Workflow Matrix for build

Workflow Name	OS	Compiler	Python	Build Flags	Test Flags
Linux_GCC_10_Python39	ubuntu-22.04	gcc, 10	3.9	-DMATERIALX_BUILD_SHARED_LIBS=ON -DMATERIALX_BUILD_MONOLITHIC=ON	
Linux_GCC_14_Python312	ubuntu-24.04	gcc, 14	3.12	None	
Linux_GCC_14_Python313	ubuntu-24.04	gcc, 14	3.13	None	test_render
Linux_GCC_CoverageAnalysis	ubuntu-24.04	gcc, None	None	-DMATERIALX_COVERAGE_ANALYSIS=ON -DMATERIALX_BUILD_RENDER=OFF -DMATERIALX_BUILD_PYTHON=OFF	coverage_analysis
Linux_Clang_13_Python39	ubuntu-22.04	clang, 13	3.9	-DMATERIALX_BUILD_SHARED_LIBS=ON	
Linux_Clang_18_Python313	ubuntu-24.04	clang, 18	3.13	None	clang_format
MacOS_Xcode_15_Python311	macos-14	xcode, 15.4	3.11	-DMATERIALX_BUILD_SHARED_LIBS=ON	
MacOS_Xcode_16_Python313	macos-15	xcode, 16.4	3.13	None	test_shaders
MacOS_Xcode_26_Python313	macos-26	xcode, 26.0	3.13	-DCMAKE_EXPORT_COMPILE_COMMANDS=ON -DMATERIALX_BUILD_DATA_LIBRARY=ON	static_analysis
MacOS_Xcode_DynamicAnalysis	macos-26	xcode, 26.0	None	-DMATERIALX_DYNAMIC_ANALYSIS=ON	dynamic_analysis
iOS_Xcode_26	macos-26	xcode, 26.0	None	-DCMAKE_SYSTEM_NAME=iOS -DCMAKE_OSX_SYSROOT=`xcrun --sdk iphoneos --show-sdk-path` -DCMAKE_OSX_ARCHITECTURES=arm64	
Windows_VS2022_Win32_Python39	windows-2022 x86	Default,	3.9	-G "Visual Studio 17 2022" -A "Win32"	
Windows_VS2022_x64_Python313	windows-2025 x64	Default,	3.13	-G "Visual Studio 17 2022" -A "x64"	test_shaders, extended_build_oiio, extended_build_mdl_sdk
Windows_VS2022_x64_SharedLibs	windows-2025 x64	Default,	None	-G "Visual Studio 17 2022" -A "x64" -DMATERIALX_BUILD_SHARED_LIBS=ON	upload_shaders

Steps For Job: `Build`

1. Sync Repository
2. If (runner.os == 'Linux') - Install Dependencies (Linux)
3. If (runner.os == 'macOS') - Install Dependencies (MacOS)
4. If (runner.os == 'Windows') - Install Dependencies (Windows)
5. If (env.IS_EXTENDED_BUILD == 'true' && *extended_build_oiio* == 'ON' && runner.os == 'Windows') - Install OpenImageIO
6. If (env.IS_EXTENDED_BUILD == 'true' && *extended_build_mdl_sdk* == 'ON' && runner.os == 'Windows') - Install MDL SDK
7. If (*python* != 'None') - Install Python ${{ matrix.python }}
8. If (*python* != 'None') - Install Python Dependencies
9. If (*clang_format* == 'ON') - Run Clang Format
10. CMake Generate
11. CMake Build
12. CMake Unit Tests
13. If (*python* != 'None') - Python Tests
14. If (*test_shaders* == 'ON' && runner.os == 'Windows') - Shader Validation Tests (Windows)
15. If (*test_shaders* == 'ON' && runner.os == 'macOS') - Shader Validation Tests (MacOS)
16. If (*coverage_analysis* == 'ON') - Coverage Analysis Tests
17. If (*static_analysis* == 'ON') - Static Analysis Tests
18. If (*test_render* == 'ON' && runner.os == 'Linux') - Initialize Virtual Framebuffer
19. If (*test_render* == 'ON') - Render Script Tests
20. If (*test_render* == 'ON') - Render Application Tests
21. If (*python* != 'None') - Upload Installed Package
22. If (*clang_format* == 'ON') - Upload Formatted Source

23. If (*upload_shaders* == 'ON') - Upload Reference Shaders
24. If (*test_render* == 'ON') - Upload Renders
25. If (*coverage_analysis* == 'ON') - Upload Coverage Report

Steps For Job: `JavaScript`

1. Sync Repository
2. Install Emscripten
3. Install Node
4. JavaScript CMake Generate
5. JavaScript CMake Build
6. JavaScript Unit Tests
7. Build Web Viewer
8. If (github.event_name != 'pull_request') - Deploy Web Viewer
9. Upload JavaScript Package

Steps For Job: `Python SDist`

1. Sync Repository
2. Install Python
3. Build SDist
4. Upload SDist

Steps For Job: `Python Wheels`

1. Sync Repository
2. Install Python 3.${{ matrix.python-minor }}
3. Download Sdist
4. Build Wheel
5. Install Wheel
6. Python Tests
7. Upload Wheel