

2017 Spring

Human-Computer Interaction

Javascript & jQuery (2)

—

Jaemin Jo 조재민

Human-Computer Interaction Laboratory
Seoul National University

Client Technologies

- HTML: HyperText Markup Language
 - 문서의 구조
- CSS: Cascading Style Sheets
 - 문서의 디자인 (표현)
- Javascript
 - 문서의 기능

Javascript

- Java와 아무 연관 없음



- 최신 명세: ECMAScript 2016
 - <http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-262.pdf>
- 웹 브라우저에 탑재된 엔진이 해석하고 실행
 - Google Chrome: V8 JavaScript Engine

Javascript의 특징

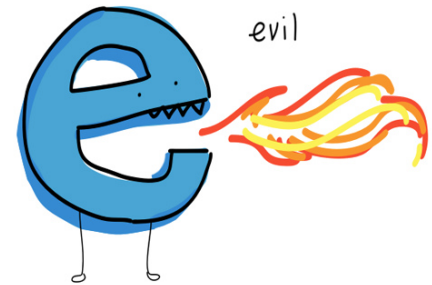
- Dynamically Typed

```
var a = 1  
a = 'string'  
a = [1, 2, 3]  
a = {q: 1, w: 2}
```

- Prototype-based
 - Object-based
 - No class!
- Functional
 - A function is first-class

Javascript의 특징

- 웹 환경의 특성상 과거 브라우저를 고려해야 함
 - 브라우저 별로 동작하지 않는 메소드 등이 존재
 - 따라서, 최신 명세들을 마음껏 쓸 수 없음
 - Compatibility Table (<https://kangax.github.io/compat-table/es6>)
 - Polyfill
- 대안: Transpilation
 - CoffeeScript, TypeScript (Microsoft)
- 따라서 실습 시간에는 *적절히 오래 된 (안전한) 코드를* 배울 것입니다.



Javascript 불러오기

- `<script></script>` 태그를 이용

```
<link rel="stylesheet" type="text/css" href="css/styles.css">  
<script src="https://code.jquery.com/jquery-3.1.0.min.js"></script>  
<script src="js/main.js"></script>
```

- 일반적으로, CSS 파일들은 `<head>` 태그 안에서
- JS 파일들은 `<body>` 태그의 끝에서
 - 왜? Progressive loading
- `<script>` 태그를 만나는 순서대로 JS가 실행됨
 - 의존성에 따른 `<script>` 태그의 순서가 중요!
 - 이후에 페이지가 완전히 로딩된 후에 실행시키는 방법을 배울 것임

Javascript 문법

- Variable (*var*)
- Loops (*for, while*)
- Control Flow (*if*)

```
var sum = 0;

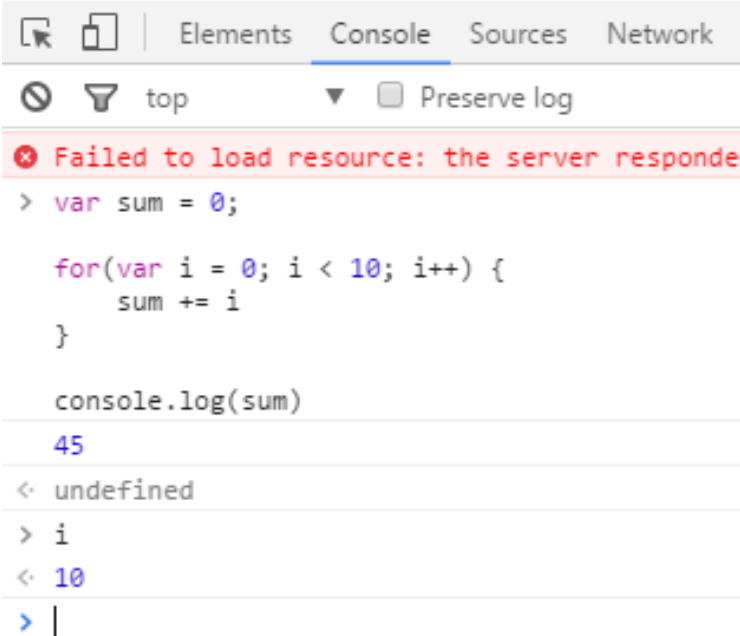
for(var i = 0; i < 10; i++) {
    sum += i
}

console.log(sum)
```

화면에 출력되는 것이 아님! 개발자 콘솔에 출력됨

자바스크립트 콘솔

- View -> Developer -> Developer Tools로 개발자 도구 실행
 - MacOS: alt+command+i, Windows: F12
 - 상단의 탭 중에 Console을 선택
 - 자바스크립트를 쉽게 연습해 볼 수 있음
 - 설정 목록에서 (Windows의 경우 F1키) Network아래의 Disable cache 에 체크할 것
 - 그 다음, 개발자 도구를 켜 상태에서 작업



The screenshot shows the Chrome Developer Tools Console. The 'Console' tab is selected. At the top, there are icons for a mouse cursor, a document, and a filter. Below these are the tabs 'Elements', 'Console', 'Sources', and 'Network'. The 'Console' tab shows a red error message: 'Failed to load resource: the server responded with a status of 404 (Not Found)'. Below the error, there is a JavaScript code snippet: `> var sum = 0;`, `for(var i = 0; i < 10; i++) {`, `sum += i`, `}`, and `console.log(sum)`. The output of the code is shown below the code: `45`. Below the output, there is a prompt `< undefined`. Below the prompt, there is a prompt `> i`. Below the prompt, there is a prompt `< 10`. Below the prompt, there is a prompt `> |`.

Javascript 문법

```
var Animal = {  
  Cat: 0,  
  Dog: 1,  
  Frog: 2  
}  
  
var pet = Animal.Cat;  
  
if(pet === Animal.Cat) console.log('야옹')  
else if(pet === Animal.Dog) console.log('멍멍')  
else if(pet === Animal.Frog) console.log('개굴')  
else console.log('??')
```

야옹

Javascript 문법

```
> var name = 'aaabccde';  
   var count = {};  
  
   for(var i = 0; i < name.length; i++) {  
       var char = name[i];  
  
       if(!count[char]) count[char] = 0;  
  
       count[char]++;  
   }  
  
   console.log(count)  
▶ Object {a: 3, b: 1, c: 2, d: 1, e: 1}  
◀ undefined
```

Javascript 문법

- Function

- 자바스크립트에서 가장 어려운 부분
- JS에서 함수는 단지 호출 가능한 객체

```
var Animal = {  
  Cat: 0,  
  Dog: 1,  
  Frog: 2  
}  
  
function getRandomPet() {  
  return Math.floor(Math.random() * 3)  
}  
  
function check(pet) {  
  if(pet === Animal.Cat) console.log('야옹')  
  else if(pet === Animal.Dog) console.log('멍멍')  
  else if(pet === Animal.Frog) console.log('개굴')  
  else console.log('??')  
}
```

```
check(getRandomPet())  
check(getRandomPet())  
check(getRandomPet())
```

멍멍

야옹

개굴

Javascript 문법

- 익명 함수
 - 이름이 없는 함수
 - 변수에 대입해서 이름을 붙여줄 수 있음!

```
var double = function(x) { return x * 2 }  
  
console.log(double(10)) // 20  
  
double = function(x) { return x * 3 }  
  
console.log(double(10)) // 30
```

```
var double = function(x) { return x * 2 }  
  
// vs  
  
function double(x) { return x * 2 }
```

차이는?

Javascript 문법

```
var words = ['I', 'love', 'my', 'dog']  
  
var mapped = words.map(function(x) { return x.length })  
  
console.log(mapped) // [1, 4, 2, 3]  
  
var reduced = mapped.reduce(function(sum, x) { return sum + x }, 0)  
  
console.log(reduced) // 10
```

Javascript 문법

```
var arr = ['a', 'b', 'c']
var obj = {
  0: 'a',
  1: 'b',
  2: 'c'
}

console.log(arr[1], obj[1]) // b b

var obj2 = {}, arr2 = []

arr.forEach(function(value, index) {
  obj2[index] = value
})

var keys = Object.keys(obj) // [0, 1, 2]

keys.forEach(function(key) {
  arr2.push(obj[key])
})

console.log(arr2, obj2)
```

The order is not guaranteed!

HTML & Javascript

- 앞에서 한 것 Python으로 하면 더 편한데요?
- HTML 문서와 인터랙션하기
- DOM (Document Object Model)
 - HTML 문서의 모든 태그들의 구조, 속성, 스타일, 값을 추가, 삭제, 수정할 수 있도록 만든 인터페이스
 - JS에서 (DOM 을 통해) HTML 문서를 조작할 수 있음

HTML & Javascript

```
<h1 id="my-title">HighlightMe</h1>
```

HighlightMe

```
var myTitle = document.getElementById('my-title')  
console.log(myTitle)  
  
myTitle.style.color = 'red'
```

```
var myTitle = document.getElementById('my-title')  
console.log(myTitle)  
  
myTitle.style.color = 'red'  
  
<h1 id="my-title">HighlightMe</h1>  
"red"
```

```
<h1 id="my-title" style="color: red;">HighlightMe</h1>
```

HighlightMe

```
myTitle.remove()
```


jQuery

- jQuery: HTML 조작을 더 편리하게 해주는 JS 라이브러리
- 몇 년 전 이야기 (웹 표준이 지켜지지 않았을 때):
 - 브라우저 별로 지원하는 기능이 달랐음 (특히 IE 6)
 - 기능을 지원하더라도 호출하기 위한 API 명세가 달랐음
 - API가 같더라도 동작이 달랐음
 - 따라서 브라우저의 버전을 판별해 중간에서 매개하는 레이어가 필요했음 -> jQuery
- 지금은?
 - 최근에는 대부분의 웹 브라우저가 웹 표준을 준수함
 - 그러나 jQuery의 간결한 문법은 여전히 강점
 - <http://youmightnotneedjquery.com/>

jQuery 시작하기

- jQuery.js 파일을 다운로드 받고 <script> 태그로 추가한다.
- 다운받기 싫다면 <script> 태그의 src 속성에 아래 주소 입력
 - <https://code.jquery.com/jquery-3.2.1.min.js>
- `<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.2.1/jquery.min.js"></script>`

jQuery

- DOM에서 요소 선택하기

Javascript

```
document.getElementById('my-title')  
document.getElementsByTagName('p')  
document.querySelectorAll('.row')
```

jQuery

```
$('#my-title')  
$('p')  
$('.row')
```

- \$는 jQuery에서 사용하는 함수 이름
- jQuery를 사용하지 않는다면 여러분도 함수 이름을 \$로 지을 수 있습니다.

jQuery

- 스타일 바꾸기

Javascript

```
var h1 = document.querySelector('h1')  
  
h1.style.color = 'red'  
h1.style.fontFamily = 'Malgun Gothic'  
h1.style.fontSize = '2em'
```

jQuery

```
var $h1 = $('h1')  
  
$h1.css('color', 'red')  
$h1.css('font-family', 'Malgun Gothic')  
$h1.css('font-size', '2em')  
  
// or  
  
$h1.css('color', 'red')  
    .css('font-family', 'Malgun Gothic')  
    .css('font-size', '2em')  
  
// or  
  
$h1.css({  
    'color': 'red',  
    'font-family': 'Malgun Gothic',  
    'font-size': '2em'  
})
```

jQuery

- 속성 (attribute) 바꾸기

Javascript

```
var img = document.querySelector('img')  
  
img.src = 'http://myurl';  
img.width = 200
```

jQuery

```
var $img = $('img')  
  
$img.attr('src', 'http://myurl')  
    .attr('width', 200)
```

jQuery

- 기타

Javascript

```
el.style.display = 'none';

if (el.classList)
    el.classList.add(className);
else
    el.className += ' ' + className;

parent.appendChild(document.createElement('p'));
el.innerHTML = '';
el.nextElementSibling
```

jQuery

```
$(el).hide(); // 화면에서 숨기기
$(el).addClass(className); // 클래스 추가하기
$(parent).append($('<p>')); // 자식 추가하기
$(el).empty(); // 내용 비우기
$(el).next(); // 다음 형제 요소 선택하기
```

- <https://www.w3schools.com/jquery/>
- <http://youmightnotneedjquery.com/>

이벤트

- Initialize 함수를 호출해라.
- 리스트의 마지막 요소를 지워라.
- “Game Over” 를 출력해라.

이벤트

- 문서가 로드 되었을 때 (*이벤트, event*),
- Initialize 함수를 호출해라 (*핸들러, handler*).
- “삭제” 버튼이 눌렸을 때,
- 리스트의 마지막 요소를 지워라.
- 10초가 지났을 때,
- “Game Over” 를 출력해라.

이벤트

- 문서가 로드 되었을 때,
- Initialize 함수를 호출해라.
- “삭제” 버튼이 눌렸을 때,
- 리스트의 마지막 요소를 지워라.
- 10초가 지났을 때,
- “Game Over” 를 출력해라.

```
$(document).ready(function(){  
    initialize()  
})  
$(document).ready(initialize)  
$(initialize)
```

```
$('button.delete').click(function(){  
    $('#my-list li:last-child').remove()  
})
```

```
setTimeout(function(){  
    console.log('Game Over')  
}, 10 * 1000)
```

이벤트 핸들러

- 이벤트 핸들러: Javascript에서 어떤 이벤트가 발생했을 때 실행되는 코드
 - 브라우저 내부: HTML 문서가 로드되었을 때 (DOMContentLoaded)
 - 사용자: 사용자가 버튼을 클릭했을 때 (click), 사용자가 키를 눌렀을 때 (keydown)
 - 브라우저 외부: 다른 서버로 보낸 요청에 대한 응답이 돌아왔을 때
 - Event Listener
- 이벤트를 받고 싶은 DOM 요소에 핸들러를 연결

```
var myTitle = document.querySelector('h1')

myTitle.addEventListener('click', function(){
  alert('clicked!')
})

myTitle.onclick = function(){
  alert('clicked!')
}
```

alert가 한번? 두번?

이벤트 핸들러

- 자주 사용되는 이벤트

```
var $button = $('button'),
    $textInput = $('input[type=text]')

$button.click(function() { })
$button.on('click', function() { })
$button.on('dblclick', function() { })

$(window).on('keydown', function() { })
$(document).on('ready', function() { })

$textInput.on('change', function() { })
$textInput.on('focus', function() { })

$button.on('mouseover', function() { })
        .on('mouseout', function() { })
```

- https://www.tutorialspoint.com/javascript/javascript_events.htm
- (mouseover, mouseout) 과 (mouseenter, mouseleave) 의 차이는 무엇일까?

예제

- 이제까지 배운 내용으로 todo list를 만들어 봅시다.

```
<!DOCTYPE html>
<html><head></head>
  <body>
    <ol id="todos">
    </ol>

    <section>
      <input type="text" id="todo">
      <button id="add">추가</button>
    </section>

    <script src="https://ajax.googleapis.com/ajax/
    <script src="index.js"></script>
  </body>
</html>
```

```
$(function(){
  $('#add').click(function(){
    var todo = $('#todo').val()

    if(todo.length === 0) return

    $('#todos').append($('- ').text(todo))
    $('#todo').val('').focus()
  })
})

```

1. 잠자기
2. 씻기
3. 학교가기

예제

- 삭제를 지원하려면?

```
$(function(){
  $('#add').click(function(){
    var todo = $('#todo').val()

    if(todo.length === 0) return

    var $li = $('<li>').text(todo)
    $li.append(
      $('<button>')
        .text('삭제')
        .click(function() {
          $li.remove();
        })
    )

    $('#todos').append($li)

    $('#todo').val('').focus()
  })
})
```

1. 잠자기 삭제
2. 씻기 삭제
3. 학교가기 삭제
4. 먹기 삭제

1. 잠자기 삭제
2. 씻기 삭제
3. 먹기 삭제

예제

- CSS 추가하기
- <http://getbootstrap.com/>

1. 잠자기
2. 씻기
3. 학교가기

My Todo List

1. 수업듣기
2. 숙제하기
3. 수업듣기

할일	내용	추가
----	----	----

Javascript 기본 함수들

- Array
 - forEach, filter, map, reduce, sort, indexOf, ...
- String
 - toLowerCase, replace, indexOf, includes, ...
- Object
 - Object.keys(obj), Object.values(obj), ...
- Math
 - Math.random, floor, ceil, round, max, min, log, exp, sin, cos, ...

숙제 – 계산기 인터페이스 디자인 하기

- 계산기 인터페이스 디자인 하기

- 3개의 계산기 인터페이스를 디자인하고 구현한다.

- 기능

- 0-9 까지의 숫자 버튼, 사칙연산 버튼 (+, -, /, *), 계산 버튼 (=), 리셋 버튼 (C, clear)
- 계산 결과가 나오는 영역이 있어야 함 (디스플레이)
- 마우스 입력으로만 동작
- “1 0 2”: 디스플레이에 102 표시
- “1 0 2 + 4”: 디스플레이 4 표시
- “1 0 2 + 4 = ”: 디스플레이에 계산 결과 (102 + 4) 106 표시
- “1 0 2 + 4 * ”: 디스플레이에 106 표시
- “1 0 2 + 4 * 5”: 디스플레이에 5 표시
- “1 0 2 + 4 * 5 = ”: 디스플레이에 530 표시
- “1 0 2 + 4 * 5 = = ”: 디스플레이에 2650 표시

숙제 – 계산기 인터페이스 디자인 하기

- 구현의 편의를 위해
 - 음수 및 소수점 입력 기능 없음 (단, 계산 중간 결과로는 나타날 수 있음)
 - 0으로 나누는 것 없음
 - 중간 결과 모두 JS의 기본 수치형으로 처리
 - 마우스 입력만 고려
- 위의 계산기의 인터페이스를 특정 사용자 그룹을 고려하여 디자인
 - 최소 3개의 사용자 그룹을 각자 설정할 것
 - 기능 추가 가능 / 레이아웃 자유롭게 변경 가능
 - 각 사용자 그룹의 특성, 제약점, 태스크, 자주 사용하는 기능들을 고려하여 인터페이스를 디자인 할 것
 - 이 디자인은 웹 페이지에서 *전환* 가능해야 함.

숙제 – 계산기 인터페이스 디자인 하기



숙제 – 계산기 인터페이스 디자인 하기

- 마감: 2017년 4월 12일 수요일 23시 55분
- 제출할 것
 - 최소 3개의 계산기 인터페이스를 구현한 웹 페이지의 소스코드를 압축해서 제출
 - 어떤 사용자 그룹들을 설정했고 그룹의 특성이 어떻게 디자인에 반영되었는지 적기
 - html 내에 포함
 - 자신의 디자인을 정당화(justification) 할 것
 - 구현의 정확성 보다 인터페이스 디자인의 타당성이 중요
 - 최신 버전의 Chrome에서 채점
 - 모호한 동작은 <https://www.online-calculator.com/> 을 참고
 - 질문은 eTL 질문 게시판을 이용 (이메일로 보내지 말 것!)

JS에 대해서 더 알고 싶다면

- 변수를 선언할 때 var 대신 let 써보기
 - 함수를 선언할 때 function(a, b) { return a + b } 대신 (a, b) => a + b 써보기 (arrow function)
 - 함수를 리턴 하는 함수 만들어보기
 - class 사용해보기
-
- 왜 값이 같은 지 확인할 때 == 대신 ===을 쓰나요?
 - var a = function() {} 와 function a() {} 는 정말 같나요?
 - \$(function(){ initialize(); }) 와 \$(initialize) 는 정말 같나요?
 - function() {} 과 () => {} 는 정말 같나요?
 - (mouseover, mouseout) 과 (mouseenter, mouseleave) 의 차이는 무엇인가요?