提出日：2024年4月26日

先進情報プロジェクト実習

テーマIT・1 第2回レポート

学籍番号：C0A22113

氏名：成田彩華

# 設定ファイルの確認

設定ファイルの確認を行なっていく。以下のコマンドを入力する。

$ sudo apache2ctl configtest

以下のようなエラーが出るかを確認する。出なければ設定ができている。

AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set the 'ServerName' directive globally to suppress this message

Syntax OK

このようなエラーが出た場合には、以下のコマンドを入力し、設定ファイルをroot権限で作成する。

sudo touch /etc/apache2/conf-available/fqdn.conf

このファイルの内容は次の通りである。

ServerName myserver

以下のコマンドで、この設定を有効にしてApache2を再起動する。

$ sudo a2enconf fqdn

$ sudo systemctl reload apache2

先ほどのコマンドでエラーが出ないか再度確認する。

$ sudo apache2ctl configtest

Syntax OK

上記のようにSyntax OKのみ表示されれば問題なく設定できている。

# PHPのインストール

PHPのインストールを進めていく。以下のコマンドを入力し、インストールしていく。

$ sudo apt install php libapache2-mod-php

この操作後に追加で21.3 MB のディスク容量が消費されます。

続行しますか? [Y/n]

このように聞かれたらYを入力して続行する。するとPHPがインストールされる。

PHPのバージョンを確認し、インストールされているか確認する。

$ php -v

PHP 8.1.2-1ubuntu2.14 (cli) (built: Aug 18 2023 11:41:11) (NTS)

Copyright (c) The PHP Group

Zend Engine v4.1.2, Copyright (c) Zend Technologies

    with Zend OPcache v8.1.2-1ubuntu2.14, Copyright (c), by Zend Technologies

このようにバージョンが表示されているため、問題なくインストールがされている。

# PHPの動作確認

PHPの動作確認を行なっていく。/var/www/html/testに以下の内容のinfo.phpを作成する。

<?php

phpinfo();

?>

UbuntuのIPアドレスを確認する。

$ hostname -I

192. 168.64.7

ブラウザからUbuntuのWebサーバにあるinfo.phpにアクセスする。

<http://192.168.64.7/test/info.php>をウェブサーバーに入力するとアクセスできる。

「PHP Version 8.1.2-1ubuntu2.14」のように書かれている表が表示されれば、PHPは動作していることが確認できる。

接続できたことが確認できたため、セキュリティのため、パーミッションを変更しておく。Webサイトからアクセスするユーザはwww-dataであるため、所有者をwww-dataにしておく。また、このファイルは所有者が読み込みできれば良いため、以下のコマンドで、読み込み権限だけ与えておく設定に変えておく。

$ sudo chown www-data:www-data /var/www/html/test/info.php

$ sudo chmod 400 /var/www/html/test/info.php

再度、Webブラウザからinfo.phpにアクセスし、パーミッション変更前と同じ画面になれば、パーミッション変更が正常に行えている。

# SQLite3のインストール

SQLite3のインストールを行なっていく。次のコマンドでインストールする。

$ sudo apt install sqlite3

SQLiteの起動していく。以下のコマンドを入力して起動する。

$ sqlite3

SQLite version 3.37.2 2022-01-06 13:25:41

Enter ".help" for usage hints.

Connected to a transient in-memory database.

Use ".open FILENAME" to reopen on a persistent database.

cqlite>

このようにsqlite>というプロンプトに入力する状態になり、起動が完了する。この除隊でSQLiteが扱える。

ヘルプは以下のコマンドで表示する。

sqlite> .help

コマンドの説明が表示される。

使用するデータベースで設定されている文字コードについては、次のコマンドで確認できる。UTF-8になっているはずなので特に問題はない。

sqlite> PRAGMA encoding;

UTF-8

SQLiteを終了する時は次のコマンドで行う。

sqlite> .exit

または、

sqlite> .quit

CTRL + cでは終了できないので注意する。上記のコマンドを入力すると元のLinuxのターミナルのプロンプトに戻る。

# SQLite3のデータベース作成

データベースの作成を行なっていく。まずは、以下のコマンドでSQLiteを起動する。

$ sqlite

新しいデータベースファイルを作成する。次のコマンドでは、新規作成でデータベースをオープンしている。

sqlite> .open –new /var/www/html/test/test.db

２回目以降はデータベースを指定してオープンすれば良い。

sqlite> .open /var/www/html/test/test.db

ここでは、工科太郎などを登録したテーブルを作成してみる。

sqlite> create table students(stid INTEGER, name TEXT);

sqlite> insert into students values(12345678, ‘工科太郎’);

sqlite> insert into students values(23456789, ‘工科花子’);

sqliute> insert into students values(34567890, ‘工科次郎’);

1行目でテーブルを作成し、2,3,4行目でテーブルにデータを登録している。

登録されているデータを取得していく。

sqlite> select \* from students;

このように取得できる。

12345678|工科太郎

23456789|工科花子

34567890|工科次郎

登録されているデータをアップデートしていく。stidの値を使いデータを指定する。

sqlite> update students set name=”工科二郎” where stid=”34567890”;

表示してみると、指定されたデータがアップデートされている。

sqlite> select \* from students;

12345678|工科太郎

23456789|工科花子

34567890|工科二郎

データを削除する。stidの値を使いデータを指定する。

sqlite> delete from students where stid=”34567890”;

表示してみると、指定されたデータが削除されている。

sqlite> select \* from students;

12345678|工科太郎

23456789|工科花子

接続中のデータベースを確認する。以下のコマンドを入力することで確認できる。

sqlite> .databases

main: /var/www/html/test/test.db r/w

テーブルの一覧を表示する。以下のコマンドを入力し、確認する。

sqlite> .tables

students

# PHPからSQLiteへ接続してデータベースを利用する

php.iniの設定を行う。「/etc/php/8.1/apache2/php.ini」の「;extension=pdo\_sqlite」となっている部分のセミコロンを削除する。

$ gedit /etc/php/8.1/apache2/php.ini

936行目あたりが次のようになっている

;extension=pdo\_mysql

;extension=pdo\_oci

;extension=pdo\_odbc

;extension=pdo\_pgsql

;extension=pdo\_sqlite

;extension=pgsql

;extension=shmop

940行目のセミコロンを削除し、次のようにする

;extension=pdo\_mysql

;extension=pdo\_oci

;extension=pdo\_odbc

;extension=pdo\_pgsql

extension=pdo\_sqlite

;extension=pgsql

;extension=shmop

さらに、「/etc/php/8.1/apache2/php.ini」の「;extension=sqlite3」となっている部分のセミコロンを削除する。948行目あたりが次のようになっている。

;extension=soap

;extension=sockets

;extension=sodium

;extension=sqlite3

;extension=tidy

;extension=xsl

951行目のセミコロンを削除し、次のようにする。

;extension=soap

;extension=sockets

;extension=sodium

extension=sqlite3

;extension=tidy

;extension=xsl

設定を変更したら、以下のコマンドを入力しApache2を再起動する。

$ sudo systemctl reload apache2

複数あるphp.iniはどれが現在の使用されている設定ファイルなのかを確認していく。PHPの動作確認でアクセスした<http://192.168.64.7/test/info.php> の「Loaded Configuration File」に「/etc/php/8.1/apache2/php.ini」と書いてある。よって、現在使用されているphp.iniは/etc/php/8.1/apache2/php.iniであることがわかる。

php-sqlite3をインストールする。以下のコマンドを入力し、インストールする。

$ sudo apt install php-sqlite3

この操作後に追加で 154 kB のディスク容量が消費されます。

続行しますか? [Y/n]

このように聞かれたらYを入力し、続行する。すると、php-sqlite3がインストールされる。

以下のコマンドを入力し、Apache2を再起動させ、インストールを完了させる。

$ sudo systemctl reload apache2

# データベースを使用するPHPのサンプルを動かしてみる

授業ページにあるdbtest.php動かしていく。先に作成した「/var/www/html/test/test.db」を使用するため、このデータベースファイルと同じディレクトリにdbtest.phpを設置する。よってdbtest.phpのパスは「/var/www/html/test/dbtest.php」となる。

設置が完了したのち、Webブラウザからdbtest.phpにアクセスする。以下のように表示されていれば、正常に動作していることが確認できる。

array(4) { [0]=> int(12345678) ["stid"]=> int(12345678) [1]=> string(12) "工科太郎" ["name"]=> string(12) "工科太郎" }

array(4) { [0]=> int(23456789) ["stid"]=> int(23456789) [1]=> string(12) "工科花子" ["name"]=> string(12) "工科花子" }

セキュリティの観点から、パーミッションを変更する。所有者をwww-dataにし、このphpファイルは書き込まれると、正常に動作しない可能性があるため、所有者にのみ読み込み権限を与える。グループ、その他には権限を与えない。以下のコマンドで変更していく。

$ sudo chown www-data:www-data /var/www/html/test/dbtest.php

$ sudo chmod 400 /var/www/html/test/dbtest.php

変更が完了したら再度dbtest.phpにアクセスする。変更前と同じように表示されれば正常に変更が完了していることがわかる。

# PDOを使用してデータベースを使用するPHPのサンプルを動かしてみる

PDO(PHP Data Object)はPHPのデータ接続クラス。PDOは同じ命令で複数のデータベースに接続ができるようになるデータ接続クラスである。なお、PDOはPHPの標準機能であるため、追加でインストールする必要はない。

マルチバイト文字を使用できるようにするため、php-mbstringをインストールする。以下のコマンドを入力し、インストールする。

$ sudo apt install php-mbstring

この操作後に追加で 1,732 kB のディスク容量が消費されます。

続行しますか? [Y/n]

このように聞かれたらYを入力し、続行する。するとphp-mbstringがインストールされる。

授業ページにあるdbtest2.phpを動かしていく。先に作成した「/var/www/html/test/test.db」を使うので、このデータベースファイルと同じディレクトリにdbtest2.phpを設置する。よってdbtest2.phpのパスは、「/var/www/html/test/dbtest2.php」となる。設置が完了したのち、Webブラウザからdbtest2.phpにアクセスする。以下のように表示されていれば、正常に動作していることが確認できる。

12345678 工科太郎

所有者、グループ、パーミッションの設定を行う。Webサイトからアクセスするユーザはwww-dataであるため、所有者とグループはwww-dataにする。また、パーミッションは所有者のみが読み込み権限を持っていればよい。以下のコマンドで入力し、設定を適応させる。

$ sudo chown www-data:www-data /var/www/html/test/dbtest2.php

$ sudo chmod 400 /var/www/html/test/dbtest2.php

# PHPでデータベースに登録された項目を色々と表示する

先で触れたdbtest2.phpを修正し、「工科太郎」の名前だけ表示されるようにしてみる。

dbdata2.phpの8行目を次のように修正する。

$sql = “select name from students shere stid = :stid”;

任意の項目をforeachで表示する。これで「工科太郎」の名前のみ表示されるようになった。

次のようになるように表示する。

12345678

工科太郎

dbtest.phpの８行目、16行目、35行目、36行目を修正する。８行目を次のように修正する。

$sql = “select \* from students where stid = :stid”;

16行目を次のように修正する。

$rec = $stmt->fetchAll()

35行目、36行目を次のように修正する。

<?php foreach ($rec as list($stid\_item, $name\_item)):?>

<?php echo $stid\_item . “<br>” . $name\_item;?>

これで任意の項目に対し、idと名前が改行され表示されるようになった。

# PHPエラーが見えなくて困る場合

PHPがうまく動かない場合のエラー内容の確認方法について述べていく。先頭部分に次のように書いておくと、正常に動作した場合のみを除き、エラーがブラウザに表示されるようになる。

<?php

ini\_set(‘display\_errors’, 1);

ini\_set(‘error\_reporting’, E\_ALL);

# 加点項目

以下のように表示されるよう、データベースとそのデータを用いて表示するPHPファイルの作成を行う。

テキスト

中程度の精度で自動的に生成された説明

まずは、データベースの作成を行う。新たにtest.dbの中にtableを作成し、データの登録を行う。以下のコマンドを入力し、テーブルを作成する。

$ sqlite3

sqlite> .open /var/www/html/test/test.db

sqlite> create table animals(name TEXT, food TEXT);

sqlite> insert into animals values(‘猿’, ‘バナナ’);

sqlite> insert into animals values(‘馬’, ‘にんじん’);

sqlite> insert into animals values(‘アルパカ’, ‘タピオカミルクティー’);

sqlite> select \* from animals

猿|バナナ

馬|にんじん

アルパカ|タピオカミルクティー

これでテーブルの作成が完了した。

次に表示するためのPHPファイルであるtest2\_db.phpファイルを作成していく。授業サイトにある修正後のdbtest2.phpを参考にし、test2\_db.phpファイルを作成していく。以下の内容をtest2\_db.phpに書き込んでいく。

<?php

ini\_set('display\_errors', 1);

ini\_set('error\_reporting', E\_ALL);

$dsn = 'sqlite:/var/www/html/test/test.db';

try {

$pdo = new PDO($dsn);

$sql = "select \* from animals";

$stmt = $pdo->prepare($sql);

$stmt -> execute();

$rec = $stmt->fetchAll();

} catch(PODException $e) {

exit(mb\_convert\_encoding($e->getMessage(), 'UTF-8', 'SJIS-win'));

}

function printItem($str)

{

return htmlspecialchars($str, ENT\_QUOTES, 'UTF-8');

}

?>

<!DOCTYPE html>

<html lang="ja">

<head>

<meta charset="utf-8">

<title>animals</title>

</head>

<body>

<table border="1">

<?php foreach ($rec as list($name\_item, $food\_item)):?>

<tr>

<?php echo "<th>" . $name\_item . "</th> <td>" . $food\_item . "</td>";?>

</tr>

<?php endforeach; ?>

</table>

</body>

</html>

上記のように、<table>タグを用いて、表作成している。<table>タグの中のborder属性をborder=1で設定することで、二重線での格子で表示することができる。<tr>タグは行を表すため、データベースの1行を読み込み表示させる前後で囲う。<th>タグは、行名に当たるタグであるため、太字で表示されている。今回は動物名を太字にする指示であるため、name要素を<th>タグで囲う。<td>タグは列の要素にあたる。今回は指示を満たすため、food要素を<td>で囲う。

<http://192.168.64.7/test/test2_db.php> にアクセスし、動作確認を行う。指示通り表示されていれば問題ない。

所有者、グループ、パーミッション設定を行なっていく。Webブラウザからアクセスするユーザはwww-dataである。よって所有者、グループ共にwww-dataにする必要がある。以下のコマンドで所有者、グループを変更していく。

$ sudo chown www\_data:www-data /var/www/html/test/test2\_db.php

このtest2\_db.phpファイルは所有者にのみ読み込み権限を与えておけば良い。以下のコマンドで権限を変更していく。

$ sudo chmod 400 /var/www/html/test/test2\_db.php

再度、<http://192.168.64.7/test/test2_db.php> にアクセスし、先ほどと同じく、指示通り表示されていれば問題なく変更が行えている。