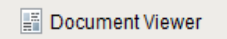
Document Viewer - Deprecated

This feature was removed from Ignition in version 7.9.9

General

Component Palette Icon:



Description

The document viewer is capable of loading and displaying a document that is available over the network at a URL. It is capable of displaying simple HTML and RTF documents. Although HTML links will be followed, it is not a fully functional interactive web browser. Its HTML support is rudimentary at best, and there is no JavaScript support. See the system.net.openURL function for a more robust solution for launching webpages, PDFs, etc.

This is component is useful for viewing machine manuals or operator protocol in HTML or RTF format. Note that in addition to HTML URLs (like "http://www.google.com"), you can load files as well using the URL format for files. Some examples:

- file://localhost/C:/myfolder/file.txt
- file://MyFileServer/resources/manuals/instructions.rtf

Properties

Name	Description	Property Type	Scripting	Category
Antialias	Draw with antialias on? Makes text smoother.	boolean	.antialias	Appearance
Background Color	The background color of the component.	Color	.background	Appearance
Border	The border surrounding this component. NOTE that the border is unaffected by rotation.	Border	.border	Common
Content Type	The content type of this document. Example: <code>text/html</code>	String	.contentType	Data
Enabled	If disabled, a component cannot be used.	boolean	.componentEnabled	Common
Font	Font of text on this component.	Font	.font	Appearance
Foreground Color	The foreground color of the component.	Color	.foreground	Appearance
Link Action	What happens when the user clicks on a hyperlink inside this document, if it is an HTML document.	int	.linkAction	Behavior
Mouseover Text	The text that is displayed in the tooltip which pops up on mouseover of this component.	String	.toolTipText	Common
Name	The name of this component.	String	.name	Common
Opaque	If false, backgrounds are not drawn. If true, backgrounds are drawn.	boolean	.opaque	Common
Page URL	Set this to a URL to display that page. If the url starts<P>with '/', it is assumed to be relative to the Gateway's HTTP address.	String	.page	Data
Text	The text of the document. Should match the content type.	String	.text	Data
Visible	If disabled, the component will be hidden.	boolean	.visible	Common

Scripting

Scripting Functions

This component does not have scripting functions associated with it.

Extension Functions

This component does not have extension functions associated with it.

Event Handlers

focus

focusGained

This event occurs when a component that can receive input, such as a text box, receives the input focus. This usually occurs when a user clicks on the component or tabs over to it.

.source	The component that fired this event.
.oppositeComponent	The other component involved in this focus change. That is, the component that lost focus in order for this one to gain it, or vise versa.

focusLost

This event occurs when a component that had the input focus lost it to another component.

.source	The component that fired this event
.oppositeComponent	The other component involved in this focus change. That is, the component that lost focus in order for this one to gain it, or vise versa.

hyperlink

hyperlinkUpdate

An integer that indicates whether the state was changed to "Selected" (on) or "Deselected" (off). Compare this to the event object's constants to determine what the new state is.

.source	The component that fired this event
.URL	The anchor URL of the hyperlink that was clicked, or None if the anchor isn't a valid URL
.description	Text representation of the anchor, useful in cases when the anchor wasn't a valid URL.
.eventType	Represents what kind of action was taken on the hyperlink, either "ACTIVATED", "ENTERED", or "EXITED". To test the event, use code like: str(event.eventType) == "ACTIVATED", str(event.eventType) == "ENTERED", str(event.eventType) == "EXITED"

key

keyPressed

An integer that indicates whether the state was changed to "Selected" (on) or "Deselected" (off). Compare this to the event object's constants to determine what the new state is.

.source	The component that fired this event.
.keyCode	The key code for this event. Used with the `keyPressed` and `keyReleased` events. See below for the key code constants.
.keyChar	The character that was typed. Used with the `keyTyped` event.
.keyLocation	Returns the location of the key that originated this key event. Some keys occur more than once on a keyboard, e.g. the left and right shift keys. Additionally, some keys occur on the numeric keypad. This provides a way of distinguishing such keys. See the KEY_LOCATION constants, the keyTyped event always has a location of KEY_LOCATION_UNKNOWN.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

keyReleased

Fires when a key is released and the source component has the input focus. Works for all characters, including non-printable ones, such as SHIFT and F3.

.source	The component that fired this event.
.keyCode	The key code for this event. Used with the `keyPressed` and `keyReleased` events. See below for the key code constants.
.keyChar	The character that was typed. Used with the `keyTyped` event.
.keyLocation	Returns the location of the key that originated this key event. Some keys occur more than once on a keyboard, e.g. the left and right shift keys. Additionally, some keys occur on the numeric keypad. This provides a way of distinguishing such keys. See the KEY_LOCATION constants in the documentation, the keyTyped event always has a location of KEY_LOCATION_UNKNOWN.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

keyTyped

Fires when a key is pressed and then released when source component has the input focus. Only works for characters that can be printed on the screen.

.source	The component that fired this event.
.keyCode	The key code for this event. Used with the `keyPressed` and `keyReleased` events. See below for the key code constants.
.keyChar	The character that was typed. Used with the `keyTyped` event.
.keyLocation	Returns the location of the key that originated this key event. Some keys occur more than once on a keyboard, e.g. the left and right shift keys. Additionally, some keys occur on the numeric keypad. This provides a way of distinguishing such keys. See the KEY_LOCATION constants in the documentation, the keyTyped event always has a location of KEY_LOCATION_UNKNOWN.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

mouse

mouseClicked

This event signifies a mouse click on the source component. A mouse click the combination of a mouse press and a mouse release, both of which must have occurred over the source component. Note that this event fires *after* the pressed and released events have fired.

.source	The component that fired this event.
.button	The code for the button that caused this event to fire.
.clickCount	The number of mouse clicks associated with this event.
.x	The x-coordinate (with respect to the source component) of this mouse event.
.y	The y-coordinate (with respect to the source component) of this mouse event.
.popupTrigger	Returns True (1) if this mouse event is a popup trigger. What constitutes a popup trigger is operating system dependent, which is why this abstraction exists.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

mouseEntered

This event fires when the mouse enters the space over the source component.

.source	The component that fired this event.
.button	The code for the button that caused this event to fire.
.clickCount	The number of mouse clicks associated with this event.
.x	The x-coordinate (with respect to the source component) of this mouse event.
.y	The y-coordinate (with respect to the source component) of this mouse event.
.popupTrigger	Returns True (1) if this mouse event is a popup trigger. What constitutes a popup trigger is operating system dependent, which is why this abstraction exists.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

mouseExited

This event fires when the mouse leaves the space over the source component.

.source	The component that fired this event.
.button	The code for the button that caused this event to fire.
.clickCount	The number of mouse clicks associated with this event.
.x	The x-coordinate (with respect to the source component) of this mouse event.
.y	The y-coordinate (with respect to the source component) of this mouse event.
.popupTrigger	Returns True (1) if this mouse event is a popup trigger. What constitutes a popup trigger is operating system dependent, which is why this abstraction exists.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

mousePressed

This event fires when a mouse button is pressed down on the source component.

.source	The component that fired this event.
.button	The code for the button that caused this event to fire.
.clickCount	The number of mouse clicks associated with this event.
.x	The x-coordinate (with respect to the source component) of this mouse event.
.y	The y-coordinate (with respect to the source component) of this mouse event.
.popupTrigger	Returns True (1) if this mouse event is a popup trigger. What constitutes a popup trigger is operating system dependent, which is why this abstraction exists.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

mouseReleased

This event fires when a mouse button is released, if that mouse button's press happened over this component.

.source	The component that fired this event.
.button	The code for the button that caused this event to fire.
.clickCount	The number of mouse clicks associated with this event.
.x	The x-coordinate (with respect to the source component) of this mouse event.
.y	The y-coordinate (with respect to the source component) of this mouse event.
.popupTrigger	Returns True (1) if this mouse event is a popup trigger. What constitutes a popup trigger is operating system dependent, which is why this abstraction exists.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

mouseMotion

mouseDragged

Fires when the mouse moves over a component after a button has been pushed.

.source	The component that fired this event.
.button	The code for the button that caused this event to fire.
.clickCount	The number of mouse clicks associated with this event.
.x	The x-coordinate (with respect to the source component) of this mouse event.
.y	The y-coordinate (with respect to the source component) of this mouse event.
.popupTrigger	Returns True (1) if this mouse event is a popup trigger. What constitutes a popup trigger is operating system dependent, which is why this abstraction exists.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

mouseMoved

Fires when the mouse moves over a component, but no buttons are pushed.

.source	The component that fired this event.
.button	The code for the button that caused this event to fire.
.clickCount	The number of mouse clicks associated with this event.
.x	The x-coordinate (with respect to the source component) of this mouse event.
.y	The y-coordinate (with respect to the source component) of this mouse event.
.popupTrigger	Returns True (1) if this mouse event is a popup trigger. What constitutes a popup trigger is operating system dependent, which is why this abstraction exists.
.altDown	True (1) if the Alt key was held down during this event, false (0) otherwise.
.controlDown	True (1) if the Control key was held down during this event, false (0) otherwise.
.shiftDown	True (1) if the Shift key was held down during this event, false (0) otherwise.

propertyChange

propertyChange

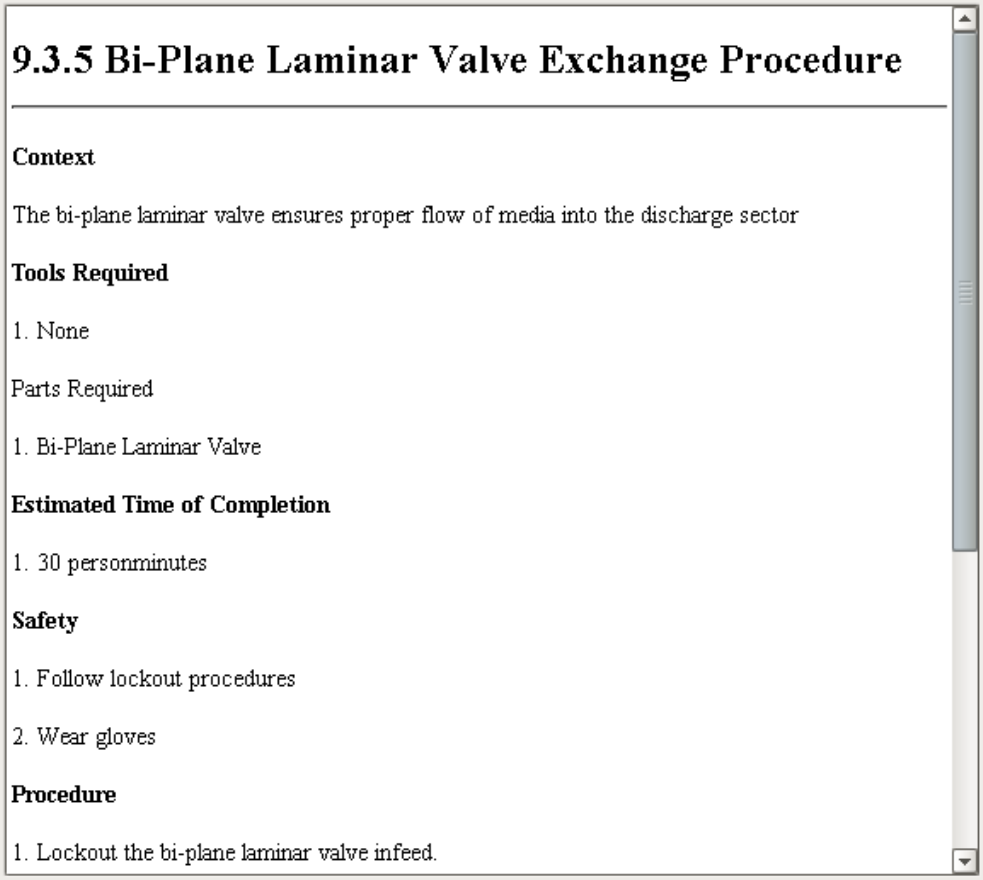
Fires whenever a *bindable* property of the source component changes. This works for standard and custom (dynamic) properties.

.source	The component that fired this event.
.newValue	The new value that this property changed to.
.oldValue	The value that this property was before it changed.
.propertyName	The name of the property that changed. NOTE: remember to always filter out these events for the property that you are looking for! Components often have many properties that change.

Customizers

- Component Customizers

Examples



9.3.5 Bi-Plane Laminar Valve Exchange Procedure

Context

The bi-plane laminar valve ensures proper flow of media into the discharge sector

Tools Required

1. None

Parts Required

1. Bi-Plane Laminar Valve

Estimated Time of Completion

1. 30 personminutes

Safety

1. Follow lockout procedures

2. Wear gloves

Procedure

1. Lockout the bi-plane laminar valve infeed.

Property Name	Value
Page URL	http://localhost:8088/main/system/webdev/test/Procedures/example.html