

We use optional cookies to improve your experience on our websites, such as through social media connections, and to display personalized advertising based on your online activity. If you reject optional cookies, only cookies necessary to provide you the services will be used. You may change your selection by clicking “Manage Cookies” at the bottom of the page. [Privacy Statement](#) [Third-Party Cookies](#)

Accept


Reject

Manage cookies

Microsoft Ignite

Nov 19–22, 2024

Register now >

 | **Learn**

Discover ▾

Product documentation ▾

Development languages ▾

Topics ▾

🔍 Sign in

.NET

Languages ▾

Features ▾

Workloads ▾

APIs ▾

Troubleshooting

Resources ▾

Download .NET

 Filter by title

- .NET fundamentals documentation
- > Get started
- > Overview
- > What's new in .NET
- > Fundamental coding components
- > Runtime libraries
- > Execution model
 - Common Language Runtime (CLR)
 - Managed execution process
- > Assemblies
- > Reflection
- > Dependency loading
- > Versioning
- > Configure .NET Runtime
 - Settings
 - Compilation settings
 - Debugging and profiling settings**
 - Garbage collector settings
 - Globalization settings
 - Networking settings
 - Threading settings
 - WPF settings
- Troubleshoot app launch failures

Learn / .NET /

⊕ ✎ ⋮

Runtime configuration options for debugging and profiling

Article • 12/05/2023 • 3 contributors

👍 Feedback

In this article

- Enable diagnostics
- Enable profiling
- Profiler GUID
- Profiler location
- Show 2 more

This article details the settings you can use to configure .NET debugging and profiling.

⚠️ Note

.NET 6 standardizes on the prefix `DOTNET_` instead of `COMPlus_` for environment variables that configure .NET run-time behavior. However, the `COMPlus_` prefix will continue to work. If you're using a previous version of the .NET runtime, you should still use the `COMPlus_` prefix for environment variables.

Enable diagnostics

- Configures whether the debugger, the profiler, and EventPipe diagnostics are enabled or disabled.
- If you omit this setting, diagnostics are enabled. This is equivalent to setting the value to `1`.

🔗 Expand table

Setting name		Values
runtimeconfig.json	N/A	N/A
Environment variable	<code>COMPlus_EnableDiagnostics</code> or <code>DOTNET_EnableDiagnostics</code>	<code>1</code> - enabled <code>0</code> - disabled

 Download PDF

Enable profiling


- Configures whether profiling is enabled for the currently running process.
- If you omit this setting, profiling is disabled. This is equivalent to setting the value to `0`.

 Expand table

	Setting name	Values
runtimeconfig.json	N/A	N/A
Environment variable	CORECLR_ENABLE_PROFILING	0 - disabled 1 - enabled

Profiler GUID


- Specifies the GUID of the profiler to load into the currently running process.

 Expand table

	Setting name	Values
runtimeconfig.json	N/A	N/A
Environment variable	CORECLR_PROFILER	string-guid

Profiler location

- Specifies the path to the profiler DLL to load into the currently running process (or 32-bit or 64-bit process).
- If more than one variable is set, the bitness-specific variables take precedence. They specify which bitness of profiler to load.
- For more information, see [Finding the profiler library](#).

 Expand table

	Setting name	Values
Environment variable	CORECLR_PROFILER_PATH	string-path
Environment variable	CORECLR_PROFILER_PATH_32	string-path
Environment variable	CORECLR_PROFILER_PATH_64	string-path

Export perf maps and jit dumps

- Enables or disables selective enablement of perf maps or jit dumps. These files allow third party tools, such as the Linux `perf` tool, to identify call sites for dynamically generated code and precompiled ReadyToRun (R2R) modules.
- If you omit this setting, writing perf map and jit dump files are both disabled. This is equivalent to setting the value to `0`.
- When perf maps are disabled, not all managed callsites will be properly resolved.
- Depending on the Linux kernel version, both formats are supported by the `perf` tool.
- Enabling perf maps or jit dumps causes a 10-20% overhead. To minimize performance impact, it's recommended to selectively enable either perf maps or jit dumps, but not both.

The following table compares perf maps and jit maps.

[Expand table](#)

Format	Description	Supported on
<i>Perf maps</i>	Emits <code>/tmp/perf-<pid>.map</code> , which contains symbolic information for dynamically generated code. Emits <code>/tmp/perfinfo-<pid>.map</code> , which includes ReadyToRun (R2R) module symbol information and is used by PerfCollect .	Perf maps are supported on all Linux kernel versions.
<i>Jit dumps</i>	The jit dump format supersedes perf maps and contains more detailed symbolic information. When enabled, jit dumps are output to <code>/tmp/jit-<pid>.dump</code> files.	Linux kernel versions 5.4 or higher.

[Expand table](#)

Setting name		Values
runtimeconfig.json	N/A	N/A
Environment variable	<code>COMPlus_PerfMapEnabled</code> or <code>DOTNET_PerfMapEnabled</code>	<code>0</code> - disabled <code>1</code> - perf maps and jit dumps both enabled <code>2</code> - jit dumps enabled <code>3</code> - perf maps enabled

Perf log markers

- Enables or disables the specified signal to be accepted and ignored as a marker in the perf logs.
- If you omit this setting, the specified signal is not ignored. This is equivalent to setting the value to `0`.

[Expand table](#)

Setting name		Values
runtimeconfig.json	N/A	N/A
Environment variable	<code>COMPlus_PerfMapIgnoreSignal</code> or <code>DOTNET_PerfMapIgnoreSignal</code>	<code>0</code> - disabled <code>1</code> - enabled

ⓘ Note

This setting is ignored if [DOTNET_PerfMapEnabled](#) is omitted or set to `0` (that is, disabled).

Collaborate with us on GitHub

The source for this content can be found on GitHub, where you can also create and review issues and pull requests. For more information, see [our contributor guide](#).



.NET feedback

.NET is an open source project. Select a link to provide feedback:

 [Open a documentation issue](#)

 [Provide product feedback](#)

