

# HOW TO DEVELOP UI APPS



COUNTERSOFT

# How to Develop UI Apps

## Contents

|                               |   |
|-------------------------------|---|
| 1. Introduction .....         | 2 |
| 2. Create Project .....       | 3 |
| 3. Structure .....            | 3 |
| 4. App.Manifest.....          | 4 |
| 5. Attributes .....           | 5 |
| 6. Project Setup.....         | 6 |
| 7. Methods and Services ..... | 6 |
| 8. Permissions .....          | 7 |
| 9. Deployment.....            | 7 |
| 10. References Examples ..... | 8 |

## 1. Introduction

Gemini has lots of features built in, which you can explore in our [docs](#). However there are times when you need something to work very specifically to your requirements in which case you can create your own apps using the Gemini App Framework.

This guide will show you how to create and deploy a basic Gemini UI widget app. We will be using our open-sourced [Checklist](#) app as reference. The main widget screen is a tab on the items page similar to the screenshot below.

|                        |              |               |             |          |             |           |
|------------------------|--------------|---------------|-------------|----------|-------------|-----------|
| Additional Information | Comments (0) | Followers (0) | History (1) | Time (0) | Email Reply | Checklist |
|------------------------|--------------|---------------|-------------|----------|-------------|-----------|

- Code Committed
- Documentation Updated
- Release Notes Compiled
- PDF Generated
- Paperwork Submitted

You can have an optional configuration screen in Customize -> Apps -> AppName. You can set permissions for specific projects and then use this permission set to validate user access in your app.

» [Set up permissions for UIApp.](#)

|            |                      |
|------------|----------------------|
| Projects   | <input type="text"/> |
| Can View   | <input type="text"/> |
| Can Create | <input type="text"/> |
| Can Edit   | <input type="text"/> |

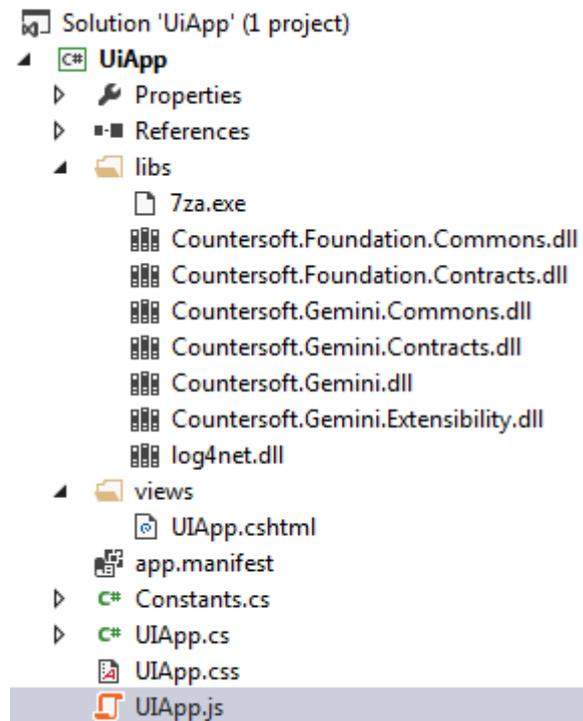
[Save](#)

## 2. Create Project

Open Visual Studio and download our project template *Gemini UI Widget App Project Template* from Tools -> Extensions and Updates -> Online. Close and open Visual Studio again in order to select the new template from File -> New -> Project -> Visual C#. Enter a *Name* and *Solution name* and make sure you rename the files UIApp.cshtml, UIApp.css and UIApp.js to the Solution name.

## 3. Structure

The new template will create a project with the following structure.



### Libs

Contains all relevant DLL's the app uses.

### Views

Contains Razor views – the user interface for your app if applicable.

### Manifest

The [app.manifest](#) file describes your app.

### CSS and Javascript

UIApp.css and UIApp.js are optional files. If you want to use them, then make sure they are referenced in the UIApp.cshtml file.

### Constants

Constants.cs contains the AppGuid and AppControlGuid.

## 4. App.Manifest

The manifest file describes your app and is used during deployment.

```

1 <app>
2   <guid>774C1984-CE45-4601-A8CC-87757E879129</guid>
3   <name>UIApp</name>
4   <description>UIApp Description</description>
5   <version-major>1</version-major>
6   <version-minor>0</version-minor>
7   <version-patch>0</version-patch>
8   <publisher>Someone</publisher>
9   <released>2014-01-23T11:31:00Z</released>
10  <debug>true</debug>
11 </app>

```

| Option        | Description  |
|---------------|--|
| GUID          | Globally unique identifier (GUID) for your app   |
| Name          | Application title displayed in screen setup for Administrators to identify your app  |
| Description   | Brief description of the application   |
| Version Major | <a href="#">Semantic versioning</a> for your app   |
| Version Minor |  |
| Version Patch |  |
| Publisher     | Name of the organization/individual who owns the app   |
| Released      | Date the app was released  |
| Debug         | Can be <i>true</i> or <i>false</i> . When <i>true</i> app is not cached and can be changed on disk (useful during development) |

## 5. Attributes

Set attributes to describe your app and configuration screen.

```

17 namespace UiApp
18 {
19     [AppType(AppTypeEnum.Widget),
20     AppGuid(Constants.AppGuid),
21     AppControlGuid(Constants.AppControlId),
22     AppAuthor("Countersoft"),
23     AppKey("UiApp"),
24     AppName("UiApp"),
25     AppDescription("UI sample description"),
26     AppRequiresConfigScreen(true),
27     AppRequiresCreatePermission(true),
28     AppRequiresEditPermission(true)]

```

### Attributes

| Option   | Description   |
|--|---|
| AppType  | The type of app i.e. Widget, Event, Timer etc.  |
| AppGuid  | Globally unique identifier (GUID) for your app  |
| AppControlGuid   | Global unique identifier for the control. One app can have multiple controllers   |
| AppAuthor  | Your name   |
| AppKey   | URL identifier for the app<br><a href="http://gemini/workspace/2/app/UIApp/view">http://gemini/workspace/2/app/UIApp/view</a>   |
| AppUrl   | URL identifier for the method<br><a href="http://gemini/workspace/2/app/UIApp/view">http://gemini/workspace/2/app/UIApp/view</a>  |
| AppName  | Application title displayed in Screen setup for Administrators to identify your app   |
| AppDescription   | Brief description of the application  |
| AppRequiresConfigScreen  | Can be <i>true</i> or <i>false</i> . Show a configuration page in Customize -> Apps   |
| AppRequiresCreatePermission<br>AppRequiresEditPermission<br>AppRequiresViewPermission<br>AppRequiresDeletePermission | Can be set to <i>true</i> or <i>false</i> . Show create, edit, view and delete permission fields in configuration page. Control which user can do which action in the app |

## 6. Project Setup

- a) Replace all Countersoft DLL's in the libs folder with the latest Gemini DLL's from your %Gemini%/bin installation folder.
- b) Open UIApp.cs and change the namespace, class name and all other references from "UIApp" to something unique. Use a name relevant to its function.
- c) Replace the AppGuid and AppControlGuid in Constants.cs with two new GUID's.

Open the app.manifest and make sure the Guid is changed to the AppGuid. You can change the name and description as well. Leave the debug property as "true" for as long as you are developing your app. Change it to "false" when the app is ready to go live.

## 7. Methods and Services

These are the available methods you can override.

| Name              | Description  |
|-------------------|--|
| Show(IssueDto)    | Called when we need to show the content of the app. Requires an IssueDto as a parameter. |
| Caption(IssueDto) | Called every time we show the app's name. Requires an IssueDto as a parameter.           |
| Configuration()   | Override this method to show your own customized configuration view                      |

The app's controller is derived from the main Gemini controller and that means you have access to most of Gemini's services.

```

31 public class UIApp1 : BaseAppController
32 {
33     public override WidgetResult Show(IssueDto issue = null)
34     {
        ..
        ..
        ..
    }

```

List of some important services, which offer methods to retrieve, create, update and delete items.

| Name                   | Description                          |
|------------------------|--------------------------------------|
| ProjectTemplateManager | Provides access to project templates |
| ProjectManager         | Provides access to projects          |
| IssueManager           | Provides access to issues            |

|                        |  |
|------------------------|--|
| UserManager            | Provides access to users                             |
| CustomFieldManager     | Provides access to custom fields                     |
| NavigationCardsManager | Provides access to workspaces                        |
| MetaManager            | Provides access to meta data (status, priority etc.) |

## 8. Permissions

As mentioned earlier you can show an optional [configuration screen](#) in Customize -> Apps -> AppName. This is where you can define which projects the groups have access to.

In the screenshot below you can see how to retrieve the projects for each action, set in the configuration screen.

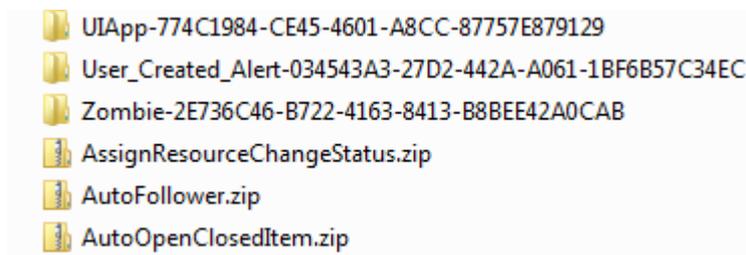
```

30 public class UIApp : BaseAppController
31 {
32     public override WidgetResult Show(IssueDto issue = null)
33     {
34         WidgetResult result = new WidgetResult();
35
36         List<ProjectDto> viewableProjects = ProjectManager.GetAppViewableProjects(this);
37         List<ProjectDto> editableProjects = ProjectManager.GetAppEditableProjects(this);
38         List<ProjectDto> creatableProjects = ProjectManager.GetAppCreateableProjects(this);
39         List<ProjectDto> deletableProjects = ProjectManager.GetAppDeleteableProjects(this);
40     }

```

## 9. Deployment

Apps are deployed to the App\_data/Apps folder which is located where you installed Gemini on your web server.



ZIP files are the packaged apps and the folders are deployed apps.

- We use Post-Build Events in the project to generate the ZIP file. Build your solution and you should find your app's zip file in the bin/target directory.
- Stop your Gemini application pool and copy the ZIP into %Gemini%/App\_Data /apps
- Start your Gemini application pool and this should extract the content of the ZIP into a folder

The app should now be available in Gemini. Make sure all relevant permissions are assigned for the app in Customize -> Apps -> AppName and also enable the app in Customize -> Template -> Process screens.

When you re-deploy the app then make sure the app's ZIP file and the folder are deleted from %Gemini%/App\_Data/apps before you copy in the new ZIP file.

## 10. References Examples

Several Gemini app examples with source code <http://countersoft.github.io/>

Docs <http://docs.countersoft.com/developing-custom-apps/>