Caccia alle talpe

```
var N = 5;
var score = 0;
var symbols = { "grass.png": 0, "head.png": 100, "rear.png": -200 };

function createMole() {
  var mole = document.createElement("img");
  mole.setAttribute("src", Object.keys(symbols)[0]);
  mole.addEventListener("click", function () {
    display.innerHTML = score += symbols[mole.getAttribute("src")];
    if (score < 0) document.body.innerHTML = "hai perso !";
    if (score >= 1000) document.body.innerHTML = "hai vinto !";
  });
  setInterval(function () {
    mole.setAttribute(
      "src",
      Object.keys(symbols)[Math.floor(Math.random() * 3)]
    );
  }, 1000);
  document.body.appendChild(mole);
}

for (var i = 0; i < N; i++) {
  for (var j = 0; j < N; j++) createMole();
  document.body.appendChild(document.createElement("br"));
}
var display = document.createElement("div");
document.body.appendChild(display);
```

Campo Minato

```
let N = 10;
let K = 5;
let tiles = [];

for (
  let i = 0;
  i < N;
  ++i, document.body.appendChild(document.createElement("br"))
) {
  for (let j = 0; j < N; ++j) {
    let tile = document.body.appendChild(document.createElement("img"));
    tile.setAttribute("src", "piastrella.gif");
    tile.mina = i * N + j < K;
    tile.vicini = [];
```

```
      for (let y = Math.max(0, i - 1); y <= Math.min(N - 1, i + 1); y++) {
        for (let x = Math.max(0, j - 1); x <= Math.min(N - 1, j + 1); ++x) {
          tile.vicini.push(y * N + x);
        }
      }
      tile.addEventListener("click", function (e) {
        click(e.target);
      });
      tiles.push(tile);
    }
  }

  for (let i = tiles.length - 1; i >= 0; i--) {
    let j = Math.floor(Math.random() * (i - 1));
    [tiles[i].mina, tiles[j].mina] = [tiles[j].mina, tiles[i].mina];
  }

  function click(t) {
    if (t.getAttribute("src") == "piastrella.gif") {
      if (t.mina) {
        for (let i in tiles) {
          if (tiles[i].mina) {
            tiles[i].setAttribute("src", "mina.gif");
          }
        }

        document.body.appendChild(document.createTextNode("Hai perso!"));
      } else {
        let mine = 0;

        for (let i in t.vicini) {
          mine += tiles[t.vicini[i]].mina;
        }

        t.setAttribute("src", "sq" + mine + ".gif");

        for (let i in t.vicini) {
          if (mine == 0) {
            click(tiles[t.vicini[i]]);
          }
        }

        if (++K == N * N) {
          document.body.appendChild(document.createTextNode("hai vinto "));
        }
      }
    }
  }
```

Ingrandimento cerchi

```html
<html>
  <body>
    <script>
      var svg = {};
      var svgHeight = null;
      var svgWidth = null;
      var circleRMax = 40;
      var actualR = circleRMax;
      var direction = true;
      var speed = 2;
      var circles = new Array();
      window.onload = main;
      function main() {
        var div = document.createElement("div");
        svg = document.createElementNS("http://www.w3.org/2000/svg",
"svg");
        svg.setAttribute("width", 500);
        svg.setAttribute("height", 500);
        svg.style.border = "1px black solid";
        div.appendChild(svg);
        document.getElementsByTagName("body")[0].appendChild(div);

        window.onclick = function (event) {
          addCircle(event.x, event.y, actualR);
        };

        window.setInterval(function () {
          actualR += speed * (direction ? -1 : 1);
          if (actualR <= 0 || actualR >= circleRMax) direction =
!direction;
          for (var i = 0; i < circles.length; i++)
            circles[i].setAttribute("r", actualR);
        }, 50);
      }
      function addCircle(x, y, r) {
        var circle = document.createElementNS(
          "http://www.w3.org/2000/svg",
          "circle"
        );
        circle.setAttribute("cx", x);
        circle.setAttribute("cy", y);
        circle.setAttribute("r", r);
        circle.setAttribute("fill", "green");
        circles.push(circle);
        svg.appendChild(circle);
      }
    </script>
  </body>
</html>
```

Ingrandimento valori

```html
<!DOCTYPE html>
<head>
  <title>Soluzione secondo esercizio compito 5/5/2017:</title>
</head>

<body>
  <h1>Hi <i>to all</i> people</h1>

  <h3>That's all</h3>

  <script>
    for (var level = 6; level > 0; level--) {
      var current = document.querySelectorAll("h" + level);

      for (var i = 0; i < current.length; i++) {
        var newHeader = document.createElement(
          "h" + (level + 1 > 6 ? 6 : level + 1)
        );
        newHeader.innerHTML = current[i].innerHTML.toUpperCase();
        current[i].parentNode.replaceChild(newHeader, current[i]);
      }
    }
  </script>
</body>
```

Menu pizze

```js
function createRow(nameText, ingredientsText, priceText, total) {
  var name = document.createElement("td");
  name.innerHTML = nameText;
  name.addEventListener("mouseover", function () {
    name.innerHTML = ingredientsText;
  });
  name.addEventListener("mouseout", function () {
    name.innerHTML = nameText;
  });

  var price = document.createElement("td");
  price.innerHTML = priceText;
  price.addEventListener("click", function () {
    total.innerHTML = parseFloat(priceText) + parseFloat(total.innerHTML);
  });

  var row = document.createElement("tr");
  row.appendChild(name);
  row.appendChild(price);
  return row;
}

var menus = document.getElementsByClassName("menu");

while (menus.length > 0) {
  var menu = menus[0];
  var table = document.createElement("table");
  var total = document.createElement("div");
  total.innerHTML = "0";
  menu.parentNode.insertBefore(table, menu);
  menu.parentNode.insertBefore(total, menu);
  menu.parentNode.removeChild(menu);

  var divs = menu.getElementsByTagName("div");
  while (divs.length > 0) {
    var blocks = divs[0].textContent.split(";");
    table.appendChild(createRow(blocks[0], blocks[1], blocks[2], total));
    menu.removeChild(divs[0]);
  }
}
```

Query Tempo

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <title>Documento senza titolo</title>
  </head>
  <body>
    <script>
      failed = false;
      function get(url, callback, failure, obj) {
        var request = new XMLHttpRequest();
        request.open("GET", url);
        request.onreadystatechange = function () {
          if (request.readyState === 4 && request.status === 200) {
            var type = request.getResponseHeader("content-type").split("
")[0];
            if (type == "application/json")
              callback(JSON.parse(request.responseText), obj);
            else failure("Formato risposta sconosciuto");
          }
        };
        request.send(null);
      }
      window.onload = main;
      function main() {
        var elements = document.getElementsByClassName("wind");
        for (var i = 0; i < elements.length; i++) {
          get(
            "http://api.openweathermap.org/data/2.5/weather?q={" +
              elements[i].innerHTML,
            function (obj, element) {
              element.innerHTML = obj["wind"]["speed"];
            },
            function (err) {
              if (!failed) {
                failed = true;
                alert("Errore di comunicazione:\n" + err);
              }
            },
            elements[i]
          );
        }
      }
    </script>
    <div class="wind">Venice,it</div>
    <div class="wind">Rome,it</div>
  </body>
</html>
```

TicTacToe

```
<html>
  <head>
    <meta charset="UTF-8" />
    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0"
/>
    <link rel="stylesheet" href="style.css" />
    <link rel="preconnect" href="https://fonts.gstatic.com" />
    <link
      href="https://fonts.googleapis.com/css2?family=Itim&display=swap"
      rel="stylesheet"
    />
    <script src="script.js"></script>
    <title>Tic-Tac-Toe</title>
  </head>

  <body>
    <main class="background">
      <section class="title">
        <h1>Tic Tac Toe</h1>
      </section>
      <section class="display">
```

```
          Player <span class="display-player playerX">X</span>'s turn
        </section>
        <section class="container">
          <div class="tile"></div>
          <div class="tile"></div>
          <div class="tile"></div>
          <div class="tile"></div>
          <div class="tile"></div>
          <div class="tile"></div>
          <div class="tile"></div>
          <div class="tile"></div>
          <div class="tile"></div>
        </section>
        <section class="display announcer hide"></section>
        <section class="controls">
          <button id="reset">Reset</button>
        </section>
      </main>
      <script>
          window.addEventListener("DOMcontentLoaded", () => {
              const tiles = Array.from(document.querySelectorAll(".tile"));
              const playerDisplay = document.querySelector(".display-
  player");
              const resetButton = document.querySelector("#reset");
              const announcer = document.querySelector(".announcer");

              let board = ["", "", "", "", "", "", "", "", ""];
              let currentPlayer = "X";
              let isGameActive = true;

              const PLAYERX_WON = "PLAYERX_WON";
              const PLAYER_O_WON = "PLAYER_O_WON";
              const TIE = "TIE";

              const winningConditions = [
                  [0, 1, 2],
                  [3, 4, 5],
                  [6, 7, 8],
                  [0, 3, 6],
                  [1, 4, 7],
                  [2, 5, 8],
                  [0, 4, 8],
                  [2, 4, 6],
              ];

              function handleResultValidation() {
                  let roundWon = false;

                  for (let i = 0; i < 8; ++i) {
                  const winCondition = winningConditions[i];
                  const a = board[winCondition[0]];
                  const b = board[winCondition[1]];
                  const c = board[winCondition[2]];
```

```
                if (a === "" && b === "" && c === "") {
                    continue;
                }

                if (a == b && b == c) {
                    roundWon = true;
                    break;
                }
                }

                if (roundWon) {
                announce(currentPlayer === "X" ? PLAYERX_WON :
PLAYER_O_WON);
                isGameActive = false;
                return;
                }

                if (!board.includes("")) {
                announce(TIE);
                }
            }

            const announce = (type) => {
                switch (type) {
                case PLAYER_O_WON:
                    announcer.innerText = "Player <span class
='playerO'>O</span>>";
                    break;
                case PLAYERX_WON:
                    announcer.innerText = "Player <span class =
'playerX'>X</span>>";
                    break;
                case TIE:
                    announcer.innerText = "Tie";
                }

                announcer.classList, remove("hide");
            };

            const isValidAction = (tile) => {
                if (tile.innerText === "X" || tile.innerText === "O") {
                return false;
                }

                return true;
            };

            const updateBoard = (index) => {
                board[index] = currentPlayer;
            };

            const changePlayer = () => {
                playerDisplay.classList.remove(`player${currentPlayer}`);
                currentPlayer = currentPlayer === "X" ? "O" : "X";
                playerDisplay.innerText = currentPlayer;
                playerDisplay.classList.add(`player${currentPlayer}`);
            };

            const userAction = (tile, index) => {
                if (isGameActive && isValidAction(tile)) {
                tile.innerText = currentPlayer;
                tile.classList.add(`player${currentPlayer}`);
                updateBoard(index);
                handleResultValidation();
                changePlayer();
                }
            };

            const resetBoard = () => {
                board = ["", "", "", "", "", "", "", "", ""];
                isGameActive = true;
                announcer.classList.add("hide");

                if (currentPlayer === "O") {
                changePlayer();
                }

                tiles.forEach((tile) => {
                tile.innerText = "";
                tile.classList.remove("playerX");
                tile.classList.remove("playerO");
                });
            };

            tiles.forEach((tile, index) => {
                tile.addEventListener("click", () => userAction(tile,
index));
            });

            resetButton.addEventListener("click", resetBoard);
        });
    </script>
  </body>
</html>
```