Sign in

dotnet / runtime Public

Notifications | Fork 4.7k | Star 15.2k

Code | Issues 5k+ | Pull requests 324 | Discussions | Actions | Projects 6 | Securi

runtime / src / coreclr / src / inc / clrconfig.h

215 lines (183 loc) · 8.83 KB

Code | Blame | Raw

```
// Licensed to the .NET Foundation under one or more agreements.
// The .NET Foundation licenses this file to you under the MIT license.
// See the LICENSE file in the project root for more information.
//
// ----------------------------------------------------------------------------------------------
// CLRConfig.h
//

//
// Unified method of accessing configuration values from environment variables, registry and config
// This class replaces all GetConfigDWORD and GetConfigString methods in EEConfig and REGUTIL. To c
// flag, add an entry in the table in file:CLRConfigValues.h.
//
//
//
//
// ----------------------------------------------------------------------------------------------


#include "utilcode.h"
#include "holder.h"

#ifndef __CLRConfig_h__
#define __CLRConfig_h__

class CLRConfig
```

```
{
public:
    //
    // Types
    //

    // Setting each option results in some change to the config value lookup method. Default behavi
    // the following order):
    // * Look at environment variables (prepending COMPlus to the name)
    // * Look at the framework registry keys (HKCU\Software\Microsoft\.NETFramework then
    //     HKLM\Software\Microsoft\.NETFramework)
    // * Look at the available config files (system, application, host and user). For details see T
    //     Link to BOTR documentation
    enum LookupOptions {
        // If set, don't look in environment variables.
        IgnoreEnv = 0x1,
        // If set, do not prepend "COMPlus_" when doing environment variable lookup.
        DontPrependCOMPlus_ = 0x2,
        // If set, don't look in HKLM in the registry.
        IgnoreHKLM = 0x4,
        // If set, don't look in HKCU in the registry.
        IgnoreHKCU = 0x8,
        // If set, look only in the system config file, ignoring other config files.
        // (This option does not affect environment variable and registry lookups)
        ConfigFile_SystemOnly = 0x40,
        // If set, reverse the order of config file lookups (application config file first).
        // (This option does not affect environment variable and registry lookups)
        ConfigFile_ApplicationFirst = 0x80,
        // Remove any whitespace at beginning and end of value.  (Only applicable for
        // *string* configuration values.)
        TrimWhiteSpaceFromStringValue = 0x100,

        // Legacy EEConfig-style lookup.
        EEConfig_default = 0,
    };

    // Struct used to store information about where/how to find a Config DWORD.
    // NOTE: Please do NOT create instances of this struct. Use the macros in file:CLRConfigValues.
    typedef struct ConfigDWORDInfo
    {
        LPCWSTR name;
        DWORD defaultValue;
        LookupOptions options;
    } ConfigDWORDInfo;

    // Struct used to store information about where/how to find a Config String.
    // NOTE: Please do NOT create instances of this struct. Use the macros in file:CLRConfigValues.
    typedef struct ConfigStringInfo
    {
        LPCWSTR name;
        LookupOptions options;
    } ConfigStringInfo;

    //
    // Declaring structs using the macro table in CLRConfigValues.h
    //

    // These macros declare ConfigDWORDInfo structs.
    #define RETAIL_CONFIG_DWORD_INFO(symbol, name, defaultValue, description) \
        static const ConfigDWORDInfo symbol;
    #define RETAIL_CONFIG_DWORD_INFO_EX(symbol, name, defaultValue, description, lookupOptions) \
        static const ConfigDWORDInfo symbol;

    // These macros declare ConfigStringInfo structs.
    #define RETAIL_CONFIG_STRING_INFO(symbol, name, description) \
        static const ConfigStringInfo symbol;
    #define RETAIL_CONFIG_STRING_INFO_EX(symbol, name, description, lookupOptions) \
        static const ConfigStringInfo symbol;

    // TEMPORARY macros that declare strings. These are used for config value accesses that haven't
    // moved over to CLRConfig yet. Once all accesses have been moved, these macros (and correspond
    // instantiations in file:../utilcode/CLRConfig.cpp) should be removed.
    #define RETAIL_CONFIG_DWORD_INFO_DIRECT_ACCESS(symbol, name, description) \
        static const LPCWSTR symbol;
    #define RETAIL_CONFIG_STRING_INFO_DIRECT_ACCESS(symbol, name, description) \
        static const LPCWSTR symbol;

    //
    // Debug versions of the macros
    //
#ifdef _DEBUG
    #define CONFIG_DWORD_INFO(symbol, name, defaultValue, description) \
        static const ConfigDWORDInfo symbol;
    #define CONFIG_DWORD_INFO_EX(symbol, name, defaultValue, description, lookupOptions) \
        static const ConfigDWORDInfo symbol;
    #define CONFIG_STRING_INFO(symbol, name, description) \
        static const ConfigStringInfo symbol;
    #define CONFIG_STRING_INFO_EX(symbol, name, description, lookupOptions) \
        static const ConfigStringInfo symbol;
    #define CONFIG_DWORD_INFO_DIRECT_ACCESS(symbol, name, description) \
        static const LPCWSTR symbol;
    #define CONFIG_STRING_INFO_DIRECT_ACCESS(symbol, name, description) \
    #define CONFIG_STRING_INFO_DIRECT_ACCESS(symbol, name, description) \

    #undef CONFIG_STRING_INFO_DIRECT_ACCESS

    //
    // Methods to do config value (DWORD and String) lookups.
    //
    static BOOL IsConfigEnabled(const ConfigDWORDInfo & info);

    // Look up a DWORD config value.
    static DWORD GetConfigValue(const ConfigDWORDInfo & info);

    // Look up a DWORD config value.
    static DWORD GetConfigValue(const ConfigDWORDInfo & info, bool acceptExplicitDefaultFromRegutil

    // Look up a string config value.
    // You own the string that's returned.
    static LPWSTR GetConfigValue(const ConfigStringInfo & info);

    // Look up a string config value, passing it out through a pointer reference. Reports out of me
    // errors (HRESULT E_OUTOFMEMORY).
    // You own the string that's returned.
    static HRESULT GetConfigValue(const ConfigStringInfo & info, __deref_out_z LPWSTR * outVal);

    //
    //
    // Check whether an option is specified (e.g. explicitly listed) in any of the CLRConfig
    // locations: environment or registry (with or without COMPlus_) or any config file.
    // The result is therefore a conservative approximation (some settings do not actually
    // take effect everywhere and no setting is valid both with and without COMPlus_)
    //
    static BOOL IsConfigOptionSpecified(LPCWSTR name);

    // Free a string returned by GetConfigValue
    static void   FreeConfigString(__in __in_z LPWSTR name);

private:

    // Helper method to translate LookupOptions to REGUTIL::CORConfigLevel
    static REGUTIL::CORConfigLevel GetConfigLevel(LookupOptions options);

    //
    // Helper methods.
    //

    // Helper method to check if a certain option is set in a ConfigDWORDInfo struct.
    static inline BOOL CheckLookupOption(const ConfigDWORDInfo & info, LookupOptions option)
    {
        LIMITED_METHOD_CONTRACT;
        return ((info.options & option) == option) ? TRUE : FALSE;
    }

    // Helper method to check if a certain option is set in a ConfigStringInfo struct.
    static inline BOOL CheckLookupOption(const ConfigStringInfo & info, LookupOptions option)
    {
        LIMITED_METHOD_CONTRACT;
        return ((info.options & option) == option) ? TRUE : FALSE;
    }

    // Helper method to check if a certain option is set in an options enum.
    static inline BOOL CheckLookupOption(LookupOptions infoOptions, LookupOptions optionToCheck)
    {
        LIMITED_METHOD_CONTRACT;
        return ((infoOptions & optionToCheck) == optionToCheck) ? TRUE : FALSE;
    }

    static HRESULT TrimWhiteSpace(LPCWSTR wszOrig, __deref_out_z LPWSTR * pwszTrimmed);
};

inline CLRConfig::LookupOptions operator|(CLRConfig::LookupOptions lhs, CLRConfig::LookupOptions rh
{
    return static_cast<CLRConfig::LookupOptions>(static_cast<DWORD>(lhs) | static_cast<DWORD>(rhs))
}

typedef Wrapper<LPWSTR, DoNothing, CLRConfig::FreeConfigString, NULL> CLRConfigStringHolder;

#endif //__CLRConfig_h__
```