

OverwatchCollection

Code source

Sven Wikberg

19/06/2017

Table des matières

Page d'un héros.....	5
Page des événements.....	6
Page d'un événement.....	7
Page des autres objets	8
Page utilisateur (login/sign in et my account).....	9
Page administrateur.....	12
Fonction ajout ou suppression d'objet pour un utilisateur	14
Classe MyPdo	15
Classe Oc_Dao	16
Class Oc_Display.....	27
Fichier de config base de données	38
Style du site web (CSS)	39

Page index.php

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\index.php

lundi 19 juin 2017 10:37

```
1 <!--
2     Auteur: Sven Wikberg
3     Date: 19/06/2017
4     Description: Page d'accueil
5 -->
6 <!doctype html>
7 <?php
8 session_start();
9
10 require_once ('class/class.oc_dao.php');
11 require_once ('class/class.oc_display.php');
12 ?>
13 <html lang="en">
14     <head>
15         <meta charset="utf-8">
16         <title>OverwatchCollection</title>
17         <link rel="stylesheet" href="css/style-main.css">
18     </head>
19     <body>
20         <header>
21             <?php
22                 OcDisplay::DisplayNavbar ();
23             ?>
24         </header>
25         <section id="index">
26             <h1>Overwatch Collection</h1>
27             <div class="flex_row">
28                 <div style="width: 70%;">
29                     <p>Bienvenue sur Overwatch Collection! Le site qui vous permet
30                         enfin d'avoir un suivi de vos objets dans Overwatch™.
31                         Vous n'avez pas besoins de créer compte si vous voulez seulement
32                         naviguer parmis les héros et les objets, par contre
33                         si vous voulez pouvoir sélectionner des objets et avoir un
34                         suivis ainsi que des statistiques, vous êtes obligé d'avoir
35                         compte.</p>
36                     <p>Ce site est divisé en plusieurs parties, d'abords il y a la
37                         partie "Heroes", cette partie contient les héros du jeu, pour
38                         chaque
39                         héro il y a un résumé, quelques informations et évidemment la
40                         liste, triée pas rareté, des objets du héros en question.
41                         <br>Ensuite, il y a la partie "Events" qui contient les
42                         événements du jeu, comme par exemple Halloween ou Noël, et de la
43                         même manière
44                         que pour les héros, il y a la liste des objets de l'événement.
45                         <br>Après ça il y a la partie "Rewards" qui contiens les objets
46                         qui ne sont relié à aucun héros, comme les icons de joueurs par
47                         exemple.
48                         <br>Et finalement il y a la partie "My Account" qui est
49                         accessible uniquement pour les utilisateurs connecté, qui
50                         affiche
51                         entre autre des statistiques sur les objets sélectionné et un
52                         classement des utilisateurs.</p>
53                 </div>
54                 <div style="width: 25%;">
55                     <h2>Users list</h2>
56                     <?php
57                         OcDisplay::DisplayListUsers ();
58                 </div>
59             </div>
60         </section>
61     </body>
62 
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\index.php

lundi 19 juin 2017 10:37

```
46             ?>
47         </div>
48     </div>
49     </section>
50     <footer>
51         <p>This site is not affiliated with Blizzard Entertainement. ©2016
52             Blizzard Entertainment, Inc. All rights reserved /
53             Author : Sven Wikberg </p>
54     </footer>
55 </body>
</html>
```

Page des héros

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\heroes.php                                         lundi 19 juin 2017 10:45
1  <!--
2      Auteur: Sven Wikberg
3      Date: 19/06/2017
4      Description: Page des heros
5  -->
6  <!doctype html>
7  <?php
8  session_start();
9
10 require_once ('class/class.oc_dao.php');
11 require_once ('class/class.oc_display.php');
12 ?>
13 <html lang="en">
14     <head>
15         <meta charset="utf-8">
16         <title>OverwatchCollection</title>
17         <link rel="stylesheet" href="css/style-main.css">
18     </head>
19     <body>
20         <header>
21             <?php
22                 OcDisplay::DisplayNavbar ();
23             ?>
24         </header>
25         <section id="heroes">
26             <?php
27                 OcDisplay::DisplayHeroesByRole (4);
28             ?>
29         </section>
30         <footer>
31             <p>This site is not affiliated with Blizzard Entertainement. ©2016
32             Blizzard Entertainment, Inc. All rights reserved /
33             Author : Sven Wikberg </p>
34         </footer>
35     </body>
36 </html>
```

Page d'un héros

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\hero.php

lundi 19 juin 2017 10:48

```
1  <!--
2   Auteur: Sven Wikberg
3   Date: 19/06/2017
4   Description: Page d'un heros
5   -->
6   <!doctype html>
7   <?php
8   session_start();
9
10  require_once('class/class.oc_dao.php');
11  require_once('class/class.oc_display.php');
12  require_once('function/func.user_reward.php');
13 ?>
14  <html lang="en">
15    <head>
16      <meta charset="utf-8">
17      <title>OverwatchCollection</title>
18      <link rel="stylesheet" href="css/style-main.css">
19    </head>
20    <body>
21      <header>
22        <?php
23          OcDisplay::DisplayNavbar();
24        ?>
25      </header>
26      <section id="hero_info">
27        <?php
28          OcDisplay::DisplayHeroInfo($_GET['id']);
29        ?>
30      </section>
31      <section id="hero_abilities">
32        <?php
33          OcDisplay::DisplayHeroAbilities($_GET['id']);
34        ?>
35      </section>
36      <section id="rewards">
37        <?php
38          OcDisplay::DisplayHeroRewards($_GET['id']);
39        ?>
40      </section>
41      <footer>
42        <p>This site is not affiliated with Blizzard Entertainment. ©2016
43          Blizzard Entertainment, Inc. All rights reserved /
44          Author : Sven Wikberg </p>
45      </footer>
46    </body>
47  </html>
```

Page des événements

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\events.php

lundi 19 juin 2017 10:49

```
1  <!--
2   Auteur: Sven Wikberg
3   Date: 19/06/2017
4   Description: Page des evenements
5 -->
6 <!doctype html>
7 <?php
8 session_start();
9
10 require_once ('class/class.oc_dao.php');
11 require_once ('class/class.oc_display.php');
12 ?>
13 <html lang="en">
14   <head>
15     <meta charset="utf-8">
16     <title>OverwatchCollection</title>
17     <link rel="stylesheet" href="css/style-main.css">
18   </head>
19   <body>
20     <header>
21       <?php
22         OcDisplay::DisplayNavbar ();
23       ?>
24     </header>
25     <section id="events">
26       <?php
27         OcDisplay::DisplayEvents (2);
28       ?>
29     </section>
30     <footer>
31       <p>This site is not affiliated with Blizzard Entertainement. ©2016
32           Blizzard Entertainment, Inc. All rights reserved /
33           Author : Sven Wikberg </p>
34     </footer>
35   </body>
36 </html>
```

Page d'un événement

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\event.php

lundi 19 juin 2017 10:50

```
1  <!--
2      Auteur: Sven Wikberg
3      Date: 19/06/2017
4      Description: Page d'un evenement
5  -->
6  <!doctype html>
7  <?php
8  session_start();
9
10 require_once('class/class.oc_dao.php');
11 require_once('class/class.oc_display.php');
12 require_once('function/func.user_reward.php');
13
14 ?>
15 <html lang="en">
16     <head>
17         <meta charset="utf-8">
18         <title>OverwatchCollection</title>
19         <link rel="stylesheet" href="css/style-main.css">
20     </head>
21     <body>
22         <header>
23             <?php
24                 OcDisplay::DisplayNavbar();
25             ?>
26         </header>
27         <section id="event_info">
28             <?php
29                 OcDisplay::DisplayEventInfo($_GET['id']);
30             ?>
31         </section>
32         <section id="rewards">
33             <?php
34                 OcDisplay::DisplayEventRewards($_GET['id']);
35             ?>
36         </section>
37         <footer>
38             <p>This site is not affiliated with Blizzard Entertainement. ©2016
39             Blizzard Entertainment, Inc. All rights reserved /
40             Author : Sven Wikberg </p>
41         </footer>
42     </body>
43 </html>
```

Page des autres objets

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\rewards.php  
lundi 19 juin 2017 10:51  
1  <!--  
2      Auteur: Sven Wikberg  
3      Date: 19/06/2017  
4      Description: Page des autres objets cosmétiques  
5  -->  
6  <!doctype html>  
7  <?php  
8  session_start();  
9  
10 require_once('class/class.oc_dao.php');  
11 require_once('class/class.oc_display.php');  
12 require_once('function/func.user_reward.php');  
13 ?>  
14 <html lang="en">  
15     <head>  
16         <meta charset="utf-8">  
17         <title>OverwatchCollection</title>  
18         <link rel="stylesheet" href="css/style-main.css">  
19     </head>  
20     <body>  
21         <header>  
22             <?php  
23                 OcDisplay::DisplayNavbar();  
24             ?>  
25         </header>  
26         <section id="others_rewards_info">  
27             <h1>Others Rewards</h1>  
28             <p>Certains des objets cosmétiques dans Overwatch n'ont pas d'héros  
29                 associé, par exemple les icônes d'utilisateurs, étant donnée qu'elle  
                  sont faites pour le compte et pas un héros spécifique. C'est sur cette  
                  page qu'elle seront répertoriées, triées par catégories et par raretés.  
                  </p>  
30         </section>  
31         <section id="rewards">  
32             <?php  
33                 OcDisplay::DisplayOtherRewards();  
34             ?>  
35         </section>  
36         <footer>  
37             <p>This site is not affiliated with Blizzard Entertainment. ©2016  
38                 Blizzard Entertainment, Inc. All rights reserved /  
39                 Author : Sven Wikberg </p>  
40         </footer>  
41     </body>  
42 </html>
```

Page utilisateur (login/sign in et my account)

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\user.php
lundi 19 juin 2017 10:53

1  <!--
2   Auteur: Sven Wikberg
3   Date: 19/06/2017
4   Description: Page utilisateur
5 -->
6 <!doctype html>
7 <?php
8 session_start();
9
10 require_once('class/class.oc_dao.php');
11 require_once('class/class.oc_display.php');
12
13 if (isset($_GET['action'])) { // selon l'action, la page recuper, teste ou process
des données différentes
14     $myget = '';
15     if($_GET['action'] == 'login'){ // l'action login teste les données entrées par
l'utilisateur afin de le connecter ou pas
16         if (isset($_POST['username']) && isset($_POST['password'])) {
17             $username = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_ENCODED);
18             $password = sha1(filter_input(INPUT_POST, 'password',
FILTER_SANITIZE_ENCODED));
19
20             $user = OcDao::SelectUserByUsername($username); // on récupere les
données de l'utilisateur grace a son nom d'utilisateur
21
22             if($user == null) // si l'on ne récupère rien, ca veut dire que ce nom
d'utilisateur n'existe pas, donc on informe l'utilisateur grace a
l'erreur en GET
23                 $myGet = '?msg=wrongUn';
24             elseif ($user['password'] == $password) { // si tout est juste
25                 if($user['is_banned'] == 1) { // si l'utilisateur est banni
26                     $myGet = '?msg=banned';
27                 } else{ // si l'utilisateur n'est pas banni, on le connecte
28                     $_SESSION['id_connected'] = $user['id_user'];
29                 }
30             }
31             else { // si le mot de passe est faux on met l'erreur en GET afin
d'informer l'utilisateur
32                 $myGet = '?msg=wrongPw';
33             }
34         }
35     } elseif ($_GET['action'] == 'signin') { // l'action sign in ajoute un nouvel
utilisateur la dans la base avec les données entrées
36         if (isset($_POST['username']) && isset($_POST['password']) && isset($_POST[
'email'])) {
37             $username = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_ENCODED);
38             $password = sha1(filter_input(INPUT_POST, 'password',
FILTER_SANITIZE_ENCODED));
39             $email = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL);
40
41             $tmp = OcDao::InsertUser($username, $password, $email);
42             if($tmp == null){
43                 $myGet = '?msg=signOk';
44             } else {
45                 if($tmp == 1062){
46                     $myGet = '?msg=duplicate';
47                 }
48             }
49         }
50     }
51 }
52
53 
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\user.php

lundi 19 juin 2017 10:53

```
48         }
49     }
50 } elseif ($_GET['action'] == 'deco') { // l'action deco met la valeur de
51 // l'id_connected a null, id_connected qui donne l'info de l'id de utilisateur
52 // connecté
53     session_destroy();
54 } elseif ($_GET['action'] == 'update') { // l'action update met à jour les
55 // données de l'utilisateur avec les données qu'il a entrées
56     if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null){
57         // si l'utilisateur est connecter
58         if(isset($_POST['username']) && isset($_POST['email'])){
59             $username = filter_input(INPUT_POST, 'username',
60 FILTER_SANITIZE_ENCODED);
61             $email = filter_input(INPUT_POST, 'email');
62
63             $tmp = OcDao::UpdateUserByIdNoPwd($_SESSION['id_connected'],
64             $username, $email);
65             if($tmp == null){
66                 $myGet = '?msg=updateOK';
67             } else {
68                 if($tmp == 1062){
69                     $myGet = '?msg=updateDuplicate';
70                 }
71             }
72         }
73     }
74     header('Location: user.php' . $myGet);
75 }
76 ?>
77 <html lang="en">
78     <head>
79         <meta charset="utf-8">
80         <title>OverwatchCollection</title>
81         <link rel="stylesheet" href="css/style-main.css">
82     </head>
83     <body>
84         <header>
85             <?php
86                 OcDisplay::DisplayNavbar();
87             ?>
88         </header>
89         <?php
90             if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null){
91                 // si l'utilisateur est connecter il peut se deconnecter
92                 if(isset($_GET['goto'])){
93                     if($_GET['goto'] == 'updating'){
94                         OcDisplay::DisplayAccountInfoUpdating($_SESSION['id_connected']);
95                     }
96                 } else {
97                     OcDisplay::DisplayAccountInfo($_SESSION['id_connected']);
98
99                     OcDisplay::DisplayAccountStats($_SESSION['id_connected']);
100                }
101            } else {
102                if(isset($_GET['msg'])) {
103                    echo $_GET['msg'];
104                }
105            }
106        </?php>
107    </body>
108 </html>
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\user.php                                         lundi 19 juin 2017 10:53
98      if($_GET['msg'] == 'signOk') // l'utilisateur a bien ete ajoute a la
99          base de donnees'
100         echo '<h2>Account created</h2>';
101         OcDisplay::DisplayLogin();
102     OcDisplay::DisplaySignin();
103 }
104 ?>
105 <footer>
106     <p>This site is not affiliated with Blizzard Entertainment. ©2016
107     Blizzard Entertainment, Inc. All rights reserved /
108     Author : Sven Wikberg </p>
109 </footer>
110 </body>
111 </html>
```

Page administrateur

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\admin.php  
lundi 19 juin 2017 10:55  
1  <!--  
2      Auteur: Sven Wikberg  
3      Date: 19/06/2017  
4      Description: Page administrateur  
5  -->  
6  <!doctype html>  
7  <?php  
8  session_start();  
9  
10  
11 require_once('class/class.oc_dao.php');  
12 require_once('class/class.oc_display.php');  
13  
14 if(isset($_SESSION['id_connected'])){ // si pas admin, on ne peut pas accéder à  
15 cette page  
16     $user = OcDao::SelectUserById($_SESSION['id_connected']);  
17     if($user['is_admin'] == 0){  
18         header('Location: index.php');  
19     }  
20 } else {  
21     header('Location: index.php');  
22 }  
23 if (isset($_GET['action'])) { // selon l'action, la page récupère, teste ou process  
des données différentes  
24     if($_GET['action'] == 'ban'){ // on banni l'utilisateur sélectionné  
25         if(isset($_GET['id'])){  
26             OcDao::BanUserById($_GET['id']);  
27         }  
28     } elseif($_GET['action'] == 'unban'){ // on débanni l'utilisateur sélectionné  
29         if(isset($_GET['id'])){  
30             OcDao::UnbanUserById($_GET['id']);  
31         }  
32     } elseif($_GET['action'] == 'delete'){ // on supprime l'utilisateur sélectionné  
33         if(isset($_GET['id'])){  
34             OcDao::DeleteUserById($_GET['id']);  
35         }  
36     }  
37     header('Location: admin.php');  
38 }  
39 ?>  
40 <html lang="en">  
41     <head>  
42         <meta charset="utf-8">  
43         <title>OverwatchCollection</title>  
44         <link rel="stylesheet" href="css/style-main.css">  
45     </head>  
46     <body>  
47         <header>  
48             <?php  
49             OcDisplay::DisplayNavbar();  
50             ?>  
51         </header>  
52         <section id="admin">  
53             <h1>Admin Page</h1>  
54             <table border="1" style="width:100%;">  
55                 <tr>
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\admin.php                                         lundi 19 juin 2017 10:55
56          <td><h2>Clean Users</h2></td>
57          <td><h2>Banned Users</h2></td>
58      </tr>
59      <tr>
60          <td>
61              <?php
62                  OcDisplay::DisplayListCleanUserWithBan ();
63              ?>
64          </td>
65          <td>
66              <?php
67                  OcDisplay::DisplayListBannedUserWithUnbanDelete ();
68              ?>
69          </td>
70      </tr>
71  </table>
72 </section>
73 <footer>
74     <p>This site is not affiliated with Blizzard Entertainment. ©2016
    Blizzard Entertainment, Inc. All rights reserved /
    Author : Sven Wikberg </p>
75 </footer>
76 </body>
77 </html>
```

Fonction ajout ou suppression d'objet pour un utilisateur

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\function\func.user_reward.php

lundi 19 juin 2017 10:56

```
1 <?php
2 /*
3     Auteur: Sven Wikberg
4     Date: 19/06/2017
5     Description: Fonction de ajout ou de suppression d'objets pour un utilisateur
6 */
7 if (isset($_GET['action'])) {
8     if($_GET['action'] == 'add_user_reward' && isset($_GET['id_reward'])){
9         if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null){
10             // si l'utilisateur est connecter
11             $id_user = $_SESSION['id_connected'];
12             $id_reward = $_GET['id_reward'];
13
14             if(OcDao::IsCreatedUserReward($id_user, $id_reward)){ // test si
15                 l'enregistrement est deja crée
16                 OcDao::DeleteUserReward($id_user, $id_reward); // si c'est le cas on
17                 l'enlève
18             } else {
19                 OcDao::InsertUserReward($id_user, $id_reward); // si ce n'est pas le
20                 cas on l'ajoute
21             }
22     }
23 ?>
```

Classe MyPdo

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.my_pdo.php  
lundi 19 juin 2017 10:59  
1 <?php  
2 /*  
3     Auteur: Sven Wikberg  
4     Date: 19/06/2017  
5     Description: classe de connexion a la base de données  
6 */  
7 require_once('config/config_db.php');  
8  
9 class MyPdo{  
10     private static $_myPdo;  
11  
12     public static function GetMyPdo(){  
13         try {  
14             if (!isset($_myPdo)) {  
15                 $dbc = new PDO('mysql:host=' . DB_HOST . ';dbname=' . DB_NAME . '',  
16                             DB_USER, DB_PWD, array(  
17                             PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8',  
18                             PDO::ATTR_PERSISTENT => true,  
19                             PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION));  
20             }  
21         } catch (PDOException $e) {  
22             print "Erreur !: " . $e->getMessage() . "<br/>";  
23             die();  
24         }  
25         return $dbc;  
26     }  
27 }
```

Classe Oc_Dao

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php  
lundi 19 juin 2017 11:01  
1 <?php  
2 /*  
3     Auteur: Sven Wikberg  
4     Date: 19/06/2017  
5     Description: Classe de récupération des données  
6 */  
7  
8 require_once('class/class.my_pdo.php');  
9  
10 class OcDao{  
11  
12     // récupère tous les heros dans l'ordre de base/de l'id  
13     static function SelectHeroes() {  
14         $req = 'SELECT id_hero, name, description, id_role, health, armour, shield,  
15             real_name, age, height, affiliation, base_of_operations, difficulty  
16             FROM heroes';  
17         $sql = MyPdo::GetMyPdo() ->prepare($req);  
18         $sql->execute();  
19  
20         return $sql->fetchAll(PDO::FETCH_ASSOC);  
21     }  
22  
23     // récupère un heros grace a son id  
24     static function SelectHeroById($id) {  
25         $req = 'SELECT id_hero, name, description, id_role, health, armour, shield,  
26             real_name, age, height, affiliation, base_of_operations, difficulty  
27             FROM heroes  
28             WHERE id_hero = :id';  
29         $sql = MyPdo::GetMyPdo() ->prepare($req);  
30         $sql->bindParam(':id', $id, PDO::PARAM_INT);  
31         $sql->execute();  
32  
33         $tmpReturn = $sql->fetch(PDO::FETCH_ASSOC);  
34  
35         if(count($tmpReturn) > 0){  
36             return $tmpReturn;  
37         } else {  
38             return false;  
39         }  
40  
41     // récupere les capacités d'un hero grace a son id  
42     static function SelectAbilitiesByIdHero($id) {  
43         $req = 'SELECT abilities.id_ability, abilities.name, abilities.description,  
44             abilities.id_hero, abilities.is_ultimate  
45             FROM abilities  
46             WHERE abilities.id_hero = :id  
47             ORDER BY abilities.is_ultimate';  
48         $sql = MyPdo::GetMyPdo() ->prepare($req);  
49         $sql->bindParam(':id', $id, PDO::PARAM_INT);  
50         $sql->execute();  
51  
52         $tmpReturn = $sql->fetchAll(PDO::FETCH_ASSOC);  
53  
54         if(count($tmpReturn) > 0){  
55             return $tmpReturn;  
56         } else {
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php

lundi 19 juin 2017 11:01

```
55         return false;
56     }
57 }
58
59 // récupère tous les roles dans l'ordre de base/de l'id
60 static function SelectRoles() {
61     $req = 'SELECT id_role, name FROM roles';
62     $sql = MyPdo::GetMyPdo()->prepare($req);
63     $sql->execute();
64
65     return $sql->fetchAll(PDO::FETCH_ASSOC);
66 }
67
68 // récupère toutes les qualités/raretés des objets dans l'ordre de base/de l'id
69 static function SelectQualities() {
70     $req = 'SELECT id_quality, name FROM qualities';
71     $sql = MyPdo::GetMyPdo()->prepare($req);
72     $sql->execute();
73
74     return $sql->fetchAll(PDO::FETCH_ASSOC);
75 }
76
77 // récupère tous les type d'objets dans l'ordre de base/de l'id
78 static function SelectRewardTypes() {
79     $req = 'SELECT id_reward_type, name FROM reward_types';
80     $sql = MyPdo::GetMyPdo()->prepare($req);
81     $sql->execute();
82
83     return $sql->fetchAll(PDO::FETCH_ASSOC);
84 }
85
86 // récupère tous les événements dans l'ordre chronologique de la date de debut
87 static function SelectEvents() {
88     $req = 'SELECT id_event, name, start_date, end_date FROM events ORDER BY
events.start_date ASC';
89     $sql = MyPdo::GetMyPdo()->prepare($req);
90     $sql->execute();
91
92     return $sql->fetchAll(PDO::FETCH_ASSOC);
93 }
94
95 // récupère un evenement grace a son id
96 static function SelectEventById($id) {
97     $req = 'SELECT id_event, name, start_date, end_date FROM events WHERE
id_event = :id';
98     $sql = MyPdo::GetMyPdo()->prepare($req);
99     $sql->bindParam(':id', $id, PDO::PARAM_INT);
100    $sql->execute();
101
102    $tmpReturn = $sql->fetch(PDO::FETCH_ASSOC);
103
104    if(count($tmpReturn) > 0) {
105        return $tmpReturn;
106    } else {
107        return false;
108    }
109 }
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php

lundi 19 juin 2017 11:01

```
110 // récupère tous les utilisateurs dans l'ordre alphabétique
111 static function SelectUsers() {
112     $req = 'SELECT id_user, username, email, is_banned, is_admin FROM users
113     ORDER BY username ASC';
114     $sql = MyPdo::GetMyPdo()->prepare($req);
115     $sql->execute();
116
117     return $sql->fetchAll(PDO::FETCH_ASSOC);
118 }
119
120 // récupère tous les utilisateurs non-bannis dans l'ordre alphabétique (avec les
121 // admin en premier)
121 static function SelectCleanUsers() {
122     $req = 'SELECT id_user, username, email, is_banned, is_admin FROM users
123     WHERE is_banned = 0 ORDER BY is_admin DESC, username ASC';
124     $sql = MyPdo::GetMyPdo()->prepare($req);
125     $sql->execute();
126
127     return $sql->fetchAll(PDO::FETCH_ASSOC);
128 }
129
130 // récupère tous les utilisateurs non-bannis dans l'ordre alphabétique (qui ne
131 // sont pas admin)
132 static function SelectCleanUsersNoAdmin() {
133     $req = 'SELECT id_user, username, email, is_banned, is_admin FROM users
134     WHERE is_banned = 0 AND is_admin = 0 ORDER BY username ASC';
135     $sql = MyPdo::GetMyPdo()->prepare($req);
136     $sql->execute();
137
138     return $sql->fetchAll(PDO::FETCH_ASSOC);
139 }
140
141 // récupère tous les utilisateurs bannis dans l'ordre alphabétique (qui ne sont
142 // pas admin)
143 static function SelectBannedUsers() {
144     $req = 'SELECT id_user, username, email, is_banned, is_admin FROM users
145     WHERE is_banned = 1 AND is_admin = 0 ORDER BY username ASC';
146     $sql = MyPdo::GetMyPdo()->prepare($req);
147     $sql->execute();
148
149     return $sql->fetchAll(PDO::FETCH_ASSOC);
150 }
151
152 // modifie le champ is_banned de l'utilisateur sélectionné, en le passant à 1
153 // (pour bannir l'utilisateur)
154 static function BanUserById($id) {
155     $req = 'UPDATE users SET is_banned=1 WHERE id_user = :id';
156     $sql = MyPdo::GetMyPdo()->prepare($req);
157     $sql->bindParam(':id', $id, PDO::PARAM_INT);
158     $sql->execute();
159
160     // modifie le champ is_banned de l'utilisateur sélectionné, en le passant à 0
161     // (pour débannir l'utilisateur)
162     static function UnbanUserById($id) {
163         $req = 'UPDATE users SET is_banned=0 WHERE id_user = :id';
```

C:\Users\wikbergs.info\Desktop\OverwatchCollection\web\class\class.oc_dao.php

lundi 19 juin 2017 11:01

```
158     $sql = MyPdo::GetMyPdo()->prepare($req);
159     $sql->bindParam(':id', $id, PDO::PARAM_INT);
160     $sql->execute();
161 }
162
163 // supprime l'utilisateur selectionné
164 static function DeleteUserById($id) {
165     $req = 'DELETE FROM users WHERE id_user = :id';
166     $sql = MyPdo::GetMyPdo()->prepare($req);
167     $sql->bindParam(':id', $id, PDO::PARAM_INT);
168     $sql->execute();
169 }
170
171 // modifie l'enregistrement d'un utilisateur en les replacant par les parametres
172 // de la fonction
173 static function UpdateUserByIdNoPwd($id, $username, $email){
174     try{
175         $req = "UPDATE users SET username='$username', email='$email' WHERE
176             id_user=:id";
177         $sql = MyPdo::GetMyPdo()->prepare($req);
178         $sql->bindParam(':id', $id, PDO::PARAM_INT);
179         $sql->execute();
180     } catch (PDOException $e) {
181         print_rr($e);
182         return $e->errorInfo[1];
183     }
184     return null;
185 }
186
187 // recuper tous les heros et les range par role dans un tableau
188 static function SelectHeroesInArrayOfRole() {
189     $req = 'SELECT id_hero, name, id_role FROM heroes WHERE id_role = :id';
190     $sql = MyPdo::GetMyPdo()->prepare($req);
191     $tmpReturn = Array();
192
193     foreach (OcDao::SelectRoles() as $role) {
194         $sql->bindParam(':id', $role['id_role'], PDO::PARAM_INT);
195         $sql->execute();
196         $tmpReturn[$role['name']] = $sql->fetchAll(PDO::FETCH_ASSOC);
197     }
198
199     return $tmpReturn;
200 }
201
202 // ajoute un enregistrement a la table de liaison "users_rewards"
203 static function InsertUserReward($id_user, $id_reward){
204     $req = 'INSERT INTO users_rewards(id_user, id_reward) VALUES
205         (:id_user,:id_reward)';
206     $sql = MyPdo::GetMyPdo()->prepare($req);
207     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
208     $sql->bindParam(':id_reward', $id_reward, PDO::PARAM_INT);
209     $sql->execute();
210
211     // supprime un enregistrement a la table de liaison "users_rewards"
212     static function DeleteUserReward($id_user, $id_reward){
213         $req = 'DELETE FROM users_rewards WHERE id_user = :id_user AND id_reward =
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php

lundi 19 juin 2017 11:01

```
        :id_reward';
212     $sql = MyPdo::GetMyPdo()->prepare($req);
213     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
214     $sql->bindParam(':id_reward', $id_reward, PDO::PARAM_INT);
215     $sql->execute();
216 }
217
218 // test si un enregistrement de la table de liaison "users_rewards" existe, oui
219 // -> true / non -> false
220 static function IsCreatedUserReward($id_user, $id_reward){
221     $req = 'SELECT users_rewards.id_user, users_rewards.id_reward
222           FROM users_rewards
223           WHERE id_user = :id_user
224             AND id_reward = :id_reward';
225     $sql = MyPdo::GetMyPdo()->prepare($req);
226     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
227     $sql->bindParam(':id_reward', $id_reward, PDO::PARAM_INT);
228     $sql->execute();
229
230     if($sql->fetchAll(PDO::FETCH_ASSOC) == NULL) {
231         return false;
232     } else {
233         return true;
234     }
235
236 // recuper les id des objets d'un hero qu'un utilisateur a selectionné
237 static function SelectIdRewardsByIdHeroAndidUser($id_hero, $id_user){
238     $req = 'SELECT rewards.id_reward
239           FROM rewards
240           JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
241           WHERE rewards.id_hero = :id_hero
242             AND users_rewards.id_user = :id_user';
243     $sql = MyPdo::GetMyPdo()->prepare($req);
244     $sql->bindParam(':id_hero', $id_hero, PDO::PARAM_INT);
245     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
246     $sql->execute();
247
248     $tmpReturn = NULL;
249
250     $cpt = 0;
251     foreach($sql->fetchAll(PDO::FETCH_ASSOC) as $array) {
252         $tmpReturn[$cpt] = $array['id_reward'];
253
254         $cpt++;
255     }
256
257     return $tmpReturn;
258 }
259
260 // recuper les id des objets d'un evenement qu'un utilisateur a selectionné
261 static function SelectIdRewardsByIdEventAndidUser($id_event, $id_user){
262     $req = 'SELECT rewards.id_reward
263           FROM rewards
264           JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
265           WHERE rewards.id_event = :id_event
266             AND users_rewards.id_user = :id_user';
```

C:\Users\wikbergs.info\Desktop\OverwatchCollection\web\class\class.oc_dao.php

lundi 19 juin 2017 11:01

```
267     $sql = MyPdo::GetMyPdo()->prepare($req);
268     $sql->bindParam(':id_event', $id_event, PDO::PARAM_INT);
269     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
270     $sql->execute();
271
272     $tmpReturn = NULL;
273
274     $cmpt = 0;
275     foreach($sql->fetchAll(PDO::FETCH_ASSOC) as $array) {
276         $tmpReturn[$cmpt] = $array['id_reward'];
277
278         $cmpt++;
279     }
280
281     return $tmpReturn;
282 }
283
284 // recuper les id des objets d'aucun hero qu'un utilisateur a selectionné
285 static function SelectIdRewardsByNoIdHeroAndidUser($id_user) {
286     $req = 'SELECT rewards.id_reward
287             FROM rewards
288             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
289             WHERE (rewards.id_hero = 0 OR rewards.id_hero = NULL)
290             AND users_rewards.id_user = :id_user';
291     $sql = MyPdo::GetMyPdo()->prepare($req);
292     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
293     $sql->execute();
294
295     $tmpReturn = NULL;
296
297     $cmpt = 0;
298     foreach($sql->fetchAll(PDO::FETCH_ASSOC) as $array) {
299         $tmpReturn[$cmpt] = $array['id_reward'];
300
301         $cmpt++;
302     }
303
304     return $tmpReturn;
305 }
306
307 // recuper tous les objets d'un heros et les range d'abord par catégorie et
308 // ensuite par rareté
309 static function SelectRewardsInArrayOfQualityAndTypeByIdHero($id) {
310     $req = 'SELECT rewards.id_reward, rewards.name, rewards.cost,
311             rewards.id_currency, rewards.id_event
312             FROM rewards
313             WHERE rewards.id_quality = :id_quality
314             AND rewards.id_reward_type = :id_reward_type
315             AND rewards.id_hero = :id_hero
316             ORDER BY rewards.name';
317     $sql = MyPdo::GetMyPdo()->prepare($req);
318
319     $rewardTypes = OcDao::SelectRewardTypes();
320     $qualities = OcDao::SelectQualities();
321     $tmpReturn = Array();
322
323     foreach ($rewardTypes as $rewardType) {
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php

lundi 19 juin 2017 11:01

```
322     foreach ($qualities as $quality) {
323         $sql->bindParam(':id_reward_type', $rewardType['id_reward_type'], PDO
324             ::PARAM_INT);
325         $sql->bindParam(':id_quality', $quality['id_quality'], PDO::PARAM_INT
326             );
327         $sql->bindParam(':id_hero', $id, PDO::PARAM_INT);
328         $sql->execute();
329
330         $tmp = $sql->fetchAll(PDO::FETCH_ASSOC); // afin de ne pas mettre du
331         // vide dans le tableau
332         if(isset($tmp[0]))
333             $tmpReturn[$rewardType['name']][$quality['name']] = $tmp;
334     }
335
336     if(count($tmpReturn) > 0) {
337         return $tmpReturn;
338     } else {
339         return false;
340     }
341
342     // récupère tous les objets qui ne sont pas associé a un hero et les range
343     // d'abord par catégorie et ensuite par rareté
344     static function SelectRewardsInArrayOfQualityAndTypeByNoIdHero() {
345         $req = 'SELECT rewards.id_reward, rewards.name, rewards.cost,
346             rewards.id_currency, rewards.id_event
347             FROM rewards
348             WHERE rewards.id_quality = :id_quality
349             AND rewards.id_reward_type = :id_reward_type
350             AND (rewards.id_hero = 0 OR rewards.id_hero = NULL)
351             ORDER BY rewards.name';
352         $sql = MyPdo::GetMyPdo()->prepare($req);
353
354         $rewardTypes = OcDao::SelectRewardTypes();
355         $qualities = OcDao::SelectQualities();
356
357         foreach ($rewardTypes as $rewardType) {
358             foreach ($qualities as $quality) {
359                 $sql->bindParam(':id_reward_type', $rewardType['id_reward_type'], PDO
360                     ::PARAM_INT);
361                 $sql->bindParam(':id_quality', $quality['id_quality'], PDO::PARAM_INT
362                     );
363                 $sql->execute();
364
365                 $tmp = $sql->fetchAll(PDO::FETCH_ASSOC); // afin de ne pas mettre du
366                 // vide dans le tableau
367                 if(isset($tmp[0]))
368                     $tmpReturn[$rewardType['name']][$quality['name']] = $tmp;
369             }
370         }
371
372         return $tmpReturn;
373     }
374
375     // récupère tous les objets d'un evenement et les range d'abord par catégorie et
376     // ensuite par rareté
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php lundi 19 juin 2017 11:01

```
370     static function SelectRewardsInArrayOfQualityAndTypeByIdEvent ($id) {
371
372         // il y a une jointure externe (LEFT JOIN) dans la requête car pas tous les
373         // objet qu'on veut récupérer on un hero qui leur est associé
374         $req = 'SELECT rewards.id_reward, rewards.name as r_name, rewards.cost,
375             rewards.id_currency, rewards.id_hero, heroes.name as h_name
376             FROM rewards
377             LEFT JOIN heroes ON heroes.id_hero = rewards.id_hero
378             WHERE rewards.id_quality = :id_quality
379             AND rewards.id_reward_type = :id_reward_type
380             AND rewards.id_event = :id_event
381             ORDER BY rewards.name';
382         $sql = MyPdo::GetMyPdo()->prepare($req);
383
384         $rewardTypes = OcDao::SelectRewardTypes();
385         $qualities = OcDao::SelectQualities();
386
387         foreach ($rewardTypes as $rewardType) {
388             foreach ($qualities as $quality) {
389                 $sql->bindParam(':id_reward_type', $rewardType['id_reward_type'], PDO::PARAM_INT);
390                 $sql->bindParam(':id_quality', $quality['id_quality'], PDO::PARAM_INT);
391                 $sql->bindParam(':id_event', $id, PDO::PARAM_INT);
392                 $sql->execute();
393
394                 $tmp = $sql->fetchAll(PDO::FETCH_ASSOC); // afin de ne pas mettre du
395                 // vide dans le tableau
396                 if(isset($tmp[0]))
397                     $tmpReturn[$rewardType['name']][$quality['name']] = $tmp;
398             }
399         }
400
401         if(count($tmpReturn) > 0){ // s'il n'y a rien, il y a un probleme, donc on
402             retourne false
403             return $tmpReturn;
404         } else {
405             return false;
406         }
407     }
408
409     // récupère les infos d'un utilisateur en fonction de son 'username' (utilisé
410     // pour la connection)
411     static function SelectUserByUsername($username) {
412         $req = 'SELECT id_user, username, password, email, is_banned, is_admin FROM
413             users WHERE username = :username';
414         $sql = MyPdo::GetMyPdo()->prepare($req);
415         $sql->bindParam(':username', $username, PDO::PARAM_STR);
416         $sql->execute();
417
418         return $sql->fetch(PDO::FETCH_ASSOC);
419     }
420
421     // récupère les infos d'un utilisateur en fonction de son id
422     static function SelectUserById($id) {
423         $req = 'SELECT id_user, username, password, email, is_banned, is_admin FROM
424             users WHERE id_user = :id';
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php

lundi 19 juin 2017 11:01

```
418     $sql = MyPdo::GetMyPdo()->prepare($req);
419     $sql->bindParam(':id', $id, PDO::PARAM_INT);
420     $sql->execute();
421
422     return $sql->fetch(PDO::FETCH_ASSOC);
423 }
424
425 // insère un nouvel enregistrement dans la table "users"
426 // retourne le code de l'erreur s'il y en a une, ou null s'il n'y en a pas
427 static function InsertUser($username, $email, $password) {
428     try{
429         $req = 'INSERT INTO users (username, password, email, is_banned) VALUES
430             (:username, :email, :password, 0)';
431         $sql = MyPdo::GetMyPdo()->prepare($req);
432         $sql->bindParam(':username', $username);
433         $sql->bindParam(':email', $email);
434         $sql->bindParam(':password', $password);
435         $sql->execute();
436     } catch (PDOException $e) {
437         print_rr($e);
438         return $e->errorInfo[1];
439     }
440     return null;
441 }
442
443 // compte le nombre d'objets en tous
444 static function SelectCountReward(){
445     $req = 'SELECT COUNT(id_reward) AS c FROM rewards';
446     $sql = MyPdo::GetMyPdo()->prepare($req);
447     $sql->execute();
448
449     return $sql->fetch(PDO::FETCH_ASSOC) ['c'];
450 }
451
452 // compte le nombre d'objets d'un utilisateur
453 static function SelectCountRewardByIdUser($id) {
454     $req = 'SELECT COUNT(id_reward) AS c FROM users_rewards WHERE id_user = :id';
455     $sql = MyPdo::GetMyPdo()->prepare($req);
456     $sql->bindParam(':id', $id, PDO::PARAM_INT);
457     $sql->execute();
458
459     return $sql->fetch(PDO::FETCH_ASSOC) ['c'];
460 }
461
462 // compte le nombre d'objets qu'a chaque evenement
463 static function SelectCountRewardEvents() {
464     $req = 'SELECT events.name, COUNT(rewards.id_reward) AS c
465             FROM rewards
466             JOIN events ON rewards.id_event = events.id_event
467             GROUP BY events.id_event
468             ORDER BY events.start_date';
469     $sql = MyPdo::GetMyPdo()->prepare($req);
470     $sql->execute();
471
472     return $sql->fetchAll(PDO::FETCH_ASSOC);
473 }
```

C:\Users\wikbergs.info\Desktop\OverwatchCollection\web\class\class.oc_dao.php lundi 19 juin 2017 11:01

```
474     // compte le nombre d'objets qu'a l'utilisateur pour chaque evenement
475     static function SelectCountRewardEventsByIdUser($id) {
476         // la 1ere partie de la requete recuper les evenements et les nombre
477         // d'objet de l'utilisateur pour cet evenement
478         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
479         // aucun objet
480         // on fait une union des deux pour avoir tous les evenement
481         $req = '(SELECT events.start_date, events.name,
482             COUNT(users_rewards.id_reward) AS c
483             FROM events
484             JOIN rewards ON rewards.id_event = events.id_event
485             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
486             WHERE users_rewards.id_user = :id1
487             GROUP BY events.id_event)
488             UNION
489             (SELECT events.start_date, events.name, 0 AS c
490             FROM events
491             WHERE events.id_event NOT IN
492                 (SELECT events.id_event
493                 FROM events
494                 JOIN rewards ON rewards.id_event = events.id_event
495                 JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
496                 WHERE users_rewards.id_user = :id2
497                 GROUP BY events.id_event))
498             ORDER BY start_date ASC';
499         $sql = MyPdo::GetMyPdo()->prepare($req);
500         $sql->bindParam(':id1', $id, PDO::PARAM_INT);
501         $sql->bindParam(':id2', $id, PDO::PARAM_INT);
502         $sql->execute();
503
504         return $sql->fetchAll(PDO::FETCH_ASSOC);
505     }
506
507     // compte le nombre d'objets qu'a chaque hero
508     static function SelectCountRewardHeroes() {
509         $req = 'SELECT heroes.name, COUNT(rewards.id_reward) AS c
510             FROM rewards
511             JOIN heroes ON rewards.id_hero = heroes.id_hero
512             GROUP BY heroes.id_hero
513             ORDER BY heroes.name';
514         $sql = MyPdo::GetMyPdo()->prepare($req);
515         $sql->execute();
516
517         return $sql->fetchAll(PDO::FETCH_ASSOC);
518     }
519
520     // compte le nombre d'objets qu'a l'utilisateur pour chaque hero
521     static function SelectCountRewardHeroesByIdUser($id) {
522         // la 1ere partie de la requete recuper les evenements et les nombre
523         // d'objet de l'utilisateur pour cet evenement
524         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
525         // aucun objet
526         // on fait une union des deux pour avoir tous les evenement
527         $req = '(SELECT heroes.name, COUNT(users_rewards.id_reward) AS c
528             FROM heroes
529             JOIN rewards ON rewards.id_hero = heroes.id_hero
530             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
531             WHERE users_rewards.id_user = :id1
532             GROUP BY heroes.name)
533             UNION
534             (SELECT heroes.name, COUNT(users_rewards.id_reward) AS c
535             FROM heroes
536             JOIN rewards ON rewards.id_hero = heroes.id_hero
537             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
538             WHERE users_rewards.id_user = :id2
539             GROUP BY heroes.name)
540             ORDER BY heroes.name ASC';
541         $sql = MyPdo::GetMyPdo()->prepare($req);
542         $sql->bindParam(':id1', $id, PDO::PARAM_INT);
543         $sql->bindParam(':id2', $id, PDO::PARAM_INT);
544         $sql->execute();
545
546         return $sql->fetchAll(PDO::FETCH_ASSOC);
547     }
548
549     // compte le nombre d'objets qu'a l'utilisateur pour chaque item
550     static function SelectCountRewardItemsByIdUser($id) {
551         // la 1ere partie de la requete recuper les evenements et les nombre
552         // d'objet de l'utilisateur pour cet evenement
553         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
554         // aucun objet
555         // on fait une union des deux pour avoir tous les evenement
556         $req = '(SELECT items.name, COUNT(users_rewards.id_reward) AS c
557             FROM rewards
558             JOIN items ON rewards.id_item = items.id_item
559             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
560             WHERE users_rewards.id_user = :id1
561             GROUP BY items.name)
562             UNION
563             (SELECT items.name, COUNT(users_rewards.id_reward) AS c
564             FROM rewards
565             JOIN items ON rewards.id_item = items.id_item
566             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
567             WHERE users_rewards.id_user = :id2
568             GROUP BY items.name)
569             ORDER BY items.name ASC';
570         $sql = MyPdo::GetMyPdo()->prepare($req);
571         $sql->bindParam(':id1', $id, PDO::PARAM_INT);
572         $sql->bindParam(':id2', $id, PDO::PARAM_INT);
573         $sql->execute();
574
575         return $sql->fetchAll(PDO::FETCH_ASSOC);
576     }
577
578     // compte le nombre d'objets qu'a l'utilisateur pour chaque item
579     static function SelectCountRewardItemByIdUser($id) {
580         // la 1ere partie de la requete recuper les evenements et les nombre
581         // d'objet de l'utilisateur pour cet evenement
582         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
583         // aucun objet
584         // on fait une union des deux pour avoir tous les evenement
585         $req = '(SELECT items.name, COUNT(users_rewards.id_reward) AS c
586             FROM rewards
587             JOIN items ON rewards.id_item = items.id_item
588             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
589             WHERE users_rewards.id_user = :id1
590             GROUP BY items.name)
591             UNION
592             (SELECT items.name, COUNT(users_rewards.id_reward) AS c
593             FROM rewards
594             JOIN items ON rewards.id_item = items.id_item
595             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
596             WHERE users_rewards.id_user = :id2
597             GROUP BY items.name)
598             ORDER BY items.name ASC';
599         $sql = MyPdo::GetMyPdo()->prepare($req);
600         $sql->bindParam(':id1', $id, PDO::PARAM_INT);
601         $sql->bindParam(':id2', $id, PDO::PARAM_INT);
602         $sql->execute();
603
604         return $sql->fetchAll(PDO::FETCH_ASSOC);
605     }
606
607     // compte le nombre d'objets qu'a l'utilisateur pour chaque item
608     static function SelectCountRewardItemByName($name) {
609         // la 1ere partie de la requete recuper les evenements et les nombre
610         // d'objet de l'utilisateur pour cet evenement
611         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
612         // aucun objet
613         // on fait une union des deux pour avoir tous les evenement
614         $req = '(SELECT items.name, COUNT(users_rewards.id_reward) AS c
615             FROM rewards
616             JOIN items ON rewards.id_item = items.id_item
617             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
618             WHERE users_rewards.id_user = :id1
619             GROUP BY items.name)
620             UNION
621             (SELECT items.name, COUNT(users_rewards.id_reward) AS c
622             FROM rewards
623             JOIN items ON rewards.id_item = items.id_item
624             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
625             WHERE users_rewards.id_user = :id2
626             GROUP BY items.name)
627             ORDER BY items.name ASC';
628         $sql = MyPdo::GetMyPdo()->prepare($req);
629         $sql->bindParam(':id1', $name, PDO::PARAM_INT);
630         $sql->bindParam(':id2', $name, PDO::PARAM_INT);
631         $sql->execute();
632
633         return $sql->fetchAll(PDO::FETCH_ASSOC);
634     }
635
636     // compte le nombre d'objets qu'a l'utilisateur pour chaque item
637     static function SelectCountRewardItemByNameAndUser($name, $id) {
638         // la 1ere partie de la requete recuper les evenements et les nombre
639         // d'objet de l'utilisateur pour cet evenement
640         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
641         // aucun objet
642         // on fait une union des deux pour avoir tous les evenement
643         $req = '(SELECT items.name, COUNT(users_rewards.id_reward) AS c
644             FROM rewards
645             JOIN items ON rewards.id_item = items.id_item
646             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
647             WHERE users_rewards.id_user = :id1
648             GROUP BY items.name)
649             UNION
650             (SELECT items.name, COUNT(users_rewards.id_reward) AS c
651             FROM rewards
652             JOIN items ON rewards.id_item = items.id_item
653             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
654             WHERE users_rewards.id_user = :id2
655             GROUP BY items.name)
656             ORDER BY items.name ASC';
657         $sql = MyPdo::GetMyPdo()->prepare($req);
658         $sql->bindParam(':id1', $name, PDO::PARAM_INT);
659         $sql->bindParam(':id2', $id, PDO::PARAM_INT);
660         $sql->execute();
661
662         return $sql->fetchAll(PDO::FETCH_ASSOC);
663     }
664
665     // compte le nombre d'objets qu'a l'utilisateur pour chaque item
666     static function SelectCountRewardItemByNameAndUserAndEvent($name, $id, $event) {
667         // la 1ere partie de la requete recuper les evenements et les nombre
668         // d'objet de l'utilisateur pour cet evenement
669         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
670         // aucun objet
671         // on fait une union des deux pour avoir tous les evenement
672         $req = '(SELECT items.name, COUNT(users_rewards.id_reward) AS c
673             FROM rewards
674             JOIN items ON rewards.id_item = items.id_item
675             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
676             WHERE users_rewards.id_user = :id1
677             AND users_rewards.id_event = :event
678             GROUP BY items.name)
679             UNION
680             (SELECT items.name, COUNT(users_rewards.id_reward) AS c
681             FROM rewards
682             JOIN items ON rewards.id_item = items.id_item
683             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
684             WHERE users_rewards.id_user = :id2
685             AND users_rewards.id_event = :event
686             GROUP BY items.name)
687             ORDER BY items.name ASC';
688         $sql = MyPdo::GetMyPdo()->prepare($req);
689         $sql->bindParam(':id1', $name, PDO::PARAM_INT);
690         $sql->bindParam(':id2', $id, PDO::PARAM_INT);
691         $sql->bindParam(':event', $event, PDO::PARAM_INT);
692         $sql->execute();
693
694         return $sql->fetchAll(PDO::FETCH_ASSOC);
695     }
696
697     // compte le nombre d'objets qu'a l'utilisateur pour chaque item
698     static function SelectCountRewardItemByNameAndUserAndEventAndReward($name, $id, $event, $reward) {
699         // la 1ere partie de la requete recuper les evenements et les nombre
700         // d'objet de l'utilisateur pour cet evenement
701         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
702         // aucun objet
703         // on fait une union des deux pour avoir tous les evenement
704         $req = '(SELECT items.name, COUNT(users_rewards.id_reward) AS c
705             FROM rewards
706             JOIN items ON rewards.id_item = items.id_item
707             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
708             WHERE users_rewards.id_user = :id1
709             AND users_rewards.id_event = :event
710             AND users_rewards.id_reward = :reward
711             GROUP BY items.name)
712             UNION
713             (SELECT items.name, COUNT(users_rewards.id_reward) AS c
714             FROM rewards
715             JOIN items ON rewards.id_item = items.id_item
716             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
717             WHERE users_rewards.id_user = :id2
718             AND users_rewards.id_event = :event
719             AND users_rewards.id_reward = :reward
720             GROUP BY items.name)
721             ORDER BY items.name ASC';
722         $sql = MyPdo::GetMyPdo()->prepare($req);
723         $sql->bindParam(':id1', $name, PDO::PARAM_INT);
724         $sql->bindParam(':id2', $id, PDO::PARAM_INT);
725         $sql->bindParam(':event', $event, PDO::PARAM_INT);
726         $sql->bindParam(':reward', $reward, PDO::PARAM_INT);
727         $sql->execute();
728
729         return $sql->fetchAll(PDO::FETCH_ASSOC);
730     }
731
732     // compte le nombre d'objets qu'a l'utilisateur pour chaque item
733     static function SelectCountRewardItemByNameAndUserAndEventAndRewardAndHero($name, $id, $event, $reward, $hero) {
734         // la 1ere partie de la requete recuper les evenements et les nombre
735         // d'objet de l'utilisateur pour cet evenement
736         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
737         // aucun objet
738         // on fait une union des deux pour avoir tous les evenement
739         $req = '(SELECT items.name, COUNT(users_rewards.id_reward) AS c
740             FROM rewards
741             JOIN items ON rewards.id_item = items.id_item
742             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
743             WHERE users_rewards.id_user = :id1
744             AND users_rewards.id_event = :event
745             AND users_rewards.id_reward = :reward
746             AND users_rewards.id_hero = :hero
747             GROUP BY items.name)
748             UNION
749             (SELECT items.name, COUNT(users_rewards.id_reward) AS c
750             FROM rewards
751             JOIN items ON rewards.id_item = items.id_item
752             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
753             WHERE users_rewards.id_user = :id2
754             AND users_rewards.id_event = :event
755             AND users_rewards.id_reward = :reward
756             AND users_rewards.id_hero = :hero
757             GROUP BY items.name)
758             ORDER BY items.name ASC';
759         $sql = MyPdo::GetMyPdo()->prepare($req);
760         $sql->bindParam(':id1', $name, PDO::PARAM_INT);
761         $sql->bindParam(':id2', $id, PDO::PARAM_INT);
762         $sql->bindParam(':event', $event, PDO::PARAM_INT);
763         $sql->bindParam(':reward', $reward, PDO::PARAM_INT);
764         $sql->bindParam(':hero', $hero, PDO::PARAM_INT);
765         $sql->execute();
766
767         return $sql->fetchAll(PDO::FETCH_ASSOC);
768     }
769
770     // compte le nombre d'objets qu'a l'utilisateur pour chaque item
771     static function SelectCountRewardItemByNameAndUserAndEventAndRewardAndHeroAndReward($name, $id, $event, $reward, $hero, $reward2) {
772         // la 1ere partie de la requete recuper les evenements et les nombre
773         // d'objet de l'utilisateur pour cet evenement
774         // la 2eme partie recuper les evenements pou lesquelle l'utilisateur n'a
775         // aucun objet
776         // on fait une union des deux pour avoir tous les evenement
777         $req = '(SELECT items.name, COUNT(users_rewards.id_reward) AS c
778             FROM rewards
779             JOIN items ON rewards.id_item = items.id_item
780             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
781             WHERE users_rewards.id_user = :id1
782             AND users_rewards.id_event = :event
783             AND users_rewards.id_reward = :reward
784             AND users_rewards.id_hero = :hero
785             AND users_rewards.id_reward2 = :reward2
786             GROUP BY items.name)
787             UNION
788             (SELECT items.name, COUNT(users_rewards.id_reward) AS c
789             FROM rewards
790             JOIN items ON rewards.id_item = items.id_item
791             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
792             WHERE users_rewards.id_user = :id2
793             AND users_rewards.id_event = :event
794             AND users_rewards.id_reward = :reward
795             AND users_rewards.id_hero = :hero
796             AND users_rewards.id_reward2 = :reward2
797             GROUP BY items.name)
798             ORDER BY items.name ASC';
799         $sql = MyPdo::GetMyPdo()->prepare($req);
800         $sql->bindParam(':id1', $name, PDO::PARAM_INT);
801         $sql->bindParam(':id2', $id, PDO::PARAM_INT);
802         $sql->bindParam(':event', $event, PDO::PARAM_INT);
803         $sql->bindParam(':reward', $reward, PDO::PARAM_INT);
804         $sql->bindParam(':hero', $hero, PDO::PARAM_INT);
805         $sql->bindParam(':reward2', $reward2, PDO::PARAM_INT);
806         $sql->execute();
807
808         return $sql->fetchAll(PDO::FETCH_ASSOC);
809     }
810 }
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php lundi 19 juin 2017 11:01

```
526             WHERE users_rewards.id_user = :id1
527             GROUP BY heroes.id_hero)
528         UNION
529             (SELECT heroes.name, 0 AS c
530             FROM heroes
531             WHERE heroes.id_hero NOT IN
532                 (SELECT heroes.id_hero
533                 FROM heroes
534                 JOIN rewards ON rewards.id_hero = heroes.id_hero
535                 JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
536                 WHERE users_rewards.id_user = :id2
537                 GROUP BY heroes.id_hero))
538             ORDER BY name ASC';
539     $sql = MyPdo::GetMyPdo()->prepare($req);
540     $sql->bindParam(':id1', $id, PDO::PARAM_INT);
541     $sql->bindParam(':id2', $id, PDO::PARAM_INT);
542     $sql->execute();
543
544     return $sql->fetchAll(PDO::FETCH_ASSOC);
545 }
546 ?
547 ?>
```

Class Oc_Display

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php  
lundi 19 juin 2017 11:04  
1 <?php  
2 /*  
3     Auteur: Sven Wikberg  
4     Date: 19/06/2017  
5     Description: classe d'affichage des donnees  
6 */  
7 class OcDisplay{  
8  
9     // affiche la barre de navigation, avec des changement selon si on est connecter  
10    ou pas  
11    static function DisplayNavbar(){  
12        $display = '';  
13  
14        $display .= '<ul>';  
15        $display .= '<li><a href="index.php">Index</a></li>';  
16        $display .= '<li><a href="heroes.php">Heroes</a></li>';  
17        $display .= '<li><a href="events.php">Events</a></li>';  
18        $display .= '<li><a href="rewards.php">Others Rewards</a></li>';  
19        if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null){  
20            if(fmmatch('*user.php', $_SERVER['PHP_SELF'])){  
21                $display .= '<li><a href="user.php?action=deco">Log Out</a></li>';  
22            } else {  
23                $display .= '<li><a href="user.php">My Account</a></li>';  
24            }  
25        } else {  
26            $display .= '<li><a href="user.php">Login/Sign In</a></li>';  
27        }  
28  
29        $display .= '</ul>';  
30  
31        echo $display;  
32    }  
33  
34    // affiche une liste des utilisateur non bannis avec des bouton pour bannir  
35    // les utilisateurs  
36    static function DisplayListCleanUserWithBan(){  
37        $clean_users = OcDao::SelectCleanUsersNoAdmin(); // retourne les utilisateur  
38        non bannis (sans les admins)  
39        $display = '';  
40  
41        $display .= '<ul style="height:100%; width:300px; overflow:hidden;  
42        overflow-y:auto;">';  
43        foreach ($clean_users as $clean_user) { // on affiche les utilisateur non  
44        bannis  
45        $display .= '<li>' . $clean_user['username'] . ' / ' . $clean_user[  
46        'email'];  
47        $display .= '&nbsp;<a href="admin.php?action=ban&id=' . $clean_user[  
48        'id_user'] . '"></a>';  
49        $display .= '</li>';  
50    }  
51    $display .= '</ul>';  
52  
53    echo $display;  
54 }  
55  
56 // affiche un liste des utilisateurs bannis avec des boutons pour les dé-bannir
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
ou les supprimer
50     static function DisplayListBannedUserWithUnbanDelete() {
51         $banned_users = OcDao::SelectBannedUsers(); // retourne les utilisateurs
52         bannis (sans les admins)
53         $display = '';
54
55         $display .= '<ul style="height:100%; width:300px; overflow:hidden;
56         overflow-y:auto;">';
57         foreach ($banned_users as $banned_user) { // on affiche les utilisateur bannis
58             $display .= '<li>' . $banned_user['username'] . ' / ' . $banned_user[
59             'email'];
60             $display .= '&nbsp;<a href="admin.php?action=unban&id=' . $banned_user[
61             'id_user'] . '"></a>';
62             $display .= '&nbsp;<a href="admin.php?action=delete&id=' . $banned_user[
63             'id_user'] . '"></a>';
64             $display .= '</li>';
65         }
66         $display .= '</ul>';
67
68         echo $display;
69     }
70
71     // affiche les objets d'un evenement triés, avec les liens pour les selectionner
72     si l'utilisateur est connecté
73     static function DisplayEventRewards($id_event) {
74         $rewards_array = OcDao::SelectRewardsInArrayOfQualityAndTypeByIdEvent(
75             $id_event); // retourne un tableau de rewards
76
77         if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null) // si l'utilisateur est connecter on va chercher l'id des rewards qu'il a
78             $rewards_owned = OcDao::SelectIdRewardsByIdEventAndidUser($id_event,
79             $_SESSION['id_connected']);
80         else
81             $rewards_owned = false;
82
83         if($rewards_array){
84             $display = '';
85
86             foreach ($rewards_array as $key => $type) {
87                 $quality_count = count($type); // on compte le nombre de boite il y
88                 par catégorie afin de definir la taille de cell-ci
89                 $display .= '<div class="rewards_type" style="width:' . (
90                     $quality_count == 4 ? '100' : $quality_count * 24 - 0.5) . '%>';
91                 $display .= '<h2>' . $key. '</h2>';
92                 $display .= '<div>';
93                 foreach ($type as $key => $quality) {
94                     $display .= '<div>';
95                     $display .= '<h3 class="' . $key. '">' . $key. '</h3><ul>';
96                     foreach ($quality as $key => $reward) {
97
98                         // si l'id reward correspond a un element du tableau alors
99                         l'utilisateur a la reward, donc on l'affiche vert
100                         if($rewards_owned && in_array($reward['id_reward'],
101                             $rewards_owned)){
102                             $css_string = 'style="color: #00c600;"';
103                         }
104                     }
105                 }
106             }
107         }
108     }
109 }
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
91 } else{
92     $css_string = '';
93 }
94
95 // si un hero est associer a l'objet on l'affiche sinon, on
96 // affiche juste le nom de l'objet
97 // le lien envoie 3 variables en GET:
98 // id, qui est l'id de l'événement, qui sert à revenir
99 // sur la bonne page
100 // action, qui signifie qu'il faut faire une action en
101 // l'occurrence ajouter un "user_reward"
102 // id_reward, qui est l'id de l'objet sur lequel on a
103 // cliqué
104
105 $display .= '<li>';
106 if(isset($_SESSION['id_connected']) && $_SESSION[
107 'id_connected'] != null) // si un utilisateur est connecté
108 on affiche les liens pour ajouter/enlever un "user_reward"
109     $display .= '<a ' . $css_string. ' href="event.php?id=' .
110     $id_event. '&action=add_user_reward&id_reward=' . $reward[
111     'id_reward']. '">';
112     $display .= $reward['r_name'] . ($reward['h_name'] != NULL ?
113     ' / ' . $reward['h_name'] : '');
114     if(isset($_SESSION['id_connected']) && $_SESSION[
115     'id_connected'] != null)
116         $display .= '</a>';
117     $display .= '</li>';
118 }
119
120 $display .= '</ul>';
121 $display .= '</div>';
122 $display .= '</div>';
123 }
124
125 echo $display;
126 } else {
127     header('Location: events.php');
128 }
129
130 // affiche les informations de base d'un événement
131 static function DisplayEventInfo($id_event){
132     $event = OcDao::SelectEventById($id_event); // retourne l'enregistrement de
133     // l'événement depuis la base
134
135     if($event){
136         $display = '<h1>' . $event['name']. '</h1>';
137         $display .= '<div><p>Beginning Date: ' . $event['start_date']. '</p><p>Ending Date: ' . $event['end_date']. '</p></div>';
138
139         echo $display;
140     } else {
141         header('Location: events.php');
142     }
143 }
144
145 // affiche un tableau des événements avec un paramètre pour le nombre de colonnes
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
du tableau
136     static function DisplayEvents($nb_col_max) { // $nb_col_max = nombre de colonnes
137         maximum pour les tableaux de heros
138         $events = OcDao::SelectEvents();
139         $display = '';
140         $nb_col_current = 0; // nombre de colonnes actuelle
141
142         $display .= '<table>';
143         foreach ($events as $event) {
144             $nb_col_current++;
145
146             if($nb_col_current == 1) // si le nombre de colonne actuelle vaut 1,
147             c'est qu'on est au debut d'une nouvelle ligne donc on ouvre une balise
148             <tr>
149                 $display .= '<tr>';
150                 $display .= '<td><a href="event.php?id=' . $event['id_event']. '"><div>' .
151                 $event['name']. '</div><div>From: ' .
153                 $event['start_date']. ' To: ' . $event['start_date']. '</div></a></td>';
154             if($nb_col_current == $nb_col_max) { // si le nombre de colonne actuelle
155             vaut le nombre de colonne max, c'est qu'on est a la fin de la ligne donc
156             on ferme une balise <tr>
157                 $display .= '</tr>';
158                 $nb_col_current = 0;
159             }
160         }
161     }
162
163     $display .= '</table>';
164
165     echo $display;
166 }
167
168 // affiche les objets d'un hero triés, avec les liens pour les selectionner si
169 // l'utilisateur est connecté
170 static function DisplayHeroRewards($id_hero) {
171     $rewards_array = OcDao::SelectRewardsInArrayOfQualityAndTypeByIdHero($id_hero
172 ); // retourne un tableau de rewards
173
174     if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null) // si l'utilisateur est connecter on va chercher l'id des rewards qu'il a
175     $rewards_owned = OcDao::SelectIdRewardsByIdHeroAndidUser($id_hero,
176     $_SESSION['id_connected']);
177     else
178         $rewards_owned = false;
179
180     if($rewards_array){
181         $display = '';
182
183         foreach ($rewards_array as $key => $type) {
184             $quality_count = count($type); // on compte le nombre de boite il y
185             par catégorie afin de definir la taille de cell-ci
186             $display .= '<div class="rewards_type" style="width:' . (
187             $quality_count == 4 ? '100' : $quality_count * 24 - 0.5) . '%>';
188             $display .= '<h2>' . $key. '</h2>';
189             $display .= '<div>';
190             foreach ($type as $key => $quality) {
191                 $display .= '<div>';
192                 $display .= '<h3 class="" . $key. '">' . $key. '</h3><ul>';
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
c'est qu'on est au debut d'une nouvelle ligne donc on ouvre une
balise <tr>
224     $display .= '<tr>';
225     $display .= '<td width=' . 100 / $nb_col_max . '%>';
226     $display .= '<h2>' . $ability['name'] . '</h2><p>' . $ability[
227     'description'] . '</p>';
228     $display .= ($ability['is_ultimate'] == 0 ? '<h3>Ultimate</h3>' : '');
229     $display .= '</td>';
230
231     if($nb_col_current == $nb_col_max) { // si le nombre de colonne
actuelle vaut le nombre de colonne max, c'est qu'on est a la fin de
la ligne donc on ferme une balise <tr>
232         $display .= '</tr>';
233         $nb_col_current = 0;
234     }
235 }
236 $display .= '</table>';
237
238 echo $display;
239 } else {
240     header('Location: heroes.php');
241 }
242 }

// affiche les informations de base d'un hero
244 static function DisplayHeroInfo($id_hero){
245     $hero = OcDao::SelectHeroById($id_hero); // retourne l'enregistrement du
heros depuis la base
246
247     if($hero){
248         $display = '<h1>' . $hero['name'] . '</h1>';
249         $display .= '<p>' . $hero['description'] . '</p>';
250         $display .= '<div><p>Real Name: ' . $hero['real_name'] . '</p><p>Base Of
Operations: ' . $hero['base_of_operations'] . '</p></div>';
251         $display .= '<div><p>Health: ' . $hero['health'] . '</p><p>Armour: ' . $hero
['armour'] . '</p><p>Shield: ' . $hero['shield'] . '</p></div>';
252         $display .= '<div><p>Affiliation: ' . $hero['affiliation'] .
'</p><p>Difficulty: ' . $hero['difficulty'] . '</p></div>';
253
254         echo $display;
255     } else {
256         header('Location: heroes.php');
257     }
258 }

// affiche un tableau de hero pour chaque role, avec un parametre pour le nombre
de colonne de ces tableaux
261 static function DisplayHeroesByRole($nb_col_max){ // $nb_col_max = nombre de
colonnes maximum pour les tableaux de heros
262     $heroes_array = OcDao::SelectHeroesInArrayOfRole(); // retourne un tableau,
trié par role, de tableau de héros
263     $display = '';
264     $nb_col_current = 0; // nombre de colonnes actuelle
265
266     foreach ($heroes_array as $key => $role) { // pour chaque role on fait un
tableau afin d'ordonner et de regrouper les heros
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
268     $display .= '<div>';
269     $display .= '<h1>' . $key . '</h1>';
270     $display .= '<table>';
271     foreach ($role as $hero) {
272         $nb_col_current++;
273
274         if($nb_col_current == 1) // si le nombre de colonne actuelle vaut 1,
275         c'est qu'on est au début d'une nouvelle ligne donc on ouvre une
276         balise <tr>
277             $display .= '<tr>';
278             $display .= '<td><a href="hero.php?id=' . $hero['id_hero'] . '"><div>' . $hero['name'].'
281             '</div></a></td>';
282             if($nb_col_current == $nb_col_max){ // si le nombre de colonne
283             actuelle vaut le nombre de colonne max, c'est qu'on est à la fin de
284             la ligne donc on ferme une balise <tr>
285                 $display .= '</tr>';
286                 $nb_col_current = 0;
287             }
288
289             $nb_col_current = 0; // on remet à 0 pour le prochain rôle
290             $display .= '</table>';
291             $display .= '</div>';
292         }
293         echo $display;
294     }
295
296     // affiche une liste des utilisateurs et administrateurs non bannis
297     static function DisplayListUsers(){
298         $users = OcDao::SelectCleanUsers(); // récupère le utilisateur depuis la
299         base de données
300
301         $display = '';
302         $display .= '<ul style="height:100%; width:170px; overflow:hidden;
303         overflow-y:auto;">';
304         foreach ($users as $user) {
305             $display .= '<li>' . $user['username'] . ($user['is_admin'] == 1 ? ' /
306             Admin' : '') . '</li>';
307         }
308         $display .= '</ul>';
309
310         echo $display;
311     }
312
313     // affiche les objets non liés à un héros de façon triée, avec les liens pour les
314     sélectionner si l'utilisateur est connecté
315     static function DisplayOtherRewards(){
316         $rewards_array = OcDao::SelectRewardsInArrayOfQualityAndTypeByNoIdHero(); // //
317         retourne un tableau de rewards
318
319         if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null) // //
320         si l'utilisateur est connecté on va chercher l'id des récompenses qu'il a
321         $rewards_owned = OcDao::SelectIdRewardsByNoIdHeroAndIdUser($_SESSION[
322         'id_connected']);
323     else
324         $rewards_owned = false;
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
311
312     $display = '';
313
314     foreach ($rewards_array as $key => $type) {
315         $quality_count = count($type); // on compte le nombre de boite il y par
316         // catégorie afin de definir la taille de celle-ci
317         $display .= '<div class="rewards_type" style="width:' . ($quality_count
318         == 4 ? '100' : $quality_count * 24 - 0.5) . '%>';
319         $display .= '<h2>' . $key. '</h2>';
320         $display .= '<div>';
321         foreach ($type as $key => $quality) {
322             $display .= '<div>';
323             $display .= '<h3 class="' . $key. '">' . $key. '</h3><ul>';
324             // si l'id reward correspond a un element du tableau alors
325             // l'utilisateur a la reward, donc on l'affiche vert
326             if($rewards_owned && in_array($reward['id_reward'],
327             $rewards_owned)){
328                 $css_string = 'style="color: #00c600;"';
329             } else{
330                 $css_string = '';
331             }
332             // le lien envoie 2 variables en GET:
333             // action, qui signifie qu'il faut faire une action en
334             // l'occurrence ajouter un "user_reward"
335             // id_reward, qui est l'id de l'objet sur lequel on a
336             // cliqué
337             $display .= '<li>';
338             if(isset($_SESSION['id_connected']) && $_SESSION['id_connected']
339             != null) // si un utilisateur est connecté on affiche le liens
340             pour ajouter/enlever un "user_reward"
341             $display .= '<a ' . $css_string. '
342             href="rewards.php?action=add_user_reward&id_reward=' . $reward
343             ['id_reward']. '"';
344             $display .= $reward['name'];
345             if(isset($_SESSION['id_connected']) && $_SESSION['id_connected']
346             != null)
347                 $display .= '</a>';
348             $display .= '</li>';
349         }
350         $display .= '</ul>';
351         $display .= '</div>';
352     }
353     echo $display;
354 }
355
356 // affichage du formulaire de connection avec les message d'erreurs, en GET,
357 s'il y en a
358 static function DisplayLogin(){
359     $display = '<section id="login"><form action="user.php?action=login"
360     method="post ">
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
355             <fieldset>
356                 <legend><h2>Login</h2></legend>
357                 <p>Username :</p>
358                 <input maxlength="25" required type="text" name="username"
359                 value="">;
360
361         if(isset($_GET['msg'])) {
362             if($_GET['msg'] == 'wrongUn') // le nom d'utilisateur n'est pas valide
363                 $display .= 'Username not valid';
364             elseif($_GET['msg'] == 'banned') // l'utilisateur est banni
365                 $display .= 'Your account has been banned';
366
367             $display .=    '<br>
368                         <p>Password:</p>
369                         <input required type="password" name="password" value="">';
370
371         if(isset($_GET['msg'])) {
372             if($_GET['msg'] == 'wrongPw') // le mot de passe n'est pas valide
373                 $display .= 'Wrong password';
374
375             $display .=    '<br><br>
376                         <input type="submit" value="Submit">
377                         </fieldset>
378                     </form></section>';
379         echo $display;
380     }
381
382 // affichage du formulaire de création de compte avec les message d'erreurs, en
383 // GET, s'il y en a
384 static function DisplaySignin(){
385     $display = '<section id="sign_in">
386             <form action="user.php?action=signin" method="post">
387                 <fieldset>
388                     <legend><h2>Sign in</h2></legend>
389                     <p>Username* :</p>
390                     <input maxlength="25" required type="text"
391                     name="username" value="">;
392
393         if(isset($_GET['msg'])) {
394             if($_GET['msg'] == 'duplicate') // nom d'utilisateur ou email déjà utilisé
395                 $display .= 'Username or email already used';
396
397             $display .=
398                     '<p>Email* :</p>
399                     <input maxlength="100" required type="email"
400                     name="email" value=""><br>
401                     <p>Password* :</p>
402                     <input minlength="8" required type="password"
403                     name="password" value=""><br><br>
404                     <input type="submit" value="Submit">
405                     </fieldset>
406                 </form>
407             </section>';
408         echo $display;
409     }
410
411 // affiche les informations du compte de l'utilisateur connecté
412 static function DisplayAccountInfo($id_user) {
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
407     $user = OcDao::SelectUserById($id_user);
408
409     $display = '';
410     $display .= '<section id="account_info">';
411     $display .= '<div id="title">';
412     $display .= '<h1>My Account </h1>; // '&nbsp' sert à forcer un
413     espace en html
414     $display .= '<a href="user.php?goto=updating"></a>';
416     $display .= '</div>';
417
418     if(isset($_GET['msg'])) {
419         if($_GET['msg'] == 'updateDuplicate')
420             $display .= '<h3>Username/Email already used</h3>';
421
422         $display .= '<div class="flex_row">';
423         $display .= '<div style="width:70%">';
424         $display .= '<p>Username: ' . $user['username'] . '</p>';
425         $display .= '<p>Email: ' . $user['email'] . '</p>';
426         $display .= '</div>';
427         if($user['is_admin']) {
428             $display .= '<div>';
429             $display .= '<a href="admin.php"><p>Admin page</p></a>';
430             $display .= '</div>';
431
432         $display .= '</div>';
433         $display .= '</section>';
434
435         echo $display;
436     }
437
438 // affiche le formulaire qui permet de modifier les infos de l'utilisateur
439 static function DisplayAccountInfoUpdating($id_user) {
440     $user = OcDao::SelectUserById($id_user);
441
442     $display = '';
443     $display .= '<section id="account_info_updating">';
444     $display .= '<div id="title">';
445     $display .= '<h1>Updating account info</h1>';
446     $display .= '</div>';
447
448     $display .= '<form action="user.php?action=update" method="post">';
449     $display .= '<p>Username :</p>';
450     $display .= '<input maxlength="25" required type="text" name="username"
451     value="' . $user['username'] . '"><br>';
452     $display .= '<p>Email: </p>';
453     $display .= '<input maxlength="100" required type="email" name="email"
454     value="' . $user['email'] . '"><br><br>';
455     $display .= '<input type="submit" value="Submit">';
456     $display .= '</form>';
457
458     echo $display;
459 }
460
461 // affiche la section des statistiques d'un utilisateur
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
460     static function DisplayAccountStats($id_user) {
461         echo '<section id="account_stats">';
462         OcDisplay::DisplayMainProgressBar($id_user);
463         OcDisplay::DisplayEventsProgressBar($id_user);
464         OcDisplay::DisplayHeroesProgressBar($id_user);
465         echo '</section>';
466     }
467
468     // affiche la barre de progression principale (des tous les objets) d'un
469     // utilisateur
470     static function DisplayMainProgressBar($id_user) {
471         $all_count = OcDao::SelectCountReward(); // retourne le nombre d'objet en
472         // tout
473         $user_count = OcDao::SelectCountRewardByIdUser($id_user); // retourne le
474         // nombre d'objet d'un utilisateur
475         $display = '';
476
477         $display .= '<div>';
478         $display .= '<div class="flex_row">';
479         $display .= '<h2>All rewards</h2>';
480         $display .= '<h2>' . $user_count . '/' . $all_count . '</h2>';
481         $display .= '</div>';
482         $display .= OcDisplay::GetProgressBar($all_count, $user_count, 100, 30);
483         $display .= '</div>';
484         echo $display;
485     }
486
487     // affiche les barres de progression des evenements d'un utilisateur
488     static function DisplayEventsProgressBar($id_user) {
489         $array_events_count = OcDao::SelectCountRewardEvents(); // retourne le
490         // nombre d'objets de chaque evenement
491         $array_events_user_count = OcDao::SelectCountRewardEventsByIdUser($id_user);
492         // retourne le nombre d'objets de l'utilisateur pour chaque evenement
493         $display = '';
494
495         $display .= '<div>';
496         $display .= '<h2>Events</h2>';
497         $display .= '<div class="flex_row">';
498         for ($i=0; $i < count($array_events_count); $i++) {
499             $display .= '<div style="width:48%;>';
500             $display .= '<div class="flex_row">';
501             $display .= '<h4>' . $array_events_count[$i]['name'] . '</h4>';
502             $display .= '<h4>' . $array_events_user_count[$i]['c'] . '/' .
503             $array_events_count[$i]['c'] . '</h4>';
504             $display .= '</div>';
505             $display .= OcDisplay::GetProgressBar($array_events_count[$i]['c'],
506             $array_events_user_count[$i]['c'], 100, 25);
507             $display .= '</div>';
508         }
509         $display .= '</div>';
510         $display .= '</div>';
511         echo $display;
512     }
513
514     // affiche les barres de progression des heros d'un utilisateur
515     static function DisplayHeroesProgressBar($id_user) {
516         $array_heroes_count = OcDao::SelectCountRewardHeroes(); // retourne le
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```
nombre d'objets de chaque hero
510 $array_heroes_user_count = OcDao::SelectCountRewardHeroesByIdUser($id_user);
// retourne le nombre d'objets de l'utilisateur pour chaque hero
511 $display = '';
512
513 $display .= '<div>';
514 $display .= '<h2>Heroes</h2>';
515 $display .= '<div class="flex_row">';
516 for ($i=0; $i < count($array_heroes_count); $i++) {
517     $display .= '<div style="width:32%;">';
518     $display .= '<div class="flex_row">';
519     $display .= '<h4>' . $array_heroes_count[$i]['name'] . '</h4>';
520     $display .= '<h4>' . $array_heroes_user_count[$i]['c'] . '/' .
521     $array_heroes_count[$i]['c'] . '</h4>';
522     $display .= OcDisplay::GetProgressBar($array_heroes_count[$i]['c'],
523     $array_heroes_user_count[$i]['c'], 100, 15);
524     $display .= '</div>';
525 }
526 $display .= '</div>';
527 echo $display;
528 }
529
530 // retourne une chaine pour afficher une barre de progression grace aux
parametre de la fonction
531 // max_value est la valeur maximum de la progress bar
532 // current_value est la valur actuelle de la barre
533 // width_percent est la largeur de la barre en pourcentage
534 // height_px est la hauteur de la barre en pixel
535 static function GetProgressBar($max_value, $current_value, $width_percent,
536 $height_px) {
537     $display = '';
538     $display .= '<div class="bar_ext" style="width: ' . $width_percent . '%;
height: ' . $height_px . 'px; border: 1px solid black;">';
539     $display .=      '<div style="width: ' . ($current_value / $max_value) * 100
. '%; background-color: black; height: ' . $height_px . 'px;">';
540     $display .=      '</div>';
541     $display .= '</div>';
542
543     return $display;
544 }
545 ?>
```

Fichier de config base de données

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\config\config_db.php  
lundi 19 juin 2017 11:07  
1 <?php  
2 /*  
3     Auteur: Sven Wikberg  
4     Date: 19/06/2017  
5     Description: D'affichage des donnees  
6 */  
7  
8 define('DB_HOST', 'localhost');  
9 define('DB_NAME', 'overwatchcollection');  
10 define('DB_USER', 'oc_admin');  
11 define('DB_PWD', 'overwatch123');  
12 ?>
```

Style du site web (CSS)

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css                               lundi 19 juin 2017 11:09
1   /*
2    * Auteur: Sven Wikberg
3    * Date: 19/06/2017
4    * Description: Page de style du site
5   */
6   @font-face {
7     font-family: "BigNoodle";
8     src: url('../font/BigNoodleToo.ttf');
9   }
10  @font-face {
11    font-family: "BigNoodle";
12    font-style: italic;
13    src: url('../font/BigNoodleTooOblique.ttf');
14  }
15  @font-face {
16    font-family: "Futura";
17    src: url('../font/Futura.ttf');
18  }
19
20 body{
21   font-family: "BigNoodle", Helvetica, Arial, sans-serif;
22   max-width: 900px;
23   margin-left: auto;
24   margin-right: auto;
25 }
26
27 h1{
28   color: #FF9C1E;
29   font-size: 40px;
30   font-style: italic;
31 }
32
33 h2{
34   font-size: 30px;
35   font-style: italic;
36 }
37
38 h3{
39   font-size: 25px;
40   font-style: italic;
41 }
42
43 h4{
44   font-size: 25px;
45 }
46
47 p{
48   font-family: "Futura", Helvetica, Arial, sans-serif;
49   text-align: justify;
50   font-style: normal;
51   font-weight: normal;
52 }
53
54 footer{
55   padding: 5px;
56   margin: 0px;
57   margin-top: 30px;
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css

lundi 19 juin 2017 11:09

```
58     background-color: #cccccc;
59     border-radius: 15px;
60 }
61 footer p{
62     margin: 0px;
63 }
64
65 header ul{
66     padding: 30px 20px;
67     margin: 0px;
68     list-style-type: none;
69     display: flex;
70     flex-direction: row;
71     justify-content: space-between;
72     flex-wrap: wrap;
73     background-color: #cccccc;
74     border-radius: 15px;
75 }
76
77 header ul li{
78     font-size: 30px;
79 }
80
81 header ul li a{
82     text-decoration: none;
83     color: #606060;
84     padding: 5px;
85 }
86
87 header ul li a:hover{
88     color: #FF9C1E;
89     cursor: pointer;
90     background-color: #a0a0a0;
91     border-radius: 5px;
92 }
93
94 .flex_row{
95     display: flex;
96     flex-direction: row;
97     justify-content: space-between;
98     flex-wrap: wrap;
99 }
100
101 section#index li{
102     font-size: 22px;
103 }
104
105 section#heroes{
106     display: flex;
107     flex-direction: row;
108     justify-content: space-between;
109     flex-wrap: wrap;
110 }
111
112 section#heroes table a div{
113     font-size: 22px;
114 }
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css

lundi 19 juin 2017 11:09

```
115 section#heroes table a img{  
116     border-radius: 5px;  
117 }  
118  
119 section#heroes table td{  
120     border-radius: 5px;  
121 }  
122  
123 section#heroes table td:hover{  
124     background-color: #FF9C1E;  
125 }  
126  
127  
128 section#hero_info div{  
129     display: flex;  
130     flex-direction: row;  
131     justify-content: space-between;  
132     flex-wrap: wrap;  
133 }  
134  
135 section#hero_abilities{  
136     margin-top: 50px;  
137 }  
138  
139 section#hero_abilities table tr td{  
140     border: 3px solid black;  
141     border-radius: 10px;  
142 }  
143  
144 section#rewards{  
145     margin-top: 50px;  
146     display: flex;  
147     flex-direction: row;  
148     justify-content: space-between;  
149     flex-wrap: wrap;  
150 }  
151  
152 section#rewards h2{  
153     margin: 0;  
154 }  
155  
156 section#rewards h3{  
157     margin: 10px 0px;  
158 }  
159  
160 section#rewards ul{  
161     height:200px;  
162     width:170px;  
163     overflow:hidden;  
164     overflow-y:auto;  
165 }  
166  
167 section#rewards div{  
168     display: flex;  
169     flex-direction: column;  
170 }  
171
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css

lundi 19 juin 2017 11:09

```
172 section#rewards div.rewards_type{  
173     border: 3px solid black;  
174     border-radius: 10px;  
175     margin-top: 10px;  
176 }  
177  
178 section#rewards div div{  
179     display: flex;  
180     flex-direction: row;  
181     justify-content: space-between;  
182     flex-wrap: wrap;  
183 }  
184  
185 section#rewards div div div{  
186     display: flex;  
187     flex-direction: column;  
188 }  
189  
190 section#rewards div div div h3.Common{  
191     color: black;  
192 }  
193  
194 section#rewards div div div h3.Rare{  
195     color: #01C2FD;  
196 }  
197  
198 section#rewards div div div h3.Epic{  
199     color: #FE01FE;  
200 }  
201  
202 section#rewards div div div h3.Legendary{  
203     color: #FF9C1E;  
204 }  
205  
206 section#rewards div div div li{  
207     font-size: 22px;  
208     border-radius: 3px;  
209 }  
210  
211 section#events{  
212     margin-top: 30px;  
213 }  
214  
215 section#events table a div{  
216     font-size: 25px;  
217     font-style: italic;  
218 }  
219  
220 section#events table tr td{  
221     padding: 10px;  
222     border-radius: 10px;  
223 }  
224  
225 section#events table tr td:hover{  
226     background-color: #FF9C1E;  
227 }  
228
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css                                         lundi 19 juin 2017 11:09
229  section#events table tr td img{
230      border-radius: 10px;
231  }
232
233  section#event_info div{
234      display: flex;
235      flex-direction: row;
236      justify-content: space-between;
237      flex-wrap: wrap;
238  }
239
240  section#account_info div#title{
241      display: flex;
242      flex-direction: row;
243      justify-content: flex-start;
244      align-items: center;
245      flex-wrap: wrap;
246  }
247
248  section#account_stats div.bar_ext{
249      border-radius: 5px;
250  }
251
252  section#admin table ul{
253      margin: 10px;
254  }
255
256  section#admin table ul li{
257      font-family: "Futura", Helvetica, Arial, sans-serif;
258      font-size: 18px;
259  }
```