**Difference between Docker Image and Docker Container:**

| ***Docker Image*** | ***Docker Container*** |
| --- | --- |
| It is **Blueprint** of a Container. (**package**) | It is **instance** of the Image,  **Running environment** for image. |
| Image is created only once. | Containers are created any number of times using image. |
| Images are **immutable**. | Containers changes only if old image is deleted and new is used to build the container. |
| Images does not require computing resource to work. | Containers requires computing resources to run as they run as Docker Virtual Machine. |
| To make a docker image, you have to write script in Docker file. | To make container from image, you have to run “docker build .” command |
| Docker Images are used to package up applications and pre-configured server environments. | Containers use server information and file system provided by image in order to operate. |
| Images can be shared on Docker Hub.(public repository)(**portable**) | It makes no sense in sharing a running entity, always docker images are shared. |
| There is no such running state of Docker Image. | Containers uses RAM when created and in running state. |

|  |  |  |
| --- | --- | --- |
|  | ***Docker Container*** | ***Virtual Machine*** |
| **Isolation** | **OS-level** process **isolation** | **Hardware-level** process **isolation** |
| **OS** | Each container can **share OS** | Each VM has a **separate OS** |
| **Size** | Containers are **lightweight** (KBs/MBs) | VMs are of few GBs. (**Heavyweight**) |
| **Prebuild** | Pre-built docker containers are **easily** available | Ready-made VMs are **difficult** to find |
| **Portable** | Containers are destroyed and re-created rather than moving | VMs can move to new host easily |
| **Resource usage** | **Less** resource usage | **More** resource usage |
| **Boot-Time** | Boots in a **few seconds**. | It takes a **few minutes** for VMs to boot. |
| **Runs on** | Dockers make use of the **execution engine**. | VMs make use of the **hypervisor**. |
| **Memory Efficiency** | No space is needed to virtualize, hence **less memory.** | Requires entire OS to be loaded before starting the surface, so less efficient. |
| **Isolation** | Prone to adversities as no provisions for isolation systems. | Interference possibility is minimum because of the efficient isolation mechanism. |
| **Deployment** | Deploying is **easy** as only a single image, containerized can be used across all platforms. | Deployment is comparatively **lengthy** as separate instances are responsible for execution. |
| **Usage** | Docker has a complex usage mechanism consisting of both third party and docker managed tools. | Tools are easy to use and simpler to work with. |