33- 作业报告

一、功能介绍

1. Canvas（画板）

基础操作：

窗口缩放：支持画板视图的放大缩小

缩放控制：滚轮或方向键拖动画布 / Ctrl+ 滚轮精确缩放，支持触控板

对象选择：单点选中，支持框选功能，可批量选择几何对象

对象管理：

拖动功能：选择任意几何对象拖动，根据父子关系自动生成变化图像

删除功能：删除父对象时自动清除所有子对象

隐藏 / 显示功能， show 按钮显示所有隐藏对象

撤回与重做功能：Ctrl+Z, Ctrl+Y

保存读取功能：Ctrl+S, Ctrl+L

全选功能：Ctrl+A

隐藏对象：Ctrl+H

显示设置：

标签管理：设置标签，支持显示 / 隐藏标签

样式设置：设置对象大小 / 线型

构建预览：创建所需接收的最后一个几何对象确定前显示预览

智能显示：不合法交点会暂时隐藏

界面布局：

工具面板：Tools 可以拖动到面板的其他位置

2. 点（Point）

创建方式：

空白处创建：使用点工具直接点击空白处

线 / 圆上创建：使用点工具点击在线或圆上

交点创建：选择模式 / 点工具点击两个线 / 圆的交点

辅助创建：在其他工具需要点的时候可点击空白处 / 线或圆上 / 交点

3. 圆（Circle）

创建方式：

两点作圆：依次选取圆心，圆上一点，创建圆

三点作圆：依次选取圆上三点创建圆

点和半径：依次选取圆心，半径（接收两点或线段）创建圆

圆弧：

创建方式：依次选取圆心，起始点，末端方向，逆时针作圆弧

半圆：

创建方式：依次选取两点逆时针方向作半圆

4. 线（Line）

创建方式：选取两点生成直线 / 射线 / 线段

5. 工具（Tools）

基础工具：

中点连线：选取两点 / 线段，创建中点

平行线：选取点，线（或线，点）创建直线

垂线：选取点，线（或线，点）创建垂线

中垂线：选取两点 / 线段，创建中垂线

角平分线：依次选取端点，顶点，端点，创建角平分线

切线：选取点，圆 / 圆弧（或圆 / 圆弧，点）生成切线，自动生成切点

交点工具：选取两几何对象创建交点

变换工具：

中心对称：选取几何对象，中心点，创建中心对称

轴对称：选取几何对象，轴线，创建轴对称

度量工具：

距离：依次选取两点 / 线段计算距离

角度：依次选取端点，顶点，端点，计算角度

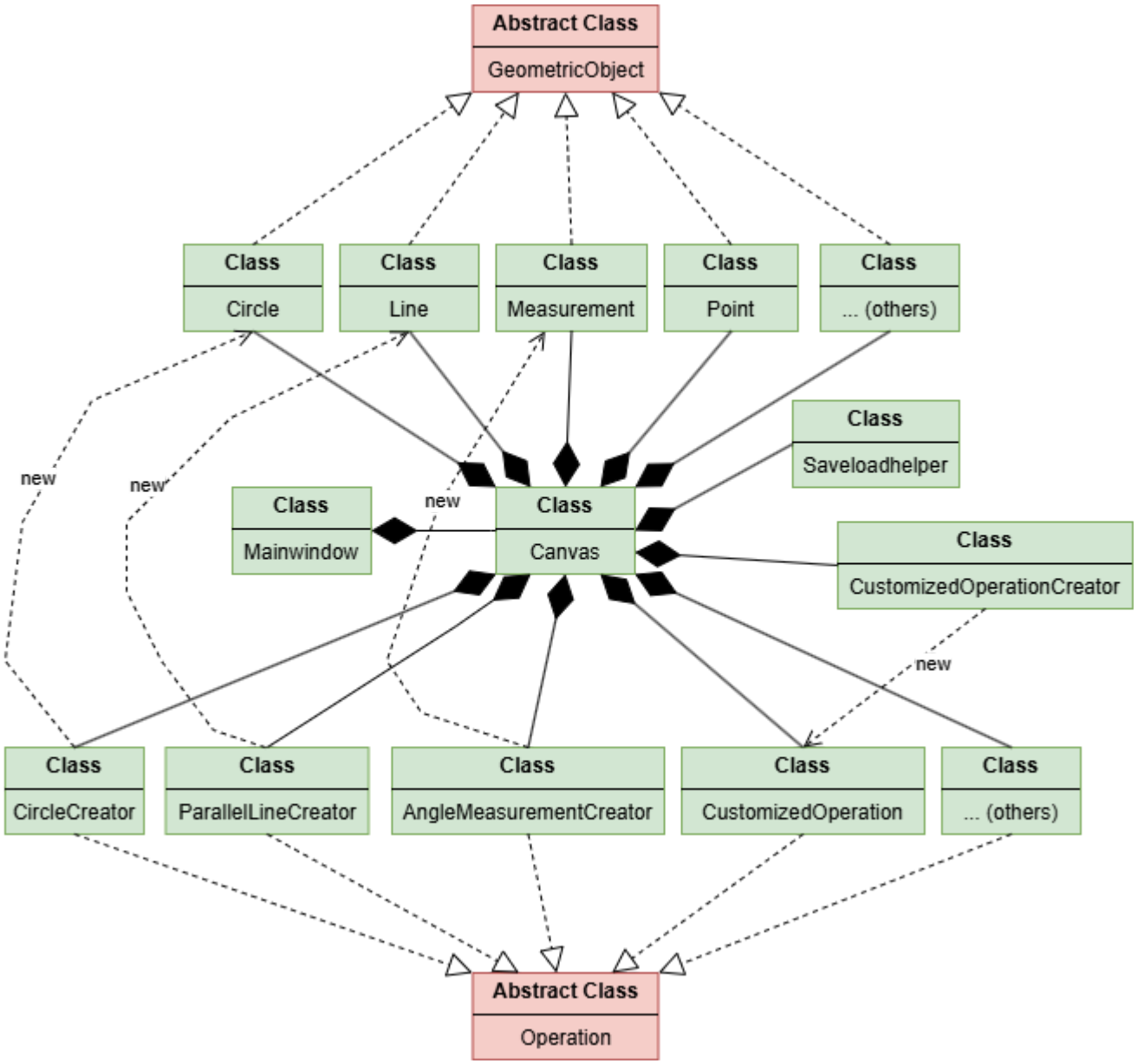
自定义工具：

使用说明：接受若干几何对象，自动判断父子关系，推断得到输入和输出，依此构造新的工具。

给予用户利用已有几何对象和工具“编程”的权力。

二、项目各模块与类设计细节

基本架构：如图所示。



几何对象的管理方式：

每个几何对象记录自己的父对象和子对象。子对象的位置、是否合法等信息由父对象决定。宏观来看，父子对象的关系使所有几何对象形成一个有向无圈图。

Point 的地位是特殊的。只有 Point 允许没有父对象。其他类型的子类在构造时必须提供其父对象和构造方式。

在每次 update 的过程中，先按照顺序更新每个几何对象的位置，然后再依次显示在屏幕上。

三、小组成员分工情况

陈羿桥：Mainwindow 前端设计；圆相关功能；功能测试；宣发

丁彦辰：Canvas 相关功能；点；文件功能；撤销重做；部分工具

刘小康：抽象类设计；线；弧与半圆；度量功能；交点；部分工具

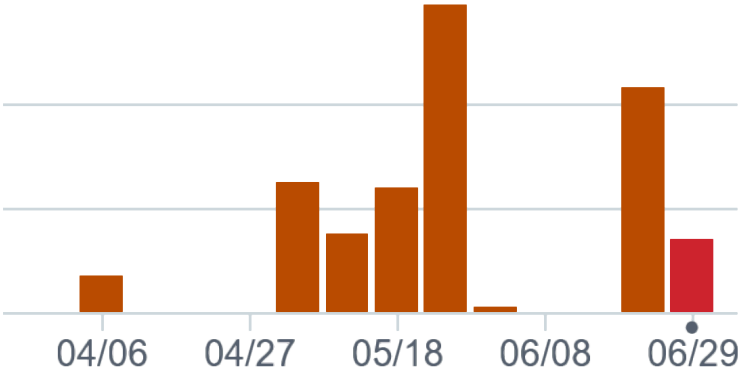
四、GitHub 仓库相关数据

1.项目的高关注度

路演结束后，尤其是期末周之后，本项目的 GitHub 仓库的访问（visit）量与克隆（clone）量显著增长 。这些数据直观地反映了本项目的热度，以及同学们对我们开发成果的认可。这份关注是宝贵的支持，也激励着本小组继续打磨代码，朝着“打造优秀几何画板应用”的目标迈进。

2.团队成员持续工作

仓库于 2025 年 4 月 8 日创建，过去三个月里，本仓库一共有 190 余次 commits 。团队成员保持高频开发节奏：路演前完成程序基本框架搭建，期末周结束后并也未松懈丝毫，而是持续优化功能、完善细节，最终打磨出当前版本。



五、总结与反思

本项目依托严谨的架构设计与高效的团队协作，成功实现了几何画板核心功能模块。在技术实现方面，以较为清晰的类接口设计与相对严密的对象关系管理，构建出一套几何对象体系，充分彰显面向对象编程的封装、继承与多态特性。开发过程中，小组成员积极探索 C++14 与 C++17 新标准，灵活运用 lambda 表达式、结构化绑定等高级语法，优化代码结构与执行效率，显著提升程序的可维护性与扩展性。

不过，项目推进中也暴露出部分技术细节的处理不足。例如，GeometricObject 类存在构造函数执行后需手动调用 flush () 函数才能正常运行的情况，小组未能及时采用工厂模式进行封装优化，导致潜在的安全风险与代码冗余。这一问题反映出成员在设计模式应用与代码优化意识上仍有提升空间。尽管如此，这些实践中的经验教训，促使我们对软件开发流程有了更深刻的理解简洁，不仅巩固了小组成员的专业技术能力，更在协作问题解决中积累了宝贵经验，为更加复杂的软件开发筑牢了根基。