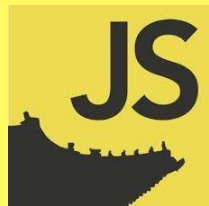


{ } + [ ]



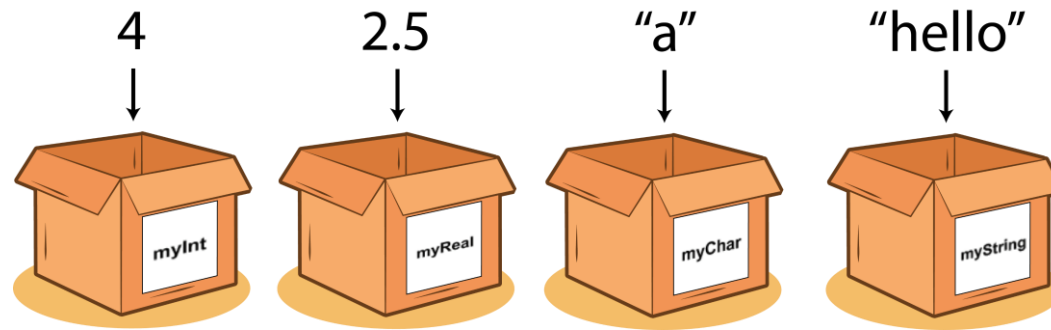
# Javascript 02

- 변수와 동적 UI -

{ Na Daeyoung } + [ [soriru@handong.edu](mailto:soriru@handong.edu) ]

# 변수(variable)

- 변수란?
  - 컴퓨터 내부에 데이터를 임시로 저장하는 공간
  - 데이터를 저장하는 상자라고 생각



# 변수(variable)-선언

- 변수 선언( 키워드 + 변수명 + 변수의 초기값 )

```
<script>  
  let temp=0;  
</script>
```

변수 선언 키워드

변수명

변수의 초기값(생략 가능)

- 변수 선언 키워드

- **let** : 새로 도입된 변수 선언 키워드
- **var** : 오래된 변수 선언 키워드
- **const** : 변경할 수 없는 상수형 변수를 만들 때 사용 (상수, constant)

```
let name = "철수";    // 이름이라는 변수를 만들고 "철수"를 저장  
const age = 20;      // age 변수를 만들고 20 저장 (변경 불가)  
var city = "서울";   // 옛날 방식 (지금은 거의 안 씀)
```

# 변수(variable)-키워드의 비교

- Let, var, const의 비교

구분	var	let	const
등장 시기	아주 오래됨 (ES6 이전)	ES6(2015)부터	ES6(2015)부터
재선언	가능	불가	불가
값 변경	가능	가능	불가
범위(Scope)	함수 단위	블록 단위 {}	블록 단위 {}
호이스팅(끌어올림)	선언이 위로 끌어올려짐 (예상치 못한 동작 가능)	호이스팅은 되지만 "일시적 사각지대(TDZ)" 때문에 안전	호이스팅은 되지만 TDZ 때문에 안전
주 사용 용도	현재는 잘 안 씀 (구 코드 유지용)	바뀔 수 있는 값	바뀌지 않는 값(상수)

# 변수의 이름 지정

- 변수 작성 규칙

- 변수명은 숫자로 시작할 수 없음 `let 1temp; let 3score; //두 변수 모두 오류`

- 변수명에는 공백이 포함되면 안됨 `let tem case; //공백 사용 오류`

- 변수명에는 알파벳, 숫자, \_, \$ 사용가능 `let $temp3; //사용가능`

- 대소문자를 구별 `let current; let Current; //다른 변수로 처리`

- 예약어(keyword)는 사용 불가

- `let, const, if, for, function` 같은 JavaScript 예약어는 이름으로 쓰면 안 됨

- `let function=3; //예약어 오류`

# 변수의 이름 지정

- 변수 작명 관례

- 문법적으로는 강제는 아니지만, 가독성을 위해 **권장되는 규칙**

- camelCase 사용 (단어 연결 시 첫 글자는 소문자, 다음 단어는 대문자)

```
let userName; let totalPrice;  
//camelCase 사용 시작은 소문자, 두 단어 연결시 대문자
```

- 상수는 대문자 + 언더스코어

```
const MAX_VALUE = 100; const PI = 3.14;
```

- 이름은 의미 있게 짓기

```
let a = 20;           // 의미 없음  
let age = 20;         // 의미 있는 이름  
let userAge = 20;     // 더 구체적
```

# 변수의 값 할당

- 변수 생성 시 초기 값 할당( 변수 초기화 )

```
let studentAge = 25 // 변수 생성과 값의 할당이 한 번에 이루어짐 초기화
```

대입 연산자, 할당 연산자 라고 부름  
**'='** 은 같다의 의미가 아님

```
let studentAge=25;  
studentAge=34;  
studentAge=22;
```

*오른쪽의 값을 왼쪽 변수에 대입한다고 해석*

코드가 동작하면 studentAge에는 어떤 값이 있을까?

# 자신의 이름을 변수 저장하여 출력

- 아래 코드를 동작시키면 어떤 결과가?

```
<body>
  <h1>이름을 출력</h1>
  <script>
    let name="한동이";
    document.getElementsByTagName('h1')[0].innerHTML=name;
  </script>
</body>
```

- 이름을 name란 변수를 만들어서 저장
- h1 태그에 자신의 이름이 나타나도록 selector로 지정 후 내용 변경



# 잠깐! - 데이터 타입

- Javascript의 변수는 타입을 가짐
  - Javascript로 변수를 생성하면 어떤 데이터가 저장되어 있는지에 따라서 타입이 정해짐
  - Data type: 데이터가 가지는 형태
    - Number(숫자형): 정수나 실수 형태의 데이터

```
let age1 = 20; // 정수  
let pi = 3.14; // 실수
```

- String(문자형): 문자나 문자열 형태의 데이터

```
let name='A'; //문자  
let greeting='안녕하세요'; //문자열  
let age2="20"; //문자열
```

age1에  
저장된 값과  
age2에  
저장된 값은  
같을까?

# Javascript에서 출력

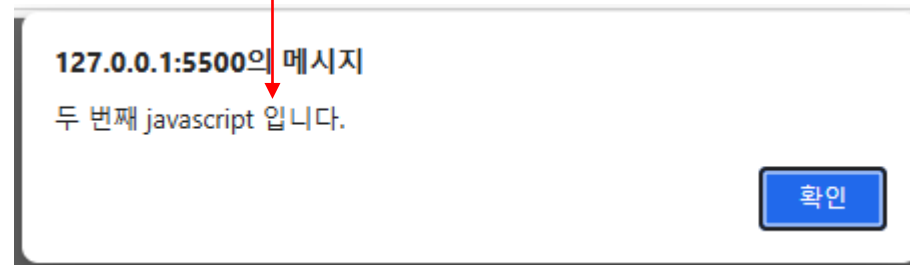
- 화면에 출력하는 함수
  - `window.alert()`: 간단한 메시지용 팝업창으로 출력
  - `window.confirm()`: [확인]과 [취소]버튼이 있어서 사용자의 선택을 받을 수 있음
  - `document.write()`: html 문서에 태그를 직접 출력(잘 사용하지 않음)
  - `innerHTML` 속성: HTML 요소 내부에 출력(실제 화면 출력에 가장 많이 사용)
  - `console.log()`: 개발자 도구 console화면에 출력, 주로 debug용으로 사용
- `window.prompt()`: 출력과 입력을 모두 수행

# Javascript에서 출력-alert()

- window.alert()

- 작은 팝업 창을 띄워서 사용자에게 메시지 출력

```
<body>  
  <script>alert("두 번째 javascript 입니다.");</script>  
</body>
```



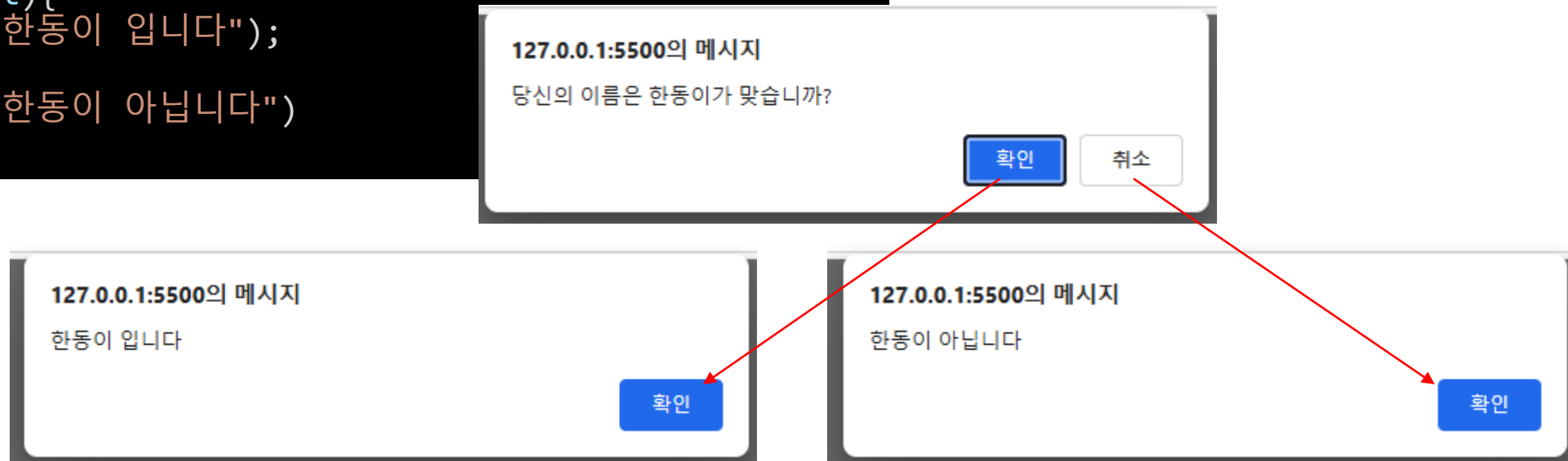
- 화면에 자신의 이름을 alert()으로 출력해보자

# Javascript에서 출력-confirm()

- window.confirm()

- 확인창
- [확인], [취소] 버튼이 있는 작은 팝업 창을 띄우는 용도
- 사용자의 선택(return value)에 따라 다른 동작을 만들 수 있음
  - True[확인] 아니면 false[취소] 값이 반환

```
let checklist=confirm("당신의 이름은 한동이가 맞습니까?");  
  
if(checklist){  
    alert("한동이 입니다");  
}else{  
    alert("한동이 아닙니다")  
}
```

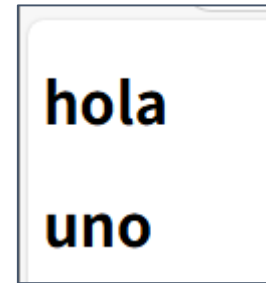


# Javascript에서 출력-document.write()

- document.write()

- 화면에 html 태그를 직접 출력하는 함수

```
<body>
  <h1>hola</h1>
  <script>
    document.write("<h1>uno</h1>");
  </script>
</body>
```



H1태그로 uno 출력

- 자신의 학번을 studentID 변수에 저장하고 h1태그의 크기로 출력하는 코드 작성

# Javascript에서 출력-innerHTML속성

- innerHTML속성

- InnerHTML 속성은 html 요소의 내부의 텍스트를 의미
- 반드시 Selector와 함께 사용하여야 함(document.getElementById() 등)

```
<body>
  <h2 id="print">너의 이름은?</h2>
  <script>
    document.getElementById('print').innerHTML="타키";
  </script>
</body>
```

타키

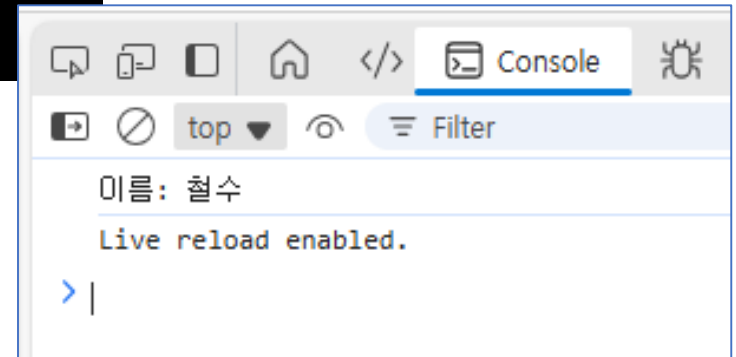
- Html의 특정 요소를 변경하기 때문에 해당 요소가 생성되어 있어야 함
- 문제) 자신의 이름을 h2 태그에 출력하는 코드를 작성

# Javascript에서 출력-console.log()

- console.log()

- 웹브라우저의 개발자 도구 console에 출력하는 함수
- 주로 debug용으로 사용됨

```
let name = "철수";  
console.log("이름:", name); // 이름: 철수
```



# Javascript에서 출력-prompt()

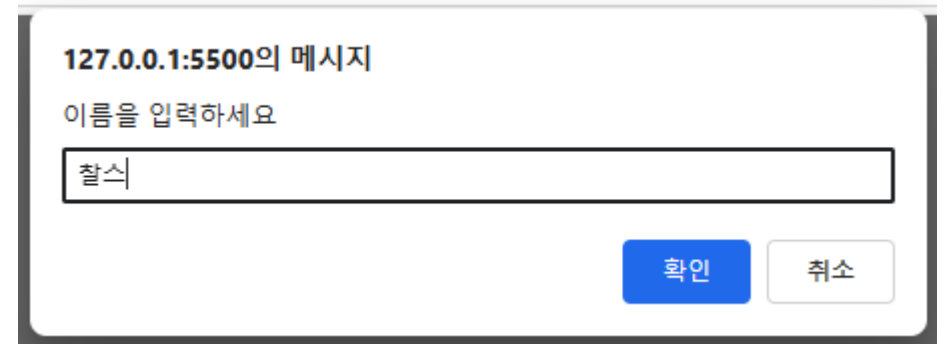
- `window.prompt()`

- 출력과 입력을 같이 실행할 수 있는 역할
- 프롬프트창을 출력
- 사용자로 부터 간단한 값을 입력받을 때 사용
  - 사용자의 입력에 따라 다르게 동작

```
let name=prompt("이름을 입력하세요","찰스");  
if(name=='철수'){  
    alert("hello");  
}
```

출력 메시지

기본 입력값



127.0.0.1:5500의 메시지

이름을 입력하세요

확인 취소



# 변수와 문자열을 같이 출력하려면?

- 변수에 저장되어 있는 값과 문자열을 같이 출력

## 1) '+' 기호로 변수와 문자열 연결

```
<script>
  let studentId=2250001;
  let name="한동이";

  alert("당신의 학번은:"+studentId+"\n당신의 이름은:"+name+"입니다.");
</script>
```

127.0.0.1:5500의 메시지

당신의 학번은:2250001  
당신의 이름은:한동이입니다.

확인

## 2) 템플릿 리터럴

- 백틱(backtick, `)을 사용
- \${} 문법으로 변수나 표현식을 문자열 안에 삽입 가능

```
<h1>더하기 계산</h1>
<h2 id="result">결과</h2>
<script>
  let x = 5;
  let y = 10;

  document.getElementById('result').innerHTML=`${x} + ${y} = ${x + y}`;
</script>
```

더하기 계산

5 + 10 = 15

# 문제- 이름을 입력 받고 출력

- Prompt로 자신의 이름을 입력 받고 p 태그에 출력하는 코드를 작성

127.0.0.1:5500의 메시지

당신의 이름은 무엇인가요?



## JavaScript 입력 & 출력 실습

안녕하세요, 한식이님! 환영합니다

# 문제- 이름을 입력 받고 출력 답안

```
<!DOCTYPE html>
<html lang="ko">
<head>
  <meta charset="UTF-8">
  <title>JavaScript 입력과 출력</title>
</head>
<body>
  <h1>JavaScript 입력 & 출력 실습</h1>

  <!-- 출력될 위치 -->
  <p id="message">여기에 결과가 표시됩니다.</p>

  <script>
    // 1. 사용자에게 이름을 입력받기
    let name = prompt("당신의 이름은 무엇인가요?");

    // 2. p 요소의 내용을 변경하기 (템플릿 리터럴 활용)
    document.getElementById("message").innerHTML = `안녕하세요, ${name}님! 환영합니다`;
  </script>
</body>
</html>
```

# Javascript와 동적 UI

- 동적 UI란?

- Dynamic User Interface
- 사용자의 행동(클릭, 입력, 스크롤 등)에 따라 화면이 즉시 변하고 반응하는 사용자 인터페이스(UI)

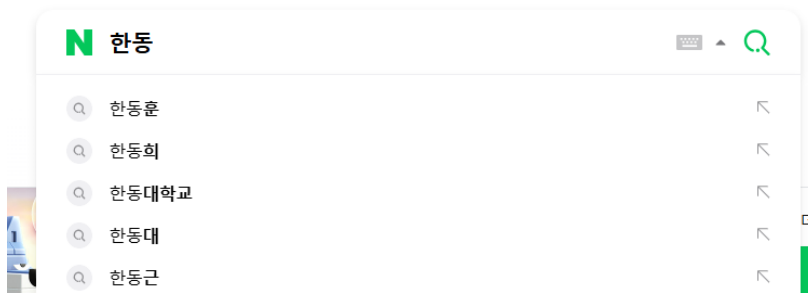
구분	정적 UI	동적 UI
특징	항상 같은 화면	사용자의 행동에 따라 변화
기술	HTML + CSS	HTML + CSS + JavaScript
예시	단순 글자, 이미지	버튼 클릭, 자동완성, 장바구니

- 장점

- 사용자 경험(UX) 향상
- 즉각적인 피드백 제공
- 현대적인 웹사이트의 핵심 요소

# Javascript와 dynamic user interface

- 동적 UI의 예시
  - 검색창 자동완성 (네이버, 구글)
  - 좋아요 버튼 클릭 → 하트 색상 변경
  - 쇼핑몰 장바구니 → 상품 개수 증가
  - 카카오톡 웹 → 메시지 입력하면 바로 반영



# { 감사합니다 } +[ Thankyou ]

{  
본 교재 또는 강좌는 2025년도 과학기술정보통신부 및 정보통신기획평가원에서 주관하여 진행하는 'SW중심대학사업'의  
결과물입니다.  
}