Section 1: Cloud Foundry Overview

# Do you understand Cloud Foundry concepts like spaces, organizations, routes, services, domains, users, quotas?

## Spaces

Every application and service is scoped to a space. Each org contains at least one space. A space provides users with access to a shared location for application development, deployment, and maintenance. Each space role applies only to a particular space.

Reference:

<https://docs.pivotal.io/pivotalcf/1-9/concepts/roles.html#spaces>

## Organizations

An org is a development account that an individual or multiple collaborators can own and use. All collaborators access an org with user accounts. Collaborators in an org share a resource quota plan, applications, services availability, and custom domains.

By default, an org has the status of active. An admin can set the status of an org to suspended for various reasons such as failure to provide payment or misuse. When an org is suspended, users cannot perform certain activities within the org, such as push apps, modify spaces, or bind services.

Reference:

## <https://docs.pivotal.io/pivotalcf/1-9/concepts/roles.html#orgs>

## Routes

The Elastic Runtime Gorouter routes requests to applications by associating an app with an address, known as a route. We call this association a **mapping**. Use the cf CLI [cf map-route](http://cli.cloudfoundry.org/en-US/cf/map-route.html)command to associate an app and route.

The routing tier compares each request with a list of all the routes mapped to apps and attempts to find the best match. For example, the Gorouter would make the following matches for the two routes myapp.shared-domain.example.com and myapp.shared-domain.example.com/products:

|  |  |
| --- | --- |
| **Request** | **Matched Route** |
| http://myapp.shared-domain.example.com | myapp.shared-domain.example.com |
| http://myapp.shared-domain.example.com/contact | myapp.shared-domain.example.com |
| http://myapp.shared-domain.example.com/products | myapp.shared-domain.example.com/products |
| http://myapp.shared-domain.example.com/products/123 | myapp.shared-domain.example.com/products |
| http://products.shared-domain.example.com | No match; 404 |

The Gorouter does not use a route to match requests until the route is mapped to an app. In the above example, products.shared-domain.example.com may have been created as a route in Cloud Foundry, but until it is mapped to an app, requests for the route receive a 404.

Reference:

<https://docs.pivotal.io/pivotalcf/1-9/devguide/deploy-apps/routes-domains.html#routes>

## Services

Services are integrated with Cloud Foundry by implementing a documented API for which the cloud controller is the client; we call this the Service Broker API. This should not be confused with the cloud controller API, often used to refer to the version of Cloud Foundry itself; when one refers to “Cloud Foundry v2” they are referring to the version of the cloud controller API. The services API is versioned independently of the cloud controller API.

Reference:

<https://docs.pivotal.io/pivotalcf/1-9/services/overview.html>

## Domains

Domains indicate to a developer that requests for any route created from the domain will be routed to Elastic Runtime. This requires DNS to be configured out-of-band to resolve the domain name to the IP address of a load balancer configured to forward requests to the CF routers. For more information about configuring DNS, see the [DNS for Domains](https://docs.pivotal.io/pivotalcf/1-9/devguide/deploy-apps/routes-domains.html#domains-dns) section.

Reference:

<https://docs.pivotal.io/pivotalcf/1-9/devguide/deploy-apps/routes-domains.html#domains>

## Users

Using the cf Command Line Interface (CLI), an administrator can create users and manage user roles. Cloud Foundry uses role-based access control, with each role granting permissions in either an organization or an application space.

Reference:

<https://docs.pivotal.io/pivotalcf/1-6/adminguide/cli-user-management.html>

## Quotas

Quota plans are named sets of memory, service, and instance usage quotas. For example, one quota plan might allow up to 10 services, 10 routes, and 2 GB of RAM, while another might offer 100 services, 100 routes, and 10 GB of RAM. Quota plans have user-friendly names, but are referenced in Cloud Foundry (CF) internal systems by unique GUIDs.

Reference:

<https://docs.pivotal.io/pivotalcf/1-9/adminguide/quota-plans.html>

# How do you login to Cloud Foundry?

Set Up Your PWS Account

This step establishes your PWS account and access to the PWS Apps Manager for your account.

Sign up for a Pivotal Web Services account: <https://console.run.pivotal.io>

The Apps Manager displays after you create an organization (org). By default, PWS configures each new org with a space named development.

Reference#

<http://docs.run.pivotal.io/starting/index.html>

<https://login.run.pivotal.io/login>

# How do you deploy an application? What are three activities involved?

Three steps are -

1. Set Up Your PWS Account
2. Install the cf CLI
3. Deploy an Application to PWS

Reference:

<http://docs.run.pivotal.io/starting/index.html>

# Can you remember the steps Cloud Foundry goes through when deploying applications? What components are involved?

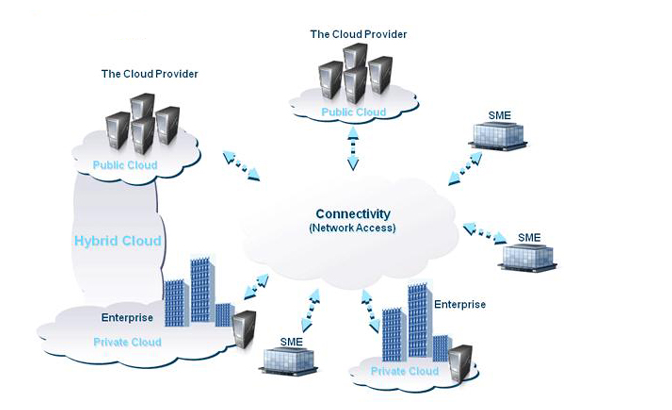
You deploy an app to Cloud Foundry by running a cf push command from the Cloud Foundry Command Line Interface (cf CLI). Refer to the Installing the cf CLI topic for more information. Between the time that you run cf push and the time that the app is available, Cloud Foundry performs the following tasks:

1. Uploads and stores app files
2. Examines and stores app metadata
3. Creates a “droplet” (the Cloud Foundry unit of execution) for the app
4. Selects an appropriate Diego cell to run the droplet
5. Starts the app

Reference:

<http://docs.run.pivotal.io/devguide/deploy-apps/deploy-app.html>

# What is the difference between a public, private and hybrid cloud?



Reference:

<http://thecloudtutorial.com/cloudtypes.html>

<http://www.dummies.com/programming/networking/comparing-public-private-and-hybrid-cloud-computing-options/>

# What infrastructures does Cloud Foundry run on?

Cloud Foundry (CF) has become the industry standard. It is an [open source](https://github.com/cloudfoundry) platform that you can deploy to run your apps on your own computing infrastructure, or deploy on an IaaS like AWS, vSphere, or OpenStack. You can also use a PaaS deployed by a commercial [CF cloud provider](https://www.cloudfoundry.org/learn/certified-providers/). A broad [community](https://www.cloudfoundry.org/community/) contributes to and supports Cloud Foundry. The platform’s openness and extensibility prevent its users from being locked into a single framework, set of app services, or cloud.

Reference#

<https://docs.cloudfoundry.org/concepts/overview.html>

# What is BOSH? Why is it useful?

BOSH creates and deploys virtual machines (VMs) on top of a physical computing infrastructure, and deploys and runs Cloud Foundry on top of this cloud. To configure the deployment, BOSH follows a manifest document.

Reference:

<http://bosh.io/>

<https://docs.cloudfoundry.org/concepts/overview.html>

# What is staging? What does it do?

Cloud Foundry has used two architectures for managing application containers: [Diego](https://docs.pivotal.io/pivotalcf/1-9/concepts/diego/diego-architecture.html) and Droplet Execution Agents (DEAs). For information about how DEA applications are staged, see the [Staging Apps with DEAs](https://docs.pivotal.io/pivotalcf/1-9/concepts/architecture/execution-agent.html#staging) section of the [Droplet Execution Agent](https://docs.pivotal.io/pivotalcf/1-9/concepts/architecture/execution-agent.html) topic.

Reference:

<https://docs.pivotal.io/pivotalcf/1-9/concepts/how-applications-are-staged.html>

# Do you know the difference between restarting, restaging and redeploying and application? How does each of these affect the services, environment-variables available to an application?

Reference#

<https://docs.pivotal.io/pivotalcf/1-9/devguide/deploy-apps/start-restart-restage.html>

<https://docs.pivotal.io/pivotalcf/1-7/devguide/deploy-apps/environment-variable.html>

# What is meant by ephemeral? What are the design implications for an application?

# What are the 12 Factor Design patterns? Could you list each one from memory?

I. Codebase

One codebase tracked in revision control, many deploys

II. Dependencies

Explicitly declare and isolate dependencies

III. Config

Store config in the environment

IV. Backing services

Treat backing services as attached resources

V. Build, release, run

Strictly separate build and run stages

VI. Processes

Execute the app as one or more stateless processes

VII. Port binding

Export services via port binding

VIII. Concurrency

Scale out via the process model

IX. Disposability

Maximize robustness with fast startup and graceful shutdown

X. Dev/prod parity

Keep development, staging, and production as similar as possible

XI. Logs

Treat logs as event streams

XII. Admin processes

Run admin/management tasks as one-off processes

Reference:

<https://12factor.net/>

# Why does Cloud Foundry rely on environment-variables?

# Can you manage environment-variables manually? If so how?

Reference:

# Can you name two predefined environment-variables available to any application?