⓪ MediaInfo源代码分析 2：API函数

==

MediaInfo源代码分析系列文章列表：

==

本文主要分析MediaInfo的API函数。它的API函数位于MediaInfo.h文件中的一个叫做MediaInfo的类中。

该类如下所示，部分重要的方法已经加上了注释：

```cpp
//MediaInfo类
class MEDIAINFO_EXP MediaInfo
{
public :
    //Constructor/Destructor
    MediaInfo ();
    ~MediaInfo ();
    //File
        /// Open a file and collect information about it (technical information and tags)
        /// @brief Open a file
        /// @param File_Name Full name of file to open
        /// @retval 0 File not opened
        /// @retval 1 File opened
    //打开文件
    size_t Open (const String &File_Name);
        /// Open a Buffer (Begin and end of the stream) and collect information about it (technical information and tags)
        /// @brief Open a buffer
        /// @param Begin First bytes of the buffer
        /// @param Begin_Size Size of Begin
        /// @param End Last bytes of the buffer
        /// @param End_Size Size of End
        /// @param File_Size Total size of the file
        /// @retval 0 File not opened
        /// @retval 1 File opened
    //打开一段内存！
    size_t Open (const ZenLib::int8u* Begin, size_t Begin_Size, const ZenLib::int8u* End=NULL, size_t End_Size=0, ZenLib::int64u File_Size=0);
        /// Open a stream and collect information about it (technical information and tags)
        /// @brief Open a stream (Init)
        /// @param File_Size Estimated file size
        /// @param File_Offset Offset of the file (if we don't have the beginning of the file)
        /// @retval 0 File not opened
        /// @retval 1 File opened
    //打开一个流和收集的关于它的信息
    size_t Open_Buffer_Init (ZenLib::int64u File_Size=(ZenLib::int64u)-1, ZenLib::int64u File_Offset=0);
        /// Open a stream and collect information about it (technical information and tags)
        /// @brief Open a stream (Continue)
        /// @param Buffer pointer to the stream
        /// @param Buffer_Size Count of bytes to read
        /// @return a bitfield \n
        ///         bit 0: Is Accepted  (format is known)
        ///         bit 1: Is Filled    (main data is collected)
        ///         bit 2: Is Updated   (some data have beed updated, example: duration for a real time MPEG-TS stream)
        ///         bit 3: Is Finalized (No more data is needed, will not use further data)
        ///         bit 4-15: Reserved
        ///         bit 16-31: User defined
    //打开一个流和收集的关于它的信息
    size_t Open_Buffer_Continue (const ZenLib::int8u* Buffer, size_t Buffer_Size);
        /// Open a stream and collect information about it (technical information and tags)
        /// @brief Open a stream (Get the needed file Offset)
        /// @return the needed offset of the file \n
        ///         File size if no more bytes are needed
    ZenLib::int64u Open_Buffer_Continue_GoTo_Get ();
        /// Open a stream and collect information about it (technical information and tags)
        /// @brief Open a stream (Finalize)
        /// @retval 0 failed
        /// @retval 1 succeed
    //打开一个流和收集的关于它的信息
    size_t Open_Buffer_Finalize ();
        /// If Open() is used in "PerPacket" mode, parse only one packet and return
```

```cpp
60.          /// @brief Read one packet (if "PerPacket" mode is set)
61.          /// @return a bitfield \n
62.          ///          bit 0: A packet was read
63.     size_t Open_NextPacket ();
64.          /// (NOT IMPLEMENTED YET) Save the file opened before with Open() (modifications of tags)
65.          /// @brief (NOT IMPLEMENTED YET) Save the file
66.          /// @retval 0 failed
67.          /// @retval 1 suceed
68.     size_t Save ();
69.          /// Close a file opened before with Open() (without saving)
70.          /// @brief Close a file
71.          /// @warning without have saved before, modifications are lost
72.     void Close ();
73.
74.     //General information
75.          /// Get all details about a file in one string
76.          /// @brief Get all details about a file
77.          /// @param Reserved Reserved, do not use
78.          /// @pre You can change default presentation with Inform_Set()
79.          /// @return Text with information about the file
80.     //返回文件信息
81.     String Inform (size_t Reserved=0);
82.
83.     //Get
84.          /// Get a piece of information about a file (parameter is an integer)
85.          /// @brief Get a piece of information about a file (parameter is an integer)
86.          /// @param StreamKind Kind of stream (general, video, audio...)
87.          /// @param StreamNumber Stream number in Kind of stream (first, second...)
88.          /// @param Parameter Parameter you are looking for in the stream (Codec, width, bitrate...), in integer format (first paramet
     second parameter...)
89.          /// @param InfoKind Kind of information you want about the parameter (the text, the measure, the help...)
90.          /// @return a string about information you search \n
91.          ///          an empty string if there is a problem
92.     //获取一部分文件的信息 (参数是一个整数)
93.     String Get (stream_t StreamKind, size_t StreamNumber, size_t Parameter, info_t InfoKind=Info_Text);
94.          /// Get a piece of information about a file (parameter is a string)
95.          /// @brief Get a piece of information about a file (parameter is a string)
96.          /// @param StreamKind Kind of stream (general, video, audio...)
97.          /// @param StreamNumber Stream number in Kind of stream (first, second...)
98.          /// @param Parameter Parameter you are looking for in the stream (Codec, width, bitrate...), in string format ("Codec", "Widt
     ..) \n
99.          ///          See MediaInfo::Option("Info_Parameters") to have the full list
100.         /// @param InfoKind Kind of information you want about the parameter (the text, the measure, the help...)
101.         /// @param SearchKind Where to look for the parameter
102.         /// @return a string about information you search \n
103.         ///          an empty string if there is a problem
104.    //获取一部分文件的信息 (参数是一个字符串)
105.    String Get (stream_t StreamKind, size_t StreamNumber, const String &Parameter, info_t InfoKind=Info_Text, info_t SearchKind=Info_
     ame);
106.
107.    //Set
108.         /// (NOT IMPLEMENTED YET) Set a piece of information about a file (parameter is an integer)
109.         /// @brief (NOT IMPLEMENTED YET) Set a piece of information about a file (parameter is an int)
110.         /// @warning Not yet implemented, do not use it
111.         /// @param ToSet Piece of information
112.         /// @param StreamKind Kind of stream (general, video, audio...)
113.         /// @param StreamNumber Stream number in Kind of stream (first, second...)
114.         /// @param Parameter Parameter you are looking for in the stream (Codec, width, bitrate...), in integer format (first paramet
     second parameter...)
115.         /// @param OldValue The old value of the parameter \n if OldValue is empty and ToSet is filled: tag is added \n if OldValue i
     illed and ToSet is filled: tag is replaced \n if OldValue is filled and ToSet is empty: tag is deleted
116.         /// @retval >0 succeed
117.         /// @retval 0 failed
118.    size_t Set (const String &ToSet, stream_t StreamKind, size_t StreamNumber, size_t Parameter, const String &OldValue=String());
119.         /// (NOT IMPLEMENTED YET) Set a piece of information about a file (parameter is a string)
120.         /// @warning Not yet implemented, do not use it
121.         /// @brief (NOT IMPLEMENTED YET) Set information about a file (parameter is a string)
122.         /// @param ToSet Piece of information
123.         /// @param StreamKind Kind of stream (general, video, audio...)
124.         /// @param StreamNumber Stream number in Kind of stream (first, second...)
125.         /// @param Parameter Parameter you are looking for in the stream (Codec, width, bitrate...), in string format
126.         /// @param OldValue The old value of the parameter \n if OldValue is empty and ToSet is filled: tag is added \n if OldValue i
     illed and ToSet is filled: tag is replaced \n if OldValue is filled and ToSet is empty: tag is deleted
127.         /// @retval >0 succeed
128.         /// @retval 0 failed
129.    size_t Set (const String &ToSet, stream_t StreamKind, size_t StreamNumber, const String &Parameter, const String &OldValue=String
     ());
130.
131.    //Output_Buffered
132.         /// Output the written size when "File_Duplicate" option is used.
133.         /// @brief Output the written size when "File_Duplicate" option is used.
134.         /// @param Value The unique name of the duplicated stream (begin with "memory://")
135.         /// @return The size of the used buffer
136.    size_t Output_Buffer_Get (const String &Value);
137.         /// Output the written size when "File_Duplicate" option is used.
138.         /// @brief Output the written size when "File_Duplicate" option is used.
139.         /// @param Pos The order of calling
140.         /// @return The size of the used buffer
141.    size_t Output_Buffer_Get (size_t Pos);
142.
143.    //Info
```

```cpp
144.         /// Configure or get information about MediaInfoLib
145.         /// @param Option The name of option
146.         /// @param Value The value of option
147.         /// @return Depend of the option: by default "" (nothing) means No, other means Yes
148.         /// @post Known options are: \n
149.         ///       * (NOT IMPLEMENTED YET) "BlockMethod": Configure when Open Method must return (default or not command not understoo
     "1") \n
150.         ///             "0": Immediatly \n
151.         ///             "1": After geting local information \n
152.         ///             "2": When user interaction is needed, or whan Internet information is get
153.         ///       * "Complete": For debug, configure if MediaInfoLib::Inform() show all information (doesn't care of InfoOption_NoSho
     ag): shows all information if true, shows only useful for user information if false (No by default)\n
154.         ///       * "Complete_Get": return the state of "Complete" \n
155.         ///       * "Language": Configure language (default language, and this object); Value is Description of language (format: "Co
     n1;Colum2\n...) \n
156.         ///             Column 1: Unique name ("Bytes", "Title") \n
157.         ///             Column 2: translation ("Octets", "Titre") \n
158.         ///       * "Language_Get": Get the language file in memory
159.         ///       * "Language_Update": Configure language of this object only (for optimisation); Value is Description of language (f
     at: "Column1;Colum2\n...) \n
160.         ///             Column 1: Unique name ("Bytes", "Title") \n
161.         ///             Column 2: translation ("Octets", "Titre") \n
162.         ///       * "Inform": Configure custom text, See MediaInfoLib::Inform() function; Description of views (format: "Column1;Colu
     ..) \n
163.         ///             Column 1: code (11 lines: "General", "Video", "Audio", "Text", "Other", "Begin", "End", "Page_Begin", "Pa
     Middle", "Page_End") \n
164.         ///             Column 2: The text to show (exemple: "Audio: %FileName% is at %BitRate/String%") \n
165.         ///       * "ParseUnknownExtensions": Configure if MediaInfo parse files with unknown extension\n
166.         ///       * "ParseUnknownExtensions_Get": Get if MediaInfo parse files with unknown extension\n
167.         ///       * "ShowFiles": Configure if MediaInfo keep in memory files with specific kind of streams (or no streams); Value is
     cription of components (format: "Column1;Column2\n...) \n\n
168.         ///             Column 1: code (available: "Nothing" for unknown format, "VideoAudio" for at least 1 video and 1 audio, "
     eoOnly" for video streams only, "AudioOnly", "TextOnly") \n
169.         ///             Column 2: "" (nothing) not keeping, other for keeping
170.         ///       * (NOT IMPLEMENTED YET) "TagSeparator": Configure the separator if there are multiple same tags (" | " by default)\
171.         ///       * (NOT IMPLEMENTED YET) "TagSeparator_Get": return the state of "TagSeparator" \n
172.         ///       * (NOT IMPLEMENTED YET) "Internet": Authorize Internet connection (Yes by default)
173.         ///       * (NOT IMPLEMENTED YET) "Internet_Title_Get": When State=5000, give all possible titles for this file (one per line
     n
174.         ///             Form: Author TagSeparator Title TagSeparator Year\n...
175.         ///       * (NOT IMPLEMENTED YET) "Internet_Title_Set": Set the Good title (same as given by Internet_Title_Get) \n
176.         ///             Form: Author TagSeparator Title TagSeparator Year
177.         ///       * "Info_Parameters": Information about what are known unique names for parameters \n
178.         ///       * "Info_Parameters_CSV": Information about what are known unique names for parameters, in CSV format \n
179.         ///       * "Info_Codecs": Information about which codec is known \n
180.         ///       * "Info_Version": Information about the version of MediaInfoLib \n
181.         ///       * "Info_Url": Information about where to find the last version
182.      //配置
183.      String      Option (const String &Option, const String &Value=String());
184.         /// Configure or get information about MediaInfoLib
185.         /// @param Option The name of option
186.         /// @param Value The value of option
187.         /// @return Depend of the option: by default "" (nothing) means No, other means Yes
188.         /// @post Known options are: See MediaInfo::Option()
189.      static String Option_Static (const String &Option, const String &Value=String());
190.         /// @brief (NOT IMPLEMENTED YET) Get the state of the library
191.         /// @retval <1000 No information is available for the file yet
192.         /// @retval >=1000_<5000 Only local (into the file) information is available, getting Internet information (titles only) is n
     inished yet
193.         /// @retval 5000 (only if Internet connection is accepted) User interaction is needed (use Option() with "Internet_Title_Get"
     n
194.         ///         Warning: even there is only one possible, user interaction (or the software) is needed
195.         /// @retval >5000<=10000 Only local (into the file) information is available, getting Internet information (all) is no finish
     yet
196.         /// @retval <10000 Done
197.      size_t           State_Get ();
198.         /// @brief Count of streams of a stream kind (StreamNumber not filled), or count of piece of information in this stream
199.         /// @param StreamKind Kind of stream (general, video, audio...)
200.         /// @param StreamNumber Stream number in this kind of stream (first, second...)
201.         /// @return The count of fields for this stream kind / stream number if stream number is provided, else the count of streams
       this stream kind
202.      size_t           Count_Get (stream_t StreamKind, size_t StreamNumber=(size_t)-1);
203.
204. private :
205.      MediaInfo_Internal* Internal;
206.
207.      //Constructor
208.      MediaInfo (const MediaInfo&);                        // Prevent copy-construction
209.      MediaInfo& operator=(const MediaInfo&);              // Prevent assignment
210. };
```

由代码可见，MediaInfo实际上不仅仅可以读取一个特定路径的文件【Open (const String &File_Name);】

,而且可以读取一块内存中的数据【Open (const ZenLib::int8u* Begin, size_t Begin_Size, const ZenLib::int8u* End=NULL, size_t End_Size=0, ZenLib::int64u File_Size=0)】（这个函数目前还没有使用过）。

如果想要获取完整的信息，可以使用【Inform (size_t Reserved=0)】，并且使用【Option (const String &Option, const String &Value=String())】进行配置。

如果只想获得特定的信息，可以使用【Get (stream_t StreamKind, size_t StreamNumber, size_t Parameter, info_t InfoKind=Info_Text)】。

具体使用方法参见 http://blog.csdn.net/leixiaohua1020/article/details/11902195 。

文章标签： mediainfo   api   源代码

个人分类： MediaInfo

所属专栏： 开源多媒体项目源代码分析