

Web : Utilisation de Vite avec PHP

Bachelier en Informatique développement d'applications

Cédric Peeters

cedric.peeters@hers.be



1.1 Projet PHP

- Dans htdocs de xampp, on crée le dossier pour notre projet. Ici je l'ai nommé « ViteExo3Chap2 » car je vais reprendre l'exercice 3 du chapitre 2. Dedans, j'y crée un dossier « public » et un dossier « src ». Pour reprendre la structure proposée sur le site de Bootstrap (<https://getbootstrap.com/docs/5.3/getting-started/vite/>), ajoutons un sous-dossier « js » et un sous-dossier « scss » dans le dossier « src ».
- Je récupère les fichiers de l'exercice 3 du chapitre 2 depuis Moodle. Je place :
 - le fichier php dans le dossier « public » et je le renomme « index.php » juste pour me faciliter les choses au niveau de l'URL qu'il faudra taper pour charger cette page.
 - le fichier css dans le dossier « src/scss ».

1.2 Vite (1/9)

- Le projet existe déjà. On ouvre une invite de commande dans le projet (donc dans le dossier « ViteExo3Chap2 ») afin d'y ajouter Vite :

- Pour avoir le fichier package.json, on y tape la commande suivante :

`npm init -y.`

- On y installe Vite à l'aide de cette commande :

`npm install -D vite`

(le -D est pour dire que c'est une dépendance de développement)

1.2 Vite (2/9)

- Dans le fichier package.json, en plus du script « test » présent par défaut, on ajoute le script "dev" et le script "build".

```
    "scripts": {  
      "test": "echo \"Error: no test specified\" && exit 1",  
      → "dev": "vite",  
      → "build": "vite build"  
    },
```

- Dans le dossier « src/js », on ajoute un fichier « main.js ».
- Et dans ce fichier, on lui indique d'importer notre fichier css :

```
import '../scss/exercice3.css'
```

1.2 Vite (3/9)

- Dans notre fichier PHP (qui est dans le dossier « public »), il faut ajouter une balise `<script>` afin de lui indiquer qu'il doit charger ce fichier javascript que l'on vient de créer. Et vu que ce fichier indique où trouver notre fichier css, on peut retirer la ligne qui permet d'inclure notre css dans notre fichier php. Et on ajoute une autre balise `<script>`.

```
<head>
  <title>Chap 2 : exercice 3</title>
  <meta charset="utf-8"> <!-- permet décrire des accents dans le texte -->
  <script src="http://localhost:5173/src/js/main.js" type="module"></script>
  <script src="http://localhost:5173/@vite/client" type="module"></script>
</head>
```

- Plus d'explications sur ces deux balises `<script>` au slide suivant.

1.2 Vite (4/9)

- Dans l'une de ces balises script, on précise l'attribut « src » auquel on va donner l'URL de notre serveur Vite (à adapter si votre serveur démarre sur autre chose que le port par défaut qui est 5173) que l'on complète avec le chemin de où se trouve notre fichier main.js. On indique également l'attribut type pour lui donner la valeur « module » afin que l'import du fichier js se fasse correctement.
- La seconde de ces balises script nous permettra d'activer le hot reload de Vite.
- Ces deux balises viennent de la section « Backend Integration » du site officiel de Vite : <https://vite.dev/guide/backend-integration.html>

1.2 Vite (5/9)

- Dans notre invite de commande (qui est censée être ouverte à la racine du projet, donc dans le dossier « ViteExo3Chap2 » dans le cas présent), on démarre le serveur de développement via la commande suivante :

`npm run dev`

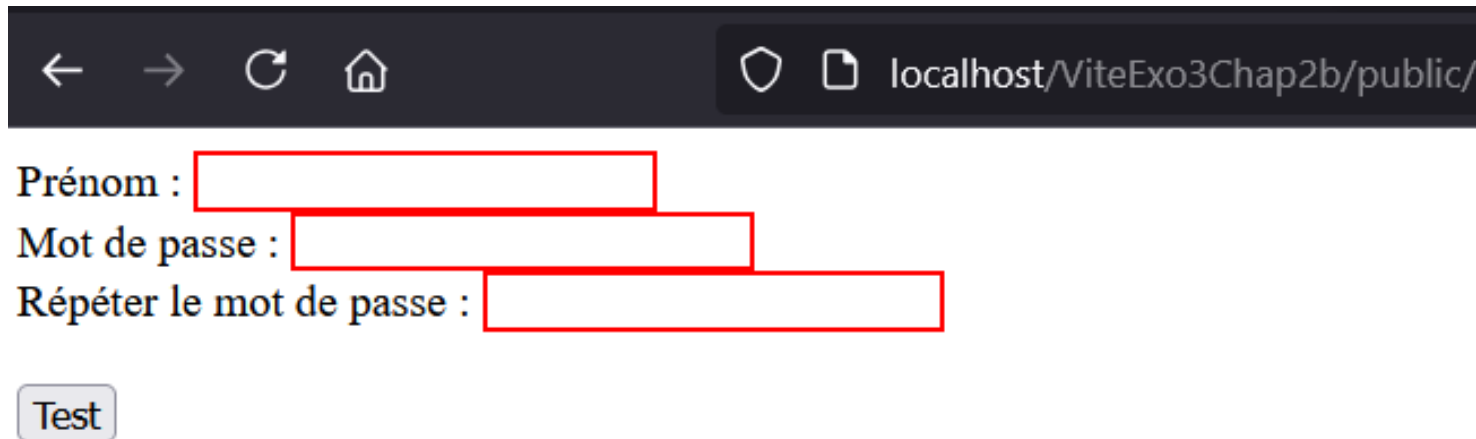
Ce qui donne quelque chose comme ceci une fois fait :

```
VITE                ready in 334 ms

[] Local:   http://localhost:5173/
[] Network: use --host to expose
[] press h + enter to show help
```

1.2 Vite (6/9)

- Vu qu'on utilise PHP, on va démarrer xampp. Dedans, on lance Apache comme on en a l'habitude (et MySQL si vous en avez besoin).
- On se rend à l'URL où se trouve notre fichier index.php et le site se comporte normalement :




A screenshot of a web browser window. The address bar shows the URL `localhost/ViteExo3Chap2b/public/`. Below the address bar, there is a login form with three input fields: "Prénom :", "Mot de passe :", and "Répéter le mot de passe :". Each input field is outlined with a red border. Below the input fields, there is a button labeled "Test".

1.2 Vite (7/9)

- On vient de dire que le site se comporte normalement, mais c'est parce que Vite est démarré. Sans lui, Apache de xampp ne parviendrait pas à inclure le fichier css puisqu'on l'inclut via le script js qui est dans main.js (et c'est vite qui parvient à inclure notre fichier css via le fichier js).

1.2 Vite (8/9)

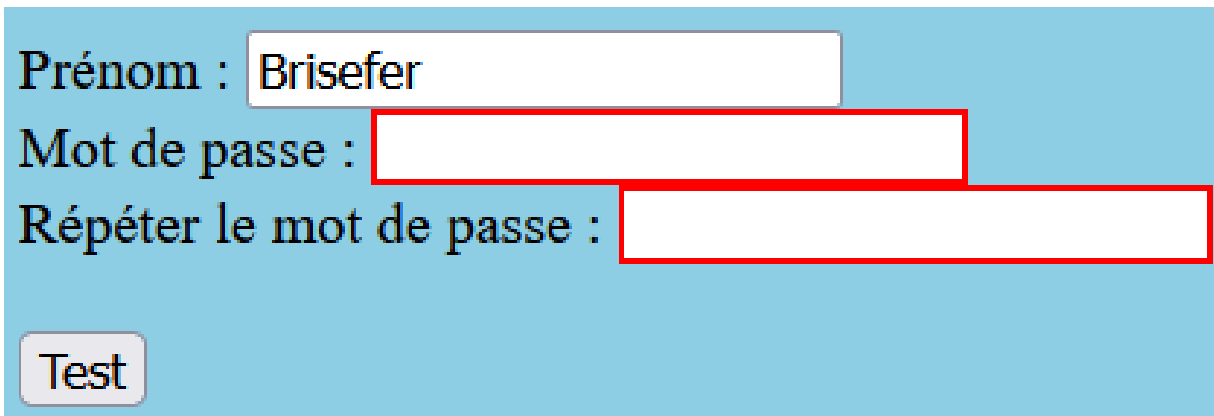
- Modifions légèrement notre fichier css afin d'avoir une couleur en arrière-plan :



```
body{  
    background-color : #8ecee4;  
}  
  
.inputValide{  
  
}  
  
.inputInvalide{  
    border: 2px solid;  
    border-color: red;  
}
```

1.2 Vite (9/9)

- Et c'est là qu'on voit l'intérêt de Vite avec le hot reload. On n'a pas besoin de recharger notre page : Vite va juste appliquer cette modification sans recharger la page à chaque fois qu'on enregistre une modification dans notre fichier css ou notre fichier js.



A screenshot of a web form on a light blue background. The form contains three input fields: 'Prénom : Brisefer', 'Mot de passe :', and 'Répéter le mot de passe :'. The 'Mot de passe' and 'Répéter le mot de passe' fields are outlined with a red border. Below the inputs is a 'Test' button. This illustrates the hot reload feature where changes are applied without a full page refresh.

Prénom : Brisefer

Mot de passe :

Répéter le mot de passe :

Test

1.3 Image en arrière-plan (1/5)

- Maintenant qu'on a changé la couleur de l'arrière-plan, essayons de changer l'arrière-plan en css pour y mettre une image (que j'ai placée aussi dans le dossier « src/img », ce dossier « img » a donc été créé à côté du dossier « scss » et du dossier « js » et que j'ai nommée l'image background.jpg) :

```
body{  
    /* background-color : #8ecee4; */  
    background: url(../img/background.jpg) ;  
}
```

1.3 Image en arrière-plan (2/5)

- Malgré tout, ça ne fonctionne pas. Car il cherche cette image au niveau du serveur vite (et non du dossier local où on travaille).
- Il faut donc ajouter un fichier « vite.config.js » au niveau du dossier du projet (donc dans le dossier « ViteExo3Chap2 » dans notre cas).
- Dans ce fichier, on y colle la configuration disponible sur <https://vite.dev/guide/backend-integration.html> et on la modifie.

1.3 Image en arrière-plan (3/5)

- On y ajoute :
 - l'import de `defineConfig`,
 - un attribut « `server` » et on lui donne l'URL de notre serveur vite (ici celle par défaut en local) sous la clé « `origin` ». Ainsi, il prefixera nos chemins avec cette URL.
 - On modifie la valeur donnée à la clé « `input` » dans « `rollupOptions` » pour y mettre le chemin relatif de là où se situe notre `main.js`.

(code complet de ce fichier au slide suivant)

1.3 Image en arrière-plan (4/5)

- Contenu du fichier :

```
import {defineConfig} from "vite"

export default defineConfig({
  server:{
    origin: 'http://localhost:5173'
  },
  build: {
    // generate .vite/manifest.json in outDir
    manifest: true,
    rollupOptions: {
      // overwrite default .html entry
      input: './src/js/main.js',
    },
  },
})
```

- On relance Vite pour que ces changements de configuration soient pris en compte.

1.3 Image en arrière-plan (5/5)

- Après avoir relancé Vite puis rechargé la page, on voit que l'arrière-plan est bien présent. Ici on a également modifié la couleur du texte dans le css pour que ça ressorte en blanc sur cet arrière-plan foncé.



Prénom :

Mot de passe :

Répéter le mot de passe :

1.4 Bootstrap (1/4)

- Afin d'inclure Bootstrap, modifions tout d'abord l'extension de notre fichier css afin qu'il soit maintenant en .scss plutôt que .css.
- Modifions également dans le fichier main.js afin de refléter ce changement : `import '../scss/exercice3.scss'`
- Installons SASS via la commande suivante en étant à la racine du projet (donc en étant dans le dossier « ViteExo3Chap2 ») :

```
npm install -D sass-embedded
```

1.4 Bootstrap (2/4)

- Installons Bootstrap via la commande suivante en étant à la racine du projet (donc en étant dans le dossier « ViteExo3Chap2 ») :

```
npm i --save bootstrap @popperjs/core
```

- Ensuite on importe Bootstrap dans notre fichier `exercice3.scss`. Ici je l'ai fait via l'option A en ajoutant :
`@import "bootstrap/scss/bootstrap";`

(code complet du fichier au slide suivant)

1.4 Bootstrap (3/4)

- L'import de Bootstrap a été fait au début du fichier :

```
@import "bootstrap/scss/bootstrap";

body{
    /* background-color : #8ecee4; */
    color : white;
    background: url(../img/background.jpg) ;
}

h1{
    color: blue;
}

.inputValide{

}

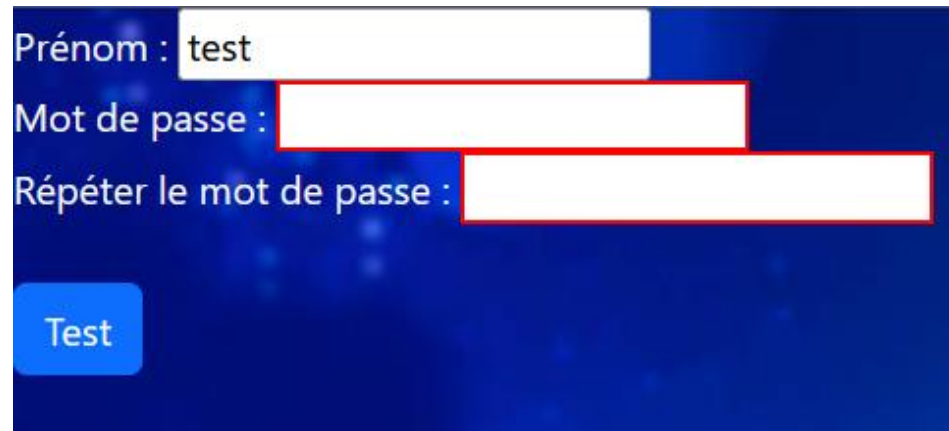
.inputInvalide{
    border: 2px solid;
    border-color: red;
}
```

1.4 Bootstrap (4/4)

- J'ajoute la classe souhaitée à mon bouton qui est en fin de formulaire de mon fichier PHP :

```
<INPUT class="btn btn-primary" TYPE=submit VALUE="Test">  
</FORM>
```

- Et lorsque l'on recharge la page, on voit bien que le style caractéristique de Bootstrap a été appliqué au bouton :



Prénom :

Mot de passe :

Répéter le mot de passe :

The screenshot shows a web form on a dark blue background. It contains three input fields: a text field for 'Prénom' with the value 'test', and two password fields for 'Mot de passe' and 'Répéter le mot de passe'. At the bottom left is a blue submit button with the text 'Test' in white. The button has rounded corners and a slight shadow, characteristic of Bootstrap's 'btn btn-primary' class.

1.5 Build (1/9)

- Vite nous servait pour le développement. On se doute bien qu'en production, on n'aura pas à la fois un serveur Vite et un serveur Apache (ici via xampp) qui sont lancés en même temps juste pour faire tourner notre site.
- On va donc pouvoir « bundler » notre projet pour être en production.
- Cela se fera à l'aide de la commande suivante en étant à la racine (donc en étant dans le dossier « ViteExo3Chap2 ») :

```
npm run build
```

1.5 Build (2/9)

- Cette commande va vous créer un dossier « dist ». Ceci signifie « distributable ». Et c'est là-dedans que notre site à mettre en production a été construit par cette commande.
- Dans ce dossier « dist », vous avez :
 - Un dossier « .vite » qui contient le manifest au format json.
 - Un dossier « assets » qui contient le js compilé en un seul fichier, le scss et l'image de background.
 - Votre page php.

1.5 Build (3/9)

- Notre problème, si on essaye de visualiser ce site comme on en a l'habitude avec xampp, c'est que les balises `<script>` qu'on avait dans notre code étaient spécifique à Vite. Or ici on utilise uniquement Apache (via xampp).
- On doit donc changer ces deux balises pour qu'elles prennent en compte les fichiers générés présents dans assets.
- On pourrait ouvrir le fichier php présent dans « dist » et changer ces balises. Mais ça serait un peu bête de modifier là. C'est mieux de modifier à la source. Donc de modifier le fichier qui est dans « public ».

1.5 Build (4/9)

- Dans le fichier php qui est dans « public », dans le <head>, on va faire un « if else » pour lui indiquer quel <script> placer dans le <head> selon qu'on soit en développement ou en production.
- Pour cela, j'ajoute des balises php. J'y déclare une variable \$production qui contiendra soit true quand on est en production, soit false quand on est en développement.
- Je fais suivre d'un if() qui teste cette variable. Ainsi, si on est en production alors on fera les nouveaux imports (propres à notre dossier dist) et sinon, c'est qu'on est en développement et on fera les imports qu'on a utilisés jusqu'à maintenant.

(code au slide suivant)

1.5 Build (5/9)

- Voici le code du <head> dans notre fichier php :

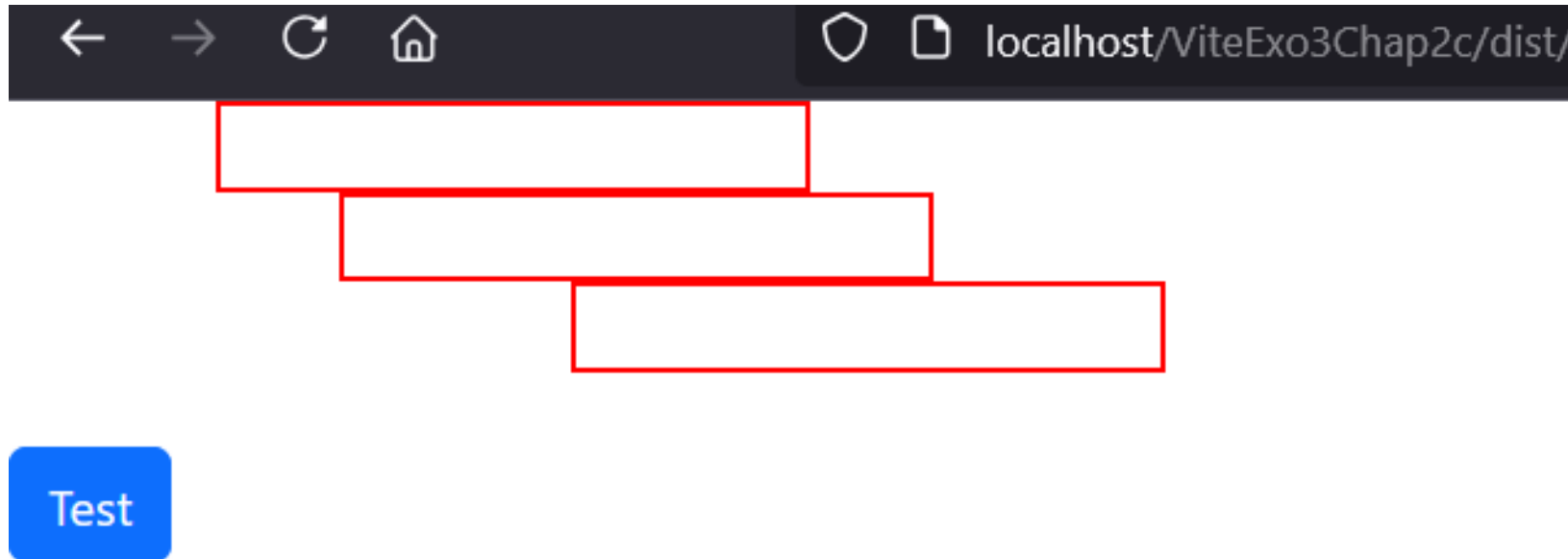
```
<head>
<title>Chap 2 : exercice 3</title>
<meta charset="utf-8"> <!-- permet décrire des accents dans le texte -->
<?php
    $production = true;
    if($production){
        $manifestJson = json_decode(file_get_contents('./.vite/manifest.json'), true);
        ?>
        <script src="<?php echo $manifestJson['src/js/main.js']['file'] ?>" type="module"></script>
        <link type="text/css" rel="stylesheet" href="<?php echo $manifestJson['src/js/main.js']['css'][0] ?>" />
    <?php
    } // fin du if
    else{
        ?>
        <script src="http://localhost:5173/src/js/main.js" type="module"></script>
        <script src="http://localhost:5173/@vite/client" type="module"></script>
    <?php
    } // fin du else
    ?>
</head>
```

1.5 Build (6/9)

- Maintenant qu'on a ça, on peut refaire la commande `npm run build`.
- Lorsque vous voudrez développer en utilisant Vite, vous placerez votre variable `$production` à `false`. Et ça utilisera les `<script>` nécessaires pour vite. Vous utiliserez donc `npm run dev` pour lancer Vite.
Là vous afficherez votre site avec l'URL « public » (cfr slide 8).
- Lorsque vous voudrez déployer en production, vous placerez cette variable `$production` à `true` puis vous ferez votre `npm run build`.
Là vous afficherez votre site avec l'URL « dist » (cfr plus loin) en utilisant juste Apache (de xampp) sans avoir besoin de Vite.

1.5 Build (7/9)

- Si j'essaye d'aller à l'URL « dist », j'ai un souci d'affichage car l'image d'arrière-plan n'est pas visible. Et du coup, mon texte étant en blanc, il n'est pas visible sur un fond blanc.



1.5 Build (8/9)

- Si vous ouvrez le fichier css qui a été généré suite au `npm run build`, et qui se trouve dans « `dist/assets` », vous verrez que le problème vient du fait que l'image en arrière plan n'a pas la bonne URL (l'URL est « `assets/nomGénéréDeImage.jpg` ». L'image est bien dans le dossier « `assets` », mais le fichier css aussi est dans ce même dossier. Donc il y a le « `assets` » en trop dans l'URL. Comme on le voit ici :

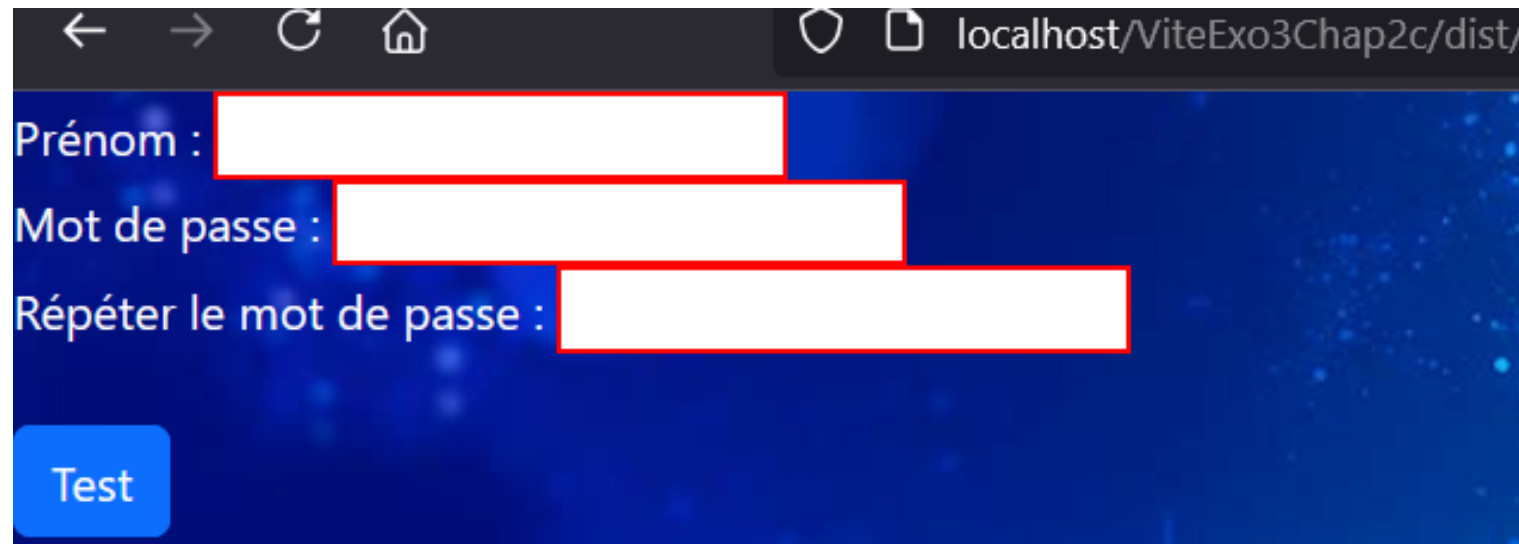
```
url(/assets/background-D_RrpxMZ.jpg)
```

- Corrigeons cette URL pour n'avoir que le nom de l'image :

```
url(background-D_RrpxMZ.jpg)
```

1.5 Build (9/9)

- Maintenant rechargeons notre page et on voit que l'arrière-plan s'affiche correctement :



- Cela fonctionne. Mais il est clair que normalement le fichier généré devrait être correct et on ne devrait pas avoir à le modifier. Cependant je n'ai pas encore trouvé comment ne pas avoir cette erreur. On se contentera donc de ce système D pour l'instant.