LMX_GetFeatureInfo

The information on this page refers to LM-X 5.4 or newer, which introduced a new LMX_SHARE_CLOUD setting for the int nShareCode field.

The LMX_GetFeatureInfo function retrieves information for a checked out feature.

Prototype

```
LMX_STATUS LMX_GetFeatureInfo
(
   LMX_HANDLE LmxHandle,
   const char *szFeatureName,
   LMX_FEATURE_INFO *pFI
);
```

Parameters

LmxHandle
[in] LM-X handle.

szFeatureName
[in] Feature name.

pFI
[out] Pointer to the feature information structure. For more details, see the table of LMX_FEATURE_INFO fields in Remarks, below.

Return values

On success, this function returns the status code LMX_SUCCESS.

On failure, this function returns an error code in the format described in Return codes.

Remarks

Use this API call to retrieve feature settings, as described in the following table. Note that trial licenses have substantially less information available than other license types, because they are a pre-configured license type and do not have the same flexibility.

You can call LMX_GetFeatureInfo from the heartbeat callback functions (see Heartbeats for further information).

LMX_FEATURE_INFO contains the following fields.

LMX_FEATURE_INFO field	Description
char szFeatureName [LMX_MAX_NAME_LENGTH] char szVendorName [LMX_MAX_NAME_LENGTH]	These strings contain the feature name and vendor name. The feature name is case-insensitive.
char szStartDate [LMX_MAX_SHORT_STRING_LENGTH] char szEndDate [LMX_MAX_SHORT_STRING_LENGTH]	These strings contain the start and expire date in format "YYYY-MM-DD". If no limitations are set, the strings will be empty.
char szActualExpireTime [LMX_MAX_SHORT_STRING_LENGTH]	This string contains the actual license expire time in format "YYYY-MM-DD HH:MM". If a borrow, grace or trial license is in use, that is the expiration time returned. Otherwise, the expiration time of the local or network license is returned.
char szMaintenanceStartDate [LMX_MAX_SHORT_STRING_LENGTH] char szMaintenanceEndDate [LMX_MAX_SHORT_STRING_LENGTH]	These strings contain the start and expire date for optional maintenance in the format "YYYY-MM-DD". If no limitations are set, the strings will be empty.

char szIssuedDate [LMX_MAX_SHORT_STRING_LENGTH]	This string contains the license issue date in the format "YYYY-MM-DD" if specified in the license.
char szPlatforms [LMX_MAX_SHORT_STRING_LENGTH]	This field contains the platforms the license can be used with.
char szComment [LMX_MAX_FIELD_LENGTH]	This field contains the COMMENT string content of the license.
char szData [LMX_MAX_FIELD_LENGTH]	This field contains the DATA string content of the license.
char szLicensee [LMX_MAX_FIELD_LENGTH]	This field contains the LICENSEE string content of the license.
char szKeyComment [LMX_MAX_FIELD_LENGTH]	This field contains the string content embedded within the KEY field.
char szOptions [LMX_MAX_FIELD_LENGTH]	This field contains the string content embedded within the OPTIONS field.
char szSN [LMX_MAX_FIELD_LENGTH]	This field contains the string content embedded within the SN field.
char szKey [LMX_MAX_FIELD_LENGTH]	This field contains the KEY signature of the license.
int nAvailableLicCount	This integer contains the number of licenses available on the license server: -1 (Unlimited network licenses [same as LMX_UNLIMITED_COUNT]) 0 (Local licenses) 1 - 2147483647 (Normal network licenses [same as LMX_MIN_COUNT to LMX_MAX_COUNT])
int nUsedLicCount	This integer contains the number of licenses taken from the license server for the particular feature. The number is within the range LMX_MIN_COUNT to LMX_MAX_COUNT.
int nMajorVer int nMinorVer	This integer contains major and minor version numbers. Note that for trial licenses these numbers will both be zero regardless of the version specified in your checkout.
int nSoftLimit	This integer contains the soft limit for the specific feature. This is only available for network licenses and must be less than the count of available licenses.
int nShareCode	This integer contains the type of sharing in use for the specific feature. This flag can be set to LMX_SHARE_HOST, LMX_SHARE_USER, LMX_SHARE_CUSTOM, LMX_SHARE_VIRTUAL, and/or LMX_SHARE_CLOUD for network licenses, and LMX_SHARE_TS or LMX_SHARE_SINGLE for local licenses. If there is no sharing in use, this flag is set to LMX_SHARE_NONE. To test whether a specific type of sharing is enabled, use the following (this example checks whether host sharing is enabled): if (FI.nShareCode & LMX_SHARE_HOST)
LMX_LICENSE_TYPE eLicenseType	This enum contains the license type, which can be one of the following: • LMX_TYPE_LOCAL • LMX_TYPE_NETWORK • LMX_TYPE_BORROW • LMX_TYPE_GRACE • LMX_TYPE_TRIAL Note that trial licenses have substantially less information available than other license types, and these fields will be zero or empty.
LMX_KEYTYPE eKeyType	This integer contains the keytype, which can be one of the following: • LMX_KEYTYPE_EXCLUSIVE • LMX_KEYTYPE_ADDITIVE • LMX_KEYTYPE_TOKEN

LMX_TOKEN_DEPENDENCY *pTokenDependency	This pointer holds information about any token dependencies the feature has. (Applies only to network licenses.) This is always set to NULL when using LMX_GetFeatureInfo. You can use LMX_GetLicenseInfo to obtain token dependencies for token-based features. See LMX_GetLicenseInfo for usage of linked lists.
char szPath [LMX_MAX_SHORT_STRING_LENGTH]	This field contains the path for the license, which will be one of the following: • The hostname for a network license • The path to the license file for a local license • "Embedded" for license checkout from a string • "Securestore" for a borrow, trial or grace license
int nServerPort	This integer contains the port number used by the license server. It is set only when eLicenseType is LMX_TYPE_NETWORK.
int nClientLicenseHostids int nServerLicenseHostids LMX_HOSTID ClientLicenseHostid [LMX_MAX_HOSTIDS] LMX_HOSTID ServerLicenseHostid [LMX_MAX_HOSTIDS]	These fields contain information about how many and which HostIDs the feature is bound to. • If nClientLicenseHostid is 0, the application is not locked to client host. • If nServerLicenseHostid is 0, the application is not locked to license server.
int nHoldMinutes int nBorrowHours int nGraceHours	These integers contain information about the number of hold minutes, borrow hours, and grace hours set for the feature in the license.
int nActualBorrowHours	This integer contains information about the actual number of borrow hours, which may be restricted by an administrator on the license server. If no restrictions are set, then this will be the value specified by the license.
char szUniqueID [LMX_MAX_SHORT_STRING_LENGTH]	This string can be used to identify a unique issued license. You can use this information to track specific licenses or for blacklisting. A unique license is identified by the feature name and this string, for example: {F2, 12345678901234567890}
int nSystemClockCheck	This integer contains information about whether a system clock check has been performed. For standalone licenses, this applies to the license client. For network licenses, this applies to the license server, but not the license client. Possible values are: 1 (Enabled) 0 (Disabled)
int nHostidLicenseMatchRate	This integer contains the HostID matching rate as specified by the license. Possible values are 0 - 100.
int nHostidActualMatchRate	This integer contains the actual HostID matching performed at checkout. Using this information, you can see the results of the actual HostID verification. Possible values are 0 - 100.
int nUserBasedCount	This integer contains the user-based reservation count required for the license. Possible values are: -1 The user-based reservation count is applied to all licenses. Use integer nAvailableLicCount. 0 Disabled >0 Specific number of licenses that are required to be reserved.

int nHostBasedCount	This integer contains the host-based reservation count required for the license. Possible values are: -1 (The host-based reservation count is applied to all licenses. Use integer nAvailableLicCount.) 0 (Disabled.) >0 (Specific number of licenses that are required to be reserved.)
int sTimeZones [LMX_MAX_SHORT_STRING_LENGTH] int nTimeZonesCount	This array of integers contains the allowed time zones relative to GMT. For example, an entry of "-5" specifies the timezone GMT -5. If the array is empty, all time zones are allowed.
int nTrialUses	This integer contains the number of possible uses for trial licenses. Default value is -1, which means that trial uses is not set.
int nBlacklisted	This integer contains information about whether a license has been blacklisted. Possible values are: 1 - Enabled 0 - Disabled

Example

You can use the following code to retrieve information for a checked out feature. On success, the function prints out the feature name and vendor name.

```
#include <lmx.h>
#include <stdio.h>

LMX_HANDLE h;

int main() {
  LMX_FEATURE_INFO FI;

  exit_on_error(LMX_Init(&h));
  exit_on_error(LMX_Checkout(h, "f2", 1, 0, 1));
  exit_on_error(LMX_GetFeatureInfo(h, "f2", &FI));
  printf("FeatureName : %s\n", FI.szFeatureName);
  printf("VendorName : %s\n", FI.szVendorName);
  return 0;
}
```

You can use the following code to check the time remaining to the expiration of a feature. Depending on which type of license is in use, the function either prints out information that the feature does not expire, or it returns the number of hours before the feature expires.

```
#include <lmx.h>
#include <stdio.h>

LMX_HANDLE h;

int main()
{
  LMX_FEATURE_INFO FI;

  exit_on_error(LMX_Init(&h));
  exit_on_error(LMX_Checkout(h, "f2", 1, 0, 1));
  exit_on_error(LMX_GetFeatureInfo(h, "f2", &FI));
  int nTimeLeft;
  nTimeLeft = LMX_GetExpireTime(h, "f2");
  if (nTimeLeft == -2)
    printf("This feature does not expire\n");
  else if (nTimeLeft == -1)
    printf("This feature is expired\n");
  else /* feature has not yet expired */
    printf("Hours left for this feature: %d\n", nTimeLeft);
  return 0;
}
```