View Security and Access Privileges

Individual access privileges to graphical objects in a View application can be implemented by adding users to the View target and assigning each user an appropriate integer value as access level (privilege level).
Every time a user logs into the application, the system variable #AccessLevel is loaded with the user's assigned access level value. Values may range from 0 to 999.
Graphical objects and application scripts can use the #AccessLevel variable to interlock functionality.

In addition to its password and assigned access level, each user account has a couple additional properties to govern certain general application options like

- Exit from the View application
- Changing data
- Open Inspector Windows like e.g. Variable Inspector
- Context Menus (Right Click)

(Also see Figure 4 further down in this document)

Step 1: Plan how many Access Levels (Privilege Levels) an Application requires.

Generally, it is advisable to consider access privileges already when starting a new application as this allows integrating security 'from the ground up' and may slightly impact overall application aspects, like screen architecture etc.
It is however possible to add this functionality to an application at any time.

In many applications two privilege levels, like an engineering administrator and an operator may suffice.
Example:
The administrator requires login with password and has access to all functions in the View application, machine operation and engineering functions like setup screens etc.
An operator has access to machine operation, but cannot access engineering functions.

Note: Since the value of #AccessLevel may range from 0 to 999, it is however possible to implement any more sophisticated distinction of operator-, or engineering access rights.

Step 2: Create User Accounts on the View Target and assign each User an appropriate Access Level Value.

By default, each View target has two user accounts already predefined. Both system accounts may be customized (in a limited way) and cannot be deleted.

1.) The Administrator User (master)

Username: master
Password: control

master is loaded with an access level value of 999 and has all application options enabled.

This account is equivalent an administrator that always has highest privilege level.
Properties of the master user cannot be changed, except password.

Note: It may be advisable to change the password to a non-default value.
Caution: When the master password is lost, there is no recovery option.

2.) The Standard User (fxDefault)

Username: fxDefault
Password: none assigned

fxDefault is loaded with an access level value of 0 and by default has all application options enabled.

fxDefault is comparable a standard user with low privilege level.
It is not possible to assign a password or a different access level value to the fxDefault user. Application options may however be changed.

Like mentioned in the previous section, the two predefined users may suffice for many applications. When a View application starts, fxDefault is logged in by default. After any other logged-in user like e.g. master logs out, the fxDefault user is active again.

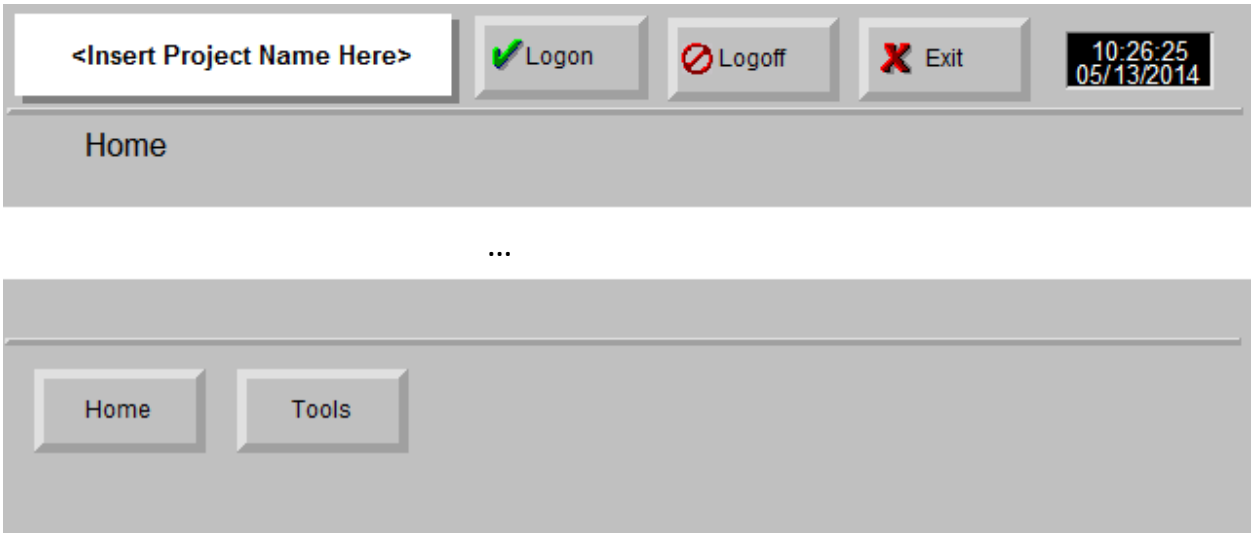
If an application wants to distinguish more than two major privilege levels, it needs to create
3.) Additional User Accounts.

- User accounts are created on the View target. I.e. this configuration is *not* part of the Machine Edition project.
- A View application needs to execute the EditUserList script function in order to create, modify or delete user accounts.
- To execute the EditUserList function, a user with an access level value of 999 must be logged in like e.g. master.

Each new View target in Machine Edition has a standard screen header, footer and tools screen predefined with buttons to log on, log off, to edit the user list and other administrative functions.

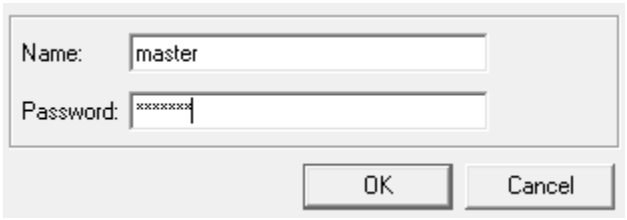
To work on user accounts it may be helpful to first run the unmodified Standard View application and use its predefined objects.

Figure 1 Initial screen of a new unmodified View target



Hit the Logon Button to open the Logon Dialog

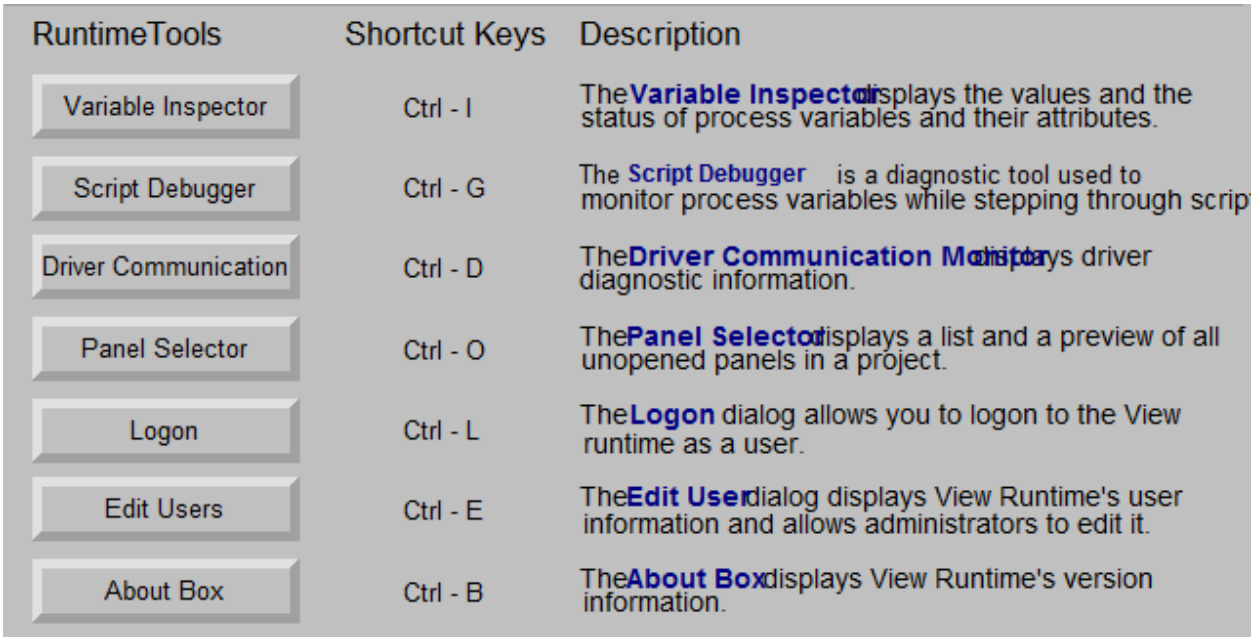
Figure 2 The Logon Dialog



Log on as User = master
Password = control

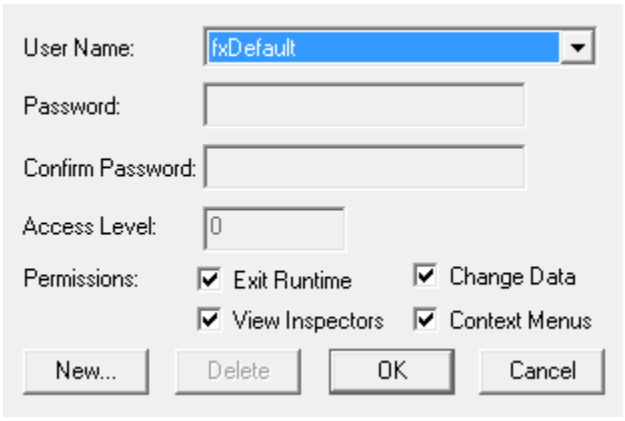
Hit the Tools Button to enter the Tools Screen

Figure 3 The predefined Tools Screen



Hit the Edit Users Button to open the Edit User List Dialog

Figure 4 The Edit User List Dialog



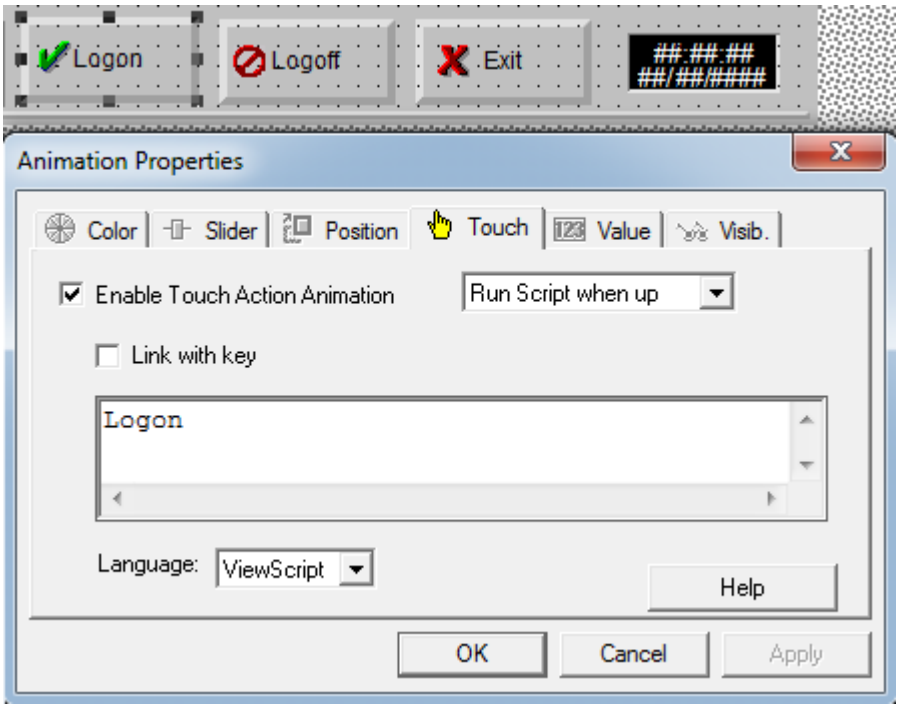
The Edit Users Button executes the EditUserList function. This function opens the dialog shown above and allows creating, deleting and modifying user accounts.

The above button and Tools Screen objects are only present when creating a new View project. If a View target is added to an existing project, the application needs to add appropriate elements on its own.

In addition, many applications may want to use their own architecture and look + feel of graphical objects.

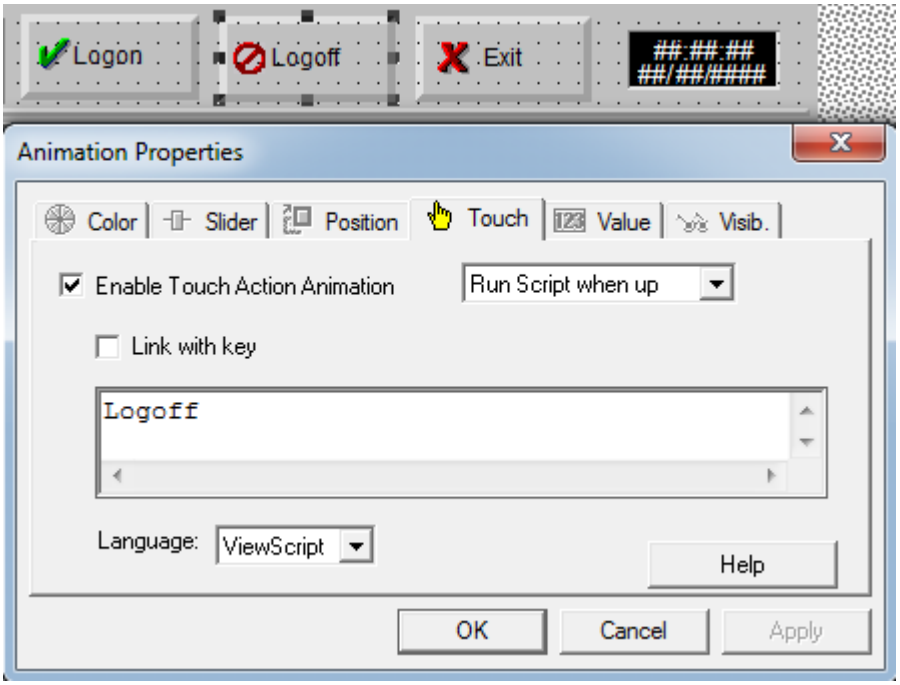
The following figures show animation properties of the previously used buttons. Applications can define their own buttons or other graphical objects with animation properties to log on or log off from the application, as well the capability to edit the user list.

Figure 5 How to implement a Logon function.



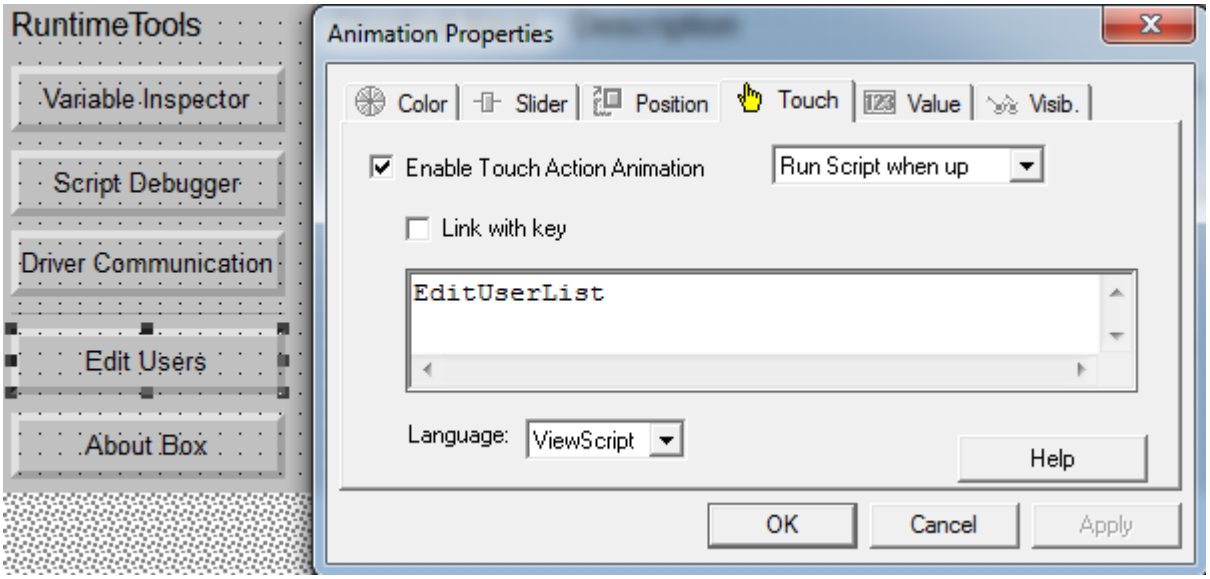
A Logon function should always be reachable for each user.

Figure 6 How to implement a Logoff function.



A function to log off is required and should be reachable for each user.

Figure 7 How to access the User List.

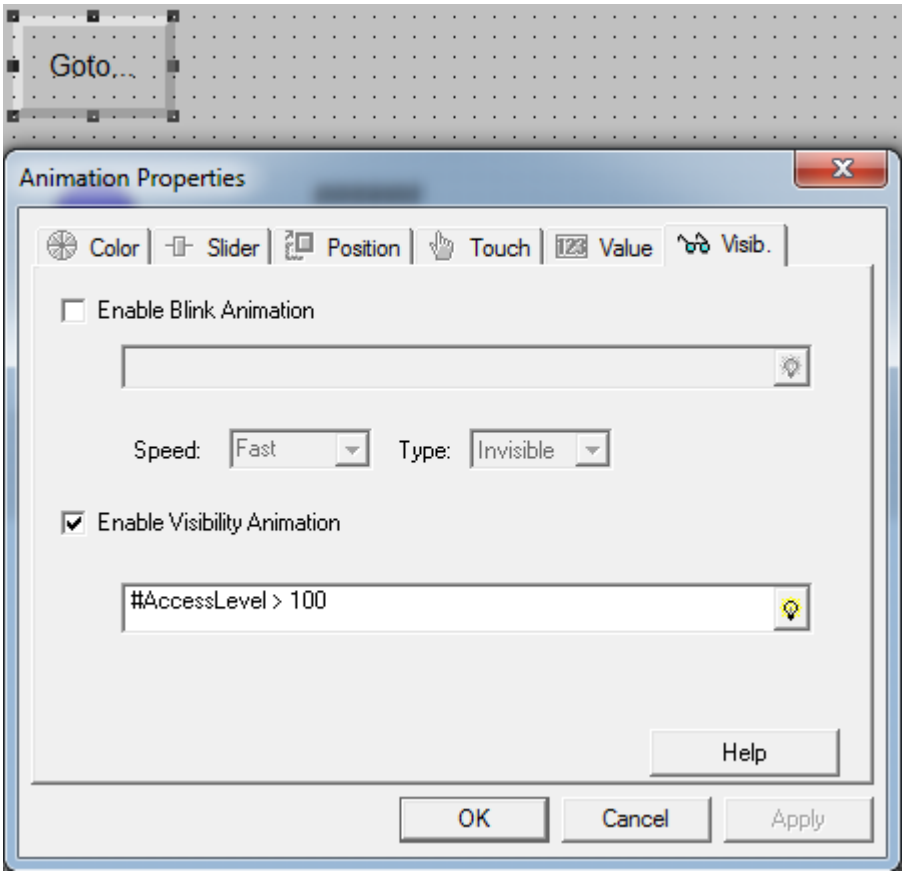


A function to edit the user list is required and only reachable for users with an access level value of 999.

Step 3: Use the #Access Level variable in View animated Objects

The most meaningful object animation types to use #AccessLevel are *Visibility, Touch and Color*.

1. How to animate Visibility of an Object



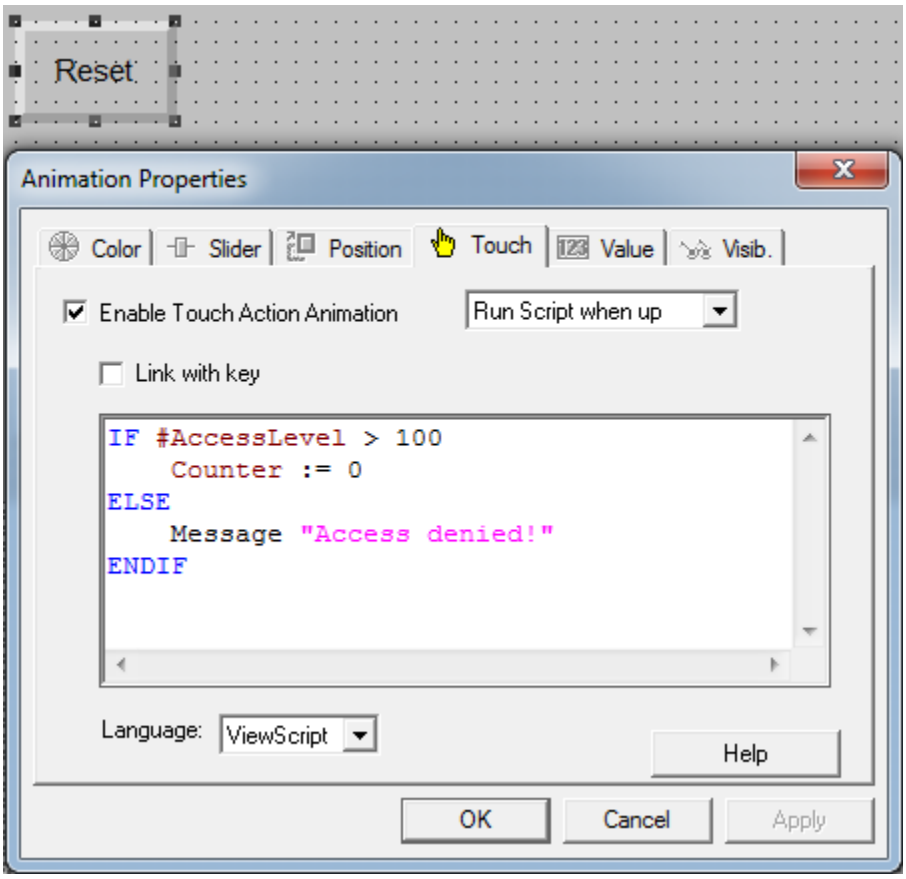
In this example the pushbutton is only visible for users with an access level value > 100. The visibility animation is useful to hide special setup actions, navigations or display values from users that would not have permission for the associated action or display.

If the touch animation of this button was e.g. the only option to navigate to an engineering setup screen, this screen would not be accessible for access levels below or equal 100.

Note: Graphic objects can be configured to more than one animation type as appropriate for the application, like e.g. Touch *and* Visibility.

Tip: As far as whole screens are concerned, an access level value may also be assigned in the 'Security' property of a View screen. Users with a lower access level value than assigned in this property cannot view the screen.

2. How to control Touch Actions of an Object

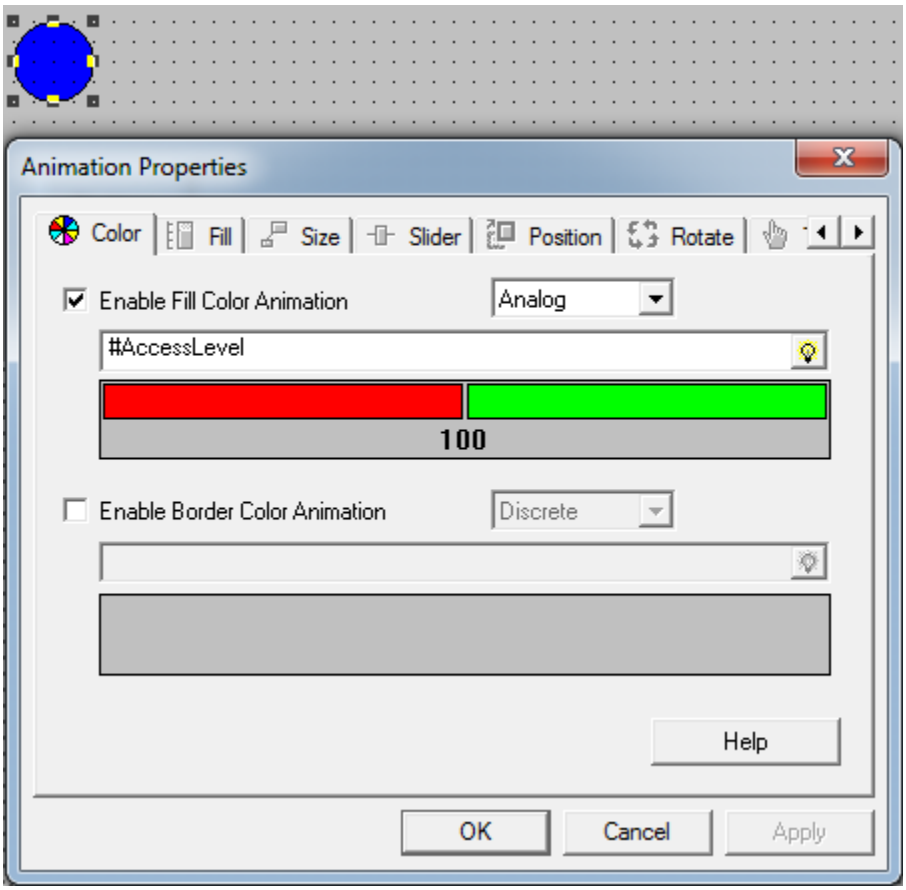


In this example the pushbuttons touch action executes a script (Touch Animation Script).

For users with an access level value > 100, the script writes a value of 0 into Counter. For users with an access level <= 100 it displays a message box indicating access is denied.
This type of animation is useful to simply prevent certain actions for users that would not have permission for the associated action.
The script could also differentiate valid actions, depending on the current user's access level value.

Tip: The #AccessLevel variable can also be used in application scripts to differentiate actions based on the current user's access level.

3. How to animate Fill Color of an Object



In this example the background color of the circle is red for users with an access level value <= 100 and green for users with an access level > 100. The color animation may be useful to visually indicate the currently available user privilege on certain objects.