The Simplified ObjectScript Source Folder Structure For Package Manager

Evgeny Shvarov 24 January 2020

Deployment, ObjectScript, ObjectScript Package Manager (ZPM), InterSystems IRIS, Open Exchange

Hi Developers!

Recently we released the updated version 0.1.3 of ObjectScript Package Manager (ZPM) which comes with the support of simplified ObjectScript sources folder structure.

What 'simplified' does mean?

Before 0.1.3 ZPM expected the following structure:

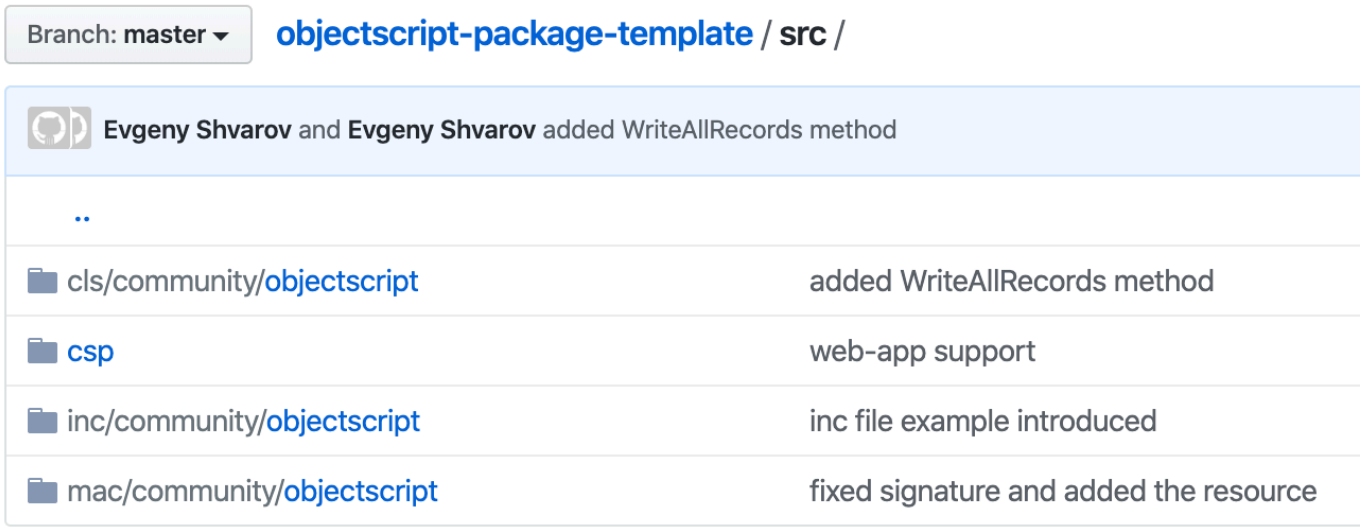
```
/src

---/cls  - for ObjectScript classes

---/cls/package_name/class_name.cls

---/cls/package_name/class_name2.cls

---/mac - or Mac ObjectScript routines

---/mac/package_name/mac_routine.mac

---/mac/package_name/mac_routine2.mac

---/inc - for ObjectScript macro include files.

---/inc/package_name/include_file.inc
```

The example of the structure could be found here and looks like the following:

```
src % tree

.

|____inc

| |____community

| | |____objectscript

| | | |____macroexample.inc

|____cls

| |____community

| | |____objectscript
```

```
| | | |____RestExample.cls

| | | |____ClassExample.cls

| | | |____PersistentClass.cls

|____mac

| |____community

| | |____objectscript

| | | |____MacExample.mac

|____csp

| |____hello.csp

src %
```

Or the same on Github:



And the module.xml for this looks like this (only the part related to resources):

```
<?xml version="1.0" encoding="UTF-8"?>
<Export generator="Cache" version="25">
  <Document name="objectscript-package-template.ZPM">
    <Module>
      <Name>objectscript-package-template</Name>
      <Version>1.0.3</Version>
      <Packaging>module</Packaging>
      <SourcesRoot>src</SourcesRoot>
      <Resource Name="community.objectscript.PKG"/>
      <Resource Name="community.objectscript.macroexample.INC"/>
      <Resource Name="community.objectscript.MacExample.MAC"/>
      <Resource Name="community.objectscript.settings.GBL"/>
...
    </Module>
```

```
  </Document>
</Export>
```

The new ZPM release supports the case where you can put all your ObjectSrcipt sources .cls, .mac, .inc, int into one folder with subfoloders as packages. E.g.:

/src/package_name/class_name.cls

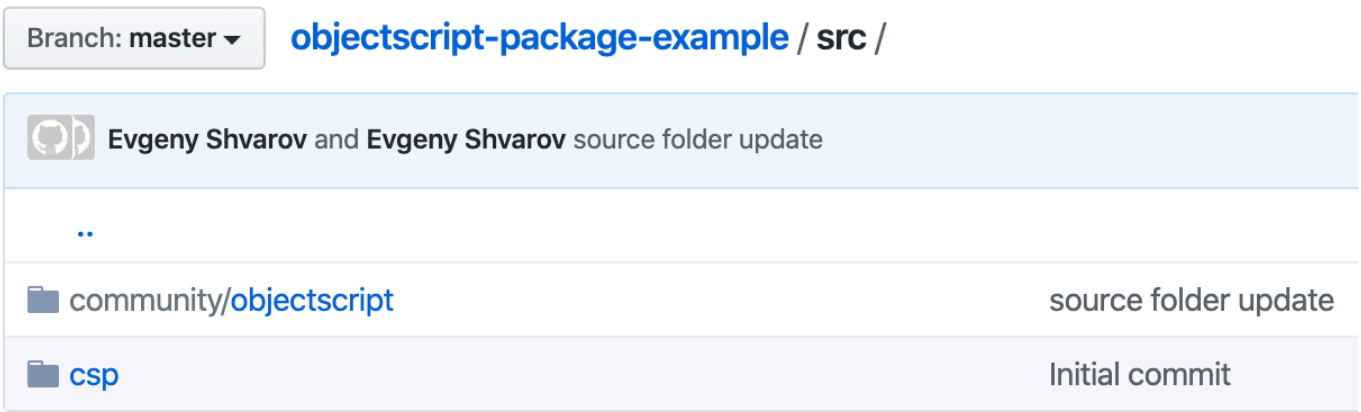
/src/package_name/mac_routine.mac

/src/package_name/include_file.inc

The example of the supported structure could be found here and looks like the following:

```
src % tree

.

|____csp

| |____hello.csp

|____community

| |____objectscript

| | |____RestExample.cls

| | |____ClassExample.cls

| | |____macroexample.inc

| | |____PersistentClass.cls

| | |____MacExample.mac

src %
```

Or the same on Github:



Minus 2 folders (inc, mac) and one subfolder (cls).

Here is the related module.xml (only Resources part):

```
<?xml version="1.0" encoding="UTF-8"?>
<Export generator="Cache" version="25">
  <Document name="objectscript-package-example.ZPM">
    <Module>
      <Name>objectscript-package-example</Name>
      <Version>1.0.0</Version>
      <Packaging>module</Packaging>
      <SourcesRoot>src</SourcesRoot>
      <Resource Name="community.objectscript.PKG"/>
      <Resource Name="community.objectscript.settings.GBL"/>
    </Module>
  </Document>
</Export>
```

If 'inc' and 'mac' files belong to the same ObjectScript package as ObjectScript classes it could be described with one 'package.subpackage.PKG' element.

E.g. the element:

```
 <Resource Name="community.objectscript.PKG"/>
```

gathers into the one package all the following files from '/src/community/objectscript' folder:

```
|____community

| |____objectscript

| | |____RestExample.cls

| | |____ClassExample.cls

| | |____macroexample.inc

| | |____PersistentClass.cls

| | |____MacExample.mac
```

And will gather any new if it appears in the folder.

As a result of the **new supported folder structure**, we are having fewer '<Resource>' elements in 'module.xml' if mac and inc belong to the same package and we don't need to describe any and each of them. And it's not mandatory to use /cls folders - you can start with whatever folder you want - each folder below it pointed in <SourceRoot> element will be considered as ObjectScript package name.

The newly supported format doesn't substitute the previous one -it keeps working. More options - more value!

Choose the structure you like and submit your ObjectScript libraries to ZPM via Open Exchange! Stay tuned!

Continue reading with the next part: Setting Up Your Own InterSystems ObjectScript Package Manager Registry.
Also, check the previous part: Installing Samples BI module with ObjectScript Package Manager.

Check the related application on InterSystems Open Exchange.

- + 4
- 1
- 1
- 168
- 0

Evgeny Shvarov 24 January 2020

Reply

Source URL: https://community.intersystems.com/post/simplified-objectscript-source-folder-structure-package-manager