

如何用 ChatGPT 帮助程序员更高效的写出更好的代码？

原创 我是哥飞 哥飞 2022-12-08 08:30:35 广东

ChatGPT 是什么，以及如何使用 ChatGPT，本篇文章不介绍了，稍微搜索一下就能找到很多有用的信息。

今天这篇文章，我们来说，如何用 ChatGPT 帮助程序员更高效的写出更好的代码。

ChatGPT 很强大，有很多应用场景，相信这几天大家看得够多了，这里不多做介绍。

让我们进入正题，一边看例子，一边看 ChatGPT 如何帮我们编程。

十年前，哥飞做了一个在线扫描未注册域名的工具网站，名叫秋玉米，核心代码其实就是用 `socket` 连接域名的 `whois` 根服务器的 43 端口，发送域名，查询域名信息，如果有返回说明域名已被注册，返回找不到说明域名尚未被注册，或者是被保留域名。

当年很多人问哥飞，如何实现又快又稳定的查询域名是否注册，到底用的哪个接口，哥飞笑而不语。

今天，哥飞问了 ChatGPT，AI 居然很快就给出来了正确答案，而且给出的示例代码是直接能运行的。

请写一个用 php 发起 socket 连接 .com 域名 whois 根服务器的程序，用于查询 abc456.com 这个域名是否被注册了。



<?php



```
// 连接 whois 根服务器
$host = "whois.verisign-grs.com";
$port = 43;
$domain = "abc456.com";

// 创建 socket
$socket = socket_create(AF_INET, SOCK_STREAM, 0);

// 连接到 whois 根服务器
$result = socket_connect($socket, $host, $port);
if (!$result) {
    echo "连接失败";
    exit();
}

// 发送请求
socket_write($socket, $domain . "\r\n");

// 读取返回值
$response = socket_read($socket, 1024);

// 关闭 socket
socket_close($socket);

// 判断域名是否已被注册
if (strpos($response, "No match for") !== false) {
    echo "域名未被注册";
} else {
    echo "域名已被注册";
}

?>
```

你看，我的提问里，故意没提 43 端口，也没提 .com 域名的 whois 根服务器到底是哪一个，也没有告诉 AI 要怎么判断域名是否被注册了，但是 AI 给出的示例代码把这些都考虑到了。

```
1  <?php
2
3  // 连接 whois 根服务器
4  $host = "whois.verisign-grs.com";
5  $port = 43;
6  $domain = "abc456.com";
7
8  // 创建 socket
9  $socket = socket_create(AF_INET, SOCK_STREAM, 0);
10
11 // 连接到 whois 根服务器
12 $result = socket_connect($socket, $host, $port);
13 if (!$result) {
14     echo "连接失败";
15     exit();
16 }
17
18 // 发送请求
19 socket_write($socket, $domain . "\r\n");
20
21 // 读取返回值
22 $response = socket_read($socket, 1024);
23
24 // 关闭 socket
25 socket_close($socket);
26
27 var_dump($response);
28
29 // 判断域名是否已被注册
30 if (strpos($response, "No match for") !== false) {
31     echo "域名未被注册";
32 } else {
33     echo "域名已被注册";
34 }
35
36 ?>
```

我把代码复制下来，没有任何其它改动，仅仅是增加了一行 `var_dump` 语句来打印返回结果，方便大家看清楚判断域名是否注册的原理。

```
$ php com.php
string(1024) "    Domain Name: ABC456.COM
    Registry Domain ID: 1656447674_DOMAIN_COM-VRSN
    Registrar WHOIS Server: whois.ename.com
    Registrar URL: http://www.ename.net
    Updated Date: 2022-05-03T14:44:07Z
    Creation Date: 2011-05-16T18:30:54Z
    Registry Expiry Date: 2023-05-16T18:30:54Z
    Registrar: eName Technology Co., Ltd.
    Registrar IANA ID: 1331
    Registrar Abuse Contact Email: abuse@ename.com
    Registrar Abuse Contact Phone: 86.4000044400
    Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhi
bited
    Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferP
rohibited
    Name Server: NS3.DNS.COM
    Name Server: NS4.DNS.COM
    DNSSEC: unsigned
    URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2022-12-07T10:31:50Z <<<

For more information on Whois status codes, please visit https://icann.org/epp

NOTICE: The expiration date displayed in this record is the date the
registrar's"
域名已被注册
```

执行代码，没有Bug，没有报错，直接给出来了我要的结果，域名 abc456.com 已经被注册了。

恐怖如斯！

我当年可是通过多番搜索查各种资料，才搞清楚了查询未注册域名的原理，现在 AI 直接一问就给出答案来了。

穿越修仙小说里才有的住在戒指里的有问必答老爷爷，现在人人都可以拥有了。

不需要穿越，不需要主角光环，只需要 ChatGPT 就足够了。

再来一个例子，下午，同事问哥飞，如何判断一批几万个手机号的运营商，我直接把问题扔给了ChatGPT，得到了满意的回答。

如何区分一个中国手机号的运营商？



可以通过手机号的前三位或前四位数字来区分一个中国手机号的运营商。



- 中国移动的手机号以 134、135、136、137、138、139、147、150、151、152、157、158、159、178、182、183、184、187、188、198 开头。
- 中国联通的手机号以 130、131、132、155、156、166、185、186 开头。
- 中国电信的手机号以 133、153、170、173、177、180、181、189、199 开头。
- 卫星通信的手机号以 1349 开头。

另外，也可以使用相应的 API 接口来查询手机号的运营商信息。

请列举一些免费查询手机号运营商的API



1. 手机号码归属地查询 API: <https://www.showapi.com/api/lookPoint/189>
2. 手机号码归属地查询 API: <https://www.juhe.cn/docs/api/id/11>
3. 手机号归属地查询 API: <https://www.tianapi.com/apiview/161>
4. 手机号码归属地查询 API: <https://www.ipip.net/product/mobile.html>
5. 手机号码归属地查询 API:
<https://market.aliyun.com/products/56928004/cmapi012061.html>



我们还可以向 ChatGPT 问编程语法使用经验，如如何用JS实现获取数组最大值？

SM

获取数组中的最大值，用JS实现



To find the maximum value in an array in JavaScript, you can use the `Math.max()` method, which returns the largest of zero or more numbers. For example, to find the maximum value in an array named `data`, you could use the following code:



Copy code

```
var data = [3, 4, 5, 6];
var maxValue = Math.max(...data);
console.log(maxValue); // Output: 6
```

In this example, the `Math.max()` method is called with the `...data` spread operator, which expands the elements of the `data` array as separate arguments to the `Math.max()` method. This allows the `Math.max()` method to find the maximum value in the array, which is then assigned to the `maxValue` variable. The maximum value is then logged to the console using `console.log()`.

Try again

用 ChatGPT 可以省去翻看查找文档的时间，请看下面几个例子。

你可以简单的问 ChatGPT 某个库的用法，如 `moment.js` 如何获取时间戳。



Moment.js 是一个 JavaScript 库，它可以用来处理时间和日期。要获取时间戳（即从 1970 年 1 月 1 日开始到某个日期的毫秒数），可以使用 Moment.js 的 `valueOf()` 函数：

Copy code

```
var timestamp = moment().valueOf();
```

在上面的代码中，`moment()` 函数用于获取当前时间，然后再使用 `valueOf()` 函数来获取时间戳。如果需要获取某个特定日期的时间戳，可以使用 Moment.js 的 `moment(dateString)` 函数，其中 `dateString` 是一个字符串，表示某个日期。例如：

Copy code

```
var dateString = "2022-12-07";  
var timestamp = moment(dateString).valueOf();
```

这样就可以获取指定日期的时间戳了。

Try again

如何用 `moment.js` 获取一个指定时间戳的6天前时间戳，ChatGPT 不仅给出例子来了，还一步一步讲解如何实现，甚至在最后告诉你可以用一句话连起来写。

SM

moment.js 获取6天前的时间戳，以指定时间戳为起点



如果您想以指定的时间戳作为起点，获取6天前的时间戳，您可以使用moment.js库中的`subtract`方法。



首先，需要创建一个moment对象，表示指定的时间戳：

Copy code

```
var timestamp = 1669910400000;  
var momentTime = moment(timestamp);
```

然后，使用`subtract`方法获取6天前的时间：

Copy code

```
var sixDaysAgo = momentTime.subtract(6, 'days');
```

最后，使用moment对象的`valueOf`方法获取它对应的时间戳：

Copy code

```
var timestamp = sixDaysAgo.valueOf();
```

您也可以在一行代码中完成上述操作：

Copy code

```
var timestamp = moment(1669910400000).subtract(6, 'days').valueOf();
```

请注意，上述示例假设您的时区为世界标准时间（UTC）。如果您在不同的时区，则可能需要使用moment对象的其他方法来处理时区问题。

Try again

ChatGPT 不仅能理解6天前，也能理解一周前。



如果您希望使用 Moment.js 获取一周前的时间戳，您可以使用 Moment.js 的 `subtract` 方法。

例如，假设您想要获取当前时间一周前的时间戳，您可以使用以下代码：

Copy code

```
var moment = require('moment');

// 获取当前时间
var now = moment();

// 使用 subtract 方法减去一周，即可得到一周前的时间
var aWeekAgo = now.subtract(7, 'days');

// 使用 unix 方法获取时间戳
var aWeekAgoTimestamp = aWeekAgo.unix();
```

在上面的代码中，我们首先引入了 Moment.js 库，然后使用 `moment` 方法获取了当前时间，接着使用 `subtract` 方法减去了一周，最后使用 `unix` 方法将时间转换为时间戳。

请注意，如果您想要获取一个具体时间一周前的时间戳，您可以按照相同的方式处理那个时间，而不是当前时间。

希望这些信息能够帮助您。如果您有任何其他问题，请随时发送消息给我。

Try again

你还可以直接提要求，让 ChatGPT 帮你写程序，如下面这个例子。



帮我使用 element + Vue2.7 实现一个简单的表格展示



可以使用 Element UI 和 Vue.js 来实现一个简单的表格展示功能。首先，需要将 Element UI 安装到您的 Vue.js 项目中。您可以使用以下命令完成安装：

Copy code

```
npm install element-ui --save
```

然后，在您的 Vue.js 应用程序中导入 Element UI：

Copy code

```
import ElementUI from 'element-ui';
import 'element-ui/lib/theme-chalk/index.css';
```

```
Vue.use(ElementUI);
```

接下来，您可以在您的组件的模板中使用 ``<el-table>`` 组件来创建一个表格。例如：

```
Copy code

<template>
  <el-table :data="tableData">
    <el-table-column
      prop="name"
      label="Name"
      width="180"
    >
    </el-table-column>
    <el-table-column
      prop="address"
      label="Address"
    >
    </el-table-column>
  </el-table>
</template>
```

在这个例子中，我们创建了一个名为 `tableData` 的数据源，并通过 ``<el-table>`` 组件将其展示在表格中。我们还使用了 ``<el-table-column>`` 组件来定义表格中的列，并通过 `prop` 属性来绑定数据源中的字段。

在实际应用中，您可以通过从后端服务器获取数据来填充表格。例如，您可以在组件的 `created` 生命周期钩子函数中发送 AJAX 请求来获取数据，然后将获取到的数据更新到 `tableData` 中。

```
Copy code

<script>
export default {
  data() {
    return {
      tableData: []
    };
  },
  created() {
    //
```

还可以问 ChatGPT 如何调用 ChatGPT 的接口实现连续对话。

如何在请求 ChatGPT API 时进行连续对话



ChatGPT 是一种基于 GPT-3 模型的聊天机器人，它能够进行连续的对话。要在请求 ChatGPT API 时进行连续对话，您需要将每次请求中的上下文设置为前一次请求的响应。这



样，ChatGPT 就能根据上下文继续对话。

给我一个这样的nodejs例子



虽然我不能提供具体的代码示例，但是我可以告诉您一般来说需要做什么。下面是一个简单的 node.js 代码示例，展示了如何使用 ChatGPT 进行连续对话：



Copy code

```
const fetch = require('node-fetch');

async function chatWithChatGPT(prompt) {
  // 设置请求参数
  const body = {
    prompt: prompt,
    token: '<YOUR_CHATGPT_TOKEN>',
    n: 1, // 请求的响应数量
  };

  // 发送请求
  const response = await fetch('https://api.openai.com/v1/completions', {
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
      'Authorization': `Bearer ${process.env.OPENAI_SECRET_KEY}`,
    },
    body: JSON.stringify(body),
  });

  // 解析响应
  const responseJson = await response.json();

  // 获取 ChatGPT 的响应
  const responseText = responseJson.data[0].text;

  // 返回响应
  return responseText;
}

async function main() {
  let context = '';

  while (true) {
    // 输入用户输入
    const input = await getUserInput();
```

```

// 将用户输入和上下文拼接起来
const prompt = context + input;

// 使用 ChatGPT 进行对话
const response = await chatWithChatGPT(prompt);

// 更新上下文
context = response;

// 输出 ChatGPT 的响应
console.log(response);
}
}



main();


```

请注意，上面的代码只是一个简单的示例，并不能直接使用。您需要替换`<YOUR_CHATGPT_TOKEN>`和`process.env.OPENAI_SECRET_KEY`的值，才能使用自己的 ChatGPT 令牌和 OpenAI 秘密密钥进行调用。

试试python?



好的，这里是一个使用 Python 语言实现的示例代码，展示了如何使用 ChatGPT 进行连续对话：  

 Copy code

```

import requests

def chat_with_chatgpt(prompt):
    # 设置请求参数
    data = {
        'prompt': prompt,
        'token': '<YOUR_CHATGPT_TOKEN>',
        'n': 1, # 请求的响应数量
    }

    # 发送请求
    response = requests.post('https://api.openai.com/v1/completions', json=data, headers={
        'Content-Type': 'application/json',
        'Authorization': f'Bearer {os.environ["OPENAI_SECRET_KEY"]}'
    })

```

```
# 解析响应
response_json = response.json()

# 获取 ChatGPT 的响应
response_text = response_json['data'][0]['text']

# 返回响应
return response_text

def main():
    context = ''

    while True:
        # 输入用户输入
        input = get_user_input()

        # 将用户输入和上下文拼接起来
        prompt = context + input


        # 使用 ChatGPT 进行对话
        response = chat_with_chatgpt(prompt)

        # 更新上下文
        context = response

        # 输出 ChatGPT 的响应
        print(response)

main()
```

和上面的代码示例一样，这段代码也只是一个简单的示例，并不能直接使用。您需要替换`<YOUR_CHATGPT_TOKEN>`和`os.environ["OPENAI_SECRET_KEY"]`的值，才能使用自己的 ChatGPT 令牌和 OpenAI 秘密密钥进行调用。

 Try again

当你编程没思路时，还可以向 ChatGPT 问思路。



如何设计一个无代码平台插件系统



设计一个无代码平台插件系统最主要的目的是让用户能够轻松地定制平台的功能。要实现这一目标，可以采取如下步骤：



1. 为平台插件系统定义一个清晰的架构，确定插件的角色和功能。插件可以包括用户界面元素、后台功能、数据处理逻辑等。
2. 为插件开发人员提供一套易于使用的工具和指南，帮助他们设计并创建插件。这些工具和指南可以包括可视化的插件编辑器、插件开发文档、插件样例等。
3. 在平台上提供一个插件商店，让用户可以浏览和下载各种插件。插件商店应该提供完整的信息和插件评价，方便用户选择合适的插件。
4. 建立一个插件审核机制，确保插件的质量和安全性。插件审核人员可以对提交的插件进行审查，并给出审核结果。
5. 为用户提供详细的插件安装和使用说明，帮助他们轻松地将插件安装到平台上，并使用他们的功能。
6. 定期更新平台插件系统，

如果你有取名困难症，你还可以让 ChatGPT 帮你取名和随机生成数据。



字段：库存名称，单位。帮我随机生成20条数据。要英文



Here are 20 examples of randomly generated data for the fields "Inventory Name" and "Unit":



1. Inventory Name: Pencils, Unit: Dozen
2. Inventory Name: Paper, Unit: Ream
3. Inventory Name: Stapler, Unit: Each
4. Inventory Name: Printer Ink, Unit: Cartridge
5. Inventory Name: Toner, Unit: Bottle
6. Inventory Name: Water Bottle, Unit: Each
7. Inventory Name: Coffee, Unit: Pound
8. Inventory Name: Milk, Unit: Gallon
9. Inventory Name: Bread, Unit: Loaf
10. Inventory Name: Batteries, Unit: Pack
11. Inventory Name: Soap, Unit: Bar
12. Inventory Name: Hand Sanitizer, Unit: Bottle
13. Inventory Name: Paper Towels, Unit: Roll
14. Inventory Name: Trash Bags, Unit: Box
15. Inventory Name: Tissues, Unit: Box

当然，如果你突然忘记了冒泡排序应该怎么写，也可以问 ChatGPT。



好的，那么来看看如何用Python实现一个冒泡排序算法。

首先，冒泡排序是一种比较简单的排序算法。它的基本思想是：比较相邻两个元素，如果顺序错误就交换位置。这样一直重复，直到没有需要交换的元素为止。

下面是一个实现冒泡排序的简单示例：

```
def bubble_sort(nums):  
    # 外层循环控制要比较几次  
    for i in range(len(nums) - 1):  
        # 内层循环控制每一次比较的两个元素  
        for j in range(len(nums) - i - 1):  
            if nums[j] > nums[j + 1]:  
                # 如果顺序错误，就交换两个元素的位置  
                nums[j], nums[j + 1] = nums[j + 1], nums[j]
```

上面的代码中，我们定义了一个名为`bubble_sort`的函数，它接收一个名为`nums`的列表作为输入，返回排序后的列表。在函数中，我们使用了两层循环来实现冒泡排序的基本过程。

在使用这个函数时，我们可以像下面这样调用它：

```
# 定义一个无序列表  
nums = [3, 1, 4, 2, 5]  
  
# 调用冒泡排序函数，排序后的结果保存在变量sorted_nums中  
sorted_nums = bubble_sort
```

写代码遇到 Bug 怎么办？不要慌，问问 ChatGPT。



在使用 Axios 时，在 `then` 方法中访问 `this` 可能会有困难。这是因为 `then` 方法会创建一个新的作用域，在这个作用域中，`this` 的值可能不是你期望的。



解决这个问题的一种方法是在使用 `then` 方法之前先将 `this` 存储到一个变量中，然后在 `then` 方法中使用这个变量。例如：

Copy code

```
let self = this;
axios.get('/user?ID=12345')
  .then(function (response) {
    self.data = response.data;
  })
  .catch(function (error) {
    console.log(error);
  });
```

另一种方法是使用箭头函数语法，因为箭头函数不会创建新的作用域，所以它们会继承外部作用域中的 `this`。例如：

Copy code

```
axios.get('/user?ID=12345')
  .then((response) => {
    this.data = response.data;
  })
  .catch((error) => {
    console.log(error);
  });
```

这两种方法都可以解决问题，具体选择哪一种方法取决于你的喜好。希望这对你有帮助。

想要啥，都可以问 ChatGPT 要。



我需要一个element ui 和Vue.js生成一个表格内嵌表单验证规则 且需要美化滚动条



要使用 element UI 和 Vue.js 生成一个表格内嵌表单验证规则，你可以使用 element UI 的 `el-table` 组件和 `el-form` 组件。具体实现方法如下：



1. 在表格中使用 `el-table` 组件，并在每一列中添加需要验证的表单组件，例如 `el-input`。
2. 使用 `el-form` 组件包裹整个表格，并在表单组件上添加验证规则。
3. 在表格中使用 `el-form-item` 组件包裹每一列，并在表单项组件上添加验证规则。

完整代码如下：

Copy code

```
<el-table :data="tableData" border>
  <el-table-column prop="name" label="姓名" width="180">
    <el-form-item label="姓名" prop="name" :rules="[ { required: true,
message: '请输入姓名', trigger: 'blur' } ]">
      <el-input v-model="tableData.name"></el-input>
    </el-form-item>
  </el-table-column>
  <el-table-column prop="age" label="年龄" width="180">
    <el-form-item label="年龄" prop="age" :rules="[ { required: true,
message: '请输入年龄', trigger: 'blur' } ]">
      <el-input v-model="tableData.age"></el-input>
    </el-form-item>
  </el-table-column>
</el-table>
```

如果需要美化滚动条，你可以使用一些第三方滚动条组件，例如 [vue-perfect-scrollbar](#)。你可以使用如下代码来实现：

Copy code

```
<template>
<div>
  <el-table :data="tableData" border>
    <el-table-column prop="name" label="姓名" width="180">
      <el-form-item label="姓名"
```



请继续



好了，例子就先讲这些了，剩下的交给大家去探索了。

本篇文章用到的部分图片来自于我的好友千古壹号建的微信群，如感兴趣，可以扫码入群。



ChatGPT首席体验群



该二维码7天内(12月15日前)有效，重新进入将更新

如果觉得本篇文章对你有启发，请给我一个赞，谢谢您！

#ChatGPT