Macoratti.net Docker - Uma introdução básica - I

Neste artigo vou apresentar os conceitos básicos relativos ao Docker sob o ponto de vista de um desenvolvedor .NET.

'Docker é uma plataforma que permite "criar, enviar e executar qualquer aplicativo, em qualquer lugar".
Ele tornou-se muito popular em pouco tempo e hoje é considerado um padrão na maneira de resolver um dos aspectos mais caros do software: a implantação.'

Em palavras mais simples, o Docker é uma ferramenta que permite que desenvolvedores, administradores de sistemas, etc. implantem facilmente seus aplicativos em uma sandbox (chamada contêiner) para serem executados em sistema operacional host, ou seja, no Linux, Windows ou Mac.

O principal benefício do Docker é que ele permite que os usuários empacotem um aplicativo com todas as suasdependências em uma unidade padronizada para desenvolvimento de software. Ao contrário das máquinas virtuais, os contêineres não têm a alta sobrecarga e, portanto, permitem um uso mais eficiente do sistema e dos recursos subjacentes.

A palavra docker traduzida para o português significa estivador ou trabalhador portuário.

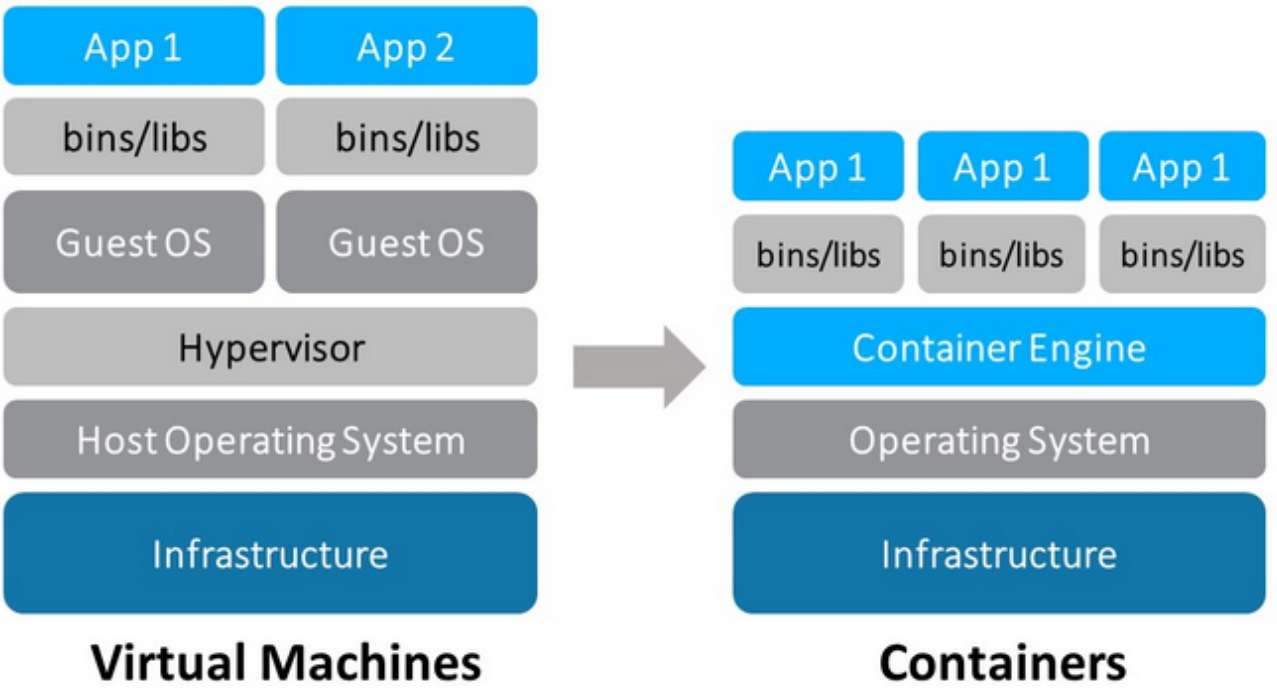
O significado da palavra remete aos contêineres que são carregados nos portos e ao fato do Docker chamar de contêineres os espaços reservados na memória que são executados independentemente e isoladamente de outros contêineres ou do próprio host.

Assim, o Docker é uma tecnologia de virtualização de ambientes parecida com uma máquina virtual. É uma tecnologia open-source escrita na linguagem Go.

Então qual a diferença entre Contêiner e Máquina Virtual

A diferença é que um contêiner no docker compartilha o Kernel com o sistema operacional do host. Isso faz com que o desempenho aumente e o consumo de memória do contêiner diminua.

Na figura abaixo podemos ver a diferença entre um Contêiner Docker uma máquina virtual:



Podemos notar que um contêiner Docker não precisa de um Hypervisor nem de um Sistema Operacional(SO) completo para funcionar; ele compartilha o Kernel a partir do SO do host.

Assim o overhead de um contêiner é o mínimo necessário, pois cada contêiner normalmente carrega apenas um processo, que é aquele responsável pela entrega do serviço desejado.

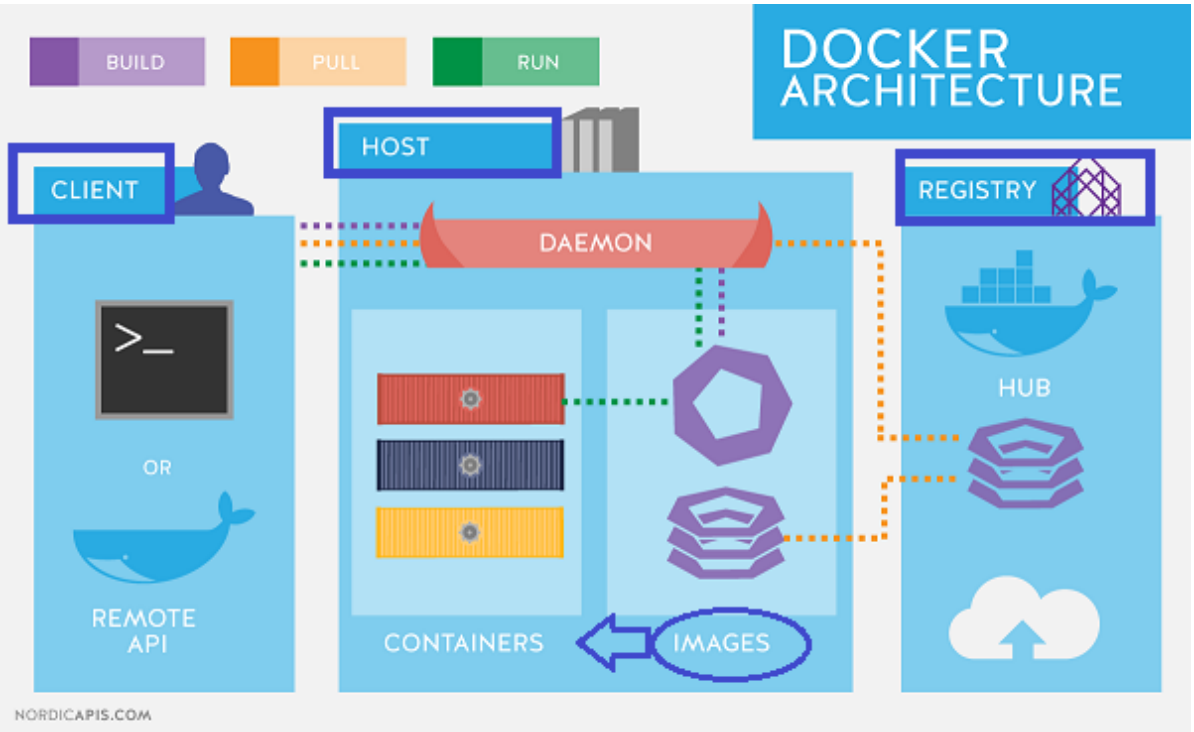
Dessa forma o Docker pode ser visto como um serviço de adminstração de contêineres, que são processos segregados a partir dos quais podemos configurar, inicializar e construir aplicações de forma isolada do sistema host e dos demais contêineres.

Para poder ser executado, um contêiner possui associado a ele um sistema de arquivos completo e isolado que contêm as dependências e bibliotecas necessárias.

Este sistema de arquivos somente-leitura são conhecidos como Imagens; e, a partir das imagens é que os contêiners são criados.

Arquitetura Docker

Na figura abaixo temos uma visão resumida da arquitetura Docker:



Observe que temos um Registry que é um repositório de Images Docker a partir de onde podemos obter imagens prontas para criar os nosso contêineres. No endereço https://hub.docker.com/ temos um repositório com centenas de imagens prontas que podemos baixar e usar para criar contêineres.

Nota: *No endereço https://hub.docker.com/r/microsoft/dotnet/ temos imagens oficiais para .NET Core e ASP .NET Core para Linux e Windows Nano Server.*

No Host temos a imagem que foi baixada do repositório, ou criada por você. A partir dela criamos o processo que é oContêiner Docker.

Podemos acessar esse contêiner de um Client via linha de comando ou via API Remota.

Que fique bem claro que um Contêiner esta associado a um processo na memória enquanto que a imagem pode ser pensada como um modelo de arquivo com um sistema de arquivos a partir do qual um contêiner é criado.

Para que serve o Docker ?

O objetivo do Docker é criar, testar e implementar aplicações em um ambiente separado da máquina original, chamado de contêiner, onde o desenvolvedor consegue empacotar sua aplicação de forma padrão com níveis de isolamento.

Mas qual a vantagem do Docker ???

Segue algumas vantagens do Docker:

1. Os contêineres são uma imagem em execução ao invés de serem um ambiente read-only;
2. A rapidez com a qual um produto de software pode ser disponibilizado;
3. A possibilidade de configurar diferentes ambientes de forma rápida;
4. A possibilidade de diminuir o número de incompatibilidades entre as aplicações disponíveis;

5. Os arquivos podem ser compartilhados entre o contêiner e o host, inclusive utilizando a nuvem para o processo;
6. Criar e mudar a infraestrutura é muito mais simples com o contêiner docker, pois as imagens do docker são construídas através de arquivos de definição
7. A possibilidade de transformar uma aplicação em imagem docker permite que ela seja alocada como container em ambientes diferentes

Na próxima parte do artigo vamos mostrar como instalar o Docker e como criar contêineres a partir de imagens existentes.