|  |  |  |
| --- | --- | --- |
| Bochs Developers Guide | | |
| [Prev](http://docs.google.com/configure-scripting.html) | Chapter 2. About the code | [Next](http://docs.google.com/timers.html) |

2.5. Log Functions

The *logfunctions* class is one of the base classes of Bochs. It supports 4 log levels (debug, info, error, panic) and 4 possible "actions" that can be done when a log event occurs. Most of the higher level C++ classes of Bochs inherit this class to make the logging configuration per object (here called "module") possible. In the Bochs sources the log events appear as macros (BX\_DEBUG, BX\_INFO, BX\_ERROR, BX\_PANIC) and they call the related logfunction methods, unless the symbol BX\_NO\_LOGGING is set to 1. This is the definition in *bochs.h*:

typedef class BOCHSAPI logfunctions  
{  
 char \*name;  
 char \*prefix;  
 int onoff[N\_LOGLEV];  
 class iofunctions \*logio;  
 // default log actions for all devices, declared and initialized  
 // in logio.cc.  
 BOCHSAPI\_CYGONLY static int default\_onoff[N\_LOGLEV];  
public:  
 logfunctions(void);  
 logfunctions(class iofunctions \*);  
 ~logfunctions(void);  
  
 void info(const char \*fmt, ...) BX\_CPP\_AttrPrintf(2, 3);  
 void error(const char \*fmt, ...) BX\_CPP\_AttrPrintf(2, 3);  
 void panic(const char \*fmt, ...) BX\_CPP\_AttrPrintf(2, 3);  
 void ldebug(const char \*fmt, ...) BX\_CPP\_AttrPrintf(2, 3);  
 void fatal (const char \*prefix, const char \*fmt, va\_list ap, int exit\_status);  
 void ask (int level, const char \*prefix, const char \*fmt, va\_list ap);  
 void put(const char \*p);  
 void put(const char \*n, const char \*p);  
 void setio(class iofunctions \*);  
 void setonoff(int loglev, int value) {  
 assert (loglev >= 0 && loglev < N\_LOGLEV);  
 onoff[loglev] = value;  
 }  
 const char \*get\_name() const { return name; }  
 const char \*getprefix() const { return prefix; }  
 int getonoff(int level) const {  
 assert (level>=0 && level<N\_LOGLEV);  
 return onoff[level];  
 }  
 static void set\_default\_action(int loglev, int action) {  
 assert (loglev >= 0 && loglev < N\_LOGLEV);  
 assert (action >= 0 && action < N\_ACT);  
 default\_onoff[loglev] = action;  
 }  
 static int get\_default\_action(int loglev) {  
 assert (loglev >= 0 && loglev < N\_LOGLEV);  
 return default\_onoff[loglev];  
 }  
} logfunc\_t;

2.5.1. Methods

Here is a short description of some *logfunctions* methods.

* The *constructor* registers a new log module with default values. The module's log prefix is empty and the log levels are set up with default actions.
* The *destructor* removes the log module from the table.
* The *info()*, *error()*, *panic()* and *ldebug()* methods are called via macros to create a log event of the related level.
* The *fatal()* method is called if a log event occurs and it's action is set to "fatal". It is used to shut down the Bochs simulation.
* The *ask()* method is called if a log event occurs and it's action is set to "ask". It sends an event to the simulator interface and depending on the return value the simulation continues or it is terminated by calling *fatal()*. The simulator interface either prompts the user on the console or calls some platform / gui specific code to handle the *ask* request.
* The *put()* methods are used to set up the log module prefix in that appears in the log file and the log module name that appears in the config interface. If the name is not specified, the prefix is used instead.
* The *setio()* method sets up the *iofunctions* class for the log file output. This method is only used by the *logfunctions* constructors.
* The *getonoff()* and *setonoff()* methods are used by the config interface to display and change the log actions for a Bochs facility.
* The *get\_default\_action()* and *set\_default\_action()* methods are also used by the config interface to set up the default action for a log level.
* The *get\_name()* and *getprefix()* methods return the strings set up with the *put()* method. The config interface is also using them to build the menu / dialog to set up the log functions.

|  |  |  |
| --- | --- | --- |
| [Prev](http://docs.google.com/configure-scripting.html) | [Home](http://docs.google.com/index.html) | [Next](http://docs.google.com/timers.html) |
| Configure Scripting | [Up](http://docs.google.com/about-the-code.html) | Internal timers |