链滴

什么是事件驱动编程?

作者：Vanessa
原文链接：https://ld246.com/article/1549942206210
来源网站：链滴

2019-02-11

回答

事件驱动编程是一种涉及通过发送和接受事件来构建应用程序的范式。当程序触发事件时，他可以通运行注册到该事件和上下文中的任意回调函数来进行响应，同时还可以将相关数据传递给响应的函数在这种模式下如果程序没有订阅任何函数，当事件被触发时也不会因为事件被发送到"异次元空间"而出错误。

一个常见的例子就是 DOM 元素上的事件监听，如 click 和 mouseenter，当点击或鼠标事件发生时调函数就会被运行，如：

```
document.addEventListener("click", function(event) {
    // 当用户点击 document 时，就会打印出 "hi"
    console.log('hi')
})
```

如果没有 DOM 的化，可以这样使用事件驱动：

```
const hub = createEventHub()
hub.on("message", function(data) {
  console.log(`${data.username} said ${data.text}`)
})
hub.emit("message", {
  username: "John",
  text: "Hello?"
})
```

在上面的代码片断中，通过 on 可以订阅 message 事件，通过 emit 可触发订阅好的 message 事件向其传递参数。

原文链接：什么是事件驱动编程?

加分回答

- 需要遵循发布订阅范式：消息的发送者不会将消息直接发送给特定的接收者，也不需要知道有哪些阅者的存在，但他需要将发布的消息分为不同的类别。同样，订阅者可以表达对一个或多个类别的兴，只接收感兴趣的消息，无需了解哪些发布者的存在。
- 通过运行任意被事件订阅的回调函数来响应事件的发生。
- 一个简单的发布订阅示例：

```
class PubSub {
    constructor(){
      this.events = {};
   }

   pub(eventName, data) {
      if (this.events[eventName]) {
         this.events[eventName].forEach(function(fn) {
            fn(data);
         });
      }
   }

   sub(eventName, fn) {
      this.events[eventName] = this.events[eventName] || [];
      this.events[eventName].push(fn);
   }
}
const pubsub = new PubSub()
pubsub.sub('message', (data) => {
   console.log(`sub message: ${data}`)
})
pubsub.pub('message', 'hi, b3log') // sub message: hi, b3log
```

返回总目录

每天 30 秒