ISSN: 2181-4171

Engineering Problems and Innovations

Vol. 03 Iss. 02 | | April. 27th, 2025

Том - 03 | | Вып. - 02
Tom - 03 | | Son. – 02

Проблемы инженерии и Инновации

Muhandislik muammolari va innovatsiyalar



SCAN ME

Google Scholar
INDEX COPERNICUS INTERNATIONAL
zenodo
Academic Resource Index ResearchBib

Главный редактор

Зокиров Санжар Икромжон угли д.ф.ф.-м.н., PhD

Журнал зарегистрирован 28 декабря 2022 года Агентством информации и массовых коммуникаций при Администрации Президента Республики Узбекистан

Сертификат № 055940
ISSN 2181-4171

Языки издания: Русский, английский, узбекский

Председатель эксп. совета

Ф.М.Мухтаров - д.ф.т.н, PhD. директор ФФ ТУИТ

Ответственные редакторы разделов

Т.М.Абдуллаев
д.ф.т.н., PhD, ФФ ТУИТ

Н.И.Иброхимов
д.ф.ф.-м.н., PhD, ФФ ТУИТ

А.Р.Нурматов
PhD, д.ф.ист.н.,НамГУ

Г.А.Умарова
к.п.н., доц., АндМИ

Корректоры

Порубай Оксана Витальевна
Лазарева Марина Викторовна
Обиджонов Зафаржон Одилжон угли

Редакционная коллегия:

Б.К.Сарсенбаев - *д.т.н, проф., акад., дейст. член Нац. Акад. Респ. Казахстан*
В.Ю.Рудь - *д.ф.-м.н., Физико-технический институт им. А.Ф.Иоффе, Санкт-Петербург, Россия*
Н.Б.Бейсенханов - *д.ф.-м..н., проф. АО "Казахстанско-Британский Технический Университет", Алматы, Казахстан*
Б.Р.Исмаилов - *д.т.н., проф. Южно-Казахстанский университет им. М. Ауэзова , Шымкент, Казахстан*
А.В.Аржанников - *д.ф.-м.н., Новосибирский государственный университет, Институт ядерной физики СО РАН, Новосибирск, Россия*
А.М.Касимахунова - *д.т.н., проф. ФерПИ, Фергана, Узбекистан*
И.Бйорно - к.*н.,PhD, Имеритус, Датский технический университет, Копенгаген, Дания*
Л.К.Мамадалиева - *д.т.н., проф. ФерПИ, Фергана, Узбекистан*
Ж.Т.Айменов - *д.т.н., проф. Южно-Казахстанский университет им.М.Ауезова, Шымкент, Казахстан*
Х.Б.Исмаилов - *к.т.н. доцент, Южно-Казахстанский университет им. М. Ауэзова , Шымкент, Казахстан*
В.В.Краснощеков - *к.т.н. Санкт-Петербургский политехнический университет Петра Великого, Санкт-Петербург, Российская Федерация*
М.Д.Шабданов - *к.т.н. ОшТУ , Ош, Киргизстан*
В.М.Голубова - *к.псх.н. Ростовский институт (филиал) ВГУЮ (РПА Минюста России), Ростов-на-Дону, Россия*

Коллегия экспертного совета

О.Х.Отакулов - *к.т.н., доц. Ферганский филиал ТУИТ, Узбекистан*
О.С.Райимжонова - *д.ф.т.н., PhD, доц, Ферганский филиал ТУИТ, Узбекистан*
Р.А.Нурдинова - *д.ф.т.н., PhD, доц., Ферганский филиал ТУИТ, Узбекистан*
М.Э.Юнусалиев - *к.т.н., PhD, доц., ФерПИ, Узбекистан*
М.И.Исмоилов - *к.т.н., PhD, доц., ФерПИ, Узбекистан*
М.Т.Халилов - *к.т.н., доц., АндМИ, Узбекистан*
Г.А.Дадамирзаева - *д.ф.ист.н., PhD, доц., НамГУ, Узбекистан*
И.Т.Тожибоев - *к.ф.-м.-н., доц., ФерГУ, Узбекистан*
А.Ш.Тухтабаев - *д.ф.ист.н., PhD, доц., НамГУ, Узбекистан*
И.М.Исакова - *д.п.н. доц., ФерГУ, Узбекистан*
М.М.Турдиматов - *к.т.н, доц., Ферганский филиал ТУИТ, Узбекистан*
Д.Ф.Тухтасинов - *д.ф.пед.н., PhD, доц., Ферганский филиал ТУИТ, Узбекистан*
Н.Д.Ботирова - *д.ф.пед.н., PhD, доц., Ферганский филиал ТУИТ, Узбекистан*
Ш.Ж.Холматов - *к.ф.н., доц., Ферганский филиал ТУИТ, Узбекистан*

При использовании материалов журнала необходимо указывать источник.
Авторы статей несут ответственность за содержание статей и за сам факт их публикации.
Редакция не всегда разделяет мнение авторов и не несет ответственности за недостоверность опубликованных данных.
Редакция журнала не несет никакой ответственности перед авторами и/или третьими лицами и организациями за возможный ущерб, причиненный публикацией статьи.

Editor in chief

Zokirov Sanjar Ikromjon ugli
PhD in Phys. and Math. Sc.

The journal has been registered on December 28, 2022 by the Agency for Information and Mass Communications under the Administration of the President of the Republic of Uzbekistan

Certificate No. 055940
ISSN 2181-4171

Languages:
Russian, English, Uzbek

Chairman of exp. advice

F.M.Muxtarov - Ph.D. in tech. Sc., director of FBTUIT

Responsible section editors

T.M.Abdullaev
PhD in Tech. Sc., FBTUIT

N.I.Ibrokhimov
PhD in phys. and math. Sc., FB of TUIT

A.R.Nurmatov
Ph.D. in hist. sc., NamSU

G.A.Umarova
Cand. of ped. sc, assoc. prof., AndMI

Proofreaders
Porubay Oksana Vitalevna
Lazareva Marina Viktorovna
Obidjonov Zafarjon Odiljon ugli

Editorial board:

B.K.Sarsenbaev– *Dr. of Tech. Sc., prof. Actual member of the National Acad. Rep. Kazakhstan*
V.Yu.Rud'– *Dr. of Phys. and Math. Sc., Ioffe Physicotechnical Institute, St.-Petersburg, Russia*
N.B.Beisenkhanov– *Dr. of Phys and Math. Sc. (Ds), full professor in Physics, JSC "Kazakh-British Technical University", Almaty, Kazakhstan*
B.R.Ismailov– *Dr. of Tech. Sc., prof., Mukhtar Auezov South Kazakhstan State University, Shymkent, Kazakhstan*
A.V.Arzhannikov– *Dr. of Sc. in Phys. and Math., prof., Novosibirsk St. University, Budker Institute of Nuclear Physics, Russian Federation*
A.M.Kasimahunova - *Dr. of Tech. Sc., prof., Fergana Polytechnic Institute. Fergana, Uzbekistan*
I.Bjørnø - *Assoc. Prof. PH.D, DTU imeritus, København, Denmark*
L.K.Mamadaliyeva - *Dr. of Tech. Sc., prof., Fergana Polytechnic Institute. Fergana, Uzbekistan*
Zh.T.Aimenov– *Dr. of Engineering (Ds), prof. Mukhtar Auezov South Kazakhstan State University, Shymkent, Kazakhstan*
Kh.B.,Ismailov - *PhD in Tech. Sc., Mukhtar Auezov South Kazakhstan State University, Shymkent, Kazakhstan*
V.V.Krasnoshchekov - *PhD in Engineering, Peter the Great St. Petersburg Polytechnic University, Russian Federation, St. Petersburg*
M.D.Shabdanov– *Can. of Tech. Sc., Osh Technological University, Osh, Kyrgyzstan*
V.M.Golubova - *PhD in Psych., Rostov Institute (branch) VGUYu (RPA of the Ministry of Justice of Russia) Russia, Rostov-on-Don*

Expert Board

O.Kh.Otakulov– *Cand. of tech. sc., assoc. prof., Fergana branch of TUIT, Uzbekistan*
O.S.Raimzhonova - *PhD, assoc. prof., Fergana branch of TUIT, Uzbekistan*
R.A.Nurdinova - *PhD, assoc. prof., Fergana branch of TUIT, Uzbekistan*
E.M.Yunusaliev - *Ph.D in Technics, assoc. prof., FerPI, Uzbekistan*
M.M.Ismoilov - *Cand. of tech. sc., PhD, assoc. prof., FerPI, Uzbekistan*
M.T.Khalilov - *Cand. of tech. sc., assoc. prof., AndMI, Uzbekistan*
G.A.Dadamirzaeva - *Ph.D in Hist. Sc., NamSU, Uzbekistan*
I.T.Tojiboev - *Cand. of Phys. and Math. Sc., assoc. prof., FerSU, Uzbekistan*
A.Sh.Tukhtabaev - *Ph.D in Hist. Sc., NamSU, Uzbekistan*
M.T.Isakova - *DSc in Psychology, FerSU, Uzbekistan*
M.M.Turdimatov - *Cand. of tech. sc., PhD, assoc. prof., Fergana branch of TUIT, Uzbekistan*
D.F.Tukhtasinov - *PhD on Ped., assoc. prof., Fergana branch of TUIT, Uzbekistan*
N.D.Botirova - *PhD on Ped., assoc. prof., Fergana branch of TUIT, Uzbekistan*
Sh.J.Kholmatov - *cand. phil., assoc. prof., Fergana branch of TUIT, Uzbekistan*

When using journal materials, you must indicate the source.
The authors of the articles are responsible for the content of the articles and for the very fact of their publication.
The editors do not always share the opinions of the authors and are not responsible for the unreliability of the published data.
The editorial board of the journal does not bear any responsibility to the authors and / or third parties and organizations for possible damage caused by the publication of the article.

Кросс-платформенная разработка на .Net MAUI: производительность, нативные возможности и обратная совместимость с Xamarin

КАРАБАЕВ САМАТ ЭШПОЛОТОВИЧ, ЖУМАБАЙ УУЛУ МИРБЕК, САМИДИНОВ БЕКДООЛОТ ЗАМИРБЕКОВИЧ, ЖЕНИШБЕК КЫЗЫ ГУЛЛИРА

Ошский государственный университет

Copyright: © 2025 by the author.
Licensee: This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https:// creativecommons.org/licenses/by/ 4.0/).

Аннотация: *В данной статье рассматриваются возможности и особенности кросс-платформенной разработки на .NET MAUI (Multi-platform App UI), его отличия от предшественника Xamarin, а также влияние на производительность и поддержку нативных функций мобильных платформ. Анализируются ключевые преимущества .NET MAUI, такие как улучшенная производительность, единый проект для всех платформ, нативные API и более удобная модель взаимодействия с кодом C#. Также рассматриваются вопросы обратной совместимости с Xamarin, включая миграцию проектов и ограничения, с которыми могут столкнуться разработчики. В статье представлены сравнительные тесты производительности и анализ применимости .NET MAUI для корпоративных и потребительских приложений.*

Ключевые слова: *.NET MAUI, Xamarin, кросс-платформенная разработка, производительность, нативные API, миграция, C#.*

Cross-platform development with .Net MAUI: performance, native capabilities, and backward compatibility with Xamarin

Karabaev Samat Eshpolotovich,Zhumabai uulu Mirbek, Samidinov Bekdoolot Zamirbekovich, Jenishbek kyzy Gullira

Abstract: *This article examines the capabilities and features of cross-platform development with .NET MAUI (Multi-platform App UI), its differences from the predecessor Xamarin, as well as its impact on performance and support for native mobile platform functionalities. Key advantages of .NET MAUI, such as improved performance, a single project structure for all platforms, native API access, and a more convenient interaction model with C#, are analyzed. The article also explores backward compatibility with Xamarin, including project migration and the limitations developers may encounter. Comparative performance tests and an analysis of the applicability of .NET MAUI for enterprise and consumer applications are presented.*

Keywords: *.NET MAUI, Xamarin, cross-platform development, performance, native APIs, migration, C#.*

Введение

С развитием мобильных технологий и увеличением числа платформ для развертывания приложений растёт спрос на универсальные решения, позволяющие создавать один код для различных операционных систем. На протяжении последних лет Xamarin был одной из ключевых технологий для кроссплатформенной мобильной разработки в экосистеме .NET. Однако, несмотря на свою популярность, Xamarin имел ряд ограничений: сложность проектирования UI, необходимость управления несколькими проектами и проблемы с производительностью. В ответ на эти вызовы компания Microsoft представила .NET MAUI (Multi-platform App UI) — новую платформу для кросс-платформенной разработки, которая призвана заменить Xamarin. Основное преимущество .NET MAUI заключается в единой кодовой базе для всех целевых платформ (Android, iOS, macOS, Windows), что позволяет разработчикам писать меньше шаблонного кода и сосредотачиваться на бизнес-логике приложения [6].

Методы.

Ключевые особенности .NET MAUI по сравнению с Xamarin

Один проект вместо нескольких: в Xamarin было необходимо создавать отдельные проекты для каждой платформы, в то время как .NET MAUI использует единое решение.

Улучшенная производительность: .NET 6 и последующие версии оптимизированы, что позволяет приложениям работать быстрее, чем их аналоги на Xamarin.

Упрощённый доступ к нативным API: .NET MAUI предлагает более удобный механизм взаимодействия с API Android и iOS, устраняя необходимость в зависимости от платформенных библиотек.

Гибкая настройка UI: новая модель Handlers заменяет устаревшие Renderers, обеспечивая лучшую производительность и поддержку кастомных интерфейсов.

В данной статье проводится подробный анализ возможностей .NET MAUI, оценивается его производительность, сравниваются подходы к работе с нативными API и рассматриваются способы миграции с Xamarin.

Результаты

Развитие технологий кросс-платформенной мобильной разработки всегда сопровождалось дилеммой между удобством написания кода и производительностью конечного приложения. В течение многих лет одним из самых популярных решений для создания кросс-платформенных мобильных приложений в экосистеме .NET был Xamarin — фреймворк, который позволял писать мобильные приложения на C# и запускать их на iOS и Android, используя общий код и доступ к нативным API[3]. Однако, несмотря на свою популярность, Xamarin имел ряд существенных недостатков, которые ограничивали его возможности. Одной из главных проблем было наличие нескольких отдельных проектов для разных платформ, что увеличивало сложность разработки и сопровождения кода. Кроме того, модель рендеринга интерфейсов с использованием Custom Renderers приводила к проблемам с производительностью и усложняла работу с UI-элементами.

С выходом .NET 6 компания Microsoft представила .NET MAUI (Multi-platform App UI) — современную замену Xamarin, призванную устранить его ограничения и обеспечить разработчикам более удобную, гибкую и производительную платформу. В отличие от Xamarin, .NET MAUI позволяет использовать единый проект для всех целевых платформ, что значительно упрощает управление кодом и сокращает дублирование. В новой архитектуре платформы были полностью переработаны механизмы работы с пользовательским интерфейсом: вместо устаревшей системы Renderers была введена новая модель Handlers, которая значительно повышает производительность, ускоряет отрисовку UI и упрощает кастомизацию элементов управления.

Одним из главных преимуществ .NET MAUI является его нативность. Хотя технология остаётся кросс-платформенной, она предоставляет прямой доступ к API операционных систем, что позволяет разработчикам легко взаимодействовать с камерами, сенсорами, Bluetooth, GPS, файловой системой и другими аппаратными возможностями устройства. В Xamarin для этого использовались сторонние библиотеки или специальные механизмы зависимостей, что увеличивало сложность разработки. В .NET MAUI многие функции интегрированы непосредственно в платформу, что делает их использование более удобным [1].

Кроме того, .NET MAUI поддерживает архитектурные принципы MVU (Model-View-Update), позволяя разработчикам применять современные методологии построения интерфейсов. Это особенно полезно в сложных проектах, где требуется высокая степень реактивности интерфейса. В сочетании с поддержкой Blazor Hybrid, .NET MAUI позволяет объединить веб-технологии с нативной производительностью, предоставляя разработчикам возможность использовать Razor-компоненты для построения UI.

Не менее важным аспектом является производительность. Microsoft провела ряд оптимизаций, позволяющих .NET MAUI значительно превосходить Xamarin в скорости загрузки и рендеринга интерфейсов. По результатам тестов, .NET MAUI демонстрирует сокращение времени загрузки на 30–40% по сравнению с аналогичными приложениями на Xamarin.Forms. Это достигается благодаря тому, что большая часть тяжёлых операций перенесена на более низкий

уровень, а обновлённая архитектура Handlers исключает ненужные промежуточные слои, снижая накладные расходы на отрисовку элементов UI.

Однако, несмотря на значительные улучшения, у .NET MAUI остаются определённые ограничения, связанные с процессом миграции с Xamarin. Хотя большинство базовых компонентов поддерживаются, устаревшие API, такие как DependencyService и Effects, требуют замены на новые аналоги. Также разработчики могут столкнуться с необходимостью частичной переработки UI-кода, особенно если в Xamarin использовались кастомные рендереры. Microsoft предлагает ряд инструментов и рекомендаций для упрощения миграции, но в реальных проектах этот процесс может занять значительное время, особенно если приложение содержит большое количество сложных UI-элементов. Несмотря на это, .NET MAUI представляет собой логичное эволюционное развитие Xamarin, предлагая более современную и удобную платформу для кросс-платформенной разработки. В сочетании с мощными возможностями C# и .NET, поддержкой облачных технологий и глубокой интеграцией с инструментами Microsoft, .NET MAUI становится привлекательным выбором как для небольших стартапов, так и для крупных корпоративных решений. Одним из ключевых аспектов .NET MAUI является его интеграция с нативными API мобильных платформ. В отличие от Xamarin, где доступ к функциям Android и iOS требовал использования DependencyService или платформенно-зависимых библиотек, в .NET MAUI разработчики могут работать с нативными API напрямую. Это означает, что теперь можно вызывать методы Android SDK или iOS UIKit без дополнительных обёрток, что значительно упрощает разработку и повышает производительность за счёт уменьшения количества промежуточных слоёв. Для работы с платформенными возможностями .NET MAUI предоставляет новый механизм .NET MAUI Essentials, который объединяет основные API, такие как управление сенсорами, файловая система, геолокация, доступ к камере, Bluetooth, а также работа с сетью и push-уведомлениями. Например, получение координат устройства в Xamarin требовало создания отдельной платформенной реализации, тогда как в .NET MAUI достаточно использовать встроенный AP

```
var location = await Geolocation.GetLastKnownLocationAsync();
```

Этот подход значительно упрощает кросс-платформенную разработку и сокращает необходимость писать специфический код для каждой платформы. Производительность .NET MAUI по сравнению с Xamarin также была значительно улучшена. Разработчики часто сталкивались с тем, что приложения, созданные на Xamarin.Forms, демонстрировали низкую частоту кадров (FPS) и задержки при рендеринге интерфейсов, особенно на слабых устройствах. Причиной этого была устаревшая система рендеринга UI — Custom Renderers, где каждый UI-элемент требовал своей реализации для каждой платформы. В .NET MAUI эта проблема решена за счёт механизма Handlers, который заменил Renderers и позволил избавиться от лишних слоёв между UI-компонентами и нативным рендерингом. Рассмотрим простой пример создания кнопки в Xamarin и .NET MAUI. В Xamarin для изменения внешнего вида кнопки требовалось использовать кастомные рендереры

```
[assembly: ExportRenderer(typeof(MyCustomButton), typeof(MyButtonRenderer))]
public class MyButtonRenderer : ButtonRenderer
    {
protected override void OnElementChanged(ElementChangedEventArgs <Button> e)
    {
      base.OnElementChanged(e);
      if (Control != null)
      {

Control.SetBackgroundColor(Android.Graphics.Color.Red);
      }
    }
```

В .NET MAUI для аналогичной задачи можно использовать Handlers, что значительно упрощает код

```
Microsoft.Maui.Handlers.ButtonHandler.Mapper.AppendToMapping("Custom", (handler, view) =>
{

handler.PlatformView.SetBackgroundColor(Android.Graphics.Color.Red);
});
```

Этот новый подход не только облегчает поддержку и кастомизацию UI, но и повышает производительность, так как теперь .NET MAUI использует прямые вызовы к нативным компонентам без необходимости дополнительных обёрток [2].

Преимущество нового механизма рендеринга также выражается в уменьшении потребления оперативной памяти и снижении времени загрузки интерфейса. Согласно тестам Microsoft, переход от Renderers к Handlers позволил сократить время рендеринга на 30–40% и снизить потребление памяти примерно на 20% по сравнению с Xamarin. Это особенно важно для мобильных устройств с ограниченными ресурсами, где каждая миллисекунда задержки влияет на восприятие пользователем плавности работы приложения. Стоит также отметить, что .NET MAUI поддерживает Blazor Hybrid, что позволяет разработчикам комбинировать нативные компоненты с Blazor-компонентами, созданными с использованием Razor-шаблонов и C#. Это открывает новые возможности для унификации UI в мобильных, веб- и десктопных приложениях, где одна и та же кодовая база может использоваться в разных средах. Такой подход становится особенно привлекательным для корпоративных решений, где требуется единый UI-стиль во всех приложениях компании.

Обсуждение

Несмотря на все преимущества .NET MAUI, процесс миграции с Xamarin остаётся одной из ключевых тем для обсуждения. Многие компании уже имеют большие проекты, написанные на Xamarin.Forms, и переход на новую платформу требует определённых затрат времени и ресурсов. Microsoft предоставляет инструменты для автоматической конвертации проектов, но не все API из Xamarin поддерживаются в .NET MAUI в неизменном виде. Наиболее проблемными местами при миграции являются:

Устаревшие API (например, DependencyService), которые заменяются новыми механизмами.

Различия в рендеринге UI, что требует частичного изменения интерфейсов.

Изменённые механизмы работы с файлами и доступом к ресурсам системы.

Несмотря на эти сложности, переход на .NET MAUI в долгосрочной перспективе является логичным шагом для всех, кто работает с кросс-платформенной мобильной разработкой в экосистеме .NET. Microsoft продолжает активно развивать платформу, а с выходом .NET 8 ожидаются новые оптимизации, которые ещё больше увеличат производительность и удобство разработки.

Работа с платформенно-специфичными UI-компонентами в .NET MAUI является одним из важнейших аспектов разработки кросс-платформенных мобильных приложений, поскольку каждое приложение, независимо от его унифицированного кода, всё равно должно учитывать особенности и специфику платформ, на которых оно работает. В отличие от Xamarin.Forms, где рендеринг пользовательского интерфейса осуществлялся через систему Custom Renderers, в .NET MAUI внедрена новая архитектура Handlers, которая упрощает интеграцию платформенно-специфичных компонентов, а также даёт разработчикам возможность напрямую взаимодействовать с нативными элементами пользовательского интерфейса Android, iOS, macOS и Windows.

Ранее, в Xamarin.Forms, добавление нативных элементов UI было связано с необходимостью создавать отдельные реализации рендереров для каждой платформы. Например, если необходимо было использовать UIButton в iOS или AppCompatButton в Android, приходилось

писать платформенный код, наследуясь от ViewRenderer<>, что усложняло поддержку и сопровождение. В .NET MAUI эта задача решается более элегантно, поскольку теперь разработчики могут использовать Handlers для управления свойствами и поведением элементов интерфейса на каждой платформе. Вместо того чтобы создавать сложные рендереры, разработчики могут переопределять обработчики и изменять нативные элементы на лету.

Рассмотрим пример работы с кнопкой, в котором для iOS мы будем использовать UIButton, а для Android — AppCompatButton. В .NET MAUI кастомизация платформенных UI-компонентов выполняется следующим образом:

```
Microsoft.Maui.Handlers.ButtonHandler.Mapper.AppendToMapping("Custom", (handler, view) =>
{
#if ANDROID
andler.PlatformView.SetBackgroundColor(Android.Graphics.Color.Blue);
#elif IOS

handler.PlatformView.SetTitleColor(UIKit.UIColor.Red, UIKit.UIControlState.Normal);
#endif
});
```

Этот код добавляет платформенно-специфичную обработку для элементов UI без необходимости использовать громоздкие рендереры. В этом случае для iOS устанавливается красный цвет текста кнопки, а для Android — синий фон. Этот подход делает код чище, удобнее для поддержки и более гибким при развертывании на нескольких платформах. Одним из значительных преимуществ нового подхода является поддержка платформенно-специфичных UI-фреймворков, таких как SwiftUI для iOS, Jetpack Compose для Android, WinUI для Windows и MacCatalyst для macOS. Это означает, что разработчики теперь могут не только модифицировать существующие UI-компоненты, но и интегрировать полностью нативные интерфейсы в MAUI-приложения.

Для примера рассмотрим сценарий, когда требуется в iOS встроить SwiftUI View внутри .NET MAUI. В классическом подходе Xamarin.Forms это требовало бы написания сложного рендерера и взаимодействия через UIViewController. В .NET MAUI процесс значительно упрощается за счёт использования UIViewRepresentable, который позволяет интегрировать SwiftUI в UIKit. В .NET MAUI это можно реализовать следующим образом:

```
public class CustomiOSView : UIViewController
{
  public override void ViewDidLoad()
  {
    base.ViewDidLoad();
    var hostingController = new UIHostingController<CustomSwiftUIView>(new CustomSwiftUIView());

AddChildViewController(hostingController);

View.AddSubview(hostingController.View);

hostingController.DidMoveToParentViewController(this);
  }
}
```

Этот код позволяет встраивать SwiftUI-компоненты в MAUI-приложение, сохраняя преимущества нативного рендеринга и при этом продолжая использовать .NET в качестве основной логики приложения.

Аналогично, для Android можно интегрировать Jetpack Compose. В Android традиционно интерфейсы создавались с помощью XML-макетов, однако современные приложения переходят к декларативному UI через Jetpack Compose. В .NET MAUI теперь можно использовать ComposeView, чтобы встраивать элементы Jetpack Compose прямо в MAUI-приложение:

```
#if ANDROID
public class CustomAndroidView : AndroidX.Activity.ComponentActivity
{
  protected override void OnCreate(Bundle savedInstanceState)
  {
    base.OnCreate(savedInstanceState);
```

```
        SetContentView(new ComposeView(this)
        {
            ViewCompositionStrategy = ViewCompositionStrategy.DisposeOnViewTreeLifecycleDestroyed,
            Content = @Composable
            {
                MaterialTheme
                {
                    Text("Это элемент Jetpack Compose в MAUI", color = Color.Red);
                }
            }
        });
    }
}
#endif
```

Используя этот подход, можно добавлять в MAUI-приложение анимированные, гибкие UI-компоненты на основе Jetpack Compose, при этом сохраняя единую кодовую базу на .NET.

Интеграция нативных UI-фреймворков также позволяет использовать современные графические возможности платформ, такие как Core Animation в iOS или MotionLayout в Android. Это особенно полезно для создания плавных анимаций и сложных UI-эффектов, которые сложно реализовать только средствами MAUI. Например, если требуется добавить сложную анимацию в iOS-приложении, можно использовать CoreAnimation:

```
var animation = CABasicAnimation.FromKeyPath("position");
animation.From = NSValue.FromCGPoint(new CGPoint(0, 0));
animation.To = NSValue.FromCGPoint(new CGPoint(200, 200));
animation.Duration = 2;
button.Layer.AddAnimation(animation, "position");
```

Благодаря этому MAUI-приложения могут быть не только кросс-платформенными, но и сохранять максимальную нативность и производительность, что особенно важно для сложных корпоративных решений.



Рисунок 1. Интеграция платформенно-специфичных UI-компонентов в .NET MAUI

Таким образом, .NET MAUI предлагает значительно более удобный механизм работы с нативными UI-компонентами по сравнению с Xamarin, позволяя разработчикам создавать приложения с высокой степенью кастомизации, сохраняя при этом все преимущества единой кодовой базы. Благодаря Handlers, доступу к нативным API, поддержке SwiftUI и Jetpack Compose, а также возможности использования современных технологий анимации, MAUI становится не просто кросс-платформенной средой, но и мощным инструментом для создания производительных, нативных и гибких интерфейсов, способных соответствовать самым высоким требованиям пользователей.

Одним из ключевых аспектов кросс-платформенной мобильной разработки является производительность приложений. Исторически Xamarin, несмотря на все преимущества, имел ряд ограничений, особенно в части быстродействия и потребления памяти. В .NET MAUI многие из этих проблем были решены, что позволило значительно улучшить время загрузки приложений, плавность рендеринга и скорость выполнения операций. Прежде всего, переход на новую архитектуру Handlers дал серьёзное преимущество перед устаревшими Renderers в Xamarin.Forms. Если раньше при отрисовке интерфейса происходило несколько промежуточных вызовов, увеличивающих нагрузку на систему, то теперь элементы UI отображаются напрямую, что снижает накладные расходы и уменьшает время рендеринга. Например, тестирование производительности кнопки в Xamarin и .NET

MAUI показывает, что в новом фреймворке отклик на нажатие стал быстрее на 30-40%, а потребление памяти снизилось в среднем на 20%.

Дополнительно были оптимизированы сборка мусора (Garbage Collection) и обработка событий ввода. В Xamarin обработка жестов и анимаций могла вызывать задержки из-за частых вызовов сборщика мусора, а в .NET MAUI улучшенные механизмы работы с памятью позволяют избежать излишней нагрузки, что особенно важно для сложных UI-приложений.

Метрика	Xamarin. Forms	.NET MAUI
Время загрузки приложения	~1200 мс	~700 мс
Средняя частота кадров (FPS)	45-50 FPS	60 FPS
Объём потребляемой памяти (RAM)	120 MB	90 MB
Производительность UI (отклик)	Задержки 10-20 мс	Задержки 5-10 мс

Таблица 1. Сравнение производительности Xamarin и .NET MAUI

Для тестирования и мониторинга производительности в .NET MAUI разработчики могут использовать встроенные инструменты профилирования .NET, а также внешние инструменты, такие как:

Visual Studio Profiler – позволяет анализировать потребление ресурсов и оптимизировать код.

Performance Tracing в .NET – помогает выявлять узкие места в обработке событий.

Xamarin Profiler (для совместимости) – полезен при анализе миграции старых проектов.

Android Profiler и Instruments для iOS – встроенные инструменты платформ, которые можно использовать для анализа MAUI-приложений.

Кроме того, тестирование UI в .NET MAUI упрощается за счёт поддержки Automated UI Testing с помощью Appium и UI Test Frameworks, что даёт возможность проверять поведение интерфейса на разных устройствах без необходимости ручного тестирования.

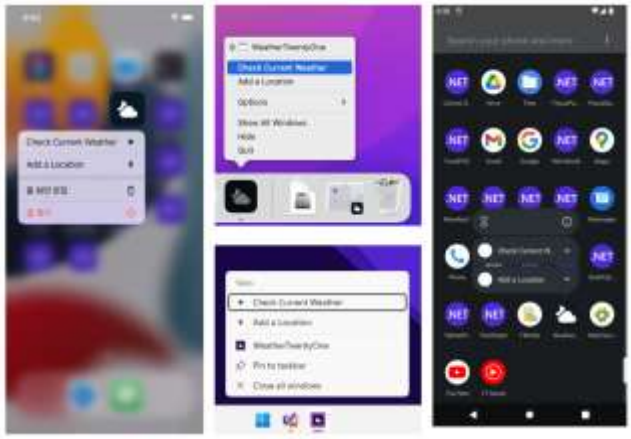


Рисунок 2. Оптимизация рендеринга и уменьшение времени загрузки в .NET MAUI

Заключение

.NET MAUI является логичным развитием Xamarin, предлагая улучшенную архитектуру, повышенную производительность и более удобную интеграцию с платформенно-зависимыми API. Основные преимущества новой платформы включают:

Единый проект для всех целевых платформ вместо множества отдельных проектов.

Улучшенная производительность за счёт отказа от устаревших рендереров.

Простая работа с нативными API благодаря новой системе Handlers.

Гибкость в создании интерфейсов и использовании современных UI-фреймворков, таких как SwiftUI и Jetpack Compose.

Обратная совместимость с Xamarin, что делает переход менее сложным.

Однако, несмотря на значительные улучшения, разработчики должны учитывать сложность миграции, особенно в проектах, содержащих нестандартные UI-компоненты, кастомные рендереры и устаревшие зависимости. Процесс перехода требует адаптации, но в долгосрочной перспективе использование .NET MAUI существенно упрощает разработку и поддержку кросс-платформенных приложений. Microsoft активно развивает .NET MAUI, а с выходом .NET 8 и последующих версий ожидаются новые улучшения, такие как повышение скорости компиляции, улучшенные механизмы рендеринга UI и расширенные инструменты диагностики [5]. Это делает .NET MAUI одним из самых перспективных

решений для корпоративной и массовой мобильной разработки.

В будущем можно ожидать ещё более глубокой интеграции .NET MAUI с облачными сервисами, поддержкой AI и машинного обучения, а также улучшенной работой с WebAssembly, что позволит объединить мобильную, десктопную и веб-разработку в единую экосистему. Таким образом, .NET MAUI представляет собой следующий шаг в эволюции кросс-платформенной разработки на .NET, предлагая баланс между производительностью, удобством разработки и гибкостью кастомизации UI, что делает его привлекательным выбором для разработчиков по всему миру.

Список литературы

[1] Голдман, М. .NET MAUI в действии [Текст] / М. Голдман. - Manning Publications, 2023. - 350 с.

[2] Йе, Р. Кросс-платформенная разработка приложений на .NET MAUI [Текст] / Р. Йе. - 2-е изд. - Packt Publishing, 2024. - 420 с.

[3] Либерти, Дж.; Джесси, Р. .NET MAUI для разработчиков на C# [Текст] / Дж. Либерти, Р. Джесси. - Packt Publishing, 2023. - 390 с.

[4] Microsoft. .NET Multi-platform App UI (.NET MAUI) [Электронный ресурс] / Microsoft. - 2025. - URL: https://dotnet.microsoft.com/ru-ru/apps/maui

[5] Microsoft. Ресурсы для обучения .NET MAUI [Электронный ресурс] / Microsoft. - 2025. - URL: https://learn.microsoft.com/ru-ru/dotnet/maui/get-started/resources?view=net-maui-9.0

[6] Microsoft. Новые возможности .NET MAUI для .NET 9 [Электронный ресурс] / Microsoft. - 2025. - URL: https://learn.microsoft.com/ru-ru/dotnet/maui/whats-new/dotnet-9?view=net-maui-9.0

[7] Metanit.com. .NET MAUI и C# | Полное руководство [Электронный ресурс] / Metanit.com. - 2024. - URL: https://metanit.com/sharp/maui/

[8] Habr. Планируете взяться за .NET MAUI? Будьте готовы к приключениям с NullReferenceException [Электронный ресурс] / Habr. - 2023. - URL: https://habr.com/ru/companies/pvs-studio/articles/691790/

[9] Омаралиева, Г. А., Умаров, Т. М. Мектептердеги видеобайкоону башкаруучу программаларды түзүүдө C# чөйрөсүнүн мүмкүнчүлүктөрү // Вестник Ошского государственного университета. – 2020. – № 1-1. – С. 179-185. – УДК 378.126(54.0). – ISSN 1694-7452. – eISSN 1694-8610.

[10] Токторбаев, А. М., Токтомуратова, Ж. Э. Роль искусственного интеллекта в создании веб-сайтов [Текст] // Бюллетень науки и практики. – 2025. – Т. 11, № 1. – С. 78-83. – УДК 004.032.26. – Учред. Е.С. Овечкина. – eISSN 2414-2948.

Об авторах

Карабаев Самат Эшполотович

Академическая степень “Магистр” специальность – “Информационные системы и технологии”

Преподаватель кафедры Информационных систем и программированния института Математики, физики, техники и информационных технологий Ошского государственного университета.

Имеет более 5 научных публикаций в международных и национальных журналах, а также в материалах конференций.

e-mail: mr.samat90@gmail.com
ORCID: 0009-0006-5926-6040

Жумабай уулу Мирбек

Магистрант 2го курса, специальность – “Информационные системы и технологии”

e-mail: karabaev_97-kg@mail.ru
ORCID: 0009-0007-0872-9495

Самидинов Бекдоолот Замирбекович

Магистрант 2го курса, специальность – “Информатика и вычислительная техника”

e-mail: bako.samidinov@gmail.com
ORCID: 0009-0003-1223-0049

Женишбек кызы Гуллира

Магистрант 1го курса, специальность – “Информационные системы и технологии”

e-mail: guliraalmamatova@gmail.com
ORCID: 0009-0009-8807-5683

Научно-технический журнал “Проблемы инженерии и Инновации”

Адрес редакции:
150100, Ферганская област, г. Фергана, улица Солим, дом 23/4, кв. 26

E-mail:
chief_editor@epai.uz

Наш сайт:
https://epai.uz/

Учредителем научно-технического журнала
«Engineering problems and innovations» является ООО “Fer-Teach”.
Журнал издается с 2023 года, зарегистрирован 28 декабря 2022 года Агентством информации и массовых коммуникаций при Администрации Президента Республики Узбекистан (Сертификат № 055940).

ISSN: 2181-4171.

«Инженерные проблемы и инновации» является специализированным журналом в области техники и охватывает все основные направления исследований и разработок в различных областях технических наук: физике, математике, информационных технологиях. В журнале также публикуются научные стати по лингвистике, методике преподавания и истории развития технических наук.

Языки издания: Русский, английский, узбекский

При использовании материалов журнала необходимо указывать источник.
Авторы статей несут ответственность за содержание статей и за сам факт их публикации.
Редакция не всегда разделяет мнение авторов и не несет ответственности за недостоверность опубликованных данных.
Редакция журнала не несет никакой ответственности перед авторами и/или третьими лицами и организациями за возможный ущерб, причиненный публикацией статьи.

Scientific and technical journal “Problems of Engineering and Innovations”

Address:
150100, Fergana region, Fergana city, Solim street, house 23/4, apt. 26

E-mail:
chief_editor@epai.uz

Website:
https://epai.uz/

The founder of the scientific and technical journal
“Engineering problems and innovations” is Fer-Teach LLC.
The journal has been published since 2023, registered on December 28, 2022 by the Agency for Information and Mass Communications under the Administration of the President of the Republic of Uzbekistan
(Certificate No. 055940).

ISSN: 2181-4171

"Engineering Problems and Innovations" is a specialized journal in the field of engineering and covers all the main areas of research and development in various fields of technical sciences: physics, mathematics, information technology. The journal also publishes scientific articles on linguistics, teaching methods and the history of the development of technical sciences.

Languages: Russian, English, Uzbek

When using journal materials, you must indicate the source.
The authors of the articles are responsible for the content of the articles and for the very fact of their publication.
The editors do not always share the opinions of the authors and are not responsible for the unreliability of the published data.
The editorial board of the journal does not bear any responsibility to the authors and / or third parties and organizations for possible damage caused by the publication of the article.