FlowTasks – Developer Guide

Contents

[Intro 1](#_Toc325468765)

[Requirements 1](#_Toc325468766)

[Get Started 1](#_Toc325468767)

[Parameters 3](#_Toc325468768)

[Connectors 4](#_Toc325468769)

[Task list 4](#_Toc325468770)

[Get Involved 5](#_Toc325468771)

# Intro

FlowTasks is a framework for building human or business processes. These types of process or **workflow** can be defined like a collection of activities or **tasks** that some user has to do. A user in this case can be a real person or any automated system.

FlowTasks make easier to develop this type of application.

# Requirements

In order to use FlowTasks you need:

Windows workflow foundation 4.5

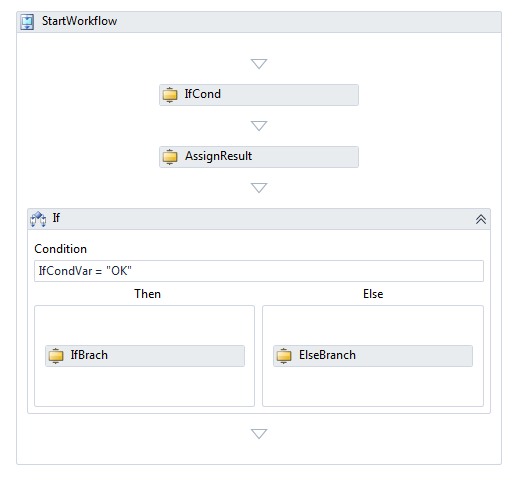
AppFabric

SQL server 2008 or higher

# Get Started

The best way to start using FlowTasks is to have a look at the examples that you can find in Flow.Workflows.Test project. Let’s have a look at SampleWf4.xamlx.

If you want to create a workflow managed by FlowTasks you need to use first of all the StartWorkflow activity. This is a custom activity; all these activities are in the Flow.Tasks.Workflow project. Inside this activity you can define any type of workflow.



If you want the workflow to wait for some user intervention you need to use the ApproveTask activity. IfCond is an activity of this type.

Code snippet: ApproveTask

Implementation = () => new ApproveTask

{

AssignedToUsers = "{r.Dev}",

CorrelationId = CorrelationId,

DefaultResult = "Activity Expired",

Description = "This is desc for my approve task",

DisplayName = "Approve my task",

TaskCode = TaskCode,

Title = "This is the title for my approve",

UiCode = "ApproveTask",

ExpiresIn = "10d",

OnInit = new ActivityFunc<TaskStatus, TaskStatus>

{

Argument = \_onInit,

Handler = new CreateOnClientInit

{

DisplayName = "CreateOnClientInit",

Request = \_onInit

}

},

OnComplete = new ActivityFunc<TaskStatus, TaskStatus>

{

Argument = \_onComplete,

Handler = new CreateOnClientComplete

{

DisplayName = "CreateOnClientComplete",

Request = \_onComplete

}

}

};

## **Parameters**

AssignedToUsers: a comma separated list of users or roles. In this example {r.Dev} means assign this activity to all users that belong to Dev role. FlowTasks has an authentication library called Flow.Users, but you could use your own.

CorrelationId: this is an Id used to distinguish an activity in a parallel workflow.

DefaultResult: this is the result the activity sends to the next task when the task is completed and the user doesn’t specify a different message.

Description: this is the description of the task the user will see.

Title: this is the title of the task the user will see.

TaskCode: this is the code given to the activity. It will appear to the user. You can sort or group tasks based on this code.

UiCode: this tells the system what type of control to display to the user.

ExpiresIn: this specifies when the task should expire. 10d means 10 days. You could say also 30m which means 30 minutes.

OnInit: this is the event that runs before the task is assigned to users.

OnComplete: this is event that runs after the task is completed.

# Connectors

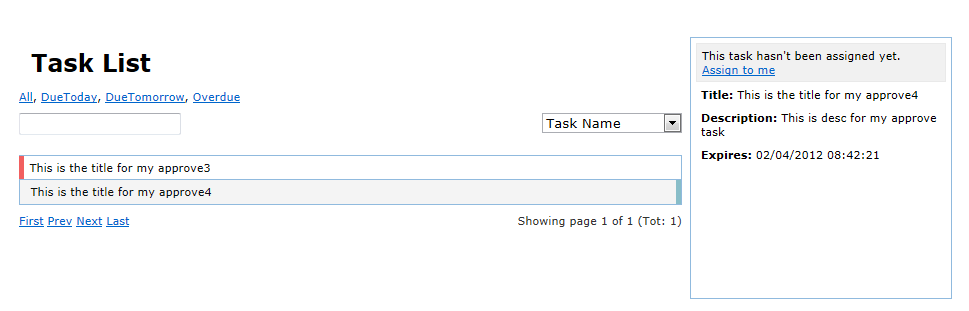
How do you start a workflow? To start a workflow you need some sort of event that triggers the workflow to begin its process. This event could be anything it depends on the workflow you are writing.

In Flow.Connectors you can find some examples. DocsOnFolder is a service that looks if someone drops a document in a folder, if that happen then it will store the document in Flow.Docs and then starts a document approval workflow.

Twitter is a service that checks if someone sends a tweet with a specific pattern, if that happen a workflow will start.

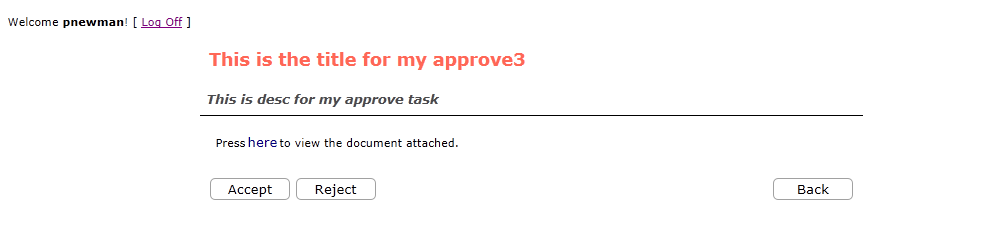
# Task list

But when an approval task has been created how do we complete the task? This is when the Task list comes into play. You can see an example of Task list in Flow.Task.Web.



The Task list has all the tasks that have been created and that are still not completed. Of course you’ll see only the tasks that are relevant to you, based on your login credentials and the task configuration (AssignedToUsers parameter).

When you then select on the task the user interface for that particular task is displayed (UiCode parameter).



In this simple example you can “Accept” or “Reject” the task, this will send the result back to the workflow which will then proceed with its execution.

# Get Involved

FlowTasks is an open source project; its development is still on the first stages. Any help would be much appreciated!

And please do let me know your comment at sydney.shown@gmail.com