使用Nodejs激活GitHub Copilot

用于运行实验以评估Copilot可行性的演示项目

说明

- 将练习文件夹下载到本地
- 打开NodeServer.js，并开始编写一个Nodejs服务器，根据初始文本检查第一个建议
- 打开test.js文件并分析当前测试
- 打开命令提示符并运行测试 (运行 `mocha test.js`)
- 查看结果，应该显示如下内容：

```
mocha test.js
server is listening on port 3000

  Node Server

    √ should return "key not passed" if key is not passed

  1 passing (34ms)
```

- 在NodeServer.js中实现练习中描述的其余方法（不要忘记在Visual Studio Code中打开color.json文件，以便Copilot获得更好的推荐上下文）
- 在Test.js文件中添加测试功能的方法
- 运行测试以验证一切正常
- 打开dockerfile文件，并填写它，以便使用Node镜像创建一个能够运行Web服务器的Docker容器
- 创建命令以在端口4000上运行docker
- 测试应用程序是否在端口4000上运行

- 在**nodeserver.js**文件中，你可以像这样输入一行新的注释 //运行curl命令来测试服务器

这样我们就可以看到CoPilot根据当前文件生成了一个curl命令，可以在命令行中执行

- 也可以更具体一点，比如：//运行curl命令测试daysBetweenDates方法

这样它会为特定方法生成一个测试

练习

练习包括使用Nodejs构建一个Web服务器，以处理各种功能的请求。

服务器必须处理的请求包括：

- **/Get** :

返回一个hello world消息

- **/DaysBetweenDates**:

计算两个日期之间的天数

通过查询字符串接收2个参数date1和date2，并计算这两个日期之间的天数。

- **/Validatephonenumber**:

通过查询字符串接收名为phoneNumber的参数 用西班牙格式验证phoneNumber，例如+34666777888 如果phoneNumber有效则返回"valid" 如果phoneNumber无效则返回"invalid"

- **/ValidateSpanishDNI**:

通过查询字符串接收名为dni的参数 计算DNI字母 如果DNI有效则返回"valid" 如果DNI无效则返回"invalid"

我们将创建自动化测试来检查功能是否正确实现。开发完成后，我们将使用Docker构建一个容器

- **/ReturnColorCode**:

通过查询字符串接收名为color的参数

读取colors.json文件并返回rgba字段

从查询字符串获取color变量

遍历colors.json中的每种颜色以找到该颜色

返回code.hex字段

- **/TellMeAJoke**:

调用笑话API并使用axios返回一个随机笑话

- **/MoviesByDirector**:

（这将需要浏览https://www.omdbapi.com/apikey.aspx并请求一个免费的API密钥）

通过查询字符串接收名为director的参数

调用电影API并使用axios返回该导演的电影列表

返回完整的电影列表

- **/ParseUrl**:

从查询字符串中检索名为someurl的参数

解析URL并返回协议、主机、端口、路径、查询字符串和哈希

返回解析的主机

- **/ListFiles**:

获取当前目录

获取当前目录中的文件列表

返回文件列表

- **/GetFullTextFile**:

读取sample.txt并返回包含单词"Fusce"的行

(注意此实现，因为通常在分析之前会读取文件的全部内容，因此内存使用高，当文件太大时可能会失败)

- **/GetLineByLinefromtTextFile**:

逐行读取sample.txt

创建一个按行读取文件的承诺，并返回包含单词"Fusce"的行列表

返回行的列表

- **/CalculateMemoryConsumption**:

返回进程的内存消耗，以2位小数GB为单位

- **/MakeZipFile**:

使用zlib创建一个名为sample.gz的zip文件，其中包含sample.txt

- **/RandomEuropeanCountry**:

创建一个包含欧洲国家及其ISO代码的数组

从数组中返回一个随机国家

返回国家及其ISO代码

GitHub Copilot实验练习

可以使用Copilot实验室插件执行以下任务，当前为预览功能，可能会有一些错误。

确保安装GitHub Copilot实验室扩展：https://marketplace.visualstudio.com/items?itemName=GitHub.copilot-labs

打开GitHub Copilot扩展以查看所有可用的功能。

- **Explain**

选择validatePhoneNumber方法中具有正则表达式的行，在“解释”部分点击“向Copilot询问”。你将看到一个详细解释正则表达式中不同符号的含义。

- **语言翻译**

选择一些源代码，比如这行：

```
var randomCountry = countries[Math.floor(Math.random() * countries.length)];
```

在“语言翻译”部分选择Python并单击“向Copilot询问”按钮，你应该看到新的Python代码。

- 可读性

选择MakeZipFile的内容

在“可读性”部分，点击“可读性”，查看如何添加注释以及将具有简短名称的变量重命名为更易理解的名称。

-- 添加类型

待定

-- 修复错误

在练习中，不应该有错误，因为大部分代码将由CoPilot完成。我们可以强制一些错误，然后测试调试功能。

制造一些错误，比如：

在for循环中将开头更改为（将0更改为1）：

```
for (var i = 1
```

选择文本，在“刷子”部分单击“修复错误”按钮。

-- 调试

选择包含变量的一些文本行，比如：

```
var queryData = url.parse(req.url, true).query;
var color = queryData.color;
var colorFound = "not found";
```

选择文本，在“刷子”部分点击“调试”按钮。

-- 整理

待定

-- 列出步骤

选择没有注释的一些代码行，在“刷子”部分点击“列出步骤”按钮。

-- 使健壮

选择来自输入的一些文本，例如来自查询字符串的变量：

```
    var queryData = url.parse(req.url, true).query;
    var date1 = queryData.date1;
    var date2 = queryData.date2;
```

在“刷子”部分点击“使健壮”按钮，你会看到添加了额外的验证。

-- 分块

待定

-- 文档

选择一行（例如一个方法或if子句的开始）

```
else if (req.url.startsWith('/GetFullTextFile'))
```

在“刷子”部分点击“文档”按钮，你会看到在这一行之前添加了解释代码功能的注释。

-- 生成测试

待定