Docker + R

Rich FitzJohn

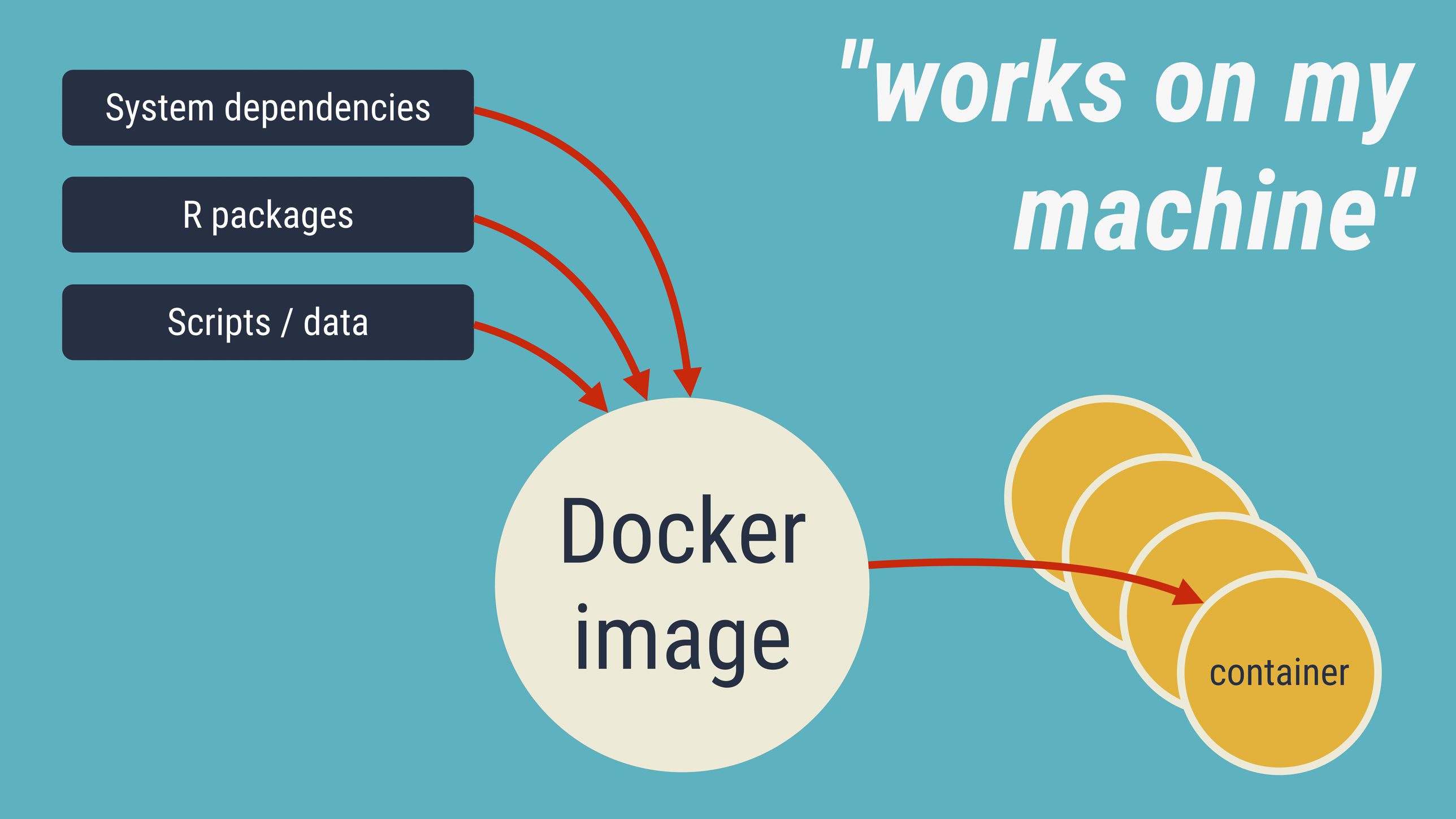
Imperial College London & rOpenSci

richfitz

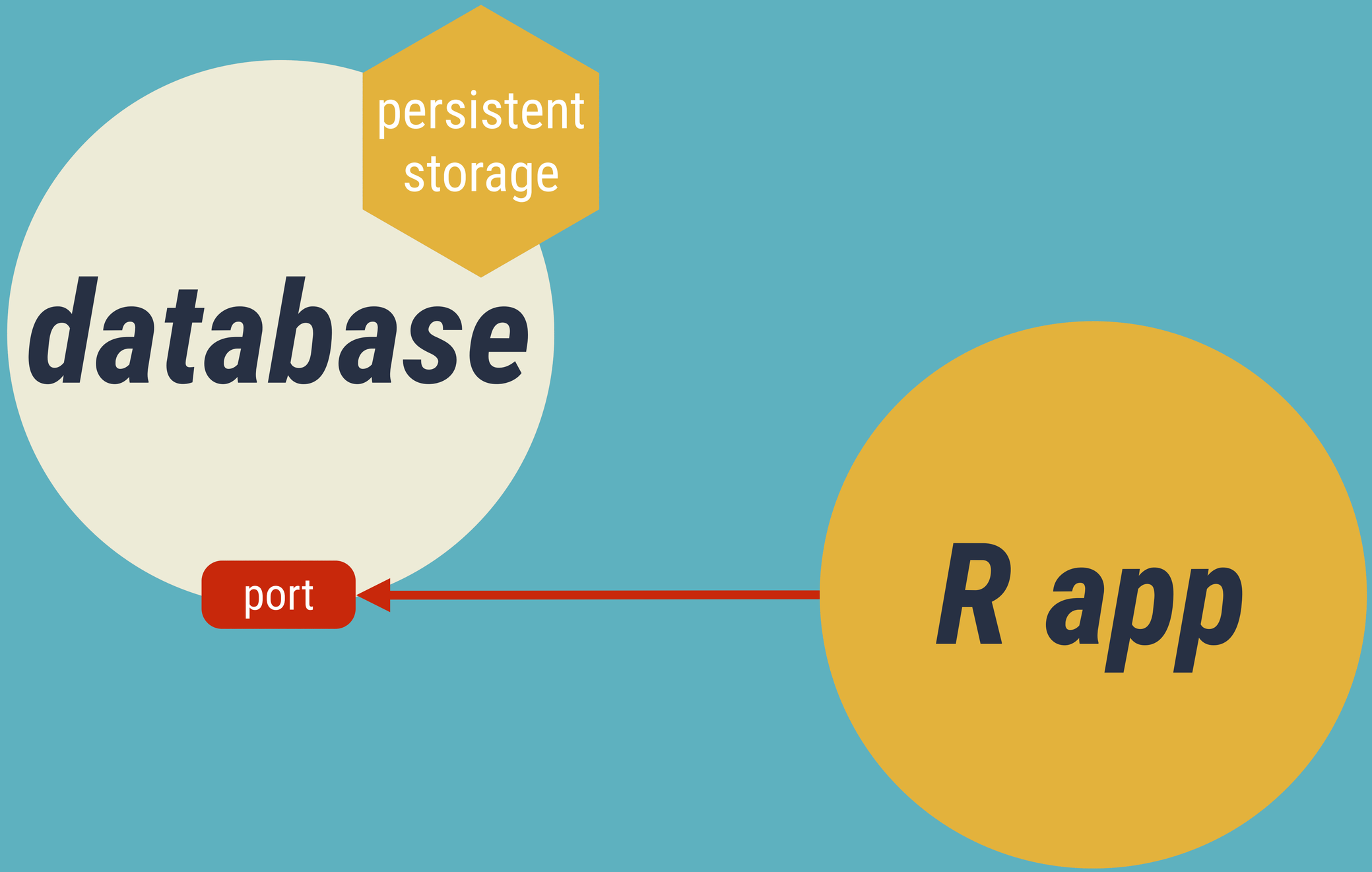

what is docker?

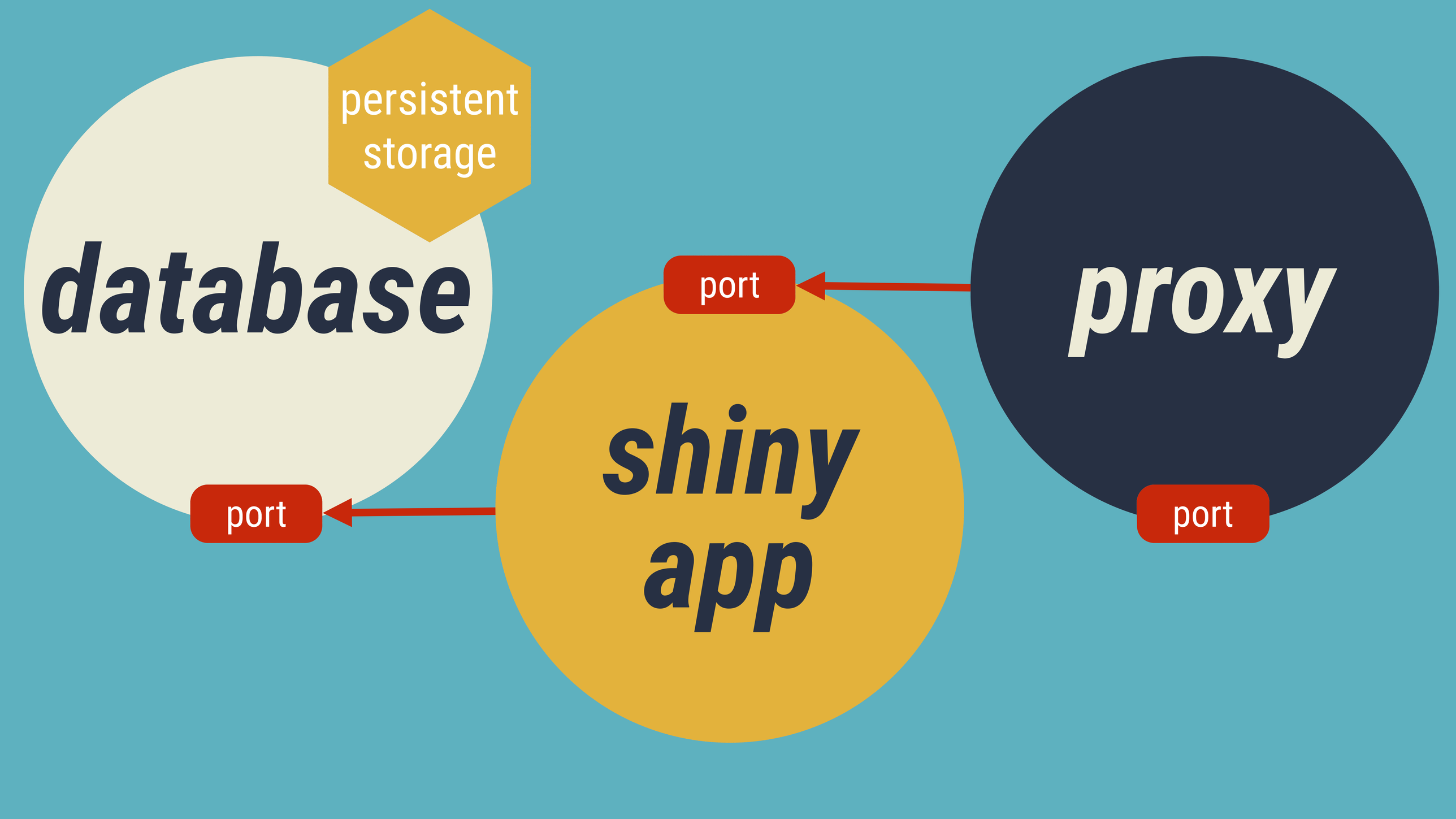
Docker is a software technology providing operating-system-level virtualization also known as containers, promoted by the company Docker, Inc. Docker provides an additional layer of abstraction and automation of operating-system-level virtualization on Windows and Linux. Docker uses the resource isolation features of the Linux kernel such as cgroups and kernel namespaces, and a union-capable file system such as OverlayFS and others to allow independent "containers" to run within a single Linux instance, avoiding the overhead of starting and maintaining virtual machines.


why use docker?
© Lode Van de Velde



"works on my machine"


persistent
storage
database
port
R app


persistent
storage
database
port
port
shiny
app
proxy
port


EUROMAX
CHINA SHIPPING LINE
CSCL GLOBE
HONG KONG

swagger

```
/containers/create:
  post:
   summary: "Create a container"
   consumes:
      - "application/json"
   parameters:
      - name: "name"
        in: "query"
        description: "Assign the specified name to the container."
        type: "string"
   responses:
      201:
        description: "Container created successfully"
        schema:
          type: "object"
         description: "OK response to ContainerCreate operation"
         properties:
            Id:
              description: "The ID of the created container"
              type: "string"
```

where

```
/containers/create:
  post:
   summary: "Create a container"
   consumes:
      - "application/json"
   parameters:
      - name: "name"
        in: "query"
        description: "Assign the specified name to the container."
        type: "string"
   responses:
      201:
        description: "Container created successfully"
        schema:
          type: "object"
         description: "OK response to ContainerCreate operation"
         properties:
            Id:
              description: "The ID of the created container"
              type: "string"
```

parameters

```
/containers/create:
  post:
    summary: "Create a container"
    consumes:
       - "application/json"
    parameters:
       - name: "name"
         in: "query"
         description: "Assign the specified name to the container."
         type: "string"
    responses:
       201:
         description: "Container created successfully"
         schema:
           type: "object"
          description: "OK response to ContainerCreate operation"
          properties:
             Id:
               description: "The ID of the created container"
               type: "string"
```

```
/containers/create:
  post:
    summary: "Create a container"
    consumes:
       - "application/json"
    parameters:
       - name: "name"
         in: "query"
         description: "Assign the specified name to the container."
         type: "string"
    responses:
       201:
         description: "Container created successfully"
         schema:
           type: "object"
          description: "OK response to ContainerCreate operation"
          properties:
             Id:
               description: "The ID of the created container"
               type: "string"
```

returning

```
/containers/create:
  post:
    summary: "Create a container"
    consumes:
      - "application/json"
    parameters:
      - name: "name"
        in: "query"
        description: "Assign the specified name to the container."
        type: "string"
    responses:
      201:
        description: "Container created successfully"
        schema:
          type: "object"
          description: "OK response to ContainerCreate operation"
          properties:
            Id:
              description: "The ID of the created container"
              type: "string"
```

90 methods

10,000 lines

12 versions

```
if tmpfs:
    if version_lt(version, '1.22'):
        raise host_config_version_error('tmpfs', '1.22')
    self["Tmpfs"] = convert_tmpfs_mounts(tmpfs)

if userns_mode:
    if version_lt(version, '1.23'):
        raise host_config_version_error('userns_mode', '1.23')
    self['UsernsMode'] = userns_mode

if pids_limit:
    if version_lt(version, '1.23'):
        raise host_config_version_error('pids_limit', '1.23')
    self["PidsLimit"] = pids_limit

if isolation:
    if version_lt(version, '1.24'):
        raise host_config_version_error('isolation', '1.24')
    self['Isolation'] = isolation

if auto_remove:
    if version_lt(version, '1.25'):
        raise host_config_version_error('auto_remove', '1.25')
    self['AutoRemove'] = auto_remove
```

if else hell

How to write a function

```
add <- function(a, b) {
  a + b
}
```

How to build a function

```
add <- function(a, b) {
  a + b
}
args <- alist(a =, b =)
body <- quote(a + b)
add <- as.function(c(args, body))
```

How to draw an Owl.

"A fun and creative guide for beginners"

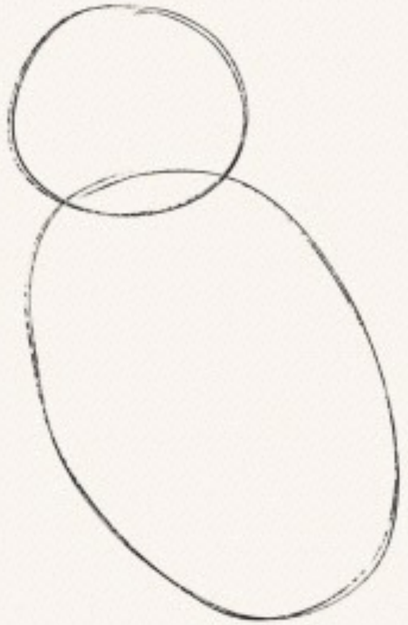


Fig 1. Draw two circles

Fig 2. Draw the rest of the damn Owl

stevedore

```
docker <- stevedore::docker_client()
docker$containers$run(...)
```

Testing packages

1. Install database
2. Configure & set up passwords
3. Use database in package tests
4. Make sure you clean up properly!

```
echo mysql-server mysql-server/root_password password $MYSQL_PASSWORD | \
        debconf-set-selections
echo mysql-server mysql-server/root_password_again password $MYSQL_PASSWORD | \
        debconf-set-selections
apt-get install -y mysql-server

systemctl stop mysql
mv /var/lib/mysql /mnt/data/mysql
ln -s /mnt/data/mysql /var/lib/mysql

echo "alias /var/lib/mysql/ -> /mnt/data/mysql," >> \
    /etc/apparmor.d/tunables/alias
sudo systemctl restart apparmor
systemctl start mysql

mysql -u root -p$MYSQL_PASSWORD -e 'show databases;'| grep teamcity > /dev/null
if [ "$?" = "1" ]; then
    cat > /tmp/database-setup.sql <<EOF
CREATE DATABASE $TEAMCITY_DB_NAME DEFAULT CHARACTER SET utf8 COLLATE utf8_bin;
CREATE USER '$TEAMCITY_DB_USER'@'%' IDENTIFIED BY '$TEAMCITY_DB_PASS';
GRANT ALL ON $TEAMCITY_DB_NAME.* TO '$TEAMCITY_DB_USER'@'%';
EOF
    mysql -u root -p$MYSQL_PASSWORD < /tmp/database-setup.sql
    rm /tmp/database-setup.sql
fi
```

Testing packages

```
env <- c("POSTGRES_PASS" = "s3cret!")
db <- docker$containers$run("postgres", ports = "2222:5432",
                            rm = TRUE, detach = TRUE,
                            env = env)
```

Testing packages

```
env <- c("POSTGRES_PASS" = "s3cret!")
db <- docker$containers$run("postgres", ports = "2222:5432",
                            rm = TRUE, detach = TRUE,
                            env = env)
con <- dbConnect(Postgres(), host = "localhost", port = 2222,
                 user = "postgres", password = "s3cret!")
dbWriteTable(con, "table", mydata)
```

Testing packages

```
env <- c("POSTGRES_PASS" = "s3cret!")
db <- docker$containers$run("postgres", ports = "2222:5432",
                            rm = TRUE, detach = TRUE,
                            env = env)
con <- dbConnect(Postgres(), host = "localhost", port = 2222,
                 user = "postgres", password = "s3cret!")
dbWriteTable(con, "table", mydata)
dbGetQuery(con, "SELECT * FROM table LIMIT 20")
```

Testing packages

```
env <- c("POSTGRES_PASS" = "s3cret!")
db <- docker$containers$run("postgres", ports = "2222:5432",
                            rm = TRUE, detach = TRUE,
                            env = env)
con <- dbConnect(Postgres(), host = "localhost", port = 2222,
                 user = "postgres", password = "s3cret!")
dbWriteTable(con, "table", mydata)
dbGetQuery(con, "SELECT * FROM table LIMIT 20")
db$stop()
```

stevedore

github.com/richfitz/stevedore

Rich FitzJohn

Imperial College London & rOpenSci