Faculty of Computer Science, Dalhousie University *10-Feb-2026*

DGIN 5201 — Digital Transformation

Lecture 9: Web Server

Location: McCain 2170 Instructor: Vlado Keselj
Time: 10:05–11:25

Previous Lecture

- Creating a simple web page (e1)
  - ssh, shell commands (bash),
  - file permissions
  - editors, emacs
- Printable page form (e2)

Review of e2 Form

- Let us take another look at the form in e2

Example e2: `public_html/dgin5201/e2/index.html`

```
<html><head><title>Conference Registration</title></head>
<body>
<h1>Conference Registration</h1>

<p>This is a registration page for CoMS.<br/>
For additional documents, please check <a
href="material">here</a>.<br/>
Please enter your information below to register:

<table>
<tr><th align=right>First and last name:</th>
<td>______________________________</td></tr>
<tr><th align=right>Email:</th>
<td>______________________________</td></tr>
<tr><th>Area of Interest (DB, HI, DS):</td>
<td>______________________________</td></tr>
</table>
```

Review of e2 Form

- Let us take another look at the form in e2
- And let us review two topics:
  - significance of touch typing
  - text editors, emacs

Aside: Touch Typing

- If you don’t use touch typing, consider learning it
- A relatively simple and not popular skill, but
  - actually important, and even more and more relevant
- Also known as *blind typing*, and *touch keyboarding*
- Reference: `https://en.wikipedia.org/wiki/Touch_typing`

Touch Typing

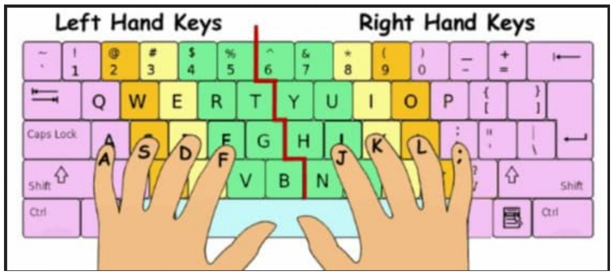

Image Source: `https://official-typing-test.com`

Editors, Emacs

- Need to be comfortable with a general text editor
- Some Emacs important Emacs commands:
  - `C-x C-s` — save
  - `C-x C-c` — quit
  - `C-z` — suspend to the command line
  - `fg` — go back from the command line
  - A useful tutorial: `https://www2.lib.uchicago.edu/keith/tcl-course/emacs-tutorial.html`

Web Server

- We continue with exercises
- Through `timberlea` we access files available to the web server
- Web server: Apache
  - picks up files and serves them to clients
  - allows us to customize and configure its behaviour
- We move to e3: “providing material”

Example e3: Available Material

Exercise e3 Overview

- We have a simple registration form on web
- It can be printed and filled manually
- There is a link to available ”material”
  - but no ”material” yet
  - Let us provide material

Example e3: Next Iteration of Our Site: Available Material

- Let us make a copy of our `e2` site
- First, go back to the directory above `e2`:

```
cd ~/public_html/dgin5201
```

- Use command `pwd` to check your directory
- Copy `e2` to `e3` as an exact copy:

```
rsync -av e2/ e3/
```

- Check the new site `e3` in the browser
- `rsync` is a very useful utility for copying directory structures
  - it works locally as well as over ssh
  - it copies incrementally differences, which is important if two sites are large and mostly equal
  - it may preserve permissions if we use option `-a`

Example 3: Make material available

- Create readable and accessible (‘executable’) directory `material` (permissions: `rwxr-xr-x`)
- Copy PDF from: `~vlado/public/dt-mini-conf.pdf` into directory `material`
- Setup permissions for the directory `material` to be all readable and accessible (`rwxr-xr-x`), and for the file `dt-mini-conf.pdf` to be all readable (`rw-r--r--`)
- Try to access material link on the page. Does it work? Why?

Example 3: Prepare `.htaccess` in `material` directory

- Prepare file `.htaccess` and make it all readable (`rw-r--r--`):

```
Options Indexes
```

- Check material access now
- Add the following line to `.htaccess` and try accessing again:

```
Options Indexes
AddDescription "DT Conference Poster (PDF)" dt-mini-conf.pdf
```

- Add “and Information” to “DT Conference Poster” and access
- Add the following line and try again:

```
Options Indexes
IndexOptions DescriptionWidth=*
AddDescription "DT Conference Poster..." dt-mini-conf.pdf
```

- `.htaccess` file is used to configure Apache web server behaviour
  - can be used to provide a simple password-protected access

Concepts Review: Example 3 (e3)

- Creating something that looks like form when printed
- HTML tags: `head, title, h1, p, br, a, table, tr, th, td`
- HTML attribute: `<th align=right>` `<a href="...">`
- `bash` shell: `cp`, using path, `~vlado`
- `rsync` command, `-av` options
- Accessing directory via browser
- `.htaccess` file for the Apache server: `Options Indexes, AddDescription`

Hands-on e4: Password Protection

Example e4: Next Iteration of Our Site: Password Protection

- User `rsync` again to make a copy of `e3` to `e4`
- Example 4 (e4) will be used to demonstrate password protection

Example 4: Simple Password Protection

- `cd` to `e4` directory and let us prepare a password
- In a locally-only readable file `pw` (`rw-------`) we can save a password for our reference: `dt dt5201`
- Prepare the password for the site using the command:

```
htpasswd -bc .htpasswd dt dt5201
```

- Make the file `.htpasswd` all-readable and check its contents
- Prepare the file `.htaccess` and make it all readable:

```
AuthType Basic
AuthName dgin5201
AuthUserFile /users/webhome/<your_csid>/dgin5201/e4/.htpasswd
AuthGroupFile /dev/null
<Limit GET POST>
require user dt
</Limit>
```

- Check that site is password-protected

Summary of e4

- Files and permissions copied from `e3`
- `pw` file with permissions `rw-------`
- `.htpasswd` file with permissions `rw-r--r--` and appropriate content set up with the `htpasswd` command
- `.htpasswd` file with permissions `rw-r--r--` and content set up for password protection as given in class

Concepts Review: Example 4

- `htpasswd` command, password saved as hash
- Using `.htaccess` for password-controlled access

Unix-style Customization

Slide notes:

Unix-style Customization

- Unix-style customization is typically text-based
- Example: bash customization
  - aliases: rm, mv, cp, em
  - .profile and .bashrc files
- Example: Emacs customization
  - .emacs file
- Earlier example: Apache customization
  - .htaccess, .htpasswd files

The Unix-style operating systems, such as Linux, mostly use text-based configuration in various scenarios, and it is useful to be able to edit those plain-text files to customize the system and different software utilities, and for solving different technical problems.

Bash shell customization: The Bash shell customization is one such example. For example, the command `rm` is used to remove a file. Its default behaviour is usually such that after a command such as `rm file1` it will immediatelly remove the file named `file1`. In practice, this may easily lead to a mistake of accidentally removing a file that we did not want to be removed. This can be fixed by creating an alias for the command `rm` in Bash using the command:

```
alias rm='rm -i'
```

This command would create the alias `rm`, which when typed into the shell causes the actual command '`rm -i`' to be executed, which uses the option `-i` of the program `rm` to always ask user for a confirmation if they really want to remove the given file.