**TP1 : IMC en Javascript**

**Contexte**

On se propose de réaliser un système simple qui permet de calculer l’indice de masse corporel.

L’indice de masse corporel est donné par la formule suivante :

**IMC = poids(kg)/taille(m)2**

Selon la valeur de IMC :

|  |  |  |
| --- | --- | --- |
| 18.5 ≤ IMC < 25 | 25 ≤ IMC < 30 | 30 ≤ IMC |
| Normal | Surpoids | Obésité |

* Ouvrir le fichier calcul-imc-javascript.html avec le bloc note ou l'éditeur Notepad++,
* Ajouter les balises script en fin de page avant le body fermant:

*<script type='text/javascript' language='javascript'>  
  
</script>*

*</body>  
</html>*  
  
Nous devons commencer par déclarer les variables permettant de stocker le poids et la taille de l'internaute. Nous allons en profiter pour déclarer deux autres variables, la première pour stocker le calcul de l'IMC et la seconde pour mémoriser la position de l'élément à afficher sur la graduation.

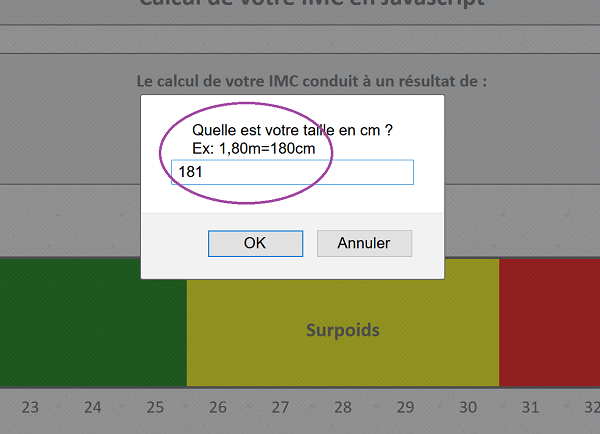
* En conséquence, déclarer les variables suivantes, entre les balises script :

*var taille; var poids; var imc; var indicateur;*  
  
La **fonction Javascript Prompt** retourne la valeur saisie par l'utilisateur, au travers de la boîte de dialogue qu'elle génère. Il nous en faut deux, une pour la question sur la taille et une autre pour la question sur le poids.

* A la suite du code, ajouter les deux lignes suivantes :

*var taille = prompt('Quelle est votre taille en cm ?\nEx: 1,80m=180cm');  
var poids = prompt('Quel est votre poids en Kg ?');*  
  
Chaque boîte de dialogue doit afficher le message qui lui est passé en paramètre.

* Enregistrer les modifications et basculer sur le navigateur,
* Enfoncer la touche F5 du clavier pour rafraîchir la page Web,
* Saisir une taille en cm à l'invite de la première boîte de message puis valider,
* Saisir un poids dans la seconde et valider,

  
  
Les boîtes de dialogue disparaissent à tour de rôle et l'exécution du code s'arrête. En effet à ce stade, nous avons bien réceptionné les valeurs tapées par l'internaute, mais aucun code ne suit pour les traiter. Constatez, comme l'illustre la capture ci-dessus, la présentation de la question sur deux lignes grâce aux caractères d'échappement insérés dans le flux du paramètre.   
  
**Vérifier la validité des données**  
  
La **fonction Javascript parseInt()** renvoie l'entier de la chaîne qui lui est passée en paramètre. Si la valeur retournée n'est pas définie, nous saurons que l'utilisateur n'a pas saisi un nombre. La **fonction Javascript isNaN()** permet justement de savoir si la valeur retournée et passée en paramètre est non définie ou non représentable. Par exemple, si le test : *if(isNaN(parseInt(taille)))* renvoie True, alors la taille fournie n'est pas correcte.

* A la suite du code, ajouter les bornes de l'instruction conditionnelle comme suit :

*if(!isNaN(parseInt(taille)))  
{  
}  
else  
{  
alert('La taille saisie est incorrecte !\nVeuillez réactualiser par la touche F5');  
}*

* Enregistrer les modifications et basculer sur le navigateur,
* Enfoncer la touche F5 pour rafraîchir la page et exécuter le nouveau code,
* Dans la boîte de dialogue pour la taille, saisir un texte : essai par exemple,
* Valider puis saisir un poids et valider de nouveau.

Comme vous le constatez, l'erreur est bien interceptée par le test de l'instruction conditionnelle réalisé par notre code. La partie entière retournée pour une chaîne de caractères n'est pas définie, donc le traitement pour le calcul de l'Imc ne peut pas se poursuivre.  
  
En revanche, si vous relancez la page et que vous tapez un nombre comme 180, aucune alerte ne se déclenche et le code se termine, puisqu'aucun traitement n'est prévu pour l'instant.  
  
  
Avant d'établir la même vérification sur le poids, nous souhaitons un critère plus draconien sur la taille. Pour des raisons de logique, nous souhaitons que la taille soit comprise entre 120 et 250cm, soit 1,20m et 2,50m. La syntaxe permettant de recouper des critères et exprimer le **ET**(AND en VBA), est la séquence suivante : **&&**. Le **OU** quant à lui s'écrit de la sorte : **||**, soit deux barres verticales accolées.

* Revenir dans l'éditeur de code,
* Modifier l'instruction If comme suit :

*if(!isNaN(parseInt(taille)) && taille>120 && taille<250)*  
  
Pour que le traitement démarre, nous demandons à vérifier trois critères ensemble grâce aux recoupements du ET (&&). La partie entière doit être définie et en même temps le nombre doit à la fois être supérieur strictement à 120 et inférieur strictement à 250.  
  
Si vous refaites un test, et que vous tapez cette fois-ci une valeur numérique comme 90, l'issue est la même que précédemment, mais en raison du deuxième critère non vérifié.  
  
Nous devons tester des règles identiques sur la saisie du poids. Nous pourrions continuer de recouper les critères dans la même instruction If. Mais comme nous souhaitons adresser un message spécifique en cas de saisie erronée pour ce dernier, nous choisissons de le construire dans la branche du If.

* Entre les accolades de l'instruction If, ajouter le test suivant :

*if(!isNaN(parseInt(poids)) && poids>20 && poids<500)  
{  
}  
else  
{  
alert('La taille saisie est incorrecte !\nVeuillez réactualiser par la touche F5');*

*}*  
  
Si la partie entière du poids est bien définie et que ce dernier est à la fois strictement supérieur à 20 et strictement inférieur à 500, alors nous considérons que la saisie est valide. Dans le cas contraire (else), nous adressons un message (alert) à l'internaute pour l'en informer.

* Enregistrer les modifications, basculer sur le navigateur et rafraîchir la page,
* Saisir une taille conforme : 180 par exemple et valider,
* Saisir un texte, par exemple essai  : à la place du poids, et valider,

Comme précédemment, le code intercepte l'erreur mais grâce à la seconde instruction conditionnelle cette fois. En revanche, lorsque les deux saisies sont numériques et satisfont à l'ensemble des critères, aucune alerte ne se déclenche, mais le code se termine. En effet, aucun traitement n'est encore prévu en cas de succès, entre les accolades du second If. 

**Calcul de l'IMC**  
Si ce deuxième test réalisé par cette seconde instruction if est vérifié, comme le premier l'est aussi, alors nous pouvons enclencher les instructions de calcul pour l'IMC. Ces lignes de codes doivent intervenir entre les accolades de la deuxième instruction If. Nous pourrons alors restituer la valeur dans le calque prévu à cet effet, directement sur la page HTML. Puis il s'agira d'afficher le curseur (**balise Html img**) au-dessus de la graduation pour la représentation graphique.

* Revenir dans l'éditeur de code entre les accolades du second If,

Comme nous l'avons énoncé plus haut, le calcul de l'indice de masse corporelle consiste à diviser le poids de la personne par la taille au carré. Ce résultat doit être grossi 10000 (\*10 000) fois pour être ramené à l'échelle de valeurs. Le résultat obtenu sera un nombre réel que nous stockerons dans la **variable imc**, déclarée à cet effet.  
  
Pour la représentation graphique, nous devons conserver un nombre sans décimale. Nous pourrions exploiter la fonction parseInt() mais la **méthode round()** de l'**objet Javascript Math** permet d'arrondir à l'entier le plus proche. Ainsi 24,4 conduit à 24 tandis que 24,6 conduit à 25.

* Entre les accolades du second If, ajouter les quatre instructions suivantes :

*imc = (poids \* 10000) / (taille\*taille);  
alert(imc);  
imc = Math.round(imc);  
alert(imc);*

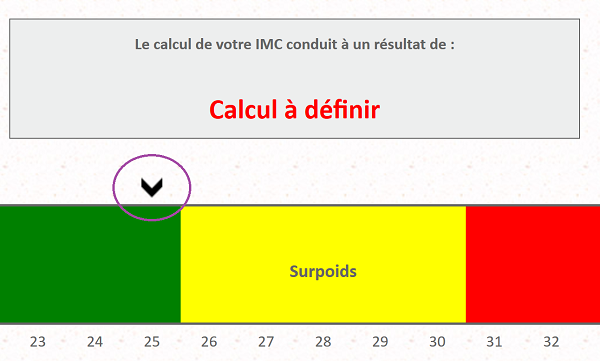
* Enregistrer les modifications, basculer sur le navigateur et rafraîchir la page,
* Saisir 180 pour la taille par exemple et valider,
* Saisir 80 pour le poids par exemple et valider,

Le premier résultat affiché par la **fonction alert** conduit en effet à un nombre réel (24,691) avec beaucoup de décimales. Après traitement par la **méthode round** de l'**objet Math**, le second alert fournit le nombre entier le plus proche, 25 ici.   
  
**Interactions HTML**  
Nous devons exploiter ce résultat du calcul de l'IMC visuellement sur la page Web. Il faut afficher l'icône de l'image correspondant à la graduation. Chaque image possède un identifiant incrémenté (img1,img2 etc...). Mais la graduation ne commence pas à 1, elle commence à 18. Ainsi si l'IMC vaut 18, nous devons afficher l'image dont l'identifiant est img(18-17), soit img1.  
  
C'est la raison pour laquelle nous avions déclaré la variable indicateur. Elle doit faire correspondre le résultat de l'Imc avec l'identifiant de l'image en place sur la graduation : indicateur = imc - 17. Dès lors, nous pourrons désigner précisément l'image à afficher par le **code Javascript**. Comme nous l'avons appris dans la [formation pour débuter la programmation en Javascript](https://www.bonbache.fr/debuter-la-programmation-web-en-javascript-220.html), c'est la **méthode getElementById()** de l'**objet document** qui permet de faire référence à un élément Html par son Id. Et c'est ensuite la **propriété style** qui permet d'accéder aux attributs à régler, ici en l'occurrence **visibility**.

* A la suite du code, ajouter les deux instructions suivantes :

*indicateur = imc-17;  
document.getElementById('img' + indicateur).style.visibility = 'visible';*  
  
Nous accédons à l'**objet img** en désignant son identifiant grâce à la concaténation de la chaîne fixe ('img') et de la valeur de l'indicateur ramenée à la position correspondante (+ indicateur). Nous réglons la **propriété visibility** de l'image ainsi désignée sur **visible**, pour la faire apparaître sur la page Web, de façon dynamique.

* Passer les deux alert précédents en commentaires (//),
* Enregistrer les modifications, basculer sur le navigateur et rafraîchir la page,
* Saisir une taille et valider, puis saisir un poids et valider,

  
  
Comme vous le constatez, le petit curseur correspondant à la valeur de l'IMC calculée et ramenée à un nombre entier, s'affiche sur la représentation graphique. Le résultat est plus clair et plus plaisant pour l'internaute.

En l'état, ce calque ne peut donc pas être piloté par le **code Javascript**. Nous devons commencer par lui attribuer un identifiant.

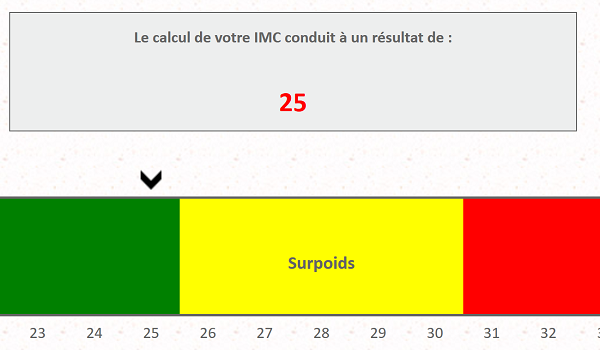
* Repérer la ligne qui contient « calcul à définir » et ajouter l'identifiant res\_imc comme suit :

*<div style='height:40px; width:100%; font-size:24px; font-weight:bold; color:red;'****id=' res\_imc'****>Calcul à définir</div>*

* Puis à la suite du code précédent, dans la seconde instruction If, ajouter la ligne suivante :

*document.getElementById('res\_imc').innerText = imc;*  
  
Comme vous le constatez, c'est la **propriété innerText** d'un élément HTML ainsi désigné qui permet d'accéder à son contenu brut. Il existe aussi la propriété innerHtml qui permet de modifier le contenu du balisage interne de l'élément.

* Enregistrer les modifications, basculer sur le navigateur et rafraîchir la page,
* Saisir une taille et valider, puis saisir un poids et valider,

  
  
En même temps que nous affichons l'indicateur visuel de position, nous parvenons à inscrire le résultat numérique du calcul, dans le calque précisément désigné par le **code Javascript**. Ce dernier prend la mise en forme définie par les attributs dans le code Html puisque nous n'avons modifié que le texte (propriété innerText).  
  
L'application devient intéressante et dynamique mais nous prendrons soin de la parfaire dans le TP suivant, de manière à ce que le code se déclenche sur l'événement d'un clic sur un bouton, et non plus au chargement de la page. Nous aborderons ainsi les contrôles Input avec la gestion de leurs événements.

**TP2 :** [Interactions Html et événements Javascript](https://www.bonbache.fr/interactions-html-et-evenements-javascript-222.html)

* Copier/ Coller le fichier calcul-imc-javascript.html
* Sur la nouvelle copie, modifier remplacer les lignes de 40 à 49 par les lignes suivantes :

<div style="float:left;width:24%;height:100px;"></div>

<div style="float:left;width:15%;height:100px;text-align:center;">

<div style="width:auto;height:100px;display:block;position:relative;background-color:#EDEEEE;border:#666666 1px solid;padding-top:12px;">

<div style="height:30px;width:100%;"></div>

<div style="height:30px;width:100%;"></div>

<div style="height:40px;width:100%;"></div>

</div>

</div>

<div style="float:left;width:7%;height:100px;"></div>

<div style="float:left;width:30%;height:100px;text-align:center;">

<div style="width:auto;height:100px;display:block;position:relative;background-color:#EDEEEE;border:#666666 1px solid;padding-top:12px;">

<div style="height:50px;width:100%;font-size:14px;font-weight:bold;">Le calcul de votre IMC conduit à un résultat de :</div>

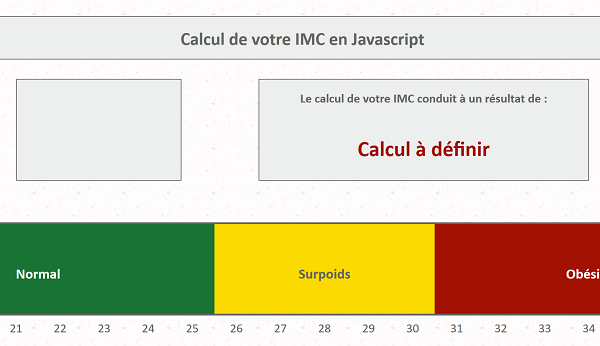
<div style="height:40px;width:100%;font-size:24px;font-weight:bold;color:#a21001;">Calcul à définir</div>

</div>

</div>

<div style="float:left;width:24%;height:100px;"></div>

Vous devez obtenir la page suivante :



* En ligne 44, créer une zone de saisie entre les bornes du Div, comme suit :

*<input type='text' value='Votre poids en Kg' />*  
  
Le type réglé sur text permet de définir une zone de saisie. Notez l'attribut supplémentaire **value** qui permet de définir le texte par défaut inscrit dans le Textbox.

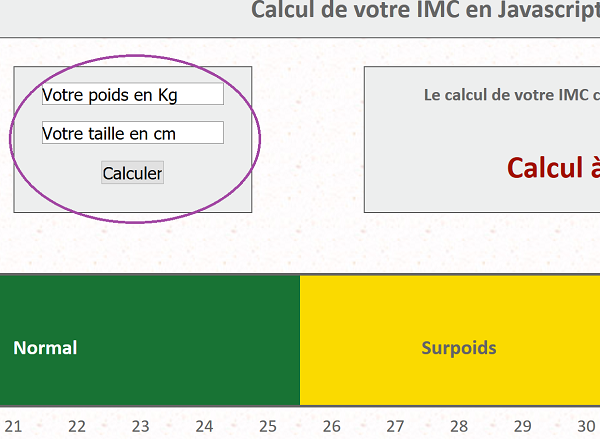
* En ligne 45, créer la zone de saisie pour l'information sur la taille, entre les bornes du div :

*<input type='text' value='Votre taille en cm' />*

* Puis, en ligne 46, dans le div, ajouter le code Html pour créer le bouton :

*<input type='button' value='Calculer' />*  
  
Ce qui donne au final, pour la section complète :  
  
*<div style='float:left; width:15%; height:100px; text-align:center;'>  
<div style='width:auto; height:100px; display:block; position:relative; background-color:#EDEEEE; border:#6666661px solid;padding-top:12px;'>  
<div style='height:30px; width:100%;'>****<input type='text' value='Votre poids en Kg' />****</div>  
<div style='height:30px; width:100%;'>****<input type='text' value='Votre taille en cm' />****</div>  
<div style='height:40px; width:100%;'>****<input type='button' value='Calculer' />****</div>  
</div>  
</div>*

* Enregistrer les modifications, basculer sur le navigateur et rafraîchir la page,



* Dans la section du script, créer la **fonction vider** comme suit :

*function vider(texte)  
{  
  
}*  
  
La **variable texte** doit correspondre au contenu de la zone de texte cliquée. Si son contenu est celui d'origine, il doit être vidé, sinon il doit être conservé. Une variable se déclare à la volée en paramètre de fonction, soit sans le mot clé var, comme nous l'avions appris pour les cas usuels. Nous devons vérifier à l'aide d'une **instruction conditionnelle if**, si le contenu de la zone est soit le texte : *Votre poids en Kg*, soit le texte : *Votre taille en cm*. Souvenez-vous, le **OU** s'exprime à l'aide des symboles **||** et le **ET** à l'aide des symboles **&&**, en Javascript.

* Entre les accolades de la fonction, créer le test Javascript suivant :

*if(texte=='Votre poids en Kg' || texte=='Votre taille en cm')  
return '';  
else  
return texte;*

Traduisons ce que nous venons de coder. Si le texte compris dans la zone de saisie cliquée vaut soit : *Votre poids en Kg*, soit : *Votre taille en cm*, alors on retourne une valeur vide pour effacer son contenu. Dans le cas contraire, on retourne sa propre valeur pour la conserver intacte.  
  
Pour que cette fonction soit utile, encore faut-il qu'elle soit appelée et que son résultat soit affecté. Elle doit être appelée au clic sur l'une des zones de texte qui doit récupérer et afficher la valeur renvoyée par l'instruction return. Et ces événements, nous savons désormais les générer et les gérer.

* Dans les contrôles input du poids et de la taille, ajouter l'événement suivant :

*onclick='this.value = vider(this.value);'*  
  
Ce qui donne :  
  
*<div style='height:30px;width:100%;'><input type='text' value='Votre poids en Kg'****onclick='this.value = vider(this.value);'****/></div>  
<div style='height:30px;width:100%;'><input type='text' value='Votre taille en cm'****onclick='this.value = vider(this.value);'****/></div>*

Au clic sur une zone de texte (onclick=), nous affectons son contenu (this.value=) à la valeur retournée par le traitement de la fonction vider, à laquelle nous passons ce même contenu pour analyse (vider(this.value)) par l'instruction if.

* Enregistrer les modifications, basculer dans le navigateur et rafraîchir la page,
* Cliquer sur la première zone de texte, celle du poids,

Comme vous le constatez, son contenu se vide pour laisser place à la saisie.

* Saisir une valeur numérique, par exemple : 85,
* Cliquer alors dans la seconde zone de saisie pour vider son contenu,
* Saisir une taille en centimètres, par exemple : 181,
* Cliquer de nouveau dans la première zone de texte,

Cette fois, comme son contenu n'est plus celui d'origine, grâce au **test de l'instruction If** dans la **fonction vider**, il n'est pas purgé et la saisie peut être poursuivie ou modifiée.

**Traitement Javascript sur Clic**  
Nous avons appris à **générer des événements** et à y relier des **fonctions de traitement Javascript** à enclencher. **Au clic sur le bouton Calculer**, nous devons déclencher le code d'une fonction capable de calculer l'IMC de la personne en fonction des renseignements fournis dans les deux zones de texte.

* Revenir dans l'éditeur de code de la page Html,
* Dans la section script, sous la fonction vider, créer la **fonction calculer**, comme suit :

*function calculer()  
{  
}*

* Mettre le code du TP1 entre les accolades de la fonction calculer()

Les **variables taille et poids** sont celles qui nous intéressent dans un premier temps. Dans ce **code Javascript** récupéré, elles sont renseignées grâce à une saisie de l'internaute par le biais d'une boîte de dialogue prompt. Or désormais, elles doivent récupérer les valeurs contenues dans les **contrôles input** correspondants. Mais comme vous le savez désormais, pour piloter des éléments Html par le code Javascript, encore faut-il que ces éléments possèdent un identifiant. Nous devons donc ajouter un **attribut id** pour chacune des zones de texte.

* Attribuer les **identifiants zone\_poids et zone\_taille** respectivement aux contrôles Html input du poids et de la taille, comme suit :

*<div style='height:30px;width:100%;'><input type='text'****id='zone\_poids'****value='Votre poids en Kg' onclick='this.value = vider(this.value);'/></div>  
<div style='height:30px;width:100%;'><input type='text'****id='zone\_taille'****value='Votre taille en cm' onclick='this.value = vider(this.value);'/></div>*

* Par la même occasion, attribuer l'identifiant res\_imc au calque (div) contenant le texte : Calcul à définir, comme suit :

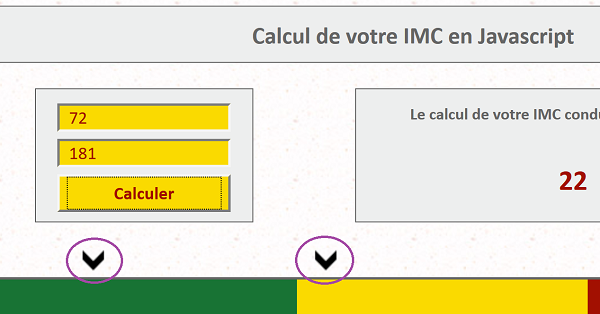
*<div style='height:40px; width:100%; font-size:24px; font-weight:bold; color:#a21001;'****id='res\_imc'****>Calcul à définir</div>*  
  
Ce calque est en effet appelé par le **code Javascript** que nous avons récupéré afin d'y inscrire le résultat du calcul de l'IMC.  
  
Dans ce code justement, il s'agit désormais d'adapter les affectations des variables correspondant au poids et à la taille. C'est toujours la **méthode getElementById** de l'**objet document** qui permet d'accéder à un élément Html reconnu par son Id. Et c'est la **propriété value** qui permet de désigner le contenu d'un contrôle input et non plus innerText que nous avions utilisé pour un calque.

* En conséquence, dans le code Javascript, adapter l'affectation des deux variables comme suit :

*taille = document.getElementById('zone\_taille').value;  
poids = document.getElementById('zone\_poids').value;*  
  
Maintenant que les variables du problème sont connues et mémorisées, la suite du code se charge du calcul de l'Imc, calcul que nous avions bâti dans la formation précédente. Nous avions pour ce faire, commencé par réaliser des tests à l'aide de l'instruction conditionnelle if, afin de nous assurer de la validité des saisies. D'ailleurs, dans chaque branche else, en cas de saisie erronée donc, nous pourrions réinitialiser le contenu de la zone de texte incriminée, pour inciter l'internaute à recommencer, comme suit :  
  
*...  
else  
{  
alert('Le poids entré n'est pas correct !rnVeuillez réactualiser par la touche F5');****document.getElementById('zone\_poids').value = 'Votre poids en Kg';*** *}  
}  
else  
{  
alert('La taille saisie n'est pas correcte !rnVeuillez réactualiser par la touche F5');****document.getElementById('zone\_taille').value = 'Votre taille en cm';*** *}*  
  
Ce n'est pas terminé bien sûr. Le code est encapsulé dans une fonction. Et pour qu'une fonction s'exécute, elle doit être appelée, précisément ici, au clic sur le bouton Calculer.

* Dans le code Html, pour le bouton Calculer, ajouter **l'événement onClick** avec **l'appel de la fonction calculer()**,
* Enregistrer les modifications et basculer sur le navigateur,
* Rafraîchir la page avec le raccourci CTRL + F5 afin de vider le cache,
* Saisir un poids dans la première zone de texte, par exemple : 85,
* Puis taper une taille dans la seconde zone de saisie, par exemple : 181,
* Enfin, cliquer sur le **bouton Calculer**,

Forcément le calcul produit un résultat différent. En conséquence, l'indicateur se place au-dessus d'une nouvelle graduation. Mais comme l'illustre la capture ci-dessous, l'ancien n'est pas masqué. Les deux subsistant et le résultat porte à confusion.  
  
Dans le TP précédent le problème ne se posait pas car le calcul était réalisé au chargement de la page. Et un chargement induit la réinitialisation de tous les contrôles et de toutes les variables.



C'est la **variable calculée indicateur** qui permet d'afficher l'image correspondante sur la graduation. L'astuce que nous proposons est de sortir sa déclaration de la fonction pour en faire une **variable publique**. La durée de vie d'une variable publique persiste au-delà des bornes de la fonction. En d'autres termes, à la deuxième exécution du code, elle est censée conserver la valeur de l'ancienne graduation. Donc si cette valeur n'est pas nulle, nous devons commencer par masquer l'image correspondante avant de poursuivre le code de la fonction.

* Dans la fonction calculer, supprimer la déclaration suivante : ***var indicateur;***,
* Au tout début du code Javascript, juste après l'ouverture de la balise script, ajouter la déclaration suivante :

*var indicateur=0;*  
  
En même temps que nous déclarons la variable, nous l'initialisons à 0. Cette valeur permet de savoir qu'aucune image n'a encore été affichée. Comme elle est déclarée en dehors de toute fonction, sa portée est publique. Elle conserve en mémoire la valeur qui lui est affectée par la **fonction calculer**, au-delà de l'exécution du code. Donc au début de la fonction calculer nous devons réaliser un test consistant à savoir si cette valeur vaut toujours 0. Si ce n'est pas le cas, l'image correspondante doit d'abord être masquée.

* Dans la fonction calculer, après la déclaration de variables, ajouter le test suivant :

*if(indicateur!=0)  
document.getElementById('img' + indicateur).style.visibility='hidden';*

* Enregistrer les modifications et basculer sur le navigateur,
* Rafraîchir la page à l'aide de la combinaison CTRL + F5,
* Puis réaliser deux tests consécutifs comme précédemment,

Cette fois, grâce à notre **variable publique**, l'ancienne image est masquée avant que la nouvelle ne soit affichée. Grâce aux **instructions conditionnelles**, aux **événements** générés par les contrôles et aux **fonctions**, nous sommes désormais en mesure de réaliser de véritables petites applications clientes. Dans un prochain volet, nous aborderons les traitements récursifs à l'aide des boucles, pour aller plus loin dans la puissance qu'offre le Javascript.

Source :

https://www.bonbache.fr/