Министерство образования Республики Беларусь

Учреждение образования

«Белорусский государственный университет

информатики и радиоэлектроники»

Институт информационных технологий

Факультет повышения квалификации и переподготовки

Кафедра МПСС

**ОТЧЕТ**

По лабораторным работам 5-6

по дисциплине ТКП

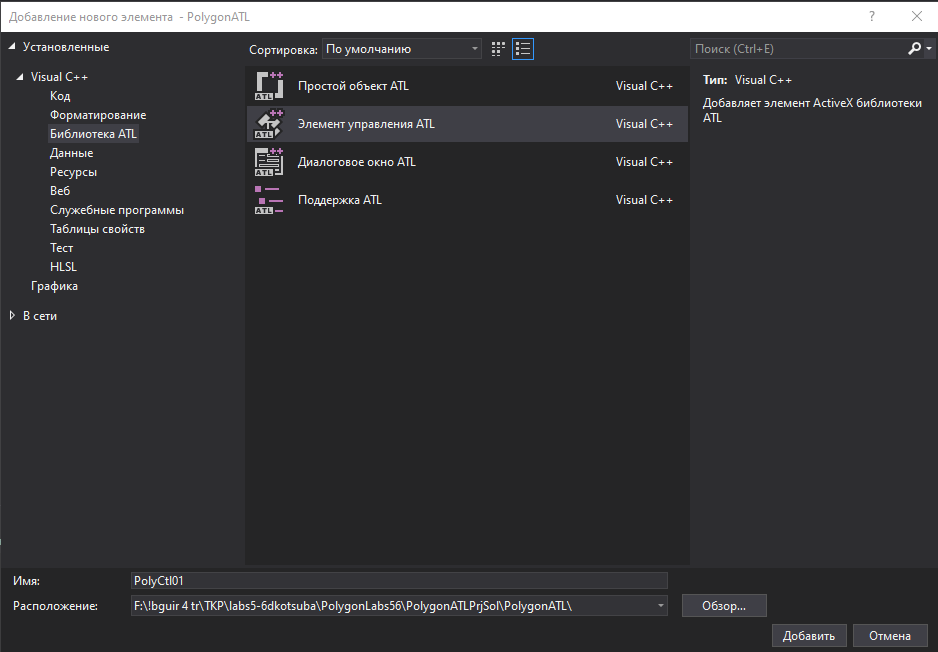
Тема: «Использование ATL для создания ActiveX элемента управления»

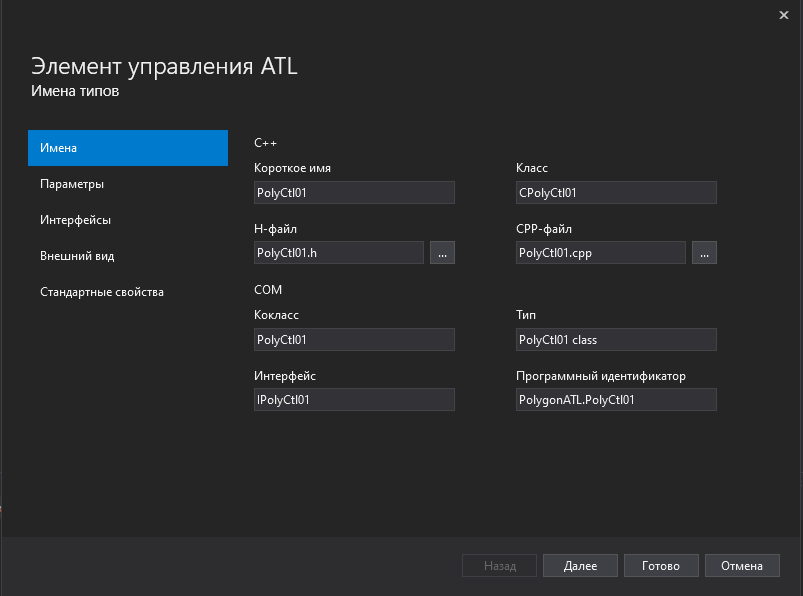
Минск 2021

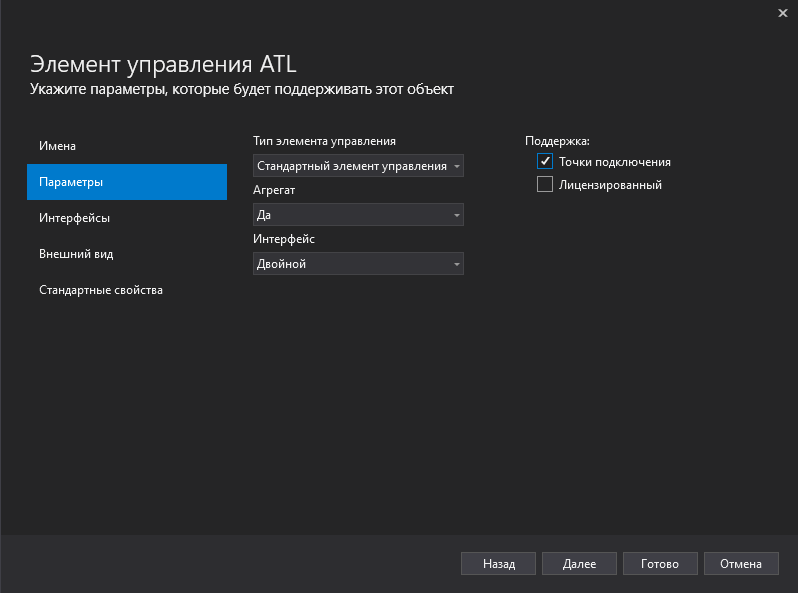
**Цель работы:** Создать элемент управления, отображающий вписанный в окружность многоугольник. Добавить к элементу свойство, отображающее количество сторон многоугольника и создать код для обновления изображения элемента управления после изменения значения этого свойства.

**Ход работы:**

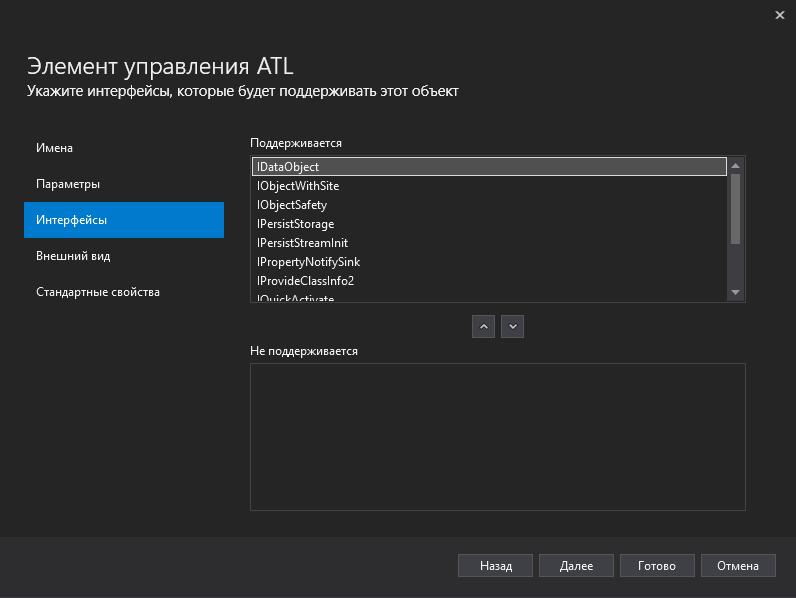
1. Создание проекта PolygonATL типа «Проект ATL» -> «Библиотека динамической компоновки (DLL)»;
2. Добавление в проект нового элемента – элемент управления ATL PolyCtl01. Построение и тестирование ActiveX

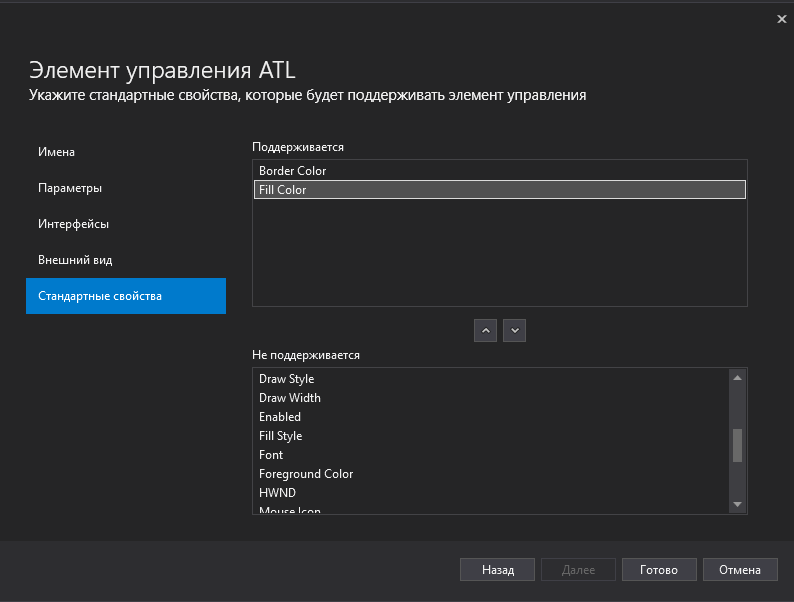


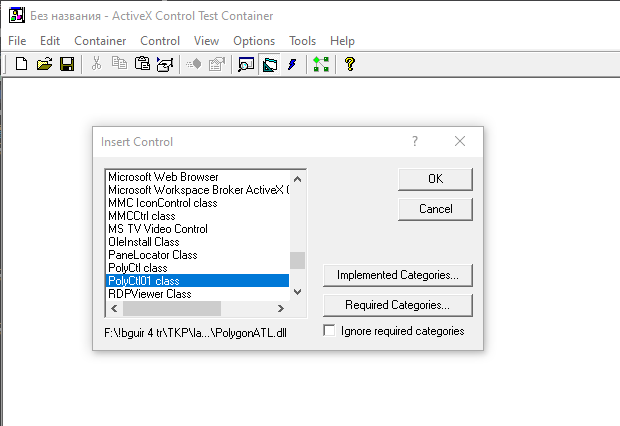


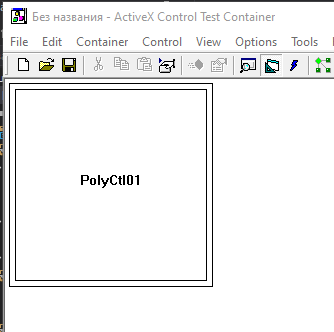


Добавим все элементы

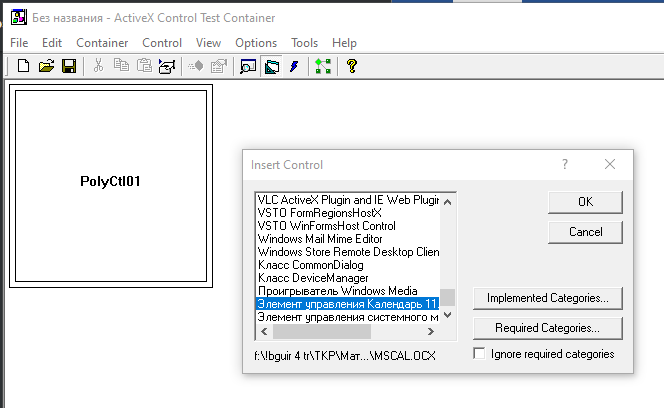


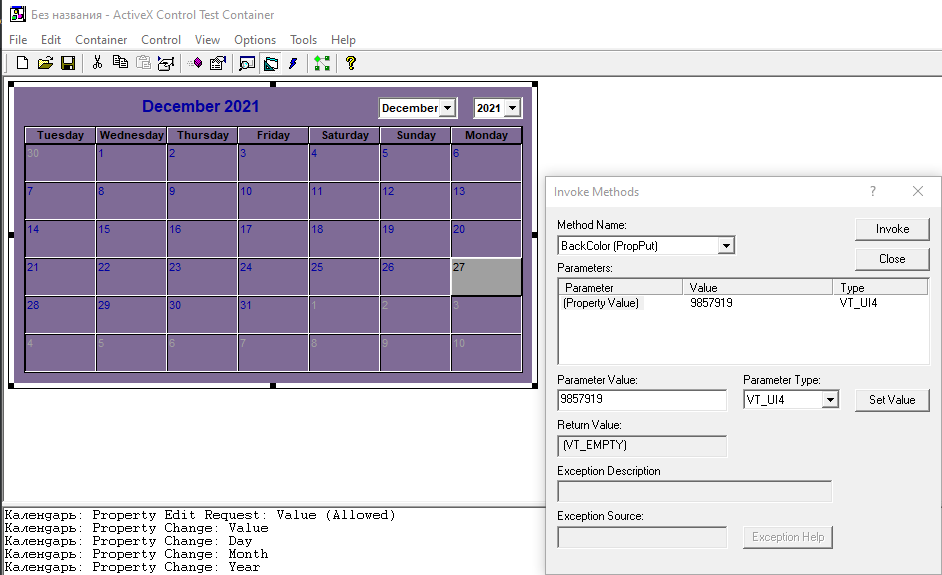




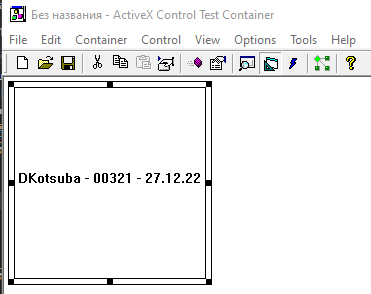


Тестирование элемента управления Календарь

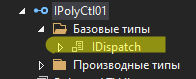


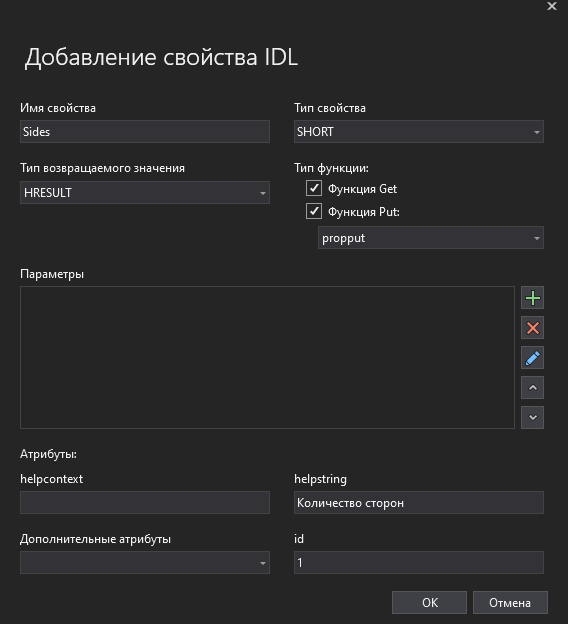


Внесем изменения в проект и протестируем



1. Добавление свойства к ActiveX Control



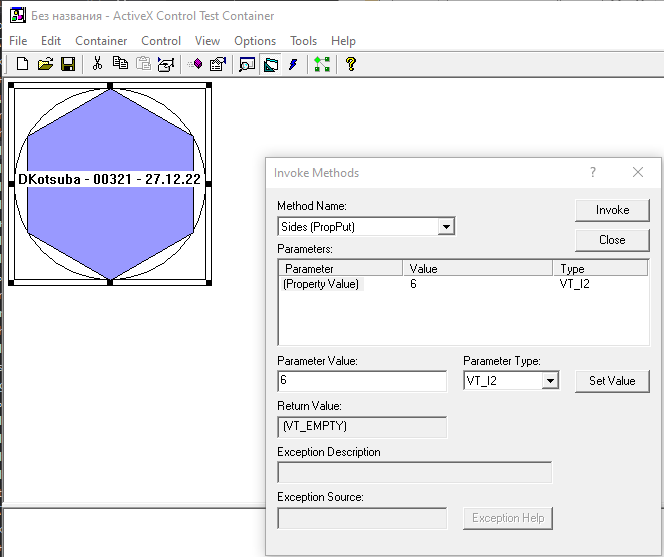


Добавим инициализацию необходимых переменных, функций, добавим реализацию.

|  |
| --- |
| // PolyCtl01.h […]  CPolyCtl01()  {  m\_nSides = 5;  m\_clrFillColor = RGB(153,153,255);  m\_clrBorderColor = RGB(0, 0, 102);  }  […]  /\*--------------------------------------------------------------------\*/  HDC hdc = di.hdcDraw;  COLORREF colFore;  HBRUSH hOldBrush, hBrush;  HPEN hOldPen, hPen;  // Translate m\_colFore into a COLORREF type  OleTranslateColor(m\_clrFillColor, NULL, &colFore);  // Create and select the colors to draw the circle  hPen = (HPEN)GetStockObject(BLACK\_PEN);  hOldPen = (HPEN)SelectObject(hdc, hPen);  hBrush = (HBRUSH)GetStockObject(WHITE\_BRUSH);  hOldBrush = (HBRUSH)SelectObject(hdc, hBrush);  Ellipse(hdc, rc.left, rc.top, rc.right, rc.bottom);  // Create and select the brush that will be used to fill the polygon  hBrush = CreateSolidBrush(colFore);  SelectObject(hdc, hBrush);  CalcPoints(rc);  Polygon(hdc, &m\_arrPoint[0], m\_nSides);  // Select back the old pen and brush and delete the brush we created  SelectObject(hdc, hOldPen);  SelectObject(hdc, hOldBrush);  DeleteObject(hBrush);  /\*--------------------------------------------------------------------\*/  […] |

|  |
| --- |
| // PolyCtl01.cpp  STDMETHODIMP CPolyCtl01::get\_Sides(SHORT\* pVal)  {  // TODO: Добавьте сюда код реализации  \*pVal = m\_nSides;  return S\_OK;  }  STDMETHODIMP CPolyCtl01::put\_Sides(SHORT newVal)  {  // TODO: Добавьте сюда код реализации  if (newVal < 3) {  return ERROR("Сторон не может быть меньше 3-х");  }  else if (newVal > 10) {  return ERROR("Сторон не может быть больше 10-ти");  }  else m\_nSides = newVal;  FireViewChange();  return S\_OK;  }  void CPolyCtl01::CalcPoints(const RECT& rc)  {  const double pi = 3.14159265358979;  POINT ptCenter;  double dblRadiusx = (rc.right - rc.left) / 2;  double dblRadiusy = (rc.bottom - rc.top) / 2;  double dblAngle = 3 \* pi / 2; // Start at the top  double dblDiff = 2 \* pi / m\_nSides; // Angle each side will make  ptCenter.x = (rc.left + rc.right) / 2;  ptCenter.y = (rc.top + rc.bottom) / 2;  // Calculate the points for each side  for (int i = 0; i < m\_nSides; i++)  {  m\_arrPoint[i].x = (long)(dblRadiusx \*  cos(dblAngle) + ptCenter.x + 0.5);  m\_arrPoint[i].y = (long)(dblRadiusy \*  sin(dblAngle) + ptCenter.y + 0.5);  dblAngle += dblDiff;  } |

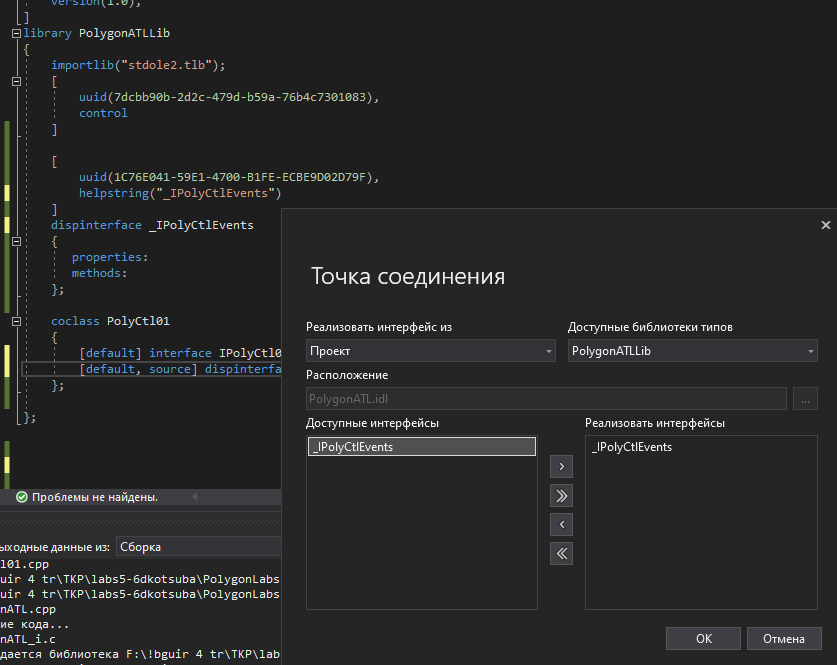
Протестируем



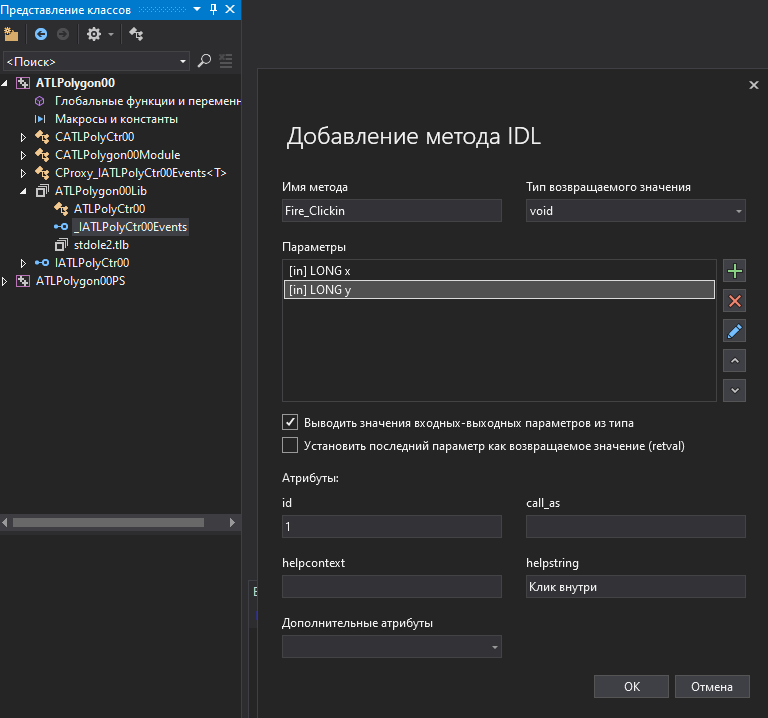
1. Добавление поддержки точки подключения.

Добавление в IDL-файл диспетчерского интерфейса, GUID

|  |
| --- |
| [  uuid(1C76E041-59E1-4700-B1FE-ECBE9D02D79F),  helpstring("Buddy Events")  ]  dispinterface DBuddyEvents  {  properties:  methods:  }; |



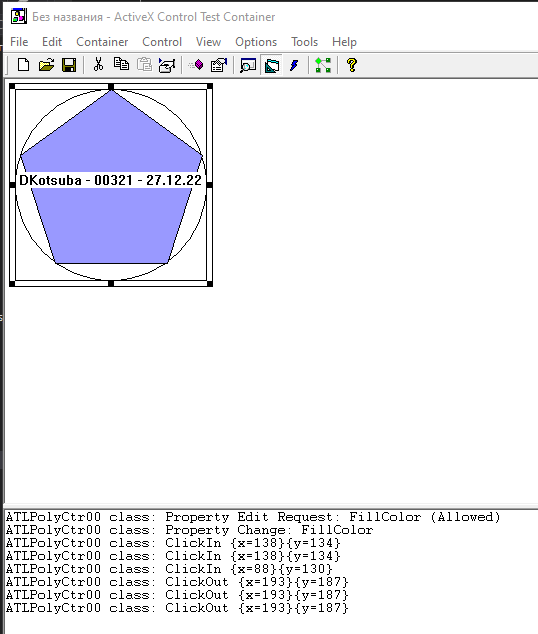
1. Добавление методов Fire\_Clickin, Fire\_Clickout в интерфейс \_IATLPolyCtl00Events с помощью мастера



1. Реализация кода отрисовки многоугольника в методе OnDraw в файле

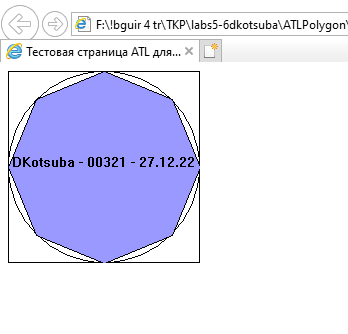
|  |
| --- |
| // ATLPolyCtr00.cpp  LRESULT CATLPolyCtr00::OnLButtonDown(UINT uMsg, WPARAM wParam, LPARAM lParam, BOOL& bHandled) {  HRGN hRgn;  WORD xPos = LOWORD(lParam); // horizontal position of cursor  WORD yPos = HIWORD(lParam); // vertical position of cursor  CalcPoints(m\_rcPos);  // Create a region from our list of points  hRgn = CreatePolygonRgn(&m\_arrPoint[0], m\_nSides, WINDING);  // If the clicked point is in our polygon then fire the ClickIn  // event otherwise we fire the ClickOut event  if (PtInRegion(hRgn, xPos, yPos))  Fire\_ClickIn(xPos, yPos);  else  Fire\_ClickOut(xPos, yPos);  // Delete the region that we created  DeleteObject(hRgn);  return 0;  } |

1. Добавление методов Fire\_ClickIn и Fire\_ClickOut, тестирование



1. Добавление javascript-скрипта в htm-файл.

|  |
| --- |
| <OBJECT ID="PolyCtl" CLASSID="CLSID:e46063a6-f27c-4ba7-b9c8-67de75e6cfc3"></OBJECT>  <SCRIPT>  function PolyCtl:: ClickIn(x, y)  {  PolyCtl.Sides = PolyCtl.Sides + 1;  }  function PolyCtl:: ClickOut(x, y) {  PolyCtl.Sides = PolyCtl.Sides - 1;  }  </SCRIPT> |



Проект ***ATLPolygon00***

|  |
| --- |
| // ATLPolyCtr00.cpp: реализация CATLPolyCtr00  #include "pch.h"  #include "ATLPolyCtr00.h"  // CATLPolyCtr00  STDMETHODIMP CATLPolyCtr00::get\_Sides(SHORT\* pVal)  {  // TODO: Добавьте сюда код реализации  \*pVal = m\_nSides;  return S\_OK;  }  STDMETHODIMP CATLPolyCtr00::put\_Sides(SHORT newVal)  {  // TODO: Добавьте сюда код реализации  if (newVal < 3) {  return ERROR("Сторон не может быть меньше 3-х");  }  else if (newVal > 10) {  return ERROR("Сторон не может быть больше 10-ти");  }  else m\_nSides = newVal;  FireViewChange();  return S\_OK;  }  void CATLPolyCtr00::CalcPoints(const RECT& rc)  {  const double pi = 3.14159265358979;  POINT ptCenter;  double dblRadiusx = (rc.right - rc.left) / 2;  double dblRadiusy = (rc.bottom - rc.top) / 2;  double dblAngle = 3 \* pi / 2; // Start at the top  double dblDiff = 2 \* pi / m\_nSides; // Angle each side will make  ptCenter.x = (rc.left + rc.right) / 2;  ptCenter.y = (rc.top + rc.bottom) / 2;  // Calculate the points for each side  for (int i = 0; i < m\_nSides; i++)  {  m\_arrPoint[i].x = (long)(dblRadiusx \*  cos(dblAngle) + ptCenter.x + 0.5);  m\_arrPoint[i].y = (long)(dblRadiusy \*  sin(dblAngle) + ptCenter.y + 0.5);  dblAngle += dblDiff;  }  }  LRESULT CATLPolyCtr00::OnLButtonDown(UINT uMsg, WPARAM wParam, LPARAM lParam, BOOL& bHandled) {  HRGN hRgn;  WORD xPos = LOWORD(lParam); // horizontal position of cursor  WORD yPos = HIWORD(lParam); // vertical position of cursor  CalcPoints(m\_rcPos);  // Create a region from our list of points  hRgn = CreatePolygonRgn(&m\_arrPoint[0], m\_nSides, WINDING);  //If the clicked point is in our polygon then fire the ClickIn  // event otherwise we fire the ClickOut event    if (PtInRegion(hRgn, xPos, yPos))  Fire\_ClickIn(xPos, yPos);  else  Fire\_ClickOut(xPos, yPos);    // Delete the region that we created  DeleteObject(hRgn);  return 0;  } |

|  |
| --- |
| // ATLPolygon00.cpp: реализация экспортов DLL.  #include "pch.h"  #include "framework.h"  #include "resource.h"  #include "ATLPolygon00\_i.h"  #include "dllmain.h"  using namespace ATL;  // Используется, чтобы определить, можно ли выгрузить DLL средствами OLE.  \_Use\_decl\_annotations\_  STDAPI DllCanUnloadNow(void)  {  return \_AtlModule.DllCanUnloadNow();  }  // Возвращает фабрику класса для создания объекта требуемого типа.  \_Use\_decl\_annotations\_  STDAPI DllGetClassObject(\_In\_ REFCLSID rclsid, \_In\_ REFIID riid, \_Outptr\_ LPVOID\* ppv)  {  return \_AtlModule.DllGetClassObject(rclsid, riid, ppv);  }  // DllRegisterServer - добавляет записи в системный реестр.  \_Use\_decl\_annotations\_  STDAPI DllRegisterServer(void)  {  // регистрирует объект, библиотеку типов и все интерфейсы в библиотеке типов  HRESULT hr = \_AtlModule.DllRegisterServer();  return hr;  }  // DllUnregisterServer - удаляет записи из системного реестра.  \_Use\_decl\_annotations\_  STDAPI DllUnregisterServer(void)  {  HRESULT hr = \_AtlModule.DllUnregisterServer();  return hr;  }  // DllInstall - добавляет и удаляет записи системного реестра для конкретного пользователя и компьютера.  STDAPI DllInstall(BOOL bInstall, \_In\_opt\_ LPCWSTR pszCmdLine)  {  HRESULT hr = E\_FAIL;  static const wchar\_t szUserSwitch[] = L"user";  if (pszCmdLine != nullptr)  {  if (\_wcsnicmp(pszCmdLine, szUserSwitch, \_countof(szUserSwitch)) == 0)  {  ATL::AtlSetPerUserRegistration(true);  }  }  if (bInstall)  {  hr = DllRegisterServer();  if (FAILED(hr))  {  DllUnregisterServer();  }  }  else  {  hr = DllUnregisterServer();  }  return hr;  } |

|  |
| --- |
| ; ATLPolygon00.def: объявляет параметры модуля.  LIBRARY  EXPORTS  DllCanUnloadNow PRIVATE  DllGetClassObject PRIVATE  DllRegisterServer PRIVATE  DllUnregisterServer PRIVATE  DllInstall PRIVATE |

|  |
| --- |
| // ATLPolygon00.idl: источник языка IDL для ATLPolygon00  //  // Этот файл будет обработан средством MIDL для  // создание библиотеки типов ($handlersafeidlname$.tlb) и кода маршалирования.  #include "olectl.h"  import "oaidl.idl";  import "ocidl.idl";  [  object,  uuid(92d1b9c0-2131-47d1-92c8-979c1ce5c592),  dual,  nonextensible,  pointer\_default(unique)  ]  interface IATLPolyCtr00 : IDispatch  {  [propput, bindable, requestedit, id(DISPID\_BORDERCOLOR)]  HRESULT BorderColor([in]OLE\_COLOR clr);  [propget, bindable, requestedit, id(DISPID\_BORDERCOLOR)]  HRESULT BorderColor([out, retval]OLE\_COLOR\* pclr);  [propput, bindable, requestedit, id(DISPID\_BORDERWIDTH)]  HRESULT BorderWidth([in]long width);  [propget, bindable, requestedit, id(DISPID\_BORDERWIDTH)]  HRESULT BorderWidth([out, retval]long\* width);  [propput, bindable, requestedit, id(DISPID\_FILLCOLOR)]  HRESULT FillColor([in]OLE\_COLOR clr);  [propget, bindable, requestedit, id(DISPID\_FILLCOLOR)]  HRESULT FillColor([out, retval]OLE\_COLOR\* pclr);  [propget, id(1), helpstring("Количество сторон")] HRESULT Sides([out, retval] SHORT\* pVal);  [propput, id(1), helpstring("Количество сторон")] HRESULT Sides([in] SHORT newVal);  };  [  uuid(5d3b508f-3ce0-4522-a4c3-3bad4b7ce3c6),  version(1.0),  ]  library ATLPolygon00Lib  {  importlib("stdole2.tlb");  [  uuid(fb723a2a-8aa7-4743-978a-1db8e5257254)  ]  dispinterface \_IATLPolyCtr00Events  {  properties:  methods:  [id(1), helpstring("Клик внутри")] void ClickIn([in] LONG x,[in] LONG y);  [id(2), helpstring("Клик вне")] void ClickOut([in] LONG x, [in] LONG y);  };  [  uuid(227d0672-6176-4dce-b654-2f0a47870912),  control  ]  coclass ATLPolyCtr00  {  [default] interface IATLPolyCtr00;  [default, source] dispinterface \_IATLPolyCtr00Events;  };  }; |

|  |
| --- |
| #pragma once  using namespace ATL;  template <class T>  class CProxy\_IATLPolyCtr00Events : public IConnectionPointImpl<T, &\_\_uuidof(\_IATLPolyCtr00Events), CComDynamicUnkArray>  {  // ПРЕДУПРЕЖДЕНИЕ: Мастер может повторить создание этого класса  public:  HRESULT Fire\_ClickIn(LONG x, LONG y)  {  HRESULT hr = S\_OK;  T\* pThis = static\_cast<T\*>(this);  int cConnections = m\_vec.GetSize();  for (int iConnection = 0; iConnection < cConnections; iConnection++)  {  pThis->Lock();  CComPtr<IUnknown> punkConnection = m\_vec.GetAt(iConnection);  pThis->Unlock();  IDispatch\* pConnection = static\_cast<IDispatch\*>(punkConnection.p);  if (pConnection)  {  CComVariant avarParams[2];  avarParams[1] = x;  avarParams[1].vt = VT\_I4;  avarParams[0] = y;  avarParams[0].vt = VT\_I4;  DISPPARAMS params = { avarParams, NULL, 2, 0 };  hr = pConnection->Invoke(1, IID\_NULL, LOCALE\_USER\_DEFAULT, DISPATCH\_METHOD, &params, NULL, NULL, NULL);  }  }  return hr;  }  HRESULT Fire\_ClickOut(LONG x, LONG y)  {  HRESULT hr = S\_OK;  T\* pThis = static\_cast<T\*>(this);  int cConnections = m\_vec.GetSize();  for (int iConnection = 0; iConnection < cConnections; iConnection++)  {  pThis->Lock();  CComPtr<IUnknown> punkConnection = m\_vec.GetAt(iConnection);  pThis->Unlock();  IDispatch\* pConnection = static\_cast<IDispatch\*>(punkConnection.p);  if (pConnection)  {  CComVariant avarParams[2];  avarParams[1] = x;  avarParams[1].vt = VT\_I4;  avarParams[0] = y;  avarParams[0].vt = VT\_I4;  DISPPARAMS params = { avarParams, NULL, 2, 0 };  hr = pConnection->Invoke(2, IID\_NULL, LOCALE\_USER\_DEFAULT, DISPATCH\_METHOD, &params, NULL, NULL, NULL);  }  }  return hr;  }  }; |

|  |
| --- |
| // ATLPolyCtr00.h: объявление CATLPolyCtr00  #pragma once  #include "resource.h" // основные символы  #include <atlctl.h>  #include "ATLPolygon00\_i.h"  #include "\_IATLPolyCtr00Events\_CP.h"  #include <math.h>  #if defined(\_WIN32\_WCE) && !defined(\_CE\_DCOM) && !defined(\_CE\_ALLOW\_SINGLE\_THREADED\_OBJECTS\_IN\_MTA)  #error "Однопотоковые COM-объекты не поддерживаются должным образом платформой Windows CE, например платформами Windows Mobile, в которых не предусмотрена полная поддержка DCOM. Определите \_CE\_ALLOW\_SINGLE\_THREADED\_OBJECTS\_IN\_MTA, чтобы принудить ATL поддерживать создание однопотоковых COM-объектов и разрешить использование его реализаций однопотоковых COM-объектов. Для потоковой модели в вашем rgs-файле задано значение 'Free', поскольку это единственная потоковая модель, поддерживаемая не-DCOM платформами Windows CE."  #endif  using namespace ATL;  // CATLPolyCtr00  class ATL\_NO\_VTABLE CATLPolyCtr00 :  public CComObjectRootEx<CComSingleThreadModel>,  public CStockPropImpl<CATLPolyCtr00, IATLPolyCtr00>,  public IPersistStreamInitImpl<CATLPolyCtr00>,  public IOleControlImpl<CATLPolyCtr00>,  public IOleObjectImpl<CATLPolyCtr00>,  public IOleInPlaceActiveObjectImpl<CATLPolyCtr00>,  public IViewObjectExImpl<CATLPolyCtr00>,  public IOleInPlaceObjectWindowlessImpl<CATLPolyCtr00>,  public ISupportErrorInfo,  public IConnectionPointContainerImpl<CATLPolyCtr00>,  public CProxy\_IATLPolyCtr00Events<CATLPolyCtr00>,  public IObjectWithSiteImpl<CATLPolyCtr00>,  public IServiceProviderImpl<CATLPolyCtr00>,  public IPersistStorageImpl<CATLPolyCtr00>,  public ISpecifyPropertyPagesImpl<CATLPolyCtr00>,  public IQuickActivateImpl<CATLPolyCtr00>,  #ifndef \_WIN32\_WCE  public IDataObjectImpl<CATLPolyCtr00>,  #endif  public IProvideClassInfo2Impl<&CLSID\_ATLPolyCtr00, &\_\_uuidof(\_IATLPolyCtr00Events), &LIBID\_ATLPolygon00Lib>,  public IPropertyNotifySinkCP<CATLPolyCtr00>,  public IObjectSafetyImpl<CATLPolyCtr00, INTERFACESAFE\_FOR\_UNTRUSTED\_CALLER>,  public CComCoClass<CATLPolyCtr00, &CLSID\_ATLPolyCtr00>,  public CComControl<CATLPolyCtr00>  {  public:  CATLPolyCtr00()  {  m\_nSides = 5;  m\_clrFillColor = RGB(153, 153, 255);  m\_clrBorderColor = RGB(0, 0, 102);  }  DECLARE\_OLEMISC\_STATUS(OLEMISC\_RECOMPOSEONRESIZE |  OLEMISC\_CANTLINKINSIDE |  OLEMISC\_INSIDEOUT |  OLEMISC\_ACTIVATEWHENVISIBLE |  OLEMISC\_SETCLIENTSITEFIRST  )  DECLARE\_REGISTRY\_RESOURCEID(IDR\_ATLPOLYCTR00)  BEGIN\_COM\_MAP(CATLPolyCtr00)  COM\_INTERFACE\_ENTRY(IATLPolyCtr00)  COM\_INTERFACE\_ENTRY(IDispatch)  COM\_INTERFACE\_ENTRY(IViewObjectEx)  COM\_INTERFACE\_ENTRY(IViewObject2)  COM\_INTERFACE\_ENTRY(IViewObject)  COM\_INTERFACE\_ENTRY(IOleInPlaceObjectWindowless)  COM\_INTERFACE\_ENTRY(IOleInPlaceObject)  COM\_INTERFACE\_ENTRY2(IOleWindow, IOleInPlaceObjectWindowless)  COM\_INTERFACE\_ENTRY(IOleInPlaceActiveObject)  COM\_INTERFACE\_ENTRY(IOleControl)  COM\_INTERFACE\_ENTRY(IOleObject)  COM\_INTERFACE\_ENTRY(IPersistStreamInit)  COM\_INTERFACE\_ENTRY2(IPersist, IPersistStreamInit)  COM\_INTERFACE\_ENTRY(ISupportErrorInfo)  COM\_INTERFACE\_ENTRY(IConnectionPointContainer)  COM\_INTERFACE\_ENTRY(ISpecifyPropertyPages)  COM\_INTERFACE\_ENTRY(IQuickActivate)  COM\_INTERFACE\_ENTRY(IPersistStorage)  #ifndef \_WIN32\_WCE  COM\_INTERFACE\_ENTRY(IDataObject)  #endif  COM\_INTERFACE\_ENTRY(IProvideClassInfo)  COM\_INTERFACE\_ENTRY(IProvideClassInfo2)  COM\_INTERFACE\_ENTRY(IObjectWithSite)  COM\_INTERFACE\_ENTRY(IServiceProvider)  COM\_INTERFACE\_ENTRY\_IID(IID\_IObjectSafety, IObjectSafety)  END\_COM\_MAP()  BEGIN\_PROP\_MAP(CATLPolyCtr00)  PROP\_DATA\_ENTRY("\_cx", m\_sizeExtent.cx, VT\_UI4)  PROP\_DATA\_ENTRY("\_cy", m\_sizeExtent.cy, VT\_UI4)  #ifndef \_WIN32\_WCE  PROP\_ENTRY\_TYPE("BorderColor", DISPID\_BORDERCOLOR, CLSID\_StockColorPage, VT\_UI4)  #endif  PROP\_ENTRY\_TYPE("BorderWidth", DISPID\_BORDERWIDTH, CLSID\_NULL, VT\_I4)  #ifndef \_WIN32\_WCE  PROP\_ENTRY\_TYPE("FillColor", DISPID\_FILLCOLOR, CLSID\_StockColorPage, VT\_UI4)  #endif  // Записи примеров  // PROP\_ENTRY\_TYPE("Имя свойства", dispid, clsid, vtType)  // PROP\_PAGE(CLSID\_StockColorPage)  END\_PROP\_MAP()  BEGIN\_CONNECTION\_POINT\_MAP(CATLPolyCtr00)  CONNECTION\_POINT\_ENTRY(IID\_IPropertyNotifySink)  CONNECTION\_POINT\_ENTRY(\_\_uuidof(\_IATLPolyCtr00Events))  END\_CONNECTION\_POINT\_MAP()  BEGIN\_MSG\_MAP(CATLPolyCtr00)  CHAIN\_MSG\_MAP(CComControl<CATLPolyCtr00>)  DEFAULT\_REFLECTION\_HANDLER()  MESSAGE\_HANDLER(WM\_LBUTTONDOWN, OnLButtonDown)  END\_MSG\_MAP()  // Прототипы обработчика:  // LRESULT MessageHandler(UINT uMsg, WPARAM wParam, LPARAM lParam, BOOL& bHandled);  // LRESULT CommandHandler(WORD wNotifyCode, WORD wID, HWND hWndCtl, BOOL& bHandled);  // LRESULT NotifyHandler(int idCtrl, LPNMHDR pnmh, BOOL& bHandled);  LRESULT OnLButtonDown(UINT uMsg, WPARAM wParam, LPARAM lParam, BOOL& bHandled);  // ISupportsErrorInfo  STDMETHOD(InterfaceSupportsErrorInfo)(REFIID riid)  {  static const IID\* const arr[] =  {  &IID\_IATLPolyCtr00,  };  for (int i=0; i<sizeof(arr)/sizeof(arr[0]); i++)  {  if (InlineIsEqualGUID(\*arr[i], riid))  return S\_OK;  }  return S\_FALSE;  }  // IViewObjectEx  DECLARE\_VIEW\_STATUS(VIEWSTATUS\_SOLIDBKGND | VIEWSTATUS\_OPAQUE)  // IATLPolyCtr00  POINT m\_arrPoint[10];  void CalcPoints(const RECT& rc);  public:  HRESULT OnDraw(ATL\_DRAWINFO& di)  {  RECT& rc = \*(RECT\*)di.prcBounds;  // Ограничивает область вырезания прямоугольником, указанным в di.prcBounds  HRGN hRgnOld = nullptr;  if (GetClipRgn(di.hdcDraw, hRgnOld) != 1)  hRgnOld = nullptr;  bool bSelectOldRgn = false;  HRGN hRgnNew = CreateRectRgn(rc.left, rc.top, rc.right, rc.bottom);  if (hRgnNew != nullptr)  {  bSelectOldRgn = (SelectClipRgn(di.hdcDraw, hRgnNew) != ERROR);  }  Rectangle(di.hdcDraw, rc.left, rc.top, rc.right, rc.bottom);  /\*--------------------------------------------------------------------\*/  HDC hdc = di.hdcDraw;  COLORREF colFore;  HBRUSH hOldBrush, hBrush;  HPEN hOldPen, hPen;  // Translate m\_colFore into a COLORREF type  OleTranslateColor(m\_clrFillColor, NULL, &colFore);  // Create and select the colors to draw the circle  hPen = (HPEN)GetStockObject(BLACK\_PEN);  hOldPen = (HPEN)SelectObject(hdc, hPen);  hBrush = (HBRUSH)GetStockObject(WHITE\_BRUSH);  hOldBrush = (HBRUSH)SelectObject(hdc, hBrush);  Ellipse(hdc, rc.left, rc.top, rc.right, rc.bottom);  // Create and select the brush that will be used to fill the polygon  hBrush = CreateSolidBrush(colFore);  SelectObject(hdc, hBrush);  CalcPoints(rc);  Polygon(hdc, &m\_arrPoint[0], m\_nSides);  // Select back the old pen and brush and delete the brush we created  SelectObject(hdc, hOldPen);  SelectObject(hdc, hOldBrush);  DeleteObject(hBrush);  /\*--------------------------------------------------------------------\*/  SetTextAlign(di.hdcDraw, TA\_CENTER|TA\_BASELINE);  //LPCTSTR pszText = \_T("ATLPolyCtr00");  LPCTSTR pszText = \_T("DKotsuba - 00321 - 27.12.22");  #ifndef \_WIN32\_WCE  TextOut(di.hdcDraw,  (rc.left + rc.right) / 2,  (rc.top + rc.bottom) / 2,  pszText,  lstrlen(pszText));  #else  ExtTextOut(di.hdcDraw,  (rc.left + rc.right) / 2,  (rc.top + rc.bottom) / 2,  ETO\_OPAQUE,  nullptr,  pszText,  ATL::lstrlen(pszText),  nullptr);  #endif  if (bSelectOldRgn)  SelectClipRgn(di.hdcDraw, hRgnOld);  DeleteObject(hRgnNew);  return S\_OK;  }  OLE\_COLOR m\_clrBorderColor;  void OnBorderColorChanged()  {  ATLTRACE(\_T("OnBorderColorChanged\n"));  }  LONG m\_nBorderWidth;  void OnBorderWidthChanged()  {  ATLTRACE(\_T("OnBorderWidthChanged\n"));  }  OLE\_COLOR m\_clrFillColor;  void OnFillColorChanged()  {  ATLTRACE(\_T("OnFillColorChanged\n"));  }  STDMETHOD(\_InternalQueryService)(REFGUID /\*guidService\*/, REFIID /\*riid\*/, void\*\* /\*ppvObject\*/)  {  return E\_NOTIMPL;  }  DECLARE\_PROTECT\_FINAL\_CONSTRUCT()  HRESULT FinalConstruct()  {  return S\_OK;  }  void FinalRelease()  {  }  SHORT m\_nSides;  STDMETHOD(get\_Sides)(SHORT\* pVal);  STDMETHOD(put\_Sides)(SHORT newVal);    };  OBJECT\_ENTRY\_AUTO(\_\_uuidof(ATLPolyCtr00), CATLPolyCtr00) |

|  |
| --- |
| <HTML>  <HEAD>  <meta http-equiv="Content-Type" content="text/html; charset=windows-1251"/>  <TITLE>Тестовая страница ATL для объекта ATLPolyCtr00</TITLE>  </HEAD>  <BODY>  <OBJECT ID="ATLPolyCtr00" CLASSID="CLSID:227d0672-6176-4dce-b654-2f0a47870912"></OBJECT>  <SCRIPT>  function PolyCtl:: ClickIn(x, y)  {  PolyCtl.Sides = PolyCtl.Sides + 1;  }  function PolyCtl:: ClickOut(x, y) {  PolyCtl.Sides = PolyCtl.Sides - 1;  }  </SCRIPT>  </BODY>  </HTML> |

**Выводы:**

В ходе выполнения лабораторной работы получен опыт создания элемента ActiveX управления, отображающего вписанный в окружность многоугольник, добавления собственных методов к элементу, вызова их напрямую и через возникновения события.