

```

1  <?php
2  /*****
3  AUTEUR      : Constantin Herrmann
4  LIEU        : CFPT Informatique Genève
5  DATE        : 14.06.2018
6
7  TITRE PROJET: KidsGames Geneva Score
8
9  TITRE PAGE  : points_counting.c
10 DESCRIPTION : Ce script contient toutes les fonctions concernant le calcul des
    • points pour les matchs
11 VERSION     : 1.0
12 *****/
13
14 /**
15  * Met à jour les scores et le classement général
16  *
17  * @param array un tableau contenant les infos des équipes ainsi que les scores
    • obtenus lors du match
18  * @param string l'id du match dans lequel les scores on été obtenus
19  */
20 function UpdateScore($Teams, $idGame){
21
22     // Va rechercher les infos du match
23     $game = GetAllMatchInfos($idGame);
24     // Nous récupérons les résultats du match avant la mise à jour des nouveaux
    • résultats
25     $old_results = GetMatchResults($idGame, $game);
26
27     // On indique que le match est terminé
28     SetGameToDone($idGame);
29     $game = GetAllMatchInfos($idGame);
30
31     // On parcourt le tableau des équipes
32     for ($team=0; $team < count($Teams); $team++) {
33
34         // On garde l'id de l'équipe dans une variable a part
35         $idTeam = $Teams[$team]['infos']['numero'];
36         // Idem que pour l'id mais pour le score
37         $score = $Teams[$team]['score'];
38
39         // On effectue la mise a jour du score dans la base de données
40         // ATTENTION : On ne met pas la variable $new_score mais bien la variable
    • $score
41         static $query = null;
42
43         if ($query == null) {
44             $req = "UPDATE `plays` SET `score` = :score WHERE `idTeam` = :idTeam AND
    • `idGame` = :idGame";
45             $query = connecteur()->prepare($req);
46         }
47         try {
48             $query->bindParam(':idTeam', $idTeam, PDO::PARAM_STR);
49             $query->bindParam(':idGame', $idGame, PDO::PARAM_STR);
50             $query->bindParam(':score', $score, PDO::PARAM_STR);
51             $query->execute();
52             $query->fetch();
53         }

```

```

54     catch (Exception $e) {
55         error_log($e->getMessage());
56     }
57 }
58
59
60 // Nous récupérons les résultats du match après la mise a jour des nouveaux
    • résultats
61 $new_results = GetMatchResults($idGame, $game);
62
63 /* On recherche de quel classement il s'agit
64 Tchouckball = B
65 KinBall = A
66 Course Agile = C */
67 $classment = GetTypeOfClassement($idGame);
68 $c_label = null;
69 if ($classment['sport'] == 1) {
70     $c_label = 'B';
71 }elseif ($classment['sport'] == 2) {
72     $c_label = 'A';
73 }elseif ($classment['sport'] == 3) {
74     $c_label = 'C';
75 }
76
77 // On parcourt toutes les équipes du match
78 for ($team=0; $team < count($Teams); $team++) {
79     // On garde l'id de l'équipe dans une variable a part
80     $idTeam = $Teams[$team]['infos']['numero'];
81     // Idem que pour l'id mais pour le score
82     $score = $Teams[$team]['score'];
83
84     // On déclare 2 variables provisoires représentant le vieux score et le
    • nouveau
85     // par défaut on les met a NULL
86     $old_provisoire = null;
87     $new_provisoire = null;
88
89     // on parcourt tout les anciens résultats
90     for ($res=0; $res < count($old_results); $res++) {
91         // Si l'id du détenteur du résultat est égal a l'id de l'équipe dans la
    • boucle
92         if ($old_results[$res]['id'] == $idTeam) {
93             // On assigne le score à la variable provisoire : old
94             $old_provisoire = $old_results[$res];
95             break;
96         }
97     }
98
99     // Idem que la boucle ci-dessus, sauf que cette fois-ci on le fait pour le
    • nouveau score
100     for ($res=0; $res < count($new_results); $res++) {
101         if ($new_results[$res]['id'] == $idTeam) {
102             $new_provisoire = $new_results[$res];
103             break;
104         }
105     }
106
107 // On va récupérer les données du classement général des équipes
108 $team = GetMatchClassement($idTeam, $c_label);

```

```

108     $main = GetMainClassement($idIeam, $c_label);
109
110     // Création du tableau qui contiendra les nouvelles données à entrer
111     $final_new_results = [];
112
113     /*
114     On calcule le nombres de points obtenus grace a la formule : points =
    • pointsactuels + (nouveaux_points - anciens_points)
115
116     SI C'EST UN MODIFICATION DE SCORES ERRONES :
117     Ex:
118         Points actuels = 6;
119         nouveaux_points = 1; -> L'équipe à fait égalité avec une autre
120         anciens_points = 3; -> L'équipe avait gagné le match, mais c'etait des
    • mauvais résultats
121
122         points = 6 + ( 1 - 3 ) = 4
123
124     SI C'EST UN NOUVEAU MATCH :
125     Ex:
126         Points actuels = 6;
127         nouveaux_points = 3; -> L'équipe à gagné son match
128         anciens_points = 0; -> L'équipe n'avait pas encore joué le match
129
130         points = 6 + ( 3 - 0 ) = 9
131
132
133     On fait idem pour les points marqués et les points reçus
134     */
135     $final_new_results['p'] = $main['p'] + ($new_provisoire['points'] -
    • $old_provisoire['points']);
136     $final_new_results['m'] = $main['m'] + ($new_provisoire['m'] -
    • $old_provisoire['m']);
137     $final_new_results['r'] = $main['r'] + ($new_provisoire['r'] -
    • $old_provisoire['r']);
138
139     // On met à jour la table principale
140     UpdateMainTable($idTeam, $final_new_results['p'], $final_new_results['m'],
    • $final_new_results['r'], $c_label);
141 }
142 }
143
144 /**
145 * Retourne un tableau avec les points et les score par équipe pour un match en
    • question
146 *
147 * @param string l'id du match
148 * @param array le match en question
149 * @return array un tableau avec tout les scores ainsi que les poitns obtenus par
    • équipe
150 *
151 *     [index]
152 *     ['id'] -> l'id de l'équipe
153 *     ['m'] -> points marqués par l'équipe
154 *     ['r'] -> nombre de points encaisser
155 *     ['d'] -> la différence de points
156 *     ['points'] -> les points recus grâce au match (3 si victoire, 1 si
    • égalité, 0 si défaite)
157 */
158 function GetMatchResult($idGame, $game){

```

```

157 function GetMatchResults($idGame, $game){
158     static $query = null;
159
160     if ($query == null) {
161         $req = "SELECT `idTeam` as id, `score` FROM `plays` WHERE `idGame` = :idGame
        • ORDER BY `score` DESC";
162         $query = connecteur()->prepare($req);
163     }
164
165     try {
166         $query->bindParam(':idGame', $idGame, PDO::PARAM_STR);
167         $query->execute();
168         $res = $query->fetchAll(PDO::FETCH_ASSOC);
169     }
170     catch (Exception $e) {
171         error_log($e->getMessage());
172         $res = false;
173     }
174
175     $result = $res;
176     // On va créer les résultats pour le match
177     $points = MakeTheResults($result, $game);
178     return $points;
179 }
180
181 /**
182  * Retourne un tableau avec les points et les score par équipe pour un match en
    • question
183  *
184  * @param array les équipes qui ont participer au match
185  * @param array le match en question
186  * @return array un tableau avec tout les points obtenus par équipe
187  *
188  *         ['id'] -> l'id de l'équipe
189  *         ['m'] -> points marqués par l'équipe
190  *         ['r'] -> nombre de points encaisser
191  *         ['d'] -> la différence de points
192  *         ['points'] -> les points recus grâce au match (3 si victoire, 1 si
    • égalité, 0 si défaite)
193  */
194 function MakeTheResults($teams, $game){
195     $result = [];
196
197     // On parcourt toutes les équipes
198     for ($temp=0; $temp < count($teams); $temp++) {
199         $result[$temp]['id'] = $teams[$temp]['id'];
200         $result[$temp]['m'] = $teams[$temp]['score'];
201         $result[$temp]['r'] = 0;
202         $result[$temp]['d'] = $teams[$temp]['score'];
203         $result[$temp]['points'] = 0;
204         for ($r=0; $r < count($teams); $r++) {
205             if ($r != $temp) {
206                 $result[$temp]['d'] = $result[$temp]['d'] - $teams[$r]['score'];
207                 $result[$temp]['r'] = $result[$temp]['r'] + $teams[$r]['score'];
208             }
209         }
210     }
211     // Création des points d'après leur position dans le match
212     $result = MakethePoints($result, $game):

```

```

212     $result = MakeThePoints($teams, $game);
213     return $result;
214 }
215
216 /**
217  * Caclucule les points pour chaque équipe
218  *
219  * @param array les équipes classées par résultats obtenus lors du match
220  * @param array le match en question
221  * @return array un tableau avec tout les points pour chaque équipe
222  */
223 function MakeThePoints($teams, $game){
224     $table = $teams;
225
226     // Si il y a 2 équipes (Tchouckball)
227     if (count($teams) == 2) {
228         // On récupère les points marqués par les équipes
229         $t1 = $teams[0]['m'];
230         $t2 = $teams[1]['m'];
231
232         // Si les score de la première équipe est plus grand que celui de la deuxième
233         if ($t1 > $t2) {
234             // Le gagnant reçoit 3 points et le perdant 0
235             $table[0]['points'] = 3;
236             $table[1]['points'] = 0;
237         }
238         // Si les deux scores sont à égalité
239         elseif ($t1 == $t2) {
240             // On vérifie si c'est un nouveau match ou si il à déjà été joué (donc
241             • MODIFICATION)
242             if ($game['infos']['played'] == "1") {
243                 $table[0]['points'] = 1;
244                 $table[1]['points'] = 1;
245             }
246             else{
247                 $table[0]['points'] = 0;
248                 $table[1]['points'] = 0;
249             }
250             // Si le score de la première équipe est plus petit que le score de la
251             • deuxième
252             elseif ($t1 < $t2) {
253                 // Le gagnant reçoit 3 points et le perdant 0
254                 $table[0]['points'] = 0;
255                 $table[1]['points'] = 3;
256             }
257         }
258         // Si il y a 3 équipes (KinBall et course agile)
259         elseif(count($teams) == 3){
260             $t1 = $teams[0]['m'];
261             $t2 = $teams[1]['m'];
262             $t3 = $teams[2]['m'];
263             // Si le score de l'équipe 1 est plus grand que tout les autres scores
264             if ($t1 > $t2 && $t1 > $t3) {
265                 // Alors l'équipe 1 reçoit 3 points
266                 $table[0]['points'] = 3;
267
268                 // Si le score de l'équipe 2 est plus grand que celui de l'équipe 3

```

```

269     if ($t2 > $t3) {
270         // L'équipe 2 reçoit 1 points
271         $table[1]['points'] = 1;
272         // L'équipe 3 reçoit 0 points
273         $table[2]['points'] = 0;
274     }
275     // L'équipe 2 et l'équipe 3 ont les mêmes scores
276     elseif ($t2 == $t3) {
277         // Les 2 équipes reçoivent 1 point
278         $table[1]['points'] = 1;
279         $table[2]['points'] = 1;
280     }
281 }
282 // Si les deux premières équipes sont égalités ET que la troisième équipe a un
    • score inférieur aux deux autres
283 elseif ($t1 == $t2 && $t1 > $t3) {
284     // Les deux premières équipes reçoivent 1 point
285     $table[0]['points'] = 1;
286     $table[1]['points'] = 1;
287
288     // La troisième équipe reçoit aucun point
289     $table[2]['points'] = 0;
290 }
291 // Si toutes les équipes sont à égalités
292 elseif ($t1 == $t2 && $t1 == $t3) {
293     // Si le match a déjà été joué
294     if ($game['infos']['played'] == "1") {
295         // Toutes les équipes reçoivent 1 point
296         $table[0]['points'] = 1;
297         $table[1]['points'] = 1;
298         $table[2]['points'] = 1;
299     }
300
301     // Si le match n'a pas encore été joué
302     else{
303         // Toutes les équipes reçoivent aucun point
304         $table[0]['points'] = 0;
305         $table[1]['points'] = 0;
306         $table[2]['points'] = 0;
307     }
308 }
309 }
310
311 return $table;
312 }
313
314 /**
315  * Met à jour la table du classement principale
316  *
317  * @param string id de l'équipe
318  * @param string les points à ajouter
319  * @param string les points marqués
320  * @param string les points reçus
321  * @param string le classement dans lequel ajouter les points
322  */
323 function UpdateMainTable($idTeam, $points_to_add, $points_m, $points_r,
    • $classement){
324     try {

```

```

325     $req = "UPDATE `Teams` SET `p_$classement` = $points_to_add, `m_$classement` =
    • $points_m, `r_$classement` = $points_r WHERE `id` = $idTeam";
326     $query = connecteur()->prepare($req);
327     $query->execute();
328     $query->fetch();
329 }
330 catch (Exception $e) {
331     error_log($e->getMessage());
332 }
333 }
334
335 /**
336  * Retorune l'id du Sport du match
337  *
338  * @param string id du match
339  * @return array un tableau avec l'id du match
340  *          ['idSport'] -> l'id du sport
341  */
342 function GetTypeOfClassement($idGame){
343     static $query = null;
344
345     if ($query == null) {
346         $req = "SELECT `idSport` as sport FROM `Games` WHERE `id` = :idGame";
347         $query = connecteur()->prepare($req);
348     }
349     try {
350         $query->bindParam(':idGame', $idGame, PDO::PARAM_STR);
351         $query->execute();
352         $res = $query->fetch(PDO::FETCH_ASSOC);
353     }
354     catch (Exception $e) {
355         error_log($e->getMessage());
356         $res = false;
357     }
358     return $res;
359 }
360
361 /**
362  * Met à jour le champ "played" de la tables games
363  * Met le champ à 1, ce qui indique que le match à été joué
364  *
365  * @param string id du match (game)
366  */
367 function SetGameToDone($id){
368     static $query = null;
369
370     if ($query == null) {
371         $req = "UPDATE `Games` SET `played`= 1 WHERE `id` = :id";
372         $query = connecteur()->prepare($req);
373     }
374     try {
375         $query->bindParam(':id', $id, PDO::PARAM_STR);
376         $query->execute();
377         $query->fetch();
378     }
379     catch (Exception $e) {
380         error_log($e->getMessage());
381     }

```

```

382 }
383
384 /**
385  * Retourne le Score pour une équipe dans un match (game)
386  *
387  * @param string id de l'équipe
388  * @param string id du match (game)
389  * @return array un tableau avec le scores
390  *          ['score'] -> le score de l'équipe dans le match
391  */
392 function GetScoreForTeamInMatch($idTeam, $idGame){
393     static $query = null;
394
395     if ($query == null) {
396         $req = "SELECT `score` FROM `plays` WHERE `idTeam` = :idTeam AND `idGame` =
        • :idGame";
397         $query = connecteur()->prepare($req);
398     }
399     try {
400         $query->bindParam(':idTeam', $idTeam, PDO::PARAM_STR);
401         $query->bindParam(':idGame', $idGame, PDO::PARAM_STR);
402         $query->execute();
403         $res = $query->fetch(PDO::FETCH_ASSOC);
404     }
405     catch (Exception $e) {
406         error_log($e->getMessage());
407         $res = false;
408     }
409     return $res;
410 }
411

```