

АВТОМАТИЗАЦИЯ РАБОТЫ В NANOCAD С ПОМОЩЬЮ VISUAL BASIC FOR APPLICATIONS



В этой статье мы рассмотрим один из вариантов автоматизации работы проектировщиков в САПР nanoCAD, позволяющий в значительной степени использовать параметризацию построений и значительно сокращать сроки выполнения различных задач.

Многие проектировщики используют MS Excel для выполнения математических вычислений в табличной форме. Однако функционал программы этим не ограничивается. С помощью встроенного в продукты Microsoft Office языка программирования Visual Basic for Applications (VBA) можно взаимодействовать с объектной моделью nanoCAD и с другими продуктами на этой платформе. Ниже мы на простом и универсальном примере продемонстрируем такую возможность: создадим и настроим слой, нарисуем прямоугольник, поставим к нему размеры и вставим текст, содержащий площадь фигуры. В примере используются MS Excel 2010 и nanoCAD 5.0, но версия программных продуктов не имеет особого значения.

Для начала необходимо запустить Excel и подготовить данные для построения (рис. 1).

Данные в ячейках B1 и B2 записаны простым числом, но могут быть вычислены и с помощью формулы. Теперь необхо-

димо переключиться на встроенную в программу систему разработки и отладки программного кода. Нажимаем Alt + F11, появляется окно среды разработки (рис. 2).

	A	B	C
1	Высота прямоугольника:	1500	мм
2	Ширина прямоугольника:	1800	мм

Рис. 1

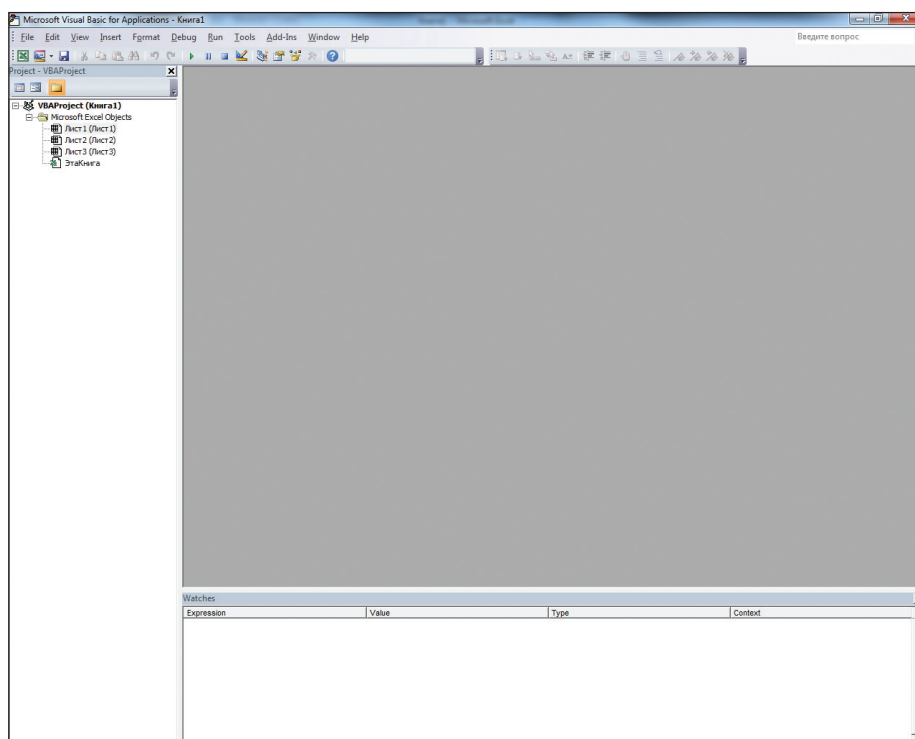


Рис. 2

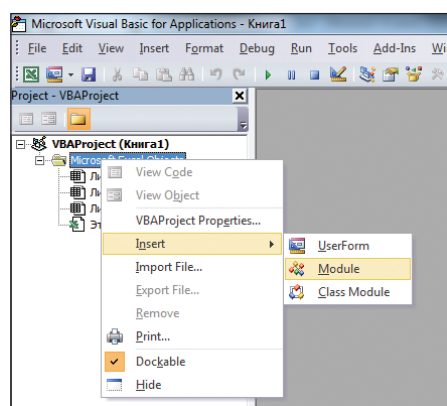


Рис. 3

Для удобства отладки кода необходимо подключить определенные библиотеки. В меню выбираем *Tools* → *References...*, после чего подключаем nanoCAD Type Library (NCAuto.dll) и OdaX Type Library (OdaX_csd.dll). По умолчанию обе библиотеки расположены в папке *C:\Program Files\Nanosoft\nanoCAD5.0\bin*. Теперь необходимо создать модуль. Правый клик в дереве проекта по папке *Microsoft Excel Objects*, в выпадающем списке выбираем *Insert* → *Module* (рис. 3). В появившемся окне подготовим основу будущей программы:

```
OptionExplicit 'требуется явное объявление
                'всех переменных в файле
Public app As nanoCAD.Application 'глобальная
                'переменная приложения
nanoCAD
Public ThisDrawing As nanoCAD.
                'Document 'глобальная переменная,
                'отвечающая за активный документ
nanoCAD
Sub my_drawing() 'функция построения
                'объектов
Set app = GetObject("", "nanoCAD.
                'Application") 'подключаемся к nanoCAD
app.Visible = True 'переключаем фокус
                'на окно nanoCAD
Set ThisDrawing = app.ActiveDocument
                'подключаемся к активному документу,
                'открытому в nanoCAD
```

После явного объявления переменных работать с ними становится значительно удобнее: достаточно ввести имя переменной, поставить точку — и программа подкажет все параметры и методы, относящиеся к объекту данного типа. Текст после апострофа — комментарии к коду. Объявим переменные с параметрами будущей фигуры и присвоим им значения:

```
Dim r_height As Double, r_width As
Double 'переменные, содержащие числа
двойной точности
r_height = Range("B1").Value 'высота
                'прямоугольника
r_width = Range("B2").Value 'ширина
                'прямоугольника
```

Создадим новый слой, назовем его толщиной и цветом:

```
Dim layer As AcadLayer 'переменная типа
                'слоя
Set layer = ThisDrawing.Layers.
                'Add("Автоматические построения")
                'создаем новый слой
layer.Color = 21 'бледно-красный цвет
layer.LineWeight = acLnWt050 'толщина
                'линии 0,50 мм
ThisDrawing.ActiveLayer = layer 'переключаем
                'текущий слой на созданный
```

Предложим пользователю выбрать координаты вставки объекта:

```
Dim insert_point() As Double 'массив
                'переменных, содержащих числа двойной
                'точности
insert_point = ThisDrawing.Utility.
                'GetPoint("0,0,0", "Укажите точку вставки
                'объекта") 'теперь в переменной
                'insert_point(0) содержится координата
                'X, указанная пользователем на экране,
                'в переменной insert_point(1) содержится
                'координата Y, а в переменной
                'insert_point(2) – координата Z
```

Построение прямоугольника, состоящего из четырех отрезков:

```
Dim x1 As Double, x2 As Double, y1 As
                'Double, y2 As Double
x1 = insert_point(0) 'координата по X
                'левой грани
x2 = x1 + r_width 'координата по X
                'правой грани
y1 = insert_point(1) 'координата по Y
                'нижней грани
y2 = y1 + r_height 'координата по Y
                'верхней грани
Dim pt1(2) As Double, pt2(2) As Double,
                'pt3(2) As Double, pt4(2) As Double
                'создаем четыре массива переменных,
                'отвечающих за координаты углов
                'прямоугольника в пространстве
pt1(0) = x1 'X нижнего левого угла
pt1(1) = y1 'Y нижнего левого угла
pt2(0) = x2 'X нижнего правого угла
pt2(1) = y1 'Y нижнего правого угла
pt3(0) = x2 'X верхнего правого угла
pt3(1) = y2 'Y верхнего правого угла
pt4(0) = x1 'X верхнего левого угла
```

```
pt4(1) = y2 'Y верхнего левого угла
Dim obj As AcadLine 'переменная типа
                '"линия"
Set obj = ThisDrawing.ModelSpace.
                'AddLine(pt1, pt2) 'построение нижней
                'грани
Set obj = ThisDrawing.ModelSpace.
                'AddLine(pt2, pt3) 'построение правой
                'грани
Set obj = ThisDrawing.ModelSpace.
                'AddLine(pt3, pt4) 'построение верхней
                'грани
Set obj = ThisDrawing.ModelSpace.
                'AddLine(pt4, pt1) 'построение нижней
                'грани
```

Построение размеров:

```
Dim pt5(2) As Double, pt6(2) As Double
pt5(0) = (x1 + x2) / 2 'X текста горизон-
                'тального размера
pt5(1) = y1 - 500 'Y текста горизонталь-
                'ного размера
pt6(0) = x2 + 500 'X текста вертикаль-
                'ного размера
pt6(1) = (y1 + y2) / 2 'Y текста верти-
                'кального размера
Dim dimrot As AcadDimRotated 'пере-
                'менная типа "повернутый размер"
Set dimrot = ThisDrawing.ModelSpace.
                'AddDimRotated(pt1, pt2, pt5, 0) 'строим
                'горизонтальный размер
Set dimrot = ThisDrawing.ModelSpace.
                'AddDimRotated(pt2, pt3, pt6, 3.1416 /
                '2) 'строим вертикальный размер
                'последний параметр в скобках опреде-
                'ляет угол поворота размера в радианах
```

Вычислим площадь фигуры и вставим MТекст в центр прямоугольника:

```
Dim pt7(2) As Double
pt7(0) = (x1 + x2) / 2 'X текста площади
pt7(1) = (y1 + y2) / 2 'Y текста площади
Dim txt As AcadMText 'переменная типа
                '"Мтекст"
Set txt = ThisDrawing.ModelSpace.
                'AddMText(pt7, 3000, CStr((x2 - x1) * (y2
                '- y1) / 1000000))
                'второй параметр определяет длину
                'Мтекста, третий параметр – текст Мтек-
                'ста
txt.AttachmentPoint =
                'acAttachmentPointMiddleCenter
                'изменение свойства Мтекста "Вырав-
                'нивание" на "Середина по центру"
```

Завершение базовой функции:

```
EndSub
```

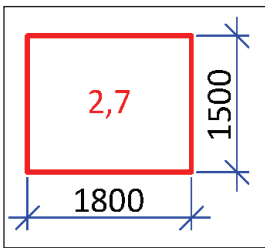


Рис. 4

Для запуска написанной программы достаточно нажать F5 (после этого не забудьте указать точку вставки в nanoCAD). Результат работы программы представлен на рис. 4.

Пример демонстрирует возможность создания базовых примитивов. Больше информации по объектной модели nanoCAD можно найти на просторах сети Интернет, в частности на сайтах <http://vbamodel.narod.ru> и www.alex160570.narod.ru/AcadVBA/vba01.htm.

Стоит отметить, что писать так много строк однотипного кода, чтобы построить одну линию, трудоемко и скучно, велик риск допустить ошибки. Для большего удобства можно создать пользовательские функции для построения отдельных примитивов. В качестве примера рассмотрим функцию построения отрезка. Функция получает в качестве параметров координаты X и Y начала и конца отрезка и строит его:

```
Sub my_line(x1 As Double, y1 As Double,
x2 As Double, y2 As Double)
Dim pt1(2) As Double, pt2(2) As Double
pt1(0) = x1 'X первой точки отрезка
pt1(1) = y1 'Y первой точки отрезка
pt2(0) = x2 'X второй точки отрезка
pt2(1) = y2 'Y второй точки отрезка
Dim obj As AcadLine
Set obj = ThisDrawing.ModelSpace.
AddLine(pt1, pt2) 'построение отрезка
из первой точки во вторую
EndSub
```

Дополнительные пользовательские функции необходимо расположить после базовой функции, то есть после строки EndSub, относящейся к базовой функции.

Теперь для построения прямоугольника можно использовать сокращенный код:

```
Dim x1 As Double, x2 As Double, y1 As Double, y2 As Double
x1 = insert_point(0) 'координата по X
левой грани
x2 = x1 + r_width 'координата по X правой грани
```

```
y1 = insert_point(1) 'координата по Y
нижней грани
y2 = y1 + r_height 'координата по Y
верхней грани
my_line x1, y1, x2, y1 'построение нижней грани
my_line x2, y1, x2, y2 'построение правой грани
my_line x2, y2, x1, y2 'построение верхней грани
my_line x1, y2, x1, y1 'построение нижней грани
```

Аналогичные пользовательские функции можно написать для любых задач, которые необходимо выполнить более одного раза. В то же время пользователю доступно множество операторов, таких как If... Then...Else, For...Next, Do...Loop и т.п. Для тех пользователей, которые только знакомятся с программированием, разобраться с возможностями языка не составит большого труда — для этого есть справка и многочисленные сайты с примерами решений задач.

Для удобства запуска созданной программы можно расположить кнопку прямо на листе Excel. Для этого на вкладке Разработчик нужно выбрать Вста-

вить → Элементы управления формы → Кнопка (рис. 5).

В появившемся окне следует выбрать нашу базовую функцию, после чего разместить кнопку на листе (рис. 6-7).

В один модуль можно добавить любое количество функций, создать для каждой функции кнопку и запускать их все из одной точки.

Иногда возникает необходимость имитировать ввод тех или иных команд в командную строку. Для этого можно использовать следующую конструкцию:

```
ThisDrawing.SendCommand "CIRCLE"
&vbCr& "100,100,0" &vbCr& "1000"
&vbCr
```

Данная команда построит окружность с центром в точке X:100, Y:100, Z:0 с радиусом 1000 единиц чертежа.

Для отладки создаваемых программ можно выводить данные в виде всплывающих окон или в командную строку nanoCAD. После выполнения следующего кода:

```
MsgBox "x1: " & x1
```

появится диалоговое окно со значением переменной x1 (рис. 8).

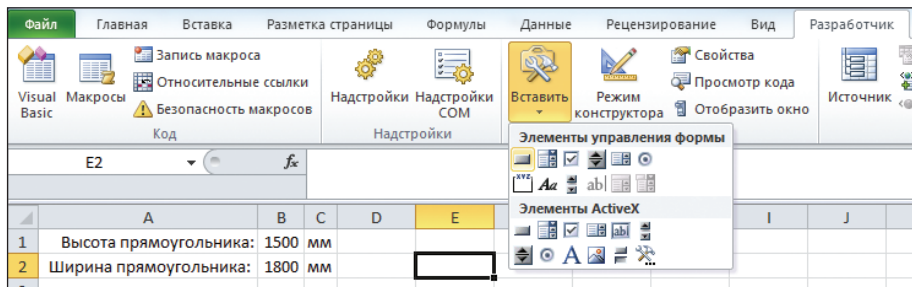


Рис. 5

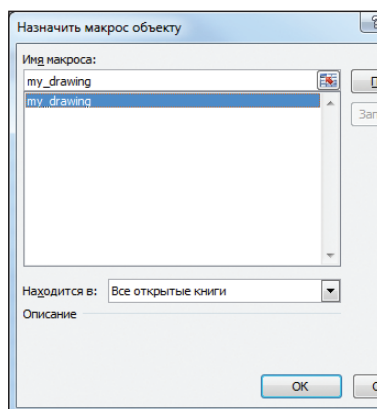


Рис. 6

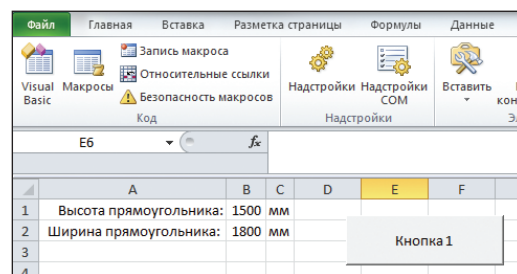


Рис. 7

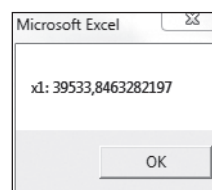


Рис. 8

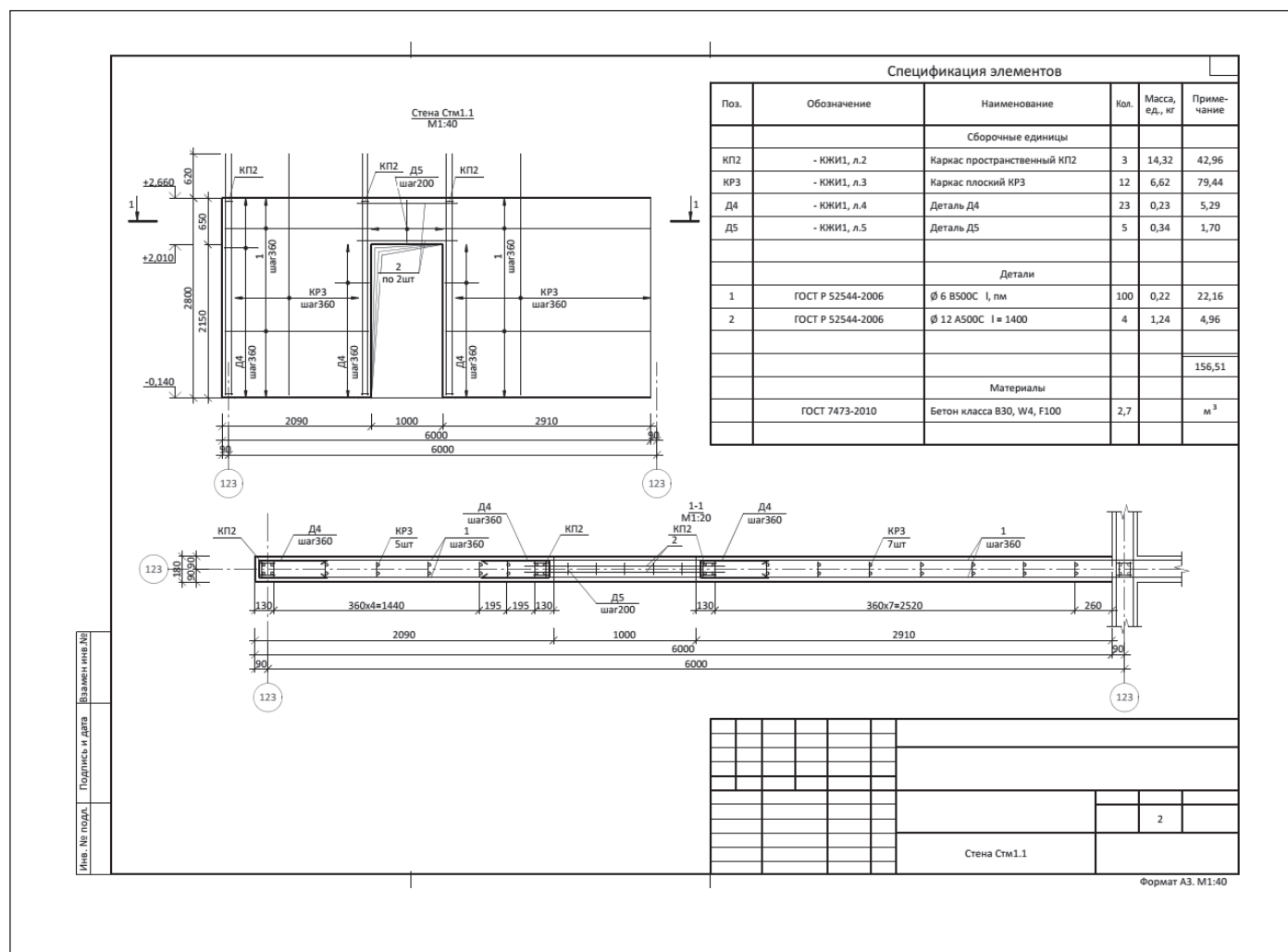


Рис. 9

Для вывода текста в командную строку paпoCAD можно использовать следующий код:

```
ThisDrawing.Utility.Prompt "x1: " & x1
&" y1: " & y1
```

Проектное бюро "Фордевинд" на протяжении уже 22 лет занимается проектированием железобетонных конструкций и все последние годы ведет активную работу по автоматизации рутинных задач. На сегодняшний день подготовлены и размещены в свободном доступе (<http://fordewind.org/wiki>) следующие программы:

- Последовательная нумерация (например, свай);
- Подсчет суммы длин выделенных отрезков, полилиний, дуг;

- Округление координат отрезков, полилиний и 3dface с заданной точностью;
- Обнуление координаты Z отрезков ("сплющивание" чертежа);
- Автоматическое создание листов для чертежей, оформленных в модели;
- Нумерация листов в правом верхнем углу;
- Нумерация листов в основной надписи;
- Сбор данных со спецификаций для подсчета ведомости расхода стали.

Нами разработана программа для автоматизации построения разверток монолитных железобетонных стен. После задания исходных данных (около 30 параметров) на выходе получаем практически готовый чертеж стены с учетом фактической геометрии, наличия проемов, примыкания других стен и прочего (рис. 9).

С помощью этой же программы можно, нажав одну кнопку, получить готовые

чертежи на все необходимые изделия (КЖИ). Такой подход к работе не только позволяет в десятки, если не в сотни раз сократить время на подготовку документации, но и полностью исключить необходимость ее проверки.

Мы описали способы построения новых примитивов. Но объектная модель paпoCAD позволяет редактировать и существующие в чертеже примитивы, в том числе объекты paпoCAD СПДС. К сожалению, уместить такой объем информации в одной статье не представляется возможным, так что этому будет посвящена следующая публикация.

Дмитрий Руденко,
ГИП проектного бюро "Фордевинд"
E-mail: dmitry.rudenko@fordewind.org