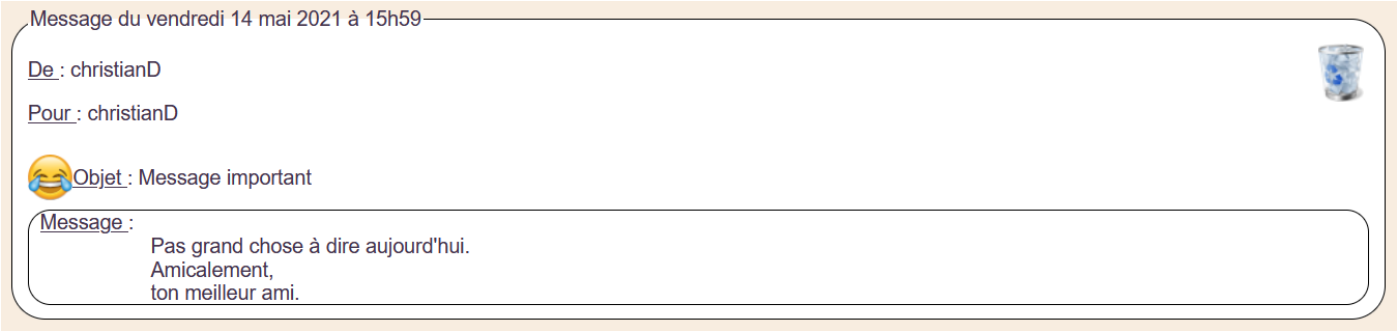
OBJECTIFS : L'objectif de ce TP est d'écrire une page web qui comporte un ***formulaire***. Il permettra de générer et d'envoyer des messages à vos camarades de classe. Les messages pourront être lus ensuite supprimés, dans l'espace Boite Aux Lettres de chacun, sur le site https://nsibranly.fr/.



DOCUMENT A RENDRE : Ce travail est évalué. Vous aurez à créer un fichier nommé *formulaires.html* qui contiendra le formulaire. Vous le transférerez dans votre espace de https://nsibranly.fr/ , avec le code *web* pour le transfert. Vous insérerez un lien vers ce fichier *formulaires.html* dans votre page *index.html* et pourrez ainsi tester son fonctionnement.

1. Démarrage :

⇨ Télécharger le zip contenant les fichiers nécessaires à la réalisation de ce TP à partir du lien disponible dans le bloc « *Html Css* » de nsibranly.fr . Enregistrer les fichiers dans un répertoire nommé « *formulaire* » par exemple.

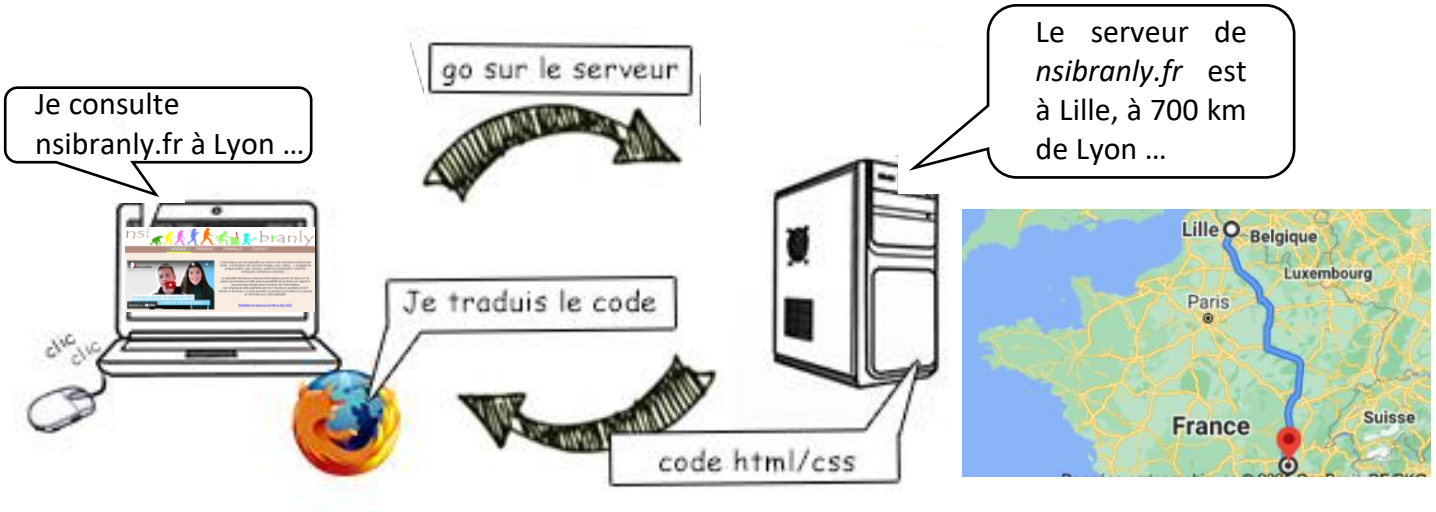
⇨ Créer un nouveau fichier enregistré sous le nom *formulaire.html* dans ce même répertoire et le compléter avec le minimum requis :

```
<!DOCTYPE html>
<html lang="fr">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Chez moi</title>
        <link rel="icon"  type="img/png" href="poste.png"/>
        <link rel="stylesheet" href="formulaire.css">
    </head>
    <body>



    </body>
</html>
```

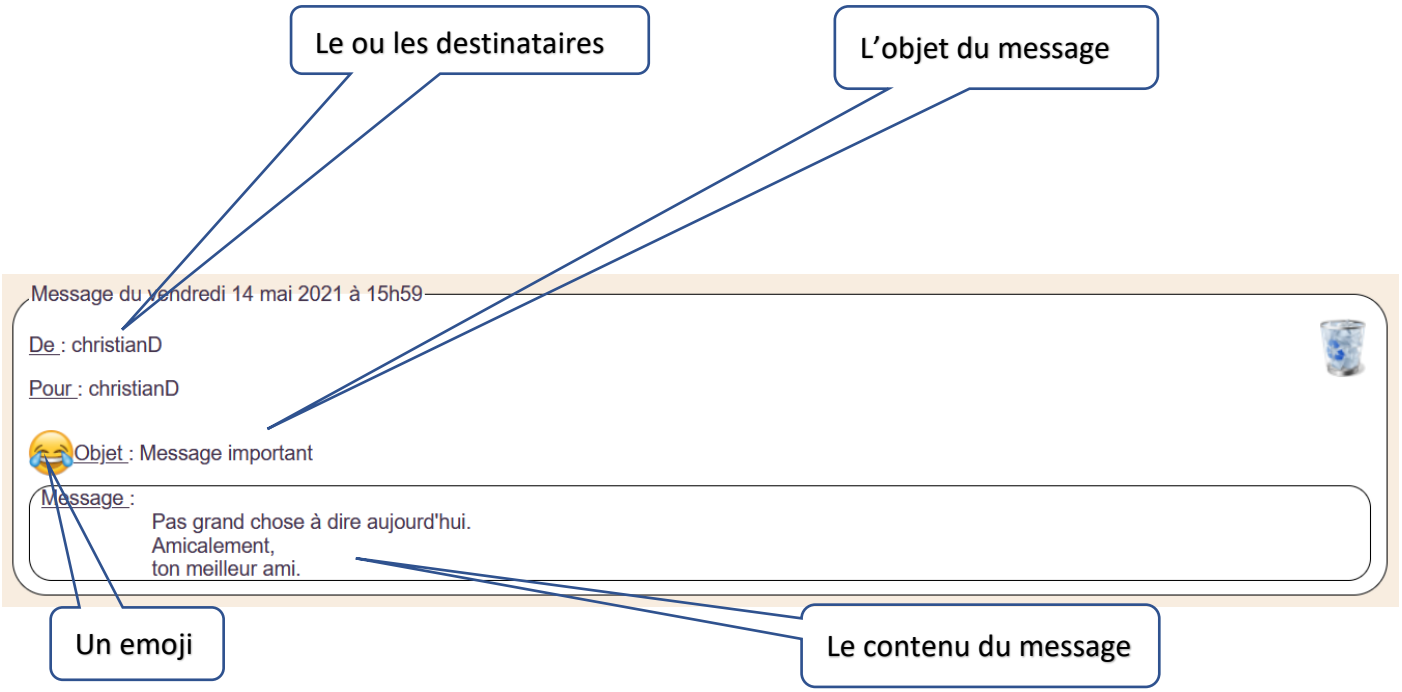
⇨ Créer un nouveau fichier enregistré sous le nom *formulaire.css* que l'on laisse vide pour l'instant.

Le fichier image qu'on vous propose pour mettre en icone, est donné ci-contre (il est dans le répertoire zip téléchargé en début de Tp). Vous pourrez prendre une autre image en ayant pris soin de réduire sa largeur à environ 30 px avec le logiciel *Gimp*.

Un formulaire permet d'envoyer des données sur le serveur qui héberge le site web :



Dans ce Tp, le formulaire que l'on va créer va permettre de recueillir les informations suivantes données par le visiteur de la page :



La structure html prévue à cet effet est un bloc dont le nom de la balise est *<form>* comme ….

```html
<!DOCTYPE html>
<html lang="fr">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Chez moi</title>
        <link rel="icon"  type="img/png" href="poste.png"/>
        <link rel="stylesheet" href="formulaire.css">
    </head>
    <body>
        <form action="">


        </form>
    </body>
</html>
```

2. Démarrage et méthode GET

⇨ Compléter le script comme ci-dessous :

```
<body>
    <form action="" method="GET">
        <input type="text" name="objet">
        <input type="submit" value="Envoyer" name="bouton">
    </form>
</body>
```

⇨ Dans le navigateur saisir ***bonjour*** dans le champ de saisi, et cliquer sur *Envoyer*

⇨ Observer ce qui a changé dans la barre d'adresse URL du navigateur :

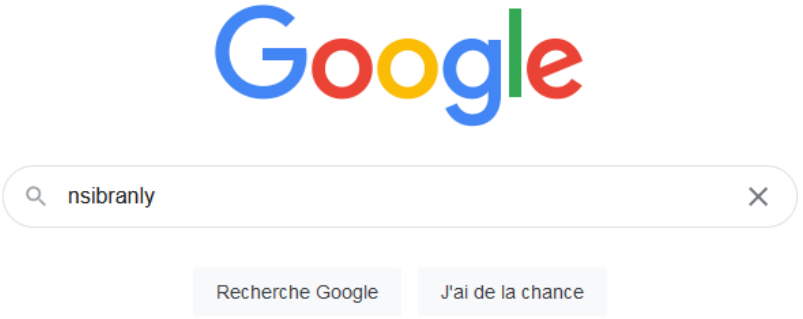
..... /formulaires.html?objet=bonjour&bouton=Envoyer

Résultat : En cliquant sur le bouton, on constate que les informations saisies (pour le bouton ce sera la value 'envoyer') des 2 éléments du bloc <*form*> sont stockées dans des variables qui reprennent le nom de l'attribut **name** des balises et le tout est rajouté à la fin de l'adresse de la page web :

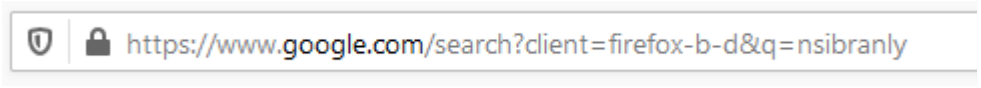
..... /formulaires.htm?objet=bonjour&bouton=Envoyer

Effectivement, pour transmettre des informations au serveur, on utilise souvent un protocole de requête ***https*** (Hyper-Text-Transfer-Protocol-Secure) nommé ***GET*** qui écrit les données à transmettre au serveur, directement dans l'URL.

C'est ce protocole qui est par exemple utilisé par Google pour transmettre les éléments de votre recherche à leur serveur. Par exemple, en effectuant la recherche :

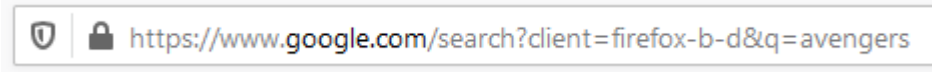


On retrouve dans la barre d'adresse :



⇨ Copiez cette URL dans le presse-papier (*Ctrl C*), rafraichissez la page google.com et collez cette adresse copiée dans la barre d'adresse …… vous retrouverez le résultat de la recherche précédente.

⇨ Remplacer **directement dans l'URL** le mot *nsibranly* par *avengers* ……. vous aurez un résultat correspondant à la recherche liée au mot clé avengers …



3. Attributs des 2 balises <input> :

Les 2 éléments input se positionnent sur la même ligne dans le navigateur : 

Questions : ⇨ S'agit-il d'éléments *inline* ou d'éléments *bloc* ?

⇨ La balise <input> est-elle orpheline ?

⇨ Rajouter les attributs *placeholder* et *autofocus* dans la balise <*input*> de type *text*

Question : Quel est l'effet constaté sur la page affichée ?

```
<form action="" method="GET">
    <input type="text" name="objet" placeholder="Objet" autofocus >
    <input type="submit" value="Envoyer" name="bouton">
</form>
```

⇨ Modifier la valeur de l'attribut *value* de la balise input de type *submit*.

Question : Quel est l'effet constaté sur la page affichée ?

Remarque importante : Les attributs *type* et *name* sont les plus importants. Il ne s'agit pas de les modifier. La valeur des attributs *name* sera utilisée sur le serveur pour reconnaître les données reçues :

Il est possible de rajouter d'autres attributs. Par exemple :

```
<input type="text"  name="objet" placeholder="Objet"  minlength="4" maxlength="8" size="10" required>
```

Mais pour notre messagerie, les attributs supplémentaires (*minlenght, required*) ne sont pas le bienvenu car un message peut bien être envoyé avec un *objet* vide. L'attribut *maxlenght* serait utile pour limiter la longueur de la chaine de caractère saisie.

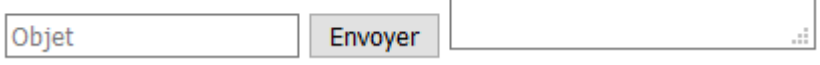
4. Balise <textaera> :

La balise <input type='text'> permet de saisir une phrase. Elle nous permettra de saisir l'objet du message que l'on souhaite envoyer. Au contraire, la balise <textarea> permet de saisir un texte de plusieurs lignes. Elle nous permettra de saisir le contenu du message à envoyer.

⇨ Rajouter une balise de ce type dans votre script :

```
<form action="" method="GET">
    <input type="text" name="objet" placeholder="Objet">
    <input type="submit" value="Envoyer" name="bouton">
    <textarea name="message"></textarea>
</form>
```

Ce qui provoquera l'affichage suivant dans votre navigateur :



Questions : ⇨ <textaera> est-il un élément *inline* ou un élément *bloc* ?

⇨ La balise < textaera > est-elle orpheline ?

⇨ Saisir le texte suivant dans le champ de saisi de cette balise et cliquer sur le bouton :

> Bonjour,
> ma grand-mère m'a dit que le mois de mai était le plus beau de l'année.
> A-t-elle toujours raison ?

Les 3 données envoyées qui ont comme attribut *name* respectifs : *objet*, *bouton* et *message*, s'écrivent-elles dans l'URL ? :

/formulaires.html?objet=&bouton=Envoyer&message=Bonjour%2C%0D%0Ama+grand-mère+m'a+dit+que+le+mois+de+mai+était+le+plus+beau+de+l'année.+%0D%0AA-t-elle+toujours+raison+%3F%0D%0A

⇨ Sur le navigateur, pouvez-vous agrandir, la zone texte du *texarea* en agissant avec la souris sur le coin inférieur droit ?

⇨ Si le texte saisi est trop grand, une barre de défilement se met-elle en place ?

On peut rajouter des attributs supplémentaires à cette balise. Pour notre application, on limitera la saisie à 1000 caractères et on imposera la taille de la zone de saisie (modifiable ensuite avec le CSS).

```
<textarea name="message" rows="5" cols="40" maxlength="1000" placeholder="Message"></textarea>
```

5. Balise <select> :

La balise <select> permet à l'internaute de choisir un item dans une liste prédéfinie. On l'utilise ici pour pouvoir choisir le destinataire du message que l'on souhaite envoyer. Compléter votre script de la façon suivante :

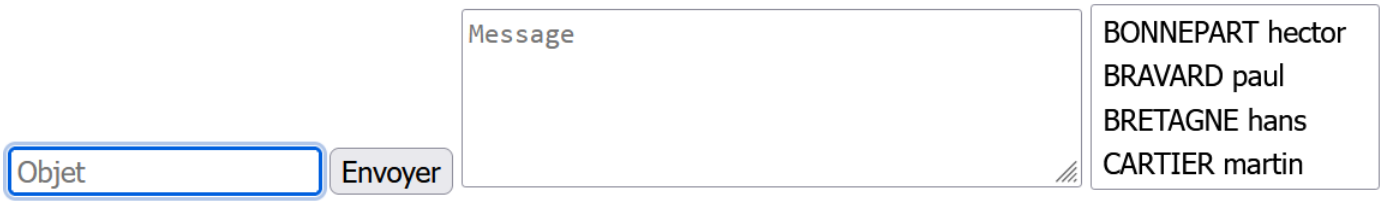
Pour éviter de tout taper, ouvrir le fichier *form.html* téléchargé en début de tp. Il vous donne le contenu entre les balises *<select></select>* . Faites un copié-collé dans votre fichier *html*.

```
<select name="destinataire[]" multiple>
    <option value="hectorB">BONNEPART hector</option>
    <option value="paulB">BRAVARD   paul</option>
    <option value="hansB">BRETAGNE hans</option>
    <option value="martinC">CARTIER martin</option>
    <option value="matheoC">CHOULET matheo</option>
    <option value="thomasD">DEGREMONT thomas</option>
    <option value="guillaumeD">DUPIN guillaume</option>
    <option value="yahelG">GUILLAUME yahel</option>
    <option value="raidenH">HAMZA raiden</option>
    <option value="yannJ">JOUBERT yann</option>
    <option value="agathaJ">JOURDAN agatha</option>
    <option value="eliasR">REZZOUG elias</option>
    <option value="malikS">SEGHIR malik</option>
    <option value="antoineS">SELLEM antoine</option>
    <option value="quentinT">TJAMPENS quentin</option>
</select>
```

Sur un destinataire, par exemple :

```
<option value="hectorB">BONNEPART hector</option>
```

l'attribut *value* **ne doit pas être modifié**. Par contre, pas de problème pour remplacer BONNEPART hector par un surnom affectueux : MonChéri



Dans votre navigateur, il s'affichera la page :

Comme l'attribut *multiple* a été rajouté dans la balise <select>, il est possible de sélectionner plusieurs items en même temps. A cet effet, le *name* attribué à cette balise devient une liste : `name="destinataire[]"`

Questions :

⇨ <select> est-il un élément *inline* ou un élément *bloc* ?

⇨ La balise < select > est-elle orpheline ?

⇨ Saisir les 2 premiers items de la liste et cliquer sur le bouton.

Dans l'URL, sont écrites les valeurs des 4 données envoyées.
Parmi celles-ci, combien ont une valeur nulle ?

```
/formulaire.html?objet=&bouton=Envoyer&message=&destinataire[]=hectorB&destinataire[]=paulB
```

⇨ Est-ce-que c'est la valeur de l'attribut *value* qui est envoyée ?

```
<option value="hectorB">BONNEPART hector</option>
<option value="paulB">BRAVARD   paul</option>
```

⇨ Rajouter l'attribut *size* qui limite le nombre d'items visibles et *required* qui n'accepte pas l'envoie si aucun item a été sélectionné *(on peut envoyer un message vide, mais difficile d'en envoyer s'il n'y a pas de destinataire …)*

```
<select name="destinataire[]"  size="3" multiple required>
```

⇨ Dans Visual Studio Code, il est possible pour y voir plus clair, de réduire l'affichage entre les balises <select> et </select>. Pour cela, cliquer sur la flèche ˅

```
<select name="destinataire[]" multiple>
    <option value="hectorB">BONNEPART hector</option>
    <option value="paulB">BRAVARD   paul</option>
    <option value="hansB">BRETAGNE hans</option>
    <option value="martinC">CARTIER martin</option>
    <option value="matheoC">CHOULET matheo</option>
    <option value="thomasD">DEGREMONT thomas</option>
```

On obtient ainsi :

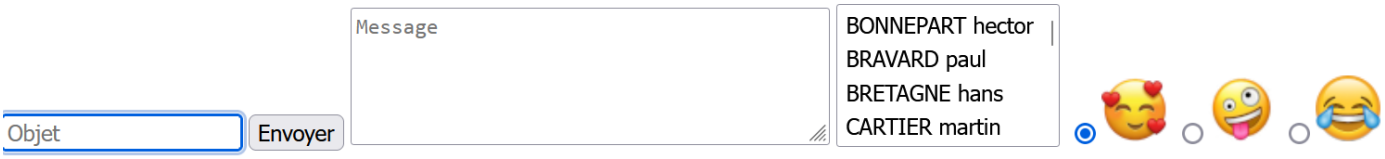
```
<select name="destinataire[]" multiple>···
</select>
```

6. Balise <input type= 'radio'> :

La balise <input> de type 'radio' permet de réaliser un choix en cochant des cases. On se sert de cette structure pour sélectionner l'image de l'emoji que l'on associera au message envoyé. Compléter votre script de la façon suivante :

```
<form action="" method="GET">
    <input type="text" name="objet" placeholder="Objet">
    <input type="submit" value="Envoyer" name="bouton">
    <textarea name="message" rows="5" cols="40" maxlength="1000" placeholder="Message"></textarea>
    <select name="destinataire[]"  size="3" multiple required>···
    </select>
    <input type="radio"  name="emoji" value="amour.png"><img src="amour.png">
    <input type="radio"  name="emoji" value="normal.png" ><img src="normal.png">
    <input type="radio"  name="emoji" value="hilare.png" ><img src="hilare.png">
</form>
```

Les 3 fichiers images se trouvent normalement déjà dans le répertoire de votre script. Dans votre navigateur, il s'affichera la page :

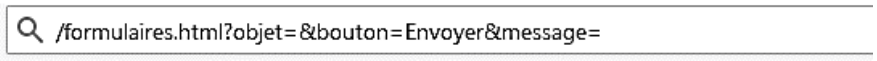


L'attribut *name* de ces 3 balises <input> est le même. Il est égal à '*emoji*'. Seul 1 seul émoji pourra être sélectionner. Ainsi la valeur de la donnée envoyée sera repérée par le name '*emoji*' et sera égale ici soit à '*amour*', soit à '*normal*', soit à '*hilare*'.

Questions : ⇨ <input type='*radio*'> est-il un élément *inline* ou un élément *bloc* ?

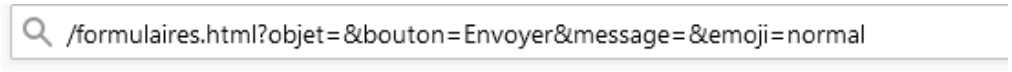
⇨ La balise <input type='*radio*'> est-elle orpheline ?

⇨ Sans rien cocher, rien sélectionner dans la liste, rien écrire, cliquer sur le bouton. La barre d'URL devient :



Cela signifie-t-il que les données envoyées sous les *name objet* et *message* sont des strings vides et que celles envoyées sous les *name destinataire[]* et *emoji* n'ont même pas été générées ?

⇨ Cocher  , cliquer sur le bouton et observer l'URL …..
La donnée envoyée sous le name emoji a-t-elle été générée ? Quelle est sa valeur ?



Améliorations : On rajoute l'attribut *checked* sur le 1er <input> afin de sélectionner par défaut un émoji. Sur smartphone, cocher précisément avec ses doigts la bonne zone n'est pas évident. Afin d'étendre cette zone à l'image entière, on encapsule l'image dans une balise <label> que l'on lie avec la balise <input type='radio'> par un *id* qui prend une valeur commune :

```
<input type="radio"  name="emoji" value="amour.png" id="emoji_1" checked ><label for="emoji_1"><img src="amour.png"></label>
<input type="radio"  name="emoji" value="normal.png" id="emoji_2" ><label for="emoji_2"><img src="normal.png"></label>
<input type="radio"  name="emoji" value="hilare.png" id="emoji_3" ><label for="emoji_3"><img src="hilare.png"></label>
```

⇨ Tester cette modification et l'étendre aux 2 autres émoji.

⇨ Reprendre le fichier *index.html* qui est actuellement sur votre espace sur *nsibranly.fr*, soit par un clic droit en faisant « *Afficher code source de la page* » puis copiez-collez dans un nouveau fichier nommé *index.html*, soit en reprenant le fichier *index.html* qui est déjà sur le site dans votre espace.

⇨ Dans ***index.html***, ajouter un lien vers le fichier *formulaire.html*
Ajouter également un lien qui permet de revenir sur la page *web* du site *nsibranly.fr*

```
<body>
    <ul>
        <a href="blague.html">   <li>Ma blague à 2 balles</li>    </a>
        <a href="formulaire.html">   <li>Formulaire</li>    </a>
        <a href="https://www.nsibranly.fr/">   <li>Retour Nsi Branly</li></a>
    </ul>
</body>
```

⇨ Dans ***formulaire.html***, ajouter un lien vers le fichier *index.html*

```
<body>
    <form action="../formulaires.php" method="POST">
        <input type="text" name="objet" placeholder="Objet" autofocus >
        <input type="submit" value="Envoyer" name="bouton">
        <textarea name="message"  cols="40" rows="5" placeholder="Message"></textarea>
        <select name="destinataire[]" multiple>···
        </select>
        <input type="radio"  name="emoji" id="emoji_1" value="amour.png" checked><label for="emoji_1"><img src="amour.png"></label>
        <input type="radio"  name="emoji" id="emoji_2" value="normal.png"><label for="emoji_2"><img src="normal.png"></label>
        <input type="radio"  name="emoji" id="emoji_3" value="hilare.png"><label for="emoji_3"><img src="hilare.png"></label>
    </form>
    <a href="index.html"><button>Retour sur ma page</button></a>
</body>
```

⇨ La balise <*form*></*form*> permet d'obtenir un élément de type *inline* ou de type *bloc* ?

⇨ Tester le bon fonctionnement des liens en local.

7. Premier test de la page :

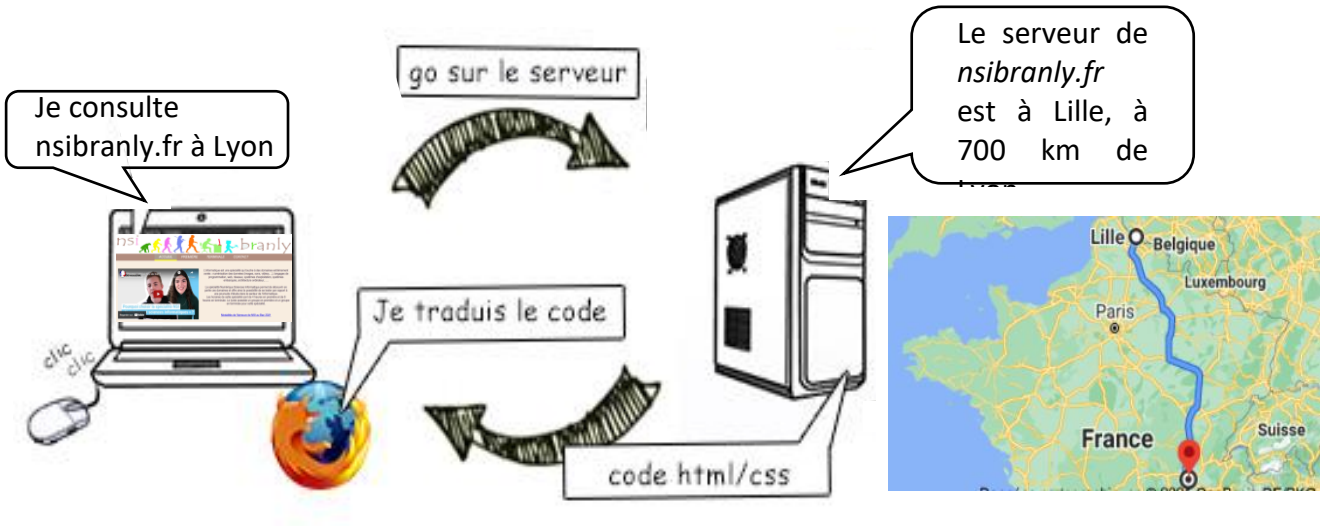
Votre script ne vous permet pas d'afficher une page sensationnelle dans votre navigateur. Il a néanmoins le mérite d'être fonctionnel. On va le tester dès à présent en le transférant ici dans votre espace sur le site https://nsibranly.fr …. mais pas si vite, il y a encore quelques manipulations à faire …

7.1. Comment envoie-t-on réellement les données sur le serveur :

Le formulaire mis au point permet simplement de recueillir les données que l'expéditeur veut envoyer à ses camarades de classe. Ces données sont :

- L'objet du message contenu dans une variable dont le nom est : *objet*
- Le texte du message contenu dans une variable dont le nom est : *message*
- Le ou les destinataires contenus dans une liste dont le nom est : *destinataire[]*
- Le nom d'un emoji contenu dans une variable dont le nom est : *emoji*

Ces données vont être numérisées **par le navigateur** en suivant une méthode de requête **GET** ou **POST**. Ces 2 méthodes font parties d'un protocole nommé « *https » (* **H**yper **T**ext **T**ransfer **P**rotocole **S**ecure) inventé dans les années 1990. Il permet le transfert de pages web entre le serveur et le client (l'internaute).



Avec la méthode ***GET***, les données saisies par l'internaute sont écrites dans l'URL et sont transférées ensuite avec cette URL, lorsque l'on clique sur le bouton <input type='submit'>.

C'était bien pratique jusqu'à présent dans ce Tp pour avoir une idée du contenu des données envoyées. Cette méthode GET présente néanmoins 2 inconvénients majeurs :

- Les données sont visibles dans l'URL, ce qui est problématique par exemple lorsque l'on transfert un mot de passe, ….
- Les données transmises doivent être des strings (pas de transfert possible de fichiers binaires : images, vidéos) et la taille des données est limitée.

La seconde méthode de transfert que propose le protocole ***https***, s'appelle ***POST*** et ne présente pas ces inconvénients. On l'utilisera ici pour réaliser notre messagerie.

7.2. Oui, mais concrètement que fait-on ? :

1 ⇨ Modifier le nom de la méthode dans la balise <*form*> en remplaçant GET par ***POST*** ..

2 ⇨ Ajouter dans la balise <*form*> , la commande qui sera exécutée sur le serveur lorsque les données que l'on envoie arrivent ….

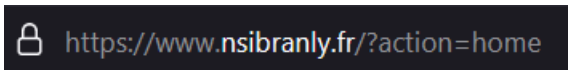
```
<form action="/index.php?action=home" method="POST">
```

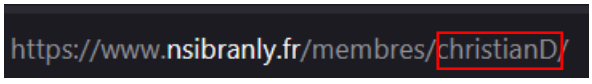
Après réception des données sur le serveur, ici le fichier ***index.php*** du site *nsibranly.fr* sera exécuté avec une donnée Get nommée 'action' et dont la valeur est 'home'. Un code déjà écrit sur le serveur (en langage php) s'exécute alors et permet de récupérer les données envoyées par le formulaire pour les écrire dans une Base De Données (bdd) qui se trouve aussi sur ce serveur à Lille. Cette bdd est une sorte de gros fichier qui permet de conserver tous les messages envoyés par chacun d'entre vous. Les données reçues sont reconnues par rapport à leur nom qui est celui de l'attribut **name** écrit dans chaque balise. C'est pour cela qu'il faut bien faire attention à cet attribut qui est très important.

D'autres scripts, écrits aussi en php, permettent ensuite de récupérer dans cette bdd les messages qui vous sont destinés, pour les écrire en *html* dans la page web de votre *bal*, sur nsibranly.fr . Ces choses là seront vues en classe de terminale …..

3 ⇨ Sur le site nsibranly.fr, se loguer et transférer avec le code *web* vos fichiers *index.html, formulaire.html*, *formulaire.css* (vide pour l'instant) et les fichiers images *poste.png, amour.png, normal.png, hilare.png* .

4 ⇨ Aller sur votre fichier *index.html* qui se trouve dans votre zone personnelle :

- Soit en cliquant sur votre avatar dans l'onglet web de nsibranly.fr,
- Soit, si vous êtes connecté, écrivant dans l'URL : 

en
- Soit, si vous n'êtes pas connecté, en écrivant dans l'URL : 

Votre répertoire s'appelle :

prénom + 1ère lettre de votre nom de famille en Majuscule

4 ⇨ Aller sur votre fichier *formulaire.html* .

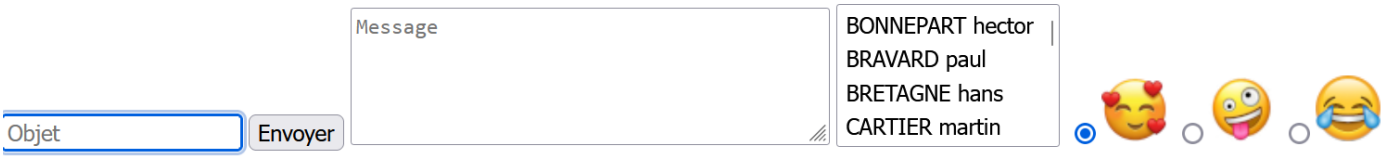
5 ⇨ Vous pouvez envoyer un message à vous-même pour commencer. Les messages sont à consulter dans l'onglet *bal* de *nsibranly.fr* .

8. Mise en forme de votre formulaire en ajoutant du CSS :

L'objectif de cette partie est d'obtenir grâce au Css, un formulaire digne de ce nom.

On retravaillera à présent en mode local à partir du fichier *formulaires.html* qui se trouve sur votre ordinateur. Ce n'est qu'à la fin de ce travail que l'on transfèrera les fichiers modifiés sur le serveur.

L'objectif est de créer le fichier formulaires.css qui permette d'arriver à obtenir un formulaire qui part de :



A quelque chose de bien plus joli.