

Les Frameworks

Vincent Baylly

Présentation du plan de cours



Intégration Simple sans outil de développement

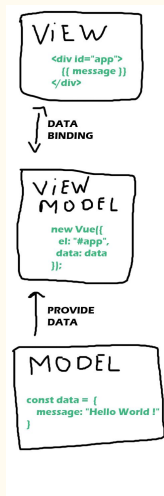
Création d'une page simple avec un composant Vue JS:

- Connaître les éléments nécessaires au fonctionnement du framework
- Comprendre les différents composants
- Reprendre les acquis du web avec VueJS

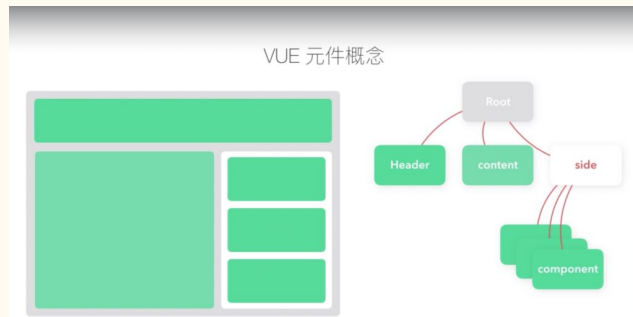
```
<!DOCTYPE html>
<html>
  <head>
    <title>Mon Site Web</title>
    <script src="https://cdn.jsdelivr.net/npm/vue/dist/vue.js"></script>
  </head>
  <body>
    <div id="app">{{ message }}</div>
  </body>
</html>
<script>
  var app = new Vue({
    el: "#app",
    data: {
      message: "Hello Vue !",
    },
  });
</script>
```

Introduction à vueJS

- L'instance
- Le cycle de vie d'une instance
- Les propriétés
- Les propriétés combinées
- Les directives

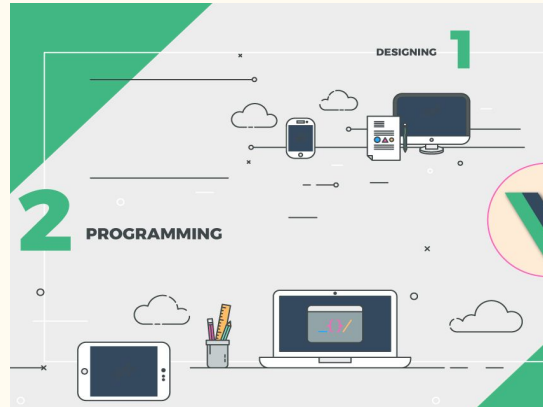


- Les filtres
- Les événements
- Les composants
- Les composants dynamiques
- Les échanges de données
- Les transitions



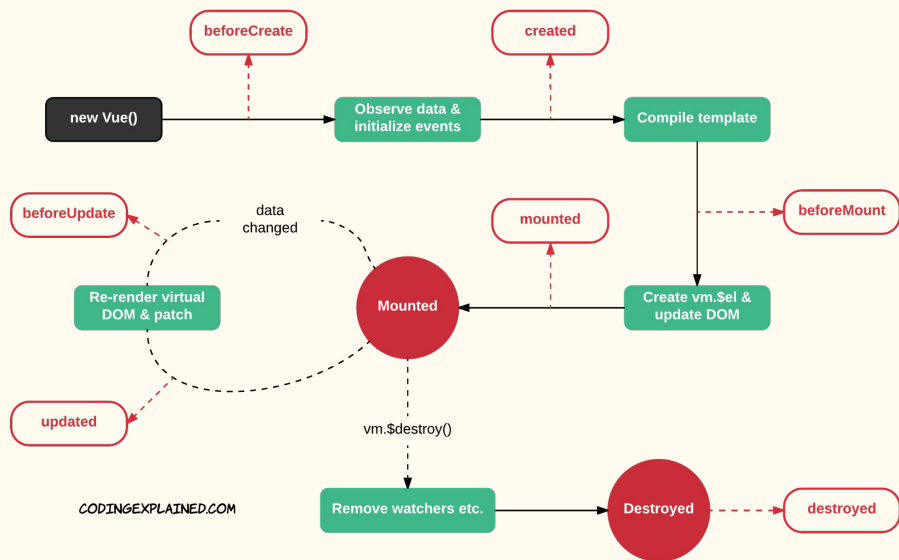
Intégration de vuejs dans vos sites

- Prise en main et concepts importants liés aux interfaces riches
- Affichage des variables via le HTML
- Dynamisation des rendus
- Ajouter des éléments HTML
- JS dans le DOM
- Manipulation des formulaires
- Les inputs et les variables
- Gestion des événements



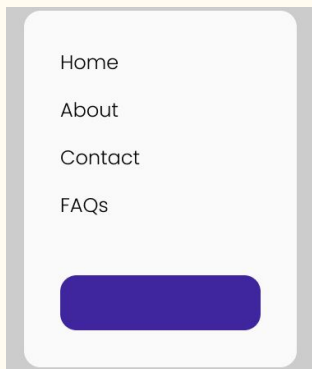
Cycle de vie et flux du framework

- La gestion des transitions vue.js
- Les composants et leurs cycles de vie
- L'instance de l'application
- Le templating avec VueJS
- La création



Navigation dans le framework

- La syntaxe des templates
- Le « binding »
- Mixins locaux et globaux
- Plugins
- Directives personnalisées
- Support de TypeScript dans Vue-Cli
- Création d'une application SPA avec vue-router
- Allure du lien actif
- Indicateurs de navigation globaux
- Indicateurs de navigation par route
- Indicateurs de navigation par composant
- Routage
- Transition



Documentation et Correction du code

- Structure du code
- Documenter le code
- Test et débogage du code
- Produire une grille de validation
- Déployer la version bêta du produit



Test d'une solution

- Tester le produit selon la grille de validation
- Tester la version bêta par des personnes tierces
- Analyser les résultats des tests utilisateurs
- Rédiger un rapport de test
- Rendre son code robuste
- Corriger les erreurs



Préparer une livraison

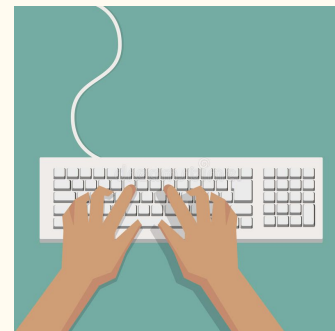
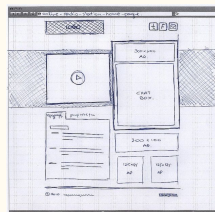
- Optimiser son code
- Préparer la version finale à publier
- Publier le produit final



Références



- [digitalocean](#)
- [vuejs.org](#)
- [router.vuejs.org](#)
- [tutorialspoint](#)
- [openclassrooms](#)
- [udemy](#)



Qu'est ce qu'un
Framework ?

—

Définition

Un framework est un utilitaire qui permet d'avoir accès à des outils ou plus particulièrement à du code déjà écrit.

Pour faire une correspondance on pourrait voir un framework comme un lieu de travail (une usine ou un atelier). Celui-ci contiendrait déjà tous les outils nécessaires pour réaliser un produit.

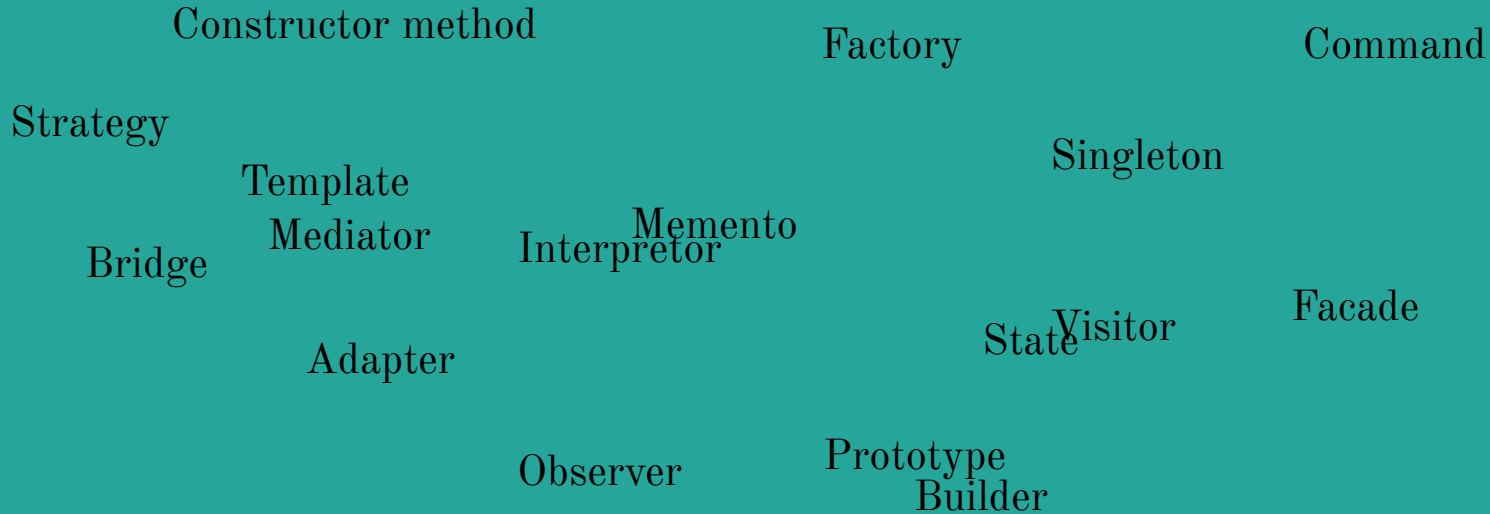


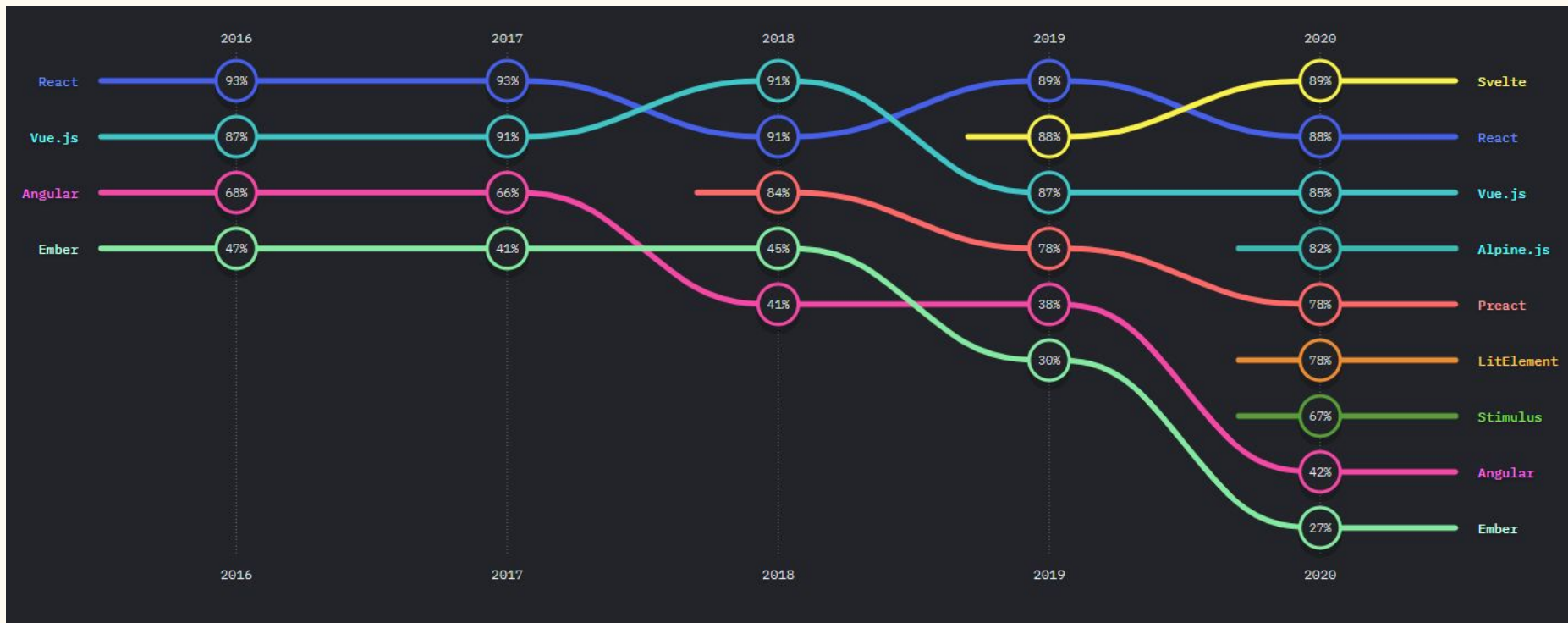
Composition d'un Framework

- Un langage:
 - Il sera défini dans un langage bien précis
 - Il sera recommandé dans l'utilisation d'un contexte particulier (Frontend, Backend, Echange avec une base de données...)
- Une Architecture:
 - La définition du framework mettra en place une architecture particulière. Une façon de concevoir le rendu ainsi qu'une implémentation du code (POO, Procédural, multithreading, synchrone/asynchrone, requetage...)
 - Utilisera des standards de développement (patron de conception, recommandation de sécurité)

Patron de Conception ?

Qui peut dire ce qu'est un Patron de conception et à quoi il sert ?





State of Javascript - Framework frontend

Quiz!

- Qu'est ce qu'un framework ?
 - Des informations déjà établies, telles que du code HTML et CSS prédéfinis, qui peuvent être référencées pour créer des pages Web.
 - Une structure qui maintient des éléments ensemble
 - Du Javascript ou du JQuery déjà codé destiné à être implémenté dans un site Web
 - Des fragments de code destinés à être assemblés dans un certain ordre.
- Il y a 2 types de framework, Frontend et backend. Vrai ou Faux
- Quel sont les avantages d'utiliser Bootstrap ?
 - La facilité d'utilisation
 - Une conception reactif/adaptatif (responsive design)
 - Compatibilité avec un cellulaire
 - Toutes ces réponses
- Nommer 3 Frameworks

Création d'un jeu en Javascript

—

Shifumi!

Nous allons dans un fichier html créer le jeu roche, papier, ciseaux avec les règles suivantes:

Choisissez, roche, papier ou ciseaux. - la roche bat les ciseaux, mais se fait battre par le papier
- le papier bat la roche, mais se fait battre par des ciseaux - ciseaux bat le papier, mais est brisé par la roche.



Shifumi!

Dans un nouveau répertoire appelé Shifumi, ouvrir un nouveau fichier

On peut utiliser le raccourci `html:5` dans Visual Studio Code pour créer notre gabarit de page `html`.

```
<!DOCTYPE html>
<html lang="fr">
  <head>
    <title>Roche Papier Ciseaux</title>
  </head>
  <body></body>
</html>
```

Shifumi!

On ajoute 3 boutons pour chacun des choix du jeu et une division qui affichera le résultat.
Qui a gagné entre l'ordinateur ou l'utilisateur!

```
<button>Roche</button>  
<button>Papier</button>  
<button>Ciseaux</button>  
<div class="result"></div>
```

Nous pouvons ajouter la gestion de nos boutons dans les balises `<script>`

```
<script>  
  const buttons = document.querySelectorAll("button");  
  console.log(buttons);  
</script>
```

Shifumi!

On ajoute l'évènement sur chacun des boutons:

```
for (let i = 0; i < buttons.length; i++) {  
    buttons[i].addEventListener("click", function () {  
        console.log("Fonctionne");  
    });  
}
```

Nous pouvons conserver la valeur du bouton sélectionné pour connaître le choix de l'utilisateur.

Pour cela on remplace notre `console.log` par la stockage du choix dans une variable joueur comme ceci:

```
const playerChoice = buttons[i].innerHTML;
```

Shifumi!

Que faire pour définir aléatoirement le choix de notre joueur robot ?!

Comment choisir un chiffre au hasard ?

Comment transformer ce chiffre pour que sa valeur soit 0, 1 ou 2 ?

Comment choisir parmi les valeurs possibles ?

On stockera cette information dans une variable

```
const robotChoice=buttons[Math.floor(Math.random() * buttons.length)].innerHTML;
```

```
console.log(`Joueur ${playerChoice} VS Robot ${robotChoice} `);
```

Shifumi!

On peut maintenant
ajouter les règles du
jeu:

```
let result = "";

//Logique du jeu
if (playerChoice === robotChoice) {
    result = "Egalité";
} else if (
    //La roche bat ciseaux
    (playerChoice === "Roche" && robotChoice === "Ciseaux") ||
    //La roche se fait battre par le papier
    (playerChoice === "Papier" && robotChoice === "Roche") ||
    //le papier se fait battre par des ciseaux
    (playerChoice === "Ciseaux" && robotChoice === "Papier")
) {
    result = "Gagné";
} else {
    result = "Perdu";
}
```


Shifumi!

Affiche le résultat à
l'utilisateur

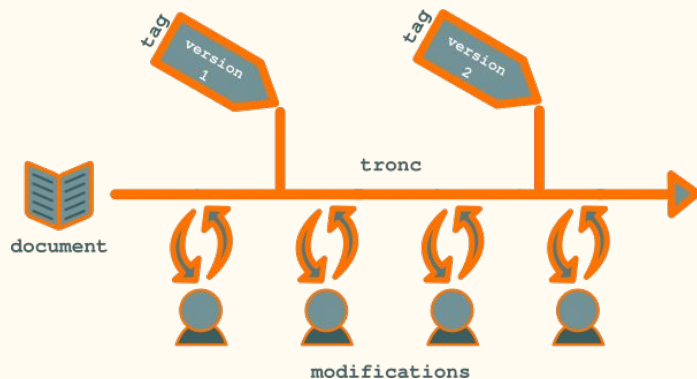
```
document.querySelector(".result").innerHTML = `  
  Joueur: ${playerChoice} <br/>  
  Robot: ${robotChoice} <br/>  
  ${result} !  
`;  
;
```

Mise en place d'un
gestionnaire de version

—

Gestionnaire de version

Nous avons fait notre application pas à pas mais une fois terminé, il est impossible de revoir la démarche d'implémentation de notre application. Que ce soit pour nous ou pour un autre développeur qui souhaite comprendre notre code. Il serait également plus facile de partager notre code et de pouvoir le comparer.

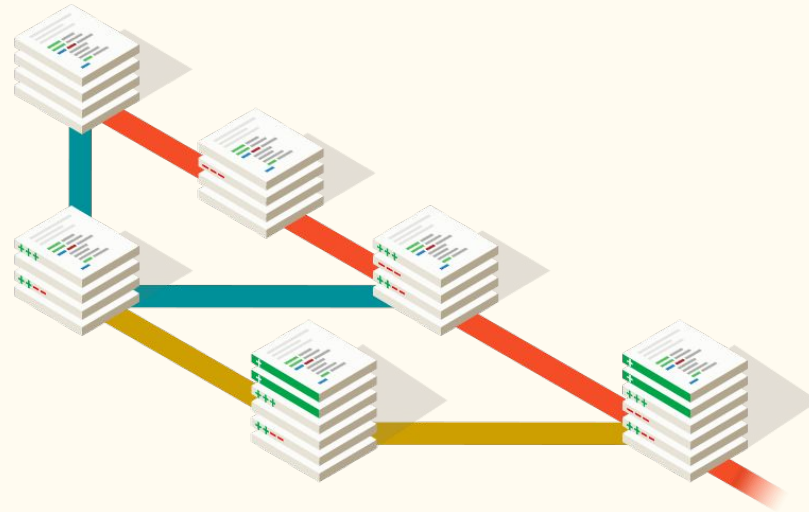


Git



Sur windows,télécharger l'utilitaire pour utiliser les commandes Git:

<https://git-scm.com/download/win>



GitHub



Nous allons maintenant nous créer un compte dans GitHub pour partager notre code.

Allez sur l'adresse:

<https://github.com/join>

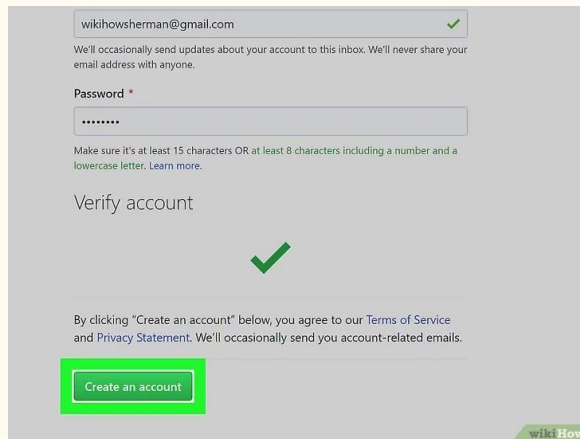
Saisissez vos informations personnelles

Cliquez sur le bouton vert:Créer un compte.

Nous utiliserons le plan gratuit dans un premier temps pour avoir accès au serveur d'hébergement.

Vous devez faire vérifier votre courriel maintenant.

Allez dans la boîte de courriel que vous avez utilisée et cliquez sur valider dans le courriel reçu



The screenshot shows the GitHub account creation page. At the top, there is a text input field containing the email address 'wikihowsherman@gmail.com' with a green checkmark to its right. Below this, a small line of text states: 'We'll occasionally send updates about your account to this inbox. We'll never share your email address with anyone.' This is followed by a 'Password *' label and a password input field with masked characters '.....'. A note below the password field says: 'Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter. Learn more.' The next section is titled 'Verify account' and features a large green checkmark. At the bottom, a line of text reads: 'By clicking "Create an account" below, you agree to our Terms of Service and Privacy Statement. We'll occasionally send you account-related emails.' Below this text is a green button with the text 'Create an account'. The 'wikiHow' logo is visible in the bottom right corner of the form area.

Reprise des étapes

Nous allons maintenant reprendre le jeu que nous avons codé et que nous allons versionner pour garder des états importants (une livraison, une version, un changement de framework ou d'utilitaire).

On doit dans un premier temps créer ce que l'on appelle le repository: Répertoire contenant les différentes versions de nos fichiers de code.

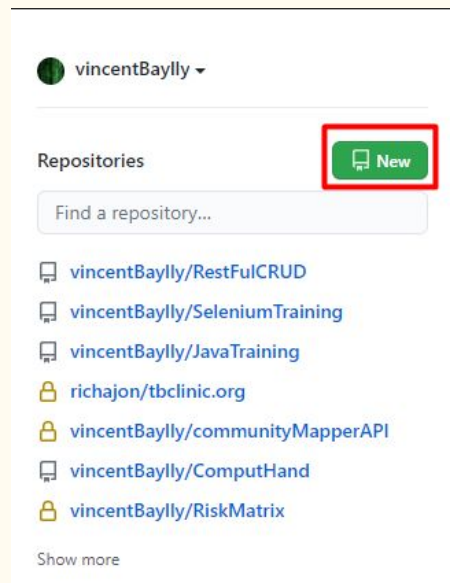
Ce répertoire sera répliqué à distance pour être accessible par plusieurs personnes en tout temps.

GitHub



Nous allons voir maintenant comment créer un nouveau dépôt (repository) dans GitHub pour enregistrer votre code sur le serveur de GitHub.

Une fois connecté sur votre compte gitHub, vous pouvez cliquer sur nouveau repository.



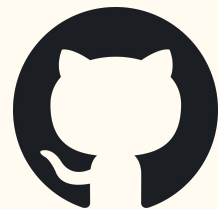
GitHub

Ajoutez les informations nécessaires pour créer votre repository.

Vous pouvez choisir si votre repository sera visible ou non par les autres utilisateurs.

Nous allons voir les autres points:

- README
- .gitignore
- Licence




Create a new repository

A repository contains all project files, including the revision history. Already have a repository? [Import a repository.](#)

Owner *


Repository name *

 vincentBaylly ▾


 /

Great repository names are short and memorable. Need inspiration? How about...

Description (optional)

☒  Public

Anyone on the internet can see this repository. You choose who can commit.

☐  Private

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ Add a README file

This is where you can write a long description for your project. [Learn more.](#)

☐ Add .gitignore

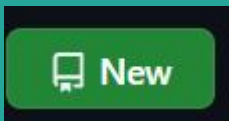
Choose which files not to track from a list of templates. [Learn more.](#)

☐ Choose a license

A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

Aller sur son compte github et créer un repository en cliquant sur le bouton:




Le “répertoire” repository distant vient d'être créé.

Il faut le lier à un repertoire sur notre machine pour pouvoir synchroniser et livrer du code sur le répertoire distant.

Owner *

Repository name *

 vincentBaylly ▾


 /

shifumi ✓


Great repository names are short and memorable. Need inspiration? How

Description (optional)

Jeu de Roche, Papier, Ciseaux

☒  **Public**

Anyone on the internet can see this repository. You choose who can comm

☐  **Private**

You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**

Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

On doit ajouter les outils (fichiers) nécessaires au bon fonctionnement de la gestion de notre répertoire local avec la commande:

```
git init
```

Placez vous donc dans le répertoire où vous avez mis le fichier index.html.
Idéalement renommez le répertoire qui contient shifumi afin de garder le même nom que le “répertoire” repository que l’on a créé sur github.

Git



- README

Ce fichier est un fichier qui résumera votre application. Il utilise les notations Markdown que vous pourrez trouver [ici](#)

```
# SeleniumTraining

Selenium test exercice for Selenium Training

## Installation

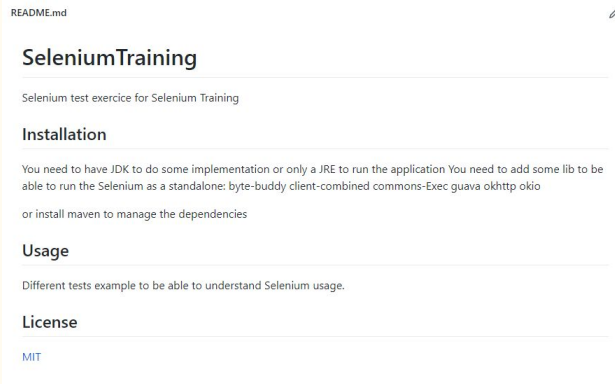
You need to have JDK to do some implementation or only a JRE to run the application
You need to add some lib to be able to run the Selenium as a standalone:
byte-buddy
client-combined
commons-Exec
guava
okhttp
okio

or install maven to manage the dependencies

## Usage

Different tests example to be able to understand Selenium usage.

## license
[MIT](https://choosealicense.com/licenses/mit/)
```



Créez ce fichier README.md dans le répertoire shifumi et ajoutez le dans la liste des fichiers à gérer par git.

Le fichier peut contenir ces informations de base:

```
# Shifumi
Jeu de Roche Papier Ciseaux en Javascript.

## Installation
Lancer le fichier index.html pour executer le code html et javascript
de l'application

## License
[Apache 2.0 License] (https://img.shields.io/crates/l/:crate.svg?style=flat)
```

On l'ajoutera dans la liste git avec la commande:

```
git add README.md
git commit -m "first commit"
```

Git



- .gitignore

Nous ne souhaitons pas forcément livrer tous nos fichiers sur le dépôt.

Par exemple, des fichiers de configuration de votre IDE, des fichiers contenant des informations sur les mots de passe de connexion à une Base de données, etc

Nous allons donc utiliser un fichier (.gitignore) pour dire que nous ne voulons pas versionner ces fichiers.

Git

- .gitignore

Exemple de fichier
pour des applications frontend:

```
.DS_Store
node_modules/
/dist/

# local env files
.env.local
.env.*.local

# Log files
npm-debug.log*
yarn-debug.log*
yarn-error.log*

# Editor directories and files
.idea
.vscode
*.suo
*.ntvs*
*.njsproj
*.sln
*.sw*
```



Il faut paramétrer l'adresse du répertoire distant:

```
git remote add origin https://github.com/[nomdevotrecompte]/shifumi.git
```

Nous pouvons maintenant livrer le fichier sur le repository distant:

```
git push -u origin main
```

Ou pourra seulement faire un

```
git push
```

pour les prochaines livraisons

Git



- LICENCE

Les référentiels publics sur GitHub sont souvent utilisés pour partager des logiciels open source. Pour que votre référentiel soit véritablement open source, vous devrez le concéder sous licence afin que les autres soient libres d'utiliser, de modifier et de distribuer le logiciel.

Vous pouvez utiliser le site <https://choosealicense.com/> pour connaître la licence la plus appropriée pour vos applications. Cela n'a pas d'importance pour mettre du code de pratique dans un repository