```
1  /* -*- Mode: C++; tab-width: 2; indent-tabs-mode: nil; c-basic-offset: 2 -*-
2   *
3   * The contents of this file are subject to the Netscape Public License
4   * Version 1.0 (the "NPL"); you may not use this file except in
5   * compliance with the NPL.  You may obtain a copy of the NPL at
6   * http://www.mozilla.org/NPL/
7   *
8   * Software distributed under the NPL is distributed on an "AS IS" basis,
9   * WITHOUT WARRANTY OF ANY KIND, either express or implied. See the NPL
10  * for the specific language governing rights and limitations under the
11  * NPL.
12  *
13  * The Initial Developer of this code under the NPL is Netscape
14  * Communications Corporation.  Portions created by Netscape are
15  * Copyright (C) 1998 Netscape Communications Corporation.  All Rights
16  * Reserved.
17  */
18
19 /*
20  * nsCacheManager- The boss of cache architecture. Contains all "external"
21  * functions for use by people who don't care/want to know the internals
22  * of the cache architecture.
23  *
24  * - Gagan Saksena 09/15/98
25  *
26  * Design and original implementation by Gagan Saksena 02/02/98
27  *
28  */
29
30 #ifndef _CacheManager_H_
31 #define _CacheManager_H_
32
33 #if 0
34 #   include "nsISupports.h"
35 #endif
36
37 #include "prlog.h"
38
39 #include "nsMonitorable.h"
40 #include "nsCacheModule.h"
41 #include "nsCacheObject.h"
42
43 class nsMemModule;
44 class nsDiskModule;
45 class nsCachePref;
46 class nsCacheBkgThd;
47
48 class nsCacheManager : public nsMonitorable //: public nsISupports
49 {
50
51 public:
52
```

```
55      {
56         MEM =0,
57         DISK=1
58      };
59
60      nsCacheManager();
61      ~nsCacheManager();
62
63      PRInt32                 AddModule(nsCacheModule* i_cacheModule);
64                              // InsertModule
65      PRBool                  Contains(const char* i_url) const;
66
67      /* Number of modules in the cache manager */
68      PRInt16                 Entries() const;
69
70      /* Singleton */
71      static nsCacheManager*  GetInstance();
72
73      nsDiskModule*           GetDiskModule() const;
74
75      nsMemModule*            GetMemModule() const;
76
77      nsCacheModule*          GetModule(PRInt16 i_index) const;
78
79      nsCacheObject*          GetObj(const char* i_url) const;
80
81      nsCachePref*            GetPrefs(void) const;
82
83      // Initialize the cache manager. Constructor doesn't call this.
84      // So you must specify this separately.
85      void                    Init();
86
87      void                    InfoAsHTML(char* o_Buffer) const;
88
89      PRBool                  IsOffline(void) const;
90
91      void                    Offline(PRBool bSet);
92
93      PRBool                  Remove(const char* i_url);
94
95      const char*             Trace() const;
96
97      /* Performance measure- microseconds */
98      PRUint32                WorstCaseTime(void) const;
99
100   protected:
101
102     PRBool                  ContainsExactly(const char* i_url) const;
103
104     nsCacheModule*          LastModule() const;
105     //PRBool                  Lock(void);
106     //void                    Unlock(void);
```

```
109         class MgrMonitor
110         {
111         public:
112             MgrMonitor() { nsCacheManager::GetInstance()->Lock();}
113             ~MgrMonitor() { nsCacheManager::GetInstance()->Unlock();}
114         };
115
116         friend MgrMonitor;
117     */
118
119     private:
120         nsCacheBkgThd*      m_pBkgThd;
121         nsCacheModule*      m_pFirstModule;
122         PRMonitor*          m_pMonitor;
123         PRBool              m_bOffline;
124         nsCachePref*        m_pPrefs;
125
126         nsCacheManager(const nsCacheManager& cm);
127         nsCacheManager& operator=(const nsCacheManager& cm);
128     };
129
130     inline
131     nsDiskModule* nsCacheManager::GetDiskModule() const
132     {
133         PR_ASSERT(m_pFirstModule && m_pFirstModule->NextModule());
134         return (m_pFirstModule) ? (nsDiskModule*) m_pFirstModule->NextModule() : NULL;
135     }
136
137     inline
138     nsMemModule* nsCacheManager::GetMemModule() const
139     {
140         PR_ASSERT(m_pFirstModule);
141         return (nsMemModule*) m_pFirstModule;
142     }
143
144     inline
145     nsCachePref* nsCacheManager::GetPrefs(void) const
146     {
147         PR_ASSERT(m_pPrefs);
148         return m_pPrefs;
149     }
150
151     inline
152     PRBool nsCacheManager::IsOffline(void) const
153     {
154         return m_bOffline;
155     }
156
157     inline
158     void  nsCacheManager::Offline(PRBool i_bSet)
159     {
160         m_bOffline = i_bSet;
```

```
163   #endif
164
```