
Artículo

[Jose-Tomas Salvador](#) · 7 mayo, 2021 Lectura de 3 min

[Open Exchange](#)

Cómo implementar las soluciones de Python embebidas en InterSystems IRIS con el administrador de paquetes ZPM

¡Hola desarrolladores!

Recientemente anunciamos la versión de prueba con Python embebido en InterSystems IRIS - echad un vistazo a [este vídeo en inglés](#) de [@Bob Kuszewski](#).

Python embebido ofrece la posibilidad de cargar y ejecutar el código Python en el servidor de InterSystems IRIS. Podéis utilizar módulos de librerías de Python instalándolos con pip, como NumPy, Pandas, etc., o podéis escribir vuestros propios módulos de Python como archivos .py independientes.

Bien, pero una vez que ya estáis satisfechos con el desarrollo de vuestra solución de Python embebida en IRIS, hay otra cuestión muy importante: cómo desplegar la solución.

Una de las opciones que podéis considerar es utilizar el [Administrador de paquetes ZPM](#), que os voy a describir en este artículo.

Quiero presentaros un [repositorio de plantillas](#) que introduce un módulo ZPM desplegable y muestra cómo construirlo.

El ejemplo es muy sencillo y contiene un [sample.py](#), que demuestra el uso de las librerías Pandas y NumPy de python y la clase [test.cls](#) de ObjectScript que las llama.

La solución podría instalarse con ZPM de la siguiente forma:

```
zpm "install iris-python-template"
```

Nota: aseguraos de que el IRIS en que instaláis el módulo es alguna de las versiones de prueba que ya soportan Python embebido. Por ejemplo, podéis utilizar la imagen:

```
intersystemsdc/iris-ml-community:2020.3.0.302.0-zpm
```

Con los comandos:

```
docker run --rm --name my-iris -d --publish 9091:1972 --publish 9092:52773 intersystemsdc/iris-ml-community:2020.3.0.302.0-zpm
docker exec -it my-iris iris session IRIS
```

```
USER>zpm "install iris-python-template"
```

```
[iris-python-template] Reload START
```

```
...
```

```
[iris-python-template] Activate SUCCESS
```

[El módulo instala](#) el archivo de python sample.py y el archivo de ejemplo titanic.csv junto con test.cls en el sistema.

Por ejemplo sample.py expone la función meanage() que acepta la ruta del archivo csv y calcula el valor medio utilizando las bibliotecas NumPy y Pandas.

La clase ObjectScript test.cls carga el módulo de python con la siguiente [línea de código](#):

```
set tt=##class(%SYS.Python).Import("sample")
```

luego proporciona la ruta al archivo csv y recoge el resultado de la función.

Así es como podéis probar el módulo instalado:

```
USER>d ##class(dc.python.test).Today()  
2021-02-09  
USER>d ##class(dc.python.test).TitanicMeanAge()  
mean age=29.69911764705882  
USER>
```

¡Muy bien! Lo siguiente es: ¿cómo desplegar los módulos de Python embebidos?

Podéis añadir la siguiente [línea a module.xml](#):

```
<FileCopy Name="python/" Target="{mgrdir}python/" />
```

la línea copia todos los archivos de Python desde la carpeta de Python del repositorio a la carpeta de Python dentro de la carpeta /mgr en la instalación de IRIS.

Esto permite importar los módulos de Python desde ObjectScript a través del método `##class(%SYS.Python).Import()`.

Además, si queréis que los archivos de datos se empaqueten en el módulo ZPM, comprobad otra [línea FileCopy](#) en el módulo que importa la [carpeta data](#) desde el repositorio junto con [titanic.csv](#) en el paquete:

```
<FileCopy Name="data/" Target="{mgrdir}data/" />
```

¡Y eso es todo!

¡No dudéis en utilizar la plantilla como base para vuestros proyectos con Python embebido para IRIS!

¡Cualquier pregunta o comentario será bienvenido!

[#InterSystems Package Manager \(IPM\)](#) [#Python](#) [#InterSystems IRIS](#)
[Ir a la aplicación en InterSystems Open Exchange](#)