NAME

dde − Execute a Dynamic Data Exchange command

SYNOPSIS

package require dde 1.4

dde servername ?**−force**? ?**−handler** *proc*? ?**−** **−**? ?*topic*?

dde execute ?**−async**? ?**−binary**? *service topic data*

dde poke ?**−binary**? *service topic item data*

dde request ?**−binary**? *service topic item*

dde services *service topic*

dde eval ?**−async**? *topic cmd* ?*arg arg ...*?

DESCRIPTION

This command allows an application to send Dynamic Data Exchange (DDE) command when running under Microsoft Windows. Dynamic Data Exchange is a mechanism where applications can exchange raw data. Each DDE transaction needs a *service name* and a *topic*. Both the *service name* and *topic* are application defined; Tcl uses the service name **TclEval**, while the topic name is the name of the interpreter given by **dde servername**. Other applications have their own *service names* and *topics*. For instance, Microsoft Excel has the service name **Excel**.

DDE COMMANDS

The following commands are a subset of the full Dynamic Data Exchange set of commands.

dde servername ?**−force**? ?**−handler** *proc*? ?**−** **−**? ?*topic*?

dde servername registers the interpreter as a DDE server with the service name **TclEval** and the topic name specified by *topic*. If no*topic* is gi ven, **dde servername** returns the name of the current topic or the empty string if it is not registered as a service. If the given *topic* name is already in use, then a suffix of the form " #2" or " #3" is appended to the name to make it unique. The command's result will be the name actually used. The **−force** option is used to force registration of precisely the given *topic* name.

The **−handler** option specifies a Tcl procedure that will be called to process calls to the dde server. If the package has been loaded into a safe interpreter then a **−handler** procedure must be defined. The procedure is called with all the arguments provided by the remote call.

dde execute ?**−async**? ?**−binary**? *service topic data*

dde execute takes the *data* and sends it to the server indicated by *service* with the topic indicated by *topic*. Typically, *service* is the name of an application, and *topic* is a file to work on. The *data* field is given to the remote application. Typically, the application treats the *data* field as a script, and the script is run in the application. The **−async** option requests asynchronous invocation. The command returns an error message if the script did not run, unless the **−async** flag was used, in which case the command returns immediately with no error. Without the **−binary** option all data will be sent in unicode. For dde clients which don't implement the CF_UNICODE clipboard format, this will automatically be translated to the system encoding. You can use the **−binary** option in combination with the result of **encoding convertto** to send data in any other encoding.

dde poke ?**−binary**? *service topic item data*

dde poke passes the *data* to the server indicated by *service* using the *topic* and *item* specified. Typically, *service* is the name of an application. *topic* is application specific but can be a

command to the server or the name of a file to work on. The *item* is also application specific and is often not used, but it must always be non-null. The *data* field is given to the remote application. Without the **−binary** option all data will be sent in unicode. For dde clients which don't implement the CF_UNICODE clipboard format, this will automatically be translated to the system encoding. You can use the **−binary** option in combination with the result of **encoding convertto** to send data in any other encoding.

dde request ?**−binary**? *service topic item*

dde request is typically used to get the value of something; the value of a cell in Microsoft Excel or the text of a selection in Microsoft Word. *service* is typically the name of an application, *topic* is typically the name of the file, and *item* is application-specific. The command returns the value of *item* as defined in the application. Normally this is interpreted to be a string with terminating null. If **−binary** is specified, the result is returned as a byte array.

dde services *service topic*

dde services returns a list of service-topic pairs that currently exist on the machine. If *service* and *topic* are both empty strings ({}), then all service-topic pairs currently available on the system are returned. If *service* is empty and *topic* is not, then all services with the specified topic are returned. If *service* is non-empty and *topic* is, all topics for a given service are returned. If both are non-empty, if that service-topic pair currently exists, it is returned; otherwise, an empty string is returned.

dde eval ?**−async**? *topic cmd* ?*arg arg ...*?

dde eval evaluates a command and its arguments using the interpreter specified by *topic*. The DDE service must be the **TclEval** service. The **−async** option requests asynchronous invocation. The command returns an error message if the script did not run, unless the **−async** flag was used, in which case the command returns immediately with no error. This command can be used to replace send on Windows.

DDE AND TCL

A Tcl interpreter always has a service name of **TclEval**. Each different interpreter of all running Tcl applications must be given a unique name specified by **dde servername**. Each interp is available as a DDE topic only if the **dde servername** command was used to set the name of the topic for each interp. So a **dde services TclEval {}** command will return a list of service-topic pairs, where each of the currently running interps will be a topic.

When Tcl processes a **dde execute** command, the data for the execute is run as a script in the interp named by the topic of the **dde execute** command.

When Tcl processes a **dde request** command, it returns the value of the variable given in the dde command in the context of the interp named by the dde topic. Tcl reserves the variable **$TCLEVAL$EXECUTE$RESULT** for internal use, and **dde request** commands for that variable will give unpredictable results.

An external application which wishes to run a script in Tcl should have that script store its result in a variable, run the **dde execute** command, and then run **dde request** to get the value of the variable.

When using DDE, be careful to ensure that the event queue is flushed using either **update** or **vwait**. This happens by default when using **wish** unless a blocking command is called (such as **exec** without adding the **&** to place the process in the background). If for any reason the event queue is not flushed, DDE commands may hang until the event queue is flushed. This can create a deadlock situation.

EXAMPLE

This asks Internet Explorer (which must already be running) to go to a particularly important website:

```
package require dde
dde execute -async iexplore WWW_OpenURL http://www.tcl-lang.org/
```

SEE ALSO

tk(n), winfo(n), send(n)

KEYWORDS

application, dde, name, remote execution