

Caiman web

Generated by Doxygen 1.9.1

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Data Structure Index	3
2.1 Data Structures	3
3 File Index	5
3.1 File List	5
4 Data Structure Documentation	7
4.1 Administrator Class Reference	7
4.1.1 Detailed Description	7
4.1.2 Constructor & Destructor Documentation	7
4.1.2.1 __construct()	8
4.1.3 Member Function Documentation	8
4.1.3.1 addGame()	8
4.1.3.2 getConsoleFolderName()	9
4.1.3.3 getListConsole()	10
4.1.3.4 updateGame()	10
4.1.3.5 uploadGame()	11
4.1.3.6 uploadGameImage()	12
4.2 AdministratorController Class Reference	13
4.2.1 Detailed Description	14
4.2.2 Constructor & Destructor Documentation	14
4.2.2.1 __construct()	14
4.2.3 Member Function Documentation	15
4.2.3.1 formHandler()	15
4.2.3.2 printHTML()	17
4.3 Categories Class Reference	18
4.3.1 Detailed Description	18
4.3.2 Constructor & Destructor Documentation	18
4.3.2.1 __construct()	18
4.3.3 Member Function Documentation	19
4.3.3.1 addCategorie()	19
4.3.3.2 addCategorieToGame()	19
4.3.3.3 delCategorieFromGame()	20
4.3.3.4 getCategoriesOfGame()	20
4.3.3.5 getListAllCategories()	21
4.4 DashboardController Class Reference	21
4.4.1 Detailed Description	22
4.4.2 Constructor & Destructor Documentation	22
4.4.2.1 __construct()	22
4.4.3 Member Function Documentation	22

4.4.3.1 formHandler()	23
4.4.3.2 htmlFormUpdatePassword()	23
4.4.3.3 printHTML()	24
4.5 Download Class Reference	25
4.5.1 Detailed Description	25
4.5.2 Member Function Documentation	25
4.5.2.1 downloadCaiman()	26
4.6 DownloadController Class Reference	26
4.6.1 Detailed Description	27
4.6.2 Constructor & Destructor Documentation	27
4.6.2.1 __construct()	27
4.6.3 Member Function Documentation	27
4.6.3.1 formHandler()	28
4.6.3.2 printHTML()	28
4.7 Games Class Reference	29
4.7.1 Detailed Description	29
4.7.2 Constructor & Destructor Documentation	29
4.7.2.1 __construct()	29
4.7.3 Member Function Documentation	30
4.7.3.1 addGameToFavoris()	30
4.7.3.2 checkIfGamelsAlreadyInFavoris()	31
4.7.3.3 getAllGames()	31
4.7.3.4 getFavoriteGamesOfUser()	32
4.7.3.5 getGameDetail()	32
4.7.3.6 getGamesInCategorie()	33
4.7.3.7 getListOfGameWithTimeUser()	33
4.7.3.8 getRequestGames()	34
4.7.3.9 getTimeInGameUser()	34
4.7.3.10 removeGameFromFavoris()	35
4.8 GamesController Class Reference	35
4.8.1 Detailed Description	36
4.8.2 Constructor & Destructor Documentation	36
4.8.2.1 __construct()	36
4.8.3 Member Function Documentation	37
4.8.3.1 formHandler()	37
4.8.3.2 getGameDetail()	38
4.8.3.3 getGamesFromCategorie()	39
4.8.3.4 getListAllGames()	40
4.8.3.5 getRequestedGames()	40
4.8.3.6 printHTML()	41
4.8.3.7 recherchFull()	42
4.8.3.8 recherchNotFull()	43

4.9 iController Interface Reference	44
4.9.1 Detailed Description	45
4.10 IndexController Class Reference	45
4.10.1 Detailed Description	46
4.10.2 Member Function Documentation	46
4.10.2.1 formHandler()	46
4.10.2.2 printHTML()	46
4.11 Login Class Reference	47
4.11.1 Detailed Description	47
4.11.2 Member Function Documentation	47
4.11.2.1 checkLogin()	47
4.12 LoginController Class Reference	48
4.12.1 Detailed Description	48
4.12.2 Constructor & Destructor Documentation	49
4.12.2.1 __construct()	49
4.12.3 Member Function Documentation	49
4.12.3.1 formHandler()	49
4.12.3.2 printHTML()	50
4.13 MainController Class Reference	50
4.13.1 Detailed Description	50
4.13.2 Constructor & Destructor Documentation	50
4.13.2.1 __construct()	50
4.13.3 Member Function Documentation	51
4.13.3.1 allowAccessTo()	51
4.13.3.2 errorHandler()	51
4.14 Signin Class Reference	52
4.14.1 Detailed Description	52
4.14.2 Constructor & Destructor Documentation	52
4.14.2.1 __construct()	52
4.14.3 Member Function Documentation	53
4.14.3.1 checkIfEmailAlreadyTaken()	53
4.14.3.2 checkIfUsernameAlreadyTaken()	53
4.14.3.3 newUser()	54
4.15 SigninController Class Reference	55
4.15.1 Detailed Description	55
4.15.2 Constructor & Destructor Documentation	56
4.15.2.1 __construct()	56
4.15.3 Member Function Documentation	56
4.15.3.1 formHandler()	56
4.15.3.2 printHTML()	57
4.16 User Class Reference	57
4.16.1 Detailed Description	57

4.16.2 Constructor & Destructor Documentation	58
4.16.2.1 __construct()	58
4.16.3 Member Function Documentation	58
4.16.3.1 getPrivateAccount()	58
4.16.3.2 updatePassword()	59
4.16.3.3 updatePrivateAccount()	60
4.17 UserData Class Reference	61
4.17.1 Detailed Description	61
4.17.2 Constructor & Destructor Documentation	61
4.17.2.1 __construct()	61
4.17.3 Member Function Documentation	61
4.17.3.1 getUserData()	61
4.17.3.2 getUsersByUsername()	62
4.18 UsersController Class Reference	63
4.18.1 Detailed Description	63
4.18.2 Constructor & Destructor Documentation	64
4.18.2.1 __construct()	64
4.18.3 Member Function Documentation	64
4.18.3.1 formHandler()	64
4.18.3.2 htmlDetailUser()	65
4.18.3.3 htmlrecherchUsers()	65
4.18.3.4 htmlrequestUser()	66
4.18.3.5 printHTML()	67
5 File Documentation	69
5.1 common/head.php File Reference	69
5.1.1 Detailed Description	69
5.1.1.1 BDCC	69
5.2 common/navbar.php File Reference	69
5.2.1 Detailed Description	69
5.2.1.1 BDCC	70
5.3 controllers/administratorController.php File Reference	70
5.3.1 Detailed Description	70
5.3.1.1 BDCC	70
5.4 controllers/controllers.php File Reference	70
5.4.1 Detailed Description	70
5.4.1.1 BDCC	71
5.5 controllers/dashboardController.php File Reference	71
5.5.1 Detailed Description	71
5.5.1.1 BDCC	71
5.6 controllers/downloadController.php File Reference	71
5.6.1 Detailed Description	71

5.6.1.1 BDCC	72
5.7 controllers/gamesController.php File Reference	72
5.7.1 Detailed Description	72
5.7.1.1 BDCC	72
5.8 controllers/indexController.php File Reference	72
5.8.1 Detailed Description	72
5.8.1.1 BDCC	73
5.9 controllers/interfaceController.php File Reference	73
5.9.1 Detailed Description	73
5.9.1.1 BDCC	73
5.10 controllers/loginController.php File Reference	73
5.10.1 Detailed Description	73
5.10.1.1 BDCC	74
5.11 controllers/mainController.php File Reference	74
5.11.1 Detailed Description	74
5.11.1.1 BDCC	74
5.12 controllers/signinController.php File Reference	74
5.12.1 Detailed Description	74
5.12.1.1 BDCC	75
5.13 controllers/usersController.php File Reference	75
5.13.1 Detailed Description	75
5.13.1.1 BDCC	75
5.14 index.php File Reference	75
5.14.1 Detailed Description	75
5.14.1.1 BDCC	76
5.15 models/administrator.php File Reference	76
5.15.1 Detailed Description	76
5.15.1.1 BDCC	76
5.16 models/categorie.php File Reference	76
5.16.1 Detailed Description	76
5.16.1.1 BDCC	77
5.17 models/class.php File Reference	77
5.17.1 Detailed Description	77
5.17.1.1 BDCC	77
5.18 models/download.php File Reference	77
5.18.1 Detailed Description	77
5.18.1.1 BDCC	78
5.19 models/games.php File Reference	78
5.19.1 Detailed Description	78
5.19.1.1 BDCC	78
5.20 models/login.php File Reference	78
5.20.1 Detailed Description	78

5.20.1.1 BDCC	79
5.21 models/signin.php File Reference	79
5.21.1 Detailed Description	79
5.21.1.1 BDCC	79
5.22 models/user.php File Reference	79
5.22.1 Detailed Description	79
5.22.1.1 BDCC	80
5.23 models/userdata.php File Reference	80
5.23.1 Detailed Description	80
5.23.1.1 BDCC	80
Index	81

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Administrator	7
Categories	18
Download	25
Games	29
iController	44
AdministratorController	13
DashboardController	21
DownloadController	26
GamesController	35
IndexController	45
LoginController	48
SigninController	55
UsersController	63
Login	47
MainController	50
mainController	
AdministratorController	13
DashboardController	21
DownloadController	26
GamesController	35
IndexController	45
LoginController	48
SigninController	55
Signin	52
User	57
UserData	61

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Administrator	7
AdministratorController	13
Categories	18
DashboardController	21
Download	25
DownloadController	26
Games	29
GamesController	35
iController	44
IndexController	45
Login	47
LoginController	48
MainController	50
Signin	52
SigninController	55
User	57
UserData	61
UsersController	63

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

index.php		
Index of the website	75	
test.php	??	
common/ footer.php	??	
common/ head.php		
Head of file	69	
common/ modal.php	??	
common/ navbar.php		
Navbar html	69	
controllers/ administratorController.php		
Class used to handle request for the administrator page	70	
controllers/ controllers.php		
File used to include all the controller of the project	70	
controllers/ dashboardController.php		
Class used to handle request for the dashboard of the user	71	
controllers/ downloadController.php		
Class used to handle the page of download	71	
controllers/ gamesController.php		
Class used to handle request for the games pages	72	
controllers/ indexController.php		
Class used to handle request for the index	72	
controllers/ interfaceController.php		
Interface used to implement function to display the html and the handle the requested content	73	
controllers/ loginController.php		
Class used to handle request for the login page	73	
controllers/ mainController.php		
Main class of the controller used to implement basic function	74	
controllers/ signinController.php		
Class used to handle request to create an account	74	
controllers/ usersController.php		
Class used to handle request of the user of the website	75	
css/ style.css	??	
models/ administrator.php		
Class used to handle request for the administrator	76	
models/ categorie.php		
Class used to handle request for the table categorie	76	

models/class.php	
Class used to handle include all models	77
models/download.php	
Class used to handle the download of Caiman	77
models/games.php	
Class servant a gerer les requetes en lien avec la table game	78
models/login.php	
Class used to connect an user	78
models/signin.php	
Class used to create a new user	79
models/user.php	
Class use to manage user	79
models/userdata.php	
Class use to manage user data	80

Chapter 4

Data Structure Documentation

4.1 Administrator Class Reference

Public Member Functions

- [__construct](#) ()
- [addGame](#) (string \$name, string \$description, string \$imageName, int \$consoleId, \$gameFileName)
- [uploadGame](#) (\$gameFileName, \$consoleId)
- [updateGame](#) (\$idGame, \$name, \$description, \$consoleId)
- [uploadGameImage](#) (\$imageFileName)
- [getConsoleFolderName](#) (\$id)
- [getListConsole](#) ()

Data Fields

- **\$search_username** = null
- **\$search_password** = null
- **\$arrayInfo** = null
- **\$psUploadGame** = null
- **\$psUploadFile** = null
- **\$psUpdateGame** = null

4.1.1 Detailed Description

Definition at line 10 of file administrator.php.

4.1.2 Constructor & Destructor Documentation

4.1.2.1 __construct()

`__construct ()`

default contructor

Definition at line 33 of file administrator.php.

```

34     {
35         if ($this->dbh == null) {
36             try {
37                 $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
38                     PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
39                     PDO::ATTR_PERSISTENT => true
40                 ));
41                 // get list of console
42                 $sqlGetListConsole = "SELECT * FROM consol";
43                 $this->psGetListConsole = $this->dbh->prepare($sqlGetListConsole);
44                 $this->psGetListConsole->setFetchMode(PDO::FETCH_ASSOC);
45
46                 // upload game
47                 $sqlUploadGame = "INSERT INTO game (name, description, imageName, idConsole, idFile)
48                     VALUES (:insert_name, :insert_description, :insert_imageName, :insert_idConsole,
49                     :insert_idFile)";
50                 $this->psUploadGame = $this->dbh->prepare($sqlUploadGame);
51                 $this->psUploadGame->setFetchMode(PDO::FETCH_ASSOC);
52
53                 // upload file
54                 $sqlUploadFile = "INSERT INTO file (filename, dateUpdate)
55                     VALUES (:insert_filename, NOW() )";
56                 $this->psUploadFile = $this->dbh->prepare($sqlUploadFile);
57                 $this->psUploadFile->setFetchMode(PDO::FETCH_ASSOC);
58
59                 // update game
60                 $sqlUpdateGame = "UPDATE game SET name = :update_name, description =
61                     :update_description, idConsole = :update_idConsole WHERE id = :update_id";
62                 $this->psUpdateGame = $this->dbh->prepare($sqlUpdateGame);
63                 $this->psUpdateGame->setFetchMode(PDO::FETCH_ASSOC);
64
65                 // get folder name of console
66                 $sqlGetNameConsoleFolder = "SELECT folderName FROM consol WHERE id = :console_id";
67                 $this->psGetNameConsoleFolder = $this->dbh->prepare($sqlGetNameConsoleFolder);
68                 $this->psGetNameConsoleFolder->setFetchMode(PDO::FETCH_ASSOC);
69             } catch (PDOException $e) {
70                 print "Erreur !: " . $e->getMessage() . "<br>";
71                 die();
72             }
73         }
74     }

```

4.1.3 Member Function Documentation

4.1.3.1 addGame()

```

addGame (
    string $name,
    string $description,
    string $imageName,
    int $consoleId,
    $gameFileName )

```

add a game to the database

Parameters

string	<i>\$name</i>	
string	<i>\$description</i>	
string	<i>\$imageName</i>	
integer	<i>\$consoleId</i>	
string	<i>\$gameFileName</i>	

Returns

void

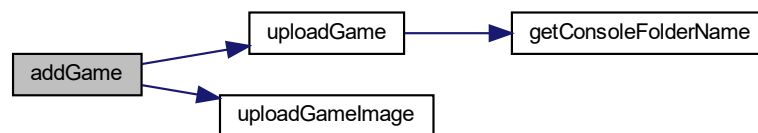
Definition at line 84 of file administrator.php.

```

85     {
86
87         if ($this->uploadGame($gameFileName, $consoleId) && $this->uploadGameImage($imageName)) {
88             try {
89                 $this->psUploadFile->execute(array(':insert_filename' => $gameFileName));
90             } catch (PDOException $e) {
91                 print "Erreur !: " . $e->getMessage() . "<br>";
92                 die();
93             }
94             $lastInsertId = $this->dbh->lastInsertId();
95             try {
96                 $this->psUploadGame->execute(array(':insert_name' => $name, ':insert_description' =>
97                 $description, ':insert_imageName' => $imageName, ':insert_idConsole' => $consoleId, ':insert_idFile'
98                 => $lastInsertId));
99             } catch (PDOException $e) {
100                 print "Erreur !: " . $e->getMessage() . "<br>";
101                 die();
102             }
103         }
104     }

```

Here is the call graph for this function:

**4.1.3.2 getConsoleFolderName()**

```

getConsoleFolderName (
    $id )

```

get the path name of an console

Parameters

int	<i>\$id</i>	
-----	-------------	--

Returns

array of game

Definition at line 202 of file administrator.php.

```

203     {
204         $returnArray = null;
205         try {
206             $this->psGetNameConsoleFolder->execute(array(':console_id' => $id));

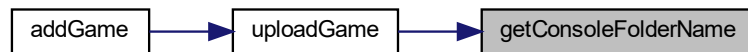
```

```

207         $returnArray = $this->psGetNameConsoleFolder->fetchAll();
208     } catch (PDOException $e) {
209         print "Erreur !: " . $e->getMessage() . "<br>";
210         die();
211     }
212     return $returnArray[0]['folderName'];
213 }

```

Here is the caller graph for this function:



4.1.3.3 getListConsole()

```
getListConsole ( )
```

returns list of all consoles

Returns

array of game

Definition at line 220 of file administrator.php.

```

221 {
222     $returnArray = null;
223     try {
224         $this->psGetListConsole->execute();
225         $returnArray = $this->psGetListConsole->fetchAll();
226     } catch (PDOException $e) {
227         print "Erreur !: " . $e->getMessage() . "<br>";
228         die();
229     }
230     return $returnArray;
231 }

```

4.1.3.4 updateGame()

```

updateGame (
    $idGame,
    $name,
    $description,
    $consoleId )

```

update da of a game

Parameters

int	<i>\$idGame</i>	
string	<i>\$name</i>	
string	<i>\$description</i>	
int	<i>\$consoleId</i>	

Returns

void

Definition at line 151 of file administrator.php.

```

152     {
153         try {
154             $this->psUpdateGame->execute(array(':update_name' => $name, ':update_description' =>
155             $description, ':update_idConsole' => $consoleId, ':update_id' => $idGame));
156         } catch (PDOException $e) {
157             print "Erreur !: " . $e->getMessage() . "<br>";
158             die();
159         }
160     }

```

4.1.3.5 uploadGame()

```

uploadGame (
    $gameFileName,
    $consoleId )

```

upload a game

Parameters

int	<i>\$gameFileName</i>	
int	<i>\$consoleId</i>	

Returns

bool

Definition at line 111 of file administrator.php.

```

112     {
113         $uploadIsValid = false;
114         $target_dir = "../games/" . $this->getConsoleFolderName($consoleId) . "/";
115
116         $target_file = basename($_FILES["fileGame"]["name"]);
117         $uploadOk = 1;
118         $fileType = strtolower(pathinfo($target_file, PATHINFO_EXTENSION));
119
120         //rename file
121         $newfilename = $gameFileName . '.' . $fileType;
122
123
124         // Check if file already exists
125         if (file_exists($target_file)) {
126             echo "Sorry, file already exists.";
127             $uploadOk = 0;
128         }
129         if ($uploadOk == 0) {

```

```

130         echo "Sorry, your file was not uploaded.";
131         // if everything is ok, try to upload file
132     } else {
133         if (move_uploaded_file($_FILES["fileGame"]["tmp_name"], $target_dir . $newfilename)) {
134             $uploadIsValid = true;
135         } else {
136             //Sorry, there was an error uploading your file
137         }
138     }
139     return $uploadIsValid;
140 }

```

Here is the call graph for this function:



Here is the caller graph for this function:



4.1.3.6 uploadGameImage()

```

uploadGameImage (
    $imageFileName )

```

upload an image

Parameters

string	<i>\$imageFileName</i>	
--------	------------------------	--

Returns

bool

Definition at line 167 of file administrator.php.

```

168     {

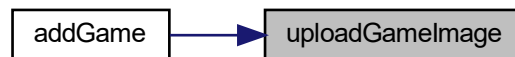
```

```

169     $uploadIsValid = false;
170     $target_dir = "img/games/";
171
172     $target_file = basename($_FILES["image"]["name"]);
173     $uploadOk = 1;
174
175     //rename file
176     $newfilename = $imageFileName;
177
178     // Check if file already exists
179     if (file_exists($target_file)) {
180         echo "Sorry, file already exists.";
181         $uploadOk = 0;
182     }
183     if ($uploadOk == 0) {
184         echo "Sorry, your file was not uploaded.";
185         // if everything is ok, try to upload file
186     } else {
187         if (move_uploaded_file($_FILES["image"]["tmp_name"], $target_dir . $newfilename)) {
188             $uploadIsValid = true;
189         } else {
190             //Sorry, there was an error uploading your file
191         }
192     }
193     return $uploadIsValid;
194 }

```

Here is the caller graph for this function:

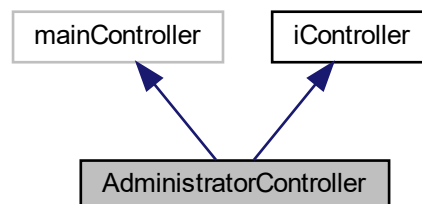


The documentation for this class was generated from the following file:

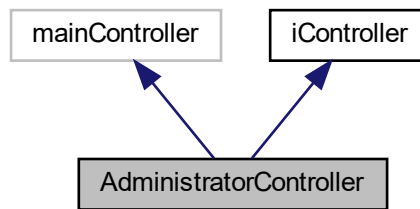
- [models/administrator.php](#)

4.2 AdministratorController Class Reference

Inheritance diagram for AdministratorController:



Collaboration diagram for AdministratorController:



Public Member Functions

- [formHandler\(\)](#)
- [__construct\(\)](#)
- [printHTML\(\)](#)

Data Fields

- `$administrator`
- `$game`
- `$categorie`

4.2.1 Detailed Description

Definition at line 10 of file administratorController.php.

4.2.2 Constructor & Destructor Documentation

4.2.2.1 __construct()

```
__construct ( )
```

default constructor

Definition at line 190 of file administratorController.php.

```
191 {  
192     $this->administrator = new Administrator();  
193     $this->game = new Games();  
194     $this->categorie = new Categories();  
195 }
```

4.2.3 Member Function Documentation

4.2.3.1 formHandler()

formHandler ()

used to handle if the user has resquest something

Returns

void

Implements [iController](#).

Definition at line 23 of file administratorController.php.

```

24 {
25
26     if (isset($_GET['e'])) {
27         $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_STRING);
28         //redirige l'utilisateur qui n'a pas les bon droits
29         $this->allowAccessTo(array(1));
30     }
31     // update game
32     if ($this->e == "updateGame") {
33         if (isset($_GET['id'])) {
34             $requestGame = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_STRING);
35             $this->idGameToUpdate = $requestGame;
36         } else {
37             header('Location:' . $_SERVER['HTTP_REFERER']);
38             exit;
39         }
40     }
41
42     // add game categorie
43     if ($this->e == "addGameCategorie") {
44         if (isset($_GET['id'])) {
45             $requestGame = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_STRING);
46             $this->idGameToUpdate = $requestGame;
47         } else {
48             header('Location:' . $_SERVER['HTTP_REFERER']);
49             exit;
50         }
51     }
52
53     // add categorie to game
54     if ($this->e == "addGameCategorieAdd") {
55         if (isset($_GET['idGame'])) {
56             $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_STRING);
57         } else {
58             header('Location:' . $_SERVER['HTTP_REFERER']);
59             exit;
60         }
61         if (isset($_GET['idCategorie'])) {
62             $idCategorie = filter_input(INPUT_GET, 'idCategorie', FILTER_SANITIZE_STRING);
63         } else {
64             header('Location:' . $_SERVER['HTTP_REFERER']);
65             exit;
66         }
67
68         $this->categorie->addCategorieToGame($idGame, $idCategorie);
69         header('Location:' . $_SERVER['HTTP_REFERER']);
70         exit;
71     }
72
73     // delete categorie from a game
74     if ($this->e == "delGameCategorie") {
75         if (isset($_GET['idGame'])) {
76             $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_STRING);
77         } else {
78             header('Location:' . $_SERVER['HTTP_REFERER']);
79             exit;
80         }
81         if (isset($_GET['idCategorie'])) {

```

```

82     $idCategorie = filter_input(INPUT_GET, 'idCategorie', FILTER_SANITIZE_STRING);
83 } else {
84     header('Location:' . $_SERVER['HTTP_REFERER']);
85     exit;
86 }
87
88 $this->categorie->delCategorieFromGame($idGame, $idCategorie);
89 header('Location:' . $_SERVER['HTTP_REFERER']);
90 exit;
91 }
92
93 //add game
94 if ($this->e == "addGameUpload") {
95     if (isset($_POST['name'])) {
96         $name = filter_input(INPUT_POST, 'name', FILTER_SANITIZE_STRING);
97     } else {
98         header('Location:' . $_SERVER['HTTP_REFERER']);
99         exit;
100     }
101
102     if (isset($_POST['description'])) {
103         $description = filter_input(INPUT_POST, 'description', FILTER_SANITIZE_STRING);
104     } else {
105         header('Location:' . $_SERVER['HTTP_REFERER']);
106         exit;
107     }
108
109     if (isset($_POST['imageName'])) {
110         $imageName = filter_input(INPUT_POST, 'imageName', FILTER_SANITIZE_STRING);
111     } else {
112         header('Location:' . $_SERVER['HTTP_REFERER']);
113         exit;
114     }
115
116     if (isset($_POST['console'])) {
117         $consoleId = filter_input(INPUT_POST, 'console', FILTER_SANITIZE_STRING);
118     } else {
119         header('Location:' . $_SERVER['HTTP_REFERER']);
120         exit;
121     }
122
123     if (isset($_POST['gameFileName'])) {
124         $gameFileName = filter_input(INPUT_POST, 'gameFileName', FILTER_SANITIZE_STRING);
125     } else {
126         header('Location:' . $_SERVER['HTTP_REFERER']);
127         exit;
128     }
129
130
131     $this->administrator->addGame($name, $description, $imageName, $consoleId, $gameFileName);
132
133     header('Location:' . $_SERVER['HTTP_REFERER']);
134     exit;
135 }
136
137 //add game
138 if ($this->e == "updateGameUpdate") {
139     if (isset($_POST['name'])) {
140         $name = filter_input(INPUT_POST, 'name', FILTER_SANITIZE_STRING);
141     } else {
142         header('Location:' . $_SERVER['HTTP_REFERER']);
143         exit;
144     }
145
146     if (isset($_POST['description'])) {
147         $description = filter_input(INPUT_POST, 'description', FILTER_SANITIZE_STRING);
148     } else {
149         header('Location:' . $_SERVER['HTTP_REFERER']);
150         exit;
151     }
152
153     if (isset($_POST['console'])) {
154         $consoleId = filter_input(INPUT_POST, 'console', FILTER_SANITIZE_STRING);
155     } else {
156         header('Location:' . $_SERVER['HTTP_REFERER']);
157         exit;
158     }
159
160     if (isset($_POST['idGame'])) {
161         $idGame = filter_input(INPUT_POST, 'idGame', FILTER_SANITIZE_STRING);
162     } else {
163         header('Location:' . $_SERVER['HTTP_REFERER']);
164         exit;
165     }
166
167
168     $this->administrator->updateGame($idGame, $name, $description, $consoleId);

```



```

169
170     header('Location:' . $_SERVER['HTTP_REFERER']);
171     exit;
172 }
173
174
175     if ($this->e == "addCategorie") {
176         if (isset($_POST['name'])) {
177             $name = filter_input(INPUT_POST, 'name', FILTER_SANITIZE_STRING);
178         }
179
180         if (isset($name)) {
181             $this->categorie->addCategorie($name);
182         }
183     }
184 }

```

4.2.3.2 printHTML()

```
printHTML ( )
```

print the html for the requested content

Returns

html

Implements [iController](#).

Definition at line 203 of file administratorController.php.

```

204 {
205
206     $html = '<main style="margin-top:20px">
207         <div class="container-md">';
208     $html .= $this->errorHandler();
209     if ($this->e == null) {
210         $html .= $this->htmlAdministratorHome();
211     }
212
213     if ($this->e == "addGame") {
214         $html .= $this->htmlNewGame();
215     }
216
217     if ($this->e == "updateGame") {
218         $html .= $this->htmlUpdateGame();
219     }
220
221     if ($this->e == "addCategorie") {
222         $html .= $this->htmlAddCategorie();
223     }
224
225     if ($this->e == "addGameCategorie") {
226         $html .= $this->htmlAddCategorieToGame();
227     }
228
229
230
231     $html .= "</div></main> ";
232     echo $html;
233 }

```

The documentation for this class was generated from the following file:

- controllers/[administratorController.php](#)

4.3 Categories Class Reference

Public Member Functions

- [__construct\(\)](#)
- [getListAllCategories\(\)](#)
- [getCategoriesOfGame\(int \\$idGame\)](#)
- [addCategorie\(string \\$categorieName\)](#)
- [addCategorieToGame\(int \\$idGame, int \\$idCategorie\)](#)
- [delCategorieFromGame\(int \\$idGame, int \\$idCategorie\)](#)

4.3.1 Detailed Description

Definition at line 10 of file categorie.php.

4.3.2 Constructor & Destructor Documentation

4.3.2.1 __construct()

`__construct()`

default constructor

Definition at line 31 of file categorie.php.

```

32     {
33         if ($this->dbh == null) {
34             try {
35                 $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
36                     PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
37                     PDO::ATTR_PERSISTENT => true
38                 ));
39                 //get all categories
40                 $sqlGetAllCategories = "SELECT * FROM categorie";
41                 $this->psGetAllCategories = $this->dbh->prepare($sqlGetAllCategories);
42                 $this->psGetAllCategories->setFetchMode(PDO::FETCH_ASSOC);
43
44                 //add categorie
45                 $sqlAddCategorie = "INSERT INTO categorie (name) VALUES (:categorie_name)";
46                 $this->psAddCategorie = $this->dbh->prepare($sqlAddCategorie);
47                 $this->psAddCategorie->setFetchMode(PDO::FETCH_ASSOC);
48
49                 //add categorie to game
50                 $sqlAddCategorieToGame = "INSERT INTO gamehascategorie (idGame,idCategorie) VALUES
(:insert_idGame, :insert_idCategorie)";
51                 $this->psAddCategorieToGame = $this->dbh->prepare($sqlAddCategorieToGame);
52
53                 //check if game has a specific categorie
54                 $sqlCheckIfGameHasCategorie = "SELECT * FROM gamehascategorie WHERE idCategorie =
:insert_idCategorie AND idGame = :insert_idGame";
55                 $this->psCheckIfGameHasCategorie = $this->dbh->prepare($sqlCheckIfGameHasCategorie);
56                 $this->psCheckIfGameHasCategorie->setFetchMode(PDO::FETCH_ASSOC);
57
58                 //del categorie from game
59                 $sqlDelCategorieFromGame = "DELETE FROM gamehascategorie WHERE idCategorie =
:del_idCategorie AND idGame = :del_idGame";
60                 $this->psDelCategorieFromGame = $this->dbh->prepare($sqlDelCategorieFromGame);
61
62                 //get categories of a game
63                 $sqlGameCategorie = "SELECT c.name, c.id FROM `gamehascategorie` as ghc
64                     LEFT JOIN categorie as c
65                     ON ghc.idCategorie = c.id
66                     LEFT JOIN game as g
67                     ON ghc.idGame = g.id
68                     WHERE idGame = :search_id";
69                 $this->psGameCategorie = $this->dbh->prepare($sqlGameCategorie);
70                 $this->psGameCategorie->setFetchMode(PDO::FETCH_ASSOC);
71             } catch (PDOException $e) {
72                 print "Erreur !: " . $e->getMessage() . "<br>";
73                 die();
74             }
75         }
76     }

```

4.3.3 Member Function Documentation

4.3.3.1 addCategorie()

```
addCategorie (
    string $categorieName )
```

add a new categorie in the database

Parameters

string	<i>\$categorieName</i>	
--------	------------------------	--

Returns

html

Definition at line 120 of file categorie.php.

```
121 {
122     try {
123         $this->psAddCategorie->execute(array(':categorie_name' => $categorieName));
124         $result = $this->psAddCategorie->fetchAll();
125     } catch (PDOException $e) {
126         print "Erreur !: " . $e->getMessage() . "<br>";
127         die();
128     }
129     return $result;
130 }
```

4.3.3.2 addCategorieToGame()

```
addCategorieToGame (
    int $idGame,
    int $idCategorie )
```

add a categorie to a game

Parameters

integer	<i>\$idGame</i>	
integer	<i>\$idCategorie</i>	

Returns

void

Definition at line 139 of file categorie.php.

```
140 {
141     $result = null;
```

```

142         try {
143             $this->psCheckIfGameHasCategorie->execute(array(':insert_idCategorie' => $idCategorie,
144                 ':insert_idGame' => $idGame));
145             $result = $this->psCheckIfGameHasCategorie->fetchAll();
146         } catch (PDOException $e) {
147             print "Erreur !: " . $e->getMessage() . "<br>";
148             die();
149         }
150         if ($result == null) {
151             try {
152                 $this->psAddCategorieToGame->execute(array(':insert_idCategorie' => $idCategorie,
153                     ':insert_idGame' => $idGame));
154             } catch (PDOException $e) {
155                 print "Erreur !: " . $e->getMessage() . "<br>";
156                 die();
157             }
158         }
159     }
160 }

```

4.3.3.3 delCategorieFromGame()

```

delCategorieFromGame (
    int $idGame,
    int $idCategorie )

```

delete a coterie of a game

Parameters

integer	<i>\$idGame</i>	
integer	<i>\$idCategorie</i>	

Returns

void

Definition at line 165 of file categorie.php.

```

166     {
167         try {
168             $this->psDelCategorieFromGame->execute(array(':del_idCategorie' => $idCategorie,
169                 ':del_idGame' => $idGame));
170         } catch (PDOException $e) {
171             print "Erreur !: " . $e->getMessage() . "<br>";
172             die();
173         }
174     }
175 }

```

4.3.3.4 getCategoriesOfGame()

```

getCategoriesOfGame (
    int $idGame )

```

get the categories of a game

Parameters

integer	<i>\$idGame</i>	
---------	-----------------	--

Returns

list of categories

Definition at line 102 of file categorie.php.

```

103     {
104         try {
105             $this->psGameCategorie->execute(array(':search_id' => $idGame));
106             $result = $this->psGameCategorie->fetchAll();
107         } catch (PDOException $e) {
108             print "Erreur !: " . $e->getMessage() . "<br>";
109             die();
110         }
111         return $result;
112     }

```

4.3.3.5 getListAllCategories()

getListAllCategories ()

returns list of all categories

Returns

array with all list

Definition at line 83 of file categorie.php.

```

84     {
85
86         try {
87             $this->psGetAllCategories->execute();
88             $result = $this->psGetAllCategories->fetchAll();
89         } catch (PDOException $e) {
90             print "Erreur !: " . $e->getMessage() . "<br>";
91             die();
92         }
93         return $result;
94     }

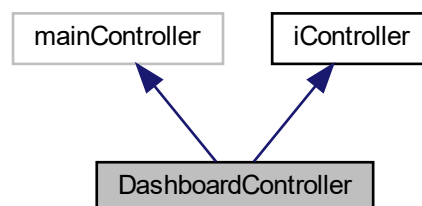
```

The documentation for this class was generated from the following file:

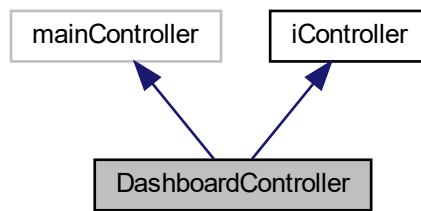
- [models/categorie.php](#)

4.4 DashboardController Class Reference

Inheritance diagram for DashboardController:



Collaboration diagram for DashboardController:



Public Member Functions

- [formHandler \(\)](#)
- [__construct \(\)](#)
- [printHTML \(\)](#)
- [htmlFormUpdatePassword \(\)](#)

Data Fields

- `$game`

4.4.1 Detailed Description

Definition at line 11 of file `dashboardController.php`.

4.4.2 Constructor & Destructor Documentation

4.4.2.1 __construct()

```
__construct ( )
```

default constructor

Definition at line 64 of file `dashboardController.php`.

```
65     {  
66         $this->game = new Games();  
67     }
```

4.4.3 Member Function Documentation

4.4.3.1 formHandler()

```
formHandler ( )
```

used to handle if the user has resquest something

Returns

void

Implements [iController](#).

Definition at line 21 of file dashboardController.php.

```

22     {
23         $_SESSION['title'] = "Caiman: Dashboard";
24         $this->allowAccessTo(array(1, 3));
25
26         $oldPassword = null;
27         $newPasswordRepeat = null;
28         $newPassword = null;
29
30         if (isset($_GET['e'])) {
31             $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_SPECIAL_CHARS);
32         }
33         // form update
34         if ($this->e == "updatePassword") {
35             $_SESSION['title'] = "Caiman: Update password";
36             if (isset($_POST['oldPassword'])) {
37                 $oldPassword = filter_input(INPUT_POST, 'oldPassword', FILTER_SANITIZE_STRING);
38             }
39             if (isset($_POST['newPassword'])) {
40                 $newPassword = filter_input(INPUT_POST, 'newPassword', FILTER_SANITIZE_STRING);
41             }
42             if (isset($_POST['newPasswordRepeat'])) {
43                 $newPasswordRepeat = filter_input(INPUT_POST, 'newPasswordRepeat',
44                 FILTER_SANITIZE_STRING);
45             }
46             if (isset($oldPassword) && isset($newPassword) && isset($newPasswordRepeat)) {
47                 $_SESSION['user']->updatePassword($newPassword, $newPasswordRepeat, $oldPassword);
48             }
49         }
50
51         // update if account if visible or not
52         if ($this->e == "updatePrivateAccount") {
53             if ($_SESSION['user']->idUser != -1) {
54                 $_SESSION['user']->updatePrivateAccount();
55                 header('Location:' . $_SERVER['HTTP_REFERER']);
56             }
57         }
58     }
59 }
```

4.4.3.2 htmlFormUpdatePassword()

```
htmlFormUpdatePassword ( )
```

create the html of the form to update the user's password

Returns

html

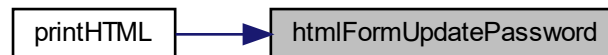
Definition at line 258 of file dashboardController.php.

```

259     {
260         $html = '<div class="d-inline-flex p-2 jumbotron width100 DarkJumbotron "
                style="background-color: #161b22;" >
261             <div class="container">
262                 <div class="row"><h2>Update your password</h2></div>
263                 <div class="row">
264
265                     <form action="?r=dashboard&e=updatePassword" method="post">
266                         <div class="form-group">
267                             <label for="oldPassword">Old password</label>
268                             <input type="password" class="form-control" id="oldPassword" name="oldPassword"
269                             placeholder="Old password">
270                         </div>
271                         <div class="form-group">
272                             <label for="newPassword">Password</label>
273                             <input type="password" class="form-control" id="newPassword" name="newPassword"
274                             placeholder="New password">
275                         </div>
276                         <div class="form-group">
277                             <label for="newPasswordRepeat">Password</label>
278                             <input type="password" class="form-control" id="newPasswordRepeat"
279                             name="newPasswordRepeat" placeholder="New password repeat">
280                         </div>
281                         <button type="submit" class="btn btn-primary">Submit</button>
282                     </form>
283                 </div>
284             </div>';
285         return $html;
286     }

```

Here is the caller graph for this function:

**4.4.3.3 printHTML()**

printHTML ()

print the html for the requested content

Returns

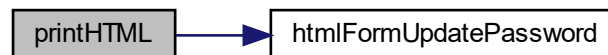
html

Implements [iController](#).

Definition at line 75 of file `dashboardController.php`.

```
76     {
77
78         $html = '<main style="margin-top:20px ">
79             <div class="container-md">';
80         echo $_SESSION['error'];
81         $html .= $this->errorHandler();
82
83         if ($this->e == null) {
84             $html .= $this->htmlFormHead();
85             $html .= $this->htmlFavoriteGames();
86             $html .= $this->htmlGameTime();
87         }
88
89         if ($this->e == "updatePassword") {
90             $html .= $this->htmlFormUpdatePassword();
91         }
92
93         $html .= "</div></main> ";
94
95         echo $html;
96     }
```

Here is the call graph for this function:



The documentation for this class was generated from the following file:

- [controllers/dashboardController.php](#)

4.5 Download Class Reference

Public Member Functions

- [downloadCaiman](#) ()

4.5.1 Detailed Description

Definition at line 10 of file `download.php`.

4.5.2 Member Function Documentation

4.5.2.1 downloadCaiman()

```
downloadCaiman ( )
```

used to download caiman

Returns

void

Definition at line 18 of file download.php.

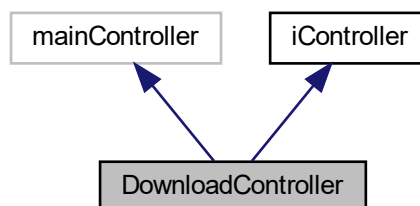
```
19  {
20      $filename = '../release/caiman.jpg'; // of course find the exact filename....
21      header('Pragma: public');
22      header('Expires: 0');
23      header('Cache-Control: must-revalidate, post-check=0, pre-check=0');
24      header('Cache-Control: private', false); // required for certain browsers
25      header('Content-Type: application/jpg');
26
27      header('Content-Disposition: attachment; filename="' . basename($filename) . '.jpg');
28      header('Content-Transfer-Encoding: binary');
29      header('Content-Length: ' . filesize($filename));
30
31      readfile($filename);
32
33      exit;
34  }
```

The documentation for this class was generated from the following file:

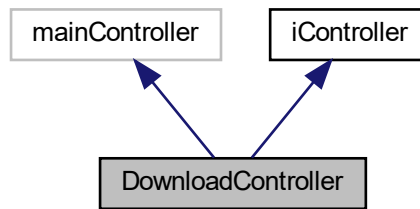
- [models/download.php](#)

4.6 DownloadController Class Reference

Inheritance diagram for DownloadController:



Collaboration diagram for DownloadController:



Public Member Functions

- [__construct\(\)](#)
- [formHandler\(\)](#)
- [printHTML\(\)](#)

Data Fields

- `$download`

4.6.1 Detailed Description

Definition at line 10 of file downloadController.php.

4.6.2 Constructor & Destructor Documentation

4.6.2.1 __construct()

```
__construct ( )
```

default constructor

Definition at line 18 of file downloadController.php.

```
19 {  
20     $this->download = new Download();  
21 }
```

4.6.3 Member Function Documentation

4.6.3.1 formHandler()

formHandler ()

used to handle if the user has resquest something

Returns

void

Implements [iController](#).

Definition at line 28 of file downloadController.php.

```

29 {
30     $_SESSION['title'] = "Caiman: Download";
31     if (isset($_GET['e'])) {
32         $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_STRING);
33     }
34
35     if ($this->e == null) {
36
37         if ($_SESSION['user']->idUser != -1) {
38             $this->e = "user";
39         } else {
40             $this->e = "visitor";
41         }
42     }
43
44     if ($this->e == "download") {
45
46         if ($_SESSION['user']->idUser != -1) {
47             $this->download->downloadCaiman();
48         } else {
49             header('?r=login');
50         }
51     }
52 }
```

4.6.3.2 printHTML()

printHTML ()

print the html for the resquested content

Returns

html

Implements [iController](#).

Definition at line 61 of file downloadController.php.

```

62 {
63     $html = '<main style="margin-top:20px">
64         <div class="container-md">';
65     $html .= $this->errorHandler();
66     if ($this->e == "user") {
67         $html .= $this->htmlUserDownload();
68     }
69
70     if ($this->e == "visitor") {
71         $html .= $this->htmlVisitorDownload();
72     }
73     $html .= "</div></main> ";
74     echo $html;
75 }
```

The documentation for this class was generated from the following file:

- controllers/[downloadController.php](#)

4.7 Games Class Reference

Public Member Functions

- [__construct](#) ()
- [getAllGames](#) ()
- [getRequestGames](#) (string \$gameName)
- [getTimeInGameUser](#) (int \$idUser, int \$idGame)
- [getListOfGameWithTimeUser](#) (int \$idUser)
- [getGameDetail](#) (int \$idGame)
- [getGamesInCategorie](#) (int \$idCategorie)
- [getFavoriteGamesOfUser](#) (int \$idUser)
- [addGameToFavoris](#) (int \$idUser, int \$idGame)
- [removeGameFromFavoris](#) (int \$idUser, int \$idGame)
- [checkIfGamelsAlreadyInFavoris](#) (int \$idUser, int \$idGame)

4.7.1 Detailed Description

Definition at line 9 of file games.php.

4.7.2 Constructor & Destructor Documentation

4.7.2.1 __construct()

```
__construct ( )
```

default contructor

Definition at line 37 of file games.php.

```

38     {
39         if ($this->dbh == null) {
40             try {
41                 $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
42                     PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
43                     PDO::ATTR_PERSISTENT => true
44                 ));
45                 //get all games
46                 $sqlGetAllGames = "SELECT * FROM game";
47                 $this->psGetAllGames = $this->dbh->prepare($sqlGetAllGames);
48                 $this->psGetAllGames->setFetchMode(PDO::FETCH_ASSOC);
49
50                 //get request games
51                 $sqlRequestGames = "SELECT * FROM game WHERE name LIKE :search_game";
52                 $this->psRequestGames = $this->dbh->prepare($sqlRequestGames);
53                 $this->psRequestGames->setFetchMode(PDO::FETCH_ASSOC);
54
55                 //get Time in game user
56                 $sqlTimeInGame = "SELECT * FROM timeingame WHERE idGame = :search_idGame AND idUser =
:search_idUser ";
57                 $this->psGetTimeInGame = $this->dbh->prepare($sqlTimeInGame);
58                 $this->psGetTimeInGame->setFetchMode(PDO::FETCH_ASSOC);
59
60                 //get game with time user
61                 $sqlGetGameWithTime = "SELECT * FROM timeingame WHERE idUser = :search_idUser ORDER BY
timeInMinute DESC";
62                 $this->psGetGameWithTime = $this->dbh->prepare($sqlGetGameWithTime);
63                 $this->psGetGameWithTime->setFetchMode(PDO::FETCH_ASSOC);
64
65                 //get detail game

```

```

66         $sqlGameDetail = "SELECT * FROM game WHERE id = :search_id";
67         $this->psGameDetail = $this->dbh->prepare($sqlGameDetail);
68         $this->psGameDetail->setFetchMode(PDO::FETCH_ASSOC);
69
70         //add game to favoris
71         $sqlAddGameToFavoris = "INSERT INTO favoritegame (idGame, idUser)
72         VALUES (:search_idGame, :search_idUser)";
73         $this->psAddGameToFavori = $this->dbh->prepare($sqlAddGameToFavoris);
74         $this->psAddGameToFavori->setFetchMode(PDO::FETCH_ASSOC);
75
76         //remove game to favoris
77         $sqlRemoveGameFormFavoris = "DELETE FROM favoritegame
78         WHERE idUser = :search_idUser AND idGame = :search_idGame";
79         $this->psRemoveGameFromFavori = $this->dbh->prepare($sqlRemoveGameFormFavoris);
80         $this->psRemoveGameFromFavori->setFetchMode(PDO::FETCH_ASSOC);
81
82         //check if already in favoris
83         $sqlCheckIfAlreadyFavoris = "SELECT * FROM favoritegame
84         WHERE iduser = :search_idUser AND idGame = :search_idGame";
85         $this->psCheckIfFavoris = $this->dbh->prepare($sqlCheckIfAlreadyFavoris);
86         $this->psCheckIfFavoris->setFetchMode(PDO::FETCH_ASSOC);
87
88         //get favorite game of user
89         $sqlFavoriteGameOfUser = "SELECT g.name, g.id, g.imageName FROM `favoritegame` as fg
90         LEFT JOIN game as g
91         ON fg.idGame = g.id
92         LEFT JOIN user as u
93         ON fg.iduser = u.id
94         WHERE iduser = :search_id";
95         $this->psFavoriteGameOfUser = $this->dbh->prepare($sqlFavoriteGameOfUser);
96         $this->psFavoriteGameOfUser->setFetchMode(PDO::FETCH_ASSOC);
97
98
99
100        //get list of games in a categorie
101        $sqlGameInCategorie = "SELECT g.name, g.id, g.imageName FROM `gamehascategorie` as ghc
102        LEFT JOIN game as g
103        ON ghc.idGame = g.id
104        LEFT JOIN categorie as c
105        ON ghc.idCategorie = c.id
106        WHERE idCategorie = :search_id";
107        $this->psGameInCategorie = $this->dbh->prepare($sqlGameInCategorie);
108        $this->psGameInCategorie->setFetchMode(PDO::FETCH_ASSOC);
109
110    } catch (PDOException $e) {
111        print "Erreur !: " . $e->getMessage() . "<br>";
112        die();
113    }
114    }
115    }

```

4.7.3 Member Function Documentation

4.7.3.1 addGameToFavoris()

```

addGameToFavoris (
    int $idUser,
    int $idGame )

```

returns add a game to a user's favorites

Returns

array

Definition at line 249 of file games.php.

```

250     {
251
252         try{

```

```

253         $this->psAddGameToFavoris->execute(array(':search_idUser' => $idUser, ':search_idGame' =>
    $idGame));
254
255     }catch (PDOException $e) {
256         print "Erreur !: " . $e->getMessage() . "<br>";
257         die();
258     }
259 }

```

4.7.3.2 checkIfGameIsAlreadyInFavoris()

```

checkIfGameIsAlreadyInFavoris (
    int $idUser,
    int $idGame )

```

returns if a game is already in favorite

Returns

void

Definition at line 284 of file games.php.

```

285     {
286         $boolResult = true;
287
288         try{
289             $this->psCheckIfFavoris->execute(array(':search_idUser' => $idUser, ':search_idGame' =>
    $idGame));
290             $result = $this->psCheckIfFavoris->fetchAll();
291             if ($result != null) {
292                 $boolResult = false;
293             }
294
295         }catch (PDOException $e) {
296             print "Erreur !: " . $e->getMessage() . "<br>";
297             die();
298         }
299         return $boolResult;
300     }

```

4.7.3.3 getAllGames()

```

getAllGames ( )

```

returns all games from the database

Returns

array of all games

Definition at line 122 of file games.php.

```

123     {
124
125         try{
126             $this->psGetAllGames->execute();
127             $result = $this->psGetAllGames->fetchAll();
128
129
130         }catch (PDOException $e) {
131             print "Erreur !: " . $e->getMessage() . "<br>";
132             die();
133         }
134         return $result;
135     }

```

4.7.3.4 getFavoriteGamesOfUser()

```
getFavoriteGamesOfUser (  
    int $idUser )
```

returns the list of the favorite game of a user

Returns

array of games

Definition at line 230 of file games.php.

```
231     {  
232         try{  
233             $this->psFavoriteGameOfUser->execute(array(':search_id' => $idUser));  
234             $result = $this->psFavoriteGameOfUser->fetchAll();  
235  
236         }catch (PDOException $e) {  
237             print "Erreur !: " . $e->getMessage() . "<br>";  
238             die();  
239         }  
240         return $result;  
241     }  
242 }
```

4.7.3.5 getGameDetail()

```
getGameDetail (  
    int $idGame )
```

returns details of a specif game

Returns

array with game detail

Definition at line 194 of file games.php.

```
195     {  
196  
197         try{  
198             $this->psGameDetail->execute(array(':search_id' => $idGame));  
199             $result = $this->psGameDetail->fetchAll();  
200  
201         }catch (PDOException $e) {  
202             print "Erreur !: " . $e->getMessage() . "<br>";  
203             die();  
204         }  
205         return $result;  
206     }  
207 }
```


4.7.3.6 getGamesInCategorie()

```
getGamesInCategorie (
    int $idCategorie )
```

returns games of a certain category

Returns

array of games

Definition at line 214 of file games.php.

```
215     {
216         try{
217             $this->psGameInCategorie->execute(array('search_id' => $idCategorie));
218             $result = $this->psGameInCategorie->fetchAll();
219         }catch (PDOException $e) {
220             print "Erreur !: " . $e->getMessage() . "<br>";
221             die();
222         }
223         return $result;
224     }
```

4.7.3.7 getListOfGameWithTimeUser()

```
getListOfGameWithTimeUser (
    int $idUser )
```

returns returns the games the player has played

Returns

array of games

Definition at line 176 of file games.php.

```
177     {
178         try{
179             $this->psGetGameWithTime->execute(array('search_idUser' => $idUser));
180             $result = $this->psGetGameWithTime->fetchAll();
181
182
183         }catch (PDOException $e) {
184             print "Erreur !: " . $e->getMessage() . "<br>";
185             die();
186         }
187         return $result;
188     }
```

4.7.3.8 getRequestGames()

```
getRequestGames (
    string $gameName )
```

returns games whose name matches the search

Returns

array of games

Definition at line 141 of file games.php.

```
142     {
143
144         try{
145             $this->psRequestGames->execute(array(':search_game' => '%'.$gameName.'%'));
146             $result = $this->psRequestGames->fetchAll();
147
148
149         }catch (PDOException $e) {
150             print "Erreur !: " . $e->getMessage() . "<br>";
151             die();
152         }
153         return $result;
154     }
```

4.7.3.9 getTimeInGameUser()

```
getTimeInGameUser (
    int $idUser,
    int $idGame )
```

returns play time of a user for a specific game

Returns

array time in game

Definition at line 160 of file games.php.

```
161     {
162         try{
163             $this->psGetTimeInGame->execute(array(':search_idGame' => $idGame, ':search_idUser' =>
164             $idUser));
165             $result = $this->psGetTimeInGame->fetchAll();
166         }catch (PDOException $e) {
167             print "Erreur !: " . $e->getMessage() . "<br>";
168             die();
169         }
170         return $result;
171     }
```

4.7.3.10 removeGameFromFavoris()

```
removeGameFromFavoris (
    int $idUser,
    int $idGame )
```

returns remove a game to a user's favorites

Returns

void

Definition at line 266 of file games.php.

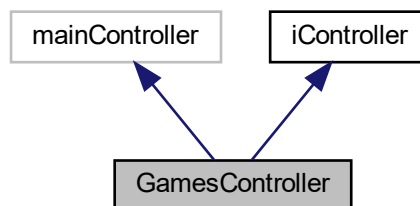
```
267     {
268
269         try{
270             $this->psRemoveGameFromFavori->execute(array(':search_idUser' => $idUser,':search_idGame' =>
                $idGame));
271
272
273         }catch (PDOException $e) {
274             print "Erreur !: " . $e->getMessage() . "<br>";
275             die();
276         }
277     }
```

The documentation for this class was generated from the following file:

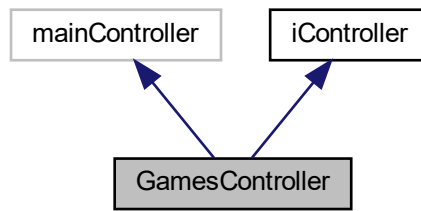
- [models/games.php](#)

4.8 GameController Class Reference

Inheritance diagram for GameController:



Collaboration diagram for GameController:



Public Member Functions

- `__construct ()`
- `formHandler ()`
- `printHTML ()`
- `getListAllGames ()`
- `getRequestedGames ()`
- `getGameDetail ()`
- `getGamesFromCategorie ()`
- `recherchFull ()`
- `recherchNotFull ()`

Data Fields

- `$games`
- `$categorie`
- `$requestedgame = null`

4.8.1 Detailed Description

Definition at line 10 of file `gamesController.php`.

4.8.2 Constructor & Destructor Documentation

4.8.2.1 `__construct()`

```
__construct ( )
```

default constructor

Definition at line 23 of file `gamesController.php`.

```
24     {  
25         $this->games = new Games();  
26         $this->categorie = new Categories();  
27     }
```

4.8.3 Member Function Documentation

4.8.3.1 formHandler()

formHandler ()

used to handle if the user has resquest something

Returns

void

Implements [iController](#).

Definition at line 34 of file gamesController.php.

```

35     {
36         $_SESSION['title'] = "Caiman: Games";
37         $requestGame = null;
38         $result = null;
39         if (isset($_GET['e'])) {
40             $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_STRING);
41         }
42
43         if ($this->e == "requestGame") {
44
45             if (isset($_POST['gameName'])) {
46                 $requestGame = filter_input(INPUT_POST, 'gameName', FILTER_SANITIZE_STRING);
47                 $_SESSION['title'] = "Caiman: Search " . $requestGame;
48             }
49
50             if (isset($requestGame)) {
51                 $result = $this->games->getRequestGames($requestGame);
52             }
53         }
54
55         if ($this->e == "detail") {
56
57             if (isset($_GET['idGame'])) {
58                 $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_STRING);
59                 $this->idGame = $idGame;
60             }
61
62             if (isset($idGame)) {
63                 $result = $this->games->getGameDetail($idGame);
64                 $_SESSION['title'] = "Caiman: " . $result[0]["name"];
65             }
66         }
67
68         if ($this->e == "categorie") {
69             $_SESSION['title'] = "Caiman: Categorie";
70             if (isset($_GET['idCategorie'])) {
71                 $idcategory = filter_input(INPUT_GET, 'idCategorie', FILTER_SANITIZE_STRING);
72                 $this->idcategory = $idcategory;
73             }
74
75             if (isset($idcategory)) {
76                 $result = $this->games->getGamesInCategorie($idcategory);
77             }
78         }
79
80         if ($this->e == "addFavoris") {
81             $_SESSION['title'] = "Caiman: Favorite";
82             if (isset($_GET['idGame'])) {
83                 $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_NUMBER_INT);
84             }
85
86             if (isset($idGame)) {
87                 $result = $this->games->addGameToFavoris($_SESSION['user']->idUser, $idGame);
88                 header('Location:' . $_SERVER['HTTP_REFERER']);
89                 $_SESSION['error'] = "Favorite added";
90             }
91         }
92     }

```

```

93
94     if ($this->e == "removeFavoris") {
95         if (isset($_GET['idGame'])) {
96             $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_NUMBER_INT);
97         }
98
99         if (isset($idGame)) {
100             $result = $this->games->removeGameFromFavoris($_SESSION['user']->idUser, $idGame);
101             header('Location:' . $_SERVER['HTTP_REFERER']);
102             $_SESSION['error'] = "Favorite removed";
103         }
104     }
105
106     $this->requestedgame = $result;
107 }

```

4.8.3.2 getGameDetail()

getGameDetail ()

create the page of a specific game

Returns

html

Definition at line 190 of file gamesController.php.

```

191 {
192
193     $gameDetail = $this->games->getGameDetail($this->idGame);
194     $category = $this->category->getCategoriesOfGame($this->idGame);
195     $html = "";
196
197     $html .= '</br>
198         <div class="row">
199             <div class="col">
200                 
201             </div>
202             <div class="col">
203                 <h2 class="card-title">' . $gameDetail[0]['name'] . '</h2>
204                 <p class="card-title">' . $gameDetail[0]['description'] . '</p>
205                 </br>
206                 <div class="list-group">' ;
207                 if ($_SESSION['user']->idUser != -1) {
208                     if ($this->games->checkIfGameIsAlreadyInFavoris($_SESSION['user']->idUser,
209 $gameDetail[0]['id'])) {
210                         $html .= '<a class="btn btn-outline-success " href="?r=games&e=addFavoris&idGame=' .
211 $gameDetail[0]['id'] . '" role="button">Add to favorite</a>' ;
212                     } else {
213                         $html .= '<a class="btn btn-outline-warning " href="?r=games&e=removeFavoris&idGame=' .
214 $gameDetail[0]['id'] . '" role="button">Remove favorite</a>' ;
215                     }
216                     if ($_SESSION['user']->role == 1) {
217                         $html .= '</br> <a class="btn btn-outline-danger " href="?r=administrator&e=updateGame&id=' .
218 $gameDetail[0]['id'] . '" role="button">Update game</a>' ;
219                         $html .= '</br> <a class="btn btn-outline-danger "
220 href="?r=administrator&e=addGameCategories&id=' . $gameDetail[0]['id'] . '" role="button">Update/add
221 categories</a>' ;
222                     }
223                 }
224                 $html .= '</div>
225                 <h3 class="card-title">Categories</h3>
226                 <div class="list-group">' ;
227                 foreach ($category as $key => $cat) {
228                     $html .= '<a href="?r=games&e=category&idCategorie=' . $cat['id'] . '">button type="button"
229 class="btn btn-outline-success btnCategorie margintop10">' . $cat['name'] . '</button></a>' ;
230                 }
231                 $html .= '
232             </div>
233         </div>
234     </div>
235 }

```

```
232
233     ' ;
234
235
236     $html .= " ;
237     return $html;
238 }
```

Here is the caller graph for this function:



4.8.3.3 getGamesFromCategorie()

```
getGamesFromCategorie ( )
```

create a list of game of a specific categorie

Returns

html

Definition at line 245 of file gamesController.php.

```
246 {
247
248     $html = '<div class="cardGameBox box">';
249     $listGamesBrut = $this->requestedgame;
250
251     foreach ($listGamesBrut as $key => $game) {
252
253         $html .= $this->createCardHTML($game);
254     }
255
256     $html .= '</div>';
257     return $html;
258 }
```

Here is the caller graph for this function:



4.8.3.4 getListAllGames()

```
getListAllGames ( )
```

create the html of the list of all the games

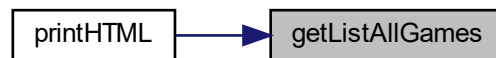
Returns

html

Definition at line 149 of file gamesController.php.

```
150     {
151         $html = '<div class="cardGameBox box">';
152
153         $listGamesBrut = $this->games->getAllGames();
154
155         foreach ($listGamesBrut as $key => $games) {
156
157             $html .= $this->createCardHTML($games);
158         }
159
160         $html .= '</div>';
161         return $html;
162     }
163 }
```

Here is the caller graph for this function:



4.8.3.5 getRequestedGames()

```
getRequestedGames ( )
```

create a list of the requested games

Returns

html

Definition at line 170 of file gamesController.php.

```
171     {
172
173         $html = '<div class="cardGameBox box">';
174         $listGamesBrut = $this->requestedgame;
175         foreach ($listGamesBrut as $key => $games) {
176
177             $html .= $this->createCardHTML($games);
178         }
179
180         $html .= '</div>';
181     }
```

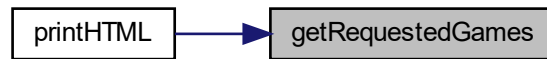


```

182         return $html;
183     }

```

Here is the caller graph for this function:



4.8.3.6 printHTML()

```
printHTML ( )
```

print the html for the resquested content

Returns

html

Implements [iController](#).

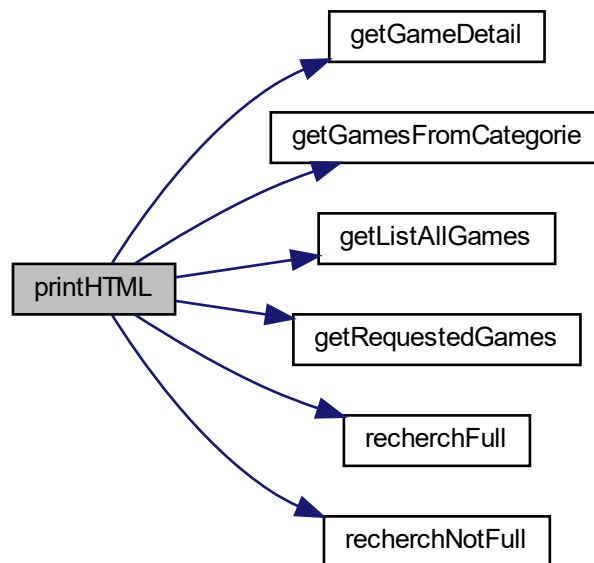
Definition at line 115 of file gamesController.php.

```

116     {
117         $html = '<main style="margin-top:20px">
118         <div class="container-md">';
119         $html .= $this->errorHandler();
120         if ($this->e == null) {
121             $html .= $this->recherchFull();
122             $html .= $this->getListAllGames();
123         }
124
125         if ($this->e == "requestGame") {
126             $html .= $this->recherchFull();
127             $html .= $this->getRequestedGames();
128         }
129
130         if ($this->e == "detail") {
131             $html .= $this->recherchNotFull();
132             $html .= $this->getGameDetail();
133         }
134
135         if ($this->e == "categorie") {
136             $html .= $this->recherchFull();
137             $html .= $this->getGamesFromCategorie();
138         }
139
140         $html .= "</div></main> ";
141         echo $html;
142     }

```

Here is the call graph for this function:



4.8.3.7 recherchFull()

```
recherchFull ( )
```

create the html of a form to research game and to display the list of categorie

Returns

html

Definition at line 297 of file gamesController.php.

```

298     {
299         $html = "";
300
301         $html .= '<div class="jumbotron DarkJumbotron" style="background-color: #161b22;">
302         <div class="container">
303             <h1 class="display-5">Research</h1>
304
305             <form class="row g-3" action="?r=games&e=requestGame" method="post">
306
307                 <div class="col-auto">
308                     <input type="texte" class="form-control" id="gameName" name="gameName"
placeholder="Mario">
309                 </div>
310                 <div class="col-auto">
311                     <button type="submit" class="btn btn-success mb-3">Research</button>
312                 </div>
313             </form>
314             <h4>Categories:</h4>
315             <p>
316
317         ' ;
  
```

```

318         foreach ($this->categorie->getListAllCategories() as $key => $cat) {
319             $html .= ' <a class="btn btn-outline-success btnCategorie "
href="?r=games&e=categorie&idCategorie=' . $cat['id'] . '" role="button">' . $cat['name'] . ' </a>';
320         }
321         $html .= '
322     </p>
323 </div>
324 </div>';
325
326     return $html;
327 }

```

Here is the caller graph for this function:



4.8.3.8 recherchNotFull()

```
recherchNotFull ( )
```

create the html of a form to research game

Returns

html

Definition at line 334 of file gamesController.php.

```

335     {
336         $html = "";
337
338         $html .= ' <div class="card " style="background-color: #0d1117;">
339         <div class="card-body container DarkJumbotron">
340             <h2 class="card-title ">Research</h2>
341
342             <form class="row g-3" action="?r=games&e=requestGame" method="post">
343
344                 <div class="col-auto">
345                     <input type="text" class="form-control" id="gameName" name="gameName"
placeholder="Mario">
346                 </div>
347                 <div class="col-auto">
348                     <button type="submit" class="btn btn-success mb-3">Research</button>
349                 </div>
350             </form>
351         </div>
352         </div>';
353
354         return $html;
355     }

```

Here is the caller graph for this function:

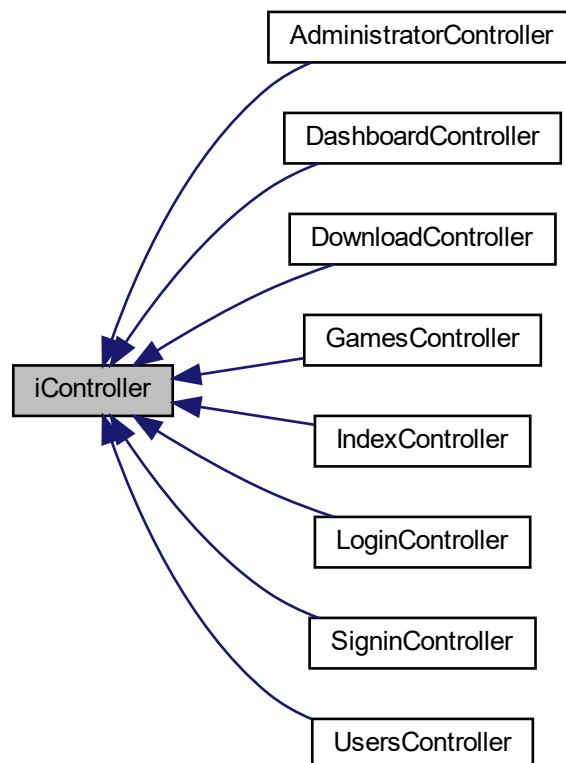


The documentation for this class was generated from the following file:

- [controllers/gamesController.php](#)

4.9 iController Interface Reference

Inheritance diagram for iController:



Public Member Functions

- `formHandler ()`
- `printHTML ()`

4.9.1 Detailed Description

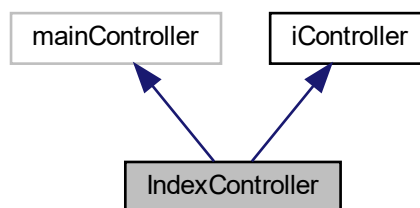
Definition at line 10 of file `interfaceController.php`.

The documentation for this interface was generated from the following file:

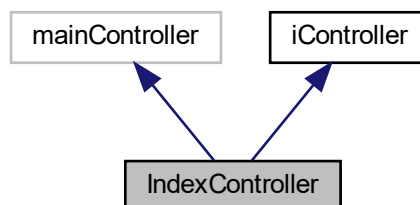
- `controllers/interfaceController.php`

4.10 IndexController Class Reference

Inheritance diagram for IndexController:



Collaboration diagram for IndexController:



Public Member Functions

- `formHandler ()`
- `printHTML ()`

4.10.1 Detailed Description

Definition at line 10 of file indexController.php.

4.10.2 Member Function Documentation

4.10.2.1 formHandler()

formHandler ()

used to handle if the user has resquest something

Returns

void

Implements [iController](#).

Definition at line 18 of file indexController.php.

```
19 {
20     $_SESSION['title'] = "Caiman: Home";
21 }
```

4.10.2.2 printHTML()

printHTML ()

print the html for the requested content

Returns

html

Implements [iController](#).

Definition at line 29 of file indexController.php.

```
30 {
31     $html = "";
32
33
34     $html .= '
35     <main style="margin-top:20px ">
36         <div class="container-md">
37             ' ;
38     $html .= $this->errorHandler();
39     $html .= '
40         <div class="jumbotron jumbotron-fluid DarkJumbotron width100" style="background-color: #161b22;">
41             <div class="row py-lg-5">
42                 <div class="col-lg-6 col-md-8 mx-auto">
43                     <h1 class="fw-light greenText">Caiman</h1>
44                     <p class="lead text-muted">The easiest way to use emulators.</p>
45                     <p>
46                         <a href="?r=download" class="btn btn-success my-2">Download</a>
47                         <a href="?r=login" class="btn btn-success my-2">Create account</a>
48                         <a href="?r=games" class="btn btn-success my-2">Watch games list</a>
49                     </p>
50                 </div>
51             </div>
52         </div>
53     </div>
54
55     </main>
56
57 ' ;
58     echo $html;
59 }
```

The documentation for this class was generated from the following file:

- controllers/[indexController.php](#)

4.11 Login Class Reference

Public Member Functions

- [checkLogin\(\)](#)

Data Fields

- `$search_username` = null
- `$search_password` = null
- `$arrayInfo` = null

4.11.1 Detailed Description

Definition at line 9 of file login.php.

4.11.2 Member Function Documentation

4.11.2.1 checkLogin()

`checkLogin ()`

check if there is a match

Returns

bool

Definition at line 48 of file login.php.

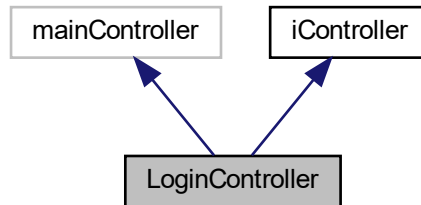
```
49     {
50         $returnArray = null;
51         try{
52             $this->psLogin->execute(array(':search_username' => $this->search_username));
53             $result = $this->psLogin->fetchAll();
54             if ($result != null) {
55                 if (password_verify( $this->search_password,$result[0]["password"]) ) {
56                     $returnArray = $result;
57                     $_SESSION['error'] = "Welcome back: ". $result[0]['username'];
58                 }else
59                 {
60                     $_SESSION['error'] = "Invalid log in";
61                 }
62             }
63         }
64         }catch (PDOException $e) {
65             print "Erreur !: " . $e->getMessage() . "<br>";
66             die();
67         }
68         return $returnArray;
69     }
```

The documentation for this class was generated from the following file:

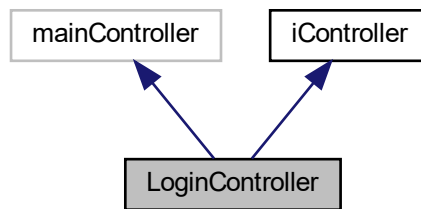
- `models/login.php`

4.12 LoginController Class Reference

Inheritance diagram for LoginController:



Collaboration diagram for LoginController:



Public Member Functions

- [__construct](#) ()
- [formHandler](#) ()
- [printHTML](#) ()

Data Fields

- `$login`

4.12.1 Detailed Description

Definition at line 10 of file loginController.php.

4.12.2 Constructor & Destructor Documentation

4.12.2.1 __construct()

`__construct ()`

default constructor

Definition at line 18 of file loginController.php.

```
19 {
20     $this->login = new Login();
21 }
```

4.12.3 Member Function Documentation

4.12.3.1 formHandler()

`formHandler ()`

used to handle if the user has request something

Returns

void

Implements [iController](#).

Definition at line 28 of file loginController.php.

```
29 {
30     if (isset($_GET['e'])) {
31         $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_STRING);
32     }
33
34     if ($this->e == "login") {
35         $_SESSION['title'] = "Caiman: Login";
36         if (isset($_POST['username'])) {
37             $username = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_STRING);
38         }
39         if (isset($_POST['password'])) {
40             $password = filter_input(INPUT_POST, 'password', FILTER_SANITIZE_STRING);
41         }
42
43         if (isset($password) && isset($username)) {
44             $this->login->search_username = $username;
45             $this->login->search_password = $password;
46
47             $usersInfos = $this->login->checkLogin();
48
49             if (isset($usersInfos)) {
50                 $_SESSION['user'] = new User($usersInfos[0]['username'], $usersInfos[0]['email'],
51                 $usersInfos[0]['idRole'], $usersInfos[0]['id']);
52                 header('Location: ' . $_SERVER['HTTP_REFERER']);
53                 exit;
54             }
55         }
56     }
```

4.12.3.2 printHTML()

```
printHTML ( )
```

print the html for the requested content

Returns

void

Implements [iController](#).

Definition at line 65 of file loginController.php.

```
66 {
67
68     $html = '<main style="margin-top:20px ">
69         <div class="container-md">';
70     $html .= $this->errorHandler();
71     $html .= $this->htmlFormHead();
72
73     $html .= "</div></main> ";
74
75     echo $html;
76 }
```

The documentation for this class was generated from the following file:

- [controllers/loginController.php](#)

4.13 MainController Class Reference

Public Member Functions

- [__construct](#) ()
- [allowAccessTo](#) (\$allowAccessTold)
- [errorHandler](#) ()

4.13.1 Detailed Description

Definition at line 10 of file mainController.php.

4.13.2 Constructor & Destructor Documentation

4.13.2.1 __construct()

```
__construct ( )
```

default constructor

Definition at line 16 of file mainController.php.

```
17 {
18 }
```

4.13.3 Member Function Documentation

4.13.3.1 allowAccessTo()

```
allowAccessTo (
    $allowAccessToId )
```

used to set the acces of a page you need to give the the list of role who can acces the page

Parameters

<i>[type]</i>	\$allowAccessToId
---------------	-------------------

Returns

void

Definition at line 26 of file mainController.php.

```
27     {
28
29         $isValid = false;
30         foreach ($allowAccessToId as $key => $validId) {
31             if ($validId == $_SESSION['user']->role) {
32                 $isValid = true;
33             }
34         }
35
36         if ($isValid == false) {
37             header('Location: index.php');
38             $_SESSION['error'] = "You can't access this page!";
39             exit;
40         }
41     }
```

4.13.3.2 errorHandler()

```
errorHandler ( )
```

create the html of an error

Returns

html

Definition at line 48 of file mainController.php.

```
49     {
50         $html = "";
51         if (isset($_SESSION['error']) && $_SESSION['error'] != null) {
52
53             $html .= '
54             <div class=" warningJumbotron errorMessageDiv" style="background-color: #161b22; ">
55
56                 <h5>' . $_SESSION['error'] . '</h5>
57
58             </div>';
59             $_SESSION['error'] = null;
60         }
61         return $html;
62     }
```

The documentation for this class was generated from the following file:

- controllers/[mainController.php](#)

4.14 Signin Class Reference

Public Member Functions

- [__construct\(\)](#)
- [newUser\(\)](#)
- [checkIfUsernameAlreadyTaken\(\)](#)
- [checkIfEmailAlreadyTaken\(\)](#)

Data Fields

- `$insert_username` = null
- `$insert_password` = null
- `$insert_password_repeat` = null
- `$insert_email` = null

4.14.1 Detailed Description

Definition at line 9 of file `signin.php`.

4.14.2 Constructor & Destructor Documentation

4.14.2.1 __construct()

```
__construct ( )
```

default constructor

Definition at line 31 of file `signin.php`.

```

32     {
33         if ($this->dbh == null) {
34             try {
35                 $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
36                     PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
37                     PDO::ATTR_PERSISTENT => true
38                 ));
39
40                 // check if email already used
41                 $sqlrequestEmail = "SELECT * FROM user WHERE email = :search_email ";
42                 $this->psCheckEmail = $this->dbh->prepare($sqlrequestEmail);
43                 $this->psCheckEmail->setFetchMode(PDO::FETCH_ASSOC);
44
45                 // check if username already used
46                 $sqlrequestUsername = "SELECT * FROM user WHERE username = :search_username ";
47                 $this->psCheckUsername = $this->dbh->prepare($sqlrequestUsername);
48                 $this->psCheckUsername->setFetchMode(PDO::FETCH_ASSOC);
49
50                 $sqlInsert = "INSERT INTO user (username, password, email)
51                     VALUES (:insert_username, :insert_password, :insert_email)";
52                 $this->psInsert = $this->dbh->prepare($sqlInsert);
53                 $this->psInsert->setFetchMode(PDO::FETCH_ASSOC);
54             } catch (PDOException $e) {
55                 print "Erreur !: " . $e->getMessage() . "<br>";
56                 die();
57             }
58         }
59     }

```

4.14.3 Member Function Documentation

4.14.3.1 checkIfEmailAlreadyTaken()

checkIfEmailAlreadyTaken ()

check that the email is not already taken

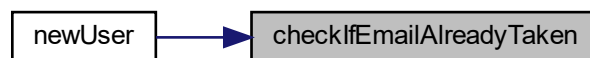
Returns

bool

Definition at line 116 of file signin.php.

```
117 {
118     $istaken = true;
119     try {
120         $this->psCheckEmail->execute(array(':search_email' => $this->insert_email));
121         $result = $this->psCheckEmail->fetchAll();
122         if ($result == null) {
123             $istaken = false;
124         }
125     } catch (PDOException $e) {
126         print "Erreur !: " . $e->getMessage() . "<br>";
127         die();
128     }
129     return $istaken;
130 }
```

Here is the caller graph for this function:



4.14.3.2 checkIfUsernameAlreadyTaken()

checkIfUsernameAlreadyTaken ()

check that the username is not already taken

Returns

bool

Definition at line 95 of file signin.php.

```
96 {
97     $istaken = true;
98     try {
99         $this->psCheckUsername->execute(array(':search_username' => $this->insert_username));
100         $result = $this->psCheckUsername->fetchAll();
101         if ($result == null) {
102             $istaken = false;
103         }
104     } catch (PDOException $e) {
105         print "Erreur !: " . $e->getMessage() . "<br>";
106         die();
107     }
108     return $istaken;
109 }
```

4.14.3.3 newUser()

newUser ()

add a mew user in the database

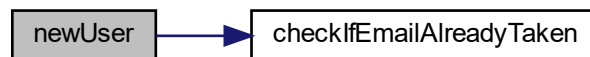
Returns

void

Definition at line 66 of file signin.php.

```
67     {
68         $isValid = true;
69         if ($this->checkIfEmailAlreadyTaken()) {
70             $_SESSION['error'] = "Email already used";
71             $isValid = false;
72         }
73         if ($this->checkIfUsernameAlreadyTaken()) {
74             $_SESSION['error'] = "Username alredy used";
75             $isValid = false;
76         }
77         if ($isValid) {
78
79
80             try {
81                 $this->psInsert->execute(array(':insert_username' => $this->insert_username,
82                 ':insert_password' => password_hash($this->insert_password, PASSWORD_DEFAULT), ':insert_email' =>
83                 $this->insert_email));
84                 $_SESSION['error'] = "Account created";
85             } catch (PDOException $e) {
86                 print "Erreur !: " . $e->getMessage() . "<br>";
87                 die();
88             }
89         }
90     }
```

Here is the call graph for this function:

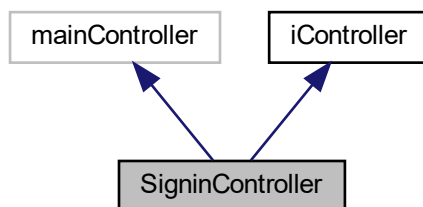


The documentation for this class was generated from the following file:

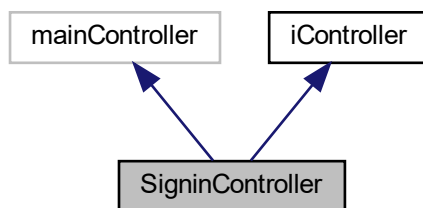
- [models/signin.php](#)

4.15 SigninController Class Reference

Inheritance diagram for SigninController:



Collaboration diagram for SigninController:



Public Member Functions

- [__construct\(\)](#)
- [formHandler\(\)](#)
- [printHTML\(\)](#)

Data Fields

- `$signin`

4.15.1 Detailed Description

Definition at line 10 of file `signinController.php`.

4.15.2 Constructor & Destructor Documentation

4.15.2.1 __construct()

`__construct ()`

default contructor

Definition at line 18 of file signinController.php.

```
19     {
20         $this->signin = new Signin();
21     }
```

4.15.3 Member Function Documentation

4.15.3.1 formHandler()

`formHandler ()`

used to handle if the user has resquest something

Returns

void

Implements [iController](#).

Definition at line 28 of file signinController.php.

```
29     {
30         if (isset($_GET['e'])) {
31             $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_SPECIAL_CHARS);
32         }
33         if ($this->e == "signin") {
34             if (isset($_POST['username'])) {
35                 $this->signin->insert_username = filter_input(INPUT_POST, 'username',
36                     FILTER_SANITIZE_SPECIAL_CHARS);
37             }
38             if (isset($_POST['password'])) {
39                 $this->signin->insert_password = filter_input(INPUT_POST, 'password',
40                     FILTER_SANITIZE_SPECIAL_CHARS);
41             }
42             if (isset($_POST['passwordRepeat'])) {
43                 $this->signin->insert_password_repeat = filter_input(INPUT_POST, 'passwordRepeat',
44                     FILTER_SANITIZE_SPECIAL_CHARS);
45             }
46             if (isset($_POST['email'])) {
47                 $this->signin->insert_email = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL);
48             }
49             if (isset($this->signin->insert_password) && isset($this->signin->insert_username) &&
50                 isset($this->signin->insert_password_repeat) && isset($this->signin->insert_email)) {
51                 if ($this->signin->insert_password != $this->signin->insert_password_repeat) {
52                     $_SESSION['error'] = "Password does not match";
53                     header('Location:' . $_SERVER['HTTP_REFERER']);
54                     exit;
55                 }
56             }
57         }
58     }
```



```

56
57         $this->signin->newUser();
58
59         header('Location:' . $_SERVER['HTTP_REFERER']);
60         exit;
61     } else {
62         $_SESSION['error'] = "form not completed";
63         header('Location:' . $_SERVER['HTTP_REFERER']);
64         exit;
65     }
66 }
67 }

```

4.15.3.2 printHTML()

```
printHTML ( )
```

print the html for the requested content

Returns

void

Implements [iController](#).

Definition at line 75 of file `signinController.php`.

```

76     {
77     }

```

The documentation for this class was generated from the following file:

- [controllers/signinController.php](#)

4.16 User Class Reference

Public Member Functions

- [__construct](#) (string \$usernamep, string \$emailp, string \$idRolep, int \$idUserp)
- [updatePassword](#) (string \$newPassword, string \$newPasswordRepeat, string \$oldPassword)
- [updatePrivateAccount](#) ()
- [getPrivateAccount](#) ()

Data Fields

- **\$username**
- **\$email**
- **\$role**
- **\$idUser**

4.16.1 Detailed Description

Definition at line 10 of file `user.php`.

4.16.2 Constructor & Destructor Documentation

4.16.2.1 __construct()

```
__construct (
    string $usernamep,
    string $emailp,
    string $idRolep,
    int $idUserp )
```

default constructor

Parameters

string	<i>\$usernamep</i>	
string	<i>\$emailp</i>	
string	<i>\$idRolep</i>	
integer	<i>\$idUserp</i>	

Definition at line 26 of file user.php.

```
27     {
28
29         $this->username = $usernamep;
30         $this->email = $emailp;
31         $this->role = $idRolep;
32         $this->idUser = $idUserp;
33     }
```

4.16.3 Member Function Documentation

4.16.3.1 getPrivateAccount()

```
getPrivateAccount ( )
```

use to know if the account is privat or not

Returns

bool

Definition at line 109 of file user.php.

```
110     {
111         $dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
112             PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
113             PDO::ATTR_PERSISTENT => true
114         ));
115
116         try {
117             $sqlGetPrivateAccount = "SELECT privateAccount FROM user WHERE id = :id_user";
118             $psGetPrivateAccount = $dbh->prepare($sqlGetPrivateAccount);
119             $psGetPrivateAccount->setFetchMode(PDO::FETCH_ASSOC);
```

```

120         $psGetPrivateAccount->execute(array(':id_user' => $this->idUser));
121         $result = $psGetPrivateAccount->fetchAll();
122     } catch (PDOException $e) {
123         print "Erreur !: " . $e->getMessage() . "<br>";
124         die();
125     }
126
127     return $result[0]['privateAccount'];
128 }

```

Here is the caller graph for this function:



4.16.3.2 updatePassword()

```

updatePassword (
    string $newPassword,
    string $newPasswordRepeat,
    string $oldPassword )

```

update the user password in the database

Parameters

string	<i>\$newPassword</i>	
string	<i>\$newPasswordRepeat</i>	
string	<i>\$oldPassword</i>	

Returns

int

Definition at line 43 of file user.php.

```

44     {
45         $dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
46             PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
47             PDO::ATTR_PERSISTENT => true
48         ));
49
50         $hasBeenUpdated = 1;
51         if (password_verify($oldPassword, $this->getUserPassword())) {
52
53             if ($newPasswordRepeat == $newPassword) {
54                 try {
55                     $sqlUpdatePassword = "UPDATE user SET password = :update_password WHERE id =
: id_user";
56                     $psUpdatePassword = $dbh->prepare($sqlUpdatePassword);
57                     $psUpdatePassword->execute(array(':update_password' => password_hash($newPassword,
PASSWORD_DEFAULT), ':id_user' => $this->idUser));

```

```

58
59         $hasBeenUpdated = 0;
60     } catch (PDOException $e) {
61         print "Erreur !: " . $e->getMessage() . "<br>";
62         die();
63     }
64     } else {
65         $hasBeenUpdated = 2;
66     }
67 } else {
68     $hasBeenUpdated = 4;
69 }
70
71 return $hasBeenUpdated;
72 }

```

4.16.3.3 updatePrivateAccount()

updatePrivateAccount ()

update if the account is private or not

Returns

void

Definition at line 79 of file user.php.

```

80 {
81     $dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
82         PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
83         PDO::ATTR_PERSISTENT => true
84     ));
85
86     $userIsPrivate = $this->getPrivateAccount();
87
88     if ($userIsPrivate == 0) {
89         $userSetPrivateTo = 1;
90     } else {
91         $userSetPrivateTo = 0;
92     }
93
94     try {
95         $sqlUpdatePrivateAccount = "UPDATE user SET privateAccount = :update_private_account WHERE
id = :id_user";
96         $psUpdatePrivateAccount = $dbh->prepare($sqlUpdatePrivateAccount);
97         $psUpdatePrivateAccount->execute(array(':update_private_account' => $userSetPrivateTo,
':id_user' => $this->idUser));
98     } catch (PDOException $e) {
99         print "Erreur !: " . $e->getMessage() . "<br>";
100         die();
101     }
102 }

```

Here is the call graph for this function:



The documentation for this class was generated from the following file:

- [models/user.php](#)

4.17 UserData Class Reference

Public Member Functions

- [__construct](#) ()
- [getUsersByUsername](#) (\$username)
- [getUserData](#) (\$idUser)

4.17.1 Detailed Description

Definition at line 10 of file userdata.php.

4.17.2 Constructor & Destructor Documentation

4.17.2.1 __construct()

`__construct ()`

default constructor

Definition at line 21 of file userdata.php.

```

22     {
23         if ($this->dbh == null) {
24             try {
25                 $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
26                     PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
27                     PDO::ATTR_PERSISTENT => true
28                 ));
29             }
30             // get list of user by username
31             $sqlRequestUsers = "SELECT * FROM user WHERE username LIKE :search_username AND
privateAccount = 0";
32             $this->psGetUsersByUsername = $this->dbh->prepare($sqlRequestUsers);
33             $this->psGetUsersByUsername->setFetchMode(PDO::FETCH_ASSOC);
34
35             // get list of user by username
36             $sqlGetDataUser = "SELECT * FROM user WHERE id =:search_idUser";
37             $this->psGetDataUser = $this->dbh->prepare($sqlGetDataUser);
38             $this->psGetDataUser->setFetchMode(PDO::FETCH_ASSOC);
39             } catch (PDOException $e) {
40                 print "Erreur !: " . $e->getMessage() . "<br>";
41                 die();
42             }
43         }
44     }

```

4.17.3 Member Function Documentation

4.17.3.1 getUserData()

`getUserData (`
 \$idUser `)`

get data of a specific user

Parameters

int	<i>\$iduser</i>	
-----	-----------------	--

Returns

array

Definition at line 70 of file userdata.php.

```
71     {
72         try {
73             $this->psGetDataUser->execute(array(':search_idUser' => $iduser));
74             $result = $this->psGetDataUser->fetchAll();
75         } catch (PDOException $e) {
76             print "Erreur !: " . $e->getMessage() . "<br>";
77             die();
78         }
79         return $result;
80     }
```

4.17.3.2 getUsersByUsername()

```
getUsersByUsername (
    $username )
```

get users by their username

Parameters

string	<i>\$username</i>	
--------	-------------------	--

Returns

array of game

Definition at line 52 of file userdata.php.

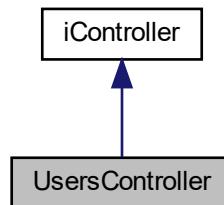
```
53     {
54         try {
55             $this->psGetUsersByUsername->execute(array(':search_username' => ':' . $username . ':'));
56             $result = $this->psGetUsersByUsername->fetchAll();
57         } catch (PDOException $e) {
58             print "Erreur !: " . $e->getMessage() . "<br>";
59             die();
60         }
61         return $result;
62     }
```

The documentation for this class was generated from the following file:

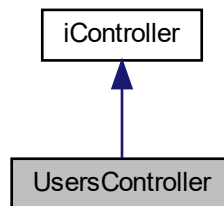
- [models/userdata.php](#)

4.18 UsersController Class Reference

Inheritance diagram for UsersController:



Collaboration diagram for UsersController:



Public Member Functions

- [formHandler\(\)](#)
- [__construct\(\)](#)
- [printHTML\(\)](#)
- [htmlrecherchUsers\(\)](#)
- [htmlrequestUser\(\)](#)
- [htmlDetailUser\(\)](#)

Data Fields

- **\$userData**

4.18.1 Detailed Description

Definition at line 11 of file `usersController.php`.

4.18.2 Constructor & Destructor Documentation

4.18.2.1 `__construct()`

`__construct ()`

default constructor

Definition at line 52 of file `usersController.php`.

```
53     {  
54         $this->userData = new UserData();  
55         $this->game = new Games();  
56     }
```

4.18.3 Member Function Documentation

4.18.3.1 `formHandler()`

`formHandler ()`

used to handle if the user has request something

Returns

void

Implements [iController](#).

Definition at line 24 of file `usersController.php`.

```
25     {  
26         $_SESSION['title'] = "Caiman: Users";  
27  
28         if (isset($_GET['e'])) {  
29             $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_SPECIAL_CHARS);  
30         }  
31  
32         // request user by their username  
33         if ($this->e == "researchUser") {  
34             $_SESSION['title'] = "Caiman: Search";  
35             if (isset($_POST['username'])) {  
36                 $this->requestUsername = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_STRING);  
37             }  
38         }  
39  
40         // show detail of a user  
41         if ($this->e == "detailUser") {  
42             $_SESSION['title'] = "Caiman: User detail";  
43             if (isset($_GET['idUser'])) {  
44                 $this->idUser = filter_input(INPUT_GET, 'idUser', FILTER_SANITIZE_STRING);  
45             }  
46         }  
47     }
```


4.18.3.2 htmlDetailUser()

```
htmlDetailUser ( )
```

create a page with the details of a user

Returns

html

Definition at line 151 of file `usersController.php`.

```
152     {
153         $html = "";
154
155         $html .= '<div class="card " style="background-color: #0d1117;">
156         <div class="card-body container DarkJumbotron">
157             <h2 class="card-title ">Results</h2>
158             <div class="list-group">
159                 ';
160
161         foreach ($this->userData->getUsersByUsername($this->requestUsername) as $key => $user) {
162
163             $html .= '<a class="btn btn-outline-success btnCategorie margintop10 "
164             href="?r=users&e=detailUser&idUser=' . $user['id'] . '" role="button">' . $user['username'] . '</a>';
165         }
166
167         $html .= '
168         </div>
169         </div>';
170
171         return $html;
172     }
```

Here is the caller graph for this function:



4.18.3.3 htmlrecherchUsers()

```
htmlrecherchUsers ( )
```

create the form to search users

Returns

html

Definition at line 95 of file usersController.php.

```

96     {
97         $html = "";
98
99         $html .= '<div class="card  " style="background-color: #0d1117;">
100         <div class="card-body container DarkJumbotron">
101             <h2 class="card-title ">Research</h2>
102
103             <form class="row g-3" action="?r=users&e=researchUser" method="post">
104
105                 <div class="col-auto">
106                     <input type="text" class="form-control" id="username" name="username"
placeholder="username">
107                 </div>
108                 <div class="col-auto">
109                     <button type="submit" class="btn btn-success mb-3">Research</button>
110                 </div>
111             </form>
112         </div>
113     </div>';
114
115     return $html;
116 }

```

Here is the caller graph for this function:

**4.18.3.4 htmlrequestUser()**

htmlrequestUser ()

create the list of user requested by their username

Returns

html

Definition at line 123 of file usersController.php.

```

124     {
125         $html = "";
126
127         $html .= '<div class="card  " style="background-color: #0d1117;">
128         <div class="card-body container DarkJumbotron">
129             <h2 class="card-title ">Results</h2>
130             <div class="list-group">
131                 '
132
133             foreach ($this->userData->getUsersByUsername($this->requestUsername) as $key => $user) {
134
135                 $html .= '<a class="btn btn-outline-success btnCategorie margintop10 "
href="?r=users&e=detailUser&idUser=' . $user['id'] . '" role="button">' . $user['username'] . '</a>';
136             }

```

```

137
138     $html .= '
139         </div>
140     </div>
141 </div>';
142
143     return $html;
144 }

```

4.18.3.5 printHTML()

```
printHTML ( )
```

print the html for the requested content

Returns

void

Implements [iController](#).

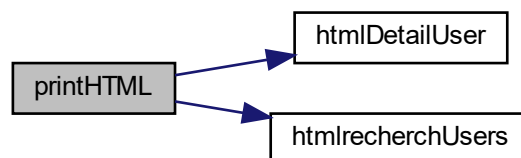
Definition at line 64 of file `usersController.php`.

```

65     {
66
67         $html = '<main style="margin-top:20px ">
68         <div class="container-md">';
69
70         if ($this->e == null) {
71             $html .= $this->htmlrecherchUsers();
72         }
73
74         if ($this->e == "researchUser") {
75             $html .= $this->htmlrecherchUsers();
76             $html .= $this->htmlDetailUser();
77         }
78
79         if ($this->e == "detailUser") {
80             $html .= $this->htmlDataUser();
81             $html .= $this->htmlFavoriteGameUser();
82             $html .= $this->htmlTimeInGameUser();
83         }
84
85         $html .= "</div></main> ";
86
87         echo $html;
88     }

```

Here is the call graph for this function:



The documentation for this class was generated from the following file:

- [controllers/usersController.php](#)

Chapter 5

File Documentation

5.1 common/head.php File Reference

head of file

5.1.1 Detailed Description

head of file

5.1.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.2 common/navbar.php File Reference

Navbar html.

5.2.1 Detailed Description

Navbar html.

5.2.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.3 controllers/administratorController.php File Reference

Class used to handle request for the administrator page.

Data Structures

- class [AdministratorController](#)

5.3.1 Detailed Description

Class used to handle request for the administrator page.

5.3.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.4 controllers/controllers.php File Reference

file used to include all the controller of the project

5.4.1 Detailed Description

file used to include all the controller of the project

5.4.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.5 controllers/dashboardController.php File Reference

Class used to handle request for the dashboard of the user.

Data Structures

- class [DashboardController](#)

5.5.1 Detailed Description

Class used to handle request for the dashboard of the user.

5.5.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.6 controllers/downloadController.php File Reference

Class used to handle the page of download.

Data Structures

- class [DownloadController](#)

5.6.1 Detailed Description

Class used to handle the page of download.

5.6.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.7 controllers/gamesController.php File Reference

Class used to handle request for the games pages.

Data Structures

- class [GamesController](#)

5.7.1 Detailed Description

Class used to handle request for the games pages.

5.7.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.8 controllers/indexController.php File Reference

Class used to handle request for the index.

Data Structures

- class [IndexController](#)

5.8.1 Detailed Description

Class used to handle request for the index.

5.8.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.9 controllers/interfaceController.php File Reference

interface used to implement function to display the html and the handle the requested content

Data Structures

- interface [iController](#)

5.9.1 Detailed Description

interface used to implement function to display the html and the handle the requested content

5.9.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.10 controllers/loginController.php File Reference

Class used to handle request for the login page.

Data Structures

- class [LoginController](#)

5.10.1 Detailed Description

Class used to handle request for the login page.

5.10.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.11 controllers/mainController.php File Reference

main class of the controller used to implement basic function

Data Structures

- class [MainController](#)

5.11.1 Detailed Description

main class of the controller used to implement basic function

5.11.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.12 controllers/signinController.php File Reference

Class used to handle request to create an account.

Data Structures

- class [SigninController](#)

5.12.1 Detailed Description

Class used to handle request to create an account.

5.12.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.13 controllers/usersController.php File Reference

Class used to handle request of the user of the website.

Data Structures

- class [UsersController](#)

5.13.1 Detailed Description

Class used to handle request of the user of the website.

5.13.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.14 index.php File Reference

index of the website

Variables

- `if(!isset($_SESSION['user'])) $r_page = filter_input(INPUT_GET, 'r', FILTER_SANITIZE_SPECIAL_CHARS)`

5.14.1 Detailed Description

index of the website

5.14.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.15 models/administrator.php File Reference

Class used to handle request for the administrator.

Data Structures

- class [Administrator](#)

5.15.1 Detailed Description

Class used to handle request for the administrator.

5.15.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.16 models/categorie.php File Reference

Class used to handle request for the table categorie.

Data Structures

- class [Categories](#)

5.16.1 Detailed Description

Class used to handle request for the table categorie.

5.16.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.17 models/class.php File Reference

Class used to handle include all models.

5.17.1 Detailed Description

Class used to handle include all models.

5.17.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.18 models/download.php File Reference

Class used to handle the download of Caiman.

Data Structures

- class [Download](#)

5.18.1 Detailed Description

Class used to handle the download of Caiman.

5.18.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.19 models/games.php File Reference

Class servant a gerer les requetes en lien avec la table game.

Data Structures

- class [Games](#)

5.19.1 Detailed Description

Class servant a gerer les requetes en lien avec la table game.

5.19.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.20 models/login.php File Reference

Class used to connect an user.

Data Structures

- class [Login](#)

5.20.1 Detailed Description

Class used to connect an user.

5.20.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.21 models/signin.php File Reference

Class used to create a new user.

Data Structures

- class [Signin](#)

5.21.1 Detailed Description

Class used to create a new user.

5.21.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.22 models/user.php File Reference

Class use to manage user.

Data Structures

- class [User](#)

5.22.1 Detailed Description

Class use to manage user.

5.22.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

5.23 models/userdata.php File Reference

Class use to manage user data.

Data Structures

- class [UserData](#)

5.23.1 Detailed Description

Class use to manage user data.

5.23.1.1 BDCC

Author

Lorenzo Bauduccio lorenzo.bdcc@eduge.ch

Copyright

Copyright (c) 2021 BDCC

Index

- `__construct`
 - Administrator, [7](#)
 - AdministratorController, [14](#)
 - Categories, [18](#)
 - DashboardController, [22](#)
 - DownloadController, [27](#)
 - Games, [29](#)
 - GamesController, [36](#)
 - LoginController, [49](#)
 - MainController, [50](#)
 - Signin, [52](#)
 - SigninController, [56](#)
 - User, [58](#)
 - UserData, [61](#)
 - UsersController, [64](#)
- `addCategorie`
 - Categories, [19](#)
- `addCategorieToGame`
 - Categories, [19](#)
- `addGame`
 - Administrator, [8](#)
- `addGameToFavoris`
 - Games, [30](#)
- `Administrator`, [7](#)
 - `__construct`, [7](#)
 - `addGame`, [8](#)
 - `getConsoleFolderName`, [9](#)
 - `getListConsole`, [10](#)
 - `updateGame`, [10](#)
 - `uploadGame`, [11](#)
 - `uploadGameImage`, [12](#)
- `AdministratorController`, [13](#)
 - `__construct`, [14](#)
 - `formHandler`, [15](#)
 - `printHTML`, [17](#)
- `allowAccessTo`
 - MainController, [51](#)
- `Categories`, [18](#)
 - `__construct`, [18](#)
 - `addCategorie`, [19](#)
 - `addCategorieToGame`, [19](#)
 - `delCategorieFromGame`, [20](#)
 - `getCategoriesOfGame`, [20](#)
 - `getListAllCategories`, [21](#)
- `checkIfEmailAlreadyTaken`
 - Signin, [53](#)
- `checkIfGamelsAlreadyInFavoris`
 - Games, [31](#)
- `checkIfUsernameAlreadyTaken`
 - Signin, [53](#)
- `checkLogin`
 - Login, [47](#)
- `common/head.php`, [69](#)
- `common/navbar.php`, [69](#)
- `controllers/administratorController.php`, [70](#)
- `controllers/controllers.php`, [70](#)
- `controllers/dashboardController.php`, [71](#)
- `controllers/downloadController.php`, [71](#)
- `controllers/gamesController.php`, [72](#)
- `controllers/indexController.php`, [72](#)
- `controllers/interfaceController.php`, [73](#)
- `controllers/loginController.php`, [73](#)
- `controllers/mainController.php`, [74](#)
- `controllers/signinController.php`, [74](#)
- `controllers/usersController.php`, [75](#)
- `DashboardController`, [21](#)
 - `__construct`, [22](#)
 - `formHandler`, [22](#)
 - `htmlFormUpdatePassword`, [23](#)
 - `printHTML`, [24](#)
- `delCategorieFromGame`
 - Categories, [20](#)
- `Download`, [25](#)
 - `downloadCaiman`, [25](#)
- `downloadCaiman`
 - `Download`, [25](#)
- `DownloadController`, [26](#)
 - `__construct`, [27](#)
 - `formHandler`, [27](#)
 - `printHTML`, [28](#)
- `errorHandler`
 - MainController, [51](#)
- `formHandler`
 - AdministratorController, [15](#)
 - DashboardController, [22](#)
 - DownloadController, [27](#)
 - GamesController, [37](#)
 - IndexController, [46](#)
 - LoginController, [49](#)
 - SigninController, [56](#)
 - UsersController, [64](#)
- `Games`, [29](#)
 - `__construct`, [29](#)
 - `addGameToFavoris`, [30](#)

- checkIfGamelsAlreadyInFavoris, 31
 - getAllGames, 31
 - getFavoriteGamesOfUser, 31
 - getGameDetail, 32
 - getGamesInCategorie, 32
 - getListOfGameWithTimeUser, 33
 - getRequestGames, 33
 - getTimeInGameUser, 34
 - removeGameFromFavoris, 34
- GamesController, 35
 - __construct, 36
 - formHandler, 37
 - getGameDetail, 38
 - getGamesFromCategorie, 39
 - getListAllGames, 39
 - getRequestedGames, 40
 - printHTML, 41
 - recherchFull, 42
 - recherchNotFull, 43
- getAllGames
 - Games, 31
- getCategoriesOfGame
 - Categories, 20
- getConsoleFolderName
 - Administrator, 9
- getFavoriteGamesOfUser
 - Games, 31
- getGameDetail
 - Games, 32
 - GamesController, 38
- getGamesFromCategorie
 - GamesController, 39
- getGamesInCategorie
 - Games, 32
- getListAllCategories
 - Categories, 21
- getListAllGames
 - GamesController, 39
- getListConsole
 - Administrator, 10
- getListOfGameWithTimeUser
 - Games, 33
- getPrivateAccount
 - User, 58
- getRequestedGames
 - GamesController, 40
- getRequestGames
 - Games, 33
- getTimeInGameUser
 - Games, 34
- getUserData
 - UserData, 61
- getUsersByUsername
 - UserData, 62
- htmlDetailUser
 - UsersController, 64
- htmlFormUpdatePassword
 - DashboardController, 23
- htmlrecherchUsers
 - UsersController, 65
- htmlrequestUser
 - UsersController, 66
- iController, 44
- index.php, 75
- IndexController, 45
 - formHandler, 46
 - printHTML, 46
- Login, 47
 - checkLogin, 47
- LoginController, 48
 - __construct, 49
 - formHandler, 49
 - printHTML, 49
- MainController, 50
 - __construct, 50
 - allowAccessTo, 51
 - errorHandler, 51
- models/administrator.php, 76
- models/categorie.php, 76
- models/class.php, 77
- models/download.php, 77
- models/games.php, 78
- models/login.php, 78
- models/signin.php, 79
- models/user.php, 79
- models/userdata.php, 80
- newUser
 - Signin, 53
- printHTML
 - AdministratorController, 17
 - DashboardController, 24
 - DownloadController, 28
 - GamesController, 41
 - IndexController, 46
 - LoginController, 49
 - SigninController, 57
 - UsersController, 67
- recherchFull
 - GamesController, 42
- recherchNotFull
 - GamesController, 43
- removeGameFromFavoris
 - Games, 34
- Signin, 52
 - __construct, 52
 - checkIfEmailAlreadyTaken, 53
 - checkIfUsernameAlreadyTaken, 53
 - newUser, 53
- SigninController, 55
 - __construct, 56
 - formHandler, 56

- printHTML, [57](#)
- updateGame
 - Administrator, [10](#)
- updatePassword
 - User, [59](#)
- updatePrivateAccount
 - User, [60](#)
- uploadGame
 - Administrator, [11](#)
- uploadGameImage
 - Administrator, [12](#)
- User, [57](#)
 - __construct, [58](#)
 - getPrivateAccount, [58](#)
 - updatePassword, [59](#)
 - updatePrivateAccount, [60](#)
- UserData, [61](#)
 - __construct, [61](#)
 - getUserData, [61](#)
 - getUsersByUsername, [62](#)
- UsersController, [63](#)
 - __construct, [64](#)
 - formHandler, [64](#)
 - htmlDetailUser, [64](#)
 - htmlrecherchUsers, [65](#)
 - htmlrequestUser, [66](#)
 - printHTML, [67](#)