INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA PARAÍBA Campus Cajazeiras

Curso: Analise e Desenvolvimento de Sistemas	Campus: Cajazeiras – PB	Data: 29/05/2017
Professor: Ricardo de Sousa Job.	**Disciplina:** Programação Orientada a Serviços – 2017.1	
Aluno: Aluísio José Pereira.		

1) Qual a diferença entre o ENTRYPOINT e CMD?

Ao trabalhar com Dockerfile com uma série de comandos que de certa forma nos parece redudantes algo parecido com isso é os comandos ENTRYPOINT e o CMD. Portanto vamos tentar enteder quais são suas peculiaridades na tentativa de diferenciá-los. Tanto o ENTRYPOINT como o CMD permitem que você especifique o comando de inicialização de uma imagem, mas há diferenças sutis entre eles. Muitas vezes você quer escolher uma ou outra, mas elas também podem ser usadas em conjunto. Em uma análise mais comparativa temos que tanto o ENTRYPOINT como CMD oferecem uma maneira de identificar qual executável deve ser executado quando um contêiner é iniciado a partir da sua imagem. Na verdade, se você deseja que sua imagem seja executável (sem argumentos de linha de comando adicionais **docker run**), você deve especificar um ENTRYPOINT ou CMD.
Sendo assim, temos que o CMD basicamente define o comando e/ou os parâmetros padrão, que podem ser substituídos da linha de comando quando o contêiner do docker é executado. Já o ENTRYPOINT configura um recipiente que será executado como um executável.

Tentando executar uma imagem que não tenha um ENTRYPOINT ou CMD declarado resultará em um erro.

```
$ docker run alpine

FATA[0000] Error response from daemon: No command specified
```

Sabento que o ENTRYPOINT ou CMD que você especifica no seu Dockerfile identifica o executável padrão para sua imagem. Temos que as principais diferenças entre a execulsão do ENTRYPOINT e do CMD resulta da flexibilidade de escolher o execultável desejavel ao iniciar um conteiner.

Por exemplo, digamos que temos o seguinte Dockerfile

```
FROM ubuntu:trusty

CMD ping localhost
```

Se construímos esta imagem (com a tag "demo") e executá-la, veremos a seguinte saída:

```
$ docker run -t demo

PING localhost (127.0.0.1) 56(84) bytes of data.

64 bytes from localhost (127.0.0.1): icmp_seq=1 ttl=64 time=0.051 ms

64 bytes from localhost (127.0.0.1): icmp_seq=2 ttl=64 time=0.038 ms

^C
```

```
--- localhost ping statistics ---

2 packets transmitted, 2 received, 0% packet loss, time 999ms

rtt min/avg/max/mdev = 0.026/0.032/0.039/0.008 ms
```

Percebendo assim que o *ping* foi executado automaticamente quando o contêiner foi iniciado.

Por outro lado sendo possível substituir o CMD padrão, especificando um argumento após o nome da imagem ao iniciar o contêiner:

```
$ docker run demo hostname

6c1573c0d4c0
```

Execultando assim o nome do host no lugar do *ping*.

Já com ENTRYPOINT teriamos

```
$ docker run --entrypoint hostname demo

075a2fa95ab7
```

Passível assim de perceber que seria mais facil substituir o CMD em constraste com o ENTRYPOINT, sendo assim, este último deve ser usado em cenários em que você deseja que o contêiner se comporte de forma exclusiva como se fosse o executável que está envolvido. Ou seja, quando você não quer ou espera que o usuário substitua o executável que você especificou. Cabe destacar que existem casos que faz muito sentido usar ambos juntos para especificar o executável padrão de dada imagem a combinação de ENTRYPOINT e CMD permite que você especifique o executável padrão para sua imagem ao mesmo tempo que fornece argumentos padrão para aquele executável que pode ser substituído pelo usuário. Exemplo.

```
FROM ubuntu:trusty

ENTRYPOINT ["/bin/ping","-c","3"]

CMD ["localhost"]
```

REFERÊNCIAS

CenturyLink. **Dockerfile:** ENTRYPOINT vs CMD. Disponível em: <https://www.ctl.io/developers/blog/post/dockerfile-entrypoint-vs-cmd/>. Acesso em: 29 mai. 2017.