

13. 자바스크립트와 첫 만남

13-1 자바스크립트로 무엇을 할까

13-2 웹 브라우저가 자바스크립트를 만났을 때

13-3 자바스크립트 용어와 기본 입출력 방법

13-4 자바스크립트 스타일 가이드



자바스크립트로 무엇을 할까

웹 요소를 제어합니다

- 웹 요소를 가져와서 필요에 따라 스타일을 변경하거나 움직이게 할 수 있음
- 웹 사이트 UI 부분에 많이 활용
- 예) 마우스 포인터를 올렸을 때 펼쳐지는 메뉴
한 화면에서 탭을 눌러 내용만 바뀌도록 하는 콘텐츠

다양한 라이브러리를 사용할 수 있습니다

- 웹을 중심으로 하는 서비스가 늘어나면서 브라우저에서 처리해야 할 일이 늘어남 → 라이브러리와 프레임워크가 계속 등장
- 예) 시각화를 위한 d3.js, 머신러닝을 위한 tensorflow.js
DOM 조작을 위한 jQuery 등
- 예) 웹 애플리케이션 개발을 위한 React, Angular, Vue 등

웹 애플리케이션을 만듭니다

- 최근의 웹 사이트는 사용자와 실시간으로 정보를 주고 받으며 애플리케이션처럼 동작
- 예) 온라인 지도의 길찾기 서비스, 데이터 시각화 서비스
공개된 API를 활용한 다양한 서비스

서버를 구성하고 서버용 프로그램을 만들 수 있습니다

- node.js : 프론트엔드 개발에 사용하던 자바스크립트를 백엔드 개발에서 사용할 수 있게 만든 프레임워크

웹 브라우저가 자바스크립트를 만났을 때

웹 문서 안에 자바스크립트 작성하기

- <script> 태그와 </script> 태그 사이에 자바스크립트 소스 작성
- 웹 문서 안의 어디든 위치할 수 있지만, 주로 </body> 태그 앞에 작성
- 자바스크립트 소스가 있는 위치에서 실행됨.

 **Do it!** 내부 자바스크립트 사용하기

예제 파일 13\script-1.html

(... 생략 ...)

```
<body>
  <h1 id="heading">자바스크립트</h1>
  <p id="text">위 텍스트를 클릭해 보세요</p>

  <script>
    var heading = document.querySelector('#heading');
    heading.onclick = function() {
      heading.style.color = "red";
    }
  </script>
</body>
```

(... 생략 ...)



이곳에 자바스크립트 소스를 작성합니다

외부 스크립트 파일 연결해서 작성하기

자바스크립트 소스를 별도의 파일(*.js)로 저장한 후 웹 문서에 연결

기본형 <script src="외부 스크립트 파일 경로"></script>

 **Do it!** 외부 스크립트 사용하기(JS 파일)

예제 파일 13\js\chage-color.js

```
var heading = document.querySelector('#heading');
heading.onclick = function() {
  heading.style.color = "red";
}
```

 **Do it!** 외부 스크립트 사용하기(HTML 파일)

예제 파일 13\script-2.html

(... 생략 ...)

```
<body>
  <h1 id="heading">자바스크립트</h1>
  <p id="text">위 텍스트를 클릭해 보세요</p>

  <script src="js/change-color.js"></script>
</body>
```

(... 생략 ...)



이곳에서 외부 스크립트 파일을 연결합니다.

자바스크립트

위 텍스트를 클릭해 보세요

자바스크립트

위 텍스트를 클릭해 보세요

웹 브라우저가 자바스크립트를 만났을 때

웹 브라우저에서 스크립트를 해석하는 과정



Do it! 웹 브라우저가 스크립트를 해석하는 과정 살펴보기

예제 파일 13\script-1.html

```
1 <!DOCTYPE html>
2 <html lang="ko"> ← HTML 분석기
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>글자색 바꾸기</title>
7   <style>
8     body { text-align: center; } ← CSS 분석기
9     (... 생략 ...)
10  </style>
11 </head>
12 <body>
13   <h1 id="heading">자바스크립트</h1>
14   <p id="text">위 텍스트를 클릭해 보세요</p>
15
16   <script>
17     var heading = document.querySelector('#heading');
18     heading.onclick = function() {
19       heading.style.color = "red";
20     }
21   </script> ← 자바스크립트 해석기
22 </body>
23 </html>
```

- ① 1행에 있는 <!DOCTYPE html>를 보고 현재 문서가 웹 문서라는 것을 인식
→ 이제부터 HTML 표준에 맞춰 소스를 읽기 시작
- ② 웹 문서에서 HTML 태그의 순서와 포함 관계 확인
- ③ 태그 분석이 끝나면 7~14행의 스타일 정보 분석.
- ④ 20행의 <script> 태그를 만나면 자바스크립트 해석기에게 스크립트 소스 넘김
자바스크립트 해석기에서 <script>와 </script> 사이의 소스를 분석하고 해석
- ⑤ ②에서 분석한 HTML과 ③에서 분석한 CSS 정보에 따라 웹 브라우저 화면에 표시
- ⑥ 웹 브라우저에서 사용자가 동작하면 자바스크립트를 실행해서 결과를 화면에 표시

웹 브라우저가 자바스크립트를 만났을 때

웹 브라우저에서 스크립트를 해석하는 과정



Do it! 웹 브라우저가 스크립트를 해석하는 과정 살펴보기

예제 파일 13\script-1.html

```
1 <!DOCTYPE html>
2 <html lang="ko"> ← HTML 분석기
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>글자색 바꾸기</title>
7   <style>
8     body { text-align: center; } ← CSS 분석기
9     (... 생략 ...)
10  </style>
11 </head>
12 <body>
13   <h1 id="heading">자바스크립트</h1>
14   <p id="text">위 텍스트를 클릭해 보세요</p>
15
16   <script>
17     var heading = document.querySelector('#heading');
18     heading.onclick = function() {
19       heading.style.color = "red";
20     }
21   </script> ← 자바스크립트 해석기
22 </body>
23 </html>
```

- ① 1행에 있는 <!DOCTYPE html>를 보고 현재 문서가 웹 문서라는 것을 인식
→ 이제부터 HTML 표준에 맞춰 소스를 읽기 시작
- ② 웹 문서에서 HTML 태그의 순서와 포함 관계 확인
- ③ 태그 분석이 끝나면 7~14행의 스타일 정보 분석.
- ④ 20행의 <script> 태그를 만나면 자바스크립트 해석기에게 스크립트 소스 넘김
자바스크립트 해석기에서 <script>와 </script> 사이의 소스를 분석하고 해석
- ⑤ ②에서 분석한 HTML과 ③에서 분석한 CSS 정보에 따라 웹 브라우저 화면에 표시
- ⑥ 웹 브라우저에서 사용자가 동작하면 자바스크립트를 실행해서 결과를 화면에 표시

자바스크립트 용어와 기본 입출력 방법

식과 문

식(expression)

- 값을 만들어 낼 수 있다면 모두 식이 될 수 있다
- 식은 변수에 저장된다



자바스크립트의 다양한 식 나타내기

```
inch * 2.54 // 연산식은 식입니다.  
"안녕하세요?"; // 문자열도 식입니다.  
5 // 숫자도 식입니다.
```

문(statement)

- 문의 끝에는 세미콜론(;)을 붙여서 구분하는게 좋다
- 넓은 의미에서 식이나 값을 포함할 수 있다


자바스크립트 용어와 기본 입출력 방법

간단한 입출력 방법

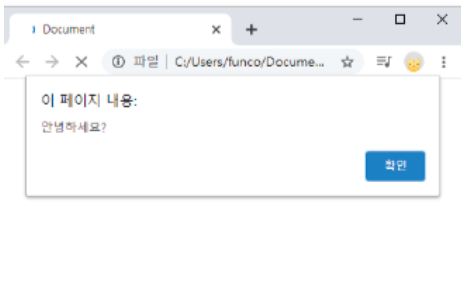
알림 창 출력

'확인' 버튼이 있는 메시지 창 표시

기본형 `alert(메시지)`

 **Do it!** 알림 창 만들기


```
alert("안녕하세요?")
```



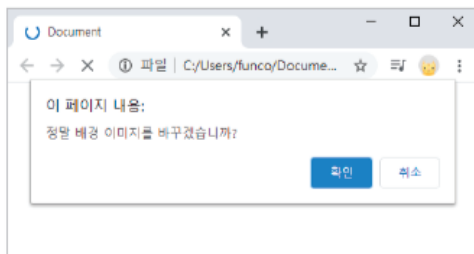
확인 창 출력

- '확인' 과 '취소' 버튼이 있는 창 표시
- 클릭하는 버튼에 따라 프로그램 동작

기본형 `confirm(메시지)`

 **Do it!** 확인 창 만들기


```
var reply = confirm("정말 배경 이미지를 바꾸겠습니까?");
```



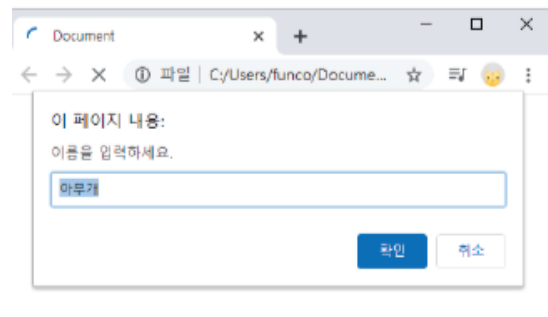
프롬프트 창에서 입력받기

- 텍스트 필드가 있는 창 표시
- 사용자 입력 값을 가져와 프로그램에서 사용

기본형 `prompt(메시지)` 또는 `prompt(메시지, 기본값)`

 **Do it!** 프롬프트 창의 기본값 지정하기

```
var name = prompt("이름을 입력하세요.", "아무개");
```




자바스크립트 용어와 기본 입출력 방법

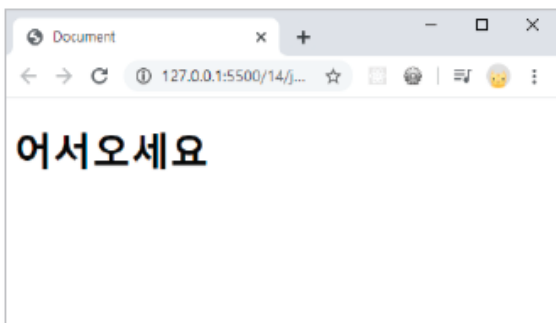
간단한 입출력 방법


document.write()

- 아직 document 객체를 배우지 않았으므로 웹 문서(document)에서 괄호 안의 내용을 표시(write)하는 명령문 이라는 정도로만 알아둘 것
- 괄호 안에서 큰따옴표(" ")나 작은 따옴표(' ') 사이에 입력한 내용은 웹 브라우저 화면에 그대로 표시됨
- 따옴표 안에는 HTML 태그도 함께 사용할 수 있음

 Do it! document.write() 문으로 제목 표시하기

```
<script>
document.write("<h1>어서오세요</h1>");
</script>
```



 Do it! 이름 받아서 화면에 표시하기

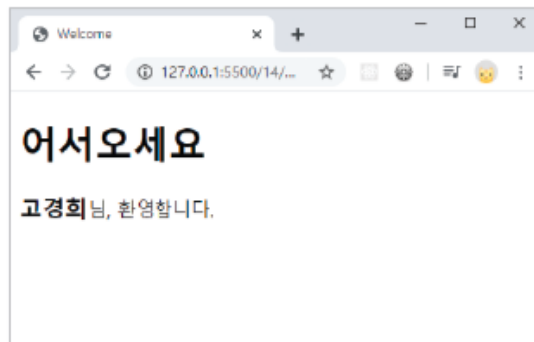
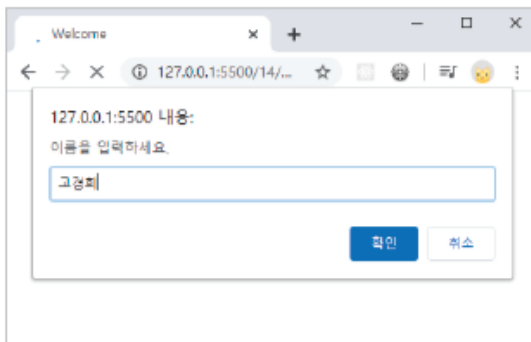
예제 파일 13\greeting.html

```
<script>
var name = prompt("이름을 입력하세요.");
document.write("<b><big>" + name + "</big></b>님, 환영합니다.");
</script>
```

① 프롬프트 창에 입력받은 내용을 name 변수에 저장합니다.

② name의 변수값과 "님, 환영합니다."를 연결해 웹 브라우저 창에 표시합니다.

이름:김윤



자바스크립트 용어와 기본 입출력 방법

간단한 입출력 방법

console.log()

웹 브라우저 개발자 도구 창에 포함되어 있는 창

- 괄호 안의 내용을 콘솔 창에 표시
- 괄호 안에 변수가 들어갈 수도 있고, 따옴표 안에 텍스트를 넣을 수도 있음
- 따옴표 안에 태그는 사용할 수 없음



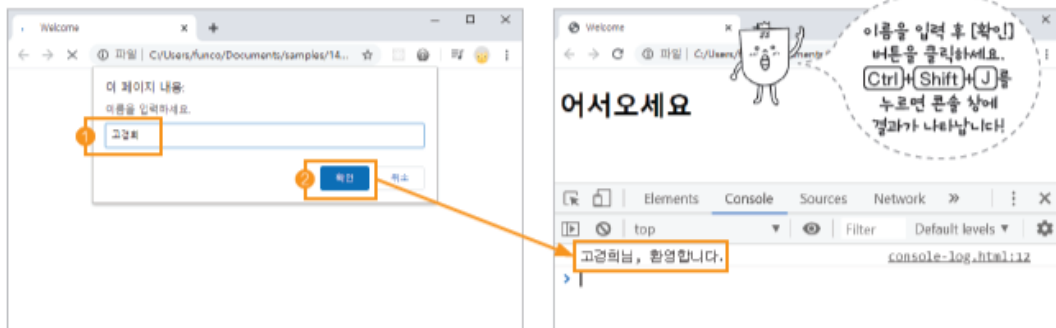
Do it! 이름 받아서 콘솔 창에 표시하기

예제 파일 13\console-log.html

```
<body>
  <h1>어서오세요</h1>

  <script>
    var name = prompt("이름을 입력하세요.");
    console.log(name + "님, 환영합니다.");
  </script>
</body>
```

웹 브라우저에서 콘솔 창을 열려면
웹 브라우저 화면에서
[Ctrl] + [Shift] + [J] 를 누르세요



자바스크립트 스타일 가이드

스타일 가이드란

코딩 규칙을 '스타일 가이드', '코딩 컨벤션', '코딩 스타일', '표준 스타일' 등으로 부름

코딩 규칙이 왜 필요할까?

- 자바스크립트는 다른 프로그래밍 언어에 비해 데이터 유형이 유연해서 오류 발생이 잦다
- 오픈 소스에 기여하거나 누군가와 공유할 소스라면 더욱 깔끔한 소스가 중요하다
- 팀 프로젝트를 진행한다면 통일된 코딩 규칙이 필요하다
- 코딩 규칙에 따라 작성된 웹 사이트는 유지 보수도 수월하고 그만큼 비용도 줄어든다

자바스크립트 스타일 가이드

구글 자바스크립트 스타일 가이드 (google.github.io/styleguide/jsguide.html) 또는

에어비앤비 자바스크립트 스타일 가이드(github.com/airbnb/javascript) 참고

회사 프로젝트의 경우 팀 내에서 상의해서 결정

Google JavaScript Style Guide	
Table of Contents	
1 Introduction	5.8 Control structures
1.1 Terminology notes	5.9 this
1.2 Guide notes	5.10 Equality Checks
	5.11 Disallowed features
2 Source file basics	6 Naming
2.1 File name	6.1 Rules common to all identifiers
2.2 File encoding: UTF-8	6.2 Rules by identifier type
2.3 Special characters	6.3 Camel case: defined
3 Source file structure	7 JSDoc
3.1 License or copyright information, if present	7.1 General form
3.2 @fileoverview, JSDoc, if present	7.2 Markdown
3.3 goog.module statement	7.3 JSDoc tags
3.3.3 goog.module Exports	7.4 Line wrapping
3.4 ES modules	7.5 Top-level comments
3.5 goog.setTestOnly	7.6 Class comments
3.6 goog.require and goog.requireType statements	7.7 Enum and typedef comments
3.7 The file's implementation	7.8 Method and function comments
4 Formatting	7.9 Property comments
4.1 Braces	7.10 Type annotations
4.2 Block indentation: +2 spaces	7.11 Visibility annotations
4.3 Statements	
4.4 Column limit: 80	8 Policies
4.5 Line wrapping	8.1 Issues unspecified by Google Style: Be Consistent!

Airbnb JavaScript Style Guide() {
<i>A mostly reasonable approach to JavaScript</i>
Note: this guide assumes you are using Babel , and requires that you use babel-preset-airbnb or the equivalent. It also assumes you are installing shims/polyfills in your app, with airbnb-browser-shims or the equivalent.
downloads: 16.2M/month downloads: 12M/month gitter join chat
This guide is available in other languages too. See Translation
Other Style Guides
<ul style="list-style-type: none">• ES5 (Deprecated)• React• CSS-in-JavaScript• CSS & Sass• Ruby
Table of Contents
1. Types
2. References
3. Objects
4. Arrays
5. Destructuring
6. Strings
7. Functions

자바스크립트 스타일 가이드

자바스크립트 소스를 작성할 때 지켜야 할 기본 규칙

1. 코드를 보기 좋게 들여쓴다

- 'Tab' 키나 '스페이스바'를 눌러 2칸이나 4칸 들여쓰
- 최근에는 공백 2칸 들여쓰기를 많이 사용함

2. 세미콜론으로 문장을 구분한다

- 세미콜론을 붙일 것을 권장함
- 소스는 한 줄에 한 문장만 작성하는 것이 좋다

세미콜론 규칙

```
var n = 10 // 권장하지 않음
var n = 10; // 권장함
var n = 10; var sum = 0; // 권장하지 않음
```

3. 공백을 넣어 읽기 쉽게 작성한다

식별자나 연산자, 값 사이에 공백을 넣어 읽기 쉽게 작성한다

공백 규칙

```
var sum=num+10; // 권장하지 않음
var sum = num + 10; // 권장함
```

4. 코드를 설명하는 주석을 작성한다

- 한 줄 주석 : 슬래시 2개(//) 바로 뒤에 작성
- 여러 줄 주석 : 여는 기호(/*)를 맨 앞에, 닫는 기호(*/)를 맨 뒤에 넣고 그 사이에 주석 내용을 작성
- 주석 사이에 또다른 주석을 넣을 수 없음



한 줄 주석 작성 방법

```
var today = new Date(); // 현재 날짜 가져오기
var h = today.getHours(); // 시간 추출하기
```



여러 줄 주석 작성 방법

```
/* 현재 날짜를 가져와
   시와 분, 초로 추출하고
   화면에 표시하는 스크립트
*/
function startTime() { ..... }
```

5. 식별자는 정해진 규칙을 지켜 작성한다

- 첫 글자는 반드시 영문자나 언더스코어(_), 달러 기호(\$)로 시작해야 한다
- 두 단어 이상이 하나의 식별자를 만들 때 단어 사이에 공백을 둘 수 없다
- 예약어는 식별자로 사용할 수 없다