データベース及び演習 （2023年6月12日）

水谷祐生[1,a)]

概要：本稿は，情報処理学会論文誌ジャーナルに投稿する原稿を執筆する際，および論文採択後に最終原稿を準備する際の注意点等をまとめたものである．大きく分けると，論文投稿の流れと，LaTeX と専用のスタイルファイルを用いた場合の論文フォーマットに関する指針，および論文の内容に関してするべきこと，するべきでないことをまとめたべからずチェックリストからなる．本稿自体も LaTeX と専用のスタイルファイルを用いて執筆されているため，論文執筆の際に参考になれば幸いである．

Database and Exercises (version 2012/10/12)

Yusei Mizutani[1,a)]

1. 機能概要

教科書 9 章、10 章に記されているサンプルコードにはユーザーからメールアドレス、パスワード、出身地、生年月日を受け付け、メールを用いて認証を行う会員登録機能と、会員登録を行なった後に専用ページに遷移するためのログイン機能、認証を行なった後、専用ページから退出するログアウト機能、認証されたユーザーを一覧することのできる管理者機能が実装されている．

また、全体の操作の流れは、ログイン画面から新規登録画面に遷移し、新規登録画面でメールアドレス、パスワード、出身地、生年月日を受け付ける．その後、入力したメールアドレスで登録確認を行うとログイン者専用画面に遷移する流れとなっている．

2. 利用技術

2.1 PHP

PHP とは「ピー・エイチ・ピー」と読み、Personal Home Page がその由来である．

主に、Web で利用される HTML 形式のようなハイパーテキストを閲覧者の操作によって動的な画面を作ることを得意としている．

PHP の特徴として誰でも無償で利用できるオープンソースのサーバサイド・スクリプト言語であることが挙げられる．さらに、無償で利用できるからと言ってメンテナンスがされていないなどはなく、マニュアル、バグ修正なども十分に行われているため、安心して利用できる．

また、多様な DB、ライブラリへの対応、デバッグのしやすさ、その習得のしやすさなどからプログラミング言語を初めて学ぶ人たちにも扱いやすい言語となっている．

2.2 MariaDB/MySQL

MySQL とはオープンソースソフトウェアの RDBMD(relational database management system) であり、現在は Oracle 社が管理している．

一方、MariaDB とは、MySQL から派生した RDBMD であり、MySQL のソースコードをベースにして、新機能追加やソースコードの改善が組み込まれている．

両者共に互換性があり、同じ SQL 文を用いてデータベース操作を行なっているが、もちろん違いもある．

[1] 情報処理学会
IPSJ, Chiyoda, Tokyo 101–0062, Japan
[†1] 現在，情報処理大学
Presently with Johoshori Uniersity
[a)] joho.taro@ipsj.or.jp

両者の違いとして、無償で全ての機能が使えるかどうかである.MariaDB は完全な GPL ライセンスでその全ての機能が使用可能である。一方で、MySQL は特別な機能は有料のライセンスに含まれるとする、デュアルライセンス方式を採用している.

2.3 CSS

CSS とは（Cascading Style Sheets）は、「シー・エス・エス」と読み、Web ページのデザインやレイアウトを制御する言語である. 主に HTML や XHTML などで作成されるウェブページにスタイルを適用したい場合に用いられている.

CSS を使用しなくとも、HTML には center や font タグなどの装飾目的の要素や属性が存在しているため、HTML だけでウェブページの見栄えを制御することもできなくはない. しかし、HTML は情報構造を定義するための言語であり、見栄えの制御のために本来の役割とは違った使い方をすると、 文書の情報構造がでたらめになってまうため、スタイリングに HTML を用いるべきではないとされている.

そこで HTML では文書構造のみを定義して、 スタイルについては CSSow 用いたスタイリングシートで指定することが推奨されている. そうすることで他の閲覧環境に依存せず、HTML を本来の役割で使用することが可能になった.

2.4 Smarty

Smarty とは、2000 年代前半にリリースされたスクリプト言語である PHP のテンプレートエンジンである.

テンプレートエンジンとは「機能を記述する内容（PHP）」と「画面の表示内容（HTML & CSS など）」を分けて管理できるツールである.

単純なページや数ページ程度ではテンプレートエンジンの恩恵を感じることができないが、PC とスマートフォンで別々のレイアウトを用意したい時や、膨大なページ数を用意する場合、その便利さを感じることができる.

従来は開発の際、HTML の中に PHP を埋め込む手法を取っていた. しかし、PHP と HTML が混在したファイルは、2 つのルールが一緒に書かれているので読み取りにくい点が難点であった. 両方のプログラミング言語が混在して 1 つのファイルに書かれている場合、修正にはページ数が多いほど膨大な時間がかかり、保守やメンテナンスがしづらいといった問題があった.

そこで、Smatry を用いることで、機能と表示内容の分離、つまり HTML と PHP を別のファイルで記述することができる. こうすることで、プログラマーとデザイン担当の作業ファイルを完全に分割することができるため、開発を効率的に行うことが可能になった.

2.5 HTTP

HTTP(HTTP: HyperText Transfer Protocol) とは、Web 情報をやりとりするプロトコルの一種であり、ホームページやブログを閲覧する際、HTTP を用いてサーバとクライアント間でやり取りが行われている.

特徴として、動作がとてもシンプルであることが挙げられる. 情報のやり取りは常に、クライアントが要求を出し、サーバが応答を返します。HTTP は 1 リクエスト 1 レスポンスを返すルールがあるため、どちらかが多くなることはあり得ない. また、あるリクエストに対するレスポンスは必ず同じものになる特徴がある.

このように、HTTP は完結で分かりやすい特性を持つことから、Web サーバと Web ブラウザ間のやり取り以外に、スマホやアプリからのサーバ機能呼び出しや、サーバ間のサービス呼び出しなどに幅広く使われている.

2.6 SMTP/IMAP/POP

SMTP とはシンプル・メール・トランスファー・プロトコルと読み、Simple Mail Transfer Protocol がその由来である. 主に、インターネットなどの TCP / IP ネットワークで用いられており、メールを送信するための通信プロトコルの一つである.

一方で POP とはポスト・オフィス・プロトコルと読み、Post Office Protocol がその由来である. こちらも SMTP 同様にインターネットなどの TCP / IP ネットワークで用いられているプロトコルであるが、SMTP と異なり、メールを受信するための通信プロトコルである.

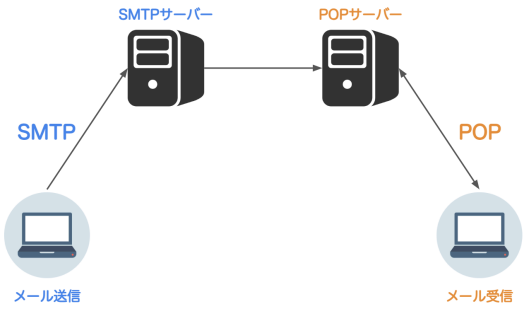

図 1 メール送受信の図

メール送受信の仕組みとしては、上記の図のようになっており、

(1) SMTP でメールをサーバーに送信する
(2) SMTP サーバーから POP サーバーに受信したメールの内容を送信する
(3) POP サーバーから受け取る側の PC にメールの内容を送信する

といった流れになっている.

3. システム設計

3.1 システム概要

今回の課題で用いたサンプルコードは，必要な場合に応じてクラスを分け，それぞれのクラスの名前に対して見合ったメソッド、変数を追加している. また、作成したしたクラスを継承し、別のファイルでもクラス内のメソッドを使用できるようにしている. そうすることで、一つのファイルに処理を何百行と書く必要がなくなり、システムの保守・管理・運用を容易に行うことが可能になる.

また、今回の会員登録時に用いた入力フォームではパスワードの長さの入力チェック、つまりバリテーションを行うために、PEAE ライブラリに実装している QuickForm2 を用いて、それを Smarty と連携させることで実装している.

会員専用画面である、登録後にユーザー一覧画面を表示する際は、ユーザ一覧のデータを見やすいように分割して表示するために Pager を用いて実装している.

3.2 画面遷移

3.2.1 一般会員

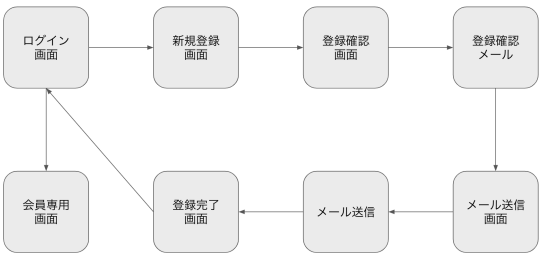

図 2 サンプルプログラムの画面遷移の図

図 2 は今回のサンプルプログラムにおける画面遷移を表したものであり、以下にその詳細を記す.

(1) ログインを行い、ログイン者専用のページに遷移する
(2) 1 で新規登録をしていない場合は、新規登録画面を押して、新規登録画面に遷移する
(3) 新規登録画面に必要な項目を入力する
(4) 登録ボタンを押すと、入力内容の確認画面に遷移する
(5) 入力した内容に間違いがなかったら、登録確認用のメールを入力されたメールアドレスに送信する
(6) メールの内容を確認し、本登録が完了できたら登録完了画面に遷移し、新規登録処理を完了する
(7) その後、ログイン画面に再度遷移する
(8) ログイン画面で、新規登録画面で入力したメールアドレスとパスワードを再度入力することで、ログイン者専用のページに遷移する

といった流れになっている.

3.2.2 管理者

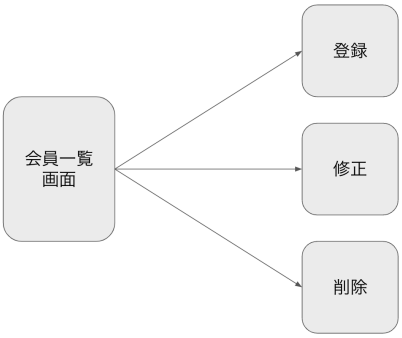

図 3 管理者から見た画面遷移の図

図 3 は今回のサンプルプログラムにおける管理者が行える操作を表したものであり、以下にその詳細を記す.

(1) 管理者専用の画面の新規登録ボタンを押すと図 2 の新規登録画面に遷移する
(2) 管理者専用画面の中のの会員一覧の更新ボタンを押すと、登録された会員情報の内容を修正することができる更新画面へ遷移する
(3) 管理者専用の画面の中の会員一覧に存在する削除ボタンを押すことで、登録された会員情報を削除することができる、削除確認画面に遷移する

といった流れになっている.

3.3 データベース設計

今回作成したサービスでは三つのテーブルが存在する. 一つ目は、日本の都道府県を閲覧する際に使用される ken テーブル、二つ目は、名前、メールアドレス、パスワードなど会員情報を管理するための member テーブル、三つ目はメールを用いて本登録を行っていないユーザーを仮登録として保持しておくための premember テーブル、これら三つのデータを MariaDB/MySQL を通して行うことでシステムを動かしている.

以下に、それぞれのテーブルについて記述していく.

3.3.1 ken テーブル

```
CREATE TABLE ken (
    id      SMALLINT,
    ken     VARCHAR(10),
    PRIMARY KEY (id)
);
```

上の SQL 文は都道府県のデータを格納するために定義されたテーブルである.

フィールドについて説明すると、id とはカラムを識別するため主キー、ken は都道府県の名前をを格納している.

3.3.2 member テーブル

```
CREATE TABLE member (
    id
    MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT,
    username    VARCHAR(50),
    password    VARCHAR(128),
    last_name   VARCHAR(50),
    first_name  VARCHAR(50),
    birthday    CHAR(8),
    ken         SMALLINT,
    reg_date    DATETIME,
    cancel      DATETIME,
    PRIMARY KEY (id)
);
```

上の SQL 文は本登録されたユーザーのデータを格納するために定義されたテーブルである.

以下、それぞれのフィールドについて説明する

(1) id とはカラムを識別するため主キー、また UNSIGNED NOT NULLAUTO_INCREMENT とは、カラムを追加される時に自動で順番に id フィールドに数字を格納するという意味である. ただし、0 は受け付けない.

(2) username フィールドは文字列で定義されていて、カッコ内にある 50 とは、50 バイト分の文字列まで格納できるという意味である.

(3) username フィールドは文字列で定義されていて、カッコ内にある 50 とは、50 バイト分の文字列まで格納できるという意味である.

(4) password フィールドは文字列で定義されていて、カッコ内にある 128 とは、128 バイト分の文字列まで格納できるという意味である.

(5) last_name フィールドは文字列で定義されていて、カッコ内にある 50 とは、50 バイト分の文字列まで格納できるという意味である.

(6) first_name フィールドは文字列で定義されていて、カッコ内にある 50 とは、50 バイト分の文字列まで格納できるという意味である. また、出身地の情報は都道府県を直接入れているわけではなく、ken テーブルで定義されている id を参照して管理している. こうすることでデータ量の多い都道府県の文字列よりもデータ量の少ない数値で管理できるためサーバーの負荷を軽減することができる.

(7) ken フィールドは数値で定義されていて、SMALLINT とは、-32,767 から 32,767 までの小さい整数を格納することができる意味である.

(8) reg_date フィールドは DARETIME 型で定義されていて、YYYY-MM-DD 形式で格納されている

(9) cancel フィールドは DARETIME 型で定義されていて、YYYY-MM-DD 形式で格納されている

3.3.3 premember テーブル

```
CREATE TABLE premember (
    id
    MEDIUMINT UNSIGNED NOT NULL AUTO_INCREMENT,
    username    VARCHAR(50),
    password    VARCHAR(128),
    last_name   VARCHAR(50),
    first_name  VARCHAR(50),
    birthday    CHAR(8),
    ken         SMALLINT,
    link_pass   VARCHAR(128),
    reg_date    DATETIME,
    PRIMARY KEY (id)
);
```

上の SQL 文は仮登録されたユーザーのデータを格納するために定義されたテーブルである.

基本的なフィールドは member テーブルとあまり差異はないが、一部異なる部分もある.link_pass フィールドである. このフィールドは本登録テーブルと仮登録テーブルの情報を結びつけるためのフィールドである.

3.4 システム詳細

表 1 各処理における分岐の表

type	action	ボタン	処理	表示画面
無し			認証	会員画面 TOP
無し			未認証	ログイン画面
authenticate	-	-	認証処理 OK	会員画面 TOP
authenticate	-	-	認証処理 NG	ログイン画面
regist	form	-	-	入力画面
regist	confirm	-	入力チェック OK	確認画面
regist	confirm	-	入力チェック NG	入力画面
regist	complete	戻る	-	入力画面
regist	complete	登録	INSERT	完了画面
modify	form	-	-	入力画面
modify	confirm	-	入力チェック OK	確認画面
modify	confirm	-	入力チェック NG	入力画面
modify	complete	戻る	-	入力画面
modify	complete	登録	UPDATE	完了画面
delete	confirm	-	-	入力画面
delete	complete	-	DELETE	完了画面
modify	-	-	ログアウト処理	ログイン画面

表 2 は，type と action の値によって，振り分けられるときの条件を表したものである. 次の画面へ遷移されるときリンクをクリックして画面を表示させるときか、送信ボタンを押して画面を表示させるときのいずれかである. 送信ボタンを押して、画面遷移する場合は、HTTP の GET メソッドにより、パラメーターを URL に載せて、サーバに送信している. 送信ボタンを押して画面遷移する場合は、HTTP の POST メソッドを用いて、ユーザーからは確認

できないよう安全に値が送信される.

ここで、GET メソッドと POST メソッドの違いについて説明する.

GET メソッドは URL にパラメータを含めることで、サーバーに情報を送る HTTP リクエストの一つである. 主に、あるページを取得するためにサーバーにリクエストを送る際にこのメソッドが用いられる.

POST メソッドとはパラメータをリクエストヘッダーに含めることで、サーバーとデータのやり取りを行う HTTP リクエストの一つである.GET メソッドと明確に異なる点として、サーバーに送った情報がユーザーには見ることができない点が挙げられる. この特徴により POST メソッドは、主にデータを追加する際、今回のサンプルプログラムだとユーザーの仮登録、本登録をする際、個人情報などといった第三者に見られてはいけない情報を送信する際に用いられる.

メソッドの決め方としては、$this- type の値でメソッドが決まり、メソッドの内部では$this- action の値で処理を振り分けていく. また, 送信ボタンが二つある画面では, ボタンの名称を振り分けに利用している.

3.5 変数設計

3.6 各画面設計

各画面は QuickForm2、Smarty により、PHP ファイルをもとに自動生成されている.

画面描写に必要なファイルは smatry ディレクトリの templates ディレクトリ内にある.tpl ファイルである. これらのファイルを読み込むことで、画面を描写することができる.

3.6.1 ログイン画面

ログイン画面の描写には、login.tpl を用いている

index.php の情報を参照することでセッション管理を行なっていて、ログイン済みであると認証されなかった場合は、このページに遷移するようになっている.

また、入力欄に入力した内容をブラウザ側で保持しつつ出るために、$form.username.label には, ユーザー名を格納し、$form.username.html はユーザー名の入力欄、HTML のコードに置き換わる. 同様に、$form.password.label はパスワード名に、$form.password.html はパスワー ド入力欄に HTML のコードに置き換わる.

3.6.2 仮登録表示画面

画面の描写には premember.tpl が用いられる.

新規登録画面から確認画面でユーザーが登録情報を確認した後に遷移する画面であり、メールを送信した旨を伝えるための画面である.

3.6.3 会員専用画面

画面の描写には member_top.tpl が用いられる.

ログインを終えた会員と index.php にセッションが残っている場合は子も画面に遷移する.

3.6.4 管理者専用画面

画面の描写には system_login.tpl が用いられる.

管理者用のパスワードを入力し、認証を終えると、このページに遷移する. 会員一覧へのリンクは$SCRIPT_ NAME?type=list& action=form となり$ SCRIPT_ NAME は system.php に置き換わる。 クリックすると system.php に対して type=list と action=form が送信され、会員一覧を表示する。

4. 実装

4.1 実装環境

表 2 用いたツール名とそのバージョンの表

名前	バージョン
OS	macOS Venture バージョン 13.4
XAMPP	7.4.28
PHP	7.4.28

今回は以上の表のような環境で動作確認を行なった.

4.2 環境設定

今回のサンプルプログラムを動作させるために必要な XAMPP の環境構築は以下のように行なった.

(1) 公式サイトから XAMPP をインストールする
(2) 設定を行い、XAMPP コントロールパネルを起動させる
(3) Web サーバーである Apache を起動させる

といった流れになっている.

次に実際に今回のシステムを動作させるためのプログラムの配置は以下の手順で行なった

(1) 「環境構築から実践的なシステム作成まで完全習得! PHP7+MariaDB/MySQL マスターブック」に記されているサンプルデータにアクセスする
(2) そこにある Section77 の内容を引用するために、XAMPP ディレ クトリに php_libs を作成し、htdocs と php_libs に引用したサンプルプログラムをすべて移す. データの Section77 に入っているすべてのファイルを移す。
(3) コピペしたフォルダにあるファイルにアクセス権を付与する

MySQL の初期設定は以下の手順で行なった.

(1) sql フォルダを作成する
(2) サンプルプログラムの Section76、Section86 からにある SQL ファイルをコピーして、sql フォルダに格納する
(3) MySQL にアクセスし、予め、データベースやテーブル、初期ユーザーをインサートしておく

といった流れになっている.

今回のサンプルプログラムでは、メールを受け取って、本登録を完了する機能が実装されている. そこで、gmail のメール受け取りに関する設定が必要になったため、その手順を以下に記す.

(1) gmail の Web ページにアクセスする
(2) 右上の歯車のアイコンをクリックする
(3) 押すと、「全ての設定を表示」という項目が表示されるためそれをクリックする
(4) 「メール転送と POP/IMAP」をクリックする
(5) 「IMAP アクセス」のステータスを「IMAP を有効にする」に変更する

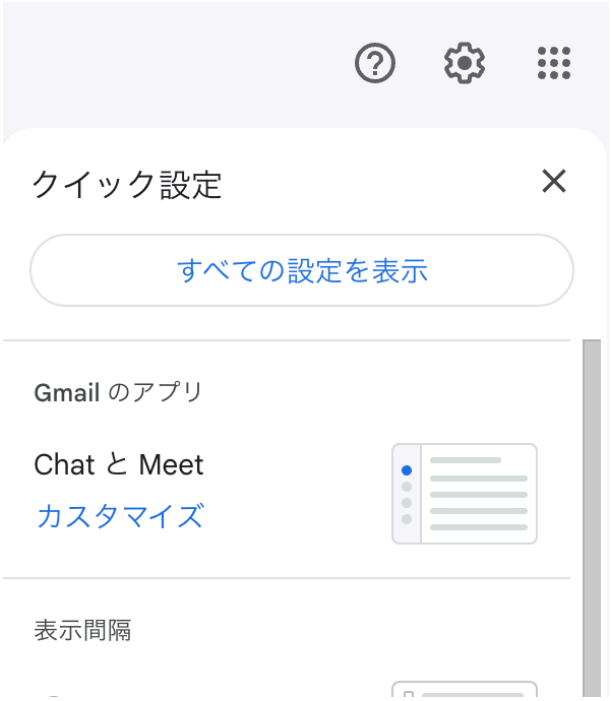

図 4 全ての設定を表示ボタンが表示されている画面の図

図 5 IMAP を有効にした時の画面図

以上の手順を踏むと、Gmail のコンソール画面のうち IMAP アクセス」のステータスを「IMAP を有効にする」に変更でき、IMAP サーバーから POP サーバーにメールの内容を送信できるようになる.

また、本来 Smatry を利用する際はインストールするまでに必要な手順を踏む必要があるが、サンプルプログラムを引用することでその必要は無くなったため、説明は省略することにする.

4.3 動作検証

画面遷移の説明の際に用いた図 2 の手順を踏んで行なった. まずは一般会員に関する動作検証を行なった.

(1) ログイン画面では自分の入力した内容がフォームにもそのまま反映されるかを検証した
(2) 新規登録画面ではログイン画面同様、フォームにもそのまま反映されるかに加えデータベースに格納されている都道府県情報が選択できているかを検証した
(3) 登録確認画面ではユーザーが新規登録画面で入力した値が premember テーブルに格納され、表示できているか検証
(4) メール確認後、本登録が完了した旨を伝える画面が表示されるかを検証
(5) 再びログイン画面に遷移した時、新規登録で登録したメールアドレス、パスワードでログインを行えるか検証
(6) 会員専用画面で、登録内容の修正、削除、ログアウトが行えるか検証
(7) 「IMAP アクセス」のステータスを「IMAP を有効にする」に変更する
(8) 「メール転送と POP/IMAP」をクリックする
(9) 「IMAP アクセス」のステータスを「IMAP を有効にする」に変更する

次には管理者に関する動作検証を行なった.

(1) ログイン画面で管理者用のパスワードを入力することで管理者用のページに遷移することができるかを検証した
(2) 管理者ページから会員一覧ページに遷移できるか検証
(3) 会員一覧ページでは、会員情報を全て閲覧できるようになっているか、またそれを削除、修正することができるかを検証
(4) メール確認後、本登録が完了した旨を伝える画面が表示されるかを検証
(5) 再びログイン画面に遷移した時、新規登録で登録したメールアドレス、パスワードでログインを行えるか検証

以上のように一般会員、管理者の動作検証を行ったところ、画面遷移や表示内容、処理について全く不具合が確認できなかったため、システムは正常に動作していると判断した.

5. まとめ

今回のサンプルプログラムでは PHP と MySQL を連携

させて作る簡単な会員登録アプリを作成した. 普段、個人またはチームでアプリ開発に取り組む際は、便利なライブラリに頼ったり、Firebase に認証機能を任せていたりしているためメール認証で使われる SMTP サーバー、POP サーバーについて深く知ることができたことはとても良い経験になった.

アプリ全体を通して、いつデータベースにアクセスして削除、追加、閲覧を行ったらよいのか、また、セッションとクッキーの管理、バリテーション、メール送信機能が実装されているため、基本的なアプリの流れを知ることができるアプリであった.

また、今回のアプリは新規登録を行なってログインをして、ログイン者専用ページを表示するといったとてもシンプルなものであったが、今後、会員専用ページに機能を拡張していくことでより面白いアプリにすることができる基盤となるアプリであるとも考えた.