动手实验：使用 Chrome DevTools 排查常见错误

预计所需时间： 45 分钟

在这个实验中，您将通过直接在元素、控制台和源面板中工作来学习如何使用 Chrome DevTools。

正如您所知道的：

> Chrome DevTools 是一组直接内置于 Google Chrome 浏览器中的网页开发工具。DevTools 允许您动态编辑页面并快速诊断问题，从而更快地构建更好的网站。

目标：

完成此实验后，您将能够：

- 查找、编辑和调试节点
- 使用控制台调试 JavaScript
- 在 JavaScript 代码中使用断点

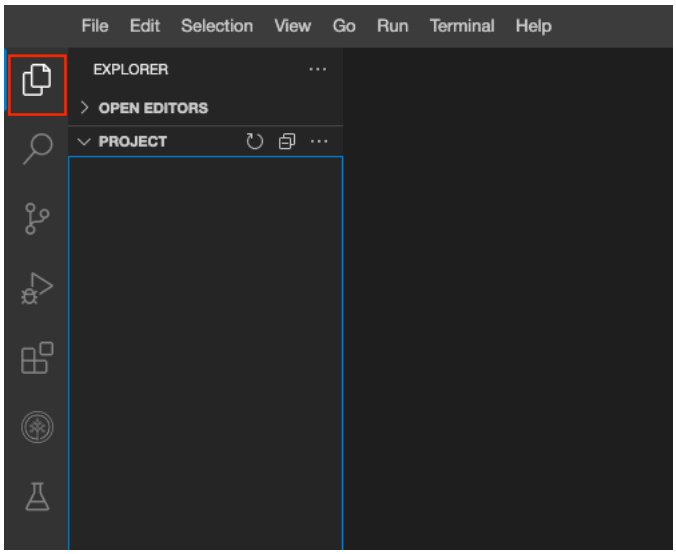
先决条件：

- 本实验要求您在 Chrome 网络浏览器中调试应用程序。但您也可以使用其他浏览器，尽管您会看到类似的功能，但菜单选项和标签名称会有所不同。

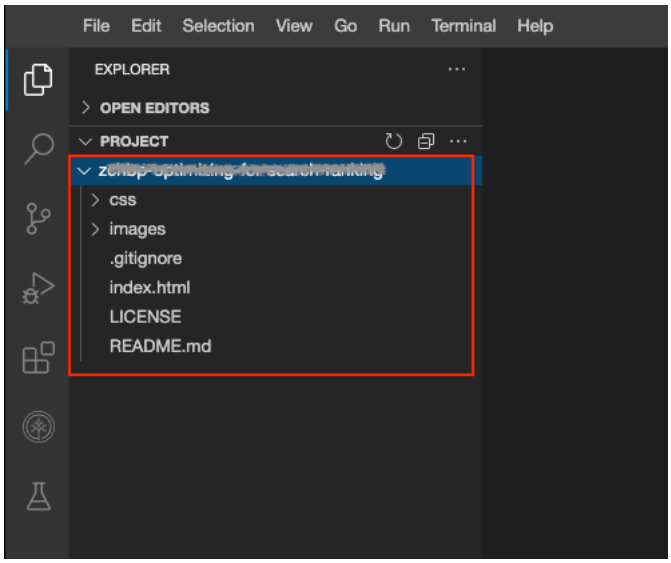
在云IDE中处理文件

如果您是云IDE的新手，本节将向您展示如何在云IDE中创建和编辑项目中的文件。

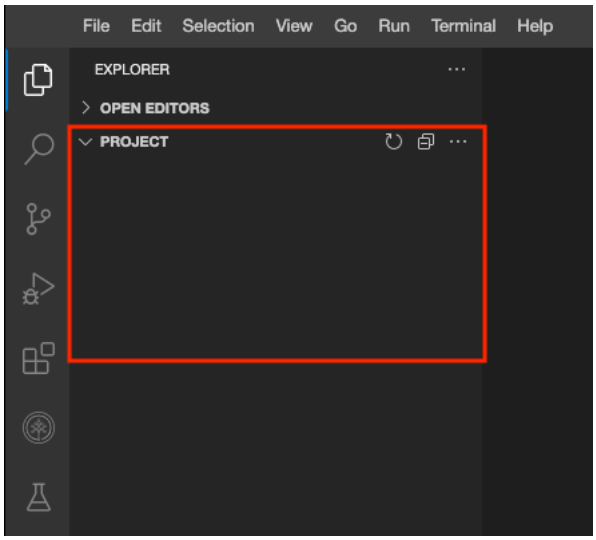
要查看云IDE中的文件和目录，请单击此文件图标以显示它。



如果您已克隆（使用 `git clone` 命令）样板/起始代码，则它看起来如下：

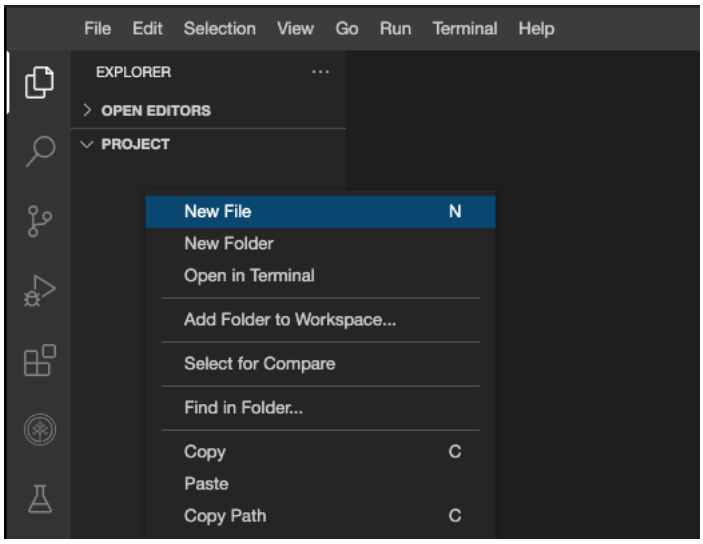


否则，一个空项目看起来是这样的：



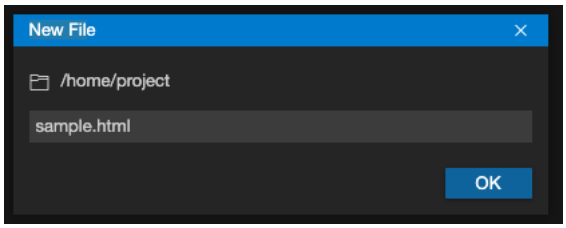
创建新文件

您可以右键单击并选择“新建文件”选项以在项目中创建文件。

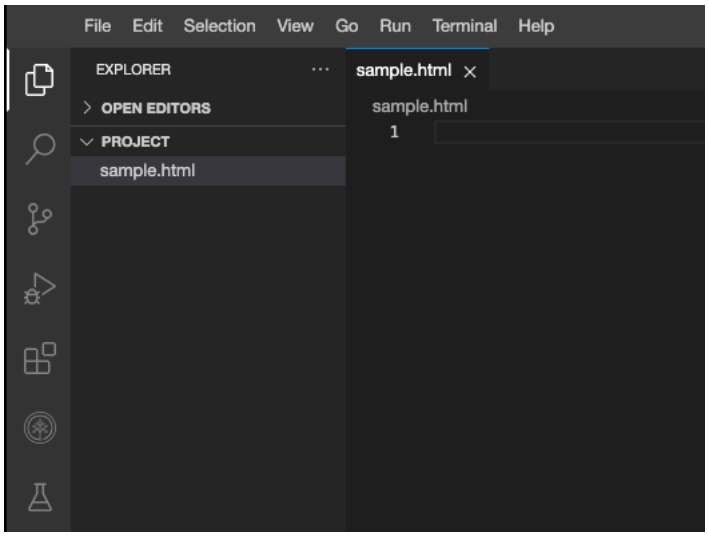


您也可以选择 文件 -> 新建文件 来执行相同操作。

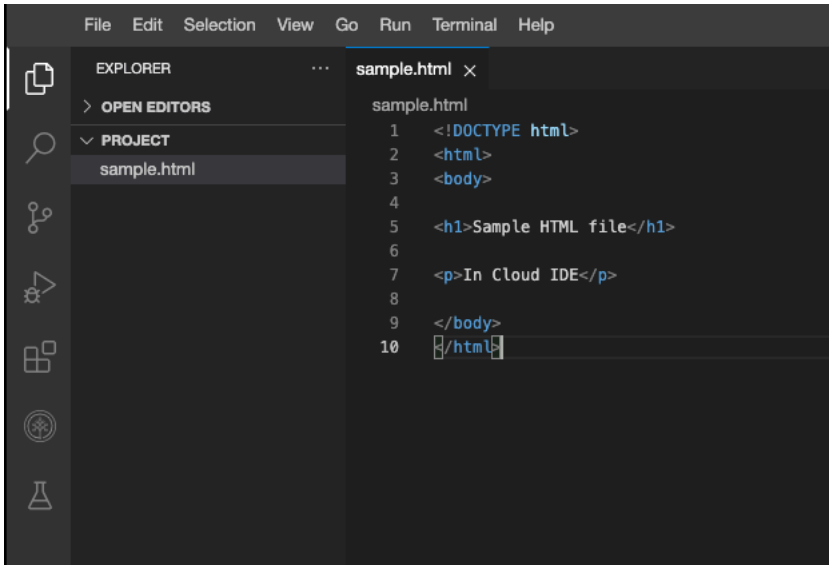
然后会提示您输入新文件的名称。在下面的示例中，我们创建了 `sample.html`。



单击目录结构中的文件名 `sample.html` 将在右侧窗格中打开该文件。您可以创建所有不同类型的文件；例如，`FILE_NAME.js` 用于JavaScript文件。

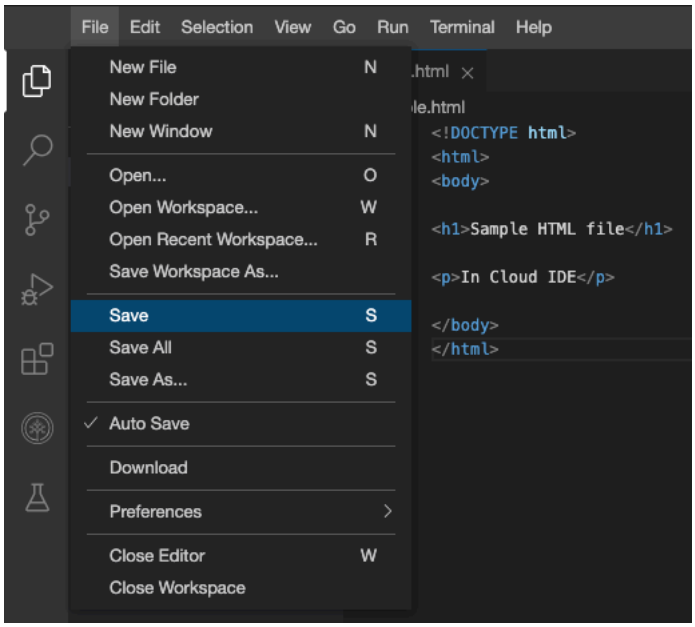


在示例中，我们只是粘贴了一些基本的html代码，然后保存了文件。



并通过以下方式保存：

- 进入菜单。
- 在Mac上按 `⌘ Cmd + S` 或在Windows上按 `CTRL + S`。
- 或者，它也可以为您自动保存。



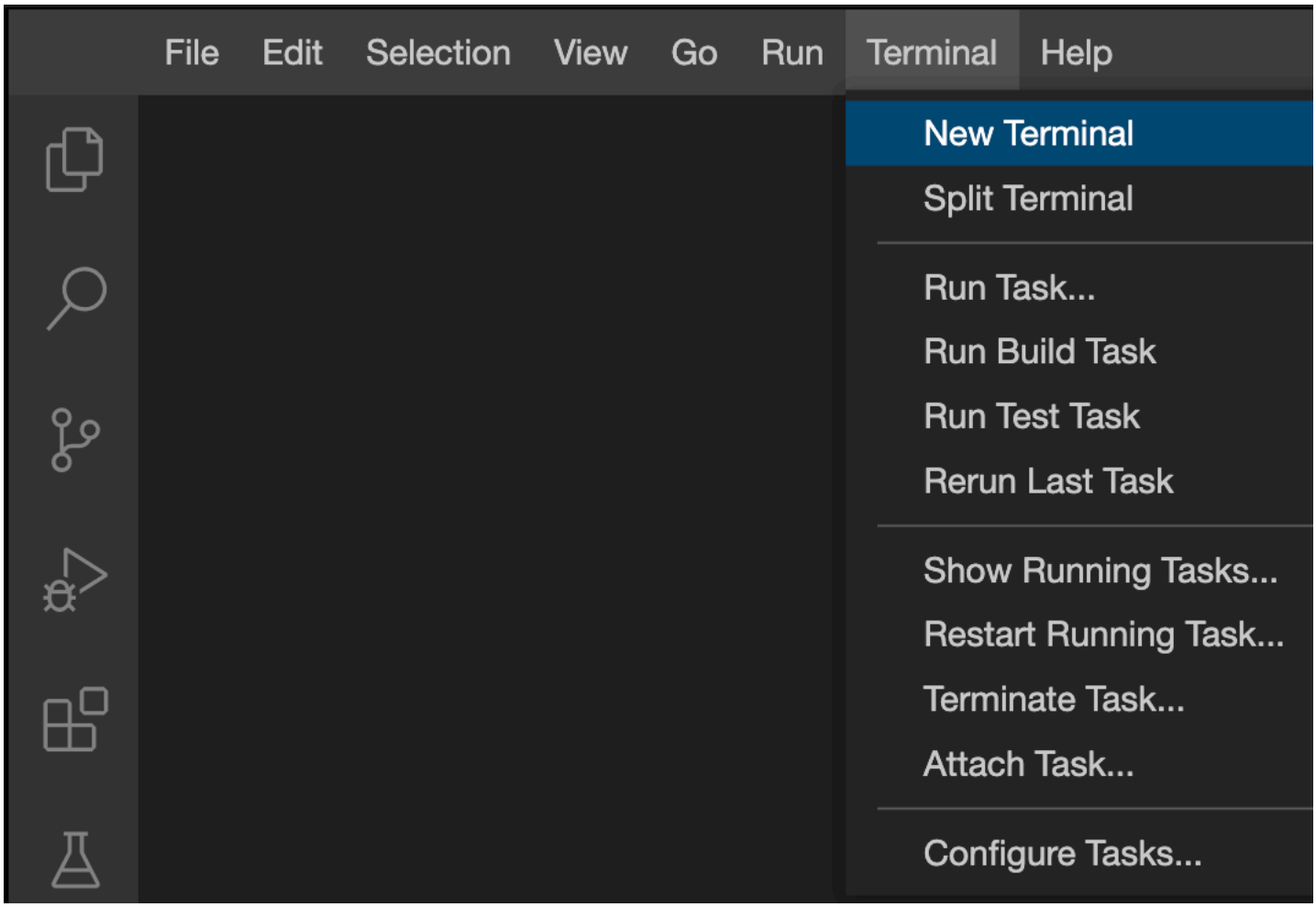
设置：创建 HTML 页面

在资源管理器的项目部分创建一个名为 `index.html` 的文件，并将以下代码粘贴到其中。这将帮助您完成实验。

```
<!DOCTYPE html>
<html>
  <head>
    <title>Working with Chrome DevTools</title>
  </head>
  <body>
    <h1>Welcome to Chrome DevTools Lab</h1>
    <ul>
      <li>Michelangelo</li>
      <li>Raphael</li>
    </ul>
    <p style="white-space: pre">
      "To be, or not to be: that is the question:
Whether 'tis nobler in the mind to suffer
The slings and arrows of outrageous fortune,
Or to take arms against a sea of troubles,
And by opposing end them? To die: to sleep;
No more; and by a sleep to say we end
The heart-ache and the thousand natural shocks
That flesh is heir to, 'tis a consummation
Devoutly to be wish'd. To die, to sleep;
To sleep: perchance to dream: ay, there's the rub;
For in that sleep of death what dreams may come
When we have shuffled off this mortal coil,
Must give us pause: there's the respect
That makes calamity of so long life;" - Hamlet, Act III, Scene I
    </p>
    <ul>
      <li>Eggs</li>
      <li>Milk</li>
      <li>Bread</li>
      <li>Cereal</li>
    </ul>
  </body>
</html>
```

测试 HTML 页面

1. 通过编辑器中的菜单打开终端窗口：**终端 > 新终端**。



2. 在终端上运行以下命令以托管您的网页。

```
python3 -m http.server
```

Launch Application

3. 您可以复制该网址并粘贴到您的 Chrome 浏览器中。

练习 1：打开 Chrome 开发者工具

在本实验中，我们将查看 Chrome 开发者工具中最常见的面板。

但首先，由于不同用户需要快速访问开发者工具 UI 的不同部分，因此有多种方法可以启动开发者工具。

- 要访问 DOM 或 CSS，请右键单击页面上的元素并选择 **检查** 以打开元素面板。您还可以使用 `Command+Option+C (Mac)` 或 `Control+Shift+C (Windows, Linux, ChromeOS)`。
- 要访问控制台面板并查看日志消息或运行 JavaScript，请按 `Command+Option+J (Mac)` 或 `Control+Shift+J (Windows, Linux, ChromeOS)`。

开发者工具面板

1. 元素面板：此面板允许开发人员检查和修改网页的 HTML 和 CSS。它以层次结构显示 DOM 树，使开发人员能够轻松定位和操作特定元素。
2. 控制台面板：控制台面板是一个交互式 JavaScript 控制台，允许开发人员运行代码并实时查看结果。它可用于调试和排除 JavaScript 错误，以及测试和实验代码。
3. 源代码面板：源代码面板允许开发人员查看和调试网页的 JavaScript 代码。它具有断点、逐步调试和性能分析等功能，可用于查找和修复代码中的错误。
4. 网络面板：网络面板显示有关网页网络请求的详细信息，例如 HTTP 头、请求和响应体以及时间。此数据可用于通过识别和解决网络瓶颈来提高 Web 应用程序的性能。
5. 性能面板：性能面板提供有关网页性能的信息，包括页面加载时间、帧率和内存使用等指标。它可用于识别性能瓶颈并优化 Web 应用程序性能。

练习 2：检查一个节点

让我们检查一下我们 HTML 页面中的一个节点，看看它在 DOM 树中的位置。

右键点击“**Michelangelo**“ 并选择 **检查**。

DevTools 元素面板打开，DOM 树中高亮显示 `<li>Michelangelo</li>`。

在 DevTools 的左上角，点击 **检查** 图标。

然后选择“Raphael“ 文本。在 DOM 树中，`<li>Raphael</li>` 现在被高亮显示。

练习 3：搜索和编辑

我们现在可以使用 Chrome 开发者工具来搜索和编辑节点。但请记住，在这里所做的任何更改都不会被保存，也不会在服务器上更新。一切都完全在您的浏览器内进行。

搜索节点

转到元素面板，按 `Control+F (Windows)` 或 `Command+F (Mac)`。搜索栏会出现在元素标签的底部。在其中输入“not to be"。哈姆雷特的第一句话将被高亮显示。

编辑内容

右键点击“Bread"，选择元素中的 `li`，然后按下键盘上的删除键。

练习 4：使用控制台

现在让我们讨论如何使用控制台。正如你所知道的：

> 控制台面板包含一个交互式的 JavaScript 控制台，允许开发人员实时执行代码并查看结果。它可以用于调试和排除 JavaScript 错误，以及测试和实验代码。

开始在 HTML 页面中添加一些 JavaScript。

在最后一个闭合的 `ul` 后创建一个脚本部分，如下所示：

```
<script type="text/javascript">console.log("Hello Console!");</script>
```

输出可以看作是在控制台打印的消息“Hello Console!”。

练习 5：查找节点并在控制台调试

让我们在控制台中记录更多信息。

首先，我们将打印 `h1` 元素的文本内容。然后我们将检查是否存在 `h2`。

在控制台中记录信息有两个原因：

- 确保代码按正确的顺序执行。
- 检查某一时刻变量的值。

```
<script type="text/javascript">
  console.log("Hello Console!");
  const h1 = document.querySelector("h1");
  console.log(h1.textContent);
  console.assert(document.querySelector("h2"), "h2 not found!");
</script>
```

`console.assert()` 方法会在断言为 false 时向控制台写入错误信息。如果断言为 true，则什么也不会发生。

练习 6：源面板（第 1 部分）

我们现在将使用源面板。

在 `script` 标签之前添加以下 HTML。

```
<div>
  <label for="num1">Number 1</label>
  <input placeholder="Number 1" id="num1" />
  <label for="num2">Number 2</label>
  <input placeholder="Number 2" id="num2" />
  <button onClick="calculateSum()">Add Number 1 and Number 2</button>
  <p id="calculatorOutput"></p>
</div>
```

接下来，在 `script` 标签的末尾添加以下 Javascript。

```
function calculateSum() {
  var inputs = document.querySelectorAll("input");
  var addend1 = inputs[0].value;
  var addend2 = inputs[1].value;
  var sum = addend1 + addend2;
  var label = document.getElementById("calculatorOutput");
  label.textContent = addend1 + " + " + addend2 + " = " + sum;
}
```

在框中输入两个数字并按下按钮。我们看到错误的答案！

练习 7：源面板 第 2 部分

我们现在将使用 Chrome 开发者工具来调试和解决这个问题。

在开发者工具中，转到源面板并在行上设置断点： `var sum = addend1 + addend2;`，通过点击左侧的行号。

在页面上，再次按下按钮以将数字相加（并调用我们的函数）。

这一次，您的代码将在该行停止。将鼠标悬停在 `addend1` 和 `addend2` 上，您会注意到它们的类型为 `string` 而不是 `number`。

您还可以在源的作用域部分查看这两个变量。

让我们通过将该行更改为来修复我们的代码：

```
var sum = parseInt(addend1) + parseInt(addend2);
```

并再次调试！

恭喜！您已完成 Chrome DevTools 的实验。

摘要：

在本实验中，您使用 Chrome DevTools 检查 HTML 节点，编辑它们，调试 JavaScript，并使用控制台。

作者

Muhammad Yahya

(C) IBM Corporation. 保留所有权利。