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 «ТОПОР» ДЛЯ «КОМПАС−3D»**

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

по дисциплине

«Основы разработки САПР» (ОРСАПР)

Выполнил:

студент гр. 581

\_\_\_\_\_\_\_\_\_ Устинов В.А.

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

Проверил:

к.т.н., доцент каф. КСУП \_\_\_\_\_\_\_\_\_\_ Калентьев А.А.

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

**РЕФЕРАТ**

Пояснительная записка, 38 страницы, 23 рисунков, 12 таблиц, 12 источников.

Ключевые слова: САПР, КОМПАС−3D, Плагин для САПР, Плагин топора, C#, Windows Forms.

Объектом исследования являются технологии разработки плагинов для САПР.

Предметом исследования является применение технологий разработки плагинов, для автоматизации построения топоров разных размеров и параметров в САПР КОМПАС−3D.

Цель работы: создание программы для автоматизации построения топора в САПР КОМПАС−3D.

Для создания использовались Microsoft Visual Studio 2022 (Windows Forms), .NET Framework 4.7.2, NUnit 3.14.0, NUnit3TestAdapter 3.17.0, StyleCop.Analyzers 1.1.118, StyleCop.Analyzers.Unstable 1.2.0.556, ReSharper, Fine Code Coverage, GitHub.

В результате работы было создано приложение Windows Forms, взаимодействующее с САПР КОМПАС−3D.

Областью применения являются предприятия, связанные с моделированием топора.

**СОДЕРЖАНИЕ**

[1 ВВЕДЕНИЕ 4](#_Toc23763)

[2 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ 5](#_Toc22434)

[3 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ 7](#_Toc31758)

[4 ВЫБОР ИНСТРУМЕНТОВ И СРЕДСТВ РЕАЛИЗАЦИИ 8](#_Toc363)

[5 НАЗНАЧЕНИЕ ПЛАГИНА 9](#_Toc17893)

[6 ОБЗОР АНАЛОГОВ 9](#_Toc13689)

[7 ОПИСАНИЕ РЕАЛИЗАЦИИ 11](#_Toc16292)

[8 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЕЙ 20](#_Toc9887)

[9 ТЕСТИРОВАНИЕ ПЛАГИНА 24](#_Toc28841)

[9.1 Функциональное тестирование 24](#_Toc10064)

[9.2 Модульное тестирование 30](#_Toc20472)

[9.3 Нагрузочное тестирование 32](#_Toc3928)

[10 ЗАКЛЮЧЕНИЕ 35](#_Toc1958)

[11 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 36](#_Toc5152)

**1 ВВЕДЕНИЕ**

САПР – организационно−техническая система, входящая в структуру проектной организации и осуществляющая проектирование при помощи комплекса средств автоматизированного проектирования [1].

API (Application Programming Interface) — набор правил и протоколов, с помощью которых различные программные приложения могут взаимодействовать друг с другом и обмениваться данными, повышая тем самым функциональность и эффективность работы [2].

Для разработки плагина для САПР прежде всего необходимо выбрать объект проектирования, подходящую для выбранного объекта САПР, средства разработки плагина (язык программирования и дополнительные средства разработки, выбор может быть основан на наличии API для выбранной САПР на конкретном языке).

Плагин автоматизации построения топора необходим и может быть использован на предприятиях, занимающихся моделированием топоров, поскольку он упростит процесс моделирования и снизит нагрузку на моделлеров.

**2 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ**

Этапы проведения работ по разработке плагина «ТОПОР» для САПР «Компас 3D» приведены в таблице 2.1.

Таблица 2.1 − Этапы проведения работ по разработке плагина «Топор» для САПР «Компас 3D».

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Этап | Состав работ | Наименование документа | | Обозначение | Разработано согласно | | Сроки выполнения |
| 1 | Создание технического задания | Техническое задание | | − | ГОСТ 34.602−2020 | | Не позднее 8.10.2024 |
| 2 | Создание проекта системы | Проект системы | | − | ОС ТУСУР 01−2021 | | Не позднее 29.10.2024 |
| 3 | Реализация плагина | Программный код | | − | RSDN Magazine #1−2004 | | Не позднее 10.12.2024 |
| Документ с тремя вариантами дополнительной функциональности плагина для согласования | |
| Модульные тесты | |
| 4 | 1. Доработка плагина 2. Создание пояснительной записки | | Программный код | − | 1. RSDN Magazine #1−2004 2. ОС ТУСУР 01−2021 | Не позднее 31.12.2024 | | |
| Модульные тесты |
| Пояснительная записка |

На этапе реализации плагина потребовалось внесение корректировок, что привело к изменению конечной версии UML−диаграммы классов. Следует отметить, что такие изменения не являются редкостью: в реальных проектах они зачастую обусловлены изменяющимися требованиями и предпочтениями заказчиков, тогда как в данном случае их причиной выступила недостаточная проработанность диаграмм разработчиками.

Кроме того, на этапе выбора объекта проектирования были недостаточно глубоко изучены особенности API КОМПАС−3D. Несмотря на то, что это не создало существенных сложностей при написании кода, возникли значительные временные затраты, превышающие изначальные прогнозы. Из данного опыта можно сделать вывод о необходимости более тщательного изучения инструментов разработки на этапе планирования, чтобы минимизировать риски, связанные с недостаточной подготовленностью разработчиков.

# 3 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ

Топор − рубящее орудие, чаще короткое древковое и имеющее рабочее лезвие, расположенное вдоль рукоятки (топорища) [3].

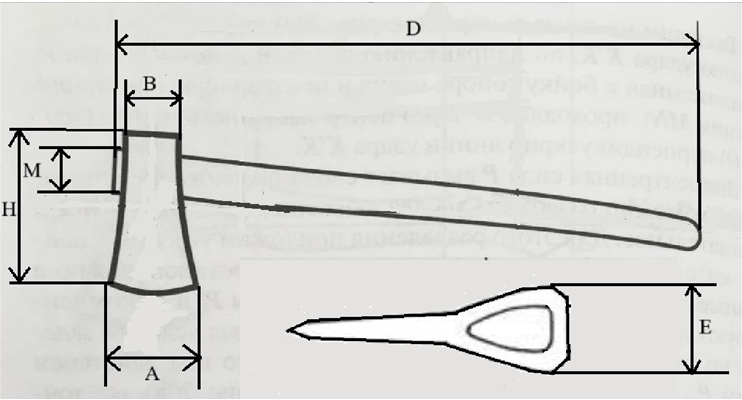


Рисунок 3.1 − Модель топора

**Изменяемые параметры для предмета проектирования** (также все обозначения показаны на рисунке 3.1):

− Длина лезвия топора А (100−300мм);

− Длина топорища H (80−150мм);

− Длина ручки топора D (больше в 3 раза длинны лезвия А);

− Длинна обуха B (Составляет 80% от А);

− Ширина рукояти M (составляет 20% от А);

− Ширина топорища E (составляет 30% от длинны обуха В).

**4 ВЫБОР ИНСТРУМЕНТОВ И СРЕДСТВ РЕАЛИЗАЦИИ**

При создании плагина использовались следующие инструменты:

− WindowsForms и .NET Framework 4.7.2;

− GitHub;

− ReSharper;

− Fine Code Coverage;

− StyleCop.Analyzers 1.1.118;

− StyleCop.Analyzers.Unstable 1.2.0.556;

− NUnit 3.14.0;

− NUnit3TestAdapter 3.17.0.

Плагин был создан на технологии Windows Forms, поддерживающей широкий набор функций для разработки приложений, включая элементы управления, графику, привязку данных и ввод пользователя [4], а также .NET Framework 4.7.2, программной платформе основанной на сервероцентрической модели.

GitHub − платформа с возможностями хранения, распространения и совместной работы над написанием кода. Git − система управления версиями, которая интеллектуально отслеживает изменения в файлах [5].

ReSharper − расширение для Microsoft Visual Studio, помогающее программировать эффективнее. Позволяет исследовать, улучшать, писать и обслуживать код [6].

Fine Code Coverage − расширение для Microsoft Visual Studio, визуализирующий покрытие кода модульными тестами [7].

StyleCop − средство для контроля кода, автоматически находящее синтаксические ошибки.

# 5 НАЗНАЧЕНИЕ ПЛАГИНА

Назначение разрабатываемого плагина обусловлено быстрым моделированием топоров разных видов. Благодаря данному расширению, производители топоров смогут наглядно рассмотреть спроектированную модель, при необходимости перестроить под необходимые им параметры.

**6 ОБЗОР АНАЛОГОВ**

Первым аналогом является приложения «Teapot Plugin» [8] для Компас−3D, который предназначен для создания различных форм чайников. Он демонстрирует, как можно расширить функциональность КОМПАС−3D для создания 3D−моделей через пользовательский интерфейс плагина. Плагин может служить примером того, как организовать создание объектов с помощью параметров, передаваемых из пользовательского интерфейса в процесс моделирования. Данный аналог является прямым для разрабатываемого плагина «Топор». Интерфейс взаимодействия представлен на рисунке 6.1.



Рисунок 6.1 − Интерфейс приложения «Teapot Plugin»

Вторым аналогом является плагин для генерации модели клинка в КОМПАС−3D. Проект написан на C# и ориентирован на моделирование лезвий [9]. Плагин "Клинок" для КОМПАС−3D предназначен для автоматизированного создания модели клинка. Он разработан на языке C# и интегрируется в среду КОМПАС−3D, предоставляя пользователю инструменты для параметрического проектирования лезвий. Проект находится в открытом доступе на GitHub, что позволяет изучить его и проанализировать любому желающему.

Пользовательский интерфейс представлен на рисунке 6.2.



Рисунок 6.2 − Интерфейс приложения «Плагин для Компас 3D “Клинок”»

# 7 ОПИСАНИЕ РЕАЛИЗАЦИИ

UML − это стандартный язык визуального моделирования, предназначенный для следующего использования:

− моделирование бизнеса и подобных процессов;

− анализ, проектирование и внедрения программных систем.

UML − это общий язык для бизнес−аналитиков, архитекторов и разработчиков программного обеспечения, используемый для описания, спецификации, проектирования и документирования существующих или новых бизнес−процессов, структуры и поведения артефактов программных систем. [10]

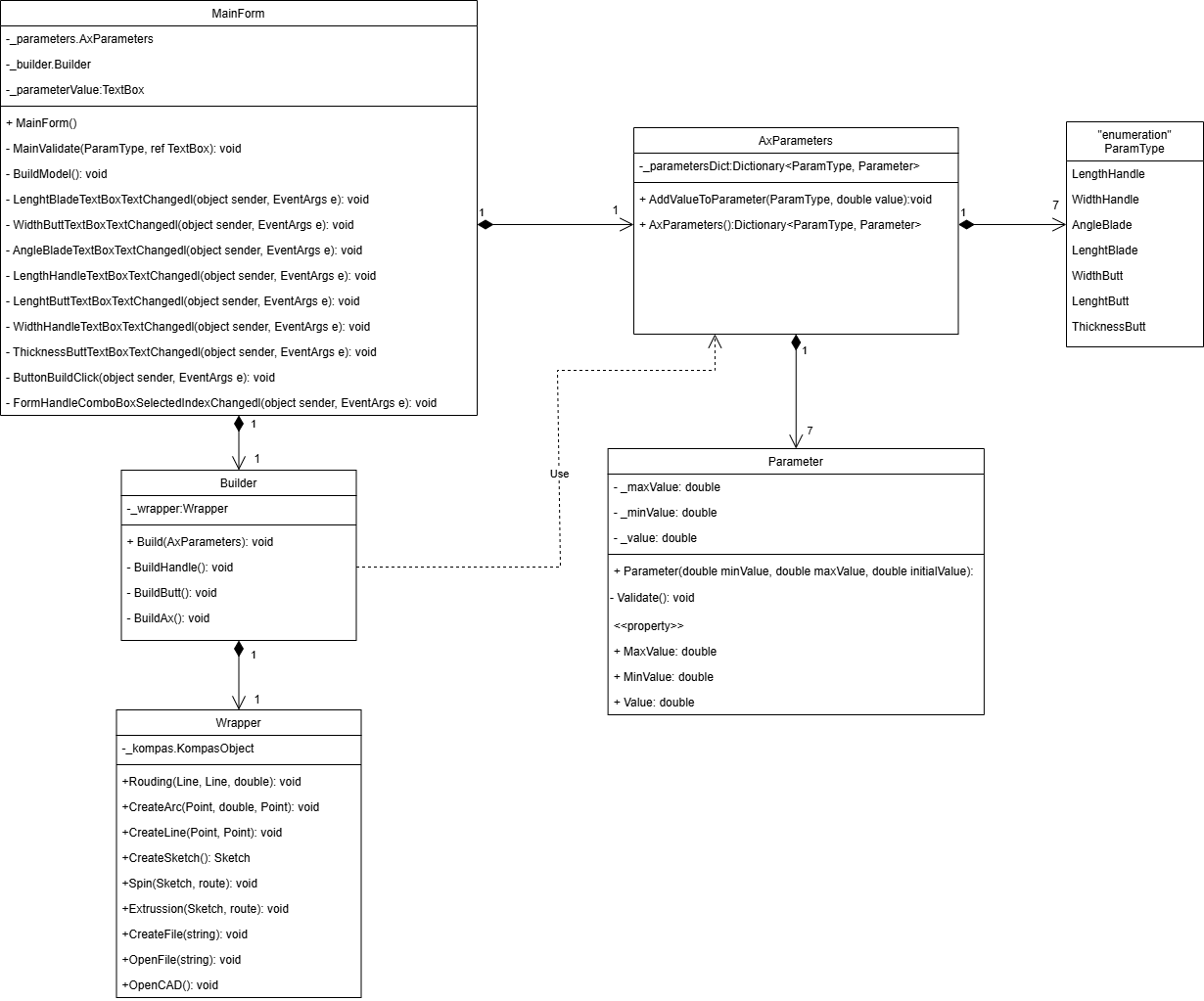
UML диаграмма классов до и после реализации плагина «Топор» представлена на рисунках (7.1, 7.2).  


Рисунок 7.1 − UML диаграмма классов до реализации плагина «Топор»

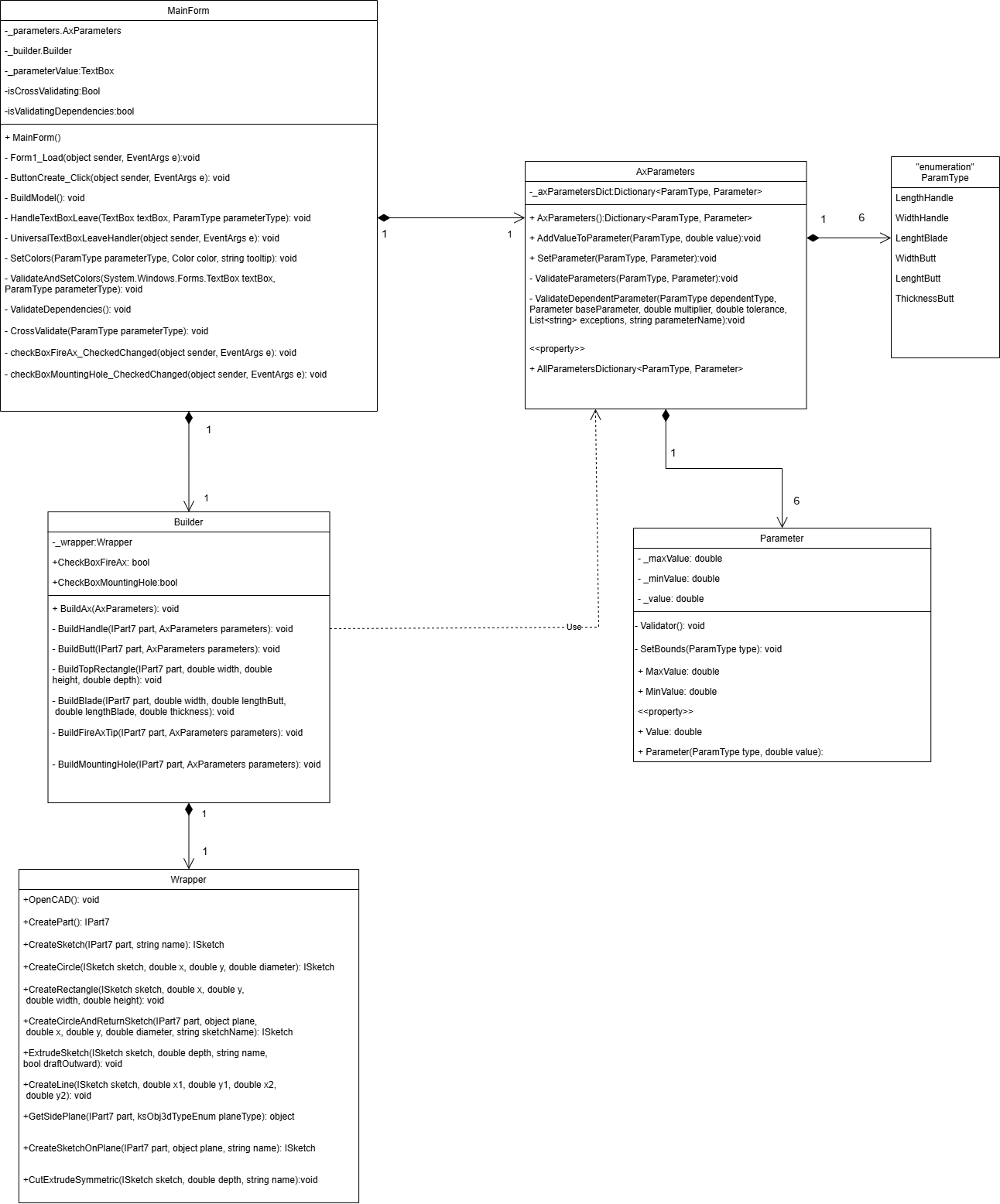


Рисунок 7.2 − UML диаграмма классов после реализации плагина «Топор»

В таблицах ниже представлена информация о свойствах и методах каждого из классов (7.1-7.10).

Таблица 7.1 − Свойства класса MainForm

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_parameters | AxParameters | Поле хранящее в себе объект класса Parameters |
| \_builder | Builder | Поле хранящее в себе объект класса Builder |
| \_isCrossValidating | Bool | Указывает, выполняется ли в данный момент перекрестная валидация |
| \_isValidatingDependencies | Bool | Указывает, выполняется ли в данный момент проверка зависимостей |

Таблица 7.2 − Методы класса MainForm

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| Form1\_Load | object sender,  EventArgs e | Инициализация ряда параметров при загрузке формы |
| MainForm | − | Конструктор MainForm |
| ButtonCreate\_Click | object sender,  EventArgs e | Обработчик нажатия на кнопку "Создать" |
| HandleTextBoxLeave | TextBox textBox, ParamType parameterType | Универсальный обработчик выхода из текстбокса. Выполняет валидацию и обновляет зависимости |
| UniversalTextBoxLeaveHandler | object sender, EventArgs e | Универсальный обработчик события выхода из текстового поля |
| SetColors | ParamType parameterType, Color color, string tooltip | Метод для установки цвета и подсказки для текстбокса в зависимости от типа параметра |
| ValidateAndSetColors | TextBox textBox, ParamType parameterType | Универсальный метод валидации, объединяет первичную и вторичную проверки |
| ValidateDependencies | − | Выполняет валидацию зависимых параметров, связанных с длиной лезвия, длиной обуха, а также других параметров ножа. Метод предотвращает повторный запуск валидации, если она уже выполняется. |
| CrossValidate | ParamType parameterType | Выполняет перекрестную валидацию параметров в зависимости от их типа. |

Продолжение таблицы 7.2

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| checkBoxFireAx\_CheckedChanged | object sender, EventArgs e | Обработчик изменения состояния чекбокса "Пожарный топор". |
| checkBoxMountingHole\_CheckedChanged | object sender, EventArgs e | Обработчик изменения состояния чекбокса "Отверстие для подвеса". |

Таблица 7.3 − Свойства класса AxParameters

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_axParameters | Dictionary<ParameterType, Parameter> | Словарь, содержащий параметры модели |
| AllParameters | Dictionary<ParamType, Parameter> | Свойство для доступа ко всем параметрам модели |

Таблица 7.4 − Методы класса AxParameters

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| SetParameter | ParamType parameterType, Parameter parameter | Устанавливает новый параметр в словарь |
| ValidateParameters | ParamType parameterType, Parameter parameter | Валидирует зависимости заданного параметра |
| ValidateDependentParameter | ParamType dependentType, Parameter baseParameter, double multiplier, double tolerance, List<string> exceptions, string parameterName | Валидирует зависимый параметр на основе базового параметра |

Таблица 7.5 − Свойства класса Builder

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_wrapper | Wrapper | Хранит в себе объект обёртки API |

Продолжение таблицы 7.5

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| IsFireAx | bool | Флаг для активации функциональности "Пожарный топор" |
| IsMountingHole | bool | Флаг для активации функциональности "Отверстие для подвеса" |

Таблица 7.6 − Методы класса Builder

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| Builder | − | Инициализирует объект обертки для работы с API Компас |
| BuildAx | AxParameters parameters | Метод для построения топора с заданными параметрами |
| BuildHandle | IPart7 part, AxParameters parameters | Метод для построения рукояти топора |
| BuildButt | IPart7 part, AxParameters parameters | Метод для построения обуха топора |
| BuildTopRectangle | IPart7 part, double width, double height, double depth | Метод для построения прямоугольника сверху обуха |

Продолжение таблицы 7.6

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| BuildBlade | IPart7 part, double width, double lengthButt, double lengthBlade, double thickness | Метод для построения рубящей части топора |
| BuildFireAxTip | IPart7 part, AxParameters parameters | Метод для построения наконечника пожарного топора |
| BuildMountingHole | IPart7 part, AxParameters parameters | Метод для создания отверстия для подвеса топора |

Таблица 7.7 − Свойства класса Parameter

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_maxValue | double | Максимально допустимое значение параметра |
| \_minValue | double | Минимально допустимое значение параметра |
| \_value | double | Значение параметра |

Таблица 7.8 − Методы класса Parameter

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| Value | − | Get и Set для Value |
| MinValue | − | Get для MinValue |
| MaxValue | − | Get для MaxValue |
| Parameter | ParamType type, double value | Конструктор, который принимает тип параметра и автоматически задает границы |

Продолжение таблицы 7.8

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| SetBounds | ParamType type | Устанавливает границы параметра на основе типа |
| Validator | − | Валидация значения |

Таблица 7.9 – Свойства класса Wrapper

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_kompas | KompasObject | Поле, хранящее в себе экземпляр программы Компас |
| \_part | ksPart | Поле, хранящее в себе основную модель |
| \_sketchEntity | ksEntity | Поле, хранящее в себе текущий эскиз |
| \_plane | ksEntity | Поле, хранящее в себе текущий вид |

Таблица 7.10 – Методы класса Wrapper

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Выходные параметры | Описание |
| CreatePart | − | Ipart7 | Метод для создания части в 3D документе |
| CreateSketch | Ipart7 part, string name | Isketch | Метод для создания эскиза на заданной части |
| CreateRectangle | Isketch sketch, double x, double y, double width, double height | − | Метод для создания прямоугольника в эскизе |
| CreateCircle | Isketch sketch, double x, double y, double diameter | − | Метод для создания круга в эскизе |

Продолжение таблицы 7.10

|  |  |  |  |
| --- | --- | --- | --- |
| CreateCircleAndReturnSketch | IPart7 part, object plane, double x, double y, double diameter, string sketchName | ISketch | Создает эскиз на указанной плоскости, добавляет в него окружность и возвращает созданный эскиз |
| ExtrudeSketch | −ISketch sketch, double depth, string name, bool draftOutward | − | Метод для экструзии эскиза |
| CreateLine | ISketch sketch, double x1, double y1, double x2, double y2 | − | Метод для создания линии в заданном эскизе |
| GetSidePlane | IPart7 part, ksObj3dTypeEnum planeType | object | Получение стандартной плоскости на основе типа |
| CreateSketchOnPlane | IPart7 part, object plane, string name | ISketch | Создает новый эскиз на указанной плоскости в заданной детали |
| CutExtrudeSymmetric | ISketch sketch, double depth, string name | − | Вырезать выдавливанием симметрично из указанного эскиза |

В отличии от диаграммы классов проекта системы диаграмма классов после реализации плагина имеет следующие отличия:

1. В MainForm появились новые поля isCrossValidating и isValidatingDependencies. Обработчики были событий TextBoxTextChanged были заменены на TextBoxLeave. Были добавлены обработчики событий checkboxFireAx\_checkedChange и checkBoxMountingHole\_CheckedChanged являющиеся модификациями для топора. А также добавлены методы для выполнения валидации ValidateDependencies, ValidateAndSetColors, SetColors,   
   CrossValidate.
2. В классе AxParameters были добавлены методы валидации ValidateParameters, ValidateDependentParameter, а также метод SetParameter.
3. В класс Parameter был добавлен метод SetBounds и отредактирован конструктор Parameter, а также поля Min и MaxValue стали доступными только на чтение.
4. В класс Builder были добавлены поля CheckBoxFireAx, CheckBoxMountingHole. Были обновлены методы BuildHandle, BuildButt, BuildAx. А также были добавлены новые методы BuildTopRectangle, BuildBlade, BuildFireAxTip, BuildMountingHole.
5. В класс Wrapper были удалены методы OpenFile, CreateFile, Spin, CreateARC, Rouding. Обновлены старые методы CreateSketch, CreateLine, ExtrudeSketch и добавлены новые методы CreateRectangle, CreateCircleAndReturnSketch, ExtrudeSketch, CreateSketchOnPlane, GetSidePlane, CutExtrudeSymmetric.

Описанные выше изменения были внесены по ходу создания плагина, его модификации, а также при детальном изучении и подходе при построении топора.

Решено было отказаться от угла наклона лезвия в виду его некорректной реализации в контексте ручки топора в Компас 3Д при использовании API7.

Также часть изменений связана с дополнительной функциональностью разработанной в рамках 5 лабораторной, они затронули MainForm, Builder. (на UML−диаграмме).

**8 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЕЙ**

При запуске приложения открывается форма для заполнения параметров объекта (рисунок 8.1).

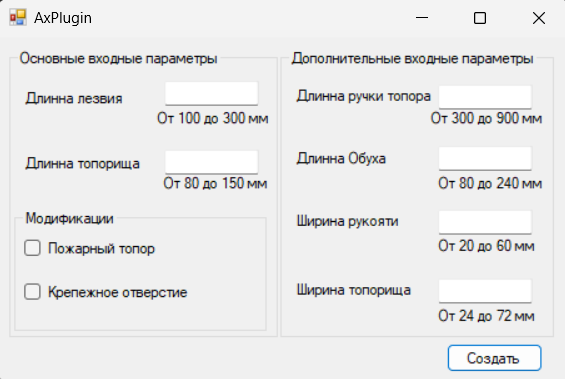


Рисунок 8.1 – Начальная форма в момент запуска приложения

При нажатии на кнопку с незаполненными или неверно заполненными полями не будет происходить ничего, при правильном же заполнении откроется КОМПАС−3D и начнётся построение модели по заданным параметрам.   
 При наведении на незаполненное поле выведется подсказка по заполнению (рисунок 8.2).

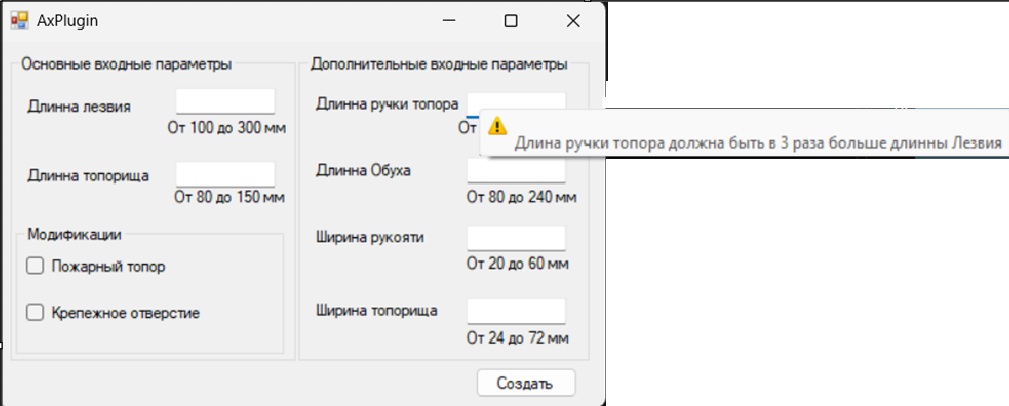


Рисунок 8.2 – Подсказка по заполнению

При неверном заполнении поля (выход за допустимые пределы) текстбокс будет подсвечен красным, а текст подсказки будет более точным (рисунок 8.3).

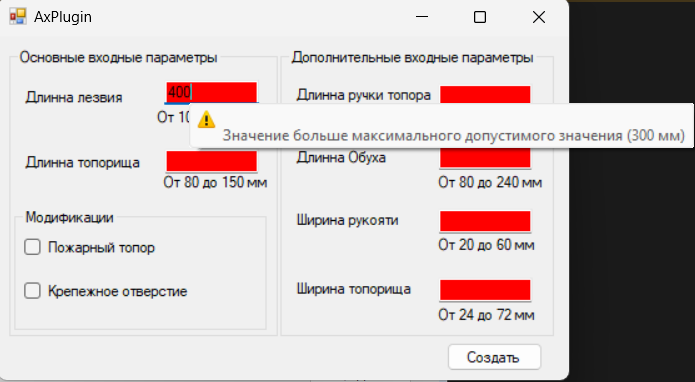


Рисунок 8.3 – Длина лезвия выходит за максимальные пределы (больше 300)

При этом, некорректно заполненные (непрошедшие собственную валидацию) или незаполненные поля не будут мешать возможно корректному заполнению зависимых от них полей (рисунок 8.4).

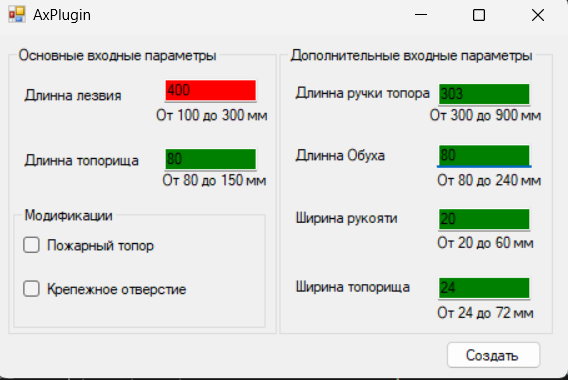


Рисунок 8.4 – Ошибка в собственной валидации длины лезвия не влияет на её длину обуха, ручки, ширины топора.

Значение 303 является допустимым (т.к. (100−1) \* 3 максимально допустимое значение для длинны ручки равняется 303) и подсвечивается зелёным. Помимо этого, можно заметить, что при наведении на корректно заполненный текстбокс не выводится никакая подсказка.

Помимо собственной ошибки также может быть вызвана ошибка в зависимых параметрах. Для решения каждой из таких ошибок текст подсказки меняет своё значение и выдаёт рекомендуемые к заполнению параметры для пользователя. Пользователь может изменить значение для любого из зависимых параметров для получения корректных результатов. При наличии ошибок в нескольких зависимых параметрах от одного у пользователя есть два пути решения:

1. Изменить сначала основной параметр, подходящий под хотя бы одну из валидаций, а после изменить оставшийся зависимый от него параметр.
2. Изменить оба зависимых от основного параметра.

К сожалению, система не будет настолько подробно описывать вызванные ошибки, поэтому пользователю придётся самому принимать решение.  
 Помимо этого при появлении в текстбоксе некорректных символов (буквы, символы) – текстбокс приобретает красный цвет и подсвечивается что строка имеет неверный формат.

Также пользователю доступен выбор значений для двух чекбоксов, что будет влиять на форму Топора.

**9 ТЕСТИРОВАНИЕ ПЛАГИНА**

## 9.1 Функциональное тестирование

Во время использования плагина, плагин обрабатывает ошибки следующим образом.

На рисунках 9.1 и 9.2 представлен результат обработок ошибок системой для зависимых параметров длина и диаметр ручки.

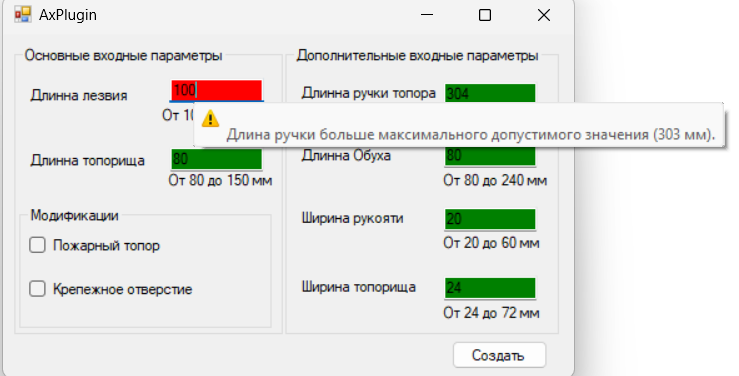


Рисунок 9.1 – Ошибка валидации зависимых параметров длины ручки и диаметра ручки

При ошибке в валидации зависимых параметрах основной зависимый параметр приобретает красный цвет и стандартные подсказка в нем изменяется на подсказку для получения корректных значений. Доказательства правильности выведенных подсказок представлены на рисунках 9.2 и 9.3.

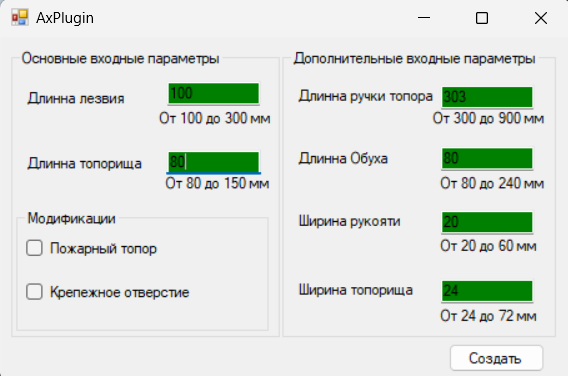


Рисунок 9.2 – Уменьшение длинны ручки до рекомендованных 303 мм

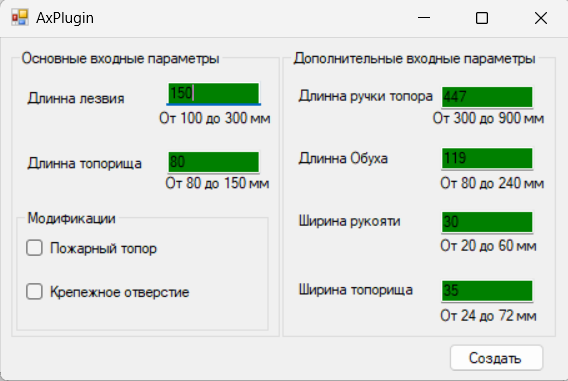


Рисунок 9.3 – Увеличение длины ручки до рекомендованных 447 мм

Ещё один вариант валидации возможен при некорректности сразу в нескольких связанных параметрах (рисунок 9.4)

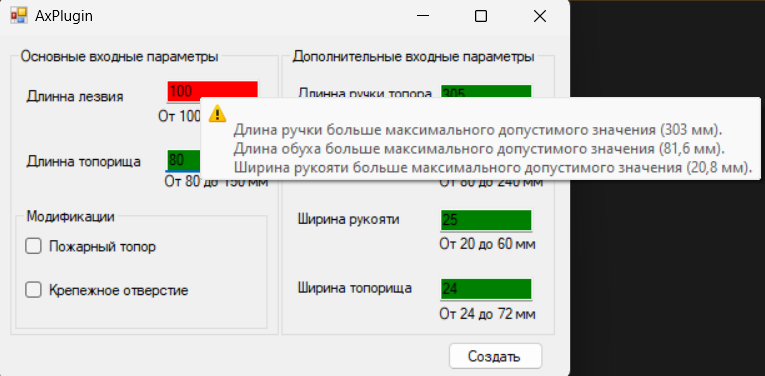


Рисунок 9.4 – Ошибка в валидации длины ручки и длинны обуха, а также ширины рукояти.

К сожалению, при такой ошибке валидации пользователь должен сам принимать более подходящие ему решения.

На рисунке 9.5 представлено заполнение формы минимальными параметрами.

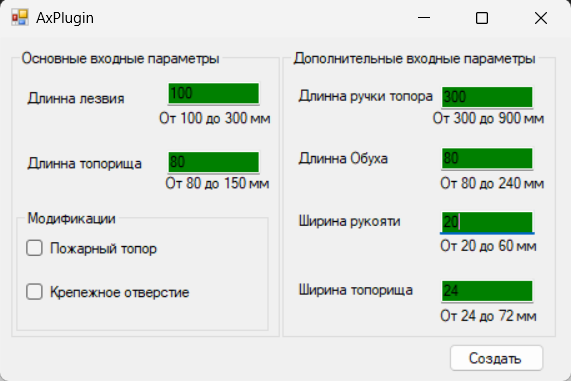


Рисунок 9.5 – Минимальные параметры

На рисунке 9.6 представлен результат построения модели с минимальными параметрами.

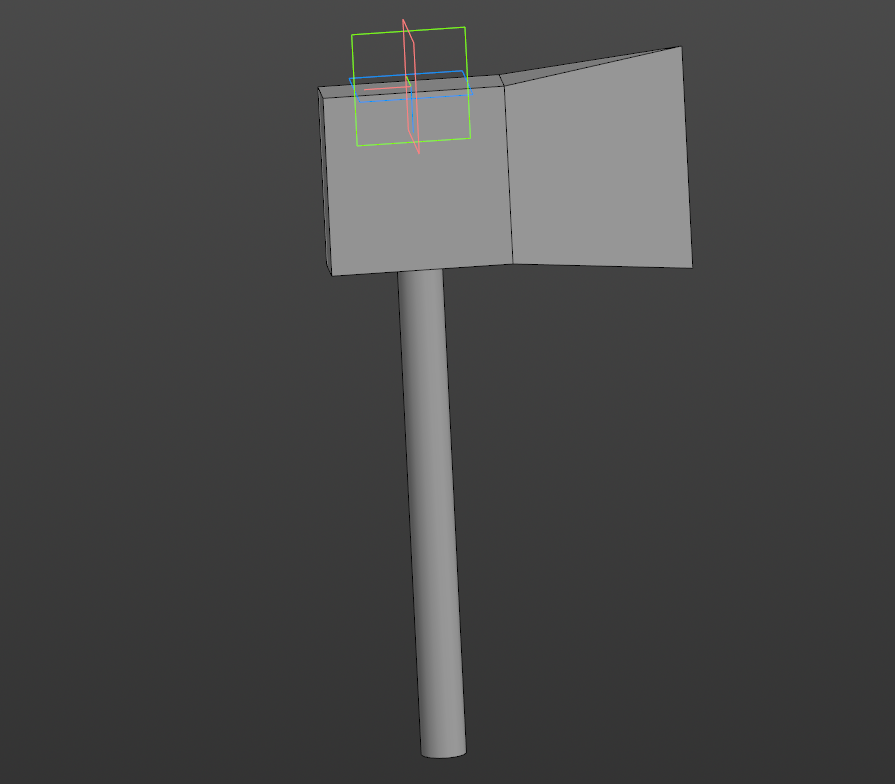


Рисунок 9.6 – Модель по минимальным параметрам

На рисунке 9.7 представлено заполнение формы максимальными параметрами.

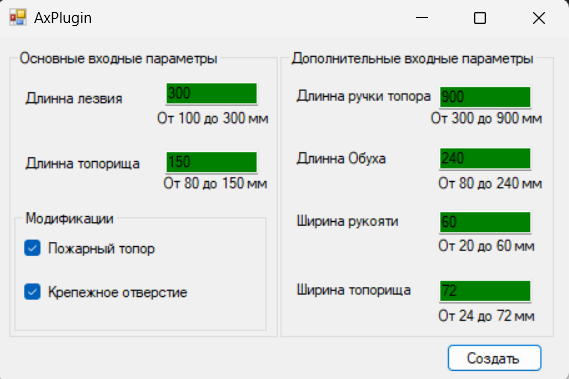


Рисунок 9.7 – Максимальные параметры

На рисунке 9.8 представлен результат построения модели с максимальными параметрами.

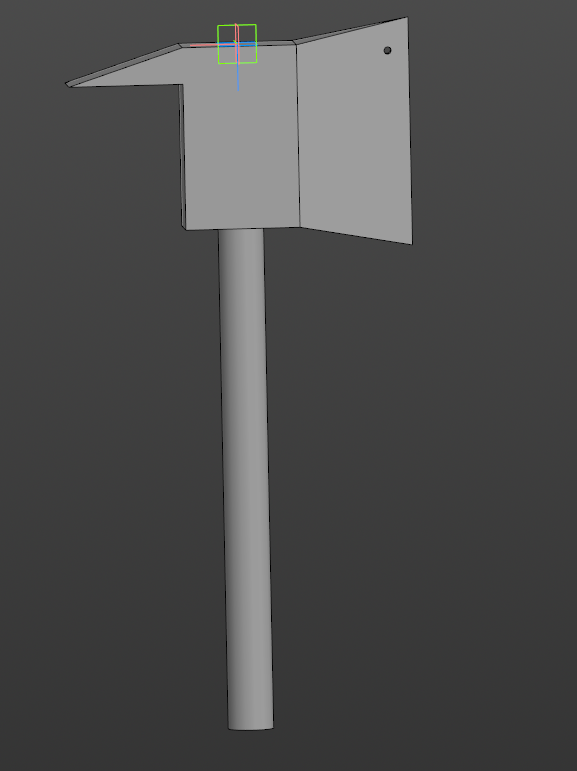


Рисунок 9.8 – Модель по максимальным параметрам

На рисунке 9.9 представлено заполнение формы стандартными параметрами.

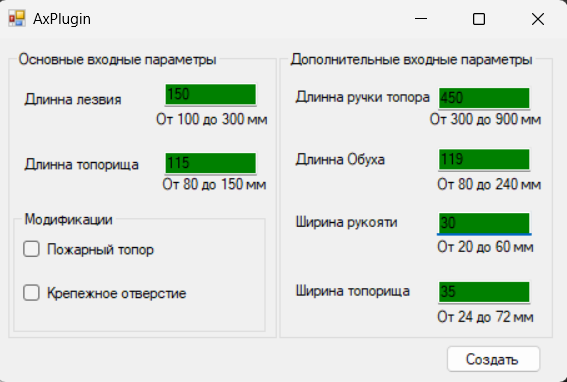


Рисунок 9.9 – Стандартные параметры

На рисунке 9.10 представлен результат построения модели с стандартными параметрами.

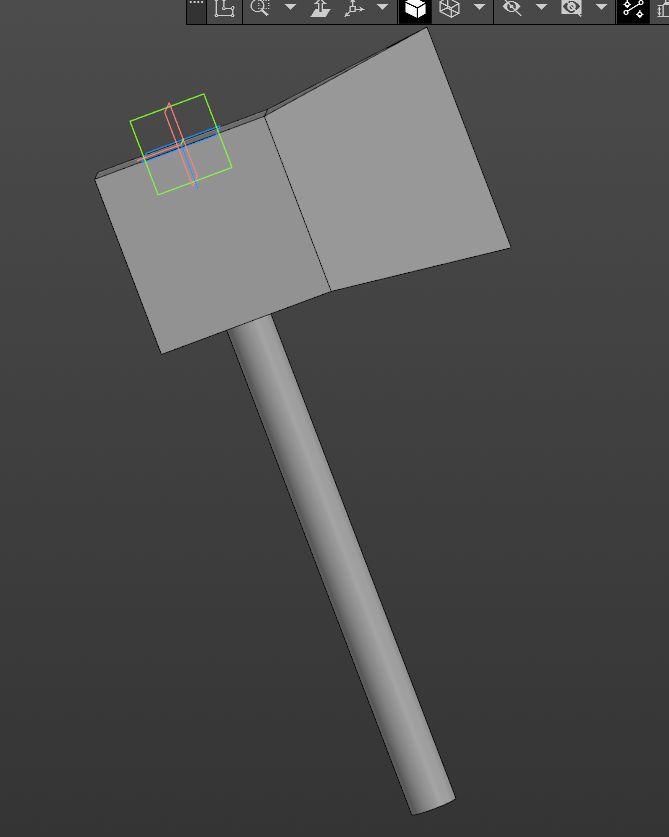


Рисунок 9.10 – Модель по стандартным параметрам

## 9.2 МОДУЛЬНОЕ ТЕСТИРОВАНИЕ

На рисунке 9.11 представлено количество написанных Unit−тестов, а также что их выполнение происходит корректно.

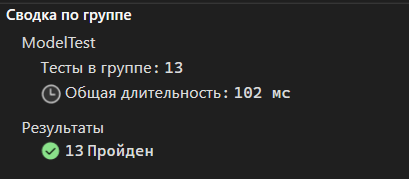


Рисунок 9.11 – Количество написанных Unit−тестов

Необходимо было написать тесты для 2−ух классов: Parameter и AxParameters. В таблице 9.1 представлены все написанные тесты и их описание.

Таблица 9.1 – Unit−тесты

|  |  |
| --- | --- |
| Название теста | Описание теста |
| Позитивный тест инициализации Parameter\_Constructor\_InitializesCorrectly | Проверяет работу конструктора Parameter |
| Позитивный тест геттера MinValue\_Get\_ReturnsCorrectValue | Проверяет работу get у MinValue |
| Позитивный тест геттера MaxValue\_Get\_ReturnsCorrectValue | Проверяет работу get у MaxValue |
| Позитивный тест геттера Value\_Get\_ReturnsCorrectValue | Проверяет работу get Value |
| Негативный тест сеттера Value\_Set\_LessThanMinValue\_ThrowsException() | Проверяет вызов исключения при Value<MinValue |
| Позитивный тест сеттера Value\_Set\_ValidValue\_SetsCorrectly | Проверяет работу set у Value |
|  | |

Продолжение таблицы 9.1

|  |  |
| --- | --- |
| Название теста | Описание теста |
| Негативный тест сеттера Value\_Set\_GreaterThanMaxValue\_ThrowsException | Проверяет вызов исключения при Value>MaxValue |
| Позитивный тест геттера AllParameters\_Get\_ReturnsNotNullDictionary | Проверяет, что после инициализации коллекция параметров не равна null. |
| Позитивный тест сеттера AllParameters\_Set\_ValidDictionary\_SetsCorrectly | проверяет, что свойство AllParameters корректно устанавливает словарь параметров. |
| Негативный тест геттера AllParameters\_Set\_NullValue\_ThrowsArgumentNullException | Проверяет, что при попытке присвоить null в AllParameters выбрасывается исключение ArgumentNullException. |
| Негативный тест  SetParameter\_NullParameter\_ ThrowsArgumentNullException | Этот тест проверяет, что метод SetParameter класса AxParameters выбрасывает исключение ArgumentNullException, если параметр, передаваемый в метод, равен null. |
| Позитивный тест ValidateParameter SetParameter\_ValidParameter\_SetsCorrectly | Проверяет, что метод SetParameter корректно устанавливает значение параметра. |

Продолжение таблицы 9.1

|  |  |
| --- | --- |
| Название теста | Описание теста |
| Позитивный тест SetParameter\_ValidDependentValue\_DoesNotThrowException | Этот тест проверяет, что метод SetParameter корректно добавляет в коллекцию зависимый параметр, который имеет допустимое значение, и при этом не выбрасывает исключений. |

На рисунке 9.12 также представлен скриншот плагина, измеряющего процент покрытия модульными тестами



Рисунок 9.12 – Результаты плагина

## 9.3 НАГРУЗОЧНОЕ ТЕСТИРОВАНИЕ

На рисунке 9.13 представлен график зависимости памяти ОЗУ от построения модели, а на рисунке 9.14 представлен график зависимости времени от построения модели.

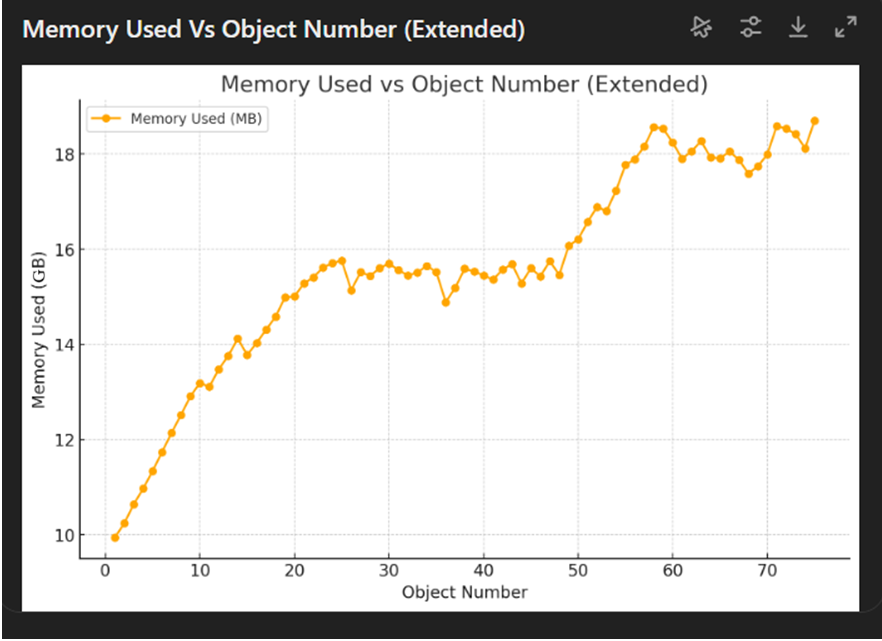


Рисунок 9.13 – График зависимости памяти ОЗУ от количества построенных моделей

Из графика 9.15 можно сделать вывод, что график показывает рост потребляемой памяти с каждым объектом, начиная от ~10 ГБ и приближаясь к 16 ГБ к 75−му объекту.

Достигнув 16 ГБ, система, вероятно, начнёт использовать виртуальную память, что приведёт к резкому снижению производительности и увеличению времени построения.

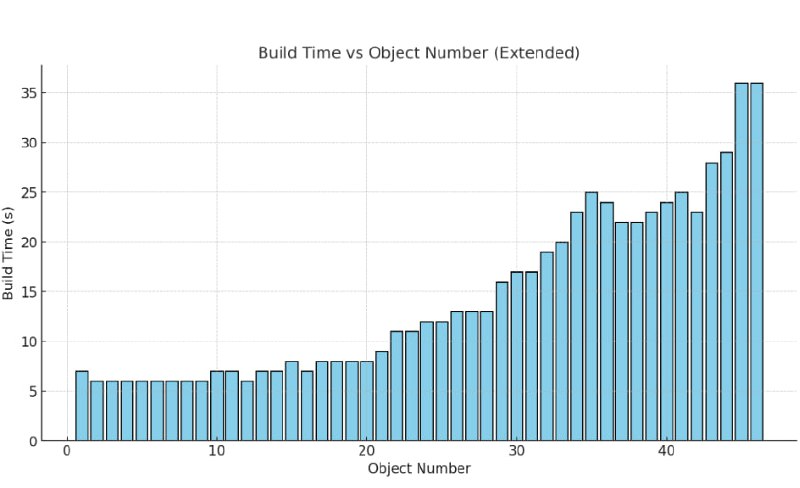


Рисунок 9.14 – График гистограммы построения модели

По графику можно сделать вывод, что основное время построения модели от 0 до 58 с. До примерно 30−го объекта время создания топора стабильно и держится в диапазоне 6–9 секунд, что свидетельствует об эффективной обработке объектов с одинаковыми параметрами.

После 30−го объекта наблюдается увеличение времени построения, достигающее более 25–36 секунд к 75−му объекту. Это может быть связано с накоплением промежуточных данных, влияющих на производительность, или с недостаточной оптимизацией алгоритмов.

**10 ЗАКЛЮЧЕНИЕ**

В ходе выполнения лабораторных работ был создан плагин для КОМПАС−3D, который автоматически генерирует модель топора на основе параметров, заданных пользователем. Каждая лабораторная работа была нацелена на достижение этой цели: выбор объекта моделирования и системы автоматизированного проектирования способствовал изучению соответствующих материалов, а техническое задание помогло уточнить направление работы. Проектирование системы требовало учитывать множество деталей, чтобы избежать крупных изменений кода в будущем, а написание кода включало работу с API и взаимодействие с пользовательским интерфейсом, использование инструментов стандартизации кода, таких как StyleCops, произвело положительное впечатление: код стал более понятным и удобным для дальнейшей работы.

**11 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. ГОСТ 23501.101−87 «Системы автоматизированного проектирования. Основные положения» (дата обращения 13.12.2024)
2. API [Электронный ресурс]. − Режим доступа [https://itglobal.com/ru−ru/company/glossary/api/](https://itglobal.com/ru-ru/company/glossary/api/) (дата обращения 28.09.2024)
3. Топор ГОСТ [Электронный ресурс]. − Режим доступа [https://zlatoust−air.ru/blog/sovety−pokupatelyam/topor−vidy−kharakteristiki−naznachenie−i−vybor/](https://zlatoust-air.ru/blog/sovety-pokupatelyam/topor-vidy-kharakteristiki-naznachenie-i-vybor/)

(дата обращения 18.09.2024)

1. Windows Forms [Электронный ресурс]. − Режим доступа [https://learn.microsoft.com/ru−ru/dotnet/desktop/winforms/overview/?view=netdesktop−9.0](https://learn.microsoft.com/ru-ru/dotnet/desktop/winforms/overview/?view=netdesktop-9.0) (дата обращения 13.12.2024)
2. Github [Электронный ресурс]. − Режим доступа [https://docs.github.com/ru/get−started/start−your−journey/about−github−and−git](https://docs.github.com/ru/get-started/start-your-journey/about-github-and-git) (13.12.2024)
3. ReSharper [Электронный ресурс]. − Режим доступа [https://www.jetbrains.com/ru−ru/resharper/](https://www.jetbrains.com/ru-ru/resharper/) (13.12.2024)
4. Fine Code Coverage [Электронный ресурс]. − Режим доступа <https://marketplace.visualstudio.com/items?itemName=FortuneNgwenya.FineCodeCoverage> (13.12.2024)
5. Teapot Plugin [Электронный ресурс]. − Режим доступа <https://github.com/kurocha/teapot> (дата обращения 29.09.2024)
6. Плагин для компас 3D “Клинок”. [Электронный ресурс]. − Режим доступа https://github.com/eskorpado/kompas−blade−plugin (дата обращения 05.10.2024)
7. UML [Электронный ресурс]. − Режим доступа [https://www.uml−diagrams.org/](https://www.uml-diagrams.org/) (дата обращения 07.10.2024)