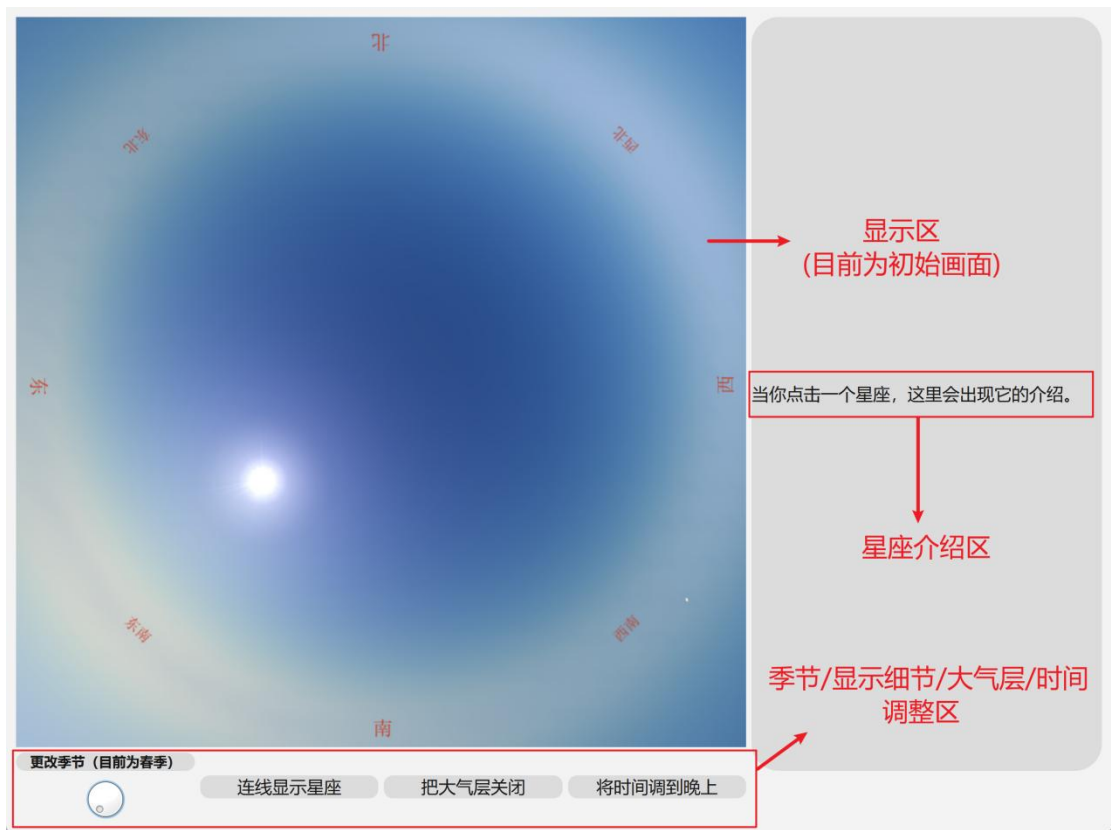


Qt 作业完成报告

何忻远 袁钲茗

1.程序功能介绍

本程序实现了观看星图以及星座信息快速查询的功能。如图：

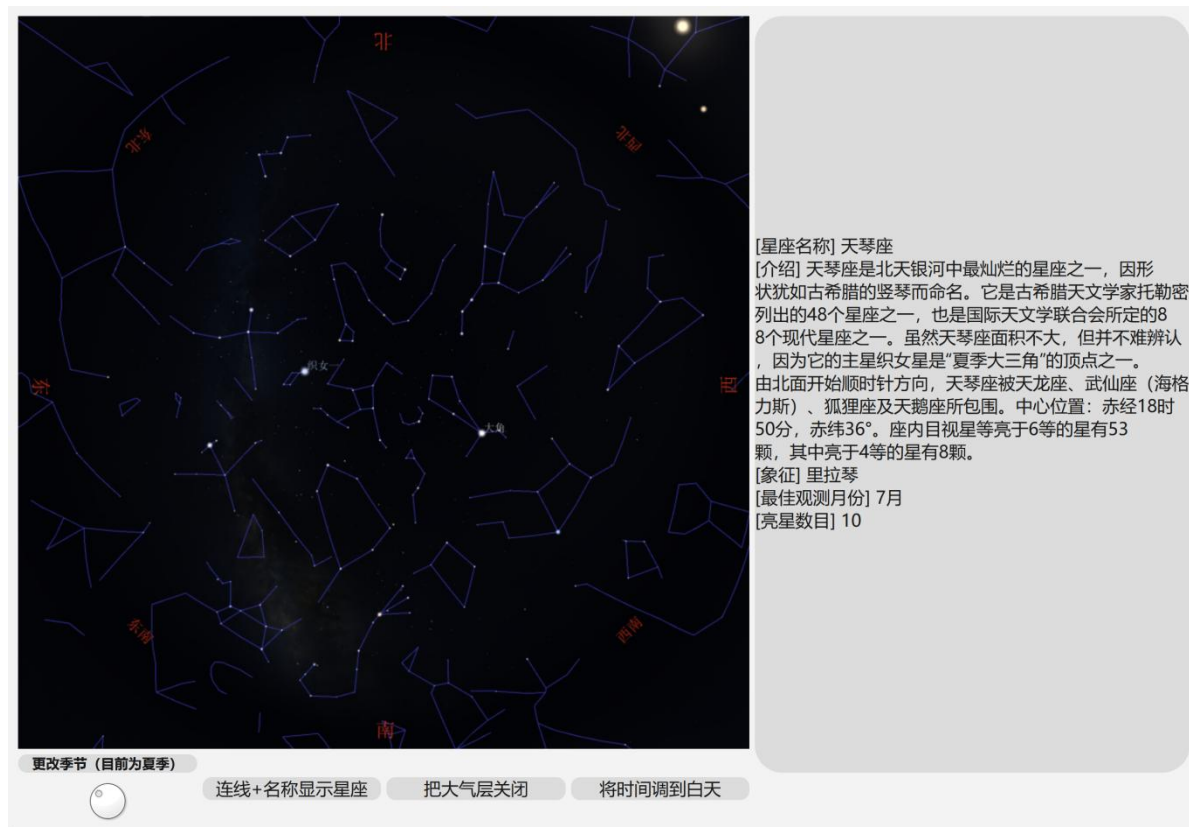


当调整“调整区”中的按钮时，显示区的内容会跟着改变。

预设的观测地点在北大，并且提供了：

- 春夏秋冬四个季节的观测数据（时间分别为 2024 年的 4 月 1 日、7 月 1 日、10 月 1 日、1 月 1 日）
- 大气层的开/关两种形态
- 时间的白天（上午 10 点）和晚上（晚上 22 点）
- 星座的显示细节调节（不连线显示、连线显示、连线+名称显示）。

当大气层打开的时候，大气层的调整按钮会显示“把大气层关闭”，点一下就可以把大气层关闭，反之亦然。白天，大气层打开时，显示白天所见的天空；关闭则显示白天无大气层时星星的位置（由于我们生活在有大气层的环境，所以无法肉眼观测到这种情况）。晚上，大气层关闭时，显示的星星会更亮，但是也无法显示星座的连线和名称。这样设计是因为关闭大气层时，观测到的星星更亮更多，不方便认识星座。因此，只有在打开大气层的晚上，才可以正常地调整“显示细节”（即：显示星座的连线，或者连线+名称）。



在打开大气层的晚上，点击星图中的星星及其附近的位置，就可以看到这颗星星所属星座的介绍了。介绍包括了：

- 星座名称
- 星座介绍（概览、一些会有命名故事、黄道十二星座(eg.双鱼座)会有代表日期）
- 象征
- 最佳观测月份
- 亮星数目

与此同时，我们埋了两个彩蛋，期待使用者的探索。它们的触发方式在本文档的最后一页，供使用者参考。

2.项目模块与类的设计

项目分为 4 个模块，分别如下：

- ▼ 头文件
 - addConstellation.h
 - ClickableLabel.h
 - ImageSwitcher.h
 - widget.h

- addConstellation.h ——初始化星座介绍、初始化星座的位置、得到鼠标所点击位置的星座介绍。

- ClickableLabel.h ——处理鼠标在星图上的点击位置，返回 x,y 坐标。
- ImageSwitcher.h ——切换图片
- widget.h ——主界面以及各个按钮的点击事件

以下是程序中主要类和函数的作用。

addConstellation.h { class Constellation → 单个星座的信息
class StarPosition → 单个星座在某一时间下的位置

ClickableLabel.h → class ClickableLabel → 获取点击位置

ImageSwitcher.h → class ImageSwitcher → 切换图片

widget.h → class Widget → 窗口

总览

```
class Constellation { //单个星座的信息
public:
    QString name;
    QString intro;
    QString symbol;
    short stars;
    short best_month;
    int R[4]; // Rectangle: 左上x, 左上y, 右下x, 右下y 目前的作用已经被StarPosition中的R所取代
    QVector<QPair<int, int>> star_pos;

    Constellation(QString name, QString intro, QString symbol, short stars, short best_month) : //初始化单个星座的信息
    | name(name), intro(intro), symbol(symbol), stars(stars), best_month(best_month) {}

    QString getConstellationInfo(const QString comp_name) const;
    //比对传入的comp_name参数与里面的name是否相同 如果相同就返回星座信息 否则返回空字符串
};
```

Class Constellation

```
void initializeConstellation(const QString& fileString, QVector<Constellation>& CTS);
//从文件地址 fileString 初始化所有星座的信息

void initializePosition(StarPosition pos, QVector<Constellation>& CTS);
// 由单个星座在某一时间下的位置 来增补 单个星座的信息 没用

void initializeMonthlyPosition(const QString& fileString, QVector<StarPosition>& STP);
//从文件地址 fileString 初始化所有星座在某一时间下的位置

QVector<QString> getInfos(const int x, const int y,
                           const QVector<StarPosition>& STP, const QVector<Constellation>& CTS);
//由x,y坐标得到当前鼠标所点击位置的星座介绍
```

addConstellation.h 中的函数及其作用

```
class StarPosition{ //单个星座在某一时间下的位置
public:
    QString name;
    QVector<QPair<int, int>> pos;
    int R[4]; // Rectangle: 左上x, 左上y, 右下x, 右下y
    StarPosition(QString name, QVector<QPair<int, int>> pos);
};
```

Class StarPosition

解释: R 记录了可以围住星座的最小矩形

```

class ClickableLabel : public QLabel {
    Q_OBJECT

public:
    explicit ClickableLabel(QWidget *parent = nullptr);

signals:
    void clicked(QPoint pos);
    void constellationClicked(const QString &name);

protected:
    void mousePressEvent(QMouseEvent *event) override;
};

```

Class ClickableLabel

```

class ImageSwitcher : public QWidget {
    Q_OBJECT

public:
    ImageSwitcher(QWidget *parent = nullptr, QStackedWidget* UIstack = nullptr);

    // 初始化图片
    void initializeImages(const QVector<QString> &imagePaths);

    // 切换到下一张图片，带透明度变化（实现失败）
    void switchToNextImage();

    // 切换到指定图片
    void switchToAppointedImage(QString str);

signals:
    void constellationClicked(const int x, const int y);

private:
    QStackedWidget *stackedWidget;
    QVector<QLabel*> labels;
    int currentIndex;
    QMap<QString, int> maploader;
    QMap<QString, QRect> constellationRegions;
    // 创建透明度动画（实现失败）
    QPropertyAnimation* createFadeAnimation(QWidget *widget, qreal startValue, qreal endValue, int duration);

    // 增添：显示星座介绍（已经换成新的实现方法）
    void showConstellationInfo(const QString &constellationName);

private slots:
    // 切换完成后调用的槽函数
    void onFadeOutFinished();
    void handleLabelClicked(const QPoint &pos);
};

```

Class ImageSwitcher

解释：有用的函数和变量只有红色荧光笔画出的部分，剩下的都是废稿。

```

private slots:
    void on_dial_valueChanged(int value); 季节变换

    void on_atmosphereButton_clicked();    大气层开关

    void on_detailsButton_clicked();       细节调整

    void on_timeButton_clicked();          时间调整

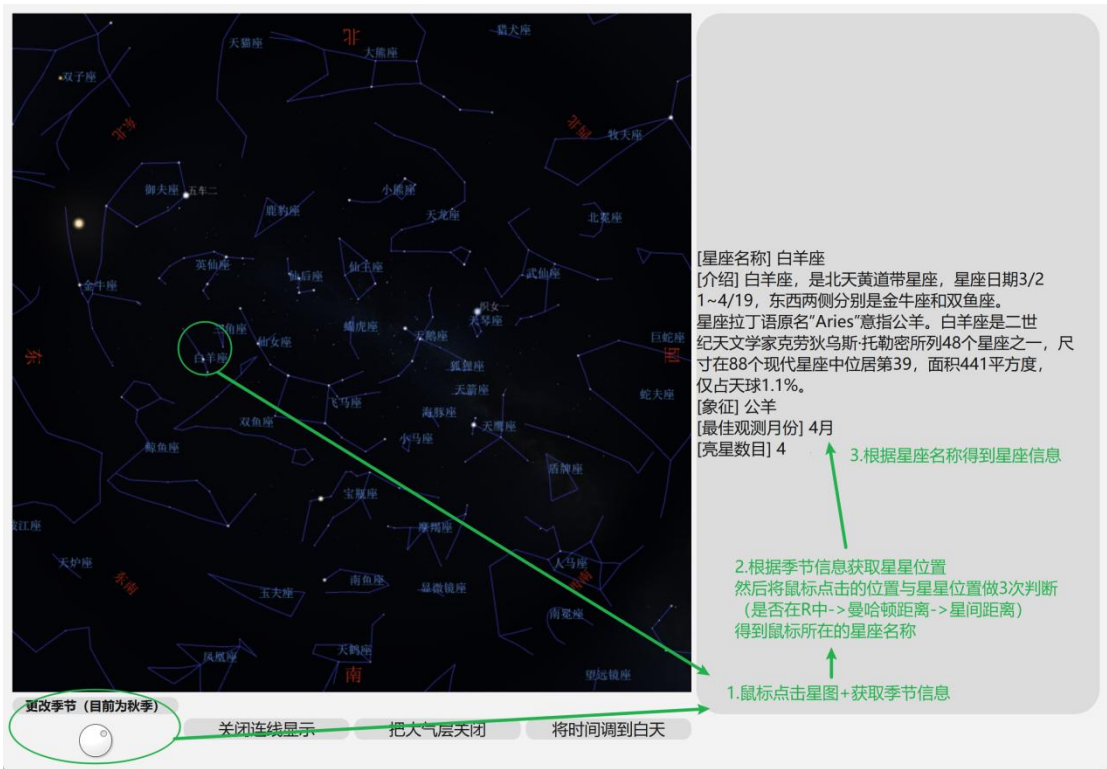
    void showConstellationInfo(const int x, const int y);|
    // 作用：传入x,y坐标，返回星座介绍

private:
    Ui::Widget *ui;
    int season; // 季节
    bool atmosphere; // 是否打开大气层
    int details; // 显示的详细程度
    int time; // 时间
    ImageSwitcher* imageSwitcher;
    QString getPicName();
    QLabel *infoLabel; // 增添
    QVector<ClickableLabel*> labels; // 增添
    QString file_place; // 文件存放的地址 主要是图片, 文字...
    QVector<Constellation> CTS; // 星座 狠狠地标记!
    QVector<StarPosition> STP[4]; // 4-7-10-1月的星星位置!
    QString introFilePath; // 介绍文档的路径
    QString starposFilePath[4]; // 星星位置文档的路径 分别为4月-7月-10月-1月

```

Widget.h 中的变量和函数及其作用

以下为实现星座位置判断的具体方法。



其中第 1 步使用到 ClickableLable.h 以获取点击的位置。季节信息的获取直接读取 widget.h 中的 season 变量。

第 2 步和第 3 步则调用了 widget.h 中的 showConstellationInfo 函数，它会间接的调用 addConstellation.h 中的 getInfos 函数，getInfos 函数会返回一个 QString 数组，里面装着可能的返回的星座内容。然后 widget.h 中的 showConstellationInfo 函数负责将其投射到屏幕上。

以下为星座介绍和星座位置的初始化逻辑。

- Intro.txt ---initializeConstellation 函数--->储存于 QVector<Constellation> CTS 中
 - 其中 CTS 的每一个元素都是一个星座的介绍内容
 - intro.txt 格式如下:

```
name: 天鹅座
intro: 天鹅座是北天星座之一, ..... (内容略) 这颗星是夏季大三角的顶点之一。
symbol: 天鹅
stars: 6
best_month: 9 月

name: 仙女座
intro: 仙女座是全天 88 个现代星座之一, .....也是最小星座南十字座面积的十倍以上。
symbol: 仙女
```


stars: 16

best_month: 11 月

• mmdd-2200.txt(mmdd = 0101, 0401, 0701, 1001)---initializeMonthlyPosition 函数
--->储存于 QVector<StarPosition> STP[4]中

- 其中 STP[i]中的每一个元素是第 i 个季节中某一个星座的位置（点）的集合
- mmdd-2200.txt 格式如下：

后发座： 21,133

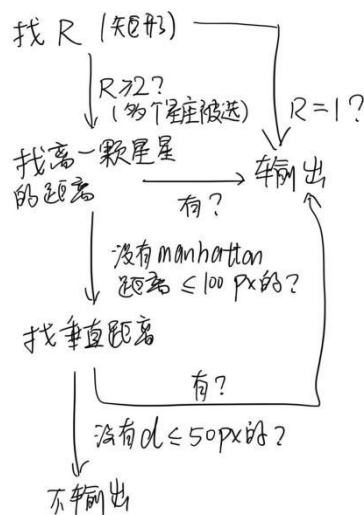
190,151

213,302

猎犬座： 354,237

394,303

以下为判断该选哪个星座的逻辑。



3.小组成员分工情况

何忻远（组长）：

实现了文档内容“intro.txt”中对星座内容的介绍和把 .txt 文件内容读取到程序内的方法（addConstellation.h 中）；

优化了星座信息的显示（将其从星图正上方移至星图的右侧）；

实现了“由星座位置找到星座名称，并以此找到对应的星座信息”的方法；

设计并美化了 UI，加入了彩蛋；

实现了 widget 中的所有控件、槽函数及变量；

完成作业报告，录制演示视频。

袁钰茗：

完成了对图像（mmdd-2200.txt）及鼠标等的信息收集；

实现了从点击图像到显示文字的信息对接；

上传文档。

4.项目总结与反思

在完成此项目的过程中，我组成员了解了 Qt 的基本性质及其应用；同时学到了制作以及美化 UI 界面的一些技巧，使用类来统筹、管理存储的方法。

与此同时我们发现：在分工过程中也存在一些不合理的任务分配；Qt 一些复杂的特性导致某些简单的功能难以实现，比如我们放弃了星图切换时的渐变显示。

通过这次 qt 大作业，我们获得了许多宝贵的经验，看似简单的功能背后其实也需要相当的代码支撑，这也导致我们没能实现原本预想中的部分功能。但总的来说，通过团队项目的开发，我们不仅提升了自身的技术能力，掌握了部分 qt 相关知识，也学会了如何更好地协作和沟通。我相信这些经验将对我们未来的发展产生积极影响。

5.借物表&彩蛋

intro.txt 中的星座介绍部分来自于维基百科，感谢它，真的很详细！

星图来自于 Stellarium。这是一款非常方便星空观测软件，完全免费。我们的星图就是在这个软件里截的图。没有它的图例支持，我们几乎不可能完成这个项目。



彩蛋：

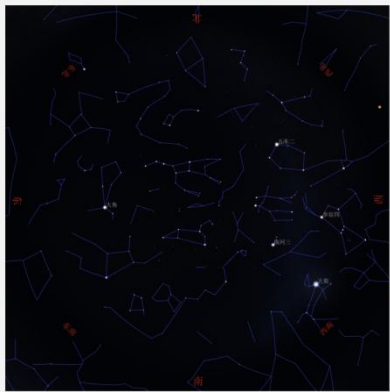
1.之前提到：只有在打开大气层的晚上，才可以正常地调整“显示细节”（即：显示星座的连线，或者连线+名称）。如果你不在这个前提条件下，连按 10 次显示细节调整的按钮，星座介绍区会出现：

只有在打开大气层的晚上，才可以显示星座的连线和名称哦~

的提示。

2.深色模式

看这个空出来的位置（绿色箭头），点一下进入深色模式



[星座名称] 麒麟座

[介绍] 麒麟座又名独角兽座（希腊: Μιωνόξ πύκν）是在天球赤道上的一个偏淡星座，它的名字在希腊的意思是独角兽。

它是由17世纪的荷兰制图员普朗修斯（Petrus Plancius）所创建的星座。与它接壤的星座在西北是蝎户座，北边是双子座，南方是犬夫座和长蛇座的方向。与它接壤的星座还有小犬座、天兔座和船尾座。日本语的麒麟座(きりん)，则是指中文的鹿豹座(Camelopardalis)。

[象征] 麒麟

[最佳观测月份] 2月

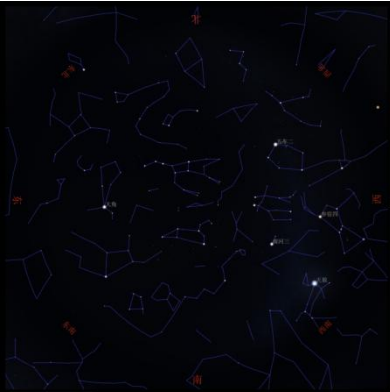
[亮星数目] 5

更改季节 (当前为春季)

连线+名称显示星座

把大气层关闭

将时间调到白天



恭喜!你找到了切换到深色模式的方法!

更改季节 (当前为春季)

连线+名称显示星座

把大气层关闭

将时间调到白天