Chandru k

Kchandu05@gmail.com

Abstract

This document is user manual for detailed understating of Task Management and Status Reporting

task management and Status Reporting

Project management using UiPath

# Framework Component functions

Table 1 shows the calling structure of the framework. That is, which workflows are called, the order in which they are called, and the state of the main state machine where you can find the workflow invoke.

|  |  |  |
| --- | --- | --- |
| Table 1 – Component Call tree structure | | |
| Component file names and locations | | State where it is called |
|  |  |  |
| Main.xaml |  |  |
|  | Framework\InitAllSettings.xaml | Read Config |
|  | DataLoad.xaml | Send Task |
|  | ExtractReqData.xaml | Send Task, Status Report |
|  | AdhocRequirement.xaml | Send Task |
|  | StatusReportData.xaml | Status Report |

# Global Variables

The Global Variables are those variables whose scope is the main program, or main workflow. They can be found in the main.xaml workflow file, by first clicking anywhere inside the main state machine and then clicking the variables pane. Table 2 is a list of the project’s global variables.

These are used to store information that will be available throughout the runtime of the process. It is important to understand where each variable is written and where it is read.

There red cell background represents workflows in which the variable is written and the green cell background workflow in which it is read.

|  |  |  |  |
| --- | --- | --- | --- |
| Table 2 – Global Variable list | | | |
| Name | Data Type | Is written in workflow | Is read in workflow |
| Config | Dictionary(String,Object) | InitAllSettings.xaml | DataLoad.xaml,AdhocRequirement.xaml,ExtractReqdata.xaml, StatusReportData.xaml |
| Trigger | String | Main.xaml |  |

# Read Config State

## InitAllSettings.xaml Workflow

This workflow outputs a settings Dictionary with key/value pairs to be used in the project. Settings are read from local config file.

|  |  |  |
| --- | --- | --- |
| Table 3 – InitAllSettings.xaml Arguments and values | | |
| DataType and Name | Argument Type | Values |
| String: in\_ConfigFile | Input | “Data\Config.xlsx” |
| String[]: in\_ConfigSheets | Input | {“Settings”,”Constants”} |
| Dictionary(x:String,x:Object): out\_Config | Output | config |

# Send Task State

## DataLoad.xaml Workflow

This workflow loads the requirement data from source and send task list to developer, Tester, Client/BA and Manager.

|  |  |  |
| --- | --- | --- |
| Table 4 – DataLoad.xaml Arguments and values | | |
| DataType and Name | Argument Type | Values |
| String: in\_dataFile | Input | config(“FilePath”).ToString |
| Dictionary(x:String,x:Object): config | Input | config |

## ExtractReqData.xaml Workflow

This workflow extracts the needed data columns from requirement sheet and store it in text file based on input data table

|  |  |  |
| --- | --- | --- |
| Table 5 – ExtractReqData.xaml Arguments and values | | |
| DataType and Name | Argument Type | Values |
| String: colList | Input | config("ColConditionforDEV").ToString |
| String: tableData | Input | config("tableData").ToString |
| DataTable: in\_filteredReq1 | Input | filteredReq1 |
| String: out\_tableData | Output | outputData |

## AdhocRequirement.xaml Workflow

This workflow checks for any adhoc requirement pdf file placed in input folder. If it finds any, data in pdf is extracted and placed in Adhocrequirement excel in output folder and send details to manager and Project Team

|  |  |  |
| --- | --- | --- |
| Table 6 – AdhocRequirement.xaml Arguments and values | | |
| DataType and Name | Argument Type | Values |
| Dictionary(x:String,x:Object): config | Input | config |

# Status Report State

## StatusReportData.xaml Workflow

This workflow use the extracted status report source data and use the data to send email to client.

|  |  |  |
| --- | --- | --- |
| Table 7 – AdhocRequirement.xaml Arguments and values | | |
| DataType and Name | Argument Type | Values |
| DataTable: in\_statusreportData | Input | statusreportData |
| Dictionary(x:String,x:Object): in\_config | Input | config |