动手实验 – Promise 回调 (10 分钟)

实验目标：

完成本实验后，您将能够：

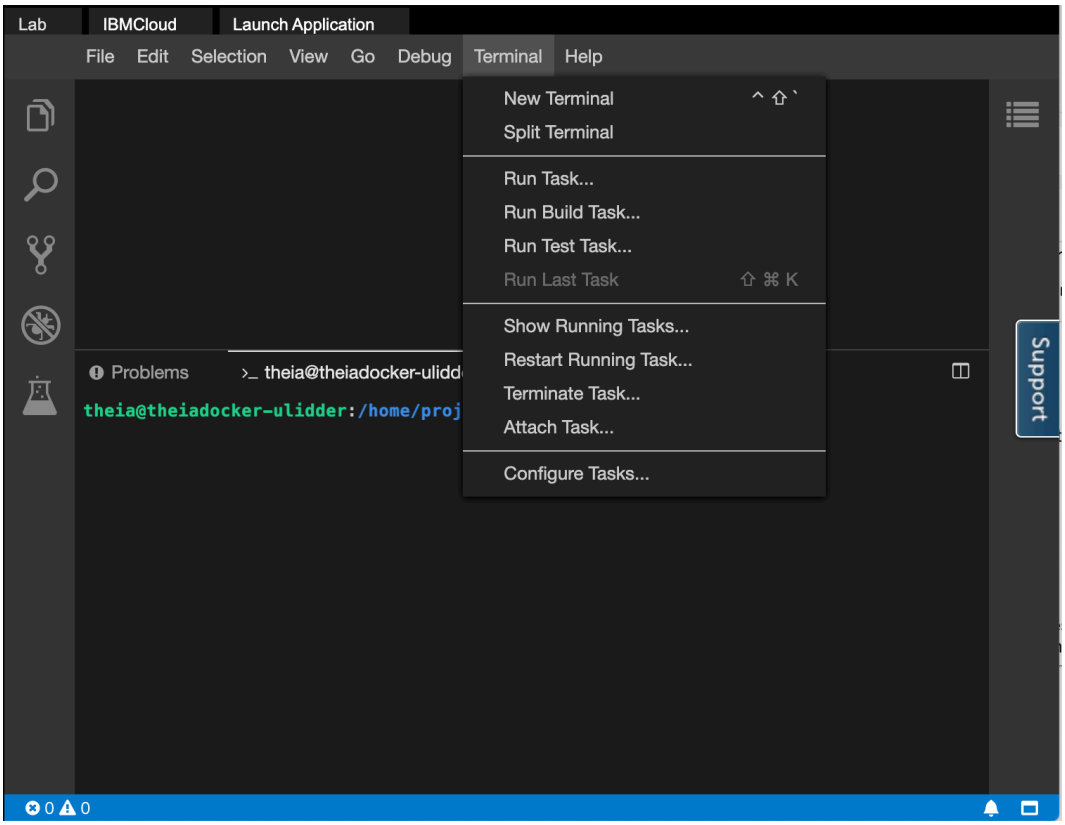
- 描述 Promise 回调
- 创建一个使用 Promise 的 Node.js 应用程序

先决条件

- 基础的 JavaScript 知识

任务 1：创建并执行一个带有 Promise 的脚本

1. 通过编辑器菜单打开终端窗口：终端 > 新终端。



2. 切换到你的项目文件夹。

```
cd /home/project
```

3. 通过运行以下命令创建一个新文件。

```
touch promisescript.js
```

5. 列出当前目录中的文件，以查看文件是否已创建。

```
ls
```

6. 打开文件 **promisescript.js** 并进行编辑。

Open **promisescript.js** in IDE

7. 将以下代码粘贴到文件中并保存。你能查看文件并猜测控制台日志执行的顺序吗?

```
//Creating a promise method. The promise will get resolved when timer times out after 6 seconds.
let myPromise = new Promise((resolve,reject) => {
    setTimeout(() => {
      resolve("Promise resolved")
    },6000)})
//Console log before calling the promise
console.log("Before calling promise");
//Call the promise and wait for it to be resolved and then print a message.
myPromise.then((successMessage) => {
    console.log("From Callback " + successMessage)
  })
//Console log after calling the promise
  console.log("After calling promise");
```

- 上面的代码创建了一个承诺 (`myPromise`)，在 6 秒后以消息 `Promise resolved` 解决。
- 脚本首先在控制台记录 `Before calling promise`。
- 然后，它设置 `then` 方法来处理承诺的解决，但回调尚未执行。
- 紧接着，脚本在控制台记录 `After calling promise`。
- 6 秒后，承诺解决，回调在控制台记录 `From Callback Promise resolved`。

8. 前往终端并运行文件以验证你的猜测。

```
node promisescript.js
```

9. 控制台日志执行的顺序告诉你，承诺调用后面的语句是一个接一个地执行，而承诺同时被解决或拒绝。

练习：创建并执行一个包含两个 Promise 的脚本

1. 创建一个脚本，其中有两个方法返回 Promise – 一个 Promise 应在 6 秒超时后解决，另一个应在 3 秒超时后解决。以这样的方式调用 Promise，使得第二个 Promise 在第一个 Promise 解决后被调用。

▼ 点击这里查看代码

```
let myPromise1 = new Promise((resolve,reject) => {
    setTimeout(() => {
      resolve("Promise 1 resolved")
    },6000)})
let myPromise2 = new Promise((resolve,reject) => {
    setTimeout(() => {
      resolve("Promise 2 resolved")
    },3000)})
  myPromise1.then((successMessage) => {
    console.log("From Callback " + successMessage)
    myPromise2.then((successMessage) => {
      console.log("From Callback " + successMessage)
    })
  })
```

```
theia@theia-ksundararaja:/home/project$ node promisescript.js
From Callback Promise 1 resolved
From Callback Promise 2 resolved
```

2. 修改代码以顺序调用承诺，并查看输出如何变化。

▼ 点击这里查看代码

```
let myPromise1 = new Promise((resolve,reject) => {
    setTimeout(() => {
      resolve("Promise 1 resolved")
    },6000)})
let myPromise2 = new Promise((resolve,reject) => {
    setTimeout(() => {
```

```
        resolve("Promise 2 resolved")
      },3000)})
    myPromise1.then((successMessage) => {
      console.log("From Callback " + successMessage)
    })
    myPromise2.then((successMessage) => {
    console.log("From Callback " + successMessage)
  })
```

```
theia@theia-ksundararaja:/home/project$ node promisescript.js
From Callback Promise 2 resolved
From Callback Promise 1 resolved
```

恭喜！您已完成 Promise-Callback 实验。

概述

在本实验中，您学习了如何以两种方式使用带回调的 Promise：

- 如何在第一个 Promise 解析后调用第二个 Promise。
- 如何顺序调用 Promise。

接下来，您将学习如何使用这些方法改善服务器端功能。

Author(s)

Lavanya

© IBM Corporation. All rights reserved.