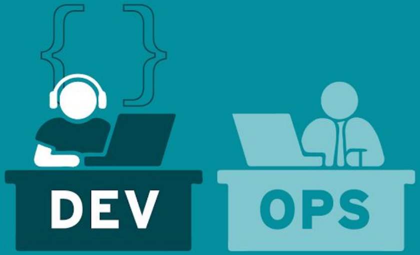


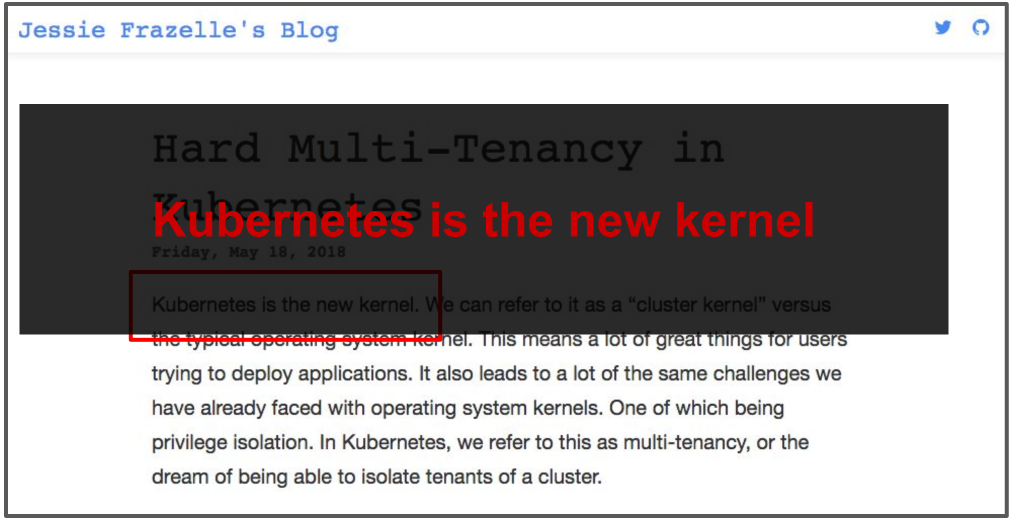
Developing applications on OpenShift as regular developers

DevConf CZ 2019
C228
Friday, January 25 • 3:00pm - 4:50pm

JOSH
GRANT
JAN
TOMAS
JORGE
MAREK

Who do you see on stage?

What’s all this OpenShift and Kubernetes thing?



Jessie Frazelle's Blog

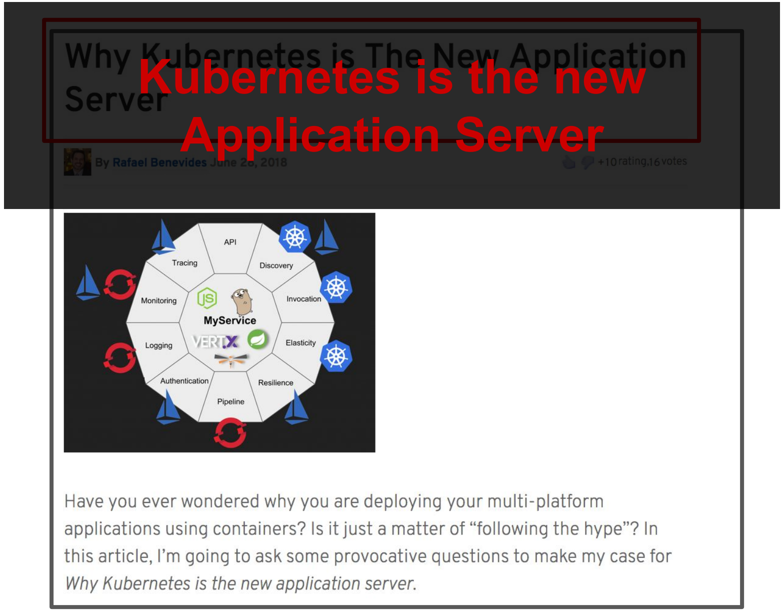
Hard Multi-Tenancy in Kubernetes

Kubernetes is the new kernel

Friday, May 18, 2018

Kubernetes is the new kernel. We can refer to it as a "cluster kernel" versus the typical operating system kernel. This means a lot of great things for users trying to deploy applications. It also leads to a lot of the same challenges we have already faced with operating system kernels. One of which being privilege isolation. In Kubernetes, we refer to this as multi-tenancy, or the dream of being able to isolate tenants of a cluster.

https://blog.jessfraz.com/post/hard-multi-tenancy-in-kubernetes/



Why Kubernetes is The New Application Server

Kubernetes is the new Application Server

By Rafael Benevides June 28, 2018 +10 rating, 16 votes

Have you ever wondered why you are deploying your multi-platform applications using containers? Is it just a matter of "following the hype"? In this article, I'm going to ask some provocative questions to make my case for *Why Kubernetes is the new application server.*

https://developers.redhat.com/blog/2018/06/28/why-kubernetes-is-the-new-application-server/

kubernetes Documentation Blog Partners Community Case Studies English v1.11

Production-Grade Container Orchestration

Automated container deployment, scaling, and management

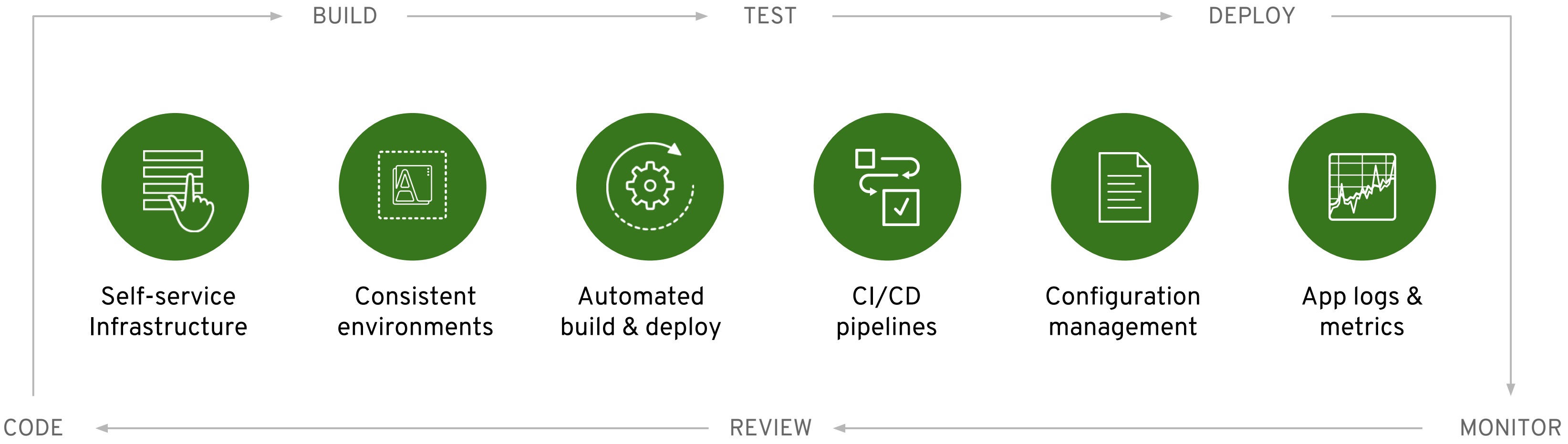
Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications.

It groups containers that make up an application into logical units for easy management and discovery. Kubernetes builds upon 15 years of experience of running production workloads at Google, combined with best-of-breed ideas and practices from the community.

OpenShift extends Kubernetes to make **developer lives easier**

The Kubernetes platform for developers

HOW OPENSHIFT ENABLES DEVELOPER PRODUCTIVITY



BUILD

TEST

DEPLOY

Self-service Infrastructure

Consistent environments

Automated build & deploy

CI/CD pipelines

Configuration management

App logs & metrics

CODE

REVIEW

MONITOR

SPRING & JAVA EE	MICROSERVICES	FUNCTIONS
LANGUAGES	DATABASES	APPLICATION SERVICES
LINUX	RED HAT OPENSHIFT	WINDOWS*

What’s all this Developer Experience thing?

Developer Experience (**DX**) is the equivalent to **User Experience** (**UX**) when **the user** of the software or system **is a developer**.

“It just needs to work! Not look good!”

User Experience (UX)

VS

Developer Experience (DX)

Developers want to have choice

- Choice of architectures
- Choice of programming languages
- Choice of databases
- Choice of application services
- Choice of development tools
- Choice of build and deploy workflows

They don’t want to have to worry about the infrastructure.

Developers want tools to be productive

- Every function they need at hand
- Stable
- Easy to use
- Performant
- No disruptive
- Intuitive

They don’t want to change the way they work.

WHY IT MATTERS {

DX | Developer Experience

Users of your technology are happier, promote it more, and stay longer when the product has good DX

“If you are using a product that combines enterprise-grade functionality with an unusable experience, then your life will be filled with frustration and pain”

Developers Are People Too!

JUST KIDDING

How can we **improve**?

VMs? Containers?
All I want to do is program!
Jeesh.
Chipped
USB
Vertigo
TILDE
5
ROEL BOB '15

Focus on what’s **important**!!!

Focus on code

```
if(top!=self)
function calcWidth() {
    var wW = 0;
    if (typeof window.innerWidth == 'number') {
        wW = window.innerWidth;
    } else if (document.documentElement && docum
        wW = document.documentElement.clientWid
    } else if (document.body && document.body.c
        wW = document.body.clientWidth;
    } else if (
        wW = docum
    }
    if (sH = document.documentElement.scrollH
        var wH = window.innerHeight || docume
        = !document.all && (sH > wH)
        sW =
                'menu', 'wid
```

INTRODUCING: OpenShift-Do (aka. odo)

A developer friendly command line tool

odo is a CLI tool that provides developers with:

- **Simple language to understand**
- **Fast and automated source code deployments**
- **In-context work**
- **Easy iterative development cycle**

Who odo is for?

- Developers not familiar with OpenShift/Kubernetes
- Developers that don’t want to deal with yaml/json
- It does NOT replace oc or kubectl, though minimizes the need to use them

WORKSHOP

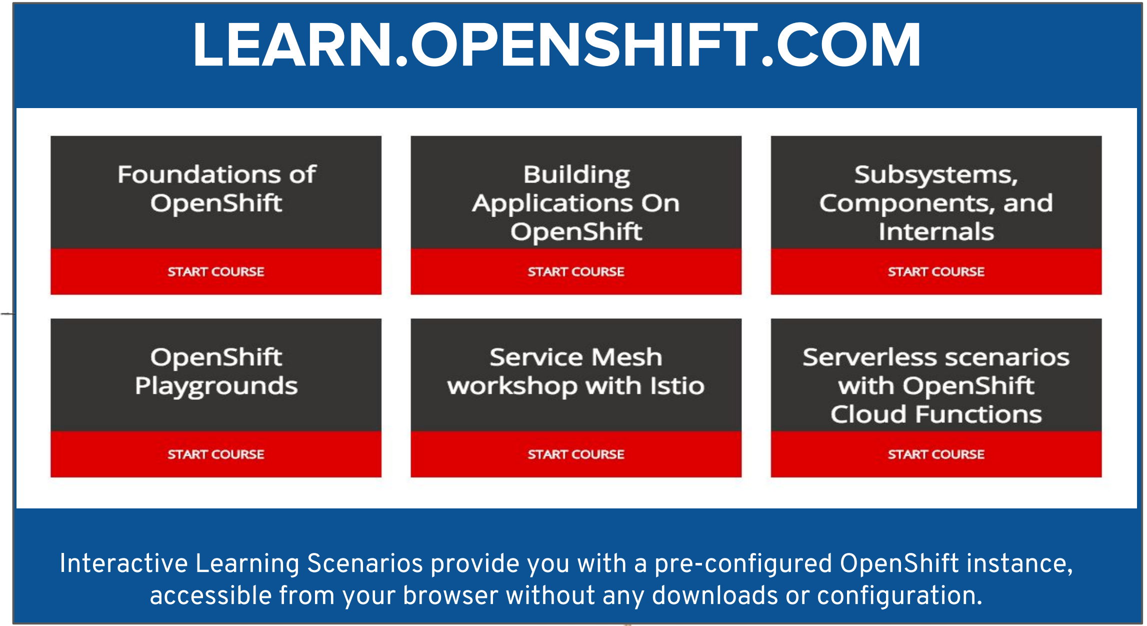
Where?

https://github.com/redhat-developer/odo

Designing a CLI is easy.

Effective CLI design is difficult

Contributions are welcome



LEARN.OPENSHIFT.COM

Foundations of OpenShift
START COURSE

Building Applications On OpenShift
START COURSE

Subsystems, Components, and Internals
START COURSE

OpenShift Playgrounds
START COURSE

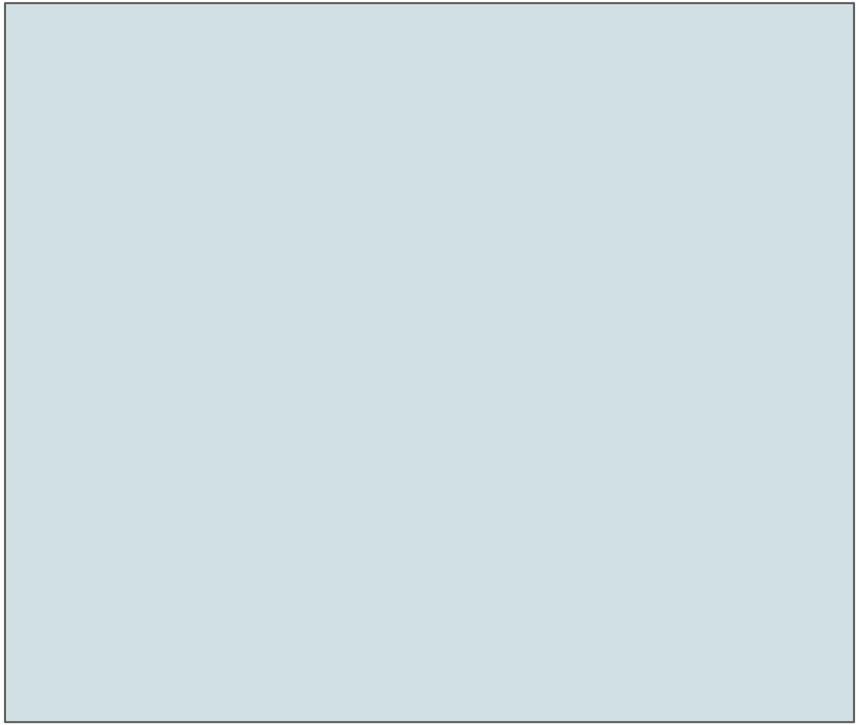
Service Mesh workshop with Istio
START COURSE

Serverless scenarios with OpenShift Cloud Functions
START COURSE

Interactive Learning Scenarios provide you with a pre-configured OpenShift instance, accessible from your browser without any downloads or configuration.

odo basics?

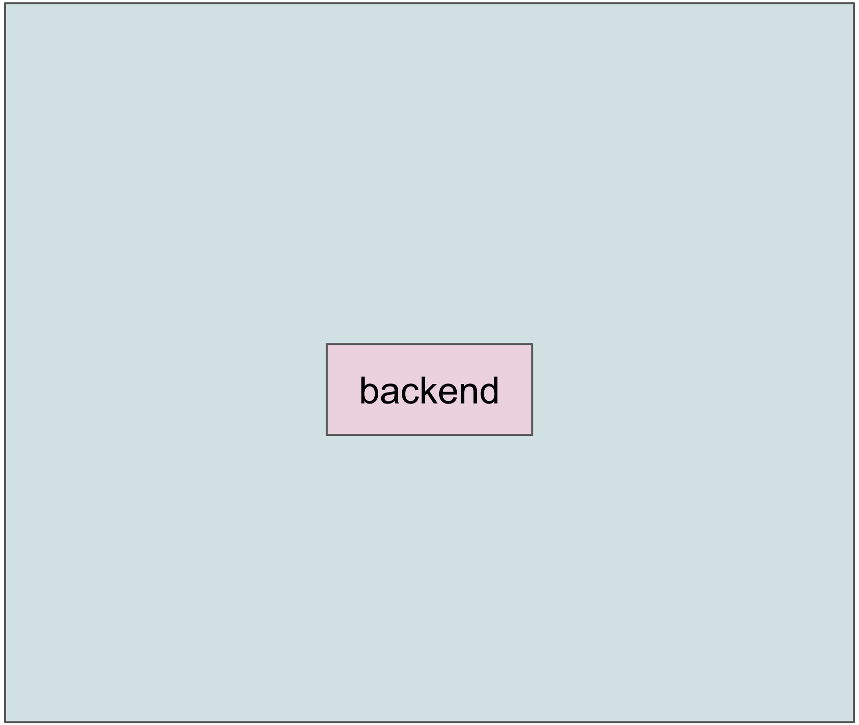
In odo, applications are the basics



```
$ odo application create odo-demo
```

- 🟦 Application
- 🟪 Component
- 🟥 Service

In odo, components are the centerpiece



```
$ odo create wildfly
```

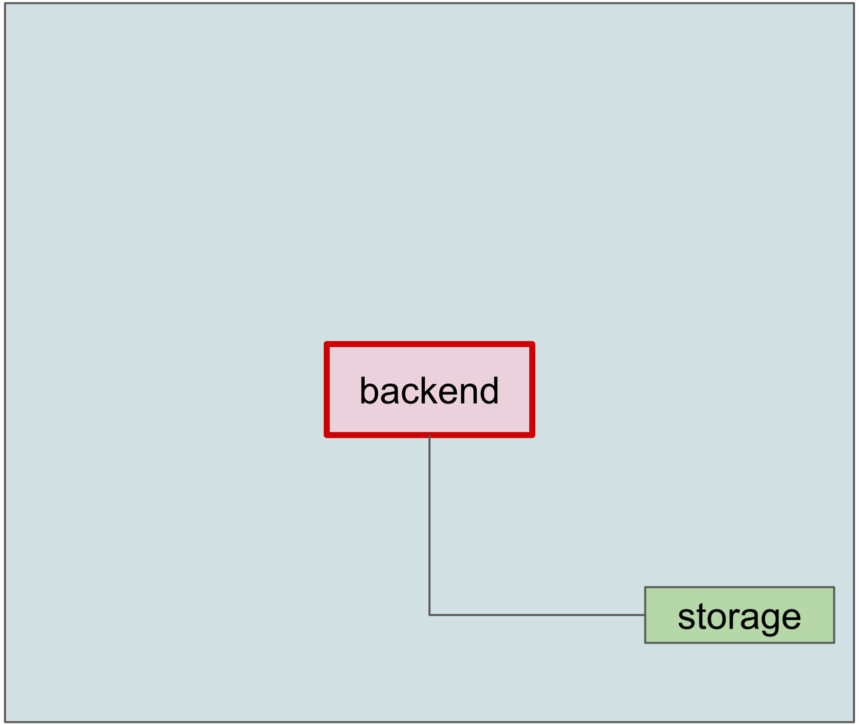
- From local (Iterative development)
- From binary (Iterative development)
- From git (cloud based development)

🟦 Application

🟪 Component

🟥 Service

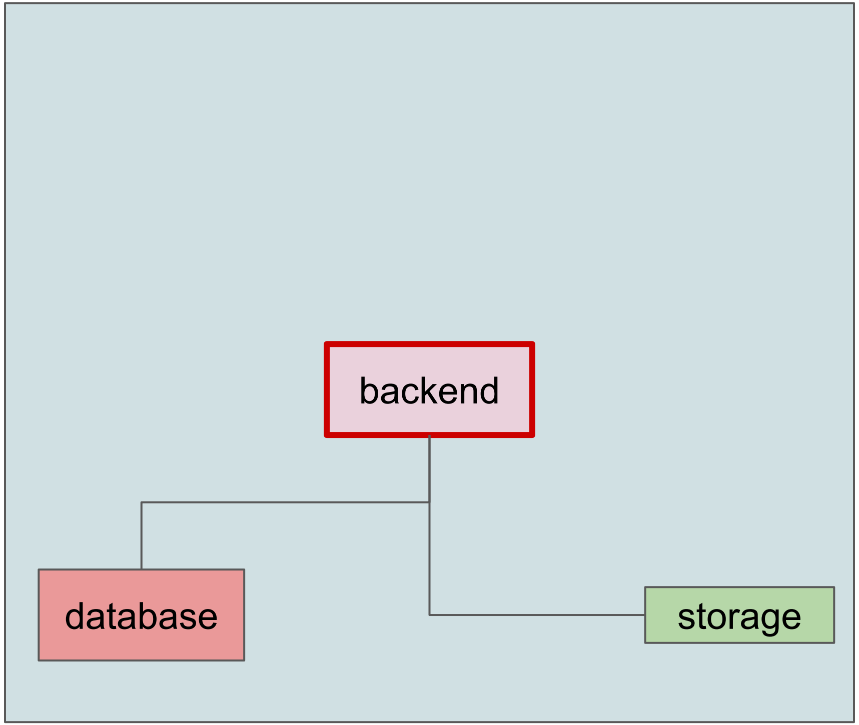
In odo, everything is in context



```
$ odo storage create
        --path=/opt/my-app/data
        --size=1Gi
```

- ⬜ Application
- ⬜ Component
- 🟥 Service

In odo, everything is in context



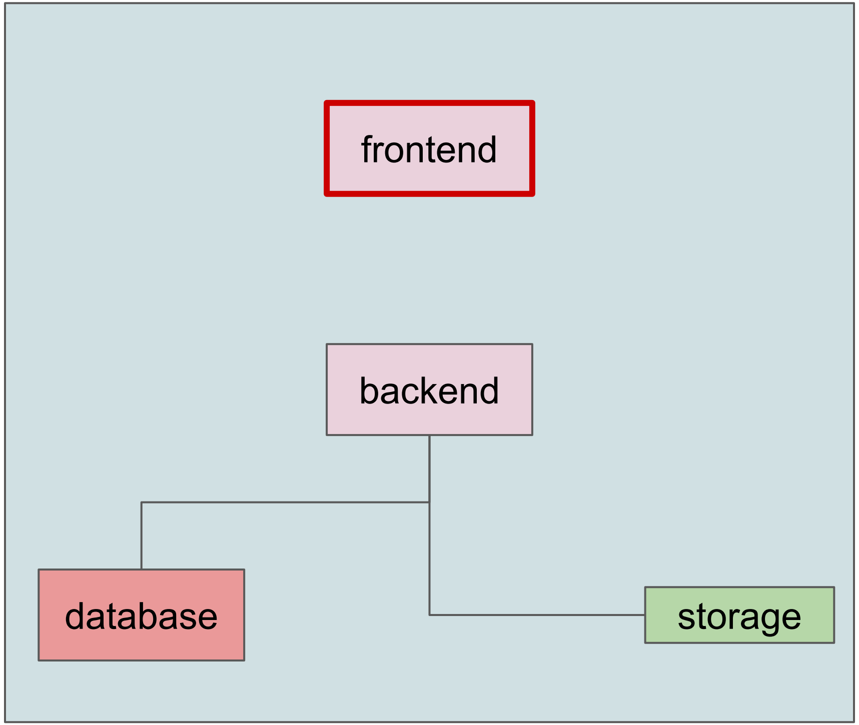
```
$ odo catalog list
$ odo service create database
```

▨ Application

▨ Component

▨ Service

In odo, everything is in context



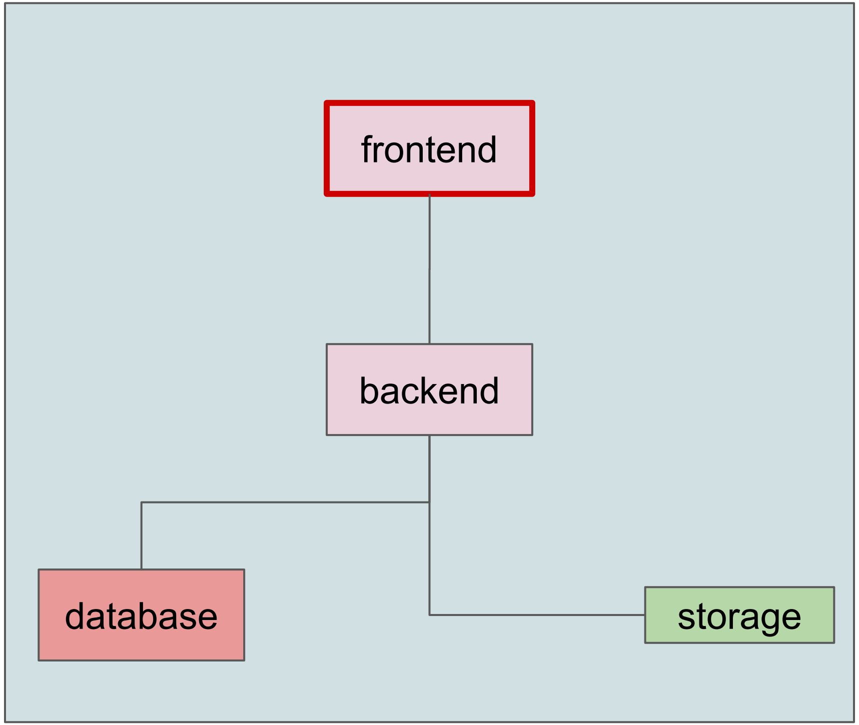
```
$ odo create httpd frontend
```

▨ Application

▨ Component

🟥 Service

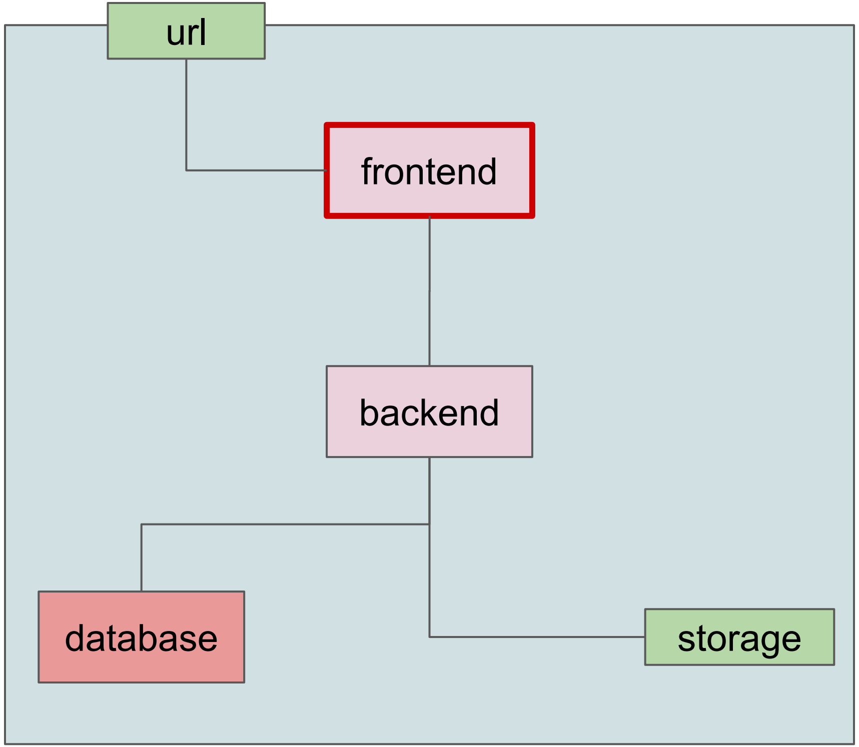
In odo, everything is in context



```
$ odo link backend
```

- 🟦 Application
- 🟪 Component
- 🟥 Service

In odo, everything is in context



```
$ odo url create
```

⬜	Application
▨	Component
🟥	Service

Iterative development

- odo push
  - ○ Push sources that will be built in the container
  - ○ Push binary directly to the container

- odo watch
  - ○ Watches for local changes (on sources or binary) and pushes automatically