

# PHP

演習1 問合わせフォームの作成

**16** 時間目

HTML8時間目で学んだフォームは、画面上に表示するのみでした。

ここでは、実際にformに入力した内容が  
webサイトに管理者送信される仕組みを学びます。

# 演習内容

## お問い合わせフォーム

名前

メールアドレス

年齢

選択してください ▼

コメント

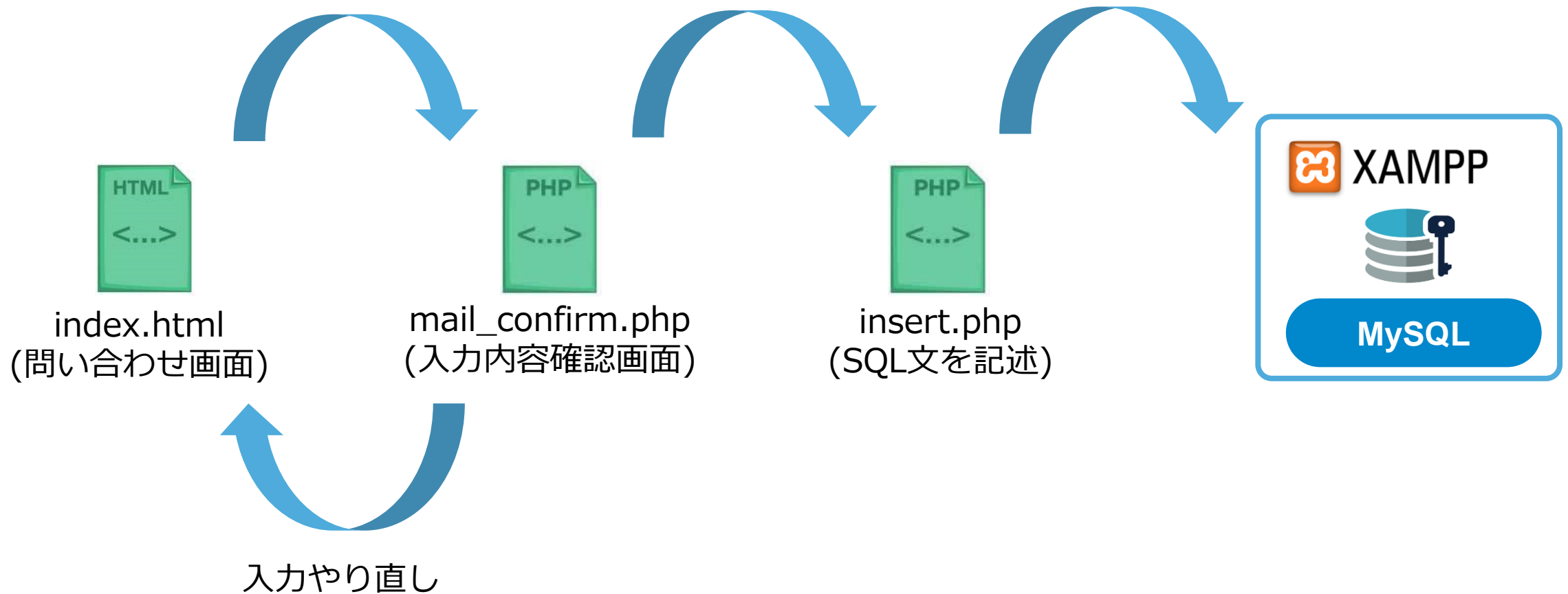
送信する

このformに必要な項目入力し、  
「送信する」をクリックすると、  
DBに“入力した項目”を格納できるようにする。

# 問い合わせフォームが動く仕組み

HTMLのフォームで入力された情報がPHPファイルへ引き渡される。

PHPファイルから、DBへ通信接続がされ、HTMLから引き渡された情報が、DBへ格納(SQLのinsert文) される。



# 問合わせフォーム作成の手順

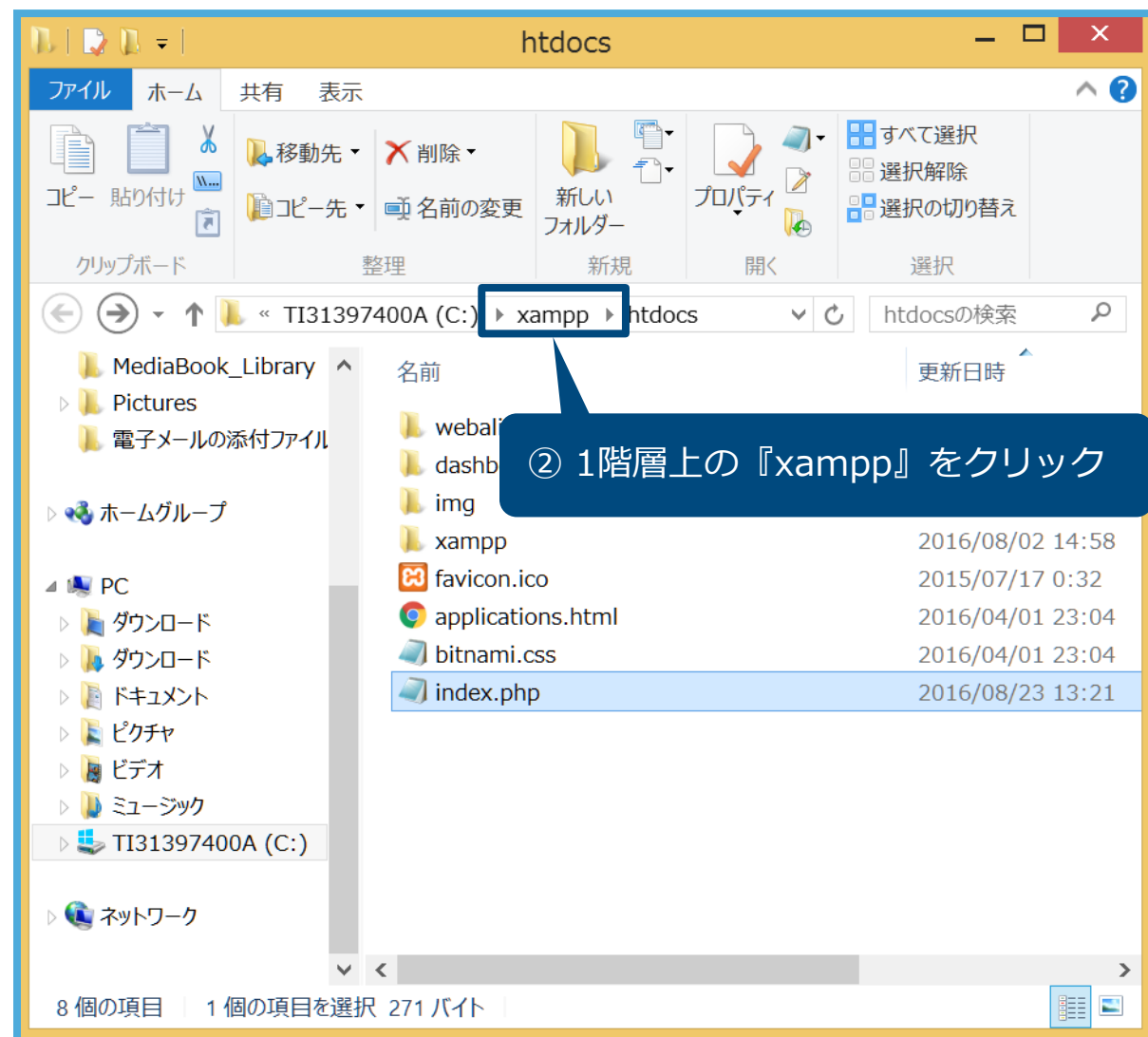
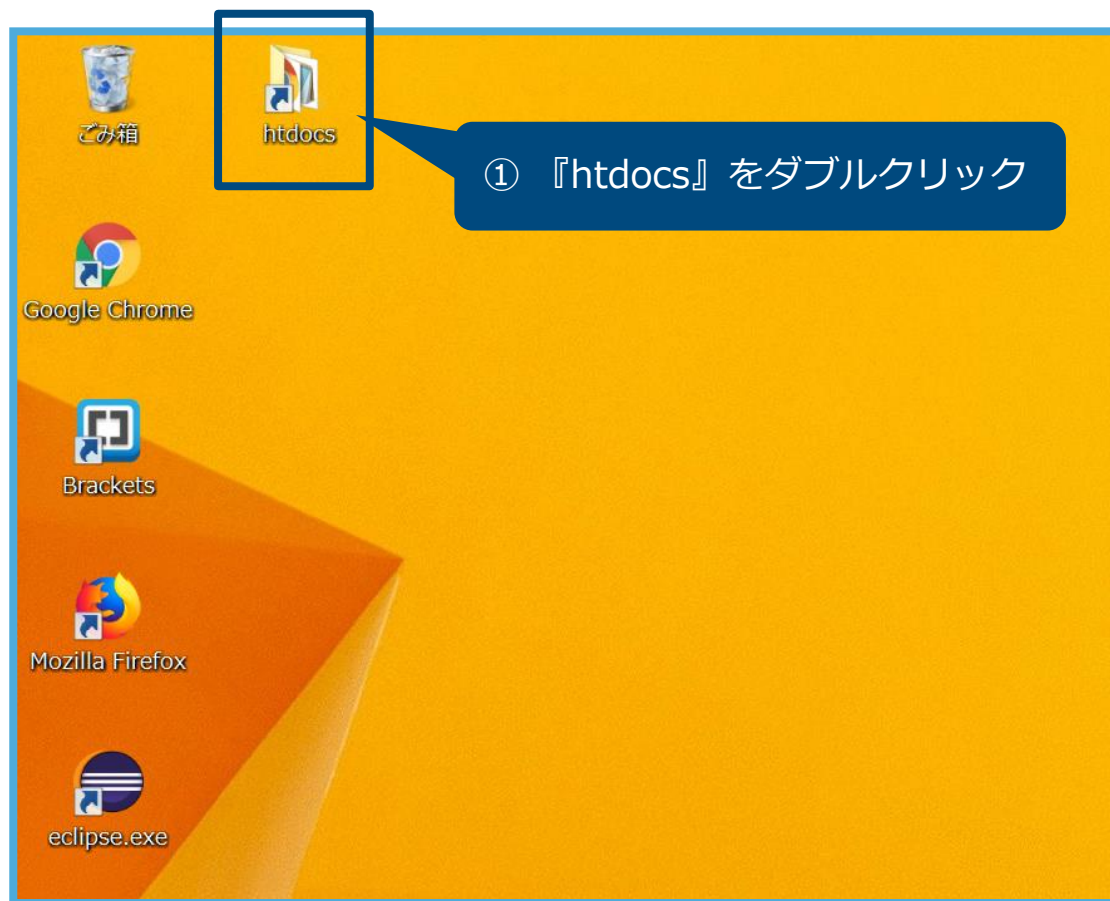
- 1 設定ファイルの編集
- 2 XAMPP (MySQL) 上にtableを作成する
- 3 HTMLとCSSを作成する
- 4 PHPファイルを作成する

# 問合わせフォーム作成の手順

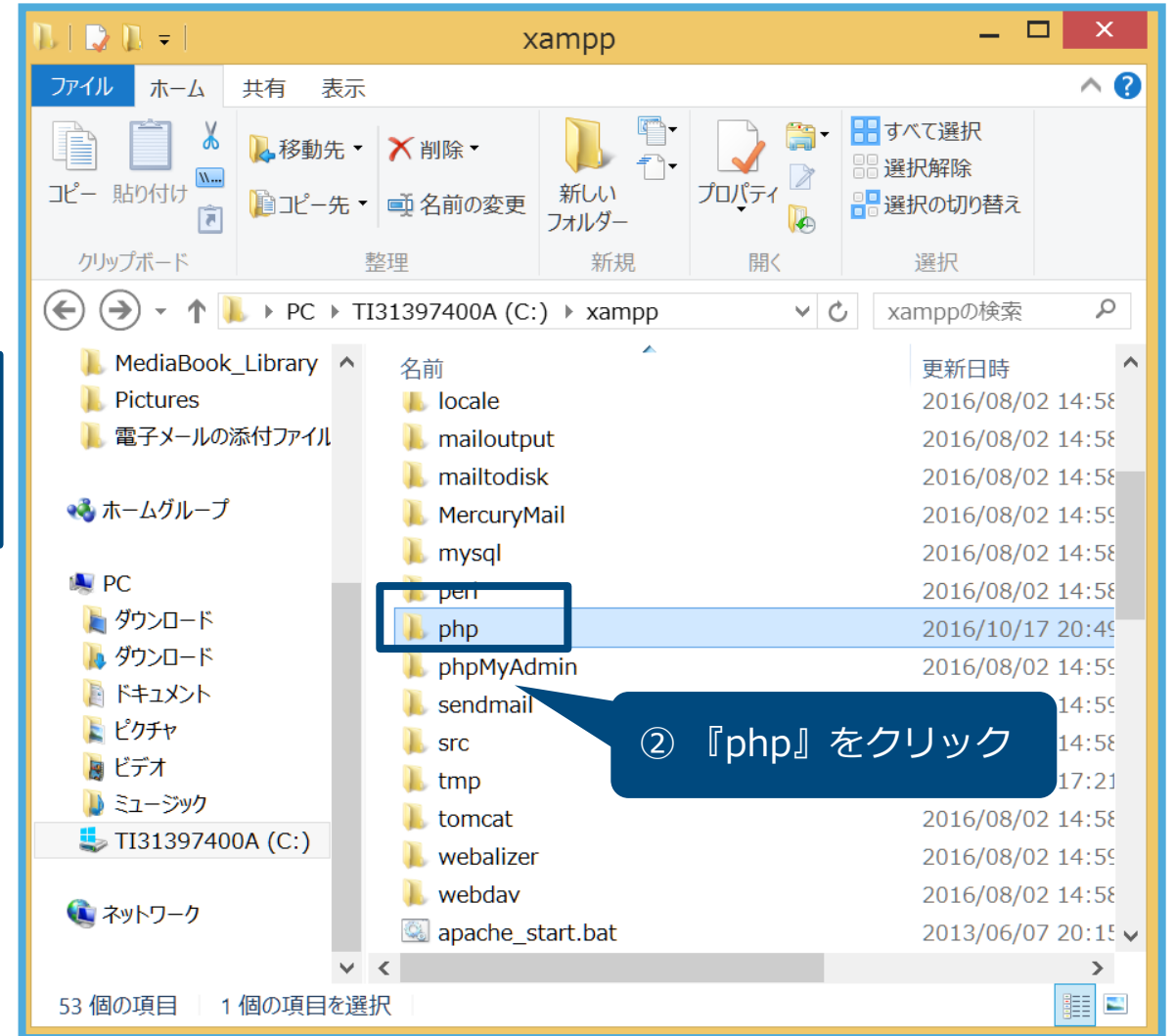
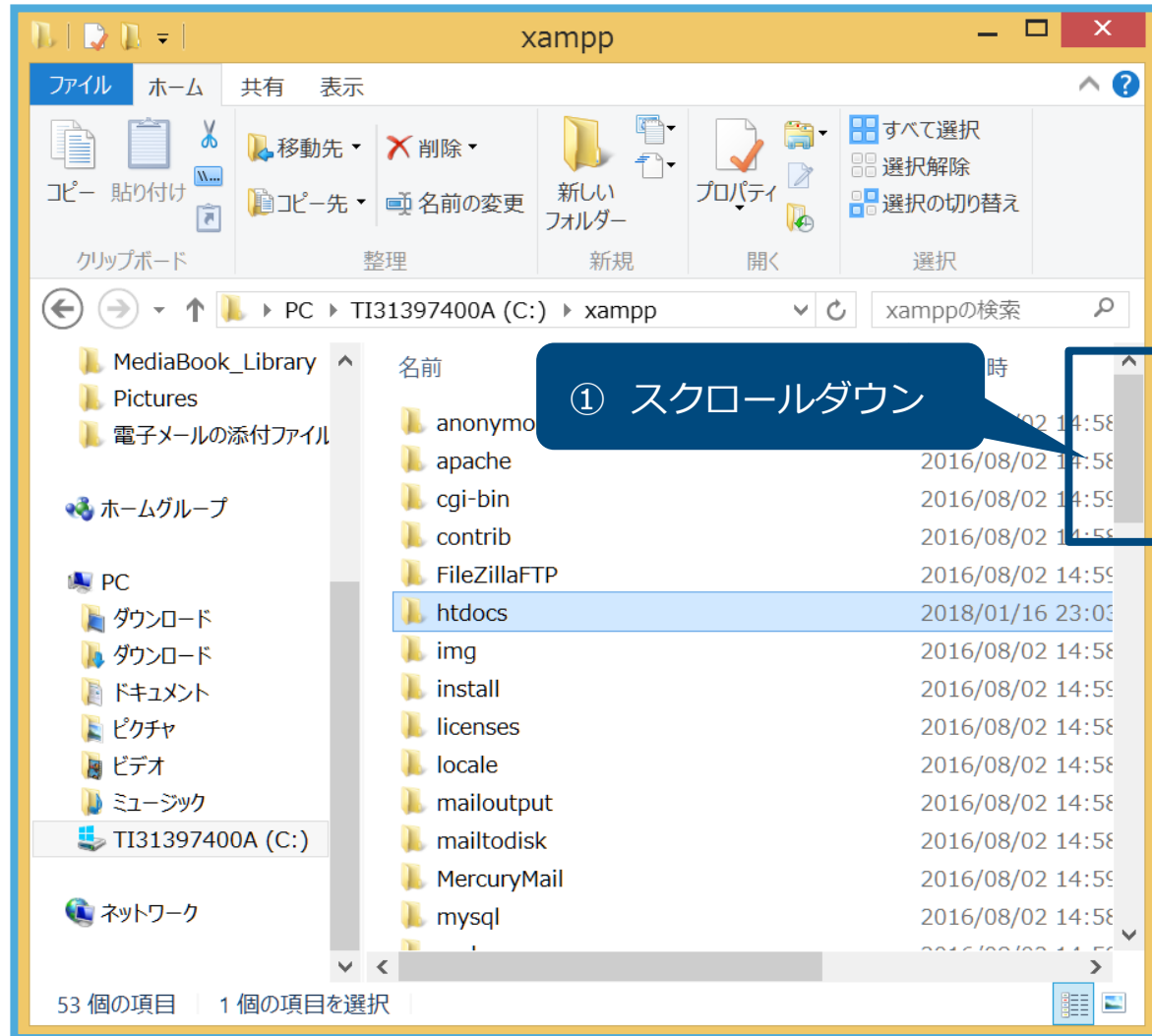
まず、設定ファイルの編集をします。

設定ファイルを編集しないと、  
フォームから送信した文字列が文字化けして  
MySQLに登録されてしまいます。

# 設定ファイルの編集

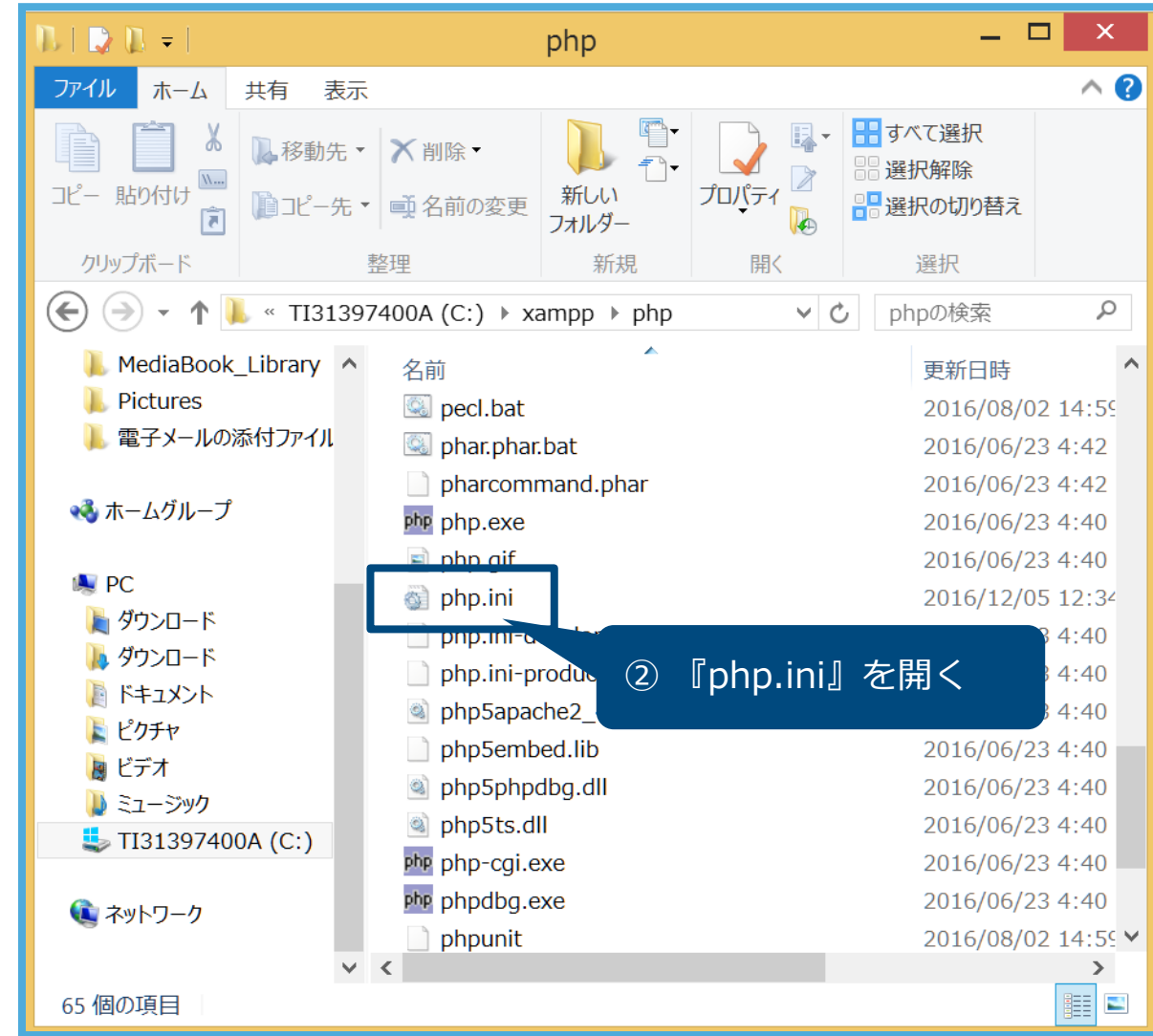
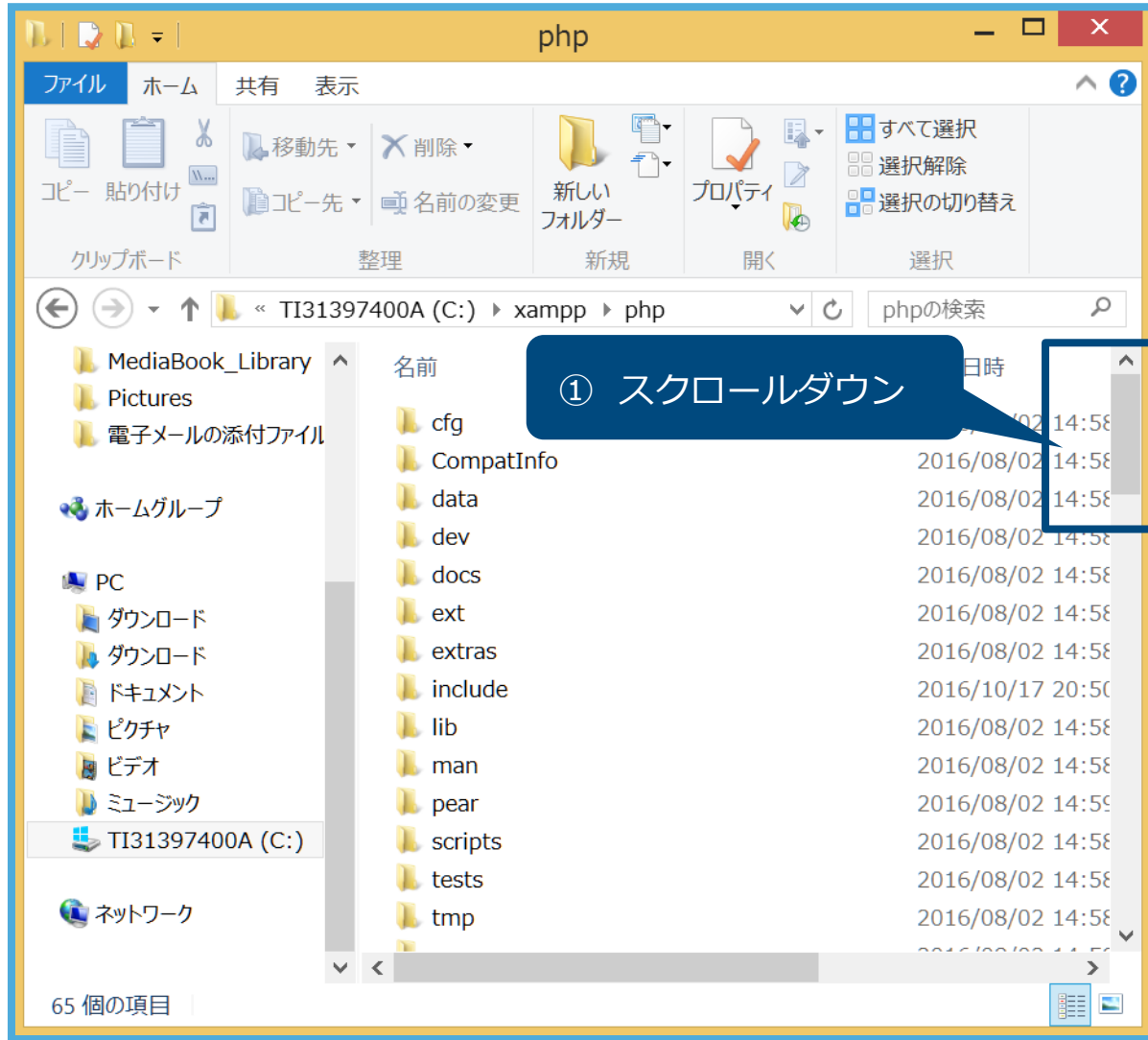


# 設定ファイルの編集

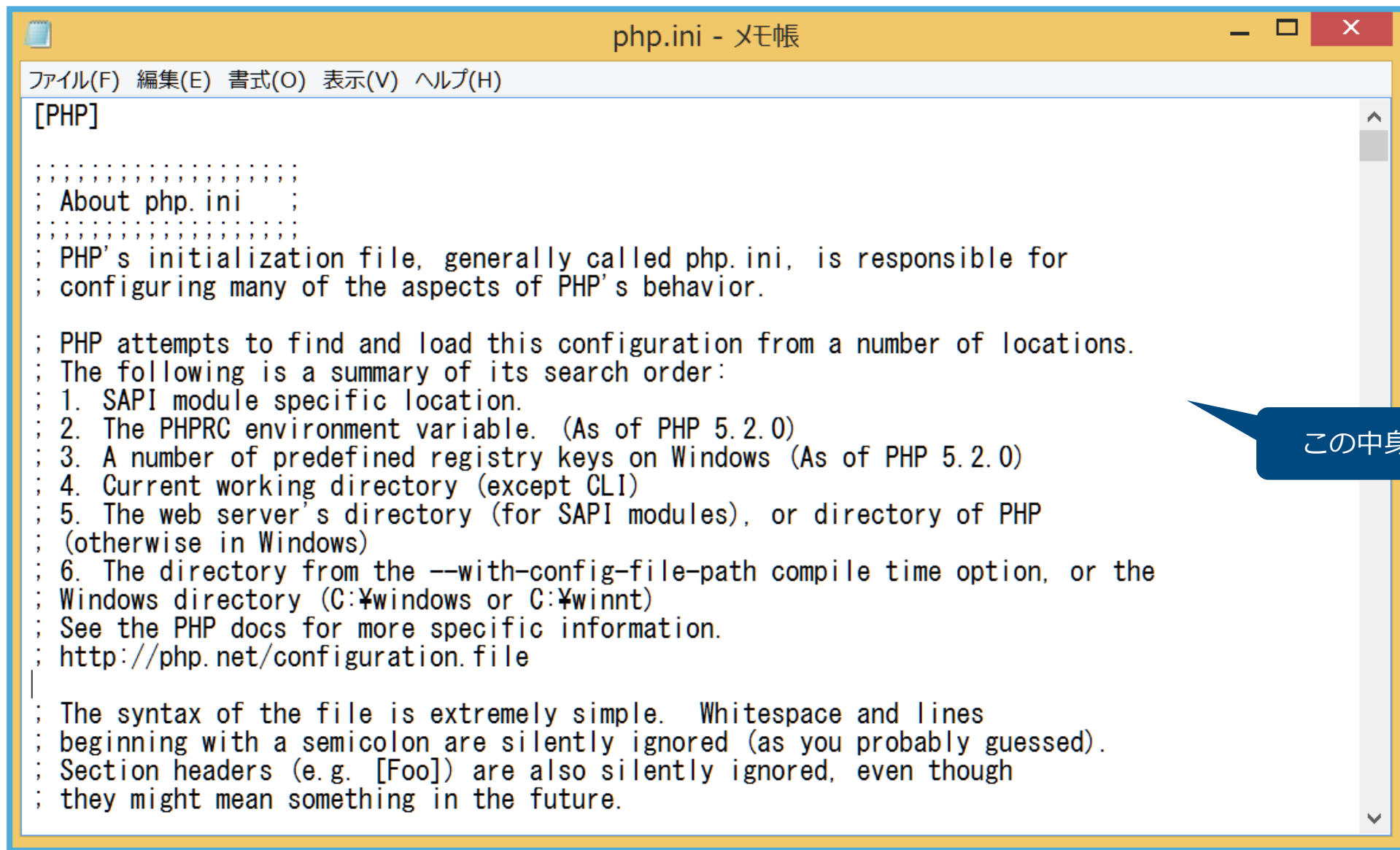




# 設定ファイルの編集



# 設定ファイルの編集



```
php.ini - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
[PHP]
;
; .....
; About php.ini
; .....
; PHP's initialization file, generally called php.ini, is responsible for
; configuring many of the aspects of PHP's behavior.
;
; PHP attempts to find and load this configuration from a number of locations.
; The following is a summary of its search order:
; 1. SAPI module specific location.
; 2. The PHPRC environment variable. (As of PHP 5.2.0)
; 3. A number of predefined registry keys on Windows (As of PHP 5.2.0)
; 4. Current working directory (except CLI)
; 5. The web server's directory (for SAPI modules), or directory of PHP
; (otherwise in Windows)
; 6. The directory from the --with-config-file-path compile time option, or the
; Windows directory (C:\windows or C:\winnt)
; See the PHP docs for more specific information.
; http://php.net/configuration.file
;
; The syntax of the file is extremely simple. Whitespace and lines
; beginning with a semicolon are silently ignored (as you probably guessed).
; Section headers (e.g. [Foo]) are also silently ignored, even though
; they might mean something in the future.
```

この中身を変更していく

# 設定ファイルの編集

```
php.ini - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
;com.autoregister_typelib = true

; register constants casesensitive
; http://php.net/com.autoregister-casesensitive
;com.autoregister_casesensitive = false

; show warnings on duplicate constant registrations
; http://php.net/com.autoregister-verbose
;com.autoregister_verbose = true

; The default character set code-page to use when passing strings to and from COM objects.
; Default: system ANSI code page
;com.code_page=

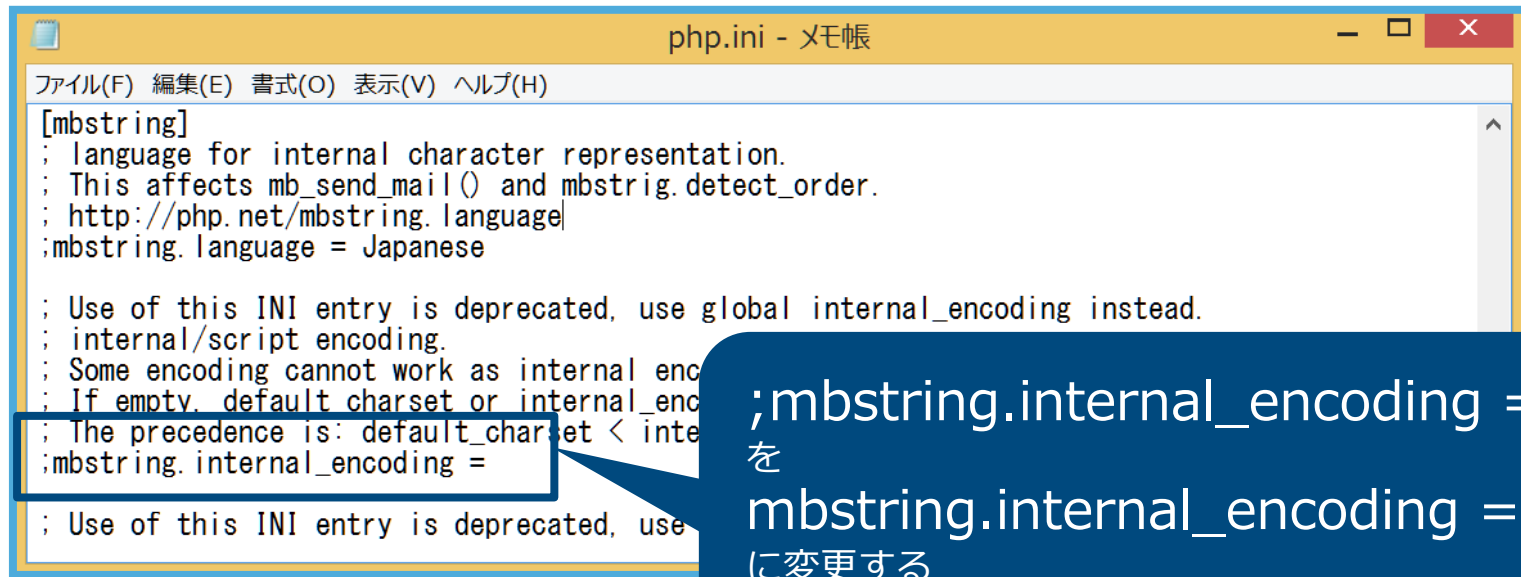
[mbstring]
; language for internal character representation
; This affects mb_send_mail() and mbstri
; http://php.net/mbstring.language
;mbstring.language = Japanese

; Use of this INI entry is deprecated, u
; internal/script encoding.
; Some encoding cannot work as internal
; If empty, default_charset or internal_encoding or iconv.internal_encoding is used.
; The precedence is: default_charset < internal_encoding < iconv.internal_encoding
```

前ページと同様に、php.iniから、下記を探し『;』を削除する

```
;mbstring.internal_encoding = EUC-JP  
;mbstring.http_input = auto  
;mbstring.http_output = SJIS  
;mbstring.encoding_translation = Off  
;mbstring.detect_order = auto  
;mbstring.substitute_character = none  
;mbstring.func_overload = 0
```

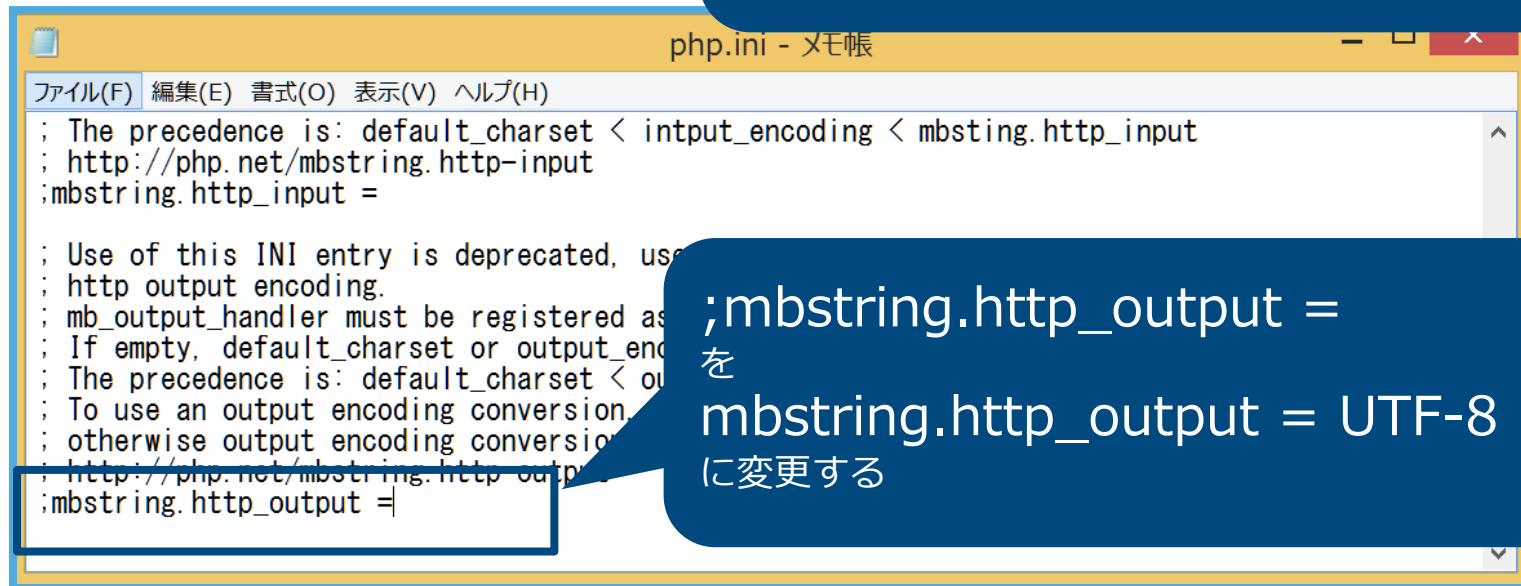
# 設定ファイルの編集



```
php.ini - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
[mbstring]
; language for internal character representation.
; This affects mb_send_mail() and mbstring.detect_order.
; http://php.net/mbstring.language
;mbstring.language = Japanese

; Use of this INI entry is deprecated, use global internal_encoding instead.
; internal/script encoding.
; Some encoding cannot work as internal encoding.
; If empty, default charset or internal encoding is used.
; The precedence is: default_charset < internal_encoding < mbstring.internal_encoding
;mbstring.internal_encoding =

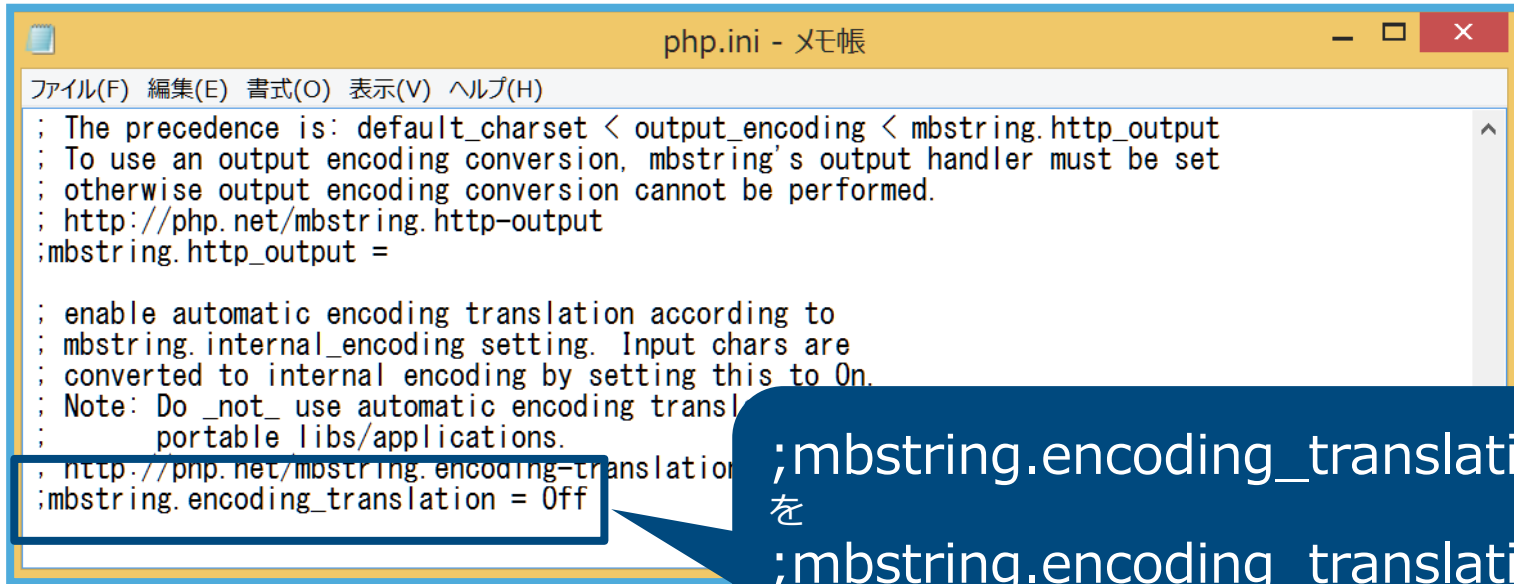
; Use of this INI entry is deprecated, use
```



```
php.ini - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
; The precedence is: default_charset < input_encoding < mbstring.http_input
; http://php.net/mbstring.http-input
;mbstring.http_input =

; Use of this INI entry is deprecated, use
; http output encoding.
; mb_output_handler must be registered as a normal function.
; If empty, default_charset or output_encoding is used.
; The precedence is: default_charset < output_encoding < mbstring.http_output
; To use an output encoding conversion, use mb_output_handler.
; otherwise output encoding conversion is not used.
; http://php.net/mbstring.http-output
;mbstring.http_output =
```

# 設定ファイルの編集



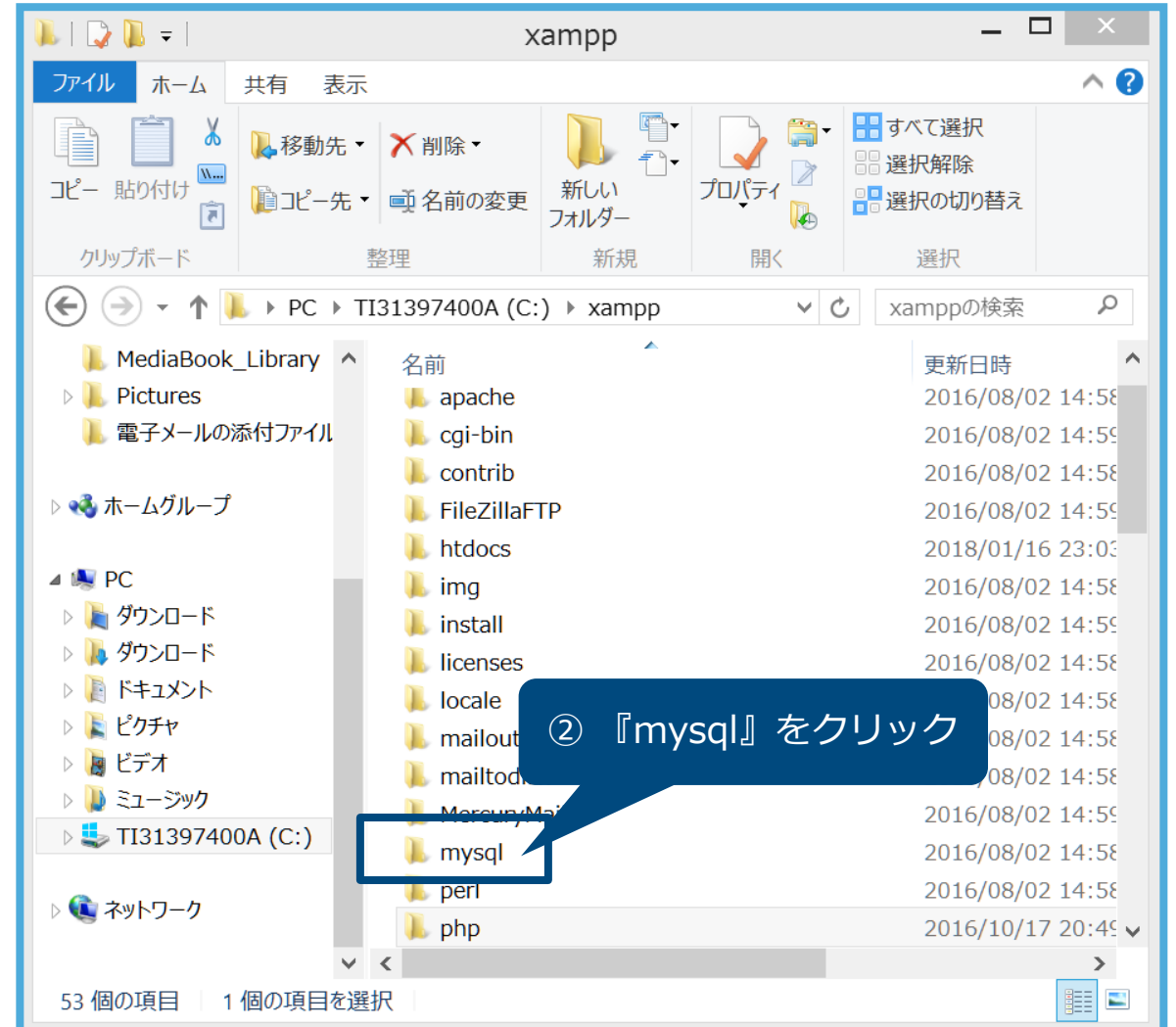
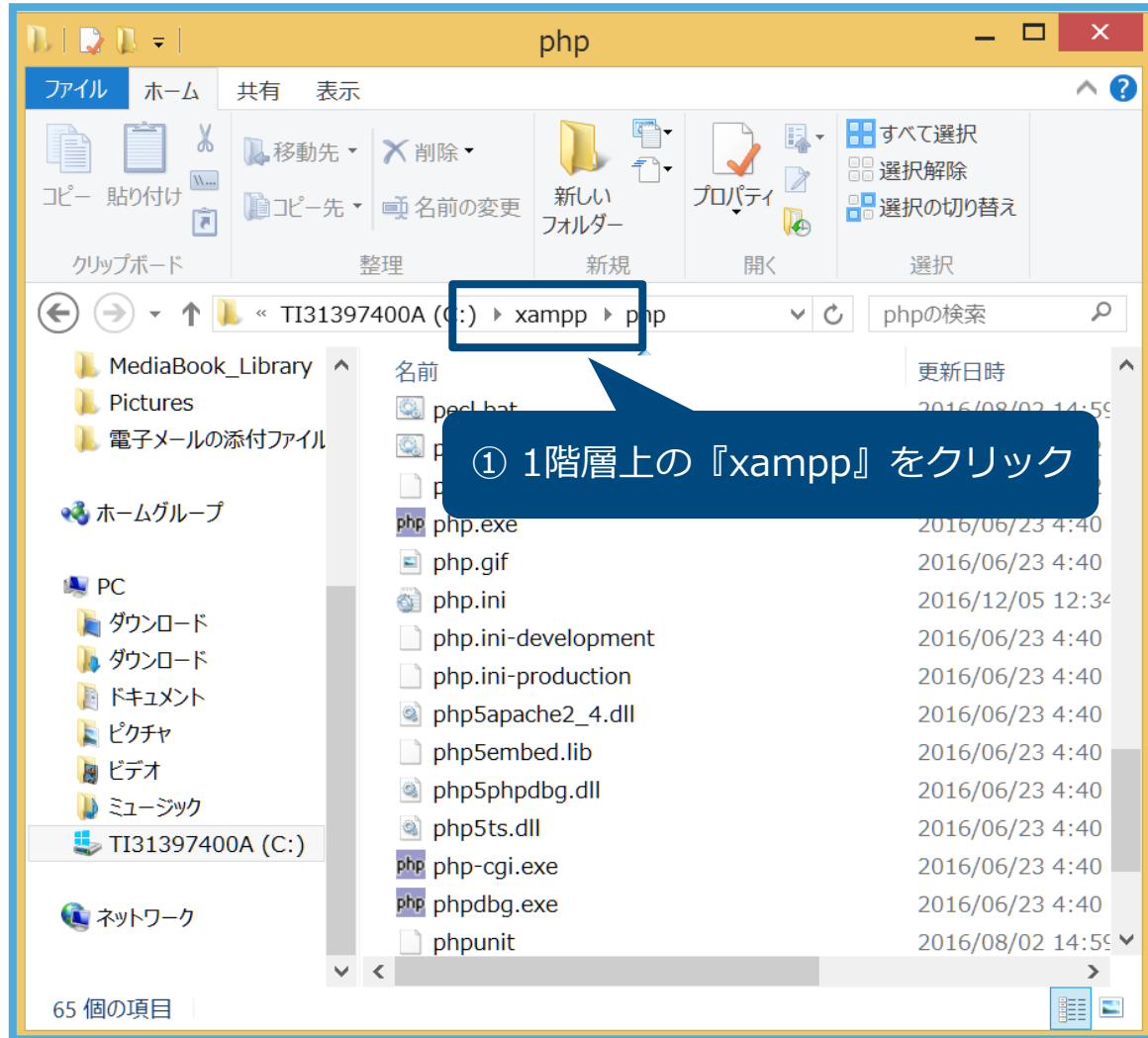
```
php.ini - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
; The precedence is: default_charset < output_encoding < mbstring.http_output
; To use an output encoding conversion, mbstring's output handler must be set
; otherwise output encoding conversion cannot be performed.
; http://php.net/mbstring.http-output
;mbstring.http_output =

; enable automatic encoding translation according to
; mbstring.internal_encoding setting. Input chars are
; converted to internal encoding by setting this to On.
; Note: Do _not_ use automatic encoding translation in
; portable libs/applications.
; http://php.net/mbstring.encoding-translation
;mbstring.encoding_translation = Off
```

;mbstring.encoding\_translation = off  
を  
;mbstring.encoding\_translation = on  
に変更する

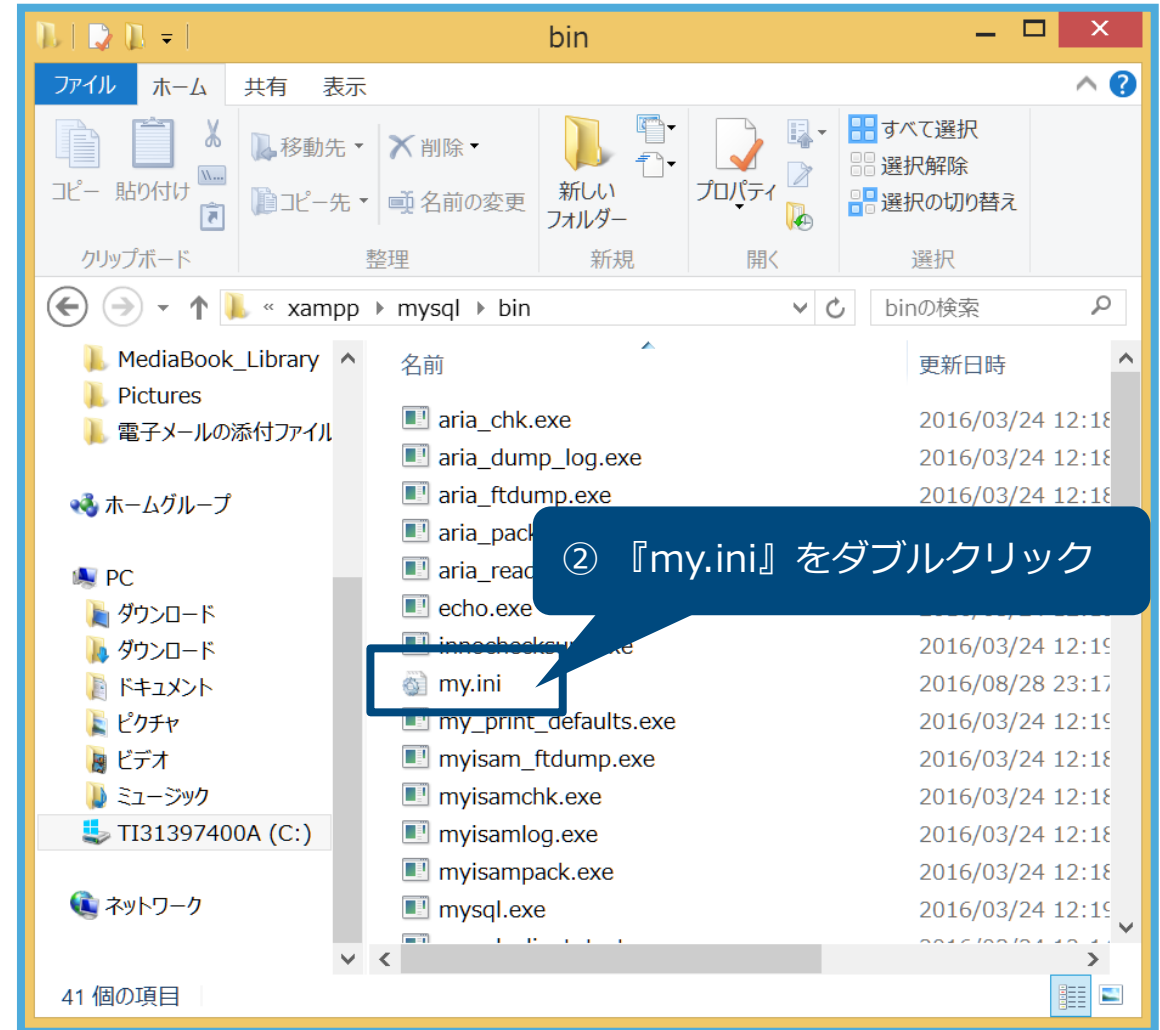
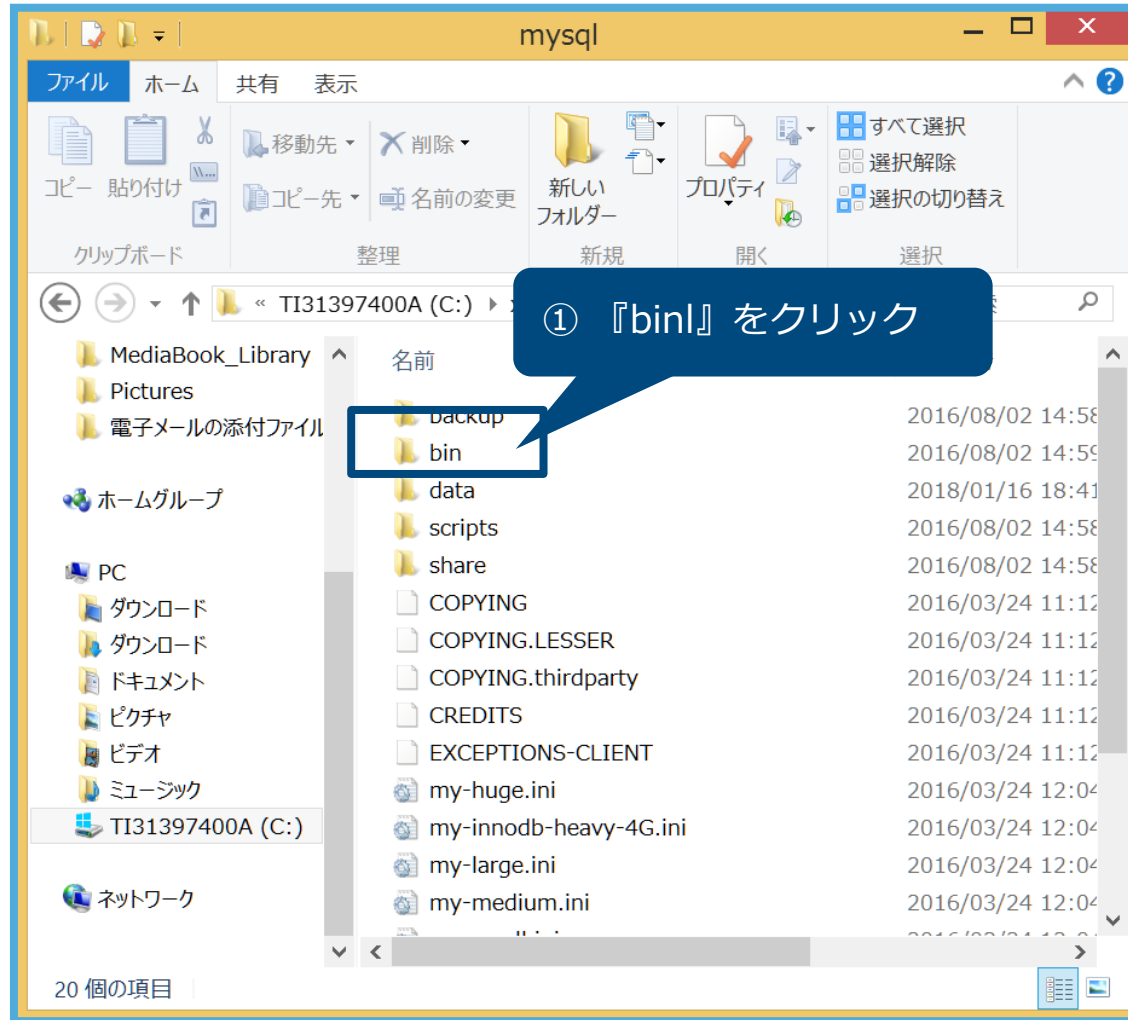
ここまで編集したら『php.ini』を保存

# 設定ファイルの編集



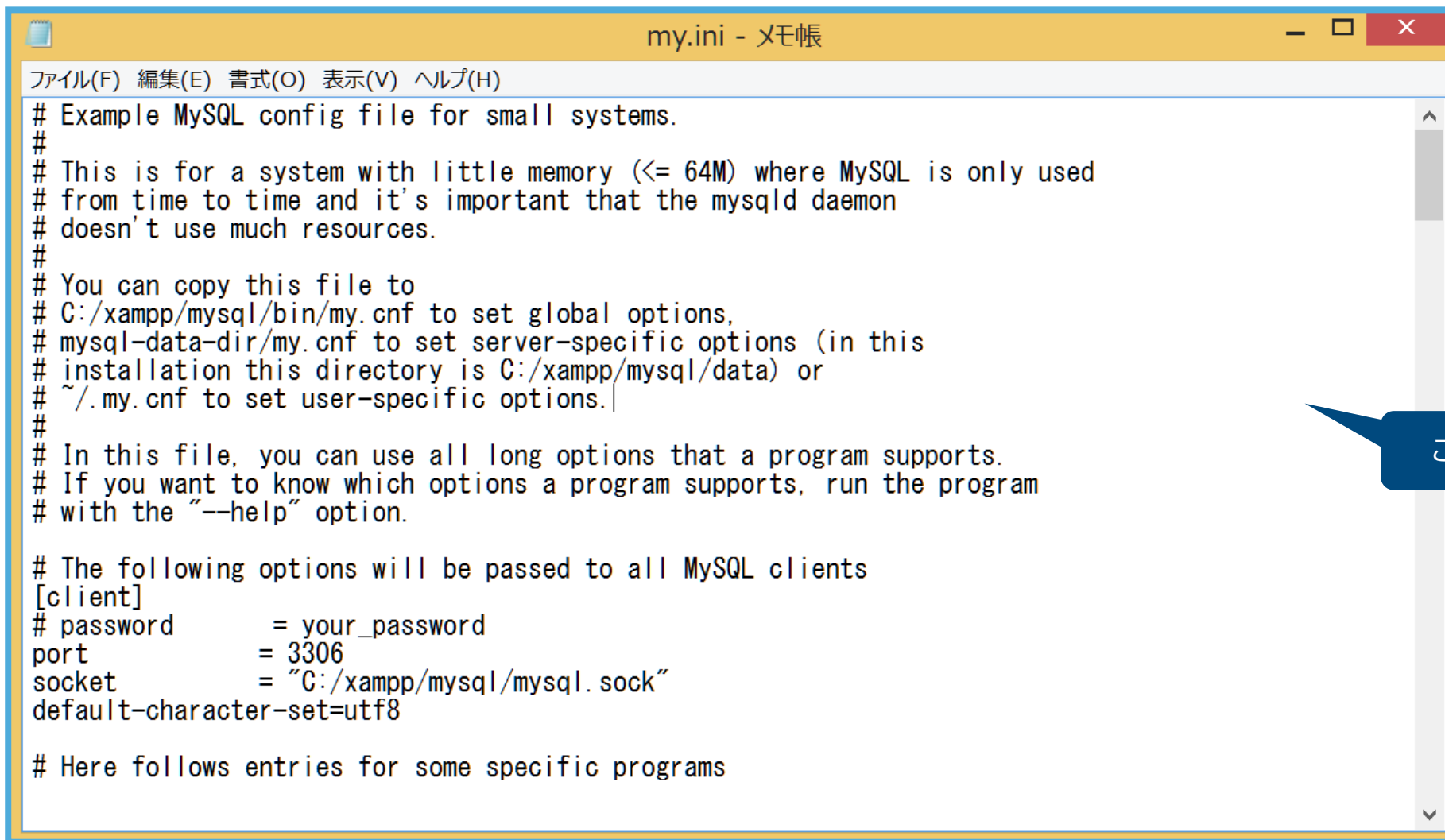


# 設定ファイルの編集





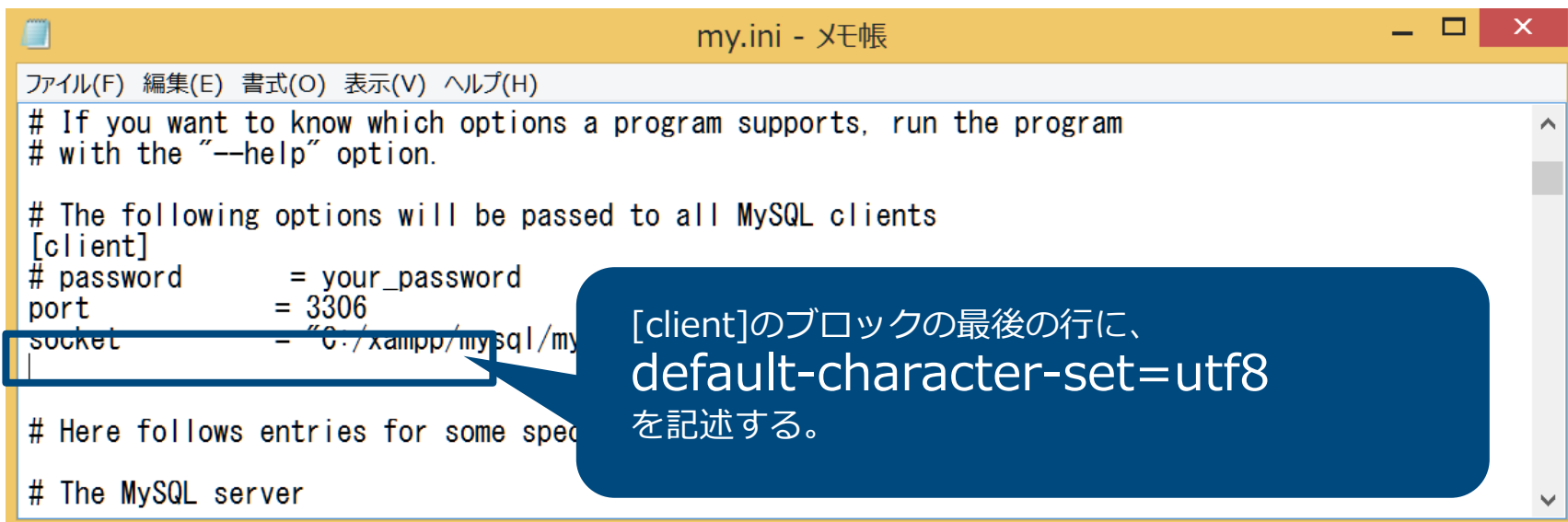
# 設定ファイルの編集

A screenshot of a Notepad window titled "my.ini - メモ帳". The window contains the text of a MySQL configuration file. The text is as follows:

```
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
# Example MySQL config file for small systems.
#
# This is for a system with little memory (<= 64M) where MySQL is only used
# from time to time and it's important that the mysqld daemon
# doesn't use much resources.
#
# You can copy this file to
# C:/xampp/mysql/bin/my.cnf to set global options,
# mysql-data-dir/my.cnf to set server-specific options (in this
# installation this directory is C:/xampp/mysql/data) or
# ~/.my.cnf to set user-specific options.
#
# In this file, you can use all long options that a program supports.
# If you want to know which options a program supports, run the program
# with the "--help" option.
#
# The following options will be passed to all MySQL clients
[client]
# password          = your_password
port                = 3306
socket              = "C:/xampp/mysql/mysql.sock"
default-character-set=utf8
# Here follows entries for some specific programs
```

この中身を変更していく

# 設定ファイルの編集

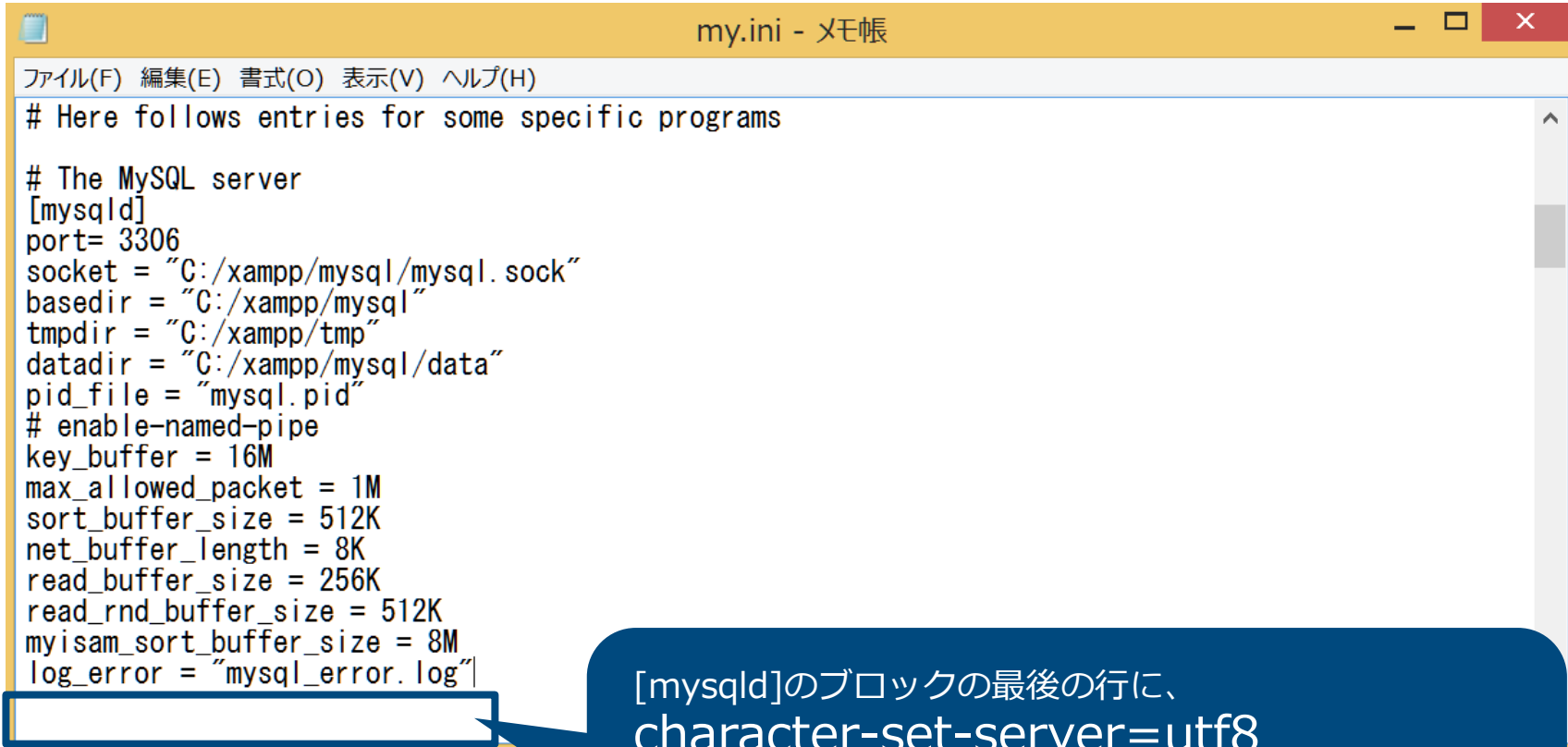


```
my.ini - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
# If you want to know which options a program supports, run the program
# with the "--help" option.

# The following options will be passed to all MySQL clients
[client]
# password      = your_password
port            = 3306
socket          = "C:/xampp/mysql/mysql.sock"
# Here follows entries for some specific users
# The MySQL server
```

[client]のブロックの最後の行に、  
**default-character-set=utf8**  
を記述する。

# 設定ファイルの編集



```
my.ini - メモ帳
ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)
# Here follows entries for some specific programs

# The MySQL server
[mysqld]
port= 3306
socket = "C:/xampp/mysql/mysql.sock"
basedir = "C:/xampp/mysql"
tmpdir = "C:/xampp/tmp"
datadir = "C:/xampp/mysql/data"
pid_file = "mysql.pid"
# enable-named-pipe
key_buffer = 16M
max_allowed_packet = 1M
sort_buffer_size = 512K
net_buffer_length = 8K
read_buffer_size = 256K
read_rnd_buffer_size = 512K
myisam_sort_buffer_size = 8M
log_error = "mysql_error.log"
```

[mysqld]のブロックの最後の行に、  
character-set-server=utf8  
skip-character-set-client-handshake  
を記述する。

# 設定ファイルの編集

my.ini - メモ帳

ファイル(F) 編集(E) 書式(O) 表示(V) ヘルプ(H)

```
innodb_lock_wait_timeout = 50

## UTF 8 Settings
#init-connect='SET NAMES utf8'
#collation_server=utf8_unicode_ci
#character_set_server=utf8
#skip-character-set-client-handshake
#character_sets_dir="C:/xampp/mysql/share/charsets"

[mysqldump]
quick
max_allowed_packet = 16M
|

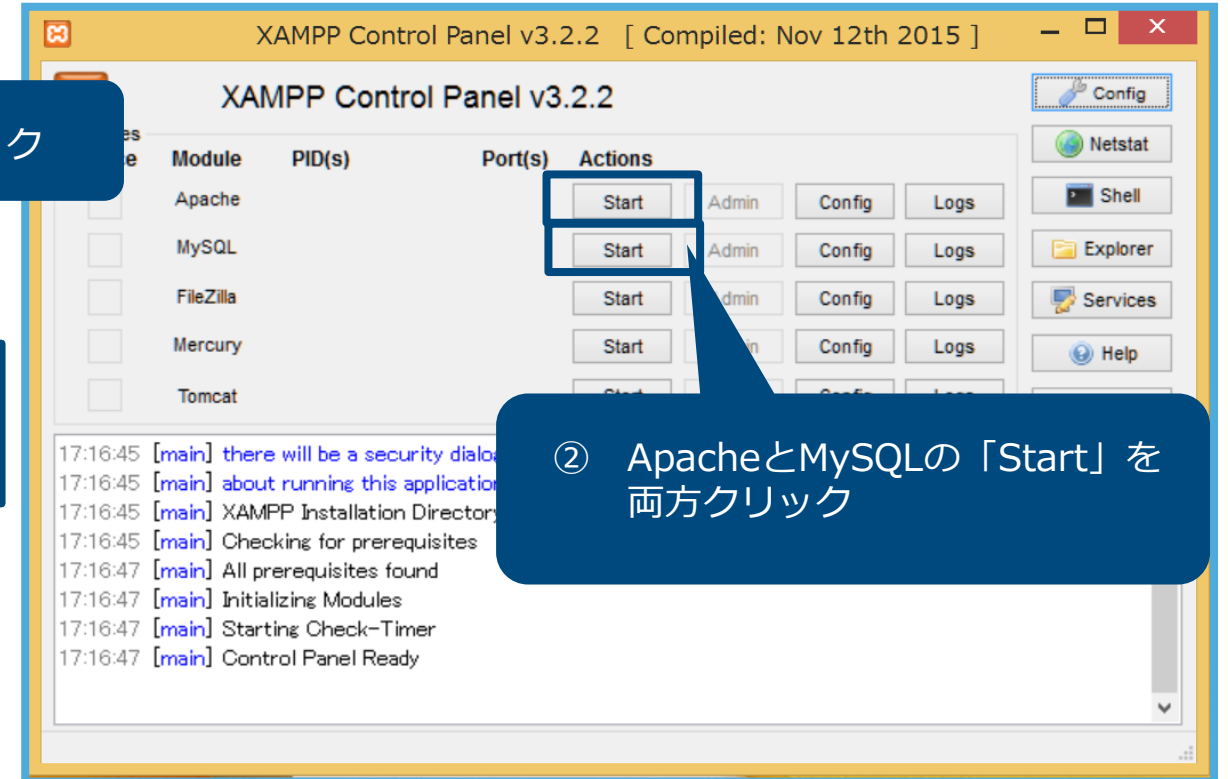
[mysql]
default-character-set=utf8
no-auto-rehash
# Remove the next comment character if you are not familiar with SQL
#safe-updates
|
```

[mysqldump]のブロックの最後の行に、  
**default-character-set=utf8**  
を記述する。

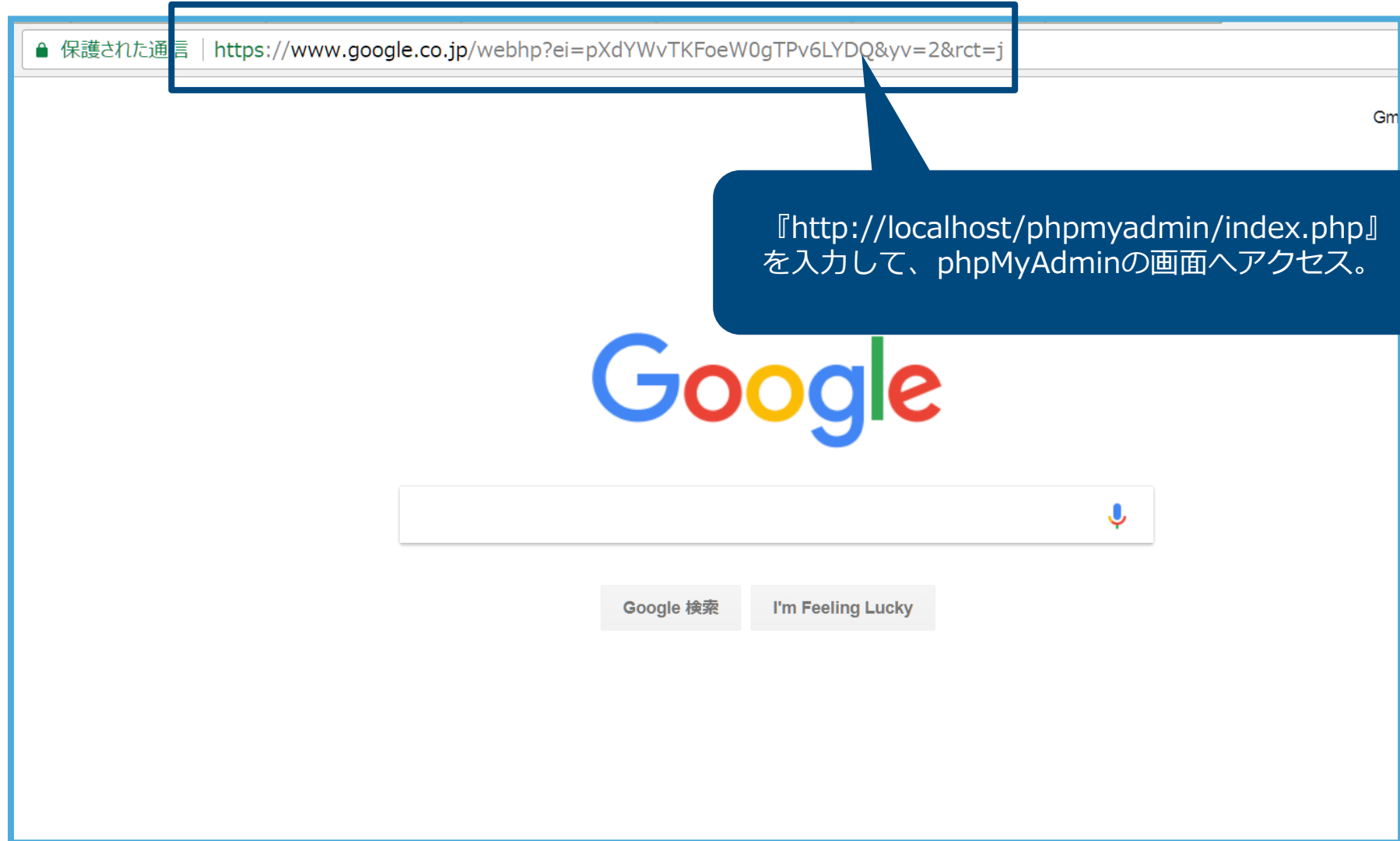
[mysql]のブロックの最後の行に、  
**default-character-set=utf8**  
を記述する。

ここまで編集したら『my.ini』を保存

# XAMPPの起動



# phpMyAdminへアクセス



# データベースの選択

① 『lesson01』 をクリックして開く。

※lesson01でなくとも、自分が作成したデータベースであればOK

② 『SQL』 をクリック。

The image shows a two-step process in phpMyAdmin. The first step shows the 'Servers' list on the left sidebar with 'lesson01' highlighted. The second step shows the 'SQL' tab selected in the top navigation bar, with a table list below it.

**Step 1: Selecting the Database**

The phpMyAdmin interface shows the 'Servers' list on the left sidebar. The 'lesson01' database is highlighted. The 'Appearance settings' panel on the right shows the language set to '日本語 - Japanese' and the theme set to 'pmahomme'.

**Step 2: Executing SQL**

The 'SQL' tab is selected in the top navigation bar. The table list below shows the following tables:

テーブル	操作
addresslist	表示 構造 検索 挿入 空にする 削除
productlist	表示 構造 検索 挿入 空にする 削除
vegetable	表示 構造 検索 挿入 空にする 削除

At the bottom, it indicates '11 テーブル 合計' (Total 11 tables).

# テーブル作成





# テーブル構造を考えよう

左のフォームを作る場合、テーブル構造は右のようになる

お問い合わせフォーム

名前

メールアドレス

年齢

選択してください

コメント

送信する

テーブル名は『contactform』

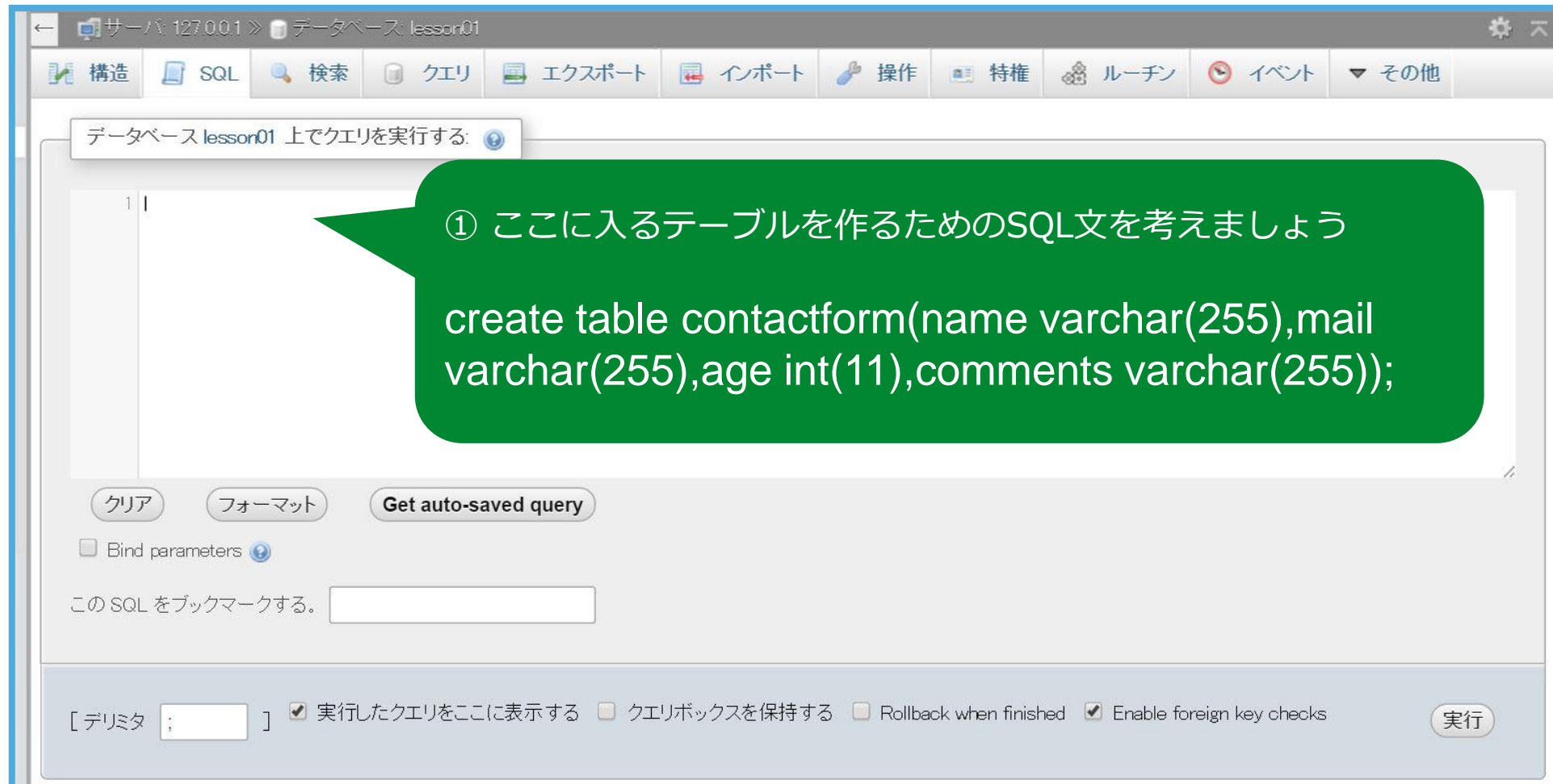
name	mail	age	comments
山田 太郎	abcd@yahoo.co.jp	25	こんにちは
佐藤 次郎	a994@gmail.com	30	おはよう
田中 花子	zzz22@yahoo.co.jp	22	さようなら
木村 明子	yamada@internous.co.jp	32	ありがとう

# テーブルを作成するSQL文を考えましょう

ヒント

『XAMPP&MySQL3時間目』のテーブル作成を参照してください。





# テーブルができた

② 『構造』をクリック

① 『contactform』というテーブルが作成出来た

③ contactformのテーブル構造が開いた

The screenshot shows the phpMyAdmin interface. On the left, the database 'lesson01' is selected, and the table 'contactform' is highlighted in the table list. The main panel shows the 'contactform' table structure. The table has four columns: 'name', 'mail', 'age', and 'comments', all of type 'varchar(255)' or 'int(11)'. The 'name' column is the primary key. The 'comments' column is also a primary key. The table structure is displayed in a table format with columns for column number, name, data type, collation, attributes, NULL, default value, and other options.

#	名前	データ型	照合順序	属性	NULL	デフォルト値	その他	操作
1	name	varchar(255)			はい	NULL		変更 削除 主
2	mail	varchar(255)			はい	NULL		変更 削除 主
3	age	int(11)			はい	NULL		変更 削除 主
4	comments	varchar(255)			はい	NULL		変更 削除 主

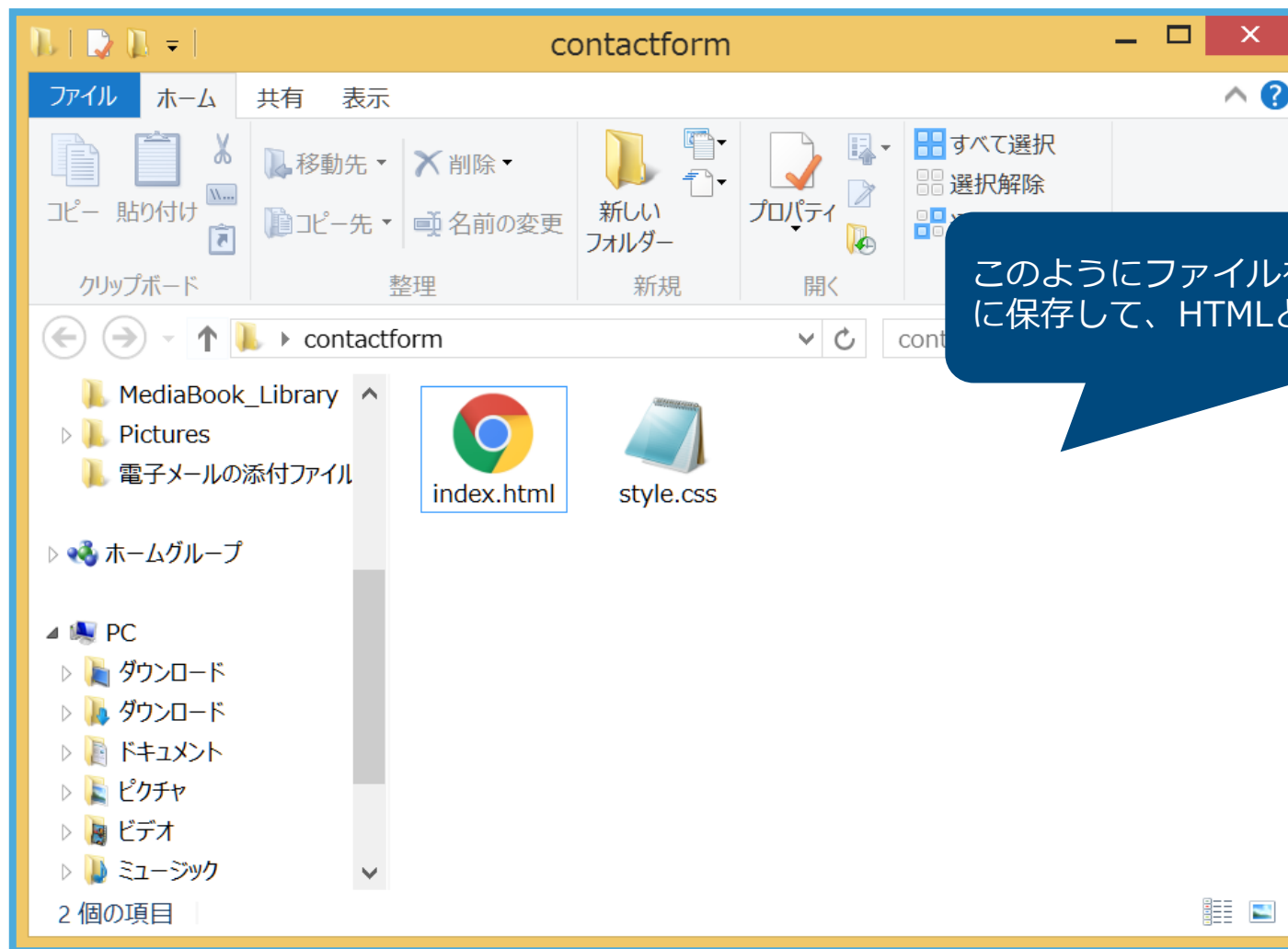
11 テーブル

テーブル構造を確認する テーブルを追跡する カラムを移動させる

個のカラムを追加する comments の後へ 実行

+ インデックス

# HTMLとCSSを作成



このようにファイルをフォルダ名「contactform」に保存して、HTMLとCSSを作成する。

## HTMLとCSSで下記のフォームを作成しましょう

ヒント

『HTML8時間目』のフォーム作成と、CSSを参照してください。

## お問い合わせフォーム

名前

メールアドレス

年齢

コメント

送信する

`<input type="..." name="XXX"/>`  
の部分は、『XXX』と記述すること

年齢は、18～65歳まで登録できるようにする。  
※JavaScriptのfor文でループ処理を使用すること。

`form method="XXX" action="XXX"`  
の部分は、『XXX』と記述すること

index.html(htmlファイル)

```
<!doctype HTML>
<html lang="ja">

<head>
  <meta charset="utf-8">
  <title>お問い合わせフォームを作る</title>
  <link rel="stylesheet" type="text/css" href="style.css">
</head>

<body>

  <h1>お問い合わせフォーム</h1>
  <form method="XXX" action="XXX">
    <div>
      <label>名前</label>
      <br>
      <input type="text" class="text" size="35" name="XXX">
    </div>

    <div>
      <label>メールアドレス</label>
      <br>
      <input type="text" class="text" size="35" name="XXX">
    </div>
```

[次のページへ](#)

## 前のページからの続き

```
<div>
  <label>年齢</label>
  <br>
  <select class="dropdown" name="XXX">
    <option>選択してください</option>
    <script>
      for (var i = 18; i <= 65; i++) {
        document.write("<option>" + i + "歳</option>");
      }
    </script>
  </select>
</div>

<div>
  <label>コメント</label>
  <br>
  <textarea cols="35" rows="7" name="XXX"></textarea>
</div>

<div>
  <input type="submit" class="submit" value="送信する">
</div>
</form>

</body>
</html>
```



style.css(CSSファイル)

```
body{font-family: "Hiragino Kaku Gothic ProN","メイリオ", sans-serif;
}
```

```
h1{ margin: 0 auto;
    margin-top:30px;
    margin-bottom:20px;
    border-left:4px red solid;
    background-color:#f1f1f1;
    padding-top:5px;
    padding-bottom:5px;
    padding-left:15px;
    width:500px;
    font-size:20px;
}
```

```
form{margin: 0 auto;
    background-color:#fbfbfb;
    border:3px #ebebeb solid;
    border-radius: 10px;
    width:500px;
}
```

次のページへ

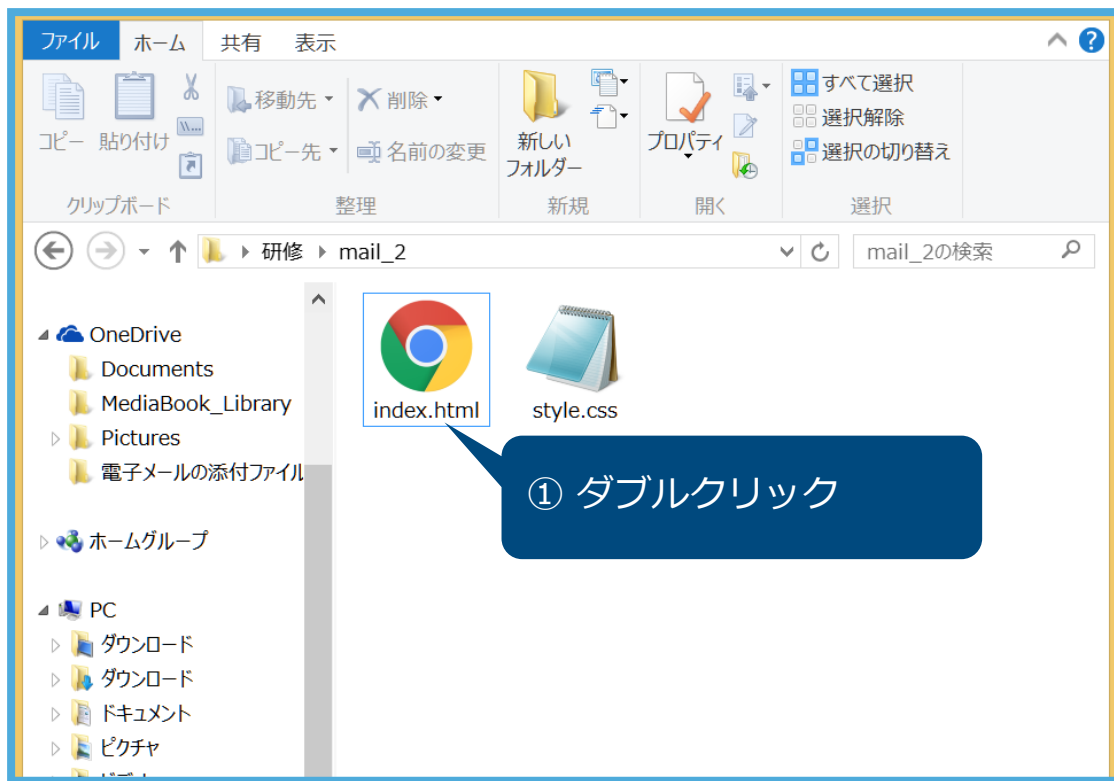
## 前のページからの続き

```
form div{
  padding:15px;
}

.text, textarea, .dropdown{
  border:1px solid #dfdfff;
  padding:5px;
  color: #999;
  background: #fff;
}

.submit{
  border:1px solid lightgray;
  padding: 4px 10px;
  color:white;
  background-color:#bfbfbf;
  border-radius: 5px;
  border-bottom: solid 2px gray;
}
```

# HTMLとCSSが完成



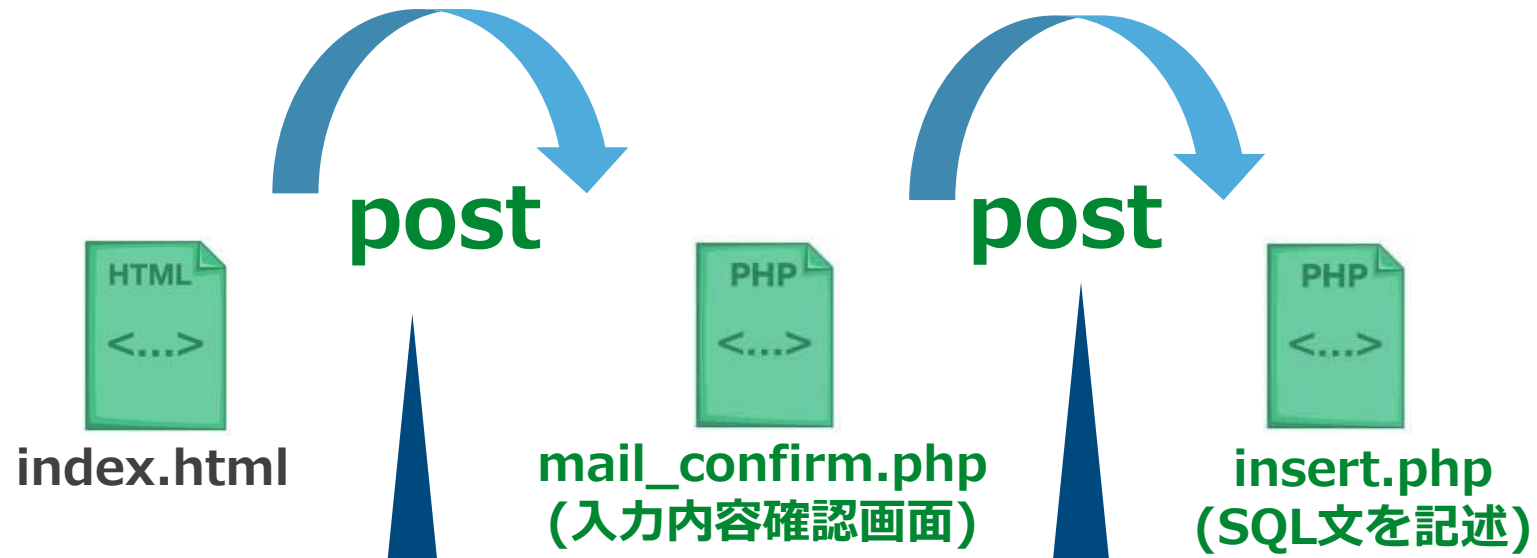
# formの中にある XXX の解説

methodには、“**post**”と記述、actionには、“**ファイル名**”を記述

```
<form method="post" action="ファイル名
```

```
</form>
```

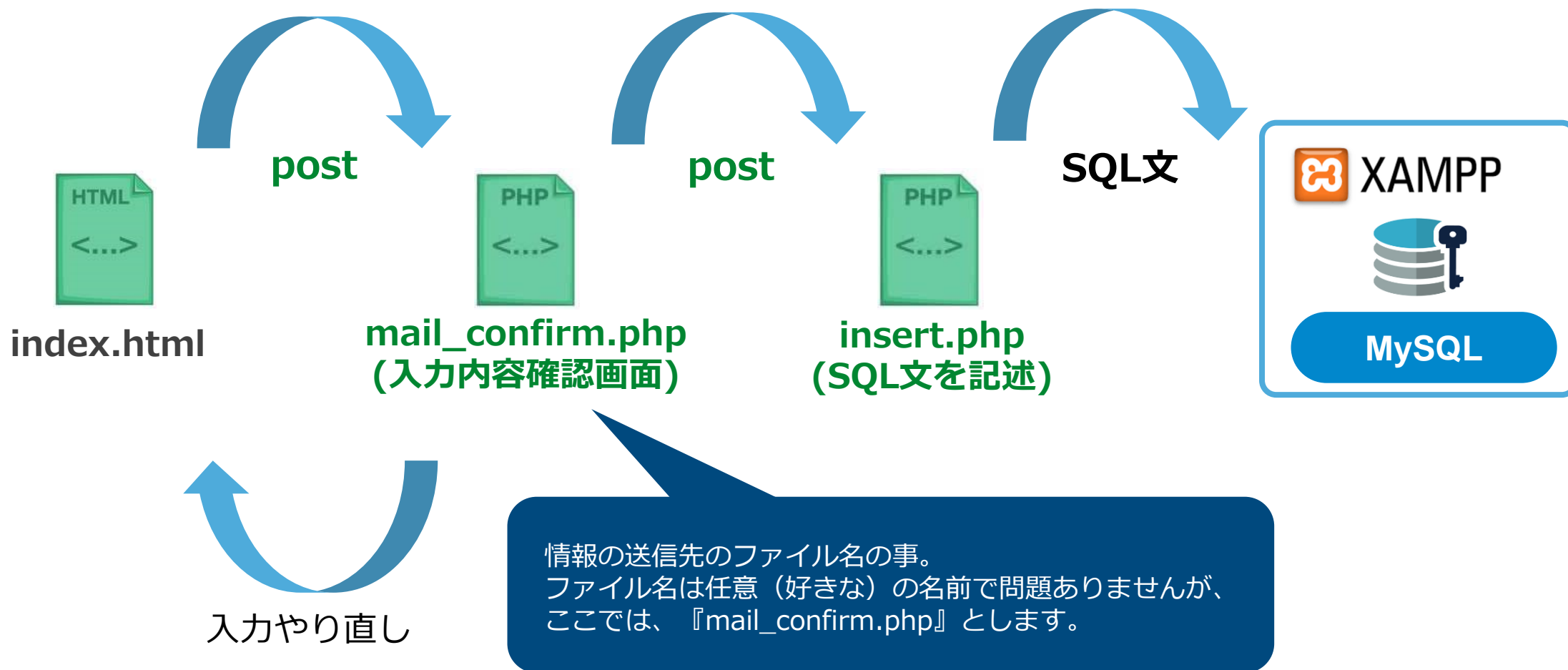
# postとは



このようにファイル間で、何らかの情報を引き渡す方法のことを『**postメソッド(=post通信)**』という。

※insert.phpからXAMPP (MySQL) への情報の引き渡しは、postメソッドではなく、SQL文になる。

# ファイル名とは



## formの中にある XXX の解説

つまり・・・XXXの部分は下記のように記述する

```
<form method=“post” action=“mail_confirm.php”>
```

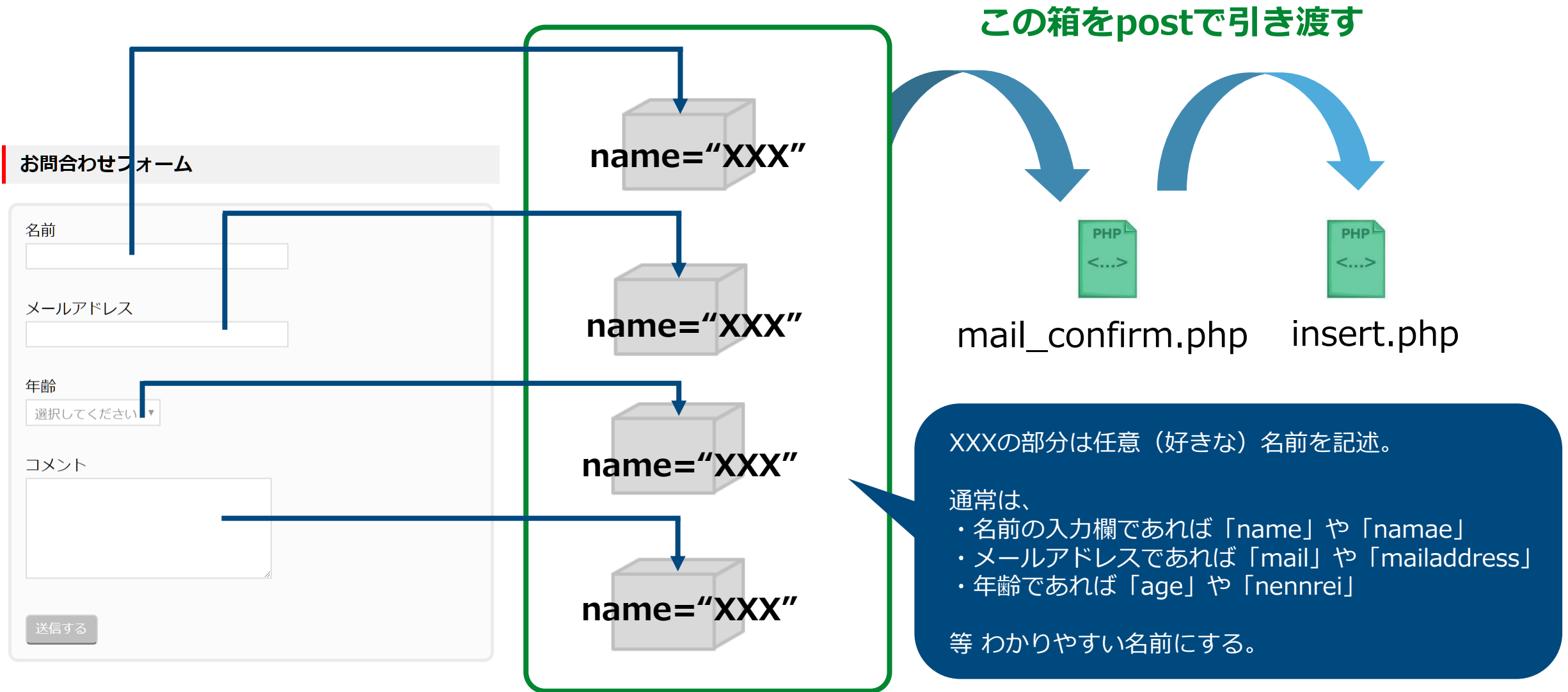
```
</form>
```

input . . . の nameには、**"箱の名前"**を記述

```
<div>  
  <label>名前</label>  
  <br>  
  <input type="text" class="text" size="35" name="箱の名前">  
</div>
```



# formの中にある XXX の解説



つまり・・・XXXの部分は下記のように記述する

```
<form method="post" action="mail_confirm.php">  
  <div>  
    <label>名前</label>  
    <br>  
    <input type="text" class="text" size="35" name="name">  
  </div>  
  
  <div>  
    <label>メールアドレス</label>  
    <br>  
    <input type="text" class="text" size="35" name="mail">  
  </div>
```

・・・ここから下も同様に name="XXX" に任意のわかりやすい箱の名前を記述する

# 今のHTMLの状態

index.html(htmlファイル)

```
<!doctype HTML>
<html lang="ja">

<head>
  <meta charset="utf-8">
  <title>お問い合わせフォームを作る</title>
  <link rel="stylesheet" type="text/css" href="style.css">
</head>

<body>

  <h1>お問い合わせフォーム</h1>
  <form method="post" action="mail_confirm.php">
    <div>
      <label>名前</label>
      <br>
      <input type="text" class="text" size="35" name="name">
    </div>

    <div>
      <label>メールアドレス</label>
      <br>
      <input type="text" class="text" size="35" name="mail">
    </div>
```

次のページへ

# 今のHTMLの状態

## 前のページからの続き

```
<div>
  <label>年齢</label>
  <br>
  <select class="dropdown" name="age">
    <option>選択してください</option>
    <script>
      for (var i = 18; i <= 65; i++) {
        document.write("<option>" + i + "歳</option>");
      }
    </script>
  </select>
</div>

<div>
  <label>コメント</label>
  <br>
  <textarea cols="35" rows="7" name="comments"></textarea>
</div>

<div>
  <input type="submit" class="submit" value="送信する">
</div>
</form>

</body>
</html>
```

### < ドロップダウンメニューのHint >

```
<select class="XXX" name="XXX">
  <option>AAA</option>
  <option>BBB</option>
  <option>CCC</option>
</slect>
```

と、

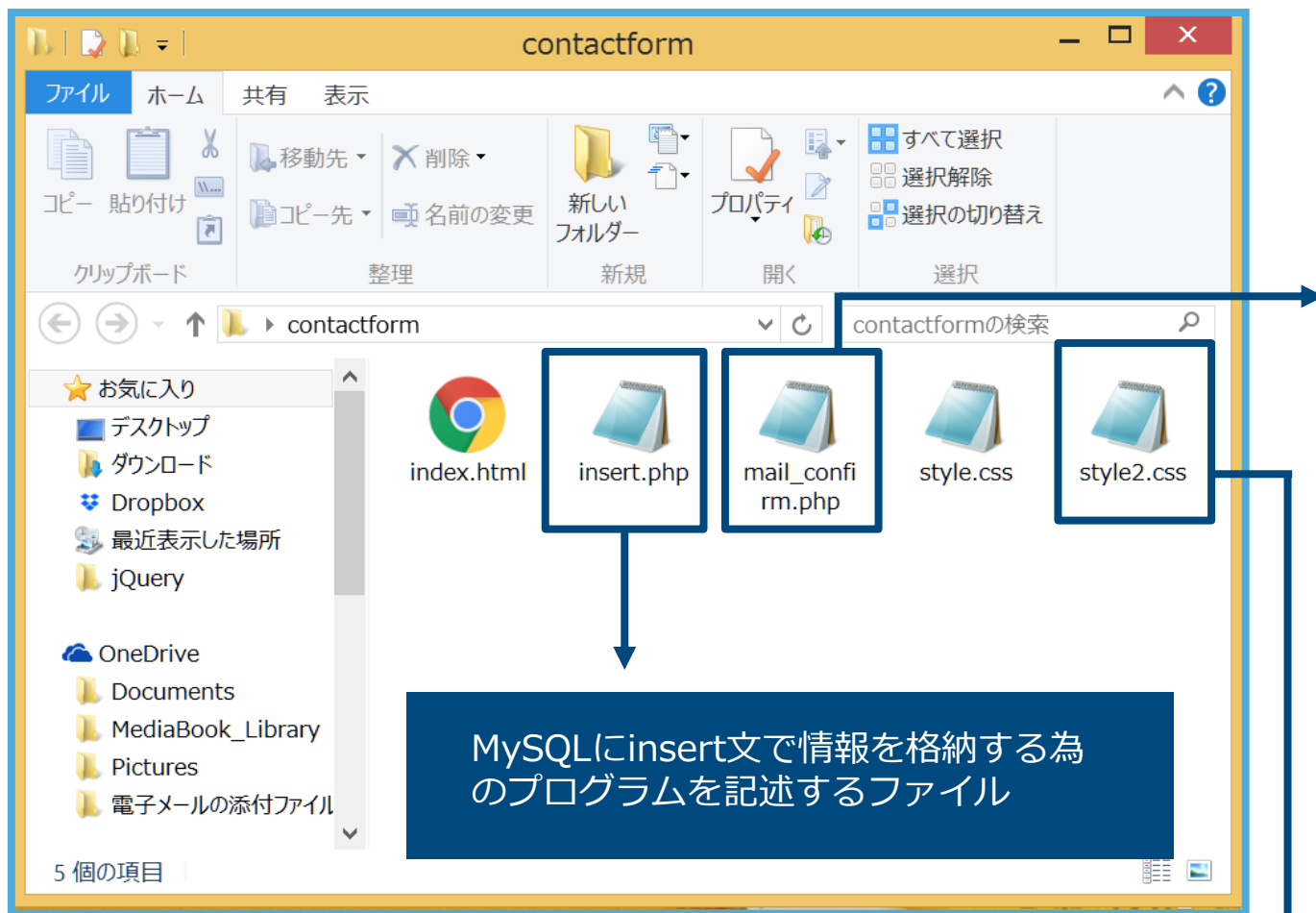
```
<select class="XXX" name="XXX">
  <option value="AAA">AAA</option>
  <option value="BBB">BBB</option>
  <option value="CCC">CCC</option>
</slect>
```

は、同じ意味です。

## ここからPHPファイルを作成します

- 1 設定ファイルの編集
- 2 XAMPP (MySQL) 上にtableを作成する
- 3 HTMLとCSSを作成する
- 4 **PHPファイルを作成する**

# 問い合わせフォーム作成の手順



フォームに入力した情報を確認する画面

### お問い合わせ内容確認

お問い合わせ内容はこちらで宜しいでしょうか？  
よろしければ「送信する」ボタンを押して下さい。

名前

メールアドレス

年齢

コメント

戻って修正する 登録する

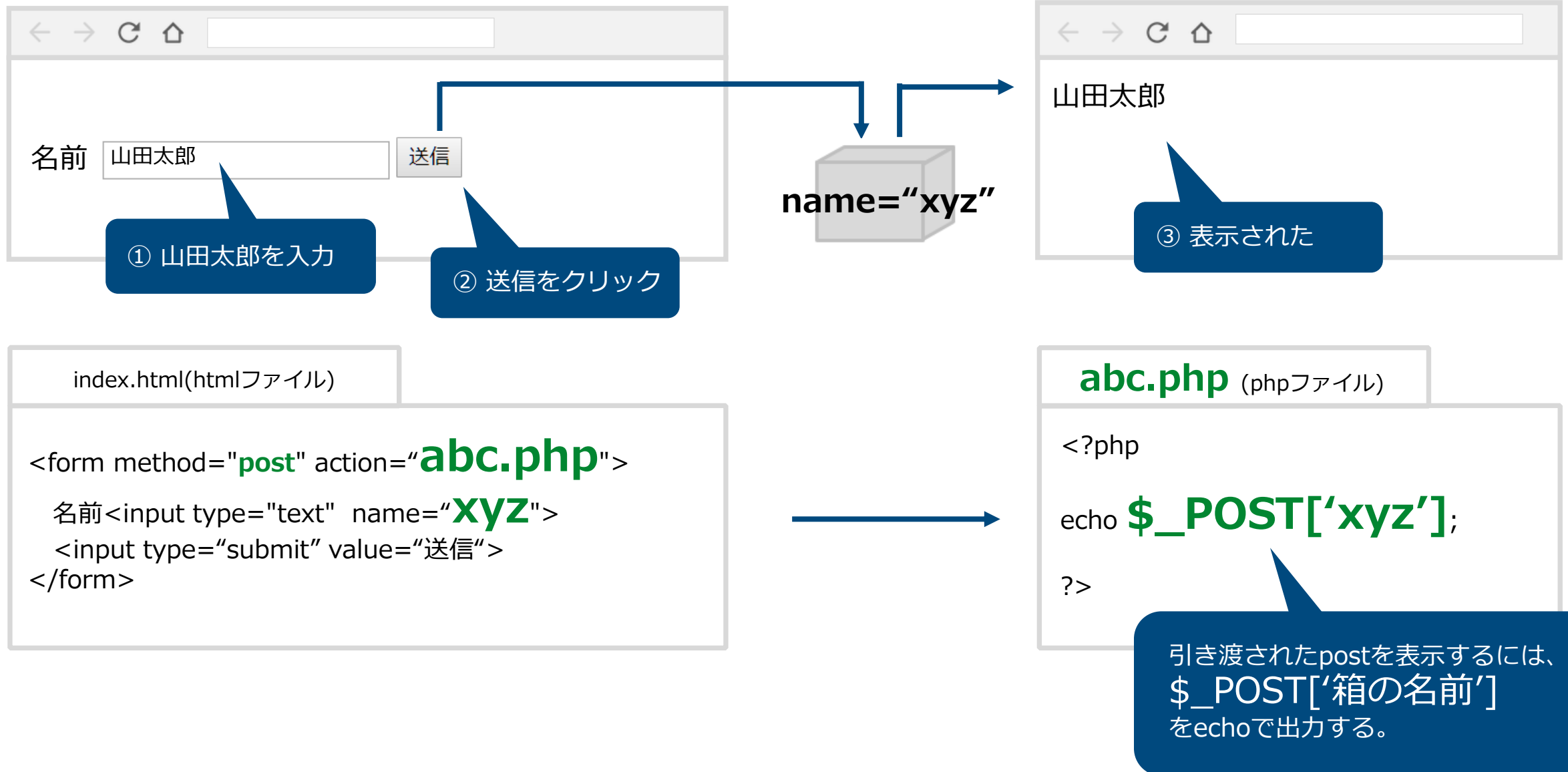
『mail\_confirm.php(フォームに入力した情報を確認する画面)』用のCSSを記述するファイル

```
echo $_POST['箱の名称'];
```

postを扱う際の決まり文句

form内の `<input ... name="〇〇〇">` の箱の名称  
formの入力欄ごとに指定する必要あり

# PHP側でのpostの使い方





# 『mail\_confirm.php』を作成する

mail\_confirm.php(PHPファイル)

```
<!doctype HTML>
<html lang="ja">

<head>
  <meta charset="utf-8">
<title>お問い合わせフォームを作る</title>
  <link rel="stylesheet" type="text/css" href="style2.css">
</head>

<body>
  <h1>お問い合わせ内容確認</h1>

  <div class="confirm">
    <p>お問い合わせ内容はこちらで宜しいでしょうか？
      <br> よろしければ「送信する」ボタンを押して下さい。
    </p>
    <p>名前
      <br>
      <?php echo $_POST['name']; ?>
    </p>

    <p>メールアドレス
      <br>
      <?php echo $_POST['mail']; ?>
    </p>
```

『index.html』から引き渡された『name』という箱を表示するという意味。

次のページへ

# 『mail\_confirm.php』を作成する

## 前のページからの続き

```
<p>年齢  
<br>  
<?php echo $_POST['age']; ?>  
</p>
```

```
<p>コメント  
<br>  
<?php echo $_POST['comments']; ?>  
</p>
```

```
<form action="index.html">  
  <input type="submit" class="button1" value="戻って修正する" />  
</form>
```

```
<form action="insert.php" method="post">  
  <input type="submit" class="button2" value="登録する" />  
  <input type="hidden" value="<?php echo $_POST['name']; ?>" name="name">  
  <input type="hidden" value="<?php echo $_POST['mail']; ?>" name="mail">  
  <input type="hidden" value="<?php echo $_POST['age']; ?>" name="age">  
  <input type="hidden" value="<?php echo $_POST['comments']; ?>" name="comments">  
</form>
```

```
</div>  
</body>  
</html>
```

問合わせ画面に戻るボタン。

- ・ action="〇〇〇"は、リンク先のURL(ファイル名) のこと。
- ・ value="〇〇〇"は、ボタン上に表示されるテキストのこと。

『index.html』から引き渡されたpostをここで、再度箱の中に格納し、『insert.php』へ引き渡す。

type="hidden"にすることで、代入した内容を隠す（web上に表示しない）ことが出来る。

# 『style2.css』を作成する

style2.css(CSSファイル)

```
body{font-family: "Hiragino Kaku Gothic ProN","メイリオ", sans-serif;  
}
```

```
h1{ margin: 0 auto;  
    margin-top:30px;  
    margin-bottom:20px;  
    border-left:4px red solid;  
    background-color:#f1f1f1;  
    padding-top:5px;  
    padding-bottom:5px;  
    padding-left:15px;  
    width:500px;  
    font-size:20px;  
}
```

```
.confirm{margin: 0 auto;  
    background-color:#fbfbfb;  
    border:3px #ebebeb solid;  
    border-radius: 10px;  
    width:450px;  
    padding-top:20px;  
    padding-left:20px;  
    padding-right:20px;  
    padding-bottom:60px;  
}
```

[次のページへ](#)

# 『style2.css』を作成する

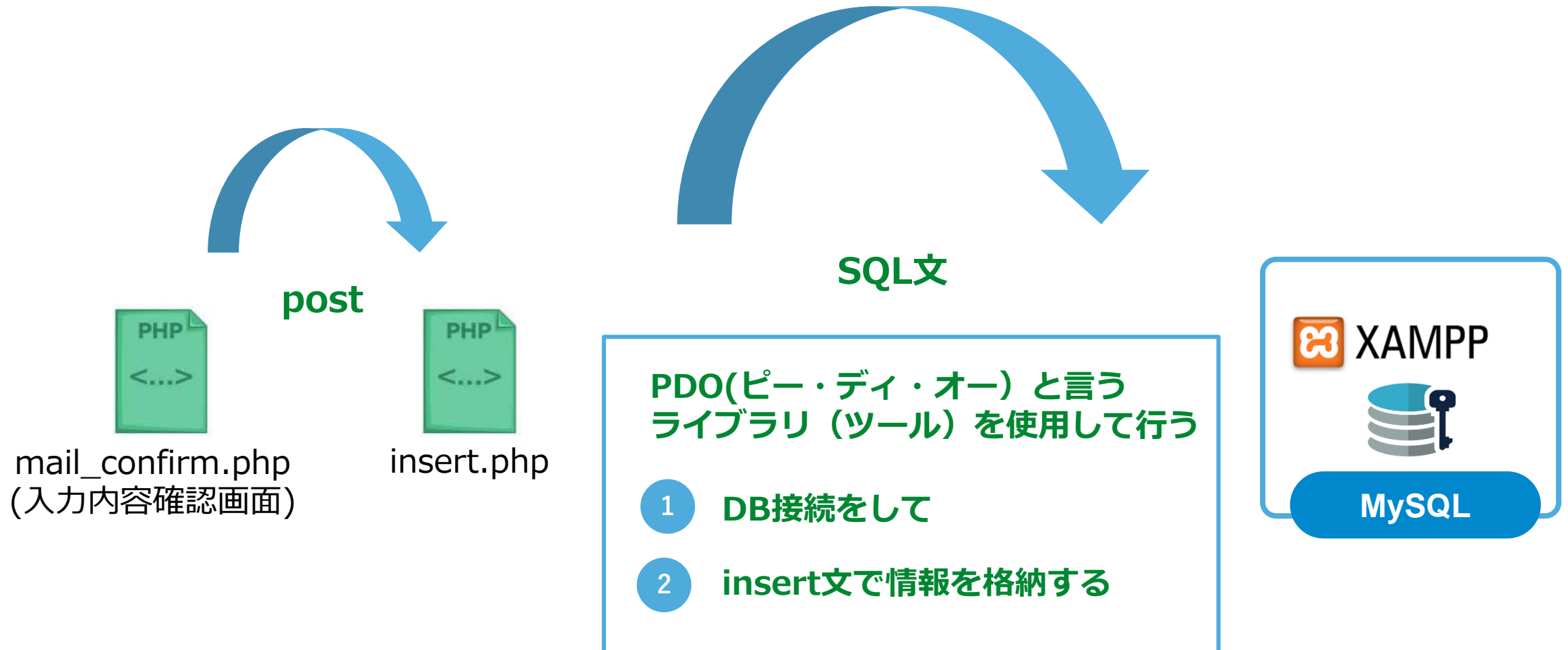
## 前のページからの続き

```
form{float:left;  
    margin-right:15px;  
}
```

```
.button1{border:1px solid lightgray;  
    padding: 4px 10px;  
    color:white;  
    background-color:#bfbfbf;  
    border-radius: 5px;  
    border-bottom: solid 2px gray;  
}
```

```
.button2{border:1px solid lightgray;  
    padding: 4px 10px;  
    color:white;  
    background-color:darkorange;  
    border-radius: 5px;  
    border-bottom: solid 2px #E08600;  
}
```

# PHP上でSQL文を書く方法



# PDOの使い方

```
mb_internal_encoding("utf8");
```

DBへ情報を格納する際に、文字化けしないようにするための決まり文句

```
$pdo = new PDO("mysql:dbname=lesson01;host=localhost;" , "root" , "");
```

DBと接続する為の決まり文句

MySQLに接続し、  
データベース「lesson01」  
を利用するという意味

通常は、DB用のサーバー名を  
記述しますが、今回は、  
自分のPC(=ローカル環境)の  
XAMPPを使用しているので、  
このような記述になる。

サーバー接続する際の  
IDとパスワードを記述。  
XAMPPの初期状態での  
IDとパスワードは  
ID : root  
PW : なし  
なので、このような記述になる。

# PDOの使い方

```
$pdo->exec("insert into テーブル名(カラム名,カラム名)values
```

```
("'".$_POST['箱の名前'].",'".$_POST_['箱の名前'].');"");
```

通常のSQL文(insert文)と  
同じ記述の仕方

DBへ送る為の情報。

下記のような並びになるので間違えないように注意。

『』 『"』 『.』 \$\_POST['箱の名前'] 『.』 『"』 『'』  
複数の情報を送る場合は、カンマ区切りにすること。

DBへ格納する際に記述するSQL文(insert文)の前  
に記述する決まり文句

※CRUDのうち、select文のみ『exec』でなく  
『query』を利用しますが、詳細は後述します。

# 『insert.php』を作成する

insert.php(PHPファイル)

```
<?php
mb_internal_encoding("utf8");

$pdo = new PDO("mysql:dbname=lesson01;host=localhost;" , "root" , "");

$pdo->exec("insert into contactform(name,mail,age,comments)
values('".$_POST['name']."' , '".$_POST['mail']."' , '".$_POST['age']."' , '".$_POST['comments']."' );");

?>

<!doctype HTML>
<html lang="ja">

<head>
<meta charset="utf-8">
<title>お問い合わせフォームを作る</title>
<link rel="stylesheet" type="text/css" href="style2.css">
</head>
```

環境によって異なる。  
※デフォルト（＝初期）設定では、  
『 "root", "" 』でOK。

インターネットの研修所内では、2017.12月時点  
では、『 "root", "mysql" 』で設定すること。

次のページへ



# 『insert.php』を作成する

## 前のページからの続き

```
<body>

  <h1>お問い合わせフォーム</h1>

  <div class="confirm">
    <p>お問い合わせ有難うございました。<br>3営業日以内に担当者よりご連絡差し上げます。
  </p>

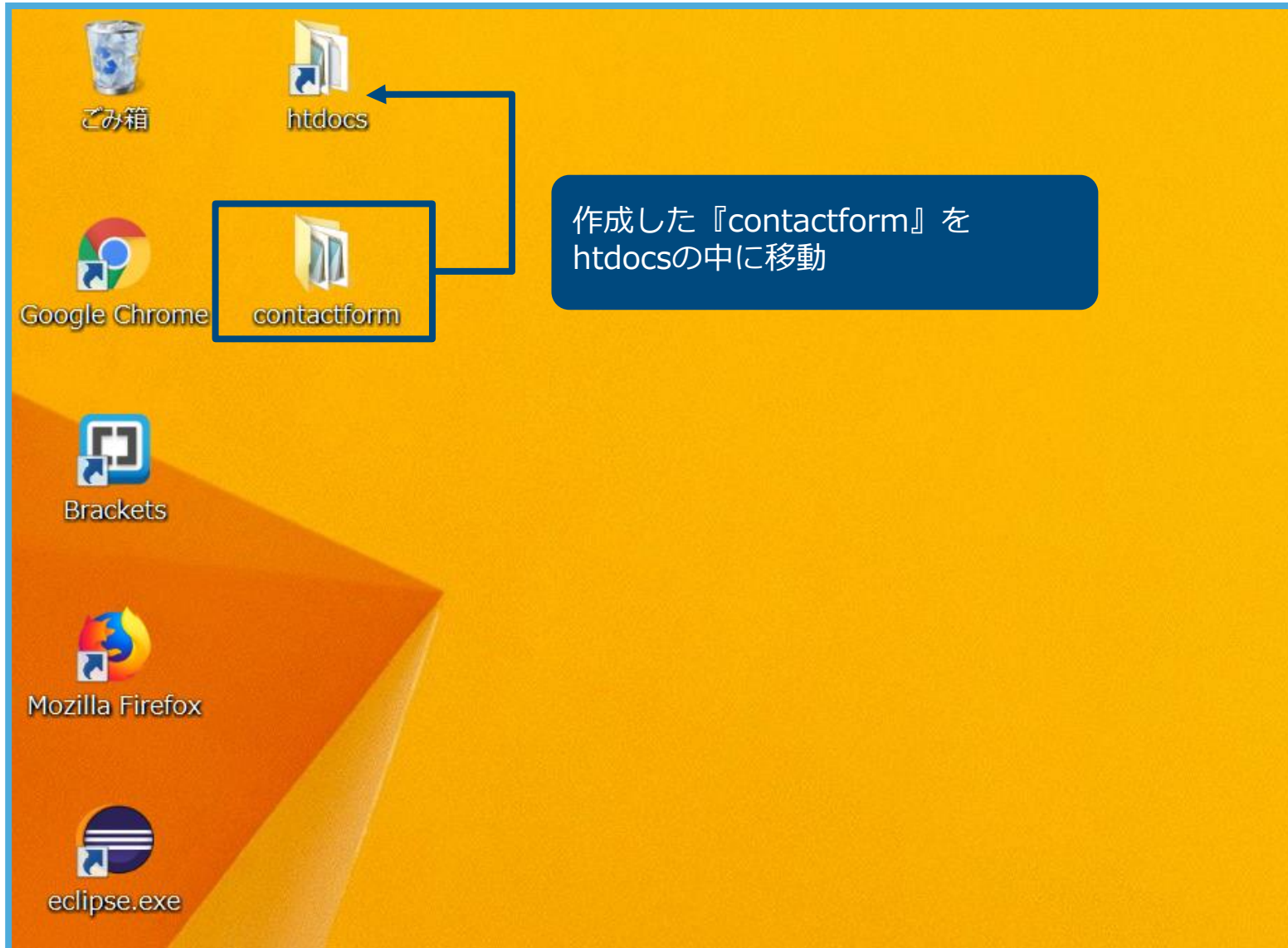
  </div>

</body>
</html>
```

これで完成

問合せフォームが動くかテストしてみよう

# htdocsに『contactformを移動』



# 作成したページへアクセス



# 登録画面

## お問い合わせフォーム

名前

山田太郎

『山田太郎』 と入力

メールアドレス

yamadataro@gmail.com

『yamadataro@gmail.com』 と入力

年齢

25歳

『25歳』 を選択

コメント

山田太郎です。

『山田太郎です。』 と入力

送信する

『送信する』 をクリック

# 登録確認画面

## お問い合わせ内容確認

お問い合わせ内容はこちらで宜しいでしょうか？  
よろしければ「送信する」ボタンを押して下さい。

名前

山田太郎

メールアドレス

yamadataro@gmail.com

年齢

25歳

コメント

山田太郎です。

戻って修正する

登録する

## お問い合わせフォーム

お問い合わせ有難うございました。  
3営業日以内に担当者よりご連絡差し上げます。

登録フォーム画面へ戻る

# 登録後のXAMPP(MySQL)の画面

⚠ Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy

✓ Showing rows 0 - 0 (1 total, Query took 0.0019 seconds.)

```
SELECT * FROM `contactform`
```

[ E

☐ すべて表示 | 行数: 25 ▼ Filter rows:

+ オプション

name	mail	age	comments
山田太郎	yamadataro@gmail.com	25	山田太郎です。

登録した内容がDBに格納されている