Міністерство освіти і науки України  
Тернопільський національний економічний університет  
Факультет комп’ютерних інформаційних технологій

Кафедра економічної кібернетики

Комплексне практичне індивідуальне завдання

з дисципліни «Конструювання ПЗ»

на тему:

«Гра на швидкість реакції»

Виконав:

студент групи ПЗС-32

Літвінчук М.С.

Перевірив:

Сирник О. Й.

Тернопіль 2015

Вступ

Всі ми, напевно, маємо справу з комп`ютерною технікою. І більшість з нас користується при цьому таким аксесуаром, як комп`ютерна миша. Власники ж ноутбуків мають у своєму розпорядженні тачпад – сенсорна панель управління курсором. Безперечно, кожен опинявся в такій ситуації, коли потрібно дуже швидко закрити, приховати вікно, сторінку, закрити банер і т.д. Дана гра розроблена якраз для тренування навиків поводження з мишою, тачпадом. Тут вам знадобиться максимальна концентрація, спритність і швидкість прийняття рішень.

Системи керування версій

В контексті розробки програмних систем, керування версіями можна розглядати як мистецтво управління паралельними змінами. Зазвичай системи керування версіями (англ. source code management) працюють як центральні сховища (repository) і мають інтерфейс для доступу і спільного використання коду.

Для прикладу допустимо, що двом розробникам потрібно працювати з одним і тим самим файлом. Вони можуть відкрити і редагувати файл, перезаписувати, витираючи зміни здійсненні один одним. Або ж один розробник очікує, поки інший завершить редагування файлу, байдикуючи.

Система керування версіями є рішинням цих проблеми. Вона здатна слідкувати за змінами кожного із файлів сховища і користувач може бачити не тільки поточний стан файлу, але й усі зміни, які відбувались за час його існування. Завдяки цьому можливо здійснювати відкат (rollback) помилкового коду, забезпечуючи повернення до робочої версії. Певні версії файлу можна помічати як кінцеві і продовжуючи розробку завжди мати доступ до копій версії, які на даний момент вважаються кінцевими.

Окрім цього, системи керування версіями дозволяють декільком програмістам одночасно працювати над один кодом. Кожний із них може отримати (checkout) копію коду із сховища, а після завершення роботи з ним помістити (commit) назад. У цьому випадку вважається, що версія прийнята. Звісно ж дані системи можуть прослідкувати які зміни були здійснені користувачем.

Варто також зазначити, що система управління версіями здатна злити зміни здійснені різними користувачами, лише якщо вони здійснені у різних частинах файлу. Якщо ж розробники відредагували один і той же код у своїх локальних копіях, то перший із них зможе помістити код у репозиторій, а другий зможе зробити те ж саме тільки якщо вирішить яким чином поєднати зміни першого зі своїми. Тільки після вирішення конфлікту сховище дозволить зберегти файл.

Git – це розподілена система керування версіями файлів, яка була створена Лінуксом Торвальдсом у 2005-му році для керування розробкою ядра Linux. До проектів, які використовують дану систему також належать: Android, Drupal, Cairo, GNU Core Utilities, Mesa, Wine, Chromium, jQuery, PHP, NASM, MediaWiki, osCommerce, проекти Twitter та Yahoo, а також деякі дистрибутиви Linux.

Система розроблялась як набір програм командної стрічки з параметрами, спеціально розроблених для використання у сценаріях, що дозволило створювати на її базі спеціалізовані системи контролю версій.

Сховище Git являє собою каталог файлової системи, в якому знаходяться:

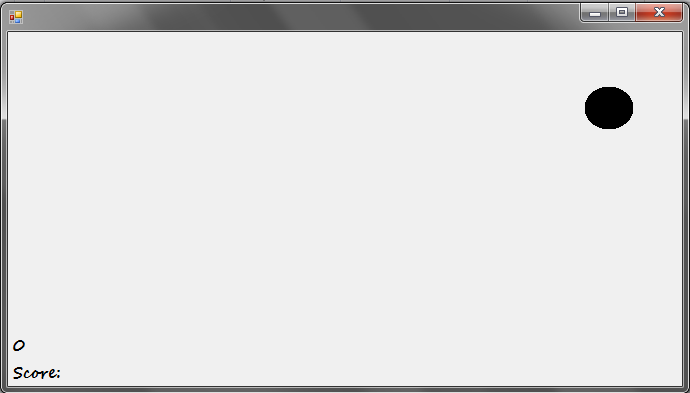
* файли налаштувань сховища
* файли журналів, які зберігають зберігають операції, здійснені над сховищем
* індекс – описує розташування файлів
* сховище, яке власне і зберігає файли

Файли в Git можуть знаходитись в одному із трьох станів: зафіксованому, зміненому і підготовленому. Зафіксований файл той, який вже збережений у локальні базі. До змінених файлів відносяться ті, які змінились, але ще не були зафіксовані. Підготовлені файли – це змінені файли, що помічені для включення у наступну фіксацію (commit).

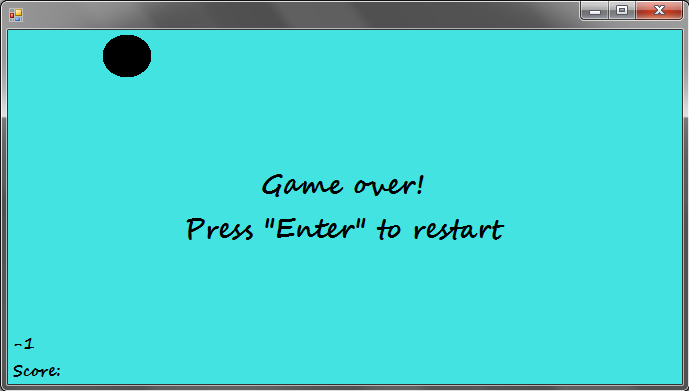
Тому в проектах, що використовують Git є три частини: каталог Git (Git repository), робочий каталог (working directory) і область підготовлених файлів (staging area).

Практична частина

Суть даної гри полягає в тому, щоб якнайшвидше навести курсор миші на м`ячик і натиснути на нього. Після натиснення він пропадає і з`являється вже на зовсім іншому місці. За кожне вдале натискання гравець отримує 1 бал, за натискання повз м`ячика гравцеві віднімається 1 бал. Гра закінчиться, коли бали гравця будуть менші нуля. На рисунку №1 зображено головну ігрову зону:

Рис.1 Ігрова зона

На рисунку №2 рображено вікно гри, коли користувач програв:

Рис.2 Вікно програшу

Код програми

Реалізація події натискання мишею на м`ячик:

private void ball\_Click(object sender, EventArgs e)

{

speed\_left = -speed\_left;

speed\_top = -speed\_top;

timer1.Enabled = false;

timer1.Enabled = true;

int p1 = playground.Size.Width;

int p2 = playground.Size.Height;

int s1 = ball.Size.Width;

int s2 = ball.Size.Height;

ball.Location = new Point(rand.Next(p1 -s1),rand.Next(p2 - s2));

ball.BackColor = Color.FromArgb(rand.Next(1, 250), rand.Next(1, 250), rand.Next(1, 250));

points += 1;

points\_lbl.Text = points.ToString();

}

Реалізація зіткнення м`ячика з границями ігрової області:

private void timer2\_Tick(object sender, EventArgs e)

{

ball.Left += speed\_left; //Рух м`яча

ball.Top += speed\_top;

if (ball.Left <= playground.Left)

{

speed\_left = -speed\_left;

}

if (ball.Right >= playground.Right)

{

speed\_left = -speed\_left;

}

if (ball.Top <= playground.Top)

{

speed\_top = -speed\_top;

}

if (ball.Bottom >= playground.Bottom)

{

speed\_top = -speed\_top;

}

if (points < 0)

{

gameover\_lbl.Visible = true;

timer1.Enabled = false;

timer2.Enabled = false;

}

}

Реалізація методу, коли на м`ячик не було вчасно натиснуто:

private void timer1\_Tick(object sender, EventArgs e)

{

speed\_left = -speed\_left;

speed\_top = -speed\_top;

int p1 = playground.Size.Width;

int p2 = playground.Size.Height;

int s1 = ball.Size.Width;

int s2 = ball.Size.Height;

ball.Location = new Point(rand.Next(p1 - s1), rand.Next(p2 - s2));

ball.BackColor = Color.FromArgb(rand.Next(1, 250), rand.Next(1, 250), rand.Next(1, 250));

points -= 1;

points\_lbl.Text = points.ToString();

}

Висновки

Гра на на швидкість реакції – чудовий спосіб провести час із користю та задоволенням.

Список використаних джерел

1. Jeffrey Richter – CLR via C# (4th Edition) (Developer Reference)
2. Herbert Schildt – C# 4.0 The Complete Reference
3. Microsoft Developer Network (MSDN)
4. 3. Пугачев С., Шериев А., Кичинский К. Разработка приложений для Windows 8 на языке C#; БХВ-Петербург, 2013. - 416 c.