AC Python Documentation

import ac

The following function are available when you import “ac” module inside your python document.

-ac.getCarState(<CAR_IDENTIFIER>, <INFO_IDENTIFIER> , /*OPTIONAL*/ <OPTIONAL_IDENTIFIER>):
returns the <INFO_IDENTIFIER> type of information associated to car <CAR_IDENTIFIER>. The optional identifier can be omitted, it is used for special infos where they require a specific tyre, as described in the following section. The <OPTIONAL_IDENTIFIER> and it can be one of the following values:
On the left a synthetic description is given using blue color

FL,	Front Left
FR,	Front Right
RL,	Rear Left
RR,	Rear Right

Using the following <INFO_IDENTIFIER>s ac.getCarState returns a scalar value:

SpeedMS,	Current speed using Meters/Seconds [0, …]
SpeedMPH,	Current speed using Miles/Hour [0, …]
SpeedKMH,	Current speed using Kilometers/Hour [0, …]
Gas,	Pression on the Gas pedal [0,1]
Brake,	Pression on the Brake pedal [0,1]
Clutch,	Pression on the Clutch pedal [0,1]
Gear,	Current Gear [0,max gear]
BestLap,	Best Lap in milliseconds [0, …]
CGHeight,	Height of the center of gravity of the car from the ground [0, …]
DriftBestLap,	Best Lap points in Drift mode [0, …]
DriftLastLap,	Last Lap points in Drift mode [0, …]
DriftPoints,	Current Lap points in Drift mode [0, …]
DriveTrainSpeed,	Speed Delivered to the wheels [0, …]
RPM,	Engine’s rounds per minute [0, …]
InstantDrift,	Current drift points in Drift Mode [0, …]
IsDriftInvalid,	Current Drift is valid/invalid in Drift Mode {0,1}
IsEngineLimiterOn,	Engine Limiter On/Off {0,1}
LapCount,	Current Session Lap count [0, …]
LapInvalidated,	Is current Lap invalidated (by going out on the grass) {0, 1}
LapTime,	Current LapTime in milliseconds [0, …]
LastFF,	Last Force Feedback signal sent to the Wheel [0, …]
LastLap,	Last Lap in milliseconds [0, …]
NormalizedSplinePosition,	Position of the car on the track in normalized [0,1]

PerformanceMeter,	Projection of how many seconds is the current time far from the current best lap [0, …]
Steer,	Radians of steer rotation [-2pi,2pi]
TurboBoost,	Turbo gain on engine torque for specific vehicles [0,...]
Caster,	Caster Angle in radians

Using the following <INFO_IDENTIFIER>s ac.getCarState returns a 3D vector (with x,y,z components):

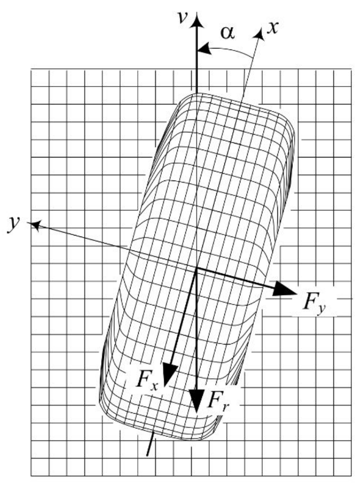
AccG,	Gravity acceleration on the vehicle's GC x,y,z = [0, …]
LocalAngularVelocity,	Get the angular velocity of the car, using the car as origin x,y,z =[0, …]
LocalVelocity,	Get the velocity using the car as origin x,y,x = [0, …]
SpeedTotal,	Get all the speed representation x= kmh, y = mph, z = ms
Velocity,	Current velocity vector x,y,z = [0, …]
WheelAngularSpeed,	Current Wheel angular speed x,y,z = [0, …]
WorldPosition	Current Car Coordinates on map x,y,z = [0,...]

Using the following <INFO_IDENTIFIER>s ac.getCarState returns a 4D vector (with w,x,y,z components):

CamberRad,	The camber angle in Radiants for each tyre
CamberDeg,	The camber angle in Degree for each tyre
SlipAngle,	Slip angle, angle between the desired direction and the actual direction of the vehicle [0, 360], degrees.
SlipRatio,	Slip Ration of the tyres x,y,z,w = [0,1]
Mz,	Self Aligning Torque x,y,z,w = [0, …]
Load,	Current load on each tyre x,y,z,w = [0,...]
TyreRadius,	Radius of any Tyre x,y,z,w = [0,...]
NdSlip,	
TyreSlip,	
Dy,	
CurrentTyresCoreTemp	Current core temperature of the tyres °C x,y,z,w = [0,...]
ThermalState,	Current temperature of the tyres °C x,y,z,w = [0,...]
DynamicPressure,	Current pressure of the tyres psi x,y,z,w =[0, …]
TyreLoadedRadius,	Radius of the tyre under load x,y,z,w =[0, …]
SuspensionTravel,	Suspension vertical travel x,y,z,w =[0, …]
TyreDirtyLevel,	Quantity of dirt on the tyres x,y,z,w =[0, 10)

Using the following <INFO_IDENTIFIER>s combined with the <OPTIONAL_IDENTIFIER> ac.getCarState returns a 3D vector (with x,y,z components) related to the <OPTIONAL_IDENTIFIER> wheel:

TyreContactNormal,	Normal vector to tyre's contact point (z)
TyreContactPoint,	Tyre contact point with the tarmac
TyreHeadingVector,	Tyre Heading Vector (x)
TyreRightVector,	Tyre Right Vector (y)



Using the following <INFO_IDENTIFIER>s combined with the <OPTIONAL_IDENTIFIER> ac.getCarState returns a scalar vector (with x,y,z components) related to the <OPTIONAL_IDENTIFIER> index O:

Aero , o=0	drag Coefficient
Aero, o=1	lift Coefficient front
Aero, o=2	lift Coefficient rear

Example : ac.getCarState(0,TyreContactPoint,FR)

On failure ac.getCarState returns 0.

GENERAL INFO :

-ac.getDriverName(<CAR_ID>)
<CAR_ID> must be the car ID, 0 for the player's car
returns as a string the driver's name of the <CAR_ID> car.
This function returns the car name on success, -1 otherwise.

-ac.getTrackName(<CAR_ID>)
<CAR_ID> must be the car ID, 0 for the player's car
returns as a string the track's name where <CAR_ID> is running.
This function returns the track's name on success, -1 otherwise.

-ac.getTrackConfiguration(<CAR_ID>)
<CAR_ID> must be the car ID, 0 for the player's car
returns as a string the track's configuration where <CAR_ID> is running.
This function returns the track's name on success, -1 otherwise.

-ac.getCarName(<CAR_ID>)
<CAR_ID> must be the car ID, 0 for the player's car
returns as a string the car's name of the <CAR_ID> car.
This function returns the car name on success, -1 otherwise.

-ac.getLastSplits(<CAR_ID>)
<CAR_ID> must be the car ID, 0 for the player's car
returns as a Python List the car's last splits of the <CAR_ID> car.
This function returns the Python list with the splits on success, -1 otherwise.

-ac.isCarInPitline(<CAR_ID>)
<CAR_ID> must be the car ID, 0 for the player's car
This function returns 1 if the car is currently in the Pitline

-ac.isCarInPit(<CAR_ID>)

<CAR_ID> must be the car ID, 0 for the player’s car
This function returns 1 if the car is currently in the Pitbox

`-ac.isConnected(<CAR_ID>)`
<CAR_ID> must be the car ID, 0 for the player’s car
This function returns 1 if the car is currently connected

`-ac.getCarBallast(<CAR_ID>)`
<CAR_ID> must be the car ID, 0 for the player’s car
This function returns the ballast value of the car

`-ac.getCarMinHeight(<CAR_ID>)`
<CAR_ID> must be the car ID, 0 for the player’s car
This function returns the minimum height of the car

`-ac.getServerName()`
This function returns the name of the joined server

`-ac.getServerIP()`
This function returns the IP of the joined server

`-ac.getServerHttpPort()`
This function returns the Http port of the joined server

`-ac.getServerSlotsCount()`
This function returns total number of occupied slot in the server

`-ac.getCarsCount()`
This function returns the session’s max number of cars.

`-ac.getCarLeaderboardPosition(<CAR_ID>)`
<CAR_ID> must be the car ID, 0 for the player’s car
This function returns position of the car on the Leaderboard

`-ac.getCarRealTimeLeaderboardPosition(<CAR_ID>)`
<CAR_ID> must be the car ID, 0 for the player’s car
This function returns position of the car on realtime

`-ac.getCarFFB()`
This function returns the FFB Gain value for the player current car

`-ac.setCarFFB(<VALUE>)`
<VALUE>is the value taht will be added to the current FFB gain
This function sets the FFB Gain value for the player current car

CAMERA MANAGEMENT

`-ac.setCameraMode(<INFO_IDENTIFIER>)`
<INFO_IDENTIFIER> can be (CM):
Cockpit
Car
Drivable
Track
Helicopter
OnBoardFree
Free
Random
ImageGeneratorCamera
Start

`-ac.getCameraMode()`
This function returns a <INFO_IDENTIFIER> above

`-ac.isCameraOnBoard(<CAR_ID>)`
<CAR_ID> must be the car ID, 0 for the player's car
This function returns 1 on success, -1 otherwise.

`-ac.setCameraCar(<CAMERA_ID>,<CAR_ID>)`
<CAMERA_ID> must be the F6 camera index
<CAR_ID> must be the car ID, 0 for the player's car
This function returns 1 on success, -1 otherwise.

`-ac.getCameraCarCount(<CAR_ID>)`
<CAR_ID> must be the car ID, 0 for the player's car
This function returns the number of F6 cameras, -1 otherwise.

`-ac.focusCar(<CAR_ID>)`
<CAR_ID> must be the car ID, 0 for the player's car
This method get switch the actor to the selected car
This function returns 1 on success, -1 otherwise.

`-ac.getFocusedCar()`
This method get the selected car index

DEBUG
`-ac.log(<VALUE>):`
<VALUE> must be a string

Use ac.log if you want to send some text to the AC log.txt file
The function returns 1 on success

`-ac.console(<VALUE>)`
<VALUE> must be a string
Use ac.console to send a string to the AC console
The function returns 1 on success

GENERAL APP MANAGEMENT:

`-ac.newApp(<VALUE>)`
<VALUE> must be a string
Creates a new app and returns the corresponding Identifier.
returns the App ID on success, -1 otherwise

`-ac.setTitle(<CONTROLO_IDENTIFIER>,<TITLE>)`
<TITLE> must be a string, <CONTROL_IDENTIFIER> must be a form
This function will set the title of the specified App by <CONTROL_IDENTIFIER>.
The function returns 1 on success, -1 otherwise

`-ac.setSize(<CONTROLO_IDENTIFIER>,<WIDTH>,<HEIGHT>)`
<WIDTH>,<HEIGHT> must be a floating point numbers
This function will set the size of a control specified by <CONTROL_IDENTIFIER> .
The function returns 1 on success, -1 otherwise

`-ac.addLabel(<CONTROL_IDENTIFIER>,<VALUE>)`
<VALUE> must be a string
It is possible to add a label to a Window, we need to pass the Window to ac.addLabel and the label name.
The function returns 1 on success, -1 otherwise

`-ac.setPosition(<CONTROLO_IDENTIFIER>,<X>,<Y>)`
<X>,<Y> must be a floating point numbers
Use ac.setPosition to set the control's position specified by <CONTROL_IDENTIFIER> in the app.
The function returns 1 on success, -1 otherwise

`-ac.setIconPosition(<CONTROLO_IDENTIFIER>,<X>,<Y>)`
<X>,<Y> must be a floating point numbers, <CONTROL_IDENTIFIER> must be a form
Use ac.setPosition to set the new icon’s position instead of the default one.
The function returns 1 on success, -1 otherwise

`-ac.setTitlePosition(<CONTROLO_IDENTIFIER>,<X>,<Y>)`
<X>,<Y> must be a floating point numbers, <CONTROL_IDENTIFIER> must be a form
Use ac.setPosition to set the new title’s position inside the app.
The function returns 1 on success, -1 otherwise

`-ac.getPosition(<CONTROLO_IDENTIFIER>)`

<CONTROL_IDENTIFIER> is the identifier of a control

Use ac.getPosition to get the control's position in the parent window, this function returns a python tuple width,height.

The function returns the position as a tuple x,y on success, -1 otherwise

`-ac.setText(<CONTROL_IDENTIFIER>, <VALUE>)`

<VALUE> must be a string, <CONTROL_IDENTIFIER> is the control that we want to set the text to

Set the text of the control specified by <CONTROL_IDENTIFIER>, with the <VALUE> text passed as an argument.

The function returns 1 on success, -1 otherwise

`-ac.getText(<CONTROLO_IDENTIFIER>)`

<CONTROL_IDENTIFIER> is the control that we want to get the text from

Use ac.getText to get the control's text.

This function returns the coordinates x,y of the control on success, -1 otherwise

`-ac.setBackgroundOpacity(<CONTROL_IDENTIFIER>, <VALUE>)`

<VALUE> must be a floating point value between 0 and 1

Use ac.setBackgroundOpacity to change the alpha channel of the desired control.

The function returns 1 on success, -1 otherwise

`-ac.drawBackground(<CONTROL_IDENTIFIER>, <VALUE>)`

<VALUE> must be 0 or 1

Use ac.drawBackground to set the background visible (1)(DEFAULT) or transparent (0)

The function returns 1 on success, -1 otherwise

`-ac.drawBorder(<CONTROL_IDENTIFIER>, <VALUE>)`

<VALUE> must be 0 or 1

Use ac.drawBorder to draw the border of the desired control (1) (DEFAULT) or not (0)

The function returns 1 on success, -1 otherwise

`-ac.setBackgroundTexture(<CONTROL_IDENTIFIER>, <PATH>)`

<PATH> starts from Assetto Corsa root folder, <CONTROL_IDENTIFIER> must be a control identifier

Use ac.setBackgroundTexture to draw a specified texture stored in the path specified by <PATH> as background image for the control specified by <CONTROL_IDENTIFIER>.

The function returns 1 on success, -1 otherwise

`-ac.setFontAlignment(<CONTROL_IDENTIFIER>, <ALIGNMENT>)`

<ALIGNMENT> is one of the following strings:

“left “

“right”

“center”

Use ac.setFontAlignment to set the font alignment of the control text as specified by the <ALIGNMENT> string.
The function returns 1 on success, -1 otherwise

-ac.setBackgroundColor(<CONTROL_IDENTIFIER>, <R>,<G>,<B>)
<PATH> starts from Assetto Corsa root folder
Use ac.setBackgroundColor to set the background color of the window as specified by the R,G,B values
The function returns 1 on success, -1 otherwise

-ac.setVisible(<CONTROL_IDENTIFIER>, <VALUE>)
<VALUE> must be 0 or 1
It is possible to hide the object using the function ac.setVisible with VALUE set to 1.
The function returns 1 on success, -1 otherwise

-ac.addOnAppActivatedListener(<CONTROL_IDENTIFIER>, <VALUE>)
<VALUE> must be a function name defined inside the Python script,
<CONTROL_IDENTIFIER> must be an app.
This method set the <VALUE> function as callback function for the event of app selection on the task bar.
The function returns 1 on success, -1 otherwise

-ac.addOnAppDismissedListener(<CONTROL_IDENTIFIER>, <VALUE>)
<VALUE> must be a function name defined inside the Python script,
CONTROL_IDENTIFIER> must be an app.
This method set the <VALUE> function as callback function for the event of app deselection on the task bar.
The function returns 1 on success, -1 otherwise

-ac.addRenderCallback(<CONTROL_IDENTIFIER>, <VALUE>)
<VALUE> must be a function name defined inside the Python script
This method set the <VALUE> function as callback function for the finished rendering event.
The function returns 1 on success, -1 otherwise

-ac.setFontColor(<CONTROL_IDENTIFIER>,<R>,<G>,<B>,<A>)
<CONTROL_IDENTIFIER> must be a ControlIdentifier, <R>,<G>,<B>,<A> are the color value scaled from 0 to 1
This function returns 1 on success, -1 otherwise

-ac.setFontSize(<CONTROL_IDENTIFIER>, <VALUE>)
This method set <VALUE> as new new size of the control’s font.
The function returns 1 on success, -1 otherwise

-ac.initFont(0, <FONTNAME>, <ITALIC>, <BOLD>)
This method loads the font in memory (stored as font+italic+bold). Should be used at the initialization of the application. The font needs to be saved in the “content/fonts” folder.

<FONTNAME> must be the name of the base font (real name, not filename, so it can be “Arial”)
<ITALIC> must be a 0 for non-italic font, 1 for italic font (instance will be “Arial Italic”)
<BOLD> must be a 0 for non-bold font, 1 for bold font (instance will be “Arial Bold” or “Arial Italic Bold” is also italic)
The function returns 1 on success, -1 otherwise

`-ac.setCustomFont(<CONTROL_IDENTIFIER>, <FONTNAME>, <ITALIC>, <BOLD>)`
This method create or replace the font of a control. Should not be done without calling “iniFont”. The font needs to be saved in the “content/fonts” folder.
<CONTROL_IDENTIFIER> is the control owner of the font
<FONTNAME> must be the name of the base font (real name, not filename, so it can be “Arial”)
<ITALIC> must be a 0 for non-italic font, 1 for italic font (instance will be “Arial Italic”)
<BOLD> must be a 0 for non-bold font, 1 for bold font (instance will be “Arial Bold” or “Arial Italic Bold” is also italic)
The function returns 1 on success, -1 otherwise

SPECIFIC CONTROL MANAGEMENT:

Button:
`-ac.addButton(<CONTROL_IDENTIFIER>, <VALUE>)`
<VALUE> must be a string, <CONTROL_IDENTIFIER> must be a form
The function adds a Button to the window specified by <CONTROL_IDENTIFIER>
The function returns the Button ID on success, -1 otherwise

`-ac.addOnClickedListener(<CONTROL_IDENTIFIER>, <VALUE>)`
<VALUE> must be a function name defined in this file
It is possible to associate the button with an event to trigger when it is clicked using this function
The function returns 1 on success, -1 otherwise

Graph:
`-ac.addGraph(<CONTROL_IDENTIFIER>, <VALUE>)`
<VALUE> must be a string
The function adds a Graph to the window specified in <CONTROL_IDENTIFIER>
The function returns the Graph ID on success, -1 otherwise

`-ac.addSerieToGraph(<CONTROL_IDENTIFIER>, <R>,<G>,<B>)`

<R>,<G>,<B> must be floating point numbers between 0 and 1
To plot some data it is necessary to add a Serie. A serie is a succession of points to plot on the graph.
When adding a serie you must specify the color of the serie as argument
The function returns 1 on success, -1 otherwise

-ac.addValueToGraph(<CONTROL_IDENTIFIER,<SERIE_INDEX>,<VALUE>)
<SERIE_INDEX> is the Serie ID in the graph that where <VALUE> will be added.
The function returns 1 on success, -1 otherwise

-ac.setRange(<CONTROL_IDENTIFIER>,<MIN_VALUE>,<MAX_VALUE>,<MAX_POINTS>)
<MIN_VALUE>,<MAX_VALUE>,<MAX_POINTS> must be floating point numbers
In order to plot the data inside the Graph it is necessary to specify the amplitude of the ordinates and the maximum number of points to store in memory
The function returns 1 on success, -1 otherwise

Spinner:
-ac.addSpinner(<CONTROL_IDENTIFIER>, <VALUE>)
<VALUE> must be a string
It is possible to add a Spinner using the function
The function returns the Spinner ID on success, -1 otherwise

-ac.setRange(<CONTROL_IDENTIFIER>, <MIN_VALUE>,<MAX_VALUE>)
<MIN_VALUE>,<MAX_VALUE> must be floating point numbers
It is possible to set the min and max values of the Control:
The function returns 1 on success, -1 otherwise

-ac.setValue(<CONTROL_IDENTIFIER>, <VALUE>)
<VALUE> must be floating point number
This function set the "value" parameter of the specific Control if this is an available parameter.
This function affects controls like Spinner, Progress Bar or Check Box
The function returns 1 on success, -1 otherwise

-ac.getValue(<CONTROL_IDENTIFIER>)
<VALUE> must be floating point number
This function returns the "value" parameter of the specific Control if this is an available parameter.
The function returns the value on success, -1 otherwise

-ac.setStep(<CONTROL_IDENTIFIER>,<VALUE)
<CONTROL_IDENTIFIER> must be a Spinner ID <VALUE> must be floating point number
Set the value added or subtracted when the + or - button is pressed in a Spinner controller.
The function returns 1 on success, -1 otherwise

-ac.addOnValueChangeListener(<CONTROL_IDENTIFIER>, <VALUE>)

<VALUE> must be a function name defined inside the Python script
It is now possible to associate the spinner with an event to trigger when one of the two buttons is pressed
The function returns 1 on success, -1 otherwise

Progress Bar :

`-ac.addProgressBar(<CONTROL_IDENTIFIER>, <VALUE>)`
<VALUE> must be a string
It is possible to add a Progress Bar using the function
The function returns the Progress Bar ID on success, -1 otherwise

Input Text :

`-ac.addInputText(<CONTROL_IDENTIFIER>, <VALUE>)`
<VALUE> must be a string
It is possible to add an Input Text Field using the function
The function returns the Input Text ID on success, -1 otherwise

`-ac.setFocus(<CONTROL_IDENTIFIER>, <FOCUS>)`
<CONTROL_IDENTIFIER> must be an Input Text, <FOCUS> must be 0 or 1
If FOCUS is 1, this function sets the Input Text as first responder.
The function returns 1 on success, -1 otherwise

`-ac.addOnValidateListener(<CONTROL_IDENTIFIER>, <VALUE>)`
<VALUE> must be a function name defined inside the Python script
It is possible to associate the <CONTROL_IDENTIFIER> with an event to trigger when the enter key is pressed
The function returns 1 on success, -1 otherwise

List Box :

`-ac.addListBox(<CONTROL_IDENTIFIER>,<NAME>)`
<CONTROL_IDENTIFIER> must be a window identifier
This method adds a List Box to the window specified by CONTROL_IDENTIFIER
The function returns the ListBox ID on success, -1 otherwise

`-ac.addItem(<CONTROL_IDENTIFIER>,<NAME>)`
<CONTROL_IDENTIFIER> must be a ListBox identifier
This method adds a List Box item to the List Box specified.
The item's label is specified by the Name string.
This function returns the ListBox Item ID on success, -1 otherwise

`-ac.removeItem(<CONTROL_IDENTIFIER>,<ID>)`
<CONTROL_IDENTIFIER> must be a ListBox identifier
This method removes from the List Box the item with ID as identifier
This function returns the size of the List Box on success, -1 otherwise

`-ac.getItemCount(<CONTROL_IDENTIFIER>)`

<CONTROL_IDENTIFIER> must be a ListBox identifier
This function returns the size of the List Box on success, -1 otherwise

-ac.setItemNumberPerPage(<CONTROL_IDENTIFIER>,<NUMBER>)
<CONTROL_IDENTIFIER> must be a ListBox identifier, <NUMBER> is the number of element to be displayed desired for each page
This function sets the number of element displayed for each page in a List Box.
This function returns 1 on success, -1 otherwise

-ac.highlightListBoxItem(<CONTROL_IDENTIFIER>,<ID>)
<CONTROL_IDENTIFIER> must be a ListBox identifier, <ID> is the element to be selected
This function sets the list box item with <ID> as identifier as selected.
This function returns 1 on success, -1 otherwise

-ac.addOnListBoxSelectionListener(<CONTROL_IDENTIFIER>, <VALUE>)
<VALUE> must be a function name defined inside the Python script
Control identifier must be a List Box controller otherwise the function does nothing and returns 0.
This method set the <VALUE> function as callback function for the event of item SELECTION inside a ListBox.
The callback function receives as input parameters the Item's NAME and its ID (his position inside the list-box).
The function returns 1 on success, -1 otherwise

-ac.addOnListBoxDeselectionListener(<CONTROL_IDENTIFIER>, <VALUE>)
<VALUE> must be a function name defined inside the Python script
Control identifier must be a List Box controller otherwise the function does nothing and returns 0.
This method set the <VALUE> function as callback function for the event of item DESELECTION inside a ListBox.
The callback function receives as input parameters the Item's NAME and its ID (his position inside the list-box).
The function returns 1 on success, -1 otherwise

-ac.setAllowDeselection(<CONTROL_IDENTIFIER>,<ALLOW_DESELECTION>)
<CONTROL_IDENTIFIER> must be a ListBox identifier, <ALLOW_DESELECTION> must be 0 or 1
Passing true as a parameter, when the user clicks on a selected item the item is de-selected.
In this way there could be 0 or 1 selected item at a given time.
If also ac.setAllowMultiSelection is set as true there can be more than 1 selected items at a given time.
This function returns 1 on success, -1 otherwise

-ac.setAllowMultiSelection(<CONTROL_IDENTIFIER>,<ALLOW_MULTI_SELECTION>)

<CONTROL_IDENTIFIER> must be a ListBox identifier, <ALLOW_MULTI_SELECTION> must be 0 or 1
Passing true as a parameter, when the user clicks on a different item the item is added to the selected item list.
In this way there could me more than one selected items at a given time
This function returns 1 on success, -1 otherwise

`-ac.getSelectedItems(<CONTROL_IDENTIFIER>)`
<CONTROL_IDENTIFIER> must be a ListBox identifier
This method returns the list of the selected items at a given time.
This function returns a Python list of Long on success, -1 otherwise

Check Box :
`-ac.addCheckBox(<CONTROL_IDENTIFIER>,<VALUE>)`
<CONTROL_IDENTIFIER> must be a form ,<VALUE> must be the form's name
This function adds a checkbox to the current form passed as <CONTROL_IDENTIFIER>.
The function returns the checkbox created on success, -1 otherwise

`-ac.addOnCheckBoxChanged(<CONTROL_IDENTIFIER>, <VALUE>)`
<VALUE> must be a function name defined inside the Python script
Control identifier must be a Check Box controller otherwise the function does nothing and returns 0.
This method set the <VALUE> function as callback function for the event of checkbox SELECTION or DESELECTION inside a ListBox.
The callback function receives as input parameters the CheckBox's NAME and its value, 1 if selected, -1 otherwise.
The function returns 1 on success, -1 otherwise

Text Box :
`-ac.addTextBox(<CONTROL_IDENTIFIER>,<NAME>)`
<CONTROL_IDENTIFIER> form identifier, <NAME> text box name
This method adds a text box (scrollable if the text is longer than the textbox, to the current form.
THIS CONTROL IS NOT CURRENTLY WORKING YET, so no set text has been exposed

GRAPHICS AND RENDERING :
`-ac.newTexture(<PATH>)`
<PATH> must be a string, the path is considered from AC installation directory.
This method loads in memory the texture specified by path.
This method returns the texture identifier on success, -1 otherwise.
`-ac.glBegin(<PRIMITIVE_ID>)`
<PRIMITIVE_ID> must be an int corresponding to the following ints:
0 : Draw lines
1 : Draw lines Strip
2 : Draw triangles
3 : Draw quads.

Begin a rendering of the specified type.
This function returns 1 on success, -1 otherwise

`-ac.glEnd(void)`
Finishes the render of a previous specified primitive
This function returns 1 on success

`-ac.glVertex2f(<X>,<Y>)`
<X>,<Y> must be a floating point numbers
Adds a 2d point to the rendering queue.
This function returns 1 on success, -1 otherwise

`-ac.glColor3f(<R>,<G>,<B>)`
<R>,<G>,<B> rgb coordinates scaled from 0 to 1, must be a floating point numbers
set the current rendering color to <R>,<G>,<B> color.
This function returns 1 on success, -1 otherwise

`-ac.glColor4f(<R>,<G>,<B>,<A>)`
<R>,<G>,<B> rgb coordinates scaled from 0 to 1, <A> alpha component from 0 to 1, all the values must be a floating point numbers
set the current rendering color to <R>,<G>,<B> color, with <A> as transparency.
This function returns 1 on success, -1 otherwise

`-ac.glQuad(<X>,<Y>,<WIDTH>,<HEIGHT>)`
<X>,<Y>,<WIDTH>,<HEIGHT> must be a floating point numbers
draw a quad quickly without using glBegin, … , glEnd
This function returns 1 on success, -1 otherwise

`-ac.glQuadTextured(<X>,<Y>,<WIDTH>,<HEIGHT>,<TEXTURE_ID>)`
<X>,<Y>,<WIDTH>,<HEIGHT> must be a floating point numbers, <TEXTURE_ID> is the id of the texture previously loaded.
draw a quad quickly without using glBegin, … , glEnd
This function returns 1 on success, -1 otherwise