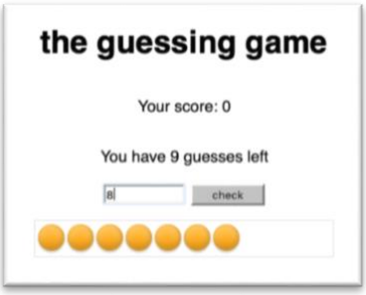
Assignment 7

Now that you know a little more about the Document Object Model, events and how forms work, combine those skills to solve this week’s lab assignment.

It's time for another guessing game, this time it's some sort of limbo version of guess-the-number-game. **Guess lower than or equal to the random number to score**.



the guessing game

Your score: 0

You have 9 guesses left

8

One turn in the game is played like this:

1. A random number of marbles are generated by the computer.
2. The player guesses how many marbles were generated (a number between 1 and 10).
3. The player clicks a button or presses the Return key to check the answer.
4. The random number of marbles are revealed, visually and,
5. if the guess is lower than, or equal to the random number of marbles, the score is increased by the guessed number.
6. The number of turns left in the game is decreased.

When 10 turns have been played, the game is over, and it must be impossible to guess again.

See video for a more comprehensive demonstration: https://youtu.be/JJ2XvuJzefY

Basic requirements

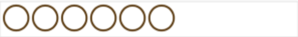
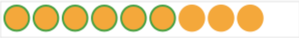
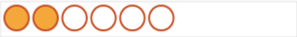
- The HTML should contain a form, an input text field and a submit button.
- When pressing the return key or clicking the submit button…
    - Number of turns left of the game should be displayed on the page,
    - A new random number should be generated,
    - The number entered in the text field should be compared to the random number
    - The random number should be represented as html elements, added to the page, using createElement and appendChild and setAttribute (if the random number is 5, five elements should appear on the page). Style them as you wish.
    - if the player's guess is equal to or less than the random number, the score should increase by the player's guess.
    - The current score should be updated on the page.
    - The input text field should be emptied and focused (marquee is blinking).
- When 10 guesses have been made, the form should stop working.

Presentation

1. **Name your files** `assignment7_yourUsername.html + yourUsername.css + yourUsername.js`
2. Compress them and upload the zip file to pingpong.
3. Present your solution to Johan or Jasmin at one of the labs (upload first!)

★ Grade 5 (deadline Tue Mar 5, 07:59)

Continuing from the basic requirements, your game should also meet these requirements:

1. When the player enters her guess in the input field, the number of marbles at the bottom of the page should reflect this, e.g. if the player guesses 4, then 4 outlined marbles are shown on the screen.
   
2. When the player checks the answer
   a. The random number is illustrated by filled marbles (just like the basic requirements)
   b. If the guess is lower than or equal to the random number, the outline color of the player guess turns green.
   
   c. If it turns out that the player guessed on a number too high, the outline is changed to red instead.
   

Theory

Head First JavaScript programming

Chapter 6 and 9 in the book.

Mozilla developer network

Events: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Building_blocks/Events

DOM elements: https://developer.mozilla.org/en-US/docs/Web/API/element

W3schools

DOM:	https://www.w3schools.com/js/js_htmldom.asp
Adding/removing elements:	https://www.w3schools.com/js/js_htmldom_nodes.asp
Event listeners:	https://www.w3schools.com/js/js_htmldom_eventlistener.asp
The input element:	https://www.w3schools.com/tags/tag_input.asp
Setting focus:	https://www.w3schools.com/jsref/met_input_time_focus.asp
Get input value:	https://www.w3schools.com/jsref/prop_text_value.asp

Slides

Lecture 6: https://slides.com/jkohlin/js-dom

Lecture 7: https://slides.com/jkohlin/js-dom2