**Exercise: Asynchronous Programming**

Problems for exercises and homework for the ["JS Applications" course @ SoftUni](https://softuni.bg/courses/js-applications)

**Working with Remote Data**

For the solution of some of the following tasks, you will need to use an up-to-date version of the **local REST service**, provided in the lesson’s resources archive. You can [read the documentation here](https://github.com/softuni-practice-server/softuni-practice-server).

## Bus Schedule

Write a JS program that tracks the progress of a bus on it’s route and announces it inside an info box. The program should display which is the upcoming stop and once the bus arrives, to request from the server the name of the next one. Use the skeleton from the provided resources.

The bus has two states – **moving** and **stopped**. When it is **stopped**, only the button “**Depart**” is **enabled**, while the info box shows the name of the **current** stop. When it is **moving**, only the button “**Arrive**” is **enabled**, while the info box shows the name of the **upcoming** stop. Initially, the info box shows "**Not Connected**" and the "**Arrive**" button is **disabled**. The ID of the first stop is "**depot**".

When the "**Depart**" button is clicked, make a GET request to the server with the ID of the current stop to address http://localhost:3030/jsonstore/bus/schedule/:id (replace the highlighted part with the relevant value). As a response, you will receive a JSON object in the following format:

stopId {

name: stopName,

next: nextStopId

}

Update the info box with the information from the response, disable the “Depart” button and enable the “Arrive” button. The info box text should look like this (replace the highlighted part with the relevant value):

Next stop {stopName}

When the "**Arrive**" button is clicked, update the text, disable the “Arrive” button and enable the “Depart” button. The info box text should look like this (replace the highlighted part with the relevant value):

Arriving at {stopName}

Clicking the buttons successfully will cycle through the entire schedule. If invalid data is received, show "**Error**" inside the info box and **disable** both buttons.

### Examples

Initially, the info box shows “Not Connected” and the arrive button is disabled.





When Depart is clicked, a request is made with the first ID. The info box is updated with the new information and the buttons are changed:





Clicking Arrive, changes the info box and swaps the buttons. This allows Depart to be clicked again, which makes a new request and updates the information:



