Cloud Foundry - Services

People matter, results count.

Objectives of CF - Services

- Purpose:
  - To learn pivotal cloud foundry services.
- Product:
  - Managed Services
  - Pivotal CF Services
  - PWS and App Direct Services
  - Creating and Binding Services
- Process:
  - To learn and deploy a service.

Table of Contents

- Managed (Marketplace) Services
- Pivotal CF standard Services
- PWS and App Direct Services
- Creating and Binding Services
  - Provisioning Service
  - Using the CLI
  - Using the Pivotal CF App Manager Console
  - Binding to a Service
  - User Provided Services

MANAGED (MARKETPLACE) SERVICES

What is a Service?

- Service is an external application dependency or component such as
  - Database
  - Message Queue
  - Monitoring App
  - Security
  - Hadoop instance
  - Generic Service Endpoint (Web Services)
  - Other dependent applications

Features and Functionality

- Provide functionality to our applications
- External to our applications
  - Add-on provisioned alongside an application.
- May be shared among many applications
  - Example – Relational DB, Messaging system
- Are bound to (associated with) an application
  - Using a “Service Broker”
- Provide connection information to application via environment variables
  - **VCAP_SERVICES**

Why use a service?

- Applications are the deployment unit
  - Must be self-contained
- Anything else they need is provided by the PaaS
  - By a service
- Services in a PaaS are
  - One of the main possible charging units / elements
    - Instead of hardware resources like an IaaS
  - Make commercial PaaS possible
  - Enable charge-back in your organization

Two Types of Services

- Managed Services (a.k.a. “Marketplace” Services)
  - Available ‘out-of-the-box’
  - Selected from marketplace ‘catalog’
  - Instances provisioned by PaaS, for use by application

CF Managed

User Managed

- User Defined Services
  - Services running external to Cloud Foundry
  - Connection information stored and used to connect
  - PaaS does not provision resources, only supplies connection information.

Custom services

Also CF Managed

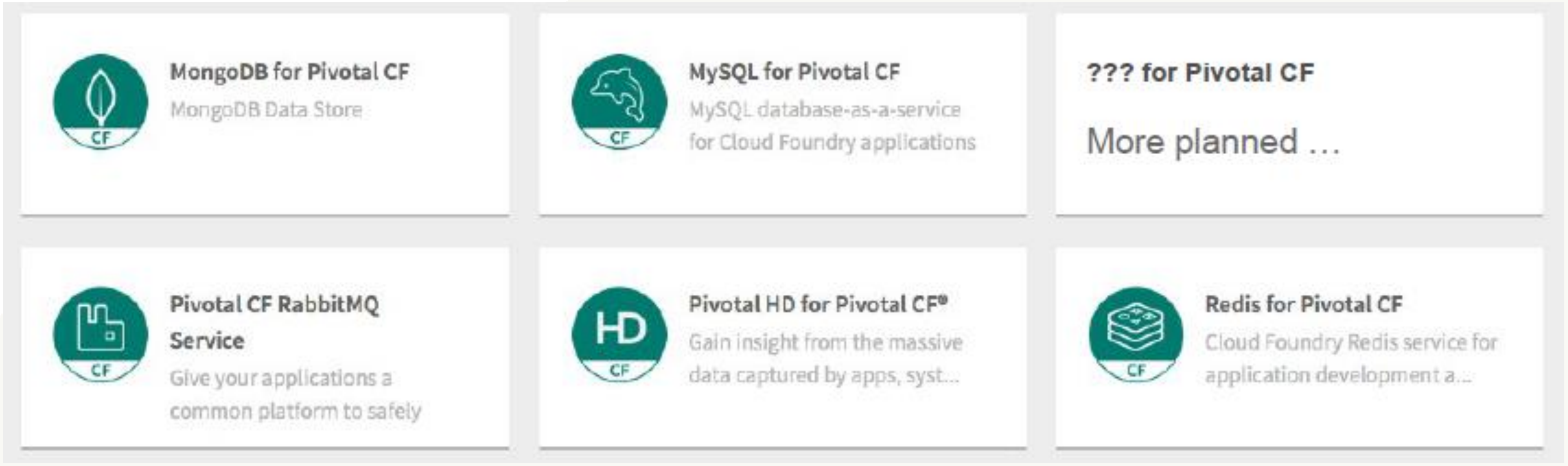
- Custom Built Services
  - Custom service created and installed into Cloud Foundry
    - Looks like any other managed service once added
    - Appears in the Marketplace
  - Alternative to user-defined services
    - Full integration with CF
- Requirements
  - Custom development of *service broker*
  - And *operator* installation into Cloud Foundry

Accessing External Services

- Typically these exist already
  - Your ERP and CRM systems (Oracle, SAP …)
  - Mainframe developed/running in-house
  - Cloud-based services such as sales, CRM or payroll
- Two options
  - User-defined service
    - Our ops people continue to manage and provision
  - Custom Service
    - CF uses service-broker to provision and bind

Accessing Managed Services

- Easily available via Marketplace
  - Allow us to sign-up, select plans, etc
  - Once bound to application, can be used easily
- Many pre-packaged services for Pivotal CF
  - See https://network.pivotal.io/products



MongoDB for Pivotal CF
MongoDB Data Store

MySQL for Pivotal CF
MySQL database-as-a-service for Cloud Foundry applications

??? for Pivotal CF
More planned …

Pivotal CF RabbitMQ Service
Give your applications a common platform to safely

Pivotal HD for Pivotal CF®
Gain insight from the massive data captured by apps, syst...

Redis for Pivotal CF
Cloud Foundry Redis service for application development a...

Managed Services

- Services preconfigured and made available to Cloud Foundry
  - Typically by operations personnel
- On-premise/private cloud
  - Our company controls what is available
  - Services typically run in our data-centre
- Public cloud
  - Cloud provided controlled
  - Services may run anywhere – locality considerations

PIVOTAL CF SERVICES

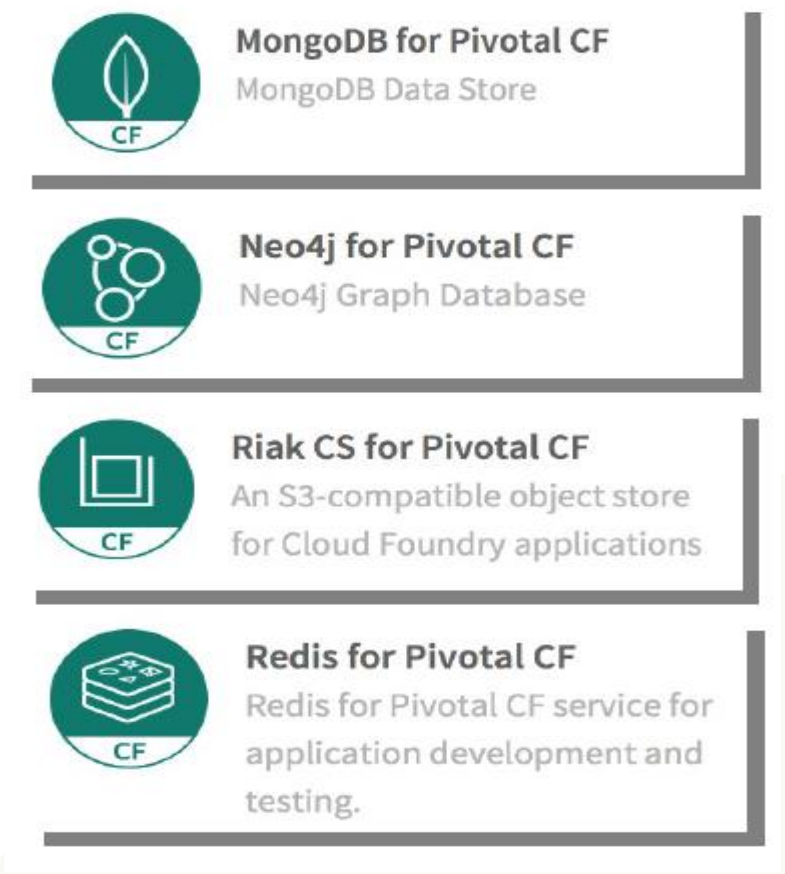
What Services are Available

- Whatever our company chooses
  - Services added after Pivotal CF installation by Ops
  - Available as .pivotal files from Pivotal Network
    - See https://network.pivotal.io
- The section discusses several services
  - They may not be available to your private cloud

SQL Databases - MYSQL

- Free, Open Source Relational Database
  - GPL licensing
  - High-Available, clustered, synchronously replicated using MariaDB Galera Cluster
    - Each mode has a copy of each DB
    - Writes to any DB are replicated to all copies
    - Client connections routed to primary, on failure proxy routes to a healthy node
  - Suitable for production use

NoSQL Data Services



MongoDB for Pivotal CF
MongoDB Data Store

Neo4j for Pivotal CF
Neo4j Graph Database

Riak CS for Pivotal CF
An S3-compatible object store for Cloud Foundry applications

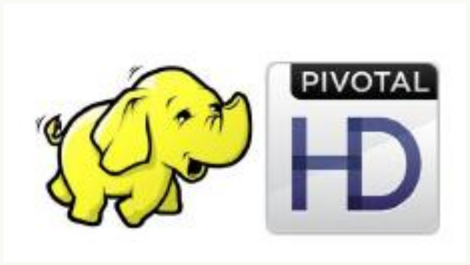
Redis for Pivotal CF
Redis for Pivotal CF service for application development and testing.

- MongoDB – Popular Document store
- Neo4j – Graph Database
- Riak CS – “Cloud Store” for accessing Amazon S3-like file storage
- Redis – Popular Key / Value store from Pivotal

- Available Soon: Elasticssearch, Memcached , Gemfire, DataStax (Cassandra)
- Not *intended* for production use

Pivotal HD for Cloud Foundry

- Directly from Pivotal CF, provision Hadoop resources to power data-centric Cloud Foundry Apps
  - HDFS, Map-Reduce, Hive, Hbase, YARN, Zookeeper ..
    - Or connect to existing Hadoop cluster
  - HAWQ: Pivotal's fast, distributed SQL engine
    - Perform deep, complex analytics in SQL (or R, Madlib, etc)
    - Run Hadoop jobs directly from Pivotal CF application
- Runtime service broker integration
  - Instant credential generation & binding to shared PHD cluster



RabbitMQ

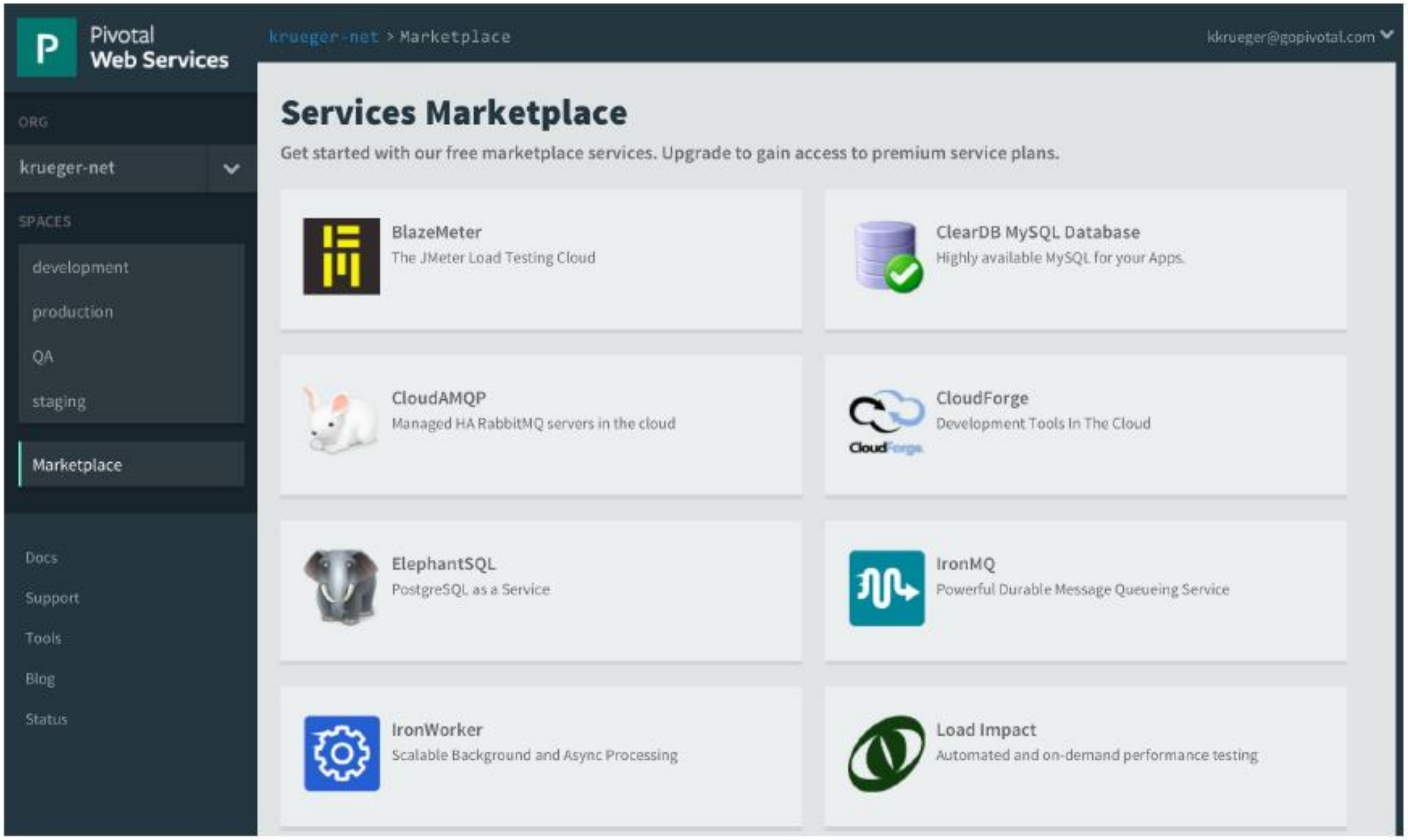
- Single HA cluster deployment of RabbitMQ
  - Runtime service broker integration
    - Instant credential generation, binding to shared RMQ cluster
    - Unique virtual host per binding
  - Is intended for production use

PWS AND APPS DIRECT

Pivotal Web Services (PWS)

- Public Cloud Foundry instance
  - Hosted on AWS
  - Provides extensive marketplace of services via App Direct
    - Some free
    - Some pay-per use
  - Examples
    - Postgres DB, MYSQL, MongoDB, Redis
    - Rabbit MQ
    - Blazemeter monitoring
    - Many, many more …

Marketplace Home Page in App Manager Console



Pivotal Web Services

krueger-net > Marketplace

kkrueger@gopivotal.com

ORG

krueger-net

SPACES

development

production

QA

staging

Marketplace

Docs

Support

Tools

Blog

Status

Services Marketplace

Get started with our free marketplace services. Upgrade to gain access to premium service plans.

BlazeMeter
The JMeter Load Testing Cloud

ClearDB MySQL Database
Highly available MySQL for your Apps.

CloudAMQP
Managed HA RabbitMQ servers in the cloud

CloudForge
Development Tools In The Cloud

ElephantSQL
PostgreSQL as a Service

IronMQ
Powerful Durable Message Queueing Service

IronWorker
Scalable Background and Async Processing

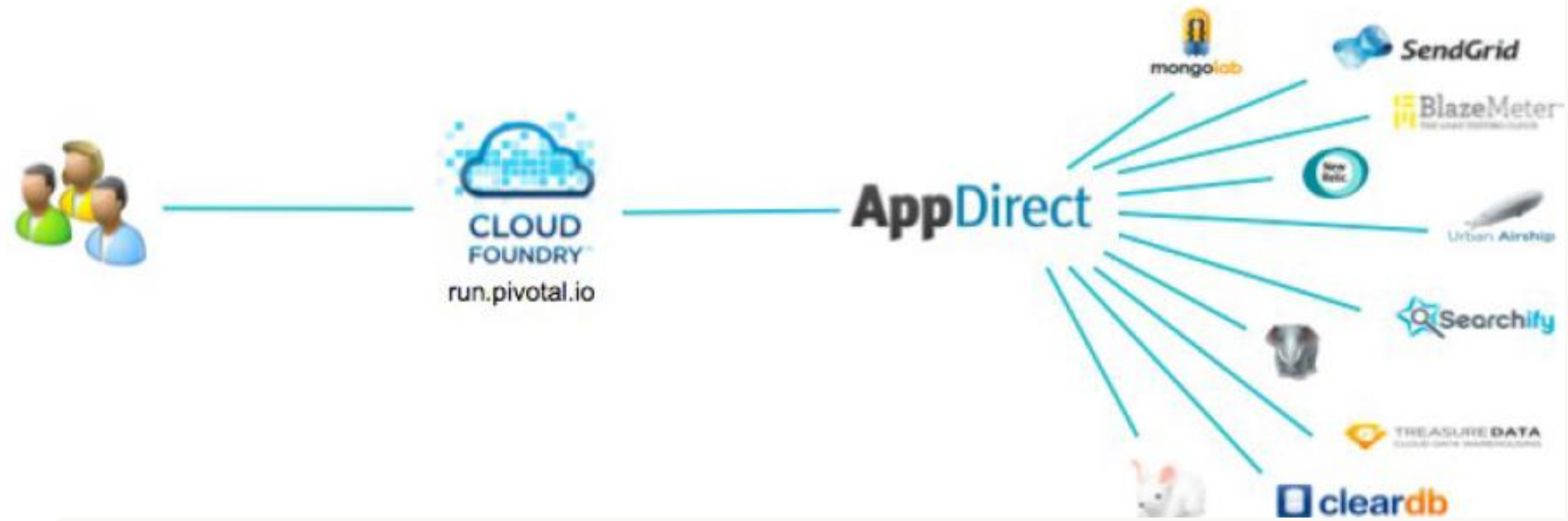
Load Impact
Automated and on-demand performance testing

App-Direct

- Commercial provider of services
  - Provide third-party service market
    - Teamed up with well-known providers, like Redis Labs
    - Various plans and fees
  - The provider of services offered by PWS
  - Our company may choose to use App-Direct as well
- Services run by providers
  - For example a Redis instance would run at Redis Labs
  - External to our data-centre
  - Locality issues: performance, connectivity, security, legal

PWS Cloud – Foundry Services

- Marketplace services in PWS offered via App-Direct



http://www.appdirect.com

Running Cloud Foundry On Premise

- All services typically run in our data-centre
  - Many may already exits
    - Databases, message-brokers, mail servers …
  - CF ops decide what services to install and/or make available to applications
- Our company may choose to use third-party services
  - For services it does not wish to manage
  - Cloud-based services such as sales, CRM, payroll
  - And/or a Commercial provider like AppDirect

Using a Public Service

- Services provided for us by our cloud provider
  - For example PWS
  - We have little control over what services are offered
  - Services may not be hosted by cloud provider
- Considerations around service location
  - Network reliability
  - Legal jurisdiction of host servers
  - Security

CREATING AND BINDING SERVICES

Provisioning Services → Service Vs. Service Instance

- Services provision services instances
  - For example
    - ClearDB service provisions MySQL databases.
    - Offers different plans (fees, SLAs)
  - We may get a dedicated server, or share a multi-tenant server

Provisioning – Operator View

- Available services depend on CF setup
  - Must be installed and configured by CF Ops
    - Either via Pivotal CF Operator's Console (*Ops Manager*)
    - Using **cf** CLI
    - Or using the BOSH provisioning tool
- Once Ops have deployed a service to your CF instance
  - It appears in the ***marketplace***
  - Can be made available to our application = **provisioning**

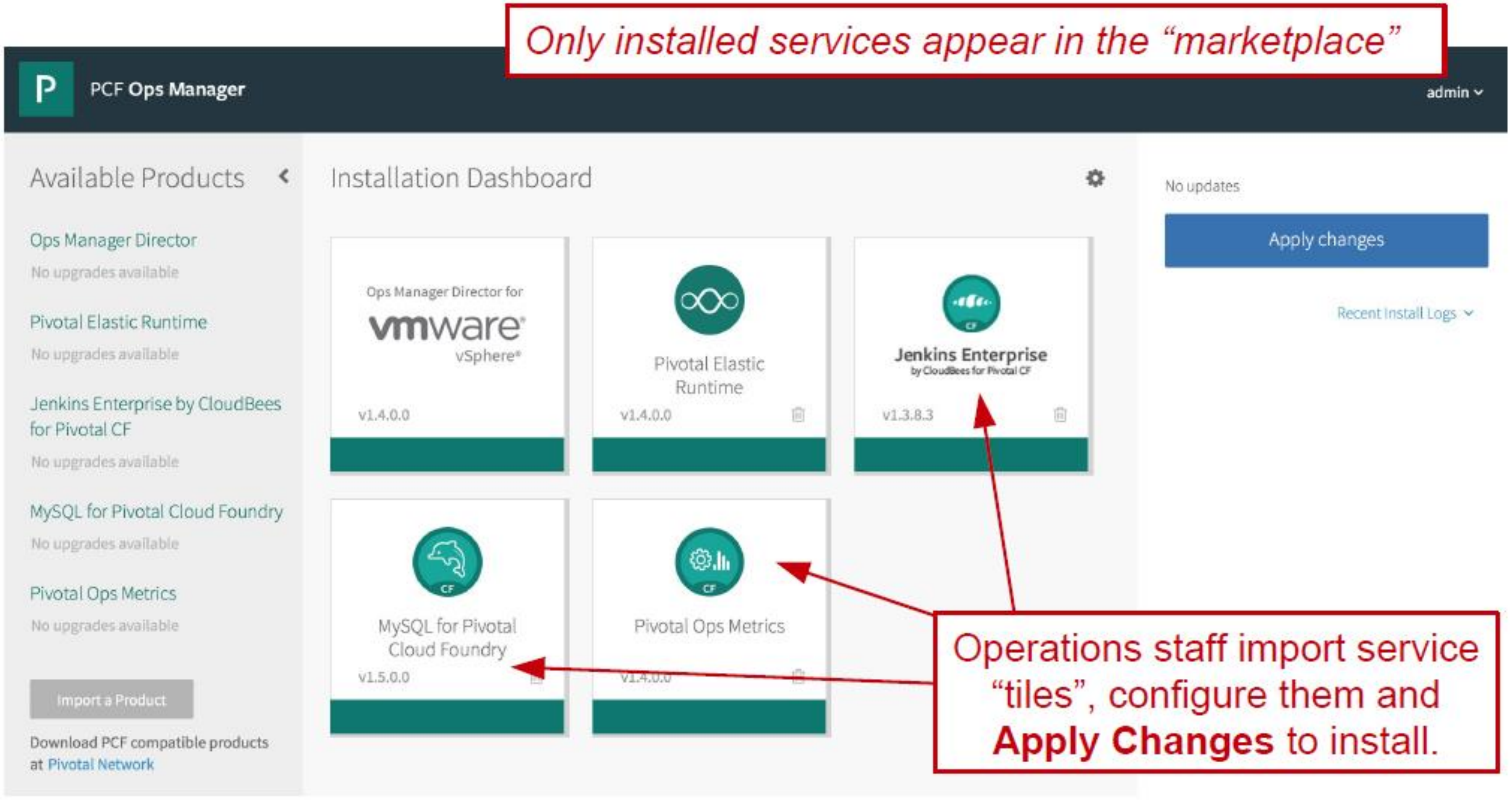
OPERATOR

cf create-service-broker

cf enable-service-access

Service "Tiles" in PCF Ops Manager

OPERATOR



Only installed services appear in the "marketplace"

PCF Ops Manager

admin

Available Products

Ops Manager Director
No upgrades available

Pivotal Elastic Runtime
No upgrades available

Jenkins Enterprise by CloudBees for Pivotal CF
No upgrades available

MySQL for Pivotal Cloud Foundry
No upgrades available

Pivotal Ops Metrics
No upgrades available

Import a Product

Download PCF compatible products at Pivotal Network

Installation Dashboard

Tile	Version
Ops Manager Director for vmware vSphere	v1.4.0.0
Pivotal Elastic Runtime	v1.4.0.0
Jenkins Enterprise by CloudBees for Pivotal CF	v1.3.8.3
MySQL for Pivotal Cloud Foundry	v1.5.0.0
Pivotal Ops Metrics	v1.4.0.0

No updates

Apply changes

Recent Install Logs

Operations staff import service "tiles", configure them and **Apply Changes** to install.

Provisioning – Developer View

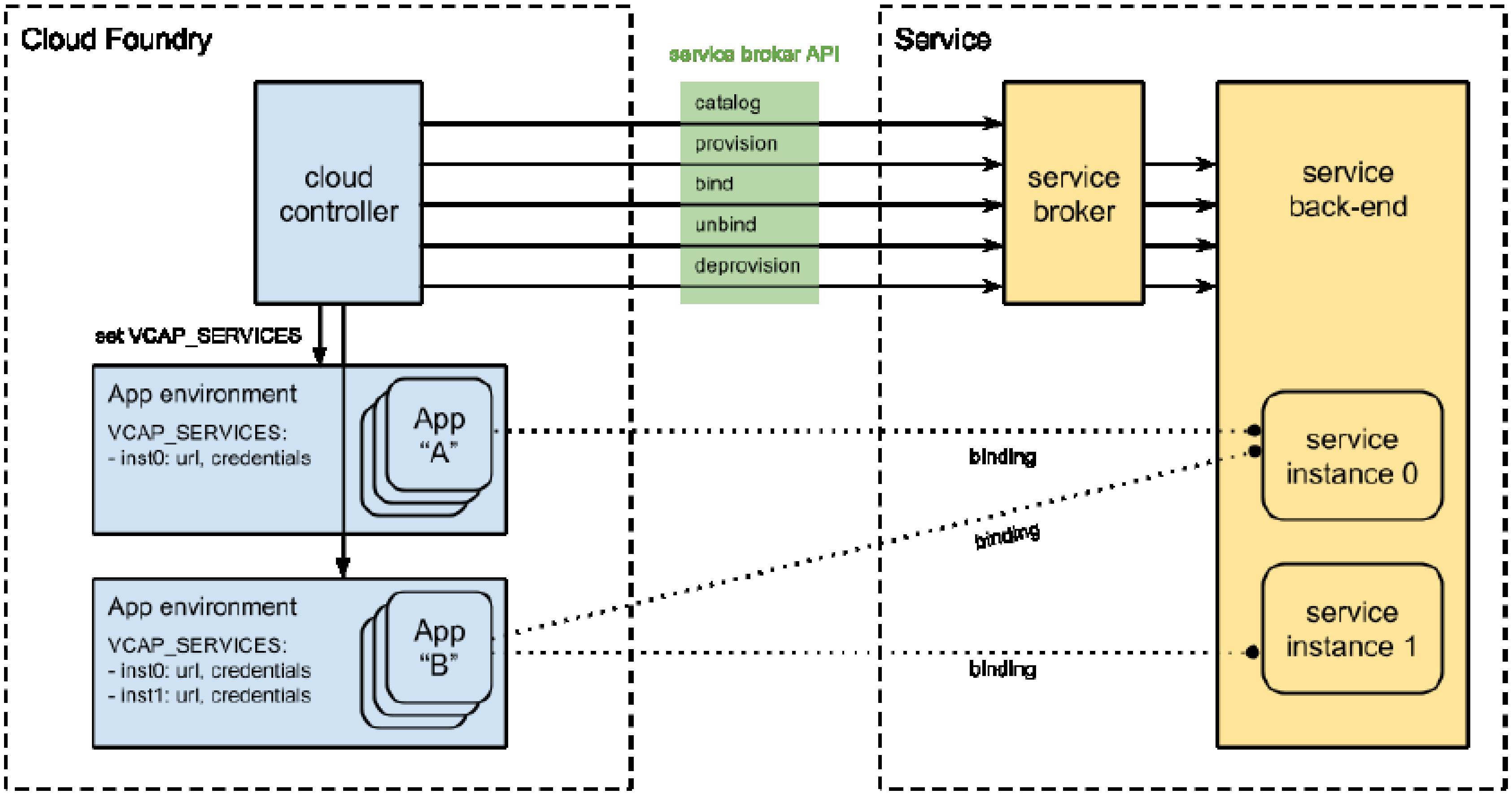
- Only concerned with what a developer has to do
  - Create (provision) a service
  - Bind it to your application

DEVELOPER

cf create <myService>

cf bind <myApp> <myService>

Services Overview – Service Brokers



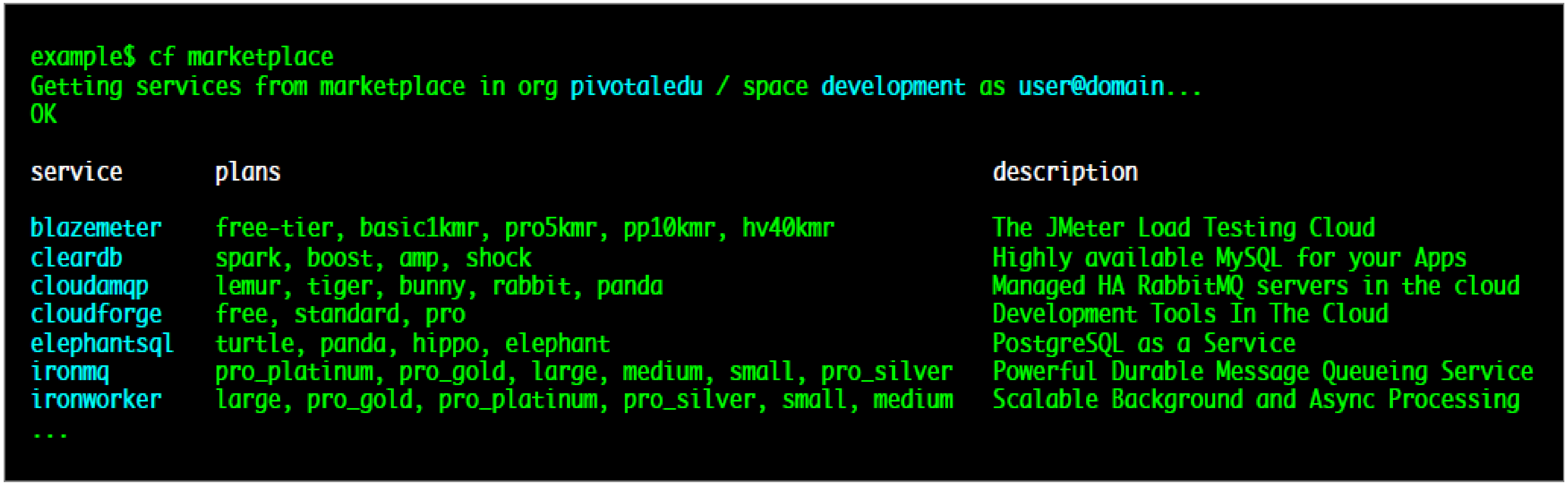
Creating a Service Instance

- Actually we are Provisioning an instance of a service
  - It must already exist in CF marketplace
- Use App Manager or **cf create-service**
  - Allows selection of service and plan
- Service instance becomes available to current space
  - And any applications running in it
  - For multiple spaces, run **create-service** in each space

Using the CLI

Finding Available Services (*Command Line Interface*)

- Check marketplace for available services
  - Essentially a service catalog

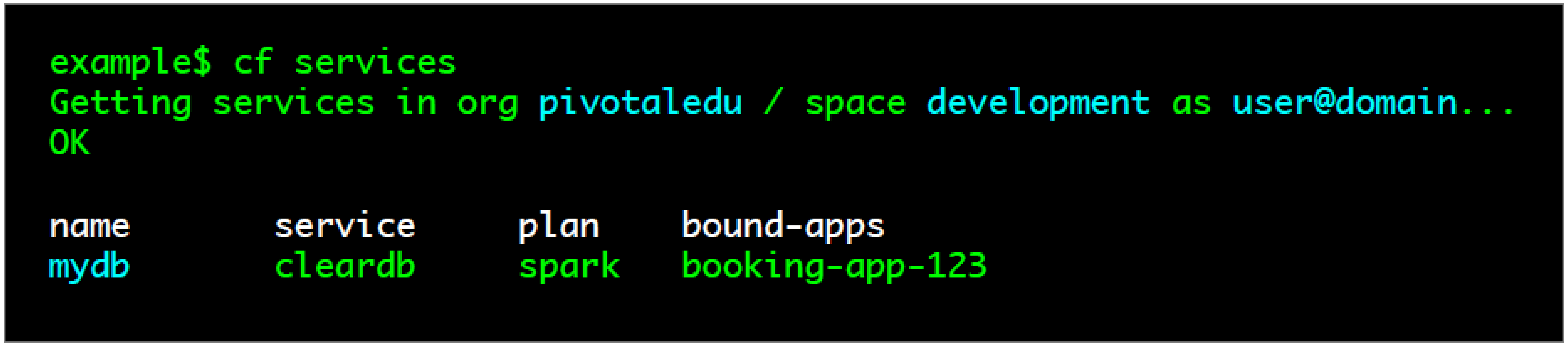


```
example$ cf marketplace
Getting services from marketplace in org pivotaledu / space development as user@domain...
OK

service        plans                                                     description

blazemeter     free-tier, basic1kmr, pro5kmr, pp10kmr, hv40kmr          The JMeter Load Testing Cloud
cleardb        spark, boost, amp, shock                                  Highly available MySQL for your Apps
cloudamqp      lemur, tiger, bunny, rabbit, panda                        Managed HA RabbitMQ servers in the cloud
cloudforge     free, standard, pro                                       Development Tools In The Cloud
elephantsql    turtle, panda, hippo, elephant                            PostgreSQL as a Service
ironmq         pro_platinum, pro_gold, large, medium, small, pro_silver  Powerful Durable Message Queueing Service
ironworker     large, pro_gold, pro_platinum, pro_silver, small, medium  Scalable Background and Async Processing
...
```

Finding Existing Service Instances (*Command Line Interface*)

- List existing services instance
  - In current space

- In this example: one service instance called mysql

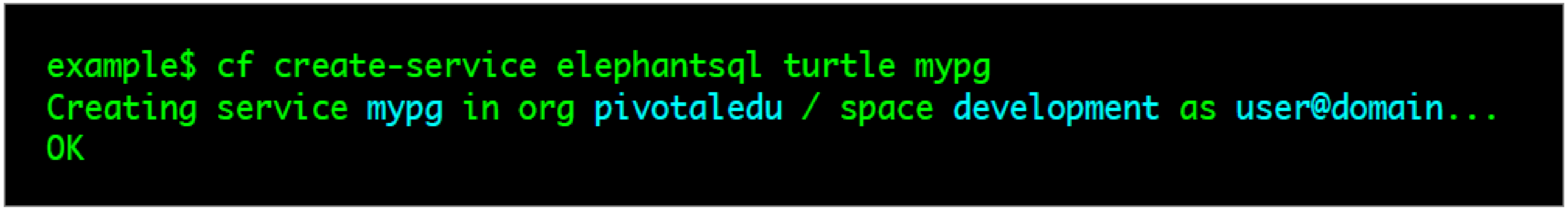


```
example$ cf services
Getting services in org pivotaledu / space development as user@domain...
OK

name          service      plan     bound-apps
mydb          cleardb      spark    booking-app-123
```

- Remember, to change spaces
  - **cf target –s [space-name]**

Provisioning a new Service Instance (*Command Line Interface*)

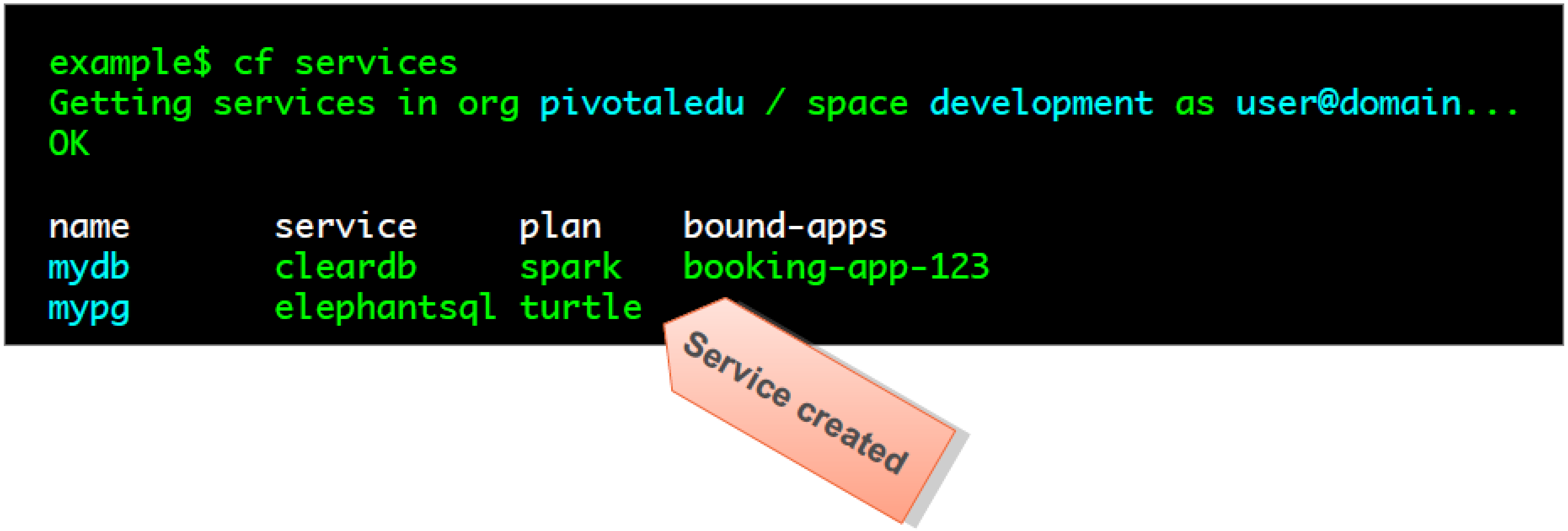
- Provision a new service instance
  - Added to current space
  - Give it a name
  - Choose the correct plan or contract
- Usage
  - **cf create-service [service-name] [plan-name] [instance-name]**



```
example$ cf create-service elephantsql turtle mypg
Creating service mypg in org pivotaledu / space development as user@domain...
OK
```

Finding Existing Service Instances
(*Command Line Interfaces*)

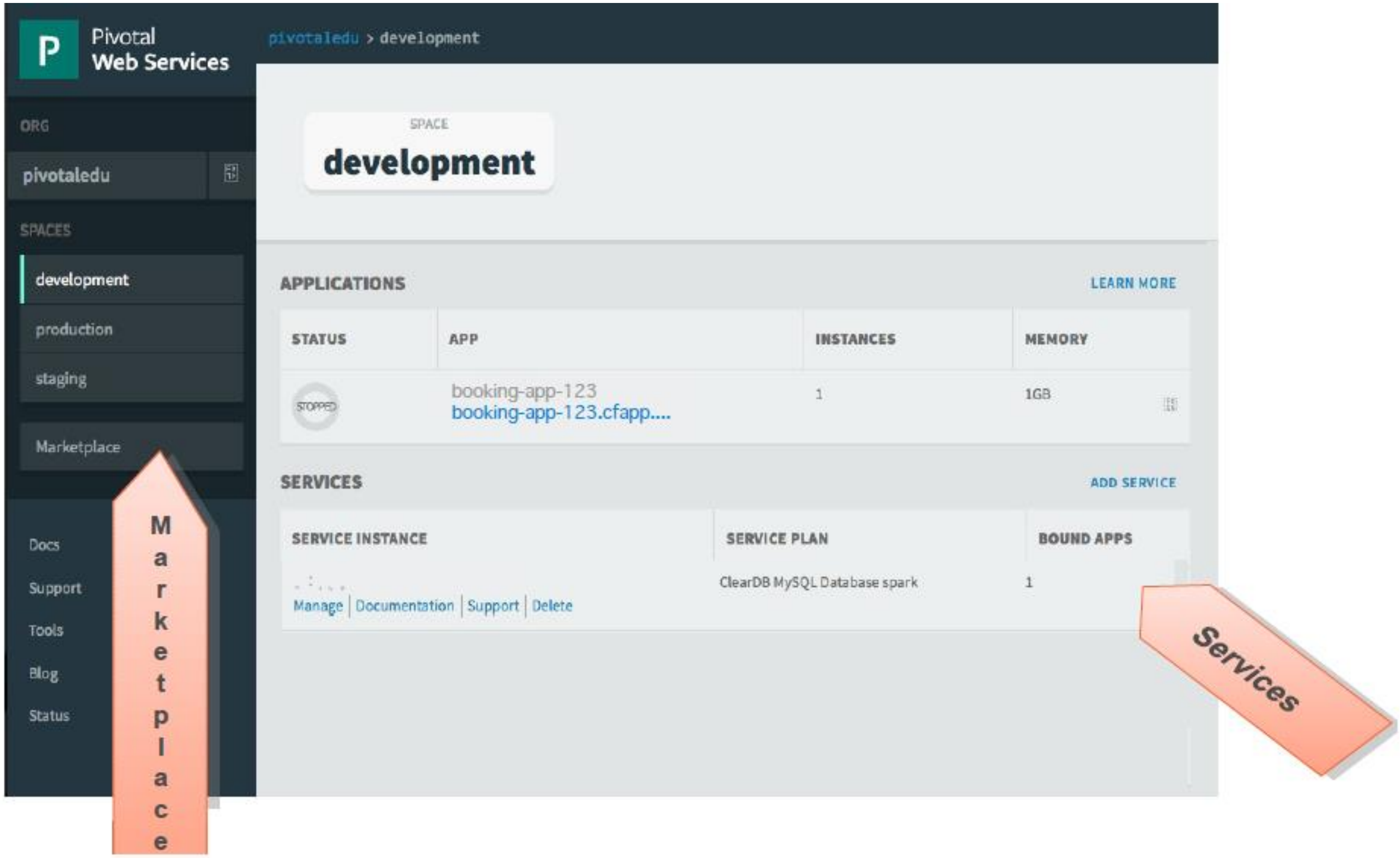
- List service instances again for current space
  - New service instance now appears



```
example$ cf services
Getting services in org pivotaledu / space development as user@domain...
OK

name          service      plan    bound-apps
mydb          cleardb      spark   booking-app-123
mypg          elephantsql turtle
```

Using the Pivotal CF App Manager Console

Provisioning Service Instances
GUI



SPACE
development

APPLICATIONS

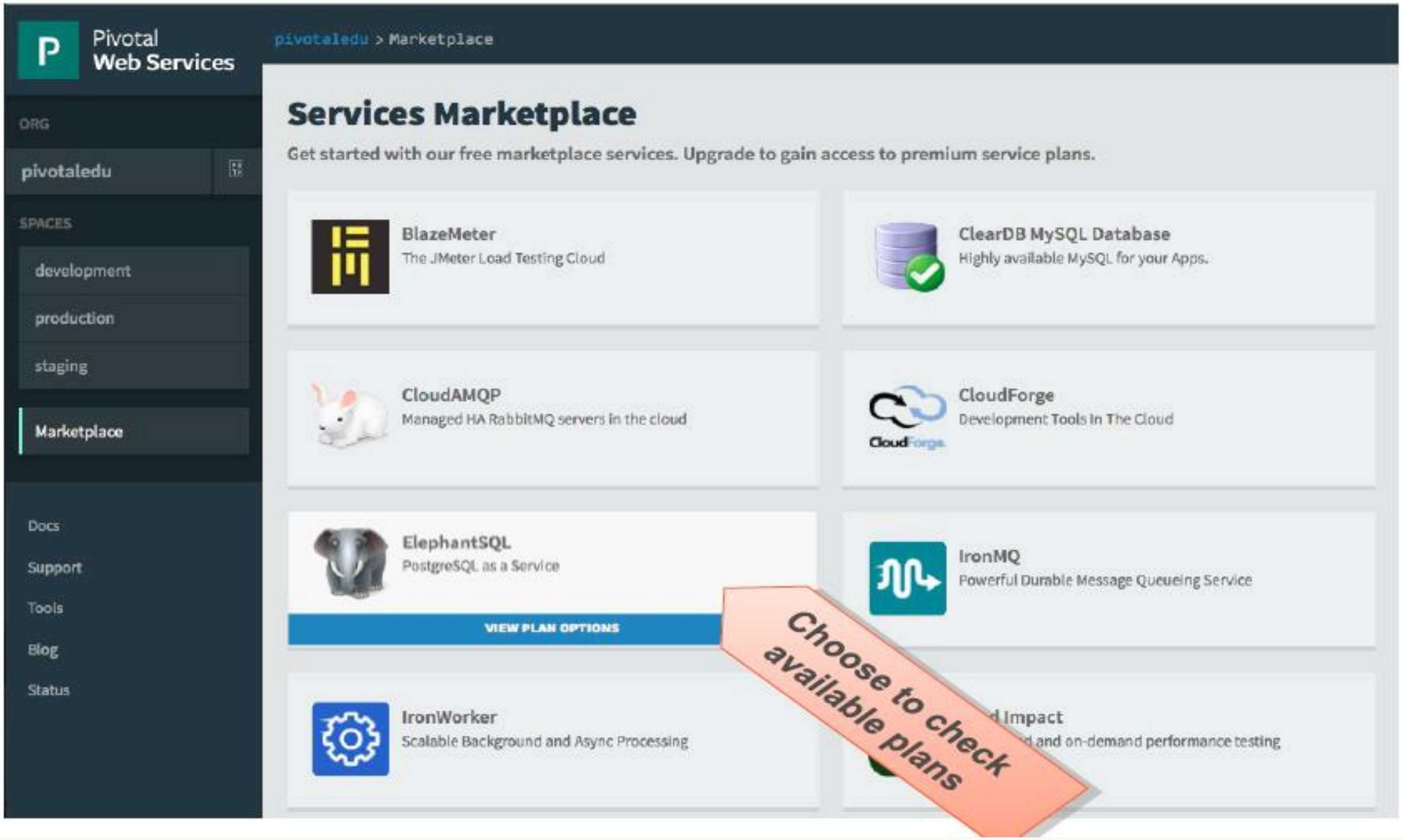
STATUS	APP	INSTANCES	MEMORY
STOPPED	booking-app-123 booking-app-123.cfapp....	1	1GB

SERVICES

SERVICE INSTANCE	SERVICE PLAN	BOUND APPS
	ClearDB MySQL Database spark	1

Learning & Culture
All work described was performed by Capgemini or a Capgemini affiliate

Finding Available Services – (*Service Selection*)



Services Marketplace

Get started with our free marketplace services. Upgrade to gain access to premium service plans.

BlazeMeter
The JMeter Load Testing Cloud

ClearDB MySQL Database
Highly available MySQL for your Apps.

CloudAMQP
Managed HA RabbitMQ servers in the cloud

CloudForge
Development Tools In The Cloud

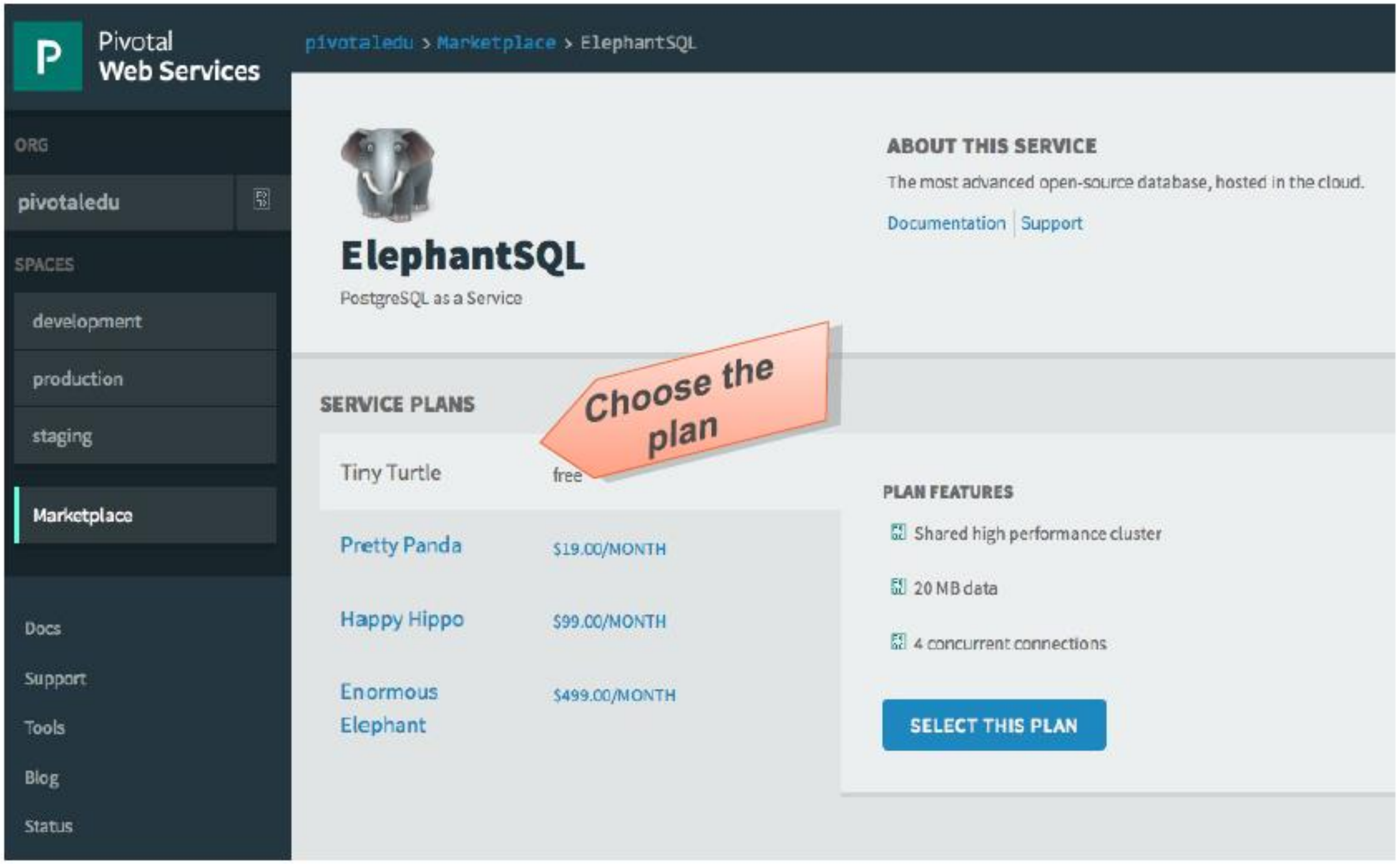
ElephantSQL
PostgreSQL as a Service

IronMQ
Powerful Durable Message Queueing Service

IronWorker
Scalable Background and Async Processing

d Impact
d and on-demand performance testing

Provisioning a new Service Instance – (*Pick a Plan*)



pivotaledu > Marketplace > ElephantSQL

ElephantSQL

PostgreSQL as a Service

ABOUT THIS SERVICE

The most advanced open-source database, hosted in the cloud.

Documentation | Support

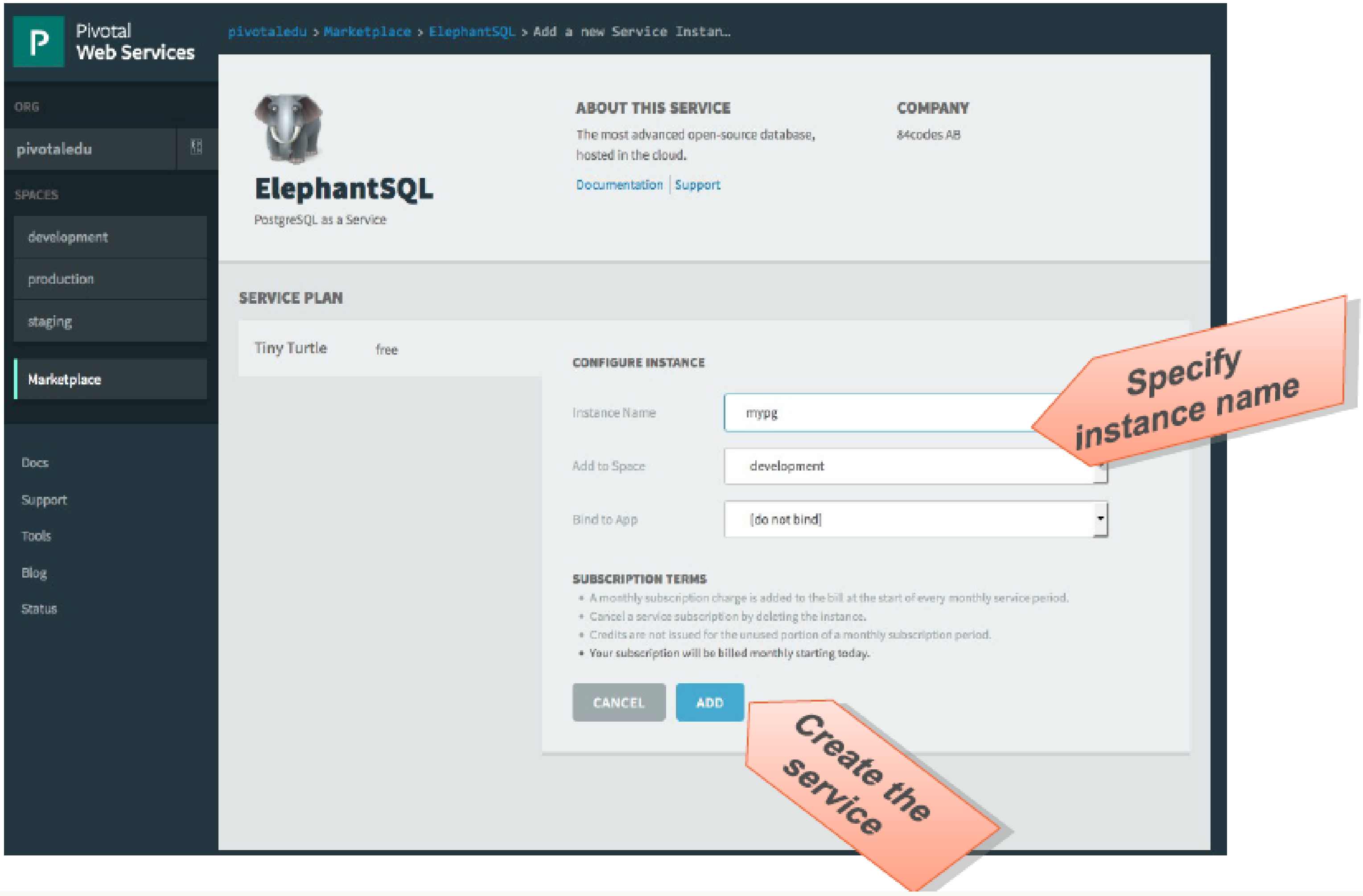
SERVICE PLANS

Tiny Turtle	free
Pretty Panda	$19.00/MONTH
Happy Hippo	$99.00/MONTH
Enormous Elephant	$499.00/MONTH

PLAN FEATURES

- Shared high performance cluster
- 20 MB data
- 4 concurrent connections

Provisioning a new Service Instance – (*Provision (create) service*)



pivotaledu > Marketplace > ElephantSQL > Add a new Service Instan...

ElephantSQL
PostgreSQL as a Service

ABOUT THIS SERVICE
The most advanced open-source database, hosted in the cloud.
Documentation | Support

COMPANY
84codes AB

SERVICE PLAN
Tiny Turtle free

CONFIGURE INSTANCE

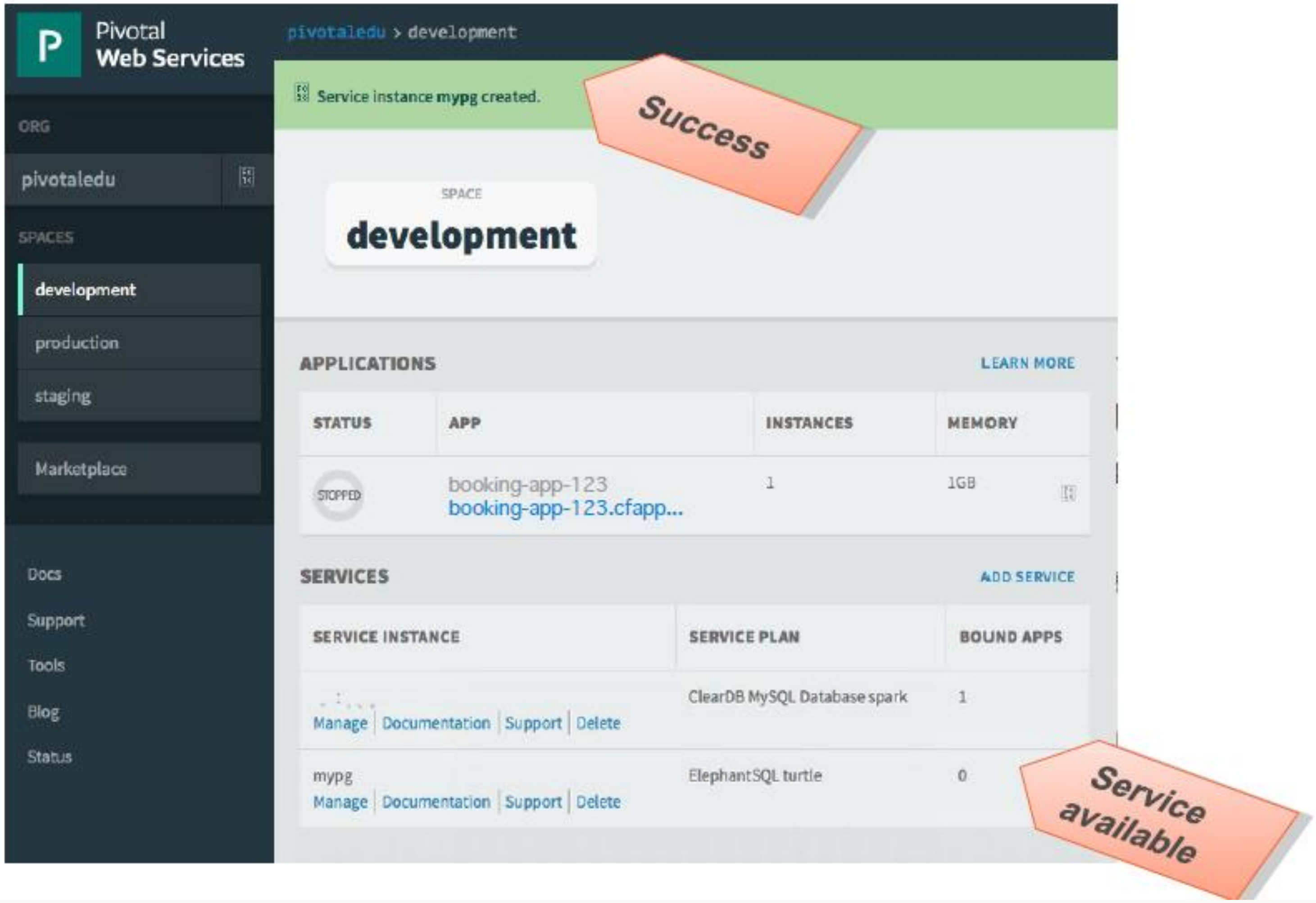
Instance Name	mypg
Add to Space	development
Bind to App	[do not bind]

SUBSCRIPTION TERMS
- A monthly subscription charge is added to the bill at the start of every monthly service period.
- Cancel a service subscription by deleting the instance.
- Credits are not issued for the unused portion of a monthly subscription period.
- Your subscription will be billed monthly starting today.

CANCEL ADD

Learning & Culture
All work described was performed by Capgemini or a Capgemini affiliate

Provisioning a new Service Instance – (*Complete*)



Service instance mypg created.

APPLICATIONS

STATUS	APP	INSTANCES	MEMORY
STOPPED	booking-app-123 booking-app-123.cfapp...	1	1GB

SERVICES

SERVICE INSTANCE	SERVICE PLAN	BOUND APPS
Manage | Documentation | Support | Delete	ClearDB MySQL Database spark	1
mypg Manage | Documentation | Support | Delete	ElephantSQL turtle	0

Learning & Culture

All work described was performed by Capgemini or a Capgemini affiliate

Binding to a Service

Accessing Service Instances from an App? (*Traditional way*)

- Traditionally, for an application to access a service instance, connection properties are required
- For example: a database instance
  - Need to know service address / port, credentials
    - Such as a JDBC connection
- May be hard-coded, provided through the environment or a configuration file
- Typically service-specific code is required

Accessing Service Instances from an App? (*Traditional way*)

- Configuration files

Ruby

```
development:
  adapter: mysql2
  encoding: utf8
  database: pivotaldb
  username: pivotal
  password: pivotal
  host: myDbHost
  port: 3306
```

Groovy

```
datasource {
    driverClassName = "com.mysql.jdbc.Driver"
    username = "pivotal"
    password = "pivotal"
    url = "jdbc:mysql://myDbHost:3306/pivotaldb"
}
```

Java

```
datasource.driverClassName="com.mysql.jdbc.Driver"
datasource.username="pivotal"
datasource.password="pivotal"
datasource.url="jdbc:mysql://myDbHost:3306/pivotaldb"
```

Accessing Service Instances from an App?
(*The CloudFoundry way*)

- In CloudFoundry, you bind the service instance to apps
  - Connection credentials are negotiated / defined for you
  - Application code only needs service name and type/kind
    - Example: a Postgres instance with name "**mypg**"
  - Service details injected into application by CF
    - VCAP_SERVICES
  - Any changes (host/port/credentials) are managed external to the application.

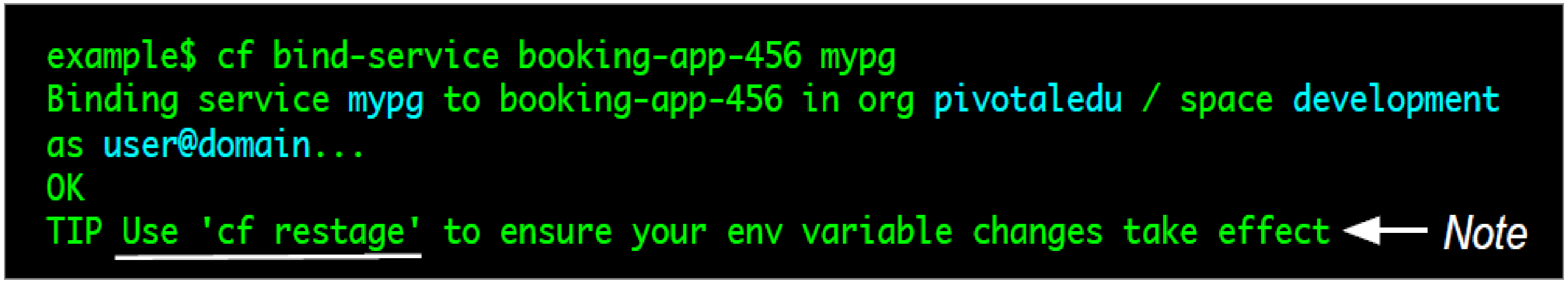
Example VCAP_SERVICES Property

```
VCAP_SERVICES=
{
  cleardb-n/a: [
    {
      name: "cleardb-1",
      label: "cleardb-n/a",
      plan: "spark",
      credentials: {
        name: "ad_c6f4446532610ab",
        hostname: "us-cdbr-east-03.cleardb.com",
        port: "3306",
        username: "b5d435f40dd2b2",
        password: "ebfc00ac",
        uri: "mysql://b5d435f40dd2b2:ebfc00ac@us-cdbr-east-
                  03.cleardb.com:3306/ad_c6f4446532610ab",
        jdbcUrl: "jdbc:mysql://b5d435f40dd2b2:ebfc00ac@us-
                  cdbr-east-03.cleardb.com:3306/ad_c6f4446532610ab"
      }
  ...
```

ClearDB is the MySQL instance offered through App Direct

Using a Service – Cloud Foundry
(*Binding using the CLI*)

- Binding associates an application to a service instance.
  - Use **cf bind-service**
  - Syntax
    - **cf bind-service [app_name] [service_name]**



```
example$ cf bind-service booking-app-456 mypg
Binding service mypg to booking-app-456 in org pivotaledu / space development
as user@domain...
OK
TIP Use 'cf restage' to ensure your env variable changes take effect
```

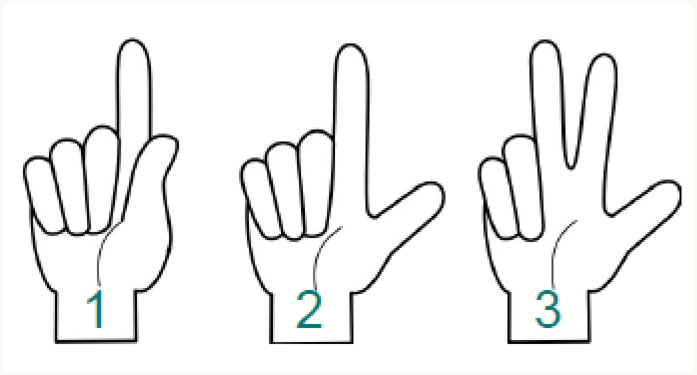
Using a Service – Cloud Foundry
(*Binding using a Manifest*)

- Add a services section to our application in the manifest
  - Example manifest.yml

```
---
applications:
- name: booking-app-456
  memory: 256M
  instances: 2
  host: booking-app-456
  domain: cfapps.io
  path: target/booking-app.war
  # services, one per line
  services:
  - mypg
  - mydb
```

Using a Service – (*The CloudFoundry way*)

- Cloud Foundry provides application with VCAP_SERVICES environment variable
  - Which contains connection details / credentials in JSON.
- How can an application obtain the credentials?
- Three options:
  - Manual
    - Explicit low-level code
  - Custom library
    - Explicit code, higher level interface
  - Auto Configuration
    - CF does it for you



Using a Service – (*Application View*)
1. Manually

- Manual configuration
  - Access VCAP_SERVICES environment variable
  - In our code, parse the JSON (see next slide)
- Very low-level but works in most languages
  - Fall-back option when options 2 and 3 aren't possible

Recall: VCAP_SERVICES Property

```
VCAP_SERVICES=
{
  cleardb-n/a: [
    {
      name: "cleardb-1",
      label: "cleardb-n/a",
      plan: "spark",
      credentials: {
        name: "ad_c6f4446532610ab",
        hostname: "us-cdbr-east-03.cleardb.com",
        port: "3306",
        username: "b5d435f40dd2b2",
        password: "ebfc00ac",
        uri: "mysql://b5d435f40dd2b2:ebfc00ac@us-cdbr-east-
                            03.cleardb.com:3306/ad_c6f4446532610ab",
        jdbcUrl: "jdbc:mysql://b5d435f40dd2b2:ebfc00ac@us-
                   cdbr-east-03.cleardb.com:3306/ad_c6f4446532610ab"
      }
  ...
```

Just a very long string in JSON format

Parse to extract these credentials

Using a Service – (*Application View*)
2. Custom Library

- Avoid manual parsing using a cloud-aware library
  - Cloud foundry aware helper code
    - Language/framework dependent
    - Parses VCAP_SERVICES for us
  - JVM: use Spring Cloud project
  - Node.js: use *cfruntime* object

Derived from VCAP_SERVICES

```
for (ServiceInfo service : cloud.getServiceInfos() ) {
    if (service instanceof MysqlServiceInfo)
        connectionURI = ((MysqlServiceInfo)service).getJdbcUri();
    } ...
```

Java Example

Using a Service – (*Application View*) 3. Auto Configuration

- Cloud Foundry creates the service connection for us
  - Not always supported, depends on:

1. The buildpack
   – Some builpacks support auto-configuration, others do not.
2. The framework
   – Spring, grails, Lift, Rails currently supported.
3. The uniqueness of the service type
   – For example, can auto-configure ONE database connection

CF doesn't know which is which if there are two or more

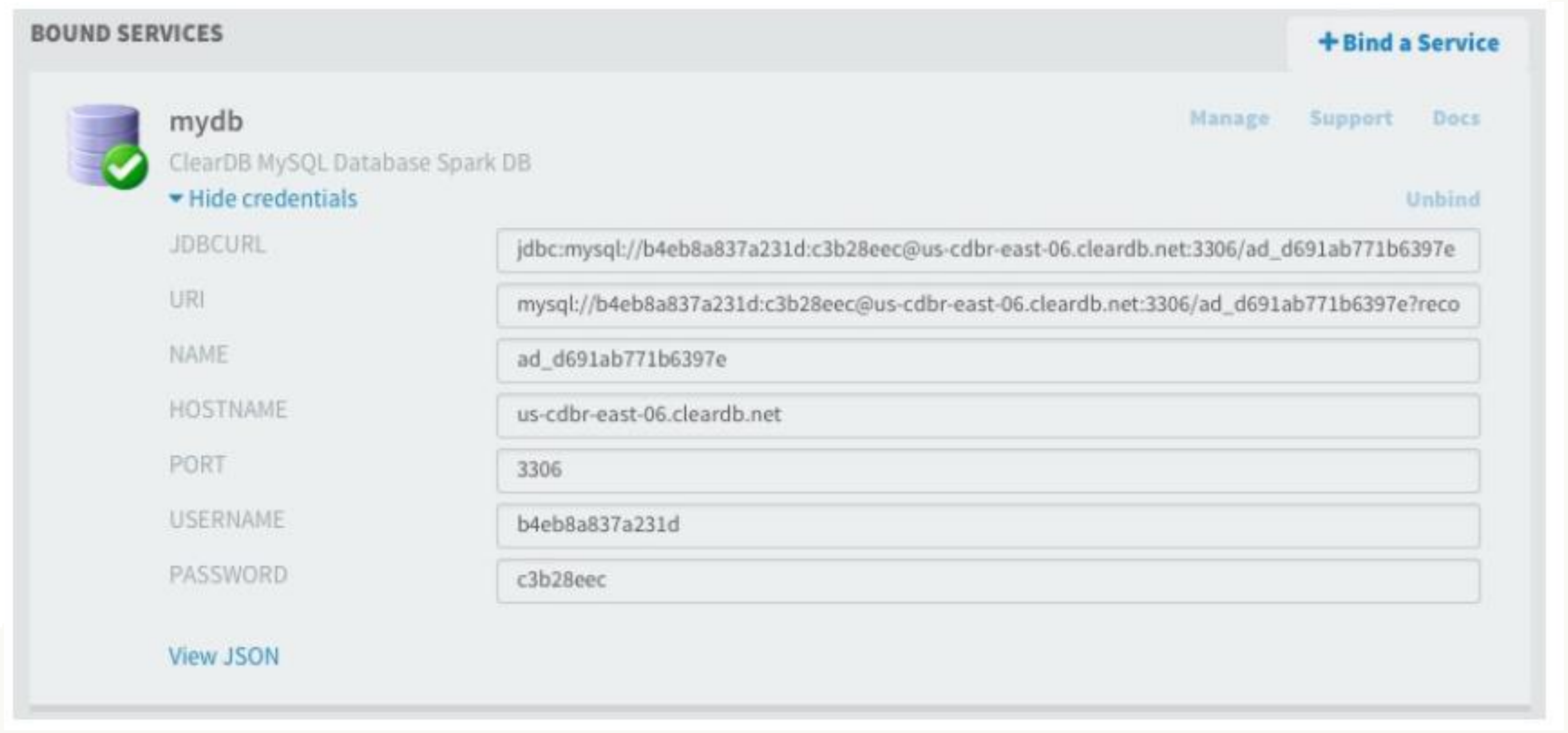
Accessing Connection Information

- Recall
  - Connection information once bound is in **VCAP_SERVICES**
  - Every application’s environment is logged at startup

- Once application is staged, view connection information using
  - cf env [app-name]
  - Look for **VCAP_SERVICES** in the output

Accessing Connection Information -2

- Connection information also available via App manager:


BOUND SERVICES

mydb

ClearDB MySQL Database Spark DB

JDBCURL	jdbc:mysql://b4eb8a837a231d:c3b28eec@us-cdbr-east-06.cleardb.net:3306/ad_d691ab771b6397e
URI	mysql://b4eb8a837a231d:c3b28eec@us-cdbr-east-06.cleardb.net:3306/ad_d691ab771b6397e?reco
NAME	ad_d691ab771b6397e
HOSTNAME	us-cdbr-east-06.cleardb.net
PORT	3306
USERNAME	b4eb8a837a231d
PASSWORD	c3b28eec

User Provided Services

User Provided Service Instances

- User-provided service instances are service instances
  - Already provisioned outside of **C**loud **F**oundry
  - Behave like other service instances once created
  - Are little more than predefined configurations
    - A “mock” service for providing credentials
- When bound they provide service instance configuration (including credentials) to applications
  - Avoids hard coding service instance endpoints

http://docs.cloudfoundry.org/devguide/services/user-provided.html

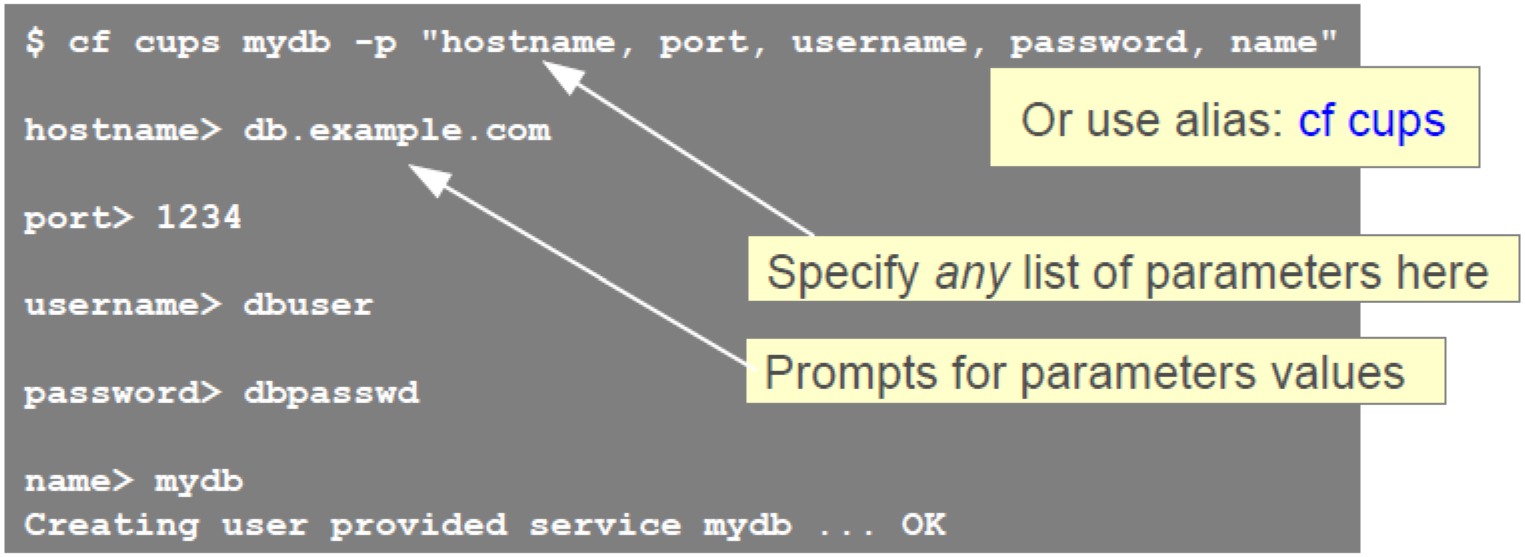
Use Cases
User Provided Service Instances

- These are typically legacy or existing instances of a service (databases, queues, email, etc)
  - Applications connect to the same instance
    - With CF services, applications get different instances
  - Typically used with CF on-premise
    - Easy integration of your CF PaaS with our existing systems

- Credentials passing used to inject the same credential set into each applicaiton

Defining User Provided Services - 1

- Use **cf create-user-provided-service** command
  - Provide name and parameters/credentials
  - All applications bound to same instance in same way



```
$ cf cups mydb -p "hostname, port, username, password, name"

hostname> db.example.com

port> 1234

username> dbuser

password> dbpasswd

name> mydb
Creating user provided service mydb ... OK
```

Or use alias: cf cups
Specify *any* list of parameters here
Prompts for parameters values

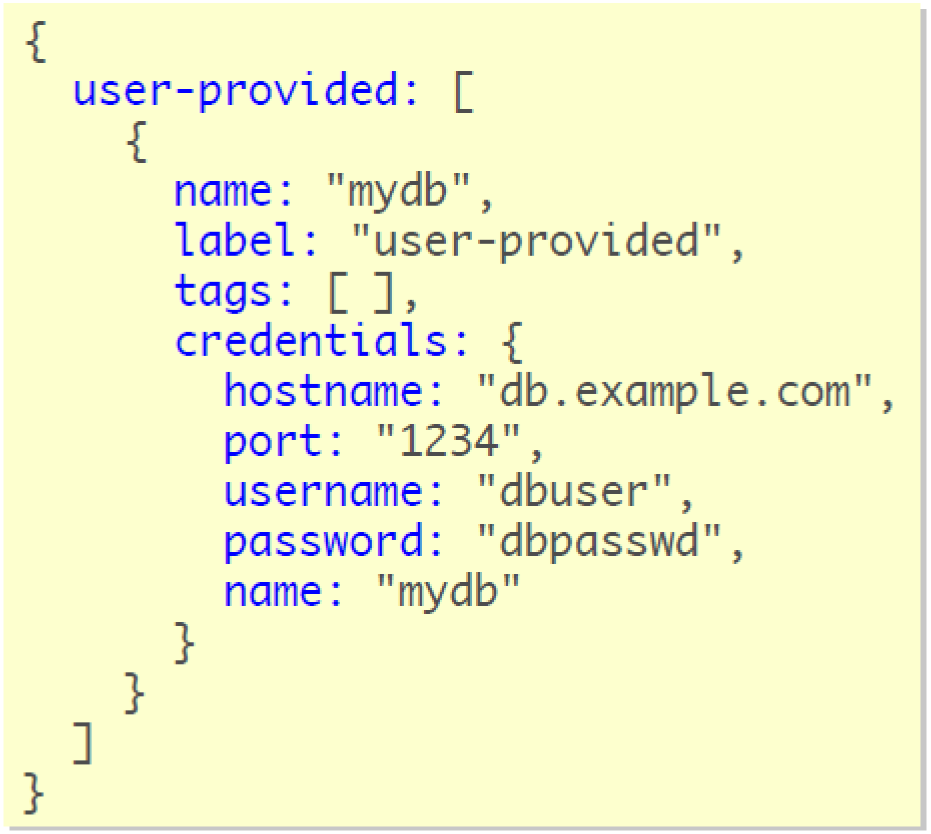
Defining User Provided Services - 2

- Or define within application's manifest.yml

```
---
applications:
- name: spring-music
  memory: 512M
  instances: 1
  host: spring-music
  domain: cfapps.io
  path: build/libs/spring-music.war
  services:
    mydb:
      label: user-provided
      credentials:
        uri: postgres://dbuser:dbpass@db.example.com:1234/dbname
        username: pivotal
        password: pivotal
```

User Provided Services - Accessing

- Bound service properties available in **VCAP_SERVICES environment variable**
- In our code
  - Access variable
  - Parse JSON
  - Use to connect



```
{
  user-provided: [
    {
      name: "mydb",
      label: "user-provided",
      tags: [ ],
      credentials: {
        hostname: "db.example.com",
        port: "1234",
        username: "dbuser",
        password: "dbpasswd",
        name: "mydb"
      }
    }
  ]
}
```

Example: Application with Multiple Services

```
VCAP_SERVICES: {
  "rediscloud": [
   {
    "credentials": {
      "hostname": "redisvr...com",
      "password": "wU974wucDT45Jc",
      "port": "19016"
    },
    "label": "rediscloud",
    "name": "session-replication",
    "plan": "25mb",
    "tags": [
     "Data Stores",
     "Cloud Databases",
     "Developer Tools",
     "Data Store",
     "key-value",
     "redis"
    ]
   }
  ],
```

```
"user-provided": [
   {
    "credentials": {
     "uri": "http://review.cfapps.io"
    },
    "label": "user-provided",
    "name": "reviews",
    "syslog_drain_url": "",
    "tags": []
   },
   {
    "credentials": {
     "uri": "http://products.cfapps.io"
    },
    "label": "user-provided",
    "name": "products",
    "syslog_drain_url": "",
    "tags": []
   }
  ]
 }
```

Recap

Services

CF Service

accessing Service

VCAP_SERVICE

access

User Defined Service

People matter, results count.

About Capgemini

With more than 130,000 people in 44 countries, Capgemini is one of the world's foremost providers of consulting, technology and outsourcing services. The Group reported 2012 global revenues of EUR 10.3 billion.
Together with its clients, Capgemini creates and delivers business and technology solutions that fit their needs and drive the results they want. A deeply multicultural organization, Capgemini has developed its own way of working, the Collaborative Business Experience^{TM}, and draws on Rightshore$^{®}$, its worldwide delivery model.

www.capgemini.com