Assignment 4 (HF, major subject)

Due: Mon 16.11.2015; 12:00h (1 Week)

Goals

After doing these exercises,

- You know your way around MySQL
- You can store data to databases with PHP scripts
- You know how to secure user credentials

Task 1: Hangman High-Score

In tutorial 03 you started to create the hangman game, where the player needs to guess a word letter by letter. If a letter is correct, it is revealed at the correct position of a word, all other letters are displayed as underscores.

Now create a way to store a “high-score”. The score includes the following:

- Guessed word
- Number of attempts
- Player name or alias
- Date

Use PHP in combination with a MySQL database to store the data. The player can see the list of high-scores below the game or on a separate page.

Put all your source code files and resources in the folder ‘task1’.

Task 2: Note-Taking Web-app

Take a peek at a web page that can store user notes, e.g. Google Keep. Your task is to create a similar note-taking web-app. For now, it is okay to stick to PHP and MySQL (no AJAX so far), but since the lecture already introduced jQuery, you are allowed to use it to manipulate the DOM or add animations.

Consider the following requirements:

- The user can **add** notes
    - A note consists of a title and a text. Make sure to use the correct datatype in the database.
    - Each note has a unique ID that is invisible to the user. It’s an attribute in the HTML element and can be used to identify the note.
    - Ensure that SQL injections are not possible, when the user saves a note.

- The user can **delete** notes
  - Each note displays a small checkbox
  - It is possible to surround all the notes with a <form> element. Then you can have a "Delete" button that sends the IDs of all the checkboxes to the server. The script on the server takes care of removing the notes from the database and sending an updated output to the client.
- The user needs to be able to **sign-up** and **log-in** afterwards.
  - Create a simple registration page that takes a user name and a password. Send the data to a registration handler that hashes the password and stores the username-password-combination into the database. Use a 'users' table for that.
  - The log-in happens on the landing page. Use PHP-sessions to save the state (`$_SESSION['loggedIn']`).

Tips:

- Use a database handler class (see breakout session in tutorial 04) to encapsulate all transactions. You can even think of an authentication handler. Make use of object-oriented programming.
- It is okay to solve the task differently, than what we propose above (e.g. not using checkboxes to delete a note).
- Be creative! Maybe the notes can have different background colors or predefined labels?
- If anything in the task description is unclear, please contact us in a timely manner!
- Remember, you write code humans – not for machines. Make sure to comment your code as much as reasonable.

Put all your source code files and resources in the folder 'task2'.

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. You are welcome to do this with every assignment this year.
Put your proposition in the folder 'task3'.

Submission

Please turn in your solution via UniWorX. You can form groups of up to three people.
After the submission deadline, push your solution to our GitHub repository.