


814 lines (746 loc) · 75.6 KB

CodeBlame

Raw

```
1 // Licensed to the .NET Foundation under one or more agreements.
2 // The .NET Foundation licenses this file to you under the MIT license.
3 // See the LICENSE file in the project root for more information.
4 //*****
5 // CLRConfigValues.h
6 //
7
8 //
9 // Unified method of accessing configuration values from environment variables,
10 // registry and config file.
11 //
12 //*****
13
14 // IMPORTANT: Before adding a new config value, please read up on naming conventions (see
15 // code:#NamingConventions)
16 //
17 // =====
18 // CONTENTS
19 // =====
20 // * How to define config values (see code:#Define)
21 // * How to access config values (see code:#Access)
22 // * Naming conventions (see code:#NamingConventions)
23 //
24 //
25 // =====
26 // #Define - Use one of the following macros to define config values. (See code:#DWORDS and code:#S
```

```
27 // =====
28 //
29 // By default, all macros are DEBUG ONLY. Add the "RETAIL_" prefix to make the config value available
30 //
31 // #DWORDs:
32 // -----
33 // CONFIG_DWORD_INFO(symbol, name, defaultValue, description)
34 // -----
35 // Use this macro to define a basic DWORD value. CLRConfig will look in environment variables (adding
36 // COMPlus_ to the name), the registry (HKLM and HKCU), and all the config files for this value. To
37 // where CLRConfig looks, use the extended version of the macro below. IMPORTANT: please follow the
38 // code:NamingConventions for the symbol and the name!
39 //
40 // Example: CONFIG_DWORD_INFO(INTERNAL_AllowCrossModuleInlining, W("AllowCrossModuleInlining"), 0,
41 //
42 // -----
43 // CONFIG_DWORD_INFO_EX(symbol, name, defaultValue, description, lookupOptions)
44 // -----
45 // To customize where CLRConfig looks to get a DWORD, use the extended (_EX) version of the macro.
46 // of options and their descriptions, see code:CLRConfig.LookupOptions
47 //
48 // Example: CONFIG_DWORD_INFO_EX(INTERNAL_EnableInternetHREFexes, W("EnableInternetHREFexes"), 0,
49 // (CLRConfig::LookupOptions) (CLRConfig::IgnoreEnv | CLRConfig::IgnoreHKCU))
50 //
51 // #Strings:
52 // -----
53 // CONFIG_STRING_INFO(symbol, name, description)
54 // -----
55 // Defines a string value. Same rules apply as DWORDs.
56 //
57 // -----
58 // CONFIG_STRING_INFO_EX(symbol, name, description, lookupOptions)
59 // -----
60 // Extended version of the String macro. Again, similar to the DWORD extended macro.
61 //
62 //
63 // =====
64 // #Access - Use the following overloaded method to access config values.
65 // =====
66 // From anywhere, use CLRConfig::GetConfigValue(CLRConfig::<symbol>) to access any value defined in
67 // file.
68 //
69 //
70 // =====
71 // #NamingConventions
72 // =====
```

```
73 // -----
74 // #Symbol - used to access values from the source. (using CLRConfig::<symbol>)
75 // -----
76 // The symbol for each config value is named as such:
77 // ### <Class>_<feature area>_<name> ###
78 //
79 // <Class> indicates which of the following buckets the value is in:
80 // * INTERNAL ? this value is for internal (CLR team) use only
81 // * UNSUPPORTED ? this value is available to partners/developers, but is not officially supported
82 // * EXTERNAL ? this value is available for anyone to use and is publicly documented
83 //
84 // Examples:
85 // * INTERNAL_Security_FullAccessChecks
86 // * UNSUPPORTED_Security_DisableTransparency
87 // * EXTERNAL_Security_LegacyHMACMode
88 //
89 // -----
90 // #Name - the name of the registry value or environment variable that CLRConfig looks up.
91 // -----
92 // The name of each value is the same as the symbol, with one exception. Names of external values c
93 // contain the EXTERNAL prefix.
94 //
95 // For compatibility reasons, current names do not follow the convention.
96 //
97 // Examples:
98 // * W("INTERNAL_Security_FullAccessChecks")
99 // * W("UNSUPPORTED_Security_DisableTransparency")
100 // * W("Security_LegacyHMACMode") <----- (No EXTERNAL prefix)
101
102 ///
103 /// AppDomain
104 ///
105 CONFIG_DWORD_INFO(INTERNAL_ADDumpSB, W("ADDumpSB"), 0, "Not used")
106 CONFIG_DWORD_INFO(INTERNAL_ADForceSB, W("ADForceSB"), 0, "Forces sync block creation for all object
107 CONFIG_DWORD_INFO(INTERNAL_ADLogMemory, W("ADLogMemory"), 0, "Superseded by test hooks")
108 CONFIG_DWORD_INFO(INTERNAL_ADTakeDHSnapShot, W("ADTakeDHSnapShot"), 0, "Superseded by test hooks")
109 CONFIG_DWORD_INFO(INTERNAL_ADTakeSnapShot, W("ADTakeSnapShot"), 0, "Superseded by test hooks")
110 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_EnableFullDebug, W("EnableFullDebug"), "Heavy-weight check
111
112 ///
113 /// Jit Pitching
114 ///
115 RETAIL_CONFIG_DWORD_INFO(INTERNAL_JitPitchEnabled, W("JitPitchEnabled"), (DWORD)0, "Set it to 1 to
116 RETAIL_CONFIG_DWORD_INFO(INTERNAL_JitPitchMemThreshold, W("JitPitchMemThreshold"), (DWORD)0, "Do Ji
117 RETAIL_CONFIG_DWORD_INFO(INTERNAL_JitPitchMethodSizeThreshold, W("JitPitchMethodSizeThreshold"), (D
118 RETAIL_CONFIG_DWORD_INFO(INTERNAL_JitPitchTimeInterval, W("JitPitchTimeInterval"), (DWORD)0, "Time
```

```
110 #define _CONFIG_DEBUG_INFO (INTERNAL_STACK_FRAME_SIZE > 0 || \ STACK_FRAME_SIZE > 0) /* false
```



```
741 // DO NOT ADD ANY MORE CONFIG SWITCHES TO THIS SECTION!
742 // **
743 CONFIG_DWORD_INFO_EX(INTERNAL_ActivatePatchSkip, W("ActivatePatchSkip"), 0, "Allows an assert when
744 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_AlwaysUseMetadataInterfaceMapLayout, W("AlwaysUseMetadataI
745 CONFIG_DWORD_INFO(INTERNAL_AssertOnUnneededThis, W("AssertOnUnneededThis"), 0, "While the ConfigDWC
746 CONFIG_DWORD_INFO_EX(INTERNAL_AssertStacktrace, W("AssertStacktrace"), 1, "", CLRConfig::EEConfig_c
747 CONFIG_DWORD_INFO_EX(INTERNAL_clearNativeImageStress, W("clearNativeImageStress"), 0, "", CLRConfig
748 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_CPUFamily, W("CPUFamily"), "")
749 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_CPUFeatures, W("CPUFeatures"), "")
750 RETAIL_CONFIG_DWORD_INFO_EX(EXTERNAL_DisableConfigCache, W("DisableConfigCache"), 0, "Used to disab
751 RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(EXTERNAL_DisableStackwalkCache, W("DisableStackwalkCache"),
752 RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(UNSUPPORTED_DoubleArrayToLargeObjectHeap, W("DoubleArrayToLa
753 CONFIG_STRING_INFO(INTERNAL_DumpOnClassLoad, W("DumpOnClassLoad"), "Dumps information about loaded
754 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_ExpandAllOnLoad, W("ExpandAllOnLoad"), "")
755 CONFIG_STRING_INFO_DIRECT_ACCESS(INTERNAL_ForcedRuntime, W("ForcedRuntime"), "Verify version of CLR
756 CONFIG_DWORD_INFO_EX(INTERNAL_ForceRelocs, W("ForceRelocs"), 0, "", CLRConfig::EEConfig_default)
757 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_GenerateLongJumpDispatchStubRatio, W("GenerateLongJumpDisp
758 CONFIG_DWORD_INFO_EX(INTERNAL_HashStack, W("HashStack"), 0, "", CLRConfig::EEConfig_default)
759 CONFIG_DWORD_INFO(INTERNAL_HostManagerConfig, W("HostManagerConfig"), (DWORD)-1, "")
```

```
760 CONFIG_DWORD_INFO(INTERNAL_HostTestThreadAbort, W("HostTestThreadAbort"), 0, "")
761 RETAIL_CONFIG_DWORD_INFO_EX(UNSUPPORTED_IgnoreDllMainReturn, W("IgnoreDllMainReturn"), 0, "Don't ch
762 CONFIG_STRING_INFO(INTERNAL_InvokeHalt, W("InvokeHalt"), "Throws an assert when the given method is
763 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_MaxStackDepth, W("MaxStackDepth"), "")
764 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_MaxStubUnwindInfoSegmentSize, W("MaxStubUnwindInfoSegments
765 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_MaxThreadRecord, W("MaxThreadRecord"), "")
766 CONFIG_DWORD_INFO(INTERNAL_MessageDebugOut, W("MessageDebugOut"), 0, "")
767 RETAIL_CONFIG_DWORD_INFO_EX(EXTERNAL_NativeImageRequire, W("NativeImageRequire"), 0, "", CLRConfig:
768 CONFIG_DWORD_INFO_EX(INTERNAL_NestedEhOom, W("NestedEhOom"), 0, "", CLRConfig::EEConfig_default)
769 #define INTERNAL_NoGuiOnAssert_Default 1
770 RETAIL_CONFIG_DWORD_INFO_EX(INTERNAL_NoGuiOnAssert, W("NoGuiOnAssert"), INTERNAL_NoGuiOnAssert_Def
771 RETAIL_CONFIG_DWORD_INFO_EX(EXTERNAL_NoProcedureSplitting, W("NoProcedureSplitting"), 0, "", CLRCon
772 CONFIG_DWORD_INFO_EX(INTERNAL_NoStringInterning, W("NoStringInterning"), 1, "Disallows string inter
773 RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(EXTERNAL_NotifyBadAppCfg, W("NotifyBadAppCfg"), "Whether to
774 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_PauseOnLoad, W("PauseOnLoad"), "Stops in SystemDomain::ini
775 CONFIG_DWORD_INFO(INTERNAL_PerfAllocsSizeThreshold, W("PerfAllocsSizeThreshold"), 0x3FFFFFFF, "Log
776 CONFIG_DWORD_INFO(INTERNAL_PerfNumAllocsThreshold, W("PerfNumAllocsThreshold"), 0x3FFFFFFF, "Log fa
777 CONFIG_STRING_INFO(INTERNAL_PerfTypesToLog, W("PerfTypesToLog"), "Log facility LF_GCALLOC logs obje
778 RETAIL_CONFIG_DWORD_INFO(EXTERNAL_Prepopulate1, W("Prepopulate1"), 1, "")
779 CONFIG_STRING_INFO(INTERNAL_PrestubGC, W("PrestubGC"), "")
780 CONFIG_STRING_INFO(INTERNAL_PrestubHalt, W("PrestubHalt"), "")
781 RETAIL_CONFIG_STRING_INFO(EXTERNAL_RestrictedGCStressExe, W("RestrictedGCStressExe"), "")
782 CONFIG_DWORD_INFO_EX(INTERNAL_ReturnSourceTypeForTesting, W("ReturnSourceTypeForTesting"), 0, "Allc
783 RETAIL_CONFIG_DWORD_INFO_EX(UNSUPPORTED_RSStressLog, W("RSStressLog"), 0, "Allows turning on loggin
784 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_SaveThreadInfo, W("SaveThreadInfo"), "")
785 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_SaveThreadInfoMask, W("SaveThreadInfoMask"), "")
786 CONFIG_DWORD_INFO(INTERNAL_SBDumpOnNewIndex, W("SBDumpOnNewIndex"), 0, "Used for Syncblock debuggin
787 CONFIG_DWORD_INFO(INTERNAL_SBDumpOnResize, W("SBDumpOnResize"), 0, "Used for Syncblock debugging. I
788 CONFIG_DWORD_INFO(INTERNAL_SBDumpStyle, W("SBDumpStyle"), 0, "Used for Syncblock debugging. It's be
789 RETAIL_CONFIG_STRING_INFO_DIRECT_ACCESS(UNSUPPORTED_ShimDatabaseVersion, W("ShimDatabaseVersion"),
790 RETAIL_CONFIG_DWORD_INFO(UNSUPPORTED_SleepOnExit, W("SleepOnExit"), 0, "Used for lrak detection. I'
791 CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_StubLinkerUnwindInfoVerificationOn, W("StubLinkerUnwindInf
792 RETAIL_CONFIG_DWORD_INFO_EX(UNSUPPORTED_SuccessExit, W("SuccessExit"), 0, "", CLRConfig::EEConfig_c
793 RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(EXTERNAL_SymbolReadingPolicy, W("SymbolReadingPolicy"), "Spe
794 RETAIL_CONFIG_DWORD_INFO(UNSUPPORTED_TestDataConsistency, W("TestDataConsistency"), FALSE, "Allows
795 RETAIL_CONFIG_DWORD_INFO_EX(EXTERNAL_ThreadGuardPages, W("ThreadGuardPages"), 0, "", CLRConfig::EEC
796 RETAIL_CONFIG_DWORD_INFO_EX(EXTERNAL_Timeline, W("Timeline"), 0, "", CLRConfig::EEConfig_default)
797 RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(UNSUPPORTED_TotalStressLogSize, W("TotalStressLogSize"), "To
798
799 #ifdef _DEBUG
800 RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(EXTERNAL_TraceIUnknown, W("TraceIUnknown"), "")
801 RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(EXTERNAL_TraceWrap, W("TraceWrap"), "")
802 #endif
803
804 RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(EXTERNAL_TURNOFFDEBUGINFO, W("TURNOFFDEBUGINFO"), "")
805 RETAIL_CONFIG_DWORD_INFO_EX(EXTERNAL_UseMethodDataCache, W("UseMethodDataCache"), FALSE, "Used during
```

```
805     RETAIL_CONFIG_DWORD_INFO(EXTERNAL_UseParentMethodData, W("UseParentMethodData"), TRUE, "Used during  
806     RETAIL_CONFIG_DWORD_INFO(EXTERNAL_UseParentMethodData, W("UseParentMethodData"), TRUE, "Used during  
807     CONFIG_DWORD_INFO_DIRECT_ACCESS(INTERNAL_VerifierOff, W("VerifierOff"), "")  
808     RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(EXTERNAL_VerifyAllOnLoad, W("VerifyAllOnLoad"), "")  
809     // **  
810     // PLEASE MOVE ANY CONFIG SWITCH YOU OWN OUT OF THIS SECTION INTO A CATEGORY ABOVE  
811     //  
812     // DO NOT ADD ANY MORE CONFIG SWITCHES TO THIS SECTION!  
813     // **  
814     //-----
```