Tic Tac Toe

Einführung

In dieser Übung wollen wir das Spiel Tic-Tac-Toe, auch bekannt als ‚Dodlschach' in JavaScript, HTML und CSS ohne Unterstützung von JavaScript Bibliotheken realisieren.

1. HTML Layout

Im ersten Schritt bauen wir die HTML und CSS Struktur für unser Spiel, in dem wir versuchen einen möglichen Stand des Spieles durch HTML darzustellen. Das Spiel besteht aus 3x3 Feldern, die mit einem HTML Table sehr gut abgebildet werden können. Als Markierung für besetzte Felder dienen uns die Großbuchstaben X und O.

Wir legen zuerst die HTML Datei mit der Tabelle in einer neuen HTML Datei, tictactoe.html, an:

tictactoe.html angularjs\assignments\tictactoe

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4      </head>
5      <body>
6          <h1>Tic Tac Toe</h1>
7
8          <table>
9              <tr>
10                 <td>X</td>
11                 <td>O</td>
12                 <td></td>
13             </tr>
14             <tr>
15                 <td></td>
16                 <td>O</td>
17                 <td></td>
18             </tr>
19             <tr>
20                 <td></td>
21                 <td></td>
22                 <td></td>
23             </tr>
24         </table>
25     </body>
26 </html>
```

2 CSS Styling

Damit unser Spielfeld mehr einem eigentlichen Spielfeld als einem halb-leeren HTML Table ähnelt, fügen wir eine CSS Datei - tictactoe.css - hinzu, die wir über ein <link> Element im <head> der HTML Datei referenzieren. Das <table> Element erweitern wir um ein class Attribut mit dem Wert 'gameboard'.

tictactoe.html angularjs\assignments\tictactoe

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <link rel="stylesheet" href="tictactoe.css">
5      </head>
6      <body>
7          <h1>Tic Tac Toe</h1>
8
9          <table class="gameboard">
10             <tr>
11                 <td>X</td>
12                 <td>O</td>
```

Die css Datei befüllen wir mit folgendem Inhalt. Dies setzt die Breite, Höhe, Border, Schriftgröße und Ausrichtung aller <td> Elemente.

tictactoe.css angularjs\assignments\tictactoe

```
1  .gameboard tr td {
2      width: 100px;
3      height: 100px;
4      border: solid black 2px;
5      font-size: 80px;
6      vertical-align: center;
7      text-align: center;
8  }
```

3 Benutzerinteraktion

Im 3. Schritt erweitern wir unsere Seite um eine Javascript datei, tictactoe.js, welche die eigentliche Spiellogik beinhalten wird. Damit wir auf Click-Events reagieren können, müssen wir für jedes Feld eine Methode im onclick- Attribut hinterlegen. Dies kann man entweder manuell in der HTML Datei machen oder programmatisch über JavaScript. Die Methode setupBoard iteriert über alle Felder eines Elementes 'gameboard', setzt den Inhalt auf Leer und setzt das Attribut 'onclick' auf eine Zeichenkette 'setField(x,y)' wobei x und y durch die aktuelle Spalte respektive Zeile ersetzt wird:

tictactoe.js angularjs\assignments\tictactoe

```
1  function setupBoard() {
2      var board = document.getElementsByClassName('gameboard')[0];
3      var rows = board.getElementsByTagName('tr');
4      for(var r = 0; r < rows.length; r++) {
5          var row = rows[r];
6          var fields = row.getElementsByTagName('td')
7          for(var c = 0; c < fields.length; c++) {
8              fields[c].innerHTML = '';
9              fields[c].setAttribute('onclick', 'setField(' + c + ',' + r + ')');
10         }
11     }
12 }
```

Damit wir das Spielfeld manipulieren können, erstellen wir eine Methode, die einen geziehlten Zugriff über x/y Koordinaten auf ein einzelnes Feld erlaubt. Wir nennen diese Methode getField:

tictactoe.js angularjs\assignments\tictactoe

```
13
14 function getField(x, y) {
15     var board = document.getElementsByClassName('gameboard')[0];
16     var row = board.getElementsByTagName('tr')[y];
17     return row.getElementsByTagName('td')[x];
18 }
```

Weiters merken wir uns in einer Variablen 'turn' welcher Spieler, X oder O, gerade an der Reihe ist. In der setField Methode (die wir vorher als Attribut gesetzt haben), holen wir uns das entsprechende Feld und setzen den Inhalt der Zelle über 'innerHTML' auf den string des aktuellen Spielers ('turn'). Schließlich setzen wir die 'turn' Variable jeweils auf den anderen Spieler. Am Ende rufen wir noch die setupBoard Methode auf, damit das Spielfeld auch wirklich initialisiert wird:

tictactoe.js angularjs-samples\assignments\tictactoe

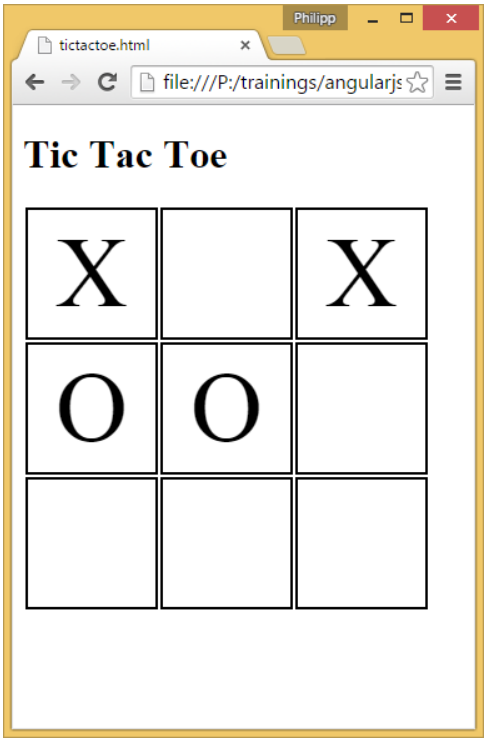
```
19
20 var turn = 'X';
21 function setField(x, y) {
22     if (!game[y][x] && !winner) {
23         getField(x, y).innerHTML = turn;
24         game[y][x] = turn;
25         turn = turn == 'X' ? 'O' : 'X';
26         checkWinner();
27     }
28 }
```

In der HTML Datei müssen wir die JS Datei nun noch laden. Die Stelle an der diese geladen wird ist in diesem Fall besonders relevant, da setupBoard davon ausgeht, dass das Spielfeld bereits im Dokument vorhanden ist. Wir setzen das Script tag daher an das Ende des Dokuments:

tictactoe.html angularjs\assignments\tictactoe

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <link rel="stylesheet" href="tictactoe.css">
5      </head>
6      <body>
7          <h1>Tic Tac Toe</h1>
8
9          <table class="gameboard">
10             <tr>
11                 <td>X</td>
12                 <td>O</td>
13                 <td></td>
14             </tr>
15             <tr>
16                 <td></td>
17                 <td>O</td>
18                 <td></td>
19             </tr>
20             <tr>
21                 <td></td>
22                 <td></td>
23                 <td></td>
24             </tr>
25         </table>
26
27         <script src="tictactoe.js"></script>
28     </body>
29 </html>
```

Nach einem Laden der HTML datei (vom Dateisystem aus) können wir die Felder mit einem einfachen Klick setzen:



4 Spiellogik

Momentan macht unsere Variante von tic-tac-toe nur begrenzt Freude: Man kann Felder überschreiben und auch gibt es keine Feststellung falls jemand das Spiel gewonnen hat.

Damit wir den aktuellen Spielstand mitverfolgen könne, legen wir ein Array von Arrays an, dass den Zeilen und Spalten des Feldes entspricht:

tictactoe.js angularjs\assignments\tictactoe

```
14 var game = [
15     [null, null, null],
16     [null, null, null],
17     [null, null, null]
18 ];
```

Wir fügen eine Methode hasWon() ein, die den Namen des Gewinners oder null ergibt, je nachdem ob es einen Gewinner gibt oder nicht:

tictactoe.js angularjs\assignments\tictactoe

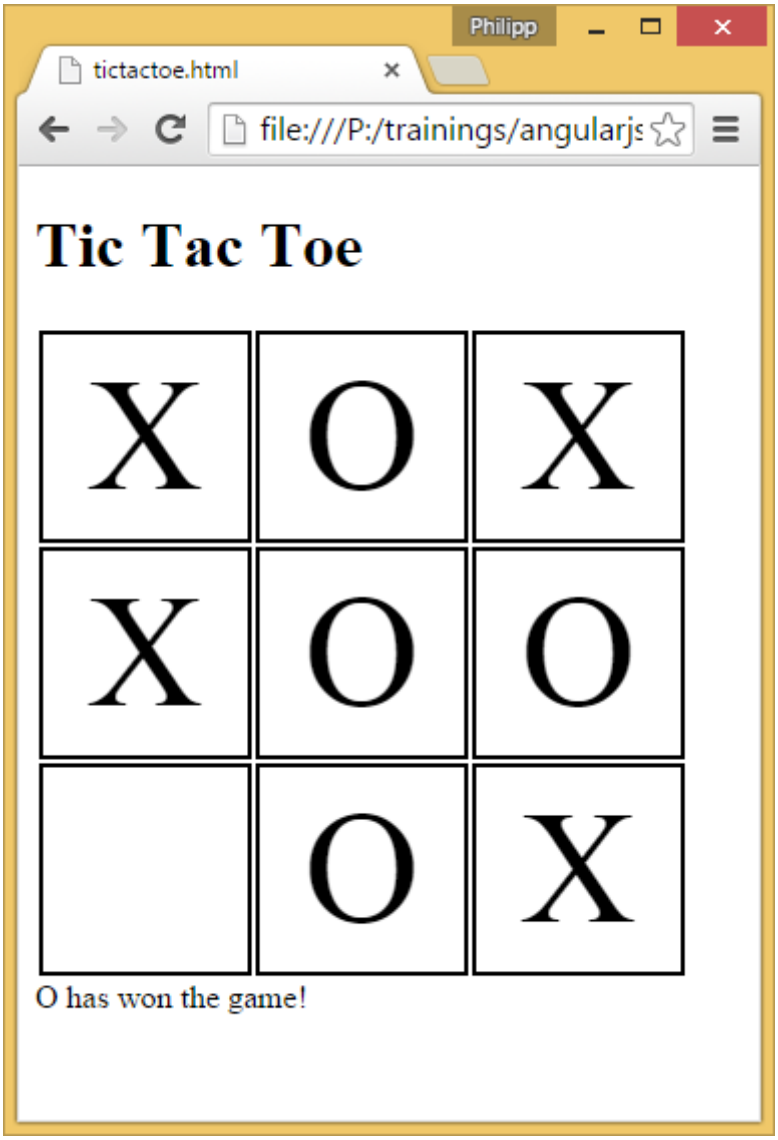
```
19
20 var winner = null;
21
22 function hasWon() {
23     var rowCheck = [
24         game[0][0] == game[0][1] && game[0][0] == game[0][2],
25         game[1][0] == game[1][1] && game[1][0] == game[1][2],
26         game[2][0] == game[2][1] && game[2][0] == game[2][2]
27     ];
28
29     var colCheck = [
30         game[0][0] == game[1][0] && game[0][0] == game[2][0],
31         game[0][1] == game[1][1] && game[0][1] == game[2][1],
32         game[0][2] == game[1][2] && game[0][2] == game[2][2]
33     ];
34
35     var diagCheck = [
36         game[0][0] == game[1][1] && game[0][0]== game[2][2],
37         game[2][0] == game[1][1] && game[2][0]== game[0][2]
38     ];
39

40     for (var i = 0; i < 3; i++) {
41         if (rowCheck[i]) {
42             return game[i][0];
43         }
44
45         if (colCheck[i]) {
46             return game[0][i];
47         }
48     }
49
50     for (var i = 0; i < 2; i++) {
51         if (diagCheck[i]) {
52             return game[1][1];
53         }
54     }
55
56     return null;
57 }
```

Die 'setField' Methode ändern wir so ab, dass überprüft wird, ob das aktuelle Feld bereits gesetzt ist und ob es bereits einen sieger (‚winner') gibt. Nur wenn es noch frei ist und noch kein Gewinner festgestellt werden konnte, setzen wir es wie bisher. Zusätzlich merken wir uns den gesetzten Wert in der 'game' Variablen und rufen die checkWinner() methode auf, die überprüft ob es einen Gewinner gibt und wenn dies der Fall ist, eine entsprechende Meldung setzt:

tictactoe.js angularjs\assignments\tictactoe

```
65 function checkWinner() {
66     var w = hasWon();
67     if (w) {
68         winner = w;
69         var elem = document.getElementsByClassName('winnermsg')[0];
70         elem.innerHTML = winner + ' has won the game!';
71     }
72 }
73
74 var turn = 'X';
75 function setField(x, y) {
76     if (!game[y][x] && !winner) {
77         getField(x, y).innerHTML = turn;
78         game[y][x] = turn;
79         turn = turn == 'X' ? 'O' : 'X';
80         checkWinner();
81     }
82 }
83
84 setupBoard();
```

Nach diesen Änderungen, können Sie den Browser aktualisieren und eine Runde ‚Dodl'-Schach in HTML/Javascript spielen ☺ Viel Freude damit!


Philipp
tictactoe.html
file:///P:/trainings/angularjs
Tic Tac Toe
X O X
X O O
O X
O has won the game!