

Handling JSON data: Client, meet server

Samkeun Kim <skim@hknu.ac.kr>

<http://cyber.hknu.ac.kr/>

From: Webville MegaCorps Marketing
Subject: Re: 42nd Annual Bit to Byte Race results page

Hey Web Design Team,

We really like the updates you've made to the website.

We have a problem though: nobody in our office knows XML! So we don't know how to add new finishers to the race website.

We've tried, but every time we get it wrong, it makes the website do some strange things... Finishers don't show, or fields disappear from the page even though they're in the XML file. It's very odd.

What we'd really like is some way to just type into a few boxes and click a button to add a finisher. Can you make this happen?

And if we make a mistake, can you make it so we don't break the whole site?

I know it's only three days until we all fly out to Hawaii, but we'd really like this working before we go. Do you think you can make it in time?

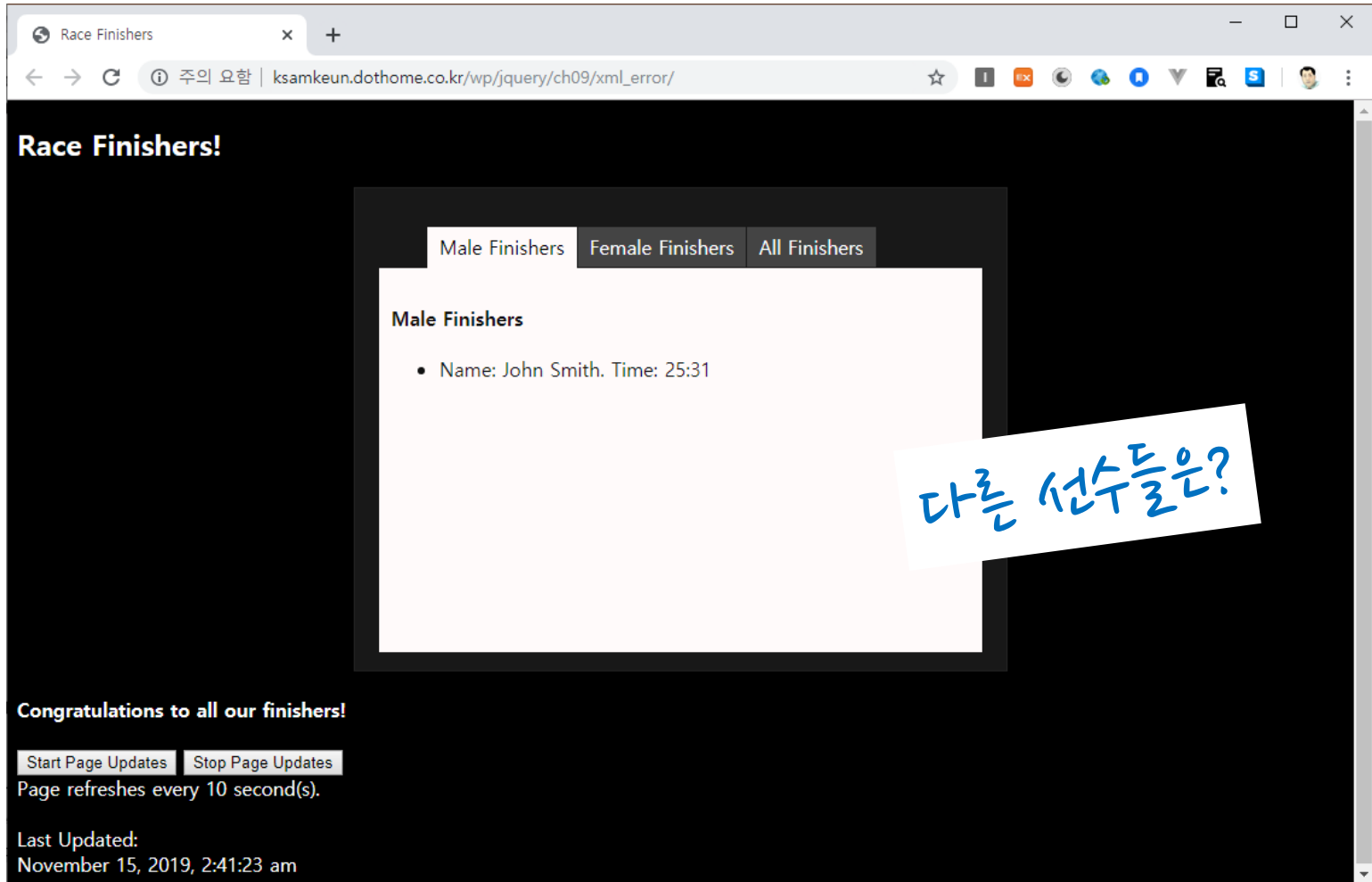
--

Dionah C. Housney
Head of Marketing
Webville MegaCorp



*Webville MegaCorp's Marketing Department
doesn't know XML.*

XML errors break the page

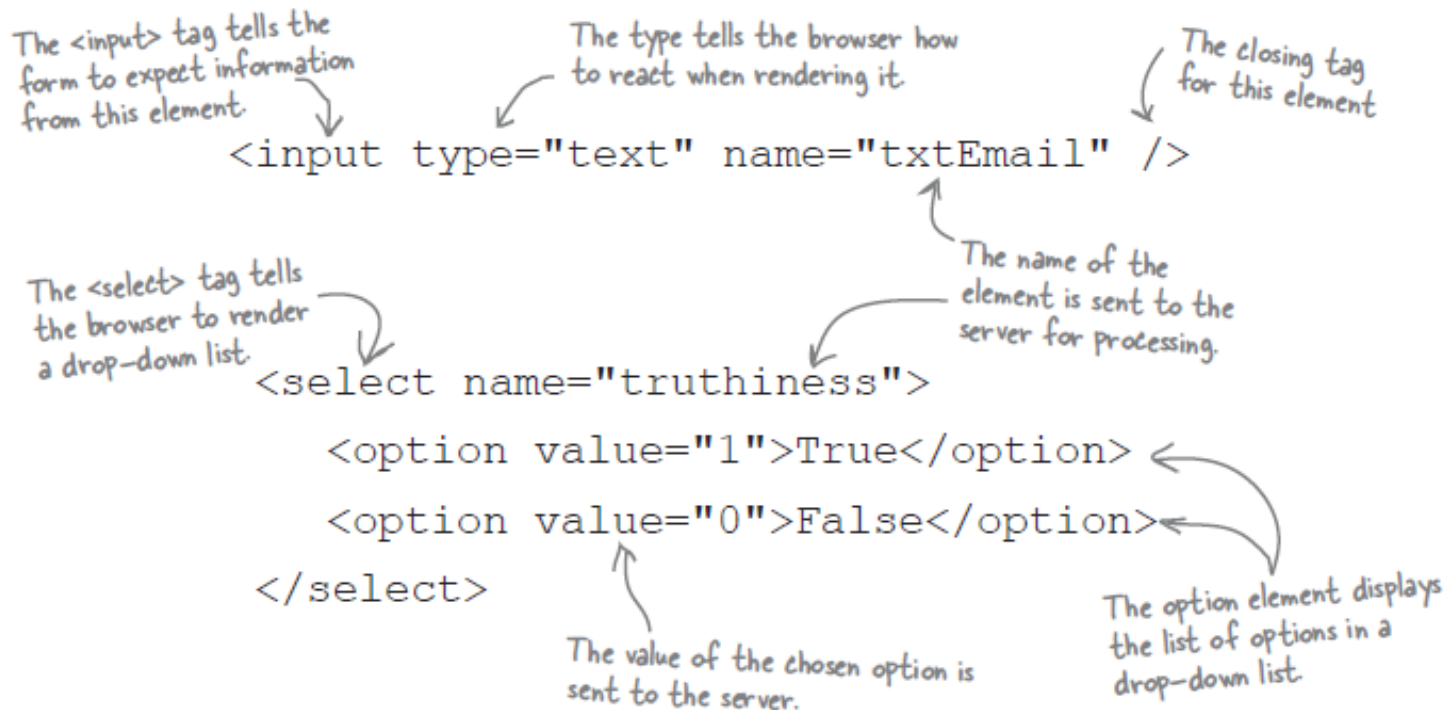


http://ksamkeun.dothome.co.kr/wp/jquery/ch09/xml_error/

Collect data from a web page

폼(form)을 통해 다양한 정보를 수집해서 서버에 보내 처리할 수 있다.

폼에는 다양한 엘리먼트(태그)가 있어서 여러 가지 타입의 데이터를 수집할 수 있다.



Ready Bake HTML & CSS

```
<ul class="idTabs">
  <li><a href="#male">Male Finishers</a></li>
  <li><a href="#female">Female Finishers</a></li>
  <li><a href="#all">All Finishers</a></li>
  <li><a href="#new">Add New Finisher</a></li>
</ul>
<div id="male">
  <h4>Male Finishers</h4><ul id="finishers_m"></ul>
</div>
<div id="female">
  <h4>Female Finishers</h4><ul id="finishers_f"></ul>
</div>
<div id="all">
  <h4>All Finishers</h4> <ul id="finishers_all"></ul>
</div>
<div id="new">
  <h4>Add New Finisher</h4>
  <form id="addRunner" name="addRunner" action="service.php" method="POST">
    First Name: <input type="text" name="txtFirstName" id="txtFirstName" /> <br>
    Last Name: <input type="text" name="txtLastName" id="txtLastName" /> <br>
    Gender: <select id="ddlGender" name="ddlGender">
      <option value="">--Please Select--</option>
      <option value="f">Female</option>
      <option value="m">Male</option>
    </select><br>
    Finish Time:
    <input type="text" name="txtMinutes" id="txtMinutes" size="10" maxlength="2" />(Minutes)
    <input type="text" name="txtSeconds" id="txtSeconds" size="10" maxlength="2" />(Seconds)
    <br><br>
    <button type="submit" name="btnSave" id="btnSave">Add Runner</button>
    <input type="hidden" name="action" value="addRunner" id="action">
  </form>
</div>
```

```
#main {
  background:#181818;
  color:#111;
  padding:15px 20px;
  width:600px;
  border:1px solid #222;
  margin:8px auto;
}
```

Add the new tab, called
"Add New Finisher."

Add a new HTML form for collecting
and posting data to the server.

The action tells the form where to
be sent for processing.

The method
how the data
will be sent to
the server.

A hidden HTML field. We'll
use this more in a little bit.



index.html

실습과제 20-1 Test Drive

Race Finishers! x +

← → ↻ ⓘ 주의 요함 | ksamkeun.dothome.co.kr/wp/jquery/ch09/step1/ ☆ ⓘ

Male Finishers Female Finishers All Finishers Add New Finisher

All Finishers

First Name:

Last Name:

Gender: --Please Select-- ▾

Finish Time: (Minutes) (Seconds)

Congratulations to all our finishers!

Page refreshes every 10 second(s).

Last Updated:
November 15, 2019, 7:44:18 pm

<http://ksamkeun.dothome.co.kr/wp/jquery/ch09/step1/>

What to do with the data

Now we need to send the data collected by the form to the server and store it somehow. To do that, we're going to use another language, **PHP**, to insert the data into a *database*. Don't worry! We'll get you up to speed on **PHP** and databases in a bit, but first let's focus on how we get our form data to the server.

There are two methods of sending the data to the server using **HTTP**: **GET** and **POST**. The main difference between **GET** and **POST** is how the data is sent to the server.

GET will append the form field names and values onto the end of the **URL** as key/value pairs. **PHP** can read this information out of an **associative array** called **`$_GET[]`**, which is sent to the server when the form is submitted. The data is visible after the **?** in the **URL**.

POST sends the data—also in an **associative array**, but encoded differently—and is not visible to the end user in the **URL**. The **`$_POST[]`** associative array contains all the information from the form elements. This, like the **`$_GET[]`** array, is a series of key/value pairs of the form element names and values.

HTTP GET method

```
<form id="my_form" method="get" action="x.php">
  <input type="text" name="a" value="1" />
  <input type="text" name="b" value="2" />
</form>
```

Send the data to the server.

x.php?a=1&b=2

```
<?php
echo $_GET["a"] ; // Writes out "1"
echo $_GET["b"] ; // Writes out "2"
?>
```



그런데 Form이 정보 자체를 보낸다면 jQuery나 그 밖의 것들이 왜 필요한거죠?

Yes, the form could send the information...

HTTP POST method

```
<form id="my_form" method="post" action="x.php">
  <input type="text" name="a" value="1" />
  <input type="hidden" name="c" value="3" />
</form>
```

Send the data to the server.

x.php

```
<?php
echo $_POST["a"] ; // Writes out "1"
echo $_POST["c"] ; // Writes out "3"
```

We're going to use POST for this chapter.

jQuery와 Ajax를 사용하면 데이터를 보내거나 받을 때 페이지 전체를 reload 할 필요가 없다!

그러나 서버에 데이터를 보내기 전에 전송에 알맞은 형태로 바꿔야 한다.

Format the data before you send it

serialize

```
<form id="my_form">
  <input type="text" name="a" value="1" />
  <input type="text" name="b" value="2" />
  <input type="hidden" name="c" value="3" />
</form>
```

```
$("#my_form").serialize();
```

The form ID selector

The serialize method

End result

```
a=1&b=2&c=3
```

jquery & Ajax를 사용하여 서버에 정보를 전송
하기 전에 Ajax 콜이 보낼 수 있으면서 서버가
이해할 수 있는 형식으로 바꿔야 한다!

serializeArray

```
<form id="my_form">
  <input type="text" name="a" value="1" />
  <input type="hidden" name="c" value="3" />
</form>
```

```
$("#my_form:input").serializeArray();
```

The form's ID selector, followed
by the HTML element input filter.
This tells the selector to only look
at HTML elements of type "input."

Call the
serializeArray
method.

End result

```
[
  {
    name: "a",
    value: "1"
  },
  {
    name: "c",
    value: "3"
  }
]
```

JSON 객체 형식!

Send the data to the server

post() method:

서버에 데이터를 보내기 위해 전용으로 만들어진 단축 메소드

post() 메소드 파라미터:

서버 주소를 지정하는 URL, 보낼 데이터, POST 요청이 성공했을 때 실행할 콜백 함수

jQuery shortcut → `$.post` (*데이터를 보낼 곳(서버 url)*) (*URI 직렬화시켜야 함!*) (*반환된 객체(데이터)*)
`url_to_send, data, function(json) {`
`});`
The URL you want to send the data to
The data you want to send, which has been serialized already
Run this callback function.
The returned data, in an object called json. Don't worry about this one right now; we'll get to it a bit later in the chapter.

jQuery Code Magnets

```
$('#btnSave').click(function() {  
    var data = $("#addRunner :input").serializeArray();  
    $.post($("#addRunner").attr('action'), data, function(json){  
        if (json.status == "fail") {  
            alert(json.message);  
        }  
        if (json.status == "success") {  
            alert(json.message);  
            clearInputs();  
        }  
    }, "json");  
});  
  
function clearInputs() {  
    $("#addRunner :input").each(function() {  
        $(this).val('');  
    });  
}  
  
$("#addRunner").submit(function() {  
    return false;  
});
```

Prepare all the form fields for sending to the server.

Get the action attribute of the form you want to post.

Check the return value from the server, set in the PHP code, to see if the POST was successful or not.

Use an HTML element filter to access all the input fields in the form, and set them all to be empty.

Cancel the default submit action of the form to allow the jQuery code in the button click event to take care of sending the data.



실습과제 20-2 Test Drive

The screenshot shows a web browser window with the URL `ksamkeun.dothome.co.kr`. The page title is "Race Finishers!". Below the title, there are tabs for "Male Finishers", "Female Finishers", and "All Finishers", along with a link "Add New Finisher". The "All Finishers" tab is active, showing a form with the following fields:

- First Name:
- Last Name:
- Gender:
- Finish Time: (Minutes) (Seconds)

Below the form is a button labeled "Add Runner". At the bottom of the page, there is a message: "Congratulations to all our finishers!". Below this message are two buttons: "Start Page Updates" and "Stop Page Updates". Below these buttons is the text: "Page refreshes every 10 second(s).". At the very bottom, it says "Last Updated: November 15, 2019, 8:09:33 pm".

On the right side of the browser window, the developer tools are open, showing the "Network" tab. The "Filter" dropdown is set to "All". The "Headers" sub-tab is selected for the request `service.php`. The "Preview" sub-tab shows the response, which is "Connected!".

Handwritten text in blue ink on the right side of the image says: "Form 데이터를 저장할 장소?" (Where to save the form data?).

<http://ksamkeun.dothome.co.kr/wp/jquery/ch09/step2/>

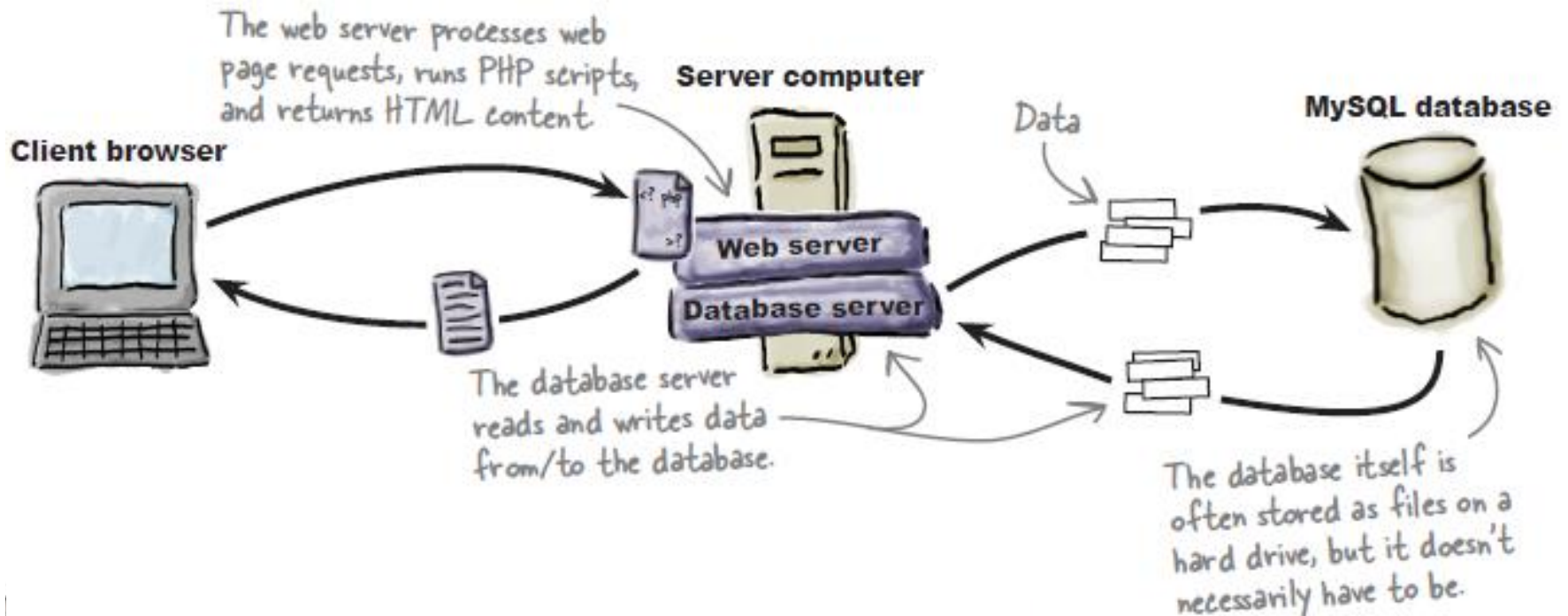
Store your data in a MySQL database

Relational Database Management Systems (RDBMS) are extremely organized applications designed to store, organize, and remember relationships between your various pieces of data.

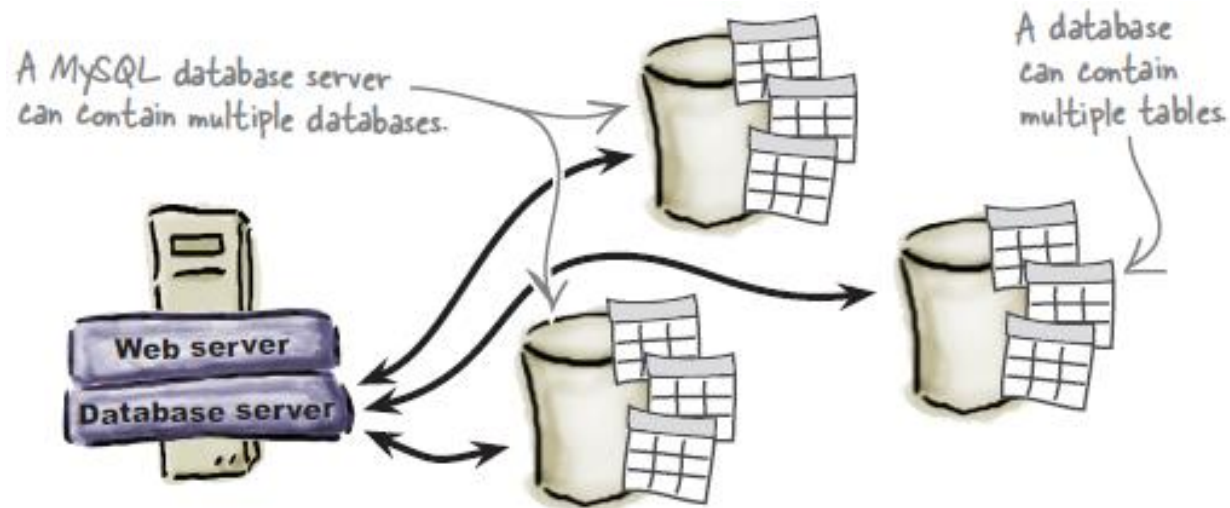
Often called *database servers*, they come in various shapes and sizes (and costs). For our purposes, we'll use a free database server called **MySQL**. You communicate with a database server in a language it can understand, which in our case is **SQL**. A database server typically runs alongside a web server, sometimes on the same server, and they work in concert to read and write data and deliver web pages.

The "SQL" in MySQL stands for Structured Query Language.

MySQL stores data inside of database tables.



MySQL databases are organized into *tables*, which store information as rows and columns of related data. Most web applications use **one or more tables** inside a **single** database, sort of like different file folders within a file cabinet.



SQL is the query language used to communicate with a MySQL database.

Create your database to store runner info

Ready Bake SQL

자신의 Dothome ID (예: ksamkeun)

~~use hfjq_race_info;~~

← Tell the script that the next piece relates to your new database.

```
CREATE TABLE runners(  
    runner_id INT not null AUTO_INCREMENT,  
    first_name VARCHAR(100) not null,  
    last_name VARCHAR(100) not null,  
    gender VARCHAR(1) not null,  
    finish_time VARCHAR(10),  
    PRIMARY KEY (runner_id)  
);
```

<http://ksamkeun.dothome.co.kr/wp/jquery/ch09/runners.sql>

실습과제 20-3 Test Drive

<http://ksamkeun.dothome.co.kr/myadmin/>

1. Select the database 'ksamkeun' in the left sidebar.

2. Select the 'SQL' tab in the top navigation bar.

3. Write the SQL query in the editor:

```
1 use YOUR_DOTHOME_ID;
2
3 CREATE TABLE runners(
4     runner_id INT not null AUTO_INCREMENT,
5     first_name VARCHAR(100) not null,
6     last_name VARCHAR(100) not null,
7     gender VARCHAR(100) not null,
8     finish_time VARCHAR(10),
9     PRIMARY KEY (runner_id)
10 );
```

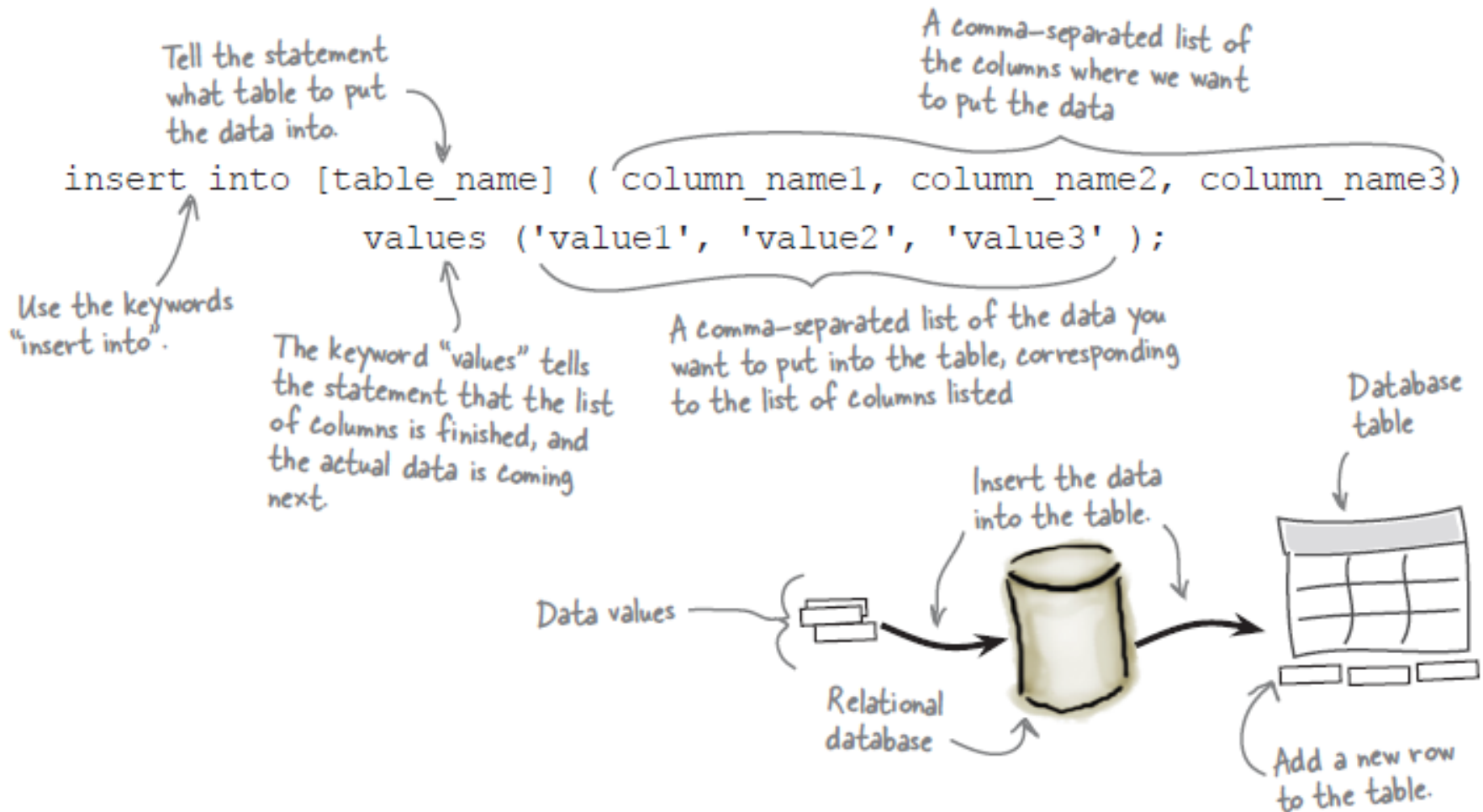
4. Click the '실행' (Execute) button at the bottom right.

자신의 닉네임 ID (Your nickname ID)

3. write

4

Anatomy of an insert statement



Exercise

```
insert into runners (first_name, last_name, gender, finish_time) values ('Jacob','Walker','m','25:54');  
insert into runners (first_name, last_name, gender, finish_time) values ('Mary','Brown','f','26:01');  
insert into runners (first_name, last_name, gender, finish_time) values ('Jenny','Pierce','f','26:04') ;  
insert into runners (first_name, last_name, gender, finish_time) values ('Frank','Jones','m','26:08');  
insert into runners (first_name, last_name, gender, finish_time) values ('Bob','Hope','m','26:38');  
insert into runners (first_name, last_name, gender, finish_time) values ('Jane','Smith','f','28:04');  
insert into runners (first_name, last_name, gender, finish_time) values ('Ryan','Rice','m','28:24');  
insert into runners (first_name, last_name, gender, finish_time) values ('Justin','Jones','m','29:14');
```

Copy & Paste!!

실습과제 20-4 Test Drive

서버: localhost 데이터베이스: ksamkeun

구조 SQL 검색 질의 마법사 내보내기 가져오기 테이블 작업 더보기

데이터베이스 ksamkeun에 SQL 질의를 실행:

```
1 insert into runners (first_name, last_name, gender, finish_time) values ('Jacob','Walker','m','25:54');
2 insert into runners (first_name, last_name, gender, finish_time) values ('Mary','Brown','f','26:01');
3 insert into runners (first_name, last_name, gender, finish_time) values ('Jenny','Pierce','f','26:04');
4 insert into runners (first_name, last_name, gender, finish_time) values ('Frank','Jones','m','26:08');
5 insert into runners (first_name, last_name, gender, finish_time) values ('Bob','Hope','m','26:38');
6 insert into runners (first_name, last_name, gender, finish_time) values ('Jane','Smith','f','28:04');
7 insert into runners (first_name, last_name, gender, finish_time) values ('Ryan','Rice','m','28:24');
8 insert into runners (first_name, last_name, gender, finish_time) values ('Justin','Jones','m','29:14');
9
```

제거 형식 Get auto-saved query

☐ Bind parameters

[구분자 ;] ☒ 이 질의를 다시 보여줌 ☐ 쿼리 박스 유지 ☐ Rollback when finished

☒ 외래 키(foreign key) 점검 사용

실행

	last_name	gender	finish_time
	Walker	m	25:54
	Brown	f	26:01
	Pierce	f	26:04
	Jones	m	26:08
	Hope	m	26:38
	Smith	f	28:04
	Rice	m	28:24
	Jones	m	29:14

수정 복사 삭제 4 Frank

수정 복사 삭제 5 Bob

수정 복사 삭제 6 Jane

수정 복사 삭제 7 Ryan

수정 복사 삭제 8 Justin

모두 체크 선택한 것을: 수정 복사 삭제 내보내기

Use PHP to access the data

웹 브라우저는 PHP와 아무
관계도 없고, PHP 스크립트
를 실행할 수도 없다!



Unlike HTML web pages, which can be opened locally in a web browser, PHP scripts must always be "opened" through a URL from a web server.

This PHP script is just a bunch of meaningless code to the web browser.

The web server understands this PHP code and runs the script!

A quick way to tell if a web page is being delivered by a web server is to look for the URL starting with "http:". Web pages opened as local files always start with "file:".



PHP를 지원하는 웹 서버는 PHP
스크립트를 실행해서 이것을 브라우
저가 이해할 수 있는 HTML 웹
페이지로 바꾼다!

PHP scripts must be
run on a web server
or they won't work.

PHP and MySQL?
I thought we were learning
jQuery here! What gives?



Handle POST data on the server

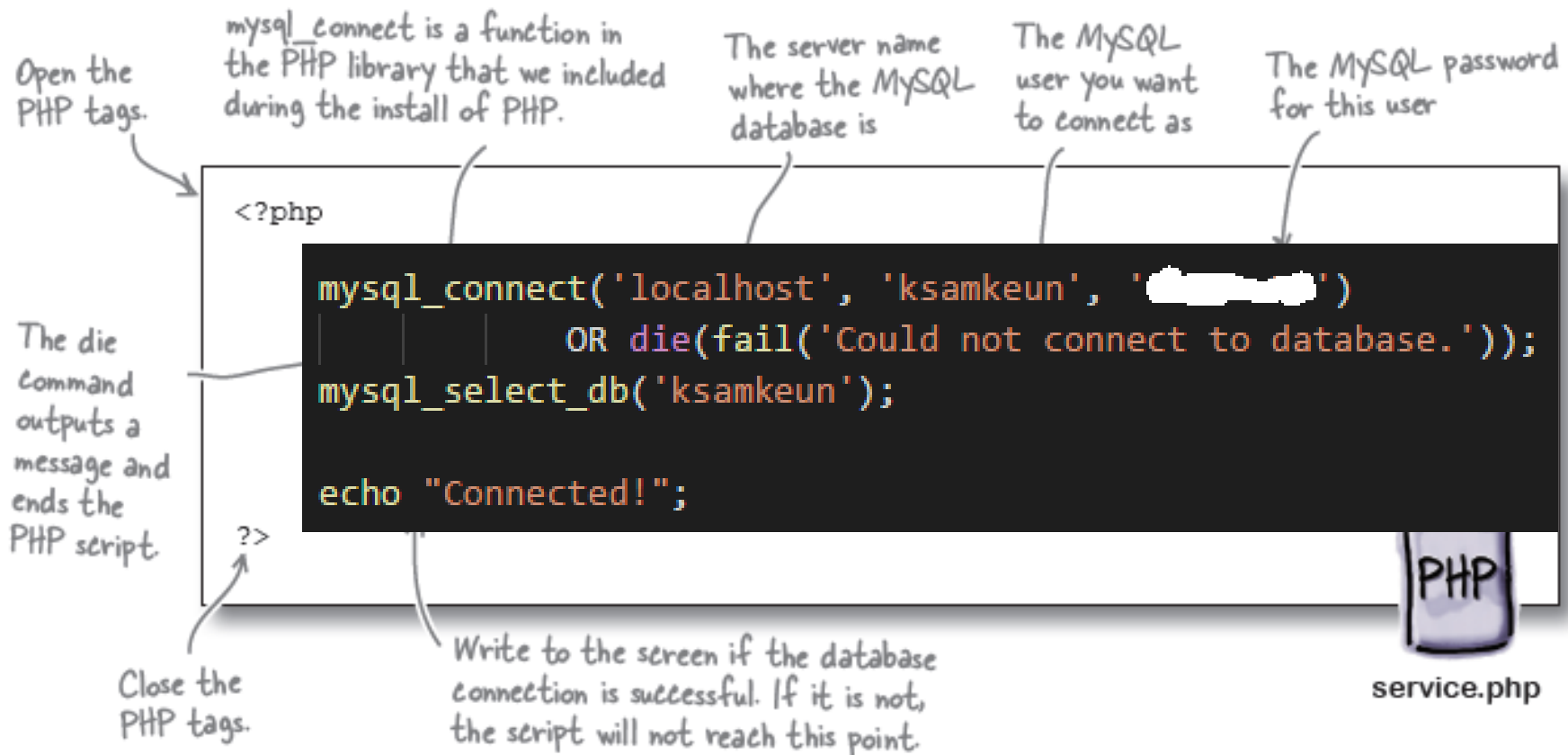
Write the value to the screen.

폼에서 데이터를 수집한
HTML 엘리먼트의 이름

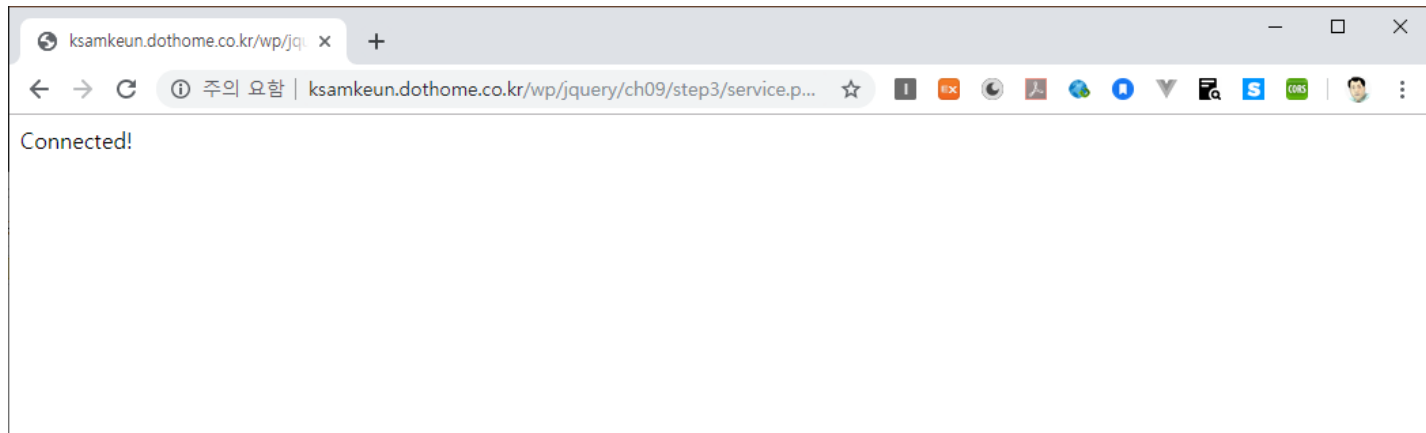
```
echo $_POST["txtFirstName"];
```

POST 방식으로 PHP에 전달된 데이터를
제어하기 위해 자동으로 생성된 배열 이름

Connect to a database with PHP



실습과제 20-5 Test Drive



<http://ksamkeun.dothome.co.kr/wp/jquery/ch09/step3/service.php>

Use select to read data from a database

`select column_name1, column_name2 from table_name order by column_name1 asc`

A comma-separated list of the columns that we want to pull the data from.

Tell the statement what table to pull the data from.

The "asc" keyword tells "order by" how to order the results (asc for ascending, desc for descending).

The "select" keyword kicks off the statement

The "from" keyword tells the statement that the list of desired columns is finished, and where to get the data that comes next.

The "order by" keyword, followed by one or more column names, sorts the returned data in whatever order we tell it.

The SQL select statement retrieves columns of data from one or more tables and returns a resultset.

Sharpen your pencil

The list of columns you need to select

How you want the data ordered

```
SELECT first_name, last_name, gender, finish_time FROM runners order by finish_time ASC
```

The table where you want to get the data from

+ 옵션			
first_name	last_name	gender	finish_time ▲ 1
John	Smith	m	25:31
Jacob	Walker	m	25:54
Mary	Brown	f	26:01
Jenny	Pierce	f	26:04
Frank	Jones	m	26:08
Bob	Hope	m	26:38
Jane	Smith	f	28:04
Ryan	Rice	m	28:24
Justin	Jones	m	29:14
sam	kim	m	60:20

PHP Code Magnets

```
<?php

$query = "SELECT first_name, last_name, gender, finish_time FROM runners
order by finish_time ASC ";
$result = db_connection($query);

while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
    print_r($row);
}

function db_connection($query) {
    mysql_connect('127.0.0.1', 'runner_db_user', 'runner_db_password')
    OR die('Could not connect to database.');
```

mysql_select_db('hfjq_race_info');

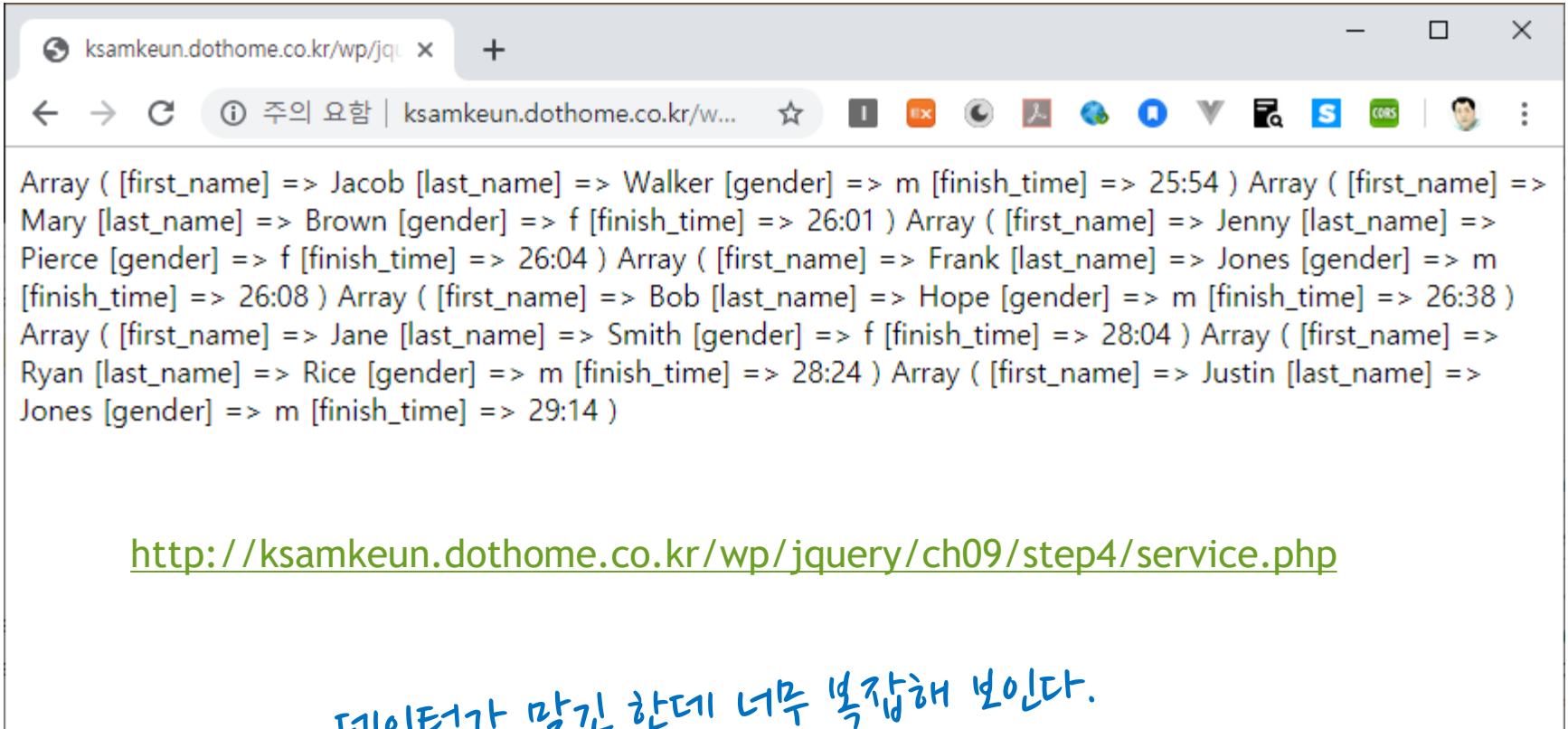
```
    return mysql_query($query);
}

?>
```



service.php

실습과제 20-6 Test Drive



Array ([first_name] => Jacob [last_name] => Walker [gender] => m [finish_time] => 25:54) Array ([first_name] => Mary [last_name] => Brown [gender] => f [finish_time] => 26:01) Array ([first_name] => Jenny [last_name] => Pierce [gender] => f [finish_time] => 26:04) Array ([first_name] => Frank [last_name] => Jones [gender] => m [finish_time] => 26:08) Array ([first_name] => Bob [last_name] => Hope [gender] => m [finish_time] => 26:38) Array ([first_name] => Jane [last_name] => Smith [gender] => f [finish_time] => 28:04) Array ([first_name] => Ryan [last_name] => Rice [gender] => m [finish_time] => 28:24) Array ([first_name] => Justin [last_name] => Jones [gender] => m [finish_time] => 29:14)

<http://ksamkeun.dothome.co.kr/wp/jquery/ch09/step4/service.php>

데이터가 맞긴 한데 너무 복잡해 보인다.

더 깔끔하게 표시할 방법은 없는가?

JSON to the rescue!

Other than the root element, there's no logical connection between the elements. That's why we had to "find()" each one.

XML

```
<?xml version="1.0" encoding="utf-8"?>
<books>
  <book>
    <title>The Color of Magic</title>
    <author>Terry Pratchett</author>
    <year>1983</year>
  </book>
  <book>
    <title>Mort</title>
    <author>Terry Pratchett</author>
    <year>1987</year>
  </book>
  <book>
    <title>And Another thing...</title>
    <author>Eoin Colfer</author>
    <year>2009</year>
  </book>
</books>
```

Multiple copies of the tags increase the amount of data being transferred.

vs.

JSON

```
{
  books: [
    {
      title: 'The Color of Magic',
      author: 'Terry Pratchett',
      year: 1983
    },
    {
      title: 'Mort',
      author: 'Terry Pratchett',
      year: 1987
    },
    {
      title: 'And Another thing...',
      author: 'Eoin Colfer',
      year: 2009
    }
  ]
}
```

The root element

The array that stores the data

The name/value pair, separated by the colon

Each property is separated by commas.

Each object is separated by commas.

Each object is enclosed in curly braces { }.

A string value is enclosed in quotes.

Numbers don't need quotes.

jQuery + JSON = Awesome!!

JSON 데이터 획득 전용 메소드:

jQuery shortcut. → \$.getJSON(url_to_load, function(json) {
Call the getJSON method. →
The URL you want to load the data from → url_to_load
Run this callback function. → function(json) {
The returned data, in an object called json (more on this in a sec). → json
});

.getJSON() => .ajax() 메소드에 대한 단축 메소드

```
$.ajax({  
  url: url_to_load,  
  dataType: 'json',  
  data: json,  
  success: function(json){  
  
  };  
});
```

그런데 현재 우리가 가지고
있는 데이터가 JSON 포맷
이 아니다. 이 배열을
JSON으로 바꿀 수 있나?

Yes, we can.



A few PHP rules...

```
<div><span> Hello
<?php
    echo "Bob";
?>
</span></div>
```

```
<?php
$u = "USA"; // OK
$home_country = "Ireland"; // OK
$another-var = "Canada"; // Causes an error
?>
```

```
<?php
for ($i = 1; $i <= 10; $i++) {
    echo $i;
}
while ($j <= 10) {
    echo $j++;
}
$a = array(1, 2, 3, 17);
foreach ($a as $v) {
    echo "Current value: $v.\n";
}
?>
```

A few (more) PHP rules...

```
<?php
$my_arr2 = array('USA', 'China',
'Ireland');

echo $my_arr2[2]; // Prints "Ireland"

$arr = array("foo" => "bar", 12 => true);
echo $arr["foo"]; // Prints "bar"
echo $arr[12]; // Prints true
?>
```

```
<?php
    echo "Bob";
    print_r($my_arr2);
?>
```

배열 요소를 화면에 표시

If 문은 JavaScript 문법과 똑같다.

```
<?php
if ($x > $y){
    echo "x is greater than y";
}
elseif ($x == $y) {
    echo "x is equal to y";
}
else {
    echo "x is smaller than y";
}
?>
```

Format the output using PHP

PHP의 **json_encode** 함수는 배열을 인자로 받아서 JSON 객체 형식의 문자열로 인코딩해서 반환한다:

Write the value out to whatever called the file—i.e., a browser or an ajax call, etc. → `echo json_encode(array_name);`

Call this PHP function to encode the array in the JSON format. →

Pass in an array to encode. →

PHP의 **array_push** 함수는 배열의 마지막에 새 항목을 추가한다:

Create a new, empty array. → `$my_array = array();`

Pass in any information you want to add to the array. In this case, another associative array is getting added to the `$my_array` array. → `array('my_key' => 'my_val')`

Call the `array_push` function with parameters. → `array_push($my_array,`

Pass the destination array as the first parameter. → `$my_array,`

A name/value pair being added to this array. → `array('my_key' => 'my_val'));`

Exercise

```
function startAJAXcalls() {  
    if(repeat) {  
        setTimeout( function() {  
            getDBRacers();  
            startAJAXcalls();  
        },  
        FREQ  
    );  
}
```

Call the new function on a scheduled basis.

Use the getJSON jQuery method to call the service.php file.

```
function getDBRacers() {  
    $.getJSON("service.php", function(json) {  
        alert(json.runners.length);  
    });  
    getTimeAjax();  
}
```

The data returned from the getJSON call

Like other arrays, this also has a length property.

The json object contains an array called runners. It got this name from the json_encode method in PHP.



my_scripts.js

<?php

The database query to
get the runners

```
$query = "SELECT first_name, last_name, gender, finish_time FROM runners  
order by finish_time ASC ";
```

```
$result = db_connection($query);
```

```
$runners = array();
```

Create a new array to
hold our returned values.

Loop through the
resultset, getting
associative arrays back.

```
while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {  
    array_push($runners, array('fname' => $row['first_name'], 'lname' =>  
$row['last_name'], 'gender' => $row['gender'], 'time' => $row['finish_time']));  
}
```

```
echo json_encode(array("runners" => $runners));
```

```
exit;
```

Put the returned
data into our own
associative array.

Encode our associative array in
the JSON format and write it
to whatever called it.

```
function db_connection($query) {
```

```
mysql_connect('localhost', 'ksamkeun', 'XXXXXXXXXX')  
OR die(fail('Could not connect to database.'));  
mysql_select_db('ksamkeun');
```

DB명: 자신의 닉네임 ID

```
return mysql_query($query);
```

Return the resultset to
whatever called this function.

```
} Handler functions to deal with  
errors or successes in our scripts
```

```
function fail($message) {
```

```
    die(json_encode(array('status' => 'fail', 'message' => $message)));
```

```
}
```

```
function success($message) {
```

```
    die(json_encode(array('status' => 'success', 'message' => $message)));
```

```
}
```

?>

service.php

실습과제 20-7 Test Drive

The screenshot shows a web browser window with the URL `ksamkeun.dothome.co.kr/wp/jquery/ch09/step5/`. The page title is "Race Finishers!". The main content area has a header "Race Finishers!" and a sub-header "Male Finishers". Below the sub-header, there is a list of finishers: Jacob, Walker, Mary, Brown, Jenny, Pierce, Frank, Jones, Bob, Hope, Jane, Smith, Ryan, Rice, Justin, Jones, and Samkeun, Kim. The page also includes a "Congratulations to all our finishers!" message and a "Start Page Updates" button. The network inspector is open, showing a list of requests. The selected request is "step5/" with a status of 200. The response is a JSON array of finishers:

```
{runners: [{fname: "Jacob", name: "Walker", gender: "m", time: "1"}, {fname: "Jacob", name: "Walker", gender: "m", time: "1"}, {fname: "Mary", name: "Brown", gender: "f", time: "1"}, {fname: "Jenny", name: "Pierce", gender: "f", time: "1"}, {fname: "Frank", name: "Jones", gender: "m", time: "1"}, {fname: "Bob", name: "Hope", gender: "m", time: "1"}, {fname: "Jane", name: "Smith", gender: "f", time: "1"}, {fname: "Ryan", name: "Rice", gender: "m", time: "1"}, {fname: "Justin", name: "Jones", gender: "m", time: "1"}, {fname: "Samkeun", name: "Kim", gender: "m", time: "1"}]}
```

<http://ksamkeun.dothome.co.kr/wp/jquery/ch09/step5/>

Access data in the JSON object

PHP의 **json_encode** 함수는 연관배열을 JSON 형식의 문자열로 인코딩한다.

json_encode 함수는 JSON 객체를 반환한다.

JSON 객체를 다룰 때는 **도트(.) 표기법**을 사용해서 속성에 바로 접근할 수 있다.

이 경우 JSON 객체에 속성은 **runners** 배열 하나뿐이다.

Runners 배열을 가져온 다음 연관배열의 키를 통해 그 주자가 남성인지 여성인지 바로 알 수 있다.

Sharpen your pencil

```
function getDBRacers() {  
    $.getJSON("service.php", function(json) {  
        if (json.runners.length > 0) {  
            $('#finishers_m').empty();  
            $('#finishers_f').empty();  
            $('#finishers_all').empty();  
            $.each(json.runners, function() {  
                var info = '<li>Name: ' + this['fname'] + ' ' + this['lname'] + '. Time: ' +  
                this['time'] + '</li>';  
                if(this['gender'] == 'm'){  
                    $('#finishers_m').append( info );  
                }else if(this['gender'] == 'f'){  
                    $('#finishers_f').append( info );  
                }else{}  
                $('#finishers_all').append( info );  
            });  
        }  
    });  
    getTimeAjax();  
}
```

Get the information from the service.php file.

Check if there is data in the runners array.

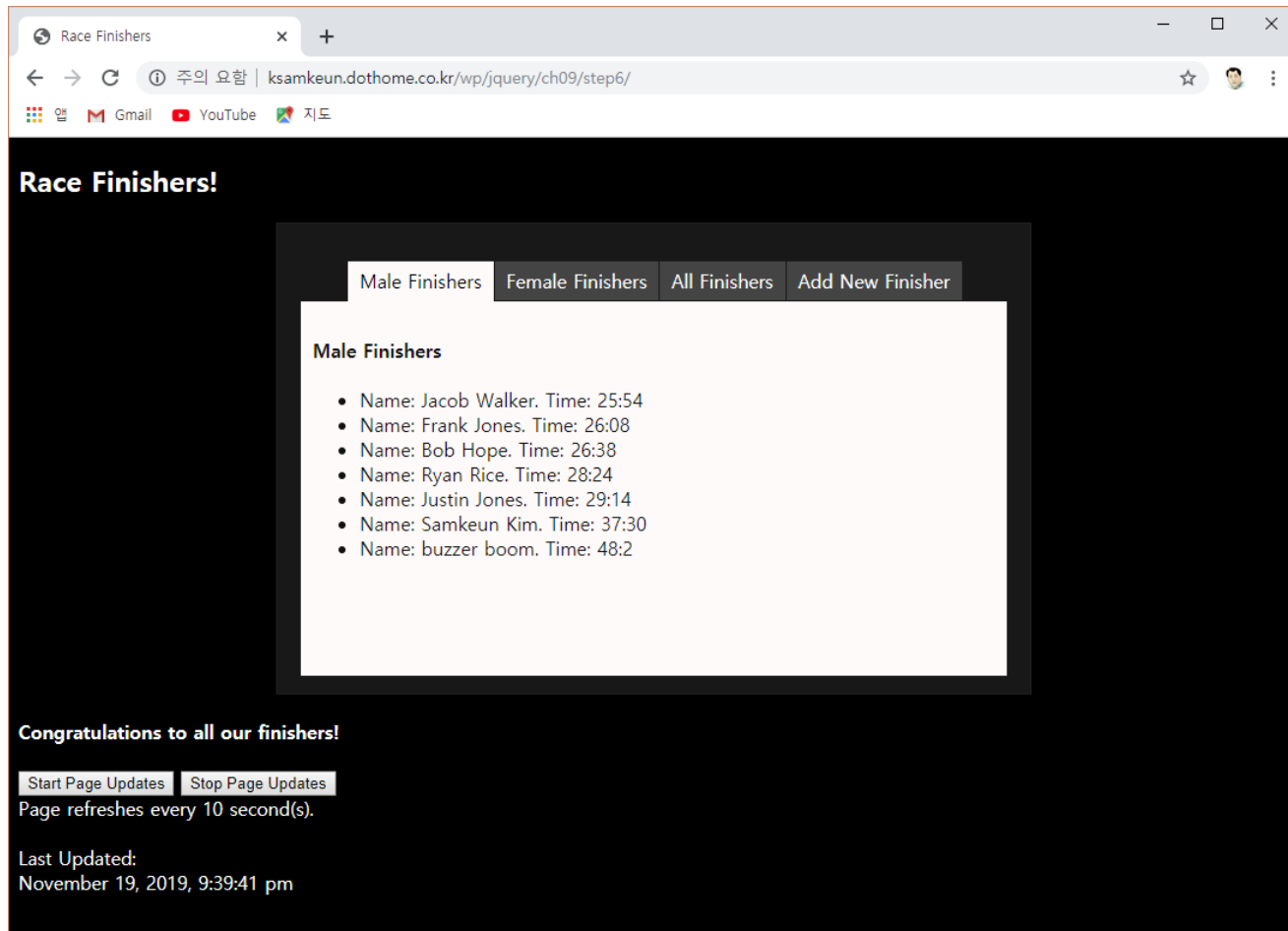
Empty out the lists again.

Check if the current object property of gender is m or f.

Add the runner to the all_runners list.



실습과제 20-8 Test Drive



Race Finishers!

Male Finishers Female Finishers All Finishers Add New Finisher

Male Finishers

- Name: Jacob Walker. Time: 25:54
- Name: Frank Jones. Time: 26:08
- Name: Bob Hope. Time: 26:38
- Name: Ryan Rice. Time: 28:24
- Name: Justin Jones. Time: 29:14
- Name: Samkeun Kim. Time: 37:30
- Name: buzzer boom. Time: 48:2

Congratulations to all our finishers!

Start Page Updates Stop Page Updates

Page refreshes every 10 second(s).

Last Updated:
November 19, 2019, 9:39:41 pm

<http://ksamkeun.dothome.co.kr/wp/jquery/ch09/step6/>

Data sanitization and validation in PHP

```
<?php
    htmlspecialchars($_POST["a"]); // Encode the strings into safer web and database values
    empty($_POST["b"]); // The "empty" method checks if the value is empty
    preg_match('', $var); // This is a "Regular Expression". It checks $var against a set pattern
?>
```

Converts some special HTML entities in a format that is safe for the database

Check if a string is empty or not.

A regular expression matching function. The pattern matching using regular expressions can be very specific, so you can really control the type of data entered.

데이터 소독에는 이 외에도 **htmlentities**, **trim**, **stripslashes**, **mysql_real_escape_string** 등 여러 가지 함수 등이 있다.

Use the same PHP file for multiple purposes

PHP 파일에서 조건 로직을 사용하여 POST 또는 GET 요청을 했는지 찾을 수 있다.

① POST 요청 - P6에서 폼에 추가했던 hidden 필드:

```
<input type="hidden" name="action" value="addRunner" id="action">
```

- POST에서 이 값을 찾을 수 있음

② GET 요청 - getJSON 콜에 URL 파라미터를 갖도록 수정

```
$.getJSON("service.php?action=getRunners", function(json) {
```

Use this to tell the PHP function to run the code associated with getting the runners from the database

Ack! Everyone's clearing their desks and heading to the airport! So, we know how to complete the form now, right?

⇒ PHP 파일에서 조건에 맞는 해당 코드만 실행되게 할 수 있음

Update your **service.php** file with the following code. It will handle both the GET and POST of information. You'll also need to include the **db_connection**, **success**, and **fail** functions from before.

```
<?php
if ($_POST['action'] == 'addRunner') {
    $fname = htmlspecialchars($_POST['txtFirstName']);
    $lname = htmlspecialchars($_POST['txtLastName']);
    $gender = htmlspecialchars($_POST['ddlGender']);
    $minutes = htmlspecialchars($_POST['txtMinutes']);
    $seconds = htmlspecialchars($_POST['txtSeconds']);
    if(preg_match('/[^\w\s]/i', $fname) || preg_match('/[^\w\s]/i', $lname)) {
        fail('Invalid name provided.');
```

Check if there was a value of addRunner POSTed to the server. This is our hidden field from earlier.

Data sanitization of the information in the `$_POST` array

Data validation ensures that something was entered.

```
    }
    if( empty($fname) || empty($lname) ) {
        fail('Please enter a first and last name.');
```

Call the fail function, if the validation fails.

```
    }
    if( empty($gender) ) {
        fail('Please select a gender.');
```

Tell the database to insert a new record... ...and check if it was successful or not.

```
    }
    $time = $minutes.":".$seconds;
    $query = "INSERT INTO runners SET first_name='$fname', last_name='$lname',
gender='$gender', finish_time='$time'";
    $result = db_connection($query);
    if ($result) {
        $msg = "Runner: ".$fname." ".$lname." added successfully" ;
        success($msg);
    } else { fail('Insert failed.')} exit;
}elseif($_GET['action'] == 'getRunners'){
    $query = "SELECT first_name, last_name, gender, finish_time FROM runners order by
finish_time ASC ";
    $result = db_connection($query);
    $runners = array();
    while ($row = mysql_fetch_array($result, MYSQL_ASSOC)) {
        array_push($runners, array('fname' => $row['first_name'], 'lname' => $row['last_name'],
'gender' => $row['gender'], 'time' => $row['finish_time']));
    }
    echo json_encode(array("runners" => $runners));
    exit;
}
```

Check if the getRunners value was sent in the URL string.

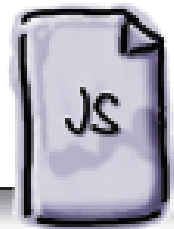
Get and return the runners.



Do this!

Update the **getJSON** call to include a URL parameter called **action** with a value of **getRunners** to tell the **service.php** file to return the runners.

```
function getDBRacers() {  
    $.getJSON("service.php?action=getRunners", function(json) {  
        if (json.runners.length > 0) {  
            $('#finishers_m').empty();  
            .  
            .  
            }  
        });  
    getTimeAjax();  
}
```



my_scripts.js

실습과제 20-9 Test Drive

The screenshot shows a web browser window with the URL `ksamkeun.dothome.co.kr/wp/jquery/ch09/e...`. The page title is "Race Finishers!". It displays a form for adding a runner with the following details: First Name: Samkeun, Last Name: Kim, Gender: Male, Finish Time: 37 (Minutes) 30 (Seconds). A message above the form states: "ksamkeun.dothome.co.kr 내용: Runner: Samkeun Kim added successfully". Below the form is a button labeled "Add Runner".

At the bottom of the page, it says "Congratulations to all our finishers!" and "Page refreshes every 10 second(s)". The last update is noted as "Last Updated: July 3, 2019, 4:36:00 pm".

Overlaid on the right is a database management tool window showing the "runners" table. The table has columns: `runner_id`, `first_name`, `last_name`, `gender`, and `finish_time`. The data is as follows:

runner_id	first_name	last_name	gender	finish_time
1	Jacob	Walker	m	25:54
2	Mary	Brown	f	26:01
3	Jenny	Pierce	f	26:04
4	Frank	Jones	m	26:08
5	Bob	Hope	m	26:38
6	Jane	Smith	f	28:04
7	Ryan	Rice	m	28:24
8	Justin	Jones	m	29:14
9	Samkeun	Kim	m	37:30

The row for runner_id 9 is highlighted with a red box. The database tool also shows a SQL query: `SELECT * FROM 'runners'`.

<http://ksamkeun.dothome.co.kr/wp/jquery/ch09/end/>

Race Finishers!

Male Finishers

Female Finishers

All Finishers

Add New Finisher

Male Finishers

- Name: Jacob Walker. Time: 25:54
- Name: Frank Jones. Time: 26:08
- Name: Bob Hope. Time: 26:38
- Name: Ryan Rice. Time: 28:24
- Name: Justin Jones. Time: 29:14
- Name: Samkeun Kim. Time: 37:30

Congratulations to all our finishers!

[Start Page Updates](#)[Stop Page Updates](#)

Page refreshes every 10 second(s).

Last Updated:

July 3, 2019, 4:43:36 pm

Q & A

