NAME

ECTk – the electrochemical tool kit

SYNOPSIS

ectk

DESCRIPTION

This manual page gives an overview of ECTk, the electrochemical tool kit. ECTk is a shell-like application based on Tcl implementing several routines to conduct electrochemical experiments. When invoked, ECTk reads Tcl commands from standard input, or a script file, and prints command results and error messages to standard output. It runs until the **exit** command is invoked. If there exists a file **ectkrc.tcl** in the home directory of the user, ectk evaluates the file as a Tcl script just before reading the first command from standard input. ECTk loads as well the Tk library.

USAGE

The electrochemical tool kit allows to define a sequence of electrochemical experiments. The software communicates with an IO board using the comedi library (see its documentation for more information).

The sequence of processes to be processed is defined in the ECTk process loop. The **ps** command displays this loop.

The definition and execution of an electrochemical experiment involves typically the following three steps:

1. Definition of the ECTk process loop.

2. Running the experiment.

3. Post processing.

The ECTk process loop is defined using the ECTk commands for electrochemical process definition. The loop always has to start with a **cell -on** and end with a **cell -off** process. For example, the following command sequence defines a cyclic voltammetry experiment:

cell -on
cv -eInitial -0.5 **-eFinal** 1.1 **-scanRate** 0.10 **-nCycles** 2
cell -off

The **ps**, or the **log** command can be used to check the ECTk process loop. The command **clear** clears the loop.

Note that for each process defined in the ECTk loop, a new Tcl command is defined in order to allow to manipulate it (for example changing parameters). The **ps** lists the name of these commands.

The experiment is started with the command **start** and ends once the **cell -off** procedure is terminated. During the experiment, the ECTk console, invoked with the command **console**, can be used to monitor the progress. The commands **monitor** and **graph** can be used to display and graph the measured values.

Once an experiment is finished, the acquired data can be graphed and saved. Therefore, use the **-plot** and **-save** options of the Tcl command associated to the electrochemical process you would like to graph or save, or use the **graph** and **save** commands.

VARIABLES

ECTk defines in the namespace **::ectk::** the following Tcl variables:

::ectk::chanCtrl Channel number of the IO board used to control the potential

::ectk::chanVfo Channel number of the IO board used to read the potential (Vfo = potential follower)

::ectk::chanIfo Channel number of the IO board used to read the current (Ifo = current follower

::ectk::comediFile Comedi file (e.g. /dev/comedi0) used to communicate with the IO board. The default value assigned is /dev/comedi0.

::ectk::commentStr String added in front of text in data files. Default value is "# ".

::ectk::expName If defined, gives the name of the overall experiment. This variable is used by several commands (e.g. **log**, **expSave**) in order to assign a name to the experiment.

::ectk::graphUpdateInterval Minimal time, in seconds, between two successive updates of the graph displayed by the **graph** command. Default value is 0.5 sec.

::ectk::IOboardUpdateInterval Minimal time, in Milli-seconds, between two successive updates of the IO board AO channel. Default value is 1 milisec.

::ectk::liscence Contains the licence agreement for ECTk.

::ectk::monitorUpdateInterval Minimal time, in seconds, between two successive updates of the monitor displayed by the **monitor** command. Default value is 1.0 sec.

::ectk::plotArgs Variable used internally by the various plot commands. The user should not change it directly but rather use the **plotStyle** command.

::ectk::plotVars Variable used internally by the various plot commands. The user should not change it directly but rather use the **plotVars** command.

::ectk::procList Array containing all defined procedures. The index is the PID of the procedure as displayed by the **ps** command.

::ectk::user Name of the user of ECTk. By default this variables contains the value of **tcl_platform(user)**. This variable is used by the **log**, and **ps** commands.

ELECTROCHEMICAL DATA

During execution of an experiment (initiated by the **start** command), ECTk updates the following global Tcl variables containing information about the measured data:

t time in Seconds at which the data were updated (measured since the start of the current ECProcedure).

E potential in Volts.

I current in Amperes.

Q charge in Coulombs.

Qneg
negative charge in Coulombs.

Qpos
positive charge in Coulombs.

These variables are updated with the rate controlled by the variable **::ectk::IOboardUpdateInterval**, which gives the minimal time interval between two successive updates.

Except **E**, all variables are read-only variables. If during execution of an experiment **E** is assigned to a value, then at the next update of the IOboard, this value will be used to output on the AO channel. This is particularly useful in preProcess operations (to pre-polarize the working-electrode for example).

ELECTROCHEMICAL PROCESSES

In ectk version 1, the following electrochemical processes are defined:

cell	allows to switch on and off the potensiostat.
cv	defines a cyclic voltammetry experiment.
ca	defines a chronoamperommetry experiment.
step	defines a potential step.
pulse	defines a pulse train.
trigger	defines a trigger in order to stop an electrochemical experiment based on a user defined condition.
userProc	defines a user defined procedure.

For a generic description of electrochemical processes see the *process* man-page. For a detailed description of each electrochemical process, see its own man-page.

COMMANDS

Besides the standard Tcl commands, ectk defines its own specific commands:

clear	removes electrochemical processes from the ECTk procedure loop.
IODevice	configures the interface to the IO board.
connect	connects to the IO board using a configuration file.
cons	writes to the ECTk console (see bellow).
console	opens the ECTk console. By default, the electrochemical processes display information about their status (starting, specific executions, etc...) on this console. The user can as well write to it using the **cons** command.
IOStat	returns statistics about IO operations done.
log	gives detailed information of the defined processes in the ECTk process loop.
lrec	lists the ec records loaded in the memory of ectk.
monitor	opens the ECTk monitor displaying the potential and current.
notes	manipulates notes associated with an experiment.
ps	lists the sequence of processes defined.
start	starts the sequence of defined electrochemical processes. Note that if no **cell -on** process was defined, this command has no effect.
stop	stops immediately the ongoing experiment.
terminate	terminates an ECTk process.
wait	waits during a defined time interval.
who	displays the ECTk process loop and the list of ec records.

For a detailed description of each command, see its own man-page.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

process(1ectk), procedure(1ectk), tk(3tk), tcl(3tcl), comedi(7)

NAME

ECTk procedure – generic description of ECTk procedures

DESCRIPTION

This manual page gives an overview of the common features of the ECTk procedures. ECTk procedures are ECTk processes able to record and manipulate electrochemical data. Besides all features described in the ectk *process* man-page, ECTk procedures have dedicated options for data recording and manipulation.

ELECTROCHEMICAL DATA RECORDS

An ECTk procedure contains a list of electrochemical data records. Each record contains:

- time (in seconds) of the record; time origin is the moment where the procedure was activated

- potential (in Volts)

- current (in Amperes)

- anodic charge (in Coulomb)

- cathodic charge (in Coulomb)

- total charge (in Coulomb)

If available to the user, the record is stored in the Tcl namespace having the same name as the Tcl command associated to the ECTk procedure. In this namespace are defined the following variables **t**, **E**, **I**, **Qpos**, **Qneg**, **Q** and **mark**. For example, if a ECTk procedure with the associated Tcl command **foo** exists, then a namespace **::foo** is as well created. The full qualifier of the variable **t** would then be **::foo::t**.

The options (see below) **-loop**, **-readFirstRecord** and **-readNextRecord** allows to loop through the list of data records. Note that these options run in the namespace of the ECTk procedure.

USAGE

Each ECTk procedure can be manipulated using it's associate command. The command created during the procedure creation can be seen in the list provided by the **ps** command.

The following is the list of all valid options of ECTk procedures (note that the options already described in the ECTk *process* man-page are not repeated here):

-cut *cutCondition*

selects a sub-set of data from the ECTk procedure by applying a cut condition. The cut condition has to be written in valid Tcl syntax. The data record is available through the variables of the associated namespace.

For example **-cut {$E>**-1.1**}** selects only data for potentials greater than -1.1 Volts.

This option is used together with other options, such as **-plot** for example.

-dtRecord *timeInterval*

sets the time interval (in sec) between two successive recording of data. Default value is 0.1s which means that every 0.1s data acquired by ECTk are recorded (stored) in the electrochemical data record of the ECTk procedure.

-dtOut *timeInterval*

sets the time interval (in sec) between two outputs of the recored data. This option influences the data output from the **-plot** and **-save** options. It does not affect the stored data.

-loop *loopScript* ?*beginMark*? ?*endMark*?
invokes a Tcl script for each electrochemical data record in the ECTk procedure. During this loop, the data are available to the script via the Tcl namespace associated to the ECTk procedure. The *loopScript* runs in this namespace. Can be combined with the **-cut** option. If the *beginMark* and *endMark* are specified, only the records from *beginMark* to *endMark* are considered.

For example **-loop {cons $I}** will output to the ECTk console the current for each data record.

-max ?*beginMark*? ?*endMark*?
returns the potential with the largest current value. Further, the record with the maximal current is available in the Tcl namespace associated to the ECTk procedure. Can be combined with the **-cut** option. If the *beginMark* and *endMark* are specified, only the records from *beginMark* to *endMark* are scanned for the maximal current value.

-min ?*beginMark*? ?*endMark*?
returns the the potential with the smallest current value. Further, the record with the minimal current is available in the Tcl namespace associated to the ECTk procedure. Can be combined with the **-cut** option. If the *beginMark* and *endMark* are specified, only the records from *beginMark* to *endMark* are scanned for the minimal current value.

-plot ?*beginMark*? ?*endMark*?
plots the electrochemical data stored in the ECTk procedure. Can be combined with the **-cut** option. If the *beginMark* and *endMark* are specified, only the records from *beginMark* to *endMark* are plotted.

The coordinates to be used for the X and Y axes can be defined with the **-x** and **-y** options.

-print ?*beginMark*? ?*endMark*?
prints the recored data in form of a table. Can be combined with the **-cut** option. If the *beginMark* and *endMark* are specified, only the records from *beginMark* to *endMark* are printed.

-readFirstRecord
copies the first record from the ECTk procedure to the Tcl namespace associated to the ECTk procedure. Returns the time of the record.

-readNextRecord
copies the next record from the ECTk procedure to the Tcl namespace associated to the ECTk procedure. Returns the time of the record or -1 if the end of the data is reached.

-save *fileName* ?*beginMark*? ?*endMark*?
saves to a file the electrochemical data stored in the ECTk procedure. Can be combined with the **-cut** option. If the *beginMark* and *endMark* are specified, only the records from *beginMark* to *endMark* are saved.

-stat returns statistics about the recored data.

-x ?*xCoordinate*?
defines the X-coordinate for plotting. As *xCoordinate* any valid Tcl mathematical expression computing the coordinate is valid. All variables of the associated namespace are available. If no *xCoordinate* is specified, returns the current X-coordinate in use.

For example **-x {$E}** will use the potential as X-coordinate.

-y ?*yCoordinate*?
defines the Y-coordinate for plotting. As *yCoordinate* any valid Tcl mathematical expression computing the coordinate is valid. All variables of the associated namespace are available. If no *yCoordinate* is specified, returns the current Y-coordinate in use.

For example **-y {log(abs($I))}** will use the logarithm of the current as Y-coordinate.

When the procedure command is invoked without any option, it returns the process ID.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ectk(1ectk), process(1ectk), ps(3ectk)

NAME

ECTk process – generic description of ECTk processes

DESCRIPTION

This manual page gives an overview of the common features of the ECTk processes. ECTk processes are the basic objects in ECTk. An ECTk process is created and loaded into the ECTk process loop with its associated command such as **trigger** for example. Once created, each ECTk process receives a unique process ID. Further a Tcl command is created in order the process can be manipulated.

USAGE

Each ECTk process can be manipulated using it's associate command. The command created during the process creation can be seen in the list provided by the **ps** command. The following is the list of all valid options of ECTk processes:

-help returns a short description of the valid options.

-info returns information about the process.

-name returns the name of the process.

-nRecords
returns -1, unless the ECTk process is able to store electrochemical data, in which case it returns the number of records stored.

-quiet process does no longer display info about its status on the ECTk console.

-verbose
process displays info about its status on the ECTk console.

-preProcess *script*
Tcl script to be executed at the moment the process gets activated.

-postProcess *script*
Tcl script to be executed at the moment the process gets terminated.

When the process command is invoked without any option, it returns the process ID.

Some ECTk processes are able to further record and manipulate electrochemical data. These processes are called ECTk *procedure*s and are further described in their dedicated man-page.

EXAMPLE

The following command will create a trigger process which stops the process cv4 (which is supposed to be loaded in the ectk process loop) after 0.2 sec:

```
trigger -postProcess {terminate [cv4]} \
    -triggerScript {if {$t>0.2} {return 0} else {return 1}}
```

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ectk(1ectk), procedure(1ectk), ps(3ectk)

NAME

IODevice – configures the interface to the IO board.

SYNOPSIS

IODevice *option* ?*arg*?

DESCRIPTION

Configures the interface to the IO board. ECTk uses the comedi library in order to communicate with the IO board. ECTk can handle IO boards up to 16 AI and 16 AO channels. The **IODevice** command is used to configure the relevant parameters of the comedi interface. The comedi library communicates with the IO board with a particular comedi device file /dev/comediN, where N is 0,1,2,3, etc. The name of the comedi device file is stored by ectk in the Tcl variable **::ectk::comediFile**.

Note that the IO board has to be properly installed with its associated comedi driver in order ECTk can communicate with it. See the comedi man pages to learn how to configure your board with comedi. Once this configuration done, the communication with board in ECTk is straight forward.

The **IODevice** command calls directly functions from the comedi library.

The valid *option*s are:

IODevice -board
returns the name of the connected IO board

IODevice -connect ?*configFile*?
connects to the IO board. If the argument *configFile* is provided, **IOdevice** first evaluates *configFile*. The configuration file is a valid Tcl script which should define the specific configuration variables for the potensiostat to be driven. Displays an error message in case the board cannot be connected.

IODevice -deviceFile ?*comediFile*?
If no argument *comediFile* is provided, returns the content of the Tcl variable **::ectk::comediFile**, which by defaults is set to /dev/comedi0 when ECTk is started. If the argument *comediFile* is set, **IOdevice** assigns **::ectk::comediFile** to the provided argument. The variable **::ectk::comediFile** is used to determine the file to be used to communicate with the IO board.

IODevice -driver
returns the loaded driver for the connected IO board

IODevice -read *chan*
reads the value of the AI channel *chan* from the connected board. Multiplies by **$::ectk::AI_gain(***chan***)** and adds **$::ectk::AI_offset(***chan***)** before returning the value.

IODevice -write *chan data*
writes **$::ectk::AO_gain(***chan***)***data*+**$::ectk::AO_offset(***chan***)** to the channel *chan* from the connected IO board.

VARIABLES

The following Tcl variables are used by ECTk for IO operations:

::ectk::AI_gain	Tcl array containing the gain of each AI channel. Default values are 1.
::ectk::AO_gain	Tcl array containing the gain of each AO channel. Default values are 1.
::ectk::AI_offset	Tcl array containing the offset of each AI channel. Default values are 0.

::ectk::AO_offset	Tcl array containing the offset of each AO channel. Default values are 0.
::ectk::chanCtrl	AO channel number of the IO board used to control the potential. Default value is 1.
::ectk::chanVfo	AI channel number of the IO board used to read the potential (Vfo = potential follower). Default value is 1.
::ectk::chanIfo	AI channel number of the IO board used to read the current (Ifo = current follower) Default value is 2.
::ectk::comediFile	Name of the comedi device file associated with the IO bard. Set be default to /dev/comedi0.
::ectk::IOboardUpdateInterval	Minimal time, in Milli-seconds, between two successive updates of the IO board AI and AO channels. Default value is 1 [ms].

EXAMPLES

Connects to an IO board

IODevice -connect

Reads and returns the value of the AI channel number 3

IODevice -read 3

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ectk(1ectk), comedi(7)

NAME

IOStat – displays statistics about IO operations done

SYNOPSIS

IOStat

DESCRIPTION

Displays statistics about IO operations done since the experiment was started (i.e. since the **start** was invoked). The returned statistics are:

- number of IO operations done so far

- mean time interval between two IO operations

- standard deviation of the time intervals between two IO operations

- minimal and maximal time interval between two IO operations.

The time interval between two IO operations is controlled by the variable **::ectk::IOboardUpdateInterval**.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ectk(1ectk), IOdevice (3ectk)

NAME

ca – defines a chronoamperommetry experiment in the ECTk process loop

SYNOPSIS

ca ?*option*? ?*arg*?

DESCRIPTION

The **ca** command allows to define a chronoamperommetry experiment and upload it to the ECTk process loop. **ca** is a ECTk procedure and implements all common features from ECTk procedures. After creation of the process, the command defines a new Tcl command **ca#** where **#** is the process ID associated by ECTk during the creation of the step. The **ps** command allows to see the command created. This command allows to manipulate the created ECTk procedure. The same options are valid for the **ca** and **ca#** command.

The **ca** command returns the process ID.

The following *option*s are defined besides the common options of all ECTk procedures:

-stepTime ?*t*?
: sets the step time to *t* sec. The step time is the number of seconds, after start of the chronoamperommetry experiment, when the step takes place. If *t* is not provided returns the current step time.

-stepDuration ?*d*?
: sets the step duration to *d* sec. If *d* is not provided returns the current step duration.

-eInitial ?*E*?
: sets the start potential to *E* Volts. If *E* is not provided returns the current start potential.

-eFinal ?*E*?
: sets the potential of the step to *E* Volts. If *E* is not provided returns the current step potential.

EXAMPLES

Defines a chronoamperommetry experiment starting at 0.2V and doing a step to 0.75V after 10sec during 30sec:

```
ca -stepTime 10 -stepDuration 30 -eInitial 0.2 V -eFinal 0.75
```

Defines an electro-deposition experiment. The electrode is first polarized at 0.4V during 5min. The electro-deposition is stopped by the trigger once a total negative charge of 0.003C was deposited at 0.7V (the experiment is as well stopped after 100sec in case not enough charge was deposited):

```
trigger \
-triggerScript {if {$Q<-3e-3} {return STOP} else {return CONT}} \
-action {terminate $myca}

set myca [ca -stepTime 300 -stepDuration 100 \
-eInitial 0.4 V -eFinal 0.7]
```

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

procedure(3ectk), ps(3ectk)

NAME

clear – removes electrochemical processes in the ECTk process loop

SYNOPSIS

clear *?ProcessIDsList?*

DESCRIPTION

Removes electrochemical processes in the ECTk process loop specified in *ProcessIDsList*. *ProcessIDsList* is a Tcl list of IDs of the processes to be removed. If no *ProcessIDsList* is provided, all processes are removed. The IDs must be valid PIDs as displayed by the **ps** command.

The clear command is implemented in Tcl and is part of the ECTklib. It loops over all entries of the *ProcessList* and calls the **remove** command with the listed IDs.

EXAMPLES

Removes all electrochemical processes currently loaded in the ECTk process loop:

clear

Removes the electrochemical processes with IDs 3,4 and 7

clear {3 4 7}

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ps(3ectk)

NAME

connect – connects to the IO board using a configuration file

SYNOPSIS

connect ?*option*? ?*configFile*?

DESCRIPTION

Connects to the IO board using the configuration file given by *configFile*.

The configuration file is a valid Tcl script which should define the specific configuration variables for the potensiostat to be driven. Note that ECTk will first execute the Tcl script in *configFile* and then connect to the board.

Displays an error message in case the board cannot be connected.

If **connect** is called with the option **-info** displays the current configuration of the connected board.

IMPLEMENTATION

The **connect** command is implemented in Tcl/Tk and is part of the ECTklib. It calls the ECtk command **IODevice -connect** *configFile*.

CONFIGURATION FILE FORMAT

The configuration file can be any valid Tcl script which should define the specific configuration variables for the potensiostat to be driven.

Typically the channel number configuration and their gains should be defined for a specific instrument.

For example the following Tcl script defines the configuration for a specific potensiostat:

```
puts "+-----------------------------------+"
puts "| Configuration for my potensiostat |"
puts "+-----------------------------------+"

# comedi device file associated with the IO board
set ::ectk::comediFile /dev/comedi0

# configuration of the IO channels
set ::ectk::chanIfo 1
set ::ectk::chanVfo 0
set ::ectk::chanCtrl 0
set ::ectk::AI_gain($::ectk::chanIfo) -1.0
set ::ectk::AI_gain($::ectk::chanVfo) -1.0

connect -info
```

The script ends by calling **connect -info** in order to display the configuration.

If this script is saved in mypotensiostat.tcl, then the IO board can be connected using

connect mypotensiostat.tcl

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

IODevice(3ectk)

NAME

cons – sends text to the ECTk console

SYNOPSIS

cons ?*string*?

DESCRIPTION

Sends characters given by *string* to the ECTk console. Note that, independently if the ECTk console is visible or not, data sent via the **cons** command are recorded in the ECTk console.

The command **cons** behaves as the Tcl **puts** command, i.e. it allows the utilization of delayed evaluations.

IMPLEMENTATION

The **cons** command is implemented in Tcl/Tk and is part of the ECTklib. It communicates with the **.console** widget, which is a **.text** Tk widget.

EXAMPLES

Sends a simple text to the ECTk console:

cons "This text is sent to the ECTk console"

Sends the list of all processes currently in the ECTk process loop to the ECTk console:

cons [ps]

Sends the name of the current user of ECTk to the ECTk console:

cons "Current user is $::ectk::user"

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

console(3ectk)

NAME

console – manipulates the ECTk console

SYNOPSIS

console ?*option*?

DESCRIPTION

Manipulates the ECTk console. The ECTk console displays messages that where generated with the **cons** command. If no *option* is provided, the ECTk console is made visible (same as **console -show**).

Note that, independently if the ECTk console is visible or not, data sent via the **cons** command are recorded in the ECTk console.

The valid *option*s are:

console -clear
clears the content of the ECTk console

console -hide
hides the ECTk console

console -show
shows the ECTk console

IMPLEMENTATION

The **console** command is implemented in Tcl/Tk and is part of the ECTklib. It defines the **.console** widget, which is a **.text** Tk widget.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

cons(3)

NAME

cv – defines a cyclic voltammetry experiment in the ECTk process loop

SYNOPSIS

cv ?*option*? ?*arg*?

DESCRIPTION

The **cv** command allows to define a cyclic voltammetry experiment and upload it to the ECTk process loop. **cv** is a ECTk procedure and implements all common features from ECTk procedures. After creation of the process, the command defines a new Tcl command **cv#** where **#** is the process ID associated by ECTk during the creation of the cyclic voltammetry experiment. The **ps** command allows to see the command created. This command allows to manipulate the created ECTk procedure. The same options are valid for the **cv** and **cv#** command.

The **cv** command returns the process ID.

The following *option*s are defined besides the common options of all ECTk procedures:

-scanRate ?*s*?
sets the scan rate to *s* Volts/sec. If *s* is not provided returns the current scan rate.

-eInitial ?*E*?
sets the start potential to *E* Volts. If *E* is not provided returns the current start potential.

-eFinal ?*E*?
sets the end potential to *E* Volts. If *E* is not provided returns the current end potential.

-nCycles ?*n*?
sets the number of cycles to *n*. If *n* is not provided returns the current number of cycles.

EXAMPLE

Defines a cyclic voltammetry experiment with 10 scans from -0.5 to 1.1 Volts at 100 mV/sec:

cv -eInitial -0.5 **-eFinal** 1.1 **-scanRate** 0.10 **-nCycles** 10

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

procedure(3ectk), ps(3ectk)

NAME

log – Prints the info of all electrochemical processes in the ECTk loop

SYNOPSIS

log *?OutChannelID?*

DESCRIPTION

Prints the info of all electrochemical processes currently loaded in the ECTk process loop to *OutChannelID*, as listed by the **ps** command. *OutChannelId* must be an identifier for an open channel such as a Tcl standard channel (**stdout** or **stderr**), the return value from an invocation of **open** or **socket**. The channel must have been opened for output.

If no *channelId* is specified then it defaults to **stdout**.

The log command is implemented in Tcl and is part of the ECTklib. It loops over all entries of the global array **::ectk::ProcList** and calls the listed commands with the **-info** option.

EXAMPLES

Writes the info of all electrochemical processes currently loaded:

```
log
```

Writes the info of all electrochemical processes currently loaded to the file "log.txt":

```
set chan [open "log.txt" w]
log $chan
close $chan
```

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ps(3ectk), remove(3ectk)

NAME

lrec – lists the ec records loaded in the memory of ectk

SYNOPSIS

lrec

DESCRIPTION

Returns a table with information about all ec records in the memory of ectk:

ECRECORD	the name of the ec record.
PID	the ec record ID.
OF RECORDS	the number of records within the ec record.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

who(3ectk)

NAME

monitor – opens a window displaying the potential and current

SYNOPSIS

monitor ?*updateInterval*?

DESCRIPTION

Opens a window displaying the potential and current. The update rate of the displayed values is given by the global variable **::ectk::monitorUpdateInterval** which gives the minimal time in seconds between two updates. The default value at start of ECTk is 1.0 sec.

If the optional argument *updateInterval* is set, **monitor** assigns this value to **::ectk::monitorUpdateInterval**, which is then used as new update interval.

The **monitor** command is implemented in Tcl/Tk and is part of the ECTklib.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ectk(1)

NAME

notes – manipulates notes to be added to an experiment

SYNOPSIS

notes ?*option*? ?*arg*?

DESCRIPTION

The **notes** command allows to define and manipulate notes to be associated with an experiment. The command **expSave** uses these notes to store them to disk.

When invoked without *option*, the **note** command displays all notes preceded with the index number of each entry.

The valid *option*s are:

notes -add
adds a note

notes -clear
clears all notes

notes -insert *index*
insert a note before the line *index*

notes -print
return all notes

notes -remove *index*
remove the note on line *index*

IMPLEMENTATION

The **notes** command is implemented in Tcl and is part of the ECTklib. It defines the list **:ectk::notes** to store the various entries.

The **notes** command is implemented in such way that it is possible to use variables inside notes. Before returning the results (with **notes** or **notes -print**), the variables are substituted using the **subst** command.

EXAMPLES

Adds a simple note:

```
notes -add {Working electrode = gold rod}
```

Defines a note entry using variables:

```
set d 1.5
set h 2.0
notes -add {Working electrode:}
notes -add { - Nickel wire}
notes -add { - Diameter = $::d mm}
notes -add { - Height   = $::h mm}
notes -add { - Surface  = [expr {$::h*3.14*$::d}] mm^2}
```

Note how the variables have to be accessed via their full path when used in a **notes** entry. The last command illustrates as well how other Tcl commands, in this example **expr**, can be used to generate more complex entries. Whenever a command such as **notes -print** is invoked, the current value of the variables **d** and **h** will be used to compute the surface.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

expSave(3ectk)

NAME

ps – returns the list of all processes currently loaded in the ECTk process loop

SYNOPSIS

ps

DESCRIPTION

Returns a table with the list of all processes currently loaded in the ECTk process loop. The table lists the following information about the processes:

OWNER — the process owner. Processes needed for the execution of ECTk are listed as system processes.

PID — the process ID.

PROCESS NAME — the name of the process.

CMD — the command that can be used to manipulate the process.

STATUS — the process status; either Waiting, Active or Terminated.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

who(3ectk)

NAME

pulse – defines a pulse train in the ECTk process loop

SYNOPSIS

pulse ?*option*? ?*arg*?

DESCRIPTION

The **pulse** command allows to define a pulse train and upload it to the ECTk process loop. **pulse** is a ECTk procedure and implements all common features from ECTk procedures. After creation of the process, the command defines a new Tcl command **pulse#** where **#** is the process ID associated by ECTk during the creation of the cyclic voltammetry experiment. The **ps** command allows to see the command created. This command allows to manipulate the created ECTk procedure. The same options are valid for the **pulse** and **pulse#** command.

The **pulse** command returns the process ID.

The following *option*s are defined besides the common options of all ECTk procedures:

-pulseHigh ?*h*?
: sets the high value of the pulse to *h* Volts. If *h* is not provided returns the current pulse high value.

-pulseLow ?*l*?
: sets the low value of the pulse to *l* Volts. If *l* is not provided returns the current pulse low value.

-Ton ?*t*?
: sets the pulse on-time to *t* msec. If *t* is not provided returns the current pulse-on time.

-Toff ?*t*?
: sets the pulse off-time to *t* msec. If *t* is not provided returns the current pulse-off time.

-duration ?*d*?
: sets the total duration of the pulse train to *t* sec. If *d* is not provided returns the current pulse train duration.

EXAMPLE

Defines a pulse train of 10 sec with pulses from -0.5 to 1.1 Volts of 20 msec duration and 80 msec pulse-off time:

pulse -pulseHigh 1.1 **-pulseLow** -0.5 \
-Ton 20 **-Toff** 80 **-duration** 10

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

procedure(3ectk), ps(3ectk)

NAME

save – saves the values acquired during an ECProcedure

SYNOPSIS

save *ECprocCmd fileName*

DESCRIPTION

Saves the values acquired during an ECProcedure. *ECProcCmd* is the Tcl command associated to the electrochemical procedure (which can be found with the **ps** command). *fileName* is the name of the file where the data have to be saved.

The save command is implemented in Tcl and is part of the ECTklib. It invokes the command **$ECProcCmd -save $fileName** and is therefore equivalent to the **-save** sub-command.

EXAMPLE

Save the data acquired during a cyclic voltammetry experiment in the file cvData.txt:

save cv3 cvData.txt

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ps(3ectk)

NAME

start – starts an ectk experiment

SYNOPSIS

start

DESCRIPTION

Starts an ectk experiment. It does this by activating the first ECProcess (which has to be a **cell -on** process) in the ectk process loop. Each process is then subsequently activated until the **cell -off** process. The order of execution of the processes is the order as displayed by the **ps** command.

Note that if no **cell -on** process was defined, this command has no effect. If no **cell -off** was defined, then experiment will not stop. The user can still stop the experiment using the **stop** command.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ectk(1ectk), stop(3ectk), ps(3ectk)

NAME

step – defines a step in the ECTk process loop

SYNOPSIS

step ?*option*? ?*arg*?

DESCRIPTION

The **step** command allows to define a step and upload it to the ECTk process loop. **step** is a ECTk procedure and implements all common features from ECTk procedures. After creation of the process, the command defines a new Tcl command **step#** where **#** is the process ID associated by ECTk during the creation of the step. The **ps** command allows to see the command created. This command allows to manipulate the created ECTk procedure. The same options are valid for the **step** and **step#** command.

The **step** command returns the process ID.

The following *option*s are defined besides the common options of all ECTk procedures:

-stepValue ?*s*?
sets the step hight *s* Volts. If *s* is not provided returns the current step height.

-stepDuration ?*d*?
sets the step duration to *d* sec. If *d* is not provided returns the current step duration.

EXAMPLES

Defines a step of 0.8 V with a duration of 35 sec:

step -stepValue 0.8 **-stepDuration** 35

Defines an electro-deposition experiment. The electro-deposition is stopped by the trigger once a total negative charge of 0.003 C was deposited (the experiment is as well stopped after 100 sec in case not enough charge was deposited):

```
trigger \
-triggerScript {if {$Q<-3e-3} {return STOP} else {return CONT}} \
-action {terminate $mystep}
set mystep [step -stepValue -0.7 -stepDuration 100]
```

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

procedure(3ectk), ps(3ectk)

NAME

stop − stops an ectk experiment

SYNOPSIS

stop

DESCRIPTION

Stops an ectk experiment. Any process currently executed is stopped. The last potential that was outputted is holden. All processes from the ectk process loop are set in the Terminated status.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ectk(1ectk), stop(3ectk)

NAME

terminate – terminate an electrochemical processes in the ECTk process loop

SYNOPSIS

terminate *ProcessID*

DESCRIPTION

Forces an electrochemical processes in the ECTk process loop to terminate. The *ProcessID* is the PID of the processe to be terminated as displayed by the **ps** command.

Once the process is terminated, it will execute the postProcess script and give control to the next following process in the ECTk loop. The process is not removed from the ECTk loop.

EXAMPLES

Terminate the electrochemical processe with PID 3

terminate 3

Usually the **terminate** command is used in triggers to stop a given process. In this example an electrodeposition is stopped by a trigger once a total negative charge of 0.003C was deposited (the experiment is as well stopped after 100s in case not enough charge was deposited):

**trigger **
-trigger {if {$Q<-3e-3**} {return STOP} else {return CONT}} **
-action {terminate $mystep**}**
set mystep **[step -stepValue** -0.7 **-stepDuration** 100**]**

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

process(1ectk), ps(3ectk), trigger(3ectk)

NAME

wait – waits during a defined time interval

SYNOPSIS

wait *N*

DESCRIPTION

Waits during *N* seconds. During this time interval, ECTk is sleeping and no other operations, besides waiting, are performed. In particular any value set on the AO channel of the connected IO board will be hold.

During the waiting interval it is not possible ot invoke any other command including the **stop** command.

IMPLEMENTATION

The **wait** command is implemented in Tcl and is part of the ECTklib.

EXAMPLES

Waits during 2.3 seconds

wait 2.3

Defines a cyclic voltammetry experiment with prepolarisation. The electrode is polarised during 5 sec at -0.5 Volts before starting the actual experiment:

cv -einitial -0.5 **-efinal** 1.2 **-scanrate** 0.10 \
-preProcess {**IOdevice -write** *::$ectk::chanCtrl* -0.5; **wait** 5}

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ECTk(1ectk)

NAME

who – displays the ECTk process loop and the list of ec records

SYNOPSIS

who

DESCRIPTION

Returns a table with the list of all processes currently loaded in the ECTk process loop and a table with all ec records in the memory of ectk.

The first table lists the following information about the processes:

OWNER	the process owner. Processes needed for the execution of ECTk are listed as system processes.
PID	the process ID.
PROCESS NAME	the name of the process.
CMD	the command that can be used to manipulate the process.
STATUS	the process status; either Waiting, Active or Terminated.

The second table lists the following information about the ec records currenlty loaded in the memory of ECTk:

ECRECORD	the name of the ec record.
PID	the ec record ID.
OF RECORDS	the number of records within the ec record.

IMPLEMENTATION

The **who** command is implemented in Tcl and is part of the ECTklib. It calls succesively the ectk commands **ps** and **lrec**.

COPYRIGHT

AUTHOR

Rolf Wuthrich <wuthrich@encs.concordia.ca>
Send bug reports or comments to the above address.

SEE ALSO

ps(3ectk), lrec(3ectk)