

不围棋stage1实验报告——team10

大体分工：

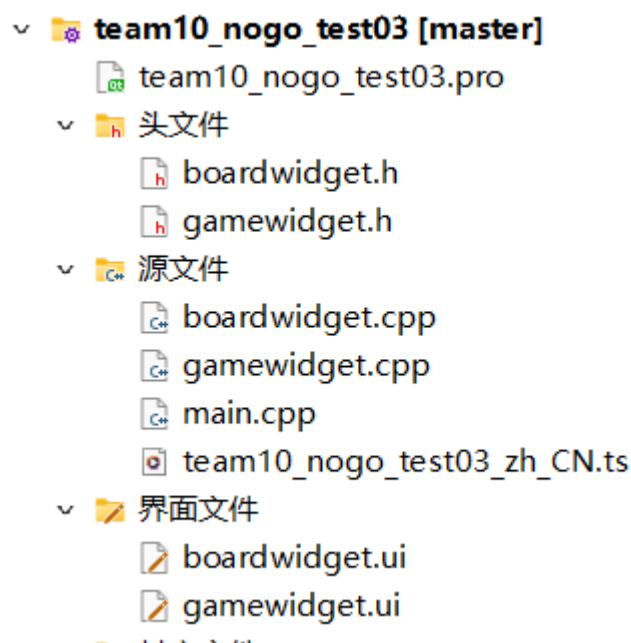
董毅同学负责围棋界面的设计以及先后落黑白子的逻辑实现

祁文轩同学负责判断胜负以及不围棋规则的逻辑实现

冯悦同学负责计时与重新开始的逻辑实现，修改了开始界面实现换棋盘大小。

概述：

本组当前已实现棋盘大小调整功能，另有鼠标跟随事件的功能，可以让棋手清楚的看清自己下一步棋将落于哪个网格。各代码功能已有详细的注释，本部分报告展示部分代码。以下为实现框架（后期可能更改）



gamewidget负责初始化游戏状态，选择棋盘大小，重新开始游戏以及后续内部逻辑的实现。

包含：选择棋盘大小按钮 重新开始按钮

选择棋盘后弹出boardwidget界面

boardwidget负责棋盘与棋子界面、先后落子以及鼠标移动事件，计时功能的实现

同时实现吃子，自杀，判断胜负功能

包含：棋盘 计时器 更改时间按钮

代码介绍：

gamewidget中

主要函数及其作用

```
42
43 ▶ void GameWidget::initWidget()//打开围棋界面，在onSizeButtonxClicked()中被调用 {...}
62
63
64 ▶ void GameWidget::newGame()//在当前棋盘大小下重新开始游戏 {...}
68 ▶ void GameWidget::onSizeButton9Clicked()//打开9*9棋盘 {...}
74 ▶ void GameWidget::onSizeButton11Clicked()//打开11*11棋盘 {...}
80 ▶ void GameWidget::onSizeButton13Clicked()//打开13*13棋盘 {...}
86 ▶ void GameWidget::restartGame()//不同于newGame函数，此函数可重新选择新的棋盘大小 {...}
98
```

构造函数 (gamewidget界面实现)

```
GameWidget::GameWidget(QWidget *parent)
: QWidget(parent)
, ui(new Ui::GameWidget)
{
    setWindowTitle("NoGo");
    ui->setupUi(this);
    // 创建 "Begin" 按钮
    QPushButton *beginButton = new QPushButton("Begin 9*9", this);
    beginButton->setGeometry(QRect(10, 10, 150, 30));
    //调整棋盘大小
    QPushButton *sizeButton11 = new QPushButton("11*11", this);
    sizeButton11->setGeometry(QRect(10, 50, 100, 30));
    QPushButton *sizeButton13 = new QPushButton("13*13", this);
    sizeButton13->setGeometry(QRect(10, 90, 100, 30));
    //新游戏和改棋盘大小
    QPushButton *newGame = new QPushButton("New Game", this);
    newGame->setGeometry(QRect(10, 150, 100, 50));
    QPushButton *reStart = new QPushButton("New Boardsize", this);
    reStart->setGeometry(QRect(10, 210, 120, 50));

    //连接
    connect(beginButton, &QPushButton::clicked, this, &GameWidget::onSizeButton9Clicked);
    connect(sizeButton11, &QPushButton::clicked, this, &GameWidget::onSizeButton11Clicked);
    connect(sizeButton13, &QPushButton::clicked, this, &GameWidget::onSizeButton13Clicked);
    connect(newGame, &QPushButton::clicked, this, [this]() { this->newGame(); });
    connect(reStart, &QPushButton::clicked, this, &GameWidget::restartGame);
}
```

boardwidget中:

主要函数及其作用

```

37
38 ▶ void BoardWidget::paintEvent(QPaintEvent *)//实现棋盘，棋子，标记鼠标位置，上一次落子位置的绘制 {...}
110
111 ▶ void BoardWidget::mouseReleaseEvent(QMouseEvent *event)//鼠标点击，获取坐标，调用downPiece {...}
133
134 ▶ void BoardWidget::mouseMoveEvent(QMouseEvent *event)//鼠标移动显示即将落子的位置。 {...}
149
150 ▶ void BoardWidget::setTrackPos(const QPoint &value)//包含更新棋盘 {...}
155
156 ▶ void BoardWidget::initBoard()//初始化棋盘 {...}
161
162 ▶ void BoardWidget::downPiece(int x, int y) {...}
193
194 ▶ void BoardWidget::switchNextPlayer()//此函数用于实现棋手之间的转换 {...}
199
200 ▶ void BoardWidget::newGame()//开始新游戏 {...}
221
222 ▶ void BoardWidget::initTime()//初始化时间 {...}
240
241 ▶ void BoardWidget::onTimerTimeout()//不是定时60s后执行，而是每秒执行（设置的间隔是1000ms），修改显示的时间 {...}
256
257 ▶ void BoardWidget::onChangeTimeButtonClicked()//更改时间，点击后可输入新的时间间隔 {...}
278
279 ▶ Board BoardWidget::getBoard() {...}
283
284 ▶ void BoardWidget::setReceivePlayers(const QSet<int> &value) {...}
288
289 ▶ void BoardWidget::initVisited() {...}
299
300 ▶ bool BoardWidget::isSuicidalMove(int x, int y)//检查落子是否为自杀行为（导致己方棋子没气） {...}
307
308 ▶ bool BoardWidget::hasLiberties( Board tempBoard, QVector<QVector<bool>> &visited, int x, int y, int color)
309 ▶ //判断在（x，y）坐标下的color色棋子是否有气 {...}
332
333 ▶ bool BoardWidget::capturesOpponent(int x, int y)//判断是否吃掉对方棋子 {...}
357
358 ▶ void BoardWidget::gameOver(int loser)//游戏结束，显示输家信息并开始新游戏 {...}
365

```

构造函数（boardwidget界面实现）

```

BoardWidget::BoardWidget(int boardSize, QWidget *parent) ://获取棋盘大小，创建棋盘界面
    QWidget(parent),
    visited(15, QVector<bool>(15, false)),
    trackPos(28, 28)
{
    setWindowTitle("NoGo");
    //设置棋盘大小
    BOARD_WIDTH = boardSize;
    BOARD_HEIGHT = boardSize;
    setFixedSize(WIDGET_SIZE);
    setMouseTracking(true);

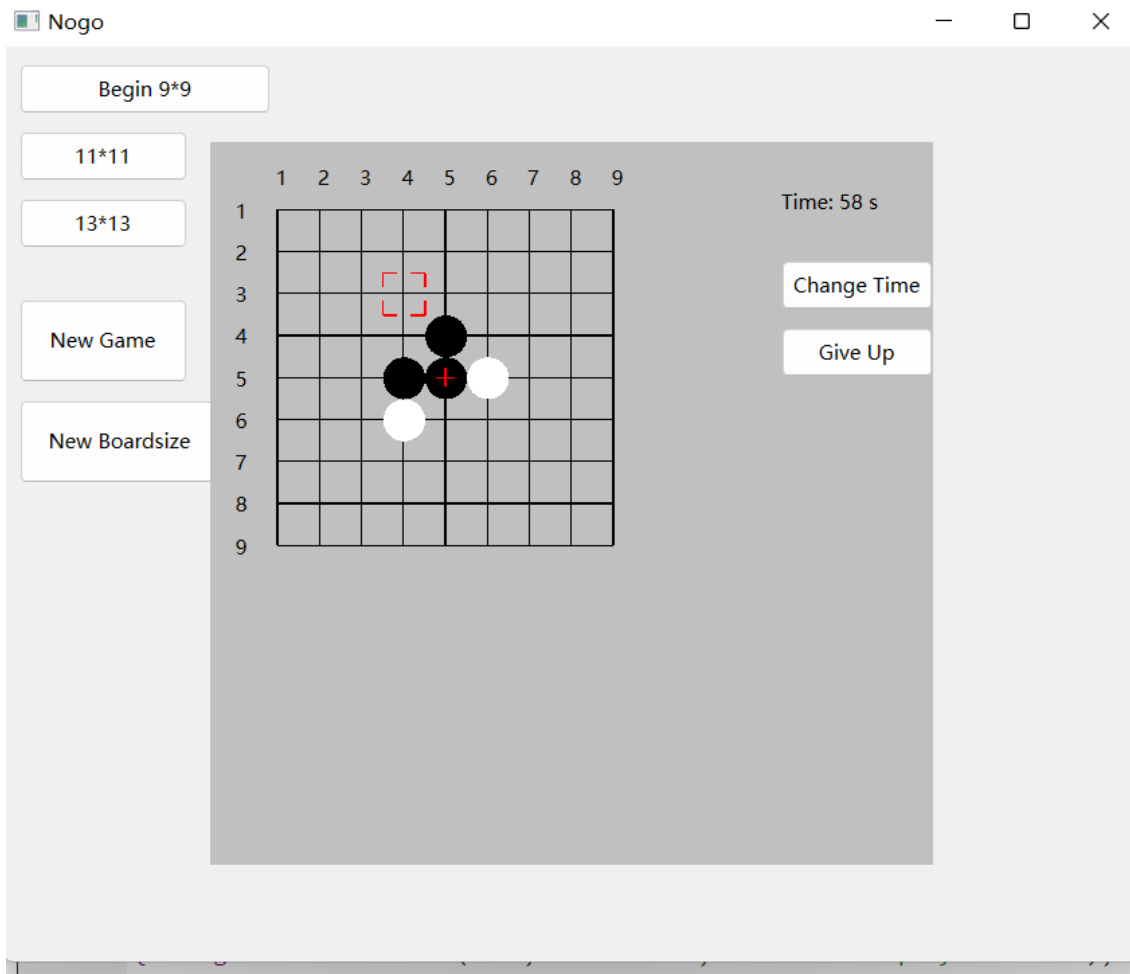
    //修改倒计时时间按钮
    QPushButton *changeTimeButton = new QPushButton("Change Time", this);
    changeTimeButton->setGeometry(QRect(340, 70, 90, 30));
    connect(changeTimeButton, &QPushButton::clicked, this, &BoardWidget::onChangeTimeButtonClicked);

    //初始化计时器和棋盘
    initTime();//initBoard用到了initTime初始化的timer指针，二者顺序不可交换
    initBoard();
}

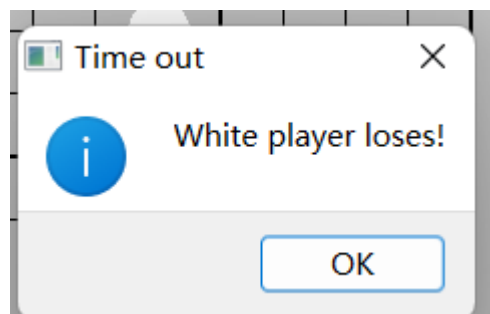
```

运行界面

游戏界面



胜负判断界面



吃子，自杀，超时，认输均会弹出胜负界面，弹出名称Time out或Game over。

遇到的困难及解决

若一次运行程序先后进行两次不同棋盘大小的对局，总是崩溃。原因：已存在一个布局，只是hide了没有delete。

删除布局后还是没能解决问题，又因为解引用了空指针崩溃。原因：delete后没处理指针，且必须使用类型转换后赋值给指针才不会报错，在这里卡了很久。

实现hasLiberties函数时出现未知bug导致board状态被莫名其妙更改，修改了记录已访问位置的方法后问题消失。

感谢

感谢孙亚辉老师和两位助教的悉心教导

感谢第十小组的成员

感谢中国人民大学和信息学院提供的学习平台