自定义的window.onload事件不触发

原因：

- 1、window.onload事件在异步回调里定义
- 2、在定义window.onload事件之前，浏览器的资源已经加载完了。

示例1：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <img src="https://seopic.699pic.com/photo/50115/5158.jpg_wh1200.jpg" />

    <script type="text/javascript">

       function A(){
        setTimeout(function(){
            console.log("loading");
            window.addEventListener("load",function(){
                console.log("load1");
            });
        },0);

        window.addEventListener("load",function(){
            console.log("load2");
        });
       };

       A();
    </script>
</body>
</html>
```

结果：打印load1

示例2：

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
```

```
    <script type="text/javascript">

       function A(){
        setTimeout(function(){
            console.log("loading");
            window.addEventListener("load",function(){
                console.log("load1");
            });
        },0);

        window.addEventListener("load",function(){
            console.log("load2");
        });
       };

       A();
    </script>
</body>
</html>
```

结果：不打印load1

结论：

两者的区别在于前者加载了一个img，加载img是异步操作，setTimeout回调执行的时候img还未加载完，因此定义的onload可以在后续触发。

后者没有加载img，当setTimeout的回调执行的时候，onload已经触发过了，此时定义的onload事件没有机会触发了。

posted @ 2022-12-16 10:37 胡姐姐 阅读(430) 评论(0)