Chap40 이벤트

▼ 이벤트 드리븐 프로그래밍

이벤트가 발생하면 호출될 함수를 브라우저에게 알려 호출을 위임한다. 왜냐하면 개발자는 사용자가 언제 이벤트를 발생(EX 버튼 누르기)시킬 지 모르기 때문에 브라우저에게 함수 호출을 위임한다. **여기서 함수 호출을 이벤트 핸들러라하고 위임하느 것을 이벤트 핸들러 등록이라고 한다.**

이처럼 이벤트와 그에 대응하는 함수를 통해 사용자와 애플리케이션은 상호작용한다. **프로그램의 흐름을 이벤트 중심으로 제어하는 프로그래밍 방식**을 **이벤트 드리븐 프로그래밍**이라고 한다.

▼ 이벤트 타입

- ▼ 마우스 이벤트
- ▼ 키보드 이벤트
- ▼ 포커스 이벤트
- ▼ 폼 이벤트
- ▼ 값 변경 이벤트
- ▼ DOM 뮤테이션 이벤트
- ▼ 뷰 이벤트
- ▼ 리소스 이벤트

▼ 이벤트 핸들러 등록

- ▼ 이벤트 핸들러 어트리뷰트 방식

  HTML 요소의 어트리뷰트 중에는 이벤트에 대응하는 이벤트 핸들러 어트리뷰트가 있다.

  이벤트 핸들러 어트리뷰트 이름은 onClick과 같이 on접두사와 이벤트의 종류를 나타내는 이벤트 타입으로 이루어져 있다.

주의할 점은 이벤트 핸들러 어트리뷰트 값으로 **함수 참조가 아닌 함수 호출문 등의 문을 할당한다는 점**이다.

이벤트 핸들러 어트리뷰트 값은 사실 암묵적으로 생성될 이벤트 핸들러의 함수 몸체를 의미한다.

→ 이벤트 핸들러 어트리뷰트 값으로 여러 개의 문을 할당할 수 있다.

이벤트 핸들러 어트리뷰트 방식은 오래된 코드에서 간혹 이 방식을 사용한 것이 있기 때문에 알아둘 필요는 있지만 더는 사용하지 않는 것이 좋다.

왜냐하면 HTML과 자바스크립트는 관심사가 다르므로 혼재하는 것보다 분리하는 것이 좋다.

▼ 이벤트 핸들러 프로퍼티 방식

window객체와 Documnet, HTMLElement 타입의 DOM 노드 객체는 이벤트에 대응하는 이벤트 핸들러 프로퍼티를 가지고 있다.

이벤트 핸들러 프로퍼티에 함수를 바인딩하면 이벤트 핸들러가 등록된다.

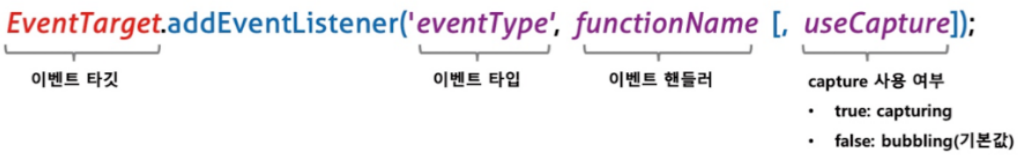
이벤트 핸들러를 등록하기 위해서는 이벤트를 발생시킬 객체를 이벤트 타깃과 이벤트의 종류를 나타내는 문자열인 이벤트 타입 그리고 이벤트 핸들러를 지정할 필요가 있다.

이벤트 핸들러는 대부분 이벤트를 발생시킬 이벤트 타깃에 바인딩한다. 하지만 반드시 이벤트 이벤트 타깃에 바인딩해야 하는 것은 아니다. 이벤트 핸들러는 이벤트 타깃 또는 전파된 이벤트를 캐치할 DOM노드 객체에 바인딩한다.

이벤트 핸들러 프로퍼티 방식은 HTML과 자바스크립트가 뒤섞이는 문제를 해결할 수 있지만 이벤트 핸들러 프로퍼티 하나의 이벤트 핸들러만 바인딩할 수 있다는 점이

단점이다.

▼ addEventListener 메서드 방식



`EventTarget.prototype.addEventListener` 메서드를 사용하여 이벤트 핸들러를 등록할 수 있다.

앞서 살펴본 이벤트 핸들러 어튜리뷰트 방식과 이벤트 핸들러 프로퍼티 방식은 DOM Level 0부터 제공되던 방식이다.

이벤트 핸들러 프로퍼티 방식은 하나 이상의 이벤트 핸들러를 등록할 수 없지만 `addEventListener` 메서드는 하나 이상의 이벤트 핸들러를 등록할 수 있다.

→ 이때 이벤트 핸들러는 등록된 `순서대로` 호출된다.

▼ 이벤트 전파

이벤트 전파 : DOM 트리 상에 존재한는 DOM 요소 노드에서 발생한 이벤트는 DOM트리를 통해 전파되는 것

이벤트 전파 단계

1. 캡처링 단계 : 이벤트 상위 요소에서 하위요소 방향으로 전파
2. 타깃 단계 : 이벤트가 이벤트 타깃에 도달
3. 버블링 단계 : 이벤트가 하위 요소에서 상위 요소 방향으로 전파

대부분의 이벤트는 캡처링과 버블링을 통해 전파된다. 하짐나 다음 이벤트는 버블링을 통해 전파되지 않는다. 이 이벤트들은 버블링을 통해 이벤트를 전파하는지 여부를 나타내는 이벤트 객체의 공통 프로퍼티 event.bubbles의 값이 모두 false이다.

- 포커스 이벤트 : focus/blur
- 리소스 이벤트 : load/unload/abort/error
- 마우스 이벤트 : mouseenter / mouseleave

위 이벤트들은 버블링되지 않으므로 이벤트 타깃의 상위 요소에서 위 이벤트를 캐치하려면 캡처링 단계의 에빈트를 캐치해야 한다.

▼ 이벤트 위임

여러개의 하위 DOM요소에 각각 이벤트 핸들러를 등록하는 대신 하나의 상위DOM요소에 이벤트 핸들러를 등록하는 방법