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 «Карданная вилка»   
для «КОМПАС-3D v.20»**

Проект системы по лабораторному проекту по дисциплине   
«Основы разработки САПР»  
«Построение подвесных полок в системе КОМПАС-3D v.20»

Выполнил:  
Студент гр. 588-3

\_\_\_\_\_\_\_\_\_\_\_\_/ Галичанина Е.Е.

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_20\_\_г.

Руководитель:  
к.т.н, доцент каф. КСУП

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

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_20\_\_г.

Томск 2022

Оглавление

1 Описание САПР 3

1.1 Описание программы 3

1.2 Описание API 4

1.3 Обзор аналогов 4

2 Описание предмета проектирования 4

3 Проект программы 4

3.1 Диаграмма классов 4

3.2 Макеты пользовательского интерфейса 5

Список использованных источников 8

**1 Описание САПР**

**1.1 Описание программы**

Проектирование новых видов и образцов машин, оборудования, устройств, аппаратов, приборов и других изделий представляет сложный и длительный процесс, включающий в себя разработку исходных данных, чертежей, технической документации, необходимых для изготовления опытных образцов и последующего производства, и эксплуатации объектов проектирования.

***Проектирование*** — это комплекс работ с целью получения описаний нового или модернизируемого технического объекта, достаточных для реализации или изготовления объекта в заданных условиях. В процессе проектирования возникает необходимость создания описания, необходимого для построения еще не существующего объекта. Получаемые при проектировании описания бывают окончательными или промежуточными. Окончательные описания представляют собой комплект конструкторско-технологической документации в виде чертежей, спецификаций, программ для ЭВМ и автоматизированных комплексов и т.д. [1]

Основной целью автоматизации является повышение качества исполнения процесса. Автоматизированный процесс обладает более стабильными характеристиками, чем процесс, выполняемый в ручном режиме. Во многих случаях автоматизация процессов позволяет повысить производительность, сократить время выполнения процесса, снизить стоимость, увеличить точность и стабильность выполняемых операций.

***Система автоматизированного проектирования (САПР)***— это организационно-техническая система, состоящая из комплекса средств автоматизации проектирования, взаимодействующего с подразделениями проектной организации и выполняющая автоматизированное проектирование.

Однако, с приходом на отечественный рынок иностранных систем, широкое распространение получили аббревиатуры *CAD* (Computer Aided Design), которую можно перевести, как проектирование с применением компьютера, и *CAD-system*, которую можно перевести, как система для проектирования с помощью компьютера. [2]

В настоящее время в среде специалистов по САПР многие термины утратили свой первоначальный смысл, а термин *САПР* теперь обозначает программу для автоматизированного проектирования.

Для реализации плагина будет использоваться программа   
«КОМПАС-3D» версии 20.

***Компас-3D*** – это система трехмерного моделирования деталей и сборок, используемая для проектирования изделий в машиностроении и строительстве — от изделий народного потребления до авиа-, судостроения и продукции военного назначения. [3]

Система «КОМПАС-3D» отличается проектированием изделий любой сложности, простотой освоения, бесплатной технической поддержкой, автоматизацией отраслевых задач и многим другим.

**1.2 Описание API**

Аббревиатура ***API*** расшифровывается как «Application Programming Interface» (интерфейс программирования приложений, программный интерфейс приложения). [4]

Сегодня встречаются задачи, решение которых не реализованы в CAD-системах. Чаще всего это очень узкоспециализированные задачи, которые встречаются на каком-то конкретном предприятии или подотрасли. Для решения подобных задач и их автоматизации используется API.

***API КОМПАС-3D*** — это ориентированные на прикладного программиста инструментальные средства разработки приложений (библиотек конструктивов, прикладных САПР) на базе системы КОМПАС. API КОМПАС-3D включает в свой состав API 5 и API 7.

Главным интерфейсом API системы КОМПАС является ***KompasObject.***[5] Методы этого интерфейса, реализуют наиболее общие функции работы с документами системы, системными настройками, файлами, а также дают возможность получить указатели на другие интерфейсы (интерфейсы динамического массива, работы с математическими функциями, библиотек моделей или фрагментов и различных структур параметров определенного типа).

Ниже в таблице 1.1 представлены основные свойства и методы интерфейса KompasObject.

Таблица 1.1 – Методы и свойства интерфейса KompasObject

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| Document3D() |  | ksDocument | Метод для получения указателя на интерфейс трехмерного графического документа (детали или сборки) |

Продолжение таблицы 1.1

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| GetParamStruct(short structType) | structType – тип интерфейса параметров | StructType2D | Метод для получения указателя на интерфейс графического документа (чертежа или фрагмента) |
| Visible |  | bool | Свойство видимости приложения |
| Quit() |  |  | Метод для закрытия активного окна приложения КОМПАС |

В таблице 1.2 представлены методы интерфейса ksEntity, необходимые для разработки плагина.

Таблица 1.2 – Методы интерфейса ksEntity

|  |  |  |
| --- | --- | --- |
| Название | Тип возвращаемых данных | Описание |
| Create() | bool | Создать объект в модели |
| GetDefinition() | IUnkown | Получить указатель на интерфейс параметров объектов и элементов |
| Update() | bool | Изменить свойства объекта (используя ранее установленные свойства) |

В таблице 1.3 представлены свойства и методы интерфейса ksDocument2D, необходимые для разработки плагина

Таблица 1.3 – Методы интерфейса ksDocument2D

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| ksRectangle(  ksRectangleParam param, int style) | param – параметры прямоугольника.  style – стиль линии. | int | Получить указатель на прямоугольник на двумерной плоскости либо 0 в случае ошибки |
| ksCircle(  double xc, double yc, double rad, int style) | xc, yc – координаты центра окружности.  rad – радиус окружности.  style – стиль линии. | int | Получить указатель на окружность на двумерной плоскости либо 0 в случае ошибки |

В таблице 1.4 представлены свойства и методы интерфейса ksDocument3D, необходимые для разработки плагина.

Таблица 1.4 – Методы интерфейса ksDocument3D

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание | |
| Create (bool invisible, bool \_typeDoc) | invisible – признак режима редактирования документа (true – невидимый режим, false – | bool | Создать документ-модель (деталь или сборку) | |
|  | видимый режим),  typeDoc – тип документа (true – деталь, false – сборка). |  | |  |
| GetPart(int type) | type – тип компонента из перечисления Типы компонентов. | ksPart | | Получить указатель на интерфейс компонента в соответствии с заданным типом |

В таблице 1.5 представлены методы интерфейса ksPart, необходимые для разработки плагина.

Таблица 1.5 – Свойства и методы интерфейса ksPart.

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Входные параметры | Тип возвращаемых данных | Описание |
| EntityCollection(short objType) | objType – тип объектов, содержащихся в массиве. | ksEnintyCollection | Формирует массив объектов и возвращает указатель на его интерфейс |
| GetDefaultEntity(short objType) | objType – тип объекта | ksEntity | Получить указатель на интерфейс объекта, создаваемого системой по умолчанию |
| GetPart(int type) | Type – тип компонента | ksPart | Получить указатель на интерфейс компонента в соответствии с заданным типом |
| NewEntity(short objType) | objType – тип объекта | ksEntity | Создать новый интерфейс объекта и получить указатель на него |

В таблице 1.6 представлены типы объектов документа-модели, необходимые для разработки плагина.

Таблица 1.6 – Некоторые типы объектов документа-модели

|  |  |  |
| --- | --- | --- |
| Идентификатор объекта | Название объекта | Интерфейс параметров |
| o3d\_unknown | Неизвестный (включает все объекты) |  |
| o3d\_planeXOZ | Плоскость XOZ | ksPlaneParam |
| o3d\_planeYOZ | Плоскость YOZ | ksPlaneParam |
| o3d\_planeXOY | Плоскость XOY | ksPlaneParam |
| o3d\_sketch | Эскиз | ksSketchDefinition |
| o3d\_face | Грань | ksFaceDefinition |
| o3d\_baseExtrusion | Базовая операция выдавливания | ksBaseExtrusionDefinition |

**1.3 Обзор аналогов**

**MechaniCS**

CSoft Development MechaniCS представляет собой приложение к AutoCAD или Autodesk Inventor, предназначенное для оформления чертежей в соответствии с ЕСКД, проектирования систем гидропневмоэлементов, зубчатых зацеплений, валов, инженерного анализа, расчета размерных цепей и создания пользовательских библиотек.

MechaniCS обеспечивает специалиста всем необходимым для проектирования машиностроительных объектов: более чем двумя тысячами стандартов (включая ГОСТ, ОСТ, DIN и ISO) и унифицированными компонентами, возможностью создавать собственные интеллектуальные объекты, выполнять инженерные расчеты с отображением результатов на модели, оформлять проекции чертежей по ЕСКД и многим другим.

Все детали общей конструкторско-технологической базы обладают интеллектом и являются объектно-зависимыми. При изменении параметров одной детали все связанные с ней объектно-зависимые детали изменятся автоматически, причем в соответствии с их параметрами в базе. Такая технология — мощный инструмент многовариантного проектирования, залог повышения качества выпускаемых проектов. Важно, что этот подход одинаково доступен пользователям AutoCAD и Autodesk Inventor.

MechaniCS дает конструктору возможность учитывать не только геометрические параметры стандартных элементов, но и их механические свойства. На объекты в сборочных чертежах (при использовании AutoCAD) можно накладывать геометрические и параметрические зависимости, использовать предустановленные зависимости при их размещении на чертеже.

Продукт прост и понятен в освоении. Приступить к его использованию можно буквально сразу после его установки. [3]

Интерфейс пакета показан на рисунке 1.1.

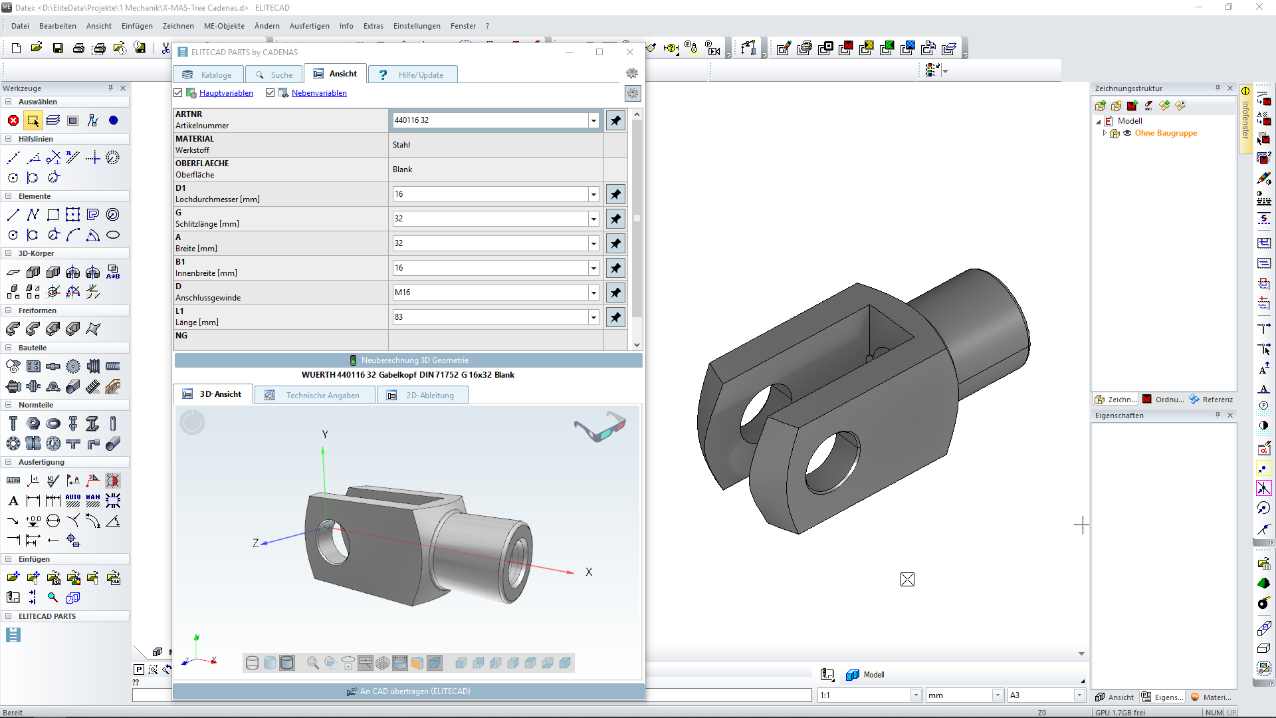


Рисунок 1.1 — Интерфейс пакета MechaniCS

**2 Описание предмета проектирования**

Предметом проектирования является карданная вилка.

Вилка, скользящая кардана образует шлицевый наконечник который вместе со шлицевой втулкой образует подвижное шлицевое соединение, компенсирующее изменение длины карданного вала в результате перемещения заднего моста. Данная деталь изготовляется из стали 45 которая имеет следующие химический состав и механические свойства.[7]

Под желание заказчика в карданной вилке могут изменять параметры, приведенные ниже:

1. A — высота детали от 20 до 30 мм;
2. Б — высота основания от 7 до 17 мм;
3. В — ширина основания детали от 12 до 22 мм;
4. Г — длина основания детали от 12 до 22 мм;
5. Д — диаметр отверстия в основании от 1 до 5 мм;
6. Е — диаметр на стенке детали от 5 до 11 мм;
7. Ж — ширина стенки детали от 2 до 6 мм.

Плагин имеет следующие зависимости:

1. Ширина(В) и длина(Г) основания детали должны быть ровны;

2. Разница между шириной основания детали(В) и диаметров на стенке детали(Е) должна быть больше или ровна 4 мм.

На рисунке 2.1 показан общий вид карданной вилки:

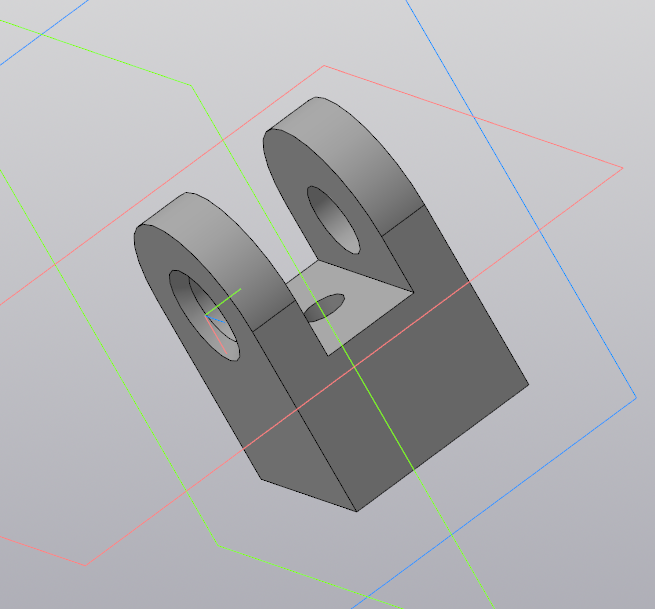


Рисунок 2.1 – Общий вид карданной вилки

На рисунках 2.2 и 2.3 представлен чертеж карданной вилки с указанными параметрами:

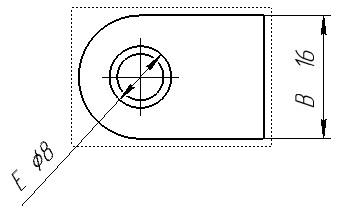


Рисунок 2.2 – Вид сбоку

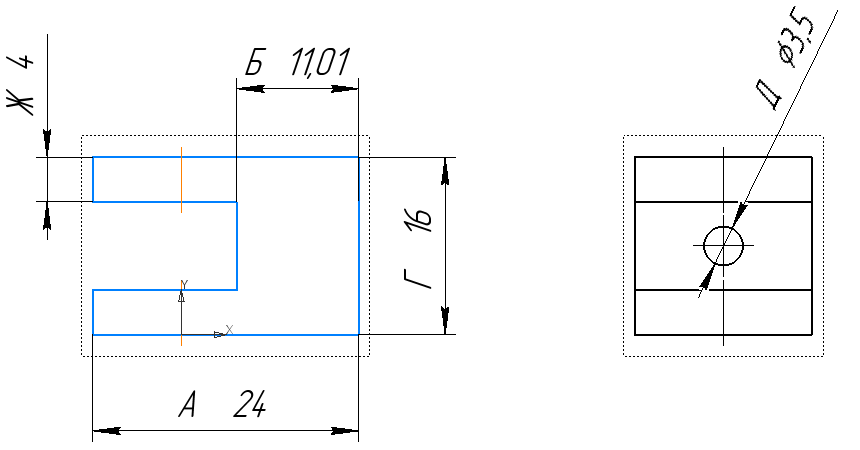


Рисунок 2.3 – Вид спереди и снизу

**3 Проект программы**

**3.1 Диаграмма классов**

Диаграмма классов определяет типы классов системы и различного рода статические связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции классов и ограничения, которые накладываются на связи между классами.[8] Целью создания диаграммы классов является графическое представление статической структуры элементов системы. Диаграмма классов плагина представлена на рисунке 3.1.

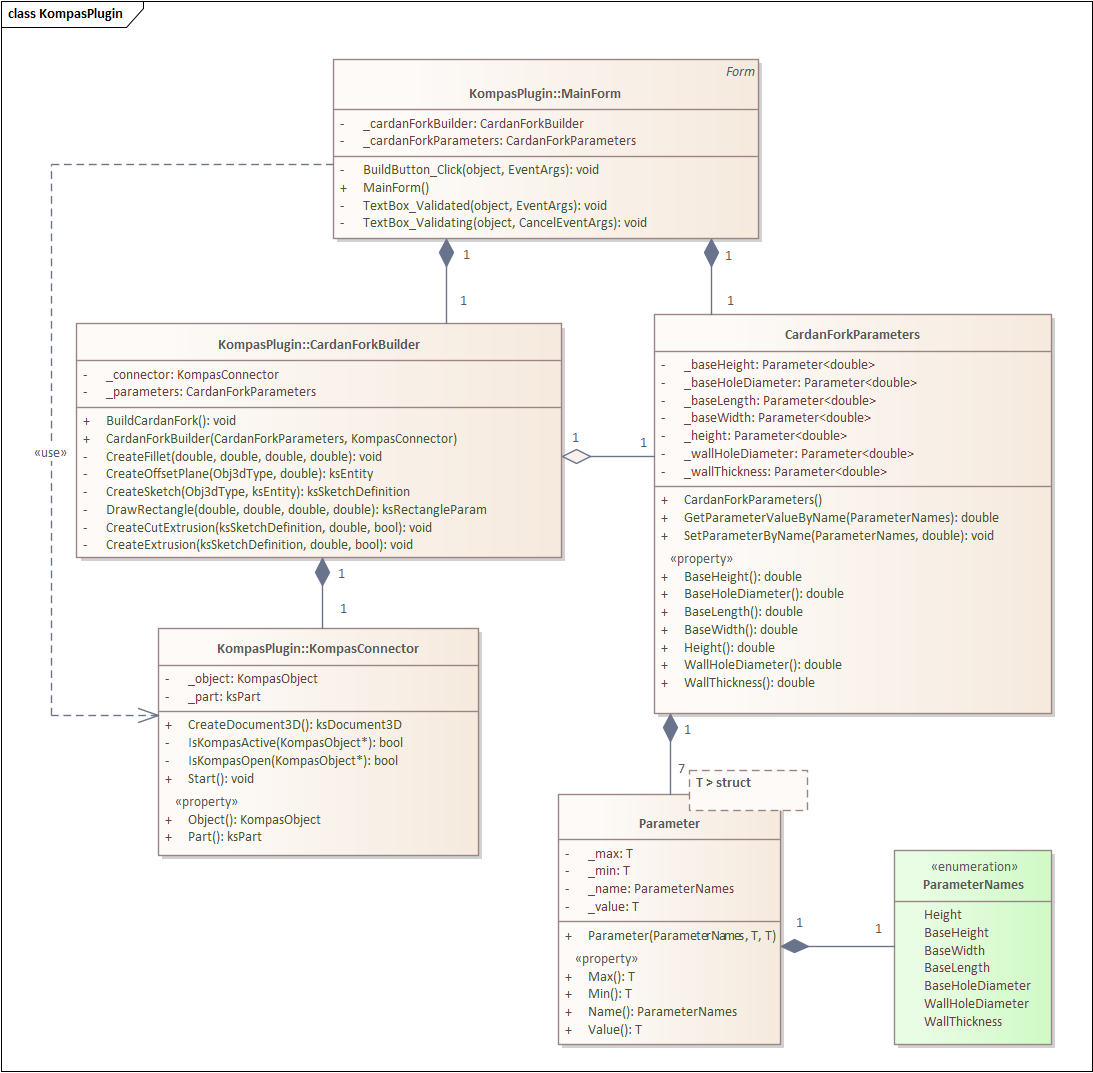


Рисунок 3.1 — Диаграмма классов плагина «Карданная вилка»

* MainForm является главным элементом управления для обработки действий в графическом интерфейсе;
* CardanForkBuilder – выполняет построение детали, в данном классе находятся поля и методы, которые будут использовать API КОМПАС-3D
* CardanForkParameters – содержит в себе параметры карданной вилки;
* Parameter – данный шаблонный класс будет создаваться для каждого параметра, нужен для хранения в себе основной информации о параметре;
* ParameterNames – содержит названия параметров для избегания ошибок и удобной валидации;
* KompasConnector – класс связи с КОМПАС – 3D.

В таблицах 3.1 – 3.6 представлены описания полей, свойств и методов разработанных классов.

Таблица 3.1 – Класс MainForm

|  |  |  |
| --- | --- | --- |
| Название | Тип  возвращаемых данных | Описание |
| \_cardanForkBuilder | CardanFork  Builder | Закрытое поле. Объект класса построителя |
| \_cardanForkParameters | CardanFork  Parameters | Закрытое поле. Объект класса с параметрами |
| BuildButton(object sender, EventArgs e) |  | Обработчик нажатия кнопки “Построить” |
| TextBox\_Validated(object sender, EventArgs e) |  | Устанавливает стиль для проверенного значения |
| TextBox\_Validating(object sender, EventArgs e) |  | Общий метод валидации текстбокса |
| MainForm() | MainForm | Конструктор главной формы с необходимой инициализацей |

Таблица 3.2 – Класс Parameter

|  |  |  |
| --- | --- | --- |
| Название | Тип  возвращаемых данных | Описание |
| \_min | Заданное при инициализации | Закрытое поле. Минимальное значение параметра. |
| \_max | Заданное при инициализации | Закрытое поле. Максимальное значение параметра. |
| \_value | Заданное при инициализации | Закрытое поле. Значение параметра. |
| \_name | ParameterNames | Закрытое поле. Имя параметра. |
| Name() | ParameterNames | Возвращает и устанавливает имя параметра. |
| Value() | Заданное при инициализации | Возвращает и устанавливает значение параметра, проверяется входит ли в диапазон между минимальным и максимальным. |
| Min() | Заданное при инициализации | Возвращает и устанавливает минимальное значение. |
| Max() | Заданное при инициализации | Возвращает и устанавливает максимальное значение, проверяется чтобы было больше минимального. |
| Parameter(Parameter  Names,  double,double,double) | Parameter | Конструктор параметра |

Таблица 3.3 – Класс CardanForkParameters

|  |  |  |
| --- | --- | --- |
| Название | Тип  возвращаемых данных | Описание |
| \_baseHeight | Parameter<double> | Закрытое поле. Параметр высоты основания |
| \_baseHoleDiameter | Parameter<double> | Закрытое поле. Параметр диаметра отверстия в основании |
| \_baseLength | Parameter<double> | Закрытое поле. Параметр длины основания |
| \_baseWidth | Parameter<double> | Закрытое поле. Параметр ширины основания |
| \_height | Parameter<double> | Закрытое поле. Параметр высоты детали |
| \_wallHoleDiameter | Parameter<double> | Закрытое поле. Параметр диаметра отверстия на стенке детали |
| \_wallThickness | Parameter<double> | Закрытое поле. Параметр ширины стенки детали |
| SetValueByName(  ParameterNames,  doublу) |  | Устанавливает значение параметра по имени |
| GetValueByName (ParameterNames) | double | Возвращает значение параметра. |
| CardanForkParameters() |  | Конструктор параметров модели |
| BaseHeight | double | Задаёт/возвращает высоту основания |
| BaseHoleDiameter | double | Задаёт/возвращает диаметр отверстия в основании |
| BaseLength | double | Задаёт/возвращает длину основания |
| BaseWidth | double | Задаёт/возвращает ширину основания |
| Height | double | Задаёт/возвращает высоту детали |
| WallHoleDiameter | double | Задаёт/возвращает диаметр отверстия в стенке |
| WallThickness | double | Задаёт/возвращает толщину стенок |

Таблица 3.4 – Класс ModelBuilder

|  |  |  |
| --- | --- | --- |
| Название | Тип  возвращаемых данных | Описание |
| \_parameters | CardanForkParameters | Закрытое поле. Хранит объект параметров вилки |
| \_connector | KompasConnector | Закрытое поле. Объект класса конектора для связи с КОМПАС-3D |
| BuildCardanFork  () |  | Создаёт модель карданной вилки |
| CardanForkBuilder() | double | Конструктор ModelBuilder |
| DrawRectangle(double,  double, double,  double) | ksRectangleParam | Метод рисующий прямоугольник по заданным парамтерам |
| CreateExtrusion  (ksSketchDefinition,  depth, bool) |  | Метод осуществляющий выдавливание |
| CreateСгеExtrusion  (ksSketchDefinition,  depth, bool) |  | Метод осуществляющий вырезание |
| CreateSketch(Obj3dType,  double) | ksSketch  Definition | Создаёт новый эскиз в плоскости |
| CreateFillet(double,  double, double) |  | Метод создающий фаску на ребре |

Таблица 3.5 – Класс KompasWrapper

|  |  |  |
| --- | --- | --- |
| Название | Тип  возвращаемых данных | Описание |
| \_kompasObject | KompasObject | Закрытое поле. Объект API КОМПАС-3D |
| \_part | ksPart | Закрытое поле. Компонент сборки |
| Object() | KompasObject | Свойство. Возвращает объект интерфейса API |
| Part() | ksPart | Свойство. Возвращает компонент сборки |
| CreareDocument3D() | ksDocument3d | Запускает окно создания 3D-модели |
| IsKompasActive  (KompasObject\*) | bool | Метод смотрит, есть ли КОМПАС-3D в таблице запущенных объектов |
| IsKompasOpen  (KompasObject\*) | bool | Метод запускающий КОМПАС-3D |
| Start() |  | Метод начала работы с КОМПАС-3D, вызывает сначала IsKompasActive() и если тот возвращает false вызывает IsKompasOpen |

**3.2 Макеты пользовательского интерфейса**

Пользовательский интерфейс представляет собой форму для ввода параметров. На форме присутствует чертёж с параметрами для демонстрации параметров карданной вилки и поля для ввода. Пользователь вводит значения самостоятельно, опираясь на подсказки, отображенные около полей. При нажатии на кнопку «Построить» проводится проверка зависимых параметров и, если условия соблюдены, строится 3D-модель карданной вилки. На рисунке 3.2 представлен макет пользовательского интерфейса.

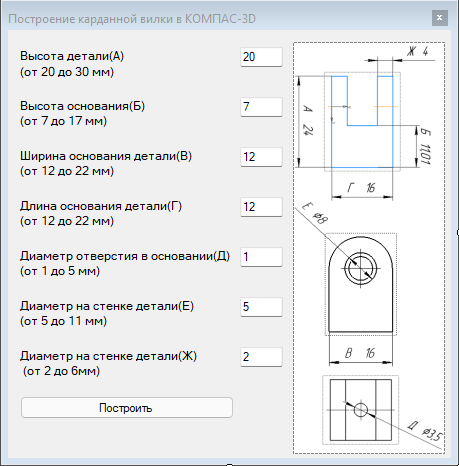


Рисунок 3.2 — Макет пользовательского интерфейса

Проверка правильности ввода значений проводится по ходу заполнения полей. Если поле заполнено неправильно, то есть пользователь ввел значение, превышающее границы, то оно подсвечивается красным цветом, сигнализирующем об ошибке.

Если же введены некорректные значения, и пользователь решил построить модель, несмотря на них, кнопка построения будет неактивна, пока не будут введены корректные значения (рисунок 3.3).

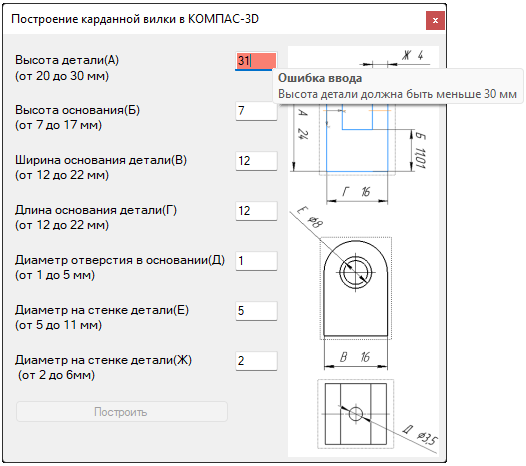


Рисунок 3.3 — Пример обработки ошибок при построении модели

**Список использованных источников**

1. Общие сведения о САПР [Электронный ресурс]. – Режим доступа: http://www.hi-edu.ru/e-books/xbook116/01/part-002.htm, свободный (дата обращения: 16.10.2021).

2. Обзор популярных систем автоматического проектирования (CAD) [Электронный ресурс]. – Режим доступа: https://www.pointcad.ru/novosti/obzor-sistem-avtomatizirovannogo-proektirovaniya, свободный (дата обращения: 16.10.2021).

3. КОМПАС-3D: О программе [Электронный ресурс]. – Режим доступа: https://kompas.ru/kompas-3d/about/, свободный (дата обращения: 18.10.2021).

4. Что такое API? [Электронный ресурс]. – Режим доступа: https://dev.by/news/chto-takoe-api-prostym-yazykom, свободный (дата обращения: 18.10.2021).

5. Работа с API КОМПАС-3D [Электронный ресурс]. – Режим доступа: https://habr.com/ru/company/ascon/blog/328088/, свободный (дата обращения: 21.10.2021).

6. MechaniCS. [Электронный ресурс]. — Режим доступа: https://axoft.ru/vendors/CSoft-Development/CSoft-MechaniCS/ (дата обращения 13.03.2022).

7. Карданная вилка [Электронный ресурс]. – Режим доступа: https://gkgk.ru/publications/articles/articles\_54.html/, свободный (дата обращения: 13.03.2022).

8. Фаулер М. UML. Основы. 3-е издание / М.Фаулер. – 3-е изд., пер. с англ. – СПб.: Символ-Плюс, 2019. – 192 с.