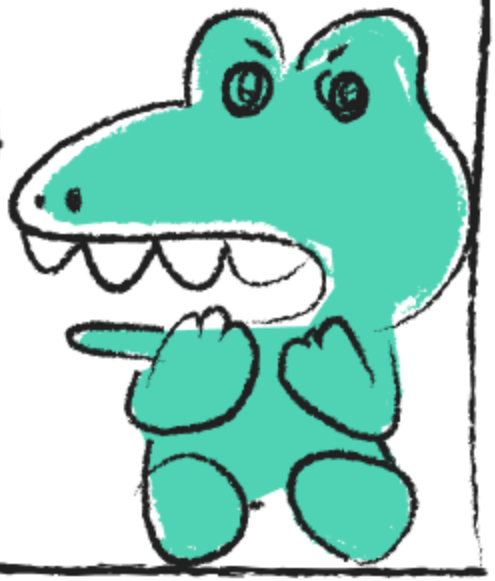
# Java Script 기초

JS의 기초를 뿌셔보자!

# 자바스크립트란?

웹페이지에 생동감을 불어넣기 위해
만들어진 프로그래밍 언어



# 소스코드 위치

# 소스코드 위치

자바스크립트는 일반적으로 HTML 문서의 <head></head> 사이에 위치

그러나 그 외의 위치에 둘수도 있고 외부파일이나 다른 서버를 통해 참조하는 방식도 가능!

- 내부 스크립트 예시

  HTML 문서 내부에 자바스크립트 소스코드를 두는 유형

  <head></head> 혹은 <body></body> 에 둘 수 있으며 양쪽에 모두 있어도 상관 없음

```
<head>
<script>
    alert('자바스크립트는 head 안 script 태그로 감싼 부분에 기술');
</script>
</head>
```

# var

var은 함수 전체에 걸쳐 유효
초기값을 지정하지 않는다면 값이 설정될 때까지 undefined 값을 가짐

```
var i; // 선언, "undefined"가 저장됨

var sum = 0; // 선언과 초기화

var i, sum; // 한 번에 여러 개의 변수를 함께 선언할 수 있음

var i=0, sum=10, message="Hello"; // 선언과 초기화를 동시에 해줄 수 있음

name = "javascript"; // 선언되지 않은 변수는 전역 변수가 됨
```

# let

let은 변수가 선언된 블록, 구문 또는 표현식 내에서만 유효
var 키워드가 블록 범위를 무시하고 전역 변수나 함수 지역 변수로 선언되는 것과 다름

```
let i; // 선언, "undefined"가 저장됨

let sum = 0; // 선언과 초기화

let i, sum; // 한 번에 여러 개의 변수를 함께 선언할 수 있음

let i=0, sum=10, message="Hello"; // 선언과 초기화를 동시에 해줄 수 있음
```

# Const

블록 범위의 상수 선언 ( let과 같은 범위)
상수의 값은 재할당할 수 없으며 다시 선언할 수 없음
처음 선언할 때 반드시 초기화
보통 대문자를 사용해서 선언

```
const MY_NUM = 7;
```

# 출력문

# 브라우저 콘솔창을 이용한 출력

console.log()

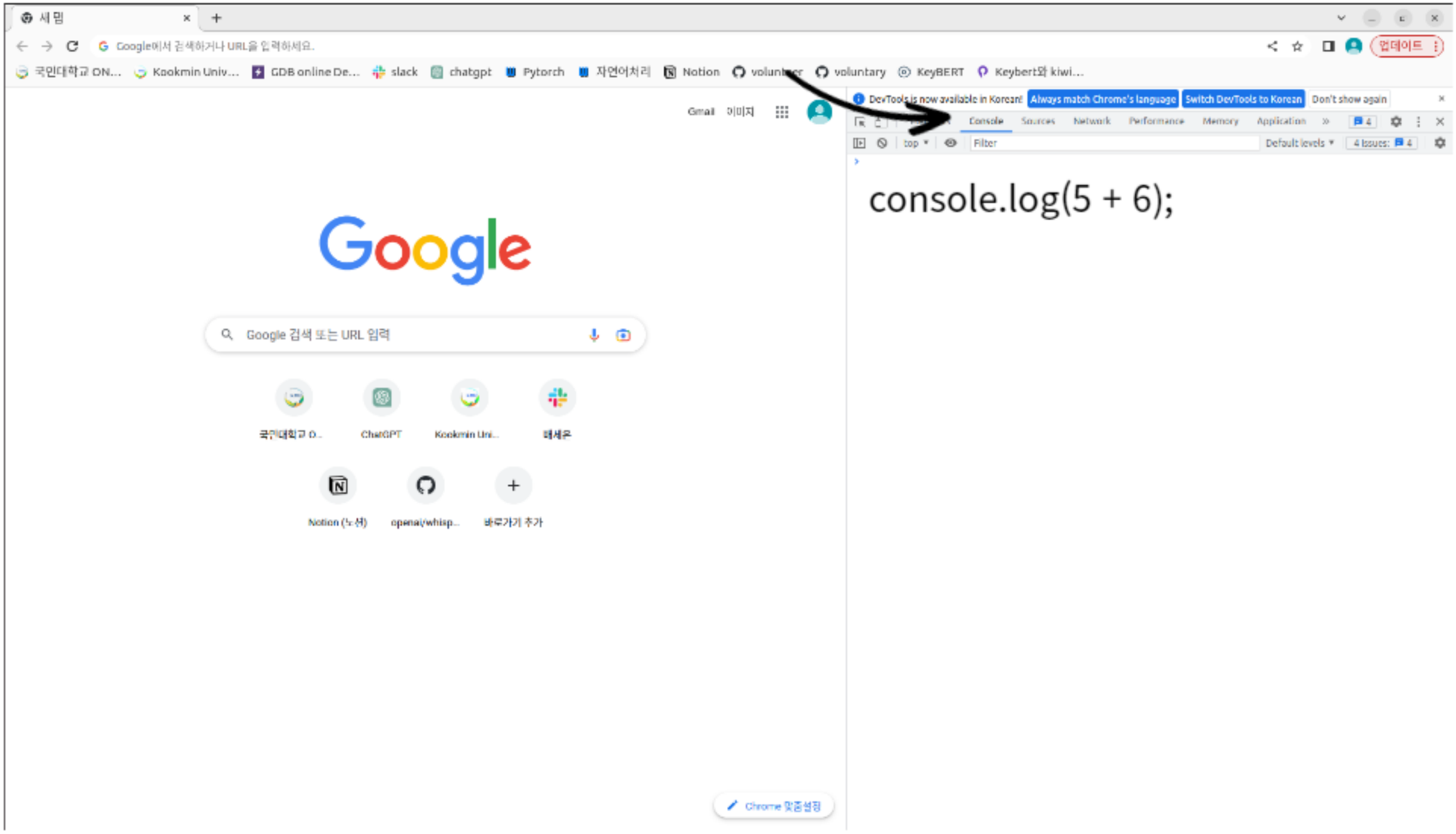
자바스크립트 코드에서 진행상황을 출력하거나 개발을 위해 참고하기 위한 값들을 출력하기 위한 용도로 사용

- 5 + 6 연산결과 웹브라우저 출력 예시

```
<body>
  <script>
    console.log(5 + 6);
  </script>
</body>
```

*script태그 내의 모든 코드 끝에 세미콜론( ;)넣기!

# F12를 누르면 나오는 화면



# HTML 문서에 출력

document.write()

괄호 안에 들어가있는 것을 페이지에 쓰는 함수

- 5 + 6 연산 결과 웹브라우저 출력 예시

```
<body>
  <script>
    document.write(5 + 6);
  </script>
</body>
```

# Alert 창을 이용한 출력

웹브라우저에서 오픈되는 조그만 경고창(alert)을 이용한 출력
보통 프로그램에서 에러, 경고, 사용자 입력을 위해 많이 사용

- 5 + 6 연산결과 웹브라우저 출력 예시

```
<script>
alert(5 + 6);
</script>
```

# HTML 문서의 특정 부분에 출력

HTML 문서의 특정요소를 찾아 해당 콘텐츠를 대체해 출력

JS에서 가장 보편적으로 HTML 문서를 통해 핸들링하는 방법

- 기존 HTML 소스 유지하며 부분적으로 변경하여 5 + 6 연산결과 웹브라우저 출력 예시

```
<body>
    <div id = "result">
    </div>
    <script>
        document.getElementById("result").innerHTML = 5 + 6;
    </script>
</body>
```

# 조건문

C, JAVA와 매우 유사

if, else, switch 등

# if, else if

조건식이 참인 경우 어떤 실행을 할 것인가 결정지을 수 있는 조건문
else 와 결합해 조건 범위나 조건을 세분화하는 것이 가능

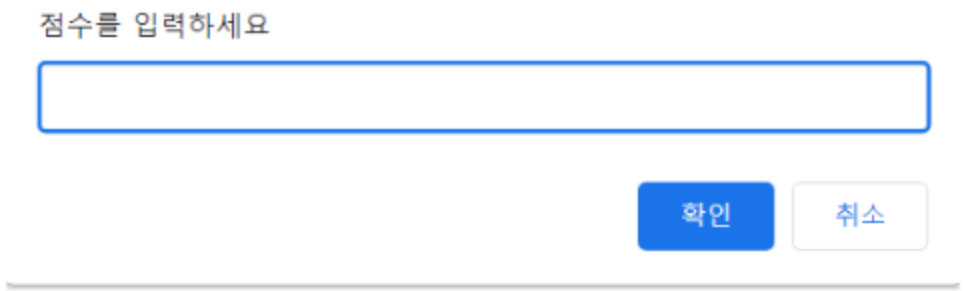
- score 에 따라 해당 점수 구간의 성적 출력 예시

```
var score = 85;

if (score >= 90)
      console.log('A');
else if (score >= 80)
      console.log('B');
else if (score >= 70)
      console.log('C');
else
      console.log('F');
```

# 입력 + if문 예제

```
let score = window.prompt('점수를 입력하세요');
      if (score >= 90)
            console.log('A');
      else if (score >= 80)
            console.log('B');
      else if (score >= 70)
            console.log('C');
      else
            console.log('F');
```



# switch

입력값에 따라 처리를 다르게 하는 경우 사용
내용적으로는 if ~ else if 와 유사

- level에 따라 등급 출력 예시

```
var level = 'B';

switch(level) {
	case 'A' :
		console.log('VIP 등급');break;
	case 'B' :
		console.log('일반 등급');break;
	default :
		console.log('등급없음');break;
}
```

# 반복문

C, JAVA와 매우 유사

for, while, forEach, for-in 등

# for

기본 구조는 시작, 종료조건, 증감식을 가지는 형태

- 세개의 값을 가지는 배열을 선언하고 배열값을 모두 출력하는 예시

```
const colors = ['red', 'blue', 'green'];
for (let i = 0; i < colors.length; i++) {
      console.log( colors[i] );
}
```

# for문 예제 (HTML + JS)

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<style>
		div{
		border: 1px solid blue;
		padding: 10px;
		width: 200px;
		line-height: 1.5em;
	}
	</style>
</head>
<body>
	<script>
		var num = parseInt(prompt("출력할 구구단의 숫자를 입력"));
		document.write("<div>");

		여기에 알맞은 코드를 넣어주세요!

		document.write("</div>");
	</script>
</body>
</html>
```

# for문 예제 (HTML + JS)

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<style>
		div{
		border: 1px solid blue;
		padding: 10px;
		width: 200px;
		line-height: 1.5em;
	}
	</style>
</head>
<body>
	<script>
		var num = parseInt(prompt("출력할 구구단의 숫자를 입력"));
		document.write("<div>");
		for (var i = 1; i <= 9; i++) {
			document.write(num + "*" + i + "=" + (num * i) + "<br>");
		}
		document.write("</div>");
	</script>
</body>
</html>
```

# while

조건이 성립하는 경우 계속 반복하는 구문
무한 루프가 되지 않도록 코드 중간에 조건을 반드시 변경해 주어야 함

```
const colors = ['red', 'blue', 'green'];
var i = 0;
while (colors[i] != null) {
  console.log( colors[i] );
  i++;
}
```

ГОНА

# 대괄호로 만들기

배열 대괄호([])를 사용하여 만드는 방식 (빈 배열 만들기)

```
// 배열 생성 (빈 배열)

var arr = [];

arr[0] = 'zero';
arr[1] = 'one';
arr[2] = 'two';

for (var i = 0; i < arr.length; i++) {
    console.log(arr[i]);
}
```

# 대괄호로 만들기

배열 대괄호([])를 사용하여 만드는 방식 (초기 값 할당)

```
// 배열 생성 (초기 값 할당)

var arr = ['zero', 'one', 'two'];

for (var i = 0; i < arr.length; i++) {
      console.log(arr[i]);
}
```

# 대괄호로 만들기

배열 대괄호([])를 사용하여 만드는 방식 (배열 크기 지정)
값이 할당되지 않아서 undefind 3번 출력

```
// 배열 생성 (배열 크기 지정)
// 쉼표 개수만큼 크기가 지정됨
var arr = [,,,];

for (var i = 0; i < arr.length; i++) {
      console.log(arr[i]);
}
```

# Array()로 만들기

Array()를 사용하여 만드는 방식 (빈 배열 만들기)

```
// 배열 생성 (빈 배열)

var arr = new Array();

arr[0] = 'zero';
arr[1] = 'one';
arr[2] = 'two';

for (var i = 0; i < arr.length; i++) {
    console.log(arr[i]);
}
```

# Array()로 만들기

Array()를 사용하여 만드는 방식 (초기 값 할당)

```
// 배열 생성 (초기 값 할당)

var arr = new Array('zero', 'one', 'two');

for (var i = 0; i < arr.length; i++) {
      console.log(arr[i]);
}
```

# Array()로 만들기

Array()를 사용하여 만드는 방식 (배열 크기 지정)
값이 할당되지 않아서 undefind 숫자만큼 출력

```
// 배열 생성 (배열 크기 지정)
// 숫자만큼 배열 크기가 지정됨
var arr = new Array;

for (var i = 0; i < arr.length; i++) {
      console.log(arr[i]);
}
```

# 함수

# 함수의 특징

- 함수는 function 키워드로 시작
- 함수는 정의/생성할 수 있으며, 출력 가능
- 객체의 특성 덕분에 함수를 변수나 배열에 배정 가능
- 다른 함수를 호출할 때, 인자를 넘겨줄 수 있음

# 함수 표현식

함수 표현식(function expression)은 함수 리터럴(literal)이라고 표현 가능

```
// 함수 리터럴(함수 표현식)이라고 한다.
const hello = function() {
      return 'hllo!';
}

// 변수 hello는 함수 리터럴로 변수를 함수처럼 실행할 수 있다.
hello();
```

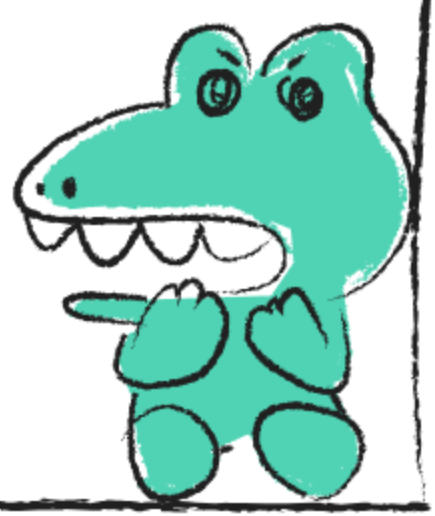
# 함수 예제

함수 표현식(function expression)은 함수 리터럴(literal)이라고 표현 가능

```
const square = function (number) {
  return number * number;
}

const x = square(4);
// `x` 의 값은?
```

# Random



# Math.random()

JS에서 난수를 생성하기 위해서는, Math.random() 함수를 사용
이 함수는 0~1(1은 미포함) 구간에서 부동소수점의 난수를 생성

```
const rand1 = Math.random();
const rand2 = Math.random();
const rand3 = Math.random();

document.write(rand1 + '<br>');
document.write(rand2 + '<br>');
document.write(rand3 + '<br>');
```

# min <= number <= max

함수를 이용해서 범위 지정해서 랜덤 숫자 출력하기

```
const rand = function (min, max) {
    return Math.floor(Math.random() * (max - min + 1)) + min;
}

document.writeln(rand(1, 3));
document.writeln(rand(77, 88));
```

# Math.floor()

랜덤으로 정수를 생성하기 위해서는, Math.floor() 함수를 사용
Math.floor() 함수는 소수점 1번째 자리를 버림하여 정수를 리턴

```
const rand1 = Math.floor(Math.random());
const rand2 = Math.floor(Math.random() * 10);
const rand3 = Math.floor(Math.random() * 4) + 2;

document.write(rand1 + '<br>');    // 항상 0
document.write(rand2 + '<br>');    // 0 <= random <= 9
document.write(rand3 + '<br>');    //  2 <= random <= 5
```

# 함수 활용(feat. 나만의 웹)

```
<div id="quiz">
    <p> 다음중 내가 가장 좋아하는 색깔은??? </p>
    <form name="form">
        <input type = "radio" name = "colorName" value = "v1">빨강<br>
        <input type = "radio" name = "colorName" value = "v2">노랑<br>
        <input type = "radio" name = "colorName" value = "v3">초록<br>
        <input type = "radio" name = "colorName" value = "v4">파랑<br>
        <input type = "button" name = "선택" value = "정답 확인" onclick "checkAnswer()">
    </form>
    <p id="check"></p>
    <script type = "text/javascript">

            // 함수 선언!
            //만약 정답에 해당되는 칸이 체크되어 있다면 정답입니다 출력!
            //아니면 오답입니다 출력

    </script>
</div>
```

마지막으로

발표를 들어주셔서
감사합니다