*Документация на проект 2 по ООП*

Тема 8 – SVG файлове

Изготвил: Габриел Тупаров, 2MI0800131

Съдържание:

1. Увод
2. Преглед на предметната област
3. Проектиране
4. Реализация и тестване
5. Заключение

Линк към github хранилище на финалната версия:

[icydingo29/SVG-File-Editor at icydingo29-patch-1 (github.com)](https://github.com/icydingo29/SVG-File-Editor/tree/icydingo29-patch-1)

1. Увод

**Описание и идея на проекта** - Проектът „SVG файлове“ е създаден с целта да улесни създаването и обработването на файлове от типа “scalable vector graphics”. Това са вид изображения, които не използват пиксели. Програмата работи в диалогов режим, като въведените от потребителя данни се запазват в текстови файлове, годни за по-нататъшна обработка.

**Цел и задачи на разработката** – Цел на разработката е естествено да се осъществят функционалности, които могат да обработват данните, въведени от потребителя. След като данните биват преработени и оформени в подходящ вид програмата ги запазва в текстови (svg) файлове, за да може в бъдеще те отново да бъдат достъпни за промяна или заличаване. Осъществена е валидация при всяко вкарване на данни от страна на потребителя, за да се елиминират потенциални грешки, застрашаващи работата както и на програмата, така и на потребителят. При всяко отваряне на файл се прави проверка, дали файлът е отворен успешно, за да се предотврати губене на информация.

**За работа с файловете са реализирани 6 функции.**

Функция 1- „open <file>“

За отваряне на файл. Ако такъв не съществува, се извежда подходящо съобщение и се създава нов с въведеното име. Ако вече има отворен файл, се извежда подходящо съобщение. Ако не е въведено име на файл, който да бъде отворен, се извежда подходящо съобщение. Ако всичко е изрядно, файлът се отваря, всичките данни от него се вкарват в програмата, и той се затваря отново. За целта на реализация, обаче, той все още се счита за „отворен“ за обработка, като смисълът на това е да принуждава потребителя да запазва промените, които е направил, или да се откаже от тях.

Функция 2 – „close“

За „затваряне“ на файл за обработка, без да бъдат запазени промените, направени върху него.

Функция 3 – „save“

За „запазване“ на файл. Промените, направени върху текущия файл, се записват върху

текущия файл. Файлът се затваря за обработка.

Функция 4 – „saveas <file>“

За “запазване“ на файл. Промените, направени върху текущия файл, се записват в друг файл. Ако въведеното име на файл съвпада с името на сегашния отворен файл, променените данни се записват върху текущия файл. Файлът се затваря за обработка.

Функция 5 – „help“

За извеждане инструкции как се използват разнообразните команди.

Функция 6 – „exit“

За изход от програмата. Ако има незаписани промени или файл е отворен за обработка, се извежда подходящо съобщение. Ако не, програмата приключва.

**За работа с данните, извлечени от файловете, са реализирани 5 функции.** Всяка една от тях прави проверка дали е отворен файл за обработка и ако не е, извежда съобщение.

Функция 1 – „print“

Извежда на конзолата всички фигури, които са заредени в паметта.

Функция 2 – „create“

Създава нова фигура, която ще бъде добавена към списъка със заредените от файла фигури. Може да се добави правоъгълник, кръг, или права линия. И в трите случая се прави проверка на вход, за да се елиминира възможността за некоректно въведени данни.

Функция 3 – „erase <n>”

От списъка с фигури изтрива тази с пореден номер n. Ако n е невалидно или липсва, се извежда подходящо съобщение.

Функция 4 – „translate [<n>]”

За транслиране на фигури. Ако n е указано, транслира само нея, а ако не е, транслира абсолютно всички форми. За целта е нужно да бъдат въведени параметрите за изместване, като функцията е направена така, че да няма значение, дали вертикалното или хоризонталното изместване е въведено първо.

(translate vertical=10 horizontal=50 ще върши същото действие като translate horizontal=50 vertical=10)

Функция 5 – „within <option>”

Извежда на екрана всички фигури, които изцяло се съдържат в даден регион. Потребителят мое да укаже чрез какъв да бъде регионът – кръг (circle) или правоъгълник (rectangle).

1. Преглед на предметната област

Проектът е изграден с множество класове, които опростяват процеса на структуриране, обработване и писане на данни. Създадени са също и помощни структури, със същата цел. Ключова роля играе концепцията на наследяването и на полиморфизма.

Наследяването е начин за създаване на нови класове чрез използване на компоненти и поведение на вече съществуващи класове.

При полиморфизма едни и същи действия се реализират по различен начин в зависимост от обектите, върху които се прилагат. Действията се наричат полиморфни. Класовете, върху които ще се прилага, в случай имат общ родител, тоест са производни на един и същи клас. В базовия (родителския) клас се дефинира виртуален метод, които съответства на полиморфното действие. В производния клас се предефинира виртуалния метод.

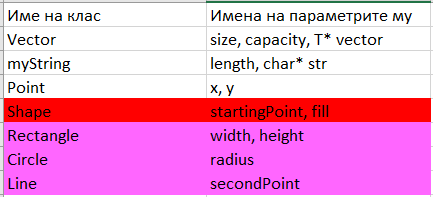
На указателите към базовия клас може да се присвоят адреси на обекти от производните класове, което позволява съществуването на хетерогенни контейнери.

Проблем би могъл да бъде входът, който потребителят въвежда, но програмата проверява дали въведените данни са в коректен формат. Ако не са, се извежда подходящо съобщение.

Всеки човек с достъп до програмата има една и съща роля, тъй като апликацията не е направена да дава различни привилегии на различни потребители.

1. Проектиране

На следната таблица са показани трите основни класа, както и имената на техните характеристики.



Vector е клас със същите функционалности като std::vector. Включен е в проекта за улеснение на работата с данни.

myString е клас със същите функционалности като std::string. Включен е в проекта за улеснение на работата с масиви от низове. Той съхранява масив от символи както и неговата дължина. Вместо да обработваме масиви от символи всеки път, използваме този клас, за да въвеждаме командите, които програмата да изпълнява, както и данните за разнообразните обекти. Други начини, по който този клас е от помощ, е при генерирането на име на файл, в който да бъдат записани променените данни, което е работата на една от главните функции. Улеснено е също и разпознаването на въведена команда, тък като сравняваме обекти директно от myString, а не низове, което прави кода по-разбираем. Освен това myString има и фунцкионалности за промяна на входа на потребителя, за да форматира входа в подходящ вид.

Point е клас, съдържащ данните на една точка в координатната система на svg файла.

Shape е базовия клас, на който останалите класове в проекта за производни. Затова той е оцветен в червено, а наследниците му в розово. В него се съдържат универсалните за всяка форма параметри – начална точка, обект от тип Point, и цвят fill. Shape съдържа и всички функционалностти, които наследниците му трябва да имат предефинирани. Shape е абстрактен клас и не могат да бъдат създадени негови инстанции, но могат да бъдат създадени показатели.

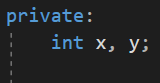
Rectangle е първият наследник на Shape. Като надграждане има допълнителните параметри width и height. Width е широчината на правоъгълника, тоест хоризонталното отместване спрямо началната точка, а height е височината на правоъгълника, тоест вертикалното отместване спрямо началната точка.

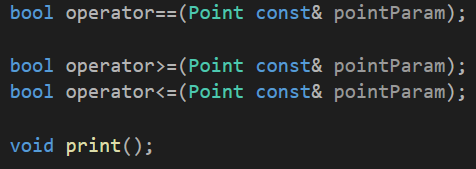
Circle е вторият наследник на Shape. Като надграждане има допълнителният параметър radius, което е радиусът на кръга.

Line е последният наследник на Shape. Като надграждане има допълнителният параметър secondPoint, което е крайната точка на правата линия.

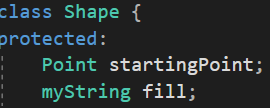
За улеснение са реализирани подходящи селектори и мутатори, конструктори без параметри, конструктори със параметри, оператор =, оператор ==, „копи-конструктор“ и други на класовете, които ги изискват, или с които би се работило по-лесно с реализирани такива функционалости.

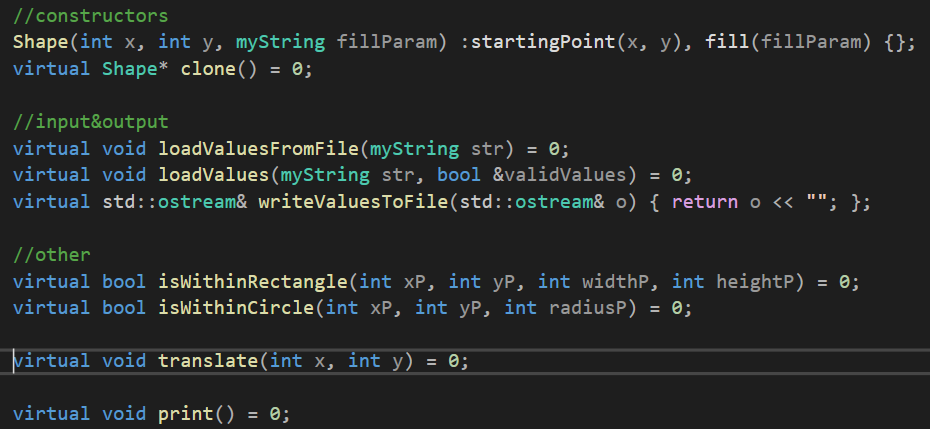
Реализирани са и помощни функции, чиято цел е да правят кода на главните команди по-четим и по-кратък.

1. Реализация и тестване
   1. Най-малкият и най-просто устроен клас е Point.

Параметрите му са от тип int. Освен селектори и мутатори, констуктор без параметри, конструктор с параметри и оператор =, са реализирани и следните функционалости: 

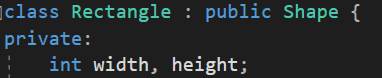
print() – за извеждане на конзолата; операторът >= проверява дали текущата точка е по-далече от центъра на координатната система в сравнение с подадена точка; операторът <= проверява дали текущата точка е по-близо до центъра на координатната система в сравнение с подадена точка; операторът == прави проверка дали текущата точка е еквивалентна на подадена.

**4.2** Следващия клас е Shape. Освен конструктор с параметри са реализирани и следните функционалности:



сlone() – за създаване на указател към обект от тип Shape; loadValuesFromFile() - за зареждане на данни от файл; loadValues() – за зареждане на данни от обект от тип myString

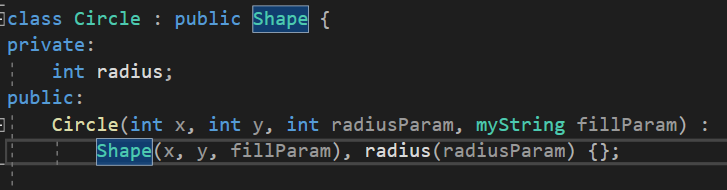
isWithinRectangle() – проверява дали текущата фигура е вътре в правоъгълник със зададените параметри; isWithinCircle() - проверява дали текущата фигура е вътре в кръг със зададените параметри;

**4.3** Първият наследник на Shape – класът Rectangle 

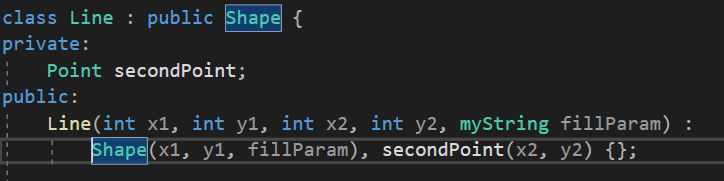
Rectangle предефинира всички виртуални функции на Shape. Конструкторът му по подразбиране изглежда по следния начин:



**4.4** Вторият наследник на Shape – класът Circle



**4.5** Третият наследник на Shape – класът Line



Единственият случай, където има динамична памет, е в помощните класове Vector и myString. Класовете се грижат за собствената си динамична памет, така че да се намалят memory leak-овете до 0. Като оптимизация би могло да се предложи разширяването на допустимият вход за командите – така при грешки по невнимание няма да има нужда да се въвежда цялата команда наново. (например вместо “save” потребителят да напише “saev” и програмата да разбере, коя команда потребителят е искал да въведе)

Програмата е тествана много пъти в процеса на писане, както и със тестовия сценарии, включен в zip архива.

1. Заключение

Проектът „SVG файлове“ има за цел да опрости монотонното създаване на този тип файлове. Въвеждането и извеждането на фигурите се постига с диалоговия режим на работа.

Като идея за бъдещи подобрения е приложението да има връзка с интернет, така че да може файловете да се качват в страница, която може да бъде отваряна от всички хора, които трябва да имат достъп до файловете. Може и да се направи мобилна версия, която да действа в синхрон с компютърната, така че да може потребителят да променя файловете си от всякакъв уред.