Subject: Re: .ini file helpers
Posted by mirek on Tue, 10 Jul 2012 20:19:14 GMT
View Forum Message <> Reply to Message

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
