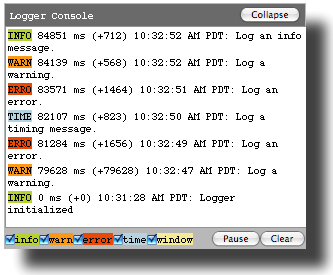
YUI Library: Logger

2006-07-05 **v0.11**

Simple Use Case (LogReader)

```
<div id="myLogger"></div>
<script>
var myLogReader = new
   YAHOO.widget.LogReader("myLogger");
</script>
```



Instantiates a new LogReader object, myLogReader, which is bound to a div whose id attribute is 'myLogger'. The result will be a visual LogReader display.

To create a LogReader that floats outside the page context, omit the reference to a context div. Your LogReader will then be appended to the page and positioned absolutely. If the YUI Drag & Drop Library is included on the page, it will be draggable.

Constructor (LogReader)

```
YAHOO.widget.LogReader([str html id or obj element
   reference, obj configuration object]);
```

Arguments:

(1) **HTML element (string or object):** An optional reference to an HTML id string or element object binds the LogReader to an existing page element.

(2) **Configuration object (object):** An optional object literal defines LogReader settings. All properties of a LogReader instance can be set via the constructor by using this object.

FireBug

FireBug is a Firefox extension that combines a debugging console, XHR spy, and DOM inspector. Logger writes to the FireBug console by default, allowing you to see Logger messages even when the LogReader UI is not present (or when it is hidden). Download FireBug via the Mozilla Extensions interface under the Firefox Tools menu.

Dependencies

Logger requires the YAHOO object, Dom, Event, and the Logger CSS file; Drag and Drop is optional.

Simple Use Case (Logger)

```
YAHOO.log("My log message", "my category", "my
   source");
```

Logs a message to the default console and to FireBug, if enabled. Categories ("info", "warn", &c.) and sources can be added on the fly.

Constructor (LogWriter)

Creates a separate, named bucket for your log messages:

```
YAHOO.widget.LogWriter(str sSource);
```

Arguments:

(1) **Source (string):** The source of log messages. The first word of the string will be used to create a LogReader filter checkbox. The entire string will be prepended to log messages so they can be easily tracked by their source.

Solutions

Log a message using a pre-styled logging category:

```
YAHOO.log("My log message", "warn");
```

Create a new logging category on the fly:

```
YAHOO.log("My log message.", "new category");
```

Log a message, **creating a new "source" on the fly**:

```
YAHOO.log("My log message", "warn", "newSource");
```

Reset the global contents of the Logger, clearing all LogReaders:

```
YAHOO.widget.Logger.reset();
```

Hide and show the logging console:

```
myLogReader.hide();
myLogReader.show();
```

Pause and resume output to the console:

```
myLogReader.pause();
myLogReader.resume();
```

Instantiate your own LogWriter to write log messages categorized by their source:

```
MyClass.prototype.myLogWriter = new
   YAHOO.widget.LogWriter("MyClass of MyModule");
var myInstance = new MyClass();
myInstance.myLogWriter.log("This log message can now
   be filtered by its source, MyClass"); //"MyClass
   of MyModule", the full name of the source, will
   be prepended to the actual log message
```

YAHOO.widget.Logger Static Methods:

log(sMsg, sCategory, sSource)
disableFirebug()
enableFirebug()
getStack()
getStartTime()
reset()

YAHOO.widget.Logger Custom Events:

categoryCreateEvent
sourceCreateEvent
newLogEvent
logResetEvent

LogReader Properties:

logReaderEnabled (b)
footerEnabled (b)
verboseOutput (b)
newestOnTop (b)

LogReader Methods:

hide()
show()
pause()
resume()
setTitle()

LogWriter Methods:

log(sMsg, sCategory, sSource**)**

Pre-styled Categories

info
warn
error
time
window

(Pass in other categories to `log()` to add to this list.)