用 MIDP 2.0 GAME API 对手机游戏进行优化开发

韩 冬[1,3],李 炜[2]
(1.同济大学 软件学院,上海 201804;
2.安徽大学 计算机科学与技术学院,安徽 合肥 230039;
3.安徽电子信息职业技术学院 计算机系,安徽 蚌埠 233030)

摘 要:文中主要研究了J2ME开发技术中MIDP 2.0游戏开发包的使用和J2ME手机游戏的优化方法与技巧,对上海美通公司提供的五款不完善的手机游戏进行了优化和改进。原游戏包一味使用面向对象技术,使用了过多的类嵌套,这样就不可避免地造成类膨胀,产生大量的、无用的重复代码,增加了系统开销,大大降低了程序的效率。文中采取的主要优化策略是尽可能地使用有限的面向对象的方法,尽可能地使用结构化编程的方法,移除不必要的继承关系、多余的类和接口,查找游戏的瓶颈,对游戏程序使用混淆器,以减少系统的开销,提高游戏的运行性能。经过测试,证明采取的优化策略是完全可行的。

关键词:手机游戏;移动开发;MIDP 2.0;J2ME;优化
中图分类号:TP311.52 **文献标识码**:A **文章编号**:1673-629X(2009)01-0180-04

Develop and Optimize J2ME Mobile Game with MIDP 2.0 GAME API

HAN Dong[1,3], LI Wei[2]
(1. School of Software Engineering, Tongji University, Shanghai 201804, China;
2. Computer Science and Technology College, Anhui University, Hefei 230039, China;
3. Department of Computer Science, Anhui Vocational College of Electronics & Information Technology, Bengbu 233030, China)

Abstract: Mainly explores how to develop J2ME mobile telephone games with MIDP 2.0 GAME API, and how to optimize J2ME mobile telephone games. In the meantime, five existing industrial mobile telephone games provided by Mtone Co. Ltd. have been optimized and improved. Object-oriented technology, such as class nesting, is used improperly in original games. It inevitably leads to class expansions and large numbers of useless repetitious code, and penalties to the performance. The primary optimization strategies applied are limiting the usage of object-oriented programming and instead using structured programming where possible, removing unnecessary inheritance, redundant classes and interfaces, tackling of the performance bottle-neck and obfuscating programs. With all these optimization strategies improve the performance of these significantly and reduce greatly the size of the jar file. It shows that the game optimization techniques produced and discussed in this paper are really helpful to improve the performance of the mobile games, which could be further used in the future mobile game development.

Key words: mobile telephone game; mobile development; MIDP 2.0; J2ME; optimization

0 引 言

移动通讯发展到现在已经逐步进入了一个黄金时期,手机网上游戏将成为移动互联网的主流应用,其中J2ME成为手机游戏开发的主流平台。由于硬件设备的限制,J2ME的移动开发比J2SE和J2EE的开发更加困难。内存、CPU、输入输出等诸多方面均受限制的手机,要求设计人员在开发时必须仔细地评估自己所需要的性能,彻底地优化代码,才能保证程序的有效运行。因此,优化技术在J2ME的移动应用开发中占有举足轻重的地位。

收稿日期:2008-05-05
基金项目:安徽省自然科学研究项目(KJ2008B097)
作者简介:韩 冬(1969-),男,安徽蚌埠人,副教授,硕士,研究方向为大型数据库系统和移动开发;李 炜,副教授,博士,研究方向为软件形式化方法和软件工程。

1 使用 MIDP 2.0 游戏开发包

MIDP 2.0是由JCP制订的,该规范的制订是由大约50个公司共同参与设计。MIDP 2.0规范设计的目的是定义一个新体系结构以及相应的API,从而为第

三方的移动信息设备应用的开发提供一个开放的标准环境。在 MIDP 1.0 规范中,并未提供专门面向游戏开发的功能。开发人员不得不自己编写如精灵(Sprite)、分块图层(TiledLayer)等游戏专用类,这使得游戏开发的复杂度增高,并且容易影响游戏性能。MIDP 2.0 的一个显著变化就是加入了一组用于游戏开发的类库[1],即 GAME API,这组简洁的 API 放在 javax.microedition.lcdui.game 包中(中文译作 MIDP 2.0 游戏开发包)。MIDP 2.0 游戏开发包并不是一个完整的游戏引擎,事实上它除了必要的游戏对象封装外并没有提供任何和游戏逻辑有关的封装。MIDP 2.0 游戏开发包的提出是为了在底层消减游戏复杂性,将很多游戏的基本元素由系统 API 来完成,尽量减少 Java 代码调用带来的额外的性能损失。因为系统 API 可以在底层直接委托给本地代码来执行,而本地代码可以由各个设备厂商进行充分的优化。

这个新的游戏开发包提供了 5 个新类:GameCanvas、Layer、LayerManager、Sprite 和 TiledLayer。使用这些类的基本思想是:游戏界面由图层组成,也即背景和游戏人物分布在不同的图层上,而每一个图层都可以分别通过程序控制,功能强大的图层能够帮助开发者高效地建立复杂的场景。其中,GameCanvas 类是 Canvas 类的派生类,它在 Canvas 类的基础上增强了游戏开发所需的功能,如脱机屏幕缓冲、直接按键查询等。该类提供了一个基本的游戏屏幕[2],在它的基础上可以很方便地进行游戏图形界面的设计,实现与用户的交互。TiledLayer 类和 Sprite 类都是从抽象类 Lay 类中派生的,主要用于实现游戏中的活动主体和动画背景等。而 LayerManager 类则可以管理游戏中的多个 Layer 类,实现前景和背景的混合输出,非常方便地实现用户视角的变换。

2　J2ME 手机 RPG 游戏的设计与实现

2.1　游戏的策划

欲优化的 J2ME 手机游戏是五款精巧的 RPG 游戏[3],游戏的名称分别是 Dig、Hunt、Feeding、Kongfu、Climb,Dig 和 Hunt 是以斜 45 度地图为背景的 2.5D 滚屏游戏,其余三款则是 2D 的单屏游戏。Dig 描述的是一个小男孩挖掘宝物,宝物有宝箱、元宝和金币,挖掘到的宝物不同,得分也不同;赢家是规定时间内的得分最高者。Dig 游戏的运行界面如图 1 所示。

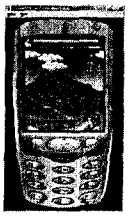

图 1　Dig 游戏

2.2　游戏的架构

考虑到以后的可维护性、扩展性、软件重用和开发的方便性,游戏设计了引擎,分为若干模块,有公用模块、媒体处理模块、地图及场景模块、人物模块、消息处理模块、场景绘制模块、网络传送模块等。游戏除了包含 engine 包以外,还包含 dig 包、hunt 包、climb 包、feeding 包、kongfu 包、sanjie 包、guessdual 包等。Dig 游戏的类图如图 2 所示。

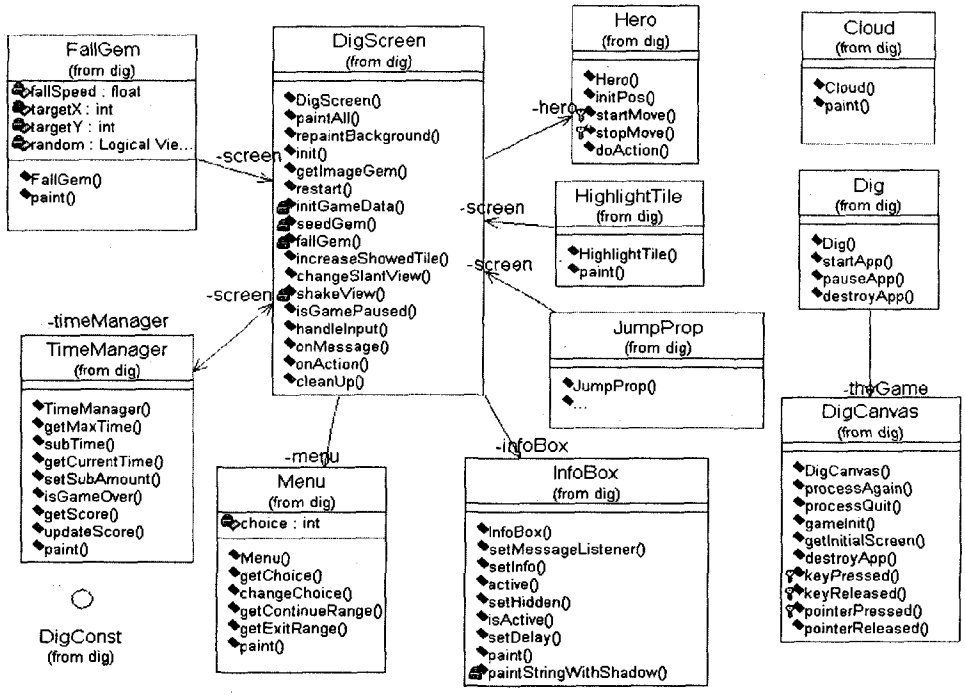

图 2　Dig 的类图

2.3　游戏的实现

游戏的线程主要由 3 个部分组成:检测键盘输入(与玩家交互)、更新游戏场景(处理游戏逻辑)和绘制游戏画布。以 Dig 游戏为例,首先有继承 MIDlet 的 Dig 类,在其中的构造函数 Dig() 中生成了 DigCanvas 的实例。DigCanvas(游戏画布)是 PlayCanvas 的子类,

而 PlayCanvas 是 GameCanvas 的子类。游戏画布可以看作是一个游戏控制器,负责等待并获得用户的输入,进行相应的处理(改变游戏的状态和移动图层),绘制改变后的游戏界面。一般情况下,实现线程接口(Runnable),在该类的 run 方法中实现游戏主循环。其他几个游戏的实现也与此类似。

3 J2ME 手机 RPG 游戏的优化与改进

3.1 游戏中存在的缺陷

原游戏虽精巧,但存在很多缺陷。首先,游戏不能够健壮地运行。进一步仔细阅读原手机游戏的程序代码,发现还存在以下的缺陷:过多的类嵌套;不必要的继承关系;多余的、不需要的成员函数;可以精简的接口;大量的、重复的冗余代码。这些造成了手机游戏的运行性能降低,具体表现在运行速度慢和占用较多的内存,故有必要对其进行优化。

3.2 具体的优化措施

3.2.1 类和接口的精简

采取以下方法对类和接口进行精简。

* 移除常量接口。
* 移除多余的、不必要的继承关系。
* 移除多余的、不必要的类和接口。
* 移除多余的、不必要的函数。

精简类和接口后得到的 Dig 类图,如图 3 所示。

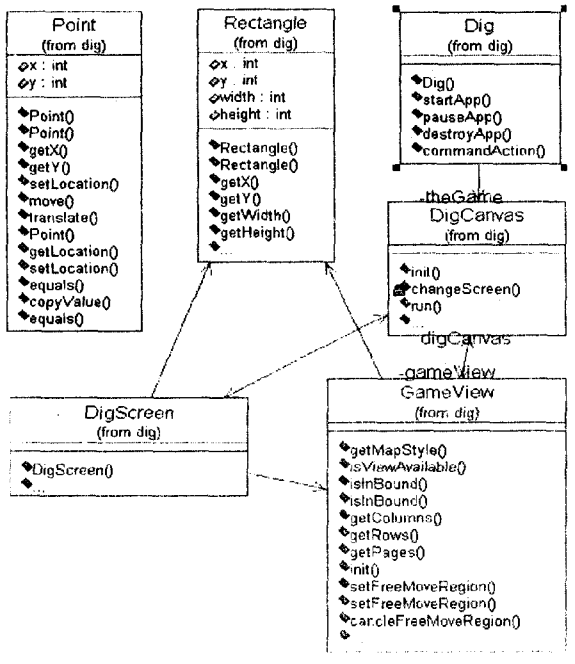

图 3 精简类和接口后得到的 Dig 类图

3.2.2 进一步的优化和改进

采取以下方法对手机游戏进行进一步的优化和改进[4]。

* 查找游戏瓶颈[5]。
* 尽可能地把类中的成员变量改为局部变量。
* 尽可能地把类中的成员变量定义为 static 和 final static。
* 尽可能地把类中的成员方法定义为 final 和 final static。
* 使用 Sprite 类。尽量使用数组,少用 Vector 类。合并多余的函数。
* 不用的对象赋值为 null。少用 String,多用 StringBuffer。
* 使用混淆器[6](使用 ProgGuard 3.7 beta2 自动检测和删除没有用到的类、方法和数据成员,去掉包的层次关系,混淆保护代码,最大限度地缩小 JAR 文件的大小)。

4 实验结果

运行优化前的原手机游戏程序,记录消耗的系统开销(即装载游戏的时间和游戏正常运行时占用的内存);同样,运行优化后的手机游戏程序,记录消耗的系统开销。所得结果,如表 1 和表 2 所示。运行环境为:Windows 2000 Server、Service Pack 3、J2ME Wireless Toolkit 2.2、256M 内存、80G 硬盘容量、x86 Family 6 Model 8 Stepping 1 AuthenticAMD~1499MHz。

由表 1,优化后游戏装载时间分别比原来减少了 22.73%、34.62%、25.79%、24.65%、16.70%。由表 2,优化后游戏运行时占用的内存分别比原来减少了 17.56%、16.30%、4.80%、5.85%、7.51%。表 3 是优化前后的 JAR 文件大小的比较,优化后的手机游戏 JAR 文件的大小比优化前减少了 821kB,减少的百分比为 71%。表 4 是优化前后总的代码行数(LOC)的比较,优化后的手机游戏总代码行数与优化前相比减少了 12729 行,减少的百分比为 59%。

表 3 优化前后的 JAR 文件大小比较

	优化前	优化后	减少的尺寸	减少的百分比
JAR 文件大小	1.12MB	326kB	821kB	71%

表 4 优化前后总的代码行数(LOC)的比较

	优化前的 LOC	优化后的 LOC	减少的 LOC	减少的百分比
总的代码行数	21597	8868	12729	59%

通过对以上实验数据的分析,笔者认为优化是成功的。也证实文中提出的关于 J2ME 手机游戏性能优化的方法是切实可行的。

5 结束语

原游戏包一味使用面向对象技术,对于各个类之

表 1　优化前后游戏运行速度的对比

游戏名	优化前后	游戏的装载时间(s)				
		1	2	3	Average	Improvements
Dig	Before	79.37	79.86	79.28	79.50	22.73%
	After	61.10	61.84	61.36	61.43	
Hunt	Before	6.17	5.68	6.77	6.21	34.62%
	After	4.15	3.64	4.39	4.06	
Climb	Before	4.70	5.19	5.23	5.04	25.79%
	After	3.89	3.22	4.10	3.74	
Feeding	Before	3.63	3.53	3.55	3.57	24.65%
	After	2.80	2.39	2.88	2.69	
Kongfu	Before	5.29	5.19	5.32	5.27	16.70%
	After	4.34	4.87	3.96	4.39	

表 2　优化前后游戏内存使用的对比

游戏名	优化前后	游戏从开始装载至正常运行所占用的最大内存(Bytes)				
		1	2	3	Average	Improvements
Dig	Before	263624	279436	264052	269037	17.56%
	After	220460	220764	224152	221792	
Hunt	Before	139296	145278	143662	142745	16.30%
	After	118800	120240	119398	119479	
Climb	Before	337612	333864	334523	335333	4.80%
	After	318975	321827	316958	319253	
Feeding	Before	345232	354532	354396	351387	5.85%
	After	328791	336948	326743	330827	
Kongfu	Before	350420	350000	333992	344781	7.51%
	After	310678	327263	318689	318877	

间的功能做了过于细致的划分,使用了过多的类嵌套,这样就不可避免地造成类膨胀,产生大量的、无用的重复代码,大大降低了程序的效率。当放弃原先的设计,采取尽可能地合并类、尽可能地减少类的封装和继承的优化方法后,提高了程序的运行效率。通过对手机游戏优化这一课题的研究,可以看到:不能沿用以前在 PC 平台开发时遵循的传统软件工程理论,来进行手机游戏的优化开发。手机游戏优化开发既需要借鉴 PC 游戏中的成功经验,更需要针对运行设备资源极为有限的特点,而采取有限的面向对象、尽可能地使用结构化编程的方法。

参考文献:

[1] 施　铮,王小彗,杜雪梅,等. J2ME 技术参考手册[M]. 北京:电子工业出版社,2004:312 - 327.

[2] J2ME homepage[EB/OL]. 2007 - 08 - 26. http://java.sun.com/javame/index.jsp.

[3] Zhang Weishan, Han Dong. Object - Orientation is Evil to Mobile Game: Experience from Industrial Mobile RPGs[C/OL]. The 2007 International Conference on Embedded Software and Systems (ICESS 2007). Lecture Notes in Computer Science (LNCS). 2007 - 06 - 30. http://www.springerlink.com/content/r57w642l1395265k/.

[4] Zhang Weishan, Han Dong, Kunz T. Mobile Game Development: Object - Orientation or Not[C]//31st Annual International Computer Software and Applications Conference (COMPSAC 2007). Beijing, China: IEEE Computer Society, 2007:601 - 608.

[5] 詹建飞. J2ME 开发精解[M]. 北京:电子工业出版社,2006:33 - 37.

[6] 李振鹏,龚　剑. J2ME 手机游戏开发技术详解[M]. 北京:清华大学出版社,2006:45 - 55.