그래픽을 위한 웹 기술 기본 II

Web technology basics for graphics

자바스크립트

자바스크립트 객체는 “Everything”
a String, a Number, an Array, a Date....
속성과 메소드(properties and methods).

Properties and Methods

Properties are values associated with an object.
Methods are actions that can be performed on objects.

Properties:

car.name=Fiat

car.model=500

car.weight=850kg

car.color=white

Methods:

car.start()

car.drive()

car.brake()

자바스크립트

자바스크립트 문자열 객체 선언:

```
var txt = "Hello";
```

String 객체를 생성 한 것
String 객체는 길이 속성 값을 가지고 있음
String 객체는 몇 개의 함수도 가지고 있음

http://www.w3schools.com/jsref/jsref_obj_string.asp

자바스크립트

자바스크립트 객체 선언 (key : value)

```
first name : John
last  name : Doe
age        : 50
eyecolor   : blue
```

```
<!DOCTYPE html>
<html>
<body>
<script>
        var person=new Object();
        person.firstname="John";
        person.lastname="Doe";
        person.age=50;
        person.eyecolor="blue";
        document.write(person.firstname + " is " + person.age + " years old.");
</script>
</body>
</html>
```

자바스크립트

Accessing Object Properties

objectName.propertyName

```
var message = "Hello World!";
var x = message.length;
```

12

Accessing Object Methods

objectName.methodName()

```
var message = "Hello world!";
var x = message.toUpperCase();
```

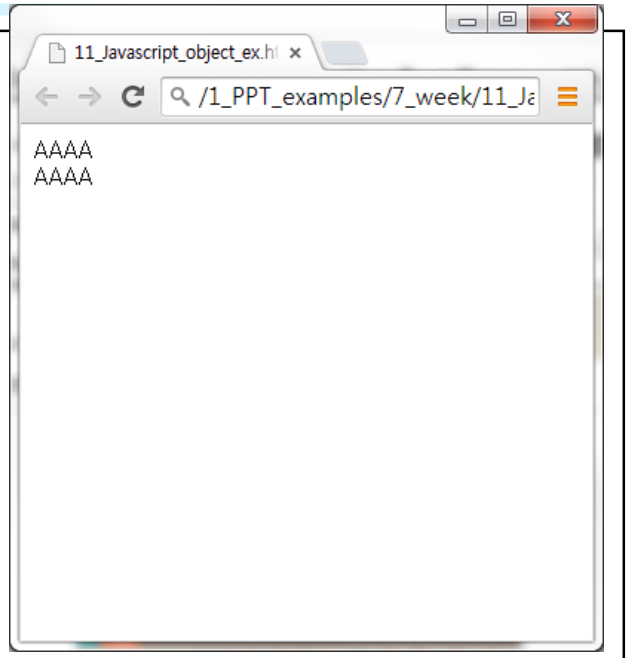
HELLO WORLD!

자바스크립트

```
<!DOCTYPE html>
<html>
<body>

<script>
var person={
  firstname : "John",
  lastname  : "AAAA",
  id        :  5566
};
document.write(person.lastname + "<br>");
document.write(person["lastname"] + "<br>");
</script>

</body>
</html>
```



자바스크립트 함수

호출을 당할 때 실행 되는 코드 블록 단위 모음

호출 되는 시기 :

- 이벤트가 발생 할 때
- 자바스크립트 코드에서 호출 할 경우
- 자동으로 호출 되는 경우

JavaScript 기본 함수 구문

```
function functionName(){
  some code to be executed
}
```

JavaScript 인수를 사용한 함수 구문

```
function functionName(var1, var2){
  some code to be executed
}
```

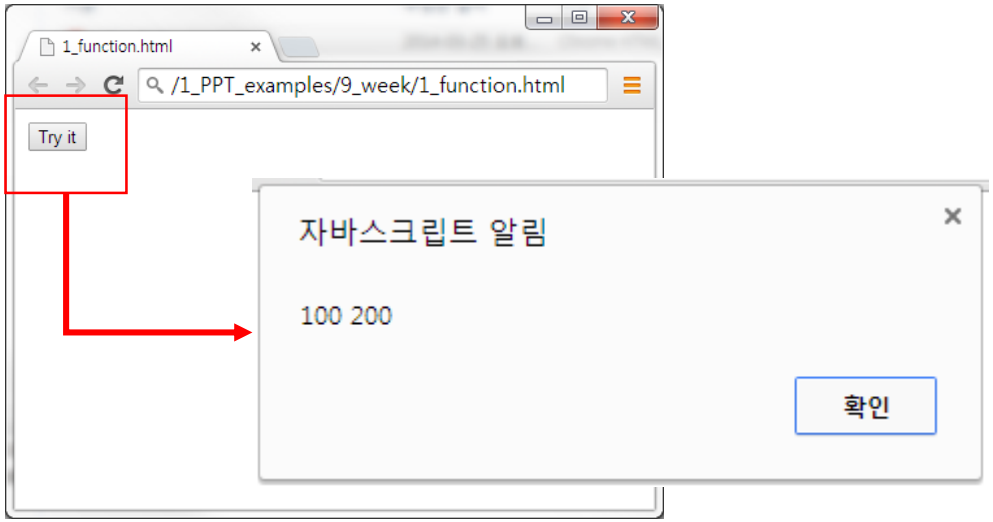
자바스크립트 함수

인수 함수 예제

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function myFunction(a1, a2){
        alert(a1 + " " + a2);
}
</script>
</head>
<body>

<button onclick="myFunction('100','200')">Try it</button>

</body>
</html>
```



자바스크립트 함수

JavaScript 반환 값을 가지는 함수

값을 반환하여 사용하고자 할 때 사용

```
function myFunction(){
        var x = 5;
        return x;
}
```

```
var myVar = myFunction();
```

```
document.getElementById("demo").innerHTML=myFunction();
```

자바스크립트 함수

반환 함수 예제

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function myFunction(a,b){
  return a*b;
}
</script>
</head>
<body>
<p id="demo"></p>

<script>
document.getElementById("demo").innerHTML=myFunction(2,3);
</script>

</body>
</html>
```

자바스크립트 함수

익명 함수 : function(…){ … };

```
var function1 = function(){
        alert("function1");
};
```

function 뒤에 이름이 없고 변수에 넣어서 사용 하는 것이 특징

선언적 함수 : function 함수이름(…){ … };

```
function function1(){
        alert("function1");
}
```

function 뒤에 이름을 작성하여 사용 하는 것이 특징

이 둘의 차이점은??

```
function1(); (X)

var function1 = function(){
        alert("function1");
};

function1();
```

```
function1();

function function1(){
        alert("function1");
}

function1();
```

자바스크립트 전역/지역 변수

Local JavaScript 변수

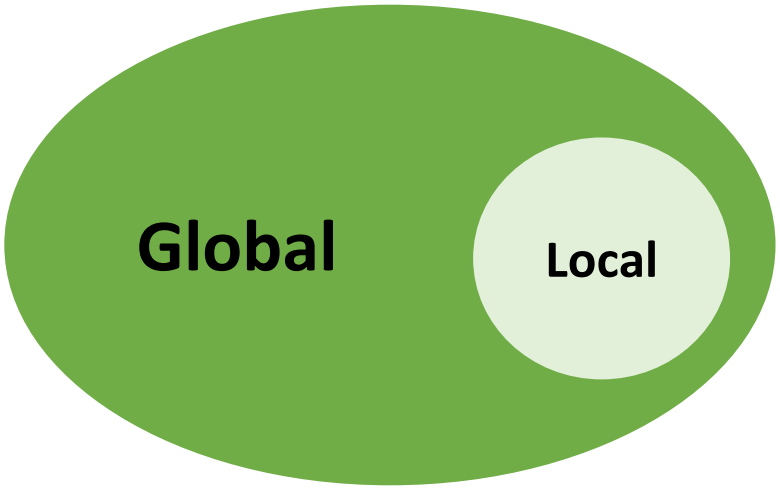
JavaScript 함수 내에서 선언 된 변수로 함수 내에서 활용 가능

함수 내에서 선언 되므로 각 함수마다 같은 이름을 가진 변수들을 선언 할 수 있음

Local 변수는 함수가 끝남과 동시에 삭제됨

Global JavaScript Variables

모든 스크립트 함수에 공유하는 변수




```
function f1(){
        var a = 1;
        alert(a);
}
function f2(){
        var a = 2;
        alert(a);
}
```

```
var a = 1;
function f1(){
        alert(a);
}
function f2(){
        alert(a);
}
```

자바스크립트 전역/지역 변수

Local and Global 함수 예제

```
<!DOCTYPE html>
<html><head>
<script>
var g_a = 100;
function f1(){
  var a = ‘f1’;
  alert(a+’ ‘+g_a);
}
function f2(){
  var a = ‘f2’;
  alert(a+’ ‘+g_a);
}
</script></head><body>

<button onclick="f1()">function 1</button>
<button onclick="f2()">function 2</button>

</body></html>
```

자바스크립트 연산자

산술 연산자 (y=5)

연산자	설명	예제	X 결과	Y 값
+	더하기	x=y+2;	7	5
-	빼기	x=y-2;	3	5
*	곱하기	x=y*2;	10	5
/	나누기	x=y/2;	2.5	5
%	분할 나머지	x=y%2;	1	5
++	증가	x=++y;	6	6
		x=y++;	5	6
--	감소	x=--y;	4	4
		x=y--;	5	4

연산자	예제	같은 의미	결과
=	x=y		x=5
+=	x+=y	x=x+y	x=15
-=	x-=y	x=x-y	x=5

연산자	예제	같은 의미	결과
*=	x*=y	x=x*y	x=50
/=	x/=y	x=x/y	x=2
%=	x%=y	x=x%y	x=0

자바스크립트 연산자

문자열에 사용되는 + 연산자

+연산자는 문자열 변수를 합치는데 사용 될 수도 있다

```
txt1="What a very ";
txt2="nice day";
txt3=txt1+txt2
```

What a nice day

```
txt1="What a very";
txt2="nice day";
txt3=txt1+" "+txt2
```

What a nice day

문자열 및 번호 추가

```
x =5+5
y ="5"+5
z ="Hello"+5
```

10

55

Hello5

자바스크립트 연산자

비교 연산자 (x=5)

연산자	설명	비교	반환값
==	동일합니다	x==8	*false*
		x==5	*true*
===	정확히 같은지 (값 타입)	x==="5"	*false*
		x===5	*true*
!=	같지 않다	x!=8	*true*
!==	같지 않다 (값 타입)	x!=="5"	*true*
		x!==5	*false*
>	보다 큰	x>8	*false*
<	보다 작은	x<8	*true*
>=	보다 크거나 같은	x>=8	*false*
<=	보다 작거나 같은	x<=8	*true*

자바스크립트 연산자

논리 연산자 (x=6, y=3)

연산자	설명	결과 예제
&&	그리고	(x < 10 && y > 1) is true
||	또는	(x==5 || y==5) is false
!	아닌	!(x==y) is true

비교 연산자 예제

```
<script type="text/javascript">
function checkAge(){
        ???
}
</script>
<body>
몇 살 입니까???<input type="text" id="age">
<input type="button" onclick="checkAge()" value="보내기">

</body>
```

자바스크립트 조건문

다른 조건에 따라 서로 다른 작업을 수행하는데 사용

코드를 작성 할 때, 결정에 따른 다른 작업을 수행 할 때 조건문을 사용

- if statement
  - Execute some code only if a specified condition is true
- if...else statement
  - Execute some code if the condition is true and
    another code if the condition is false
- if...else if....else statement
  - use this statement to select one of many blocks of code to be executed
- switch statement
  - use this statement to select one of many blocks of code to be executed

Syntax (If Statement)

```
if (condition)
  {
    code to be executed if condition is true
  }
```

자바스크립트 조건문

Syntax (if ... else Statement)

```
if (condition)
{
  code to be executed if condition is true
}
else
{
  code to be executed if condition is not true
}
```

```
if (time<20)
{
  x = “Good day”
}
else
{
  x = “Good evening”
}
```

자바스크립트 조건문

Syntax (if ... else if ... else Statement)

```
if (condition1)
{
  code to be executed if condition1 is true
}
else if (condition2)
{
  code to be executed if condition2 is true
}
else
{
  code to be executed if neither condition1 nor condition2 is true
}
```

자바스크립트 조건문

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function myFunction(){
  var x="";
  var time=new Date().getHours();
  if (time<12){                                  x="아침 인사"; }
  else if (time>12 && time <18){    x="점심 인사"; }
  else {                                         x= " 저녁과 새벽"; }
  document.getElementById("demo").innerHTML=x;
}
</script>
</head>
<body>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
</body>
</html>
```

자바스크립트 조건문

다른 조건에 따른 동작을 수행하기 위해 사용

Syntax

```
switch(n)
{
  case 1:
    execute code block 1
    break;
  case 2:
    execute code block 2
    break;
  default:
    code to be executed if n is different
    from case 1 and 2
}
```

자바스크립트 조건문

```
<!DOCTYPE html>
<html>
<body>
<button onclick="myFunction()">Try
it</button>
<p id="demo"></p>
<script>
function myFunction()
{
 var x;
 var d=new Date().getDay();
 switch (d)
 {
  case 0:
   x="Today it's Sunday";
   break;
  case 1:
   x="Today it's Monday";
   break;
  case 2:
   x="Today it's Tuesday";
   break;
  case 3:
   x="Today it's Wednesday";
   break;
  case 4:
   x="Today it's Thursday";
   break;
  case 5:
   x="Today it's Friday";
   break;
  case 6:
   x="Today it's Saturday";
   break;
 }

document.getElementById("demo").innerH
TML=x;
}
</script>
</body>
</html>
```

자바스크립트 반복문

루프는 코드를 여러 번 실행 할 수 있다

다른 값으로 동일한 코드를 실행할 때 편리

(A)

```
document.write(cars[0] + "<br>");
document.write(cars[1] + "<br>");
document.write(cars[2] + "<br>");
document.write(cars[3] + "<br>");
document.write(cars[4] + "<br>");
document.write(cars[5] + "<br>");
```

(B)

```
for (var i=0;i<cars.length;i++)
{
  document.write(cars[i] + "<br>");
}
```

자바스크립트 반복문

Syntax

```
for (statement 1; statement 2; statement 3)
{
  the code block to be executed
}
```

Statement 1은 루프의 시작점을 설정
Statement 2는 실행하기 위한 조건을 정의
Statement 3은 매 실행마다의 간격

```
for (var i=0; i<5; i++)
{
  x=x + "The number is " + i + "<br>";
}
```

i가 0~4까지 하나씩 증가하면서 총 5번 실행

자바스크립트 반복문

```
<!DOCTYPE html>
<html> <body>

<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
<script>
function myFunction()
{
  var x="";
  for (var i=0; i<5; i++)
  {
    x=x + "The number is " + i + "<br>";
  }
  document.getElementById("demo").innerHTML=x;
}
</script>
...
```

자바스크립트 반복문

```
<!DOCTYPE html>
<html>
<body>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>

<script>
function myFunction()
{
  var x;
  var txt="";
  var person={fname:"John",lname:"Doe",age:25};

  for (x in person)
  {
    txt=txt + person[x];
  }
  document.getElementById("demo").innerHTML=txt;
}
</script>
</body></html>
```

자바스크립트 반복문

지정된 조건이 “참”인 경우에 대해 반복 실행

Syntax

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

```
var i = 0;
while (i<5)
{
  x=x + "The number is " + i + "<br>";
  i++;
}
```

자바스크립트 반복문

10번 반복하는 While 문 예제

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function myFunction(){
  var x="",i=0;
  while (i<10){
    x=x + "The number is " + i + "<br>";
    i++;
  }
  document.getElementById("demo").innerHTML=x;
}
</script>
</head>
<body>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
</body>
</html>
```

자바스크립트 반복문

무조건 한번은 실행이 되야 하는 반복 코드에 사용
한번 실행이 된 후 그 뒤에 조건을 검사하여 “참 " 일 경우 계속 실행

Syntax(Do/While Loop)

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

```
var i = 0;
do
{
  x=x + "The number is " + i + "<br>";
  i++;
}
while (i<5);
```

자바스크립트 반복문

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function myFunction(){
 var x="",i=0;
 do
 {
  x=x + "The number is " + i + "<br>";
  i++;
 }
 while (i<5)

 document.getElementById("demo").innerHTML=x
}
</script>
</head>
<body>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
</body>
</html>
```

자바스크립트 Break/Continue

break 문 : 반복문을 중단 하는 구문

continue 문 : 반복문 중 아래 구문을 실행 시키지 않고 다음 값으로 넘어감

```
<!DOCTYPE html>
<html><head>
<script type="text/javascript">
function myFunction(){
  var x="",i = 0;
  for (i=0;i<10;i++)
  {
  if (i==3)
    break;
  x=x + "The number is " + i + "<br>";
  }
  document.getElementById("demo").innerHTML=x;
}
</script></head><body>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
</body>
</html>
```

자바스크립트 Break/Continue

break 문 : 반복문을 중단 하는 구문

continue 문 : 반복문 중 아래 구문을 실행 시키지 않고 다음 값으로 넘어감

```
<!DOCTYPE html>
<html><head>
<script type="text/javascript">
function myFunction(){
  var x="",i = 0;
  for (i=0;i<10;i++)
  {
  if (i==3)
    continue;
  x=x + "The number is " + i + "<br>";
  }
  document.getElementById("demo").innerHTML=x;
}
</script></head><body>
<button onclick="myFunction()">Try it</button>
<p id="demo"></p>
</body>
</html>
```

자바스크립트 배열

하나의 변수에 여러 값을 저장하는데 사용

배열 생성 및 값 할당

- 배열을 생성하는 3가지 방법
  - 기본적인 방법:

```
var myCars=["Saab","Volvo","BMW"];
```

  - 규칙적인 방법:

```
var myCars=new Array();
myCars[0]="Saab";
myCars[1]="Volvo";
myCars[2]="BMW";
```

  - 간결한 방법:

```
var myCars=new Array("Saab","Volvo","BMW");
```

자바스크립트 숫자

**자바스크립트는 오직 하나의 숫자 타입만을 가지며,
숫자들은 쓰여질 때 소수점이 있거나, 없이 쓸 수 있다.**

var pi=3.14;

→ 소수점 o

var x=34;

→ 소수점 x

var y=123e5; // 12300000

→ 지수 표기법(큰 숫자)

var z=123e-5; // 0.00123

→ 지수 표기법(작은 숫자)

**또한, 자바스크립트 숫자는 항상
64-bit 부동 소수점이다**

자바스크립트 숫자

숫자 정확도!?

- 정수(Integers)는 15자리까지 정확한 것으로 간주되고 있음

  var x = 999999999999999; -> 실질적으로 999999999999999 로 x가 나올 것

  But

  var x = 9999999999999999; -> 10000000000000000 으로 나올 것임

- 소수의 최대 보여지는 수는 17자리 수,

  (하지만 부동 소수점 연산은 항상 100% 정확하지 않음)

  var x = 0.2 + 0.1;

  X = 0.30000000000000004

  해결 방법!! >>>>>>> 곱셈과 나눗셈을 이용

  ex) var x = (0.2*10 + 0.1 * 10) / 10

자바스크립트 배열

진수(Hexadecimal) & 무한대(Infinity)

- 자바스크립트는 0x로 시작하는 경우 16진수로 해석

var x = 0xFF -> 255

> **!** **그러므로 숫자를 쓸 때 앞에 0을 사용하지 말 것.**
> **간혹 진수로 해석 하는 경우가 있음**

- 기본적으로 10진수를 보여주지만 toString() 메소드를 통해 16(hex), 8(octal), 2(binary)로 표현 가능

var x = 128; > x.toString(16) // 80, x.toString(8) // 200, x.toString(2) // 10000000

- Infinity는 산술 계산시 자바스크립트가 표현 할 수 있는 최대의 숫자가 넘을 경우 출력되는 값 또는 어떤 값을 0으로 나눌 경우

NaN (Not a Number)

- 예약어로 값이 숫자가 아닌 것을 표현 할 때 사용

자바스크립트 배열

숫자는 숫자이거나 객체일 수 있다!?

var x = 123; **VS** var x = new Number(123);

숫자 객체

! 하지만 사용 방법이 다를 뿐 모두 객체로 인식
그렇기 때문에 new를 이용해서 생성하는 것은 오히려 불필요한 작성

Properties:
- MAX VALUE
- MIN VALUE
- NEGATIVE INFINITY
- POSITIVE INFINITY
- NaN
- prototype
- constructor

Methods:
- toExponential()
- toFixed()
- toPrecision()
- toString()
- valueOf()

자바스크립트 문자열

자바스크립트 문자열은 텍스트를 저장하고 조작하는데 사용

문자열 사용법 / 기능

- **단일 (') 또는 이중 따옴표(")를 사용**

```
var carname="Volvo XC60";
var carname='Volvo XC60';
```

- **문자열의 문자 위치를 접근 할 수 있음 (0부터 시작)**

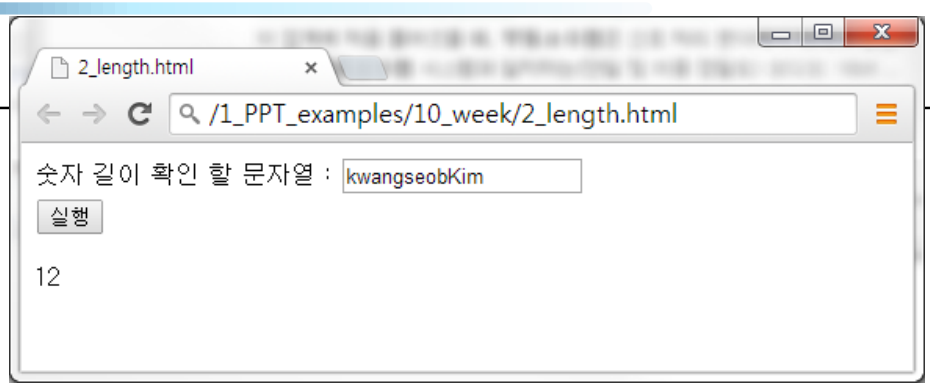
```
var character=carname[7];      // if character="abcdefghi";  result 'i'
```

- **문자열의 길이를 확인 할 수 있음**

```
var txt = "Hello World!";                    txt.length -> 12
```

자바스크립트 문자열

문자열 길이 예제




```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function getLength(){
        var txt = document.getElementById("txt").value;
        var text_context = document.getElementById("result");
        text_context.innerHTML = txt.length;
}

</script>
</head>
<body>
        숫자 길이 확인 할 문자열 : <input type="text" id="txt"><br>
        <input type="button" onclick="getLength()" value="실행">
        <p id="result"> </p>
</body>
</html>
```

자바스크립트 문자열

문자열 사용법 / 기능(con't)

- **문자열 또는 문자 찾기**

  indexOf() 메서드를 사용하여 첫 번째 발견된 위치를 반환

```
var str = "Hello world, welcome to the universe.";
var n = str.indexOf("welcome");
```

- **대소문자 변환**

  toUpperCase() / toLowerCase()를 통한 대소문자 변환 가능 (영어)

```
var txt  = "Hello World!";     // String
var txt1 = txt.toUpperCase(); // txt1 is txt converted to upper
var txt2 = txt.toLowerCase(); // txt2 is txt converted to lower
```

자바스크립트 문자열

문자열 찾기 예제

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function searchString(){
var text_context = document.getElementById("search").innerHTML;
var txt = document.getElementById("txt").value;
var result = document.getElementById("result");
result.innerHTML = text_context.indexOf(txt);
}
</script>
</head>
<body>
<p id="search">안녕하세요, 다들 즐거운 하루 되셨나요??? </p>
찾을 할 문자열 : <input type="text" id="txt"><br>
<input type="button" onclick="searchString()" value="실행">
<p id="result"> </p>
</body>
</html>
```

자바스크립트 문자열

문자열 사용법 / 기능(con't)

- **일치하는 내용 검색**

  Match()를 통해 문자열에서 일치하는 콘텐츠를 검색 할 수 있음

```
var str="Hello world!";
document.write(str.match("world") + "<br>");
document.write(str.match("World") + "<br>");
document.write(str.match("world!"));
```

- **문자열 또는 문자 대체**

  Replace()를 통해 일치하는 문자열 또는 문자를 설정 값으로 대체

```
str = "Please visit Microsoft!"
var n = str.replace("Microsoft","W3Schools");
```

자바스크립트 문자열

문자열 사용법 / 기능(con't)

- **문자열(String)을 배열(Array)로 변환**

  split()를 통해 일정한 패턴을 통해 자동으로 배열로 저장

```
txt="a,b,c,d,e"   // String
txt.split(",");   // Split on commas
txt.split(" ");   // Split on spaces
txt.split("|");   // Split on pipe
```

이 외에도 개발자가 직접 정의 할 수 있음

Properties:

- length
- prototype
- constructor

Methods:

- charAt()
- charCodeAt()
- concat()
- fromCharCode()
- indexOf()
- lastIndexOf()
- match()
- replace()
- search()
- slice()
- split()
- substr()
- substring()
- toLowerCase()
- toUpperCase()
- valueOf()

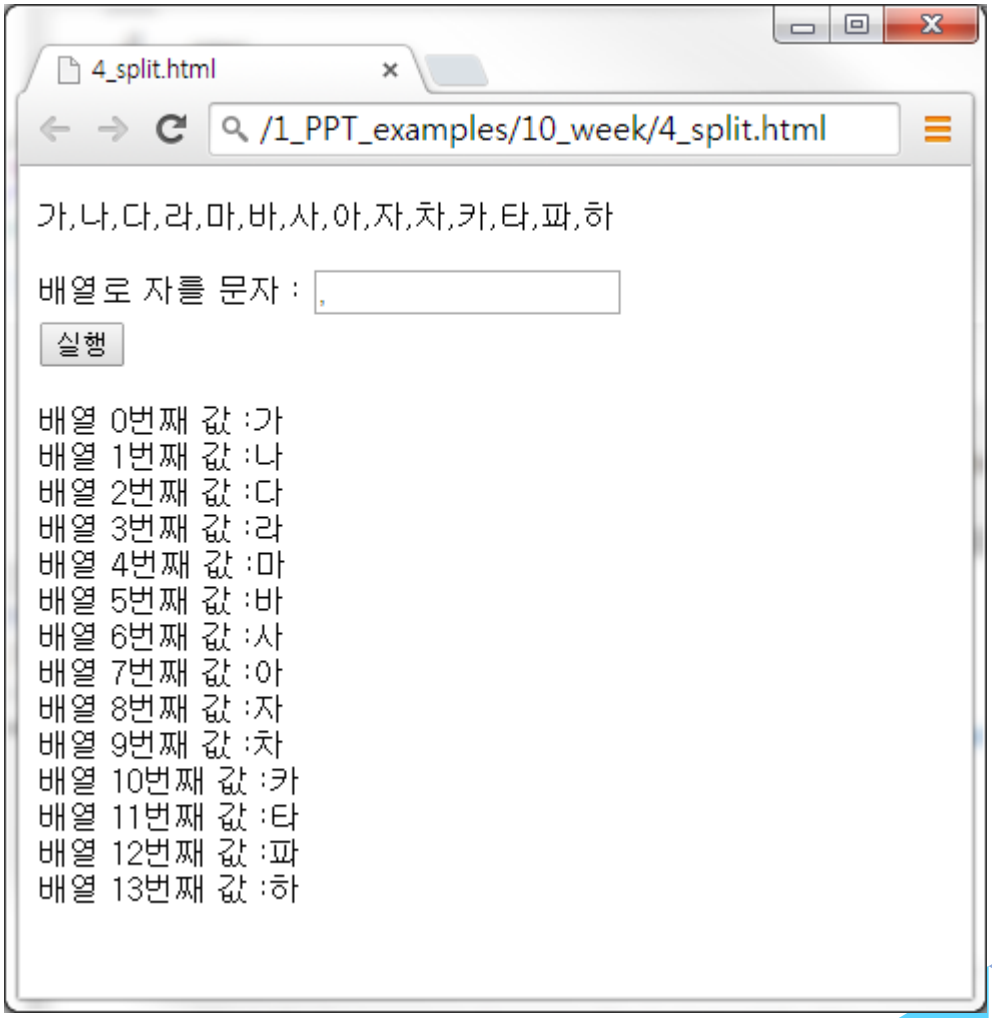
자바스크립트 문자열

문자열 자르기 예제

```
…
<script type="text/javascript">
function searchString(){
        var text_context = document.getElementById("search").innerHTML;
        var txt = document.getElementById("txt").value;
        var result = document.getElementById("result");
        var re = text_context.split(txt);                    var tmp="";
        for(var i=0; i<re.length; i++){
                tmp += "배열 "+i+"번째 값 :"+re[i]+'<br>';
        }
        result.innerHTML = tmp;
}
</script>
</head>
<body>
<p id="search">가,나,다,라,마,바,사,아,자,차,카,타,파,하</p>
배열로 자를 문자 : <input type="text" id="txt"><br>
<input type="button" onclick="searchString()" value="실행">
<p id="result"> </p>
…
```

자바스크립트 문자열

문자열 자르기 예제



자바스크립트 문자열

Date 객체는 날짜 및 시간 작업에 사용

Date 객체 생성

- Date() 생성자를 통해 생성
- 총 4개의 초기 생성 하는 방법 존재:

```
new Date() // current date and time
new Date(milliseconds) //milliseconds since 1970/01/01
new Date(dateString)
new Date(year, month, day, hours, minutes, seconds, milliseconds)
```

로컬 시간이나 UTC(범용 또는 GMT)시간을 사용. 년, 월, 일, 시, 분, 초, 밀리 초 단위의 객체를 얻음

```
var today = new Date()
var d1 = new Date("October 13, 1975 11:13:00")
var d2 = new Date(79,5,24)
var d3 = new Date(79,5,24,11,33,0)
```

자바스크립트 Date

Date 설정

- 시간 또는 날짜를 조작 할 때 사용
- 특정 날짜(2010년 1월 14일)로 변경 하는 예제 :

```
var myDate=new Date();
myDate.setFullYear(2010,0,14);
```

- 상대적 조작

```
var myDate=new Date();
myDate.setDate(myDate.getDate()+5);
```

자바스크립트 Date

날짜 비교

- 비교 연산자를 통해 두 날짜를 비교 할 수 있음

```
var x = new Date();
x.setFullYear(2100,0,14);
var today = new Date();

if (x>today)
{
  alert("Today is before 14th January 2100");
}
else
{
  alert("Today is after 14th January 2100");
}
```

자바스크립트 Boolean

Boolean 값이 아닌 Boolean 값을 변환 할 때 사용

Boolean 객체 생성

- “true” 또는 “false”

```
var myBoolean=new Boolean();
```

자바스크립트 Math

Math 객체는 수학 작업을 수행 할 때 사용
여러 가지 수학적 상수와 메서드가 포함

Syntax for using properties/methods of Math:

```
var x=Math.PI;
var y=Math.sqrt(16);
```

수학 상수

Math.E
Math.PI
Math.SQRT2
Math.SQRT1_2
Math.LN2
Math.LN10
Math.LOG2E
Math.LOG10E

수학 메소드

Math.round()
Math.random()
Math.max()
Math.min()

http://www.w3schools.com/jsref/jsref_obj_math.asp

자바스크립트 Math

Math 예제 (round)

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function myFunction(){
        document.getElementById("result").innerHTML=Math.round(2.5);
}
</script>
</head>
<body>
<input type="button" onclick="myFunction()" value="실행">
<p id="result"> </p>
</body>
</html>
```