试题列表

以下序号对应试题基础源代码压缩包名称:

01 元旦派送

02 找一找

03 黑猫与白猫

04 实现类型判断

05 数组去重

元旦派送

介绍

为欢度元旦,我们一起做一个页面提供给客户送礼品活 动使用。

准备

开始答题前,需要先打开本题的项目代码文件夹,目录结构如下:

├──

│

├──

├──

L──

css

L── style .css

effect .gif

index .html

js

├── index .js

L── jquery .js

其中:

· css\style .css 是样式文件 。

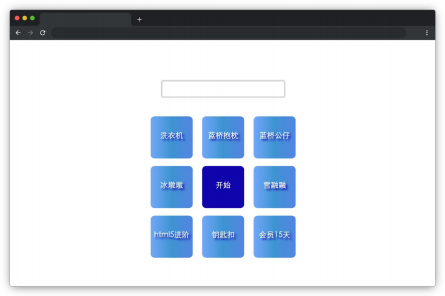
· index.html 是主页面。

· js\jquery.js 是 jQuery 文件。

· js\index .js 是需要补充的 js 文件。

· effect.gif 是最终实现的效果图。

在浏览器中预览 index.html 页面效果如下:



初始效果

目标

找到 index.js 中 rolling 函数,使用 jQuery 或者 js 完善此函数,达到以下效 果:

1. 选击开始后,以 class 为 li1 的元素为起选,黄色背景 (.active 类) 在奖项上顺时针转动。

2. 当转动停止后,将获奖提示显示到页面的 id 为 p1 元素中。获奖提示 必须包含奖项的名称, 该名称需与题目提供的名称完全一致。

3. 转动时间间隔和转动停止条件已给出,请勿修改。

完成后的效果见文件夹下面的 gif 图,图片名称为 effect.gif (提示:可以通过 VS Code 或者浏览器预览 gif 图片)。

规定

· 转动时间间隔和转动停止条件已给出,请勿修改。

· 请严格按照考试步骤操作,切勿修改考试默认提供项目中的文件名称、文

件夹路径等。

· 请勿修改项目中提供的 id、 class、函数名等名称,以免造成无法判题通

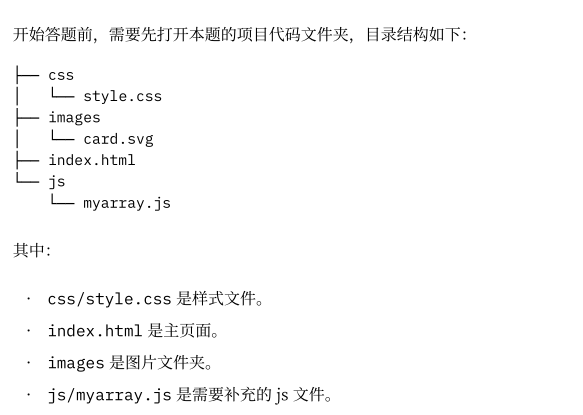
过。

找一找

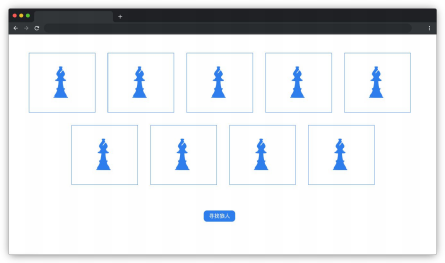
介绍

"狼制杀"是一款多制参与的策略类桌面游戏。本题我们一起完成一个简易的狼制杀 游戏,让我们找到其中的狼制。

准备



。



初始效果

目标

在本题 index .html 已经给出的数组中, 我们可以通过数组的 filter 方法: cardList.filter((item) => item.category == "werewolf") 返回一个都 是狼制的新数组。但是技术其管为了考验大家的技术,规定了在代码中任何地方都不 能出现 filter 关键字。所以我们需要封装一个 myarray 方法来实现类似数组 filter 的功能。

1. 狼制比较狡猾,筛选狼制的条件可能会变化,例如 item.name,请实现一 个通用的方法。

2. 完成封装后, 页面效果会自动完成, 效果见文件夹下 effect.gif (请使 用 VS Code 或者浏览器打开 gif 图片)。

规定

· 禁止在代码中出现 filter 关键字。

· 请严格按照考试步骤操作,切勿修改考试默认提供项目中的文件名称、文

件夹路径等。

· 请勿修改项目中提供的 id、 class、函数名等名称,以免造成无法判题通

过。

黑猫与白猫

介绍

本题我们将实现一个黑猫与白猫变化的效果。

准备

开始答题前,需要先打开本题的项目代码文件夹,目录结构如下:

├──

├──

│

│

│

├──

L──

effect.gif

images

├── h1.jpg

├── h2.jpg

L── w1.jpg

index.html

js

L── trafficcats.js

其中:

· index.html 是其页面。

· images 是图片文件夹。

· js/trafficcats.js 是需要补充的 js 文件。

· effect.gif 是最终实现的效果图。

在浏览器中预览 index.html 页面效果如下

初始效果

目标

完成 js/trafficcatts.js 文件中的 blackcat、whitecat 和 shite 函数,达 到以下效果:

1. 页面加载完成 5 秒后变成黑猫。

2. 在变成黑猫的 5 秒后,变成白猫(即 10 秒后变成白猫)。

3. 随后不再变化。

4. 请通过修改 display 属性来显示不同颜色的猫的图片。

规定

· 请通过修改 display 属性来显示不同颜色的猫的图片,以免造成无法判题

通过。

· 请勿修改项目中提供的 id、 class、函数名称、已有样式,以免造成无法

判题通过。

· 请严格按照考试步骤操作,切勿修改考试默认提供项目中的文件名称、文

件夹路径等。

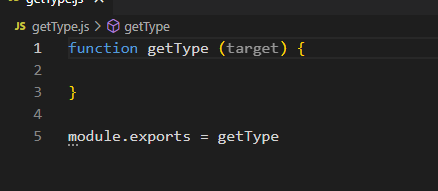
# **实现类型判断**

## **挑战介绍**

本节我们来挑战一道大厂面试真题 —— 实现类型判断。

## **挑战准备**

新建一个 getType.js 文件，在文件里写一个名为 getType 的函数，并导出这个函数，如下图所示：



## **挑战内容**

请封装 getType 的函数，能够以字符串的形式精准地返回数据类型。

要求返回的类型全部由小写字母组成。

示例：

| **输入** | **输出** |
| --- | --- |
| true | 'boolean' |
| 100 | 'number' |
| 'abc' | 'string' |
| 100n | 'bigint' |
| null | 'null' |
| undefined | 'undefined' |
| Symbol('a') | 'symbol' |
| [] | 'array' |
| {} | 'object' |
| function fn() {} | 'function' |
| new Date() | 'date' |
| /abc/ | 'regexp' |
| new Error() | 'error' |
| new Map() | 'map' |
| new Set() | 'set' |
| new WeakMap() | 'weakmap' |
| new WeakSet() | 'weakset' |

# **实现数组去重**

## **挑战介绍**

本节我们来挑战一道大厂面试真题 —— 实现数组去重。

## **挑战准备**

新建一个 myUniq.js 文件，在文件里写一个名为 myUniq 的函数，并导出这个函数，如下图所示：



## **挑战内容**

请封装一个函数，用来去掉数组中重复的元素。

该函数返回的是一个新的数组。

**考虑数组**中出现对象、null、undefined特殊情况。

示例：

输入：[1,1,1,2,2,3,3,4,5,1]

输出：[1,2,3,4,5]

输入：[1,undefined,undefined]

输出：[1,undefined]