“GluttonousSnake”程序文档

## 程序功能

带GUI的贪吃蛇游戏，亮点如下：

1. 程序完成度高，有一套完整的计分和游戏难度递增体系
2. GUI美观，画面友好，音效清新，CPU占用率低，实现了高帧率刷新
3. 玩家体验好，经过用户调研改进了不少功能
4. 速度可无极调控，避免枯燥乏味
5. 设置了ico图标，使用windeployqt工具完成了程序的发布，完成度高

## 程序用法

1. 在开始界面，点击“开始游戏”，游戏开始；点击“退出游戏”，游戏结束
2. 使用上下左右方向键来控制蛇的移动
3. 在游戏中，有3种食物，分别是：
   1. 普通食物：吃完+1分
   2. 大仓鼠：吃完+3分，在等级大于等于3时以20%概率出现和消失
   3. 药丸：吃完不加分，但是身体会缩短一半，在等级等于4时以10%概率出现和消失
4. 游戏会随着分数的增加而提升等级，也即游戏难度，难度越高，移动速度越快
5. 蛇可以穿墙，也就是说除了撞到自己身体之外都不会死掉
6. 游戏中按ESC键可以暂停游戏或者退出游戏

## 设计思路



首先介绍两个核心的数据结构：

一是map，一个bool型的二维数组，当蛇身覆盖了该区域（坐标）即为true，否则为false。这个数据结构为防止食物和道具时出现在蛇身上提供了基础。

二是body，一个Point型的一维数组。Point是自己定义的类，存储点的横纵坐标。因为蛇头具有特殊的地位，它是方向键直接控制的对象所以单独设立一个Point型的head变量。也分离出tail，因为它和head都涉及到蛇改变移动方向时图片的旋转（矩阵旋转）。为了方便后续操作，还为Point重载了运算符==和!=。这个数据结构为蛇自身的碰撞检测、蛇的移动、蛇吃食物与道具提供了基础。

在自动移动和画面刷新方面，我们采用了Qt中的Qtimer定时器。该程序总共有两个定时器：

一是重绘定时器KeyPressTimer，为了保证FPS达到60，我们设置为每隔16ms发出一个timeout信号，经过connect函数的连接，被内建的update()函数接收，它随后自动调用经过重写paintEvent()函数，实现所有部件的重绘。

二是自动移动定时器AutoMoveTimer。我们设定为每隔150ms发出一个timeout信号，经过connect函数的连接，被自定义的AutoMove()函数接收。该函数依照现有的方向调用对应的UpMove,DownMove,RightMove,LeftMove四个单步移动函数，进行蛇身的移动和map数组的刷新。

在键盘事件监听方面，我们重写了keyPressEvent函数，无阻塞地监听四个方向键和escape键。检测到方向键被按下后，keyPressEvent调用对应的四个单步移动函数。

在等级系统方面，我们将等级和自动移动速度直接挂钩。即等级越高，自动移动定时器发出timeout信号的频率就高，达到蛇身移动速度加快的效果。

在食物与道具系统方面，食物包括普通食物和大仓鼠，道具包括药丸。三者的实现方式大致相同，以最复杂的药丸举例：与其相关的变量有：

1. bool型的ExistItemFood 变量记录药丸是否存在。
2. Point型的ItemFood变量记录药丸的位置。

函数有：

1. 返回值为bool型的CheckItemEat函数判断是否吃到了食物。
2. 返回值为bool型的ShowItemFood函数决定是否出现。
3. EatItemFood函数进行吃了药丸之后的操作。

具体机理如下：

1. 当多次调用时，ShowItemFood函数以10%的概率返回true。在paintEvent函数里，若其为true且满足等级条件，则使用DrawItemFood来绘制药丸。
2. CheckItemFood在每一次单步移动的时候都会被调用，它检查Point型的head是否和ItemFood相等，若相等则表明吃到了药丸，释放出EatSignal信号，返回值为true。此时执行EatItemFood函数，它使BodyCount减半。

在移动算法方面：

蛇身基本移动的逻辑是基于蛇头head的移动来完成的。我们通过监听键盘事件确定蛇头的下一步移动方向，然后修改蛇头的坐标，然后从尾至头后一个单元依次将蛇身每一个单元的位置信息修改为前面一个单元的位置信息，通俗来说就是每一节身体都向前移动到上一节身体位置。完成移动后，更新此时的方向状态direction。如果方向direction与当前的单步移动函数方向相反，则该函数将不执行。

实现穿墙功能的基本逻辑也是只关注蛇头的坐标，如果蛇头的横纵坐标出现了小于0或大于地图坐标边界值的情况，我们可以认为蛇进入了穿墙状态，此时只需要根据蛇此时的移动方向将其横坐标或纵坐标修改为穿墙后的值，接着重复之前“每一节身体都向前移动到上一节身体位置”的逻辑即可实现穿墙。

当蛇在移动过程中发生了自身碰撞时，即触发了CheckBodyCrush函数，为了保持游戏美观性，我们会将蛇头保持撞前的位置，通过将蛇头位置回退至原来位置实现，之后触发EndGame函数进入游戏结束流程。

在图形绘制方面：

在该程序中，我们主要使用了Qpixmap，Qicon，和Qpainter三个QT常用类来实现图片的绘制，文字说明部分则使用Qlabel类实现，在mainwindow中，我们将各个部件分别写入不同的draw函数中，然后再painterevent绘图事件中统一调用，完成主界面的图形绘制。

Qpushbutton的重写：

为了完成游戏开始界面的按钮美化，我们重写了一个自定义类Button继承QPushButton，并在该类中重写了mousePressEvent，mouseReleaseEvent，enterEvent，leaveEvent实现在鼠标移动至图标，鼠标点击，鼠标释放是按钮的动画效果，并使用Qpixmap和Qicon类完成按钮的图片显示。

Qdial控件实现无级变速：

通过QT中的Qdial类绘制出旋钮，当Qdial控件的value值发生变化时，ValueChanged函数发出信号，及各国connect函数的连接，槽函数speedchange接受信号，用Qdial的value值来修改AutoMoveTimer的值和level的值实现移动速度的等级的修改。

游戏失败后的重置：

我们通过RestGame函数重置初始方向direction变量，等级level变量，得分score变量，并将所有种类食物的存在标识重置为false，将Qdial控件的值回复到默认值30。之后重新调用CreateMap，CreateSnake，CreateFood，StartTimer函数实现地图，蛇，食物的重新生成和计时器的重置。

游戏音效：使用Qsound类;调用setloop函数实现BGM的循环播放。音效文件已经包装在Qt的资源文件qrc中，并且加了前缀和别名，在程序中直接使用“:/xxx/xxx”即可。因为QSound只支持wav格式的音频文件，所以游戏内的BGM,食物音效，游戏结束音效都是wav格式文件。

## 运行截图















