Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

**РАЗРАБОТКА ПЛАГИНА «СКАМЬЯ» ДЛЯ «КОМПАС-3D»**

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

по дисциплине

«Основы разработки САПР» (ОРСАПР)

Выполнил:

студент гр. 581

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Панишева Д.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Руководитель:

к.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Калентьев А.А.

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

# СОДЕРЖАНИЕ

[Введение 3](#_Toc154662118)

[2 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ 4](#_Toc154662119)

[2.1 Описание предмета проектирования 5](#_Toc154662120)

[2.2 Выбор инструментов и средств реализации 6](#_Toc154662121)

[2.3 Назначение плагина 6](#_Toc154662122)

[3 ОБЗОР АНАЛОГОВ ПЛАГИНА 7](#_Toc154662123)

[4 ОПИСАНИЕ РЕАЛИЗАЦИИ 8](#_Toc154662124)

[5 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ 12](#_Toc154662125)

[6 ТЕСТИРОВАНИЕ ПЛАГИНА 15](#_Toc154662126)

[6.1 Функциональное тестирование 15](#_Toc154662127)

[6.2 Модульное тестирование 16](#_Toc154662128)

[6.3 Нагрузочное тестирование 17](#_Toc154662129)

[Заключение 21](#_Toc154662130)

[Список источников 22](#_Toc154662131)

# Введение

Автоматизация моделирования имеет огромное значение для развития науки, техники и производства в современном обществе. В настоящее время автоматизация – основной способ повышения производительности и эффективности труда инженерно-технических работников, занимающихся моделированием сложных устройств. Использование автоматизации в проектировании позволяет создавать все более сложные технические объекты и гибко реагировать на появление новых решений и технологий в той или иной области техники. Она позволяет значительно повысить точность расчетов, выбрать наилучшие варианты для реализации на основе строгого математического анализа, оценить технологические и экономические характеристики производства, значительно повысить качество конструкторской документации, существенно сократить сроки проектирования и передачи конструкторской документации в производство, эффективнее использовать технологическое оборудование с программным управлением [1].

Таким образом, целью данной работы является разработка плагина, автоматизирующего построение модели «Скамья» для системы автоматизированного проектирования КОМПАС-3D с помощью интегрированной среды разработки Visual Studio 2022 Сommunity [2].

Интегрированная среда разработки Visual Studio — это стартовая площадка для написания, отладки и сборки кода, а также последующей публикации приложений. Интегрированная среда разработки (IDE) представляет собой многофункциональную программу, которую можно использовать для различных аспектов разработки программного обеспечения.

# 2 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ

В рамках лабораторных работ в соответствии с технически заданием требовалось разработать плагин, на языке программирования C#. Плагин на основе входных параметров, интегрируясь с системой «КОМПАС-3D», строит модель «Скамья» [3].

Сроки реализации данного проекта:

* выбор темы и создание git-репозитория (18.09.24 - 24.09.24);
* составление технического задания (25.09.24 - 08.10.24);
* составление проекта системы (09.10.24 - 22.10.24);
* прототип плагина (20.11.24 - 03.12.24);
* готовый плагин (04.12.24 - 31.12.24);

Необходимо чтобы плагин позволял задавать параметры по умолчанию, а также изменять входные параметры забора, такие как:

* длина скамьи;
* высота сиденья;
* высота ножки;
* длина ножки;
* ширина сиденья;

В ходе анализа реализации плагина были выявлены проблемы в сложно читаемой документации API для САПР КОМПАС-3D. Из положительных сторон можно выделить, что в открытом доступе есть множество различных примеров кода по правильному использованию API.

## 2.1 Описание предмета проектирования

Назначение разрабатываемого плагина обусловлено быстрым моделированием скамеек разных типов. Благодаря данному расширению, мастера по скамьям могут наглядно рассмотреть спроектированную модель, при необходимости перестроить под необходимые им параметры.

На рисунке 2.1 представлена модель скамьи.

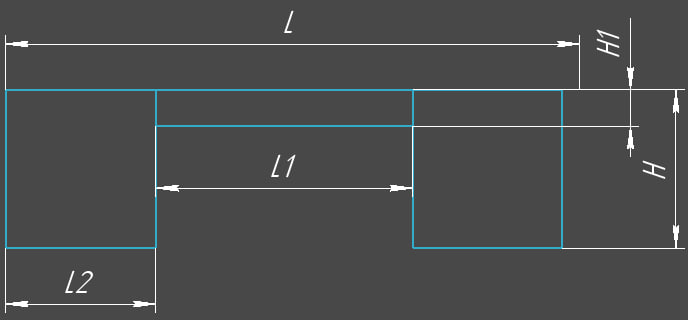


Рисунок 2.1 — Чертеж скамьи с размерами, вид с боку

Параметры скамьи:

* длина скамьи L (100-200см);
* высота сиденья H1 (10-25см);
* высота ножки H (20-40см);
* длина ножки L2 (20-50см);
* ширина сиденья H2 (30-60см);
* длина скамьи L1 равна длине ножки L2 помноженной на два и вычистить длину скамьи L. Длина сидячей поверхности будет определятся автоматически в зависимости от размера длины скамьи.

АС должна иметь пользовательский интерфейс с возможностью изменения значений, представленных выше, и последующим построении объекта «Скамья» в САПР КОМПАС-3D. В плагине должны проходить проверки значений, вводимых пользователем. Реализуемый плагин должен обеспечивать обработку ошибочных ситуаций, возникающих в процессе работы. При нажатии на кнопку «Построить» должна проходить проверка правильности ввода данных. Если данные некорректные, то должно высветиться окно с ошибкой построения и не будут применяться введенные параметры.

## 2.2 Выбор инструментов и средств реализации

На основе требований к техническому заданию программа выполнена на языке программирования C# в среде Microsoft Visual Studio 2022 с использованием .NET 6.0, библиотеки для Kompas 3D [4].

Инструментом тестирования и создания модульных тестов был выбран тестовый фреймворк NUnit [5].

Для реализации пользовательского интерфейса был использован Windows Forms [6].

## 2.3 Назначение плагина

Назначение плагина обусловлено быстрым моделированием скамеек разных размеров. Благодаря данному расширению, мебельщики могут наглядно рассмотреть спроектированную модель, при необходимости перестроить под необходимые им параметры.

# 3 ОБЗОР АНАЛОГОВ ПЛАГИНА

Плагин "Bench Design" для SketchUp [7]

Основная функциональность: Плагин Bench Design предназначен для создания различных моделей скамеек в формате 3D для использования в SketchUp. Сходство заключается в возможности создания трехмерных моделей скамеек.

Скриншот пользовательского интерфейса представлен на рисунке 3.1.

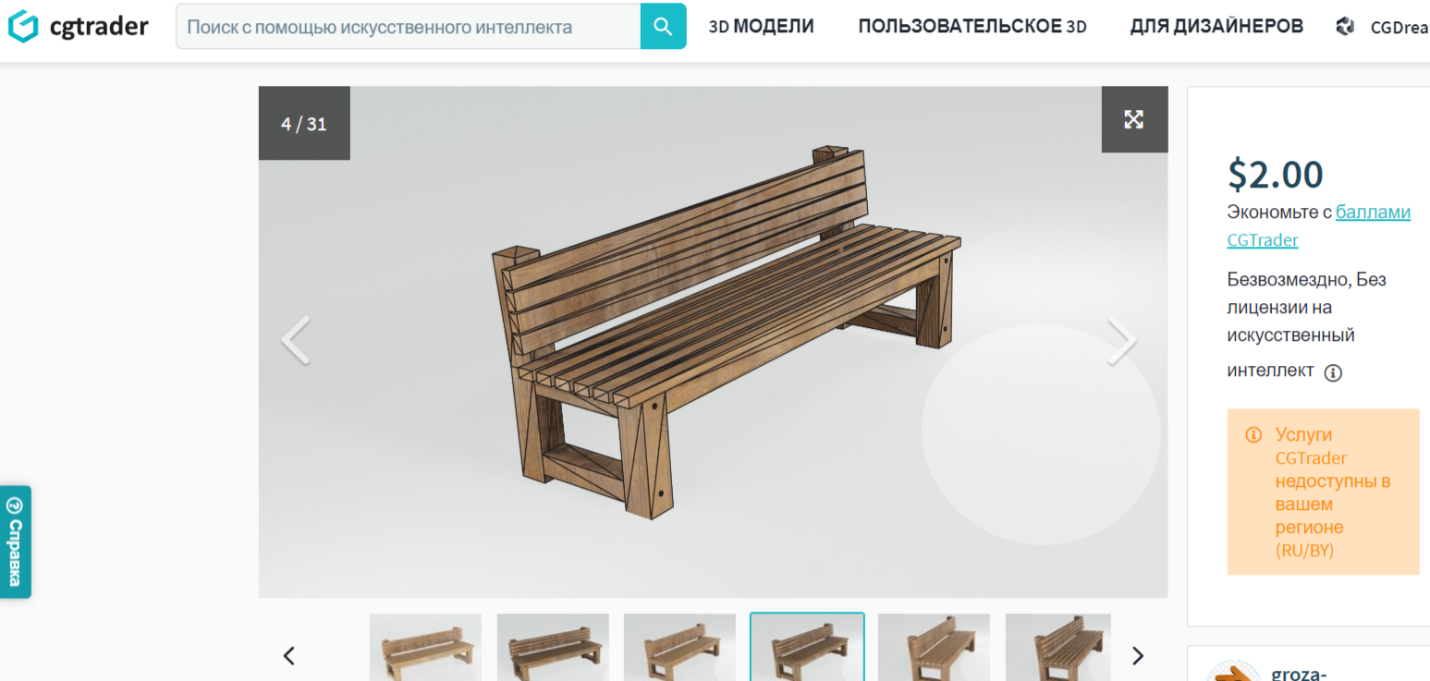


Рисунок 3.1 – Скриншот пользовательского интерфейса Bench Design

Плагин "Seat Generator" для Blender [8]

Основная функциональность: плагин Seat Generator предназначен для автоматизированного создания сидений, в том числе скамеек, в программе Blender. Сходство заключается в создании элементов мебели.

Скриншот пользовательского интерфейса представлен на рисунке 3.2.

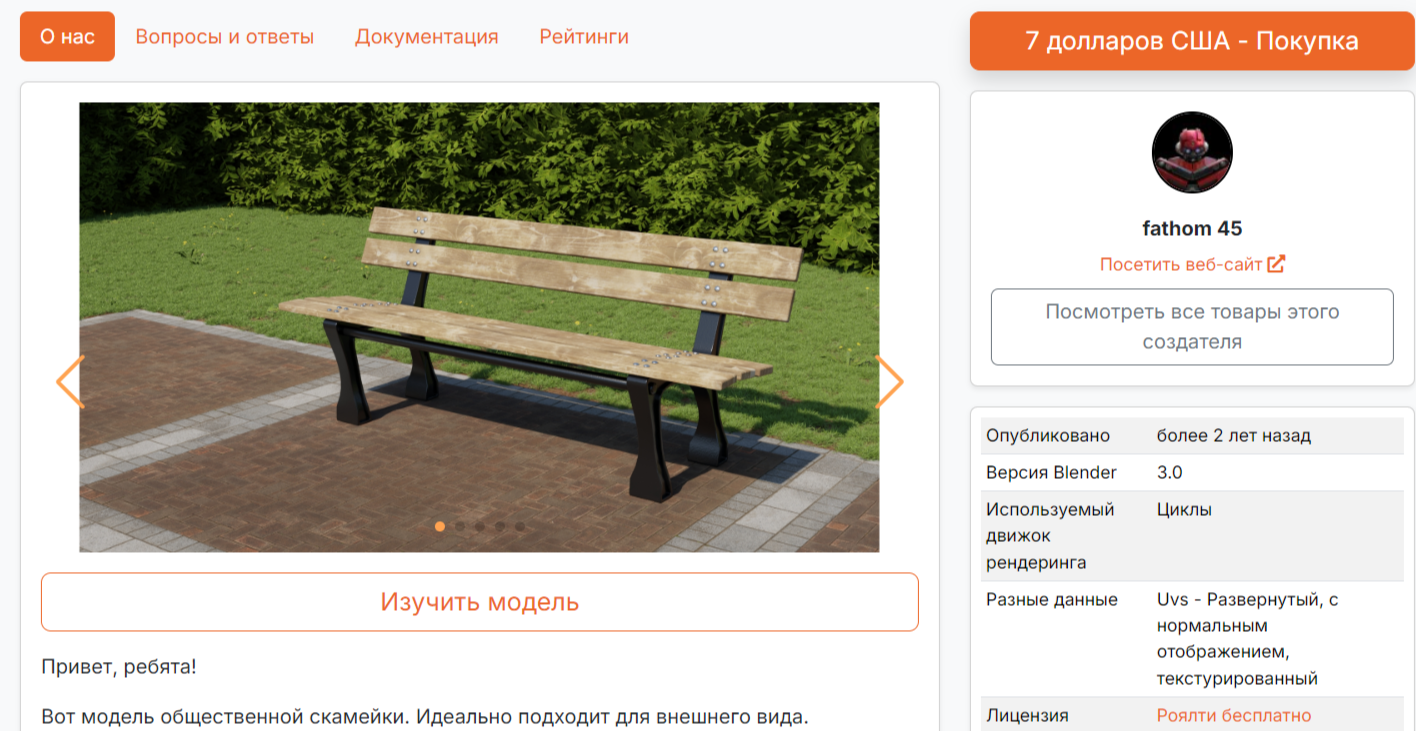


Рисунок 3.2 – Скриншот пользовательского интерфейса Blender

Оба аналога позволяют создавать трёхмерные модели скамеек, однако каждый из них имеет свои уникальные особенности и применяется в различных программах для моделирования.

# 4 ОПИСАНИЕ РЕАЛИЗАЦИИ

UML диаграмма классов представляет собой графическую интерпретацию классов системы, их атрибутов, методов и взаимосвязей между ними [9].

На рисунке 4.1 представлена диаграмма классов.

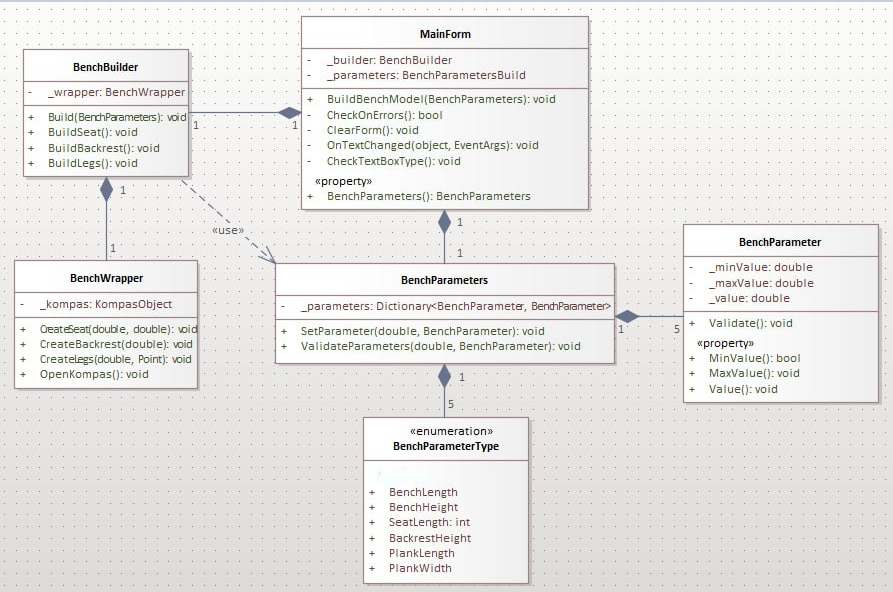


Рисунок 4.1 – Диаграмма классов

Разберем основные классы проекта:

 –  MainForm – является главным окном приложения;

 –  BenchParameters - класс, хранящий в себе все параметры модели;

 –  BenchParameter – описывает параметр модели;

 –  BenchBuilder - класс, отвечающий за строительство модели скамейки.

– BenchWrapper – осуществляет взаимодействие с компасом;

– BenchParameterType – тип параметра.

В итоговом проекте созданы следующие классы и методы, которые отображены на итоговой диаграмме классов (рисунок 4.2).

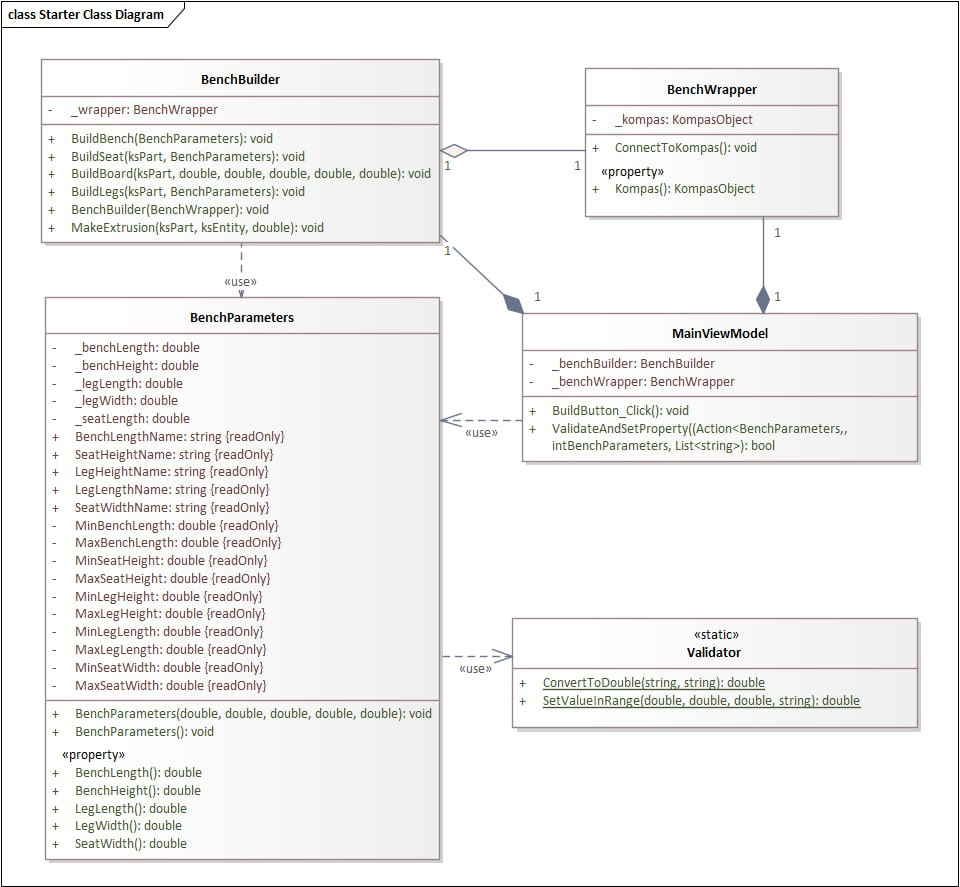


Рисунок 4.2 – Итоговая диаграмма классов

В таблицах 4.1-4.9 представлена информация о свойствах и методах каждого из классов.

Таблица 4.1 – Свойства класса BenchBuilder

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_ wrapper | BenchWrapper | Объект-обертка для работы с компас |

Таблица 4.2 – Методы класса BenchBuilder

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| BuildBench | BenchParameters | void | Строит скамью на основе параметров |
| BuildSeat | ksPart, BenchParameters | void | Строит сиденье скамьи |
| BuildBoard | ksPart, double, double, double, double | void | Строит доску скамьи |
| BuildLegs | ksPart, BenchParameters | void | Строит ножки скамьи |
| BenchBuilder | BenchWrapper | void | Конструктор класса |
| MakeExtrusion | ksPart, kEntity, double | void | Выполняет операцию выдавливания |

Таблица 4.3 – Свойства класса BenchWrapper

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_ kompas | KompasObject | Объект компас |

Таблица 4.4 – Методы класса BenchWrapper

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| ConnectToKompas | - | void | Подключается к компасу |
| Kompas | - | KompasObject | Возвращает объект компас |

Таблица 4.5 – Свойства класса BenchParameters

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_ benchLength | double | Длина скамейки |
| \_benchHeight | double | Высота скамейки |
| \_legLength | double | Длина ножек |
| \_legWidth | double | Ширина ножек |
| \_seatLength | double | Длина сиденья |
| BenchLengthName | string (readOnly) | Имя свойства длины скамейки |
| SeatHeightName | string (readOnly) | Имя свойства высоты сиденья |
| LegHeightName | string (readOnly) | Имя свойства высоты ножек |
| LegLengthName | string (readOnly) | Имя свойства длины ножек |
| SeatWidthName | string (readOnly) | Имя свойства ширины сиденья |
| MinBenchLength | double (readOnly) | Минимальная длина скамейки |

Окончание таблицы 4.5

|  |  |  |
| --- | --- | --- |
| MaxBenchLength | double (readOnly) | Максимальная длина скамейки |
| MinBenchHeight | double (readOnly) | Минимальная высота скамейки |
| MaxBenchHeight | double (readOnly) | Максимальная высота скамейки |
| MinSeatHeight | double (readOnly) | Минимальная высота сиденья |
| MaxSeatHeight | double (readOnly) | Максимальная высота сиденья |
| MinLegHeight | double (readOnly) | Минимальная высота ножек |
| MaxLegHeight | double (readOnly) | Максимальная высота ножек |
| MinLegLength | double (readOnly) | Минимальная длина ножек |
| MaxLegLength | double (readOnly) | Максимальная длина ножек |
| MinSeatWidth | double (readOnly) | Минимальная ширина сиденья |
| MaxSeatWidth | double (readOnly) | Максимальная ширина сиденья |

Таблица 4.6 – Методы класса BenchParameters

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| BenchParameters | double, double, double, double, double | void | Конструктор класса |
| BenchParameters | - | void | Конструктор без параметров |
| BenchLength | - | double | Возвращает длину скамейки |
| BenchHeight | - | double | Возвращает высоту скамейки |
| LegLength | - | double | Возвращает длину ножек |
| LegWidth | - | double | Возвращает ширину ножек |
| SeatWidth | - | double | Возвращает ширину сиденья |

Таблица 4.7 – Свойства класса MainViewModel

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_ benchBuilder | BenchBuilder | Объект для построения скамейки |
| \_benchWrapper | BenchWrapper | Объект-обертка для работы с компас |

Таблица 4.8 – Методы класса MainViewModel

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| BoildButton\_Click | - | void | Обработчик нажатия кнопки создания скамейки |
| ValidateAndSetProperty | Action<BenchParameters>, int, BenchParameters, List<string> | bool | Проверяет и устанавливает свойство |

Таблица 4.9 – Методы класса Validator (static)

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| ConvertToDouble | string, string | double | Конвертирует строку в число |
| SetValuelnRange | double, double, double, double, string | double | Устанавливает значение в пределах диапазона |

Разберем основные изменения классов проекта после реализации:

Класс BenchBuilder был дополнен новыми методами:

* + BuildBoard(kPart, double, double, double, double, double) для создания досок с параметрами.
  + MakeExtrusion(ksPart, ksEntity, double) для выполнения операции выдавливания.
  + Добавлен конструктор BenchBuilder(BenchWrapper).

Класс BenchWrapper получил новый метод:

* + ConnectToKompas() для подключения к объекту KompasObject.

В классе BenchParameters изменена структура хранения параметров:

* + Вместо коллекции параметров теперь используются отдельные поля: \_benchLength, \_benchHeight, \_legLength, \_legWidth, \_seatLength.
  + Добавлены свойства для каждого параметра, обеспечивающие доступ и управление длиной, шириной, высотой и другими параметрами.

Класс MainForm обновлен:

* + Убрана связь с объектом BenchParameters.
  + Связь между MainForm и BenchWrapper изменена на более явную композицию, что упростило взаимодействие.
  + Реализован метод BuildButton\_Click(), который вызывает построение с использованием BenchBuilder.

Новый статический класс Validator:

* + Добавлены методы для проверки и преобразования параметров:

ConvertToDouble(string, string) — для преобразования значений.

SetValueInRange(double, double, double, string) — для проверки и корректировки значений в заданном диапазоне.

# 5 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ

Макет пользовательского интерфейса включает в себя окно, в котором пользователь вводит параметры для дальнейшего построения модели «Скамейка» в САПР КОМПАС-3D.

На рисунке 5.1 представлен пользовательский интерфейс плагина.

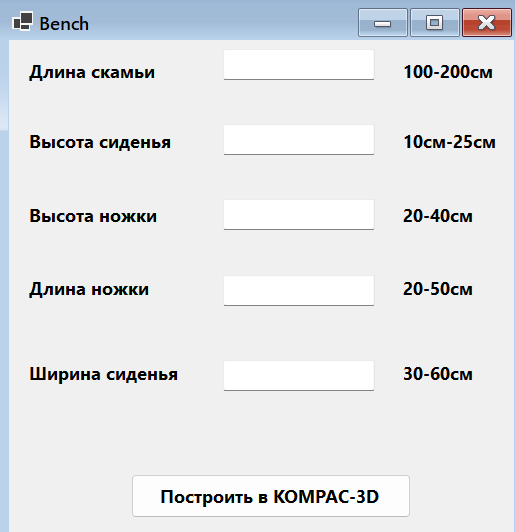


Рисунок 5.1 – Пользовательский интерфейс плагина

Если все данные были введены правильно, то поле ввода будет подсвечиваться зеленым и после нажатия кнопки "Построить" модель скамейки будет создана (Рисунки 5.2 и 5.3)

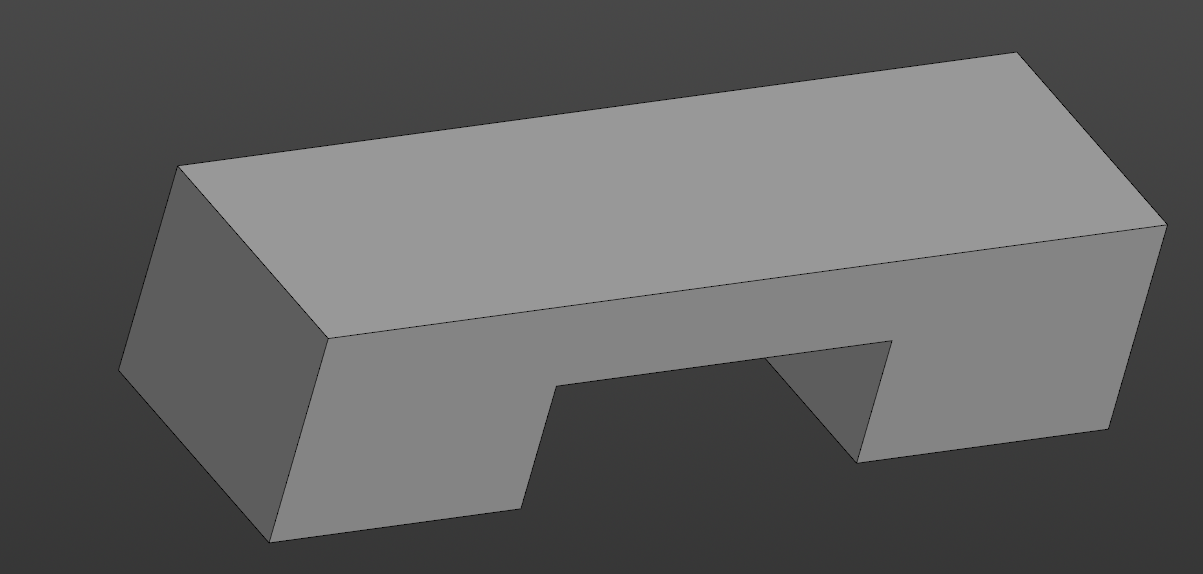


Рисунок 5.2 – Модель скамейки в КОМПАС-3D



Рисунок 5.3 – Подсветка полей зеленым, когда параметры введены в верном диапазоне

В случае некорректного ввода параметров, программа подсвечивает поле ввода красным и выводит ошибку (Рисунки 5.4 и 5.5).

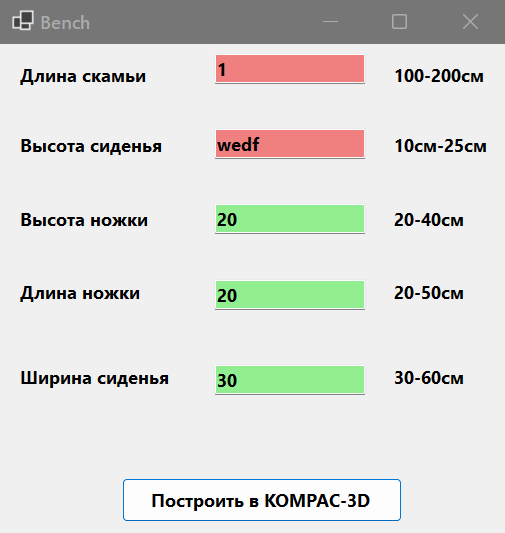


Рисунок 5.4 - Подсветка полей красным, когда параметры введены в не верном диапазоне

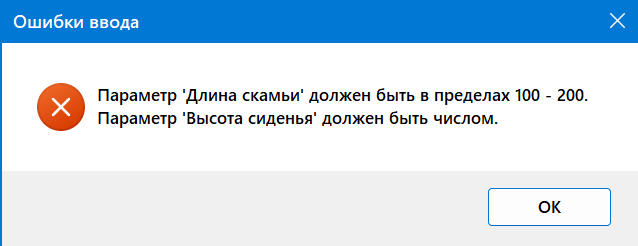


Рисунок 5.3 – Реакция плагина на некорректный ввод

Возможные варианты ошибок и их решений перечислены в таблице 5.1.

Таблица 5.1 – Ошибки и их варианты исправления

|  |  |
| --- | --- |
| **Текст ошибки** | **Варианты исправления** |
| Параметр A должен быть задан в следующем диапазоне: [\* - \*] | Задать значение для параметра A входящее в требуемый диапазон |
| Числовое поле параметра A содержит недопустимый символ | Удалить в поле параметра A недопустимый символ |

# 6 ТЕСТИРОВАНИЕ ПЛАГИНА

Тестирование позволяет убедиться в работоспособности программы, выявлять ошибки при изменении какого-либо функционала.

## 6.1 Функциональное тестирование

При функциональном тестировании проверялась корректность работы плагина «Скамья», а именно соответствие полученного результата в виде трехмерной модели с входными параметрами [10].

Результаты тестирование минимальных (длина скамьи 100 см, высота скамьи 50 см, ширина ножки 20 см, длина ножки 20 см, ширина сидячей поверхности 10 см) и максимальных (длина скамьи 200 см, высота скамьи 70 см, ширина ножки 50 см, длина ножки 30 см, ширина сидячей поверхности 20 см) параметров модели «Киянка» представлена на рисунках 6.1 и 6.2.



Рисунок 6.1 – Результаты построения минимальных (слева) и максимальных (справа) параметров модели «Скамья»

## 6.2 Модульное тестирование

В целях проверки корректности работы методов и свойств классов при помощи тестового фреймворка NUnit проведено модульное тестирование [11]. проверялись открытые поля и методы.

На рисунке 6.3 представлены результаты модульного тестирования классов проектов, а именно: BenchLogicTests, PanelTests.

Степень покрытия проектов — сто процентов. Было написано 22 теста.

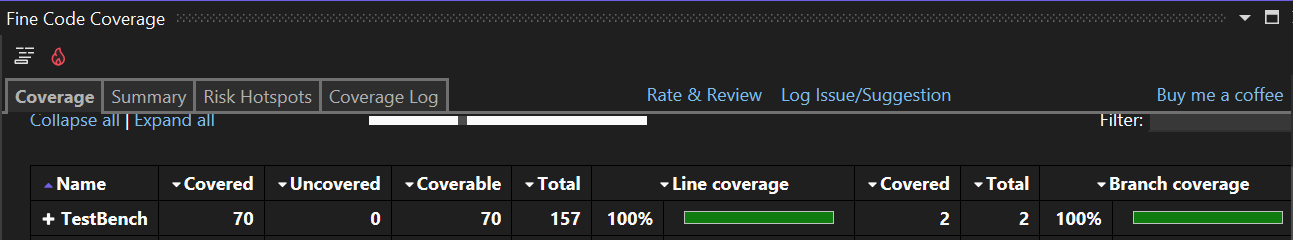


Рисунок 6.2 – обнаружение степени покрытия проектов с помощью Fine Code Coverage

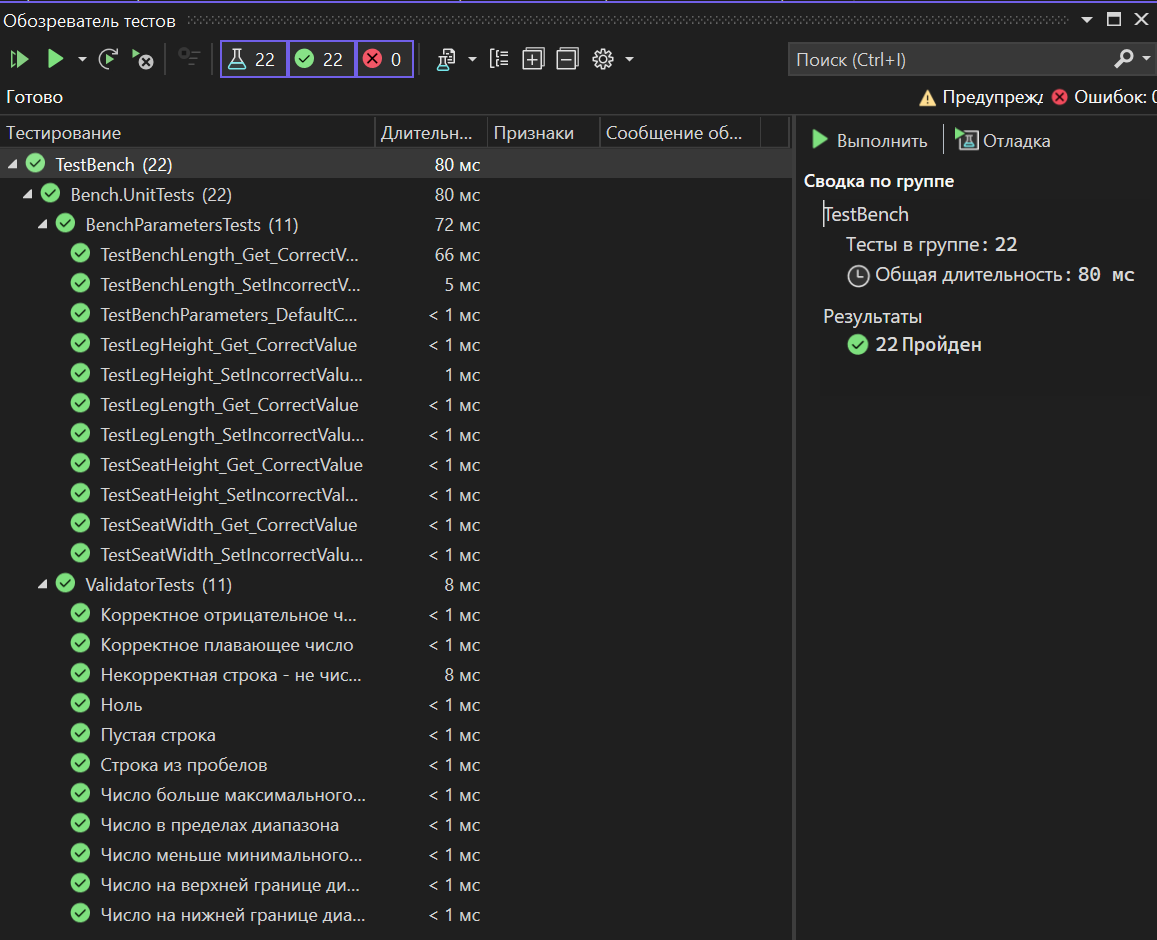


Рисунок 6.3 – Результаты модульного тестирования

**6.3 Нагрузочное тестирование**

В целях проверки производительности работы плагина, было проведено нагрузочное тестирование [12]. Тестирование производилось на ПК со следующей конфигурацией:

* AMD Ryzen 5 7535HS with Radeon Graphics 3.30 GHz;
* 16,0 GB ОЗУ (15,2 GB usable);
* Windows 11 Pro, 21H2, процессор x64.

На рисунке 6.4 представлен код для проведения нагрузочного тестирования. В нём использовался секундомер («Stopwatch»), который засекал время от начала построения. После успешного построения фигуры производилась запись результатов в текстовый файлы «log.txt».

|  |
| --- |
| var stopWatch = new Stopwatch();  var benchParameters = new BenchParameters(150, 20, 30, 40, 40);  var compassWrapper = new BenchWrapper();  var benchBuilder = new BenchBuilder(compassWrapper);  var streamWriter = new StreamWriter("log.txt", true);  var count = 0;  while (true)  {  stopWatch.Start();  const double gigabyteInByte = 0.000000000931322574615478515625;  compassWrapper.ConnectToKompas();  benchBuilder.BuildBench(benchParameters);  var computerInfo = new ComputerInfo();  var usedMemory = (computerInfo.TotalPhysicalMemory  - computerInfo.AvailablePhysicalMemory)  \* gigabyteInByte;  var logEntry = $"{++count}\t{stopWatch.Elapsed:hh\\:mm\\:ss}\t{usedMemory}";  streamWriter.WriteLine(logEntry);  Console.WriteLine(logEntry);  streamWriter.Flush();  stopWatch.Reset();  } |

Рисунок 6.4 – Код для нагрузочного тестирования

На рисунке 6.5 и 6.6 представлено тестирование зацикленного построения модели со следующими параметрами:

* длина скамьи: 150 см;
* высота сиденья: 20 см;
* высота ножки: 30 см;
* длина ножки: 40 см;
* ширина сиденья: 40 см;

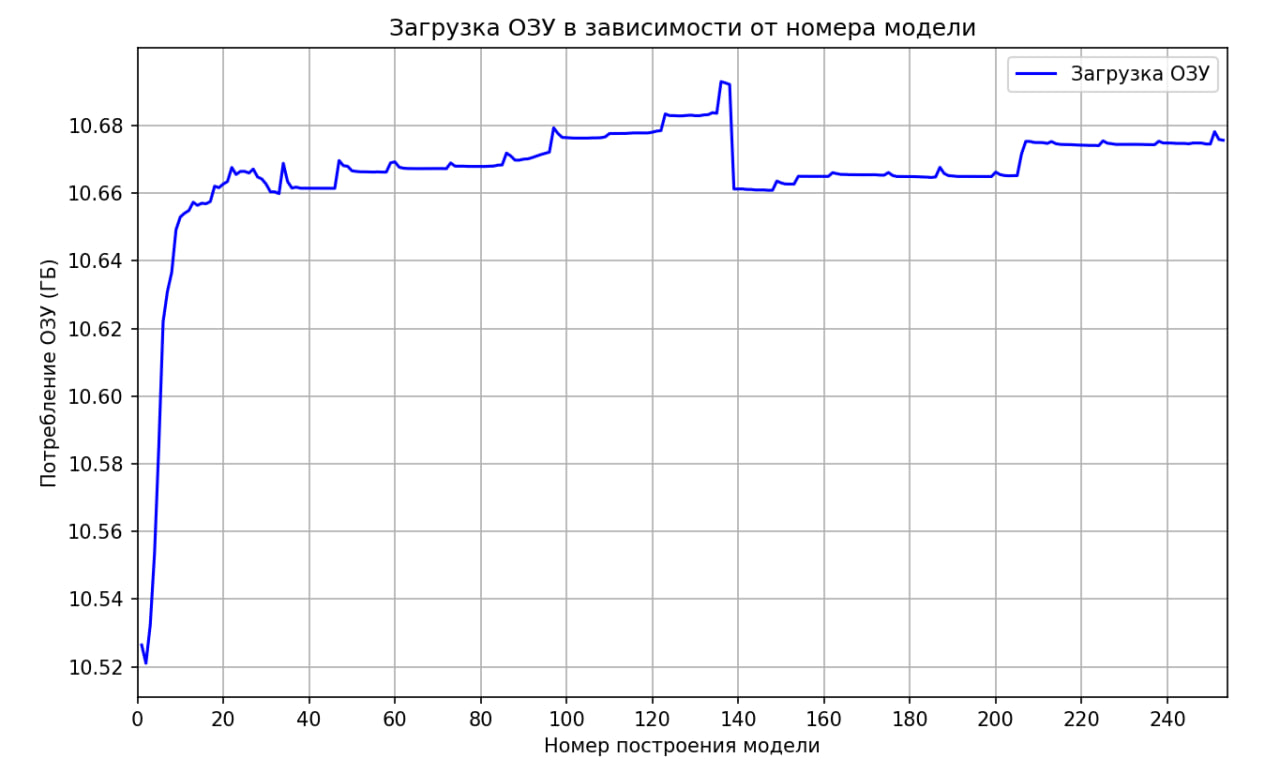


Рисунок 6.5 – График зависимости количества потребляемой памяти от количества 3D-моделей со средними параметрами



Рисунок 6.6 – График гистограммы построения модели со средними параметрами

Исходя из представленных графиков, можно сделать следующие выводы:

Во-первых, потребление оперативной памяти при построении моделей постепенно увеличивается по мере роста номера модели. Однако в определенные моменты наблюдаются скачки и резкие падения потребления памяти, что может быть связано с процессами очистки памяти или особенностями загрузки данных. Это может указывать на использование механизма управления памятью, например, периодической очистки кэша или работы файлов подкачки.

Во-вторых, анализ времени построения моделей показывает, что оно распределено относительно равномерно, без значительных всплесков. Однако в последнем интервале (26–27 секунд) наблюдается заметный рост количества моделей, что может указывать на наличие сложных моделей, требующих больше времени для обработки. В целом, разброс времени постройки моделей относительно небольшой, что свидетельствует о стабильности работы системы.

В-третьих, несмотря на постепенный рост загрузки оперативной памяти, процесс построения моделей не сталкивается с резкими падениями производительности, что говорит о достаточно эффективном управлении ресурсами. Однако в отдельных случаях возможны кратковременные задержки, связанные с перераспределением памяти.

Таким образом, на основе представленных данных можно заключить, что система обладает устойчивым механизмом управления памятью и построением моделей, но в некоторых случаях возможны скачки потребления оперативной памяти, которые могут быть связаны с очисткой или перераспределением ресурсов.

# Заключение

В ходе выполнения лабораторных работ были изучены предметная область проектирования, предмет проектирования, аналоги предмета проектирования, API, функциональное и нагрузочное тестирование и на основании полученных данных была спроектирована UML диаграмма классов, разработан плагин для создания 3D моделей «Скамья» в САПР КОМПАС-3D, и проведено функциональное и нагрузочное тестирование плагина.

# Список источников

1. Автоматизация вычислительных процедур в прикладных задачах инженерного проектирования [Электронный ресурс]. – URL: https://scienceforum.ru/2014/article/2014000201 (дата обращения: 10.12.2024).

2. Visual Studio [Электронный ресурс]. – URL: <https://visualstudio.microsoft.com/ru/> (дата обращения: 10.12.2023).

3. КОМПАС-3D. Официальный сайт САПР КОМПАС [Электронный ресурс]. — Режим доступа: https://kompas.ru/ (дата обращения 22.10.2024).

4. КОМПАС-3D для разработчиков [Электронный ресурс]. – URL: https://kompas.ru/solutions/developers/ (дата обращения: 10.12.2024).

5. NUnit [Электронный ресурс]. – URL: <https://nunit.org/> (дата обращения: 10.12.2024).

6. Windows Forms [Электронный ресурс]. – URL: https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/controls/how-to-create-a-multipane-user-interface-with-windows-forms?view=netframeworkdesktop-4.8 (дата обращения: 10.12.2024).

7. САПР [Электронный ресурс]. − Режим доступа https://secrets.tinkoff.ru/glossarij/sapr/ (дата обращения 20.10.2024)

8. ScetchUp [Электронный ресурс] – режим доступа <https://www.sketchup.com/en/products/sketchup-for-web> (дата обращения 20.10.24)

9. UML [Электронный ресурс]. − Режим доступа <https://www.uml-diagrams.org> (дата обращения 25.10.2024)

10.Функциональное тестирование [Электронный ресурс]. – URL: https://daglab.ru/funkcionalnoe-testirovanie-programmnogo-obespechenija/ (дата обращения: 25.12.2024).

11. Модульное тестирование [Электронный ресурс]. – URL: <https://habr.com/ru/post/169381/> (дата обращения: 25.12.2024).

12. Нагрузочное тестирование [Электронный ресурс]. – URL: https://learn.microsoft.com/ru-ru/visualstudio/test/quickstart-create-a-load-test-project?view=vs-2022 (дата обращения: 25.12.2024).