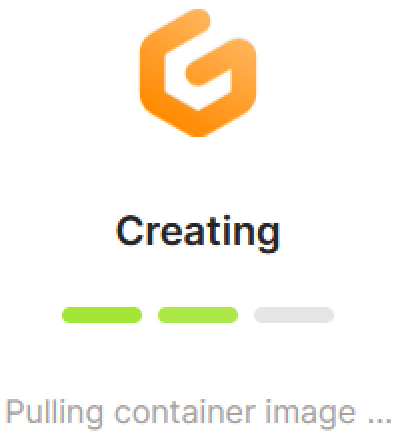
Gitpod

Dev environments as a code

- Gitpod is a **container-based** development platform that **provides ready to code developer environments**
- Development teams can **describe their dev environment as code** and spin up a new fresh developer environment as needed
- Gitpod solves some challenging tasks such as
  - Local environment configuration
  - “Oh but it works on my machine!” situations
- Developer environments provisioned are **ephemeral** and include
  - Source code
  - Linux shell
  - File system
  - Database engine
  - VS Code

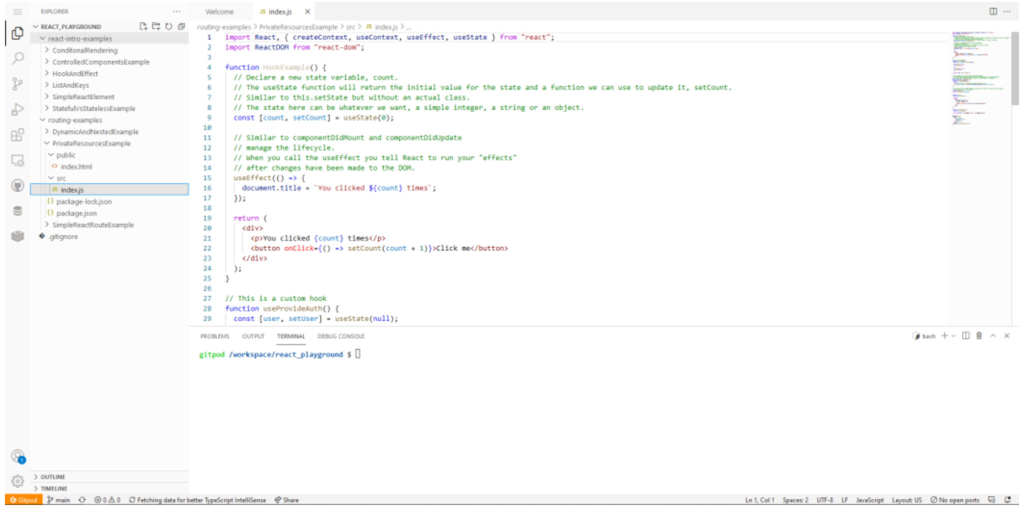
Start coding

- Navigate on a GitHub repository
- Add the *gitpod.io/#* prefix in the URL bar (e.g. *gitpod.io/#https://github.com/gitpod-io/website*)
- Wait for the environment to prepare
- Code ☺



Creating

Pulling container image ...



```
import React, { createContext, useContext, useEffect, useState } from "react";
import ReactDOM from "react-dom";

function HookExample() {
  // Declare a new state variable, count.
  // The useState function will return the initial value for the state and a function we can use to update it, setCount.
  // Similar to this.setState but without an actual class.
  // The state here can be whatever we want, a simple integer, a string or an object.
  const [count, setCount] = useState(0);

  // Similar to componentDidMount and componentDidUpdate
  // manage the lifecycle.
  // When you call the useEffect you tell React to run your "effects"
  // after changes have been made to the DOM.
  useEffect(() => {
    document.title = `You clicked ${count} times`;
  });

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>Click me</button>
    </div>
  );
}

// This is a custom hook
function useProvideAuth() {
  const [user, setUser] = useState(null);
```

```
gitpod /workspace/react_playground $
```

Configure your dev environment

- The Gitpod configuration is defined in a file *.gitpod.yml* that need to be saved in the project’s root directory
- This configuration file contains
  - Tasks definition
  - Port definition
  - Installing databases capability
  - Many other features see here [*https://www.gitpod.io/blog/gitpodify*](https://www.gitpod.io/blog/gitpodify) for a complete list

Task definition

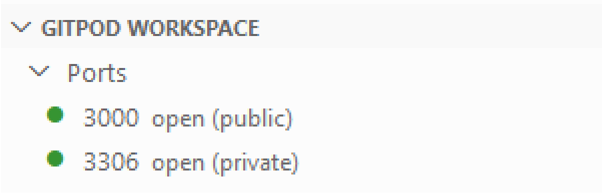
- Used when you need to initialize or install dependencies on your dev environment
- There’re 3 tasks types depending when you’d like to run your task
  - Before → useful for prebuilds happen as first very task
  - Init → most used task to install dependencies
  - Command → happens after the init task
  - See the tasks execution timeline here *https://www.gitpod.io/docs/config-start-tasks*
- You can define multiple tasks that will run in parallel in different terminals
- You can wait for a task to complete with
  - *gp sync-done <name>*
  - *gp sync-await <name>*

```yaml
tasks:
  - name: Install npm-run-all and install all the projects dependencies
    init: npm install
    command: npm run install-all && gp sync-done install-all-task # The name 'install-all-task' is arbitrary

  - name: Start a project
    init: gp sync-await install-all-task # This will wait the install all task
    command: |
      cd react-intro-examples/SimpleReactElement
      npm run start
```

Port definition

- When you’ll run your app it will spin up a web-server listening for clients to a certain port, or you can also start a Database service
- You need to disclosure this ports to Gitpod in order to
  - Being able to test your application in your browser
  - Secure your data from unauthorized usage



GITPOD WORKSPACE

Ports

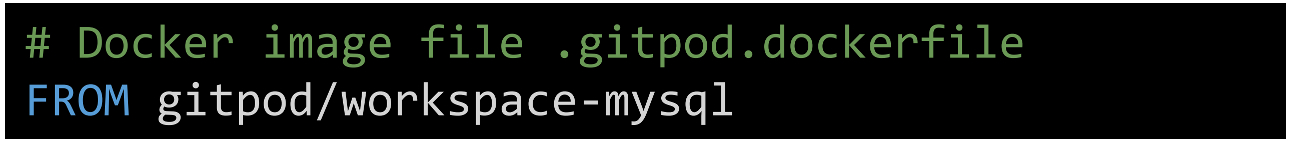
● 3000 open (public)

● 3306 open (private)

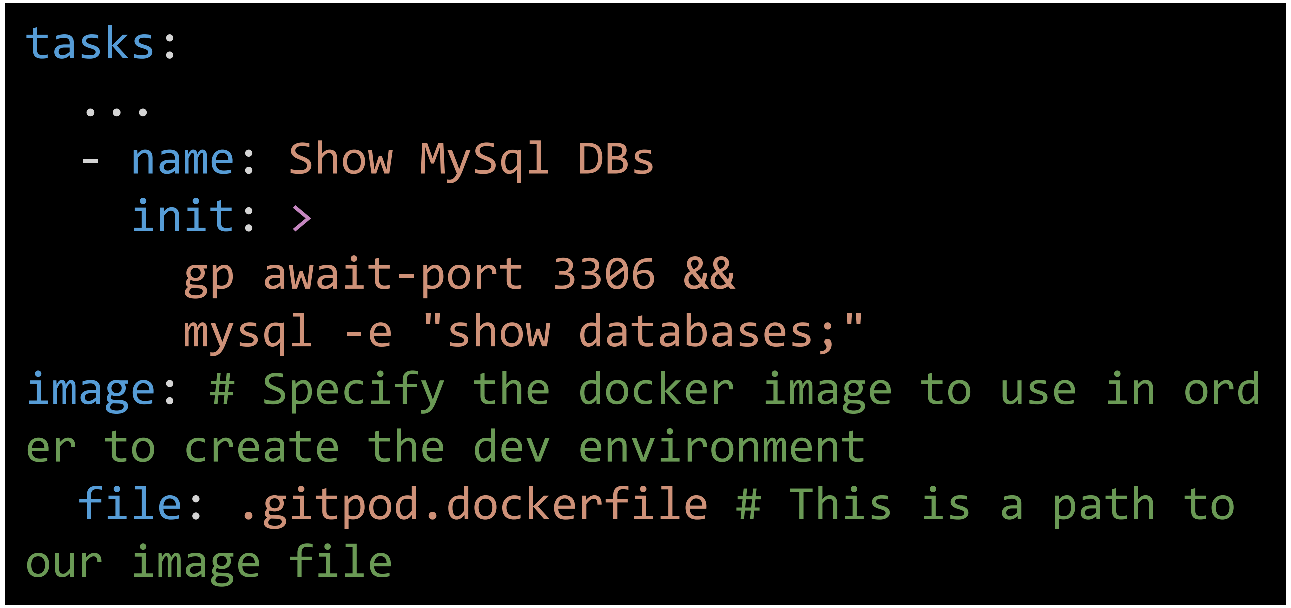
```
ports:
  - port: 3000 # The development web-
server, must be public in order to test your ap
p in your browser
    visibility: public
  -
 port: 3306 # The DB port, this must be private
    visibility: private
```

Install a Database

- You can install a DB by starting from a Gitpod's Docker image that contains the DB service
- You need to define a docker image file starting from a provided image (in our example is *.gitpod.dockerfile*)
- Then you can set the docker image file path in the *.gitpod.yml* configuration file to be used by Gitpod
- Finally you'll be able to use all the DB's command in the terminal or in your *.gitpod.yml* config file useful to prepare your DB
- NB is important to wait for the DB to be ready, this can be done using the *gp await-port <port>* command



```
# Docker image file .gitpod.dockerfile
FROM gitpod/workspace-mysql
```



```
tasks:
  ...
  - name: Show MySql DBs
    init: >
      gp await-port 3306 &&
      mysql -e "show databases;"
image: # Specify the docker image to use in ord
er to create the dev environment
  file: .gitpod.dockerfile # This is a path to
our image file
```