

Configurar un servicio REST en IAM y publicarlo en el Developer Portal

Artículo

[Alberto Fuentes](#) · Ago 29, 2019



Lectura de 3 min

Configurar un servicio REST en IAM y publicarlo en el Developer Portal

¡Hola a todos!

Hoy me gustaría contaros cómo podemos añadir la documentación de un servicio REST al Developer Portal de InterSystems API Manager.

¿Por dónde empezamos?

Necesitamos tener instalado [InterSystems IRIS 2019.2](#) junto con **InterSystems API Manager (IAM)**.

En el post [Presentación de InterSystems API Manager](#), David nos cuenta cómo instalarlo, configurarlo y hacer una primera prueba.

Crear un servicio REST

A continuación necesitamos implementar el servicio REST que vamos a publicar en IAM.

En este caso vamos a utilizar el servicio REST de ejemplo que se utiliza en el tutorial [Cómo definir interfaces REST en InterSystems IRIS](#).

El código del ejemplo y las instrucciones de instalación están disponibles en <https://github.com/intersystems/FirstLook-REST>

Después de instalar el ejemplo, tendremos disponible una serie de nuevas clases en el paquete *Demo* incluyendo el servicio REST que está en la clase **Demo.CoffeeMakerRESTServer**.

```
CoffeeMakerRESTServer.cls
IRIS-Test Demo CoffeeMakerRESTServer test()
1 /// Use or operation of this code is subject to acceptance of the license available in the code repository for this code.
2 ///
3 Class Demo.CoffeeMakerRESTServer Extends %CSP.REST
4 {
5
6     Parameter HandleCorsRequest = 1;
7
8     XData UrlMap [ XMLNamespace = "http://www.intersystems.com/urlmap" ]
9     {
10     <Routes>
11     <Route Url="/test" Method="GET" Call="test"/>
12     <Route Url="/coffeemakers" Method="GET" Call="GetAll" />
13     <Route Url="/coffeemaker/:id" Method="GET" Call="GetCoffeeMakerInfo" />
14     <Route Url="/newcoffeemaker" Method="POST" Call="NewMaker" />
15     <Route Url="/coffeemaker/:id" Method="PUT" Call="EditMaker" />
16     <Route Url="/coffeemaker/:id" Method="DELETE" Call="RemoveCoffeeMaker"/>
17     </Routes>
18     }
19
20     /// Tester method
21     ClassMethod test() As %Status
22     {
23         Set cm = {
24             "coffeemakerID": "20",
25             "name": "My Coffee Maker",
26             "brand": "Coffee Inc",
27             "price": "23.45",
```

Configurar un servicio REST en IAM y publicarlo en el Developer Portal

Published on InterSystems Developer Community (<https://community.intersystems.com>)

Si hemos cargado y configurado correctamente el ejemplo deberíamos poder acceder a sus servicios, por ejemplo :

```
$ curl http://localhost:52773/rest/coffeemakerapp/test
```

```
{"coffeemakerID":"20","name":"My Coffee Maker","brand":"Coffee Inc","price":"23.45","numcups":"1","color":"green","img":"img/coffee1.png"}
```

Documentación del servicio REST

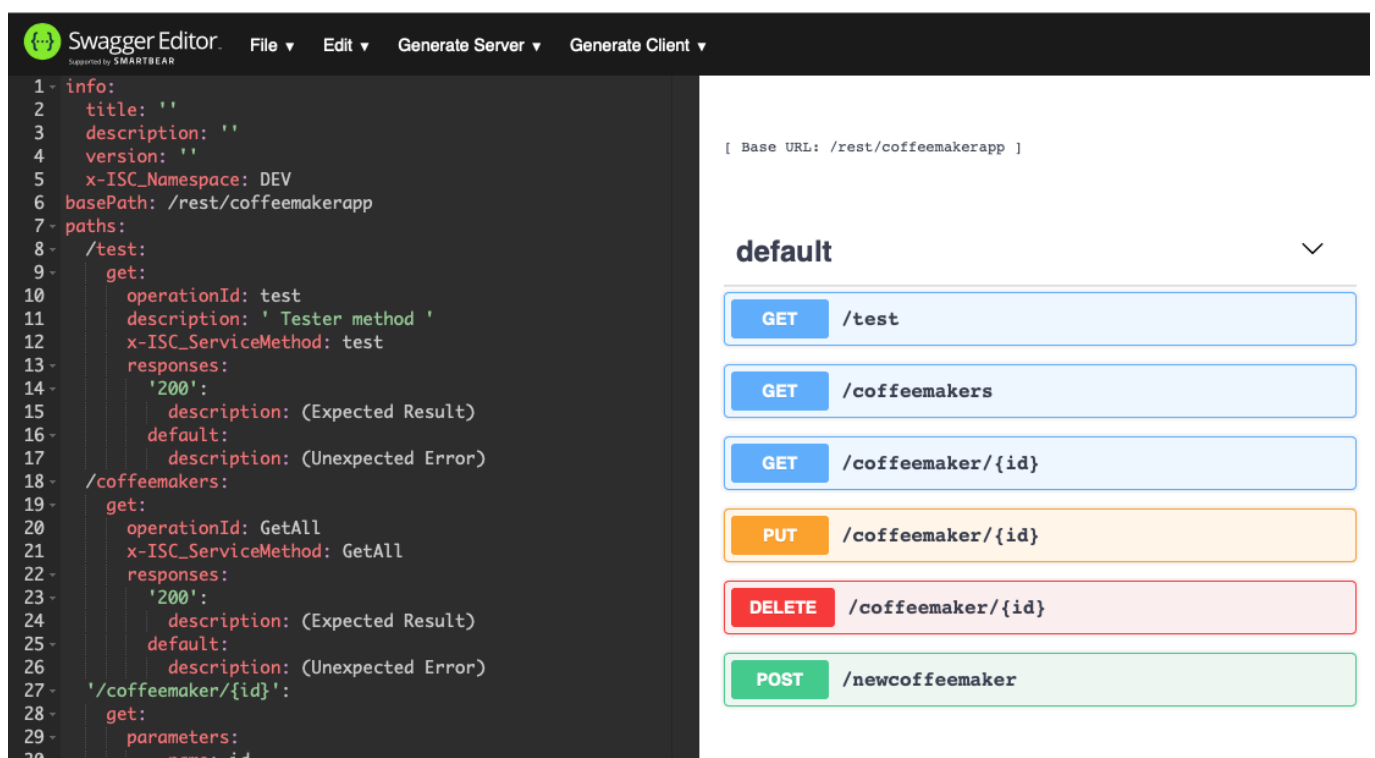
Cuando implementamos un servicio REST nos interesa poder facilitar la documentación (**especificaciones**) de cómo funciona para que otros desarrolladores puedan utilizarlo.

InterSystems IRIS nos permite obtener las especificaciones de nuestros servicios REST en formato [Open API](#). No sólo eso, sino que además también nos permite crear un esqueleto para implementar el servicio a partir de sus especificaciones. Pudiendo así trabajar tanto con el paradigma **Design-First** or **API-First**. Si os interesan estas funcionalidades, no dejéis de echarle un vistazo a [La aplicación IRIS API Explorer](#).

En el caso que nos ocupa, obtendremos las especificaciones (en JSON) de nuestro recién creado servicio REST en la URL en:

```
http://<server>:<port>/api/mgmt/v1/<namespace>/spec/rest/coffeemakerapp/
```

Podemos copiar el JSON con la especificación y visualizarlo directamente en alguna herramienta *online* como [Swagger Editor](#). Al pegar el contenido en Swagger Editor, éste lo convierte automáticamente de *JSON* a *yaml*:



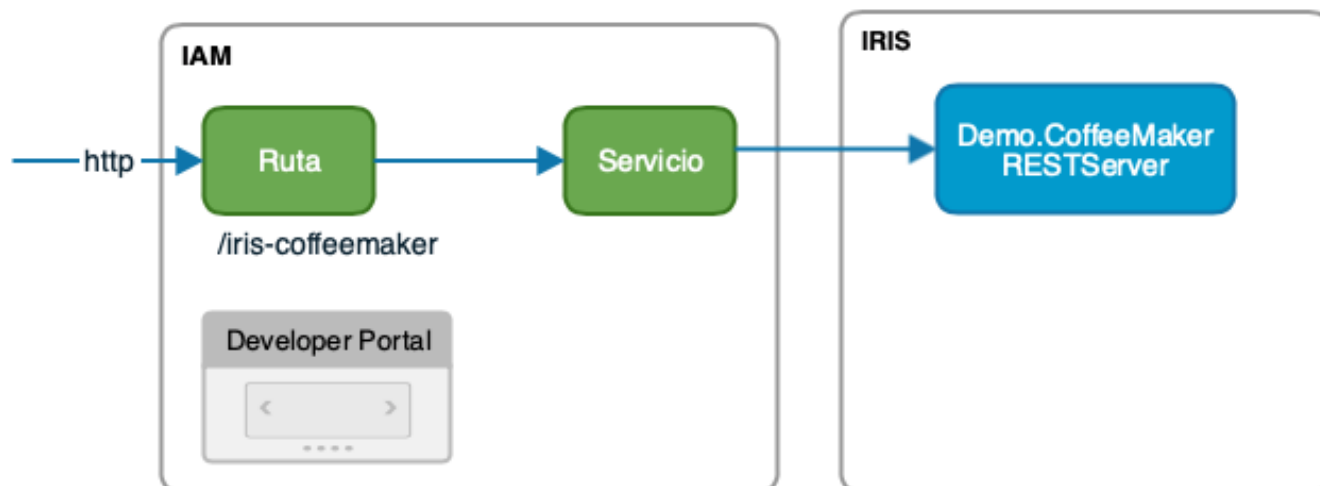
The screenshot displays the Swagger Editor interface. On the left, a code editor shows the API specification in YAML format. The specification includes an 'info' section with title, description, and version, a 'basePath' of '/rest/coffeemakerapp', and several 'paths' with their respective HTTP methods and descriptions. The paths include a test endpoint, a list of coffee makers, and a single coffee maker endpoint with GET, PUT, and DELETE methods, as well as a new coffee maker endpoint with a POST method.

```
1 - info:
2   title: ''
3   description: ''
4   version: ''
5   x-ISC_Namespace: DEV
6   basePath: /rest/coffeemakerapp
7 - paths:
8   /test:
9     get:
10      operationId: test
11      description: 'Tester method '
12      x-ISC_ServiceMethod: test
13      responses:
14        '200':
15          description: (Expected Result)
16        default:
17          description: (Unexpected Error)
18   /coffeemakers:
19     get:
20      operationId: GetAll
21      x-ISC_ServiceMethod: GetAll
22      responses:
23        '200':
24          description: (Expected Result)
25        default:
26          description: (Unexpected Error)
27   '/coffeemaker/{id}':
28     get:
29       parameters:
30         - name: id
```

On the right, the Swagger Editor visualizes the API specification as a list of endpoints. The base URL is '/rest/coffeemakerapp'. The endpoints are:

- GET /test
- GET /coffeemakers
- GET /coffeemaker/{id}
- PUT /coffeemaker/{id}
- DELETE /coffeemaker/{id}
- POST /newcoffeemaker

Configurar un servicio y una ruta en IAM



Servicio

En IAM, un **servicio** es una API que queremos exponer a consumidores externos. Así que en primer lugar, vamos a crear un servicio que se comuniquen con el servicio REST que acabamos de crear.

Podemos crear el servicio utilizando la API REST de gestión de IAM:

```
curl -i -X POST --url http://localhost:8001/services/ --data 'name=iris-coffee-service' --data 'url=http://172.24.28.166:52773/rest/coffeemakerapp'
```

O también podemos crearlo igualmente a través del portal de gestión:

iris-coffee-service

connect_timeout	60000
created_at	August 28th 2019, 3:11:27pm
host	172.24.28.166
id	c828f884-5799-41d9-a12a-51ac92165e9e
name	iris-coffee-service
path	/rest/coffeemakerapp
port	52773
protocol	http
read_timeout	60000
retries	5
updated_at	January 19th 1970, 4:16:37am
write_timeout	60000

Ruta

Una **ruta** es la manera de hacer que las peticiones que lleguen a IAM se redirijan a un servicio concreto. En nuestro caso crearemos una ruta que encamine al servicio REST que hemos creado aquellas peticiones que se dirijan a */iris-coffeemaker*.

Igual que los servicios, las rutas podemos crearlas haciendo uso de la API de gestión:

```
curl -i -X POST --url http://localhost:8001/services/iris-coffee-service/routes --data 'paths[]=/iris-coffeemaker'
```

o a través del portal de gestión:

Viewing Route

created_at	August 27th 2019, 11:55:30am
hosts	
id	5747cc78-1c13-47c8-a3bb-54e2b3c128e1
methods	
paths	/iris/test-service
preserve_host	false
protocols	http
regex_priority	0
service	iris-soap-test
strip_path	true
updated_at	August 27th 2019, 12:39:54pm

Con el servicio y la ruta configurados en IAM, podemos ahora invocar el método de prueba utilizando la ruta que acabamos de configurar.

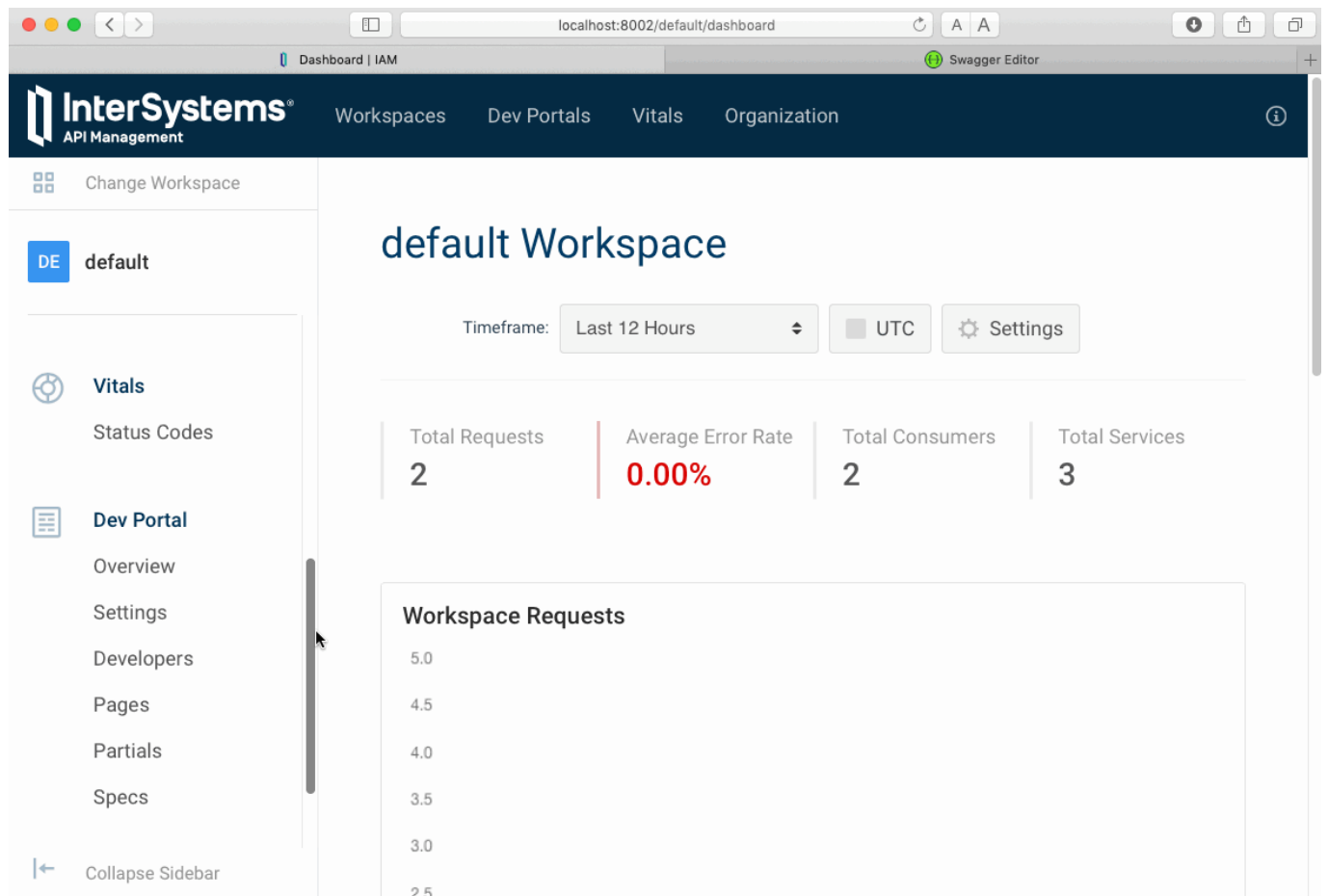
```
$ curl http://localhost:8000/iris-coffeemaker/test
```

```
{"coffeemakerID":"20","name":"My Coffee Maker","brand":"Coffee  
Inc","price":"23.45","numcups":"1","color":"green","img":"img/coffee1.png"}
```

Publicar especificaciones en el Developer Portal

Por último, sólo falta añadir una nueva especificación de un servicio en IAM y se publicará en el Developer Portal:

- IAM (portal de gestión) > Specs > Añadir la nueva especificación (podemos copiarla desde Swagger o el JSON que nos devolvía IRIS).
- Accedemos al Developer Portal y ¡ya tenemos disponible la especificación de nuestro servicio!



[#API](#) [#API REST](#) [#InterSystems API Manager \(IAM\)](#) [#InterSystems IRIS](#)
20 2 0 1 191

Log in or sign up to continue
Añade la respuesta

URL de fuente: <https://es.community.intersystems.com/post/configurar-un-servicio-rest-en-iam-y-publicarlo-en-el-developer-portal>