FlowTasks – How to create a Task View

Contents

[Introduction 1](#_Toc330554510)

[Coding the View 2](#_Toc330554511)

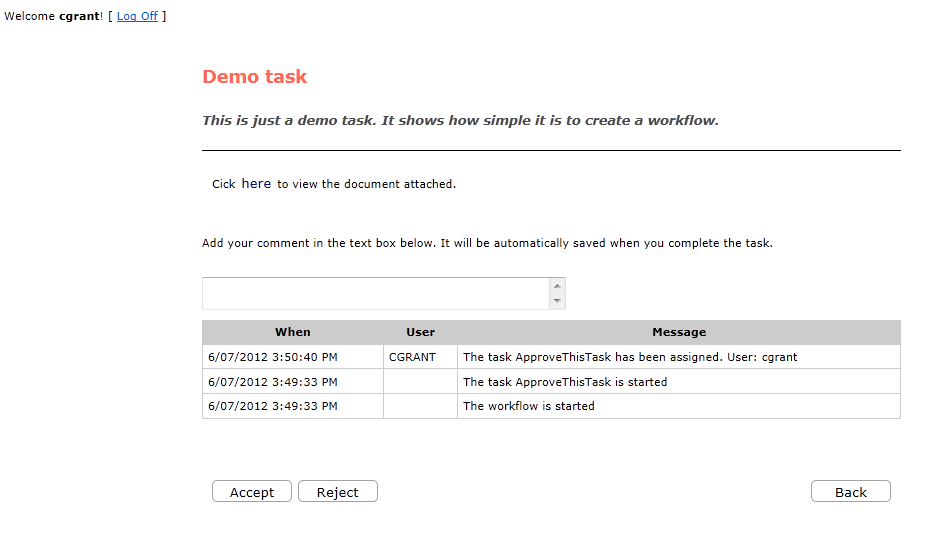
[Deploying the View 5](#_Toc330554512)

[Styling 5](#_Toc330554513)

# Introduction

After you created a workflow, using the steps described in the document FlowTasks How To Create a Workflow.docx, you can then work on the kind of views the user should see when completing the tasks.

Note that if you followed the above mentioned document this is the type of view the user saw when attempting to complete the task.

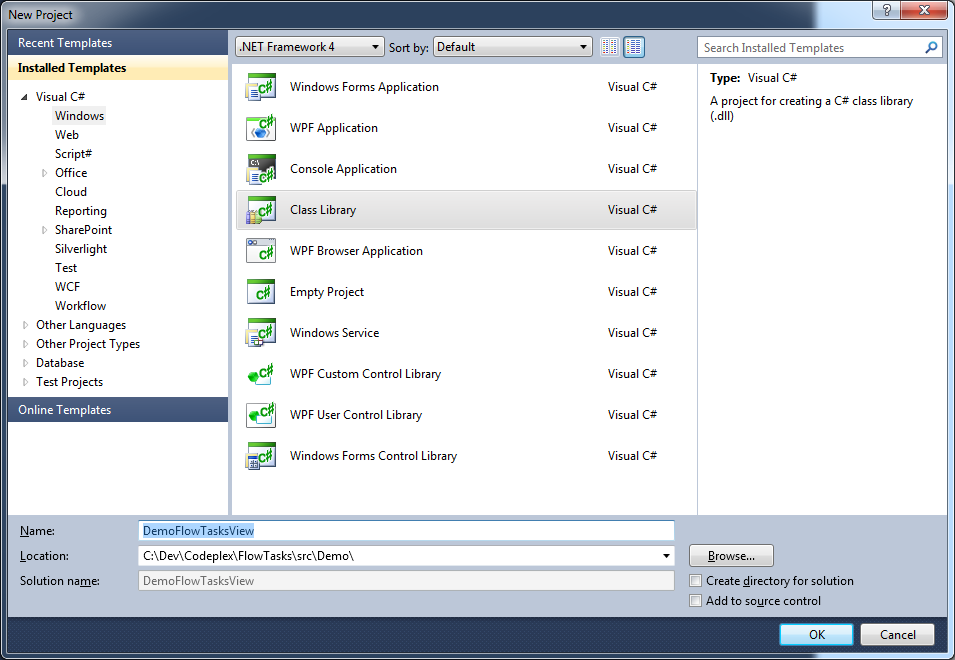


But what if you want to plug in your own user interface?

The code for the example can be found under Demo/DemoFlowTasksView folder.

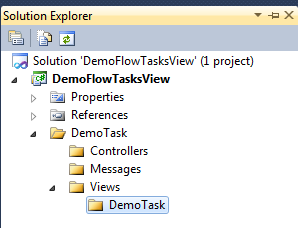
# Coding the View

1. Create a class library project.



Rename the project DemoFlowTasksWiew.

1. Create the view structure like this:



1. Add the following references

Flow.Docs.Contract

Flow.Users.Contract

Flow.Tasks.Contract

Flow.Tasks.View

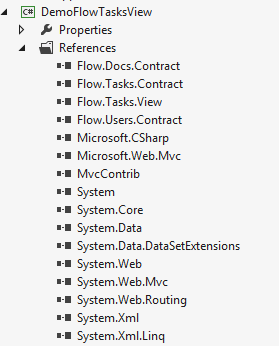
System.Web

System.Web.Mvc

System.Web.Routing

And then using NuGet add MvcContrib.

You should have all these libraries now:



1. Add the following files:

DemoTask\Controllers\DemoTaskController.cs

using System;

using System.Web.Mvc;

using Flow.Tasks.View;

using Flow.Tasks.Contract;

using Flow.Tasks.View.Models;

using Flow.Tasks.Contract.Messages;

using Flow.Tasks.Contract.Interface;

using Flow.Users.Contract;

using Flow.Docs.Contract;

namespace DemoFlowTasksView.DemoTask.Controllers

{

public class DemoTaskController : BaseController

{

public DemoTaskController(IFlowUsersService usersService,

IFlowTasksService tasksService, IFlowDocsDocument document) :

base(usersService, tasksService, document) { }

[HttpGet]

public ActionResult Index()

{

return View();

}

[HttpPost]

public ActionResult Index(FormCollection values, TaskModel task)

{

var request = new AssignTaskToRequest

{

User = HttpContext.User.Identity.Name,

TaskOid = Guid.Parse(values["TaskOid"])

};

\_tasksService.AssignTaskTo(request);

InitDocumentControl(task);

InitCommentsControl(task);

return View(task);

}

}

}

DemoTask\Messages\RegistrationMessage.cs

using MvcContrib.PortableAreas;

namespace DemoFlowTasksView.DemoTask.Messages

{

public class RegistrationMessage : IEventMessage

{

public RegistrationMessage(string message)

{

\_message = message;

}

private readonly string \_message;

public override string ToString()

{

return \_message;

}

}

}

DemoTask\DemoTaskRegistration.cs

using MvcContrib.PortableAreas;

using System.Web.Mvc;

using DemoFlowTasksView.DemoTask.Messages;

namespace DemoFlowTasksView

{

public class DemoTaskRegistration : PortableAreaRegistration

{

public override void RegisterArea(AreaRegistrationContext context, IApplicationBus bus)

{

bus.Send(new RegistrationMessage("Registering Demo Task Portable Area"));

base.RegisterArea(context, bus);

context.MapRoute(

"DemoTask",

"DemoTask/{controller}/{action}",

new { controller = "DemoTask", action = "index" });

}

public override string AreaName

{

get { return "DemoTask"; }

}

}

}

DemoTask\Views\DemoTask\index.cshtml

@model Flow.Tasks.View.Models.TaskModel

@{

ViewBag.Title = "Demo Task View";

Html.Assets().Styles.Add("/Content/demotask.css");

}

@Html.Partial("\_HeaderPartial")

<div class="demoTask">

This is the demo View. Here you can add everything you need for the user.

<div>

@Html.Partial("\_CommandsPartial")

# Deploying the View

1. To deploy the new view just copy *DemoFlowTasksView.dll* under the bin folder of the TaskList application, see the document *FlowTasks Installation Guide.docx* for more information about the TaskList.
2. Then you have to tell the workflow that your task has to open the new view.

We can use the workflow you created with *FlowTasks How To Create a Workflow.docx* but change the line of code highlighted below.

public DemoTask()

{

Implementation = () => new ApproveTask

{

AssignedToUsers = "{r.Dev}",

CorrelationId = CorrelationId,

DefaultResult = "Activity Expired",

Description = "This is just a demo task. It shows how simple it is to create a workflow.",

DisplayName = "Demo task",

TaskCode = TaskCode,

Title = "Demo task",

**UiCode = "DemoTask"**,

ExpiresIn = "10d",

OnInit = new ActivityFunc<TaskStatus, TaskStatus>

{

Argument = \_onInit,

Handler = new CreateOnClientInit

{

DisplayName = "CreateOnClientInit",

Request = \_onInit

}

},

...

This is it! Start the workflow like you did in *FlowTasks How To Create a Workflow.docx* and now you should see your new view.

# Styling

You can have your own set to CSS files for all your views. You may have noticed this line of code in the file index.cshtml

Html.Assets().Styles.Add("/Content/demotask.css");

Now if you create a css file called *demotask.css* and add this content:

.demoTask {

font-size:1.5em;

color: blue;

padding:10px;

}

Then copy the file under the Content folder of the TaskList application and open the task, this is what you should see.

