# C:\Users\Administrator\Desktop\代码关系图.jpg

# 代码清单

前端部分：

游戏开始按钮控制脚本

export default class Game\_con extends Laya.Script {

constructor() {

super();

/\*\* @prop {name:money, tips:"金钱", type:Int, default:1000}\*/

let money = 1000;

/\*\* @prop {name:numType, tips:"数字类型示例", type:Number, default:1000}\*/

let numType = 1000;

/\*\* @prop {name:strType, tips:"字符串类型示例", type:String, default:"hello laya"}\*/

let strType = "hello laya";

/\*\* @prop {name:boolType, tips:"布尔类型示例", type:Bool, default:true}\*/

let boolType = true;

}

onClick(){

Laya.Scene.open("Game/gameChose.scene")

}

}

游戏选择剧本控制按钮脚本

export default class Gamechose extends Laya.Script {

constructor() {

super();

/\*\* @prop {name:intType, tips:"整数类型示例", type:Int, default:1000}\*/

let intType = 1000;

/\*\* @prop {name:numType, tips:"数字类型示例", type:Number, default:1000}\*/

let numType = 1000;

/\*\* @prop {name:strType, tips:"字符串类型示例", type:String, default:"hello laya"}\*/

let strType = "hello laya";

/\*\* @prop {name:boolType, tips:"布尔类型示例", type:Bool, default:true}\*/

let boolType = true;

}

onClick(){

Laya.Scene.open("Game/gameStartchose.scene")

}

}

游戏加载按钮控制脚本

export default class gameLoad extends Laya.Script {

constructor() {

super();

/\*\* @prop {name:intType, tips:"整数类型示例", type:Int, default:1000}\*/

let intType = 1000;

/\*\* @prop {name:numType, tips:"数字类型示例", type:Number, default:1000}\*/

let numType = 1000;

/\*\* @prop {name:strType, tips:"字符串类型示例", type:String, default:"hello laya"}\*/

let strType = "hello laya";

/\*\* @prop {name:boolType, tips:"布尔类型示例", type:Bool, default:true}\*/

let boolType = true;

}

onClick(){

Laya.Scene.open("Game/gameStart1.scene")

}

}

游戏势力选择进入游戏控制脚本

export default class gameStartchose extends Laya.Script {

constructor() {

super();

/\*\* @prop {name:intType, tips:"整数类型示例", type:Int, default:1000}\*/

let intType = 1000;

/\*\* @prop {name:numType, tips:"数字类型示例", type:Number, default:1000}\*/

let numType = 1000;

/\*\* @prop {name:strType, tips:"字符串类型示例", type:String, default:"hello laya"}\*/

let strType = "hello laya";

/\*\* @prop {name:boolType, tips:"布尔类型示例", type:Bool, default:true}\*/

let boolType = true;

}

onClick(){}//触发点击转换界面

loadScene(){}//进入加载界面

}

返回按钮控制脚本

export default class gameReturn extends Laya.Script {

constructor() {

super();

/\*\* @prop {name:intType, tips:"整数类型示例", type:Int, default:1000}\*/

let intType = 1000;

/\*\* @prop {name:numType, tips:"数字类型示例", type:Number, default:1000}\*/

let numType = 1000;

/\*\* @prop {name:strType, tips:"字符串类型示例", type:String, default:"hello laya"}\*/

let strType = "hello laya";

/\*\* @prop {name:boolType, tips:"布尔类型示例", type:Bool, default:true}\*/

let boolType = true;

}

onClick(){

Laya.Scene.open("Game/gameStart.scene")

}

}

游戏主界面逻辑代码

export default class gameMainUi extends Laya.Scene {

constructor() {

super();

/\*\* @prop {name:intType, tips:"整数类型示例", type:Int, default:1000}\*/

let intType = 1000;

/\*\* @prop {name:numType, tips:"数字类型示例", type:Number, default:1000}\*/

let numType = 1000;

/\*\* @prop {name:strType, tips:"字符串类型示例", type:String, default:"hello laya"}\*/

let strType = "hello laya";

/\*\* @prop {name:boolType, tips:"布尔类型示例", type:Bool, default:true}\*/

let boolType = true;

}

openMessage(){}//打开详细信息界面

openSetting(){}//打开设置界面

openMilitary(){}//打开军事系统

openInterior(){}//打开内政系统

openTactics(){}//打开策略系统

openPersonnal(){}//打开人事系统

openNext(){}//下一回合按钮控制

setCityMessage(){}//城池信息设置

setMessage(){}//详细信息界面控制

setMap(){}//放入地图并控制地图

}

游戏界面加载并转换游戏界面

export default class gameLoading extends Laya.Scene{

constructor() {

super();

/\*\* @prop {name:intType, tips:"整数类型示例", type:Int, default:1000}\*/

let intType = 1000;

/\*\* @prop {name:numType, tips:"数字类型示例", type:Number, default:1000}\*/

let numType = 1000;

/\*\* @prop {name:strType, tips:"字符串类型示例", type:String, default:"hello laya"}\*/

let strType = "hello laya";

/\*\* @prop {name:boolType, tips:"布尔类型示例", type:Bool, default:true}\*/

let boolType = true;

}

loadScene(){}//加载进入场景

open(){}//打开场景

close(){}//关闭完成后，调用此方法（如果有关闭动画，则在动画完成后执行）

}

排行榜设置脚本

export default class setRanking extends Laya.Script{

constructor() {

super();

/\*\* @prop {name:intType, tips:"整数类型示例", type:Int, default:1000}\*/

let intType = 1000;

/\*\* @prop {name:numType, tips:"数字类型示例", type:Number, default:1000}\*/

let numType = 1000;

/\*\* @prop {name:strType, tips:"字符串类型示例", type:String, default:"hello laya"}\*/

let strType = "hello laya";

/\*\* @prop {name:boolType, tips:"布尔类型示例", type:Bool, default:true}\*/

let boolType = true;

}

onAwake(){

setImage()//在组件加入舞台后进行赋值操作

}

}

后端部分：

## 类常量设计及实现：

军队类属性的定义及实现

class Army{

constructor(armyID,armyMainID,armyAssisiant1ID,armyAssisiant2ID,armySoldier,armyFood,armyMoney){

this.armyID=armyID;

this.armyMainID=armyMainID;

this.armyAssisiant1ID=armyAssisiant1ID;

this.armyAssisiant2ID=armyAssisiant2ID;

this.armySoldier=armySoldier;

this.armyFood=armyFood;

this.armyMoney=armyMoney;

}

} //军队类属性的定义

set ArmyID(armyID)

get ArmyID() //军队ID的实现

set ArmyMainID( armyMainID)

get ArmyMainID() //军队主将ID的实现

set ArmyAssisiant1ID( armyAssisiant1ID)

get ArmyAssisiant1ID() //军队副将1ID的实现

set ArmyAssisiant2ID( armyAssisiant2ID)

get ArmyAssisiant2ID() //军队副将1ID的实现

set ArmySoldier( armySoldier)

get ArmySoldier() //军队士兵的实现

set ArmyFood( armyFood)

get ArmyFood() //军队兵粮的实现

set ArmyMoney( armyMoney)

get ArmyMoney() //军队财富的实现

城池类属性的定义及实现

class City{

constructor(cityID,cityName,cityFood,cityFarm,cityMoney,cityBussiness,cityDefense,citySoldier,cityLoyal,cityBelongFactionID){

this.cityID=cityID;

this.cityName=cityName;

this.cityFood=cityFood;

this.cityFarm=cityFarm;

this.cityMoney=cityMoney;

this.cityBussiness=cityBussiness;

this.cityDefense=cityDefense;

this.citySoldier=citySoldier;

this.cityLoyal=cityLoyal;

this.cityBelongFactionID=cityBelongFactionID;

} //城池类属性的定义

set CityID(cityID)

get CityID() //城池ID的实现

set CityName(cityName)

get CityName() //城池姓名的实现

set CityFood(cityFood)

get CityFood() //城池粮草的实现

set CityFarm( cityFarm)

get CityFarm() //城池农业的实现

set CityMoney( cityMoney)

get CityMoney() //城池财富的实现

set CityBussiness(cityBussiness)

get CityBussiness() //城池商业的实现

set CityDefense(cityDefense)

get CityDefense() //城池防御的实现

set CitySoldier( citySoldier)

get CitySoldier() //城池士兵的实现

set CityLoyal( cityLoyal)

get CityLoyal() //城池民心的实现

set CityBelongFactionID( cityBelongFactionID)

get CityBelongFactionID() //城池所属势力ID的实现

势力类属性的定义及实现

class Faction{

constructor(FactionID,FactionName,FactionLeaderPersonID,FactionCapitalID){

this.FactionID=FactionID;

this.FactionName=FactionName; this.FactionLeaderPersonID=FactionLeaderPersonID;

this.FactionCapitalID=FactionCapitalID;

}

}//势力类属性的定义

set FactionID( FactionID)

get FactionID() //势力ID的实现

set FactionName(FactionName)

get FactionName() //势力名字的实现

setFactionLeaderPersonID( FactionLeaderPersonID)

get FactionLeaderPersonID() 势力领导者ID的实现

set FactionCapitalID( FactionCapitalID)

get FactionCapitalID()//所属势力首都ID的实现

武将类属性的定义及实现

class Persons{

constructor(personID,personName,personBirthYear,personCommand,personMilitary,personPolitics,personTrick,personLoyal,personCityID,personState){

this.personID=personID;

this.personName=personName;

this.personBirthYear=personBirthYear;

this.personCommand=personCommand;

this.personMilitary=personMilitary;

this.personPolitics=personPolitics;

this.personTrick=personTrick;

this.personLoyal=personLoyal;

this.personCityID=personCityID;

this.personState=personState;

}

}

set PersonID( personID)

get PersonID() //武将ID的实现

set PersonName( personName)

this.personName = personName;

get PersonName() //武将名字的实现

set PersonBirthYear( personBirthYear)

get PersonBirthYear() //武将出生年份的实现

set PersonCommand( personCommand)

get PersonCommand() //武将命令的实现

set PersonMilitary( personMilitary)

get PersonMilitary() //武将命令的实现

set PersonPolitics( personPolitics)

get PersonPolitics() //武将政治的实现

set PersonTrick( personTrick)

get PersonTrick() //武将智力的实现

set PersonLoyal( personLoyal)

get PersonLoyal() //武将忠诚的实现

set PersonCityID( personCityID)

get PersonCityID() //武将城池ID的实现

set PersonState( personState)

get PersonState() //武将状态的实现

对战类属性的定义及实现

class Battle{

export default class Battle

constructor(BattleID,BattleName,BattleAttackArmyID,BattleAttackCityName,BattleDefendArmyID,BattleFieldCityID,BattleResult){

this.BattleID=BattleID;

this.BattleName=BattleName;

this.BattleAttackArmyID=BattleAttackArmyID;

this.BattleDefendArmyID=BattleDefendArmyID;

this.BattleFieldCityID=BattleFieldCityID;

this.BattleResult=BattleResult; this.BattleAttackCityName=BattleAttackCityName;

}

}//对战类实现

set BattleID( BattleID)

get BattleID()//对战ID的实现

set BattleName( BattleName)

get BattleName()//对战名字的实现

set BattleAttackArmyID( BattleAttackArmyID)

get BattleAttackArmyID()//对战进攻方ID的实现

set BattleDefendArmyID( BattleDefendArmyID)

get BattleDefendArmyID()//对战防守方ID的实现

set BattleFieldCityID( BattleFieldCityID)

get BattleFieldCityID()//对战胜利方ID的实现

set BattleResult( BattleResult)

get BattleResult()//对战结果的实现

set BattleAttackCityName(BattleAttackCityName)

get BattleAttackCityName(BattleAttackCityName)

//对战进攻方城市ID的实现

时间类属性的定义及实现

class Time {

constructor(year,season){

this.year=year;

this.season=season;

}

}

set Year( year)

get Year() //年份的实现

set Season( season)

get Season()//季节的实现

玩家类属性的定义及实现

class Citys {

constructor(playerID,playerScore){

this.playerID=playerID;

this.playerScore=playerScore;

}

}

set PlayerID( playerID)

get PlayerID()//玩家ID的实现

set PlayerScore( playerScore)

get PlayerScore()//玩家成绩的实现

# 2类函数实现

军队类函数的实现

armyChangeMainID( number)//主将修改

armyChangeAssisiant1ID( number)//副将1修改

armyChangeAssisiant2ID( number)//副将2修改

armyChangeSoldier( number)//士兵修改

armyChangeFood( number)//粮草修改

armyChangeMoney( number)//金钱修改

城池类函数的实现

cityChangeFood( number)//粮草改变

cityChangeFarm( number)//农业改变

cityChangeMoney(number)//金钱改变

cityChangeBusiness( number)//商业改变

cityChangeDefense( number)//城防改变

cityChangeSoldier( number)//士兵数改变

cityChangeLoyal( number)//民忠改变

cityChangeBelongFactionID( number)//所属势力改变

势力类函数的实现

FactionChangeLeaderPersonID( number)势力主公更换

FactionChangeCapitalID( number)//势力首都城池更换

武将类函数的实现

personChangeLoyal( number)//武将忠诚改变

PersonChangeCityID( cityID)//所在城池改变

personChangeState(state)//状态改变

对战类函数的实现

BattleNameGeneration(BattleAttackCityName)//战役名称生成

BattleResultGeneration(BattleAttackArmyID,BattleDefendArmyID, BattleFieldCityID)//战役结果生成

时间类函数的实现

timeSeasonChange( season)//季节变化

timeYearChange( year)//新年快乐

玩家类函数的实现

PlayerChangeMaxScore( MaxScore)//玩家最高成绩改变