Chapitre 6
Tableaux

1. Tableaux à dimension unique

Nous avons entrevu dans le chapitre Développement à partir d'algorithmes le potentiel des tableaux à dimension unique et à dimensions multiples, voyons leur prise en compte sous JavaScript.

1.1 Syntaxe

En JavaScript, un tableau à dimension unique est une variable mémoire "composite" dans laquelle il va être possible de stocker plusieurs données indépendantes, y compris de types différents, avec une indexation de chacune des valeurs par un numéro (ou indice).

L'accès à chaque donnée du tableau se fera donc par l'intermédiaire de cette valeur d'indice.

Une particularité quant à cet indice, sa valeur pour la première cellule du tableau est 0.

Le langage JavaScript fournit plusieurs façons de créer un tableau :

– la syntaxe littérale,

– la syntaxe dite "Programmation orientée objet".

Avec une syntaxe littérale, la déclaration d'un tableau de nom `tabSemaine` de sept cellules contenant les libellés des jours d'une semaine se fait comme suit :

```
var tabSemaine = ["Lundi", "Mardi", "Mercredi", "Jeudi",
"Vendredi", "Samedi", "Dimanche"];
```

Vous noterez que la déclaration s'est accompagnée de l'initialisation de chacune des cellules du tableau `tabSemaine` (de la cellule d'indice 0 à la cellule d'indice 6).

Avec une syntaxe "Programmation orientée objet", vous auriez :

```
var tabSemaine = new Array("Lundi", "Mardi", "Mercredi", "Jeudi",
"Vendredi", "Samedi", "Dimanche");
```

Nous aurions pu déclarer le tableau `tabSemaine` sans lui affecter des valeurs. Des affectations ultérieures peuvent être envisagées, comme par exemple pour le Lundi :

```
tabSemaine[0] = "Lundi";
```

Ce qui est vraiment particulier dans la gestion des tableaux sous JavaScript est l'extrême souplesse autorisée :

– pas de dimensionnement a priori (il est toujours possible d'étendre la taille du tableau en fonction des besoins),
– possibilité de mélanger dans un même tableau des données de types différents,
– possibilité d'utiliser des tableaux associatifs (tableaux pour lesquels les indices sont remplacés par des valeurs textuelles).

Pour accéder dans un traitement au contenu d'une valeur de tableau rangée à une position d'indice particulière, la syntaxe sera :

```
document.write("Le 4ième jour de la semaine est " + tabSemaine[3]);
```

Remarque

Il faut toujours se rappeler que la numérotation des indices débute à zéro.

Enfin, sachez que JavaScript propose une multitude de méthodes s'appliquant sur les tableaux (`Array`). Vous pourrez facilement par ces méthodes insérer, supprimer, repérer des éléments d'un tableau. Il existe même des méthodes de tri (`sort`, `reverse`) pour classer facilement les valeurs contenues dans un tableau sans avoir recours à l'écriture fastidieuse d'un algorithme de tri.

1.2 Exercice n°14 : Décompte des nombres pairs dans un tableau

Sujet

Détermination du nombre de nombres pairs dans un tableau (saisie préalable des valeurs à prévoir au clavier).

Corrigé (partiel) en JavaScript

```
/* Déclaration de variables locales */
/*
i          : Compteur de boucle
nbPairs    : Cumul du nombre de nombres pairs
tableau    : Tableau des nombres
*/
var i, nb_pairs;
var tableau = new Array;

/* Initialisations */
nbPairs = 0;
for (i=1; i<=5; i++)
{
    tableau[i] = parseInt(prompt("tableau[" + i +"] : "));
}

/* Détermination du nombre de nombres pairs dans le tableau */
for (i=1; i<=5; i++)
{
    if (tableau[i]%2 == 0)
    {
        nbPairs = nbPairs + 1;
    }
}

/* Affichage du résultat */
document.write("Le tableau contient " + nbPairs + " nombres pairs");
```

Commentaires du code JavaScript

Rien de vraiment nouveau n'est présenté dans ce script hormis le calcul du modulo. Ce calcul sert ici à déterminer la parité de chaque contenu de cellules du tableau. Il est réalisé par l'intermédiaire l'opérateur `%`.

Vous aurez peut-être noté que dans ce script la cellule d'indice 0 n'a pas été utilisée (la numérotation par la boucle `for` débute à 1). Ce choix rend sans doute plus compréhensible l'algorithme (il n'y a que les informaticiens qui s'accommodent de la numérotation à partir de zéro !).

2. Tableaux à dimensions multiples

Il est fréquent que l'on ait besoin de tableau à dimensions multiples pour gérer des problématiques, notamment en mathématique, en statistique...

JavaScript offre cette possibilité.

2.1 Syntaxe

Comme pour les tableaux à dimension unique, JavaScript permet de déclarer les tableaux à dimensions multiples de plusieurs façons :

– avec une syntaxe littérale,

– avec une syntaxe dite "Programmation orientée objet".

Avec une syntaxe dite "Programmation orientée objet" (encore appelée JSON - *JavaScript Object Notation*), la déclaration d'un tableau de nom `tabMatrice` de deux lignes subdivisées en quatre colonnes avec initialisation se fait comme suit :

```
/* Déclaration du tableau tabMatrice */
var tabMatrice tableau = new Array();

/* Déclaration de la première "ligne" du tableau tabMatrice */
tabMatrice[0]=new Array()

/* Initialisation des 4 "colonnes" de la première "ligne" */
tabMatrice[0][0] = "Un";
```

```
tabMatrice[0][1] = "Deux";
tabMatrice[0][2] = "Trois";
tabMatrice[0][3] = "Quatre";

/* Déclaration de la deuxième "ligne" du tableau tabMatrice */
tabMatrice[1]=new Array()

/* Initialisation des 4 "colonnes" de la deuxième "ligne" */
tabMatrice[1][0] = "Onze";
tabMatrice[1][1] = "Douze";
tabMatrice[1][2] = "Treize";
tabMatrice[1][3] = "Quatorze";
```

2.2 Exercice n°15 : Mini-tableur

Sujet

Soit le tableau `tb` à deux dimensions comportant quatre lignes et cinq colonnes. Réaliser les traitements suivants :

– saisir au clavier des valeurs dans les trois premières lignes et les quatre premières colonnes (on conserve la dernière ligne et la dernière colonne libres pour des additions de lignes et de colonnes),
– additionner les colonnes en dernière ligne et les lignes en dernière colonne.

Corrigé (partiel) en JavaScript

```
 /* Déclaration de variables locales */
var tb = new Array(5);
var numLigne, numColonne;
var valeur;

/* Déclaration de 5 "colonnes" par "ligne" pour le tableau tb */
for (var numLigne=1; numLigne<tb.length; numLigne++)
{
    /* Création des "colonnes" (numérotées de 0 à 5) */
    tb[numLigne]=new Array(6);
}

/* Initialisation du tableau tb (3 lignes * 4 colonnes) par une saisie clavier */
for (numLigne = 1; numLigne <= 3; numLigne++) {
    for (numColonne = 1; numColonne <= 4; numColonne++) {
        valeur = parseInt(prompt("tableau[" + numLigne + "][" + numColonne + "] = "));
        tb[numLigne][numColonne] = valeur;
    }
}

/* Mise à zéro des totaux en ligne n°4 */
```

```
for (numColonne=1; numColonne<=5; numColonne++)
{
    tb[4][numColonne] = 0;
}

/* Mise à zéro des totaux en colonne n°5 */
for (numLigne=1; numLigne<=4; numLigne++)
{
    tb[numLigne][5] = 0;
}

/* Détermination des totaux en ligne n°4 et en colonne n°5 */
for (numLigne=1; numLigne<=3; numLigne++)
{
    for (numColonne=1; numColonne<=4; numColonne++)
    {
        /* Totalisation en ligne n°4 */
        tb[4][numColonne] = tb[4][numColonne]
        + tb[numLigne][numColonne];
        /* Totalisation en colonne n°5 */
        tb[numLigne][5] = tb[numLigne][5]
        + tb[numLigne][numColonne];
        /* Totalisation générale en ligne n°4-colonne n°5 */
        tb[4][5] = tb[4][5] + tb[numLigne][numColonne];
    }
}

/* Affichage du total général */
/* NB : Total de 78 étant donné la technique de remplissage retenue
du tableau tb */
document.write("Total général en tb[4][5] = " + tb[4][5]);
```

Commentaires du code JavaScript

Le tableau `tb` est dans un premier temps déclaré comme étant un tableau à une seule dimension avec cinq cellules, implicitement numérotées de 0 à 4 comme suit :

```
var tb = new Array(5);
```

Vous noterez que la ligne de numéro zéro ne sera pas utilisée par la suite. Ce choix a été fait car l'utilisation ultérieure de cette ligne ne serait pas très intuitive.

Dans un second temps les lignes de ce tableau sont elles-mêmes subdivisées en colonnes, comme ceci :

```
/* Déclaration de 5 "colonnes" par "ligne" pour le tableau tb */
for (var numLigne=1; numLigne<tb.length; numLigne++)
{
    /* Création des "colonnes" (numérotées de 0 à 5) */
    tb[numLigne]=new Array(6);
}
```