Assignment 2 (minor subject / Nebenfach)

Due: Wed 04.11.2015; 12:00h (1 Week)

Goals

These exercises you will help you...

- Get to know server side programming with PHP
- Handle form data to allow you to create interactive websites
- Dive into the workings and advantages of (associative) arrays

Task 1: Typing: Static, Dynamic, Duck.

In the tutorial we briefly talked about static and dynamic typing. Now, do an online research about what the term ‘duck typing’ means. Then answer the following questions:

a) Is duck typing possible in PHP? Give an example for your answer.
b) What is the difference between static and dynamic typing?
c) Can duck typing occur with static typing? Explain your answer.
d) Last week, we talked about JavaScript. Is duck typing possible there?
e) Name two advantages of dynamic typing and two advantages of static typing.

Put your solution as .txt or .pdf file in the folder ‘task1’. Please cite the articles correctly (provide links to your sources)

Task 2: Simple Quiz App.

PHP is evaluated on the server. This also means, the source code of PHP scripts is never sent to the clients. In other words, the users cannot look into it. This makes server side scripting essential when we want to hide information, which is necessary to create a quiz game for the browser.

Your quiz should do the following:

- Users need to answer **at least three** different multiple-choice questions.
- There are **three answer options** for each question. Only one answer is correct.
- The user can submit his or her answer with a **button**.
- After submitting the answers, the quiz should **highlight which ones were correct** and which ones were incorrect, e.g. by coloring the text green.
- The user can **re-submit** the form until all answers are correct. Once all answers are correct, there is a congratulation message.
- After each submission, it should be displayed **how many** answers are correct (e.g. “*You answered 2 out of 3 questions (66%) correctly*”)

Also, consider the following aspects to optimize your solution:

- It should be possible to easily add, change or remove questions without ruining the entire script. Take a look on how associative arrays might work for you.
- Redundant code is difficult to maintain, because changes need to be done in multiple locations. Make sure you do not write redundant code. Functions and loops definitely are useful here.
- Remember that you do not write code for a computer – you write it for other humans to read and improve it. Therefore, you should write comments as much as you can, unless they are absolutely trivial. We think that in well-written code one out of five lines is a comment.

Put your solution as single PHP file in the folder 'task 2'.

Task 3: Propose a task for the future or for the exam

Take into consideration what this week's lecture and tutorials are about and propose a task for this assignment sheet in future runs of this course. Also, if you want to, you can propose a task for the final exam. We reserve the right to actually use it.

Submission

Please turn in your solution via UniWorX. You can form groups of up to three people.

If you have questions or comments before the submission, please do not hesitate to contact one of the tutors. You can find their email addresses on UniWorX. Remember, that they also want to enjoy their weekends ☺

Also, feel free to ask questions by tweeting to https://twitter.com/MMN_WS1516. We try to react in a timely manner. That way, all the followers benefit from your questions. You can also leave us feedback about the tasks in this assignment (e.g. difficulty, attractiveness)