Subject: .ini file helpers
Posted by mirek on Thu, 28 Jun 2012 22:38:55 GMT
View Forum Message <> Reply to Message

INI_BOOL, INI_STRING, INI_INT now can be used to encapsulate reading parameters from .ini file (while also providing overview and info about parameters), as demonstrated by reference/INI example.

```
#include <Core/Core.h>

using namespace Upp;

namespace Config {

INI_BOOL(flag1, false, "This is bool parameter 1")
INI_BOOL(flag2, true, "This is bool parameter 2")

INI_STRING(text, "default text", "Text parameter");

INI_INT(number, 123456, "Number parameter");

};

CONSOLE_APP_MAIN
{
 DDUMP(Config::flag1);
 DDUMP(Config::flag2);
 DDUMP(Config::text);
 DDUMP(Config::number);

 LOG(GetIniInfoFormatted());

 Config::number = 321;
 DDUMP(Config::number);

 LOG(GetIniInfoFormatted());
}
```

Subject: Re: .ini file helpers
Posted by mirek on Tue, 03 Jul 2012 07:56:59 GMT
View Forum Message <> Reply to Message

After some initial experiences, we are now using "namespace Ini" instead of "namespace Config", also classes used to represent flags are now public, so that they can be placed in .h:

In .cpp:

```
namespace Ini {
INI_BOOL(flag1, false, "This is bool parameter 1");
};
```

In .h:

```
namespace Ini {
Bool flag1;
};
```

Subject: Re: .ini file helpers
Posted by dolik.rce on Tue, 10 Jul 2012 19:14:42 GMT
View Forum Message <> Reply to Message

Hi Mirek,

Today I tried to use the Ini* helper functions, but I found out that they are not very intuitive. I believe that with a little work, they might become much more useful.

I see these problems:
1) INI_DOUBLE is missing.

2) Once you call GetIniKey, the file is loaded and it is not possible to reload configuration. In a daemons/services, it is often desirable to reload config without restarting.

3) There are currently two separate mechanisms SetIniFile+GetIniKey function and the Ini namespace with INI_* macros. I didn't notice any connection between those two, but they are both shown in the reference/INI example.

Now tell me where I'm wrong If I'm not, then I think all these problems are solvable.

The first one is trivial.

The second could be solved simply by using global static VectorMap for the key-value pairs and adding ReloadIniFile() function. Something like:static StaticMutex sMtx;
static char sIniFile[256];
static bool s_ini_loaded;
static VectorMap<String, String> s_ini_key;

```
void SetIniFile(const char *name) {
 Mutex::Lock __(sMtx);
 strcpy(sIniFile, name);
}

void ReloadIniFile(const char *name) {
 Mutex::Lock __(sMtx);
 if(*name)
  strcpy(sIniFile, name);
 s_ini_loaded = true;
 s_ini_key = LoadIniFile(*sIniFile ? sIniFile : ~ConfigFile("q.ini"));
 #ifdef PLATFORM_WIN32
  if(s_ini_key.GetCount() == 0)
   s_ini_key = LoadIniFile(~GetExeDirFile("q.ini"));
  if(s_ini_key.GetCount() == 0)
   s_ini_key = LoadIniFile("c:\\q.ini");
 #endif
 #ifdef PLATFORM_POSIX
  if(s_ini_key.GetCount() == 0)
   s_ini_key = LoadIniFile(GetHomeDirFile("q.ini"));
 #endif
}

String GetIniKey(const char *id, const String& def) {
 if(!s_ini_loaded)
  ReloadIniFile(sIniFile);
 return s_ini_key.Get(id, def);
}
```

But I think there is better solution and it is related to the problem 3). I propose to drop the GetIniKey function altogether (or keep it just as wrapper for backward compatibility). If we rewrite LoadIniStream to guess value types (not very difficult) and store them as Value instead of String AND add a reference to original Ini::IniXYZ variable into the IniInfo struct, we could iterate through IniInfos after each reload and update the values of the variables in the Ini namespace. In this way, the values declared in code would be possible to overwrite by values from INI file, even at multiple times if desired. Also, GetIniKey would be unnecessary, as Ini::MyVar would be always up to date and shorter to write.

I'm not sure if I expressed myself clear enough, ask if there is anything non-understandable or if you see any unsolvable problem within this proposal...

Best regards,
Honza

Subject: Re: .ini file helpers

dolik.rce wrote on Tue, 10 July 2012 15:14Hi Mirek,

Today I tried to use the Ini* helper functions, but I found out that they are not very intuitive. I believe that with a little work, they might become much more useful.

I see these problems:
1) INI_DOUBLE is missing.

Well, seemed sort of unnecessary...

Quote:
2) Once you call GetIniKey, the file is loaded and it is not possible to reload configuration. In a daemons/services, it is often desirable to reload config without restarting.

Fair enough, even if I cannot really imagine when I would want to change config (something you do usually a couple of times per month, at most) withou restarting. You would have to code your daemon really carefully to allow this...

Quote:
3) There are currently two separate mechanisms SetIniFile+GetIniKey function and the Ini namespace with INI_* macros. I didn't notice any connection between those two, but they are both shown in the reference/INI example.

INI_* are just top-level encapsulation for GetIniKey. In fact, GetIniKey is very old part of U++, but it proved quite resilient We have to keep it if only for maintaining BW compatibility.

Quote:
The second could be solved simply by using global static VectorMap for the key-value pairs and adding ReloadIniFile() function. Something like:
}[/code]

How about this:

```
void ReloadIniFile()
{
 s_ini_loaded = false;
}

void SetIniFile(const char *name) {
 Mutex::Lock __(sMtx);
 strcpy(sIniFile, name);
```

```
 ReloadIniFile();
}
```

Quote:
But I think there is better solution and it is related to the problem 3). I propose to drop the GetIniKey function altogether (or keep it just as wrapper for backward compatibility). If we rewrite LoadIniStream to guess value types (not very difficult)

IME, guessing value types is difficult AND wrong...

Mirek

Subject: Re: .ini file helpers
Posted by mirek on Tue, 10 Jul 2012 20:36:46 GMT
View Forum Message <> Reply to Message

INI_DOUBLE added.

BTW, I feel like we would rather need INI_INT64 and make it understand 'K', 'M', 'G' ('T') (like buffer=16M). Adding to RM for now...

Subject: Re: .ini file helpers
Posted by dolik.rce on Tue, 10 Jul 2012 21:03:08 GMT
View Forum Message <> Reply to Message

mirek wrote on Tue, 10 July 2012 22:19dolik.rce wrote on Tue, 10 July 2012 15:141)
INI_DOUBLE is missing.

Well, seemed sort of unnecessary...Well, at my work many different things is configured using configuration files, including doubles, arrays and other weird things

mirek wrote on Tue, 10 July 2012 22:19dolik.rce wrote on Tue, 10 July 2012 15:14
2) Once you call GetIniKey, the file is loaded and it is not possible to reload configuration. In a daemons/services, it is often desirable to reload config without restarting.

Fair enough, even if I cannot really imagine when I would want to change config (something you do usually a couple of times per month, at most) without restarting. You would have to code your daemon really carefully to allow this... It is quite common, just look at how many initscripts in /etc/init.d of any linux distro have a reload option

mirek wrote on Tue, 10 July 2012 22:19INI_* are just top-level encapsulation for GetIniKey. Could

you please explain that? I don't see any relation between these two things. (Maybe I'm just blind)

mirek wrote on Tue, 10 July 2012 22:19How about this:

```
void ReloadIniFile()
{
 s_ini_loaded = false;
}

void SetIniFile(const char *name) {
 Mutex::Lock __(sMtx);
 strcpy(sIniFile, name);
 ReloadIniFile();
}Ok, that would work too
```

mirek wrote on Tue, 10 July 2012 22:19IME, guessing value types is difficult AND wrong...Ok, I take that back. We don't have to guess, we know what the type is At least where programmer tells us with INI_* macro. And if he tells us, it would be IMHO much more logical to serve it to him already converted to correct type.[/quote]

mirek wrote on Tue, 10 July 2012 22:19INI_DOUBLE added.

BTW, I feel like we would rather need INI_INT64 and make it understand 'K', 'M', 'G' ('T') (like buffer=16M). Adding to RM for now...I think the functions for suffixes are good idea. Also, allowing hexadecimal numbers would be nice

Honza

Subject: Re: .ini file helpers
Posted by mirek on Wed, 11 Jul 2012 08:12:15 GMT
View Forum Message <> Reply to Message

mirek wrote on Tue, 10 July 2012 22:19INI_* are just top-level encapsulation for GetIniKey. Could you please explain that? I don't see any relation between these two things. (Maybe I'm just blind)
[/quote]

Well, there is something like key=value type of configuration file which we want to support. We want to support reading such files and obtaining values for specific keys. Hence GetIniKey.

On higher level, we would like framework to perhaps take care about some of these values - parse the Value for appropriate type, store into global variable. That is INI_* stuff.

Quote:
mirek wrote on Tue, 10 July 2012 22:19How about this:

```
void ReloadIniFile()
{
 s_ini_loaded = false;
}

void SetIniFile(const char *name) {
 Mutex::Lock __(sMtx);
 strcpy(sIniFile, name);
 ReloadIniFile();
}Ok, that would work too
```

Unfortunately, it is not that simple - we are using ONCELOCK_(loaded) in Ini variables, will need some more adjusting (as would need your version).

Mirek

Subject: Re: .ini file helpers
Posted by dolik.rce on Wed, 11 Jul 2012 09:10:36 GMT
View Forum Message <> Reply to Message

Ok, now I see where I did a mistake. I called GetIniFormatted before SetIniFile to see the defualts. But GetIniFormatted calls GetIniKey and it fixes the file to q.ini, so the subsequent call to SetIniFile is useless. I guess all my confusion came from that.

I'll have a closer look at the reloading problem...

Honza

Subject: Re: .ini file helpers
Posted by mirek on Wed, 11 Jul 2012 09:43:41 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Wed, 11 July 2012 05:10
I'll have a closer look at the reloading problem...

Honza

I will fix that soon. But to be fast enough, I am afraid thread__ variables will be needed

Mirek

Subject: Re: .ini file helpers

Posted by mirek on Fri, 13 Jul 2012 10:30:46 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Tue, 10 July 2012 17:03
BTW, I feel like we would rather need INI_INT64 and make it understand 'K', 'M', 'G' ('T') (like buffer=16M). Adding to RM for now...I think the functions for suffixes are good idea. Also, allowing hexadecimal numbers would be nice

Honza[/quote]

KMGT + 0x and INI_INT64 implemented...

Mirek

Subject: Re: .ini file helpers
Posted by mirek on Tue, 24 Jul 2012 07:24:31 GMT
View Forum Message <> Reply to Message

Code changed to use DCL pattern to avoid mutex on (repeated) read path (hopefully I got all the lock-free voodoo right

---