PHP Basics

Marcelo Gaspar
drmpgaspar@gmail.com
https://github.com/Mpgaspar/phpbasics-pill.git

PHP BASICS

27th November 2019

Overview

In this project you will learn the basic notions of the famous PHP language which is so used in the world of web development.

List of tasks to be performed:
01. Priority of each task
02. Title and description of each of them
03. Difficulty level
04. Estimated time for each task.
05. Record of incidents that were detected during project execution.
06. Record of lessons learned.
07. Project calendar
08. The Chronogram of the project.
09. Quality control measurements.
10. Quality metrics
11. Requirements documentation.
12. Risk documentation in the event that these exist
13. Documentation about the git WORKFLOW you are going to use
14. Documentation about the tools used in the project

Priority (1-3) 1 = low, 2 = moderate, 3 = high

A: Organization	3
B: Study PHP documentation	2
C: Repository in git	3
D: Test PHP server	2
E: Create PHP files	3
F: Test PHP functions	3
G: Create PHP classes	2

A: **Organization** = structuring all tasks by priority and estimated execution time.

B: **Study PHP documentation** = just study the language.

C: **Repository in git** = put the project into a git repository to organize changes during construction.

D: **Test PHP server** = just test php server.

E: **Create PHP files** = create all files .php (print.php , iterators.php, operators.php, dates.php, conditionals.php, types.php, maths.php, functions.php, strings.php, arrays.php, phpinfo.php, poo.php)

F: **Test PHP functions** = test all php functions.

G: **Create PHP classes** = create 4 classes with my own builders and destroyers.

Difficulty level (1-3) 1 = low, 2 = moderate, 3 = hard

A: Organization and Documentation	2
B: Study PHP documentation	2
C: Repository in git	1
D: Test PHP server	2
E: Create PHP files	3
F: Test PHP functions	3
G: Create PHP classes	3

Time (hours)
A: Organization and Documentation = 4h
B: Study PHP documentation = 4h
C: Repository in git = 1h
D: Test PHP server = 2h
E: Create PHP files = 10h
F: Test PHP functions = 5h
G: Create PHP classes = 6h

H: Incidents = 8h (20% total time)

Total time: 40h

Incidents

- Difficulty fulfilling the given time for each task
- Some tasks were underestimated
- Difficulty to create some PHP functions
- Stop the project to study PHP documentation
- Knowledge limitation in POO
- Difficulty to create some PHP classes

Lessons

- Better organize myself to accomplish the tasks
- I learned to prioritize the most important tasks
- Use git to our advantage
- Understand how classes works
- Improve my knowledge in PHP basics
- Improve my knowledge in work with PHP classes
- Better understand PHP functions
- Take questions and discussed project organization with colleagues
- Improve your knowledge in project management.
- Have patience and be humble

Project Calendar

Main tasks
27/11 → Organization and Create repository in git
28/11 → Study PHP documentation
29/11 → Test PHP server and create PHP files
30/11 → Test PHP functions
01/12 → Create PHP classes
02/12 → Test classes
03/12 → Test and finish documentation

TASK/ DAY	WED	THU	FRI	SAT	SUN	MON	TUE
	27/11	**28/11**	**29/11**	**30/11**	**01/12**	**02/12**	**03/12**
A	X						X
B		X		X	X		
C	X						
D			X				
E			X	X	X		
F				X	X	X	
G					**X**	**X**	

Chronogram of the project

27/11 - Organization and division of tasks by priorities
- Start documentation
- Put repository in git

28/11 - Study PHP documentation
- Work with git

29/11 - Test PHP server
- Create PHP files

30/11 - Study PHP documentation
- Continue create PHP files
- Test PHP functions

01/12 - Study PHP documentation
- Continue create PHP files
- Continue test PHP functions
- Create PHP classes

02/12 - Finish PHP classes
- Finish test PHP functions

03/12 - Finish documentation
- Push to GitHub repository

Quality control measurements

The quality control measurements are used to analyze as well as evaluate the quality of the different processes involved in a project against the standards of the organization or on the requirements specified during the project management planning.

- Did you use clean code?
- Does the site have all the specifications requested?
- Did you use PHP functions and classes?
- Do the commit messages properly explain the development flow of the project?
- Have all functions been tested throughout the project realization process?

Requirements

- You must use GIT

•You must use a version of PHP higher than 5.1

•Create a clear and orderly directory structure

•Both the code and the comments must be written in English

•Use the camelCase code style to define variables and functions

•In the case of using HTML, never use online styles

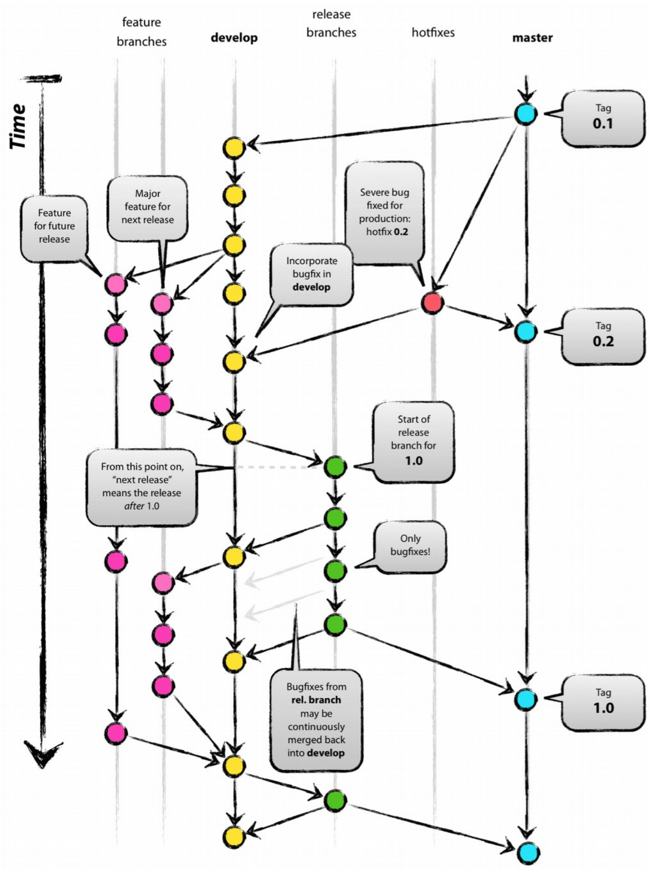
•In the case of using different programming languages always define the implementation in separate terms

•Remember that it is important to divide the tasks into several sub-tasks so that in this way you can associate each particular step of the construction with a specific commit

•You should try as much as possible that the commits and the planned tasks are the same

•Delete files that are not used or are not necessary to evaluate the project

Risk documentation
Incidents (disease, technical failure, poor performance, unrealistic deadlines) may change:
+ Statement of work (SOW)
+ Work breakdown structure (WBS)
+ Budget
+ Schedule
+ Execution plan


feature branches
develop
release branches
hotfixes
master
Time
Tag 0.1
Feature for future release
Major feature for next release
Severe bug fixed for production: hotfix 0.2
Incorporate bugfix in develop
Tag 0.2
Start of release branch for 1.0
From this point on, "next release" means the release *after* 1.0
Only bugfixes!
Bugfixes from rel. branch may be continuously merged back into develop
Tag 1.0

Git Workflow
Described by Vincent Driessen in 2010.
I used 3 branches:
- Master
- Develop
- Feature

+ “Master” is always ready to be released on LIVE, with everything fully tested and approved (production-ready).

+ “Develop” is the branch to which all feature branches are merged and where all tests are performed. Only when everything’s been thoroughly checked and fixed it can be merged to the Master.

Project Tools
- Visual Studio Code
- Git and GitHub
- Google Chrome Developer Tools

Resources:

- https://validator.w3.org/
- https://www.w3schools.com/php/
- What can PHP do? [Official Website]: https://www.php.net/manual/es/intro-whatcando.php
- https://stackoverflow.com/
- README Guidelines Example: https://gist.github.com/PurpleBooth/109311bb0361f32d87a2