

웹앱 과정 #1

웹 환경을 구성하는 기술들

2019.03

박재성

목차

1. 과정 소개

2. 웹 환경의 이해

3. express + SQLite

4. HTML

5. CSS

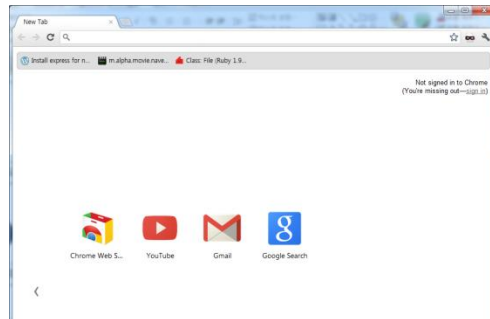
1. 과정 소개

과정 소개

-	월	화	수	목	금
강사	박재성	박재성	박재성	손찬욱	손찬욱
8:30	웹 환경에 대한 이해 - 기본개념 - 개발환경 설명 - 개발환경 설치 npm, express.js, SQLite - 기본 실습 (서버 수 행, DB 연동)	JavaScript #1 - 언어 소개 - 문법 설명 - 언어 특징	Transpiler & 프레임워크 소개 프로젝트 환경 구성 - package.json - Webpack - Babel (Transpiling)	HTML5 #1 - 개요 - 신기술 소개 - Geolocation - Video/Audio - Web Animation API	웹앱 PJT 영화 후기 관리 시스템 - DB 설계 - Ajax 활용 - HTML5 활용
9:30					
10:30					
11:30					
	점심				
1:30	HTML - 이론 - 실습	JavaScript # 2 - DOM - BOM - Event - Ajax - 실습	ES6+ - let, const - Class - Arrow function - Template - Modules - Promise (Fetch) - Aync/Await	HTML5 #2 - File API(Reader API) - Canvas - Storage	웹앱 PJT
2:30					과정 정리
3:30	CSS - 이론 - 실습			성능 분석	필기 테스트
4:30					

2. 웹 환경의 이해

기본 개념



http://www.naver.com

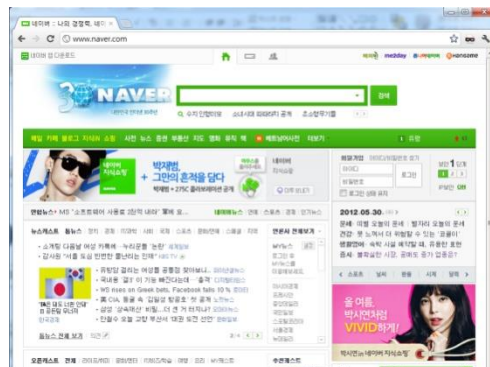
Index.html

Select * from person;

1 | 홍길동 남

server

DB



개발 환경 설명

HTML/CSS/JavaScript

SQLite
(Database)

(Node.js)
Server-side JavaScript

Express.js
(Web server)

개발 환경 설치



- V8 자바스크립트 엔진에 기반
- 이벤트 기반의 Non-blocking I/O
- 내장 HTTP 서버 라이브러리를 포함

서버사이드 자바스크립트 런타임

개발 환경 설치

<https://nodejs.org/>에 접속해서 node.js를 다운로드 한다.

→ 교육장 컴퓨터 상황에 따라 최신 버전이 올바르게 동작하지 않을 수 있어, 8.x 버전을 설치를 권장한다.

<https://nodejs.org/dist/latest-v8.x/>

Download for macOS (x64)

10.15.2 LTS

Recommended For Most Users

[Other Downloads](#) | [Changelog](#) | [API Docs](#)

11.10.1 Current

Latest Features

[Other Downloads](#) | [Changelog](#) | [API Docs](#)

서버 구성해 보기

- 아래의 코드를 “server.js”로 저장

```
var http = require("http");

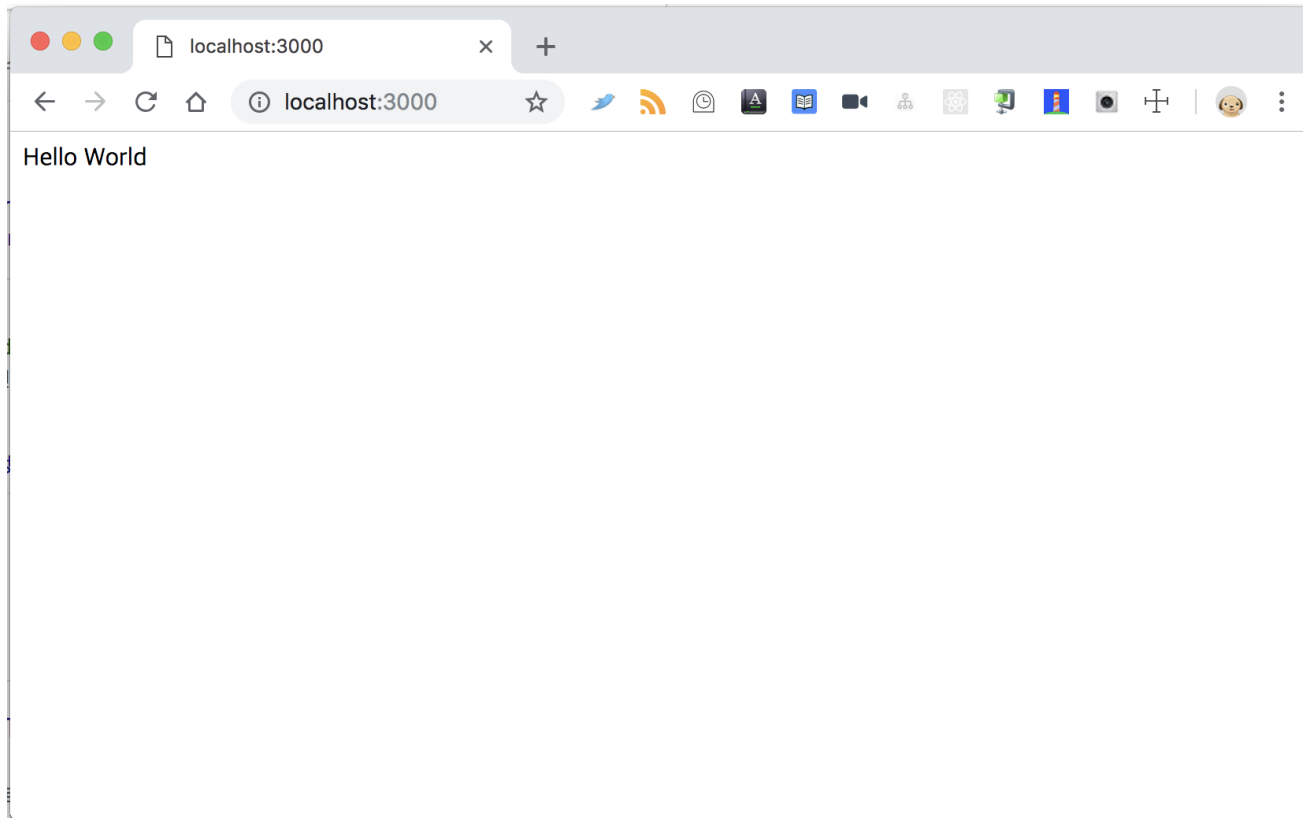
http.createServer(function (request, response) {
  response.writeHead(200, {
    "Content-Type" : "text/html"
  });

  response.write("Hello World");
  response.end();
}).listen(3000);
```

→ 파일 : server.js

서버 확인해 보기

- 명령어 창에 “node server.js”를 입력 후, 실행
- 브라우저 주소 창에 “localhost:3000” 입력 후, 확인



패키지 관리자



Node Package Manager

노드 패키지들을 npm 레지스트리로부터 설치하고 관리할 수 있는 패키지 관리자

- <https://npmjs.com>를 통해 패키지를 검색, 필요한 패키지를 쉽게 설치할 수 있다.
- 설치된 패키지는 “./node_modules” 폴더 아래에 설치된다.

```
$ npm install 패키지명 # install은 i로 축약해 사용할 수 있다.  
$ npm install 패키지명@버전 # 특정 버전 설치하는 경우  
$ npm install 패키지명1 패키지명2 ... # 동시에 여러 개의 패키지를 설치
```

```
$ npm uninstall 패키지명 # 패키지 제거  
$ npm 패키지명 -v # 설치된 패키지 버전 확인
```

npm proxy 설정

교육장 네트워크 보안 설정에 따른 proxy 설정

→ <https://docs.npmjs.com/misc/config>

```
$ npm config set proxy http://10.241.3.7:8080  
$ npm config set https-proxy http://10.241.3.7:8080  
$ npm config set strict-ssl false
```

```
# 현재 npm 설정목록 확인  
$ npm config ls -l
```

package.json 파일 #1

노드 패키지의 애플리케이션 정보를 구성하는 설정파일

→ <https://docs.npmjs.com/files/package.json>

```
$ npm init # 애플리케이션 정보 파일 생성 명령  
package name: (webapp)  
...
```

```
{  
  "name": "webapp",  
  "version": "1.0.0",  
  "description": "",  
  "main": "index.js",  
  "scripts": {  
    "test": "echo \"Error: no test specified\" && exit 1"  
  },  
  "author": "",  
  "license": "ISC",  
  "dependencies": {  
    "express": "^4.16.4",  
  },  
  "devDependencies": {  
    "sqlite3": "^4.0.2"  
  }  
}
```

package.json 파일 #2

프로젝트에 package.json 파일이 존재하는 경우,
프로젝트에 필요한 의존성 패키지들을
다음의 명령어를 통해 설치할 수 있다.

```
$ npm install
```

프로젝트에 사용되는 의존성 또는 개발 의존성 패키지 설치 명령어

```
# 패키지 설치 및 dependency 항목에 추가  
$ npm install 패키지명 --save (또는 -S)
```

```
# 패키지 설치 및 devDependency 항목에 추가  
$ npm install 패키지명 --save-dev (또는 -D)
```

Module: 다양한 모듈 명세



- 모듈은 기능 단위(또는 파일)로 분리해 필요한 기능들을 가져와 사용할 수 있는 방법을 제공한다.
- 표준이 존재하지 않아, 2009년 AMD와 CommonJS 명세가 제안되고 사용되었으며, 이후 ESM이 ES6에 포함되었다.



이름	실행 유형	실행 구문	실행 방법	모듈 단위
RequireJS (AMD 기반)	비동기식	define require	콜백 호출	모듈 정의 영역
CommonJS	동기식	module.export require	객체 반환	파일
ES6 Module	동기식/비동기식	export import	객체 반환	모듈 정의 영역

Module: CommonJS/RequireJS

CommonJS 형식: 파일 단위 모듈

```
// export (A.js)  
module.export = { ... }
```

```
// import (B.js)  
require("파일명");
```



RequireJS (AMD) 형식: 정의 단위 모듈

```
// export  
define(["sample"], function () {  
    ...  
    return module;  
});  
  
// import  
require(["jQuery", "app/sample"], function() {  
    // callback  
});
```



실습 파일 다운로드

→ 다운로드 주소는 수업에서 공지

<https://goo.gl/TWVxai>

3. Express.js + SQLite

express

<https://expressjs.com>

Node.js를 위한 빠르고 개방적인 간결한
웹 프레임워크

설치 명령어

```
# express.js 설치  
$ npm install express --save-dev
```

요청에 대한 이해: GET/POST

GET은 다음과 같이, 주소창에 '?'뒤에 key=value 형태로 파라미터가 전달되는 경우

- 브라우저마다 처리 가능한 최대 크기가 다르며, 100kb ~ 200kb 내외
- 정보가 주소 창에 노출되는 형태이므로, 보안 민감한 정보는 사용하지는 않는다.

GET 방식의 요청 (Request)

http://www.naver.com?key=value&key2=value2

POST는 HTML <form>을 통해 값이 서버에 전달 되는 경우

아이디

@naver.com

비밀번호



비밀번호 재확인



이름

- 최대 크기 제한이 없으므로, 대용량 데이터 전송 가능
- 데이터가 노출되지 않으므로, 보안 정보 전송 가능

서버 실행해 보기

- 아래의 코드를 “express.js”로 저장

```
const express = require("express"); // express 모듈을 import
const app = express(); // express 초기화
const port = 3000; // 서버 포트번호

app.get("/", function(req, res) {
  res.send("Hello World!");
});

app.listen(port, function() {
  console.log("Example app listening on port " + port + "!");
});
```

- 명령어 창에 “**node express.js**”를 통해 실행
- 브라우저 주소 창에 “localhost:3000” 입력 후, 확인

Routing

요청 방법과 경로에 따라 수행될 콜백 함수를 지정한다.

```
app.METHOD(path, callback [, callback ...]);
```

```
app.get("/", function(req, res) {  
    res.send('GET request to the homepage')  
});  
  
app.post("/", function(req, res) { ... }  
  
// get 또는 post 상관없이 처리  
app.all("/secret", function(req, res, next) { ... }
```

Routing

다양한 값을 통해 매칭 path를 설정할 수 있다.

```
app.get("/about", function(req, res) { ... });

// 또는 정규식을 사용
app.get("/ab?cd", ... )    // acd 또는 abcd 매칭
app.get("/ab+cd", ... )    // abcd, abbcd, abbbcd, ...
app.get("/ab*cd", ... )    // abcd, abxcd, abRANDOMcd, ab123cd, ...
app.get("/ab(cd)?e", ... ) // /abe와 /abcde를 매칭
app.get(/.*fly$/, ... )    // fly로 끝나는 모든 요청과 매칭 (ex. dragonfly)

// Rout parameters
// /users/hong/books/abcd 와 같이 호출시,
// { "usres": "hong", "books": "abcd" }로 값을 액세스 할수 있다.
app.get("/users/:userId/books/:bookId", ...);
```

→ 파일 : routing.js

nodemon



node를 래핑한 패키지로, 파일 변경이 발생하면 자동으로 node 인스턴스를 재실행 하도록 한다.

```
$ npm install -g nodemon
```

```
# 기존 실행  
$ node index.js
```

```
# nodemon으로 실행. index.js가 변경되면 자동으로 재실행 된다.  
$ nodemon index.js
```

```
[nodemon] 1.18.4  
[nodemon] to restart at any time, enter `rs`  
[nodemon] watching: *.*  
[nodemon] starting `node index.js`  
Example app listening on port 3000!
```

request, response

라우팅 콜백 함수에는 request와 response 파라미터가 전달된다.

- request: HTTP 요청에 대한 속성들을 포함
→ API 문서: <http://expressjs.com/en/4x/api.html#req>
- response: HTTP 응답에 대한 속성들을 포함
→ API 문서: <http://expressjs.com/en/4x/api.html#res>

```
app.get("/about", function(request, response) {  
  request.params; // rout parameters 형태로 요청된 값  
  request.body;   // key-value 쌍으로 이뤄진 요청된 값  
  request.query;  // ?param=val& ... 형태의 쿼리로 요청된 값  
  
  // HTTP 응답 처리  
  response.send({ some: "json" });  
  response.send('<p>some html</p>');  
});
```

Middleware

각 요청 작업 수행 시, 미들웨어를 통해 추가적인 작업이 수행되게 할수 있다.
(ex. 로깅)

```
var express = require('express');
var app = express();

app.get('/', function(req, res, next) {
  next();
})

app.listen(3000);
```

미들웨어 함수가 적용되는 HTTP 메소드.

미들웨어 함수가 적용되는 경로(라우트).

미들웨어 함수.

미들웨어 함수에 대한 콜백 인수(일반적으로 "next"라 불림).

미들웨어 함수에 대한 HTTP 응답 인수(일반적으로 "res"라 불림).

미들웨어 함수에 대한 HTTP 요청 인수(일반적으로 "req"라 불림).

→ <https://expressjs.com/ko/guide/using-middleware.html>

Middleware

간단한 time 로깅 미들웨어 사용 예

```
var express = require("express");
var app = express();

// 요청이 발생할 때 마다 현재 time 정보를 전달
var requestTime = function (req, res, next) {
  req.requestTime = Date.now();
  next(); // 앱 내의 다음 미들웨어 함수 호출
};

app.use(requestTime);

app.get("/", function (req, res) {
  var responseText = "Hello World!";

  // 요청이 처리될 때 마다 미들웨어에서 전달된 time 정보를 반환한다.
  responseText += "Requested at: " + req.requestTime;
  res.send(responseText);
});

app.listen(3000);
```

Middleware: body-parser

body 파싱 미들웨어로 <form>을 통한 데이터 전송의 값들을 편리한 형태로 파싱해 사용할 수 있다.

```
$ npm install body-parser --save-dev
```

```
var bodyParser = require("body-parser");

// URL-encoded와 JSON 형태 파싱 지원 설정
// url 인코딩 라이브러리 사용설정 - true: qs / false: querystring (true는 deprecate 처리됨)
// → qs는 querystring에서 nested 객체 표현이 가능 (ex. a[b]=1 → '{a: { b: 1}}'로 표현됨)
app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json()); // for parsing application/json

app.post("/post", function(req, res) {
  console.log(req.body); // <form>을 통해 전달된 값
  res.send("okay");
});
```

디렉토리 리스팅: serve-index

특정 디렉토리에 있는 파일/디렉토리 목록을 볼수 있게 한다.

다음의 명령어를 통해 설치:

```
$ npm install serve-index --save-dev
```

```
var express = require("express");  
var serveIndex = require("serve-index");  
var app = express();
```

// 모든 디렉토리 목록이 보이도록 설정한다.

```
app.use("/", express.static(__dirname + "/"), serveIndex(__dirname + "/", {  
  icons: true  
}));
```

server-index 선언 위치

server-index 미들웨어는 라우팅에 영향을 주기 때문에, 제일 마지막에 사용하도록 선언되어야 한다.

```
app.get("/user", ...);  
app.post("/create", ...);
```

```
// 다른 라우팅이 처리된 후, 제일 마지막에 선언
```

```
app.use("/", express.static(__dirname + "/"), serveIndex(__dirname + "/", {  
  icons: true  
}));
```

```
app.listen(port, ... );
```

Template

- 템플릿은 구성하고자 하는 문자열 패턴에 데이터를 바인딩 하는 것만으로 손쉽게 원하는 값을 얻을 때 사용할 수 있다.
- 사용할 템플릿 엔진의 별도 설치가 필요하다.

```
# 템플릿 엔진은 여러 가지가 있으나, 가장 유명한 pug(구 jade)를 설치한다.  
$ npm install pug --save-dev
```

템플릿

데이터

생성 결과

```
html  
head  
  title= title  
body  
  h1= message
```

+

```
{  
  title: "Hey",  
  message: "Hello there!"  
}
```



```
<html>  
  <head><title>Hey</title></head>  
  <body>  
    <h1>Hello there!</h1>  
  </body>  
</html>
```


Template

템플릿을 사용하기 위해, 다음의 순서에 따라 설정한다.

1) views 폴더에 '**파일명**.pug'라는 이름으로 템플릿을 설정하고 저장한다.

```
html
  head
    title= title
  body
    h1= message
```

2) 템플릿 엔진의 사용을 설정한다.

```
app.set("view engine", "pug")
```

3) 처리할 routing에 response 객체의 res.render()에 사용할 템플릿의 파일명과 바인딩할 데이터를 설정한다.

```
app.get("/", function (req, res) {
  res.render("파일명", { title: "Hey", message: "Hello there!" });
});
```

→ 파일: template.js



<https://www.sqlite.org>

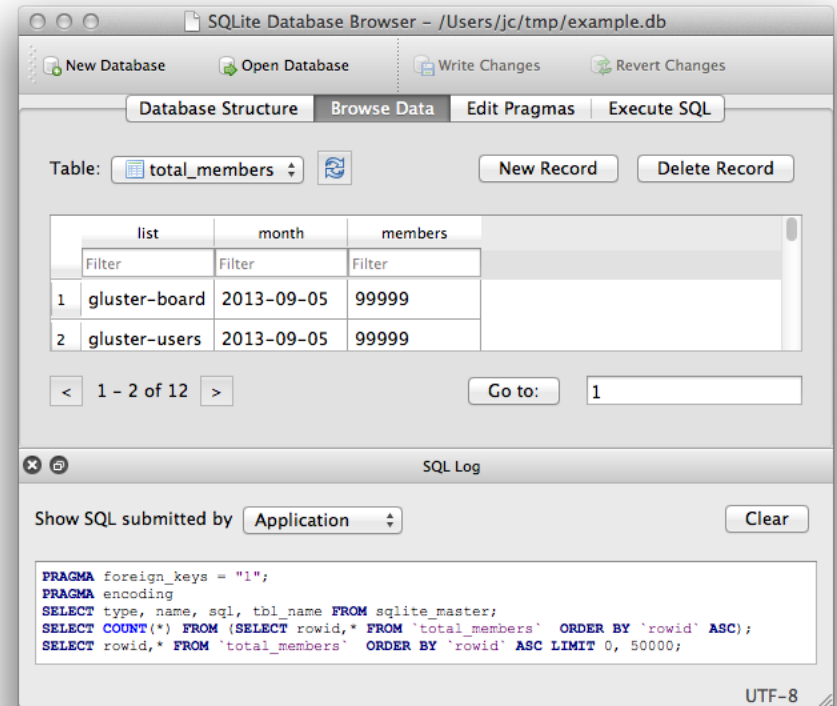
응용 프로그램에 넣어 사용하는 비교적
가벼운 데이터베이스

설치 명령어

```
# Install SQLite3 bindings for Node.js  
$ npm install sqlite3 --save
```

DB Browser for SQLite

GUI SQLite
데이터베이스 관리자



<https://sqlitebrowser.org/> 에서 자신의 운영체제에 맞는 버전을 다운로드

SQL(Structured Query Language)?

- ‘구조화 질의어’는
관계형 데이터 베이스(RDMS) 자료
검색, 관리 데이터 베이스 생성, 수정 등을
위해 사용된다.

UPDATE clause [UPDATE country
SET clause [SET population = population + 1
WHERE clause [WHERE name = 'USA';

Expression
Statement
Expression
Predicate

CRUD 문법 설명

1. Create

insert into *TABLE* (column,column2) values (value, value2);

2. Read

select column, column2 from *TABLE*;

select * from *TABLE* where column = "pk";

select * from *TABLE* where column like "1 %";

3. Update

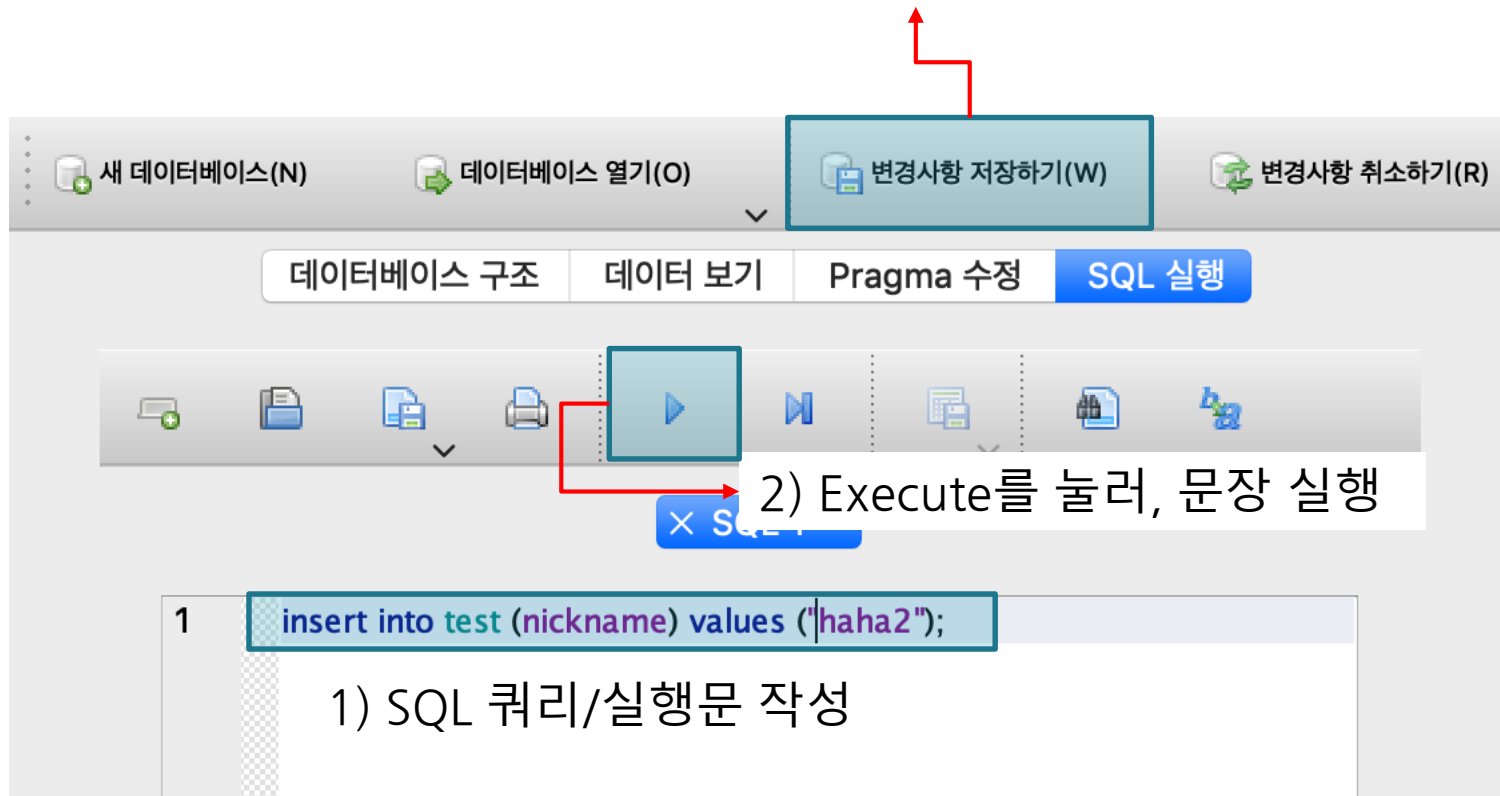
update *TABLE* set column2 = "test" where column = "pk";

4. Delete

delete *TABLE* from where column = "pk";

SQL 실행문 수행 과정

3) 모든 작업이 종료되면, '변경사항 저장'을 통해 DB에 반영



실습 시작#1

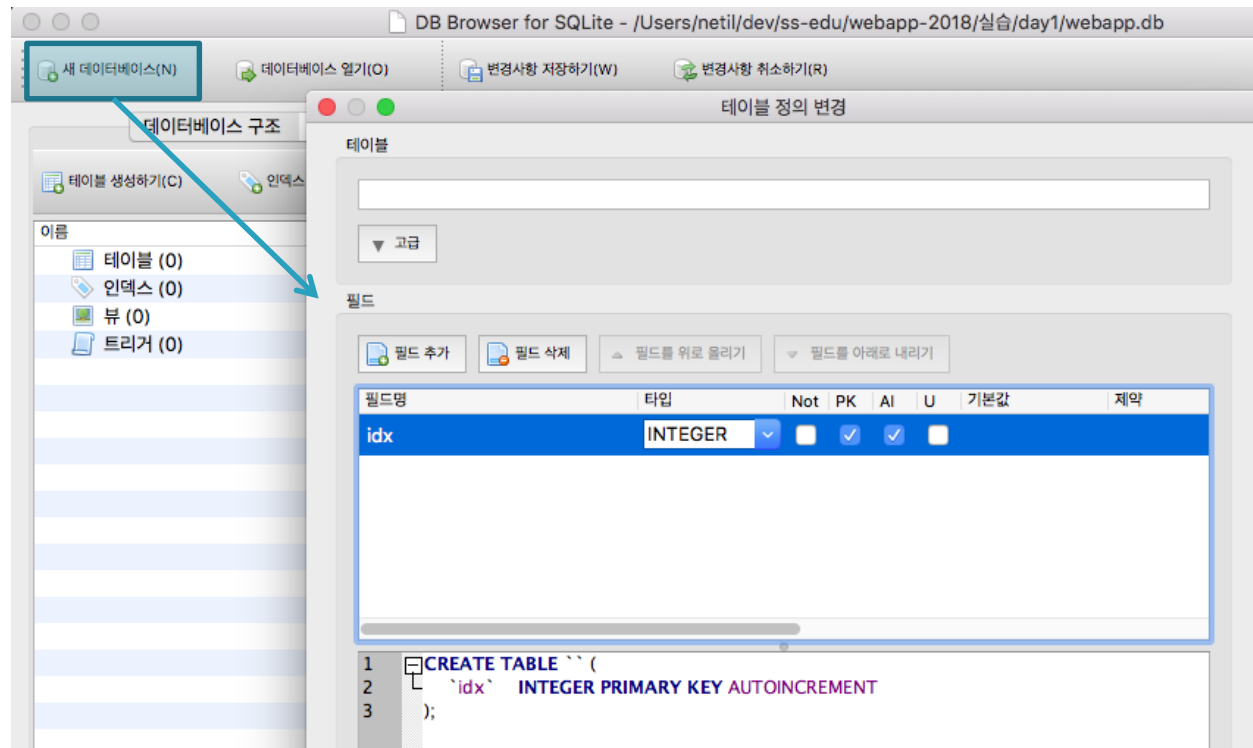
- WebApps DB생성
- TODO Table생성

column :

- idx : pk이고 int이며 자동증가
- todo : text.
- complete : 는 int(기본값 0)

실습 시작#1

```
CREATE TABLE todo (  
  idx      INTEGER PRIMARY KEY AUTOINCREMENT,  
  todo     TEXT,  
  complete INTEGER DEFAULT 0  
);
```



실습 시작#2

직접 SQL 문장을 작성해
todo 레코드 추가하기

실습 시작#3

직접 SQL을 사용해
todo 데이터 가져오기

Connection

```
var sqlite3 = require("sqlite3").verbose();
var db = new sqlite3.Database("./WebApps.db"); // 사용될 DB 파일을 선택

// DB에 대한 작업이 순차적으로 진행되게 한다.
db.serialize(function () {
  db.run("CREATE TABLE foo (num)");
  db.run("INSERT INTO todo (todo) VALUES (?)", "웹앱 교육", function() {
    // insert 작업 후, 실행할 코드
  });

  db.each("SELECT * FROM table-name", function (err, row) {
    console.log(row);
  });
});

db.close(); // 모든 작업 후, DB를 닫는다.
```

실습 시작#4

간단한 db 추가, 내용 출력을 해봅니다.

DB

- Table : test
- Column :
 - id int 자동증가
 - nickname text

→ 파일 : quiz/db.js

4. HTML

HTML

- 문서의 구조를 만드는 언어
- Markup language
- UI 개발의 중요한 부분
- 글쓰기 할 때 글의 구조를 잡는 것이 중요하듯
html 구조를 잘 만드는 것이 중요

Semantic markup

의미에 맞는 태그의 사용



좋은 문서의 구조

DTD(Document Type Definition)

문서 형식 정의는 문서들을 일정한 규칙을 정하여 통합하고, 다양한 문서간의 표준을 제시하기 위해 사용된다.

HTML5

```
<!DOCTYPE html>
```


HTML tag

tag는 다음의 형식으로 이루어져 있다.

```
<tag-name>  
  <tag-a>  
    <tag-b>  
      text value  
    </tag-b>  
  </tag-a>  
</tag-name>
```

- 일반적으로 시작 tag와, 닫는 tag가 쌍으로 이루어 진다.
- tag내에 또 다른 tag 또는 텍스트 등이 포함될 수 있다.
- Tree 구조를 갖는다.

HTML tag 속성

tag에 따라 사용되는 속성이 다르다.

```
<a href=http://www.naver.com/ target="_new" title="바로가기">  
네이버
```

```
</a>
```

```
<div style="width:500px;height:200px;"> ... </div>
```

head

문서의 각종 정보 및 리소스를 정의

```
<head>  
  <title> ... </title>  
  <script> ... </script>  
  <link rel="stylesheet" type="text/css" ... />  
</head>
```

meta

```
<meta http-equiv="Content-Type"  
      content="text/html; charset=utf-8">
```

→

```
<meta charset="utf-8">
```

script/link

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

```
// inline
```

```
</script>
```

```
<script type="text/javascript" src="test.js" ></script>
```

```
<style type="text/css">
```

```
// inline
```

```
</style>
```

```
<link rel="stylesheet" type="text/css" href="test.css" />
```

제목

h1 ~ h6

- 문서의 제목을 나타내는 markup
- h1 ~6 까지 나타냄
- heading

[네이버를 시작페이지로 >](#) | [주니어네이버](#) [해피빈](#)


[메일](#) [카페](#) [블로그](#) [지식iN](#) [쇼핑](#) [Pay](#) [TV](#) [사전](#) [뉴스](#) [증권](#) [부동산](#) [지도](#) [영화](#) [뮤직](#) [책](#) [웹툰](#)
[더보기](#)
6 [소희](#)

서울대 출신 7인이 개발한 “뇌새김”
“영어가 조금씩 들리니까 신기해요”
지금 신청하면 평생무료 11시까지!

본 혜택은 이후에도 동일하게 적용될 수 있음 / 캔탈 후 평생무료

[연합뉴스](#) > "내일 맑지만 쌀쌀해요"...동해안에는 새벽까지 강한 비

[네이버뉴스](#) [연예](#) [스포츠](#) [경제](#)
[뉴스스탠드](#) > [전체 언론사](#) | [MY 뉴스](#)

[조선일보](#)
[The Korea Herald](#)
[Net Korea](#)
[에럴드경제](#)
[OSEN](#)
[일간스포츠](#)
[중앙일보](#)
[SBS](#)
[한겨레](#)
[KJD](#)
[데일리안](#)
[프레시안](#)
[농민신문](#)
[metro](#)
[SPOTV NEWS](#)
[kbc 광주방송](#)
[경인일보](#)
[기호일보](#)

네이버를 더 안전하고 편리하게 이용하세요.

[NAVER 로그인](#)
[아이디·비밀번호 찾기](#)
[회원가입](#)
10.18. (목) [스포츠](#)

3/6


[농구](#) 전자랜드, 높이의 KCC 꺾으며 3연승으로 단독 선두

[배구](#) '요스바니 38득점' OK저축은행, 우리카드 꺾고 개막 2...

[야구](#) “미안해요” 반복한 한화 직원들...대전구장은 11년 만...


세상에 없던 프리미엄 라이프

속초 테르바움

남북 연결의 중심점, 속초의 가치.

총 199세대 74·84·106·112㎡

선착순 분양중

문의전화 **1644-1127**
[국](#)
[비즈니스](#)
[FARM](#)
[스쿨잼](#)
[공연전시](#)
[법률](#)
[동물공감](#)
[연애·결혼](#)
[테크](#)
[감성충전](#)


[쇼핑](#)
[상품](#)
[쇼핑몰](#)
[MEN](#)


그룹핑

div, span

- 엘리먼트를 그룹핑 하는 엘리먼트
- div (division) - 블록 엘리먼트
- span - 인라인 엘리먼트


네이버를 시작페이지로 > | [중국어네이버](#) [해피빈](#)





[메일](#) [카페](#) [블로그](#) [지식IN](#) [쇼핑](#) [Pay](#) [TV](#) [사전](#) [뉴스](#) [증권](#) [부동산](#) [지도](#) [영화](#) [뮤직](#) [책](#) [웹툰](#) | [더보기](#)

6 [소희](#)



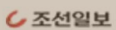


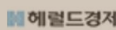








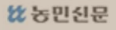





서울대 출신 7인이 개발한 “뇌새김”
“영어가 조금씩 들리니까 신기해요”
지금 신청하면 평생무료 11시까지만!

본 혜택은 이후에도 동일하게 적용될 수 있음 / 캔탈 후 평생무료

[연합뉴스](#) > "내일 말지만 쌀쌀해요"...동해안에는 새벽까지 강한 비

[네이버뉴스](#) [연예](#) [스포츠](#) [경제](#)


[뉴스스탠드](#) > [전체 언론사](#) | [MY 뉴스](#)

 조선일보	 The Korea Herald	 Net Korea	 헤럴드경제	 OSEN	 일간스포츠
 중앙일보	 SBS	 한겨레	 KJD	 데일리안	 프레시안
 농민신문	 metro	 SPOTVNEWS	 kbc 광주방송	 경인일보	 기호일보

네이버를 더 안전하고 편리하게 이용하세요.
[NAVER 로그인](#)
아이디·비밀번호 찾기 [회원가입](#)

10.18. (목) | [스포츠](#) **3/6** < >

농구 전자랜드, 높이의 KCC 꺾으며 3연승으로 단독 선두
배구 '오스바니 38득점' OK저축은행, 우리카드 꺾고 개막 2...
야구 "미안해요" 반복한 한화 직원들...대전구장은 11년 만...



세상에 없던 프리미엄 라이프
속초 테르바움
남북 연결의 중심점, 속초의 가치.
총 199세대 74·84·106·112㎡
선착순 분양중 문의전화 **1644-1127**

국

비즈니스

FARM

스쿨잼

공연전시

법률

동물공감

연애·결혼

테크

감성충전

< >

쇼핑

상품

쇼핑몰

MEN

목록

dl, ul, ol

- 목록을 나타내는 엘리먼트
- dl (정의 목록) definition
- ul (비순차 목록) unordered
- ol (순차 목록) ordered

네이버를 시작페이지로 > | [주니어네이버](#) [해피빈](#)


[메일](#) [카페](#) [블로그](#) [지식iN](#) [쇼핑](#) [Pay](#) [▶TV](#) [사전](#) [뉴스](#) [증권](#) [부동산](#) [지도](#) [영화](#) [뮤직](#) [책](#) [웹툰](#)

더보기 ▾

서울대 출신 7인이 개발한 “뇌새김”
“영어가 조금씩 들리니까 신기해요”
지금 신청하면 평생무료 11시까지만!

본 혜택은 이후에도 동일하게 적용될 수 있음 / 캔탈 후 평생무료

[연합뉴스](#) > "내일 맑지만 쌀쌀해요"...동해안에는 새벽까지 강한 비

[네이버뉴스](#) [연예](#) [스포츠](#) [경제](#)
[뉴스스탠드](#) > [전체 언론사](#) | [MY 뉴스](#)

종합/경제 1/63

방송/통신

IT

영자지

스포츠/연예

매거진/전문지

지역

[BUSINESSwatch](#) 10.18.16:19 편집 [+ 구독](#)


[인사이드 스토리]세법을 모르는 세무공무원
 [일자리, 기업이 답이다]④새 술은 새 부대에
 원리금이 소득 70% 넘으면 돈 더 못 빌린다
 [재계3·4세 시즌2]⑩동화약품, 낡고 취약한 지배구조
 美 환율보고서 '무사' 통과...증시 다음 변수는?
 [인사이드 스토리]한국GM, 법인분리 논란의 전말

급상승 검색어

DataLab. 급상승 트래킹 >

1~10위

11~20위

- 1 터키 알파고 직업
- 2 알파고
- 3 신이
- 4 고창환
- 5 써커펀치
- 6 소희
- 7 강서구 pc방 살인
- 8 불꽃페미엑션
- 9 우수아이아
- 10 김창환

2018.10.18. 22:28:00 기준 (?)



세상에 없던 프리미엄 라이프
속초 테르바움
 남북 연결의 중심점, 속초의 가치.
 총 199세대 74·84·106·112㎡

선착순 분양중

문의전화 **1644-1127**

강조

em, strong

- em (emphasized) - 강조
- strong - 중요

form

- form

→ 데이터를 전송할 때 사용.

- input

→ type(button, text, radio...)

→ label과 같이 사용

- select

→ option

단락/개행

p, br

- p (paragraph) - 단락을 나타낼 때
- br (line break) - 줄바꿈 할 때

링크/이미지

a, img

- a (anchor)

→ 이동할 때

- img

→ 이미지 사용할 때

table

table

- 데이터를 표현할 때 사용.
 - caption : 테이블 제목
 - colgroup : 컬럼을 그룹핑할 때 사용.
 - col : 컬럼
 - thead : 머릿글
 - th : 제목
 - tfoot : 바닥글
 - tbody : 본문
 - tr : 열
 - td : 행

구분	신자산번호	제조사	기기번호	모델명	TYPE	소유유형	지역
PC	A12334654	DELL	3QZG615	GX620	DESKTOP	공용	젤존
PC	A12334654	삼성전자	3QZG615	GX620	NOTEBOOK	공용	벤처
PC	A12334654	DELL	3QZG615	GX620	DESKTOP	공용	젤존
PC	A12334654	삼성전자	3QZG615	GX620	NOTEBOOK	공용	벤처
PC	A12334654	DELL	3QZG615	GX620	DESKTOP	공용	젤존
PC	A12334654	DELL	3QZG615	GX620	DESKTOP	공용	젤존
PC	A12334654	삼성전자	3QZG615	GX620	NOTEBOOK	공용	벤처
PC	A12334654	DELL	3QZG615	GX620	DESKTOP	공용	젤존
PC	A12334654	삼성전자	3QZG615	GX620	NOTEBOOK	공용	벤처
PC	A12334654	DELL	3QZG615	GX620	DESKTOP	공용	젤존

```
<table cellspacing="0" border="1" summary="유형별 자산목록리스트" class="tbl_type">
  <caption>유형별 자산목록</caption>
  <colgroup>
    <col width="12%"><col width="12%" span="6">
  </colgroup>
  <thead>
    <tr>
      <th scope="col">구분</th>
      <th scope="col">신자산번호</th>
    ...
  </tr>
</thead>
<tbody>
  <tr>
    <td>PC</td>
    <td>A12334654</td>
    ...
  </tr>
</tbody>
</table>
```

간단한 마크업 실습

div

TODO

할 일을 입력하세요.

추가

- 완료 ☐ 청소 하기.
- 완료 ☐ 빨래 하기.
- 완료 ☐ 숙제하기
- 완료 ☐ 문서 작성하기
- 완료 ☐ 병원가기
- 완료 ☐ 세금내기

h1

form

- textarea
- input[type=submit]

ul

- li
 - label
 - input

→ 파일 : html/todo.html

5. CSS

CSS

- 프리젠테이션 스타일을 마크업에 적용
- Presentation language
- Cascading Style Sheet

```
[css selector] {  
    style : value;  
    style2 : value2;  
}
```

```
#id {  
    border : 1px solid red;  
}
```

CSS Selector

- #id : 아이디
- .class : 클래스
- input : 태그
- input[type=text] : 속성
- , : or
- E F : 자손 노드
- E > F : 자식 노드

→ 파일 : css/01_selector.html

스타일 적용방법

- External

```
<head>  
  <link rel="stylesheet" type="text/css" href="some.css"/>  
</head>
```

- Internal

```
<head>  
  <style type="text/css">  
    ...  
  </style>  
</head>
```

- Inline

```
<span style="color:red;"> ... </span>
```

Cascading

우선 순위에 따라 스타일 결정

상속#1

- 부모의 속성이 자식에게 전달

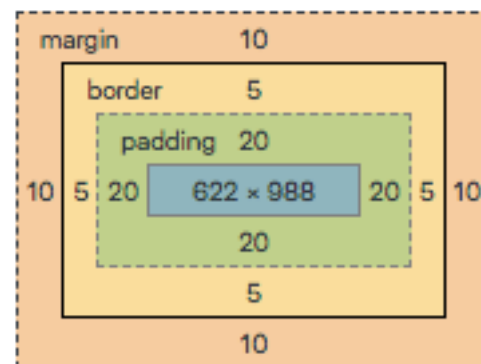
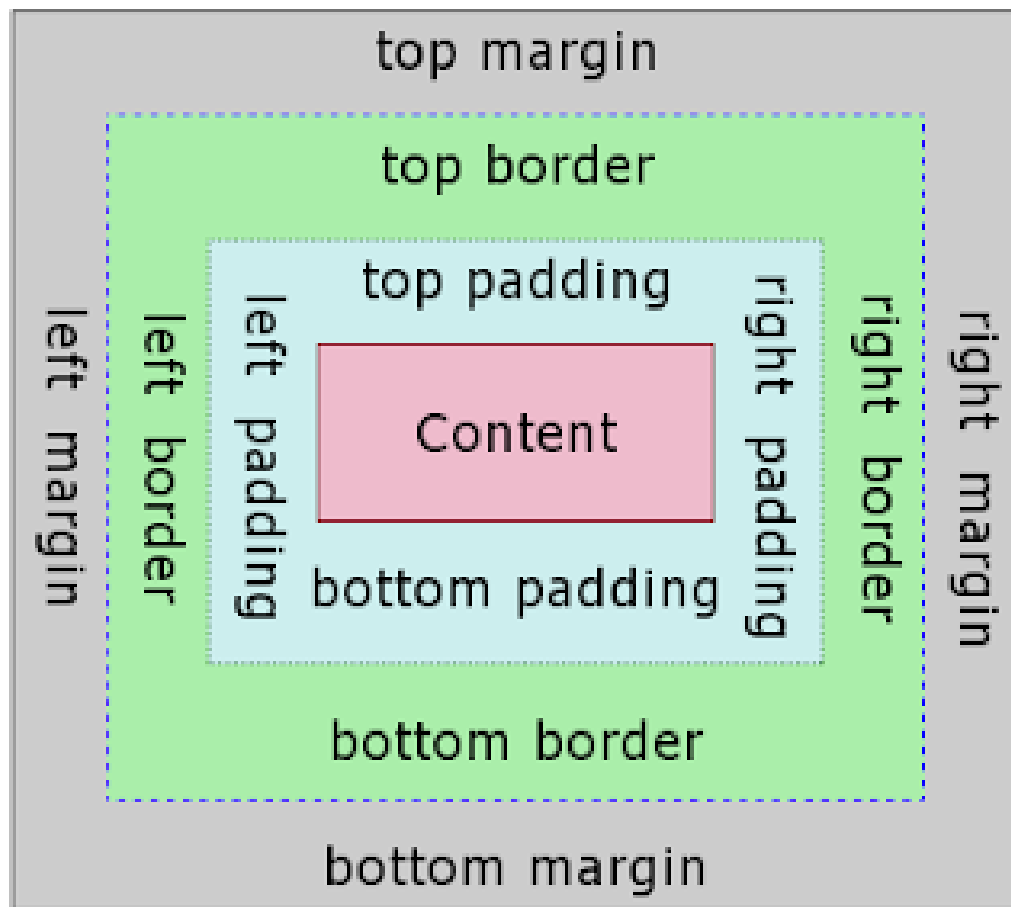
```
<style type="text/css">  
  div{color:red}  
</style>
```

```
<div>CSS  
  <span>상속</span>  
</div>
```

상속#2

- 유의점
 - 박스모델의 속성은 상속되지 않는다.
 - border, padding, margin

박스 모델



세분화#1

우선순위를 결정

Tag Selector

- tag : 1
- class : 10
- id : 100

세분화#2

<style>

```
#test{  
  background-color : red;  
}  
.show {  
  background-color : blue;  
}  
div{  
  background-color : yellow;  
}
```

</style>

<div id="test" class="show"></div> <!-- color? -->

Cascade

같은 우선순위일 경우 마지막

```
<style type="text/css">  
  div { color:red; }  
  div { color:blue; }  
</style>
```

```
<div>CSS  
  <span>상속</span>  
</div>
```

Block Element

body : 페이지 본문
div : 블록레벨 그룹핑 엘리먼트
h1~ h6 : 머리글
p : 단락
ul : 순서가 없는 목록
ol : 순서가 있는 목록
dl : 정의목록
dt : 정의용어
dd : 설명정의
li : 리스트 아이템
blockquote : 블록 레벨 인용문 사용
hr : 가로줄
form : 입력 폼
pre : 미리 서식을 지정한 텍스트
fieldset : 필드셋 라벨
col : 테이블 컬럼

colgroup : 테이블 컬럼 그룹
embed : 외부 콘텐츠
object : 개체 임베딩
caption : 테이블 설명
table : 테이블
tbody : 테이블 바디
textarea : 텍스트 입력 폼
tfoot : 테이블 푸터
th : 테이블 헤더
thead : 테이블 헤더
tr : 테이블 행

Inline Element

a : 앵커태그
abbr : 줄임말
address : 물리적인 주소
br : 문단 개행
cite : 짧은 인용
em : 강조(이탤릭체)
font : 폰트 외양
i : 이탤릭체
iframe : 인라인 하위 윈도우
img : 이미지 임베딩

input : 입력필드
label : 폼 엘리먼트 라벨
legend : 필드셋 제목
q : 한 문장내 짧은 인용문
span : 인라인 그룹핑 엘리먼트
select : 선택가능목록
strong : 굵은 강조
summary : 테이블 요약정보
td : 테이블 데이터

표시

- display

- none, block, inline, inline-block
- display가 none일 때 **자리를 차지하고 있지 않음**
- 중요 콘텐츠 일 때 위치를 안 보이는 곳으로 이동

- visibility

- visible, hidden
- visibility가 hidden일 때 **자리를 차지하고 있음**
- 스크린 리더가 읽음

표시

```
<style type="text/css">
  #display span{display:none}
  #visibility span{visibility:hidden}
</style>
```

```
<div>
  <span>CSS</span>
  <em>Normal</em>
</div>
```

```
<div id="display">
  <span>CSS</span>
  <em>Display</em>
</div>
```

```
<div id="visibility">
  <span>CSS</span>
  <em>visibility</em>
</div>
```

CSS Normal
Display
visibility

→ 파일 : css/02_display.html

넘침

- overflow

- 스크롤을 보이게 하거나 숨기고 영역을 벗어날 때 내용을 숨김
- visible , hidden , scroll, auto

넘침

```
<style type="text/css">  
  #hidden{width:100px; overflow:hidden;}  
  #scroll{width:100px; overflow:scroll;}  
</style>
```

```
<div>  
asdfasdfsadfasdfasfsdfsafasfsafasdfasf  
</div>
```

```
<div id="hidden">  
asdfasdfsadfasdfasfsdfsafasfsafasdfasf  
</div>
```

```
<div id="scroll">  
asdfasdfsadfasdfasfsdfsafasfsafasdfasf  
</div>
```

asdfasdfsadfasdfasfsdfsafasfsafasdfasf
asdfasdfsadfa
asdfasdfsaf

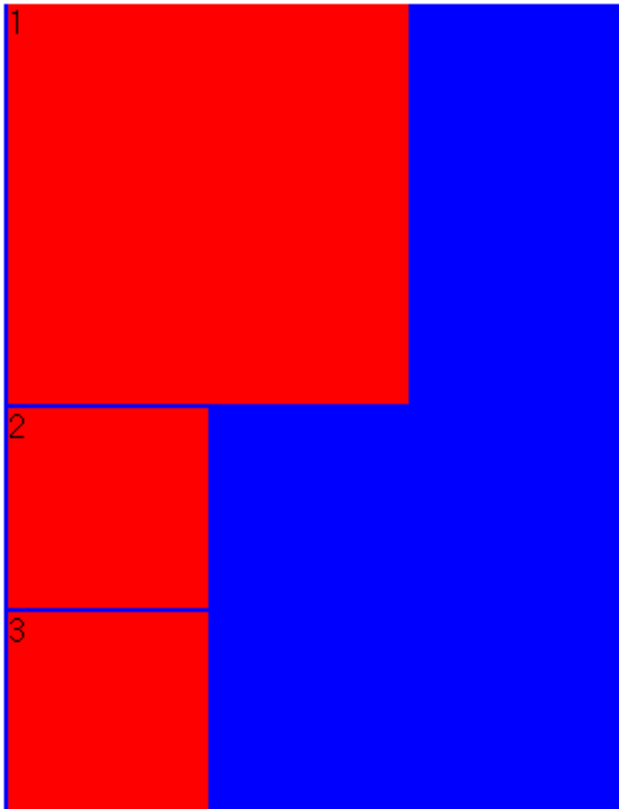
→ 파일 : css/03_overflow.html

흐름

문서의 흐름과 관계없이
화면의 배치를 함

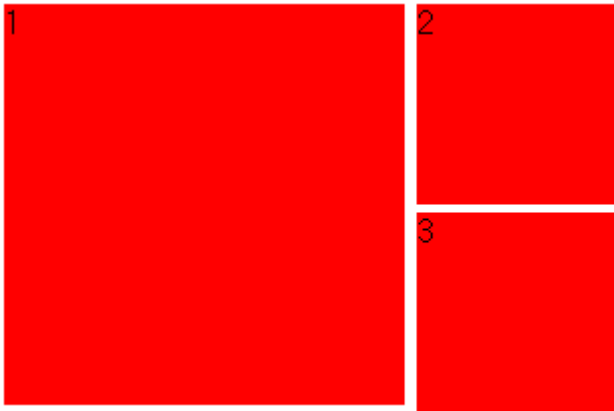
- **float**
 - float:left - 좌측으로 float
 - float:right - 우측으로 float
 - float:none - float 속성 초기화
- **clear**
 - clear : both - float을 해제

하임



```
<div id="wrap">  
  <div id="first">1</div>  
  <div id="second">2</div>  
  <div id="third">3</div>  
</div>
```

하

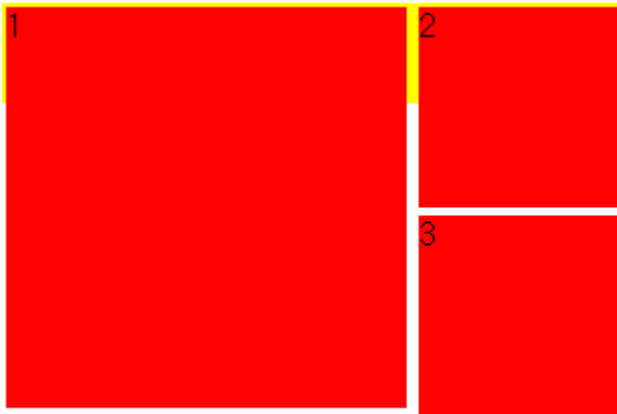


```
#first{  
    float:left;  
}
```

```
#second {  
    float:right;  
}
```

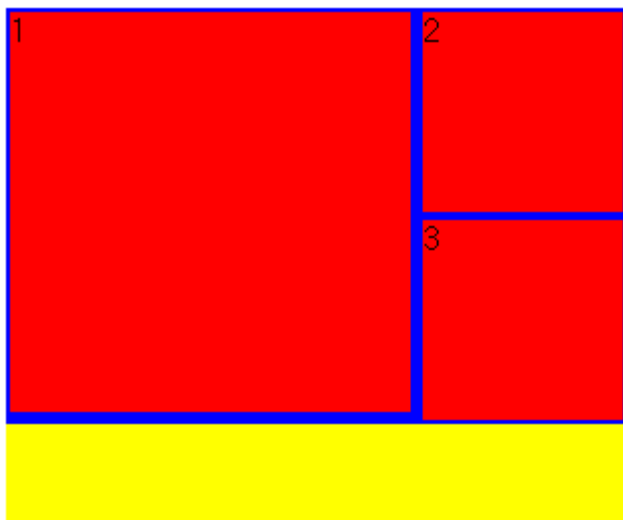
```
#third {  
    float:right;  
}
```

하임



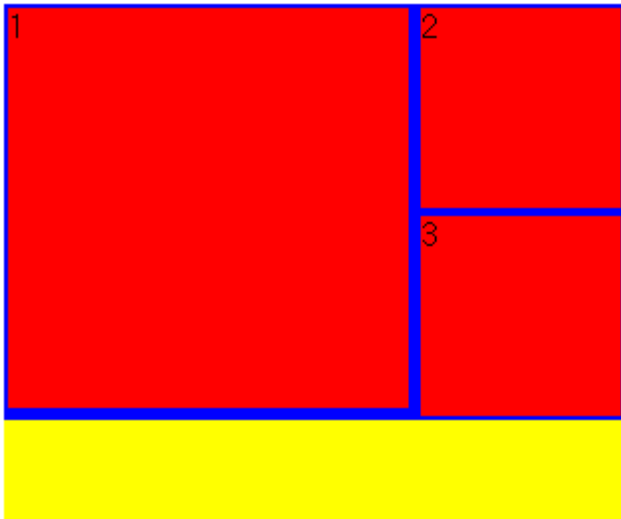
```
#first{  
    float:left;  
}  
  
#second{  
    float:right;  
}  
  
#third{  
    float:right;  
}  
  
#footer{  
    width : 310px;  
    height : 50px;  
    background-color:yellow;  
}
```


레이아웃



```
<div id="wrap">  
  <div id="first">1</div>  
  <div id="second">2</div>  
  <div id="third">3</div>  
  <div id="clear"></div>  
</div>  
<div id="footer"></div>
```

하임



→ 파일 : css/04_float.html

```
#first{  
    float:left;  
}  
  
#second{  
    float:right;  
}  
  
#third{  
    float:right;  
}  
  
#footer{  
    width : 310px;  
    height : 50px;  
    background-color:yellow;  
}  
  
#clear{  
    clear:both;  
}
```

위치

- position
 - 위치를 조절함.
 - static
 - 기본 값
 - top, left... 위치 스타일 적용 안됨.
 - relative
 - 상대 좌표로 이동
 - absolute
 - 절대 좌표로 이동
 - z-index로 레이어 레벨 정리
 - fixed
 - 고정 위치

→ 파일 : css/05_position.html

크기

- width & height
 - 박스모델의 너비와 높이 설정.
 - Inline 엘리먼트는 height 속성이 적용 안됨.

간격

- margin & padding
 - 박스모델에서 border를 경계로 **안쪽과 바깥쪽의 여백** 설정
 - margin : 바깥 여백
 - padding : 안쪽 여백

테두리

- border
 - 박스모델의 **경계선**
 - solid, dotted...
 - border: 1px solid #ff0000;

배경

- background
 - background속성으로 **배경의 색과 이미지** 등을 설정
 - CSS sprite으로 이용.
 - background:#0000ff url(...) 50% 3px no-repeat
 - background-color
 - background-image
 - background-position
 - background-repeat

→ 파일 : css/06_background.html

폰트

- font
 - 글꼴의 스타일을 지정
 - font, color, letter-spacing
 - text-align, text-decoration ,
 - text-indent, vertical-align, white-space 등

동작

- Animation CSS3

기존의 정적인 스타일과 달리 **동적인 스타일**이 가능

- transition

→ 특정 조건으로 시간 안에 스타일을 변경

- transform

→ 엘리먼트를 회전, 비율, 위치 등을 변경

- animation

→ 시작~종료등 포인트를 지정하여 애니메이션

Transition

opacity: 0;

transition-property: opacity; //속성

transition-duration: 1s; //시간

transition-delay : 1s; //대기 시간

transition-timing-function: ease; //움직임

Transform

transition: transform 2s ease-in-out;

transform: scale(1.5); //크기 변경

//translate(위/치)

//rotate(회전)

//skew(기울기)

Animation

```
@keyframes anima {  
  from { //%로 가능  
    opacity: 0;  
    height: 10px;  
    transform: scale(1);  
  }  
  to {  
    opacity: 1.0;  
    height: 300px;  
    transform: rotate(1.5);  
  }  
}
```

→ 파일 : css/07_animation.html

```
div{  
  animation-name: anima ;  
  //이름  
  animation-duration: 2s;  
  //시간  
  animation-iteration-count: infinite;  
  //반복 횟수  
  animation-timing-function: ease;  
  //움직임  
  animation-direction: alternate;  
  //진행 방향  
}
```

//animation-play-state :
진행상태(paused, running)
//animation-delay :
대기 시간

간단한 CSS 실습

→ 파일 : `html/todo.html` 사용

TODO

할 일을 입력하세요.

추가

- 완료 ☐ 청소 하기.
- 완료 ☐ 빨래 하기.
- 완료 ☐ 숙제하기
- 완료 ☐ 문서 작성하기
- 완료 ☐ 병원가기
- 완료 ☐ 세금내기

- body - 배경색은 #EDC877
- h1-color는 #583A38
- #wrap - 넓이는 600px
margin : 상하60px 중앙정렬

TODO



- ☐ 청소 하기.
- ☐ 빨래 하기.
- ☐ 숙제하기
- ☐ 문서 작성하기
- ☐ 병원가기
- ☐ 세금내기

- #write - border를 1px solid #38384 text-align으로 중앙 정렬
배경색을 #8EA55D

TODO



- ☐ 청소 하기.
- ☐ 빨래 하기.
- ☐ 숙제하기
- ☐ 문서 작성하기
- ☐ 병원가기
- ☐ 세금내기

- `#todolist` - margin, padding을 0px로 셋팅
border를 1px solid #383840
배경색을 #DD955A
color을 #583A38

TODO

- `#todolist li` - `list-style`을 제거(`none`)
하단 `border`만 `1px solid #383840`
`padding`은 `10px`

TODO

할 일을 입력하세요.		추가
<input type="checkbox"/>	청소 하기.	
<input type="checkbox"/>	빨래 하기.	
<input type="checkbox"/>	숙제하기	
<input type="checkbox"/>	문서 작성하기	
<input type="checkbox"/>	병원가기	
<input type="checkbox"/>	세금내기	

- **#write textarea**

width: 400px;
height: 40px;
color: #FFFFFFF;
border: 1px solid #383840;
background: none;
margin-top: 20px;
overflow: hidden;

- **#write input[type=submit]**

padding: 15px;
border: 0;
border-radius: 4px;
color: #fff;
background-color: #87ABC5;
font-weight: bold;
position: relative; bottom : 18px;

고맙습니다.
