Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

Кафедра компьютерных систем в управлении и проектировании (КСУП)

**РАЗРАБОТКА ПЛАГИНА «ПИСЬМЕННЫЙ СТОЛ» ДЛЯ САПР AUTOCAD 2022**

Пояснительная записка

по дисциплине «Основы разработки САПР»

Студент гр. 589-1

\_\_\_\_\_\_\_\_\_ Д. С. Ахроменко

« 28 » октября 2022

Руководитель

к.т.н., доцент каф. КСУП

\_\_\_\_\_\_\_\_\_ А. А. Калентьев

« 28 » октября 2022

Томск 2022

**Оглавление**

[Введение 2](#_Toc117858418)

[1 ОПИСАНИЕ САПР 5](#_Toc117858419)

[1.1 Описание программы 5](#_Toc117858420)

[1.2 Описание API 6](#_Toc117858421)

[1.3 Обзор аналогов 20](#_Toc117858422)

[2 ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ 23](#_Toc117858423)

[3 ПРОЕКТ ПРОГРАММЫ 27](#_Toc117858424)

[3.1 Диаграмма классов 27](#_Toc117858425)

[3.2 Макеты пользовательского интерфейса 33](#_Toc117858426)

[4 ТЕСТИРОВАНИЕ ПЛАГИНА 40](#_Toc117858427)

[4.1 Модульное тестирование 40](#_Toc117858428)

[4.2 Функциональное тестирование 45](#_Toc117858429)

[4.3 Нагрузочное тестирование 64](#_Toc117858430)

[Заключение 70](#_Toc117858431)

[Список использованных источников 71](#_Toc117858432)

# Введение

Автоматизация проектирования занимает особое место среди информационных технологий. Во-первых, автоматизация проектирования – это синтетическая дисциплина, составными частями которой являются многие другие современные информационные технологии: здесь используются персональные компьютеры и рабочие станции, вычислительные сети и телекоммуникационные технологии, методы вычислительной математики, статистики, математического программирования, дискретной математики, искусственного интеллекта и т.д. А во-вторых, практически любому инженеру-разработчику необходимо знание основ автоматизации проектирования и умение работать со средствами САПР, поскольку в настоящее время проектные подразделения, конструкторские бюро и офисы насыщены компьютерами [1].

На сегодняшний день проектирование осуществляется за счет применения систем автоматизированного проектирования (САПР), которые решают весь комплекс задач – от анализа технического задания до разработки полного объема конструкторской и технологической документации. Это достигается за счет объединения современных технических средств и математического обеспечения, параметры и характеристики которых выбираются с максимальным учетом особенностей задач проектно-конструкторского процесса [2].

САПР представляет собой крупные организационно-технические системы, состоящие из комплекса средств автоматизации проектирования, взаимосвязанного с проектными подразделениями конкретной организации. Основная цель создания САПР – повышение эффективности труда инженеров, включая:

* сокращение трудоемкости проектирования и планирования;
* сокращение сроков проектирования;
* сокращение себестоимости проектирования и изготовления, уменьшение затрат на эксплуатацию;
* повышение качества и технико-экономического уровня результатов проектирования;
* сокращение затрат на натурное моделирование и испытания.

Как показывает практический опыт, предприятия, ведущие разработку без САПР или лишь с малой степенью их использования, оказываются неконкурентоспособными как из-за больших материальных и временных затрат на проектирование, так и из-за невысокого качества проектов [1].

В каждой крупной САПР есть собственные средства для разработки, которые предоставляют разработчикам возможность расширить функциональность данной системы под нужды конкретного проекта. Данным средством является программный интерфейс приложения (Application Programming Interface, API) – описание способов (набор классов, процедур, функций, структур или констант), которыми одна компьютерная программа может взаимодействовать с другой программой [3]. API упрощает процесс программирования при создании приложений, абстрагируя базовую реализацию и предоставляя только объекты или действия, необходимые разработчику.

Расширение функциональности в основном подразумевает разработку плагина или библиотеки на основе предоставляемого API. Плагин (англ.   
plug-in, от plug in – «подключать») – это независимо компилируемый программный модуль, динамически подключаемый к основной программе и предназначенный для расширения и (или) использования ее   
возможностей [4].

В качестве САПР, которая предоставляет API и для которой поставлена задача разработки плагина, выбрана система Autodesk AutoCAD 2022.

# ОПИСАНИЕ САПР

## Описание программы

AutoCAD – это двух- и трехмерная система автоматизированного проектирования и черчения, разработанная американской компанией Autodesk [5]. Первая версия системы была выпущена в 1982 году. AutoCAD и специализированные приложения на его основе нашли широкое применение в машиностроении, строительстве, архитектуре и других отраслях промышленности. С помощью данной системы архитекторы, инженеры и строители создают точные 2D-чертежи и 3D-модели [6].

Ранние версии AutoCAD оперировали небольшим числом элементарных объектов, таким как круги, линии, дуги и текст, из которых составлялись более сложные объекты. Однако на современном этапе возможности AutoCAD весьма широки и намного превосходят первоначальные.

В области двумерного проектирования AutoCAD по-прежнему позволяет использовать элементарные графические примитивы для получения более сложных объектов. Кроме того, программа предоставляет обширные возможности работы со слоями и аннотативными объектами (размерами, текстом, обозначениями). Использование механизма внешних ссылок (XRef) позволяет разбивать чертеж на составные файлы, за которые ответственны различные разработчики, а динамические блоки расширяют возможности автоматизации 2D-проектирования обычным пользователем без использования программирования.

Также AutoCAD включает в себя полный набор инструментов для комплексного трехмерного моделирования: поддерживается твердотельное, поверхностное и полигональное моделирование. В программе реализовано управление трехмерной печатью и поддержка облаков точек (возможность работы с результатами 3D-сканирования) [5].

## Описание API

AutoCAD представляет собой гибкую платформу разработки специализированных приложений, предназначенных для автоматизированного проектирования. Открытая архитектура позволяет адаптировать AutoCAD под конкретные задачи пользователей. Образцами такой адаптации могут служить САПР на базе AutoCAD для различных отраслей проектирования, а также тысячи надстроек, разработанных участниками сети Autodesk Developer Network [7].

**ObjectARX SDK**

Среда программирования ObjectARX используется для адаптации и расширения функциональных возможностей AutoCAD и продуктов на его основе. Она обеспечивает непосредственный доступ к структурам базы данных AutoCAD, графической системе и определениям встроенных команд.

**AutoCAD .NET API**

В состав ObjectARX SDK также входит управляемый API, который часто называют AutoCAD .NET API. Для адаптации и расширения функциональных возможностей AutoCAD и продуктов на его основе может применяться любой язык программирования, поддерживающий .NET. Обеспечивается непосредственный доступ к структурам базы данных AutoCAD, определениям встроенных команд и другим внутренним программным элементам [7].

Основными строительными блоками AutoCAD .NET API являются объекты. Каждый объект представляет собой конкретную часть AutoCAD. В AutoCAD .NET API существует множество различных типов объектов. Некоторые из объектов, представленных в AutoCAD .NET API:

* графические объекты (линии, дуги, текст, размеры);
* параметры стиля, такие как слои, типы линий и размерные стили;
* организационные структуры (вид, видовой экран);
* чертеж и сама программа AutoCAD.

Объекты структурированы иерархически, корневым объектом является AutoCAD Application. Эту иерархическую структуру часто называют «объектной моделью» (Object Model).

На рисунке 1.1 показаны взаимоотношения объекта Application и сущностей, каждая из которых (Entity) находится в BlockTableRecord, например в пространстве модели – Model Space [8].



Рисунок 1.1 – Иерархия объектов в AutoCAD

Объект Application – корневой объект AutoCAD .NET API. Из объекта Application можно получить доступ к главному окну AutoCAD, а также к любому открытому чертежу. Поскольку есть доступ к чертежу, можно взаимодействовать и с его объектами (за работу с открытыми файлами чертежей отвечает объект Document).

Объект Document, который на самом деле является чертежом AutoCAD, – это часть объекта DocumentCollection. Document обеспечивает доступ к объекту базы данных Database, связанному с объектом Document.

Объект Database содержит все графические и большинство неграфических объектов AutoCAD. К графическим объектам относятся видимые объекты, которые составляют чертеж: линии, окружности, растровые изображения и т.д. Другими словами, пользователь может видеть такой объект на экране и манипулировать им. Неграфическими объектами являются невидимые (информационные) объекты, такие как слои, типы линий, размерные стили, таблицы стилей [8].

Принцип создания и работы плагина для AutoCAD:

* создание проекта «Библиотека классов» (Class Library);
* добавление ссылок на необходимые библиотеки AutoCAD .NET API (из ObjectARX SDK);
* написание программного кода плагина;
* загрузка созданного плагина в AutoCAD (запуск AutoCAD и выполнение команды NETLOAD);
* отладка плагина (при необходимости).[9]

При создании плагинов для AutoCAD основными используемыми пространствами имен являются:

* **Autodesk.AutoCAD.ApplicationServices** – позволяет получить доступ к приложению AutoCAD;
* **Autodesk.AutoCAD.EditorInput** – позволяет получить доступ к редактору (Editor) AutoCAD;
* **Autodesk.AutoCAD.DatabaseServices** – предоставляет доступ к базе данных и сущностям AutoCAD;
* **Autodesk.AutoCAD.Geometry** – группирует все, что относится к геометрии в AutoCAD;
* **Autodesk.AutoCAD.Runtime** – отвечает за регистрацию   
  команд [9, 10].

В таблицах 1.1 – 1.34 представлены свойства и методы классов и структур AutoCAD .NET API, использованные при разработке плагина «Письменный стол».

Таблица 1.1 – Свойства класса Application

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| DocumentManager | DocumentCollection | Свойство, предоставляющее доступ к объекту DocumentManager |

Таблица 1.2 – Методы класса Application

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращаемый тип | Описание |
| Аргумент | Тип | Описание |
| Show  Modeless  Window | formToShow | Form | Окно WPF, которое нужно отобразить | void | Отображает окно WPF в немодальном режиме. Применение данного метода вместо стандартного метода Show() предотвращает неожиданное поведение программы |

Таблица 1.3 – Свойства класса DocumentCollection

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| MdiActiveDocument | Document | Предоставляет доступ к активному документу с расширением MDI |

Таблица 1.4 – Свойства класса Document

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| Database | Database | Возвращает объект базы данных, который используется данным документом |

Таблица 1.5 – Методы класса Document

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращае-мый тип | Описание |
| Аргумент | Тип | Описание |
| Lock  Document | – | – | – | Document  Lock | Блокирует текущий документ. Используется для предотвращения конфликтов с другими запросами в базе данных при взаимодействии с AutoCAD через немодальное диалоговое окно |

Таблица 1.6 – Свойства класса Database

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| BlockTableId | ObjectId | Возвращает идентификатор объекта в таблице BlockTable базы данных |
| TransactionManager | TransactionManager | Предоставляет доступ к TransactionManager для текущей базы данных |

Таблица 1.7 – Методы класса TransactionManager

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращае-мый тип | Описание |
| Аргумент | Тип | Описание |
| Start  Transaction | – | – | – | Transaction | Начинает новую транзакцию |

Таблица 1.8 – Методы класса Transaction

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращаемый тип | Описание |
| Аргумент | Тип | Описание |
| Add  Newly  Created  Object | obj | DBObject | Объект для добавления в транзакцию или удаления из транзакции | void | Добавляет объект в транзакцию или удаляет его из транзакции |
| add | bool | Показывает, что нужно сделать с объектом (добавить или удалить) |

Окончание таблицы 1.8

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращаемый тип | Описание |
| Аргумент | Тип | Описание |
| GetObject | id | ObjectId | Уникальный идентификатор объекта в транзакции | DBObject | Получает объект, на данный момент открытый в транзакции, по его id |
| mode | OpenMode | Объявляет права доступа к объекту (чтение, запись, уведомление) |
| Commit | – | – | – | void | Фиксирует изменения, внесенные во все объекты DBObject, открытые в течение транзакции, после чего закрывает их |

Таблица 1.9 – Свойства класса DBObject

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| ObjectId | ObjectId | Предоставляет доступ к уникальному идентификатору объекта |

Таблица 1.10 – Методы класса DBObject

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращае-мый тип | Описание |
| Аргумент | Тип | Описание |
| Erase | – | – | – | void | Устанавливает бит стирания объекта |
| UpgradeOpen | – | – | – | void | Если в данный момент объект открыт только для чтения (OpenMode.ForRead), данный метод обновляет объект, чтобы открыть его для записи (OpenMode.ForWrite). В процессе изменения доступа объект закрывается, тем самым вызывая любое соответствующее уведомление |

Таблица 1.11 – Методы класса BlockTableRecord

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращаемый тип | Описание |
| Аргумент | Тип | Описание |
| Append  Entity | entity | Entity | Добавляемая сущность | ObjectId | Добавляет сущность в базу данных и в таблицу блоков |
| GetBlock  ReferenceIds | directOnly | bool | Показывает, включать ли в список возвращаемых ссылок только те ссылки на блоки, которые имеют прямые ссылки на данную BlockTableRecord. Если данное значение равно true и блок является вложенным, родительские ссылки на данный блок не будут учитываться | ObjectId  Collection | Возвращает список ссылок на блоки, которые прямо или косвенно (через вложенность блоков) ссылаются на данный блок |
| force  Validity | bool | Показывает, должны ли старые чертежи, которые были загружены по требованию, быть загружены полностью, чтобы найти их идентификаторы BlockReferences. Данный параметр применим, только если directOnly равно false |

Таблица 1.12 – Свойства класса Brep

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| Edges | BrepEdgeCollection | Предоставляет доступ ко всем ребрам, связанным с данным объектом |
| Vertices | BrepVertexCollection | Предоставляет доступ ко всем вершинам, связанным с данным объектом |

Таблица 1.13 – Свойства класса BrepEntity

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| SubentityPath | FullSubentityPath | Предоставляет доступ к полному пути подсущности в AutoCAD для текущего объекта |

Таблица 1.14 – Методы класса Solid3D

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращае-мый тип | Описание |
| Аргумент | Тип | Описание |
| Extrude | region | Region | Область выдавливания | void | Создает твердое тело посредством выдавливания области region на заданное расстояние по высоте height с заданным углом конусности taperAngle |
| height | double | Высота выдавливания |
| taperAngle | double | Угол конусности |
| FilletEdges | subentityIds | Subentity  Id[] | Идентификаторы объектов ребер, к которым будет применено закругление | void | Создает закругления на ребрах объекта |
| radius | Double  Collection | Радиус закругления на соответствующем ребре объекта |
| startSetback | Double  Collection | Начальный отступ закругления на соответствующем ребре объекта |
| endSetback | Double  Collection | Конечный отступ закругления на соответствующем ребре объекта |

Окончание таблицы 1.14

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращае-мый тип | Описание |
| Аргумент | Тип | Описание |
| Revolve | region | Region | Объект области, подлежащей вращению | void | Создает твердое тело путем вращения области вокруг оси, определяемой проекцией линии от точки оси axisPoint до точки axisPoint + axisDir на плоскость, содержащую область. Таким образом, ось вращения всегда будет компланарной области. Ось вращения не должна проходить через область |
| axisPoint | Point3d | Точка на линии, которая будет проецироваться для создания оси вращения |
| axisDir | Vector  3d | Вектор, представляющий направление линии, которая будет проецироваться для создания оси вращения |
| angleOf  Revolution | double | Угол вращения в радианах |
| Boolean  Operation | type | Boolean  Operation  Type | Тип булевой операции (объединение, пересечение, вычитание) | void | Выполняет булеву операцию между двумя 3D-объектами |
| solid | Solid3d | Второй 3D-объект |

Таблица 1.15 – Свойства класса Entity

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| GeometricExtents | Extents3d | Обращается к угловым точкам (в системе координат WCS) прямоугольной области (с ребрами, параллельными осям X, Y и Z в WCS), которая охватывает 3D-объем сущности, и возвращает эти точки в виде экземпляра класса Extents3d |

Таблица 1.16 – Методы класса Entity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращае-мый тип | Описание |
| Аргумент | Тип | Описание |
| TransformBy | transform | Matrix3d | Входная матрица преобразования | void | Применяет матрицу преобразования к указанному объекту |
| SetDatabase  Defaults | – | – | – | void | Задает свойствам объекта (цвет, слой, тип линии, шкалу типов линий, видимость, толщину линии и т.п.) значения по умолчанию для той базы данных, в которой объект находится на данный момент. Если объект еще не является частью какой-либо базы данных, используется текущая база данных в редакторе AutoCAD |

Таблица 1.17– Методы класса SymbolTable

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращае-мый тип | Описание |
| Аргумент | Тип | Описание |
| Add | value | SymbolTable  Record | Значение идентификатора записи | ObjectId | Добавляет значение идентификатора записи как в базу данных, содержащую таблицу, так и в саму таблицу |
| Has | key | string | Ключ – имя записи в виде строки | bool | Возвращает значение true, если таблица содержит запись с именем, соответствующим ключу. Если совпадений не найдено, возвращается значение false |

Таблица 1.18 – Свойства класса SymbolTableRecord

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| Name | string | Возвращает название записи в символьной таблице |

Таблица 1.19 – Свойства класса ObjectIdCollection

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| Count | int | Возвращает общее количество объектов в коллекции |

Таблица 1.20 – Свойства класса DBObjectCollection

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| Count | int | Возвращает общее количество объектов в коллекции |

Таблица 1.21 – Методы класса DBObjectCollection

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращаемый тип | Описание |
| Аргумент | Тип | Описание |
| Add | value | DBObject | Объект для добавления в коллекцию | int | Добавляет объект value в коллекцию |
| Remove | value | DBObject | Объект, подлежащий удалению из коллекции | void | Удаляет объект value из коллекции |

Таблица 1.22 – Свойства класса Circle

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| Center | Point3d | Предоставляет доступ к центру окружности |
| Diameter | void | Предоставляет доступ к диаметру окружности |

Таблица 1.23 – Методы класса Region

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращаемый тип | Описание |
| Аргумент | Тип | Описание |
| CreateFrom  Curves | curve  Sergments | DBObject | Коллекция кривых для создания замкнутых циклов | DBObject  Collection | Создает набор объектов Region из замкнутых циклов, которые представлены кривыми |

Таблица 1.24 – Свойства класса Polyline

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| Closed | bool | Определяет, должна ли ломаная линия быть замкнутой (т.е. имеется ли отрезок, проведенный от последней вершины к первой) или нет |

Таблица 1.25 – Методы класса Polyline

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращаемый тип | Описание |
| Аргумент | Тип | Описание |
| AddVertexAt | index | int | Индекс вершины | void | Добавляет новую вершину к ломаной линии |
| pt | Point2d | Точка расположения вершины |
| bulge | double | Выпуклость вершины |
| startWidth | double | Начальная ширина вершины |
| endWidth | double | Конечная ширина вершины |

Таблица 1.26 – Свойства класса Vertex

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| Point | Point3d | Предоставляет доступ к базовой точке данной вершины как к объекту Point3d |

Таблица 1.27 – Свойства структуры ObjectId

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| IsNull | bool | Определяет, имеет ли идентификатор объекта значение null |
| Null | ObjectId | Предоставляет доступ к идентификатору null-объекта. Это свойство позволяет использовать более компактные выражения и помогает компилятору генерировать более производительный код (т.е компилятор «знает», что ObjectId равен нулю, вместо того чтобы искать null-объект) |

Таблица 1.28 – Свойства структуры FullSubentityPath

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| SubentId | SubentityId | Предоставляет доступ к постоянной копии встроенного идентификатора подсущности |

Таблица 1.29 – Свойства структуры Extents3d

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| MinPoint | Point3d | Возвращает минимальную точку |
| MaxPoint | Point3d | Возвращает максимальную точку |

Таблица 1.30 – Свойства структуры Point2d

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| X | double | Возвращает координату по оси X |
| Y | double | Возвращает координату по оси Y |

Таблица 1.31 – Методы структуры Point2d

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращае-мый тип | Описание |
| Аргумент | Тип | Описание |
| GetVectorTo | point | Point2d | Заданная точка | Vector2d | Вычисляет и возвращает вектор в двумерном пространстве относительно указанной точки |

Таблица 1.32 – Свойства структуры Point3d

|  |  |  |
| --- | --- | --- |
| Название | Возвращаемый тип | Описание |
| Origin | Point3d | Возвращает начальную точку трехмерного пространства |
| X | double | Возвращает координату по оси X |
| Y | double | Возвращает координату по оси Y |
| Z | double | Возвращает координату по оси Z |

Таблица 1.33 – Методы структуры Point3d

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращае-мый тип | Описание |
| Аргумент | Тип | Описание |
| GetVectorTo | point | Point3d | Заданная точка | Vector3d | Вычисляет и возвращает вектор в трехмерном пространстве относительно указанной точки |

Таблица 1.34 – Методы структуры Matrix3d

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Название | Входные аргументы | | | Возвращаемый тип | Описание |
| Аргумент | Тип | Описание |
| Displacement | vector | Vector3d | Входной вектор для вычисления смещения | Matrix3d | Возвращает вектор смещения объекта |
| Rotation | angle | double | Угол вращения | Matrix3d | Возвращает матрицу поворота на указанный угол вокруг указанной оси с вектором, проходящим через начальную точку вращения. Направление вращение определяется правилом правой руки |
| axis | Vector3d | Ось вращения |
| center | Point3d | Начальная точка вращения |

## Обзор аналогов

**Auto-Furniture**

Плагин Auto-Furniture – инструмент от компании FurnitureSoft, который работает с базой данных параметрических мебельных блоков для САПР AutoCAD. Данная программа упрощает и ускоряет проектирование мебели (рисунок 1.2).



Рисунок 1.2 – Интерфейс программы Auto-Furniture

В Auto-Furniture можно выбрать блок, наиболее близкий к проектируемому мебельному изделию, и задать ему размеры. Все изменения немедленно отражаются в окне предварительного просмотра. После нажатия кнопки «Вставить» (Paste) выбранный блок с заданными размерами вставляется в то место документа AutoCAD, которое указывает пользователь. Существует опция предварительного просмотра объекта с разных ракурсов (сверху, справа, спереди) для более удобной работы с программой.

Кроме размеров, плагин позволяет изменять и некоторые другие параметры мебельных изделий: типы полок, выдвижных ящиков, стеллажей; количество ящиков, полок или дверей. Также в Auto-Furniture есть дополнительные настройки для таких предметов, как кухни и   
шкафы-купе [11].

Примеры мебельных изделий, созданных в AutoCAD с помощью данного плагина, приведены на рисунке 1.3.



Рисунок 1.3 – Примеры мебельных изделий, созданных с помощью   
Auto-Furniture в САПР AutoCAD

**SymbolDesigner**

SymbolDesigner – инструмент для создания параметрических объектов в AutoCAD, разработанный компанией CAD Division. Параметрические объекты САПР рассматриваются в качестве скриптов языка PPM, а геометрические объекты – в качестве результатов выполнения скриптов языка PPM (рисунок 1.4).



Рисунок 1.4 – Интерфейс программы SymbolDesigner

Язык PPM также предоставляет средства для создания различных аннотаций, сложных размеров и любых других 2D-объектов. Следовательно, параметрический объект можно определить в точных единицах измерения.

SymbolDesigner позволяет ускорить процесс создания параметрических объектов, так как предоставляет возможность найти и исправить любые проблемы при проектировании модели; пользователь может визуализировать свой эскиз и просмотреть окончательные результаты сценария; можно также автоматизировать создание геометрических объектов с помощью встроенных инструментов. Кроме того, SymbolDesigner позволяет изменять параметры и точки параметров. Таким образом, можно наблюдать поведение параметрического объекта САПР в реальном времени [12].

# ОПИСАНИЕ ПРЕДМЕТА ПРОЕКТИРОВАНИЯ

Предметом проектирования является письменный стол.

Письменный стол [13] – стол для письменных работ и аналогичных занятий сидя. Под рабочей поверхностью письменного стола находятся выдвижные ящики, в которых обычно хранятся канцелярские принадлежности и документы.

Трехмерная модель письменного стола может применяться как для согласования его внешнего вида с заказчиком, так и для визуализации, изготовления рабочих чертежей и т. д.

К изменяемым параметрам модели относятся:

* + длина столешницы L1 (от 800 до 1200 мм);
  + ширина столешницы B (от 1/2 \* L1 до 750 мм);
  + высота столешницы H1 (от 30 до 40 мм);
  + тип ножек (с круглым или квадратным основанием);
  + высота ножек H2 (от 690 до 740 мм);
  + диаметр основания ножки D (от 50 до 70 мм), если выбрано круглое основание;
  + длина основания ножки A (от 50 до 70 мм), если выбрано квадратное основание;
  + количество ящиков для канцелярии N (от 3 до 5 шт);
  + длина ящиков для канцелярии L2 (от 250 до L1 \* 1/3 мм).

Параметры плагина имеют следующие зависимости:

* минимальное значение параметра B должно быть не меньше 1/2 значения параметра L1;
* выбор типа ножек определяет форму и размер их основания (для круглых ножек – диаметр, для квадратных ножек – длина);
* максимальное значение параметра L2 не должно превышать 1/3 значения параметра L1.

На рисунках 2.1 – 2.2 представлена 3D-модель письменного стола.



Рисунок 2.1 – 3D-модель письменного стола с круглыми основаниями ножек



Рисунок 2.2 – 3D-модель письменного стола с квадратными основаниями ножек

Чертеж моделируемого объекта приведен на рисунке 2.3.





Рисунок 2.3 – Чертеж письменного стола

В качестве дополнительного задания добавлена возможность выбирать тип ручек ящиков для канцелярии – ручку-рейлинг, ручку-скобу (задается по умолчанию) или ручку-кнопку. Внешний вид всех трех типов ручек показан на рисунках 2.4 – 2.6.

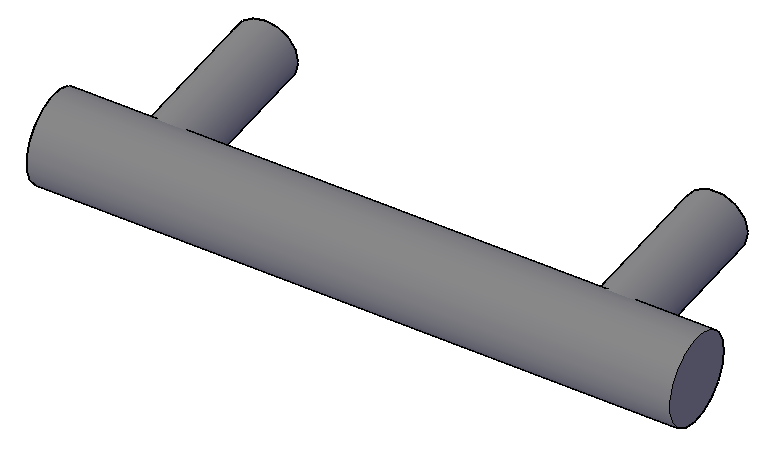


Рисунок 2.4 – Ручка-рейлинг

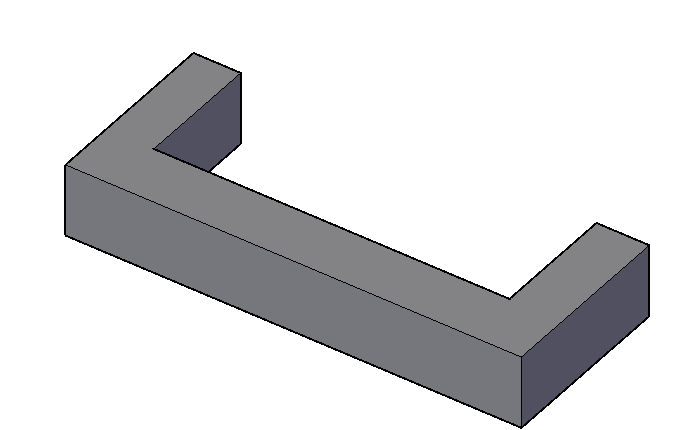


Рисунок 2.5 – Ручка-скоба

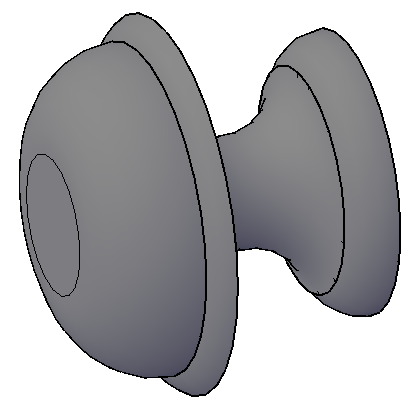


Рисунок 2.6 – Ручка-кнопка

Изменяемые параметры ручек ящиков для канцелярии:

* ручка-рейлинг: расстояние между крепежными ножками от 70 до 90 мм;
* ручка-скоба: расстояние между креплениями от 80 до 100 мм;
* ручка-кнопка: диаметр основания от 50 до 70 мм.

# ПРОЕКТ ПРОГРАММЫ

## Диаграмма классов

Диаграмма классов описывает типы объектов системы и различного рода статические отношения, которые существуют между ними. На диаграммах классов отображаются также свойства классов, операции классов и ограничения, которые накладываются на связи между объектами [14]. Целью создания диаграммы классов является графическое представление статической структуры декларативных элементов системы.

Диаграмма классов плагина «Письменный стол» представлена на рисунке 3.1.

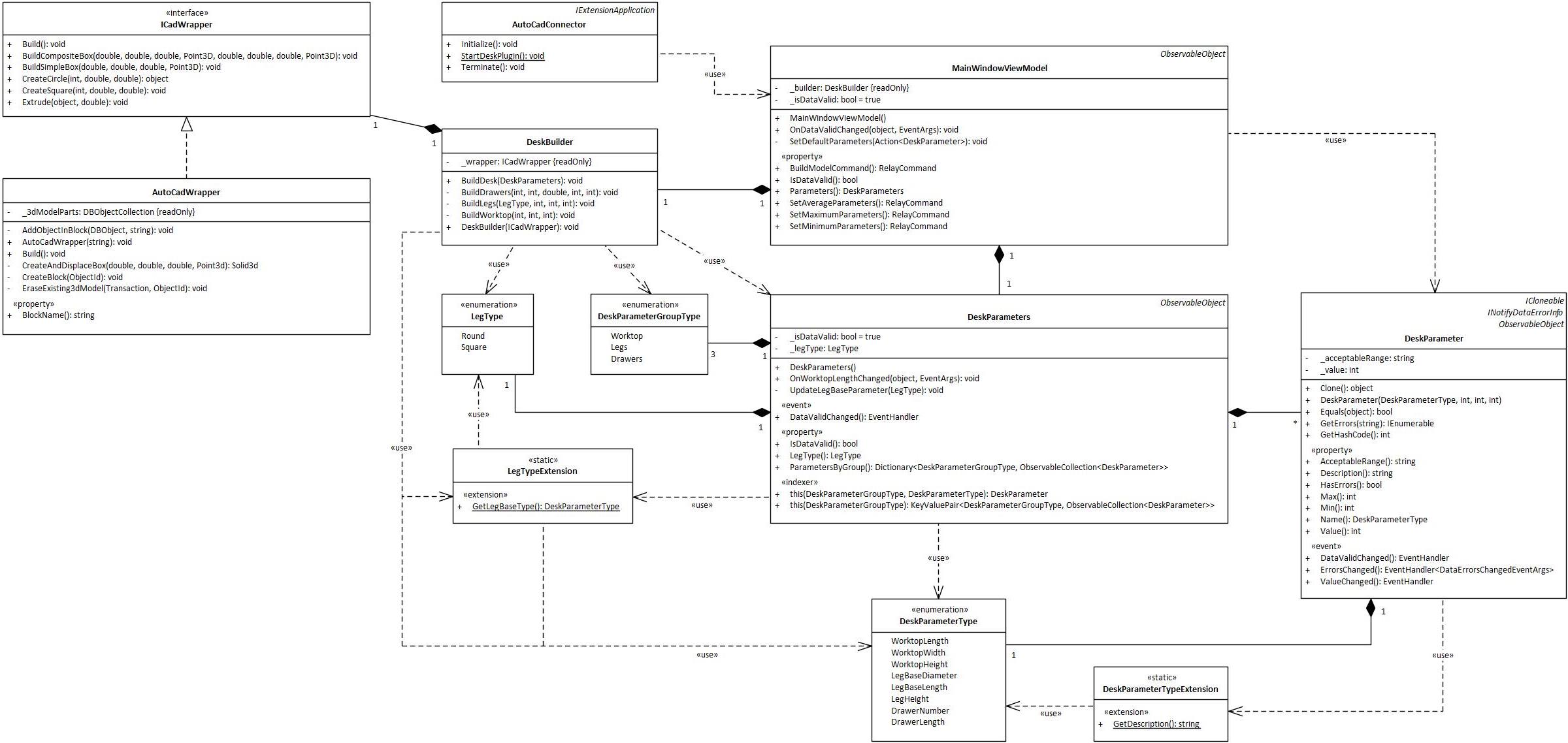


Рисунок 3.1 – UML-диаграмма классов плагина «Письменный стол»

Класс AutoCADConnector реализует интерфейс IExtensionApplication, необходимый для запуска плагина в программе AutoCAD.

Обобщенный класс DeskParameter<T> содержит данные о каком-либо числовом параметре модели, приведенном в перечисления ParametersType. Данный класс хранит минимально возможное и максимально возможное значение, которое может принимать параметр, а также его текущее значение. Свойство Name возвращает имя параметра, имеющее тип ParametersType. Свойство Description возвращает текстовое описание параметра (для этого используется метод статического класса ParametersTypeExtension, расширяющего перечисление ParametersType).

Все параметры модели письменного стола хранятся в классе DeskParameters. Данный класс содержит список числовых параметров модели, каждый элемент которого представляет собой объект DeskParameter<double>, а также хранит текущий тип ножек письменного стола, соответствующий элементу перечисления LegsType.

Для построения модели используется класс DeskBuilder, в котором метод BuildDesk() строит 3D-модель объекта на основании переданного ему списка параметров.

Итоговая диаграмма классов, построенная после выполнения дополнительного задания и рефакторинга кода, представлена на рисунке 3.2.

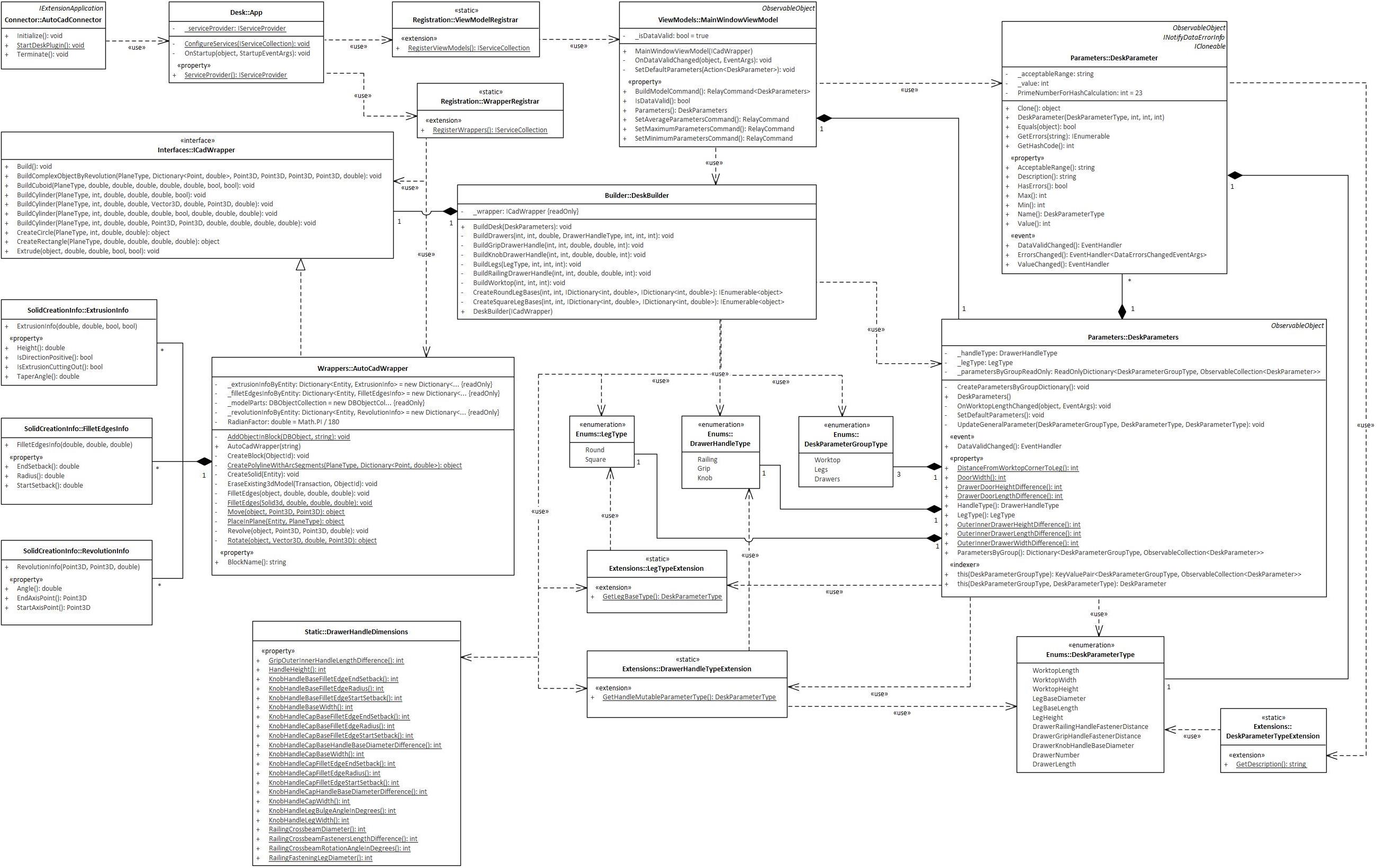


Рисунок 3.2 – Итоговая UML-диаграмма классов плагина «Письменный стол»

Как видно по рисунку 3.2, плагин «Письменный стол» разработан с применением принципа инверсии управления (Inversion of Control, IoC) и шаблона внедрения зависимостей (Dependency Injection, DI) для реализации данного принципа. Это помогает отделить выполнение задачи от ее реализации, упрощает переключение между различными реализациями (как в случае с различными реализациями интерфейса ICadWrapper при использовании разных САПР), повышает модульность программы и, следовательно, облегчает ее тестируемость за счет изолирования компонента или «подмены» (mocking) его зависимостей, а также предоставления компонентам возможности взаимодействовать через контракты [15].

Также принято решение ослабить зависимость между моделью представления главного окна плагина MainWindowViewModel и классом для построения 3D-модели DeskBuilder. Поскольку MainWindowViewModel напрямую использует из DeskBuilder только метод Build(), «привязывая» данный метод к одной из своих команд BuildModelCommand, то дальнейшее хранение объекта DeskBuilder в классе MainWindowViewModel можно расценить как нецелесообразное. Для реализации связи «использование» с построителем MainWindowViewModel использует интерфейс ICadWrapper.

Интерфейс классов-оболочек ICadWrapper в итоговом варианте разработки содержит гораздо больше сигнатур методов, чем предполагалось на этапе проектирования. Для избавления от дублирования кода в него в виде методов вынесены основные операции, выполняемые для построения трехмерных объектов в САПР (параллелепипеда, цилиндра и т.д.).

Класс-оболочка AutoCAD AutoCadWrapper теперь хранит двумерные детали письменного стола в разных словарях, в зависимости от того, каким образом из них должны быть получены трехмерные объекты: выдавливанием (класс ExtrusionInfo) или вращением (класс RevolutionInfo) и нужно ли закруглять ребра полученного 3D-объекта (класс FilletEdgesInfo).

Чтобы реализовать выбор типов ручек ящиков для канцелярии, введены дополнительные классы и перечисления. Перечисление DrawerHandleType содержит три типа ручек. Как и перечисление LegType, DrawerHandleType является частью класса параметров DeskParameters.

Перечисление DrawerHandleType использует статический класс DrawerHandleTypeExtension, который устанавливает связь между типом ручки из перечисления DrawerHandleType c типом параметра из перечисления DeskParameterType, т.к. в окне плагина необходимо изменять надписи к полю ввода изменяемого параметра в соответствии с выбором типа ручек.

С появлением дополнительного параметра HandleType в классе параметров письменного стола DeskParameters появилась потребность в раздельном хранении перечня параметров, которые отображаются пользователю и которые он может изменять, и более «общего» перечня параметров, содержащего все возможные параметры с их ограничениями и недоступного для внешнего изменения. По этой причине создан закрытый, доступный только для чтения словарь групп параметров и их коллекций \_parametersByGroupReadOnly.

Ранее для изменения надписи к полю ввода параметра, связанного с ножками письменного стола, а также определения, какого типа ножки должны быть построены с введенными пользователем размерами, предполагалось использовать метод UpdateLegBaseParameter(). Однако с появлением типа ручек ящиков для канцелярии HandleType стало необходимо совершать подобные действия еще и с перечислением DrawerHandleType. Таким образом, в класс DeskParameters добавлен метод UpdateGeneralParameter(), применяемый как для свойства LegType, так и для свойства HandleType.

Появление большого числа целочисленных констант, используемых для построения ручки-кнопки, привело к выносу их в отдельный статический класс DrawerHandleDimensions, который используется классом DeskBuilder в процессе построения того или иного типа ручек.

## Макеты пользовательского интерфейса

Макет пользовательского интерфейса представлен на рисунке 3.2.

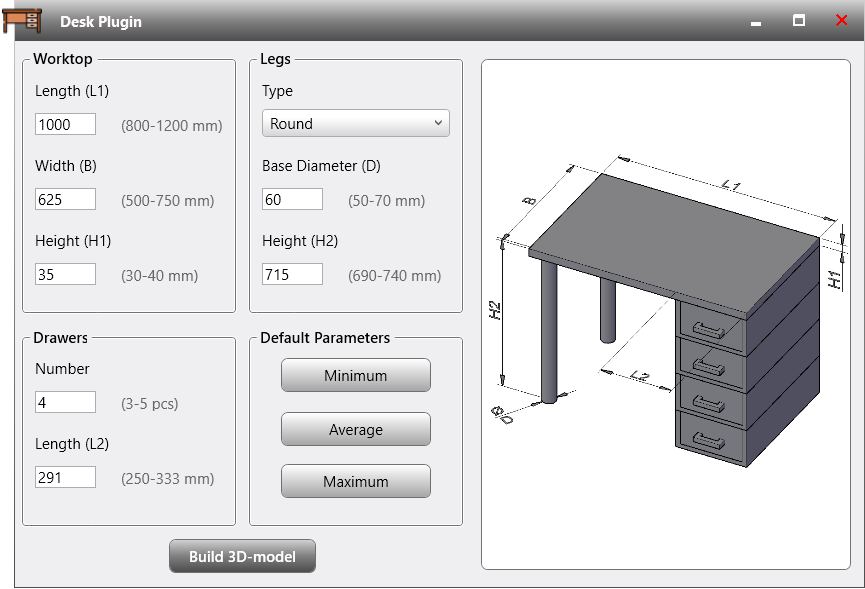


Рисунок 3.2 – Макет пользовательского интерфейса

Окно пользовательского интерфейса (рисунок 3.2) содержит:

* поля для ввода параметров модели письменного стола;
* надписи рядом с полями ввода, указывающие допустимый диапазон значений параметров;
* изображение модели письменного стола с буквенными обозначениями параметров;
* набор кнопок для задания параметров по умолчанию;
* кнопку построения модели, по нажатию на которую в AutoCAD будет производиться построение модели.

Все элементы пользовательского интерфейса сгруппированы по своему функциональному назначению для удобства взаимодействия с программой: так, длина, ширина и высота столешницы образуют группу параметров для построения столешницы (Worktop), кнопки задания параметров по умолчанию выделены в отдельную группу (Default Parameters) и т.д.

Кнопки в группе параметров по умолчанию (Default Parameters) позволяют выбрать минимально возможные, усредненные (задаются по умолчанию при запуске плагина) и максимально возможные параметры модели. Пример задания параметров по умолчанию приведен на рисунке 3.3.

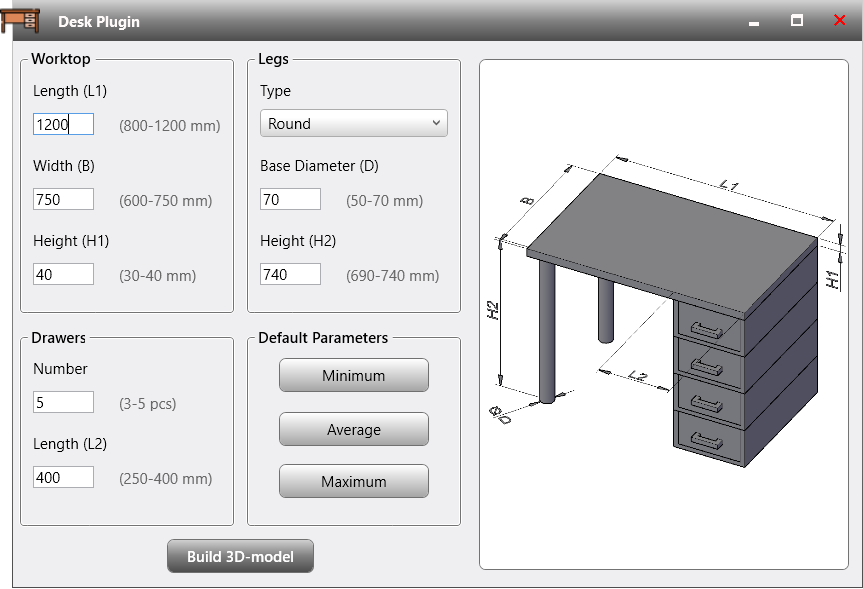


Рисунок 3.3 – Пример задания максимальных параметров модели письменного стола

Если пользователь вводит недопустимые значения параметров, кнопка построения модели становится недоступной, а поле для ввода, содержащее недопустимое значение параметра, выделяется красным цветом. Для большего привлечения внимания пользователя к допущенной ошибке справа от поля ввода отображается значок – красный круг с белым восклицательным знаком внутри. При наведении курсора мыши на поле, в котором произошла ошибка, появляется всплывающая подсказка с пояснением ошибки. Кроме того, поля для ввода параметров, зависимых от данного параметра, также подсвечиваются красным цветом, а справа от них на месте надписи с диапазоном допустимых значений появляется надпись Error (рисунок 3.4).

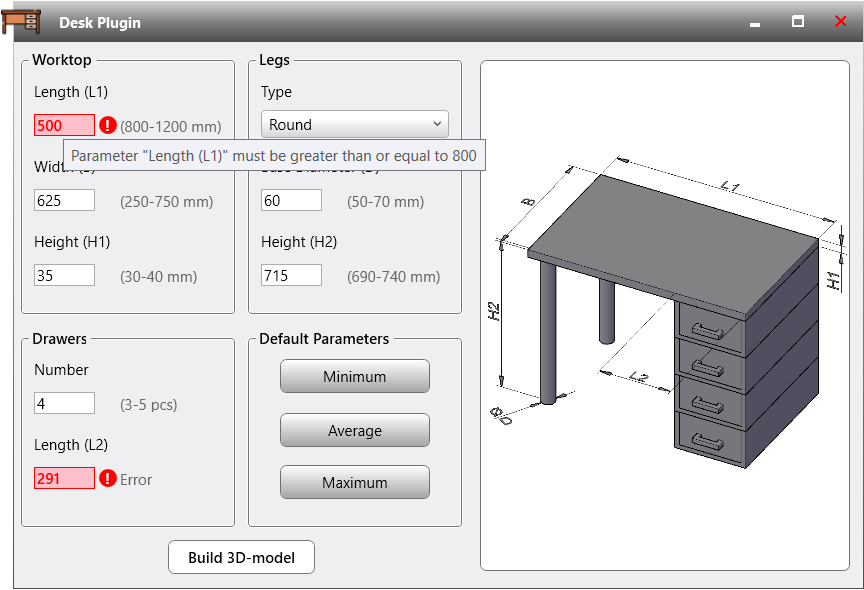


Рисунок 3.4 – Пример обработки ввода значения параметра, выходящего за пределы допустимого диапазона

По умолчанию выбранный тип ножек письменного стола – «круглые» (Round). В зависимости от выбранного из выпадающего списка типа ножек стола изменяется надпись над полем ввода размера основания ножек, а также обновляется изображение 3D-модели письменного стола, как показано на рисунках 3.5 – 3.6.

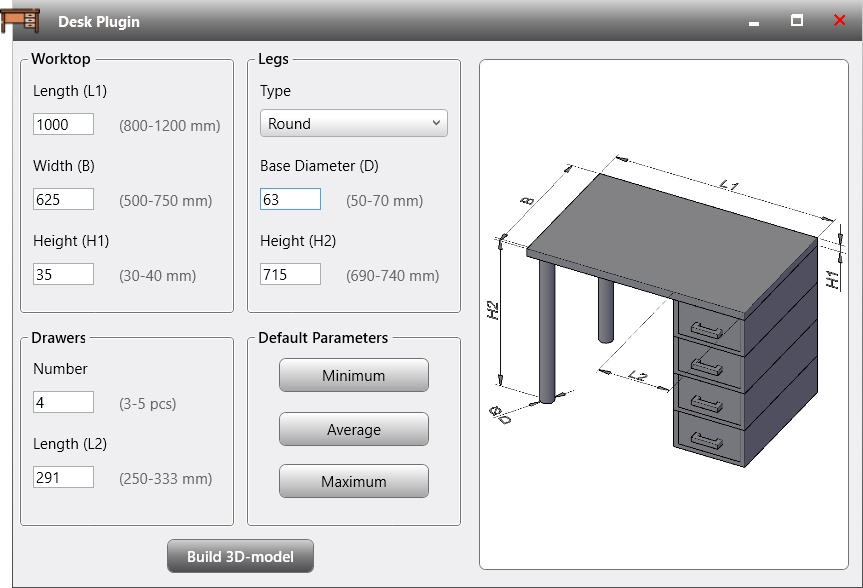


Рисунок 3.5 – Изменение надписи к полю ввода размера основания ножек и изображения 3D-модели письменного стола при выборе круглого типа основания

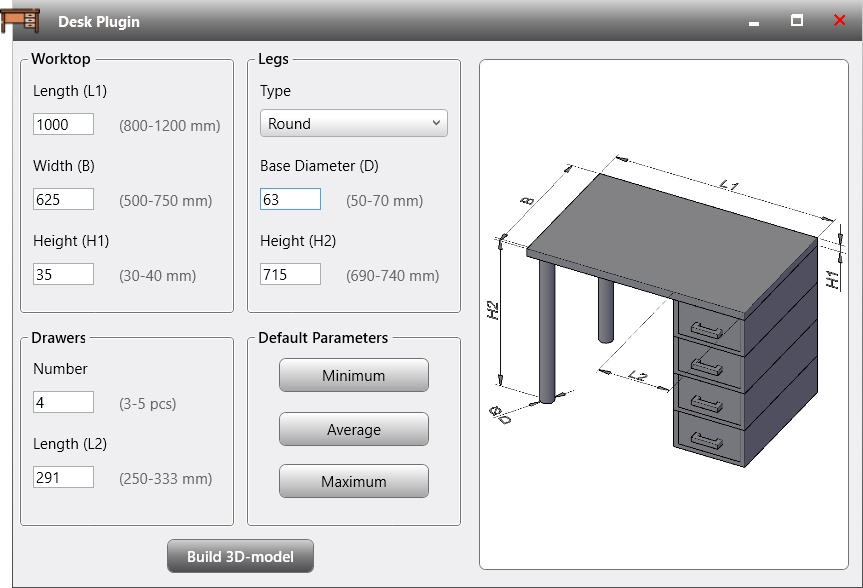


Рисунок 3.6 – Изменение надписи к полю ввода размера основания ножек и изображения 3D-модели письменного стола при выборе квадратного типа основания

Кроме того, при возникновении ошибки в текстовом поле для ввода размера основания ножек письменного стола содержание всплывающей подсказки об ошибке изменяется в зависимости от того, какой тип ножек выбран пользователем в данный момент (рисунки 3.7 – 3.8).

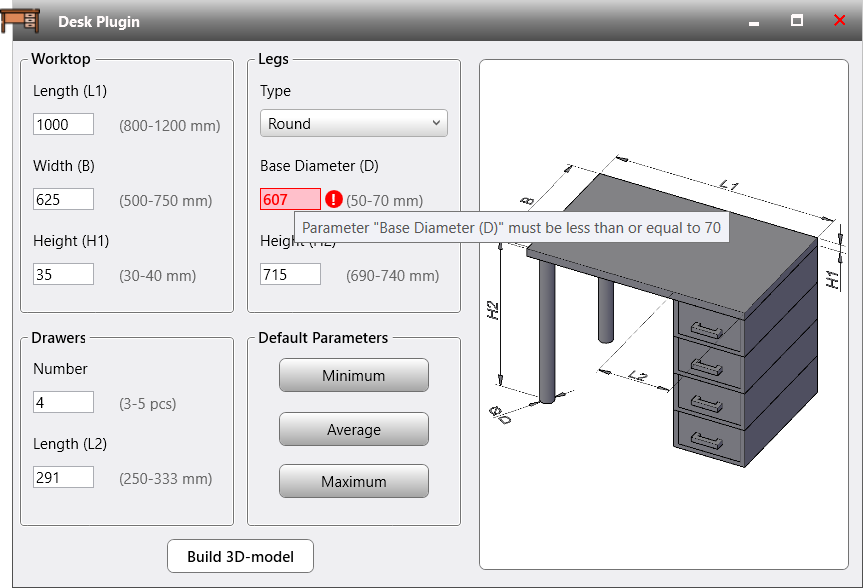


Рисунок 3.7 – Содержание всплывающей подсказки в момент, когда пользователем выбрано круглое основание ножек

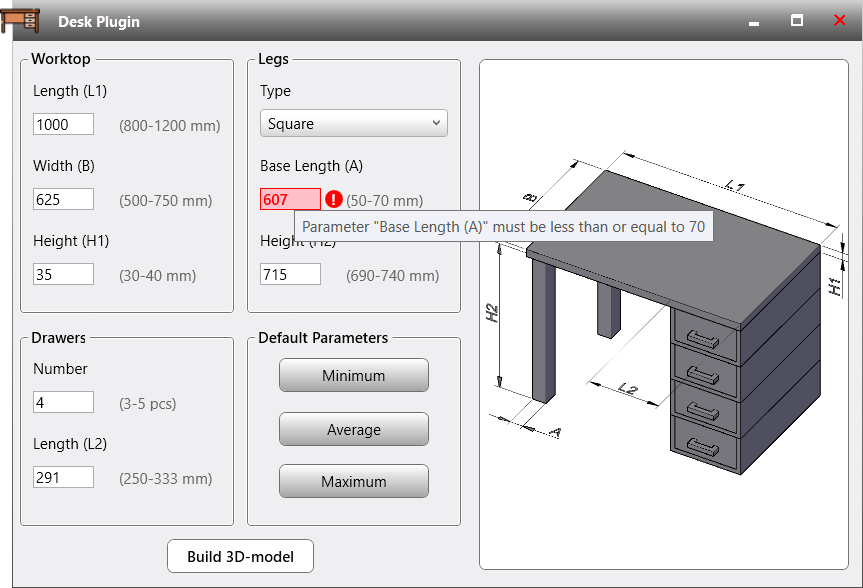


Рисунок 3.8 – Содержание всплывающей подсказки в момент, когда пользователем выбрано квадратное основание ножек

После реализации дополнительной функциональности в пользовательский интерфейс плагина внесены следующие изменения:

* в группу параметров «Ящики для канцелярии» (Drawers) добавлен выпадающий список, из которого можно выбрать тип ручек ящиков для канцелярии;
* под данным выпадающим списком размещено текстовое поле для ввода параметра, связанного с тем или иным типом ручек.

По умолчанию выбранный тип ручек ящиков для канцелярии письменного стола – «ручка-скоба» (Grip). В зависимости от выбранного из выпадающего списка типа ручек ящиков изменяется надпись над полем ввода изменяемого параметра ручек, а также обновляется изображение 3D-модели письменного стола, как показано на рисунках 3.9 – 3.11.

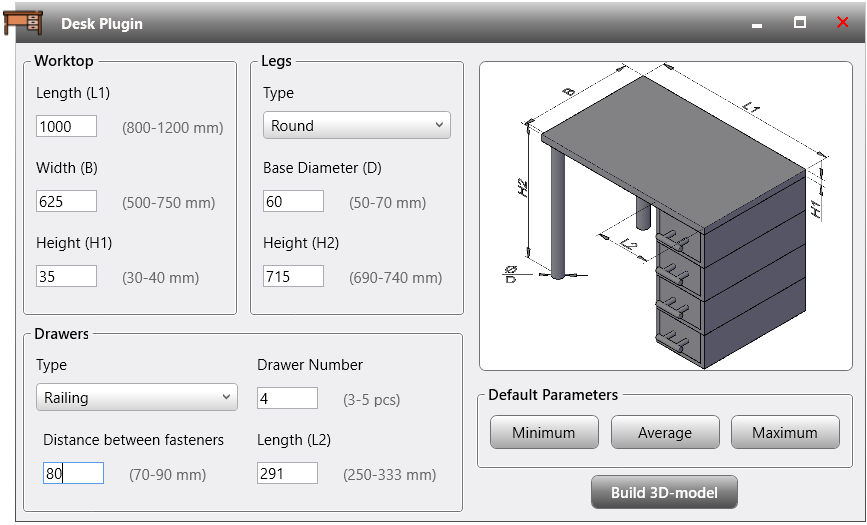


Рисунок 3.9 – Изменение надписи к полю ввода изменяемого параметра ручек ящиков для канцелярии и изображения 3D-модели письменного стола при выборе ручки-рейлинг

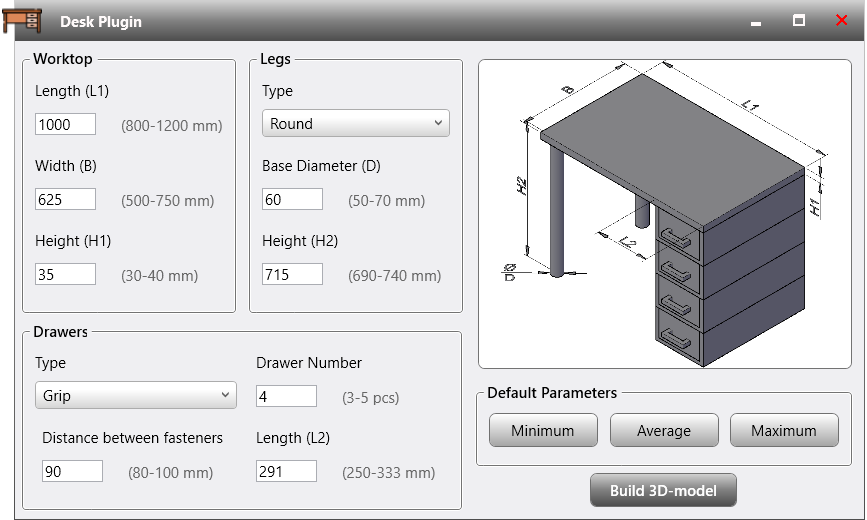


Рисунок 3.10 – Изменение надписи к полю ввода изменяемого параметра ручек ящиков для канцелярии и изображения 3D-модели письменного стола при выборе ручки-скобы

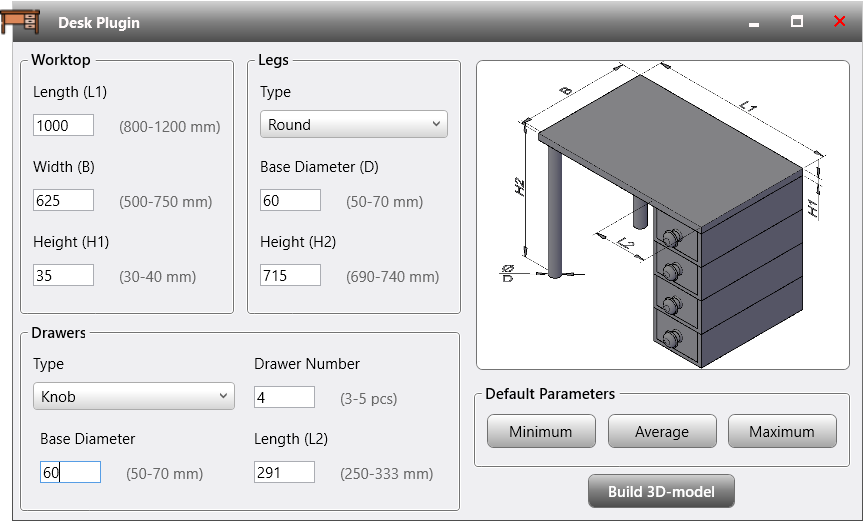


Рисунок 3.11 – Изменение надписи к полю ввода изменяемого параметра ручек ящиков для канцелярии и изображения 3D-модели письменного стола при выборе ручки-кнопки

# ТЕСТИРОВАНИЕ ПЛАГИНА

## Модульное тестирование

Юнит-тестирование – это тестирование отдельного элемента изолированно от остальной системы. Относительно парадигмы объектно-ориентированного программирования системой является вся программа, а отдельным элементом – класс или его метод. Юнит-тестирование предназначено для проверки правильности работы отдельно взятого класса. Чтобы исключить из результатов тестирования влияние потенциальных ошибок других классов, тестируемый класс должен быть максимально изолирован, т. е. не использовать объекты и методы других классов. Данное требование в итоге позволяет иначе взглянуть на взаимодействие классов и выполнить рефакторинг на уменьшение связности классов [16].

В целях проверки корректности свойств и методов классов бизнес-логики плагина «Письменный стол» проведено модульное тестирование классов проекта параметров письменного стола Parameters, а также классов проекта моделей представления ViewModels. Тестирование проведено для всех открытых свойств и методов классов данных проектов при помощи библиотеки модульного тестирования для языков платформы .NET. NUnit версии 3.13.3. Всего написано 80 юнит-тестов.

Результаты прохождения данных тестов представлены на   
рисунках 4.1 – 4.8.

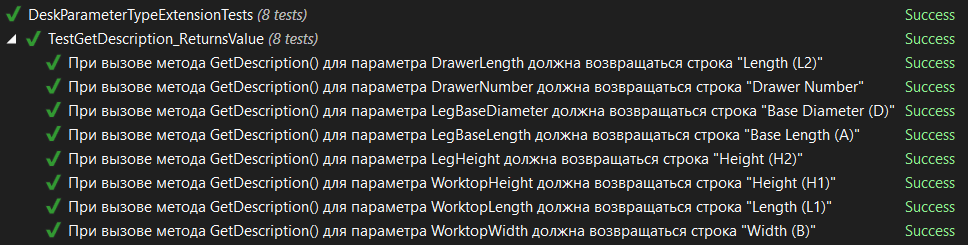


Рисунок 4.1 – Список пройденных модульных тестов для класса DeskParameterTypeExtentsion

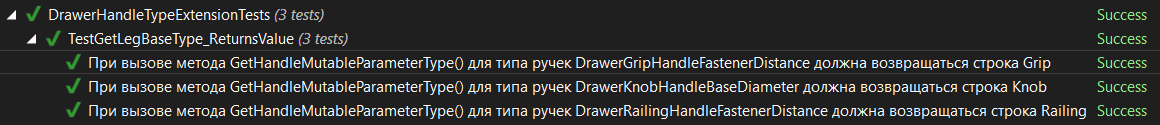


Рисунок 4.2 – Список пройденных модульных тестов для класса DrawerHandleTypeExtension

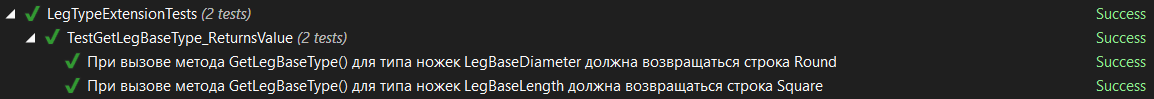


Рисунок 4.3 – Список пройденных модульных тестов для класса LegTypeExtension

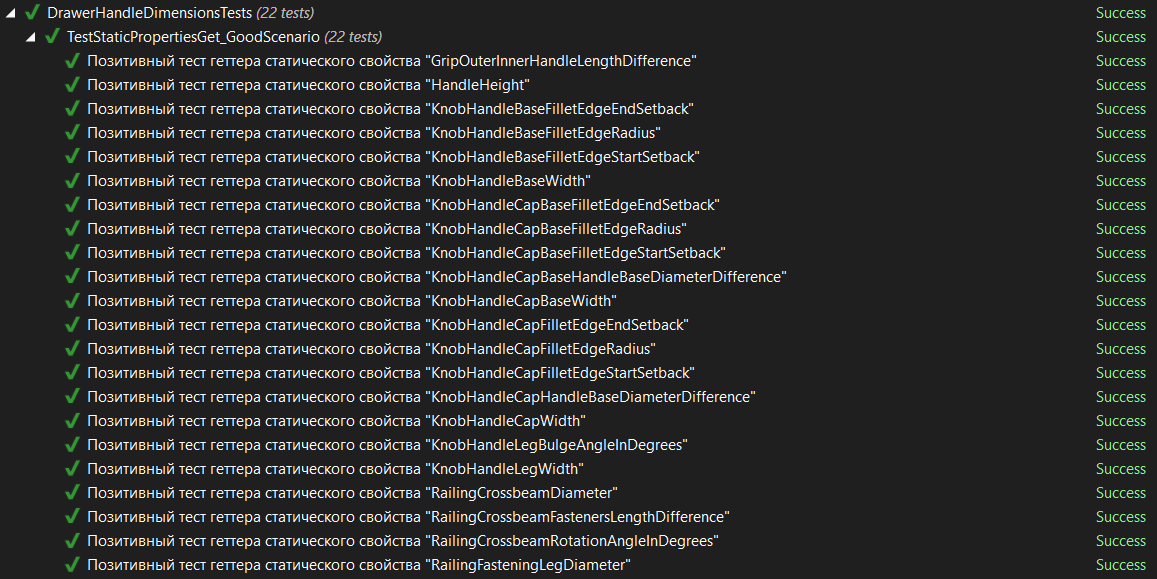


Рисунок 4.4 – Список пройденных модульных тестов для класса DrawerHandleDimensions

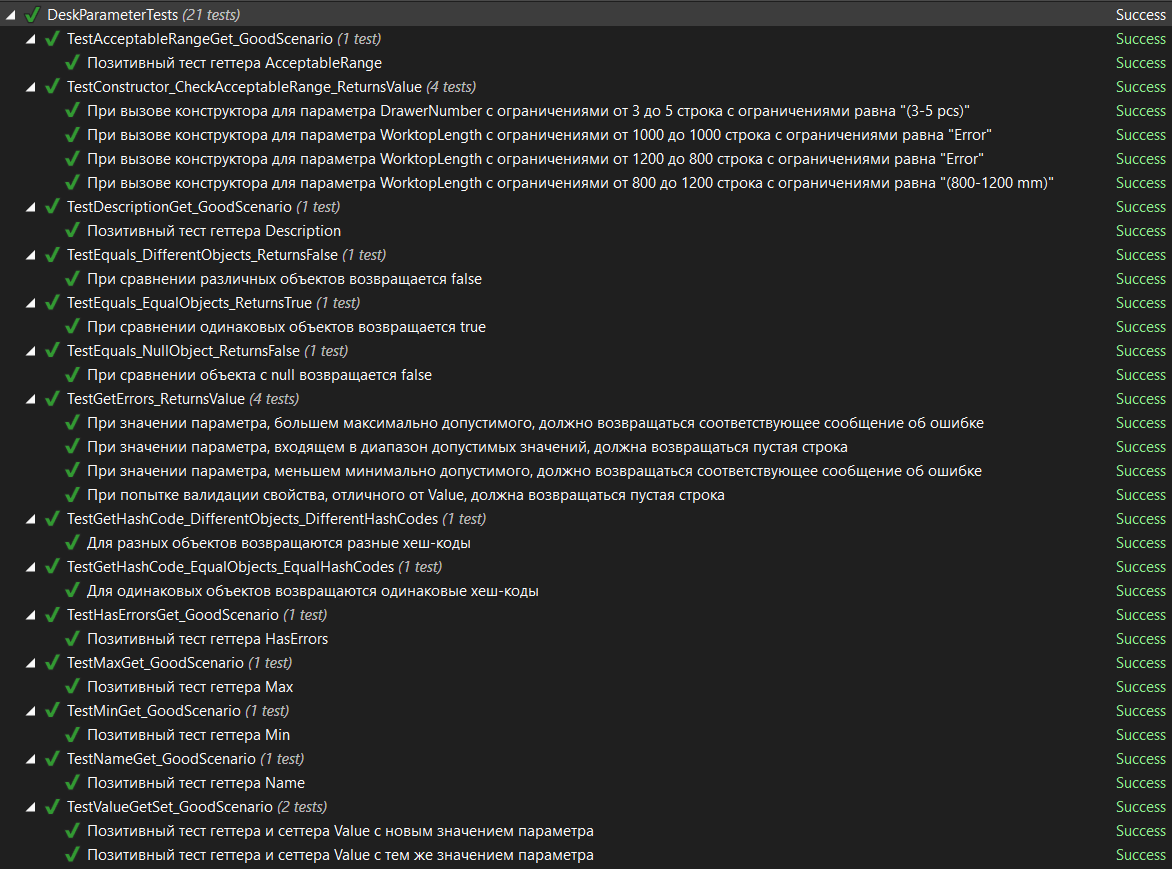


Рисунок 4.5 – Список пройденных модульных тестов для класса DeskParameter

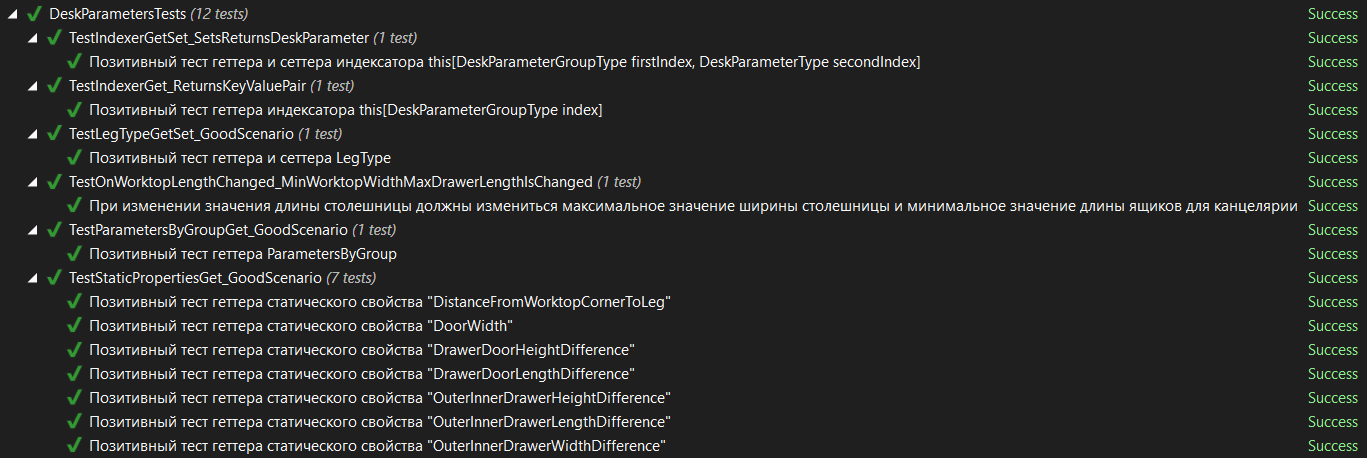


Рисунок 4.6 – Список пройденных модульных тестов для класса DeskParameters

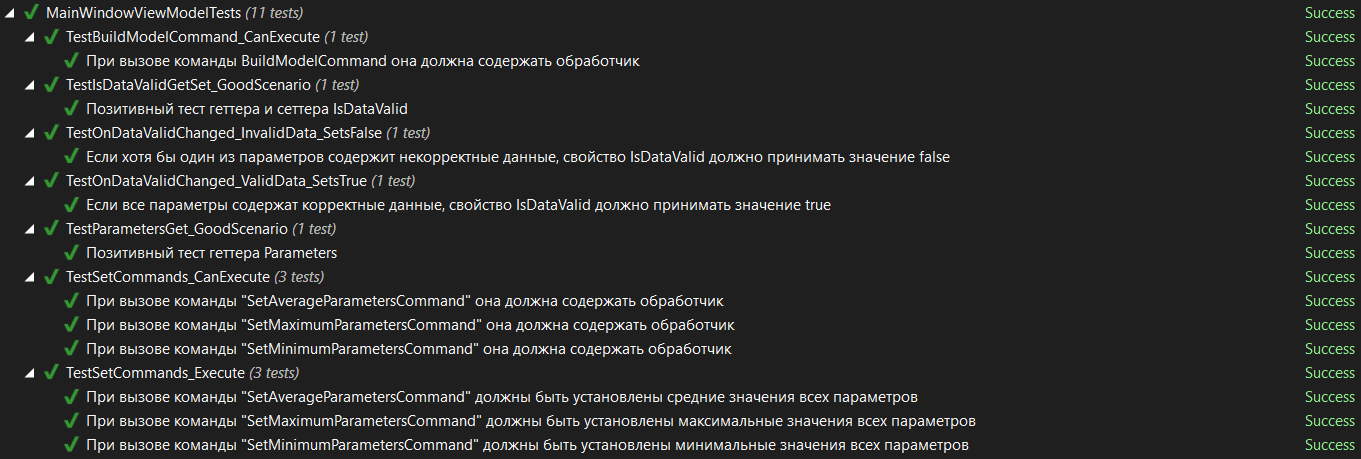


Рисунок 4.7 – Список пройденных модульных тестов для класса MainWindowViewModel

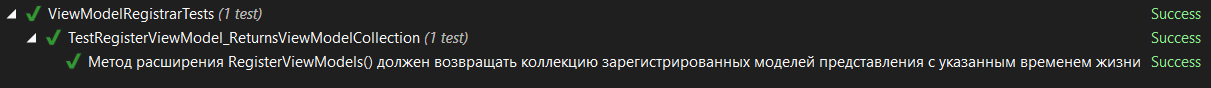


Рисунок 4.8 – Список пройденных модульных тестов для класса ViewModelRegistrar

Степень покрытия проектов ViewModels и Parameters модульными тестами приведена на рисунках 4.9 – 4.10.

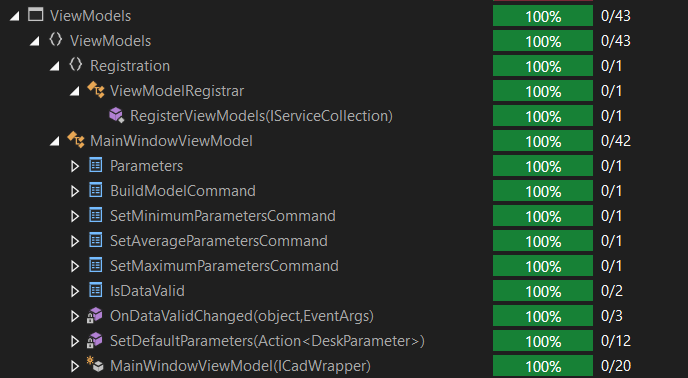


Рисунок 4.9 – Степень покрытия проекта ViewModels модульными тестами

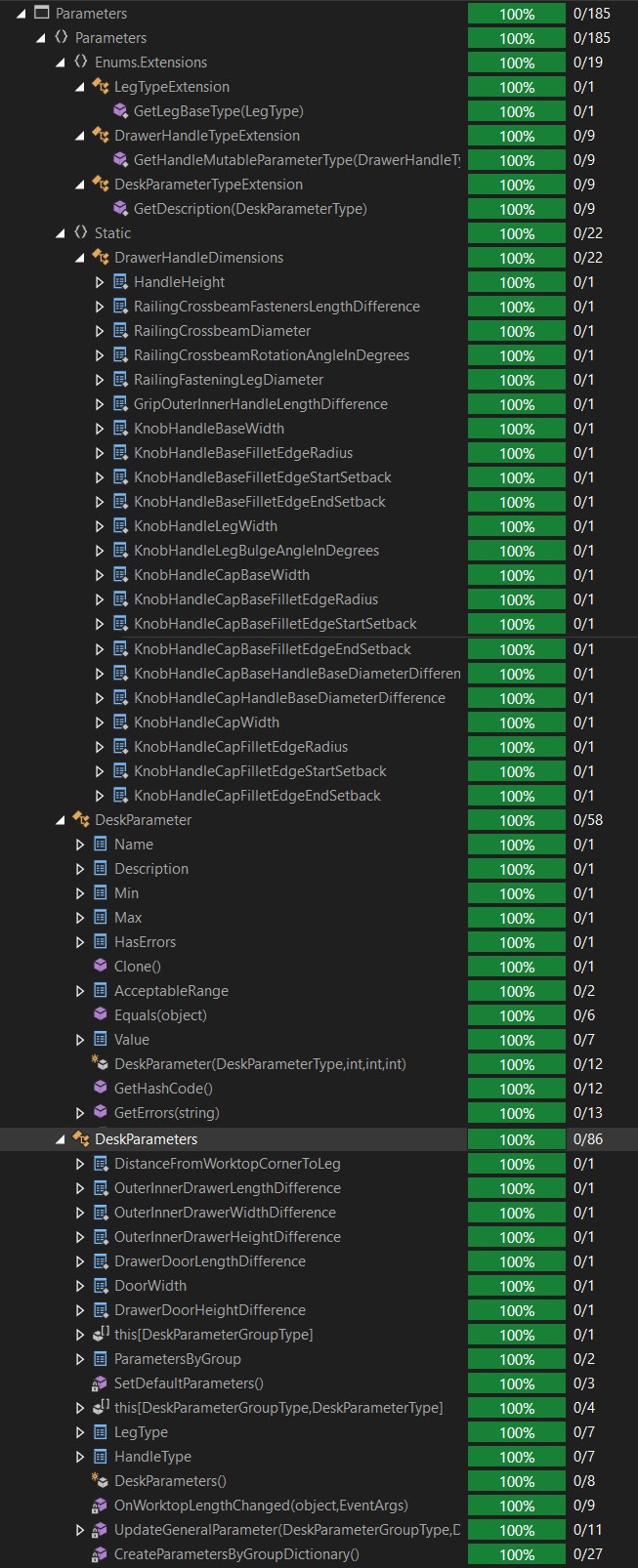


Рисунок 4.10 – Степень покрытия проекта Parameters модульными тестами

## Функциональное тестирование

Функциональное тестирование – это тестирование функциональности объекта, т. е. правильно ли объект выполняет свои функции. Фактически выполняется проверка правильности выходных данных при соответствующих входных данных [16].

При функциональном тестировании проверена корректность работы плагина «Письменный стол», а именно – соответствие полученного результата в виде трехмерной модели письменного стола входным параметрам.

На рисунке 4.11 приведен результат построения трехмерной модели письменного стола с круглыми основаниями ножек, ручками-рейлинг ящиков для канцелярии и минимальными параметрами:

* + длина столешницы L1 – 800 мм;
  + ширина столешницы B – 400 мм;
  + высота столешницы H1 – 30 мм;
  + диаметр основания ножек D – 50 мм;
  + высота ножек H2 – 690 мм;
  + расстояние между крепежными ножками ручки-рейлинг – 70 мм;
  + количество ящиков для канцелярии N – 3 шт;
  + длина ящиков для канцелярии L2 – 250 мм.

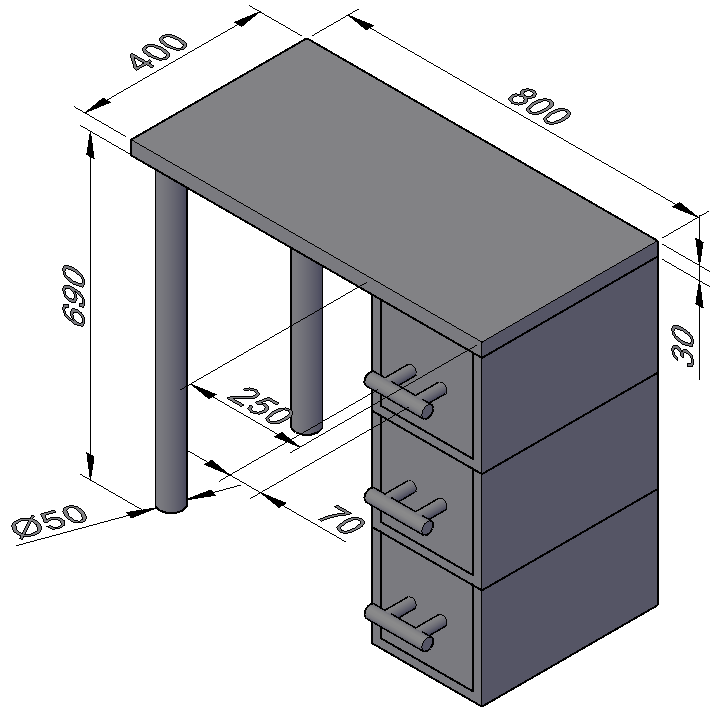


Рисунок 4.11 – Результат построения 3D-модели письменного стола c круглыми основаниями ножек, ручками-рейлинг и минимальными параметрами

На рисунке 4.12 приведен результат построения трехмерной модели письменного стола с круглыми основаниями ножек, ручками-скобами ящиков для канцелярии и минимальными параметрами:

* + длина столешницы L1 – 800 мм;
  + ширина столешницы B – 400 мм;
  + высота столешницы H1 – 30 мм;
  + диаметр основания ножек D – 50 мм;
  + высота ножек H2 – 690 мм;
  + расстояние между крепежными элементами ручки-скобы – 80 мм;
  + количество ящиков для канцелярии N – 3 шт;
  + длина ящиков для канцелярии L2 – 250 мм.

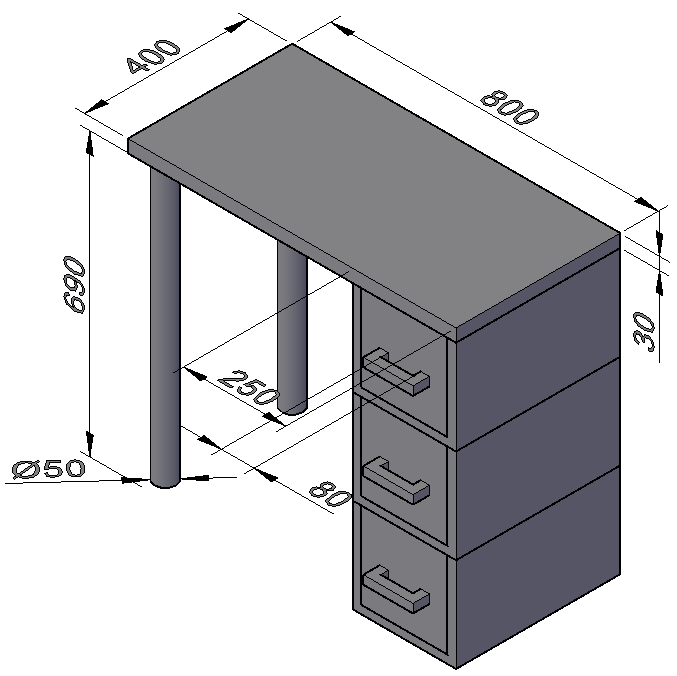


Рисунок 4.12 – Результат построения 3D-модели письменного стола c круглыми основаниями ножек, ручками-скобами и минимальными параметрами

На рисунке 4.13 приведен результат построения трехмерной модели письменного стола с круглыми основаниями ножек, ручками-кнопками ящиков для канцелярии и минимальными параметрами:

* + длина столешницы L1 – 800 мм;
  + ширина столешницы B – 400 мм;
  + высота столешницы H1 – 30 мм;
  + диаметр основания ножек D – 50 мм;
  + высота ножек H2 – 690 мм;
  + диаметр основания ручки-кнопки – 50 мм;
  + количество ящиков для канцелярии N – 3 шт;
  + длина ящиков для канцелярии L2 – 250 мм.

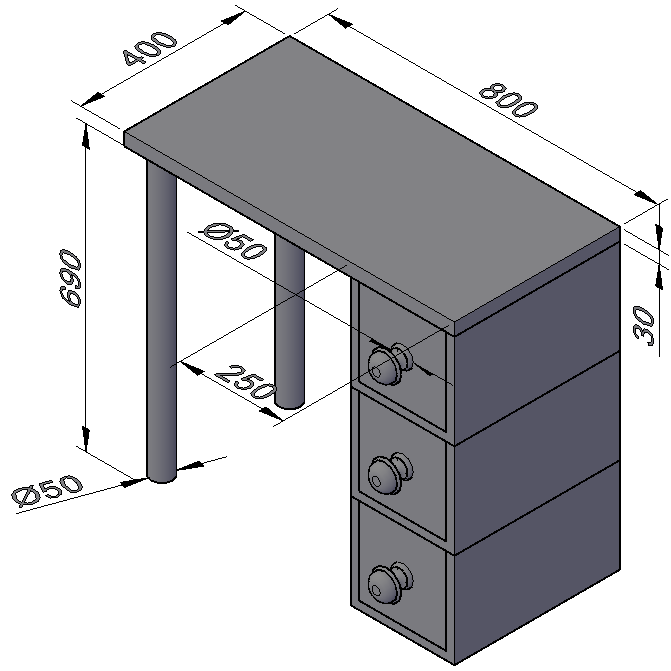


Рисунок 4.13 – Результат построения 3D-модели письменного стола c круглыми основаниями ножек, ручками-кнопками и минимальными параметрами

На рисунке 4.14 приведен результат построения трехмерной модели письменного стола с квадратными основаниями ножек, ручками-рейлинг ящиков для канцелярии и минимальными параметрами:

* + длина столешницы L1 – 800 мм;
  + ширина столешницы B – 400 мм;
  + высота столешницы H1 – 30 мм;
  + длина основания ножек A – 50 мм;
  + высота ножек H2 – 690 мм;
  + расстояние между крепежными ножками ручки-рейлинг – 70 мм;
  + количество ящиков для канцелярии N – 3 шт;
  + длина ящиков для канцелярии L2 – 250 мм.

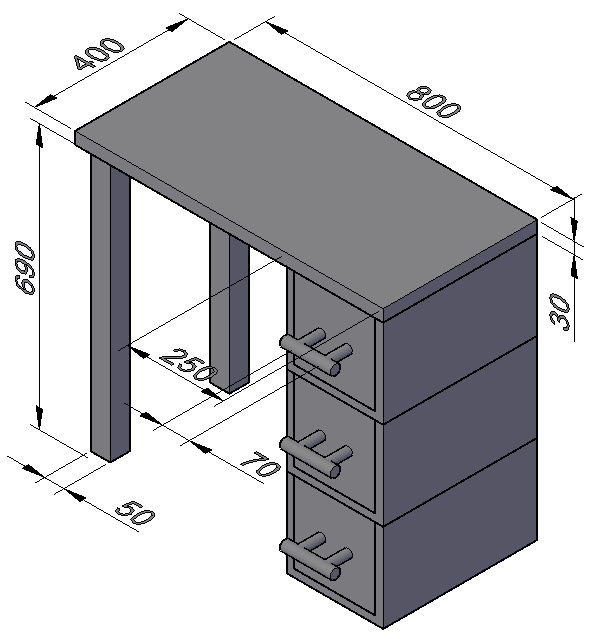


Рисунок 4.14 – Результат построения 3D-модели письменного стола c квадратными основаниями ножек, ручками-рейлинг и минимальными параметрами

На рисунке 4.15 приведен результат построения трехмерной модели письменного стола с квадратными основаниями ножек, ручками-скобами ящиков для канцелярии и минимальными параметрами:

* + длина столешницы L1 – 800 мм;
  + ширина столешницы B – 400 мм;
  + высота столешницы H1 – 30 мм;
  + длина основания ножек A – 50 мм;
  + высота ножек H2 – 690 мм;
  + расстояние между крепежными элементами ручки-скобы – 80 мм;
  + количество ящиков для канцелярии N – 3 шт;
  + длина ящиков для канцелярии L2 – 250 мм.

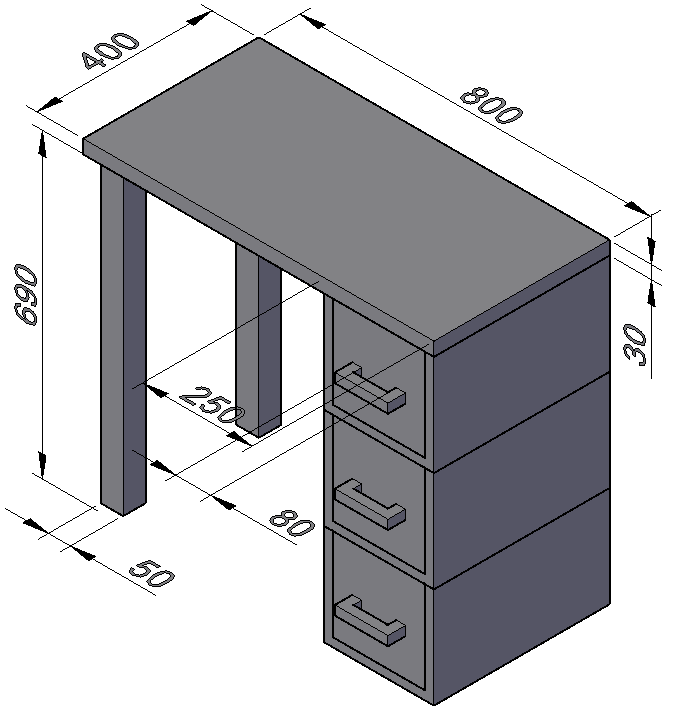


Рисунок 4.15 – Результат построения 3D-модели письменного стола c квадратными основаниями ножек, ручками-скобами и минимальными параметрами

На рисунке 4.16 приведен результат построения трехмерной модели письменного стола с квадратными основаниями ножек, ручками-кнопками ящиков для канцелярии и минимальными параметрами:

* + длина столешницы L1 – 800 мм;
  + ширина столешницы B – 400 мм;
  + высота столешницы H1 – 30 мм;
  + длина основания ножек A – 50 мм;
  + высота ножек H2 – 690 мм;
  + диаметр основания ручки-кнопки – 50 мм;
  + количество ящиков для канцелярии N – 3 шт;
  + длина ящиков для канцелярии L2 – 250 мм.

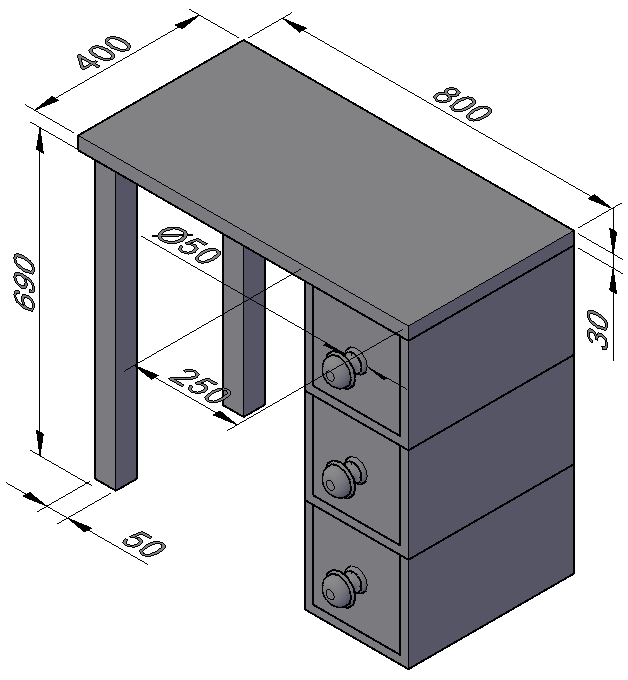


Рисунок 4.16 – Результат построения 3D-модели письменного стола c квадратными основаниями ножек, ручками-кнопками и минимальными параметрами

На рисунке 4.17 приведен результат построения трехмерной модели письменного стола с круглыми основаниями ножек, ручками-рейлинг ящиков для канцелярии и средними параметрами:

* + длина столешницы L1 – 1000 мм;
  + ширина столешницы B – 625 мм;
  + высота столешницы H1 – 35 мм;
  + диаметр основания ножек D – 60 мм;
  + высота ножек H2 – 715 мм;
  + расстояние между крепежными ножками ручки-рейлинг – 80 мм;
  + количество ящиков для канцелярии N – 4 шт;
  + длина ящиков для канцелярии L2 – 291 мм.

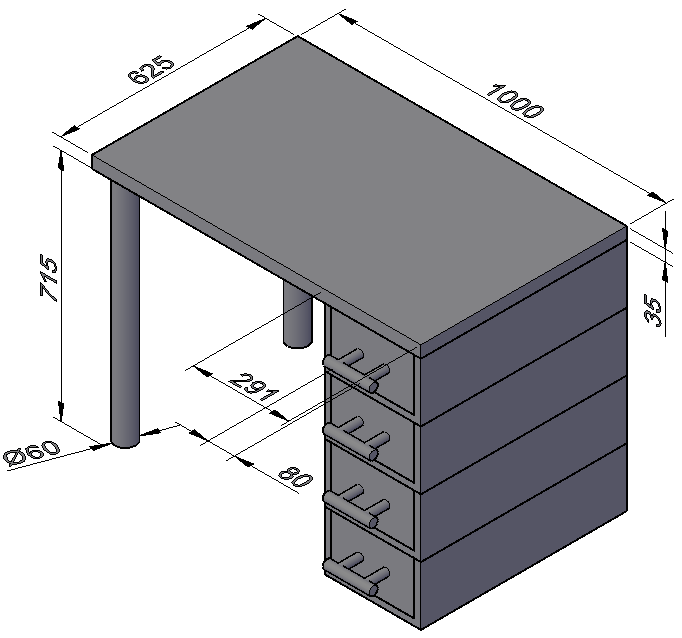


Рисунок 4.17 – Результат построения 3D-модели письменного стола c круглыми основаниями ножек, ручками-рейлинг и средними параметрами

На рисунке 4.18 приведен результат построения трехмерной модели письменного стола с круглыми основаниями ножек, ручками-скобами ящиков для канцелярии и средними параметрами:

* + длина столешницы L1 – 1000 мм;
  + ширина столешницы B – 625 мм;
  + высота столешницы H1 – 35 мм;
  + диаметр основания ножек D – 60 мм;
  + высота ножек H2 – 715 мм;
  + расстояние между крепежными элементами ручки-скобы – 90 мм;
  + количество ящиков для канцелярии N – 4 шт;
  + длина ящиков для канцелярии L2 – 291 мм.

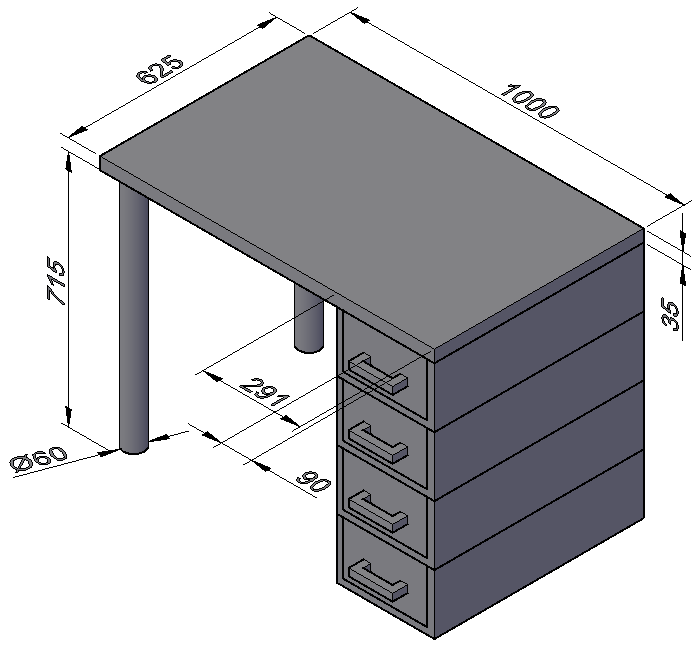


Рисунок 4.18 – Результат построения 3D-модели письменного стола c круглыми основаниями ножек, ручками-скобами и средними параметрами

На рисунке 4.19 приведен результат построения трехмерной модели письменного стола с круглыми основаниями ножек, ручками-кнопками ящиков для канцелярии и средними параметрами:

* + длина столешницы L1 – 1000 мм;
  + ширина столешницы B – 625 мм;
  + высота столешницы H1 – 35 мм;
  + диаметр основания ножек D – 60 мм;
  + высота ножек H2 – 715 мм;
  + диаметр основания ручки-кнопки – 60 мм;
  + количество ящиков для канцелярии N – 4 шт;
  + длина ящиков для канцелярии L2 – 291 мм.

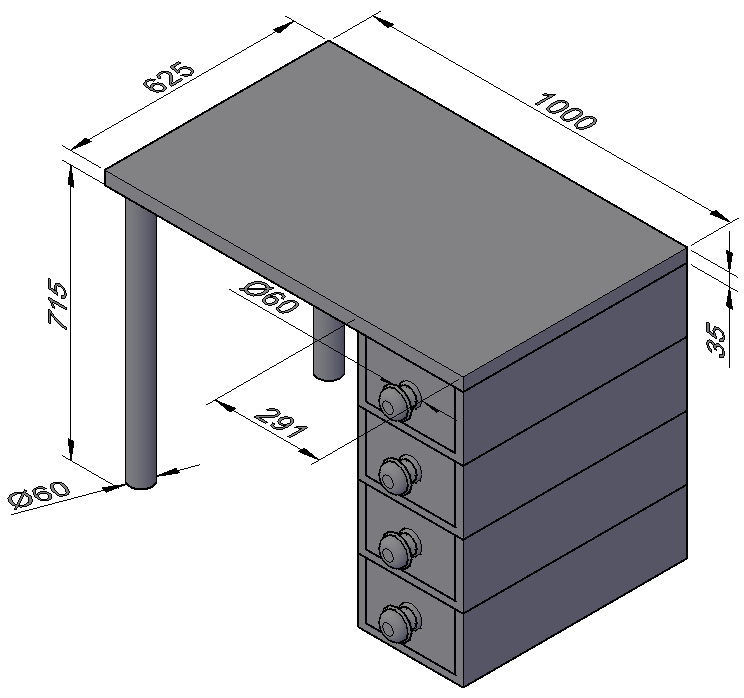


Рисунок 4.19 – Результат построения 3D-модели письменного стола c круглыми основаниями ножек, ручками-кнопками и средними параметрами

На рисунке 4.20 приведен результат построения трехмерной модели письменного стола с квадратными основаниями ножек, ручками-рейлинг ящиков для канцелярии и средними параметрами:

* + длина столешницы L1 – 1000 мм;
  + ширина столешницы B – 625 мм;
  + высота столешницы H1 – 35 мм;
  + длина основания ножек A – 60 мм;
  + высота ножек H2 – 715 мм;
  + расстояние между крепежными ножками ручки-рейлинг – 80 мм;
  + количество ящиков для канцелярии N – 4 шт;
  + длина ящиков для канцелярии L2 – 291 мм.

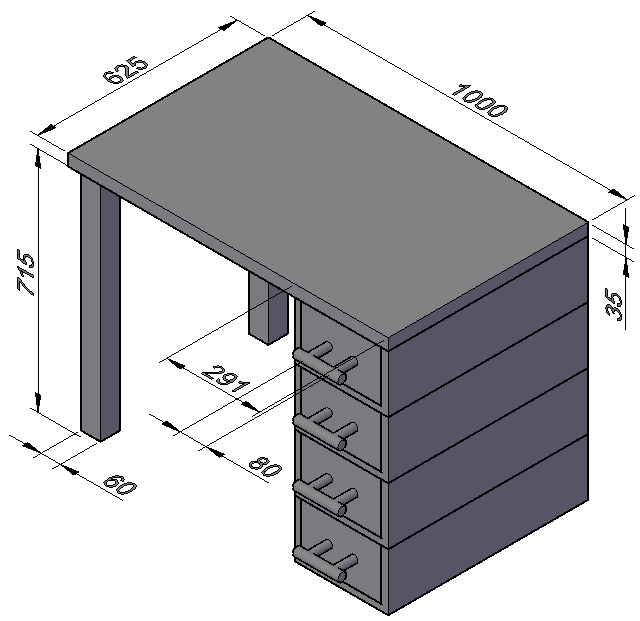


Рисунок 4.21 – Результат построения 3D-модели письменного стола c квадратными основаниями ножек, ручками-рейлинг и средними параметрами

На рисунке 4.22 приведен результат построения трехмерной модели письменного стола с квадратными основаниями ножек, ручками-скобами ящиков для канцелярии и средними параметрами:

* + длина столешницы L1 – 1000 мм;
  + ширина столешницы B – 625 мм;
  + высота столешницы H1 – 35 мм;
  + длина основания ножек A – 60 мм;
  + высота ножек H2 – 715 мм;
  + расстояние между крепежными элементами ручки-скобы – 90 мм;
  + количество ящиков для канцелярии N – 4 шт;
  + длина ящиков для канцелярии L2 – 291 мм.

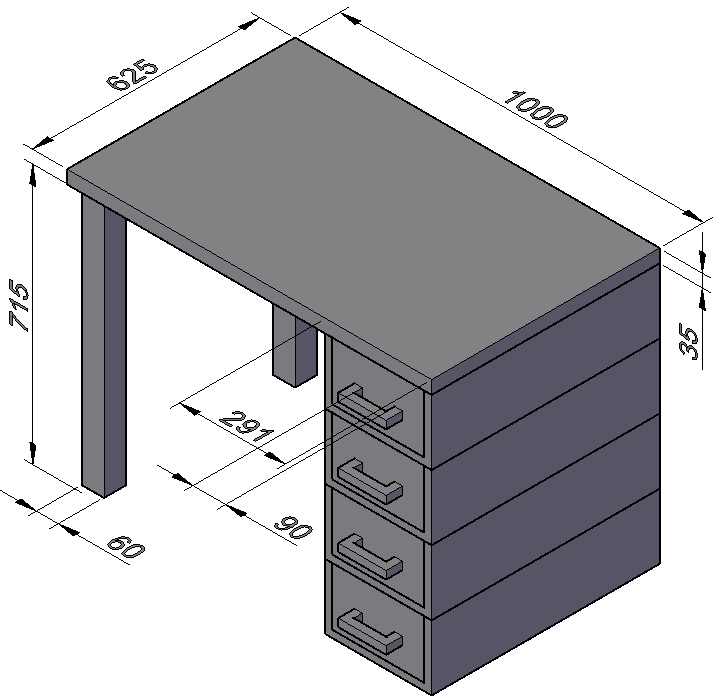


Рисунок 4.22 – Результат построения 3D-модели письменного стола c квадратными основаниями ножек, ручками-скобами и средними параметрами

На рисунке 4.23 приведен результат построения трехмерной модели письменного стола с квадратными основаниями ножек, ручками-кнопками ящиков для канцелярии и средними параметрами:

* + длина столешницы L1 – 1000 мм;
  + ширина столешницы B – 625 мм;
  + высота столешницы H1 – 35 мм;
  + длина основания ножек A – 60 мм;
  + высота ножек H2 – 715 мм;
  + диаметр основания ручки-кнопки – 60 мм;
  + количество ящиков для канцелярии N – 4 шт;
  + длина ящиков для канцелярии L2 – 291 мм.

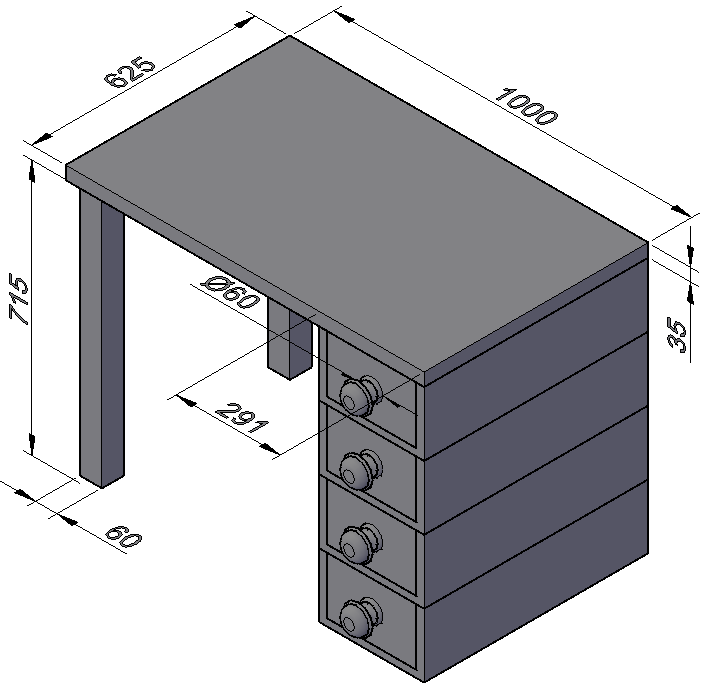


Рисунок 4.23 – Результат построения 3D-модели письменного стола c квадратными основаниями ножек, ручками-кнопками и средними параметрами

На рисунке 4.24 приведен результат построения трехмерной модели письменного стола с круглыми основаниями ножек, ручками-рейлинг ящиков для канцелярии и максимальными параметрами:

* + длина столешницы L1 – 1200 мм;
  + ширина столешницы B – 750 мм;
  + высота столешницы H1 – 40 мм;
  + диаметр основания ножек D – 70 мм;
  + высота ножек H2 – 740 мм;
  + расстояние между крепежными ножками ручки-рейлинг – 90 мм;
  + количество ящиков для канцелярии N – 5 шт;
  + длина ящиков для канцелярии L2 – 400 мм.

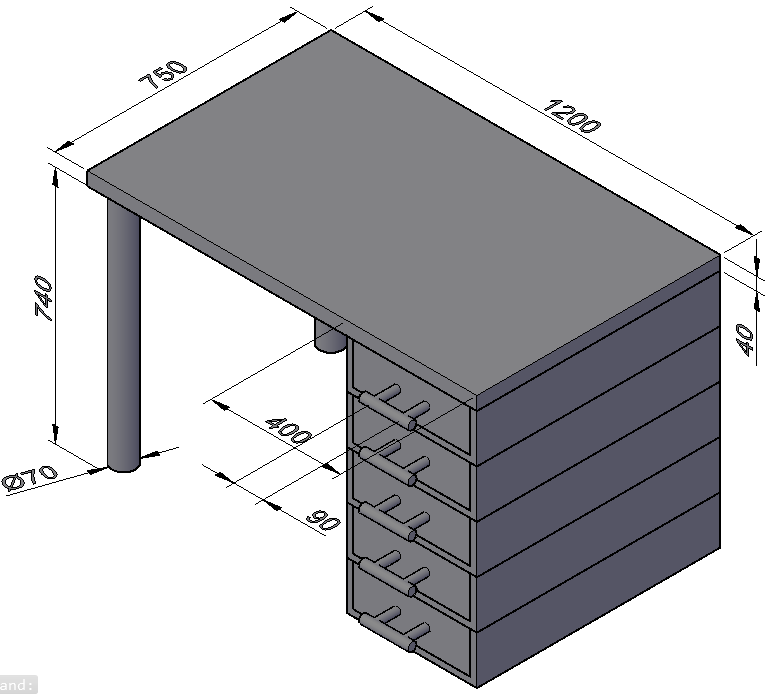


Рисунок 4.24 – Результат построения 3D-модели письменного стола c круглыми основаниями ножек, ручками-рейлинг и максимальными параметрами

На рисунке 4.25 приведен результат построения трехмерной модели письменного стола с круглыми основаниями ножек, ручками-скобами ящиков для канцелярии и максимальными параметрами:

* + длина столешницы L1 – 1200 мм;
  + ширина столешницы B –750 мм;
  + высота столешницы H1 – 40 мм;
  + диаметр основания ножек D – 70 мм;
  + высота ножек H2 – 740 мм;
  + расстояние между крепежными элементами ручки-скобы –   
    100 мм;
  + количество ящиков для канцелярии N – 5 шт;
  + длина ящиков для канцелярии L2 – 400 мм.

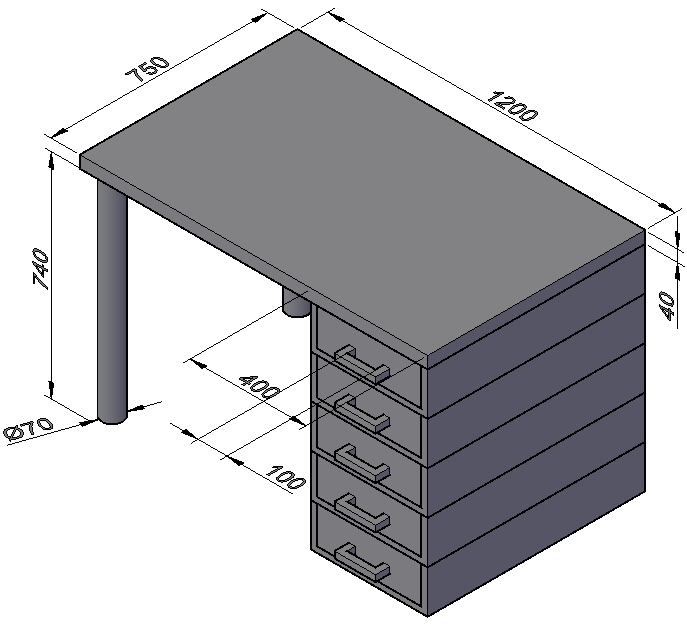


Рисунок 4.25 – Результат построения 3D-модели письменного стола c круглыми основаниями ножек, ручками-скобами и максимальными параметрами

На рисунке 5.26 приведен результат построения трехмерной модели письменного стола с круглыми основаниями ножек, ручками-кнопками ящиков для канцелярии и максимальными параметрами:

* + длина столешницы L1 – 1200 мм;
  + ширина столешницы B – 750 мм;
  + высота столешницы H1 – 40 мм;
  + диаметр основания ножек D – 70 мм;
  + высота ножек H2 – 740 мм;
  + диаметр основания ручки-кнопки – 70 мм;
  + количество ящиков для канцелярии N – 5 шт;
  + длина ящиков для канцелярии L2 – 400 мм.

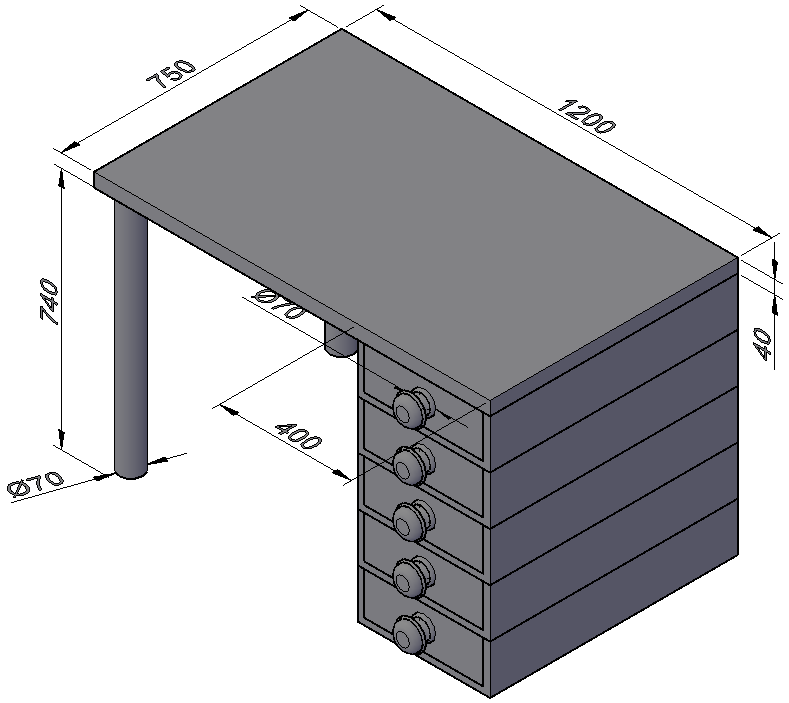


Рисунок 4.26 – Результат построения 3D-модели письменного стола c круглыми основаниями ножек, ручками-кнопками и максимальными параметрами

На рисунке 4.27 приведен результат построения трехмерной модели письменного стола с квадратными основаниями ножек, ручками-рейлинг ящиков для канцелярии и максимальными параметрами:

* + длина столешницы L1 – 1200 мм;
  + ширина столешницы B – 750 мм;
  + высота столешницы H1 – 40 мм;
  + длина основания ножек A – 70 мм;
  + высота ножек H2 – 740 мм;
  + расстояние между крепежными ножками ручки-рейлинг – 90 мм;
  + количество ящиков для канцелярии N – 5 шт;
  + длина ящиков для канцелярии L2 – 400 мм.

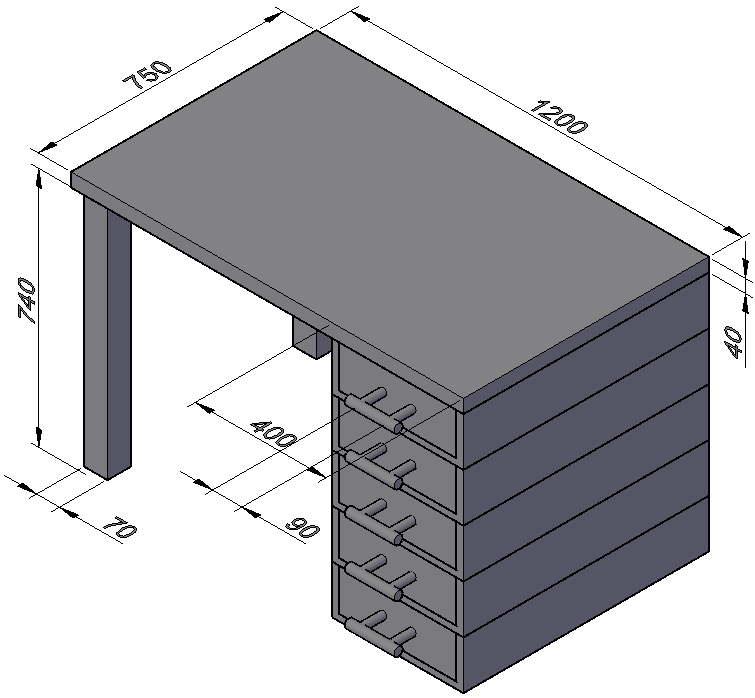


Рисунок 4.27 – Результат построения 3D-модели письменного стола c квадратными основаниями ножек, ручками-рейлинг и максимальными параметрами

На рисунке 4.28 приведен результат построения трехмерной модели письменного стола с квадратными основаниями ножек, ручками-скобами ящиков для канцелярии и максимальными параметрами:

* + длина столешницы L1 – 1200 мм;
  + ширина столешницы B – 750 мм;
  + высота столешницы H1 – 40 мм;
  + длина основания ножек A – 70 мм;
  + высота ножек H2 – 740 мм;
  + расстояние между крепежными элементами ручки-скобы –   
    100 мм;
  + количество ящиков для канцелярии N – 5 шт;
  + длина ящиков для канцелярии L2 – 400 мм.

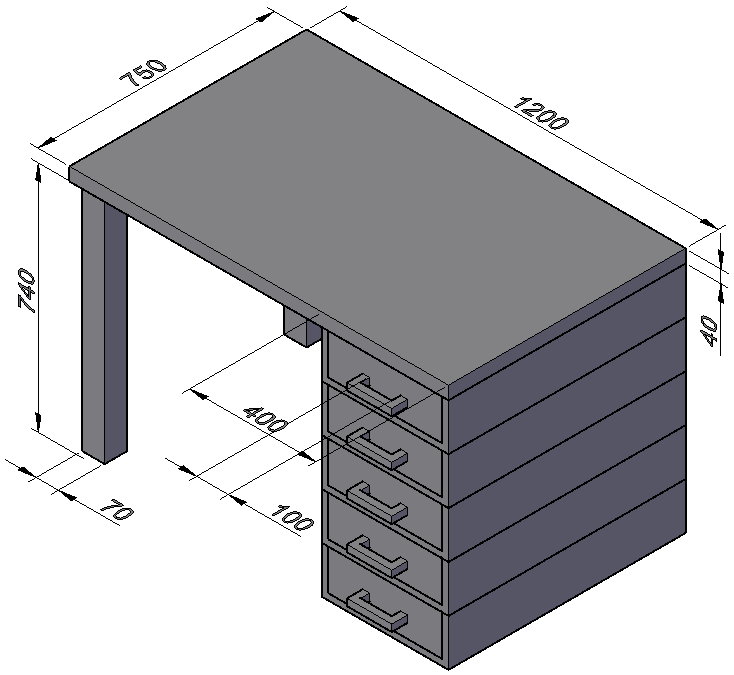


Рисунок 4.28 – Результат построения 3D-модели письменного стола c квадратными основаниями ножек, ручками-скобами и максимальными параметрами

На рисунке 4.29 приведен результат построения трехмерной модели письменного стола с квадратными основаниями ножек, ручками-кнопками ящиков для канцелярии и максимальными параметрами:

* + длина столешницы L1 – 1200 мм;
  + ширина столешницы B – 750 мм;
  + высота столешницы H1 – 40 мм;
  + длина основания ножек A – 70 мм;
  + высота ножек H2 – 740 мм;
  + диаметр основания ручки-кнопки – 70 мм;
  + количество ящиков для канцелярии N – 5 шт;
  + длина ящиков для канцелярии L2 – 400 мм.

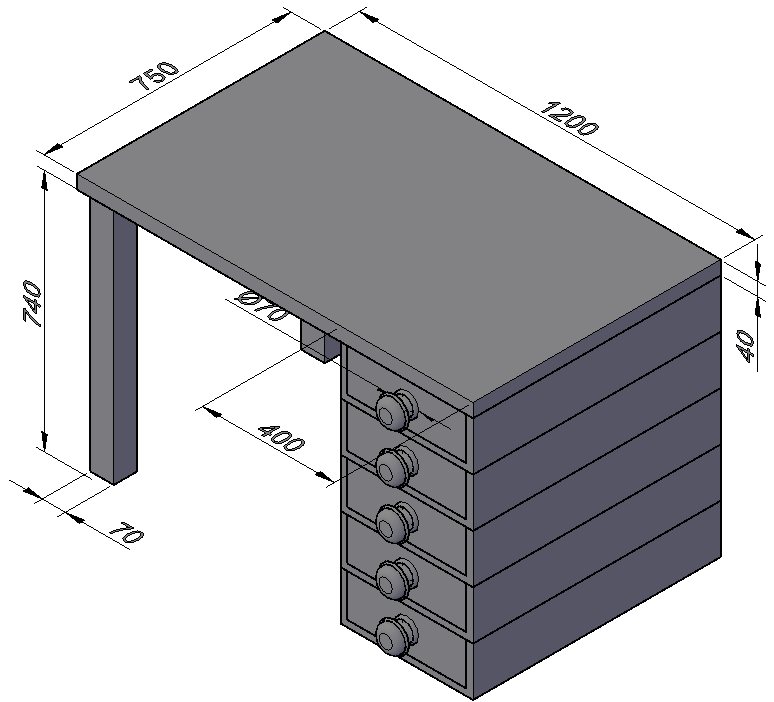


Рисунок 4.29 – Результат построения 3D-модели письменного стола c квадратными основаниями ножек, ручками-кнопками и максимальными параметрами

## Нагрузочное тестирование

Нагрузочное тестирование – это определение качества работы продукта под определенной нагрузкой на заданном интервале времени. Основная цель нагрузочного тестирования состоит в определении максимальной нагрузки, которую может выдержать система, а также в выявлении «узких мест» приложения, влияющих на производительность [17].

Нагрузочное тестирование плагина «Письменный стол» проведено на ПК со следующей конфигурацией:

* процессор AMD Ryzen 5 2500U (2 ГГц);
* оперативная память объемом 8 ГБ (доступно 6,9 ГБ);
* видеокарта AMD Radeon Vega 8;
* операционная система Windows 10 Home x64.

В общей сложности проведено три нагрузочных тестирования плагина: со средними, минимальными и максимальными параметрами.

Первое нагрузочное тестирование заключалось в построении трехмерной модели письменного стола с минимальными параметрами:

* + длина столешницы L1 – 800 мм;
  + ширина столешницы B – 400 мм;
  + высота столешницы H1 – 30 мм;
  + диаметр основания ножек D – 50 мм;
  + высота ножек H2 – 690 мм;
  + расстояние между крепежными элементами ручки-скобы – 80 мм;
  + количество ящиков для канцелярии N – 3 шт;
  + длина ящиков для канцелярии L2 – 250 мм.

Данное тестирование заняло около 55 минут, по итогам его проведения построено 47842 3D-модели письменного стола. Результаты тестирования приведены на рисунках 4.30 – 4.31.

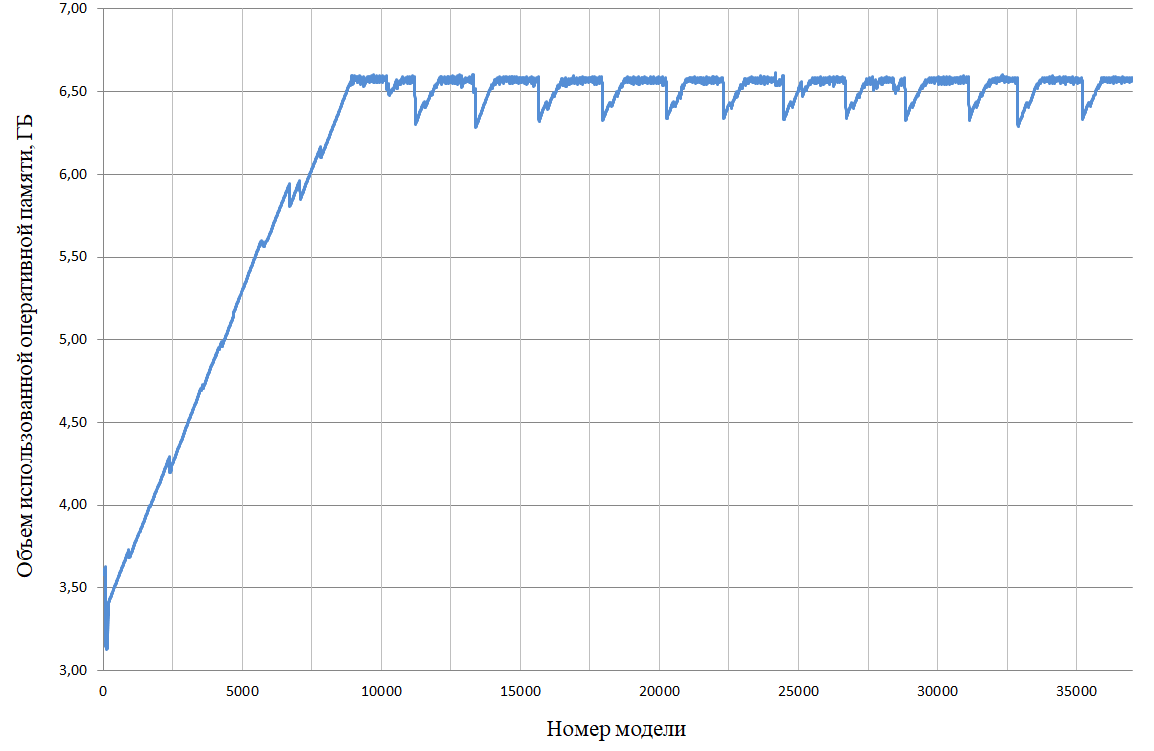


Рисунок 4.30 – График зависимости количества потребляемой оперативной памяти от количества 3D-моделей с минимальными параметрами

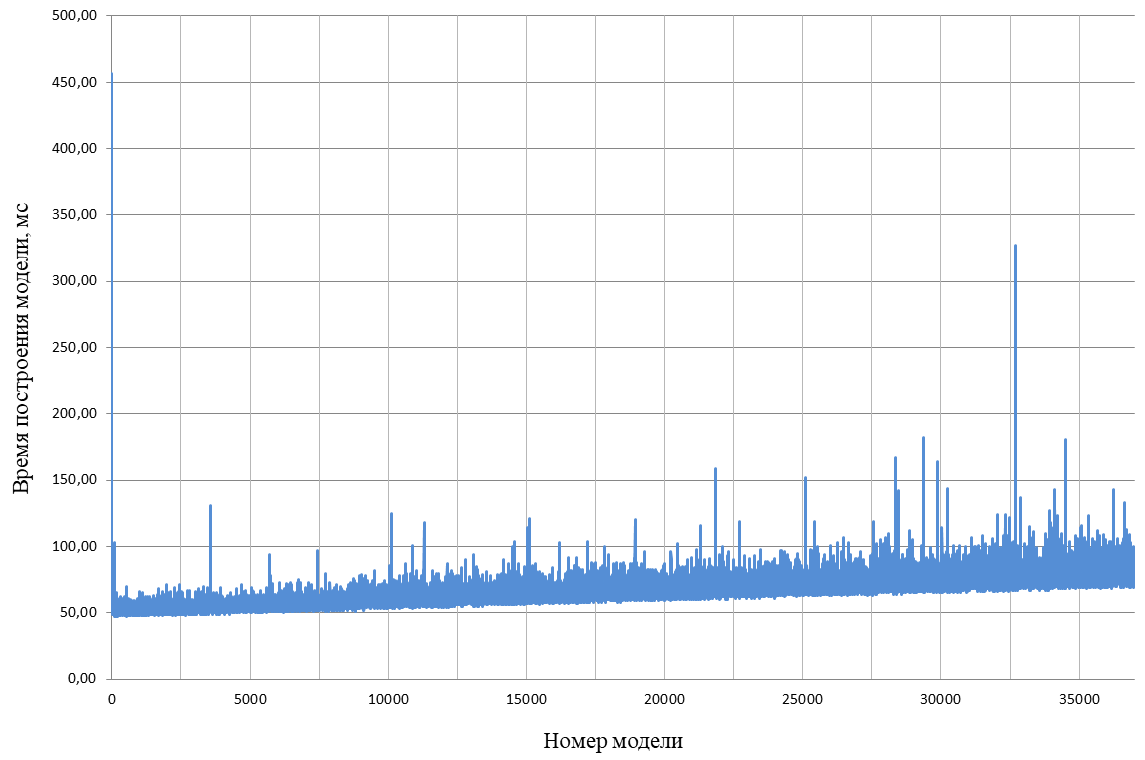


Рисунок 4.31 – График зависимости времени построения одной 3D-модели с минимальными параметрами от количества 3D-моделей

Второе нагрузочное тестирование заключалось в построении трехмерной модели письменного стола со средними параметрами:

* + длина столешницы L1 – 1000 мм;
  + ширина столешницы B – 625 мм;
  + высота столешницы H1 – 35 мм;
  + диаметр основания ножек D – 60 мм;
  + высота ножек H2 – 715 мм;
  + расстояние между крепежными элементами ручки-скобы – 90 мм;
  + количество ящиков для канцелярии N – 4 шт;
  + длина ящиков для канцелярии L2 – 291 мм.

Данное тестирование заняло около 45 минут, по итогам его проведения построено 37013 3D-моделей письменного стола. Результаты тестирования приведены на рисунках 4.32 – 4.33.

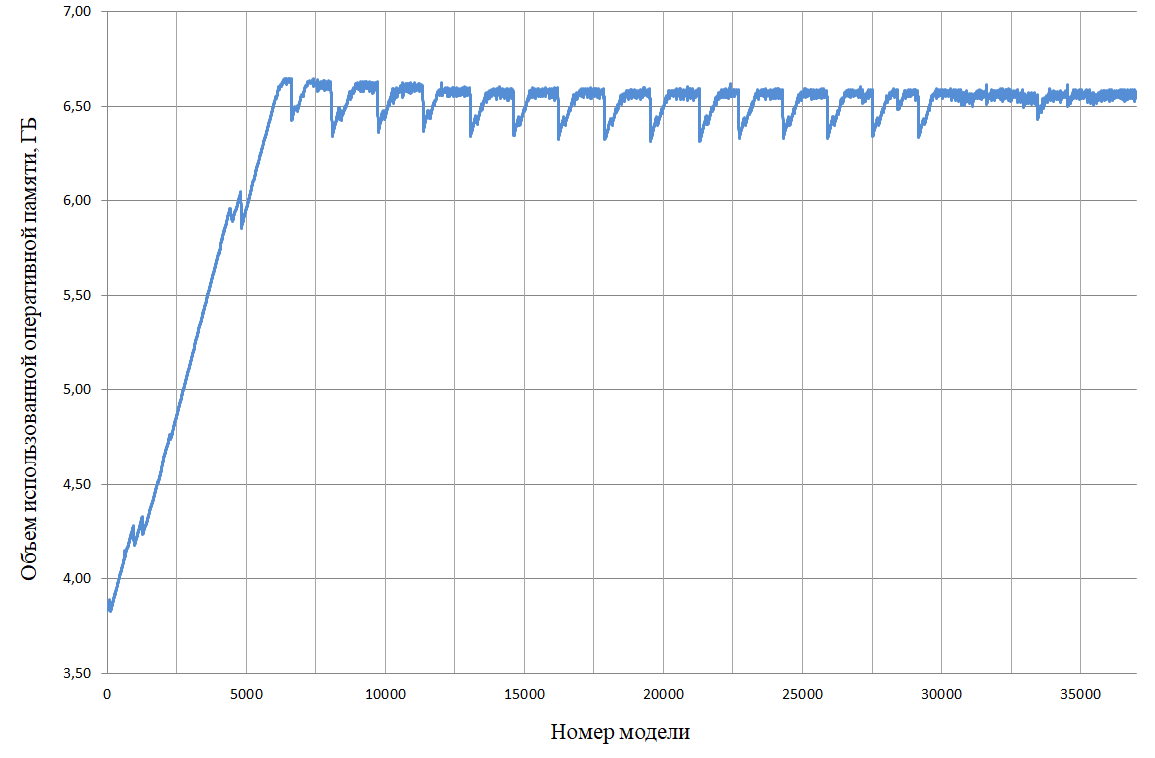


Рисунок 4.32 – График зависимости количества потребляемой оперативной памяти от количества 3D-моделей со средними параметрами

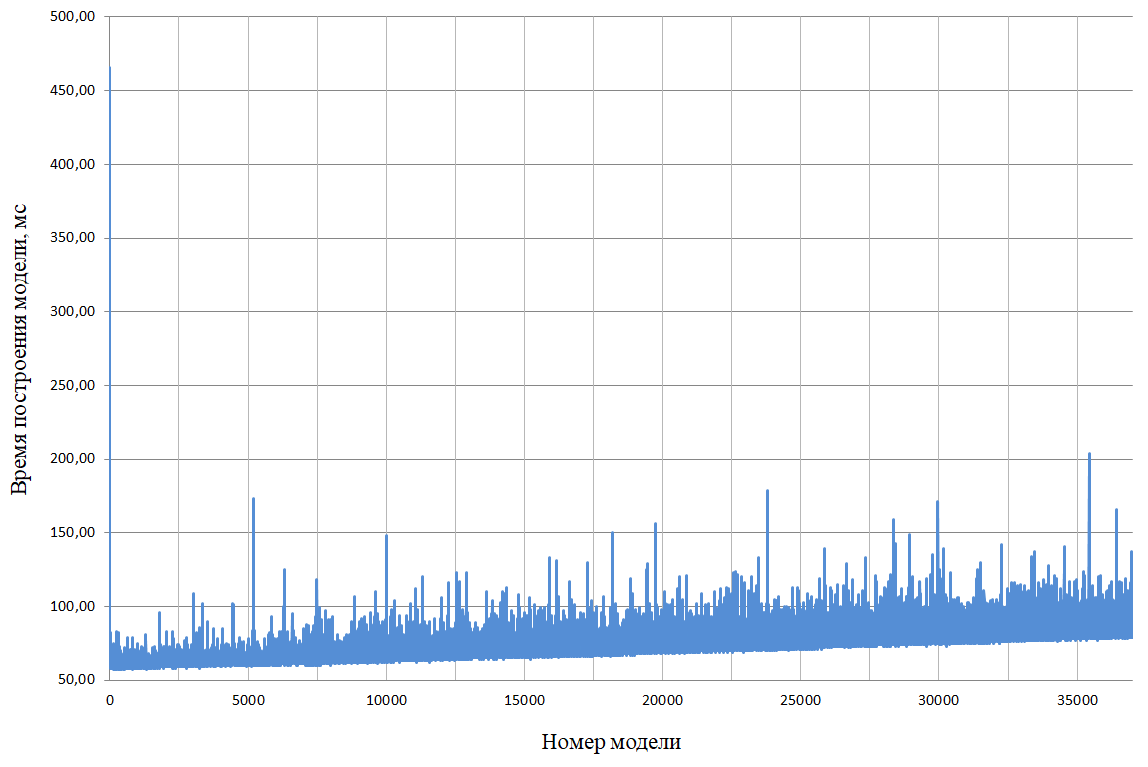


Рисунок 4.33 – График зависимости времени построения одной 3D-модели со средними параметрами от количества 3D-моделей

Третье нагрузочное тестирование заключалось в построении трехмерной модели письменного стола с максимальными параметрами:

* + длина столешницы L1 – 1200 мм;
  + ширина столешницы B –750 мм;
  + высота столешницы H1 – 40 мм;
  + диаметр основания ножек D – 70 мм;
  + высота ножек H2 – 740 мм;
  + расстояние между крепежными элементами ручки-скобы –   
    100 мм;
  + количество ящиков для канцелярии N – 5 шт;
  + длина ящиков для канцелярии L2 – 400 мм.

Данное тестирование заняло около 46 минут, по итогам его проведения построено 30345 3D-моделей письменного стола. Результаты тестирования приведены на рисунках 4.34 – 4.35.

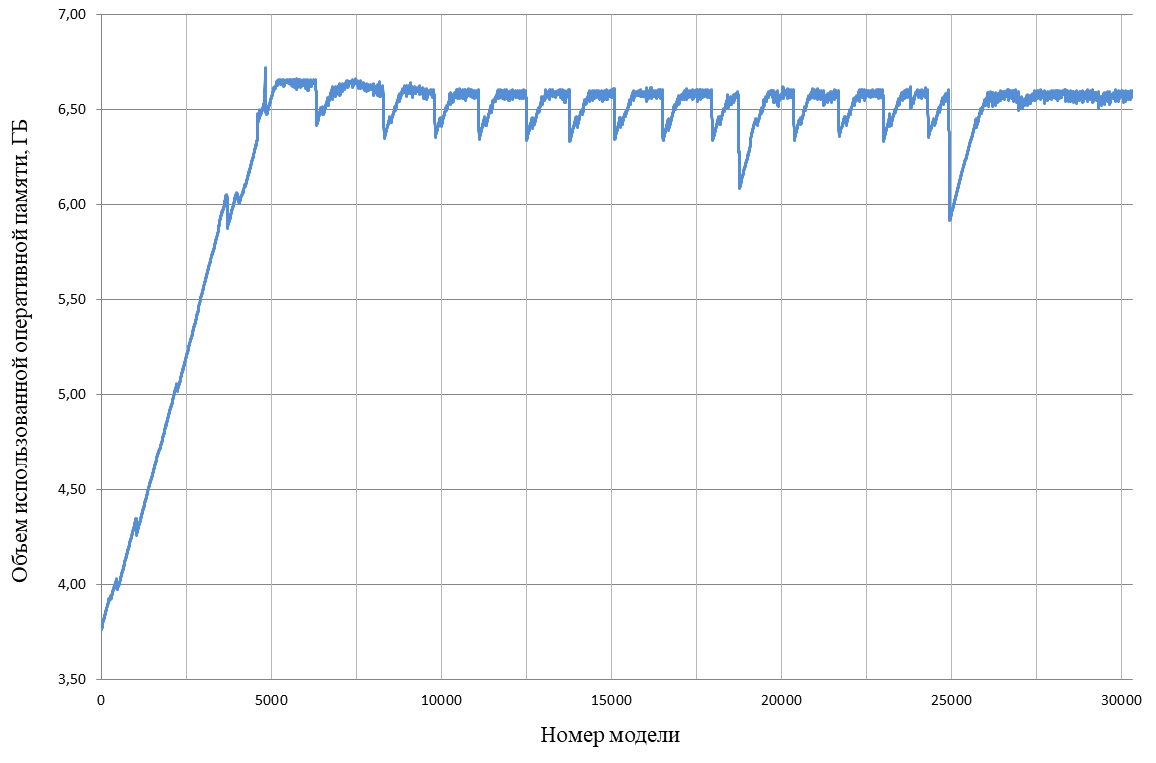


Рисунок 4.34 – График зависимости количества потребляемой оперативной памяти от количества 3D-моделей с максимальными параметрами

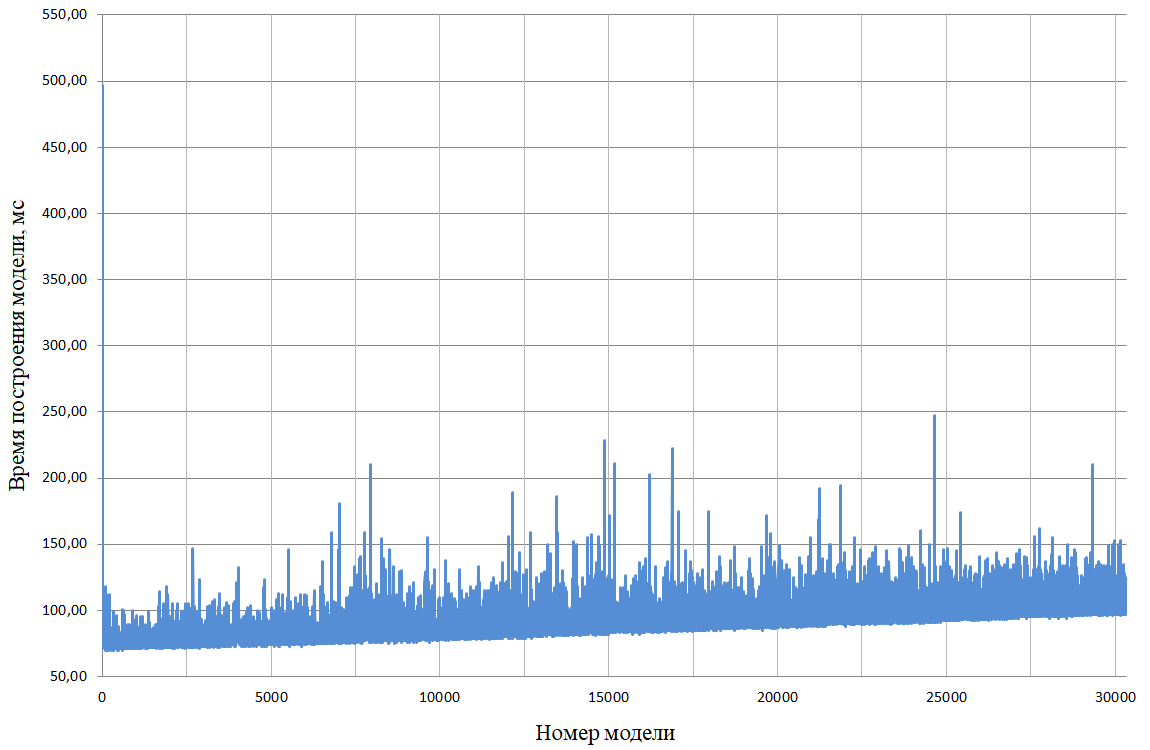


Рисунок 4.35 – График зависимости времени построения одной модели с максимальными параметрами от количества 3D-моделей

Исходя из результатов всех трех проведенных тестирований, можно сделать ряд выводов.

Во-первых, объем оперативной памяти, затрачиваемый плагином на построение трехмерных моделей письменного стола, линейно увеличивается до достижения предела объема оперативной памяти. По окончании свободного места оперативная память частично очищается, после чего данный процесс повторяется.

Во-вторых, во всех трех случаях нагрузочного тестирования наблюдается достижение предела потребления физически доступной оперативной памяти (рисунки 4.30, 4.32, 4.34), однако этот предел достигается тем быстрее, чем выше сложность 3D-модели. Так, объем потребляемой оперативной памяти впервые за соответствующее тестирование составил 6,6 ГБ при построении 9731-й 3D-модели с минимальными параметрами (рисунок 4.30), 6175-й 3D-модели со средними параметрами (рисунок 4.32) и 4809-й 3D-модели с максимальными параметрами (рисунок 4.34).

В-третьих, как можно заключить по данным графиков на рисунках 4.30, 4.32 и 4.34, при построении 3D-моделей письменного стола с минимальными параметрами используемая оперативная память очищается гораздо реже, чем при построении 3D-моделей со средними или максимальными параметрами. Кроме того, при построении 3D-моделей с минимальными параметрами периодически освобождается объем памяти, меньший по сравнению с тем объемом, что требуется очищать при построении 3D-моделей со средними или максимальными параметрами.

В-четвертых, основываясь на данных графиков, изображенных на рисунках 4.31, 4.33 и 4.35, можно заключить, что время построения 3D-моделей со средними параметрами постепенно увеличивается с каждым следующим построением. При этом время построения 3D-моделей с минимальными или максимальными параметрами практически не изменяется на протяжении процесса построения.

# Заключение

В ходе выполнения лабораторных работ изучены основные этапы проектирования программного продукта, исследована предметная область объекта построения, произведено знакомство с программным интерфейсом AutoCAD .NET API, а также рассмотрены некоторые аналоги плагина «Письменный стол».

На начальном этапе разработки составлено техническое задание, далее разработан проект системы, в котором приведены UML-диаграммы классов и макет пользовательского интерфейса будущего плагина. На заключительном этапе разработки написана пояснительная записка к готовому продукту. Также проведено функциональное, модульное и нагрузочное тестирование разработанного плагина.

Результатом выполнения лабораторных работ является плагин для САПР AutoCAD 2021, который выполняет построение трехмерной модели письменного стола с различными изменяемыми параметрами.

# Список использованных источников

Норенков И.П. Основы автоматизированного проектирования. М: МГТУ, 2002. 336 с.

Мазеин П.Г., Шаламов А.В. Сквозное автоматизированное проектирование в CAD/CAM системах: учеб. пособие. Челябинск: ЮУрГУ, 2002. 83 с.

API [Электронный ресурс]: Википедия – свободная энциклопедия. URL: <https://ru.wikipedia.org/wiki/API> (дата обращения: 22.09.2022).

Плагин [Электронный ресурс]: Википедия – свободная энциклопедия. URL: <https://ru.wikipedia.org/wiki/Плагин> (дата обращения 22.09.2022).

AutoCAD [Электронный ресурс]: Википедия – свободная энциклопедия. URL: <https://ru.wikipedia.org/wiki/AutoCAD> (дата обращения: 22.09.2022).

AutoCAD Software [Электронный ресурс]: официальный сайт компании Autodesk. URL: <https://www.autodesk.com/products/autocad/overview> (дата обращения: 23.09.2022).

AutoCAD Platform Technologies [Электронный ресурс]: официальный сайт компании Autodesk. URL: https://www.autodesk.com/developer-network/platform-technologies/autocad (дата обращения: 25.09.2022).

AutoCAD 2022 Developer and ObjectARX Help [Электронный ресурс]: официальная справка Autodesk AutoCAD 2022 Help. URL: https://help.autodesk.com/view/OARX/2022/RUS/?guid=GUID-C3F3C736-40CF-44A0-9210-55F6A939B6F2 (дата обращения: 25.09.2022).

Создание плагинов для AutoCAD с помощью .NET API (часть 1 – первые шаги) [Электронный ресурс]: веб-сайт в формате системы тематических коллективных блогов Хабр. URL: https://habr.com/ru/post/235723/ (дата обращения 27.09.2022).

Введение в .NET. AutoCAD .NET API – презентация онлайн [Электронный ресурс]: просмотр PPT-файлов онлайн. URL: https://ppt-online.org/476500 (дата обращения: 28.09.2022).

Auto-Furniture [Электронный ресурс]: сайт компании FurnitureSoft. URL: <https://www.furnituresoft-software.com/auto> (дата обращения: 29.09.2022).

SymbolDesigner Beta [Электронный ресурс]: сайт компании CAD Division. URL: <http://www.caddivision.com/en/software/symboldesigner> (дата обращения: 29.09.2022).

ГОСТ 20400-2013. Продукция мебельного производства. Термины и определения. М.: Стандартинформ, 2014. 41 с.

Фаулер М. UML. Основы. 3-е издание. СПб.: Символ-Плюс, 2019. 192 с.

Инверсия управления и Внедрение зависимостей (IoC & DI) [электронный ресурс]: CODE BLOG (shwanoff.ru). URL: https://shwanoff.ru/ioc-and-di/ (дата обращения: 26.10.2022).

Калентьев А.А., Гарайс Д.В., Горяинов А.Е. Новые технологии в программировании: учеб. пособие. Томск: Эль Контент, 2014. 176 с.

Поговорим о нагрузочном тестировании [Электронный ресурс]: веб-сайт в формате системы тематических коллективных блогов Хабр. URL: <https://habr.com/ru/company/veeam/blog/578942/> (дата обращения: 27.10.2022).