```cpp
#ifndef _SMP_BASE_H
#define _SMP_BASE_H

#include "Helper.h"
#include <PArray.h>

#if _MSC_VER > 1000
#pragma once
#endif
#endif

/// Список фигур основания
enum SurfaceType
{
    circle,     /// Круг
    triangle,   /// Треугольник
    square,     /// Квадрат
    pentagon,   /// Пятиугольник
    hexagon,    /// Шестиугольник
    trigon,     /// Тригон
    rhombus35,  /// Ромб с углом 35 градусов
    rhombus55,  /// Ромб с углом 55 градусов
    rhombus80,  /// Ромб с углом 80 градусов
};

/// Список способов задания основного размера детали
enum SizeType
{
    byOuterRadius,  /// Размер по радиусу описанной окружности
    byInnerRadius,  /// Размер по радиусу вписанной окружности
    bySide,         /// Размер по длине стороны
};

/// <summary>
/// Основной класс СМП
/// </summary>
class SMPBase
{
public:
    /// Интерфейс компонентов IApplicationPtr
    IApplicationPtr application;
    /// Интерфейс компонентов HMODULE
    HMODULE hmodule;
    /// Интерфейс компонентов IKompasDocumentPtr
    IKompasDocumentPtr active_document;
    /// Интерфейс компонентов IKompasDocument3DPtr
    IKompasDocument3DPtr k_document_3d;
    /// Интерфейс компонентов IDocument3D
    IDocument3DPtr doc3D;
    /// Интерфейс компонентов IPart
    IPartPtr part;
    /// Интерфейс компонентов IPart7
    IPart7Ptr part7;
    /// Интерфейс контейнера трехмерных объектов
    IModelContainerPtr model_container;
```

```
56      /// Интерфейс элемента эскиза основания
57      ISketchPtr base_sketch;
58      /// Интерфейс элемента плоскости основания
59      IPlane3DByOffsetPtr base_plane;
60      /// Координаты центра основания
61      struct coordinates {
62          double X = 0.0; /// Координата X
63          double Y = 0.0; /// Координата Y
64          double Z = 0.0; /// Координата Z
65      } coordinates;
66      /// Фигура основания
67      SurfaceType surfaceType = SurfaceType::square;
68      /// Тип размера
69      SizeType sizeType = SizeType::bySide;
70      /// Основной размер
71      double size = 22;
72      /// Высота
73      double height = 7;
74      /// Наличие отверстия
75      bool hasHole = false;
76      /// Радиус отверстия
77      double holeRadius = 0;
78      /// Радиус скругления
79      double roundingRadius = 3.0;
80      /// Угол наклона пластины α
81      double angleAlpha = 10;
82      /// Кнопка "Применить"
83      int button = 0;
84
85      /// Получение указателя приложения
86      void init(IApplicationPtr application, HMODULE hmodule);
87      /// Присвоение модели параметры документа
88      void initDocumentParameters();
89      /// Получение радиуса описанной окружности модели
90      double getOuterRadius();
91      /// Получение радиуса вписанной окружности модели
92      double getInnerRadius();
93      /// Получение длины стороны модели
94      double getSide();
95      /// Получение итоговой высоты модели
96      double getHeight();
97      /// Получение количества сторон правильного многоугольника
98      unsigned short getNGonSideCount();
99      /// Получение величины угла ромба
100     double getRhombusAngle();
101     /// Проверка: основной размер - радиус описанной окружности
102     bool isByOuterRadius();
103     /// Проверка: основной размер - радиус вписанной окружности
104     bool isByInnerRadius();
105     /// Проверка: основной размер - длины стороны
106     bool isBySide();
107     /// Проверка: основание - треугольник
108     bool isTriangle();
```

```
109     /// Проверка: основание - квадрат
110     bool isSquare();
111     /// Проверка: основание - пятиугольник
112     bool isPentagon();
113     /// Проверка: основание - шестиугольник
114     bool isHexagon();
115     /// Проверка: основание - ромб с углом 35 градусов
116     bool isRhombus35();
117     /// Проверка: основание - ромб с углом 55 градусов
118     bool isRhombus55();
119     /// Проверка: основание - ромб с углом 80 градусов
120     bool isRhombus80();
121     /// Проверка: основание - правильный многоугольник
122     bool isNGon();
123     /// Проверка: основание - ромб
124     bool isRhombus();
125     /// Проверка: основание - тригон
126     bool isTrigon();
127     /// Проверка: основание - круг
128     bool isCircle();
129     /// Проверка: наличие и величина радиуса отверстия
130     bool checkHasHole();
131
132 protected:
133     /// Радиус описанной окружности
134     double outerRadius = 0;
135     /// Радиус вписанной окружности
136     double innerRadius = 0;
137     /// Длины стороны
138     double side = 0;
139     /// Смещение итоговой модели по X
140     double x_delta = 0;
141     /// Смещение итоговой модели по Y
142     double y_delta = 0;
143     /// Список координат точек фигуры основания
144     vector<tuple<double, double>> points;
145
146     /// Создание основания
147     void addEmbodiment();
148     /// Создание основания
149     void drawBase();
150     /// Создание основания - круга
151     void drawCircle(IDrawingContainerPtr drawing_container);
152     /// Создание основания - правильный многоугольник
153     void drawNGon(IDrawingContainerPtr drawing_container);
154     /// Создание основания - тригон
155     void drawTrigon(IDrawingContainerPtr drawing_container);
156     /// Создание основания - ромб
157     void drawRhombus(IDrawingContainerPtr drawing_container);
158     /// Создание выдавливания
159     void addExtrusion();
160     /// Создание скругления боковых рёбер
161     void addRounding();
```

```
162     /// Создание отверстия
163     void addHole();
164     /// <summary>
165     /// Получение интерфейса контейнера объектов вида графического
          документа
166     /// </summary>
167     /// <param name="sketch_document">Эскиз</param>
168     IDrawingContainerPtr getDrawingContainer(IFragmentDocumentPtr
          sketch_document);
169     /// <summary>
170     /// Получение координат точки основания
171     /// </summary>
172     /// <param name="index">Индекс точки</param>
173     void getPoint(size_t index, double* x, double* y);
174     /// Получение координат первой точки основания
175     void getPointFirst(double* x, double* y);
176     /// Получение координат последней точки основания
177     void getPointLast(double* x, double* y);
178     /// Добавление координат точки основания
179     void addPoint(double x, double y);
180 };
181
```