SK planet

동GO동락 | T아카데미

[온라인] JavaScript 기본

제1강. 자바스크립트 개요

자바스크립트란?

웹 브라우저에 내장되어 있는 스크립트(script) 언어

객체지향(object oriented) 프로그래밍

인터프리터(interpreter) 방식

동적이면서 약한 타입(dynamic and weak typing) 언어

웹 개발환경의 변화

1994년 Brendan Eich가 개발

1995년 넷스케이프 네비게이터 2.0에 탑재

모든 웹 브라우저에 내장된 클라이언트 측 스크립트 언어

2005년 초 Ajax 기술의 보급

2008년부터 가열된 웹 브라우저간의 자바스크립트 엔진 성능 경쟁

웹 개발환경의 변화 (계속)

서버 측에서의 자바스크립트 프로그래밍

2009년 Ryan Dahl이 Node.js 개발

HTML 5의 중심에 놓여 웹 표준으로서 위상을 가짐

모바일 환경까지 그 응용 범위의 세를 더욱 확장

모바일 앱 개발환경과 자바스크립트의 위상

2007년 아이폰의 등장과 함께 시작된 스마트폰의 열풍

엄청난 모바일 앱 시장의 성장

스마트 폰 플랫폼의 다양성이 바로 문제의 발단

- 개발 비용의 증가, 유지보수 비용의 증가

모바일 웹 브라우저의 신속한 HTML 5 지원

jQuery Mobile, Sencha Touch 와 같은 모바일 웹 앱 개발 프레임워크 등장

모바일 앱 개발환경과 자바스크립트의 위상 (계속)

모바일 웹 앱(mobile web app) - 속도의 한계

하이브리드 모바일 앱(hybrid mobile app) 대안

모바일 시대의 앱 개발에 있어 자바스크립트의 위상과 중요성이 날로 높아짐

자바스크립트 개발도구 설치

통합개발환경 Eclipse를 기반으로 한 Aptana Studio

오픈소스

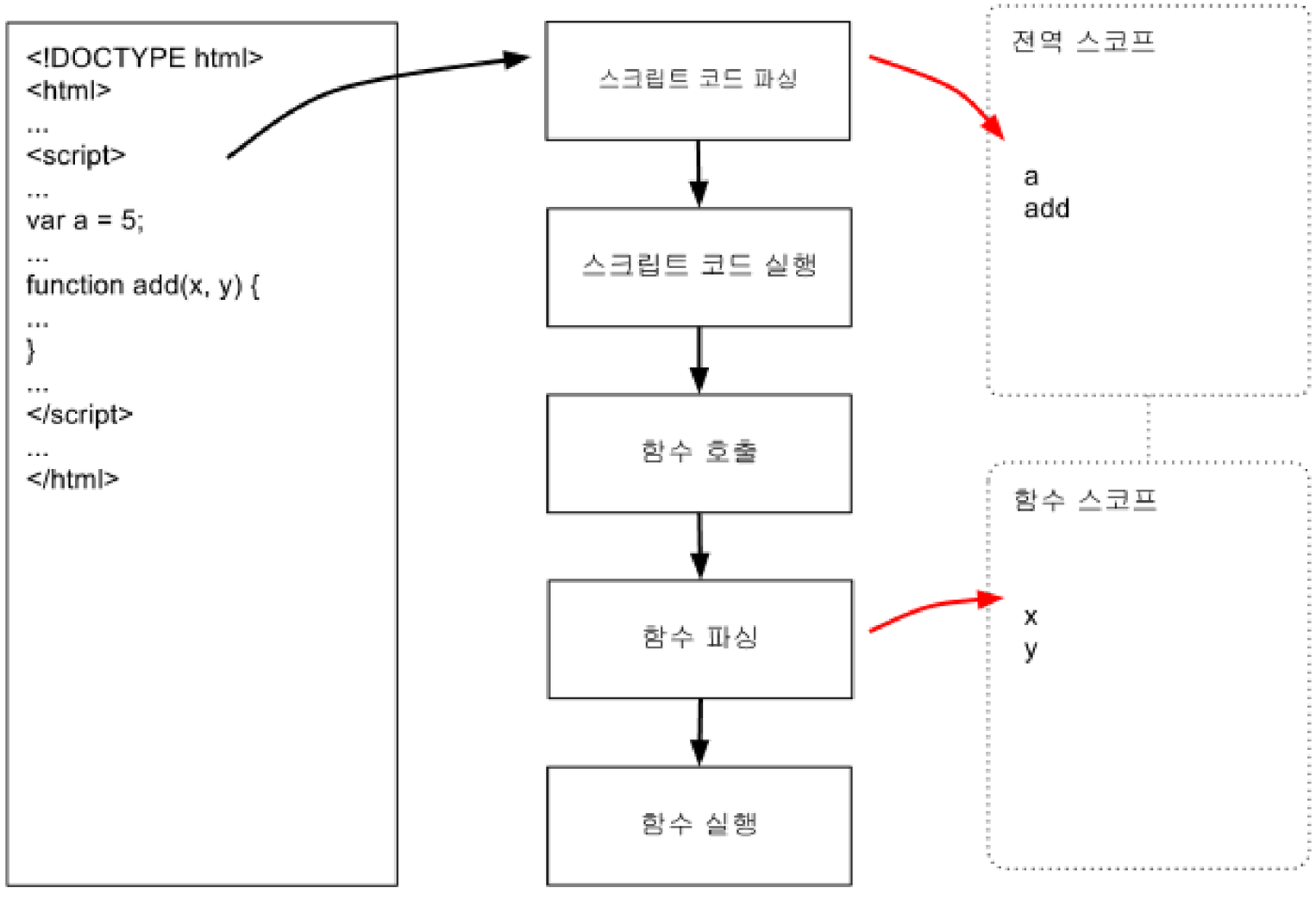
Windows, Mac OS X, Linux 등의 멀티플랫폼을 지원

SK planet

동GO동락 | T아카데미

2강. 변수, 데이터 타입, 리터럴

<script> 태그와 자바스크립트 - 자바스크립트 코드의 실행



자바스크립트 출력하기 - HTML 페이지에서의 자바스크립트 기술 방식

인라인 스크립트 방식

- <script> 태그 아래 자바스크립트 코드를 직접 작성

외부 스크립트 방식

- 자바스크립트 코드를 외부 파일(확장자는 js)에 작성하고 이 파일의 이름을 <script> 태그의 src 속성 값으로 지정

외부 스크립트와 인라인 스크립트는 동시에 작성 불가 (인라인스크립트의 내용은 무시됨)

<script> 태그는 <head> 태그 아래 혹은 <body> 태그 아래에 작성 가능.

자바스크립트 출력하기 - 실습 02-01

문장규칙

(1) 문장의 끝에 세미콜론(;)을 붙입니다.

(2) 대소문자를 구별합니다.

(3) 문장에서의 화이트 스페이스 문자는 무시됩니다.

주석

```
// 행의 끝까지를 주석으로 처리하는 단일 행 주석

/*
  여러 라인에 걸쳐 주석 블록을
  구성할 수 있는 복수 행 주석
*/
```

변수 선언과 var

변수 : 프로그램이 어떤 값을 메모리에 저장해 두고 다시 사용하기 위한 공간

var 키워드 : 자바스크립트는 값을 저장하기 위한 공간을 확보하기 위해 var 키워드를 이용해 변수를 선언

```
var 변수명 [= 초기값];
```

변수 선언 시 초기값을 지정하지 않을 경우, 값을 저장할 때까지 그 변수는 undefined 상태임

변수 선언과 var

```
var a;  v
ar b, c;
var d = 0;
var e = 2, f = 4;
var g = 6, h;
var i, j = 8;
```

변수 선언 시 var 키워드 생략이 가능하지만, 변수 스코프 문제가 발생할 수 있으므로 var 키워드는 생략하지 않는 것이 좋습니다.

식별자 규칙

(1) 첫번째 문자는 [A-Za-z_$] 만 사용합니다.

(2) 나머지 문자는 [A-Za-z_$0-9] 만 사용합니다.

kor_score

averageScore

phone1

_amount

$val

(3) 자바스크립트 예약어는 사용할 수 없습니다.

값에 의한 데이터 타입 결정

저장되는 값에 따라 변수의 데이터 타입이 바뀝니다.

```
var k = 10;
var l = '문자열';
l = 12;
```

값에 의한 데이터 타입 결정 - 실습 02-02

값을 저장하는 기본형 데이터 타입

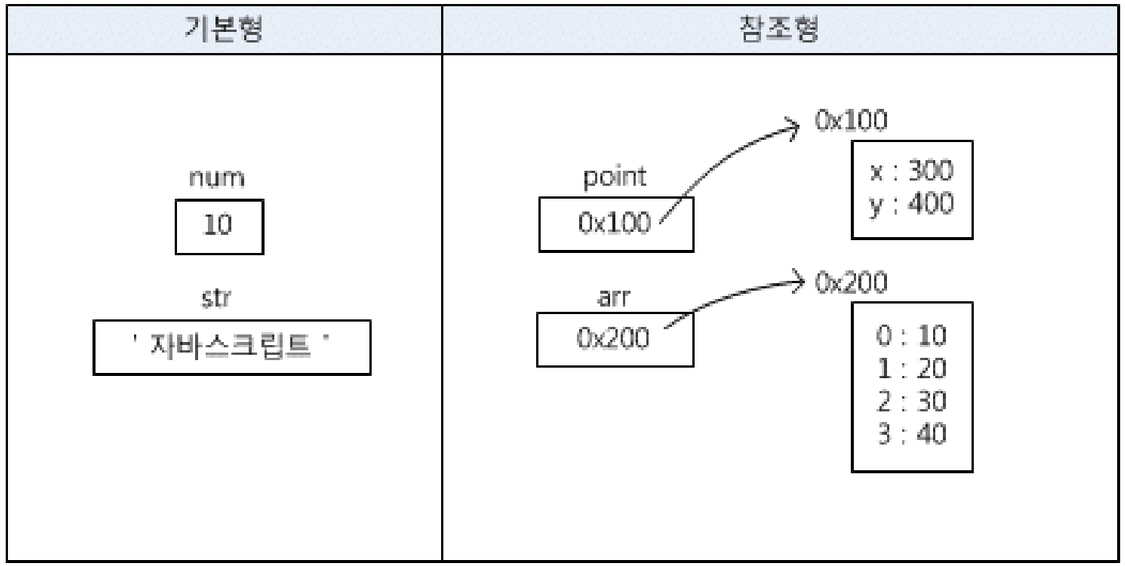
데이터 타입	특징
숫자	자바스크립트는 정수 값과 실수 값을 구분하지 않습니다. 모든 숫자는 IEEE 754 표준에 의 해 정의된 8바이트 크기의 실수로 표현하며 ±5 × 10 10^{308}
문자열(string)	유니코드 문자나 숫자, 문장부호들의 시퀀스로 텍스트를 표현합니다.작은따옴표(') 혹은 큰 따 옴표(") 쌍으로 문자열을 둘러싸서 문자열을 표현합니다. 단일 문자 표현은 길이가 1인 문자 열로 표현합니다.
불리언(boolean)	불리언 형은 참/거짓의 진리 값 두 개를 표현하는데, true 또는 false 값을 가집니다.
null	예약어 null은 보통 참조 타입과 함께 쓰여, 어떠한 객체도 나타내지 않는 특수한 값으로 사용 합니다.
undefined	undefined는 변수는 선언되었으나 값이 할당된 적이 없는 변수에 접근하거나, 존 재하지 않 는 객체 프로퍼티에 접근할 경우 반환되는 값입니다.

참조값을 저장하는 참조형 데이터 타입

데이터 타입	특징
배열(array)	배열은 데이터 값들의 모음입니다. 배열의 각 데이터 값에는 0부터 시작하는 인덱스라는 번 호가 있어서 배열명 다음에 대괄호([])로 둘러싼 인덱스를 써서 값을 가져올 수 있습니다.
객체(object)	객체는 이름이 붙은 값들의 모음입니다. 이 이름이 붙은 값들을 프로퍼티라고 하며, 객체 프 로퍼티는 객체명 다음에 점(.)과 프로퍼티명을 붙이거나, 객체명 다음에 대괄호([])로 둘러 싼 프로퍼티명의 문자열을 써서 접근할 수 있습니다.
함수(function)	자바스크립트에서 함수는 객체 프로퍼티에 할당될 수 있는 실행가능한 코드를 가지고 있는 데이터 타입입니다.

기본형 데이터 타입과 참조형 데이터 타입의 변수 선언과 메모리 할당

```
// 기본형 데이터 타입
var num = 10;
var str = '자바스크립트';
```




```
// 참조형 데이터 타입
var point = { x : 300, y : 400 };
var arr = [ 10, 20, 30, 40 ];
```

기본형 데이터 타입과 참조형 데이터 타입의 변수 선언과 메모리 할당 – 실습 02-03

리터럴

리터럴(Literal): 프로그램의 코드 상에 데이터의 값을 표현하는 방식

자바스크립트의 리터럴:

- 숫자 리터럴, 문자열 리터럴, 배열 리터럴, 객체 리터럴, 함수 리터럴, 불리언 리터럴, 그리고 undefined와 null 리터럴

숫자 리터럴

자바스크립트에서 숫자 리터럴 중 정수(fixed point) 리터럴은 10진수, 16진수로 표현

16진수 리터럴은 '0x'나 '0X'로 시작하고 뒤에 16진수 숫자들을 붙여 표현

16진수 숫자는 0~9과, 10~15를 표현하는 a(혹은 A)~f(혹은 F) 중 하나를 사용

8진수 리터럴은 0으로 시작하고 뒤에 0~7사이의 숫자들을 붙여 표현

```
0
32767
0xff
0377
```

숫자 리터럴 (계속)

자바스크립트에서 숫자 리터럴 중

부동소수점(floating point) 리터럴 표기법으로는 실수표기법과 지수표기법이 있음

실수표기법: [digits][.digits]

지수표기법: [digits][.digits][(e|E)[(+|-)]digits]

3.14
98.45
.333333333333
1.5e10
5.25E-21

숫자 리터럴 (계속)

Infinity, NaN를 포함한 모든 숫자 데이터는 typeof 연산의 결과 “number”를 반환

숫자 리터럴 (계속)

상수	정의
Infinity	무한대를 나타내는 특수한 숫자 값입니다. isFinite() 함수가 false값을 반환하면 Infinity값임을 확인할 수 있으며, == 비교연산이 가능합니다.
NaN	숫자가 아님을 나타내는 특수한 숫자 값입니다. NaN값에 대한 확인은 == 비교연산 을 사용할 수 없으며, 반드시 isNaN() 함수를 사용해야 합니다.
Number.MAX_VALUE	숫자로 표현할 수 있는 가장 큰 수(1.7976931348623157 × 10 있습니다.
Number.MIN_VALUE	숫자로 표현할 수 있는 가장 작은 수(5 × 10
Number.NaN	NaN과 동일합니다.
Number.POSITIVE_INFINITY	양의 무한대(Infinity)를 나타내는 특수한 숫자 값입니다.
Number.NEGATIVE_INFINITY	음의 무한대(-Infinity)를 나타내는 특수한 숫자 값입니다.

숫자 리터럴 - 실습 02-04

문자열 리터럴

문자열 리터럴은 유니코드 문자들을 작은따옴표(') 또는 큰따옴표(")로 둘러싸서 표현

작은따옴표로 만든 문자열내에서는 큰따옴표를 포함할 수 있고, 큰따옴표로 만든 문자열에서는 작은따옴표를 포함할 수 있음

문자열 리터럴은 한 줄을 넘을 수 없으며, 만일 문자열 리터럴 내에서 줄바꿈을 표현하고자 한다면 이스케이프 시퀀스 '₩n'를 사용합니다.

```
''    // 빈 문자열
""    // 빈 문자열
'문자열 리터럴'
'작은따옴표로 만든 문자열에는 큰따옴표(")를 포함할 수 있습니다.'
"큰따옴표로 만든 문자열에는 작은따옴표(')를 포함할 수 있습니다."
```

문자열 리터럴 (계속)

이스케이프 시퀀스	유니코드	의미
₩0	₩u0000	널 문자
₩b	₩u0008	백스페이스
₩t	₩u0009	수평 탭
₩n	₩u000A	줄바꿈 문자
₩v	₩u000B	수직 탭
₩f	₩u000C	폼 피드 (새로운 페이지로 이동)
₩r	₩u000D	캐리지 리턴 (현재 행의 처음으로 이동)
₩”	₩u0022	큰 따옴표
₩’	₩u0027	작은 따옴표
₩₩	₩u005C	역슬래시
₩xXX		두 개의 16진수 XX에 의해 지정되는 Latin-1 문자
₩uXXXX		네 개의 16진수 XXXX에 의해 지정되는 유니코드 문자

문자열 리터럴 (계속)

역슬래시 자체를 출력하려면 이스케이프 시퀀스 ₩₩을 사용

작은따옴표 자체를 출력하려면 ₩'을, 큰따옴표 자체를 출력하려면 ₩"을 사용

'작은따옴표로 만든 문자열에 ₩₩₩'를 통해 작은따옴표(₩')를 포함할 수 있습니다.'
"큰따옴표로 만든 문자열에 ₩₩₩"를 통해 큰따옴표(₩")를 포함할 수 있습니다.";
'이 문자열은 줄바꿈을 통해 첫번째 행을 출력하고,₩n이어 두번째 행을 출력합니다.'
'원주율 ₩u03c0는 원의 둘레와 지름의 비율입니다.'

유니코드 문자 ₩u03c0는 수학기호 π(파이)

문자열 리터럴 - 실습 02-05

배열 리터럴

배열: 데이터 값들의 모음

배열의 원소인 데이터 값은 배열명 다음에 대괄호([])로 둘러싼 인덱스(index)를 써서 그 값을 가져오거나 변경할 수 있음

배열의 인덱스는 0부터 시작해서, 배열의 원소 개수 - 1까지의 유효범위를 가짐

배열의 원소의 개수는 length 프로퍼티에 접근해 알 수 있음

자바스크립트에서 배열의 원소는 어떠한 데이터 타입라도 사용할 수 있음

배열 리터럴은 콤마(,)로 구분한 값을 대괄호([])로 감싸서 표현함

배열 리터럴 (계속)

```
[ ]    // 빈 배열
[ 95, 91, 100, 286, 95.33 ]       // 숫자를 원소로 갖는 배열
[ '홍길동', '이순신', '강감찬' ]          // 문자열을 원소로 갖는 배열
```

```
// 배열을 원소로 갖는 배열
[ ['을지문덕', '연개소문'], ['광개토대왕', '세종대왕'], ['김유신', '계백'] ]
```

배열 리터럴 - 실습 02-06

객체 리터럴

객체: 이름이 붙어 있은 데이터 값들의 모음

프로퍼티:

- 이름이 붙어 있는 값

- 객체의 프로퍼티는 객체명 다음에 점(.)과 프로퍼티명을 붙이거나, 객체명 다음에 대괄호([])로 둘러싼 프로퍼티명의 문자열을 써서 접근할 수 있음

- 프로퍼티에는 배열, 함수, 객체를 포함한 어떠한 값이라도 담을 수 있음

메소드:

- 함수가 프로퍼티에 저장될 경우 프로퍼티명이 메소드명이 됨

- 객체가 소유한 메소드를 호출하기 위해서는 객체명 다음에 점(.)을 붙이고 함수를 호출하기 위해 메소드명과 ()를 사용함

객체 리터럴 (계속)

객체 리터럴

- 중괄호({ })로 전체를 감싸고, 프로퍼티명과 프로퍼티값을 콜론(:)으로 구분하며, 각각의 프로퍼티들을 콤마(,)로 분리해 작성

- 객체 리터럴의 프로퍼티명을 식별자뿐만 아니라 문자열로도 기술할 수 있음

```
var point = { “x” : 300, “y” : 200 };              // (x, y) 좌표 객체
var rectangle = {
      topLeft : { x : 300, y : 200 },  bott        // 좌측상단 (x, y) 좌표 객체
      omRight : { x : 350, y : 280 },  siz         // 우측하단 (x, y) 좌표 객체
      e : function() {                             // 면적 계산 메소드
        return (this.bottomRight.x - this.topLeft.x) * (this.bottomRight.y - this.topLeft.y);
      }
};
```

객체 리터럴 - 실습 02-07

함수 리터럴

함수

- 어떤 입력 값을 이용해 미리 정의된 로직을 처리하여 결과를 반환하는 구조

자바스크립트에서 함수

- 객체 프로퍼티에도 할당될 수 있는 실행가능한 코드 값

- 객체 프로퍼티에 저장된 함수를 객체의 메소드라고 부름

- 함수도 데이터 타입의 하나로 다룰 수 있기 때문에 변수, 배열, 객체 프로퍼티에 저장할 수 있으며, 다른 함수의 전달인자로 넘겨줄 수도 있음

함수 리터럴 (계속)

함수 리터럴

- function 키워드에 이어 소괄호(( )) 안에 함수 호출 시 전달하는 인수들 (arguments)에 의해 최기화되는 매개 변수(parameters) 목록, 마지막으로 함수 의 몸체 중괄호({ }) 안에 함수가 수행해야 할 실행코드와 결과 값을 반환하는 return 문으로 구성됨

- 만일 반환 값이 없는 return 문을 기술하거나, return 문 자체가 없으면 undefined 값을 반환함

```
var add = function(op1, op2) {
      return op1 +op2;
};
```

함수 리터럴 - 실습 02-08

불리언 리터럴

불리언 리터럴: 참/거짓의 진리 값 두 가지를 표현

- true 또는 false 값을 가짐
- 불리언 값은 일반적으로 비교의 결과로 생성되며, 제어 구조 내에서 조건을 판단하는 곳에 주로 사용됨
- 불리언 값은 산술연산에서 true는 숫자 1로, false는 숫자 0로 변환되며, 문자열 연산에서 true는 문자열 “true”로, false는 문자열 “false”로 변환되고, 논리연산에서 0, undefined, null, NaN, “” 값은 false로 변환되어 평가됨

```
var isOpened = false;  // isOpend는 false 값을 가진 불리언 변수가 됨
!isOpend sc            // isOpend가 false 일 때 !(not 연산)에 의해 전체 식의 결과가 true가 됨
ore >= 70              // score의 값이 70 이상이면 이 식은 true로 평가됨
```

불리언 리터럴 - 실습 02-09

undefined와 null

undefined

- 변수가 선언은 되었지만 값이 할당된 적이 없는 변수에 접근하거나, 존재하지 않는 객체 프로퍼티에 접근할 경우 반환되는 값
- 논리 연산에서 false로, 산술 연산에서는 NaN로, 문자열 연산에서는 “undefined”로 변환되어 연산됨

null

- 예약어
- 보통 참조 타입과 함께 쓰여, 어떠한 객체도 나타내지 않는 특수한 값으로 사용
- 논리 연산에서 false로 변환되며, 산술 연산에서는 0으로, 문자열 연산에서는 “null”로 변환되어 연산됨

undefined와 null

```
var a;              // 변수 선언 시 명시적인 초기화 할당 값이 없을 때 undefined값이 할당
var obj = { };
obj.prop; o     // obj 객체 내에 prop 프로퍼티는 존재하지 않으므로 undefined를 반환
bj = null;        // null값을 이용해 객체 참조를 제거
```

undefined와 null - 실습 02-10

3강. 연산자, 제어문

산술연산자

연산자	의미	특징
+	덧셈연산	피연산자가 숫자일 경우 덧셈 연산을 행합니다. 피연산자가 불리언일 경우 true는 1로, false는 0으로 변환 되어 덧셈연산을 행합니다. 피연산자의 한 쪽이 문자열일 경우 나머지 피연산자도 문자열 로 변환되어 문자열 접합연산을 행합니다. 피연산자의 한 쪽이 객체일 경우 객체는 문자열로 변환되고, 나머지 피연산자도 문자열로 변환되어 문자열 접합연산을 행 합니다.
-	뺄셈연산	피연산자가 모두 숫자일 경우 뺄셈 연산을 행합니다.
*	곱셈연산	피연산자가 모두 숫자일 경우 곱셈 연산을 행합니다.
/	나눗셈연산	피연산자가 모두 숫자일 경우 나눗셈 연산을 행합니다.
%	나머지연산	피연산자가 모두 숫자일 경우 나머지 연산을 행합니다.

산술연산자

연산자	의미	특징
++	전치증가연산	피연산자의 값을 1 증가시킵니다. 대입연산자와 함께 사용하 면 먼저 피연산자의 값을 1 증가시키고, l-value에 증가된 피 연산자의 값을 대입합니다.
++	후치증가연산	피연산자의 값을 1 증가시킵니다. 대입연산자와 함께 사용하 면 먼저 l-value에 현재의 피연산자의 값을 대입하고, 이 후 피연산자의 값을 1 증가시킵니다.
--	전치감소연산	피연산자의 값을 1 감소시킵니다. 대입연산자와 함께 사용하 면 먼저 피연산자의 값을 1 감소시키고, l-value에 감소된 피 연산자의 값을 대입합니다.
--	후치감소연산	피연산자의 값을 1 감소시킵니다. 대입연산자와 함께 사용하 면 먼저 l-value에 현재의 피연산자의 값을 대입하고, 이 후 피연산자의 값을 1 감소시킵니다.

산술연산자 - 실습 03-01

대입연산자

연산자	특징	예
=	l-value에 r-value 를 대입	x = 5;
+=	l-value에 r-value를 더한 값을 l-value에 대입	x = 5; x += 3; // 8
-=	l-value에서 r-value를 뺀 값을 l-value에 대입	x = 5; x -= 3; // 2
*=	l-value에 r-value를 곱한 값을 l-value에 대입	x = 5; x *= 3; // 15
/=	l-value를 r-value로 나눈 값을 l-value에 대입	x = 5; x /= 3; // 1.6666666666666667
%=	l-value를 r-value로 나눈 나머지 값을 l-value에 대입	x = 5; x %= 3; // 2

대입연산자

연산자	특징	예
&=	l-value에 r-value를 비트 AND 연산한 값을 l-value에 대입	x = 5; x &= 2; // 0
|=	l-value에 r-value를 비트 OR 연산한 값을 l-value에 대입	x = 5; x |= 2; // 7
^=	l-value에 r-value를 비트 XOR 연산한 값을 l-value에 대입	x = 5; x ^= 2; // 7
<<=	l-value에 r-value 만큼 좌측 시프트한 값을 l-value에 대입	x = 5; x <<= 2; // 20
>>=	l-value에 r-value 만큼 우측 시프트한 값을 l-value에 대입	x = 5; x >>= 2; // 1
>>>=	l-value에 r-value 만큼 부호없는 우측 시프트한 값을 l-value 에 대입	x = 5; x >>>= 2; // 1

비교연산자

연산자	특징	예

==	좌우 표현식의 평가가 동일할 경우 true
	반환 undefined와 null을 동등하다고
	평가
	문자열과 숫자를 비교할 경우 숫자를 문자열로 변환 후 평가
	숫자와 불리언 값을 비교할 경우 true는 1로, false는 0으로
	변환 후 평가
	객체를 숫자 또는 문자열과 비교할 경우 객체의
	valueOf() 또는 toString() 변환 값인 기본 타입 값으로 평
	가
		5 == 5 // true
		5 == '5' // true
		true == 1 // true

!=	좌우 표현식의 평가가 다를 경우 true 반환	5 != 3 // true
<	좌측 표현식의 결과가 우측 표현식의 결과보다 작은 경우 true 반환	5 < 3 // false
<=	좌측 표현식의 결과가 우측 표현식의 결과보다 작거나 같은 경우 true 반환	5 <= 3 // false

비교연산자

연산자	특징	예
>	좌측 표현식의 결과가 우측 표현식의 결과보다 큰 경우 true 반환	5 > 3 // true
>=	좌측 표현식의 결과가 우측 표현식의 결과보다 크거나 같은 경우 true 반환	5 >= 3 // true
===	좌우 표현식의 평가가 동일하고, 데이터 타입도 같을 경우 true 반환 == 와 다르게 타입 변환을 하지 않음	5 === '5' // false
!==	좌우 표현식의 평가가 다르거나, 혹은 데이터 타입이 다른 경우 true 반환 == 와 다르게 타입 변환을 하지 않음	5 !== '5' // true

비교연산자 - 실습 03-03

논리연산자

연산자	특징	예
&&	좌우 표현식의 평가가 모두 true인 경우 true 반환 좌측 표현식의 평가가 false일 경우 우측 표현식은 실행되 지 않음	true && true // true true && false // false false && true // false false && false // false
||	좌우 표현식의 평가 중 어느 하나가 true인 경우 true 반 환 좌측 표현식의 평가가 true일 경우 우측 표현식은 실행되 지 않음	true || true // true true || false // true false || true // true false || false // false
!	표현식의 평가가 false일 경우 true를, true인 경우 false 를 반환	!true // false !false // true

논리연산자 - 실습 03-04

비트연산자

연산자	특징	사용 예
&	좌우 표현식의 평가 비트가 모두 1인 경우 1 반환	5 & 2 // 0
|	좌우 표현식의 평가 비트 중 비트 하나가 1인 경우 1 반환	5 | 2 // 7
^	좌우 표현식의 평가 비트가 서로 다른 비트인 경우 1 반환	5 ^ 2 // 7
~	표현식의 평가 비트를 반전	~5 // -6
<<	표현식의 평가 비트를 좌측으로 시프트	5 << 2 // 20
>>	표현식의 평가 비트를 우측으로 시프트	5 >> 2 // 1
>>>	표현식의 평가 비트를 우측으로 시프트하면서 좌측을 0으로 채움	-5 >>> 2 // 1073741822

기타연산자

연산자	특징	예
,	좌측 피연산자부터 우측 피연산자로 피연산자를 계속 해서 평가	var x = 1, y = 2, z =3;
delete	피연산자로 지정된 객체의 프로퍼티나 배열의 원소를 삭제	delete arr[2]; // true 혹은 false 반환
instanceof	피연산자의 객체 타입 조사	arr instanceof Array // true 혹은 false 반환
new	새로운 객체를 생성하고, 이를 초기화하기 위한 생성 자 함수를 호출	var arr = new Array(); // 참조값을 반환
typeof	피연산자의 데이터 타입을 문자열로 반환	typeof '문자열' // string
void	피연산자의 값을 무시하고 undefined를 반환	void 0
?:	조건식 ? 식1 : 식2. 조건식의 결과가 true일 경우 식 1을, false일 경우 식2를 평가	(score >= 70) ? '합격' : '불합격'

기타연산자 - 실습 03-05

if 문

if 문은 조건표현식의 평가에 따라 특정 문장을 실행할 수 있는 기능을 제공하는 기본적인 분기 제어문

```
if (조건표현식)
    문장;
```

```
if (조건표현식) {
    문장#1;
    ...
} else {
    문장#2;
    ...
}
```

if 문

```
if (조건표현식#1) {
   문장#1;
   …
} else if (조건표현식#2) {
   문장#2;
   …
} else {
   문장#3;
   …
}
```

if 문에서 사용되는 조건들은 상호 배타적

if 문 - 실습 03-06

switch 문

switch 다음에 오는 () 안에는 주로 변수가 오지만, 반환 값을 제공하는 어떤 표현식이든 올 수 있음

switch의 블록({ })에는 여러 개의 case 문과 한 개의 선택적 default 문이 사용됨

표현식이 반환하는 값에 의해 case 문이 실행되며, 실행된 case문에 break 문이 없을 경우 switch 블록을 빠져나가지 못하고, 다음 case 문들도 실행됨에 유의

switch 문

```
switch (표현식) {
  case 값#1:
    문장 #1;
    break;
  case 값#2:
    문장 #2;
    break;
  case 값#3:
    문장 #3;
    break;
  default:
    문장 #4;
    break;
}
```

switch 문 - 실습 03-07

&&와 ||를 이용한 조건문

논리연산자 &&와 ||의 짧은 논리 회로 연산자(Short Circuit Logical Operator)의 성격을 이용

&& 연산자의 경우 좌측 피연산자의 평가 값을 true로 변환할 수 있으면 우측 피연산자의 표현식을 평가 할 수 있어 if 문과 같은 기능을 수행

|| 연산자의 경우 좌측 피연산자의 평가 값이 false로 변환할 수 있는 경우 우측 피연산자를 평가해 값을 반환

&&와 ||를 이용한 조건문 - 실습 03-08

while 문

while 문은 기본적인 반복문으로, 표현식이 true로 평가되면 몸체에 해당하는 블록 내의 문장들을 실행

문장을 모두 실행한 뒤 다시 표현식을 평가하는데 이 때 false로 평가되면 while 문이 종료되고 프로그램의 다음 문장이 실행

주의할 점은 평가식을 false로 만들 수 있는 종료 조건을 만드는 것

종료 조건을 만들지 않으면 while 문은 무한 루프에 빠짐

```
while (표현식) {
   문장들
}
```

while 문 - 실습 03-09

do...while 문

do...while 문은 while 문과 비슷한 성격을 가진 반복문으로, 표현식이 true로 평가되면 몸체에 해당하는 블록 내의 문장들을 실행

최소한 한번은 블록 내의 문장들을 실행

do...while 문 끝에 세미콜론(;)이 붙임

문장을 모두 실행한 뒤 다시 표현식을 평가하는데 이 때 false로 평가되면 do...while 문이 종료되고 프로그램의 다음 문장이 실행

```
do {
   문장들
} while (표현식);
```

do...while 문 - 실습 03-10

for 문

for 문은 초기식을 한 번만 실행

조건식의 평가 결과가 true일 경우 블록 내의 문장 실행

블록 내 문장 실행이 완료되면 증감식이 실행

다시 조건식의 평가 결과에 따라 for 문의 계속 혹은 종료 결정

```
for (초기식; 조건식; 증감식) {
    문장들
}
```

for 문 - 실습 03-11

for...in 문

for...in 문은 객체의 프로퍼티나 배열의 원소에 대해 순서대로 반복처리를 실행

변수에는 객체로부터 취한 프로퍼티명이 저장되거나, 배열로부터 취한 인덱스 번호가 저장

```
for (변수 in 객체 혹은 배열) {
   문장들
}
```

for...in 문 - 실습 03-12

예외란

예외란 예외적인 상황이나 에러가 발생했음을 나타내는 객체

자바스크립트는 런타임에서 에러가 발생할 때마다 예외를 발생시킴.

프로그램에서 throw 문을 사용하여 명시적으로 예외를 발생시킬 수도 있음

throw 문에서 사용하는 표현식의 결과 타입은 대부분 Error 객체 혹은 Error 객체를 상속받은 객체이지만, 에러메시지를 담은 문자열이나 에러코드를 나타내는 숫자값도 유용하게 사용

```
throw 표현식;
throw new Error('에러메시지');
```

try...catch...finally

try...catch...finally 문은 자바스크립트의 예외처리 기법

예외가 발생하면 자바스크립트 인터프리터는 정상적인 프로그램 실행을 즉시 중단하고 가장 가까운 예외처리기로 처리를 넘김

예외처리기는 try...catch...finally 블록의 catch 절을 사용해 작성

예외를 처리할 try...catch...finally 블록이 없는 함수에서 예외가 발생하면 함수를 호출했던 블록으로 예외가 전파되며, 어떠한 예외처리기도 찾지 못하면이 예외는 에러로 취급

try...catch...finally

```
try {
   // 정상적으로 처리되어야 할 코드들을 기술합니다.
   // 코드 실행 중 런타임에서 에러가 발생하여 예외가 발생하거나,
   // throw 문을 통해 예외를 직접 발생시킬 수도 있으며,
   // 호출한 함수를 통해 예외가 전파될 수도 있습니다.
} catch (e) {
   // 예외가 발생할 경우에만 실행되는 블록으로
   // 예외와 관련된 정보를 ( )에 선언된 변수 e를 통해 참조합니다.
   // 이 블록에서 예외를 처리할 수도 있고
   // 예외를 무시할 수 도 있으며,
   // throw 문을 통해 예외를 다시 발생시킬 수도 있습니다.
} finally {
   // try 블록이 모두 실행 완료되거나,
   // 예외가 발생하여 catch 블록이 실행된 후에도
   // 무조건 실행이 필요한 코드를 이 블록에 기술합니다.
}
```

try...catch...finally - 실습 03-13

4강. 함수

함수의 정의

일반적으로 함수란 어떤 입력 값을 이용해 미리 정의된 로직을 처리하여 결과를 반환하도록 만든, 여러번에 걸쳐 호출에 의해 실행될 수 있는 코드 블록

함수는 매개변수(parameter) 혹은 전달인자(argument)라고 불리는 지역변수를 가질 수 있는데, 이 매개변수는 함수를 호출하는 시점에 값을 갖게 됨

함수는 반환값을 가질 수 있는데, 반환값은 return 문에 의해 함수를 호출한 표현식의 결과로 전달됨

자바스크립트에서 함수의 역할

역할	특징
호출 루틴으서의 함수	호출에 의해 실행되며, 매개변수에 전달된 값을 이용해 미리 정의된 로직을 처리하여 결과를 반환 코드 블록
데이터로서의 함수	변수, 객체 프로퍼티, 배열원소에 저장될 수 있고, 매개변수의 전달 값 또는 다른 함수의 반환 값으로도 사용이 가능
메소드로서의 함수	객체 프로퍼티에 저장되어 객체를 통해 호출하는데 사용
생성자 함수	new 연산자와 함께 사용하여 객체를 생성하고 초기화하는데 사용

함수를 정의하는 3가지 방법

function 문을 이용해 함수를 정의

```
function 함수명([매개변수 목록]) {
   문장들
   [return 반환값;]
}
```

함수 리터럴을 이용해 함수를 정의

```
var 변수명 = function([매개변수 목록]) {
   문장들
   [return 반환값;]
};
```

함수를 정의하는 3가지 방법

Function 생성자를 이용해 함수를 정의

```
var 변수명 = new Function(['매개변수' [, '매개변수'[,...]]], '문장들');
var 변수명 = new Function('[매개변수 목록]', '문장들');
```

함수를 정의하는 3가지 방법 - 실습 04-01

변수의 전역스코프와 지역스코프

스코프(scope)란 코드에서 변수를 참조할 수 있는 해당 변수의 유효 범위를 결정하는 개념

자바스크립트는 코드 전체에서 접근 가능한 전역(global) 스코프와 함수 내의 코드에서만 접근 가능한 지역(local) 스코프만 존재

전역 스코프를 가진 변수를 전역 변수, 지역 스코프를 가진 변수를 지역 변수라고 함

자바스크립트는 블록({ }) 스코프가 존재하지 않음에 유의

변수의 전역스코프와 지역스코프 - 실습 04-02

함수의 매개변수와 스코프

함수의 매개변수는 지역스코프를 갖는 지역변수로, 함수를 호출하는 표현식으로부터 그 값을 전달 받음

전역스코프의 전역변수 값을 함수의 매개변수로 전달할 때 그 값이 함수의 지역스코프의 매개변수로 복사됨

이 전달된 값이 기본형일 경우 전역변수와 매개변수는 상호 독립적인 값이 되며, 참조값이 전달된 경우 전역변수와 매개변수가 동일 객체를 참조하게 됨

함수의 매개변수와 스코프 - 실습 04-03

함수의 매개변수와 인자

자바스크립트 함수는 함수 정의 시 선언된 매개변수에 상관없이 인자를 전달할 수 있으며, 어떤 데이터 타입의 값이라도 전달할 수 있음

선언된 매개변수보다 적은 수의 인자 값들을 전달하면, 값을 전달받지 못한 남은 매개변수들은 undefined 값을 갖게 됨

생략이 가능한 매개변수는 매개변수 목록의 끝에 위치하도록 하여 임의로 생략할 수 있게 만드는 것이 중요

함수의 매개변수와 인자 - 실습 04-04

함수와 명시적인 이름을 가진 인자 전달

함수 호출 시 매개변수를 객체로 전달하여, 메소드 내에서 전달된 객체의 프로퍼티를 인자로 활용하는 방법

코드의 가독성을 더욱 높일 수 있으며, 매개변수의 순서에 상관없이 값을 자유롭게 전달할 수 있고, 생략된 매개변수를 명확히 표현할 수 있는 특징이 있음

함수와 명시적인 이름을 가진 인자 전달 - 실습 04-05

함수에 인자로 함수를 전달

자바스크립트에서 함수는 데이터의 속성을 갖고 있어서, 변수, 객체의 프로퍼티에도 저장될 수 있으며, 함수의 전달 인자로도 사용될 수 있음

함수에 인자로 함수를 전달 - 실습 04-06

함수에 함수 리터럴을 전달하는 익명함수

함수 리터럴을 이용해 정의한 함수를 이름 없는 함수, 즉 익명함수(unnamed function)라고 부름

이름이 없기 때문에 주로 변수, 객체 프로퍼티에 저장하거나, 함수의 인자 혹은 반환 값으로 주로 사용

함수에 함수 리터럴을 전달하는 익명함수 - 실습 04-07

중첩함수

자바스크립트에서 함수는 다른 함수 안에 중첩되어 정의될 수 있음

중첩함수(inner function)는 함수 내에서만 호출할

수 있으며, 함수 외부에서는 직접 호출할 수 없음

이런 특징으로 특정 함수에서만 필요한 기능을 외부에 노출시키지 않고
구현할 수 있으며, 함수 내부에 선언된 변수에 접근할 수 있기 때문에
객체지향의 정보은닉이라는 특징을 구현하는데 사용될 수 있음

중첩함수 - 실습 04-08

스코프 체인

지역 스코프는 함수 단위로 관리되며, 이 지역 스코프를 관리하는 객체를 호출 객체

함수의 매개변수, 지역변수가 호출 객체의 프로퍼티

전역 스코프를 나타내는 객체를 전역 객체, 루트(root) 객체라고함

전역 변수와 전역 함수는 전역 객체의 프로퍼티와 메소드

var 키워드 없이 변수를 선언하면 전역 객체에 등록되어 전역 변수가 되므로, 함수의 지역 변수로 선언하기 위해선 반드시 var 키워드를 사용해 변수를 선언해야 함

스코프 체인는 전역 객체와 함수 호출 시 생성된 호출 객체를 생성 순서대로 연결한 리스트 함

수는 함수가 호출되는 시점을 기준으로 스코프 체인에 연결되어 있는 모든 것들에 접근 가능

스코프 체인 - 실습 04-09

콜백함수

콜백함수는 직접 호출하는 함수가 아닌 어떤 특정 시점이나 조건을 만족했을 때 호출될 수 있도록 라이브러리 함수의 인자로 전달되는 함수를 말함

콜백함수 - 실습 04-10

비공개 속성/함수를 만들 수 있는 함수 클로저

클로저 : 실행될 코드와 함수의 유효 범위, 다시 말해 함수의 호출 객체와 연결된 스코프 체인의 조합

함수의 지역변수에 대한 중첩함수를 만들면 비공개 속성과 접근자 메소드를 구현해 객체지향의 정보은닉을 실현할 수 있음

비공개 속성/함수를 만들 수 있는 함수 클로저 - 실습 04-11

5강. 객체

객체의 정의

자바스크립트의 객체는 이름과 값으로 구성된 프로퍼티들의 집합, 이름이 붙어 있은 데이터 값들의 모음이며, 복합 타입

자바스크립트의 모든 객체는 Object 생성자 함수를 상속하며, 공통으로 Object에서 상속받은 프로퍼티와 메소드가 있음

생성자: 객체를 정의하는 방법

객체 리터럴을 사용

```
var empty = { };
var point = { x : 20, y : 30 };
var student = {
 name : '홍길동',
 age : 20,
 phone : '010-1234-5678',
 toString : function() {
   return '{ name : ' +this.name +', age : ' +this.age + ', phone : ' +this.phone +' }';
 }
};
```

생성자: 객체를 정의하는 방법

new 연산자와 Object() 생성자 함수 호출를 이용해 객체를 만들고, 그 객체에 프로퍼티를 추가

Object() 생성자 함수를 사용하면 객체 리터럴 { }와 같이 빈 객체를 생성할 수 있음

```
var empty = new Object();
var point = new Object();
point.x = 20;
point.y = 30;
```

생성자: 객체를 정의하는 방법

```
var student = new Object();
student.name = '홍길동'; stu
dent.age = 20;
student.phone = '010-1234-5678';
student.toString = function() {
 return '{ name : ' +this.name +', age : ' +this.age +
        ', phone : ' +this.phone +' }';
};
```

생성자: 객체를 정의하는 방법

원하는 타입의 객체를 생성하고 초기화하기 위해서 필요한 것이 생성자 함수를 직접 정의하는 것

생성자 함수는 객체를 만드는 틀

생성자의 이름은 new 연산자와 생성자 함수 호출을 통해 만들게 될 객체의 타입을 분명하게 나타내는 것이 좋음

생성자의 이름 첫문자를 대문자로 하는 이유도 다른 함수와 구별하기 위함

생성자 함수의 역할은 this 키워드가 나타내는 객체를 초기화할 뿐 다른 역할은 수행하지 않는 것을 원칙으로 함

생성자 함수의 구현 시 return 문은 사용하지 않음

생성자: 객체를 정의하는 방법

```
function Student(n, a, p) {
 this.name = n;
 this.age = a;  t
 his.phone = p;
 this.toString = function() {
   return '{ name : ' +this.name +', age : ' +this.age +
         ', phone : ' +this.phone +' }';
 }
}
```

new 연산자, 생성자 함수, 그리고 this

new 연산자가 생성하는 객체에는 아무런 프로퍼티도 정의되어 있지 않음

객체 생성 이후 new 연산자는 지정된 생성자 함수를 호출하여 명시된 인자들을 전달하고 방금 생성된 새 객체도 this 키워드를 통해 전달함

생성자 함수는 이 this 키워드를 사용하여 나름대로의 새 객체를 초기화하는데 이 때 this는 생략할 수 없음

생성자 함수를 이용한 객체 생성

자바스크립트가 제공하는 기본 생성자 함수들이 아닌 사용자 정의 생성자 함수를 이용해 객체를 생성하는 방법도 기존의 방법과 동일

```
var student1 = new Student('홍길동', 20, '010-1234-5678');
var student2 = new Student('이순신', 40, '010-2468-1357');
```

생성자 함수를 이용한 객체 생성 - 실습 05-01

객체의 멤버

객체의 프로퍼티와 메소드가 객체의 멤버

객체의 멤버는 일반적으로 생성자 함수 내에서 this 키워드를 이용해 추가되며, this 키워드는 new 연산자에 의해 새로 생성된 객체를 나타냄 자바스크립트에서 객체의 멤버는 런타임 시 동적으로 추가와 삭제가 가능하므로 주의가 필요

객체의 멤버

```
var rect1 = new Rectangle(100, 120, 20, 30);
var rect2 = new Rectangle(250, 300, 30, 50);

rect1.area = function() {
 return this.width * this.height;
};

document.writeln('rect1.area(): ' +rect1.area() +'<br/>'); document.writel
n('rect2.area(): ' +rect2.area() +'<br/>');  // TypeError 발생
```

멤버 접근

특정 객체를 통하지 않고 생성자 함수를 통해서 접근되는 프로퍼티나 호출되는 메소드가 필요할 경우도 있음

Math() 생성자 함수의 모든 메소드는 생성자 함수를 통해 호출됨

이러한 멤버 접근 방법은 생성자 함수의 프로퍼티나 메소드를 만들면 전역변수나 전역함수를 사용하는 것보다 이름 충돌의 문제를 해결해줄 수 있다는 점에서 유용

```
Math.random() ;       // 0 ~ 1 사이의 수를 반환합니다.
Math.round(1.58);    // 2를 반환합니다.
Math.PI;                  // 원주율을 반환합니다.
```

멤버 접근 - 실습 05-02

프로토타입 객체

자바스크립트의 모든 객체는 프로토타입이라 불리는 객체를 내부적으로 참조

프로토타입 객체의 실체는 new 연산자와 Object 생성자 함수 호출을 통해 생성한 Object 객체

객체는 프로토타입 객체에 있는 프로퍼티를 상속

프로토타입 객체를 이용하면 좀 더 효율적으로 메소드를 추가할 수 있음

prototype 프로퍼티

new 연산자는 빈 객체를 생성하고, 해당 객체의 prototype 프로퍼티를 생성자 함수의 prototype 프로퍼티 값을 이용해 설정

모든 함수에는 prototype 프로퍼티가 있으며, 함수가 정의될 때 생성되고 초기화됨

프로토타입 객체는 생성자 함수와 연결되고, 이 생성자 함수를 통해서 생성되는 객체들은 생성자 함수와 연결된 프로토타입 객체의 프로퍼티들을 똑같이 상속

프로토타입 객체가 메소드나 상수와 같은 프로퍼티들을 위치시키기에 적합한 곳

prototype 프로퍼티

프로토타입 객체에 의한 상속을 통해 메모리 사용량을 줄일 수 있으며, 프로토타입 객체에 프로퍼티가 변경되거나 추가되어도 기존 객체들 역시 변경되거나 추가된 프로퍼티를 바로 사용할 수 있다는 장점이 있음

```
Rectangle.prototype.area = function() {
 return this.width * this.height;
};
```

prototype 프로퍼티 - 실습 05-03

constuctor 속성

자바스크립트의 모든 객체는 객체를 초기화하는데 사용하는 생성자 함수를 참조하는 constructor 프로퍼티를 가지고 있음

constructor 프로퍼티를 이용해 객체의 타입을 판단할 수 있으며, 객체 생성도 가능

constuctor 속성 - 실습 05-04

네임스페이스

자바스크립트는 네임스페이스 구조를 지원하지 않음

빈 객체를 이용해 네임스페이스와 같은 기능을 제공할 수 있음

네임스페이스를 정의하기 위해 구현 코드가 없는 생성자 함수를 작성하고, 생성자 함수에 프로퍼티를 추가하는 것과 동일한 방법으로 하위의 생성자 함수를 정의

네임스페이스를 사용할 경우 하위 생성자 함수를 이용해 객체를 생성할 경우, 네임스페이스를 포함한 FQN#으로 생성자 함수를 호출해야 함에 유의

6강. 상속

객체 생성 과정

① new 연산자는 빈 객체를 생성합니다.
② 생성자 함수는 this 키워드를 통해 전달된 새로운 객체에 생성자 함수 내에 작성된 프로퍼티 혹은 메소드를 추가하는 초기화 작업을 수행합니다.
③ 새로운 객체의 prototype 프로퍼티에 생성자 함수의 prototype 프로퍼티 값이 전달되어 객체와 생성자 함수는 동일한 프로토타입 객체를 참조하게 됩니다.
④ this가 가리키는 객체를 반환합니다.

프로토타입 체인과 프로토타입 멤버 상속

자바스크립트는 프로토타입 기반의 상속 메커니즘을 제공

객체를 생성하면 객체의 생성자 함수의 프로토타입 객체에 정의된 멤버를 상속

객체의 생성자 함수의 프로토타입 객체와 Object() 생성자 함수의 프로토타입 객체가 연결되어 있음

객체를 거쳐 프로토타입 객체로 멤버를 검색할 수 있는 연결을 프로토타입 체인(prototype chain)이라 함

멤버에 대한 검색은 맨 먼저 객체를 대상으로 수행하고, 만일 해당 멤버를 찾지 못하면 프로토타입 객체를 다음으로 검색하고, 마지막으로 Object 생성자 함수의 프로토타입 객체까지 검색하게 됨

Object 객체의 주요 멤버

멤버	설명
constructor	객체 생성자 함수를 참조하는 프로퍼티
toString()	객체의 문자열 표현을 반환
valueOf()	객체의 기본형 표현을 반환
hasOwnProperty(prop)	객체가 직접 prop 프로퍼티를 갖고 있는가? (프로토타입 객체의 프로퍼티는 false를 반환)
propertyIsEnumerable(prop)	for...in 문으로 prop 프로퍼티/메소드를 열거할 수 있는가?
isPrototypeOf(obj)	호출 객체가 obj 객체의 프로토타입인가?

Finction 객체의 주요 멤버

멤버	설명
apply(obj, [arg1 , arg2 , … argn])	첫번째 매개변수 obj는 객체를 전달해 함수 내부의 this에 할당되고, 두번째 매개변수는 함수의 인자들을 저장한 배열 객체로 호출 함수에 전달됩니다. 만 일 obj 값이 null인 경우 전역 객체가 this에 할당됩니다.
call(obj, arg1, arg2, … argn)	첫번째 매개변수 obj는 객체를 전달해 함수 내부의 this에 할당되고, 두번째 매개변수부터는 함수의 인자들로 호출 함수에 전달됩니다. 만일 obj 값이 null인 경우 전역 객체가 this에 할당됩니다.
toString()	함수를 정의하는 코드를 반환

프로토타입 체인과 프로토타입 멤버 상속

모든 함수는 Function 객체의 프로토타입 멤버를 상속하고, 모든 함수의 프로토타입 객체는 Object 객체의 프로토타입 객체를 상속하며, 모든 생성자 함수의 객체는 Object 객체의 프로토타입 객체를 상속함

프로토타입 멤버 상속은 하위 생성자 함수의 prototype 프로퍼티가 new 연산자와 상위 생성자 함수 호출을 통해 생성된 객체를 참조해서 구현하며, 상위 생성자 함수의 프로토타입 멤버를 상속하게 됨

프로토타입 객체가 상위 생성자 함수를 통해 만들어졌기 때문에 constructor 프로퍼티는 상위 생성자 함수를 참조하고 있어 constructor 프로퍼티를 하위 생성자 함수로 변경함

프로토타입 체인과 프로토타입 멤버 상속 - 실습 06-01

객체 멤버 상속

프로토타입 멤버 뿐만 아니라 특정 객체의 멤버를 상속해야 하는 경우 Function 객체의 메소드 apply() 혹은 call()을 사용해 생성자 체이닝(constructor chaining)을 구현

상위 생성자 함수의 멤버를 new 연산자와 하위 생성자 함수를 통해 생성할 객체의 멤버로 추가하는 객체 멤버 상속을 수행함

하위 생성자 함수의 프로토타입 객체가 new 연산자와 상위 생성자 호출을 통해 생성된 객체가 되는데 이 객체 내에는 이미 객체 멤버 상속을 통해 프로퍼티를 가지고 있으므로 프로 토타입 객체의 프로퍼티를 제거하는 과정이 필요함

객체 멤버 상속 - 실습 06-02

객체의 타입 검사

자바스크립트는 타입에 대한 제약이 약해 직접 타입을 검사하는 방법을 익힐 필요가 있음

특히 프로토타입 상속을 한 사용자 정의 객체를 사용할 때는 typeof 연산자, instanceof 연산자, constructor 프로퍼티, toString() 메소드 등의 도움을 얻어 판단할 필요가 있음

객체의 타입 검사 - 실습 06-03

7강. 내장객체

Array

Array 객체는 배열을 다루기 위한 객체로, 배열의 원소에 대한 추가, 삭제, 정렬 등의 조작 기능을 제공

new 연산자와 생성자 함수 호출을 이용해 Array 객체를 생성할 수 있지만, 배열 리터럴로도 많이 사용함

자바스크립트에서 배열은 크기가 자동으로 증가되는 특징을 가지고 있음

```
var students = new Array('홍길동', '이순신', '강감찬');
var students = new Array();
var students = new Array(10);
var students = ['홍길동', '이순신', '강감찬'];
var students = [ ];
```

Array

멤버	설명
concat(arr)	매개변수로 전달된 배열 객체를 현재 배열 객체에 연결
join(del)	배열의 원소를 구분문자열 del로 연결해 문자열을 반환
slice(start [,end])	start 인덱스에서 end-1 인덱스의 원소로 배열을 생성하여 반환
splice(start, cnt [,rep [,...]])	start 인덱스에서 cnt 개의 원소를 rep,...로 치환
pop()	배열 끝의 원소를 꺼내 배열에서 삭제
push(data1 [,data2,...])	배열 끝에 원소를 추가
shift()	배열 처음에서 원소를 꺼내 삭제
unshift(data1 [,data2,...])	배열 처음에 원소들을 추가
reverse()	인덱스의 역순으로 배열을 정렬
sort([func])	기본 오름차순으로 정렬. func은 두 인자를 가지며, 두 인자가 같을 때 0을, 첫번째 인자가 크면 1을, 두번째 인자 크면 -1을 반환하는 콜백함수
length	배열의 크기
toString()	[원소1, 원소2, ...] 형식의 문자열 반환

Array - 실습 07-01

String

String 객체는 문자열을 다루기 위한 랩퍼(wrapper) 객체로, 문자열의 검색, 부분 문자열의 추출, 변환 등의 조작 기능을 제공

new 연산자와 생성자 함수를 이용해 String 객체를 생성할 수 있지만 리터럴로 표현하는 것이 일반적임

```
var greeting = new String('안녕하세요');
var greeting = '안녕하세요';
```

String

멤버	설명
indexOf(substr [, start])	start 인덱스로부터 정방향으로 부분문자열 substr과 일치하는 인덱스 반환
lastIndexOf(substr [,start])	start 인덱스로부터 역방향으로 부분문자열 substr과 일치하는 인덱스 반환
charAt(n)	n 번 인덱스의 문자를 반환
slice(start [,end])	문자열에서 start 인덱스에서 end -1 인덱스 사이의 문자를 반환
substring(start [,end])	문자열에서 start 인덱스에서 end - 1 인덱스 사이의 문자를 반환
substr(start [,cnt])	문자열에서 start 인덱스에서 cnt 개의 문자를 반환
split(str [,limit])	문자열을 분할문자열 str로 분할하여 그 결과를 배열로 반환(limt은 최대 분할수)
match(reg)	정규표현 reg로 문자열을 검색, 일치한 부분문자열 반환
replace(reg, rep)	정규표현 reg로 문자열을 검색, 일치한 부분을 rep로 치환

String

멤버	설명
search(reg)	정규표현 reg로 문자열을 검색, 일치한 맨 처음 문자위치 반환
toLowerCase()	소문자로 치환
toUpperCase()	대문자로 치환
concat(str)	문자열 뒤에 문자열 str을 연결한 새로운 문자열을 반환
length	문자열의 길이 반환

String - 실습 07-02

Number

Number 객체는 숫자를 다루기 위한 랩퍼 객체

new 연산자와 생성자 함수를 이용해 Number 객체를 생성할 수 있지만 리터럴로 표현하는 것이 일반적임

```
var num = new Number(100);
var num = 100;
```

Number

멤버	설명
MAX_VALUE	최대값 (생성자 함수의 프로퍼티)
MIN_VALUE	최소값 (생성자 함수의 프로퍼티)
NaN	숫자값이 아님 (생성자 함수의 프로퍼티)
NEGATIVE_INFINITY	음수의 무한대 (생성자 함수의 프로퍼티)
POSITIVE_INFINITY	양수의 무한대 (생성자 함수의 프로퍼티)
toString(n)	n진수 값으로 변환
toExponential(n)	지수형식으로 변환(n은 소수점 이하의 행수)
toFixed(n)	소수점 이하 자리수 n에 맞춰 변환(자리수가 부족한 경우 0으로 채움)
toPrecision(n)	전체 유효 자리수 n에 맞춰 변환(자리수가 부족한 경우 0으로 채움)

Number - 실습 07-03

Math

멤버	설명
abs(n)	n에 대한 절대값 반환
max(n1, n2)	n1과 n2 중 큰 값을 반환
min(n1, n2)	n1과 n2 중 작은 값을 반환
pow(base, n)	base값의 n제곱을 반환
random()	0~1 사이의 난수 반환
ceil(n)	n 이상의 최소 정수를 반환
floor(n)	n 이하의 최대 정수를 반환
round(n)	반올림 값을 반환
SQRT1_2	½의 제곱근을 반환
SQRT2	2의 제곱근을 반환
sqrt(n)	n의 제곱근을 반환
PI	원주율을 반환

Math

멤버	설명
cos(n)	n의 코사인 값을 반환
sin(n)	n의 사인 값을 반환
tan(n)	n의 탄젠트 값을 반환
acos(n)	n의 아크 코사인 값을 반환
asin(n)	n의 아크 사인 값을 반환
atan(n)	n의 아크 탄젠트 값을 반환
E	상용로그 밑에 해당하는 상수 e
LN2	2의 상용로그 값
LN10	10의 상용로그 값
LOG2E	2를 밑으로 한 e의 로그 값
LOG10E	10을 밑으로 한 e의 로그 값
log(n)	n의 상용로그 값
exp(n)	지수함수로 e의 n승 값

Math - 실습 07-04

Date

Date 객체는 리터럴 표현이 존재하지 않으며, new 연산자와 생성자 호출을 통해 객체를 생성해야 함

Date 객체는 1970년 1월 1일 0시 0분 0초를 기준으로 시각을 관리하며, Date 객체의 월 지정값은 0~11임

```
var d = new Date();
var d = new Date('2012/12/10');
var d = new Date(2012, 11, 10, 22, 10, 33, 500);
```

Date

멤버	설명
getFullYear()	년(4자리)
getMonth()	월(0~11)
getDate()	일(1~31)
getDay()	요일(0:일요일 ~ 6:토요일)
getHours()	시(0~23)
getMinutes()	분(0~59)
getSeconds()	초(0~59)
getMilliseconds()	밀리초(0~999)
getTime()	1970년 1월 1일 0시를 기준으로 경과된 밀리초
getTimezoneOffset()	세계협정시와의 시차
setFullYear(y)	년(4자리)
setMonth(m)	월(0~11)
setDate(d)	일(1~31)
setHours(h)	시(0~23)
setMinutes(m)	분(0~59)
setSeconds(s)	초(0~59)
setMilliseconds(ms)	밀리초(0~999)
setTime(ts)	1970년 1월 1일 0시를 기준으로 경과된 밀리초

Date

멤버	설명
parse(str)	문자열을 해석해 1970년 1월 1일 0시를 기준으로 경과된 밀리초 를 반환 (생성자 함수를 통해 호출)
UTC(y, m, d [,h [,mm [,s [,ms]]]])	전달 인자를 이용, 1970년 1월 1일 0시를 기준으로 경과된 밀리 초를 반환 (생성자 함수를 통해 호출)
toGMTString()	그리니치 표준시를 문자열로 반환
toUTCString()	세계협정시를 문자열로 반환
toLocaleString()	지역정보에 따른 날짜와 시각을 문자열로 반환
toDateString()	날짜부분을 문자열로 반환
toTimeString()	시각부분을 문자열로 반환
toLocaleDateString()	지역정보에 따른 날짜부분을 문자열로 반환
toLocaleTimeString()	지역정보 따른 시각부분을 문자열로 반환
toString()	일시를 문자열로 반환

Date - 실습 07-05

JSON

JSON은 자바스크립트에서 객체를 텍스트(plain text)로 표현하는 방식

JSON은 시스템 간의 데이터 교환의 수단으로 사용되고 있는 XML보다 용량이 작고, 파싱에 걸리는 시간이 절약되는 이유로 시스템 간의 데이터 교환에 많이 사용되고 있으며, 웹 환경에 최적화되어 있는 자바스크립트에서 바로 객체화해 사용할 수 있기 때문에 생산성과 편의성 또한 높음

JSON은 숫자, 문자열, 불리언, 배열, 객체를 표현할 수 있음

```
' {
   "name" : "홍길동",
   "address" : "서울특별시 관악구 낙성대동",
   "orderItems" : [
      {"item" : "멈추면, 비로소 보이는 것들", "price" : 15000, "qty" : 1 },
      {"item" : "어떻게 원하는 것을 얻는가?", "price" : 18000, "qty" : 1 }
   ],
   "baroOn" : true
} '
```

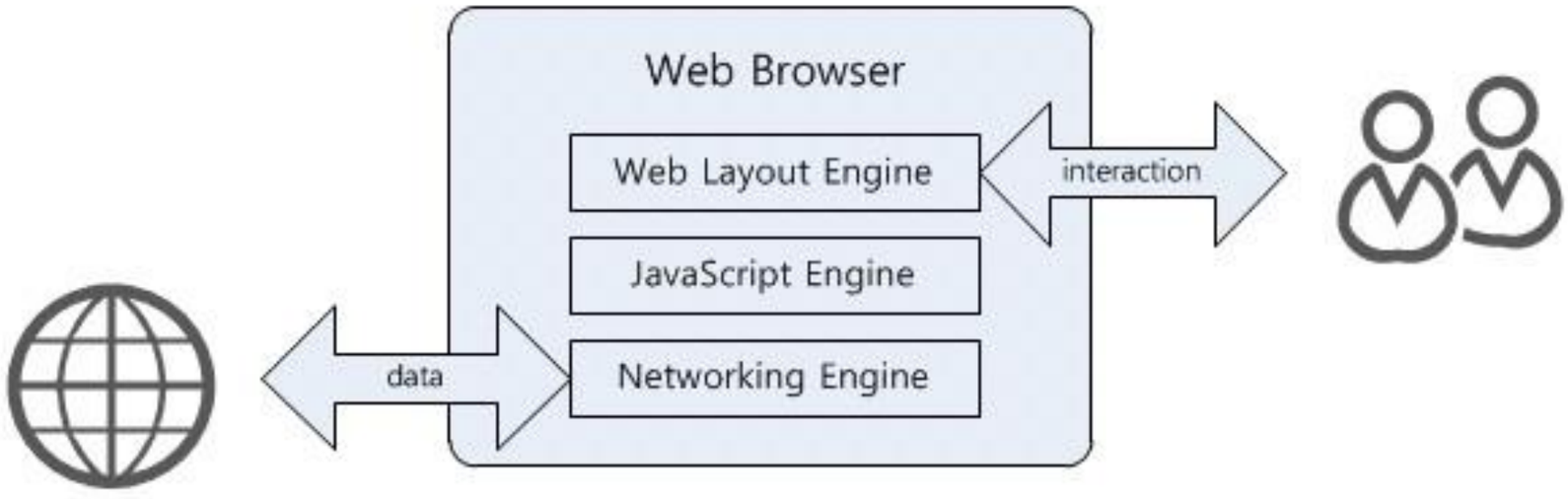
JSON.parse() 함수를 이용해 JSON 문자열을 파싱하고 자바스크립트 객체로 변환

JSON.stringify() 함수를 이용해 자바스크립트 객체를 JSON 문자열로 변환

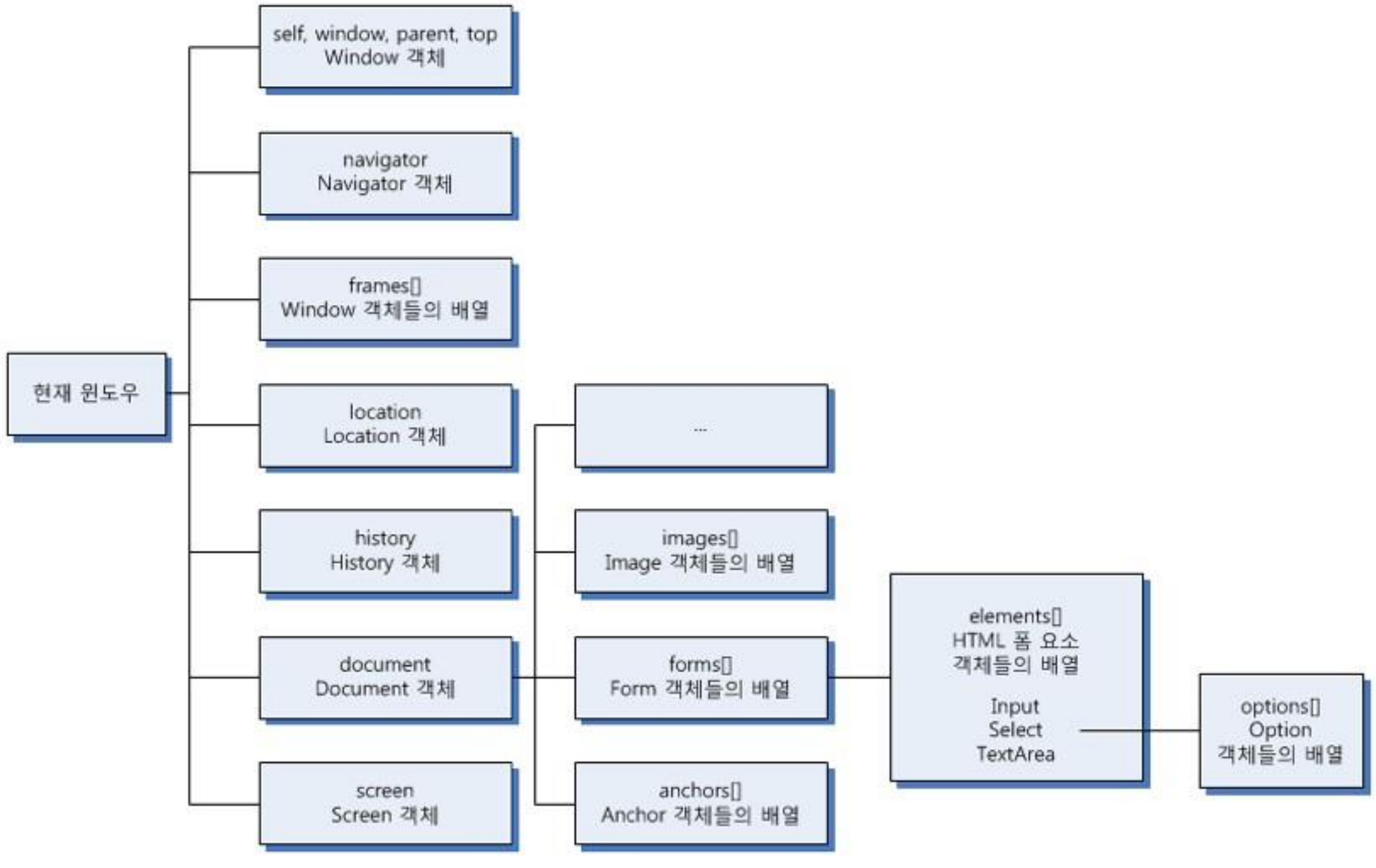
JSON - 실습 07-06

8강. 웹브라우저와 자바스크립트

웹 브라우저와 자바스크립트 실행환경



웹 브라우저 객체


현재 윈도우
self, window, parent, top
Window 객체
navigator
Navigator 객체
frames[]
Window 객체들의 배열
location
Location 객체
history
History 객체
document
Document 객체
screen
Screen 객체
...
images[]
Image 객체들의 배열
forms[]
Form 객체들의 배열
anchors[]
Anchor 객체들의 배열
elements[]
HTML 폼 요소
객체들의 배열
Input
Select
TextArea
options[]
Option
객체들의 배열

웹 브라우저 객체

레벨 0 DOM에서는 클라이언트 측 객체 계층구조를 통해 다양하게 접근

```
...
<body>
<form name="fm">
 이름 : <input type="text" name="name" size="10" />
 <input type="submit" value="송신" />
</form>
</body>
...

document.fm.name.value document.forms[0
].elements[0].value document.forms['fm'].ele
ments['name'].value document['fm']['name'].v
alue
```

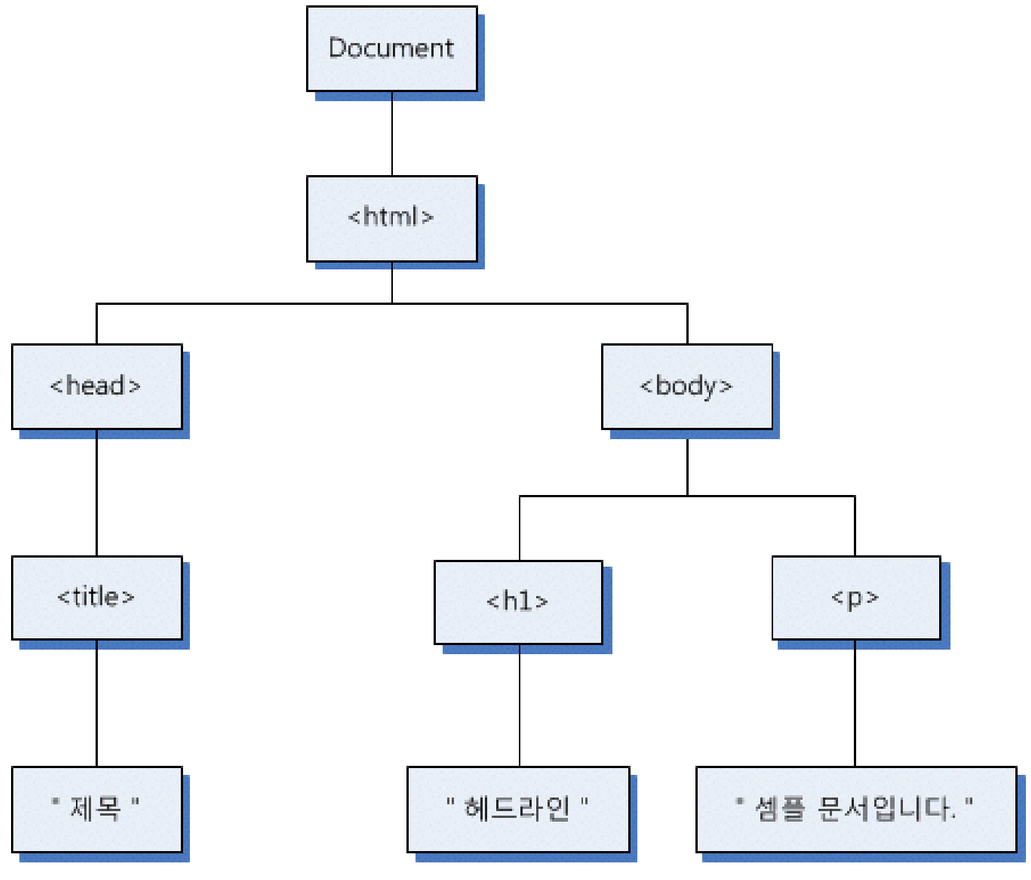
HTML DOM

DOM 표준 레벨 1은 1998년 10월에 표준화되었으며, Node와 Element.Attr, Document 같은 코어 DOM 인터페이스들이 정의되어 있고,
HTML에 특화되어 있는 다양한 인터페이스들도 정의되어 있음

레벨 2는 2000년 11월에 표준화되었으며, 레벨 1을 확장하고, 문서 이벤트와 CSS 관련 표준 API를 정의하고, Core 모듈, HTML 모듈, CSS 모듈, Events 모듈과 같은 모듈화로 구성되었음

HTML DOM

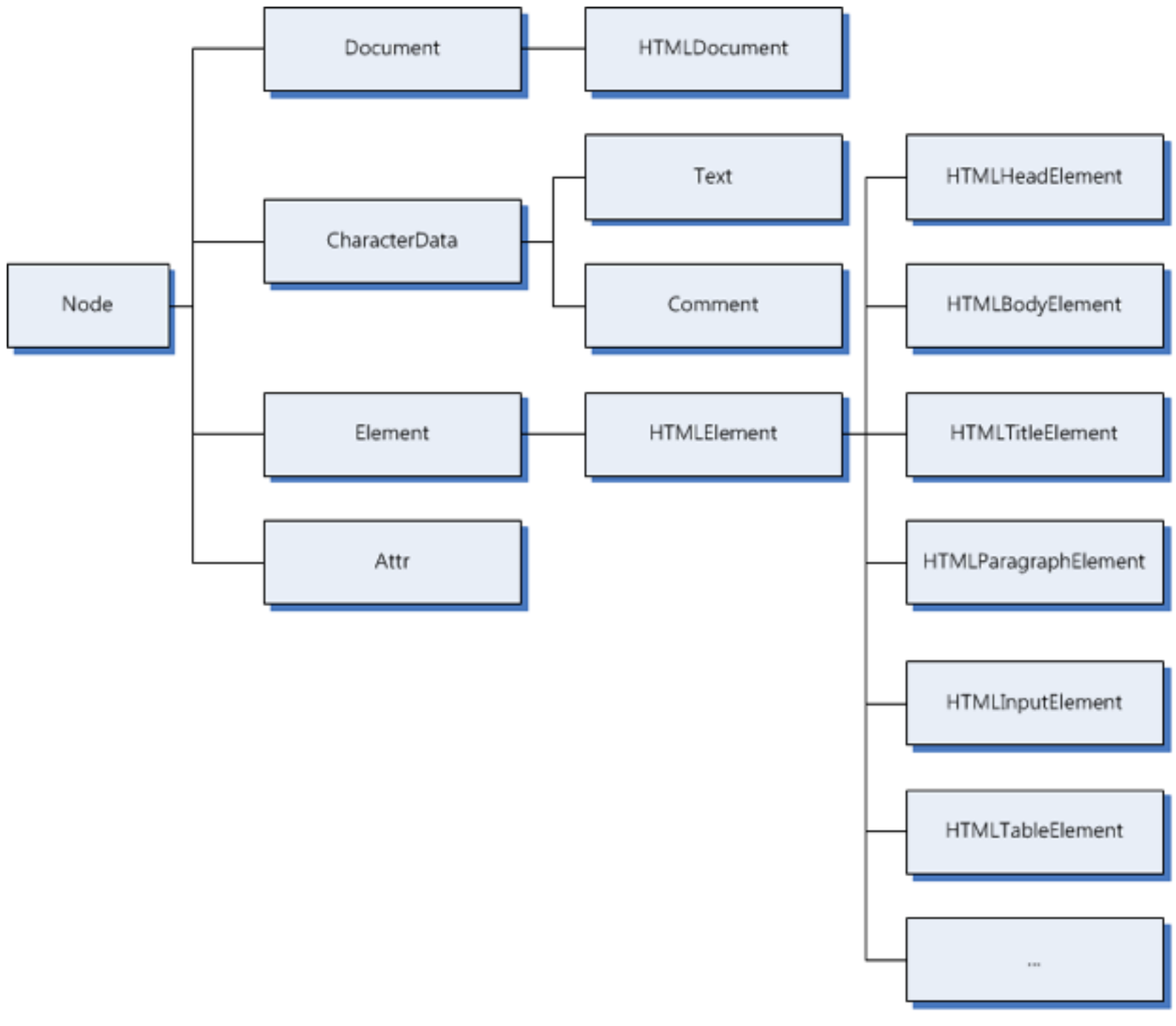
```
<html>
 <head>
   <title>제목</title>
 </head>
 <body>
   <h1>헤드라인</h1>
   <p>
     샘플 문서입니다.
   </p>
 </body>
</html>
```



주요 노드 타입

인터페이스 타입	nodeType 상수	nodeType 값
Element	Node.ELEMENT_NODE	1
Attr	Node.ATTRIBUTE_NODE	2
Text	Node.TEXT_NODE	3
Comment	Node.COMMENT_NODE	8
Document	Node.DOCUMENT	9

DOM API의 인터페이스 계층 구조



DOM HTML API를 이용한 HTML 문서 요소 접근 및 조작

API 용도	멤버	설명
문서 순회	Node.childNodes	Node 객체의 배열 객체처럼 작동하는 NodeList 객체로 자식노드 객체들에 대한 참 조를 가지고 있습니다.
	Node.firstChild	첫번째 자식노드 객체를 참조합니다.
	Node.lastChild	마지막 자식노드 객체를 참조합니다.
	Node.nextSibling	인접한 다음 형제노드 객체를 참조합니다.
	Node.previousSibling	인접한 이전 형제노드 객체를 참조합니다.

DOM HTML API를 이용한 HTML 문서 요소 접근 및 조작

API 용도	멤버	설명
문서 내 요소 찾기	Document.getElementByTagName(tag)	태그명을 문자열 tag로 전달해 해당 태그 들의 NodeList 객체를 반환합니다. NodeList 객체는 배열 객체와 같은 방법 으로 사용할 수 있습니다.
	Document.getElemetById(id)	id 속성이 같은 유일 노드 객체의 참조를 반환합니다.
	HTMLDocument. getElementsByName(name)	name 속성이 같은 요소들의 HTMLCollection 객체를 반환합니다.
문서 수정하기	CharacterData.data	data 프로퍼티에 문자열을 제공해 텍스트 노드의 텍스트를 변경할 수 있습니다.
	Node.appendChild()	자식노드 객체를 추가합니다.
	Node.removeChild()	자식노드 객체를 제거합니다.

DOM HTML API를 이용한 HTML 문서 요소 접근 및 조작

API 용도	멤버	설명
문서에 새로운 내용 추가하기	Document.createElement()	새로운 Element 객체를 생성합니다.
	Document.createTextNode()	새로운 TextNode 객체를 생성 합니 다.
	Node.appendChild()	자식노드 객체를 추가합니다.
	Node.insertBefore()	노드 객체를 앞에 추가합니다.
	Node.replaceChild()	자식노드 객체를 교체합니다.
	Node.innerHTML	DOM 표준은 아니나 사실상의 표준 프로퍼티인 innerHTML에 HTML 텍스트 문자열을 기술하면 간단히 새 로운 내용을 추가할 수 있습니다.

DOM HTML API를 이용한 HTML 문서 요소 접근 및 조작 - 실습 08-01

이벤트와 이벤트 핸들러

이벤트	이벤트핸들러 프로퍼티	발생 시점	대상 요소
abort	onabort	이미지 로딩이 중단되었을 때	img
load	onload	문서, 이미지 로딩이 완료되었을 때	body, img
unload	onunload	다른 페이지로 이동할 때	body
click	onclick	마우스로 클릭할 때	
dbclick	ondbclick	마우스로 더블클릭할 때	
mousedown	onmousedown	마우스 버튼을 눌렀을 때	
mouseup	onmouseup	마우스 버튼을 놓았을 때	
mousemove	onmousemove	마우스의 포인터를 이동하였을 때	
mouseout	onmouseout	요소로부터 마우스가 벗어날 때	
mouseover	onmouseover	요소 안으로 마우스 포인터가 들어왔을 때	
contextmenu	oncontextmenu	컨텍스트 메뉴가 표시되기 전	body

이벤트와 이벤트 핸들러

이벤트	이벤트핸들러 프로퍼티	발생 시점	대상 요소
keypress	onkeypress	키를 눌렀을 때	form 요소, body
keydown	onkeydown	키를 누르고 있을 때	form 요소, body
keyup	onkeyup	키를 떼었을 때	form 요소, body
change	onchange	요소의 내용이 변경되었을 때	input(text), select, textarea
reset	onreset	reset 버튼이 눌렸을 때	form
submit	onsubmit	submit 버튼이 눌렸을 때. form 내용을 전송하지 않으려면 false를 반환합니다.	form
blur	onblur	요소가 포커스를 잃었을 때	button, input, label, select, textarea, body
focus	onfocus	요소가 포커스를 얻었을 때	button, input, label, select, textarea, body
resize	onresize	요소의 사이즈가 변경되었을 때	body
scroll	onscroll	스크롤할 때	body

이벤트 핸들러 등록과 load 이벤트

HTML 문서가 파싱되고 외부 컨텐츠 로딩이 완료되면 웹 브라우저에서는 load 이벤트가 발생하는데, 이 때 load 이벤트의 이벤트 핸들러로 등록된 함수가 실행됨

load 이벤트 핸들러 함수를 자바스크립트 프로그램의 진입점으로 활용해 HTML 요소에 대한 이벤트 핸들러 등록을 하는 초기화 작업을 수행할 수 있어 HTML 코드와의 분리가 용이함

이벤트 핸들러 등록과 load 이벤트

```
function 이벤트핸들러함수() {
    이벤트 발생 시 수행할 문장들
}

window.onload = function() {
 var htmlElement = document.getElementById('id값');
 htmlElement.이벤트핸들러프로퍼티명 = function() {

    이벤트 발생 시 수행할 문장들 };
 또는
 htmlElement.이벤트핸들러프로퍼티명 = 함수명;

};
```

이벤트 핸들러 등록과 load 이벤트

```
function 이벤트핸들러함수() {
 이벤트 발생 시 수행할 문장들
}

function 초기화함수() {
 var htmlElement = document.getElementById('id값');
 htmlElement.이벤트핸들러프로퍼티명 = function() { 이벤트 발생 시 수행할 문장들 };
 또는
 htmlElement.이벤트핸들러프로퍼티명 = 이벤트핸들러함수명;

}

window.onload = 초기화함수명;
```

이벤트 핸들러 등록과 load 이벤트 - 실습 08-02

DOM 이벤트 핸들링

앞서 소개된 이벤트 핸들링 방식은 오래 전부터 클라이언트 측 자바스크립트에서 사용되어 온 방식인데, 여기에는 한 개의 HTML 요소에 둘 이상의 이벤트 처리가 안된다는 문제점이 있음

DOM 레벨 2의 이벤트 리스너(event listener) 는 한 개의 HTML 요소에서 복수의 이벤트와 연관을 맺을 수 있도록 하는 기능을 제공하며, 설정했던 이벤트 리스너를 쉽게 제거할 수도 있음

이벤트 리스너와 관련된 DOM API가 브라우저 호환성 문제가 있음

이벤트 리스너를 사용하는 경우에는 크로스 브라우징(cross browsing) 을 고려한 코드를 작성해야 함(MS의 인터넷 익스플러러도 버전 9부터는 DOM 레벨 2 이벤트 모델 표준을 따르고 있음)

DOM 이벤트 핸들링

```
var htmlElement = document.getElementById('id값');

// FireFox, Chrome과 같은 비IE 웹브라우저의 경우 htmlElement.add
EventListener(이벤트명, 이벤트핸들러함수명, false);

// IE 웹 브라우저의 경우 htmlElement.attachEvent(이벤트핸들러프로퍼
티명, 이벤트핸들러함수명);
```

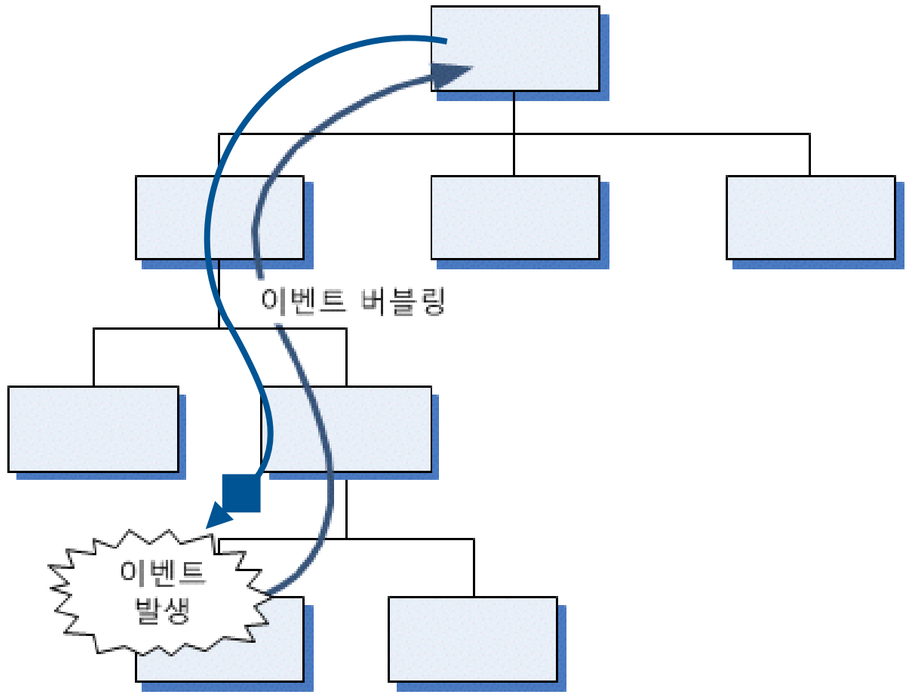
DOM 이벤트 핸들링

```
var htmlElement = document.getElementById('id값');

// FireFox, Chrome과 같은 비IE 웹브라우저의 경우 htmlElement.removeE
ventListener(이벤트명, 이벤트핸들러함수명, false);

// IE 웹 브라우저의 경우 htmlElement.detachEvent(이벤트핸들러프로퍼
티명, 이벤트핸들러함수명);
```

DOM 이벤트 핸들링 - 실습 08-03

DOM 이벤트 전파

이벤트 흐름 단계	개요
캡처 단계	웹 브라우저에서 이벤트가 발생하고, 문서 의 최상위 객체인 Document 객체에서 이 벤트가 발생한 타겟 객체의 상위 객체까지 이벤트가 전파되는 단계
타겟 단계	이벤트가 발생한 객체로 전파되는 단계
버블링 단계	타겟 객체의 상위 객체에서 다시 최상위 객 체인 Document 객체까지 전파되는 단계



DOM 이벤트 전파 - 실습 08-04

웹브라우저별 이벤트 핸들링 모델

비교항목	비 IE (FireFox, Chrome 등)	IE
이벤트리스너 등록	addEventListener()	attachEvent()
이벤트리스너 제거	removeEventListener()	detachEvent()
이벤트 발생원 참조	e.tartget	window.event.srcElement
마우스 왼쪽 버튼	e.button의 값이 0	window.event.button의 값이 1
마우스 가운데 버튼	e.button의 값이 1	window.event.button의 값이 4
마우스 오른쪽 버튼	e.button의 값이 2	window.event.button의 값이 2
이벤트버블링 억제	e.stopPropagation()	window.event.cancelBubble = true
이벤트디폴트동작 억제	e.preventDefault()	window.event.returnValue = false

웹브라우저 호환성 테스트

```
// 크로스 브라우징이 적용된 이벤트 리스너 등록 함수
function addListener(el, ev, handler) {
 // FireFox, Chrome과 같은 비 IE 웹브라우저의 경우
 if (el.addEventListener) {  el.addEventListener
 (ev, handler, false);
 } else { // IE 웹 브라우저의 경우 el.att
   achEvent('on' +ev, handler);
 }
}
```

웹브라우저 호환성 테스트

```
// 크로스 브라우징이 적용된 이벤트 리스너 제거 함수
function removeListener(el, ev, handler) {
 // FireFox, Chrome과 같은 비 IE 웹브라우저의 경우
 if (el.removeEventListener) {  el.removeEventL
 istener(ev, handler, false);
 } else { // IE 웹 브라우저의 경우 el.det
   achEvent('on' +ev, handler);
 }
}
```

웹브라우저 호환성 테스트

```
// 크로스 브라우징이 적용된 이벤트 발생원 조사 함수
function getEventSource(e) {
 // FireFox, Chrome과 같은 비 IE 웹브라우저의 경우
 if (e.target) {
   return e.target;
 else {  // IE 웹 브라우저의 경우
   return window.event.srcElement;
 }
}
```

웹브라우저 호환성 테스트

```
// 크로스 브라우징이 적용된 마우스 이벤트 버튼 확인 함수
function getButtonType(e) {
 // FireFox, Chrome과 같은 비 IE 웹브라우저의 경우
 if (e.target) {
   switch (e.button) {
     case 0:
       return '왼쪽 버튼';
     case 1:
       return '가운데 버튼';
     case 2:
       return '오른쪽 버튼';
   }
```

```
 } else {  // IE 웹 브라우저의 경우 s
  witch (window.event.button) {
                case 1:  retu
              rn '왼쪽 버튼';
                case 4:  retur
             n '가운데 버튼';
                case 2:  retur
             n '오른쪽 버튼';
          }
        }
      }
```

웹브라우저 호환성 테스트

```
// 크로스 브라우징이 적용된 이벤트 버블링 억제 함수
function cancelBubbling(e) {
 // FireFox, Chrome과 같은 비 IE 웹브라우저의 경우
 if (e.stopPropagation) {
   e.stopPropagation();
 } else {  // IE 웹 브라우저의 경우 windo
   w.event.cancelBubble = true;
 }
}
```

웹브라우저 호환성 테스트

```
// 크로스 브라우징이 적용된 이벤트 디폴트 동작 억제 함수
function cancelEvent(e) {
 // FireFox, Chrome과 같은 비 IE 웹브라우저의 경우
 if (e.preventDefault) {
   e.preventDefault();
 } else {  // IE 웹 브라우저의 경우 wind
   ow.event.returnValue = false;
 }
}
```

SK planet

동GO동락 | T아카데미

9강. jQuery와 jQuery Mobile 개요

jQuery는 John Resig에 의해 2006년 1월 BarCamp NYC에서 발표된 오픈소스 자바스크립트 프레임워크

'Write Less, Do More'

DOM의 쉬운 조작에서부터, 쉽고 일관된 이벤트 처리, 각종 효과 기능 제공을 통한 쉬운 시각화 지원, 쉬운 Ajax 지원, 쉬운 내장형 객체의 확장까지 자바스크 립트의 전반을 지원하는 프레임워크

jQuery UI라고 불리는 UI 기능을 포함해 jQuery 기반의 플러그인 라이브러리가 많음

jQuery와 jQuery Mobile

jQuery Mobile 스마트 디바이스 프레임워크는 현재 가장 많은 스마트 디바이스 지원 범위를 가지는 프레임워크

2010년 8월 프로젝트를 공식 발표함

jQuery Mobile은 iOS 휴먼 인터페이스 가이드라인의 흐름을 적용한 스마트 디바이스용 프레임워크로 폭넓은 스마트 디바이스 지원과 더불어 시멘틱 웹의 HTML을 전제로 설계되어있기 때문에 높은 접근성을 제공함

마크 업 기반의 개발 스타일을 채택하고 있어서 복잡한 자바스크립트 코드없이 HTML 태그의 속성 조작만으로도 간단한 웹 앱 제작이 가능

HTML 5와 CSS 3 기반의 사용자 인터페이스, 페이지 이동 애니메이션, Ajax 통신 지원 등 스마트 디바이스에 최적화된 기능을 제공하며, jQuery 기능을 활용하면 좀 더 고급스러운 기능을 가진 웹 앱을 구현할 수 있음

SK planet

동GO동락 | T아카데미

10강. jQuery 기본

jQuery 핵심 개념

jQuery는 자바스크립트에서 DOM 객체를 질의(query)해 조작한다는 의미를 강조하듯이, 문서 탐색 및 조작을 위한 보다 쉬운 노드 접근 기술을 제공

HTML DOM과 자바스크립트 사이의 상호작용을 간단하게 해주는 것이 jQuery의 가장 큰 특징

jQuery는 HTML DOM 객체에 접근 하기 위해 셀렉터(selector) 매커니즘을 사용

CSS 셀렉터로 문서의 특정 노드를 찾을 수 있음

jQuery()와 $()

jQuery()는 HTML DOM의 특정 노드에 접근하거나, DOM 요소를 생성하거나, DOM 구조의 로딩을 완료한 후 실행할 콜백함수를 지정하기 위해 사용하는 함수

()안에 매개변수로 셀렉터를 기술하면 셀렉터를 해석해 HTML DOM 구조 상의 노드를 찾아 jQuery 객체로 변환해 반환하고, HTML 태그 문자열이 매개변수로 전달되면 HTML DOM 요소를 생성하고, 매개변수로 함수가 전달되면 콜백함수로 등록하여 DOM 로딩 완료 시 호출함

jQuery 객체가 반환되면 해당 객체와 이벤트를 바인딩하거나, 효과를 연결하는 작업이 가능

jQuery의 별칭으로 $를 사용할 수 있어 $()를 더 많이 사용함

jQuery 객체의 메소드는 jQuery 객체 자신을 반환 값으로 전달하기 때문에 메소드 체인 구성이 가능

jQuery(document).ready(function() {...})

jQuery 이벤트인 문서 로딩 이벤트에 대한 이벤트 핸들러를 구현할 때 사용

매개변수로 전달된 document 객체를 jQuery 객체로 변환하고, jQuery 객체의 프로토타입 멤버인 ready() 메소드의 매개변수인 함수가 문서 로딩 이벤트의 이벤트 핸들러 함수로 등록됨

주로 익명함수를 매개변수로 사용 $(document).ready(function() {...}) 형태로도 사용

jQuery(document).ready(function() {...}) (계속)

HTML 문서는 가장 먼저 DOM 구조가 로딩되고, 이어서 이미지와 같은 요소 컨텐츠들이 로딩

이후 Window 객체가 로딩되고, 웹 브라우저에서 Window 객체의 load 이벤트가 발생되는데, ready() 메소드에 전달되는 이벤트 핸들러 함수의 호출 시점은 DOM 구조가 로딩이 완료되었을 때임

화면의 초기 상태를 제어하는 코드나 요소의 이벤트 핸들러를 등록하는 코드가 주로 기술됨

jQuery(document).ready(function() {...}) (계속)

jQuery(document).ready(function() {...})와 jQuery(function() {...})는 기본적으로 문서 로딩 이벤트 핸들러를 여러 개 등록할 수 있도록 지원

이 함수들은 HTML의 load 이벤트와는 다른 시점의 이벤트를 지원하기 때문에, DOM 구조 로딩 후 리소스들의 로딩이 완료되기 전과 후를 분리해 처리할 수 있음

jQuery(document).ready(function() {...})- 실습 10-01

11강. 셀렉터

셀렉터

jQuery의 셀렉터(selector)는 문서 내의 노드를 쉽게 식별하고 참조하기 위해 사용하는 기술로 CSS 셀렉터와 동일

jQuery가 자체적으로 제공하는 필터도 있음

jQuery에서는 $() 함수의 매개변수 값으로 셀렉터를 전달하면 셀렉터를 해석해 DOM에서 노드를 찾아 jQuery 객체로 변환해서 반환함

이 때 사용하는 셀렉터의 종류는 기본 셀렉터, 계층 셀렉터, 속성 셀렉터 그리고 필터 셀렉터가 있음

HTML 요소 접근과 jQuery 객체화 - 기본 셀렉터

```
$('*')
$('h3')
$('.content')
$('#message')
$('div, .header, #menu')
```

HTML 요소 접근과 jQuery 객체화 - 기본 셀렉터 (계속)

분류	셀렉터	기능
Basic	*	모든 요소를 선택
	#id	id 값을 가진 요소를 선택
	.class	class 속성 값을 가진 요소를 선택
	element	요소를 선택
	selector1, selector2, selector3	복수의 셀렉터 중 일치하는 것을 선택

HTML 요소 접근과 jQuery 객체화 - 계층 셀렉터

$('div#lectures li')
$('div > p')
$('#first +li')
$('#second ~ li')

HTML 요소 접근과 jQuery 객체화 - 계층 셀렉터 (계속)

분류	셀렉터	기능
Hierarchy	ancestor descendant	ancestor 요소의 모든 자손 요소 descendant 를 선택
	parent > child	parent 요소 자식 요소를 선택
	prev + next	prev 요소 다음의 형제 요소 next를 선택
	prev ~ siblings	prev 요소 다음의 형제 요소들 siblings를 선택

HTML 요소 접근과 jQuery 객체화 - 속성 셀렉터

```
$('div#lectures li[href='#lec03']')
$('div#reference li[href^='http://']")
$('[src][alt]')
```

HTML 요소 접근과 jQuery 객체화 - 속성 셀렉터 (계속)

분류	셀렉터	기능
Attribute	[attr]	attr 속성을 가지는 요소를 선택
	[attr=value]	attr 속성 값이 value인 요소를 선택
	[attr!=value]	attr 속성 값이 value가 아닌 요소를 선택
	[attr^=value]	attr 속성 값이 value로 시작하는 요소를 선택
	[attr$=value]	attr 속성 값이 value로 끝나는 요소를 선택
	[attr*=value]	attr 속성 값에 value를 포함하는 요소를 선택
	[selector1][selector2] [selector3]	셀렉터에 일치하는 속성을 가지고 있는 요소를 선 택

HTML 요소 접근과 jQuery 객체화 - 필터 셀렉터

분류	셀렉터	기능
Basic	:animated	show, hide, slideUp, slideDown 등의 명령으 로 현재 애니메이션 중인 요소를 선택
	:first	첫번째 요소를 선택
	:last	마지막 요소를 선택
	:not(selector)	selector에 의해 선택된 요소들 이외의 것을 선택
	:even	짝수번째의 요소를 선택
	:odd	홀수번째 요소를 선택

HTML 요소 접근과 jQuery 객체화 - 필터 셀렉터 (계속)

분류	셀렉터	기능
Basic	:eq(index)	index와 같은 index의 요소를 선택
	:gt(index)	index보다 큰 index의 요소를 선택
	:lt(index)	index보다 작은 index의 요소를 선택
	:header	h1, h2,.. 요소를 선택

HTML 요소 접근과 jQuery 객체화 - 필터 셀렉터 (계속)

분류	셀렉터	기능
Content	:contains(text)	text 문자열을 내용으로 가지고 있는 요소를 선택
	:empty	요소에 텍스트가 없을 경우 요소를 선택
	:has(selector)	selector에 일치하는 자손 요소를 가지고 있는 요 소를 선택
	:parent	:empty와 반대로 요소에 텍스트가 있을 경우 선 택
Visibility	:hidden	보이지 않는 요소를 선택
	:visible	보이는 요소를 선택

HTML 요소 접근과 jQuery 객체화 - 필터 셀렉터 (계속)

분류	셀렉터	기능
Child	:nth-child(index)	index번째 자식 요소를 선택. index는 1부터 시 작함에 유의.
	:nth-child(even | odd)	짝수/홀수번째 자식 요소를 선택
	:nth-child(Nn)	N배수번째 자식 요소를 선택
	:first-child	첫번째 자식 요소를 선택
	:last-child	마지막 자식 요소를 선택
	:only-child	자식 노드가 하나인 요소를 선택

HTML 요소 접근과 jQuery 객체화 - 필터 셀렉터 (계속)

분류	셀렉터	기능
Forms	:hidden	숨겨진 입력 요소를 선택
	:enabled	사용 가능 상태에 있는 요소를 선택
	:disabled	사용 불가능 상태에 있는 요소를 선택
	:checked	체크 상태에 있는 요소를 선택
	:selected	선택 상태에 있는 요소를 선택

HTML 요소 접근과 jQuery 객체화 - 실습 11-01

SK planet

동GO동락 | T아카데미

12강. jQuery 이벤트

이벤트 종류

구분	이벤트	설명
문서 로딩	ready	해당 DOM 로딩이 완료되었을 때
마우스	click	마우스로 클릭할 때 (자바스크립트 이벤트)
	dbclick	마우스로 더블클릭할 때 (자바스크립트 이벤트)
	focusin	요소가 포커스를 얻을 때
	focusout	요소가 포커스를 잃을 때 (자바스크립트 이벤트)
	hover	mouseenter와 mouseleave를 하나로 묶음
	mouseenter	요소 안으로 마우스가 들어왔을 때 (자식 노드에서는 이벤트 감지 안함)
	mouseleave	요소로부터 마우스가 벗어날 때 (자식 노드에서는 이벤트 감지 안함)
	mousedown	마우스 버튼을 눌렀을 때 (자바스크립트 이벤트)
	mouseup	마우스 버튼을 놓았을 때 (자바스크립트 이벤트)
	mousemove	마우스의 포인터를 이동하였을 때 (자바스크립트 이벤트)
	mouseout	요소로부터 마우스가 벗어날 때 (자바스크립트 이벤트)
	mouseover	요소 안으로 마우스 포인터가 들어왔을 때 (자바스크립트 이벤트)

이벤트 종류

구분	이벤트	설명
키보드	keypress	키를 눌렀을 때 (자바스크립트 이벤트)
	keydown	키를 누르고 있을 때 (자바스크립트 이벤트)
	keyup	키를 떼었을 때 (자바스크립트 이벤트)
	focusin	요소가 포커스를 얻을 때
	focusout	요소가 포커스를 잃을 때 (자바스크립트 이벤트)
폼	focus	요소가 포커스를 얻을 때 (자바스크립트 이벤트)
	blur	요소가 포커스를 잃을 때 (자바스크립트 이벤트)
	change	요소의 값이 변경되었을 때 (자바스크립트 이벤트)
	select	사용자가 텍스트를 선택했을 때 (자바스크립트 이벤트)
	submit	폼의 내용을 전송할 때 (자바스크립트 이벤트)
웹브라우저	resize	요소의 사이즈가 변경되었을 때 (자바스크립트 이벤트)
	scroll	스크롤할 때 (자바스크립트 이벤트)

이벤트 등록 및 제거

jQuery는 이벤트 처리에 있어 크로스 브라우징 제공

이벤트를 요소에 등록하기 위해선 먼저 셀렉터를 이용해 이벤트를 처리하고자 하는 요소를 찾아 jQuery 객체를 생성한 후, on(), one(), off() 메소드 등을 이용하여 특정 이벤트에 대한 이벤트 핸들러를 등록 하거나 제거함

이벤트 등록 - on() 메소드

on() 메소드는 현재 선택된 요소에 대해 하나 이상의 이벤트와 이벤트 핸들러 함수를 연결하는 메소드

.on(events [, selector] [, data], handler(event))
.on(events-map [, selector] [, data])
events: 스페이스로 구분된 하나 이상의 이벤트 문자열입니다.
selector: 셀렉터 문자열은 이벤트가 트리거 되는 현재 선택된 요소의 자손 요소를 필터링 합니다. 만일 null 값을 갖거나 생략되면 현재 선택된 요소에 이벤트가 도달했을 때 트리거 됩니다.
data: 이벤트가 트리거 되면 handler 함수에서 event.data 프로퍼티로 참조할 수 있습니다.
handler(event): 이벤트가 트리거 되면 실행되는 이벤트 핸들러 함수입니다.
events-map: 스페이스로 구분된 하나 이상의 이벤트 문자열과 이벤트 핸들러 함수를 키:값으로 구성한 맵입니다.

이벤트 등록 - on() 메소드 (계속)

```
$(‘p’).on(‘click’, function() {
 alert( $(this).text() );
});
$("body").on("click", "p", function(){
 alert( $(this).text() );
});
function myHandler(event) {
 alert(event.data.foo);
};
$(‘p’).on(‘click’, {foo: ‘bar’}, myHandler);
$(‘form’).on(‘submit’, false);
$(‘form’).on(‘submit’, function(event) {
 event.preventDefault();
});
$(‘form’).on(‘submit’, function(event) {
 event.stopPropagation();
});
```

이벤트 등록 - on() 메소드 - 실습 12-01

이벤트 등록 - one() 메소드

one() 메소드 역시 현재 선택된 요소에 대해 하나 이상의 이벤트와 이벤트 핸들러 함수를 연결하는 메소드이지만, on() 메소드와 달리 한 번만 실행

.one(events [, selector] [, data], handler(event))
.one(events-map [, selector] [, data])
events: 스페이스로 구분된 하나 이상의 이벤트 문자열입니다.
selector: 셀렉터 문자열은 이벤트가 트리거 되는 현재 선택된 요소의 자손 요소를 필터링 합니다. 만일 null 값을 갖거나 생략되면 현재 선택된 요소에 이벤트가 도달했을 때 트리거 됩니다.
data: 이벤트가 트리거 되면 handler 함수에서 event.data 프로퍼티로 참조할 수 있습니다.
handler(event): 이벤트가 트리거 되면 실행되는 이벤트 핸들러 함수입니다.
events-map: 스페이스로 구분된 하나 이상의 이벤트 문자열과 이벤트 핸들러 함수를 키:값으로 구성한 맵입니다.

이벤트 등록 - one() 메소드 - 실습 12-02

이벤트 제거 - off() 메소드

off() 메소드는 현재 선택된 요소에 대해 하나 이상의 이벤트와 이벤트 핸들러 함수의 연결을 제거하는 메소드

.off(events [, selector] [, handler(event)])
.off(events-map [, selector])

events: 스페이스로 구분된 하나 이상의 이벤트 문자열입니다.
selector: 셀렉터 문자열은 현재 선택된 요소의 자손 요소를 필터링합니다. handler(event): 이벤트와 연결을 제거하고자 하는 하나 이상의 이벤트에 연결되어 있는 이벤트 핸들러 함수입니다.
events-map: 이벤트와 연결을 제거하고자 하는 스페이스로 구분된 하나 이상의 이벤트 문자열과 이벤트 핸들러 함수를 키:값으로 구성한 맵입니다.

이벤트 제거 - off() 메소드 - 실습 12-03

이벤트 취소와 이벤트 전파 차단

jQuery에서 이벤트 처리를 위해 이벤트 핸들러로 등록된 콜백함수가 호출될 때 인자로 jQuery 이벤트 객체가 전달되는데 이 이벤트 객체를 이용하면 쉽게 이벤트의 흐름을 제어할 수 있음

요소의 기본 액션을 취소하기 위해선 이벤트 등록 함수(on(), one())의 콜백함수에 전달되는 jQuery 이벤트 객체의 preventDefault() 메소드를 호출함

특정 요소에서 발생한 이벤트의 버블링을 차단하기 위해선 이벤트 등록 함수(on(), one())의 콜백함수에 전달되는 jQuery 이벤트 객체의 stopPropagation() 메소드를 호출함

이벤트 취소와 이벤트 전파 차단 (계속)

```
$('form').on('submit', function(event) {
 ...
 event.preventDefault();
});

$('body').on('click', 'a', function(event){
 ...
 event.preventDefault();
});

$('p').click(function(event){
 event.stopPropagation();
 ...
});
```

이벤트 취소와 이벤트 전파 차단 - 실습 12-04, 12-05

13강. jQuery 효과

기본효과는 HTML 요소를 보이거나 감추는 기능

hide() 메소드는 요소에 대해 dispay:none을 적용하고, show()는 dispaly:block 또는 display:inline을 적용함

hide(), show(), toggle() 메소드를 사용하면 요소의 display 속성 값을 jQuery 데이터 캐시에 저장해 두기 때문에 복원 시 display 설정 을 이전 값으로 돌려줌

기본효과

hide() 메소드는 요소가 사라지는 애니메이션 효과를 주기 위해 사용

.hide()

.hide(duration [, callback])

.hide([duration] [, easing] [, callback])

duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문자열 또는 숫자 값입니다. ‘slow’, normal’, ‘fast’ 문자열을 사용하 수 있습니다. (각각 200, 400, 600) callback: 애니메이션이 끝났을 때 호출되는 함수입니다.

easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

기본효과

show() 메소드는 요소가 나타나는 애니메이션 효과를 주기 위해 사용

.show()

.show(duration [, callback])

.show([duration] [, easing] [, callback])

duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문자열 또는 숫자 값입니다. ‘slow’, normal’, ‘fast’ 문자열을 사용하 수 있습니다. (각각 200, 400, 600)

callback: 애니메이션이 끝났을 때 호출되는 함수입니다.

easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

기본효과

toggle() 메소드는 show()와 hide()를 번갈아 실행시켜 애니메이션 효과를 주기 위해 사용

.toggle([duration] [, callback])
.toggle([duration] [, easing] [, callback])
.toggle(showOrHide)

duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문자열 또는 숫자 값입니다. ‘slow’, normal’, ‘fast’ 문자열을 사용하 수 있습니다. (각각 200, 400, 600)

callback: 애니메이션이 끝났을 때 호출되는 함수입니다.
easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

showOrHide: 요소를 나타낼 것인지(true), 숨길 것인지(false) 알려주는 불리언 값입니다.

기본효과 - 실습 13-01

페이딩

페이딩은 HTML 요소의 불투명도(opacity)를 조절하는 기능으로
fadeIn(), fadeOut(), fadeToggle(), fadeTo() 메소드를 사용해
요소를 점점 밝게 또는 점점 흐리게 보이도록 조절함

페이딩

fadeIn() 메소드는 요소가 dispay:none일 경우 display:block 또는 display:inline으로 복원하고, opacity를 0에서 1로 변경하여 서서히 나타나는 애니메이션 효과를 주기 위해 사용

.fadeIn(duration [, callback])

.fadeIn([duration] [, easing] [, callback])

duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문자열 또는 숫자 값입니다. 'slow', normal', 'fast' 문자열을 사용하 수 있습니다. (각각 200, 400, 600)

callback: 애니메이션이 끝났을 때 호출되는 함수입니다.

easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

페이딩

fadeOut() 메소드는 opacity를 1에서 0으로 변경하고, 모두 사라지면 dispay:none으로 변경해 서서히 사라지는 애니메이션 효과를 주기 위해 사용

.fadeOut(duration [, callback])

.fadeOut([duration] [, easing] [, callback])

duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문자열 또는 숫자 값입니다. ‘slow’, normal’, ‘fast’ 문자열을 사용하 수 있습니다. (각각 200, 400, 600) callback: 애니메이션이 끝났을 때 호출되는 함수입니다.

easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

페이딩

fadeToggle() 메소드는 fadeIn()과 fadeOut()를 번갈아 실행시켜 애니메이션 효과를 주기 위해 사용

.fadeToggle([duration] [, easing] [, callback]) duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문 자열 또는 숫자 값입니다. ‘slow’, normal’, ‘fast’ 문자열을 사용하 수 있 습니다. (각각 200, 400, 600)
callback: 애니메이션이 끝났을 때 호출되는 함수입니다.
easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

페이딩

fadeTo() 메소드는 0 ~ 1 사이의 값을 가진 불투명도(opacity)를 지정하여 서서히 사라지거나 나타나는 애니메이션 효과를 주기 위해 사용

.fadeTo(duration, opacity [, callback])

.fadeTo(duration, opacity [, easing] [, callback])

duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문자열 또는 숫자 값입니다. ‘slow’, normal’, ‘fast’ 문자열을 사용하 수 있습니다. (각각 200, 400, 600)

opacity: 0 ~ 1 사이의 불투명도를 나타내는 숫자 값입니다.

callback: 애니메이션이 끝났을 때 호출되는 함수입니다.

easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

페이딩 - 실습 13-02

슬라이딩

슬라이딩은 HTML 요소의 높이를 조절하여 접었다 폈다 (collapsible) 할 수 있는 기능

slideDown(), slideUp(), slideToggle() 메소드를 이용해 아코디언(accordian) 혹은 네비게이션 등에 많이 사용

이미지를 사용할 경우엔 반드시 블록 요소에 감싸서 블록 요소에 슬라이딩 효과를 주어야 애니메이션 진행 과정에서 픽셀 깨짐 현상들이 발생하지 않음

슬라이딩

slideDown() 메소드는 요소의 높이를 0에서 원래 값으로 변경하면서 펼치는 애니메이션 효과를 주기 위해 사용

.slideDown([duration] [, callback])

.slideDown([duration] [, easing] [, callback])

duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문자열 또는 숫자 값입니다. ‘slow’, normal’, ‘fast’ 문자열을 사용하 수 있습니다. (각각 200, 400, 600)

callback: 애니메이션이 끝났을 때 호출되는 함수입니다.

easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

슬라이딩

slideUp() 메소드는 요소의 높이를 현재 높이에서 0으로 변경하면서 접히는 애니메이션 효과를 주기 위해 사용

slideUp() 메소드가 수행을 완료하면 요소는 dispaly:none 상태가 됨

.slideUp([duration] [, callback])

.slideUp([duration] [, easing] [, callback])

duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문자열 또는 숫자 값입니다.

'slow', normal', 'fast' 문자열을 사용하 수 있습니다. (각각 200, 400, 600)

callback: 애니메이션이 끝났을 때 호출되는 함수입니다.

easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

슬라이딩

slideToggle() 메소드는 slideUp()과 slideDown()을 번갈아 실행시켜 애니메이션 효과를 주기 위해 사용

.slideToggle([duration] [, callback])
.slideToggle([duration] [, easing] [, callback])
duration:애니메이션이 실행되는 시간(1/1000초 단위)을 설정하는 문자열 또는 숫자 값입니다. ‘slow’, normal’, ‘fast’ 문자열을 사용하 수 있습니다. (각각 200, 400, 600)
callback: 애니메이션이 끝났을 때 호출되는 함수입니다.
easing: 트랜지션에 사용하는 easing 함수 이름의 문자열입니다. easing 함수는 플러그인이기 때문에 관련 라이브러리 파일이 필요합니다.

슬라이딩 - 실습 13-03

14강. jQuery Mobile 컴포넌트 - 페이지, 버튼, 툴바

단일 페이지와 다중 페이지에서 페이지 전환

jQuery Mobile은 웹을 근간으로 하는 기술이기 때문에 화면 간의 전환에 HTML의 링크 <a>태그를 이용함

jQuery Mobile은 단일 HTML 페이지 내에서 data- role=”page” 속성을 가진 여러 태그들을 페이지 컨테이너로 사용하 며, 가장 먼저 기술한 페이지 컨테이너를 보여줌

나머지 페이지 컨테이너는 표시되지 않으며, 특정 페이지 컨테이너로의 전환은 <a>태그의 href 속성의 id 셀렉터를 통해 이루어짐

단일 페이지와 다중 페이지에서 페이지 전환 - 실습 14-01, 14-02

페이지 전환 애니메이션

효과	특징
fade	페이드인/페이드아웃 효과를 적용합니다.
pop	화면 가운데서 페이지가 확대되며 나타납니다.
flip	화면 가운데 Y축을 기준으로 360도 회전하면서 나타납니다.
turn	화면 좌측의 Y축을 기준으로 360도 회전하면서 나타납니다.
flow	이전 화면은 밑으로 빠져 흘러가 사릿고, 새로운 화면이 흘러가다 밑에서 올라옵니다
slidefade	좌우 슬라이딩과 페이드 효과가 함께 적용됩니다.
slide	좌우 슬라이딩 효과가 적용됩니다.
slideup	페이지가 위로 슬라이딩됩니다.
slidedown	페이지가 아래로 슬라이딩됩니다.

페이지 전환 애니메이션

페이지 전환 애니메이션을 설정하려면 <a> 태그에 data-transition 속성 값으로 효과 이름을 기술

아무런 설정을 하지 않았다면 fade가 적용

<a> 요소에 대해 data-direction 속성 값을 “reverse”로 설정하면 적용된 애니메이션 방향과 반대 방향의 효과를 줌

왼쪽으로 슬라이딩하는 slide의 경우 data-direction 속성에 “reverse”를 설정하면 페이지가 오른쪽으로 슬라이딩함

페이지 전환 애니메이션 - 실습 14-03

페이지 돌아가기

첫 번째로 페이지 컨테이너에 data-add-back-btn="true" 속성을 기술하면 자동으로 헤더 좌측에 Back 버튼이 생성

data-back-btn-text와 data-back-btn-theme 속성을 설정해 텍스트와 적용 테마를 변경할 수 있음

두 번째로 <a>태그의 data-rel 속성에 "back" 값을 지정

href 속성을 직접 지정하는 하는 것보다 data-rel="back" 속성을 기술해 사용할 것을 권장함

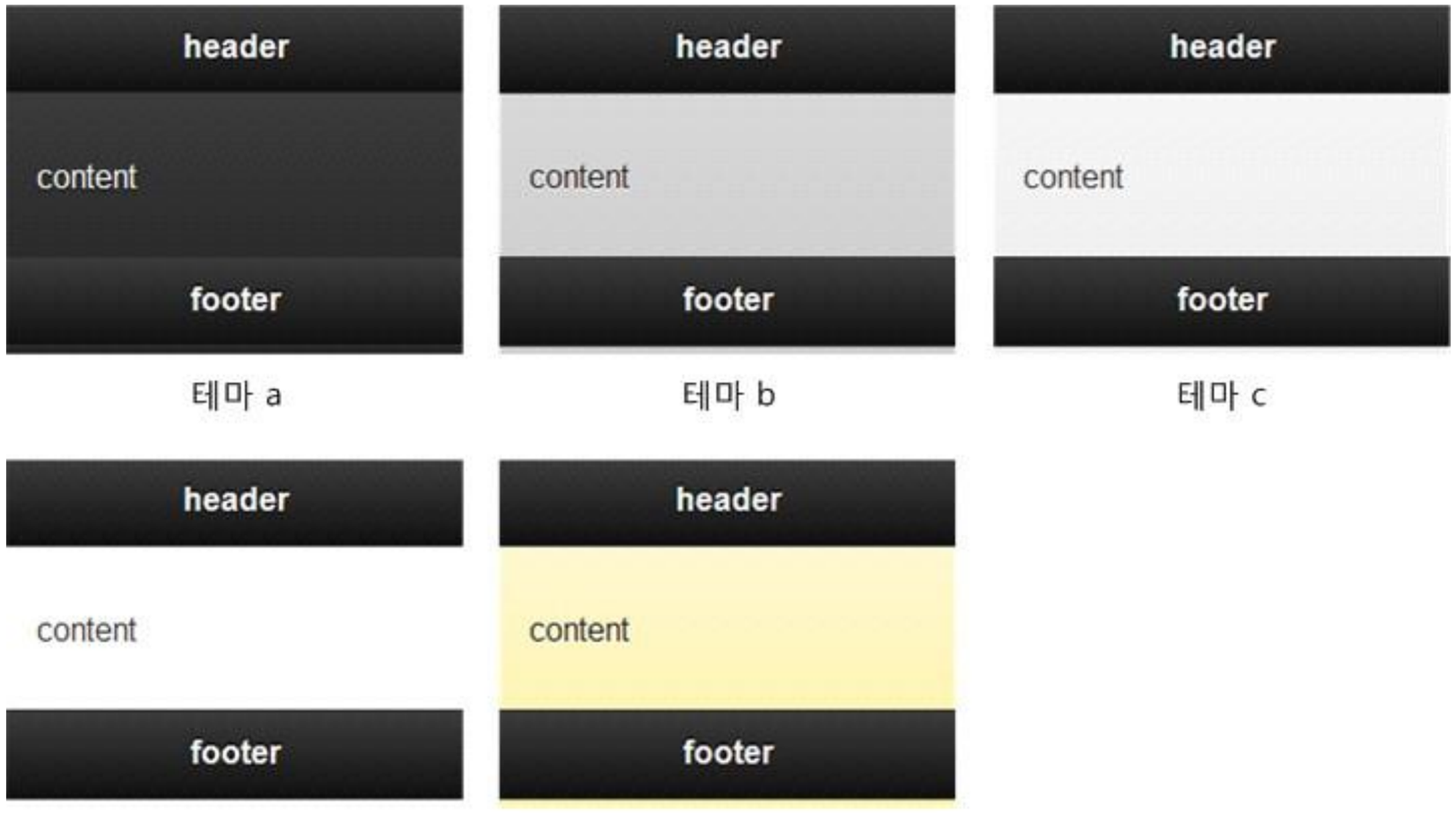
페이지 돌아가기 - 실습 14-04

페이지에 테마 적용하기

jQuery Mobile은 기본 테마에 a(black), b(blue), c(gray), d(dark gray), e(yellow)까지 5개의 스왓치를

data-role=”page” 속성을 가지고 있는 페이지 컨테이너의 data-theme 속성에 5개의 값 중 하나를 정하면 기본 테마 색상이 변경됨

페이지에 테마 적용하기 - 페이지 컨테이너에 테마 적용



jQuery Mobile에서 버튼은 <a>태그나 <button>, <input> 태그를 사용해 만듭니다.
<a>태그를 버튼으로 만들기 위해서는 data-role=”button” 속성을 추가하면 됩니다.

기본 버튼과 옵션

옵션	특징
data-mini	기본값 false. true 일 경우 버튼의 세로 높이와 글자의 크기를 작게 합 니다.
data-inline	기본값 false. true일 경우 버튼의 폭을 텍스트에 맞춰 줄입니다.
data-corner	기본값 true. false일 경우 버튼의 모서리를 직각으로 만듭니다.
data-shadow	기본값 true. false일 경우 버튼의 그림자 스타일을 적용하지 않습니다.

기본 버튼과 옵션 - 실습 14-05

아이콘 적용하기

jQuery Mobile에서는 버튼에 아이콘을 추가하기 위해서 data-icon 속성에 값을 추가하면 됨

아이콘은 기본적으로 왼쪽에 위치하며, 위치를 변경하기 위해 data-iconpos 속성 값으로 left, right, top, buttom 값을 사용하여 변경함

data-iconpos 속성에 “notext” 값을 설정하면 아이콘만 있고 텍스트는 없는 버튼이 표시됨

아이콘 적용하기 - jQuery Mobile이 제공하는 표준 아이콘들

아이콘 적용하기 - 실습 14-06

그룹버튼 만들기

여러 버튼을 태그로 묶고 data-role=”controlgroup” 속성을 기술하면 세로로 그룹화된 버튼이 생성됨

 태그에 data-type=”horizontal” 속성을 추가하고 각 버튼에 data-line=”true”를 지정하면 가로 방향의 그룹버튼을 만들 수 있음

그룹버튼 만들기 - 실습 14-07

버튼에 테마 적용하기

jQuery Mobile은 기본 테마에 a(black), b(blue), c(gray), d(dark gray), e(yellow)까지 5개의 스왓치를 제공

버튼에 data-theme 속성에 5개의 값 중 하나를 정하면 기본 테마 색상이 변경됨

버튼에 테마 적용하기

jQuery Mobile은 기본 테마에 a(black), b(blue), c(gray), d(dark gray), e(yellow)까지 5개의 스왓치를 제공

버튼에 data-theme 속성에 5개의 값 중 하나를 정하면 기본 테마 색상이 변경됨

버튼에 테마 적용하기 - 버튼에 테마 적용

헤더 바와 풋터 바

data-role="header" 속성을 가진 태그를 헤더 바(header bar)

data-role="footer" 속성을 가진 태그를 풋터 바(footer bar)

헤더 바 안에는 <a>태그를 최대 두 개까지 가질 수 있는데 먼저 기술된 <a> 태그는 좌측 버튼으로, 나중에 기술된 <a> 태그는 우측 버튼으로 생성됨

헤더 바 안에 <a> 태그 한 개가 있을 경우 <a> 태그에 class="ui-btn-right" 속성을 적용해 우측 버튼으로 만들 수도 있음

풋터 바는 풋터 바 내에 <a> 태그를 여럿 추가할 수 있지만 폭을 벗어나면 행을 바꿔 추가하게 됨

풋터 바는 추가된 내용들을 왼쪽 정렬하기 때문에 가운데 정렬을 고려할 필요가 있음

헤더 바와 풋터 바는 컨텐트 영역의 내용이 많아 스크롤이 되면 위 아래로 사라지는데 이를 방지하기 위해선 data-position="fixed" 속성을 기술하면 해결됨

컨텐트 영역을 탭 하는 순간 헤더 바와 풋터 바를 슬라이딩 애니메이션을 통해 위 아래로 사라지게 하려면 data-fullscreen="true" 속성을 주면 됨

헤더 바와 풋터 바 - 실습 14-08

네비게이션 바

네비게이션 바는 data-role=”navbar” 속성을 태그에 추가해 사용

네비게이션 바는 헤더 바나 풋터 바에 포함될 수 있는 탭 형태의 버튼 그룹

초기 활성화 버튼을 표시하고자 하면 <a> 태그에 class=”ui-btn-active” 속성을 지정하면 됨

기본적으로 아이콘의 위치는 top이고, 일괄적으로 아이콘의 위치를 변경하려면 data-role=”navbar” 속성을 가진 태그에 data-iconpos 속성을 추가하면 됨

한 행에 최대 5개까지 탭 버튼을 추가할 수 있으며 5개를 넘어가면 순서대로 한 행에 두 개씩 여러 행으로 탭 버튼을 배치함

네비게이션 바 - 실습 14-09

툴바에 테마 적용하기

jQuery Mobile은 기본 테마에 a(black), b(blue), c(gray), d(dark gray), e(yellow)까지 5개의 스왓치를 제공

툴바에 data-theme 속성에 5개의 값 중 하나를 정하면 기본 테마 색상이 변경됨

툴바에 테마 적용하기 - 툴바에 테마 적용

테마 a

테마 b

테마 c

테마 d

테마 e

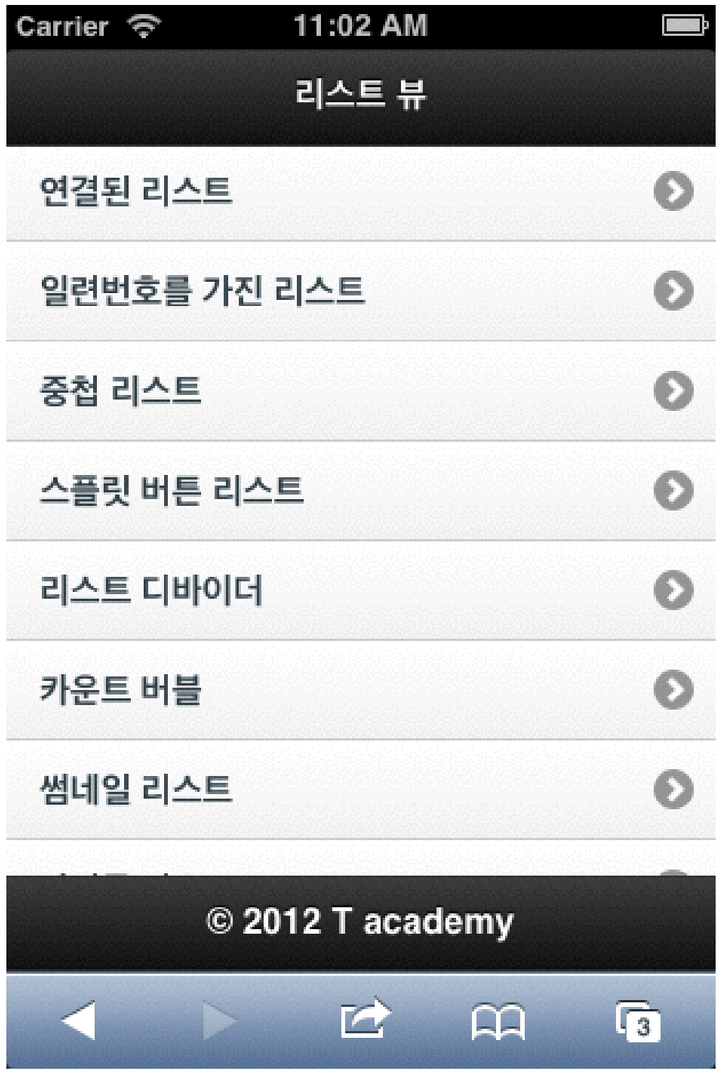
15강. jQuery Mobile 컴포넌트 - 리스트 뷰

리스트 뷰

jQuery Mobile에서는 리스트뷰(Listview)라는 리스트를 표시하는 UI를 제공

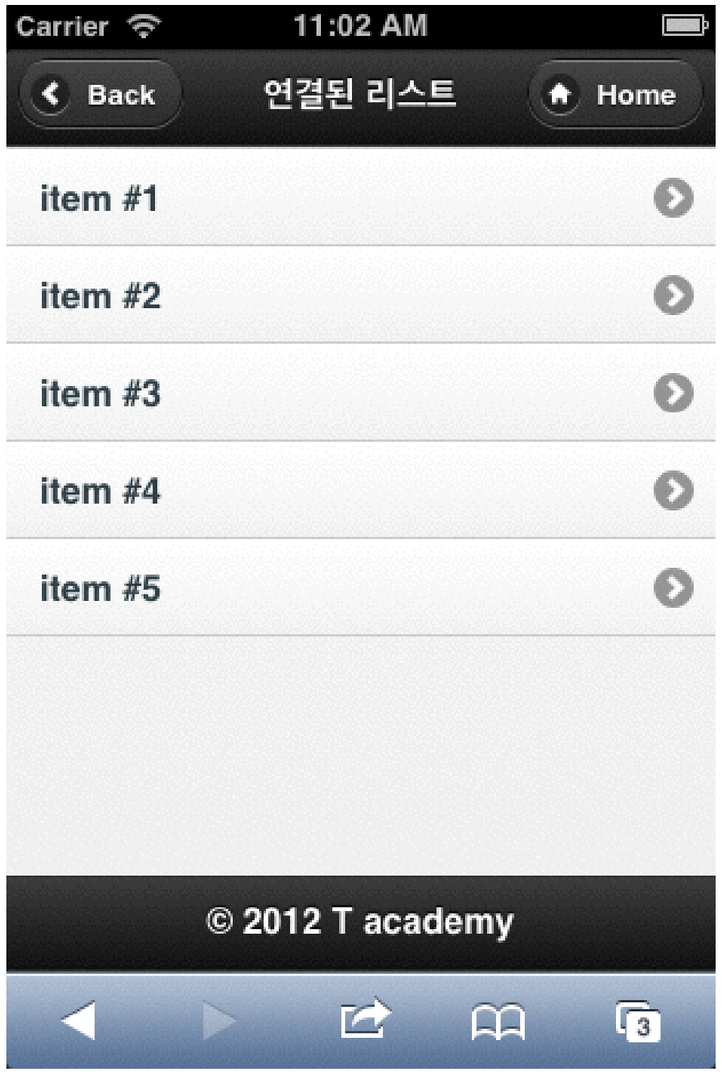
리스트는 <ul>, <ol> 태그를 기반으로 만들며, data- role=”list view” 속성을 추가하면 스마트 디바이스에 최적화된 리스 트를 제공함

리스트 뷰


Carrier
11:02 AM
리스트 뷰
연결된 리스트
일련번호를 가진 리스트
중첩 리스트
스플릿 버튼 리스트
리스트 디바이더
카운트 버블
썸네일 리스트
© 2012 T academy

리스트의 항목 <li> 태그 안에 <a> 태그를 입력하면 오른쪽 끝에 화살표가 표시가 되는 연결된 리스트(linked list)가 생성됨

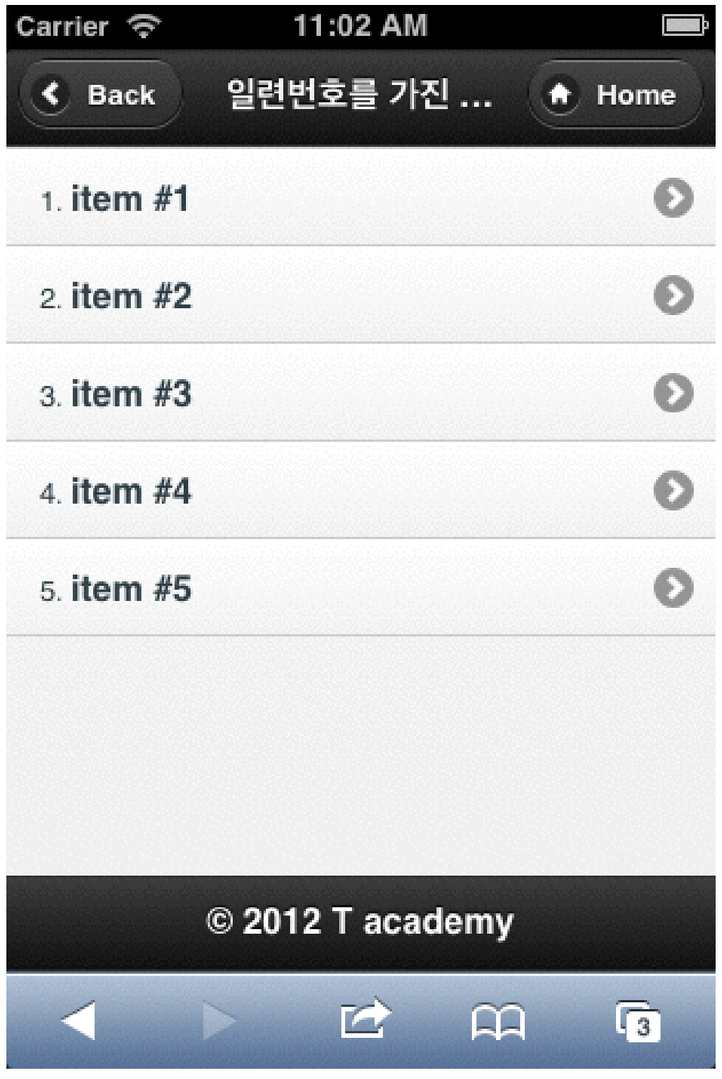
연결된 리스트


Carrier
11:02 AM
Back
연결된 리스트
Home
item #1
item #2
item #3
item #4
item #5
© 2012 T academy

일련번호를 가진 리스트

 기반의 리스트를 구성하면 일련번호가 표시되는 리스트 생성

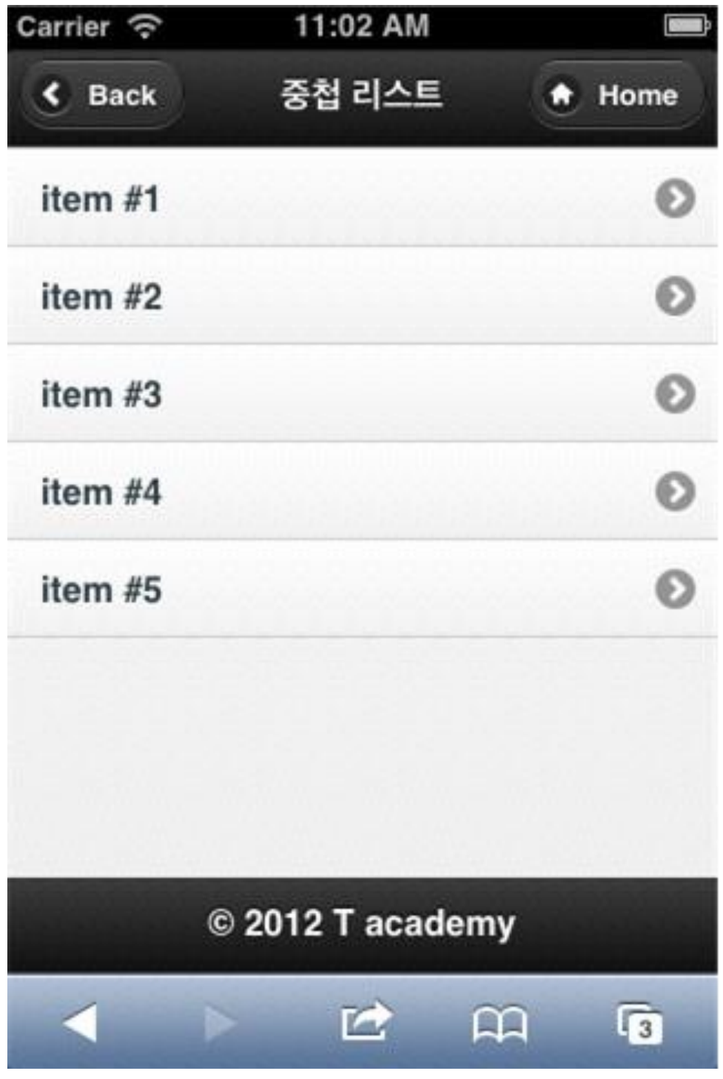
일련번호를 가진 리스트

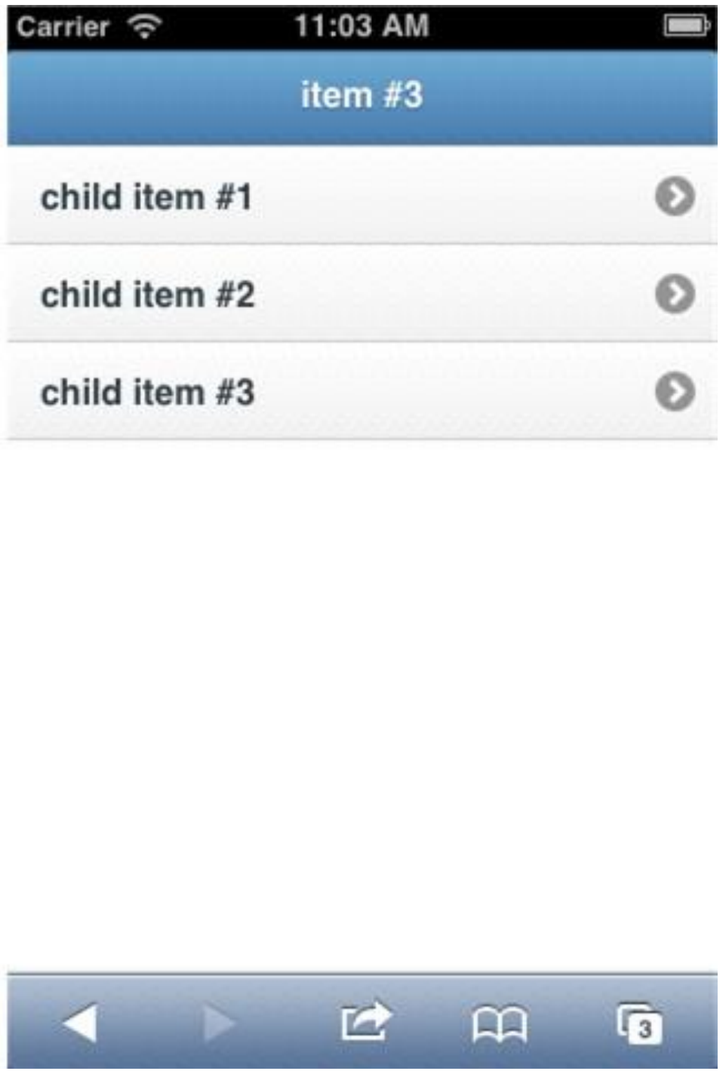

Carrier
11:02 AM
Back
일련번호를 가진 ...
Home
1. item #1
2. item #2
3. item #3
4. item #4
5. item #5
© 2012 T academy

중첩 리스트

리스트의 항목 <li> 안에 중첩된 <ul> 또는 <ol> 기반의 연결 리스트를 구성할 수 있는데, 하위 리스트에는 상위 리스트에서 선택한 항목이 헤더에 표시되고, 기본적으로 테마 b가 적용됨

중첩 리스트


Carrier
11:02 AM
Back
중첩 리스트
Home
item #1
item #2
item #3
item #4
item #5
© 2012 T academy


Carrier
11:03 AM
item #3
child item #1
child item #2
child item #3

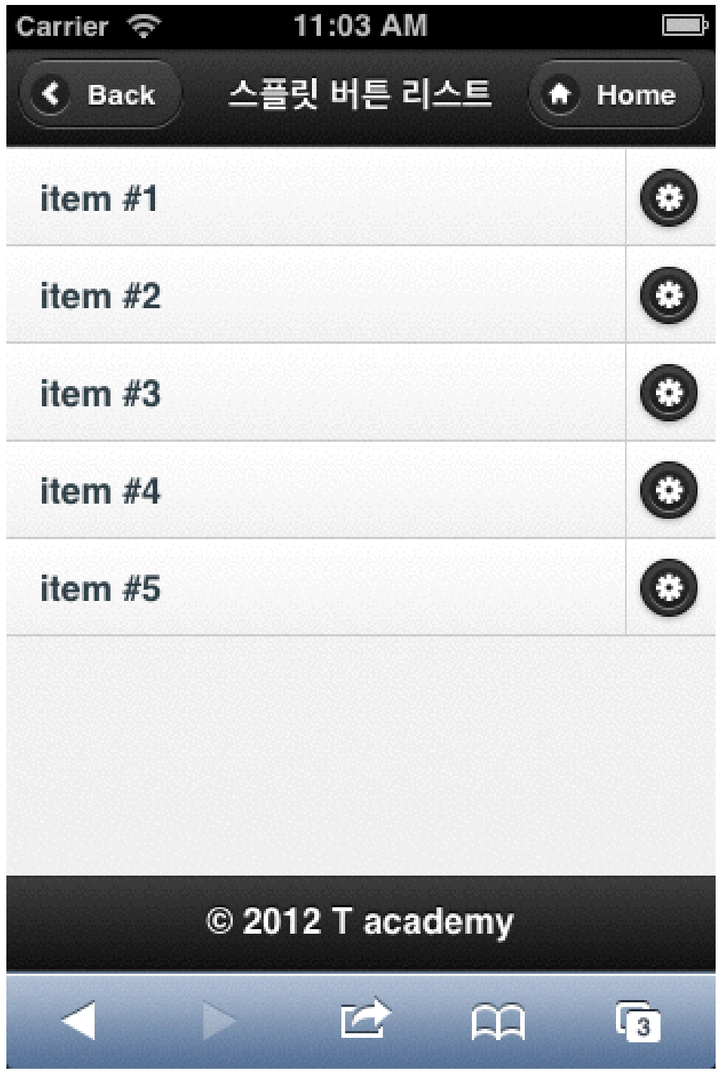
스플릿 버튼 리스트

리스트의 항목 <li> 태그에 <a> 태그를 2개 작성하면 스플릿 버튼 리스트(split button list)가 생성됨

스플릿 버튼 리스트는 첫번째 <a> 태그의 텍스트는 출력하고, 두번째 <a>태그는 연결 아이콘만 표시됨

리스트에 data-split-icon 속성을 추가해 아이콘을 변경할 수 있고, data-split-theme 속성으로 아이콘의 테마(기본은 테마 b)를 변경할 수 있음

스플릿 버튼 리스트


Carrier
11:03 AM
Back
스플릿 버튼 리스트
Home
item #1
item #2
item #3
item #4
item #5
© 2012 T academy
3

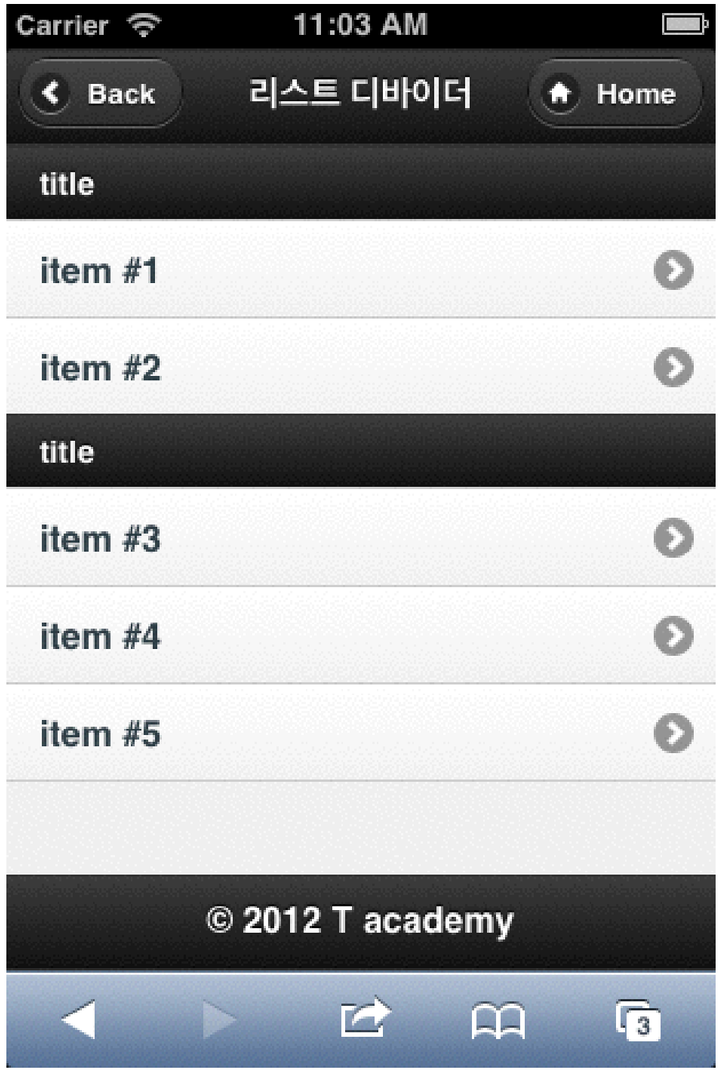
리스트 디바이더

리스트의 <li> 태그에 data-role="list-divider" 속성을 추가하면 리스트 디바이더(list divider)를 생성

리스트 디바이더는 주로 카테고리나 제목을 표시하는 경우 많이 사용함

리스트에 data-dividertheme 속성을 추가해 리스트 디바이더의 테마(기본 테마 b)를 변경할 수 있음

리스트 디바이더


Carrier
11:03 AM
Back
리스트 디바이더
Home
title
item #1
item #2
title
item #3
item #4
item #5
© 2012 T academy
3

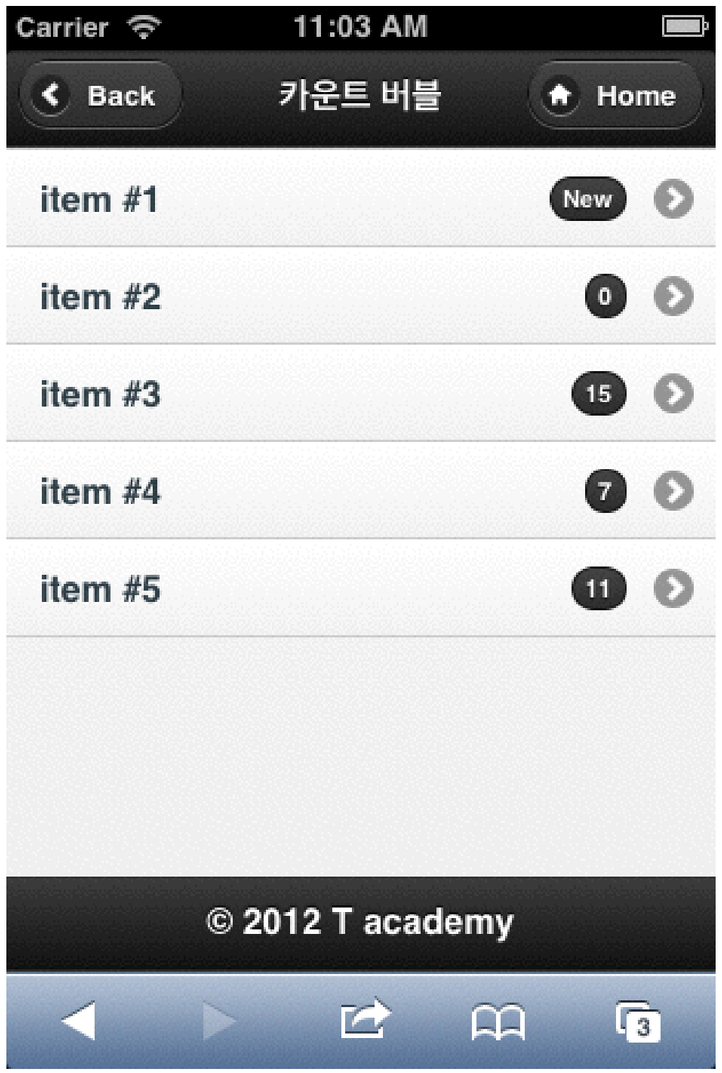
카운트 버블

리스트의 <li> 태그 내의 영역에 class="ui-li-count" 속성을 추가하면 항목의 오른쪽에 카운트 버블(count bubble)을 표시함

주로 새로운 소식의 건수 혹은 신규 등을 표시하는 데 사용

리스트에 data-count-theme 속성을 추가해 카운트 버블의 테마를 변경할 수 있음

카운트 버블

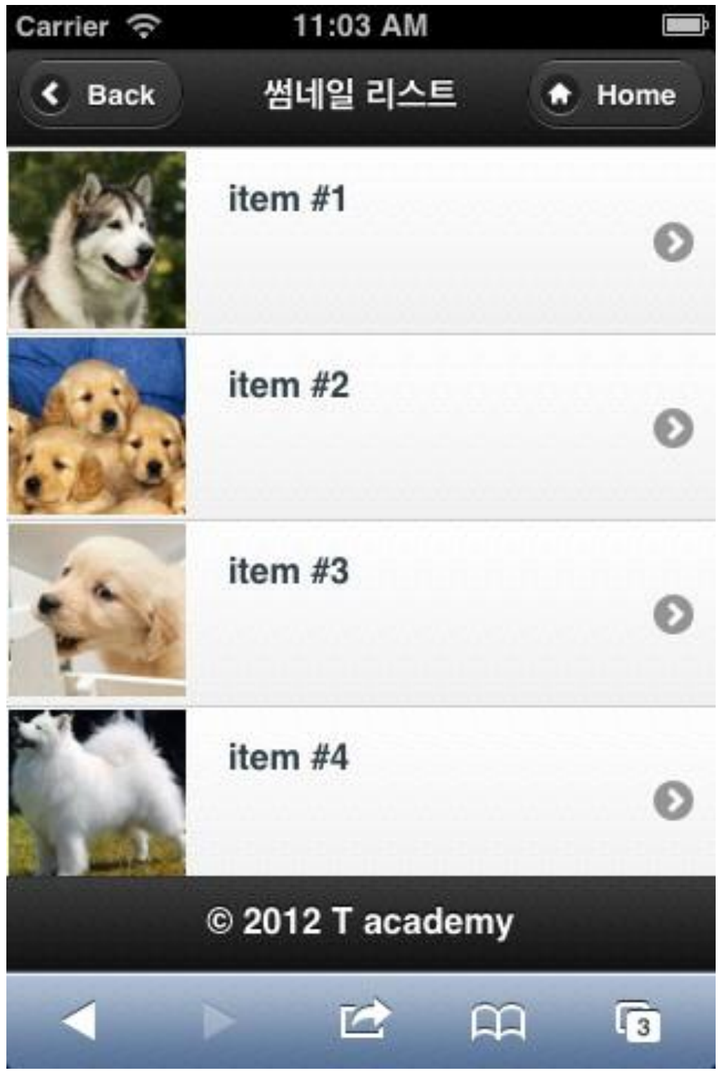

Carrier
11:03 AM
Back
카운트 버블
Home
item #1
New
item #2
0
item #3
15
item #4
7
item #5
11
© 2012 T academy
3

썸네일 리스트

리스트의 <li> 태그 안에 <img> 태그를 사용하면 왼쪽에 이미지가 표시되는 썸네일 리스트(thumbnail list)를 만들 수 있음

썸네일 리스트는 왼쪽 영역에 가로 80px, 세로 80px로 이미지를 표시

썸네일 리스트

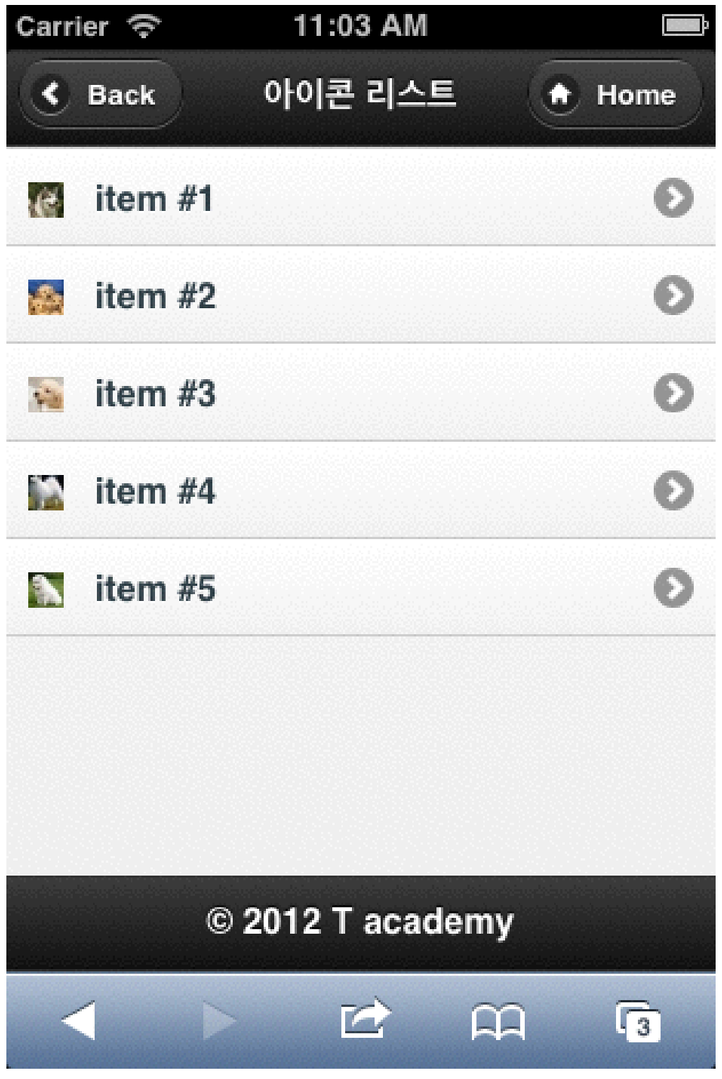

Carrier
11:03 AM
Back
썸네일 리스트
Home
item #1
item #2
item #3
item #4
© 2012 T academy

아이콘 리스트

리스트의 <li> 태그 안에 <img> 태그에 class="ui-li-icon" 속성을 추가하면 아이콘처럼 더 작은 이미지로 표시되는 아이콘 리스트를 만들 수 있음

아이콘 리스트는 왼쪽 영역에 가로 16px, 세로 16px로 이미지를 표시

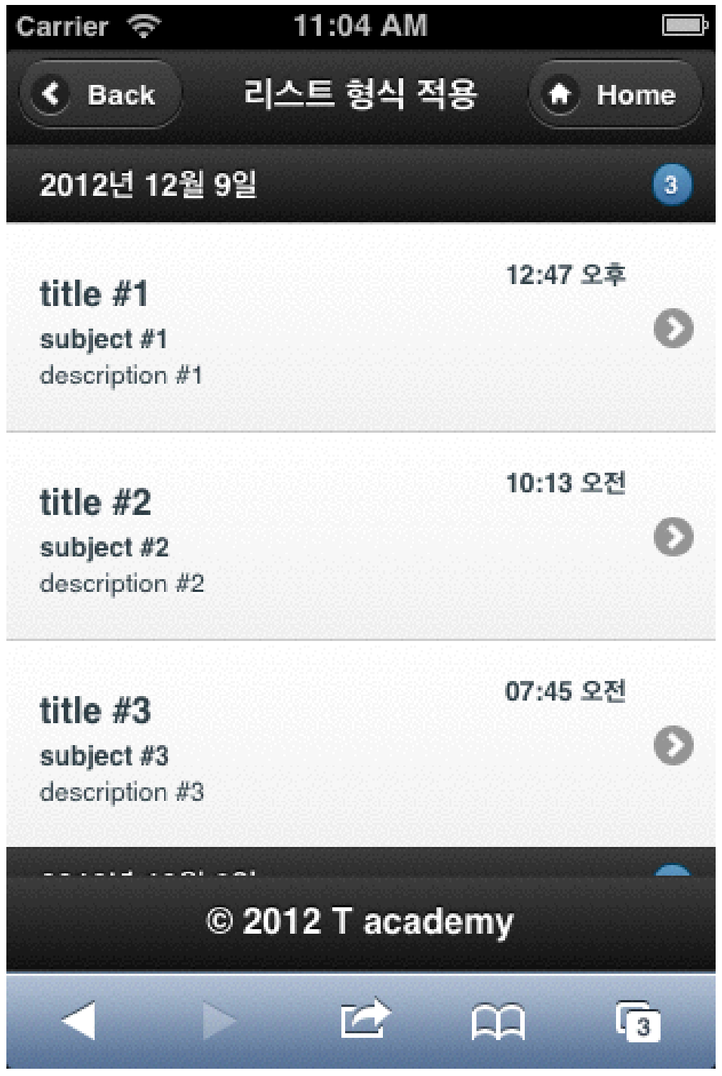
아이콘 리스트


Carrier
11:03 AM
Back
아이콘 리스트
Home
item #1
item #2
item #3
item #4
item #5
© 2012 T academy

컨텐트 형식 적용하기

리스트 내의 <li> 태그에 <h#>,<p>,과 같은 태그들을 이용해서 형식을 적용할 수 있으며, <p>와 같은 태그에 class="ui-li-aside" 속성을 추가하여 오른쪽 상단에 표시되게 할 수도 있음

컨텐트 형식 적용하기

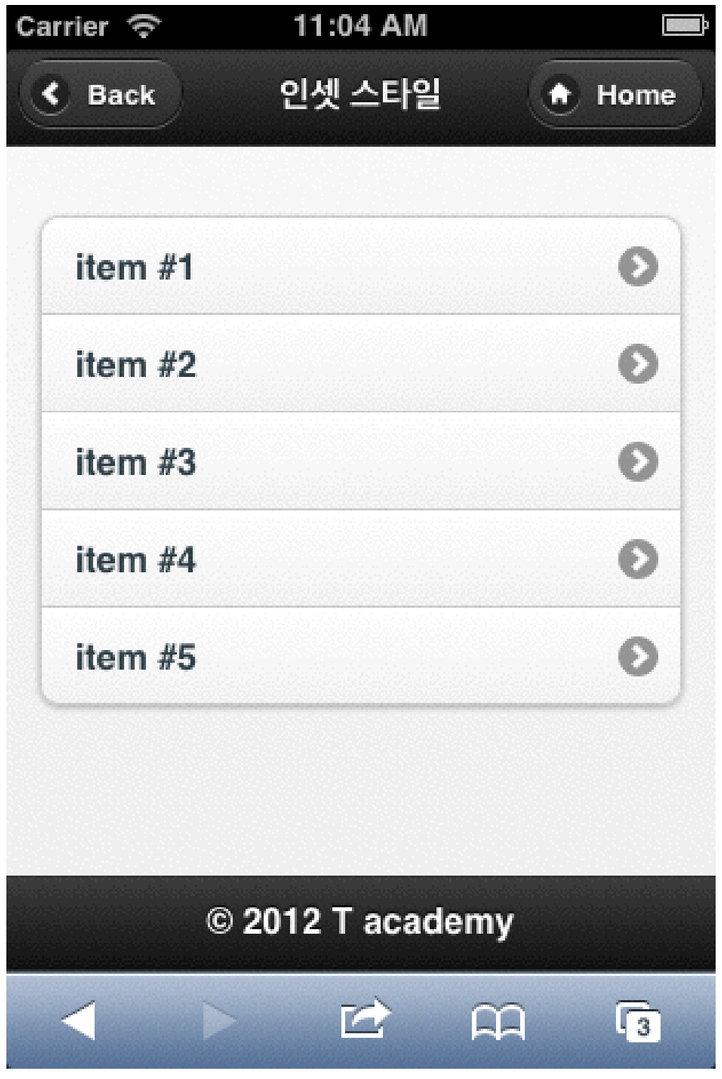

Carrier
11:04 AM
Back
리스트 형식 적용
Home
2012년 12월 9일
3
title #1
12:47 오후
subject #1
description #1
title #2
10:13 오전
subject #2
description #2
title #3
07:45 오전
subject #3
description #3
© 2012 T academy
3

인셋 스타일

리스트에 data-inset="true" 속성을 추가하면 인셋 스타일(inset style)의 리스트를 만들 수 있음

이 속성이 적용되면 모서리가 둥글고 여백이 있는 리스트를 표시함

인셋 스타일


Carrier
11:04 AM
Back
인셋 스타일
Home
item #1
item #2
item #3
item #4
item #5
© 2012 T academy

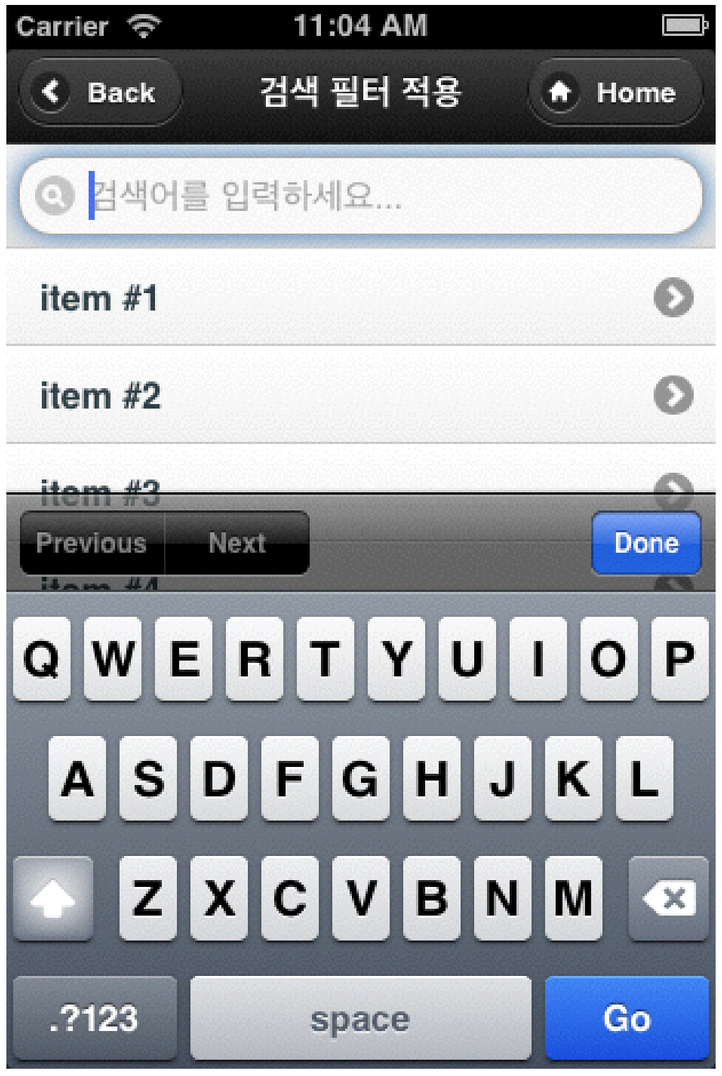
검색 필터 적용 리스트

jQuery Mobile은 리스트 내의 항목을 검색할 있는 검색 필터 구현 기능을 제공

리스트에 data-filter="true" 속성을 추가하면 리스트 상단에 검색 상자가 표시됨

리스트에 data-filter-placeholder 속성과 문자열 값을 함께 추가하면 플레이스 홀더 값으로 표시됨

검색 필터 적용 리스트

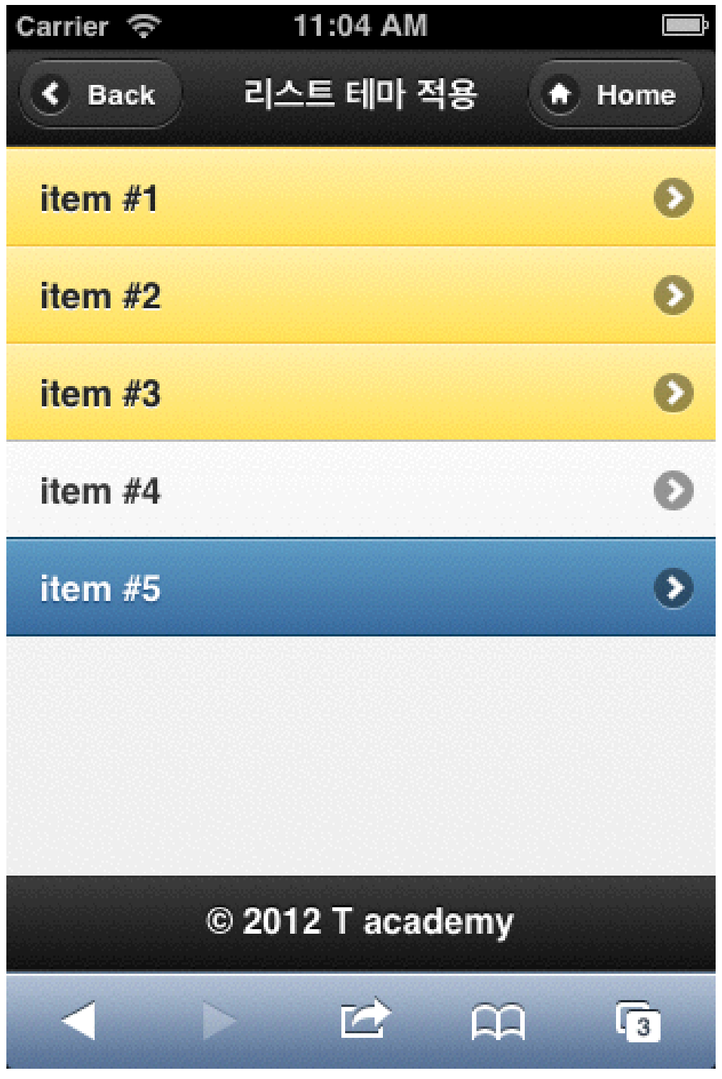

Carrier
11:04 AM
Back
검색 필터 적용
Home
검색어를 입력하세요...
item #1
item #2
item #3
Previous
Next
Done
Q W E R T Y U I O P
A S D F G H J K L
Z X C V B N M
.?123
space
Go

리스트 뷰에 테마 적용하기

리스트 전체의 테마는 리스트에 data-theme 속성을 추가해 변경할 수 있음

data-theme 속성은 <li> 태그에도 지정할 있어서 항목 별로도 다른 테마를 설정할 수 있음

리스트 뷰에 테마 적용하기



리스트 뷰 - 실습 15-01

16강. jQuery Mobile 컴포넌트 - 폼

필드 컨테이너

jQuery Mobile은 스마트 디바이스에서 사용자 입력을 최적화한 폼 요소를 제공

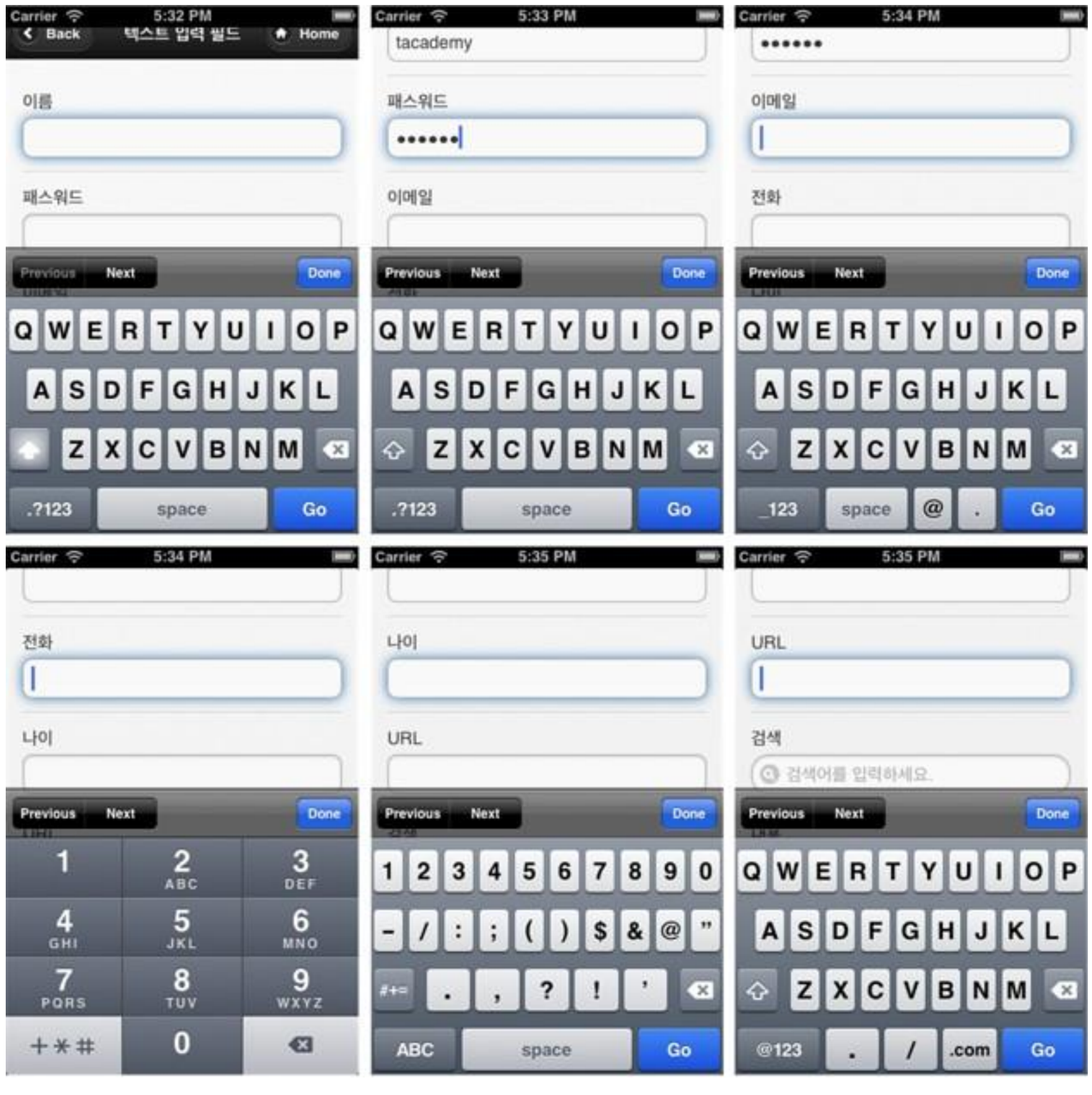
이러한 폼 요소를 사용하기 위해선 data-role="fieldcontain" 속성을 가진 요소 안에 폼 입력 요소를 추가해야 함

이 속성을 적용하면 화면 폭에 따라 라벨의 위치가 폼 요소의 좌측 혹은 좌측 상단 위에 놓이게 됨

텍스트 입력 필드

jQuery Mobile은 텍스트 입력을 위한 <input> 태그의 다양한 타입을 지원하며, 입력 타입에 맞춰서 스마트 디바이스의 소프트 키보드가 제공됨

텍스트 입력 필드

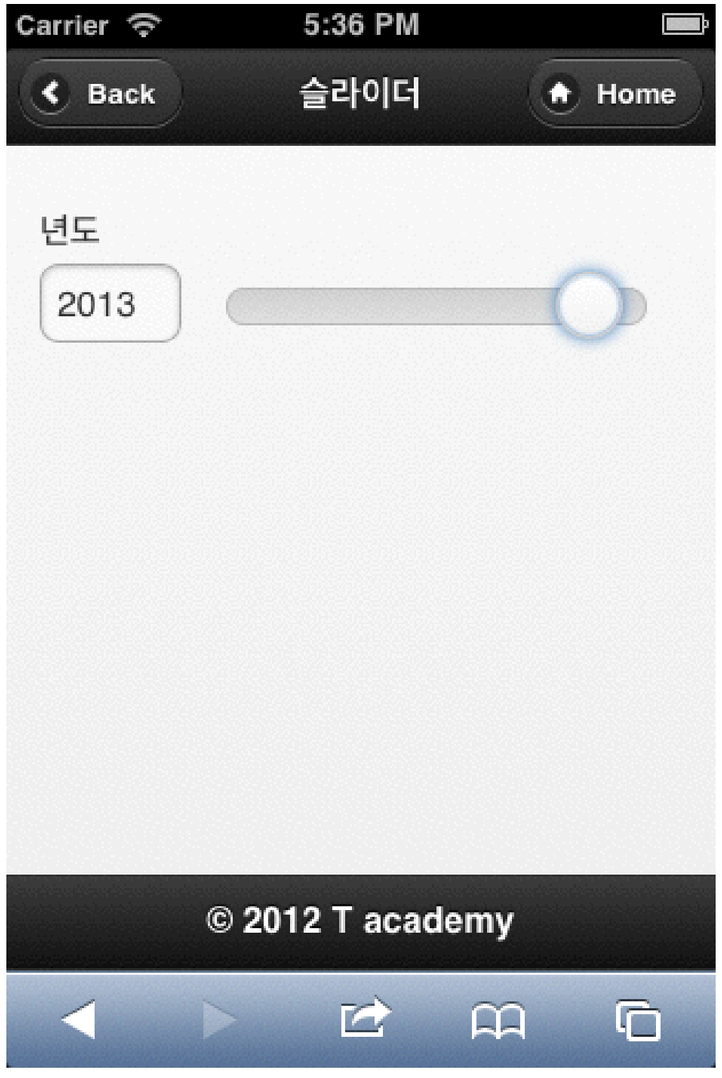

Carrier 5:32 PM
Back
텍스트 입력 필드
Home
이름
패스워드
Previous Next Done
Carrier 5:33 PM
tacademy
패스워드
이메일
Carrier 5:34 PM
이메일
전화
Carrier 5:34 PM
전화
나이
Carrier 5:35 PM
나이
URL
Carrier 5:35 PM
URL
검색
검색어를 입력하세요.

jQuery mobile은 HTML 5에 추가된 range 타입의 input 태그를 지원함

범위를 나타내기 위해 min, max 속성을 지정하고 초기값을 value 속성에 지정

step 속성이 있어서 슬라이딩 시 증감 설정 가능

슬라이더



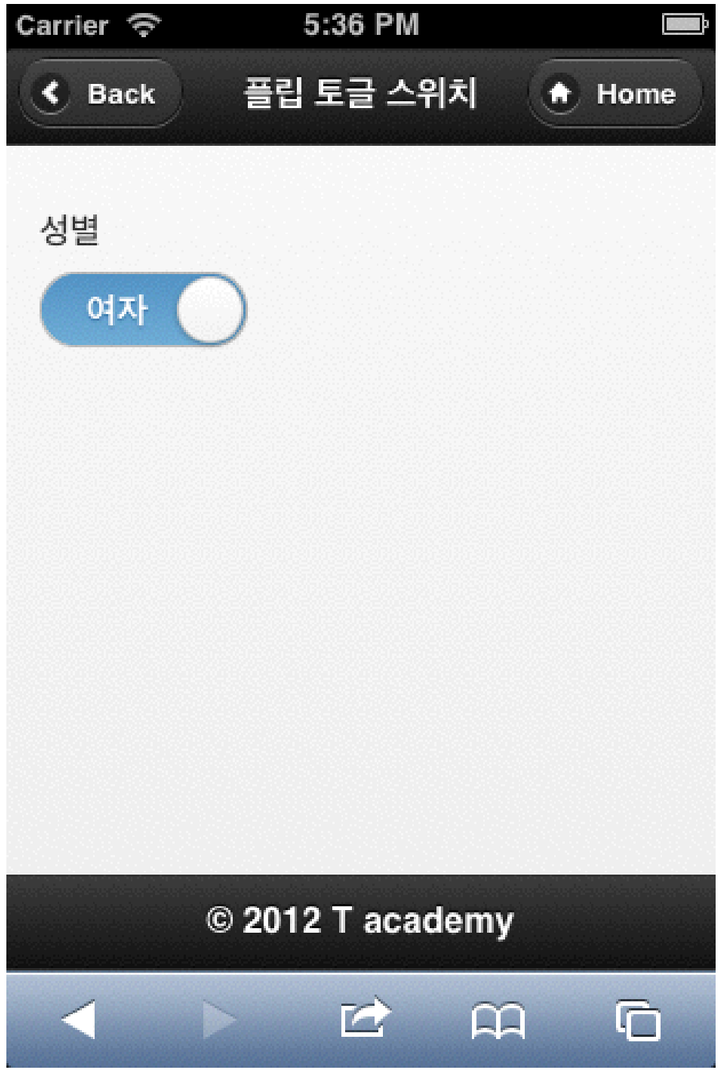
플립 토글 스위치

플립 토글 스위치(flip toggle switch)는 on/off 스위치처럼 동작하는 사용자 인터페이스

select 요소에 data-role=”slider” 속성을 추가하면 사용 가능

일반적으로 스위치 on에 해당하는 값을 두 번째 옵션으로 지정해서 사용함

플립 토글 스위치



라디오 버튼

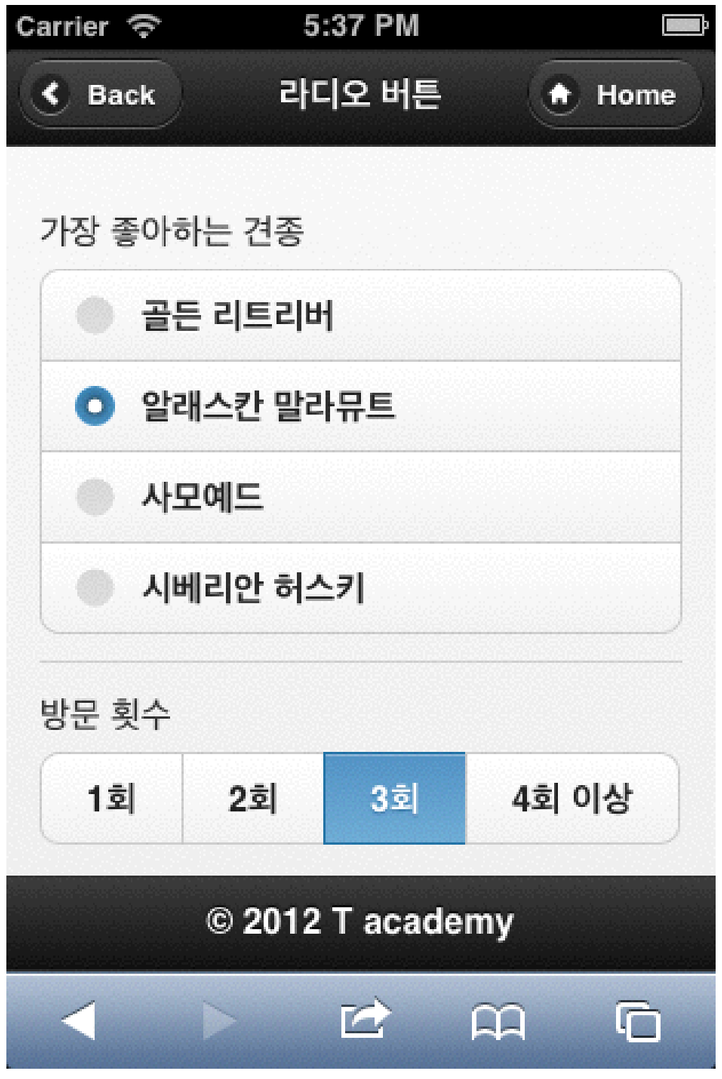
jQuery Mobile에서는 라디오 버튼을 사용하기 위해 radio 타입의 <input> 태그와 <label>을 묶어 사용함

라디오 버튼 그룹을 만들기 위해 data-role="controlgroup" 속성을 가진 <fieldset> 태그 안에 위치시키면 사용하기 쉬운 라디오 버튼이 생성됨

<fieldset> 태그에 <legend> 태그를 추가해 라디오 버튼 그룹에 대한 라벨을 생성함

기본적으로 세로로 출력되며 <fieldset> 태그에 data-type="horizontal" 속성을 추가하면 가로로 배치됨

라디오 버튼

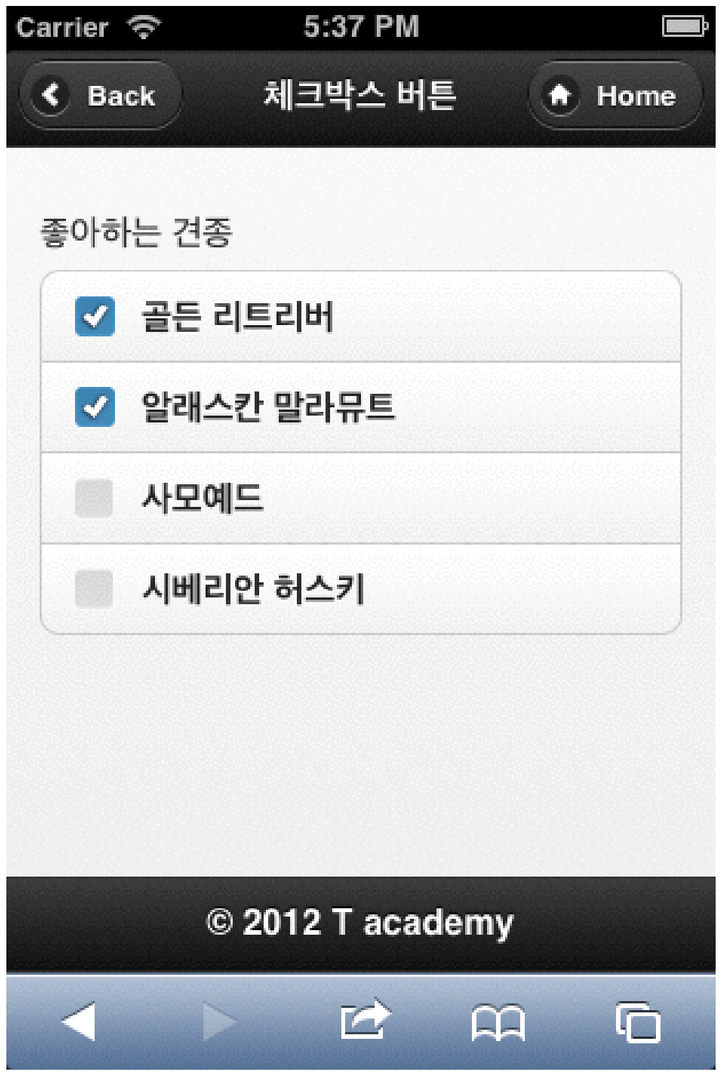

Carrier
5:37 PM
Back
라디오 버튼
Home
가장 좋아하는 견종
골든 리트리버
알래스칸 말라뮤트
사모예드
시베리안 허스키
방문 횟수
1회
2회
3회
4회 이상
© 2012 T academy

체크박스 버튼

jQuery Mobile에서는 체크박스 버튼을 사용하기 위해 checkbox 타입의 <input> 태그와 <label>을 묶어 사용

체크박스 버튼 그룹을 만들기 위해 data-role=”controlgroup” 속성을 가진 <fieldset> 태그 안에 위치시키면 사용하기 쉬운 체크박스 버튼이 생성됨 <fieldset> 태그에 <legend> 태그를 추가해 체크박스 버튼 그룹에 대한 라벨을 생성함

체크박스 버튼


Carrier
5:37 PM
Back
체크박스 버튼
Home
좋아하는 견종
골든 리트리버
알래스칸 말라뮤트
사모예드
시베리안 허스키
© 2012 T academy

셀렉트 메뉴는 네이티브 옵션 메뉴를 기반으로 동작

data-native-menu="false" 속성을 지정하면 jQuery Mobile 이 제공하는 팝업 방식의 셀렉트 메뉴를 사용

다중 선택 기능은 data-native-menu 속성의 값이 false일 경우에 만 가능하며, mutiple="mutiple" 속성을 설정해 사용

셀렉트 메뉴


Carrier
5:38 PM
Back
셀렉트 메뉴
Home
가장 좋아하는 대형 견종
선택하세요
가장 좋아하는 중소형 견종
선택하세요
Previous
Next
Done
✓ 선택하세요
골든 리트리버
알래스칸 말라뮤트
사모예드
시베리안 허스키


Carrier
5:38 PM
Back
셀렉트 메뉴
Home
가장 좋아하는 대형 견종
선택하세요
포메라니안
웰시코기
푸들
비글
가장 좋아
© 2012 T academy


Carrier
5:39 PM
Back
셀렉트 메뉴
Home
가장 좋아하는 대형 견종
알래스칸 말라뮤트
가장 좋아하는 중소형 견종
포메라니안, 웰시코기
2
© 2012 T academy

폼에 테마를 지정하기 위해서는 폼을 포함하고 있는 data-role이 적용된 요소나 페이지 전체에 테마를 적용하는 방법과 입력 요소에 개별적으로 테마를 적용하는 방법이 가능

폼에 테마 적용하기


Carrier 5:26 PM
폼 테마 적용
Home
텍스트 입력 필드
입력하세요.
슬라이더
0
플립 토글 스위치
Off
라디오 버튼(세로)
© 2012 T academy
Carrier 5:27 PM
폼 테마 적용
Home
텍스트 입력 필드
입력하세요.
슬라이더
0
플립 토글 스위치
Off
라디오 버튼(세로)
© 2012 T academy
Carrier 5:27 PM
폼 테마 적용
Home
텍스트 입력 필드
입력하세요.
슬라이더
0
플립 토글 스위치
Off
라디오 버튼(세로)
© 2012 T academy
Carrier 5:28 PM
폼 테마 적용
Home
텍스트 입력 필드
입력하세요.
슬라이더
0
플립 토글 스위치
Off
라디오 버튼(세로)
© 2012 T academy
Carrier 5:29 PM
폼 테마 적용
Home
텍스트 입력 필드
입력하세요.
슬라이더
0
플립 토글 스위치
Off
라디오 버튼(세로)
© 2012 T academy

폼 - 실습 16-01

17강. jQuery Mobile 컴포넌트
- 컨텐트 서식 적용, 테마롤러

레이아웃 그리드

스마트 디바이스는 스크린 폭이 그다지 넓지 않기 때문에 다중 컬럼 레이아웃을 사용하는 것을 일반적으로 비권장

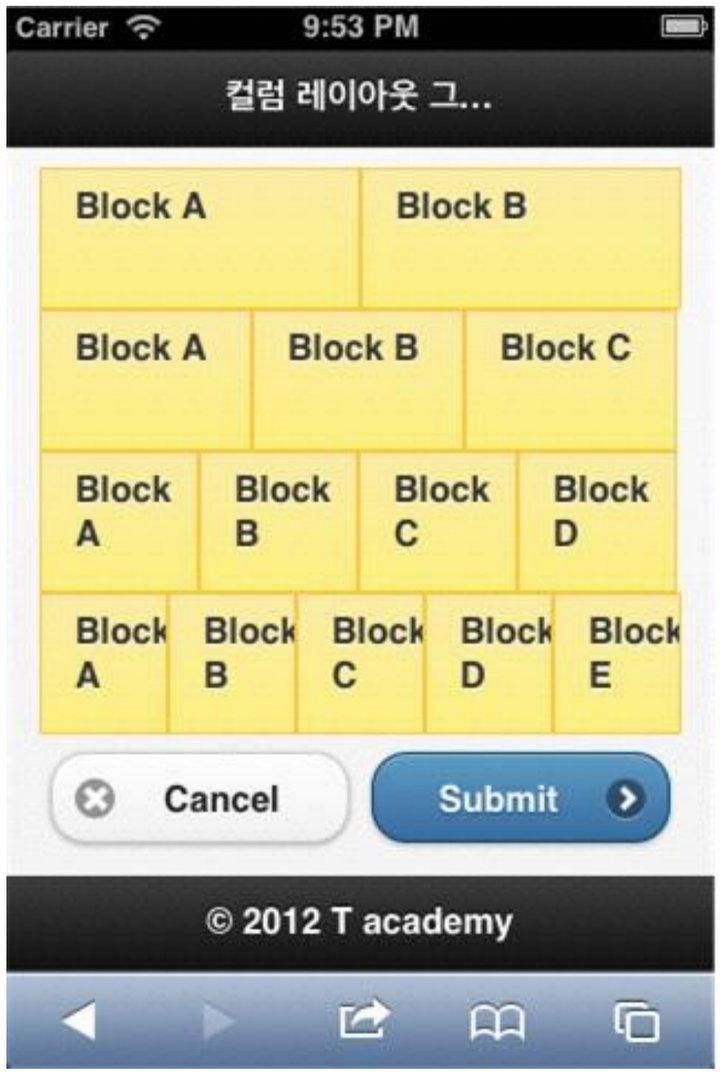
버튼이나 네비게이션 탭과 같이 옆으로 작은 요소들을 배치해야 하는 경우 jQuery Mobile은 ui-grid로 불리는 블록 스타일 클래스를 통해 CSS기반의 다중 컬럼 분할 기능을 제공함

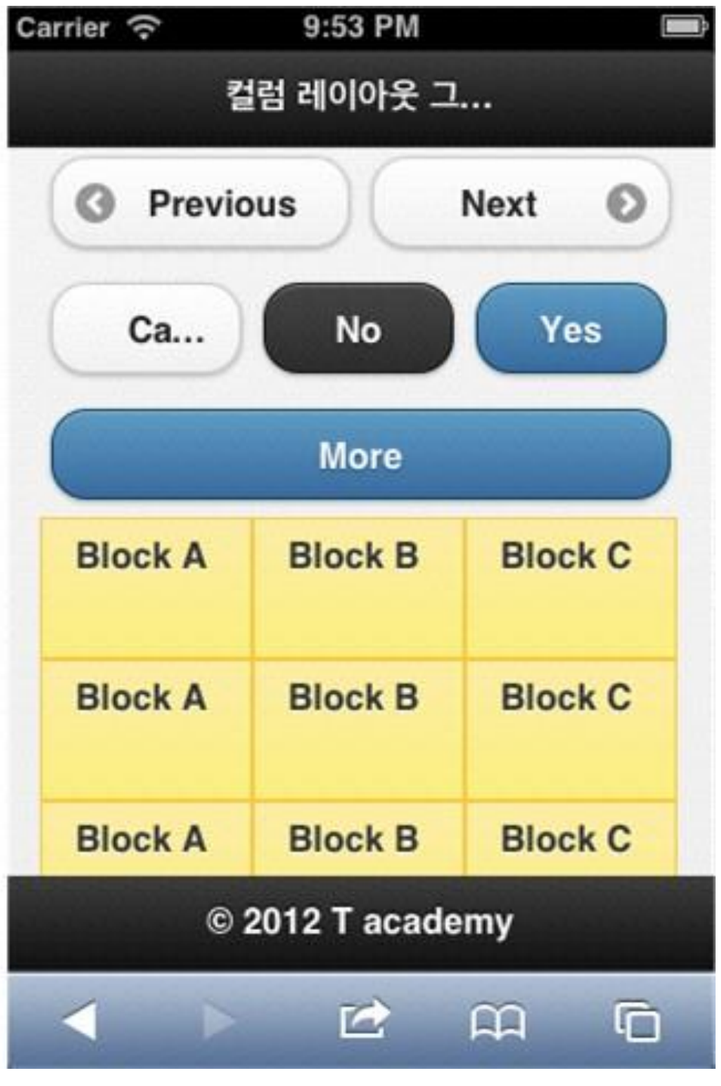
레이아웃 그리드

클래스	설명
ui-grid-a	두 개의 컬럼을 균등 분할해 레이아웃을 구성합니다.
ui-grid-b	세 개의 컬럼을 균등 분할해 레이아웃을 구성합니다.
ui-grid-c	네 개의 컬럼을 균등 분할해 레이아웃을 구성합니다.
ui-grid-d	다섯 개의 컬럼을 균등 분할해 레이아웃을 구성합니다.

그리드는 폭을 100% 사용하고, 패딩과 마진이 없으며, 보더와 백그라운드가 없는 보이지 않는 컨테이너임
그리드 컨테이너 내에서 자식 요소는 순차적으로 ui-block-a/b/c/d/e 클래스를 지정하여 배치함
다중 행을 구성할 때 ui-block-a 클래스가 적용된 요소가 각 행의 맨 앞에 배치됨

레이아웃 그리드


Carrier
9:53 PM
컬럼 레이아웃 그...
Block A
Block B
Block A
Block B
Block C
Block A
Block B
Block C
Block D
Block E
Cancel
Submit
© 2012 T academy


Carrier
9:53 PM
컬럼 레이아웃 그...
Previous
Next
Ca...
No
Yes
More
Block A
Block B
Block C
Block A
Block B
Block C
Block A
Block B
Block C
© 2012 T academy

콜렙서블 컨텐트 블록

data-role="collapsible" 속성을 이용하면 탭(tap) 조작으로 내용을 접었다 폈다하는 콜렙서블 컨텐트 블록(collapsible content block)을 만들 수 있음

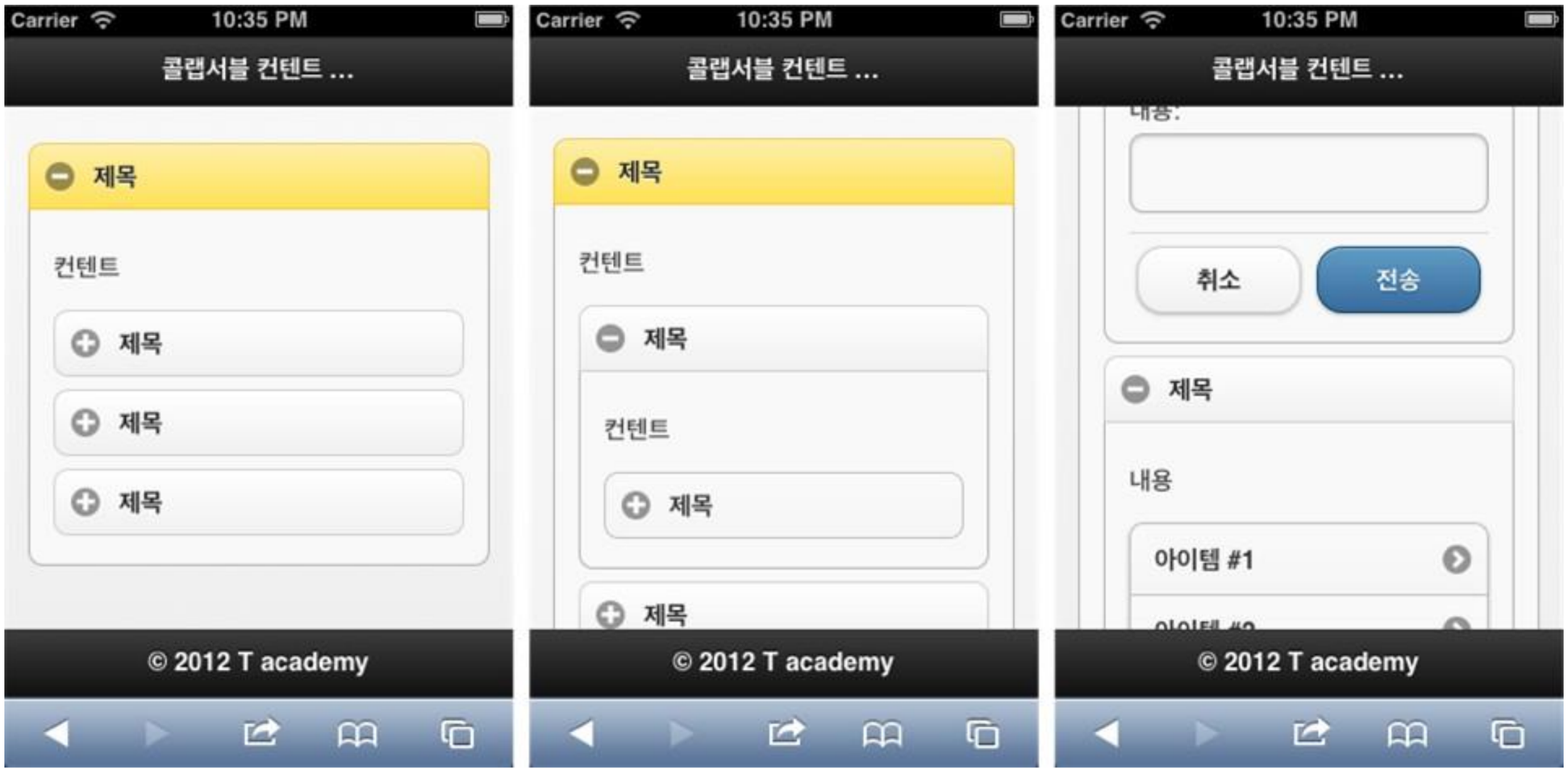
콜렙서블 컨텐트 블록에서는 h1 ~h6 요소를 이용해 제목을 설정

콜렙서블 컨텐트 블록은 블록이 접힌 상태를 기본값으로 가지며, data-collapsed="false" 속성을 지정해 펼친 상태를 표시할 수 있음

기본적으로 인셋이 적용되어 있기 때문에 인셋을 제거하기 위해선 data-inset="false" 속성을 지정하면 됨

data-mini="true" 속성으로 컴팩트 버전을 만들 수 있으며, data-collapsed-icon 과 data-expanded-icon 속성을 이용해 아이콘을 변경할 수 있음

콜랩서블 컨텐트 블록

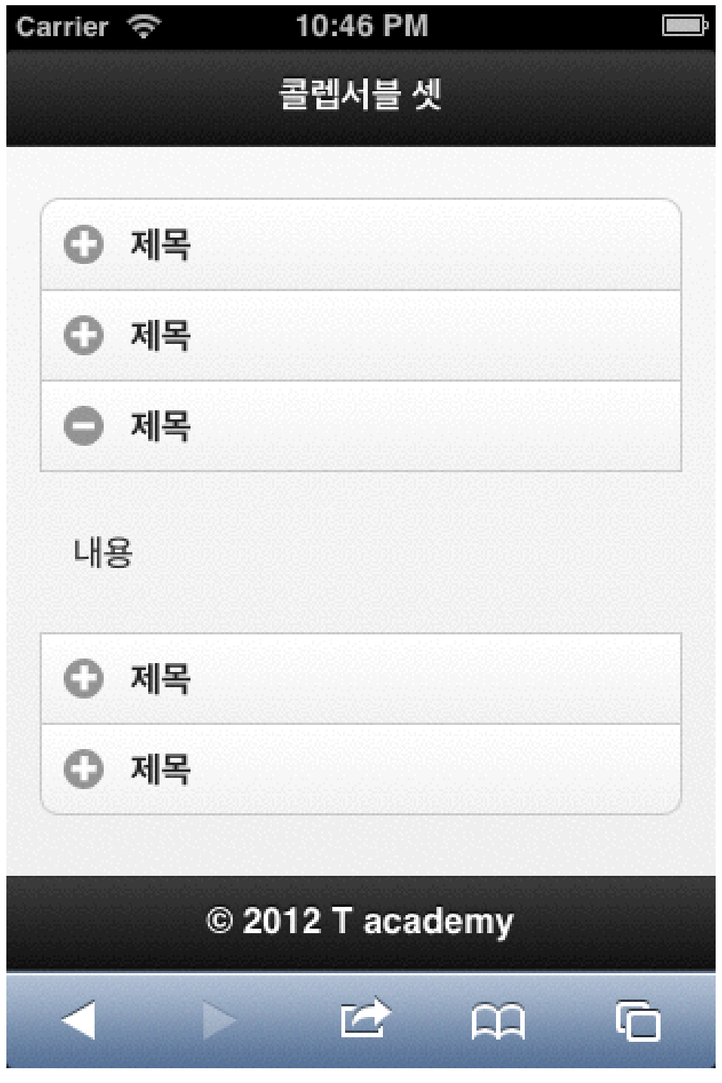

Carrier
10:35 PM
콜랩서블 컨텐트 ...
제목
컨텐트
제목
제목
제목
© 2012 T academy
Carrier
10:35 PM
콜랩서블 컨텐트 ...
제목
컨텐트
제목
컨텐트
제목
제목
© 2012 T academy
Carrier
10:35 PM
콜랩서블 컨텐트 ...
취소
전송
제목
내용
아이템 #1
© 2012 T academy

콜렙서블 셋(아코디언)

data-role=”collapsible-set” 속성을 이용하면 탭(tap) 조작으로 내용을 접었다 폈다하는 콜렙서블 셋(collapsible set)을 생성

콜렙서블 셋은 모든 블록이 접힌 상태를 기본값으로 가지며, data-collapsed=”false” 속성을 지정한 블록은 펼친 상태를 표시

콜렙서블 셋(아코디언)

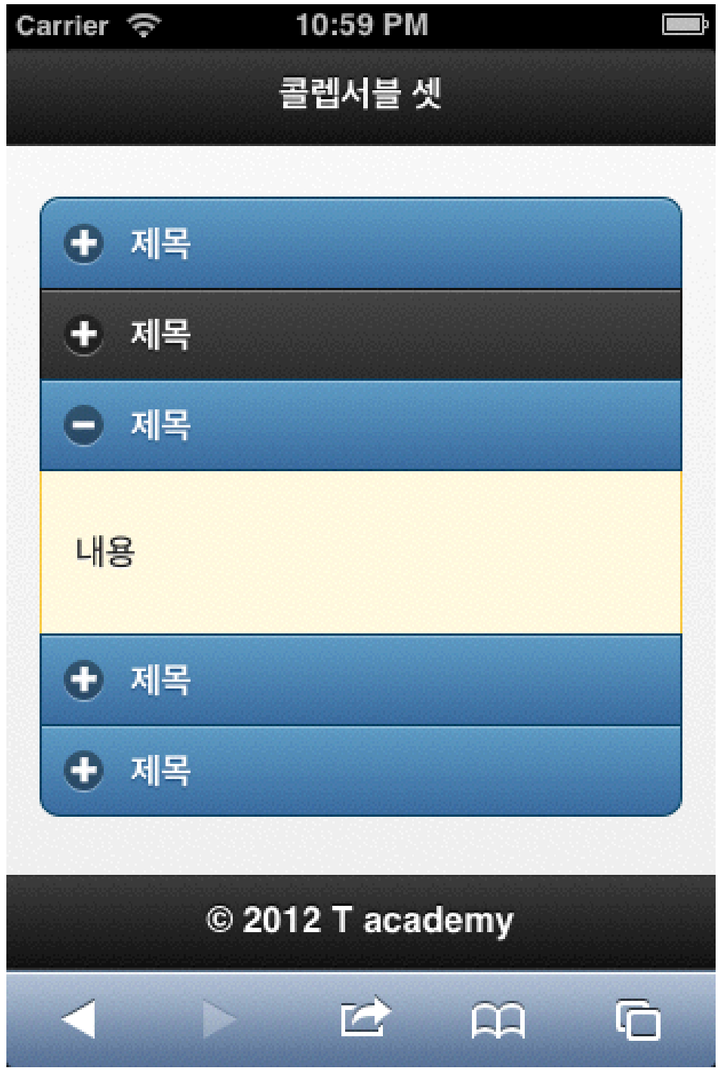


콜랩서블 컨텐트 블록과 콜렙서블 셋에 테마 적용하기

콜렙서블 셋의 data-theme와 data-content-theme 속성을 사용하면 제목 영역의 테마와 컨텐트 영역의 테마를 일괄 지정할 수 있음

또한 콜렙서블 컨텐트 블록마다 이 속성을 적용하면 개별적인 테마 구성이 가능함

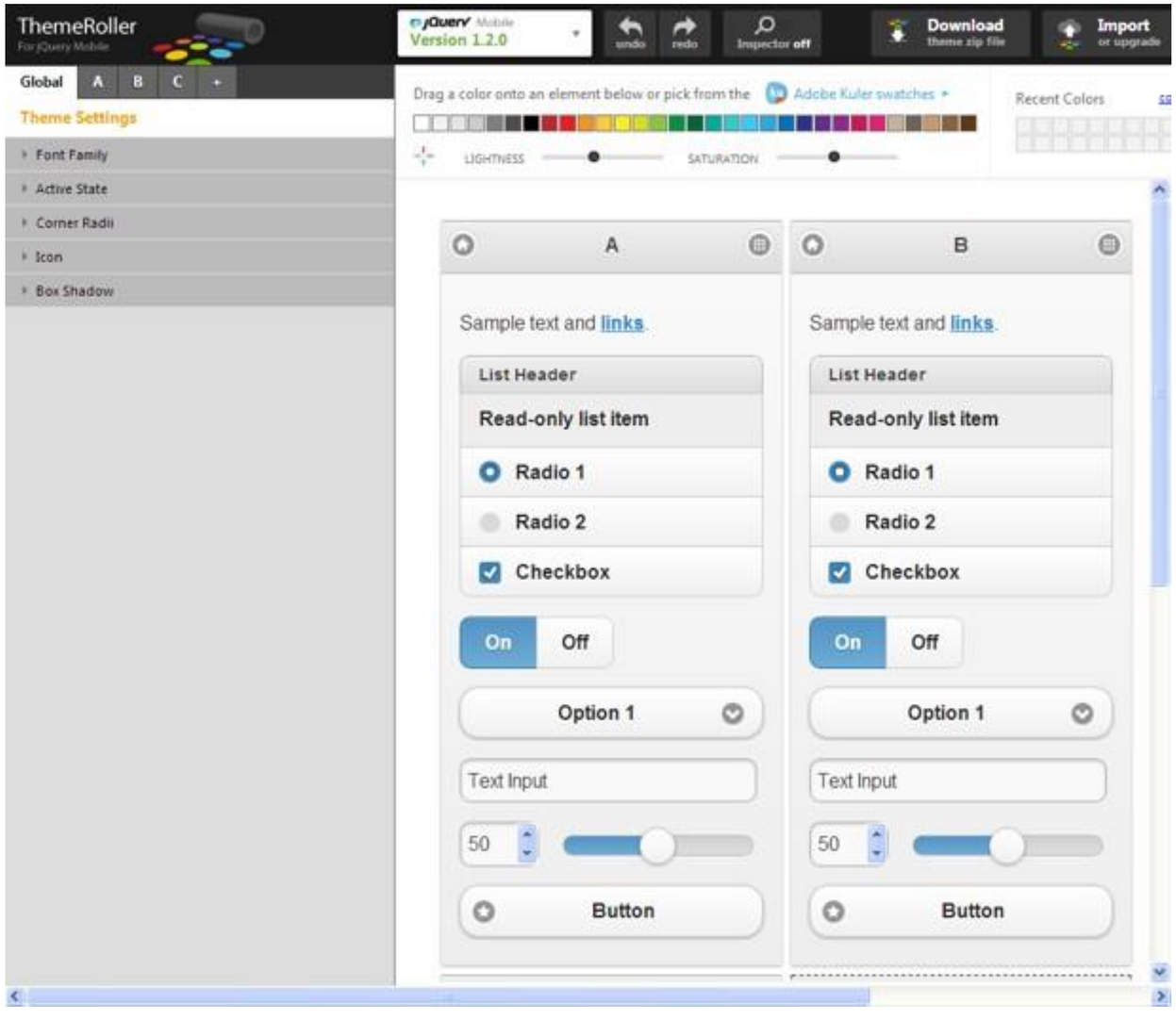
콜랩서블 컨텐트 블록과 콜렙서블 셋에 테마 적용하기



jQuery Mobile에서는 테마롤러라는 온라인 도구를 이용해 직관적으로 테마를 디자인할 수 있는 도구를 제공함

이 도구를 통해 테마 디자인 작업을 완료하면 사용자 정의 테마의 CSS 파일을 다운로드 받을 수 있음

테마롤러 디자인하기 - 테마롤러 온라인 도구



http://themeroller.jquerymobile.com/

컨텐트 서식 적용 및 테마롤러 사용하기 - 17-01.html

SK planet

동GO동락 | T아카데미

18강. jQuery Mobile 이벤트

mobileinit 이벤트

jQuery Mobile이 구동되면 document 객체에서 mobileinit 이벤트가 트리거 되는데, on() 메소드에서 이 mobileinit 이벤트에 대한 이벤트 핸들러를 등록하면 jQuery Mobile 환경설정에 대한 사용자 초기화가 가능함

```
$(document).on("mobileinit", function(event){
 // 사용자 정의 환경 설정 초기화를 실행합니다.
});
```

mobileinit 이벤트

스크립트는 mobileinit 이벤트가 jQuery Mobile 라이브러리 로딩 시 발생하므로, jQuery 라이브러리 참조 구문과 jQuery Mobile 라이브러리 참조 구문 사이에 위치시킴

```
<script src="js/jquery-1.11.1.js"></script>
<script src="js/custom-scripting.js"></script>
<script src="js/jquery.mobile-1.4.2.js"></script>
```

mobileinit 이벤트는 페이지 컨테이너 단위의 초기화가 아닌 전체 웹 앱의 초기화를 의미하며 앱이 최초 구동 시 한 번 발생하는 이벤트임

기본환경 설정

jQuery Mobile의 기본 환경 설정은 mobileinit 이벤트 발생 시 이벤트 핸들러에서 $.extend 메소드로 $.mobile 객체를 확장해 기본 설정을 변경 가능

```
$(document).on("mobileinit", function(){
 $.extend(  $.mobile , {
   foo: bar
 });
});
```

또는

```
$(document).on("mobileinit", function(){
 $.mobile.foo = bar;
});
```

기본환경 설정

설정 프로퍼티	기본값	설명
activeBtnClass	“ui-btn-active”	활성화된 버튼에 사용되는 CSS 클래스
activePageClass	“ui-page-active”	현재 페이지에 대한 CSS 클래스
ajaxEnabled	true	Ajax 방식의 페이지 전환을 사용할지에 대한 설 정
allowCrossDomainPages	false	기존의 페이지 전환 방식을 도메인 간에서도 사 용할 지에 대한 설정. 폰 갭을 사용하면 true로 설정해야 함
autoInitializePage	true	DOM이 준비되면 $.mobile.initialize() 메소 드를 자동으로 호출할 지에 대한 설정
defaultDialogTransition	‘pop’	다이얼로그 화면 전환 기본 설정
defaultPageTransition	‘fade’	페이지 전환 기본 설정

기본환경 설정

설정 프로퍼티	기본값	설명
ns	“”	data- 속성에 사용되는 네임스페이스. “myns-”로 지정하면 data- 속성이 모두 data-myns- 속성으 로 바뀜. 이 프로퍼티를 변경하면 테마CSS 변경 이 필요
pageLoadErrorMessage	“Error Loading Page”	Ajax를 통해 페이지 로딩이 실패할 때 보여줄 텍 스트
pageLoadErrorMessageTheme	“e”	에러 메시지박스가 사용하는 테마
transitionFallbacks.[transition]	“fade”	특정 화면 전환의 3D 변환을 지원하지 않는 브라 우저에 대한 대체 화면 전환

페이지 초기화 이벤트 (pagebeforecreate, pagecreate, pageinit)

jQuery Mobile은 페이지의 태그들과 관련된 플러그 인들을 자동 초기화함

페이지 플러그 인에 의해 제어되는 자동 초기화는 실제 자동 초기화 전과 후에 이벤트를 디스패치(event dispatch)하는데, 이 때 사전 초기화와 사후 초기화를 통해 페이지를 조작할 수 있음

이 이벤트는 매번 페이지가 나타나고 사라질 때마다 발생하는 페이지 전환 이벤트와 달리 페이지 당 한 번만 발생한다는 점에 주의

jQuery Mobile에서는 페이지를 직접 로딩하든 Ajax를 이용한 페이지 로딩이든 간에 상관없이 작동하는 pageinit 이벤트를 ready 이벤트 대신 사용할 것을 권장함

페이지 초기화 이벤트 (pagebeforecreate, pagecreate, pageinit)

pagebeforecreate ⇨ pagecreate ⇨ pageinit

```
$( '#aboutPage' ).on( 'pagebeforecreate',function(event){
 // 위젯이 자동 초기화(auto-initialization)되기 전 이 페이지를 여기서 조작합니다.
});
$( '#aboutPage' ).on( 'pagecreate',function(event){
 // 페이지가 DOM에 생성되었을 때 pagecreate 이벤트가 발생하며 커스텀 위젯을 만드
는
 // 작업 등을 여기서 합니다.
});
$( '#aboutPage' ).on( 'pageinit',function(event){
 // 페이지에서 필요한 초기화 작업를 여기서 조작합니다.
});
```

페이지 변경 이벤트 (pagebeforechange, pagechange, pagechangefailed)

페이지 간의 네비게이션은 $.mobile.changePage() 메소드 호출을 통해 일어남

이 메소드는 우리가 이동하고자 하는 페이지가 DOM에 로드되고 추가된 후, 현재 페이지와 다음 페이지 사이의 페이지 전환 애니메이션이 시작되도록 하는 것을 담당함

이 과정에서 changePage() 메소드는 두 개의 이벤트를 발생 시키는데, pagebeforechange 이벤트가 먼저 발생하고, 변경 요청의 성공 여부에 따라 pagechange 이벤트 혹은 pagechangefailed 이벤트가 발생함

pagebeforechange ⇨ pagechange (성공 시)
⇨ pagechangefailed (실패 시)

페이지 전환 이벤트 (pagebeforeshow, pagebeforehide, pageshow, pagehide)

페이지 전환 이벤트는 현재 페이지를 새 페이지로의 변경을 애니메이션 처리하는 데 사용

페이지 전환 이벤트는 페이지가 나타나거나 사라지기 전후 발생

pagebeforehide ⇨ pagebeforeshow ⇨ pagehide ⇨ pageshow

페이지 로드 이벤트 (pagebeforeload, pageload, pageloadfailed)

외부 페이지가 DOM으로 로드될 때 발생하는 이벤트 입니다. 가장 먼저 pagebefor eload 이벤트가 발생하고, 다음으로 pageload나 pageloadfailed 이 벤트가 발생합니다.

pagebeforeload ⇨ pageload
⇨ pageloadfailed

tap, taphold 이벤트

tap 이벤트는 터치 발생후 트리거되는 이벤트이고 taphold는 터치를 계속 유지하면 트리거되는 이벤트

taphold 이벤트 발생을 위한 유지시간은
$.event.special.tap.tapholdThreshold 프로퍼티에 설정

기본 값은 750(ms)

swipe, swipeleft, swiperight 이벤트

1초 안에 수평 30픽셀 이상, 수직 75픽셀 이하 드래그가 발생하면 swipe 이벤트가 발생
모든 swipe 이벤트는 조건을 만족할 때 발생하게 되며, 설정 프로퍼티를 통해 변경이 가능

swipeleft 이벤트는 왼쪽으로 움직이는 스와이프 이벤트가 발생했을 때 트리거 되고, swiperight 이벤트는 오른쪽으로 움직이는 스와이프 이벤트가 발생했을 때 트리거 됨

swipe, swipeleft, swiperight 이벤트

설정 프로퍼티	기본값	설명
$.event.special.swipe.scrollSuppressionThredshold	10px	수평 이동이 이 이상이면 스크롤이 일어 나지 않도록 합니다
$.event.special.swipe.durationThredshold	1000ms	스와이프 이동 최대 시간
$.event.special.swipe.horizontalDistanceThredshold	30px	스와이프 수평이동의 최소값
$.event.special.swipe.verticalDistanceThredshold	75px	스와이프 수직이동의 최대값

orientationchange 이벤트

디바이스가 수직 혹은 수평으로 방향을 바꾸면 orientationchange 이벤트가 발생

이벤트가 발생할 때 window 객체의 orientation 프로퍼티 값을 이용하면 세부 제어가 가능

jQuery Mobile 이벤트 - 실습 18-01.html

SK planet

동GO동락 | T아카데미

19장. jQuery Mobile 유틸리티 메소드

페이지 간의 네비게이션은 $.mobile.changePage() 메소드 호출을 통해 일어남

이 메소드는 우리가 이동하고자 하는 페이지가 DOM에 로드되고 추가된 후, 현재 페이지와 다음 페이지 사이의 페이지 전환 애니메이션이 시작되도록 하는 것을 담당

프로퍼티	타입	기본값	설명
allowSamePageTrans ition	boolean	false	동일한 현재 페이지에 대 한 요청에 페이지 전환 적
changeHash	boolean	true	로케이션 바의 해시를 갱 신할 지 여부
data	object, string	undefined	Ajax 페이지 요청을 통해 전달할 데이터
dataUrl	string	undefined	chnagePage() 완료 시 로케이션을 갱신 할 때 사
pageContainer	jQuery	$.mobile.pageContainer	페이지를 포함하는 요소
reloadPage	boolean	false	강제로 페이지를 리로드
reverse	boolean	false	페이지가 나타날 때의 방
role	string	undefined	페이지를 나타낼 때 사용 하는 data-role 의 기본
showLoadMsg	boolean	true	외부 페이지 로딩 시 로딩 메시지 표시 여부
transition	string	$.mobile.defaultPageTran sition	페이지를 나타낼 때의 기 본 전환 애니메 이션
type	string	“get”	페이지 요청 메소드 (get/ post)

외부 페이지를 로딩해 DOM에 추가하는 역할을 담당changePage()가 내부적으로 호출하는 이 메소드는 첫 번째 인자로 전달된 URL에 해당하는 페이지를 백그라운드에서 로딩하는 역할을 하며 현재 페이지에 영향을 주지는 않음 두 번째 인자로 프로퍼티를 이용해 객체 리터럴을 전달

프로퍼티	타입	기본값	설명
data	object, string	undefined	Ajax 페이지 요청을 통 해 전달할 데이터
loadMsgDelay	number	50	로딩 메시지가 나타나기 전의 지연 시간(ms)
pageContainer	jQuery collection	$.mobile.pageContainer	로드된 후 페이지를 포함 하는 요소
reloadPage	boolean	false	강제로 페이지를 리로드
role	string	undefined	페이지를 나타낼 때 사용하는 data-role의 기본 값
showLoadMsg	boolean	false	외부 페이지 로딩 시 로 딩 메시지 표시 여부
type	string	“get”	페이지 요청 메소드 (get/post)

$.mobile.loader.prototype.options의 옵션들을 이용하거나 프로퍼티를 이용해 객체 리터럴을 전달하면 페이지 로딩 메시지를 표시하거나 보이지 않게 함

프로퍼티	타입	기본값	설명
theme	string	“a”	메시지에 대한 테마 스와치
text	string	“loading”	로딩 메시지 문자열
textonly	boolean	false	true일 경우 메시지가 나타나면 스피너 이미지가 사라짐
textVisible	boolean	false	true일 경우스피너 아래 메시지가 보임
html	string	“”	로더의 innnerHTML을 교체


Utilities
메시지 로더
Default Loader
Text Only
Theme a
페이지 로딩 중 (Theme e)
Theme d
Theme e
Custom HTML
Hide
© 2012 T academy

실습
- 페이지 로딩 메시지 처리 적용

매개변수로 전달 받은 다양한 URL 문자열 값들을 파싱해 URL의 각 구성 컴포넌트들을 쉽게 접근 할 수 있는 객체를 반환하는 유틸리티 메소드

URL 파싱 중 프로토콜이나 호스트와 같은 컴포넌트들이 생략되었을 경우 해당 컴포넌트는 공백 문자열로 처리됨

프로퍼티	설명
hash	'#' 문자로 시작되는 URL의 부분
host	URL에서 host와 port
hostname	URL에서 호스트의 이름
href	파싱된 본래 URL
pathname	URL에 의해 참조되는 파일 또는 디렉토리의 경로
port	URL에 기술된 포트 번호
protocol	URL의 프로토콜. ':' 문자로 끝남
search	URL에서 '?'문자로 시작하는 질의
authority	URL의 username, password, host
directory	pathname에 포함된 디렉토리명. filename은 제외
domain	URL의 protocol과 authority
filename	pathname에 포함된 파일명. directory는 제외
hrefNoHash	hash를 제외한 URL
hrefNoSearch	search와 hash를 제외한 URL
password	authority에 포함된 암호

$.mobile.path.parseUrl()

```
var obj =
$.mobile.path.parseUrl("http://forbeto:password@mycompany.com:8080/mail/inbox?msg=1234");
//  obj.href: http://forbeto:password@mycompany.com:8080/mail/inbox?msg=1234&type=unread#
msg-content
//  obj.hrefNoHash:   http://forbeto:password@mycompany.com:8080/mail/inbox?msg=1234&type=unread
//  obj.hrefNoSearch: http://forbeto:password@mycompany.com:8080/mail/inbox
//  obj.domain: o    http://forbeto:password@mycompany.com:8080
//  bj.protocol: o    http:
//  bj.authority: o   forbeto:password@mycompany.com:8080
//  bj.username:       forbeto
//  obj.password:     password mycompa
//  obj.host: obj.  ny.com:8080
//  hostname: obj     mycompany.com
//  .port: obj.path8080
//  name: obj.dir       /mail/inbox
//  ectory: obj.file   /mail/
//  name: obj.sea    inbox
//  rch: obj.hash:   ?msg=1234&type=unread
//                   #msg-content
```

파일이나 디렉토리의 상대 경로를 절대 경로로 변환해주는 유틸리티 메소드

첫 번째 매개변수는 문자열로 상대 경로의 파일 또는 디렉토리를 전달

두 번째 매개변수는 변환을 적용할 절대 경로를 문자열로 전달

```
// /a/b/c/file.html 을 반환
var absPath =
$.mobile.path.makePathAbsolute("file.html", "/a/b/c/
bar.html");
```

```
// /a/foo/file.html 을 반환
var absPath = $.mobile.path.makePathAbsolute("../../
foo/file.html", "/a/b/c/bar.html");
```

상대 URL을 절대 URL로 변환해주는 유틸리티 메소드

첫 번째 매개변수는 문자열로 상대 URL을 전달

두 번째 매개변수는 변환을 적용할 절대 URL을 문자열로 전달

```
// http://foo.com/a/b/c/file.html 을 반환
var absUrl = $.mobile.path.makeUrlAbsolute("file.html", "http://foo.com/a/b/c/test.html");

// http://foo.com/a/foo/file.html 을 반환
var absUrl = $.mobile.path.makeUrlAbsolute("../../foo/file.html", "http://foo.com/a/b/c/
test.html");

// http://foo.com/bar/file.html 을 반환
var absUrl = $.mobile.path.makeUrlAbsolute("//foo.com/bar/file.html", "http://foo.com/a/
b/c/test.html");

// http://foo.com/a/b/c/test.html?a=1&b=2 을 반환
var absUrl = $.mobile.path.makeUrlAbsolute("?a=1&b=2", "http://foo.com/a/b/c/
test.html");

// http://foo.com/a/b/c/test.html#bar 을 반환
var absUrl = $.mobile.path.makeUrlAbsolute("#bar", "http://foo.com/a/b/c/test.html");
```

$.mobile.path.isSameDomain()

두 개의 URL이 서로 같은 도메인인지를 비교해주는 유틸리티 메소드

도메인이 같을 경우 true를 반환

```
// true 를 반환
var same = $.mobile.path.isSameDomain("http://foo.com/a/file.html", "http://foo.com/a/b/c/test.html");
// false 를 반환
var same = $.mobile.path.isSameDomain("file://foo.com/a/file.html", "http://foo.com/a/b/c/test.html");
// false 를 반환
var same = $.mobile.path.isSameDomain("https://foo.com/a/file.html", "http://foo.com/a/b/c/test.html");
// false 를 반환
var same = $.mobile.path.isSameDomain("http://foo.com/a/file.html", "http://bar.com/a/b/c/test.html");
```

$.mobile.path.isRelativeUrl()

URL이 상대 URL인지 비교해주는 유틸리티 메소드

상대 URL일 경우 true를 반환하고 절대 URL일 경우 false를 반환

```
// false 를 반환
var isRel = $.mobile.path.isRelativeUrl("http://foo.com/a/file.html");
// true 를 반환
var isRel = $.mobile.path.isRelativeUrl("//foo.com/a/file.html");
// true 를 반환
var isRel = $.mobile.path.isRelativeUrl("/a/file.html");
// true 를 반환
var isRel = $.mobile.path.isRelativeUrl("file.html");
// true 를 반환
var isRel = $.mobile.path.isRelativeUrl("?a=1&b=2");
// true 를 반환
var isRel = $.mobile.path.isRelativeUrl("#foo");
```

$.mobile.path.isAbsoluteUrl()

URL이 절대 URL인지 비교해주는 유틸리티 메소드

절대 URL일 경우 true를 반환하고 상대 URL일 경우 false를 반환

```
// true 를 반환
var isAbs = $.mobile.path.isAbsoluteUrl("http://foo.com/a/file.html");
// false 를 반환
var isAbs = $.mobile.path.isAbsoluteUrl("//foo.com/a/file.html");
// false 를 반환
var isAbs = $.mobile.path.isAbsoluteUrl("/a/file.html");
// false 를 반환
var isAbs = $.mobile.path.isAbsoluteUrl("file.html");
// false 를 반환
var isAbs = $.mobile.path.isAbsoluteUrl("?a=1&b=2");
// false 를 반환
var isAbs = $.mobile.path.isAbsoluteUrl("#foo");
```

[온라인] JavaScript 기본

Ver1.0 | 2015년 5월 19일

펴 낸 곳 | T아카데미

책임편집 | 에스피테크놀러지㈜ 교육사업팀

주　　소 | 151-919 서울특별시 관악구 낙성대동 서울대연구공원
SK텔레콤연구소 2층

전자우편 |tacademy@skplanet.com

전화번호 | 070-7461-6931~6933