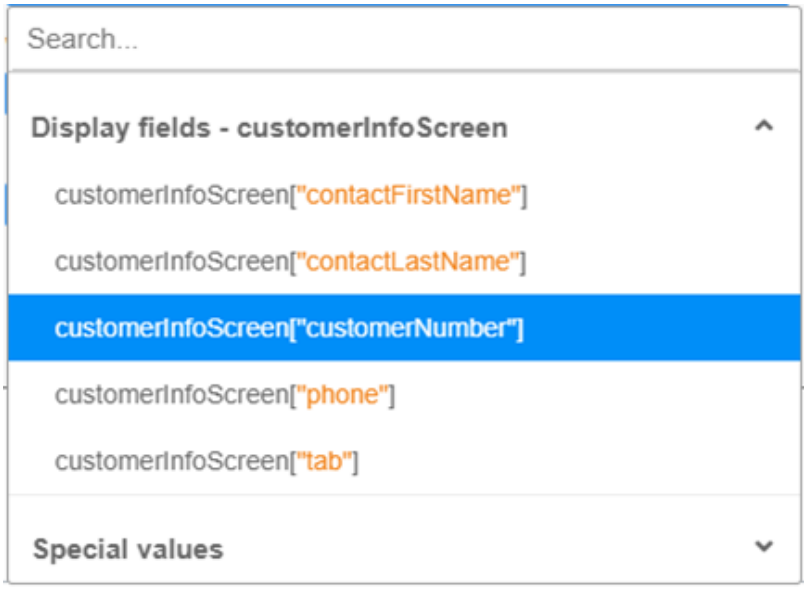
Working with Data in Routines

Suggestion Sources

To help you create business logic routines, Profound.js discovers as much as possible about your environment. This is why it helps to have your screen designed and database connections configured in advance of setting up your business logic. It allows Profound.js to discover the appropriate information and then reliably suggest the possible answers to the questions asked by the low-code plugins.

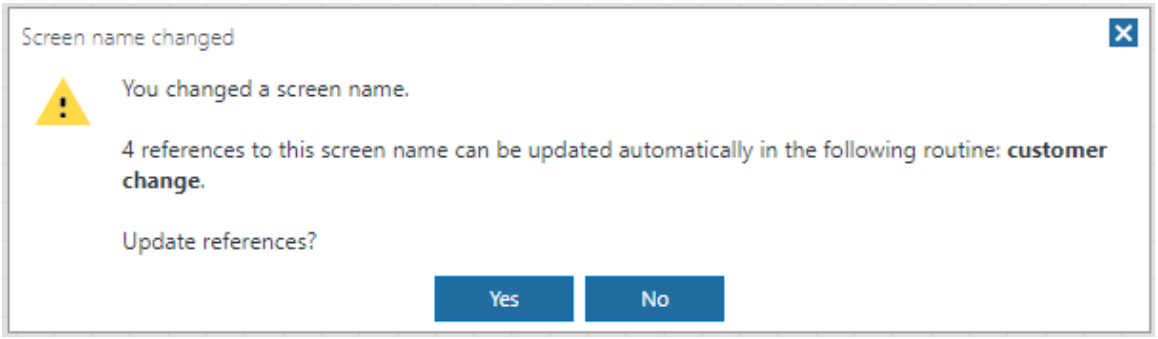
For example, as you design your screens, you may be adding fields from the database repository or binding custom fields to certain widget properties. Profound.js keeps track of this information and suggests it when appropriate.



Profound.js uses dozens of suggestion sources to ensure that you don't mistype table names, column names, screen names, field names, routine names, file names, widget property names, CSS class names, and other references as you are building your business logic.

Automatic Reference Updates

When the source of information changes, Profound.js will search your routines for any references to the original information and show the appropriate warning. In many cases, it will be able to adjust the references in your routines automatically.



Work Variables and Global Properties

Sometimes, the actions in your routines may need to store, calculate, or pass additional data that is not a direct reference to your screen or the database. This is where work variables and global properties are useful. Many plugins will allow you to store the results of an operation in a work variable or a global property. If the information is temporary in nature and only needs to be accessed within the immediate routine, use a work variable. If the information you're retrieving may have to be accessed by other routines, use a global property.

Once a work variable or a global property is captured, it becomes part of the suggestion list when using other plugins. Global properties are formatted as follows by Profound.js: **globals["property name"]**.

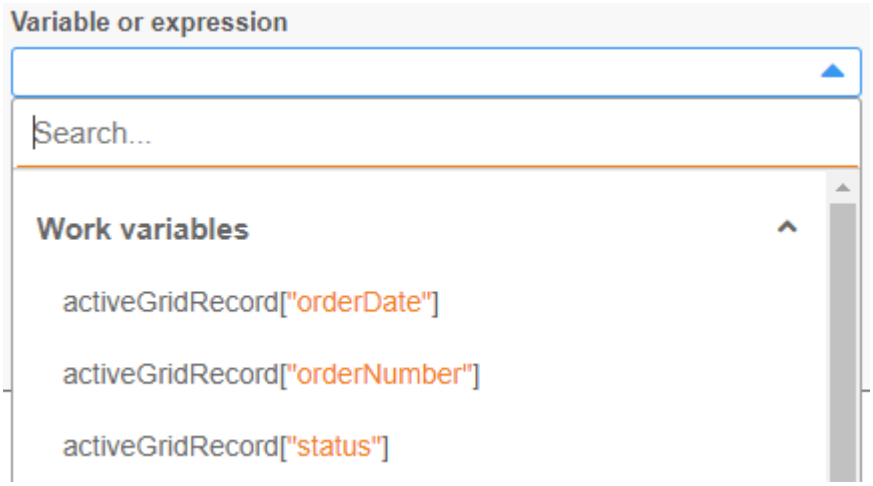
Classifying Variables and Properties

Each work variable and each global property is automatically classified as a simple value, a record, or a list of records. This allows Profound.js to suggest these within the right context. For example, if you're inserting new data into a database table from a work variable, Profound.js will only suggest variables that are classified as records. However, if you choose to specify each column value individually, Profound.js will only suggest simple values.

A variable or property that is classified as a record may have additional information associated with it, such as column data if that record was originally retrieved from a known database table. Profound.js will track this type of information and add it to its list of suggestions.

Implicit Work Variables

While some variables are created and named explicitly by you, there are certain variables that are created implicitly by Profound.js to make it more convenient for you to build the desired business logic. For example, when you create a routine that is attached to a Widget that belongs to a grid, an implicit record variable named *activeGridRecord* is created.



Session Properties

We know that work variables are accessed within the current routine, while global properties can be accessed from any routine within your Rich Display file. If you have multiple Rich Display files and you have the need to share information between them, this is where session properties can be useful.

There are many plugins that allow you to save information into the session, so that subsequent Node.js programs or Rich Display modules can retrieve and use that information.