JOLT ADVANTAGE GROUP

# Reusable JSON Config

## Component description

Alex Ruiz (mruiz@joltag.com)
Julio Velázquez (jvelazques@joltag.com)
Luis Ordoñez (lordonez@joltag.com)

Phone: 855-238-5658 | 8875 Hidden River Parkway | Ste.300 | Tampa, FL 33637 | JoltAG.com

# Dependencies.

- UiPath.Web.Activities
- .Json File or Orchestrator Json String Asset.

# Description.

Component to read a JSON string and make it an JObject for use in the future sequences of the project, in this configuration file you could save anything you need multiple times in the project, you can separate by application, window or anything the LD/SA decides by using Nodes.

JObject will save the complete configuration file with multiple nodes.
JToken will save only the node needed.

## Config File

Default JSON Config File separated by their node:

- Global

```
  "Global": {
    "Orchestrator_Queue": "Orchestrator Queue Name",
    "Timeout_Short": "5000",
    "Timeout_Long": "60000",
    "Process_Name": "Name of the process",
    "Screenshots_Folder": "Path_To_Folder",
    "Username": "{Username}",
    "Login_Username": "UsernameAPP",
    "Login_Password": "passwordAPP_NOPROTECTION",
    "LogMessage_Start": "Message for logging at the start",
    "LogMessage_Finish": "Message for logging at the end"
  },
```

- Reporter (Used for the Exception Email Re-Usable Component).

```
  "Reporter": {
    "To": "support@joltag.com",
    "Subject": "subject",
    "Port": PortInt32 or “StringNumber”,
    "Server": "smtp.office365.com",
    "Asset_Outlook_Credentials": "OrchestratorAssetCredentials"
  },
```

- Assets

```
  "Assets": {
    "AssetNameUiPath": "AssetNameInOrchestrator",
    "AssetNameUiPath1": "AssetNameInOrchestrator1"
  }
```

### Nodes

The JSON can have one, or multiple nodes that are separated as required for the developer needs, in this example the only 3 nodes are: Global, Reporter and Assets, previously shown.

In case of needing one more you just need to close the last node and add a coma (,) at the end and Open a new one, like so:

```
  }, <- Closing previous node and starting a new one.
  "NewNode": {
    "NewKey":"NewValue"
  }
```

The node “NewNode” is created.

### Keys

You can add as many as keys as you want in their node.
To add a new key you just need a coma (,) and create a new key, the key is the way you will call the value in UiPath.

```
  "NewNode": {
    "NewKey":"NewValue",
    "NewKey2":"NewValue2"
  }
```

The key “NewKey2” is added.

### Values

The values of the keys can be either strings, int32, booleans & arrays.

```
  "DataTypes": {
    "Number": 100,
    "Name":"JoltAG",
    "Boolean":true,
    "Array":["Alex","Julio","Luis"],
    "NullValues":null
  }
```

# Arguments.

## Input.

- JSON file or json string stored as an asset in orchestrator.

## Output.

- JObject variable.

# Step by step description.

**CJ 1.0 -** This is the first step, when invoking the component, it starts taking an orchestrator asset to retrieve a string that has the json configuration.

> **CJ1.0 Note:** This part can be either an asset or a read text file for the JSON text.

**CJ 2.0 -** This multiple assign is for when you need variables of the environment each start of the run or information that you can simply add with a .Replace in the json STRING variable.

**CJ 3.0 -** Deserializes the JSON string to a JObject.

# Notes.

When calling anything from the JObject in UiPath you can either call the whole Jobject or use a Jtoken to use only the node needed for that part of the process using an argument in a new sequence.

- **JObject**
  Config being the JObject variable.
  `Config("Global")("Process_Name").ToString`
  UiPath outputs from the Global Node, the Process Name key value.
- **JToken**
  Config being the JObject variable.
  `Reporter = Config("Reporter").ToString`
  UiPath saves ONLY the node "Reporter" in a Jtoken variable called "Reporter".

- JObject will have all the nodes and keys in the json file.
- Jtoken will only have 1 node with all of the keys specific to that node.

When using the Jtoken inside of a sequence then you will just need to use:
`Reporter("To").ToString`
To retrieve the value from the "To" key in the Jtoken created with only the "Reporter" node.

# Exceptions.

- **UiPath.Core.Activities.OrchestratorHttpException:** Could not find an asset with this name
- **Newtonsoft.Json.JsonReaderException:** Wrong json syntax