



## Assignment 3 (HF, major subject)

---

*Due: Mon 09.11.2015; 12:00h (1 Week)*

### Goals

After doing these exercises,

- You will know how to use cookies and inform the users about them
- You can utilize sessions to persist data and make web apps stateful

### Task 1: Cookie Box

As you have heard in the tutorial, web site operators are required to inform the users about the use of cookies.

Create a re-usable “cookie-box” that displays a disclaimer at the bottom of the screen (fixed). It says that the user accepts cookies unless manually turning it off for this website. It then goes away if:

- The user hits an “X”-button in the top-right corner of the box.
- The user performs at least three clicks on the site.
- Scrolls to the bottom of the document.

The box should not reappear after any of the above has occurred. However, if the user leaves the site within a 10 second time frame, **no** cookie is stored in the browser.

The purpose of this box is to use it in any project of yours that relies on cookies. Complete this task with JavaScript and/or PHP.

Optional:

- Movements attract attention. Use CSS3 animations to make the cookie-box slide up from the bottom.

Put your solution in the folder ‘task1’.

### Task 2: Codebreaker

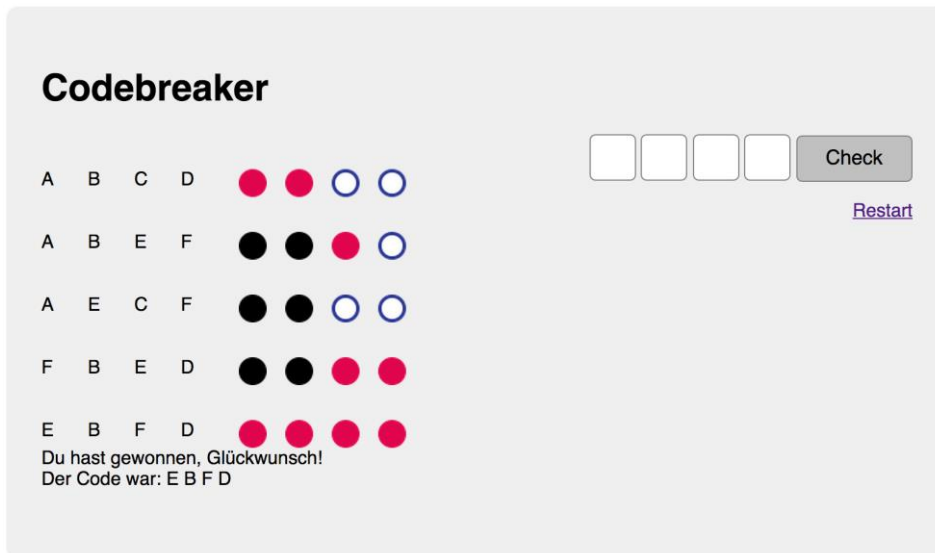
This is a quite work-intensive task, so you might want to start early during the week.

You are asked to create a game where the player has to guess a four-letter code. The code is formed of a random combination of the letters *A, B, C, D, E, F*, or *G*. Every letter can only appear once in it. The player enters his or her guess into a form and receives a hint until they found out the code. The hint has the following semantics:



- A **red dot** indicates that one of the guessed letters is part of the code and also at the correct position.
- A **black dot** indicates that one of the guessed letters is part of the code but not at the correct position
- A **white dot** indicates that a letter is not part of the code.

Example: The code is **EBFD** here. If the player enters these guesses (on the left), the web page should display the corresponding hints (colored dots):



Consider these requirements:

- Use PHP and sessions to keep track of all the attempts.
- Create a random four-letter code from the alphabet { A, B, C, D, E, F, G } when the user starts the game (i.e. visits the site for the first time).
- The user can guess at most ten times. If the code is incorrect then, the game is lost and the code is revealed to the player.
- The hints are presented as four colored dots. You can use regular image files for this.
- All previous guesses are displayed.
- The player can restart the game.

Optional:

- Sanity-check the player's input. For example: do not count a guess if it contains a letter twice or a letter from outside the limited alphabet.
- Show a leaderboard after the game is finished. The player can enter their name and save it. Use cookies for this.

Remember, you write code humans – not for machines. Make sure to comment your code as much as reasonable.

Put your solution in the folder 'task2'.



### Task 3: Cookie Theft

Explain in your own words what cookie theft is, how it is done and which countermeasures exist. Provide links to your sources.

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.