프런트 엔드와 백 엔드

프런트 엔드(Front-End): 눈으로 보이는 모든 부분을 동적 웹 페이지로 제작

언어: HTML, CSS, Javascript, jQuery

백 엔드(Back-End): 사용자가 입력한 데이터를 데이터베이스에 저장하거나 검사하는 등 눈에 보지 않는 영역을 개발

언어: ASP, PHP, JSP

디자인 개발 순서

1. 개발 의뢰

2. 기획안 작성

3. UI 시안

4. 프런트 엔드(Front-End)

5. 백 엔드(Back-End)

자바스크립트

- 프런트 엔드 개발 언어, 문서에 동작 부여

- 웹 페이지, 웹 서버, 웹 에플리케이션, 모바일 앱, 데스크탑 앱, 게임, 예술

크롬 브라우저 개발자 도구 패널

Elements: HTML 요소에 적용된 스타일을 검사할 수 있다

Console: 자바스크립트 오류 체크는 물론 디버깅을 할 수 있다

Source: 브라우저가 자바스크립트 소스를 파싱 (해오는 과정을 보여준다) 소스 패널도 오류 체크와 디버깅을 할 수 있다

Visual Studio Code 화면 살펴보기



뷰포트

화면 요소가 보여지는 영역으로 같은 HTML문서가 실행된 장치에 따라 다르게 보이는 이유가 장치마다 뷰포트 적용 방식이 다르기 때문임

- 모바일 환경에서도 데스크톱 환경처럼 웹 페이지 전체를 자연스럽게 브라우징 할 수 있도록 풀브라우징(full-browsing) 지원

- 모바일 기기는 화면 크기와 해상도가 다양하고 pc화면과 비슷한 해상도로 표시되므로, 축소 시 작아지는 글씨의 가독성을 해결하기 위해 뷰포트 사용

뷰포트 메타 태그

<meta> 태그는 <head> 태그 안에 포함시킨다

<meta> 태그 뷰포트 속성

- width: 화면의 너비(px, device-width)

- height: 화면의 높이(px, device-height)

- initial-scale: 초기 화면 확대 비율(0 ~ 10.0)

- minimum-scale: 화면 축소 최소 비율(0 ~ 10.0)

- maximum-scale: 화면 확대 최대 비율(0 ~ 10.0)

- user-scalable: 사용자의 확대 혹은 축소 가능 여부(yes or no)

device-width, device-height는 모바일 장치의 실제 크기를 기준으로 뷰포트의 너비, 높이 설정

자바 스크립트 선언문

<head> 혹은 <body>에 <script>로 선언

- html 문서 안에서 선언

<script>안에 작성</script>

- 외부에서 불러오기

<script src=”파일 경로”></script>

주석 처리

html: <!-- 주석 -->

css: /\* 주석 \*/

javascript: // 주석 /\* 주석 \*/

코트 입력시 주의 사항

- 대소 문자를 구분하여 작성

New date(); 🡪 new Date();

- 코드에 마지막에는 세미 클론 작성

document.write(“hello”);

- 따옴표의 겹침 주의

document.write(“hello”hello”hello”);

🡪 document.write(“hello\”hello\”hello”); or document.write(“hello’hello’hello”);

입출력

브라우저 출력: document.write(“hello”);

메시지 출력: alert(“hello”);

프롬프트 다이얼로그 입력: var imput = prompt(“출력문”, ”기본값”);

확인 다이얼로그 입력: var imput = confirm(“출력문”)

변수 선언;

var 변수명 = 값;

자료형 종류: 문자형, 숫자형, 논리형, undefined $ null

자료형 구별 방법: (typeof 변수)

식별자 규칙

- 키워드 가용 불가

- 숫자로 시작할 수 없음

- 특수 문자는 \_와 $만 사용 가능

- 공백문자 사용 불가

연산자

산술 연산자

+: 더하기

- : 빼기

\* : 곱하기

/ : 나누기

%: 나머지

문자 결합 연산자

- 여러 개의 문자를 하나의 문자형 데이터로 결합한다.

문자형 + 문자형 = 문자형

문자형 + 숫자형 = 문자형

비트 논리 연산자

&: AND 연산 1010(10) & 0011(3) = 0010(2)

| : OR 연산 1010(10) | 0011(3) = 1011(11)

^ : XOR 연산 1010(10) & 0011(3) = 1001(9)

~ : NOT 연산 ~ 1010(10) = 0101(5)

대입 연산자

a = b a = b

a += b a = a + b

a -= b a = a - b

a \*= b a = a \* b

a /= b a = a / b

a %= b a = a % b

a &= b a = a & b

a |= b a = a | b

a ^= b a = a ^ b

증감 연산자

a++ a-- 후위 연산자 : 반환 후 증간 연산자 적용

++a --a 전위 연산자 : 반환 전 증간 연산자 적용

비교 연산자

A > B A가 B 보다 크다

A < B A가 B 보다 작다

A >= B A가 B 보다 크거나 같다

A <= B A가 B 보다 작거나 같다

A == B 자료형(type)이 숫자형이든 문자형이든 상관없이 표기된 값이 일치하면 true

A != B 자료형(type)이 숫자형이든 문자형이든 상관없이 표기된 값이 다르면 true

A === B자료형(type)과 표기된 값이 모두 일치하면 true

A !== B 자료형(type) 또는 표기된 값이 일치하지 않으면 true

논리 연산자

|| OR연산자

&& AND연산자

! NOT연산자

연산자 우선순위

1. ()

2. 단항 연산자(++, --, !)

3. 산술 연산자(\*, /, %, +, -)

4. 비교 연산자(>, >=, <, <=, ==, ===, !==, !=)

5. 논리 연산자(&&, ||)

6. 대입 연산자(=, +=, -=, \*=, /=, %=)

삼항 조건 연산자

조건식 ? (true일 때 리턴 값) : (false일 때 리턴 값);

ex) var num = a > b ? 1 : 2

형변환

암시적 형변환: 자바스트립트가 필요에 의해 형을 자동으로 변환시키는 것.

숫자형 + 문자형 = 문자형

논리형 + 문자형 = 문자형

논리형 + 숫자형 = 숫자형

명시적 형변환: 개발자가 어떤 형으로 바꿀지 명시해주는 것

문자를 숫자형으로 전환

- 정수형

parseInt(“문자형 숫자”)

Number(“문자형 숫자”)

- 실수형

parseFloat(“문자형 숫자”)

Number(“문자형 숫자”)

숫자를 문자로 전환

일반 문자 변환: String(숫자) / 숫자.toString()

16진수 변환: 숫자.toString(16)

실수 문자형: 숫자.toFixed(자리수)

조건문

if문

if – 조건문

else if – 다중 조건

else – 탈퇴 여부

if(조건){코드}

else if(조건){코드}

else{코드}

switch문

switch – 조건문

case – 조건

default – 탈퇴 여부

switch(변수 or 값){

case 값1: 코드1;

break;

case 값2: 코드2;

break;

default: 코드3;

break;

}

반복문

for문 – 반복 횟수에 비중을 들 때 사용

for(초깃값; 조건식; 증감식) {코드;}

while문 – 조건에 비중을 들 때 사용

while(조건식) {코드;}

do while 문 – while 문에서 코드 실행 후 조건을 검사

do{코드;}while(조건식)

break문 – 반복문 강제 종료

continue문 – 반복문에서 아래 코드 무시 후 증감식/조건식 수행

배열

여러 개의 데이터를 하나의 변수에 저장한 자료형

var 변수 = [값1, 값2, 값3];

값들은 인덱스 0번부터 시작함

값1 = 인덱스 0, 값2 = 인덱스 1

함수

데이터를 전달받아 처리한 후 결과를 리턴하는 코드블록

일반 함수 선언

function 함수명(){코드;} 사용 = 함수명();

익명 함수 선언

var 참조 변수 = function(){코드;} 사용 = 참조변수();

일반 함수 익명 함수 차이점

- 일반 함수는 호출문이 먼저 나와도 호출되지만 익명함수는 먼저 호출 할 수 없다.

매개변수

함수를 호출할 때 외부에서 받은 값을 저장해서 함수 내에서 사용 가능 하도록 한다.

function 함수명 (매개 변수1, 매개 변수2){ 코드; }

함수명(데이터1, 데이터2);

매개변수가 없는데 외부에서 값을 받은 경우

배열 arguments에 저장된다.

function 함수명(){

arguments; }

함수명(데이터1, 데이터2);

return

결과값 반환 및 함수 탈출

전역변수 – 자바 스크립트 코드 안 어디에서나 사용가능한 변수

지역변수 – 특정 함수 안에서만 사용가능한 변수

즉시 실행 함수

함수 선언과 동시에 함수를 호출

예시) var result(function(a, b) { return a+b; } (10, 20) );

재귀 함수

함수 정의문 내에서 함수를 선언해서 다시 호출하는 것

function myFunc() {

코드;

myFunc(); }

콜백 함수

함수를 매개 변수로 전달하는 함수

function myFunc(callback) {

코드;

callback(결과값); }

function theFunc(매개 변수) {

코드; }

myFunc(theFunc);

내장 함수

parseInt() – 문자형 데이터를 정수형 데이터로 변환

parseFloat() – 문자형 데이터를 실수형 데이터로 변환

String() – 문자형 데이터로 변환

Number() – 숫자형 데이터로 변환

Boolean() - 논리형 데이터로 변환 0, null, undefined, 빈문자를 제외한 모든 데이터에 대해 true 반환

isNaN() – 숫자가 아니면 true 반환

isFinite() – 숫자이면 true 반환

eval() – 수식이나 자바스크립트 코드를 문자형 데이터로 전달받아 실행한 후 결과를 반환

시간과 관련 내장 함수

setTimeout(function, millisecond) – 일정 시간 후 함수를 한 번 실행

setInterval(function, millisecond) – 일정시간 마다 함수를 반복해서 실행

clearTimeout(id) – 일정 시간 후 함수를 한 번 실행하는 것을 중지(setTimeout 중지)

clearInterval(id) – 일정 시간마다 함수를 반복하는 것을 중지(setInterval 중지)

인코딩 디코딩 관련 내장 함수

encodeURL(url) – 영문, 숫자, 일부 기호( ; : / = ? & @ + - \* . )제외 문자를 유니코드 값으로 인코딩

encodeURLComponent(url) – 영문, 숫자를 제외한 모든 문자를 인코딩

decodeURL(encodedurl) – 유니코드 값을 디코딩하여 문자화(encodeURL 디코딩)

decodeURLComponent(encodedurl) – 유니코드 값을 디코딩하여 문자화(encodeURLComponent 디코딩)

객체

자바스크립트는 객체(object) 기반 언어로 객체는 기능과 속성을 가진다.

객체의 종류

내장 객체(기본 객체): 자바스크립트 엔진에 내장

Ex) 배열(Array), 날짜(Date), 문자(String), 수학(Math)

브라우저 객체(BOM): 브라우저에 계층 구조로 내장되어 있는 객체

Ex) window, screen, location, history, navigator

문서 객체(DOM): HTML 문서 구조, HTML 태그들을 브라우저가 객체화 한 것

Ex) Html, head , body, img,...

내장 객체 생성

var 참조 변수 = new 생성 함수();

객체 기본 사용법

객체.메서드(); 객체의 메서드 실행

변수 = 객체.속성; 객체의 속성값을 가져옴

객체.속성 = 값; 객체의 속성값 변경

날짜 객체(Date)

var 변수 = new Date(); – 현재 날짜와 시간 값으로 초기화된 객체 생성

var 변수 = new Date(년도, 월-1, 일); – 년, 월(0 ~ 11 – 인덱스 값), 일

var 변수 = new Date(“년도/월/일”); - 년, 월, 일

var 변수 = new Date(년도, 월-1, 일, 시, 분, 초); - 년, 월(0 ~ 11 – 인덱스 값), 일, 시, 분, 초

메서드

날짜 정보를 가져올 때(get)

getFullYear() – 연도 정보를 가져옴

getMonth() – 월 정보를 가져옴(1월:0 ~ 12월:11)

getDate() – 일 정보를 가져옴

getDay() – 요일 정보를 가져옴(일:0 ~ 토:6)

getHours() – 시 정보를 가져옴

getMinutes() – 분 정보를 가져옴

getSeconds() – 초 정보를 가져옴

getMilliSeconds() – 밀리초 정보를 가져옴

getTime() – 1970년 1월 1일부터 경과된 시간을 밀리초로 표기

toUTCString() – UTC 표현 방식으로 문자형 데이터로 반환

toLocaleString() – 로컬 표현 방식으로 문자형 데이터로 변환

toLocaleTimeString() – 로컬 표현 방식으로 시간 정보만 문자형 데이터로 반환

날짜 정보를 수정할 때(set)

setFullYear() – 연도 정보만 수정

setMonth() – 월 정보만 수정

setDate() – 일 정보만 수정

setHours() – 시 정보만 수정

setMinutes() – 분 정보만 수정

setSeconds() – 초 정보만 수정

setMilliseconds() – 밀리초 정보만 수정

setTime() – 1970년 1월 1일부터 경과된 시간을 밀리초로 정보만 수정

숫자 객체(Math)

Math.abs(숫자) – 숫자의 절대값

Math.max(숫자1, 숫자2, 숫자3) – 숫자들 중 가장 큰 값 반환

Math.min(숫자1, 숫자2, 숫자3) – 숫자들 중 가장 작은 값 반환

Math.pow(숫자, 제곱값) – 숫자의 거듭 제곱 값을 반환

Math.random() – 0~1 사이의 난수 반환

Math.round(숫자) – 소수점 첫째 자리에서 반올림한 정수 값 반환

Math.ceil(숫자) – 소수점 첫째 자리에서 무조건 올림하여 정수 값 반환

Math.floor(숫자) – 소수점 첫째 자리에서 무조건 내림하여 정수 값 반환

Math.sqrt(숫자) – 숫자의 제곱근을 반환

Math.PI – 원주율 3.141592…..

문자열 객체(String)

var 변수 = new String(“문자열”);

var 변수 = “문자열”;

문자열 관련 메서드

charAt(index) - index번호에 해당하는 문자 반환

indexOf(“찿을문자”) – 왼쪽부터 “찿을문자”와 일치하는 첫문자의 인덱스 번호 반환. 찾는 문자가 없으면 -1반환

lastIndexOf(“찿을문자”) – 오른쪽부터 “찿을문자”와 일치하는 첫문자의 인덱스 번호를 반환. 찾는 문자가 없으면 -1 반환

match(“찿을문자”) – 왼쪽부터 “찿을문자”와 일치하는 첫문자의 인덱스 번호 반환. 찾는 문자가 없으면 null 반환

search(“찿을문자”) – 왼쪽부터 “찿을문자”와 일치하는 첫문자의 인덱스 번호 반환, 찾는 문자가 없으면 -1 반환

replace(“바꿀 문자”, “새문자”) – 문자열에서 “바꿀문자”를 찾아 “새문자”로 변경

slice(a, b) – 인덱스 a에서 b앞까지 문자열 반환 인덱스 b가 생략되면 a부터 끝까지 반환

substring(a, b) – 인덱스 a에서 b앞까지 문자열 반환

substr(a, 문자개수) – 인덱스 a부터 지정한 문자개수만큼 반환

split(“문자”) – 지정한 “문자”를 기준으로 문자 데이터를 나눠 배열에 저장하여 반환

toLowerCase() – 문자열 전부 소문자로 변환

toUpperCase() – 문자열을 전부 대문자로 변환

length – 문자열에서 문자 개수 반환

concat(“새로운 문자”) – 문자열에 “새로운 문자”를 결합

charCodeAt(index) – index에 해당하는 문자를 반환

trim() – 문자열 앞뒤 공백 제거

배열 객체(Array)

배열은 여러 개의 데이터를 한꺼번에 저장하는 자료형으로 모든 자료형을 배열안에 넣을 수 있다.

var 변수 = [값1, 값2, 값3, ……]

var 변수 = new Array();

var 변수 = new Array(값1, 값2, 값3 …..);

배열 관련 메서드

join(연결문자) – 연결문자를 기준으로 1개의 문자형 데이터로 변환

reverse() – 배열 객체 데이터 순서를 거꾸로 바꾼 후 변환

sort() – 배열 객체의 데이터를 오름차순으로 정렬

slice(index1, inde2) – 배열 객체의 데이터 중 인덱스 구간(index1부터 index2 이전까지)만큼 잘라서 반환

concat() – 2개의 배열 객체를 하나로 결합

pop() – 배열 객체의 마지막 인덱스의 데이터를 삭제. 삭제된 데이터는 반환

push(new data) – 배열 객체의 마지막 인덱스에 “new data”삽입

shift() – 배열 객체의 첫번째 인덱스의 데이터 삭제, 삭제된 데이터는 반환

unshift(new data) – 배열 객체의 맨 앞의 인덱스에 “new data” 삽입

length – 배열에 저장된 총 데이터의 개수 반환

사용자 객체

개발자가 새로운 객체를 생성

var 객체명 = new Object();

객체명.속성 = “값1”;

객체명.속성 = “값2”;

var 객체명 = {

속성 : “값1”;

속성 : “값2”;

}

in 키워드

in 키워드를 사용하면 해당속성이 객체 안에 있는지 확인할 수 있다.

‘속성’ in 객체명 = true or false

with 키워드

사용할 객체를 지정하여 코드를 간결화 시켜준다

with (객체명) {

속성;

}

객체 생성자 함수

new 키워드로 객체를 생성할 수 있는 함수를 의미한다.

function 함수명(매개변수1, 매개변수2, …){

this.속성명 = “값” or 매개변수;

this.함수명 = function(){

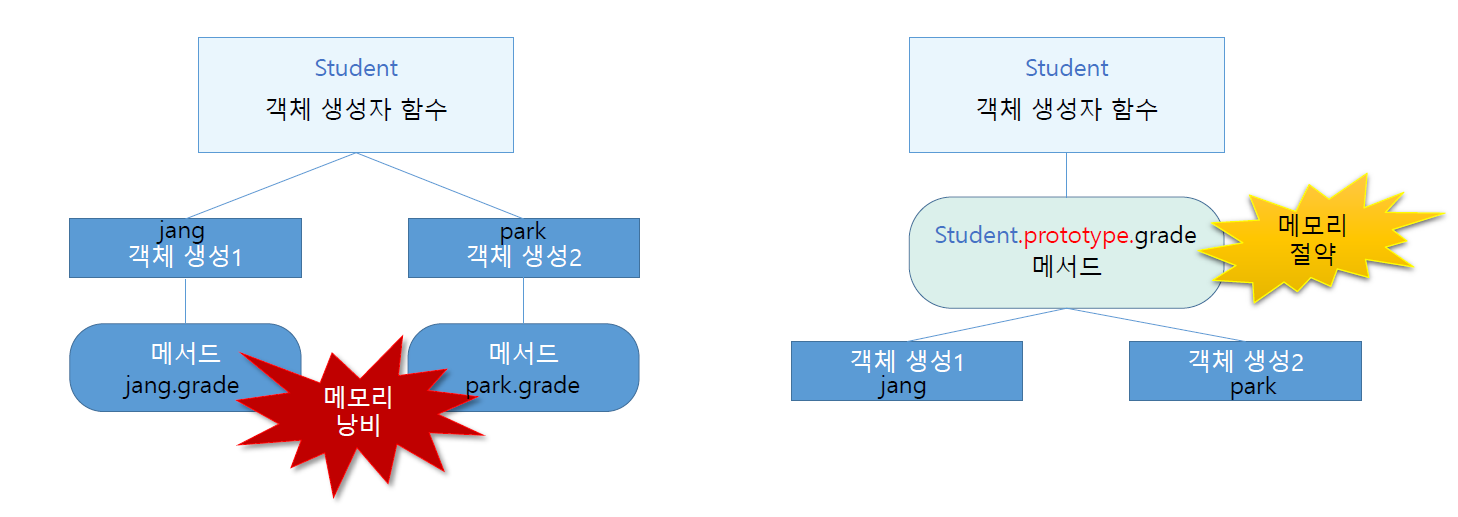
자바스트립트 코드; }

}

var 참조 변수 = new 함수명(값1, 값2, …);

prototype 메서드

생성자 함수로 생성된 객체가 공통으로 가지는 공간



function 함수명(매개변수1, 매개변수2, …) {

this.속성명 = “값” or 매개변수;

}

함수명.prototype.함수명 = function(){

자바 스크립트 코드

}

var 참조 변수 = new 함수명(값1, 값2, …);

브라우저 객체 모델 (BOM, Browser Object Model)

브라우저에 내장된 객체를 브라우저 객체라고 한다

최상위 객체는 window 객체이며 계층적 구조로 이루어져 있다.

window

- document: body, div, img, a, from. input…

- screen: width, height…

- location: href, host…

- history: back, forward…

- navigator, appName, appVersion…

window 객체

open(“URL”, “새 창 이름”, “새 창 옵션”) - URL페이지를 새창으로 나타냄. 새로운 window 객체 생성.

속성

width – 새 창의 너비 (픽셀 값)

height – 새 창의 높이 (픽셀 값)

left – 새 창의 수평(x축) 위치

top – 새 창의 수직(y축) 위치

scrollbars – 새 창의 스크롤바 유무 (yes or no)

location – 새 창의 URL주소창 유무 (yes or no)

menubar – 메뉴의 유무 (yes or no)

resizable – 화면 크기 조절가능 유무 (yes or no)

status – 상태표시줄 유무 (yes or no)

toolbars – 두구상자영역의 유무 (yes or no)

alert(“메시지”) – “메시지”의 내용으로 경고장 출력

prompt(“질문”, “기본값”) – “질문”으로 사용자의 입력을 받음

confirm(“질문”) – “질문”에 대한 [확인], [취소]버튼으로 사용자의 입력을 받음

setInterval(function(){코드}, 시간) – 지속적으로 일정한 시간 간격으로 함수를 호출

setTimeout(function(){코드}, 시간) – 단 한 번 일정한 시간 후 함수 호출

clearInterval – setInterval()메서드 제거

clearTimeout – setTimeout()메서드 제거

moveBy(x, y) – 현재 창을 기준으로 새 창의 상대적 위치 이동

moveTo(x, y) – 새 창의 절대적 위치 이동

resizeBy(x, y) – 현재 창을 기준으로 새 창의 상대적 크기 변경

resizeTo(x, y) – 새 창의 절대적 크기 변경

scrollBy(x, y) – 현재 창을 기준으로 새 창의 스크롤 위치를 상대적으로 이동

scrollTo(x, y) – 새 창의 스크롤 위치를 절대적으로 이동

focus() – 창의 초점을 맞춤

blur() – 창에 맞춘 초점을 제거

close() – 창 닫음

screen 객체

- 사용자의 모니터 정보(속성)을 제공하는 객체

screen.width – 화면의 너비값

screen.height – 화면의 높이값

screen.availWidth – 작업표시줄을 제외한 화면의 너비값(실제 화면에서 사용가능한 너비)

screen.availHeight – 작업표시줄을 제외한 화면의 높이값(실제 화면에서 사용가능한 높이)

screen.colorDepth – 사용 가능한 색상수

screen.pixelDepth – 한 픽셀당 비트 수

location 객체

- URL에 대한 정보(속성) 과 새로 고침(reload) 메서드 제공

location.href – URL 주소

location.hash – URL의 해시값 (#에 명시된 값) : #beta

location.hostname – 호스트 이름 : localhost

location.host – 호스트 이름과 포트 번호 : localhost:30763

location.protocol – 프로토콜 종류 : http:

location.search – URL 의 쿼리(요청값) : ?pageNum=1

location.assign(url) – url주소로 페이지 이동

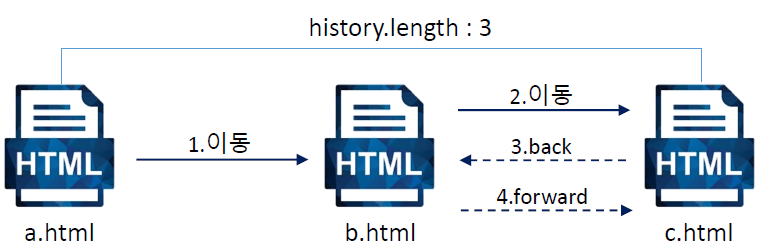
location.reload() – 새로고침

location.replace(url) – url 주소로 페이지 이동 (assign과 차이점 : 이전 페이지로 되돌아가지 못함)

history 객체

- 사용자가 방문한 사이트의 기록을 남기고 이전 방문 사이트와 다음 방문 사이트로 다시 돌아갈

수 있는 속성과 메서드 제공



history.back() – 이전 방문 사이트

history.forward() – 다음 방문 사이트

history.go(이동숫자) – 이동숫자에 입력한 숫자만큼의 이전/다음 방문 사이트로 이동

history.length – 방문 기록에 저장된 목록 개수

navigator 객체

- 방문자가 사용하는 브라우저 정보와 운영체제 정보를 제공하는 객체

navigator.appCodeName – 브라우저 코드 이름. 현재 모든 브라우저는 ‘Mozilla’

navigator.appName – 브라우저 이름. 현재 모든 브라우저는 ‘Netscape’

navigator.appVersion – 브라우저 버전 정보. 현재 모든 브라우저는 ‘5.0(Windows)’

navigator.language – 브라우저가 사용하는 언어

navigator.product – 브라우저의 엔진 이름. 크롬의 경우 ‘Gecko’

navigator.platform – 사용중인 운영체제의 시스템 환경(운영체제가 64bit라도 브라우저가 32bit면‘Win32’ 라고 출력)

navigator.online – 온라인 상태 여부

navigator.userAgent – 웹 브라우저의 전체 정보

문서 객체 모델(DOM, Document Object Model)

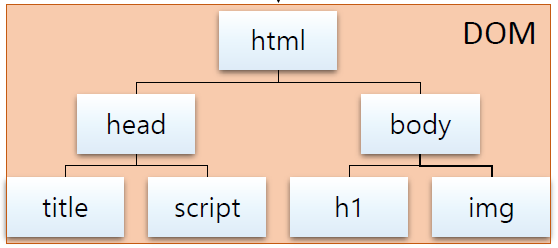
브라우저는 각 HTML 태그를 하나의 객체로 만든다. 이를 HTML DOM 객체 또는 DOM이라고 한다

HTML 페이지가 출력된 후 DOM 객체를 통해 HTML 태그가 출력된 모양과 콘텐츠를 제어할 수 있다

DOM트리

- 브라우저는 HTML 태그의 포함 관계에 따라 DOM 객체들을 트리(tree) 구조로 만든다.

- 자바스크립트 코드는 DOM 트리를 따라 DOM 객체들에 접근할 수 있다.



문서 객체 선택 메서드

document.getElementById(“id”) – id 속성이 일치하는 문서 객체를 가져옴

document.getElementsByName(“name”) – name 속성이 일치하는 문서 객체를 배열로 가져옴

document.getElementsByTagName(“tagName”) – tagName 속성이 일치하는 모든 태그의 문서 객체를 배열로 가져옴

document.getElementsByClassName(“className”) – className 속성이 일치하는 모든 태그의 문서 객체를 배열로 가져옴

document.querySelector(“선택자”) – CSS 선택자로 가장 처음 선택되는 문서 객체를 가져옴

document.querySelectorAll(“선택자”) – CSS 선택자를 통해 선택되는 문서 객체를 배열로 가져옴

innerHTML

DOM 객체의 innerHTML은 시작 태그와 종료 태그 사이에 있는 HTML 콘텐츠를 나타내는 문자열 정보다.

문서 객체 스타일 조작

style 식별자 변환

스타일 속성 -> 자바 스크립트 스타일 식별자

background-image -> backgroundImage

background-color -> backgroundColor

box-sizing -> boxSizing

list-style -> listStyle

스타일 객체

style.color – 글자색

style.fontSize – 글자 크기

style.backgroundColor – 배경색

style.width – 너비

style.border – 테두리

style.margin – 상하좌우 여백