Nexamas

Nexamas.UI Documentation

A bilingual evaluation and application-development guide for the Community Preview and the Professional product boundary.

Community Preview contract

1.0.0-preview.1

This documentation describes the approved 1.0.0-preview.1 Preview contract. Public package download remains disabled until Website Phase 40.9.

Source: unified Website Phase 40.4 content contract

Updated: 2026-07-28

Contents

01 Overview

02 System requirements

03 NuGet and ZIP installation

04 Quick start

05 First application checklist

06 Application shell

07 Themes and surfaces

08 Controls overview

09 Layout and sizing

10 Data surfaces

11 Showcase guide

12 Community license

13 Community vs Professional

14 Versioning

15 Troubleshooting

16 Release notes

17 Support policy

01 Overview

Nexamas.UI is a proprietary Windows Forms UI SDK with one MAS public API model across .NET Framework 4.8 and .NET 10 for Windows.

The product, NuGet package, assembly, and namespace root use the identity Nexamas.UI. Public consumer types use the stable MAS* family, including MASApplication, MASApplicationWindow, MASButton, MASDataGrid, and related types.

- Application and window composition through public application facades.
- Typed control factories, layout builders, themes, materials, localization and RTL foundations.
- Data presentation and workflow surfaces, plus measured quality and render-verification infrastructure.
- One canonical product model on both supported target frameworks.

Product boundary

Nexamas.UI is a Windows desktop SDK. It does not claim macOS, Linux, mobile, browser, general PDF export, or operating-system printer integration in this Preview.

02 System requirements

The documented proof lane is Windows desktop, Windows Forms, Release|x64, with two supported target frameworks.

Requirement	Supported contract
Operating system	Windows desktop
UI host	Windows Forms
Target frameworks	net48 and net10.0-windows10.0.19041.0
Documented release lane	Release|x64
Consumer languages	C# and Visual Basic
Modern SDK	.NET 10 SDK
Classic build	Visual Studio/MSBuild with .NET Framework 4.8 targeting support

Release envelope

Do not present x86 or AnyCPU as certified commercial release targets unless a later proof explicitly adds them.

03 NuGet and ZIP installation

The Preview uses package identity Nexamas.UI and version 1.0.0-preview.1. Public delivery is intentionally gated until Phase 40.9.

Preview publication gate

The commands below describe the approved package contract. They will become publicly actionable only after the release artifacts, license, legal pages, hashes, and final verification gates pass.

.NET 10 project file xml

```xml
<Project Sdk="Microsoft.NET.Sdk">
  <PropertyGroup>
    <OutputType>WinExe</OutputType>
    <TargetFramework>net10.0-windows10.0.19041.0</TargetFramework>
    <UseWindowsForms>true</UseWindowsForms>
    <EnableWindowsTargeting>true</EnableWindowsTargeting>
    <PlatformTarget>x64</PlatformTarget>
    <Prefer32Bit>false</Prefer32Bit>
  </PropertyGroup>
  <ItemGroup>
    <PackageReference Include="Nexamas.UI" Version="1.0.0-preview.1" />
  </ItemGroup>
</Project>
```

Local or private feed configuration xml

```xml
<?xml version="1.0" encoding="utf-8"?>
<configuration>
  <packageSources>
    <clear />
    <add key="NexamasPreview" value="PATH_OR_FEED_PROVIDED_BY_NEXAMAS" />
    <add key="NuGetOfficial" value="https://api.nuget.org/v3/index.json" />
  </packageSources>
</configuration>
```

- ZIP delivery must contain the exact approved package, license, release notes, and SHA-256 manifest.
- Do not copy assemblies manually between target-framework folders when NuGet is available.
- Community may be cached in a private internal company repository, but may not be mirrored to a public feed.

04 Quick start

Create a Windows Forms application, reference Nexamas.UI, then attach or create a MAS application window.

Visual Basic first window vb

```vb
Option Strict On
Option Explicit On

Imports System
Imports System.Windows.Forms
Imports Nexamas.UI.Application
Imports Nexamas.UI.Controls
Imports Nexamas.UI.Layout

Public NotInheritable Class MainForm
    Inherits Form

    Private ReadOnly _window As MASApplicationWindow

    Public Sub New()
        Text = "Nexamas UI Quickstart"
        Width = 1000
        Height = 700
        StartPosition = FormStartPosition.CenterScreen

        _window = MASApplication.AttachWindow(
            owner:=Me,
            configure:=Sub(window As MASApplicationWindow)
                Dim title As MASTitle = window.Controls.AddTitle("Nexamas UI")
                Dim message As MASLabel = window.Controls.AddLabel("One MAS API on net48 and .NET 10.")
                Dim action As MASButton = window.Controls.AddButton("Ready")

                AddHandler action.Click,
                    Sub(sender As Object, e As EventArgs)
                        window.Services.Toasts.Success("The package consumer is alive.", "Quickstart")
                    End Sub

                window.Controls.LayoutPage(
                    Sub(page As MASApplicationLayoutPageBuilder)
                        page.Padding(MASLayoutSpacing.Spacious)
                        page.Spacing(MASLayoutSpacing.Medium)
                        page.FullWidth(title, MASSize.FillWidth)
                        page.FullWidth(message, MASSize.FillWidth)
                        page.ActionGroup(
                            Sub(actions As MASApplicationActionGroupLayoutBuilder)
                                actions.AlignCenter().EqualItemWidth().Add(action, MASSize.Default)
                            End Sub)
                    End Sub)
            End Sub)
    End Sub

    Protected Overrides Sub OnFormClosed(e As FormClosedEventArgs)
        If _window IsNot Nothing Then _window.Dispose()
        MyBase.OnFormClosed(e)
    End Sub
End Class
```

C# first window

csharp

```
using System;
using System.Windows.Forms;
using Nexamas.UI.Application;
using Nexamas.UI.Controls;
using Nexamas.UI.Layout;

internal sealed class MainForm : Form
{
    private readonly MASApplication _application;
    private readonly MASApplicationWindow _window;

    internal MainForm()
    {
        Text = "Nexamas UI Quickstart";
        Width = 1000;
        Height = 700;

        _application = MASApplication.Create();
        _window = _application.CreateWindow(this);

        MASTitle title = _window.Controls.AddTitle("Nexamas UI");
        MASLabel message = _window.Controls.AddLabel("One MAS API on net48 and .NET 10.");
        MASButton action = _window.Controls.AddButton("Ready");

        _window.Controls.LayoutPage(page =>
        {
            page.Padding(MASLayoutSpacing.Spacious);
            page.Spacing(MASLayoutSpacing.Medium);
            page.FullWidth(title, MASSize.FillWidth);
            page.FullWidth(message, MASSize.FillWidth);
            page.ActionGroup(actions => actions.AlignCenter().EqualItemWidth().Add(action,
MASSize.Default));
        });
    }

    protected override void OnFormClosed(FormClosedEventArgs e)
    {
        _window?.Dispose();
        _application?.Dispose();
        base.OnFormClosed(e);
    }
}
```

Lifetime rule

Dispose each MASApplicationWindow. When using MASApplication.Create(), dispose the application after all owned windows are closed.

05 First application checklist

Use a small vertical slice before moving a production screen: one window, a title, a label, one action, and a simple layout.

- Choose net48 or net10.0-windows10.0.19041.0 and x64.
- Reference the exact Preview version supplied by the approved feed or ZIP.
- Create a normal WinForms Form and attach a MASApplicationWindow.
- Add controls through window.Controls typed factories.
- Compose the page through LayoutPage and MASSize rather than hard-coded pixel placement.
- Dispose the window and application facade correctly.
- Test keyboard focus, DPI, resize behavior, localization direction, and representative data.

Evaluation advice

Evaluate Nexamas.UI by migrating one representative workflow, not by judging an isolated button.

06 Application shell

The public application layer is intended to coordinate windows, navigation, layout, commands, overlays, feedback, and application-owned services.

Use MASApplication.AttachWindow(...) for a small one-form application. Use MASApplication.Create() and CreateWindow(...) when one application instance owns multiple windows.

Application composition sketch vb

```vb
Dim application As MASApplication = MASApplication.Create()
Dim window As MASApplicationWindow = application.CreateWindow(owner:=Me)

Dim navigation = window.Controls.AddNavigationRail()
Dim title = window.Controls.AddTitle("Operations")
Dim content = window.Controls.AddPanel()

window.Controls.LayoutPage(
    Sub(page)
        page.Padding(MASLayoutSpacing.Spacious)
        page.Spacing(MASLayoutSpacing.Medium)
        page.FullWidth(title, MASSize.FillWidth)
        page.FullWidth(content, MASSize.Fill)
    End Sub)
```

- Keep window lifetime explicit.
- Use application-owned service gateways instead of global singletons.
- Treat Showcase composition as an example surface, not as the authority for public claims.
- Do not call Friend/internal systems directly from customer applications.

07 Themes and surfaces

Nexamas.UI provides public theme and surface gateways while deeper design, validation, and studio infrastructure remains internal unless separately exposed.

Public theme boundary vb

```vb
Dim app As MASApplication = MASApplication.Create()
Dim window As MASApplicationWindow = app.CreateWindow(owner:=Me)

' Use the public application/window theme gateway.
' Keep theme selection at the application boundary rather than styling each control independently.
Dim themeGateway = window.Theme
```

- Select themes at the application or window boundary.
- Use shared spacing, size, material, and state recipes for visual consistency.
- Verify contrast and focus states in both light and dark themes.
- Theme Studio and Surface Designer are not claimed as public SDK APIs in this Preview.

08 Controls overview

The approved inventory contains 80 classified controls across Community, shared Community + Pro routes, and Professional.

Community	Community + Pro	Professional
51	**9**	**20**
foundation controls	shared controls	advanced controls

- Application and navigation: titles, command surfaces, navigation, panels, overlays, and feedback.
- Input and selection: buttons, text input, selectors, menus, lists, trees, editors, and focus behavior.
- Data and workflow: grids, views, master-detail, planning, dashboard, filter, chart, and file workflows.
- Advanced Professional controls are sold by edition policy only when the source, public API, and proof are present.

Inventory is not a readiness shortcut

A classified control is not automatically a promise that every scenario, performance envelope, or designer route is production-ready. Use the release notes and known-issues evidence for the exact build.

09 Layout and sizing

Use public layout builders, MASSize, and MASLayoutSpacing to express intent instead of positioning every element manually.

Page layout example vb

```
window.Controls.LayoutPage(
    Sub(page As MASApplicationLayoutPageBuilder)
        page.Padding(MASLayoutSpacing.Spacious)
        page.Spacing(MASLayoutSpacing.Medium)
        page.FullWidth(header, MASSize.FillWidth)
        page.FullWidth(dataGrid, MASSize.Fill)
        page.ActionGroup(
            Sub(actions)
                actions.AlignEnd().EqualItemWidth()
                actions.Add(cancelButton, MASSize.Default)
                actions.Add(saveButton, MASSize.Default)
            End Sub)
    End Sub)
```

- Use FillWidth or Fill for surfaces that should resize with the window.
- Use action groups for aligned command rows.
- Test minimum sizes, DPI scaling, long localized text, and RTL direction.
- Avoid depending on internal responsive or virtualization classes as public APIs.

10 Data surfaces

DataGrid and DataView have documented public consumer surfaces. Additional professional workflow controls are governed by the approved edition inventory.

Data surface entry points vb

```vb
Dim grid As MASDataGrid = window.Controls.AddDataGrid()
Dim view As MASDataView = window.Controls.AddDataView()

' Bind through the documented public consumer surface for your release.
' Validate large-data behavior with representative customer data before shipping.
```

- Start with representative row counts and real column shapes.
- Verify selection, editing, keyboard navigation, sorting, filtering, and resize behavior required by your workflow.
- Treat internal virtualization engines as implementation detail unless a public gateway explicitly exposes a contract.
- Professional-only controls must not be copied, reflected, or invoked from Community through internal APIs.

11 Showcase guide

The Showcase demonstrates verified composition and customer journeys. It does not redefine package, edition, licensing, or API truth.

- Use screenshots and sample source that consume public Nexamas.UI APIs only.
- The example application may be public because it consumes the SDK; the proprietary SDK source remains private.
- Professional demonstrations must run through the approved remote or licensed boundary and must not send Nexamas.UI.Pro binaries to the browser.
- Validate the sample on Windows, Windows Forms, x64, and the documented target framework before attaching it to a release.

Phase boundary

The runnable sample, final screenshot set, and browser evidence are completed in Website Phase 40.5.

12 Community license

Community is free for personal, educational, non-commercial, commercial, SaaS, internal-business, paid-application, and subscription-product use.

- No runtime fees, license key, seat limit, user limit, or project limit.
- The unmodified runtime DLL may be included inside a finished application that needs it.
- Original applications, templates, plug-ins, add-ons, controls, and libraries may be sold when they add independent value.
- A matching unmodified package may be cached in one organization’s private repository.
- Do not sell Nexamas.UI by itself, rename it, repackage it, or publish a public mirror/feed.
- Do not remove ownership notices or create a competing derivative SDK.
- Reverse engineering and internal modification are prohibited except where applicable law requires an exception.
- Attribution is not mandatory. A friendly recommendation is appreciated when Nexamas.UI helps your product.

Legal review required

This page summarizes the approved product policy. The final public license text must pass professional legal review before Phase 40.9 publication.

13 Community vs Professional

Community is intended to be useful without artificial runtime limits. Professional adds advanced controls, licensed development rights, direct support targets, and private delivery.

Capability	Community	Professional
Application/window model, themes, DPI, input, layout, localization and RTL	Included	Included
Community foundation controls	51 controls	All Community controls
Shared controls with advanced Professional routes	Useful base behavior on 9 controls	Base plus licensed advanced routes
Professional controls	Not included	20 controls across analytics, workflow, workspace, editors and navigation
Target frameworks	.NET Framework 4.8 and .NET 10 for Windows	Same targets
Consumer languages	C# and Visual Basic	C# and Visual Basic
Commercial, SaaS and internal-business use	Allowed	Allowed
Runtime royalties or end-user seats	None	None
Continuous internet requirement	None	None for licensed use
Samples and adoption assets	Basic documentation, recipes and beginner samples	Advanced product-control samples and commercial templates
Render verification and team quality workflows	Public verification gateway	Advanced baselines, reports and team workflows
Direct technical support	Not guaranteed	Included while the support/update term is active
Package distribution	Public prerelease planned for Website Phase 40.9	Private delivery only; paid launch belongs to Commercialization Phase 16

Plan	Price	Term / purpose
Community	€0	No time limit
Professional Annual	€399	Development rights for 12 months; applications published while licensed continue to run
Professional 3 Years	€999	Development rights for 36 months; applications published while licensed continue to run
Professional Perpetual	€1,499	Perpetual development and redistribution rights for every eligible version
Perpetual Updates & Support Renewal	€199	Does not change or extend the perpetual right to use eligible versions
Professional Trial	€0	45 days from activation; evaluation only
Enterprise	Custom quote	Defined by order form; may include perpetual rights

Runtime deployment

Paid plans have no runtime royalties or end-user seat charges. Applications published while a valid term license was held continue to run.

14 Versioning

Nexamas.UI uses Semantic Versioning. The first public Community Preview is 1.0.0-preview.1, while AssemblyVersion and FileVersion remain 1.0.0.0 for the approved baseline.

Change	Version effect
Compatible fix, security fix, documentation correction	Patch
Compatible additive public capability	Minor
Intentional public or documented behavioral break	Major with migration evidence and owner approval
Evaluation build before stable authorization	Prerelease identifier such as preview.1

- Community and Professional carry the same product version.
- A published package version is immutable; changed bytes require a new version.
- Do not use “Lifetime Updates.” A perpetual license means perpetual use of eligible versions, not unlimited future updates.
- Update eligibility is determined by the trusted release timestamp and the customer entitlement.

15 Troubleshooting

Check the target framework, platform, package source, public API usage, and disposal pattern before investigating deeper product behavior.

Symptom	Check
Package cannot be restored	Confirm the approved feed/ZIP path, exact prerelease version, and NuGet source priority.
.NET 10 build rejects WinForms types	Use net10.0-windows10.0.19041.0, UseWindowsForms=true, EnableWindowsTargeting=true, and x64.
Window or service remains alive after closing	Dispose MASApplicationWindow and the owning MASApplication for explicit-lifetime applications.
Layout clips localized text	Test minimum size, DPI, Fill/FillWidth behavior, and long RTL/LTR strings.
A claimed route is missing	Verify that it is public in the exact package. PDF/printer output and several internal studios are not public Preview features.

- Record package version, target framework, Windows version, architecture, and a minimal reproduction.
- For Community, use documentation and public updates; direct support is not guaranteed.
- For Professional, include license entitlement details without posting private license material publicly.

16 Release notes

1.0.0-preview.1 is the approved first public Community Preview identity. It does not rewrite the closed internal 1.0.0 engineering baseline.

- One canonical product model for net48 and net10.0-windows10.0.19041.0.
- Stable Nexamas.UI package identity and MAS* public consumer family.
- Approved Community, Community + Pro, and Professional control classification.
- Bilingual website documentation and PDFs generated from one content contract.
- No public Professional binary, checkout, or production trial delivery in Website Phase 40.

Not yet a public download

The Preview download link is activated only in Website Phase 40.9 after packaging, legal, privacy, SEO, hash, browser, and owner-approval gates pass.

17 Support policy

Community relies on documentation and public updates. Professional includes a non-SLA initial-response target and Enterprise services require a separate agreement.

Edition	Support boundary
Community	Documentation, release notes and general updates; no guaranteed direct response.
Professional	Initial-response target: By the end of the next Nexamas business day; not a contractual SLA and no resolution-time guarantee.
Enterprise	A contractual SLA is available only when explicitly included in the order form.

- The Professional target is an initial response by the end of the next Nexamas business day.
- This target is not a contractual SLA and does not guarantee resolution time.
- Perpetual use rights are separate from update and support eligibility.
- The optional €199 annual renewal adds 12 months of updates and support; it does not renew or extend perpetual use because that right does not expire.