 반영하고, 사용상의 개선점을 수정한다.

- **✓ 전체 주제도 데이터 통합:** 청주시 정책지도에 있는 모든 주제도의 데이터를 수집하여 내장 or 외부파일로 제공. 데이터 양이 많을 경우 메모리 영향 점검.
- **✓ 주제도 상세정보 표시:** 각 레이어에 대한 설명/출처/업데이트일 등을 별도의 정보 창(예: "i 정보" 아이콘 누르면 모달)에 표시하도록 구현.
- **✓ 주소-좌표 변환 기능 개선:** .xlsx 업로드 시 주소만 있는 경우의 처리 로직을 개선. 인터넷 연결 시 Kakao/VWorld API로 좌표 받아 마커 표시 + 오프라인 시 안내 메시지 (예: "좌표 정보가 없어 표시 불가" 등) 제공.
- **✓ UI 다듬기:** 디자인 요소 폴리싱 - 시각적 일관성 (청주시 CI 색 적용 등), 아이콘 그래픽 추가, 로딩 인디케이터 (데이터 로드 시 "로딩중..." 애니메이션) ⁴⁴ 등 추가. 불필요한 알럿 제거 등.
- **✓ 성능 최적화:** 주제도 많아질 경우 초기 로드 지연 방지 위해 **지연 로딩** 도입. 필요 시 Intersection Observer 등 활용하여 사이드바 스크롤시 가시 항목만 DOM 생성 등 최적화.
- **✓ 다국어/문구 가다듬기:** 내부 직원 대상이므로 한글 위주이지만, 필요시 용어 통일 및 매뉴얼 문구 추가 (예: "파일 선택시 엑셀의 '현장목록' 시트를 읽습니다" 같은 안내).

2단계까지 마치면 실제 업무에 투입할 수 있는 **완성본**에 가까워집니다.

3-3. 3단계 - 고도화 및 추가 기능

목표: 사용자 경험 향상과 시스템의 활용도를 높이는 부가 기능을 단계적으로 구현한다.

- **완전 오프라인 지도 지원:** 2단계까지는 기본 기능이 동작하지만, 여전히 배경지도는 온라인 의존일 수 있습니다. 3단계에서는 청주시 전체 영역의 오프라인 지도타일을 생성하여 앱에 포함시킵니다. 필요시 외장 메모리나 별도 다운로드를 통해 용량 문제를 해결하고, Leaflet에 `L.tileLayer.offline` 플러그인을 적용하거나 PWA 캐싱 전략을 구현해 인터넷 없이도 지도 배경이 나오는 환경**을 만듭니다.
- **PWA 기능**:** 웹앱을 Progressive Web App으로 제작하여, 아이폰/아이패드 홈화면에 앱 아이콘으로 추가하고, 전체화면 모드로 구동되도록 합니다. 이렇게 하면 마치 네이티브 앱처럼 사용할 수 있고, 오프라인 캐시도 용이해집니다. (`manifest.json` 생성 및 Service Worker 등록 등 작업 필요)
- **GPS 연동: 현장 활용도를 높이기 위해** 사용자 현재위치 표시** 기능을 추가합니다. 브라우저의 Geolocation API를 이용하여 GPS 좌표를 받아오고, 지도에 현위치 마커 및 정확도를 나타내는 원을 표시합니다. 이를 통해 담당자가 자신의 위치와 대상지를 한눈에 파악하고, 가까운 순서로 점검을 진행하는 데 도움을 줄 수 있습니다.
- **길찾기/외부연계: 특정 마커(출장지)를 탭했을 때** 길찾기 버튼**을 눌러, 카카오맵 또는 네이버지도 등 설치된 앱으로 연동하는 기능을 넣을 수 있습니다. (예: `window.location.href = "kakaomap://route?sp="+현재위치+"&ep="+목적지좌표;`). iOS와 안드로이드에서 동작을 검증해야 합니다.
- **사용자 메모/사진 첨부: 점검 업무 중 현장에서** 메모나 사진을 남기는 기능**을 지도에 추가할 수 있습니다. 예를 들어 특정 점검지 마커를 꼭 누르면 “메모 추가” 옵션이 나오고, 텍스트 입력이나 사진 파일 첨부를 받아 임시 표시하거나, Excel로 다시 내보내는 등의 확장입니다. 이는 작업 기록을 현장에서 디지털하게 수집하는데 도움이 되며, 추후 데이터 피드백 루프를 형성합니다.
- **권한 및 보안: 내부용 시스템이지만, 만약을 위해** 간단한 인증 절차**(비밀번호 입력이나 장치 등록)를 둘 수도 있습니다. 특히 PWA로 배포할 경우 URL이 알려지면 외부인이 접근할 수 있으므로, 앱 구동 시 사전 로그인(사내 인트라망 통해 토큰 발급 등) 고려 가능합니다. 초기엔 필요 없지만 고도화 단계에서 검토합니다.
- **실시간 데이터 연동**:** 정책지도 데이터 중 실시간 변동이 있는 항목(예: 공공자전거 위치 등)이 있다면, Wi-Fi 연결 시 실시간 API를 불러와 갱신하는 기능을 넣을 수 있습니다. 그러나 일반적인 행정시설 위치 등은 고정되어 있을 가능성이 높아 우선순위는 낮습니다.

以上과 같은 추가 기능들은 필요성과 우선순위를 고려해 선택적으로 구현하면 됩니다. 특히 **GPS 현위치**와 **PWA 지원**은 현장 활용성 측면에서 큰 가치가 있으므로 적극 권장되는 고도화 항목입니다.

4. 기능/기술 요소 체크리스트

마지막으로, 제안된 구현에 대한 **개발 체크리스트**를 제공합니다. 이 체크리스트를 따라가며 개발 및 테스트하면 주요 요구 사항을 빠뜨리지 않고 만족시킬 수 있을 것입니다:

- **[] 동일 UI 구현:** 청주시 정책지도 메뉴 구조와 항목명을 그대로 사용하여 친숙한 인터페이스 제공 (카테고리, 주제도명 확인).
- **[] 지도 표시:** PC 웹과 iPhone Safari에서 지도 타일과 축척바, 기본 컨트롤이 정상 표시되는지 확인.
- **[] 레이어 토글:** 각 주제도 체크박스 On/Off 시 해당 레이어의 마커/도형이 나타나고 사라지는지, 여러 개 동시 표시도 문제없는지 테스트.
- **[] 데이터 정확성:** 내장한 시설물 좌표가 실제 위치와 맞는지 검증 (임의 몇 개 마커의 주소와 지도를 대조).
- **[] xlsx 업로드:** 샘플 Excel을 통해 마커 생성이 잘 이루어지는지, 한글/특수문자 데이터도 깨지지 않고 표시되는지 확인. 좌표 없는 경우 경고 처리 등.
- **[] 반응형 동작:** 브라우저 창 크기를 줄였을 때 메뉴가 적절히 숨겨지고 나타나는지, 모바일 기기 세로모드/가로 모드 전환 시 UI가 어긋나지 않는지 확인.
- **[] 터치 인터랙션:** 지도 이동, 확대/축소, 마커 클릭 등이 터치 환경에서 원활하게 동작하는지 (두손가락 줌, 터치 스크롤 등).
- **[] 오프라인 테스트:** 개발 기기에서 인터넷을 끈 후 지도 및 데이터 표시가 어떻게 되는지 확인. (가능한 캐시/내장 타일로 일부 지도 보이는지, 아니면 메시지 안내 필요).

- [] **성능 점검:** 수백 개 이상의 마커를 표시해도 앱이 버벅이거나 크래시 되지 않는지 (특히 메모리 제한이 있을 수 있는 모바일에서).
- [] **다환경 테스트:** Windows, macOS의 Chrome/Edge, iOS Safari, Android Chrome 등으로 모두 접속하여 주요 기능이 정상 작동하는지 크로스체크.
- [] **문서화:** 내부 직원들이 사용할 간단한 **매뉴얼** 또는 가이드 작성 (예: “① 정책도 선택 ② 파일 불러오기” 등의 순서 안내), Excel 준비 방법 안내 포함.
- [] **유저 피드백 수렴:** 시연 후 수집된 의견(불편사항, 추가 희망기능 등)을 정리하여 향후 개선 로드맵에 반영.

以上的 제안과 체크리스트를 따라 개발을 진행하면, **청주시 정책지도** 기능을 iPhone Safari부터 PC 브라우저까지 일관된 UX로 제공하는 **경량 웹지도 시스템**을 구현할 수 있을 것입니다. 이 시스템은 인터넷 연결이 어려운 현장 업무에서도 유용하게 쓰일 수 있고, 향후 필요에 따라 유연하게 기능을 확장해 나갈 기반을 갖추게 됩니다.

참고 자료: 청주시 지도모아 기존 시스템 UI 및 코드 분석 ^{1 2}, 웹 지도 라이브러리 특성 비교 ^{25 29}, 오프라인 타일 활용 방안 ^{38 39}, 브라우저 Excel 파싱 기술 ⁴¹ 등. 이는 구현 과정에서 기술 선택과 설계의 근거로 활용되었습니다. 보다 나은 서비스를 위해 추후 사용자 피드백과 최신 기술 동향을 지속적으로 반영할 것을 권장합니다.

^{1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 42 43 44} 청주시꺼.txt

file:///file-XSCWcJHYbX9hkEy4xdAxpK

^{22 23} 최신 지도·정보 한눈에...청주시, '지도모아' 새단장 - 노컷뉴스

<https://www.nocutnews.co.kr/news/6343042>

²⁴ 청주시 지도모아 공간정보시스템

<http://map.cheongju.go.kr/main.do>

^{25 26 27 28 29 30 31 32 33 34 36 37} Leaflet vs OpenLayers. Pros and cons of both libraries | Geoapify

<https://www.geoapify.com/leaflet-vs-openlayers/>

³⁵ Is it possible to use the openlayers where we dont have internet ...

<https://stackoverflow.com/questions/49841038/is-it-possible-to-use-the-openlayers-where-we-dont-have-internet-connectivity>

^{38 39} How to create local map tiles for OpenLayers 2 offline use? - Geographic Information Systems Stack Exchange

<https://gis.stackexchange.com/questions/207769/how-to-create-local-map-tiles-for-openlayers-2-offline-use>

⁴⁰ OpenLayers Offline Storage Example

<https://wichita.ogs.ou.edu/OpenLayers-2.12/examples/offline-storage.html>

⁴¹ json - How to parse Excel (XLS) file in Javascript/HTML5 - Stack Overflow

<https://stackoverflow.com/questions/8238407/how-to-parse-excel-xls-file-in-javascript-html5>