1. websystem基礎実験 Express編

今回使用するリポジトリは以下のとおりです.

https://github.com/sudahiroshi/websystem2

1.1. Paiza Cloudの起動

新規サーバ作成時にNode.jsとMySQLをクリック（タップ）で選択しておいてください.

前回はMySQLを使ってデータの登録や一覧、変更、削除などを学習した。今回は、node.jsによる少し高度なWebサーバを構築する手法を学習する。

1.2. Expressとは？

Expressとは、node.jsのライブラリの1つで、開発グループによると「Node.js のための高速で、革新的な、最小限のWebフレームワーク」と説明されている。 フレームワークとは、ライブラリよりも簡単に目的を果たせるソフトウェアのことである。基本的には、mainに相当する部分を各自で書くのがライブラリで、main相当を準備してくれるのがフレームワークと思って良い。 前回使用した「mysql」はライブラリであったので、mainは各自で書いたはずである。 それに対してExpressは各自がmainを書く必要がない。（正確にはmain相当を書くこともできるが、書かない書き方もできる）

1.3. Expressのインストール

それではexpressをインストールする前に、今回はこのリポジトリをcloneしてから作業を開始しよう。

```
~$ git clone https://github.com/sudahiroshi/websystem2.git
~$ cd websystem2
~websystem2$
```

続いて、expressをインストールする。この作業は，動かすソースコードが置いてあるディレクトリで実行しなければならない.　（websystem2ディレクトリに移動していることを確認してから実行してください）

```
~websystem2$ npm install express
+ express@4.16.4
added 58 packages from 49 contributors and audited 135 packages in 2.726s
found 0 vulnerabilities

~websystem2$
```

1.4. Expressを利用した簡単なプログラム

それでは、Expressを使用した簡単なプログラムserver6.jsを見てみよう。

```
const express = require('express');
const server = express();

server.get('/', function( req, res ) {
    res.send( 'Hello, world' );
});

server.listen( 80, function() {
    console.log( 'listening on port 80' );
});
```

1行目でexpressを読み込んで、2行目でexpressを利用するための変数serverを用意している。 4〜6行目で、「/」に対するアクセスが来たらHello worldを返すように設定し、8〜10行目で、80番ポートで待ち受けをする。

それでは実行してWebブラウザからアクセスしてみよう。

```
~websystem2$ sudo node server6.js
listening on port 80
```

ここで、4〜6行目はURLとして「/」が指定された場合に、function以降を実行するという記述であり、すぐにこの部分が実行されるわけではない。 このような仕組みをJavaScriptではコールバック関数と呼ぶ。

ファイル名を指定した記述を増やすと、URLごとに異なる結果を返すことが可能となる。 例えば、7行目に以下のプログラムを加えると、URLの末尾にmorningが付いた場合に「Good morning, world」と返す。

```
server.get('/morning', function( req, res ) {
    res.send( 'Good morning, world' );
});
```

1.5. テンプレートエンジン

さて、前節のプログラムでExpressの基本的な使い方を理解したと思うので、一歩先に進めてテンプレートエンジンを使ってみよう。 テンプレートとはHTMLの雛形であり，これを使用するためのプログラムがテンプレートエンジンである． node.jsには多数のテンプレートエンジンが用意されている． 代表的なテンプレートエンジンを以下に示す．

名称	特徴
EJS	HTMLベースでとっつきやすい
Pub	Jadeが旧名称で，一時期よく使われていた
Hogan.js	Twitterが開発している
Mustache	様々なプログラム言語に対応している

名称	特徴
Handlebars	Mustacheの強化版

1.5.1. テンプレートの例

まずはEJSを使った簡単なテンプレートの例を示す。ここで、8行目の<%=から%>以外は、そのままWebブラウザに渡される。そして、8行目のtitleは、メインとなるプログラムから渡されたデータのうち、titleと命名された内容が表示される。

```
<!DOCTYPE html>
<html lang="ja"
<head>
<meta charset="UTF-8">
</head>
<body>

    <h1><%=title %></h1>

</body>
</html>
```

上記の雛形を使うプログラムserver6-2.jsを示す。2行めでテンプレートエンジンEJSを使用することを宣言し、5行目で拡張子ejsのファイルに対して、EJSを使うようににしている。 8行目で、テンプレートファイルindex.ejsを使用し、キーtitleに文字列Expressを代入している。この部分にはハッシュが使われている。この部分はハッシュなので、いくつでも記載可能であり、ここで付けたキー（今はtitle）がテンプレートに伝えられる。

```
const express = require('express');
const ejs = require('ejs');
const server = express();

server.set( 'ejs', ejs.renderFile );

server.get('/', function( req, res ) {
    res.render('index.ejs', {title: 'Express' });
});

server.listen( 80, () => {
    console.log( 'listening on port 80' );
});
```

上記のプログラムを動かすために，以下のようにejsパッケージをインストールする.

```
~/websystem2$ npm install ejs
+ ejs@2.6.1
updated 1 package and audited 135 packages in 1.06s
```

```
found 0 vulnerabilities

~/websystem2$
```

それでは，テンプレートエンジンejsを使うサーバプログラムserver6-2.jsを起動しよう．

```
~/websystem2$ sudo node server6-2.js
listening on port 80
```

Webブラウザでアクセスすると，大きめの文字で「Express」と表示されたはずである．

1.6. MySQLから取得したデータをWebで表示する

1.6.1. おさらい：簡単に演習用のMySQLを設定する

前回の資料に書きましたが，MySQLの設定を行うのはタイヘンなので，簡単に設定する方法を示します．
前提条件として，gitコマンドでwebsystem2をcloneしてあり，cdコマンドでディレクトリを移動していることが条件です．

```
~/websystem2$ sudo mysql < init.sql
~/websystem2$
```

1.6.2. テーブルを作る

前回使用した，都道府県の人口などのデータを使用します． 以下のようにして，テーブルをつくってください．これも前回のとおりです．

```
~/websystem2$ mysql -u node -pwebsystem web
mysql> create table example (
 id int auto_increment not null primary key,
 都道府県 varchar(100),
 人口 int,
 男性 int,
 女性 int,
 大学 int,
 国立大学 int,
 公立大学 int,
 私立大学 int,
 学生数 int,
 男子学生 int,
 女子学生 int );
Query OK, 0 rows affected (0.03 sec)

mysql> describe example;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
```

```
+--------------+--------------+------+-----+---------+----------------+
| id           | int(11)      | NO   | PRI | NULL    | auto_increment |
| 都道府県     | varchar(100) | YES  |     | NULL    |                |
| 人口         | int(11)      | YES  |     | NULL    |                |
| 男性         | int(11)      | YES  |     | NULL    |                |
| 女性         | int(11)      | YES  |     | NULL    |                |
| 大学         | int(11)      | YES  |     | NULL    |                |
| 国立大学     | int(11)      | YES  |     | NULL    |                |
| 公立大学     | int(11)      | YES  |     | NULL    |                |
| 私立大学     | int(11)      | YES  |     | NULL    |                |
| 学生数       | int(11)      | YES  |     | NULL    |                |
| 男子学生     | int(11)      | YES  |     | NULL    |                |
| 女子学生     | int(11)      | YES  |     | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
12 rows in set (0.02 sec)

mysql>
```

続けて，データをロードしよう．

```
mysql> load data local infile 'example.csv' into table example
fields terminated by ',' enclosed by '"'
(都道府県, 人口, 男性, 女性, 大学, 国立大学, 公立大学,私立大学, 学生数, 男子学生, 女子
学生 );
Query OK, 47 rows affected (0.04 sec)
Records: 47  Deleted: 0  Skipped: 0  Warnings: 0

mysql>
```

1.7. Express + EJS + MySQL

1.7.1. パッケージのインストール

すでにExpressとEJSを取得しているので，mysqlも取得します． mysqlからexitして，以下のコマンドを実行します．

```
~/websystem2$ npm install mysql
+ mysql@2.17.1
added 11 packages from 15 contributors in 1.594s
~/websystem2$
```

1.7.2. server7

Express + EJS + MySQLという構成のプログラムserver7.jsを以下に示します． 大まかな流れを表で示します．

行番号	内容

行番号	内容
1～4	ライブラリのロードとexpressの初期化
6～12	接続するMySQLの情報
14～21	MySQLからデータを読み込んで表示する（内訳は以下の通り）
14	Webブラウザからアクセスが来たら，以下の関数を実行
15	データを取得するSQL文
16～18	エラー処理
19	テンプレートファイルとして`sql.ejs`を使用する．取得したデータは変数rowsに入っている．

```
const express = require('express');
const server = express();
const ejs = require('ejs');
const mysql = require('mysql');

var connection = mysql.createConnection({
    host: 'localhost',
    port: 3306,
    user: 'node',
    password: 'websystem',
    database: 'web'
});

server.get('/', function( req, res ) {
    connection.query('select id, 都道府県, 人口 from example order by 人口
desc limit 10;', (error, rows, fields) => {
        if( error ) {
            console.log('Query Error');
        }
        res.render( 'sql.ejs', { content: rows });
    });
});

server.listen( 80, function() {
    console.log( 'listening on port 80' );
});
```

なお，テンプレート`sql.ejs`は以下の通り． このファイルを呼出した`server7.js`の20行目によると，全データがキー`content`に入れられている． そこで`sql.ejs`の8行目で，全データの1つ1つを取り出すループを実行している． 10～12行目で，result内の必要な項目からデータを取り出し，Webブラウザに返している．

```
<!DOCTYPE html>
<html lang=ja>
<head>
<meta charset="UTF-8">
</head>
<body>
    <table>
        <% for( let result of content ) { %>
        <tr>
            <td><%= result.id %></td>
            <td><%= result['都道府県'] %></td>
            <td><%= result['人口'] %></td>
        </tr>
        <% } %>
    </table>

</body>
</html>
```

以下のようにして実行して，Webブラウザで接続してみよう．

```
~/websystem2$ sudo node server7.js
```

県ごとの人口が多い順に10個の都道府県が表示される．

1.8. パラメータの取得

現在の`server7.js`では，静的なページと変わらない． ここでは，表示する都道府県数と人口区分をWebページから指定できるようにする．

指定できるようにするためには，いくつかやりかたがあるが，ここでは`sql.ejs`を変更して，`form`を追加する方法を取る．

これを考慮したテンプレート```sql2.ejs'''を以下に示す．

```
<!DOCTYPE html>
<html lang=ja>
<head>
<meta charset="UTF-8">
</head>
<body>
    <table>
        <% for( let result of content ) { %>
        <tr>
            <td><%= result.id %></td>
            <td><%= result['都道府県'] %></td>
            <td><%= result['人口'] %></td>
        </tr>
```

```
        <% } %>
    </table>
    <form action="/" method="get">
        並び順：<select name="sorting">
            <option value="人口">人口</option>
            <option value="男性">男性</option>
            <option value="女性">女性</option>
        </select>
        表示数：<input type="text" name="number" size="10">
        <input type="submit" value="並び替える">
    </form>
</body>
</html>
```

続いて，上記のテンプレートに合わせたサーバプログラムserver8.jsを以下に示す． 発行したSQLをコンソールに表示するべく，server7.jsと少しだけ構造を変えている． 変更された行を以下に示す．

行番号	内容
15,16	送られてきたformの内容を変数に代入．送られてこなかったら標準的な値を代入する．
17	変数を組み込んだSQL文の組み立て
23	使用するテンプレートの変更

```
const express = require('express');
const server = express();
const ejs = require('ejs');
const mysql = require('mysql');

var connection = mysql.createConnection({
    host: 'localhost',
    port: 3306,
    user: 'node',
    password: 'websystem',
    database: 'web'
});

server.get('/', function( req, res ) {
    let sorting = req.query.sorting || '人口';
    let number = req.query.number || 10;
    let query = 'select id, 都道府県, ' + sorting + ' from example order by
' + sorting + ' desc limit ' + number + ';';
    console.log( query );
    connection.query( query, (error, rows, fields) => {
        if( error ) {
            console.log('Query Error');
        }
        res.render( 'sql2.ejs', { content: rows });
    });
});
```

```
server.listen( 80, function() {
    console.log( 'listening on port 80' );
});
```