

Curl

这是一个结合了多个 curl 示例的 bash 脚本：

```
#!/bin/bash
```

```
# 基本的 GET 请求
```

```
echo " 基本的 GET 请求:"
```

```
curl http://example.com
```

```
# 带自定义头的 GET 请求
```

```
echo "\n 带自定义头的 GET 请求:"
```

```
curl -H "User-Agent: MyUserAgent/1.0" http://example.com
```

```
# POST 请求
```

```
echo "\nPOST 请求:"
```

```
curl -X POST -d "key1=value1&key2=value2" http://example.com/post
```

```
# 带 JSON 数据的 POST 请求
```

```
echo "\n 带 JSON 数据的 POST 请求:"
```

```
curl -X POST -H "Content-Type: application/json" -d '{"key1\":"value1\","key2\":"value2\"}' http://example.com/post
```

```
# 使用代理
```

```
echo "\n 使用代理:"
```

```
curl -x http://proxy.example.com:8080 http://example.com
```

```
# 基本认证
```

```
echo "\n 基本认证:"
```

```
curl -u username:password http://example.com/auth
```

```
# 下载文件
```

```
echo "\n 下载文件:"
```

```
curl -O http://example.com/file.zip
```

```
# 跟随重定向
```

```
echo "\n 跟随重定向:"
```

```
curl -L http://example.com/redirect
```

```
# 将输出保存到文件
```

```
echo "\n 将输出保存到文件:"
```

```
curl http://example.com -o output.txt
```

```
# 自定义 HTTP 方法
```

```
echo "\n 自定义 HTTP 方法 (DELETE):"
```

```
curl -X DELETE http://example.com/delete/resource
```

```
# 详细输出
```

```
echo "\n 详细输出:"
```

```
curl -v http://example.com
```

```
# 连接超时
```

```
echo "\n 连接超时:"
```

```
curl --connect-timeout 5 http://example.com
```

```
echo "\n 脚本执行完成。"
```

将此脚本保存为 .sh 扩展名，例如 curl_examples.sh。确保在类 Unix 环境或 Windows 子系统 for Linux (WSL) 中使其可执行：

```
chmod +x curl_examples.sh
```

然后，你可以运行它：

```
./curl_examples.sh
```

Windows 注意事项： - 如果你在 Windows Cmd 或 PowerShell 中运行此脚本，你需要使用 bash 或 wsl（如果你安装了 WSL），或者确保 curl 在你的 PATH 中以便直接执行：- bash curl_examples.sh 或 wsl ./curl_examples.sh
- 请记住，此脚本使用 bash 语法，因此在 Windows 命令提示符或 PowerShell 中不原生兼容，除非通过 WSL 或 bash 环境运行。然而，curl 命令本身与 Windows 版本的 curl 兼容。