Caiman web

Generated by Doxygen 1.9.1

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

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

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

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

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.

```
{
    if ($this->dbh == null) {
        try {
            $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
                PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
                PDO::ATTR_PERSISTENT => true
            ));
            // get list of console
            $sqlGetListConsole = "SELECT * FROM consol";
            $this->psGetListConsole = $this->dbh->prepare($sqlGetListConsole);
            $this->psGetListConsole->setFetchMode(PDO::FETCH_ASSOC);

            // upload game
            $sqlUploadGame = "INSERT INTO game  (name, description, imageName, idConsole, idFile)
            VALUES (:insert_name, :insert_description, :insert_imageName, :insert_idConsole,
:insert_idFile)";
            $this->psUploadGame = $this->dbh->prepare($sqlUploadGame);
            $this->psUploadGame->setFetchMode(PDO::FETCH_ASSOC);

            // upload file
            $sqlUploadFile = "INSERT INTO file  (filename, dateUpdate)
            VALUES (:insert_filename, NOW() )";
            $this->psUploadFile = $this->dbh->prepare($sqlUploadFile);
            $this->psUploadFile->setFetchMode(PDO::FETCH_ASSOC);

            // update game
            $sqlUpdateGane = "UPDATE game  SET name = :update_name, description =
:update_description, idConsole = :update_idConsole WHERE id = :update_id";
            $this->psUpdateGame = $this->dbh->prepare($sqlUpdateGane);
            $this->psUpdateGame->setFetchMode(PDO::FETCH_ASSOC);

            // get folder name of console
            $sqlGetNameConsoleFolder = "SELECT folderName FROM consol WHERE id = :console_id";
            $this->psGetNameConsoleFolder = $this->dbh->prepare($sqlGetNameConsoleFolder);
            $this->psGetNameConsoleFolder->setFetchMode(PDO::FETCH_ASSOC);
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
    }
}
```

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	*$name*	
string	*$description*	
string	*$imageName*	
integer	*$consoleId*	
string	*$gameFileName*	

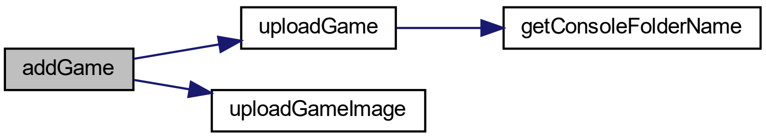
Returns

void

Definition at line 84 of file administrator.php.

```
    {

        if ($this->uploadGame($gameFileName, $consoleId) && $this->uploadGameImage($imageName)) {
            try {
                $this->psUploadFile->execute(array(':insert_filename' => $gameFileName));
            } catch (PDOException $e) {
                print "Erreur !: " . $e->getMessage() . "<br>";
                die();
            }
            $lastInsertId = $this->dbh->lastInsertId();
            try {
                $this->psUploadGame->execute(array(':insert_name' => $name, ':insert_description' =>
    $description, ':insert_imageName' => $imageName, ':insert_idConsole' => $consoleId, ':insert_idFile'
    => $lastInsertId));
            } catch (PDOException $e) {
                print "Erreur !: " . $e->getMessage() . "<br>";
                die();
            }
        }
    }
```

Here is the call graph for this function:



4.1.3.2 getConsoleFolderName()

```
getConsoleFolderName (
                $id )
```

get the path name of an console

Parameters

int	*$id*	

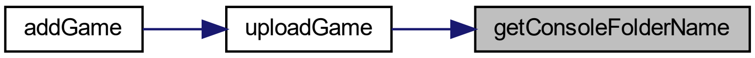
Returns

array of game

Definition at line 202 of file administrator.php.

```
    {
        $returnArray = null;
        try {
            $this->psGetNameConsoleFolder->execute(array(':console_id' => $id));
            $returnArray = $this->psGetNameConsoleFolder->fetchAll();
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $returnArray[0]['folderName'];
    }
```

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.

```
    {
        $returnArray = null;
        try {
            $this->psGetListConsole->execute();
            $returnArray = $this->psGetListConsole->fetchAll();
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $returnArray;
    }
```

4.1.3.4 updateGame()

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

update da of a game

Parameters

int	*$idGame*	
string	*$name*	
string	*$description*	
int	*$consoleId*	

Returns

void

Definition at line 151 of file administrator.php.

```
    {
        try {
            $this->psUpdateGame->execute(array(':update_name' => $name, ':update_description' =>
$description, ':update_idConsole' => $consoleId, ':update_id' => $idGame));
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
    }
```

4.1.3.5 uploadGame()

```
uploadGame (
                $gameFileName,
                $consoleId )
```

upload a game

Parameters

int	*$gameFileName*	
int	*$consoleId*	

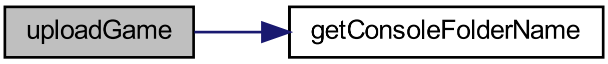
Returns

bool

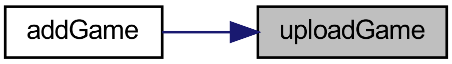
Definition at line 111 of file administrator.php.

```
    {
        $uploadIsValid = false;
        $target_dir = "../games/" . $this->getConsoleFolderName($consoleId) . "/";

        $target_file =  basename($_FILES["fileGame"]["name"]);
        $uploadOk = 1;
        $fileType = strtolower(pathinfo($target_file, PATHINFO_EXTENSION));

        //rename file
        $newfilename = $gameFileName . '.' . $fileType;


        // Check if file already exists
        if (file_exists($target_file)) {
            echo "Sorry, file already exists.";
            $uploadOk = 0;
        }
        if ($uploadOk == 0) {
            echo "Sorry, your file was not uploaded.";
            // if everything is ok, try to upload file
        } else {
            if (move_uploaded_file($_FILES["fileGame"]["tmp_name"], $target_dir . $newfilename)) {
                $uploadIsValid = true;
            } else {
                //Sorry, there was an error uploading your file
            }
        }
        return $uploadIsValid;
    }
```

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	*$imageFileName*	

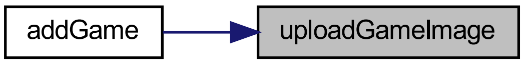
Returns

bool

Definition at line 167 of file administrator.php.

```
    {
        $uploadIsValid = false;
        $target_dir = "img/games/";

        $target_file =  basename($_FILES["image"]["name"]);
        $uploadOk = 1;

        //rename file
        $newfilename = $imageFileName;

        // Check if file already exists
        if (file_exists($target_file)) {
            echo "Sorry, file already exists.";
            $uploadOk = 0;
        }
        if ($uploadOk == 0) {
            echo "Sorry, your file was not uploaded.";
            // if everything is ok, try to upload file
        } else {
            if (move_uploaded_file($_FILES["image"]["tmp_name"], $target_dir . $newfilename)) {
                $uploadIsValid = true;
            } else {
                //Sorry, there was an error uploading your file
            }
        }
        return $uploadIsValid;
    }
```

Here is the caller graph for this function:

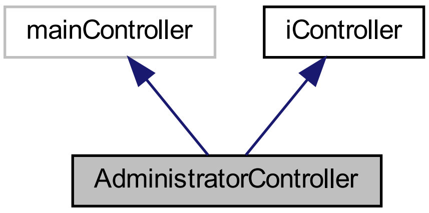


The documentation for this class was generated from the following file:

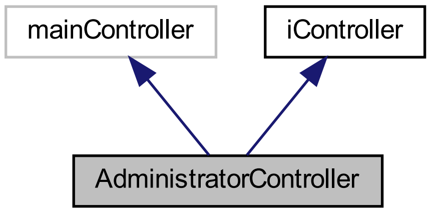
- models/administrator.php

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.

```
    {
        $this->administrator  = new Administrator();
        $this->game = new Games();
        $this->categorie = new Categories();
    }
```

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.

```
{

  if (isset($_GET['e'])) {
    $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_STRING);
    //redirige l'utilisateur qui n'a pas les bon droits
    $this->allowAccessTo(array(1));
  }
  // update game
  if ($this->e == "updateGame") {
    if (isset($_GET['id'])) {
      $requestGame = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_STRING);
      $this->idGameToUpdate = $requestGame;
    } else {
      header('Location:' . $_SERVER['HTTP_REFERER']);
      exit;
    }
  }

  // add game categorie
  if ($this->e == "addGameCategorie") {
    if (isset($_GET['id'])) {
      $requestGame = filter_input(INPUT_GET, 'id', FILTER_SANITIZE_STRING);
      $this->idGameToUpdate = $requestGame;
    } else {
      header('Location:' . $_SERVER['HTTP_REFERER']);
      exit;
    }
  }

  // add categorie to game
  if ($this->e == "addGameCategorieAdd") {
    if (isset($_GET['idGame'])) {
      $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_STRING);
    } else {
      header('Location:' . $_SERVER['HTTP_REFERER']);
      exit;
    }
    if (isset($_GET['idCategorie'])) {
      $idCategorie = filter_input(INPUT_GET, 'idCategorie', FILTER_SANITIZE_STRING);
    } else {
      header('Location:' . $_SERVER['HTTP_REFERER']);
      exit;
    }

    $this->categorie->addCategorieToGame($idGame, $idCategorie);
    header('Location:' . $_SERVER['HTTP_REFERER']);
    exit;
  }

  // delete categorie from a game
  if ($this->e == "delGameCategorie") {
    if (isset($_GET['idGame'])) {
      $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_STRING);
    } else {
      header('Location:' . $_SERVER['HTTP_REFERER']);
      exit;
    }
    if (isset($_GET['idCategorie'])) {
      $idCategorie = filter_input(INPUT_GET, 'idCategorie', FILTER_SANITIZE_STRING);
    } else {
      header('Location:' . $_SERVER['HTTP_REFERER']);
      exit;
    }

    $this->categorie->delCategorieFromGame($idGame, $idCategorie);
    header('Location:' . $_SERVER['HTTP_REFERER']);
    exit;
  }

  //add game
  if ($this->e == "addGameUpload") {
    if (isset($_POST['name'])) {
      $name = filter_input(INPUT_POST, 'name', FILTER_SANITIZE_STRING);
    } else {
      header('Location:' . $_SERVER['HTTP_REFERER']);
      exit;
     }

     if (isset($_POST['description'])) {
       $description = filter_input(INPUT_POST, 'description', FILTER_SANITIZE_STRING);
     } else {
       header('Location:' . $_SERVER['HTTP_REFERER']);
       exit;
     }

     if (isset($_POST['imageName'])) {
       $imageName = filter_input(INPUT_POST, 'imageName', FILTER_SANITIZE_STRING);
     } else {
       header('Location:' . $_SERVER['HTTP_REFERER']);
       exit;
     }

     if (isset($_POST['console'])) {
       $consoleId = filter_input(INPUT_POST, 'console', FILTER_SANITIZE_STRING);
     } else {
       header('Location:' . $_SERVER['HTTP_REFERER']);
       exit;
     }

     if (isset($_POST['gameFileName'])) {
       $gameFileName = filter_input(INPUT_POST, 'gameFileName', FILTER_SANITIZE_STRING);
     } else {
       header('Location:' . $_SERVER['HTTP_REFERER']);
       exit;
     }


     $this->administrator->addGame($name, $description, $imageName, $consoleId, $gameFileName);

     header('Location:' . $_SERVER['HTTP_REFERER']);
     exit;
   }

   //add game
   if ($this->e == "updateGameUpdate") {
     if (isset($_POST['name'])) {
       $name = filter_input(INPUT_POST, 'name', FILTER_SANITIZE_STRING);
     } else {
       header('Location:' . $_SERVER['HTTP_REFERER']);
       exit;
     }

     if (isset($_POST['description'])) {
       $description = filter_input(INPUT_POST, 'description', FILTER_SANITIZE_STRING);
     } else {
       header('Location:' . $_SERVER['HTTP_REFERER']);
       exit;
     }

     if (isset($_POST['console'])) {
       $consoleId = filter_input(INPUT_POST, 'console', FILTER_SANITIZE_STRING);
     } else {
       header('Location:' . $_SERVER['HTTP_REFERER']);
       exit;
     }

     if (isset($_POST['idGame'])) {
       $idGame = filter_input(INPUT_POST, 'idGame', FILTER_SANITIZE_STRING);
     } else {
       header('Location:' . $_SERVER['HTTP_REFERER']);
       exit;
     }


     $this->administrator->updateGame($idGame, $name, $description, $consoleId);

     header('Location:' . $_SERVER['HTTP_REFERER']);
     exit;
   }


   if ($this->e == "addCategorie") {
     if (isset($_POST['name'])) {
       $name = filter_input(INPUT_POST, 'name', FILTER_SANITIZE_STRING);
     }

     if (isset($name)) {
       $this->categorie->addCategorie($name);
     }
   }
 }
```

4.2.3.2 printHTML()

```
printHTML ( )
```

print the html for the resquested content

Returns

html

Implements iController.

Definition at line 203 of file administratorController.php.

```
  {

    $html = '<main style="margin-top:20px">
        <div class="container-md">';
    $html .= $this->errorHandler();
    if ($this->e == null) {
      $html .= $this->htmlAdministratorHome();
    }

    if ($this->e == "addGame") {
      $html .= $this->htmlNewGame();
    }

    if ($this->e == "updateGame") {
      $html .= $this->htmlUpdateGame();
    }

    if ($this->e == "addCategorie") {
      $html .= $this->htmlAddCategorie();
    }

    if ($this->e == "addGameCategorie") {
      $html .= $this->htmlAddCategorieToGame();
    }



    $html .= "</div></main> ";
    echo $html;
  }
```

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.

```
{
    if ($this->dbh == null) {
        try {
            $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
                PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
                PDO::ATTR_PERSISTENT => true
            ));
            //get all categories
            $sqlGetAllCategories = "SELECT * FROM categorie";
            $this->psGetAllCategories = $this->dbh->prepare($sqlGetAllCategories);
            $this->psGetAllCategories->setFetchMode(PDO::FETCH_ASSOC);

            //add categorie
            $sqlAddCategorie = "INSERT INTO categorie (name) VALUES (:categorie_name)";
            $this->psAddCategorie = $this->dbh->prepare($sqlAddCategorie);
            $this->psAddCategorie->setFetchMode(PDO::FETCH_ASSOC);

            //add categorie to game
            $sqlAddCategorieToGame = "INSERT INTO gamehascategorie (idGame,idCategorie) VALUES
(:insert_idGame, :insert_idCategorie)";
            $this->psAddCategorieToGame = $this->dbh->prepare($sqlAddCategorieToGame);

            //check if game has a specific categorie
            $sqlCheckIfGameHasCategorie = "SELECT * FROM gamehascategorie WHERE idCategorie =
:insert_idCategorie AND idGame = :insert_idGame";
            $this->psCheckIfGameHasCategorie = $this->dbh->prepare($sqlCheckIfGameHasCategorie);
            $this->psCheckIfGameHasCategorie->setFetchMode(PDO::FETCH_ASSOC);

            //del categorie from game
            $sqlDelCategorieFromGame = "DELETE FROM gamehascategorie  WHERE idCategorie =
:del_idCategorie AND idGame = :del_idGame";
            $this->psDelCategorieFromGame = $this->dbh->prepare($sqlDelCategorieFromGame);

            //get categories of a game
            $sqlGameCategorie = "SELECT c.name, c.id FROM `gamehascategorie` as ghc
            LEFT JOIN categorie as c
            ON ghc.idCategorie = c.id
            LEFT JOIN game as g
            ON ghc.idGame = g.id
            WHERE idGame = :search_id";
            $this->psGameCategorie = $this->dbh->prepare($sqlGameCategorie);
            $this->psGameCategorie->setFetchMode(PDO::FETCH_ASSOC);
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
    }
}
```

4.3.3 Member Function Documentation

4.3.3.1 addCategorie()

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

add a new categorie in the database

Parameters

string	*$categorieName*	

Returns

html

Definition at line 120 of file categorie.php.

```
    {
        try {
            $this->psAddCategorie->execute(array(':categorie_name' => $categorieName));
            $result = $this->psAddCategorie->fetchAll();
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

4.3.3.2 addCategorieToGame()

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

add a categorie to a game

Parameters

integer	*$idGame*	
integer	*$idCategorie*	

Returns

void

Definition at line 139 of file categorie.php.

```
    {
        $result = null;
        try {
            $this->psCheckIfGameHasCategorie->execute(array(':insert_idCategorie' => $idCategorie,
     ':insert_idGame' => $idGame));
            $result = $this->psCheckIfGameHasCategorie->fetchAll();
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        if ($result == null) {
            try {
                $this->psAddCategorieToGame->execute(array(':insert_idCategorie' => $idCategorie,
     ':insert_idGame' => $idGame));
            } catch (PDOException $e) {
                print "Erreur !: " . $e->getMessage() . "<br>";
                die();
            }
        }
    }
```

4.3.3.3 delCategorieFromGame()

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

delete a cotegorie of a game

Parameters

integer	*$idGame*	
integer	*$idCategorie*	

Returns

void

Definition at line 165 of file categorie.php.

```
    {
        try {
            $this->psDelCategorieFromGame->execute(array(':del_idCategorie' => $idCategorie,
     ':del_idGame' => $idGame));
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
    }
```

4.3.3.4 getCategoriesOfGame()

```
getCategoriesOfGame (
              int $idGame )
```

get the categories of a game

Parameters

integer	*$idGame*	

Returns

list of categories

Definition at line 102 of file categorie.php.

```
    {
        try {
            $this->psGameCategorie->execute(array(':search_id' => $idGame));
            $result = $this->psGameCategorie->fetchAll();
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

4.3.3.5 getListAllCategories()

```
getListAllCategories ( )
```

returns list of all categories

Returns

array with all list

Definition at line 83 of file categorie.php.

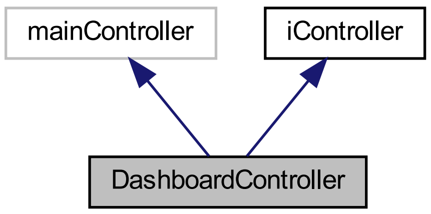
```
    {

        try {
            $this->psGetAllCategories->execute();
            $result = $this->psGetAllCategories->fetchAll();
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

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

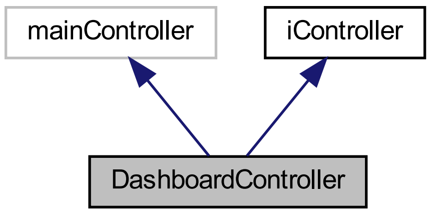
- models/categorie.php

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.

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

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.

```
    {
        $_SESSION['title'] = "Caiman: Dashboard";
        $this->allowAccessTo(array(1, 3));

        $oldPassword = null;
        $newPasswordRepeat = null;
        $newPassword = null;

        if (isset($_GET['e'])) {
            $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_SPECIAL_CHARS);
        }
        // form update
        if ($this->e == "updatePassword") {
            $_SESSION['title'] = "Caiman: Update password";
            if (isset($_POST['oldPassword'])) {
                $oldPassword = filter_input(INPUT_POST, 'oldPassword', FILTER_SANITIZE_STRING);
            }
            if (isset($_POST['newPassword'])) {
                $newPassword = filter_input(INPUT_POST, 'newPassword', FILTER_SANITIZE_STRING);
            }
            if (isset($_POST['newPasswordRepeat'])) {
                $newPasswordRepeat = filter_input(INPUT_POST, 'newPasswordRepeat',
    FILTER_SANITIZE_STRING);
            }

            if (isset($oldPassword) && isset($newPassword) && isset($newPasswordRepeat)) {
                $_SESSION['user']->updatePassword($newPassword, $newPasswordRepeat, $oldPassword);
            }
        }

        // update if account if visible or not
        if ($this->e == "updatePrivateAccount") {

            if ($_SESSION['user']->idUser != -1) {
                $_SESSION['user']->updatePrivateAccount();
                header('Location:' . $_SERVER['HTTP_REFERER']);
            }
        }
    }
```

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.
```
    {
        $html = '<div class="d-inline-flex p-2 jumbotron  width100 DarkJumbotron "
style="background-color: #161b22;" >
        <div class="container">
        <div class="row"><h2>Update your password</h2></div>
        <div class="row">


        <form action="?r=dashboard&e=updatePassword" method="post">
            <div class="form-group">
                <label for="oldPassword">Old password</label>
                <input type="password" class="form-control" id="oldPassword" name="oldPassword"
placeholder="Old password">
            </div>
            <div class="form-group">
                <label for="newPassword">Password</label>
                <input type="password" class="form-control" id="newPassword" name="newPassword"
placeholder="New password">
            </div>
            <div class="form-group">
                <label for="newPasswordRepeat">Password</label>
                <input type="password" class="form-control" id="newPasswordRepeat"
name="newPasswordRepeat" placeholder="New password repeat">
            </div>

            <button type="submit" class="btn btn-primary">Submit</button>
        </form>
        </div>
                </div>
                </div>';
        return $html;
    }
```

Here is the caller graph for this function:

printHTML
htmlFormUpdatePassword

4.4.3.3 printHTML()

```
printHTML ( )
```

print the html for the resquested content

Returns

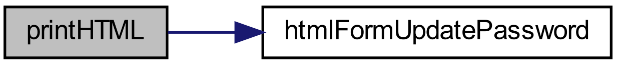
html

Implements iController.

Definition at line 75 of file dashboardController.php.

```
{

    $html = '<main  style="margin-top:20px ">
    <div class="container-md">';
    echo $_SESSION['error'];
    $html .= $this->errorHandler();

    if ($this->e == null) {
        $html .= $this->htmlFormHead();
        $html .= $this->htmlFavoriteGames();
        $html .= $this->htmlGameTime();
    }

    if ($this->e == "updatePassword") {
        $html .= $this->htmlFormUpdatePassword();
    }

    $html .= "</div></main> ";

    echo $html;
}
```

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.

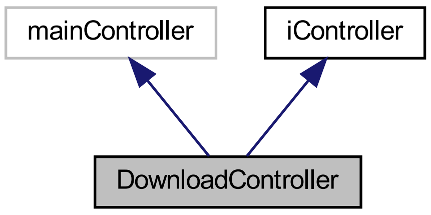
```
    {
        $filename = '../release/caiman.jpg'; // of course find the exact filename....
        header('Pragma: public');
        header('Expires: 0');
        header('Cache-Control: must-revalidate, post-check=0, pre-check=0');
        header('Cache-Control: private', false); // required for certain browsers
        header('Content-Type: application/jpg');

        header('Content-Disposition: attachment; filename="' . basename($filename) . '";');
        header('Content-Transfer-Encoding: binary');
        header('Content-Length: ' . filesize($filename));

        readfile($filename);

        exit;
    }
```

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

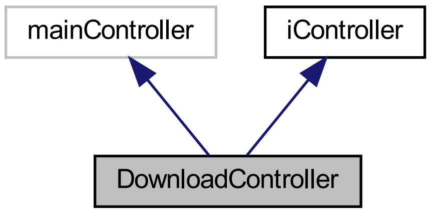
- models/download.php

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.

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

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.

```
{
  $_SESSION['title'] = "Caiman: Download";
  if (isset($_GET['e'])) {
    $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_STRING);
  }

  if ($this->e == null) {

    if ($_SESSION['user']->idUser != -1) {
      $this->e = "user";
    } else {
      $this->e = "visitor";
    }
  }

  if ($this->e == "download") {

    if ($_SESSION['user']->idUser != -1) {
      $this->download->downloadCaiman();
    } else {
      header('?r=login');
    }
  }
}
```

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.

```
{
  $html = '<main style="margin-top:20px">
      <div class="container-md">';
  $html .= $this->errorHandler();
  if ($this->e == "user") {
    $html .= $this->htmlUserDownload();
  }

  if ($this->e == "visitor") {
    $html .= $this->htmlVisitorDownload();
  }
  $html .= "</div></main> ";
  echo $html;
}
```

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)
- checkIfGameIsAlreadyInFavoris (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.

```
    {
        if ($this->dbh == null) {
            try {
                $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
                    PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
                    PDO::ATTR_PERSISTENT => true
                ));
                //get all games
                $sqlGetAllGames = "SELECT * FROM game";
                $this->psGetAllGames = $this->dbh->prepare($sqlGetAllGames);
                $this->psGetAllGames->setFetchMode(PDO::FETCH_ASSOC);

                //get request games
                $sqlRequestGames = "SELECT * FROM game WHERE name LIKE :search_game";
                $this->psRequestGames = $this->dbh->prepare($sqlRequestGames);
                $this->psRequestGames->setFetchMode(PDO::FETCH_ASSOC);

                //get Time in game user
                $sqlTimeInGame = "SELECT * FROM timeingame WHERE idGame = :search_idGame AND idUser =
       :search_idUser ";
                $this->psGetTimeInGame = $this->dbh->prepare($sqlTimeInGame);
                $this->psGetTimeInGame->setFetchMode(PDO::FETCH_ASSOC);

                //get game with time user
                $sqlGetGameWithTime = "SELECT * FROM timeingame WHERE idUser = :search_idUser ORDER BY
       timeInMinute DESC";
                $this->psGetGameWithTime = $this->dbh->prepare($sqlGetGameWithTime);
                $this->psGetGameWithTime->setFetchMode(PDO::FETCH_ASSOC);

                //get detail game
```

```
           $sqlGameDetail = "SELECT * FROM game WHERE id = :search_id";
           $this->psGameDetail = $this->dbh->prepare($sqlGameDetail);
           $this->psGameDetail->setFetchMode(PDO::FETCH_ASSOC);

           //add game to favoris
           $sqlAddGameToFavoris = "INSERT INTO favoritegame  (idGame, idUser)
           VALUES (:search_idGame, :search_idUser)";
           $this->psAddGameToFavori = $this->dbh->prepare($sqlAddGameToFavoris);
           $this->psAddGameToFavori->setFetchMode(PDO::FETCH_ASSOC);

           //remove game to favoris
           $sqlRemoveGameFormFavoris = "DELETE FROM favoritegame
           WHERE idUser = :search_idUser AND idGame = :search_idGame";
           $this->psRemoveGameFromFavori = $this->dbh->prepare($sqlRemoveGameFormFavoris);
           $this->psRemoveGameFromFavori->setFetchMode(PDO::FETCH_ASSOC);

           //check if already in favoris
           $sqlCheckIfAlreadyFavoris = "SELECT * FROM favoritegame
           WHERE iduser = :search_idUser AND idGame = :search_idGame";
           $this->psCheckIfFavoris = $this->dbh->prepare($sqlCheckIfAlreadyFavoris);
           $this->psCheckIfFavoris->setFetchMode(PDO::FETCH_ASSOC);

           //get favorite game of user
           $sqlFavoriteGameOfUser = "SELECT g.name, g.id, g.imageName FROM `favoritegame` as fg
           LEFT JOIN game as g
           ON fg.idGame = g.id
           LEFT JOIN user as u
           ON fg.iduser = u.id
           WHERE iduser = :search_id";
           $this->psFavoriteGameOfUser = $this->dbh->prepare($sqlFavoriteGameOfUser);
           $this->psFavoriteGameOfUser->setFetchMode(PDO::FETCH_ASSOC);



            //get list of games in a categorie
            $sqlGameInCategorie = "SELECT g.name, g.id, g.imageName FROM `gamehascategorie` as ghc
            LEFT JOIN game as g
            ON ghc.idGame = g.id
            LEFT JOIN categorie as c
            ON ghc.idCategorie = c.id
            WHERE idCategorie = :search_id";
            $this->psGameInCategorie = $this->dbh->prepare($sqlGameInCategorie);
            $this->psGameInCategorie->setFetchMode(PDO::FETCH_ASSOC);

        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
    }
}
```

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.

```
{

    try{
        $this->psAddGameToFavori->execute(array(':search_idUser' => $idUser,':search_idGame' =>
  $idGame));

    }catch (PDOException $e) {
        print "Erreur !: " . $e->getMessage() . "<br>";
        die();
    }
}
```

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.

```
    {
        $boolResult = true;

        try{
            $this->psCheckIfFavoris->execute(array(':search_idUser' => $idUser,':search_idGame' =>
      $idGame));
            $result = $this->psCheckIfFavoris->fetchAll();
            if ($result != null) {
                $boolResult = false;
            }

        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $boolResult;
    }
```

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.

```
    {

        try{
            $this->psGetAllGames->execute();
            $result = $this->psGetAllGames->fetchAll();


        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

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.

```
    {
        try{
            $this->psFavoriteGameOfUser->execute(array(':search_id' => $idUser));
            $result = $this->psFavoriteGameOfUser->fetchAll();


        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

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.

```
    {

        try{
            $this->psGameDetail->execute(array(':search_id' => $idGame));
            $result = $this->psGameDetail->fetchAll();


        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

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.

```
    {
        try{
            $this->psGameInCategorie->execute(array(':search_id' => $idCategorie));
            $result = $this->psGameInCategorie->fetchAll();
        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

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.

```
    {
        try{
            $this->psGetGameWithTime->execute(array(':search_idUser' => $idUser));
            $result = $this->psGetGameWithTime->fetchAll();


        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

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.

```
    {

        try{
            $this->psRequestGames->execute(array(':search_game' => '%'.$gameName.'%'));
            $result = $this->psRequestGames->fetchAll();


        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

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.

```
    {
        try{
            $this->psGetTimeInGame->execute(array(':search_idGame' => $idGame, ':search_idUser' =>
       $idUser));
            $result = $this->psGetTimeInGame->fetchAll();
        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

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.

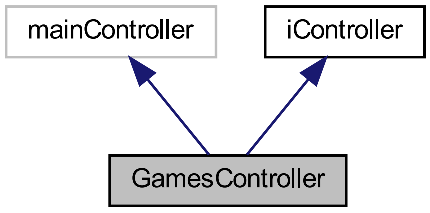
```
    {

        try{
            $this->psRemoveGameFromFavori->execute(array(':search_idUser' => $idUser,':search_idGame' =>
    $idGame));


        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
    }
```

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

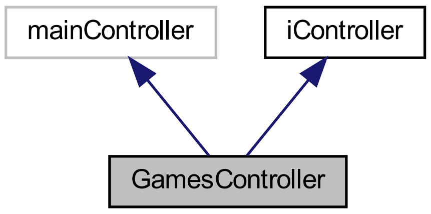
- models/games.php

4.8 GamesController Class Reference

Inheritance diagram for GamesController:



Collaboration diagram for GamesController:



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.

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

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.

```
    {
        $_SESSION['title'] = "Caiman: Games";
        $requestGame = null;
        $result = null;
        if (isset($_GET['e'])) {
            $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_STRING);
        }

        if ($this->e == "requestGame") {

            if (isset($_POST['gameName'])) {
                $requestGame = filter_input(INPUT_POST, 'gameName', FILTER_SANITIZE_STRING);
                $_SESSION['title'] = "Caiman: Search " . $requestGame;
            }

            if (isset($requestGame)) {
                $result = $this->games->getRequestGames($requestGame);
            }
        }

        if ($this->e == "detail") {

            if (isset($_GET['idGame'])) {
                $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_STRING);
                $this->idGame = $idGame;
            }

            if (isset($idGame)) {
                $result = $this->games->getGameDetail($idGame);
                $_SESSION['title'] = "Caiman: " . $result[0]["name"];
            }
        }

        if ($this->e == "categorie") {
            $_SESSION['title'] = "Caiman: Categorie";
            if (isset($_GET['idCategorie'])) {
                $idcategory = filter_input(INPUT_GET, 'idCategorie', FILTER_SANITIZE_STRING);
                $this->idcategory = $idcategory;
            }

            if (isset($idcategory)) {
                $result = $this->games->getGamesInCategorie($idcategory);
            }
        }


        if ($this->e == "addFavoris") {
            $_SESSION['title'] = "Caiman: Favorite";
            if (isset($_GET['idGame'])) {
                $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_NUMBER_INT);
            }

            if (isset($idGame)) {
                $result = $this->games->addGameToFavoris($_SESSION['user']->idUser, $idGame);
                header('Location:' . $_SERVER['HTTP_REFERER']);
                $_SESSION['error'] = "Favorite added";
            }
        }

        if ($this->e == "removeFavoris") {
            if (isset($_GET['idGame'])) {
                $idGame = filter_input(INPUT_GET, 'idGame', FILTER_SANITIZE_NUMBER_INT);
            }

            if (isset($idGame)) {
                $result = $this->games->removeGameFromFavoris($_SESSION['user']->idUser, $idGame);
                header('Location:' . $_SERVER['HTTP_REFERER']);
                $_SESSION['error'] = "Favorite removed";
            }
        }

        $this->requestedgame = $result;
    }
```

4.8.3.2 getGameDetail()

```
getGameDetail ( )
```

create the page of a specific game

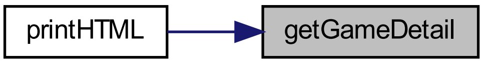
Returns

html

Definition at line 190 of file gamesController.php.

```
    {

        $gameDetail = $this->games->getGameDetail($this->idGame);
        $category = $this->categorie->getCategoriesOfGame($this->idGame);
        $html = '';

        $html .= '</br>
            <div class="row">
                <div class="col">
                    <img class="detailImage" src="./img/games/' . $gameDetail[0]['imageName'] . '."  >
                </div>
                <div class="col">
                    <h2 class="card-title">' . $gameDetail[0]['name'] . '</h2>
                    <p class="card-title">' . $gameDetail[0]['description'] . '</p>
                    </br>
        <div class="list-group">';
        if ($_SESSION['user']->idUser != -1) {
            if ($this->games->checkIfGameIsAlreadyInFavoris($_SESSION['user']->idUser, $gameDetail[0]['id'])) {
                $html .= '<a class="btn btn-outline-success " href="?r=games&e=addFavoris&idGame=' . $gameDetail[0]['id'] . '" role="button">Add to favorite</a>';
            } else {
                $html .= '<a class="btn btn-outline-warning " href="?r=games&e=removeFavoris&idGame=' . $gameDetail[0]['id'] . '" role="button">Remove favorite</a>';
            }
        }
        if ($_SESSION['user']->role == 1) {
            $html .= '</br> <a class="btn btn-outline-danger " href="?r=administrator&e=updateGame&id=' . $gameDetail[0]['id'] . '" role="button">Update game</a>';
            $html .= '</br> <a class="btn btn-outline-danger " href="?r=administrator&e=addGameCategorie&id=' . $gameDetail[0]['id'] . '" role="button">Update/add categories</a>';
        }

        $html .= '</div>
                    <h3 class="card-title">Categories</h3>
                    <div class="list-group">';

        foreach ($category as $key => $cat) {
            $html .= '<a href="?r=games&e=categorie&idCategorie=' . $cat['id'] . '"<button type="button" class="btn btn-outline-success btnCategorie margintop10">' . $cat['name'] . '</button></a>';
        }

        $html .= '  </div>
                    </div>
                </div>
            </div>

            ';


        $html .= ";
        return $html;
    }
```

Here is the caller graph for this function:



4.8.3.3 getGamesFromCategorie()

```
getGamesFromCategorie ( )
```

crate a list of game of a specific categorie

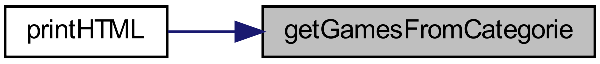
Returns

html

Definition at line 245 of file gamesController.php.

```
    {

        $html = '<div class="cardGameBox box">';
        $listGamesBrut = $this->requestedgame;

        foreach ($listGamesBrut as $key => $game) {

            $html .= $this->createCardHTML($game);
        }

        $html .= '</div>';
        return $html;
    }
```

Here is the caller graph for this function:



4.8.3.4 getListAllGames()

```
getListAllGames ( )
```

create the html of the list of all the games

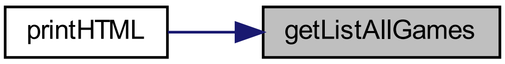
Returns

html

Definition at line 149 of file gamesController.php.

```
    {
        $html = '<div class="cardGameBox box">';

        $listGamesBrut = $this->games->getAllGames();

        foreach ($listGamesBrut as $key => $games) {


            $html .= $this->createCardHTML($games);
        }

        $html .= '</div>';
        return $html;
    }
```

Here is the caller graph for this function:



4.8.3.5 getRequestedGames()

```
getRequestedGames ( )
```

create a list of the requested games

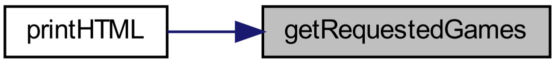
Returns

html

Definition at line 170 of file gamesController.php.

```
    {

        $html = '<div class="cardGameBox box">';
        $listGamesBrut = $this->requestedgame;
        foreach ($listGamesBrut as $key => $games) {


            $html .= $this->createCardHTML($games);
        }

        $html .= '</div>';
        return $html;
    }
```

Here is the caller graph for this function:



4.8.3.6 printHTML()

`printHTML ( )`

print the html for the resquested content

Returns

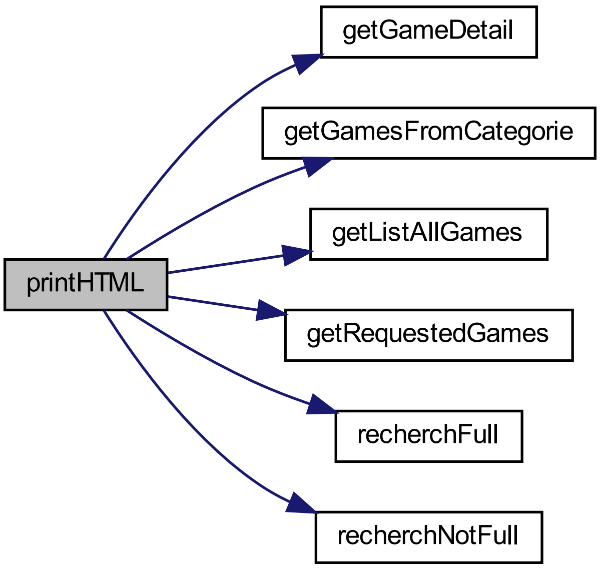
html

Implements iController.

Definition at line 115 of file gamesController.php.

```
    {
        $html = '<main style="margin-top:20px">
        <div class="container-md">';
        $html .= $this->errorHandler();
        if ($this->e == null) {
            $html .= $this->recherchFull();
            $html .= $this->getListAllGames();
        }

        if ($this->e == "requestGame") {
            $html .= $this->recherchFull();
            $html .= $this->getRequestedGames();
        }

        if ($this->e == "detail") {
            $html .= $this->recherchNotFull();
            $html .= $this->getGameDetail();
        }

        if ($this->e == "categorie") {
            $html .= $this->recherchFull();
            $html .= $this->getGamesFromCategorie();
        }

        $html .= "</div></main> ";
        echo $html;
    }
```

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

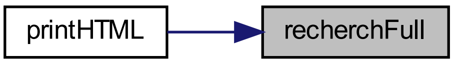
Returns

html

Definition at line 297 of file gamesController.php.

```
    {
        $html = "";

        $html .= '<div class="jumbotron DarkJumbotron  " style="background-color: #161b22;">
        <div class="container">
          <h1 class="display-5">Research</h1>

          <form class="row g-3" action="?r=games&e=requestGame" method="post">

            <div class="col-auto">
                <input type="texte" class="form-control" id="gameName" name="gameName"
       placeholder="Mario">
            </div>
            <div class="col-auto">
                <button type="submit" class="btn btn-success mb-3">Research</button>
            </div>
          </form>
          <h4>Categories:</h4>
          <p>

          ';
        foreach ($this->categorie->getListAllCategories() as $key => $cat) {
            $html .= '<a class="btn btn-outline-success btnCategorie "
    href="?r=games&e=categorie&idCategorie=' . $cat['id'] . '" role="button">' . $cat['name'] . '</a>';
        }
        $html .= '
        </p>
          </div>
      </div>';

        return $html;
    }
```

Here is the caller graph for this function:



4.8.3.8 recherchNotFull()

`recherchNotFull ( )`

create the html of a form to research game

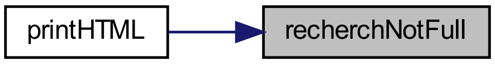
Returns

html

Definition at line 334 of file gamesController.php.

```
    {
        $html = "";

        $html .= '<div class="card  " style="background-color: #0d1117;">
        <div class="card-body container DarkJumbotron">
          <h2 class="card-title ">Research</h2>

          <form class="row g-3" action="?r=games&e=requestGame" method="post">

            <div class="col-auto">
                <input type="texte" class="form-control" id="gameName" name="gameName"
    placeholder="Mario">
            </div>
            <div class="col-auto">
                <button type="submit" class="btn btn-success mb-3">Research</button>
            </div>
          </form>
          </div>
      </div>';

        return $html;
    }
```

Here is the caller graph for this function:

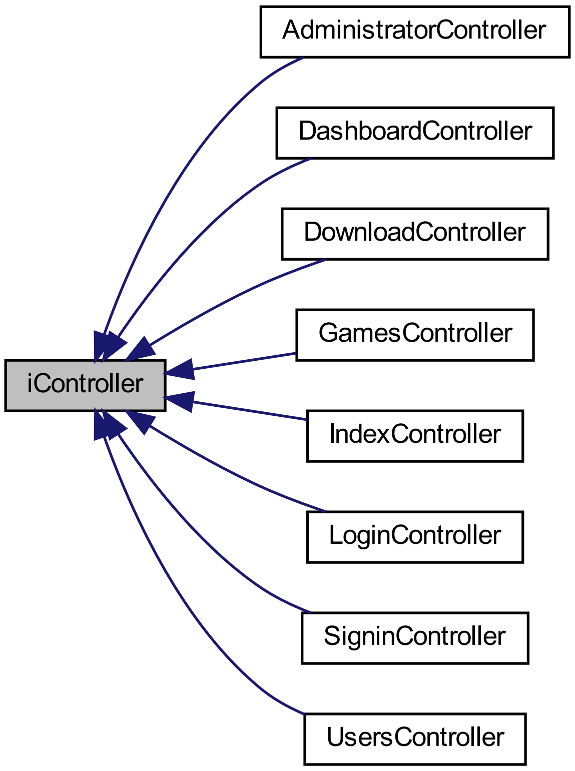


The documentation for this class was generated from the following file:

- controllers/gamesController.php

4.9 iController Interface Reference

Inheritance diagram for iController:



Public Member Functions

- **formHandler** ()
- **printHTML** ()

4.9.1 Detailed Description

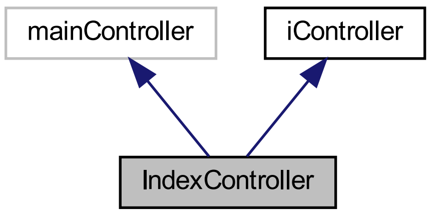
Definition at line 10 of file interfaceController.php.

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

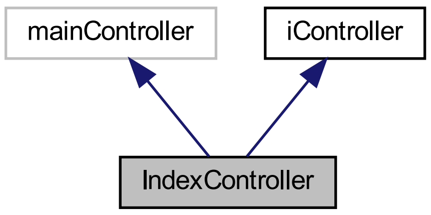
- controllers/interfaceController.php

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.

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

4.10.2.2 printHTML()

`printHTML ( )`

print the html for the resquested content

Returns

html

Implements iController.

Definition at line 29 of file indexController.php.

```
{
  $html = "";


  $html .= '
  <main style="margin-top:20px ">
    <div class="container-md">
    ';
  $html .= $this->errorHandler();
  $html .= '
      <div class="jumbotron jumbotron-fluid DarkJumbotron width100" style="background-color: #161b22;">
          <div class="row py-lg-5">
            <div class="col-lg-6 col-md-8 mx-auto">
              <h1 class="fw-light greenTexte">Caiman</h1>
              <p class="lead text-muted">The easiest way to use emulators.</p>
              <p>
                <a href="?r=download" class="btn btn-success my-2">Download</a>
                <a href="?r=login" class="btn btn-success my-2">Create acount</a>
                <a href="?r=games" class="btn btn-success my-2">Watch games list</a>
              </p>
            </div>
          </div>
      </div>
    </div>

  </main>

';
  echo $html;
}
```

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.

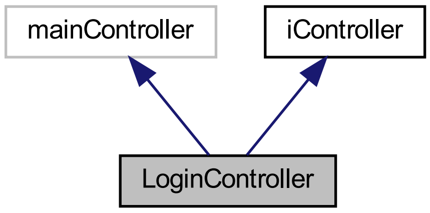
```
    {
        $returnArray = null;
        try{
            $this->psLogin->execute(array(':search_username' => $this->search_username));
            $result = $this->psLogin->fetchAll();
            if ($result != null) {
                if (password_verify( $this->search_password,$result[0]["password"])   ) {
                    $returnArray = $result;
                    $_SESSION['error'] = "Welcome back: ". $result[0]['username'];
                }else
                {
                    $_SESSION['error'] = "Invalid log in";
                }
            }

        }catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $returnArray;
    }
```

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

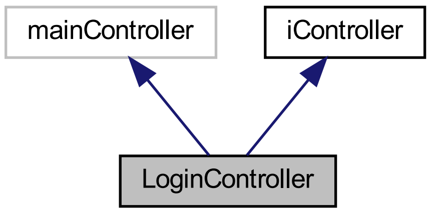
- models/login.php

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.

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

4.12.3 Member Function Documentation

4.12.3.1 formHandler()

```
formHandler ( )
```

used to handle if the user has resquest something

Returns

void

Implements iController.

Definition at line 28 of file loginController.php.

```
{
  if (isset($_GET['e'])) {
    $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_STRING);
  }

  if ($this->e == "login") {
    $_SESSION['title'] = "Caiman: Login";
    if (isset($_POST['username'])) {
      $username = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_STRING);
    }
    if (isset($_POST['password'])) {
      $password = filter_input(INPUT_POST, 'password', FILTER_SANITIZE_STRING);
    }

    if (isset($password) && isset($username)) {
      $this->login->search_username = $username;
      $this->login->search_password = $password;

      $usersInfos = $this->login->checkLogin();

      if (isset($usersInfos)) {
        $_SESSION['user'] = new User($usersInfos[0]['username'], $usersInfos[0]['email'],
  $usersInfos[0]['idRole'], $usersInfos[0]['id']);
        header('Location:' . $_SERVER['HTTP_REFERER']);
        exit;
      }
    }
  }
}
```

4.12.3.2 printHTML()

```
printHTML ( )
```

print the html for the resquested content

Returns

void

Implements iController.

Definition at line 65 of file loginController.php.

```
{

  $html = '<main  style="margin-top:20px ">
      <div class="container-md">';
  $html .= $this->errorHandler();
  $html .= $this->htmlFormHead();

  $html .= "</div></main> ";

  echo $html;
}
```

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

- controllers/loginController.php

4.13 MainController Class Reference

Public Member Functions

- __construct ()
- allowAccessTo ($allowAccessToId)
- 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.

```
  {
  }
```

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

[type]	$allowAccess↩ ToId

Returns

void

Definition at line 26 of file mainController.php.

```
    {

        $isValid = false;
        foreach ($allowAccessToId as $key => $validId) {
            if ($validId == $_SESSION['user']->role) {
                $isValid = true;
            }
        }

        if ($isValid == false) {
            header('Location: index.php');
            $_SESSION['error'] = "You can't access this page!";
            exit;
        }
    }
```

4.13.3.2 errorHandler()

```
errorHandler ( )
```

create the html of an error

Returns

html

Definition at line 48 of file mainController.php.

```
    {
        $html = "";
        if (isset($_SESSION['error']) && $_SESSION['error'] != null) {

            $html .= '
          <div class=" warningJumbotron errorMessageDiv" style="background-color: #161b22; ">

                  <h5>' . $_SESSION['error'] . '</h5>

          </div>';
            $_SESSION['error'] = null;
        }
        return $html;
    }
```

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.

```
    {
        if ($this->dbh == null) {
            try {
                $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
                    PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
                    PDO::ATTR_PERSISTENT => true
                ));

                // check if email alredy used
                $sqlrequestEmail = "SELECT * FROM user WHERE email = :search_email ";
                $this->psCheckEmail = $this->dbh->prepare($sqlrequestEmail);
                $this->psCheckEmail->setFetchMode(PDO::FETCH_ASSOC);

                // check if username alredy used
                $sqlRequestUsername = "SELECT * FROM user WHERE username = :search_username ";
                $this->psCheckUsername = $this->dbh->prepare($sqlRequestUsername);
                $this->psCheckUsername->setFetchMode(PDO::FETCH_ASSOC);

                $sqlInsert = "INSERT INTO user  (username, password, email)
                              VALUES (:insert_username, :insert_password, :insert_email)";
                $this->psInsert = $this->dbh->prepare($sqlInsert);
                $this->psInsert->setFetchMode(PDO::FETCH_ASSOC);
            } catch (PDOException $e) {
                print "Erreur !: " . $e->getMessage() . "<br>";
                die();
            }
        }
    }
```

4.14.3 Member Function Documentation

4.14.3.1 checkIfEmailAlreadyTaken()

`checkIfEmailAlreadyTaken ( )`

check that the email is not already taken

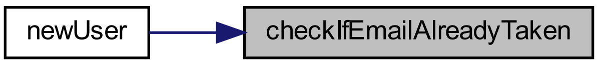
Returns

bool

Definition at line 116 of file signin.php.

```
    {
        $istaken = true;
        try {
            $this->psCheckEmail->execute(array(':search_email' => $this->insert_email));
            $result = $this->psCheckEmail->fetchAll();
            if ($result == null) {
                $istaken = false;
            }
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $istaken;
    }
```

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.

```
    {
        $istaken = true;
        try {
            $this->psCheckUsername->execute(array(':search_username' => $this->insert_username));
            $result = $this->psCheckUsername->fetchAll();
            if ($result == null) {
                $istaken = false;
            }
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $istaken;
    }
```

4.14.3.3 newUser()

```
newUser ( )
```

add a mew user in the database

Returns

void

Definition at line 66 of file signin.php.

```
{
    $isValid = true;
    if ($this->checkIfEmailAlreadyTaken()) {
        $_SESSION['error'] = "Email already used";
        $isValid = false;
    }
    if ($this->checkifUsernameAlreadyTaken()) {
        $_SESSION['error'] = "Username alredy used";
        $isValid = false;
    }
    if ($isValid) {


        try {
            $this->psInsert->execute(array(':insert_username' => $this->insert_username,
':insert_password' => password_hash($this->insert_password, PASSWORD_DEFAULT), ':insert_email' =>
$this->insert_email));
            $_SESSION['error'] = "Acount created";
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
    }
}
```

Here is the call graph for this function:

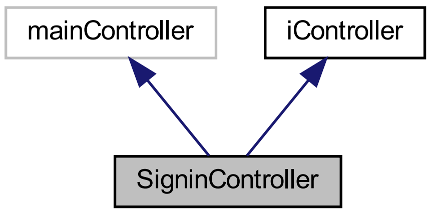
newUser
checkIfEmailAlreadyTaken

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

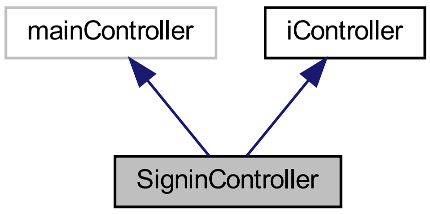
- models/signin.php

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.

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

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.

```
    {
        if (isset($_GET['e'])) {
            $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_SPECIAL_CHARS);
        }
        if ($this->e == "signin") {
            if (isset($_POST['username'])) {
                $this->signin->insert_username = filter_input(INPUT_POST, 'username',
    FILTER_SANITIZE_SPECIAL_CHARS);
            }
            if (isset($_POST['password'])) {
                $this->signin->insert_password = filter_input(INPUT_POST, 'password',
    FILTER_SANITIZE_SPECIAL_CHARS);
            }
            if (isset($_POST['passwordRepeat'])) {
                $this->signin->insert_password_repeat = filter_input(INPUT_POST, 'passwordRepeat',
    FILTER_SANITIZE_SPECIAL_CHARS);
            }
            if (isset($_POST['email'])) {
                $this->signin->insert_email = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL);
            }


            if (isset($this->signin->insert_password) && isset($this->signin->insert_username) &&
    isset($this->signin->insert_password_repeat) && isset($this->signin->insert_email)) {

                if ($this->signin->insert_password != $this->signin->insert_password_repeat) {
                    $_SESSION['error'] = "Password does not match";
                    header('Location:' . $_SERVER['HTTP_REFERER']);
                    exit;
                }


                $this->signin->newUser();

                header('Location:' . $_SERVER['HTTP_REFERER']);
                exit;
            } else {
                $_SESSION['error'] = "form not completed";
                header('Location:' . $_SERVER['HTTP_REFERER']);
                exit;
            }
        }
    }
```

4.15.3.2 printHTML()

```
printHTML ( )
```

print the html for the resquested content

Returns

void

Implements iController.

Definition at line 75 of file signinController.php.

```
    {
    }
```

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	*$usernamep*	
string	*$emailp*	
string	*$idRolep*	
integer	*$idUserp*	

Definition at line 26 of file user.php.

```
    {

        $this->username = $usernamep;
        $this->email = $emailp;
        $this->role = $idRolep;
        $this->idUser = $idUserp;
    }
```

4.16.3 Member Function Documentation

4.16.3.1 getPrivateAccount()

```
getPrivateAccount ( )
```

use to know if the account is privat or not

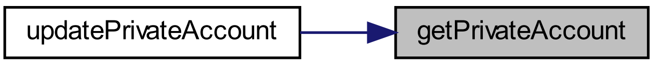
Returns

bool

Definition at line 109 of file user.php.

```
    {
        $dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
            PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
            PDO::ATTR_PERSISTENT => true
        ));

        try {
            $sqlGetPrivateAccount = "SELECT privateAccount FROM user WHERE id = :id_user";
            $psGetPrivateAccount = $dbh->prepare($sqlGetPrivateAccount);
            $psGetPrivateAccount->setFetchMode(PDO::FETCH_ASSOC);
            $psGetPrivateAccount->execute(array(':id_user' => $this->idUser));
            $result = $psGetPrivateAccount->fetchAll();
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }

        return $result[0]['privateAccount'];
    }
```

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	*$newPassword*	
string	*$newPasswordRepeat*	
string	*$oldPassword*	

Returns

int

Definition at line 43 of file user.php.

```
    {
        $dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
            PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
            PDO::ATTR_PERSISTENT => true
        ));

        $hasBeenUpdated = 1;
        if (password_verify($oldPassword, $this->getUserPassword())) {

            if ($newPasswordRepeat == $newPassword) {
                try {
                    $sqlUpdatePassword = "UPDATE user  SET password = :update_password WHERE id =
    :id_user";
                    $psUpdatePassword = $dbh->prepare($sqlUpdatePassword);
                    $psUpdatePassword->execute(array(':update_password' => password_hash($newPassword,
    PASSWORD_DEFAULT), ':id_user' => $this->idUser));
```

```
                    $hasBeenUpdated = 0;
                } catch (PDOException $e) {
                    print "Erreur !: " . $e->getMessage() . "<br>";
                    die();
                }
            } else {
                $hasBeenUpdated = 2;
            }
        } else {
            $hasBeenUpdated = 4;
        }

        return $hasBeenUpdated;
    }
```

4.16.3.3 updatePrivateAccount()

`updatePrivateAccount ( )`

update if the account is private or not

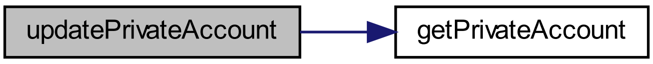
Returns

void

Definition at line 79 of file user.php.

```
    {
        $dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
            PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
            PDO::ATTR_PERSISTENT => true
        ));

        $userisPrivate = $this->getPrivateAccount();

        if ($userisPrivate == 0) {
            $userSetPrivateTo = 1;
        } else {
            $userSetPrivateTo = 0;
        }

        try {
            $sqlUpdatePrivateAccount = "UPDATE user  SET privateAccount = :update_private_account WHERE
    id = :id_user";
            $psUpdatePrivateAccount = $dbh->prepare($sqlUpdatePrivateAccount);
            $psUpdatePrivateAccount->execute(array(':update_private_account' => $userSetPrivateTo,
    ':id_user' => $this->idUser));
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
             die();
         }
     }
```

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.

```
    {
        if ($this->dbh == null) {
            try {
                $this->dbh = new PDO('mysql:host=' . HOST . ';dbname=' . DBNAME, USER, PASSWORD, array(
                    PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
                    PDO::ATTR_PERSISTENT => true
                ));

                // get list of user by username
                $sqlRequestUsers = "SELECT * FROM user WHERE username LIKE :search_username AND
    privateAccount = 0";
                $this->psGetUsersByUsername = $this->dbh->prepare($sqlRequestUsers);
                $this->psGetUsersByUsername->setFetchMode(PDO::FETCH_ASSOC);

                // get list of user by username
                $sqlGetDataUser = "SELECT * FROM user WHERE id =:search_idUser";
                $this->psGetDataUser = $this->dbh->prepare($sqlGetDataUser);
                $this->psGetDataUser->setFetchMode(PDO::FETCH_ASSOC);
            } catch (PDOException $e) {
                print "Erreur !: " . $e->getMessage() . "<br>";
                die();
            }
        }
    }
```

4.17.3 Member Function Documentation

4.17.3.1 getUserData()

```
getUserData (
                $iduser )
```

get data of a specific user

Parameters

int	*$iduser*	

Returns

array

Definition at line 70 of file userdata.php.

```
    {
        try {
            $this->psGetDataUser->execute(array(':search_idUser' => $iduser));
            $result = $this->psGetDataUser->fetchAll();
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

4.17.3.2 getUsersByUsername()

```
getUsersByUsername (
                $username )
```

get users by their username

Parameters

string	*$username*	

Returns

array of game

Definition at line 52 of file userdata.php.

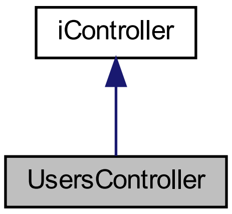
```
    {
        try {
            $this->psGetUsersByUsername->execute(array(':search_username' => '%' . $username . '%'));
            $result = $this->psGetUsersByUsername->fetchAll();
        } catch (PDOException $e) {
            print "Erreur !: " . $e->getMessage() . "<br>";
            die();
        }
        return $result;
    }
```

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

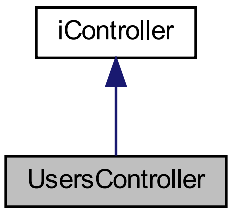
- models/userdata.php

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.

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

4.18.3 Member Function Documentation

4.18.3.1 formHandler()

`formHandler ( )`

used to handle if the user has resquest something

Returns

void

Implements iController.

Definition at line 24 of file usersController.php.

```
    {
        $_SESSION['title'] = "Caiman: Users";

        if (isset($_GET['e'])) {
            $this->e = filter_input(INPUT_GET, 'e', FILTER_SANITIZE_SPECIAL_CHARS);
        }

        // request user by their username
        if ($this->e == "researchUser") {
            $_SESSION['title'] = "Caiman: Search";
            if (isset($_POST['username'])) {
                $this->requestUsername = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_STRING);
            }
        }

        // show detail of a user
        if ($this->e == "detailUser") {
            $_SESSION['title'] = "Caiman: User detail";
            if (isset($_GET['idUser'])) {
                $this->idUser = filter_input(INPUT_GET, 'idUser', FILTER_SANITIZE_STRING);
            }
        }
    }
```

4.18.3.2 htmlDetailUser()

```
htmlDetailUser ( )
```

create a page with the details of a user

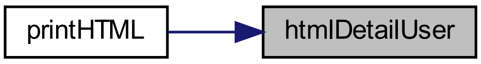
Returns

html

Definition at line 151 of file usersController.php.

```
    {
        $html = "";

        $html .= '<div class="card  " style="background-color: #0d1117;">
        <div class="card-body container DarkJumbotron">
          <h2 class="card-title ">Results</h2>
          <div class="list-group">
          ';

        foreach ($this->userData->getUsersByUsername($this->requestUsername) as $key => $user) {

            $html .= '<a class="btn btn-outline-success btnCategorie margintop10 "
   href="?r=users&e=detailUser&idUser=' . $user['id'] . '" role="button">' . $user['username'] . '</a>';
        }

        $html .= '
            </div>
          </div>
      </div>';

        return $html;
    }
```

Here is the caller graph for this function:



4.18.3.3 htmlrecherchUsers()

```
htmlrecherchUsers ( )
```

create the form to search users

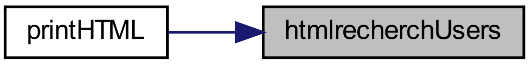
Returns

html

Definition at line 95 of file usersController.php.

```
    {
        $html = "";

        $html .= '<div class="card  " style="background-color: #0d1117;">
         <div class="card-body container DarkJumbotron">
           <h2 class="card-title ">Research</h2>

           <form class="row g-3" action="?r=users&e=researchUser" method="post">

             <div class="col-auto">
                 <input type="texte" class="form-control" id="username" name="username"
placeholder="username">
             </div>
             <div class="col-auto">
                 <button type="submit" class="btn btn-success mb-3">Research</button>
             </div>
           </form>
           </div>
       </div>';

         return $html;
     }
```

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.

```
     {
         $html = "";

         $html .= '<div class="card  " style="background-color: #0d1117;">
         <div class="card-body container DarkJumbotron">
           <h2 class="card-title ">Results</h2>
           <div class="list-group">
           ';

         foreach ($this->userData->getUsersByUsername($this->requestUsername) as $key => $user) {

             $html .= '<a class="btn btn-outline-success btnCategorie margintop10 "
href="?r=users&e=detailUser&idUser=' . $user['id'] . '" role="button">' . $user['username'] . '</a>';
         }
```

```

        $html .= '
            </div>
          </div>
      </div>';

        return $html;
    }
```

4.18.3.5 printHTML()

```
printHTML ( )
```

print the html for the resquested content

Returns

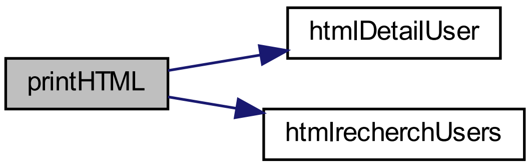
void

Implements iController.

Definition at line 64 of file usersController.php.

```
    {

        $html = '<main  style="margin-top:20px ">
        <div class="container-md">';

        if ($this->e == null) {
            $html .= $this->htmlrecherchUsers();
        }

        if ($this->e == "researchUser") {
            $html .= $this->htmlrecherchUsers();
            $html .= $this->htmlDetailUser();
        }

        if ($this->e == "detailUser") {
            $html .= $this->htmlDataUser();
            $html .= $this->htmlFavoriteGameUser();
            $html .= $this->htmlTimeInGameUser();
        }

        $html .= "</div></main> ";

        echo $html;
    }
```

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