Memory – projekt z PUM

Sprawozdanie z projektu z PUM

Autor:	Anna Wieczorek
Grupa:	IiAM1
Utworzony:	15 lipca 2014

Spis treści

1. Założenia projektu

Gra 'Memory' jest obsługiwana za pomocą urządzeń mobilnych z systemem Android.
Użytkownik musi dobrać pary schowanych ikon. Popularna gra polegająca na znalezieniu par takich samych ilustracji.
Po uruchomieniu na screen'ie urządzenia pojawi się 16(w schemacie 4x4) zakrytych obrazków w formie bloków). Po dotknięciu/kliknięciu użytkownik odwraca dwie karty - jeśli nie ma pary to karty spowrotem zostają zakryte, jeśli została odnaleziona para to karty pozostają odsłonięte

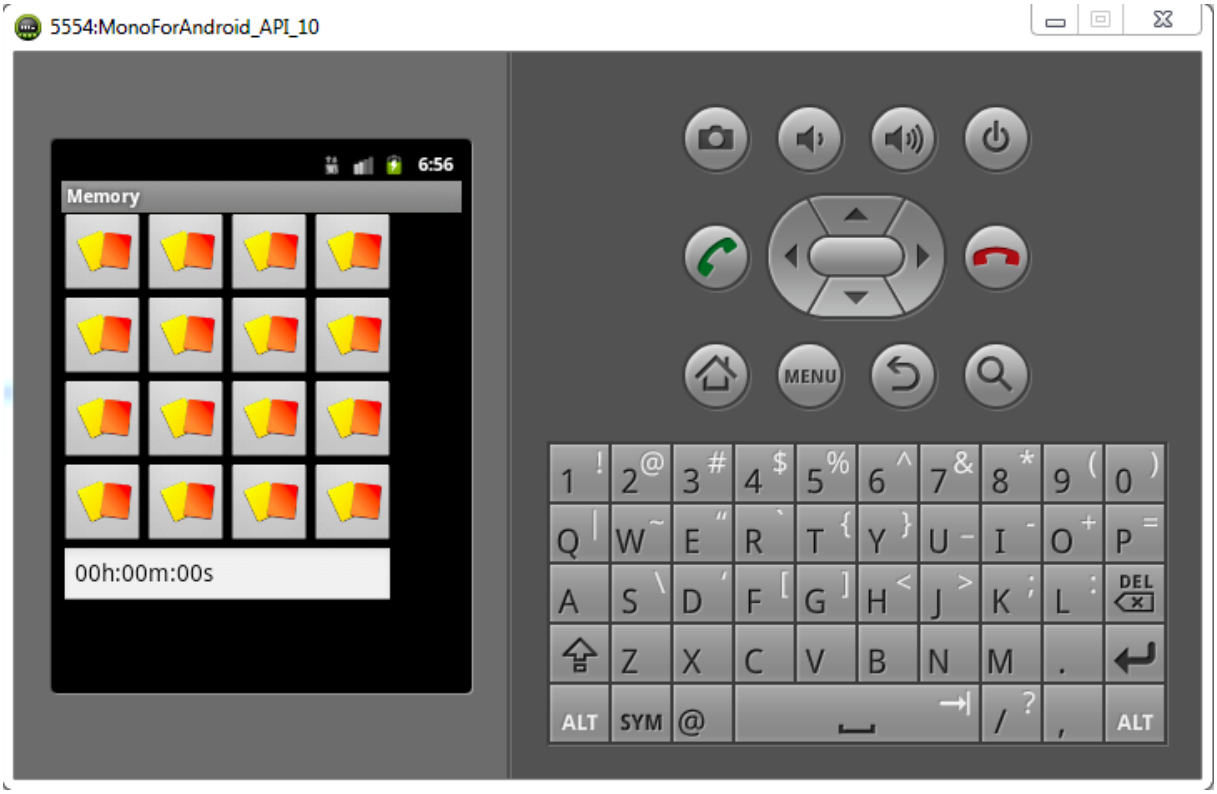

Figure 1 Widok aplikacji po uruchomieniu

2. Środowisko aplikacji

Aplikacja została napisana przy użyciu frameworka Xamarin, który umożliwia tworzenie aplikacji mobilnych działających na urządzeniach z systemem Andorid bądź iOS. Framework ten opiera się na platformie .NET i umożliwia tworzenie aplikacji wykorzystując do tego popularny język programowania C#. Co więcej, integruje się on ze środowiskiem Visual Studio oraz pozwala na uruchomienie, testowanie oraz debugowanie tworzonej aplikacji na jednym z wielu emulatorów, które także wchodzą w skład tegoż frameworka.

Głównym ograniczeniem frameworka jest fakt, że jest narzędziem płatnym a licncja free dostępna jest przez miesiąc – po tym okresie zbudowane apk przestają działać.

Jendakże jest to wydajne narzędzie, któ®e pozwala na tworzenie aplikacji zarówno na systemach Windows jak i Mac.

Xamarin pozwala także na stworzenie wirtualnej chmury pozwalającej na testowanie aplikacja na wielu urządzeniach. Nnarzędzie to automatycznie generuje przejrzyste raporty, pozwalające na szybką i sprawną weryfikację działania funkcjonalności aplikacji. Ponadto ciągła integracja wpływa na płynność w budowaniu aplikacji i usprawnia proces deploymentu.

Dzisiaj producenci Xamarin'a organizują nawet wirtualne lekcje dotyczące progromowania – lekcje odbywają się na żywo, w danej strefie czasowej o odpowiedniej porze – również płatne.

Nardzędzie jakim jest Xamarin z pewnością znajdzie zastosowanie wśród firm dostarczającym orpogramowanie na urządzenia mobile – zaletą jest platforma .NET i połączenie systemu Android – niekonwencjonalne jednak nowe technologie znajdują wiele zainterosowania wśród firm IT.

3. Specyfikacja zewnętrzna

3.1. Ekran główny aplikacji

Użytkownik po urochumieniu aplikacji na urządzeniu, bądź emulatorze widzi 16 odwróconych kart oraz timer znajdujący się pod blokiem kart, który mierzy czas od chwili odwrócenia pierwszej karty do chwili zakończenia gry, czyli odkrycia wszystkich 8 par kart.

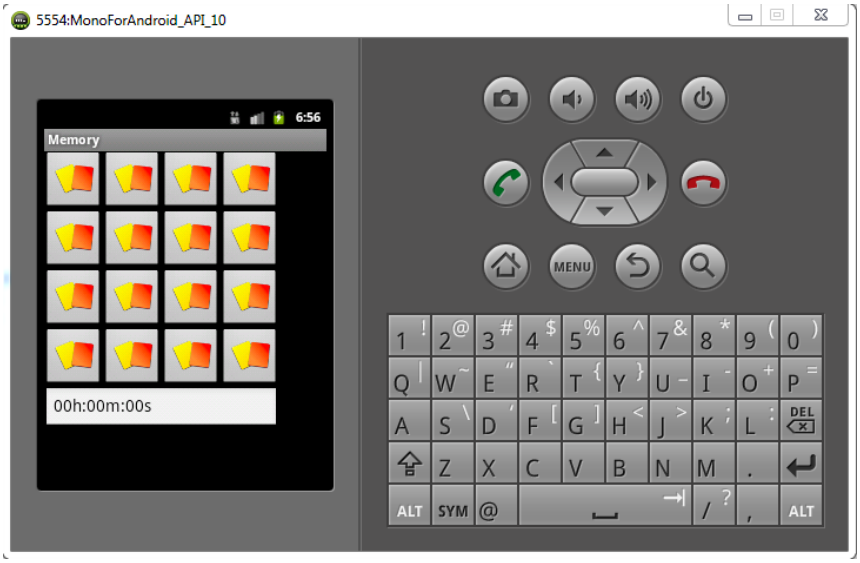

Figure 2 Ekran główny aplikacji Memory

3.2. Uruchomienie aplikacji

Gra startuje w chwili, kiedy użytkownik odwróci pierwszą kartę – wtedy widzimy, że licznik timera startuje I zaczyna mierzyć czas. Po odwróceniu kart użytkownik widzi konkretne obrazki – teraz jego zadaniem jest znalezienie karty z takim samym obrazkiem – kiedy kliknie w taki sam to karty pozostaną odsłonięte –w przeciwnym razie nie znalazł pary i karty spowrotem się odwracają.

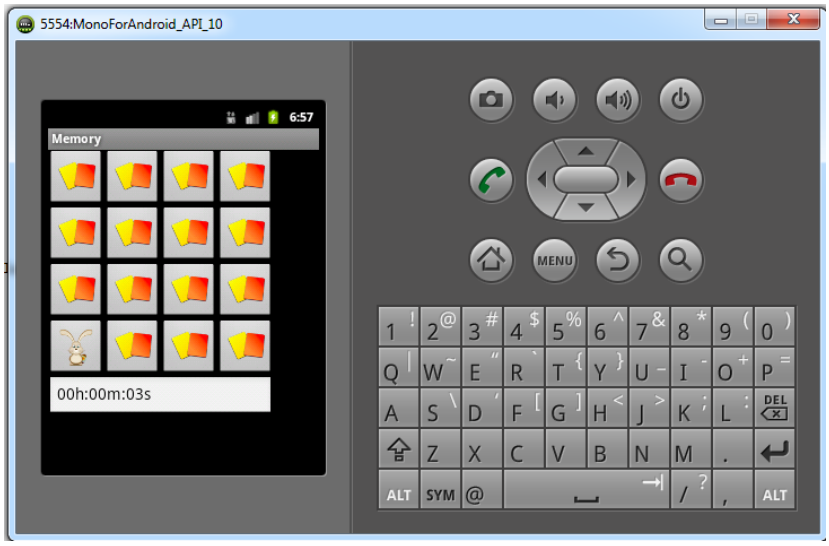

Figure 3 Odwrócenie pierwszej karty w grze

Na poniżym zdjęciu widzimy, że użytkownik nie odnajduje pary dla karty wcześniej odkrytej – po chwili karty spowrotem się zakryją.

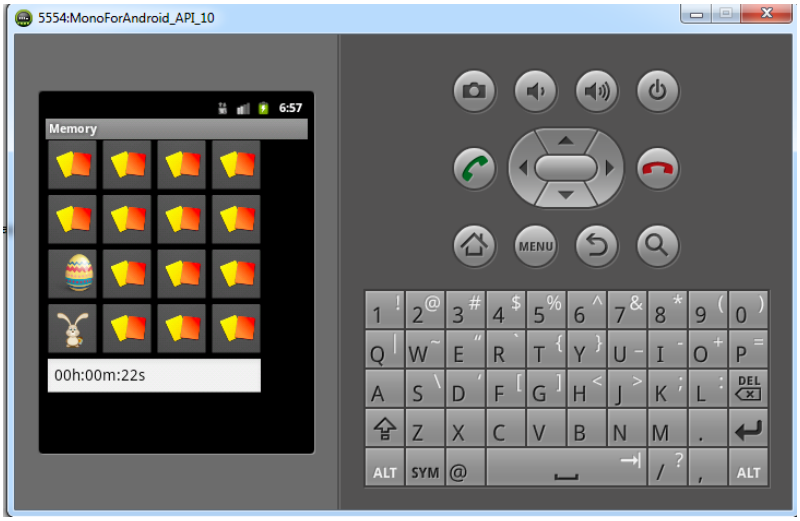

Figure 4 Nie pasujące karty w grze

3.3. Ekran w trakcie trwania gry

Celem użytkownika jest odkrycie wszystkich kart – aby to zrobić musi znajdywać kolejne pasujące pary kart. W moim przypadku są to karty wielkanocne – jako, że aplikacja została pisana w okresie Świąt Iwlkanocnych. Ikonki są bardzo podobne, uszy króliczków czy wzorki na pisankach niewiele się od siebie różnią. Jednak obrazki mogą być w każdej chwili zmienione – wystarczy zmienić pliki źródłowe – co za tym idzie można sterować poziomem trudności przez dobór obrazków i ich ilości.

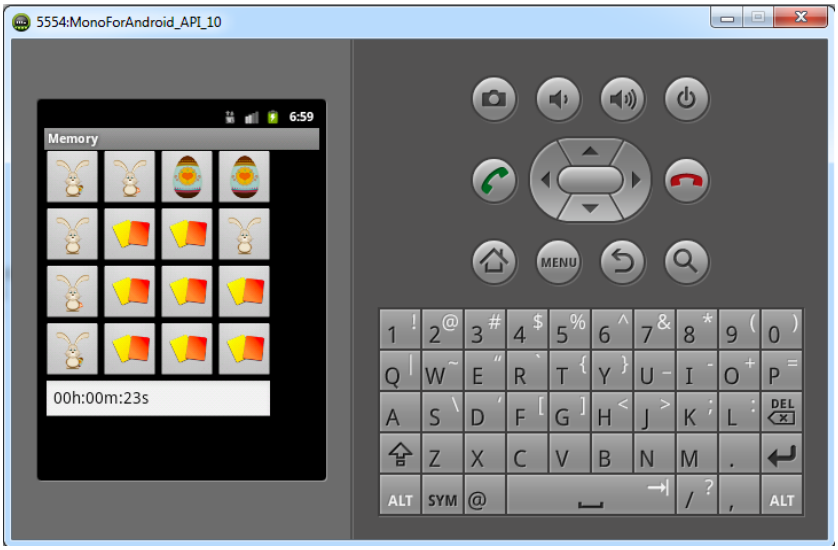

Figure 5 Ekran w trakcie trwania gry

3.4. Ekran w chwili zakończenia gry

Użytkownik kiedy wkońcu dopasuje wszystkie pary kart – wówczas licznik czasu się zatrzymuje oraz wszystki karty są odkryte – to znak zakończenia powodzeniem. W mojej grze celem jest osiągnięcie tego stanu – nie ma przegranej ☺

Figure 6 Ekran w chwili zakończenia gry

4. Specyfikacja wewnętrzna

4.1. Klasa *Card.cs*

Klasa modelująca pojedyńczą kartę. Wykorzystuję enuma stworzonego w klasie *CardState*.cs pozwalającego na wpowadzanie stanu odkrycia i zakrycia karty w talii.

Metoda `public Card(int value)` przyjmuję jako argument wartość karty jako int – każdej karcie przypisywany jest numeroraz na początku zawsze przyjmuje stan zakryty.

Poniższe metody nie zwracają wartości – określają jedynie stan karty:

- `public void Show() - metoda odkrywająca kartę w moemncie klikniecia na nią`
- `public void Hide() - metoda zakrywa kartę`

Metoda `public bool IsShown()` zwraca wartość true gdy karta jest odkryta I flase kiedy jest zakryta

Metoda `public int GetValue()` zwraca waktualną wartość karty.

4.2. Klasa *CardState.cs*

Wyliczenie stanów – są tylko dwa stany karty – zakryty bądź odkryty. Zastosowałam tutaj enum bo są tostałe, niezmienne wartości przypisywane do danej karty. Wykorzystywany w klasie modeulącej talię kart *Card*.cs

```
public enum CardState
  {
      Shown,
      Hiden

  }
```

4.3. Klasa *MainActivity.cs*

Głowna klasa w aplikacji – jako, że Xamarin został mapowany z Javy na .NET widzimy że klasa dziedziczy z klasy Activity – ogólnej klasy modelującej activity w aplikacjach na Androida.

Tworzę dwa stringi:

1. string cardSuiteImage = "card.png"; karta zakrywająca obrazek
2. string cardImage = "img?.png"; string ten symbolizuje daną kartę, znak zapytania zastępowany jest w kolejnych iteracjach liczbą całkowitą w trakcie tworzenia karty

Następnie wbudowana medota *protected override void OnCreate(Bundle bundle*), która startuje na samym początku budowania aplikacji. Dopisany został fragment kodu odpawiadający za tworzenie talii 16 kart oraz wymieszania wszystkich kart.

Talia tworzona jest jako lista – w Xamarinie jako lista javo'wa przyjmująca kolejne elementy jako karty modelowane w klasie *Card*.cs. Natępnie iterujemy do 8 jako, że kart chcemy otrzymać 16 ale jako 8 par tych samych zatem i określamy jako 8. W pętli tworzymy po dwie takie same karty i dodajemy je do talii. Po wyjściu z pętli wywołujemy metodę *Shuffle*

Metoda *public void Shuffle<T>(JavaList<T> list)* pełni zadanie wymieszania wszystkich kart w talii, w randomowy sposób. Jest to model przyjmujący argument T(u mnie będą to karty) mieszający wszystkie elementy listy(mojej talii). Metoda generyczna przyjmująca dowolny typ parametrów – w mojej klasie jest to typ stworzony przeze mnie czyli karty.

Następnie wywołujemy metodę klikająca na widok gry.

5. Podsumowanie projektu I wnioski

Niewątpliwie napisanie takiej gry było dla mnie przyjemnością – poznawanie nowych narzędzi jest rozwojowe, czasem trudne I wymaga poświecenia większej ilości czasu, jednak satysfakcja kiedy napisany program działa jest nieoceniona. Mimo, żę gra jest prosta, to napisanie jej pochłonęło nieco czasu – Xamarin, mimo, że wykorzystuje platformę .NET to opiera się na Javie i chociażby już zwykła prosta i przydatna lista w C# to tutaj implemntowana jest nieco inaczej ze względu na przestrzeń nazw. Samo uruchomienie wymagało nielada cierpliwości i wyczynu – emulator jest bardzo wony i często był problem z samym jego uruchomieniem. Mimo zaistniałych problemów praca nad tym projektem była przyjemnym czasem, rozwijającym, pokazującym podstawy tego wielkiego narzędzia jakim jest Xamarin. Grę podarowałam w postaci apk wielu znajomym, którzy w chwilach przerwy prześcigali się kto szybciej ułoży talię kart. Niestety po miesiącu już nie działała ze względu na licenjcę free oraz ważnośc sygnatury.