

# Lab 6 - Binding to Cloud Foundry Services

---

*The Spring Music application was designed to illustrate the ease with which various types of data services can be bound to and utilized by Spring applications running on Cloud Foundry. In this lab, we'll be binding the application to a MySQL database.*

*Cloud Foundry services are managed through two primary types of operations:*

## **Create/Delete**

*These operations create or delete instances of a service. For a database this could mean creating/deleting a schema in an existing multitenant cluster or creating/deleting a dedicated database cluster.*

## **Bind/Unbind**

*These operations create or delete unique credential sets for an existing service instance that can then be injected into the environment of an application instance.*

## A Bit of Review


Your instance of *Spring Music* should still be running from the end of [Lab 5](#). Visit the application in your browser by hitting the route that was generated by the CLI:


# Albums

[ view as:   | sort by: [title](#) [artist](#) [year](#) [genre](#)  | [+add an album](#) ]


Profiles: cloud,in-memory


Services:


**IV**  
**Led Zeppelin**  
1971  
Rock  


**Nevermind**  
**Nirvana**  
1991  
Rock  



**What's Going On**  
**Marvin Gaye**  
1971  
Rock  


**Are You Experienced?**  
**Jimi Hendrix Experience**  
1967  
Rock  



**The Joshua Tree**  
**U2**  
1987  
Rock  



**Abbey Road**  
**The Beatles**  
1969  
Rock  


**Rumours**  
**Fleetwood Mac**  
1977  
Rock  


**Sun Sessions**  
**Elvis Presley**  
1976  
Rock  


**Thriller**  
**Michael Jackson**  
1982  
Pop  


**Exile on Main Street**  
**The Rolling Stones**  
1972  
Rock  


**Born to Run**  
**Bruce Springsteen**  
1975  
Rock  


**London Calling**  
**The Clash**  
1980  
Rock  


The information dialog in the top right-hand corner indicates that we're currently running with an in-memory database, and that we're not bound to any services. Let's change that.

## The Services Marketplace

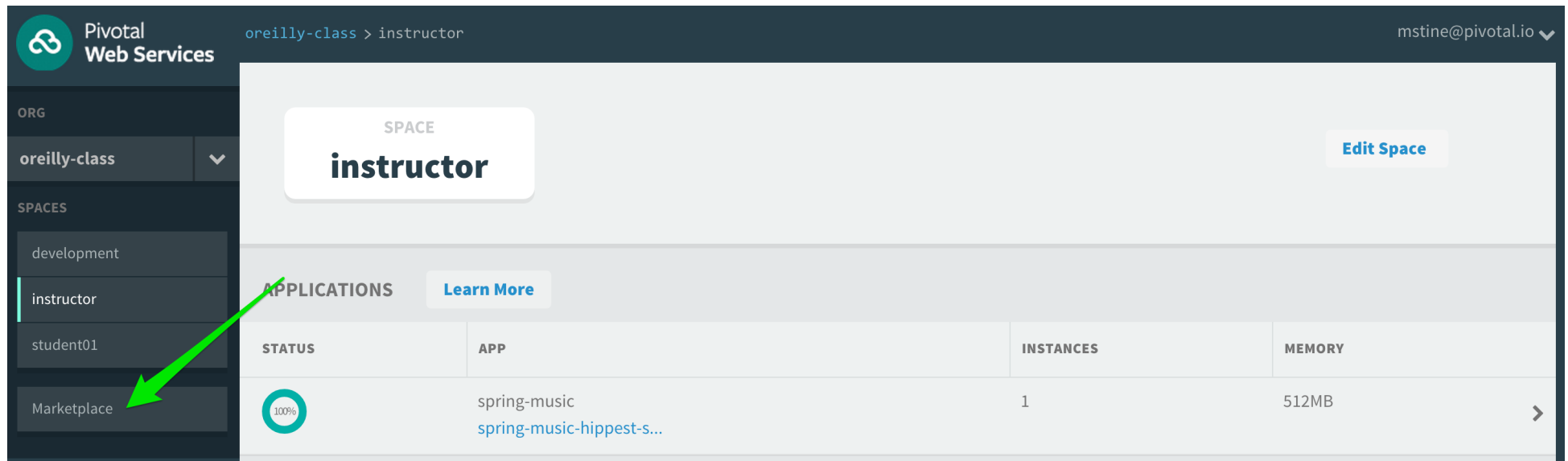
There are two ways to discover what services are available on Pivotal Web Services. The first is available on any instance of Cloud Foundry: the CLI. Just type:

```
$ cf marketplace
```

and you'll get a list of services, their available plans, and descriptions.

The second way is specific to Pivotal Cloud Foundry's Application Manager UI. If you haven't already, login to it by visiting {CF-CONSOLE-URL}.


Click on the "Marketplace" link: *The Screenshots below are from Pivotal Web Services (public managed Pivotal CF). If you are using Pivotal Cloud Foundry, you will see something similar.*



The screenshot displays the Pivotal Web Services Application Manager UI. The top navigation bar includes the Pivotal logo, the text 'Pivotal Web Services', the breadcrumb 'oreilly-class > instructor', and the user email 'mstine@pivotal.io'. The left sidebar shows the 'oreilly-class' organization with a dropdown arrow, and a list of spaces: 'development', 'instructor' (selected), 'student01', and 'Marketplace' (highlighted with a green arrow). The main content area for the 'instructor' space features an 'Edit Space' button and a table of applications. The table has columns for 'STATUS', 'APP', 'INSTANCES', and 'MEMORY'. One application is listed: 'spring-music' with a status of 100%, 1 instance, and 512MB memory. A 'Learn More' button is located above the table.

STATUS	APP	INSTANCES	MEMORY
100%	spring-music spring-music-hippest-s...	1	512MB

and you'll see the same service/plan/description listing in the browser:

**Pivotal  
Web Services**

oreilly-class > Marketplace

ORG

oreilly-class

SPACES

development

instructor

student01

Marketplace

Docs

Support


Tools


Blog


Status


## Services Marketplace


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


**Searchify**  
Custom search you control


**BlazeMeter**  
The JMeter Load Testing Cloud


**Redis Cloud**  
Enterprise-Class Redis for Developers

**ClearDB MySQL Database**  
Highly available MySQL for your Apps.

**CloudAMQP**  
Managed HA RabbitMQ servers in the cloud

**ElephantSQL**  
PostgreSQL as a Service

**SendGrid**  
Email Delivery. Simplified.

**MongoLab**  
Fully-managed MongoDB-as-a-Service

Creating and Binding to a Service Instance

1. Let's begin by creating a MySQL instance provided by Pivotal.

From the CLI, let's *create* a p-mysql service instance:

+

```
$ cf create-service p-mysql 100mb-dev spring-music-db
Creating service spring-music-db in org ACME / space instructor as mstine@pivotal.io...
OK
```

1. Next we'll *bind* the newly created instance to our `spring-music` application:

```
$ cf bs spring-music spring-music-db
Binding service spring-music-db to app spring-music in org ACME / space instructor as mstine@pivotal.io...
OK
TIP: Use 'cf restage' to ensure your env variable changes take effect
```

2. Notice the admonition to `Use 'cf restage' to ensure your env variable changes take effect`. Let's take a look at the environment variables for our application to see what's been done. We can do this by typing:

```
$ cf env spring-music
```

The subset of the output we're interested in is located near the very top, titled `System-Provided`:

System-Provided:

```
{
  "VCAP_SERVICES": { (1)
    "p-mysql": [
      {
        "credentials": {
          "hostname": "10.68.50.61",
          "jdbcUrl": "jdbc:mysql://10.68.50.61:3306/cf_ddbe0221_d1fc_478f_b693_69279b2de702?
user=mWsYAON34zbaQPAA\u0026password=CZImCZ9iSNU7HfP2",
          "name": "cf_ddbe0221_d1fc_478f_b693_69279b2de702",
          "password": "CZImCZ9iSNU7HfP2",
          "port": 3306,
          "uri":
"mysql://mWsYAON34zbaQPAA:CZImCZ9iSNU7HfP2@10.68.50.61:3306/cf_ddbe0221_d1fc_478f_b693_69279b2de702?
reconnect=true", (2)
          "username": "mWsYAON34zbaQPAA"
        },
        "label": "p-mysql",
        "name": "spring-music-db",
        "plan": "100mb-dev",
        "tags": [
          "mysql",
          "relational"
        ]
      }
    ]
  }
}

{
  "VCAP_APPLICATION": {
    "application_name": "spring-music",
    "application_uris": [
      "spring-music-unbragging-pourparler.pcf2.fe.gopivotal.com"
    ],
```

```
"application_version": "e3fa423d-69ad-41cb-bd17-0e88427dfe4c",
"limits": {
  "disk": 1024,
  "fds": 16384,
  "mem": 512
},
"name": "spring-music",
"space_id": "080ba4e6-bafc-482b-9a98-42f612eef736",
"space_name": "jfullam",
"uris": [
  "spring-music-unbragging-pourparler.pcf2.fe.gopivotal.com"
],
"users": null,
"version": "e3fa423d-69ad-41cb-bd17-0e88427dfe4c"
}
}
```

1. `VCAP_SERVICES` is a special Cloud Foundry environment variable that contains a JSON document containing all of the information for any services bound to an application.
2. Notice here the unique URI for this instance of MySQL that `spring-music` has been bound to.

3. Now let's *restage* the application, which cycles our application back through the staging/buildpack process before redeploying the application.<sup>[1]</sup>

```
$ cf restage spring-music
```

Once the application is running again, revisit or refresh the browser tab where you have the *Spring Music* application loaded:

Spring Music 🎵

# Albums

[ view as: | sort by: [title](#) [artist](#) [year](#) [genre](#) | [+add an album](#) ]

IV

Led Zeppelin

1971

Rock

Nevermind

Nirvana

1991

Rock

What's Going On

Marvin Gaye

1971

Rock

Are You Experienced?

Jimi Hendrix Experience

1967

Rock

The Joshua Tree

U2

1987

Rock

Abbey Road

The Beatles

1969

Rock

Rumours

Fleetwood Mac

1977

Rock

Sun Sessions

Elvis Presley

1976

Rock

Thriller

Michael Jackson

1982

Pop

Exile on Main Street

The Rolling Stones

1972

Rock

Born to Run

Bruce Springsteen

1975

Rock

London Calling

The Clash

1980

Rock

Profiles: cloud,postgres,postgres-cloud

Services: spring-music-db

As you can see from the information dialog, the application is now utilizing a MySQL database via the `spring-music-db` service.

- (OPTIONAL STEPS)** If you have a MySQL GUI tool handy and you are using a lab environment that has the necessary network access (ask your instructor), you can inspect the music database created. Otherwise, your instructor will demo via a Pivotal VPN connection.



5. In your DB tool, create a new server connection and populate the properties with values from the URI in your `VCAP_SERVICES` environment variable (remember `cf env spring-music` !):

---

1. In this case, we could accomplish the same goal by only *restarting* the application via `cf restart spring-music`. A *restage* is generally recommended because Cloud Foundry buildpacks also have access to injected environment variables and can install or configure things differently based on their values.