HTML CSS Javascript PHP MySQL To Build A Website – Simple Example

By W.S. Toh / Tips & Tutorials - HTML & CSS, Tips & Tutorials - Javascript, Tips & Tutorials - PHP / December 1, 2020

Welcome to a beginner's tutorial and simple example on how to use HTML, CSS, Javascript, PHP, and MySQL together to build a website. Just started with web development, and wondering how we can "put" all the different technologies together?

> **Various technologies and languages work together to create a website, each of them handling a different aspect.**

- **HTML deals with the structure and layout of web pages.**
- **CSS to handle the visuals and cosmetics.**
- **Javascript to add dynamic elements to the webpage.**
- **MySQL database to store data permanently.**
- **PHP to do server-side processing and transactions.**

Yep, some of you guys may already know that. So let us walk through the development of a simple product page in this guide – Read on!

ⓘ I have included a zip file with all the source code at the start of this tutorial, so you don't have to copy-paste everything… Or if you just want to dive straight in.

TABLE OF CONTENTS

DOWNLOAD & NOTES

Firstly, here is the download link to the example code as promised.

EXAMPLE CODE DOWNLOAD

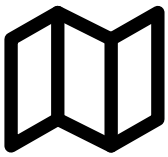
Click here to download the source code, I have released it under the MIT license, so feel free to build on top of it or use it in your own project.

QUICK NOTES

- Create a dummy database and import `1-products.sql`.
- Update `2-products.php` and change the database settings to your own.
- Launch `3-html-page.php` in your web browser and follow along.

If you spot a bug, please feel free to comment below. I try to answer questions too, but it is one person versus the entire world… If you need answers urgently, please check out my list of websites to get help with programming.

EXAMPLE DEVELOPMENT CYCLE



All right, let us now get started with the development of the products page itself.

STEP 1) THE PLAN – DIVIDE AND CONQUER

It is common for beginners to get paralyzed by “too many things to deal with” and “don’t know where to start”. So the first step in every project is to break things down and lay out a step-by-step plan. So in this example of creating a simple product page:

- **Goal & Overview**: As Captain Obvious as this may be, we need to create a page to display products from the database.
- **Server-Side Database (MySQL):** To keep things simple, the database will only capture the product name and description.
- **Server-Side PHP**: Create a script that gets product information from the database.

- **Client-Side HTML & CSS**: Use the PHP script to create an HTML page to show the products, CSS for the styling.
- **Client-Side Javascript:** Finally, use Javascript to add dynamic elements to the HTML page. For example, what happens when the user clicks on a product.

STEP 2) CREATING MYSQL DATABASE TABLES

THE SQL

1-products.sql

```
CREATE TABLE `products` (
  `product_id` int(11) NOT NULL,
  `product_name` varchar(128) NOT NULL,
  `product_description` text
) ENGINE=InnoDB DEFAULT CHARSET=latin1;

INSERT INTO `products` (`product_id`, `product_name`, `product_descripti
(1, 'Vulture Of Fortune', 'A sweated bump consents across the here separ
(2, 'Guardian Without Duty', 'Does a table migrate inside an excessive p
(3, 'Enemies Without Hope', 'A cured parameter fears behind the phenomer
(4, 'Lords Of The Void', 'The diary scores around the generalized lie.')
(5, 'Doctors And Aliens', 'The diary scores around the generalized lie.'
(6, 'Blacksmiths And Criminals', 'A considerable snail works into a purc

ALTER TABLE `products`
  ADD PRIMARY KEY (`product_id`),
  ADD KEY `product_name` (`product_name`);

ALTER TABLE `products`
  MODIFY `product_id` int(11) NOT NULL AUTO_INCREMENT, AUTO_INCREMENT=7;
```

TABLE STRUCTURE

Field	Description

product_id	Product ID. The primary key, auto-increment.
product_name	The product name. Indexed for search performance.
product_description	The product description.

MYSQL DATABASE NOTES & EXPLANATION

Personally, I will always start with the database, simply because it serves as the foundation of almost every project. It is important to get the required fields correct right from the start, or changing the database structure later can be very painful later.

STEP 3) SERVER-SIDE PHP SCRIPTS

THE SCRIPT

2-products.php

```
<?php
// (A) CONNECT TO DATABASE
// ! CHANGE SETTINGS TO YOUR OWN !
define('DB_HOST', 'localhost');
define('DB_NAME', 'test');
define('DB_CHARSET', 'utf8');
define('DB_USER', 'root');
define('DB_PASSWORD', '');
try {
  $pdo = new PDO(
    "mysql:host=" . DB_HOST . ";charset=" . DB_CHARSET . ";dbname=" . DE
    DB_USER, DB_PASSWORD, [PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION]
  );
} catch (Exception $ex) {
  die($ex->getMessage());
}
```

```
// (B) GET PRODUCTS
$stmt = $pdo->prepare("SELECT * FROM `products`");
$stmt->execute();
$products = $stmt->fetchAll(PDO::FETCH_NAMED);
```

PHP NOTES & EXPLANATION

PHP is the next piece of the puzzle and foundation, the above is an over-simplified example of how we can use PHP to fetch data from the database… This is actually kind of bad, but remember that in actual projects, this would have been made into a library or put into a `class`, object-oriented style.

STEP 4) HTML PAGE

THE SCRIPT

3-html-page.php

```
<!DOCTYPE html>
<html>
  <head>
    <title>
      Products Demo Page
    </title>

    <!-- (A) CSS & JS -->
    <link href="4-style.css" rel="stylesheet">
    <script src="5-script.js"></script>
  </head>
  <body>
    <!-- (B) BOOKS LIST -->
    <h1>OUR BOOKS</h1>
    <div id="our-books">
      <?php
      require "2-products.php";
      foreach ($products as $p) { ?>
      <div class="book-wrap" data-id="<?=$p['product_id']?>">
        <div class="book-title"><?=$p['product_name']?></div>
```

```
      <div class="book-desc"><?=$p['product_description']?></div>
    </div>
    <?php } ?>
  </div>
 </body>
</html>
```

HTML NOTES & EXPLANATION

So what is happening here!? Why is the file extension PHP instead of HTML? That's right – We are using PHP alongside HTML here. No need to be confused. Remember the script we wrote earlier to fetch product data from the database? We are simply using it on this page to generate the product's HTML, that's all.

STEP 5) THE CSS

THE SCRIPT

4-style.css

```
/* (A) WHOLE PAGE */
html, body {
  font-family: arial, sans-serif;
}

/* (B) BOOKS */
#our-books {
  max-width: 1200px;
  margin: 0 auto;
  display: grid;
  grid-template-columns: auto auto;
  grid-gap: 10px;
}
#our-books .book-wrap {
  background: #f1f1f1;
  padding: 15px;
  border: 1px solid #808080;
}
#our-books .book-wrap:hover {
  cursor: pointer;
```

```
  background: #fdffe6;
}
#our-books .book-title {
  font-weight: bold;
  font-size: 1.2em;
  color: #333;
}
#our-books .book-desc {
  margin-top: 10px;
  color: #848484;
}
```

CSS NOTES & EXPLANATION

Nothing much here actually. CSS is the part that deals with cosmetics and styles in HTML.

STEP 6) THE JAVASCRIPT

THE SCRIPT

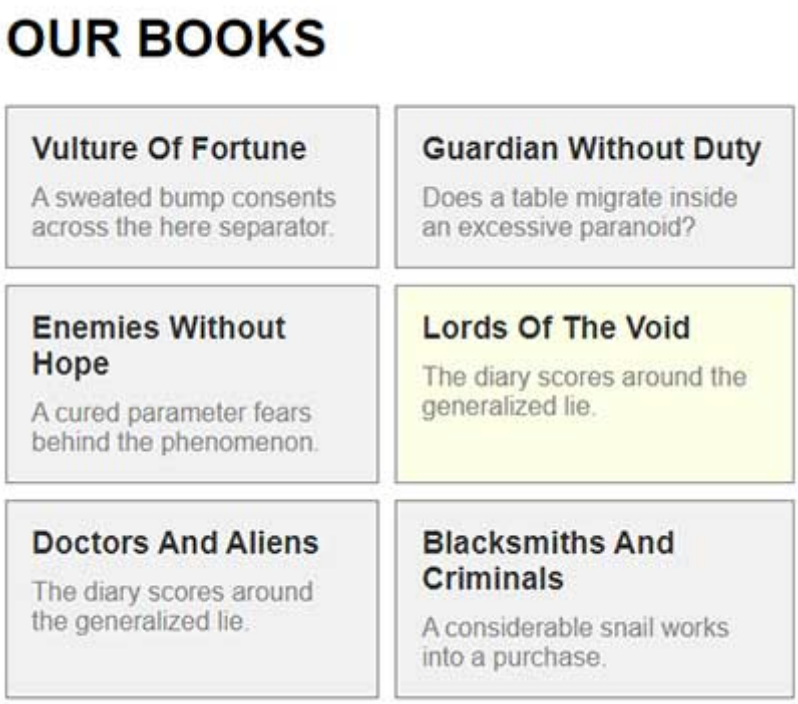
5-script.js

```
window.addEventListener("load", function(){
  for (let book of document.getElementsByClassName("book-wrap")) {
    book.addEventListener("click", function(){
      var id = this.dataset.id,
          name = this.getElementsByClassName("book-title")[0].innerHTML,
          desc = this.getElementsByClassName("book-desc")[0].innerHTML;
      alert(`You have selected - ID: ${id}, TITLE: ${name} DESC: ${desc}
    });
  }
});
```

JAVASCRIPT NOTES & EXPLANATION

All right, another confession to make, this is overly simplified. On a “normal” products page, Javascript would have been used to send an “add to cart” process to the server when an item is clicked. But you get the drift, Javascript is used to do everything dynamic – Add to cart, remove from cart, submit forms, checks, load more content, change themes, etc…

THE RESULT



Yep, this is not the prettiest looking award-winning website on the planet. But it should do well enough to demonstrate how we can use all the PHP, MySQL, HTML, CSS, and Javascript together.

USEFUL BITS & LINKS

That's it for all the code, and here are a few small extras that may be useful.

SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

Congratulations, what you have gone through in the last 15 minutes is called a "software development life cycle" (SDLC).

- Requirement Analysis – What is required? What does the client need?
- Planning – Divide-and-conquer. What needs to be done?
- Design – The database structure, software architecture.
- Development – Actual coding.
- Testing & Deployment – Test the code, put it live on the Internet.

Some people might have different steps for the SDLC, but that is the general process. Also, take note that SDLC does not stop after just one cycle – We might review our projects later, and decide that things need to change. The whole SDLC cycle starts again, and it might never really end.

FRONT-END, BACK-END, FULL-STACK

As a beginner, you need not worry too much actually. In real-world massive projects, there are usually many developers working on the same project, each handling a different component:

- Project Manager (Full-Stack) – The boss. He/she/it deals with the wide overview, processes, structure of the entire system, and even support for

other platforms such as mobile apps.

- Lead/Senior Web Developers (Full-Stack) – Does everything. HTML, CSS, Javascript, PHP, MySQL, and whatever else. AKA Superman.
- Junior Web Developers (Front End) – Creates the webpages. HTML, CSS, Javascript.
- Junior Web Developers (Back End) – Does PHP, MySQL. More on the administration panel and server-side libraries instead.

So yeah, it is only further up the ladder that we need to really deal with “everything and beyond”.

INFOGRAPHIC CHEAT SHEET

How HTML, CSS, PHP, Javascript, MySQL Work Together (Click to enlarge)

LINKS & REFERENCES

- What Is SDLC? – Stackify

THE END

Thank you for reading, and we have come to the end of this guide. I hope that it has helped you to better understand – HTML, CSS, Javascript, PHP, MySQL, they are all different components, but they all work together to create a single website or system.

It may be overwhelming at first to deal with many things at once, so take it step-by-step, and the rest will be just like learning how to cycle. Once you catch the gist and flow, it will be much easier. If you want to share anything with this guide, please feel free to comment below. Good luck and happy coding!

← Previous Post Next Post →

1 thought on “HTML CSS Javascript PHP MySQL To Build A Website – Simple Example”

MANNY
JANUARY 30, 2020 AT 1:40 AM

Looks like an great opportunity to learn and understand how all these languages are integrated to make web site.

Thank you.

Manny

Reply

Leave a Comment

Your email address will not be published. Required fields are marked *

Type here..

Name*

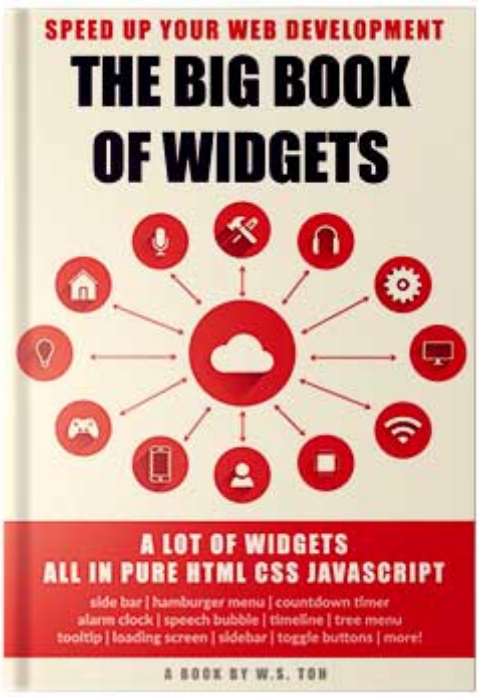
Email*

Post Comment »

ABOUT ME

W.S. Toh is a senior web developer and SEO practitioner with over 15 years of experience. Graduated from the University of London. When not secretly being an evil tech ninja, he enjoys photography and working on DIY projects.

BIG BOOK OF WIDGETS



The Big Book of Widgets is a collection of many HTML CSS JS widgets. It will help to save time and speed up development - Check it out!

FOLLOW US

Pinterest

YouTube
