웹 컴포넌트

웹 컴포넌트(Web Components)는 웹 페이지나 웹 애플리케이션을 구축할 때 사용되는 재사용 가능한 사용자 인터페이스 요소를 만드는 데 도움을 주는 표준 기술 세트입니다. 이 기술들은 웹 개발자들이 독립적이고 캡슐화된 요소를 생성하도록 하여, 코드의 재사용성과 유지보수를 향상시킵니다. 웹 컴포넌트의 주요 장점은 재사용성, 캡슐화, 간결한 코드 등입니다.웹 애플리케이션의 개발과 유지보수를 더욱 효율적으로 만들어줍니다.

웹 컴포넌트 구성

기술	설명
커스텀 엘리먼트(Custom Elements)	개발자들이 자신만의 HTML 요소를 정의하고 사용할 수 있게 해줍니다. 이를 통해 새로운 태그를 만들고, 그 태그에 특정한 동작이나 스타일을 적용할 수 있습니다.
섀도우 돔(Shadow DOM)	DOM의 일부를 별도의 독립적인 '섀도우' DOM 트리로 캡슐화합니다. 이는 스타일과 마크업을 메인 문서의 DOM으로부터 격리시키므로, 컴포넌트 간 스타일 충돌을 방지하고 코드를 더 깔끔하게 유지할 수 있게 해줍니다.
HTML 템플릿(HTML Templates)	`<template>` 태그를 사용하여 마크업을 쉽게 재사용할 수 있는 방식으로 저장할 수 있습니다. 이 태그 내부에 있는 내용은 페이지 로드 시 렌더링되지 않지만, JavaScript를 사용하여 필요할 때 인스턴스화하고 문서에 추가할 수 있습니다.
HTML 임포트(HTML Imports)	HTML 파일을 다른 HTML 파일에 임포트할 수 있는 방법을 제공합니다. 이 기능은 웹 컴포넌트를 여러 페이지에서 재사용할 때 유용합니다. 하지만 최신 웹 표준에서는 대신 JavaScript 모듈이 선호됩니다.

```
<!DOCTYPE html>
<html>
<head>
    <title>Web Components Example</title>
</head>
<body>

<!-- 커스텀 엘리먼트 사용 -->
<my-custom-element></my-custom-element>

<!-- 섀도우 돔 호스트 -->
<div id="shadow-host"></div>

<!-- 템플릿 사용 -->
<div id="template-container"></div>

<!-- 커스텀 엘리먼트 정의 -->
<script>
    class MyCustomElement extends HTMLElement {
```

```
        constructor() {
            super();
            this.innerHTML = "Hello, I'm a custom element!";
        }
    }
    customElements.define('my-custom-element', MyCustomElement);
</script>

<!-- 섀도우 돔 적용 -->
<script>
    const shadowHost = document.getElementById('shadow-host');
    const shadowRoot = shadowHost.attachShadow({ mode: 'open' });
    shadowRoot.innerHTML = `<style>p { color: red; }</style><p>Hello from
Shadow DOM!</p>`;
</script>

<!-- 템플릿 내용 -->
<template id="my-template">
    <p>Hello, this is content from the template!</p>
</template>

<!-- 템플릿 사용 -->
<script>
    const templateContainer = document.getElementById('template-
container');
    const templateContent = document.getElementById('my-
template').content;
    templateContainer.appendChild(templateContent.cloneNode(true));
</script>

</body>
</html>
```

- 서브모듈로 가져온 basic 리포지토리 코드 참고하여 동적 브라우저 구현