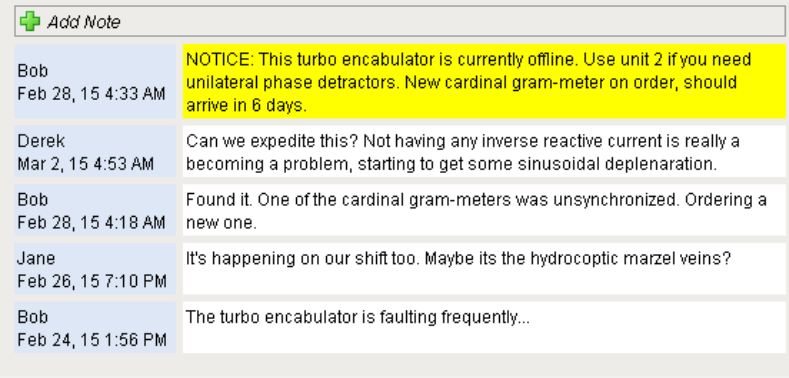
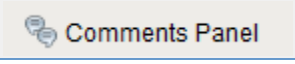
Legacy Comments Panel

General



Component Palette Icon:



INDUCTIVE UNIVERSITY

Comments Panel

Watch the Video

Description

The comments panel is used to power a blog-style comments system within your project. This can be useful for ad-hoc collaboration and communication between shifts, remote users, etc. This component is driven by a dataset that should be bound to a SQL query. Unlike most components, this component has built-in functionality to alter an external database. This is how the Add Note functionality works. You have the opportunity to alter the queries that the components uses by changing their properties.

The schema that typically drives this component involves up to two tables. One table (by default: Notes) stores all of the notes across the board. The second table (by default, ItemNotes) is used to associate notes with other things. This allows you to have different sets of notes for different screens/objects. Typically you'd bind the data to a query that joined these tables together restricting the second identifier in the ItemNotes table to the value appropriate for the window you're on. You'll also need to alter Insert Query 2's "YOURID" placeholder so that new notes get put in the right spot. You can opt out of this two-table system by simply clearing out Insert Query 2.

Users can be given the choice to remove their own comments, and comments can have files attached. To allow attachments, make sure you have a BLOB field in your notes table.

This component expects that its dataset is populated with the following columns. The names do not need to be exact, but the data type in your notes table must match.

- ID - an integer that should be the primary key for the notes table. Used for deleting and looking up attachments.
- Username - the user who added the note. Must be a string/varchar.
- Timestamp - when the note was added. Must be a Date or DateTime data type.
- NoteText - The text of the note itself. Must be a string/varchar.
- AttachmentFilename - filename for a file attached to the note. Must be a string/varchar.
- Sticky - 0 or 1 indicating whether or note the note is "sticky", which means it gets highlighted and put at the top. Must be a boolean or integer.

The comments panel also has a set of default queries to handle adding, deleting, and unsticking comments. The default queries expect certain database tables and columns to be set up. If your database is set up differently, or your dataset is different than the default, remember to change the default queries.

Table:Notes

- Id - An auto-incrementing integer that is the primary key. This maps to the ID field in the dataset.
- WhoID - A string or varchar mapping to the Username field in the dataset
- TStamp - A Date or DateTime mapping to the Timestamp field in the dataset
- Note - A string or varchar mapping to the NoteText field in the dataset.
- Filename - A string or varchar mapping to the AttachmentFilename in the dataset
- Sticky - A boolean or int mapping to the Sticky field in the dataset.
- Attachment - A blob to hold the attachment data. Blobs don't exist in MSSQL, so a varbinary() type must be used.

Table: ItemNotes

- AccountId - A user-defined field for use with Insert Query 2.
- NoteId - An integer that maps to the ID field in the dataset.

Tables: Users

- Username - A string or varchar that is used by Insert Query 1 to populate Notes.WhoID

Insert Query 1: INSERT INTO Notes (Note, WhoID, TStamp, Attachment, Filename, Sticky) VALUES (?, (SELECT Id FROM Users WHERE Username='%s'), CURRENT_TIMESTAMP, ?, ?, ?) This query will insert into your note table using the runPrepStmtGetKey() function and will be given four variables in the following order: note text, attachment blob, attachment filename, and sticky value. Also, it will pass in one string denoted by the %s. This is the name of the user that entered the note and does not need to be placed in any specific spot. If you WhoID field is a string, you can replace (SELECT Id FROM Users WHERE Username='%s') with '%s' to pass the username in directly.

Insert Query 2: INSERT INTO ItemNotes (AccountId, NoteId) VALUES (YOURID, %d) This query is optional and will insert the note id from Insert Query 1 into a mapping table of your choice. You must replace YOURID with something meaningful for your mapping table. This is most commonly done by binding this query to an expression. The reason for this second query is to have a mapping table to be joined to the note table to filter out which notes belong to a particular Comment Panel component.

Delete Query: DELETE FROM Notes WHERE Id=%d This query will use the note id from the component to delete the selected note.

Unstick Query: UPDATE Notes SET Sticky=0 WHERE Id=%d This query will use the note id from the component to set the sticky value to 0.

Download Attachment Query: SELECT Attachment FROM Notes WHERE Id=%d This query will use the note id from the component to download the attachment blob from the database.

There are many ways to set up the **Comments Panel** component, but the same first few steps exist in all cases to get started. To set up a simple **Comments Panel** example, here are the basic steps:

1. Add the **Comments Panel** component to a window.
2. Create or locate a database table to store notes in. You can follow the sample table in the "data" property of the **Comments Panel** Component.
3. Add a SQL query to the "data" property of the component to fetch the notes. Make sure to sort the records by timestamp so your newest comments are on top.
4. Modify the "Insert Query 1" property to match your notes table definition. You can remove the string from the "Insert Query 2" property.
5. Modify the "Delete Query", "Unstick Query", and "Download Attachment Query" properties to match your notes table definition.
6. Save your project and test it out! Check out the University video for a more thorough walk-through.

Properties

Name	Description	Property Type	Scripting	Category
Add Note Text	The word(s) used for the "Add Note" button.	String	.addNoteText	Appearance
Antialias	Draw with antialias on? Makes text smoother	boolean	.antialias	Appearance
Attach File Text	The word(s) used for the "Attach File" link.	String	.attachText	Appearance

Attachments Enabled	Controls whether or not files can be attached to notes.	boolean	.attachmentsEnabled	Behavior
Border	The border surrounding this component. NOTE that the border is unaffected by rotation.	Border	.border	Common
Cancel Text	The word(s) used for the "Cancel" button.	String	.cancelText	Appearance
Data	Fill this DataSet in with the notes for the desired entity. Columns are: Id, Username, Timestamp, NoteBody, Filename, IsSticky	Dataset	.data	Data
Data Quality	The data quality code for any tag bindings on this component.	int	.dataQuality	Data
Database Connection	Name of the database connection to run the queries against. Leave blank to use project's default connection.	String	.datasource	Behavior
Date Format	The format string to use for the date of the note.	String	.dateFormat	Appearance
Delete Mode	Controls if anyone can delete any note, no one can delete a note, or only owners can delete their notes	int	.deleteMode	Behavior
Delete Query	This query is used for deleting a note. <tt>%d</tt> is replaced with the note's ID	String	.deleteQuery	Behavior
Display Mode	Horizontal display mode will layout so that the comment header will be positioned to the left of the comment. Vertical display mode will have the comment header above the comment.	int	.displayMode	Behavior
Download Attachment Query	This query is used for downloading binary attachments. <tt>%d</tt> is replaced with the note's ID	String	.getAttachmentQuery	Behavior
Download Mode	What to do when an attachment is downloaded.	int	.downloadMode	Behavior
Enabled	If disabled, a component cannot be used.	boolean	.componentEnabled	Common
Font	Font of text of this component	Font	.font	Appearance
Foreground Color	The foreground color of the component.	Color	.foreground	Appearance
Header Color	The background color of the header notes	Color	.headersColor	Appearance
Insert Query 1	This insert query will insert a new note into a notes table. The placeholder <tt>%s</tt> will be replaced with the current username. The query will be run as a prepared statement, so the query also needs to accept parameters by using the <tt>?</tt>	String	.insertQuery1	Behavior

Insert Query 2	This optional insert query inserts the mapping for a new note into a mapping table. <tt>%d</tt> will be replaced with the ID of the new note. To disable this behavior, simply set this property to a blank string.	String	.insertQuery2	Behavior
Maximum Attachment Size	The maximum attachment size in bytes that will be accepted.	long	.maxAttachmentSize	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
Note Color	The background color for notes	Color	.noteColor	Appearance
Padding	The amount of padding between the notes.	int	.padding	Appearance
Skip Audit	If true, update queries originating from this component will skip the audit system. Can be important when attachments are turned on.	boolean	.skipAudit	Behavior
Sticky Header Color	The background color of the header for sticky notes	Color	.stickyHeaderColor	Appearance
Sticky Note Color	The background color for stick notes	Color	.stickyNoteColor	Appearance
Sticky Text	The word(s) used for the "Sticky" checkbox.	String	.stickyText	Appearance
Touchscreen Mode	Controls when this input component responds if touchscreen mode is enabled.	int	.touchscreenMode	Behavior
Unstick Query	This query is used for changing a note's status to be not sticky. <tt>%d</tt> is replaced with the note's ID	String	.unstickQuery	Behavior
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 scripting functions associated with it.

Event Handlers

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.

Property	Description
source	The component that fired this event
button	The code for the button that coused 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.

Property	Description
source	The component that fired this event
button	The code for the button that coused 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.

Property	Description
source	The component that fired this event
button	The code for the button that coused 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.

Property	Description
source	The component that fired this event
button	The code for the button that coused 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.

Property	Description
source	The component that fired this event
button	The code for the button that coused 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.

Property	Description
source	The component that fired this event
button	The code for the button that coused 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.

Property	Description
source	The component that fired this event
button	The code for the button that coused 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.

Property	Description
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. Note that not all components include an accurate oldValue in their events.
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
This component does not have any custom properties.

Examples

There are no examples available for this component.