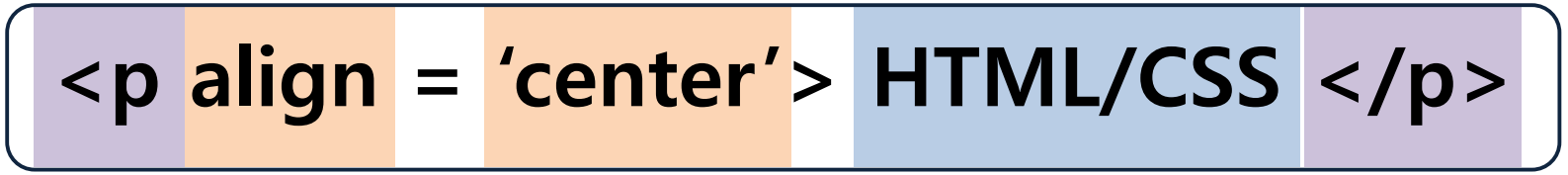
HTML 개요

▶ HTML5

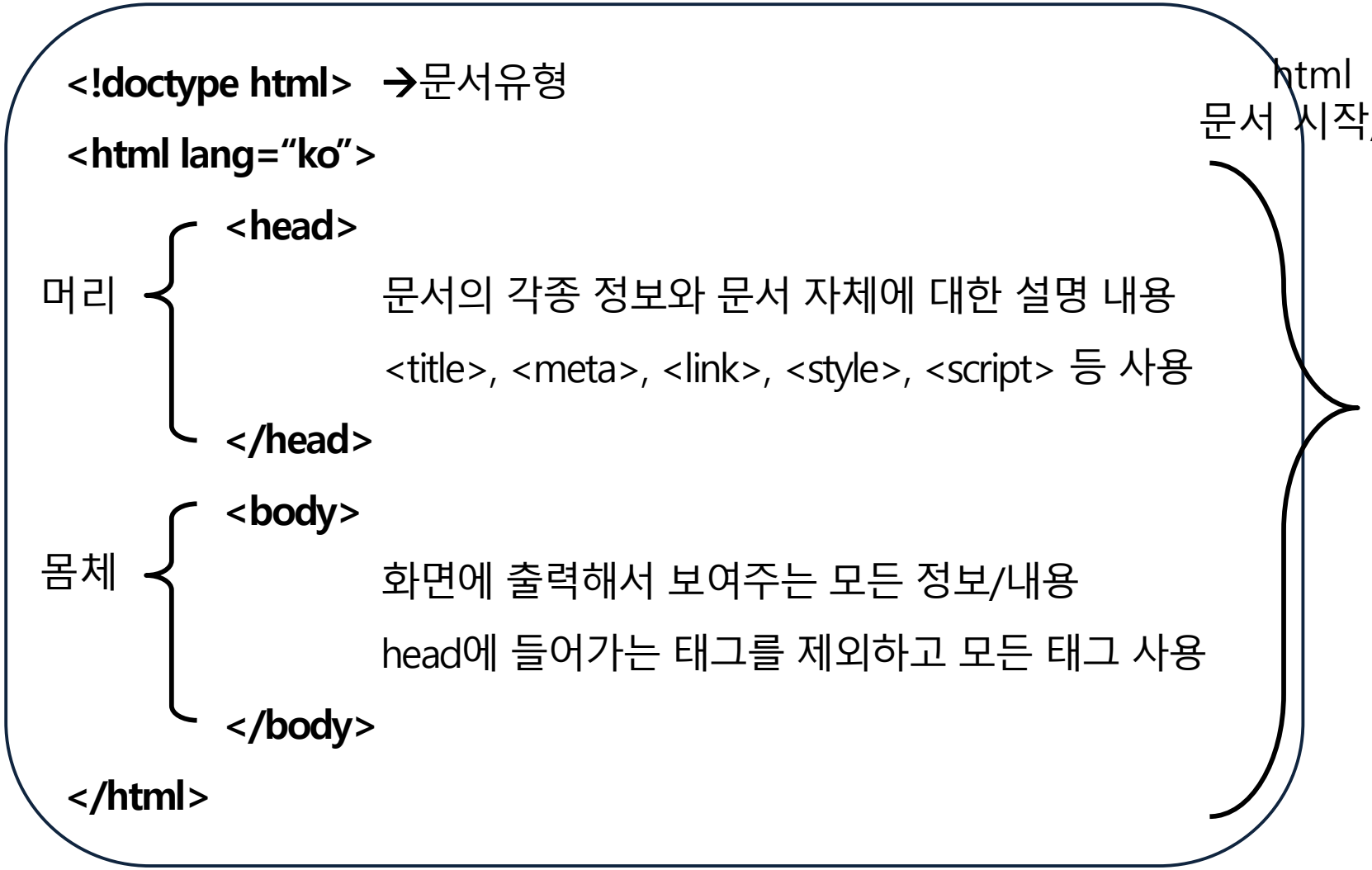
✓ 구성 요소



구성요소	내 용
태그(Tag)	'< '와 '>'로 묶인 명령어 시작태그(<태그>)와 종료태그(</태그>)를 한쌍으로 이용
요소(Element)	시작태그와 종료태그로 이루어진 모든 명령어 하나의 HTML문서는 요소들의 집합
속성 (Attribute)	요소의 시작태그에만 사용 / 명령어 구체화 역할 여러 개의 속성을 사용할 수 있으며 속성의 구분은 공백
변수/속성값 (Argument)	속성이 가지는 값, 값 입력 시 " " 혹은 ' '를 이용

HTML5

✓ 기본 구조


<!doctype html> →문서유형
<html lang="ko">
<head>
머리
문서의 각종 정보와 문서 자체에 대한 설명 내용
<title>, <meta>, <link>, <style>, <script> 등 사용
</head>
<body>
몸체
화면에 출력해서 보여주는 모든 정보/내용
head에 들어가는 태그를 제외하고 모든 태그 사용
</body>
</html>
html
문서 시작/끝

글자 관련 태그

▶ <hn></hn>

제목을 입력할 때 사용하는 태그로 폰트의 크기가 태그마다 정해져 있음

h 뒤 숫자(n)으로 구분

<hn>제목 내용</hn>

h1	첫 번째로 큰 제목
h2	두 번째로 큰 제목
h3	세 번째로 큰 제목
h4	네 번째로 큰 제목
h5	다섯 번째로 큰 제목
h6	여섯 번째로 큰 제목

▶ 줄 바꿈과 구분 줄

✓

문장을 줄 바꾸기(개행)할 때 사용

내용

✓ <hr>

페이지에 가로로 밑줄을 만들어 줄 때 사용

내용

<hr>

목록 관련 태그

▶

순서가 필요하지 않은 목록 만들 때 사용

리스트 앞에 특정 모양 출력(기본 값 : ·), 모양은 CSS 이용해 수정 가능

```
<ul>
        <li>내용1</li>
        <li>내용2</li>
        <li>내용3</li>
</ul>
```

▶

순서가 있는 목록 만들 때 사용
속성으로 문자(A, B, C…/a, b, c…), 숫자(1, 2, 3…),
로마자(Ⅰ, Ⅱ, Ⅲ…/ⅰ, ⅱ, ⅲ…) 설정
CSS로 순서 설정 가능

```
<ol type="설명문자" start="시작순서">
        <li>내용1</li>
        <li>내용2</li>
        <li>내용3</li>
</ol>
```

✓ 속성

type	순서형식을 정하는 것 (a,A,1,i,I)
Start	시작순서

표 관련 태그

▶ <table></table>

웹 문서에서 자료를 정리할 때 주로 사용
행과 열로 이루어져 있고 행과 열이 만나는 지점을 셀이라고 함

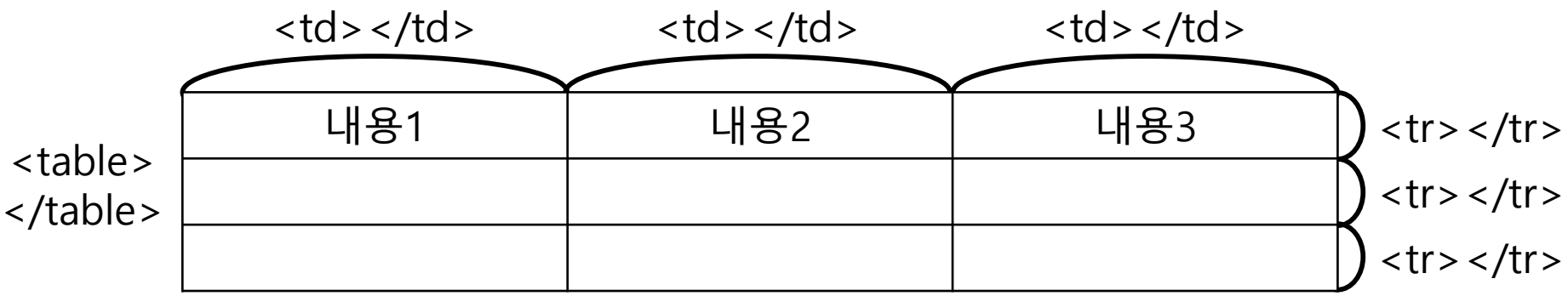
```
<table>
        <tr>
                <td>내용</td>
                <td>내용</td>
        </tr>
</table>
```

* 1행 2열 테이블 작성

✓ 태그

<tr></tr>	한 개의 행(ROW)을 만드는 태그
<td></td>	한 개의 열을 만드는 태그

▶ 테이블 구조




```
<table>
        <tr>
                <td>내용1</td>
                <td>내용2</td>
                <td>내용3</td>
        </tr>
                .
                .          X3
                .
</table>
```

영역 관련 태그

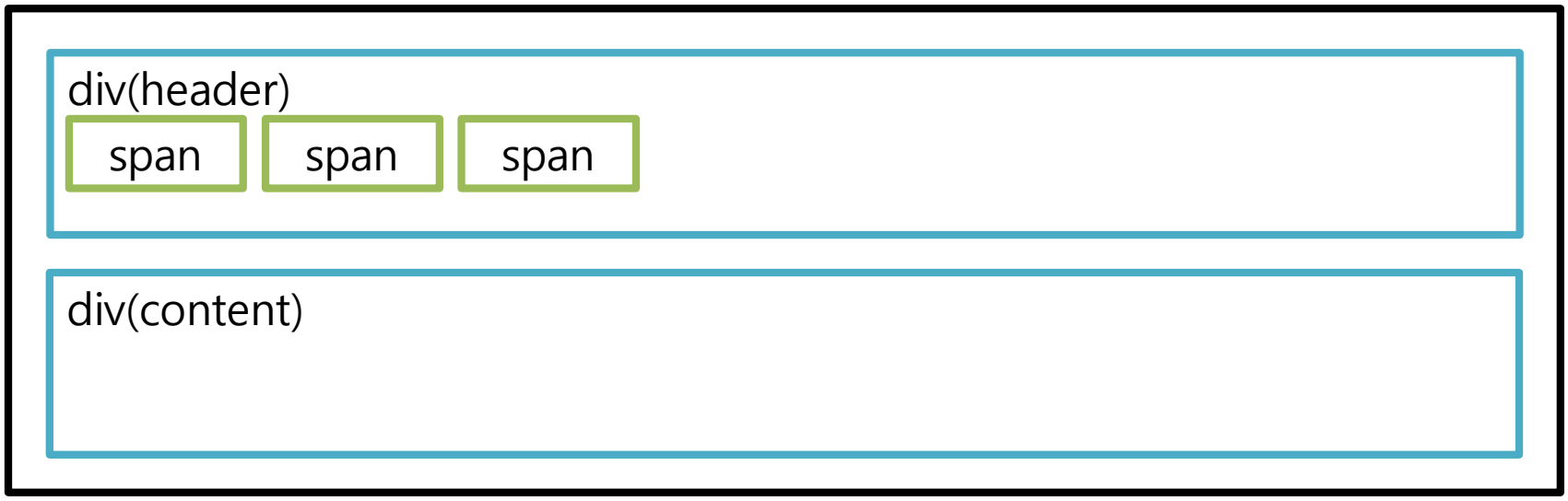
페이지 영역 분할

✓ ** **

block형식의 공간을 수직으로 분할

✓ ****

inline형식의 공간을 수평으로 분할



멀티미디어 관련 태그

▶ 이미지 태그

✓ <img/>

웹 페이지에 사진이나 그림을 삽입할 때 사용하는 태그

✓ 속성

src	삽입할 이미지 경로를 지정하는 속성
alt	이미지 설명해주는 텍스트 속성 이미지가 출력이 안되면 표시됨
width, height	이미지의 크기를 설정하는 속성

▶ 이미지 태그

✓ 웹 페이지 사용 가능한 확장자

형식	내용
GIF	파일 크기가 작아 아이콘이나 블릿 기호에 많이 사용 투명한 배경이나 움직이는 이미지를 만들 수 있음
JPG/JPEG	사진을 위해 개발, 저장을 반복하다 보면 화질이 떨어질 수 있음.
PNG	네트워크용으로 개발되어 요즘 많이 이용

하이퍼링크 태그

▶ 하이퍼 텍스트

✓ <a>

페이지에서 해당 부분을 클릭하면 지정된 페이지로 이동하는 태그로 외부 사이트 연결, 문서 내부에서 이동 가능

✓ 속성

href	링크한 페이지의 id값이나 사이트 주소 지정
target	링크 페이지가 표시될 위치(새 창, 현재 창) 지정
download	링크한 페이지를 표시하지 않고 다운로드 하는 것
rel	현재 페이지와의 관계 지정
hreflang	링크한 페이지의 언어 지정
type	페이지의 파일 유형 지정

▶ 기본 작성

✓ 문자

```
<a href="주소">
        링크표시 문구
</a>
```

✓ 이미지

```
<a href="주소">
        <img src="이미지 경로"/>
</a>
```

▶ 내부 이동 및 target이용

✓ 내부에서 이동하기

```
<p id="p1"></p>

<a href="#p1">
        링크 표시 문구
</a>
```

✓ target 이용하기

```
<a href="주소" target="_blank(새 창)" or "_self(현재)"
                    or "_parent(상위 창)" or "_top(최상위창)">
</a>
```

폼 관련 태그

▶ <form></form>

사용자가 입력한 data를 보내는 방식과 처리할 프로그램을 정하는 태그

✓ 속성

method	get : URL창에 데이터를 보내는 방식 data크기에 제한이 있음(256~4096byte) data를 볼 수 있음 post : http request에 data를 넣어 보내는 방식 data크기에 제한이 없음 data를 볼 수 없음
name	<form>태그의 고유 이름 지정, <form>을 구분하기 위해 사용
action	데이터를 처리 할 프로그램(페이지)을 지정
target	action속성의 프로그램(페이지)를 어떻게 열지 지정
autocomplete	이전 입력내용 출력하는 기능 생략하면 자동으로 출력(default : on)

▶ <fieldset></fieldset>, <legend></legend>

폼 요소를 그룹으로 묶고 묶은 폼 요소에 명칭을 붙이는 태그

```
<form action="프로그램(페이지)" method="방식">
        <fieldset name="이름">
                <legend>명칭<legend>
                <input type...>
                ...
        </fieldset>
</form>
```

✓ 속성

name	fieldset의 이름 지정
form	fieldset이 속한 form의 이름 지정

▶ <input>

사용자로부터 data를 입력 받기 위한 태그

✓ 속성

type	입력 창의 타입을 결정하는 속성 (text, checkbox, radio 등)
value	input요소의 기본 값 표시
name	input을 구별할 수 있는 명칭
min/max/step	허용하는 범위 최소값/최대값/값의 증감값
autocomplete	자동 완성 기능
height/width	입력 창의 높이와 너비
readonly	읽기 전용 필드
accept	파일 타입에 대해 사용자에게 알려주는 기능
multiple	여러 개의 값 입력 가능
placeholder	사용자 입력 전 입력 창 표시 글

▶ <input>

✓ 속성

autofocus	입력 창 커서 표시
required	필수 입력 필드 지정
list	<datalist>의 옵션 값을 <input>안에 나열
maxlength	사용자가 입력할 수 있는 글자수 제한
size	화면에서 표시하는 글자 수
minlength	최소 입력 글자(최신 크롬, 안드로이드만 지원)
formaction	실행할 프로그램 연결, submit/image일 때 사용
formenctype	전송 시 어떤 방식으로 해석할 지 지정 submit/image일 때 사용
formnovalidate	전송 시 데이터가 유효한지 여부 표시
formtarget	서버의 응답을 어디에 표시할 것인지 지정

▶ <input>

✓ type 기본 속성

text	한 줄짜리 텍스트 입력 창이 생김
password	비밀 번호 입력 창, 입력 시 ●●●으로 표시
hidden	사용자에게 보이지는 않지만 값을 넣을 수 있는 창 관리자에게 필요한 값을 넣을 때 사용
button	버튼 생성, 자체 별도 기능은 없고 스크립트에서 함수 연결하여 활용
checkbox	체크 박스 생성
file	파일 입력 양식 출력
image	이미지 형태 생성
radio	라디오 버튼 생성
submit	입력한 데이터를 다른 페이지로 넘기는 기능
reset	입력한 내용을 지우는 기능

▶ <input>

✓ hidden

```
<input type=hidden name="명칭" value="값">
```

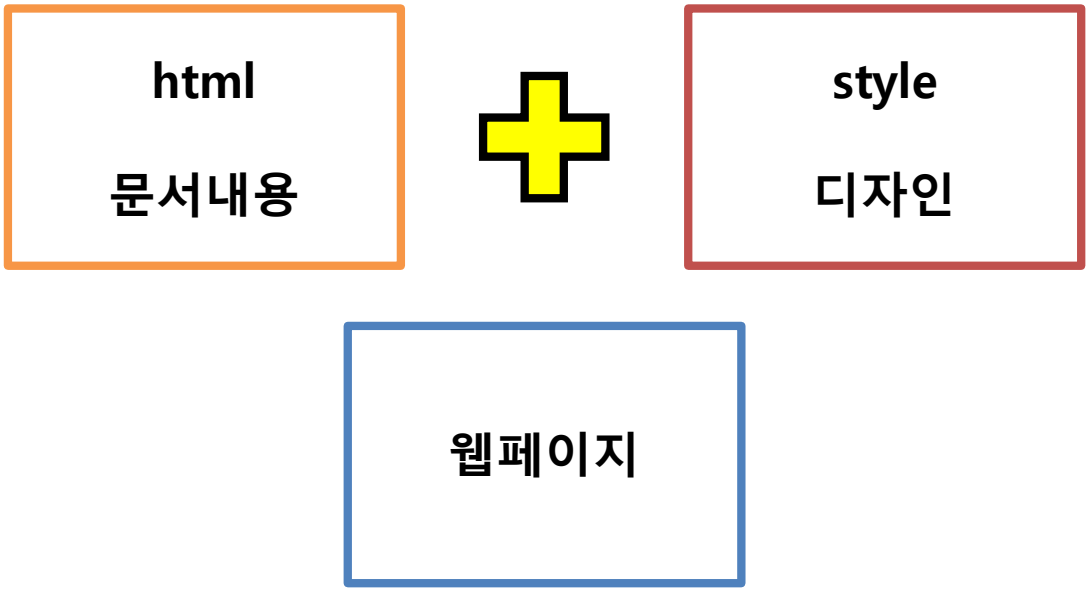
✓ hidden 속성

name	hidden type 구분자, 명칭
value	hidden에 들어갈 값 설정

CSS

▶ 웹 페이지 구성

문서 내용 작성과 꾸미는 부분을 분리하여 내용을 수정해도 디자인을 바꿀 필요가 없고 디자인을 수정해도 글 내용을 바꿀 필요가 없음
다양한 기기에도 디자인만 따로 적용하여 구동시킬 수 있음



▶ style과 stylesheet

style은 정해진 속성을 입력하여 웹 페이지를 꾸미는 것

stylesheet는 웹 페이지에서 반복적으로 쓰는 style을 모아놓은 것

✓ 스타일 기본 형식

```
p{text-align: center;}
```

선택자 style속성 속성 값

▶ CSS 적용

✓ 내부 스타일 시트

<style></style> 내부에 스타일 정보를 저장하는 방법

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

✓ 외부 스타일 시트

<link>태그를 이용하여 css파일을 읽어와서 스타일 적용

```
<link href="css 파일 경로" rel="stylesheet" type="text/css">
```

▶ CSS 적용

✓ 인라인 스타일 시트

태그 내부에 스타일 정보를 지정하는 방법

```
<html>
        <head></head>
        <body>
                <p style="스타일정보;">
        </body>
</html>
```

▶ CSS 선택자

구 분	내 용
전체 선택자	*
태그 선택자	태그(h1, p, li 등)
아이디 선택자	#아이디명
클래스 선택자	.클래스명

▶ CSS 선택자

✓ 전체 선택자

<html>에 있는 모든 태그에 적용되는 스타일

* 전체 태그는 body태그에 있는 요소 뿐만 아니라 html, head에도 적용 됨

*{설정내용;}

✓ 태그 선택자

특정 태그에 적용되는 스타일

* 여러 개 태그 선택 시에는 ' , '(쉼표)로 구별하여 작성

태그명{설정내용;}

▶ CSS 선택자

✓ 아이디/클래스 선택자

아이디 선택자 : 태그 속성 id를 설정하고 id 값을 선택자로 하는 것 (#)

클래스 선택자 : 태그 속성을 class로 설정하고
class값을 선택자로 하는 것(.)

* 클래스 선택자는 중복이 허용되나 id선택자는 중복이 허용 안 됨
  코드 상에서 에러는 없지만 javascript의 DOM에서 id값으로 페이지 요소를 가져오기 때문에 중복을 허용하지 않음
* 클래스는 중복을 허용하기 때문에 영역을 정확하게 하기 위해
  태그 선택자와 같이 쓰는 경우도 있음

#아이디 명{설정내용;}

.클래스 명{설정 내용;}

JavaScript

작성 및 실행

자바스크립트 선언

HTML에서 제공하는 <script></script>태그를 사용하며
자바스크립트 작성 영역을 설정하고 그 사이에 자바스크립트 코드 작성
type속성은 브라우저 호환성을 위해 사용되나 default값으로 생략 가능

```
<script type="text/javascript">
        자바스크립트 내용
</script>
```

* language속성과 charset속성이 있었지만 language속성은 폐기되고
  charset속성은 meta태그가 적용되기 때문에 사용할 필요 없음

데이터 출력

▶ 데이터 출력

코드	설명
document.write(내용);	브라우저 화면 상의 페이지에 값 출력
window.alert(내용);	내용을 메시지 창에 출력 * window객체는 모두 적용되는 것으로 생략 가능
innerHTML = 내용;	태그 엘리먼트의 내용을 변경하여 출력
console.log(내용);	개발자 도구 화면의 콘솔에 출력

HTML태그 접근

메소드

메소드	설명
getElementById("아이디명")	태그의 id 속성 값을 이용해 태그 엘리먼트 객체의 정보를 가져 옴
getElementsByName("이름")	태그의 name속성 값을 이용해 태그 엘리먼트의 객체 정보를 배열에 담아 가져 옴 같은 이름의 태그가 여러 개 존재할 수 있기 때문에 기본적으로 배열로 리턴
getElementsByTagName("태그명")	태그 명을 이용하여 해당 태그들의 객체 정보를 배열에 담아 가져옴

document.getElementById()

HTML태그의 id속성 값은 페이지에서 유일한 식별자 역할을 하도록 권장

* 리턴 값 : 단일 값(id는 중복 허용 안 함)

var 변수 = document.getElementById("아이디명");

* 변수는 객체를 의미하는 레퍼런스 변수

document.getElementsByName()

HTML태그의 name속성 값으로 객체 정보를 가져올 때 사용

* 리턴 값 : 배열(name은 중복 가능)

var 변수 = document.getElelemtsByName("이름");

* 변수는 배열이 됨

document.getElementsByTagName()

HTML태그의 태그 명을 이용해 태그들을 한꺼번에 가져와 순서대로 반환

* 리턴 값 : 배열(태그 중복 가능)

var 변수 = document.getElementsByTagName("태그 명");

* 변수는 배열이 됨

기본 문법

▶ 변수

✓ 변수 선언

- 변수 종류 : 멤버 변수와 지역 변수
- 멤버 변수 : 전역 변수, 기본적으로 window객체의 멤버 변수

변수에 대한 자료형은 있으나 선언하지는 않음

형식	설명
변수명 = 값;	window.변수 명 또는 this.변수명과 같은 의미 선언 시 변수 명에 var를 붙이지 않으면 전역변수로 간주
var 변수명 = 값;	변수 선언 시 변수명 앞에 var를 붙이면 지역변수

▶ 자료형

✓ 문자열(String)

"", ''로 묶여있는 리터럴

내장 객체로 String객체, 기본적인 메소드 존재

메소드	내용
toUpperCase()	모든 문자 대문자로 변환
toLowerCase()	모든 문자 소문자로 변환
length	글자 개수 조회용 멤버변수
indexOf()	찾는 문자의 순번(위치) 리턴
lastIndexOf()	뒤에서부터 찾는 문자의 순번 리턴
charAt()	찾는 위치의 문자 리턴
substring()	값을 일부분만 리턴
split()	토큰 문자로 분리한 문자열 배열 리턴

▶ 자료형

✓ 숫자(number)

정수형 숫자와 부동소수점 숫자로 구분

내장 객체로 Math객체 제공, 기본 메소드 존재

메소드	내용
Math.abs()	절대값 리턴
Math.random()	임의의 난수발생 리턴(소수점)
Math.round()	반올림처리 후 리턴
Math.floor()	부동소수점 숫자를 정수로 리턴(소수점 자리버림)
Math.ceil()	소수점 자리에서 무조건 올림

* NaN(Not a Number) : 숫자가 아닌 데이터를 숫자처럼 사용할 때 출력

▶ 연산자

연산자 종류	연산자
최우선 연산자	(), [], .
단항 연산자	++, --, + sign, - sign
산술 연산자	+, -, *, /, %
관계 연산자	>, <, >=, <=, ==, !=, ===, !==
논리 연산자	&&, ||
대입 연산자	=
복합대입연산자	+=, -=, *=, /=, %=
삼항 연산자	? : ;

* 연산자 우선순위 : 최우선 > 단항 > 산술 > 관계 > 논리 > 삼항 > 대입

▶ 제어문

✓ 조건문

if, if~else, if~else if~else, switch문, 짧은 조건문(&&, ||)

✓ 반복문

for, while, do~while, for in문

✓ 분기문

continue, break문

배열

▶ 배열

다양한 타입의 데이터를 보관하는 변수 모음으로

[ ]를 통해 생성과 초기화를 동시에 처리 가능

자료형 지정이 없어 모든 자료형 데이터로 저장 가능

* 모든 자료형 : 숫자, 문자열, 함수, Boolean, undefined, 객체

var 변수 명 = [값1(숫자), 값2(문자), 값3(객체), 함수 ….];

배열 선언

new연산자와 Array객체를 통한 배열 선언

크기를 정하지 않은 배열 선언 : var 변수 명 = new Array();

크기를 정한 배열 선언 : var 변수 명 = new Array(개수);

배열 초기화

new연산자를 활용한 초기화

var 배열변수 = new Array(값1, 값2, 값3, …, 값n);

[ ]를 활용한 초기화

var 배열변수 = [값1, 값2, 값3, …, 값n];

▶ 배열에 값 대입

값 입력 시 index번호 활용 * 번호 범위 : 0 ~ (지정 크기-1)

배열에 값 대입 : 배열[첨자] = 값;

변수에 배열 값 대입 : 변수 = 배열[첨자];

함수

▶ 함수 선언

반환 값 선언 없이 function 키워드만 이용하여 사용

function키워드에 함수 명을 작성하여 사용하는 방법(선언적 함수)과

function에 함수 명 작성하지 않고 변수에 대입하는 방법(익명 함수),

호출 없이 바로 사용하는 스스로 동작하는 함수가 있음

▶ 함수 선언

✓ function에 함수 명 작성

선언적 함수

```
function 함수명([매개변수]){
        처리 로직;
        [return 되돌려줄 값;]
}
```

▶ 함수 선언

✓ 변수에 함수를 넣어 작성

callback형태, 익명함수

```
var f1 = function ([매개변수]){
                    처리 로직;
                    [return 되돌려줄 값;]
          }
```

함수 선언

✓ 스스로 동작 함수

호출 없이 바로 실행

```
(function(){
        처리 로직
})();
```

▶ 함수 호출

함수는 반드시 선언(정의)이 되어야만 실행 가능

원하는 기능에 대한 함수를 호출하는 것을 함수 실행이라고 부르며

return 값이 있다면 리턴 값을 받을 변수가 있어야 함

return 값이 없는 경우

```
함수 명();
함수 명(인자 값1, 인자 값2, ...);
```

return 값이 있는 경우

```
var 변수 명 = 함수 명();
var 변수 명 = 함수 명(인자 값1, 인자 값2, ...);
```

▶ 매개변수와 return

✓ 매개변수(전달인자)

호출하는 코드와 호출되는 함수를 연결해주는 변수를 매개변수라 함
지정된 매개변수보다 많게 선언하고 호출하는 것을 허용하나
초과되는 매개변수는 무시함
지정된 매개변수보다 적게 선언하고 호출하는 것도 허용하나
선언이 안 된 매개변수는 undefined로 자동 설정 됨

✓ return [되돌려 줄 값]

return은 함수를 호출한 위치로 돌아가라는 의미로
return 값(되돌려 줄 값) 미지정 시 undefined자료형으로 반환 됨

매개변수 함수

함수도 하나의 자료형으로 매개변수로 전달 가능

```
function 함수 명(매개변수 명){
        매개변수 명(); // 호출
}
```

객체

객체 선언 및 호출

✓ 선언 방법

```
var 변수 명(객체 명) = {
        속성(키 값) : 값,
        속성(키 값) : 값,
        속성(키 값) : 값
};
```

✓ 속성 값 접근

```
변수 명(객체 명)['요소 명(키 값)'];
또는
변수 명(객체 명).요소 명(키 값);
```

DOM

▶ DOM(Document Object Model)

HTML에 있는 태그를 객체화하여 자바스크립트에서 다룰 수 있게 한 것으로 모든 노드 객체에 접근할 수 있는 요소와 메소드 제공

✓ 노드

HTML에 있는 태그를 구조화(트리)하였을 때 각각의 태그가 노드

✓ 요소 노드와 텍스트 노드

- 요소 노드(Elements Node) : 태그 그 자체 의미
- 텍스트 노드(Text Node) : 태그에 기록되어 있는 문자

* 텍스트 노드를 가지는 태그(h?, p 등)와 가지지 않는 태그(img 등)가 있음

* w3schools.com의 references 참조

▶ 텍스트 노드 있는 문서 객체 작성

요소 노드와 텍스트 노드를 생성하고 이를 body노드의 자식으로 포함 가능

요소노드 생성 → 텍스트 노드 생성 → 요소 노드에 텍스트 노드 추가
→ body객체에 요소 노드 추가

✓ 메소드

document.createElement(태그명)	요소노드 생성
document.createTextNode(내용)	텍스트노드 생성
객체명.appendChild(node)	태그에 자손으로 노드를 추가

▶ 텍스트 노드 없는 문서 객체 작성

요소 노드 생성하고 속성을 설정한 후 이를 body노드의 자식으로 포함 가능

요소노드 생성 → 생성된 노드 속성 설정→ body객체에 요소 노드 추가

✓ 메소드

객체명(변수).속성=속성값	태그 속성값 설정
객체명.setAttribute(속성명, 속성값)	태그에 속성값 설정
객체명.getAttribute(속성명)	태그 속성확인
객체명.appendChild(node)	태그에 자손으로 노드를 추가

▶ 문서 스타일 수정

style객체를 이용하여 문서의 스타일 변경

객체 명.style.속성 명 = 속성 값;

* 자바스크립트에서 식별자에 '-'를 쓰지 못하기 때문에
  속성 명이 CSS에서 쓰는 것과 다른 부분이 있음(ex. background-color → backgroundColor)

▶ 문서 객체 제거

페이지 내 작성된 문서의 객체(태그)를 제거하는 것

✓ 메소드

document.removeChild(객체명)	body태그 자손 태그 제거
객체명.parentNode.removeChild(객체명)	제거할 객체를 기준으로 그 상위 객체로 가서 해당객체를 삭제

이벤트

▶ 이벤트

✓ 이벤트 활용

이벤트 속성과 이벤트 핸들러(함수)를 연동하여
이벤트 발생 시 특정 기능을 하도록 하는 것

✓ 이벤트 설정 방법

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

▶ 이벤트 설정 방법

✓ 고전 이벤트 모델

요소 객체가 갖고 있는 이벤트 속성에 이벤트 핸들러를 연결하는 방법으로 이벤트 객체를 제거할 땐 속성 값에 null을 넣어주면 됨

이벤트 발생 객체는 핸들러 내부에서 this로 표현(스타일 변경 가능)

매개 변수로 이벤트 정보 전달(e, window.event) * 이벤트 객체 전달

ex. 클릭 시 이벤트 설정

```
var h = document.getElementById('id명');
h.onclick = function(){
        수행 기능 설정;
        h(this).onclick = null; // 한 번만 실행
};
```

▶ 이벤트 설정 방법

✓ 인라인 이벤트 모델

요소 내부에 이벤트를 작성하는 방식으로

인라인 방식은 <script>태그에 있는 함수를 호출하는 방식 선호

ex. 클릭 시 이벤트 설정

<h1 onclick='처리로직'></h1>

또는

<h1 onclick='스크립트 내 함수 호출'></h1>

▶ 이벤트 설정 방법

✓ 표준 이벤트 모델

W3C에서 공식적으로 지정한 이벤트 모델

한 번에 여러 개 이벤트 핸들러 설정 가능

this키워드가 이벤트 발생 객체를 의미함

ex. 클릭 시 이벤트 설정

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

✓ 메소드

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