
Test-Doc Documentation

Release 0.1.0

at15

May 30, 2017

Contents

1	README	3
1.1	How to write doc	3
1.2	Syntax quick guide	3
2	Web develop courses	5
2.1	2016 Spring	5
2.1.1	Web Development Environment Setup	5
2.1.2	Git & HTML	23
2.1.3	Instant PHP	34
2.1.4	HTTP & RESTful API	37
3	Mos courses	39
3.1	2017 Spring	39
3.1.1	Docker	39
4	Indices and tables	41

Contents:

Dongyue Web Studio course and lecture

- [View Documentation](#)
- [View Course website](#)

How to write doc

- have python and pip installed. `python`
- `pip install sphinx sphinx-autobuild` to install sphinx
- `pip install recommonmark` to install markdown support
- use md or rst file
- run `make html` to build html

Syntax quick guide

- [Markdown Guide from GitHub](#)
- [Sphinx tutorial](#)
- [Sphinx common markups](#)
- [Sphinx example Readthedocs repo](#)

Web develop courses

Contents:

2016 Spring

Courses for 2016 Spring, the last semester in Rotunda

Contents:

Web Development Environment Setup

This course mainly tells you how to setup a web develop environment

Contents:

Environment

You write and run your code on certain machine. It contains two parts, your code, and the execution environment of your code. Like your *index.php* and the *LNMP* stack. see refernece for formal explanation.

- **local**: your laptop, PC.
- **remote**: server, bare metal, vps, ec2 etc.

Simplest: Develop -> Production

You write code in your local machine, they deploy it to remote machine, which is the production environment.

NOTE: Not all local environment is develop environment, you can use your own laptop to publish your works, DONT do this in SJTU, you will get a phone call. Not all remote environment is production environment, you can use a server to write code, like you use vim or you sync code to server everytime you change something.

Problems

- You use Windows, but the server is running Ubuntu.
- You use PHP5.6, but the server uses PHP5.3.
- You use Chinese as local (GB2312), but the server uses English (UTF-8).

You may have encounter similar problems when your are installing games or softwares, they cannot be installed to path with Chinese or space, and the text are displayed as strange marks.

Solution

Use the exact same environment for your laptop and server.

But what if:

- You want to use Windows to play Dota. (BGM: I want to play some dota ...)
- You can't install dual system.
- You have a small team working on the project, you can't force everyone to install Ubuntu.

Formal: Develop -> Test -> Production

You write you code locally, push to Git host, trigger test in CI (Continuous Integration), test pass, you deploy it to production. When you are working like this, you are highly likely not alone (but still single), either you are working on some opensource projects or you are working with a team (which has no girls). DONT use this for your homework.

Problems

- Test works fine locally, but breaks in CI.
- CI pass, but the production system is down.
- CI pass, production works, your local environment can't pass test.

Solution

- Have detail guide for environment setup.
- Use the exact same environment for your laptop, CI server, Production server.

But what if:

- Someone forgot to write the guide.
- Forgot to update CI or Production server. Worse if only production server is not up to date.

Complex: Develop -> Test -> QA -> Part of Production -> Production

tl;dw

How to setup a unfamiliar environment

- DONT be afraid.
- Backup your important things if you are NOT using virtual machine.
- READ the offical documentation before google.
- Google for stackoverflow answers.
- Read blogs
- Ask others
- Give up and use docker

this kind of work is included in a position called [Dev Ops](#)

Future of Develop Environment

- Cloud develop environment, ie: [Koding](#), [c9.io](#), [Eclipse Che](#)
- Hybird, ie: [Koding](#), [Coding.net](#)
- Docker

Reference

- [Wiki Deploy Environment](#)
- [Dev Ops](#)

Workspace Setup

How to orgnaize all your projects

Problems

tl;dr

- version control and sync
- name collision, ie: fork
- too many projects, ie: [gaocegege](#)

The long story

(This is not real)

Version Control and Sync

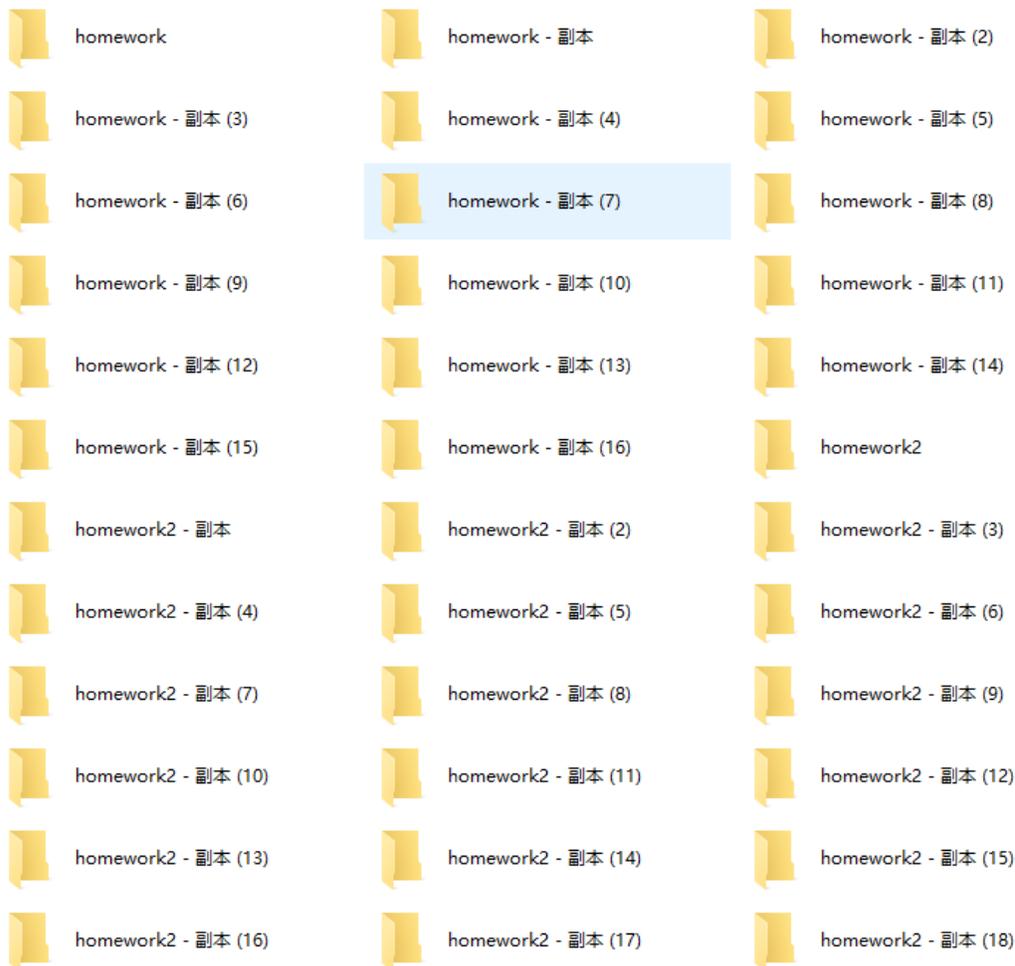
Jack is working on his homework, he want to try a new algorithm, so he copied a new folder.

```
D: /Code/  
  -- homework  
  -- homework-2
```

And Jack asked Mike to check his code, he send the code via email(QQ), and put Mike's modification in a new folder.

```
D: /Code/  
  -- homework  
  -- homework-2  
  -- homework-3  
  . . . . .
```

And he got 10 homework and 4 labs for this course, boom! (No picture no BB)



Jack got a laptop and a PC, he have to copy all the code in flash disk in order to sync, and sometimes he forgot it. Jack want to go back to his old code, he want to add tag for certain code, like `fix-memory-leak`, also he needs to work with Mike.

That's why Git, GitHub shows up.

Git is a version control system, and GitHub is the most popular Git repository host, free for opensource projects.

Name collision

Jack forked a project, [gaocegege's scrala](#)

```
D:/Code/  
-- scrala
```

But he wants to have a local copy for the original repo, so he have to rename it.

```
D:/Code/  
-- scrala  
-- gaocegege-scrala
```

He wrote a scarala that can only be for his lab project

```
D:/Code/  
-- scrala  
-- gaocegege-scrala  
-- lab-scrala
```

He is tired of renaming the repo.

Too many projects

Jack got homework, lab, outsource, opensource, for gf, for goddess's bf etc.

Solution

tl;dr

- use git
- tree
- [the Go way](#)

Put your folders in a tree

You have many source for your projects, homework, lab, outsource, opensource, for gf, for goddess's bf etc. You can have workspace like the following

```
D:/Code/  
-- homework  
  -- 2016-Spring  
    -- environment  
-- lab  
  -- cit  
    -- distributed-monitoring  
  -- cat  
    -- wangwangwang  
-- eclipse  
  -- che
```

```
-- flux
-- che
-- good-man
  -- ex
    -- python-final
  -- goodess-bf-1
  -- goodess-ex-bf
  -- master-thesis
```

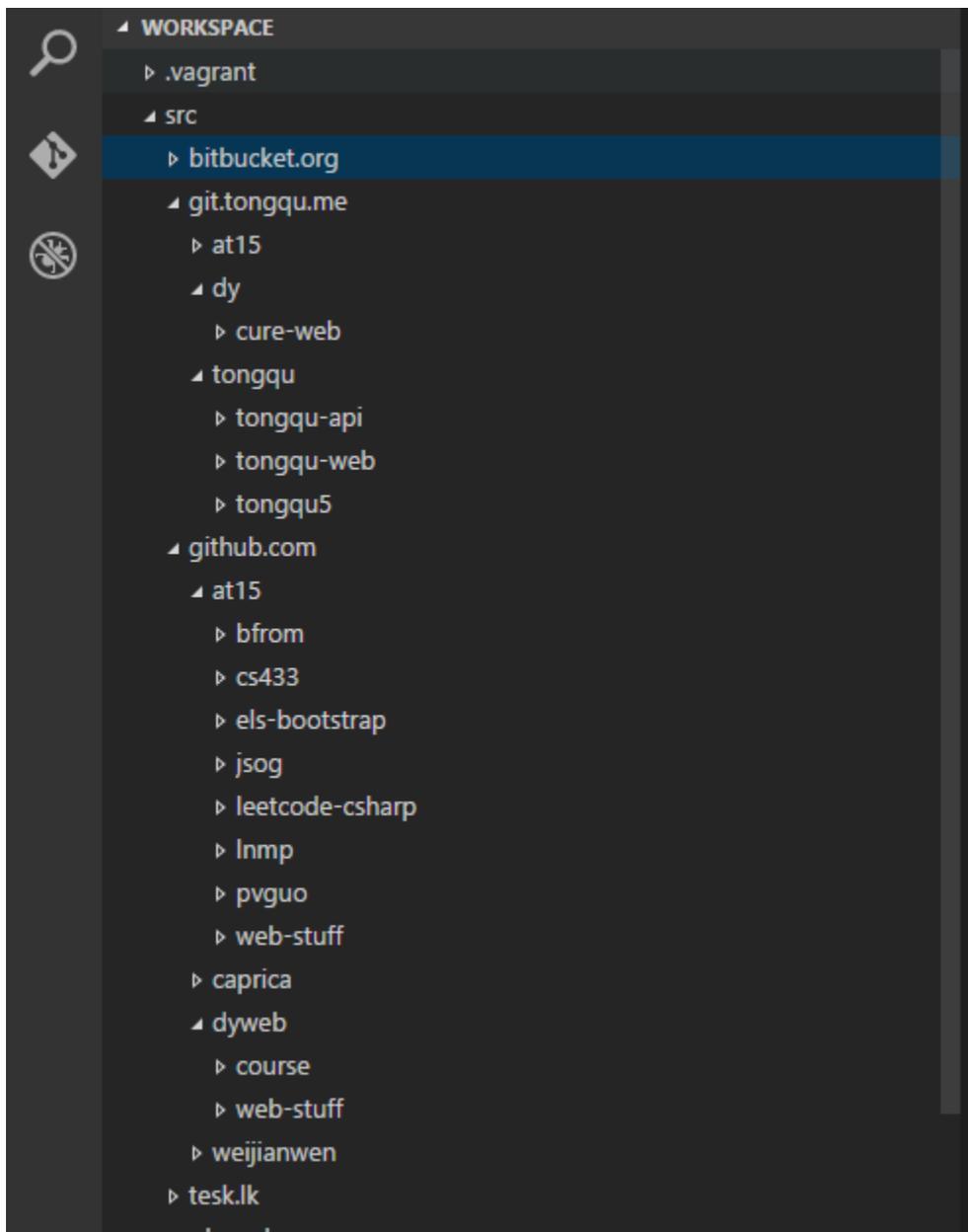
Organize by Git host

This is [the Go way](#) , if you have a parent directory for bin.

Since it's a good idea to have all your code in sync, you may have multiple Git host, GitHub for public repos, GitLab for internal projects

```
D:/Code/
-- github.com
-- at15
  -- crawler
  -- how-to-play-tennis
-- gaocegege
  -- scrala
  -- how-to-dig-holes
-- git.tongqu.me
-- tongqu
  -- tongqu5
  -- tongqu-api
  -- tongqu-web
```

the workspace for at15. the editor is [VSCode](#)



Reference

- [Git](#)
- [GitHub](#)

Editor

Problems

- [encoding](#)
- [style](#)

- hint
- large code base
- extension

Solution

gitattributes

Git can handle file encoding and line ending, use config file to override gloabl config.

Offical Doc

- encoding, UTF-8
- end of line (crlf) , sh files can only use LF

Example:

```
# Force sh file use lf
*.sh text eol=lf
Makefile text eol=lf
*.bat text eol=crlf
```

editorconfig

A editor independent style config. Some editor has it built in, most has plugin for it.

Offical Site

- encoding
- end of line (crlf)
- indent
- tab

Example

```
root = true

[*]
insert_final_newline = true
indent_style = space
charset = utf-8

[*.{sh,js,php,css,scss}]
end_of_line = lf
charset = utf-8

[*.{js,json,yml}]
indent_size = 2

[*.{php}]
indent_size = 4

[*.{md,markdown}]
```

```
indent_size = 4

[Makefile]
indent_style = tab
```

lint tools

Lint tools can change your style, some can auto fix minor problems. You can also config git hooks to force running lint tools before commit. Some git host like [phabricator](#) also support running it on server.

- Javascript [eslint](#)
- PHP [phpcs](#)

IDE

Integrated Development Environment. Editor + Code runner + Debugger etc.

Jetbrains

- [official site](#)

Student can have free license

- [WebStorm](#) Front end
- [PhpStorm](#) Front end + PHP
- [IDEA](#) Front end + Java + Scala + Groovy
- [PyCharm](#) Front end + Python

Also have plugin for [Golang](#)

How hint works

Hint is the editor show a list of words based on what you typed. It is also known as autocomplete, [intellisense](#).

Text only

The editor show hint based on the words in this file only.

TODO: image

Example

- Atom
- Sublime
- VSCode

Analysis result

The editor (most time IDE) analysis all the code used in current project, including your code, library code, standard library code. This is also called index, similar to what search engine does.

TODO: image

Example

- Eclipse
- JetBrains IDE
- Visual Studio

However, with plugins, a lot editors can also show hint based on analysis result, like VSCode works perfectly with C# if you have Visual Studio installed.

Recommended editors

- Atom
- Sublime
- VSCode

Not recommended

They are well known editors, but since using them require a lot of config, it's better to use editor or IDE that comes out with a lot of default settings and plugins. You can try if you think you can handle them.

- Vim
- Emacs

Outdated editor and IDE

If you want to waste time, try one of the following. If you want to waste other people's time, recommend those to them.

- Dreamweaver

Command line

aka: terminal, console, cli, cmd, bash etc.

What is command line

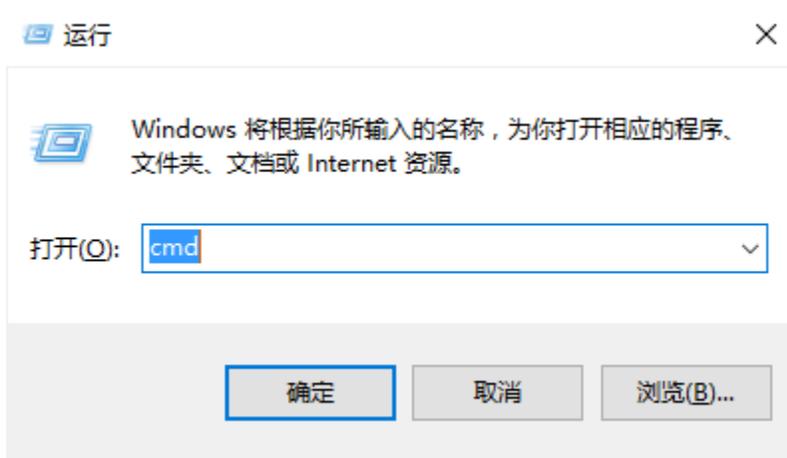
You type and run commands in plain text, and get text output. NO GUI (Graphic User Interface) interaction.

tl;dw

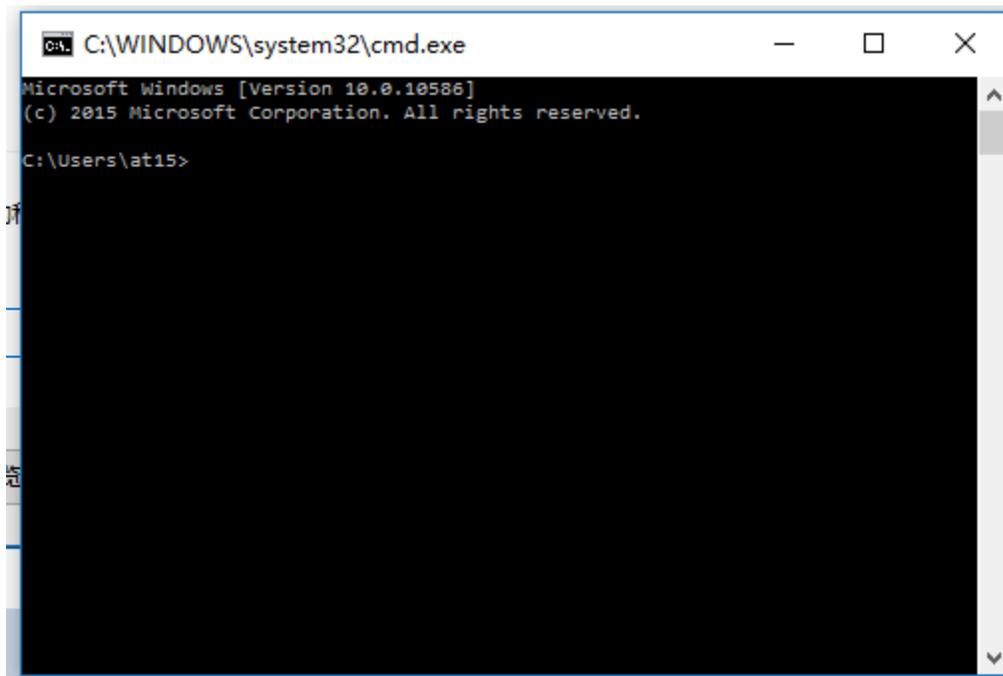
How to

Open a command line

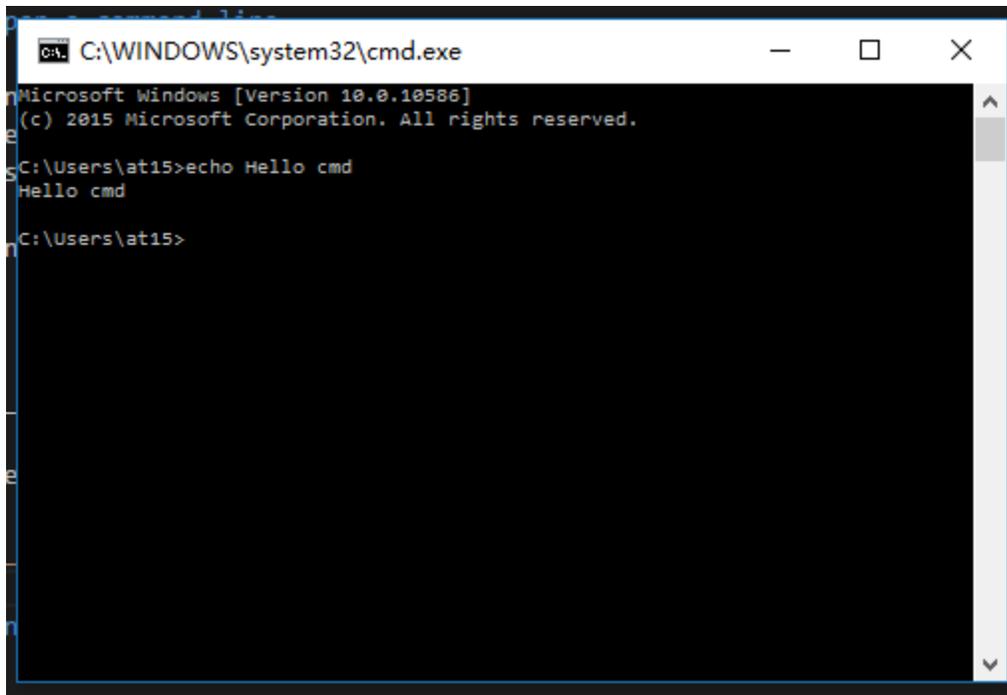
- win + R (this means you press the windows icon on your keyboard, and press the R letter at same time.)
- type cmd
- press enter



- you can see a black window



- type something like echo Hello cmd



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.10586]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\at15>echo Hello cmd
Hello cmd

C:\Users\at15>
```

- list folders `dir`

Concepts

Executeable

- binary TODO: vagrant, a c example. (windows: exe, linux: executeable)
- script TODO: a bat example, a python example.

Arguments

- a php example

Current folder

PATH

- windows config
- linux config

how to use it.

Environment variables

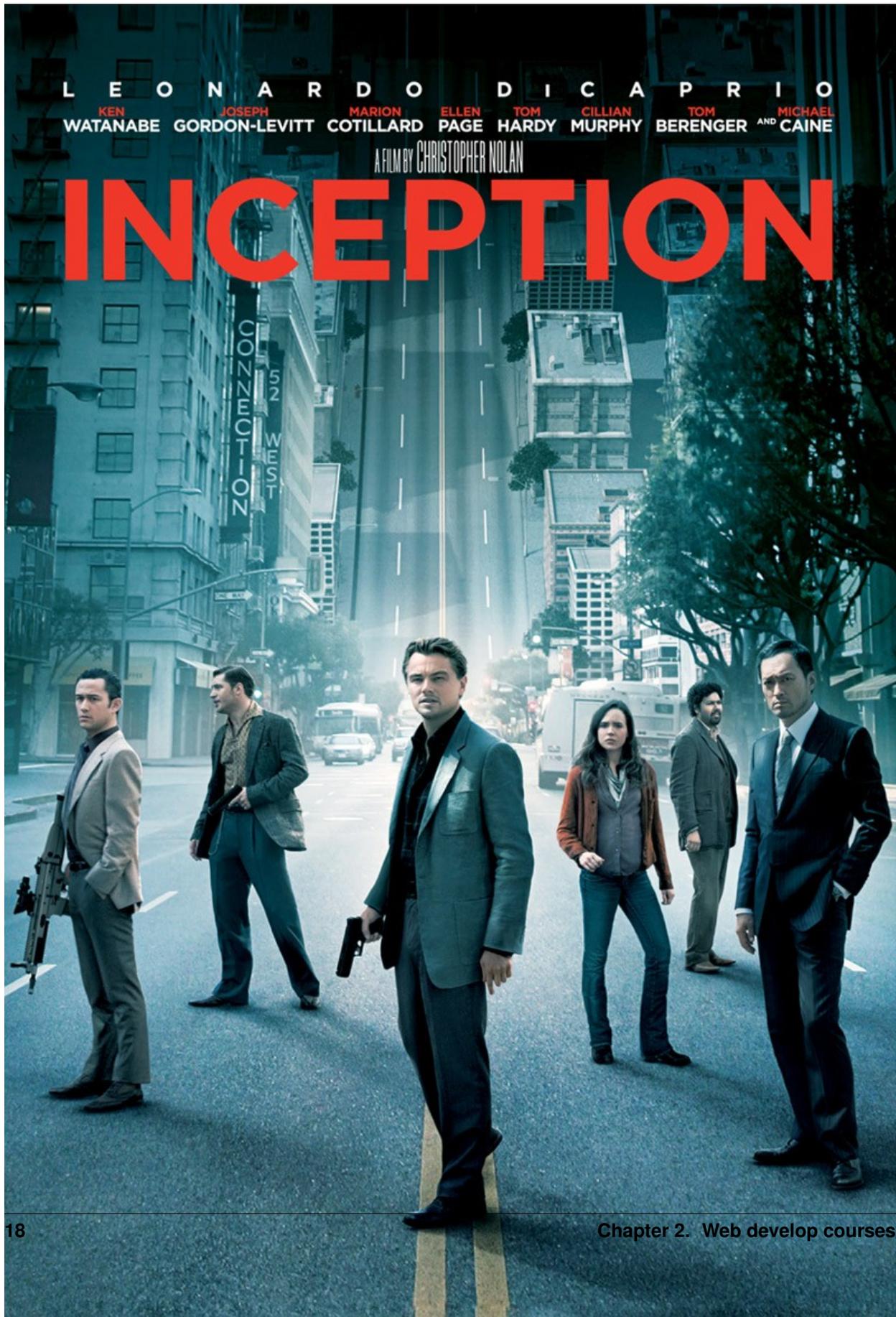
- windows config
- linux config

Autocomplete

use tab

Virtual Machine

How virtual machine works



e.... in fact, I don't know how it works....

Concepts

Host

Your laptop

Guest

The Ubuntu system running inside virtualbox/vmware/hyper-v

Shared folder

You can edit your documents on host inside guest. You can also edit document inside guest in your host.

Problems

- Network
- File sync
- Resource wasted on GUI

Hard to config

Solution

TODO: change to rst and link internal doc for vagrant

- vagrant
- docker

Vagrant

Official site

Create and configure lightweight, reproducible, and portable develop environment.

Vagrant vs Virtual machine vs Docker

- vagrant is more friendly for development
- you should not package everthing in one docker image

Concepts

Vagrantfile

Shared folder

Port

Provision

SSH

TODO: don't need to go into detail of ssh right away ...

- vagrant ssh
- putty

Commands

vagrant up

vagrant ssh

vagrant destroy

vagrant

Use the lnmp box

at15/lnmp is a preconfigured box for LNMP stack.

```
vagrant box init at15/lnmp
vagrant up
```

FAQ

TODO: time out (ssh), VT-X

Request lifecycle

We omit the low level and irrelevant part, this is the short version of the famous question [what happens when you type google.com into your browser and press enter](#).

TODO:

- my short video
- browser make a http request. (omit tcp handshake etc)

- nginx
- php
- mysql
- return to browser
- css
- javascript
- ajax (interaction without refresh the whole page)

Linux

- Cheap. No License fee, less electricity fee.
- Open source.

NOTE: Mac OSX is based on Unix, but it works like Linux distributions most of the time.

Differences between Linux and Windows in Command line

This only list some frequently asked questions when you start using command line.

Directory separator

Windows use \ as directory separator, while Linux use /

Example

Windows

partition + : \ + folder + \ + filename

```
C:\Program Files\Adobe\Photoshop CC\amtlib.dll
D:\abs-132.rmvb
```

Linux

Folder + / + filename. Linux DOES NOT have partition in path. NO C,D etc.

```
/home/at15/Downloads/abs-132.rmvb
/etc/nginx/conf.d/tongqu.lk.conf
```

However, most programming language can deal with it automatically.

Case sensitive

- **Windows is NOT case sensitive**
- **Linux IS case sensitive**

Windows

```
D:\Code\Happy.txt
D:\Code\happy.txt -> This file is not allowed
```

Linux

```
/home/at15/Downloads/abs-123.rmvb -> ok
/home/at15/Downloads/abs-123.rmvB -> ok, notice that b is upper case.
```

This could cause TROUBLE when you are using Git.

TODO: how to solve that trouble ... and image, error message for that trouble.

Password

- **Windows have *** to show you how many words you have entered for password
- **Linux shows nothing**

TODO: images

Nginx

Nginx is a static server, though it could become an application server using lua, ie: [openresty](#)

nginx [engine x] is an HTTP and reverse proxy server, a mail proxy server, and a generic TCP/UDP proxy server, originally written by Igor Sysoev.

TODO:

- config file syntax
- root
- location

Difference with Apache

- fast

PHP

The best language in the world

php-fpm php mod_php

Homework

workspace

If you don't know Git, you can skip this one until you have took the course for it or learned it by yourself.

- [] create a project on GitHub
- [] create a project on git.tongqu.me, ask @arrowrowe if you do not have an account.
- [] all the project you created must have readme, gitignore, gitattributes.
- [] clone the project to your local work space

- [] fork one of [gaocegege](#)'s repo, clone the original one and your fork to your local work space.
- [] submit your a picture of your workspace using a editor/IDE that supports open folder.

command line

For both windows and linux users

- [] change directory
- [] change partition, from C : to D :
- [] list files
- [] create, remove folder

For windows users, you can do the linx part after you have installed Git Bash, which will be introduced in the following Git course.

For linux and mac users

- [] `~~try rm -rf /~~`

TODO: @arrowrowe

vagrant

- [] have at15/lnmp up and running
- [] ssh into it.
- [] phpinfo

PHP

- [] write a script and run it from commandline, and works like following

```
$ php hello.php mie
$ Hello, this is mie!
```

- [] write a script and get its output from browser

```
http://localhost/hello.php?name=mie
Hello, this is mie!
```

You will start form the defination of environment to your how to organize your local workspace, the editor you use. How to use commandline, what is virtual machine, how vagrant makes a difference. The famous LNMP stack. The homework is quite simple, write a helloworld in php, and show the result in your browser.

Git & HTML

This course mainly tells you how to start writing HTMLs in an editor, using Git for version control.

Contents:

Git Introduce

What is Git?

The widely used source code management system for software development in the world.

History

- In 2002, the developers of the Linux kernel use [Bitkeeper](#) to maintain the project, but the Bitkeeper is not good for developing.
- In 2005, the copyright holder of BitKeeper, Larry McVoy, had withdrawn gratis use of the product after claiming that Andrew Tridgell had reverse-engineered the BitKeeper protocols.
- In 2005, Torvalds decided to develop a distributed system to replace the Bitkeeper. So, he create git.

Version Control

You won't like this:

```
D:/Schoolwork/Group-work-1
-- our-fancy-work-v1-outline-only
-- our-fancy-work-v2-victor-part-done
-- our-fancy-work-v2-cece-part-done
-- our-fancy-work-v3-victor-and-cece-merged
-- our-fancy-work-v3-victor-part-updated
-- our-fancy-work-v4-victor-part-updated-again
-- our-fancy-work-v5-victor-part-updated-again-and-cece-merged
-- our-fancy-work-v8-final
-- our-fancy-work-v9-final-updated-by-victor
-- our-fancy-work-v9-final-updated-by-cece
-- our-fancy-work-v10-final-updated-merged
-- our-fancy-work-v11-final-final
```

What if we have Git?

```
D:/Schoolwork/Group-work-1
-- our-fancy-work
-- .git # <- Hidden
```

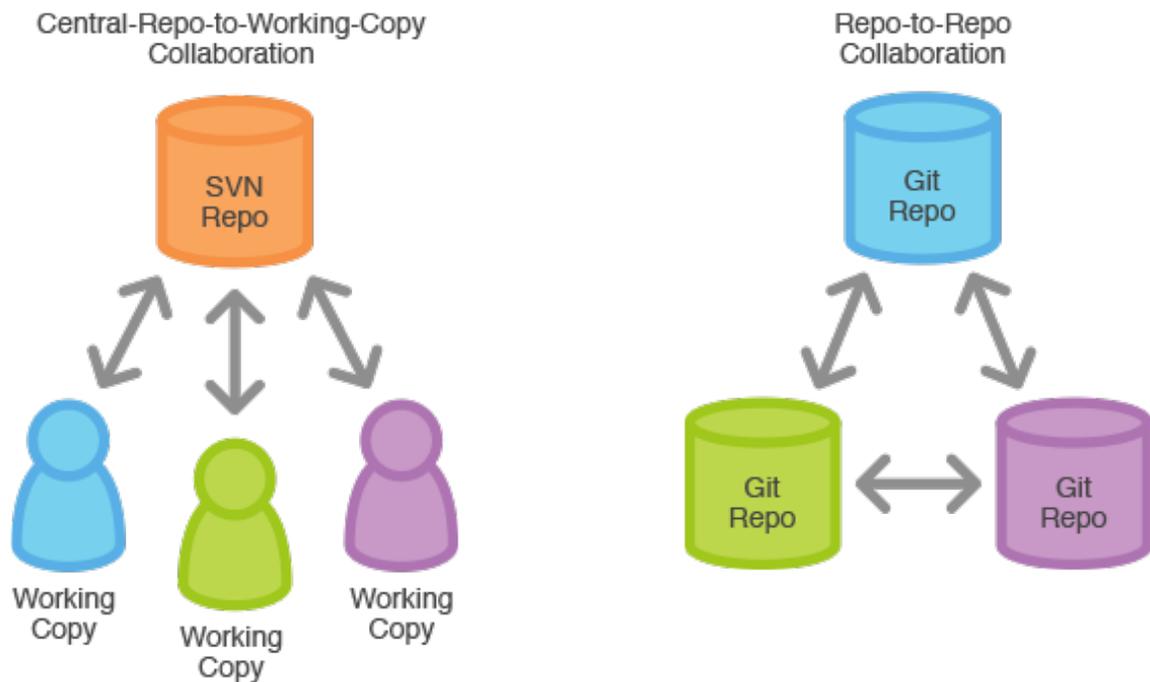
```
$ git log --graph --oneline
* 1fa35d1 final final
*   d0eb39d Merge branch 'cece'
| \
| * 10604d8 final updated by cece
* | 243bd99 final updated by victor
| /
* 8e4c071 final
*   a8e02e7 Merge branch 'master' into cece
| \
| * 222a46f victor part updated again
| * 391ea6c victor part updated
* |   7b3fe97 Merge branch 'master' into cece
| \ \
| | /
```

```
| * e919d2d victor part done
* | e0c35e5 cece part done
|/
* 8ab6c50 outline only
```

Collaboration

When you work in a team. It's necessary to integrate everyone's work.

- copying and cover the files ?
- svn, working-copy collaboration
- git, repo-to-repo collaboration



Benefit

- Detailed history.
- Better collaboration: safely write on your own, then merge.
- Peer review.

Reference

- [Wike.Git\(software\)](#)

Git Setup

Install on windows

Maybe it's complicated, but I don't use windows for several years. But with the big news for ubuntu in windows10, everything may become easy.

Install on Mac OS X or Linux

I don't want to tell you, because it's so easy.

Use command line

If you are in Windows

- Go to the directory you want to work in, and click your right mouse and choose git bash.
- Input 'git' to see the usage

if you are in Mac OS X or Linux

- Input 'git' to see the usage

if should look like this

```
$ git
usage: git [--version] [--help] [-C <path>] [-c name=value]
        [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
        [-p | --paginate | --no-pager] [--no-replace-objects] [--bare]
        [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
        <command> [<args>]
```

First command: set up your personal info

```
$ git config --global user.name [username]
$ git config --global user.email [email]
```

The commands above will let the git know who you are. the `global` option will set the attribute globally.

Remove the `global` will set the attribute inside your project.

The basic commands you have to know

Initialize a repository

```
git init
```

```
$ git init
Initialized empty Git repository in /Users/sway/WebProj/test-project/.git/
```

This command will initialize a repository in current directory.

```
$ ls -a
.  ..  .git
```

And as you see, a new subdirectory named `.git` was created in current directory which contains all your repository files.

Clone an existing repository

```
git clone [url]
```

```
$ git clone https://github.com/dyweb/web-stuff
remote: Counting objects: 695, done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 695 (delta 4), reused 0 (delta 0), pack-reused 684
Receiving objects: 100% (695/695), 1.56 MiB | 224.00 KiB/s, done.
Resolving deltas: 100% (320/320), done.
Checking connectivity... done.
```

This will copy the a existing remote repository into your local and every version of every file for the history of the project will be pulled down.

Checking the status of the repository

```
git status
```

```
$ git status
```

As the title says, this command will show you current status of the repository.

Git Workflow

Submit your local change

Last topic, we list some common commands, like `git init`, `git clone`, `git status`. Now we will track a change from your local repository to remote.

For example we create a `README.md` file in your initialized directory. Now, run `git status`

```
$ git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

  README.md
```

As we see, git found a new file named `README.md` was created in the directory, and the file hasn't been tracked.

So, we follow it's guide, run `git add` to add the file to the git index, which is called stage changes.

```
$ git add README.md
$ git status
On branch master

Initial commit

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)

   new file:   README.md
```

Now, the file has been really added to git index, but before you want to submit it to remote repository, you should use `git commit` to create record contained all your stage changes.

```
$ git commit -m"Init"
[master (root-commit) 8ecc069] Init
 1 file changed, 1 insertion(+)
 create mode 100644 README.md

$ git status
On branch master
nothing to commit, working directory clean
```

This time, run `git status`, you will see nothing to commit, it proves that your local change has been saved as a commit record.

Use `git push` to submit your commit to remote master branch.

```
$ git push origin master
```

Update your local repository from remote

When you find the remote repository has updated some changes, you can use `git pull` to fetch the latest changes.

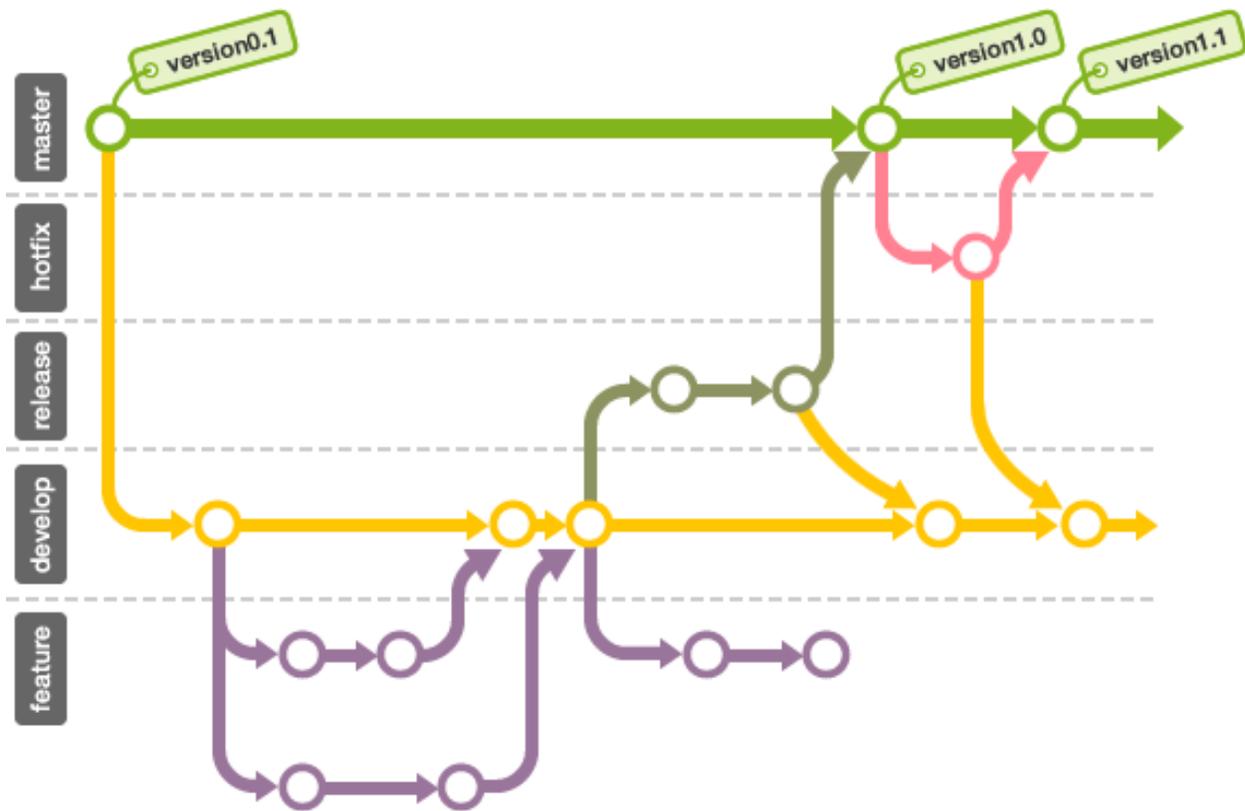
```
$ git pull origin master
```

Git Branch

What is branch?

- Nearly every VCS has some form of branching support.
- Branching means you diverge from the main line of development and continue to do work without messing with that main line.

see the following picture



Working with branches

Create branch

```
git branch <branch name>
```

Switch branch

```
git checkout <branch name>
```

It is worth mentioning that if you want to checkout to an inexistent branch, you will get error message. Like this one.

```
$ git checkout vain-branch
error: pathspec 'vain-branch' did not match any file(s) known to git.
```

But if you add `-b` option, you will create a new branch and switch to the new branch

```
$ git checkout -b vain-branch
Switched to a new branch 'vain-branch'
```

Merge branch

See the above picture, you will find sometimes branches should be merged. Run the following command, you will merge appointed branch into your current branch.

```
git merge <branch name>
```

But frequently, The operation won't execute well when same codes were modified by different people in different branch. This time, when you run `git merge`, git will tell you where are the conflicts. Like this one.

```
$ git merge gh-pages
Auto-merging index.html
CONFLICT (content): Merge conflict in index.html
Automatic merge failed; fix conflicts and then commit the result.
```

You can see the conflict is in 'index.html', so open the file and find the conflict mark.

```
<<<<<< HEAD
The codes in current branch
=====
The codes in merging branch
>>>>>> Merging branch
```

The codes under the <<<<<< HEAD is your current branch codes, the codes between the ===== and >>>>>> Merging branch is the codes in merging branch.

Here is a sample.

```
<<<<<< HEAD:index.html
<div id="footer">sway</div>
=====
<div id="footer">
sway
</div>
>>>>>> gh-pages:index.html
```

Reference

- [Learn Git Branching](#)

Git Teamwork

Commit message

Why we should write good commit messages?

- To speed up the reviewing process.
- To help us write a good release note.
- To help the future maintainers.

How to write good commit messages?

Cause we are all students, I don't think we should obey lots of rules of writing commit messages. If your messages can satisfy the following three points, I think it's ok.

- Write the accuracy first word in your message subject. Like fix, enhance, doc, style, bug ...
- Don't take your subjective emotion. Like fuck, wo cao, !!! ...
- Make the messages clear. Know the meaning of your own message at least...

Comparison of two different teamwork

Commits on Jan 5, 2016



docs(error/\$rootScope/inprog): add missing "\$timeout" ...

wytesk133 committed with petebacondarwin 10 days ago



docs(tutorial/2): add e2e test missing filename ...

monkpit committed with petebacondarwin 3 hours ago



revert: feat(\$compile): Allow ES6 classes as controllers with `bindTo...` ...

petebacondarwin committed 3 hours ago



fix(\$animateCss): only (de)register listeners when events have been a... ...

Narretz committed 25 days ago



fix(loader): use `false` as default value for `transclude` in compone... ...

sarod committed with petebacondarwin 18 days ago

Commits on Jan 3, 2016



feat(\$compile): Allow ES6 classes as controllers with `bindToControll... ...

IgalFaso committed 21 days ago



chore(*): Updated year in licence ...

leuanG committed with IgalFaso 3 days ago

Commits on Jan 1, 2016



docs: reorganize information about interpolation ...

Narretz committed on Dec 6, 2015



docs: add docs for ngPattern, ngMinlength, ngMaxlength ...

Narretz committed on Sep 24, 2015



docs(\$interpolateProvider): remove superfluous ng-app attribute ...

Narretz committed 4 days ago

1



wei authored about a month ago

Merge branch 'develop' of git.ramy-inc.com:gsj987/play-wearable-android into deve...



wei authored about a month ago

1



wei authored about a month ago

1112



Sang Xiang authored about a month ago

Merge branch 'develop' of git.ramy-inc.com:gsj987/play-wearable-android into deve...



Sang Xiang authored about a month ago

1221



Sang Xiang authored about a month ago

Merge branch 'develop' of git.ramy-inc.com:gsj987/play-wearable-android into deve...



wei authored about a month ago

1



wei authored about a month ago

111



Sang Xiang authored about a month ago

**Every commit messages will be your possible black history ~ **

Maintain multiple branches

If you only have one branches, you may meet some problems.

- Boss: The clients say the product should add xxx function! You should add it before tomorrow!
- *You add the function in you only branch hardly, you forget other programmer change some architecture yesterday. Finally, you build a incomplete version and hand out the clients*
- It's a stupid tragedy.

So, you should always maintain multiple branches in your work, especially in teamwork.

Popularly, we always maintain two different types of branches.

- Main branches
 - master

- stable
- Support branches
 - feature
 - bug
 - hotfix

Reference

- [A commit message guid by Ruan Yifeng](#)
- [Git/GitHub branching standards & conventions](#)

HTML

What is HTML?

- HyperText Markup Language.
- Describe web documents (the basic of web).
- Consist of different tags.

When you type a domain in browser, you send a request the website server.

The website server will receive your request and return a HTML string to you.

The browser will interpret the HTML string and display the web page.

A Sample HTML

```
<!DOCTYPE html>
<html>
<head>
  <title>My first html</title>
</head>
<body>
  <div style="background-color: green">
    <p> My first paragraph </p>
  </div>
  <p>My second paragraph.</p>
</body>
</html>
```

- The text between `<html>` and `</html>` describes an HTML document.
- The text between `<head>` and `</head>` provides information about the document.
- The text between `<title>` and `</title>` provides a title for the document.
- The text between `<body>` and `</body>` describes the visible page content.
- The text between `<div>` and `</div>` describes a block element.
- The text between `<p>` and `</p>` describes a paragraph.

HTML DOM TREE

If you look the above sample carefully, you will find the structure of the HTML is a tree.

```
html
 |   \
head body
 |     | \
title div p
       |
       p
```

This standard makes HTML more structuring, and be easy converted to object model which we can easily modify it.

Reference

- [W3school HTML Tutorial](#)

You will start form the defination of environment to your how to organize your local workspace, the editor you use. How to use commandline, what is virtual machine, how vagrant makes a difference. The famous LNMP stack. The homework is quite simple, write a helloworld in php, and show the result in your browser.

Instant PHP

This course mainly tells you how to get a quick start with the best programming language PHP.

Contents:

Source

The source files of this lecture can be found here: [GitHub Repo](#)

Slide

The slide can be found here:

[Slide](#)

Assignment

Hand In

- Email: web@dongyue.io
- GitHub / GitLab Tq:
 1. Crete a new repository and make it public
 2. Open an issue [here](#) and tell us your repository link
- QQ, Wechat is also OK!

Due?

- Deadline: May 2nd
- Answers available next week

Assignment I

Finish a simple To-do List application.

- Only one file “todo.php” is enough
 - Multiple files are also ok
- Try to implement displaying, creating, editing & deleting of to-do items
- No JS & CSS needed
 - Just focus on PHP coding
- (Advanced) Try to add file uploading in item creating form
 - *Tips: Use `$_FILES` superglobal*

Assignment II

The output of the following code looks strange. Try to figure out why.

```
<?php
$array = ['a', 'b', 'c', 'd'];
$array2 = [1, 2, 3, 4];
foreach ($array as &$item) {
    echo $item . PHP_EOL;
}

foreach ($array2 as $item) {
    echo $item . PHP_EOL;
}

echo implode($array, ',') . PHP_EOL;    // Outputs a,b,c,4
echo implode($array2, ',') . PHP_EOL;  // Outputs 1,2,3,4
?>
```

Assignment III

- Implement simple login logic
- 4 files needed: index.php, private.php, login.php, logout.php
- Only a user who has logged in has access to private.php
- Use Session to finish this assignment

Assignment IV

Finish an event registering site (simple-tongqu).

- You need to use implement:
 - Event List
 - Event Registration CRUD
 - * Register an event
 - * Modify registering info
 - * Exit an event
 - * Get all the registrations
 - Login is not needed
 - * You can implement it anyway
 - Keep data in disk files
 - * No database is needed
- You need to split components, logic & views
 - aka. MVC in later lectures
- Suggested structure
 - register.php
 - event.php
 - libraries/functions.php
 - libraries/Event.php
 - libraries/File.php
 - views/register_xxxx.php, views/register_yyyy.php
 - views/event_xxxx.php, views/event_yyyy.php

Reading Materials

Tutorials

- Codecademy PHP Tutorial
- *PHP and MySQL Web Development (4th edition)*
- PHP
- PHP
- PHP Fundamentals

Guides

- PHP Manual
- The Best Way to Learn PHP
- PHP: The Right Way

HTTP & RESTful API

This course gives you a brief introduction of HTTP protocol & RESTful API.

Contents:

Slide

The slide can be found here:

[Slide](#)

Assignment

Hand In

- Email: web@dongyue.io
- GitHub / GitLab Tq:
 1. Create a new repository and make it public
 2. Open an issue [here](#) and tell us your repository link
- QQ, Wechat is also OK!

Due?

- Deadline: May 16th
- Answers available next week

Assignment I

1. Learn to use Chrom DevTool & Fiddler to observe HTTP connections.
2. Select a PHP microframework (Slim/Flight/...) and implement a simple REST service of a to-do list.

```
GET /api/v1/tasks           // Get all tasks
GET /api/v1/tasks/:id      // Get a task by id
POST /api/v1/tasks         // Create a new task
PUT /api/v1/tasks/:id      // Update a given task
DELETE /api/v1/tasks/:id   // Delete a task
```

1. (Optional) Write a crawler for [Tongqu Act API](#). Here is the doc of this API.

Reading Materials

Tutorials

- [HTTP: The Protocol Every Web Developer Must Know](#)
 - [Part I](#)
 - [Part II](#)
- [HTTP](#)
- [HTTP](#)
- [RESTful API](#)
- [RESTful](#)
- [REST API Tutorial](#)

Frameworks

- [Slim](#)
- [Flight](#)

Tools

- [Fiddler](#)
- [Chrome DevTools OverView](#)
- [Wireshark](#)
- [Postman](#)
- [Scrapy](#)

original schedule is in [dyweb/web-stuff](#) repo.

CHAPTER 3

Mos courses

Contents:

2017 Spring

Courses for 2017 Spring

Contents:

Docker

Slides

[google slides](#) or offline version

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`