



МИНИСТЕРСТВО НАУКИ  
И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное  
образовательное учреждение высшего образования  
«НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»



**НГТУ  
НЭТИ** | **Факультет прикладной  
математики и информатики**

Кафедра прикладной математики

Практическая работа №1  
по дисциплине «Компьютерная графика»

## **ВВЕДЕНИЕ В ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ OpenGL**

|               |                                |
|---------------|--------------------------------|
| Студенты      | ГРОСС АЛЕКСЕЙ<br>ШИШКИН НИКИТА |
| Вариант       | 3                              |
| Группа        | ПМ-92                          |
| Преподаватель | ЗАДОРОВ Ж. А. Г.               |

Новосибирск, 2022

## Цель работы

Ознакомиться с основами использования библиотеки OpenGL и работе с примитивами.

**Вариант:** тип примитивов GL\_LINE\_STRIP.

## Практическая часть

1. Отобразить в окне множество примитивов (вершины которых задаются кликами мыши) в соответствии с вариантом задания.
2. Для завершения текущего (активного) набора (множества) примитивов и начала нового зарезервировать специальную клавишу (пробел или правый клик).
3. Для текущего набора примитивов предоставить возможность изменения цвета и координат его вершин.
4. Предусмотреть возможность удаления последнего примитива и последнего набора примитивов.

### **Дополнительные задания:**

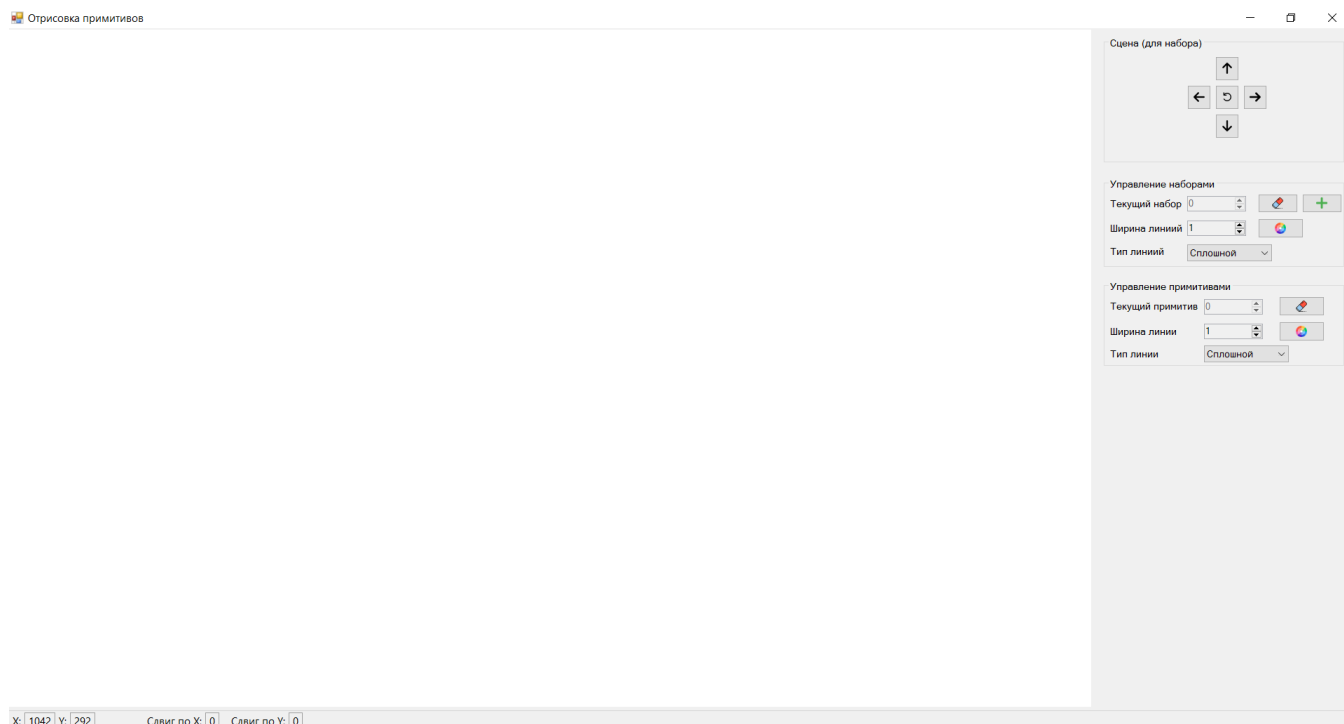
- изменение не только координат и цвета вершин примитивов, но и шаблона закрашивания и т.д.;
- изменение параметров (в том числе и удаление) не только текущего набора, но произвольного;
- изменение параметров произвольного примитива в наборе.

# Руководство пользователя

## Введение

Окно приложения разделено на две части: левая часть – **рабочая область**, правая – **панель управления**. Рабочая область предназначена для отрисовки наборов примитивов.

Общий вид окна:



Панель управления состоит из нескольких элементов:

- мини-панель управления сцены, руководясь которой пользователь может передвигать конкретный набор примитивов;
- мини-панель управления наборами примитивов;
- мини-панель управления примитивами;

В нижней части окна приложения отображаются текущие координаты курсора мыши и текущий сдвиг системы координат для выбранного набора примитивов.

## Работа с мышью в рабочей области приложения


Пользователь в рабочей области левым кликом мыши может задавать вершины, которые, в свою очередь, будут преобразовываться в примитив. Правый клик предназначен для того, чтобы прекратить отрисовку текущего примитива.

## Основные элементы панели управления и команды

### Управление сценой

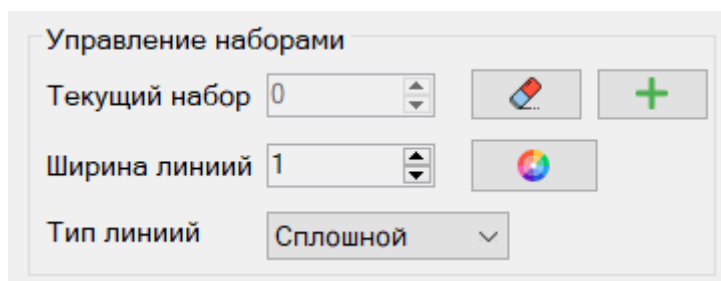
Вид мини-панели:



Данный элемент панели управления представляет собой набор из 5 кнопок, а именно стрелок, при нажатии которых смещается начало системы координат в выбранном направлении и кнопки возврата , при нажатии которой система координат вернется в исходное положение.

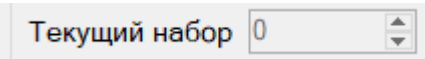
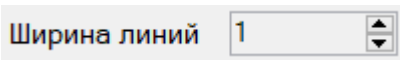
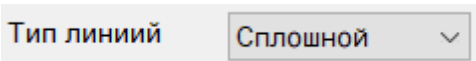
### Управление наборами




Вид мини-панели:



Данный элемент панели управления представляет собой управление наборами в рабочей области приложения.

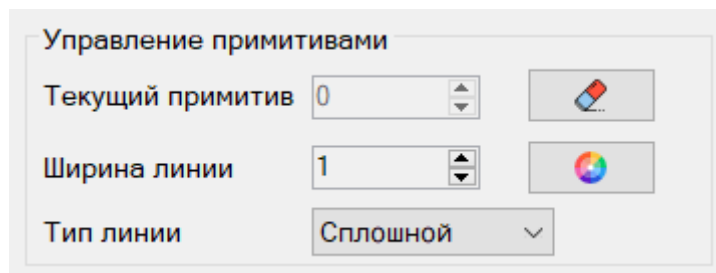
#### Элементы управления:

- числовое поле , с помощью которого можно переключаться между наборами примитивов;
- числовое поле , с помощью которого можно изменять ширину линий набора примитивов (от 1 до 5);
- выпадающий список , с помощью которого можно выбрать тип линий набора примитивов (сплошной, точечный, штриховой, штрихпунктирный);

- кнопка  для удаления текущего набора примитивов;
- кнопка  для добавления нового набора примитивов;
- кнопка  для открытия диалогового окна выбора цвета набора примитивов.

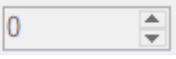
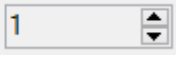
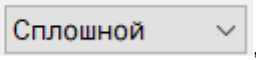


## Управление примитивами

Вид мини-панели:



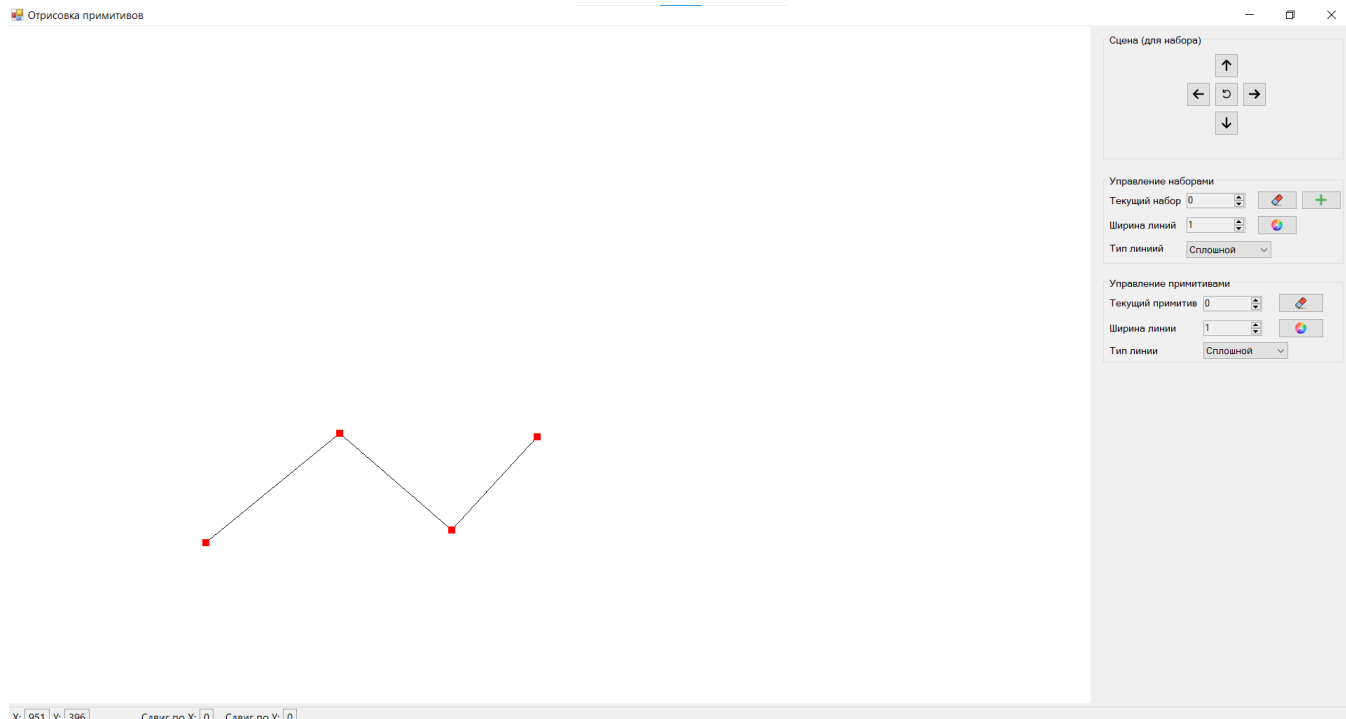
Данный элемент панели управления представляет собой управление примитивами в наборе в рабочей области приложения.

### Элементы управления:

- числовое поле **Текущий примитив** , с помощью которого можно переключаться между примитивами в наборе;
- числовое поле **Ширина линии** , с помощью которого можно изменять ширину линий выбранного примитива (от 1 до 5);
- выпадающий список **Тип линии** , с помощью которого можно выбрать тип линий выбранного примитива (сплошной, точечный, штриховой, штрихпунктирный);
- кнопка  для удаления текущего примитива из набора;
- кнопка  для открытия диалогового окна выбора цвета текущего примитива.

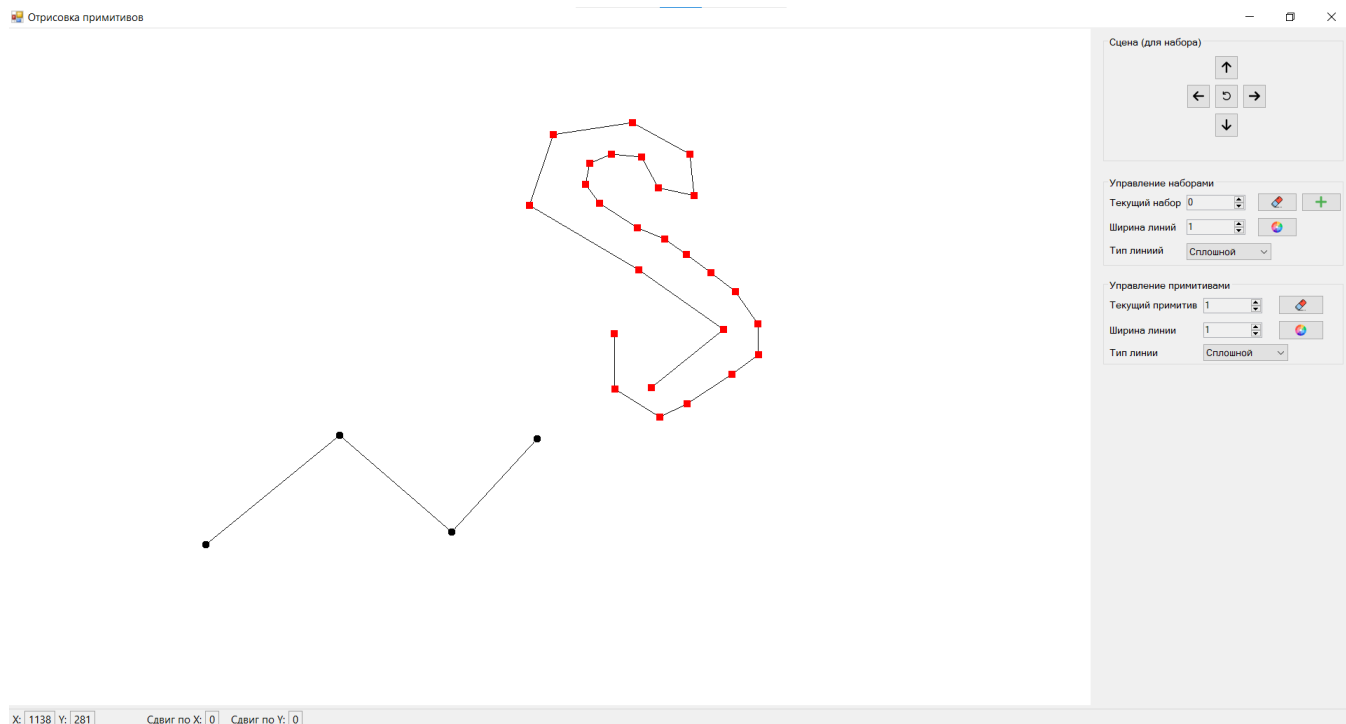
# Тестирование программы

## Отрисовка простого примитива

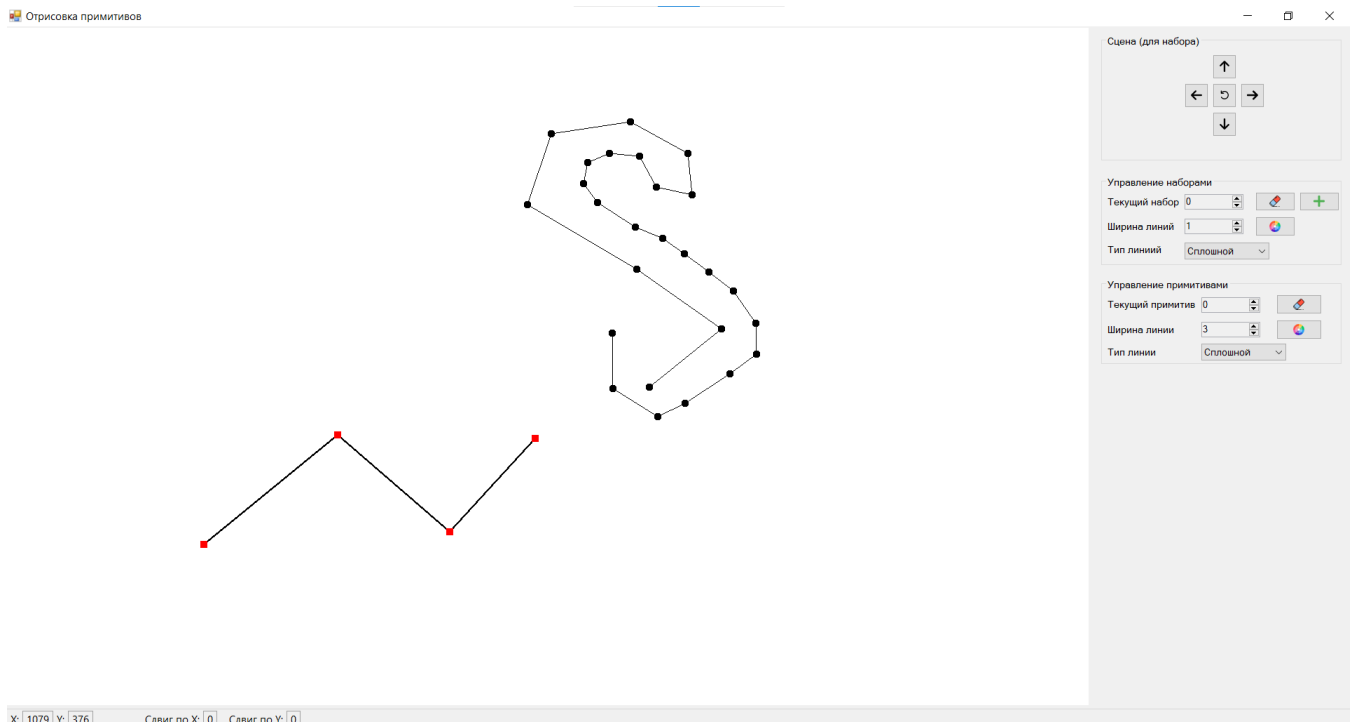


## Добавление нового примитива

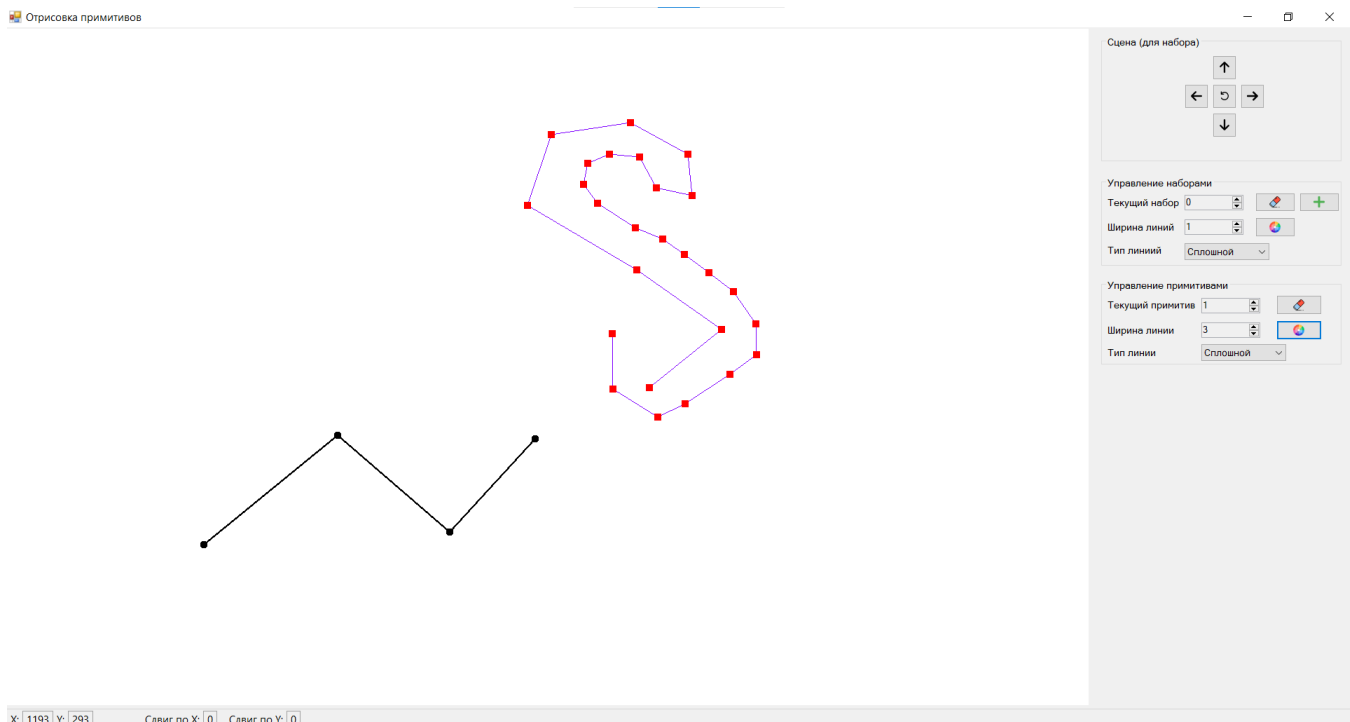
Текущий (новый) примитив выделен красными точками.



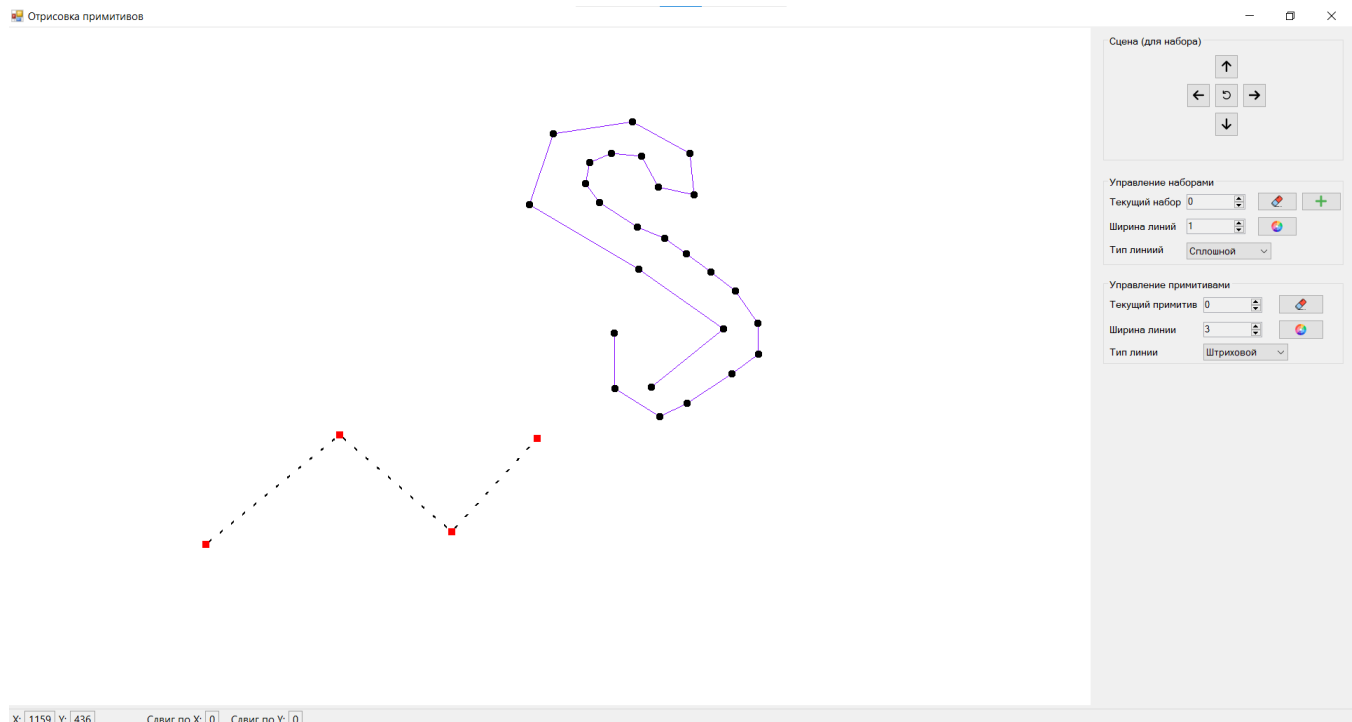
## Изменение ширины линии выбранного примитива



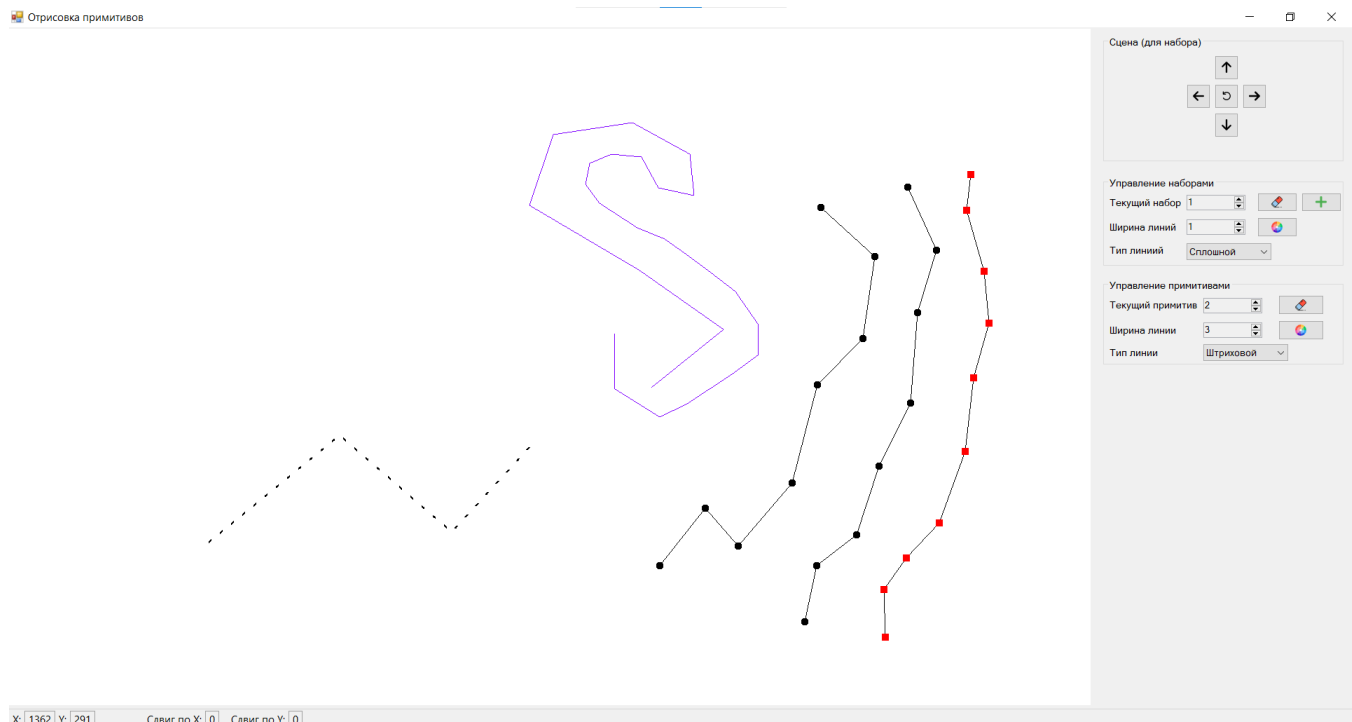
## Изменение цвета выбранного примитива



## Изменение типа линии выбранного примитива

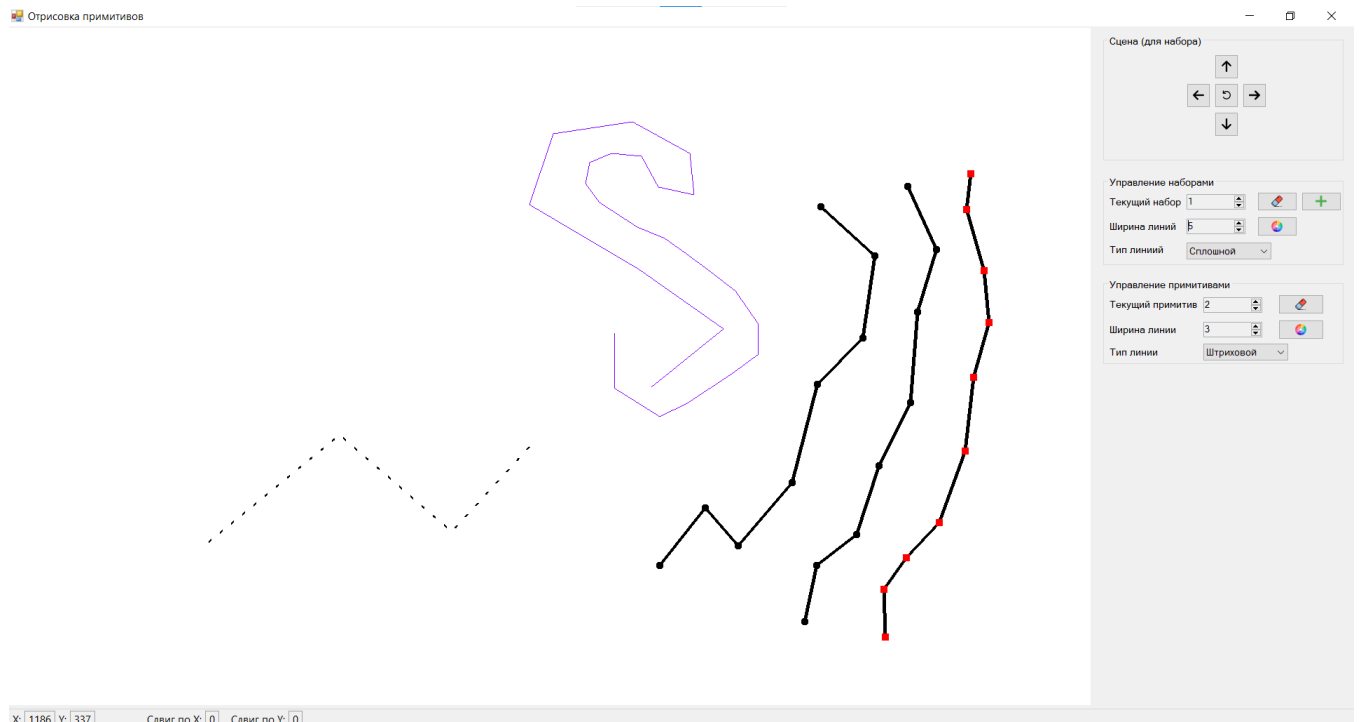


## Создание нового набора примитивов

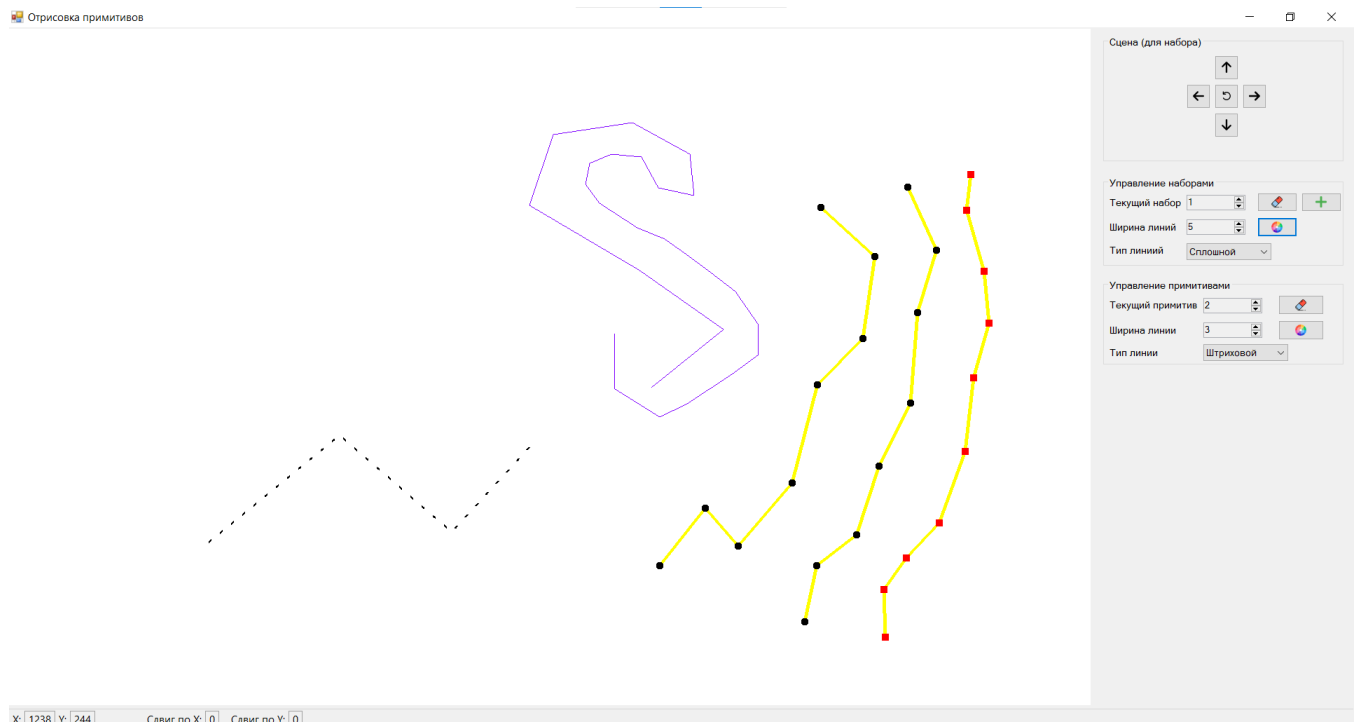




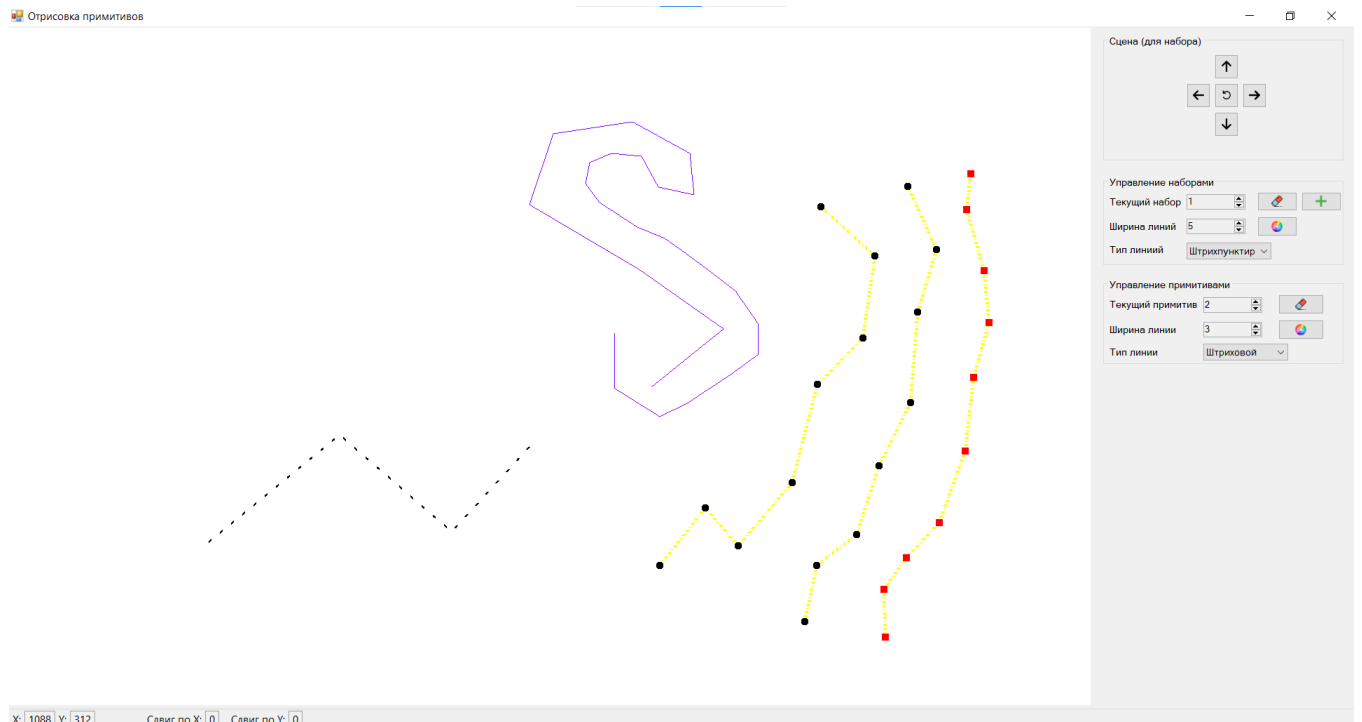
## Изменение ширины линий выбранного набора примитивов



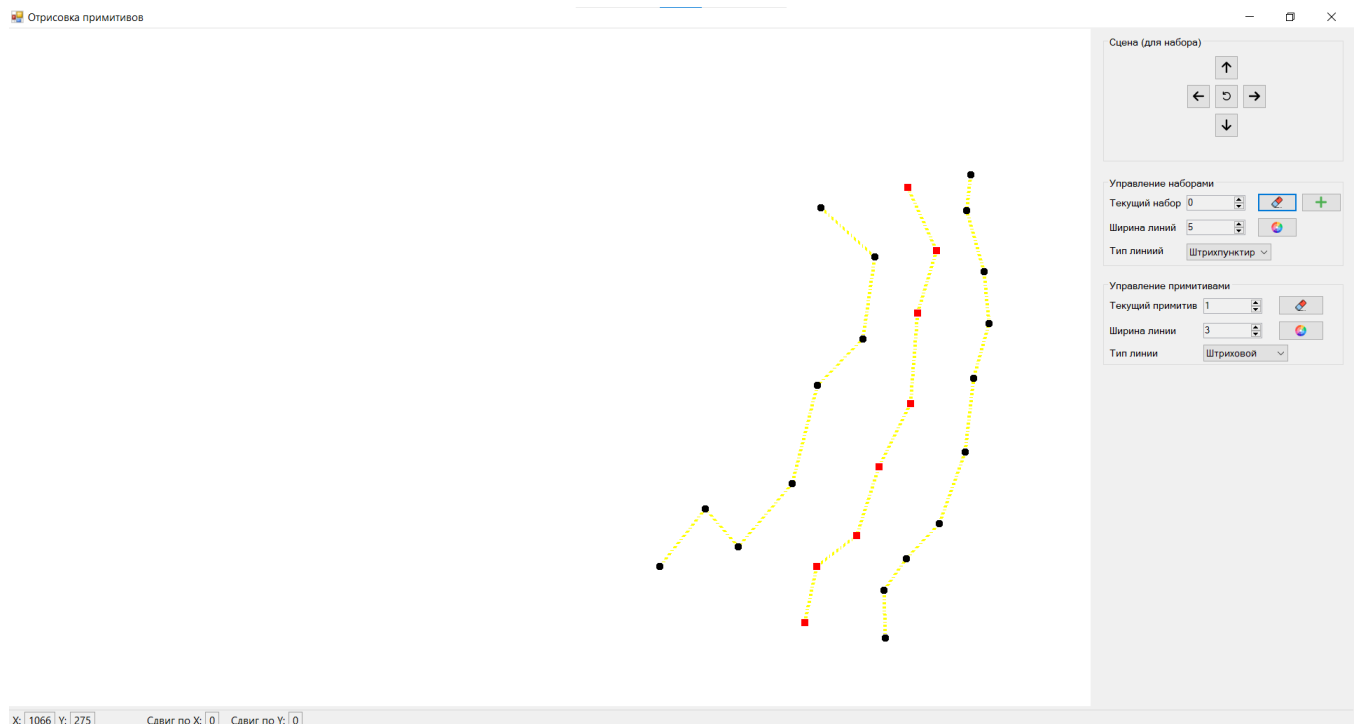
## Изменение цвета выбранного набора примитивов



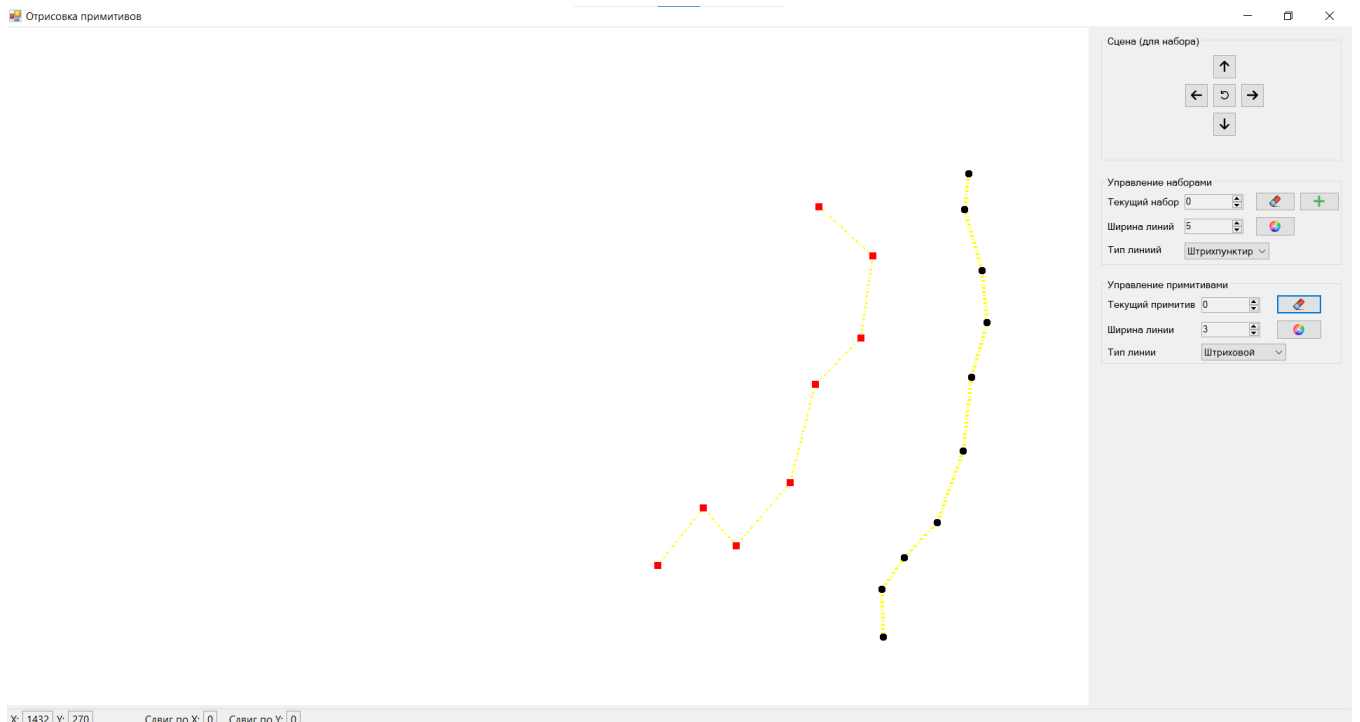
## Изменение типа линий выбранного набора



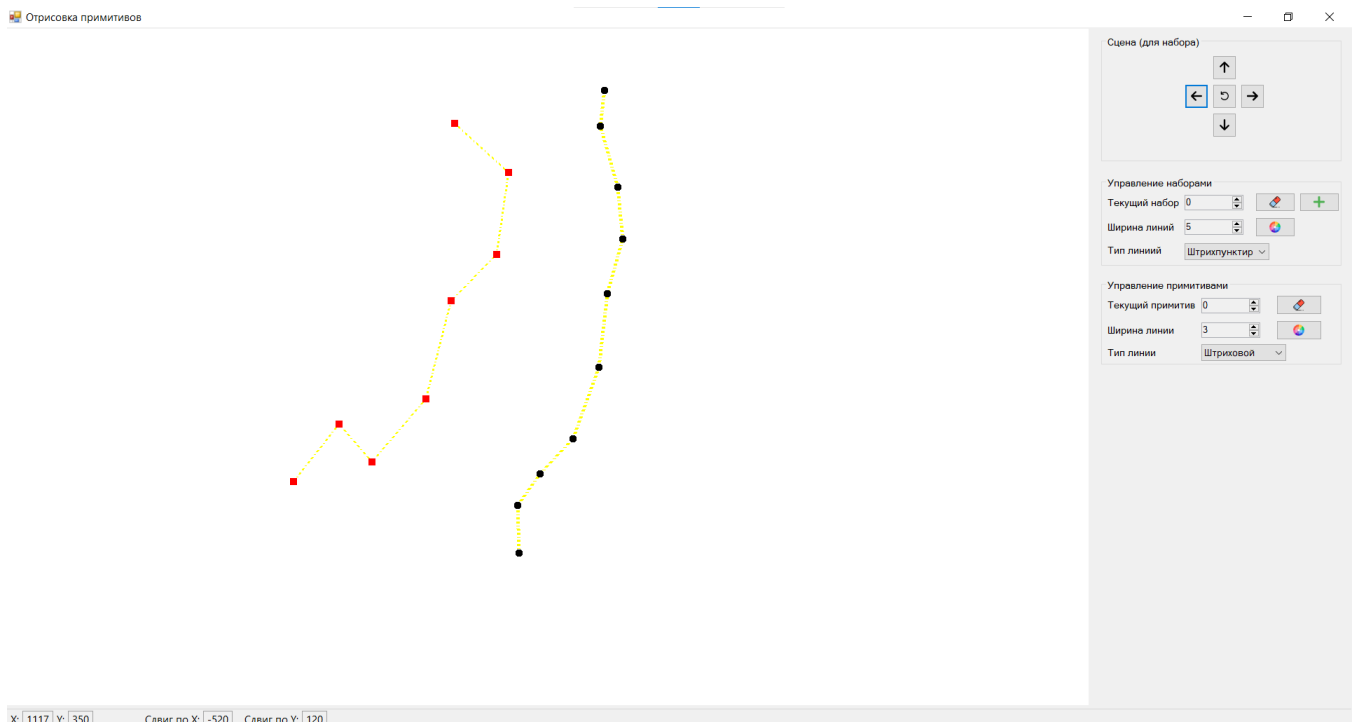
## Удаление текущего набора



## Удаление выбранного примитива в наборе



## Изменение положения выбранного набора примитивов

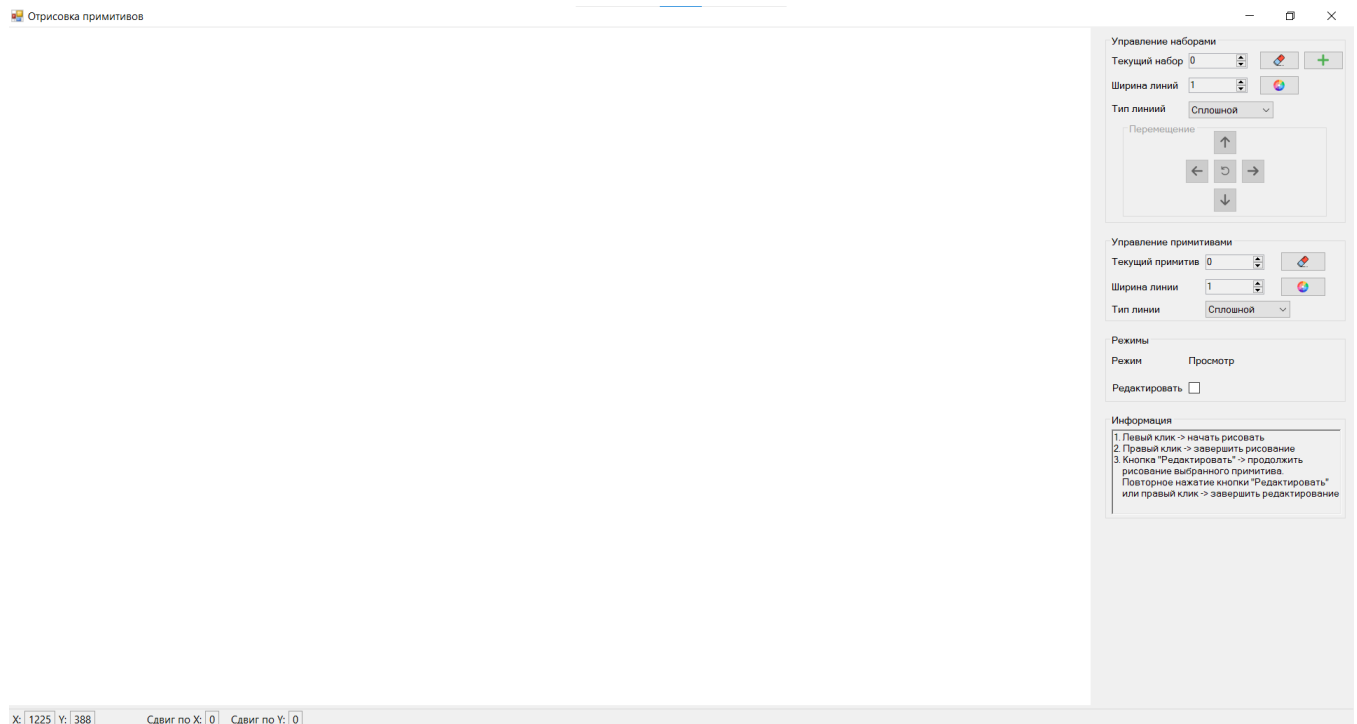


# Нововведения в программе

## Нововведения:

1. Изменение разметки приложения.
2. Добавлены режимы "Просмотр", "Рисование" и "Редактирование".
3. Добавлена кнопка редактирования выбранного примитива.
4. Добавлено информационное поле с инструкцией использования программы.

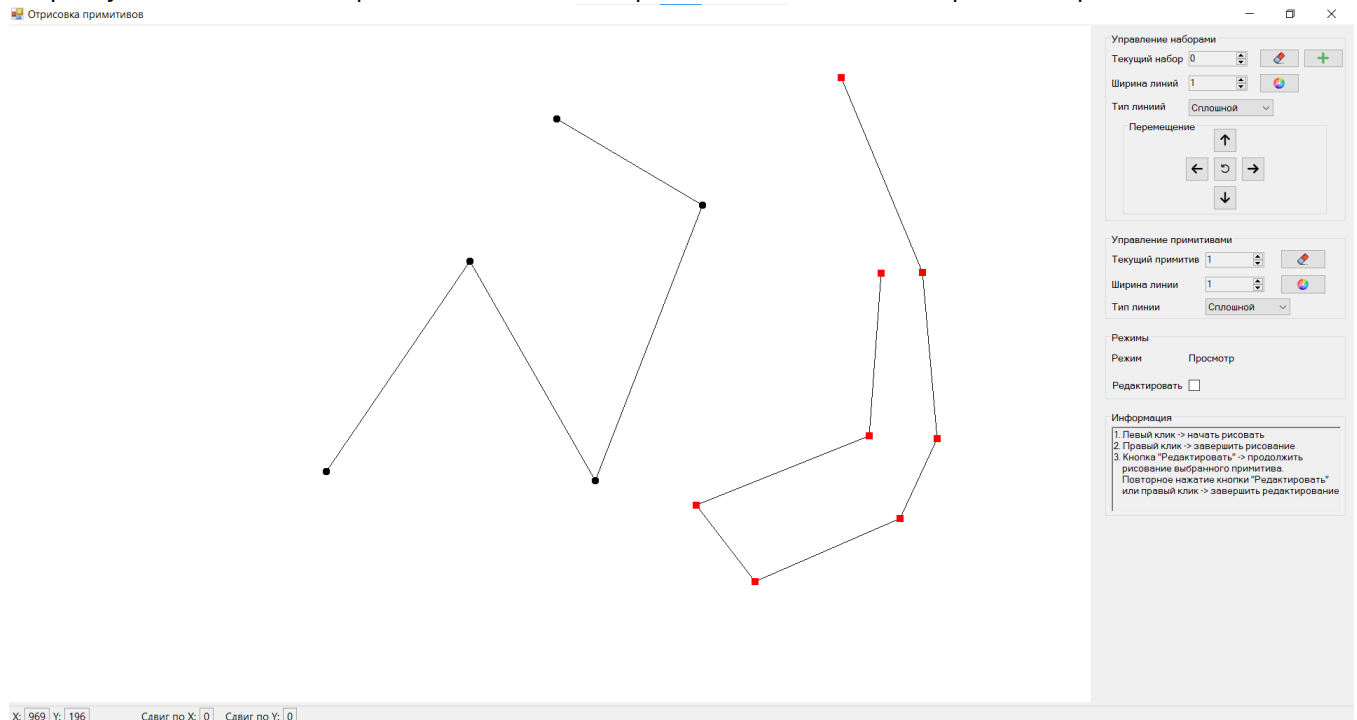
Обновленный вид окна:



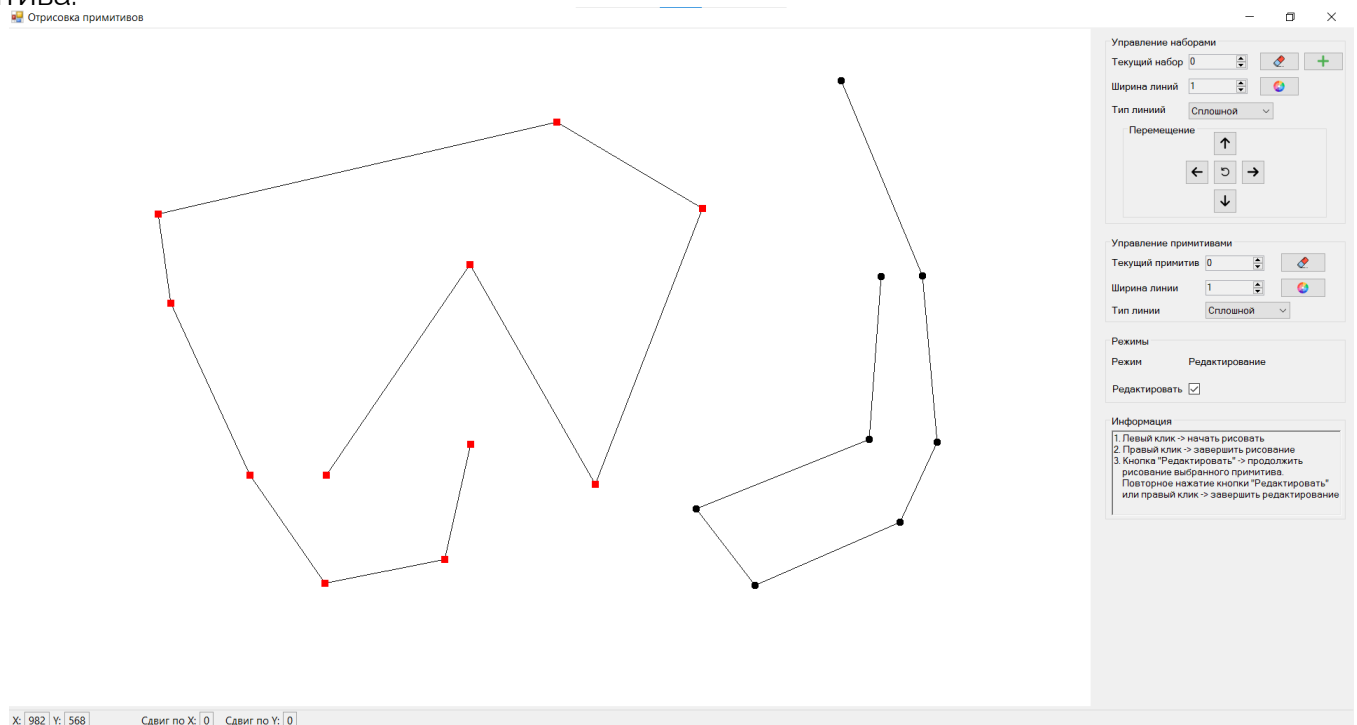
# Тестирование нововведений

## Редактирование выбранного примитива

Нарисуем несколько примитивов и выберем один из них для редактирования.



Далее, нажимая кнопку "Редактировать", продолжаем рисование текущего примитива.



# ЛИСТИНГ

## MainForm.cs

```
1 using System;
2 using System.Collections.Generic;
3 using System.Drawing;
4 using System.Windows.Forms;
5 using ComputerGraphics.Source;
6 using SharpGL;
7
8 namespace ComputerGraphics
9 {
10     public partial class MainForm : Form
11     {
12         private readonly List<List<StripLine>> _lines = new List<List<StripLine>>();
13         private readonly List<Point2D> _shifts = new List<Point2D>();
14         private readonly List<ushort> _stipples = new List<ushort>();
15         private readonly StripLine _line = new StripLine();
16         private byte _currentSet;
17         private byte _currentLine;
18         private bool _isDrawingCurrent;
19
20         public MainForm()
21         {
22             InitializeComponent();
23             comboBoxLine.SelectedIndex = 0;
24             comboBoxSet.SelectedIndex = 0;
25         }
26
27         private void GL_OpenGLInitialized(object sender, EventArgs e)
28         {
29             OpenGL gl = GL.OpenGL;
30
31             gl.Disable(OpenGL.GL_DEPTH_TEST);
32             gl.ClearColor(1f, 1f, 1f, 1f);
33             gl.Clear(OpenGL.GL_COLOR_BUFFER_BIT);
34
35             gl.MatrixMode(OpenGL.GL_PROJECTION);
36             gl.LoadIdentity();
37             gl.Ortho2D(0, GL.Width, 0, GL.Height);
38             gl.MatrixMode(OpenGL.GL_MODELVIEW);
39             gl.LoadIdentity();
40         }
41
42         private void GL_Resized(object sender, EventArgs e)
43         {
44             GL_OpenGLInitialized(sender, e);
45         }
46
47         private void GL_OpenGLDraw(object sender, RenderEventArgs args)
48         {
49             OpenGL gl = GL.OpenGL;
50
51             gl.Clear(OpenGL.GL_COLOR_BUFFER_BIT);
52
53             for (int iset = 0; iset < _lines.Count; iset++)
54             {
55                 for (int iline = 0; iline < _lines[iset].Count; iline++)
56                 {
```

```

57     var line = _lines[iset][iline];
58
59     gl.PushMatrix();
60     gl.Translate(_shifts[iset].X, _shifts[iset].Y, 0);
61     gl.Color(line.Color.R, line.Color.G, line.Color.B);
62     gl.Enable(OpenGL.GL_LINE_STIPPLE);
63     gl.LineStipple(1, line.Stipple);
64     gl.LineWidth(line.Thickness);
65     gl.Begin(OpenGL.GL_LINE_STRIP);
66
67     foreach (var p in line.Points)
68     {
69         gl.Vertex(p.X, p.Y);
70     }
71
72     gl.Disable(OpenGL.GL_LINE_STIPPLE);
73     gl.End();
74
75     // Текущий набор выделяем точками
76     if (iset == _currentSet)
77     {
78         gl.PointSize(10);
79
80         // Выделяем "активную" линию
81         if (iline == _currentLine)
82         {
83             gl.Color(1.0f, 0.0f, 0.0f);
84         }
85         else
86         {
87             gl.Color(0.0f, 0.0f, 0.0f);
88             gl.Enable(OpenGL.GL_POINT_SMOOTH);
89         }
90
91         gl.Begin(OpenGL.GL_POINTS);
92
93         foreach (var p in line.Points)
94         {
95             gl.Vertex(p.X, p.Y);
96         }
97
98         gl.End();
99
100        if (iline != _currentLine)
101        {
102            gl.Disable(OpenGL.GL_POINT_SMOOTH);
103        }
104    }
105
106    gl.PopMatrix();
107 }
108
109
110 if (_isDrawingCurrent)
111 {
112     ChangeSet.Enabled = false;
113     ChangePrimitive.Enabled = false;
114     gl.Color(_line.Color.R, _line.Color.G, _line.Color.B);
115     gl.LineWidth(_line.Thickness);
116     gl.Enable(OpenGL.GL_LINE_STIPPLE);

```

```

117         gl.LineStipple(1, _line.Stipple);
118         gl.Begin(OpenGL.GL_LINE_STRIP);
119
120         foreach (var p in _line.Points)
121         {
122             gl.Vertex(p.X, p.Y);
123         }
124
125         gl.Disable(OpenGL.GL_LINE_STIPPLE);
126         gl.End();
127
128         // Сразу выделяем линию точками
129         gl.PointSize(10);
130         gl.Color(1.0f, 0.0f, 0.0f);
131         gl.Begin(OpenGL.GL_POINTS);
132
133         foreach (var p in _line.Points)
134         {
135             gl.Vertex(p.X, p.Y);
136         }
137
138         gl.End();
139     }
140     else
141     {
142         ChangeSet.Enabled = true;
143         ChangePrimitive.Enabled = true;
144     }
145
146     gl.Finish();
147 }
148
149 private void GL_MouseClick(object sender, MouseEventArgs e)
150 {
151     if (e.Button == MouseButton.Left)
152     {
153         if (_lines.IsEmpty())
154         {
155             AddSet_Click(sender, e);
156         }
157
158         _isDrawingCurrent = true;
159
160         short mouseX = (short)e.X;
161         short mouseY = (short)(GL.Height - (short)e.Y);
162
163         _line.Points.Add(new Point2D(mouseX, mouseY));
164
165         if (_line.Points.Count == 1)
166         {
167             if (_lines[_currentSet].IsEmpty())
168             {
169                 _currentLine = 0;
170             }
171             else
172             {
173                 ChangePrimitive.Maximum = _lines[_currentSet].Count;
174                 ChangePrimitive.Value = ChangePrimitive.Maximum;
175                 _currentLine = (byte)ChangePrimitive.Maximum;
176             }
177         }
178     }
179 }

```



```

177         ChangePrimitive.Enabled = true;
178     }
179 }
180
181
182 if (e.Button == MouseButton.Right)
183 {
184     if (_line.Points.Count == 0) return;
185
186     _lines[_currentSet].Add(_line.Clone() as Stripline);
187     _line.Points.Clear();
188     _isDrawingCurrent = false;
189     Scene.Enabled = true;
190 }
191
192
193 private void GL_MouseMove(object sender, MouseEventArgs e)
194 {
195     short xPos = (short)e.X;
196     short yPos = (short)e.Y;
197
198     statusXPosValue.Text = xPos.ToString();
199     statusYPosValue.Text = yPos.ToString();
200 }
201
202
203 // Панель управления *****
204 // Управление сценой *****
205 private void UpBtn_Click(object sender, EventArgs e)
206 {
207     if (!_isDrawingCurrent && !_lines[_currentSet].IsEmpty())
208     {
209         _shifts[_currentSet] = new Point2D(_shifts[_currentSet].X,
↪ (short)(_shifts[_currentSet].Y + 40));
210         statusXShiftValue.Text = _shifts[_currentSet].X.ToString();
211         statusYShiftValue.Text = _shifts[_currentSet].Y.ToString();
212     }
213 }
214
215 private void RightBtn_Click(object sender, EventArgs e)
216 {
217     if (!_isDrawingCurrent && !_lines[_currentSet].IsEmpty())
218     {
219         _shifts[_currentSet] = new Point2D((short)(_shifts[_currentSet].X +
↪ 40), _shifts[_currentSet].Y);
220         statusXShiftValue.Text = _shifts[_currentSet].X.ToString();
221         statusYShiftValue.Text = _shifts[_currentSet].Y.ToString();
222     }
223 }
224
225 private void LeftBtn_Click(object sender, EventArgs e)
226 {
227     if (!_isDrawingCurrent && !_lines[_currentSet].IsEmpty())
228     {
229         _shifts[_currentSet] = new Point2D((short)(_shifts[_currentSet].X -
↪ 40), _shifts[_currentSet].Y);
230         statusXShiftValue.Text = _shifts[_currentSet].X.ToString();
231         statusYShiftValue.Text = _shifts[_currentSet].Y.ToString();
232     }
233 }

```

```

234     private void DownBtn_Click(object sender, EventArgs e)
235     {
236         if (!_isDrawingCurrent && !_lines[_currentSet].IsEmpty())
237         {
238             _shifts[_currentSet] = new Point2D(_shifts[_currentSet].X,
239 → (short)(_shifts[_currentSet].Y - 40));
240             statusXShiftValue.Text = _shifts[_currentSet].X.ToString();
241             statusYShiftValue.Text = _shifts[_currentSet].Y.ToString();
242         }
243     }
244
245     private void ResetBtn_Click(object sender, EventArgs e)
246     {
247         _shifts[_currentSet] = new Point2D();
248         statusXShiftValue.Text = _shifts[_currentSet].X.ToString();
249         statusYShiftValue.Text = _shifts[_currentSet].Y.ToString();
250     }
251
252     // Управление наборами *****
253     private void ChangeSet_ValueChanged(object sender, EventArgs e)
254     {
255         if (ChangeSet.Value != 0 && ChangeSet.Value == _lines.Count)
256         {
257             ChangeSet.Value--;
258         }
259
260         _currentSet = (byte)ChangeSet.Value;
261
262         if (!_lines.IsEmpty())
263         {
264             if (_lines[_currentSet].IsEmpty())
265             {
266                 ChangePrimitive.Maximum = 0;
267             }
268             else
269             {
270                 ChangePrimitive.Maximum = _lines[_currentSet].Count - 1;
271             }
272         }
273     }
274
275     private void AddSet_Click(object sender, EventArgs e)
276     {
277         // Если кнопка "Создать новый набор" нажата до завершения рисования
278         // примитива, то принудительно завершаем его рисование
279         if (_isDrawingCurrent)
280         {
281             _lines[_currentSet].Add(_line.Clone() as StriLine);
282             _line.Reset();
283             _isDrawingCurrent = false;
284             Scene.Enabled = true;
285         }
286
287         // Если нет еще ни одного набора -> создаем его
288         if (_lines.IsEmpty())
289         {
290             ChangeSet.Enabled = true;
291             _lines.Add(new List<StriLine>());
292

```

```

293         _shifts.Add(new Point2D());
294         _stipples.Add(0xFFFF);
295         return;
296     }
297
298     // Создать новый набор можно только в том случае, если предшествующий
299     // ему набор не пуст
300     if (!_lines[_currentSet].IsEmpty())
301     {
302         _line.Reset();
303
304         _lines.Add(new List<StripLine>());
305         _shifts.Add(new Point2D());
306         _stipples.Add(0xFFFF);
307
308         ChangeSet.Maximum = _lines.Count - 1;
309         ChangeSet.Value = ChangeSet.Maximum;
310
311         ChangeWidthS.Value = 1;
312         ChangePrimitive.Value = 0;
313         ChangePrimitive.Maximum = 0;
314     }
315 }
316
317 private void DeleteSet_Click(object sender, EventArgs e)
318 {
319     // Удалять можно только если не рисуется примитив
320     // либо если есть хотя бы один набор
321     if (!_isDrawingCurrent && !_lines.IsEmpty())
322     {
323         _currentSet = (byte)ChangeSet.Value;
324
325         _lines.RemoveAt(_currentSet);
326         _shifts.RemoveAt(_currentSet);
327         _stipples.RemoveAt(_currentSet);
328
329         ChangeSet_ValueChanged(sender, e);
330
331         ChangeSet.Maximum = _lines.Count == 0 ? 0 : _lines.Count - 1;
332         ChangeSet.Value = ChangeSet.Maximum;
333     }
334
335     // Не отображаем "Текущий набор", если их нет
336     if (_lines.IsEmpty())
337     {
338         //ChangeSet.Enabled = false;
339         //ChangePrimitive.Enabled = false;
340         Scene.Enabled = false;
341     }
342 }
343
344 private void ChangeWidthS_ValueChanged(object sender, EventArgs e)
345 {
346     _line.Thickness = (float)ChangeWidthS.Value;
347
348     if (!_lines.IsEmpty())
349     {
350         foreach (var line in _lines[_currentSet])
351         {
352             line.Thickness = _line.Thickness;

```

```

353     }
354 }
355 }
356
357 private void ChangeColorS_Click(object sender, EventArgs e)
358 {
359     colorDialog1.ShowDialog();
360
361     _line.Color = colorDialog1.Color;
362
363     if (!_lines.IsEmpty())
364     {
365         foreach (var line in _lines[_currentSet])
366         {
367             line.Color = _line.Color;
368         }
369     }
370 }
371
372 private void comboBoxSet_SelectedIndexChanged(object sender, EventArgs e)
373 {
374     switch (comboBoxSet.SelectedIndex)
375     {
376         case 0:
377             _line.Stipple = 0xFFFF;
378             break;
379         case 1:
380             _line.Stipple = 0x0101;
381             break;
382         case 2:
383             _line.Stipple = 0x00F0;
384             break;
385         case 3:
386             _line.Stipple = 0x1C47;
387             break;
388     }
389
390     if (!_lines.IsEmpty())
391     {
392         foreach (var line in _lines[_currentSet])
393         {
394             line.Stipple = _line.Stipple;
395         }
396     }
397 }
398
399
400 // Управление примитивами *****
401 private void ChangeColorP_Click(object sender, EventArgs e)
402 {
403     colorDialog1.ShowDialog();
404
405     _line.Color = colorDialog1.Color;
406
407     if (!_lines.IsEmpty() && !_lines[_currentSet].IsEmpty())
408     {
409         _lines[_currentSet][_currentLine].Color = colorDialog1.Color;
410     }
411 }
412

```

```

413     private void DeletePrimitive_Click(object sender, EventArgs e)
414     {
415         if (!_isDrawingCurrent && !_lines.IsEmpty())
416         {
417             _lines[_currentSet].RemoveAt(_currentLine);
418             ChangePrimitive.Value = ChangePrimitive.Value == 0 ? 0 :
↪ --ChangePrimitive.Value;
419             ChangePrimitive.Maximum = ChangePrimitive.Maximum == 0 ? 0 :
↪ --ChangePrimitive.Maximum;
420         }
421         else return;
422
423         if (_lines[_currentSet].IsEmpty())
424         {
425             DeleteSet_Click(sender, e);
426         }
427     }
428
429     private void ChangePrimitive_ValueChanged(object sender, EventArgs e)
430     {
431         if (!_isDrawingCurrent) _currentLine = (byte)ChangePrimitive.Value;
432     }
433
434     private void ChangeWidthP_ValueChanged(object sender, EventArgs e)
435     {
436         _line.Thickness = (float)ChangeWidthP.Value;
437
438         if (!_isDrawingCurrent && !_lines.IsEmpty() &&
↪ !_lines[_currentSet].IsEmpty())
439         {
440             _lines[_currentSet][_currentLine].Thickness =
↪ (float)ChangeWidthP.Value;
441         }
442     }
443
444     private void comboBoxLine_SelectedIndexChanged(object sender, EventArgs e)
445     {
446         switch (comboBoxLine.SelectedIndex)
447         {
448             case 0:
449                 _line.Stipple = 0xFFFF;
450                 break;
451             case 1:
452                 _line.Stipple = 0x0101;
453                 break;
454             case 2:
455                 _line.Stipple = 0x00F0;
456                 break;
457             case 3:
458                 _line.Stipple = 0x1C47;
459                 break;
460         }
461
462         if (!_lines.IsEmpty() && !_lines[_currentSet].IsEmpty())
463         {
464             _lines[_currentSet][_currentLine].Stipple = _line.Stipple;
465         }
466     }
467 }
468 }

```

## Point2D.cs

```
1 namespace ComputerGraphics.Source
2 {
3     public struct Point2D
4     {
5         public short X { get; }
6         public short Y { get; }
7
8         public Point2D(short x, short y) => (X, Y) = (x, y);
9     }
10 }
```

## Primitive.cs

```
1 using System;
2 using System.Drawing;
3 using System.Collections.Generic;
4
5 namespace ComputerGraphics.Source
6 {
7     public class StripLine : ICloneable
8     {
9         public List<Point2D> Points { get; private set; }
10        public Color Color { get; set; }
11        public float Thickness { get; set; }
12        public ushort Stipple { get; set; }
13
14        public StripLine()
15        {
16            Points = new List<Point2D>();
17            Color = new Color();
18            Thickness = 1.0f;
19            Stipple = 0xFFFF;
20        }
21
22        public object Clone() => new StripLine
23        {
24            Points = new List<Point2D>(Points),
25            Color = Color,
26            Thickness = Thickness,
27            Stipple = Stipple
28        };
29
30        public void Reset()
31        {
32            Points.Clear();
33            Color = new Color();
34            Thickness = 1.0f;
35            Stipple = 0xFFFF;
36        }
37    }
38 }
```

## Extensions.cs

```
1 using System.Collections.Generic;
2 using System.Linq;
3
4 namespace ComputerGraphics.Source
5 {
6     public static class EnumerableExtensions
7     {
8         public static bool IsEmpty<T>(this IEnumerable<T> collection) =>
9         ↪ !collection.Any();
10    }
```