NAME

Blt_TreeCreate − Create tree data object.

SYNOPSIS

#include <bltTree.h>

int
Blt_TreeCreate(*interp*, *name*, *tokenPtr*)

ARGUMENTS

Tcl_Interp	**interp*	(in)	Interpreter to report results back to.
const char	**name*	(in)	Name of the new tree. Can be qualified by a namespace.
Blt_Tree	**tokenPtr*	(out)	If not NULL, points to location to store the client tree token.

DESCRIPTION

This procedure creates a C-based tree data object and optionally returns a token to it. The arguments are as follows:

interp Interpreter to report results back to. If an error occurs, then interp->result will contain an error message.

name Name of the new tree object. You can think of *name* as the memory address of the object. It's a unique name that identifies the tree object. No tree object *name* can already exist. *Name* can be qualified by a namespace such as **fred::myTree**. If no namespace qualifier is used, the tree will be created in the current namespace, not the global namespace. If a qualifier is present, the namespace must already exist.

tokenPtr Holds the returned token. *TokenPtr* points to a location where it is stored. Tree tokens are used to work with the tree object. If NULL, no token is allocated. You can later use **Tcl_TreeGetToken** to obtain a token.

The new tree data object created will initially contain only a root node. You can add new nodes with **Blt_TreeCreateNode**.

Optionally a token for the tree data object is returned. Tree data objects can be shared. For example, the **tree** and **hiertable** commands may be accessing the same tree data object. Each client grabs a token that is associated with the tree. When all tokens are released (see **Blt_TreeReleaseToken**) the tree data object is automatically destroyed.

RETURNS

A standard Tcl result is returned. If TCL_ERROR is returned, then *interp->result* will contain an error message. The following errors may occur:

- There already exists a tree by the same name as *name*. You can use **Tcl_TreeExists** to determine if a tree exists beforehand.
- The tree name is prefixed by a namespace that doesn't exist. If you qualified the tree name with a namespace, the namespace must exist. Unlike Tcl procs and variables, the namespace is not automatically created for you.
- Memory can't be allocated for the tree or token.

EXAMPLE

The following example creates a new

```
Blt_Tree token;

if (Blt_TreeCreate(interp, "myTree", &token) != TCL_OK) {
    return TCL_ERROR;
```

```
        }
        printf("tree is %s\n", Blt_TreeName(token));
```

KEYWORDS

Tcl_TreeGetToken, Tcl_TreeExists, Tcl_TreeReleaseToken