Command Line Guide

Updated July 7, 2020 for use with
3DMark Professional Edition v2.12.6964

Installation

When installing the application using a command line the following options are available.

```
3dmark-setup.exe [options]
```

Command	Description
`/installpath=<install path>`	Defines the install path, default is C:\Program Files\Futuremark\3DMark
`/quiet /silent`	Silent install, displays no user interface
`/force`	Force install
`/install`	Installs the product (Default)
`/uninstall`	Uninstalls the product

⚠ Running the installer while using elevated permissions can cause the application to not function correctly. Please avoid running the installer with elevated permissions until prompted.

⚠ When specifying a custom install path, do not use unicode characters in the path to avoid issues.

Usage

Run the program from a command line that was started as an administrator, (right-click on the *cmd* shortcut, and select *Run as Administrator*).

```
3DMarkCmd.exe [options]
```

⚠ 3DMark Professional Edition license required for command line use.

Options

⚠ When *on/off* is omitted with an option, *on* is assumed.

Command	Description
`--runall`	*This command is deprecated as it does not include newer tests. For similar usage you can modify runall.3dmdef to include all desired tests and use --definition=runall.3dmdef - note that command line runs do not prevent you from trying to run a test on hardware that is not compatible with a test and will crash if hardware is not compatible.*
`--definition=<benchmark.xml>`	Name of benchmark definition XML file. Specifies the XML file that defines the tests and settings to be used. See below for XML file example.
`--loop[=<count>]`	Set the number of times to loop benchmark. The default is 1. Use 0 for infinite loop stress test, benchmark will not end until aborted.
`--audio[=on|=off]`	Play audio (default on).
`--systeminfo[=on|=off]`	Collect SystemInfo (default off).
`--systeminfomonitor[=on|=off]`	Enable SystemInfo Monitoring (default off).
`--out=<file.3dmark-result>`	Save results in the file.
`--in=<file.3dmark-result>`	Load results from the file (only for submitting online using --online=on or exporting using --export)
`--online[=on|=off]`	Send results to Futuremark Online (default off).
`--export=<file.xml>`	Export results to the XML file.
`--feature_level=<level>`	Specify DirectX feature level, which can be one of: 9_1, 9_2, 9_3, 10_0, 10_1 or 11_0. Note that

	if you limit this and attempt to run tests that require higher feature level, you will get an error that your hardware doesn't support the required features. Not available for DX12 tests and generally intended only for graphics driver debugging.
`--scalingmode=<mode>`	Specify scaling mode, which can be one of:centered, stretched.
`--log=<log-file>`	Save benchmark progress log to `<log-file>`. Logging does not affect scores. If this option is not used, the last 1000 lines of logging are saved to the default location: `C:\Users\*username*\Documents\3DMark\Log\3DMark.log`
`--debug-log`	Enable per workload debug logging. Log files for each workload run are saved to: `C:\Users\*username*\Documents\3DMark\Logs`
`--register=<product key>`	Register 3DMark with the given key.
`--unregister`	Unregister 3DMark.
`--install=<DLC file path>`	If you have a standalone 3DMark dlc file it can be installed by providing the path to the DLC file
`--uninstall`	Removes all DLCs and settings files
`--path=<arg>`	Changes the destination path for downloaded DLCs and other runtime files
`--language=<arg>`	Changes the language of the application
`--encodedParameters=<arg>`	Passes command line parameters to the application in encoded form
`--trace`	Verbose logging
`--listadapters`	List available graphics adapters and their LUIDs. Useful for determining desired adapter LUID for tests that support manually selecting the GPU - Available in DX12 tests. Apply

	desired LUID to appropriate test-specific .3dmdef file to manually select a GPU to be used when running from the command line.

⚠ You may have problems running Time Spy, Time Spy Extreme, Night Raid, Port Royal, NVIDIA DLSS Feature Test, PCI Express Feature Test and VRS Feature Test from a PowerShell terminal due to the way PowerShell works with full-screen DirectX 12 applications. Use the following Command Line option to work around this issue:

```
start /max 3DMarkCMD.exe
```

Examples

These examples assume that you have a .3dmdef definition file in the same folder as 3DMarkCMD.exe which defines your benchmark run and that you have a C:\my_path\ folder with write permissions. Replace the path as appropriate with the result folder you wish to use.

Example: Run Time Spy, save result file to myresults.3dmark-result

```
3DMarkCmd.exe --definition=timespy.3dmdef
--out=c:\my_path\myresults.3dmark-result
```

Example: Run Time Spy, save result file to myresults.3dmark-result and include hardware information scan and hardware monitoring data in the result file.

```
3DMarkCmd.exe --definition=timespy.3dmdef
--out=c:\my_path\myresults.3dmark-result --systeminfo=on
--systeminfomonitor=on
```

Example: Loop Fire Strike, including demo, indefinitely using default settings.

```
3DMarkCmd.exe --definition=firestrike.3dmdef --loop=0
--out=c:\my_path\myresults.3dmark-result
```

Example: Loop three times with customized "mybenchmark.3dmdef" settings, saving results to myresults.3dmark-result (there will be three numbered result files, one per run)

```
3DMarkCmd.exe --definition=mybenchmark.3dmdef --loop=3
--out=c:\my_path\myresults.3dmark-result
```

Example: Install DLCs downloaded seperately.

```
3DMarkCmd.exe --install="C:\downloads\"
```

Example: Change language to German.

```
3DMarkCmd.exe --language=de-DE
```

Example: Change DLC install path

```
3DMarkCmd.exe --path="D:\3DMarkDlc"
```

Definition XML files

3DMark comes with definition files that enable you to set up and run a benchmark with standard or custom settings. By default, these definitions can be found in:

```
C:\Program Files\UL\3DMark\
```

> ⚠ Definition file names changed with the release of 3DMark v1.3.708, which added the Sky Diver test. You may need to update existing scripts.

apioverhead.3dmdef	Run default API Overhead test
skydiver.3dmdef	Run default Sky Diver test
firestrike.3dmdef	Run default Fire Strike test
firestrike_extreme.3dmdef	Run default Fire Strike Extreme test
firestrike_ultra.3dmdef	Run default Fire Strike Ultra test
timespy.3dmdef	Run default Time Spy test
timespy_extreme.3dmdef	Run default Time Spy Extreme test
nightraid.3dmdef	Run default Night Raid test
portroyal.3dmdef	Run default Port Royal test
nvidiadlss.3dmdef	Run default NVIDIA DLSS test
pciexpress.3dmdef	Run default PCI Express test
vrs.3dmdef	Run default VRS tier 1 test
vrs2.3dmdef	Run default VRS tier 2 test
run_all.3dmdef	Runs all the above tests in sequence[1]
stresstest_firestrike_performance.3dmdef	Run default Fire Strike Stress Test
stresstest_firestrike_extreme.3dmdef	Run default Fire Strike Extreme Stress Test
stresstest_firestrike_ultra.3dmdef	Run default Fire Strike Ultra Stress Test

[1] Fire Strike Ultra is not included in this definition since systems with less than the recommended 3 GB of video card memory may crash when trying to run the test.

stresstest_skydiver_performance.3dmdef	Run default Sky Diver Stress Test
stresstest_timespy_performance.3dmdef	Run default Time Spy Stress Test
stresstest_timespy_extreme.3dmdef	Run default Time Spy Extreme Stress Test
stresstest_nightraid_performance.3dmdef	Run default Night Raid Stress Test
stresstest_portroyal_performance.3dmdef	Run default Port Royal Stress Test
custom_skydiver.3dmdef	Run Sky Diver with custom settings
custom_firestrike.3dmdef	Run Fire Strike with custom settings
custom_firestrike_extreme.3dmdef	Run Fire Strike Extreme with custom settings
custom_firestrike_ultra.3dmdef	Run Fire Strike Ultra with custom settings
custom_timespy.3dmdef	Run Time Spy with custom settings
custom_timespy_extreme.3dmdef	Run Time Spy Extreme with custom settings
custom_nightraid.3dmdef	Run Night Raid with custom settings
custom_portroyal.3dmdef	Run Port Royal with custom settings

Default definitions are the same as running a test from the GUI.

Custom definition files mirror the options available on the Custom tab of the GUI. Copy the appropriate custom definition file and edit it to match your desired settings. Note that custom runs only produce sub-scores, never an overall score.

Example

timespy.3dmdef

```
<?xml version="1.0" encoding="utf-8"?>
<benchmark>
  <application_info>
     <selected_workloads>
     <selected_workload name="TimeSpyDemoP"/>
     <selected_workload name="TimeSpyGt1P"/>
     <selected_workload name="TimeSpyGt2P"/>
     <selected_workload name="TimeSpyCpuP"/>
     </selected_workloads>
  </application_info>
</benchmark>
```

Test names are fairly self-explanatory, for example "TimeSpyDemoP" is the demo, "TimeSpyGt1P" the Graphics test 1 and so on. Note that you will only get an overall score when all sub-tests are included.

If all you want is to specify which tests to run (for example, to skip the Demo), just make a copy of the appropriate definition file and edit the list of tests.

stresstest_firestrike_performance.3dmdef

```
<?xml version="1.0" encoding="utf-8"?>
<benchmark>

  <application_info>
    <selected_workloads>
      <selected_workload name="FireStrikeGt1XST"/>
    </selected_workloads>
  </application_info>

  <settings>
       <setting>
         <name>loop_count</name>
```

```
          <value>20</value>
        </setting>
    </settings>
</benchmark>
```

The stress test definitions function in a similar manner to other definition files but must also contain a valid loop count setting (between 2 and 5000)

Note that when specifying file paths in a custom definition file, for example for outputting frames via image quality tool feature, do not add quotation marks around the path or the test will fail.

Changes to .3dmdef files from v1.3 to v1.4

Setting command line options with .3dmdef files

Some command line settings can now be set within .3dmdef files, for both default and custom benchmark runs.

If the same setting is set from both the command line and the .3dmdef file, the value differing from the default value will be used.

For example, if `--debug-log=off` is specified on the command line, but .3dmdef file contains the code below, then the debug log will be enabled since it is the non-default setting.

```
<setting>
   <name>enable_debug_log</name>
   <value>1</value>
 </setting>
```

The table below lists all settings that can be set within a .3dmdef file.

.3dmdef setting name	.3dmdef default value	command line switch (with default value)
enable_debug_log	0	--debug-log=off
swapchain_height	0 (auto)	
swapchain_width	0 (auto)	
enable_audio	1	--audio=on
enable_systeminfo_monitor	0	--systeminfomonitor=off
enable_systeminfo_collect	0	--systeminfo=off
gpu_count	0 (autodetect)	
scaling_mode	centered	--scalingmode=centered[2]

[2] Using the --scalingmode setting on the command line always overrides the value in .3dmdef file.

Using custom settings

Custom workloads now have separate identifiers. For example, to run Fire Strike with non-default resolution or other settings:

```
<application_info>
  <selected_workloads>
    <selected_workload name="FireStrikeDemoC"/>
    <selected_workload name="FireStrikeGt1C"/>
```

Specifying the "C" version of each workload name is required for the custom settings to have effect. If another name (for example FireStrikePhysicsP) is used, all custom settings that could potentially affect the score will be ignored.

Redundant 'preset' attribute removed

The now redundant 'preset' attribute has been removed.

3DMark v1.3.708

```
<application_info>
  <selected_workloads>
    <selected_workload name="FireStrikeDemoP" preset="default"/>
    <selected_workload name="FireStrikeGt1P" preset="default"/>

<application_info>
  <selected_workloads>
    <selected_workload name="FireStrikeDemoP" preset="custom"/>
    <selected_workload name="FireStrikeGt1P" preset="custom"/>
```

3DMark v1.4.775

```
<application_info>
  <selected_workloads>
    <selected_workload name="FireStrikeDemoP"/>
    <selected_workload name="FireStrikeGt1P"/>

<application_info>
  <selected_workloads>
    <selected_workload name="FireStrikeDemoC"/>
    <selected_workload name="FireStrikeGt1C"/>
```

GPU count setting in .3dmdef now has effect

In 3DMark v1.3 the gpu_count setting in .3dmdef file had no effect. In most sample files it had value 1.

In 3DMark v1.4 the gpu_count setting works as expected. For the majority of cases, the value should be 0 or omitted to enable auto-detection of the number of GPUs.