

```
1  <?php
2  /*
3  Fichier: sessions.php
4  Auteur: Kevin Zaffino
5  Date: 15/06/2016
6  Version:1.10
7  Description: Gere les sessions
8  Copyright (TPI 2016 - Kevin Zaffino © 2016)
9  */
10
11 session_start();
12 ?>
13
14 <?php
15 require_once './phpScript/constants.php';
16
17 /**
18  * Est-ce que l'utilisateur est loggé
19  * @return {boolean} True is loggé, autrement false
20  */
21 function isLoggedIn() {
22     return (isset($_SESSION["logged"])) ? $_SESSION["logged"] : false;
23 }
24
25 /**
26  * Définis l'etat "logged"
27  */
28 function setLogged() {
29     $_SESSION['logged'] = true;
30 }
31
32 /**
33  * Recupere le niveau de privilège de l'utilisateur
34  * @return (int)
35  */
36 function getPrivilege() {
37     return (isset($_SESSION["privilege"])) ? $_SESSION["privilege"] : PRIV_UNKNOWN;
38 }
39
40 /**
41  * Assigne le niveau de privilege en session
42  * @param (int) $value
43  */
44 function setPrivilege($value) {
45     $_SESSION["privilege"] = $value;
46 }
47
48 /**
49  * Assigne le chemin de l'avatar de l'utilisateur en session
50  * @param (string) $path
51  */
52 function setImagePath($path) {
53     $_SESSION['image'] = $path;
54 }
55
56 /**
57  * Retourne le chemin de l'avatar de l'utilisateur contenu en session
```

```
58     * @return (string)
59     */
60 function getImagePath() {
61     return (isset($_SESSION["image"])) ? $_SESSION["image"] : NULL;
62 }
63
64 /**
65  * Est-ce que l'utilisateur est un admin
66  * @return (bool) TRUE si il est admin, sinon FALSE
67  */
68 function isAdmin() {
69     return getPrivilege() == PRIV_ADMIN;
70 }
71
72 /**
73  * Est-ce que l'utilisateur est un simple utilisateur
74  * @return (bool) TRUE si oui, sinon FALSE
75  */
76 function isUser() {
77     return getPrivilege() == PRIV_USER;
78 }
79
80 /**
81  * Recupere l'id utilisateur contenu en session
82  * @return (int)
83  */
84 function getUserID() {
85     return (isset($_SESSION["uid"])) ? $_SESSION["uid"] : -1;
86 }
87
88 /**
89  * Assigne l'id de l'utilisateur en session
90  * @param type $uid
91  */
92 function setUserID($uid) {
93     $_SESSION["uid"] = $uid;
94 }
95
96 /**
97  * Retourne le nom de l'utilisateur enregistré en session
98  * @return type
99  */
100 function getUsername() {
101     return (isset($_SESSION["uname"])) ? $_SESSION["uname"] : "";
102 }
103
104 /**
105  * Assigne le nom d'utilisateur a une variable de session
106  * @param type $name
107  */
108 function setUsername($name) {
109     $_SESSION["uname"] = $name;
110 }
111
112 /**
113  * Retourne le mail de l'utilisateur
114  * @return type
```

```
115     */
116 function getUserMail() {
117     return (isset($_SESSION["umail"])) ? $_SESSION["umail"] : "";
118 }
119
120 /**
121  * Assigne le mails de l'utilisateur dans une variable de session
122  * @param type $mail
123  */
124 function setUserMail($mail) {
125     $_SESSION['umail'] = $mail;
126 }
127
128 /**
129  * Recupere le numero de l'utilisateur
130  * @return type
131  */
132 function getUserTel() {
133     return (isset($_SESSION["utel"])) ? $_SESSION["utel"] : "";
134 }
135
136 /**
137  * Assigne le numero de l'utilisateur dans une variable de session
138  * @param type $tel
139  */
140 function setUserTel($tel) {
141     $_SESSION['utel'] = $tel;
142 }
143
144 /**
145  * Recupere l'adresse de l'utilisateur
146  * @return type
147  */
148 function getUserAdress() {
149     return (isset($_SESSION["uadress"])) ? $_SESSION["uadress"] : "";
150 }
151
152 /**
153  * Assigne l'adresse de l'utilisateur dans une variable de session
154  * @param type $mail
155  */
156 function setUserAdress($mail) {
157     $_SESSION['uadress'] = $mail;
158 }
159
160 /**
161  * Detruit la session
162  */
163 function destroySession(){
164     session_destroy();
165 }
166
```