Cheatography

JavaScript Cheat Sheet

by LeSep7ieme (LeSep7ieme) via cheatography.com/20351/cs/3147/

Variables

var myvariable;
var myvariable = 5.5;
myvariable = 6;
var myvariable1, myvariable2 = 'text';
alert(typeof number); // Affiche : « number »
alert(typeof text); // Affiche : « string »
alert(typeof aBoolean); //Affiche : « boolean »
alert(typeof nothing); //Affiche: « undefined »
Opérateurs : +, -, *, /, %
var input = prompt('Entrez un texte');
var six = parseInt('6');

Boucles

while (condition)
{ instructions; }
do
{ instructions; } while (condition);
for (initialisation; condition; incrémentation)
{ instructions; }
Ordre : initialisation avant toute autre chose, condition avant chaque passage, incrémentation en fin de boucle.

number++ renvoie number et l'incrémente et ++number incrémente number et renvoie le résultat.
break permet de sortir d'une boucle tandis que continue permet de passer à l'itération suivante.

Objets et Tableaux

var myArray = ['Sébastien', 'Laurent', 42];
myArray.push('Bob'); // Ajoute à la fin
myArray.unshift('Bob'); // Ajoute au début
myArray.shift(); // Enlève au début
myArray.pop(); // Enlève à la fin
chaine.split(c); Renvoie un tableau contenant les chaines séparées par c
myArray.join(-); Renvoie une chaine concaténant celles de myArray intercalant des -
Objets littéraux :
var myArray = {clé1 : value1, clé2 : value2};
alert(myArray.clé2); // Affiche value2
myArray['clé3'] = value3; ou myArray.clé3 = value3;
Parcours :

Objets et Tableaux (cont)

for (var id in myLArray) { alert(myLArray[id]);}

Les arrays ne sont pas typés.

Évènements

click : Cliquer (appuyer puis relâcher) sur l'élément
dblclick : Double-cliquer sur l'élément
mouseover : Faire entrer le curseur sur l'élément
mouseout : Faire sortir le curseur de l'élément
mousedown : Appuyer (sans relâcher) sur le bouton gauche de la souris sur l'élément
mouseup : Relâcher le bouton gauche de la souris sur l'élément
mousemove : Faire déplacer le curseur sur l'élément
keydown : Appuyer (sans relâcher) sur une touche de clavier sur l'élément
keyup : Relâcher une touche de clavier sur l'élément
keypress : Frapper (appuyer puis relâcher) une touche de clavier sur l'élément
focus : « Cibler » l'élément
blur : Annuler le « ciblage » de l'élément
change : Changer la valeur d'un élément spécifique aux formulaires (input, checkbox, etc.)
select : Sélectionner le contenu d'un champ de texte (input, textarea, etc.)
Spécifiques à <form>
submit : Envoyer le formulaire
reset : Réinitialiser le formulaire
addEventListener() :

```
<span id="clickme">Cliquez-moi !</span>
<script>
   var element = document.getElementById('clickme');
   element.addEventListener('click', function() {
      alert("Vous m'avez cliqué !");
   }, false);
</script>
```

Bouillonnement (true) : de l'enfant vers le parent.
Capture (false) : du parent vers l'enfant.

event.clientX (resp. clientY) : position selon l'axe des X (resp. Y).

Conditions

==, !=, >, >=, <, <=
=== : égalité de contenu et type
!== : contenu ou type différent de
&&, ||, !

Conditions (cont)

```
if (condition1) {}
else if (condition2) {}
...
else {}
switch (variable) {
  case value1: ... break;
  case value2: ... break;
  ...
  default:
}
 Ternaire :
(condition) ? valueTrue : valueFalse;
```

L'opérateur ou renvoie la première valeur évaluée à true.

Les fonctions

```
function myFunction (arguments)
  { instructions; return output;}
Avec arguments facultatifs :
function myFunction (args, argOptional)
{
  if (typeof argOptional === 'undefined')
    { argOptional = value; return output; }
  instructions;
}
Fonctions anonymes :
function (arguments)
  { contenu }
var fonction1 = function (args) {};
fonction1(args);
Code isolé:
(function (args) {})();
```

Manipuler le code HTML

```
var div = getElementById('myDiv');
var divs = getElementsByTagName('div');
var query = document.querySelector('#menu .item span');
var queryAll = document.querySelectorAll('#menu .item span');
alert(query.innerHTML);
```

Manipuler le code HTML (cont)

```
alert(queryAll.length);
var link = document.getElementById('myLink');
var href = link.getAttribute('href');
link.setAttribute('href', 'http://www.monsite.com');
```

L'attribut innerText contient le texte sans le balisage pour internet explorer, c'est textContent pour les autres navigateurs.

```
var parent = child.parentNode;
```

Attributs intéressants : firstChild, lastChild, previousSibling, nextSibling (ne retournent que les éléments et pas les #text).

```
var newLink = document.createElement('a');
document.getElementById('myP').appendChild(newLink);
var newLinkText = document.createTextNode("Le Site du Zéro");
newLink.appendChild(newLinkText);
```

Méthode clondeNode(boléen) clone un noeud avec ou sans ses enfants et ses attributs.

Méthode replaceChild(child1, child2) remplace l'enfant 2 par l'enfant 1.

```
parentNode.removeChild(child);
parentNode.hasChildNodes();
parent.insertBefore(toInsert, child);
```

Manipuler le CSS

```
element.style.backgroundColor = 'blue'; // Pour les mots avec un tiret on l'enlève et on met une majuscule
getComputedStyle() :
<script>
   var text = document.getElementById('text'),
      color = getComputedStyle(text).color;
   alert(color);
</script>
```

Propriétés offset :

offsetWidth : Contient la largeur complète (width + padding + border) de l'élément.

offsetHeight : Contient la hauteur complète (height + padding + border) de l'élément.

offsetLeft : Surtout utile pour les éléments en position absolue. Contient la position de l'élément par rapport au bord gauche de son élément parent.

offsetTop : Surtout utile pour les éléments en position absolue. Contient la position de l'élément par rapport au bord supérieur de son élément parent.

offsetParent : Utile que pour un élément en position absolue ou relative ! Contient l'objet de l'élément parent par rapport auquel est positionné l'élément actuel.