Running PowerShell Core using Docker

More and more of the Microsoft ecosystem is making its way into open source platforms. One of the very interesting products coming from the Microsoft camp lately is the PowerShell Core platform which is now ported to run on multiple underlying operating system environments.

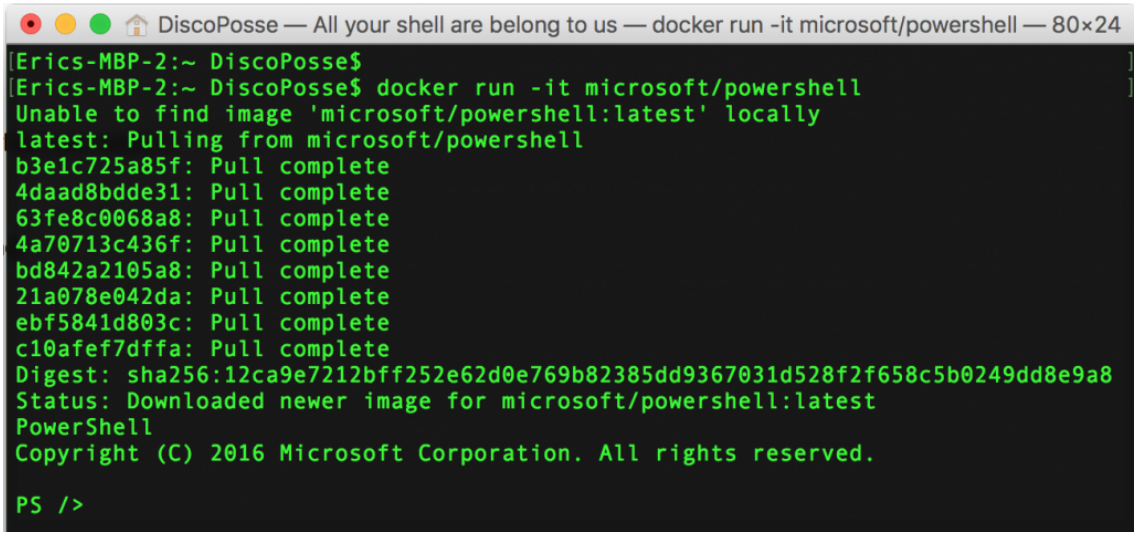
I covered the process to install the Mac OSX version which is very cool, but let's take the abstraction one level higher and look at running the PowerShell core inside a Docker candidate.

The first thing you'll want to do is head on over here to make sure you're running the nifty Docker Toolbox for your laptop or desktop environment if you haven't already got Docker available to use.

Running your first PowerShell Core container

The commands here may seem a little too easy, but that's by design. The containerized implementation makes the deployment and use of PowerShell core super easy!

Let's launch our first container with the `docker run -it microsoft/powershell` that will kick up a new container based on the image which is in the Docker public hub under the Microsoft organization. The `-it` means that we are launching in an interactive mode inside the container.



That gets you up and running to be able to run the PowerShell environment.

NOTE: There is still limited functionality compared to the full PowerShell on Microsoft environments. This is something that is changing with each release as the community and Microsoft themselves contribute towards more features.

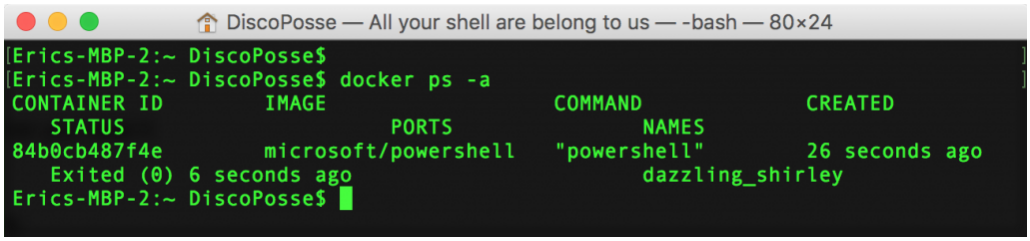
Exiting and re-entering the container

Getting out of the container is as easy as typing `exit` and the prompt. This will bring you back out to the local environment. That gives is an interesting situation where we have the container present, but it is stopped. If you run the same command as you did before, that actually launches an entirely new container.

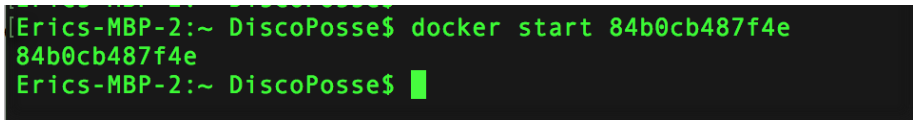
We need to do three things in order to get back in to the same container:

1. find the ID of the existing container
2. start the container using that ID
3. attach to the container

First, let's check the containers to find out the ID of the one we want using the `docker ps -a` command:



Use the `docker start [CONTAINER-ID]` command where `[CONTAINER-ID]` is the ID you see in your console:



Use the same ID and attach to the now active container with the `docker attach [CONTAINER-ID]` command:

That is all there is to it! Each time you exit, the container will automatically stop because we don't need to keep it running in the background. There are other ways to keep it running, but that is for another blog post

Removing the container is as simple as running the `docker rm [CONTAINER-ID]` where `[CONTAINER-ID]` is the ID we used before to attach to the existing container.