

УДК: 004.432

## ДОСЛІДЖЕННЯ МОВ ПРОГРАМУВАННЯ JAVA ТА C# ДЛЯ СЕРВЕРНИХ ПЛАТФОРМ ТА РОБОЧИХ СТАНЦІЙ

\*Устілкин В.В., \*Люта М.В., \*\*Розломий І.О.

\*Київський національний університет технологій та дизайну (ЧФ),  
Україна, Черкаси

\*\*Черкаський національний університет імені Богдана Хмельницького,  
Україна, Черкаси

*В роботі проведено аналіз особливостей використання відомих мов програмування Java та C#, та використання їх у розробці програмного забезпечення для серверних платформ та робочих станцій. Досліджено їх основні переваги та недоліки, проведено порівняння деяких відмінностей у їх синтаксисі. На основі існуючої динаміки популярності, показана перспективність мов програмування Java та C#. Шляхом простого синтетичного тесту було проведено порівняння їх швидкостей на платформі Java virtual machine версії 1.8 та Microsoft .NET Framework версії 4.5. В результаті запропоновано рекомендації, щодо подальшого використання мов програмування Java та C#.*

*Ключові слова: Java, C#, JVM, .NET, програмування.*

*Устилкин В.В. Люта М.В. Розломий И.А. Исследование языков программирования Java и C# для серверных платформ и рабочих станций / Киевский национальный университет технологий и дизайна, Украина, Черкасы*

*В работе проведен анализ особенностей использования известных языков программирования Java и C#, а также использование их в разработке программного обеспечения для*

*серверных платформ и рабочих станций. Исследованы их основные преимущества и недостатки, проведено сравнение некоторых различий в их синтаксисе. На основе существующей динамики популярности, показана перспективность языков программирования Java и C#. Путем простого синтетического теста было проведено сравнение их скоростей на платформе java virtual machine версии 1.8 и .NET Framework версии 4.5. В результате предложены рекомендации относительно дальнейшего использования языков программирования Java и C#.*

*Ключевые слова: Java, C#, JVM, .NET, программирование.*

*Ustilkin V.V. Luta M.V. Rozlomiyy I.O. Research programming languages Java and C# for server platforms and workstation / Kyiv national university of Technologies and Design, Kyiv, Ukraine*

*The paper analyzes the features of the use of known programming languages Java and C#, as well as their use in the development of software for server and workstation platforms. We studied their main advantages and disadvantages compared to some differences in their syntax. On the basis of the existing dynamics of popularity shows prospects of programming languages Java and C#. By simple synthetic test was conducted comparing their speeds on a platform java virtual machine version 1.8 and the .NET Framework version 4.5. As a result of proposed recommendations for future use of the programming languages Java and C#.*

*Key words: Java, C#, JVM, .NET, programming.*

**Вступ.** В наш час є дві популярні конкуруючі мови програмування, що стрімко розвиваються – мова Java та мова C#.

Не дивлячись на те, що C# з'явився значно пізніше ніж Java, вони мають багато спільного, але як кажуть, нічого ідеального не існує.

Даний випадок не є виключенням і в цій статті, проведено аналіз переваг та недоліків кожної з них.

Мови C# і Java з'явилися в різний час. Мова Java була створена задовго до появи C#. Під назвою Oak Java була розроблена компанією Sun Microsystems в 1990 р, а в 1995 була випущена перша бета-версія Java. Створення C# було анонсовано в 2000 році, а в 2002 році вийшла перша версія платформи .NET, що підтримує C#. Таким чином, якщо Java створювалась спираючись більшою мірою на досвід мов Objective C і C, то для C# такою опорою були C ++ і сама Java. І, незважаючи на свою назву, C# виявилась ближче до Java, ніж до C ++.

З точки зору розробника мови Java і C# дуже схожі. Обидві мови є строго типізованими, об'єктно-орієнтованими. Обидві увібрали в себе багато чого з синтаксису C ++, але на відміну від C ++, простіше в освоєнні для початківців. Обидва запозичили з C набір основних ключових слів і службових символів, в тому числі фігурні дужки для виділення блоків. Обидві мови спираються на збірку сміття. Обидві мови супроводжуються багатими колекціями бібліотек. Але є в мовах також свої особливості і відмінності, сильні і слабкі сторони. C# врахувала багато недоліків Java, і виправила їх у своїй реалізації. Але і Java не стоїть на місці, розвиваючись паралельно з C#.

Кік Редек з Microsoft вважає C # більш складною мовою, ніж Java. На його думку, «мова Java була побудована таким чином, щоб уберегти розробника від стрільби собі в ногу» (англ. «Java was built to keep a developer from shooting himself in the foot»), а «C # була побудована так, щоб дати розробникові пістолет, але залишити його на запобіжнику»(англ.« C# was built to give the developer a gun but leave the safety turned on ») [1, с. 36].

**Експериментальна частина.** С# є продуктом Microsoft і основним направленням даної мови є продукти для операційної системи сімейства Windows, як для стаціонарних комп'ютерів, так і для мобільних пристроїв.

Також популярність набирає ігровий двигун Unity 3d, на якому розроблюється багато ігор, в першу чергу за рахунок його кроссплатформеності (можливість запускатися на різних операційних системах) [2, с. 12].

Використання платформи Xamarin дозволяє використовувати С# для розробки програм під різні операційні системи, в тому числі і Android. Хоча система досить нова і на перший погляд дуже перспективна, точно не відомо, наскільки вона приживеться. Це пов'язано з тим, що корпорація Microsoft інколи закриває свої продукти, якщо бачить їх безперспективність. В цьому можна легко переконатись, встановивши їх основний засіб для розробки програмного забезпечення Microsoft visual studio. Якщо порівняти їх версії, то можна помітити, що більш старі версії містять такі засоби для розробки програм, які виключені в більш сучасних версіях Microsoft visual studio, наприклад J#[3, с. 18].

В такому підході Microsoft, є як переваги, так і недоліки. Перевагами є те, що на підтримку актуальності продуктів, необхідно витратити ресурси компанії, які можна використати для розвитку більш перспективних продуктів. Основним недоліком є те, що програміст вимушений буде вчити іншу мову і починати все спочатку, що для нього не є добре, як і для репутації компанії в його очах.

Java, за статистикою 2016 року займає перше місце по популярності. Область використання даної мови дуже велика – вона підтримує можливість запускати програми майже всюди – майже всі операційні системи, персональні комп'ютери, сервера, мобільні

телефони (що використовують java mini), операційна система андроїд, GPS навігатори, відео магнітофони, супутникові системи, вимірювачі тиску, пульсометри, навіть браслети для вимірювання якості сну людини, тощо... За офіційними даними розробника, Java використовується більш ніж на 3 000 000 000 (трьох мільярдах) пристроях[4, с. 8].

Підхід розробників Java в області стабільності абсолютно протилежний компанії Microsoft. Нові компоненти Java тестуються довгий час до того, як повністю вийти на ринок і не видаляються після їх виходу. Безперечно у даного методу є суттєвий недолік, новизна продукту вводиться дуже повільно. Не дивлячись на це, на сьогоднішній день остання версія мови Java майже не поступається в нововведеннях C#[5, с. 6].

**Аналіз популярності.** В зв'язку із тим, що мов програмування дуже багато, проводиться багато аналізів щодо них:

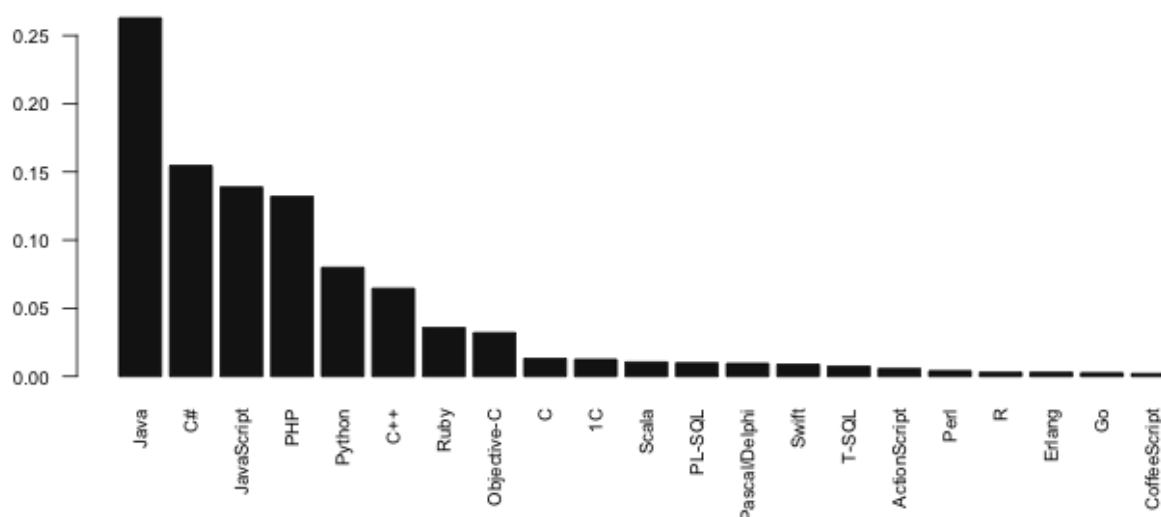


Рис. 1. Графік популярності мов програмування станом на 2016 рік

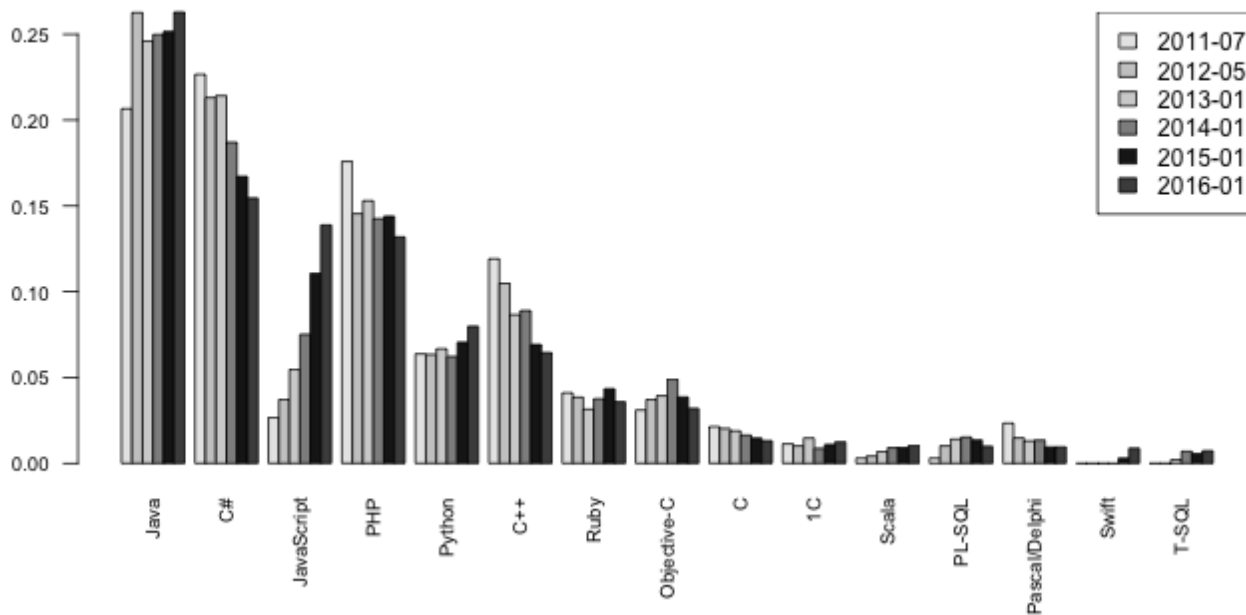


Рис. 2. Динаміка популярності мов програмування за останні роки (зліва 2011 рік, зправа 2016)

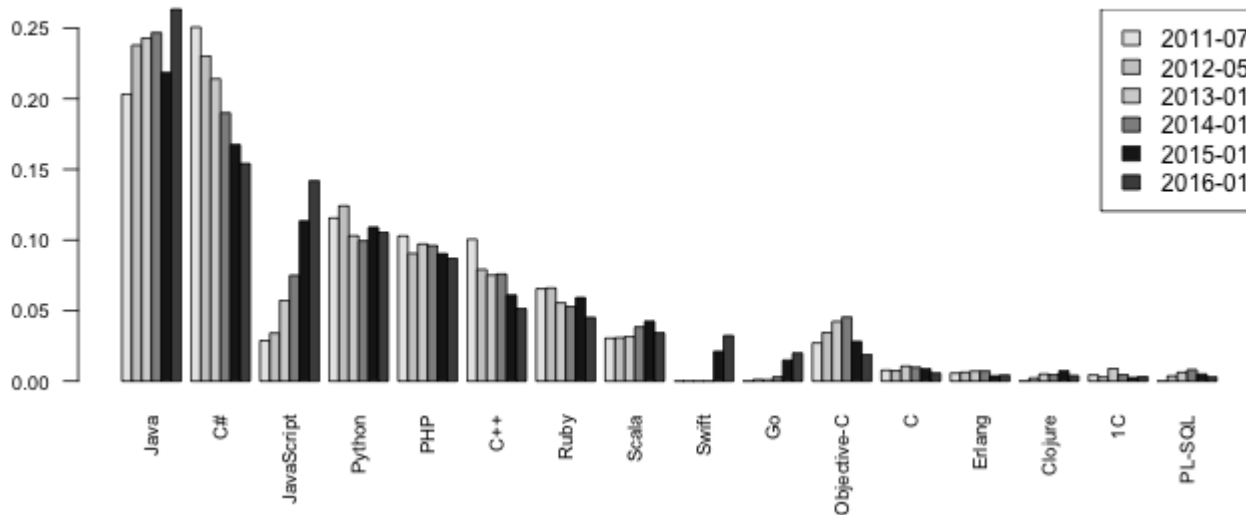


Рис. 3. Мови, яким програмісти віддають перевагу (зліва 2011 рік, справа 2016)

**Порівняння швидкості написаної програми.** Дане порівняння є синтетичним тестом. Його мета порівняти, скільки часу потрібно буде програмам написаним на C# (net 4.5) версія «Release» та Java (1.8) для того, щоб виконати свої задачі в операційній системі Windows 10.

Програми створюють процес створення однакового класу задану кількість разів (для тесту вибрано 1 000 000 000 разів), в якому виконують однакові математичні функції

<pre> 1 reference private void button1_Click(object sender, EventArgs e) {     DateTime startTime, stopTime;     double differentTime = 0;     long cycles = Convert.ToInt32(textBox1.Text);     startTime = System.DateTime.Now;      for (int i = 0; i &lt; cycles; i++)     {         ProcessLoader processLoader = new ProcessLoader();     }     stopTime = System.DateTime.Now;     differentTime = (stopTime - startTime).TotalMilliseconds;     label2.Text = "Start time: " + startTime.ToString();     label3.Text = "Stop time: " + stopTime.ToString();     label4.Text = differentTime.ToString()+" ms"; } </pre>	<pre> 3 references class ProcessLoader {     public static double result = 0;     1 reference     public ProcessLoader()     {         result*=2;         result/=2;          result *= 0.2;         result /= 0.2;          result++;     } } </pre>
--	---

Рис. 4. Код програми на C#

<pre> private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {     long startTime, stopTime, differentTime;     long cycles = Long.valueOf(jTextField1.getText());     startTime=System.currentTimeMillis();     for(int i=0;i&lt;cycles;i++){         ProcessLoader processLoader = new ProcessLoader();     }     stopTime=System.currentTimeMillis();     differentTime = stopTime-startTime;     jLabel2.setText("Start time: "+String.valueOf(startTime));     jLabel3.setText("Stop time: "+String.valueOf(stopTime));     jLabel4.setText("Different time: "+String.valueOf(differentTime)); } </pre>	<pre> public class ProcessLoader {     public static double result=0;     public ProcessLoader(){         result*=2;         result/=2;          result*=0.2;         result/=0.2;          result++;     } } </pre>
--	--

Рис. 5. Код програми на Java

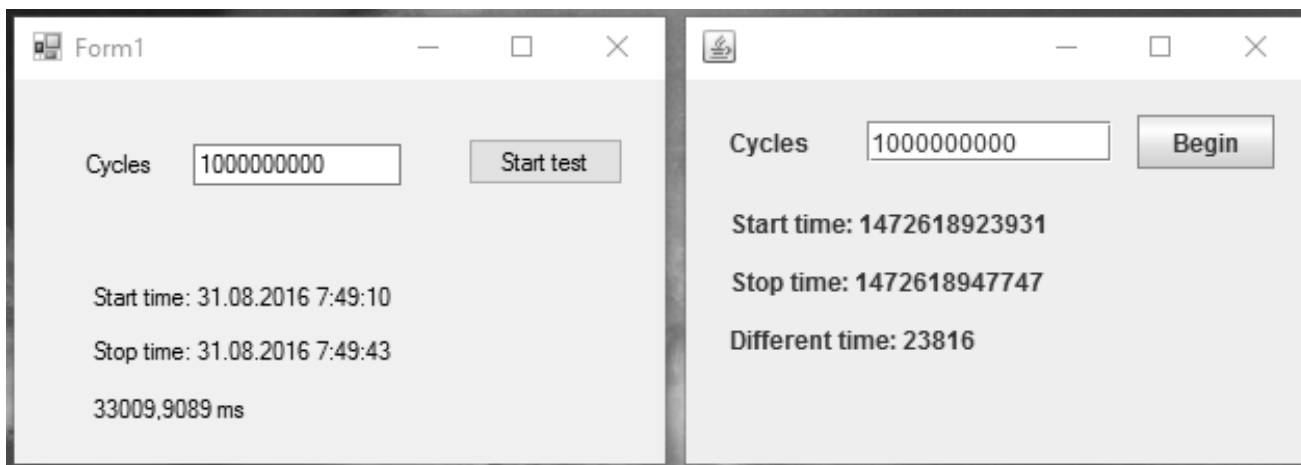


Рис. 6. Тестування швидкості виконання програми

На рис. 6 зображено 2 скриншоти з часом виконання однакової програми на одному комп'ютері, з лівої сторони C#, з правої – Java. В останньому рядку виведено час, що знадобився для виконання в мілісекундах. Програми було запущено не з редакторів, а зі створених файлів, що створювались не як Debug версії, а як Release.

В результаті тесту видно, що для виконання однакових операцій програмі на C# потрібно було більше часу ніж програмі на Java, тобто програма на Java відпрацювала в 1,38 рази швидше.

**Висновки.** На сьогоднішній день, Java популярніша за C#, і якщо врахувати графіки популярності – найближчим часом ситуація не зміниться.

Синтаксис C# більш лаконічний, тому що там замість довгих слів ставляться знаки. Особливою його перевагою є можливість розділяти класи на окремі частини та файли інструкцією «partial».

У плані швидкості розробки програмного забезпечення знову в лідерах C# – на ньому написати не складну програму з використанням GUI (графічний інтерфейс користувача) можна швидше ніж на Java.



У разі, якщо програма має бути кросплатформенною, при виборі серед C# та Java варіантів немає – Java в цьому випадку буде краще.

В швидкості проведеного синтетичного тесту Java перемогла зі значним відривом.

Таким чином, можна зробити висновок, якщо необхідна стабільність, кросплатформенність і швидкість виконання самої програми, то краще писати на Java, а якщо необхідно якомога швидше написати програму, яка працюватиме на базі операційної системи Windows — то вигідніше буде вибрати C#.

### **Література:**

1. Джеффри Рихтер CLR via C#. Программирование на платформе Microsoft.NET Framework 4.5 на языке C#: Питер. – 2016. – 896 с.
2. Эндрю Троелсен, Филипп Дзепикс Язык программирования C# 6.0 и платформа .NET 4.6: Вильямс. – 2016. – 1440 с.
3. Джозеф Албахари, Бен Албахари C# 6.0. Справочник. Полное описание языка: Вильямс. – 2016. – 1040 с.
4. Герберт Шилдт Java 8. Полное руководство: Вильямс. – 2016. – 1376 с.
5. Роберт Лафоре Структуры данных и алгоритмы в Java: Питер. – 2016. – 704 с.

### **References:**

1. Jeffrey Richter, CLR via C#. Programming Microsoft.NET Framework 4.5 platform in the language C#: Peter. - 2016. - 896 p.
2. Andrew Troelsen, Philip Dzhepiks C# 6.0 programming language and platform .NET 4.6: Williams. - 2016 - 1440.
3. Joseph Albahari, Ben Albahari C# 6.0. Directory. Full description of the language: Williams. - 2016 - 1040.
4. Herbert Schildt Java 8. Complete Guide: Williams. - 2016 - 1376 p.

5. *Robert Laforêt Data Structures and Algorithms in Java: Peter.* - 2016. - 704 p.