



# 2048小游戏展示

202031101581 潘浩霖



# TABLE OF CONTENTS

---

**01**

引言

**02**

基本功能展示

**03**

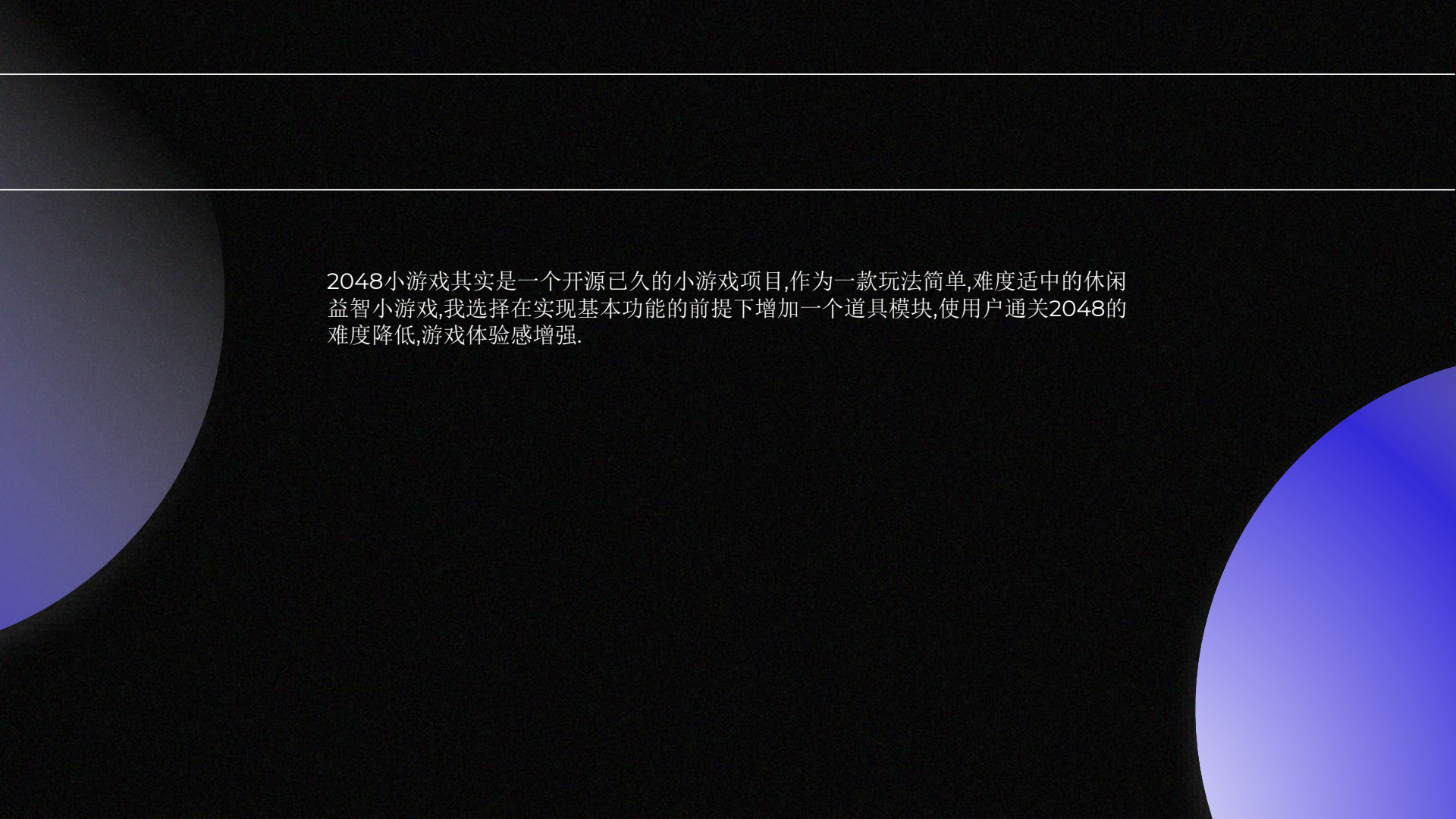
道具模块详解



The background is black with two large, semi-transparent blue circles. One circle is in the top right, and the other is in the bottom right. Two thin white horizontal lines are present: one above the text '01' and one below it.

# 01

## 引言



2048小游戏其实是一个开源已久的小游戏项目,作为一款玩法简单,难度适中的休闲益智小游戏,我选择在实现基本功能的前提下增加一个道具模块,使用户通关2048的难度降低,游戏体验感增强.



The background is black with two large, semi-transparent blue circles. One circle is in the top right corner, and the other is in the bottom right corner. Two thin white horizontal lines are also present, one above and one below the main text.

# 02

## 基本功能实现

因为设想中道具模块的重点应该是在逻辑设计与道具模块上,所以技术实现上选用了Jframe,用Jframe的优势在代码实现较简单,在生成算法写完后如果游戏运行出现问题找bug的难度偏低



2048

得分  
0

计分功能,记录当前玩家得分

操作: ↑ ↓ ← →, 按esc键重新开始

操作指示信息,帮助第一次玩的玩家如何开始游戏

道具

道具模块,辅助玩家进行游戏

# 游戏界面总览

- 顶部设置得分界面,让玩家查看当前得分
- 操作提示界面,帮助第一次游玩的玩家
- 道具系统,在滑动方块后会自动隐藏在方块界面下







03

# 道具模块详解



# 道具模块逻辑设计

01

定位

找到当前可生成新方块的位置

02

搜索

使用道具生成新方块将会生成一个一定能在下一次滑动被合并的新方块

03

选择

为确保在增强游戏体验的同时不至于让游戏过于简单,所以生成的新方块将是尽量小的方块

04

生成

在游戏界面中生成新的方块

# 几种会用道具模块的场景





# 几种会用道具模块的场景



# 几种会用道具模块的场景





# 几种会用道具模块的场景



# 道具生成方块的代码

```
public void actionPerformed(ActionEvent e) {  
    System.out.println("111");  
    int mi=2049;  
    int mi_x=0;  
    int mi_y=0;  
    int dx[]={-1,1,0,0};  
    int dy[]={0,0,-1,1};  
    for (int i = 0; i <= 3; i++) {  
        for(int j = 0; j <= 3;j++){  
            if(checks[i][j].value==0){  
                for(int ip=0;ip<4;ip++){  
                    int now_x=i+dx[ip];  
                    int now_y=j+dy[ip];  
                    if(now_x<0||now_x>3||now_y<0||now_y>3)continue;  
                    if(checks[now_x][now_y].value<mi&&checks[now_x][now_y].value>0){  
                        mi=checks[now_x][now_y].value;  
                        mi_x=i;  
                        mi_y=j;  
                    }  
                }  
            }  
        }  
    }  
    checks[mi_x][mi_y].value=mi;  
    repaint();  
    judgeGameOver();  
    this.requestFocus();  
}
```