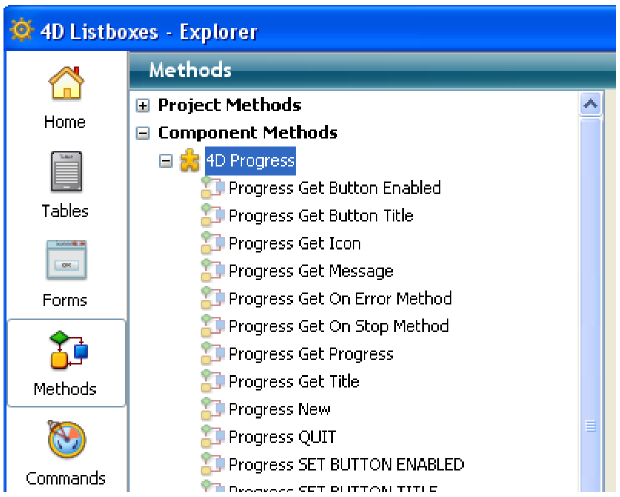
4D Progress

4D includes a built-in component named 4D Progress. This component lets you open one or more progress bars in the same window (just like the Finder on Mac OS).

Each progress bar is assigned an ID that is generated automatically by the **Progress New** method. This ID is used by all the project methods of the component to manage the properties and values in progress dialog boxes.

The project methods for this component are listed on the **Component Methods** page of the Explorer:



Progress bars
Alphabetical list of commands

Progress bars

Progress Get Button Enabled

Progress Get Button Enabled (id) -> Function result

Parameter	Type		Description
id	Longint	⇒	ID of progress bar
Function result	Boolean	↩	True = Stop button displayed; otherwise, False

Description

The **Progress Get Button Enabled** method returns **True** when the progress bar designated by the *id* parameter displays a **Stop** button. If it does not display one (default operation), the method returns **False**.

Progress Get Button Title

Progress Get Button Title (id) -> Function result

Parameter	Type		Description
id	Longint	→	ID of progress bar
Function result	Text	↩	Title of Stop button

Description

Note: This method can only be used under Windows. Under Mac OS, stop buttons do not have titles.

The **Progress Get Button Title** method returns the current title of the stop button for the progress bar designated by the *id* parameter.

By default, the title is "Stop". Note that the method returns the current title even when the **Stop** button is not displayed.

Progress Get Icon

Progress Get Icon (id) -> Function result

Parameter	Type		Description
id	Longint	⇒	ID of progress bar
Function result	Picture	⮌	Progress bar icon

Description

The **Progress Get Icon** method returns the icon of the progress bar designated by the *id* parameter.

Progress Get Message

Progress Get Message (id) -> Function result

Parameter	Type		Description
id	Longint	→	ID of progress bar
Function result	Text	↩	Progress bar message

Description

The **Progress Get Message** method returns the message of the progress bar designated by the *id* parameter.

Progress Get On Error Method

Progress Get On Error Method -> Function result

Parameter	Type		Description
Function result	Text	⮐	Name of method called in case of error (if specified)

Description

The **Progress Get On Error Method** method returns the name of the project method of the host database that is called in the event of an error when using a progress bar.

If no error method is specified, this method returns an empty string.

Progress Get On Stop Method

Progress Get On Stop Method (id) -> Function result

Parameter	Type		Description
id	Longint	⇒	ID of progress bar
Function result	Text	↩	Name of method called when Stop button is clicked (if specified)

Description

The **Progress Get On Stop Method** method returns the name of the project method of the host database that is called when a user clicks on the **Stop** button of the progress bar designated by the *id* parameter.

If no method is associated with the **Stop** button, this method returns an empty string.

Progress Get Progress

Progress Get Progress (id) -> Function result

Parameter	Type		Description
id	Longint	→	ID of progress bar
Function result	Real	↩	Progress bar value

Description

The **Progress Get Progress** method returns the current value associated with the progress bar designated by the *id* parameter.

Progress Get Title

Progress Get Title (id) -> Function result

Parameter	Type		Description
id	Longint	→	ID of progress bar
Function result	Text	↩	Progress bar title

Description

The **Progress Get Title** method returns the main title of the progress bar designated by the *id* parameter.

Progress New

Progress New -> Function result

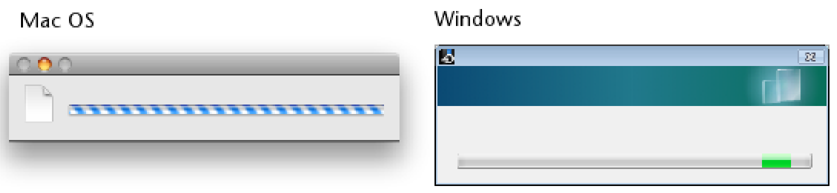
Parameter	Type		Description
Function result	Longint	↩	ID of new progress bar

Description

The **Progress New** method creates a new progress bar and returns its ID number. This number is unique throughout the duration of the progress bar but can be reused subsequently.

The first time this method is called, a local process is created and a new centered window is opened above the main window. By default, this window:

- contains an undefined progress bar
- does not have a message.



Progress QUIT

Progress QUIT (id)

Parameter	Type		Description
id	Longint	→	ID of progress bar

Description

The **Progress QUIT** method closes the progress bar referenced by the *id* parameter.

If *id* designates the only progress bar displayed, the progress window is also closed (as well as the local process). Otherwise, the window is resized so that it only contains the progress bars that are still open.

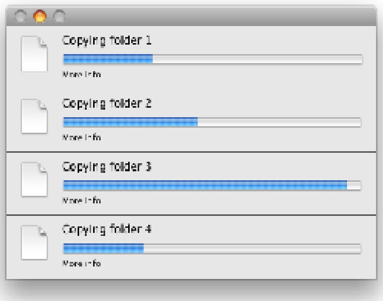
You can pass 0 in the *id* parameter in order to stop all the progress bars and close the progress window.

Example

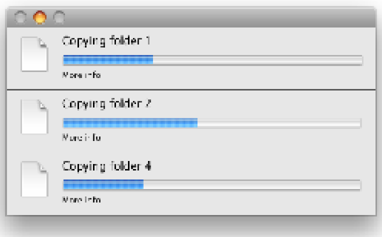
If the bar named "Copying folder 3" has the ID number 3:

```
Progress QUIT(3)
```

Before

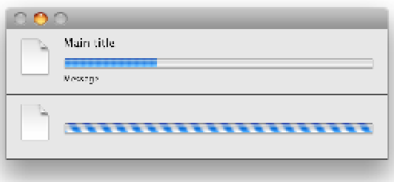


After

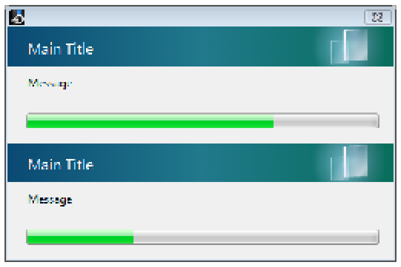


If a progress window is already open when this method is called, this window is resized so that it shows a new progress bar under the previous one(s) in the same process:

Mac OS



Windows



Progress SET BUTTON ENABLED

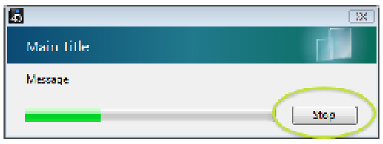
Progress SET BUTTON ENABLED (id ; button)

Parameter	Type		Description
id	Longint	⇨	ID of progress bar
button	Boolean	⇨	True = Show, False = Hide

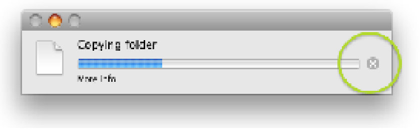
Description

The **Progress SET BUTTON ENABLED** method adds a **Stop** button to the progress bar designated by the *id* parameter.

By default, progress bars do not have **Stop** buttons. If you pass **True** in the *button* parameter, then a button is displayed:



Windows



Mac OS

You must manage the effect of clicking on this **Stop** button using the **Progress SET ON STOP METHOD** method or by testing the value of the **Progress Stopped** method.

Progress SET BUTTON TITLE

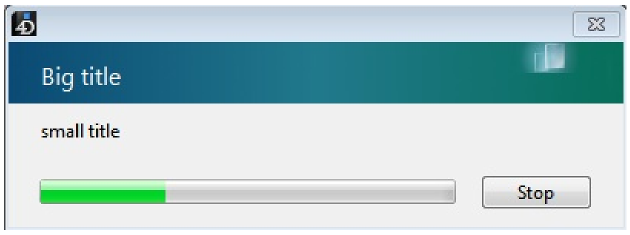
Progress SET BUTTON TITLE (id ; title)

Parameter	Type		Description
id	Longint	→	ID of progress bar
title	Text	→	Title of Stop button (Windows)

Description

Note: This method can only be used under Windows. Under Mac OS, stop buttons do not have titles.

The **Progress SET BUTTON TITLE** method changes the title of the **Stop** button for the progress bar designated by the *id* parameter. By default, the title of this button is "Stop":

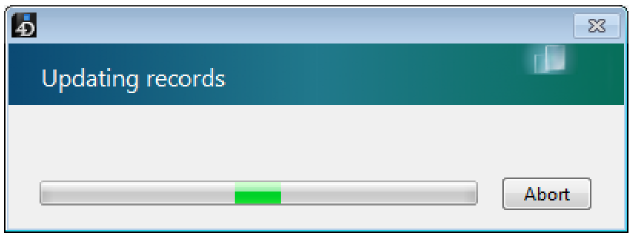


By default, progress bars do not have **Stop** buttons. If you want a progress bar to have one, you must use the **Progress SET BUTTON ENABLED** method.

Example

You want to use the title "Abort":

```
<>ID:=Progress New
...
Progress SET BUTTON TITLE(<>ID;"Abort")
```



Progress SET FONT SIZES

Progress SET FONT SIZES (sizeTitles {; sizeMessages {; sizeButtons}})

Parameter	Type		Description
sizeTitles	Longint	⇨	Size of font for titles
sizeMessages	Longint	⇨	Size of font for messages
sizeButtons	Longint	⇨	(Windows) Size of font for Stop buttons

Description

The **Progress SET FONT SIZES** method changes the font size used for the different text displayed in all the progress windows.

In the *sizeTitles*, *sizeMessages* and *sizeButtons* parameters, pass the font sizes to use. If you do not want to modify a size, pass -1 in the corresponding parameter.

Example 1

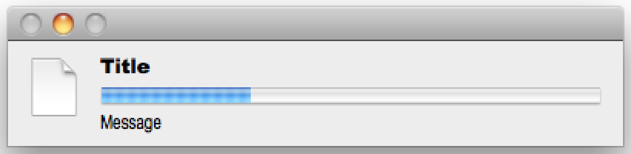
We want to change the size only for messages:

```
Progress SET FONT SIZES(-1;13)
```

Example 2

We want to change the size of titles and messages:

```
Progress SET FONTS("Arial Black";"Arial narrow")
Progress SET FONT SIZES(13;12)
```



Progress SET FONTS

Progress SET FONTS (fontTitles {; fontMessages {; fontButtons}})

Parameter	Type		Description
fontTitles	Text	⇒	Font to use for titles
fontMessages	Text	⇒	Font to use for messages
fontButtons	Text	⇒	(Windows) Font to use for Stop buttons

Description

The **Progress SET FONTS** method changes the fonts used for the different text displayed in all the progress windows.

In the *fontTitles*, *fontMessages* and *fontButtons* parameters, pass the names of the fonts to use. If you do not want to modify a font, pass an empty string ("") in the corresponding parameter.

Example

We want to change the font only for messages:

```
Progress SET FONTS("";"Arial")
```

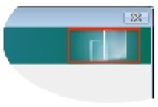
Progress SET ICON

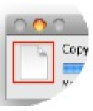
Progress SET ICON (id ; icon {; foreground})

Parameter	Type		Description
id	Longint	→	ID of progress bar
icon	Picture	→	Picture to display as icon
foreground	Boolean	→	Show progress bar in foreground

Description

The **Progress SET ICON** method modifies the icon displayed in the progress bar. By default, the following icons are displayed:


Windows



Mac OS

In *id*, you pass the unique ID of the progress bar, returned by the **Progress New** method.

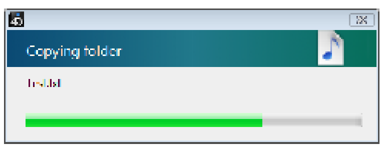
In *icon*, you pass the picture (variable or field) to use as the icon in the progress bar window. The maximum size of this icon must be:

- under Mac OS, 40 x 40 pixels
- under Windows, 40 x 80 pixels

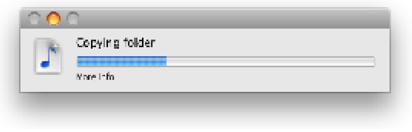
If you pass an icon with a size that is smaller than these limits, it is centered but not resized. However, when its size exceeds these limits, it is both centered and resized.

Pass **True** in *foreground* when you want to force the progress window to the foreground of the application.

Examples of custom icons:



Windows



Mac OS

Progress SET MESSAGE

Progress SET MESSAGE (id ; message {; foreground})

Parameter	Type		Description
id	Longint	⇨	ID of progress bar
message	Text	⇨	Message of progress bar
foreground	Boolean	⇨	Show progress bar in foreground

Description

The **Progress SET MESSAGE** method changes the message shown in the progress bar.

In *id*, you pass the unique ID of the progress bar, returned by the **Progress New** method.

In *message*, you pass the text you want to modify under the main title (Windows) or under the progress bar (Mac OS).

Pass **True** in *foreground* when you want to force the progress window to the foreground of the application.

Progress SET ON ERROR METHOD

Progress SET ON ERROR METHOD (methodName)

Parameter	Type		Description
methodName	Text	⇒	Name of error method

Description

The **Progress SET ON ERROR METHOD** method designates a method to execute in the event of an error when using progress bars (for example id unknown, incorrect number of parameters, etc.).

In *methodName*, you pass the name of the project method of the host database to be called in the event of an error. This method is shared by all the progress windows of the application.

Note: Since the **Progress SET ON ERROR METHOD** method belongs to a component, you must remember to assign the "Shared by components and host database" property to the *methodName* method; otherwise an error is returned.

The *methodName* method is called with three parameters:

- $1 (Longint): error number
- $2 (Text): text of the error
- $3 (Longint): unique ID of progress bar

Example

Here is an example of a method called in the event of an error. This method is declared as "shared" in Design mode:

```
C_LONGINT($1)
C_TEXT($2)
C_LONGINT($3)

C_LONGINT($ErrorID)
C_TEXT($ErrorText)
C_LONGINT($ProgressID)

$ErrorID:=$1
$ErrorText:=$2
$ProgressID:=$3
$Error:=""
$Error:=$Error+"Error number: "+String($ErrorID)+Char(Carriage return)
$Error:=$Error+$ErrorText+Char(Carriage return)
$Error:=$Error+"Progress ID: "+String($ProgressID)
ALERT($Error)
```

Progress SET ON STOP METHOD

Progress SET ON STOP METHOD (id ; methodName)

Parameter	Type		Description
id	Longint	⇨	ID of progress bar
methodName	Text	⇨	Name of stop method

Description

The **Progress SET ON STOP METHOD** method designates a method to execute when the user clicks on the **Stop** button of the progress bar.

By default, progress bars do not have **Stop** buttons. If you want a progress bar to have one, you must use the **Progress SET BUTTON ENABLED** method.

In *id*, you pass the unique ID of the progress bar, returned by the **Progress New** method.

In *methodName*, you pass the name of the project method of the host database to be called when the **Stop** button is clicked. This method receives the unique ID of the progress bar as its first parameter. Then it is executed in a new process launched by the component.

Note: Since the **Progress SET ON STOP METHOD** method belongs to a component, you must remember to assign the "Shared by components and host database" property to the *methodName* method; otherwise an error is returned.

Progress SET PROGRESS

Progress SET PROGRESS (id ; progress {; message {; foreground}})

Parameter	Type		Description
id	Longint	⇒	ID of progress bar
progress	Real	⇒	Value of progress ([0...1] or -1)
message	Text	⇒	Message of progress bar
foreground	Boolean	⇒	Show progress bar in foreground

Description

The **Progress SET PROGRESS** method modifies the value of the progress bar along with the information shown in the progress window. It is useful for updating a progress bar within a loop.

In *id*, you pass the unique ID of the progress bar, returned by the **Progress New** method.

In *progress*, you pass the current value of the progress bar. You can pass a Real value (between 0 and 1) or -1 to specify an undefined progress bar (also known as a "Barber shop" bar under Mac OS).

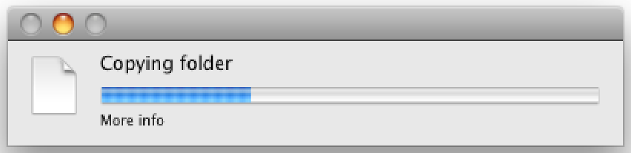
In *message*, you pass additional text to display under the main title (Windows) or under the progress bar (Mac OS). This parameter is optional.

Pass **True** in *foreground* when you want to force the progress window to the foreground of the application.

Example

Updating of progress bar:

```
$P:=Progress New // we create a new bar
 // Carry out processing in a loop
For($i;1;100)
 // ... code of processing
 // Update progress bar
   $r:=$i/100
   Progress SET PROGRESS($P;$r;"More info")
End for
 // Deletion of bar once processing is over
PROGRESS QUIT($P)
```


Copying folder

More info

Progress SET TITLE

Progress SET TITLE (id ; title {; progress {; message {; foreground}}})

Parameter	Type		Description
id	Longint	⇒	ID of progress bar
title	Text	⇒	Title of progress bar
progress	Real	⇒	Value of progress ([0...1] or -1)
message	Text	⇒	Message of progress bar
foreground	Boolean	⇒	Show progress bar in foreground

Description

The **Progress SET TITLE** method sets the title of the progress bar along with the information displayed in the progress bar window.

In *id*, you pass the unique ID of the progress bar, returned by the **Progress New** method.

In *title*, you pass the main text to display in the progress bar window.

In *progress*, you pass the current value of the progress bar (optional). You can pass a Real value (between 0 and 1) or -1 to specify an undefined progress bar (also known as a "Barber shop" bar under Mac OS).

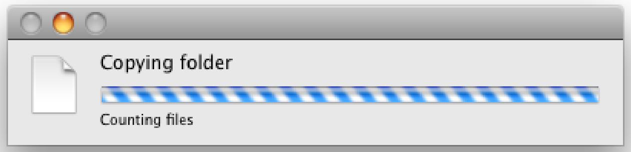
In *message*, you pass additional text to display under the main title (Windows) or under the progress bar (Mac OS). This parameter is optional.

Pass **True** in *foreground* when you want to force the progress window to the foreground of the application.

Example

Creating a simple progress bar window:

```
$P:=Progress New
Progress SET TITLE($P;"Copying folder";-1;"Counting files
```



Copying folder

Counting files

Progress SET WINDOW VISIBLE

Progress SET WINDOW VISIBLE (visible {; horPos ; vertPos {; foreground}})

Parameter	Type		Description
visible	Boolean	⇒	True = Show, False = Hide
horPos	Longint	⇒	Left coordinate of window -1 = No change
vertPos	Longint	⇒	Top coordinate of window -1 = No change
foreground	Boolean	⇒	Show progress bar in foreground

Description

The **Progress SET WINDOW VISIBLE** method manages the display of the progress bar window, if it exists.

The *visible* parameter indicates whether or not the window is visible (by default, it is visible). Pass **False** in this parameter to hide the window and **True** to show it again, if it exists.

The *horPos* and *vertPos* parameters modify the location of the progress bar window on screen. In these parameters, you pass values representing the movement in pixels to the right (*horPos*) or downwards (*vertPos*) with respect to the coordinates of the main application window (under Windows) or the screen (under Mac OS).

When you do not want to modify these coordinates (for instance, if you want to use the *foreground* parameter without changing the position of the window), pass -1 to each of these parameters.

Pass **True** in *foreground* when you want to force the progress window to the foreground of the application.

Example 1

Place the progress bar window 50 pixels from the left edge and 100 pixels from the top edge:

```
Progress SET WINDOW VISIBLE(True;50;100)
```

Example 2

Hide the progress bar window:

```
Progress SET WINDOW VISIBLE(False)
```

Example 3

Display the progress bar window and move it to the foreground without changing its position:

```
Progress SET WINDOW VISIBLE(True;-1;-1;True)
```

Progress Stopped

Progress Stopped (id) -> Function result

Parameter	Type		Description
id	Longint	➡	ID of progress bar
Function result	Boolean	↩	True = User clicked the Stop button

Description

The **Progress Stopped** method returns **True** when a user clicks on the **Stop** button of the progress bar designated by the *id* parameter.

You must call this method to test whether the user clicked the **Stop** button. The button does not trigger an event itself.

Example

Example of progress bar on a loop:

```
$ProgressID:=Progress New // create a new progress bar
 // The progress bar must have a Stop button
Progress SET BUTTON ENABLED($ProgressID;True)
For($i;1;100)
 // As long as progress is not stopped...
   If(Not(Progress Stopped($ProgressID)))
      Progress SET TITLE($ProgressID;"Test progress #"+String($ProgressID))
      Progress SET PROGRESS($ProgressID;$i/100)
      Progress SET MESSAGE($ProgressID;String(100*$i/100)+" %")
      (...)
   Else // The user clicks on Stop
      $i:=100 // We exit the loop
   End if
End for
 // Final closing of progress bar (the Stop button itself does nothing)
Progress QUIT($ProgressID)
```