KH정보교육원

JAVASCRIPT 이벤트

JavaScript - 이벤트

KH정보교육원

이벤트 활용

- 이벤트 속성과 이벤트핸들러(함수)를 연동하여 이벤트 발생시 특정기능을 하도록 하는 것

이벤트 설정방법

- 고전이벤트 모델
- 인라인 이벤트 모델
- 표준 이벤트 모델
- 마이크로소프트 인터넷 익스플로러 이벤트 모델

JavaScript - 이벤트

KH정보교육원

고전 이벤트 모델

- 요소객체가 가지고 있는 이벤트 속성에 이벤트핸들러를 연결하는 방법
- 이벤트를 제거할 때는 속성값에 null값을 넣어주면 됨
- 이벤트발생객체는 핸들러 내부에서 this로 표현 / 스타일변경 가능
- 매개변수로 이벤트 정보전달(e, window.event) ☞ 이벤트객체 전달

예) 클릭시 이벤트설정

```
var h=document.getElementById('id명');
h.onclick=function{
        수행기능 설정;
        h(this).onclick=null; //한번만 실행
};
```

JavaScript - 이벤트

KH정보교육원

인라인 이벤트 모델

- 요소내부에 이벤트를 작성하는 방법
- 인라인방식은 <scrip>태그에 있는 함수를 호출하는 방식으로 선호

예) 클릭시 이벤트설정

```
<h1 onclick='처리로직'></h1>
```

또는

```
<h1 onclick='스크립트내 함수호출'></h1>
```

JavaScript - 이벤트

KH정보교육원

표준 이벤트 모델

- w3에서 공식적으로 지정한 이벤트모델
- 한번에 여러가지 이벤트핸들러 설정 가능
- this키워드가 이벤트발생객체 의미

메소드	내용
addEventListener(이벤트이름, 핸들러, 확장)	확장 : 버블링/캡쳐링
removeEventListener(이벤트이름,핸들러)	이벤트 삭제

예) 클릭시 이벤트설정

```
var h=document.getElementById('id명');
h.addEventListener('click',function(){
        수행기능 설정;  };
```

JavaScript - 이벤트

KH정보교육원

익스플로러 이벤트 모델

- 익스플로러 브라우저 적용 모델
- 한번에 여러가지 이벤트핸들러 설정 가능
- this키워드가 이벤트발생객체(srcElement)가 아니고 window객체 의미

메소드	내용
addEventListener(이벤트이름, 핸들러, 확장)	확장 : 버블링/캡쳐링
removeEventListener(이벤트이름,핸들러)	이벤트 삭제

예) 클릭시 이벤트설정

```
var h=document.getElementById('id명');
h.addEventListener('click',function(){
        수행기능 설정;  };
```

JavaScript - 이벤트

KH정보교육원

이벤트 전달

- 버블링 방식 : 자식에서 부모노드로 올라가면서 이벤트가 실행
- 캡쳐링 방식 : 부모노드에서 자식노드로 내려가면 이벤트가 실행

☞ 익스플로러는 지원하지 않음

이벤트 차단

- 이벤트가 전달되어 모든 이벤트가 발생하는 것을 차단

메소드	내용
이벤트객체.stopPropagation()	익스플로러 제외한 브라우저
window.event.cancelBubble=true;	익스플로러

JavaScript - 이벤트

KH정보교육원

기본 이벤트제거 / 유효성 검사

- 기본이벤스 : 태그 중 이벤트핸들러를 기본적으로 가지고 있는 것
  - ☞ <a>, <input type='sbmit'> 입력양식에서 많이 사용
- 유효성검사 : 사용자가 입력한 데이터가 양식에 맞는지 검사하는 것
  - ☞ 예) 비밀번호 입력확인

기본이벤트 제거

onsubmit이벤트 속성에 이벤트 핸들러 연결할때 false를 return함

유효성 검사

이벤트 핸들러에 데이터비교 후 맞지 않으면 false를 리턴

JavaScript - 객체

KH정보교육원

객체 선언 / 호출

선언하는 방법

```
var 변수명(객체명) = {
        속성(키값) : 값 ,
        속성(키값) : 값 ,
        속성(키값) : 값
};
```

속성값 접근

```
변수명(객체명)['요소명(키값)'] ;
변수명(객체명)['요소명(키값)'] ;
```

또는

```
변수명(객체명).요소명(키값) ;
변수명(객체명).요소명(키값) ;
```

```
<script>
function test(){
        var testObject = {a:100, b:200};
        console.log(testObject);
        testObject.a = 300;
        testObject['b'] = 400;
        console.log(testObject);
}
</script>
```

변수명.요소명

변수명[요소명]

JavaScript - 객체

KH 정보교육원

in / with 키워드

- in : 객체 내부에 해당 속성이 있는지 확인하는 키워드
- with : 코드를 줄여주는 키워드, 호출시 객체명 생략 가능

in 키워드

```
속성명 in 변수명(객체명) // 있으면 true, 없으면 false
```

with 키워드

```
with(변수명(객체명)){
    속성명;
    속성명;
}
```

with 미사용시

```
변수명(객체명).속성명;
변수명(객체명).속성명;
```

JavaScript - 객체

KH정보교육원

객체 속성 추가 및 삭제

- 속성 및 매소드을 동적으로 추가 및 삭제 가능

추가

```
변수명(객체명).속성명='값';
변수명(객체명).속성명='값';
변수명(객체명).속성명=function (){
                    메소드 로직;
                    [return [리턴값]];
};
```

삭제

```
delete(변수명(객체명).속성명);
```

JavaScript - 객체

KH정보교육원

객체를 이용한 데이터 저장 및 출력

```
<script>
function test(){
        var name  = window.prompt("당신의 이름은? ");
        var age = window.prompt("당신의 나이는? ");
        var addr = window.prompt("당신의 주소는? ");
        var object = {name:name,
                      age:age,
                      addr:addr,
                      print:function(){
                         var str = "이름 : "+this.name +"₩n"+
                                   "나이 : "+this.age +"₩n"+
                                   "주소 : "+this.addr +"₩n"
                          return str;
                      }
       };
        console.log(object.print());
}
</script>
```

JavaScript - 객체

KH정보교육원

객체배열 활용

- 생성된 객체를 배열에 넣어 활용 가능

var 변수명 = [ ];//배열 생성

데이터 대입

변수명.push({ 속성명:'값',속성명:'값',속성명:'값' … });

변수명.push({ 속성명:'값',속성명:'값',속성명:'값' … });

변수명.push({ 속성명:'값',속성명:'값',속성명:'값' … });

JavaScript - 객체

KH정보교육원

생성자 함수

- this키워드를 사용하여 속성을 생성하는 함수
- new라는 키워드를 사용하여 객체 생성
- 생성자명의 첫 글자는 대문자로 시작
- instanceof로 어떤 생성자로 생성된지 확인가능

```
function 생성자명(value1, value2, value3, ....){
                this.속성명='value1',
                this.속성명='value2',
                    ....
        }
}
```

JavaScript - 객체

KH정보교육원

함수활용 객체생성 VS 생성자

- 중복 매소드를 저장하는 방식차이
  - 함수활용 : 중복되는 메소드를 객체별로 만들어서 저장
  - 생성자 : prototype이라는 내부 객체를 이용하여 저장 가능
    - ☞ 하나의 메소드를 이용해서 전체 객체가 다 활용(중복저장 X)

prototype 메소드 생성

변수명(객체명).prototype.메소드명=function( ){ };

JavaScript - 객체

KH정보교육원

캡슐화

- 생성자 함수에서 속성 선언시 this키워드를 사용하지 않고 지역변수로 선언
- this키워드 사용 setter/getter 메소드 작성
  - ☞ 클로저 활용(지역변수를 지우지 않고 사용하는 기능)

```
frunction 생성자명(value1, value2, value3, ....){
                    var 속성명='value1';
                    var 속성명='value2';
                     this.set속성명= function( ) { };
                     this.get속성명= function( ) { };
          }
}
```

JavaScript - 객체

KH정보교육원

상속

- 다른 객체를 상속받아 사용 가능
- 속성으로 객체를 추가하는 방법 / call메소드 이용하는방법

```
frunction 생성자명(value1, value2, value3, ....){
        this.속성명1=상속받을 객체명;
        this.속성명1(value1, value2);//생성자 호출
// call메소드이용
        상속받을 객체명.call(this,value1,value2);
        }
}
```

JavaScript - 함수

KH정보교육원

매개변수(전달인자)

- 호출하는 코드와 호출되는 함수를 연결해주는 변수는 매개변수라 함
- 지정된 매개변수보다 많은 개수 선언하고 호출하는 것을 허용
  - ☞ 초과되는 매개변수는 무시
- 지정된 매개변수보다 적게 선언하고 호출하는 것도 허용
  - ☞ 선언이 안된 매개변수는 undefined로 자동설정 됨.

return [되돌려줄 값]

- return은 함수를 호출한 위치로 돌아가라는 의미
- return값(되돌려줄 값)을 지정하지 않으면 undefined자료형으로 반환됨