# **Крестики-нолики (для продвинутых)**

# Цель урока

Сделать игру “крестики нолики”, уложившись в минимальное количество строк.

# Задачи

* Повторить пройденный материал
* Начать работать с event (события)
* Изучить indexOf, forEach
* Новый синтаксис стрелочные функции, тернарный оператор

# Ход урока

Всё начинается с html

Хоть стилей и немного, вынесем их в отдельный css файл, так как архитектура превыше всего

|  |
| --- |
| **<!DOCTYPE html> <html lang="en" dir="ltr">  <head>  <meta charset="utf-8">  <title>Tik Tak Toe</title>  </head>  <body>  <div class="container">  <div class="item">   </div>  <div class="item">   </div>  <div class="item">   </div>  <div class="item">   </div>  <div class="item">   </div>  <div class="item">   </div>  <div class="item">   </div>  <div class="item">   </div>  <div class="item">   </div>  </div>  <div class="isWinner">   </div>   <script type="text/javascript" src='script.js'> </script>    </body> </html>** |

CSS-код на флексах, для того, чтоб массив был одномерным. Если ребенок еще не проходил флексы, то просто скинуть стили, а если проходил, то писать стили самостоятельно:

|  |
| --- |
| **.container {  display:flex;  flex-wrap: wrap;  min-width:100px;  max-width: 400px;  }  .item {  border: 1px solid black;  border-radius: 5px;  padding:5px;  display:flex;  flex-basis: 25%;  justify-content: center;  align-items: center;  height: 50px;  }  .isWinner{  font-size: 20px;  font-weight: bold;  }** |

## События

Теперь займёмся JS. Давайте сделаем так, чтобы при нажатии на контейнер у нас срабатывал такой код:

|  |
| --- |
| **document.querySelector(".container").onclick = function () {  console.log("click"); };** |

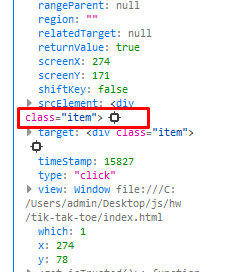
На этом месте можно показать ему синтаксис стрелочной функции, чтоб в будущем не пугался при виде необычного синтаксиса:

**document.querySelector(".container").onclick = () => {  
 console.log("click");  
};**

Отлично, теперь у нас при нажатии на контейнер всё срабатывает, а как бы нам сделать так, чтобы понимать на какую ячейку мы нажали? Для этого в javascript есть специальная переменная event, давайте выведем её в консоль и посмотрим, что получится:

|  |
| --- |
| **document.querySelector(".container").onclick = () => {  console.log(event); };** |

Видим, что в консоль выводится интересная штука, давайте её развернём и объясним студенту, что при каждом событии в эту переменную записывается вся информация про это событие: где оно произошло (clientX, clientY), что произошло (type) и с каким элементом произошло внутри таблицы (target):



Давайте теперь выведем этот элемент в консоль:

|  |
| --- |
| **document.querySelector("table").onclick = () => {  console.log(event.target); };** |

Давайте теперь внутрь этого элемента запишем X. Создадим переменную element - она будет хранить ячейку. При помощи уже знакомого нам способа .innerHTML поместим туда X:

|  |
| --- |
| **document.querySelector("table").onclick = () => {  let element = event.target;  element.innerHTML = "x"; };** |

## Смена хода

Теперь надо научиться делать так, чтобы менялись крестики и нолики - мы уже решали эту задачу в предыдущей программе. Создадим переменную, которая будет считать номер хода. Далее проверяем остаток деления номера хода на два и определяем, крестики или нолики сейчас ходят.

|  |
| --- |
| **let turn = 0; let char  document.querySelector("table").onclick = function () {  let element = event.target;  if (turn % 2 === 1) {  char = "o";  } else {  char = "x";  }**  **element.innerHTML = char;  turn++ };** |
|  |

А теперь сократим это с помощью тернарного выражения:

|  |
| --- |
| **let turn = 0; let char  document.querySelector("table").onclick = function () {  let element = event.target;  turn % 2 === 1 ? char = "o" : char = "x"**  **element.innerHTML = char;**  **turn++ };** |

Объясняем синтаксис, можно вместе с ним загуглить.

*условие выполняется ? что будет, если да : что будет, если нет*

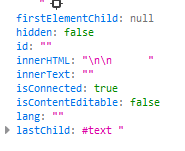
## Устраняем жульничество

Можно заметить, что нажимая на клетку, мы может переписать ее содержимое, так не должно быть. Поэтому мы сделаем так, чтобы в нее могло что то записаться только если она была пустой:

|  |
| --- |
| **let turn = 0; let char  document.querySelector(".container").onclick = () => {  let element = event.target;  turn % 2 === 1 ? char = "o" : char = "x"  if(element.innerHTML===''){  element.innerHTML = char;  turn++  }  console.log(element);  };** |

На этом этапе замечаем, что крестики и нолики перестали добавляться. Давайте разберемся из-за чего это происходит с помощью console.log(element)

Видим, что пустой innerHTML у флексов выглядит так:



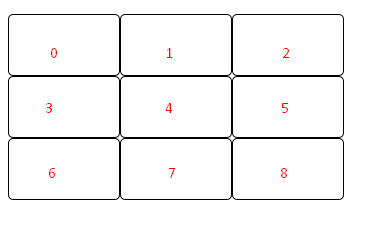
За то innerText идеально подходит! Заменяем его:

|  |
| --- |
| **let** char  document.querySelector(".container").onclick = () => {  **let** element = event.target;  turn % 2 === 1 ? char = "o" : char = "x"  **if**(element.innerText===''){  element.innerText = char;  turn++  } }; |

## 

## Проверка выигрыша

У наших клеток есть порядковый номер, который выглядит вот так:



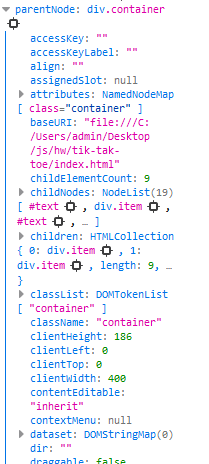
Мы создадим массив из девяти пустых строк и когда мы нажмем на n-ую клетку, крестик или нолик будет записываться в массив под тем же индексом. Нажали на нулевую по счету клетку - в нулевой элемент записался Х и тд.

|  |
| --- |
| **let** turn = 0; **let** char **let** list = [ '','','','','','','','','' ] document.querySelector(".container").onclick = () => {  **let** element = event.target;   turn % 2 === 1 ? char = "o" : char = "x"  **if**(element.innerText === ''){  element.innerText = char;  turn++  }  }; |

Нужно сделать так, чтоб javascript понимал, на какой элемент мы нажали. Для этого мы сначала обратимся к родителю элемента, на который мы нажали с помощью parentNode

|  |
| --- |
| **let** turn = 0; **let** char **let** list = [ '','','','','','','','','' ] document.querySelector(".container").onclick = () => {  **let** element = event.target;  **let** node = element.parentNode;  turn % 2 === 1 ? char = "o" : char = "x"  **if**(element.innerText === ''){  element.innerText = char;  turn++  } }; |
|  |

С помощью console,log посмотрим, что такое node



Нам отсюда нужен children, развернув которые мы видим, что каждый ребенок находится под определенным номером, начиная с нуля, но он внутри объекта HTMLCollection и что его развернуть, нужно использовать ...node.children

проверяем с помощью консоли

|  |
| --- |
| **let** turn = 0; **let** char **let** list = [ '','','','','','','','','' ] document.querySelector(".container").onclick = () => {  **let** element = event.target;   turn % 2 === 1 ? char = "o" : char = "x"  **if**(element.innerText === ''){  element.innerText = char;  **let** node = element.parentNode;  console.log(...node.children)  turn++  } }; |

Теперь у нас есть чистые элементы по порядку и чтоб элементы приобрели порядковый номер мы сделаем из них массив просто добаивив квадратные скобки к выражению:

|  |
| --- |
| **let** turn = 0; **let** char **let** list = [ '','','','','','','','','' ] document.querySelector(".container").onclick = () => {  **let** element = event.target;   turn % 2 === 1 ? char = "o" : char = "x"  **if**(element.innerText === ''){  element.innerText = char;  **let** node = element.parentNode;  console.log([...node.children])  turn++  } }; |

Помним, что нам нужен не элемент, а его порядковый номер. Обратиться к порядковому номеру можно с помощью IndexOf()

|  |
| --- |
| **let** turn = 0; **let** char **let** list = [ '','','','','','','','','' ] document.querySelector(".container").onclick = () => {  **let** element = event.target;   turn % 2 === 1 ? char = "o" : char = "x"  **if**(element.innerText === ''){  element.innerText = char;  **let** node = element.parentNode;  **let** i = ([...node.children].indexOf(element));  list[i] = char  turn++  } }; |
|  |

И с помощью строки list[i] = char закидываем наш крестик или нолик в пустую строчку в массиве.

Далее самое интересное, мы усовершенствуем эту часть кода с прошлых крестиков-ноликов:

|  |
| --- |
| **if** (area[0][0] == "X" && area[0][1] == "X" && area[0][2] == "X") {  **return** "X";  }  **if** (area[1][0] == "X" && area[1][1] == "X" && area[1][2] == "X") {  **return** "X";  }  **if** (area[2][0] == "X" && area[2][1] == "X" && area[2][2] == "X") {  **return** "X";  }  **if** (area[0][0] == "X" && area[1][0] == "X" && area[2][0] == "X") {  **return** "X";  }  **if** (area[0][1] == "X" && area[1][1] == "X" && area[2][1] == "X") {  **return** "X";  }  **if** (area[0][2] == "X" && area[1][2] == "X" && area[2][2] == "X") {  **return** "X";  }  **if** (area[0][0] == "X" && area[1][1] == "X" && area[2][2] == "X") {  **return** "X";  }  **if** (area[0][2] == "X" && area[1][1] == "X" && area[2][0] == "X") {  **return** "X";  }   **if** (area[0][0] == "0" && area[0][1] == "0" && area[0][2] == "0") {  **return** "0";  }  **if** (area[1][0] == "0" && area[1][1] == "0" && area[1][2] == "0") {  **return** "0";  }  **if** (area[2][0] == "0" && area[2][1] == "0" && area[2][2] == "0") {  **return** "0";  }  **if** (area[0][0] == "0" && area[1][0] == "0" && area[2][0] == "0") {  **return** "0";  }  **if** (area[0][1] == "0" && area[1][1] == "0" && area[2][1] == "0") {  **return** "0";  }  **if** (area[0][2] == "0" && area[1][2] == "0" && area[2][2] == "0") {  **return** "0";  }  **if** (area[0][0] == "0" && area[1][1] == "0" && area[2][2] == "0") {  **return** "0";  }  **if** (area[0][2] == "0" && area[1][1] == "0" && area[2][0] == "0") {  **return** "0";  } |

Какая часть кода здесь повторяется, а какая нет? Правильно изменяется только индекс элемента массива. Значит мы вынесем все возможные комбинации в сторону:

|  |
| --- |
| **let** turn = 0 **let** list = [ '','','','','','','','','' ] **let** combs = [  [0, 1, 2],  [3, 4, 5],  [6, 7, 8],  [0, 3, 6],  [1, 4, 7],  [2, 5, 8],  [0, 4, 8],  [2, 4, 6],  ]  document.querySelector('.container').onclick = () => {  **let** element = event.target  **let** char  turn%2 === 1 ? char = 'o' : char = 'x'  **if**(element.innerText=== '') {  element.innerText = char  **let** node = element.parentNode;  **let** i = ([...node.children].indexOf(element));   list[i] = char   turn++  } } |

Теперь нужно перебрать весь массив попутно сравнивая элемент массива list с комбинациями. Если все три равны, например list[0] и list[1] и list[2] равны одному и тому же, тот чему они равны и победил.

|  |
| --- |
| **let** turn = 0 **let** list = [ '','','','','','','','','' ] **let** combs = [  [0, 1, 2],  [3, 4, 5],  [6, 7, 8],  [0, 3, 6],  [1, 4, 7],  [2, 5, 8],  [0, 4, 8],  [2, 4, 6],  ] *// комбо элементов для победы* document.querySelector('.container').onclick = () => {  **let** element = event.target  **let** char  turn%2 === 1 ? char = 'o' : char = 'x'  **if**(element.innerText=== '') {  element.innerText = char  **let** node = element.parentNode;  **let** i = ([...node.children].indexOf(element)); *//i - индекс флекс элемента*  list[i] = char *// пуш в массив*  **for** (**let** e = 0; e < combs.length; e++) {  **if** (list[combs[e][0]] === char && list[combs[e][1]] === char && list[combs[e][2]] === char) {  document.querySelector('.isWinner').innerHTML = char + ' - WINNER'  }  }  turn++  } } |

Теперь добавим инноваций, цикл for можно заменить методом массива forEach (для каждого).

Он используется так:

array = [1,2,3,4]

array.forEach(function(item){

item\*2

})

Он вернет каждый элемент умноженный на два.

Пробуем применять к нашей функции:

|  |
| --- |
| **let** turn = 0 **let** list = [ '','','','','','','','','' ] **let** combs = [  [0, 1, 2],  [3, 4, 5],  [6, 7, 8],  [0, 3, 6],  [1, 4, 7],  [2, 5, 8],  [0, 4, 8],  [2, 4, 6],  ]  document.querySelector('.container').onclick = () => {  **let** element = event.target  **let** char  turn%2 === 1 ? char = 'o' : char = 'x'  **if**(element.innerText=== '') {  element.innerText = char  **let** node = element.parentNode;  **let** i = ([...node.children].indexOf(element));   list[i] = char  **for** (**let** e = 0; e < combs.length; e++) {  **if** (list[combs[e][0]] === char && list[combs[e][1]] === char && list[combs[e][2]] === char) {  document.querySelector('.isWinner').innerHTML = char + ' - WINNER'  }  }   turn++  } } |

Наслаждаемся кодом

## Домашнее задание

1. Читать про тернарные выражения, стрелочные функции, метод массива forEach
2. Добавить цветов в нашу игру
3. Задание в приложении

Ответ на задание:

// Дан массив с числами. Найдите сумму этих чисел используя forEach

|  |
| --- |
| **let** arr = [2,3,4,5] **let** sum = 0 arr.forEach((e) => sum+=e) console.log(sum); *//14* |