



AJUI Tip

User Manual

1	<i>Introduction.....</i>	4
1.1	What is AJUI Tip.....	4
1.2	Composition of a tooltip.....	4
1.2.1	Technical overview.....	5
1.3	The positioning of the tooltip	5
1.4	Curve the arrow of a Tip.....	7
1.5	Life cycle of a tooltip	8
2	<i>List of properties</i>	9
2.1	Global.....	9
2.2	Target.....	10
2.3	Container.....	11
2.4	Text	13
2.5	Arrow	14
2.6	Subform.....	15
2.7	Closebox.....	15
2.8	Debug.....	15
2.9	Animations	16
2.9.1	Blink.....	16
2.9.2	Jump	17
2.10	Fade in and out.....	18
3	<i>Member functions not related to a property.....</i>	18
4	<i>Component methods.....</i>	19
5	<i>Start with AJUI Tip</i>	20
5.1	Installation	20
5.2	How to use it	20
5.2.1	Prerequisites.....	20
5.2.2	Names and reserved properties.....	20
5.2.3	Setting up your first Tip.....	21
5.2.4	Use a form instead of text in your Tip.....	22
5.2.5	Adding animations	22
5.2.6	Good practices for instance storage	24
5.3	Specific cases	24
5.3.1	Listbox	24
5.3.2	Multi-pages	25
5.3.3	Subform	25
5.3.4	Using the Form 4D command in a subform	26
6	<i>Conclusion</i>	26

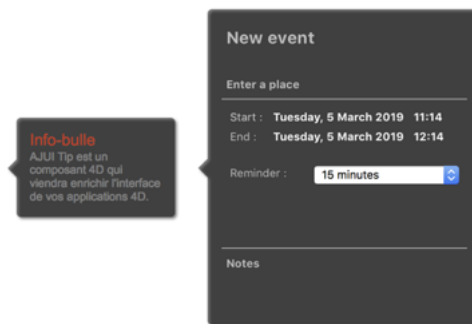
Version	Date	Comment (Change)	Author
1.0	05.03.2019	First release	Gary Criblez
1.1	27.06.2019	Small fixes	Gary Criblez
1.2	12.07.2019	Rework fade in / fade out and remove subform connector.	Gabriel Inzirillo
1.3	19.08.2019	Using the 4D Form object in a subform	Gary Criblez
1.4	05.09.2019	Curve the arrow of a Tip	Gary Criblez
1.5	13.12.2019	Modification import and export	Gary Criblez

1 Introduction

1.1 What is AJUI Tip

AJUI Tip is a component developed in the 4D language. Its use is intended for 4D developers. It allows you to dynamically generate and display tooltips (Tip) in the context of a form.

The component offers several methods to customize and manipulate them via an object and functions. These can be stored and reused in JSON formats as a template.



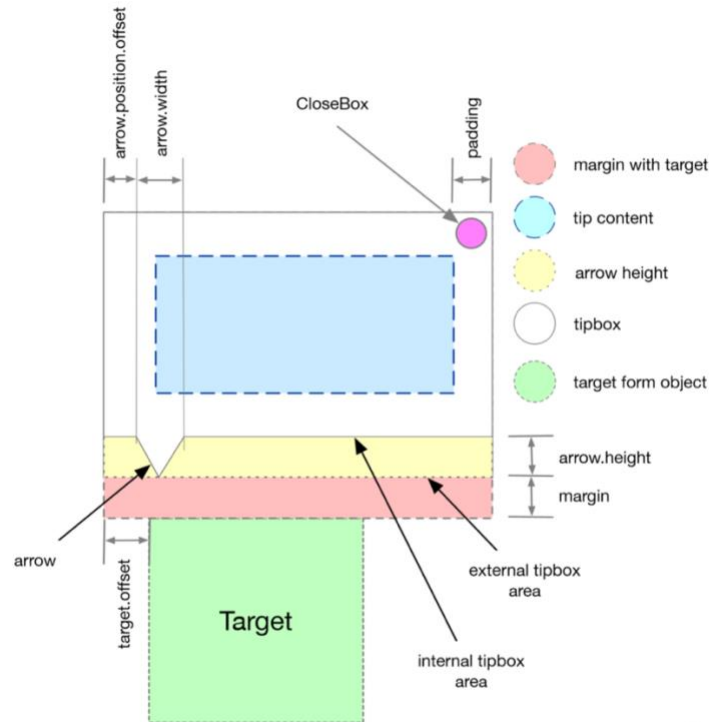
1.2 Composition of a tooltip

Basically, the tooltip is a shared form (container) present in the component that will be displayed via a subform in the target form of the main application. It contains three elements :

- **An image variable** : To build the image, the component relies on a number of properties. These will allow to draw an object in SVG language. It results from this, a set of calculations based on different criteria (content size, padding, radius, etc.) that will be performed in order to draw the outline of the tooltip and its arrow. The SVG will also be able to integrate a text box and a button to close the tooltip (closebox).
- **A sub-form** : This one allows you to integrate a form from the main database into the tooltip. The sub-form will take the place of the text box if it is defined.
- **An object variable** : Allows you to keep the properties of the tooltip required during various operations of the component..

This set forms what is called the **tooltip** or **tipbox** in English.

1.2.1 Technical overview



1.3 The positioning of the tooltip

The tooltip can be positioned manually by entering the coordinates of a point or by designating a form object as target. The component will retrieve the coordinates in this second case.

Then, it is possible to define the placement of the tooltip and its arrow :

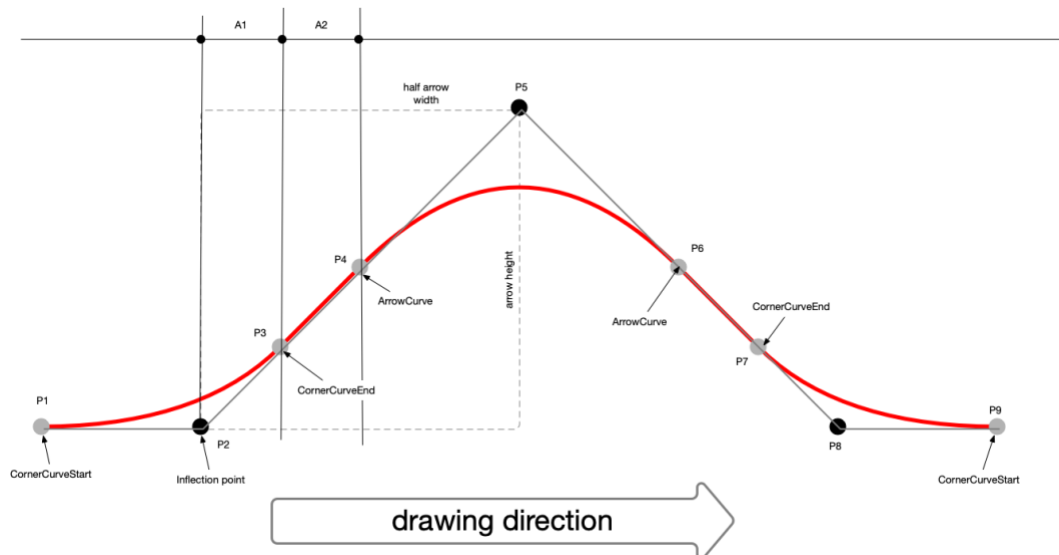
Horizontal positions:Vertical positions:

It is also possible to define a margin and/or offset with respect to the target (see the diagram in Chapter 1.2.1 - Technical overview).

1.4 Curve the arrow of a Tip

Since version 1.4 of AJUI Tip, it is possible to round off the three angles of a Tip's arrow. For this purpose, four formulas are available (see the properties concerning the arrow). In this chapter, we will explain how rounding is calculated in order to understand the use of these new formulas.

Curves calculation schema :



First of all, it is important to specify that a Tip is always drawn to the right from its upper left corner.

The first curve is calculated from the starting point (P1) to the ending point (P3) and using the inflection point (P2). The second curve is calculated from the starting point (P4) to the ending point (P6) and using the inflection point (P5). The third curve is calculated from the starting point (P7) to the ending point (P9) and using the inflection point (P8).

The three black dots on the diagram correspond to the three angles of the initial arrow. In the case of the rounding function, they will serve as an inflection point for drawing the curves.

The six grey points on the diagram are determined by ratios that you can assign for drawing the arrow using the formulas: **CornerCurveStart**, **CornerCurveEnd**, **ArrowCurve**.

On which data the ratios are applied in order to obtain these six points ?

The **CornerCurveEnd** and **ArrowCurve** ratios will be applied to the arrow height and half of the arrow width. This will allow us to obtain the X and Y values for the position of these points (P3, P4, P6 and P7).

For **CornerCurveStart**, the ratio applies only in relation to the height of the arrow. The result of this calculation will then be subtracted or added in relation to the inflection points (P2 and P8) over the width.

Applying these three ratios ensures that the rounding of the arrow is rendered consistently. AJUI Tip Lab includes an "HDI Curves" that will allow you to play with these ratios and functions.

1.5 Life cycle of a tooltip

Phase 1 : initialization and configuration

The first phase consists of **defining** the properties of the tooltip that will be displayed. To do this, the component offers a method that generates a base object representing a default definition. Then the user is free to use the functions attached to the object to customize its properties as he wishes. Note that the component proposes to import and export these objects in JSON format, this allows to use templates.

This step is very important, because all the other steps will be based on what has been defined in the object.

Phase 2 : calculations and generation

The second phase that is the most significant for the component is the creation of the tooltip based on its definition. During this phase, it will be in charge of retrieving the object's properties in order to perform a series of calculations and controls in order to generate/assign the different elements that will compose this set called the tooltip or tipbox (English).

This phase ends with the positioning of the tooltip according to the target and the associated elements that have been discussed previously.

Phase 3 : Display

The third phase consists in making the tooltip visible to the user. It is automatically triggered at the end of the previous step.

Its appearance can be done in two different ways. In the first case simply by changing the object from an invisible to a visible state. In the second case, if the user has set a fade effect in the display (fade in effect) then the component will gradually increase its opacity until it reaches its final shape.

Phase 4 : Life span and animations

In the fourth phase, the component takes care of the display duration or lifetime and the animations previously configured in phase 1. Currently two animations are available, the jump and the blink. It is not possible to launch several animations simultaneously for the same tooltip. The component will launch a single animation during this phase. The priority is the blinking if both animations are defined and activated. Animations are sensitive to the life span of the object.

By default, the lifetime of a tooltip is equal to zero (indeterminate and therefore unlimited display) and therefore does not impact the triggering of an animation. On the other hand, if the calculated time of an animation is greater than the defined lifetime then the component will not finish the animation and hide the tooltip.

Phase 5 : Closing

The fifth phase is automatically triggered and managed by the component if a lifetime is defined. Otherwise, it is up to the user to activate it either by using the close button on the tooltip (CloseBox) if it is active, or by triggering the component method that allows it to be hidden. It should be noted that if other tooltips are generated and if it is the same instance then the closing will be done automatically.

As for the closing itself, it is done in the same way as for the display but in the opposite direction. That is, either by changing from visible to invisible of the object, or by fading out, in which case the component gradually reduces the opacity of the Tip to zero %.

2 List of properties

A tooltip has a set of properties that will define its representation and interactions with the host form. In this chapter, we will review the different existing properties accessible by a formula acting as **Setter** and also as **Getter** if we do not pass them any parameter. All formulas can be called at the first level of the object.

It is recommended not to manually modify properties whose values are not defined by formulas because they are managed internally by the component.

2.1 Global

Global properties located on the second level of the object : **tip.global**

Name	Type	Description	Default value	Formula
name	string	Name of the tooltip (tip).	Empty string	TipName
tipDuration	longint	Time of appearance of the Tip (value in milliseconds). In case of a zero value, the Tip will not have a display time limit.	0	TipDuration
autoRebuild	boolean	Allows to force the reconstruction of the Tip automatically after each use of a formula (except for formulas related to animations). Enabling this mode may produce many calls, so it should be used sparingly.	False	AutoRebuild

Global – Formulas and their parameters

Formula name	Parameter(s)
TipName (name)	<ul style="list-style-type: none"> tip name (text)
TipDuration (duration)	<ul style="list-style-type: none"> Tip lifespan/display (real) <i>Value expressed in milliseconds</i>
AutoRebuild (activate)	<ul style="list-style-type: none"> Activer l'auto rebuild (booléen)

2.2 Target

Properties related to the target on the second level of the object : **tip.target**.

Name	Type	Description	Default value	Formula
name	string	Name of the target form object that will be linked to the Tip.	empty string	TargetName
coordinate	object	This object will contain the values of the target coordinates (left, top, right and bottom). The values are calculated automatically if the target is set.	0	TargetCoordinates
tipPosition	string	Position of the Tip in relation to the target.	right-middle	TipPosition
margin	longint	Allows you to set a gap between the Tip and the target.	0	TargetMargin
offset	longint	Allows you to shift the Tip relative to the target.	0	TargetOffset

Target – Formulas and their parameters

Formula name	Parameter(s)
TargetName (name)	<ul style="list-style-type: none"> Name of the target object (text)
TargetCoordinates (left { ; top { ; right { ; bottom }}})	<ul style="list-style-type: none"> Left coordinate (longint) Top coordinate (longint) (optional) Right coordinate (longint) (optional) Bottom coordinate (longint) (optional) <i>Value expressed in pixels</i>
TipPosition (position {default arrow })	<ul style="list-style-type: none"> Tip position (text) see chapter 1.3 Apply the default arrow position (Boolean) (optional)
TargetMargin (left { ; top { ; right { ; bottom }}})	<ul style="list-style-type: none"> Left margin (longint) Top margin (longint) (optional) Right margin (longint) (optional) Bottom margin (longint) (optional) <i>Value expressed in pixels</i>
TargetOffset (décalage)	<ul style="list-style-type: none"> Tip shift (longint) <i>Value expressed in pixels</i>

2.3 Container

Properties related to the container on the second level of the object : **tip.container**.

Nom	Type	Description	Par défaut	Formule - setter
backgroundColor	string	Background color of the container.	yellow	BackgroundColor
backgroundOpacity	longint	Container opacity (%).	50	BackgroundOpacity
borderColor	string	Borders color.	gray	BorderColor
borderOpacity	longint	Border opacity (%).	100	BorderOpacity
borderSize	longint	Border size	1	BorderSize
cornerRadius	longint	Value of the radius applied to the 4 corners of the Tips.	5	CornerRadius
padding	object	Contains the padding applied to the Tip.	10	Padding
maxHeight	longint	Allows you define the maximum width of the text box in the container. Not taken into account if maxWidth is greater than zero.	0	TipboxMaxHeight
maxWidth	longint	Allows you to define the maximum width of the text box in the container. Has priority over maxHeight for practical reasons of resizing the area.	0	TipboxMaxWidth
shadowOn	boolean	Enable the shadow effect.	False	ActivateShadow
shadowOffset	longint	Shadow shift in relation to the tooltip.	2	ShadowOffset

Container – Formulas and their leurs parameter(s)

Formula name	Parameter(s)
BackgroundColor (color)	<ul style="list-style-type: none"> Name/Color code (text) <i>Can be either the name of the color (e.g. "red") or the hexa value representing the RGB color (e.g. "#FF0000").</i>
BackgroundOpacity (opacity)	<ul style="list-style-type: none"> Opacity percentage (longint) <i>Value expressed in percent (0-100)</i>
BorderColor (color)	<ul style="list-style-type: none"> Name/Color code (text) <i>Can be either the name of the color (e.g. "red") or the hexa value representing the RGB color (e.g. "#FF0000").</i>
BorderOpacity (opacity)	<ul style="list-style-type: none"> Opacity percentage (longint) <i>Value expressed in percent (0-100)</i>
BorderSize (size)	<ul style="list-style-type: none"> Border size (longint) <i>Value expressed in pixels</i>
CornerRadius (radius)	<ul style="list-style-type: none"> Radius value (longint) <i>Value expressed in pixels</i>
Padding (size)	<ul style="list-style-type: none"> Internal margin size (longint) <i>Value expressed in pixels</i>
TipboxMaxHeight (width)	<ul style="list-style-type: none"> Max. container height (longint) <i>Value expressed in pixels</i>
TipboxMaxWidht (height)	<ul style="list-style-type: none"> Max. container width (longint) <i>Value expressed in pixels</i>
ActivateShadow (activate)	<ul style="list-style-type: none"> Enables the shaddow effect (boolean)
ShadowOffset (offset)	<ul style="list-style-type: none"> Offset value (longint)

2.4 Text

Properties related to the text displayed in the container on the second level of the object : **tip.text**.

Name	Type	Description	Default value	Formula
value	string	Text to be displayed.	Stylish example text	TextLabel
fontColor	string	Text color	black	TextFontColor
fontName	string	Name of the font to use. Several fonts can be entered by separating them with a comma without spaces. The component will search for the first usable font available.	Helvetica, Arial, MS Sans Serif	TextFontName
fontSize	longint	Text size.	11	TextFontSize
opacity	longint	Text opacity (%).	100	TextOpacity
rotation	string	Rotation of the text.	Empty string	TextRotation

Texte – Formulas and their parameters

Nom de formule	Paramètre(s)
TextLabel (text)	<ul style="list-style-type: none"> container text (text) <i>The text string can contain styled text.</i>
TextFontColor (color)	<ul style="list-style-type: none"> Name/Color code (text) <i>Can be either the name of the color (e.g. "red") or the hexa value representing the RGB color (e.g. "#FF0000").</i>
TextFontName (fontName)	<ul style="list-style-type: none"> Font name (text) <i>Can be a list of font names separated by a comma.</i>
TextFontSize (size)	<ul style="list-style-type: none"> Font size (longint) <i>Value expressed in pixels</i>
TextOpacity (opacity)	<ul style="list-style-type: none"> Text opacity in percentage (longint) <i>Value expressed in percent (0-100)</i>
TextRotation (orientation)	<ul style="list-style-type: none"> Direction of rotation, left or right (string)

2.5 Arrow

Properties related to the drawing of the Tip arrow on the second level of the object : **tip.arrow**.

Name	Type	Description	Default value	Formula
position	string	Position of the arrow.	middle	ArrowPosition
height	longint	Height size of the arrow.	16	ArrowHeight
width	longint	Width size of the arrow.	20	ArrowWidth
offset	longint	Allows you to add an offset of the arrow from its initial position related to the border (no effect when center and middle positions are set).	10	ArrowOffset
curved	boolean	Allows to curve the angles of the arrow.	false	Curved
arrowCurve	real	Ratio to be applied to the angle of the arrow tip (see chapter on arrow curves).	0.9	ArrowCurve
cornerCurveStart	real	Ratio to be applied, which defines the starting point of the curve for both the start and end angles of the arrow (see chapter on arrow curves).	0.5	CornerCurveStart
cornerCurveEnd	real	Ratio to be applied, which defines the end point of the curve for both the start and end angles of the arrow (see chapter on arrow curves).	0.15	CornerCurveEnd

Arrow – Formulas and their paramaters

Nom de formule	Paramètre(s)
ArrowPosition (position)	<ul style="list-style-type: none"> Arrow position (text) see chapter 1.3
ArrowHeight (height)	<ul style="list-style-type: none"> Arrow height (longint) <i>Value expressed in pixels</i>
ArrowWidth (width)	<ul style="list-style-type: none"> Arrow width (longint) <i>Value expressed in pixels</i>
ArrowOffset (décalage)	<ul style="list-style-type: none"> Arrow offset (longint) <i>Value expressed in pixels</i>
Curved (curve)	<ul style="list-style-type: none"> Curved arrow (boolean)
ArrowCurve (ratio)	<ul style="list-style-type: none"> Ratio of arrow tip (real)
CornerCurveStart (ratio)	<ul style="list-style-type: none"> Ratio to calculated the starting of the curve (real)
CornerCurveEnd (ratio)	<ul style="list-style-type: none"> Ratio to calculated the ending of the curve (real)

2.6 Subform

Properties related to the subform of the container at the second level of the object : **tip.subform**.

Name	Type	Description	Default value	Formula
name	string	Name of the form associated with the container subform.	Empty string	SubformName

Sub-form – Formulas and their paramameters

Formula name	Parameter(s)
SubformName (name)	<ul style="list-style-type: none"> Name of the form in the Host database (text)

2.7 Closebox

Properties related to the closebox located at the second level of the object : **tip.closeBox**.

Name	Type	Description	Default value	Formula
display	boolean	Display the closebox in the Tip container. When this value is set to true, the property "onClick" is also set to true.	False	CloseBoxDisplay
onClick	boolean	Enabling the onClick event of the closeBox.	False	ActivateOnClickCB
size	longint	Size of the closeBox. Must be a value between 2 min. and 20 max.	5	CloseBoxSize

Closebox – Formulas and their parameters

Formula name	Parameter(s)
CloseBoxDisplay (display)	<ul style="list-style-type: none"> Display (boolean)
ActivateOnClickCB (enable)	<ul style="list-style-type: none"> Enable the onClick event (boolean)
CloseBoxSize (size)	<ul style="list-style-type: none"> Size between 2 and 20 (longint) <p><i>Value expressed in pixels</i></p>

2.8 Debug

Properties activating debug elements on the second level of the object : **tip.debug**.

Name	Type	Description	Default value	Formula
showSVGArea	boolean	Display the drawing outline of the Tip	False	No
showSVGViewer	boolean	Launch SVGViewer on the how method execution	False	No

2.9 Animations

Properties related to the animations on the second level of the object : **tip.animation**.

Name	Type	Description	Default value	Formula
isActive	boolean	Enables animations (does not concern fade in and fade out)	True	ActivateAnimation
refresh	longint	Refresh value of animations. Expressed in number of drawing refreshes per second. This value applies to animations and for Fade in and Fade out effects.	20	AnimationRefresh

Animations – Formules et leurs paramètres

Formula name	Parameter(s)
ActivateAnimation (activation)	<ul style="list-style-type: none"> Enable animations (boolean)
AnimationRefresh (refresh)	<ul style="list-style-type: none"> Refresh value (longint) <i>Value expressed rate/secs</i>

2.9.1 Blink

Properties related to the Blink animation on the third level of the object : **tip.animation.blink**.

Name	Type	Description	Default value	Formula
number	longint	Number of Blink.	0	BlinkNumber
duration	longint	Duration of a Blink (in milliseconds).	1000	BlinkDuration
on	longint	Display time proportionally to the total duration of a blink (%).	50	BlinkOn

Blink – Formulas and their parameters

Formula name	Parameter(s)
BlinkNumber (number)	<ul style="list-style-type: none"> Number of effects (longint)
BlinkDuration (duration)	<ul style="list-style-type: none"> Duration of a blink (longint) <i>Value expressed in milliseconds</i>
BlinkOn (percent)	<ul style="list-style-type: none"> Percent time for the Blink ON (longint) <i>Value expressed in percent (0-100)</i>

2.9.2 Jump

Properties related to the Jump animation on the third level of the object : **tip.animation.jump**.

Name	Type	Description	Default value	Formula
number	longint	Number of jump effects.	0	JumpNumber
duration	longint	Duration of the jump effect, this includes the round trip (milliseconds).	1000	JumpDuration
offset	longint	Distance to go for the jump effect.	20	JumpOffset
on	longint	Time of going up proportionally to the total duration of a jump (%).	50	JumpOn

Jump – Formulas and their parameters

Formula name	Parameter(s)
JumpNumber (number)	<ul style="list-style-type: none"> Number of effects (longint)
JumpDuration (duration)	<ul style="list-style-type: none"> Duration of an effect (longint) <i>Value expressed in milliseconds</i>
JumpOffset (distance)	<ul style="list-style-type: none"> Distance de saut (longint) <i>Value expressed in pixels</i>
JumpOn (percent)	<ul style="list-style-type: none"> Percent time for the Jump ON (longint) <i>Value expressed in percent (0-100)</i>

2.10 Fade in and out

Properties related to the Fade effects on the third level of the object : **tip.animation.fade**.

Name	Type	Description	Default value	Formula
in	boolean	Enables the fade in animation	False	ActivateFadeIn
out	boolean	Enables the fade out animation	False	ActivateFadeOut
timer.in	longint	Duration of the fade in effect	1000	TimerFadeIn
timer.out	longint	Duration of the fade out effect	1000	TimerFadeOut

Fade in and out – Formulas and their paramters

Formula name	Parameter(s)
ActivateFadeIn (activer)	<ul style="list-style-type: none"> Enable the fade in effect (boolean)
ActivateFadeOut (activer)	<ul style="list-style-type: none"> Enable the fade out effect (boolean)
TimerFadeIn (durée)	<ul style="list-style-type: none"> Duration of the fade in effect (longint) <i>Value expressed in milliseconds</i>
TimerFadeOut (durée)	<ul style="list-style-type: none"> Duration of the fade out effect (longint) <i>Value expressed in milliseconds</i>

3 Member functions not related to a property

Show() and **Hide()** are two component's formulas that are not directly related to a property.

The **Show()** method allows you to launch the build of the Tip and make it visible.

The **Hide()** method makes the Tip invisible and applies a fade out if such effect has been defined. It also takes care of interrupting the various Worker(s) at work if an animation was in progress.

Export (templateName { ; TemplateFolderPath}) : This formula allows you to create a template of your AJUI Tip object in a file in JSON format. The formula requires in the first parameter the name associated with the template (file name). A second optional parameter allows you to specify the folder path, otherwise it will be exported to the default folder (.../Resources/AJUI_Tip_Templates/).

4 Component methods

AJUI_Tip_info

Parameter(s)	• None
Return value	Version number (text)
Description	This method returns the version number of the component when it has been compiled.

AJUI_TIP_loadTemplates ({ FolderPath })

Parameter(s)	• FolderPath : Chemin du dossier (texte) (optionnel)
Return value	Tip collection (object collection)
Description	This method allows you to load all the templates files contained in a directory. It returns all the content of the files as a collection of Tip objects. It is possible to define a target folder path as a parameter, if not, the method will use the default path : resources/AJUI_Tip_Templates/ .

New AJUI_Tip ({ tip })

Parameter(s)	• Template (object) (optional)
Return value	Une instance de Tip
Description	<p>This method returns an object variable that represents an instance of AJUI Tip. It contains all the properties and their default values as well as the formulas (member functions) to manipulate them. It is possible to pass an object as a parameter to it in order to import an AJUI Tip template (JSON file). The object expects as properties:</p> <ul style="list-style-type: none"> - <i>templateName</i> : Corresponds to the name of the JSON file to import (template). If the file is not found, the method will return a new instance of AJUI Tip. - <i>templatePath</i> (optional): You can specify a path to retrieve the file otherwise, the component will search in the default folder located in the resources (.../Resources/AJUI_Tip_Templates/).

AJUI_Tip_clearCache ()

Paramètre(s)	• Aucun
Valeur de retour	Aucune
Description	This method allows you to clean up instances stored internally in an interprocess variable. This is important in order to free up memory. We recommend that you use this method when you close the process that contains the form(s) that display tooltips. This method cleans only the instances of the current process.

5 Start with AJUI Tip

5.1 Installation

AJUI Tip must be placed in the component folder (Components) of your application.

5.2 How to use it

In this section, we describe the sequence of operations to be performed in order to generate a simple Tip in the context of the main form.

5.2.1 Prerequisites

AJUI Tip has a shared form (container) that is used to visually create all the elements of the Tip. However, it must be made available on the main application form. For this, AJUI Tip needs a subform object that will serve as a model. This object is provided with the "AJUI_Tip_Library.4IL" library that you can open from your application (/File/Open/Object Library...) and you just have to drag & drop the "subform Tip" object on your main form.



Why is that necessary? Because when you want to display a Tip with the "Show" method, the component will use the name passed in properties to see if a Tip object corresponding to it already exists. If this is not the case, it will duplicate the template subform you have already added and it will associate an instance of the shared form to it.

So, the first thing, when you have decided which form will host the Tip(s) you will use, is to add this object.

5.2.2 Names and reserved properties

The component uses names or properties that are specific to it and that it considers to be reserved.

- **AJUI_Tip_container** : Name of the form shared by the component.
- **AJUI_Tip_sf** : Name to be used by the subform used as a template (see prerequisites).

5.2.3 Setting up your first Tip

The first step consists in calling the initialization method (*new AJUI_Tip*) in order to retrieve in an object, an instance containing all the properties of a default Tip as well as its formulas. We recommend that you use the "Form" variable directly to store the current instance in order to easily navigate between your form method and those of your form objects.

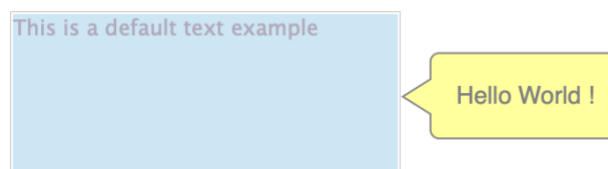
The second step consists in using the formulas of your object to define the properties of the Tip instance according to your needs. The minimum consists in defining the name of a target, i.e. the form object, to which the Tip instance will be attached. It is also possible to define precise coordinates if necessary, if the tip does not apply to a 4D form object. Then you need to define a name that will remain associated with the Tip instance. And finally make sure you define a text to display. This text can be multi-style text, it will be possible to put words or letters in bold, italic or color inside a text box.

```
$evt:=Form event
Case of
  ▾ : ($evt=On_Load)
    Form.tip:=New AJUI_Tip
    Form.tip.TargetName("test_tip")
    Form.tip.TipName("Test1")
    Form.tip.TextLabel("Hello World !")
End case
```

Finally, you must use the "Show" formula to launch the generation of the Tip instance. For example, do it on one of your form objects with an "On Clicked" event.

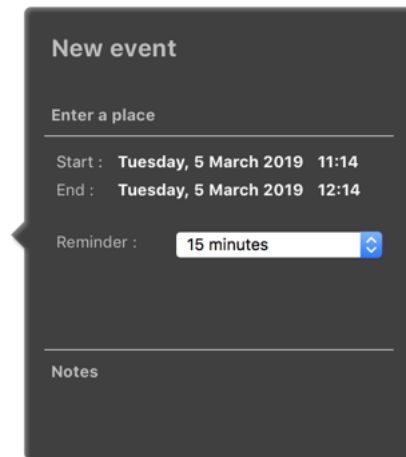
```
$evt:=Form event
  ▾ Case of
    ▾ : ($evt=On_Clicked)
      |   Form.tip.Show()
    End case
```

If the previous steps have been properly followed, you should see your Tip.



5.2.4 Use a form instead of text in your Tip

The steps in this chapter are the same except that it is possible to associate a form rather than text in a Tip. To do this, use the "*SubformName*" formula to associate the name of the form of your application you want to display in the tipBox with it.

A dark-themed tip box titled "New event". It contains a form with the following fields: "Enter a place" (text input), "Start : Tuesday, 5 March 2019 11:14" (text), "End : Tuesday, 5 March 2019 12:14" (text), "Reminder : 15 minutes" (dropdown menu), and "Notes" (text area).

It is not possible to define the display of a text AND a form in a Tip and therefore in such a case the component will use the form first. **Be careful when displaying a form, it disables fade in and fade out effects.**

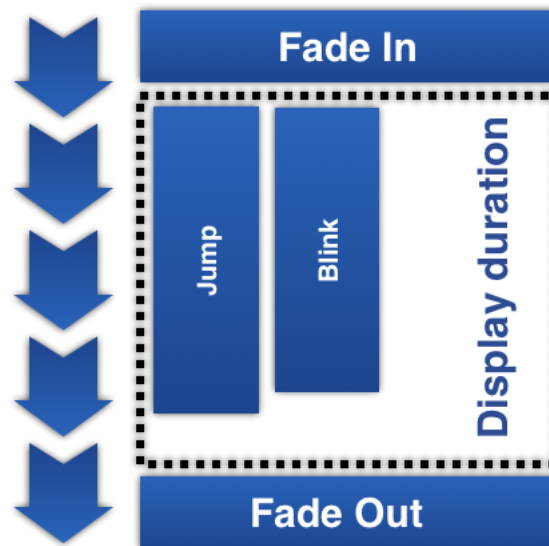
5.2.5 Adding animations

AJUI Tip provides users with the ability to animate a tooltip. (see chapter on life cycle).

Currently, the component allows the following animations to be created:

- Fade in
- Blink
- Jump
- Fade out

Fades have their own animation cycle so it is possible to activate them all. On the other hand, the other animations have a common cycle so only one can be added to the fade effects.



Let's move on to the implementation of the animations. We assume that you have prepared your tooltip as described in Chapter 4.2.3 "Setting up your first Tip". All that remains is to define the properties of the animations you want to apply.

Fade in and Fade out :

The implementation of these two effects is about the same. It is necessary to activate the effects (*activateFadeIn*, *activateFadeOut*) using the formulas. It is also possible to specify the duration of each effect.

```
Form.tip.ActivateFadeIn(True)
Form.tip.ActivateFadeOut(True)

Form.tip.TimerFadeIn(3000)
Form.tip.TimerFadeOut(3000)

Form.tip.TipDuration(3000)
```

Blink :

To set up the blink effect, three properties must be defined, including a mandatory one (*BlinkNumber*) that defines the number of cycles of the effect (a value of zero will not activate the effect). The other two properties concern the duration of a cycle as well as the proportion of time allocated (in percentage) to its visibility.

```
Form.tip.BlinkNumber(5)
Form.tip.BlinkDuration(1000)
Form.tip.BlinkOn(50)

Form.tip.Show()
```

Jump :

The setting of the jump effect is similar to that of the blink effect. It requires at least to specify the number of cycles of the effect. It is also possible to modify the duration of a cycle, the distance to be covered by the jump and the distribution (in percentage) of the duration between the first phase of the jump and the second phase.

```
Form.tip.JumpNumber(3)
Form.tip.JumpDuration(1000)
Form.tip.JumpOn(50)
Form.tip.JumpOffset(20)

Form.tip.Show()
```

5.2.6 Good practices for instance storage

It is recommended to use a different container for each different tip. A **tip** is different when its **tip** name is different. If you define an "info" tip and then an "alert" tip with specific properties, it is recommended to use a container specific to each of these definitions to avoid instance conflicts during animations (e. g. Form.info and Form.alert).

5.3 Specific cases

5.3.1 Listbox

If you want to display a tooltip on an element (cell, header, row, footer, etc.) of a listbox, you will use targeting by the use of coordinates instead of the target object's name.

Concerning the recovery of the coordinates, we advise you to retrieve the Mouse X and Mouse Y values on the click on the listbox to then determine the pointed cell. Once the calculation is done, you can define the coordinates of the cell to generate the tooltip. An example is provided with the [AJUI Tip Lab](#) test application.

The screenshot shows a window titled "Listbox" with a pink header. Below the header, there is a text box explaining that a coordinate point or rectangle can be used as a target position to display a tooltip. Below this is a table with 3 columns: Header1, Header2, and Header3. The table contains 10 rows of data (A1 to A10). A tooltip is displayed over the cell A10, showing the text "Footer: Footer 2 value: 100055". To the right of the table, there are input fields for Mouse x (246.38671875), Mouse y (445.7421875), left (145), top (445), right (265), and bottom (466). Below these fields is a "Hide Tip" button. At the bottom of the window, there is a text box containing the code: `Form.tip.setTargetCoordinates($left_1,$top_1,$right_1,$bottom_1)`. A "Close" button is located at the bottom right.

Header1	Header2	Header3
A1	10001	07.03.19
A2	10002	08.03.19
A3	10003	09.03.19
A4	10004	10.03.19
A5	10005	11.03.19
A6	10006	12.03.19
A7	10007	13.03.19
A8	10008	14.03.19
A9	10009	15.03.19
A10	10010	16.03.19

Footer: Footer 2 value: 100055

Form.tip.setTargetCoordinates(\$left_1,\$top_1,\$right_1,\$bottom_1)

5.3.2 Multi-pages

We will consider here the case where your form contains several pages and you want to display tooltips on different pages.

In practice, nothing has changed compared to the usual prerequisites. It will simply be necessary to take the caution to duplicate the subform object "AJUI_Tip_sf" on each of the pages and to add the suffix "_page" + page number. (Example: AJUI_Tip_Tip_sf_page2 for page 2). This rule does not apply to page 1 which keeps the initial name "AJUI_Tip_sf".

Important point in this regard, be careful to always generate your tooltips in the right page context. For example, if you generate your tooltip on the form loading method and then follow up with a page change, there will be a problem because the component will have created it in the first page but not in the open page.

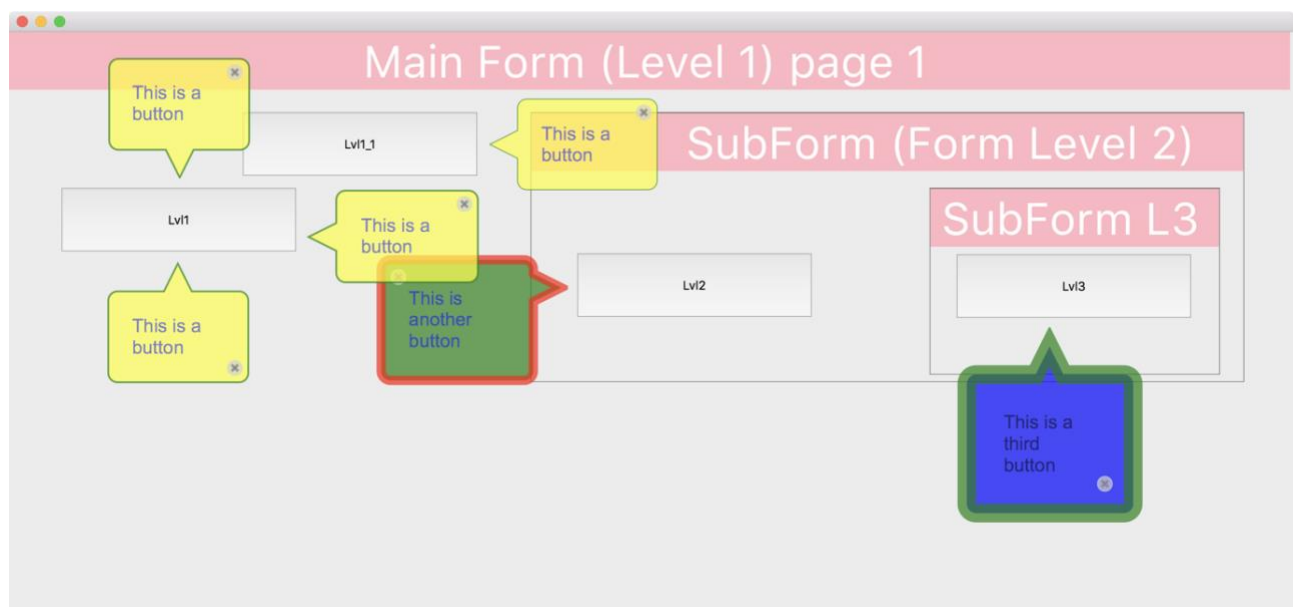
5.3.3 Subform

In this chapter we discuss the use of tooltips associated with a target object included in a subform. This case is quite simple to handle..

NB: It is important to understand that the subform object "AJUI_Tip_sf" must be at the level of the main form in order to be fully visible and in the foreground.

Once defined, the component will be responsible to rise the tooltip information to the first level so that it appears in its context.

Overview on three levels :



5.3.4 Using the Form 4D command in a subform

It is possible to associate the 4D Form object in the context of the Tip subform. To do this, you MUST define "Form.tip_sf_details:=new object ()" in the main form in which the Tip will be displayed. You can then add all the properties you need to this object. Once this is done, it is possible to communicate via this attribute between the main form and the subform loaded in the Tip.

6 Conclusion

The purpose of this document was to present the theoretical principles of the component as well as the different methods, formulas and properties at your disposal to be able to manage the construction and animation of tooltips.

As for the practical elements presented, they are intended to allow you to get off to a good start and to address some specific cases that could arise in the use of the component. For more practical information and a better visualization of the tooltip generation rendering, we invite you to try the test application (lab) that is provided with the component.

You want help for the implementation of the component `AJUI_Tip` in your application. You want to modify or extend its functionalities for a specific purpose. You want to have the source code of the `AJUI_Tip` component in order to perennize its use in your application with future versions of 4D. Feel free to contact us to discuss it.