**[생활코딩 – 웹 애플리케이션 만들기]**

* 웹 에플리케이션을 만드는 순서

구상->기획->디자인->개발->테스트(QA)->…반복…

Start small => 일단 완성하는 것이 중요.

* 구상

ㅋㅋㅋ전략 <http://goo.gl/ZuuCbh>

* 기획

어떻게 동작해야 하는가?

UI(User Interface)모델링 : 사용자가 시스템을 제어하는 조작장치를 가상으로 만들어 보기

* 모델링 도구: pencil(https://pencil.evolus.vn/), balsamiq(https://balsamiq.com/download/), powerpoint
* 인터넷과 웹의 역사

인터넷 != 웹 (인터넷 > 웹)

인터넷 > FTP, 웹, 이메일 …

1960’s 인터넷 등장 : 네트워크,

1980’s 웹 : 팀 버너스리가 인터넷을 활용해서 HTML을 전송하는 방법을 고안

* 웹브라우저, 웹서버가 서로 정보를 주고받음 http라는 방법을 이용 html언어로..
* 서버와 클라이언트

클라이언트: 웹브라우저가 설치되어있는 컴퓨터, 요청

서버: 웹서버가 설치되어있는 컴퓨터, 응답

* 웹서버: 한대 또는 다른 곳에 서버를 설치.
* Apache, Nginx, IIS, Google(구글 자사가 사용하는 서버)… 등등의 웹서버가 있음. 아파치가 1등 웹서버
* 아파치 웹서버 설치: <http://projects.apache.org/projects/http_server.html>
* Bitnami : apache, php, mysql을 한번에 설치해 줌. Serve sw를 사용하기 쉽게 함.
* localhost/index.html : 웹브라우저가 설치되어 있는 컴퓨터의 웹서버에게 index.html 요청 -> 웹서버는 htdocs에 있는 index.html파일을 읽어서 응답
* bitnami가 설치되어 있는 파일/apache2/htdos/index.html
* htdocs는 특수 디렉터리. 여기(Document Root)에 저장해야 웹서버가 읽을 수 있음
  + - * 서버제어

Bitnami를 이용해 아파치,mysql,php를 제어함 // mac, linux는 자동으로 켜지지 않음

Bitnami manager 실행시켜서 manage servers에서 mysql,aphach 켜줌

configure로 error log확인 가능

* + - * HTML 이론

웹브라우저의 요청에 따라 웹서버는 응답 (HTML을 전송).

HTML(HyperText Markup Language)

* 하이퍼텍스트 : 문서와 문서의 연결(Link)
* Markup : Tag. <시작태그> 컨텐츠 </끝태그>
  + - * HTML 실습

모델링을 HTML로 만들기

* 대제목 <h1> ~ <h6>
* 리스트 <li>

의미론적인 웹

* 정보의 의미를 더 잘 드러내주는 웹
* HTML5 : 웹을 의미론 적으로 표현할 수 있도록 하는 표준
* <ol>: 순서가 있는 list. (<-> <ul>) 네비게이션으로 쓸 부분
* <nav> 기능이 없지만, 약속되어 있는 태그를 통해서 누가봐도 네비게이션이라는 정보를 알 수 있게 함.
* <ul> 본문으로 쓸 부분
* <article> 기능이 없지만, 누가봐도 본문이라는 정보를 알 수 있게 함

사이트 완성

* index.html : 대문페이지라는 것이 약속되어 있는 특수한 파일.

쉬어가기

* 웹브라우저 : HTML CSS JavaScript -> client side technology
* 웹서버: php, MySQL, mongoDB -> server side technology
  + - * 개발도구

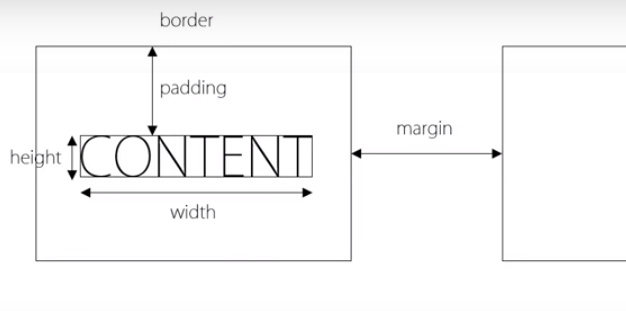
개발도구 : 삽 vs 포크레인

* 버전 관리 시스템 : SVN, Mecurial, GIT
* 코드 편집기(Editor) : atom - GitHub에서 만든 에디터 , Aptana, Sublime Text, Brackets
* Atom
  + Openfolder-> Document Root 설정
  + 확장기능 : File -> settings -> install (ctrl+ , )
  + Ex) emmet : 태그를 한방에 완성시킴 - \*4 , !html
    - * CSS 이론

CSS 기본문법

* HTML + CSS = HTML이라는 언어를 디자인하기 위한 것 (정보 + 디자인)
* <style> : html언어. 이후에 나오는 코드가 CSS문법으로 읽어질 것이라는 것 알림
* header h1 { color:red }
* 선택자 { 속성1; 속성2 }

박스모델

* 구글 개발자 도구 : 요소검사에서 값을 변경해볼 수 있음. 변경 내용은 저장되지 않음

Float

* 이미지/텍스트(컨텐츠)를 원하는 위치에 놓아줌
  + - * CSS 실습

CSS 파일 만들기

* 모든 페이지가 같은 CSS로 나올 수 있도록 CSS파일을 분리 -> link로 연결
* 중앙에 있는 CSS파일 하나만 수정하면 모든 html문서의 스타일을 바꿀 수 있음 => 중복의 제거
  + - * 프로그래밍 (JS, PHP이론)

복습과 수업 예고

* .php파일을 웹서버가 읽음 -> PHP에게 넘겨줌 -> php는 <?php>부분을 php문법으로 해석 -> php가 html파일로 만듦 -> 웹서버가 받아 웹브라우저에게 응답

JavaScript vs PHP

* bitnami에서 Php를 그냥 사용하기에는 불편한 점이 있어서 설정 바꾸는 것 필요
* bitnami/php/php.ini를 atom으로 열어서 opcache.enable = 0으로, display\_errors = On;
* 실제로는 opc=1로, dis의 ‘= On’을 지워야 성능, 보안 면에서 좋음
* 환경설정 변경 시 apache를 다시 껐다 켜야 함
* JavaScript vs PHP
  + 경쟁적인 관계, 협력적인 관계
  + HTML, CSS : 정적인 언어, 문서를 만들기 위한..
  + JavaScript, php : 동적인 언어
  + JavaScript : HTML과 CSS는 한번 화면에 그려지면 바꿀 수 없음. But 사용자가 버튼을 누르면 바뀔 수 있도록 웹브라우저에게 명령할 수 있는 프로그래밍 언어
  + PHP : html에서 정보와 html을 나눠 생각할 수 있게 함. CSS코드 분리처럼… 하나의 HTML로 여러 정보들을 표현할 수 있는 방법

웹페이지에 코드 삽입하기

* JS
  + document.write( ) : js의 출력 문법
  + 클라이언트사이드 언어이기 때문에 10+10을 출력 시, 웹브라우저에서 실행되어 출력하기 때문에 코드가 변경되지 않고 클라이언트가 계산하여 출력
* Php
  + echo : php의 출력 문법
  + 서버사이드 언어 이기 때문에 10+10을 출력 시, php는를 계산하고 코드를 20으로 바꾼 뒤 웹서버에게 전송, 웹서버는 20을 출력

데이터타입과 연산자

* 숫자를 쓸 때는 “”를 쓰면 안됨. “1000”은 문자 천
* Js는 문자를 더할 때도 + 사용, 숫자를 더할 때도 + 사용 => “10”+”10” 을 1010로 출력
* Php는 문자를 더할 때는 . 사용, 숫자를 더할 때는 + 사용 => “10”.”10”은 1010, “10”+”10”은 20으로 출력 +기호 있으면 문자여도 숫자 덧셈을 문자로 잘못 표기한 것이라 여김
* 연산자
  + php에러: bitnami/apache2/logs/error.log
  + js에러: 웹창에서 도구더보기->자바스크립트 콘솔

변수

* 변수 : 바뀔 수 있는 값 => 프로그램을 최대한 변경하지 않게 하기 위해서 사용
  + JS: name = “이고잉”;
  + PHP: $name = “이고잉”; //php는 변수 선언 시 $를 사용
* 비교 (True/False : Boolean형)
  + JS: document.write(2==1);
  + PHP: var\_dump(2==1);
* 조건문
  + JS: result = (1==2); if(result) { … } else { … };
* 로그인 기능 구현
  + JS를 이용한 로그인
    - Prompt: 경고창(사용자 입력가능 창)
  + PHP를 이용한 로그인
    - URL에 입력한 비밀번호를 표시함 -> post방식으로 바꿔야
    - 로그인.php -> 비번확인.php?password

<form action = “비번확인.php”> <input type = “text” name = “password”> <input type = “submit”> </form> // \*\*\*\*로 보이면서 입력 받으려면 type = “text”를 “password”로 바꿔주면 됨, POST방식으로 입력받으려면 <form action = “asd.php” method=”post”>하고, asd.php에서 $\_POST[“password]로 받으면 됨

* + - 비번확인.php

$pw = $\_GET[“password”]; if($pw == “1111’) {…} else {…};

반복문

* JS, PHP 반복문 구조 유사
* JS에서 document.write("<li>hello world</li>");를 html로 변경하는 과정에서 <li>hello world</li>가 HTML 코드로 의미하게 됨

배열

* JS : list = newArray(“a”,”b”,”c”);//배열 생성 list[0]; //배열 사용 list.length //배열길이
* PHP : $list = array(“a”,”b”,”c”);//배열 생성 $list[0]; //배열 사용 count($list) //배열길이

함수

* 서로 연관 되어 있는 부품들을 함수로 묶어서 하나의 부품으로 만듦
* function fname ( ) { … } //function은 약속되어 있는 함수 정의 명령문

UI vs API

* UI: User Interface.
* API: Application Programming Interface. 애플리케이션이 무엇인가를 하기 위해서 프로그래밍적으로 그 애플리케이션이 돌아가는 시스템에 전달하는 명령. 웹브라우저가 미리 만들어 놓은 것을 실행시키도록 하는 명령
* Interface: 서로 소통하기 위한 접점

프로그래밍 접근방법

* 적정 용도의 언어를 적정 수준으로 배우자
* 2Track전략: 단기적으로는 서툰 실력으로 무언가를 만들며 장기적으로는 그 언어를 마스터
* API탐험하기
  + - * JavaScript 실습

JS: HTML을 제어하는 언어. 사용자의 제어에 따라 페이지가 동적 변환

JS는 <script>안에 있을 수도 있지만 HTML태그안에도 있을 수 있음

Class

* class라는 속성을 줘서 하나로 묶을 수 있음. 그룹핑 <li class=”em”>… </li> -> .em { style }

Div

* 단순히 어떠한 태그들을 묶어주는 것
* <div id=”name”> < >< > </div> -> #name { }

HTML에 JS를 끼워넣는 것은 그리 좋은 코드는 아님

* html코드에는 id주고, </body>위에 <script> </script> 사이에 스크립트 코드 넣음
* but 이것은 여전히 중복의 문제 있음
* script.js 라는 파일에 js코드 넣고,
* html파일에는 원래 js코드위치했던 부분에 <stript src = “<http://js파일경로>”></script>로 바꾸면 중복 제거
* HTML: 정보 / CSS: 디자인 / JS: 제어

활용

* 댓글 서비스 : Disqus를 이용함 / livere.com(국내서비스)
  + Disqus.com -> login -> I want to install disqus on my site -> yes, I understand -> 코드카피 -> 댓글을 넣고 싶은 부분에 붙여넣기
* 채팅 서비스 : tawk를 이용함 / sendbird.com(국내서비스)
  + Tawk.to -> login -> admin -> property 추가버튼으로 사이트 추가 -> widget code 카피-> 원하는 위치에 붙여 넣기
    - * PHP 실습

과정

* 웹브라우저가 .php 파일을 웹서버에게 요청시, 웹서버는 php서버에게 요청 후 응답

분리

* Index.php로 분리된 html, 정보는 database에 보관
  + index.php는 정보 수정되어도 변화없음
* 정보로부터 html을 제거해서 정보만으로 이루어진 파일 만들기
  + 실제 정보는 txt파일
  + Html 정보는 php파일
  + <?php echo file\_get\_contents($\_GET['id'].".txt"); ?>
* <http://localhost/index.php> -> 불안전함
  + if( empty($\_GET['id']) == false ) {…} //id가 비어있지 않음
    - * 데이터베이스(MySQL)이론

데이터베이스

* 정보를 관리하는 전문적인 애플리케이션
  + 정보를 안전하고 효율적으로 관리하는 곳
* 파일 vs DB
  + 안전성
  + 빠르다 (인덱스)
  + 프로그래밍적 제어가능
* 관계형 DB
  + MySQL, MS-SQL, Oracle
* MySQL
  + 오픈소스
  + MariaDB : Oracle로 인수되면서 파생. MySQL과 완벽 호환.
* SQL : Structured Query Language
  + Structured : 표. Table
  + SQL문에 대한 질의를 DB서버가 처리 후 응답
  + 웹브라우저:client – 웹서버:server - (php:client – MySQL:server)

MySQL Client and Server

* MySQL monitor : 표준적인 MySQL client. DB서버가 동작하는 모습 투명하게 보여줌
  + 모든 MySQL 설치파일에 기본으로 장착되어 있음
  + 운영체제마다 실행방법 다름
  + 윈도우: win+R -> cmd -> D:\Web\_bitnami\mysql -> 입력: mysql -hlocalhost -uroot –p -> password입력: sujung05
  + cmd에서 mysql을 실행 불가하다고 뜨는 것은 path설정을 해주지 않았기 때문.
* mysql -hlocalhost -uroot –p
  + mysql : mysql monitor를 실행
  + -hlocalhost : localhost 는 mysql server의 주소. Mysql monitor입장에서 localhost는 mysql monitor가 설치되어 있는 컴퓨터 자체
  + -uroot : root는 관리자 아이디
  + -p : 비밀번호를 입력 받아라

Table

* DataBase는 Table들을 관리하는 디렉토리
* DB서버에는 여러 DB가 있을 수 있고, DB에는 여러 Table들이 있을 수 있다.

데이터베이스 보기

* show databases;

데이터베이스 생성

* CREATE DATABASE opentutorials CHARACTER SET utf8 COLLATE utf8\_general\_ci;
  + Opentutorials : 프로젝트 이름
  + CHARACTER SET utf8 : 입력하는 정보가 어떤 방식의 문자인지 알림

데이터베이스 선택

* use opentutorials;
  + 해당 디렉터리, 데이터베이스 안으로 들어감.

데이터베이스 생성

* CREATE TABLE `topic` (…);
  + ` : ‘~’버튼 아래에 있는 것. 문법적 요소와 충돌 방지
  + AUTO\_INCREMENT : 행이 추가될 때마다 자동으로 숫자 1씩 증가
* show tables; : 생성된 Table 확인
* 입력
  + 정보 입력 : 행 추가
  + 구조 추가 : 컬럼 추가, 속성 추가
  + 문자를 쓸 때는 ‘ ’
    - * MySQL 실습

MySQL

* 데이터를 파일로 저장되어 있던 부분을 개선하여 MySQL로 저장
* PHP는 웹서버와 MySQL 사이에서 중개자 역할
  + PHP는 미들웨어
  + MySQLi : PHP에 내장되어 있는 API를 사용하여 중개자 역할을 함
  + 서버접속 -> DB선택 -> 조회 -> 출력

MySQL monitor vs MySQLi

* 서버 접속
  + MySQL monitor : mysql –hlocalhost –uroot –p sujung05
  + MySQLi : $conn = mysqli\_connect(‘localhost’, ’root’, sujung05);
* DB 선택
  + MySQL monitor : mysql > use DB이름;
  + MySQLi : mysqli\_select\_db($conn, ‘DB이름’);
* 조회
  + MySQL monitor : mysql > SELECT \* FROM table이름;
  + MySQLi : $result = mysqli\_query($conn, ‘SELECT \* FROM table이름’);
* 출력
  + MySQL monitor : 테이블이 테이블 형태로 출력
    - * MySQL 연동

MySQLi라는 PHP API를 통해 DataBase와 연동한다.

* MySQLi : 연관배열 형태로 값을 가져온다.
  + $row = mysqli\_fetch\_assoc($result); echo $row[‘id’]; echo $row[‘title’]; …
  + 위의 코드 반복 시, 행들이 하나씩 출력 됨. (id=1, id=2, id=3…);
  + while($row = mysqli\_fietch\_assoc($result)) {…}
* **\*\* list.txt로 연결되어 있었던 부분을 database로 바꾸기**
  + <?php while($row = mysqli\_fetch\_assoc($result){ echo '<li><a href="<http://localhost/index.php?id=>'.$row['id'].'">'.$row['title'].'</a></li>'."\n";} ?>

웹에서 사용자 입력 받기

* Form : 사용자의 정보를 서버로 전송하는 수단
  + 사용자가 웹을 통해서 정보를 작성할 수 있도록 함
  + <form action =”http://localhost/php/3.php” method=“POST”> <p>제목: <input type:”text” name=”title”></p> <p>본문: <textarea name=”description”> </textarea></p> <input type = “submit”> </form>
    - URL길이 제한 때문에 POST방식 씀
    - Form action은 해당 링크로 데이터를 전송해서 해당 링크가 사용자가 입력한 정보를 처리하도록 함.
  + 3.php => $\_POST[‘title’] $\_POST[‘discription’]으로 출력

웹에서 입력 받은 정보를 DB에 넣기

* 처리할 php파일에서 SQL과 연동
  + $sql = "INSERT INTO topic (title,description,author,created) VALUES('".$\_POST['title']."', '".$\_POST['description']."', '".$\_POST['author']."', now())";
    - now(): MySQL에서 제공하는 API. 현재시간이 들어 옴
* 전송된 정보 DB에 넣기
  + 서버접속; DB선택; $sql=…; DB조회;
* DB에 넣은 뒤 사용자를 다시 원하는 위치로 이동시킴
  + header('Location: <http://localhost/index.php>');