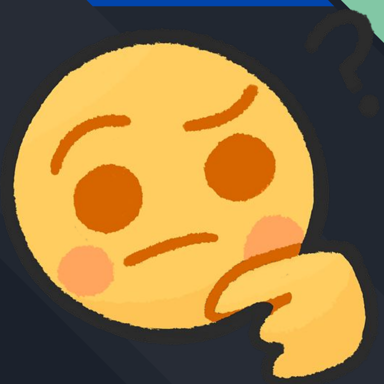
Handling multiple services in a monorepo using **Docker**

Agenda

1. Background
2. Why use Monorepos?
3. Using Docker with monorepos
4. Mentimeter questions
5. Conclusion

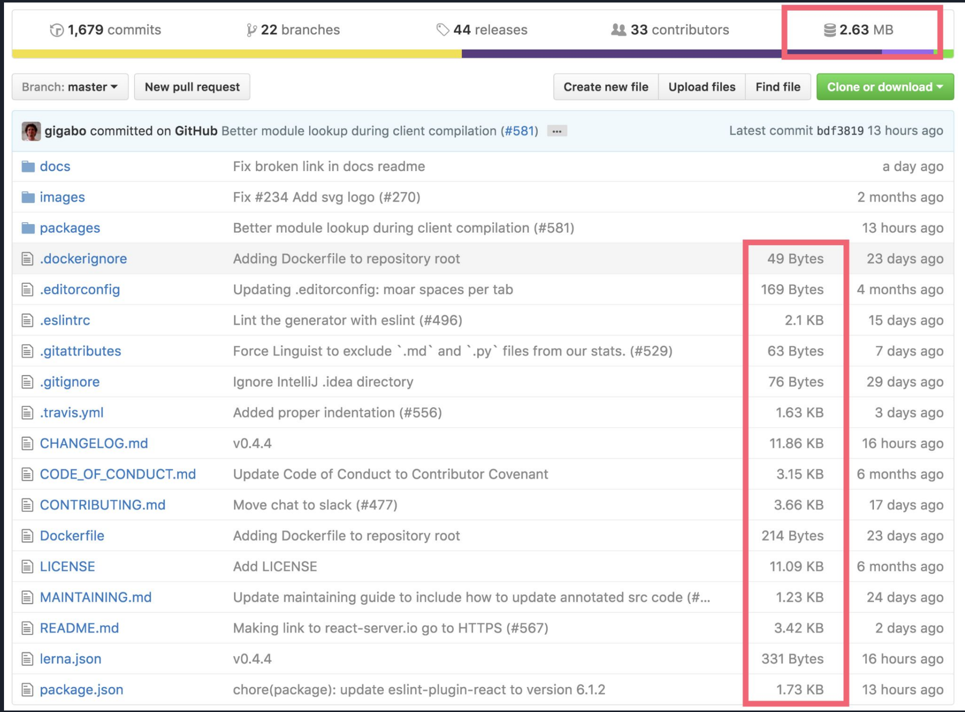
Background

What is a monorepo & why should we care about it?



Github

Repository



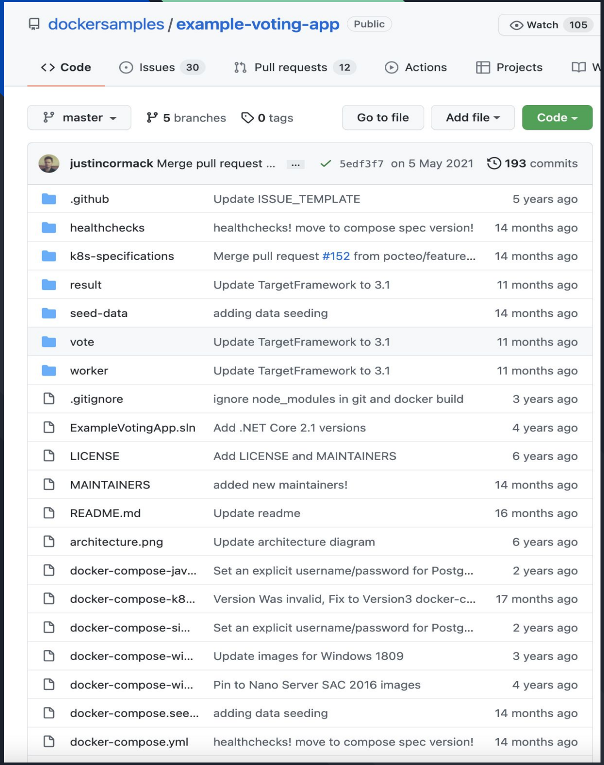
1,679 commits | 22 branches | 44 releases | 33 contributors | 2.63 MB

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

gigabo committed on GitHub Better module lookup during client compilation (#581) ... Latest commit bdf3819 13 hours ago

docs	Fix broken link in docs readme		a day ago
images	Fix #234 Add svg logo (#270)		2 months ago
packages	Better module lookup during client compilation (#581)		13 hours ago
.dockerignore	Adding Dockerfile to repository root	49 Bytes	23 days ago
.editorconfig	Updating .editorconfig: moar spaces per tab	169 Bytes	4 months ago
.eslintrc	Lint the generator with eslint (#496)	2.1 KB	15 days ago
.gitattributes	Force Linguist to exclude `.md` and `.py` files from our stats. (#529)	63 Bytes	7 days ago
.gitignore	Ignore IntelliJ .idea directory	76 Bytes	29 days ago
.travis.yml	Added proper indentation (#556)	1.63 KB	3 days ago
CHANGELOG.md	v0.4.4	11.86 KB	16 hours ago
CODE_OF_CONDUCT.md	Update Code of Conduct to Contributor Covenant	3.15 KB	6 months ago
CONTRIBUTING.md	Move chat to slack (#477)	3.66 KB	17 days ago
Dockerfile	Adding Dockerfile to repository root	214 Bytes	23 days ago
LICENSE	Add LICENSE	11.09 KB	6 months ago
MAINTAINING.md	Update maintaining guide to include how to update annotated src code (#...	1.23 KB	24 days ago
README.md	Making link to react-server.io go to HTTPS (#567)	3.42 KB	2 days ago
lerna.json	v0.4.4	331 Bytes	16 hours ago
package.json	chore(package): update eslint-plugin-react to version 6.1.2	1.73 KB	13 hours ago

Mono repository



dockersamples / example-voting-app Public

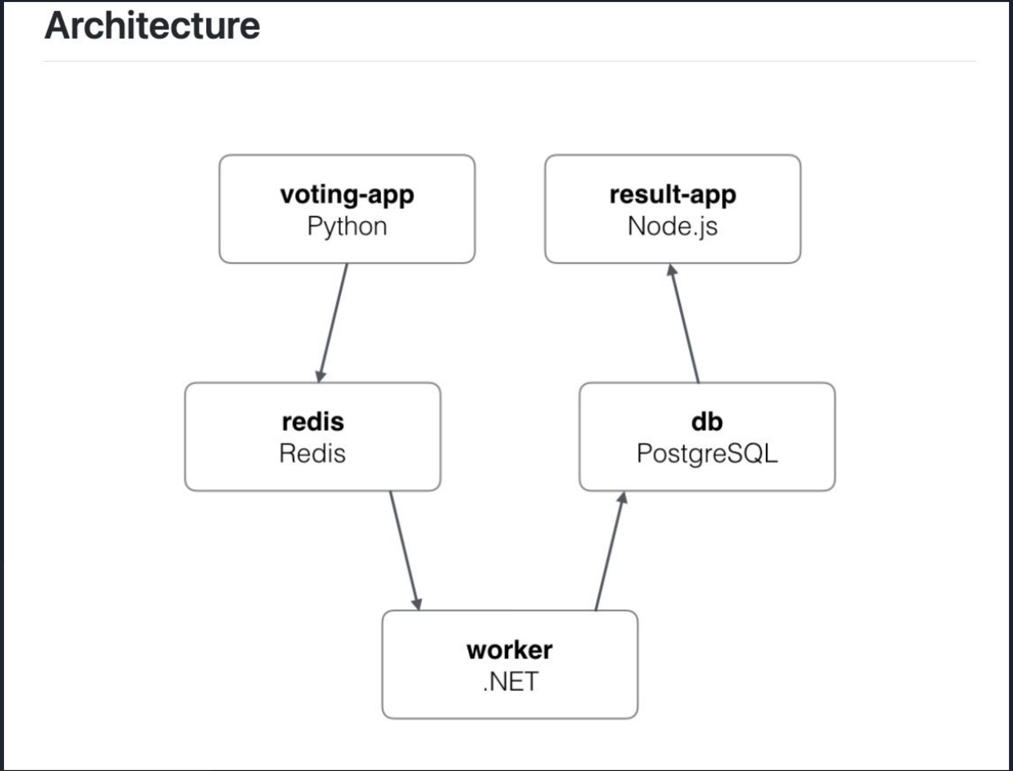
Watch 105

<> Code | Issues 30 | Pull requests 12 | Actions | Projects | W

master | 5 branches | 0 tags | Go to file | Add file | Code

justincormack Merge pull request ... ✓ 5edf3f7 on 5 May 2021 193 commits

.github	Update ISSUE_TEMPLATE	5 years ago
healthchecks	healthchecks! move to compose spec version!	14 months ago
k8s-specifications	Merge pull request #152 from pocteo/feature...	14 months ago
result	Update TargetFramework to 3.1	11 months ago
seed-data	adding data seeding	14 months ago
vote	Update TargetFramework to 3.1	11 months ago
worker	Update TargetFramework to 3.1	11 months ago
.gitignore	ignore node_modules in git and docker build	3 years ago
ExampleVotingApp.sln	Add .NET Core 2.1 versions	4 years ago
LICENSE	Add LICENSE and MAINTAINERS	6 years ago
MAINTAINERS	added new maintainers!	14 months ago
README.md	Update readme	16 months ago
architecture.png	Update architecture diagram	6 years ago
docker-compose-jav...	Set an explicit username/password for Postg...	2 years ago
docker-compose-k8...	Version Was invalid, Fix to Version3 docker-c...	17 months ago
docker-compose-si...	Set an explicit username/password for Postg...	2 years ago
docker-compose-wi...	Update images for Windows 1809	3 years ago
docker-compose-wi...	Pin to Nano Server SAC 2016 images	4 years ago
docker-compose.see...	adding data seeding	14 months ago
docker-compose.yml	healthchecks! move to compose spec version!	14 months ago



So.. are there other alternatives?

Polyrepo

Monorepo

Why use Monorepos?

Polyrepo

- Isolated teams
- Has its own dependencies
- Easy pipelines

Monorepo

- Easy onboarding
- Transparency
- Simplified dependency management

Monorepos also has it's cons.... 😧

A solution

docker

Dockerfile example (vote)

```dockerfile
# Using official python runtime base image
FROM python:3.9-slim

# add curl for healthcheck
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
    curl \
    && rm -rf /var/lib/apt/lists/*

# Set the application directory
WORKDIR /app

# Install our requirements.txt
COPY requirements.txt /app/requirements.txt
RUN pip install -r requirements.txt

# Copy our code from the current folder to /app inside the container
COPY . .

# Make port 80 available for links and/or publish
EXPOSE 80

# Define our command to be run when launching the container
CMD ["gunicorn", "app:app", "-b", "0.0.0.0:80", "--log-file", "-", "--access-logfile", "-"]
```

Dockerfile example (vote)

```dockerfile
# Using official python runtime base image
FROM python:3.9-slim

# add curl for healthcheck
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
    curl \
    && rm -rf /var/lib/apt/lists/*

# Set the application directory
WORKDIR /app

# Install our requirements.txt
COPY requirements.txt /app/requirements.txt
RUN pip install -r requirements.txt

# Copy our code from the current folder to /app inside the container
COPY . .

# Make port 80 available for links and/or publish
EXPOSE 80

# Define our command to be run when launching the container
CMD ["gunicorn", "app:app", "-b", "0.0.0.0:80", "--log-file", "-", "--access-logfile", "-"]
```

Dockerfile example (vote)

```dockerfile
# Using official python runtime base image
FROM python:3.9-slim

# add curl for healthcheck
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
    curl \
    && rm -rf /var/lib/apt/lists/*

# Set the application directory
WORKDIR /app

# Install our requirements.txt
COPY requirements.txt /app/requirements.txt
RUN pip install -r requirements.txt

# Copy our code from the current folder to /app inside the container
COPY . .

# Make port 80 available for links and/or publish
EXPOSE 80

# Define our command to be run when launching the container
CMD ["gunicorn", "app:app", "-b", "0.0.0.0:80", "--log-file", "-", "--access-logfile", "-"]
```

Dockerfile example (vote)

```dockerfile
# Using official python runtime base image
FROM python:3.9-slim

# add curl for healthcheck
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
    curl \
    && rm -rf /var/lib/apt/lists/*

# Set the application directory
WORKDIR /app

# Install our requirements.txt
COPY requirements.txt /app/requirements.txt
RUN pip install -r requirements.txt

# Copy our code from the current folder to /app inside the container
COPY . .

# Make port 80 available for links and/or publish
EXPOSE 80

# Define our command to be run when launching the container
CMD ["gunicorn", "app:app", "-b", "0.0.0.0:80", "--log-file", "-", "--access-logfile", "-"]
```

Dockerfile example (vote)

```dockerfile
# Using official python runtime base image
FROM python:3.9-slim

# add curl for healthcheck
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
    curl \
    && rm -rf /var/lib/apt/lists/*

# Set the application directory
WORKDIR /app

# Install our requirements.txt
COPY requirements.txt /app/requirements.txt
RUN pip install -r requirements.txt

# Copy our code from the current folder to /app inside the container
COPY . .

# Make port 80 available for links and/or publish
EXPOSE 80

# Define our command to be run when launching the container
CMD ["gunicorn", "app:app", "-b", "0.0.0.0:80", "--log-file", "-", "--access-logfile", "-"]
```

Dockerfile example (vote)

```
# Using official python runtime base image
FROM python:3.9-slim

# add curl for healthcheck
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
    curl \
    && rm -rf /var/lib/apt/lists/*

# Set the application directory
WORKDIR /app

# Install our requirements.txt
COPY requirements.txt /app/requirements.txt
RUN pip install -r requirements.txt

# Copy our code from the current folder to /app inside the container
COPY . .

# Make port 80 available for links and/or publish
EXPOSE 80

# Define our command to be run when launching the container
CMD ["gunicorn", "app:app", "-b", "0.0.0.0:80", "--log-file", "-", "--access-logfile", "-"]
```

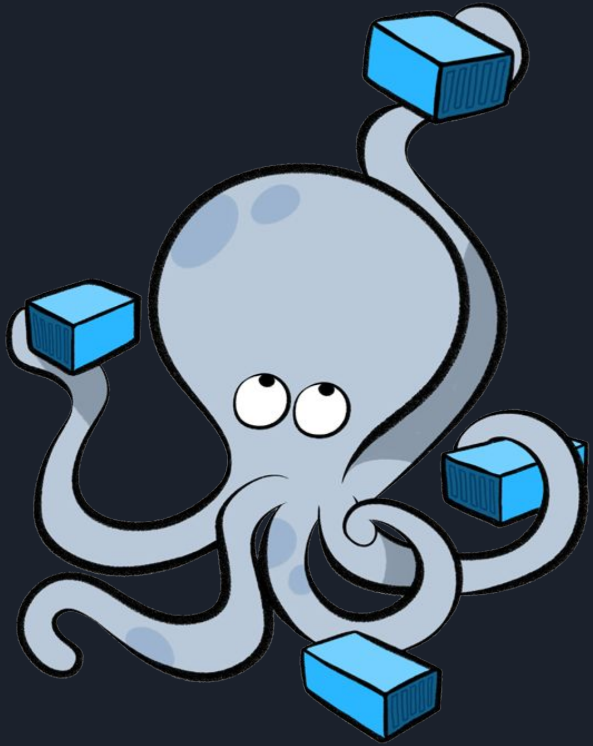
Dockerfile example (vote)

```dockerfile
# Using official python runtime base image
FROM python:3.9-slim

# add curl for healthcheck
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
    curl \
    && rm -rf /var/lib/apt/lists/*

# Set the application directory
WORKDIR /app

# Install our requirements.txt
COPY requirements.txt /app/requirements.txt
RUN pip install -r requirements.txt

# Copy our code from the current folder to /app inside the container
COPY . .

# Make port 80 available for links and/or publish
EXPOSE 80

# Define our command to be run when launching the container
CMD ["gunicorn", "app:app", "-b", "0.0.0.0:80", "--log-file", "-", "--access-logfile", "-"]
```

Dockerfile example (vote)

```dockerfile
# Using official python runtime base image
FROM python:3.9-slim

# add curl for healthcheck
RUN apt-get update \
    && apt-get install -y --no-install-recommends \
    curl \
    && rm -rf /var/lib/apt/lists/*

# Set the application directory
WORKDIR /app

# Install our requirements.txt
COPY requirements.txt /app/requirements.txt
RUN pip install -r requirements.txt

# Copy our code from the current folder to /app inside the container
COPY . .

# Make port 80 available for links and/or publish
EXPOSE 80

# Define our command to be run when launching the container
CMD ["gunicorn", "app:app", "-b", "0.0.0.0:80", "--log-file", "-", "--access-logfile", "-"]
```

Managing multiple containers



Docker Compose

Docker-compose.yml

```yaml
# version is now using "compose spec"
# v2 and v3 are now combined!
# docker-compose v1.27+ required

services:
  vote:
    build: ./vote
    # use python rather than gunicorn for local dev
    command: python app.py
    depends_on:
      redis:
        condition: service_healthy
    volumes:
     - ./vote:/app
    ports:
      - "5000:80"
    networks:
      - front-tier
      - back-tier

  result:
    build: ./result
    # use nodemon rather than node for local dev
    command: nodemon server.js
    depends_on:
```

Docker-compose.yml

```yaml
# version is now using "compose spec"
# v2 and v3 are now combined!
# docker-compose v1.27+ required

services:
  vote:
    build: ./vote
    # use python rather than gunicorn for local dev
    command: python app.py
    depends_on:
      redis:
        condition: service_healthy
    volumes:
     - ./vote:/app
    ports:
      - "5000:80"
    networks:
      - front-tier
      - back-tier

  result:
    build: ./result
    # use nodemon rather than node for local dev
    command: nodemon server.js
    depends_on:
```

Docker-compose.yml

```yaml
# version is now using "compose spec"
# v2 and v3 are now combined!
# docker-compose v1.27+ required

services:
  vote:
    build: ./vote
    # use python rather than gunicorn for local dev
    command: python app.py
    depends_on:
      redis:
        condition: service_healthy
    volumes:
     - ./vote:/app
    ports:
      - "5000:80"
    networks:
      - front-tier
      - back-tier

  result:
    build: ./result
    # use nodemon rather than node for local dev
    command: nodemon server.js
    depends_on:
```

Docker-compose.yml

```yaml
# version is now using "compose spec"
# v2 and v3 are now combined!
# docker-compose v1.27+ required

services:
  vote:
    build: ./vote
    # use python rather than gunicorn for local dev
    command: python app.py
    depends_on:
      redis:
        condition: service_healthy
    volumes:
     - ./vote:/app
    ports:
      - "5000:80"
    networks:
      - front-tier
      - back-tier

  result:
    build: ./result
    # use nodemon rather than node for local dev
    command: nodemon server.js
    depends_on:
```

Docker-compose.yml

```yaml
# version is now using "compose spec"
# v2 and v3 are now combined!
# docker-compose v1.27+ required

services:
  vote:
    build: ./vote
    # use python rather than gunicorn for local dev
    command: python app.py
    depends_on:
      redis:
        condition: service_healthy
    volumes:
     - ./vote:/app
    ports:
      - "5000:80"
    networks:
      - front-tier
      - back-tier

  result:
    build: ./result
    # use nodemon rather than node for local dev
    command: nodemon server.js
    depends_on:
```

Docker-compose.yml

```yaml
# version is now using "compose spec"
# v2 and v3 are now combined!
# docker-compose v1.27+ required

services:
  vote:
    build: ./vote
    # use python rather than gunicorn for local dev
    command: python app.py
    depends_on:
      redis:
        condition: service_healthy
    volumes:
     - ./vote:/app
    ports:
      - "5000:80"
    networks:
      - front-tier
      - back-tier

  result:
    build: ./result
    # use nodemon rather than node for local dev
    command: nodemon server.js
    depends_on:
```

Docker-compose.yml

```yaml
# version is now using "compose spec"
# v2 and v3 are now combined!
# docker-compose v1.27+ required

services:
  vote:
    build: ./vote
    # use python rather than gunicorn for local dev
    command: python app.py
    depends_on:
      redis:
        condition: service_healthy
    volumes:
     - ./vote:/app
    ports:
      - "5000:80"
    networks:
      - front-tier
      - back-tier

  result:
    build: ./result
    # use nodemon rather than node for local dev
    command: nodemon server.js
    depends_on:
```

Docker-compose.yml

```yaml
# version is now using "compose spec"
# v2 and v3 are now combined!
# docker-compose v1.27+ required

services:
  vote:
    build: ./vote
    # use python rather than gunicorn for local dev
    command: python app.py
    depends_on:
      redis:
        condition: service_healthy
    volumes:
     - ./vote:/app
    ports:
      - "5000:80"
    networks:
      - front-tier
      - back-tier

  result:
    build: ./result
    # use nodemon rather than node for local dev
    command: nodemon server.js
    depends_on:
```

Mentimeter time

Take-home

Having trouble managing all your different services?

Consider using docker-compose in a Monorepo