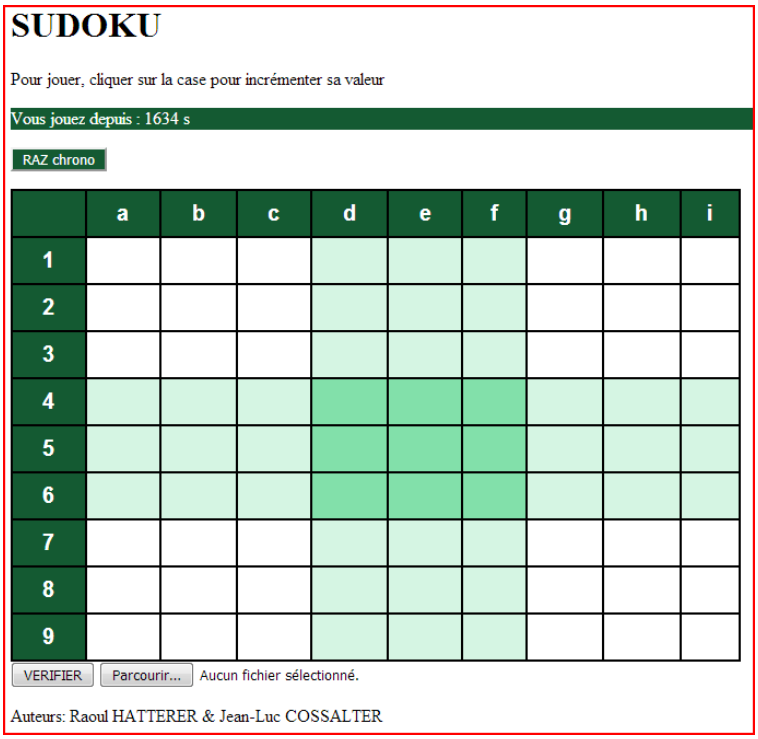
PAGE WEB pour le SUDOKU



But

Afficher sur une page Web une grille de Sudoku au format .csv et permettre à l'utilisateur de la remplir.

Fonctionnalités

- Sélection et chargement d'une grille au format .csv
- Placement d'un chiffre dans une case
- Vérification de la justesse de la grille
- Affichage du temps mis pour résoudre la grille
- Remise à zéro du chronomètre

Composition du script

Partie 1 : définition du style

En utilisant CSS, la partie style définit

- les différents formats utilisés pour l'affichage : taille, police, couleur des caractères, couleur du fond
- Les formats des boutons
- Les formats des cases du Sudoku

Partie 2 : structuration et mise en forme de la page Web

En utilisant le langage HTML, cette partie va décrire l'organisation de la page ainsi que son contenu :

- Les boutons,
- Les zones de texte
- Le tableau permettant l'affichage du sudoku

Partie 3 : fonctionnalités rendant la page Web interactive

Cette partie est écrite en utilisant le langage de programmation JavaScript, elle permet de créer des fonctions :

- De démarrage
- Associées aux boutons
- Associés aux clics de la souris

Description du code source

CSS

Code entre les balises <style> et </style>

```
table {                          ← description du tableau
 font-family: arial, sans-serif;  ← police utilisée
 font-size: 20px;                 ← taille des caractères
 border-collapse: collapse;       ← bordure : les cases du tableau sont jointes
 font-weight: bold;               ← écriture en gras
 width: 50%;                      ← taille de 50% de la largeur de la page
}
td, th {                         ← cases du tableau
 border: 2px solid #000000;       ← la ligne de bordure a une largeur de 2 pixels et est noire
 text-align: center;              ← le texte est centré
 padding: 8px;                    ← décalage de 8 pixels
 color:#145a32;                   ← couleur
}
tr:nth-child(6) {background-color:  #d5f5e3;}    ← définition des couleurs de fond
tr:nth-child(4) {background-color: #d5f5e3;}
tr:nth-child(5) {background-color:  #d5f5e3;}
th {
 background-color:  #145a32 ;color:white;
}
```

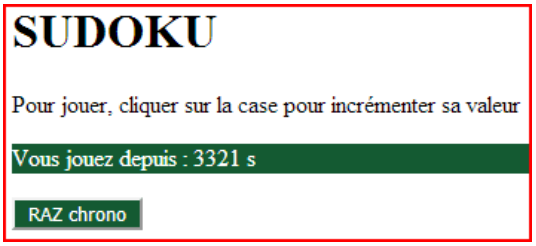
Enfin selon leur id les cases ont des couleurs de fond et des couleurs différentes :

```
#b0,#b1,#b2,#b3,#b4,#b5,#b6,#b7,#b8,#affich_tps,#RAZ{background-color:  #145a32 ;color:white;}

#a4,#a3,#a5,#a12,#a13,#a14,#a21,#a22,#a23,#a57,#a58,#a59,#a66,#a67,#a68,#a75,#a76,#a77{backg
round-color: #d5f5e3;}

#a30,#a31,#a32,#a39,#a40,#a41,#a48,#a49,#a50{background-color: #82e0aa;
```

HTML

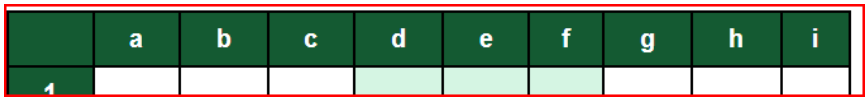
On commence par le haut de la page

```
<body>
  <h1 id="ici">SUDOKU</h1>
        <p>Pour jouer, cliquer sur la case pour incrémenter sa valeur</p>
        <p> </p>
        <p id="affich_tps"></p>
        <button id="RAZ" onclick="initialise_tps()">RAZ chrono</button>
        <p> </p>
```



Puis le tableau (la grille de Sudoku) décrit ligne par ligne en commençant par l'en-tête

```
<thead>
    <tr>
     <th></th>
                    <th
scope="col">a</th>
      <th scope="col">b</th>
      <th scope="col">c</th>
      <th scope="col">d</th>
      <th scope="col">e</th>
      <th scope="col">f</th>
      <th scope="col">g</th>
      <th scope="col">h</th>
      <th scope="col">i</th>
    </tr>
  </thead>
```




```
<tr>
  <td id="b0" >1</td>
  <td id="a0" onclick="ajoute(event)"></td>
  <td id="a1" onclick="ajoute(event)"></td>
  <td id="a2" onclick="ajoute(event)"></td>
  <td id="a3" onclick="ajoute(event)"></td>
  <td id="a4" onclick="ajoute(event)"></td>
  <td id="a5" onclick="ajoute(event)"></td>
  <td id="a6" onclick="ajoute(event)"></td>
  <td id="a7" onclick="ajoute(event)"></td>
  <td id="a8" onclick="ajoute(event)"></td>
</tr>
```

A partir de la ligne 1, chaque case :

- possède un identifiant pour affecter les bonnes couleurs et repérer la case
- et peut être cliquée pour pouvoir appeler la fonction JavaScript *ajoute* qui modifie la valeur de la case

Pour terminer : Description des 2 boutons pour charger le fichier et vérifier la grille :

```
<button id="bouttest" onclick="verif()">VERIFIER</button>
<input type="file" id="fileinput" multiple />
<div id="result"></div>
<p id="demo"></p>


<p class="author">Auteurs: Raoul HATTERER &amp;amp; Jean-Luc COSSALTER</p>
<script type="text/javascript">
```

JavaScript

Cette partie du script définit les différentes fonctions utilisées.

```
function ajoute(e) {
            var element = e.target || e.srcElement;
            var x = document.getElementById(element.id).innerHTML;
   document.getElementById(element.id).innerHTML = (x/1+1)%10;
}
```

La fonction *ajoute* est appelée par un clic sur une case de tableau qui est vide au départ. Elle consiste à lire la valeur de la case cliquée (on utilise son id), de lui rajouter 1 modulo 10 puis à réaffecter cette nouvelle valeur à la case du tableau.

La case cliquée est donc incrémentée après un clic sur celle-ci.

Le modulo 10 (%10) permet de revenir à 0 qui est la valeur affectée à une case lorsque l'on ne connaît pas encore sa valeur.

```
function readMultipleFiles(evt)
              {
      var files = evt.target.files;

      if (files) {
         f=files[0];
              var r = new FileReader();
            r.onload = (function(f) {
               return function(e) {
                  var contents = e.target.result;
                  var res = document.getElementById("result");
                  {var i; for(i=0;i<81;i++) { if (contents[2*i+(i-(i % 9))/9]>0){document.getElementById("a"+i).innerHTML
=contents[2*i+(i-(i %
9))/9];document.getElementById("a"+i).setAttribute("onclick","");document.getElementById("a"+i).setAttribute("style",
"color:red");}}}                  }

                              })(f);

            r.readAsText(f);

             }
                                                 else {
          alert("Echec lors du chargement");
               }
    }

       document.getElementById('fileinput').addEventListener('change',readMultipleFiles, false);
```

Cette fonction permet de sélectionner un fichier .csv et d'affecter à chacune des 81 cases du tableau d'affichage la valeur correspondante du tableau .csv

De plus si la valeur est supérieure à 0 (cas de la case non vide) :

- on efface la valeur de onclick, et donc la case ne pourra pas être modifiée
- on modifie le format d'affichage pour que la valeur apparaisse en rouge

Si la valeur est égale à 0 (cas de la case vide) :

- onclick reste à la valeur ajoute(envent) et donc un clic sur cette case appellera la fonction ajoute et la case pourra être modifiée
- Le format d'affichage n'est pas modifié (reste en noir)

```
function verif()
{
var test1=0;
for(var j1=0;j1<9;j1++){var ligne =[];
for(var i1=0;i1<9;i1++) {var k1=i1+9*j1; ligne.push(document.getElementById("a"+k1).innerHTML); };
ligne.sort();

if (ligne!="1,2,3,4,5,6,7,8,9") test1=test1+1;

}

var test2=0;
for(var j2=0;j2<9;j2++){var ligne =[];
for(var i2=0;i2<9;i2++) {var k2=j2+9*i2; ligne.push(document.getElementById("a"+k2).innerHTML); };
ligne.sort();

if (ligne!="1,2,3,4,5,6,7,8,9") test2=test2+1;

}if ((test1==0)&&(test2==0)) document.getElementById("demo").innerHTML = "La grille est bien remplie"; else
document.getElementById("demo").innerHTML = "La grille comporte des erreurs ou n'est pas totalement remplie";


var test3=0;
for(var carre=0;carre<9;carre++){var ligne =[];
for(var i3=0;i3<9;i3++) {var k3=((i3%3)+((i3-(i3%3))*3)+(carre%3)*3+(carre-(carre%3))*9);
ligne.push(document.getElementById("a"+k3).innerHTML); };
ligne.sort();

if (ligne!="1,2,3,4,5,6,7,8,9") test3=test3+1;

}if (((test1==0)&&(test2==0))&&(test3==0)) document.getElementById("demo").innerHTML = "La grille est bien remplie";
else document.getElementById("demo").innerHTML = "La grille comporte des erreurs ou n'est pas totalement remplie";
}
```

Dans cette fonction on effectue 3 tests :

Le test sur les lignes (test1), le teste sur les colonnes (test2) et le test sur les petits carrés (test3).

Chaque test consiste à vérifier que tous les éléments (1,2,3,4,5,6,7,8,9) sont bien présents dans la ligne, la colonne ou le carré.

Chaque test consiste en :

- Créer un tableau vide appelé ligne
- Lui ajouter tous les éléments de la zone (ligne ou colonne ou carré)
- L'ordonner
- Vérifier que le tableau ligne contient tous les éléments
- Si ce n'est pas le cas rajouter 1 au résultat du test

A la fin si tous les tests sont à 0 (cas où le tableau est rempli correctement) on indique que le tableau est rempli correctement, sinon on indique qu'il y des erreurs.

```
function initialise_tps(){
  var date = new Date();
  var t0 = date.getTime();
  temps=t0;
  Afficher_temps();
}
```

La variable temps correspond au l'heure initiale (début de partie)

Cette fonction remet tout simplement la variable temps à la valeur de l'heure courante.

```
function Afficher_temps() {
  var date = new Date();
  var t1 = date.getTime();
  t1=t1-temps;
  var tps_en_s = (t1-t1%1000)/1000
  document.getElementById("affich_tps").innerHTML = "Vous jouez depuis : "+tps_en_s+" s";
  setTimeout("Afficher_temps()",1000);
    console.log("Afficher_temps");
}
```

Cette fonction lit l'heure courante, fait la différence avec l'heure initiale (variable temps) et l'affiche dans la zone de texte prévue à cet effet.