

HTTP le protocole du Web, correction

Première NSI Lycée du Parc

Table des matières

1 Le Web	1
1.1 Les trois piliers du Web : HTTP, URL et HTML	1
1.2 Passage de paramètres dans une URL	9
2 Formulaire et passage de paramètres	11
2.1 Un premier exemple	11
2.2 Méthodes de passage des paramètres : GET ou POST	16
2.3 Eléments de formulaire	16

1 Le Web

1.1 Les trois piliers du Web : HTTP, URL et HTML



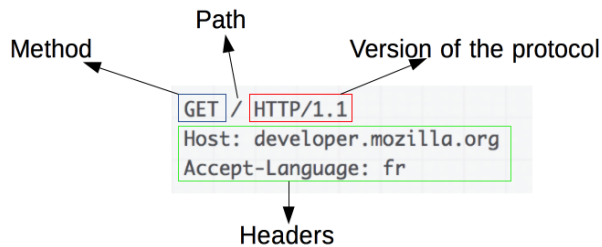
Exercice 1

QCM de type E3C2.

Pour répondre à la première question, relisez la [synthèse de cours sur HTML/CSS](#).

1. Quelle est le code HTML permettant de créer un lien hypertexte dans un document écrit en **HTML** ?
 - Réponse A : `<a>http://tip-top.fr =`
 - Réponse B : `Site du TIP-TOP ==> BONNE RÉPONSE`
 - Réponse C : `<a name="http://tip.top.fr`
2. Comment s'appelle le service qui permet de faire le lien entre une IP et un nom de domaine ?
 - Réponse A : DNS ==> **BONNE RÉPONSE**
 - Réponse B : ARP
 - Réponse C : HTTP
 - Réponse D : Internet

Exercice 2



Source : <https://developer.mozilla.org/fr/docs/Web/HTTP/Aper%C3%A7u>

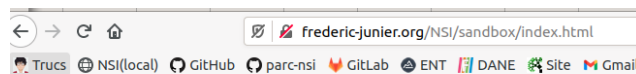
1. Avec un navigateur Web, demander la page d'adresse

<http://frederic-junier.org/NSI/sandbox/index.html>.

Ouvrir la barre d'outils de développement en appuyant sur la touche de fonction F12 et sélectionner l'onglet Réseau. On peut voir les entêtes de la requête et de la réponse HTTP.

- Que représente le code d'état de la réponse HTTP ?

*Réponse : Le code 200 indique que le serveur a pu envoyer la page demandée avec succès. On note qu'il s'agit d'une requête avec la méthode **GET***



HTTP le protocole du Web.

Le Web repose sur trois piliers :

1. Un protocole d'échange de fichiers : HTTP
2. Un système d'adressage de ressources : les URL
3. Un langage de description de contenu : HTML

Lien vers la [première page Web de l'histoire](#)

État	Méthode	Domaine	Fichier
200	GET	frederic-junier.org	index.html
	GET	frederic-junier.org	favicon.ico

- Quelles informations sur le client sont transmises au serveur dans l'entête de la requête ?

*Réponse : Le client transmet plusieurs paramètres dans sa requête : La méthode du protocole **HTTP** qui est utilisée, ici **GET** qui permet de demander une ressource La version du protocole **HTTP** ici 1.1, c'est important de se mettre d'accord sur les règles du dialogue lorsqu'on communique ! **Accept** précise les types de contenus que le client pourra interpréter **Accept-Language** précise la variante de la locale **Connection** précise si le client souhaite une connexion persistante après l'échange requête/réponse avec le serveur : on garde la même connexion*

TCP Cache-Control et If-Modified-Since contiennent des directives pour la mise en cache ou l'utilisation du cache du client User-Agent contient différentes informations sur le navigateur et le système d'exploitation du client

The screenshot shows the 'En-têtes' (Headers) tab in a web browser's developer tools. The selected request is 'GET http://frederic-junier.org/NSI/sandbox/index.html'. The response status is '200 OK'. The response headers section is expanded, showing various headers like 'Accept-Ranges: bytes', 'Content-Type: text/html', 'Date: Wed, 17 Mar 2021 20:01:12 GMT', 'Last-Modified: Tue, 21 Apr 2020 13:26:14 GMT', 'Server: Apache', and 'Vary: Accept-Encoding'. The request headers section is also expanded, showing 'Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8', 'Cache-Control: max-age=0', 'Connection: keep-alive', 'Host: frederic-junier.org', 'If-Modified-Since: Tue, 21 Apr 2020 13:26:14 GMT', 'User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:86.0) Gecko/20100101 Firefox/86.0', and others.

- Quelles informations sur le serveur sont transmises au client dans l'entête de la réponse ?

*Réponse : Le serveur transmet plusieurs paramètres dans sa réponse : **Content-Length** précise la taille en octets de la réponse **Content-Type** précise le type de contenu de la réponse, ici du **HTML** **Date** contient un horodatage de la réponse **Last-Modified** contient un horodatage de la dernière modification de la ressource **Server** précise le type de logiciel serveur, ici **Apache***

- Effectuer une nouvelle requête avec l'URL <http://frederic-junier.org/NSI/sandbox/>. Quelle différence avec la requête initiale ?

Réponse : La page renvoyée est la même mais avec le code de réponse de redirection 304

Not Modified indique qu'il n'y a pas besoin de retransmettre les ressources demandées. C'est une redirection implicite vers une ressource mise en cache. Cela survient lorsque la méthode de requête est safe (par exemple une requête GET ou HEAD), ou lorsque la requête est conditionnelle et utilise l'en-tête If-None-Match ou If-Modified-Since. Noter que si le chemin de la ressource dans l'URL ne se termine pas par un fichier alors le serveur Web renvoie le fichier par défaut *index.html* s'il existe

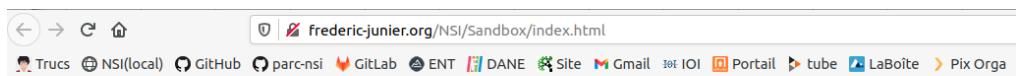
- Effectuer une nouvelle requête avec l'URL <https://frederic-junier.org/NSI/sandbox/>. Quelle différence avec la requête initiale peut-on observer dans la barre d'adresse du navigateur ?

Réponse : cette fois le petit cadenas n'est pas barré, la connexion est sécurisée : chiffrée en HTTPS et le serveur est authentifié par un certificat SSL à jour.



- Effectuer une nouvelle requête avec l'URL <http://frederic-junier.org/NSI/Sandbox/index.html>. Quel est le code d'état de la réponse ? Explication ?

Réponse : cette fois on a une erreur 404, c'est une erreur du client qui a demandé une ressource qui n'existe pas sur le serveur. Pourquoi ? C'est un problème de casse de caractère dans l'URL où il ne faut pas de majuscule sur sandbox.



Not Found

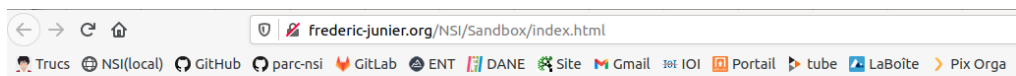
The requested URL was not found on this server.

Additionally, a 404 Not Found error was encountered while trying to use an ErrorDocument to handle the request.

État	Méthode	Domaine	Fichier
404	GET	frederic-junier.org	index.html
404	GET	frederic-junier.org	favicon.ico

- Effectuer une nouvelle requête avec l'URL <http://frederic-junier.org/NSI/interdit>. Quel est le code d'état de la réponse ? Explication ?

Réponse : cette fois on a une erreur 403 Forbidden : le serveur comprend la requête mais refuse de l'autoriser. En effet, les droits d'accès sur le répertoire interdit sont positionnés à 710 soit RWX-X—, donc aucun droit pour un utilisateur "autre"



Not Found

The requested URL was not found on this server.

Additionally, a 404 Not Found error was encountered while trying to use an ErrorDocument to handle the request.

État	Méthode	Domaine	Fichier
404	GET	frederic-junier.org	index.html
404	GET	frederic-junier.org	favicon.ico

Modification des attributs du fichier ✕

Sélectionnez les nouveaux attributs pour le dossier "interdit".

Permissions du propriétaire

☒ Lire ☒ Écrire ☒ Exécuter

Permissions de groupe

☐ Lire ☐ Écrire ☒ Exécuter

Permissions publiques

☐ Lire ☐ Écrire ☐ Exécuter

Valeur numérique :

Vous pouvez appliquer un x sous n'importe quelle position pour conserver les permissions initiales des fichiers.

☐ Récursion dans les sous-dossiers

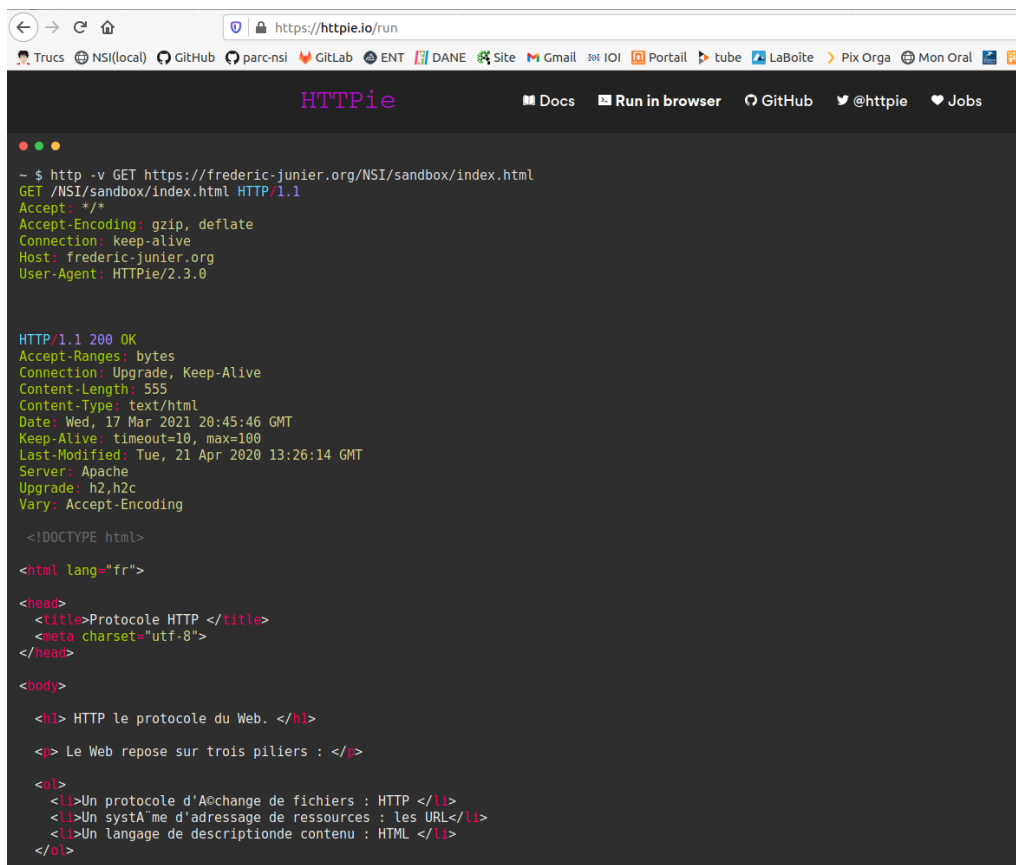
☒ Appliquer à tous les fichiers et dossiers

☐ Appliquer uniquement aux fichiers

☐ Appliquer uniquement aux dossiers

2. Le site <https://httpie.org/> propose un client HTTP en ligne de commandes permettant de décomposer les requêtes HTTP en précisant la méthode et l'URL de la ressource demandée.
 - Ouvrir la page <https://httpie.org/run>
 - Saisir la commande `http -v GET https://frederic-junier.org/NSI/sandbox/index.html`. Décrire le fonctionnement de la méthode GET du protocole HTTP : formats de la requête et de la réponse.

*Réponse : la méthode **GET** du protocole **HTTP** est celle qui s'exécute par défaut lorsqu'on effectue une requête en saisissant l'URL de la page dans la barre d'adresse du navigateur. Voir ci-dessus pour les en-têtes client/serveur. On voit aussi apparaître le contenu de la réponse : le code source **HTML** de la page demandée.*



```
~ $ http -v GET https://frederic-junier.org/NSI/sandbox/index.html
GET /NSI/sandbox/index.html HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: frederic-junier.org
User-Agent: HTTPie/2.3.0

HTTP/1.1 200 OK
Accept-Ranges: bytes
Connection: Upgrade, Keep-Alive
Content-Length: 555
Content-Type: text/html
Date: Wed, 17 Mar 2021 20:45:46 GMT
Keep-Alive: timeout=10, max=100
Last-Modified: Tue, 21 Apr 2020 13:26:14 GMT
Server: Apache
Upgrade: h2,h2c
Vary: Accept-Encoding

<!DOCTYPE html>

<html lang="fr">

<head>
  <title>Protocole HTTP </title>
  <meta charset="utf-8">
</head>

<body>

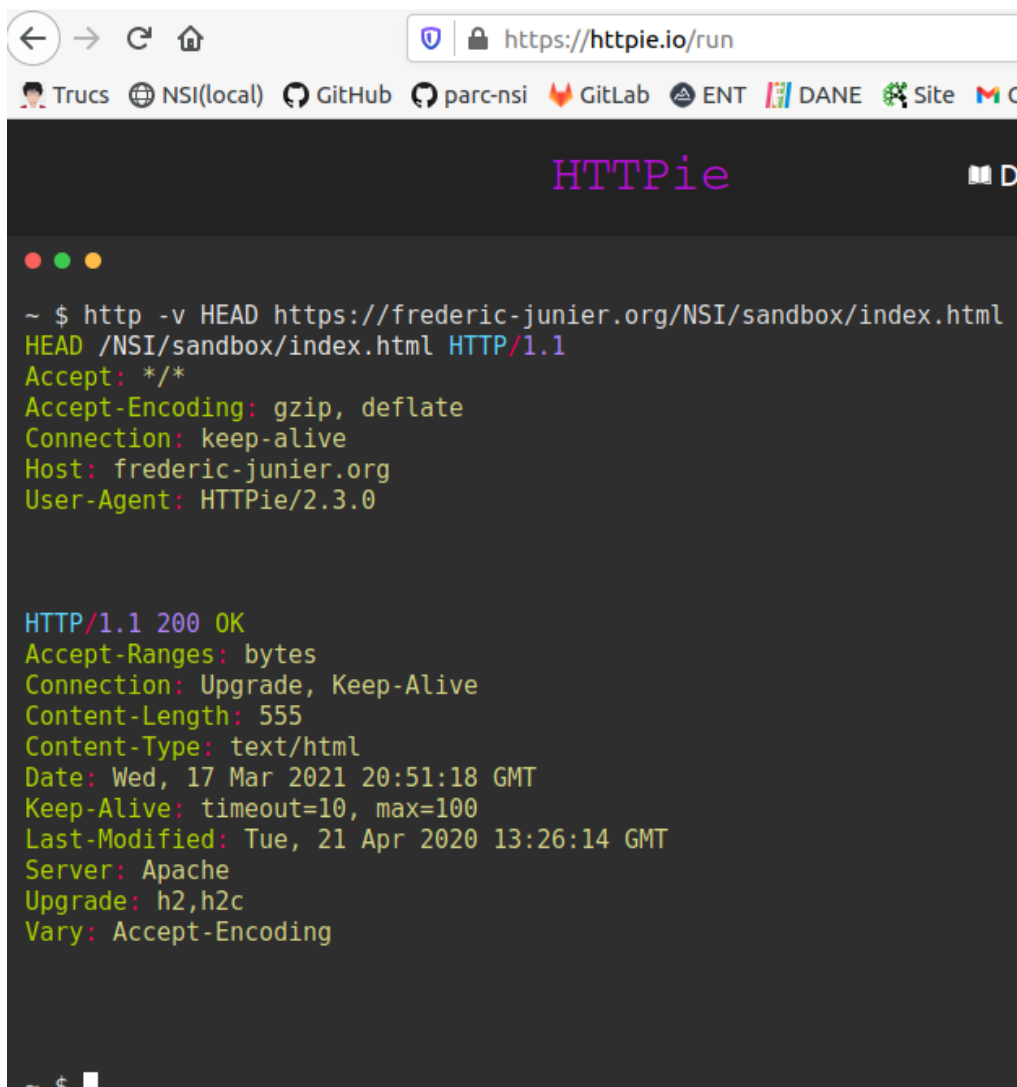
  <h1> HTTP le protocole du Web. </h1>

  <p> Le Web repose sur trois piliers : </p>

  <ol>
    <li>Un protocole d'Echange de fichiers : HTTP </li>
    <li>Un systA"me d'adressage de ressources : les URL</li>
    <li>Un langage de descriptionde contenu : HTML </li>
  </ol>
```

- Saisir la commande `http -v HEAD https://frederic-junier.org/NSI/sandbox/index.html`. Décrire le fonctionnement de la méthode `HEAD` du protocole `HTTP` : formats de la requête et de la réponse.

Réponse : Avec la méthode `HEAD` on ne récupère que les en-têtes.

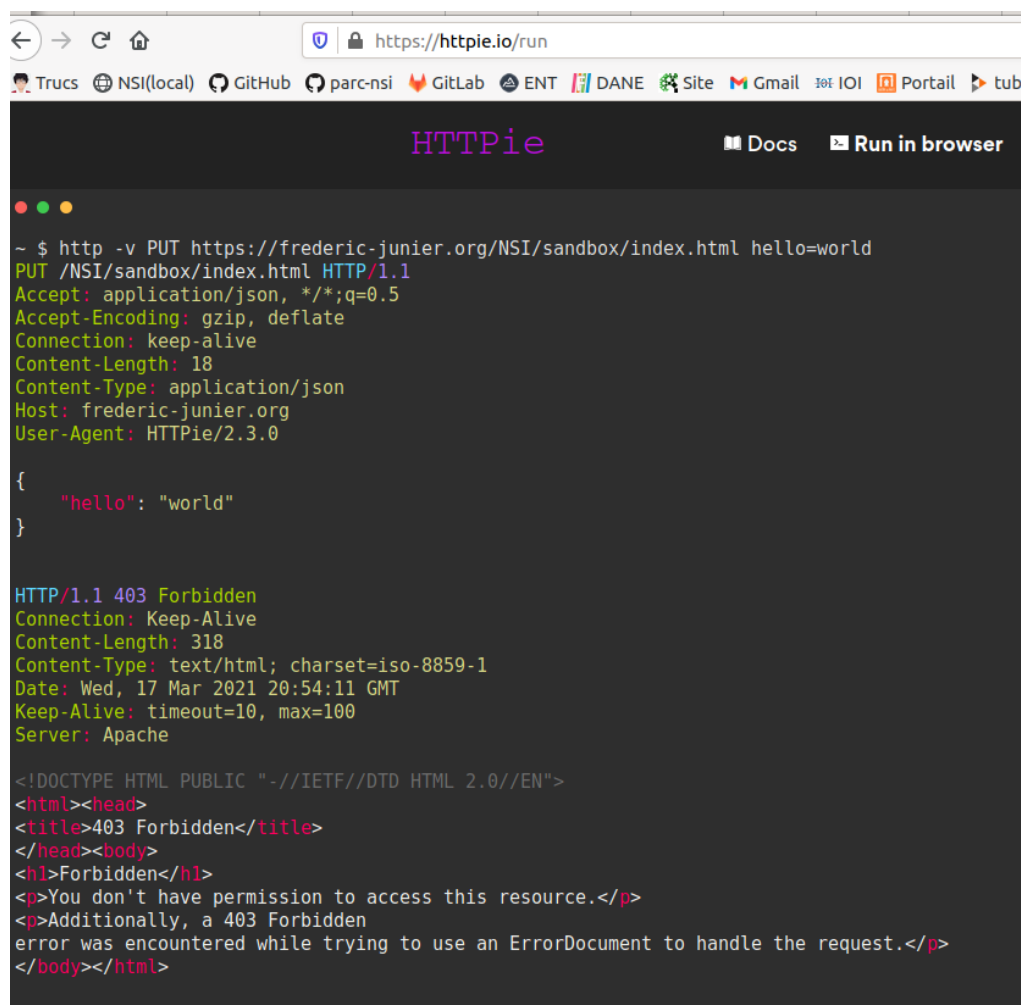


```
~ $ http -v HEAD https://frederic-junier.org/NSI/sandbox/index.html
HEAD /NSI/sandbox/index.html HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Host: frederic-junier.org
User-Agent: HTTPie/2.3.0

HTTP/1.1 200 OK
Accept-Ranges: bytes
Connection: Upgrade, Keep-Alive
Content-Length: 555
Content-Type: text/html
Date: Wed, 17 Mar 2021 20:51:18 GMT
Keep-Alive: timeout=10, max=100
Last-Modified: Tue, 21 Apr 2020 13:26:14 GMT
Server: Apache
Upgrade: h2,h2c
Vary: Accept-Encoding
```

- Saisir la commande `http -v PUT https://frederic-junier.org/NSI/sandbox/index.html hello=world`. Quel résultat obtient-on ? Explication ^a ?

Réponse: La méthode PUT remplace toutes les représentations actuelles de la ressource visée par le contenu de la requête. Ici le serveur renvoie un code d'erreur 403 FORBIDDEN car un utilisateur "autre" ne peut pas modifier la ressource (heureusement).



```
~ $ http -v PUT https://frederic-junier.org/NSI/sandbox/index.html hello=world
PUT /NSI/sandbox/index.html HTTP/1.1
Accept: application/json, */*;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Content-Length: 18
Content-Type: application/json
Host: frederic-junier.org
User-Agent: HTTPie/2.3.0

{
  "hello": "world"
}

HTTP/1.1 403 Forbidden
Connection: Keep-Alive
Content-Length: 318
Content-Type: text/html; charset=iso-8859-1
Date: Wed, 17 Mar 2021 20:54:11 GMT
Keep-Alive: timeout=10, max=100
Server: Apache

<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html><head>
<title>403 Forbidden</title>
</head><body>
<h1>Forbidden</h1>
<p>You don't have permission to access this resource.</p>
<p>Additionally, a 403 Forbidden
error was encountered while trying to use an ErrorDocument to handle the request.</p>
</body></html>
```

^aNote : Méthodes HTTP : voir <https://developer.mozilla.org/fr/docs/Web/HTTP/M%C3%A9thode>

1.2 Passage de paramètres dans une URL



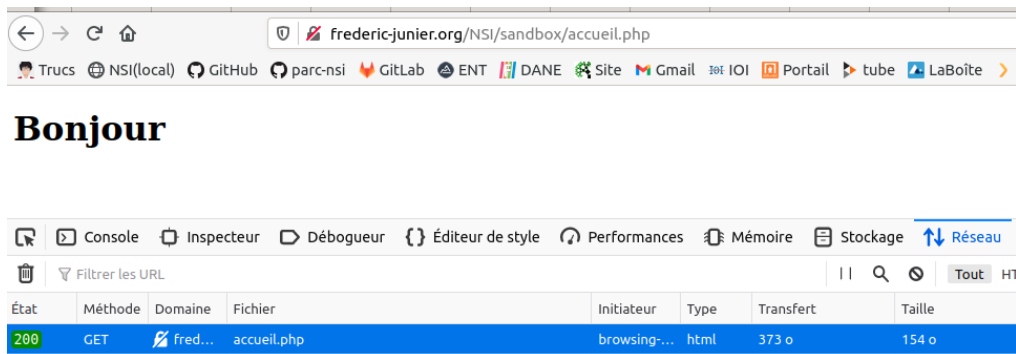
Exercice 3

Ouvrir un navigateur Web.

1. Demander la page d'adresse <http://frederic-junier.org/NSI/sandbox/accueil.php>

Quel est l'affichage obtenu ?

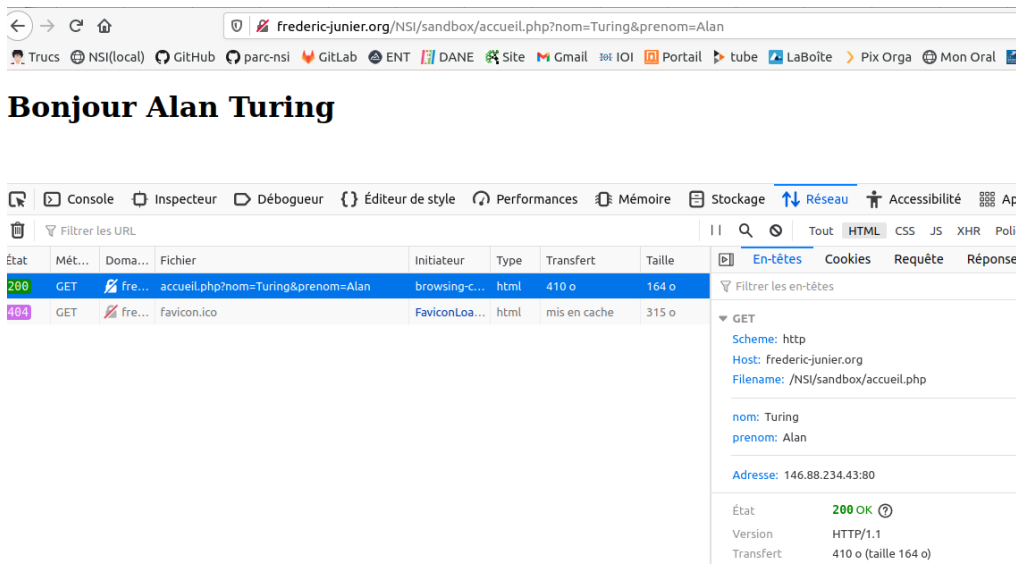
Réponse : La requête est envoyée avec la méthode GET, le code 200 de la réponse du serveur indique un succès



- Demander la page d'adresse

<http://frederic-junier.org/NSI/sandbox/accueil.php?nom=Turing&prenom=Alan>

Quel est l'affichage obtenu ? Ouvrir les outils de développement avec F12 puis sélectionner les onglets Réseau et Paramètres.



*Réponse : Dans la barre des outils de développement, onglet Réseau, on a la décomposition de l'URL : le schéma (protocole http), le nom de domaine (frederic-junier.org), le chemin vers la ressource (/NSI/sandbox/accueil.php) et nouveauté des paramètres assemblés une **chaîne de requête**, elle commence par le symbole ? puis contient une liste de paires **nom=valeur** séparées par un symbole esperluette &. Ces paramètres ne font pas partie de l'adresse de la ressource mais sont une façon pour le client de transmettre des informations au serveur.*

- Remplacer Turing par votre nom et Alan par votre prénom dans l'URL précédente. Que peut-on remarquer ? À votre avis, que se passe-t-il sur le serveur lorsqu'il reçoit la requête HTTP ?

*Réponse : côté serveur, un programme doit fabriquer la page **HTML** renvoyée au client à la volée en fonction des paramètres transmis. On parle dans ce cas de **page web dynamique**.*

Le programme s'exécutant côté serveur est écrit en *PHP*. On donne le code ci-dessous



4. Voici le contenu du fichier `accueil.php` sur le serveur. S'agit-il d'un texte écrit en *HTML* ? Faire une recherche sur la signification de l'acronyme *PHP*.

```
<!DOCTYPE html>
<html lang="fr">
<head>
  <title>Accueil </title>
  <meta charset="utf-8">
</head>
<body>
<h1>
<?php
echo "Bienvenue " . $_GET['prenom'] . " " . $_GET['nom'] ;
?>
</h1>
</body>
</html>
```

4. Enregistrer l'*URL* testée précédemment dans les marque-pages du navigateur. Ouvrir un autre onglet et cliquer sur le signet enregistré. Retrouve-t-on la même page Web ?

Réponse : il est tout à fait possible d'enregistrer une URL avec paramètres dans les signets de son navigateur.

2 Formulaire et passage de paramètres

2.1 Un premier exemple



Exemple 1

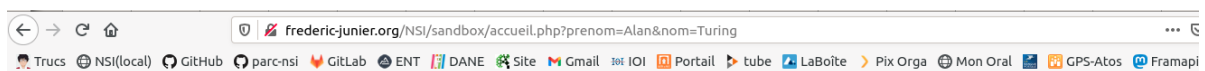
1. Ouvrir avec un navigateur Web la page d'*URL* :

<http://frederic-junier.org/NSI/sandbox/formulaire-get.html>

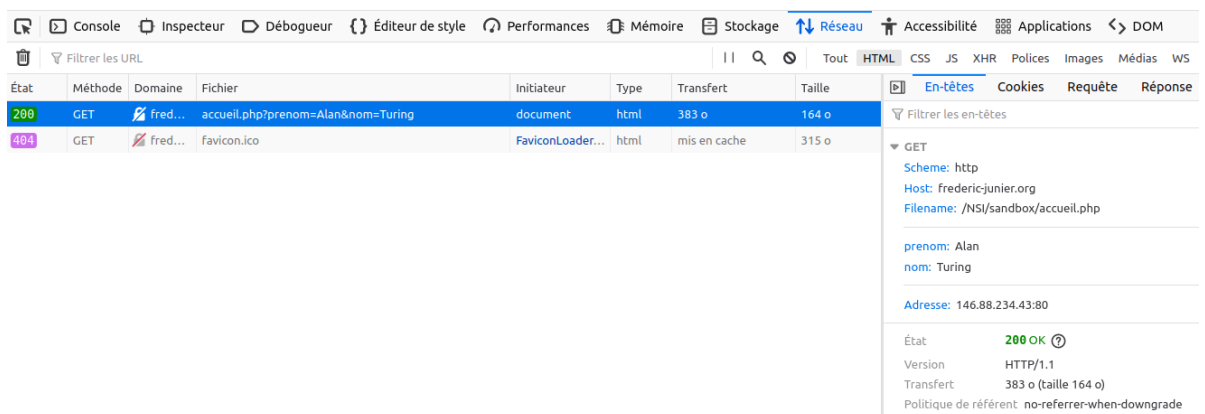
Prénom : **Nom :**

- Cliquer sur le bouton **Envoyer**. Que se passe-t-il ? Rafraîchir la page avec F5. Que se passe-t-il ?

*Réponse : en cliquant sur le bouton envoyer, on génère côté client une URL avec comme paramètres les valeurs des champs du formulaire : <http://frederic-junier.org/NSI/sandbox/accueil.php?prenom=Alan&nom=Turing>. Le client envoie au serveur sa requête avec la méthode **GET** Si on rafraîchit la page avec F5, le comportement est répété à l'identique puisque le client construit la même requête à partir de l'URL avec paramètres.*



Bonjour Alan Turing



- Changer les valeurs des champs **Prénom** et **Nom** du formulaire puis cliquer sur le bouton **Envoyer**. Que se passe-t-il ?

*Réponse : mise à jour, une partie de la logique pour générer la page web dynamique est déportée côté client (choix des paramètres dans une Interface Homme Machine <https://interstices.info/domaine/interaction-hommemachine/> représentée par le formulaire). Noter que les paramètres apparaissent dans l'onglet en-tête **GET** des outils de développement et non dans l'onglet Requête.*

- Ouvrir la fenêtre des outils de développement et afficher dans l'onglet Réseau l'entête de la requête HTTP qui devrait ressembler à celui-ci :

En-têtes	Cookies	Paramètres	Réponse	Délais
URL de la requête : http://frederic-junier.org/NSI/sandbox/accueil.php?prenom=Alan&nom=Turing				
Méthode de la requête : GET				
Adresse distante : 146.88.234.43:80				
Code d'état : 200 OK ⓘ				
Version : HTTP/1.1				
Politique de référent : no-referrer-when-downgrade				

Sélectionner l'onglet Paramètres et vérifier qu'on obtient les paramètres transmis à travers l'URL dans la chaîne de requête comme dans l'exercice 3.

- Afficher le code source de la page `formulaire-get.html` avec le raccourci clavier CTRL + U. On devrait obtenir le texte ci-dessous :

```
<!DOCTYPE html>

<html lang="fr">

<head>
<title>Formulaire HTML </title>
<meta charset="utf-8">
</head>
<body>

  <form action = "accueil.php" method="GET">
    <label for="id_prenom">Prénom :</label>
    <input type="text" id="id_prenom" name="prenom" value="Alan">
    <label for="id_nom">Nom :</label>
    <input type="text" id="id_nom" name="nom" value="Turing">
    <button type="submit" id="bouton">Envoyer</button>
  </form>

</body>
</html>
```

2. Ouvrir avec un navigateur Web la page d'URL :

<http://frederic-junier.org/NSI/sandbox/formulaire-post.html>

- Cliquer sur le bouton Envoyer. Que se passe-t-il ?

Réponse : le serveur renvoie la même page que pour la requête précédente mais cette fois les paramètres du formulaire ne sont pas transmis à travers l'URL et donc dans l'entête HTTP comme avec la méthode GET. Ils sont transmis dans le corps de la requête HTTP comme on peut le voir dans l'onglet Requête des outils de développement. On peut remarquer que la méthode de la requête HTTP a changé, il s'agit de la méthode POST.

frederic-junior.org/NSI/sandbox/accueil-post.php

Bonjour Alan Turing

Console Inspecteur Débogueur Éditeur de style Performances Mémoire Stockage Réseau Accessibilité Applications DOM

Filtrer les URL

État	Méthode	Domaine	Fichier	Initiateur	Type	Transfert	Taille	En-têtes	Cookies	Requête	Réponse
200	POST	fred...	accueil-post.php	document	html	411 o	165 o	Filtrer les en-têtes			
404	GET	fred...	favicon.ico	FaviconLoader...	html	mis en cache	315 o	Filtrer les en-têtes			

POST

Schema: http
Host: frederic-junior.org
Filename: /NSI/sandbox/accueil-post.php
Adresse: 146.88.234.43:80
État: 200 OK

frederic-junior.org/NSI/sandbox/accueil-post.php

Bonjour Alan Turing

Console Inspecteur Débogueur Éditeur de style Performances Mémoire Stockage Réseau Accessibilité Applications DOM

Filtrer les URL

État	Mét...	Dom...	Fichier	Initiateur	Type	Transfert	Taille	En-têtes	Cookies	Requête	Répon
200	POST	fr...	accueil-post.php	document	html	411 o	165 o	Filtrer les paramètres de la requête			
404	GET	fr...	favicon.ico	FaviconLo...	html	mis en cache	315 o	Filtrer les paramètres de la requête			

Données de formulaire

prenom: "Alan"
nom: "Turing"

Transmission de la requête

1 prenom=Alan&nom=Turing

- Changer les valeurs des champs **Prénom** et **Nom** du formulaire puis cliquer sur le bouton **Envoyer**. Que se passe-t-il ? Observe-t-on un changement dans l'**URL** de la requête ? dans son entête ?

*Réponse : comme on l'a écrit ci-dessus, la méthode **POST** n'envoie pas les paramètres de formulaire à travers l'**URL** et l'entête **HTTP** comme avec la méthode **GET** mais directement dans le corps de la requête.*

- Rafraîchir la page avec **F5**. Que se passe-t-il ?

*Réponse : **HTTP** est un protocole sans mémoire, pour générer la page web il faut renvoyer les paramètres qui ne sont pas dans l'**URL** d'où l'avertissement ci-dessous.*

Pour afficher cette page, les informations précédemment transmises par Firefox doivent être renvoyées. Ceci répètera toute action (telle qu'une recherche ou un ordre d'achat) entreprise précédemment.

Annuler Renvoyer

- Ouvrir la fenêtre des outils de développement et afficher dans l'onglet Réseau l'entête de la requête HTTP qui devrait ressembler à celui-ci :

En-têtes	Cookies	Paramètres	Réponse	Délais
URL de la requête : http://frederic-junier.org/NSI/sandbox/accueil-post.php				
Méthode de la requête : POST				
Adresse distante : 146.88.234.43:80				
Code d'état : 200 OK ?				
Version : HTTP/1.1				
Politique de référent : no-referrer-when-downgrade				

Sélectionner l'onglet Paramètres et vérifier qu'on retrouve les paramètres transmis dans l'URL. Quelle différence par rapport à la méthode vue en question 1 ?

En-têtes	Cookies	Paramètres	Réponse	Délais
Filtrer les paramètres de la requête				
Données de formulaire				
<pre> prenom: "Alan" nom: "Turing" </pre>				
Transmission de la requête				
<pre> 1 prenom=Alan&nom=Turing </pre>				

- Afficher le code source de la page formulaire-post.html avec le raccourci clavier CTRL + U. Quels sont les deux changements par rapport au code de formulaire-get.html ?

_Réponse :

```

<!DOCTYPE html>

<html lang="fr">

<head>
  <title>Formulaire HTML </title>
  <meta charset="utf-8">
</head>

<body>

  <!-- Changement dans la balise form par rapport à formulaire-get.html
       : la cible et la méthode -->
  <form action = "accueil-post.php" method="post">

    <label for="id_prenom">Prénom :</label>
    <input type="text" id="id_prenom" name="prenom" value="Alan">

    <label for="id_nom">Nom :</label>
    <input type="text" id="id_nom" name="nom" value="Turing">

    <button type="submit" id="bouton">Envoyer</button>

```

```
</form>

</body>
</html>
```

2.2 Méthodes de passage des paramètres : GET ou POST



Exercice 4

QCM de type E3C2.

1. Parmi les réponses suivantes, que permet d'effectuer la méthode **POST** du protocole **HTTP** ?
 - Réponse A : Définir le style d'une page web
 - Réponse B : Pirater des données bancaire
 - Réponse C : Envoyer une page web vers le client
 - Réponse D : Envoyer les données saisies dans un formulaire HTML vers un serveur => **BONNE RÉPONSE**
2. Un site internet utilise une requête **HTTP** avec la méthode **POST** pour transmettre les données d'un formulaire. Laquelle des affirmations suivantes est **incorrecte** ?
 - Réponse A : les données envoyées ne sont pas visibles => **BONNE RÉPONSE**
 - Réponse B : il est possible de transmettre des données de type binaire
 - Réponse C : les données transmises sont cryptées
 - Réponse D : il n'y a pas de restriction de longueur pour les données transmises
3. Un internaute clique sur un lien qui envoie la requête **HTTP** suivante à un serveur :
`http://jaimelaneige.com/ma_planche/traitement.php?nom=Snow&prenom=Jon`
Que demande cette requête au serveur ?
 - Réponse A : de renvoyer le fichier traitement.php en identifiant nom et prénom à Snow et Jon
 - Réponse B : d'exécuter le fichier traitement.php en identifiant nom et prénom à Snow et Jon => **BONNE RÉPONSE**
 - Réponse C : d'indiquer si Jon Snow a bien pris son traitement
 - Réponse D : de renvoyer le fichier traitement.php en affichant prénom et nom : Jon

2.3 Eléments de formulaire



Exercice 5

1. Ouvrir dans un navigateur Web la page dont l'URL est :

<https://repl.it/@fredericjunier/NSI-formulaire-exo5-radio>.

- Cliquer sur Run pour lancer le serveur, remplir le formulaire contenu dans le fichier `index.php` et envoyer les données. Quelle méthode est utilisée pour le passage des paramètres du formulaire ?

Réponse : Les paramètres du formulaire sont transmis avec la méthode *POST*

The screenshot shows a web browser window with the URL `https://NSI-formulaire-exo5-radio.fredericjunier.repl.co`. The page displays a form titled "Quel navigateur Web utilisez-vous ?". The form contains four radio buttons labeled "Safari", "Chrome", "firefox" (which is selected), and "edge". Below the radio buttons is a "Bouton d'envoi" (Submit button). The browser's developer tools are open, showing the source code of the `index.php` file. The code is an HTML form with the following structure:

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>Formulaire HTML </title>
<meta charset="utf-8">
</head>
<body>
<h1> Quel navigateur Web utilisez-vous ? </h1>
<form action="navigateur.php" method="POST">
  <input type="radio" id="safari" name="navig" value="safari">
  <label for="safari">Safari</label><br>
  <input type="radio" id="chrome" name="navig" value="chrome">
  <label for="chrome">Chrome</label><br>
  <input type="radio" id="firefox" name="navig" value="firefox">
  <label for="firefox">firefox</label><br>
  <input type="radio" id="edge" name="navig" value="edge">
  <label for="edge">edge</label><br>
  <button type="submit">Bouton d'envoi</button>
</form>
</body>
</html>
```

- Modifier les codes sources des fichiers `index.php` et `navigateur.php` pour changer la méthode de passage des paramètres du formulaire. En PHP, on récupère la valeur du paramètre `nom` avec `$_GET['nom']` si la méthode est *GET* ou `$_POST['nom']` si c'est *POST*.

The screenshot shows the same web browser window, but the form has been modified. The title is still "Quel navigateur Web utilisez-vous ?". However, the radio buttons have been replaced by a dropdown menu with the same four options: "Safari", "Chrome", "Firefox", and "Edge". The "Envoyer" (Submit) button is still present. The browser's developer tools show the updated source code for `index.php`:

```
<!DOCTYPE html>
<html lang="fr">
<head>
<title>Formulaire HTML </title>
<meta charset="utf-8">
</head>
<body>
<h1> Quel navigateur Web utilisez-vous ? </h1>
<form action="navigateur.php" method="GET">
  <select name="navig">
    <option value="safari">Safari</option>
    <option value="chrome">Chrome</option>
    <option value="firefox">Firefox</option>
    <option value="edge">Edge</option>
  </select>
  <button type="submit">Envoyer</button>
</form>
</body>
</html>
```

Réponse : code source modifié du fichier qui contient le formulaire *index.php*

```
<!DOCTYPE html>

<html lang="fr">

<head>
  <title>Formulaire HTML </title>
```

```

    <meta charset="utf-8">
</head>

<body>

    <h1> Quel navigateur Web utilisez-vous ? </h1>
    <form action="navigateur.php" method="GET">
        <select name="navig">
            <option value="safari">Safari</option>
            <option value="chrome">Chrome</option>
            <option value="firefox">Firefox</option>
            <option value="edge">Edge</option>
        </select>
        <button type="submit">Envoyer</button>
    </form>
</body>

```

Réponse : code source modifié du fichier qui génère la page à partir des données du formulaire `navigateur.php`

```

<!DOCTYPE html>

<html lang="fr">

<head>
    <title>Accueil </title>
    <meta charset="utf-8">
</head>

<body>

<h1>
<?php
echo "Vous utilisez le navigateur " . $_GET['navig'] ;
?>
</h1>

<a href="index.php">Retour au formulaire</a>
</body>
</html>

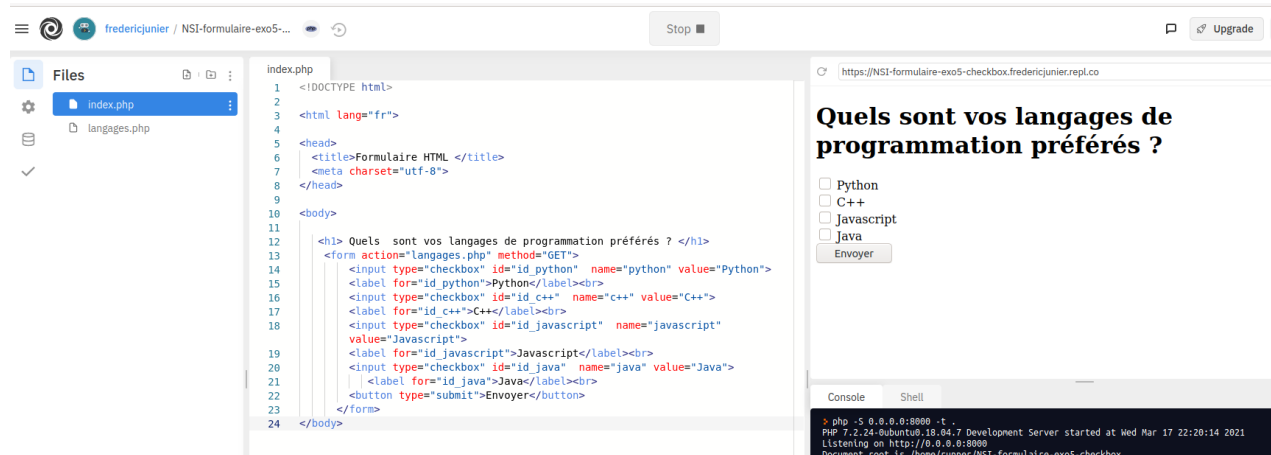
```

- Consulter la documentation sur l'élément de formulaire `<select>` contenue dans la page https://www.w3schools.com/html/html_form_elements.asp et remplacer les `<input>` de type radio du formulaire dans `index.php` par un élément `<select>` avec choix unique.
2. Ouvrir dans un navigateur Web la page dont l'URL est :

<https://repl.it/@fredericjunier/NSI-formulaire-exo5-checkbox>.

- Cliquer sur Run pour lancer le serveur, remplir le formulaire contenu dans le fichier `index.php` et envoyer les données. Quelle méthode est utilisée pour le passage des paramètres du formulaire ?

Réponse : les paramètres de formulaire sont envoyés avec la méthode *GET*



- Modifier les codes sources des fichiers `index.php` et `langages.php` pour changer la méthode de passage des paramètres du formulaire.

Réponse : code de *index.php* pour utiliser la méthode *POST*

```
<!DOCTYPE html>

<html lang="fr">

<head>
  <title>Formulaire HTML </title>
  <meta charset="utf-8">
</head>

<body>

  <h1> Quels sont vos langages de programmation préférés ? </h1>
  <form action="langages.php" method="POST">
    <input type="checkbox" id="id_python" name="python" value="Python"
    ">
    <label for="id_python">Python</label><br>
    <input type="checkbox" id="id_c++" name="c++" value="C++">
    <label for="id_c++">C++</label><br>
    <input type="checkbox" id="id_javascript" name="javascript" value
    ="Javascript">
    <label for="id_javascript">Javascript</label><br>
    <input type="checkbox" id="id_java" name="java" value="Java">
```

```

        <label for="id_java">Java</label><br>
        <button type="submit">Envoyer</button>
    </form>
</body>

```

Réponse : code de *langages.php* pour utiliser la méthode *POST*

```

<!DOCTYPE html>

<html lang="fr">

<head>
    <title>Accueil </title>
    <meta charset="utf-8">
</head>

<body>

<p>
    Vos langages préférés sont :
</p>
<ul>
<?php
foreach ($_POST as $_langage) {
    echo "<li>" . $_langage . "</li>";
}
?>
</ul>

</body>
</html>

```

- Consulter la documentation sur l'élément de formulaire `<select>` contenue dans la page https://www.w3schools.com/html/html_form_elements.asp et remplacer les `<input>` de type checkbox du formulaire dans *index.php* par un élément `<select>` avec choix multiple. Pour modifier le code PHP on s'inspirera de ce [post stackoverflow](#).

Réponse : code de *index.php* pour utiliser un formulaire `<select>` multiple

```

<!DOCTYPE html>

<html lang="fr">

<head>
    <title>Formulaire HTML </title>
    <meta charset="utf-8">

```

```

</head>

<body>

    <h1> Quels sont vos langages de programmation préférés ? </h1>
    <form action="langages.php" method="GET">
        <select id="top_langages" name="top_langages[]" multiple>
            <option value="Python">
                <label for="id_python">Python</label><br>
            <option value="c++">
                <label for="id_c++">C++</label><br>
            <option value="javascript">
                <label for="id_javascript">Javascript</label><br>
            <option value="java">
                <label for="id_java">Java</label><br>
        </select>
        <button type="submit">Envoyer</button>
    </form>
</body>

```

Réponse : code de `langages.php` pour utiliser un formulaire `<select>` multiple

```

<!DOCTYPE html>

<html lang="fr">

<head>
    <title>Accueil </title>
    <meta charset="utf-8">
</head>

<body>

<p>
    Vos langages préférés sont :
</p>
<ul>
<?php
foreach ($_GET["top_langages"] as $_langage) {
    echo "<li>" . $_langage . "</li>";
}
?>
</ul>

</body>

```

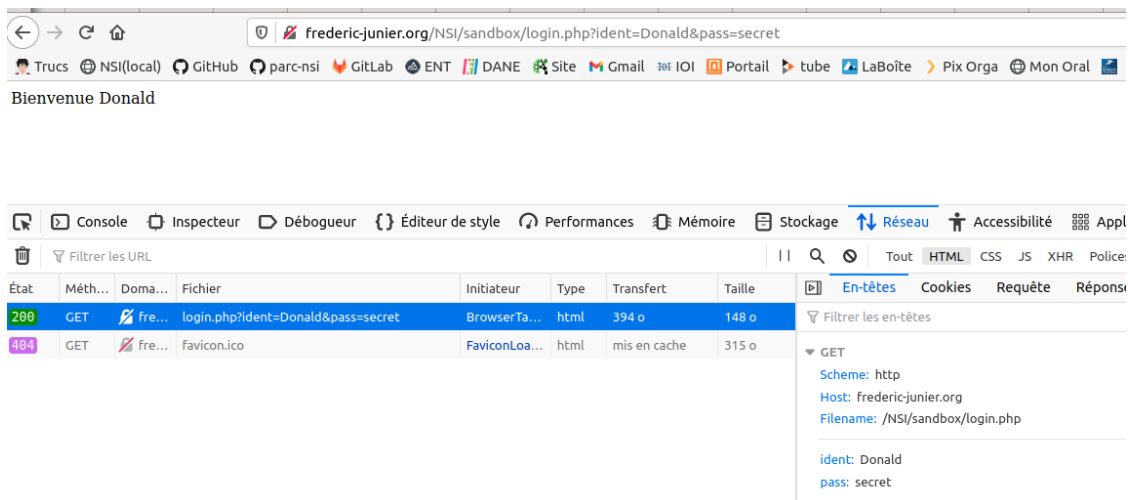
</html>

4. Ouvrir dans un navigateur Web la page dont l'URL est :

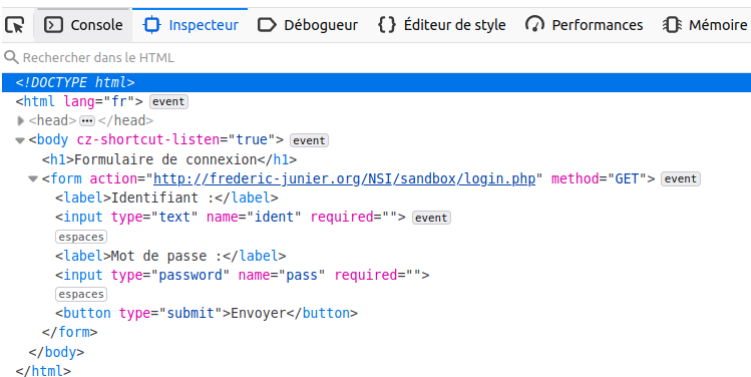
<http://frederic-junier.org/NSI/sandbox/NSI-formulaire-exo5-login-get.html>.

- La page présente un formulaire basique de connexion avec deux champs **login** et **password**. La valeur de l'identifiant est libre et le mot de passe est **secret**. Remplir le formulaire et envoyer les données. Quelle méthode de passage des paramètres est utilisée ? La transmission du mot de passe est-elle satisfaisante ?

Réponse : la transmission de mot de passe avec *GET* n'est vraiment pas une bonne idée, les paramètres sont transmis dans l'URL et donc dans les entêtes *HTTP* qui ne sont jamais chiffrés même en *HTTPS*



- Revenir sur la page du formulaire, ouvrir la fenêtre des outils de développement avec F12 et modifier le code source pour que l'envoi du mot de passe soit sécurisé.



Réponse : on effectue deux modifications dans le code source du formulaire : on utilise la méthode *POST* et on passe en *HTTPS* le protocole de l'URL cible



- Dans le schéma ci-dessous d'un échange Web sécurisé avec le protocole **HTTPS**, apparaît la notion de **certificat**. Quel est le rôle d'un **certificat** et comment est-il géré par le navigateur ?

Réponse : lire la BD ci-dessous et consulter cette page <https://support.mozilla.org/fr/kb/certificats-authentication-pour-sites-web-securises>

QUOI?! Ton site web est en "http"?!
mon pti coeur...!



Passer en https (http "secure") est le minimum requis en sécurité. C'est sécurisé via du "TLS". *



En http, tu n'as aucune garantie que ton interlocuteur est bien celui que tu penses...



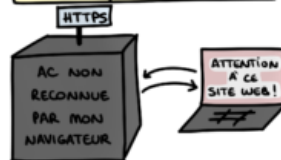
Le protocole TLS (Transport Layer Security) sécurise les échanges Internet. Une Autorité de Certification (AC) délivre un certificat qui authentifie le serveur TLS souhaité.



Ce certificat garantit l'identité du serveur et le fait qu'il soit chiffrer ses flux.



Attention: toutes les AC n'ont pas la même légitimité. Les navigateurs ont chacun leurs "AC de confiance".



Le https t'assure de l'identité de ton interlocuteur puis établit une connexion chiffrée et sécurisée.

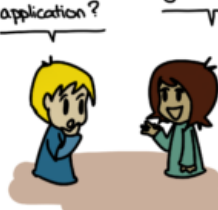


L'URL d'un site fiable est précédée d'un cadenas ou de "https".

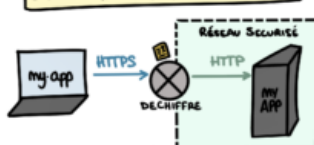


Et je mets ce certificat sur le serveur de mon application?

Il y a 3 façons de gérer le https.



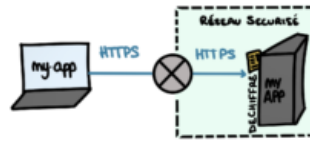
① Mon certificat est à l'entrée de mon infra. Les flux circulent en clair à l'intérieur.



② Il y a un certificat à l'entrée, et un autre sur le serveur applicatif.



③ Le certificat est sur le serveur applicatif.



L'option ① facilite le debug et l'installation de sondes...



...mais la sécurité préfère les autres:

- la ② permet de contrôler les flux entrants
- la ③ garantit la confidentialité des données



Autre avantage du https: ton site générera plus de trafic.

- ↑ il sera mieux référencé dans google
- les navigateurs ne mettront plus d'alerte dessus
- les clients seront plus susceptibles d'acheter

Donc passe en https!

