NGINX Core - Module M09 Labs

**** The Labs Begin on Page 2 ****

PLEASE READ THESE SYSTEM USE INSTRUCTIONS:

The lab systems default to text mode, if you want to take advantage of copy-and-paste functionality, or need to use multiple **Terminal** applications, you'll need to start the GUI Desktop environment using the following instructions.

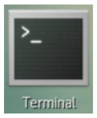
Steps to Prepare for a GUI Lab Environment:

1. Sign on to the system with the user credential **root** and password **training**.

2. Start the GUI desktop with the command:

```
$ startx
```

3. When the Graphical Desktop appears, click on the **Terminal** app icon on the left-side of the desktop to open the Terminal.



Note: You can resize the **Terminal** app to suit your preferences or to show more text output on the screen.

To Copy and Paste to a Lab System in GUI Mode:

1. Select the desired text from this lab guide and use **Cmd+c** (macOS) or **Ctrl-c** (Windows and Linux) to copy the text to the system clipboard.

2. Then click into the **Terminal** app in the lab system, and press Ctrl-Shift-V or select the Terminal application's **Edit** -> **Paste** menu item.

3. You may need to press it twice to insert the copied text.

Note: When pasting into **Vim**, you MUST be in **Insert** mode first, or the keystrokes will get mis-interpreted. If you get a strange `^M` or other character first, just delete it and re-paste again.

NOTE: The **commands** you should be typing, (or copying & pasting for convenience) are displayed like the below text; (don't type or copy the **$** prompt, just the `green` text):

```
$ command
```

The **output** from commands, or the file contents shown in commands like **vim** are displayed like this:

```
Line of output
Another line of output
```

Labs for Module M09 - Mapping Variables

Labs Included

- Lab 9.1 - Create a new Configuration File
- Lab 9.2 - Setup the map directive
- Lab 9.3 - Map with regex
- Lab 9.4 - Using if with a map

PLEASE READ!

You have been allocated **1 hour** for the labs in this section. Please leave the Virtual Lab Machine browser window/tab open after completing the first lab, then continue the section content. You will then be asked to complete the rest of the labs throughout the rest of the section.

If you let too much time lapse and the Virtual Lab Machine times out, simply restart it by clicking on the Launch button for the Lab and start again with the first lab.

Note: We have included configuration file samples in their entirety to make catching up with a particular lab easier. For example, if your lab times out and you were in a later lab, simply copy the configuration file contents into the appropriate file and continue with the next lab.

Lab 9.1: Create a New Configuration File

Learning Objectives

By the end of the lab you will be able to:

- Create a new configuration file to test mapping
- Test for results

Overview

In this exercise, you will setup a server context that sets up the /products location for use.

Steps

1. Rename the **`default.conf`** configuration file:

```
$ mv /etc/nginx/conf.d/default.{conf,bak}
```

2. Create a new configuration file named **`map_test.conf`**:

```
$ vim /etc/nginx/conf.d/map_test.conf
```

3. Create a server block in the **`map_test.conf`** file:

   Note: When in Vim, press **`i`** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New configuration is shown in **bold** text.

```
server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/map.access.log combined;
  error_log /var/log/nginx/map.error.log error;
}
```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

4. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

```
:wq
```

5. Reload your NGINX configuration with

```
$ nginx -s reload
```

End of Lab

Lab 9.2: Setup the map Directive

Learning Objectives

By the end of the lab you will be able to:

- Add a map directive that maps variables and returns a result
- Test for results

Overview

In this exercise, you will setup a server context that sets up the /products location for use.

Steps

1. Edit the **`map_test.conf`** file:

   ```
   $ vim /etc/nginx/conf.d/map_test.conf
   ```

2. Add a map directive in the **`map_test.conf`** file, keep the editor open and then add a location to the server context as shown below:

   Note: When in Vim, press **`i`** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New configuration is shown in **bold** text.

```
map $args $value {
  default "zero";
  1   "one";
  2   "two";
  3 "three";
}

server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/map.access.log combined;
  error_log /var/log/nginx/map.error.log error;
    location / {
    return 200 "the value of '$args' is $value\n";
    }
}
```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

3. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

   ```
   :wq
   ```

4. Reload your NGINX configuration with

   ```
   $ nginx -s reload
   ```

5. Test the your configuration results with:

   ```
   $ curl localhost/
   ```

```
the value of '' is zero
```

6. Test the your configuration results with:

```
$ curl localhost/?1
```

```
the value of '1' is one
```

7. Test the your configuration results with:

```
$ curl localhost/?2
```

```
the value of '2' is two
```

8. Test the your configuration results with:

```
$ curl localhost/?3
```

```
the value of '3' is three
```

9. Test the your configuration results with:

```
$ curl localhost/?32435
```

```
the value of '32435' is zero
```

   What happened to the last curl command and it's output? The secret is in the mapping of the numerals to the letters, and there being only one numeral. The longer value will not map since there are more characters in it.

End of Lab

Lab 9.3: Map with regex

Learning Objectives

By the end of the lab you will be able to:

- Modify the existing map directive to support numerals
- Test for results

Overview

In this exercise, you will be

Steps

1. Edit the **`map_test.conf`** file:

   ```
   $ vim /etc/nginx/conf.d/map_test.conf
   ```

2. Update the map directive in the **`map_test.conf`** file:

   Note: When in Vim, press **`i`** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   Replaced configuration is shown in **bold** text.

```
map $args $value {
  default 0;
  ~*^/?(\d+)$ $1;
}

server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/map.access.log combined;
  error_log /var/log/nginx/map.error.log error;
    location / {
    return 200 "the value of '$args' is $value\n";
    }
}
```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

3. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

   ```
   :wq
   ```

4. Reload your NGINX configuration with

   ```
   $ nginx -s reload
   ```

5. Test the your configuration results with:

```
$ curl localhost/
```

```
the value of ' is 0
```

6. Test the your configuration results with:

```
$ curl localhost/?1
```

```
the value of '1' is 1
```

7. Test the your configuration results with:

```
$ curl localhost/?2
```

```
the value of '2' is 2
```

8. Test the your configuration results with:

```
$ curl localhost/?3
```

```
the value of '3' is 3
```

9. Test the your configuration results with:

```
$ curl localhost/?32435
```

```
the value of '32435' is 32435
```

What has changed? The values are now represented back in numerals rather than in words, and the values for /?3 and beyond are now properly represented. Also take note that the first curl command returns a 0 instead of a blank.

End of Lab

Lab 9.4: Using if with a map

Learning Objectives

By the end of the lab you will be able to:

- Add a product image to a product html file

- Note any access errors that occur
- Use the rewrite directive to setup access
- Test for results

Overview

In this exercise, you will add an image to a product file, test to see if there is proper access to the file, and then use a rewrite directive to properly give access to the image so the product file displays it.

Steps

1. Edit the **`map_test.conf`** file:

   ```
   $ vim /etc/nginx/conf.d/map_test.conf
   ```

2. Update the map directive in the **`map_test.conf`** file:

   Note: When in Vim, press **`i`** to enter insert mode, and unless you are typing the server block manually, refer to the copy-and-paste instructions on the first page of the lab.

   New configuration is shown in **bold** text.

```
map $args $value {
  default 0;
  ~*^/?(\d+)$ $1;
}

server {
  listen 80 default_server;
  root /usr/share/nginx/html;
  server_name localhost;
  access_log /var/log/nginx/map.access.log combined;
  error_log /var/log/nginx/map.error.log error;
  if ($value ~ "^3+") {
    return 403 $value;
  }
    location / {
    return 200 "the value of '$args' is $value\n";
    }
}
```

   Note: Closely inspect the file contents if you used the copy-and-paste function. You should indent the lines as shown in the example, additionally there may be other formatting issues to fix.

3. When done adding the content to the file, quit the editor by first pressing the **ESC** key and then typing:

```
:wq
```

4. Reload your NGINX configuration with

```
$ nginx -s reload
```

5. Test the your configuration results with:

```
$ curl localhost/
```

```
HTTP/1.1 200 OK
Server: nginx/1.17.6
Date: Fri, 15 Apr 2022 15:56:18 GMT
Content-Type: application/octet-stream
Content-Length: 21
Connection: keep-alive

the value of '' is 0
```

6. Test the your configuration results with:

```
$ curl localhost/?1
```

```
HTTP/1.1 200 OK
Server: nginx/1.17.6
Date: Fri, 15 Apr 2022 15:56:18 GMT
Content-Type: application/octet-stream
Content-Length: 21
Connection: keep-alive

the value of '1' is 1
```

7. Test the your configuration results with:

```
$ curl localhost/?2
```

```
HTTP/1.1 200 OK
Server: nginx/1.17.6
Date: Fri, 15 Apr 2022 15:56:18 GMT
Content-Type: application/octet-stream
Content-Length: 21
Connection: keep-alive

the value of '2' is 2
```

8. Further test your configuration with:

```
$ curl http://localhost/?3
```

```
HTTP/1.1 200 OK
Server: nginx/1.17.6
Date: Fri, 15 Apr 2022 15:56:18 GMT
Content-Type: application/octet-stream
Content-Length: 21
Connection: keep-alive

3
```

The response to this query shows an error occurred.

9. Test the your configuration results with:

```
$ curl localhost/?32435
```

```
HTTP/1.1 403 Forbidden
Server: nginx/1.17.6
Date: Fri, 15 Apr 2022 16:09:01 GMT
Content-Type: application/octet-stream
Content-Length: 5
Connection: keep-alive

32435
```

The response to this query shows an error occurred.

End of Lab