

## TP JS 6 - Manipulation statique du DOM

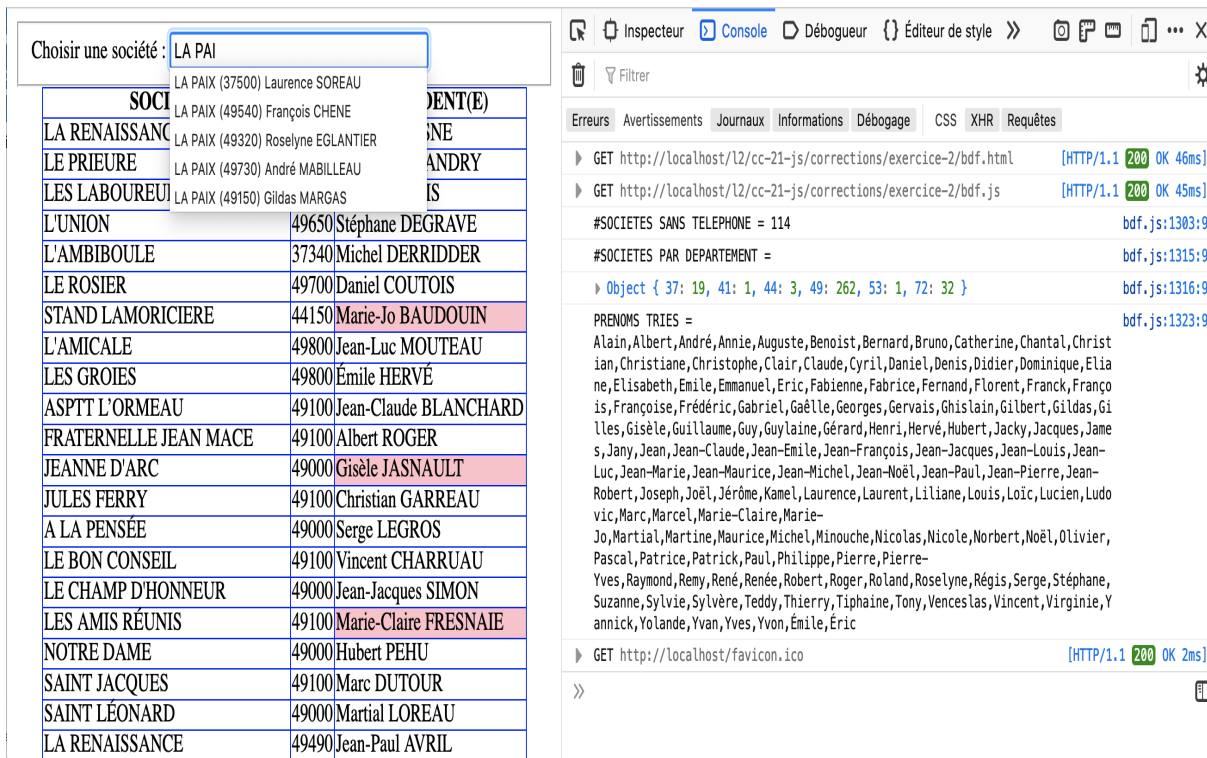
Décompressez l'archive déposée sur Moodle pour ce TP. Le dossier résultant contient différents fichiers à réutiliser ou à compléter. Pensez à consulter le site [MDN](#), et pour visualiser ce qui est attendu, ce [démonstrateur](#).

### Exercice 1. Boules de fort

Le fichier **bdf.html** importe le fichier **bdf.js** qui définit :

- La variable **societes** qui est un tableau de tableaux, chacun stockant le nom d'une société de boules de fort, son numéro de téléphone, code postal, commune et président (ou présidente). Chaque président est identifié par son prénom suivi de son nom.
- La variable **prenoms\_feminins** qui est un tableau de prénoms féminins.
- La variable **departements** qui est un tableau de numéros de départements français.
- La méthode de tableaux **supprimerDoublons()** qui renvoie une copie du tableau sur lequel elle est invoquée en en supprimant les doublons.

Complétez **bdf.js** pour obtenir la page illustrée en Figure 1.



The screenshot shows a web application with a form to choose a company and a table of companies. The browser console shows the following logs:

```

GET http://localhost/l2/cc-21-js/corrections/exercice-2/bdf.html [HTTP/1.1 200 OK 46ms]
GET http://localhost/l2/cc-21-js/corrections/exercice-2/bdf.js [HTTP/1.1 200 OK 45ms]
#SOCIETES SANS TELEPHONE = 114 bdf.js:1303:9
#SOCIETES PAR DEPARTEMENT = bdf.js:1315:9
Object { 37: 19, 41: 1, 44: 3, 49: 262, 53: 1, 72: 32 } bdf.js:1316:9
PRENOMS TRIES = bdf.js:1323:9
Alain,Albert,André,Annie,Auguste,Benoist,Bernard,Bruno,Catherine,Chantal,Christ
ian,Christiane,Christophe,Clair,Claude,Cyril,Daniel,Denis,Didier,Dominique,Elia
ne,Elisabeth,Emile,Emmanuel,Eric,Fabienne,Fabrice,Fernand,Florent,Franck,Franço
is,Françoise,Frédéric,Gabriel,Gaëlle,Georges,Gervais,Ghislain,Gilbert,Gildas,Gi
lles,Gisèle,Guillaume,Guy,Guyaine,Gérard,Henri,Hervé,Hubert,Jacky,Jacques,Jame
s,Jany,Jean,Jean-Claude,Jean-Emile,Jean-François,Jean-Jacques,Jean-Louis,Jean-
Luc,Jean-Marie,Jean-Maurice,Jean-Michel,Jean-Noël,Jean-Paul,Jean-Pierre,Jean-
Robert,Joseph,Joël,Jérôme,Kamel,Laurence,Laurent,Liliane,Louis,Loïc,Lucien,Ludo
vic,Marcel,Marie,Marie-Claire,Marie-
Jo,Martial,Martine,Maurice,Michel,Minouche,Nicolas,Nicole,Norbert,Noël,Olivier,
Pascal,Patrice,Patrick,Paul,Philippe,Pierre,Pierre-
Yves,Raymond,Remy,René,Renée,Robert,Roger,Roland,Roselyne,Régis,Serge,Stéphane,
Suzanne,Sylvie,Sylvère,Teddy,Thierry,Tiphaine,Tony,Venceslas,Vincent,Virginie,Y
annick,Yolande,Yvan,Yves,Yvon,Émile,Éric
GET http://localhost/favicon.ico [HTTP/1.1 200 OK 2ms]

```

FIGURE 1 – Page web et affichage en console attendus

- Calculez le nombre de sociétés dont le numéro de téléphone est la chaîne **xx-xx-xx-xx-xx**. Stockez le résultat dans la variable **sans\_telephone**.
- Construisez l'objet **distribution** qui associe à chaque numéro de département figurant dans le tableau **departements** le nombre de sociétés de ce département (les deux premiers chiffres du code postal d'une société correspondent à son numéro de département).

3. Extrayez du tableau `societes` les prénoms des présidents (ou présidentes), supprimez les doublons et triez les en ordre alphabétique. Stockez le résultat dans le tableau `prenoms`.
4. Le fichier HTML contient un tableau à 3 colonnes libellées "SOCIETE", "CP" et "PRESIDENT(E)". Complétez-le en ajoutant une ligne par société avec les champs attendus. Si le président d'une société a l'un des prénoms féminins enregistrés dans le tableau `prenoms_feminins`, attribuez à la case HTML correspondante la classe `presidente` qui coloriera automatiquement le fond de la cellule en rose via les règles CSS. Utilisez la propriété `classList` et les méthodes `insertRow` et `insertCell`.
5. Le code JS fourni pour cette question construit un objet avec le constructeur `Societe` pour chaque société et invoque sur cet objet la méthode `toHTML` en lui passant la **data-list HTML** d'identifiant `societes`. Une fois ce code exécuté, le champ texte de la page web suggère différentes options au visiteur dès qu'il saisit des caractères (voir Figure 1). Implémentez le constructeur `Societe` qui stocke pour chaque objet créé le nom de la société, son code postal et son président. Ajoutez ensuite, sans modifier le constructeur, la méthode `toHTML(datalist)` qui, lorsqu'elle est invoquée sur un objet `Societe` avec une data-list HTML en argument :
  - Ajoute à cette data-list un sous-élément de balise `option`.
  - Initialise l'attribut HTML `value` de cette option avec le texte concaténant le nom, code postal et nom de président stockés dans l'objet.

## Exercice 2. TUX

Le fichier `tux.html` importe `tux.js`. Complétez ce dernier pour obtenir la page illustrée en Figure 3.



FIGURE 2 – Page avant exécution JS

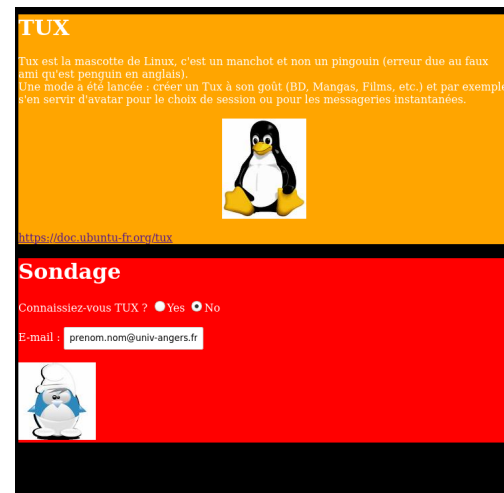


FIGURE 3 – Page après exécution JS

1. Modifiez la couleur d'arrière plan du corps du document (`<body>`) en noir. Modifiez les couleurs d'arrière plan des divisions (`<div>`) d'identifiant `intro` et `sondage`, respectivement, en orange et en rouge. Modifiez la couleur des éléments de chaque division du document en blanc.
2. Centrez toutes les images de la division d'identifiant `intro`. Pour ce faire, vous pouvez utiliser la classe `center` définie dans `tux.html`.
3. Dans la division `sondage` :
  - Tous les champs de saisie d'un email doivent prendre comme valeur `nom.prenom@univ-angers.fr`
  - Toutes les cases de valeur `non` doivent être cochées.
  - Les labels de toutes les cases de valeur `oui` et `non` doivent être, respectivement, `Yes` et `No`.
4. Ajoutez pour chaque image une description textuelle (`alt`) indiquant où se situe l'image (arborescence DOM jusqu'à l'élément `body`). Par exemple pour la première image : `alt="Image - P (parIntro) - DIV`

`(intro)"` indique que l'image est dans le paragraphe (P) d'identifiant `parIntro` qui est dans la division (DIV) d'identifiant `intro`.