

**本科毕业也论文代码**

**论文题目：微信小程序——九宫格数独设计与实现**

**指导老师： 宋桂琴**

**学生姓名： 裴文君**

**学 号： 20152101033**

**院 系： 计算机学院**

**专 业： 计算机科学与技术（师范）**

**毕业时间： 2019年6月**

**目录**

[第一章utils 1](#_Toc5984997)

[1.1 app.js 1](#_Toc5984998)

[1.2 app.json 3](#_Toc5984999)

[1.3 app.wxss 4](#_Toc5985000)

[第二章 pages 10](#_Toc5985001)

[2.1 about 10](#_Toc5985002)

[2.1.1 index,js 10](#_Toc5985003)

[2.1.2 index,json 12](#_Toc5985004)

[2.1.3 index.wxml 12](#_Toc5985005)

[2.1.4 index.wxss 14](#_Toc5985006)

[2.2 components 15](#_Toc5985007)

[2.2.1 loading.js 15](#_Toc5985008)

[2.2.2 loading.wxml 17](#_Toc5985009)

[2.2.3 loading.wxss 17](#_Toc5985010)

[2.3 index 17](#_Toc5985011)

[2.3.1 index.js 17](#_Toc5985012)

[2.3.2 index.json 43](#_Toc5985013)

[2.3.3 index.wxml 43](#_Toc5985014)

[2.3.4 index.wxss 48](#_Toc5985015)

[2.4 rank 59](#_Toc5985016)

[2.4.1 index.js 59](#_Toc5985017)

[2.4.2 index.wxml 60](#_Toc5985018)

[2.4.3 index.wxss 60](#_Toc5985019)

[2.5 record 60](#_Toc5985020)

[2.5.1 index.js 60](#_Toc5985021)

[2.5.2 index.json 71](#_Toc5985022)

[2.5.3 index.wxml 71](#_Toc5985023)

[2.5.4 index.wxss 73](#_Toc5985024)

[2.6 setting 74](#_Toc5985025)

[2.6.1 index.js 74](#_Toc5985026)

[2.6.2 index.json 78](#_Toc5985027)

[2.6.3 index.wxml 78](#_Toc5985028)

[2.6.4 index.wxss 79](#_Toc5985029)

# 第一章utils

## 1.1 app.js

//app.js

import { degree } from './utils/config.js'

App({

globalData: {

deviceInfo: null,

// 难度（遮挡率）

shadeDegree: .3,

// 性能优化

optimization: false,

requestHost: 'https://www.easy-mock.com/mock/5c7e2692e99519281016edc0/luckysodoku',

share: null,

degree: degree,

},

onLaunch: function () {

wx.getSystemInfo({

success: res => {

this.globalData.deviceInfo = res

},

})

this.globalData.shadeDegree = wx.getStorageSync('shadeDegree') || .3

// wx.getStorage({

// key: 'shadeDegree',

// success: res => {

// this.globalData.shadeDegree = res.data

// },

// })

this.globalData.optimization = wx.getStorageSync('optimization') || false

// wx.getStorage({

// key: 'optimization',

// success: res => {

// this.globalData.optimization = res.data

// },

// })

wx.request({

url: this.globalData.requestHost + '/degree',

method: 'GET',

success: res => {

if (res.data.code === 201) {

this.globalData.degree = res.data.result

}

}

})

},

onShow: function () {

wx.request({

url: this.globalData.requestHost + '/notice',

method: 'GET',

success: res => {

if (res.data.code === 201) {

if (res.data.result.showOnce) {

let hasShow = wx.getStorageSync('noticeShowOnce')

if (hasShow === '' || res.data.result.id !== hasShow) {

wx.showModal({

title: res.data.result.title,

content: res.data.result.content,

showCancel: false,

success: e => {

if (e.confirm) {

wx.setStorage({

key: 'noticeShowOnce',

data: res.data.result.id,

})

}

}

})

}

} else {

wx.showModal({

title: res.data.result.title,

content: res.data.result.content,

showCancel: false,

})

}

}

},

})

wx.request({

url: this.globalData.requestHost + '/share',

method: 'GET',

success: res => {

if (res.data.code === 201) {

this.globalData.share = res.data.result

}

}

})

},

adapterDegree(shadeDegree, returnType = 'title') {

let setDegree = parseInt(shadeDegree \* 100)

let title, range

this.globalData.degree.map(item => {

if (setDegree >= item.range[0] && setDegree <= item.range[1]) {

title = item.title

range = item.range

// 好像并不会跳过剩余的循环

return

}

})

if (returnType === 'range') {

return range

}

return title

}

})

## 1.2 app.json

{

"pages":[

"pages/index/index",

"pages/record/index",

"pages/setting/index",

"pages/rank/index",

"pages/about/index",

"pages/components/loading/loading"

],

"window":{

"navigationBarTitleText": "luckysodoku",

"navigationBarTextStyle": "white",

"navigationBarBackgroundColor": "#ffc0cb",

"backgroundTextStyle":"dark",

"backgroundColor": "#fff"

}

}

## 1.3 app.wxss

/\*\*app.wxss\*\*/

.container {

height: 100vh;

/\* display: flex;

flex-direction: column;

align-items: center;

justify-content: center; \*//\* padding: 100rpx 0; \*/

box-sizing: border-box;

margin: 0 10px;

}

button {

border-radius: 0;

}

button:after {

border: none;

}

.baseColor {

color: #ffc0cb;

}

.baseBgColor {

background: #ffc0cb;

}

.bgGreen {

background: #640764b4;

}

.bgGreen.button-hover {

background: #640764b4;

box-shadow: none;

}

.bgOrange {

background: #ffc0cb;

}

.bgOrange.button-hover {

background: #ffc0cb;

box-shadow: none;

}

.bgBlue {

background: #77cee9b6;

}

.bgBlue.button-hover {

background: #51d1f883;

box-shadow: none;

}

.btnGroup .cancleShade.button-hover {

background: #ffc0cb;

}

.empty {

text-align: center;

padding-top: 17px;

}

.empty text {

font-size: 14px;

color: #777;

}

.hdBar {

width: 100%;

height: 50px;

line-height: 50px;

border-bottom: 1px solid #eee;

clear: both;

}

.hdBar image {

width: 17px;

height: 17px;

/\* (50 - 17) / 2 \*/

padding: 17px 7px 17px 7px;

float: left;

}

.hdBar .title {

font-weight: bold;

float: left;

}

.hdBar .tip {

font-size: 10px;

color: #777;

}

.hdBar .right {

float: right;

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

height: 50px;

}

.bdContent {

text-indent: 1em;

font-size: 14px;

color: #555;

padding: 10px 0;

line-height: 2

}

/\* body \*/

.tbody.selected {

font-weight: bold;

color: #ffc0c7;

}

.view-hover {

background: rgba(255, 7, 143, 0.233);

}

/\* loading \*/

.sk-cube-grid {

width: 40px;

height: 40px;

margin: 0 auto;

background: transparent;

}

.sk-cube-grid .sk-cube {

/\* background: RGB(47, 117, 242)!important; \*/

background: #ffc0c7 !important;

width: 33%;

height: 33%;

background-color: #333;

float: left;

-webkit-animation: sk-cubeGridScaleDelay 1.3s infinite ease-in-out;

animation: sk-cubeGridScaleDelay 1.3s infinite ease-in-out;

}

.sk-cube-grid .sk-cube1 {

-webkit-animation-delay: 0.2s;

animation-delay: 0.2s;

}

.sk-cube-grid .sk-cube2 {

-webkit-animation-delay: 0.3s;

animation-delay: 0.3s;

}

.sk-cube-grid .sk-cube3 {

-webkit-animation-delay: 0.4s;

animation-delay: 0.4s;

}

.sk-cube-grid .sk-cube4 {

-webkit-animation-delay: 0.1s;

animation-delay: 0.1s;

}

.sk-cube-grid .sk-cube5 {

-webkit-animation-delay: 0.2s;

animation-delay: 0.2s;

}

.sk-cube-grid .sk-cube6 {

-webkit-animation-delay: 0.3s;

animation-delay: 0.3s;

}

.sk-cube-grid .sk-cube7 {

-webkit-animation-delay: 0s;

animation-delay: 0s;

}

.sk-cube-grid .sk-cube8 {

-webkit-animation-delay: 0.1s;

animation-delay: 0.1s;

}

.sk-cube-grid .sk-cube9 {

-webkit-animation-delay: 0.2s;

animation-delay: 0.2s;

}

@-webkit-keyframes sk-cubeGridScaleDelay {

0%, 70%, 100% {

-webkit-transform: scale3D(1, 1, 1);

transform: scale3D(1, 1, 1);

}

35% {

-webkit-transform: scale3D(0, 0, 1);

transform: scale3D(0, 0, 1);

}

}

@keyframes sk-cubeGridScaleDelay {

0%, 70%, 100% {

-webkit-transform: scale3D(1, 1, 1);

transform: scale3D(1, 1, 1);

}

35% {

-webkit-transform: scale3D(0, 0, 1);

transform: scale3D(0, 0, 1);

}

}

# 第二章 pages

## 2.1 about

### 2.1.1 index,js

// pages/about/index.js

let app = getApp()

Page({

/\*\*

\* 页面的初始数据

\*/

data: {

windowWidth: 355 - 20,

currentYear: '',

tip: null

},

/\*\*

\* 生命周期函数--监听页面加载

\*/

onLoad: function (options) {

this.setData({

windowWidth: (app.globalData.deviceInfo.windowWidth - 20) \* .7

})

let currentYear = new Date().getFullYear()

if (currentYear !== 2019) {

this.setData({

currentYear: '-' + new Date().getFullYear()

})

}

},

copy(e) {

let type = e.currentTarget.dataset.type

let data

if (type === 'email') {

data = '932089019@qq.com'

} else if (type === 'github') {

data = 'https://github.com/peibonana/learn\_luckynumber'

}

wx.setClipboardData({

data: data,

success: function (res) {

wx.showToast({

title: '已复制',

icon: 'success'

})

}

})

},

prevImg() {

},

/\*\*

\* 生命周期函数--监听页面初次渲染完成

\*/

onReady: function () {

},

/\*\*

\* 生命周期函数--监听页面显示

\*/

onShow: function () {

wx.request({

url: app.globalData.requestHost + '/tip',

method: 'GET',

success: res => {

if(res.data.code === 201){

this.setData({

tip: {

title: res.data.result.title,

content: res.data.result.content

}

})

}

}

})

},

/\*\*

\* 生命周期函数--监听页面隐藏

\*/

onHide: function () {

},

/\*\*

\* 生命周期函数--监听页面卸载

\*/

onUnload: function () {

},

/\*\*

\* 页面相关事件处理函数--监听用户下拉动作

\*/

onPullDownRefresh: function () {

},

/\*\*

\* 页面上拉触底事件的处理函数

\*/

onReachBottom: function () {

},

/\*\*

\* 用户点击右上角分享

\*/

// onShareAppMessage: function () {

// }

})

### 2.1.2 index,json

{

"navigationBarTitleText": "关于"

}

### 2.1.3 index.wxml

<!--pages/about/index.wxml-->

<view class='container'>

<view class='hdBar tip' wx:if="{{tip && tip.title}}">

<text class='title'>{{tip.title}}</text>

</view>

<view class='bdContent tip' wx:if="{{tip && tip.content}}">

{{tip.content}}

</view>

<view class='hdBar'>

<text class='title'>how to play？</text>

</view>

<view style="width:{{windowWidth}}px; height:{{windowWidth \* 1.7}}px; margin: 5px auto 10px;">

<image src='/images/demo.jpg' bindtap='prevImg'></image>

</view>

<view class='hdBar'>

<text class='title'>数据存储说明</text>

</view>

<view class='bdContent'>

在完成一轮游戏的情况下，您的成绩会缓存在本地，删除luckysudoku、卸载微信均会使成绩数据丢失。

</view>

<view class='hdBar'>

<text class='title'>用户信息与隐私说明</text>

</view>

<view class='bdContent'>

由于微信小程序严格的限制，luckysudoku现在、将来不会在未经您本人允许的情况下获取您任何信息，请放心使用。

</view>

<view class='hdBar'>

<text class='title'>Bug、建议与反馈</text>

</view>

<view class='bdContent'>

<view>Email:

<text class='link' bindtap='copy' data-type="email">932089019@qq.com<cover-image></cover-image></text>

</view>

<view>Repo:

<text class='link' bindtap='copy' data-type="github">https://github.com/peibonana/learn\_luckynumber</text>

</view>

</view>

<view class='hdBar'>

<text class='title'>菜鸟作者</text>

</view>

<view class='bdContent'>

<view>License: 裴文君</view>

</view>

<view class='footer'>

<view>luckysudoku v1.0</view>

<!-- <text>© </text> -->

<view>© 2019{{currentYear}} Lite Time,

<text bindtap='copy' class='link' data-type="website">Lite.Fun</text>

</view>

</view>

</view>

### 2.1.4 index.wxss

/\* pages/about/index.wxss \*/

image {

width: 100%;

height: 100%;

box-shadow: 1px 2px 7px rgba(0, 0, 0, 0.3);

}

.footer {

text-align: right;

font-size: 12px;

color: #777;

margin: 10px -10px 0;

background: #eee;

padding: 10px;

}

.footer view {

margin: 7px;

}

.footer .link {

font-style: italic;

}

text.link {

text-decoration: underline;

color: #337ab7;

/\* font-style: italic; \*/

}

.tip.hdBar {

border-bottom: none;

}

.tip.hdBar text {

width: 100%;

text-align: center;

font-weight: normal;

}

.tip.bdContent {

background: #eee;

margin: 0 -10px;

padding: 10px;

/\* font-size: 17px; \*//\* border-radius: 2px; \*/

}

.container .tip.bdContent:first-child {

text-indent: 0;

/\* margin-top: 10px; \*/

}

## 2.2 components

### 2.2.1 loading.js

// pages/components/loading/loading.js

Page({

/\*\*

\* 页面的初始数据

\*/

data: {

},

/\*\*

\* 生命周期函数--监听页面加载

\*/

onLoad: function (options) {

},

/\*\*

\* 生命周期函数--监听页面初次渲染完成

\*/

onReady: function () {

},

/\*\*

\* 生命周期函数--监听页面显示

\*/

onShow: function () {

},

/\*\*

\* 生命周期函数--监听页面隐藏

\*/

onHide: function () {

},

/\*\*

\* 生命周期函数--监听页面卸载

\*/

onUnload: function () {

},

/\*\*

\* 页面相关事件处理函数--监听用户下拉动作

\*/

onPullDownRefresh: function () {

},

/\*\*

\* 页面上拉触底事件的处理函数

\*/

onReachBottom: function () {

},

/\*\*

\* 用户点击右上角分享

\*/

onShareAppMessage: function () {

}

})

### 2.2.2 loading.wxml

<template name="loading">

<view class="loading" style="position: absolute; top:0; right: 0; bottom: 0; left: 0; background: rgba(0, 0, 0, .2); width: 100%; height: 100%; display: flex; justify-content: center; align-items: center; z-index: 99999">

<view class="sk-cube-grid">

<view class="sk-cube sk-cube1"></view>

<view class="sk-cube sk-cube2"></view>

<view class="sk-cube sk-cube3"></view>

<view class="sk-cube sk-cube4"></view>

<view class="sk-cube sk-cube5"></view>

<view class="sk-cube sk-cube6"></view>

<view class="sk-cube sk-cube7"></view>

<view class="sk-cube sk-cube8"></view>

<view class="sk-cube sk-cube9"></view>

</view>

</view>

</template>

### 2.2.3 loading.wxss

/\* pages/components/loading/loading.wxss \*/

## 2.3 index

### 2.3.1 index.js

//index.js

// slice只能拷贝一层

// import objDeepCopy from '../../utils/objDeepCopy.js'

// import { degree } from '../../utils/config.js'

//获取应用实例

var app = getApp()

// import { adapterDegree } from '../../utils/config.js'

import { parseTime } from '../../utils/moment.js'

Page({

data: {

// 数组

data: [],

boxSize: 17,

// 生成按钮禁用状态

btnDisabled: false,

// 生成情况

generateOk: false,

// 遮挡控制

shade: true,

// 初次渲染完成前时不显示按钮

init: false,

// panel位置

panelPosition: {

dx: -102,

dy: -112

},

panelShowAnimation: {},

showPanel: false,

boxCoords: {

x: 0,

y: 0

},

// pannel定位用

deviceInfo: null,

panelData: [1, 2, 3, 4, 5, 6, 7, 8, 9],

sideSize: 0,

// tooltip

toolTip: {

// ready, end, timing, pause, complete

type: 'ready',

content: '点击空白格子开始计时'

},

// 数独完成情况

complete: false,

// 数独剩余数字情况，索引排序

leave: [9, 9, 9, 9, 9, 9, 9, 9, 9],

menuAnimationTop: null,

menuAnimationTop: null,

menuAnimationMiddle: null,

menuAnimationBottom: null,

drawerToggle: false,

drawer: null,

showOptionAnimation: null,

toView: 'view0',

avatarPosition: {

x: -25,

y: 25

},

avatarTitle: {

text: 'luckysudoku',

step: 0

},

showResult: false,

resultIsGenerating: true,

optimization: false,

// showCanvasResult: false

// end data

showOption: false,

},

degree: .3,

percentDegree: '30%',

degreeTitle: '倔强青铜',

avatarShowTimes: 0,

generateSudokuSuccess: false,

// 回退

// history: [],

// 数独开始操作时间

startTime: 0,

pauseTime: 0,

timeInterval: null,

canvasResult: null,

panelOpen: false,

initArray(type) {

let array = new Array(9)

for (let i = 0; i < 9; i++) {

array[i] = new Array(9)

for (let j = 0; j < 9; j++) {

array[i][j] = undefined

}

}

if (type === 'init') {

this.setData({

data: array

})

}

return array

},

onLoad: function () {

let deviceInfo = app.globalData.deviceInfo

// console.log(deviceInfo)

this.setData({

deviceInfo: deviceInfo,

boxSize: (deviceInfo.windowWidth) / 9,

sideSize: (deviceInfo.windowHeight - deviceInfo.windowWidth) / 2,

})

this.initArray('init')

this.handleGenerateSudoku()

},

resetConfig() {

this.degree = app.globalData.shadeDegree

this.percentDegree = parseInt(this.degree \* 100) + '%'

this.degreeTitle = app.adapterDegree(this.degree)

this.setData({

optimization: app.globalData.optimization

})

wx.setNavigationBarTitle({

title: this.degreeTitle

})

},

reset() {

this.resetConfig()

this.hideOption()

// reset

clearInterval(this.timeInterval)

let data = this.data.data

if (data[0][0] === undefined) {

return

}

// if (!this.data.optimization) {

data.map((itemRow, idxRow) => {

itemRow.map((item, idx) => {

// 从右下角退回

if (this.data.optimization) {

item.successAnimation = this.basicAnimation(50, 0).scale(0).step().export()

} else {

item.successAnimation = this.basicAnimation(50, (8 - idx + 8 - idxRow) \* 50 - 50).scale(0).step().export()

}

})

})

// }

this.generateSudokuSuccess = false

this.startTime = 0

this.pauseTime = 0

this.timeInterval = null

this.canvasResult = null

this.setData({

data: data,

generateOk: false,

leave: [9, 9, 9, 9, 9, 9, 9, 9, 9],

toolTip: {

type: 'ready',

content: '点击空白格子开始计时'

},

btnDisabled: true,

shade: true,

complete: false,

showResult: false,

resultIsGenerating: true,

toView: 'view0'

})

},

handleGenerateSudoku() {

if (this.data.btnDisabled) {

return

}

// !== timing

if (!this.data.init || this.data.complete || this.data.toolTip.type === 'ready' || this.data.toolTip.type === 'end' || this.data.toolTip.type === 'drop') {

this.generateSudoku()

} else {

wx.showModal({

title: '提示',

content: '您本局成绩将不被记录，是否继续？',

success: res => {

if (res.confirm) {

this.generateSudoku()

} else {

wx.stopPullDownRefresh()

}

}

})

}

},

generateSudoku() {

this.reset()

let result = null

while (!this.generateSudokuSuccess) {

result = this.toGenerate()

}

wx.stopPullDownRefresh()

result.map((rowItem, rowIdx) => {

rowItem.map((item, idx) => {

result[rowIdx][idx] = {

value: item,

show: true,

// className: 'box',

x: idx,

y: rowIdx,

successAnimation: {}

}

})

})

if (this.data.shade) {

this.toggleShade(result, 'init')

} else {

this.setData({

data: result,

})

}

this.setData({

btnDisabled: false,

generateOk: true,

init: true,

})

},

toGenerate() {

// 只取值不刷新UI, 避免box为空

let array = this.initArray()

let time = new Date().getTime()

for (let j = 0; j < 9; j++) {

let idxInList = []

let notComplete = true

while (notComplete) {

idxInList = []

for (let k = 0; k < 9; k++) {

let avalibIdx = this.avalibleIdx(array[k], k, idxInList)

if (avalibIdx !== undefined) {

idxInList.push(avalibIdx)

}

}

if (idxInList.length === 9) {

notComplete = false

} else if (new Date().getTime() - time > 1000) {

return

}

}

// 要return，不map

for (let n = 0; n < idxInList.length; n++) {

array[n][idxInList[n]] = j + 1

if (j === 8 && n === 8) {

this.generateSudokuSuccess = true

return array

}

}

}

},

avalibleIdx(rowList, idxOfRowList, idxInList) {

let avalibleList = []

for (let m = 0; m < 9; m++) {

if (rowList[m] === undefined && idxInList.indexOf(m) === -1) {

if (idxOfRowList % 3 === 0) {

avalibleList.push(m)

} else {

let blockLastIndex = idxInList[idxInList.length - 1]

if ((blockLastIndex < 3 && m < 3) || ((blockLastIndex >= 3 && blockLastIndex < 6) && (m >= 3 && m < 6)) || (blockLastIndex >= 6 && m >= 6)) {

continue

} else {

if (idxOfRowList % 3 === 2) {

let blockAheadIdx = idxInList[idxInList.length - 2]

if ((blockAheadIdx < 3 && m < 3) || ((blockAheadIdx >= 3 && blockAheadIdx < 6) && (m >= 3 && m < 6)) || (blockAheadIdx >= 6 && m >= 6)) {

continue

}

}

avalibleList.push(m)

}

}

}

}

let resultList = Array.from(new Set(avalibleList))

return resultList[Math.floor(Math.random() \* resultList.length)]

},

toggleShade(newData, from = 'btn', callback) {

// console.log(this.degree)

// console.log('in')

// 点击事件默认传递一个事件对象，当参数是数组时表示初始化，并且为遮挡状态

let isArray = newData instanceof Array

let templist = isArray ? newData : this.data.data

let leave = [9, 9, 9, 9, 9, 9, 9, 9, 9]

templist.map(itemRow => (

itemRow.map((item, idx) => {

let result = isArray ? ((Math.random() >= this.degree) ? true : false) : (this.data.shade ? true : ((Math.random() >= this.degree) ? true : false))

itemRow[idx].show = result ? true : false

// itemRow[idx].className = result ? 'box' : 'box blank'

item.duplicate = []

item.fill = ''

item.rcl = false

item.showSame = false

let leaveIdx = item.value - 1

// 切换时show的item会再减1

leave[leaveIdx] = item.show ? leave[leaveIdx] - 1 : leave[leaveIdx]

})

))

this.setData({

data: templist,

shade: isArray ? true : !this.data.shade,

leave: leave

})

this.togglePanel(false)

if (from === 'init') {

// init时如果完成数独不记录

this.isComplete(templist, true)

} else {

clearInterval(this.timeInterval)

// 查看结果后处理办法

let tooltip = this.data.toolTip

tooltip = {

type: 'drop',

content: '请重新生成数独'

}

this.setData({

toolTip: tooltip

})

this.isComplete(templist)

// leave = this.data.shade ? [0,0,0,0,0,0,0,0,0] : leave

}

},

timing() {

if (this.data.toolTip.type === 'timing') {

return

}

this.startTime = this.startTime || new Date().getTime()

this.computeTime()

this.timeInterval = setInterval(() => {

this.computeTime()

}, 1000)

},

computeTime(){

this.pauseTime ? this.pauseTime += 1000 : ''

let time = Math.round(((this.pauseTime || new Date().getTime()) - this.startTime) / 1000)

let m = Math.floor(time / 60)

let s = time % 60 < 10 ? '0' + time % 60 : time % 60

let tooltip = {

type: 'timing',

content: m + ':' + s,

}

this.setData({

toolTip: tooltip

})

},

pause() {

clearInterval(this.timeInterval)

this.pauseTime = this.pauseTime || (new Date().getTime())

// console.log(this.pauseTime)

let tooltip = this.data.toolTip

this.setData({

toolTip: {

type: 'pause',

content: '用时' + tooltip.content + ', 已暂停'

}

})

this.clearStyle()

},

carryon() {

this.timing()

},

basicAnimation(duration, delay) {

let animation = wx.createAnimation({

duration: duration || 500,

timingFunction: "ease",

delay: delay || 0

});

return animation;

},

menuAnimate() {

if (this.data.drawerToggle) {

this.toggleDrawerHandler('toClose')

} else {

this.toggleDrawerHandler('toOpen')

// console.log(this.data.toolTip)

if (this.data.toolTip.type === 'timing') {

this.pause()

}

}

},

closeDrawer() {

this.toggleDrawerHandler('toClose')

},

toggleDrawerHandler(type) {

let toggle = true,

menuDx = '70%',

menuRotate = 30,

menuWidth = 30,

drawDx = '30%',

// h = 3,

w = 22

if (type === 'toClose') {

toggle = false,

menuDx = 0,

menuRotate = 0,

menuWidth = 15,

drawDx = '100%',

// h = 2,

w = 20

}

this.setData({

drawerToggle: toggle,

menuAnimation: this.basicAnimation().translate(menuDx).step().export(),

menuAnimationTop: this.basicAnimation().rotate(-menuRotate).width(w).step().export(),

menuAnimationMiddle: this.basicAnimation().width(menuWidth).step().export(),

menuAnimationBottom: this.basicAnimation().rotate(menuRotate).width(w).step().export(),

drawer: this.basicAnimation().right(drawDx).step().export()

})

},

togglePanel(toShow) {

if (this.data.optimization) {

if (!toShow) {

this.panelOpen = false

this.setData({

panelPosition: {

dx: -102,

dy: -112

},

})

} else {

this.panelOpen = true

this.setData({

panelShowAnimation: this.basicAnimation(200).scale(1).step().export()

})

}

} else {

let scale = 0

this.panelOpen = false

if (toShow) {

scale = 1

this.panelOpen = true

}

this.setData({

panelShowAnimation: this.basicAnimation(200).scale(scale).step().export()

})

}

},

clearStyle(type = 'all') {

let data = this.data.data

data.map((rowItem, rowIdx) => {

rowItem.map((item, idx) => {

if (type === 'rcl') {

item.rcl = false

} else if (type === 'same') {

item.showSame = false

} else {

item.rcl = false

item.showSame = false

}

})

})

this.setData({

data: data

})

},

tapBox(e) {

let tooltipType = this.data.toolTip.type

if (this.data.complete || tooltipType === 'pause' || tooltipType === 'end' || tooltipType === 'complete') {

return

}

this.hideOption()

let show = e.currentTarget.dataset.show

let value = e.currentTarget.dataset.value

// 激活的哪个box

let data = this.data.data

let boxCoords = this.data.boxCoords

boxCoords.x = e.currentTarget.dataset.x

boxCoords.y = e.currentTarget.dataset.y

if (show) {

if (this.panelOpen) {

// 如果点show的格子，panel还开着，先不显示相同数字

this.togglePanel(false)

} else {

this.togglePanel(false)

if (data[boxCoords.y][boxCoords.x].showSame) {

this.clearStyle()

} else {

this.clearStyle('rcl')

this.showSame(true, value)

}

this.setData({

toView: 'view' + (value - 1)

})

}

return

}

this.showSame(false)

// 判断为null的作用是避免瞬时多次点击

if (tooltipType !== 'drop' && this.timeInterval === null) {

this.timing()

}

// panel浮层位置

let panelPosition = this.data.panelPosition

// panel的一半是51

panelPosition.dx = e.detail.x - 51

panelPosition.dy = e.detail.y - 51

let screenWidth = this.data.deviceInfo.screenWidth

// 触摸点位置加半个panel超出屏幕宽度

if (e.detail.x + 51 >= screenWidth) {

panelPosition.dx = e.detail.x - 102

} else if (panelPosition.dx <= 0) {

// panel左边超出屏幕边界

panelPosition.dx = e.detail.x

}

data.map((rowItem, rowIdx) => {

rowItem.map((item, idx) => {

item.rcl = false

if (rowIdx === boxCoords.y || idx === boxCoords.x) {

item.rcl = true

}

})

})

this.about9Box(boxCoords.x, boxCoords.y).map(item => {

data[item.y][item.x].rcl = true

})

// panel数字

let panelData = [1, 2, 3, 4, 5, 6, 7, 8, 9]

if (data[boxCoords.y][boxCoords.x].fill) {

let idx = panelData.indexOf(data[boxCoords.y][boxCoords.x].fill)

panelData.splice(idx, 1)

// 始终让x居中

panelData.splice(4, 0, 'x')

}

this.setData({

panelPosition: panelPosition,

boxCoords: boxCoords,

data: data,

panelData: panelData,

})

this.togglePanel(true)

},

countDuplication(value, item, target) {

let boxCoords = this.data.boxCoords

let targetPosition = parseInt(boxCoords.y + '' + boxCoords.x)

let findTargetPositionInList = item.duplicate.lastIndexOf(targetPosition)

if (value === 'x') {

if (item.value === target.fill) {

item.duplicate.splice(findTargetPositionInList, 1)

}

} else {

if (item.value === value) {

item.duplicate.push(targetPosition)

} else {

if (findTargetPositionInList >= 0) {

item.duplicate.splice(findTargetPositionInList, 1)

}

}

}

},

showCbg() {

let data = this.data.data

data.map((itemRow, idxRow) => {

itemRow.forEach((item, idx) => {

// (idx + idxRow) \* 50 - 50 => 求 i \* j格子的值，即 j + i - 1

item.successAnimation = this.basicAnimation(50, (idx + idxRow) \* 50 - 50).scale(1).step().export()

})

})

this.setData({

data: data

})

},

isComplete(data, init = false) {

// console.log(leave)

// let result = leave.reduce((p, n) => p + n)

// if (result === 0) {

// item为show不检查为false

// item不为show的情况下item.fill为空为true, item.fill !== item.valueweitrue

let unComplete = data.some(itemRow => (itemRow.some(item => (!item.show ? (item.fill ? (item.fill !== item.value) : true) : false))))

// console.log(unComplete)

if (unComplete) {

this.setData({

complete: false,

})

return

}

// console.log(init)

if (init) {

// 取消初始化的成绩记录

let tooltip = this.data.toolTip

tooltip = {

type: 'end',

content: '运气太好了，没有一个能点的'

}

this.setData({

toolTip: tooltip,

complete: true,

shade: false

})

return

}

this.clearStyle()

this.togglePanel(false)

// console.log(this.data.toolTip)

if (this.data.toolTip.type === 'drop') {

this.setData({

complete: true,

shade: false,

showResult: false

})

return

}

this.data.optimization ? '' : this.showCbg()

clearInterval(this.timeInterval)

let tooltip = this.data.toolTip

let showTime = tooltip.type === 'ready' ? '0:00' : (tooltip.type === 'timing' ? tooltip.content : '')

tooltip = {

type: 'complete',

content: '用时' + showTime + ', 恭喜！'

}

this.setData({

toolTip: tooltip,

complete: true,

shade: false,

showResult: true

})

// 绘制结果

this.drawCanvas(showTime)

// 存缓存

let now = new Date().getTime()

let backData = {

startTime: this.startTime,

recordTime: now,

timeUse: this.pauseTime ? (this.pauseTime - this.startTime) : (now - this.startTime),

showTime: showTime,

shadeDegree: app.globalData.shadeDegree

}

// 即生成0到9的key

let storagePrimaryKey = app.adapterDegree(this.degree, 'range')[1] / 10 - 1

wx.getStorage({

key: 'record' + storagePrimaryKey,

success: function (res) {

let records = res.data

records.push(backData)

wx.setStorage({

key: 'record' + storagePrimaryKey,

data: records,

})

},

fail: () => {

let records = []

records.push(backData)

wx.setStorage({

key: 'record' + storagePrimaryKey,

data: records,

})

}

})

// 存一个总的记录，记录各等级最短时间等等

wx.getStorage({

key: 'records',

success: function (res) {

let records = res.data

// 用时短则替换

let target = records[storagePrimaryKey]

let counts = target.counts

let diffValue = (target.timeUse) - (backData.timeUse)

if (diffValue > 0 || (target.recordTime - target.startTime === 0)) {

records.splice(storagePrimaryKey, 1, backData)

}

records[storagePrimaryKey].counts = counts + 1

wx.setStorage({

key: 'records',

data: records,

})

},

fail: () => {

let records = []

for (let i = 0; i < 10; i++) {

if (i === storagePrimaryKey) {

backData.counts = 1

records.push(backData)

continue

}

records.push({

startTime: 0,

recordTime: 0,

counts: 0,

})

}

wx.setStorage({

key: 'records',

data: records,

})

}

})

// 存最近50条记录

wx.getStorage({

key: 'recordLatest',

success: function (res) {

let records = res.data

if (records.length > 50) {

records.shift()

}

records.push(backData)

wx.setStorage({

key: 'recordLatest',

data: records,

})

},

fail: () => {

let records = []

records.push(backData)

wx.setStorage({

key: 'recordLatest',

data: records,

})

}

})

// }

},

panelTap(e) {

let value = e.currentTarget.dataset.value

let boxCoords = this.data.boxCoords

let data = this.data.data

// 行和列

data.map((rowItem, rowIdx) => {

rowItem.map((item, idx) => {

// 只找show的

if (item.show) {

if ((rowIdx === boxCoords.y || idx === boxCoords.x) && (!(rowIdx === boxCoords.y && idx === boxCoords.x))) {

this.countDuplication(value, item, data[boxCoords.y][boxCoords.x])

}

}

})

})

// 九宫格

this.about9Box(boxCoords.x, boxCoords.y).map(item => {

if (boxCoords.x !== item.x || boxCoords.y !== item.y) {

let box = data[item.y][item.x]

if (box.show) {

// 排除在同行或同列的，上一步已经处理过

if (item.y !== boxCoords.y && item.x !== boxCoords.x) {

this.countDuplication(value, box, data[boxCoords.y][boxCoords.x])

}

}

}

})

// 计算剩余数字

let leave = this.data.leave.slice()

let target = data[boxCoords.y][boxCoords.x]

if (value === 'x') {

let idx = target.fill

leave[idx - 1] = leave[idx - 1] + 1

// } else if (target.fill === target.value){

} else {

if (target.fill === '') {

leave[value - 1] = leave[value - 1] - 1

} else {

let idx = parseInt(target.fill)

leave[idx - 1] = leave[idx - 1] + 1

leave[value - 1] = leave[value - 1] - 1

}

}

data[boxCoords.y][boxCoords.x].fill = (value === 'x') ? '' : value

this.isComplete(data)

this.setData({

data: data,

leave: leave

})

this.clearStyle('rcl')

this.togglePanel(false)

// 最多存100条记录

// if (this.history.length === 100) {

// this.history.pop()

// }

// this.history.push({

// boxCoords: boxCoords,

// data: data[boxCoords.y][boxCoords.x],

// panelValue: value

// })

},

about9Box(x, y) {

let range = {}

let list = []

if (x % 3 === 0) {

// x在0, 3, 6列

range.x = [x, x + 1, x + 2]

} else if (x % 3 === 1) {

// x在1，4，7列

range.x = [x - 1, x, x + 1]

} else {

// x在2，5，8列

range.x = [x - 2, x - 1, x]

}

if (y % 3 === 0) {

// y在0, 3, 6行

range.y = [y, y + 1, y + 2]

} else if (y % 3 === 1) {

// y在1，4，7行

range.y = [y - 1, y, y + 1]

} else {

// y在2，5，8行

range.y = [y - 2, y - 1, y]

}

range.y.map(y => {

range.x.map(x => {

list.push({

x: x,

y: y

})

})

})

// 返回当前九宫格坐标

return list

},

hidePanel() {

this.togglePanel(false)

},

showSame(showSame, value) {

let data = this.data.data

data.map((rowItem, rowIdx) => {

rowItem.map(item => {

if (item.show) {

if (showSame) {

if (item.value === value) {

item.showSame = true

} else {

delete item.showSame

}

} else {

delete item.showSame

}

}

})

})

this.setData({

data: data

})

},

showOption() {

if (this.data.showOption) {

this.hideOption()

} else {

this.setData({

showOption: true,

showOptionAnimation: this.basicAnimation().scale(1).step().export()

})

}

},

hideOption() {

this.setData({

showOption: false,

showOptionAnimation: this.basicAnimation(300).scale(0).step().export()

})

},

tapRowToShowSame(e) {

if(this.data.toolTip.type === 'pause'){

return

}

this.showSame(true, e.currentTarget.dataset.idx + 1)

},

moveAvatarEnd(e) {

let position = {

x: -25,

y: 25

}

this.setData({

avatarPosition: position

})

},

tapAvatar() {

let step = this.data.avatarTitle.step

if (step > 0) {

return

}

let showText = '别点我'

if (this.avatarShowTimes === 1) {

showText = '又是你'

} else if (this.avatarShowTimes === 2) {

showText = '还是你'

} else if (this.avatarShowTimes === 3) {

showText = '怎么老是你'

} else if (this.avatarShowTimes > 3) {

showText = '拉黑你'

// disabled

}

this.setData({

avatarTitle: {

text: showText,

step: 1,

}

})

// 超过三次后不再进行对话

if (this.avatarShowTimes > 3) {

this.tapAvatarShowTitle('luckysudoku', 0, 1500, null)

return

}

this.tapAvatarShowTitle('别点我', 2, 1000, null)

this.tapAvatarShowTitle('哎呦我去', 3, 2500, null)

this.tapAvatarShowTitle('我忘了自己叫啥了/(ㄒoㄒ)/~~', 4, 4000, null)

this.tapAvatarShowTitle('怎么办怎么办😭', 5, 6000, null)

this.tapAvatarShowTitle('唉，让我想想...', 6, 8000, null)

this.tapAvatarShowTitle('主人好像设定过', 7, 9000, null)

this.tapAvatarShowTitle('O(∩\_∩)O', 8, 10000, res => {

this.avatarShowTimes++

})

},

longTapAvatar(e) {

if (this.data.avatarTitle.step === 8) {

let avatarTitle = {

text: '不要乱碰本殿下',

step: 9

}

this.setData({

avatarTitle: avatarTitle

})

}

},

tapAvatarShowTitle(text, step, delay, callback) {

setTimeout(() => {

let avatarTitle = {

text: text,

step: step

}

this.setData({

avatarTitle: avatarTitle

})

callback ? callback() : ''

}, delay || 2000)

},

longTapAvatarTitle(e) {

let step = this.data.avatarTitle.step

if (step === 8 || step === 9) {

this.tapAvatarShowTitle('谢谢o(\*￣▽￣\*)ブ', 10, 0, res => {

this.tapAvatarShowTitle('luckysodoku', 0, 0, null)

})

}

},

/\*\*

\* 页面相关事件处理函数--监听用户下拉动作

\*/

onPullDownRefresh: function () {

this.handleGenerateSudoku()

},

drawCanvas(showTime) {

const ctx = wx.createCanvasContext('canvasResult')

ctx.setFontSize(12)

ctx.setFillStyle('#222222')

ctx.setTextAlign('left')

ctx.setTextBaseline('middle')

let cw = this.data.deviceInfo.windowWidth \* .7

let ch = this.data.sideSize - 5

ctx.clearRect(0, 0, cw, ch)

let padding = 10

let lineHeight = (ch - 10 - 10) / 4

let alignCenter = cw / 2

ctx.save()

ctx.setFillStyle('#ffffff')

ctx.fillRect(0, 0, cw, ch)

ctx.restore()

ctx.fillText('难度： ' + this.percentDegree, padding \* 2, padding + lineHeight \* 1 - lineHeight \* .5)

ctx.fillText('等级： ' + this.degreeTitle, padding \* 2, padding + lineHeight \* 2 - lineHeight \* .5)

ctx.fillText('用时： ' + showTime, padding \* 2, padding + lineHeight \* 3 - lineHeight \* .5)

ctx.fillText('完成时间： ' + parseTime(new Date().getTime()), padding \* 2, padding + lineHeight \* 4 - lineHeight \* .5)

ctx.save()

ctx.beginPath()

ctx.setStrokeStyle('#ffc107')

ctx.setLineWidth(2 \* 2)

// 6为字体大小的一半

ctx.moveTo(padding, padding + lineHeight \* 1 - lineHeight \* .5 - 6)

ctx.lineTo(padding, padding + lineHeight \* 4 - lineHeight \* .5 + 6)

ctx.stroke()

ctx.restore()

ctx.draw()

this.generateResult(this.data.sideSize - 5)

},

generateResult(height) {

// 最长一行时间的宽度大概为200

// 为防止底部黑边，拉伸图片，比例为分享图片比例

wx.canvasToTempFilePath({

canvasId: 'canvasResult',

x: 0,

y: 0,

width: 210,

height: height,

destWidth: 210,

destHeight: 210 / 1.27,

success: res => {

// console.log(res)

// let pic =

this.canvasResult = res.tempFilePath

this.setData({

resultIsGenerating: false

})

},

fail: res => {

// console.log(res)

setTimeout(() => {

this.generateResult(height)

}, 100)

}

})

},

/\*\*

\* 用户点击右上角分享

\*/

onShareAppMessage: function (e) {

let title, img = ''

let range = app.adapterDegree(this.degree, 'range')[1] / 10 - 1

let share = app.globalData.share

if (e.from === 'button') {

img = this.canvasResult ? this.canvasResult : ''

title = (share && share.range[range]) || '我在luckysodoku完成了一项数独挑战，成绩如下：'

// console.log(img)

} else {

title = (share && share.index) || '挑战更有趣的吧'

}

return {

title: title,

path: '/pages/index/index',

imageUrl: img,

}

},

route(e) {

let pageSize = getCurrentPages().length

// console.log(getCurrentPages())

let type = e.currentTarget.dataset.type

if(type === 'index'){

this.closeDrawer()

} else {

if (pageSize < 5) {

wx.navigateTo({

url: `/pages/${type}/index`,

})

} else {

wx.redirectTo({

url: `/pages/${type}/index`,

})

}

}

},

// onShow: function(){

// console.log(getCurrentPages())

// }

})

### 2.3.2 index.json

{

"enablePullDownRefresh": true

}

### 2.3.3 index.wxml

<view class='loading' wx:if="{{!generateOk}}">

<import src="/pages/components/loading/loading" />

<template is="loading" />

</view>

<view class="container" bindtap="hidePanel">

<view class='drawer' animation="{{drawer}}">

<view class='top'>

<movable-area>

<movable-view direction="all" x="{{avatarPosition.x}}" y="{{avatarPosition.y}}" out-of-bounds='true' damping='50' inertia='true' bindtouchend="moveAvatarEnd">

<image src='/images/logo.png' catchtap='tapAvatar' catchlongpress="longTapAvatar"></image>

</movable-view>

</movable-area>

<text catchlongtap='longTapAvatarTitle'>{{avatarTitle.text}}</text>

</view>

<!-- <navigator url="/pages/index/index"> -->

<view class='row' bindtap='route' data-type='index' hover-class='hover' hover-start-time='0' hover-stay-time='170'>

<image src='/images/ic\_apps\_black\_24dp\_2x.png'></image>

数独</view>

<!-- </navigator> -->

<!-- <navigator url="/pages/record/index"> -->

<view class='row' bindtap='route' data-type='record' hover-class='hover' hover-start-time='0' hover-stay-time='170'>

<image src='/images/ic\_trending\_up\_black\_24dp\_2x.png'></image>

统计</view>

<!-- </navigator> -->

<!-- <navigator url="/pages/rank/index"> -->

<view class='row' bindtap='route' data-type='rank' hover-class='hover' hover-start-time='0' hover-stay-time='170'>

<image src='/images/ic\_equalizer\_black\_24dp\_2x.png'></image>

排行</view>

<!-- </navigator> -->

<!-- <navigator url="/pages/setting/index"> -->

<view class='row' bindtap='route' data-type='setting' hover-class='hover' hover-start-time='0' hover-stay-time='170'>

<image src='/images/ic\_settings\_black\_24dp\_2x.png'></image>

设置</view>

<!-- </navigator> -->

<!-- <navigator url='/pages/about/index'> -->

<view class='row' bindtap='route' data-type='about' hover-class='hover' hover-start-time='0' hover-stay-time='170'>

<image src='/images/ic\_face\_black\_24dp\_2x.png'></image>

关于</view>

<!-- </navigator> -->

</view>

<view class="body">

<view class='shadow' wx:if="{{drawerToggle}}" bindtap='closeDrawer'></view>

<!-- 浮层 -->

<view class='panel' style="top: {{panelPosition.dy}}px; left: {{panelPosition.dx}}px;" animation="{{panelShowAnimation}}">

<block wx:for="{{panelData}}" wx:key="item\*this">

<view catchtap='panelTap' data-value="{{item}}" hover-class='hover' hover-start-time='0'>{{item}}</view>

</block>

</view>

<!-- options -->

<view class='optionBox' animation="{{showOptionAnimation}}">

<view class='hd'>

<text>选项</text>

<image src="/images/close.png" bindtap='hideOption'></image>

</view>

<view wx:if="{{init}}" class="btnGroup">

<button class="bgGreen" disabled="{{btnDisabled}}" catchtap="handleGenerateSudoku">重新生成</button>

<button class="{{shade ? 'bgOrange' : 'bgBlue'}}" catchtap="toggleShade">{{shade ? '查看结果' : '随机遮挡'}}</button>

</view>

</view>

<view class='top baseBgColor' style='height: {{sideSize - 5}}px;'>

<view class="{{drawerToggle ? 'menu menuToLeft' : 'menu'}}" bindtap='menuAnimate' animation="{{menuAnimation}}">

<text animation="{{menuAnimationTop}}"></text>

<text animation="{{menuAnimationMiddle}}"></text>

<text animation="{{menuAnimationBottom}}"></text>

</view>

<view class='{{toolTip.type==="timing" ? "toolTip timing" : "toolTip"}}'>{{toolTip.content}}</view>

<view class='pause' bindtap='pause' wx:if="{{toolTip.type==='timing'}}">

<text></text>

<text></text>

</view>

<view class='carry-on' bindtap='carryon' wx:if="{{toolTip.type==='pause'}}">

</view>

<view class='option' bindtap='showOption'>

<!-- <image src="/images/option.png"></image> -->

<image src='/images/ic\_lightbulb\_outline\_white\_24dp\_2x.png'></image>

</view>

<view class="title">luckysudoku--幸运数独</view>

<!-- end top -->

</view>

<view class="sudoku">

<view class='pauseShow' wx:if="{{toolTip.type==='pause'}}">PAUSE..</view>

<view wx:for="{{data}}" wx:key="" class="row">

<block wx:key="" wx:for="{{item}}">

<!-- rcl: row cross line 行与列光标香蕉 -->

<view class="{{item.show ? (item.duplicate[0] ? 'box duplicate' : 'box show') : 'box blank'}}" data-value='{{item.value}}' data-x="{{item.x}}" data-y="{{item.y}}" data-show="{{item.show}}" catchtap='tapBox' style="width: {{boxSize}}px; height: {{boxSize}}px;">

<!-- <text wx:if='{{item}}'>{{item}}</text>

<input wx:else /> -->

<view wx:if="{{!complete}}" class="{{item.rcl ? (item.showSame ? 'showSame rcl' : 'rcl') : (item.showSame ? 'showSame' : '')}}"></view>

<view class="cbg" animation="{{item.successAnimation}}"></view>

<text> {{item.show ? item.value : (item.fill || '')}}</text>

</view>

</block>

</view>

</view>

<view>

</view>

<view class='bottom'>

<view class="{{complete ? 'info result' : 'info'}}" style='height: {{sideSize - 5}}px'>

<scroll-view scroll-y class='scroll' wx:if="{{init}}" scroll-into-view="{{toView}}" scroll-with-animation wx:if="{{!complete}}">

<view class='wrap'>

<view class='item' wx:for="{{leave}}" wx:key="item\*this" wx:for-index="idx">

<view wx:if="{{item}}" style='height: {{(sideSize - 5)/3}}px; line-height: {{(sideSize - 5)/3}}px' id="view{{idx}}" data-idx="{{idx}}" catchtap='tapRowToShowSame'>

<view class='index'>{{idx+1}}</view>

<view class="{{item>0 ? 'imgs' : 'imgs sgmi'}}">

<image wx:for="{{item > 0 ? item : -item}}" wx:key="item\*this" src="/images/{{idx+1}}.jpg"></image>

</view>

<view class='counts'>{{item}}</view>

</view>

</view>

</view>

</scroll-view>

<view class='completeInfo' wx:if="{{showResult}}">

<!-- complete canvas -->

<view class='left' style="width: {{deviceInfo.windowWidth \* .7}}px">

<canvas canvas-id="canvasResult"></canvas>

</view>

<view class='right' style="width: {{deviceInfo.windowWidth \* .3 - 10}}px">

<button open-type="share" class="bgOrange" loading='{{resultIsGenerating}}'>{{resultIsGenerating ? '生成中' : '分享成绩'}}</button>

<button class='bgGreen' bindtap='handleGenerateSudoku'>再来一局</button>

</view>

</view>

<button wx:if="{{complete && !showResult}}" class='reGenerate' bindtap='handleGenerateSudoku' disabled="{{btnDisabled}}" style="line-height:{{sideSize - 5}}px">重新生成</button>

</view>

</view>

<!-- // body -->

</view>

<!-- // container -->

</view>

### 2.3.4 index.wxss

page, .container {

/\* position: relative;

width: 100%;

height: 100vh; \*/

margin: 0;

overflow: hidden;

}

.body {

width: 100%;

height: 100%;

display: flex;

flex-direction: column;

/\* justify-content: center; \*//\* align-items: center; \*/

}

.body .shadow {

position: absolute;

top: 0;

right: 0;

bottom: 0;

left: 0;

background: rgba(0, 0, 0, 0.3);

z-index: 2;

}

.body .top {

width: 100%;

position: relative;

}

.sudoku {

position: relative;

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

overflow: hidden;

/\* box-shadow: 0px 0px 7px rgba(0, 0, 0, 0.3); \*/

}

.box {

margin: 0;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

display: flex;

justify-content: center;

align-items: center;

color: #777;

background: #eee;

/\* borderradius对手机布局有影响 \*//\* border-radius: 2px; \*/

position: relative;

border-right: 1px dashed #ddd;

border-bottom: 1px dashed #ddd;

}

.box.blank {

color: #5cb85c;

}

.box .rcl, .box .showSame, .box .cbg {

position: absolute;

top: 0;

right: 0;

bottom: 0;

left: 0;

width: 100%;

height: 100%;

}

.box .rcl {

background: rgba(0, 0, 0, 0.1);

}

.box.duplicate {

color: #c94fd9;

}

.box .showSame {

/\* color: #f0ad4e; \*/

background: rgba(255, 7, 181, 0.3);

}

.box .cbg {

transform: scale(0);

background: rgba(255, 7, 201, 0.3);

}

.row {

display: flex;

direction: row;

justify-content: center;

align-content: center;

}

.row:nth-child(1) {

/\* border-top: 2px solid #ddd; \*/

}

.row:nth-child(3n) {

border-bottom: 2px solid #ddd;

}

.row:last-child {

border-bottom: none;

}

.row .box:first-child {

/\* border-left: 2px solid #ddd; \*/

}

.row .box:nth-child(3n) {

border-right: 2px solid #ddd;

}

.row .box:last-child {

border-right: none;

}

.row:nth-child(3n) .box {

border-bottom: none;

}

.title {

display: none;

font-size: 20px;

line-height: 2;

font-weight: bold;

}

.optionBox {

position: absolute;

width: 220px;

height: 170px;

top: calc(50% - 85px);

left: calc(50% - 110px);

z-index: 3;

background: #fff;

border-radius: 2px;

box-shadow: 1px 3px 7px rgba(0, 0, 0, 0.3);

transform: scale(0);

}

.optionBox .hd {

height: 30px;

line-height: 30px;

/\* border-bottom: 1px solid #eee; \*/

box-shadow: 0 1px 2px rgba(0, 0, 0, 0.3);

}

.optionBox .hd text {

font-size: 14px;

color: #222;

padding-left: 10px;

}

.optionBox .hd image {

padding: 6px;

width: 17px;

height: 17px;

line-height: 30px;

float: right;

}

.btnGroup {

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

width: 100%;

height: calc(100% - 31px);

margin: 0 auto;

}

.btnGroup button {

color: #fff;

width: 100px;

height: 40px;

border-radius: 2px;

font-size: 14px;

line-height: 40px;

padding: 0;

margin: 10px;

}

.panel {

width: 102px;

height: 102px;

/\* border: 1px solid red; \*/

text-align: center;

position: absolute;

box-shadow: 2px 3px 7px rgba(0, 0, 0, 0.3);

/\* transform: scale(0); \*/

z-index: 2;

}

.panel > view {

width: 30px;

height: 30px;

line-height: 30px;

/\* border: 1px solid blue; \*/

color: #777;

/\* background: rgba(0, 0, 0, 0.2); \*/

background: #fff;

float: left;

/\* margin, padding, border都算在width内 \*/

margin: 2px;

font-weight: bold;

}

.panel .hover {

background: #eee;

}

.toolTip {

position: absolute;

left: 10px;

bottom: 10px;

font-size: 15px;

color: #fff;

}

.toolTip.timing {

font-size: 24px;

}

.pause {

position: absolute;

left: calc(50% - 2px - 4px - 2px);

bottom: 10px;

display: flex;

flex-direction: row;

justify-content: center;

align-items: center;

}

.pause text {

width: 4px;

height: 16px;

background: #fff;

display: block;

margin: 2px;

}

.pauseShow {

position: absolute;

top: 0;

right: 0;

bottom: 0;

left: 0;

width: 100%;

height: 100%;

display: flex;

justify-content: center;

align-items: center;

z-index: 1;

/\* font-size: 2em; \*/

/\* font-size: \*/

color: #777;

/\* letter-spacing: 20px; \*/

/\* padding-left: 10px; \*/

/\* text-shadow: 1px 3px 5px rgba(0, 0, 0, 0.3); \*/

/\* background: rgba(0, 0, 0, 0.3); \*/

background: #eee;

}

.carry-on {

position: absolute;

/\* 定位在width, 为0，所以减去border的一半即可 \*/

left: calc(50% - 4px);

bottom: 10px;

width: 0;

height: 0;

border-width: 8px;

border-top-color: transparent;

border-right-color: transparent;

border-bottom-color: transparent;

border-left-color: #fff;

border-style: solid;

}

.option {

position: absolute;

right: 10px;

bottom: 7px;

}

.option image {

width: 27px;

height: 27px;

}

.menu {

position: absolute;

top: 10px;

left: 10px;

width: 100%;

z-index: 9;

}

.menu text {

/\* width: 20px; \*/

height: 2px;

background: #fff;

display: block;

margin: 4px;

border-radius: 3px;

}

.menu text:first-child, .menu text:last-child {

width: 20px;

}

.menu text:nth-child(2) {

width: 15px;

}

.container .drawer {

position: absolute;

top: 0;

right: 100%;

bottom: 0;

left: 0;

background: #fff;

z-index: 99999;

overflow: hidden;

box-shadow: 2px 3px 7px rgba(0, 0, 0, 0.3);

}

.container .drawer .top {

height: 120px;

display: flex;

justify-content: center;

align-items: center;

flex-direction: column;

white-space: nowrap;

/\* border-bottom: 1px solid #eee; \*/

box-shadow: 0 1px 2px rgba(0, 0, 0, 0.3);

/\* margin-bottom: 10px; \*/

}

.container .drawer .top movable-area {

width: 100%;

height: 100%;

left: 50%;

}

.container .drawer .top movable-view {

width: 50px;

height: 50px;

}

.container .drawer .top image {

width: 50px;

height: 50px;

/\* padding: 2px; \*/

border-radius: 50%;

box-shadow: 1px 2px 3px rgba(0, 0, 0, 0.3);

}

.container .drawer .top text {

color: #777;

font-size: 12px;

margin: 0 0 15px 0;

}

.container .drawer .row {

height: 50px;

line-height: 50px;

font-size: 14px;

border-bottom: none;

}

.container .drawer .row.hover {

/\* background: rgba(255, 193, 7, .3) \*/

background: #eee;

}

.container .drawer .row image {

width: 24px;

height: 24px;

/\* float: left; \*/

padding: 13px 7px;

}

.bottom .info {

background: #fff;

position: relative;

}

.bottom .info.result {

margin-top: -2px;

height: calc(100% + 2px);

border-top: 2px solid #ddd;

width: 100%;

}

.scroll {

/\* border: 1px solid #ddd; \*/

position: absolute;

left: 0;

right: 0;

bottom: 0;

height: 100%;

}

.scroll .item > view {

display: flex;

flex-direction: row;

justify-content: center;

align-content: center;

text-align: center;

font-size: 14px;

color: #777;

}

.scroll .item .index {

flex: 1.5;

background: rgba(255, 7, 214, 0.322);

color: #bbb;

border-right: 1px solid #ddd;

border-bottom: 1px solid #ddd;

}

.scroll .imgs {

flex: 7;

text-align: left;

line-height: 30px;

border-bottom: 1px dashed #ddd;

overflow: hidden;

display: block;

text-overflow: ellipsis;

white-space: nowrap;

}

/\* 负数 \*/

.scroll .sgmi {

text-align: right !important;

}

.scroll .counts {

flex: 1.5;

border-left: 1px dashed #ddd;

border-bottom: 1px dashed #ddd;

font-weight: bold;

}

.scroll .imgs image {

width: 20px;

height: 20px;

margin-top: 3px;

padding: 2px;

/\* transform: rotate(7deg); \*/

}

.completeInfo {

width: 100%;

height: 100%;

}

.completeInfo > view {

height: 100%;

float: left;

}

.completeInfo .right {

display: flex;

justify-content: center;

align-items: center;

flex-direction: column;

}

.completeInfo .right button {

width: 100%;

flex: 1;

font-size: 14px;

margin: 7px;

text-align: center;

padding: 0;

border-radius: 2px;

box-shadow: 1px 2px 2px rgba(0, 0, 0, 0.3);

color: #fff;

}

.reGenerate {

height: 100%;

}

## 2.4 rank

### 2.4.1 index.js

// pages/rank/index.js

Page({

/\*\*

\* 页面的初始数据

\*/

data: {

},

/\*\*

\* 生命周期函数--监听页面加载

\*/

onLoad: function (options) {

},

/\*\*

\* 生命周期函数--监听页面初次渲染完成

\*/

onReady: function () {

},

/\*\*

\* 生命周期函数--监听页面显示

\*/

onShow: function () {

},

/\*\*

\* 生命周期函数--监听页面隐藏

\*/

onHide: function () {

},

/\*\*

\* 生命周期函数--监听页面卸载

\*/

onUnload: function () {

},

/\*\*

\* 页面相关事件处理函数--监听用户下拉动作

\*/

onPullDownRefresh: function () {

},

/\*\*

\* 页面上拉触底事件的处理函数

\*/

onReachBottom: function () {

},

/\*\*

\* 用户点击右上角分享

\*/

// onShareAppMessage: function () {

// }

})

### 2.4.2 index.wxml

<!--pages/rank/index.wxml-->

<view class='container empty'>

<text>当前排行为一</text>

</view>

### 2.4.3 index.wxss

/\* pages/rank/index.wxss \*/

## 2.5 record

### 2.5.1 index.js

// pages/record/index.js

import fromNow, { computeTime } from '../../utils/moment.js'

// import { degree } from '../../utils/config.js'

// import { adapterDegree } from '../../utils/config.js'

let app = getApp()

Page({

/\*\*

\* 页面的初始数据

\*/

data: {

records: [],

countsAll: 0,

recordLatest: [],

canvasSize: 355,

showTip1: false,

showTip2: false,

loadingTip: '数据读取中...',

},

currentCanvasImg: null,

currentList: null,

/\*\*

\* 生命周期函数--监听页面加载

\*/

onLoad: function (options) {

this.setData({

canvasSize: app.globalData.deviceInfo.screenWidth - 20,

})

// 显示当前设定等级的canvas

let currentDegree = app.globalData.shadeDegree

this.renderRecords(currentDegree)

this.renderLastRecords()

// draw canvas

this.drawCanvas(currentDegree)

},

drawCanvas(degree) {

let range = app.adapterDegree(degree, 'range')

let title = app.adapterDegree(degree)

let storagePrimaryKey = range[1] / 10 - 1

wx.getStorage({

key: 'record' + storagePrimaryKey,

success: res => {

let list = []

res.data.map(item => {

list.push({

shadeDegree: parseInt(item.shadeDegree \* 100),

useTime: item.timeUse

})

})

this.draw(list, range, title)

},

fail: err => {

// 如果当前设置还未有记录，取完成数最多的一局显示

wx.getStorage({

key: 'records',

success: res => {

let result = res.data

let maxIdx = 0

result.map((item, idx) => {

if (item.counts && item.counts > 0) {

maxIdx = item.counts > result[maxIdx].counts ? idx : maxIdx

}

})

let target = result[maxIdx]

range = app.adapterDegree(target.shadeDegree, 'range')

title = app.adapterDegree(target.shadeDegree)

storagePrimaryKey = range[1] / 10 - 1

wx.getStorage({

key: 'record' + storagePrimaryKey,

success: res => {

let list2 = []

res.data.map(item => {

list2.push({

shadeDegree: parseInt(item.shadeDegree \* 100),

useTime: item.timeUse

})

})

this.draw(list2, range, title)

}

})

this.renderRecords(target.shadeDegree)

},

})

}

})

},

draw(data, range, title) {

const ctx = wx.createCanvasContext('myCanvas')

ctx.setFontSize(10)

ctx.setFillStyle('#777777')

let axisXData = range

let cw = this.data.canvasSize

let ch = this.data.canvasSize \* .7

ctx.clearRect(0, 0, cw, ch)

// 刻度大小

let tickSize = 3

let devideY = 7 // 7个刻度

let padding = {

top: 30,

right: 30,

bottom: 30,

left: 50,

}

// 坐标原点

let origin = {

x: padding.left,

y: ch - padding.bottom

}

let bottomRight = {

x: cw - padding.right,

y: ch - padding.bottom

}

let topLeft = {

x: padding.left,

y: padding.top

}

//绘制X轴

ctx.beginPath();

ctx.moveTo(origin.x, origin.y);

ctx.lineTo(bottomRight.x, bottomRight.y);

//绘制X轴箭头

ctx.lineTo(bottomRight.x - 7, bottomRight.y - 3);

ctx.moveTo(bottomRight.x, bottomRight.y);

ctx.lineTo(bottomRight.x - 7, bottomRight.y + 3);

//绘制Y轴

ctx.moveTo(origin.x, origin.y);

ctx.lineTo(topLeft.x, topLeft.y);

//绘制Y轴箭头

ctx.lineTo(topLeft.x - 3, topLeft.y + 7);

ctx.moveTo(topLeft.x, topLeft.y);

ctx.lineTo(topLeft.x + 3, topLeft.y + 7);

//绘制X方向刻度

let axisX = []

for (let i = axisXData[0]; i <= axisXData[1]; i++) {

axisX.push(i)

}

//计算刻度可使用的总宽度， 20为最右侧箭头内部遗留的距离

let avgWidth = (cw - padding.left - padding.right - 20) / (axisX[0] !== 0 ? axisX.length : (axisX.length - 1));

// 非0的刻度在原点右边一段距离处刻画

// axisX[0] !== 0 ? (origin.x += avgWidth) : ''

let xDx = axisX[0] !== 0 ? (origin.x + avgWidth) : origin.x

axisX.map((item, idx) => {

//移动刻度起点

ctx.moveTo(xDx + idx \* avgWidth, origin.y);

//绘制到刻度终点

ctx.lineTo(xDx + idx \* avgWidth, origin.y - (item !== 0 ? tickSize : 0));

//X轴刻度

ctx.setTextAlign('center')

ctx.fillText(

item,

xDx + idx \* avgWidth,

origin.y + 12);

})

// 坐标轴名字

ctx.setFontSize(10)

ctx.fillText(

'比例',

bottomRight.x + 7,

origin.y + 12);

let axisYData = []

data.map(item => {

axisYData.push(item.useTime)

})

//绘制Y方向刻度

// let axisYMax = axisYData.reduce((p, n) => (p >= n ? p : n))

// let axisYMin = axisYData.reduce((p, n) => (p <= n ? p : n))

// sort不传参排序无效？

axisYData.sort((a, b) => (a - b))

let min = axisYData[0]

let max = axisYData[axisYData.length - 1]

if (data.length < 7) {

devideY = data.length

}

// 30为y轴起始坐标预留距离， 底部10，顶部20

var avgValue = Math.round((max - min) / devideY)

var avgHeight = (ch - padding.top - padding.bottom - 30) / devideY;

for (let i = 0; i <= devideY; i++) {

ctx.setTextAlign('right')

ctx.setTextBaseline('middle')

if (devideY === 1) {

// 只有一条数据的情况下居中

let tickP = padding.top + (origin.y - padding.top) \* .3

ctx.moveTo(origin.x, tickP);

ctx.lineTo(origin.x + tickSize, tickP);

ctx.fillText(computeTime(min),

origin.x - 5,

tickP)

// 跳出循环

break

}

if (i === 0) {

ctx.moveTo(origin.x, origin.y - 10);

ctx.lineTo(origin.x + tickSize, origin.y - 10);

ctx.fillText(computeTime(min),

origin.x - 5,

origin.y - 10)

} else {

//绘制Y轴刻度

ctx.moveTo(origin.x, origin.y - 10 - i \* avgHeight);

ctx.lineTo(origin.x + tickSize, origin.y - 10 - i \* avgHeight);

//绘制Y轴文字

ctx.fillText(computeTime((avgValue \* i) + min),

origin.x - 5,

origin.y - 10 - i \* avgHeight);

}

}

ctx.setStrokeStyle('#777777')

ctx.stroke()

ctx.closePath()

ctx.fillText(

'用时',

origin.x - 5,

topLeft.y + -7)

ctx.save()

ctx.setFontSize(12)

ctx.fillText(

title,

bottomRight.x,

topLeft.y)

ctx.restore()

// 绘制数据点

let arcRadius = 1

if (data.length < 17) {

arcRadius = 2

}

// console.log(data)

data.map((item, idx) => {

ctx.beginPath()

// 在一个avgWidth左右范围内随机x坐标

// 是否除以2控制图的分散，除数越大越紧密

let randomNegative = Math.random() < .5 ? -1 : 1

let randomX = Math.random() \* avgWidth / 2 \* randomNegative

let degreeParse = this.parseShadeDegree(item.shadeDegree)

// (item.useTime - min) / avgValue \* avgHeight)

// [2, 100] 分7份每份14，取30的位置 (30 - 2) / 14 \* 每份的高度

ctx.arc(origin.x + degreeParse \* avgWidth + (degreeParse >= 1 ? randomX : 0), devideY === 1 ? (padding.top + (origin.y - padding.top) \* .3) : (origin.y - 10 - ((item.useTime - min) / avgValue \* avgHeight)), arcRadius, 0, 2 \* Math.PI)

ctx.setFillStyle('#ffc107')

ctx.fill()

ctx.closePath()

})

ctx.draw()

this.currentCanvasToImg(cw, ch)

this.currentList = data

},

currentCanvasToImg(cw, ch){

wx.canvasToTempFilePath({

// x: 0,

// y: 0,

// width: cw,

// height: ch,

// destWidth: cw,

// destHeight: cw \* 168 / 215,

canvasId: 'myCanvas',

success: res => {

this.currentCanvasImg = res.tempFilePath

},

fail: () => {

setTimeout(() => {

this.currentCanvasToImg()

}, 50)

}

})

},

parseShadeDegree(shadeDegree) {

if (shadeDegree === 0) {

return 0

} else {

// 取个位数字返回

return shadeDegree % 10 !== 0 ? shadeDegree % 10 : 10

}

},

renderRecords(degree) {

// console.log('in')

// records

let range = app.adapterDegree(degree, 'range')

wx.getStorage({

key: 'records',

success: res => {

let list = []

let countsAll = 0

// console.log(res.data)

res.data.map(item => {

let shade = parseInt(item.shadeDegree \* 100)

let selected = (shade >= range[0] && shade <= range[1]) ? true : false

list.push({

degree: app.adapterDegree(item.shadeDegree),

counts: item.counts,

showTime: item.showTime,

recordTime: fromNow(item.recordTime),

selected: selected

})

countsAll += item.counts

})

this.setData({

records: list,

countsAll: countsAll,

loadingTip: countsAll === 0 ? '请完成一局数独再来看看吧' : '读取成功，数据渲染中...'

})

},

fail: () => {

this.setData({

loadingTip: '请完成一局数独再来看看吧'

})

}

})

},

renderLastRecords() {

// 最近50条记录

wx.getStorage({

key: 'recordLatest',

success: res => {

// console.log(res)

let list = []

res.data.map(item => {

list.unshift({

recordTime: fromNow(item.recordTime),

showTime: item.showTime,

shadeDegree: parseInt(item.shadeDegree \* 100) + '%',

degree: app.adapterDegree(item.shadeDegree)

})

})

this.setData({

recordLatest: list

})

},

})

},

canvasToImg(e) {

wx.canvasToTempFilePath({

canvasId: 'myCanvas',

success: function (res) {

wx.saveFile({

tempFilePath: res.tempFilePath,

success: function (res) {

// var savedFilePath = res.savedFilePath

// show success

wx.showToast({

title: '保存成功',

icon: 'success',

duration: 1700

})

}

})

}

})

},

showTip(e) {

let type = e.currentTarget.dataset.type

if (type === 'tip1') {

this.setData({

showTip1: true

})

} else if (type === 'tip2') {

this.setData({

showTip2: true

})

}

},

drawItem(e) {

this.currentCanvasImg = null

this.currentList = null

let idx = e.currentTarget.dataset.idx

// degree值为0，1，2，3，4，5，6，7，8，9

// 为0的时候避免判断为false, 1的时候为第二级，避免为1成为第一级

let degree = idx === 0 ? 0.1 : idx \* .11

this.drawCanvas(degree)

this.renderRecords(degree)

},

/\*\*

\* 生命周期函数--监听页面初次渲染完成

\*/

onReady: function () {

},

/\*\*

\* 生命周期函数--监听页面显示

\*/

onShow: function () {

},

/\*\*

\* 生命周期函数--监听页面隐藏

\*/

onHide: function () {

},

/\*\*

\* 生命周期函数--监听页面卸载

\*/

onUnload: function () {

},

/\*\*

\* 页面相关事件处理函数--监听用户下拉动作

\*/

onPullDownRefresh: function () {

},

/\*\*

\* 页面上拉触底事件的处理函数

\*/

onReachBottom: function () {

},

/\*\*

\* 用户点击右上角分享

\*/

onShareAppMessage: function () {

let title

if(this.currentList){

let length = this.currentList.length

let degree = app.adapterDegree(this.currentList[0].shadeDegree / 100)

title = `共${this.data.countsAll}局，${length}局于${degree}`

} else {

title = `我在这里共完成${this.data.countsAll}局数独`

}

return {

title: title,

path: '/pages/index/index',

// imageUrl: this.currentCanvasImg,

}

}

})

### 2.5.2 index.json

{

"navigationBarTitleText": "统计"

}

### 2.5.3 index.wxml

<!--pages/record/index.wxml-->

<view class='container' wx:if="{{countsAll}}">

<view class='hdBar'>

<text class='title'>体验统计</text>

<image src='/images/tip.png' bindtap='showTip' data-type="tip1"></image>

<text class='tip' wx:if="{{showTip1}}">黄点表示完成的每一局数独，可长按保存到本地</text>

</view>

<canvas canvas-id="myCanvas" bindlongpress='canvasToImg' style="width: {{canvasSize}}px; height: {{canvasSize \* .7}}px;background:#fff;" />

<view class='all'>

</view>

<view class='statistic'></view>

<view class='records records1'>

<view class='thead'>

<text>等级</text>

<text>总局数</text>

<text>最短用时</text>

<text>时间</text>

</view>

<block wx:for="{{records}}" wx:if="{{item.counts}}" wx:key="showTime">

<view class="{{item.selected ? 'tbody selected' : 'tbody'}}" bindtap='drawItem' data-idx="{{index}}" hover-class="view-hover">

<text>{{item.degree}}</text>

<text>{{item.counts}}</text>

<text>{{item.showTime}}</text>

<text>{{item.recordTime}}</text>

</view>

</block>

<view class='tbody total'>

<text>\*总计</text>

<text>{{countsAll || 0}}</text>

<text>\</text>

<text>\</text>

</view>

</view>

<view class='hdBar newRecord'>

<text class='title'>最新记录</text>

<image src='/images/tip.png' bindtap='showTip' data-type="tip2"></image>

<text class='tip' wx:if="{{showTip2}}">记录您最近的50场数独完成情况</text>

</view>

<view class='records records2'>

<view class='thead'>

<text>记录时间</text>

<text>用时</text>

<text>难度</text>

<text>等级</text>

</view>

<view class='tbody' wx:for="{{recordLatest}}" wx:key="showTime">

<text>{{item.recordTime}}</text>

<text>{{item.showTime}}</text>

<text>{{item.shadeDegree}}</text>

<text>{{item.degree}}</text>

</view>

</view>

</view>

<view wx:else class='empty'>

<text>{{loadingTip}}</text>

</view>

### 2.5.4 index.wxss

/\* pages/record/index.wxss \*/

.container {

}

.newRecord {

margin-top: 17px;

border-bottom: none;

}

.records > view {

clear: both;

width: 100%;

height: 30px;

line-height: 30px;

}

.records > view.thead {

font-weight: bold;

border: 1px solid #eee;

height: 35px;

line-height: 35px;

background: #eee;

}

.records view.thead text {

font-size: 15px;

border-right: 1px solid #eee;

}

.records view.thead text:last-child {

border: none;

}

.records text {

float: left;

font-size: 14px;

text-align: center;

}

.records > view text:first-child {

width: 30%;

}

.records > view text:nth-child(2) {

width: 20%;

}

.records > view text:nth-child(3) {

width: 20%;

}

.records > view text:last-child {

width: calc(30% - 3px);

}

.records > view.tbody {

border-left: 1px dashed #eee;

border-bottom: 1px dashed #eee;

border-right: 1px dashed #eee;

}

.records view.tbody text {

border-right: 1px dashed #eee;

}

.records view.tbody text:last-child {

border: none;

}

.records .total {

font-weight: bold;

/\* background: #eee; \*/

}

.records2 {

margin-bottom: 10px;

}

## 2.6 setting

### 2.6.1 index.js

// pages/setting/index.js

// import { degree } from '../../utils/config.js'

import deepCopy from '../../utils/deepCopy.js'

let app = getApp()

Page({

/\*\*

\* 页面的初始数据

\*/

data: {

setDegree: 30,

degree: [],

showTip1: false,

showTip2: false,

showTip3: false,

optimizationChecked: false

},

/\*\*

\* 生命周期函数--监听页面加载

\*/

onLoad: function (options) {

let shade = parseInt(app.globalData.shadeDegree \* 100)

this.setData({

setDegree: shade,

optimizationChecked: app.globalData.optimization

})

this.parseDegree(shade)

},

parseDegree(shade) {

let degreeData = deepCopy(app.globalData.degree)

degreeData.map(item => {

if(shade >= item.range[0] && shade <= item.range[1]){

item.selected = true

} else {

item.selected = false

}

item.range = (item.range[0] !== 0 ? (item.range[0] + '%') : 0) + ' ~ ' + item.range[1] + '%'

})

this.setData({

degree: degreeData

})

},

slider2change(e) {

let value = e.detail.value

this.parseDegree(value)

let degree = parseFloat(value \* .01)

wx.setStorage({

key: 'shadeDegree',

data: degree,

})

app.globalData.shadeDegree = degree

},

showTip1: function () {

this.setData({

showTip1: true

})

},

showTip2: function () {

this.setData({

showTip2: true

})

},

showTip3: function(){

this.setData({

showTip3: true

})

},

changeOptimization(e) {

let result = e.detail.value

app.globalData.optimization = result

wx.setStorage({

key: 'optimization',

data: result,

})

},

clearStorage(){

wx.showModal({

title: '警告',

content: '将清除您本地所有成绩数据、设置记录。如未备份，将不可恢复，是否继续？',

success: res => {

if(res.confirm){

wx.clearStorage()

}

}

})

},

/\*\*

\* 生命周期函数--监听页面初次渲染完成

\*/

onReady: function () {

},

/\*\*

\* 生命周期函数--监听页面显示

\*/

onShow: function () {

},

/\*\*

\* 生命周期函数--监听页面隐藏

\*/

onHide: function () {

},

/\*\*

\* 生命周期函数--监听页面卸载

\*/

onUnload: function () {

},

/\*\*

\* 页面相关事件处理函数--监听用户下拉动作

\*/

onPullDownRefresh: function () {

},

/\*\*

\* 页面上拉触底事件的处理函数

\*/

onReachBottom: function () {

},

/\*\*

\* 用户点击右上角分享

\*/

// onShareAppMessage: function () {

// }

})

### 2.6.2 index.json

{

"navigationBarTitleText": "设置"

}

### 2.6.3 index.wxml

<!--pages/setting/index.wxml-->

<view class='container'>

<view class='hdBar'>

<text class='title'>性能优化</text>

<image src='/images/tip.png' bindtap='showTip1'></image>

<text class='tip' wx:if="{{showTip1}}">关闭游戏，建议重新开始</text>

<view class='right'>

<switch checked='{{optimizationChecked}}' bindchange="changeOptimization" color='#ffc107' />

</view>

</view>

<view class='hdBar'>

<text class='title'>难度</text>

<image src='/images/tip.png' bindtap='showTip2'></image>

<text class='tip' wx:if="{{showTip2}}">需要填入数字所占空格子比例。比例越高，难度越大</text>

</view>

<view class='bd'>

<slider class='slider' bindchange="slider2change" activeColor="#f0ad4e" value='{{setDegree}}' show-value/>

</view>

<view class='table'>

<view class='thead'>

<text>范围</text>

<text>程度</text>

</view>

<block wx:for="{{degree}}" wx:key='title'>

<view class="{{item.selected ? 'tbody selected' : 'tbody'}}">

<text>{{item.range}}</text>

<text>{{item.title}}</text>

</view>

</block>

</view>

<view class='hdBar'>

<text class='title'>清除数据</text>

<image src='/images/tip.png' bindtap='showTip3'></image>

<text class='tip' wx:if="{{showTip3}}">清除您本地所有数据</text>

<view class='right'>

<button size="mini" type="warn" bindtap='clearStorage'>清除</button>

</view>

</view>

</view>

### 2.6.4 index.wxss

/\* pages/setting/index.wxss \*/

.container {

justify-content: flex-start;

align-items: flex-start;

}

.slider {

width: 100%;

}

.bd {

width: 90%;

margin: 20px auto 30px;

}

.bd .slider {

margin: 0;

/\* margin: 10px; \*/

}

.table {

width: 100%;

}

.table > view {

width: 100%;

height: 35px;

border-bottom: 1px solid #eee;

}

.table > view text {

line-height: 35px;

padding-left: 7px;

width: calc(50% - 8px);

float: left;

border-right: 1px solid #eee;

font-size: 14px;

}

.table > view text:first-child {

border-left: 1px solid #eee;

width: calc(50% - 9px);

}

.table view.thead text {

border-top: 1px solid #eee;

font-weight: bold;

}