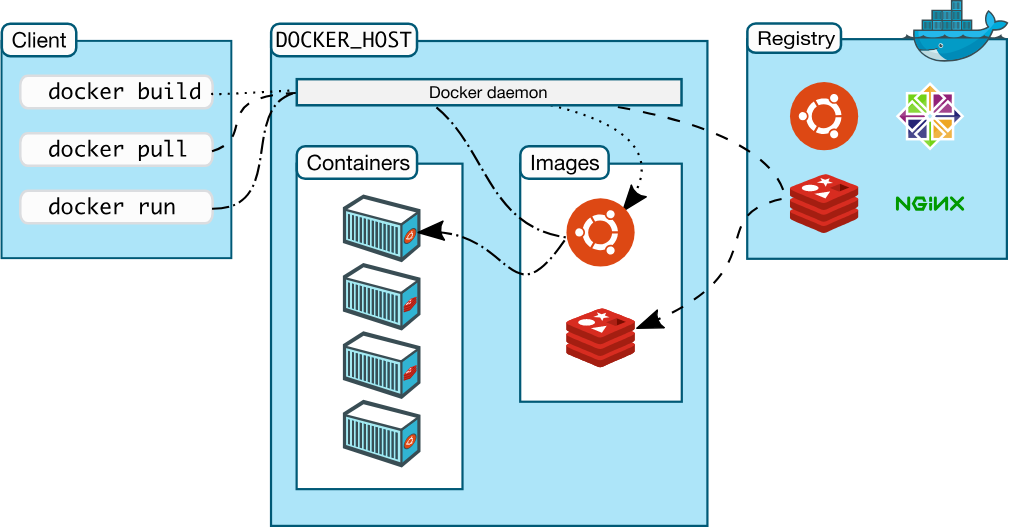
**Arquitectura de software como línea base.**

Antes de comenzar a escribir software deberíamos de tener un objetivo de lo que haremos, después un buen diseño, posterior a eso podemos iniciar con el proceso de desarrollo y terminamos con pruebas y mejora continua.

Más que bueno, el diseño de lo que vamos a desarrollar debe de estar integrado como parte de nuestro desarrollo; es decir la arquitectura del software como tal también es código, son configuraciones de aplicaciones web, configuraciones de servicios web, estructura de bases de datos, balanceadores de carga, firewalls y muchas cosas que en su mayoría ya pueden partir de líneas de código o scripts.

A la buena práctica de tener todo en un mismo lugar, la definición de la arquitectura como script, la estructura de la base de datos como proyecto, los proyectos de nuestra aplicación, así como la documentación de todo le llamamos **Línea Base.**

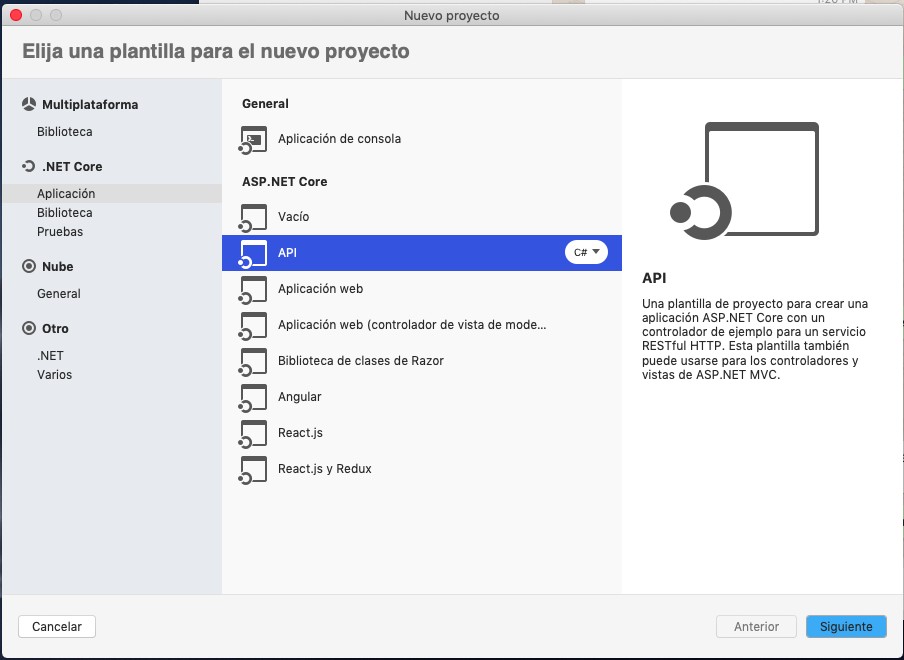
Entonces deberíamos de poder integrar esas líneas de código al desarrollo propio de nuestro software, un buen ejemplo de este tipo de desarrollo es Docker. Con Docker podemos crear ambientes de desarrollo completo con una o varias máquinas que se vean entre si, permisos, roles y otros aspectos importantes que en un principio pensaríamos que sería trabajo de la gente de infraestructura; dejando el ambiente listo para posteriormente publicarle nuestro código o nuestros archivos binarios.



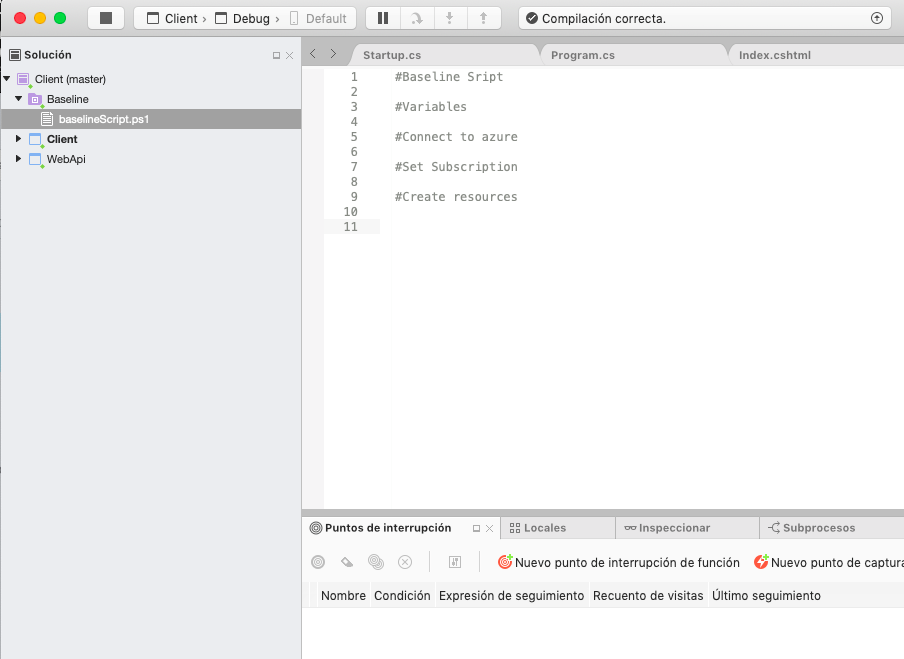
A continuación, haremos un ejemplo de la creación de una sencilla aplicación, pero con la creación del ambiente en el que la publicaremos publicar la misma, de esta forma integramos los requerimientos de la infraestructura como parte de nuestra misma aplicación de una forma que es mantenible, apegada al diseño en la que nosotros desarrolladores / arquitectos de software nos hacemos responsables de la Línea Base Completa.

El ejemplo nos enfocaremos en creación de un proyecto en Net Core con c# en Azure en simples WebApps siendo creadas con Azure client, en otro artículo posterior haremos un ejemplo con Docker.

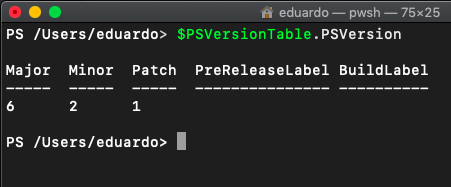
Primero crearemos un proyecto web y posteriormente un proyecto WebApi que trabajen juntos.



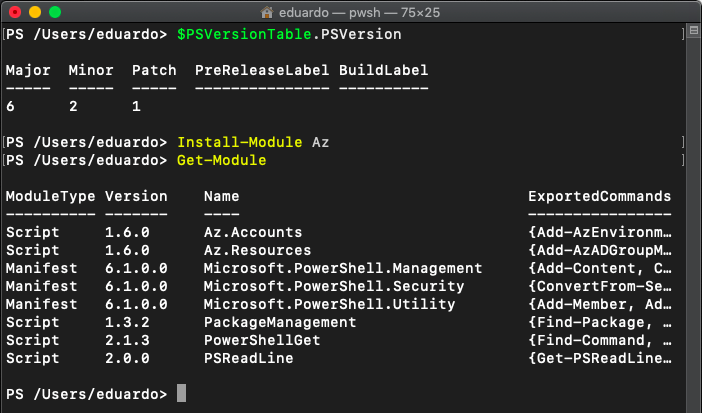
Sobre la misma solución creamos una carpeta en la que podemos colocar el archivo que usaremos para crear nuestra línea base, el archivo le ponemos la extensión de PowerShell.



Vamos a trabajar con el cliente de azure, en mi caso usaré un equipo con Mac OS instalando directo sobre la terminal y tengo instalada la versión 6 de PowerShell



También instalo el módulo de Az



Creamos el script de PowerShell como sigue:

#Baseline Sript

#Variables

$vUserName = ""

$vPassword = ""

$vTentant = ""

$subscription = ""

$randomName = -join ((48..57) + (97..122) | Get-Random -Count 10 | % {[char]$\_}) #Random string to help us to create individual names in the cloud

$webAppName = "exampleApp"+$randomName #The Name of the WebApp

$webApiName = "exampleApi"+$randomName #The Name of the WebApi

$resourceGroupName = "exampleGroup" +$randomName

$appServicePlanName = "exampleAppServicePlan" +$randomName

$region = "eastus"

$tier = "B1"

#Connect to azure

#SecurePassword converting the password to a secure string

$passwd = ConvertTo-SecureString $vPassword -AsPlainText -Force

az login -u $vUserName -p $vPassword --tenant $vTentant

#Set Subscription where we’ll be work

az account set --subscription $subscription

#Create resources

#Create Resource Group

az group create --location $region --name $resourceGroupName

#Create App Service Plan

az appservice plan create --name $appServicePlanName --resource-group $resourceGroupName --is-linux --sku $tier

#Create Web App

az webapp create --name $webAppName --plan $appServicePlanName --resource-group $resourceGroupName --runtime "DOTNETCORE|2.2"

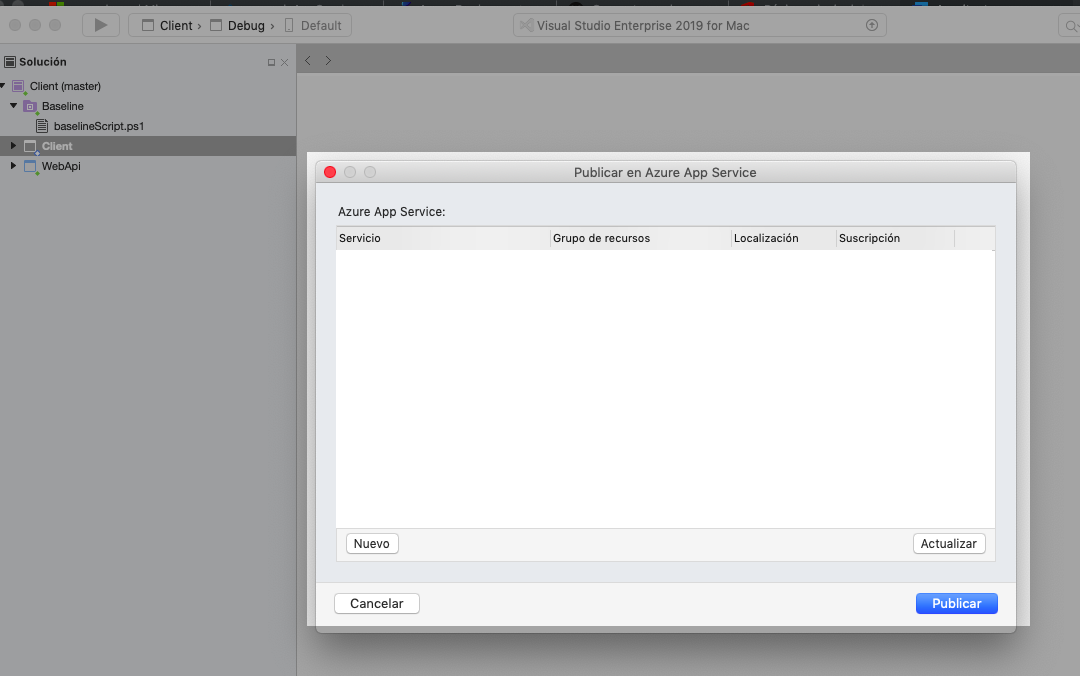
#Create Web Api

az webapp create --name $webApiName --plan $appServicePlanName --resource-group $resourceGroupName --runtime "DOTNETCORE|2.2"

Este Script nos permite conectarnos a azure, crear el Grupo de Recursos, el App Service Plan para soportar aplicaciones de linux y la Web App con NetCore 2.2 como tiempo de ejecución.

La idea es incluir este script en la solución para que junto con nuestro proyecto tengamos las instrucciones de cómo debe de ser creada nuestra arquitectura, de la misma forma podemos agregar configuraciones que normalmente crearía el Ingeniero de Infraestructura basado en su criterio o con ayuda del arquitecto, de esta forma la arquitectura queda documentada junto con la solución.

Una vez creada la arquitectura en la Nube de Azure podemos configurar nuestra solución para hacer las publicaciones o configurar nuestras definiciones de release y builds en nuestro proyecto de DevOps de Microsoft. En este ejemplo usaremos el Visual Studio en MAC



En un escenario de una aplicación empresarial normalmente se diseña la aplicación por un arquitecto, se desarrolla por un programador, se implementa la integración y liberación continua por un ingeniero de DevOp y finalmente se le da mantenimiento a la infraestructura. El problema de este modelo es que el diseño de la arquitectura, el código fuente, la línea base y los requerimientos técnicos quedan aislados de esta forma queremos dar un ejemplo de cómo la línea base que es una imagen de la arquitectura puede convivir en la misma solución de nuestro código para que los Ingenieros de DevOps puedan darle mantenimiento de la misma forma que se lo damos al código fuente.

Luis Eduardo Estrada

https://www.eduardo.mx

<https://github.com/internetgdl/ArchitectureAsBaseLine/>

<https://www.linkedin.com/in/luis-eduardo-estrada/>

Referencias:

<https://docs.microsoft.com/en-us/cli/azure/?view=azure-cli-latest>

<https://docs.microsoft.com/en-us/powershell/azure/install-az-ps?view=azps-2.4.0>