**Creational Design Pattern Report**

Group 5

Bichi Zhang

04/18/2019

Singleton

I create this design pattern for game login. When we are playing games, someone logs in to our account or when we quit or drop from a game and then quick re-login, there will be a prompt on the login screen saying that your account is online and you cannot log in. I think one account can only be logged in at one place at the same time. So the singleton pattern is a good design pattern for it.

Code:

File Name: LoginDetection.h

Author: Bichi Zhang

Date: 04/14/2019

Complier Used: C++ by Visual Studio 2017

#ifndef \_SINGLETON\_H\_

#define \_SINGLETON\_H\_

class LoginDetection

{

public:

static LoginDetection\* GetInstance();

void OutputInfor();

protected:

LoginDetection();

private:

static LoginDetection \*\_instance;

int count;

};

#endif

File Name: LoginDetection.cpp

Author: Bichi Zhang

Date: 04/14/2019

Complier Used: C++ by Visual Studio 2017

#include "LoginDetection.h"

#include <iostream>

using namespace std;

LoginDetection\* LoginDetection::\_instance = 0;

LoginDetection::LoginDetection()

{

cout << "Create Singleton ..." << endl;

count = 0;

}

LoginDetection\* LoginDetection::GetInstance()

{

if (\_instance==0)

{

\_instance = new LoginDetection();

}

return \_instance;

}

void LoginDetection::OutputInfor()

{

cout << "This account logged in " << ++count <<" times today."<< endl;

}

File Name: main.cpp

Author: Bichi Zhang

Date: 04/14/2019

Complier Used: C++ by Visual Studio 2017

#include "LoginDetection.h"

int main()

{

LoginDetection \*t1 = LoginDetection::GetInstance();

t1->OutputInfor();

LoginDetection \*t2 = LoginDetection::GetInstance();

t2->OutputInfor();

LoginDetection \*t3 = LoginDetection::GetInstance();

t3->OutputInfor();

return 0;

}

Screenshot:

