Assignment 2 (HF, major subject)

Due: Mon 02.11.2015; 12:00h (1 Week)

Goals

These exercises will…

- Show you the differences between the HTTP GET and POST methods
- Make you grow fonder with server-side scripting with PHP (potentially)
- Teach you the value of string and array functions in PHP

Task 1: What's the Problem here?

Take a look at this PHP script. It shows a number of flaws, e.g. in terms of security.

```
<!DOCTYPE html>
<html>
<head lang="en">
    <meta charset="UTF-8">
    <title>What's wrong here?</title>
</head>
<body>

<?php
function loginUser($email,$password){    return true;     }

if($_POST['submit']){
    loginUser($_POST['email'],$_POST['password']);
}
else{ ?>
<form>
    <label>
        Email:  <input type="email">
    </label>
    <label>
        Password: <input type="password">
    </label>
    <input type="submit" />
</form>
<?php } ?>
</body>
</html>
```

Put your solution in the folder 'task1'.

Task 2: Sffuhle my Wrods!

The human brain is able to easily read scrambled text. Take this example:

> Aoccdrnig to rscheearch at Cmabrigde Uinervtisy, it deosn't mttaer in waht oredr the ltteers in a wrod are, the olny iprmoetnt tihng is taht the frist and lsat ltteer be at the rghit pclae. The rset can be a toatl mses and you can sitll raed it wouthit a porbelm. Tihs is bcuseae the huamn mnid deos not raed ervey lteter by istlef, but the wrod as a wlohe.

The key is to keep the first and last letter of each word and shuffle what is in between.

Your task is to create a web-page that can scramble any text like this. The page can only consist of a textarea and a submit button. After submitting the form, the scrambled text should be displayed. Use PHP, i.e. server-side code, to complete this task.

Put your solution in the folder ‘task2’.

Task 3: Create a reliable, usable, secure contact form

Imagine a client asks you to create a contact form for their website.
The form should meet the following requirements:

1. Input fields should be available for first name, last name, email address, password, password confirmation, subject (selection from a list, e.g. “criticism”, “praise”, “other”), and a message panel.
2. Your client does not want to receive spam emails. Include a CAPTCHA. You may use a third party library for this. If you put your solution on GitHub, make sure to respect intellectual property and copyrights.
3. Make sure that the email address is valid. Do this via PHP. If the email is invalid, do not accept the form. Instead, let the user know what to change with visual feedback.
4. The password should be at least 16 characters long.
5. Make sure that the password and its confirmation match.
6. If the PHP script notices that the form is invalid, the form should not “forget” what the user entered before submitting. Hint: superglobals.

Put your solution in the folder ‘task3’.

Task 4: 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 ‘task4’.

Submission

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