题目 - 11

- 将 url 参数解析为 JS 对象
- 手写数组 flatern ，考虑多层级
- 数组去重

将 url 参数解析为 JS 对象

```
// 传统方式，分析 search
function queryToObj() {
    const res = {}
    const search = location.search.substr(1) // 去掉前面的 `?`
    search.split('&').forEach(paramStr => {
        const arr = paramStr.split('=')
        const key = arr[0]
        const val = arr[1]
        res[key] = val
    })
    return res
}
```

将 url 参数解析为 JS 对象

```
// 使用 URLSearchParams
function queryToObj() {
    const res = {}
    const pList = new URLSearchParams(location.search)
    pList.forEach((val, key) => {
        res[key] = val
    })
    return res
}
```

手写 flatern 考虑多层级

```
flat([[1,2], 3, [4,5, [6,7, [8, 9, [10, 11]]]]])
// [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]
```

数组去重

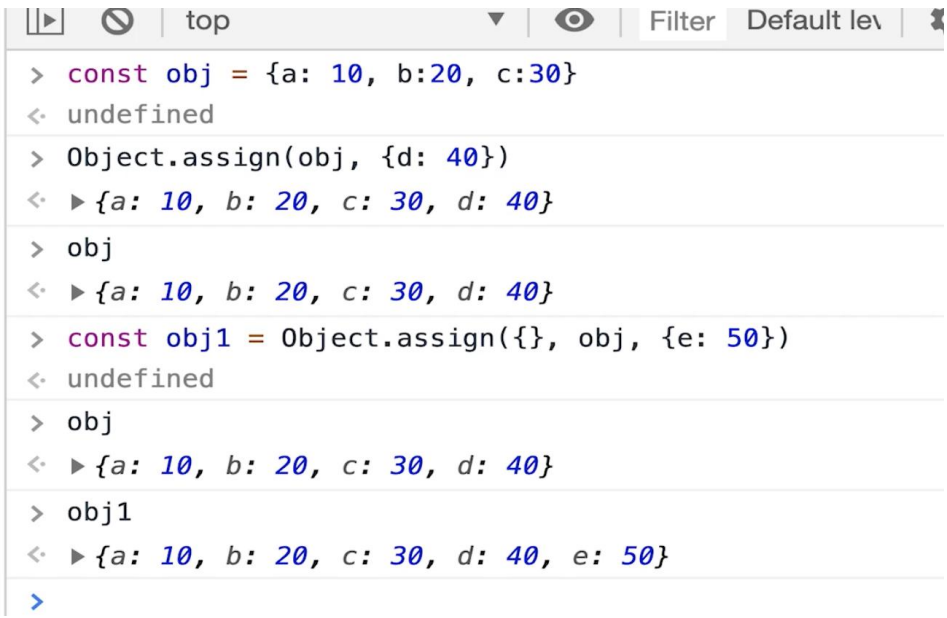
- 传统方式，遍历元素挨个比较、去重
- 使用 Set
- 考虑计算效率

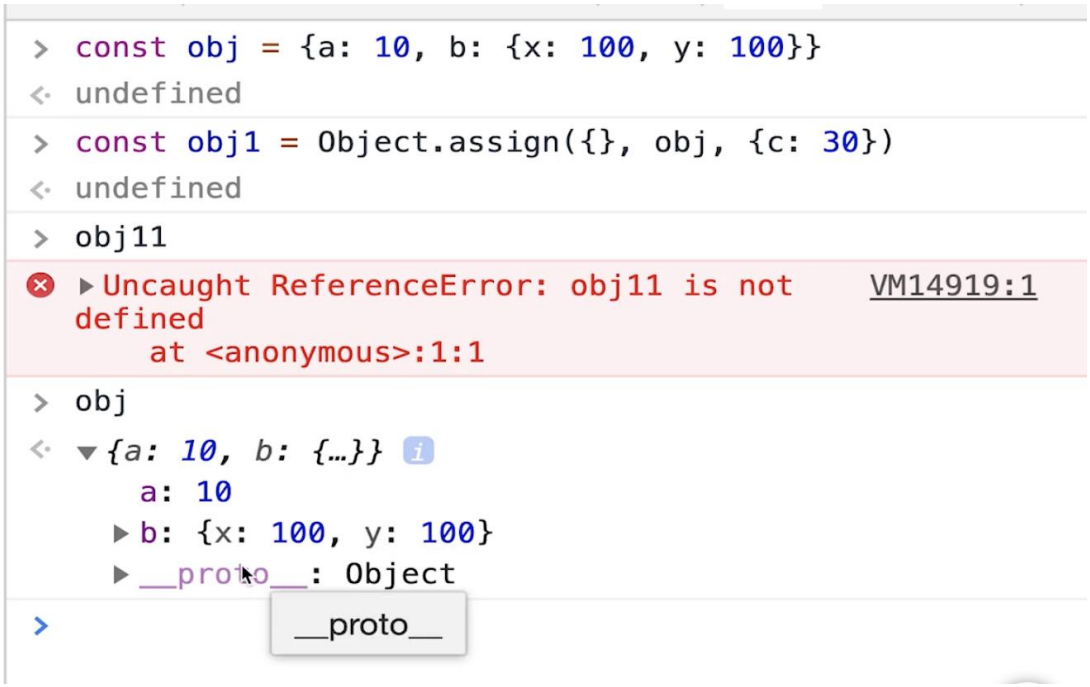
题目 - 12

- 手写深拷贝
- 介绍一下 RAF requestAnimateFrame
- 前端性能如何优化？一般从哪几个方面考虑？

注意，Object.assign 不是深拷贝！

第一层级的浅层拷贝





介绍 RAF requetAnimateFrame

- 要想动画流畅，更新频率要 60帧/s ，即 16.67ms 更新一次视图
- setTimeout 要手动控制频率，而 RAF 浏览器会自动控制
- 后台标签或隐藏 iframe 中，RAF 会暂停，而 setTimeout 依然执行

如何性能优化，从哪几个方面考虑

- 原则：多使用内存、缓存，减少计算、减少网路请求
- 方向：加载页面，页面渲染，页面操作流畅度