NAME

irccd-api - irccd Javascript API

SYNOPSIS

Irccd
Irccd.Chrono
Irccd.Directory
Irccd.File
Irccd.Logger
Irccd.Plugin
Irccd.Server
Irccd.System
Irccd.Timer
Irccd.Unicode
Irccd.Util

DESCRIPTION

This documentation shows the API reference for the official irccd Javascript API.

Any function that is indicated as *optional* in square brackets means it may not exist on your platform. A quick check in Javascript will let you test its presence.

EVENTS

The following is a list of events that Javascript plugins support. All functions are completely optional and may be omitted. If you want to support a function just implement it as global Javascript function.

onCommand

Special commands are not real IRC events. They are called from channel messages with a specific syntax using a delimiter and the plugin name.

For instance, with default irccd parameters, saying on a channel *!ask foo* will call the special command of the plugin named **ask**.

Synopsis

```
function onCommand(server, origin, channel, message)
```

Arguments

server (Server)	The current server.
origin (string)	Who invoked the command.

channel (string)	The channel where the message comes from.
message (string)	The real message, without the ! part.

onConnect

This callback is called when the irccd instance successfully connect to a server.

Synopsis

```
function onConnect(server)
```

Arguments

server (Server)	The current server.

onDisconnect

This callback is called when a server has been disconnected by any way.

Synopsis

```
function onDisonnect(server)
```

Arguments

server (Server)	The current server.

onInvite

This event is called when someone is inviting you to a channel.

Synopsis

```
function onInvite(server, origin, channel)
```

Arguments

server (Server)	The current server.
origin (string)	Who invited you.
channel (string)	On which channel you are invited to.

onJoin

User join events, this function is called when someone joins a channel.

Synopsis

function onJoin(server, origin, channel)

Arguments

server (Server)	The current server.
origin (string)	The person who joined the channel.
channel (string)	The channel the user has joined.

onKick

This event is triggered when someone has been kicked from a channel.

Synopsis

function onKick(server, origin, channel, target, reason)

Arguments

server (Server)	The current server.
origin (string)	Who kicked the person.
channel (string)	The channel.
target (string)	The kicked person.
reason (string)	An optional reason.

onLoad

This function is called when irccd instance load a plugin. If this function throws an error, the script is not loaded.

Synopsis

function onLoad()

onMe

Action emote.

Synopsis

function onMe(server, origin, channel, message)

Arguments

server (Server)	The current server.
origin (string)	The person who said something.
channel (string)	The channel.

message (string) The message sent.

onMessage

This event is triggered when someone said something on a specific channel.

Synopsis

function onMessage(server, origin, channel, message)

Arguments

server (Server)	The current server.
origin (string)	The person who said something.
channel (string)	The channel.
message (string)	The message sent.

onMode

This event is triggered when the server changed a channel mode or your mode.

Synopsis

function onMode(server, origin, channel, mode, limit, user, mask)

Arguments

server (Server)	The current server.
origin (string)	The person who changed the mode.
mode (string)	The new mode.

onNames

This event is triggered when a list of names has come.

Synopsis

function onNames(server, channel, list)

Arguments

server (Server)	The current server.
channel (string)	Which channel.
list (string)	A sequence with all users.

onNick

This event is triggered when someone changed its nickname.

Synopsis

```
function onNick(server, origin, nickname)
```

Arguments

server (Server)	The current server.
origin (string)	The old nickname.
nickname (string)	The new nickname.

onNotice

This event is triggered when someone sent a notice to you.

Synopsis

```
function onNotice(server, origin, notice)
```

Arguments

server (Server)	The current server.
origin (string)	The one who sent the notice.
message (string)	The notice message.

onPart

This event is triggered when someone has left a specific channel.

Synopsis

```
function onPart(server, origin, channel, reason)
```

Arguments

server (Server)	The current server.
origin (string)	The person who left the channel.
channel (string)	The channel.
reason (string)	An optional reason.

onReload

Request to reload the plugin.

This function does nothing in the irccd internals, it just calls a function that you can use to reload some

data.

Synopsis

```
function onReload()
```

onTopic

This event is triggered when someone changed the channel's topic.

Synopsis

```
function onTopic(server, origin, channel, topic)
```

Arguments

server (Server)	The current server.
origin (string)	The person who changed the topic.
channel (string)	The channel.
topic (string)	The new topic (may be empty).

onUnload

This event is triggered when the plugin is about to be unloaded.

Synopsis

```
function onUnload()
```

onWhois

This event is triggered when irccd gets information about a user.

Synopsis

```
function onWhois(server, info)
```

Arguments

server (Server)	The current server.
info (Object)	The whois information.

The *info* is an object with the following properties:

nickname (string)	The user nickname.
user (string)	The user name.

host (string)	The hostname.
realname (string)	The real name used.
channels (array)	An optional list of channels joined.

MODULES

The following modules are part of the official Javascript API. They are all accessible as global function, variables and objects.

All modules are categorized into pseudo namespaces that are placed into the global *Irccd* object. (e.g. *Irccd.Directory*, *Irccd.File*).

Irccd

Top level irccd Javascript module.

Contains general irccd variables and functions.

The following constants properties are defined:

version (object)	See below.
version.major (int)	The major irccd version.
version.minor (int)	The minor irccd version.
version.patch (int)	The patch irccd version.

The following objects are defined:

SystemError (function)	An exception inheriting Error thrown by some filesystem functions.

Irccd.Chrono

This class let you measure the elapsed time.

Irccd.Chrono [constructor]

Construct a new Chrono object. The timer is automatically started on construction.

Synopsis

```
Irccd.ElapsedTimer()
```

Irccd.Chrono.prototype.elapsed

Get the number of elapsed milliseconds.

Synopsis

Irccd.Chrono.prototype.elapsed()

Returns

The elapsed time in milliseconds.

Irccd.Chrono.prototype.pause

Pause the timer, without resetting the current elapsed time stored.

Synopsis

Irccd.Chrono.prototype.pause()

Irccd.Chrono.prototype.resume

Continue accumulating additional time. Has no effect if the timer is already running.

Synopsis

Irccd.Chrono.prototype.restart()

Irccd.Chrono.prototype.start

Starts or restarts accumulating time.

Synopsis

Irccd.Chrono.prototype.start()

Irccd.Directory

This module can be used to iterate, find, remove or create directories.

Use this module with care.

The following constants properties are defined:

Dot (int) list "." directory.

DotDot (int)	list ".." directory.
TypeUnknown (int)	unknown type file.
TypeDir (int)	entry is a directory.
TypeFile (int)	entry is a file.
TypeLink (int)	entry is a link.

Irccd.Directory.find

Find an entry by a pattern or a regular expression.

Synopsis

```
Irccd.Directory.find(path, pattern, recursive)
```

Arguments

path (string)	The base path.
pattern (mixed)	The regular expression or file name as string.
recursive (bool)	Set to true to search recursively (Optional, default: false).

Returns

The path to the file or undefined if not found.

Irccd.Directory.mkdir

Create a directory specified by path. It will create needed subdirectories just like you have invoked mkdir -p.

Synopsis

```
Irccd.Directory.mkdir(path, mode = 0700)
```

Arguments

path (string)	The path to the directory.
mode (string)	The mode, not available on all platforms.

Throws

Any exception on error.

Irccd.Directory.remove

Remove the directory optionally recursively.

Synopsis

```
Irccd.Directory.remove(path, recursive)
```

Arguments

path (string)	The path to the directory.
recursive (bool)	Recursively or not (Optional, default: false).

Throws

Any exception on error.

Irccd.Directory [constructor]

Open a directory.

When constructed successfully, the object has the following properties:

path (string)	the path to the directory.
entries (array)	an array for each entry containing. See below

For each entry found, the array entries contains as many objects with the following properties:

name (string)	the base file name.
type (int)	the type of file (Irccd.Directory.Type*).

Synopsis

```
Irccd.Directory(path, flags)
```

Arguments

path (string)	The path to the directory.

flags (int) The OR’ed flags: *Irccd.Directory.Dot*, *Irccd.Directory.DotDot* (Optional, default: none).

Throws

Any exception on error.

Irccd.Directory.prototype.find

Synonym of find static function but the path is taken from the directory object.

Synopsis

Irccd.Directory.prototype.find(pattern, recursive)

Arguments

pattern (mixed) The regular expression or file name.
recursive (bool) Set to true to search recursively (Optional, default: false).

Throws

Any exception on error.

Returns

The path to the file or undefined if not found.

Irccd.Directory.prototype.remove

Synonym of remove static function but the path is taken from the directory object.

Synopsis

Irccd.Directory.prototype.remove(recursive)

Arguments

recursive (bool) Recursively or not (Optional, default: false).

Throws

Any exception on error.

Irccd.File

This module is available for opening and writing files on the disk.

For convenience, some functions are available as free-functions and some as object methods. The following constants properties are defined:

SeekCur (int)	Seek from the current file position.
SeekEnd (int)	Seek from end of the file.
SeekSet (int)	Seek from beginning of the file.

Irccd.File.basename

Return the file basename as specified in basename C function.

Synopsis

```
base = Irccd.File.basename(path)
```

Arguments

path (string)	The path to the file.

Returns

The base name.

Irccd.File.dirname

Return the file directory name as specified in dirname C function.

Synopsis

```
path = Irccd.File.dirname(path)
```

Arguments

path (string) The path to the file.

Returns

The directory name.

Irccd.File.exists

Check if the file exists.

Warning: using this function is usually discouraged as it may introduce a possible race condition.

Synopsis

```
ret = Irccd.File.exists(path)
```

Arguments

path (string) The path to the file.

Throws

Irccd.SystemError on failure.

Returns

True if exists.

Irccd.File.remove

Remove the file at the specified path.

Synopsis

```
Irccd.File.remove(path)
```

Arguments

path (string) The path to the file.

Throws

Irccd.SystemError on failure.

Irccd.File.stat [optional]

Get file information at the specified path.

Synopsis

```
info = Irccd.File.stat(path)
```

Arguments

path (string)	The path to the file.

Throws

Irccd.SystemError on failure.

Returns

An object with the following properties. Not all properties are available and you must check its presence before using it.

atime (int)	The last access time.
blksize (int)	The block size.
blocks (int)	The number of blocks.
ctime (int)	The creation time.
dev (int)	The device.
gid (int)	The group numeric id.
ino (int)	The inode.
mode (int)	The mode.
mtime (int)	The modification time.
nlink (int)	The number of hard links.
rdev (int)	No description available.
size (int)	The file size.
uid (int)	The user numeric id.

Irccd.File [constructor]

Open a file specified by path with the specified mode.

Synopsis

```
Irccd.File(path, mode)
```

Arguments

path (string)	The path to the file.
mode (string)	The mode string.

The *mode* is the same as if called by fopen, see the documentation of fopen(3) for more information about modes.

Throws

Irccd.SystemError on failure.

Irccd.File.prototype.basename

Synonym of Irccd.File.basename static function but with the path taken from the object itself.

Synopsis

```
path = Irccd.File.prototype.basename()
```

Returns

The base name.

Irccd.File.prototype.close

Force close of the file, automatically called when object is collected.

Synopsis

```
Irccd.File.prototype.close()
```

Irccd.File.prototype.dirname

Synonym of Irccd.File.dirname static function but with the path taken from the object itself.

Synopsis

```
path = Irccd.File.prototype.dirname()
```

Returns

The directory name.

Irccd.File.prototype.lines

Read all lines and return an array.

Synopsis

```
list = Irccd.File.prototype.lines()
```

Throws

Irccd.SystemError on failure.

Returns

An array with all lines.

Irccd.File.prototype.read

Read the specified amount of characters or the whole file.

Synopsis

```
str = Irccd.File.prototype.read(amount)
```

Arguments

amount (int) The amount of characters or -1 to read all (Optional, default: -1).

Throws

Irccd.SystemError on failure.

Returns

The string.

Irccd.File.prototype.readline

Read the next line available.

Warning: this method is slow and its usage is discouraged on large files. Consider using Irccd.File.prototype.lines function if you want to read a file line per line.

Synopsis

```
line = Irccd.File.prototype.readline()
```

Throws

Irccd.SystemError on failure.

Returns

The next line or undefined if EOF.

Irccd.File.prototype.remove

Synonym of Irccd.File.remove static function but with the path taken from the object itself.

Synopsis

```
Irccd.File.prototype.remove()
```

Throws

Irccd.SystemError on failure.

Irccd.File.prototype.seek

Sets the position in the file.

Synopsis

Irccd.File.prototype.seek(type, amount)

Arguments

type (int)	The type of setting (*Irccd.File.SeekSet*, *Irccd.File.SeekCur*, *Irccd.File.SeekSet*).
amount (int)	The new offset.

Throws

Irccd.SystemError on failure.

Irccd.File.prototype.stat [optional]

Synonym of Irccd.File.stat static function but with the path taken from the object itself.

Synopsis

info = Irccd.File.prototype.stat()

Throws

Irccd.SystemError on failure.

Returns

The information object.

Irccd.File.prototype.tell

Get the actual position in the file.

Synopsis

pos = Irccd.File.prototype.tell()

Throws

Irccd.SystemError on failure.

Returns

The position.

Irccd.File.prototype.write

Write some characters to the file.

Synopsis

Irccd.File.prototype.write(data)

Arguments

data (string) The character to write.

Throws

Irccd.SystemError on failure.

Returns

The number of bytes written.

Irccd.Logger

This module must be used to log something. It will add messages to the logging system configured in the irccd.conf file.

For instance, if user has chosen to log into syslog, this module will log at syslog too.

Any plugin can log messages, the message will be prepended by the plugin name to be easily identifiable.

Irccd.Logger.debug

Adds a debug message, this is only appended to the journal if irccd was compiled in Debug mode.

Synopsis

Irccd.Logger.debug(message)

Arguments

message (string) The message.

Irccd.Logger.info

Log something. The message is logged only if irccd is running with verbose messages enabled.

Synopsis

Irccd.Logger.info(message)

Arguments

message (string) The message.

Irccd.Logger.warning

Log a warning. The message will always be logged.

Synopsis

Irccd.Logger.warning(message)

Arguments

message (string) The message.

Irccd.Plugin

This module let you manage plugins.

The following constants properties are defined and contain each key-value pairs from the user configuration file.

config (Object)	Contains the *[plugin.<name>]* section.
paths (Object)	Contains the *[paths.<name>]* section.
templates (Object)	Contains the *[templates.<name>]* section.

Irccd.Plugin.info

Get information about a plugin.

Synopsis

```
info = Irccd.Plugin.info(name)
```

Arguments

name (string)	The plugin identifier, if not specified the current plugin is selected.

Returns

The plugin information or undefined if the plugin was not found. The object has the following properties:

name (string)	The plugin identifier.
author (string)	The author.
license (string)	The license.
summary (string)	A short description.
version (string)	The version.

Irccd.Plugin.list

Get the list of plugins, the array returned contains all plugin names as strings.

Synopsis

```
list = Irccd.Plugin.list()
```

Returns

The list of all plugin names.

Irccd.Plugin.load

Load a plugin by name. This function will search through the standard directories.

Synopsis

Irccd.Plugin.load(name)

Arguments

name (string) The plugin identifier.

Throws

Error On errors.
ReferenceError If the plugin was not found.

Irccd.Plugin.reload

Reload a plugin by name.

Synopsis

Irccd.Plugin.reload(name)

Arguments

name (string) The plugin identifier.

Throws

Error On errors.
ReferenceError If the plugin was not found.

Irccd.Plugin.unload

Unload a plugin by name and remove it.

Synopsis

Irccd.Plugin.unload(name)

Arguments

name (string) The plugin identifier.

Throws

Error	On errors.
ReferenceError	If the plugin was not found.

Irccd.Server

This module is the object that you received in almost all IRC event (e.g. onConnect). You can use its methods to do your required actions on the server.

Irccd.Server.add

Add a new server to the irccd instance.

Synopsis

```
Irccd.Server.add(server)
```

Arguments

server (Server)	The server object to add.

Irccd.Server.find

Find a server by name.

Synopsis

```
server = Irccd.Server.find(name)
```

Arguments

name (string)	The server name.

Returns

The server object or undefined if not found.

Irccd.Server.list

List all servers in a map.

Synopsis

```
table = Irccd.Server.list()
```

Returns

The table of all servers as key-value pairs where key is the server identifier and value the object itself.

Irccd.Server.remove

Remove a server from the irccd instance and disconnect it.

Synopsis

```
Irccd.Server.remove(name)
```

Arguments

name (string)	The server name.

Irccd.Server [constructor]

Construct a new server.

Synopsis

```
Irccd.Server(info)
```

Arguments

info (object)	Object information.

The *info* argument may have the following properties:

name (string)	The unique identifier name.
hostname (string)	The host or IP address.
ipv4 (bool)	Enable ipv4 (Optional, default: true).
ipv6 (bool)	Enable ipv6, (Optional, default: true).
port (int)	The port number, (Optional, default: 6667).
password (string)	The password, (Optional, default: undefined).

channels (array)	Array of channels (Optional, default: empty).
ssl (bool)	True to use ssl, (Optional, default: false).
nickname (string)	Nickname, (Optional, default: irccd).
username (string)	User name, (Optional, default: irccd).
realname (string)	Real name, (Optional, default: IRC Client Daemon).
commandChar (string)	Plugin prefix character, (Optional, default: "!").

Warning: at least ipv4 and ipv6 must be set (which is the default).

Irccd.Server.prototype.info

Get server information.

Synopsis

```
info = Irccd.Server.prototype.info()
```

Returns

The server information. The object have the following properties:

name (string)	The server unique name.
hostname (string)	The host name.
port (int)	The port number.
ssl (bool)	True if using ssl.
channels (array)	An array of all channels.
realname (string)	The current real name.
username (string)	The user name.
nickname (string)	The current nickname.

Irccd.Server.prototype.invite

Invite the specified target on the channel.

Synopsis

```
Irccd.Server.prototype.invite(target, channel)
```

Arguments

target (string)	The target to invite.

channel (string) The channel.

Irccd.Server.prototype.isSelf

Check if the nickname targets the bot.

Synopsis

res = Server.prototype.isSelf(nickname)

Arguments

nickname (string) The nickname to check.

Returns

True if nickname is same as the bot.

Irccd.Server.prototype.join

Join the specified channel, the password is optional.

Synopsis

Irccd.Server.prototype.join(channel, password)

Arguments

channel (string) The channel to join.
password (string) An optional password.

Irccd.Server.prototype.kick

Kick the specified target from the channel, the reason is optional.

Synopsis

Server.prototype.kick(nickname, channel, reason)

Arguments

nickname (string)	The person to kick.
channel (string)	From which channel.
reason (string)	A reason (Optional, default: undefined).

Irccd.Server.prototype.me

Send an action emote.

Synopsis

```
Irccd.Server.prototype.me(target, message)
```

Arguments

target (string)	A nick or a channel.
message (string)	The message to send.

Irccd.Server.prototype.message

Send a message to the specified target or channel.

Synopsis

```
Irccd.Server.prototype.message(target, message)
```

Arguments

target (string)	The target.
message (string)	The message to send.

Irccd.Server.prototype.mode

Change irccd’s user mode or a channel mode.

Synopsis

```
Irccd.Server.prototype.mode(channel, mode, limit, user, mode)
```

Arguments

channel (string)	A channel or your nicknam.
mode (string)	The new mode.
limit (string)	An optional limit.
user (string)	An optional use.
mask (string)	An optional mas.

Irccd.Server.prototype.names

Get the list of names. This function will generate the onNames event.

Synopsis

```
Irccd.Server.prototype.names(channel)
```

Arguments

channel (string)	The channel name.

Irccd.Server.prototype.nick

Change irccd’s nickname.

Synopsis

```
Irccd.Server.prototype.nick(nickname)
```

Arguments

nickname (string)	The new nickname.

Irccd.Server.prototype.notice

Send a private notice to the specified target.

Synopsis

```
Irccd.Server.prototype.notice(nickname, message)
```

Arguments

nickname (string) The target nickname.
message (string) The notice message.

Irccd.Server.prototype.part

Leave the specified channel, the reason is optional.

Synopsis

```
Irccd.Server.prototype.part(channel, reason)
```

Arguments

channel (string) The channel to leave.
reason (string) A reason (Optional, default: undefined).

Irccd.Server.prototype.toString

Convert object as a string.

Because each server has a unique identifier, this method allows adding a server a property key.

Synopsis

```
id = Irccd.Server.prototype.toString()
```

Returns

The server identifier.

Irccd.Server.prototype.topic

Change the topic of the specified channel.

Synopsis

```
Irccd.Server.prototype.topic(channel, topic)
```

Arguments

channel (string) The channel.
topic (string) The new topic.

Irccd.Server.prototype.whois

Get whois information from a user. The function will generate onWhois event.

Synopsis

```
Irccd.Server.prototype.whois(target)
```

Arguments

target (string) The target.

Irccd.System

System inspection.

Use this module if you want to inspect the system independently.

Irccd.System.env

Get a environment variable.

Synopsis

```
value = Irccd.System.env(name)
```

Arguments

name (string) The environment variable name.

Returns

The variable or an empty string.

Irccd.System.exec

Execute a system command.

Synopsis

```
Irccd.System.exec(cmd)
```

Arguments

cmd (string) The command to execute.

Irccd.System.home

Get the home directory. This function should be used with care, plugin should not use user's home to store files.

Synopsis

```
home = Irccd.System.home()
```

Returns

The user home directory.

Irccd.System.name

Get the operating system name. Returns one of:

- Linux
- Windows
- FreeBSD
- DragonFlyBSD
- OpenBSD
- NetBSD
- macOS
- Android
- Aix
- Haiku
- iOS
- Solaris
- Unknown

Synopsis

```
name = Irccd.System.name()
```

Returns

The operating system name.

Irccd.System.popen [optional]

Wrapper for popen(3) if the function is available.

Synopsis

```
handle = Irccd.System.popen(cmd, mode)
```

Arguments

cmd (string)	The command to execute.
mode (string)	The mode (e.g. r).

Throws

Irccd.SystemError on failure.

Returns

An Irccd.File object.

Irccd.System.sleep

Sleep for seconds. Suspend the execution thread.

Synopsis

```
Irccd.System.sleep(sec)
```

Irccd.System.ticks

Get the time spent from start. Get how many milliseconds spent since the irccd startup.

Synopsis

msec = Irccd.System.ticks()

Returns

The number of milliseconds.

Irccd.System.uptime

Get the system uptime. This function returns the number of seconds elapsed since the system boot up.

Synopsis

secs = Irccd.System.uptime()

Returns

The number of seconds.

Irccd.System.usleep

Sleep for milliseconds. Suspend the execution thread.

Synopsis

Irccd.System.usleep(msec)

Arguments

msec (int) The number of milliseconds.

Irccd.System.version

Get the operating system version. Result of this function is system dependant.

Synopsis

version = Irccd.System.version()

Returns

The version as a string.

Irccd.Timer

Create repetitive or one-shot timers.

The following constants properties are defined:

Single (int)	The timer is single-shot.
Repeat (int)	The timer is looping.

Irccd.Timer [constructor]

Create a new timer object but do not start it immediately.

Synopsis

```
Irccd.Timer(type, delay, callback)
```

Arguments

type (int)	Type of timer (*Irccd.Timer.Repeat* or *Irccd.Timer.Single*).
delay (int)	The interval in milliseconds.
callback (function)	The function to call.

Example:

```
var t = new Irccd.Timer(Irccd.Timer.Repeat, 1000, function () {
        // Do your action, this will be called every 1 second.
});
```

Irccd.Timer.prototype.start

Start the timer.

Synopsis

```
Irccd.Timer.prototype.start()
```

Irccd.Timer.prototype.stop

Stop the timer.

Synopsis

Irccd.Timer.prototype.stop()

Irccd.Unicode

Check for character categories.

Irccd.Unicode.isDigit

Check if the unicode character is a digit.

Synopsis

ret = Irccd.Unicode.isDigit(code)

Arguments

code (string) The code point.

Returns

True if digit.

Irccd.Unicode.isLetter

Check if the unicode character is a letter.

Synopsis

ret = Irccd.Unicode.isLetter(code)

Arguments

code (string) The code point.

Returns

True if letter.

Irccd.Unicode.isLower

Check if the unicode character is lower case.

Synopsis

```
ret = Irccd.Unicode.isLower(code)
```

Arguments

code (string) The code point.

Returns

True if lower case.

Irccd.Unicode.isSpace

Check if the unicode character is a space.

Synopsis

```
ret = Irccd.Unicode.isSpace(code)
```

Arguments

code (string) The code point.

Returns

True if space.

Irccd.Unicode.isTitle

Check if the unicode character is title case.

Synopsis

```
ret = Irccd.Unicode.isTitle(code)
```

Arguments

code (string) The code point.

Returns

True if title case.

Irccd.Unicode.isUpper

Check if the unicode character is upper case.

Synopsis

ret = Irccd.Unicode.isUpper(code)

Arguments

code (string) The code point.

Returns

True if upper case.

Irccd.Util

Various utilities.

Irccd.Util.cut

Cut a piece of data into several lines.

The argument *data* is a string or a list of strings. In any case, all strings are first splitted by spaces and trimmed. This ensure that useless whitespaces are discarded.

The argument *maxc* controls the maximum of characters allowed per line, it can be a positive integer. If undefined is given, a default of 72 is used.

The argument *maxl* controls the maximum of lines allowed. It can be a positive integer or undefined for an infinite list.

If *maxl* is used as a limit and the data can not fit within the bounds, undefined is returned.

An empty list may be returned if empty strings were found.

Synopsis

```
lines = Irccd.Util.cut(data, maxc, maxl)
```

Arguments

data (mixed)	A string or an array of strings.
maxc (int)	Max number of colums (Optional, default: 72).
maxl (int)	Max number of lines (Optional, default: undefined).

Throws

RangeError	If maxl or maxc are negative numbers.
RangeError	If one word length was bigger than maxc.
TypeError	If data is not a string or a list of strings.

Returns

A list of strings ready to be sent or undefined if the data is too big.

Irccd.Util.format

Format a string according to the template system.

See the documentation about the template format in irccd-templates(7).

Synopsis

```
str = Irccd.Util.format(input, params)
```

Arguments

input (string)	The text to update.
params (Object)	The parameters. For each keyword you want to replace in the *input*, add a new entry into the object. Note: the special *date* object key is reserved and must be set to a timestamp if desired.

Returns

The converted text.

Remarks

Be very careful when you use this function with untrusted input. Do never pass untrusted content (e.g. user message) as input parameter.

For example, the following code is unsafe:

```
function onMessage(server, channel, origin, message)
{
	// DON'T DO THIS.
	server.message(channel, Irccd.Util.format("@{red}" + message + "@{}");
}
```

If a user sends a message like ${HOME}, it will prints the user home directory, which is a high security issue if you have environment variables with passwords.

Instead, always use a literal string using a replacement with the user input:

```
function onMessage(server, channel, origin, message)
{
	// CORRECT.
	server.message(channel, Irccd.Util.format("@{red}#{message}@{}", {
		message: message
	});
}
```

Irccd.Util.splithost

Extract the host from a user, for instance with foo!~foo@localhost, *localhost* will be returned.

Synopsis

```
hostname = Irccd.Util.splithost(user)
```

Arguments

user (string) The user to split.

Returns

The host

Irccd.Util.splituser

Extract the name from a user, for instance with foo!~bar@localhost, *foo* will be returned.

Synopsis

nick = Irccd.Util.splituser(user)

Arguments

user (string) The user to split.

Returns

The nickname.

SEE ALSO

irccd(1)