Aplikacje Internetowe 1

Sprawozdanie – Lab 4 – Zaawansowane JS w przeglądarce

Data laboratorium: ……

Autor: Nazwisko Imię

Grupa:

Wersja: 2020/2021-zima

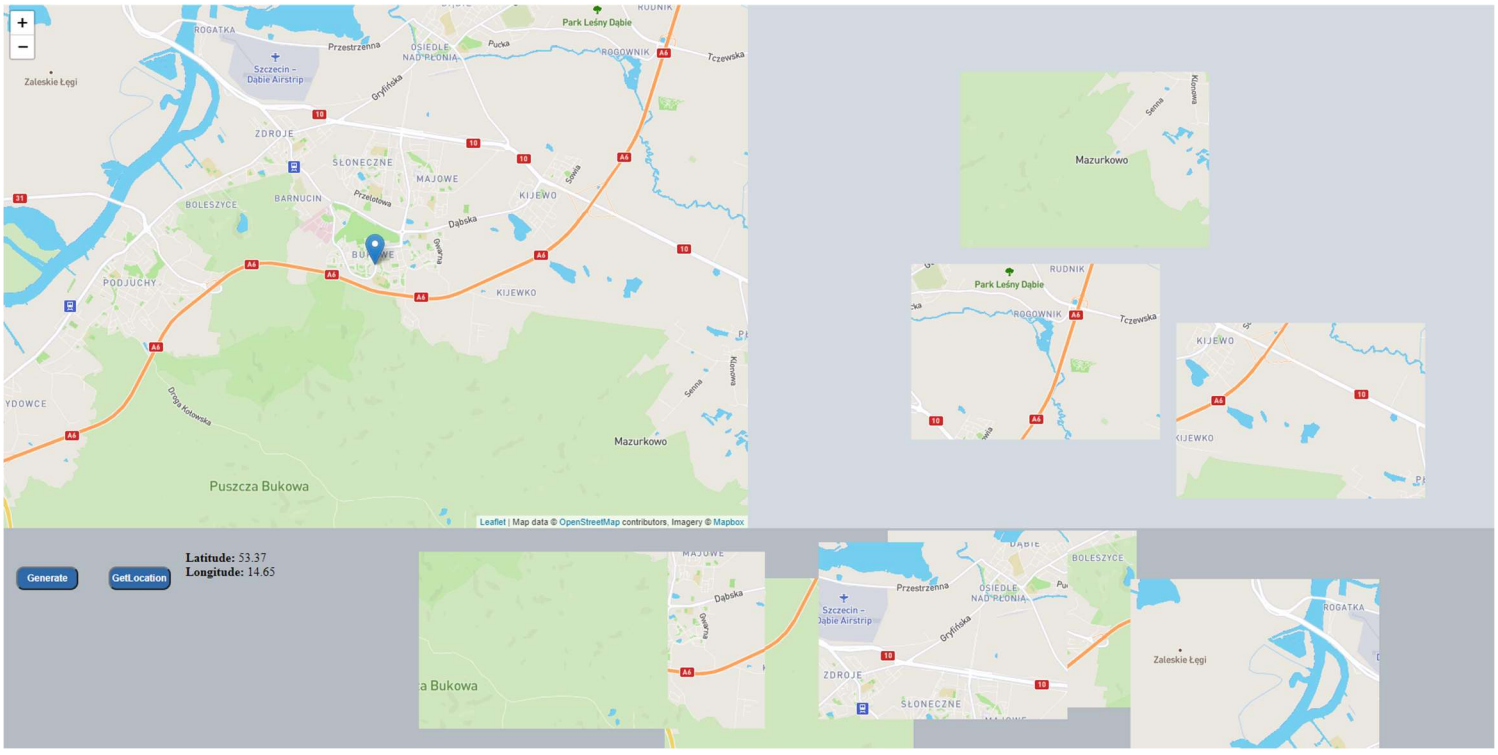
Wprowadzenie

Tutaj należy umieścić 2-3 zdania opisujące zawartość sprawozdania i streszczające przebieg laboratorium. Wypełnienie tej rubryki warunkuje dalsze sprawdzenie sprawozdania.

Zadaniem było napisać stronę która będzie wyświetlała mapę i tworzyła puzzle na jej podstawie. Do wyświetlania mapy wykorzystano bibliotekę „leaflet”.

Strona HTML ze wszystkimi elementami

Należy przedstawić HTML ze wszystkimi wymaganymi elementami. 1 punkt.



```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="utf-8"/>
    <meta name="viewport" content="width=device-width, initial-scale=1"/>
    <title><< PUZZLE >></title>
    <link rel="stylesheet" href="puzzle.css"/>
    <link rel="stylesheet" href="leaflet-1.7.1/leaflet.css"/>
    <script src="leaflet-1.7.1/leaflet.js"></script>
    <script src="leaflet-1.7.1/leaflet-image.js"></script>
    <script defer src="puzzle.js"></script>

</head>

<body>
<div id="container">

    <div id="top">
        <div id="map"></div>
        <div id="puzzle"></div>
    </div>

    <div id="bottom">
        <div id='buttons'>
            <button id="generate" class='button' onclick="generatePieces()">Generate</button>
            <button id="getLocation" class='button' onclick="getLocation()">GetLocation</button>
            <div>
            <strong>Latitude:</strong> <span id="latitude"></span><br>
            <strong>Longitude:</strong> <span id="longitude"></span><br>
            </div>
        </div>
    </div>

</div>
</body>
```

Dynamiczna mapa

Należy udokumentować działanie dynamicznej mapy, z możliwością zmiany lokalizacji i skali. 1 punkt.

```
var map = L.map('map').setView([51.505, -0.09], 13);
L.tileLayer('https://api.mapbox.com/styles/v1/{id}/tiles/{z}/{x}/{y}?access_token={accessToken}', {
    attribution: 'Map data &copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributor
    maxZoom: 18,
    id: 'mapbox/streets-v11',
    tileSize: 512,
    zoomOffset: -1,
    accessToken: 'pk.eyJ1IjoiZGF3aWR6dXQiLCJhIjoiY2tpYnMzZDU0MGQ1cDJycGV3Mm9wNGV4YyJ9.cpMFsNQVlPtHcyxxj6LioA'
}).addTo(map);
```

Pobieranie geolokalizacji

Należy przedstawić jak pobierana jest bieżąca lokalizacja użytkownika i centrowana jest mapa. 1 punkt.

...

```
let marker = L.marker([51.505, -0.09]);
function getLocation() {
    if (!navigator.geolocation) {
        alert("Geolocation not available.");
    }
    map.removeLayer(marker)
    navigator.geolocation.getCurrentPosition((position) => {
        xCor = position.coords.latitude;
        yCor = position.coords.longitude;
        document.getElementById("latitude").innerText = xCor.toFixed(2);
        document.getElementById("longitude").innerText = yCor.toFixed(2);
        map.panTo([xCor, yCor])
        marker = L.marker([xCor, yCor]);
        map.addLayer(marker);
    }, (positionError) => {
        console.error(positionError);
    }, {
        enableHighAccuracy: false
    });
}
```

Pobieranie mapy rastrowej

Należy zademonstrować funkcjonalność, w której po kliknięciu przycisku, zawartość mapy dynamicznej pobrana zostanie jako statyczny obraz do dalszego wykorzystania. 1 punkt.

```
leafletImage(map, function (err, canvas){
    if(err)
        console.log(err);
        canvas = canvas;
    console.log(canvas.width, canvas.height)

    let bottomDivRect = document.getElementById("bottom").getBoundingClientRect();
    let bPos = new vec2(bottomDivRect.left, bottomDivRect.top);
    let pSize = new vec2(canvas.width/puzzleCols, canvas.height/puzzleRows)
```

Podział mapy rastrowej na puzzle

Należy zademonstrować podział mapy rastrowej na 16 pomieszanych części. 1 punkt.

```
for(let col = 0; col<puzzleCols; col++)
for(let row = 0; row<puzzleRows; row++)
{
    let id = col*puzzleCols+row;
    let cPos = new vec2(pSize.x*col, pSize.y*row)
    let rPos = new vec2(getRandom(bPos.x, do  let cPos: vec2  lement.clientWidth-pSize.x),
    var puzzle = new PuzzlePiece(id, pSize, cPos, rPos, null)
    let puzzleCanvas = createPuzzlePiece(puzzle, canvas)
    document.body.appendChild(puzzleCanvas)
    pieces[id] = puzzle;
}
```

```
function createPuzzlePiece(puzzle, canvas)
{
    const placeholder = document.createElement('div');
    let pctPos = absToPct(puzzle.currentPos)
    let pctSize = absToPct(puzzle.size)
    placeholder.innerHTML = `<canvas class="puzzlePiece draggable" width="${puzzle.size.x}" height="${puzzle.size.y}" id="piece${puzzle.id}"
    var puzzleCanvas = placeholder.firstElementChild;
```

```
var ctx = puzzleCanvas.getContext('2d');
// ctx.fillStyle = 'rgb(200, 0, 0)';
// ctx.fillRect(0, 0, puzzleCanvas.width, puzzleCanvas.height);
console.log("Creating puzzle at pos: "+ puzzle.size.x+", "+puzzle.size.y)
ctx.drawImage(canvas, puzzle.canvasPos.x, puzzle.canvasPos.y, puzzle.size.x, puzzle.size.y, 0, 0, puzzle.size.x, puzzle.size.y);
return puzzleCanvas;
```

Drag & drop

Należy udokumentować funkcjonowanie mechanizmu drag & drop do przestawiania puzzli. 1 punkt.

```
puzzleCanvas.ondragstart = function() {
    return false;
};
puzzleCanvas.onmousedown = function(event) {
    if(!puzzleCanvas.draggable)
        return false;

    let shiftX = event.clientX - puzzleCanvas.getBoundingClientRect().left;
    let shiftY = event.clientY - puzzleCanvas.getBoundingClientRect().top;

    puzzleCanvas.style.zIndex = 1000;

    moveAt(event.pageX, event.pageY);

    function moveAt(pageX, pageY) {
        let v = new vec2(clamp(pageX - shiftX, 0, document.documentElement.clientWidth-puzzleCanvas.width),
                         clamp(pageY - shiftY, 0, document.documentElement.clientHeight-puzzleCanvas.height));
        puzzle.currentPos = v;
        puzzleCanvas.style.left = absToPct(v).x + '%';
        puzzleCanvas.style.top  = absToPct(v).y + '%';
    }

    function onMouseMove(event) {
        moveAt(event.pageX, event.pageY);
    }

    document.addEventListener('mousemove', onMouseMove);

    document.onmouseup = function() {
        document.removeEventListener('mousemove', onMouseMove);
        puzzleCanvas.onmouseup = null;
        if(Math.abs(puzzle.currentPos.x-canvas.width-puzzle.canvasPos.x)<50 &&
           Math.abs(puzzle.currentPos.y-puzzle.canvasPos.y)<50)
        {
            let v = new vec2(clamp(puzzle.canvasPos.x+canvas.width, 0, document.documentElement.clientWidth-puzzleCanvas.width),
                             clamp(puzzle.canvasPos.y, 0, document.documentElement.clientHeight-puzzleCanvas.height));
            puzzle.currentPos = v;
            puzzleCanvas.style.left = absToPct(v).x + '%';
            puzzleCanvas.style.top  = absToPct(v).y + '%';
            puzzleCanvas.setAttribute('draggable', 'false');
        }
        checkForCompletion()
    };
```

Wykrywanie dobrze ułożonej mapy

Należy udokumentować, że skrypt rozpoznaje moment, w którym mapa została poprawnie ułożona. Przykładowo z wykorzystaniem komunikatu w konsoli. 1 punkt.

```
function checkForCompletion(){
    var elements = document.getElementsByClassName("puzzlePiece");
    var complete = true
    console.log("checking")
    for(const element of elements)
        if(element.draggable)
            complete = false;

    if(complete)
        trySendNotification("Well done")
}
```

Notyfikacja po dobrze ułożonej mapie

Należy udokumentować, że skrypt po poprawnym ustawieniu puzzli wyświetli notyfikację z wykorzystaniem Notification API. 1 punkt.

Linki i uwagi

W tej sekcji należy umieścić link do działającej aplikacji listy zadań oraz dowolne uwagi.

http://www.ai-labo1.ml/puzzle/puzzle.html