Nemobot Programming Guide

Documentation Page (MUST READ!)

https://algebragame.github.io/GamePlatform/

TWO Different Platforms

- https://gamebot.algebragame.app
  - Submitted games can be accessed by texting Nemobot "[project name] [username]"
  - You don't need to be granted special access and you can start developing right away
  - Decent games will be promoted to the list of student projects
- http://game.algebragamification.com/bot
  - Submitted games can be accessed from the Nemobot menu (Games -> Student Projects)
  - Newly created accounts will need to be granted access before accessing the platform
  - Good games will be promoted to be official games (Games -> Official Games)

Note to PU student: To avoid cluttering the "Student Projects" menu, please use the first platform (https://gamebot.algebragame.app) to develop your games.

Sample Project

The source code of the “Fish-Flavored Lollipops” game can be found here.

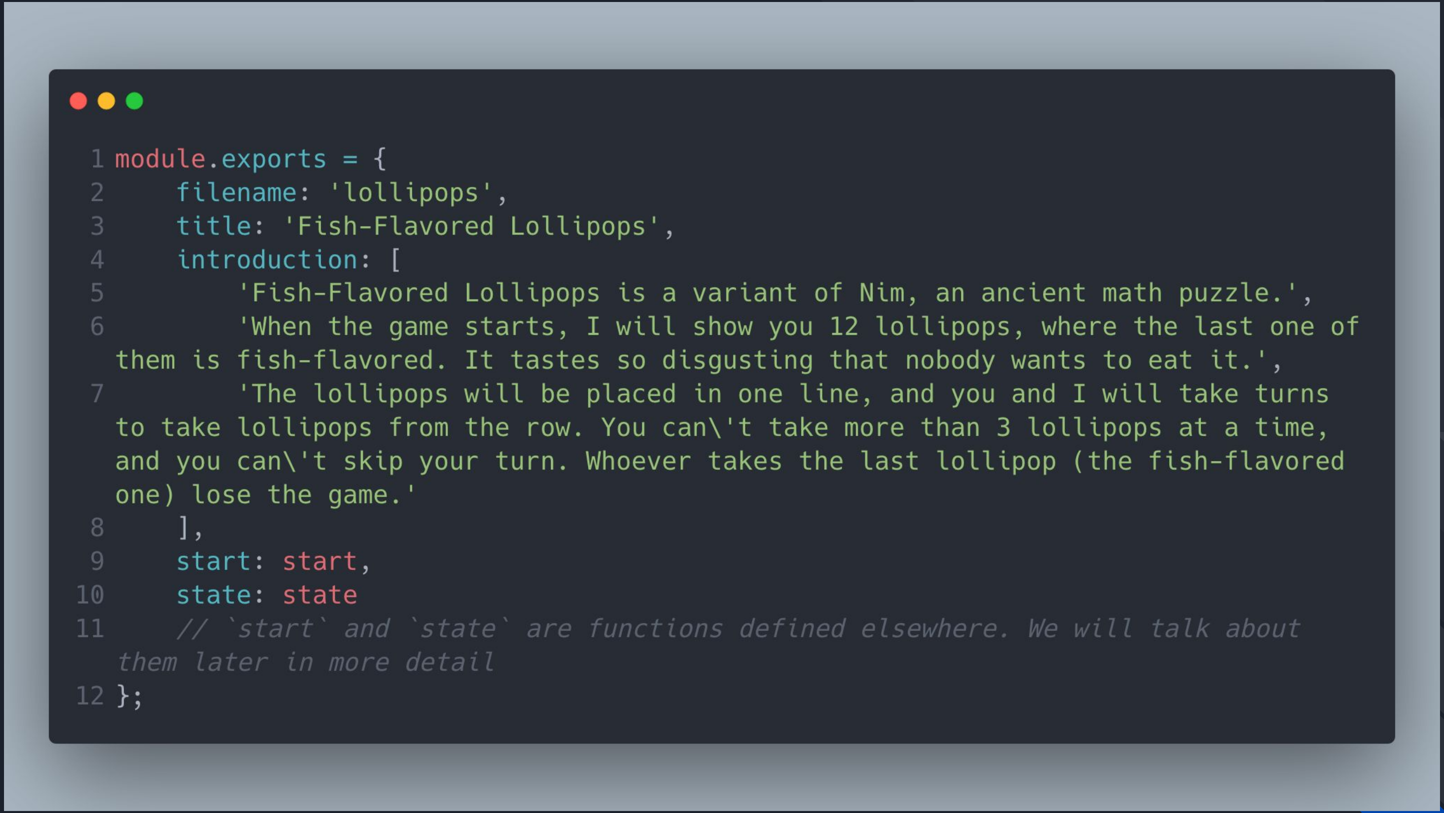
The game is one of the official games, and it’s written in less than 80 lines of code.

Using `module.exports`

A script must export the following fields using the `module.exports` syntax:

- `filename` (string)
- `title` (string)
- `introduction` (string / array of strings)
- `start` (function)
- `state` (function)

These fields are explained in details in the documentation.



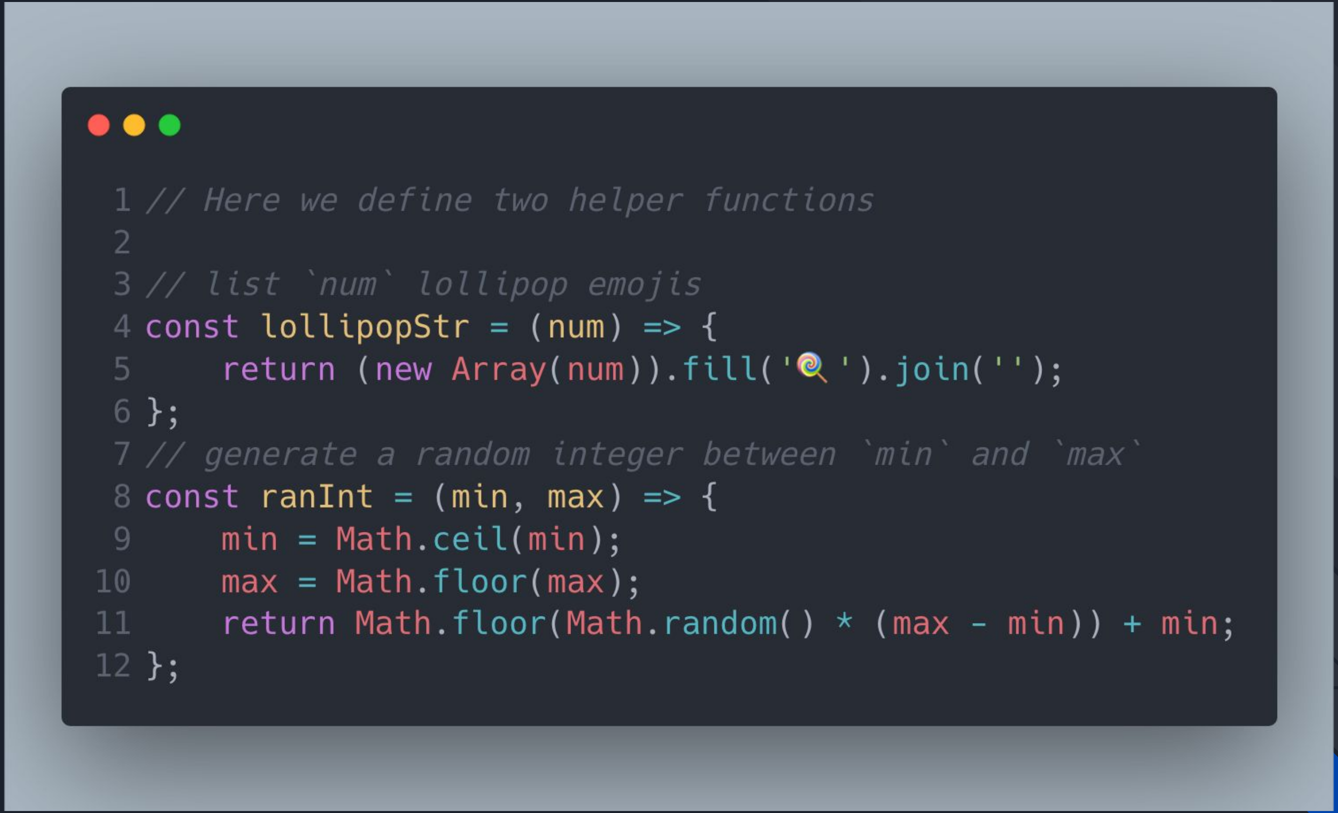
```javascript
module.exports = {
    filename: 'lollipops',
    title: 'Fish-Flavored Lollipops',
    introduction: [
        'Fish-Flavored Lollipops is a variant of Nim, an ancient math puzzle.',
        'When the game starts, I will show you 12 lollipops, where the last one of them is fish-flavored. It tastes so disgusting that nobody wants to eat it.',
        'The lollipops will be placed in one line, and you and I will take turns to take lollipops from the row. You can\'t take more than 3 lollipops at a time, and you can\'t skip your turn. Whoever takes the last lollipop (the fish-flavored one) lose the game.'
    ],
    start: start,
    state: state
    // `start` and `state` are functions defined elsewhere. We will talk about them later in more detail
};
```

`start` and `state`

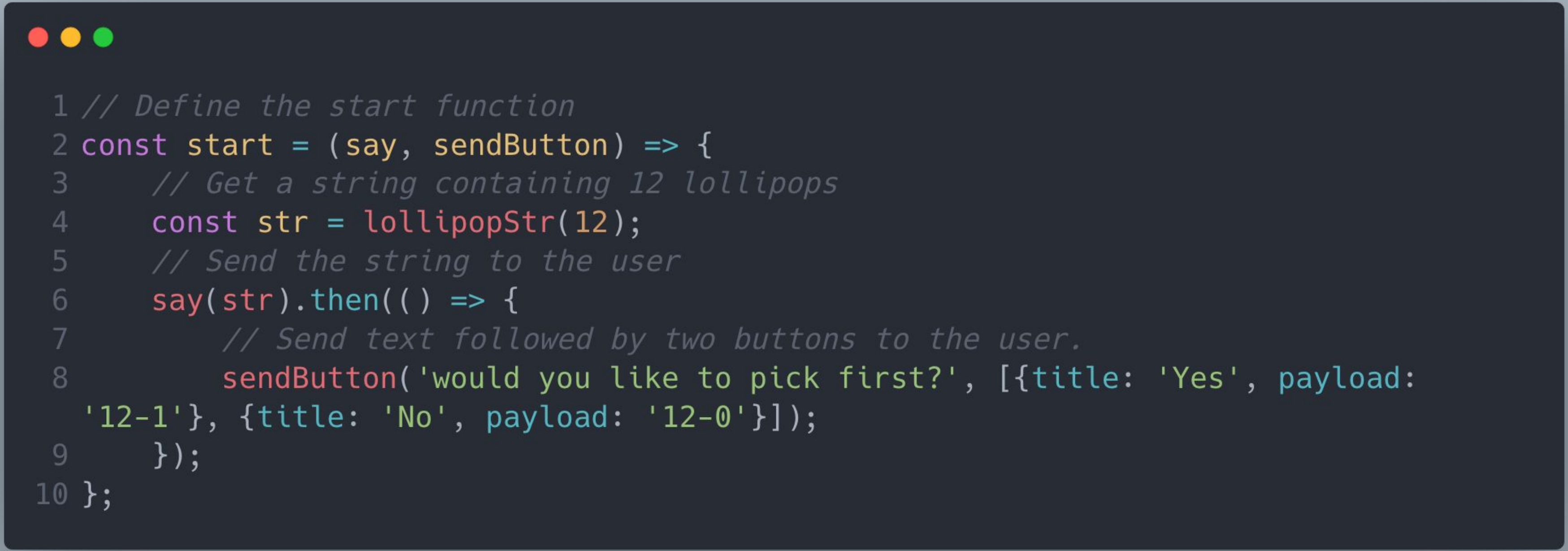
`start` and `state` are two functions that you have to define in your script and export to Nemobot.

- `start` should take three parameters:
    - `say` : a predefined function that you can use to send something to the user. You can send text, images and video/audio files. Please check the documentation for details.
    - `sendButton`: a predefined function that you can use to send multiple buttons to the user
    - `userID`: a string containing the ID of the user
- `state` should take four parameters:
    - `payload`: a string that's used to define the state
    - `say`, `sendButton` and `userID`: same as above

Let’s Go Through the Sample Code



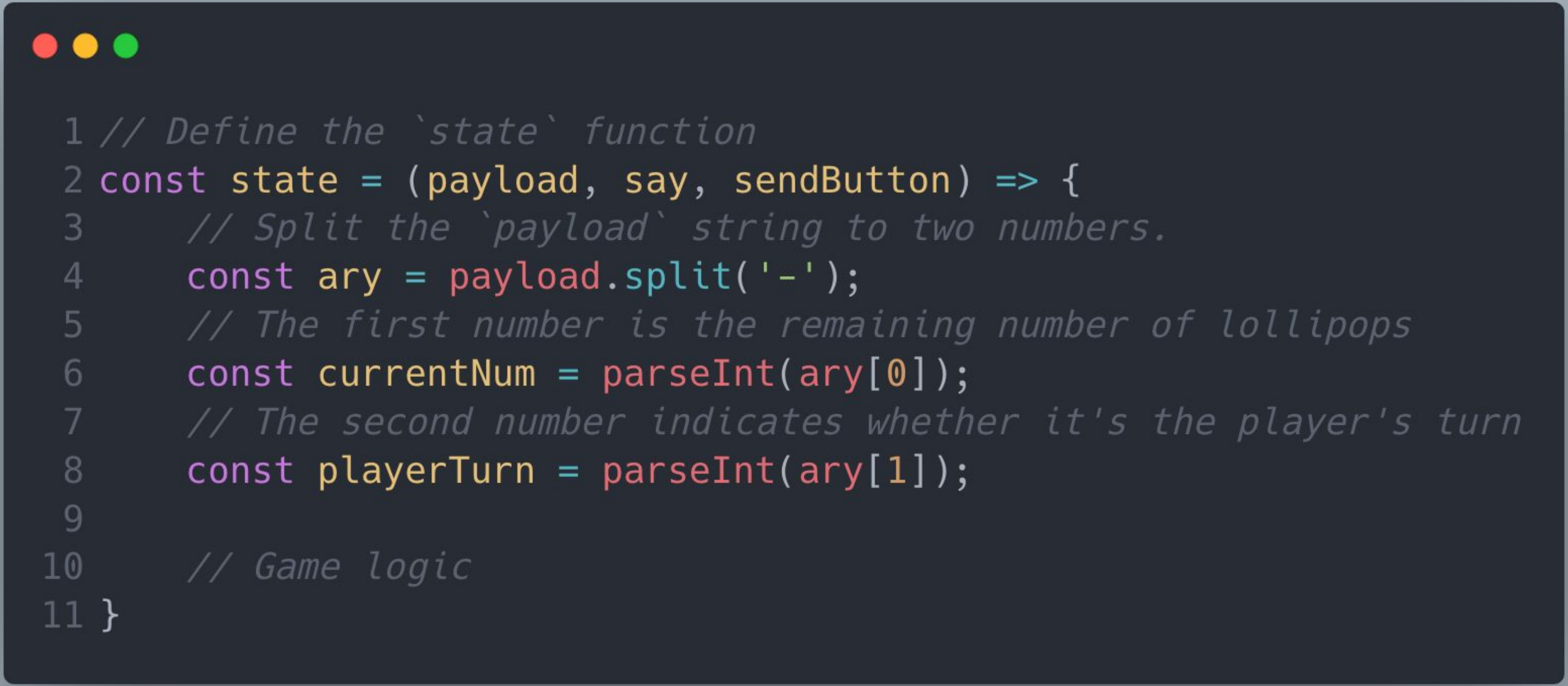
```javascript
// Here we define two helper functions

// list `num` lollipop emojis
const lollipopStr = (num) => {
    return (new Array(num)).fill('🍭').join('');
};
// generate a random integer between `min` and `max`
const ranInt = (min, max) => {
    min = Math.ceil(min);
    max = Math.floor(max);
    return Math.floor(Math.random() * (max - min)) + min;
};
```



```javascript
// Define the start function
const start = (say, sendButton) => {
    // Get a string containing 12 lollipops
    const str = lollipopStr(12);
    // Send the string to the user
    say(str).then(() => {
        // Send text followed by two buttons to the user.
        sendButton('would you like to pick first?', [{title: 'Yes', payload: '12-1'}, {title: 'No', payload: '12-0'}]);
    });
};
```

Note: When the user press on a button, Nemobot will call the `state` function and pass in the corresponding payload as a parameter.

E.g., When the "Yes" button is pressed, Nemobot will call the `state` function and pass in the payload `12-1`.



```javascript
// Define the `state` function
const state = (payload, say, sendButton) => {
    // Split the `payload` string to two numbers.
    const ary = payload.split('-');
    // The first number is the remaining number of lollipops
    const currentNum = parseInt(ary[0]);
    // The second number indicates whether it's the player's turn
    const playerTurn = parseInt(ary[1]);

    // Game logic
}
```