Queue

一个简单的、实现了简单动画的队列的可视化界面

预览链接: https://lzzzzzt.github.io/queue

实现

```
let num = 0;
```

声明一个全局变量，用于标识栈中数字

```
function createBlock(className: string, value: number,): HTMLDivElement {
    let div = document.createElement('div');
    div.classList.add(className)
    div.innerHTML = `
        <span>${value}</span>
    `
    return div;
}
```

创建`Html`元素，并为其添加类名

```
const QueueHTML = document.querySelector<HTMLDivElement>('#queue')!
```

选择DOM树中用于显示队列的`div`

```
let queue: HTMLDivElement[] = [];
```

利用`Typescript`中的`Array`模拟队列的实现

```
const BtnQueuePush = document.querySelector<HTMLButtonElement>('#queue-unshift')!
BtnQueuePush.addEventListener('click', () => {
    queue.unshift(createBlock('queue-elem', num++)) // 创建新的队列元素
    queue.reverse().forEach(value => {
        let flag = true
        // 对其中元素和DOM中元素进行比对，若相同则不需要重新渲染，减小GPU压力，优化性能
        QueueHTML.childNodes.forEach(v => {
            if (v === value) {
                flag = false
            }
        })

        if (flag) {
            // 入场动画
            value.style.animation = "0.5s ease slideInLeft-enter"
            QueueHTML.appendChild(value)
        }
    })
    queue.reverse()
})
```

获取用于入队的按钮，并添加监听器

```
const BtnQueuePop = document.querySelector<HTMLButtonElement>('#queue-pop')!
BtnQueuePop.addEventListener('click', () => {
    let val = queue.pop()
    if (val) {
        // 离场动画
        val.style.animation = "0.5s ease slideInRight-leave"
        // 动画展示完全后再删除元素
        setTimeout(() => {
            QueueHTML.removeChild<HTMLDivElement>(val!)
        }, 400)
    } else {
        // 队列为空则提示
        alert('Queue is EMPTY!')
    }
})
```

获取用于出队的按钮，并添加监听器

*css文件见源码