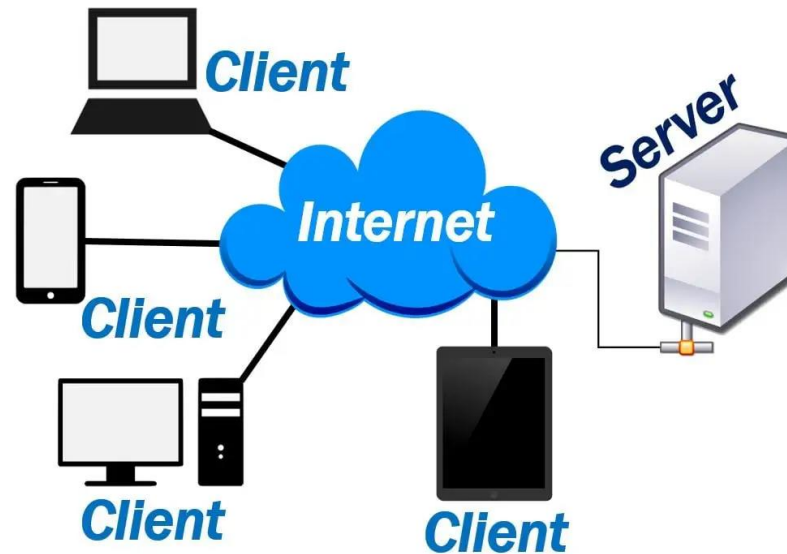




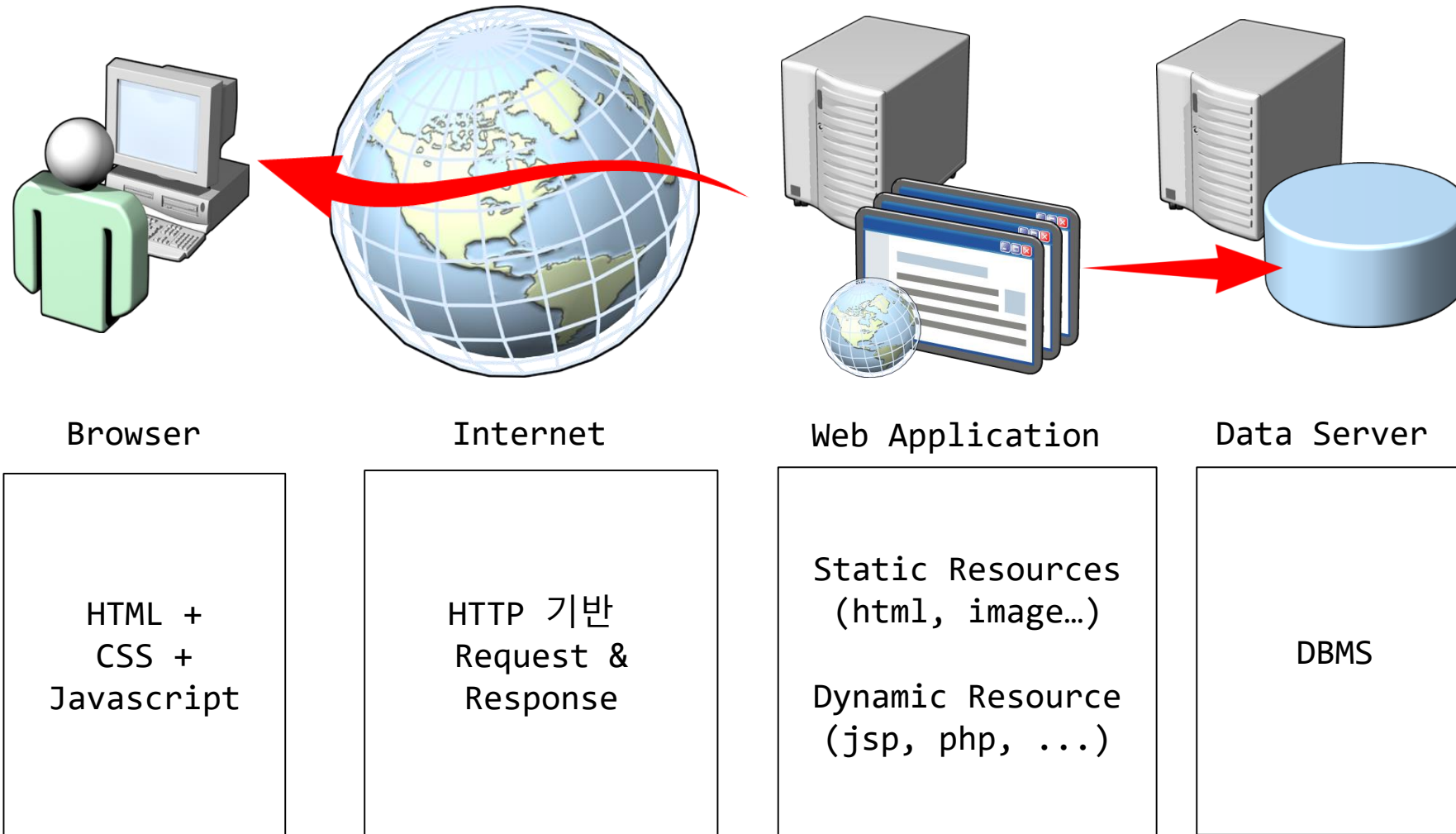
Introduction to WEB

웹 (World Wide Web, WWW)

- 연결된 컴퓨터를 통해 정보를 공유할 수 있는 범세계적 네트워크
- 네트워크로 연결된 다수의 컴퓨터에서 문서 공유
- 웹 문서 → 웹에서 다루는 문서
- 구성
 - » 웹 서버 → 클라이언트의 요청을 수신/처리하고 응답하는 역할
 - » 웹 클라이언트 → 웹 서버에 웹 문서를 요청하고 수신하는 역할

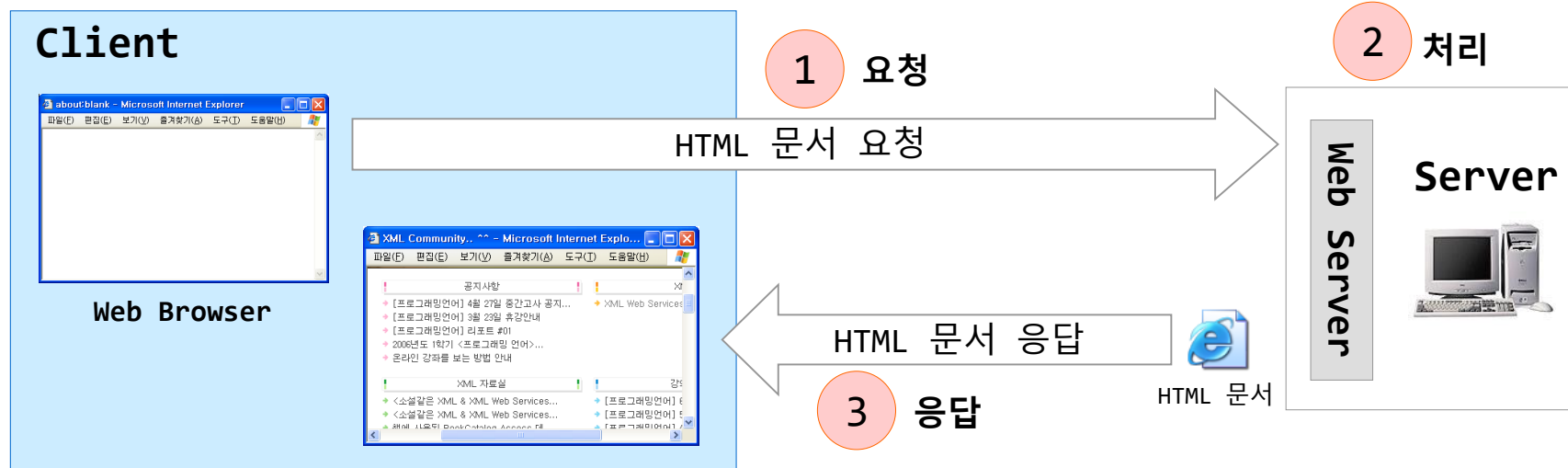


웹 애플리케이션 구조



웹 구성요소

■ 웹 애플리케이션 작동 구조



■ 웹 서버(Web Server)

» 클라이언트의 요청을 받아서 처리한 후 결과를 클라이언트에 전송해 주는 주체

■ 웹 클라이언트(Web Client)

» 필요한 데이터를 웹 서버에 요청하는 주체

» 일반적으로 웹 브라우저(Web Browser)

정적 웹 응용프로그램 vs 동적 웹 응용프로그램

- 정적 웹 응용프로그램

- » HTML, image 등 미리 만들어진 자원으로 사용자의 요청에 대해 응답
- » 미리 생성된 자원이 없으면 응답할 수 없으며
- » 작은 차이라도 각각 다른 페이지로 구성해야 함.
- » 웹 서버 수준에서 응답이 처리됨.

- 동적 웹 응용프로그램

- » 요청이 들어오면 요청에 따라 동적으로 응답할 결과를 생성
- » Web Server는 요청을 받고 적절한 웹 응용프로그램에 요청을 전달
- » CGI 를 이용한 프로세스 기반 웹 응용프로그램 서비스 또는
- » php, jsp, asp 등의 쓰레드 기반 동적 웹 응용프로그램 기술 사용

환경 구축

■ Web Browser

» Microsoft Edge → <https://www.microsoft.com/ko-kr/edge/download>



» Chrome → <https://www.google.com/intl/ko/chrome/>



» Firefox

→ <https://www.mozilla.org/en-US/firefox/new/>



**Get the browser that
protects what's important**

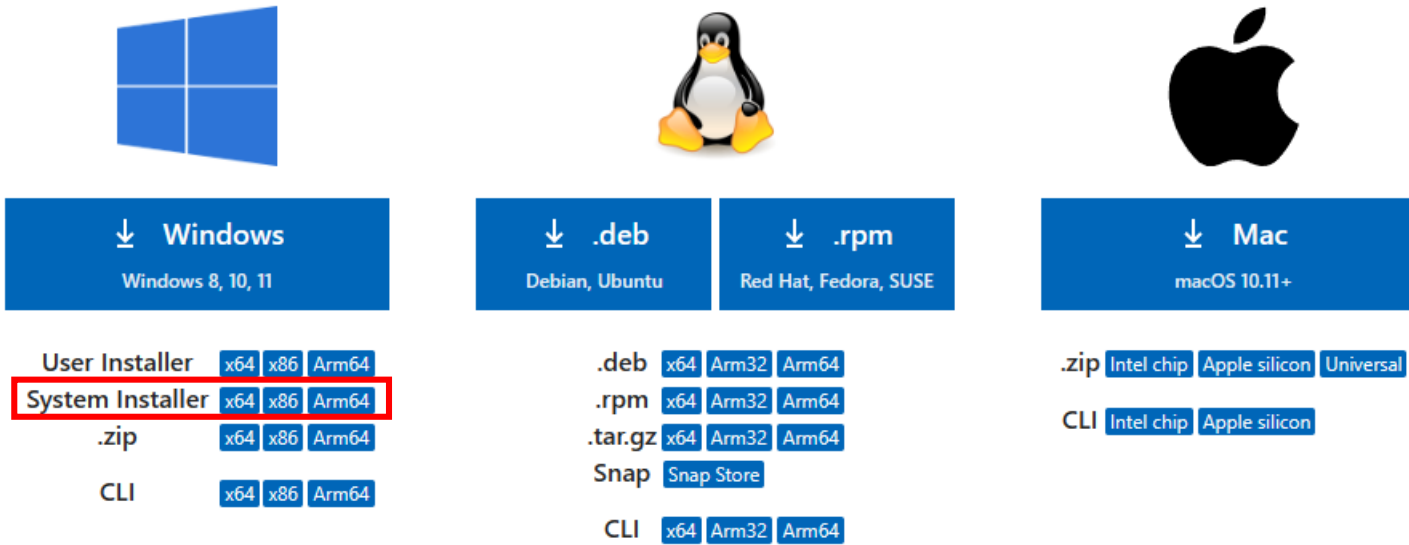
No shady privacy policies or back doors for advertisers. Just a lightning fast browser that doesn't sell you out.

[Download Firefox](#)

환경 구축

- 코드 편집기

- » Visual Studio Code → <https://code.visualstudio.com/#alt-downloads>



- » 다운로드 완료 후 설치 파일 실행

환경 구축

- 자바스크립트 실행 환경
 - » Node.js → <https://nodejs.org/ko>

Node.js®는 Chrome V8 JavaScript 엔진으로 빌드된 JavaScript 런타임입니다.

New security releases to be made available June 20th, 2023

다운로드 Windows (x64)

18.16.0 LTS 안정적, 신뢰도 높음	20.3.0 현재 버전 최신 기능
-----------------------------------	------------------------------

[다른 운영 체제](#) | [변경사항](#) | [API 문서](#) [다른 운영 체제](#) | [변경사항](#) | [API 문서](#)

LTS 일정은 [여기서](#) 확인하세요.

- » 다운로드 완료 후 설치 파일 실행

An abstract graphic on the left side of the slide, consisting of several overlapping, thin-lined triangles in a light beige color. The triangles are of various sizes and orientations, creating a complex, layered geometric pattern. The text "Introduction to HTML5" is positioned to the right of this graphic.

Introduction to HTML5

HTML

- Hyper Text Markup Language
- 웹 페이지 제작을 위한 표준 마크업 언어
- 웹페이지의 구조와 내용을 표현
- 다양한 HTML 요소(Element)로 구성
 - » HTML 요소는 브라우저가 내용을 표시하는 방법을 설명

HTML Version History

년도	버전
1989	WWW
1991	HTML 1.0
1995	HTML 2.0 (HTML Working Group)
1997	HTML 3.2 (W3C Recommendation)
1999	HTML 4.01 (W3C Recommendation)
2000	XHTML 1.0 (W3C Recommendation)
2008	HTML5 First Public Draft
2012	HTML5 Living Standard
2014	HTML5 (W3C Recommendation)
2016	HTML5.1 (W3C Candidate Recommendation)
2017	HTML5.1 2 nd Edition (W3C Recommendation)
2017	HTML 5.2 (W3C Recommendation)

웹 브라우저

- HTML 문서를 읽고 구조와 내용을 화면에 표시하는 도구
- HTML 요소는 표시대상이 아님 → 표시 방법에 대한 정보로 사용



HTML 문서 구조

```
<html>
```

```
<head>
```

```
<title>Page title</title>
```

```
</head>
```

```
<body>
```

```
<h1>This is a heading</h1>
```

```
<p>This is a paragraph.</p>
```

```
<p>This is another paragraph.</p>
```

```
</body>
```

```
</html>
```

HTML

■ 사례

```
<!DOCTYPE html> html5 표준 문서 선언
<html> html root element
<head> html 문서에 대한 정보
<title>Page Title</title> html 문서 제목 → 브라우저 tab에 표시
</head>
<body> html 본문 → 사용자에게 표시되는 영역

<h1>My First Heading</h1> 가장 큰 제목 표시
<p>My first paragraph.</p> 문단 표시

</body>
</html>
```



Document Type Declaration

- 사례

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"  
    "http://www.w3.org/TR/html4/loose.dtd">  
  
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
  
<!DOCTYPE html>
```

- 현재 문서가 따르는 표준을 선언 → 브라우저가 문서를 정확하게 표시하기 위해 필요
- 문서내에 한 번만 작성할 수 있으며 문서 최상단에 작성
- HTML5 문선 선언

```
<!DOCTYPE html>
```

HTML 요소

- html 문서의 기본 구성 요소

- 시작태그, 내용, 종료태그로 구성

`<tagname> Content goes here... </tagname>`

→ 태그 이름은 대소문자를 구분하지 않음

- 사례

```
<h1>My First Heading</h1>
```

```
<p>My first paragraph.</p>
```

```
<br> empty element → 내용을 포함하지 않는 요소 (종료태그 없이 사용)
```

시작태그	요소내용	종료태그
<h1>	My First heading	</h1>
<p>	My first paragraph.	</p>
 	없음	없음

HTML 요소

- Nested Element
 - » HTML 요소 내부에 다른 HTML 요소를 포함할 수 있음
 - » 이를 통해 계층적 문서구조를 만들 수 있음

- Nested Element 사례

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
  <h1>My First Heading</h1>
```

```
  <p>My first paragraph.</p>
```

```
</body>
```

```
</html>
```

<h1>, <p>는 <body>에 포함된 nested elements

HTML 속성 (Attributes)

- HTML 요소에 대한 추가 정보 제공
- 시작태그에 적용
- 일반적으로 name="value" 형식으로 표현
 - » 큰따옴표와 작은따옴표 모두 사용 가능 (한 개의 속성에 혼용해서 사용할 수 없음)
 - » 대소문자 구분하지 않음
- 사례

```
<a href="https://www.w3schools.com">Visit W3Schools</a>
```

```

```

```
<p title="I'm a tooltip" style="color:red;">This is a red paragraph.</p>
```

HTML Comments

- 브라우저가 처리하지 않는 영역 → 화면에 표시되지 않는 영역
- 코드를 설명하기 위해 사용하는 도구
- `<!-- 주석내용 -->` 형식으로 표현
- 사례

```
<!-- This is a comment -->  
<p>This is a paragraph.</p>  
<!-- Comments are not displayed in the browser -->
```

This is a paragraph.

```
<p>This is a paragraph.</p>  
<!-- <p>This is another paragraph </p> -->  
<p>This is a paragraph too.</p>
```

This is a paragraph.

This is a paragraph too.

Head Element

- HTML 문서를 처리하는 데 필요한 다양한 정보(Metadata)를 저장하는 영역으로 화면에 표시되지 않는 요소
- `<head> ... </head>` 로 표현되며 내부에는 `title`, `style`, `meta`, `link`, `script`, `base` 등의 요소 포함
- 사례

```
<head>
  <meta charset="UTF-8">
  <meta name="description" content="Free Web tutorials">
  <meta name="keywords" content="HTML, CSS, JavaScript">
  <meta name="author" content="John Doe">
  <meta http-equiv="refresh" content="5;url=http://developers.google.com">
</head>
```

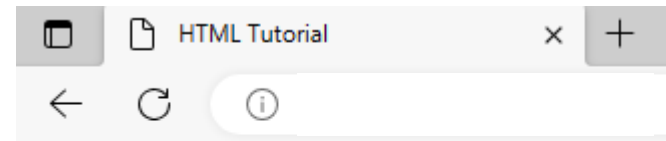
HTML Page Title

- 문서의 제목을 표시하기 위한 HTML 요소
- <head> 요소 내부에 <title>title content</title>로 표현
- 브라우저 탭에 내용이 표시됨
- 사례

```
<!DOCTYPE html>
<html>
<head>
  <title>HTML Tutorial</title>
</head>
<body>

The content of the document.....

</body>
</html>
```



The content of the document.....

HTML Headings

- 제목을 표시하기 위한 HTML 요소
- <h1> ~ <h6> 의 여섯 단계로 표현
- <h1>은 상대적으로 가장 중요한(크게 표현) 요소, <h6>은 상대적으로 가장 덜 중요한(작게 표현) 요소
- 사례

```
<h1>This is heading 1</h1>  
<h2>This is heading 2</h2>  
<h3>This is heading 3</h3>  
<h4>This is heading 4</h4>  
<h5>This is heading 5</h5>  
<h6>This is heading 6</h6>
```

This is heading 1

This is heading 2

This is heading 3

This is heading 4

This is heading 5

This is heading 6

HTML Paragraphs

- 문단을 표시하기 위한 HTML 요소
- `<p>문단내용</p>` 으로 표시
- 문단과 문단 사이에 구분을 위한 여백 적용

HTML Paragraphs

- 사례

```
<p>
This paragraph contains
a lot of lines in the source code,
but the browser ignores it.
</p>
<p>
This paragraph contains      a lot of spaces in the source      code,
but the      browser ignores it.
</p>
<p>
The number of lines in a paragraph depends on the size of the browser window. If
you resize the browser window, the number of lines in this paragraph will change.
</p>
```

This paragraph contains a lot of lines in the source code, but the browser ignores it.

This paragraph contains a lot of spaces in the source code, but the browser ignores it.

The number of lines in a paragraph depends on the size of the browser window. If you resize the browser window, the number of lines in this paragraph will change.

HTML Horizontal Rules

- 주로 문서의 내용을 구분하기 위해 사용
- `<hr>` 로 표현
- 사례

```
<h1>This is heading 1</h1>  
<p>This is some text.</p>  
<hr>
```

```
<h2>This is heading 2</h2>  
<p>This is some other text.</p>  
<hr>
```

```
<h2>This is heading 2</h2>  
<p>This is some other text.</p>
```

This is heading 1

This is some text.

This is heading 2

This is some other text.

This is heading 2

This is some other text.

HTML Line Breaks

- 줄바꿈(개행)을 표현하는 요소
-
 로 표현
- 사례

```
<p>This is<br>a paragraph<br>with line breaks.</p>
```

This is
a paragraph
with line breaks.

HTML Preformatted Text

- 복잡한 개행과 공백을 효과적으로 표현하는 요소
- <pre> 로 표현
- 사례

```
<p>
```

```
My Bonnie lies over the ocean.
```

```
My Bonnie lies over the sea.
```

```
My Bonnie lies over the ocean.
```

```
Oh, bring back my Bonnie to me.
```

```
</p>
```

My Bonnie lies over the ocean. My Bonnie lies over the sea. My Bonnie lies over the ocean. Oh, bring back my Bonnie to me.

```
My Bonnie lies over the ocean.
```

```
My Bonnie lies over the sea.
```

```
My Bonnie lies over the ocean.
```

```
Oh, bring back my Bonnie to me.
```

```
<pre>
```

```
My Bonnie lies over the ocean.
```

```
My Bonnie lies over the sea.
```

```
My Bonnie lies over the ocean.
```

```
Oh, bring back my Bonnie to me.
```

```
</pre>
```

HTML Text Formatting

- 텍스트를 다양한 방식으로 표현하는 요소
- 종류

태그	표현내용	태그	표현내용
	Bold Text (Bold)	<small>	Smaller Text
	Important Text (Bold)		Deleted Text (strike through)
<i>	Italic Text (italic)	<ins>	Inserted Text (underline)
	Emphasized Text (italic)	<sub>	Subscript Text
<mark>	Makred Text (highlighted)	<sup>	Superscript Text

- 사례

<p>Do not forget to buy <mark>milk</mark> today.</p>	Do not forget to buy milk today.
<p>My favorite color is blue <ins>red</ins>.</p>	My favorite color is blue red.
<p>This is _{subscripted} text.</p>	This is subscripted text.
<p>This is ^{superscripted} text.</p>	This is superscripted text.

HTML Quotation and Citation

- 다양한 방식의 인용을 표현하는 요소
- 종류

태그	표현내용	태그	표현내용
<blockquote>	인용된 영역 표현 (indentation)	<address>	연락처 정보 (<i>italic</i>)
<q>	짧은 인용 표현 (quotation mark)	<cite>	창작물의 제목 (<i>italic</i>)
<abbr>	약어 표현	<bdo>	출력 방향 반전

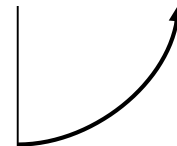
- 사례

<p>Here is a quote from WWF's website:</p>

```
<blockquote cite="http://www.worldwildlife.org/who/index.html">
For 60 years, WWF has worked to help people and nature thrive. As
the world's leading conservation organization, WWF works in nearly
100 countries. At every level, we collaborate with people around
the world to develop and deliver innovative solutions that protect
communities, wildlife, and the places in which they live.
</blockquote>
```

Here is a quote from WWF's website:

For 60 years, WWF has worked to help people and nature thrive. As the world's leading conservation organization, WWF works in nearly 100 countries. At every level, we collaborate with people around the world to develop and deliver innovative solutions that protect communities, wildlife, and the places in which they live.



HTML Quotation and Citation

■ 사례 (계속)

Browsers usually insert quotation marks around the q element.

WWF's goal is to: "Build a future where people live in harmony with nature."

```
<p>Browsers usually insert quotation marks around the q element.</p>
```

```
<p>WWF's goal is to: <q>Build a future where people live in harmony with nature.</q></p>
```

```
<p>The <abbr title="World Health Organization">WHO</abbr> was founded in 1948.</p>
```

```
<p>Marking up abbreviations can give useful information to browsers, translation systems and search-engines.</p>
```

The WHO was founded in 1948.

Marking up abbreviations can give useful information to browsers, translation systems and search-engines.

```
<p>The HTML address element defines contact information (author/owner) of a document or article.</p>
```

```
<address>
```

```
Written by John Doe.<br>
```

```
Visit us at:<br>
```

```
Example.com<br>
```

```
Box 564, Disneyland<br>
```

```
USA
```

```
</address>
```

The HTML address element defines contact information (author/owner) of a document or article.

Written by John Doe.

Visit us at:

Example.com

Box 564, Disneyland

USA

HTML Quotation and Citation

■ 사례 (계속)

```
<p>The HTML cite element defines the title of a work.</p>  
<p>Browsers usually display cite elements in italic.</p>
```

```
  
<p><cite>The Scream</cite> by Edvard Munch. Painted in 1893.</p>
```

The HTML cite element defines the title of a work.

Browsers usually display cite elements in italic.



The Scream by Edvard Munch. Painted in 1893.

```
<p>If your browser supports bi-directional override (bdo), the next line will be written from right  
to left (rtl):</p>
```

```
<bdo dir="rtl">This line will be written from right to left</bdo>
```

If your browser supports bi-directional override (bdo), the next line will be written from right to left (rtl):

tfel ot thgir morf nettirw eb lliw enil sihT

HTML Links

- 다른 웹 페이지로의 이동을 표현하는 요소
- `link text` 로 표현
- Target 속성
 - » 이동할 페이지를 표시할 위치 설정

속성 값	의미	속성 값	의미
<code>_self</code>	같은 브라우저의 같은 탭에 표시	<code>_parent</code>	상위 프레임에 표시
<code>_blank</code>	새 탭에 표시	<code>_top</code>	최상위 body에 표시

- 사례

```
<h2>The target Attribute</h2>
```

```
<a href="https://developers.google.com/" target="_blank">Visit Google!</a>
```

```
<p>If target="_blank", the link will open in a new tab.</p>
```

The target Attribute

[Visit Google!](#)

If target="_blank", the link will open in a new tab.

경로 표시

- src, href 등의 속성에 적용하는 경로는 두 가지 표현 방법으로 작성
 - » 상대 경로와 절대 경로

- 상대 경로

- » 현재 경로를 기준으로 대상 경로 표현

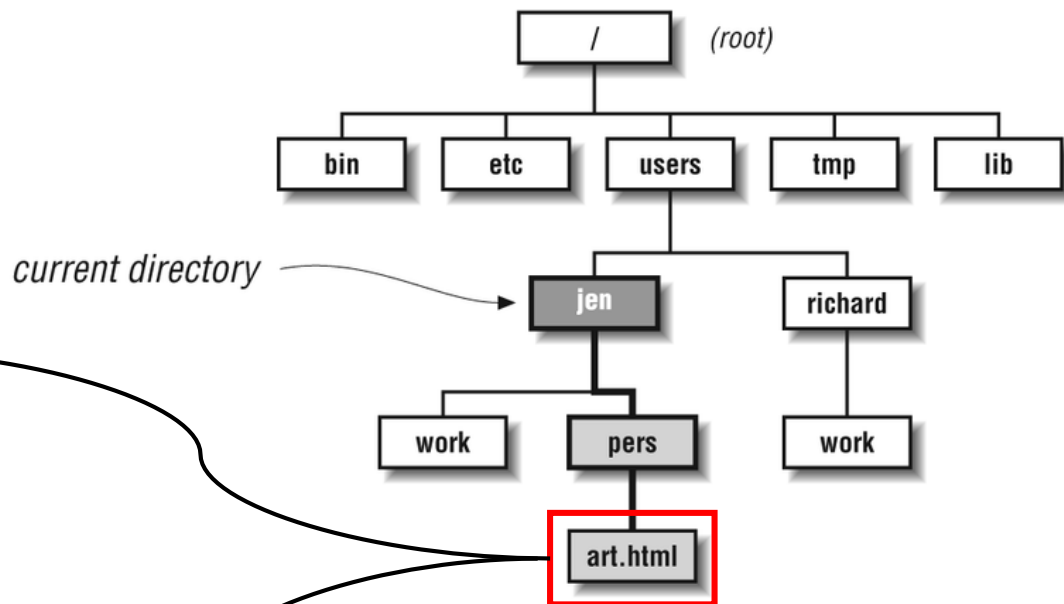
./pers/art.html
or
pers/art.html

- 절대 경로

- » 최상위 경로를 기준으로 대상 경로 표현
 - » 외부 웹사이트의 경로는 항상 절대 경로 사용

/user/jen/pers/art.html

http://www.google.com



Bookmarks

- 같은 페이지 내에서 이동 표현
- `link text` 로 표현
- 사례

```
<p><a href="#C4">Jump to Chapter 4</a></p>
<p><a href="#C10">Jump to Chapter 10</a></p>
```

```
<h2>Chapter 1</h2>
<p>This chapter explains ba bla bla</p>
```

...

```
<h2 id="C4">Chapter 4</h2>
<p>This chapter explains ba bla bla</p>
```

...

[Jump to Chapter 4](#)

[Jump to Chapter 10](#)

Chapter 1

This chapter explains ba bla bla

Chapter 2

This chapter explains ba bla bla

Chapter 3

This chapter explains ba bla bla

Chapter 4

This chapter explains ba bla bla

HTML Images

- Image를 표시하는 요소
- `` 로 표현
- 사례

```
<h2>HTML Image</h2>  

```

HTML Image



HTML Images

- Image Format

약어	설명	확장자
APNG	Animated Portable Network Graphics	.apng
GIF	Graphics Interchange Format	.gif
ICO	Microsoft Icon	.ico, .cur
JPEG	Joint Photographic Expert Group Image	.jpg, .jpeg, .jfif, .pjpeg, .pjp
PNG	Portable Network Graphics	.png
SVG	Scalable Vector Graphics	.svg

HTML Pictures

- 실행되는 시점에 최적의 이미지를 선택적으로 표시하는 요소
- `<picture><source media="device-info" srcset="image-path"></picture>` 로 표현
- 사례

```
<h2>The picture Element</h2>
```

```
<picture>           실행하는 시점의 브라우저 너비에 따라 다른 이미지 표시
  <source media="(min-width: 650px)" srcset="img_food.jpg">
  <source media="(min-width: 465px)" srcset="img_car.jpg">
  
</picture>
```

The picture Element

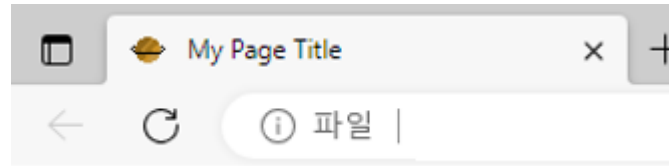


HTML Favicon

- 브라우저의 페이지 타이틀 옆에 작은 이미지를 표시하는 요소
- 파일 위치
 - » 웹사이트 최상위 경로의 favicon.ico 파일은 자동 인식
 - » 다른 경로에 저장한 경우 명시적으로 경로 설정 (사례 참고)
- 사례

```
<!DOCTYPE html>
<html>
<head>
  <title>My Page Title</title>
  <link rel="icon" type="image/x-icon" href="/images/favicon.ico">
</head>
<body>

</body>
</html>
```



HTML Favicon

- 지원되는 Favicon 형식

Browser	ICO	PNG	GIF	JPEG	SVG
Edge	Yes	Yes	Yes	Yes	Yes
Chrome	Yes	Yes	Yes	Yes	Yes
Firefox	Yes	Yes	Yes	Yes	Yes
Opera	Yes	Yes	Yes	Yes	Yes
Safari	Yes	Yes	Yes	Yes	Yes

HTML Tables

- 행과 열로 데이터를 표시하는 요소
- `<table>`, `<tr>`, `<td>`, `<th>`, `<thead>`, `<tbody>`, `<tfoot>` 등의 태그를 조합해서 표현
- 구조

The diagram illustrates the structure of an HTML table. It features a 3x4 table with the following content:

column title 1	column title 2	column title 3	column title 4
data 1-1	data 1-2	data 1-3	data 1-4
data 2-1	data 2-2	data 2-3	data 2-4
data 3-1	data 3-2	data 3-3	data 3-4

Annotations in the diagram include:

- A blue arrow labeled `table` pointing to the entire table structure.
- An orange arrow labeled `th` pointing to the first header cell (`column title 1`).
- A green arrow labeled `td` pointing to the first data cell in the second row (`data 1-3`).
- A red arrow labeled `tr` pointing to the third row of the table.

HTML Tables

■ 사례

```
<!DOCTYPE html>
<html>
<style>
table, th, td { border:1px solid black; }
</style>
<body>
```

```
<h2>A basic HTML table</h2>
```

```
<table style="width:100%">
  <tr>
    <th>Company</th><th>Contact</th><th>Country</th>
  </tr>
  <tr>
    <td>Alfreds Futterkiste</td><td>Maria Anders</td><td>Germany</td>
  </tr>
  <tr>
    <td>Centro comercial Moctezuma</td><td>Francisco Chang</td><td>Mexico</td>
  </tr>
</table>
```

```
</body>
</html>
```

A basic HTML table

Company	Contact	Country
Alfreds Futterkiste	Maria Anders	Germany
Centro comercial Moctezuma	Francisco Chang	Mexico

HTML Tables

- 셀 병합
 - » colspan → 열 병합
 - » rowspan → 행 병합

```
<!DOCTYPE html>
<html>
<body>
<table border="1" style="width:100%">
  <tr>
    <td colspan="2">data 1-1 + 1-2</td>
    <td>data 1-3</td>
  </tr>
  <tr>
    <td>data 2-1</td>
    <td>data 2-2</td>
    <td rowspan="2">data 2-3 + 3-3</td>
  </tr>
  <tr>
    <td>data 3-1</td>
    <td>data 3-2</td>
  </tr>
</table>
</body>
</html>
```

data 1-1 + 1-2		data 1-3
data 2-1	data 2-2	data 2-3 + 3-3
data 3-1	data 3-2	

HTML Lists

- 관련 있는 내용을 목록으로 표시하는 요소
- , , 등을 조합해서 표현
 - » ul → 순서 없는 목록
 - » ol → 순서 있는 목록
- 사례

```
<h2>An Unordered HTML List</h2>
<ul>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ul>
```

An Unordered HTML List

- Coffee
- Tea
- Milk

```
<h2>An Ordered HTML List</h2>
<ol>
  <li>Coffee</li>
  <li>Tea</li>
  <li>Milk</li>
</ol>
```

An Ordered HTML List

1. Coffee
2. Tea
3. Milk

HTML Lists

- 순서 없는 목록 항목 표시 종류

Value	Description
disc	Sets the list item marker to a bullet (default)
circle	Sets the list item marker to a circle
square	Sets the list item marker to a square
none	The list items will not be marked

- 순서 있는 목록 항목 표시 종류

Type	Description
type="1"	The list items will be numbered with numbers (default)
type="A"	The list items will be numbered with uppercase letters
type="a"	The list items will be numbered with lowercase letters
type="I"	The list items will be numbered with uppercase roman numbers
type="i"	The list items will be numbered with lowercase roman numbers

HTML Block and Inline Elements

- HTML 요소는 block 또는 inline 방식 중 하나로 표시됨
- block level elements
 - » 한 행에 한 개의 요소만 표시 → 다른 요소와 같은 행에 표시될 수 없음

<address>	<article>	<aside>	<blockquote>	<canvas>	<dd>
<div>	<dl>	<dt>	<fieldset>	<figcaption>	<figure>
<footer>	<form>	<h1>-<h6>	<header>	<hr>	
<main>	<nav>	<noscript>		<p>	<pre>
<section>	<table>	<tfoot>		<video>	

- inline level elements
 - » 한 행에 여러 개의 요소 표시 → 다른 요소와 같은 행에 표시 가능

<a>	<abbr>	<acronym>		<bdo>	<big>
 	<button>	<cite>	<code>	<dfn>	
<i>		<input>	<kbd>	<label>	<map>
<object>	<output>	<q>	<samp>	<script>	<select>
<small>			<sub>	<sup>	<textarea>
<time>	<tt>	<var>			

<div> 요소

- 다른 HTML 요소의 컨테이너 역할 수행
- 화면에 특정 영역을 만들고 여러 개의 HTML 요소를 모아서 표시하는 용도로 사용
- 사례

```
<div style="background-color:black;color:white;
          padding:20px;float:left;width:40%">
  <h2>Area 1</h2>
  <p>background of this area is black.</p>
  <p>text-color of this area is white</p>
</div>
<div style="background-color:white;color:black;
          padding:20px;float:left;width:40%">
  <h2>Area 2</h2>
  <p>background of this area is black.</p>
  <p>text-color of this area is white</p>
</div>
```

Area 1

background of this area is
black.

text-color of this area is white

Area 2

background of this area is
black.

text-color of this area is white

 요소

- 문서 또는 텍스트의 일부에 스타일을 적용하기 위해 사용
- 사례

```
<h1>The span element</h1>
```

```
<p>My mother has <span style="color:blue;font-weight:bold;">blue</span> eyes and my  
father has <span style="color:darkolivegreen;font-weight:bold;">dark green</span>  
eyes.</p>
```

The span element

My mother has blue eyes and my father has dark green eyes.

HTML Layout Elements

- 웹 페이지의 각 부분을 정의하기 위해 사용하는 다양한 요소
- 종류



요소	설명
<code><header></code>	문서 또는 섹션의 헤더 정의
<code><nav></code>	여러 개의 네비게이션 링크 집합 정의
<code><section></code>	문서의 일반적인 구획을 정의
<code><article></code>	문서 안에서 독립적으로 구분해 배포하거나 재사용하는 구획
<code><aside></code>	문서의 주요 내용과 간접적으로만 연관된 부분 정의 (사이드바)
<code><footer></code>	문서 또는 섹션의 푸터 정의
<code><details></code>	사용자가 보이기/숨기기를 선택할 수 있는 세부 내용 표시 영역 정의
<code><summary></code>	<code>details</code> 요소의 요약 또는 레이블 정의

HTML Audio

- 오디오를 표시하는 요소
- `<audio>`, `<source>` 등을 조합해서 표현
- 사례

```
<audio controls>  
  <source src="horse.ogg" type="audio/ogg">  
  <source src="horse.mp3" type="audio/mpeg">  
  Your browser does not support the audio element.  
</audio>
```



HTML Video

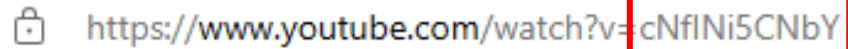
- 비디오를 표시하는 요소
- <video>, <source> 등을 조합해서 표현
- 사례

```
<video width="320" height="240" controls>  
  <source src="movie.mp4" type="video/mp4">  
  <source src="movie.ogv" type="video/ogg">  
  Your browser does not support the video tag.  
</video>
```



HTML YouTube

- iframe 요소를 사용해서 YouTube 영상 표시
- YouTube 영상에 부여되는 Video ID 필요

A screenshot of a web browser's address bar showing the URL `https://www.youtube.com/watch?v=cNfINi5CNbY`. The video ID `cNfINi5CNbY` is highlighted with a red rectangular box.

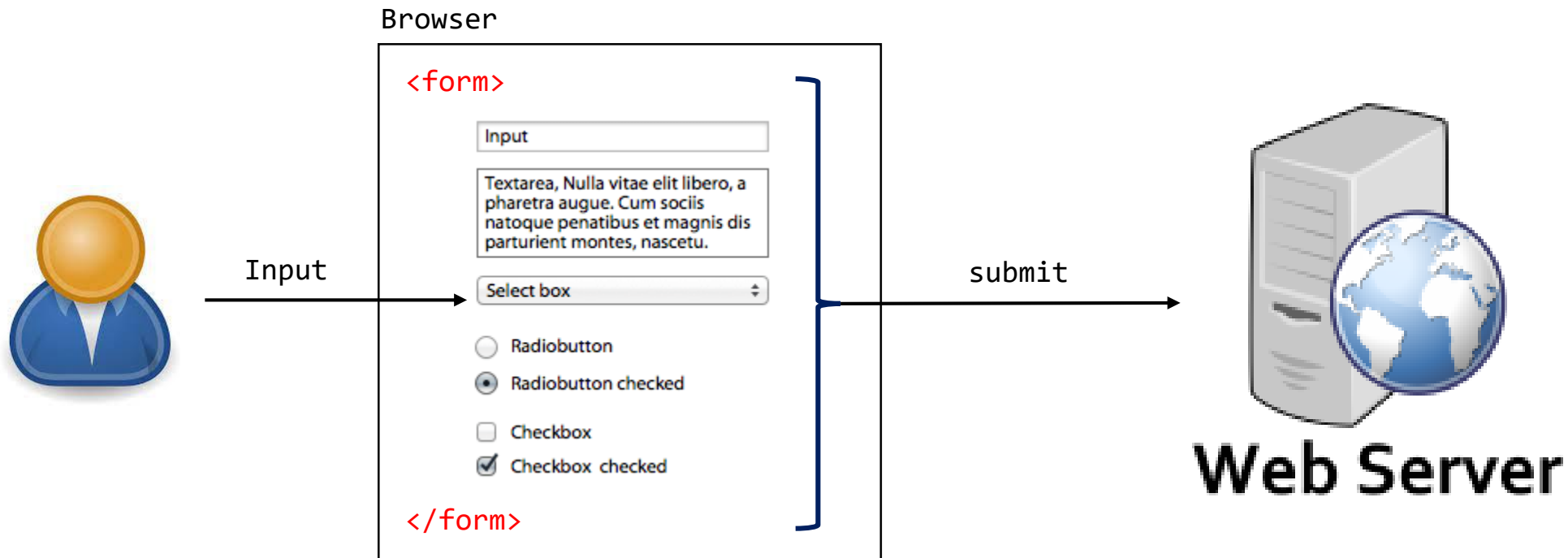
- 사례

```
<iframe width="420" height="345"  
      src="https://www.youtube.com/embed/cNfINi5CNbY">  
</iframe>
```



HTML Forms

- 사용자 입력을 받는 요소
- `<input>`, `<select>`, `<textarea>`, `<form>` 등을 조합해서 표현
- `<form>`은 개별 입력 요소를 모아서 데이터 집합을 구성하고 서버로 전송하는 역할 수행



HTML Forms

- Form 주요 속성

속성	설명
action	form을 전송할 대상 url 지정 (입력 데이터를 처리할 서버의 처리기 위치)
method	데이터 전송 방식 (GET or POST)
enctype	POST 방식 전송 데이터의 인코딩 방법 지정
autocomplete	사용자의 기존 입력값을 기반으로 자동완성 수행 여부 설정
novalidate	서버로 전송할 때 입력 데이터 유효성 검사 수행 여부 설정

HTML Forms

■ 입력 요소 종류

- `<input>`
- `<label>`
- `<select>`
- `<textarea>`
- `<button>`
- `<fieldset>`
- `<legend>`
- `<datalist>`
- `<output>`
- `<option>`
- `<optgroup>`

■ `<input>` 요소 type 종류

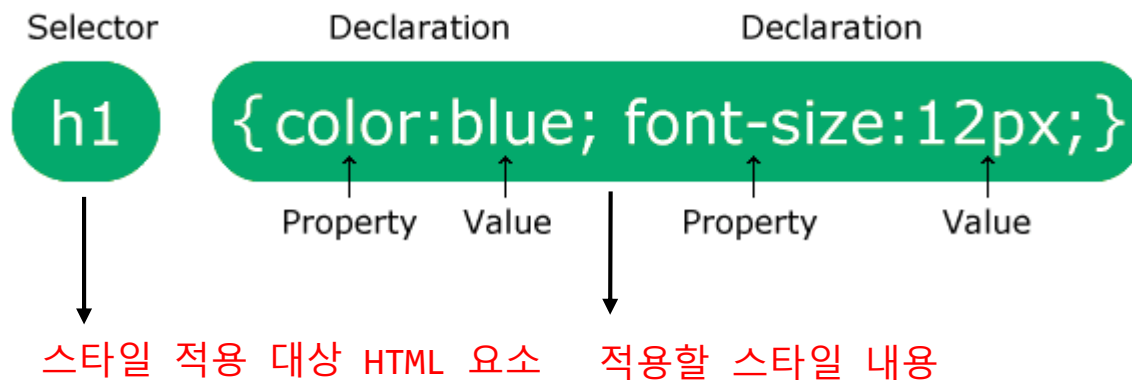
- `<input type="button">`
- `<input type="checkbox">`
- `<input type="color">`
- `<input type="date">`
- `<input type="datetime-local">`
- `<input type="email">`
- `<input type="file">`
- `<input type="hidden">`
- `<input type="image">`
- `<input type="month">`
- `<input type="number">`
- `<input type="password">`
- `<input type="radio">`
- `<input type="range">`
- `<input type="reset">`
- `<input type="search">`
- `<input type="submit">`
- `<input type="tel">`
- `<input type="text">`
- `<input type="time">`
- `<input type="url">`
- `<input type="week">`



Introduction to CSS

CSS (Cascading Style Sheet)

- 웹 페이지에 스타일을 적용하기 위한 도구
- 웹 페이지가 컴퓨터 모니터를 포함한 다양한 기기에서 표현되는 방법을 설명
- HTML 요소로부터 스타일을 분리해서 다수의 위치 또는 다수의 페이지에 반복적으로 적용되는 스타일을 효과적, 집중적으로 관리할 수 있는 도구
- CSS 구문
 - » 선택자와 선언 블록으로 구성



스타일 적용 방식

적용 방식	설명
Inline CSS	요소(태그)에 style 속성을 사용해서 직접 스타일 적용
Internal CSS	<style>...</style> 태그 영역에 스타일 정의 같은 페이지의 모든 요소에 대해 일괄 적용
External CSS	별도의 css 파일을 만들고 <link rel="stylesheet" href="..."> 태그로 적용 여러 개의 HTML 페이지에 일괄 적용

```
<h1 style="color:blue;text-align:center;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>
```

```
<style>
body {
  background-color: linen;
}
h1 {
  color: maroon;
  margin-left: 40px;
}
</style>
```

mystyle.css

```
body {
  background-color: lightblue;
}

h1 {
  color: navy;
  margin-left: 20px;
}
```

```
<link rel="stylesheet" href="mystyle.css">
```

CSS 선택자

- 직접 또는 패턴을 통해 HTML 요소를 지정하는 도구로 스타일 적용 대상을 선택하기 위해 사용
- 종류

선택자 카테고리		설명
단순선택자	name, id, class 등을 기반으로 HTML 요소 선택	
연결선택자	요소들 간의 관계를 기반으로 HTML 요소 선택	
가상클래스선택자	요소의 특정 상태를 기반으로 HTML 요소 선택	
가상요소선택자	요소의 일부만 선택	
속성선택자	속성과 속성의 값을 기반으로 HTML 요소 선택	

단순 선택자

- 종류

선택자	사례	설명
전역 선택자	*	모든 요소 선택
요소(태그) 선택자	p div	태그 이름이 p인 요소 선택 태그 이름이 div인 요소 선택
아이디 선택자	#firstname	id="firstname"인 요소 선택
클래스 선택자	.intro p.intro	class="intro"인 요소 선택 class="intro"인 p요소 선택
그룹 선택자	div, p #firstname, .intro	태그 이름이 div와 p인 요소 선택 id="firstname"인 요소와 class='intro'인 요소 선택

전역 선택자

- 문서의 모든 HTML 요소를 선택
- 사례

```
<style>
* {
  text-align: center;
  color: blue;
}
</style>
```

```
<h1>Hello world!</h1>
```

```
<p>Every element on the page will be affected by the
style.</p>
<p id="para1">Me too!</p>
<p>And me!</p>
```

Hello world!

Every element on the page will be affected by the style.

Me too!

And me!

요소(태그) 선택자

- HTML 요소 이름 (태그 이름)으로 선택
- 사례

```
<style>  
p {  
  text-align: center;  
  color: red;  
}  
</style>
```

Every paragraph will be affected by the style.

Me too!

And me!

```
<p>Every paragraph will be affected by the style.</p>  
<p id="para1">Me too!</p>  
<p>And me!</p>
```

id 선택자

- HTML 요소의 id속성으로 선택
- 사례

```
<style>  
p {  
  text-align: center;  
  color: red;  
}  
</style>
```

```
<p>Every paragraph will be affected by the style.</p>  
<p id="para1">Me too!</p>  
<p>And me!</p>
```

Every paragraph will be affected by the style.

Me too!

And me!

class 선택자

- HTML 요소의 class속성으로 선택
- 사례

```
<style>  
.center {  
  text-align: center;  
  color: red;  
}  
</style>
```

Red and center-aligned heading

Red and center-aligned paragraph.

```
<h1 class="center">Red and center-aligned heading</h1>  
<p class="center">Red and center-aligned  
paragraph.</p>
```

그룹 선택자

- 여러 개의 선택자를 묶어서 일괄 처리
- 사례

```
<style>
h1, h2, p {
  text-align: center;
  color: red;
}
</style>
```

```
<h1>Hello World!</h1>
<h2>Smaller heading!</h2>
<p>This is a paragraph.</p>
```

Hello World!

Smaller heading!

This is a paragraph.

연결 선택자

- 종류

선택자	사례	설명
descendant	space	모든 하위 요소 선택
child	>	한 단계 아래의 하위 요소 선택
adjacent sibling	+	바로 다음에 나오는 형제 요소 선택
general sibling	~	뒤에 나오는 모든 형제 요소 선택

후손(descendant) 선택자

- 어떤 요소의 모든 하위 요소 선택
- 사례

```
<style>
div p {
  background-color: yellow;
}
</style>
```

```
<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
  <section><p>Paragraph 3 in the div.</p></section>
</div>
```

```
<p>Paragraph 4. Not in a div.</p>
<p>Paragraph 5. Not in a div.</p>
```

Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3 in the div.

Paragraph 4. Not in a div.

Paragraph 5. Not in a div.

자식(child) 선택자

- 어떤 요소의 1단계 하위 요소 선택
- 사례

```
<style>
div > p {
  background-color: yellow;
}
</style>
```

Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3 in the div (inside a section element).

Paragraph 4 in the div.

```
<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
  <section>
    <!-- not Child but Descendant -->
    <p>Paragraph 3 in the div (inside a section element).</p>
  </section>
  <p>Paragraph 4 in the div.</p>
</div>
<p>Paragraph 5. Not in a div.</p>
<p>Paragraph 6. Not in a div.</p>
```

Paragraph 5. Not in a div.

Paragraph 6. Not in a div.

Adjacent Sibling 선택자

- HTML 문서 계층에서 어떤 요소와 같은 레벨이면서 바로 다음에 나오는 요소 선택
- 사례

```
<style>
div + p {
  background-color: yellow;
}
</style>
```

Paragraph 1 in the div.

Paragraph 2 in the div.

Paragraph 3. After a div.

```
<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
</div>
<p>Paragraph 3. After a div.</p>
<p>Paragraph 4. After a div.</p>
<div>
  <p>Paragraph 5 in the div.</p>
  <p>Paragraph 6 in the div.</p>
</div>
<p>Paragraph 7. After a div.</p>
<p>Paragraph 8. After a div.</p>
```

Paragraph 4. After a div.

Paragraph 5 in the div.

Paragraph 6 in the div.

Paragraph 7. After a div.

Paragraph 8. After a div.

General Sibling 선택자

- HTML 문서 계층에서 어떤 요소와 같은 레벨이면서 다음에 나오는 모든 요소 선택
- 사례

```
<style>
div ~ p {
  background-color: yellow;
}
</style>
```

Paragraph 1.

Paragraph 2.

Paragraph 3.

```
<p>Paragraph 1.</p>
```

Some code.

```
<div>
  <p>Paragraph 2.</p>
</div>
```

Paragraph 4.

```
<p>Paragraph 3.</p>
<code>Some code.</code>
<p>Paragraph 4.</p>
```

가상 클래스 선택자

- 특정 상태에 있는 HTML 요소의 선택
- 특정 상태 사례
 - » 마우스 포인터가 위에 위치한 요소
 - » 과거에 방문했던 페이지에 대한 링크 또는 처음 방문하는 페이지에 대한 링크
 - » 키보드 입력이 활성화된 입력 요소

- 구문

```
selector:pseudo-class {  
    property: value;  
}
```

가상 클래스 선택자

■ 사례

```
<style>
/* unvisited link */
a:link {
  color: red;
}
/* visited link */
a:visited {
  color: green;
}
/* mouse over link */
a:hover {
  color: hotpink;
}
/* selected link */
a:active {
  color: blue;
}
</style>
```

```
<style>
div {
  background-color: green;
  color: white;
  padding: 25px;
  text-align: center;
}

div:hover {
  background-color: blue;
}
</style>
```

```
<style>
p:first-child {
  color: blue;
}
</style>
```

가상 요소 선택자

- HTML 요소의 일부분 선택
- 일부분 사례
 - » 요소의 첫 번째 문자 또는 첫 번째 행
 - » 요소 내용의 앞 또는 뒤
- 구문

```
selector::pseudo-element {  
  property: value;  
}
```


가상 요소 선택자

■ 사례

```
<style>
p::first-letter {
  color: #ff0000;
  font-size: xx-large;
}
p::first-line {
  color: #0000ff;
  font-variant: small-caps;
}
</style>
```



<p>You can combine the ::first-letter and ::first-line pseudo-elements to add a special effect to the first letter and the first line of a text!</p>

YOU CAN COMBINE THE ::FIRST-LETTER AND ::FIRST-LINE PSEUDO-ELEMENTS TO ADD A SPECIAL EFFECT to the first letter and the first line of a text!



```
<style>
h1::before {
  content: url(smiley.gif);
}
h1::after {
  content: url(smiley.gif);
}
</style>
```

<h1>This is a heading</h1>
<p>The ::after pseudo-element inserts content after the content of an element.</p>

<h1>This is a heading</h1>

 **This is a heading** 

The ::after pseudo-element inserts content af

 **This is a heading** 

속성 선택자

- HTML 요소의 속성 및 속성 값으로 선택
- 구문

선택자	사례	설명
[attribute]	[target]	target 속성이 있는 요소 선택
[attribute=value]	[target="_blank"]	target 속성의 값이 _blank인 요소 선택
[attribute~=value]	[title~="flower"]	title 속성의 값이 독립된 단어로 flower를 포함하는 요소 선택
[attribute =value]	[lang "en"]	lang 속성의 값이 en이거나 en-으로 시작하는 요소 선택
[attribute^=value]	[href^="https"]	href 속성의 값이 https로 시작하는 요소 선택
[attribute\$=value]	[href\$=".pdf"]	href 속성의 값이 .pdf로 끝나는 요소 선택
[attribute*=value]	[href*="google"]	href 속성의 값이 google을 포함하는 요소 선택

An abstract graphic on the left side of the slide, consisting of several overlapping, irregular polygons and lines in a light beige color. The shapes are layered, creating a complex, geometric pattern that resembles a stylized, abstract drawing or a series of overlapping frames.

Introduction to Javascript

Javascript

- 동적 웹 페이지 개발에 사용되는 스크립트 언어
- 특징
 - » 스크립트 언어
 - » 상대적으로 초기 학습곡선 완만
 - » 객체활용 언어
 - » cross-platform 언어
- 적용 분야
 - » 웹 클라이언트 애플리케이션 개발
 - » 웹 서버 애플리케이션 개발
 - » 모바일 애플리케이션 개발
 - » 데스크탑 애플리케이션 개발
 - » 데이터베이스 관리

Javascript

- 1995년 Brendan Eich에 의해 개발된 후 1997년 ECMA 표준으로 전환
- Version History

버전	공식명칭	발표년도	비고
ES1	ECMAScript 1	1997	첫 번째 버전
ES2	ECMAScript 2	1998	editorial changes
ES3	ECMAScript 3	1999	
ES4	ECMAScript 4		출시되지 않음
ES5	ECMAScript 5	2009	
ES6	ECMAScript 2015	2015	

» 2016년부터 년도로 버전명이 지정되며 매년 수정 버전 발표

Javascript

■ ES5 브라우저 지원

Browser	Version	From Date
Chrome	23	Nov 2012
Firefox	21	May 2013
IE	9*	Mar 2011
IE / Edge	10	Sep 2012
Safari	6	Jul 2012
Opera	15	Jul 2013

■ ES6 브라우저 지원

Browser	Version	Date
Chrome	51	May 2016
Firefox	52	Mar 2017
Edge	14	Aug 2016
Safari	10	Sep 2016
Opera	38	Jun 2016

Javascript in Web Client Programming

- 사용자 입력 처리
- 웹 페이지 내용 및 모양의 동적 제어
- 브라우저 제어
- 웹 서버와 통신
- 브라우저 기반 웹 애플리케이션 구현

Javascript 적용

- 적용 방식

적용 방식	설명
Inline JS	요소(태그) 이벤트 속성에 javascript 코드 작성. (권장하지 않음)
Internal JS	<script>...</script> 태그 영역에 javascript 코드 작성
External JS	별도의 js 파일을 만들고 <script src="js file path"><script> 태그로 적용

» <script></script> 는 HTML의 <head> 또는 <body> 영역에 횟수에 제한 없이 작성 가능

Javascript 적용

■ 적용 방식 사례 1, 2

```
<head>
<script>
function myFunction1() {
  document.getElementById("demo").innerHTML = "Paragraph changed 1.";
}
</script>
</head>
<body>
<p id="demo">A Paragraph.</p>
<button type="button" onclick="myFunction1()">Try it 1</button>
<button type="button" onclick="myFunction2()">Try it 2</button>

<script>
function myFunction2() {
  document.getElementById("demo").innerHTML = "Paragraph changed 2.";
}
</script>
</body>
```

Paragraph changed 1.

Try it 1

Try it 2

Paragraph changed 2.

Try it 1

Try it 2

Javascript 적용

- 적용 방식 사례 3

myscript.js

```
function myFunction() {  
    document.getElementById("demo").innerHTML = "Paragraph changed.";  
}
```

```
<h2>Demo External JavaScript</h2>  
  
<p id="demo">A Paragraph.</p>  
  
<button type="button" onclick="myFunction()">Try it</button>  
  
<p>This example links to "myScript.js".</p>  
<p>(myFunction is stored in "myScript.js")</p>  
  
<script src="myScript.js"></script>
```

출력

- 화면에 정보를 출력하는 다양한 방법 제공

명령	기능
<code>innerHTML</code>	HTML 요소의 콘텐츠로 내용 출력
<code>document.write()</code>	브라우저 화면에 직접 출력
<code>window.alert()</code>	메시지 팝업으로 출력 (사용자 확인 필요)
<code>window.confirm()</code>	메시지 팝업으로 출력 (확인 / 취소 선택 필요)
<code>window.print()</code>	현재 문서를 프린터로 출력
<code>console.log()</code>	브라우저 콘솔 창에 출력

기본 구문 규칙

- 대소문자 구분
- 주석 → 브라우저가 해석 및 실행하지 않는 영역
 - » 라인 주석 (단일 행 주석) → // 여기부터 이 행 끝까지 주석
 - » 영역 주석 (다중 행 주석) → /* 주석 내용 */
- Literals
 - » 값 표현
 - » 10, 12.34, true, 'Hello', ...
- 표현식 (Expressions)
 - » 값을 반환하는 구문
 - » 값, 변수, 연산자의 조합
 - » 5 * 10, x * 20, "John" + " " + "Doe"

기본 구문 규칙

- 키워드 (예약어)

- » Javascript 문법에서 사용하는 특별한 의미를 갖는 어휘

abstract	arguments	await*	boolean	in	instanceof	int	interface
break	byte	case	catch	let*	long	native	new
char	class*	const	continue	null	package	private	protected
debugger	default	delete	do	public	return	short	static
double	else	enum*	eval	super*	switch	synchronized	this
export*	extends*	false	final	throw	throws	transient	true
finally	float	for	function	try	typeof	var	void
goto	if	implements	import*	volatile	while	with	yield

기본 구문 규칙

- 식별자 (identifiers, names)
 - » 변수, 함수, 클래스 등에 부여하는 이름
- 식별자 명명 규칙
 - » 문자, \$, _, 숫자 사용
 - » 숫자로 시작할 수 없음
- 식별자 표기 방식
 - » underscore → name, user_name
 - » pascal → Name, UserName
 - » camel → name, userName

기본 구문 규칙

- 실행문 (statements)
 - » 컴퓨터 프로그램이 실행하는 명령(instruction) 단위
 - » 실행문은 값, 연산자, 표현식, 키워드, 주석으로 구성됨
 - » Javascript 프로그램은 실행문의 목록으로 구성
- Semicolons (;)
 - » 문장의 끝 표시 (선택적)
 - » 실행문을 구분하는 역할
 - » 한 줄에 여러 개의 실행문을 작성할 경우 반드시 ;으로 실행문 구분
- 공백
 - » Javascript는 공백 무시
 - » 코드의 가독성을 높이기 위해 적절한 위치에 공백과 줄바꿈 사용

기본 구문 규칙

- 코드 블록
 - » 여러 개의 실행문을 모아서 하나의 단위로 구성
 - » { 실행문1; 실행문2; ... } 형식으로 표현

```
function myFunction() {  
    document.getElementById("demo1").innerHTML = "Hello Dolly!";  
    document.getElementById("demo2").innerHTML = "How are you?";  
}
```


변수 (Variables)

- 데이터를 저장하는 공간
- var, let, const 사용

```
var x = 5;      let x = 5;      const x = 5;    const price1 = 5;  
var y = 6;      let y = 6;      const y = 6;    const price2 = 6;  
var z = x + y;  let z = x + y;  const z = x + y; let total = price1 + price2;
```

- let
 - » ES6(2015)에서 도입
 - » let으로 선언된 변수는 값 변경 가능
 - » 재선언 불가능
 - » 사용하기 전에 반드시 선언 필요
 - » Block Scope

변수 (Variables)

- `const`

- » ES6(2015)에서 도입
- » `const`로 선언된 변수는 값 변경 불가능 (재할당 불가능) → 상수
- » 재선언 불가능
- » 사용하기 전에 반드시 선언 필요
- » Block Scope

- `var`

- » 구버전에서 지원하는 예약어 (현재도 사용 가능)
- » `var`로 선언된 변수는 값 변경 가능
- » 재선언 가능
- » 코드 순서상 뒤에 선언된 변수 사용 가능 (hoisting)
- » Global 또는 Function Scope

변수 (Variables)

▪ let, const, var 비교

	Scope	Redeclare	Reassign	Hoisted	Binds this
var	No	Yes	Yes	Yes	Yes
let	Yes	No	Yes	No	No
const	Yes	No	No	No	No

자료형 (Data Types)

- 값의 크기와 형식에 대한 정보
- Javascript 자료형 종류
 - » 문자형 (String) → "hello", 'john doe'
 - » 수치형 (정수, 실수) → 10, 12, 34
 - » 진위형 (boolean) → true, false
 - » 객체 → { a: 10, b: 20 }, [1, 2, 3, 4, 5],
new Date('2051-01-01')
- Javascript는 자료형을 표현하는 명시적인 예약어를 제공하지 않으며 저장되는 값에 따라 자동으로 자료형이 결정됨

```
// Numbers:
```

```
let length = 16;
```

```
let weight = 7.5;
```

```
// Strings:
```

```
let color = "Yellow";
```

```
let lastName = "Johnson";
```

```
// Booleans
```

```
let x = true;
```

```
let y = false;
```

```
// Object:
```

```
const person = {firstName:"John", lastName:"Doe"};
```

```
// Array object:
```

```
const cars = ["Saab", "Volvo", "BMW"];
```

```
// Date object:
```

```
const date = new Date("2022-03-25");
```

문자열 자료형 (Strings)

- 0개 이상의 문자로 구성되는 값을 표현하는 자료형
- 큰따옴표 또는 작은따옴표를 사용해서 표현 (하나의 문자열에 혼용할 수 없음)
 - » 'John Doe', "Jane Doe", 'Thomas Doe' (오류)
- 특수문자 표기 (Escape Sequence)
 - » 문자열 내부에 표현하기 어려운 문자를 표현하는 방법

코드	결과	코드	결과
\b	Backspace	\t	Horizontal Tabulator (Tab Key)
\f	Form Feed	\'	Single Quote (')
\n	New Line (Enter Key)	\"	Double Quote (")
\r	Carriage Return (Home Key)	\\	Backslash (\)

문자열 관련 함수

- Javascript는 문자열을 다루기 위한 다양한 함수 제공

함수	설명	함수	설명
length	문자열의 길이 반환	trim()	문자열 양 끝에서 공백 제거
slice()	부분문자열 반환 (start, end)	trimStart()	문자열 앞에서 공백 제거
substring()	부분문자열 반환 (start, end)	trimEnd()	문자열 뒤에서 공백 제거
substr()	부분문자열 반환 (start, length)	padStart()	문자열 앞에 문자 추가 (count, str)
replace()	문자열 대체 (old-str, new-str)	padEnd()	문자열 뒤에 문자 추가 (count, str)
replaceAll()	문자열 대체 (old-str, new-str)	charAt()	문자열내 특정 위치의 문자 반환
toUpperCase()	대문자로 변환	charCodeAt()	문자열내 특정 위치의 문자 코드 반환
toLowerCase()	소문자로 변환	split()	지정된 문자열을 기준으로 문자열 분할 후 배열로 반환
concat()	문자열 병합		

문자열 관련 함수

- Javascript는 문자열을 다루기 위한 다양한 함수 제공

함수	설명	함수	설명
indexOf	문자열의 길이 반환	trim()	문자열 양 끝에서 공백 제거
lastIndexOf	부분문자열 반환 (start, end)	trimStart()	문자열 앞에서 공백 제거
search	부분문자열 반환 (start, end)	trimEnd()	문자열 뒤에서 공백 제거
substr()	부분문자열 반환 (start, length)	padStart()	문자열 앞에 문자 추가 (count, str)
replace()	문자열 대체 (old-str, new-str)	padEnd()	문자열 뒤에 문자 추가 (count, str)
replaceAll()	문자열 대체 (old-str, new-str)	charAt()	문자열내 특정 위치의 문자 반환
toUpperCase()	대문자로 변환	charCodeAt()	문자열내 특정 위치의 문자 코드 반환
toLowerCase()	소문자로 변환	indexOf	문자열에서 부분문자열 위치 검색
concat()	문자열 병합	lastIndexOf	문자열에서 부분문자열 위치 검색
includes()	문자열에 부분문자열 포함 여부	startsWith	문자열이 특정 문자열로 시작하는지 여부
split()	지정된 문자열을 기준으로 문자열 분할 후 배열로 반환	endsWith	문자열이 특정 문자열로 끝나는지 여부

문자열 템플릿

- 유연한 문자열 표현 방법 및 문자열과 데이터를 결합하는 방법 제공

- `backticks(`)` 사용

- 사례

» 큰따옴표, 작은따옴표 등 특수문자를 쉽게 표현 `let text = `He's often called "Johnny"`;`

» 여러 줄로 나누어진 문자열 표현 `let text =
`The quick
brown fox
jumps over
the lazy dog`;`

» 문자열과 데이터 결합

```
let firstName = "John";  
let lastName = "Doe";
```

```
let price = 10;  
let VAT = 0.25;
```

```
let text = `Welcome ${firstName}, ${lastName}!`; let total = `Total: ${((price * (1 + VAT)).toFixed(2))}`;
```


연산자 (Operators)

- 값, 변수 등에 연산을 수행하는 예약된 기호
- 종류
 - » 산술연산자, 할당(대입)연산자, 비교연산자, 논리연산자, 비트연산자, 조건연산자, 자료형연산자

산술연산자

연산자	설명
+	덧셈 연산
-	뺄셈 연산
*	곱셈 연산
**	제곱 연산 (ES2016)
/	나눗셈 연산
%	나머지 연산
++	증가 연산
--	감소 연산

비교연산자

연산자	설명
==	두 값이 같으면 true 아니면 false
===	두 값과 값의 자료형이 같으면 true 아니면 false
!=	두 값이 다르면 true 아니면 false
!==	두 값이 다르거나 자료형이 다르면 true 아니면 false
>	왼쪽 값이 오른쪽 값보다 크면 true 아니면 false
<	왼쪽 값이 오른쪽 값보다 작으면 true 아니면 false
>=	왼쪽 값이 오른쪽 값보다 크거나 같으면 true 아니면 false
<=	왼쪽 값이 오른쪽 값보다 작거나 같으면 true 아니면 false

연산자 (Operators)

할당(대입)연산자

연산자	사례	비교
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
**=	x **= y	x = x ** y

논리연산자

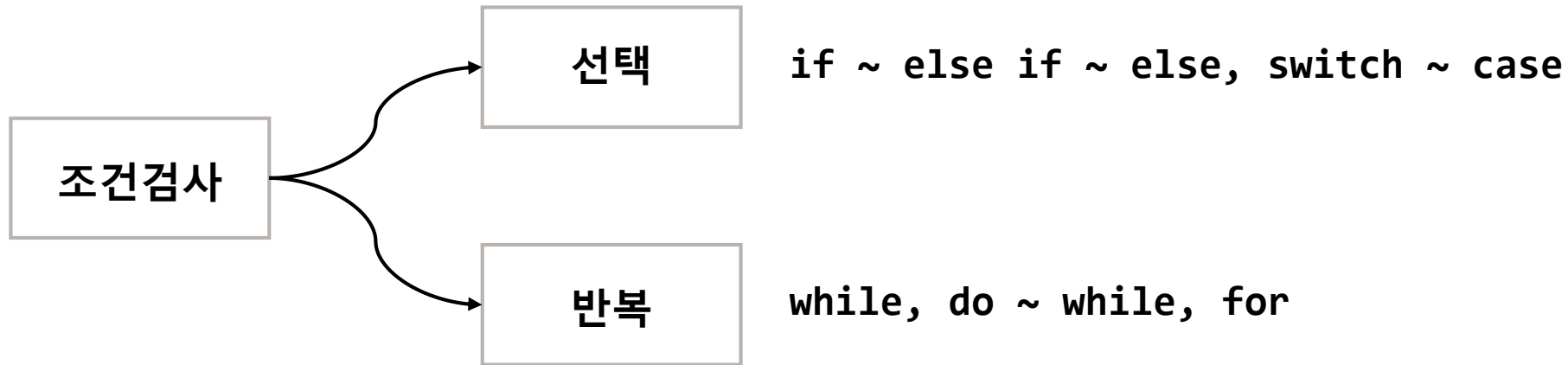
연산자	사례
&&	논리곱 연산
	논리 합 연산
!	논리 부정 연산

자료형연산자

연산자	사례
typeof	값 또는 변수의 자료형 반환
instanceof	객체의 자료형이 특정 자료형과 일치하면 true 아니면 false

제어문 (Control Statements)

- 실행 흐름의 논리적인 제어를 수행하는 구문
- 동작 구조



제어문 (Control Statements)

- if

```
if (condition) {  
    // block of code to be executed if the condition is true  
}
```

- if ~ else

```
if (condition) {  
    // block of code to be executed if the condition is true  
} else {  
    // block of code to be executed if the condition is false  
}
```

- if ~ else if ~ else

```
if (condition1) {  
    // block of code to be executed if condition1 is true  
} else if (condition2) {  
    // block of code to be executed if the condition1 is false and condition2 is true  
} else {  
    // block of code to be executed if the condition1 is false and condition2 is false  
}
```

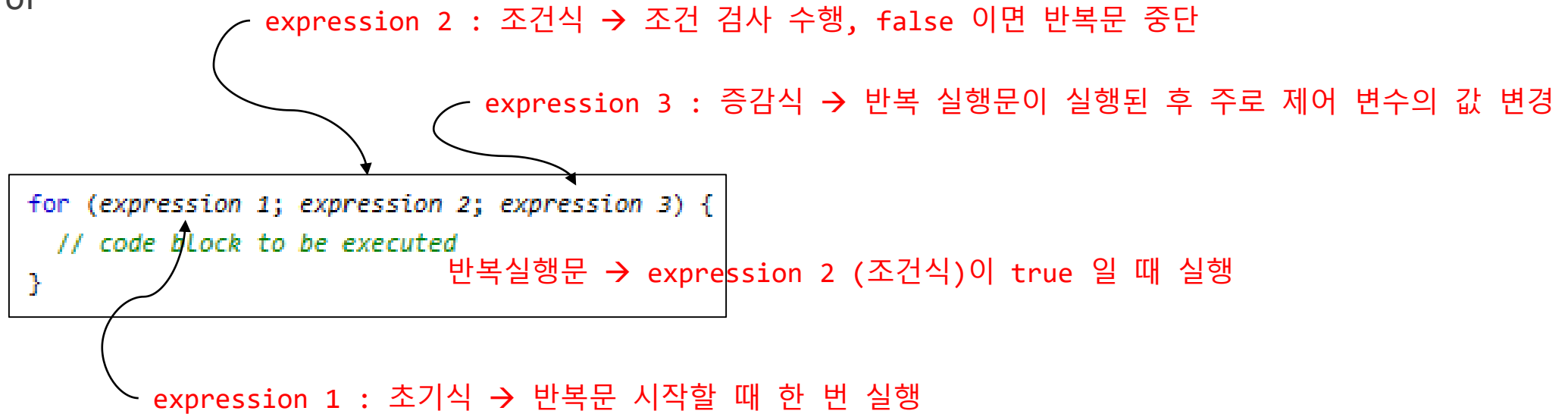
제어문 (Control Statements)

- switch ~ case

```
switch(expression) {  
    case x:  
        // code block    expression == x 일 때 실행되는 영역  
        break;    switch ~ case 문 중단  
    case y:  
        // code block    expression == y 일 때 실행되는 영역  
        break;  
    default:  
        // code block    expression != x and expression != y 일 때 실행되는 영역  
}
```

제어문 (Control Statements)

■ for



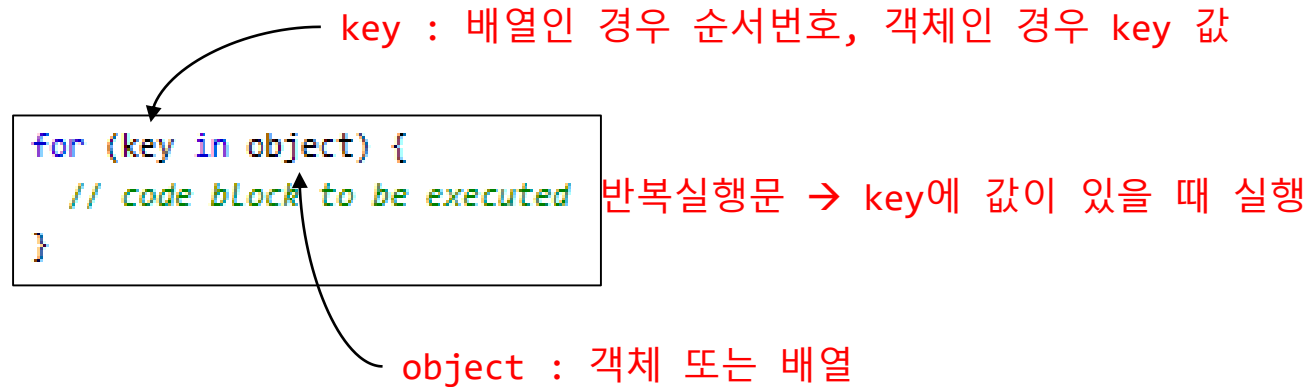
■ 실행순서

» expression 1 → expression 2 → 반복실행문 → expression 3
expression 2 → 반복실행문 → expression 3
expression 2 → 반복실행문 → expression 3
...
expression 2 → 반복실행문 → 종료

제어문 (Control Statements)

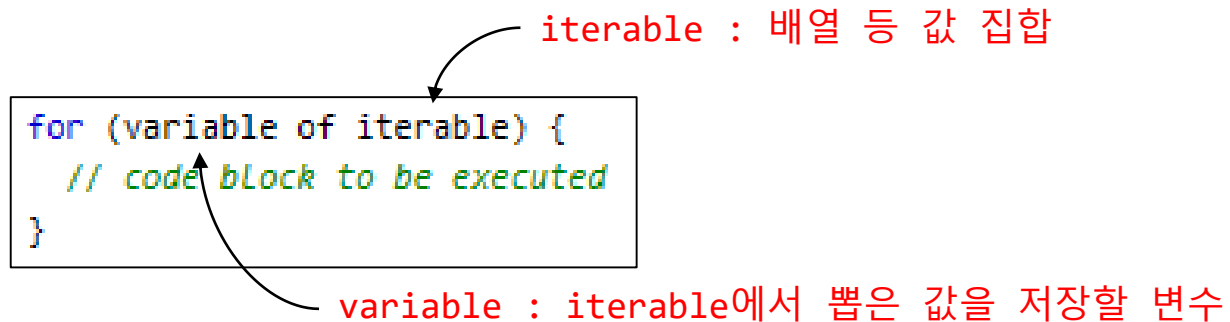
■ for in

» 객체 또는 배열의 값을 순차적으로 뽑아서 key에 저장하고 반복 실행문 실행



■ for of

» 반복 가능한 객체에서 순차적으로 값을 뽑아서 variable에 저장하고 반복 실행문 실행



제어문 (Control Statements)

- while

```
while (condition) {  
    // code block to be executed  
}
```

- do ~ while

```
do {  
    // code block to be executed  
}  
while (condition);
```

- break

- » 반복문 또는 switch ~ case 문을 종료하는 명령

- continue

- » 반복문의 처음으로 실행 위치를 이동하는 명령

```
for (let i = 0; i < 10; i++) {  
    if (i === 3) { break; }  
    text += "The number is " + i + "<br>";  
}
```

```
for (let i = 0; i < 10; i++) {  
    if (i === 3) { continue; }  
    text += "The number is " + i + "<br>";  
}
```


함수 (Function)

- 어떤 기능을 수행하는 코드 집합
- 재사용성 개선과 코드를 체계적으로 관리하는 데 유용한 도구
- 함수 정의 (함수 만들기)

```
function name(parameter1, parameter2, parameter3) {  
    // code to be executed  
}
```

함수를 만들어도 실행되지 않음

- 함수 호출
 - » 정의된 함수는 호출을 통해 실행됨

```
function_name(parameter_value1, parameter_value2, ...)
```

함수 (Function)

- 지역 변수 (Local Variables)
 - » 함수 영역에서 선언된 변수는 함수 외부에서 접근할 수 없음

```
<script>

function myFunction() {
  let carName = "Volvo";
  let text = "Inside: " + typeof carName + " " + carName;
  console.log(text)
}

let text2 = "Inside: " + typeof carName + " " + carName;
console.log(text2)

myFunction();

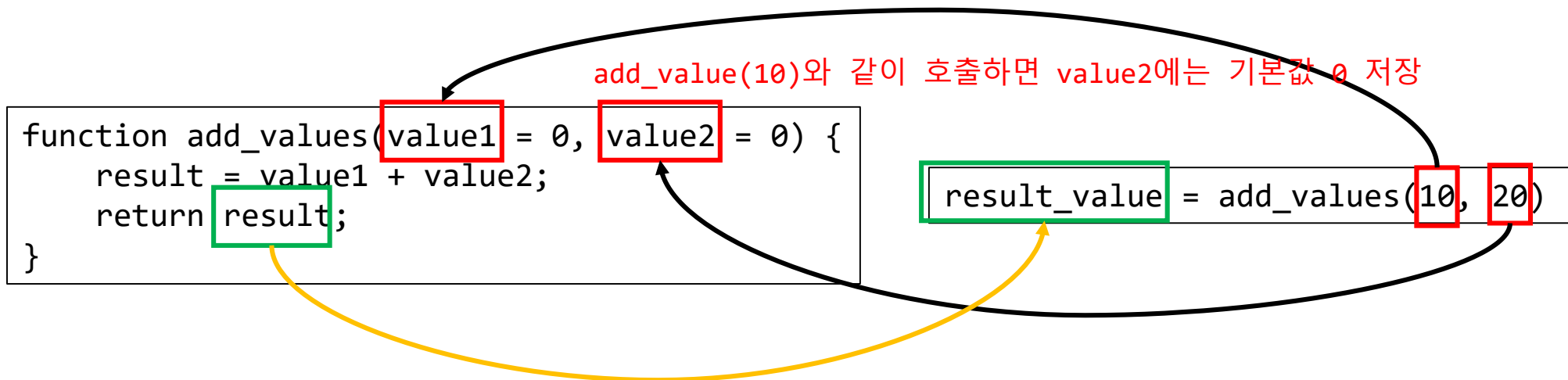
</script>
```

함수 내부에서 만들어진 carName 사용 가능

오류 : 함수 내부에서 만들어진 carName을
함수 외부에서 사용할 수 없음

함수 (Function)

- 전달인자 (parameters)
 - » 함수를 실행하는 데 필요한 데이터
 - » 함수 외부의 데이터를 함수가 사용할 수 있도록 주입하는 통로
 - » 전달인자는 필요에 따라 0개 이상 지정 가능
 - » 전달인자에 기본 값을 지정할 수 있음 (호출할 때 전달되지 않으면 기본 값 사용)
- 값 반환 (return)
 - » 함수 내부에서 생산된 값을 호출한 곳으로 돌려주는 기능
 - » return 구문 사용



함수 (Function)

- 함수를 저장하는 변수

```
function add_values(value1 = 0, value2 = 0) {  
    result = value1 + value2;  
    return result;  
}  
let add_values2 = add_values;  
add_values2(10, 20); // add_values(10, 20)과 동일한 호출
```

- 익명 함수

```
const add_values = function (value1 = 0, value2 = 0) {  
    result = value1 + value2;  
    return result;  
}  
add_values(10, 20);
```

익명함수, 화살표함수는 재사용하지 않고
한 번만 사용할 함수 형식이 필요한 경우
사용

- 화살표 함수

```
const add_values = (value1 = 0, value2 = 0) => value1 + value2;  
add_values(10, 20);
```

이벤트 (Event)

- 프로그램 실행 중에 발생한 중요한 사건 → 주로 HTML 요소에 발생한 사건
- 브라우저에 의해 발생하거나 사용자 액션에 의해 발생
- 이벤트 사례
 - » 웹 페이지가 브라우저에 로드되었을 때
 - » 입력 요소의 내용이 변경되었을 때
 - » 버튼 요소가 클릭되었을 때
- 주요한 사건에 대해 어떤 처리를 수행하기 위해 javascript 코드를 연결

```
<element event='javascript code 또는 function'>함수를 호출하는 경우 중 하나
```

- 이벤트 처리 사례

```
<button onclick="console.log(Date())">The time is?</button>
```

이벤트 (Event)

■ 주요 HTML 이벤트

	이벤트	설명
마우스	onclick	사용자가 HTML 요소를 클릭했을 때 발생
	onmouseover	사용자가 HTML 요소 위에서 마우스를 움직일 때 발생
	onmouseout	마우스 포인터가 HTML 요소 위에서 밖으로 벗어날 때 발생
	onmousedown	사용자가 마우스 버튼을 클릭했을 때 발생
	onmouseup	사용자가 마우스 버튼을 클릭했다가 떼는 순간 발생
포커스	onfocus	HTML 요소가 키보드 입력 포커스를 받을 때 발생
	onblur	HTML 요소가 키보드 입력 포커스를 잃을 때 발생
키보드	onkeydown	사용자가 키보드의 키를 누를 때 발생
	onkeyup	사용자가 키보드의 키를 눌렀다가 떼는 순간 발생
폼	onsubmit	입력 form이 서버로 전송될 때 발생
브라우저	onload	브라우저가 웹페이지의 로딩을 끝낸 경우에 발생

배열 (Arrays)

- 여러 개의 값을 저장하는 변수 (객체)

- 배열 선언

- » [] 또는 Array 클래스 사용

```
const cars = ["Saab", "Volvo", "BMW"];    const cars = new Array("Saab", "Volvo", "BMW");
```

- 배열 요소 접근

- » 첨자 ([순서번호]) 사용

```
const cars = ["Saab", "Volvo", "BMW"];    const cars = ["Saab", "Volvo", "BMW"];  
let car = cars[0];                        cars[0] = "Opel";
```

배열 (Arrays)

- 배열 관련 함수

함수	설명
length	배열 요소의 갯수 반환
toString()	배열 요소를 ,로 연결된 문자열로 변환
pop()	배열의 마지막 요소 제거
push()	배열의 끝에 새 요소 추가
shift()	배열의 첫 번째 요소 제거
unshift()	배열의 앞부분에 새 요소 추가
join()	배열 요소를 지정된 문자로 연결한 문자열로 변환
concat()	배열에 다른 배열 결합
flat()	고차원 배열을 1차원 배열로 변환
splice()	배열의 기존 요소를 다른 요소로 교체
slice()	배열의 일부 요소를 뽑아서 다른 배열 생성
sort()	배열의 요소를 오름차순으로 정렬
reverse()	배열의 요소를 역순으로 재구성

객체

- 정의
 - » 프로그램으로 제어하는 모든 대상
- 객체에는 여러 개의 속성과 메서드로 구성됨
 - » 속성 → 객체의 특성을 표현 → 변수로 구현
 - » 메서드 → 객체의 기능(동작)을 표현 → 함수로 구현

[속성]

```
name : "Jonn Doe"
email : "johndoe@example.com"
gender : "male"
```

[메소드]

```
toString : function() { ... }
getEmail : function() { ... }
```



```
let person = {
  name      : "John Doe",
  email     : "johndoe@example.com",
  gender    : "male",
  toString  : function() { ... },
  getEmail  : function() { ... }
};
```

JSON (Javascript Object Notation)

- Javascript의 객체 표현 방식 → 구조화된 데이터를 표현하기 위한 문자 기반 표준 포맷
- 웹 애플리케이션에서 데이터를 전송할 때 일반적으로 사용
- 모든 웹 브라우저에서 호환 가능한 데이터 형식으로 이기종 시스템간 데이터 교환에 광범위하게 사용

JSON (Javascript Object Notation)

- 구문 형식
 - » Key : value 형식으로 데이터 표현

자료형	표현 방법	예
number	Integer 또는 float	"number" : 1
string	큰 따옴표로 묶음	"name" : "장동건"
boolean	true 또는 false	"isResult" : true
Object	여러 개의 key/value를 입력하여 중괄호로 묶음	{"name" : "장동건", "gender" : "남자"}

Array 여러 개의 object를 대괄호로 묶음

```
{
  "employees" : [
    {"name" : "장동건", "gender" : "남자"},
    {"name" : "싸이", "gender" : "남자"},
    {"name" : "김태희", "gender" : "여자"}
  ]
}
```

JSON (Javascript Object Notation)

■ 사용 사례

```
<script>
var person = {
    "name" : "장동건",
    "address" : "서울시 강남구 역삼동 1",
    "age" : 18,
    "phone" : {
        "mobile" : "010-1234-5678",
        "office" : "02-9876-5432"
    }
};

console.log(person.name)
console.log(person.address)
console.log(person.phone.mobile)
console.log(person.phone.office)
console.log(person.age)
</script>
```

JSON (Javascript Object Notation)

- 문자열과 JSON 객체간 변환

- » JSON.parse() : 문자열 → JSON 객체 변환

```
<script>
let text = '{"employees":[" +
'{"firstName":"John","lastName":"Doe" },' +
'{"firstName":"Anna","lastName":"Smith" },' +
'{"firstName":"Peter","lastName":"Jones" }]]}';

const obj = JSON.parse(text);
console.log(obj.employees[0].firstName + obj.employees[0].lastName)
</script>
```

- » JSON.stringify() : JSON 객체 → 문자열 변환

```
<script>
data = { x: 5, y: 6 }
string_data = JSON.stringify(data);
console.log(string_data); // Expected output: '{"x":5,"y":6}'
</script>
```

Javascript 객체 유형

- 표준 내장 객체
 - » 빈번하게 사용되는 유용한 코드를 미리 구현해서 제공하는 객체
 - » 자바스크립트가 실행되는 모든 곳에서 사용 가능
 - » Number, String, Date, Array, Math 등
- HTML DOM 객체
 - » HTML 문서의 각 HTML 태그들에 대한 객체화 결과
 - » HTML 문서의 내용과 모양을 제어하는 도구
 - » W3C 표준 객체
- 브라우저 객체
 - » 브라우저를 제어하기 위해 제공되는 객체
 - » BOM(Browser Object Model)을 따르는 **비표준** 객체

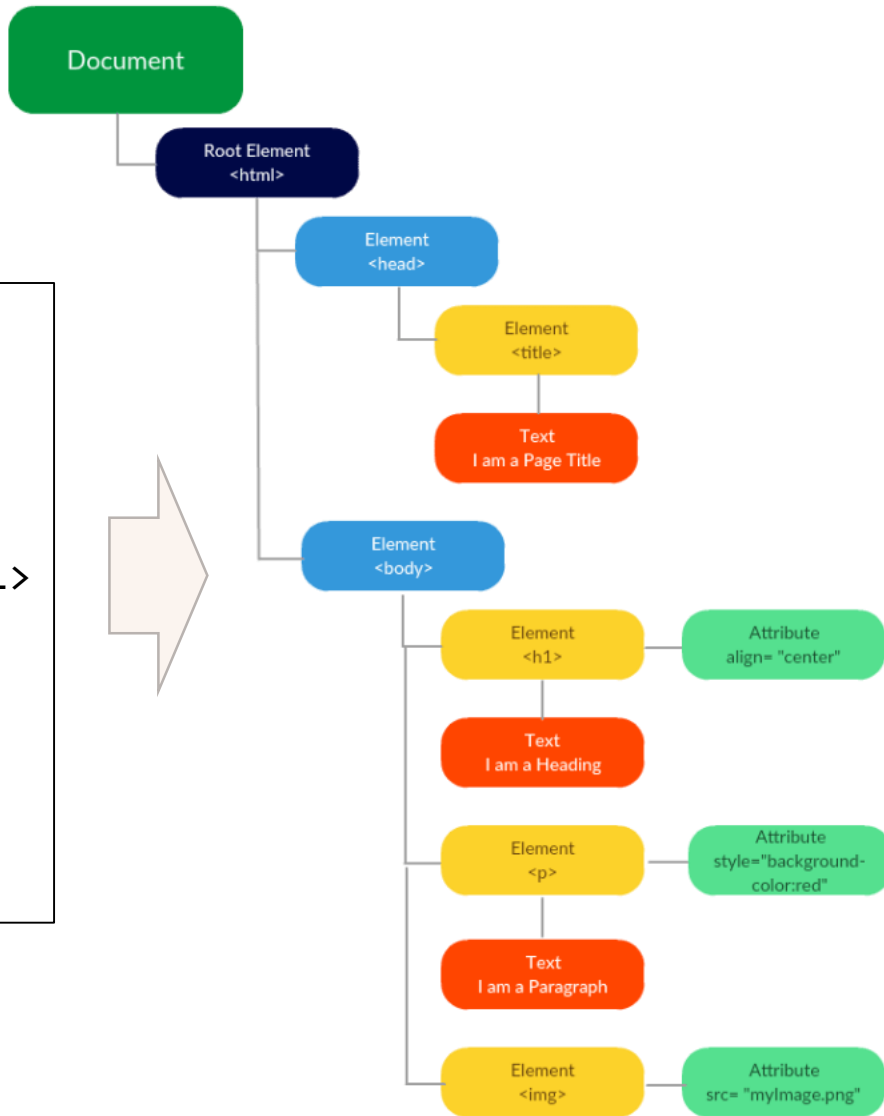
표준 내장 객체

- 주요 표준 내장 객체

객체	설명
Number	숫자를 표현하고 다룰 때 사용하는 Primitive Wrapper Object 2^53 - 1 범위의 데이터
BigInt	Number 원시 값이 안정적으로 나타낼 수 있는 2^53 - 1보다 큰 정수 표현
Math	수학적인 상수와 함수를 위한 속성과 메서드를 가진 내장 객체 Number 자료형만 지원 (BigInt와는 사용할 수 없음)
Date	시간의 한 점을 플랫폼에 종속되지 않는 형태로 표현 1970년 1월 1일 UTC 자정과의 시각 차이를 1/1000초로 나타내는 정수 값 저장
String	문자열 관련 속성과 메서드를 포함하는 내장 객체
RegExp	정규표현식 패턴을 사용해서 텍스트를 판별할 때 사용
Array	리스트 형태의 고수준 객체인 배열 관련 속성과 메서드 포함

HTML DOM 객체 변환

```
<html>
  <head>
    <title>I am Page Title</title>
  </head>
  <body>
    <h1 align="center">I am a heading</h1>
    <p style="background-color: red">
      I am a paragraph
    </p>
    
  </body>
</html>
```



HTML DOM Document 객체

- HTML 문서 전체를 참조하는 객체
 - » DOM 객체를 접근하는 경로의 시작점
 - » DOM 트리의 최상위 객체 → 브라우저는 Document 객체 생성 후 document 객체 하우
- 주요 Document 속성 및 객체

속성 / 메서드	설명
document.getElementById()	id 속성으로 HTML 요소 탐색
document.getElementsByTagName()	태그 이름으로 HTML 요소 탐색
document.getElementsByClassName()	class 속성으로 HTML 요소 탐색
document.querySelector()	선택자 구문으로 HTML 요소 탐색 (발견된 첫 요소 반환)
document.querySelectorAll()	선택자 구문으로 HTML 요소 탐색 (발견된 모든 요소 반환)

HTML DOM Document 객체

- 주요 Document 속성 및 객체 (계속)

속성 / 메서드	설명
<code>document.createElement()</code>	HTML 요소 생성
<code>document.removeChild()</code>	DOM 트리에서 HTML 요소 제거
<code>document.appendChild()</code>	DOM 트리에 HTML 요소 추가
<code>document.replaceChild()</code>	DOM 트리의 기존 HTML 요소를 다른 HTML 요소로 변경
<code>document.write()</code>	문서(화면)에 내용 출력

HTML DOM Document 객체

- 주요 DOM 객체 (HTML 요소) 제어 속성 및 메서드

속성 / 메서드	설명
<code>document.setAttribute()</code>	HTML 요소의 속성 값 변경
<code>element.innerHTML</code>	HTML 요소의 내용 변경
<code>element.attribute</code>	HTML 요소의 속성 값 변경
<code>element.style.property</code>	HTML 요소의 <code>style</code> 속성에 스타일 값 지정
<code>element.event_name</code>	HTML 요소의 <code>event</code> 에 이벤트 처리기 연결

HTML DOM Document 객체

- 주요 DOM 객체 (HTML 요소) 제어 속성 및 메서드

속성 / 메서드	설명
document.body	<body> 요소 객체
document.head	<head> 요소 객체
document.lastModified	HTML 문서의 최종 수정 시간
document.title	<title> 요소의 내용
document.URL	HTML 문서의 전체 URL

Browser Object Model

- BOM 객체
 - » 자바스크립트로 브라우저를 제어하기 위해 지원되는 객체

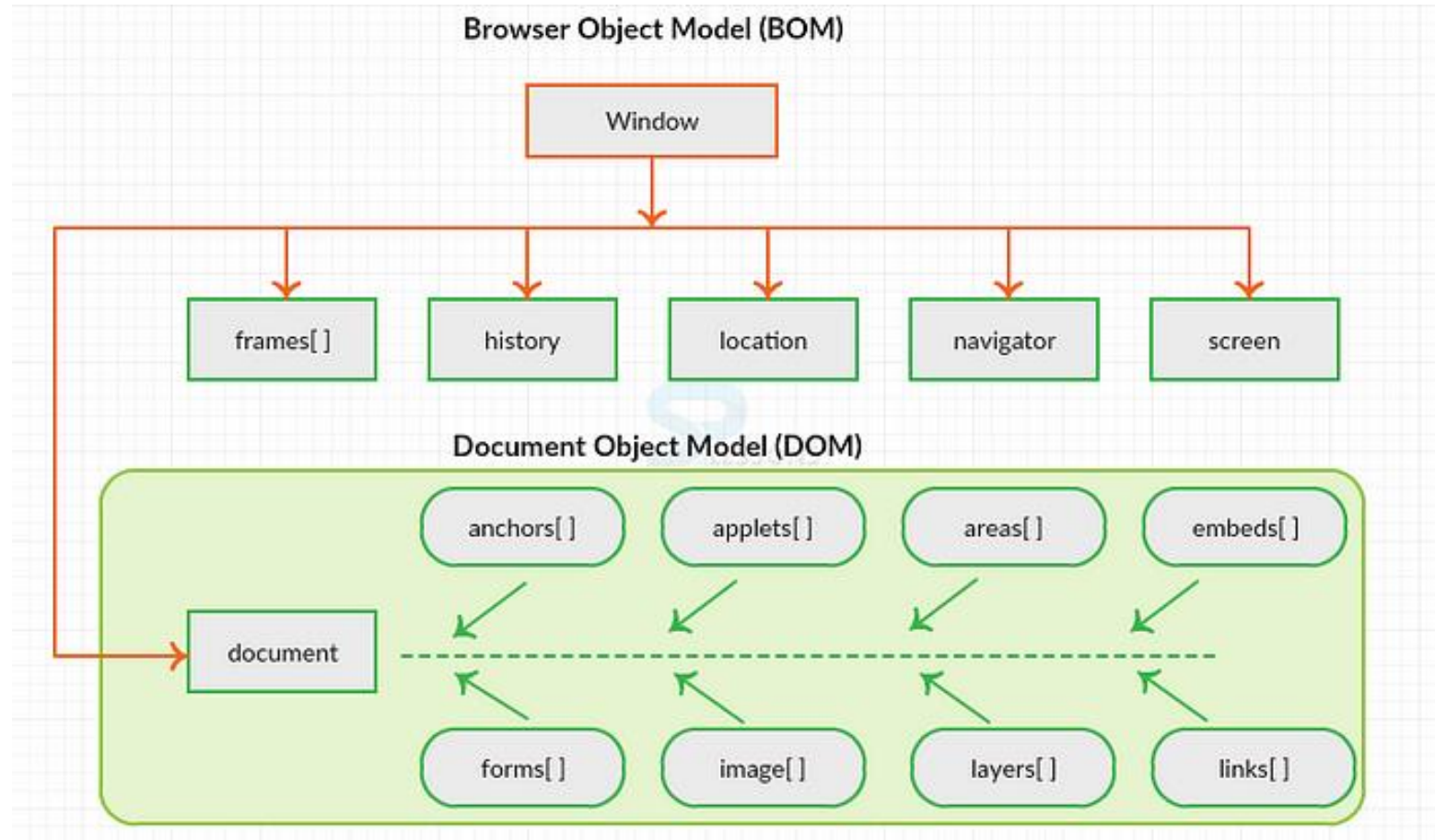
- 주요 BOM 객체

객체	설명
Window	브라우저 윈도우 제어
Navigator	브라우저에 대한 다양한 정보 제공
History	브라우저 윈도우로 접속한 URL 리스트의 히스토리 관리
Location	브라우저 윈도우로 접속한 현재 페이지의 URL 관리 (주소 입력창 기능)
Screen	브라우저가 실행되고 있는 스크린 장치에 대한 정보

- 비표준
 - » 브라우저간 BOM 객체 호환성이 보장되지 않음

Browser Object Model

- BOM 객체와 DOM 객체



Browser Object Model

▪ Window 객체 주요 속성 및 메서드

속성/메서드	설명
alert()	메시지 팝업 표시 (확인 버튼)
confirm()	메시지 팝업 표시 (확인 또는 취소 버튼)
open()	새 브라우저 윈도우 시작
close()	지정된 브라우저 윈도우 종료
setTimeout()	지정된 시간 후에 지정된 작업 1회 실행
clearTimeout	setTimeout()으로 예약된 작업 취소
setInterval()	지정된 시간마다 지정된 작업 반복 실행
clearInterval()	setInterval()로 예약된 작업 취소
scrollBy	지정된 양만큼 스크롤 업/다운
scrollTo	지정된 위치로 스크롤

Browser Object Model

- History 객체의 주요 속성 및 메서드

속성/메서드	설명
history.forward()	이전에 방문한 URL 목록에서 다음 URL로 이동
history.back()	이전에 방문한 URL 목록에서 이전 URL로 이동
history.go()	이전에 방문한 URL 목록에서 지정된 양만큼 이동

- Location 객체의 주요 속성 및 메서드

속성/메서드	설명
location.href	현재 페이지의 URL
location.hostname	현재 페이지의 도메인 이름
location.pathname	현재 페이지의 경로와 파일 이름 (도메인 이름 이후의 경로)
location.protocol	현재 페이지의 웹 프로토콜

Browser Object Model

▪ Navigator 객체의 주요 속성 및 메서드

속성/메서드	설명
navigator.userAgent	웹서버로 전송할 때 HTTP 헤더의 user-agent 필드에 저장하는 값 웹서버가 클라이언트를 인식하기 위한 목적
navigator.platform	운영체제 플랫폼 이름

▪ Screen 객체의 주요 속성 및 메서드

속성/메서드	설명
screen.width	스크린의 수평 픽셀 수
screen.height	스크린의 수직 픽셀 수
screen.availWidth	브라우저가 출력 가능한 영역의 너비
screen.availHeight	브라우저가 출력 가능한 영역의 높이
screen.pixelDepth	한 픽셀의 색을 나타내기 위해 사용되는 비트 수
screen.colorDepth	한 픽셀의 색을 나타내기 위해 사용되는 비트 수