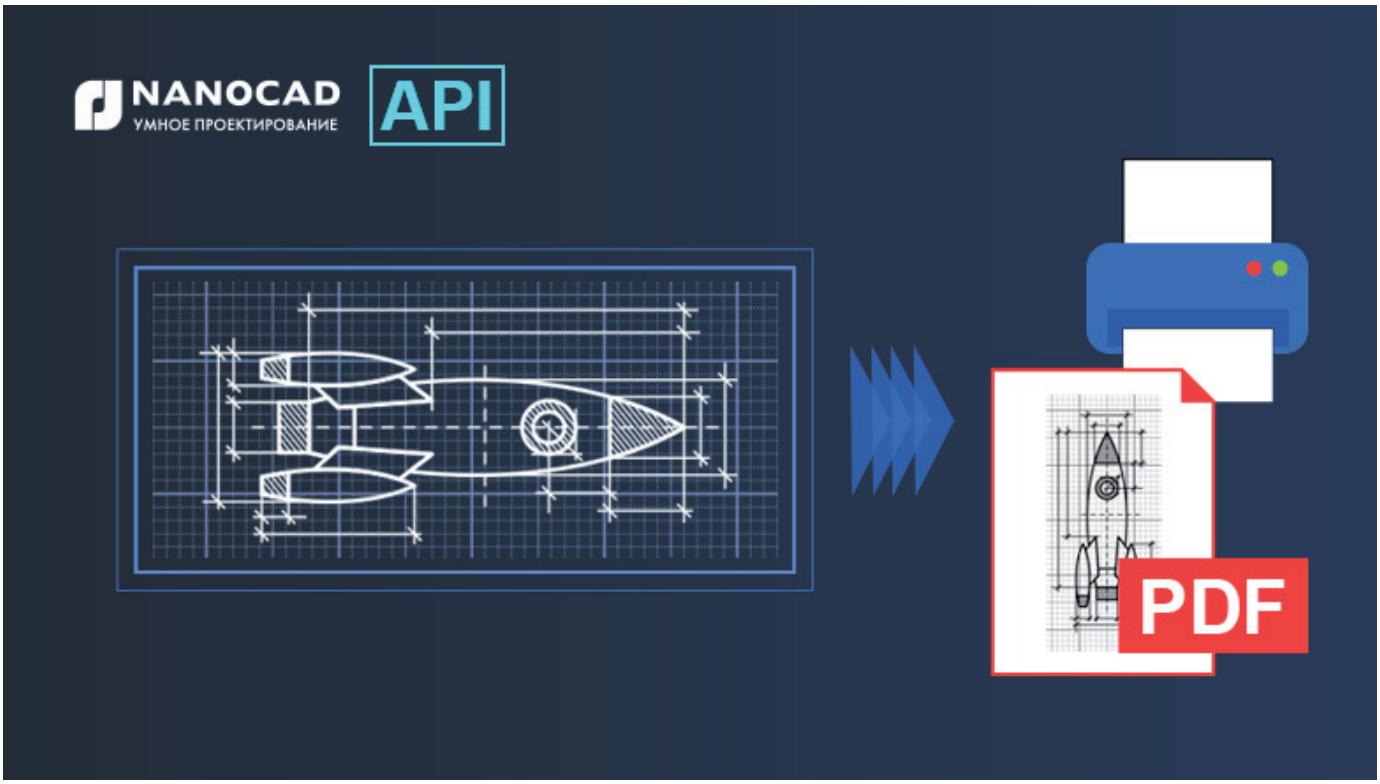


ПЕЧАТЬ ПРИ ПОМОЩИ API nanoCAD

Печать чертежей – ответственный и трудоемкий процесс. Чертеж, напечатанный в неверном масштабе или с неверной толщиной линий, может стать причиной производственных ошибок, так как будет неправильно прочитан. На настройку параметров печати и саму печать порой уходит немалая часть рабочего времени, поэтому многие проектировщики, знакомые с программированием, обращаются к программному интерфейсу САПР и получают хорошую возможность уменьшить влияние "человеческого фактора", а также сократить время, которое приходится тратить на работу с чертежами.

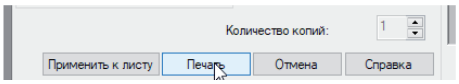
Этой статьей мы запускаем цикл материалов, посвященный API печати nanoCAD, в котором предложим ответы на традиционные вопросы начинающих САПР-программистов. Начнем с того, что разберемся, как программно отправить чертеж на печать.

Для отправки чертежа на печать создадим приложение на платформе .NET (C#).

Для создания приложения будет использован .NET API nanoCAD, а для печати – ActiveX. Фактически это три библиотеки: hostmgd.dll, hostdbmgd.dll (.NET API) и ncauto.dll (ActiveX). Библиотеки доступны в директории установки nanoCAD.

В качестве среды разработки подойдет Visual Studio версии, поддерживающей .NET Framework 4.0 и выше.

Для отправки чертежа на печать через пользовательский интерфейс nanoCAD необходимо вызвать диалоговое окно *Печать* через меню или командой *PLOT* в консоли, настроить параметры печати и нажать кнопку *Печать*.



Для отправки чертежа nanoCAD на печать программно используются методы `nanoCAD.InanoCADPlot.PlotToDevice()` в случае, когда документ нужно напечатать на бумаге, и `nanoCAD.InanoCADPlot.PlotToFile()`, когда документ требуется напечатать в файл.

Рассмотрим последовательность действий, которую нужно выполнить, чтобы отправить документ на печать через API nanoCAD:

Шаг 1. Создаем в Visual Studio проект типа "Библиотека классов (.NET Standard)". Подключаем к нему библиотеки hostmgd.dll, hostdbmgd.dll и ncauto.dll. Здесь и далее подразумевается, что вы используете Visual Studio 2019.

Шаг 2. Создаем класс, в котором будет находиться метод, отправляющий документ nanoCAD на печать.

```
public partial class Commands
{
    [Teigha.Runtime.CommandMethod("PlotDocument")]
    public void PrintDocument()
    {
    }
}
```

В последующих шагах содержится инструкция по реализации метода PrintDocument().

Шаг 3. Получаем доступ к документу nanoCAD и его настройкам:

```
// Получение ссылки на активный документ
HostMgd.ApplicationServices.Document doc =
    HostMgd.ApplicationServices.Application.DocumentManager.
    MdiActiveDocument;
nanoCAD.Document comDoc = doc.AcadDocument as nanoCAD.Document;
```

Здесь происходит симбиоз библиотек, о которых упоминалось выше: с помощью библиотеки hostmgd.dll получаем доступ к активному документу nanoCAD (переменная doc); далее, чтобы получить доступ к тем настройкам, которые нам нужны, запрашиваем из переменной doc COM-документ nanoCAD.
В дальнейшем настройки печати мы будем получать, обращаясь к свойствам и методам переменной comDoc.

Шаг 4. При помощи свойства *Plot* документа nanoCAD получаем COM-объект *Plot*, содержащий методы печати.

```
// Получение ссылки на свойство Plot активного документа
nanoCAD.Plot plot = (nanoCAD.Plot)comDoc.Plot;
```

Шаг 5. Выбираем принтер и меняем настройки листов для печати. В предыдущих шагах были проведены подготовительные действия, теперь мы переходим к настройке параметров печати.
За принтер отвечает свойство ConfigName листа документа. Каждому листу можно назначить свой принтер. Для доступа к листам документа используем коллекцию Layouts документа nanoCAD. Перебирая коллекцию, можно изменить настройки каждого листа (назначить всем листам принтер и изменить пользовательские настройки принтера, если был выбран встроенный принтер nanoCAD):

```
// Назначение принтера всем листам документа
foreach (OdaX.IAcadLayout layout in comDoc.Layouts)
{
    layout.ConfigName = "Встроенный pdf-принтер";
    nanoCAD.InanoCADPlotCustomParams customPlotSettings =
    plot.CustomPlotSettings[layout];

    // Изменяем путь сохранения напечатанного документа
    customPlotSettings.UseDWGPath = false;
    customPlotSettings.FileFolder = "C:\\PlotToDeviceExample";

    // Изменяем шаблон имени файла, в который будет напечатан документ
    customPlotSettings.UsePredefinedNames = true;
    customPlotSettings.FileMask = "<DN>_<LN>";

    // Передаем измененные пользовательские настройки в лист
    plot.CustomPlotSettings[layout] = customPlotSettings;
}
```

Здесь свойство UseDWGPath указывает, сохранять напечатанный документ в той же директории, где расположен документ (значение true), или использовать для сохранения директорию, указанную в свойстве FileFolder (значение false). При указании директории необходимо убедиться, что она существует, иначе произойдет ошибка печати.
Свойство UsePredefinedNames указывает, использовать шаблон имени файла, определенный в свойстве FileMask (значение true), или открывать диалоговое окно сохранения файла (значение false).
Для шаблонов имен файлов существует набор символов, которые будут интерпретироваться как автозаполняемые поля. В эти поля будут подставляться соответствующие свойства документа или значения:

- <DN> – имя документа;
- <LN> – имя листа;
- <UN> – имя пользователя;
- <T> – время;
- <D> – дата;
- <C1> – счетчик 1. Счетчик добавляет индекс к названию файлов, начиная с 1;
- <C2> – счетчик 01. Счетчик добавляет индекс к названию файлов, начиная с 01;
- <C3> – счетчик 001. Счетчик добавляет индекс к названию файлов, начиная с 001;
- <C4> – счетчик 0001. Счетчик добавляет индекс к названию файлов, начиная с 0001;
- <C5> – счетчик 00001. Счетчик добавляет индекс к названию файла, начиная с 00001;
- <C6> – счетчик 000001. Счетчик добавляет индекс к названию файла, начиная с 000001.

Шаг 6. Определяем перечень листов документа, которые будут отправлены на печать для метода SetLayoutsToPlot(). Если не использовать этот метод, на печать будет отправлен активный лист:

```
// Создание массива имен листов, предназначенных для печати

string[] layoutsToPlot = new string[] {
comDoc.Layouts.Item(1).Name, comDoc.Layouts.Item(2).Name };

// Вызов метода nanoCAD.InanoCADPlot.SetLayoutsToPlot()
plot.SetLayoutsToPlot(layoutsToPlot);
```

Листы, назначенные для печати этим методом, хранятся в памяти до первого вызова метода печати. После этого необходимо заново вызывать метод SetLayoutsToPlot() перед следующей отправкой документа на печать.

Шаг 7. Отправляем документ на печать. В случае встроенных принтеров, в частности, pdf-принтера, для печати можно использовать как метод PlotToDevice(), так и PlotToFile(). Каждый из методов сработает по-своему при выборе директории сохранения напечатанного файла и его имени. Если использовать метод PlotToDevice(), будут применены пользовательские настрой-

ки принтера (customPlotSettings). Если использовать метод PlotToFile(), шаблон имени файла и директория сохранения будут взяты из его параметра при вызове:

```
// Печать назначенных листов
// В этом случае будет создано два pdf-файла в директории
C:\\PlotToDeviceExample
// с именами по шаблону "Название документа_Название листа".pdf
plot.PlotToDevice();

// Для наглядности также печать в файл
// После выполнения метода PlotToDevice() назначенные для печати
листы
// будут сброшены. Методом PlotToFile() будет напечатан активный лист
документа
// и назван по шаблону "myplot_Имя документа_Имя листа".pdf
plot.PlotToFile("C:\\PlotToFileExample\\myplot_<DN>_<LN>");
```

Метод nanoCAD.InanoCADPlot.PlotToDevice() при печати на pdf-принтере использует пользовательские настройки листов, которые назначены для печати, т.е. путь сохранения *C:\\PlotToDeviceExample*, а в качестве шаблона имени будет применен шаблон "Имя документа_Имя листа.pdf" ("<DN>_<LN>").

Метод nanoCAD.InanoCADPlot.PlotToFile() при печати использует в качестве пути сохранения и имени файла данные своего параметра plotFile, даже если в пользовательских настройках указаны путь и шаблон имени файла.

Параметр plotFile может иметь следующие значения:

1. Пустая строка: plot.PlotToFile(""). Файл будет сохранен в папке документа nanoCAD, который отправляется на печать и назван так же, как сам документ, но с расширением, соответствующим принтеру, на котором печатался документ (в нашем случае .pdf).
2. Только имя файла: plot.PlotToFile("myplot_<DN>_<LN>"). Файл будет сохранен с заданным именем в папку документа, который отправляется на печать.
3. Только адрес директории сохранения файла: plot.PlotToFile("C:\\PlotToFileExample"). Файл будет сохранен в указанную директорию и назван так же, как документ-исходник – с расширением, соответствующим принтеру, на котором он печатался.
4. Адрес директории и шаблон имени файла: plot.PlotToFile("C:\\PlotToFileExample\\myplot_<DN>_<LN>"). В этом случае документ будет сохранен в указанной директории и с указанным именем файла.

При указании в параметре plotFile адреса директории, необходимо убедиться, что она существует.

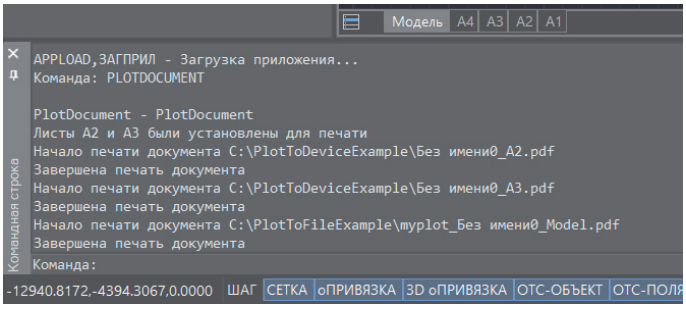
Далее все рассмотренные шаги приведены в виде команды PlotDocument, которая устанавливает всем листам документа встроенный pdf-принтер. Меняет их пользовательские настройки. Методом nanoCAD.InanoCADPlot.SetLayoutsToPlot() устанавливает два листа документа для печати и печатает эти листы методом nanoCAD.InanoCADPlot.PlotToDevice(), затем для наглядности вызывается метод nanoCAD.InanoCADPlot.PlotToFile().

```
public partial class Commands
{
    [Teigha.Runtime.CommandMethod("PlotDocument")]
    public void PrintDocument()
    {
        // Получение ссылки на активный документ
        HostMgd.ApplicationServices.Document doc =
        HostMgd.ApplicationServices.Application.DocumentManager.
        MdiActiveDocument;
        nanoCAD.Document comDoc = doc.AcadDocument as
        nanoCAD.Document;

        // Получение ссылки на редактор активного документа
        HostMgd.EditorInput.Editor ed = doc.Editor;

        // Получение ссылки на активный лист документа
        OdaX.AcadLayout activeLayout = comDoc.ActiveLayout;

        // Получение ссылки на свойство Plot активного документа
        nanoCAD.Plot plot = (nanoCAD.Plot)comDoc.Plot;

        // Назначение принтера всем листам документа.
        foreach (OdaX.IAcadLayout layout in comDoc.Layouts)
        {

            layout.ConfigName = "Встроенный pdf-принтер";
            nanoCAD.InanoCADPlotCustomParams
            customPlotSettings =
            plot.CustomPlotSettings[layout];

            // Изменяем путь сохранения напечатанного в файл
            документа
            customPlotSettings.UseDWGPath = false;
            customPlotSettings.FileFolder =
            "C:\\PlotToDeviceExample";

            // Изменяем шаблон имени файла, в который будет
            напечатан документ
            customPlotSettings.UsePredefinedNames = true;
            customPlotSettings.FileMask = "<DN>_<LN>";

            // Передаем измененные пользовательские настройки
            в лист
            plot.CustomPlotSettings[layout] = customPlotSettings;
        }

        // Назначение активному листу области печати "Границы",
        если это пространство модели,
        // или "Лист", если активный лист не является простран-
        ством модели
        if    (activeLayout.ModelType)
              comDoc.ActiveLayout.PlotType = OdaX.AcPlotType.
              acExtents;
        else comDoc.ActiveLayout.PlotType = OdaX.AcPlotType.
              acLayout;

        // Включение режима "Вписать"
        comDoc.ActiveLayout.StandardScale = OdaX.AcPlotScale.
        acScaleToFit;

        // Создание массива имен листов, предназначенных для
        печати
        string[] layoutsToPlot = new string[] { comDoc.Layouts.
        Item(1).Name, comDoc.Layouts.Item(2).Name };

        // Вызов метода nanoCAD.InanoCADPlot.SetLayoutsToPlot()
        plot.SetLayoutsToPlot(layoutsToPlot);
```

```
            // Сообщение пользователю
            ed.WriteMessage("Листы {0} и {1} были установлены для печати", layoutsToPlot[0],layoutsToPlot[1]);

            // Печать назначенных листов
            // В этом случае будет создано два pdf-файла в директории C:\\PlotToDeviceExample
            // с именами по шаблону "Название документа_Название листа".pdf
            plot.PlotToDevice();
            // Для наглядности также печать в файл
            // После выполнения метода PlotToDevice() назначенные для печати листы
            // будут сброшены. Методом PlotToFile() будет напечатан активный лист документа
            // и назван по шаблону "myplot_Имя документа_Имя листа".pdf
            plot.PlotToFile("C:\\PlotToFileExample\\myplot_<DN>_<LN>");
        }
}
```

Шаг 8. Компилируем наше приложение и загружаем полученный dll-файл в nanoCAD.

Если открыть новый документ nanoCAD и запустить в нем команду PlotDocument, то в результате будет создано три файла. Результат работы команды приведен на рисунке.



Сообщения команды *PlotDocument* в консоли nanoCAD

Было напечатано два листа, назначенных методом nanoCAD. InanoCADPlot.SetLayoutsToPlot() для печати, и при повторном вызове метода печати напечатан один активный лист.

Итак, мы разобрались с применением методов PlotToDevice(), PlotToFile() и SetLayoutsToPlot(), попутно затронув методы настройки параметров печати через API nanoCAD. В следующих статьях более подробно рассмотрим настройку параметров печати листов документа nanoCAD через API.

Светлана Мирончик,
Клуб разработчиков nanoCAD
ООО "Нанософт разработка"

Компания "Нанософт разработка" объявила о выходе программного продукта nanoCAD BIM Вентиляция

Компания "Нанософт разработка" выпускает новое приложение nanoCAD BIM Вентиляция. Продукт предназначен для автоматизированного проектирования систем вентиляции и кондиционирования зданий и сооружений с применением технологии информационного моделирования.

nanoCAD BIM Вентиляция базируется на новейшей программной технологии, которая получила название EVOS. Это обеспечивает приложению ряд неоспоримых преимуществ:

- многопользовательский режим;
- создание 3D-моделей любой степени детализации;
- мультивидовое представление модели;
- возможность типового проектирования.

Концепция программы проста: инженер создает трехмерную информационную модель с помощью интеллектуальных инструментов и готовых объектов из базы данных. Все элементы модели обладают поведением и логикой взаимодействия друг с другом, которая определяется их типом, набором свойств и даже расчетами формул. При этом документация (чертежи и спецификации) автоматически формируется на основе данных информационной модели и заточена под оформление по отечественным стандартам проектирования.

Чтобы обеспечить принятие обоснованных проектных решений в nanoCAD BIM Вентиляция реализован аэродинамический расчет с принудительным побуждением движения воздуха.

По результатам работы в программе пользователь получает:

- трехмерную информационную модель системы вентиляции здания;
- планы расположения оборудования и прокладки трасс воздуховодов;
- спецификацию оборудования, изделий и материалов.

Более подробная информация о программном продукте nanoCAD BIM Вентиляция представлена на его официальной странице (*www.nanodev.ru/products/bimventilation*).

Бесплатная оценочная версия и преимущества пользователей

Демонстрационную версию nanoCAD BIM Вентиляция можно скачать на сайте *nanodev.ru* или получить в офисах официальных дилеров компании "Нанософт разработка". В течение пробного 30-дневного периода доступны все возможности программного продукта.

Все коммерческие пользователи инженерных продуктов компании "Нанософт разработка" получают бесплатную полугодовую лицензию nanoCAD BIM Вентиляция.

Новым пользователям инженерных продуктов при покупке постоянной лицензии или действующим пользователям при продлении подписки на такую лицензию бесплатно предоставляется годовая лицензия nanoCAD BIM Вентиляция.

Купить nanoCAD BIM Вентиляция

nanoCAD BIM Вентиляция, как и другие программные продукты компании "Нанософт разработка", будет распространяться по двум схемам: пользователям предлагаются годовые и постоянные лицензии.

Стоимость годовой лицензии – 31 100 руб. (приобретение лицензии на Платформу nanoCAD 21 обязательно). Стоимость постоянной лицензии – 97 100 руб. (приобретение лицензии на Платформу nanoCAD 21 обязательно). Старт продаж ожидается в первом квартале 2022 года.

Подобрать конфигурацию программного продукта, исходя из типа лицензирования и целей бизнеса, можно на сайте *nanodev.ru* или обратившись к авторизованному партнеру в вашем регионе.