Министерство науки и высшего образования Российской Федерации Федеральное государственное автономное образовательное учреждения высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

**РАЗРАБОТКА ПЛАГИНА «ЧАЙНИК ДЛЯ «KOMPAS–3D»**

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

по дисциплине

«Основы разработки САПР» (ОРСАПР)

Выполнил:

студент гр. 581

Хайбулин Н.С.

« » 2025 г.

Руководитель:

к.т.н., доцент каф. КСУП

Калентьев А.А.

« » 2025 г.

**РЕФЕРАТ**

Пояснительная записка, 37 страницы, 27 рисунков, 22 таблицы, 10 источников.

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

Объектом исследования являются технологии разработки плагинов для САПР.

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

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

Для создания использовались: Visual Studio 2022, WindowsForms, .NET Framework 4.7.2, GitHub и GitHub Desktop, ComputerInfo (версии 0.2.0), NUnit (версии 3.14), NUnit.Analyzers, NUnit3TestAdapter, Fine Code Coverage.

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

Областью применения являются предприятия, связанные с моделированием чайника.

**СОДЕРЖАНИЕ**

[1 ВВЕДЕНИЕ 4](#_Toc187484095)

[1.1 Информация о выбранной САПР 4](#_Toc187484096)

[1.2 Описание API 4](#_Toc187484097)

[2 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ 5](#_Toc187484098)

[3 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ 6](#_Toc187484099)

[4 ВЫБОР ИНСТРУМЕНТОВ И СРЕДСТВ РЕАЛИЗАЦИИ 7](#_Toc187484100)

[5 НАЗНАЧЕНИЕ ПЛАГИНА 9](#_Toc187484101)

[6 ОБЗОР АНАЛОГОВ 10](#_Toc187484102)

[7 ОПИСАНИЕ РЕАЛИЗАЦИИ 12](#_Toc187484103)

[8 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ 21](#_Toc187484104)

[9 ТЕСТИРОВАНИЕ ПЛАГИНА 26](#_Toc187484105)

[9.1 Функциональное тестирование 26](#_Toc187484106)

[9.2 Модульное тестирование 32](#_Toc187484107)

[9.3 Нагрузочное тестирование 34](#_Toc187484108)

[10 ЗАКЛЮЧЕНИЕ 36](#_Toc187484109)

[11 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 37](#_Toc187484110)

# **1 ВВЕДЕНИЕ**

**1.1 Информация о выбранной САПР**

САПР (Система автоматизированного проектирования) — это совокупность средств и методов, предназначенных для автоматизации процессов проектирования изделий или их компонентов [1].

КОМПАС–3D – это российская импортонезависимая система трехмерного проектирования, ставшая стандартом для тысяч предприятий и сотен тысяч профессиональных пользователей.

Данная САПР широко используется для проектирования изделий основного и вспомогательного производств в таких отраслях промышленности, как машиностроение (транспортное, сельскохозяйственное, энергетическое, нефтегазовое, химическое и т.д.), приборостроение, авиастроение, судостроение, станкостроение, вагоностроение, металлургия, промышленное и гражданское строительство, товары народного потребления и т.д. [2].

Прямыми аналогами КОМПАС–3D являются САПР Autodesk Inventor и SolidWorks. К косвенным аналогам можно отнести TinkerCAD и Micro–Cap.

## **1.2 Описание API**

API (сокр. от Application Programming Interface) – это программный интерфейс приложения, который предоставляет разработчику набор функций вместе с описанием того, что эти функции делают [3].

В Компас–3D API существуют двух версий: API 5 и API 7. Обе версии реализуют различные функции системы и взаимодополняют друг друга. Отсюда, обе версии программных интерфейсов в равной мере поддерживаются и развиваются с учетом самих изменений в системе. Для реализации создания плагина Чайник будет использоваться API Компас–3D версии 5 [4].

# **2 ПОСТАНОВКА И АНАЛИЗ ЗАДАЧИ**

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

Таблица 2.1 – Этапы проведения работ по разработке плагина Чайник 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 |

Потребовалось внести ряд корректировок, которые изменили конечную версию UML–диаграммы классов, а также попутно изучать особенности работы с API Компас 3D.

# **3 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ**

Чайник – это кухонный прибор, который представляет из себя сосуд, используемый для кипячения воды. Он состоит из нескольких элементов, включая металлический корпус, крышку, ручку, носик и дно [5]. Чертёж чайника представлен на рисунке 3.1.



Рисунок 3.1 – Чертёж чайника

Изменяемые параметры для предмета проектирования (также все

обозначения показаны на рисунке 3.1):

− Диаметр дна чайника d1 (от 100 до 400мм);

− Диаметр крышки чайника d2 (от 75 до 300мм, но не больше дна d1);

− Высота чайника h1 (от 80 до 450мм);

− Высота ручки h2 (от 70 до 150мм, но не больше высоты чайника);

− Объём чайника (от 0.63 до 56.55л) – может задаваться пользователем, как и d1 или h1 и недостающий третий параметр будет автоматически заполняться в соответствии с ограничениями;

− Цвет чайника.

# **4 ВЫБОР ИНСТРУМЕНТОВ И СРЕДСТВ РЕАЛИЗАЦИИ**

При реализации плагина использовались следующие инструменты:

– Visual Studio 2022;

– WindowsForms, .NET Framework 4.7.2;

– GitHub и GitHub Desktop;

– ComputerInfo (версии 0.2.0);

– NUnit (версии 3.14);

– NUnit.Analyzers;

– NUnit3TestAdapter;

– Fine Code Coverage;

Плагин разработан с использованием технологии создания настольных приложений для операционной системы Windows – Windows Forms, которая предоставляет широкий набор функций для разработки, включая управление элементами интерфейса, работу с графикой, привязку данных и обработку ввода от пользователя [6], а также поддержку программной платформы .NET Framework 4.7.2, основанной на объектно–ориентированной модели и поддерживающая создание как клиентских, так и серверных приложений.

GitHub − это платформа, предоставляющая возможности для хранения, распространения и совместной работы над кодом. В её основе лежит система управления версиями − Git, которая эффективно отслеживает изменения в файлах [7].

ComputerInfo − это библиотека для .NET, предоставляющая удобные средства для получения системной информации, включая данные о процессоре, оперативной памяти, дисках и других аппаратных характеристиках.

NUnit − это фреймворк для модульного тестирования в .NET, позволяющий разрабатывать и запускать автоматизированные тесты. Он поддерживает гибкую настройку тестов, параметризацию и тестирование асинхронного кода.

Fine Code Coverage − это расширение для Visual Studio, которое визуализирует покрытие кода тестами прямо в IDE. Оно интегрируется с популярными инструментами тестирования и позволяет легко отслеживать участки кода, не охваченные тестами.

# **5 НАЗНАЧЕНИЕ ПЛАГИНА**

Назначение разрабатываемого плагина заключается в быстром моделировании чайников различных видов. С его помощью, производители чайников могут визуализировать спроектированную модель и при необходимости настроить её параметры под свои требования.

# **6 ОБЗОР АНАЛОГОВ**

DriveWorks – это плагин для SolidWorks, который используется для автоматического создания параметрических моделей [8]. Он позволяет пользователю настраивать различные параметры моделей через интерфейс, а также автоматически генерирует 3D–модели на основе введённых данных. DriveWorks поддерживает автоматизированную валидацию параметров и создание чертежей. Интерфейс данного плагина представлен на рисунке 6.1.



Рисунок 6.1 – Интерфейс плагина DriveWorks

Вторым аналогом является плагин EasyKitchen PRO. Он содержит в себе модули и компоненты для проектирования кухонной и корпусной мебели в программе SketchUp, плагин замены компонентов и текстур, инструменты для генерации отчёта и создания деталировки и сметы [9]. Интерфейс программы представлен на рисунке 6.2.



Рисунок 6.2 – Интерфейс плагина EasyKitchen PRO

# **7 ОПИСАНИЕ РЕАЛИЗАЦИИ**

UML–диаграмма классов – тип статической структурной диаграммы, описывающей структуру системы посредством обозначения классов, их атрибутов, методов, связей на диаграмме [10].

На рисунке 7.1 отображена диаграмма классов приложения после проектирования системы.

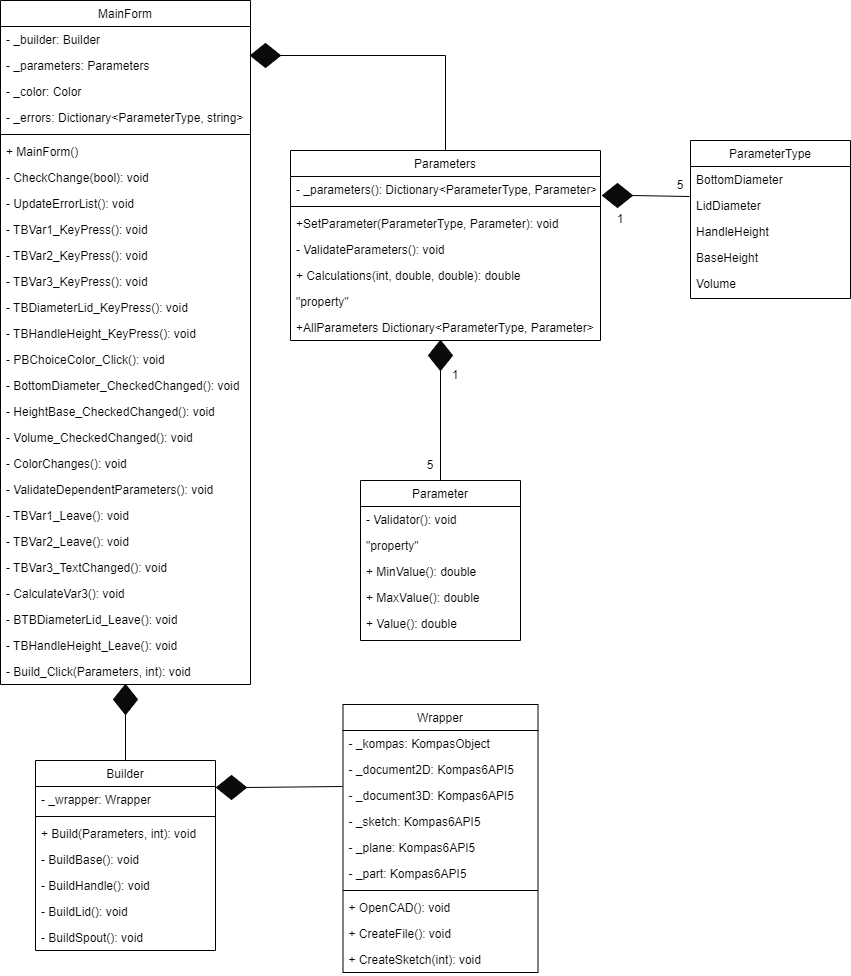


Рисунок 7.1 – Пример UML-диаграммы классов после проектирования

В таблицах ниже представлена информация о свойствах и методах каждого из классов после проектирования (таблицы 7.1–7.10).

Таблица 7.1 − Свойства класса MainForm

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_builder | Builder | Хранит в себе объект построения |
| \_parameters | Parameters | Хранит в себе параметры для объекта построения. |
| \_color | Color | Хранит в себе выбранный цвет |
| \_errors | Dictionary<ParameterType, string> | Словарь для записи ошибок |

Таблица 7.2 − Методы класса MainForm

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| MainForm | − | Конструктор главной формы |
| CheckChange | bool | Функция для изменения полей в зависимости от выбранной радиокнопки |
| UpdateErrorList | − | Обновление листа с ошибками |
| TBVar1\_KeyPress | − | Обработчик ввода для поля var1 |
| TBVar2\_KeyPress | − | Обработчик ввода для поля var2 |
| TBVar3\_KeyPress | − | Обработчик ввода для поля var3 |
| TBDiameterLid\_KeyPress | − | Обработчик ввода для диаметра крышки |
| TBHandleHeight\_KeyPress | − | Обработчик ввода для длины ручки чайника |
| PBChoiceColor\_Click | − | Обработчик выбора цвета чайника |
| BottomDiameter \_CheckedChanged | − | Обработчик выбора радиокнопки "Диаметр дна" |
| HeightBase\_CheckedChanged | − | Обработчик выбора радиокнопки "Высота чайника" |
| Volume\_CheckedChanged | − | Обработчик выбора радиокнопки "Объем чайника" |
| ColorChanges | − | Обновляет цвет текстбоксов при валидации |
| ValidateDependentParameters | − | Проверяет зависимые параметры |
| TBVar1\_Leave | − | Обработчик события ухода фокуса из поля var1 |
| TBVar2\_Leave | − | Обработчик события ухода фокуса из поля var2 |
| TBVar3\_TextChanged | − | Обработчик изменения текста в поле var3 |
| CalculateVar3 | − | Вычисляет значение var3 на основе других параметров |
| BTBDiameterLid\_Leave | − | Обработчик ухода фокуса из поля диаметра крышки |
| TBHandleHeight\_Leave | − | Обработчик ухода фокуса из поля длинны ручки чайника |
| Build\_Click | Parameters, int | Обработчик кнопки “Построить” |

Таблица 7.3 − Свойства класса Parameters

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_parameters | Dictionary<ParameterType, Parameter> | Хранит в себе словарь с параметрами объекта построения |
| Color | color | Переменная для хранения цвета, отдельно от числовых переменных |

Таблица 7.4 – Методы класса Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| Parameters | − | − | Конструктор класса Parameters |
| ValidateParameters | Parameter, Parameter | bool | Зависимая валидация параметра относительно другого параметра |
| Calculations | int, double, double | double | Функция для расчета третьего параметра по двум другим, относительно выбранной радиокнопки |

Таблица 7.5 − Свойства класса Builder

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

Таблица 7.6 − Методы класса Builder

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| Build | Parameters, int | Построение модели по заданным параметрам |
| BuildBase | − | Построение основания чайника |
| BuildHandle | − | Построение ручки чайника |
| BuildLid | − | Построение крышки чайника |

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 | Описание |
| Validator | Валидация минимальных и максимальных значений |

Таблица 7.9 – Свойства класса Wrapper

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_kompas | KompasObject | Объект для управления системой Компас3D |
| \_document2D | Kompas6API5 | Документ 2D для создания чертежей |
| \_document3D | Kompas6API5 | Документ 3D для построения моделей |
| \_sketch | Kompas6API5 | Эскиз для работы с 2D-геометрией |
| \_plane | Kompas6API5 | Плоскость для построения геометрии в 3D |
| \_part | Kompas6API5 | Деталь или компонент модели |

Таблица 7.10 − Методы класса Wrapper

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| CreateCircle | Point, double | Создание дуги по двум точкам |
| CreateLine | Point, Point | Создание линии |
| CreateSketch | − | Создание эскиза |
| OpenCAD | − | Открытие Компас3D |
| CreateArc | Point, Point, double | Создание дуги |
| CreateCylinder | Point, double, double | Создание цилиндра |

Далее, на рисунке 7.2, отображена диаграмма классов приложения после реализации системы.

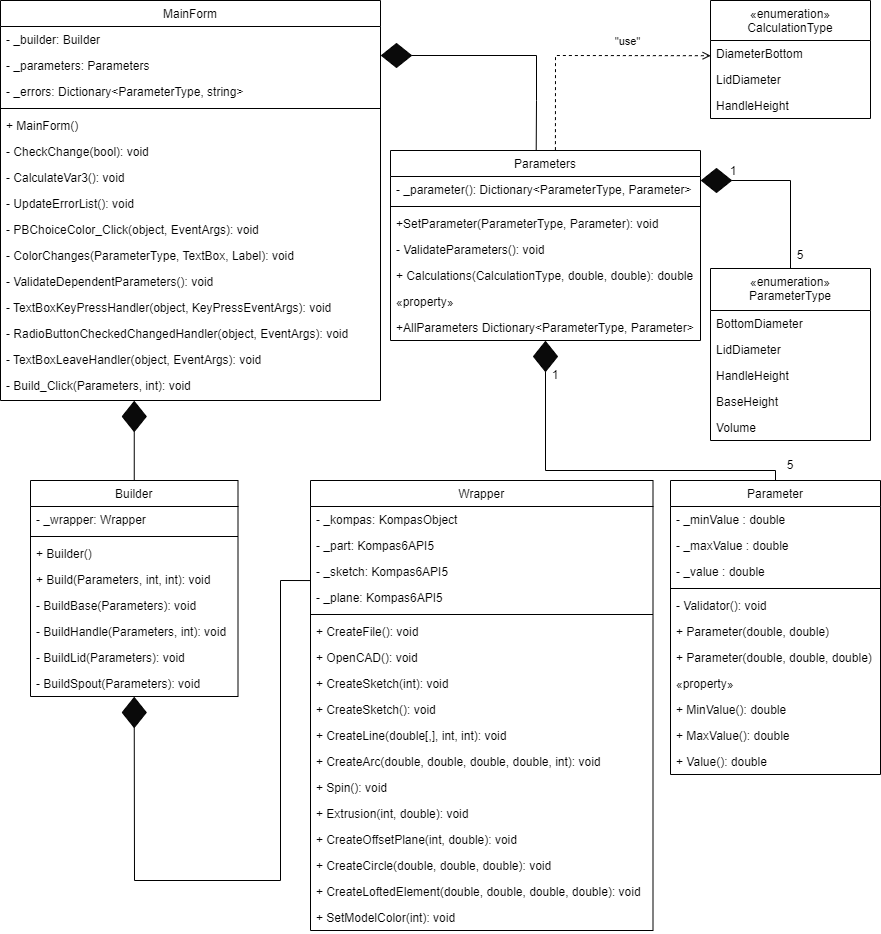


Рисунок 7.2 – UML–диаграмма классов после реализации

В таблицах ниже представлена информация о свойствах и методах каждого из классов после реализации системы (таблицы 7.11–7.20).

Таблица 7.11 − Свойства класса MainForm

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_builder | Builder | Построитель для создания модели. |
| \_parameters | Parameters | Параметры модели. |
| \_errors | Dictionary<ParameterType, string> | Словарь, содержащий строки ошибок валидации. |

Таблица 7.12 − Методы класса MainForm

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| MainForm | − | Конструктор главной формы. |
| CheckChange | bool | Функция для изменения полей при выборе другой формулы. |
| CalculateVar3 | − | Функция для вычисления значение var3 на основе других двух параметров. |
| UpdateErrorList | − | Обновление списка ошибок. |
| PBChoiceColor\_Click | object, EventArgs | Обработчик выбора цвета чайника. |
| ColorChanges | ParameterType, TextBox, Label | Обновляет цвет текстбоксов при валидации. |
| ValidateDependent Parameters | ParameterType | Проверяет зависимые параметры. |
| TextBoxKeyPressHandler | object, KeyPressEventArgs | Обработчик ввода для полей. |
| RadioButtonChecked ChangedHandler | object, EventArgs | Обработчик выбора радиокнопки “Диаметр дна”. |
| TextBoxLeaveHandler | object, EventArgs | Обработчик выхода из текстового поля. |
| Build\_Click | object, EventArgs | Обработчик кнопки “Построить”. |

Таблица 7.13 − Свойства класса Parameters

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_parameter | Dictionary<ParameterType, Parameter> | Словарь, содержащий параметры модели, ключ − тип параметра, значение − сам параметр. |

Таблица 7.14 – Методы класса Parameters

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| AllParameters | Dictionary <ParameterType, Parameter> | − | Геттер и сеттер параметров модели. |
| SetParameter | ParameterType, Parameter | − | Устанавливает параметр в словарь. Если уже существует – обновляется. |
| ValidateParameters | − | − | Выполняет валидацию параметров модели. |
| Calculations | CalculationType, double, double | double | Выполняет расчёты параметров чайника в зависимости от входных данных. |

Таблица 7.15 − Свойства класса Builder

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

Таблица 7.16 − Методы класса Builder

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| Builder | − | Новый экземпляр класса Builder. |
| Build | Parameters, int, int | Построение модели по заданным параметрам. |
| BuildBase | Parameters | Построение основания чайника. |
| BuildHandle | Parameters, int | Построение ручки чайника, с возможностью смены формы. |
| BuildLid | Parameters | Построение крышки чайника. |
| BuildSpout | Parameters | Построение носика чайника. |

Таблица 7.17 − Свойства класса Parameter

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

Таблица 7.18 − Методы класса Parameter

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | | Описание |
| Validator | − | − | | Валидация минимальных и максимальных значений. |
| Parameter | double, double | − | Конструктор для задания минимального и максимального значений. | |
| Parameter | double, double, double | − | Конструктор для задания минимального, максимального и текущего значений. | |
| MaxValue | − | double | Получает/задает максимально допустимое значение параметра. | |
| MinValue | − | double | Получает/задает минимально допустимое значение параметра. | |
| Value | − | double | Получает/задает текущее значение параметра. | |

Таблица 7.19 – Свойства класса Wrapper

|  |  |  |
| --- | --- | --- |
| Название | Тип данных | Описание |
| \_kompas | KompasObject | Объект для управления системой Компас3D. |
| \_part | Kompas6API5 | Деталь или компонент модели. |
| \_sketch | Kompas6API5 | Эскиз для работы с 2D–геометрией. |
| \_plane | Kompas6API5 | Плоскость для построения геометрии в 3D. |

Таблица 7.20 − Методы класса Wrapper

|  |  |  |
| --- | --- | --- |
| Название | Входные параметры | Описание |
| CreateFile | − | Создание документа в Компас3D. |
| OpenCAD | − | Открытие САПР Компас3D. |
| CreateSketch | int | Создание эскиза. |
| CreateSketch | − | Создание эскиза на созданной плоскости. |
| CreateLine | double[,], int, int | Создание линии. |
| CreateArc | double, double, double, double, int | Создание дуги. |
| Spin | − | Задание вращения. |
| Extrusion | int, double | Функция выдавливания. |
| CreateOffsetPlane | int, double | Создание смещенной плоскости. |
| CreateCircle | double, double, double | Создание дуги по двум точкам. |
| CreateLofted Element | double, double, double, double | Выдавливание объекта по сечениям. |
| SetModelColor | int | Изменение цвета всей модели. |

В отличии от диаграммы классов после проектирования, диаграмма классов после реализации плагина имеет следующие изменения.

1. В MainForm была убрана переменная цвета, так как нам не нужно хранить информацию на протяжении выполнения всего кода, достаточно взять цвет выполненного colorDialog и преобразовать его в целочисленное значение, а затем сразу отправить его при нажатии на кнопку построения. Помимо этого, с целью уменьшения дублирования кода у текстовых полей и радиокнопок были убраны такие события, как KeyPress, CheckedChanged, Leave и TextChanged, которые были заменены на единые обработчики событий TextBoxKeyPressHandler, RadioButtonCheckedChangedHandler и TextBoxLeaveHandler. Это позволило указать в событиях элементов формы один и тот же метод;
2. В классе Parameter была изменена настройка приватности сеттера минимального и максимального значений для того, чтобы разработчик не мог влиять на переменные напрямую. А также были добавлены два конструктора с целью задать минимально и максимально возможных значений параметра или возможность кроме границы параметра установить также и текущее значение;
3. В класс Builder, в функцию Build были добавлены две целочисленные (int) переменные во входные параметры – color (переменная цвета) и handleForm (переменная типа формы ручки, в качестве дополнительной функциональности плагина) для передачи информации о выбранных, не числовых параметрах;
4. В классе Wrapper были добавлены следующие методы: CreateOffsetPlane (создание смещенной плоскости), CreateSketch (создание скетча на созданной плоскости), CreateLoftedElement (выдавливание объекта по сечениям), Extrusion (выдавливание), Spin (выдавливание вращением), SetModelColor (изменение цвета всей модели), а также был удален метод CreateCylinder (создание цилиндра), так как данный метод было решено заменить созданием скетча и дальнейшей функцией выдавливанием вращением.
5. В классе Parameters было добавлено перечисление CalculationType, которое определяет возможные типы расчетов (диаметр дна, высота чайника, объем), а также изменен метод Calculations, где на вход вместо целочисленного аргумента теперь используется CalculationType. Это было сделано с целью улучшить читаемость и понимание кода.

Описанные выше изменения были внесены уже при написании плагина и его дальнейшей модификации, а также при более детальном изучении и подходе при построении модели чайника, в частности когда понадобились новые методы в класс Wrapper, которых не хватало для корректного построения модели.

# **8 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПОЛЬЗОВАТЕЛЯ**

При запуске плагина открывается форма, поля которой необходимо заполнить (рисунок 8.1).

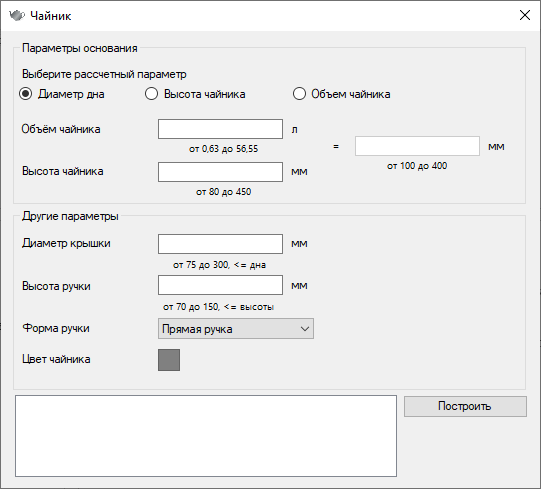


Рисунок 8.1 – Форма для заполнения параметров при запуске плагина

Первым делом стоит выбрать один из трех предложенных расчетных параметров – “диаметр дна”, “высота чайника” или “объем чайника”. Это позволит поставить в приоритет определенные потребности, которые необходимо учесть при построении модели чайника (рисунок 8.2).

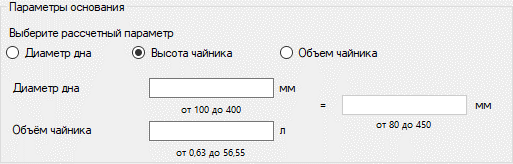


Рисунок 8.2 – Пример изменений полей при выборе расчетного параметра

В зависимости от выбранного расчетного параметра поменяется третье поле, изменять которое вручную нельзя, так как оно рассчитывается по формулам автоматически при заполнении двух других полей. Формулы представлены ниже.

Формула для расчета диаметра дна чайника (d).



Формула для расчета высоты чайника (h).



Формула для расчета объема чайника (V).



Подсказки по корректному заполнению полей представлены под ним, а единицы измерения указаны справа от поля. На рисунке 8.3 продемонстрирован пример подобных подсказок, например для параметра “диаметр крышки” необходимо ввести целое или вещественное число в диапазоне от 75 до 300 мм, при этом данное значение должно быть не больше значение параметра основания “дно чайника”.



Рисунок 8.3 – Пример подсказки для заполнения полей

В случае, если пользователь все же введет недопустимые значения он будет предупрежден о данной ошибке уведомлением в самом низу формы, а также красной подсветкой поля и подсказки, как на рисунке 8.4.

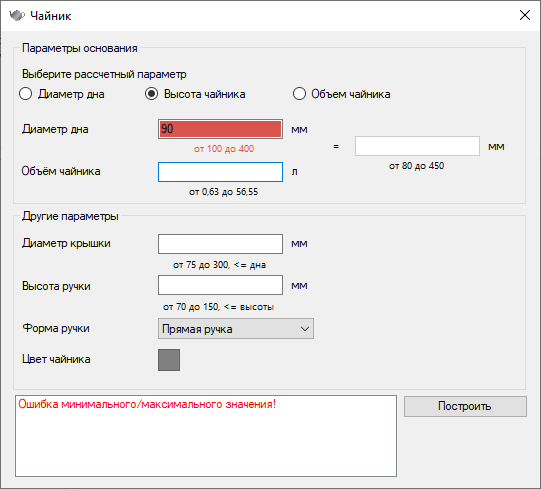


Рисунок 8.4 – Пример ошибки при недопустимом значении

Помимо основных числовых параметров в форме есть параметры для дополнительной функциональности при построении чайника. Так, можно выбрать форму ручки (рисунок 8.5) или изменить цвет чайника (рисунок 8.6).

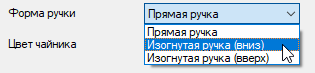


Рисунок 8.5 – Смена формы ручки

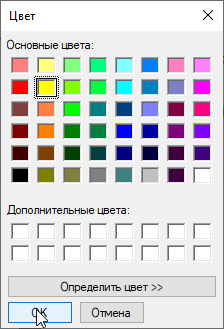


Рисунок 8.6 – Смена цвета

После заполнения всех полей и отсутствии ошибок (рисунок 8.7) пользователь может нажать на кнопку “Построить” для выполнения построения модели по заданным параметрам в Компас–3D (рисунок 8.8).

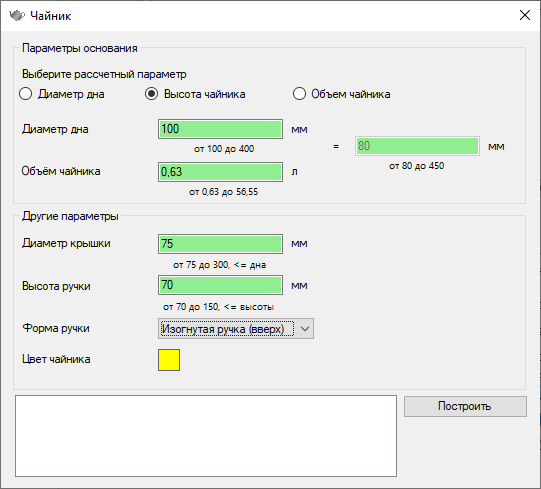


Рисунок 8.7 – Пример заполнения и отсутствия ошибок

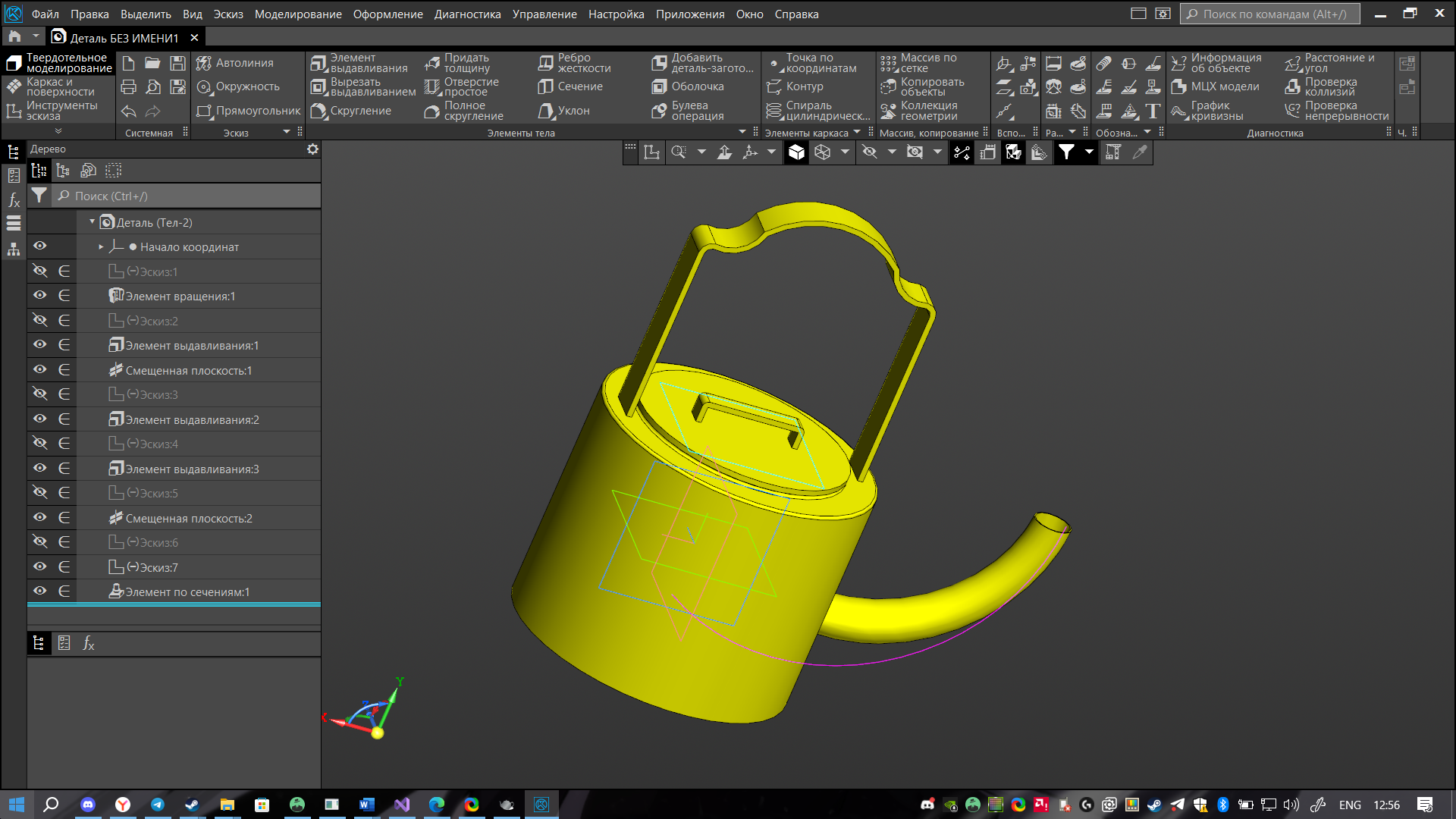


Рисунок 8.8 – Успешное построение модели чайника в Компас–3D

В случае, если пользователь так и не заполнил поля или заполнил их с ошибками, но при этом он нажмет на кнопку “Построить”, то появится уведомление о невозможности построения.

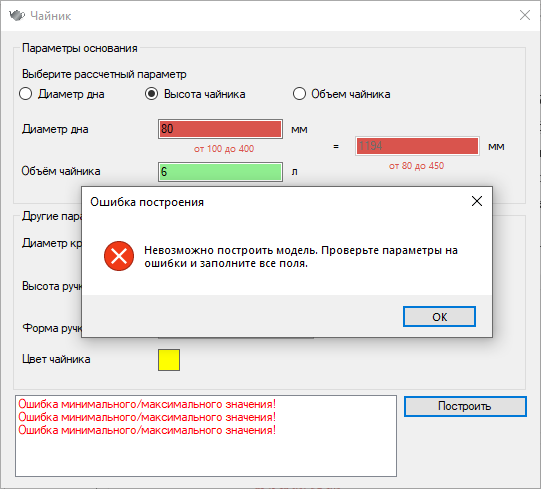


Рисунок 8.9 – Ошибка построения

# **9 ТЕСТИРОВАНИЕ ПЛАГИНА**

## **9.1 Функциональное тестирование**

Для проверки валидации некорректных данных необходимо представить возможные действия пользователя с данным интерфейсом. Предположим, что пользователь ввел корректные данные в нужном диапазоне, но зависимые параметры, которые должны быть меньше или равны другому оказываются куда больше, чем нужно. Пример таких действий показан на рисунке 9.1.

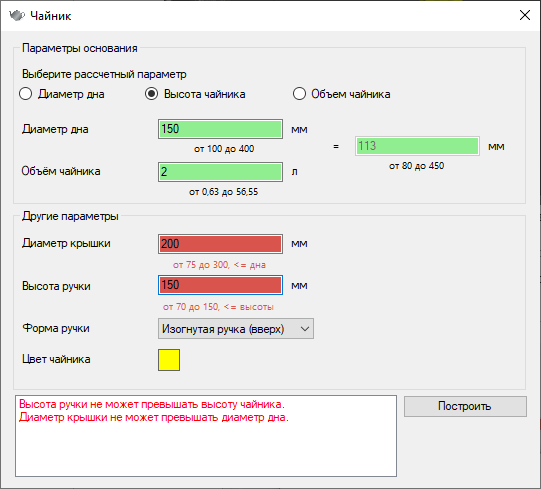


Рисунок 9.1 – Зависимая валидация параметров

Предположим, что пользователь ввел параметры, которые выходят за диапазон значений, что показано на рисунке 9.2.

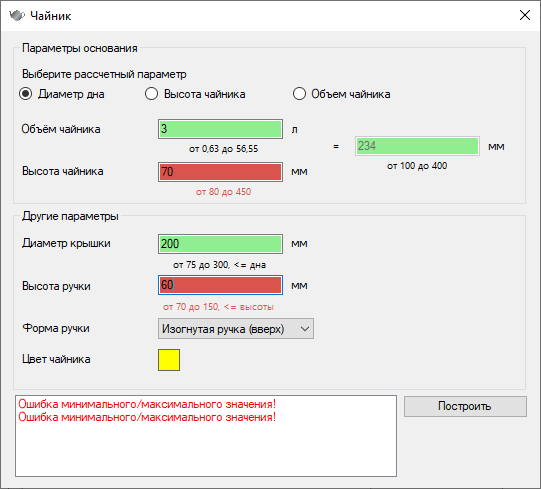


Рисунок 9.2 – Диапазонная валидация значений

Также, пользователь мог ввести некорректные данные для вычисления третьего параметра по формуле, на рисунке 9.3 продемонстрирована работа валидации на третьем расчетном параметре.

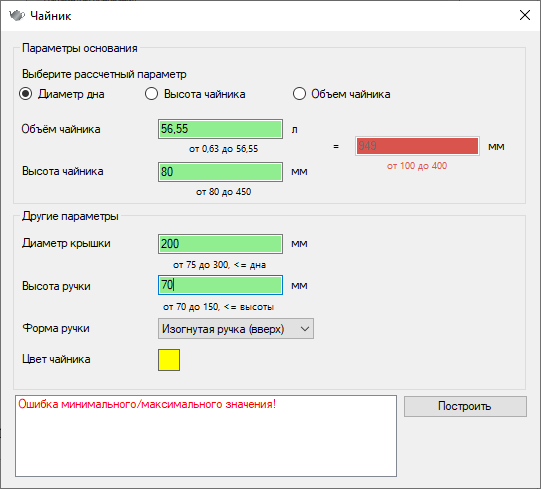


Рисунок 9.3 – Валидация третьего расчетного параметра

Валидация для проверки формата данных не требуется, так как все поля принимают только цифры и запятую для ввода (события KeyPress и условий для проверки введенного символа).

На рисунке 9.4 представлено заполнение формы минимально возможными параметрами.

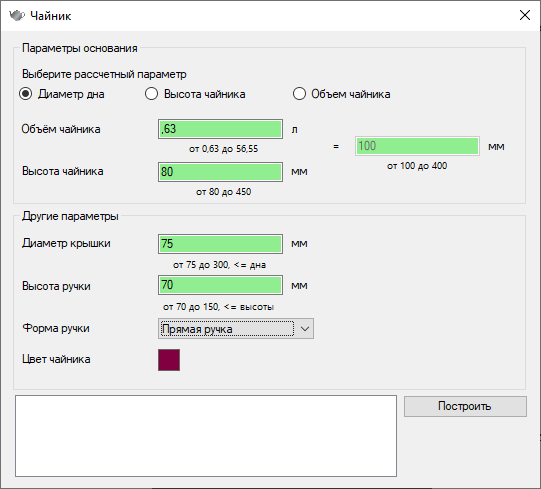


Рисунок 9.4 – Минимально возможные параметры

На рисунке 9.5 изображен результат построения модели чайника с минимально возможными параметрами.

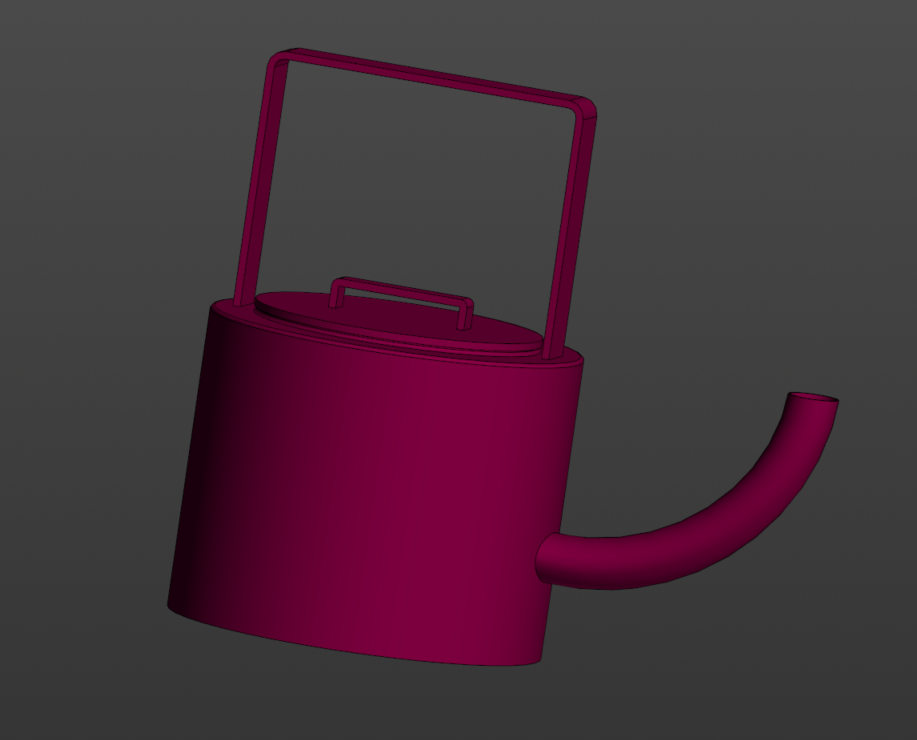
****

Рисунок 9.5 – Модель чайника с минимально возможными параметрами

На рисунке 9.6 представлено заполнение формы максимально возможными параметрами.

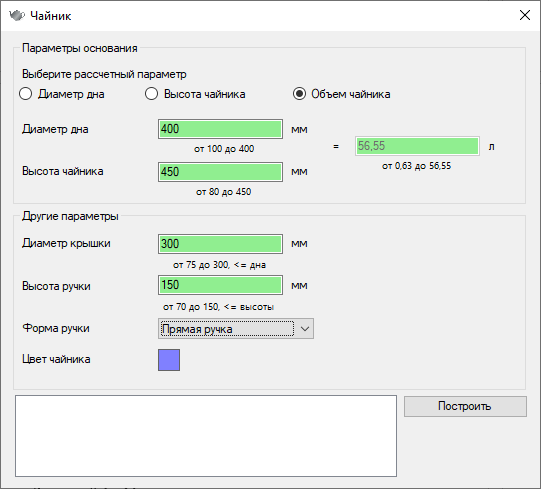


Рисунок 9.6 – Максимально возможные параметры

На рисунке 9.7 изображен результат построения модели чайника с максимально возможными параметрами.

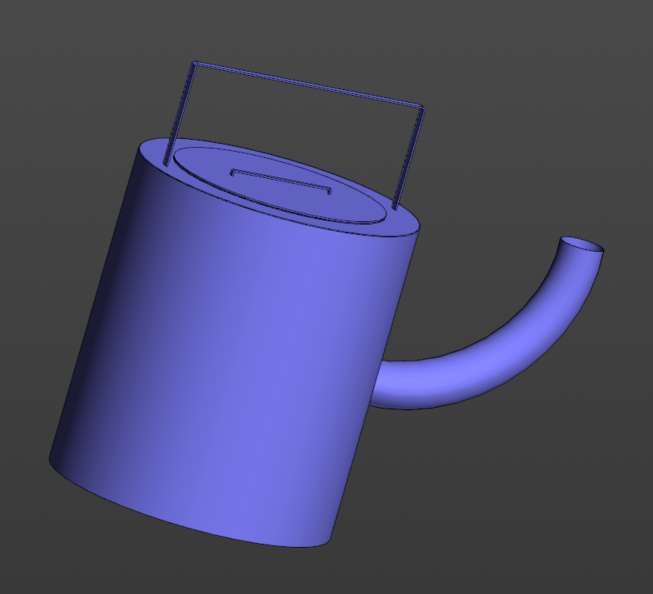
****

Рисунок 9.7 – Модель чайника с максимально возможными параметрами

На рисунке 9.8 представлено заполнение формы стандартными параметрами.

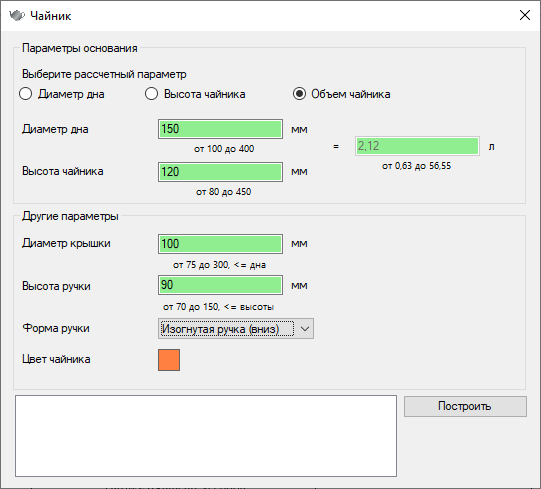


Рисунок 9.8 – Стандартные параметры

На рисунке 9.9 изображен результат построения модели чайника со стандартными параметрами.

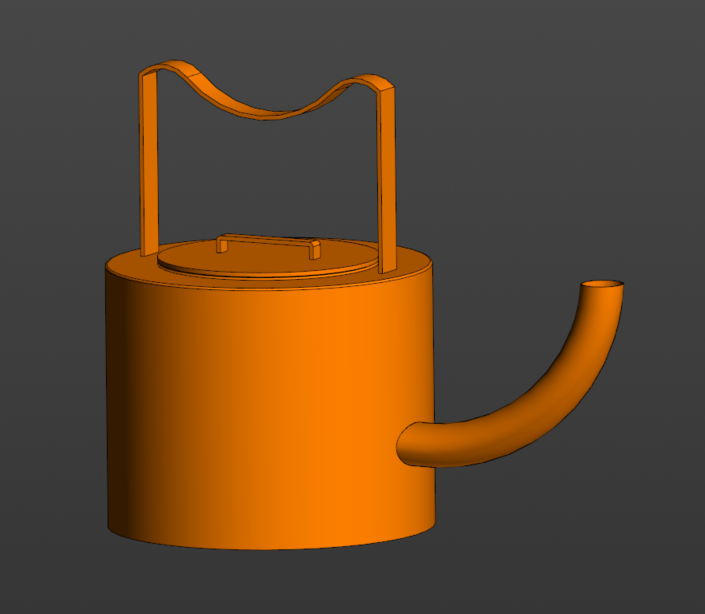
****

Рисунок 9.9 – Модель чайника со стандартными параметрами

## **9.2 Модульное тестирование**

Для обеспечения стабильности и корректной работоспособности классов плагина, таких как Parameter и Parameters были написаны Unit–тесты. В таблице 9.1 представлены названия и описания этих тестов.

Таблица 9.1 – Unit−тесты

|  |  |
| --- | --- |
| Название теста | Описание теста |
| Позитивный тест геттера MaxValue.  MaxValue\_Get\_ReturnsCorrectValue | Проверяет работу get у MaxValue. |
| Позитивный тест сеттера MaxValue.  MaxValue\_Set\_SetsCorrectValue | Проверяет работу set у MaxValue. |
| Позитивный тест геттера MinValue.  MinValue\_Get\_ReturnsCorrectValue | Проверяет работу get у MinValue. |
| Позитивный тест сеттера MinValue.  MinValue\_Set\_SetsCorrectValue | Проверяет работу set у MinValue. |
| Позитивный тест геттера Value.  Value\_Get\_ReturnsCorrectValue | Проверяет работу get у Value. |
| Позитивный тест сеттера Value.  Value\_SetWithinMinMax\_ DoesNotThrowException | Проверяет работу set у Value. |
| Негативный тест Validator.  Value\_SetBelowMinValue\_ ThrowsArgumentException | Проверяет вызов исключения при Value < MinValue. |
| Негативный тест Validator.  Value\_SetAboveMaxValue\_ ThrowsArgumentException | Проверяет вызов исключения при Value > MaxValue. |
| Позитивный тест геттера AllParameters.  AllParameters\_Get\_ReturnsCorrectDictionary | Проверяет работу get у AllParameters. |
| Позитивный тест сеттера AllParameters.  AllParameters\_Set\_SetsCorrectDictionary | Проверяет работу set у AllParameters. |
| Позитивный тест метода SetParameter.  SetParameter\_ValidParameters\_ SuccessfullyUpdatesParameter | Проверяет работу set для \_parameter. |
| Негативный тест ValidateParameters.  ValidateParameters\_InvalidDependentValues \_ThrowsArgumentException | Проверяет вызов исключения при некорректных зависимостях. |
| Проверяет корректность расчета диаметра дна.  Calculations\_BottomDiameter\_ReturnsCorrectValue | Проверяет корректную работу расчетной формулы диаметра дна. |
| Проверяет корректность расчета высоты.  Calculations\_Height\_ReturnsCorrectValue | Проверяет корректную работу расчетной формулы высоты. |
| Проверяет корректность расчета объема.  Calculations\_Volume\_ReturnsCorrectValue | Проверяет корректную работу расчетной формулы объема. |

На рисунке 9.10 Представлены количество написанных Unit–тестов, их корректное выполнение, а также время, затраченное на их выполнение.

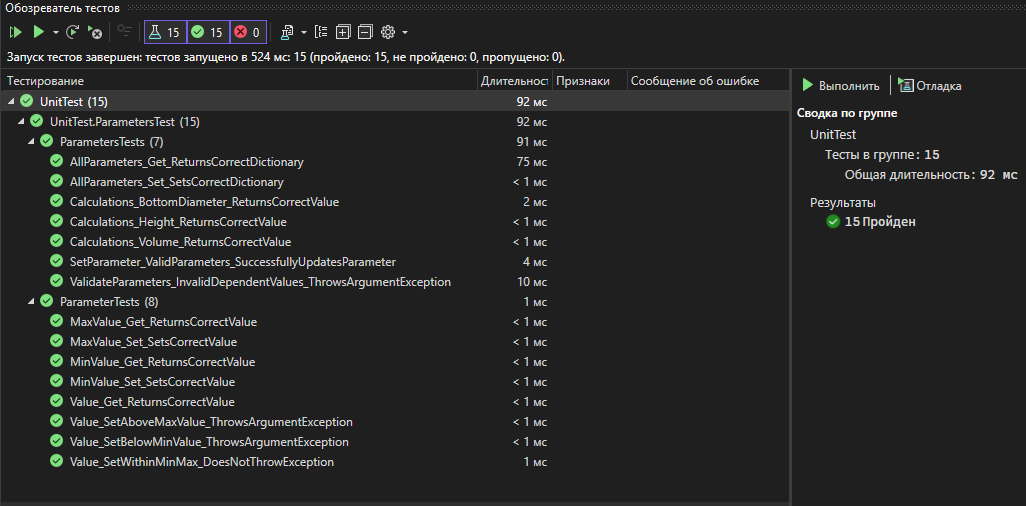


Рисунок 9.10 – Выполненное Unit–тестирование

На рисунке 9.11 изображен результат покрытия выполненных модульных тестов их количество и процент покрытия. Процент покрытия модульными тестами составил 100%.

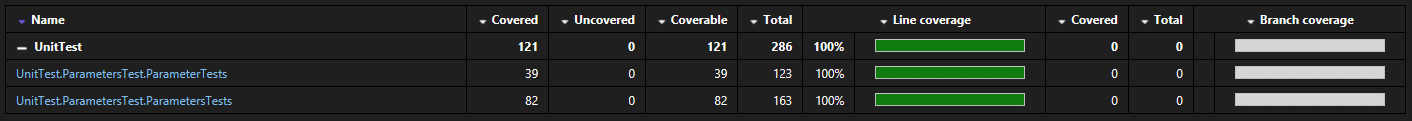


Рисунок 9.11 – Процент покрытия модульных тестов

## **9.3 Нагрузочное тестирование**

Проведенное нагрузочное тестирование (StressTester) позволило построить график зависимостей памяти ОЗУ (рисунок 9.12) и график гистограммы, затраченного на построение модели времени (рисунок 9.13) от количества построенных моделей. Количество построенных моделей достигло 118 штук, после чего плагин сильно замедлял работу системы.

Конфигурация ПК (ноутбука) во время тестирования была следующей:

− процессор AMD Ryzen 7 7735HS;

− оперативная память объемом 16 ГБ (доступно 15,2 ГБ);

− видеокарта NVIDIA GeForce RTX 4060 Laptop GPU;

− 64-разрядная операционная система Windows 10 Pro 22H2.

Нагрузочное тестирование проводилось на минимально возможных параметрах модели:

– объем чайника (volume) – 0.63 л;

– высота чайника (heightBase) – 80 мм;

– диаметр крышки (diameterLid) – 75 мм;

– длина ручки (heightHandle) – 70 мм;

– диаметр дна (diameterBottom) – 100 мм.

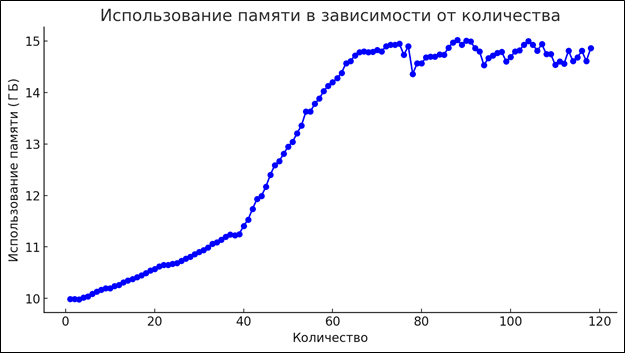


Рисунок 9.12 – График зависимости памяти ОЗУ от количества моделей с минимальными параметрами

Анализируя график на рисунке 9.12, можно сделать следующий вывод: график показывает рост потребляемой памяти с каждым новым созданным объектом, начиная от 10 ГБ, при открытии Компас–3D, приближаясь к отметке 15 ГБ (на 88–м объекте), далее этот показатель варьируется в пределах 14,5 – 15 ГБ. Максимальное количество ОЗУ вычислительной машины, на которой проводился стресс–тест равен 15,2 ГБ.

****

Рисунок 9.13 – График гистограммы построения модели с минимальными параметрами

Исходя из графика гистограммы, изображенной на рисунке 9.13, можно сделать вывод, что время, необходимое на построение модели, варьируется в пределах от 2 до 6 секунд, на первой построенной модели время построения достигло 11–и секунд, так как это было первое открытие САПР. Однако после 109–й построенной модели система периодически стала зависать и затраченное время скачкообразно увеличилось до 7–18 секунд.

# **10 ЗАКЛЮЧЕНИЕ**

В ходе выполнения лабораторных работ был создан плагин для КОМПАС−3D, который автоматически генерирует модель чайника на основе заданных пользователем параметров. Каждая из выполненных лабораторных работ была нацелена на достижение этой цели. Выбор объекта проектирования, выбор системы автоматизированного проектирования (САПР), составление технического задания. При проектировании системы требовалось учитывать множество деталей, чтобы избежать крупных изменений кода и ошибок в будущем. Написание кода включало в себя работу с API, взаимодействие с пользовательским интерфейсом, использование инструментов для стандартизации кода. Все это позволило стать коду более понятным и удобным для дальнейшей работы как пользователю−заказчику, так и исполнителю.

# **11 СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. ГОСТ 23501.101−87 «Системы автоматизированного проектирования. Основные положения» (дата обращения 22.12.2024);
2. КОМПАС–3D [Электронный ресурс] − Режим доступа: https://kompas.ru/kompas–3d/about/ (дата обращения 28.09.2024);
3. API [Электронный ресурс] − Режим доступа: https://ravesli.com/what–is–an–api/ (дата обращения 30.09.2024);
4. Справочная система SDK КОМПАС–3D – API интерфейсов версия 5 [Электронный ресурс] – Режим доступа: https://help.ascon.ru/KOMPAS\_SDK/22/ru–RU/applicate.html (дата обращения 08.10.2024);
5. ГОСТ 24788−2018 «Посуда хозяйственная стальная эмалированная. Общие технические условия»;
6. Windows Forms [Электронный ресурс]. − Режим доступа https://learn.microsoft.com/ru−ru/dotnet/desktop/winforms/overview/?view=netdesktop−9.0 (дата обращения 22.12.2024);
7. Github [Электронный ресурс]. − Режим доступа https://docs.github.com/ru/get−started/start−your−journey/about−github−and−git (22.12.2024);
8. DriveWorks [Электронный ресурс] − Режим доступа: https://www.driveworks.co.uk/ (дата обращения 08.10.2024);
9. EasyKitchen Pro [Электронный ресурс] − Режим доступа: https://redkit.pro/easykitchen/easykitchen–pro/ (дата обращения 08.10.2024);
10. Фаулер M. UML. Основы, 3–е издание. – Пер. с англ. – СПб: СимволПлюс, 2004. – 192 с.