

Artículo

[Eduardo Anglada](#) · 15 jul, 2021 Lectura de 5 min

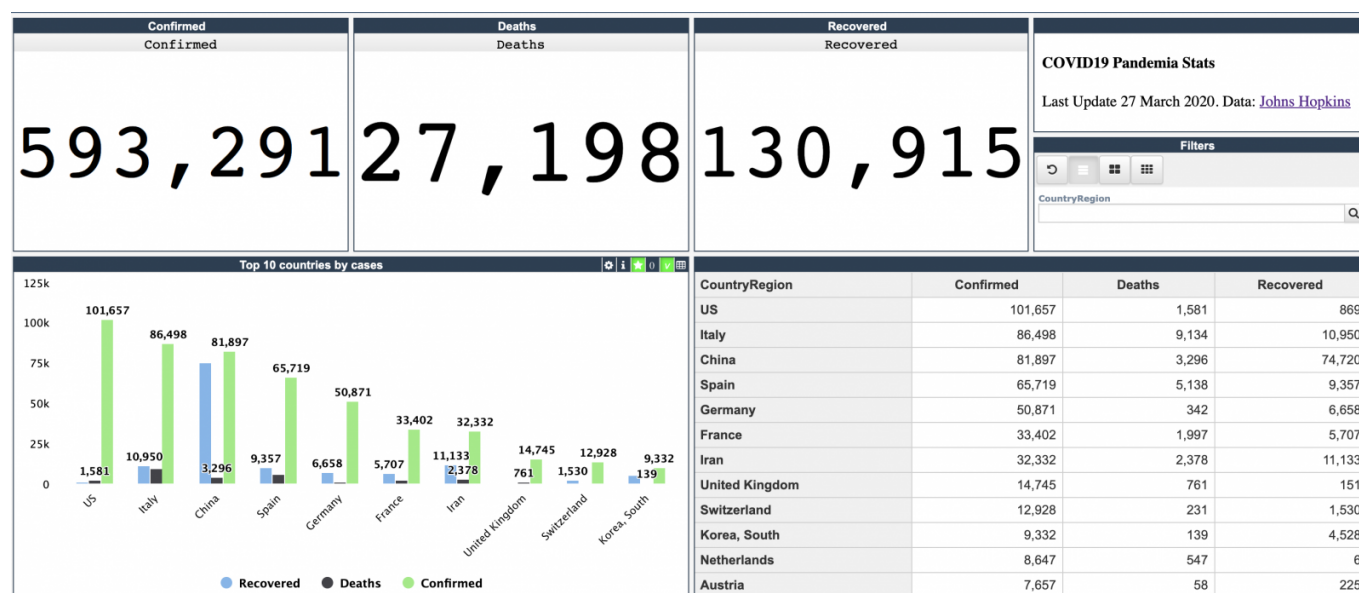
[Open Exchange](#)

## Estadísticas sobre COVID-19 con InterSystems IRIS

¡Hola a todos!

Todos los días la universidad Johns Hopkins publica nuevos datos sobre el estado de la pandemia del coronavirus COVID-19.

Creé un [sencillo panel de control con InterSystems IRIS Analytics](#) usando InterSystems IRIS Community Edition en Docker, implementado en Kubernetes GCP, para mostrar los datos clave sobre el avance de la enfermedad.



Este panel de control es un ejemplo de como la información en formato CSV podría ser analizada con IRIS Analytics e implementada en Kubernetes GCP, con la forma de InterSystems IRIS Community Edition.

Añadí el [mapa interactivo de Estados Unidos](#):

En el [siguiente panel de control se muestra la línea de tiempo](#):

Y se puede filtrar por país. Por ejemplo [en Estados Unidos](#):

El código fuente de la aplicación está [disponible en Open Exchange](#).

¿Cómo funciona?

La demo se ejecuta usando el contenedor Docker de InterSystems IRIS Community Edition y muestra los paneles de control de la Inteligencia empresarial (BI) de InterSystems IRIS usando la capa de representación DeepSee

Web con la API REST MDX2JSON. Se implementa en GCP y funciona con Google Kubernetes Engine (GKE).

¿Cómo se desarrolló?

Los datos se toman del [repositorio de Johns Hopkins](#) en formato CSV.

Las clases, los cubos y las tablas dinámicas iniciales se generaron con el módulo [Analyzethis](#). ¡Gracias, @Peter Steiwer!

El método de importación se introdujo con el método CSVTOCLASS. ¡Gracias, @Eduard Lebedyuk!

Los paneles de control se renderizan con el módulo [DeepSee Web \(DSW\)](#).

Los [artefactos \(tablas dinámicas, paneles de control\)](#) de la Inteligencia empresarial (BI) de IRIS fueron exportados con el módulo [ISC.DEV](#):

```
IRISAPP> d ##class(dev.code).workdir("/irisdev/app/src")
```

```
IRISAPP> d ##class(dev.code).export("*.dfi")
```

El código fue desarrollado utilizando [VSCode ObjectScript](#). ¡Gracias, @Dmitry Maslennikov!

Cómo construir imágenes en Docker para desarrollar e implementar

Toda la implementación se encuentra en el [Dockerfile](#). Con Dockerfile generamos una imagen con datos, aplicaciones web y módulos instalados y configurados correctamente y después implementamos la imagen en Kubernetes GCP.

Este Dockerfile es una versión modificada de esta [plantilla Dockerfile](#), que se describe muy bien [en este artículo](#).

Todos los pasos preliminares se realizan en el archivo [iris.script](#):

Aquí instalamos DeepSee Web.

```
zpm "install dsw"
```

Esto habilita a IRIS Analytics (DeepSee) para la aplicación web, csp e irisapp:

```
do EnableDeepSee^%SYS.cspServer("/csp/irisapp/")
```

Este código es necesario para que la aplicación web de análisis esté disponible sin credenciales:

```
zn "%SYS"
write "Modify MDX2JSON application security...",<!--
set webName = "/mdx2json"
set webProperties("AuthEnabled") = 64
set webProperties("MatchRoles")="%DB_IRISAPP"
set sc = ##class(Security.Applications).Modify(webName, .webProperties)
if sc<1 write $SYSTEM.OBJ.DisplayError(sc)
```

Y aquí en [Dockerfile](#) este comando ayuda a establecer la configuración de DSW.

```
COPY irisapp.json /usr/irissys/csp/dsw/configs/
```

## Implementación en Kubernetes

El procedimiento de implementación es procesado por Github Actions, y [este flujo de trabajo](#) lo administra en cada asignación al repositorio.

El flujo de trabajo de Github utiliza el Dockerfile que construimos en un paso anterior junto con la configuración de [Terraform](#) y [Kubernetes](#).

El procedimiento es idéntico al descrito en [este artículo](#) de @Mikhail Khomenko.

## Cómo ejecutarlo y desarrollarlo de forma local

Todo el mundo está invitado a ejecutar, desarrollar y colaborar en este proyecto.

Para ejecutarlo de forma local usando Docker hay que hacer lo siguiente:

Clonar el repositorio en cualquier directorio local:

```
$ git clone https://github.com/intersystems-community/objectscript-docker-template.git
```

Abrir el terminal en este directorio y ejecutar:

```
$ docker-compose build
```

Ejecutar el contenedor de IRIS:

```
$ docker-compose up -d
```

Una vez que el contenedor esté construido y funcionando, abre la aplicación en:

```
localhost:yourport/dsw/index.html#/irisapp
```

## Cómo desarrollarlo

Este repositorio está listo para codificarse en VSCode con el plugin ObjectScript. Instala [VSCode](#), [Docker](#) y [el plugin ObjectScript](#) y abre la carpeta en VSCode.

## Como contribuir

Crea un fork del [repositorio](#), haz cambios y envía pull requests. [En este video tienes más información.](#)

¡Nos encantaría ver vuestras contribuciones!

[#Analítica](#) [#CSV](#) [#Docker](#) [#GCP](#) [#Kubernetes](#) [#ObjectScript](#) [#InterSystems Package Manager \(IPM\)](#) [#InterSystems IRIS](#) [#InterSystems IRIS BI \(DeepSee\)](#)  
[Ir a la aplicación en InterSystems Open Exchange](#)

