Fest 2018 Workshop

Building Applications for Relativity and RelativityOne

September 18, 2018 - Version 1.0

Contents

1 Course Overview

As a developer, you want a solid process for building and testing your apps, so you can focus on delivering value quickly. The Application Deployment System (ADS) is the standard way to build applications on the Relativity platform. Relativity Test Helpers is our new, open source testing framework.

In this advanced session, we'll review both of these standard approaches, covering common patterns for setting up effective ADS development solutions and integration tests. Topics will include: setting up your development environment using DevVM's; Visual Studio developer tools including templates, NuGet packages, and the "publish to Relativity" extension; and GitHub open source solutions such as Gravity, Test Helpers, and more. We'll also highlight ways to ensure your applications make a smooth transition from the on-premise world to RelativityOne. You will create tests and learn how to run those test from scripts as well as gain practical experience writing tests and potentially automating tests for your solutions, ensuring your customization will be ready run with the latest release of Relativity and RelativityOne.

1.1 What You'll Learn

- Relativity Templates
- Publish to Relativity
- Unit Tests
- Integration Tests
- Run Integration Tests as CI
- Gravity API

1.2 Required Development Software

This workshop and associated development resources require a development environment that includes the following items:

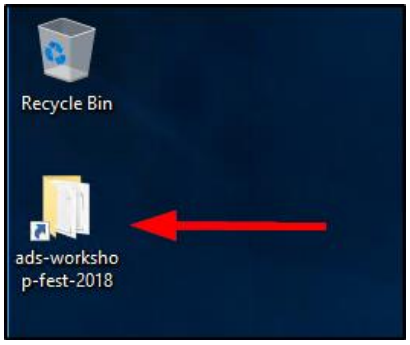
- Relativity 9.6.134.78 DevVM
- Visual Studio 2017
- Relativity Visual Studio Templates
- .NET Framework 4.6.2
- SQL Server 2016

We have provided the required software for you for this workshop.

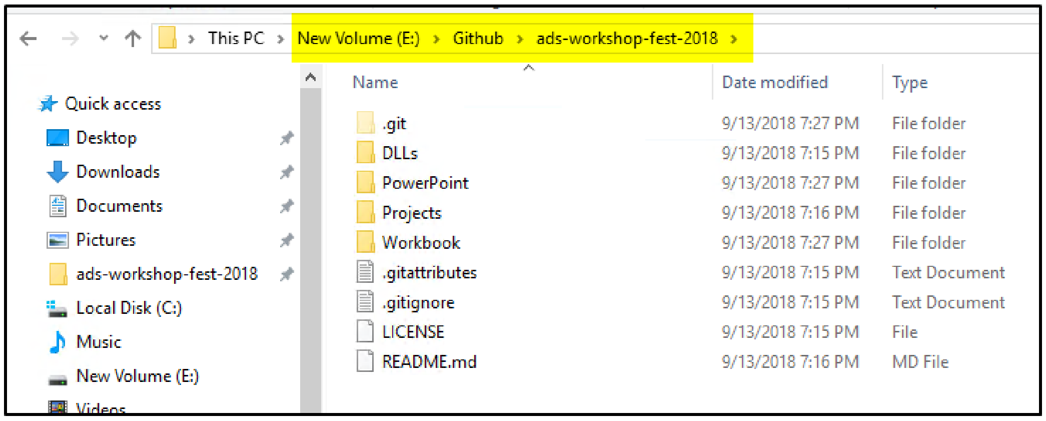
2 Getting Started

2.1 Workshop Projects

1. Make sure you have a folder shortcut on Desktop with name **ads-workshop-fest-2018**.



2. Double click on the **ads-workshop-fest-2018** folder to open it. Make sure you see the files as shown in the below screenshot.

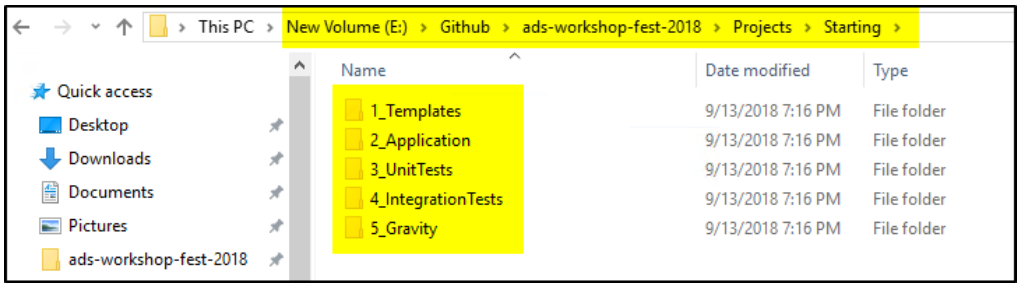


This PC › New Volume (E:) › Github › ads-workshop-fest-2018 ›

Quick access
Desktop
Downloads
Documents
Pictures
ads-workshop-fest-2018
Local Disk (C:)
Music
New Volume (E:)
Videos

Name	Date modified	Type
.git	9/13/2018 7:27 PM	File folder
DLLs	9/13/2018 7:15 PM	File folder
PowerPoint	9/13/2018 7:27 PM	File folder
Projects	9/13/2018 7:16 PM	File folder
Workbook	9/13/2018 7:27 PM	File folder
.gitattributes	9/13/2018 7:15 PM	Text Document
.gitignore	9/13/2018 7:15 PM	Text Document
LICENSE	9/13/2018 7:15 PM	File
README.md	9/13/2018 7:16 PM	MD File

3. Go to the **Projects/Starting** folder to make sure you see the folders as shown in the below screenshot. These folders contain Visual Studio projects for different sections in the workbook.



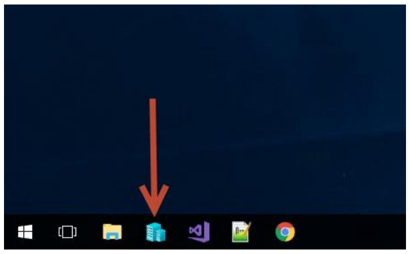
This PC › New Volume (E:) › Github › ads-workshop-fest-2018 › Projects › Starting ›

Quick access
Desktop
Downloads
Documents
Pictures
ads-workshop-fest-2018

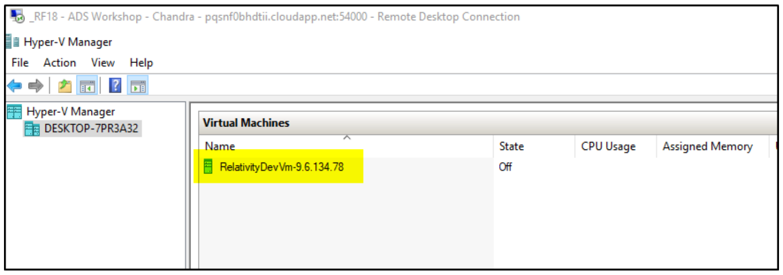
Name	Date modified	Type
1_Templates	9/13/2018 7:16 PM	File folder
2_Application	9/13/2018 7:16 PM	File folder
3_UnitTests	9/13/2018 7:16 PM	File folder
4_IntegrationTests	9/13/2018 7:16 PM	File folder
5_Gravity	9/13/2018 7:16 PM	File folder

2.2 DevVM Setup

1. Open the **Hyper-V Manager** application by clicking the blue icon in the taskbar.



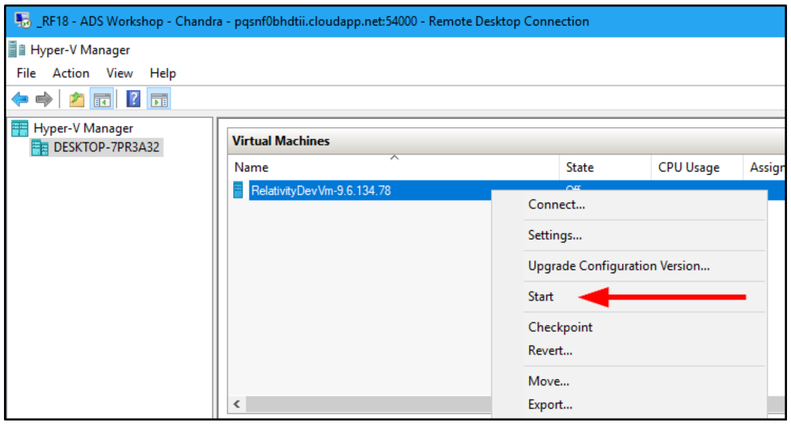
2. You will see a **Relativity 9.6.134.78 DevVM** listed in the Hyper-V Manager **Virtual Machines** section.



_RF18 - ADS Workshop - Chandra - pqsnf0bhdtii.cloudapp.net:54000 - Remote Desktop Connection

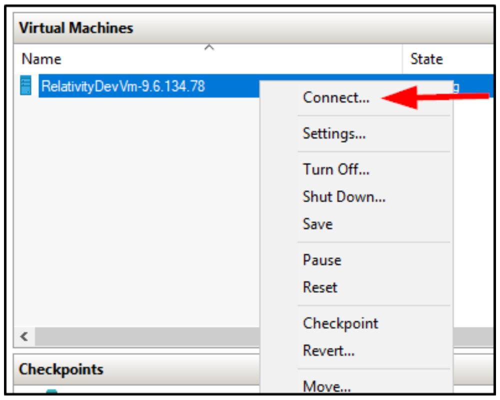
Name	State	CPU Usage	Assigned Memory
RelativityDevVm-9.6.134.78	Off		

3. Right click on the **DevVM** and select the **Start** option.

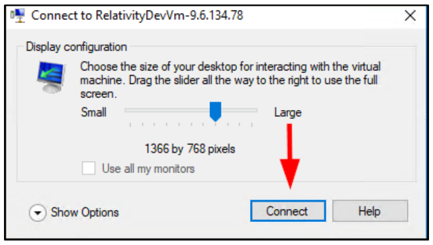


_RF18 - ADS Workshop - Chandra - pqsnf0bhdtii.cloudapp.net:54000 - Remote Desktop Connection

4. Right click on the **DevVM** and select the **Connect** option.



5. If you get a Display configuration pop-up, click the **Connect** button.


Connect to RelativityDevVm-9.6.134.78

Display configuration

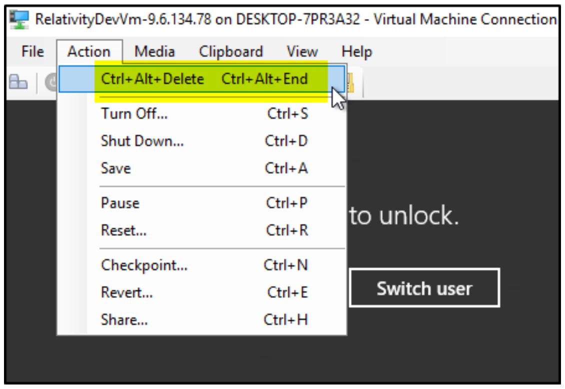
Choose the size of your desktop for interacting with the virtual machine. Drag the slider all the way to the right to use the full screen.

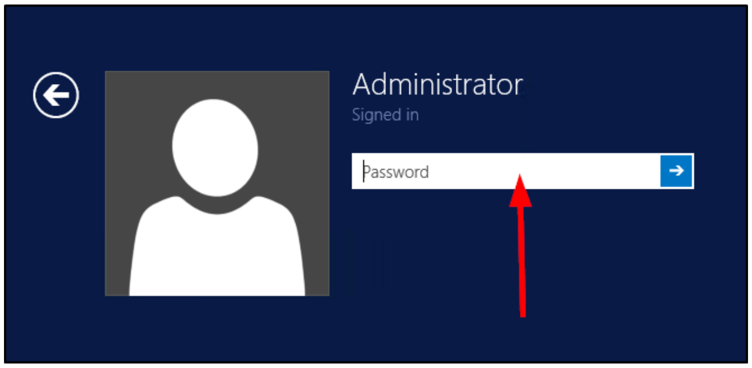
Small Large

1366 by 768 pixels

Use all my monitors

6. Select the **Action** Menu item and then select **Ctrl+Alt+Delete** option.


RelativityDevVm-9.6.134.78 on DESKTOP-7PR3A32 - Virtual Machine Connection



7. Use the below DevVM credentials highlighted in yellow to login to the VM.

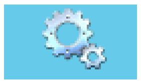
DevVM:
Windows Admin login: Administrator
Windows Admin password: Test1234!

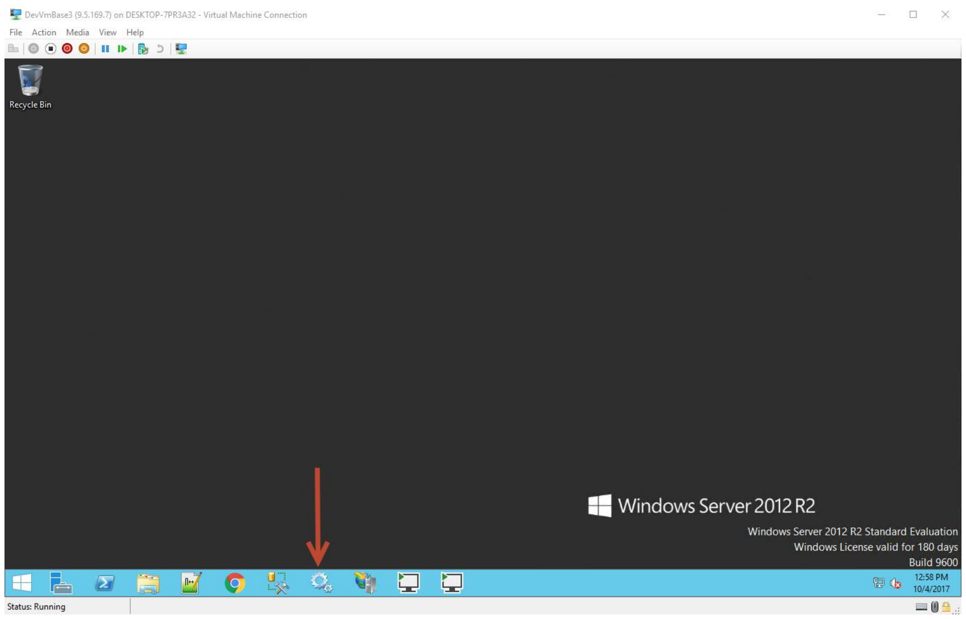
Relativity:
Admin login: relativity.admin@relativity.com
Admin password: Test1234!

SQL:
login: eddsdbo
password: Test1234!

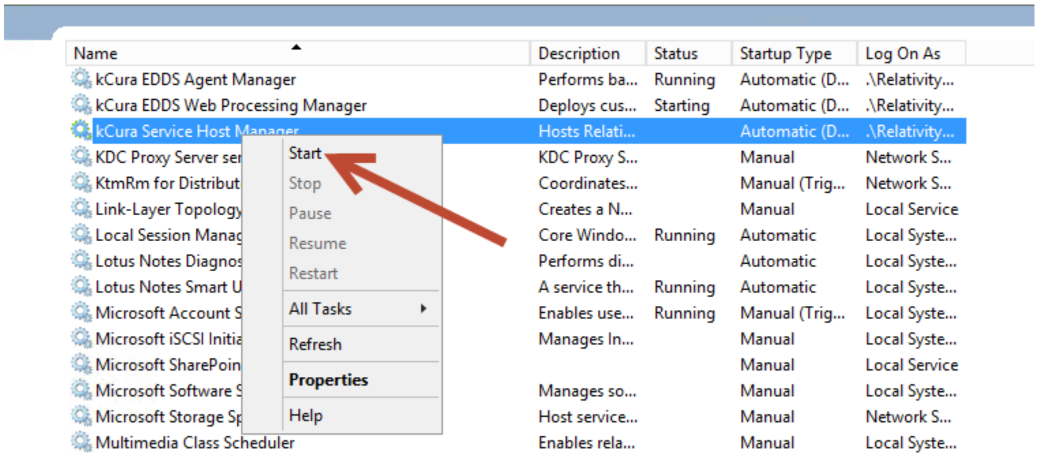
Admin login: sa
Admin password: Test1234!

8. Once you log in to the DevVM, you should see the Desktop with few applications pinned to the taskbar, which you will be using throughout the workshop.

9. Click on the **Services** program icon  in the taskbar as shown in the below screenshot.


DevVmBase3 (9.5.169.7) on DESKTOP-7PR3A32 - Virtual Machine Connection
File Action Media View Help
Recycle Bin
Windows Server 2012 R2
Windows Server 2012 R2 Standard Evaluation
Windows License valid for 180 days
Build 9600
12:58 PM
10/4/2017
Status: Running

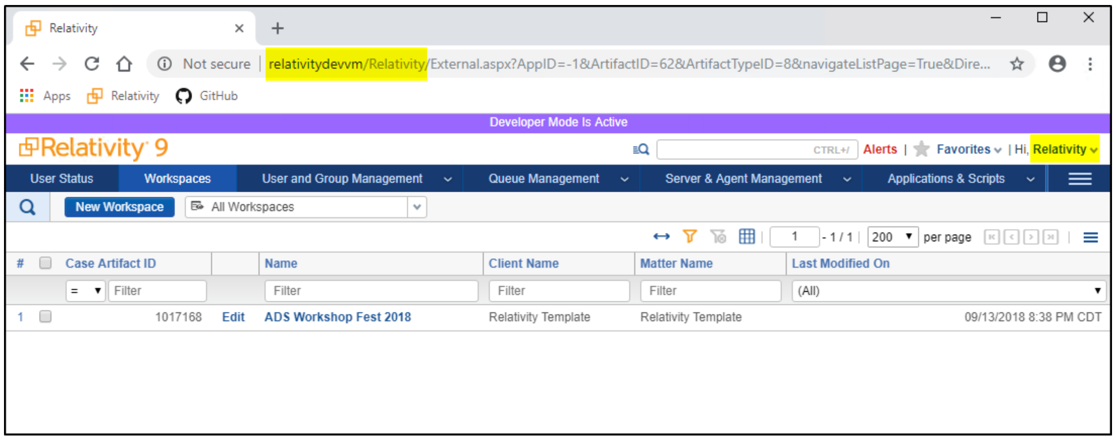
10. Make sure all the **services** listed below are **running**. If they are not already running, right click on the service and select the **Start** option,
    - Service Bus Gateway
    - Service Bus Message Broker
    - Service Bus Resource Provider
    - Service Bus VSS
    - kCura Service Host Manager
    - kCura EDDS Web Processing Manager
    - kCura EDDS Agent Manager



Name	Description	Status	Startup Type	Log On As
kCura EDDS Agent Manager	Performs ba...	Running	Automatic (D...	.\Relativity...
kCura EDDS Web Processing Manager	Deploys cus...	Starting	Automatic (D...	.\Relativity...
kCura Service Host Manager	Hosts Relati...		Automatic (D...	.\Relativity...
KDC Proxy Server ser	KDC Proxy S...		Manual	Network S...
KtmRm for Distribut	Coordinates...		Manual (Trig...	Network S...
Link-Layer Topology	Creates a N...		Manual	Local Service
Local Session Manag	Core Windo...	Running	Automatic	Local Syste...
Lotus Notes Diagnos	Performs di...		Automatic	Local Syste...
Lotus Notes Smart U	A service th...	Running	Automatic	Local Syste...
Microsoft Account S	Enables use...	Running	Manual (Trig...	Local Syste...
Microsoft iSCSI Initia	Manages In...		Manual	Local Syste...
Microsoft SharePoin			Manual	Local Service
Microsoft Software S	Manages so...		Manual	Local Syste...
Microsoft Storage Sp	Host service...		Manual	Network S...
Multimedia Class Scheduler	Enables rela...		Manual	Local Syste...

11. Open **Chrome** on your Workshop machine (Not the DevVM) and go to **http://RelativityDevVm/Relativity**. Use the below Relativity admin credentials to login. Verify that you can login to Relativity.

Relativity:
Admin login: relativity.admin@relativity.com
Admin password: Test1234!



#	Case Artifact ID		Name	Client Name	Matter Name	Last Modified On
	= Filter		Filter	Filter	Filter	(All)
1	1017168	Edit	ADS Workshop Fest 2018	Relativity Template	Relativity Template	09/13/2018 8:38 PM CDT

3 Visual Studio Templates

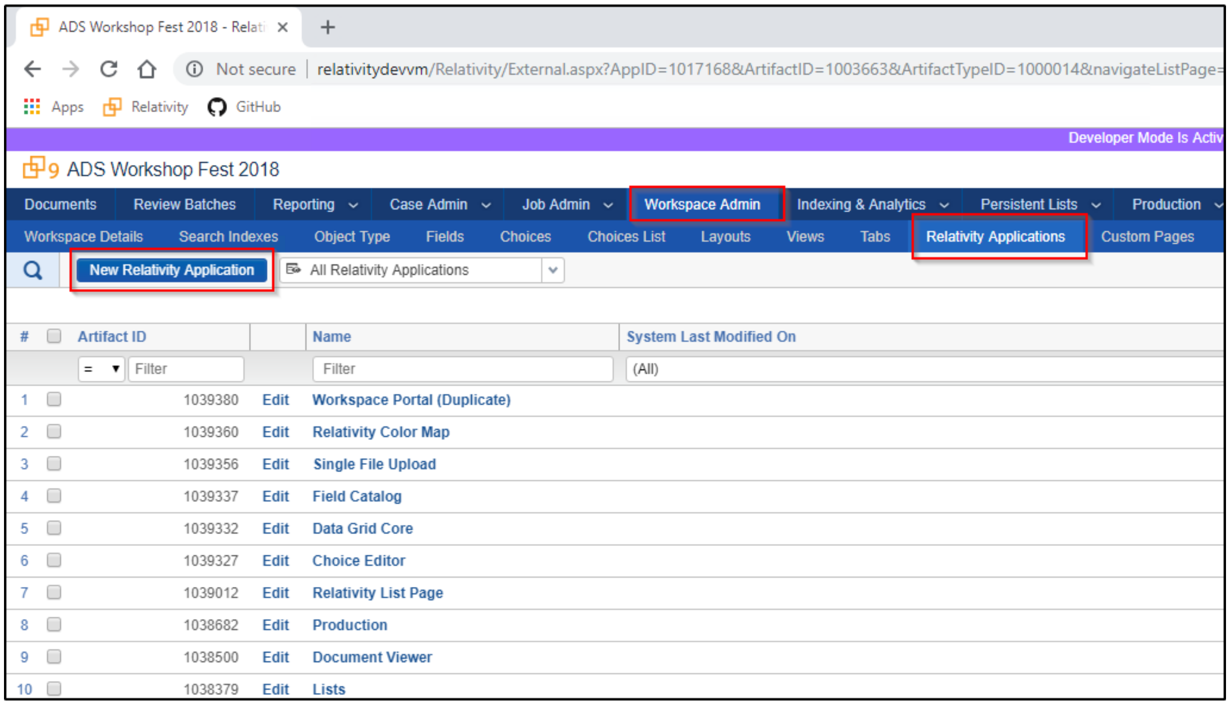
Relativity is an extensible platform that allows an administrator/developer to create customizations. Development on the Relativity Platform is made easier with the use of some tools such as Visual Studio Templates and the Publish to Relativity GUI. This section demonstrates utilizing the visual studio templates and Publish to Relativity tool to quickly iterate on application development.

3.1 Application Set Up

1. Open **Chrome** on your Workshop machine (Not the DevVM) and go to **http://RelativityDevVm/Relativity**. Use the below Relativity admin credentials to login.

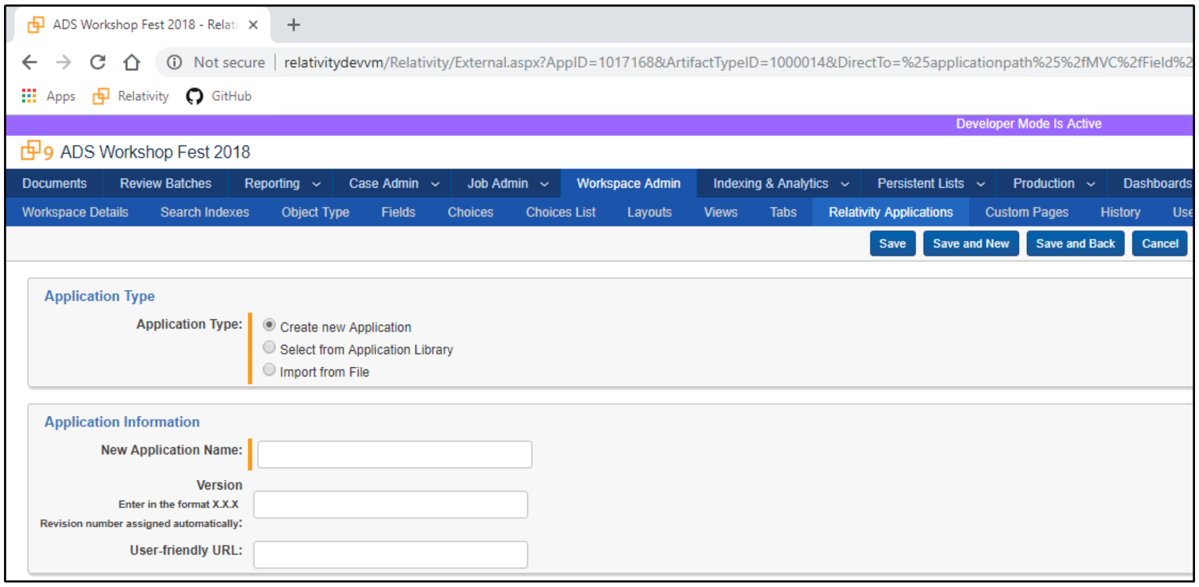
Relativity:
Admin login: relativity.admin@relativity.com
Admin password: Test1234!

2. Enter **ADS Workshop Fest 2018** and navigate to the **Relativity Applications** Tab.



#	Artifact ID		Name	System Last Modified On
	= Filter		Filter	(All)
1	1039380	Edit	Workspace Portal (Duplicate)	
2	1039360	Edit	Relativity Color Map	
3	1039356	Edit	Single File Upload	
4	1039337	Edit	Field Catalog	
5	1039332	Edit	Data Grid Core	
6	1039327	Edit	Choice Editor	
7	1039012	Edit	Relativity List Page	
8	1038682	Edit	Production	
9	1038500	Edit	Document Viewer	
10	1038379	Edit	Lists	

3. Create a new **Relativity Application** with the name of your choice.



Application Type

Application Type: [✓ Create new Application] Select from Application Library Import from File

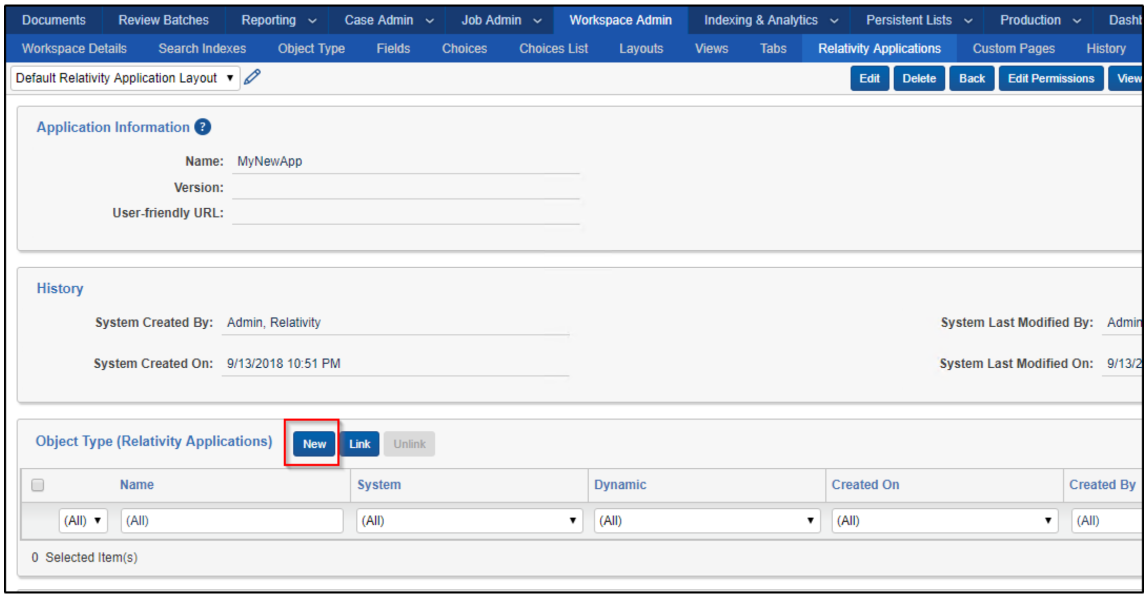
Application Information

New Application Name:

Version
Enter in the format X.X.X
Revision number assigned automatically:

User-friendly URL:

4. Create a new **Object Type** to the application from the view page. This auto-associates the objects initial views, fields, and layouts.



Application Information

Name: MyNewApp
Version:
User-friendly URL:

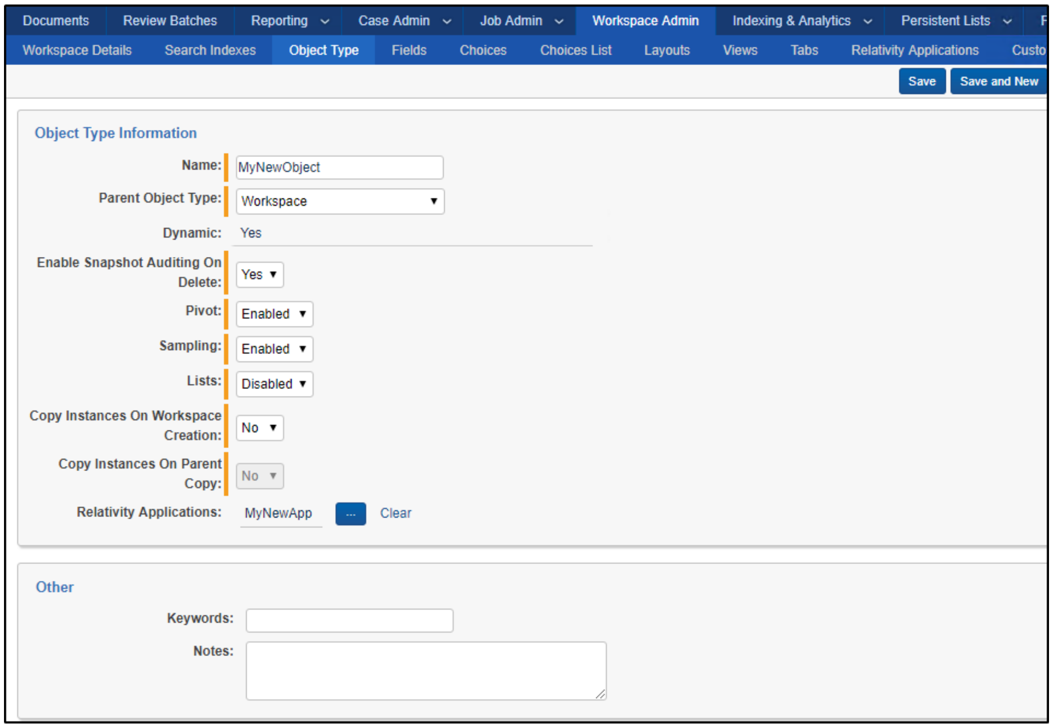
History

System Created By: Admin, Relativity
System Created On: 9/13/2018 10:51 PM
System Last Modified By: Admin
System Last Modified On: 9/13/2

Object Type (Relativity Applications)

	Name	System	Dynamic	Created On	Created By
(All)	(All)	(All)	(All)	(All)	(All)

0 Selected Item(s)



Object Type Information

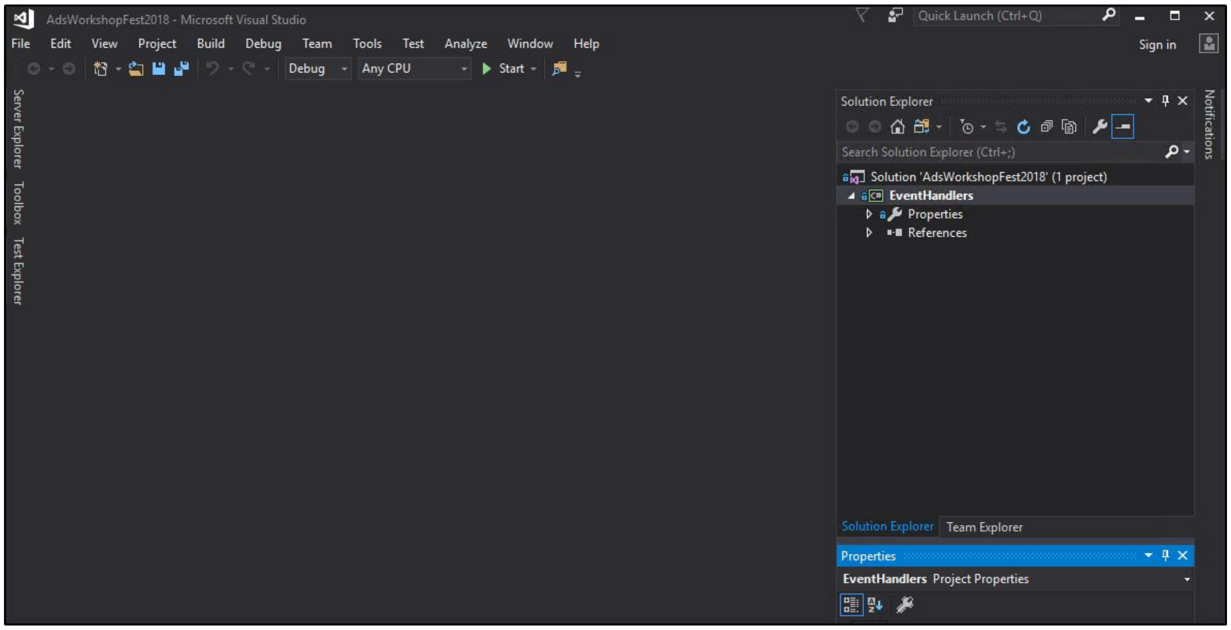
Name:	MyNewObject
Parent Object Type:	Workspace
Dynamic:	Yes
Enable Snapshot Auditing On Delete:	Yes
Pivot:	Enabled
Sampling:	Enabled
Lists:	Disabled
Copy Instances On Workspace Creation:	No
Copy Instances On Parent Copy:	No
Relativity Applications:	MyNewApp ... Clear

Other

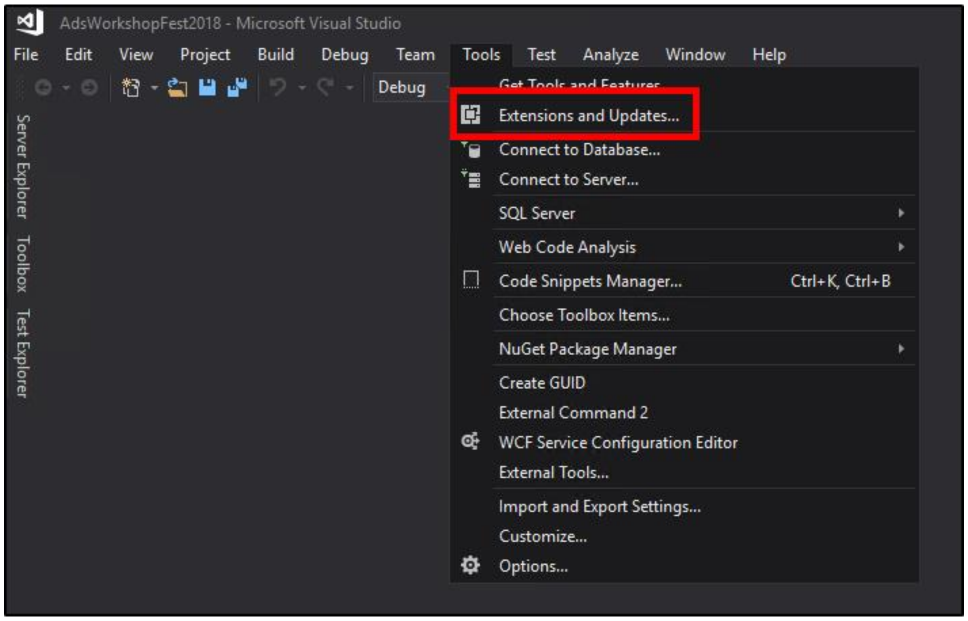
Keywords:	
Notes:	

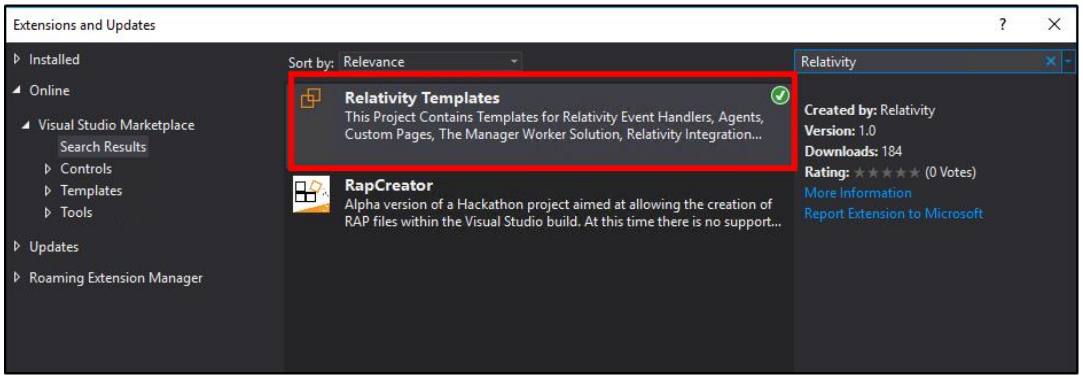
3.2 New Event Handler

1. Open the Visual Studio solution **AdsWorkshopFest2018.sln** in the **E:\Github\ads-workshop-fest-2018\Projects\Starting\1_Templates\Project** folder.


AdsWorkshopFest2018 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug Any CPU Start
Quick Launch (Ctrl+Q)
Sign in
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'AdsWorkshopFest2018' (1 project)
EventHandlers
Properties
References
Solution Explorer Team Explorer
Properties
EventHandlers Project Properties
Server Explorer Toolbox Test Explorer
Notifications

2. Verify the **Relativity Templates** are installed. You may need to **restart** Visual Studio.
3. **Tools** — **Extensions and Updates** — Search "**Relativity**"


AdsWorkshopFest2018 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug
Get Tools and Features...
Extensions and Updates...
Connect to Database...
Connect to Server...
SQL Server
Web Code Analysis
Code Snippets Manager... Ctrl+K, Ctrl+B
Choose Toolbox Items...
NuGet Package Manager
Create GUID
External Command 2
WCF Service Configuration Editor
External Tools...
Import and Export Settings...
Customize...
Options...
Server Explorer Toolbox Test Explorer


Extensions and Updates

Installed
Online
Visual Studio Marketplace
Search Results
Controls
Templates
Tools
Updates
Roaming Extension Manager

Sort by: Relevance

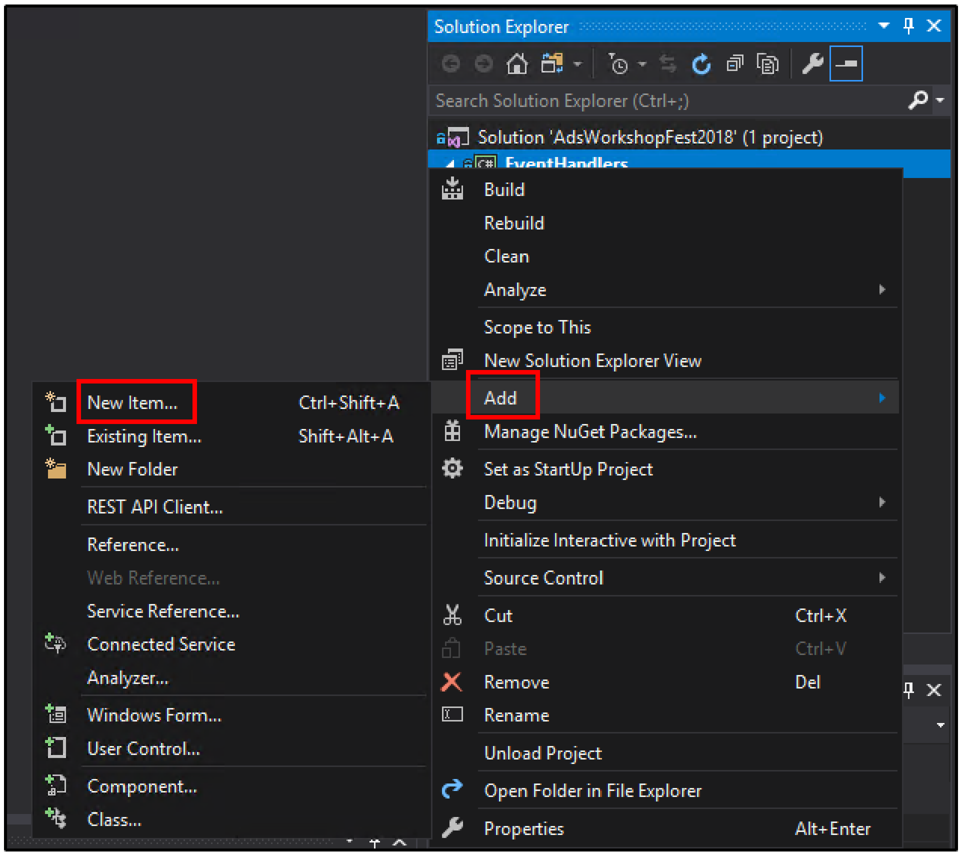
Relativity

Relativity Templates
This Project Contains Templates for Relativity Event Handlers, Agents, Custom Pages, The Manager Worker Solution, Relativity Integration...

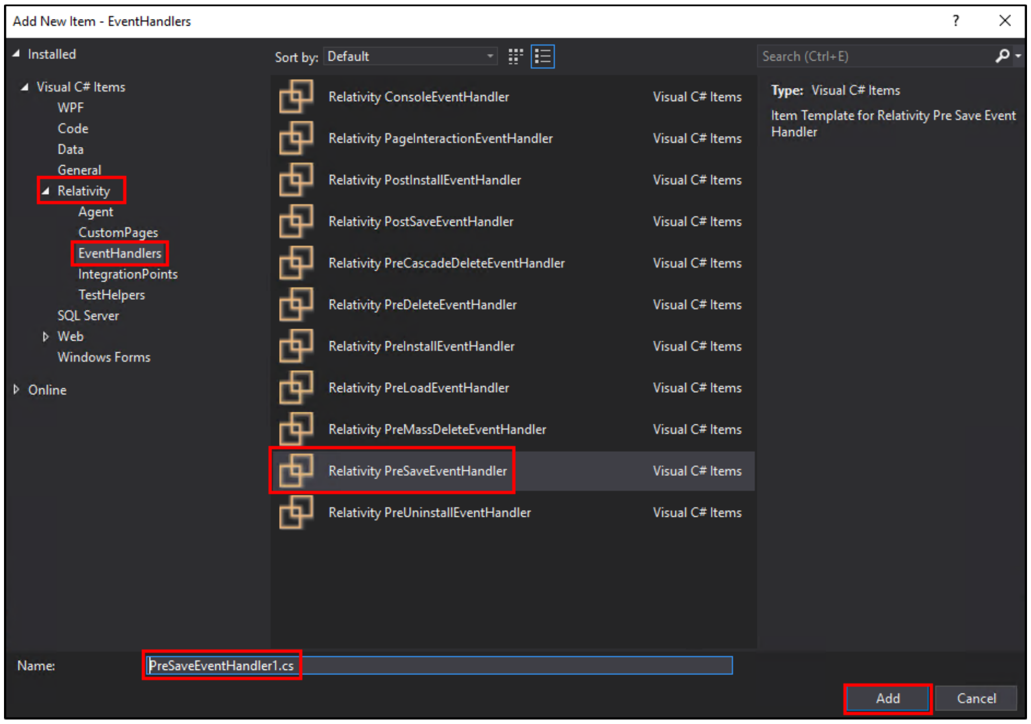
RapCreator
Alpha version of a Hackathon project aimed at allowing the creation of RAP files within the Visual Studio build. At this time there is no support...

Created by: Relativity
Version: 1.0
Downloads: 184
Rating: ★★★★★ (0 Votes)
More Information
Report Extension to Microsoft

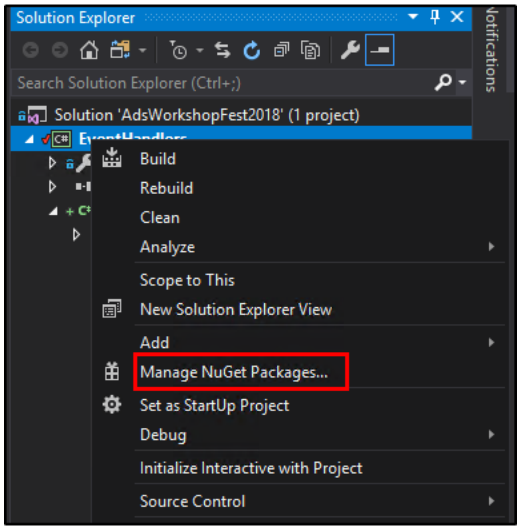
4. Right click the **EventHandlers** Project in the solution explorer. Select Add — **New Item**.



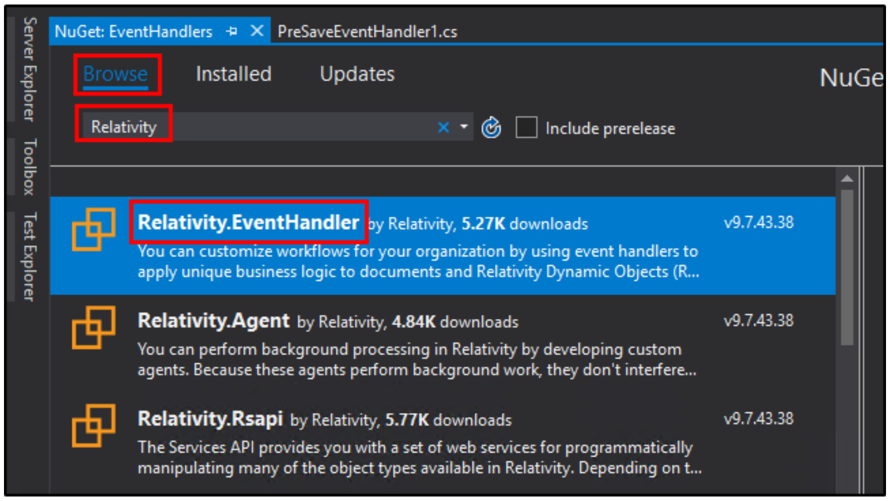
5. Under the Relativity Section, select the **pre-save event handler template.**


Add New Item - EventHandlers
Installed
Visual C# Items
WPF
Code
Data
General
Relativity
Agent
CustomPages
EventHandlers
IntegrationPoints
TestHelpers
SQL Server
Web
Windows Forms
Online
Sort by: Default
Search (Ctrl+E)
Relativity ConsoleEventHandler Visual C# Items
Relativity PageInteractionEventHandler Visual C# Items
Relativity PostInstallEventHandler Visual C# Items
Relativity PostSaveEventHandler Visual C# Items
Relativity PreCascadeDeleteEventHandler Visual C# Items
Relativity PreDeleteEventHandler Visual C# Items
Relativity PreInstallEventHandler Visual C# Items
Relativity PreLoadEventHandler Visual C# Items
Relativity PreMassDeleteEventHandler Visual C# Items
Relativity PreSaveEventHandler Visual C# Items
Relativity PreUninstallEventHandler Visual C# Items
Type: Visual C# Items
Item Template for Relativity Pre Save Event Handler
Name: PreSaveEventHandler1.cs
Add
Cancel

6. Click **Add**.
7. Install the 9.6.134.78 **Relativity.EventHandler** NuGet package.
    a. Right click on the Project.
    b. Select **Manage Nuget Packages** option.



    c. Switch to **Browse** tab.
    d. Search for "**Relativity**".
    e. Select the **Relativity.EventHandler** NuGet package.

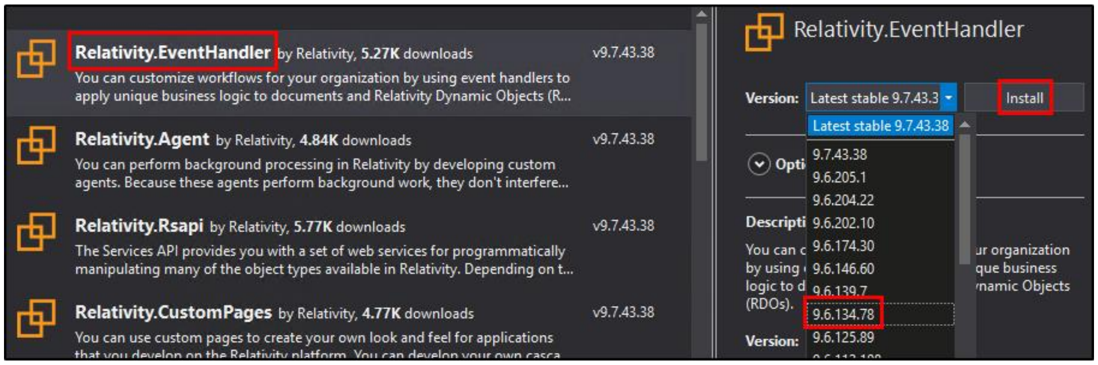


Relativity.EventHandler by Relativity, 5.27K downloads v9.7.43.38
You can customize workflows for your organization by using event handlers to apply unique business logic to documents and Relativity Dynamic Objects (R...

Relativity.Agent by Relativity, 4.84K downloads v9.7.43.38
You can perform background processing in Relativity by developing custom agents. Because these agents perform background work, they don't interfere...

Relativity.Rsapi by Relativity, 5.77K downloads v9.7.43.38
The Services API provides you with a set of web services for programmatically manipulating many of the object types available in Relativity. Depending on t...

    f. From the **Version** drop-down select the **9.6.134.78** option.
    g. Click **Install**.


Relativity.EventHandler by Relativity, **5.27K** downloads v9.7.43.38
You can customize workflows for your organization by using event handlers to apply unique business logic to documents and Relativity Dynamic Objects (R...

Relativity.Agent by Relativity, **4.84K** downloads v9.7.43.38
You can perform background processing in Relativity by developing custom agents. Because these agents perform background work, they don't interfere...

Relativity.Rsapi by Relativity, **5.77K** downloads v9.7.43.38
The Services API provides you with a set of web services for programmatically manipulating many of the object types available in Relativity. Depending on t...

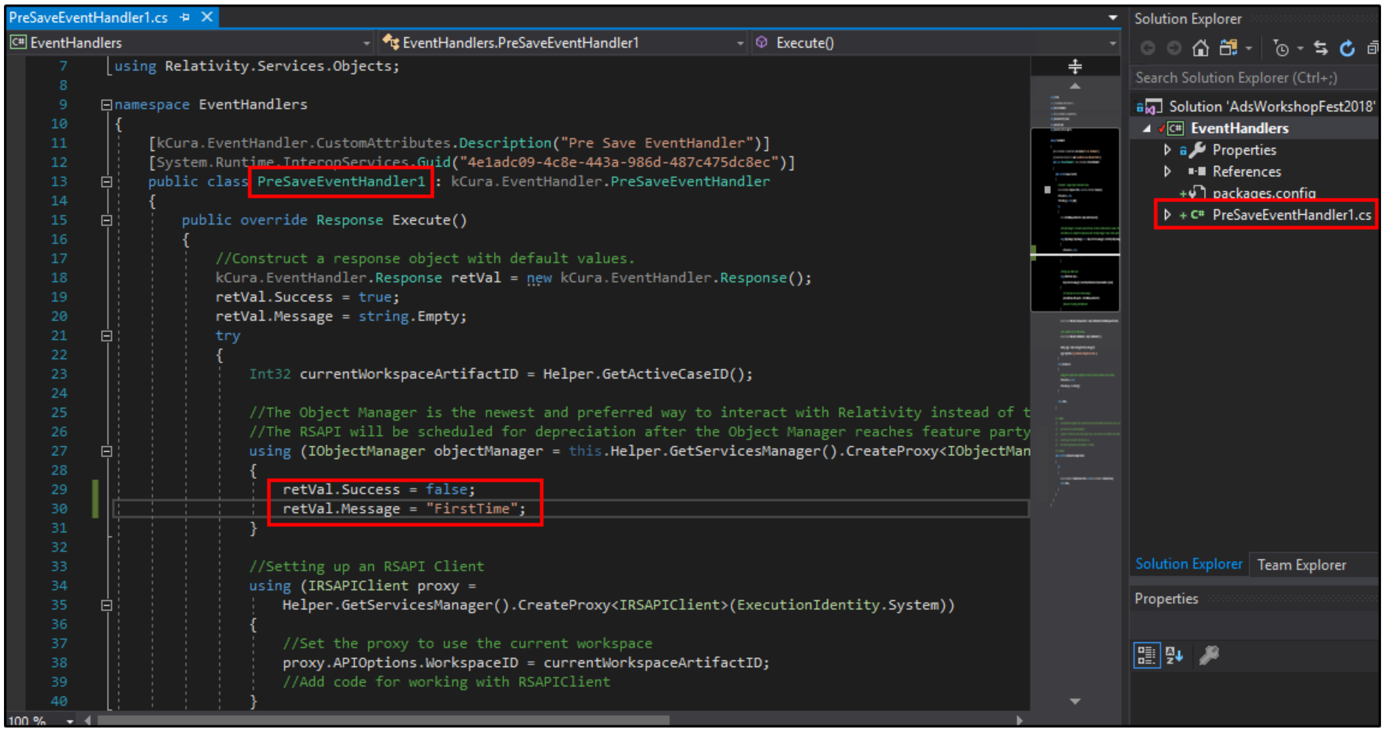
Relativity.CustomPages by Relativity, **4.77K** downloads v9.7.43.38
You can use custom pages to create your own look and feel for applications that you develop on the Relativity platform. You can develop your own casca

Relativity.EventHandler

Version: Latest stable 9.7.43.3 Install

Latest stable 9.7.43.38
9.7.43.38
9.6.205.1
9.6.204.22
9.6.202.10
9.6.174.30
9.6.146.60
9.6.139.7
9.6.134.78
9.6.125.89

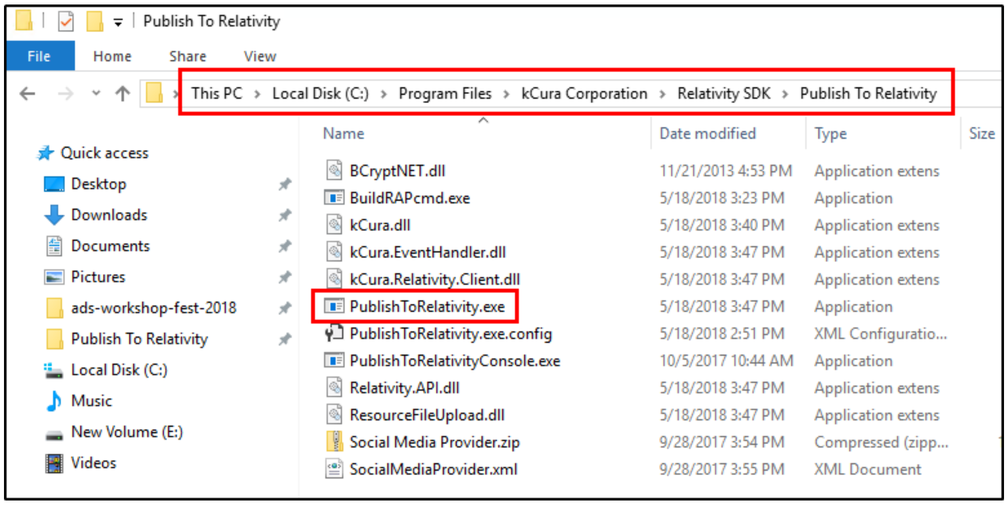
8. Update the event handler response.
   a. *Success* – false
   b. *Message* – Message of your choice



```
using Relativity.Services.Objects;

namespace EventHandlers
{
    [kCura.EventHandler.CustomAttributes.Description("Pre Save EventHandler")]
    [System.Runtime.InteropServices.Guid("4e1adc09-4c8e-443a-986d-487c475dc8ec")]
    public class PreSaveEventHandler1 : kCura.EventHandler.PreSaveEventHandler
    {
        public override Response Execute()
        {
            //Construct a response object with default values.
            kCura.EventHandler.Response retVal = new kCura.EventHandler.Response();
            retVal.Success = true;
            retVal.Message = string.Empty;
            try
            {
                Int32 currentWorkspaceArtifactID = Helper.GetActiveCaseID();

                //The Object Manager is the newest and preferred way to interact with Relativity instead of t
                //The RSAPI will be scheduled for depreciation after the Object Manager reaches feature party
                using (IObjectManager objectManager = this.Helper.GetServicesManager().CreateProxy<IObjectMan
                {
                    retVal.Success = false;
                    retVal.Message = "FirstTime";
                }

                //Setting up an RSAPI Client
                using (IRSAPIClient proxy =
                    Helper.GetServicesManager().CreateProxy<IRSAPIClient>(ExecutionIdentity.System))
                {
                    //Set the proxy to use the current workspace
                    proxy.APIOptions.WorkspaceID = currentWorkspaceArtifactID;
                    //Add code for working with RSAPIClient
                }
```

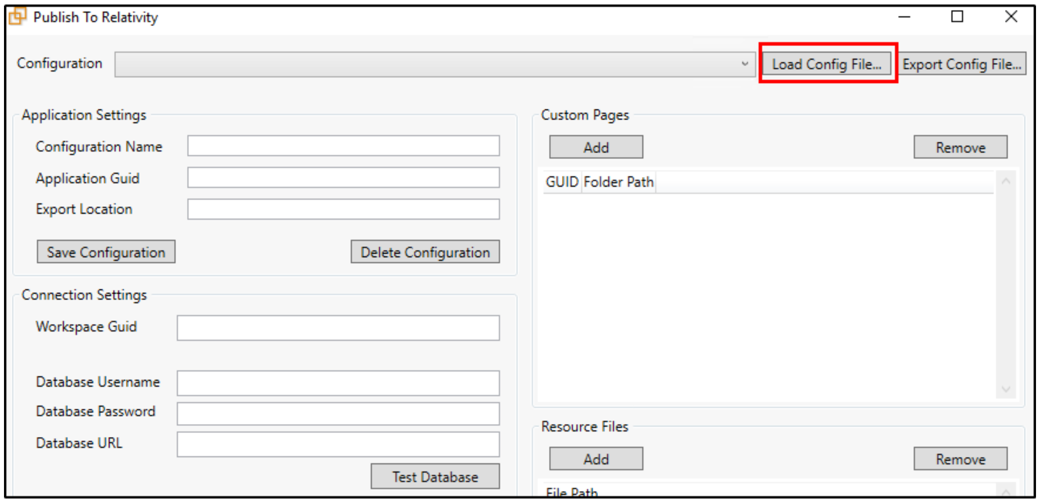
3.3 Publish

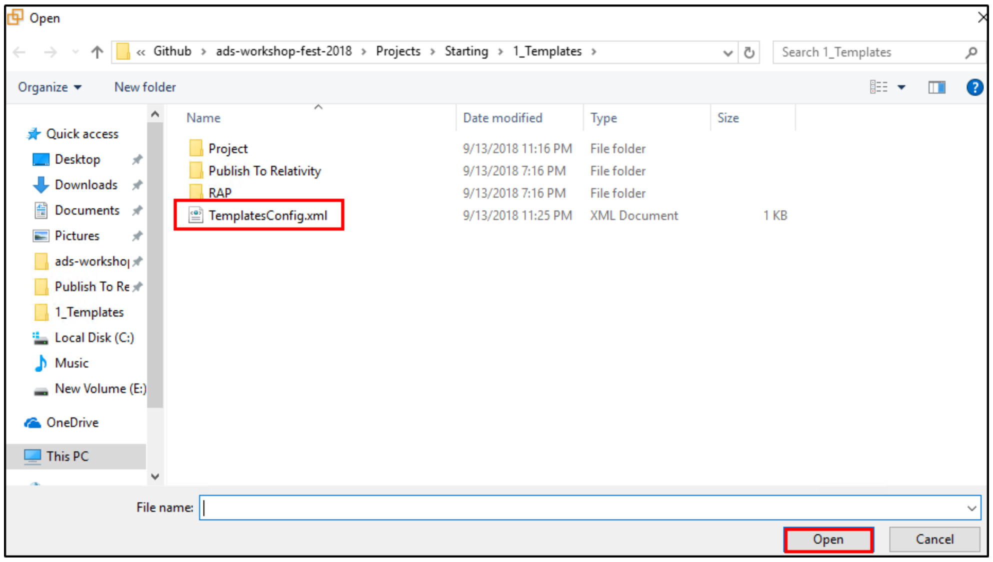
1. Build the solution.
2. **Build — Build Solution**
3. Run the Publish to Relativity executable.



Name	Date modified	Type	Size
BCryptNET.dll	11/21/2013 4:53 PM	Application extens	
BuildRAPcmd.exe	5/18/2018 3:23 PM	Application	
kCura.dll	5/18/2018 3:40 PM	Application extens	
kCura.EventHandler.dll	5/18/2018 3:47 PM	Application extens	
kCura.Relativity.Client.dll	5/18/2018 3:47 PM	Application extens	
PublishToRelativity.exe	5/18/2018 3:47 PM	Application	
PublishToRelativity.exe.config	5/18/2018 2:51 PM	XML Configuratio...	
PublishToRelativityConsole.exe	10/5/2017 10:44 AM	Application	
Relativity.API.dll	5/18/2018 3:47 PM	Application extens	
ResourceFileUpload.dll	5/18/2018 3:47 PM	Application extens	
Social Media Provider.zip	9/28/2017 3:54 PM	Compressed (zipp...	
SocialMediaProvider.xml	9/28/2017 3:55 PM	XML Document	

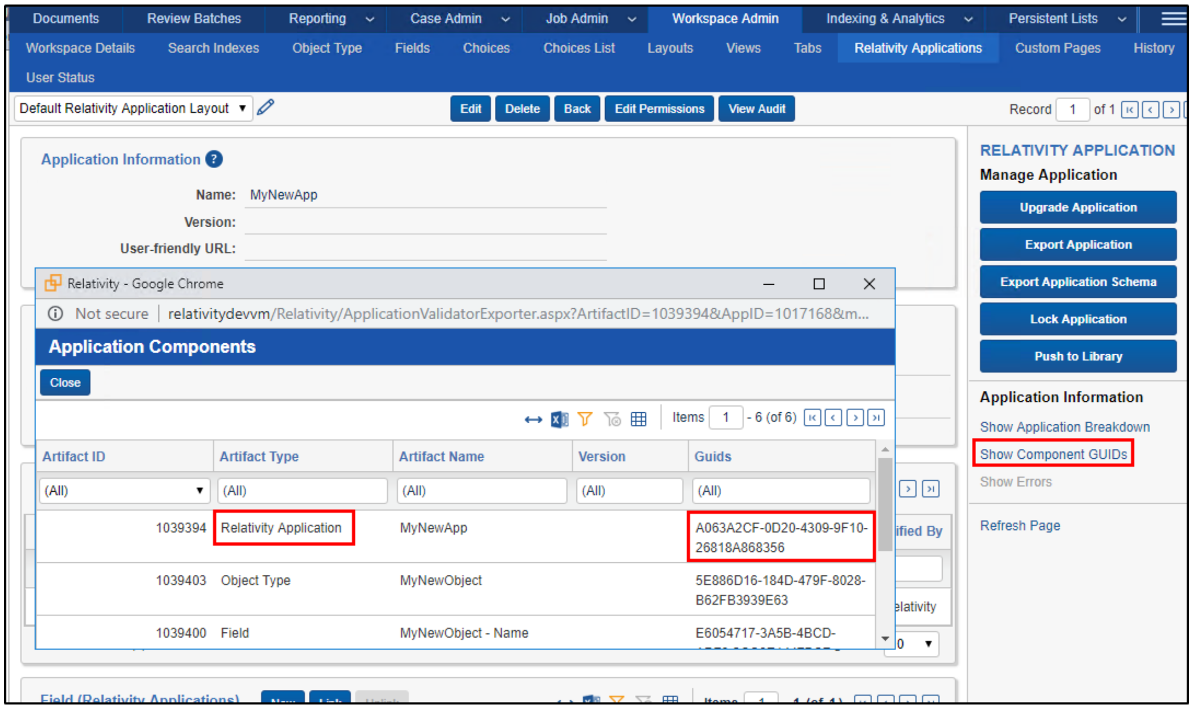
4. Click the **Load Config File** button and select the **TemplatesConfig.xml** file in the **E:\Github\ads-workshop-fest-2018\Projects\Starting\1_Templates** folder.





Name	Date modified	Type	Size
Project	9/13/2018 11:16 PM	File folder	
Publish To Relativity	9/13/2018 7:16 PM	File folder	
RAP	9/13/2018 7:16 PM	File folder	
TemplatesConfig.xml	9/13/2018 11:25 PM	XML Document	1 KB

5. Enter the **Relativity Password** (Test1234!).
6. Click on **Test Connections** to validate the connections.
7. Replace the **Application GUID** with your applications GUID.
   a. To look up your applications GUID, navigate back to Relativity Application you created in the workspace. Then select **Show Component GUIDs**.
   b. **Developer Mode** must be active.



Application Information

Name: MyNewApp

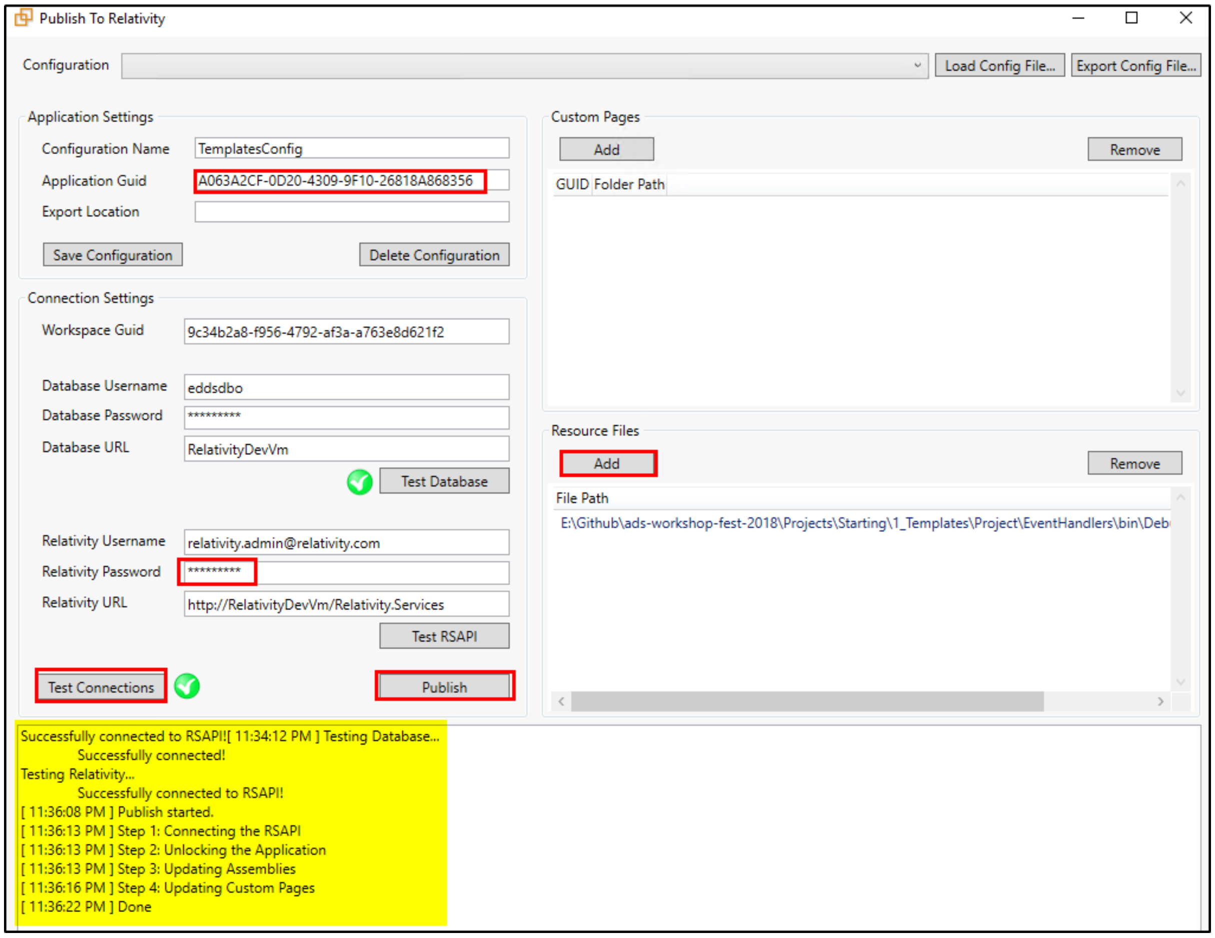
Version:

User-friendly URL:

Artifact ID	Artifact Type	Artifact Name	Version	Guids
(All)	(All)	(All)	(All)	(All)
1039394	Relativity Application	MyNewApp		A063A2CF-0D20-4309-9F10-26818A868356
1039403	Object Type	MyNewObject		5E886D16-184D-479F-8028-B62FB3939E63
1039400	Field	MyNewObject - Name		E6054717-3A5B-4BCD-

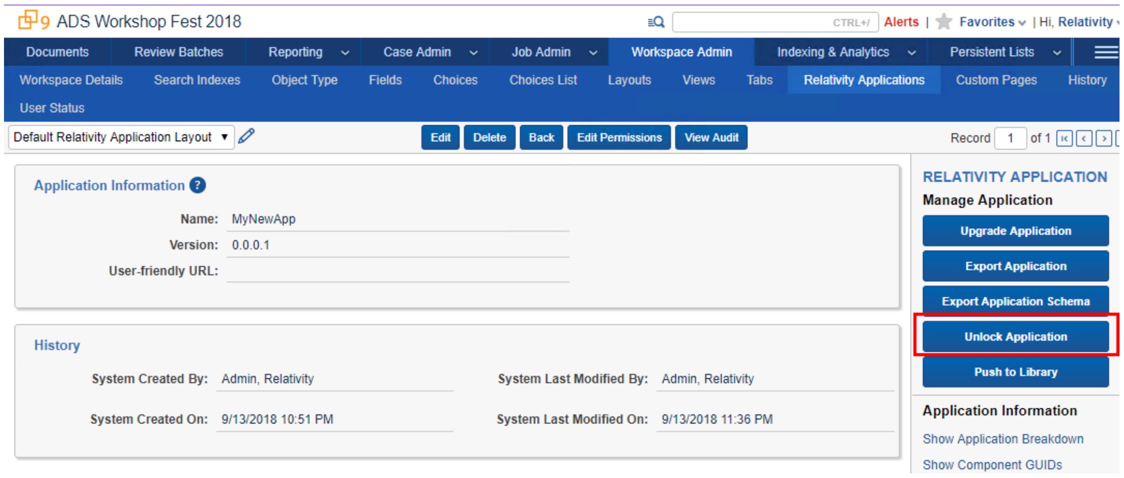
8. In the **Resource Files** Section click Add.
9. Navigate to **E:\Github\ads-workshop-fest-2018\Projects\Starting\1_Templates\Project\EventHandlers\bin\Debug** and select **EventHandlers.dll** file.

10. Click **Publish**.


Publish To Relativity
Configuration
Load Config File...
Export Config File...
Application Settings
Configuration Name: TemplatesConfig
Application Guid: A063A2CF-0D20-4309-9F10-26818A868356
Export Location
Save Configuration
Delete Configuration
Connection Settings
Workspace Guid: 9c34b2a8-f956-4792-af3a-a763e8d621f2
Database Username: eddsdbo
Database Password: *********
Database URL: RelativityDevVm
Test Database
Relativity Username: relativity.admin@relativity.com
Relativity Password: *********
Relativity URL: http://RelativityDevVm/Relativity.Services
Test RSAPI
Test Connections
Publish
Custom Pages
Add
Remove
GUID Folder Path
Resource Files
Add
Remove
File Path
E:\Github\ads-workshop-fest-2018\Projects\Starting\1_Templates\Project\EventHandlers\bin\Deb

```
Successfully connected to RSAPI![ 11:34:12 PM ] Testing Database...
        Successfully connected!
Testing Relativity...
        Successfully connected to RSAPI!
[ 11:36:08 PM ] Publish started.
[ 11:36:13 PM ] Step 1: Connecting the RSAPI
[ 11:36:13 PM ] Step 2: Unlocking the Application
[ 11:36:13 PM ] Step 3: Updating Assemblies
[ 11:36:16 PM ] Step 4: Updating Custom Pages
[ 11:36:22 PM ] Done
```

11. Open Relativity and navigate to your application created in section 3.1.
12. **Unlock** the application.



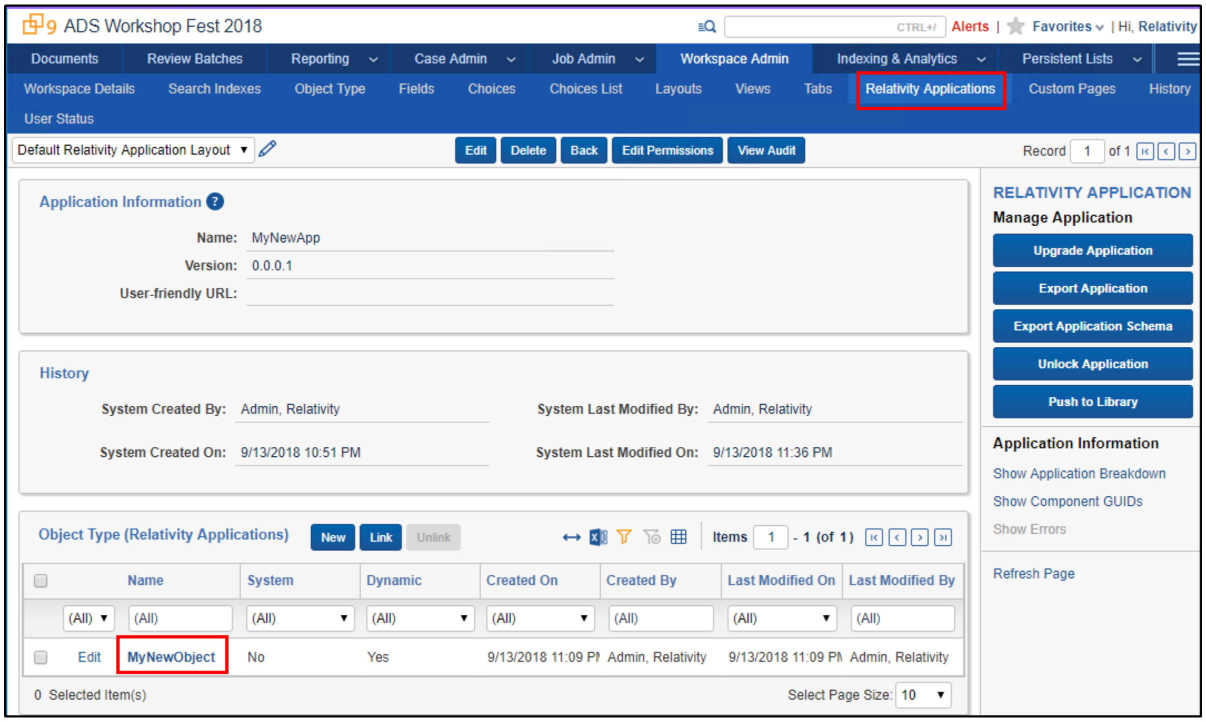
Application Information

Name:	MyNewApp
Version:	0.0.0.1
User-friendly URL:	

History

System Created By:	Admin, Relativity	System Last Modified By:	Admin, Relativity
System Created On:	9/13/2018 10:51 PM	System Last Modified On:	9/13/2018 11:36 PM

13. Navigate to the view page of the object type created in section 3.1.



Application Information

Name:	MyNewApp
Version:	0.0.0.1
User-friendly URL:	

History

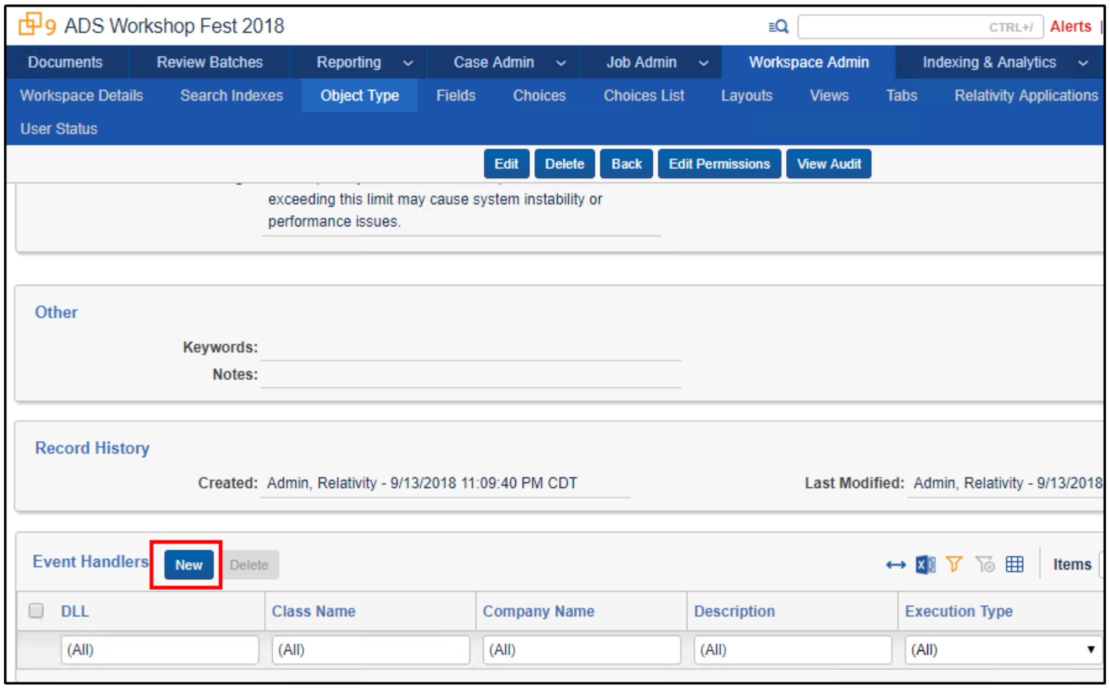
System Created By:	Admin, Relativity	System Last Modified By:	Admin, Relativity
System Created On:	9/13/2018 10:51 PM	System Last Modified On:	9/13/2018 11:36 PM

Object Type (Relativity Applications) New Link Unlink Items 1 - 1 (of 1)

		Name	System	Dynamic	Created On	Created By	Last Modified On	Last Modified By
	(All)	(All)	(All)	(All)	(All)	(All)	(All)	(All)
☐	Edit	MyNewObject	No	Yes	9/13/2018 11:09 PI	Admin, Relativity	9/13/2018 11:09 PI	Admin, Relativity

0 Selected Item(s) Select Page Size: 10

14. Click **New** on the Event handler associated item list.



exceeding this limit may cause system instability or performance issues.

Other

Keywords:

Notes:

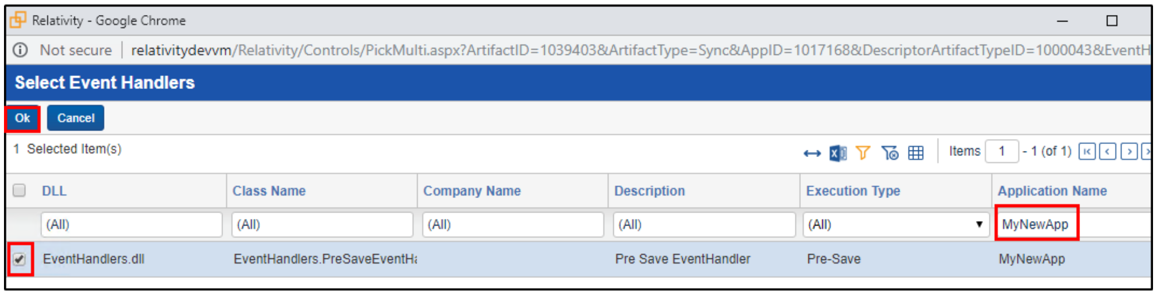
Record History

Created: Admin, Relativity - 9/13/2018 11:09:40 PM CDT Last Modified: Admin, Relativity - 9/13/2018

Event Handlers New Delete Items

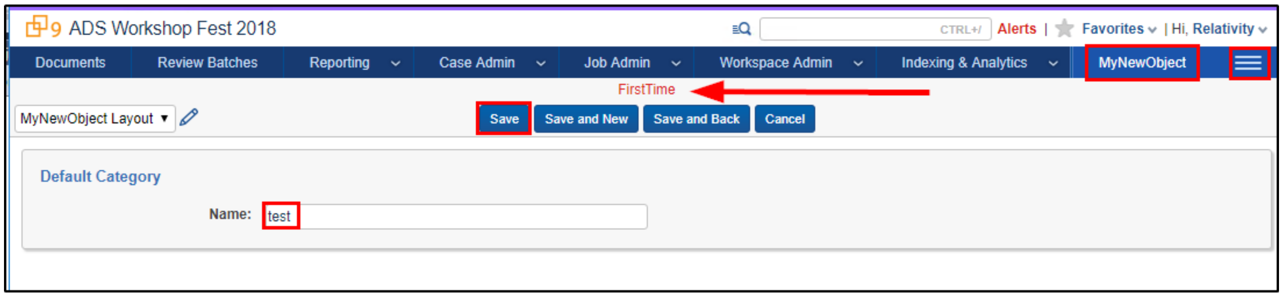
	DLL	Class Name	Company Name	Description	Execution Type
	(All)	(All)	(All)	(All)	(All)

15. Find the newly created Event Handler associated to your application.
    a. Filter by Application Name.
    b. Select it from the list and click **Ok**.


Relativity - Google Chrome
Not secure | relativitydevvm/Relativity/Controls/PickMulti.aspx?ArtifactID=1039403&ArtifactType=Sync&AppID=1017168&DescriptorArtifactTypeID=1000043&EventH
Select Event Handlers
Ok
Cancel
1 Selected Item(s)
Items 1 - 1 (of 1)

DLL	Class Name	Company Name	Description	Execution Type	Application Name
(All)	(All)	(All)	(All)	(All)	MyNewApp
EventHandlers.dll	EventHandlers.PreSaveEventH		Pre Save EventHandler	Pre-Save	MyNewApp

16. Test your event handler by creating a new instance of your object type.



Name: test

4 Sample Application Overview

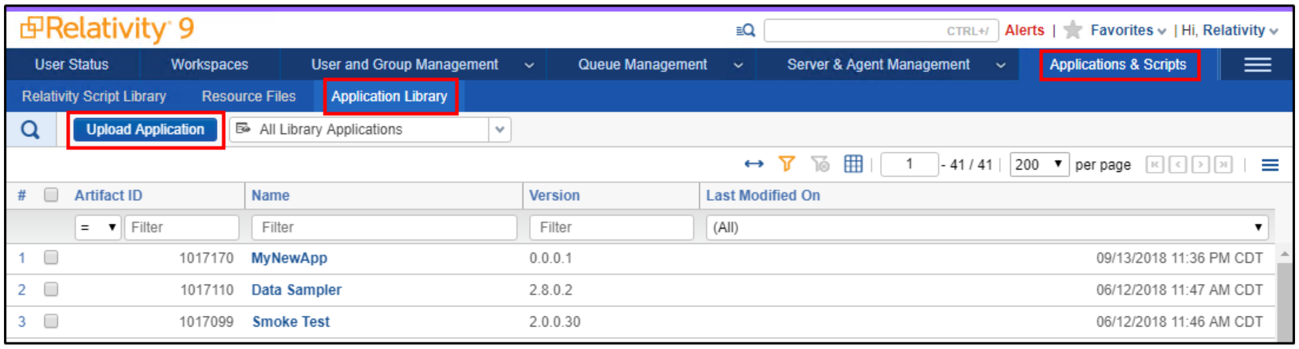
The functionality of the Sample Relativity application (ADS Workshop Fest 2018) is to calculate the selected Instance metrics and write them to the Instance Metrics Job.

The ADS Workshop Fest 2018 Relativity application consists of the following:

- **Custom Object**
  - Instance Metrics Job
    - Fields
      - Metrics
      - Status
      - Workspaces Count
      - Users Count
      - Groups Count
      - Errors
- **Event Handler**
  - PreSaveStatusUpdate
- **Agent**
  - Instance Metrics Calculator Agent

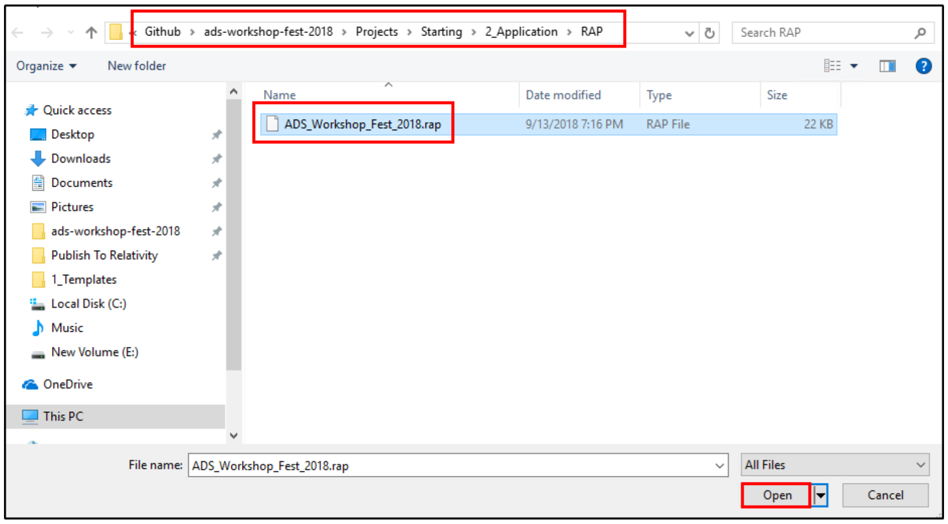
4.1 Install Sample Application

1. Go to the **Application Library** tab.
2. Click on the **Upload Application** button



#	Artifact ID	Name	Version	Last Modified On
1	1017170	MyNewApp	0.0.0.1	09/13/2018 11:36 PM CDT
2	1017110	Data Sampler	2.8.0.2	06/12/2018 11:47 AM CDT
3	1017099	Smoke Test	2.0.0.30	06/12/2018 11:46 AM CDT

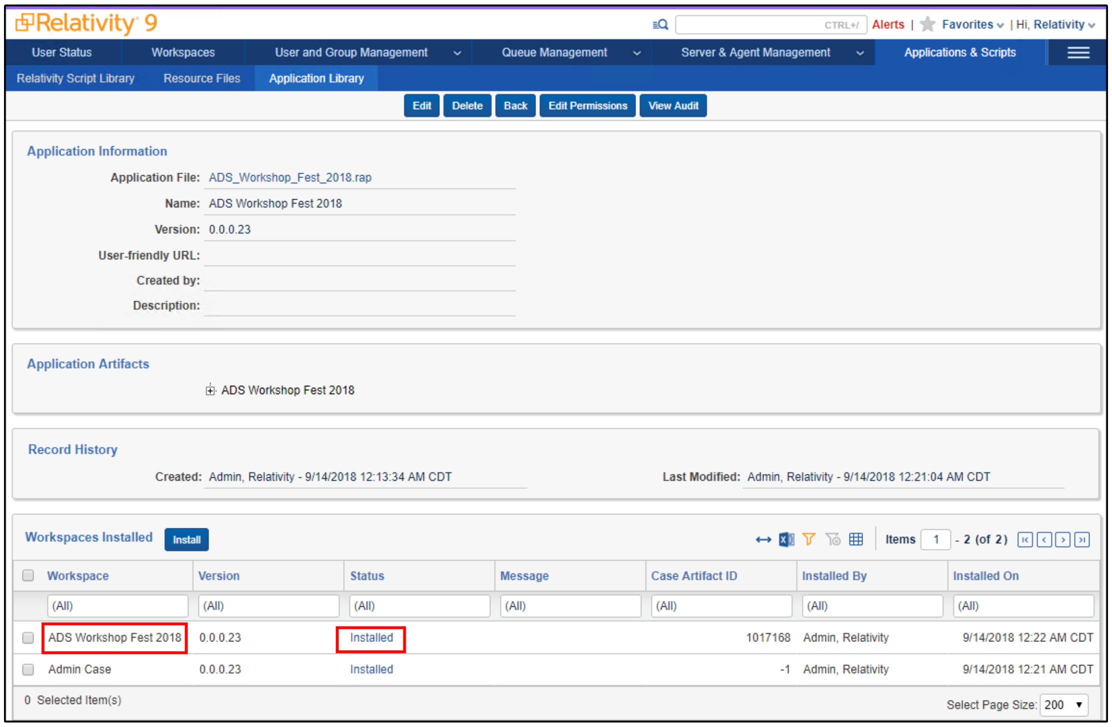
3. Click on **Choose File** button.
4. Navigate to **E:\Github\ads-workshop-fest-2018\Projects\Starting\2_Application\RAP** folder and select the **ADS_Workshop_Fest_2018.rap** file.



Name	Date modified	Type	Size
ADS_Workshop_Fest_2018.rap	9/13/2018 7:16 PM	RAP File	22 KB

File name: ADS_Workshop_Fest_2018.rap

5. Click the **Save** button.
6. Under the **Workspaces Installed** section, click the **Install** button.
7. Select the **ADS Workshop Fest 2018** workspace and click the **Save** button.



Application Information

Application File:	ADS_Workshop_Fest_2018.rap
Name:	ADS Workshop Fest 2018
Version:	0.0.0.23
User-friendly URL:	
Created by:	
Description:	

Application Artifacts

ADS Workshop Fest 2018

Record History

Created: Admin, Relativity - 9/14/2018 12:13:34 AM CDT Last Modified: Admin, Relativity - 9/14/2018 12:21:04 AM CDT

Workspaces Installed Install

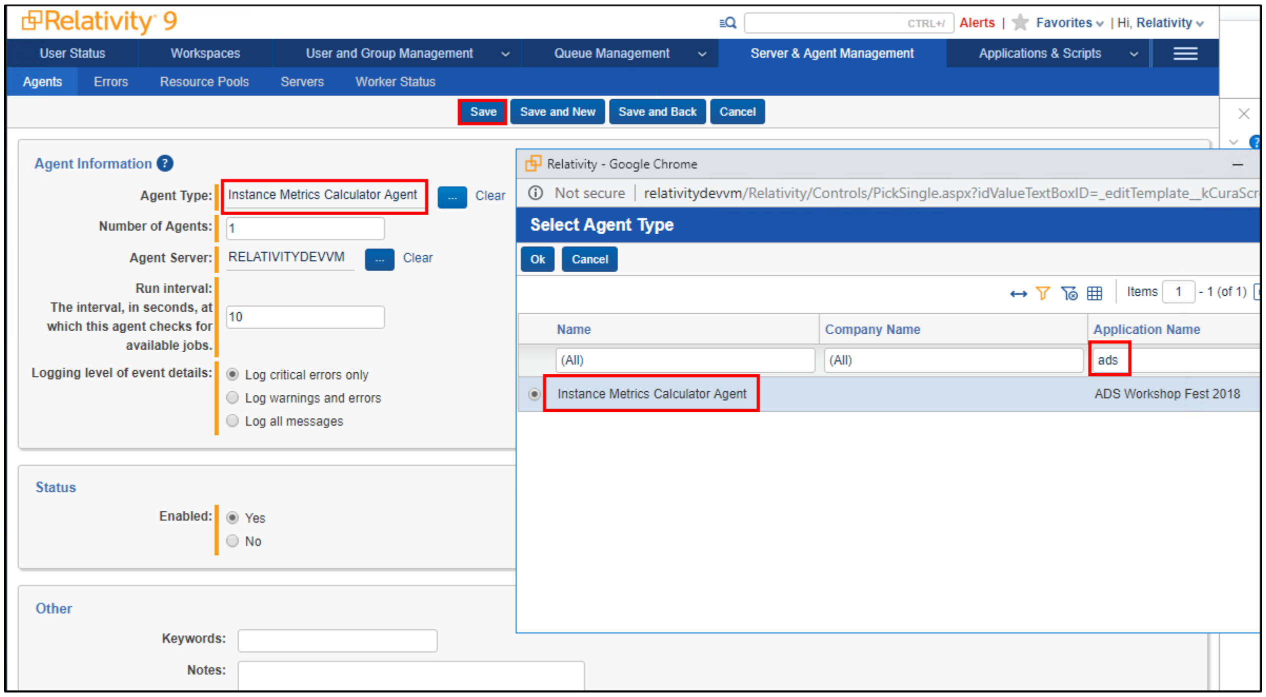
Items 1 - 2 (of 2)

	Workspace	Version	Status	Message	Case Artifact ID	Installed By	Installed On
	(All)	(All)	(All)	(All)	(All)	(All)	(All)
☐	ADS Workshop Fest 2018	0.0.0.23	Installed		1017168	Admin, Relativity	9/14/2018 12:22 AM CDT
☐	Admin Case	0.0.0.23	Installed		-1	Admin, Relativity	9/14/2018 12:21 AM CDT

0 Selected Item(s) Select Page Size: 200

4.2 Run Sample Application

1. Once the application is installed, navigate to the **Agents** tab and create an **Instance Metrics Calculator Agent** agent.



Agent Information

Agent Type: Instance Metrics Calculator Agent

Number of Agents: 1

Agent Server: RELATIVITYDEVVM

Run interval: The interval, in seconds, at which this agent checks for available jobs. 10

Logging level of event details: [✓ Log critical errors only] Log warnings and errors Log all messages

Status

Enabled: [✓ Yes] No

Other

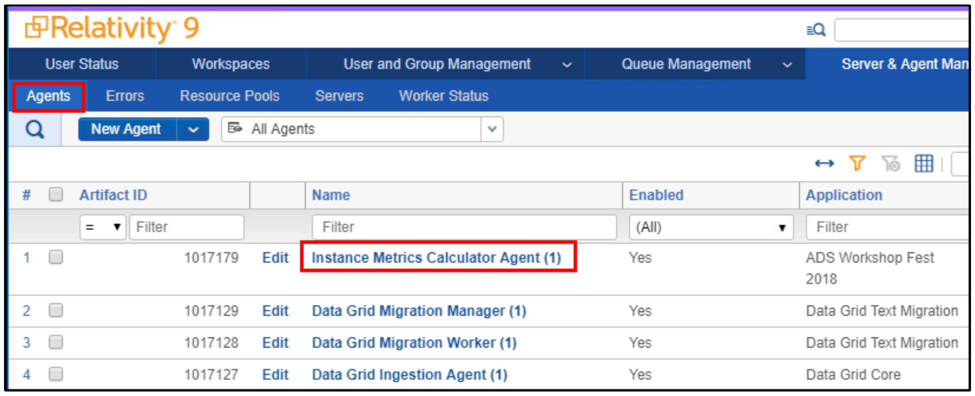
Keywords:

Notes:

Not secure | relativitydevvm/Relativity/Controls/PickSingle.aspx?idValueTextBoxID=_editTemplate__kCuraScr

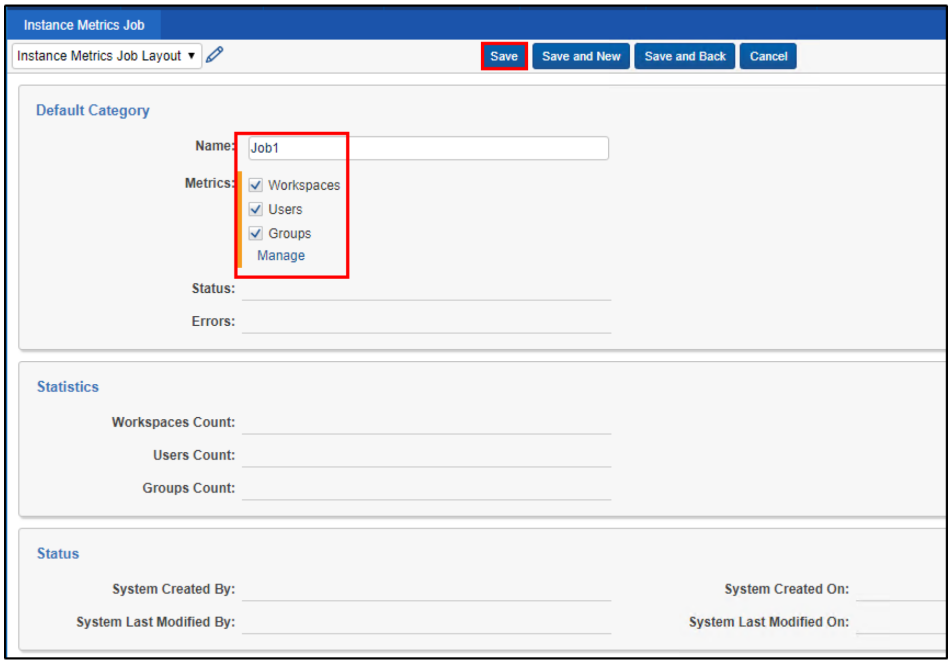
Items 1 - 1 (of 1)

Name	Company Name	Application Name
(All)	(All)	ads
Instance Metrics Calculator Agent		ADS Workshop Fest 2018



#	Artifact ID		Name	Enabled	Application
	= Filter		Filter	(All)	Filter
1	1017179	Edit	Instance Metrics Calculator Agent (1)	Yes	ADS Workshop Fest 2018
2	1017129	Edit	Data Grid Migration Manager (1)	Yes	Data Grid Text Migration
3	1017128	Edit	Data Grid Migration Worker (1)	Yes	Data Grid Text Migration
4	1017127	Edit	Data Grid Ingestion Agent (1)	Yes	Data Grid Core

2. Now navigate the **Instance Metrics Job** tab in **ADS Workshop Fest 2018** workspace.
3. Create a new **Instance Metrics Job** and click on the **Save** button.



Default Category

Name: Job1

Metrics: ✓ Workspaces ✓ Users ✓ Groups Manage

Status:

Errors:

Statistics

Workspaces Count:

Users Count:

Groups Count:

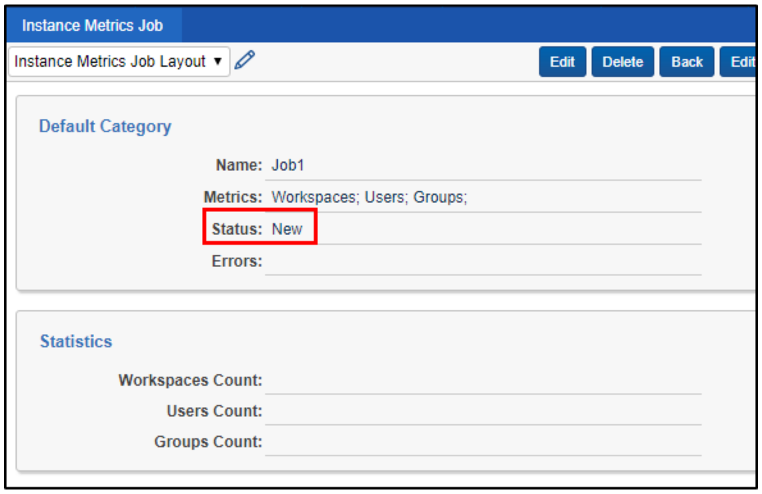
Status

System Created By:

System Created On:

System Last Modified By:

System Last Modified On:



Default Category

Name: Job1

Metrics: Workspaces; Users; Groups;

Status: New

Errors:

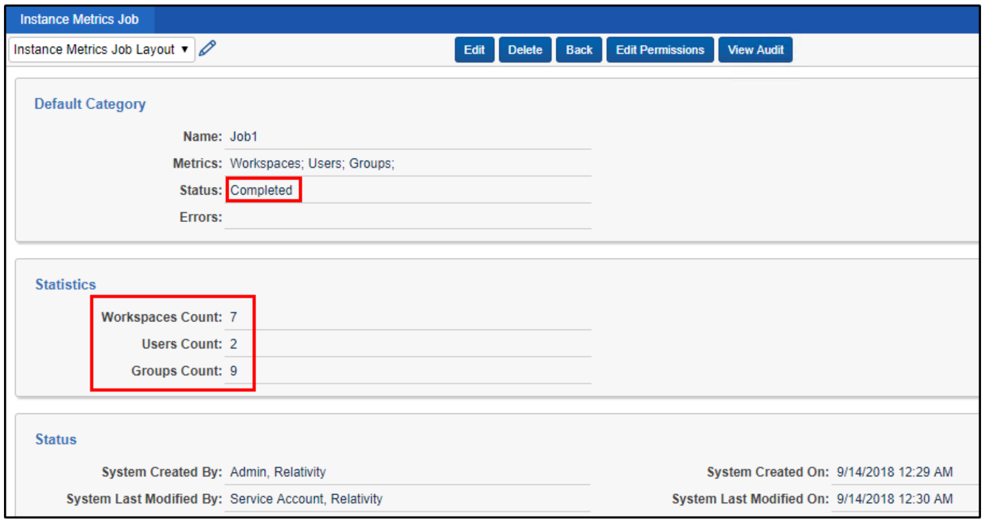
Statistics

Workspaces Count:

Users Count:

Groups Count:

4. The **Instance Metrics Calculator Agent** which is running every 10 seconds will pick up the job and calculate the metrics and save it to the job.
5. Keeping refreshing the windows every few seconds until the job status is set to **Completed**.



Default Category

Name:	Job1
Metrics:	Workspaces; Users; Groups;
Status:	Completed
Errors:	

Statistics

Workspaces Count:	7
Users Count:	2
Groups Count:	9

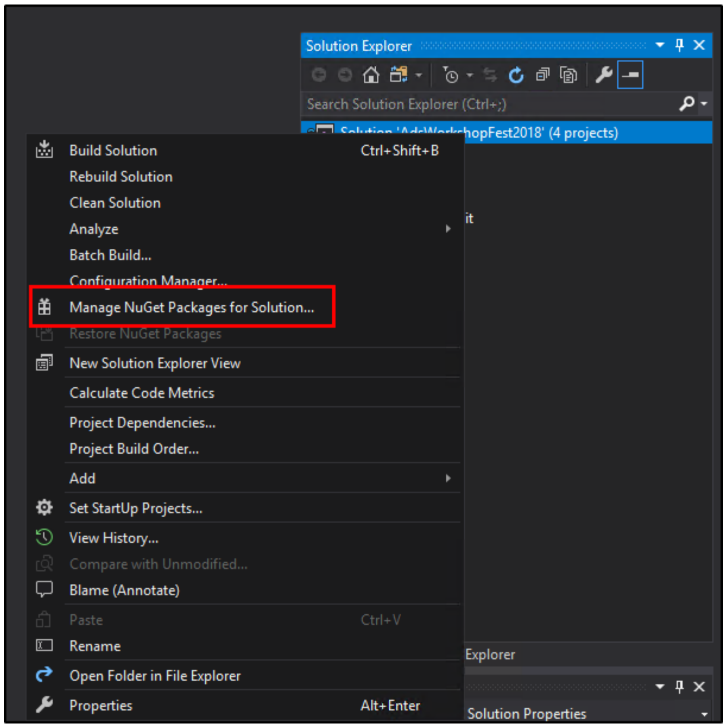
Status

System Created By:	Admin, Relativity	System Created On:	9/14/2018 12:29 AM
System Last Modified By:	Service Account, Relativity	System Last Modified On:	9/14/2018 12:30 AM

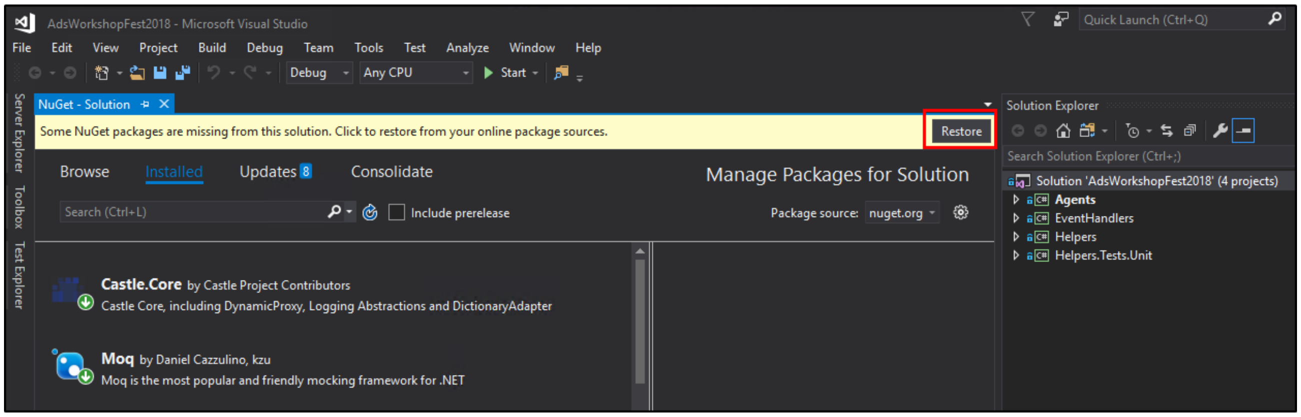
5 Unit Tests

5.1 Writing Unit Tests

1. Navigate to the **E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\Project** folder and open **AdsWorkshopFest2018.sln** file.
2. In this solution, you will find 4 projects.
    a. Agents
    b. EventHandlers
    c. Helpers
    d. Helpers.Tests.Unit
3. In this section we will be working in the **Helpers.Tests.Unit** project.
4. In **Helpers** project, there is an **RsapiHelper.cs** file which contains RSAPI calls used in the sample application.
5. Our goal for this section is to write couple unit tests for the **RetrieveJobsInWorkspaceWithStatus** in **RsapiHelper** class.
6. Right click on the solution and select **Manage NuGet Packages for Solution** option.



7. Next click the **Restore** button.

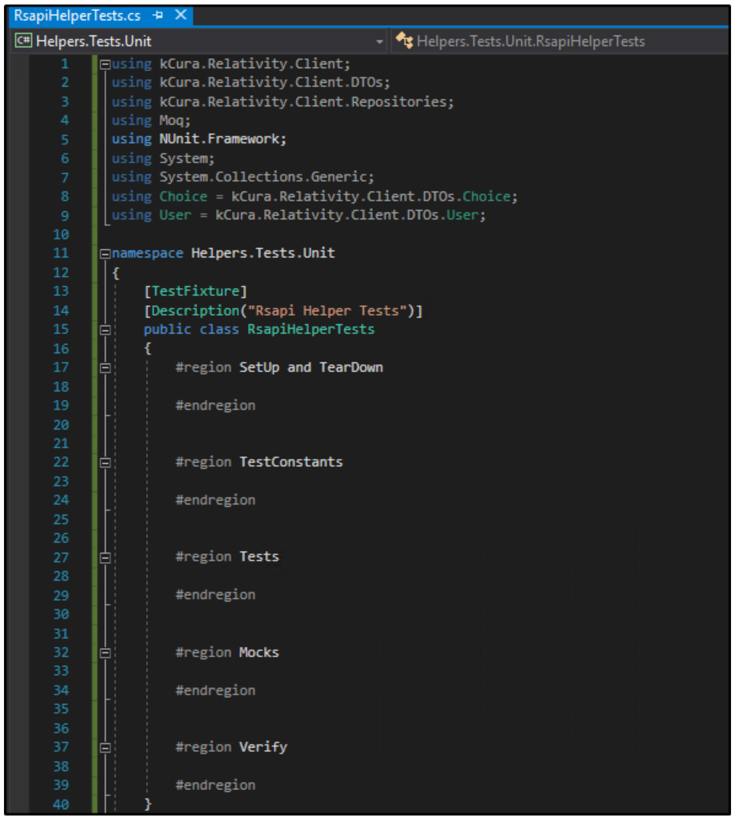

AdsWorkshopFest2018 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze Window Help
Debug Any CPU Start
Quick Launch (Ctrl+Q)
NuGet - Solution
Restore
Browse Installed Updates 8 Consolidate
Manage Packages for Solution
Search (Ctrl+L)
Include prerelease
Package source: nuget.org
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'AdsWorkshopFest2018' (4 projects)
Agents
EventHandlers
Helpers
Helpers.Tests.Unit
Server Explorer Toolbox Test Explorer

Some NuGet packages are missing from this solution. Click to restore from your online package sources.

Castle.Core by Castle Project Contributors
Castle Core, including DynamicProxy, Logging Abstractions and DictionaryAdapter

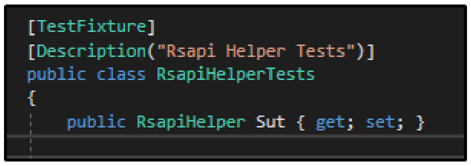
Moq by Daniel Cazzulino, kzu
Moq is the most popular and friendly mocking framework for .NET

8. Now all the NuGet packages in the solution should be restored.
9. Right click on the **Helpers.Tests.Unit** project and add a new class named **RsapiHelperTests.cs**.
10. Overwrite the default class structure with the Unit Test structure we will be using.
11. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\1_Structure.txt"** file replacing the entire **RsapiHelperTests.cs** class.

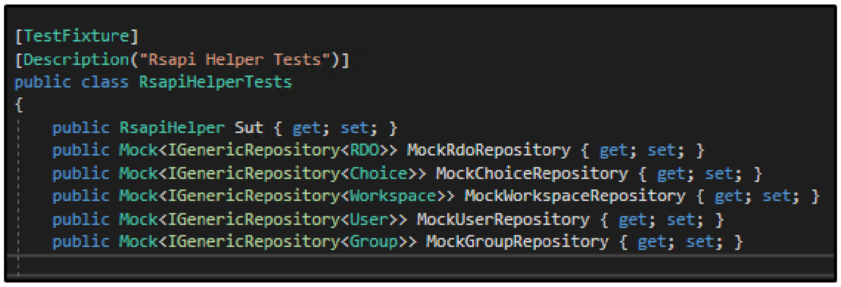


```
using kCura.Relativity.Client;
using kCura.Relativity.Client.DTOs;
using kCura.Relativity.Client.Repositories;
using Moq;
using NUnit.Framework;
using System;
using System.Collections.Generic;
using Choice = kCura.Relativity.Client.DTOs.Choice;
using User = kCura.Relativity.Client.DTOs.User;

namespace Helpers.Tests.Unit
{
    [TestFixture]
    [Description("Rsapi Helper Tests")]
    public class RsapiHelperTests
    {
        #region SetUp and TearDown

        #endregion


        #region TestConstants

        #endregion


        #region Tests

        #endregion


        #region Mocks

        #endregion


        #region Verify

        #endregion
    }
```

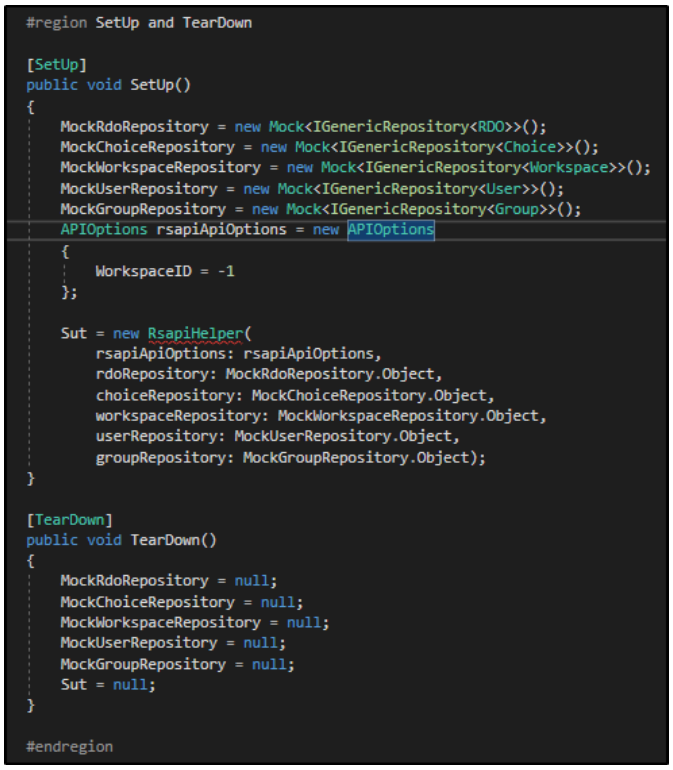
12. Add the RsapiHelper class reference which we will be our System under Test (SUT) for which we are writing unit tests.
13. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\2_System_Under_Test.txt"** file at the beginning of the **RsapiHelperTests.cs** class.


```
[TestFixture]
[Description("Rsapi Helper Tests")]
public class RsapiHelperTests
{
    public RsapiHelper Sut { get; set; }
```

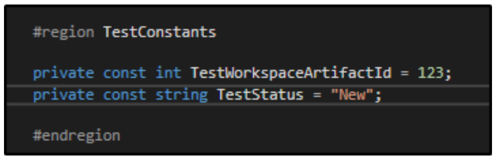
14. Next add the Mocks for various RDO repositories which are used in the Relativity application.
15. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\3_Mocks.txt"** file at the beginning of the **RsapiHelperTests.cs** class.


```
[TestFixture]
[Description("Rsapi Helper Tests")]
public class RsapiHelperTests
{
    public RsapiHelper Sut { get; set; }
    public Mock<IGenericRepository<RDO>> MockRdoRepository { get; set; }
    public Mock<IGenericRepository<Choice>> MockChoiceRepository { get; set; }
    public Mock<IGenericRepository<Workspace>> MockWorkspaceRepository { get; set; }
    public Mock<IGenericRepository<User>> MockUserRepository { get; set; }
    public Mock<IGenericRepository<Group>> MockGroupRepository { get; set; }
```

16. Next add the SetUp and TearDown methods which run once before and after all the tests execution.
17. The SetUp method is utilized to inititalize any data needed by tests.
18. The TearDown method is utilized to clean up the data used by the tests.
19. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\4_SetUp_And_TearDown.txt"** file under the **SetUp and TearDown** region in the **RsapiHelperTests.cs** class.

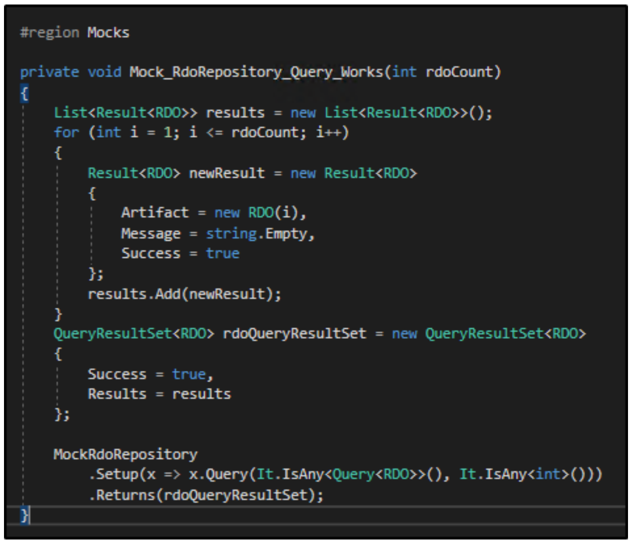

```
#region SetUp and TearDown

[SetUp]
public void SetUp()
{
    MockRdoRepository = new Mock<IGenericRepository<RDO>>();
    MockChoiceRepository = new Mock<IGenericRepository<Choice>>();
    MockWorkspaceRepository = new Mock<IGenericRepository<Workspace>>();
    MockUserRepository = new Mock<IGenericRepository<User>>();
    MockGroupRepository = new Mock<IGenericRepository<Group>>();
    APIOptions rsapiApiOptions = new APIOptions
    {
        WorkspaceID = -1
    };

    Sut = new RsapiHelper(
        rsapiApiOptions: rsapiApiOptions,
        rdoRepository: MockRdoRepository.Object,
        choiceRepository: MockChoiceRepository.Object,
        workspaceRepository: MockWorkspaceRepository.Object,
        userRepository: MockUserRepository.Object,
        groupRepository: MockGroupRepository.Object);
}

[TearDown]
public void TearDown()
{
    MockRdoRepository = null;
    MockChoiceRepository = null;
    MockWorkspaceRepository = null;
    MockUserRepository = null;
    MockGroupRepository = null;
    Sut = null;
}

#endregion
```

20. Next add the Test constants used in the Tests.
21. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\5_Test_Constants.txt"** file under the **TestConstants** region in the **RsapiHelperTests.cs** class.



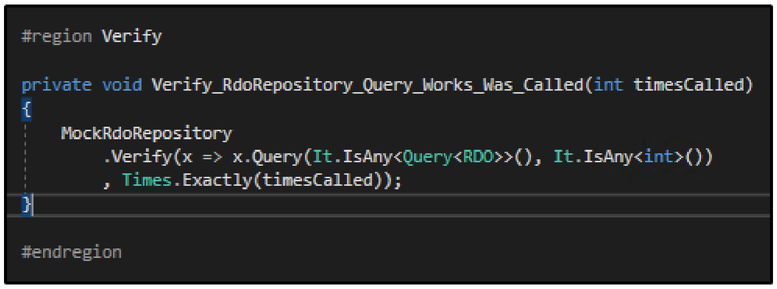
```
#region TestConstants

private const int TestWorkspaceArtifactId = 123;
private const string TestStatus = "New";

#endregion
```

22. Next add the Mock Stubs for an RSAPI call used in the Tests. Here we will be stubbing out the Query method on the RDO repository in RSAPI.
23. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\6_Mock_Stub_Golden_Flow.txt"** file under the **Mocks** region in the **RsapiHelperTests.cs** class.



```
#region Mocks

private void Mock_RdoRepository_Query_Works(int rdoCount)
{
    List<Result<RDO>> results = new List<Result<RDO>>();
    for (int i = 1; i <= rdoCount; i++)
    {
        Result<RDO> newResult = new Result<RDO>
        {
            Artifact = new RDO(i),
            Message = string.Empty,
            Success = true
        };
        results.Add(newResult);
    }
    QueryResultSet<RDO> rdoQueryResultSet = new QueryResultSet<RDO>
    {
        Success = true,
        Results = results
    };

    MockRdoRepository
        .Setup(x => x.Query(It.IsAny<Query<RDO>>(), It.IsAny<int>()))
        .Returns(rdoQueryResultSet);
}
```

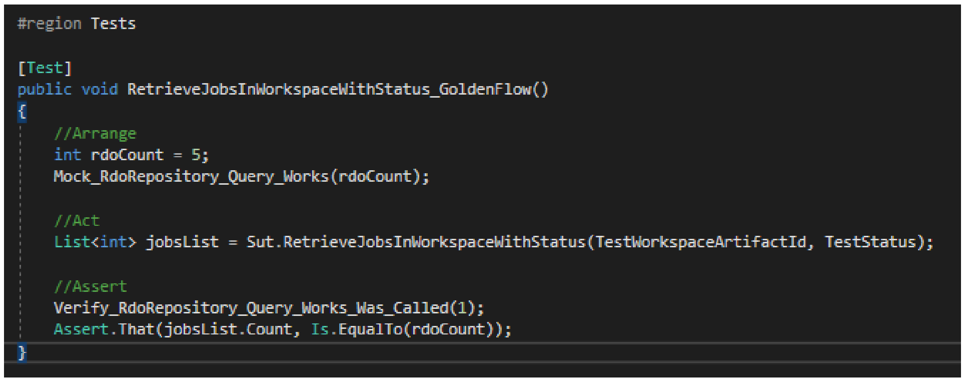
24. Next add the Verify method for the previously added Mock Stubs. Use this method we can verify if our stub was called during our test execution.
25. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\7_Mock_Stub_Golden_Flow_Verify.txt"** file under the **Verify** region in the **RsapiHelperTests.cs** class.



```
#region Verify

private void Verify_RdoRepository_Query_Works_Was_Called(int timesCalled)
{
    MockRdoRepository
        .Verify(x => x.Query(It.IsAny<Query<RDO>>(), It.IsAny<int>())
        , Times.Exactly(timesCalled));
}

#endregion
```

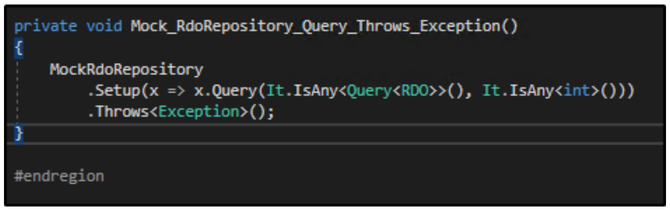
26. Next add the Golden flow test.

27. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\8_UnitTest_Golden_Flow.txt"** file under the **Tests** region in the **RsapiHelperTests.cs** class.



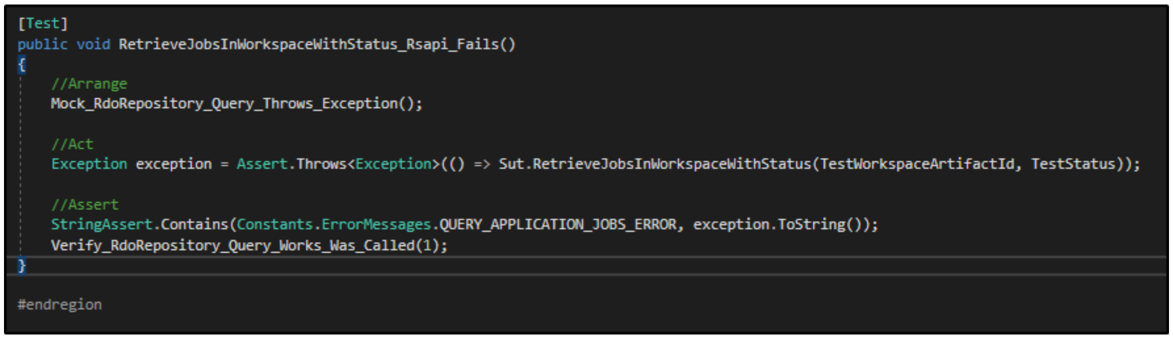
```
#region Tests

[Test]
public void RetrieveJobsInWorkspaceWithStatus_GoldenFlow()
{
    //Arrange
    int rdoCount = 5;
    Mock_RdoRepository_Query_Works(rdoCount);

    //Act
    List<int> jobsList = Sut.RetrieveJobsInWorkspaceWithStatus(TestWorkspaceArtifactId, TestStatus);

    //Assert
    Verify_RdoRepository_Query_Works_Was_Called(1);
    Assert.That(jobsList.Count, Is.EqualTo(rdoCount));
}
```

28. Next add the Mock Stubs for an RSAPI call to simulate an exception in the RSAPI call.
29. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\9_Mock_Stub_Rsapi_Fails.txt"** file under the **Mocks** region in the **RsapiHelperTests.cs** class.



```
private void Mock_RdoRepository_Query_Throws_Exception()
{
    MockRdoRepository
        .Setup(x => x.Query(It.IsAny<Query<RDO>>(), It.IsAny<int>()))
        .Throws<Exception>();
}

#endregion
```

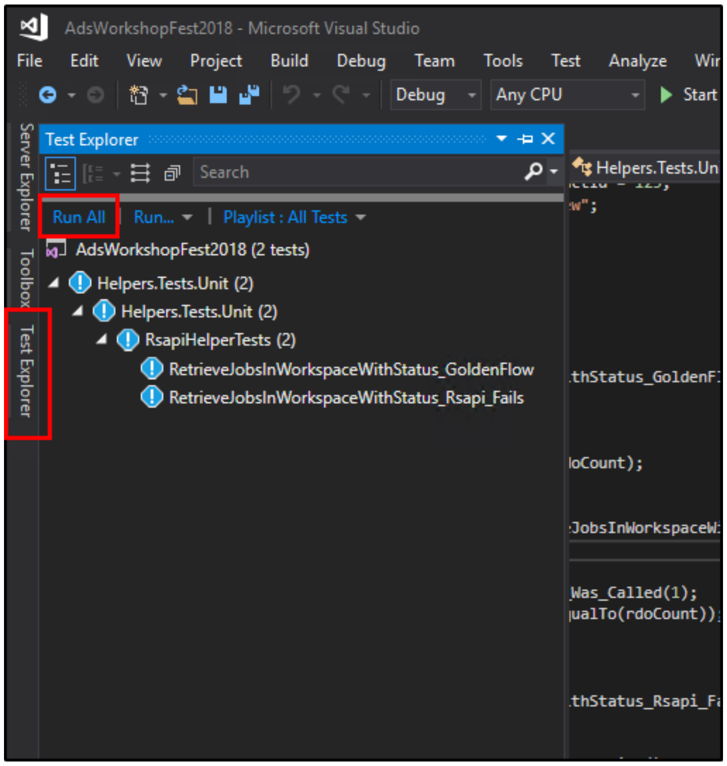
30. Next add the RSAPI Failure test.
31. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\3_UnitTests\TextFiles\10_UnitTest_Rsapi_Fails.txt"** file under the **Tests** region in the **RsapiHelperTests.cs** class.



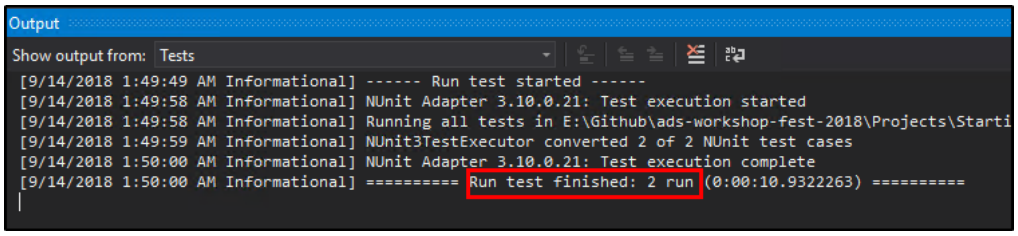
```
[Test]
public void RetrieveJobsInWorkspaceWithStatus_Rsapi_Fails()
{
    //Arrange
    Mock_RdoRepository_Query_Throws_Exception();

    //Act
    Exception exception = Assert.Throws<Exception>(() => Sut.RetrieveJobsInWorkspaceWithStatus(TestWorkspaceArtifactId, TestStatus));

    //Assert
    StringAssert.Contains(Constants.ErrorMessages.QUERY_APPLICATION_JOBS_ERROR, exception.ToString());
    Verify_RdoRepository_Query_Works_Was_Called(1);
}

#endregion
```

5.2 Running Unit Tests

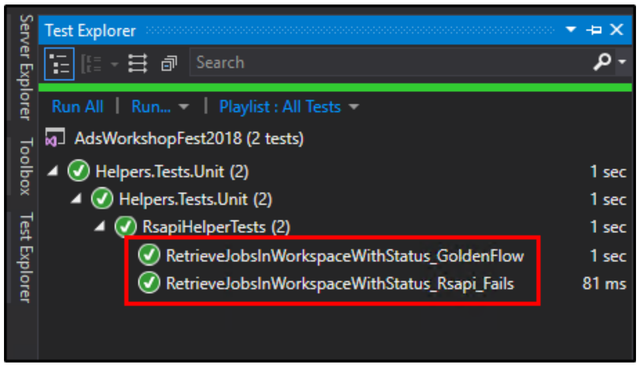
1. Build the solution to successfully build all the projects.
2. Open the **Test Explorer** by click on the **Test Explorer** tab on the left side bar of Visual Studio.
3. Click on the **Run All** link to run the 2 unit tests we just created.


AdsWorkshopFest2018 - Microsoft Visual Studio
File Edit View Project Build Debug Team Tools Test Analyze
Debug
Any CPU
Start
Server Explorer
Toolbox
Test Explorer
Search
Run All
Run...
Playlist : All Tests
AdsWorkshopFest2018 (2 tests)
Helpers.Tests.Unit (2)
Helpers.Tests.Unit (2)
RsapiHelperTests (2)
RetrieveJobsInWorkspaceWithStatus_GoldenFlow
RetrieveJobsInWorkspaceWithStatus_Rsapi_Fails
Helpers.Tests.Un

4. On successful run, click on the **Test Explorer** to see the status of the tests. Both the tests should be green.



```
[9/14/2018 1:49:49 AM Informational] ------ Run test started ------
[9/14/2018 1:49:58 AM Informational] NUnit Adapter 3.10.0.21: Test execution started
[9/14/2018 1:49:58 AM Informational] Running all tests in E:\Github\ads-workshop-fest-2018\Projects\Starti
[9/14/2018 1:49:59 AM Informational] NUnit3TestExecutor converted 2 of 2 NUnit test cases
[9/14/2018 1:50:00 AM Informational] NUnit Adapter 3.10.0.21: Test execution complete
[9/14/2018 1:50:00 AM Informational] ========== Run test finished: 2 run (0:00:10.9322263) ==========
```

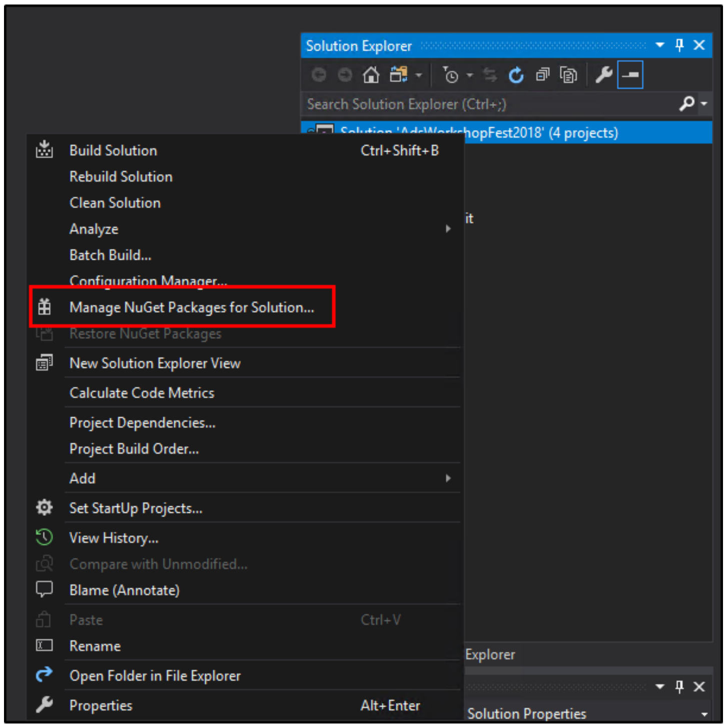


```
AdsWorkshopFest2018 (2 tests)
  Helpers.Tests.Unit (2)                                    1 sec
    Helpers.Tests.Unit (2)                                  1 sec
      RsapiHelperTests (2)                                  1 sec
        RetrieveJobsInWorkspaceWithStatus_GoldenFlow        1 sec
        RetrieveJobsInWorkspaceWithStatus_Rsapi_Fails       81 ms
```

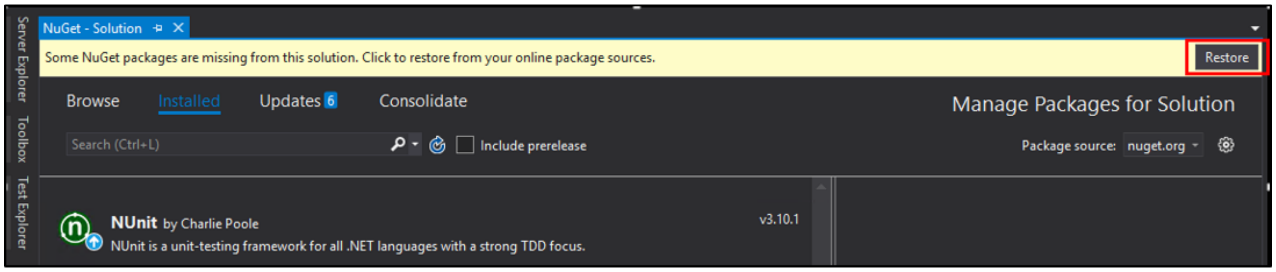
6 Integration Tests

6.1 Writing Integration Tests

1. Navigate to the **E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project** folder and open **AdsWorkshopFest2018.sln** file.
2. In this solution, you will find 4 projects.
    a. Agents
    b. Agents.Tests.Integration
    c. EventHandlers
    d. Helpers
3. In this section we will be working in the **Agents.Tests.Integration** project.
4. Right click on the solution and select **Manage NuGet Packages for Solution** option.



5. Next click the **Restore** button.

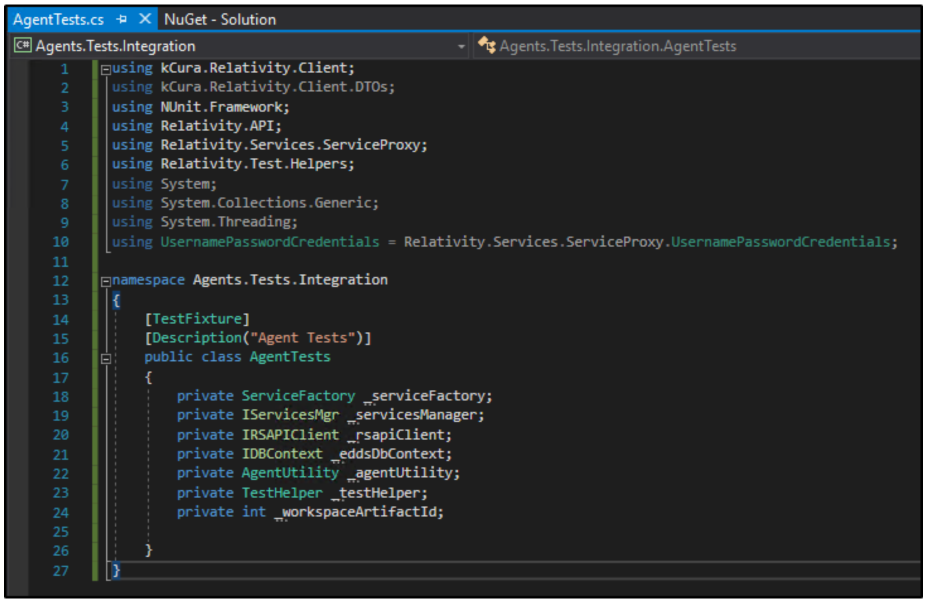


Some NuGet packages are missing from this solution. Click to restore from your online package sources.

NUnit is a unit-testing framework for all .NET languages with a strong TDD focus.

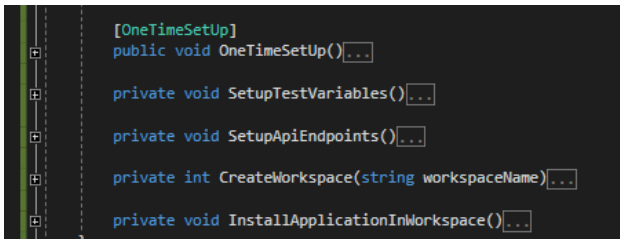
6. Now all the NuGet packages in the solution should be restored.
7. Right click on the **Agents.Tests.Integration** project and add a new class named **AgentsTests.cs**.

8. Overwrite the default class structure with the Integration Test structure we will be using.
9. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\TextFiles\1_Structure.txt"** file replacing the entire **AgentsTests.cs** class.



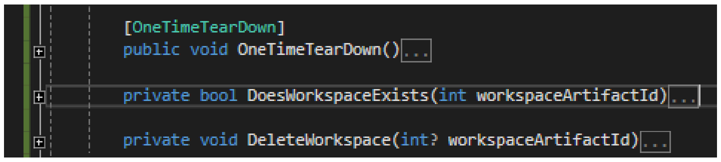
```
using kCura.Relativity.Client;
using kCura.Relativity.Client.DTOs;
using NUnit.Framework;
using Relativity.API;
using Relativity.Services.ServiceProxy;
using Relativity.Test.Helpers;
using System;
using System.Collections.Generic;
using System.Threading;
using UsernamePasswordCredentials = Relativity.Services.ServiceProxy.UsernamePasswordCredentials;

namespace Agents.Tests.Integration
{
    [TestFixture]
    [Description("Agent Tests")]
    public class AgentTests
    {
        private ServiceFactory _serviceFactory;
        private IServicesMgr _servicesManager;
        private IRSAPIClient _rsapiClient;
        private IDBContext _eddsDbContext;
        private AgentUtility _agentUtility;
        private TestHelper _testHelper;
        private int _workspaceArtifactId;

    }
}
```

10. Add the **OneTimeSetUp** method and its dependent methods to the **AgentsTests** class.
11. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\TextFiles\2_OneTimeSetUp.txt"** to the **AgentsTests** class.



```
[OneTimeSetUp]
public void OneTimeSetUp()...

private void SetupTestVariables()...

private void SetupApiEndpoints()...

private int CreateWorkspace(string workspaceName)...

private void InstallApplicationInWorkspace()...
```

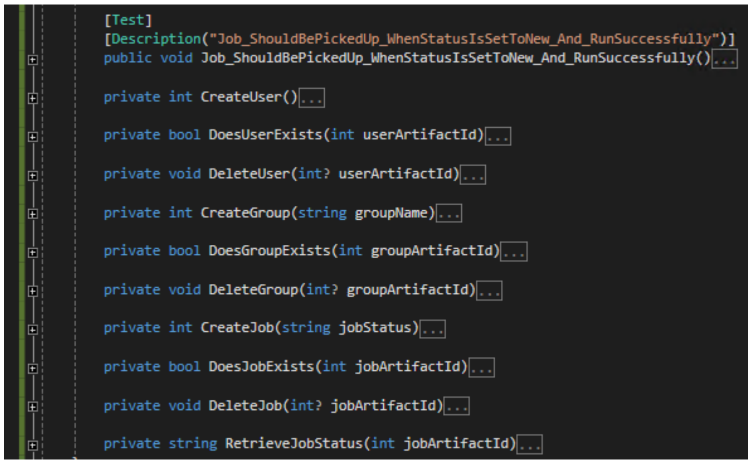
12. Add the **OneTimeTearDown** method and its dependent methods to the **AgentsTests** class.
13. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\TextFiles\3_OneTimeTearDown.txt"** to the **AgentsTests** class.



```
[OneTimeTearDown]
public void OneTimeTearDown()...

private bool DoesWorkspaceExists(int workspaceArtifactId)...

private void DeleteWorkspace(int? workspaceArtifactId)...
```

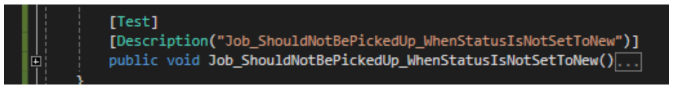
14. Add the **Job_ShouldBePickedUp_WhenStatusIsSetToNew_And_RunSuccessfully** integration test method and its dependent methods to the **AgentsTests** class.

15. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\TextFiles\4_Job_ShouldBePickedUp_WhenStatusIsSetToNew_And_RunSuccessfully.txt"** to the **AgentsTests** class.



```
[Test]
[Description("Job_ShouldBePickedUp_WhenStatusIsSetToNew_And_RunSuccessfully")]
public void Job_ShouldBePickedUp_WhenStatusIsSetToNew_And_RunSuccessfully()...

private int CreateUser()...

private bool DoesUserExists(int userArtifactId)...

private void DeleteUser(int? userArtifactId)...

private int CreateGroup(string groupName)...

private bool DoesGroupExists(int groupArtifactId)...

private void DeleteGroup(int? groupArtifactId)...

private int CreateJob(string jobStatus)...

private bool DoesJobExists(int jobArtifactId)...

private void DeleteJob(int? jobArtifactId)...

private string RetrieveJobStatus(int jobArtifactId)...
```

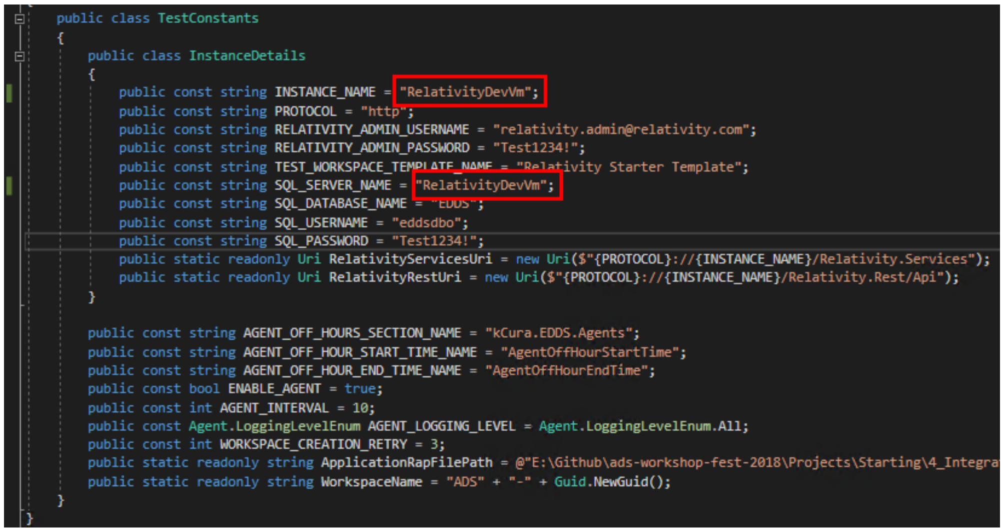
16. Add the **Job_ShouldNotBePickedUp_WhenStatusIsNotSetToNew** integration test method and its dependent methods to the **AgentsTests** class.
17. Copy and paste code from **"E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\TextFiles\5_Job_ShouldNotBePickedUp_WhenStatusIsNotSetToNew.txt"** to the **AgentsTests** class.



```
[Test]
[Description("Job_ShouldNotBePickedUp_WhenStatusIsNotSetToNew")]
public void Job_ShouldNotBePickedUp_WhenStatusIsNotSetToNew()...
```

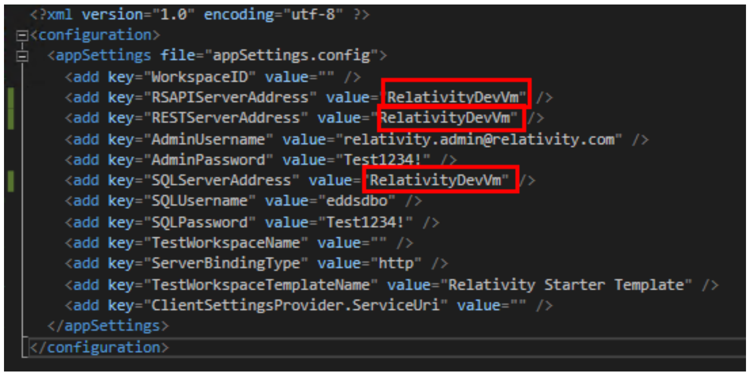
6.2 Running Integration Tests

1. Open **TestConstants.cs** file in **Agents.Tests.Integration** project.
2. Replace the **192.168.137.95** IP address with the DevVM machine name (**RelativityDevVm**).



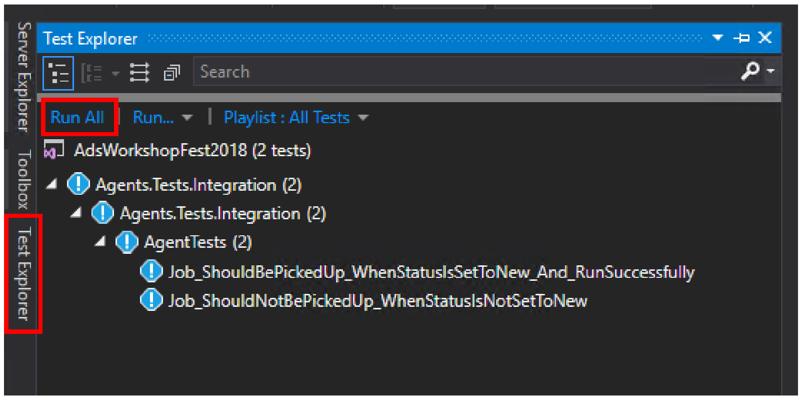
```
public class TestConstants
{
    public class InstanceDetails
    {
        public const string INSTANCE_NAME = "RelativityDevVm";
        public const string PROTOCOL = "http";
        public const string RELATIVITY_ADMIN_USERNAME = "relativity.admin@relativity.com";
        public const string RELATIVITY_ADMIN_PASSWORD = "Test1234!";
        public const string TEST_WORKSPACE_TEMPLATE_NAME = "Relativity Starter Template";
        public const string SQL_SERVER_NAME = "RelativityDevVm";
        public const string SQL_DATABASE_NAME = "EDDS";
        public const string SQL_USERNAME = "eddsdbo";
        public const string SQL_PASSWORD = "Test1234!";
        public static readonly Uri RelativityServicesUri = new Uri($"{PROTOCOL}://{INSTANCE_NAME}/Relativity.Services");
        public static readonly Uri RelativityRestUri = new Uri($"{PROTOCOL}://{INSTANCE_NAME}/Relativity.Rest/Api");
    }

    public const string AGENT_OFF_HOURS_SECTION_NAME = "kCura.EDDS.Agents";
    public const string AGENT_OFF_HOUR_START_TIME_NAME = "AgentOffHourStartTime";
    public const string AGENT_OFF_HOUR_END_TIME_NAME = "AgentOffHourEndTime";
    public const bool ENABLE_AGENT = true;
    public const int AGENT_INTERVAL = 10;
    public const Agent.LoggingLevelEnum AGENT_LOGGING_LEVEL = Agent.LoggingLevelEnum.All;
    public const int WORKSPACE_CREATION_RETRY = 3;
    public static readonly string ApplicationRapFilePath = @"E:\Github\ads-workshop-fest-2018\Projects\Starting\4_Integra
    public static readonly string WorkspaceName = "ADS" + "-" + Guid.NewGuid();
}
}
```

3. Open **App.config** file in **Agents.Tests.Integration** project.
4. Replace the **192.168.137.95** IP address with the DevVM machine name (**RelativityDevVm**).



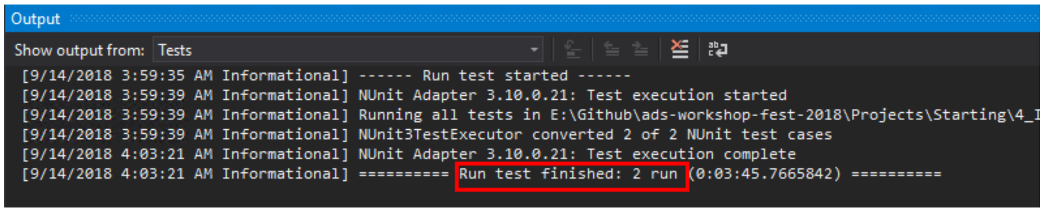
```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <appSettings file="appSettings.config">
    <add key="WorkspaceID" value="" />
    <add key="RSAPIServerAddress" value="RelativityDevVm" />
    <add key="RESTServerAddress" value="RelativityDevVm" />
    <add key="AdminUsername" value="relativity.admin@relativity.com" />
    <add key="AdminPassword" value="Test1234!" />
    <add key="SQLServerAddress" value="RelativityDevVm" />
    <add key="SQLUsername" value="eddsdbo" />
    <add key="SQLPassword" value="Test1234!" />
    <add key="TestWorkspaceName" value="" />
    <add key="ServerBindingType" value="http" />
    <add key="TestWorkspaceTemplateName" value="Relativity Starter Template" />
    <add key="ClientSettingsProvider.ServiceUri" value="" />
  </appSettings>
</configuration>
```

5. Build the solution to successfully build all the projects.
6. Open the **Test Explorer** by click on the **Test Explorer** tab on the left side bar of Visual Studio.
7. Click on the **Run All** link to run the 2 unit tests we just created.

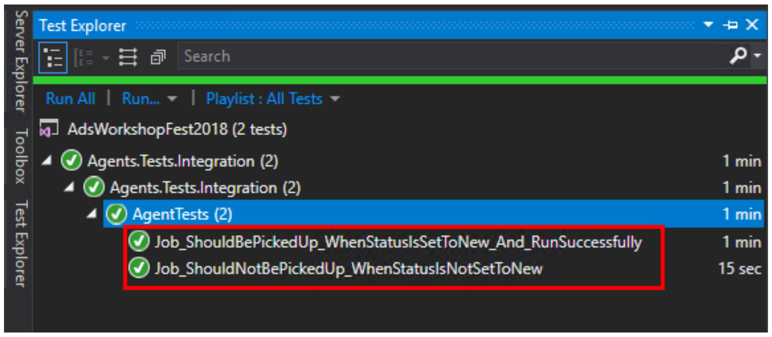


```
AdsWorkshopFest2018 (2 tests)
  Agents.Tests.Integration (2)
    Agents.Tests.Integration (2)
      AgentTests (2)
        Job_ShouldBePickedUp_WhenStatusIsSetToNew_And_RunSuccessfully
        Job_ShouldNotBePickedUp_WhenStatusIsNotSetToNew
```

8. On successful run, click on the **Test Explorer** to see the status of the tests. Both the tests should be green.



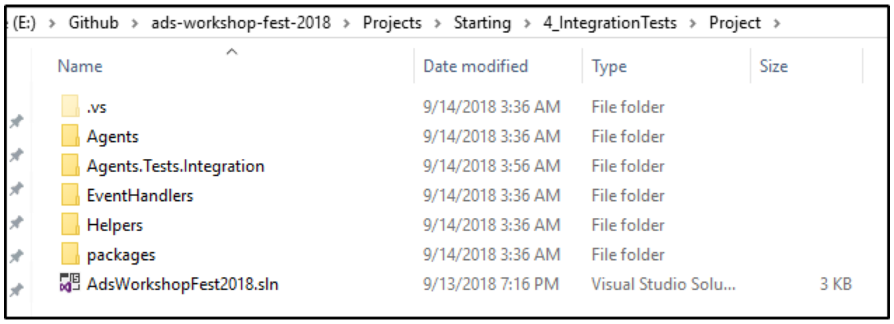
```
[9/14/2018 3:59:35 AM Informational] ------ Run test started ------
[9/14/2018 3:59:39 AM Informational] NUnit Adapter 3.10.0.21: Test execution started
[9/14/2018 3:59:39 AM Informational] Running all tests in E:\Github\ads-workshop-fest-2018\Projects\Starting\4_I
[9/14/2018 3:59:39 AM Informational] NUnit3TestExecutor converted 2 of 2 NUnit test cases
[9/14/2018 4:03:21 AM Informational] NUnit Adapter 3.10.0.21: Test execution complete
[9/14/2018 4:03:21 AM Informational] ========== Run test finished: 2 run (0:03:45.7665842) ==========
```



```
AdsWorkshopFest2018 (2 tests)
  Agents.Tests.Integration (2)                                          1 min
    Agents.Tests.Integration (2)                                        1 min
      AgentTests (2)                                                    1 min
        Job_ShouldBePickedUp_WhenStatusIsSetToNew_And_RunSuccessfully   1 min
        Job_ShouldNotBePickedUp_WhenStatusIsNotSetToNew                 15 sec
```

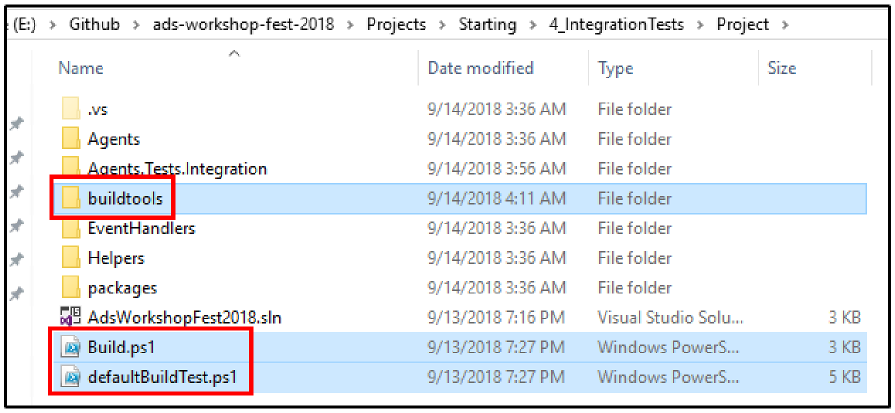
6.2 Running Integration Tests as CI

1. Open the **E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\CI_Scripts** folder
2. Here you will find 3 files and 1 folder which we will be using to run Integration Tests as a CI (Continuous Integration) process.
3. Copy all the files and folders from the CI_Scripts folder to **E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project** folder.



(E:) > Github > ads-workshop-fest-2018 > Projects > Starting > 4_IntegrationTests > Project >

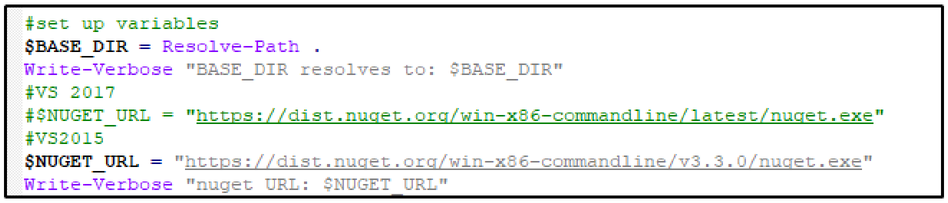
Name	Date modified	Type	Size
.vs	9/14/2018 3:36 AM	File folder	
Agents	9/14/2018 3:36 AM	File folder	
Agents.Tests.Integration	9/14/2018 3:56 AM	File folder	
EventHandlers	9/14/2018 3:36 AM	File folder	
Helpers	9/14/2018 3:36 AM	File folder	
packages	9/14/2018 3:36 AM	File folder	
AdsWorkshopFest2018.sln	9/13/2018 7:16 PM	Visual Studio Solu...	3 KB



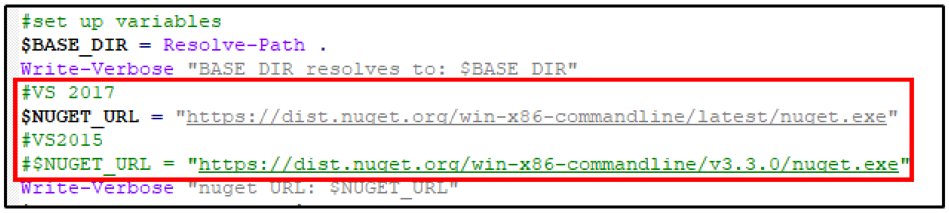
(E:) > Github > ads-workshop-fest-2018 > Projects > Starting > 4_IntegrationTests > Project >

Name	Date modified	Type	Size
.vs	9/14/2018 3:36 AM	File folder	
Agents	9/14/2018 3:36 AM	File folder	
Agents.Tests.Integration	9/14/2018 3:56 AM	File folder	
buildtools	9/14/2018 4:11 AM	File folder	
EventHandlers	9/14/2018 3:36 AM	File folder	
Helpers	9/14/2018 3:36 AM	File folder	
packages	9/14/2018 3:36 AM	File folder	
AdsWorkshopFest2018.sln	9/13/2018 7:16 PM	Visual Studio Solu...	3 KB
Build.ps1	9/13/2018 7:27 PM	Windows PowerS...	3 KB
defaultBuildTest.ps1	9/13/2018 7:27 PM	Windows PowerS...	5 KB

4. Open the **Build.ps1** script and update the **NuGet_URL** to use the one which is compatible with Visual Studio 2017.



```
#set up variables
$BASE_DIR = Resolve-Path .
Write-Verbose "BASE_DIR resolves to: $BASE_DIR"
#VS 2017
#$NUGET_URL = "https://dist.nuget.org/win-x86-commandline/latest/nuget.exe"
#VS2015
$NUGET_URL = "https://dist.nuget.org/win-x86-commandline/v3.3.0/nuget.exe"
Write-Verbose "nuget URL: $NUGET_URL"
```

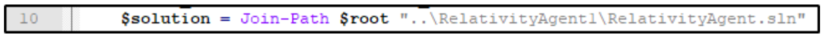


```
#set up variables
$BASE_DIR = Resolve-Path .
Write-Verbose "BASE_DIR resolves to: $BASE_DIR"
#VS 2017
$NUGET_URL = "https://dist.nuget.org/win-x86-commandline/latest/nuget.exe"
#VS2015
#$NUGET_URL = "https://dist.nuget.org/win-x86-commandline/v3.3.0/nuget.exe"
Write-Verbose "nuget URL: $NUGET_URL"
```

5. Open the **defaultBuildTest.ps1** script and update the **project paths.**
   a. Replace line 10 with the following code.

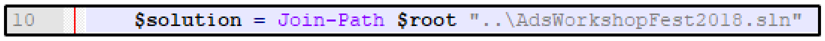
      Code:
      $solution = Join-Path $root "..\AdsWorkshopFest2018.sln"

      Before:



```
10      $solution = Join-Path $root "..\RelativityAgent1\RelativityAgent.sln"
```

      After:

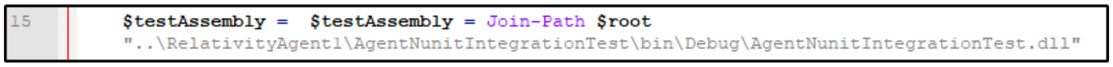


```
10      $solution = Join-Path $root "..\AdsWorkshopFest2018.sln"
```

   b. Replace line 15 with the following code.

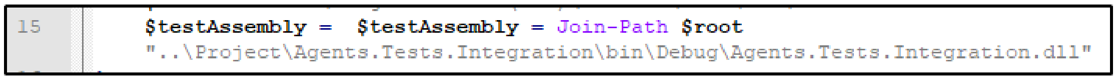
      Code:
      $testAssembly = $testAssembly = Join-Path $root "..\Project\Agents.Tests.Integration\bin\Debug\Agents.Tests.Integration.dll"

      Before:



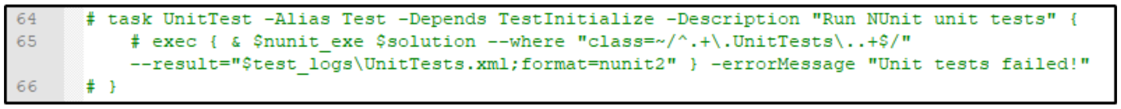
```
15      $testAssembly =  $testAssembly = Join-Path $root
        "..\RelativityAgent1\AgentNunitIntegrationTest\bin\Debug\AgentNunitIntegrationTest.dll"
```

      After:



```
15      $testAssembly =  $testAssembly = Join-Path $root
        "..\Project\Agents.Tests.Integration\bin\Debug\Agents.Tests.Integration.dll"
```

   c. Comment lines 64 – 66.

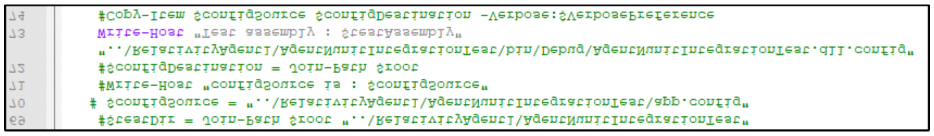


```
64  # task UnitTest -Alias Test -Depends TestInitialize -Description "Run NUnit unit tests" {
65      # exec { & $nunit_exe $solution --where "class=~/^.+\.UnitTests\..+$/"
        --result="$test_logs\UnitTests.xml;format=nunit2" } -errorMessage "Unit tests failed!"
66  # }
```

   d. Replace lines 69 - 74 with the following code.

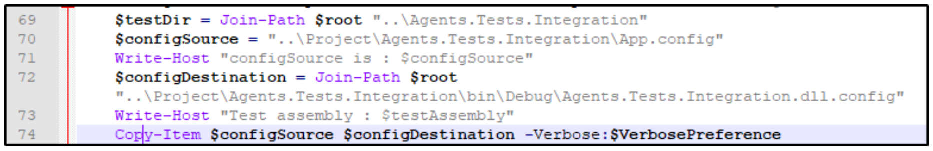
      Code:
      $testDir = Join-Path $root "..\Agents.Tests.Integration"
      $testDir = Join-Path $root "..\Agents.Tests.Integration"
      $configSource = "..\Project\Agents.Tests.Integration\App.config"
      Write-Host "configSource is : $configSource"
      $configDestination = Join-Path $root "..\Project\Agents.Tests.Integration\bin\Debug\Agents.Tests.Integration.dll.config"
      Write-Host "Test assembly : $testAssembly"
      Copy-Item $configSource $configDestination -Verbose:$VerbosePreference

      Before:



```
69      #$testDir = Join-Path $root "..\RelativityAgent1\AgentNunitIntegrationTest"
70     # $configSource = "..\RelativityAgent1\AgentNunitIntegrationTest\app.config"
71      #Write-Host "configSource is : $configSource"
72      #$configDestination = Join-Path $root
        "..\RelativityAgent1\AgentNunitIntegrationTest\bin\Debug\AgentNunitIntegrationTest.dll.config"
73      Write-Host "Test assembly : $testAssembly"
74      #Copy-Item $configSource $configDestination -Verbose:$VerbosePreference
```

After:



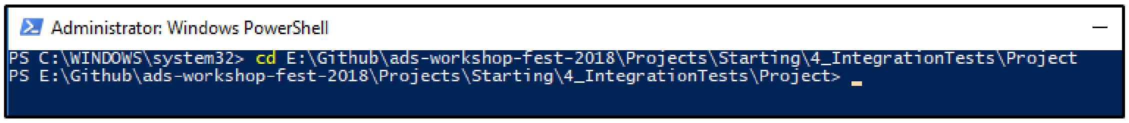
```
$testDir = Join-Path $root "..\Agents.Tests.Integration"
$configSource = "..\Project\Agents.Tests.Integration\App.config"
Write-Host "configSource is : $configSource"
$configDestination = Join-Path $root
"..\Project\Agents.Tests.Integration\bin\Debug\Agents.Tests.Integration.dll.config"
Write-Host "Test assembly : $testAssembly"
Copy-Item $configSource $configDestination -Verbose:$VerbosePreference
```

6. Open PowerShell and run the following command.

   Set-ExecutionPolicy Unrestricted

7. When prompted type **A** and press **Enter**.
8. Close the current PowerShell window and open a new PowerShell window.
9. Navigate to **“E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project”** folder by running the following command.

   cd E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project

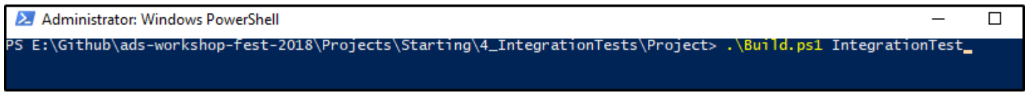


Administrator: Windows PowerShell

```
PS C:\WINDOWS\system32> cd E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project
PS E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project> _
```

10. Run the following command to run the Integration Tests.

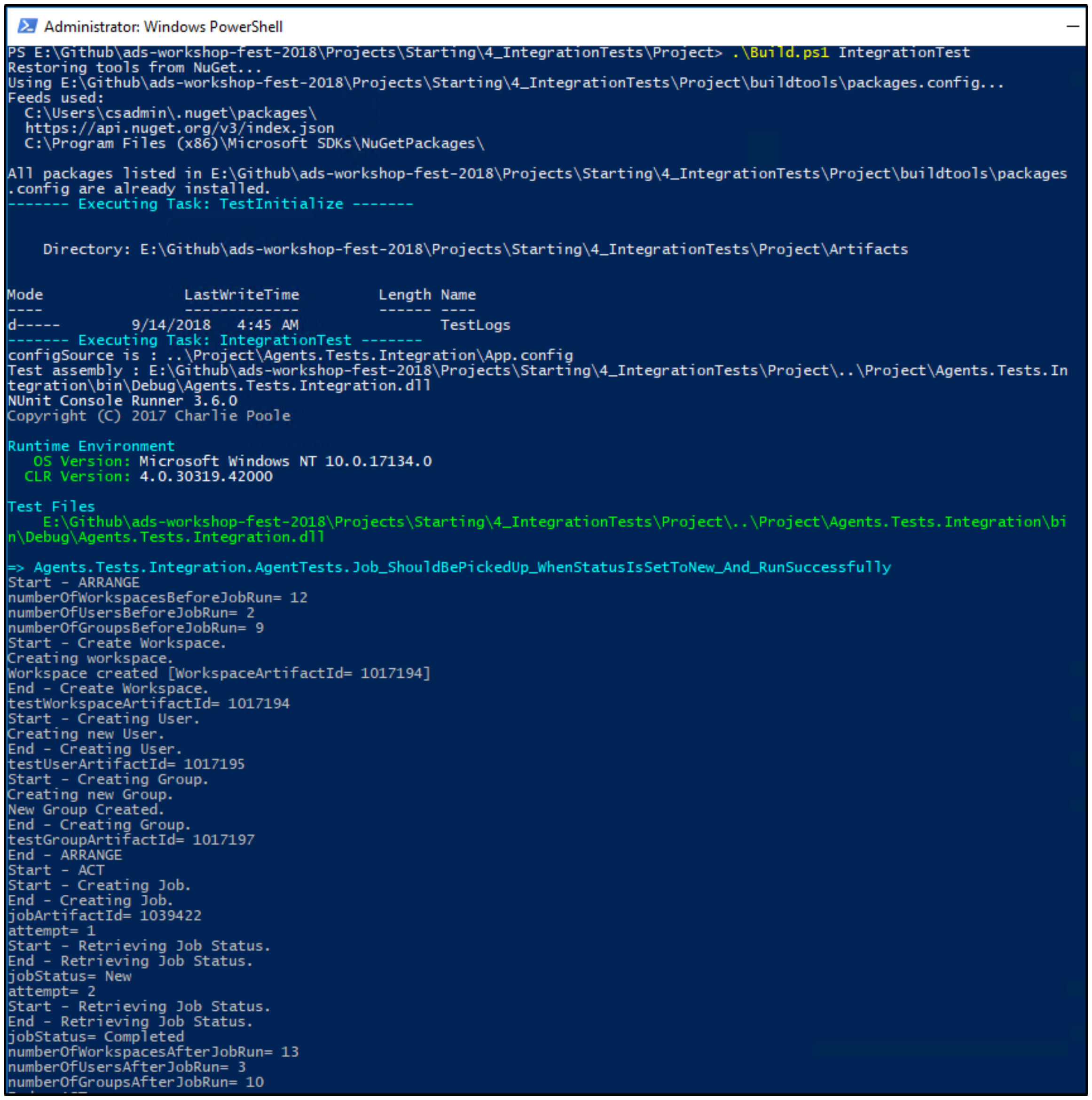
    .\Build.ps1 IntegrationTest



Administrator: Windows PowerShell

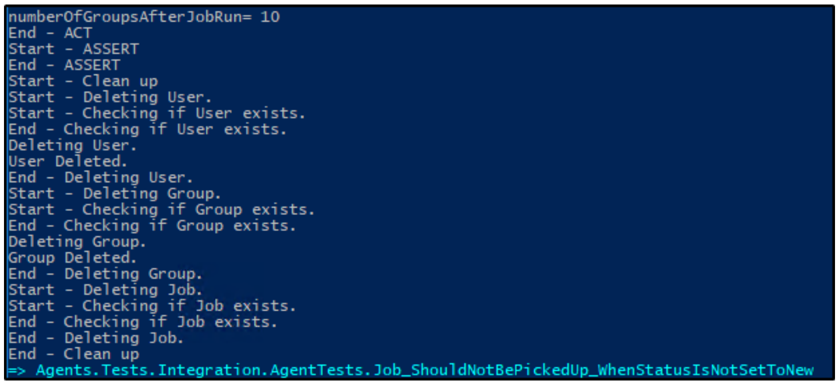
```
PS E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project> .\Build.ps1 IntegrationTest_
```

11. You will see the following output when the Integration tests are run successfully.

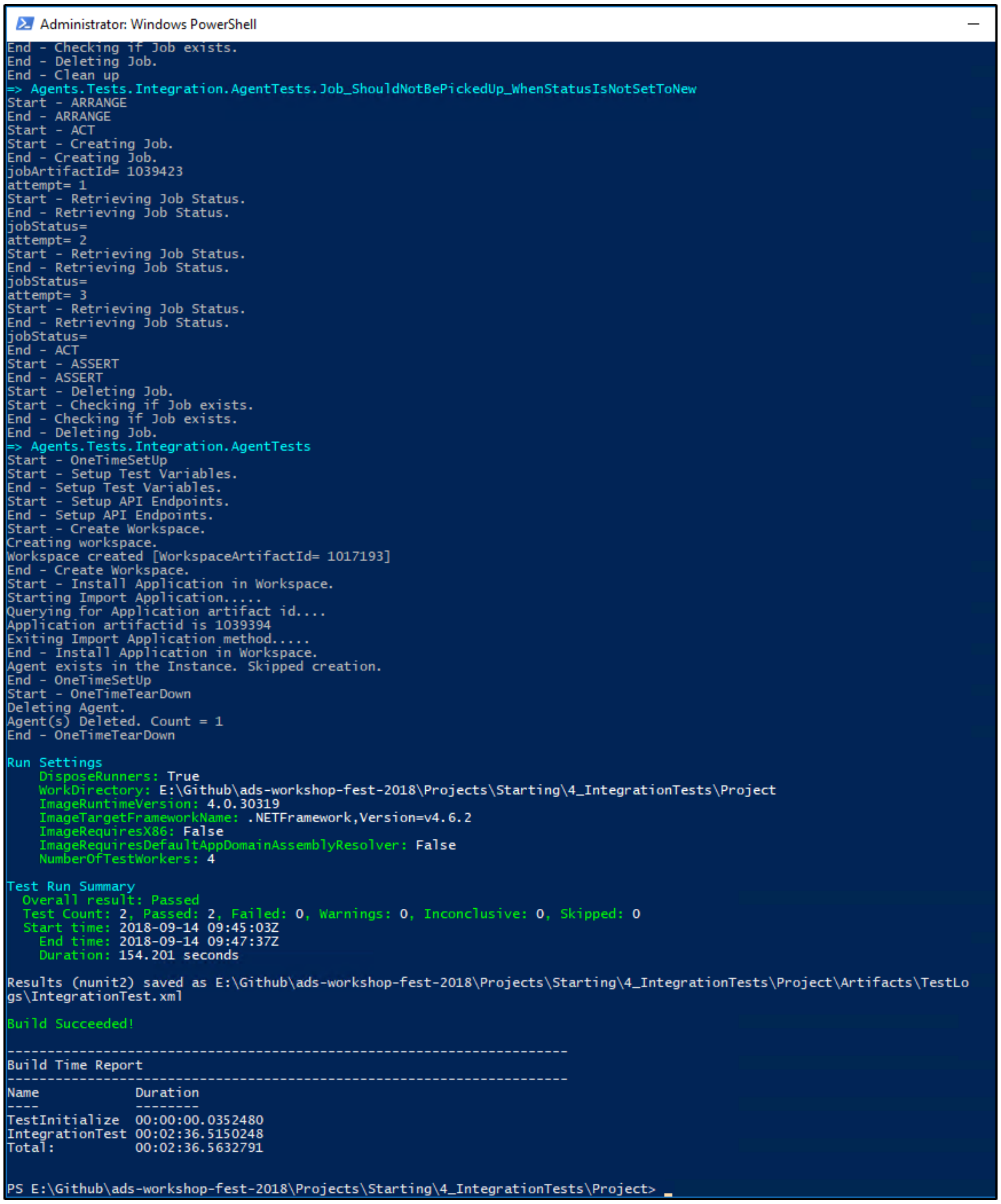


Administrator: Windows PowerShell

```
PS E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project> .\Build.ps1 IntegrationTest
Restoring tools from NuGet...
Using E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project\buildtools\packages.config...
Feeds used:
  C:\Users\csadmin\.nuget\packages\
  https://api.nuget.org/v3/index.json
  C:\Program Files (x86)\Microsoft SDKs\NuGetPackages\

All packages listed in E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project\buildtools\packages
.config are already installed.
------- Executing Task: TestInitialize -------


    Directory: E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project\Artifacts


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
d-----        9/14/2018   4:45 AM                TestLogs
------- Executing Task: IntegrationTest -------
configSource is : ..\Project\Agents.Tests.Integration\App.config
Test assembly : E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project\..\Project\Agents.Tests.In
tegration\bin\Debug\Agents.Tests.Integration.dll
NUnit Console Runner 3.6.0
Copyright (C) 2017 Charlie Poole

Runtime Environment
   OS Version: Microsoft Windows NT 10.0.17134.0
  CLR Version: 4.0.30319.42000

Test Files
    E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project\..\Project\Agents.Tests.Integration\bi
n\Debug\Agents.Tests.Integration.dll

=> Agents.Tests.Integration.AgentTests.Job_ShouldBePickedUp_WhenStatusIsSetToNew_And_RunSuccessfully
Start - ARRANGE
numberOfWorkspacesBeforeJobRun= 12
numberOfUsersBeforeJobRun= 2
numberOfGroupsBeforeJobRun= 9
Start - Create Workspace.
Creating workspace.
Workspace created [WorkspaceArtifactId= 1017194]
End - Create Workspace.
testWorkspaceArtifactId= 1017194
Start - Creating User.
Creating new User.
End - Creating User.
testUserArtifactId= 1017195
Start - Creating Group.
Creating new Group.
New Group Created.
End - Creating Group.
testGroupArtifactId= 1017197
End - ARRANGE
Start - ACT
Start - Creating Job.
End - Creating Job.
jobArtifactId= 1039422
attempt= 1
Start - Retrieving Job Status.
End - Retrieving Job Status.
jobStatus= New
attempt= 2
Start - Retrieving Job Status.
End - Retrieving Job Status.
jobStatus= Completed
numberOfWorkspacesAfterJobRun= 13
numberOfUsersAfterJobRun= 3
numberOfGroupsAfterJobRun= 10
```



```
numberOfGroupsAfterJobRun= 10
End - ACT
Start - ASSERT
End - ASSERT
Start - Clean up
Start - Deleting User.
Start - Checking if User exists.
End - Checking if User exists.
Deleting User.
User Deleted.
End - Deleting User.
Start - Deleting Group.
Start - Checking if Group exists.
End - Checking if Group exists.
Deleting Group.
Group Deleted.
End - Deleting Group.
Start - Deleting Job.
Start - Checking if Job exists.
End - Checking if Job exists.
End - Deleting Job.
End - Clean up
=> Agents.Tests.Integration.AgentTests.Job_ShouldNotBePickedUp_WhenStatusIsNotSetToNew
```



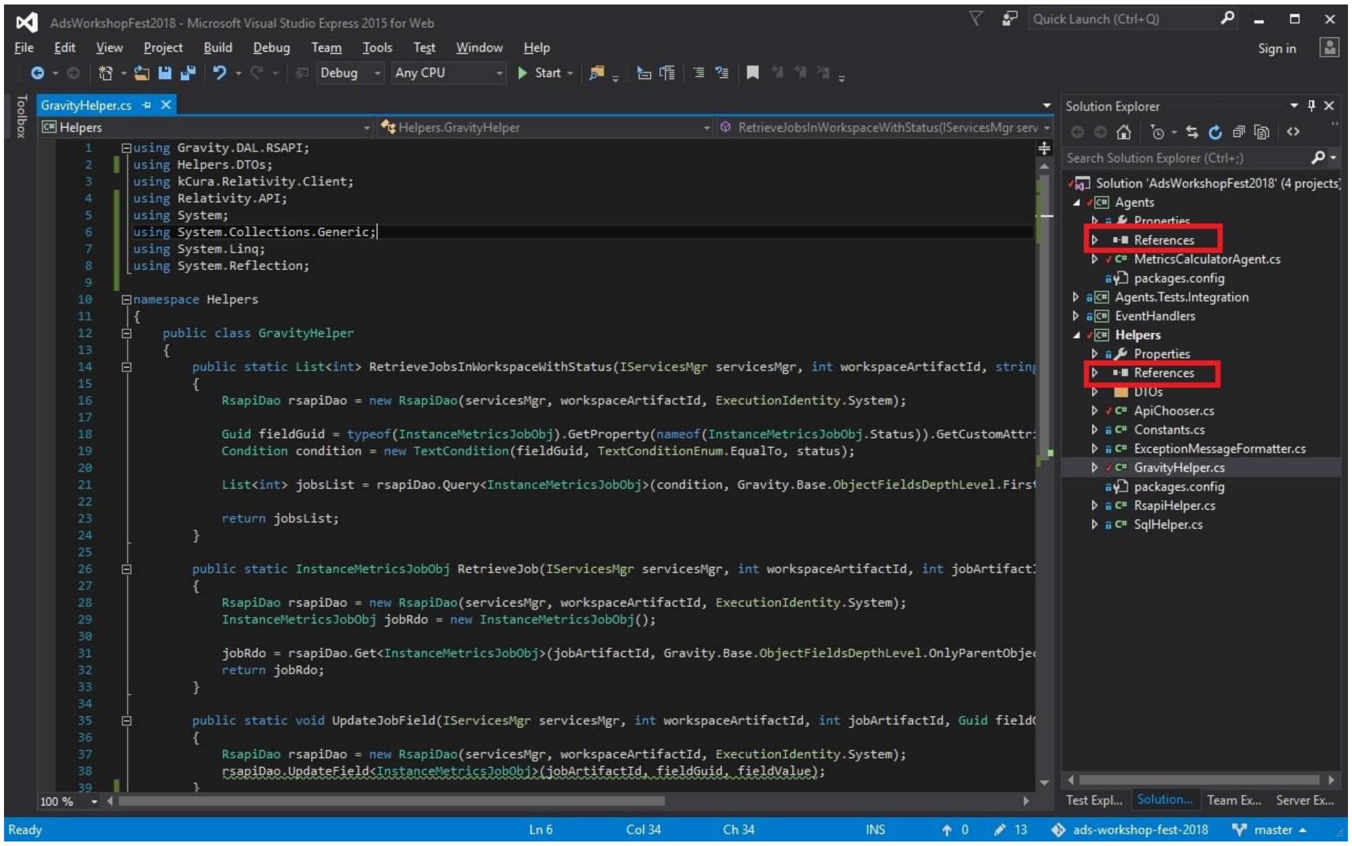
Administrator: Windows PowerShell

```
End - Checking if Job exists.
End - Deleting Job.
End - Clean up
=> Agents.Tests.Integration.AgentTests.Job_ShouldNotBePickedUp_WhenStatusIsNotSetToNew
Start - ARRANGE
End - ARRANGE
Start - ACT
Start - Creating Job.
End - Creating Job.
jobArtifactId= 1039423
attempt= 1
Start - Retrieving Job Status.
End - Retrieving Job Status.
jobStatus=
attempt= 2
Start - Retrieving Job Status.
End - Retrieving Job Status.
jobStatus=
attempt= 3
Start - Retrieving Job Status.
End - Retrieving Job Status.
jobStatus=
End - ACT
Start - ASSERT
End - ASSERT
Start - Deleting Job.
Start - Checking if Job exists.
End - Checking if Job exists.
End - Deleting Job.
=> Agents.Tests.Integration.AgentTests
Start - OneTimeSetUp
Start - Setup Test Variables.
End - Setup Test Variables.
Start - Setup API Endpoints.
End - Setup API Endpoints.
Start - Create Workspace.
Creating workspace.
Workspace created [WorkspaceArtifactId= 1017193]
End - Create Workspace.
Start - Install Application in Workspace.
Starting Import Application.....
Querying for Application artifact id....
Application artifactid is 1039394
Exiting Import Application method.....
End - Install Application in Workspace.
Agent exists in the Instance. Skipped creation.
End - OneTimeSetUp
Start - OneTimeTearDown
Deleting Agent.
Agent(s) Deleted. Count = 1
End - OneTimeTearDown

Run Settings
    DisposeRunners: True
    WorkDirectory: E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project
    ImageRuntimeVersion: 4.0.30319
    ImageTargetFrameworkName: .NETFramework,Version=v4.6.2
    ImageRequiresX86: False
    ImageRequiresDefaultAppDomainAssemblyResolver: False
    NumberOfTestWorkers: 4

Test Run Summary
  Overall result: Passed
  Test Count: 2, Passed: 2, Failed: 0, Warnings: 0, Inconclusive: 0, Skipped: 0
  Start time: 2018-09-14 09:45:03Z
    End time: 2018-09-14 09:47:37Z
    Duration: 154.201 seconds

Results (nunit2) saved as E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project\Artifacts\TestLo
gs\IntegrationTest.xml

Build Succeeded!

----------------------------------------------------------------------
Build Time Report
----------------------------------------------------------------------
Name            Duration
----            --------
TestInitialize  00:00:00.0352480
IntegrationTest 00:02:36.5150248
Total:          00:02:36.5632791

PS E:\Github\ads-workshop-fest-2018\Projects\Starting\4_IntegrationTests\Project> _
```

7 Gravity API

1 In this section, we will be using the Gravity open source API instead of RSAPI to read and update RDO values. You can read more about the Gravity API at this link: **https://github.com/relativitydev/Gravity**

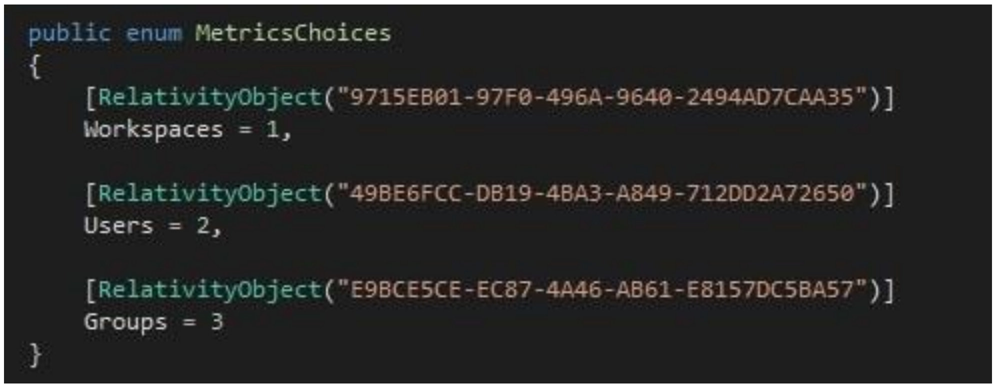
2 To use the **Gravity** API, first go to Visual Studio and add Gravity.dll as new reference for **Helpers** project and **Agents** project. You can find Gravity.dll in **ads-workshop-fest-2018\DLLs\Gravity**


AdsWorkshopFest2018 - Microsoft Visual Studio Express 2015 for Web
File Edit View Project Build Debug Team Tools Test Window Help
Debug
Any CPU
Start
GravityHelper.cs
Helpers
Helpers.GravityHelper
RetrieveJobsInWorkspaceWithStatus(IServicesMgr serv
Solution Explorer
Solution 'AdsWorkshopFest2018' (4 projects)
Agents
Properties
References
MetricsCalculatorAgent.cs
packages.config
Agents.Tests.Integration
EventHandlers
Helpers
Properties
References
DTOs
ApiChooser.cs
Constants.cs
ExceptionMessageFormatter.cs
GravityHelper.cs
packages.config
RsapiHelper.cs
SqlHelper.cs
Ready Ln 6 Col 34 Ch 34 INS
ads-workshop-fest-2018
master

```
using Gravity.DAL.RSAPI;
using Helpers.DTOs;
using kCura.Relativity.Client;
using Relativity.API;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Reflection;

namespace Helpers
{
    public class GravityHelper
    {
        public static List<int> RetrieveJobsInWorkspaceWithStatus(IServicesMgr servicesMgr, int workspaceArtifactId, string
        {
            RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId, ExecutionIdentity.System);

            Guid fieldGuid = typeof(InstanceMetricsJobObj).GetProperty(nameof(InstanceMetricsJobObj.Status)).GetCustomAttr
            Condition condition = new TextCondition(fieldGuid, TextConditionEnum.EqualTo, status);

            List<int> jobsList = rsapiDao.Query<InstanceMetricsJobObj>(condition, Gravity.Base.ObjectFieldsDepthLevel.First

            return jobsList;
        }

        public static InstanceMetricsJobObj RetrieveJob(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifact
        {
            RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId, ExecutionIdentity.System);
            InstanceMetricsJobObj jobRdo = new InstanceMetricsJobObj();

            jobRdo = rsapiDao.Get<InstanceMetricsJobObj>(jobArtifactId, Gravity.Base.ObjectFieldsDepthLevel.OnlyParentObje
            return jobRdo;
        }

        public static void UpdateJobField(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId, Guid field
        {
            RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId, ExecutionIdentity.System);
            rsapiDao.UpdateField<InstanceMetricsJobObj>(jobArtifactId, fieldGuid, fieldValue);
        }
```

3 To use the **Gravity** API, first go to Visual Studio and create enum MetricsChoices under folder DTOs in Helpers project. This enum represents the Choices for Metrics field.

*Note: To create the enum right-click on the folder **DTOs** and select **Add -> New Item**. On the new window select **Visual C# Items -> Class**. Name the file MetricsChoices.cs.*



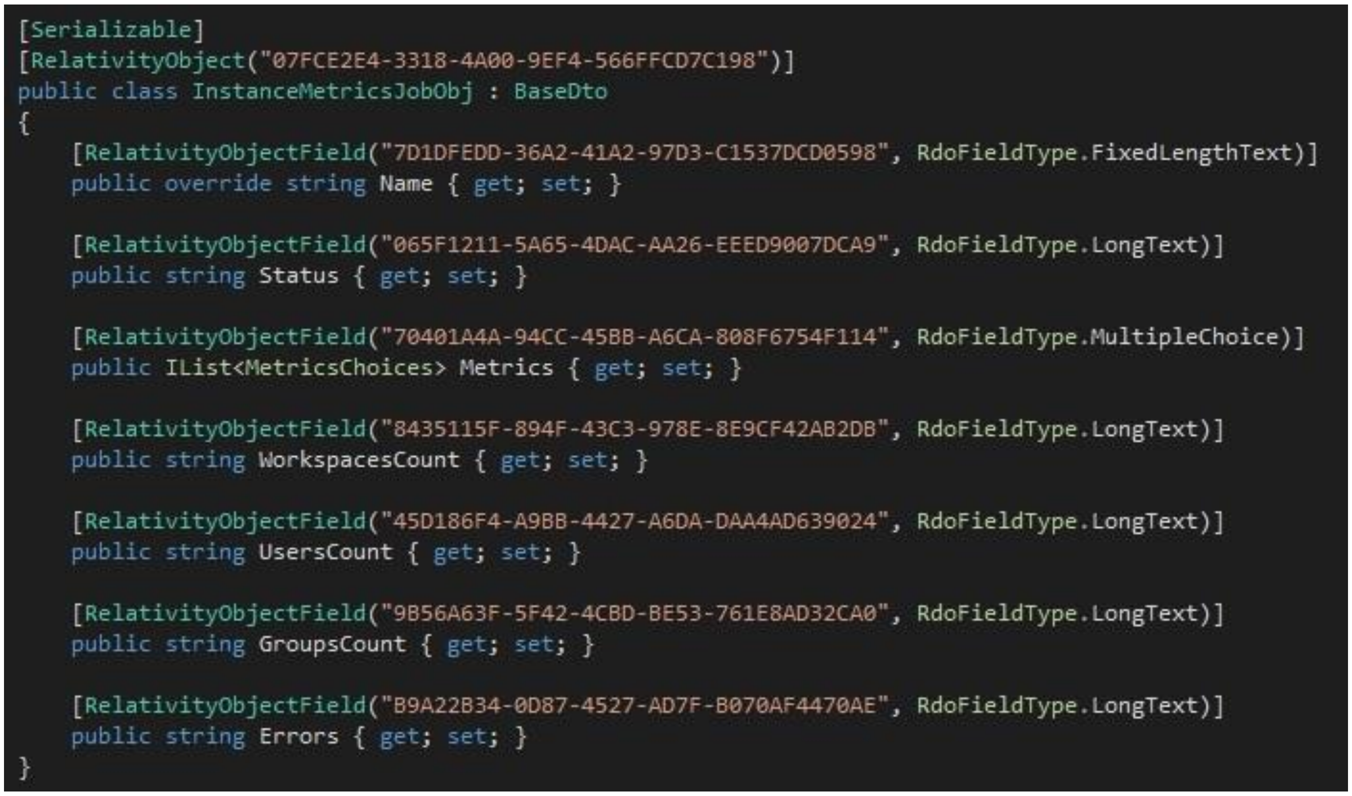
```
public enum MetricsChoices
{
    [RelativityObject("9715EB01-97F0-496A-9640-2494AD7CAA35")]
    Workspaces = 1,

    [RelativityObject("49BE6FCC-DB19-4BA3-A849-712DD2A72650")]
    Users = 2,

    [RelativityObject("E9BCE5CE-EC87-4A46-AB61-E8157DC5BA57")]
    Groups = 3
}
```

Relativity

Code to add:

```
public enum MetricsChoices
{
        [RelativityObject("9715EB01-97F0-496A-9640-2494AD7CAA35")]
        Workspaces = 1,

        [RelativityObject("49BE6FCC-DB19-4BA3-A849-712DD2A72650")]
        Users = 2,

        [RelativityObject("E9BCE5CE-EC87-4A46-AB61-E8157DC5BA57")]
        Groups = 3
}
```

4 Then create class for the RDO with its fields as properties under folder DTOs in Helpers project.
*Note: To create the class right-click on the folder **DTOs** and select **Add -> New Item**. On the new window select **Visual C# Items -> Class**. Name the file InstanceMetricsJobObj.cs.*

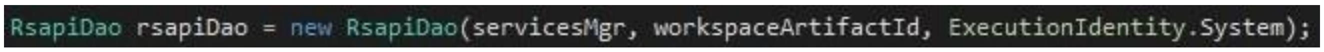


```
[Serializable]
[RelativityObject("07FCE2E4-3318-4A00-9EF4-566FFCD7C198")]
public class InstanceMetricsJobObj : BaseDto
{
    [RelativityObjectField("7D1DFEDD-36A2-41A2-97D3-C1537DCD0598", RdoFieldType.FixedLengthText)]
    public override string Name { get; set; }

    [RelativityObjectField("065F1211-5A65-4DAC-AA26-EEED9007DCA9", RdoFieldType.LongText)]
    public string Status { get; set; }

    [RelativityObjectField("70401A4A-94CC-45BB-A6CA-808F6754F114", RdoFieldType.MultipleChoice)]
    public IList<MetricsChoices> Metrics { get; set; }

    [RelativityObjectField("8435115F-894F-43C3-978E-8E9CF42AB2DB", RdoFieldType.LongText)]
    public string WorkspacesCount { get; set; }

    [RelativityObjectField("45D186F4-A9BB-4427-A6DA-DAA4AD639024", RdoFieldType.LongText)]
    public string UsersCount { get; set; }

    [RelativityObjectField("9B56A63F-5F42-4CBD-BE53-761E8AD32CA0", RdoFieldType.LongText)]
    public string GroupsCount { get; set; }

    [RelativityObjectField("B9A22B34-0D87-4527-AD7F-B070AF4470AE", RdoFieldType.LongText)]
    public string Errors { get; set; }
}
```

Code to add:

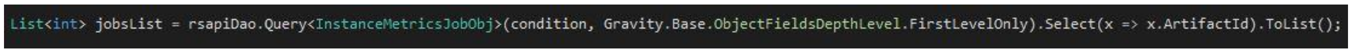
```
using Gravity.Base;
using System;
using System.Collections.Generic;

namespace Helpers.DTOs
{
        [Serializable]
        [RelativityObject("07FCE2E4-3318-4A00-9EF4-566FFCD7C198")]
        public class InstanceMetricsJobObj : BaseDto
        {
                [RelativityObjectField("7D1DFEDD-36A2-41A2-97D3-C1537DCD0598",
    RdoFieldType.FixedLengthText)]
                public override string Name { get; set; }

                [RelativityObjectField("065F1211-5A65-4DAC-AA26-EEED9007DCA9",
    RdoFieldType.LongText)]
                public string Status { get; set; }

                [RelativityObjectField("70401A4A-94CC-45BB-A6CA-808F6754F114",
    RdoFieldType.MultipleChoice)]
                public IList<MetricsChoices> Metrics { get; set; }

                [RelativityObjectField("8435115F-894F-43C3-978E-8E9CF42AB2DB",
    RdoFieldType.LongText)]
                public string WorkspacesCount { get; set; }

                [RelativityObjectField("45D186F4-A9BB-4427-A6DA-DAA4AD639024",
    RdoFieldType.LongText)]
                public string UsersCount { get; set; }

                [RelativityObjectField("9B56A63F-5F42-4CBD-BE53-761E8AD32CA0",
    RdoFieldType.LongText)]
                public string GroupsCount { get; set; }

                [RelativityObjectField("B9A22B34-0D87-4527-AD7F-B070AF4470AE",
    RdoFieldType.LongText)]
                public string Errors { get; set; }
        }
    }
```

5 To use Gravity, you have to create RsapiDao. You will see this for the methods below.



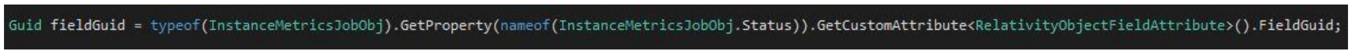
```
RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId, ExecutionIdentity.System);
```

6 When you have created the RsapiDao, you can use it for various CRUDQ operations.



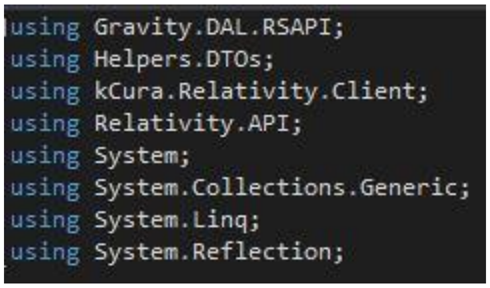
```
List<int> jobsList = rsapiDao.Query<InstanceMetricsJobObj>(condition, Gravity.Base.ObjectFieldsDepthLevel.FirstLevelOnly).Select(x => x.ArtifactId).ToList();
```

7 If you want to retrieve the GUID for a field, you can do that by using Reflection



```
Guid fieldGuid = typeof(InstanceMetricsJobObj).GetProperty(nameof(InstanceMetricsJobObj.Status)).GetCustomAttribute<RelativityObjectFieldAttribute>().FieldGuid;
```

8 Open the **GravityHelper.cs** file in **Helpers** project and add the following **using** statements

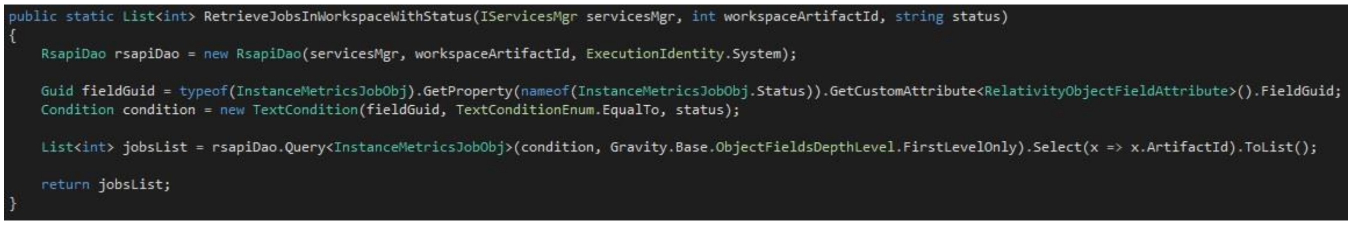


```
using Gravity.DAL.RSAPI;
using Helpers.DTOs;
using kCura.Relativity.Client;
using Relativity.API;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Reflection;
```

Code to add:

using Gravity.DAL.RSAPI;
using Helpers.DTOs;
using kCura.Relativity.Client;
using Relativity.API;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Reflection;

9 Next declare method to retrieve all jobs in a workspace for specific status



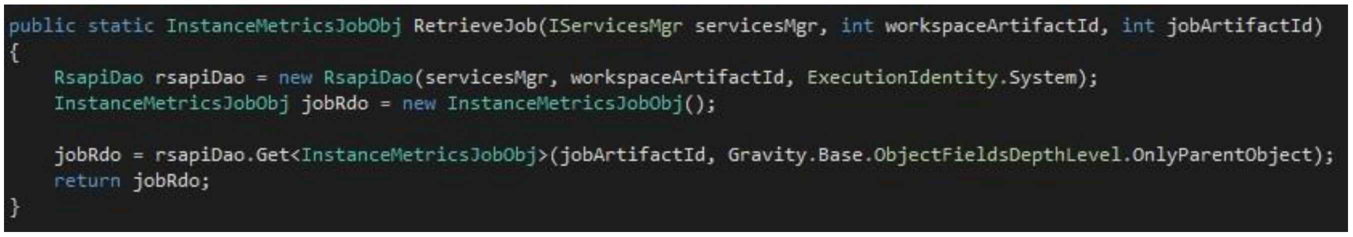
```
public static List<int> RetrieveJobsInWorkspaceWithStatus(IServicesMgr servicesMgr, int workspaceArtifactId, string status)
{
    RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId, ExecutionIdentity.System);

    Guid fieldGuid = typeof(InstanceMetricsJobObj).GetProperty(nameof(InstanceMetricsJobObj.Status)).GetCustomAttribute<RelativityObjectFieldAttribute>().FieldGuid;
    Condition condition = new TextCondition(fieldGuid, TextConditionEnum.EqualTo, status);

    List<int> jobsList = rsapiDao.Query<InstanceMetricsJobObj>(condition, Gravity.Base.ObjectFieldsDepthLevel.FirstLevelOnly).Select(x => x.ArtifactId).ToList();

    return jobsList;
}
```

Code to add:

```
public static List<int> RetrieveJobsInWorkspaceWithStatus(IServicesMgr servicesMgr, int
workspaceArtifactId, string status)
        {
                RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId,
ExecutionIdentity.System);

                Guid fieldGuid =
typeof(InstanceMetricsJobObj).GetProperty(nameof(InstanceMetricsJobObj.Status)).GetCustomAttri
bute<RelativityObjectFieldAttribute>().FieldGuid;
                Condition condition = new TextCondition(fieldGuid, TextConditionEnum.EqualTo,
status);

                List<int> jobsList = rsapiDao.Query<InstanceMetricsJobObj>(condition,
Gravity.Base.ObjectFieldsDepthLevel.FirstLevelOnly).Select(x => x.ArtifactId).ToList();
```

```
                return jobsList;
        }
```

10 In the same **GravitiyHelper.cs**, declare method to retrieve one specific job

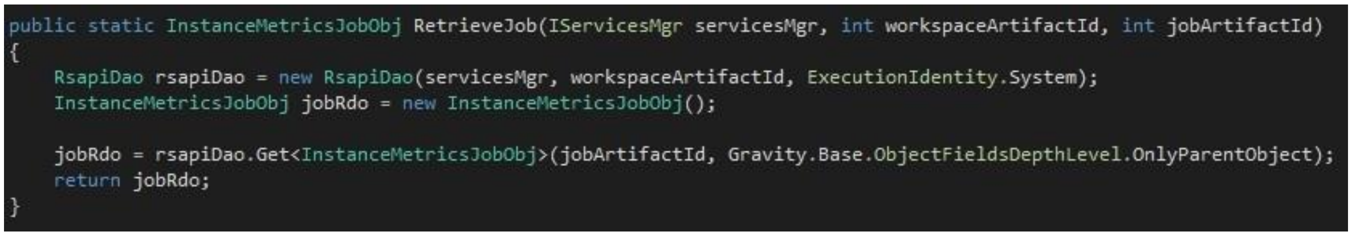


```
public static InstanceMetricsJobObj RetrieveJob(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId)
{
    RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId, ExecutionIdentity.System);
    InstanceMetricsJobObj jobRdo = new InstanceMetricsJobObj();

    jobRdo = rsapiDao.Get<InstanceMetricsJobObj>(jobArtifactId, Gravity.Base.ObjectFieldsDepthLevel.OnlyParentObject);
    return jobRdo;
}
```

<u>**Code to add:**</u>

```
public static InstanceMetricsJobObj RetrieveJob(IServicesMgr servicesMgr, int workspaceArtifactId,
int jobArtifactId)
        {
                RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId,
ExecutionIdentity.System);
                InstanceMetricsJobObj jobRdo = new InstanceMetricsJobObj();

                jobRdo = rsapiDao.Get<InstanceMetricsJobObj>(jobArtifactId,
Gravity.Base.ObjectFieldsDepthLevel.OnlyParentObject);
                return jobRdo;
        }
```

11 In the same **GravityHelper.cs**, declare method to update specific field

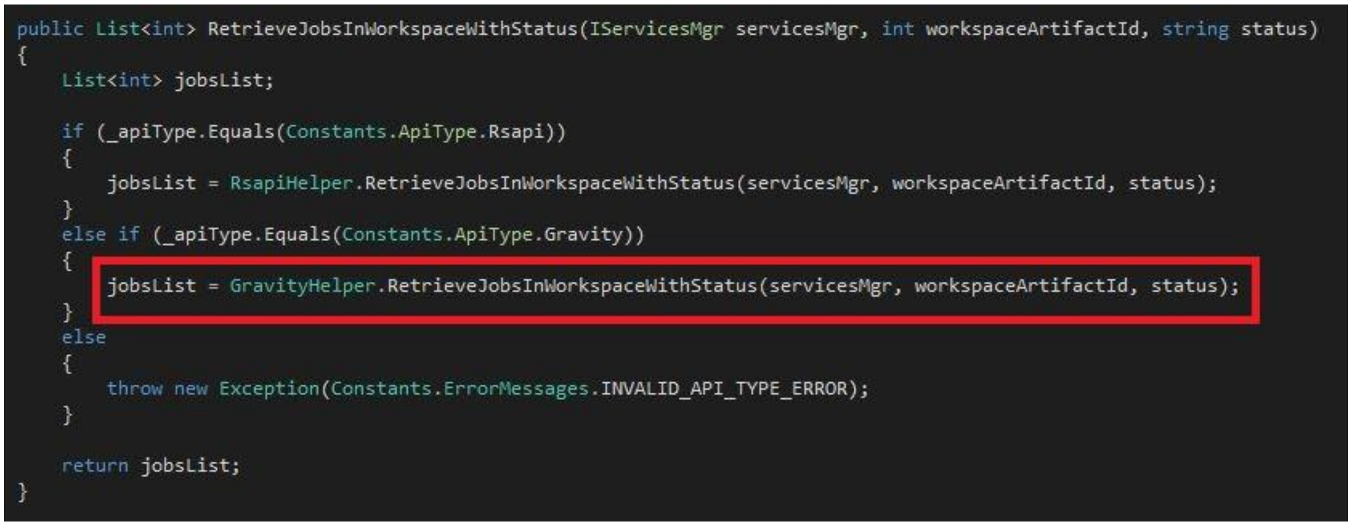


```
public static InstanceMetricsJobObj RetrieveJob(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId)
{
    RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId, ExecutionIdentity.System);
    InstanceMetricsJobObj jobRdo = new InstanceMetricsJobObj();

    jobRdo = rsapiDao.Get<InstanceMetricsJobObj>(jobArtifactId, Gravity.Base.ObjectFieldsDepthLevel.OnlyParentObject);
    return jobRdo;
}
```

<u>**Code to add:**</u>

```
public static void UpdateJobField(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId,
Guid fieldGuid, object fieldValue)
        {
                RsapiDao rsapiDao = new RsapiDao(servicesMgr, workspaceArtifactId,
ExecutionIdentity.System);
                rsapiDao.UpdateField<InstanceMetricsJobObj>(jobArtifactId, fieldGuid, fieldValue);
        }
```

12 Open the **ApiChooser.cs** file in **Helpers** project and invoke the method **GravityHelper.RetrieveJobsInWorkspaceWithStatus** (declared in step 4) in method public **List<int> RetrieveJobsInWorkspaceWithStatus(IServicesMgr servicesMgr, int workspaceArtifactId, string status)**



```
public List<int> RetrieveJobsInWorkspaceWithStatus(IServicesMgr servicesMgr, int workspaceArtifactId, string status)
{
    List<int> jobsList;

    if (_apiType.Equals(Constants.ApiType.Rsapi))
    {
        jobsList = RsapiHelper.RetrieveJobsInWorkspaceWithStatus(servicesMgr, workspaceArtifactId, status);
    }
    else if (_apiType.Equals(Constants.ApiType.Gravity))
    {
        jobsList = GravityHelper.RetrieveJobsInWorkspaceWithStatus(servicesMgr, workspaceArtifactId, status);
    }
    else
    {
        throw new Exception(Constants.ErrorMessages.INVALID_API_TYPE_ERROR);
    }

    return jobsList;
}
```

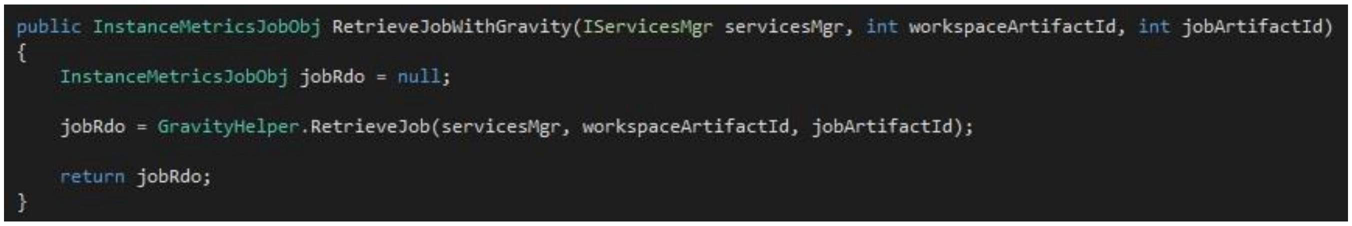
Code to comment:

jobsList = new List<int>();

Code to add:

jobsList = GravityHelper.RetrieveJobsInWorkspaceWithStatus(servicesMgr, workspaceArtifactId, status);

13 Next, create a method to invoke **GravityHelper.RetrieveJob**



```
public InstanceMetricsJobObj RetrieveJobWithGravity(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId)
{
    InstanceMetricsJobObj jobRdo = null;

    jobRdo = GravityHelper.RetrieveJob(servicesMgr, workspaceArtifactId, jobArtifactId);

    return jobRdo;
}
```

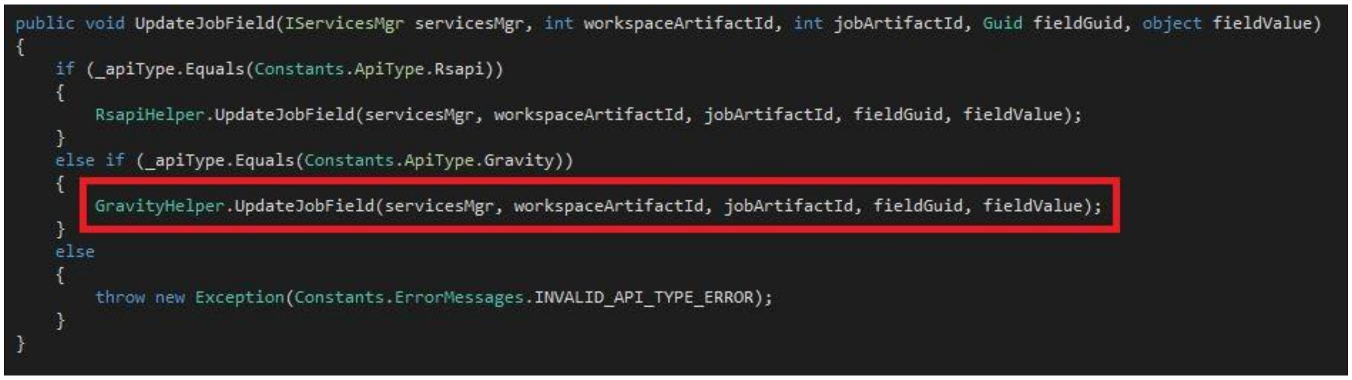
Code to add:

public InstanceMetricsJobObj RetrieveJobWithGravity(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId)
{
    InstanceMetricsJobObj jobRdo = null;

jobRdo = GravityHelper.RetrieveJob(servicesMgr, workspaceArtifactId, jobArtifactId);

return jobRdo;
}

14 Now invoke **GravityHelper.UpdateJobField** in method **public void UpdateJobField(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId, Guid fieldGuid, object fieldValue)**

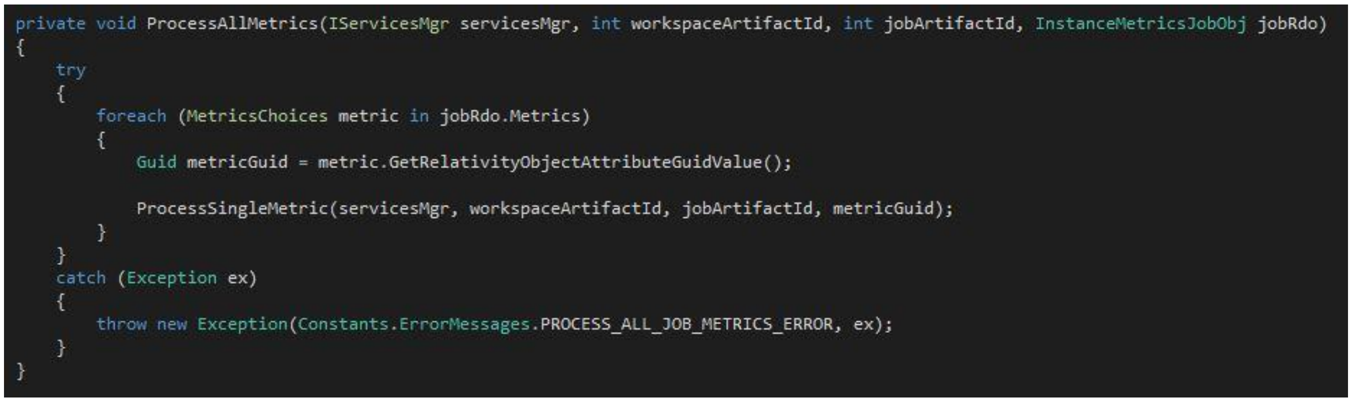


```
public void UpdateJobField(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId, Guid fieldGuid, object fieldValue)
{
    if (_apiType.Equals(Constants.ApiType.Rsapi))
    {
        RsapiHelper.UpdateJobField(servicesMgr, workspaceArtifactId, jobArtifactId, fieldGuid, fieldValue);
    }
    else if (_apiType.Equals(Constants.ApiType.Gravity))
    {
        GravityHelper.UpdateJobField(servicesMgr, workspaceArtifactId, jobArtifactId, fieldGuid, fieldValue);
    }
    else
    {
        throw new Exception(Constants.ErrorMessages.INVALID_API_TYPE_ERROR);
    }
}
```

Code to add:

GravityHelper.UpdateJobField(servicesMgr, workspaceArtifactId, jobArtifactId, fieldGuid, fieldValue);

15 Now you are ready to use the ApiChooser methods in the Agents project. Go to Agents project and open **MetricsCalculatorAgent.cs**. Create overload method **ProcessAllMetrics(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId, InstanceMetricsJobObj jobRdo)**.

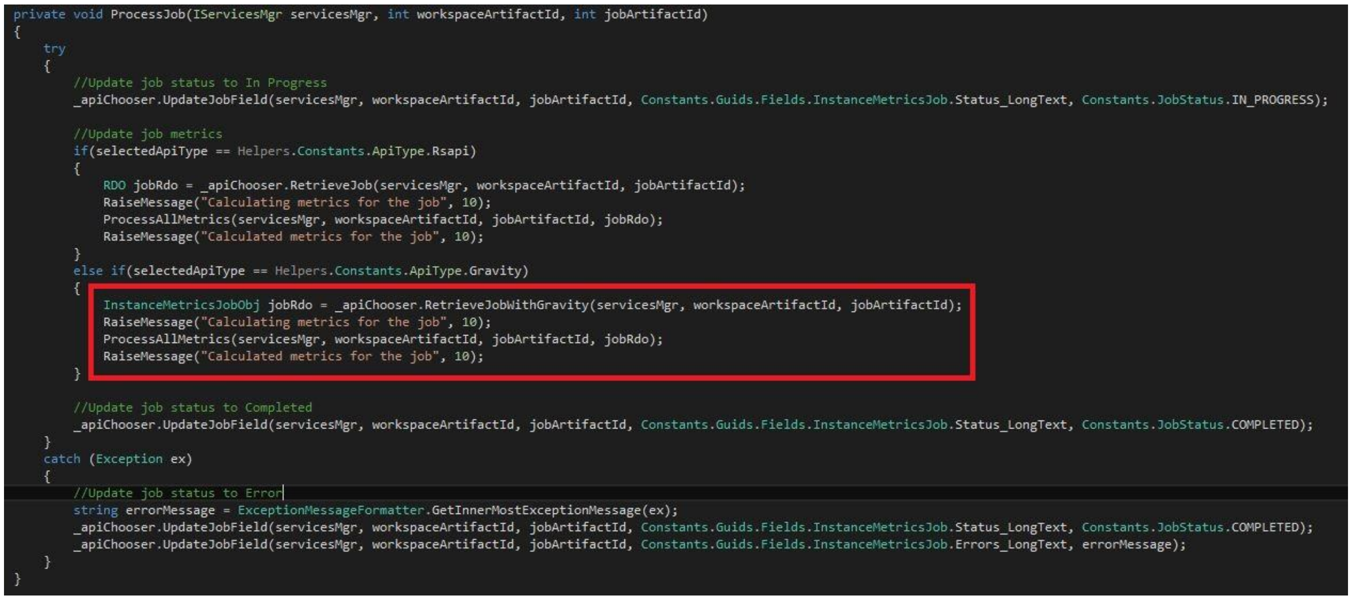


```
private void ProcessAllMetrics(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId, InstanceMetricsJobObj jobRdo)
{
    try
    {
        foreach (MetricsChoices metric in jobRdo.Metrics)
        {
            Guid metricGuid = metric.GetRelativityObjectAttributeGuidValue();

            ProcessSingleMetric(servicesMgr, workspaceArtifactId, jobArtifactId, metricGuid);
        }
    }
    catch (Exception ex)
    {
        throw new Exception(Constants.ErrorMessages.PROCESS_ALL_JOB_METRICS_ERROR, ex);
    }
}
```

Code to add:

```
private void ProcessAllMetrics(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId,
InstanceMetricsJobObj jobRdo)
        {
                try
                {
                        foreach (MetricsChoices metric in jobRdo.Metrics)
                        {
                                Guid metricGuid = metric.GetRelativityObjectAttributeGuidValue();

                                ProcessSingleMetric(servicesMgr, workspaceArtifactId, jobArtifactId,
metricGuid);
                        }
                }
                catch (Exception ex)
                {
                        throw new
Exception(Constants.ErrorMessages.PROCESS_ALL_JOB_METRICS_ERROR, ex);
                }
        }
```

16 Use RetrieveJobWithGravity to retrieve and process the job in **private void ProcessJob(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId)**



```
private void ProcessJob(IServicesMgr servicesMgr, int workspaceArtifactId, int jobArtifactId)
{
    try
    {
        //Update job status to In Progress
        _apiChooser.UpdateJobField(servicesMgr, workspaceArtifactId, jobArtifactId, Constants.Guids.Fields.InstanceMetricsJob.Status_LongText, Constants.JobStatus.IN_PROGRESS);

        //Update job metrics
        if(selectedApiType == Helpers.Constants.ApiType.Rsapi)
        {
            RDO jobRdo = _apiChooser.RetrieveJob(servicesMgr, workspaceArtifactId, jobArtifactId);
            RaiseMessage("Calculating metrics for the job", 10);
            ProcessAllMetrics(servicesMgr, workspaceArtifactId, jobArtifactId, jobRdo);
            RaiseMessage("Calculated metrics for the job", 10);
        }
        else if(selectedApiType == Helpers.Constants.ApiType.Gravity)
        {
            InstanceMetricsJobObj jobRdo = _apiChooser.RetrieveJobWithGravity(servicesMgr, workspaceArtifactId, jobArtifactId);
            RaiseMessage("Calculating metrics for the job", 10);
            ProcessAllMetrics(servicesMgr, workspaceArtifactId, jobArtifactId, jobRdo);
            RaiseMessage("Calculated metrics for the job", 10);
        }

        //Update job status to Completed
        _apiChooser.UpdateJobField(servicesMgr, workspaceArtifactId, jobArtifactId, Constants.Guids.Fields.InstanceMetricsJob.Status_LongText, Constants.JobStatus.COMPLETED);
    }
    catch (Exception ex)
    {
        //Update job status to Error
        string errorMessage = ExceptionMessageFormatter.GetInnerMostExceptionMessage(ex);
        _apiChooser.UpdateJobField(servicesMgr, workspaceArtifactId, jobArtifactId, Constants.Guids.Fields.InstanceMetricsJob.Status_LongText, Constants.JobStatus.COMPLETED);
        _apiChooser.UpdateJobField(servicesMgr, workspaceArtifactId, jobArtifactId, Constants.Guids.Fields.InstanceMetricsJob.Errors_LongText, errorMessage);
    }
}
```

Code to add:

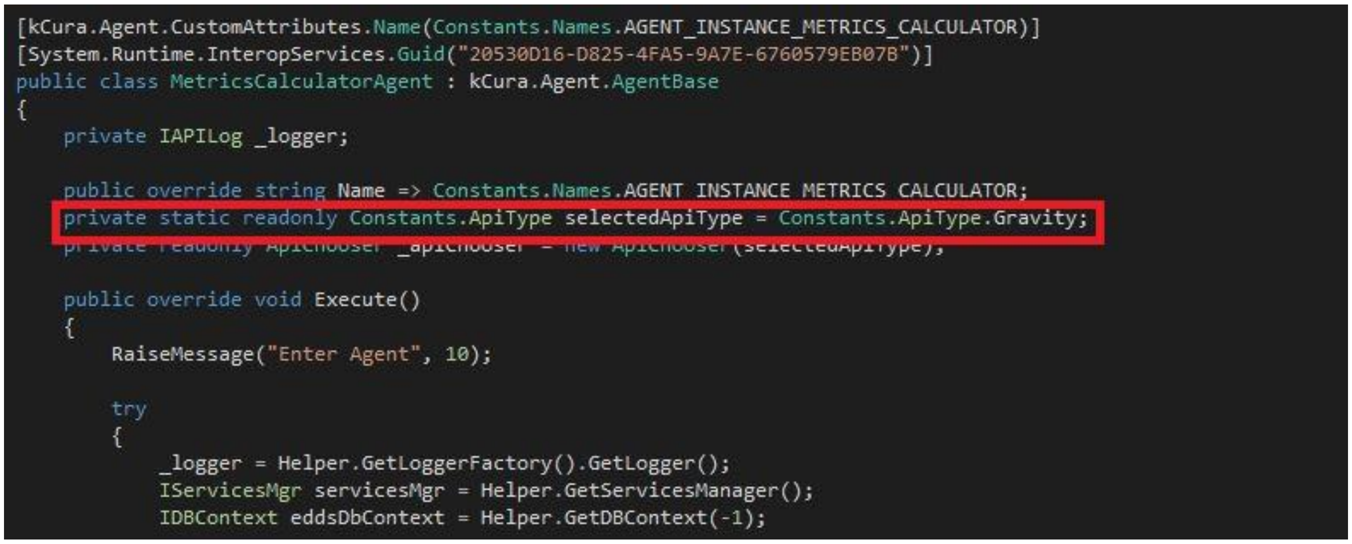
InstanceMetricsJobObj jobRdo = _apiChooser.RetrieveJobWithGravity(servicesMgr, workspaceArtifactId, jobArtifactId);

RaiseMessage("Calculating metrics for the job", 10);

ProcessAllMetrics(servicesMgr, workspaceArtifactId, jobArtifactId, jobRdo);

RaiseMessage("Calculated metrics for the job", 10);

17 To switch from RSAPI to Gravity change the field **private static readonly Constants.ApiType selectedApiType** in **MetricsCalculatorAgent.cs**



```
[kCura.Agent.CustomAttributes.Name(Constants.Names.AGENT_INSTANCE_METRICS_CALCULATOR)]
[System.Runtime.InteropServices.Guid("20530D16-D825-4FA5-9A7E-6760579EB07B")]
public class MetricsCalculatorAgent : kCura.Agent.AgentBase
{
    private IAPILog _logger;

    public override string Name => Constants.Names.AGENT_INSTANCE_METRICS_CALCULATOR;
    private static readonly Constants.ApiType selectedApiType = Constants.ApiType.Gravity;
    private readonly ApiChooser _apiChooser = new ApiChooser(selectedApiType);

    public override void Execute()
    {
        RaiseMessage("Enter Agent", 10);

        try
        {
            _logger = Helper.GetLoggerFactory().GetLogger();
            IServicesMgr servicesMgr = Helper.GetServicesManager();
            IDBContext eddsDbContext = Helper.GetDBContext(-1);
```