IERG4210 - Tutuorial 5

Benedict Mak

Handlebars - Basic

- Handlebars
  - Three elements - Template, control JS, Data

- Two ways to use Handlebars
  - Client side - Handlebars
    - Get data in the form of JSON from server
    - Populate the page with data according to the template
  - Server side - Express Handlebars
    - Get data directly from SQL server
    - Render the page with data according to the template and send to user

Handlebars - Basic (Client side)

- Handlebars can run without express
  - One template page

```
<script id="entry-template" type="text/x-handlebars-template">
<h1>{{title}}</h1>
    {{body}}
</script>
```

  - One Control JS (include handlebars and jquery library)

```
var source   = $("#entry-template").html();
var template = Handlebars.compile(source);
var data = {title: "My New Post", body: "This is my first post!"};
$('body').html(template(data));
```

http://jsfiddle.net/aytLeryb/

Handlebars - Basic (Server side)

- Let’s talk about handlebars in Nodejs
- Install express-handlebars

  npm install express3-handlebars --save
- Create the folder hierarchy ./views/layout
  - The default folder where handlebars look for your template page (extension .handlebars)
  - Can change later

Handlebars - Basic (Server side)

- In app.js

```
var express = require('express'),
    exphbs  = require('express3-handlebars'),
    app = express();


app.engine('hbs', exphbs({defaultLayout:'main', layoutsDir: 'views/', extname: '.hbs'}));
app.set('view engine', 'hbs');
app.get('/', function (req, res) {
    res.render('home', {layout: 'main'});
});
 app.listen(80);
```

http://runnable.com/VNd4Czx21R1sZDmV/express-handlebars-demo-for-node-js

Handlebars - Basic (Server side)

- app.engine('hbs', exphbs({defaultLayout: ‘main’, layoutsDir: 'views/', extname: ‘.hbs’}));
  - defaultLayout: ‘main’ - state what is your default layout
  - app.engine (‘hbs’) - need to consistent with your extension name
  - state where you store the layouts, and what is the extension name
  - Optional, default is ‘views/layouts/’, ‘.handlebars’

- app.get('/', function (req, res) { res.render('home', {layout: 'main'});});
  - ‘/’ - when user request this, execute the function
  - res.render(‘home’) - render file ‘home’ with the layout
  - layout: ‘main’ - Optional, state what is your layout file, override defaultLayout

Handlebars - Basic (Server side)

In views/main.hbs

```
<html> {{{body}}} </html>
```

{{{body}}} - placeholder for the main content to be rendered

{{{}}} - Don't escape the value inside those parentheses

Escape - turn "<" to "<"

In views/home.hbs

```
Some Content
```

http://runnable.com/VNd4Czx21R1sZDmV/express-handlebars-demo-for-node-js

Handlebars - Data

- Lets talk about how to use handlebars with data
- In the template/view page
  - Client side - the template html
  - Server side - home.hbs in our example
  - use {{ dataname }} as a placeholder for data named as “dataname”
- In JS
  - (Client Side)

```
var data = {dataname: "My New Post"};
$('body').html(template(data));
```

  - (Server Side)

```
app.get('/', function (req, res) {
    var data = {dataname: "My New Post"};
    res.render('home',  data); });
```

Handlebars - Helpers

- Help you to do something more in the template
- {{#if}}
  - If the condition is true, display the thing in the middle.
  - {{#if condition}} Display {{dataname}} {{/if}
- {{#each}}
  - {{#each items}} {{name}} {{emotion}} {{/each}}
  - Iterate the the rows in “items”

```
var data = {
  items: [
    {name: "Handlebars", emotion: "love"},
    {name: "Mustache", emotion: "enjoy"},
    {name: "Ember", emotion: "want to learn"}
  ]
};
```

Handlebars - Helpers

- You can register your own helper
  - Client Side

```
Handlebars.registerHelper('fullName', function(person) {
  return person.firstName + " " + person.lastName;
});
```

  - Server Side

```
app.get('/', function (req, res, next) {
    res.render('home', {
                helpers: {
            foo: function () { return 'foo.'; }
        }
    });
});
```

Handlebars - Helpers

- Define own helper seems very useful
  - Avoid do strings concatenation (e.g. `person.firstNam` `" + person.lastName`)
  - Hard for developers to apply output filtering
  - Also, dont use Handlebars.SafeString() to avoid HTML escape, very unsafe

Assignment 3a Hints

- Use mysql
- Install mysql
  - sudo apt-get install mysql-server (ubuntu)
  - yum install mysql-server mysql (red hat / amazon linux)
- Install mysql driver for nodejs
  - npm install mysql --save
- Login to mysql
  - mysql -u root -p
- Creata database and use
  - create database shop00;
  - use shop00;

Assignment 3a Hints

- Create table

```
CREATE TABLE categories (
catid INT(6) UNSIGNED unique PRIMARY KEY,
name VARCHAR(30) NOT NULL
);
```

  - There's a lot of other data types. Check it out.

- Security - Create another user, tight privilege is better, block access from other IPs except localhost
  - SELECT User, Host, Password FROM mysql.user;
    - It shows all the users and related host, delete all the non-localhost entries
  - CREATE USER 'dummy'@'localhost' IDENTIFIED BY 'mypass';
  - GRANT insert, select, update, delete ON shop00.* TO 'dummy'@'localhost';
    - Grant no administrative privilege
- After select which item to change, display all the sub-elements of that item first

```
var express = require('express'),
    app     = express(),
    mysql   = require('mysql'),
    connectionpool = mysql.createPool({
        host     : 'localhost',
        user     : 'root',
        password : '1234',
        database : 'shop00'
    });
app.get('/table', function(req,res){
    connectionpool.getConnection(function
(err, connection) {
        if (!err) {
            connection.query('select * from
categories',  function(err, rows) {
                if (!err) {
                    res.json(rows);
                }
                    connection.release();
            });
        }
    });
});
app.listen(8080);
```

This code will return a json response, pass the json to handlebars to show all categories or products.

Assignment 3a Hints

- How to store a picture?
  - Dont convert it into binary and store in DB
  - Use a folder to include all the photos, mark them by pid. And read it by their pid + ext (e.g 041.jpg)

- Never direct access DB from your client
- Create a REST API to do the DB work
  - http://www.nodewiz.biz/nodejs-rest-api-with-mysql-and-express/
  - Restify
  - Remember to do the authentication first