**Отчет по Главе 2: Основы XAML**

**1. Введение**

Данная глава посвящена изучению основ XAML (Extensible Application Markup Language) в контексте .NET MAUI. XAML является ключевой технологией для декларативного определения пользовательского интерфейса в приложениях .NET MAUI. Его значимость заключается в возможности отделить визуальное представление от бизнес-логики, что упрощает разработку, повышает читаемость кода и способствует лучшему взаимодействию между дизайнерами и разработчиками. Изучение XAML необходимо для эффективного создания интерфейсов настольных и мобильных приложений на .NET MAUI в рамках учебной практики.

**2. Основная часть**

**2.1. Основные концепции и принципы**

* **XAML (Extensible Application Markup Language):** Язык разметки на основе XML, используемый для инстанцирования объектов .NET и установки их свойств. В .NET MAUI XAML преимущественно используется для определения страниц и элементов управления.
* **Пространства имен (Namespaces):** XAML-файлы используют пространства имен для доступа к типам. Основные:
  + http://schemas.microsoft.com/dotnet/2021/maui (по умолчанию): для типов .NET MAUI.
  + http://schemas.microsoft.com/winfx/2009/xaml (обычно с префиксом x): для встроенных типов и директив XAML.
  + Пользовательские пространства имен (например, clr-namespace:YourAppNamespace) для доступа к типам из вашего проекта.
* **Связь XAML и C# (Code-behind):**
  + Атрибут x:Class="Namespace.ClassName" в корневом элементе XAML-файла связывает его с классом C# в файле code-behind (например, MyPage.xaml и MyPage.xaml.cs).
  + Класс в code-behind объявляется как partial, так как компилятор генерирует вторую часть этого класса на основе XAML-разметки.
  + Метод InitializeComponent() в конструкторе класса code-behind необходим для загрузки XAML и создания дерева объектов UI.
* **Именование элементов (x:Name):** Директива x:Name присваивает уникальное имя элементу в XAML, позволяя обращаться к нему из кода C#.
* **Обработчики событий:** События элементов управления (например, Clicked у Button) могут быть привязаны к методам-обработчикам в файле code-behind.
* **Создание интерфейса из кода C#:** Несмотря на то, что XAML является предпочтительным способом определения UI, его также можно полностью создавать программно на C#.
* **Метод LoadFromXaml:** Позволяет динамически загружать XAML-разметку из строки во время выполнения.

**2.2. Примеры кода с пояснениями**

**2.2.1. Создание интерфейса из кода C# (CodePage.cs)**  
В этом примере страница и ее содержимое создаются исключительно с использованием C# кода.

* Код CodePage.cs:

// Файл: CodePage.cs

using Microsoft.Maui.Controls;

namespace HelloMauiApp;

public class CodePage : ContentPage

{

public CodePage()

{

this.Title = "Code Page"; // Установка заголовка страницы

Label welcomeLabel = new Label

{

Text = "Hello from C# code!",

FontSize = 24,

HorizontalOptions = LayoutOptions.Center,

VerticalOptions = LayoutOptions.CenterAndExpand // Для лучшего центрирования

};

Button codeButton = new Button

{

Text = "Click me (C#)",

HorizontalOptions = LayoutOptions.Center

};

int codeCount = 0;

codeButton.Clicked += (sender, args) =>

{

codeCount++;

codeButton.Text = $"Clicked {codeCount} times (C#)";

};

VerticalStackLayout stackLayout = new VerticalStackLayout

{

Spacing = 15,

Padding = new Thickness(0, 20),

VerticalOptions = LayoutOptions.Center,

Children =

{

welcomeLabel,

codeButton

}

};

this.Content = stackLayout; // Установка содержимого страницы

}

}

Пояснение: В конструкторе класса CodePage создаются экземпляры Label и Button, настраиваются их свойства (текст, размер шрифта, выравнивание), и для кнопки назначается обработчик события Clicked с использованием лямбда-выражения. Затем эти элементы добавляются в VerticalStackLayout, который становится содержимым страницы.

**2.2.2. Динамическая загрузка XAML (DynamicXamlPage.cs)**  
Этот пример демонстрирует загрузку XAML-разметки, определенной в виде строки, во время выполнения.

* Код DynamicXamlPage.cs:

// Файл: DynamicXamlPage.cs

using Microsoft.Maui.Controls;

namespace HelloMauiApp;

public class DynamicXamlPage : ContentPage

{

public DynamicXamlPage()

{

this.Title = "Dynamic XAML Page"; // Установка заголовка

string pageXaml =

@"<?xml version=""1.0"" encoding=""utf-8""?>

<ContentPage xmlns=""http://schemas.microsoft.com/dotnet/2021/maui""

xmlns:x=""http://schemas.microsoft.com/winfx/2009/xaml"">

<VerticalStackLayout Spacing=""10"" Padding=""0,20"" VerticalOptions=""Center"">

<Label x:Name=""dynamicLabel"" Text=""Initial Text from Dynamic XAML"" FontSize=""18"" HorizontalOptions=""Center""/>

<Button x:Name=""dynamicButton"" Text=""Change Label (Dynamic)"" HorizontalOptions=""Center""/>

</VerticalStackLayout>

</ContentPage>";

this.LoadFromXaml(pageXaml); // Загрузка XAML из строки

// Получение доступа к элементам, загруженным из XAML

Label loadedLabel = this.FindByName<Label>("dynamicLabel");

Button loadedButton = this.FindByName<Button>("dynamicButton");

int dynamicCount = 0;

if (loadedButton != null && loadedLabel != null)

{

loadedButton.Clicked += (sender, args) =>

{

dynamicCount++;

loadedLabel.Text = $"Label changed {dynamicCount} times!";

};

}

}

}

Пояснение: XAML-разметка для страницы определяется в строковой переменной pageXaml. Метод this.LoadFromXaml(pageXaml) разбирает эту строку и создает соответствующие UI-объекты. После загрузки можно получить доступ к элементам по их x:Name с помощью метода this.FindByName<T>() и назначить им обработчики событий.

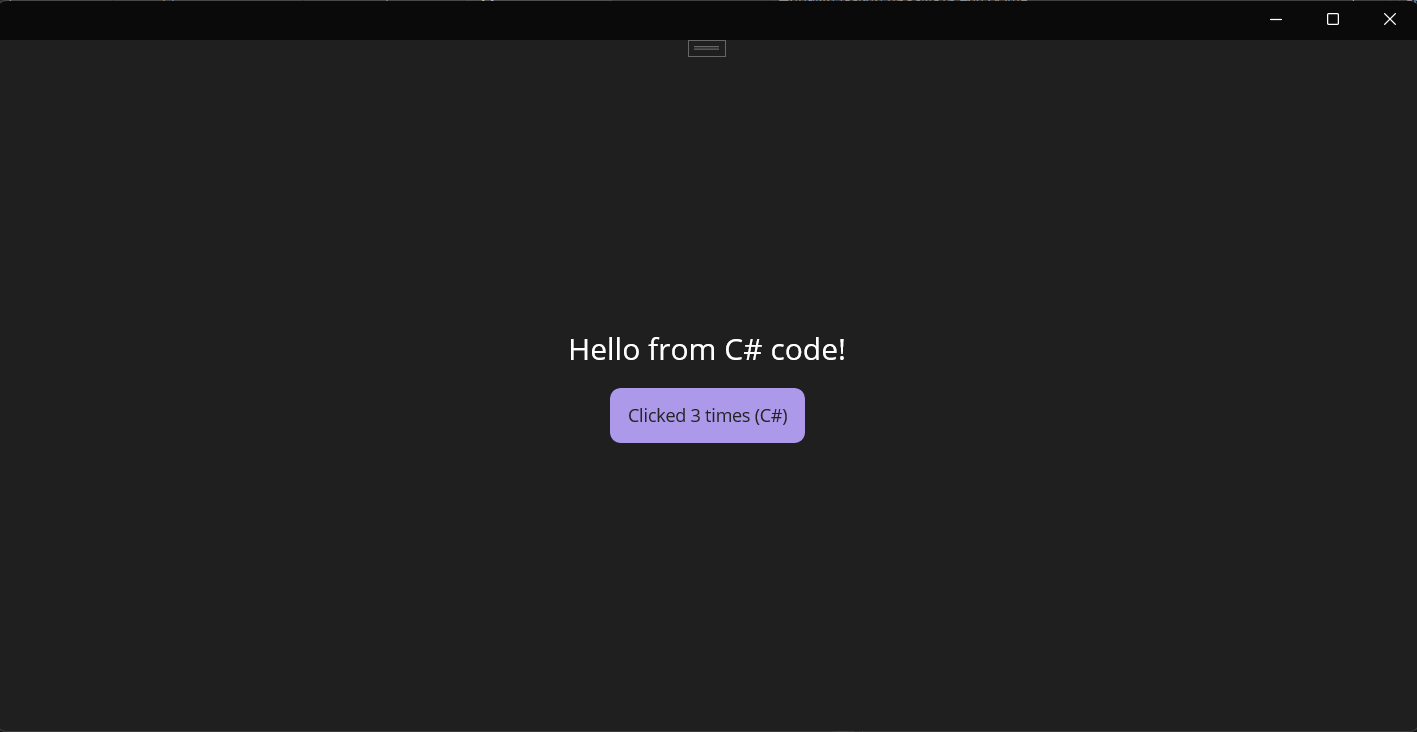
**3. Заключение**

Освоение XAML является критически важным для разработки приложений на .NET MAUI, так как он предоставляет декларативный, интуитивно понятный и мощный способ определения пользовательских интерфейсов. Понимание связи XAML с кодом C#, возможность именования элементов и обработки их событий, а также альтернативные подходы, такие как создание UI программно и динамическая загрузка XAML, обеспечивают разработчику необходимую гибкость. Эти знания напрямую применимы при создании интерфейсов настольных приложений, позволяя структурировать проект и отделять логику от представления.

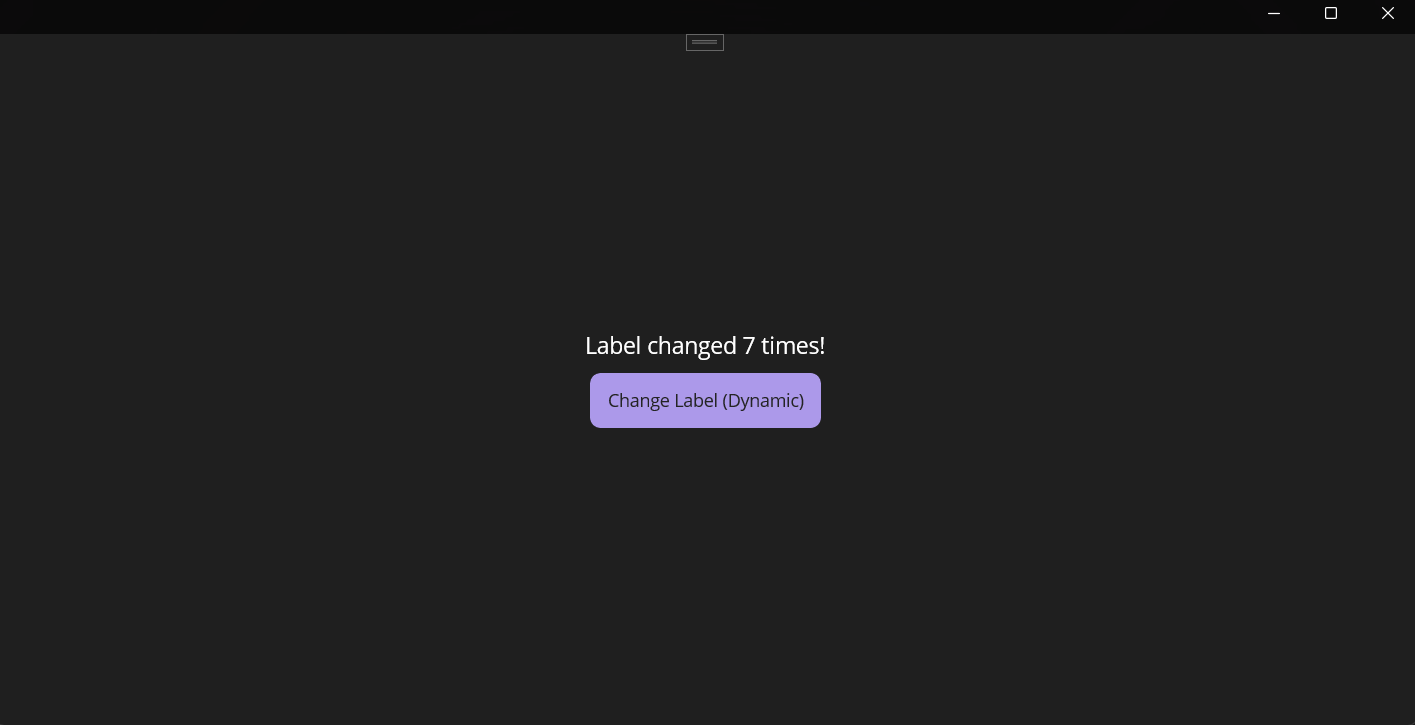
**4. Скриншоты и примеры**

Для демонстрации примеров из данной главы были созданы классы CodePage и DynamicXamlPage. Для их отображения необходимо было поочередно изменять стартовую страницу в файле App.xaml.cs, присваивая MainPage новый экземпляр соответствующей страницы:  
MainPage = new HelloMauiApp.CodePage();  
или  
MainPage = new HelloMauiApp.DynamicXamlPage();

* **Демонстрация CodePage:**

  
Рис. 2.1. Работа страницы CodePage, интерфейс которой создан программно на C#.

* **Демонстрация DynamicXamlPage:**

  
Рис. 2.2. Работа страницы DynamicXamlPage, интерфейс которой загружен динамически из XAML-строки.

**Процесс выполнения примеров:**

1. В проекте HelloMauiApp созданы новые классы C#: CodePage.cs и DynamicXamlPage.cs с кодом, приведенным выше.
2. Для отображения CodePage: в файле App.xaml.cs строка MainPage = new AppShell(); была заменена на MainPage = new HelloMauiApp.CodePage();. Приложение было скомпилировано и запущено.
3. Для отображения DynamicXamlPage: в файле App.xaml.cs строка MainPage = new HelloMauiApp.CodePage(); (или new AppShell();) была заменена на MainPage = new HelloMauiApp.DynamicXamlPage();. Приложение было скомпилировано и запущено.  
   Оба примера успешно скомпилировались и отобразили ожидаемый интерфейс. Кнопки на обеих страницах корректно обрабатывали нажатия. Проблем в процессе выполнения не возникло.

**5. Документация**

* **Ссылка на GitHub с исходным кодом:** https://github.com/RaniZee/.NET-MAUI-Showcase-Cross-Platform-Applications
* **Описание выполненных шагов:**
  1. Изучены материалы по теме "Основы XAML" (глава 2) с ресурса [metanit.com/sharp/maui/](https://www.google.com/url?sa=E&q=https%3A%2F%2Fmetanit.com%2Fsharp%2Fmaui%2F).
  2. Проанализирована структура XAML-файла, использование пространств имен, директив x:Class и x:Name.
  3. Изучен механизм взаимодействия XAML и C# code-behind, включая роль InitializeComponent() и обработчиков событий.
  4. Реализован пример создания пользовательского интерфейса полностью на C# в классе CodePage.
  5. Реализован пример динамической загрузки XAML из строки с использованием метода LoadFromXaml в классе DynamicXamlPage.
  6. Оба примера успешно протестированы путем поочередной установки их в качестве MainPage приложения и запуска на платформе Windows.