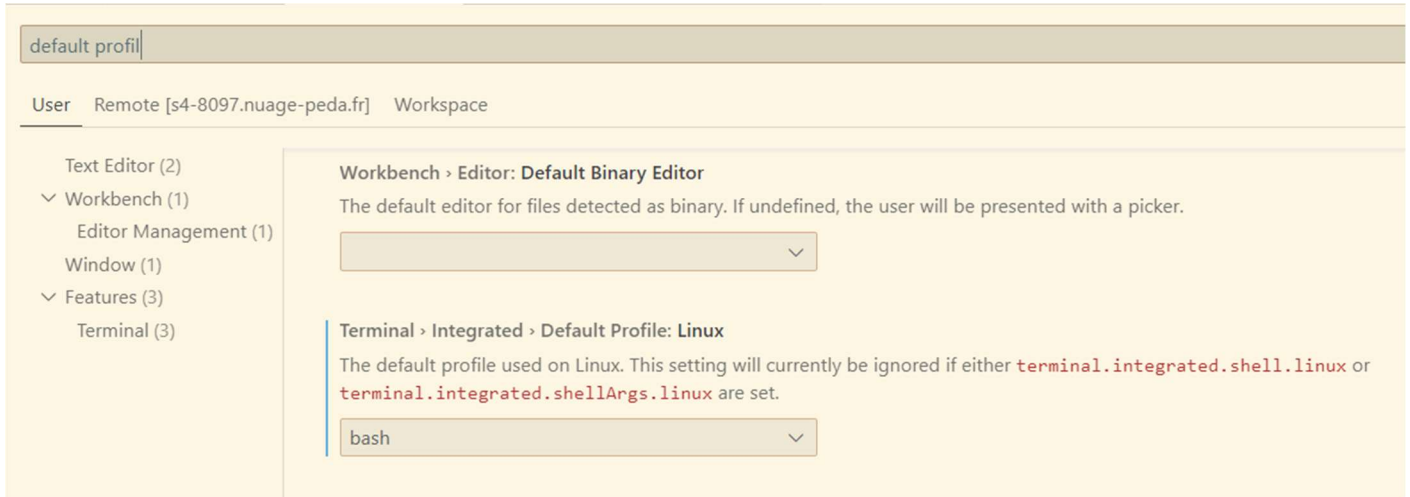


## Doc de Parisot Clément

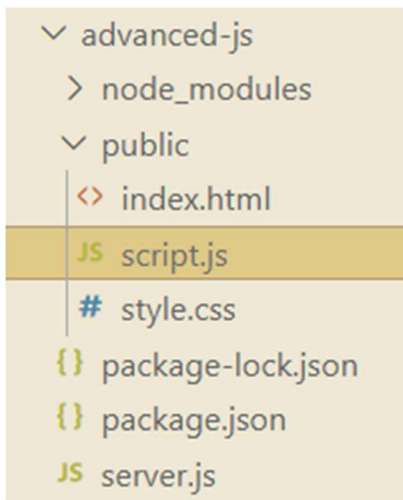
### Config du terminal :



### Check des versions :

```
login8097@REACT-8097:~/react-course$ node -v
v20.11.1
login8097@REACT-8097:~/react-course$ npm -v
10.2.4
```

### Pour un projet JS mis sur un server, architecture cible :



Code du server :

```
1  const express = require('express');
2  const app = express();
3  const path = require('path');
4  const router = express.Router();
5  app.use(express.static(__dirname + '/public'));
6  router.get('/', function (req, res) {
7    |   res.sendFile(path.join(__dirname + '/public/index.html'));
8  });
9  //add the router
10 app.use('/', router);
11 app.listen(process.env.port || 3000);
12 console.log('Running at Port 3000');
```

## JS :

Déclaration de variable :

```
1  const bonjour = 4;
2  let salut = 7;
3  var nePasUtiliser = "Pas bien"
4
```

Template String :

```
const ajouteUnBonjour = (nom) => `Bonjour, ${nom} !`;

console.log(ajouteUnBonjour`John`);
```

- Pratique quand il faut mettre des variables dans un string
- Pratique pour formater des strings

Copie de données :

```
const fruits = {  
  ukouko: 5,  
  ukouka: 5,  
  ukougk: 5,  
  ukouoa: 5,  
};
```

```
const fruits2 = {  
  ...fruits,  
  ukouko2: 5,  
  ukouka2: 5,  
  ukougk2: 5,  
  ukouoa2: 5,  
};
```

Toutes les valeurs de fruits sont dans fruits2

- Pratique pour changer des valeurs à la volés
- Pratique pour copier en ajoutant des infos
- Fonctionne aussi avec les tableaux (attention à l'ordre)

Falsy et Truthy :

```
if (!undefined && !null && !"" && !0 && !NaN) {  
  console.log("tout est faux");  
}  
console.log(undefined, null, "", 0, NaN);
```

Toutes ces valeurs sont fausses, les autres sont vraies

- Pratique pour savoir si un string contient des caractères
- 0 = false

```
const rien = "";
```

```
const jeVeuxUnNom = rien || "Pas de nom";  
console.log(jeVeuxUnNom);
```

- 
- ^ Utile pour mettre un remplacement en cas de non-input !

Sortir des éléments d'objets :

```
const { ukouko } = fruits;  
const { ukouko:ukoukoMaisAutre } = fruits2;  
console.log(ukouko);  
console.log(ukoukoMaisAutre);  
|
```

- Sortir des éléments de bibliothèques importées
- Renommer des variables pour la facilité d'utilisation
- Fonctionne aussi avec les tableaux (Attention à l'ordre)

```
const notes = [10, 6, 13, 18];  
const notesTab = notes.map((note) => note + 2);  
console.log(notesTab);  
const caPasseOuCaCasse = notes.map((note) => note >= 10);  
console.log(caPasseOuCaCasse);
```

Map et création d'un filtre (notes en dessous de 10 sont marquées « false »)

```
function filter(array, value) {  
|   return array.map((val) => (val == value ? null : val));  
| }  
|  
console.log(filter(fruitsArray, "concombre"));
```

Avec une vraie fonction + condition ternaire

ForEach :

```
function isColumnClicked(x) {  
  let allclicked = 0;  
  cases[x].forEach((carre) => (allclicked += carre.clicked));  
  return allclicked == taille;  
}
```

Passage par toutes les cases d'une rangée dans le morpion

Fetch :

```
const setBody = async () => {  
  body = await fetch("https://jsonplaceholder.typicode.com/posts/1")  
    .then((response) => response.json())  
    .then((json) => {  
      body = json.body;  
      console.log("got value in body");  
    })  
    .catch((e) => console.log(e));  
};  
setBody();
```

On récup le json et on le met dans une variable

# INSTALLATION DE REACT

Directement dans le HTML :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>React Kiment</title>
    <link rel="stylesheet" href="./style.css" />
  </head>
  <body>
    <div id="root"></div>
  </body>
</html>

> <div class="container">...
  </div>

  <script src="https://unpkg.com/react@18/umd/react.development.js"></script>
  <script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
```

Fonctionne mais ne donne pas accès à tout le confort accessible

## Mon blog

### Article de blog 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ac ea commodo consequat...

Hiérarchie simple :

```
function setupReact() {
  const container = document.getElementById("root");
  const root = ReactDOM.createRoot(container);

  const titre = React.createElement("h1", {}, "Mon blog");
  const secondTitre = React.createElement("h2", {}, "Article de blog 1");
  const p = React.createElement(
    "p",
    {},
    "Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna alic
  );

  const div = React.createElement("div", {}, [secondTitre, p]);
  const gene = React.createElement("div", {}, [titre, div]);
  root.render(gene);
}
```

## Installation d'un projet Vite :

```
create-vite@5.2.2
Ok to proceed? (y) y
✓ Project name: ... react-learn
? Select a framework: > - Use arrow-keys. Return to submit.
> Vanilla
? Select a framework: > - Use arrow-keys. Return to submit.
  Vanilla
? Select a framework: > - Use arrow-keys. Return to submit.
  Vanilla
? Select a framework: > - Use arrow-keys. Return to submit.
  Vanilla
? Select a framework: > - Use arrow-keys. Return to submit.
  Vanilla
✓ Select a framework: > React
✓ Select a variant: > JavaScript
```

```
✓ Project name: ... react-learn
✓ Select a framework: > React
✓ Select a variant: > JavaScript + SWC

Scaffolding project in /var/www/html/react-course/react-learn...

Done. Now run:
```

Ne pas oublier d'ajouter ça pour le nuage péda :

```
{} package.json X
react-learn > {} package.json > {} scripts >  dev
1  {
2    "name": "react-learn",
3    "private": true,
4    "version": "0.0.0",
5    "type": "module",
6    "scripts": {
7      "dev": "vite --port 3000",
8      "build": "vite build",
9      "lint": "eslint . --ext js,jsx --report-unused-di
10     "preview": "vite preview"
11   },
12   "dependencies": {
13     "react": "^18.2.0",
14     "react-dom": "^18.2.0"
```



Lancer le server :

```
login8097@REACT-8097:~/react-course/react-learn$ npm run dev  
  
> react-learn@0.0.0 dev  
> vite --port 3000  
  
VITE v5.1.6 ready in 193 ms  
  
→ Local:   http://localhost:3000/  
→ Network: use --host to expose  
→ press h + enter to show help
```

Architecture cible pour un projet propre :

- ▼ react-learn
  - > node\_modules
  - > public
  - ▼ src
    - 🌀 App.jsx
    - 🌀 main.jsx
  - 🔗 .eslintrc.cjs
  - 🔗 .gitignore
  - <> index.html
  - {} package-lock.json
  - {} package.json
  - 📄 README.md
  - JS vite.config.js
  - 🔗 .gitignore



TodoList en JSX :

# TodoList :

Element 1 ☐

Element 2 ☐

Element 3 ☐

Element 4 ☐

```
function App() {  
  return (  
    <>  
      <h1>TodoList :</h1>  
  
      <div>  
        <p>Element 1 <input type="checkbox" name="" id="" /></p>  
        <p>Element 2 <input type="checkbox" name="" id="" /></p>  
        <p>Element 3 <input type="checkbox" name="" id="" /></p>  
        <p>Element 4 <input type="checkbox" name="" id="" /></p>  
      </div>  
    </>  
  );  
}  
export default App;
```