

Development Setup

lab #lab01

James L. Parry
B.C. Institute of Technology

Lab Goals

The purpose of this lab is to introduce you to your development environment.

There are three tutorials to complete, and then this lab exercise to apply them.

This lab will also introduce you to github. That might be a separate tutorial in the future :-/

Suggestion: you may want to skim the slideshow first, before working your way through it.

Lab Submission

Your lab will be completed in a github repository.

Dayschool students: submit a readme to the lab dropbox. This readme should have a link to your github repository.

Distance students: send me an email with a link to your github repository.

Due: Sunday, Jan 11, 17:30 PST

Lab Marking Guideline

This lab will be marked out of 10 (no specific breakdown). It is more a matter of docking marks for boobos.

I will be making sure that you have a NetBeans project, with the proper webapp structure.

I expect to see comments in your source code!

I will check your github commit history, to see if you appear to using the process properly.

I will be making sure that it runs properly when I deploy the project into my "comp4711" virtual host folder.

Lab Preparation

Three tutorials have been provided, to setup your development environment. Please make sure you have completed them.

- [Tutorial 01a](#) XAMPP Installation
- [Tutorial 01b](#) Virtual Hosting Setup
- [Tutorial 01c](#) PHP Crash Course

Disclaimer

We will not necessarily be following "best practices" throughout this lab.

Our focus is on "baby steps", that will lead to best practices over the next few labs.

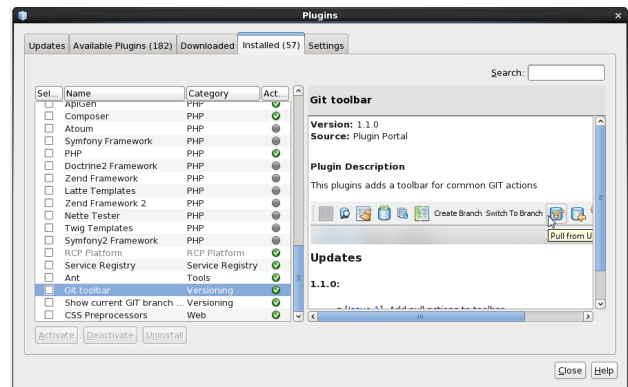
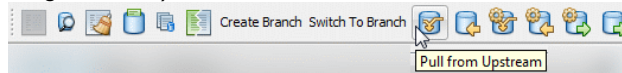
Install Git

Make sure that you have Git installed on your system, from the [git-scm](http://git-scm.com) site.



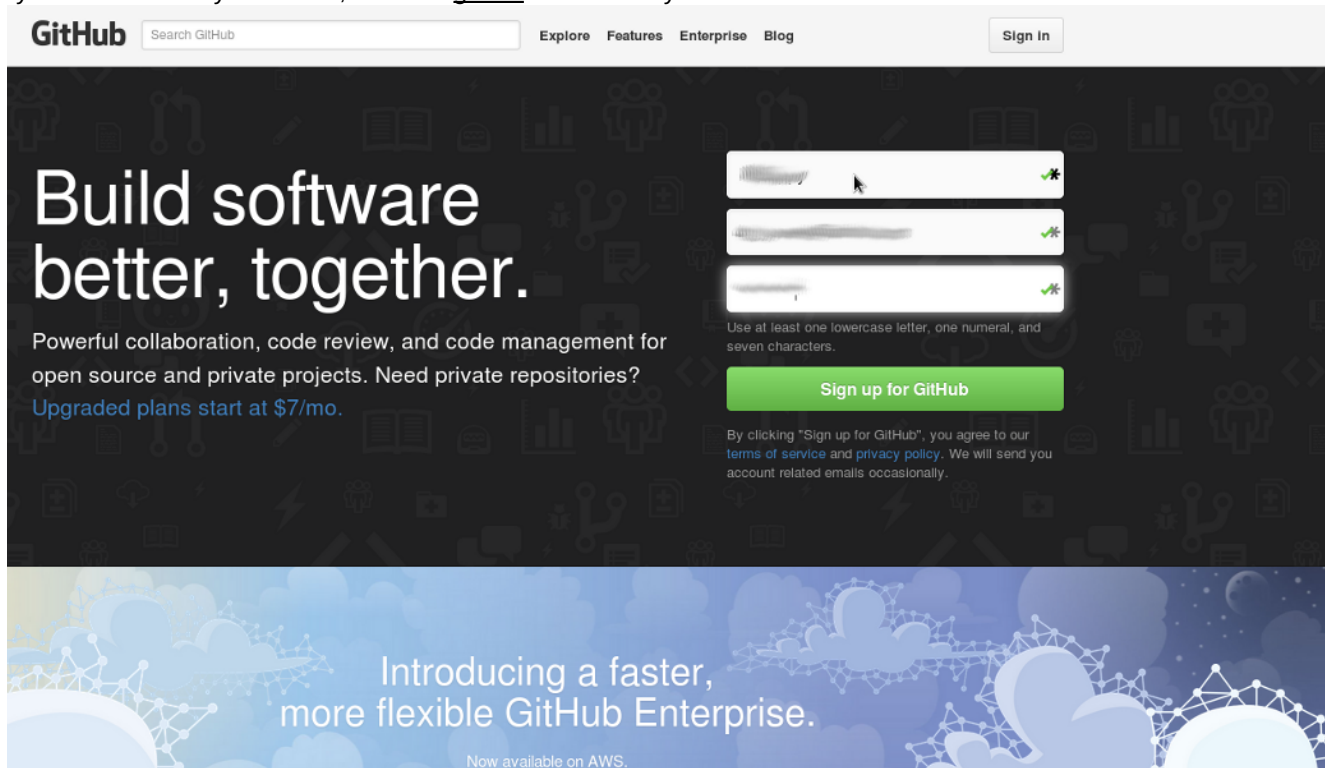
Install Git Toolbar

Also install the Git toolbar for NetBeans. It will make things easier :)



Create Your Github Account

If you do not already have one, create a [github](#) account for yourself.



Create an Empty Repository

Create a new repository to use for this lab. Name it what you like ... your virtual host mapping will need to refer to this name (as a folder) once you have cloned your repository locally.

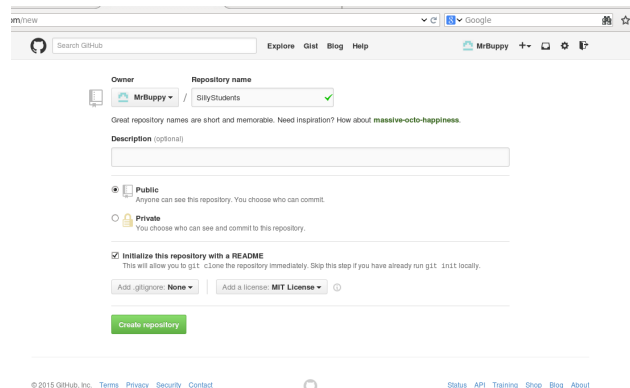
For a description, I suggest something like "COMP4711 lab 1". I left mine blank :(

If you want your repository to be private, you will have to upgrade your account to a paying one. BCIT does not cover those costs.

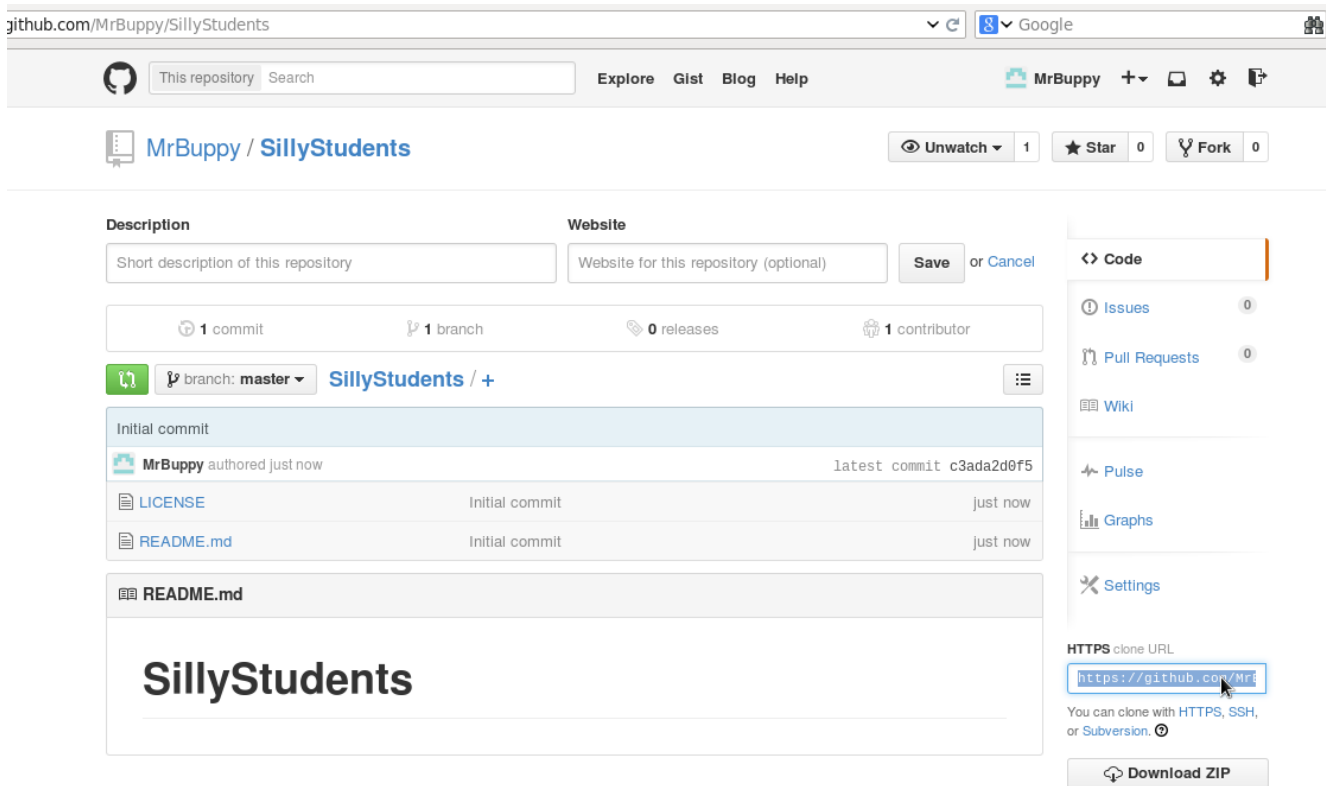
Do initialize the repository with a readme file. Changing this will give us an excuse to make sure our IDE is properly configured.

If we were working in a team at this point, it would make sense to have a gitignore directive, but we don't need it now.

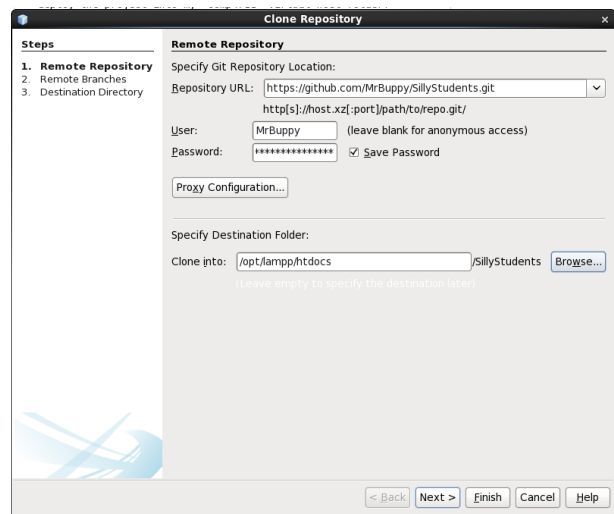
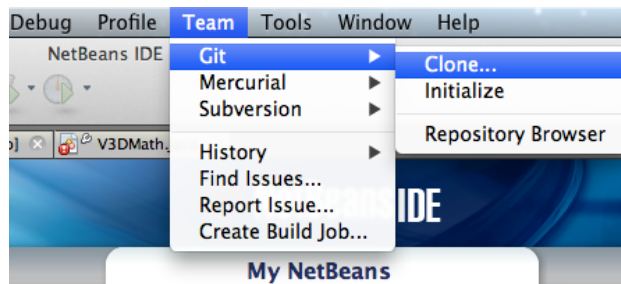
Choose a suitable license for your project. Ask me if you have any questions about which one is appropriate.



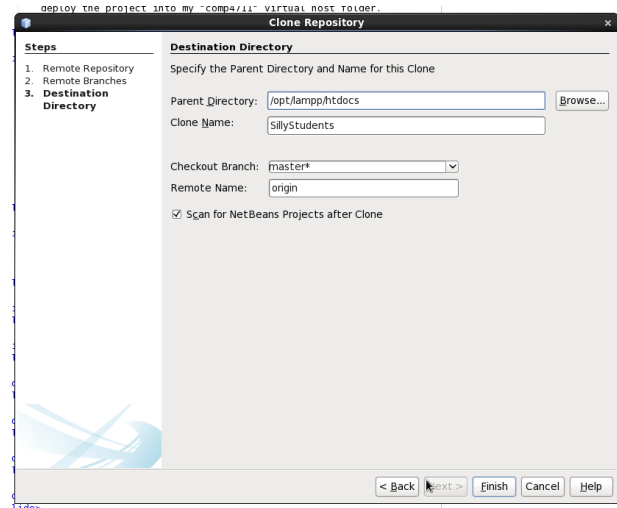
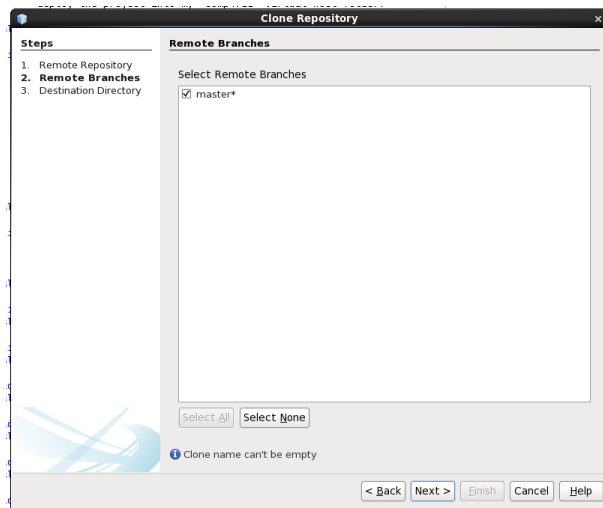
Verify Your New Repository



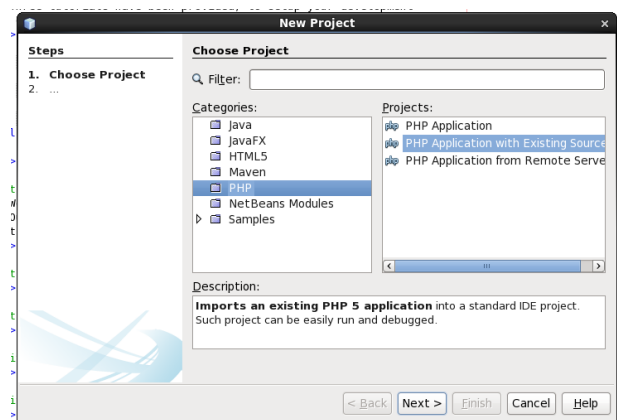
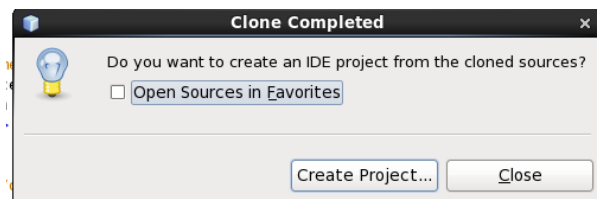
Clone Your Repository Locally



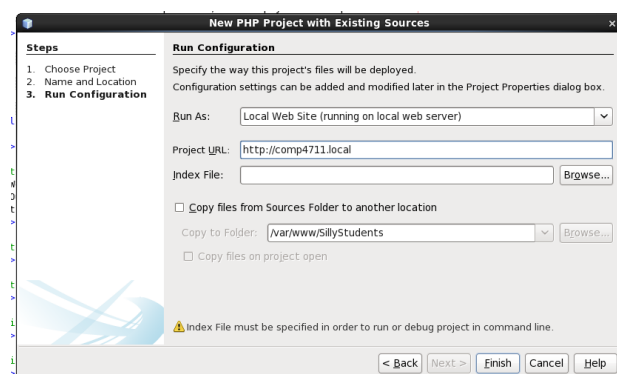
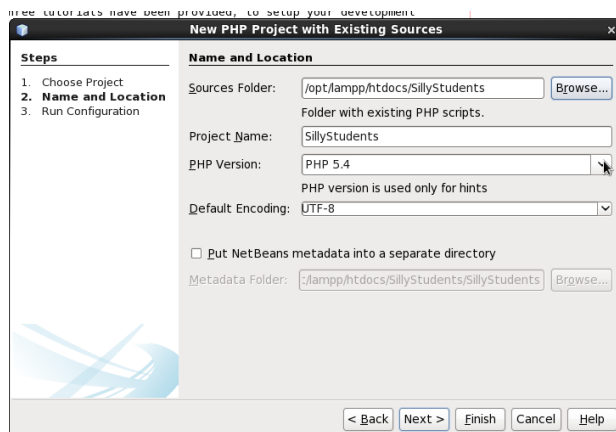
Local clone - Part 2



Local clone - Part 3

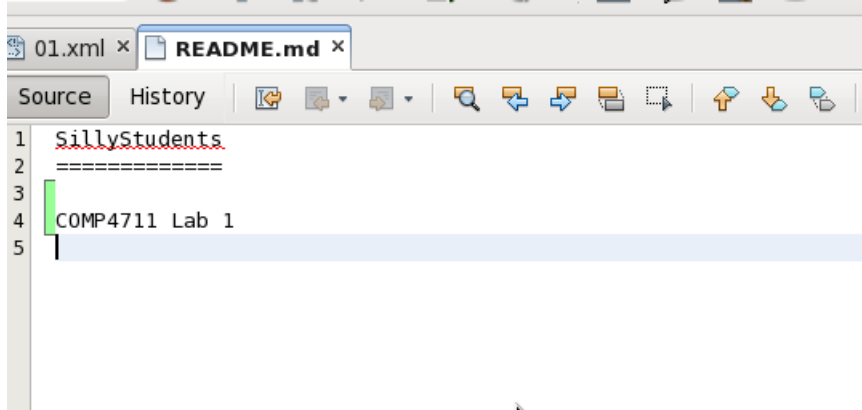


Local clone - Part 4

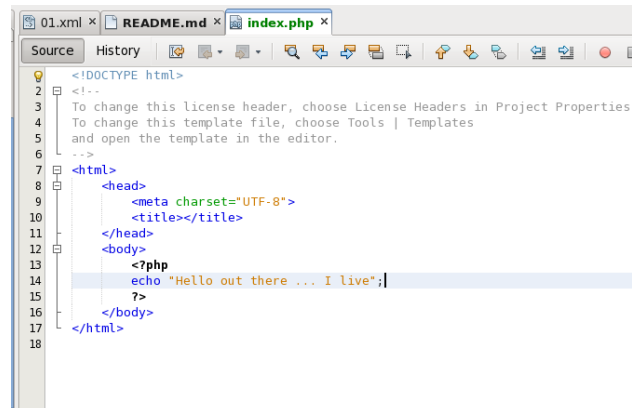
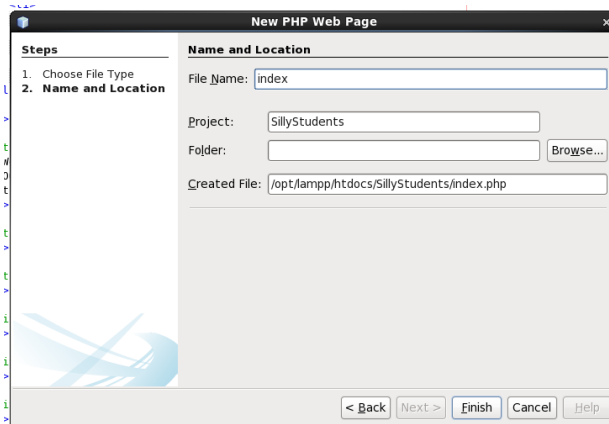


Update Your Repo Readme

Make a small change to your readme file. This is only so we can make sure our git access is properly configured.



Add a Homepage

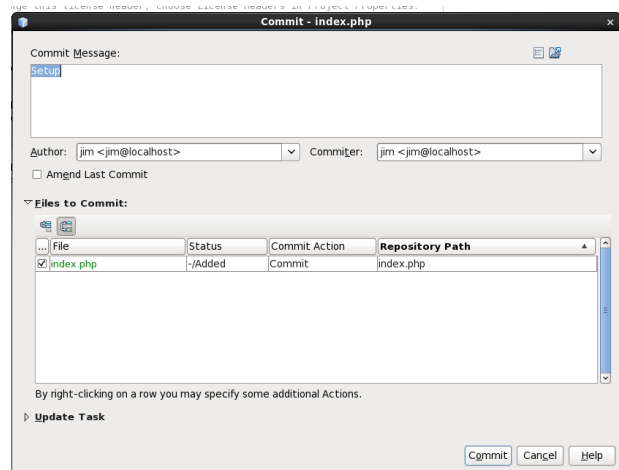
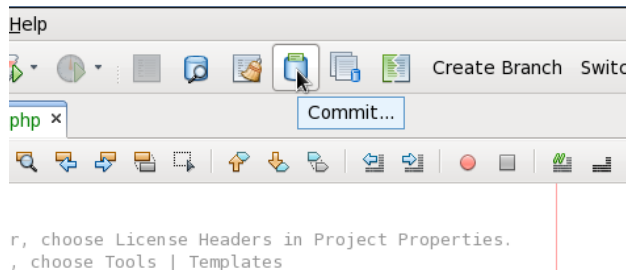


Make Sure the App Runs

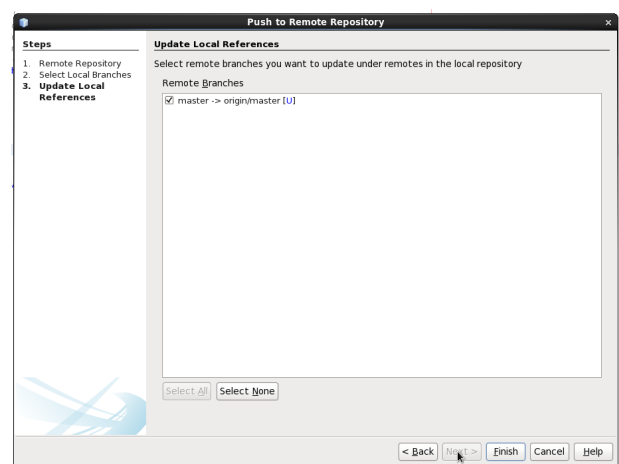
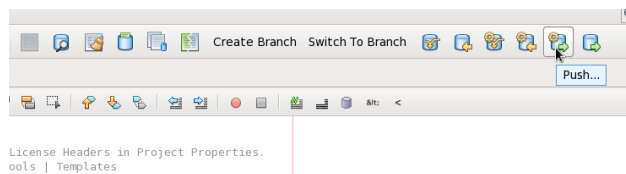
```
<VirtualHost *:80>
    ServerAdmin webmaster@dummy-host.example.com
    DocumentRoot "/opt/lampp/htdocs/SillyStudents"
    ServerName comp4711.local
    ErrorLog "logs/comp4711-error_log"
    CustomLog "logs/comp4711-access_log" common
</VirtualHost>
```



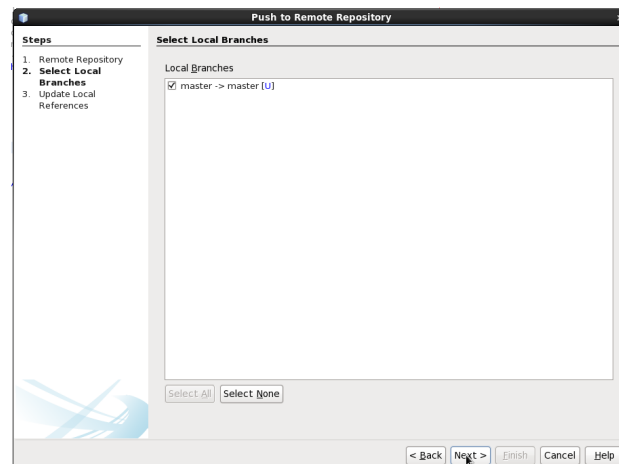
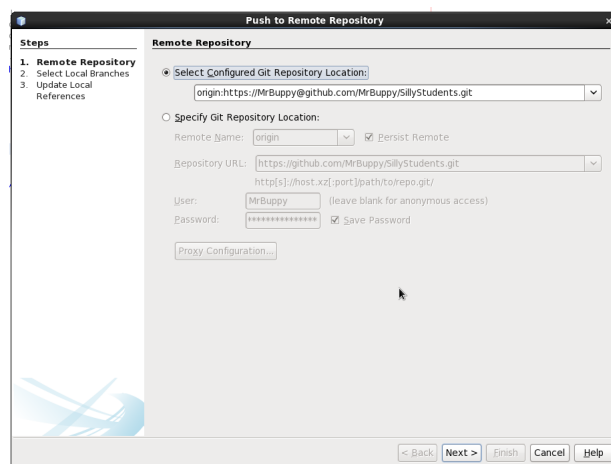
Commit Your Change



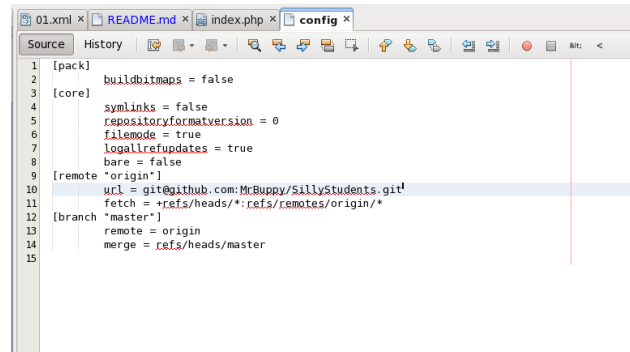
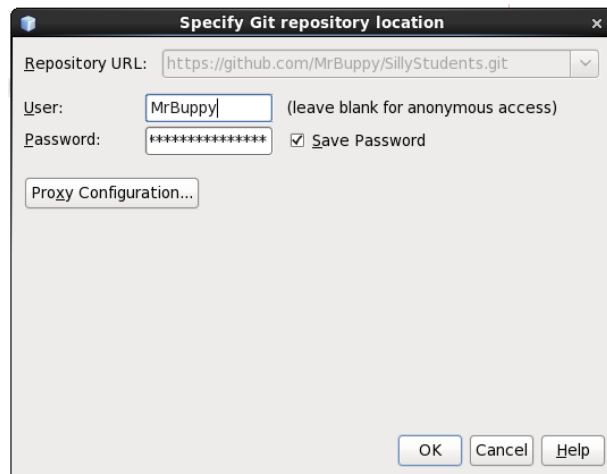
Update Your Repo



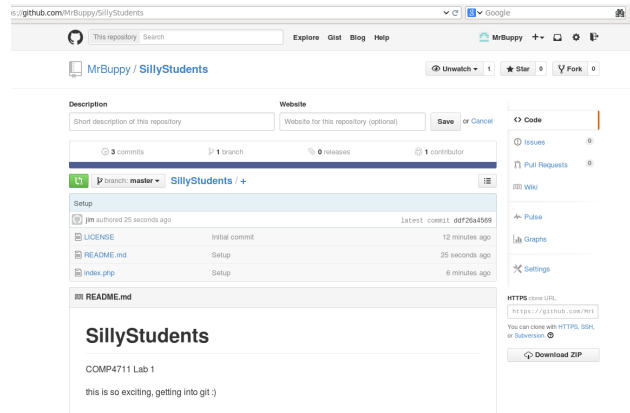
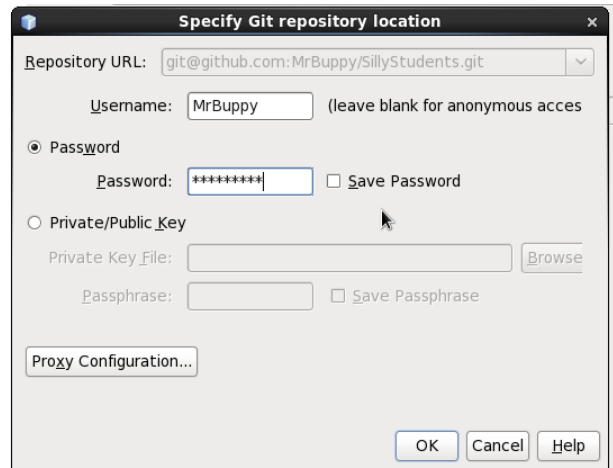
Repo Update - Part 2



Repo Update - Part 3



Repo Update - Part 4



Webapp Plan

We're going to make a student class, with some properties (surname, first name, a set of email addresses, and a set of grades) and some methods (report contact info, calculate grades average).

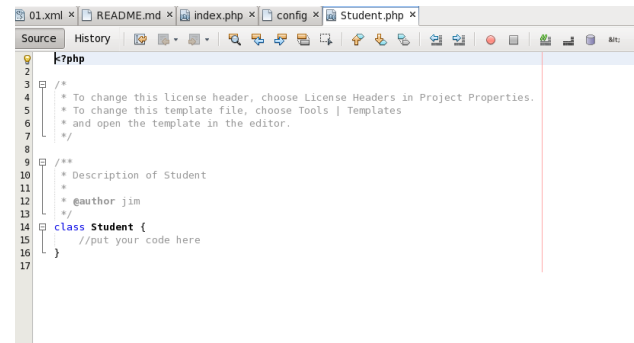
We will instantiate a few of these, and put them into an associative array that we then order and report.

Create Your Student Class

Let's start with the student class.

Make a new PHP class, Student.

Remove the closing PHP tag.

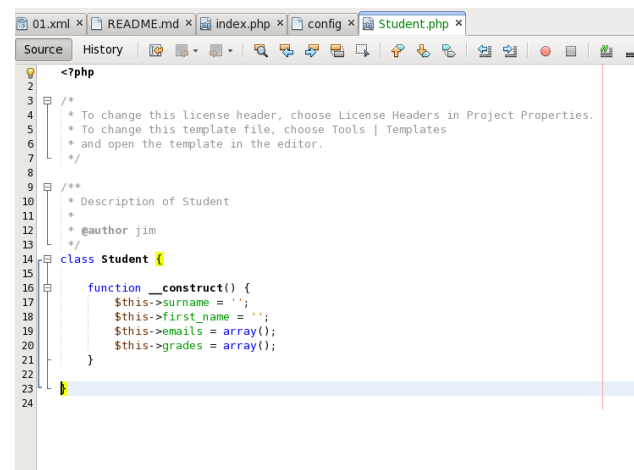


```
1 <?php
2
3
4 /*
5  * To change this license header, choose License Headers in Project Properties.
6  * To change this template file, choose Tools | Templates
7  * and open the template in the editor.
8  */
9
10 /**
11  * Description of Student
12  *
13  * @author jim
14  */
15 class Student {
16     //put your code here
17 }
```

Add a Constructor

Add a constructor, wherein we initialize the properties...

```
function __construct() {
    $this->surname = '';
    $this->first_name = '';
    $this->emails = array();
    $this->grades = array();
}
```



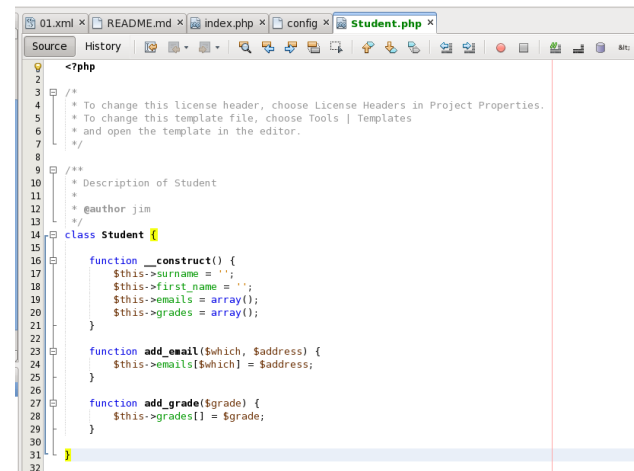
```
1 <?php
2
3
4 /*
5  * To change this license header, choose License Headers in Project Properties.
6  * To change this template file, choose Tools | Templates
7  * and open the template in the editor.
8  */
9
10 /**
11  * Description of Student
12  *
13  * @author jim
14  */
15 class Student {
16
17     function __construct() {
18         $this->surname = '';
19         $this->first_name = '';
20         $this->emails = array();
21         $this->grades = array();
22     }
23
24 }
```

Add Some Mutator Methods

Let's add a couple of mutator methods to add an email address or a grade...

```
function add_email($which,$address) {
    $this->emails[$which] = $address;
}
```

```
function add_grade($grade) {
    $this->grades[] = $grade;
}
```



```
1 <?php
2
3
4 /*
5  * To change this license header, choose License Headers in Project Properties.
6  * To change this template file, choose Tools | Templates
7  * and open the template in the editor.
8  */
9
10 /**
11  * Description of Student
12  *
13  * @author jim
14  */
15 class Student {
16
17     function __construct() {
18         $this->surname = '';
19         $this->first_name = '';
20         $this->emails = array();
21         $this->grades = array();
22     }
23
24     function add_email($which, $address) {
25         $this->emails[$which] = $address;
26     }
27
28     function add_grade($grade) {
29         $this->grades[] = $grade;
30     }
31
32 }
```

Calculate a Grades Average

We need to calculate the grades average...

```
function average() {
$total = 0;
foreach ($this->grades as $value)
$total += $value;
return $total / count($this->grades);
}
```

```

5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
*/
/**
 * Description of Student
 *
 * @author jim
 */
class Student {

    function __construct() {
        $this->surname = '';
        $this->first_name = '';
        $this->emails = array();
        $this->grades = array();
    }

    function add_email($which, $address) {
        $this->emails[$which] = $address;
    }

    function add_grade($grade) {
        $this->grades[] = $grade;
    }

    function average() {
        $total = 0;
        foreach ($this->grades as $value)
            $total += $value;
        return $total / count($this->grades);
    }
}
```

Add a Text Representation

We want to build a fancy text representation for reporting...

```
function toString() {
$result = $this->first_name . ' ' .
$this->surname;
$result .= ' ('.$this->average().")\n";
foreach($this->emails as $which=>$what)
$result .= $which . ': ' . $what. "\n";
$result .= "\n";
return '<pre>'.$result.'</pre>';
}
```

```

01.xml x | README.md x | index.php x | config x | Student.php x
Source History
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
*/
/**
 * Description of Student
 *
 * @author jim
 */
class Student {

    function __construct() {
        $this->surname = '';
        $this->first_name = '';
        $this->emails = array();
        $this->grades = array();
    }

    function add_email($which, $address) {
        $this->emails[$which] = $address;
    }

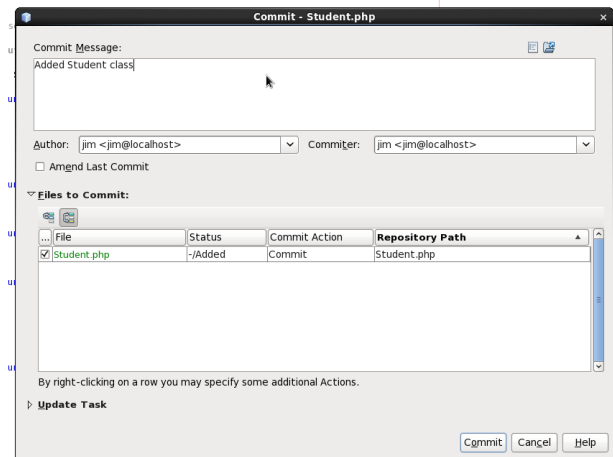
    function add_grade($grade) {
        $this->grades[] = $grade;
    }

    function average() {
        $total = 0;
        foreach ($this->grades as $value)
            $total += $value;
        return $total / count($this->grades);
    }

    function toString() {
        $result = $this->first_name . ' ' . $this->surname;
        $result .= ' ('.$this->average().")\n";
        foreach ($this->emails as $which => $what)
            $result .= $which . ': ' . $what. "\n";
        $result .= "\n";
        return '<pre>'.$result.'</pre>';
    }
}
```

Works? Update the Repo

That should do for a start. You might have to fix it a bit :-/



Back to the Homepage

Back to the main page, index.php.

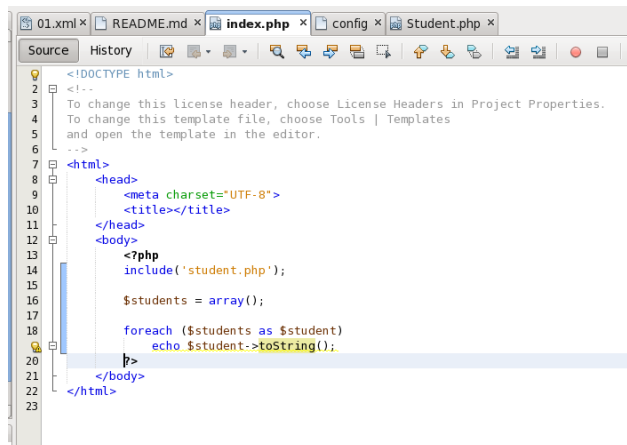
The general idea...

```
<?php
include('student.php');

$students = array();

foreach($students as $student)
echo $student->toString();

?>
```



Add Your First Student

```
$first = new Student();
$first->surname = "Doe";
$first->first_name = "John";
$first->add_email('home', 'john@doe.com');
$first->add_email('work', 'jdoe@mcdonalds.com');
$first->add_grade(65);
$first->add_grade(75);
$first->add_grade(55);
$students['j123'] = $first;
```

Add Your Second Student

```
$second = new Student();  
$second->surname = "Einstein";  
$second->first_name = "Albert";  
$second->add_email('home', 'albert@braniacs.com');  
$second->add_email('work1', 'a_einstein@bcit.ca');  
$second->add_email('work2', 'albert@physics.mit.edu');  
$second->add_grade(95);  
$second->add_grade(80);  
$second->add_grade(50);  
$students['a456'] = $second;
```

Works? Update Your Repo

Are you getting the expected result?

If so, update your repo.

If not, fix it! You should only commit and update your repo with working code.

Improve The Student Order

```
ksort($students); // one of the many sort functions
```

Works? You Know The Drill

If you run your "webapp" now, you should get a list of the two students, ordered by key (Albert then John). The output should look something like ...

```
Albert Einstein (75)  
home: albert@braniacs.com  
work1: a_einstein@bcit.ca  
work2: albert@physics.mit.edu
```

```
John Doe (65)  
home: john@doe.com  
work: jdoe@mcdonalds.com
```

Make It Look Better

You might have to add a bit of logic to get the students to show with readable spacing.

Add Yourself

Go ahead, add a third student (yourself) to the mix. In my case, this might result in the following output...

```
Jim Parry (85) work: jim_parry@bcit.ca
```

Are We There Yet?

Although not expected, you are welcome to play with the output produced, so it looks better :)

I shouldn't have to remind you to comment your code, right? We're all professionals or aspiring ones!

Congratulations!

You have completed lab #lab01: Development Setup

If you would take a minute to provide some feedback, we would appreciate it!

The next activity in sequence is: lesson02 MVC Framework Introduction

You can use your browser's back button to return to the page you were on before starting this activity, or you can jump directly to the course homepage, organizer, or reference page.