Laboratorul 08. Fitbit SDK - Application

La fel ca în laboratorul trecut, target-ul cu care vom lucru este un produs Fitbit, și anume **Fitbit Versa 3** sau **Fitbit Sense** (produse lansate anul acesta). Vom utiliza *Fitbit Software Development Kit* (*SDK*) pentru a dezvolta o aplicație (app) în acest laboratorul, folosind mediul de dezvoltare pus la dispoziție de către Fitbit (Fitbit Studio) și simulatorul (Fitbit OS Simulator). Documentația pentru dezvoltatorii de aplicații Fitbit se poate găsi pe https://dev.fitbit.com/ [https://dev.fitbit.com/].

O aplicație este de obicei un program specializat cu un scop clar (e.g. aplicație de exerciții fitness, aplicație de alarme/timere etc.), spre deosebire de o față de ceas [https://dev.fitbit.com/build/guides/clockfaces/] care afișează în principiu ora și câteva statistici ca utilizatorul să fie informat rapid când folosește smartwatch-ul.

Setup

Setup-ul este același ca în laboratorul trecut (laboratorul 7 [https://ocw.cs.pub.ro/courses/si/laboratoare/07]).

Vă recomandăm să folosiți **Windows** sau **macOS** pentru aceste laboratoare!

Exerciții

În cadrul simulatorului setați **Versa 3** sau **Sense** ca de tip device (Settings → Device Type). Vom folosi versiunea 5.0 a SDK-ului.

0. Ne propunem să creăm o aplicație cu luminițe de Crăciun (cea din GIF-ul de mai jos). Aplicația va avea trei butoane care pornesc/opresc un mic spectacol de lumini. Pentru aceasta creați un nou proiect în Fitbit Studio [https://studio.fitbit.com/] și alegeți template-ul *Minimal*. Descărcați template-ul pentru laborator de aici [https://drive.google.com/file/d/18Yg3cV50ZhMa9pV4YM9Y5tmJ2NIw37D8/view?usp=sharing] și aplicați-l în cadrul proiectului. Asigurați-vă că în `package.json` aveți setat tipul proiectului ca aplicație (**Type** → **App**). Deocamdată `app/index.js` este gol, iar proiectul nu va fi buildat cu succes.

Exercițiile din laborator se rezolvă în fișierul `app/index.js`. Template-ul conține comentarii de tip *TODO* pentru fiecare subpunct al exercițiilor. Vom utiliza doar Device APIs, documentația lor se află aici [https://dev.fitbit.com/build/reference/device-api/]. Limbajul de programare utilizat este Javascript, dar ne vom limita la funcționalitățile de bază.

1. Deocamdată `app/index.js` este gol, iar proiectul nu va fi buildat cu succes. Pentru acest exercițiu ne propunem să avem referințe către toate elementele din `index.view` și variabile pentru a putea manipula mai ușor starea luminițelor de Crăciun.

1. obțineți referințele către globuri și adăugați-le într-un array.

```
import document from "document";

// Get all the globes references by the ID defined in index.view
const globes = [
  document.getElementById("bot1"),
  document.getElementById("bot2"),
  document.getElementById("bot3"),
  document.getElementById("bot4"),
  document.getElementById("bot5"),
  document.getElementById("botmid1"),
  document.getElementById("botmid2"),
  document.getElementById("botmid3"),
  document.getElementById("botmid4"),
  document.getElementById("mid1"),
  document.getElementById("mid2"),
  document.getElementById("mid3"),
  document.getElementById("topmid"),
  document.getElementById("top"),
];
```

2. obțineți referințele către cele trei butoane.

```
const blinkLightsButton = document.getElementById("lights-1");
const slideLightsButton = document.getElementById("lights-2");
const customLightsButton = document.getElementById("lights-3");
```

3. definiți două array-uri care să conțină culorile pentru stările când luminițele globurilor sunt aprinse, respectiv când acestea sunt închise.

```
// Colors for each globe's light turned on
const globeOnColor = [
  "#DC143C",
  "#FFA500",
  "#DC143C",
  "#FFA500",
  "#DC143C",
  "#FFA500",
  "#DC143C",
  "#FFA500",
  "#DC143C",
  "#FFA500",
  "#DC143C",
  "#FFA500",
  "#FFA500",
  "#DC143C",
];

// Colors for each globe's light turned off
const globeOffColor = [
  "#CD5C5C",
  "#EEE8AA",
  "#CD5C5C",
  "#EEE8AA",
  "#CD5C5C",
  "#EEE8AA",
  "#CD5C5C",
  "#EEE8AA",
  "#CD5C5C",
  "#EEE8AA",
  "#CD5C5C",
  "#EEE8AA",
  "#EEE8AA",
  "#CD5C5C",
];
```

4. creați un array în care să țineți evidența globurilor aprinse/stinse.

```
const globeLightsOn = [
  false,
  false,
  false,
```

```
  false,
  false,
  false,
  false,
  false,
  false,
  false,
  false,
  false,
  false,
  false,
];
```

Explicații:

- Pentru a putea obține referințele către elementele UI este nevoie de Document API [https://dev.fitbit.com/build/reference/device-api/document/], în care avem acces la funcția **getElementById()**.
- Puteți folosi și alte culori dacă doriți. Pentru inspirație puteți intra aici [https://dev.fitbit.com/build/guides/user-interface/css/].

2. Pentru a putea face tranziția între jocurile de lumini avem nevoie de posibilitatea de a le stinge pe toate întâi.

1. definiți funcția **turnOffLights()** care stinge luminițele (setează culorile pe starea de luminițe închise).

```
function turnOffLights() {
  let i;
  for (i = 0; i < globes.length; i++) {
    globeLightsOn[i] = false;
    globes[i].style.fill = globeOffColor[i];
  }
}
```

2. apelați funcția definită anterior pentru a începe aplicația cu luminițele stinse.

```
turnOffLights();
```

3. Creați primul joc de lumini astfel încât atunci când apăsați primul buton toate luminițele să se aprindă și să se stingă la un interval de timp definit de voi (e.g. 200 ms). Toate butoanele trebuie să implementeze stingerea jocului de lumini deja activ pentru a nu se influența două jocuri de lumini între ele.

1. adăugați un eveniment pentru primul buton care să execute o funcție **blinkLights()** în care veți implementa jocul de lumini.

```
// We will recycle the same event handler
let intervalHandler = -1;


// First button event
blinkLightsButton.addEventListener("click", (evt) => {
  if (intervalHandler == -1) {
```

```
    intervalHandler = setInterval(blinkLights, 200);
  } else { // Turn off event and cleanup
    clearInterval(intervalHandler);
    intervalHandler = -1;
    turnOffLights();
  }
});
```

2. implementați funcția **blinkLights()** care a fost dată ca parametru la subpunctul anterior (toate luminițele să se aprindă dacă erau stinse, și să se stingă dacă erau aprinse).

```
function blinkLights() {
  let i;
  for (i = 0; i < globes.length; i++) {
      if (globeLightsOn[i]) {
        globes[i].style.fill = globeOffColor[i];
      } else {
        globes[i].style.fill = globeOnColor[i];
      }
      globeLightsOn[i] = !globeLightsOn[i];
  }
}
```

Eplicații:

- Pentru a se adăuga un eveniment la apăsarea unui buton s-a utilizat funcția **addEventListener()** pe elementul buton în care s-a setat un click event [https://dev.fitbit.com/build/guides/user-interface/svg-components/buttons/#click-events].
- Pentru a apela o funcție la un anumit interval de se utilizeaza funcția **setInterval()** care primește ca parametri o funcție de apelat și un interval de milisecunde în care să apeleze funcția dată, și întoarce un număr ce reprezintă ID-ul timer-ului setat. Pentru a dezactiva timer-ul se utilizează funcția **clearInterval()** care primește ca parametru ID-ul timer-ului. Pentru documentație intrați aici [https://dev.fitbit.com/build/reference/companion-api/globals/].
- Variabila **intervalHandler** are rolul de a reține ID-ul ultimului timer setat pentru a-l putea deseta la următoarea apăsare de buton.

4. Creați al doilea joc de lumini astfel încât atunci când apăsați al doilea buton să se aprindă o singură luminiță, restul fiind stinse. La un interval de timp definit de voi (e.g. 200 ms) să se aprindă următoarea luminiță, iar anterioara să se stingă. Toate butoanele trebuie să implementeze stingerea jocului de lumini deja activ pentru a nu se influența două jocuri de lumini între ele.

1. adăugați un eveniment pentru al doilea buton care să execute o funcție **slideLights()** în care veți implementa jocul de lumini.

```
let currentGlobeSlide; // keep track of the current globe for slide lights show
slideLightsButton.addEventListener("click", (evt) => {
  if (intervalHandler == -1) {
    currentGlobeSlide = 0;
    globes[currentGlobeSlide].style.fill = globeOnColor[currentGlobeSlide];
    globeLightsOn[currentGlobeSlide] = true;
```

```
    intervalHandler = setInterval(slideLights, 200);
  } else { // Turn off event and cleanup
    clearInterval(intervalHandler);
    intervalHandler = -1;
    turnOffLights();
  }
});
```

2. implementați funcția **slideLights()** care a fost dată ca parametru la subpunctul anterior (să se aprindă următoarea luminiță, iar anterioara să se stingă).

```
function slideLights() {
    globes[currentGlobeSlide].style.fill = globeOffColor[currentGlobeSlide];
    globeLightsOn[currentGlobeSlide] = false;

    currentGlobeSlide = (currentGlobeSlide + 1) % globes.length;
    globes[currentGlobeSlide].style.fill = globeOnColor[currentGlobeSlide];
    globeLightsOn[currentGlobeSlide] = true;
}
```

Eplicații:

- Variabila **currentGlobeSlide** are rolul de a reține luminița globului ce a fost aprinsă pentru a putea fi stinsă după ce a trecut intervalul de timp.

[Bonus] 5. Creați al treilea joc de lumini după cum doriți voi. Puteți urma ca exemplu celălalte jocuri de lumini. Toate butoanele trebuie să implementeze stingerea jocului de lumini deja activ pentru a nu se influența două jocuri de lumini între ele.

1. adăugați un eveniment pentru al treilea buton care să execute o funcție **customLights()** în care veți implementa jocul de lumini.

```
customLightsButton.addEventListener("click", (evt) => {
   if (intervalHandler == -1) {
    intervalHandler = setInterval(customLights, 1000); // adjust the interval timer
  } else { // Turn off event and cleanup
    clearInterval(intervalHandler);
    intervalHandler = -1;
    turnOffLights();
  }
});
```

2. implementați funcția **customLights()** care a fost dată ca parametru la subpunctul anterior (voi definiți acest joc de lumini).

```
function customLights() {
  // TODO complete here
  console.log("custom lights show");
}
```

Resurse

- Template Laborator [https://drive.google.com/file/d/18Yg3cV50ZhMa9pV4YM9Y5tmJ2Nlw37D8/view?usp=sharing]
- Getting Started with Fitbit SDK [https://dev.fitbit.com/getting-started/]
- Fitbit SDK - Device API Reference [https://dev.fitbit.com/build/reference/device-api/]
- Fitbit SDK Guides [https://dev.fitbit.com/build/guides/]
- Button Components Guide [https://dev.fitbit.com/build/guides/user-interface/svg-components/buttons/]

si/laboratoare/08.txt · Last modified: 2020/12/12 16:24 by stefan_radu.maftei