

# Fondamentaux du Web

## HTML, CSS, JavaScript

### Lab 0 — Mise en route

#### Préparer son environnement de développement

Licence 1 DUT1-Informatique — 2025–2026

Dr. El Hadji Bassirou TOURÉ  
École Supérieure Polytechnique  
Département Génie Informatique  
Université Cheikh Anta Diop de Dakar

#### Informations pratiques

|                      |                                                                 |
|----------------------|-----------------------------------------------------------------|
| <b>Durée estimée</b> | 1h30 à 2h (à faire chez soi avant la S1)                        |
| <b>Où</b>            | Chez vous, sur votre ordinateur personnel                       |
| <b>Prérequis</b>     | Bases d'utilisation d'un terminal + connaissance minimale HTML  |
| <b>Matériel</b>      | Un ordinateur (Windows, macOS ou Linux) + connexion Internet    |
| <b>Validation</b>    | Un fichier <code>index.html</code> poussé sur votre repo GitHub |
| <b>Rapport</b>       | 1 page : ce que j'ai fait + difficultés + lien repo             |

#### Objectifs du Lab

À la fin de ce Lab, vous aurez :

1. Compris comment les **4 outils** du développeur web (VS Code, navigateur, Git, GitHub) travaillent ensemble
2. Installé **VS Code** et l'extension **Live Server**
3. Installé et configuré **Git** avec votre nom et votre email
4. Créé un compte **GitHub** et votre dépôt **jang-afrig**
5. Cloné ce dépôt sur votre ordinateur
6. Écrit une première page HTML « Hello » et l'avoir affichée avec Live Server
7. Fait votre **premier cycle Git** : `add` → `commit` → `push`
8. Vérifié votre travail sur GitHub
9. Rédigé votre premier rapport de Lab

#### Table des matières

|   |                                   |   |
|---|-----------------------------------|---|
| 1 | Le paysage du développeur web     | 2 |
| 2 | Ce qu'on va installer et pourquoi | 3 |

|           |                                                 |           |
|-----------|-------------------------------------------------|-----------|
| <b>3</b>  | <b>Installer VS Code</b>                        | <b>3</b>  |
| 3.1       | Télécharger et installer . . . . .              | 3         |
| 3.2       | Installer l’extension Live Server . . . . .     | 3         |
| <b>4</b>  | <b>Installer et configurer Git</b>              | <b>5</b>  |
| 4.1       | Installer Git . . . . .                         | 5         |
| 4.2       | Vérifier l’installation . . . . .               | 5         |
| 4.3       | Configurer Git avec votre identité . . . . .    | 5         |
| <b>5</b>  | <b>Créer votre compte GitHub</b>                | <b>6</b>  |
| 5.1       | S’inscrire . . . . .                            | 6         |
| 5.2       | Créer votre dépôt jang-afrig . . . . .          | 6         |
| 5.3       | Récupérer l’URL du dépôt . . . . .              | 6         |
| <b>6</b>  | <b>Cloner le dépôt sur votre ordinateur</b>     | <b>7</b>  |
| 6.1       | Créer un dossier pour vos projets . . . . .     | 7         |
| 6.2       | Cloner le dépôt . . . . .                       | 7         |
| <b>7</b>  | <b>Créer votre première page HTML</b>           | <b>8</b>  |
| 7.1       | Ouvrir le dossier dans VS Code . . . . .        | 8         |
| 7.2       | Créer index.html . . . . .                      | 8         |
| 7.3       | Afficher avec Live Server . . . . .             | 8         |
| <b>8</b>  | <b>Premier cycle Git : du local vers GitHub</b> | <b>9</b>  |
| 8.1       | Ouvrir le terminal dans VS Code . . . . .       | 10        |
| 8.2       | Voir l’état avec git status . . . . .           | 10        |
| 8.3       | Préparer avec git add . . . . .                 | 10        |
| 8.4       | Enregistrer avec git commit . . . . .           | 10        |
| 8.5       | Envoyer sur GitHub avec git push . . . . .      | 10        |
| 8.6       | Vérifier sur GitHub . . . . .                   | 11        |
| <b>9</b>  | <b>Aide-mémoire</b>                             | <b>12</b> |
| 9.1       | Les commandes apprises dans ce Lab . . . . .    | 12        |
| 9.2       | Le cycle Git : la routine du semestre . . . . . | 12        |
| 9.3       | Si quelque chose ne marche pas . . . . .        | 12        |
| <b>10</b> | <b>Rapport à produire</b>                       | <b>13</b> |
| <b>11</b> | <b>Résumé et conclusion</b>                     | <b>13</b> |
| 11.1      | Et après ? . . . . .                            | 13        |

# 1 Le paysage du développeur web

Avant de cliquer quoi que ce soit, prenons 5 minutes pour comprendre **avec quels outils vous allez travailler** tout le semestre. Un développeur web travaille avec **quatre outils** qui se complètent, comme un artisan dans son atelier :

- **Un éditeur de code** — VS Code, votre *table de travail*
- **Un navigateur** — Chrome ou Firefox, votre *vitrine*
- **Un outil de versioning** — Git, votre *carnet de bord*
- **Un hébergeur** — GitHub, votre *entrepôt partagé*

Voici comment ils travaillent ensemble :

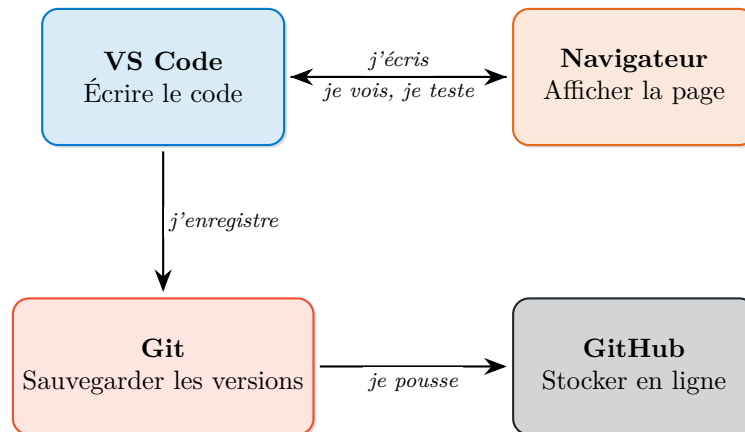


Fig. 1 : Les quatre outils du développeur web et leurs interactions.

## À retenir

- **J'écris** mon code dans VS Code et je le **teste** dans le navigateur
- **J'enregistre** mes versions avec Git sur mon ordinateur
- Je **pousse** mes versions sur GitHub pour les mettre en ligne et les sauvegarder

## 2 Ce qu'on va installer et pourquoi

| Outil              | À quoi il sert                                                                                                                                                                   |
|--------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VS Code</b>     | L'éditeur de code. C'est là que vous écrirez HTML, CSS, JS. Intelligent : il colore les mots-clés, détecte les erreurs, complète automatiquement.                                |
| <b>Live Server</b> | Extension de VS Code. Lance un petit serveur local et <b>recharge automatiquement</b> la page dans le navigateur dès que vous enregistrez le code. Gain de temps énorme.         |
| <b>Git</b>         | Outil de <i>versioning</i> . Enregistre l'historique de vos modifications. Vous pouvez revenir en arrière à n'importe quel moment. C'est un gros bouton « annuler » sans limite. |
| <b>GitHub</b>      | Site web qui héberge vos dépôts Git en ligne. Comme Google Drive, mais pour les développeurs. Vous y déposerez tout votre projet Jàng Afrig.                                     |

### Règle d'or

Ne passez jamais à l'étape suivante tant que l'étape actuelle n'a pas produit le résultat attendu. Si ça bloque, relisez calmement, cherchez la coquille. **Chercher par soi-même fait partie du métier.**

## 3 Installer VS Code

### 3.1 Télécharger et installer

1. Allez sur <https://code.visualstudio.com/>
2. Cliquez sur le bouton **Download** correspondant à votre système (Windows, macOS, Linux)
3. Lancez le fichier téléchargé et suivez l'assistant
4. Gardez les options par défaut ; sur Windows, **cochez « Ajouter à PATH »**

### Astuce VS Code

VS Code permet de faire énormément de choses via la palette de commandes : **Ctrl+Shift+P** (Windows/Linux) ou **Cmd+Shift+P** (macOS). On s'en servira plusieurs fois dans le semestre.

### 3.2 Installer l'extension Live Server

1. Ouvrez VS Code
2. Cliquez sur l'icône **Extensions** dans la barre de gauche (4 carrés)
3. Dans la barre de recherche, tapez : **Live Server**
4. Choisissez celui de **Ritwick Dey** (le plus téléchargé)
5. Cliquez sur **Install**

### Pourquoi Live Server ?

Sans Live Server, à chaque modification de code, vous devez : enregistrer → aller dans le navigateur → rafraîchir la page. Avec Live Server : vous enregistrez, la page se rafraîchit **toute seule**. Sur 30h de cours, ce petit détail vous économise des heures.

## 4 Installer et configurer Git

### 4.1 Installer Git

#### Windows

1. Allez sur <https://git-scm.com/download/win>
2. Le téléchargement démarre automatiquement
3. Lancez le fichier, cliquez **Next** partout sauf :
4. Quand on vous demande l'éditeur par défaut, choisissez « **Use Visual Studio Code** »

#### macOS

```
git --version
```

Si Git n'est pas installé, macOS vous proposera de l'installer (acceptez).

#### Linux (Ubuntu/Debian)

```
sudo apt update  
sudo apt install git
```

### 4.2 Vérifier l'installation

Ouvrez votre terminal (Git Bash sous Windows, Terminal sous macOS/Linux) et tapez :

```
git --version
```

Résultat attendu :

```
git version 2.47.1
```

✓ Si vous voyez un numéro de version, Git est installé.

### 4.3 Configurer Git avec votre identité

Git a besoin de savoir **qui vous êtes**. Chaque enregistrement portera votre nom et votre email. Tapez ces deux commandes (avec **votre** vrai nom et **votre** email étudiant) :

```
git config --global user.name "Fatou Diop"  
git config --global user.email "fatou.diop@esp.sn"
```

#### Utilisez votre vrai nom et email

Remplacez par votre vrai nom (pas un pseudo) et votre email étudiant. Ces informations apparaîtront dans l'historique Git et servent à l'évaluation.

Vérifiez :

```
git config --global user.name  
git config --global user.email
```

Vous devez voir votre nom et email s'afficher. ✓

## 5 Créer votre compte GitHub

### 5.1 S'inscrire

1. Allez sur <https://github.com/>
2. Cliquez sur **Sign up**
3. Utilisez le même email que pour Git
4. Choisissez un mot de passe solide
5. Choisissez un **nom d'utilisateur** (*username*)
6. Complétez l'inscription et vérifiez votre email

#### Choisissez un username professionnel

Votre *username* GitHub apparaîtra sur votre CV pendant toute votre carrière. Évitez :

- fatoudupays221 ✗ (trop informel)
- xxx\_killer\_xxx ✗ (à ne pas montrer à un recruteur)

Préférez :

- fatou-diop ✓
- fdiop ✓
- fatou.diop ✓

### 5.2 Créer votre dépôt jang-afrig

1. Connectez-vous sur GitHub
2. Cliquez sur le + en haut à droite → **New repository**
3. **Repository name** : jang-afrig
4. **Description** : Quiz géographique sur les 54 pays africains
5. Laissez **Public**
6. Cochez « **Add a README file** »
7. Cliquez **Create repository**

### 5.3 Récupérer l'URL du dépôt

Sur la page de votre dépôt, cliquez sur le bouton vert **Code**, onglet **HTTPS**, copiez l'URL qui finit par `.git`. Elle ressemble à :

```
https://github.com/votre-username/jang-afrig.git
```

Gardez-la de côté, on en a besoin à l'étape suivante.

## 6 Cloner le dépôt sur votre ordinateur

**Cloner** un dépôt, c'est faire une copie locale sur votre ordinateur. Vous pourrez éditer les fichiers chez vous, puis renvoyer vos modifications sur GitHub.

### 6.1 Créer un dossier pour vos projets

Dans le terminal, allez dans votre dossier personnel et créez un dossier dédié à ce cours :

```
cd ~  
mkdir cours-web  
cd cours-web
```

### 6.2 Cloner le dépôt

Utilisez la commande `git clone` avec votre URL (en remplaçant `votre-username`) :

```
git clone https://github.com/votre-username/jang-afrig.git
```

Résultat attendu :

```
Cloning into 'jang-afrig'...  
remote: Enumerating objects: 3, done.  
Receiving objects: 100% (3/3), done.
```

Un dossier `jang-afrig` a été créé. Entrez dedans :

```
cd jang-afrig  
ls
```

Vous devez voir au moins un fichier `README.md`.

## 7 Créer votre première page HTML

### 7.1 Ouvrir le dossier dans VS Code

Dans le terminal, toujours dans le dossier `jàng-afrig`, tapez :

```
code .
```

**Attention au point.** Cette commande ouvre le dossier courant dans VS Code.

#### Si `code .` ne marche pas sur macOS

Dans VS Code, `Cmd+Shift+P` → tapez « shell command » → « Install 'code' command in PATH ». Sinon, ouvrez VS Code et faites **File** → **Open Folder**.

### 7.2 Créer `index.html`

1. Dans l'Explorateur de VS Code (barre de gauche), cliquez sur l'icône « Nouveau fichier » à côté de **NJURELGEO**
2. Tapez `index.html`
3. Copiez-collez ce code (remplacez « Fatou » par **votre prénom**) :

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <meta charset="UTF-8">
  <title>Hello Fatou</title>
</head>
<body>
  <h1>Hello Fatou !</h1>
  <p>Bienvenue dans mon premier projet web : Jàng Afrig.</p>
</body>
</html>
```

Enregistrez avec `Ctrl+S` ou `Cmd+S`.

### 7.3 Afficher avec Live Server

1. Clic droit sur `index.html` dans l'Explorateur
2. **Open with Live Server**  
Votre navigateur s'ouvre sur la page. ✓

#### Testez le rechargement automatique

Dans VS Code, changez « Hello Fatou » en « Salaam Aleykum Fatou » et enregistrez. La page se met à jour **toute seule** dans le navigateur. C'est ça la magie de Live Server.

## 8 Premier cycle Git : du local vers GitHub

Vous avez créé un nouveau fichier sur votre ordinateur. Il faut maintenant l'**enregistrer dans l'historique Git**, puis l'**envoyer sur GitHub**. C'est ce qu'on appelle un *cycle Git*.

Le schéma ci-dessous montre le flux à comprendre :

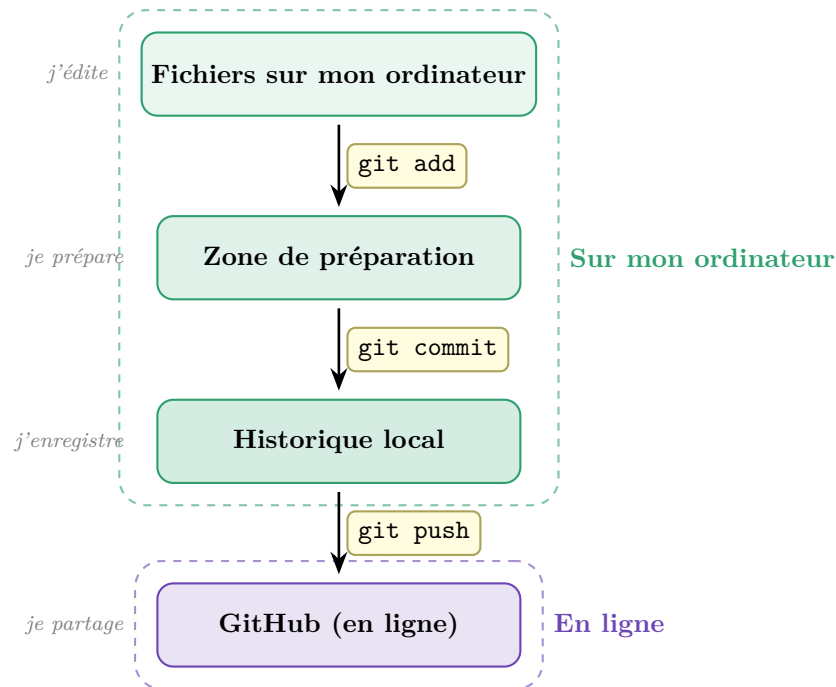


Fig. 2 : Le flux d'un cycle Git : du fichier local jusqu'à GitHub.

### L'analogie du colis

- Vous avez un cadeau chez vous (votre **fichier**).
- Vous le mettez dans un carton (`git add` → zone de préparation).
- Vous fermez le carton et lui mettez une étiquette avec un message (`git commit`).
- Vous donnez le carton à La Poste pour qu'il parte à l'étranger (`git push` → GitHub).

Chaque carton (*commit*) reste dans l'historique. Vous pouvez toujours le retrouver, même 6 mois plus tard.

## 8.1 Ouvrir le terminal dans VS Code

VS Code a un terminal intégré. Pour l'ouvrir :

- Menu **Terminal** → **New Terminal**
- Raccourci : **Ctrl+`** / **Cmd+`**

Le terminal s'ouvre en bas, déjà placé dans `jàng-afrig`.

## 8.2 Voir l'état avec `git status`

```
git status
```

Résultat :

```
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    index.html

nothing added to commit but untracked files present
```

Git dit : « Je vois un nouveau fichier, mais je ne le surveille pas encore. »

## 8.3 Préparer avec `git add`

```
git add index.html
```

Pas de message : Git est silencieux quand tout va bien. Vérifiez :

```
git status
```

```
Changes to be committed:
  new file:   index.html
```

Le fichier est **préparé**.

## 8.4 Enregistrer avec `git commit`

```
git commit -m "v0 : premier setup avec hello world"
```

Le texte entre guillemets est un **message de commit**. Décrivez brièvement ce que vous venez de faire.

```
[main a1b2c3d] v0 : premier setup avec hello world
1 file changed, 10 insertions(+)
```

✓ Commit enregistré en local.

## 8.5 Envoyer sur GitHub avec `git push`

```
git push
```

La première fois, Git vous demandera votre **username** GitHub et un **token**.

**GitHub : utilisez un Personal Access Token (PAT)**

GitHub n'accepte plus les mots de passe depuis 2021. À la place, vous avez besoin d'un **Personal Access Token (PAT)** :

1. GitHub → Settings → Developer settings → Personal access tokens → Tokens (classic)
  2. **Generate new token (classic)**
  3. Nom : « cours-web-2026 »
  4. Cochez la case **repo**
  5. **Generate** → **COPIEZ LE TOKEN TOUT DE SUITE** — vous ne le reverrez plus
  6. Utilisez ce token à la place du mot de passe
- Notez-le dans un endroit sûr (gestionnaire de mots de passe).

Résultat attendu :

```
Enumerating objects: 4, done.  
Writing objects: 100% (3/3), 394 bytes  
To https://github.com/votre-username/jang-afrig.git  
1a2b3c4..a1b2c3d main -> main
```

✓ Votre code est sur GitHub.

## 8.6 Vérifier sur GitHub

Allez sur <https://github.com/votre-username/jang-afrig>. Vous devez voir `index.html` dans la liste. Cliquez dessus pour voir son contenu.

## 9 Aide-mémoire

### 9.1 Les commandes apprises dans ce Lab

| Commande                                          | Ce qu'elle fait                       |
|---------------------------------------------------|---------------------------------------|
| <code>git --version</code>                        | Vérifier que Git est installé         |
| <code>git config --global user.name "..."</code>  | Définir votre nom Git                 |
| <code>git config --global user.email "..."</code> | Définir votre email Git               |
| <code>git clone &lt;url&gt;</code>                | Télécharger un dépôt depuis GitHub    |
| <code>git status</code>                           | Voir l'état des fichiers              |
| <code>git add &lt;fichier&gt;</code>              | Préparer un fichier                   |
| <code>git commit -m "message"</code>              | Enregistrer les fichiers préparés     |
| <code>git push</code>                             | Envoyer sur GitHub                    |
| <code>code .</code>                               | Ouvrir le dossier actuel dans VS Code |

### 9.2 Le cycle Git : la routine du semestre

1. Modifier des fichiers dans VS Code
2. `git status` (voir ce qui a changé)
3. `git add .` (préparer tous les changements)
4. `git commit -m "message"` (enregistrer en local)
5. `git push` (envoyer sur GitHub)

#### À retenir absolument

Ces 5 étapes seront votre routine à la fin de chaque Lab pendant tout le semestre. Vous allez les faire 10 fois. Au bout de 3 ou 4, vous les ferez sans réfléchir.

### 9.3 Si quelque chose ne marche pas

#### Méthode de déblocage

1. Relisez votre commande — une coquille de lettre bloque tout
2. Copiez le message d'erreur exact dans un moteur de recherche
3. Demandez à un camarade qui a réussi
4. Notez le problème et montrez-le au prof au début de la S1

## 10 Rapport à produire

### Rapport du Lab 0 — obligatoire

À la fin de ce Lab, vous produisez un **rapport court** de **1 à 2 pages maximum**. Ce sera la routine pour chaque Lab du semestre.

#### Format du fichier :

- PDF, Markdown (.md) ou Word
- Nom : `rapport-lab-0-NOM-Prenom.pdf`  
(ex. `rapport-lab-0-DIOP-Fatou.pdf`)
- Déposer dans un sous-dossier `rapports/` de votre repo GitHub

#### Contenu obligatoire (4 sections) :

1. **Ce que j'ai fait** (½ page) : outils installés, configuration Git, création du repo, premier cycle Git.
2. **Difficultés rencontrées** (¼ page) : ce qui m'a bloqué, messages d'erreur rencontrés.
3. **Comment j'ai résolu ces difficultés** (¼ page) : recherche, aide, essais.
4. **Lien vers mon repo GitHub** :  
ex. <https://github.com/fatou-diop/jang-afrig>

**Ton attendu** : honnête et direct. Si quelque chose n'a pas marché du premier coup, dites-le. L'erreur fait partie de l'apprentissage et **personne ne vous jugera dessus**. Ce qui compte, c'est que vous ayez réfléchi à ce que vous avez fait.

## 11 Résumé et conclusion

Dans ce Lab, vous avez :

1. Compris le rôle des 4 outils et leurs interactions ✓
2. Installé VS Code et Live Server ✓
3. Installé et configuré Git avec votre identité ✓
4. Créé un compte GitHub et votre premier dépôt ✓
5. Cloné ce dépôt sur votre ordinateur ✓
6. Écrit votre première page HTML avec Live Server ✓
7. Fait votre premier cycle Git : `add` → `commit` → `push` ✓
8. Rédigé votre premier rapport de Lab ✓

**Votre environnement de développement est prêt.** Vous êtes paré(e) pour la S1.

### 11.1 Et après ?

Au CM1 (S1), nous allons :

- Faire le tour de ce qu'est vraiment une page web
- Découvrir Jàng Afrig dans le détail
- Approfondir les balises HTML
- Commencer à construire la page d'accueil

#### Votre vision à 15 séances

Votre dépôt `jang-afrig` contient aujourd'hui une simple page « Hello ». Dans 15 séances, il contiendra une vraie application de quiz géographique, en ligne, accessible depuis votre téléphone, utilisant les données de 54 pays africains récupérées sur une API.

Vous pourrez la montrer à votre famille via un simple lien.

## **Lab 0 — Terminé ✓**

Votre atelier est prêt. Bienvenue dans le monde du développement web.

*Rendez-vous au CM1 !*