ClassName

Inherits From:	Superclass : Object
Conforms To:	protocols... (remove this line if no protocols are supported)
Declared In:	include directory

Class Description

The description of the class goes here.

Instance Variables

```
int         tag;
id          target;
SEL         action;
```

tag	An integer used to identify the ActionCell.
target	The object that is sent the ActionCell's action.
action	The message that the ActionCell sends to its target.

Adopted Protocols (remove if no protocols supported)

Protocol name	±€methods implemented ±€etc...

Method Types

Initializing the class	+€initialize +€alloc +€allocFromZone:
Creating and freeing instances	±€init ±€free

Class Methods

alloc
+€**alloc**

This method cannot be used to create an Application object. Use **new** instead. The method is implemented only to prevent you from using it; if you do use it, it generates an error message.

See also: +€**new**

Instance Methods

loadNibSection:owner:withNames:fromZone:
±€**loadNibSection:**(const char *)*name*
owner:*anOwner*
withNames:(BOOL)*flag*
fromZone:(NXZone *)*zone*

Loads interface objects and their names from the source identified by *name*. The source may be a section within the executable file, or a file within the application bundle, as described above for ± **loadNibSection:owner:**.

The argument *anOwner* is the object that corresponds to the ªFile's Ownerº object in Interface Builder's File window. When *flag* is YES, the objects' names are also loaded. Names *must* be loaded if you use **NXGetNamedObject()** to get at the objects, but are not otherwise required. Memory for the loaded objects is allocated from the zone specified by *zone*.

Returns non-**nil** if the section or file is successfully opened and read, and **nil** otherwise.

See also: ±€**loadNibSection:owner:withNames:fromHeader:fromZone:**

setAction:
    -€setAction:(SEL)*aSelector*

Sets the ActionCell's action method to *aSelector.* The argument of an action method sent by an ActionCell is its associated Control (the object returned by **controlView**). Returns **self**.

See also: **-€action**, **-€setTarget:**, **-€controlView**, **-€sendAction:to:** (Control)

Methods Implemented by the Delegate

app:openFile:type:
    ±€(int)**app:***sender*
        openFile:(const char *)*filename*
        type:(const char *)*aType*

Invoked from within **openFile:ok:** after it has been determined that the application can open another file. The method should attempt to open the file *filename* with the extension *aType*, returning YES if the file is successfully opened, and NO otherwise.

This method is also invoked from within **openTempFile:ok:** if neither the delegate nor the Application subclass responds to **app:openTempFile:type:**

See also: ±€**openFile:ok:**, ±€**openTempFile:ok:**

Constants and Defined Types

```
/* KITDEFINED subtypes */
#define NX_WINEXPOSED    0
#define NX_APPACT        1
#define NX_APPDEACT      2
#define NX_WINMOVED      4
#define NX_SCREENCHANGED 8

/*
 * The NXModalSession structure contains information used by the
 * system between beginModalSession:for: and endModalSession:
 * messages.  This structure can either be allocated on the stack
 * frame of the caller, or by beginModalSession:for:.  The
 * application should not access any of the elements of this
 * structure.
 */

typedef struct _NXModalSession {
    id app;
    id window;
    struct _NXModalSession *prevSession;
    int oldRunningCount;
    BOOL oldDoesHide;
    BOOL freeMe;
    int winNum;
    NXHandler *errorData;
    int reserved1;
    int reserved2;
} NXModalSession;
```