Component Palette

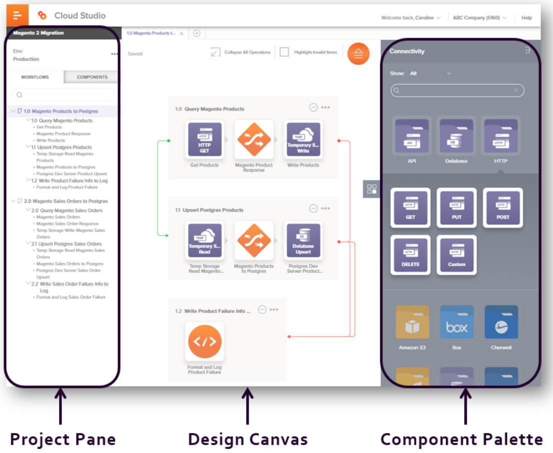
Introduction

The component palette is the collapsible panel on the right side of the project designer and the script editor in Cloud Studio. This palette contains multiple tabs that are relevant to each of these screens. As you work on projects, Cloud Studio remembers the display states that a given user was last using for a given project, including which tab was last active.

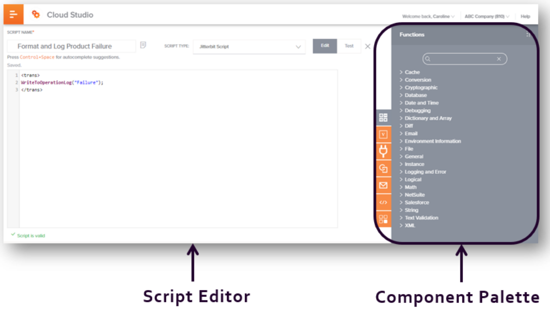
Accessing the Component Palette

There are two types of component palettes: the design component palette and the script component palette:

- **Design Component Palette:** To access the design component palette, first open a project. In an open project, the design component palette is displayed on the right of the project designer.



- **Script Component Palette:** To access the script component palette, open a script or transformation in script mode. The script component palette is displayed on the right of the script editor:



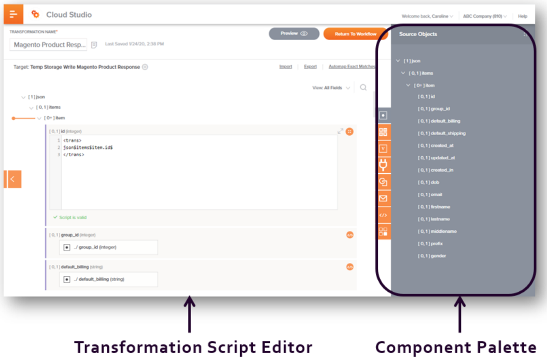
SCRIPT NAME* Format and Log Product Failure

SCRIPT TYPE: Jitterbit Script

Press Control+Space for autocomplete suggestions.

Saved.

```
<trans>
WriteToOperationLog("Failure");
</trans>
```



TRANSFORMATION NAME* Magento Product Resp...

Last Saved 1/24/20, 2:38 PM

Target: Temp Storage Write Magento Product Response

```
<trans>
json$items$item.id$
</trans>
```

✓ Script is valid

On This Page

- Introduction
- Accessing the Component Palette
- Design Component Palette
- Script Component Palette

Related Articles

- Design Canvas
- Operation Creation and Configuration
- Project Creation and Configuration
- Script Types and Creation
- Transformation Creation and Configuration

Related Topics

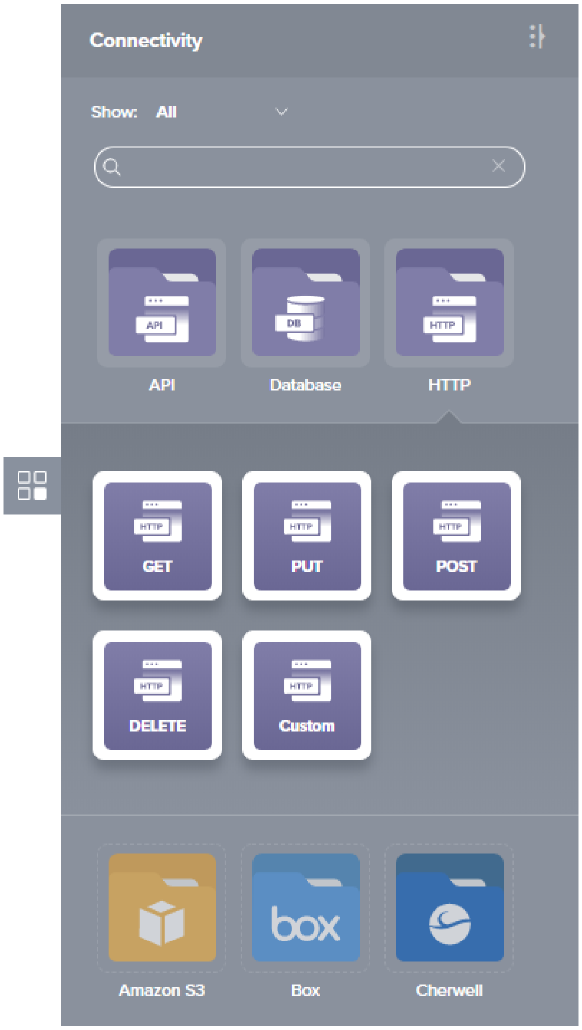
- Cloud Studio
- Cloud Studio User Interface
- Connector Builder
- Connector SDK
- Connectors
- Introduction
- Project Components
- Scripts
- Transformations

Last updated: Jan 24, 2020

Design Component Palette

The design component palette provides access to project components that can be used on the design canvas.

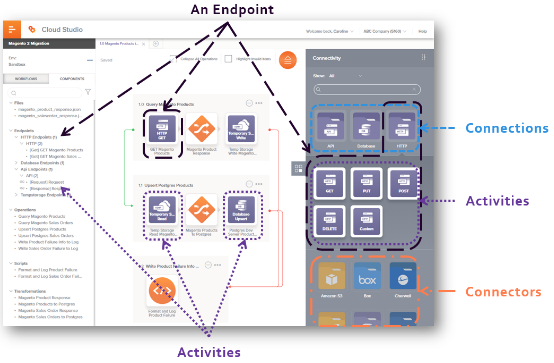
The **Connectivity** tab in the design component palette is open by default:



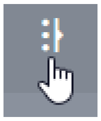
Within this tab, connectors are first configured to create connections. Activities associated with those connections can then be added to operations on the design canvas and configured as sources or targets. An endpoint refers to a specific connection and its activities.

- **Connector:** Connectors provide the interface for entering user-provided input such as credentials to create an authenticated connection. The **Connectors** filter shows the types of connectors that can be configured. In addition, you can create custom connectors using Connector Builder or the Connector SDK.
- **Connection:** A connection provides authenticated access to a data resource that has been configured using a connector. The **Endpoints** filter shows these configured connections. Existing connections can be edited by double-clicking on the connection in the palette. Existing connections can be renamed by clicking on the connection name in the palette.
- **Activity:** An activity is an interaction with a connection that can be configured with user-provided input such as data structures that represent the "request" and "response" schemas for that action. The **Endpoints** filter shows the configured connections, which can be clicked to reveal the types of activities that can be added to an operation. Those activities are then able to be dragged to operations on the design canvas, where they can be configured by double-clicking on the activity within the operation. Configured activities can act as sources (providing data within an operation) or targets (receiving data within an operation).
- **Endpoint:** An endpoint refers to a specific connection and its activities, which are added to an operation and then configured as sources or targets within the operation.

For reference, this diagram shows the relationship between connectors, connections, activities, and endpoints:



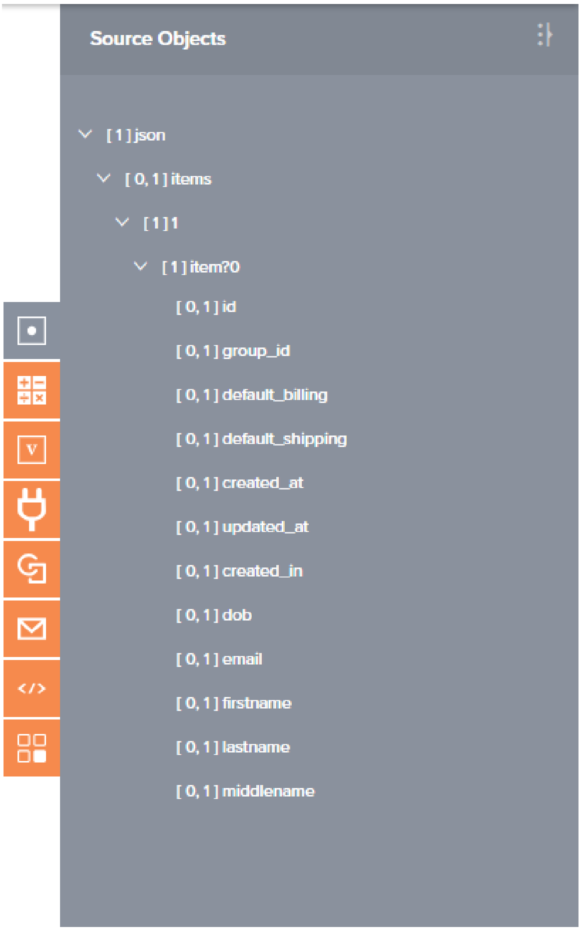
To collapse the tab to provide for more screen real estate on the design canvas, click the collapse icon in the top right:



Refer to these sections for detailed information about configuring connectivity resources:

Script Component Palette

The script component palette provides access to source objects, functions, variables, plugins, operations, notifications, other scripts, and endpoints that can be used within a script:



Source Objects

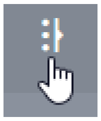
[1] json
[0, 1] items
[1] 1
[1] item?0
[0, 1] id
[0, 1] group_id
[0, 1] default_billing
[0, 1] default_shipping
[0, 1] created_at
[0, 1] updated_at
[0, 1] created_in
[0, 1] dob
[0, 1] email
[0, 1] firstname
[0, 1] lastname
[0, 1] middlename

Click on any component tab (**Functions**, **Variables**, **Plugins**, **Operations**, **Notifications**, **Scripts**, or **Endpoints**) to display information specific to the active tab:

	Source Objects: This tab is present only for Jitterbit Scripts created within a transformation. Within the script, you can reference source data by the path of the field. Add a source object to the Jitterbit Script using one of these methods: • Drag the object from the palette to the script. The full reference path to the source object using the appropriate syntax will be inserted. • Manually enter the full reference path to the source object. For an explanation of the node/field notation, refer to Nodes and Fields.
	Functions: This tab provides a list of functions, in the appropriate language, available to use in a script. Add a function to the Jitterbit Script or JavaScript using one of these methods: • Drag the function from the palette to the script. The appropriate syntax for the script language will be inserted. • Begin typing the function name and then press `Control+Space` to display a list of autocomplete suggestions. Select a function to insert the appropriate syntax for the script language. • Manually enter the appropriate syntax for the script language. For additional documentation, see Functions.
	Variables: This tab provides access to variables that are available to use globally throughout a project, including global variables, project variables, and Jitterbit variables. Add a variable to the Jitterbit Script or JavaScript using one of these methods: • Drag the variable from the palette to the script. The appropriate syntax for the script language will be inserted. • Begin typing the variable name and then press `Control+Space` to display a list of autocomplete suggestions. Select a variable to insert the appropriate syntax for the script language. • Manually enter the appropriate syntax for the script language. For additional documentation, see Variables. ⓘ **NOTE:** Local variables are not listed because they are not available globally; however you can still use them locally within a script.
	Plugins: This tab provides a list of plugins that can be run inside the script. Add a plugin to the Jitterbit Script using one of these methods: • Drag the plugin from the palette to the script. Both the `RunPlugin()` function and the plugin reference will be inserted into the script. • Begin typing the plugin name and then press `Control+Space` to display a list of autocomplete suggestions. Select a plugin to insert the plugin reference into the script. • Manually enter the plugin reference. ✓ **TIP:** Only plugins that are available to use within a script will be listed. Additional plugins can be applied on an activity within an operation. For documentation on using plugins, including how to configure a plugin at the activity level and set global variables, see Plugins.
	Operations: This tab provides a list of operations from the project that are available to use in the script. Add an operation to the Jitterbit Script using one of these methods: • Drag the operation from the palette to the script. Both the `RunOperation()` function and the operation reference will be inserted into the script. • Begin typing the operation name and then press `Control+Space` to display a list of autocomplete suggestions. Select an operation to insert the operation reference into the script. • Manually enter the operation reference. For additional documentation, see Operations.

✉	**Notifications:** This tab provides a list of notifications from the project that are available to use in the script. Add a notification to the Jitterbit Script using one of these methods: • Drag the notification from the palette to the script. Both the `SendEmailMessage()` function and the notification reference will be inserted into the script. • Begin typing the notification name and then press `Control+Space` to display a list of autocomplete suggestions. Select a notification to insert the notification reference into the script. • Manually enter the notification reference. For additional documentation, see Notifications.
</>	**Scripts:** This tab provides a list of all other scripts that were created as project components, including both Jitterbit Scripts and JavaScripts, that are available to use in the script. Add another script within the Jitterbit Script using one of these methods: • Drag the script from the palette to the script. Both the `RunScript()` function and the script reference will be inserted into the script. • Begin typing the script name and then press `Control+Space` to display a list of autocomplete suggestions. Select a script to insert the script reference into the script. • Manually enter the script reference. For additional documentation, see Scripts. ⚠ **CAUTION:** While Jitterbit Scripts can call JavaScripts, the reverse is not true at this time. Jitterbit JavaScripts cannot call other scripts of any language.
▦	**Endpoints:** This tab provides a list of endpoints from the project that are available to use in the script. Add a connection or activity to the Jitterbit Script or JavaScript using one of these methods: • Drag the connection or activity from the palette to the script. The connection or activity reference will be inserted into the script. • Begin typing the connection or activity name and then press `Control+Space` to display a list of autocomplete suggestions. Select a connection or activity to insert the appropriate reference into the script. • Manually enter the connection or activity reference. Depending on the endpoint, you may then use the **Functions** tab to add functions for which to use the connection or activity reference as the argument. For additional documentation, see Connectors.

To collapse the tab to provide for more screen real estate while editing scripts, click the collapse icon in the top right:



For more detailed information about configuring scripts, see Scripts.