



FieldCommander[®]

FCphp Programmer's Guide

FC-SW

Table of contents

About this manual	3
1. Introduction	4
1.1 FCphp and PHP	4
1.2 The FCphp setup	4
1.2.1 PHP configuration	4
1.2.2 FCphp functions	5
1.2.3 PHP extensions	5
1.2.4 Error logging	8
2. Function reference	9
2.1 General	9
2.2 Meta tags	12
2.3 Script	14
2.4 System	15
2.4.1 IP settings	15
2.4.2 DNS settings	18
2.4.3 Services	20
2.4.4 Time settings	24
2.4.5 Log settings	28
2.4.6 User settings	29
2.4.7 Certificate settings	32
2.4.8 Kiosk settings	33
2.5 Databases	34
2.5.1 Database management	34
2.5.2 Access the database	37
2.5.3 Access by SQL queries	40
Function index	43
Appendix A: Time zones	44
Notice to the user	46
Software License Agreement	46
Third-party software	47
Trademarks	48
Copyrights	48

About this manual

This document is intended for application developers and explains how to use and customize the PHP based user interface of FieldCommander.

Organisation of the documentation

Installation

Installation Guide Hardware Platform X (FCHWPXIG)

Bundles information related to the hardware, such as the installation of the device, IP networking, serial interface details and technical specifications.

Employment

User's Guide (FCSWUG)

Explains how FieldCommander is used in your application development, and documents all its features, including the databases, communication and web based configuration.

Programming

FCscript Programmer's Guide (FCSCRIPTPG)

Function reference of FieldCommander's scripting language, used to control the data flow and event management in your applications.

FCphp Programmer's Guide (FCPHPPG)

Function descriptions of the FieldCommander specific PHP interface, used to build dynamic user interfaces or reports on top of the your application.

Javascript Reference Manual (FCJSREF)

Extensive documentation of the Javascript (ECMAScript) core language used by FieldCommander for application programming.

Software options

Modbus Option Guide (FCOPT10OG)

Galaxy Option Guide (FCOPT20OG)

EIB/KNX Option Guide (FCOPT30OG)

RDA Option Guide (FCOPT40OG)

...

The "plug in" modules (optionally available) have their own documentation to explain the features, installation, configuration and programming interfaces.

1. Introduction

1.1 FCphp and PHP

The FCphp language is an important component of FieldCommander as it is used to implement your applications' user interface. It's based on the well-known PHP language though it has been extended with a large set of functions unique to FieldCommander.

PHP (PHP Hypertext Preprocessor) is a scripting language used to create dynamic Web pages. With syntax from C, Java and Perl, PHP code is embedded within HTML pages for server side execution. It is commonly used to extract data out of a database and present it on the Web page. PHP is currently one of the most popular server-side scripting systems on the Web.

We will refer to FieldCommander's hypertext language as **FCphp** to set it apart from the common PHP. Where this documentation mentions **PHP**, it refers to features or functions available in the language as developed and supported by the open-source community, based at www.php.net.

1.2 The FCphp setup

1.2.1 PHP configuration

FieldCommander uses PHP version 5.2 and contains the complete core function library and several extension (see next paragraph). A complete list of settings and variables of PHP is shown by calling the `phpinfo()` function.

Default settings

PHP runs with the recommended configuration settings with only two restrictions:

- **open_basedir** is set to the FTP root of FieldCommander. This means only files in the public part of FieldCommander's disk can be read as to protect the system files.
- several system related functions are disabled to protect the integrity of the FieldCommander software. The disabled commands are: `dl`, `shell_exec`, `passthru`, `exec`, `popen`, `system`, `proc_get_statuc`, `proc_nice`, `proc_open`, `proc_terminate` and `proc_close`.

Custom configuration

You can override the settings in the default `php.ini` by adding own configuration files. All settings can be changed except the two restrictions mentioned above.

Create a folder called "php" in the "/data" folder of your FieldCommander, and upload your custom `.ini` files here. When you restart FieldCommander, the additional `ini`-files with your own settings are used. Note that the new folder `/data/php` as seen in FTP is referred to as `/home/public/data/php` by the system.

PHP debugging

The possibility to change the PHP configuration opens the way to add remote PHP debugging capabilities to your FieldCommander. DBG-compatible editors - such as PHPEd - can help to step through your code and provide useful information while developing. PHP has to load an extension called "dbg.so-5.2.x" which can be pointed to in your custom PHP `ini` file.

Example configuration

Below is an example php.ini file which changes two configuration settings and adds remote debugging features to FieldCommander. This file should be placed in the /data/php folder.

```

;;;;;;;;;;;;;;;;;;;;;;;;;
; Resource Limits ;
;;;;;;;;;;;;;;;;;;;;;;;;;

max_execution_time = 180          ; Maximum execution time of each script, in seconds
session.cookie_lifetime = 14400  ; Lifetime in seconds of cookie or, if 0

;;;;;;;;;;;;;;;;;;;;;;;;;
; Paths and Directories ;
;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;
; Dynamic Extensions ;
;;;;;;;;;;;;;;;;;;;;;;;;;

; Directory in which the loadable extensions (modules) reside.
extension_dir = "/home/public/data/php"
extension=dbg.so-5.2.x

;;;;;;;;;;;;;;;;;;;;;;;;;
; Module Settings ;
;;;;;;;;;;;;;;;;;;;;;;;;;

[debugger]
debugger.enabled=off
debugger.profiler_enabled=off
debugger.hosts_allow=your.machine.com
debugger.hosts_deny=ALL
debugger.ports=7869, 10000/16

```

1.2.2 FCphp functions

FCphp is not limited to the core PHP functions alone, it adds an API of about 100 functions unique to FieldCommander. Most importantly, they provide access to the database and configuration settings of FieldCommander. This guide documents these functions in detail.

All FieldCommander PHP functions are prefixed with "**fc**", e.g. fcSetConfigDatabase(), in order to avoid name space conflicts. As common in PHP, the function names are not case sensitive. FCphp also has a number of predefined constants, starting with "**FC_**", eg FC_OPTION_AUDIO. They are listed with their respective functions. Note that variables *are* case sensitive.

1.2.3 PHP extensions

The PHP engine is based on the complete version 5.2 distribution with all default add ins. The following modules are already installed to provide additional functionality and chapter 1.2.4 explains how you can add your own modules.

Curl file transfer

Curl allows you to connect and communicate to many different types of servers with many different types of protocols. It supports the http, https, ftp, gopher, telnet, dict, file, and ldap protocols. libcurl also supports HTTPS certificates, HTTP POST, HTTP PUT, FTP uploading, HTTP form based upload, proxies, cookies, and user+password authentication.

CURL support	enabled
CURL version	7.16.0 + OpenSSL

For a full overview of supported functions, see: <http://www.php.net/curl>

More on Curl: <http://curl.haxx.se/>

GD library

GD is an open source code library for the dynamic creation of images by programmers. GD creates PNG, JPEG and GIF images, among other formats. GD is commonly used to generate charts, graphics, thumbnails, and most anything else, on the fly.

PHP with GD is not limited to creating just HTML output. It can also be used to create and manipulate image files in a variety of different image formats, including gif, png, jpg, wbmp, and xpm. Even more convenient, PHP can output image streams directly to a browser.

GD Version	2.0.34 (bundled with PHP)
GIF Read Support	yes
GIF Create Support	yes
JPG Support	yes
PNG Support	yes
WBMP Support	yes
XBM Support	yes

For a full overview of supported functions, see: <http://www.php.net/gd>

More on the GD library: <http://www.boutell.com/gd/>

OpenSSL

OpenSSL is a implementation of the Secure Sockets Layer (SSL v2/v3) and Transport Layer Security (TLS v1) protocols as well as a full-strength general purpose cryptography library. This module uses the functions for generation and verification of signatures and for sealing (encrypting) and opening (decrypting) data.

OpenSSL support	enabled
OpenSSL version	0.9.7g

PDO database drivers

The PHP Data Objects (PDO) extension defines a lightweight, consistent interface for accessing databases in PHP. Each database driver that implements the PDO interface can expose database-specific features as regular extension functions. FieldCommander includes drivers for SQLite3 for connecting to its local database and MySQL to connect to remote databases.

SQLite library	3317
MySQL library	5045

For a full overview of supported functions, see: <http://www.php.net/pdo>

To open a connection to the local SQLite database on FieldCommander:

```
$dbLocal = new PDO('sqlite:/home/public/database/example.db');
```

To open a connection to an external MySQL database:

```
$dbRemote = new PDO('mysql:host=www.example.com;dbname=databasename',
'username', 'password');
```

Session handling support

Session support in PHP consists of a way to preserve certain data across subsequent accesses. This enables you to build more customized applications and increase the appeal of your web site.

Auto start	off
Name	PHPSESSID
Use cookies	on
Use only cookies	off
Use trans SID	0

XML parser

The XML parser is a stream oriented that requires setting handlers to deal with the structure that the parser discovers in the document. It lets you parse, but not validate, XML documents. It supports three source character encodings also provided by PHP: US-ASCII, ISO-8859-1 and UTF-8. UTF-16 is not supported.

libXML Version	2626
libXML streams	yes
SimpleXML support	yes
SimpleXML schemas	yes
DOM/XML	no
XML namespace support	yes
XLS	enabled
libXLS version	1117
EXSLT	enabled
libexslt version	1117

For a full overview of supported functions, see: <http://www.php.net/xml>

More on XLM and PHP: <http://www.zend.com/zend/art/parsing.php>

More on SimpleXML: <http://www.zend.com/php5/articles/php5-simplexml.php>

More on XSLT: <http://xmlsoft.org/XSLT/>

Zlib compression

This module enables you to transparently read and write gzip (.gz) compressed files, through versions of most of the filesystem functions which work with gzip-compressed files.

ZLib version	113
Output compression	no

For a full overview of supported functions, see: <http://www.php.net/zlib>
More on Zlib: <http://www.gzip.org/zlib/>

1.2.4 Error logging

Warning or error messages which occur while parsing PHP files are (for security reasons) not shown in the page but logged to the file "php.log", located in the "/log" directory available over FTP. When you'd expect output from a PHP page you are working on but a blank page appears, chances are that a parse error occurred. Refer to the *User's Guide* for information how to access the file.

2. Function reference

2.1 General

fcLastError

fcLastError - Get error code of the last error in FCphp functions

Description

```
long fcLastError ()
```

Returns the error code of the last error in FCphp functions. It can be converted to a readable error message with **fcErrorMessage()**.

fcErrorMessage

fcErrorMessage - Get error code of the last error in FCphp functions

Description

```
string fcErrorMessage ( long errorcode )
```

Returns an error message related to the given code. The latest error code can be retrieved with **fcLastError()**.

fcGetSystemOptionsObject

fcGetSystemOptionsObject - Get installed options of the system

Description

```
object fcGetSystemOptionsObject ()
```

Returns an object which contains TRUE for an element when the option is available and FALSE if the option is not installed.

Example:

```
$options = fcGetSystemOptionsObject();  
  
if ( $options->HWP5 ) $hw = "Platform 5";  
if ( $options->HWP9 ) $hw = "Platform 9";  
if ( $options->HWP10 ) $hw = "Platform 10";  
if ( $options->SWS ) $sw = "Standard";  
if ( $options->SWA ) $sw = "Advanced";  
if ( $options->SWE ) $sw = "Extended";  
  
print "Hardware platform: $hw<br>";  
print "Software edition: $sw<br>";
```

option	description
HWP1	system is based on Hardware platform 1
HWP2	system is based on Hardware platform 2
HWP5	system is based on Hardware platform 5
HWP8	system is based on Hardware platform 8
HWP9	system is based on Hardware platform 9
HWP10	system is based on Hardware platform 10
SWS	system uses Software edition Standard *)
SWA	system uses Software edition Advanced *)
SWE	system uses Software edition Extended *)
OPT010	system has Modbus ASCII option
OPT015	system has Modbus RTU option
OPT020	system has Galaxy option
OPT030	system has EIB/KNX option
OPT040	system has RDA option
OPT050	system has Zigbee option
OPT130	system has Wavetrend option
OPT150	system has EnOcean option
OPT160	system has ESPA option
OPT170	system has Kadex option
OPT180	system has X10 option
OPT210	system has SOIP option (8x)
OPT211	system has SOIP option (32x)
OPT220	system has VOIP option
OPT250	system has DMX512 option
OPT280	system has extended SNMP agent
AUDIO	system has Audio support
CAM	system has Camera support
MMS	system has MMS (multimedia messaging) support
KIOSK	system has built-in display and touchscreen

*) for a list of features per edition, refer to the *User's Guide*

fcTrigger

fcTrigger - trigger a event in FieldCommander JavaScript

Description

bool **fcTrigger** (long eventsource [, long eventtype])

Returns **TRUE** for success, **FALSE** on failure. All events from FCphp are of type "evPhp" unless otherwise specified.

Example:

```
fcTrigger(1); // fire event 1 of evPhp
fcTrigger(10, evTimeout); // fire event 10 of evTimeout
```

JavaScript:

```
function main ()
{
  sleep ();
}

function handleEvent (event_source, event_type)
{
  if ( (event_source == 1) && (event_type == evPhp) )
    writeLog ("Event 1 triggered by FCphp");
}
```

fcRebootDevice

fcRebootDevice - shut down and restart FieldCommander

Description

bool **fcRebootDevice** ()

Returns **TRUE**. Calling this function will cause the currently active FCscript to be aborted. It takes about two minutes before the system is up again.

Example:

```
fcRebootDevice(); // shut down and restart
```

fcSyncDisk

fcSyncDisk - commit delayed writes to disk

Description

bool **fcSyncDisk** ()

Returns **TRUE**. Force committing of any delayed reads or writes to disk. Normally FieldCommander may delay write actions to save resources and improve overall performance.

Example:

```
fcSyncDisk(); // commit delayed writes
```

2.2 Meta tags

Meta tags are used to share information between your application script and user interface. As they are kept 'in memory', the tags cannot be used for permanent storage.

You can define our own tags but FieldCommander provides information through predefined system tags as well, see the table below for built-in tags:

meta tag name	description
_IPADDRESS	IP address of this FieldCommander, nnn.nnn.nnn.nnn
_HOSTNAME	hostname assigned to this FieldCommander
_TIME	local system time, HH:mm:ss (24 hour clock)
_DATE	current date, mm/dd/yyyy
_VERSION	version of the software (major.minor.patchlevel)
_SERIALNUMBER	serial number of this FieldCommander
_MODEL	model type of this FieldCommander
_LOAD	CPU load in percent, average of last minute
_LOAD5	CPU load in percent, average of last 5 minutes
_LOAD15	CPU load in percent, average of last 15 minutes
_UPTIME	system uptime, formatted as text
_UPTIMESEC	system uptime in seconds
_MEMFREE	available system memory in kilo bytes (kB)
_MEMUSED	allocated system memory in kilo bytes (kB)
_DISKFREE	available disk space in kilo bytes (kB)
_DISKUSED	allocated disk space in kilo bytes (kB)
_RS232_PORTS	number of RS232 ports in the system
_RS485_PORTS	number of RS485 ports in the system
_USB_PORTS	number of USB ports in the system
_NDU<bufname>	number of data units in buffer <bufname> E.g.: "_NDUmybuffer" for buffer with name "mybuffer"

fcGetMetaValue

fcGetMetavalue - Get current meta value

Description

string **fcGetMetaValue** (string metaname)

Returns the string value of the given metaname when it exists or **FALSE** otherwise. Note that the name is case sensitive.

Example:

```
// show the current IP address of FieldCommander
print fcGetMetaValue("_IPADDRESS");
```

fcSetMetaValue

fcSetMetavalue - Set new meta value

Description

bool **fcSetMetaValue** (string metaname, string metavalue)

Writes a value to the meta table. When the *meta tag name* was not used before, a new entry is added to the table. System meta tags cannot be set. Returns **TRUE** on success, **FALSE** otherwise.

Notes:

- *meta tag name* is case sensitive
- *meta tag name* is can contain a number or texts up to 64 characters
- *meta tag value* is can contain a number or texts up to 256 characters

Example:

```
// set and print a custom meta tag
fcSetMetaValue("MyTag", "my first meta tag");
print fcGetMetaValue("MyTag");
```

fcClearMetaValue

fcClearMetaValue - Delete the metavalue

Description

bool **fcClearMetaValue** (string metaname)

Delete a value from the meta table. System meta tags cannot be removed. Returns **TRUE** on success, **FALSE** otherwise.

2.3 Script

fcGetCurrentScript

fcGetCurrentScript - Get the name of the currently running FieldCommander script

Description

string **fcGetCurrentScript** ()

Returns the name of the currently running FieldCommander, or **FALSE** when no script is active. Note that the name does not include the file extension ".jse".

fcGetScripts

fcGetScripts - Get names of all available FieldCommander scripts as an array

Description

string **fcGetScripts** ()

Returns the names of all available FieldCommander scripts as an array or **FALSE** when no scripts are installed.

fcScriptStart

fcScriptStart - Start the execution of a FieldCommander script

Description

bool **fcScriptStart** (string scriptname)

Starts the given script name, including the file extension .jse. Returns **TRUE** on success, **FALSE** otherwise. This script will also start automatically when FieldCommander is powered on. Note that you can only start a script if no other script is running.

fcScriptStop

fcScriptStop - Stop the currently running FieldCommander script

Description

bool **fcScriptStop** ()

Stop the script which is currently being executed. Returns **TRUE** on success, **FALSE** otherwise.

When you stop the script execution of FieldCommander, all status information in the script application is lost. Ending of scripts is generally only done during development.

2.4 System

2.4.1 IP settings

fcGetHostName

fcGetHostName - Get the host name

Description

string **fcGetHostName** ()

Returns the currently configured host name of FieldCommander. It is used to identify itself to DNS servers, amongst others.

The actual host name can be retrieved through the metaname "_HOSTNAME". This can be different from the configured name when it was changed but the settings are not applied yet.

See also **fcGetMetaValue()**

fcGetSerialNumber

fcGetSerialNumber - Get the serial number

Description

string **fcGetSerialNumber** ()

Returns the serial number of FieldCommander which is also its MAC address.

An alternative way to fetch the serial number is through the metaname "_SERIALNUMBER".

See also **fcGetMetaValue()**

fcGetModel

fcGetModel - Get the model name

Description

string **fcGetModel** ()

Returns the FieldCommander model name.

An alternative way to fetch the model name is through the metaname "_MODEL".

See also **fcGetMetaValue()**

fcGetFirmware

fcGetFirmware - Get the firmware version

Description

string **fcGetFirmware** ()

Returns the currently installed firmware version of FieldCommander.

An alternative way to fetch the firmware version is through the metaname "_VERSION".

fcGetIpAddress

fcGetIpAddress - Get the IP address of FieldCommander

Description

string **fcGetIpAddress** ()

Returns the currently configured IP address of FieldCommander.

The actual IP address can be retrieved through the metaname "_IPADDRESS". This can be different from the configured address when it was changed but the settings are not applied yet.

fcGetIpSubnet

fcGetIpSubnet - Get the IP subnet mask

Description

string **fcGetIpSubnet** ()

Returns the IP subnet mask of FieldCommander.

fcGetIpGateway

fcGetIpGateway - Get the IP address of the default gateway

Description

string **fcGetIpGateway** ()

Returns the IP address of the default gateway.

fcGetDhcp

fcGetDhcp - Check if DHCP is enabled

Description

bool **fcGetDhcp** ()

Returns **TRUE** when IP configuration through DHCP is enabled, **FALSE** otherwise.

fcGetArpPing

fcGetArpPing - Check if ARP+PING is enabled

Description

bool fcGetArpPing ()

Returns **TRUE** when changing the IP address of FieldCommander using ARP+PING is enabled, **FALSE** otherwise.

fcSetIpAddress()

fcSetIpAddress - Set the IP address for the system

Description

bool **fcSetIpAddress** (string address)

The system has to be restarted to have effect.

Return **TRUE** for success, **FALSE** on failure

fcSetIpSubnet

fcSetIpSubnet - Set the Subnet mask for the system

Description

bool **fcSetIpSubnet** (string subnetmask)

The system has to be restarted to have effect.

Return **TRUE** for success, **FALSE** on failure

fcSetIpGateway

fcSetIpGateway- Set the default gateway address

Description

bool **fcSetIpGateway** (string gateway)

Return **TRUE** for success, **FALSE** on failure

fcSetArpPing

fcSetArpPing - Enable ARP mode

Description

bool **fcSetArpPing** (bool mode)

Return **TRUE** for success, **FALSE** on failure

fcSetDhcp

fcSetDhcp - Enable system to automatically receive a dynamic IP-address

Description

bool **fcSetDhcp** (bool mode)

NOTE: set after other IP settings

Return **TRUE** for success, **FALSE** on failure

2.4.2 DNS settings

fcGetDns

fcGetDns - Check if DNS resolution is enabled

Description

bool **fcGetDns** ()

Returns **TRUE** when Domain Name resolution through DNS is enabled, **FALSE** otherwise.

fcGetDnsPrimary

fcGetDnsPrimary - Get the IP address of the primary DNS server

Description

string **fcGetDnsPrimary** ()

Returns the IP address of the primary DNS server.

fcGetDnsSecondary

fcGetDnsSecondary - Get the IP address of the secondary DNS server

Description

string **fcGetDnsSecondary** ()

Returns the IP address of the secondary DNS server. This DNS server will be queried when the primary server did not respond or cannot be reached.

fcGetDomainName

fcGetDomainName - Get the domain name

Description

string **fcGetDomainName** ()

Returns the domain name of FieldCommander. The domain name is used to identify itself to DNS servers, amongst others.

fcSetHostName

fcSetHostName - Set hostname for the system.

Description

bool **fcSetHostName** (string hostname)

Return **TRUE** for success, **FALSE** on failure

fcSetDomainName

fcSetDomainName - Set the domain name.

Description

bool **fcSetDomainName** (string domainname)

Return **TRUE** for success, **FALSE** on failure

fcSetDns

fcSetDns - Enable DNS name resolution

Description

bool **fcSetDns** (bool mode)

Return **TRUE** for success, **FALSE** on failure

fcSetDnsPrimary

fcSetDnsPrimary - Set the primary DNS server address

Description

bool **fcSetDnsPrimary** (string address)

Return **TRUE** for success, **FALSE** on failure

fcSetDnsSecondary

fcSetDnsSecondary - Set the secondary DNS server address.

Description

bool **fcSetDnsSecondary** (string address)

Return **TRUE** for success, **FALSE** on failure

2.4.3 Services

fcGetFtp

fcGetFtp - Check if the FTP server is enabled

Description

bool **fcGetFtp** ()

Returns **TRUE** when the FTP server is enabled, **FALSE** otherwise.

fcGetFtpPort

fcGetFtpPort - Get the FTP server port number

Description

long **fcGetFtpPort** ()

Returns TCP/IP port number of the FTP server. Default is port 21.

fcGetHttpAuth

fcGetHttpAuth - Check if HTTP authentication mode is applied

Description

long **fcGetHttpAuth** ()

Returns the HTTP authentication mode applied. See **fcSetHttpAuth()** for a list of the settings.

fcGetSmtplib

fcGetSmtplib - Check if outgoing e-mail is enabled

Description

boolean **fcGetSmtplib** ()

Returns **TRUE** when outgoing e-mail over SMTP is enabled, **FALSE** otherwise.

fcGetSmtplibServer

fcGetSmtplibServer - Get the SMTP relay server

Description

string **fcGetSmtplibServer** ()

Returns the IP address or name of the SMTP relay server used for outgoing e-mail.

fcGetSmtpport

fcGetSmtpport - Get the port number of the SMTP relay server

Description

long **fcGetSmtpport** ()

Returns the TCP/IP port number of the SMTP relay server. Default is port 25.

fcGetSnmpport

fcGetSnmpport - Check if SNMP agent enabled

Description

boolean **fcGetSnmpport** ()

Returns **TRUE** when SNMP is enabled, **FALSE** otherwise.

fcGetSnmpport

fcGetSnmpport - Get the port number of the SNMP agent

Description

long **fcGetSnmpport** ()

Returns the TCP/IP port number of the SNMP agent. Default is port 161.

fcGetSnmppublic

fcGetSnmppublic - Get the read community name of SNMP

Description

string **fcGetSnmppublic** ()

Returns the public read community name of the SNMP agent. Default is "public".

fcGetSnmpprivate

fcGetSnmpprivate - Get the write community name of SNMP

Description

string **fcGetSnmpprivate** ()

Returns the private write community name of the SNMP agent. Default is "private".

fcGetUpnp

fcGetUpnp - Check if UPnP service is enabled

Description

boolean **fcGetUpnp** ()

Returns **TRUE** when UPnP is enabled, **FALSE** otherwise.

fcSetFtpPort

fcSetFtpPort - Set the FTP portnumber

Description

bool **fcSetFtpPort** (long portnumber)

Return **TRUE** for success, **FALSE** on failure

fcSetFtp

fcSetFtp - Enable/disable the FTP server

Description

bool **fcSetFtp** (bool mode)

Return **TRUE** for success, **FALSE** on failure

fcSetHttpAuth

fcSetHttpAuth - Enable/disable HTTP authentication to directories

Description

bool **fcSetHttpAuth** (long mode)

value	description
HTTP_AUTH_SECURE	authentication only in /www/secure and /www-secure/secure
HTTP_AUTH_HTTP_ROOT	authentication in complete /www root
HTTP_AUTH_HTTPS_ROOT	authentication in complete /www-secure root

Return **TRUE** for success, **FALSE** on failure

See **fcSetUserRights()**

fcSetSntp

fcSetSntp - Enable/disable SMTP server

Description

bool **fcSetSntp** (bool mode)

Return **TRUE** for success, **FALSE** on failure

fcSetSmtpPort

fcSetSmtpPort - Set the SMTP port for outgoing mail

Description

bool **fcSetSmtpPort** (long portnumber)

NOTE: enable SMTP first.

Return **TRUE** for success, **FALSE** on failure

fcSetSmtpSender

fcSetSmtpSender - Set return e-mail address (used as "from" field)

Description

bool **fcSetSmtpSender** (string emailaddress)

Return **TRUE** for success, **FALSE** on failure

fcSetSmtpServer

fcSetSmtpServer - Set the SMTP server address or name.

Description

bool **fcSetSmtpServer** (string address)

Return **TRUE** for success, **FALSE** on failure

fcSetSnmpp

fcSetSnmpp - Enable/disable SNMP agent

Description

bool **fcSetSnmpp** (bool mode)

Return **TRUE** for success, **FALSE** on failure

fcSetSnmppPort

fcSetSnmppPort - Set the port for the SNMP agent

Description

bool **fcSetSnmppPort** (long portnumber)

NOTE: enable SNMP first.

Return **TRUE** for success, **FALSE** on failure

fcSetSnmpPublic

fcSetSnmpPublic - Set the public read community name for SNMP agent

Description

bool **fcSetSnmpPublic** (string community)

Return **TRUE** for success, **FALSE** on failure

fcSetSnmpPrivate

fcSetSnmpPrivate - Set the private write community name for SNMP agent

Description

bool **fcSetSnmpPrivate** (string community)

Return **TRUE** for success, **FALSE** on failure

fcSetUpnp

fcSetUpnp - Enable/disable UPnP service

Description

bool **fcSetUpnp** (bool mode)

Return **TRUE** for success, **FALSE** on failure

2.4.4 Time settings

The NTP functions are only available in the Advanced and Extended software editions of FieldCommander.

fcGetNtp

IN SELECTED MODELS

fcGetNtp - Check if Network Time server is enabled

Description

bool **fcGetNtp** ()

Returns **TRUE** when NTP is enabled, **FALSE** otherwise.

fcGetNtpPrimary

IN SELECTED MODELS

fcGetNtp - Get the primary NTP server address

Description

string **fcGetNtpPrimary** ()

Returns the address or name of the NTP server used for atomic time synchronization.

fcGetNtpSecondary

IN SELECTED MODELS

fcGetNtpSecondary - Get the secondary NTP server address

Description

string **fcGetNtpPrimary** ()

Returns the alternative address or name of the NTP server used for atomic time synchronization.

fcGetDst

fcGetDst - Check if Daylight Saving Time is enabled

Description

string **fcGetDst** ()

Returns **TRUE** when DST is enabled, **FALSE** otherwise.

fcGetTimeZone

fcGetTimeZone - Get the current Time Zone

Description

string **fcGetTimeZone** ()

Return a string with the time zone. See **Appendix A** for a complete listing of the available time zones.

fcGetTime

fcGetTime - Get the current time

Description

string **fcGetTime** ()

Return a string with the time (HH:mm:ss)

fcGetDate

fcGetDate - Get the current date

Description

string **fcGetDate** ()

Return a string with the date (mm/dd/yyyy)

fcSetNtp**IN SELECTED MODELS**

fcSetNtp - Enable NTP mode to obtain time from NTP server.

Description

bool **fcSetNtp** (bool mode)

Return **TRUE** for success, **FALSE** on failure

fcSetNtpPrimary**IN SELECTED MODELS**

fcSetNtpPrimary- Set the primary NTP server address

Description

bool **fcSetNtpPrimary** (string address)

Return **TRUE** for success, **FALSE** on failure

fcSetNtpSecondary**IN SELECTED MODELS**

fcSetNtpSecondary - Set the secondary NTP server address

Description

bool **fcSetNtpSecondary** (string address)

Return **TRUE** for success, **FALSE** on failure

fcSetTimeZone

fcSetTimeZone - Set local time zone

Description

bool **fcSetTimeZone** (string timezone)

Return **TRUE** for success, **FALSE** on failure

See **Appendix A** for a complete listing of the available time zones.

Example: `// set time zone to Amsterdam (GMT+1)`
 `fcSetTimeZone("Amsterdam");`

fcSetDst

fcSetDst - Enable Daylight Saving Time mode.

Description

bool **fcSetDst** (bool mode)

Return **TRUE** for success, **FALSE** on failure

fcSetTime

fcSetTime - Adjust the system time

Description

bool **fcSetTime** (string time)

Return **TRUE** for success, **FALSE** on failure. Time string is in 24 hour clock format: "HH:mm:ss".

Example: // set time to 2:10 PM
 fcSetTime("14:10:00");

fcSetDate

fcSetDate - Adjust the system date

Description

bool **fcSetDate** (string date)

Return **TRUE** for success, **FALSE** on failure. Date string is in the following format: "mm/dd/yyyy".

Example: // set date to October 1st 2005
 fcSetDate("10/01/2005");

2.4.5 Log settings

fcGetLogLimit

fcGetLogLimit - Get the maximum logfile size for FCscript

Description

long **fcGetLogLimit** ()

Return an long with the maximum log size

fcGetLogLevel

fcGetLogLevel - Get the current FCscript log level

Description

long **fcGetLogLevel** ()

Return an long with the log level

fcGetLogEmpty

fcGetLogEmpty - Check if FCscript logfile is empty

Description

bool **fcGetLogEmpty** ()

Returns **TRUE** if empty, **FALSE** otherwise

fcClearLog

fcClearLog - Clear the FCscript logfile

Description

bool **fcClearLog** ()

Return **TRUE** for success, **FALSE** on failure

fcSetLogLimit

fcSetLogLimit - Set the maximum logsize for FCscript

Description

bool **fcSetLogLimit** (long size)

Return **TRUE** for success, **FALSE** on failure

fcSetLogLevel

fcSetLogLevel - Set the level of logging for FCscript

Description

bool **fcSetLogLevel** (long level)

Return **TRUE** for success, **FALSE** on failure

value	description
FC_LOG_CRITICAL	log critical errors only, advised in production environment
FC_LOG_ERROR	log errors
FC_LOG_WARNING	log warnings and errors
FC_LOG_INFO	log information, warnings and errors
FC_LOG_DEBUG	log all messages (debug level)

Example: // set FCscript to log just errors
 fcSetLogLevel(FC_LOG_ERROR);

2.4.6 User settings

FieldCommander features user profiles for controlled access to its FTP server, web server(s) and system configuration. The *User's Guide* explains the use of profiles in detail. Be aware that several function require that the a user context is set, through **fcSetUser()**, who is allowed to manage the user settings.

fcSetUser

fcSetUser - Set the active user

Description

array **fcSetUser** (string username, string password)

Authenticates the given user and sets it as the active user. Return **TRUE** on success, **FALSE** otherwise.

The functions **fcUserAdd()**, **fcUserDelete()**, **fcGetUserRights()** and **fcSetUserRights()** require that a user is active and is allowed to manage the user settings. It is controlled by the FC_USER_EDIT switch in the user rights.

fcGetUsers

fcGetUsers - Get the list of users

Description

array **fcGetUsers** ()

Returns array of user names, or **FALSE** when no users

fcCheckUserPassword

fcCheckUserPassword - Check a user's password

Description

array **fcCheckUserPassword** (string username, string password)

Return **TRUE** on success, **FALSE** otherwise.

fcSetUserPassword

fcSetUserPassword - Change user password

Description

bool **fcSetUserPassword** (string username, string oldpass, string newpass)

Return **TRUE** for success, **FALSE** on failure

Example: // set a more secure password for Peter
 fcSetUserPassword ("peter", "peter", "5exAde@2");

fcUserAdd

fcUserAdd - Add a new user to the system

Description

bool **fcUserAdd** (string username, string password)

Return **TRUE** for success, **FALSE** on failure

This function requires that the active user is allowed to manage the user settings, see **fcSetUser()**.

Example: // add a user profile for Peter
 fcSetUser ("admin", "admin");
 fcUserAdd ("peter", "peter");

fcUserDelete

fcUserDelete - Remove user from system

Description

bool **fcUserDelete** (string username)

Return **TRUE** for success, **FALSE** on failure

This function requires that the active user is allowed to manage the user settings, see **fcSetUser()**.

Note that you cannot delete the active user and the "admin" profile.

Example: // remove Peter's account
 fcSetUser ("admin", "admin");
 fcUserDelete ("peter");

fcGetUserRights

fcGetUserRights - Get the system rights of the user

Description

long **fcGetUserRights** (string username)

Returns a long value which is a combination (addition) of the users's rights or **FALSE** on failure. See **fcSetUserRights()** for a list of the available settings.

This function requires that the active user is allowed to manage the user settings, see **fcSetUser()**.

Example:

```
// check is Peter is allowed to update the system software
fcSetUser ("admin", "admin");
$rights = fcGetUserRights ("peter");
if ($rights & FC_FTP_UPDATE)
    writeLog("You can upload the new system software via FTP to /update");
else
    writeLog("Please ask your system administrator to update the system software");
```

fcSetUserRights

fcSetUserRights - Set the system rights for the user

Description

bool **fcSetUserRights** (string username, long rights [, string newpassword])

Return **TRUE** for success, **FALSE** on failure

value	description
FC_USER_EDIT	allows to add, edit and remove user profiles *)
FC_FTP_SCRIPT	FTP read/write permissions in /script
FC_FTP_DATABASE	FTP read/write permissions in /database
FC_FTP_WWW	FTP read/write permissions in /www and /www-secure
FC_FTP_LOG_DATA	FTP read/write permissions in /log and /data
FC_FTP_UPDATE	FTP read/write permissions in /update
FC_HTTP_ROOT	may authenticate for /www and /www-secure
FC_HTTP_SECURE	may authenticate for /www and /www-secure but also /www/secure and /www-secure/secure

*) Be very cautious who to give rights to manage user profiles. Users with these rights can edit and remove **your profile** too!

Returns a long value which is a combination (addition) of the available options.

This function requires that the active user is allowed to manage the user settings, see **fcSetUser()**.

Note that you cannot change the rights of the active user and the "admin" profile.

```
Example:    // give Peter access to the /update directory
            fcSetUser ("admin", "admin");
            $rights = fcGetUserRights ("peter");
            fcSetUserRights ("peter", $rights + FC_FTP_UPDATE);
```

See **fcSetHttpAuth()**

2.4.7 Certificate settings

FieldCommander implements a PHP function to generate self-signed certificates on the fly. When changing IP address of the system, this function can be called to avoid "domain name mismatch" errors in client applications (i.e. web browser and Java VM).

fcCreateCertificate

fcCreateCertificate - Generate a digital host certificate for SSL

Description

bool **fcCreateCertificate** (string country, string state, string locality, string organisation, string orgunit, string hostname [, int days])

Return **TRUE** for success, **FALSE** on failure

The *country* should contain a 2-character international country code (ISO 3166). The *state*, *locality*, *organisation* and *organisational unit* (all 64 characters maximum) provide information about the issuing party. The *hostname* should be the IP address or DNS name by which the **end user** (browser) accesses FieldCommander. When *days* is omitted, the certificate is valid for a year, otherwise define the number of days of validity here.

Notes:

- the current time and date of FieldCommander become part of the certificate; make sure they are set up correctly before creating certificate
- the digital host certificate is encrypted with a 1024 bit RSA private key
- the new certificate replaces the currently installed one and will be used for all SSL connections after FieldCommander has been restarted.

```
Example:    // create certificate for 10.0.0.1, valid for three years
            fcCreateCertificate("NL", "Noord Brabant", "Roosendaal",
                               "CER International bv", "", "10.0.0.1", 1095);

            // create certificate based on current IP address
            fcCreateCertificate("NL", "Noord Brabant", "Roosendaal",
                               "CER International bv", "", fcGetMetaValue("_IPADDRESS"));
```

2.4.8 Kiosk settings

The Kiosk functions are only applicable for the Extended software edition of FieldCommander.

fcGetKioskButtons

IN SELECTED MODELS

fcGetKioskButtons - Check if the kiosk browser buttons are enabled

Description

bool **fcGetKioskButtons** ()

Returns **TRUE** if enabled, **FALSE** otherwise

fcGetKioskKeyboard

IN SELECTED MODELS

fcGetKioskKeyboard - Check if kiosk keyboard is enabled

Description

bool **fcGetKioskKeyboard** ()

Returns **TRUE** if enabled, **FALSE** otherwise

fcGetKioskScreensaver

IN SELECTED MODELS

fcGetKioskScreensaver - Check if kiosk screen saver is enabled

Description

bool **fcGetKioskScreensaver** ()

Returns **TRUE** if enabled, **FALSE** otherwise

fcSetKioskButtons

IN SELECTED MODELS

fcSetKioskButtons - Enable buttons in kiosk mode

Description

bool **fcSetKioskButtons** (bool mode)

Return **TRUE** for success, **FALSE** on failure

fcSetKioskKeyboard

IN SELECTED MODELS

fcSetKioskKeyboard - Enable keyboard in kiosk mode

Description

bool **fcSetKioskKeyboard** (bool mode)

Return **TRUE** for success, **FALSE** on failure

fcSetKioskScreensaver**IN SELECTED MODELS**

fcSetKioskScreensaver - Enable screen saver in kiosk mode

Descriptionbool **fcSetKioskScreensaver** (bool mode)Return **TRUE** for success, **FALSE** on failure

2.5 Databases

FieldCommander's database functions provide a comfortable web interface to edit data used in an application. The tables can be accessed from FCscript too so they serve as a flexible and easy way to share data between the application and the user interface.

2.5.1 Database management

The list below has simple functions to create, edit and remove databases, tables, columns and rows.

command	function description
fcGetConfigDatabases()	Get a list of available databases
fcConfigDatabaseAdd()	Add a new database
fcConfigDatabaseDelete()	Delete an existing database
fcSetConfigDatabase()	Set database to use by the following commands
fcGetConfigTables()	Get a list of tables in current database
fcConfigTableAdd()	Add a new table to the current database
fcConfigTableDelete()	Remove a table from the current database
fcGetConfigTableColumns()	Get list of all columns in a table
fcConfigTableColumnAdd()	Add a column to a table
fcConfigTableColumnDelete()	Remove a column form a table
fcConfigTableClear()	Remove all rows in a table

You can also your own SQL queries to created and manage tables if you please, check 2.5.3 for the SQL functions.

fcGetConfigDatabases

fcGetConfigDatabases - Get a list of available databases

Description

array **fcGetConfigDatabases** ()

Returns array of database names, or **FALSE** when no databases

fcConfigDatabaseAdd

fcConfigDatabaseAdd - Add a new database to the system

Description

bool **fcConfigDatabaseAdd** (string database)

Create a new database to be placed in the /database folder in the FTP root of FieldCommander. The database names can be up to 32 characters long and should not contain quote (') or double quote (") characters.

Return **TRUE** for success, **FALSE** on failure

fcConfigDatabaseDelete

fcConfigDatabaseDelete - Remove a database from the system

Description

bool **fcConfigDatabaseDelete** (string database)

Return **TRUE** for success, **FALSE** on failure

fcSetConfigDatabase

fcSetConfigDatabase - Set the current database to be used in PHP page

Description

bool **fcSetConfigDatabase** (string database)

Return **TRUE** for success, **FALSE** on failure. This function sets the given database as the current. All subsequent database calls in the PHP page use this database.

fcGetConfigTables

fcGetConfigTables - Get a list of available tables in the current database

Description

array **fcGetConfigTables** ()

Returns array of table names, or **FALSE** when no tables

fcConfigTableAdd

fcConfigTableAdd - Add a new table to the current database

Description

bool **fcConfigTableAdd** (string tablename, string columnname)

Creates a new table in the current database and sets the first column as index. Table and column names can be up to 32 characters long and should not contain quote (') or double quote (") characters.

Return **TRUE** for success, **FALSE** on failure

Equivalent to SQL: `CREATE TABLE tablename (columnname ANY UNIQUE ON CONFLICT REPLACE);`

fcConfigTableColumnAdd

fcConfigTableColumnAdd - Add a new column to a table

Description

bool **fcConfigTableColumnAdd** (string tablename, string columnname)

Adds an columns to the given table. Column names can be up to 32 characters long and should not contain quote (') or double quote (") characters.

Return **TRUE** for success, **FALSE** on failure

Equivalent to SQL (where newcolumns is the original column list with the new columnname):
`BEGIN TRANSACTION; CREATE TEMPORARY TABLE tablename_backup (<columns>);
INSERT INTO tablename_backup SELECT <columns> FROM tablename;
DROP TABLE tablename; CREATE TABLE tablename (<columns>);
INSERT INTO tablename (<newcolumns>) SELECT <columns> FROM tablename_backup;
DROP TABLE tablename; COMMIT;`

fcConfigTableDelete

fcConfigTableDelete - Remove an existing table from the active database

Description

bool **fcConfigTableDelete** (string tablename)

Removes a columns from the given table.

Return **TRUE** for success, **FALSE** on failure

Equivalent to SQL (where newcolumns is the original column list without columnname):
`BEGIN TRANSACTION; CREATE TEMPORARY TABLE tablename_backup (<columns>);
INSERT INTO tablename_backup SELECT <columns> FROM tablename;
DROP TABLE tablename; CREATE TABLE tablename (<columns>);
INSERT INTO tablename (<newcolumns>) SELECT <columns> FROM tablename_backup;
DROP TABLE tablename; COMMIT;`

fcGetConfigTableColumns

fcGetConfigTableColumns - Get a list with column names of a table

Description

array **fcGetConfigTableColumns** (string tablename)

Returns array of column names, **FALSE** on failure

fcConfigTableClear

fcConfigTableClear - Clear all rows of data from a table

Description

bool **fcConfigTableClear** (string tablename)

Removes all rows from the table. This will permanently loose all its information but keep the table's column definition.

Return **TRUE** for success, **FALSE** on failure

2.5.2 Access the database

The list below has simple functions to set and get information in the tables:

command	function description
fcSetConfigDatabase()	Set database to use by the following commands
fcGetConfigTableRow()	Retrieve data from a specific row indicated by its index value and return an array containing the rows data
fcGetFirstConfigTableRow()	Retrieve data in the first row of a table and return an array containing the rows data
fcGetNextConfigTaleRow()	Retrieve data in the next row of the result set and return an array containing the row's data
fcGetLastConfigTableRow()	Retrieve data in the last row of a table and return an array containing the row's data
fcGetPreviousConfigTableRow()	Retrieve data in the previous row in the result set and return an array containing the row's data
fcSetConfigTableRow()	Add a row or change data in a row of a table
fcDeleteConfigTableRow()	Delete a row from a table

You can also your own SQL queries to access tables if you please, check 2.5.3 for the SQL functions. Where possible, the function descriptions here include their SQL equivalents.

fcSetConfigDatabase

fcSetConfigDatabase - Set the current database to be used in PHP page

Description

bool **fcSetConfigDatabase** (string database)

Return **TRUE** for success, **FALSE** on failure. This function sets the given database as the current. All subsequent database calls in the PHP page use this database.

fcGetConfigTableRow

fcGetConfigTableRow - Get the row with values by passing the index

Description

array **fcGetConfigTableRow** (string tablename, string indexvalue)

Returns array of column values for success, **FALSE** on failure. The array is associative which means it uses strings (the column names) for its index value.

Equivalent to SQL: `SELECT * FROM tablename WHERE <firstcolumnname> = 'indexvalue';`

Example:

```
// get row from table
fcSetConfigDatabase("exampleDB");
$row = fcGetConfigTableRow("exampleTBL", "John");

var_dump($row);

// returns:
// array(3) { ["username"]=> string(4) "John"
//           ["age"]=> string(2) "35"
//           ["eye color"]=> string(4) "blue" }
//

// access the data per element by column name
print $row["username"] . " is " . $row["age"] . " years old.";

// show the array's data and column names
while ( list($key, $cell) = each($row) )
    print "$key: $cell<br/>";

// returns:
// username: John
// age: 35
// eye color: blue

// show the array's data
foreach($row as $cell)
    print "$cell<br/>";

// returns:
// John
// 35
// blue
```

fcGetFirstConfigTableRow

fcGetFirstConfigTableRow - Get the first row of a table

Description

array **fcGetFirstConfigTableRow** (string tablename)

Returns array of column values for success, **FALSE** on failure. The array is associative which means it uses strings (the column names) for its index value, see **fcGetConfigTableRow()**.

fcGetNextConfigTableRow

fcGetNextConfigTableRow - Get the next table row

Description

array **fcGetNextConfigTableRow** (string tablename)

Returns array of column values for success, **FALSE** on failure. The array is associative which means it uses strings (the column names) for its index value, see **fcGetConfigTableRow()**.

fcGetPreviousConfigTableRow

fcGetPreviousConfigTableRow - Get the previous table row

Description

array **fcGetPreviousConfigTableRow** (string tablename)

Returns array of column values for success, **FALSE** on failure. The array is associative which means it uses strings (the column names) for its index value, see **fcGetConfigTableRow()**.

fcGetLastConfigTableRow

fcGetLastConfigTableRow - Get the last row in a table

Description

array **fcGetLastConfigTableRow** (string tablename)

Returns array of column values for success, **FALSE** on failure. The array is associative which means it uses strings (the column names) for its index value, see **fcGetConfigTableRow()**.

fcSetConfigTableRow

fcSetConfigTableRow - Add a new row to a table

Description

bool **fcSetConfigTableRow** (string tablename, array row [, bool replacemode])

Return **TRUE** for success, **FALSE** on failure.

If the index value (first column) does not match a row that is already in the table, a new row is added. When the row is in the table it will be overwritten, unless **replacemode** is set of FALSE. Fields that have no matching member in the array are filled with NULL.

The array **row** is associative which means it uses strings (the column names) for its index value.

Example:

```
// create and initiate a table row
$row = array("username"=> "Julie", "age"=> "26", "eye color"=> "blue");

var_dump($row);
// returns:
// array(3) { ["username"]=> string(5) "Julie"
//           ["age"]=> string(2) "26"
//           ["eye color"]=> string(4) "blue" }

// add row table, replace when "Julie" is already in table
fcSetConfigDatabase("exampleDB");
fcSetConfigTableRow("exampleTBL", $row, TRUE);

// alternative way to create an associate array:
$newrow["username"] = "Andrea";
$newrow["age"] = "22";
$newrow["eye color"] = "brown";
```

fcDeleteConfigTableRow

fcDeleteConfigTableRow - Remove a row from a table

Description

bool **fcDeleteConfigTableRow** (string tablename, string indexvalue)

Return **TRUE** for success, **FALSE** on failure

fcConfigTableColumnDelete

fcConfigTableColumnDelete - Removes a complete column from a table

Description

bool **fcConfigTableColumnDelete** (string tablename, string columnname)

Return **TRUE** for success, **FALSE** on failure

2.5.3 Access by SQL queries

When you need more flexibility than the previous table function provide, you can access the database using SQL (SQL92) queries. Please refer to the *User's Guide* for information regarding to the supported SQL commands.

command	function description
fcSetConfigDatabase()	Set database to use by the following commands
fcConfigTableQuery()	Execute a true SQL query
fcGetConfigTableRow()	Get (next) row from the SQL query result set
fcConfigTableQueryFinalize()	Free the current result set manually

fcSetConfigDatabase

fcSetConfigDatabase - Set the current database to be used in PHP page

Description

bool **fcSetConfigDatabase** (string database)

Return **TRUE** for success, **FALSE** on failure. This function sets the given database as the current. All subsequent database calls in the PHP page use this database.

fcConfigTableQuery

fcConfigTableQuery - Get a list of rows specified by a SQL query.

Description

bool **fcConfigTableQuery** (string SQL)

Return **TRUE** for success, **FALSE** on failure

Call this function to execute the SQL query. When the query is supposed to return results, these are fetched by calling **fcGetConfigTableRow()**.

Refer to the *User's Guide* for information regarding to the supported SQL commands.

See also **fcGetConfigTableRow()**

fcGetConfigTableRow

fcGetConfigTableRow - Retrieve a row of data from the result set

Description

array **fcGetConfigTableRow** ()

Returns array of column values of next row in result set from query, **FALSE** when no (more) results.

Example:

```
// dump Users table from Demo database as HTML table with column header
//
fcSetConfigDatabase("Demo");
fcConfigTableQuery("SELECT * FROM Users");
```

```
print "<table>";
$firstrow = true;
while ( $row = fcGetConfigTableRow() ) {
    print "<tr>";
    if ( $firstrow ) {
        while ( list($key, $cell) = each($row) )
            print "<td>$key</td>";
        print "</tr><tr>";
        $firstrow = false;
    }
    foreach($row as $cell) {
        print "<td>$cell</td>";
    }
    print "</tr>";
}
print "</table>";
```

See also **fcConfigTableQuery()**

fcConfigTableQueryFinalize

fcConfigTableQueryFinalize - Free the memory of the result set

Description

bool **fcConfigTableQueryFinalize** ()

Return **TRUE** for success, **FALSE** on failure

Free the memory occupied by the existing result set. This function is automatically called before **fcConfigTableQuery()** so normally it is not necessary to call it by yourself.

Function index

fcCheckUserPassword	30	fcGetSnmpPort	21
fcClearLog	28	fcGetSnmpPrivate	21
fcClearMetaValue	13	fcGetSnmpPublic	21
fcConfigDatabaseAdd	34, 35	fcGetTime	25
fcConfigDatabaseDelete	34, 35	fcGetTimeZone	25
fcConfigTableAdd	34, 36	fcGetUpnp	22
fcConfigTableClear	34, 37	fcGetUserRights	29, 31, 32
fcConfigTableDelete	34, 36	fcGetUsers	29
fcConfigTableQuery	41, 42	fcLastError	9
fcConfigTableQueryFinalize	41, 42	fcRebootDevice	11
fcCreateCertificate	32	fcScriptStart	14
fcDeleteConfigTableRow	37, 40	fcScriptStop	14
fcErrorString	9	fcSetArpPing	17
fcGetArpPing	16, 17	fcSetConfigDatabase	5, 34, 35, 37, 38, 40, 41
fcGetConfigDatabases	34, 35	fcSetConfigTableRow	37, 39, 40
fcGetConfigTableColumns	34, 37	fcSetDate	27
fcGetConfigTableQueryRow	41, 42	fcSetDhcp	17, 18
fcGetConfigTableRow	37-39	fcSetDns	19
fcGetConfigTables	34, 35	fcSetDnsPrimary	19
fcGetCurrentScript	14	fcSetDnsSecondary	19
fcGetDate	25, 26	fcSetDomainName	19
fcGetDhcp	16	fcSetDst	27
fcGetDns	18	fcSetFtpPort	22
fcGetDnsPrimary	18	fcSetHostName	19
fcGetDnsSecondary	18	fcSetHttpAuth	20, 22, 32
fcGetDomainName	18	fcSetKioskButtons	33
fcGetDst	25	fcSetKioskKeyboard	33
fcGetFirmware	15, 16	fcSetKioskScreensaver	34
fcGetFirstConfigTableRow	37, 39	fcSetLogLevel	29
fcGetFtp	20	fcSetLogLimit	28
fcGetFtpPort	20	fcSetMetaValue	13
fcGetHostName	15	fcSetNtp	26
fcGetHttpAuth	20	fcSetNtpPrimary	26
fcGetIpAddress	16	fcSetNtpSecondary	26
fcGetIpGateway	16	fcSetSntp	22, 23
fcGetIpSubnet	16	fcSetSntpPort	23
fcGetKioskButtons	33	fcSetSntpSender	23
fcGetKioskKeyboard	33	fcSetSntpServer	23
fcGetKioskScreensaver	33	fcSetSnmp	23
fcGetLastConfigTableRow	37, 39	fcSetSnmpPort	23, 24
fcGetLogEmpty	28	fcSetSnmpPrivate	24
fcGetLogLevel	28	fcSetSnmpPublic	24
fcGetLogLimit	28	fcSetTime	27
fcGetMetaValue	13, 15, 32	fcSetTimeZone	26
fcGetModel	15	fcSetUpnp	24
fcGetNextConfigTableRow	39	fcSetUser	29-32
fcGetNtp	24, 25	fcSetUserPassword	30
fcGetNtpPrimary	25	fcSetUserRights	22, 29, 31, 32
fcGetNtpSecondary	25	fcSyncDisk	11, 12
fcGetPreviousConfigTableRow	37, 39	fcTrigger	11
fcGetScripts	14	fcUserAdd	29, 30
fcGetSerialNumber	15	fcUserDelete	29-31
fcGetSntp	20		
fcGetSntpPort	21		
fcGetSntpServer	20		
fcGetSnmp	21		

Appendix A: Time zones

value	description
Kwajalein	GMT -12:00 Eniwetok, Kwajalein
Samoa	GMT -11:00 Midway Island, Samoa
Hawaii	GMT -10:00 Hawaii
Alaska	GMT -09:00 Alaska
Pacific	GMT -08:00 Pacific Time (US & Canada), Tijuana
Arizona	GMT -07:00 Arizona
Mountain	GMT -07:00 Mountain Time (US & Canada)
Costa_Rica	GMT -06:00 Central America
Central	GMT -06:00 Central Time (US & Canada)
Mexico	GMT -06:00 Mexico City
GMT+6	GMT -06:00 Saskatchewan
Bogota	GMT -05:00 Bogota, Lima, Quito
Eastern	GMT -05:00 Eastern Time (US & Canada)
East-Indiana	GMT -05:00 Indiana (East)
Atlantic	GMT -04:00 Atlantic Time (Canada)
Caracas	GMT -04:00 Caracas, La Paz
Santiago	GMT -04:00 Santiago
St_Johns	GMT -03:30 Newfoundland
Buenos	GMT -03:00 Buenos Aires, Georgetown
GMT+3	GMT -03:00 Greenland
GMT+2	GMT -02:00 Mid-Atlantic
Azores	GMT -01:00 Azores
Cape_Verde	GMT -01:00 Cape Verde Is.
Casablanca	GMT Casablanca, Monrovia
London	GMT Dublin, Edinburgh, Lisbon, London
Amsterdam	GMT +01:00 Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna
Prague	GMT +01:00 Belgrade, Bratislava, Budapest, Ljubljana, Prague
Paris	GMT +01:00 Brussels, Copenhagen, Madrid, Paris
Sofia	GMT +01:00 Sarajevo, Skopje, Sofija, Vilnius, Warsaw, Zagreb
GMT-2	GMT +01:00 West Central Africa
Athens	GMT +02:00 Athens, Istanbul, Minsk
Bucharest	GMT +02:00 Bucharest
Cairo	GMT +02:00 Cairo
Harare	GMT +02:00 Harare, Pretoria
Helsinki	GMT +02:00 Helsinki, Riga, Tallinn
Israel	GMT +02:00 Jerusalem

value	description
Baghdad	GMT +02:00 Baghdad
Kuwait	GMT +03:00 Kuwait, Riyadh
Moscow	GMT +03:00 Moscow, St. Petersburg, Volgograd
Nairobi	GMT +03:00 Nairobi
Tehran	GMT +03:30 Tehran
Muscat	GMT +04:00 Abu Dhabi, Muscat
Baku	GMT +04:00 Baku, Tbilisi, Yerevan
Kabul	GMT +04:30 Kabul
Yekatarinburg	GMT +05:00 Yekatarinburg
Karachi	GMT +05:00 Islamabad, Karachi, Tashkent
Calcutta	GMT +05:30 Calcutta, Chennai, Mumbai, New Delhi
Katmandu	GMT +05:45 Katmandu
Almaty	GMT +06:00 Almaty, Novosibirsk
Dhaka	GMT +06:00 Astana, Dhaka
Jayapura	GMT +06:00 Sri Jayawardenepura
Rangoon	GMT +06:30 Rangoon
Bangkok	GMT +07:00 Bangkok, Hanoi, Jakarta
Krasnoyarsk	GMT +07:00 Krasnoyarsk
Hong_Kong	GMT +08:00 Beijing, Chongping, Hong Kong, Urumgi
Irkutsk	GMT +08:00 Irkutsk, Ulaan Bataar
Kuala_Lumpur	GMT +08:00 Kuala Lumpur, Singapore
Perth	GMT +08:00 Perth
Taipei	GMT +08:00 Taipei
Tokyo	GMT +09:00 Osaka, Sapporo, Tokyo
Seoul	GMT +09:00 Seoul
Yakutsk	GMT +09:00 Yakutsk
Adelaide	GMT +09:30 Adelaide
Darwin	GMT +09:30 Darwin
Brisbane	GMT +10:00 Brisbane
Canberra	GMT +10:00 Canberra, Melbourne, Sydney
Guam	GMT +10:00 Guam, Port Moresby
Hobart	GMT +10:00 Hobart
Vladivostok	GMT +10:00 Vladivostok
Magadan	GMT +11:00 Magadan, Solomon Is., New Caledonia
Auckland	GMT +12:00 Auckland, Wellington
Fiji	GMT +12:00 Fiji, Kamchatka, Marshall Is.
GMT-13	GMT +13:00 Nuku'alofa

Notice to the user

IMPORTANT - READ CAREFULLY:

This CER End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and CER International bv for the CER software product FieldCommander, which includes computer software and may include associated media, printed materials, and "online" or electronic documentation ("SOFTWARE"). The SOFTWARE also includes any updates and supplements to the original SOFTWARE provided to you by CER. Any software provided along with the SOFTWARE that is associated with a separate end-user license agreement is licensed to you under the terms of that license agreement. By installing, copying, downloading, accessing, or otherwise using the SOFTWARE, you agree to be bound by the terms of this EULA. If you do not agree to the terms of this EULA, do not install or use the SOFTWARE; you may, however, return it to your place of purchase for a full refund within 10 days of the date you acquired it.

Software License Agreement

The SOFTWARE is protected by copyright laws and international copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE is licensed, not sold.

GRANT OF LICENSE

You may install, use, access, display, run, or otherwise interact with ("RUN") one copy of the SOFTWARE, or any prior version for the same operating system, on your FieldCommander. You are obtaining no rights on the SOFTWARE except those given in this limited license.

OWNERSHIP

You may not translate, reverse engineer, decompile, or disassemble the SOFTWARE, except and only to the extent that such activity is expressly permitted by applicable law notwithstanding this limitation. You may not rent, lease, or lend the SOFTWARE. You may not modify the SOFTWARE or merge all or any part of the SOFTWARE in another program. The SOFTWARE is licensed as a single product. Its component parts may not be separated for use on more than one COMPUTER.

NO TRANSFER

You may not sublicense the SOFTWARE. You may not transfer the SOFTWARE to a third party unless you cease all use of it, transfer all copies of it and accompanying Documentation, and the transferee agrees to be bound by the terms of this Agreement.

TERM

This License shall continue for as long as you use the SOFTWARE. However, it will terminate if you fail to comply with any of its term or conditions. Upon such termination you must immediately cease using the SOFTWARE and must follow CER International's instructions regarding return of the Software. ALL DISCLAIMERS HEREIN SHALL SURVIVE TERMINATION.

LIMITED WARRANTY

CER International warrants that (a) the HARDWARE will be free from defects in materials and workmanship under normal use and service for a period of one year from the date of receipt; and (b) the SOFTWARE accompanying the HARDWARE will perform substantially in accordance with the accompanying Product Manual(s) for a period of 90 days from the date of receipt. Any implied warranties on the HARDWARE and SOFTWARE are limited to one (1) year and 90 days, respectively. Some states/jurisdictions do not allow limitations on duration of an implied warranty, so the above limitation may not apply to you.

CUSTOMER REMEDIES

CER International's entire liability and your exclusive remedy shall be, at CER International's option, either (a) return of the price paid or (b) repair or replacement of the HARDWARE or SOFTWARE that does not meet the above Limited Warranty. This Limited Warranty is void if failure of the HARDWARE or SOFTWARE has resulted from accident, abuse, or misapplication. Any replacement HARDWARE or SOFTWARE will be warranted for the remainder of the original warranty period or 30 days, whichever is longer.

NO OTHER WARRANTIES

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CER INTERNATIONAL DISCLAIMS ALL OTHER WARRANTIES, EITHER EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WITH RESPECT TO THE SOFTWARE, THE ACCOMPANYING PRODUCT MANUAL(S) AND WRITTEN MATERIALS, AND ANY ACCOMPANYING HARDWARE. THE LIMITED WARRANTY CONTAINED HEREIN GIVES YOU SPECIFIC LEGAL RIGHTS. YOU MAY HAVE OTHERS WHICH VARY FROM STATE/JURISDICTION TO STATE/JURISDICTION

NO LIABILITY FOR CONSEQUENTIAL DAMAGES

TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CER INTERNATIONAL AND ITS SUPPLIERS SHALL NOT BE LIABLE FOR ANY OTHER DAMAGES WHATSOEVER (INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, LOSS OF BUSINESS INFORMATION, OR OTHER PECUNIARY LOSS) ARISING OUT OF THE USE OF OR INABILITY TO USE THIS CER INTERNATIONAL PRODUCT, EVEN IF CER INTERNATIONAL HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. IN ANY CASE, CER INTERNATIONAL'S ENTIRE LIABILITY UNDER ANY PROVISION OF THIS AGREEMENT SHALL BE LIMITED TO THE AMOUNT ACTUALLY PAID BY YOU FOR THE SOFTWARE.

GENERAL

This License is the entire agreement between us, supersedes any other agreement or discussions, oral or written and may not be changed except by a written signed agreement. CER International accepts no liability for any damages whatsoever arising out of the use of or inability to use this software, direct and/or indirect. This License shall be governed by and construed in accordance with the laws of the Netherlands. If any provision of this License is declared by a Court of competent jurisdiction to be invalid, illegal, or unenforceable, such provision shall be severed from the License and the other provisions shall remain in full force and effect. The parties have requested that this Agreement and all documents contemplated hereby be drawn up in English.

GOVERNMENT RESTRICTED RIGHTS

The HARDWARE, SOFTWARE and user documentation are provided with RESTRICTED RIGHTS. Use, duplication, or disclosure by the Government is subject to restrictions. Manufacturer is CER International bv, Roosendaal, the Netherlands. Should you have any questions concerning this Agreement, or if you desire to contact CER International for any reason, please write: Customer Service Dept., CER International bv, PO Box 258, NL 4700 AG Roosendaal, The Netherlands.

Third-party software

This product contains software under the GNU General Public License and other open source licenses. Copies of these licenses and information about obtaining the GPL source code is available on the web at <http://www.cer.com>. If you would like a copy of the GPL source code contained in this product shipped to you on CD for costs only covering preparing and mailing the CD, contact CER International bv, FieldCommander Product Management, PO Box 258, 4700 AG Roosendaal, The Netherlands.

Portions copyright © 1995,1998,1999,2000 by Jef Poskanzer <jef@acme.com>. All rights reserved.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Portions copyright © 1996 - 2001, Daniel Stenberg, <daniel@haxx.se>. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Trademarks

CER and FieldCommander are registered trademarks of CER International bv.

All other product names and services identified throughout this book are trademarks or registered trademarks of their respective companies. They are used throughout this manual in editorial fashion only and for the benefit of such companies. No such uses, or the use of any trade name, is intended to convey endorsement or other affiliation with this manual.

Copyrights

Copyright © 2008 CER International bv. All rights reserved.

No part of this publication may be reproduced in any form, or stored in a database or retrieval system, or transmitted or distributed in any form by any means, electronic, mechanical photocopying, recording, or otherwise, without the prior written permission of CER International bv, except as permitted by the Copyright Act of 1976 and except that program listings may be entered, stored and executed in a computer system.

THE INFORMATION AND MATERIAL CONTAINED IN THIS MANUAL ARE PROVIDED "AS IS," WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY WARRANTY CONCERNING THE ACCURACY, ADEQUACY, OR COMPLETENESS OF SUCH INFORMATION OR MATERIAL OR THE RESULT TO BE OBTAINED FROM USING SUCH INFORMATION OR MATERIAL, NEITHER CER INTERNATIONAL BV NOR THE AUTHORS SHALL BE RESPONSIBLE FOR ANY CLAIMS ATTRIBUTABLE TO ERRORS, OMISSIONS, OR OTHER INACCURACIES IN THE INFORMATION OR MATERIAL CONTAINED IN THIS MANUAL, AND IN NO EVENT SHALL CER INTERNATIONAL BV OR THE AUTHORS BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OF SUCH INFORMATION OR MATERIAL.

from Device to Enterprise

CER[®]

The latest documentation is available
from <http://www.cer.com>

FieldCommander is a product of:

CER International bv
Postbus 258
NL 4700 AG Roosendaal
The Netherlands
TEL: +31 (0)165 557417
FAX: +31 (0)165 562151
<http://www.cer.com>

Part no. FCPHPPG, revision 3.5.33