JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 25 (2022) 362–365
Received: 1 September 2022
Accepted: 29 September 2022

Comparison of the cross-platform mobile applications performance in 2D graphics processing

Porównanie wydajności wieloplatformowych aplikacji mobilnych w przetwarzaniu grafiki 2D

Adam Drzewiecki*

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract
The article presents a comparison of cross-platform frameworks in 2D graphics processing. The tests consisted in applying filters to 2D images, and then the image processing time and standard deviation were measured. The test will be conducted on the Huawei P8 and Samsung A50 phones. The comparison concerns two very popular cross-platform programming frameworks: Flutter and Xamarin.

Keywords: Flutter; Xamarin; cross-platform framework performance

Streszczenie
Artukuł przedstawia porównanie frameworków wieloplatformowych w przetwarzaniu grafiki 2D. Badania polegają na nakładaniu wybranych filtrów na obrazy 2D, a następnie jest mierzony czas nakładania dla danego filtra oraz obliczane odchylenie standardowe. Badanie zostanie przeprowadzone na telefonie Huawei P8 i Samsung A50. Porównanie dotyczy dwóch bardzo popularnych wieloplatformowych szkieletów programistycznych (framework), czyli odpowiednio Flutter oraz Xamarin.

Słowa kluczowe: Flutter; Xamarin; wydajność frameworków wieloplatformowych

*Corresponding author
Email address: adam.drzewiecki@pollub.edu.pl (A. Drzewiecki)

1. Wstęp

W dzisiejszych czasach frameworki wieloplatformowe stają się coraz bardziej popularne. Telefony komórkowe są znacznie szybsze niż kilka lat temu, dzięki czemu wydajność aplikacji napisanych w takich frameworkach jest porównywalna z wydajnością aplikacji natywnych. Wieloplatformowość pozwala na napisanie aplikacji tylko raz i taka aplikacja będzie mogła działać na wszystkich systemach mobilnych. Dzięki frameworkom wieloplatformowym można zaoszczędzić czas oraz pieniądze [1], ponieważ aplikacja może zostać napisana tylko raz i będzie działała na wszystkich systemach mobilnych. W artykule przedstawiono analizę wydajności frameworków wieloplatformowych z punktu widzenia przetwarzania grafiki 2D. Porównane zostaną frameworki Flutter oraz Xamarin. Do przeprowadzenia badań zostaną utworzone 2 takie same aplikacje w obu frameworkach. Badanie będzie polegało na nakładaniu filtrów na obrazy 2D, a następnie zostanie zmierzony czas przetwarzania grafiki oraz zostanie obliczone odchylenie standardowe.

2. Przegląd literatury

W artykule [1] przebadano wydajność frameworka Xamarin oraz aplikacji natywnej m.in. dla szybkości generowania interfejsu użytkownika. Stworzono 2 aplikacje, jedna z nich była aplikacją natywną napisaną w języku Java, a druga we frameworku Xamarin. Aplikacje zostały przetestowane na tym samym urządzeniu oraz zmierzono czas wykonywania poszczególnych operacji. Framework Xamarin okazał się gorszy z punktu widzenia wydajności niż aplikacja natywna.

W artykule [2] obiektem badań były narzędzia programistyczne służące do tworzenia aplikacji na systemie Android, zbadano Android SDK i język Java, Android NDK, Xamarin oraz Apache Cordova. Zbadano wydajność poszczególnych frameworków dla odczytu/zapisu dużych plików oraz dla sortowania tablicy. Wszystkie testy zostały przeprowadzone na urządzeniu fizycznym oraz emulatorze android. Na podstawie przeprowadzonych testów nie można jednoznacznie stwierdzić, które narzędzie pozwala tworzyć najwydajniejsze aplikacje.

W artykule [3] celem badań było porównanie wydajności frameworka Xamarin oraz języka Java na systemach Android oraz Windows 10 Mobile. Przetestowano szybkość uruchamiania aplikacji oraz szybkość odczytu i zapisu plików. Testy zostały przeprowadzone na emulatorze z systemem Android 7 oraz Windows 10 Mobile. Zebranie wyników badań na platformie Android wskazuje na wysokie spowolnienie działania spowodowane wykorzystaniem wieloplatformowej technologi, natomiast na systemie Windows 10 Mobile różnica wydajności pomiędzy aplikacją wieloplatformową oraz natywną była dużo mniejsza.

W artykule [4] obiektem badań było zbadanie wydajności przechwytywania obrazów oraz filtrów graficznych dla frameworka Xamarin. W celu przeprowadzenia badań, w środowisku Xamarin stworzona została aplikacja na systemy Android i Windows 10 Mobile (UWP). Aplikacje zostały zainstalowane na dwóch

urządzeniach fizycznych i zostały zmierzona wydajność aplikacji. Okazało się, że na czas wczytywania obrazu wpływ ma rozmiar pliku oraz zastosowanie filtru.

W artykule [5] porównano tworzenie aplikacji wykorzystując technologie natywne dla Androida oraz we frameworku Flutter. Stworzono 2 aplikacje, jedna z nich była aplikacją natywną a druga we frameworku Flutter. Do zbadania obciążenia procesora podczas operacji sortowania danych wykorzystano moduł Android Profiler oraz narzędzie DevTools. Aplikacja natywna okazała się bardziej wydajna niż aplikacja napisana we frameworku Flutter, technologia natywna miała również większe wsparcie społeczności oraz więcej dostępnych bibliotek.

3. Metoda badawcza

Badanie będzie polegało na zmierzeniu wydajności aplikacji mobilnej napisanej w różnych technologiach wieloplatformowych. Zbadany zostanie czas przetwarzania grafiki 2D oraz zmierzone odchylenie standardowe z pomiarów czasu.

Do przeprowadzenia badania zostaną utworzone dwie takie same aplikacje we frameworkach Flutter oraz Xamarin. Na obrazy będą nakładane różne filtry (np. Filtr Gaussa, Progowanie), a następnie będzie mierzony czas przetworzenia obrazu oraz odchylenie standardowe. Każdy pomiar będzie dodawany do listy, a następnie po przeprowadzeniu 100 pomiarów będzie można odczytać średni wynik dla danego filtra. Badanie zostanie przeprowadzone na 2 urządzeniach. Do przeprowadzenia badań został wybrany telefon ze starszą wersją androida o gorszych parametrach technicznych oraz telefon z najnowszą wersją Androida o lepszych parametrach. Taki zestaw urządzeń pokaże, jak aplikacje wieloplatformowe zachowują się na różnych urządzeniach oraz czy wybór technologii wieloplatformowej jest przyszłościowy.

3.1. Urządzenia badawcze

Pierwszym urządzeniem będzie telefon fizyczny Huawei P8 z systemem Android 6.0. Premiera tego telefonu miała miejsce w 2015 roku, jego parametry odbiegają od dzisiejszych standardów. Badanie pokaże, czy aplikacje wieloplatformowe są wydajne na starszych urządzeniach.

Drugim urządzeniem będzie telefon fizyczny Samsung A50 z systemem Android 11. To urządzenie jest najlepszym telefonem pod względem wydajności spośród wszystkich urządzeń badawczych. Premiera tego telefonu odbyła się w 2019 roku, w porównaniu do Huawei P8 jego parametry techniczne są dużo lepsze oraz ten telefon powinien przetwarzać grafikę wydajniej.

3.2. Scenariusze badawcze

Aby przeprowadzić badania opracowano 10 scenariuszy badawczych, które będą polegały na pomiarze czasu przetwarzania obrazu dla filtrów:

1. Skala szarości
2. Rozmycie Gaussa
3. Sepia
4. Wyostrzanie
5. Modulacja
6. Luminacja
7. Saturacja
8. Efekt Hue
9. Rozjaśnianie
10. Przyciemnianie

4. Aplikacje opracowane na potrzeby badań

Aby przeprowadzić badania, zostały utworzone 2 takie same aplikacje we frameworkach Flutter oraz Xamarin. Każda z aplikacji zawierała 2 ekrany. Na ekranie głównym możemy wybrać filtr oraz rozdzielczość obrazu, a na drugim ekranie nałożyć filtr na obraz oraz zmierzyć czas przetwarzania wybranego obrazu. Aplikacja we frameworku Xamarin została napisana w języku C# w środowisku Visual Studio, a aplikacja we frameworku Flutter w języku Dart w środowisku Visual Studio Code.

Dla każdego filtra badanie zostało przeprowadzone 30 razy, a następnie z pomiarów czasu została obliczona średnia oraz odchylenie standardowe. Aby obliczyć średnią oraz odchylenie standardowe, po przeprowadzeniu pomiaru wynik był dodawany do listy.

Rysunek 1: Aplikacja mobilna do przeprowadzenia badań.

5. Wyniki

Wyniki badań zostały umieszczone w poniższych tabelach. W każdej tabeli znajduje się średni czas przetwa-

rzania obrazu dla danego filtra oraz odchylenie standardowe ze zmierzonych wartości. Wyniki zostały podzielone na tabele dla każdego urządzenia badawczego.

5.1. Analiza wyników frameworka Xamarin

Tabela 1 oraz tabela 2 przedstawiają wyniki badań dla frameworka Xamarin dla wszystkich urządzeń badawczych. W pierwszej tabeli zostały umieszczone wyniki dla urządzenia Huawei P8.

Tabela 1: Średnie czasy nakładania filtrów dla frameworka Xamarin - Huawei P8 Lite

Nazwa filtra	Czas (ms)	Odchylenie (ms)
Skala szarości	28,2	6,9
Rozmycie Gaussa	33,7	7,2
Sepia	29,7	5,2
Wyostrzanie	30,1	6,2
Modulacja	28,8	7,4
Luminacja	26,6	5,0
Saturacja	27,4	6,4
Efekt hue	28,0	5,6
Rozjaśnianie	28,4	6,5
Przyciemnianie	28,2	5,9

Druga tabela przedstawia wyniki badań dla urządzenia Samsung A50. Możemy zobaczyć znaczną różnicę w czasie przetwarzania obrazu, telefon Samsung A50 przetwarzał grafikę prawie 10 razy szybciej niż telefon Huawei. To samo można stwierdzić o odchyleniu standardowym, wartości odchylenia standardowego są nawet 15-20 razy mniejsze w przypadku niektórych filtrów w porównaniu z urządzeniem Huawei.

Tabela 2: Średnie czasy nakładania filtrów dla frameworka Xamarin - Samsung A50

Nazwa filtra	Czas (ms)	Odchylenie (ms)
Skala szarości	2,1	0,4
Rozmycie Gaussa	3,2	0,3
Sepia	1,7	0,1
Wyostrzanie	1,8	0,3
Modulacja	1,2	0,2
Luminacja	1,4	0,3
Saturacja	2,1	0,1
Efekt hue	2,2	0,4
Rozjaśnianie	2,4	0,4
Przyciemnianie	1,7	0,2

5.2. Analiza wyników frameworka Flutter

W tabeli 3 oraz tabeli 4 zostały umieszczone wyniki dla frameworka Flutter. W porównaniu do frameworka Xamarin, framework Flutter przetwarzał grafikę 3-4 razy szybciej. Wartości odchylenia standardowego są podobne dla obu frameworków i wynoszą około 30% wartości czasu. Tabele zawierają wyniki dla obu urządzeń badawczych.

Tabela 3: Średnie czasy nakładania filtrów dla frameworka Xamarin - Huawei P8 Lite

Nazwa filtra	Czas (ms)	Odchylenie (ms)
Skala szarości	6,3	1,5
Rozmycie Gaussa	6,1	2,3
Sepia	4,7	1,2
Wyostrzanie	5,5	1,3
Modulacja	7,2	1,5
Luminacja	6,1	2,0
Saturacja	7,1	2,1
Efekt hue	4,9	1,7
Rozjaśnianie	5,4	1,5
Przyciemnianie	5,6	1,4

Druga tabela przedstawia wyniki badań dla urządzenia Samsung A50. Można zauważyć znaczną różnicę w czasie przetwarzania obrazu na korzyść urządzenia Samsung A50, różnice wartości odchylenia standardowego są także zauważalne.

Tabela 4: Średnie czasy nakładania filtrów dla frameworka Xamarin - - Samsung A50

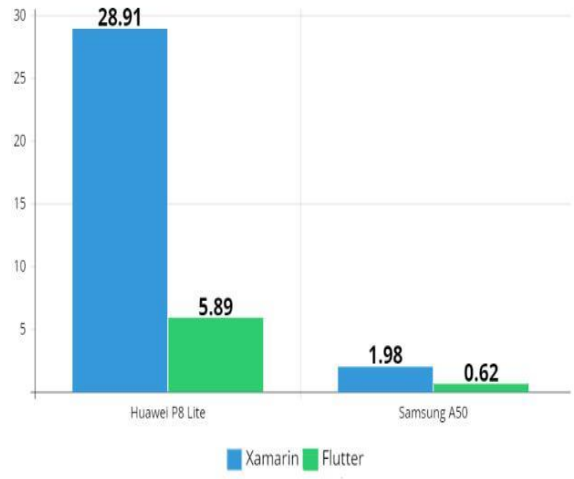
Nazwa filtra	Czas (ms)	Odchylenie (ms)
Skala szarości	0,4	0,1
Rozmycie Gaussa	1,1	0,1
Sepia	0,7	0,1
Wyostrzanie	0,3	0,2
Modulacja	0,5	0,1
Luminacja	0,8	0,2
Saturacja	0,6	0,2
Efekt hue	0,3	0,1
Rozjaśnianie	0,8	0,2
Przyciemnianie	0,7	0,2

6. Dyskusja

6.1. Interpretacja wyników

Z danych w tabelach z wynikami można wyciągnąć kilka wniosków:

- Wszystkie filtry nakładają się w podobnym czasie dla tego samego obrazu oraz urządzenia. Można stwierdzić, że samo przetwarzanie pikseli obrazu zajmuje niewiele czasu, natomiast najwięcej czasu zajmuje wyświetlenie obrazu ,
- Można zauważyć różnicę pomiędzy urządzeniami badawczymi. Na nowszym urządzeniu badawczym (Samsung A50) filtry nakładały się szybciej niż na starszym telefonie,
- Framework Flutter okazał się bardziej wydajny niż Xamarin. We Flutterze wszystkie filtry nakładały się 3-4 razy szybciej niż w Xamarinie,
- Odchylenie standardowe dla niektórych filtrów wyniosło nawet 50% wartości czasu przetwarzania dla danego filtru, więc wartości czasów cechował silny rozrzut od średniej.



Rysunek 2: Porównanie średnich ze wszystkich filtrów dla danego frameworka i urządzenia badawczego.

Powyższy wykres przedstawia średnią ze wszystkich filtrów dla danego frameworka i dla każdego urządzenia badawczego. Widać wyraźną różnicę pomiędzy frameworkiem Flutter oraz Xamarin. Największa różnica jest widoczna na starszym urządzeniu Huawei P8, gdzie framework Flutter przetwarzał grafikę 5 razy szybciej. Mniejsza różnica była na urządzeniu Samsung A50, gdzie Flutter przetwarzał grafikę 3 razy szybciej. Należy również zwrócić uwagę na różnicę wartości tych pomiarów. Na urządzeniu Samsung ta różnica wynosi tylko 1,36ms, a na Huawei P8 aż 23.02ms. Z punktu widzenia użytkownika korzystającego z aplikacji, 1,36ms może być różnicą praktycznie nie zauważalną, natomiast różnicę 23,02ms była odczuwalna w trakcie wykonywania badań, co oznacza, że framework Flutter będzie dużo lepszym wyborem jeśli aplikacja będzie wspierała stare urządzenia.

6.2. Weryfikacja hipotez

W artykule zostały postawione 2 hipotezy:

- Framework Flutter cechuje się wyższą wydajnością w przetwarzaniu grafiki 2D wśród obu frameworków wieloplatformowych,
- Urządzenia z nowszą wersją systemu Android przetwarzały grafikę wydajniej niż wersje ze starszą wersją system.

Po przeanalizowaniu wyników badań można stwierdzić, że obie hipotezy zostały spełnione. Framework Flutter przetwarzał grafikę wydajniej niż framework Xamarin. Widać również różnicę pomiędzy urządzeniami badawczymi, urządzenie z nowszą wersją Androida przetwarzało grafikę wydajniej niż urządzenie ze starszą wersją systemu. Firma Google co roku wprowadza szereg usprawnień w nowszych wersjach Androida, co ma duży wpływ na wydajność aplikacji. Należy pamiętać, że nowsze wersje systemu Android są zainstalowane na nowych urządzeniach, które bardzo często mają lepsze parametry techniczne niż urządzenia ze starszymi wersjami Androida.

6.3. Porównanie z przeglądem

Wyniki badań są bardzo podobne do tych, które można znaleźć w artykułach z przeglądu literatury.

W artykule [1] porównano wydajność frameworka Xamarin oraz aplikację natywną dla systemu Android. Po przeprowadzeniu badań okazało się, że framework Xamarin okazał się gorszy z punktu widzenia wydajności niż rozwiązanie natywne.

W artykule [5] porównano wydajność frameworka Flutter z aplikacją natywną dla systemu Android. Wyniki badań pokazały, że aplikacja napisana we Flutterze oraz aplikacja natywna cechują się podobną wydajnością. Na podstawie 2 powyższych artykułów można stwierdzić, że framework Flutter cechuje się wyższą wydajnością niż framework Xamarin, ponieważ Flutter wyróżnił się wyższą wydajnością niż Xamarin w porównaniu do aplikacji natywnej.

7. Wnioski

Na podstawie przeprowadzonych badań można wysnuć następujące wnioski:

- Framework Flutter cechuje się większą wydajnością w przetwarzaniu grafiki 2D niż framework Xamarin, wynika to najprawdopodobniej z architektury tego frameworka. Flutter do renderowania interfejsu używa silnika Skia, natomiast Xamarin korzysta ze środowiska Mono, które konwertuje elementy layoutu do kontrolek natywnych, co zajmuje więcej czasu niż renderowanie interfejsu we Flutterze,
- Urządzenia z nowszą wersją systemu Android przetwarzały grafikę wydajniej niż wersje ze starszą wersją systemu.

Literatura

[1] I. Bodia, M. Plechawska-Wójcik, Performance comparison of the Xamarin platform and native applications for Android operating system, Journal of Computer Sciences Institute, 5 (2017) 208-212, https://doi.org/10.35784/jcsi.624

[2] P. Kotarski, K. Śledź, J. Smołka, Analysis of the impact of development tools used on the performance of the mobile application, Journal of Computer Sciences Institute 6 (2018) 68-72, https://doi.org/10.35784/jcsi.642

[3] D. Wieczorek, J. Smołka, Comparison of performance multi-platform application core on Android and Windows 10 Mobile, Journal of Computer Sciences Institute 6 (2018) 87- 91. https://doi.org/10.35784/jcsi.647

[4] M. Dras, G. Fila, M. Plechawska-Wójcik, Analysis of Xamarin capabilities for building mobile multi-platform applications, Journal of Computer Sciences Institute 7 (2018) 183-190, https://doi.org/10.35784/jcsi.675

[5] D. Gałan, K. Fisz, P. Kopniak, A multi-criteria comparison of mobile applications built with the use of Android and Flutter Software Development Kits, Journal of Computer Sciences Institute 19 (2021) 107-113, https://doi.org/10.35784/jcsi.2614