

# Developing With NVIDIA Omniverse

An Introduction to Developing With NVIDIA Omniverse

# Install Supporting Tools

- **Omniverse 應用程式** 由一個 **.kit 檔案** 定義，這是一個文字檔案，用於引用應用程式中的所有擴展與設定。

- 需要使用 **Git** 和 **Visual Studio Code (VS Code)**

- **安裝:**

1. 程式碼儲存庫: Clone → [Omniverse Kit App Template](#)

- **注意:** 這裡建議 **Clone** 到檔案路徑較短的位置，較長的文件路徑可能會導致問題。該路徑名稱中盡量不包含空格

2. 開啟 VS Code → 在 VS Code 中，按 **Ctrl+Shift+P** 開啟 VS Code 視窗頂部的命令面板

3. 在指令面板中，透過輸入 **gitclone** 搜尋 Git Clone 指令，然後選擇 **Git: Clone** 選項

4. 將程式碼儲存庫的 [URL](#) 貼到指令面板中，然後按 **Enter**。

- **注意:** **Clone** 操作可能需要一些時間才能完成

5. Clone 操作完成後，系統將詢問您是否要開啟儲存庫。按一下 **“Open”**。

6. 現在已經成功 Clone 了 **Kit App Template** 儲存庫，可以準備從這個儲存庫範本建立一個 **Kit SDK 應用程式**。

# Create a Basic Application

- 請確保 kit-app-template 資料夾已在 VS Code 中開啟，並且終端視窗也已開啟。kit-app-template 資料夾下的子資料夾將顯示在 VS Code 資源管理器面板中。
- 在終端機中，輸入指令並執行 (此指令會利用 Kit SDK 資源，依你所選或輸入的資訊，建立一個基本的應用程式)：

➤ Windows:

```
.\repo.bat template new
```

➤ Linux/macOS:

```
./repo.sh template new
```

- 你會被詢問一系列問題。方向鍵選擇選項，或輸入回覆，完成後按下 Enter。請務必選擇 Kit Base Editor 作為範本。

```
Do you accept the EULA (End User License Agreement)?  
(Select Yes or No with arrow keys): Yes  
[ctrl+c to Exit]  
? Select with arrow keys what you want to create: Application>  
? Select with arrow keys your desired template: [kit_base_editor]: Kit Base Editor  
? Enter name of application .kit file [name-spaced, lowercase, alphanumeric]:  
    my_company.my_editor  
? Enter application_display_name: My Editor  
? Enter version: 0.1.0
```

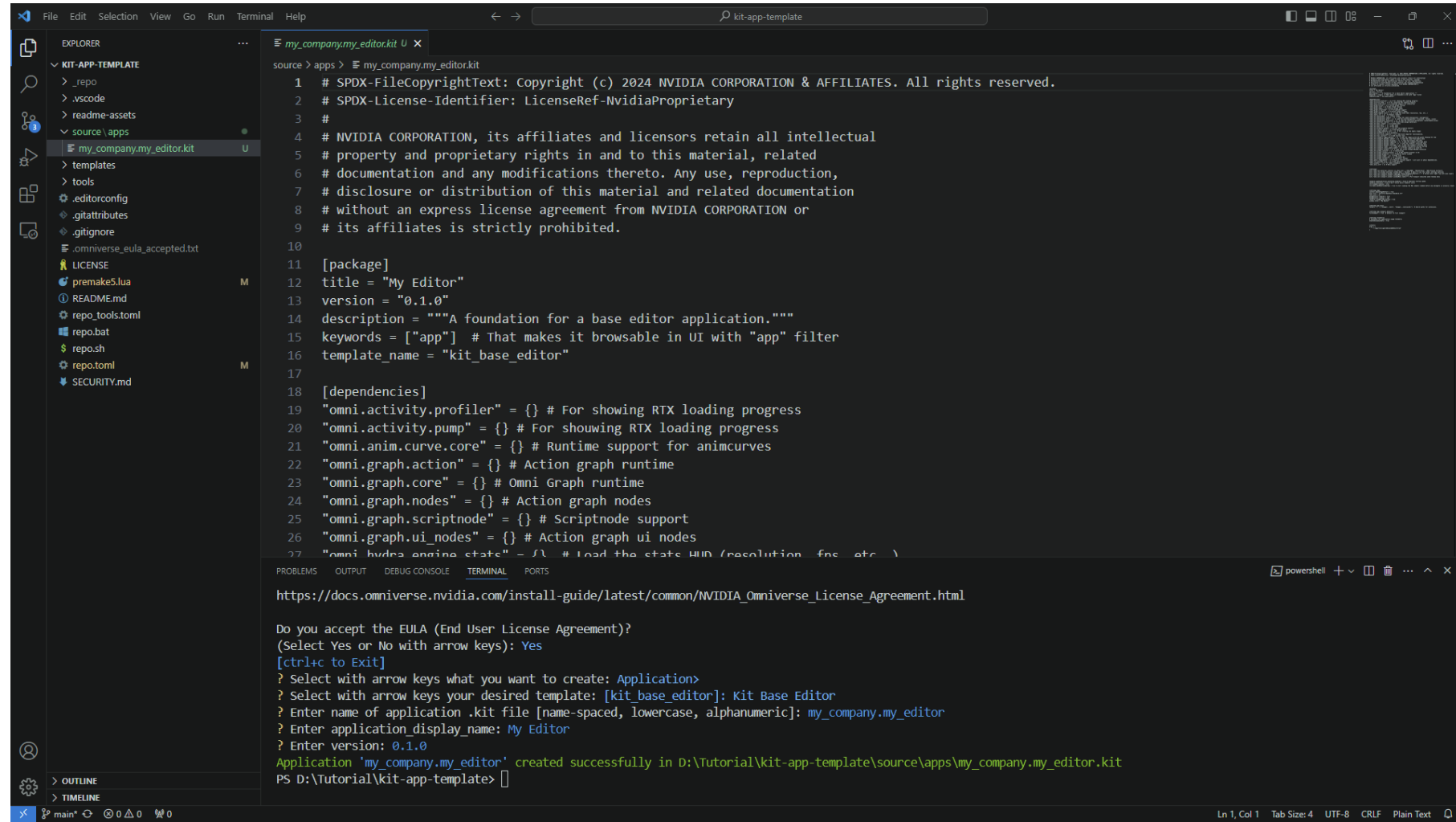
# Explore the Kit Base Editor File

- 上一個步驟中的 `template new` 指令建立了一個以你指定名稱命名的 `.kit` 文件，你可以在 **kit-app-template** 資料夾的 **source\apps** 目錄下找到該文件 (預設名稱: **my\_company.my\_editor.kit**)。

.kit 檔案的內容是文字格式

.kit 檔案本質上是一份清單，列出了你希望從 Omniverse 生態系統中提取哪些內容用於您的應用程式。

.kit 檔案中列出的所有內容都是應用程式的建構塊。



The screenshot shows a code editor with the Explorer sidebar on the left displaying the file structure of the kit-app-template project. The file `my_company.my_editor.kit` is selected under the `source\apps` directory. The main editor area displays the content of this file, which is a JSON-like configuration for a Kit Base Editor application. The configuration includes a license, package information (title, version, description, keywords, template\_name), and a list of dependencies. The terminal at the bottom shows the output of the `template new` command, confirming the successful creation of the application.

```
source > apps > my_company.my_editor.kit
1  # SPDX-FileCopyrightText: Copyright (c) 2024 NVIDIA CORPORATION & AFFILIATES. All rights reserved.
2  # SPDX-License-Identifier: LicenseRef-NvidiaProprietary
3  #
4  # NVIDIA CORPORATION, its affiliates and licensors retain all intellectual
5  # property and proprietary rights in and to this material, related
6  # documentation and any modifications thereto. Any use, reproduction,
7  # disclosure or distribution of this material and related documentation
8  # without an express license agreement from NVIDIA CORPORATION or
9  # its affiliates is strictly prohibited.
10
11 [package]
12 title = "My Editor"
13 version = "0.1.0"
14 description = ""A foundation for a base editor application.""
15 keywords = ["app"] # That makes it browsable in UI with "app" filter
16 template_name = "kit_base_editor"
17
18 [dependencies]
19 "omni.activity.profiler" = {} # For showing RTX loading progress
20 "omni.activity.pump" = {} # For showing RTX loading progress
21 "omni.anim.curve.core" = {} # Runtime support for animcurves
22 "omni.graph.action" = {} # Action graph runtime
23 "omni.graph.core" = {} # Omni Graph runtime
24 "omni.graph.nodes" = {} # Action graph nodes
25 "omni.graph.scriptnode" = {} # Scriptnode support
26 "omni.graph.ui_nodes" = {} # Action graph ui nodes
27 "omni.hydra.engine.state" = {} # Load the state HUD (resolution, fps, etc.)

https://docs.omniverse.nvidia.com/install-guide/latest/common/NVIDIA_Omniverse_License_Agreement.html

Do you accept the EULA (End User License Agreement)?
(Select Yes or No with arrow keys): Yes
[ctrl+c to Exit]
? Select with arrow keys what you want to create: Application>
? Select with arrow keys your desired template: [kit_base_editor]: Kit Base Editor
? Enter name of application .kit file [name-spaced, lowercase, alphanumeric]: my_company.my_editor
? Enter application_display_name: My Editor
? Enter version: 0.1.0
Application 'my_company.my_editor' created successfully in D:\Tutorial\kit-app-template\source\apps\my_company.my_editor.kit
PS D:\Tutorial\kit-app-template>
```

# Build the Base Application

- 要能夠執行應用程式，你需要先 **進行建置 ( build )** 。
- 在 VS Code 的終端機視窗中輸入以下指令：

## ➤ Windows:

```
.\repo.bat build
```

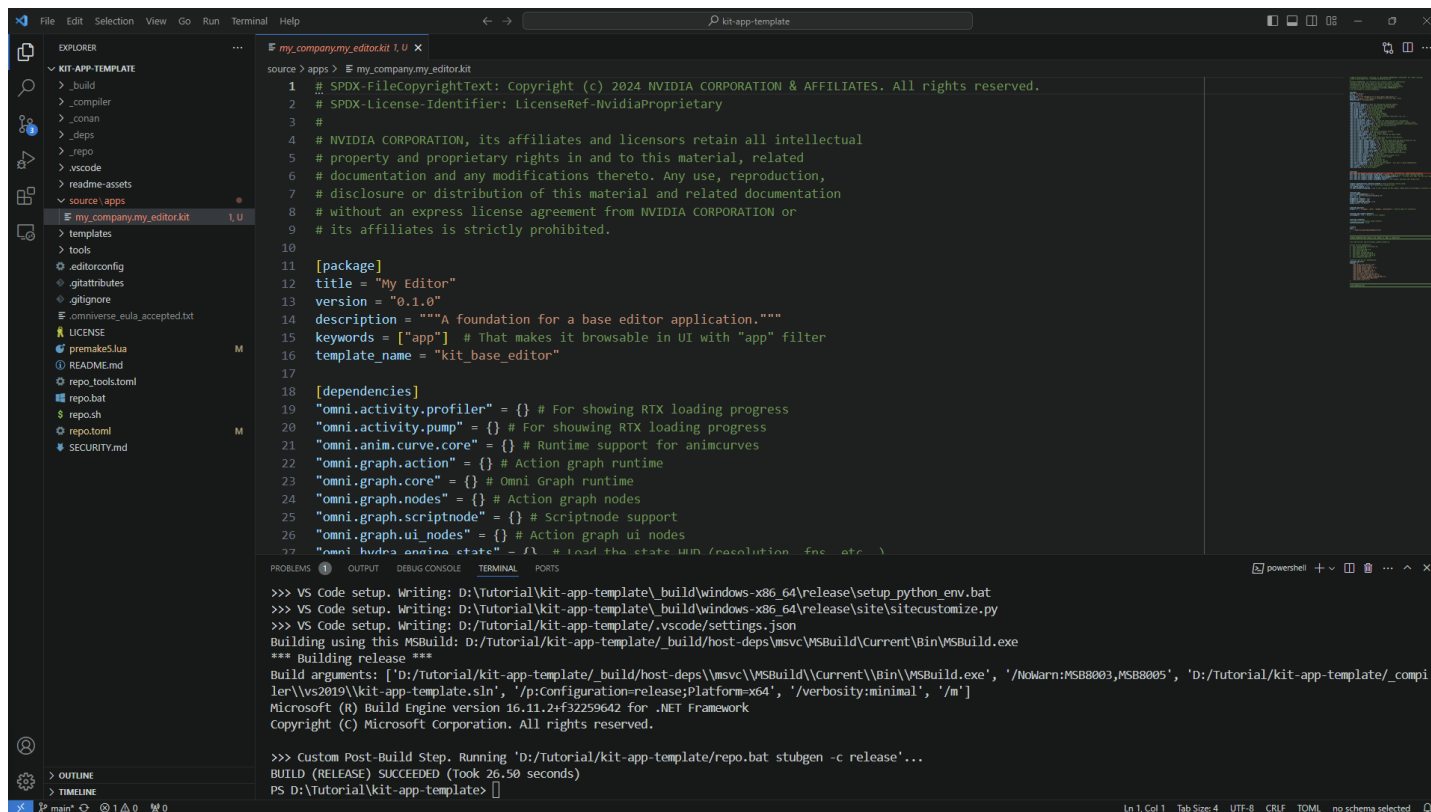
## ➤ Linux/macOS:

```
./repo.sh build
```

- 接著等待一段時間，應用程式會開始建置。  
當建置完成時，會顯示訊息：

```
BUILD (RELEASE) SUCCEEDED (<time taken>)
```

- 整個過程大約需要幾分鐘，之後終端機面板會再次出現提示符號。



```
1  # SPDX-FileCopyrightText: Copyright (c) 2024 NVIDIA CORPORATION & AFFILIATES. All rights reserved.
2  # SPDX-License-Identifier: LicenseRef-NvidiaProprietary
3  #
4  # NVIDIA CORPORATION, its affiliates and licensors retain all intellectual
5  # property and proprietary rights in and to this material, related
6  # documentation and any modifications thereto. Any use, reproduction,
7  # disclosure or distribution of this material and related documentation
8  # without an express license agreement from NVIDIA CORPORATION or
9  # its affiliates is strictly prohibited.
10
11 [package]
12 title = "My Editor"
13 version = "0.1.0"
14 description = ""A foundation for a base editor application.""
15 keywords = ["app"] # That makes it browsable in UI with "app" filter
16 template_name = "kit_base_editor"
17
18 [dependencies]
19 "omni.activity.profiler" = {} # For showing RTX loading progress
20 "omni.activity.pump" = {} # For showing RTX loading progress
21 "omni.anim.curve.core" = {} # Runtime support for animcurves
22 "omni.graph.action" = {} # Action graph runtime
23 "omni.graph.core" = {} # Omni Graph runtime
24 "omni.graph.nodes" = {} # Action graph nodes
25 "omni.graph.scripnode" = {} # Scripnode support
26 "omni.graph.ui_nodes" = {} # Action graph ui nodes
27 "omni.hudra.engine.state" = {} # Load the state HUD (resolution, fps, etc.)
28
29 >>> VS Code setup. Writing: D:\Tutorial\kit-app-template\_build\windows-x86_64\release\setup_python_env.bat
30 >>> VS Code setup. Writing: D:\Tutorial\kit-app-template\_build\windows-x86_64\release\site\sitecustomize.py
31 >>> VS Code setup. Writing: D:\Tutorial\kit-app-template\.vscode\settings.json
32 Building using this MSBuild: D:\Tutorial\kit-app-template\_build\host-deps\msvc\MSBuild\Current\Bin\MSBuild.exe
33 *** Building release ***
34 Build arguments: ['D:/Tutorial/kit-app-template/_build/host-deps/msvc/MSBuild/Current/Bin/MSBuild.exe', '/NoWarn:MSB8003,MSB8005', 'D:/Tutorial/kit-app-template/_compilers/vs2019/kit-app-template.sln', '/p:Configuration=release;Platform=x64', '/verbosity:minimal', '/m']
35 Microsoft (R) Build Engine version 16.11.24f32259642 for .NET Framework
36 Copyright (c) Microsoft Corporation. All rights reserved.
37
38 >>> Custom Post-Build Step. Running 'D:/Tutorial/kit-app-template/repo.bat stubgen -c release'...
39 BUILD (RELEASE) SUCCEEDED (Took 26.50 seconds)
40 PS D:\Tutorial\kit-app-template>
```

# Launch the Base Application

- 接著，啟動（**launch**）前面建置的應用程式。
- 在 VS Code 的終端機視窗中輸入以下指令：

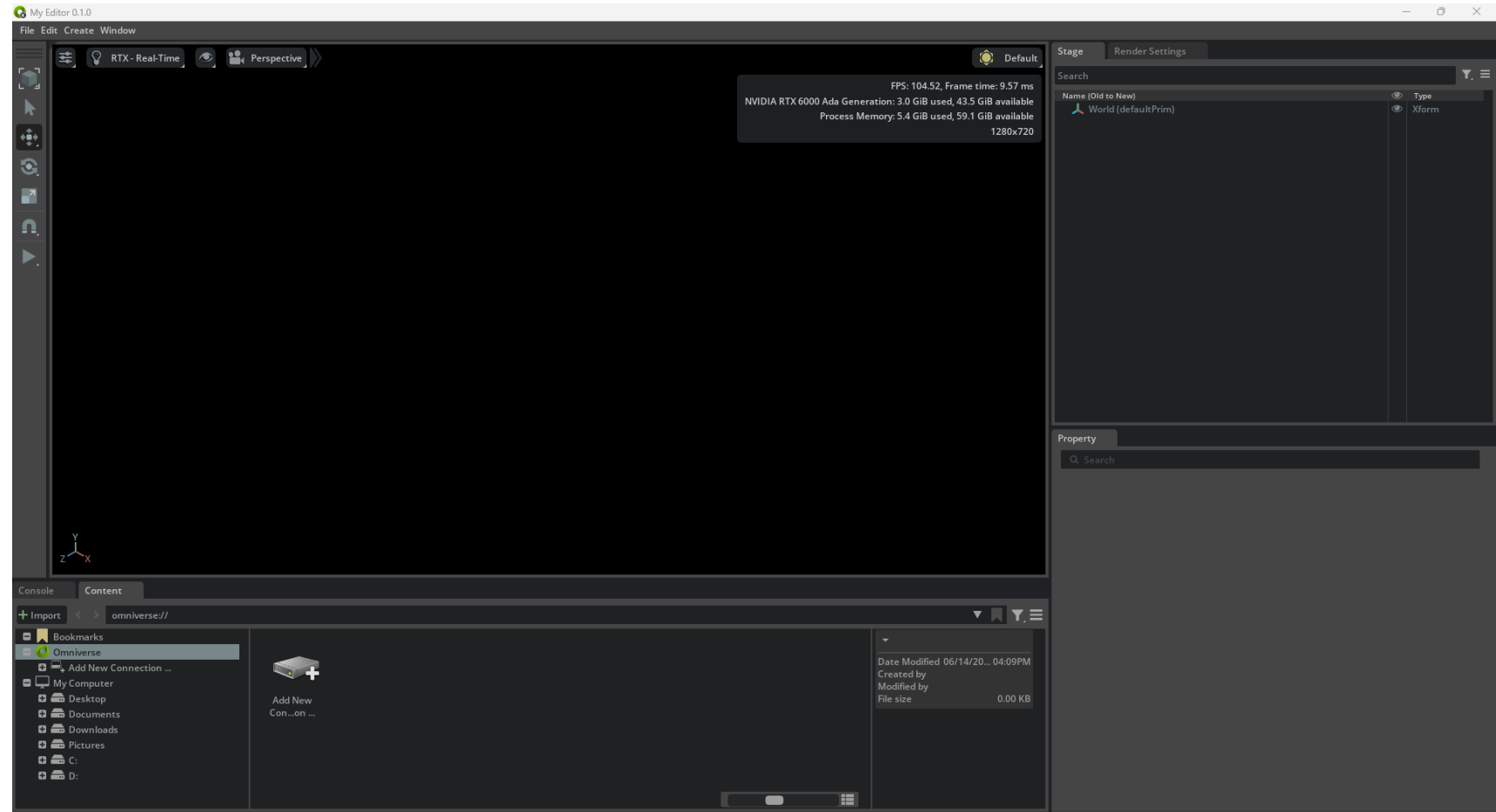
➤ **Windows:**

**.\repo.bat launch**

➤ **Linux/macOS:**

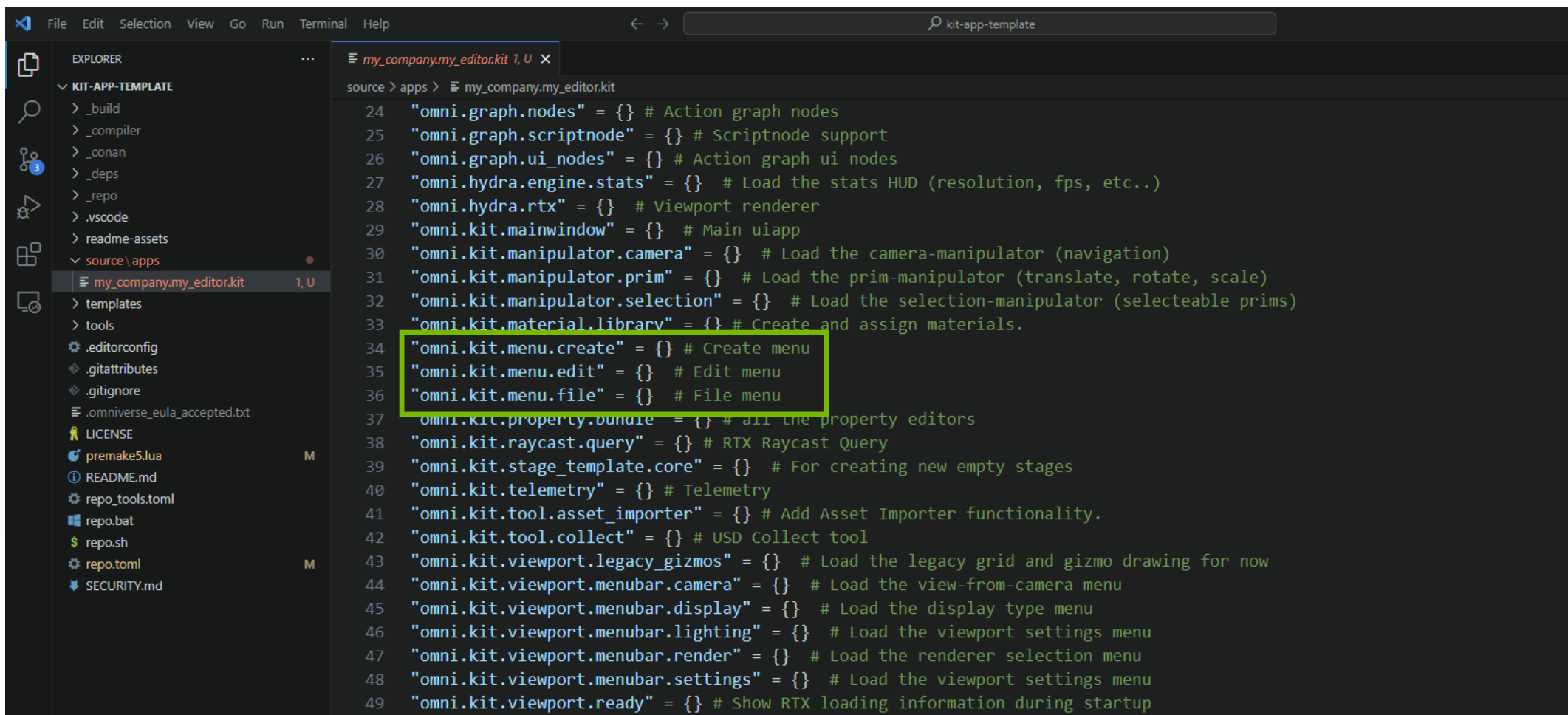
**./repo.sh launch**

- 在終端機按 Enter 啟動應用程式
- 首次啟動需數分鐘，之後會更快。啟動完成後，Omniverse 會在新視窗開啟，RTX 首次載入同樣可能需幾分鐘，完成後即可繼續。



# Explore the Base Application

- 前面完成了應用程式建置，對於介面中的選單選項，我們可以與 .kit 檔案中的**擴充功能(extension)**對應起來
- 前一步驟中可看到，應用程式有四個選單：**File**、**Edit**、**Create**、**Window**
- 在 .kit 檔中對應 (這些擴充功能在應用程式中創建這些選單)：



The screenshot shows a code editor with a dark theme. On the left, the Explorer panel displays a file tree for a project named 'KIT-APP-TEMPLATE'. The file 'my\_company.my\_editor.kit' is selected and highlighted. The main editor area shows the content of this file, which is a Lua script defining various configuration options for the application. A yellow rectangular box highlights three lines of code: 'omni.kit.menu.create', 'omni.kit.menu.edit', and 'omni.kit.menu.file'. These lines define the configuration for the 'Create', 'Edit', and 'File' menus, respectively. The code is as follows:

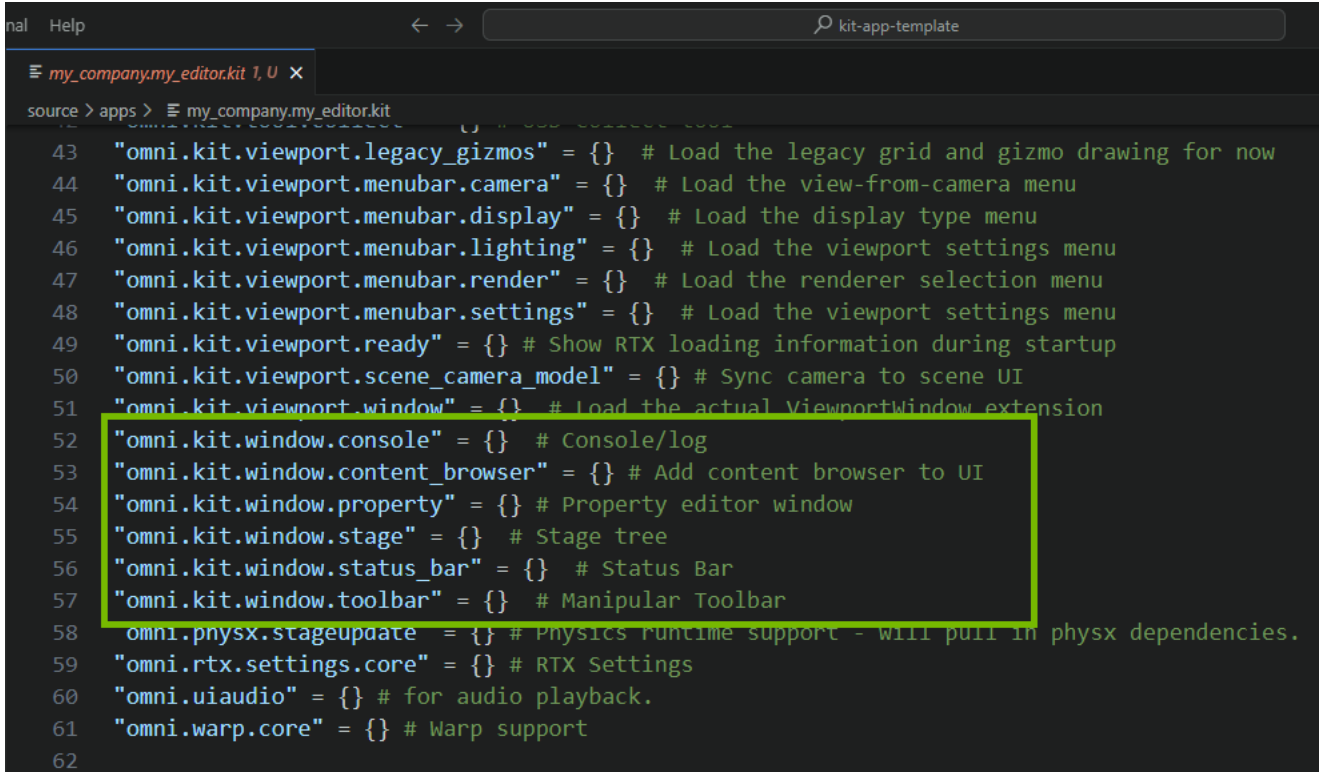
```
source > apps > my_company.my_editor.kit

24 "omni.graph.nodes" = {} # Action graph nodes
25 "omni.graph.scriptnode" = {} # Scriptnode support
26 "omni.graph.ui_nodes" = {} # Action graph ui nodes
27 "omni.hydra.engine.stats" = {} # Load the stats HUD (resolution, fps, etc..)
28 "omni.hydra.rtx" = {} # Viewport renderer
29 "omni.kit.mainwindow" = {} # Main uiapp
30 "omni.kit.manipulator.camera" = {} # Load the camera-manipulator (navigation)
31 "omni.kit.manipulator.prim" = {} # Load the prim-manipulator (translate, rotate, scale)
32 "omni.kit.manipulator.selection" = {} # Load the selection-manipulator (selecteable prims)
33 "omni.kit.material.library" = {} # Create and assign materials.
34 "omni.kit.menu.create" = {} # Create menu
35 "omni.kit.menu.edit" = {} # Edit menu
36 "omni.kit.menu.file" = {} # File menu
37 "omni.kit.property.bundle" = {} # all the property editors
38 "omni.kit.raycast.query" = {} # RTX Raycast Query
39 "omni.kit.stage_template.core" = {} # For creating new empty stages
40 "omni.kit.telemetry" = {} # Telemetry
41 "omni.kit.tool.asset_importer" = {} # Add Asset Importer functionality.
42 "omni.kit.tool.collect" = {} # USD Collect tool
43 "omni.kit.viewport.legacy_gizmos" = {} # Load the legacy grid and gizmo drawing for now
44 "omni.kit.viewport.menubar.camera" = {} # Load the view-from-camera menu
45 "omni.kit.viewport.menubar.display" = {} # Load the display type menu
46 "omni.kit.viewport.menubar.lighting" = {} # Load the viewport settings menu
47 "omni.kit.viewport.menubar.render" = {} # Load the renderer selection menu
48 "omni.kit.viewport.menubar.settings" = {} # Load the viewport settings menu
49 "omni.kit.viewport.ready" = {} # Show RTX loading information during startup
```

# Explore the Base Application

■ 在 .kit 檔案中，找到以 **omni.kit.window** 開頭的擴充功能：

Extension	功能
omni.kit.window.console	Console / 日誌視窗
omni.kit.window.content_browser	內容瀏覽器
omni.kit.window.property	屬性編輯器
omni.kit.window.stage	Stage 樹狀視圖
omni.kit.window.status_bar	狀態列
omni.kit.window.toolbar	操作工具列



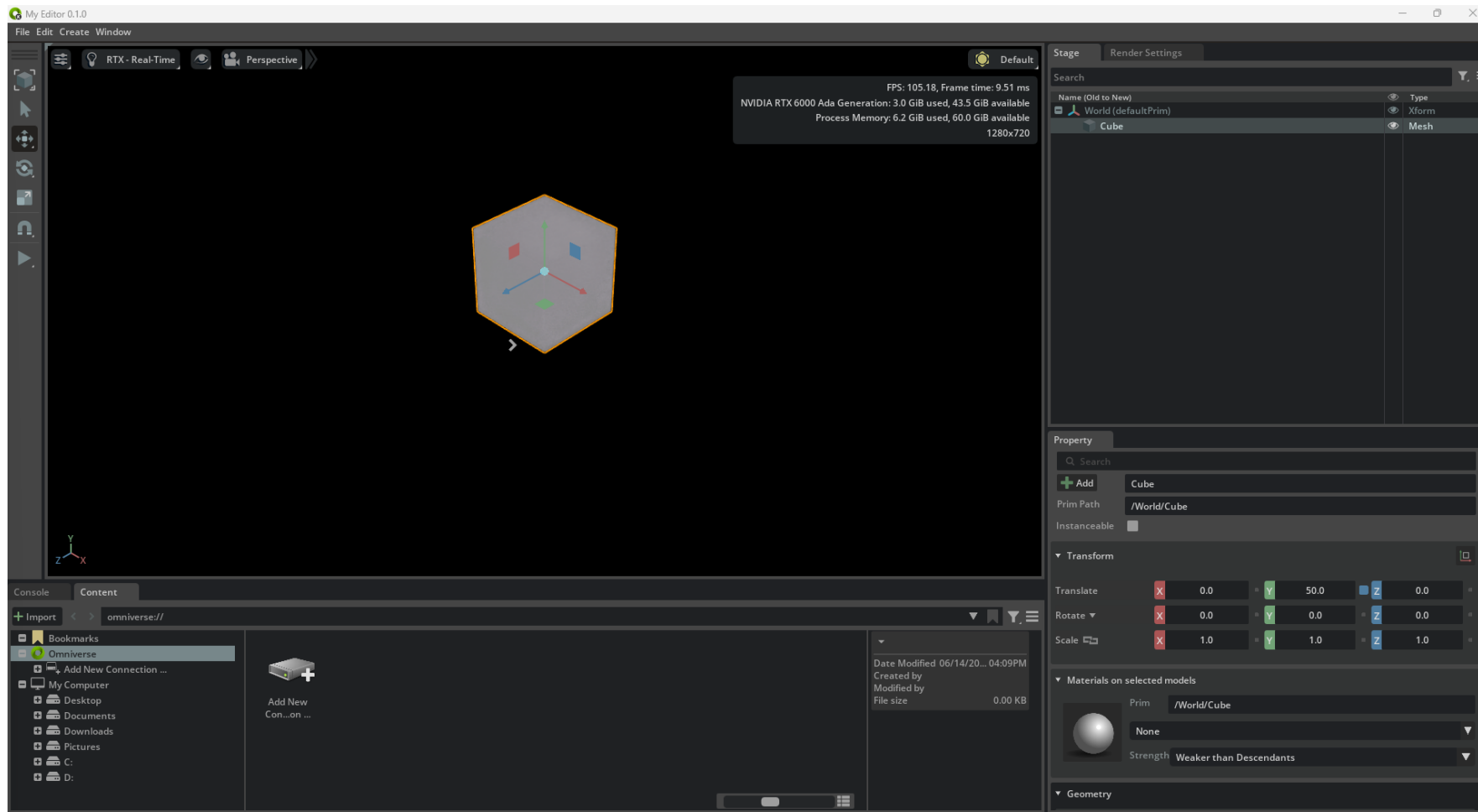
```
43 "omni.kit.viewport.legacy_gizmos" = {} # Load the legacy grid and gizmo drawing for now
44 "omni.kit.viewport.menubar.camera" = {} # Load the view-from-camera menu
45 "omni.kit.viewport.menubar.display" = {} # Load the display type menu
46 "omni.kit.viewport.menubar.lighting" = {} # Load the viewport settings menu
47 "omni.kit.viewport.menubar.render" = {} # Load the renderer selection menu
48 "omni.kit.viewport.menubar.settings" = {} # Load the viewport settings menu
49 "omni.kit.viewport.ready" = {} # Show RTX loading information during startup
50 "omni.kit.viewport.scene_camera_model" = {} # Sync camera to scene UI
51 "omni.kit.viewport.window" = {} # Load the actual ViewportWindow extension
52 "omni.kit.window.console" = {} # Console/log
53 "omni.kit.window.content_browser" = {} # Add content browser to UI
54 "omni.kit.window.property" = {} # Property editor window
55 "omni.kit.window.stage" = {} # Stage tree
56 "omni.kit.window.status_bar" = {} # Status Bar
57 "omni.kit.window.toolbar" = {} # Manipular Toolbar
58 omni.physx.stageupdate = {} # Physics runtime support - will pull in physx dependencies.
59 "omni.rtx.settings.core" = {} # RTX Settings
60 "omni.uiaudio" = {} # for audio playback.
61 "omni.warp.core" = {} # Warp support
62
```

■ 這些擴充功能會在應用程式中建立新的視窗或區塊（如 Stage、Property 窗口），並自動將它們加入 **Window 選單**。由於視窗是應用程式的核心，**Window 選單始終存在**，所有視窗都可從此選單存取。



# Explore the Base Application

- 接著，我們來探索一些應用程式的功能。
- 在應用程式的**工具列 ( Toolbar )** 中，選擇 **建立 ( Create )** > **網格 ( Mesh )** > **立方體 ( Cube )**，在**檢視區 ( Viewport )** 中建立一個立方體。現在在 **Stage** 區域中，應該可以看到一個列出的立方體 ( Cube )。



# Explore the Base Application

透過以下方式移動立方體：

1.移動物件：選左側工具列的 **Move** 工具或按 **W**，拖曳箭頭移動方塊。

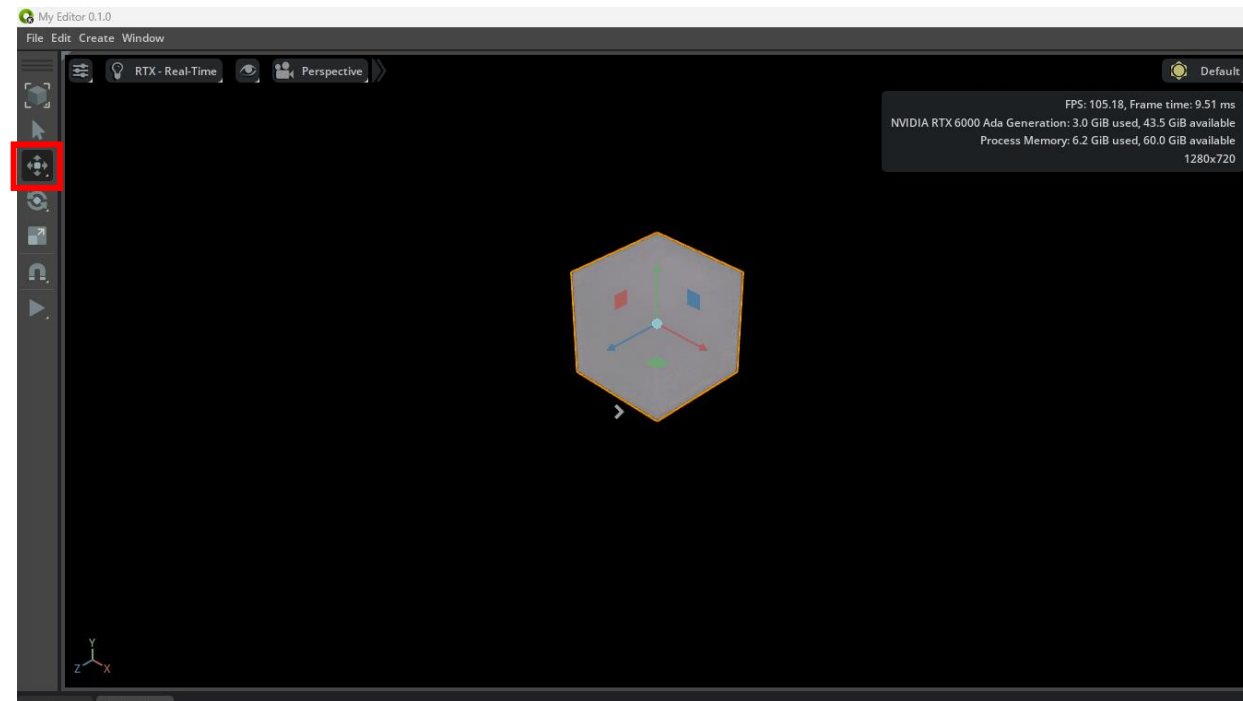
2.建立物件：可新增更多 Mesh。

3.視角操作：

- 按住 **右鍵 (RMB) + WASD** → 移動畫面
- 滾動滑鼠中鍵 (RMB 按住中滾) → 調整移動速度
- 拖曳 **RMB** → 調整視角角度
- 可組合操作，例如 RMB 拖曳同時按 WASD。

4.找回物件：若視角遺失方塊，按 **F** 聚焦。

5.完成後：關閉應用程式並進入下一步。



# USD Explorer Application

- 前面的步驟，我們已使用 **Kit Base Editor 範本** 建立簡單的應用程式，藉此理解 Omniverse 應用程式架構
- 接著，改用 **USD Explorer 範本** 建立應用程式。**USD Explorer** 功能更完整，專為大量 USD 內容的排版與檢視而設計。
- 在 VS Code 開啟終端機，執行以下指令：

➤ **Windows:**

```
.\repo.bat template new
```

➤ **Linux/macOS:**

```
./repo.sh template new
```

- 這次，使用鍵盤上的箭頭鍵選擇「USD Explorer」選項。此選項會彈出一系列關於「設定擴充功能」的問題。

```
? Select with arrow keys what you want to create: Application>
? Select with arrow keys your desired template: [omni_usd_explorer]: USD Explorer
? Enter name of application .kit file [name-spaced, lowercase, alphanumeric]: my_company.my_usd_explorer
? Enter application_display_name: My USD Explorer
? Enter version: 0.1.0
```

The application template you have selected requires a setup extension.

Setup Extension -> omni\_usd\_explorer\_setup

Configuring extension template: Omni USD Explorer Setup

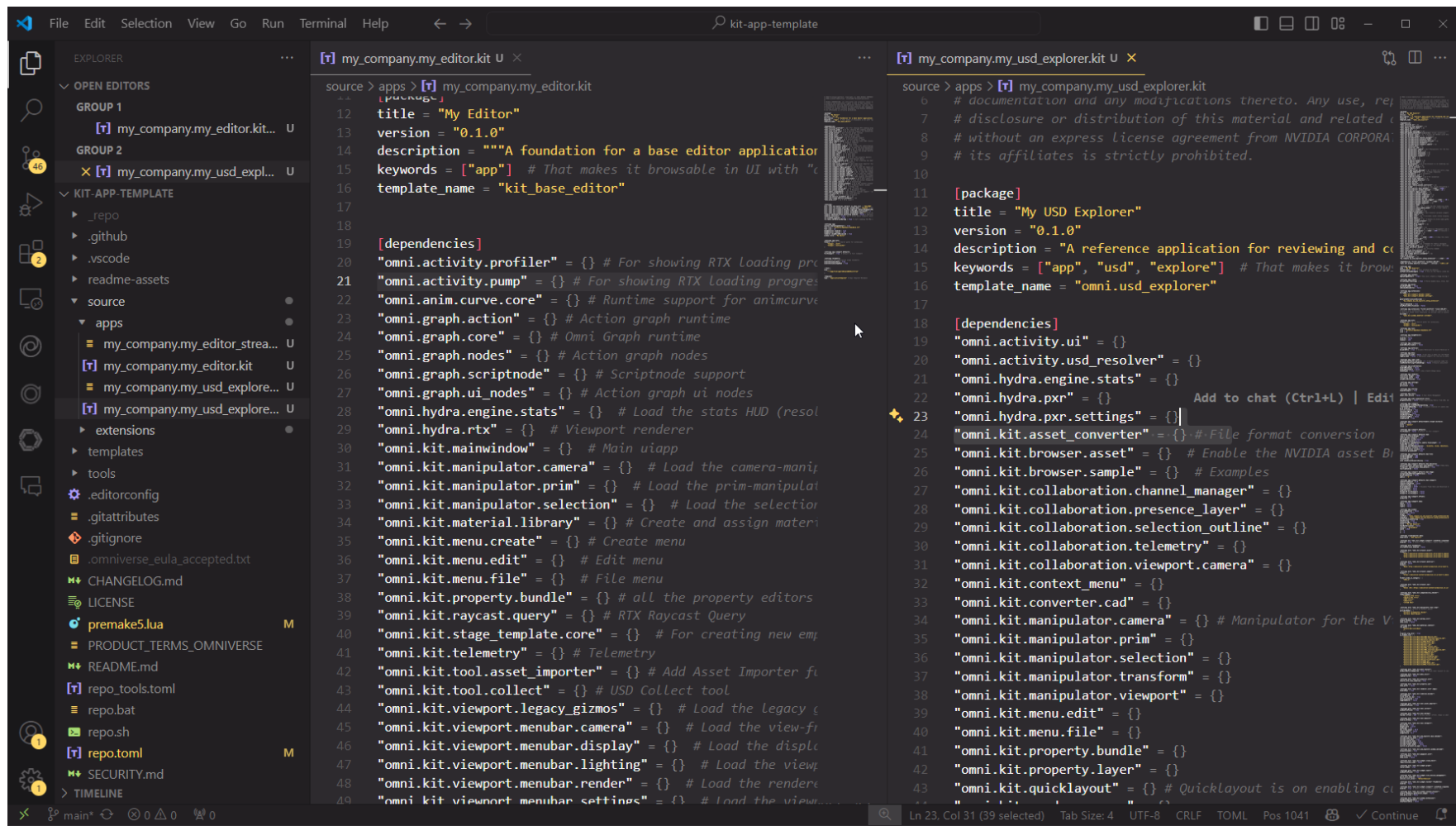
```
? Enter name of extension [name-spaced, lowercase, alphanumeric]: my_company.my_usd_explorer.setup
```

```
? Enter extension_display_name: My USD Explorer Setup Extension
```

```
? Enter version: 0.1.0
```

# Kit Base Editor vs. USD Explorer

- 在 VS Code 中，找到新建立的 .kit 檔案，路徑為：source > apps > my\_company.my\_usd\_explorer.kit
- 接著在編輯器視窗中將它打開。然後，將兩個 .kit 檔案並排排列，這樣你就可以對照比較兩個應用程式的設定。
- 檢查每個 .kit 檔案開頭的 [dependencies] 區段，並注意它們之間的差異：
- 查看每個 .kit 檔案開頭的 [dependencies] 部分，並注意這些部分有所不同。



# Build the USD Explorer

■ 在 **VS Code** 中，打開 **USD Explorer** 專案的終端機，輸入以下指令來建置你剛建立的範本：

➤ **Windows:** `.\repo.bat build`

➤ **Linux/macOS:** `./repo.sh build`

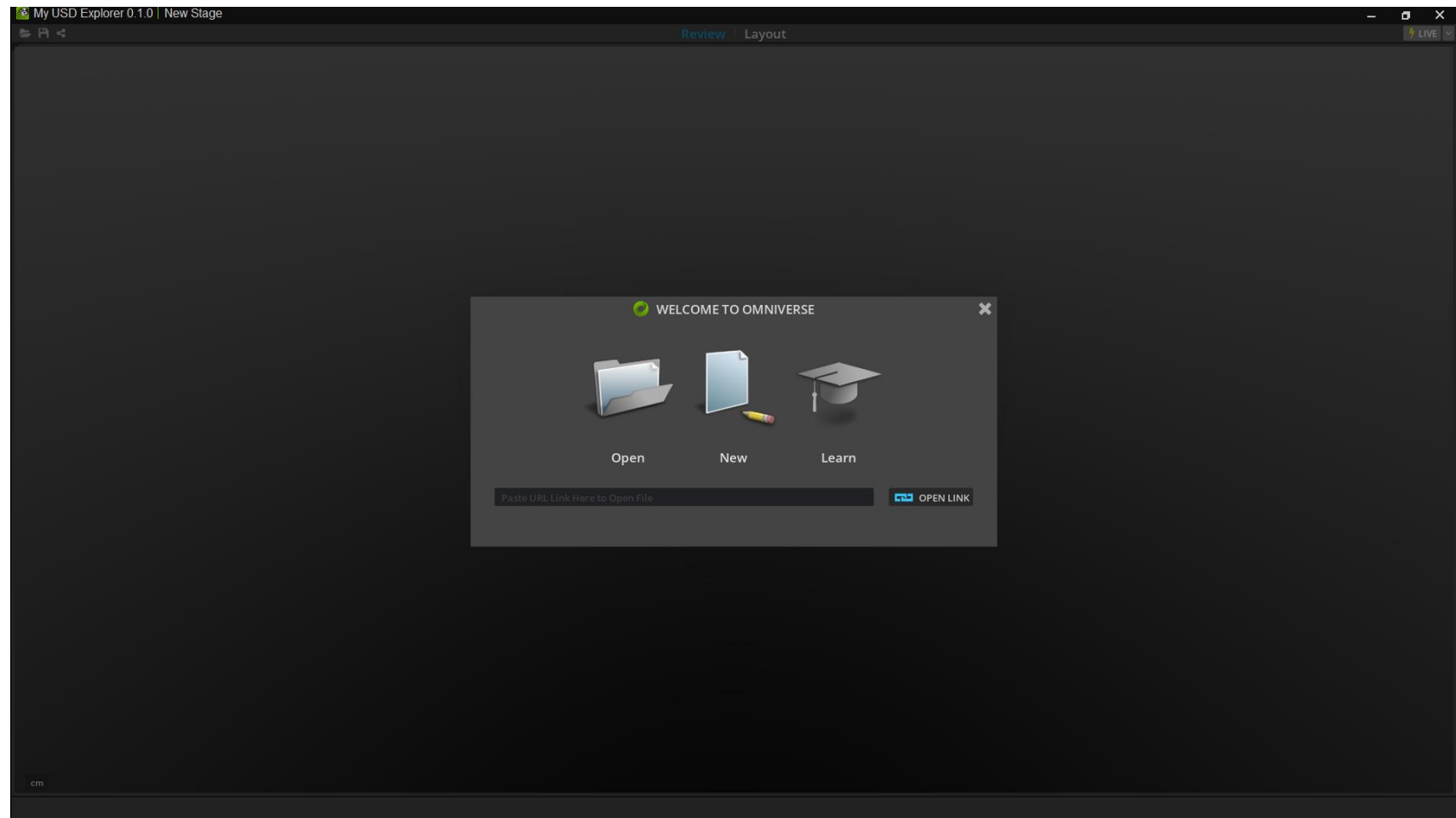
■ 建置完成後，接著輸入啟動指令：

➤ **Windows:** `.\repo.bat launch`

➤ **Linux/macOS:** `./repo.sh launch`

■ 當系統出現提示時，選擇你新建立的 **USD Explorer** 應用程式。

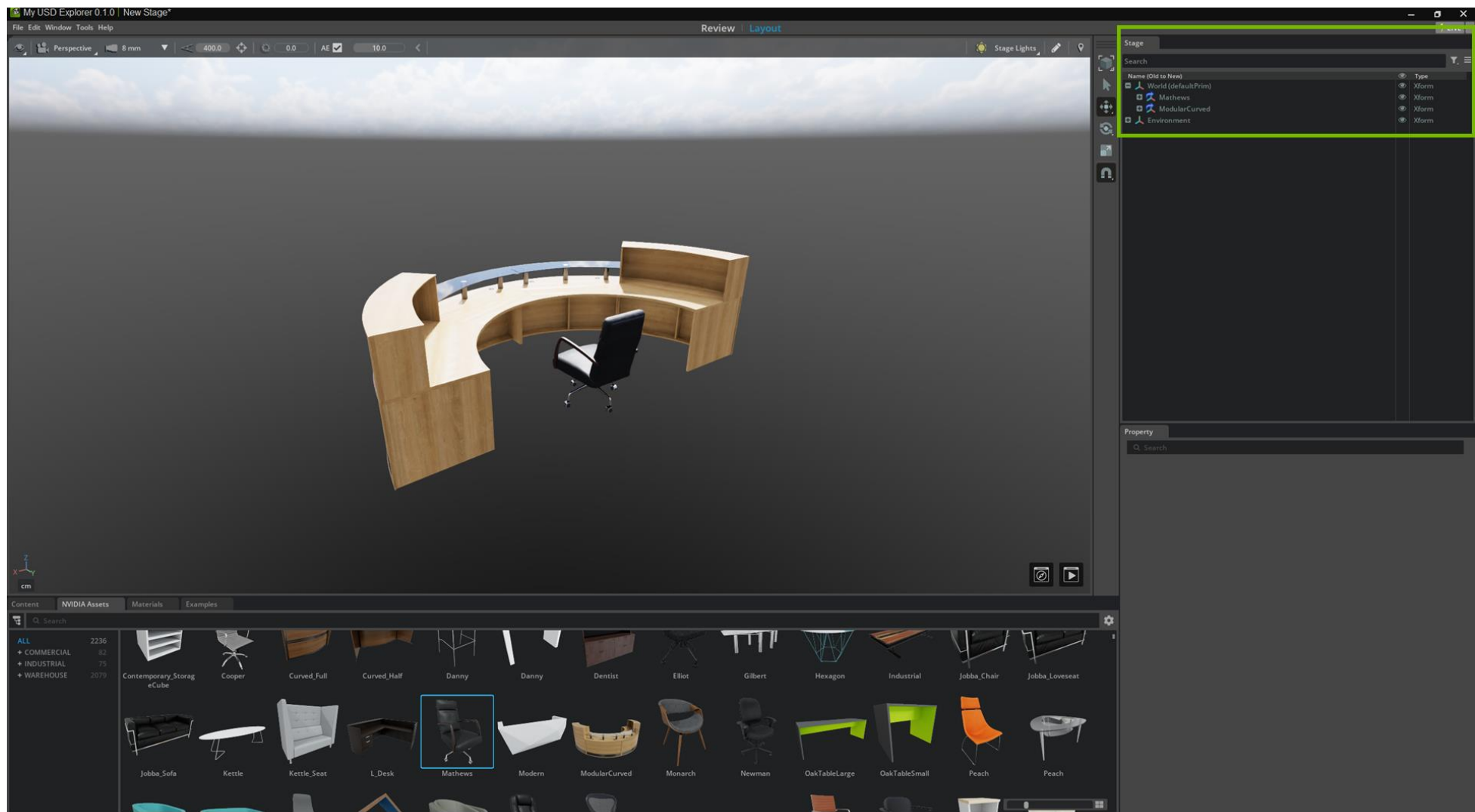
■ 稍等片刻，當啟動流程完成後，應用程式就會在新視窗中開啟。



# Build the USD Explorer

- 選擇 **New** → 進入 **Layout** 模式。
- 左下角點選 **NVIDIA Assets** 分頁。瀏覽資料夾，**雙擊物件** 以開啟，或 **拖曳物件** 到 **Viewport/Stage**。
- 所有載入的資產會列在右側 **Stage** 視窗中。

現在，你已經了解如何使用儲存庫命令從範本建立應用程序，以及如何建置和啟動它們。



# Add an Extension

補充與練習

# 什麼是 Extension ( 擴充功能 ) ？

- **Extension(擴充功能)** 是應用程式功能的獨立單元，可以透過 **Python** 或 **C++** 程式碼來定義。它們是以 **Omniverse Kit** 為基礎所建立**應用程式**的基本組件。
- 應用程式啟動時，會依據 **.kit** 檔案清單 載入對應的擴充功能，形成應用程式的外觀與功能。
- 每個範本都會預設一些擴充功能，而開發者可再加入更多以打造自訂應用程式。
- 更多範例擴充功能可在 **Extensions 文件** 中找到，開發者也能自行建立擴充功能以滿足需求。
- **.kit** 檔案中的 **[dependencies]** 區段 會列出擴充功能。要加入新擴充功能，只需在清單最後新增一行。
- 可在 **Extensions 視窗** 搜尋對應程式碼。