Flowtime Comments

Last Modified on 04/24/2017 4:39 am EDT

Get Comments

```
var comments = executor.CommentsView;
```

Add Comments

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using PNMsoft.Sequence;
using PNMsoft.Sequence.Runtime;
using PNMsoft.Sequence.Security;
using PNMsoft.Sequence.SocialFeatures;
using PNMsoft.Sequence.Messaging;
using System.Security.Principal;
using PNMsoft.Sequence.Obs;
namespace SequenceEx.Social
{
public class SocialBPM
{
public void AddCommentToWorkflow(int iWFID, string subject, string body, bool isPrivate)
{
AppDomain.CurrentDomain.SetPrincipalPolicy(PrincipalPolicy.WindowsPrincipal);
//best pracice is to use an existing engine using WorkflowRuntime.Engine (existing one)
using (IWorkflowEngine engine = new WorkflowRuntime().Run(typeof(WorkflowEngine)))
{
AuthenticatedUser user = engine.GetService().Authenticate();
using (SecurityManager.Impersonate(user))
{
IWorkflowExecutionService executionService = engine.GetServiceWithCheck();
WorkflowInstance wfI = executionService.GetWorkflowInstance(iWFID);
SocialFeaturesExecutor executor = new SocialFeaturesExecutor(wfI);
AccessibilityLevel aL;
if (isPrivate)
{
aL = AccessibilityLevel.Private;
}
else
{
aL = AccessibilityLevel.Public;
}
executor.AddComment(subject, body, aL);
}
}
}
public void AddQuestionToWorkflow(int iWFID, string subject, string body, string recipientUsername
, bool isPrivate)
{
AppDomain.CurrentDomain.SetPrincipalPolicy(PrincipalPolicy.WindowsPrincipal);
//best pracice is to use an existing engine using WorkflowRuntime.Engine (existing one)
using (IWorkflowEngine engine = new WorkflowRuntime().Run(typeof(WorkflowEngine)))
{
AuthenticatedUser user = engine.GetService().Authenticate();
using (SecurityManager.Impersonate(user))
{
IWorkflowExecutionService executionService = engine.GetServiceWithCheck();
WorkflowInstance wfI = executionService.GetWorkflowInstance(iWFID);
SocialFeaturesExecutor executor = new SocialFeaturesExecutor(wfI);
```

```
List toRecipients = new List();
UserRecipient r = new UserRecipient();
IObsService iOBS = engine.GetServiceWithCheck();
IEnumerable qUsers = iOBS.GetUsersByUserName(recipientUsername);
foreach (User u in qUsers)
{
if (u.UserId > 0)
{
r.UserId = u.UserId;
}
}
toRecipients.Add(r);
AccessibilityLevel aL;
if (isPrivate)
{
aL = AccessibilityLevel.Private;
}
else
{
aL = AccessibilityLevel.Public;
}
executor.AddQuestion(subject, body, toRecipients, aL);
}
}
}
}
}
```

Remove Comments

```
executor.DeleteComment(1);
```