# Part III
# 자바스크립트
## 1. JavaScript Basic

목차

# 1. 개요 - 역사

- 네스케이프의 라이브 커넥트 라는 서버측 기술에서 연동이 필요한 클라리언트측의 스크립트 언어였던 라이브스크립트에서 부터 시작
- 네스케이프사 가 **SUN**사와의 제휴로 이름이 자바스트립트로 변경하고 **HTML** 보완 언어로 발표 **(1995**년**)**
- **MS**의 독자적 **J**스크립트 발표등 통일되지 않은 스펙과 호환성 문제 발생
- 네스케이프사가 **1996**년 자바스크립트 스펙을 **ECMA**국제회의에 제출 **,** 첫 버전 **ECMA-262** 발표**.**
- 지금은 **ECMA-262** 세 번째 버전이 나와 있으며 이를 기반으로 한 자바 스크립트 **1.6**이 대부분 브라우저에서 지원되고 있다**.**

# 1. 개요 – Hello World

## ❑ [실습1] Hello World

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
                      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="ko">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Hello World</title>
<script type="text/javascript">
   document.write( "Hello World" );
</script>
</head>
<body>
</body>
</html>
```

**document.write( "Hello World" ) 대신 alert( "Hello World" ) 로 바꾸고 브라우저에서 확인 보세요.**

# 1. 개요 – 실행과 특징

- 브라우저 내에 내장된 자바스크립트 실행엔진 (해석기)를 통해 실행되어지고 브라우저 화면에 반영된다.
- 객체지향이라기 보다는 **객체 기반 스크립트 언어**
- 객체를 생성하느 클래스 개념은 지원하고 있지 않지만 **prototype** 이라는 객체 생성 지원 개념이 있다.

## ❑ HTML 문서 내에서 자바스크립트 사용하기

```
<html>
<body>
...
<script
type="text/javascript">
    자바스크립트  소스코드
</script>
...
</body>
</html>
```

```
<html>
<head>
<script
type="text/javascript">
    자바스크립트  소스코드
</script>
</head>
<body>
...
</body>
</html>
```

보통 헤드안에 삽입되어 실행 되나 본문 어디에도 삽입되어 실행될 수 있다

# 1. 개요 – 실행과 특징

## ❑ HTML 문서 내에서 자바스크립트 사용하기

```
<html>
<head>
<script type="text/javascript" src="hello.js"></script>
</head>
<body>
...
</body>
</html>
```

– 외부 파일를 불러서 실행할 수 있다 ( 보통 이방식으로 자바스크립트를 실행하고 관리한다 )

## ❑ [ 실습2 ]

**실습1를 외부파일에서 불러와 실행 시키는 방식으로 수정하고 확인해 보세요.**

# 2. 기본문법 – 변수 와 데이터 타입

## ❑ 기본 데이터 타입

Number, String, Boolean

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

var number = 5
var anotherNumber = new Number(5)
var pi = 3.14
var string = "Hello, I'm a string!"
var anotherString = new String( "Hello, I'm a string!" );
var used = false;



alert( typeof number)
alert( typeof anotherNumber )
alert( typeof pi )
alert( typeof string )
alert( typeof anotherString )
alert( typeof used )

</script>
```

# 2. 기본문법 – 변수 와 데이터 타입

## ❑ null 과 undefined

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

var myVar, myVar2 = null;
alert( myVar + ", " + myVar2 );
alert( myVar == myVar2 );
alert( myVar === myVar2 );

</script>
```

# 2. 기본문법 – 변수 와 데이터 타입

## ❑ 변수 이름 규칙

변수의 이름은 알파벳(대문자 A ~ Z, 소문자 a ~ z), 밑줄(_)이나 달라($)로 시작될 수 있으며, 다음에는 알파벳, 밑줄, 달라 기호에 추가로 숫자(0 ~ 9)까지 사용할 수 있다.

## ❑ 선언없이 대입시에 변수 타입이 결정된다.

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

v = "This is Test";
alert( typeof v );

v = 20;
alert( typeof v );

</script>
```

# 2. 기본문법 – 변수 와 데이터 타입

❑ 변수 범위

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

function func(){
    v1 = "hello javascript";
}

func();
alert( v1 );

</script>
```

❑ **v1 앞에 var를 붙히게 되면?**.

❑ **var와 함께 변수에 값을 대입하는 것이 좋다.**

# 2. 기본문법 - 구문

❑ 세미콜론을 반드시 구문끝에 붙혀야 하는 것은 아니다.

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

var str = "This is Test"
document.write( str )

</script>
```

❑ 한 줄로 붙혀 쓸 수는 없다.

```
<script type="text/javascript">
var str = "This is Test" document.write( str )
</script>
```

❑ 엔진이 한줄 씩 일거나 ;(세미콜론) 으로 분리된 구문을 읽어 해석해 나가기때문에

문장사이에는 \n(개행) 이나 ;(세미콜론) 으로 분리

# 2. 기본문법 - 구문

- 자바와 마찬가지로 **CamelI** 표기법이 기본

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

//// CamelCase is the norm
if(fooBar == bazBat) {}

</script>
```

- 객체 속성, 메서드에 접근 할 때에는 . 으로 접근한다.

```
<script type="text/javascript">
someObject.someMethod();
</script>
```

- 주석은 한줄 주석 // , 여러줄 주석 /* */ 모두 가능하다.

# 2. 기본문법 – 조건문

❑ If문, If~else, if~else if 등의 조건문은 다른 언어들과 다르지 않다.

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

var something = false;
if( something ) doSomething();

</script>
```

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

if(something == foobar) {
   alert("equals foobar!");
} else if(something == bazbat) {
   alert("equals bazbat!");
} else {
   alert("equals neither!");
}

</script>
```

# 2. 기본문법 – 조건문

❑ **Switch 문도 사용이 가능하다.**

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

switch( something ) {
          case "foobar":
                    // if something == "foobar"
                    alert( "foobar!" );
                    break
          case "barfoo":
                    // if something == "barfoo"
                    alert("Barfoo!");
                    break
          case "fallthru":
                    // without a break, results will cascade
                    // the result? [alert] Falling through...
                    //             [alert] fallen through
                    alert("Falling through...");
          case "fellthru":
                    alert("fallen through.");
                    break
          default:
                    // if there is no case "*" match, execute this code
                    alert("Case not found... here's a default")
}
</script>
```

# 2. 기본문법 – 반복문

- for, while, do~while 문도 다른 언어들과  틀리지 않다.

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

for(var i = 0; i < 3; i++) {
   document.write( I + "<br>" );
}

</script>
```

- [ 실습3 ] 1단에서 9단까지 구구단  출력해 보세요.
- [ 실습4 ]

*****

****

***

**

*

# 3. 객체 – 정의 / 생성

- 정보를 관리하기 위해 의미를 부여하고 분류하는 논리적 단위
- 클래스의 인스탄스(instance)  -> 구체화, 실체화
- 객체는 속성(attribute) 과 함수(Function)을 가지고 있다.
- [실습 5] 객체의 생성

```
var employee1 = new Object();
employee1.name = "홍길동";
employee1.title = "과장";
employee1.showInfo = function() {
  document.write( "이름 : " + this.name );
  document.write( "<br>" );
  document.write( "직책 : " + this.title );
}


employee1.showInfo();
```

# 3. 객체 – 정의 / 생성

- JSON ( JavaScript Object Notation )
- 자바스크립트에 객체생성을 위한 표기하는 방법
- 어떤 객체던지 표기할 수 있고 바로 생성가능
- [실습 6 - 1] 객체의 생성

```
var employee1 = {};
employee1.name = "홍길동";
employee1.title = "과장";
employee1.showInfo = function() {
  document.write( "이름 : " + this.name );
  document.write( "<br>" );
  document.write( "직책 : " + this.title );
}


employee1.showInfo();
```

## ❑ [실습 6 - 2] 객체의 생성(JSON)

```
var employee1 = {
    name: "홍길동",
    title: "과장",
    showInfo: function() {
        document.write( "이름 : " + this.name );
        document.write( "<br>" );
        document.write( "직책 : " + this.title );
    }
}


employee1.showInfo();
alert( employee1.name + " " + employee1.title );
```

## ❑ [실습 6 - 3] 객체의 생성(JSON) – 코마 사용에 조심

```
var foo = {
    name:"bar",
    nick:"buzz"
    aNumber:5,
    doStuff:function() {
        alert( "I'm" + this.name );
    },
}
```

**어디에서 에러가 있는 지 찾고 바르게 수정 해 보세요.**

# 3. 객체 – 정의 / 생성

## ❑ 생성자 사용

```
var Foo = function() {
  this.name = "bar";
  this.nick = "buzz";
  this.aNumber = 5;
  this.doFoo = function() {
    alert( "I'm " + this.name );
  };
}

var foo = new Foo();
foo.doFoo();
```

## ❑ [ 실습 6 – 4 ]

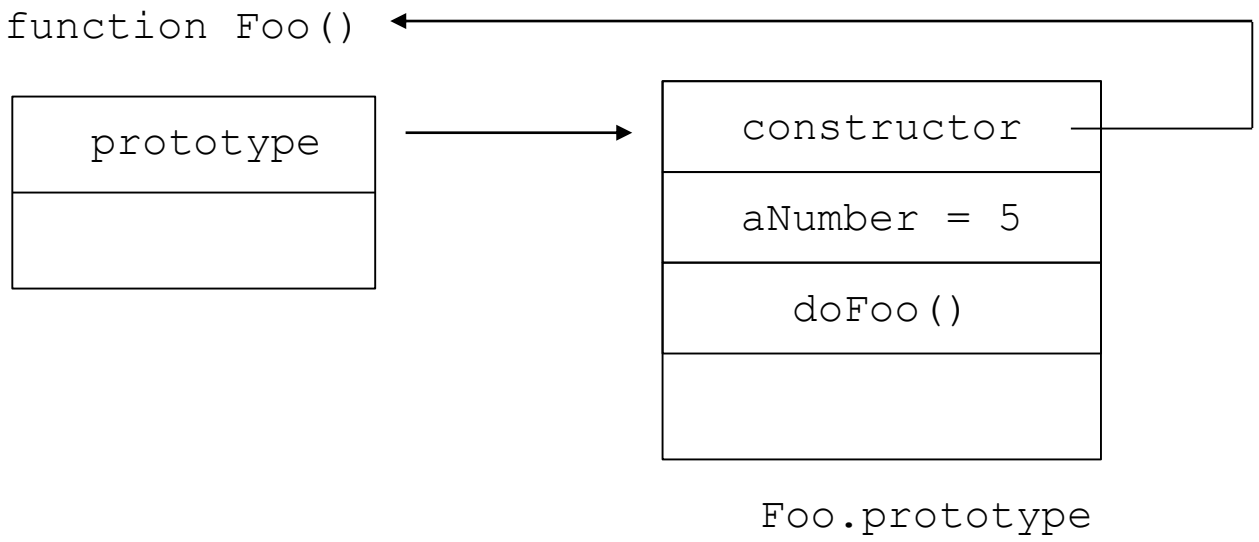
**Foo 생성자에서 name, nick 를 초기화 시킬수 있도록 수정해 보세요.**

# 3. 객체 – 정의 / 생성

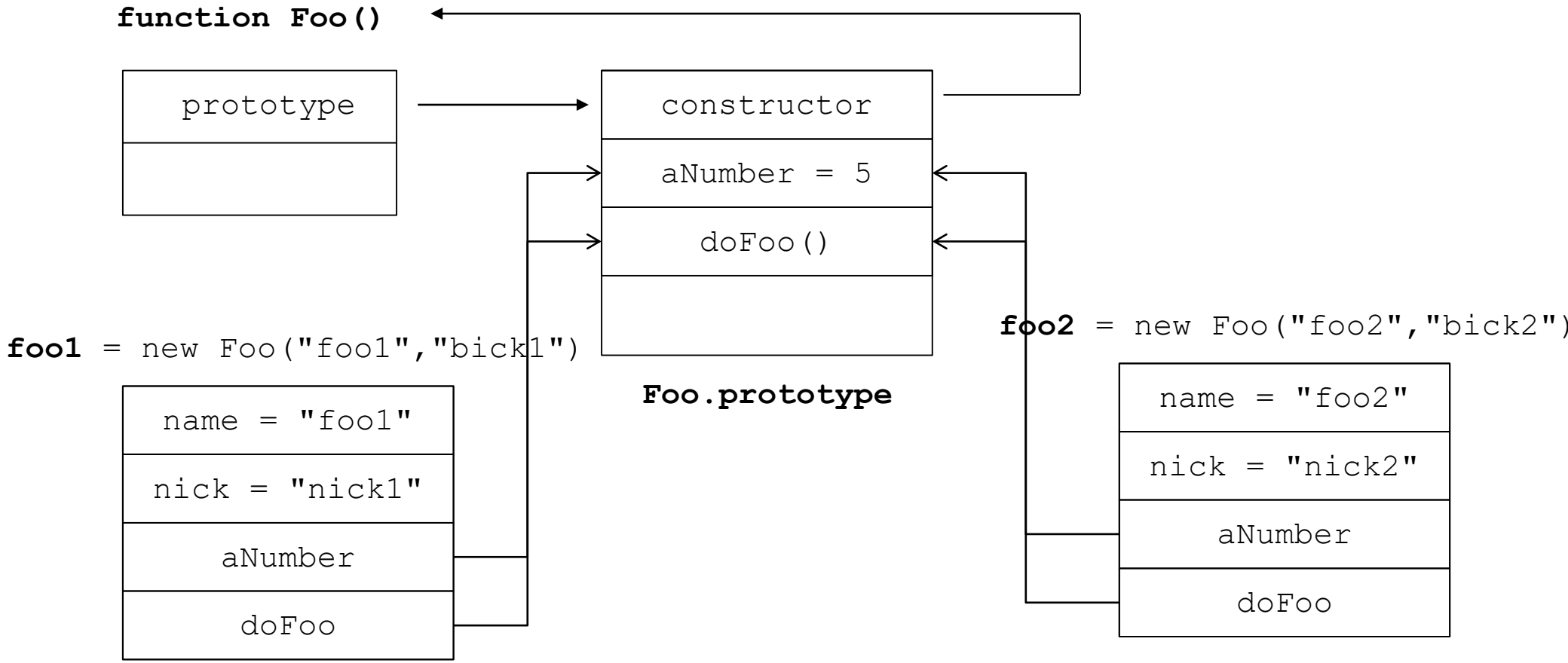
❑ **prototype 기반의 상속을 통해 객체지향 구현 ( 속성, 함수 공유 )**

```
var Foo = function( name, nick ) {
  this.name = name;
  this.nick = nick;
}

Foo.prototype.aNumber = 5;
Foo.prototype.doFoo = function() {
  alert( "I'm " + this.name );
};

var foo1 = new Foo( "foo1", "nick1" );
foo1.doFoo();

var foo2 = new Foo( "foo2", "nick2" );
foo2.doFoo();
```

## ❑ prototype 기반의 상속



## ❑ prototype 기반의 상속



❑ [ 실습예제 6 - 5 ]

다음 속성과 함수(메서드)를 가지고 있는 Class 개념을 function과 프로토타입을 사용하여 구현해 보세요.

1) 생성자 함수 Rectangle (클래스 Rectangle)

2) LeftTop 좌표 x1, y1

3) RightBottom 좌표 x2, y2

4) backgroundColor ( #fff )

5) show 함수 : 화면에 사각형을 표시

❑ [ 실습예제 6 - 5 ]

다음 속성과 함수(메서드)를 가지고 있는 Class 개념을 function과 프로토타입을 사용하여 구현해 보세요.

1) 생성자 함수 Rectangle ( x1, y1, x2, y2, color )

2) LeftTop 좌표 x1, y1

3) RightBottom 좌표 x2, y2

4) color ( #000 )

5) show 함수 : 화면에 사각형을 표시

"[width:100, height:200, color:#000] 인 사각형을 그렸습니다."

❑ **[ 실습예제 6 - 6 ]**

**1) 실습예제 6-5에 CSS를 사용해서 실제로 그려보기**

**2) position: absolute 개념 익히기.**

**3) 동적으로 객체에 속성 삽입. (border)**

# 3. 객체 – 내장 객체 Array

❑ 배열생성

**1) var a = new Array(10);**

**2) var a = new Array();**

**3) var a = new Array( 1, "ABC", true) ;**

❑ 배열 생성 (리터럴)

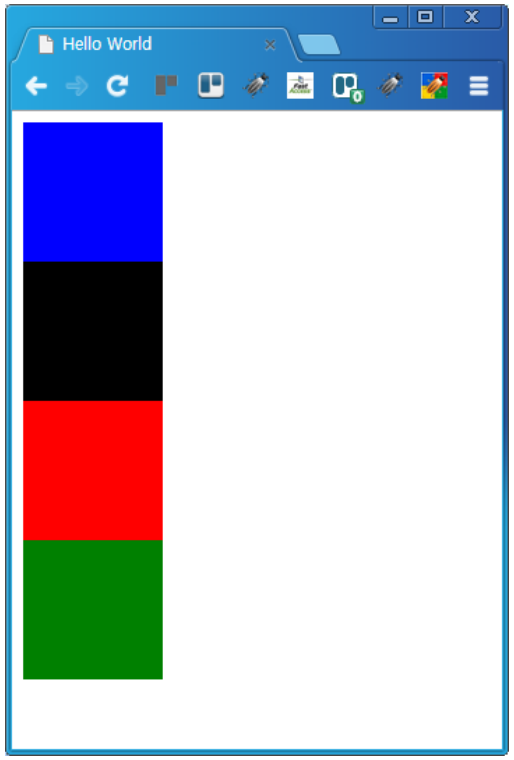
**var a = [];**

# 3. 객체 – 내장 객체 Array

❑ length 속성 : 배열의 size을 담고 있눈 속성

❑ [ 실습예제 6-7 ]

배열을 사용해서 다음과 같은 결과가 나오도록 자바스크립트 코드를 작성해 보세요.




[힌트] 색상 배열 선언에 다음 코그를 사용합니다.

```
var colors = ["blue", "black", "red", "green"];
```

## ❑ 배열과 객체의 관계

속성 접근시 실습예제와 같이 배열처럼 접근할 수 있다

**[ 실습예제 6-8 ]**

```
var employee1 = {
    name: "홍길동",
    title: "과장"
}

alert( employee1["name"] + " " + employee1["title"] );
```

# 3. 객체 – 내장 객체 Array

## ❑ 주요함수

| 종류 | 설명 |
|---|---|
| concat(array1,...) | 배열을 하나로 합친다. |
| join(str) | 배열 전체를 str구분자를 가지는 하나의 문자열로 만든다. |
| pop() | 배열의 맨 마지막 변수를 삭제한다. |
| push(item1,..) | 배열의 마지막에 변수들을 추가한다. |
| reverse() | 배열의 순서를 뒤집는다. |
| shift() | 배열의 맨 처음 값을 삭제한다. |
| slice() | 배열의 일부분만을 추출하여 새로운 배열을 만든다. |
| sort(func) | 배열을 정렬한다. |

## ❑ 주요 함수 예제

```
var hege = [ "Cecilie", "Lone" ];
var stale = [ "Emil", "Tobias", "Linus" ];
var children = hege.concat( stale );

console.log( children );
```

```
var fruits = ["Banana", "Orange", "Apple", "Mango"];
var energy = fruits.join();

console.log( energy );
```

## ❑ 주요 함수 예제

```
var fruits = [ "Banana", "Orange", "Apple", "Mango" ];

fruits.push( "Kiwi" );
console.log( fruits );

console.log( fruits.pop() );
console.log( fruits.pop() );

console.log( fruits );
```

## ❑ 주요 함수 예제

```
var fruits = [ "Banana", "Orange", "Apple", "Mango" ];

fruits.reverse();
console.log( fruits );

fruits.shift();
console.log( fruits );

var citrus = fruits.slice(1, 3);
console.log( citrus );

fruits.sort();
console.log( fruits );
```

❑ 배열처럼 다룰 수 있다.

[예제 6- 9 ]

문자열을 배열처럼 접근해서 사용하는 예제 입니다.

IE에서 잘 작동하는 지 확인해 보고 잘 작동하도록 수정해 보세요.

```
var str = "Hello, I'm a string!";
console.log( "문자열 길이는 : " + str.length );

for(var i = 0; i < str.length; i++) {
 console.log( str[i] );
}
```

## ❑ 합치기 와 자동 형 변환

**[예제 6- 10 ]**

문자열 합치기 예와 그 때, 자동 변환되는 경우들을 확인해 보세요.

```
console.log( "첫 번째 문자열 " + " 두 번째 문자열" );

var str = "number " + 5;
console.log( str + " : " + typeof( str ) );

str = 5 + "5";
console.log( str + " : " + typeof( str ) );
```

## ❑ [ 예제 6- 11 ] 주요함수 사용 예 1

```
var str = "string1 string2 string3"

alert( str.length );

var start = str.indexOf( 'string2' );
alert( start );
alert( str.substr( start ) );


// 간단히,
alert( str.substr( str.indexOf( 'string2' ) ) );

// str은 변하지 않는다.
alert( str );
```

## ❑ [ 예제 6- 12 ] 주요함수 사용 예 2

```
var str = "string1 string2 string3"

// 배열로 분리한다.
var a = string.split(' ');


// 배열을 확인하는 코드를 직접 작성해 보세요.
```

## ❑ [ 예제 6- 13 ] Escaping HTML, URLs, etc.

```
//에러
"<h3>Here's a headline!</h3>".escape();

// escape 함수는 전역함수로 제공
var escaped = escape("<h3>Here's a headline!</h3>");
var unescaped = unescape(escaped);


// URL 엔코딩
var url = "http://mysite.com/?stuff=\"안 대혁!&bar=";
var encodedURL = encodeURI(url);
var decodedURL = decodeURI(encodedURL);
```

# 3. 객체 – 내장 객체 Date

❑ Date객체는 날짜와 시간을 다루는 객체이다.

❑ 기본 사용법

```
var d = new Date();  // 현재 시간
document.write( d );
```

❑ [예제 6-14]

다음 Date객체 생성자를 사용해 Date 객체를 생성하고 document.write 를 이용해 결과를 확인해 보세요.

1) Date( year, month, day )

2) Date( yyyy, mm, dd, hh, mi, ss)

3) Date( milliseconds )

# 3. 객체 – 내장 객체 Date

## ❑ 관련 함수

| 종류 | 설명 |
|---|---|
| getYear() / setYear() | 년도 |
| getMonth() / setMonth() | 월(0:1월, 1:2월, ....., 11:12월) |
| getDate() / setDate() | 일(1일 ~ 31일) |
| getDay() / setDay() | 요일(0:일요일, 1:월요일, ...,6:토요일) |
| getHours() / setHours() | 시간(0시 ~ 23시) |
| getMinutes() / setMinutes() | 분(0 ~ 59) |
| getSeconds() / setSeconds() | 초(0시 ~ 59) |
| getMilliseconds() / setMilliseconds() | 시간(0시 ~ 23시) |
| getHours() / setHours() | 시간(0시 ~ 23시) |

## ❑ [ 실습예제 6-15 ] Date 객체 함수 사용

```
var d = new Date(2013, 0, 28);  //2013 년 1월 28일
document.write(
  " 년도: " + (d.getYear() + 1900) + "<BR>" +
  " 월: " + (d.getMonth() + 1) + "<BR>" +
  " 일은: " + d.getDate() + "<BR>" +
  " 요일은: " + d.getDay() + "<BR>" +
  " 시는: " + d.getHours() + "<BR>" +
  " 분은: " + d.getMinutes() + "<BR>" +
  " 초는: " + d.getSeconds() + "<BR>" +
  " 밀리초: " + d.getMilliseconds() + "<HR>" );

d.setYear(2014);  // 2014년 세팅
document.write(d + "<HR>");

d.setMonth( 11 ); // 12월 세팅
document.write(d + "<HR>");
```

# 3. 객체 – 내장 객체 Function

- 함수도 객체로 간주
- 함수 생성 방식

1) var sum = new Function( “a”, “b”, “return a+ b” );

2)

```
function sum ( a, b ) {
        return a+b;
}
```

3)

```
var sum = function( a, b ) {
   return a+ b;
}
```

- 함수 argument
- [실습예제 6-16 ]

  함수의 argument는 값과 객체뿐만 아니라 함수도 될 수 있음을 다음 예제로 확인해 보세요.

```
function myFunction( arg1, arg2, arg3 ) {

  // 값
  alert("I have an argument! " + arg1);

  // 객체
  alert(arg2.bar);

  // 함수
  arg3();
}

myFunction( "foo", { bar: "baz" }, function(){ alert("Victory!")} );
```

❑ [실습예제 6-16 ]

함수의 argument로 함수가 넘어 갈 경우 함수의 본문이 길어 질 경우,

```
var f = function() {
  alert( "victory!" );
}

myFunction( "foo", { bar: "baz" }, f );
```

직접 예제에 적용해서 확인해 보세요.

❑ [실습예제 6-16 ]

함수의 argument는 함수내부에서 argument객체로 참조할 수 있다.

```
function myFunction() {

  // 값
  alert("I have an argument! " + arguments[0] );

  // 객체
  alert( arguments[1].bar );

  // 함수
  arguments[2]();
}
```

직접 예제에 적용해서 확인해 보세요.

# 4. 이벤트

- **HTML DOM**은  다음과 같은 **javascript** 가 이벤트에 반응할 수 있도록 하고 있다.
  - 사용자의 마우스 클릭
  - 웹 페이지의 로딩 완료 되었을 때
  - 이미지가 로딩 되었을 때
  - HTML element에 마우스가 움직이거나 오버되었을 때
  - Input 필드가 변경 되었을 때
  - HTML form 이 submit 될 때
  - 사용자의 key 누름
- 이벤트에 대한 반응 처리 (  **HTML Event  Attribute** 에  **javascript** 를 추가한다  )

**onclick = JavaScript**

## [ 실습예제 6 – 17 ]

```
<!DOCTYPE html>
<html>
<body>
<h1 onclick="this.innerHTML='Ooops!'">Click on this text!</h1>
</body>
</html>
```

- 이벤트를 처리할 수 있는 함수 ( Event Handler )로 처리할 수 있다.

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function changetext(id){
  id.innerHTML="Ooops!";
}
</script>
</head>
<body>
<h1 onclick="changetext(this)">Click on this text!</h1>
</body>
</html>
```

❑ HTML Element 에 이벤트를 매핑할 때는 이벤트속성(Event Attribute)을 사용

[ 실습예제 6–18 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
function displayDate(){
   document.getElementById("demo").innerHTML = Date();
}
</script>
</head>
<body>
<p>버튼을 클릭하면 <em>displayDate()</em> 함수가 실행 됩니다.</p>
<button onclick="displayDate()">Try it</button>
<p id="demo"></p>
</body>
</html>
```

[ 실습예제 **6–19** ]

[ 실습예제 **6-18** ] 에 이벤트 속성 추가해 보기

**1)** 마우스오버 : **onmouseover**

**onclick** 반응과 같도록 같은 이벤트 핸들러를 사용합니다.

**2)** 마우스 아웃 : **onmouseout**

날짜가 사라지게끔 이벤트 핸들러를 새로 추가합니다.

## ❑ 자바스크립트로 특정 Element 이벤트 매핑하기

### [ 실습예제 6 – 20 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
document.getElementById("myBtn").onclick = function(){
   displayDate()
};
function displayDate(){
   document.getElementById("demo").innerHTML = Date();
}
</script>
</head>
<body>
<p>버튼을 클릭하면 <em>displayDate()</em> 함수가 실행 됩니다.</p>
<button id="myBtn">Try it</button>
<p id="demo"></p>
</body>
</html>
```

**[ 실습예제 6–21 ]**

**[ 실습예제 6-19 ] 의 이벤트속성을 사용해 이벤트 핸들러와 연결했던 것을 자바스크립트를 사용해 이벤트 핸들러와 연결( 매핑) 해 보세요.**

# 4. 이벤트

- **onload** 는  사용자가  특정 페이지에 입장 했을 때 발생한다.
- 반대로 **onunload** 가 특정 페이지를 떠나면 발생한다.
- **onload**에서 페이지의 최초작업 초기화 작업을 할 수 있을 것이다.

## [ 실습예제 6 – 22 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
function checkCookies() {
  if( navigator.cookieEnabled==true ){
     alert("쿠키를 사용할 수 있습니다.")
  } else {
     alert("쿠키를 사용할 수 없습니다.")
  }
}
</script>
</head>
<body onload="checkCookies()">
<p>브라우저에서 쿠키 사용여부를 어떻게 설정했는 지 알수 있습니다.</p>
</body>
</html>
```

**[ 실습예제 6 – 23 ]**

**BrowserDetect 객체를 사용해서 페이지에 방문하는 사용자의 브라우저 종류, 버전 그리고 OS를 알아 내는 코드를 onload 이벤트 핸들러에 추가해 보세요.**

**[참고]**

**<javascript type="text/javascript" src="browser-detect.js"></script>**

# 4. 이벤트

❑ onchange 는 Input , select 값이 변하면 발생한다.

**[ 실습예제 6 – 24 ]**

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
function myFunction() {
    var x=document.getElementById("fname");
    x.value=x.value.toUpperCase();
}
</head>
<body>
아이디 입력: <input type="text" id="fname" onchange="myFunction()">
<p>INPUT에 포커스가 없어지면, 이전 값이 변했는 지 판단해서 항상 대문자로 만드는 예제입니다.</p>
</body>
</html>
```

# 4. 이벤트

❑ onchange 는 Input , select 값이 변하면 발생한다.

**[ 실습예제 6 – 24 ]**

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript">
function myFunction() {
    var x=document.getElementById("fname");
    x.value=x.value.toUpperCase();
}
</head>
<body>
아이디 입력: <input type="text" id="fname" onchange="myFunction()">
<p>INPUT에 포커스가 없어지면, 이전 값이 변했는 지 판단해서 항상 대문자로 만드는 예제입니다.</p>
</body>
</html>
```

## ❑ onmousedown, onmouseup 그리고 onclick

## [ 실습예제 6 – 25 ]

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script>
function mDown( obj ) {
  obj.style.backgroundColor="#1ec5e5";
  obj.innerHTML="버튼을 떼주세요.";
}
function mUp( obj ) {
  obj.style.backgroundColor="#D94A38";
  obj.innerHTML="클릭하세요!";
}
</script>
</head>
<body>
<div onmousedown="mDown(this)" onmouseup="mUp(this)" style="background-
color:#D94A38;width:90px;height:20px;padding:40px;">클릭하세요!</div>
</body>
</html>
```

[과제]

더 많은 이벤트 예제들은 다음 페이지에서 참고할 수 있습니다.

**HTML DOM Events**

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

이 문서에서

**Mouse Event,**

**Keyboard Event,**

**Form Event**

의 예제들을 확인해 보세요

# Part III
# 자바스크립트
## 1. jQuery

목차

# 1. 개요 - 소개

- 2006년 Mozila의 자바스크립트 에반젤리스 Jhon Resig에 의해 개발 / 공개
- 여러 자바스크립트 라이브러리 ( prototype.js, Mootool.js 등) 중에 가장 주목 받고 있다.
- jQuery로 코딩하면 자바스크립트 코드가 간결해 진다.
- 가볍다 ( 90KB)
- IE6.0 이상, Firefox2.0 이상, Safari 3 이상, Opera 9이상, Google Chrome등의 주요 브라우저를 지원하여 클로스브라우징을 가능케 한다.

# 1. 개요 – 사용준비

❑ 다운로드 ( http://jquery.com/download/ , 최신 버전 1.9.0 )

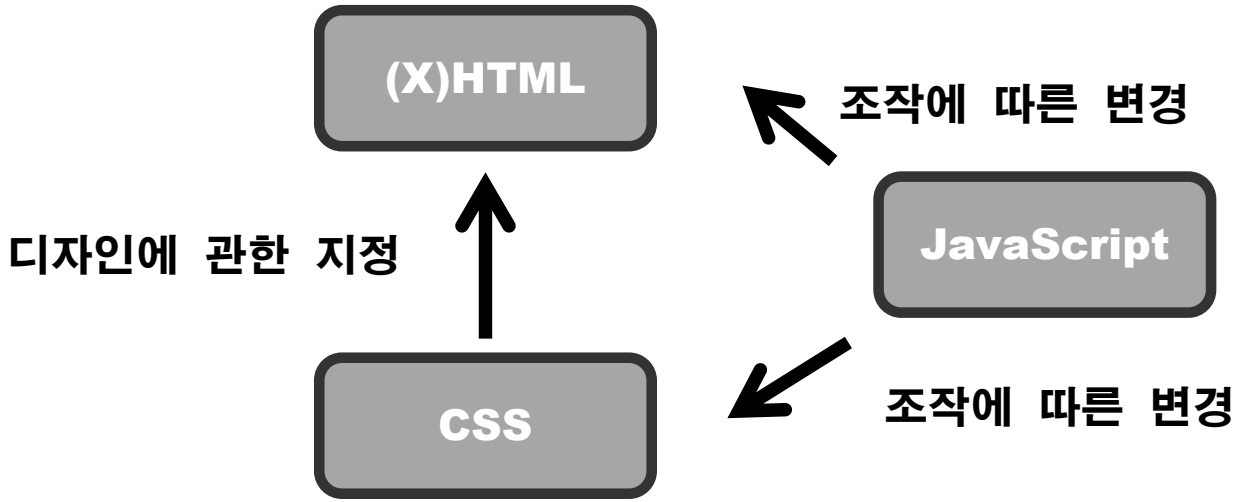
❑ 개발시에는 uncompressed 버전( jquery-1.9.0.js ) 으로 개발

❑ 릴리즈시에는 compressed 버전 ( jquery-1.9.0.min.js ) 으로 릴리즈한다.

[실습예제 1] jQuery 설치 및 버전 확인

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src="./jquery/jquery-1.9.0.js"></script>
<script>
alert( $().jquery  );
</script>
</head>
<body>
</body>
</html>
```

## ❑ (X)HTML + CSS + JavaScript ( jQuery )



## ❑ ready 함수

### [실습예제 2 - 1] javascript의 실행 타이밍.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src="./jquery/jquery-1.9.0.js"></script>
<script>
  alert("hello jquery");
</script>
</head>
<body>
<p>
    이 문장이 보이고 Hello World가 뜨면, 자바스크립트에서 HTML 엘리멘트에 접근할 수 있는 것입니다.
</p>
</body>
</html>
```

## ❑ ready 함수

### [실습예제 2 - 2] javascript의 실행 타이밍.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src="./jquery/jquery-1.9.0.js"></script>
<script>
  $( document ).ready( function() {
     alert( "Hello jQuery" );
  });
</script>
</head>
<body>
<p>
    이 문장이 보이고 Hello World가 뜨면, 자바스크립트에서 HTML 엘리멘트에 접근할 수 있는 것입니다.
</p>
</body>
</html>
```

## ❑ 선택자 ( selector )

- HTML Element를 선택하는 역할을 한다.
- 문서에서 Element 를 가져온 후, 반환된 객체 함수를 사용하여 Element를 조작하게 된다.

```
var object = $( “selector” ) ;
object.func();
```

1) CSS 에서 자주 사용하는 셀렉터
2) CSS2 셀렉터
3) CSS의 속성 셀렉터
4) jQuery 자체 필터

❑ CSS에서 자주 이용되는 셀렉터 - 태그

- 특정 HTML 태그를 컨트롤하기 위해 사용

[실습예제 3] ex3.html

1) li 엘리멘트 색상 바꾸기 예제 입니다.

2) ready function안을 비웠을 때와 비교 확인 해 보세요.

3) css() 함수의 사용법도 익혀 보세요.

❑ CSS에서 자주 이용되는 셀렉터 - ID

- 특정 id 속성을 가진 HTML 태그를 컨트롤하기 위해 사용

- ID값에 #(hash)를 붙힌다.

[실습예제 4] ex4.html

1) 특정 li 엘리멘트 색상 바꾸기 예제 입니다.

2) 아이디를 바꾸어 가며 테스트 해보세요.

❑ CSS에서 자주 이용되는 셀렉터 - 클래스

- 특정 class 속성을 가진 HTML 태그를 컨트롤하기 위해 사용

- .(dot)에 class 속성값을 지정하여 선택.

[실습예제 5] ex5.html

1) 특정 클래스를 가진 li 엘리멘트 색상 바꾸기 예제 입니다.

2) class blue를 가진 li엘리멘트도 색상을 바꿔보세요.

❑ CSS에서 자주 이용되는 셀렉터 - 자손 셀렉터

- 여러 개의 셀렉터를 스페이스로 구분 지어 특정 태그안에 있는 자식 태그까지 컨트롤 한다.

[실습예제 6] ex6.html

1) 특정 클래스를 가진 li 엘리멘트 의  엘리멘트의 색상 바꾸기 예제 입니다.

2) class blue를 가진 li엘리멘트의  엘리멘트중 id가 S1인 엘리멘트의 색상을 바꿔보세요.

❑ CSS에서 자주 이용되는 셀렉터 - 전체 셀렉터

- 전체 태그를 선택할 수 있다.

[실습예제 7] ex7.html

1) li 엘리멘트 안의 모든 자식 엘리멘트에 색상이 변경되는 예제 입니다.

2) 빨간색으로 변하지 않는 부분에 대해 왜 그런지 생각해 보세요.

❑ **CSS에서 자주 이용되는 셀렉터 - 그룹 셀렉터**

**- 여러 개의 셀렉터를 ,(콤마)로 구분하여 지정할 수 있다.**

**[실습예제 8] ex8.html**

**1) li 엘리멘트 안에 특정 엘리멘트들을 그룹핑해서 한 번에 색상을 변경하는 예제 입니다.**

**2) 아이디가 second, fourth 인 <li> 엘리멘트들은 폰트사이즈를 2.0em 으로 그리고 blue 색상으로 또 볼드 처리가 되게 스타일을 변경해 보세요.**

❑ CSS2 셀렉터 - 자식 셀렉터 ( IE6 에서 지원 안함 )

- 특정태그의 바로 아래 위치한 태그를 선택

[실습예제 9] ex9.html

1) li 엘리멘트의 자식 엘리멘트 중  엘리먼트의 색상을 변경하는 예제 입니다.

2)  엘리멘트 의 자식  엘리먼트도 같은 스타일로 적용해 보세요

.

## ❑ CSS2 셀렉터 - 인접 셀렉터 ( IE6 에서 지원 안함 )

- 특정 태그의 다음에 있는 태그를 선택할 수 있다.

[실습예제 10] ex10.html

1) 세번째 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.

2) 네 번째 <li>엘리먼트를 선택해 색상을 변경해 보세요.

## ❑ CSS2 셀렉터 - first-child 셀렉터 ( IE6 에서 지원 안함 )

- 특정 태그가 어떤 태그의 첫 엘리먼트인 경우 선택된다

**[실습예제 11] ex11.html**

1) 모든 첫번 째 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.

## ❑ CSS 속성 셀렉터 – [attribute]

- 특정 속성을 가진 태그를 선택한다.

**[실습예제 12] ex12.html**

1) **id** 속성을 가지고 있는 **<li>**엘리먼트를 선택해 색상을 변경하는 예제입니다.
2) **class** 속성을 가지고 있는 **<li>**엘리먼트를 선택해 색상을 **blue**로 변경해 보세요.

## ❑ CSS 속성 셀렉터 – [attribute='value']

- 특정 속성이 특정 값을 가지고 있는 엘리먼트를 선택한다.

**[실습예제 13] ex13.html**

1) title 속성이 "second" 속성을 가지고 있는 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.

2) title 속성이 "fourth" 속성을 가지고 있는 <li>엘리먼트를 선택해 색상을 blue로 변경해 보세요.

## ❑ CSS 속성 셀렉터 – [attribute!='value']

- 특정 속성이 특정 값을 가지고 있지 않은 엘리먼트를 선택한다.

**[실습예제 14] ex14.html**

1) class가 "blue"가 아닌 속성을 가지고 있는 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.
2) class가 "normal"이 아닌 <li>엘리먼트를 선택해 굵게 나오게 변경해 보세요.

## ❑ jQuery의 자체 필터 – first filter / last filter

- 셀렉터 안에서 첫 태그를 “first 필터”, 마지막 태그를 “last 필터”로 지정

**[실습예제 15] ex15.html**

1) 첫번째 <li>엘리먼트를 선택해 색상을 변경하는 예제입니다.

2) 마지막 <li>엘리먼트를 선택해 blue 색상으로 변경해 보세요.

## ❑ jQuery의 자체 필터 – even filter / odd filter

- 짝수 순서로 나타나는 엘리멘트는 **even** 으로 홀수 순서로 나타나는 태그는 **odd** 필터로 지정할 수 있다.

**[실습예제 16] ex16.html**

**1)** 홀수 번째 **<li>**엘리먼트를 선택해 색상을 변경하는 예제입니다.

**2)** 짝수 번째 **<li>**엘리먼트를 선택해 **blue** 색상으로 변경해 보세요.

## ❑ jQuery의 자체 필터 – contains 필터 / has 필터

- contains 필터는 특정 문자열이 포함되어 있는 엘리멘트를, has 필터는 특정 태그가 포함되어 있는 엘리멘트를 선택한다.

**[실습예제 17] ex17.html**

1) 샘플이라는 단어가 들어간 콘텐츠를 가지고 있는 엘리먼트 와  태그르 포함하고 있는 엘리멘트를 골라 색상르 변경하는 예제입니다.

## ❑ 텍스트의 변경

**.text()** : 파라미터로 문자열을 넘기면 태그안의 텍스트를 문자열로 변경한다.

[실습예제 18] ex18.html

파라미터 내용에 태그를 붙여 넘겨 보세요. “가나다라마바사아자차카타파하”

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src="./jquery/jquery-1.9.0.js"></script>
<script type="text/javascript">
  $( function() {
     $( "#p1" ).text( "가나다라마바사아자차카타파하" );
  } );
</script>
</head>
<body>
<p id="p1">이 안의 텍스트를 바꿉니다.</p>
</body>
</html>
```

## ❑ 텍스트 가져오기

**.text()** : 파라미터가 없으면 태그에 포함된 텍스트를 가져온다.

[실습예제 19] ex19.html

안녕하세요 로 바꾸고 결과를 확인해 보세요.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src="./jquery/jquery-1.9.0.js"></script>
<script type="text/javascript">
  $( function() {
     alert( $( "#p1" ).text() );
  } );
</script>
</head>
<body>
<p id="p1">안녕하세요.</p>
</body>
</html>
```

## ❑ HTML의 변경

**.html()** : 파라미터로 HTML문자열을 넘기면 태그안의 내용에 그 HTML 이 반영

[실습예제 20] ex20.html

실습예제18 와 비교, 확인해 보세요.

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src="./jquery/jquery-1.9.0.js"></script>
<script type="text/javascript">
   $( function() {
       $( "#p1" ).html(  "<strong>가나다라마바사아자차카타파하</strong>"  );
   } );
</script>
</head>
<body>
<p id="p1">이 안의  HTML를 바꿉니다.</p>
</body>
</html>
```

## ❑ HTML가져오기

**.html()** : 파라미터가 없으면 태그에 포함된 html을 그대로 가져온다.

[실습예제 21] ex21.html

실습예제19 와 결과를 비교 확인해 보세요

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<script type="text/javascript" src="./jquery/jquery-1.9.0.js"></script>
<script type="text/javascript">
  $( function() {
     alert( $( "#p1" ).html() );
  } );
</script>
</head>
<body>
<p id="p1"><strong>안녕하세요. </strong></p>
</body>
</html>
```

## ❑ HTML 삽입

- html()은 태그안의 내용을 전부 변경
- 기존의 태그안의 내용은 남긴 채, HTML을 추가 삽입할 경우
- 다음과 같은 함수를 사용

**prepend()**

**append()**

**before()**

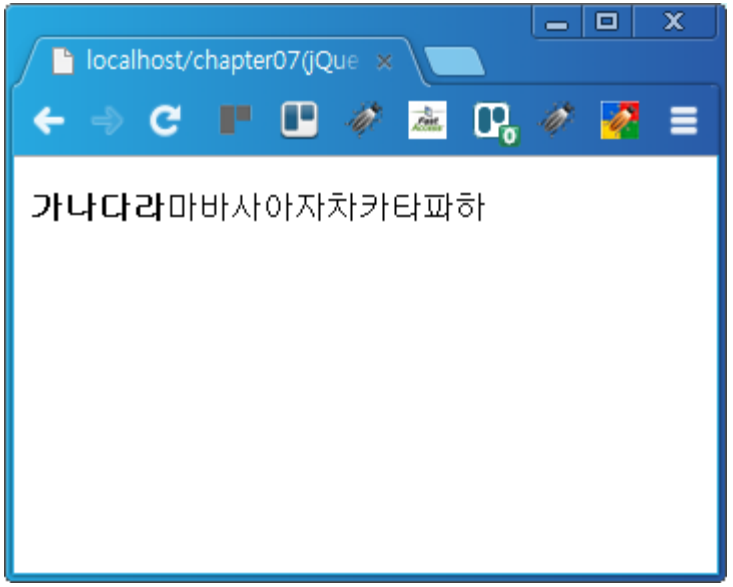
**after()**

## ❑ HTML 삽입 – prepend( )

- 지정한 태그 **안의 내용의 앞**에 파라미터 **HTML**를 삽입한다.

[실습예제 22] ex22.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

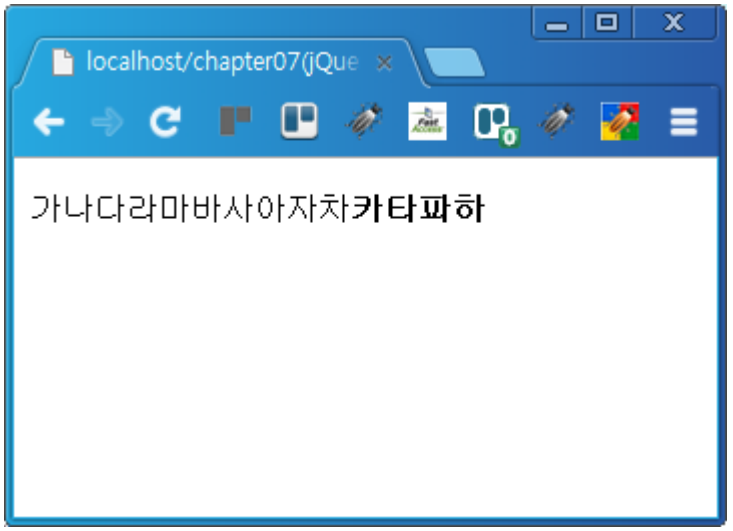


## ❑ HTML 삽입 – append( )

- 지정한 태그 **안의 내용의 뒤** 에 파라미터 **HTML**를 삽입한다.

[실습예제 23] ex23.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

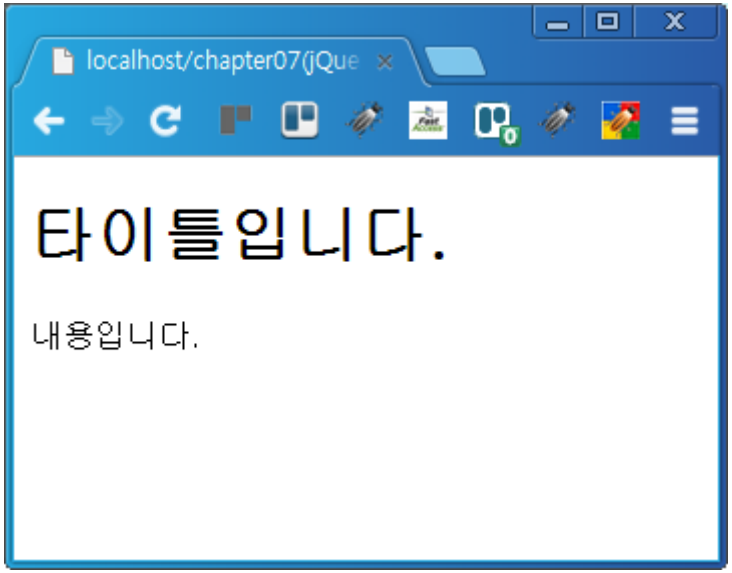


## ❑ HTML 삽입 – before( )

- 지정한 태그 **앞**에 파라미터 **HTML**를 삽입한다.

[실습예제 24] ex24.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

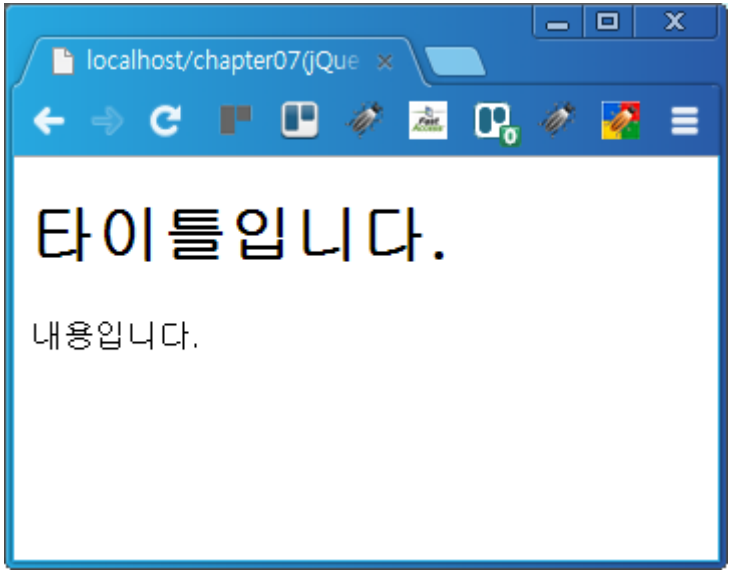


## ❑ HTML 삽입 – after( )

- 지정한 태그 **뒤**에 파라미터 **HTML**를 삽입한다.

[실습예제 25] ex25.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

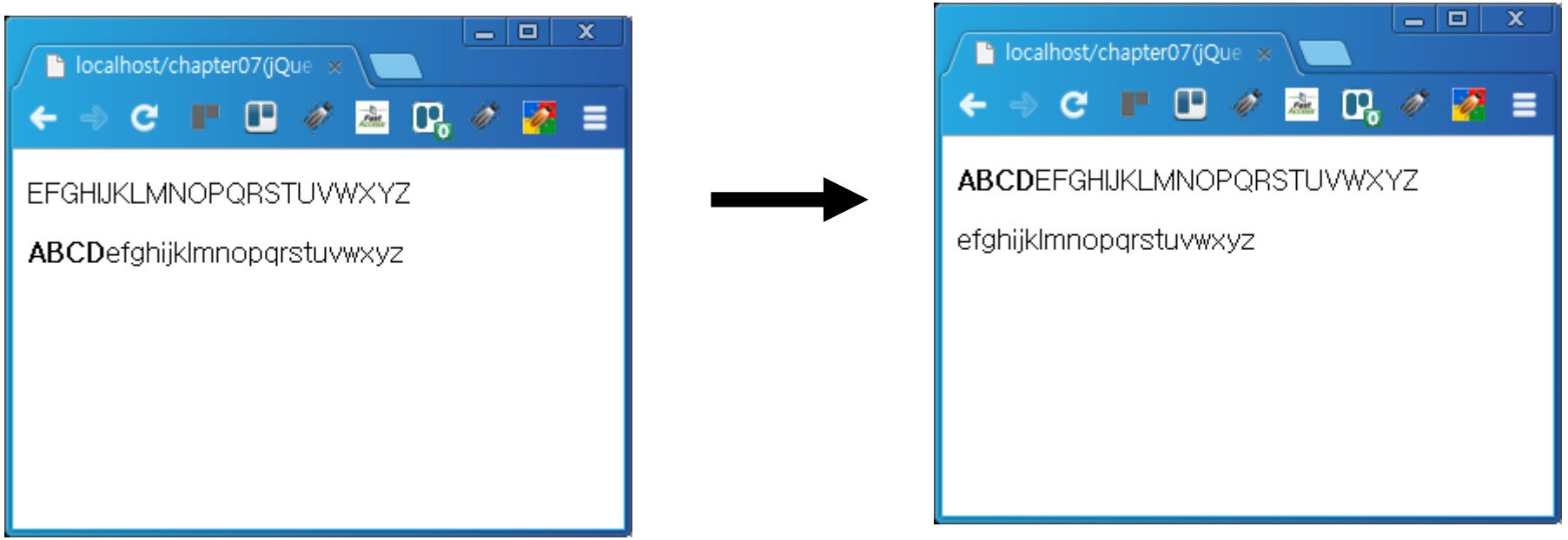


## ❑ HTML 이동 – prependTo( "이동할 곳의 선택자" )

- 선택자로 지정한 태그를 다른 태그 **안에 포함된 텍스트 앞**으로 이동

[실습예제 26] ex26.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

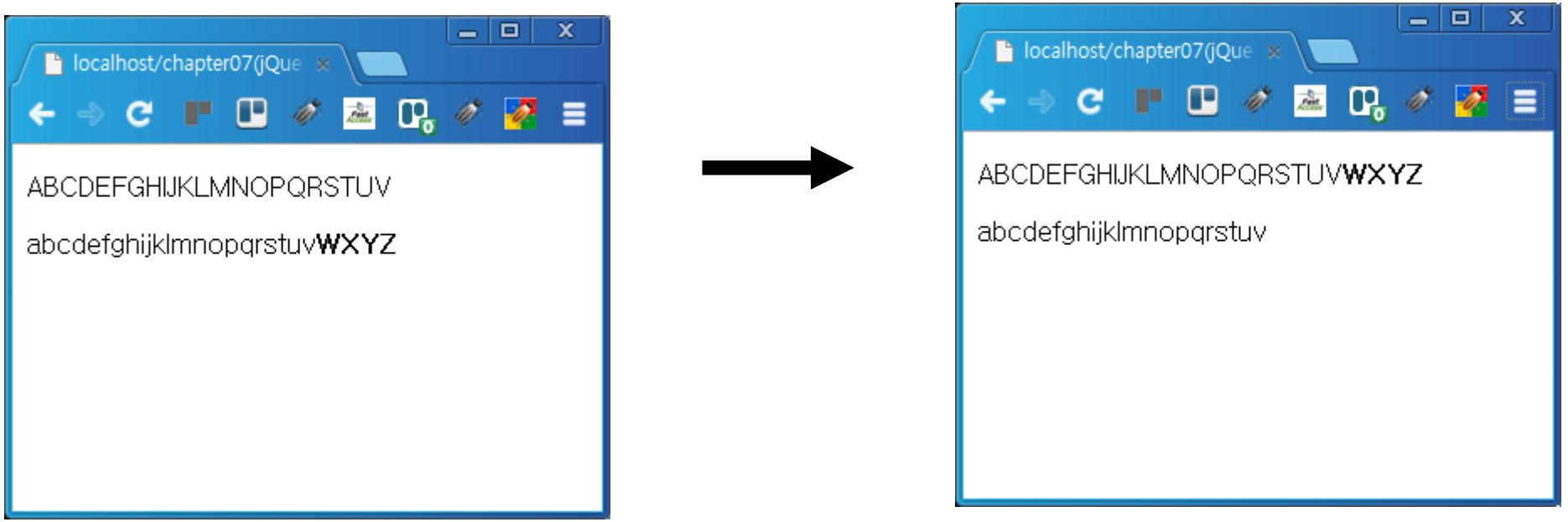


## ❑ HTML 이동 – appendTo( "이동할 곳의 선택자" )

- 선택자로 지정한 태그를 다른 태그 **안에 포함된 텍스트 뒤**로 이동

[실습예제 27] ex27.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

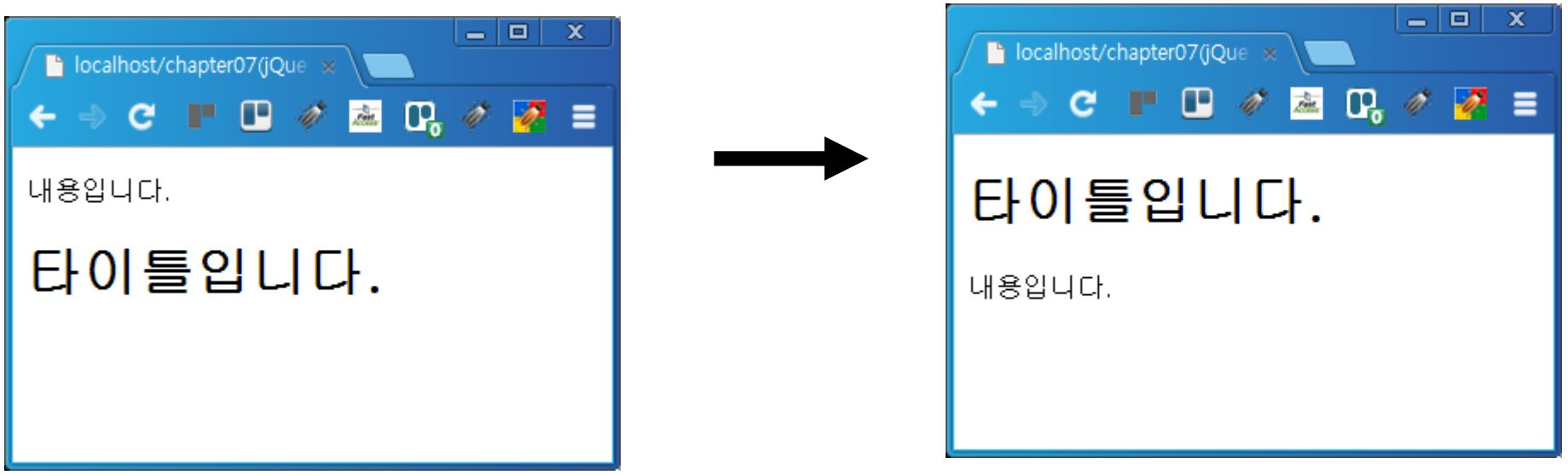


## ❑ HTML 이동 – insertBefore( "이동할 곳의 선택자" )

- 선택자로 지정한 태그를 다른 태그 **앞**으로 이동

[실습예제 28] ex28.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

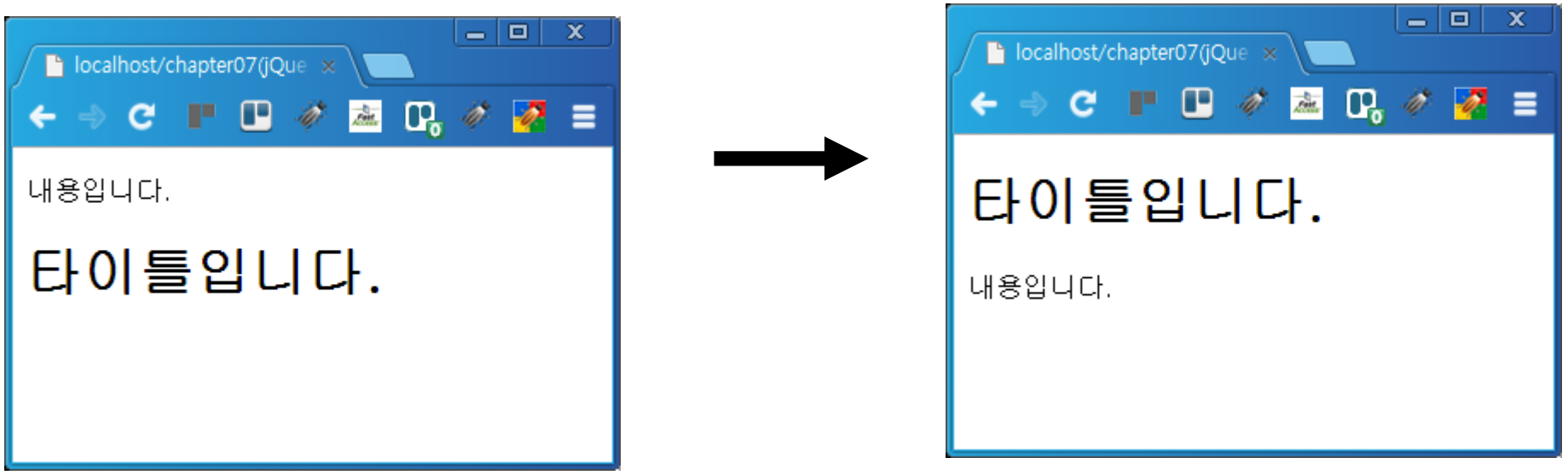


## ❑ HTML 이동 – insertAfter( "이동할 곳의 선택자" )

- 선택자로 지정한 태그를 다른 태그 **뒤**로 이동

[실습예제 29] ex29.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

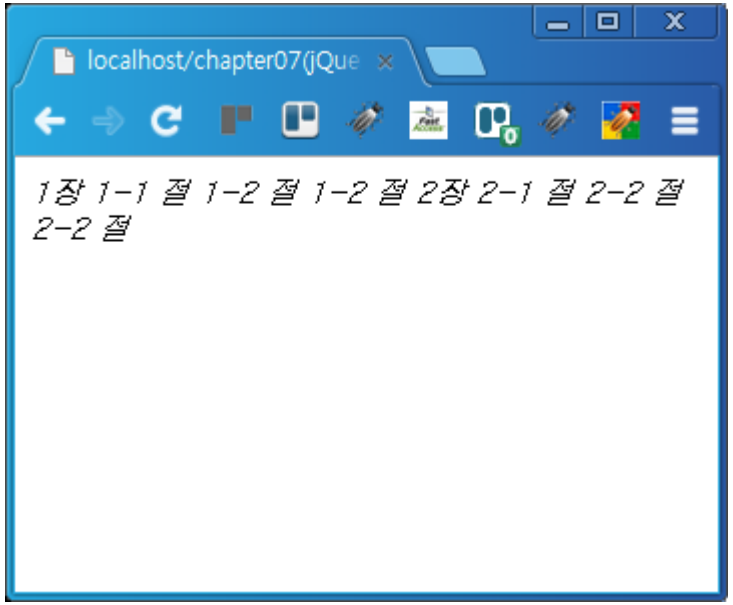


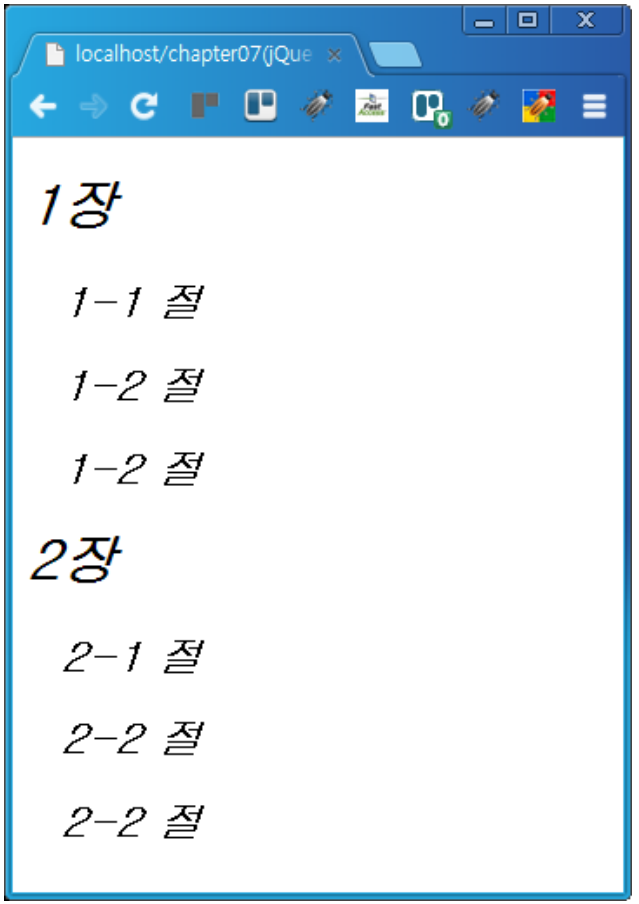
## ❑ 다른 태그로 감싸기 – wrap()

- 선택자로 지정된 각 각의 엘리먼트를 파라미터에 지정된 태그로 감싼다.

[실습예제 30] ex30.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.



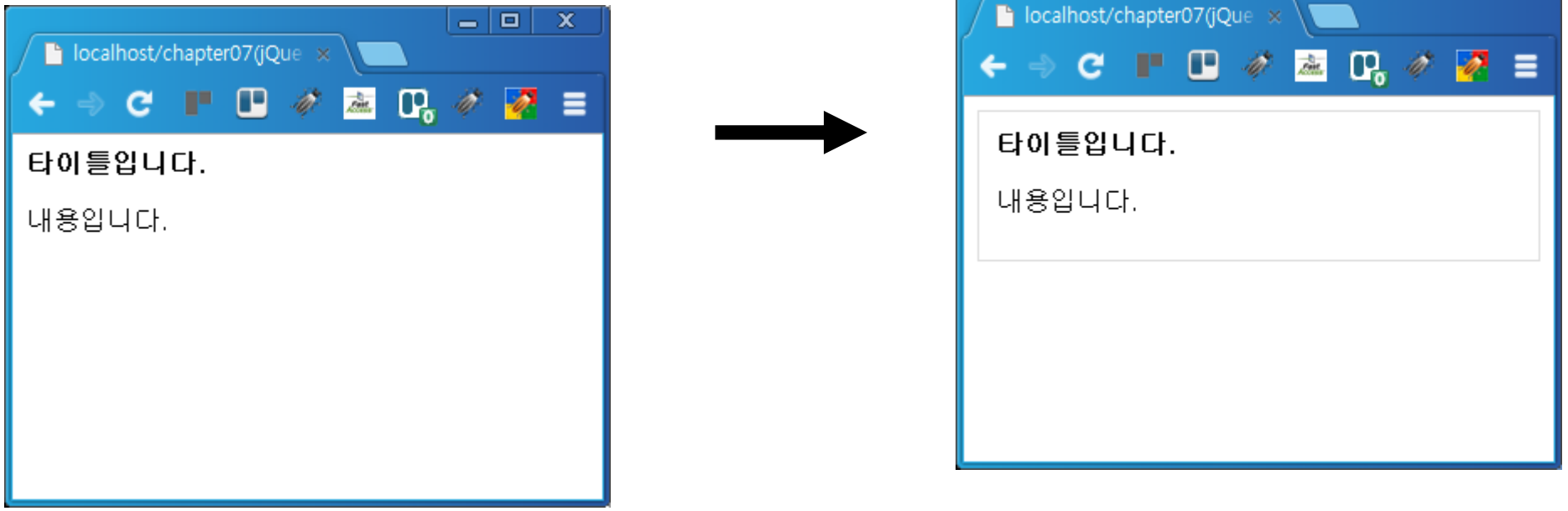



## ❑ 다른 태그로 감싸기 – wrapAll()

- 선택자로 지정된 복수의 엘리먼트를 파라미터에 지정된 하나의 태그로 감싼다.

[실습예제 31] ex31.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

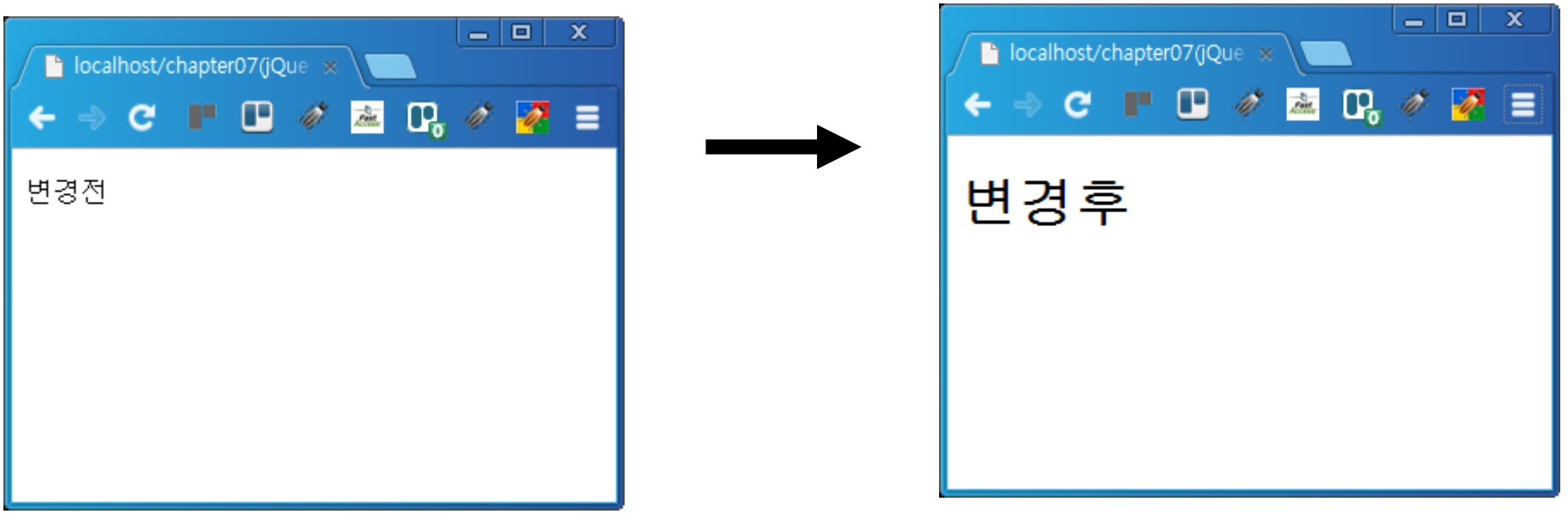


## ❑ 태그 변경 – replaceWith()

- 지정한 태그를 다른 태그로 바꾸고자 할 때
- 엘리먼트 내용까지 변경된다. (태그만 변경하는 것이 아님)

[실습예제 32] ex32.html

다음화면과 같은 결과가 나오도록 코드를 완성하세요.

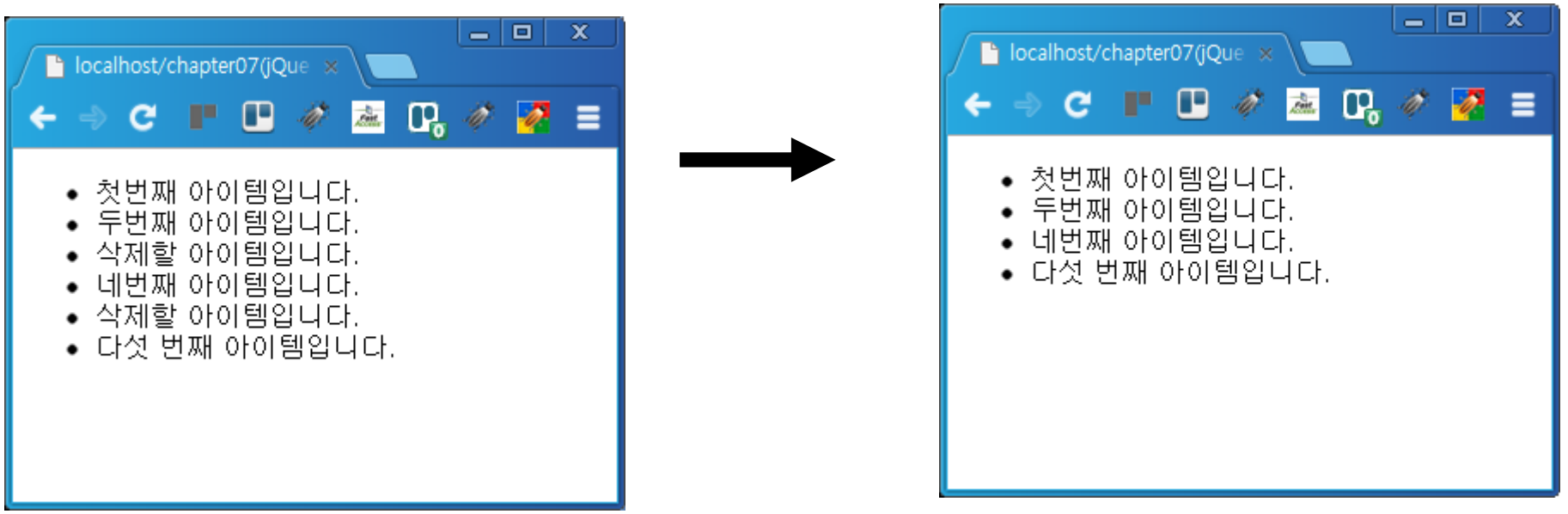


## ❑ 태그 제거 – remove()

- 선택자로 지정된 태그들을 제거한다.

[실습예제 33] ex33.html

리스트에서 삭제할 아이템들을 삭제해 보세요.



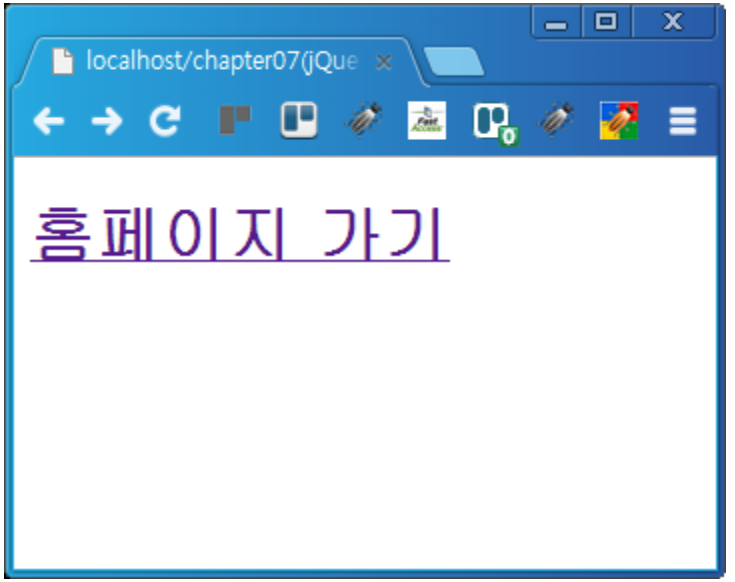
## ❑ 속성값 변경/가져오기 – attr()

- 태그의 속성값을 변경 할 수 있다.

```
$( “셀렉터” ).attr( “속성명”, “속성값” );
```

[실습예제 34] ex34.html

1) 여러분 회사 홈페이지로 링크를 바꿔 보세요.
2) 새 창에서 열리도록 target=_blank 속성을 추가해 보세요.

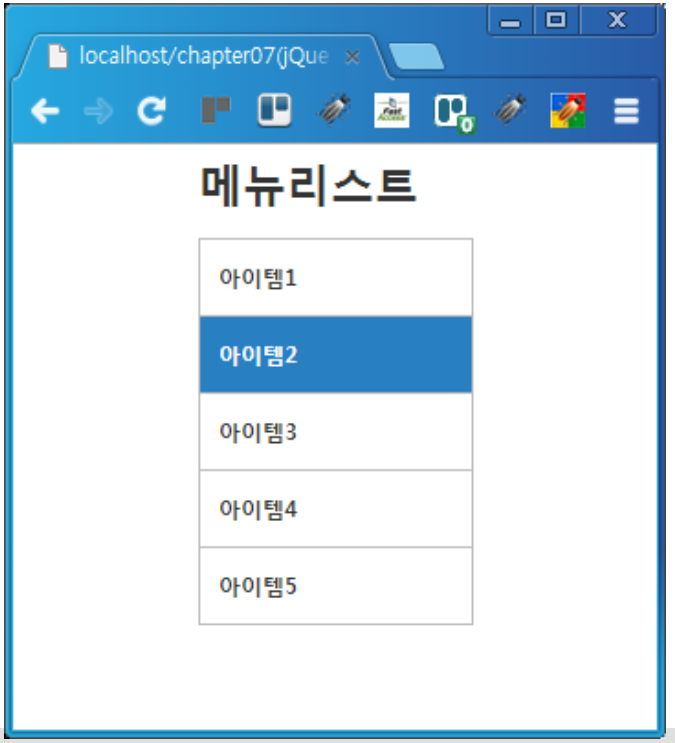


## ❑ Class 속성의 추가 와 제거 – addClass(), removeClass()

```
$( "셀렉터" ).addClass( "클래스 이름" );

$( "셀렉터" ).removeClass( "클래스 이름" );
```

[실습예제 35] ex35.html

1) ex35.html 의 CSS 살펴보기
2) 각 리스트 아이템에 mouseover 이벤트 받아서 선택된 class이름 추가하기
3) 각 리스트 아이템에 mouseout 이벤트 받아서 선택된 class이름 삭제하기



## ❑ CSS 제어 – 설정

```
$( “셀렉터” ).css( {
   속성명: “속성값”,
   속성명: “속성값”,
   …
   …
   속성명: “속성값”
} );
```

[실습예제 36] ex34.html

h1에 다음 속성값을 적용해 보세요 ( 폰트 크기 : 1.2em, 폰트 : 맑은 고딕 )

a:link, a:visited, a:active, a:hove ( color: #333, 밑줄은 없다 )

❑ HTML 로딩이 완료된 후 이벤트 처리가 되어야 한다.

```
$(  document  ).ready( function() {
    /* HTML Element에 접근이 가능하다 */
});
```

축약

```
$(  function() {
    /* HTML Element에 접근이 가능하다 */
});
```

## ❑ click 이벤트

```
$(셀렉터).click( function() {
    /* 셀렉터로 지정한 태그가 클릭되었을 때 실행하는 처리 */
});
```

[ 실습예제 37 ]

그림과 같이 버튼을 누르면 8개의 이미지가 램덤하게 하나씩 화면에 나오는 프로그램을 작성하세요.

[힌트]

```
var images = [
    "국화:Chrysanthemum.jpg", "사막:Desert.jpg", "수국:Hydrangeas.jpg", "해파리:Jellyfish.jpg",
    "코알라:Koala.jpg", "등대:Lighthouse.jpg", "펭귄:Penguins.jpg", "튤립:Tulips.jpg" ]

var result = Math.floor( Math.random() * ( images.length - 1 ) ) + 1;
```

## ❑ dbclick 이벤트

```
$(셀렉터).dbclick( function() {
    /* 셀렉터로 지정한 태그가 더블클릭 되었을 때 실행하는 처리 */
});
```

[ 실습예제 **38** ]

실습예제 **37** 에서 화면에 나타난 그림을 더블클릭하면 **img** 속성중 **alt** 속성의 내용을 **alert**창으로 나오게 하세요.

## ❑ mousedown(), mouseup() 이벤트

```
$(셀렉터).mousedown( function() {
    /* 셀렉터로 지정한 태그에  마우스 버튼이 눌렀을 때 실행하는 처리 */
});
```

[ 실습예제 39 ]

실습예제 37 에서 화면에 나타난 그림의 mousedown event 에 그림이 변하는 click 이벤트와 동일한 반응이 되도록 처리해 보세요.

# Part III
# 자바스크립트
## 3. AJAX

목차

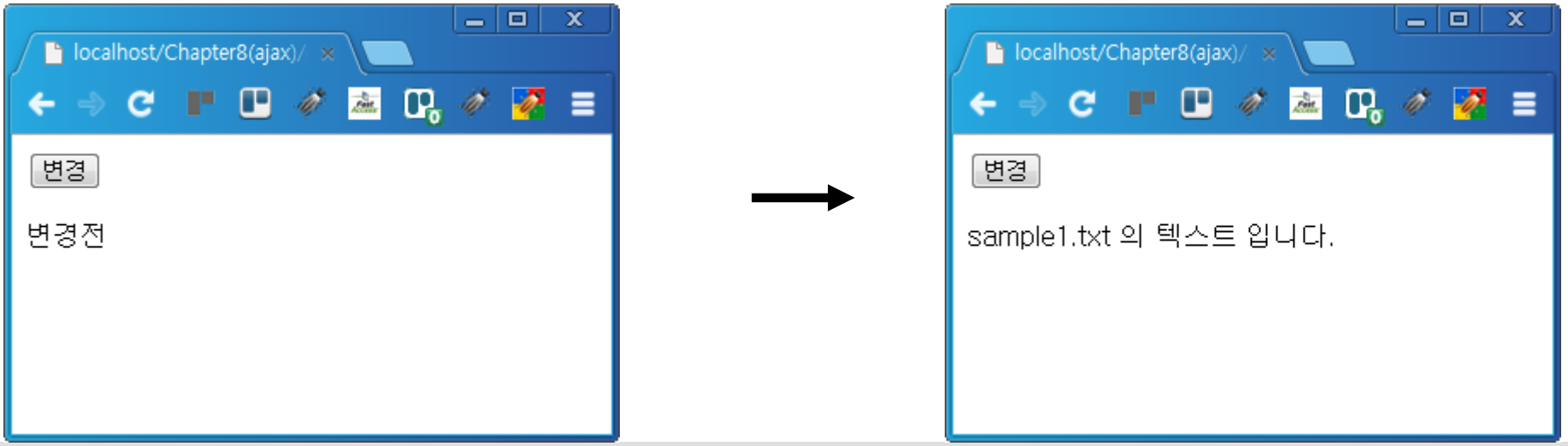
# 1. 개요

- **AJAX ( Asynchronous Javascript XML )**
- **XML**를 이용한 비동기  통신
- **JavaScript**를 이용해서 서버에서 데이터를 가져와 페이지 전체의 갱신없이 특정부분만 변경
- **XML**를 주고받은 데이터 타입으로 정의 하고 있으나 정해져 있지는 않다.
- **JSON**으로 서버와 데이터를 주로 주고 받는다.

## ❑ 웹페이지에 텍스트 삽입하기

### [실습예제 1] ex1.html

```
<script type="text/javascript">
$(function(){
        $("button").click(function(){
                $("p").load("./ex1_text.txt");
        });
});
</script>
```



## ❑ 외부 HTML 표시

### [실습예제 2] ex2.html

```
<script type="text/javascript">
$(function(){
        $("button").click(function(){
                $("div").load("./ex2_load.html");
        });
});
</script>
```

변경전 ⟶

```
<div>
  <p>ex2_load.html 의 p태그 입니다.</p>
</div>
```

## ❑ JSON 으로 Servlet과 통신

### [실습예제 2] ex3.html

```
$("button").click(function(){

    $.ajax( {
        url : "ex3Json?rnd=" + Math.floor(Math.random() * 999999999),
        type: "get",
        dataType: "json",
        data: "",
        contentType: 'application/json',
        success: function(data){
            alert( data.name );
            alert( data.message );
        },
        error: function( jqXHR, status, e ){
                alert( status + " : " + e );
        }
    } );

});
```

## ❑ JSON 으로 Servlet과 통신

### [실습예제 2] ex3JsonServlet.java

```
response.setContentType( "application/json;charset=utf-8" );

PrintWriter out = response.getWriter();
out.print( "{ \"name\": \"test\", \"message\": \"hello\" }" );
```

**JSON Object**

```
{
    name: "test",
    message: "hello"
}
```

# 2. 구현

**[실습과제]**

**회원 가입에서 아이디 중복체크**

1) **UserDao** 에서는 파라미터로 넘어온 **email**이 **User table**에 존재하는 지 유무를 확인하고 **servlet** 에서는 다음과 같은 **JSON** 을 브라우저에게 보냅니다.

   **{ result : "exist" }** , **{ result : "not exist" }**

2) **Javascript** 에서는 **jQuery Ajax** 를 사용해 서버에게 파라미터로 **email**를 보내고 서버에서 보내온 결과 **JSON**를 적절히 분석해 화면에 나타냅니다.