

# Virtuoso Overview 1.0.0

## Introduction To Virtuoso

**Virtuoso** is a new category of technology described as “**no-code platform engineering infrastructure**.” It synthesizes ideas from both *no-code platforms* and *platform engineering*, but is distinct from each.

### The Problem: Complexity in Modern Development

#### DevOps & Platform Engineering

DevOps pipelines evolved alongside agile practices to separate development from operations and to automate deployment. While powerful, they introduced friction: setting up an environment can take days before a developer can contribute.

Platform engineering emerged to address this, treating developers as the “customer” and focusing on their experience with pipelines, tools, and processes. However, platform engineering does **not** reach into the code itself. Its roots trace back to Spotify’s *Backstage*.

#### No-Code / Low-Code Platforms

These platforms aim to make application development easier, enabling “citizen developers” to build software without deep technical expertise.

But they come with trade-offs:

- Simplified environments often restrict flexibility.

- Users are cut off from underlying build systems.

- Only curated, vendor-maintained capabilities are available.

- This creates tension between citizen developers and professional coders.

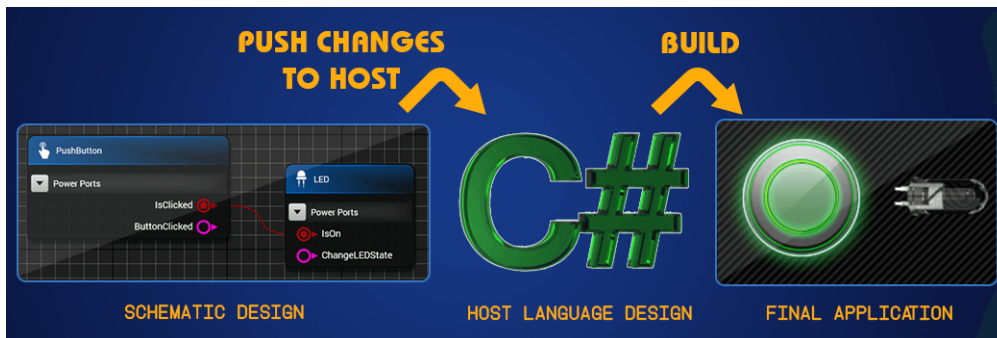
### Virtuoso: A Synthesis

Virtuoso provides **common infrastructure** to enable no-code layers on top of existing pro-code workflows.

**For no-code users:** drag-and-drop, high-level abstractions that “just work” without setup.

**For pro-coders:** a fully formed, build-ready project identical to what they would have developed manually.

This creates **seamless continuity** between no-code and pro-code. A Virtuoso-authored project can be built, run, and extended by professional developers — and even after uninstalling Virtuoso, the codebase continues to function normally.



### The Virtuoso No-Code Philosophy

No-code users should have **no or minimal knowledge** of build systems, compilers, or workflow setup.

Some no-code “activities” may require SDKs or compilers — Virtuoso incorporates platform engineering practices to automate and orchestrate these requirements.

Virtuoso relies on **Montage** (montage-software.com) for a scalable marketplace of fully automated, composable content.

### Virtuoso Architecture

Virtuoso is built on three key layers of extensibility:

#### Virtuoso Core Framework

General-purpose no-code infrastructure adaptable to any pro-code language or project type.

#### Virtuoso No-Code Platform Plugins

Extend the Core Framework to specific programming languages and project types.

Can be developed and shared by the community.

#### Virtuoso Packages

**Port Packages:** Define standardized interfaces between components across no-code platforms.

**Component Packages:** Provide drag-and-drop components, configurable and connectable via port packages.

Together, these allow professional developers to build reusable no-code content at scale.

### Why Virtuoso Matters

**Combines the ease of no-code with the flexibility of pro-code.**

**Eliminates vendor lock-in** — exported projects are fully professional, standalone, and buildable without Virtuoso.

**Scales development** by enabling communities of developers to produce no-code content for any workflow.

**In short:** Virtuoso is the bridge between citizen developers and professional coders, unlocking the speed of no-code without sacrificing the freedom of pro-code.

## Virtuoso Extensibility

### Virtuoso in the Montage Launcher

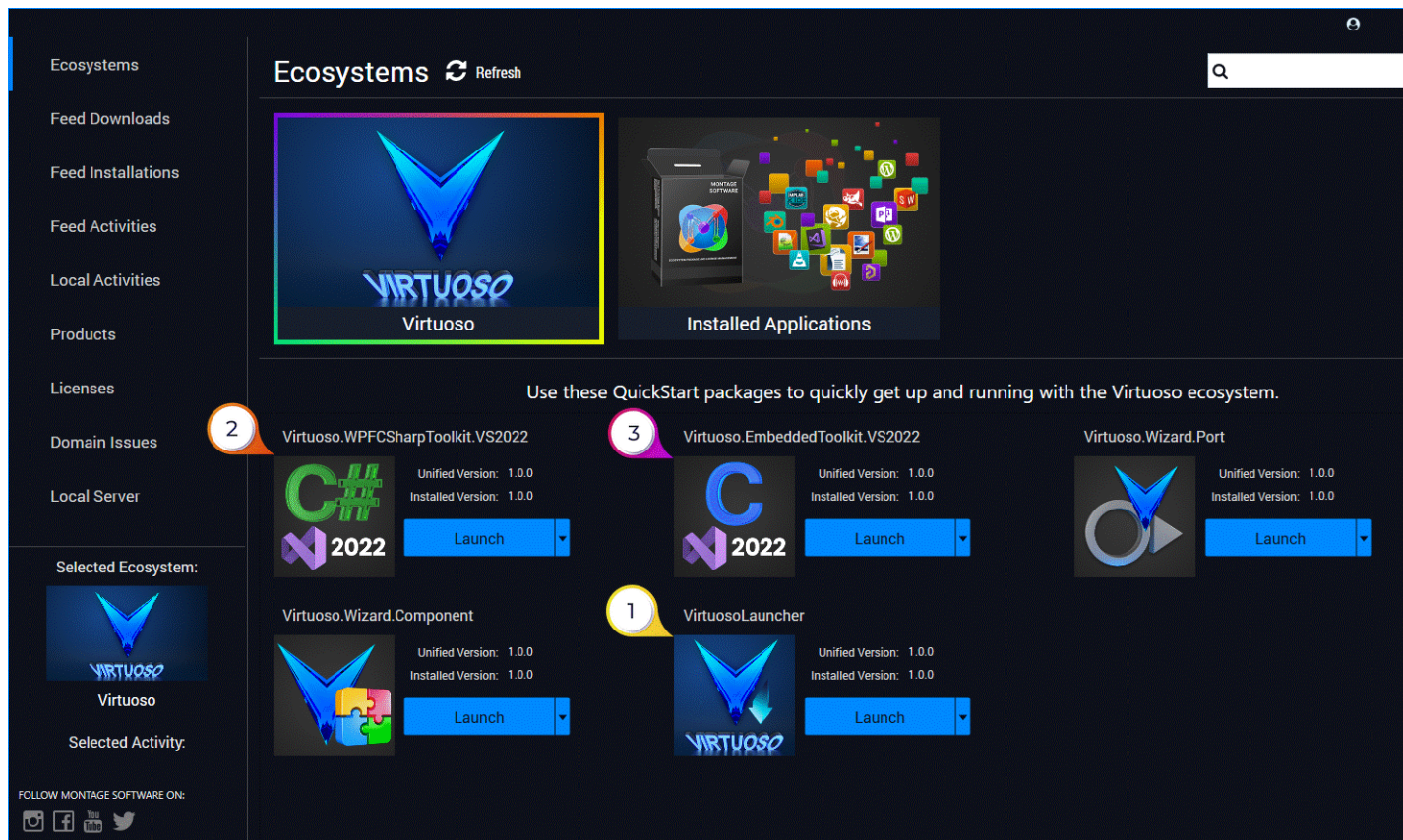
Virtuoso depends heavily on **Montage** for its distribution and extensibility:

**Marketplace:** Montage provides a two-sided marketplace of content creators and consumers, with package governance, licensing, orchestration, and discoverability.

**Ecosystem Organization:** Packages are grouped into logical *ecosystems*, making it easy to organize content across many domains of use.

### The Ecosystem Tab

In the Montage Launcher, the **Virtuoso ecosystem** includes **QuickStart packages** that demonstrate how Virtuoso is extended.



## Key Packages

### Virtuoso Launcher

Represents the **Virtuoso Core Framework**.  
Installed like a normal application on your PC.  
Provides the no-code schematic editor and shared Virtuoso tooling.

### Virtuoso.WPFSharpToolkit.VS2022

A **host plugin** that extends Virtuoso to support **C# WPF desktop apps** in Visual Studio 2022.  
With this installed, you can build no-code C# desktop applications.  
Other host plugins can be developed to target additional project types.

### Virtuoso.EmbeddedToolkit.VS2022

A toolkit for **virtualizing embedded hardware** with virtual microprocessors.  
Depends on the WPF C# Toolkit (2), which in turn depends on the Core Framework (1).

## Dependency Flow

Virtuoso packages are layered with clear dependencies:

**Launcher (Core Framework)** → base layer  
**Host Plugins (e.g., WPF Toolkit)** → extend Virtuoso into a specific language/project type  
**Toolkits (e.g., Embedded Toolkit)** → add specialized capabilities

When you install an **activity** that requires a toolkit, Montage automatically installs all dependencies — reflecting the **platform engineering principle** of automation and orchestration.

## Learn More

For more context on how Montage works, see:

Montage Overview  
Montage Launcher Tabs

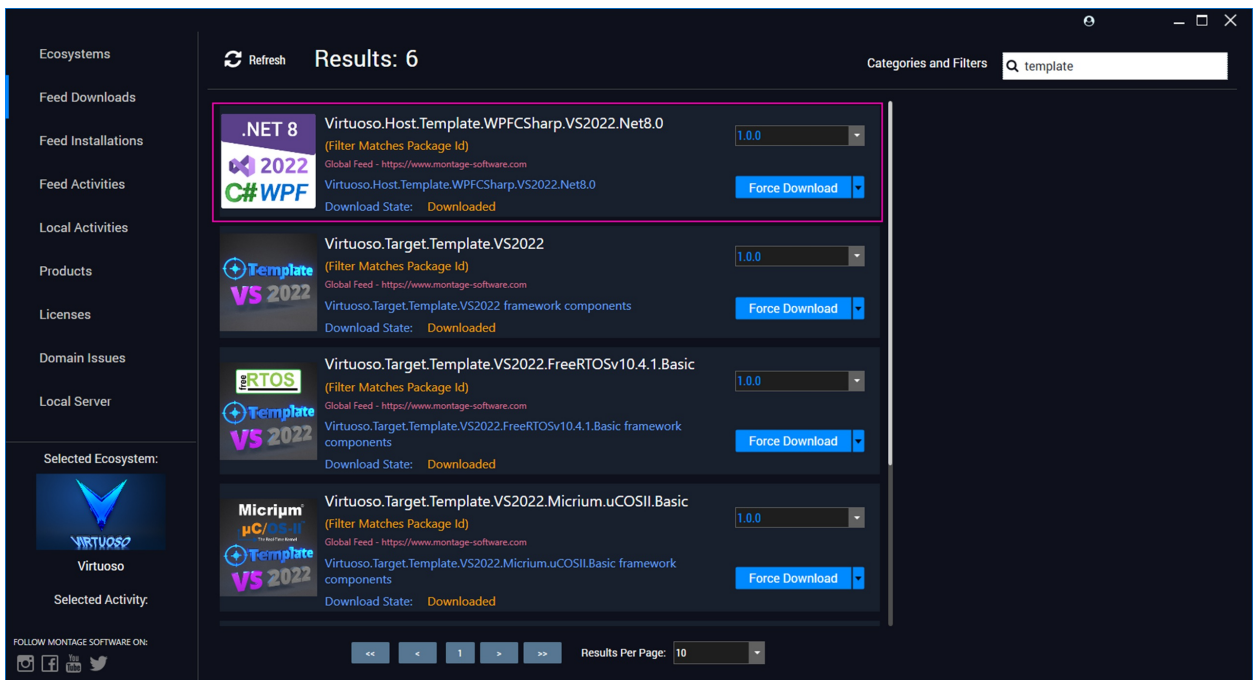
## Creating A New Host Project

### Creating a New Host Project

**Host Template packages** provide starting points for new applications. Once downloaded into Virtuoso, they appear in your environment as selectable templates. For example, the package **Virtuoso.HostTemplate.WPFSharp.VS2022.Net8.0** provides a blank **.NET 8.0 C# WPF project**. Other templates may be built with no-code designs as ready-to-use starting points.

### 1. Downloading Host Templates

Host templates are distributed as Montage packages. Once a template is downloaded, it becomes available for use inside Virtuoso.



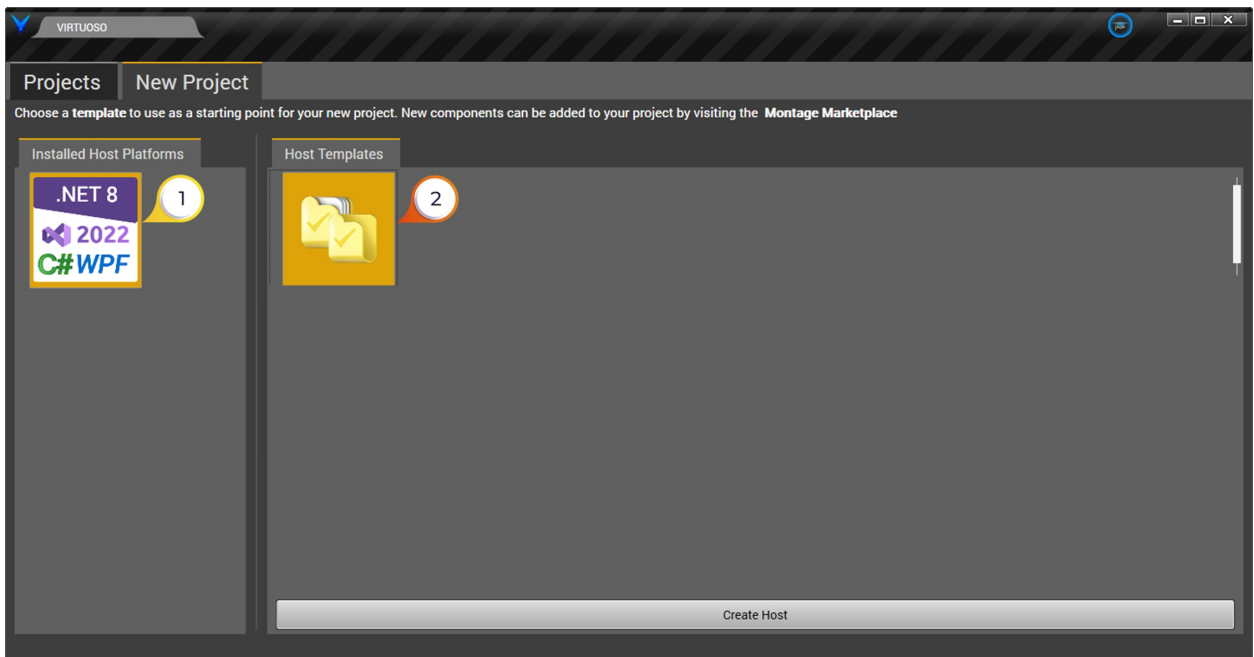
## 2. Selecting Host Platforms & Templates

When you open Virtuoso and choose the **New Project** tab:

The **Installed Host Platforms** panel (left) shows all host platforms you have installed (for example, C# desktop, NodeJS, etc...).  
The **Host Templates** panel (right) lists the available templates for the selected platform.

To create a project:

Select a host platform.  
Choose a template for that platform.  
Click **Create Host**.

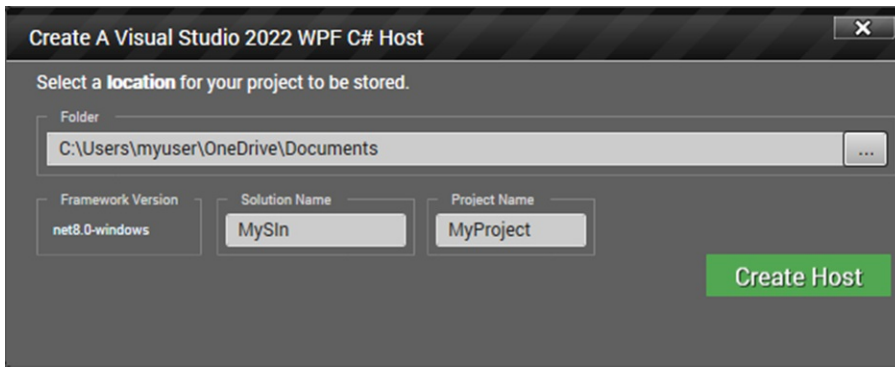


## 3. Configuring Project Details

After selecting a platform and template, Virtuoso prompts for project details:

**Location** where the project will be stored.  
**Solution Name** and **Project Name**.  
Additional platform-specific options (e.g., framework version).





#### 4. Working in the Schematic Editor

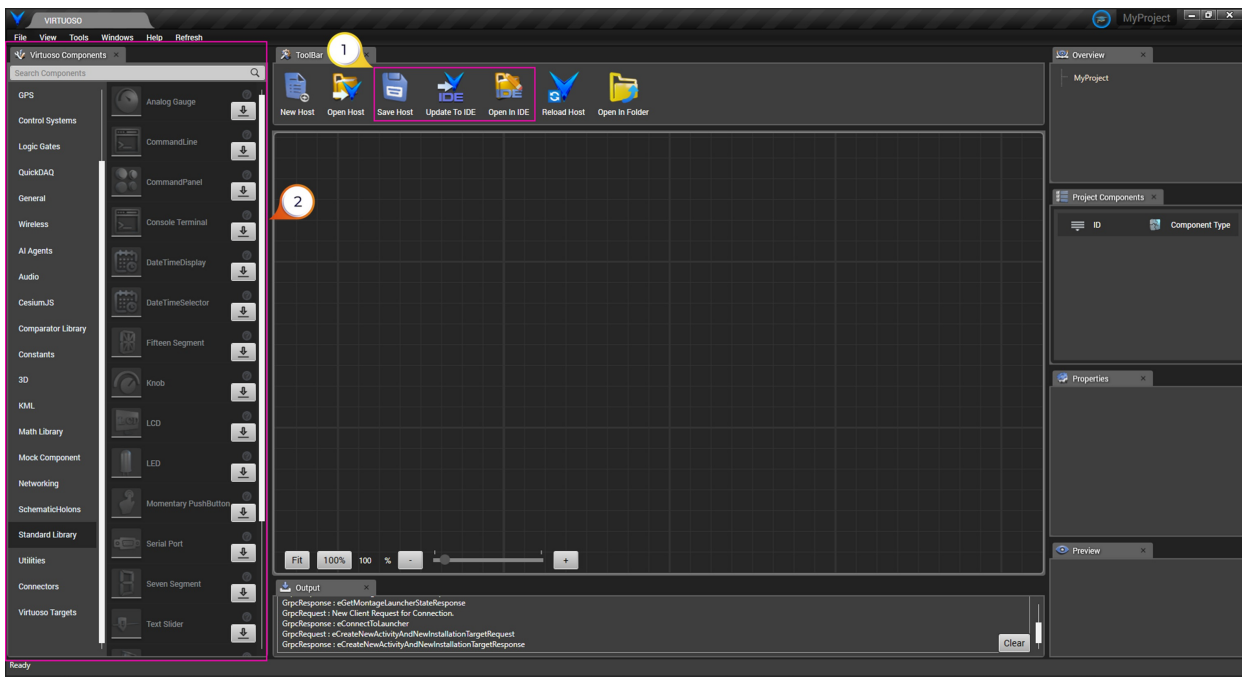
Once the host project is created, the **Virtuoso schematic editor** opens.

The schematic editor itself is provided by the **Virtuoso Core Framework**.

The **host plugin extension** determines how to save, update, and open the host-specific project (1).

It also determines which **no-code components** are usable in this host type (2).

This design allows Virtuoso to support *any host type or programming language* while sharing the same no-code editing environment.



## Building With Components

### Building with Components (LED Example)

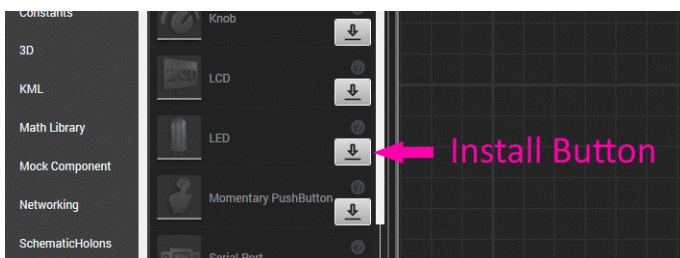
Virtuoso applications are designed in a **high-level visual programming environment**: you drag and drop components (nodes), connect them, and configure their properties. Each component may provide a dedicated **Properties** and **Preview** window to adjust its behavior or visualize its effect.

Components come from **packages**, which must be installed into your host before you can use them. Uninstalled components appear **grayed out** with an **Install** button.

#### 1) Installing a Component Package

##### From the toolbox

Uninstalled components are gray; click the **Install** button.



##### From the right-click palette

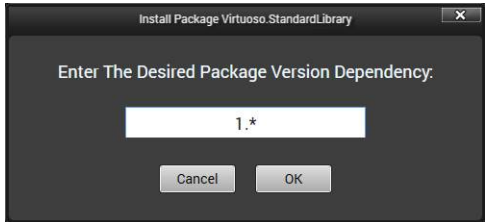
Right-click the schematic, type a name (e.g. LED), and click the **Install** button.



Toolbox = install only  
 Palette = install **and** instantiate

**Choose a version dependency**

Specify the package version. The default 1.\* means “latest minor version under major 1.”



**Watch Montage perform the install**

Virtuoso sends the request to the **Montage Launcher**, where the install appears in the task tray.



**Review installations**

In Montage Launcher → **Local Activities**, click your activity to view:

- Root installation requests (your direct choices)
- Transitive dependencies (what those packages require)

You can also manage requests here and launch your activity.

Ecosystems

Feed Downloads

Feed Installations

Feed Activities

Local Activities


Products

Licenses

Domain Issues


Local Server

Selected Ecosystem:






Virtuoso

Selected Activity:



MyProject

FOLLOW MONTAGE SOFTWARE ON:







← Back to activities...

Launch

Activity Detail For C:\Users\me\Desktop\MyHost\virtuoso\MyProject\Holon\MyProject.maprj

Root Installation Requests:







Installation Target	Requested Package	Package Type	Requested Version	Unified Result
	Virtuoso.Host.Runtime.DotNet	Montage	1.*	1.0.0
	Montage.PackageClient	Montage	1.*	1.0.0
	Montage.LicenseClient	Montage	1.*	1.0.0
	Virtuoso.StandardLibrary	Montage	1.*	1.0.0

Add

Remove

Manage

Associated Target Packages:

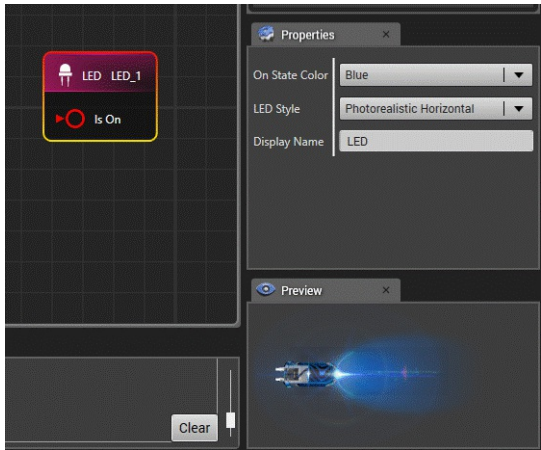
Installation Target	Package	Package Type	Unification Result	Installation State
	Virtuoso.Host.Runtime.DotNet	Montage	1.0.0	1.0.0 (Installed)
	Montage.LicenseClient	Montage	1.0.0	1.0.0 (Installed)
	Virtuoso.Montage.Licensing	Montage	1.0.0	1.0.0 (Installed)
	Montage.PackageClient	Montage	1.0.0	1.0.0 (Installed)
	Virtuoso.StandardLibrary	Montage	1.0.0	1.0.0 (Installed)
	Virtuoso.Port.DotNet.Standard	Montage	1.0.0	1.0.0 (Installed)

2) Instantiating & Configuring a Component

**Place the LED**  
Right-click schematic → type LED → select.



**Configure properties & preview**  
Change LED color, style, and display name in **Properties**. The **Preview** shows results.

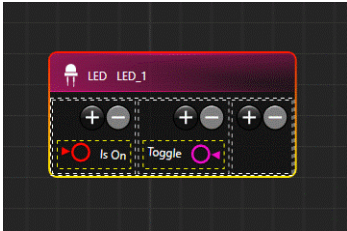


3) Editing Node Ports

**Enter Edit Mode**  
Right-click node → **Enter Edit Mode**.

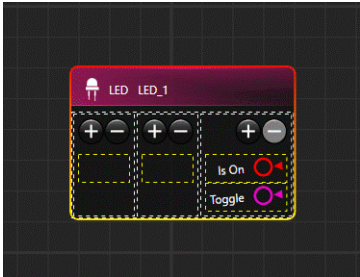


**Rearrange ports**  
Drag ports into left, right, or center (hidden) slots. Add/remove slots as needed.



### Expose hidden ports

The LED has a hidden **Toggle** event port. Add slots on the right panel by clicking the “+” button and move both **Is On** and **Toggle** there.



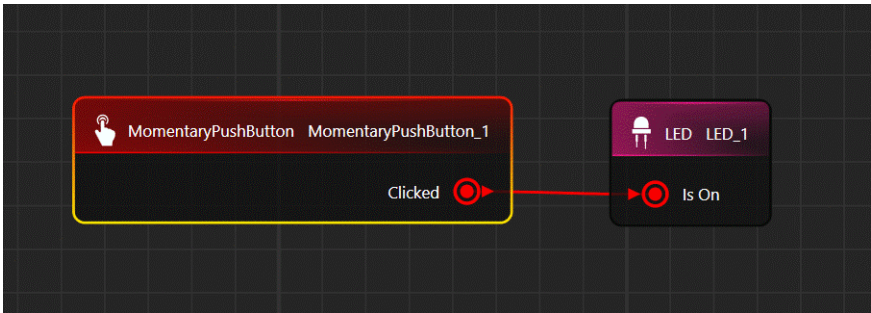
### Exit Edit Mode

Click outside the node. You can now connect other node ports.

## 4) Wiring Components

### Connect a button to the LED

Add a **Momentary Pushbutton** and connect its **Clicked** Boolean driver → LED's **Is On** input.



## 5) Moving to Visual Studio

### Update and open in IDE

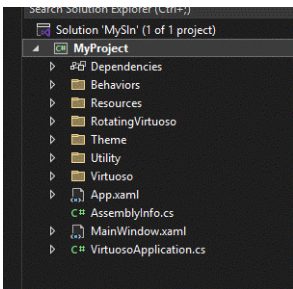
Use the toolbar:

- Save Host
- Update To IDE (push design)
- Open In IDE



### Explore project structure

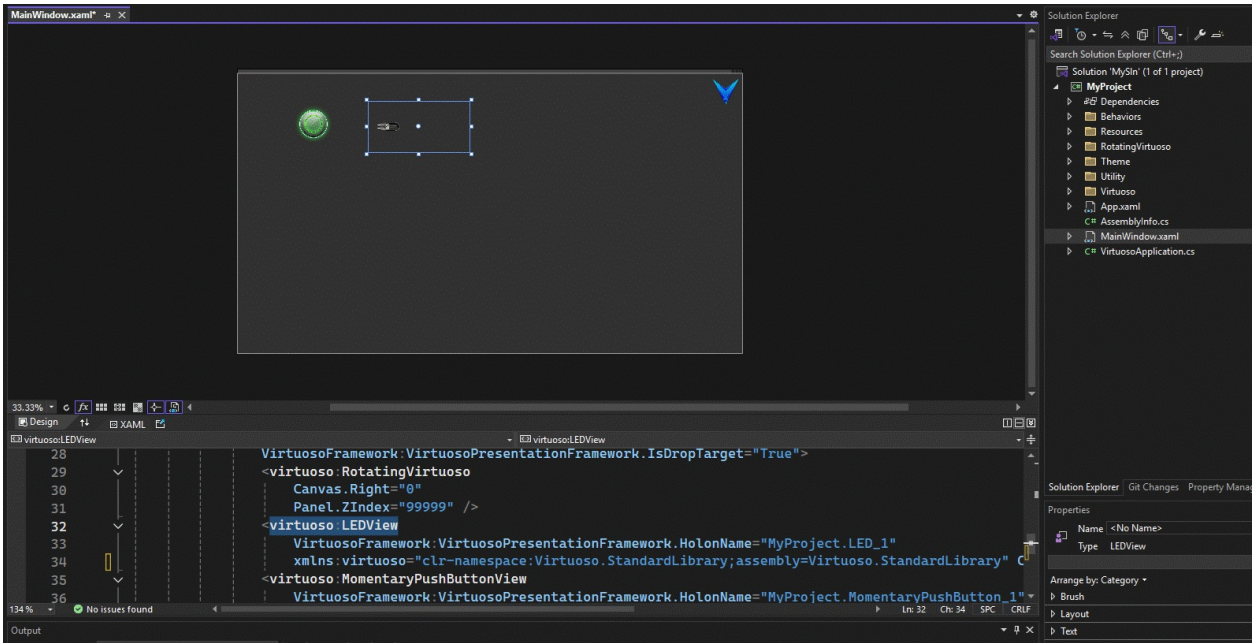
Virtuoso generates a complete Visual Studio project. Expand the tree in **Solution Explorer**.





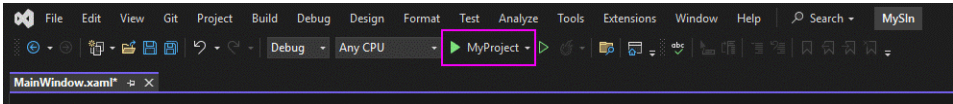
**Edit your view**

Open MainWindow.xaml. Your components appear on the canvas (top) with matching XAML (bottom). Drag and position them in the visual editor.



**Build and run**

Click the green arrow to run. Pressing the button toggles the LED. Note: some schematic components may not have a visible UI representation.



**Virtuoso Licensing**

**Virtuoso Licensing Overview**

The **Virtuoso Core Framework** is free to use for development. Commercial deployment is also permitted — many deployments are royalty-free, while others may include a **5% royalty on gross revenue**. Full details are described in the Core Framework EULA and the Virtuoso Pricing page.

This page explains how licensing applies across the Core Framework, No-Code host platforms, host component libraries, and special toolkits.

**Virtuoso Core Framework Licensing**

The Virtuoso Core Framework software includes:

- Visual programming environment** (schematic editor and scalable toolbox of components)
- No-Code host platforms** (e.g., C# WPF desktop application host), built as plugin extensions of the Core Framework
- Wizards** (Virtuoso component wizard and port wizard)

Although distinct software packages, all of these elements are part of the Core Framework and are licensed under the Core Framework EULA.

Key takeaway:

- Free for development
- Never a cost for internal use and internal deployment
- Commercial deployment: royalty-free in some cases, 5% royalty in others

**No-Code Host Platforms**

A No-Code host platform (for example, the **C# WPF desktop host**) is delivered as a **plugin extension** of the Core Framework. Because of this, it is also covered by the Core Framework EULA.

Key takeaway:



Covered by the same rules as the Core Framework (free for dev, possible royalty on deployment).

---

### **Host Component Libraries**

Host component libraries (such as Virtuoso.StandardLibrary) are licensed under the **Virtuoso Host Component End User License Agreement**.

Key takeaway:

Free to use under the Host Component EULA.

---

### **Embedded Toolkit Exception**

The Virtuoso.EmbeddedToolkit.VS2022 package enables the virtualization of embedded systems hardware. It is an example of “premium content” and is licensed separately. Please see the Virtuoso Embedded Toolkit End User License Agreement and the Virtuoso Pricing page for details.

Key takeaway:

The embedded virtualization toolkit is not covered by the Core or Host Component EULAs — check its specific license.