УДК 004.43:004.414

ОСОБЛИВОСТІ ПРЕДСТАВЛЕННЯ СИМВОЛІВ UNICODE У МОВІ ПРОГРАМУВАННЯ C#

Андренко А.С., Хацько К.О., Шебанов Є.О.
e-mail: anna.andrenko@nure.ua nataliia.khatsko@nure.ua
Науковий керівник – к.т.н., доц. Хацько Н.Є.
Харківський національний університет радіоелектроніки, кафедра ПІ
м. Харків, Україна

This research focuses on the specifics of Unicode character representation in the C# language in the .NET environment. Modern software systems actively work with text data that may contain characters from different languages, special characters, and emojis. In C#, characters are stored using the char type, which is based on UTF-16 encoding. The principles of representing Unicode characters in UTF-16 encoding are analysed, in particular the difference between a Unicode character and a UTF-16 code unit. Special attention is paid to characters that are outside the basic multilingual plane and are encoded using a surrogate pair. The practical implications of these features for software development are discussed, in particular possible errors during text string processing.

Сучасне програмне забезпечення активно працює з текстовими даними, що можуть містити символи різних мов, спеціальні знаки та емодзі. Для забезпечення коректної обробки таких даних використовуються стандарти кодування символів, зокрема Unicode. Мова програмування C# широко застосовується для створення серверних, десктопних і веб-застосунків, які обробляють текстову інформацію. У середовищі .NET та мові програмування C# символи зберігаються за допомогою типу char, який базується на кодуванні UTF-16 [1]. Це забезпечує підтримку більшості текстових символів Unicode та дозволяє створювати програмне забезпечення, орієнтоване на міжнародне використання.

Незважаючи на поширене використання UTF-16 у C#, існують особливості представлення деяких символів Unicode [2]. Більшість символів можуть бути представлені одним значенням типу char, що займає 16 біт пам’яті. Однак символи з кодовими позиціями, що перевищують цей діапазон, кодуються за допомогою двох 16-бітних значень, так званої сурогатної пари. Наприклад, багато сучасних емодзі є такими символами. Під час написання коду розробники можуть помилково вважати, що кожен символ відповідає одному значенню char, що призводить до помилок під час обробки тексту або некоректної роботи з міжнародними символами.

Метою дослідження є аналіз особливостей представлення символів Unicode у мові програмування C#, зокрема використання кодування UTF-16 та механізму сурогатної пари, а також визначення впливу кодування символів на коректну обробку текстових даних у програмному забезпеченні.

Питання коректного представлення та обробки символів Unicode, зокрема механізму сурогатних пар у кодуванні UTF-16, є актуальним у сучасних наукових дослідженнях. У роботі M. Schröder [3] досліджуються методи перевірки коректності тексту, що містить сурогатні пари, з використанням SIMD-інструкцій. У роботі R. Clausecker та D. Lemire [4] розглядається проблема некоректно сформованих UTF-16 рядків та пропонуються алгоритми їх виявлення і виправлення. Це підтверджує важливість дослідження особливостей кодування Unicode під час розробки програмного забезпечення.

Під час роботи з текстовими даними у мовах програмування важливо розуміти, яким чином символи зберігаються у пам’яті комп’ютера. У мові програмування C# для представлення одного символу використовується тип char, що займає 16 біт пам’яті. Символьні літерали у C# можуть визначатися кількома способами. Окрім безпосереднього запису символу в одинарних лапках, у C# символ може бути представлений через escape-послідовності, які використовуються для позначення спеціальних символів. Наприклад, '\n' означає переведення рядка, а '\\' використовується для запису символу зворотної похилої риски. Також символ може бути заданий через Unicode-код у форматі \uXXXX, де XXXX є шістнадцятковим значенням символу. Наприклад, запис '\u65E5' відповідає символу японського ієрогліфа 日.

У середовищі .NET тип char базується на стандарті Unicode та використовує кодування UTF-16 [2]. Unicode є універсальною системою кодування, створеною для забезпечення єдиного способу представлення символів різних мов світу, включаючи латиницю, кирилицю, ієрогліфи, математичні символи та інші знаки.

Важливо розрізняти поняття символу Unicode та кодової одиниці UTF-16. У мові програмування C# тип char представляє одну 16-бітну кодову одиницю UTF-16, яка не завжди відповідає одному символу Unicode. У більшості випадків одна кодова одиниця відповідає одному символу. Проте символи, що знаходяться за межами області базового набору символів, не можуть бути представлені одним значенням char. Для їх представлення використовується механізм сурогатної пари, у якому один символ кодується двома 16-бітними значеннями (high surrogate та low surrogate), що разом утворюють одну кодову точку Unicode. Наприклад, символ емодзі «🫔» має кодову позицію U+1F370. У кодуванні UTF-16 він представлений двома кодовими одиницями: \uD83C та \uDF70. Іншим прикладом є символ скрипкового ключа «𝄞», який має кодову позицію U+1D11E і представлений двома кодовими одиницями \uD834 та \uDD1E.

Такі символи займають два елементи типу char, тому зберігаються у типі string. Якщо припускати, що один символ відповідає одному значенню char, це може призвести до помилок при підрахунку довжини рядка, обробки тексту або ітерації по символах. Наприклад, рядок "Ні🫔!" містить чотири

символи, однак властивість Length повертає значення 5, оскільки емодзі кодується двома char. Такі помилки суттєві в інформаційних системах, орієнтованих на міжнародне використання, де текстові дані можуть містити символи різних мов, а їх неправильна обробка знижує коректність роботи програм і негативно впливає на взаємодію користувача із системою.

У C# властивість Length типу string повертає кількість 16-бітних кодових одиниць, а не фактичну кількість символів Unicode. Тому рядок, що містить символи за межами базових символів, може мати більшу довжину, ніж очікується. Це слід враховувати при перевірці текстових обмежень, фільтрації даних, порівняння рядків та реалізації алгоритмів обробки тексту. Для коректної роботи з такими символами у .NET передбачені спеціальні засоби, зокрема методи Char.IsSurrogate, Char.IsHighSurrogate та Char.IsLowSurrogate, а також структура System.Text.Rune, що дозволяє працювати з кодовими точками Unicode.

У результаті проаналізовано особливості представлення символів Unicode у мові програмування C# та використання кодування UTF-16 у середовищі .NET. Встановлено, що тип char у C# представляє 16-бітну кодову одиницю UTF-16, яка не завжди відповідає одному символу Unicode. Більшість символів представлені одним значенням char, а для символів з кодовими позиціями поза цією областю використовується сурогатна пара, де один символ Unicode кодується двома 16-бітними одиницями: high surrogate та low surrogate. Ці особливості кодування впливають на обробку текстових даних. Зокрема, один символ Unicode може займати два елементи типу char, що важливо врахувати при роботі з рядками. Наприклад, властивість Length у типі string повертає кількість 16-бітних кодових одиниць UTF-16, а не фактичну кількість символів Unicode, що може призвести до помилки під час підрахунку довжини тексту або ітерації по символах.

Отже, розуміння принципів кодування Unicode та механізму сурогатної пари є важливим для коректної обробки багатомовних текстових даних у сучасних програмних системах. Урахування цих особливостей дозволяє уникати помилок під час роботи з міжнародними символами, зокрема емодзі, та забезпечує правильне функціонування програмного забезпечення.

Список використаних джерел:

1. Microsoft Docs – Char structure (C#). URL: https://learn.microsoft.com/en-us/dotnet/api/system.char?view=net-10.0 (дата звернення: 12.03.2026).

2. The Unicode Standard. URL: https://www.unicode.org/standard/standard.html (дата звернення: 12.03.2026).

3. Validating CESU-8 Encoded Text Utilising SIMD Instructions. URL: https://dl.acm.org/doi/epdf/10.1145/3651781.3651797 (дата звернення: 12.03.2026).

4. R. Clausecker, D. Lemire. Fixing ill-formed UTF-16 strings with SIMD inst. URL: https://arxiv.org/pdf/2601.06349 (дата звернення: 12.03.2026).