Лабораторна робота №1

Знайомство з інтегрованим середовищем розробника (IDE) NetBeans

1. Підготовка програмного забезпечення.
2. Усунення можливих проблем з мовою інтерфейсу NetBeans.
3. Створення простого проекту.
4. Створення проекту з графічним інтерфейсом.

1. Підготовка програмного забезпечення

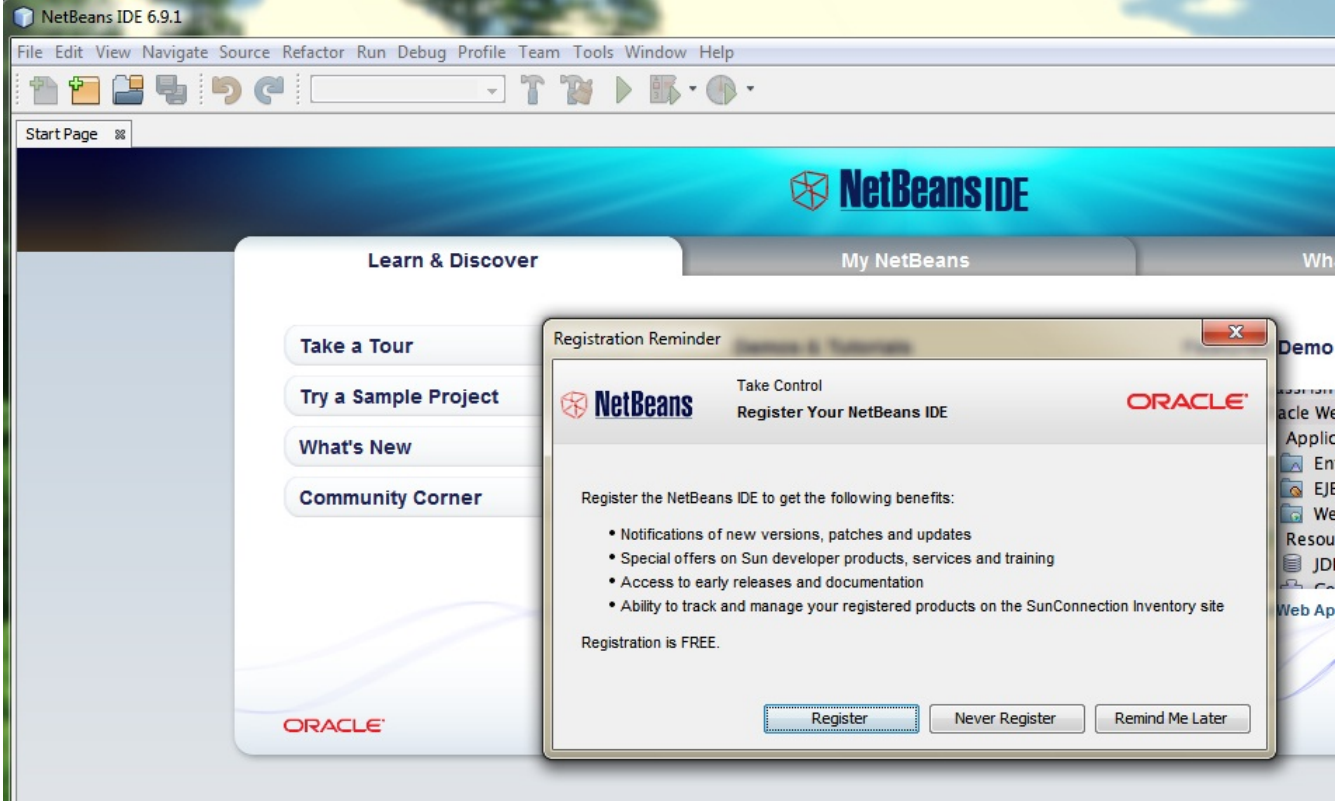
Для створення програмного забезпечення мовою Java необхідно встановити на комп'ютері два основних програмних засоби: інструментарій java-розробника (комплект компіляторів і бібліотек) з віртуальною машиною Java та інтегроване середовище розробника java-програм NetBeans. Перший засіб називається JDK (Java Development Kit — комплект для java-розробки) і, як правило, містить в себе ще й віртуальну машину Java (пакет JRE — Java Runtime Environment). JDK звичайно упаковують в один файл розміром декілька десятків Мбайт. Останню версію JDK можна завантажити безпосередньо з сайту розробника (на момент осені 2010 р. — це http://www.oracle.com/technetwork/java/javase/downloads/index.html). Вибірати слід стандартну редакцію пакета (SE — Standard Edition), якої вистачає як для навчання, так і, навіть, для середніх професійних програмних розробок. Другій засіб (NetBeans) також можна завантажити з цієї ж сторінки, або безпосередньо з сайту NetBeans (http://netbeans.org/downloads/).

Для Windows завантажуються файли JDK і NetBeans і встановлюються строго в такому ж порядку, тому що програма інсталяції NetBeans намагається знайти вже встановлений JDK і підключити його до інтегрованого середовища. При інсталяції рекомендується усі параметри вибірати за умовчанням. Значок NetBeans для запуску буде встановлений як у системне меню, так і на робочий стіл користувача.

Для Linux більшість програмного забезпечення зберігається у так званих репозітаріях. Слід у пакетному менеджері знайти і встановити тільки пакет NetBeans (інші пакети, в тому числі і JDK, будуть встановлені автоматично по пакетним залежностям). Значок NetBeans буде встановлений тільки у системне меню "Приложения => Программирование".

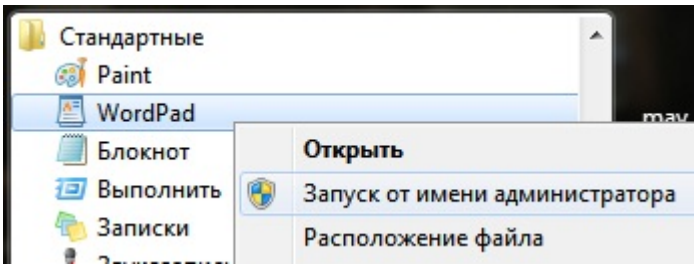
2. Усунення можливих проблем з мовою інтерфейсу NetBeans

NetBeans має багатомовний інтерфейс. На жаль немає українського, але є російський. Якщо мова операційної системи російська (Windows: Панель управления => Язык и региональные стандарты, Linux: Система => Администрирование => Язык системы), то програма інсталяції NetBeans визначає її автоматично і показує увесь процес інсталяції, а потім і інтерфейс NetBeans, саме російською. Якщо мова операційної системи інша, наприклад українська, програма інсталяції у себе її не знаходить і залишається на англійський. Мова інтерфейсу NetBeans після встановлення також буде англійською, як показано на наступному рисунку:

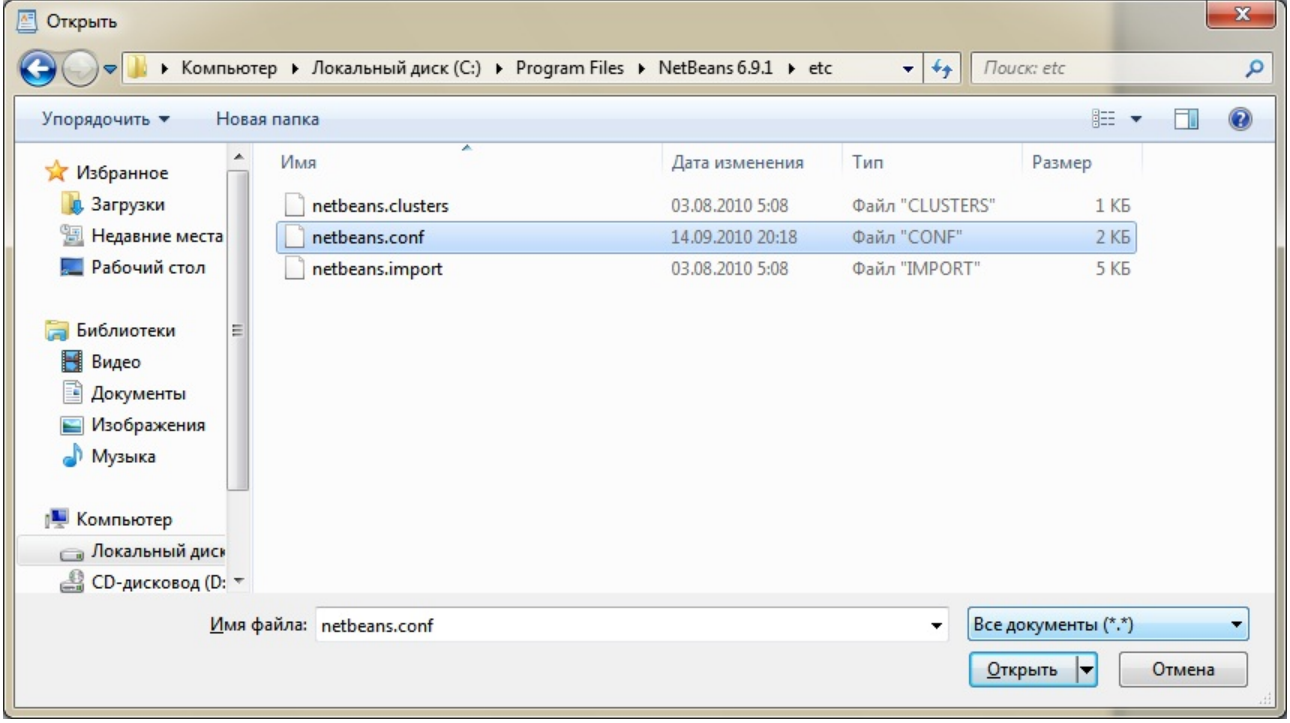


У Windows це виправляється наступним чином (NetBeans повинен бути закритим):

1. Запустіть стандартний текстовий редактор WordPad з правами адміністратора (Главное меню => Все программы => Стандартные => WordPad, клацніть правою кнопкою миші і виберіть “Запуск от имени администратора”):



2. Відкрийте файл за наступним шляхом: C:\Program Files\NetBeans 6.9.1\etc\netbeans.conf (у діалоговому вікні відкриття файлу перемкніть фільтр імен файлів на “Все файлы”, як показано на рисунку, інакше у папці etc ви не побачите жодного файлу):



3. Знайдіть у тексті параметр netbeans_default_options, значенням якого є досить довгий текст (набір параметрів) у лапках:

```
netbeans_default_options="-J-client -J-Xss2m -J-Xms32m -J-
XX:PermSize=32m -J-XX:MaxPermSize=200m -J-
Dapple.laf.useScreenMenuBar=true -J-
Dapple.awt.graphics.UseQuartz=true -J-Dsun.java2d.noddraw=true"
```

4. Додайте додатковий параметр **-J-Duser.language=ru_RU.UTF-8** відразу ж після перших лапок (слід суворо дотримуватися регістра букв, не ставити зайвих пробілів, обов'язково поставити пробіл перед наступним параметром **-J-client**, як показано на рисунку):

```
netbeans_default_options="-J-Duser.language=ru_RU.UTF-8 -J-
client -J-Xss2m -J-Xms32m -J-XX:PermSize=32m -J-XX:MaxPermSize=
200m -J-Dapple.laf.useScreenMenuBar=true -J-
Dapple.awt.graphics.UseQuartz=true -J-Dsun.java2d.noddraw=true"
```

5. Збережіть файл і закрийте вікно редактора WordPad.

Знову запустіть NetBeans. Якщо все зроблено правильно, він буде мати інтерфейс (назви пунктів меню, текст у діалогових вікнах і т.п.) російською мовою.

В Linux слід с правами root відкрити файл /usr/bin/netbeans у простому текстовому редакторі (mc, nano, vi, vim), пропустити усі перші рядки з коментарями (рядки, що починаються з символу #), і відразу після них у новому рядку додати параметр:

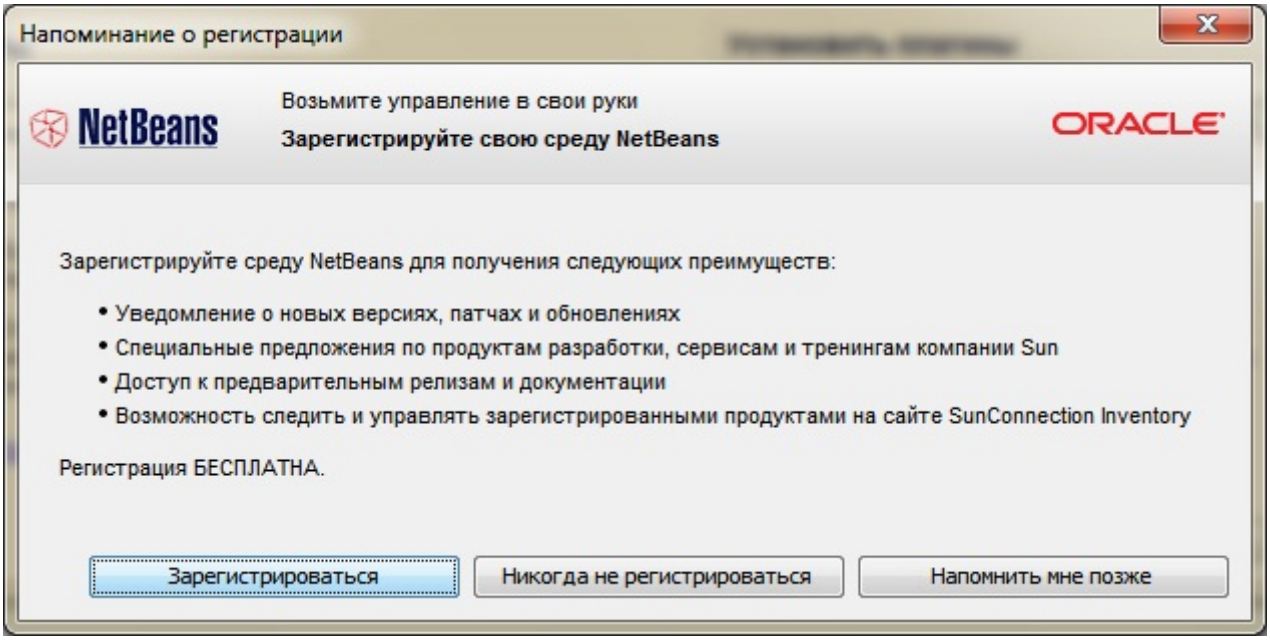
LANG=ru_RU.UTF-8

Збережіть файл і можете запускати NetBeans.

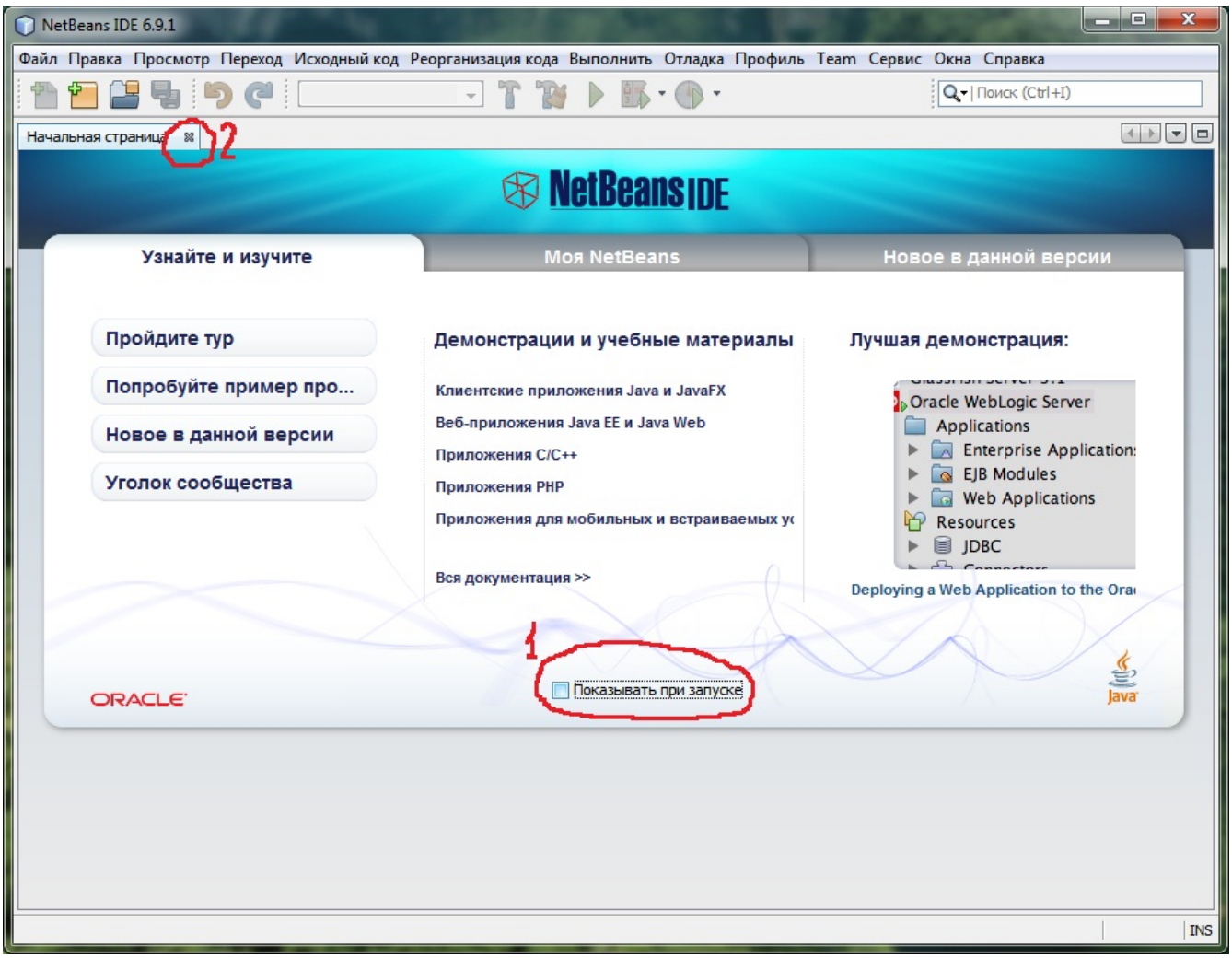
Якщо хтось бажає працювати з англомовним інтерфейсом, в усіх вказаних вище параметрах слід **ru_RU** замінити на **en_US**.

3. Створення простого проекту

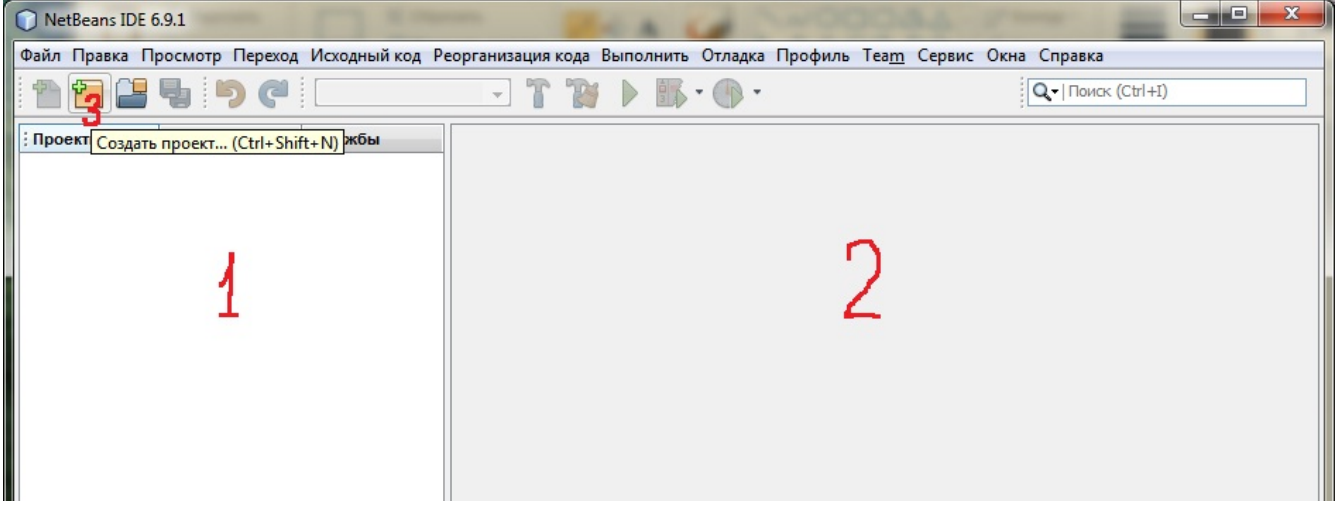
При першому запуску середовища NetBeans зазвичай з’являється діалогове вікно, в якому вам пропонується зареєструвати програмний продукт на сайті розробника. Рекомендуємо відмовитися і натиснути середню кнопку “Никогда не регистрироваться”:



Далі з’явиться початкова сторінка середовища, яка зручна для самостійного ознайомлення з NetBeans. Щоб нам вона не заважала, знімемо галочку “Показывать при запуске” (1) і хрестиком на закладці (2) закриємо цю сторінку:



Остаточно робоче вікно інтегрованого середовища NetBeans прийме вид:



Тут (1) — вікно проекту, (2) — робоча область, (3) — кнопка створення нового проекту. Будь-яка програма у NetBeans створюється у вигляді проекту, спеціальної папці, у якій зберігаються усі елементи вашої роботи.

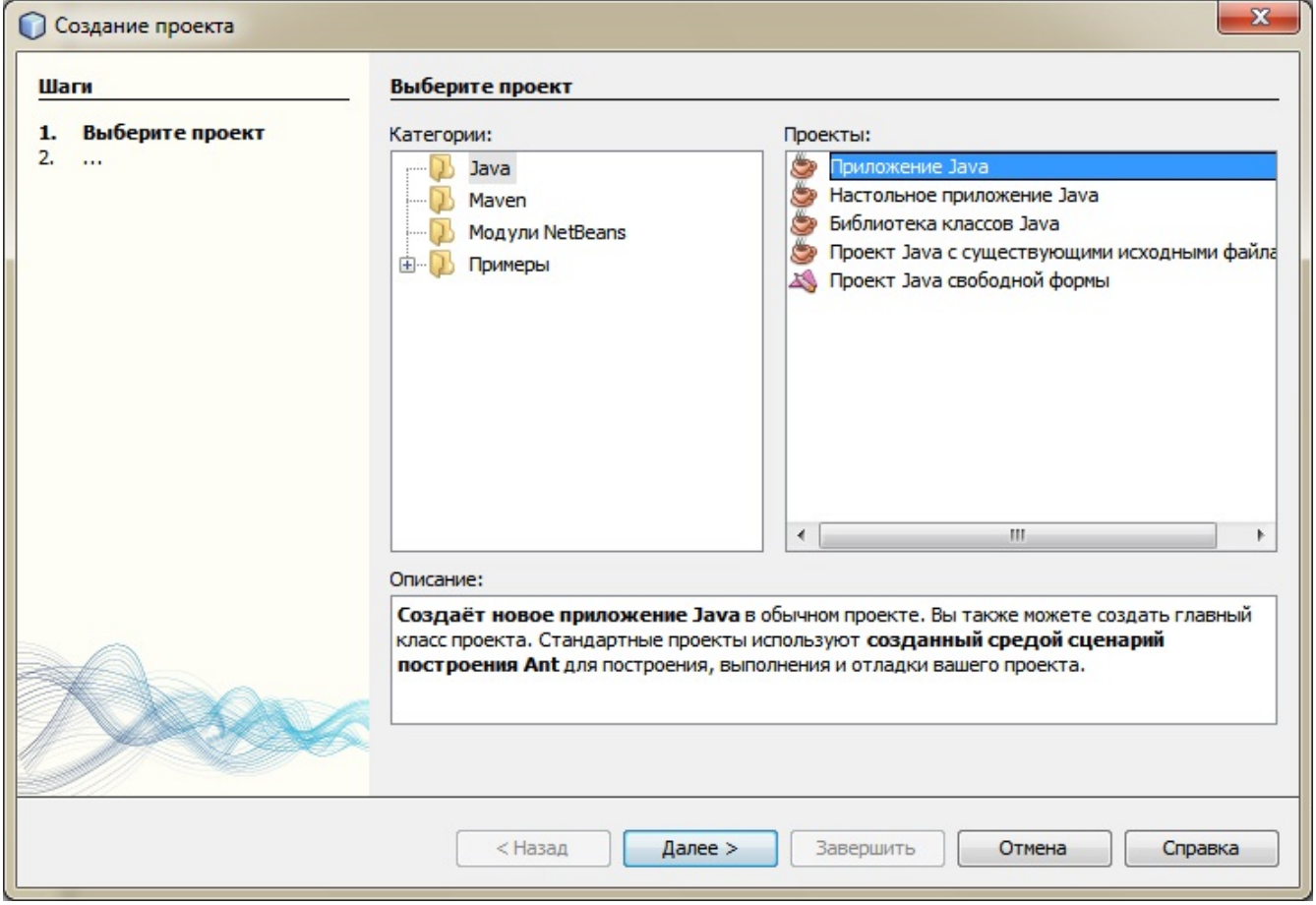
Програма на мові Java намагається моделювати задачу у вигляді опису реальних або абстрактних об'єктів приблизно так, як це робить людина. Тому при програмуванні задач (особливо складних) сам процес програмування повинен виглядати для людини більш-менш природно. Щоб описати будь-який об'єкт, треба вказати його *властивості* (характеристики) і *дії*, які може виконувати цей об'єкт. Властивості реалізуються у вигляді звичайних змінних, імена яких звичайно ж відповідають назвам властивостей. Змінні зберігають поточні значення властивостей. Дії, які може виконувати об'єкт, описуються у вигляді функцій, які прийнято називати *методами*.

Оскільки у програмі можуть функціонувати декілька схожих об'єктів, прийнято описувати не кожен окремий об'єкт, а групу (*клас*) усіх подібних об'єктів. Наприклад, якщо у вашій програмі є декілька кнопок, що виконують певні дії, то описується не кожна окрема кнопка з конкретними розмірами, місцем розташування у вікні та методом, який виконується при її натисканні, а клас усіх подібних кнопок, які можуть мати ширину, висоту, координати, колір, напис та дії, які може виконувати кнопка, реагуючи на ті чи інші події. Опис такого класу є своєрідним шаблоном, за допомогою якого можна швидко створити безліч конкретних об'єктів.

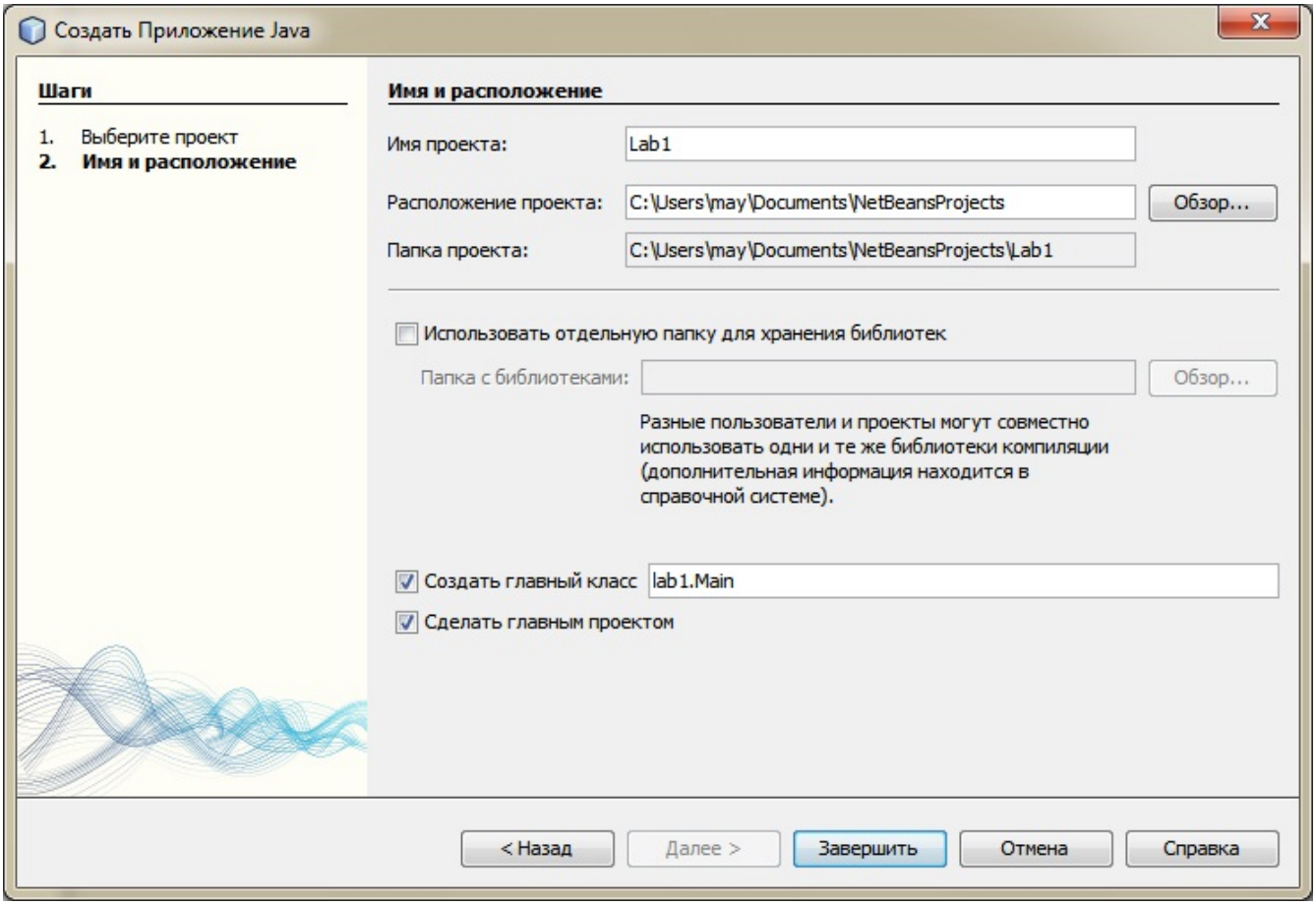
Отже програма на мові Java складається з опису усіх класів, які після запуску програми використовуються для створення усіх потрібних об'єктів. Об'єкти можуть бути статичними, для цього достатньо оголосити у відповідному місці програми змінні об'єктного типу (класу). Часто у вигляді статичних об'єктів створюють візуальні елементи інтерфейсу програми (вікна, кнопки, поля для введення даних та інші). Пам'ять для таких об'єктів відводиться відразу ж після запуску програми, і звільняється тільки після завершення роботи програми. Об'єкти можуть бути динамічними. При цьому об'єкт створюється вже в процесі роботи програми у потрібний момент на основі відповідного класу за допомогою оператора new, який динамічно відводить певний блок пам'яті, у якому розміщуються усі змінні-властивості з конкретними їх значеннями для цього об'єкту. Усі подібні об'єкти користуються методами-функціями з опису їх класу.

Для зручності класи програми групуються у *пакети*. Програма може складатися з одного пакету, а може — з декількох. Усе залежить від бажань самого автора програми. Для можливості запуску програми обов'язково визначається головний клас проекту. Щоб при звертанні до класу він зміг автоматично почати виконувати якісь дії, до нього включається стандартний метод main() (головний), який є так званою "точкою входу" у програму. При створенні проекту NetBeans автоматично створює головний клас Main, який є своєрідною оболонкою для майбутньої програми і містить порожній метод main(). Таким чином, можна швидко створити працюючу програму, просто записавши всередині методу main() необхідний алгоритм дій. У програмі необов'язково повинен бути присутнім клас Main. Можна створити свій набір класів, але один з них обов'язково слід призначити головним. Всередині такого класу обов'язково повинен бути присутнім метод main(). Інакше програма не буде "знати", з чого починати свою роботу і взагалі не запуститься.

Створимо перший проект, натиснувши відповідну кнопку на панелі інструментів:

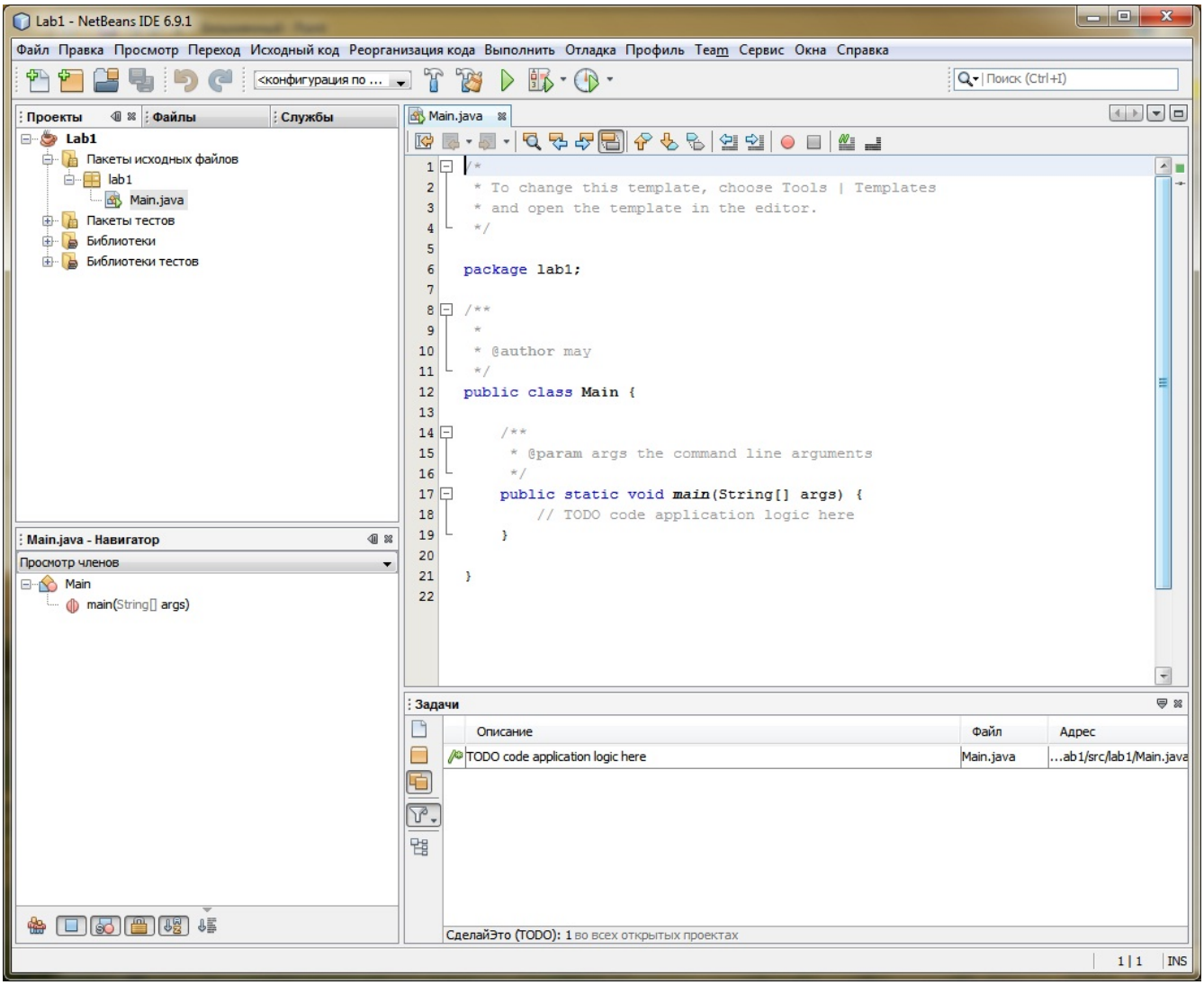


У діалоговому вікні, що з’явилося, виберемо категорію “Java”, у якій виберемо шаблон проекту з назвою “Приложение Java” (поки що завжди будемо робити такий вибір для усіх майбутніх проектів) і натиснемо кнопку “Далее”:



У наступному вікні впишемо ім’я проекту Lab1 (саме з великої літери; пам'ятайте, що регістр букв у Java має значення) і натиснемо кнопку “Завершить” (інші параметри будуть заповнені автоматично і у більшості випадків

задовольнятимуть наші потреби). NetBeans створить проект Lab1, який виглядатиме наступним чином:



У вікні проекту вгорі відображається дерево проекту. З усіх складових ми поки що будемо використовувати тільки “Пакеты исходных файлов”. Інші складові звичайно використовуються у складних проектах. Як видно, система за замовчанням створила один пакет lab1 (нагадаємо, що ім’я Lab1 відрізняється від lab1), у який включено єдиний файл Main.java. У мові Java прийнято кожний окремий клас описувати в окремому файлі. У вікні внизу навігатор показує вміст обраного файлу в структурованому вигляді: це єдиний клас Main, що містить тільки один метод main і жодної властивості.

У робочій області праворуч вгорі показується вихідний код на мові Java обраного файлу Main.java. Як видно, блок коментаря (звичайно займає декілька рядків) починається з символів /* і закінчується символами */. Коментар для одного рядка починається символами // і продовжується до кінця рядка. Коментар використовується для пояснень окремих елементів і частин програми і ігнорується компілятором. Рядок

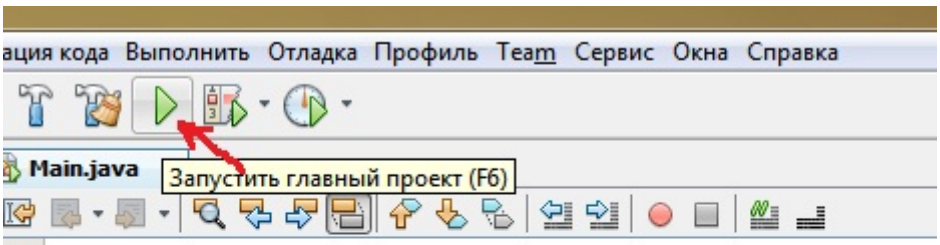
package lab1;

вказує, що усі класи цього файлу входять до пакету lab1. Опис закінчується крапкою з комою. Усі класи описуються у вигляді блоків, які складаються з заголовків і тіла класу. Заголовок public class Main означає, що далі слідує опис публічного (загальнодоступного) класу Main. Тіло блоку завжди обмежується фігурними дужками { і }. Всередину будь якого блоку може бути вкладений інший блок. Так видно, що клас Main має порожній публічний статичний метод main:

```
6  package lab1;
7
8  /**
9   *
10  * @author may
11  */
12 public class Main {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args) {
18         // TODO code application logic here
19     }
20
21 }
22
```

Слово void означає, що функція main не повертає жодного значення, тобто еквівалентна процедурі. Ознакою методу є наявність аргументів у круглих дужках (і) після імені методу. Так, наш метод може приймати масив текстових (String[]) аргументів. Навіть якщо метод не має аргументів, обов'язково слід вказати порожні дужки ().

Примусимо нашу просту програму після запуску і виконання методу main класа Main привітати нас. Для цього у тілі методу main впишемо наступний простий код:

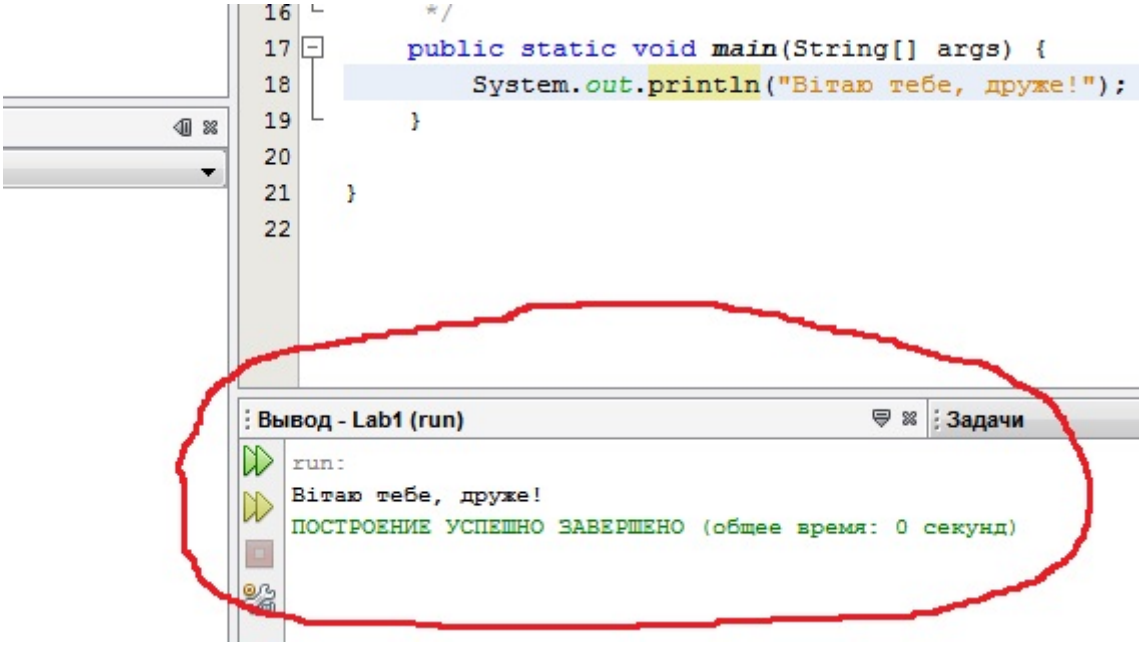



```
1  /*
2   * To change this template, choose Tools | Templates
3   * and open the template in the editor.
4   */
5
6  package lab1;
7
8  /**
9   *
10  * @author may
11  */
12 public class Main {
13
14     /**
15      * @param args the command line arguments
16      */
17     public static void main(String[] args) {
18         System.out.println("Вітаю тебе, друже!");
19     }
20
21 }
22
```

Команда привітання складається зі звертання до системного класу *System*, в якому вибирається властивість-об'єкт *out*. Цей об'єкт виконує виведення інформації на консоль (текстовий екран). Щоб примусити об'єкт виконати певну дію, слід викликати його відповідний метод. Метод *println* об'єкту *out* друкує дані, що передаються йому у вигляді текстового аргументу, і переводить текстовий курсор на початок наступного рядка (метод print працює

аналогічно, але без переведення курсора). Текстові дані — це рядок символів, обмежених подвійними лапками " і ". Будь-яка команда завершується крапкою з комою. Ієрархія об'єктів та їх методів відокремлюється крапками. Щоб код програми виглядав зрозумілим, усі вкладені блоки прийнято записувати за допомогою відступів. Так заголовок блоку визначає його місце в ієрархії блоків. Дужку { початку тіла блоку прийнято записувати наприкінці заголовка. Дужка } завершення блоку записується в останньому порожньому рядку блока на рівні першого символу заголовка. Команди у тілі блоку записуються з відступом (звичайно виконується за допомогою клавіші Tab). Так дуже чітко переглядається вміст будь-якого блоку програми.

Щоб запустити нашу програму (проект), можна клацнути мишкою відповідний значок на панелі інструментів або натиснути клавішу F6. Хід виконання програми спостерігаємо у нижньому вікні робочої області:



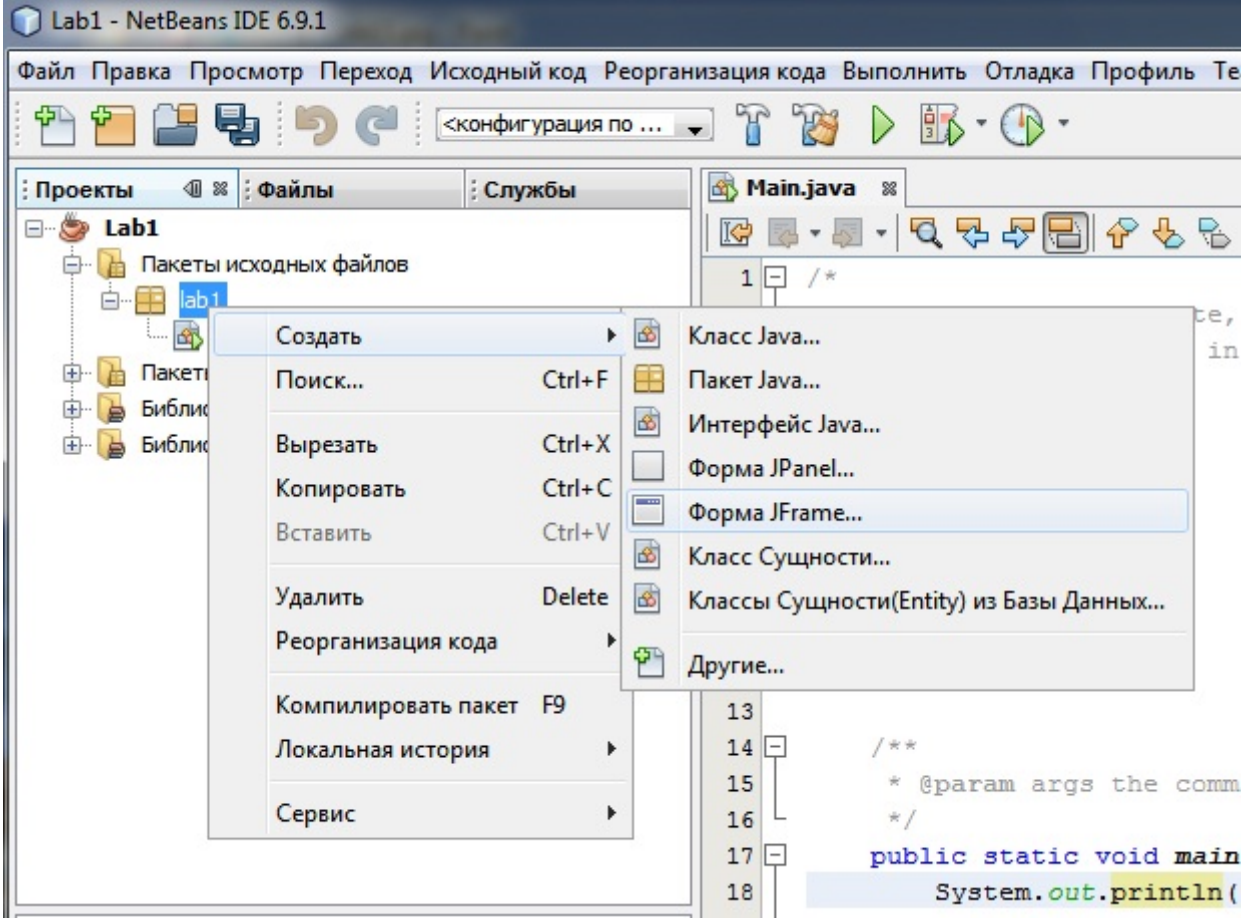
Програми, що виконують введення-виведення даних тільки за допомогою консолі (клавіатури і текстового екрану) називають консольними програмами. Таки програми створюють тоді, коли треба швидко отримати результат роботи деякого алгоритму, а вхідні та вихідні дані подаються виключно у простому символьному вигляді, коли достатньо консольних засобів. Ми створили першу просту консольну програму.

4. Створення проекту з графічним інтерфейсом

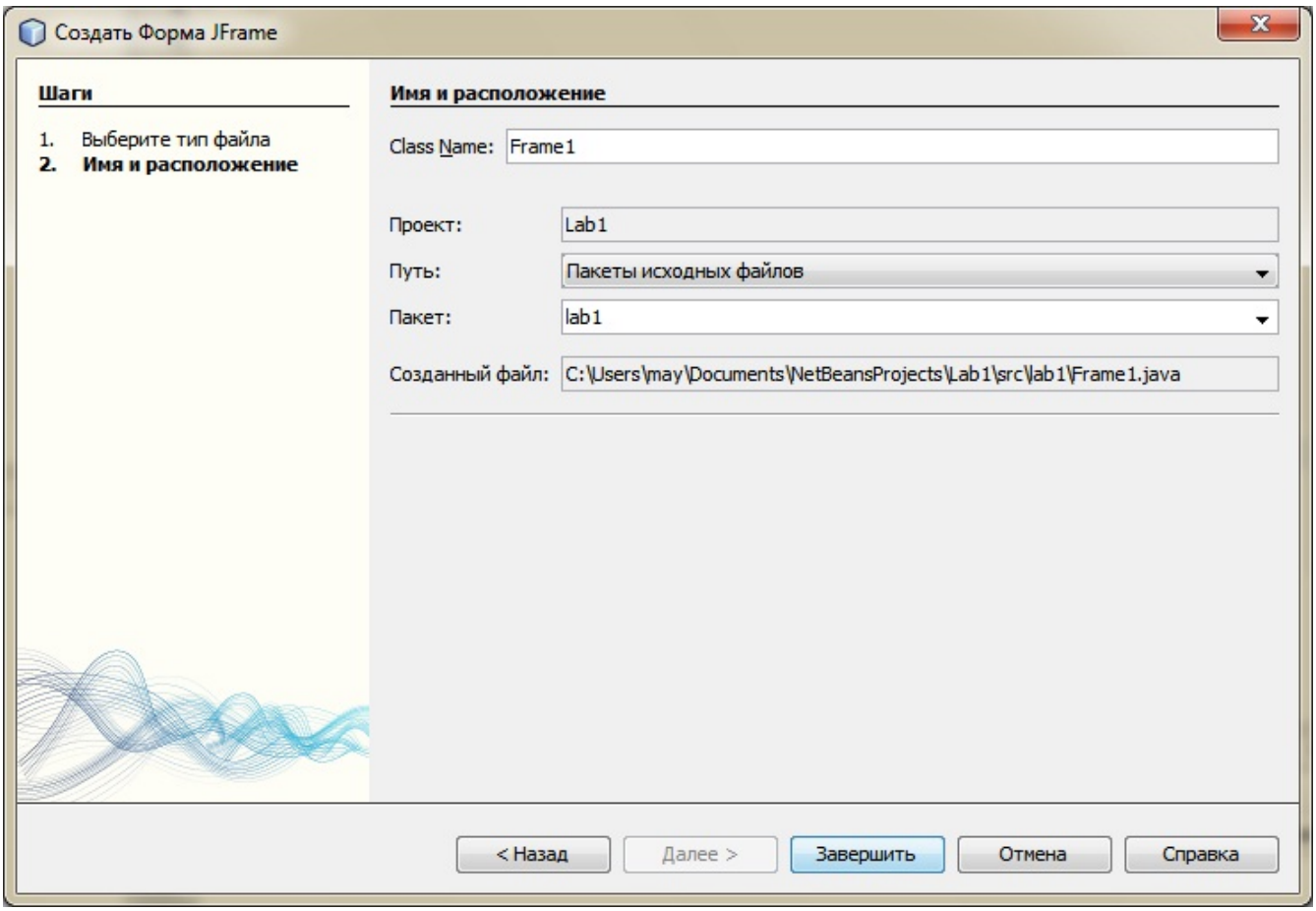
Усі елементи програми з графічним інтерфейсом — вікна, кнопки, поля введення, написи — це об'єкти. Тому в проекті повинні бути описані відповідні класи. Ще одна перевага об'єктного підходу до проектування програм — те, що відповідні класи об'єктів можуть бути створені одними програмістами, а потім багаторазово використовуватися іншими. Причому ті, хто використовує таки класи, може навіть не знати про їх внутрішню реалізацію. Достатньо вивчити декілька необхідних властивостей і методів цих класів, і вміти ними користуватися. Продемонструємо це.

У Java є декілька готових бібліотек класів різноманітних об'єктів, які можна використовувати, так би мовити, на всі випадки життя. Найбільш розповсюджені бібліотеки графічних інтерфейсних елементів — це Swing і AWT.

Продовжимо вже початий проект. Будь-яка сучасна програма з графічним інтерфейсом (у подальшому просто графічна програма) перш за все повинна мати головне вікно, в якому будуть розгортатися інші елементи. У термінах програмування вікно часто називають формою, або фреймом (Frame), тобто рамкою. У відповідності з представленою вище концепцією ми повинні додати до нашого проекту опис такого класу. У NetBeans це можна зробити за допомогою зручного візуального конструктору. Нагадаємо, що наш проект Lab1 складається з одного пакету lab1, який поки що містить лише один клас Main. Тому новий клас будемо додавати саме до пакету lab1. Клацнемо по значку пакету lab1 правою кнопкою миші і оберемо: Создать — Форма JFrame...

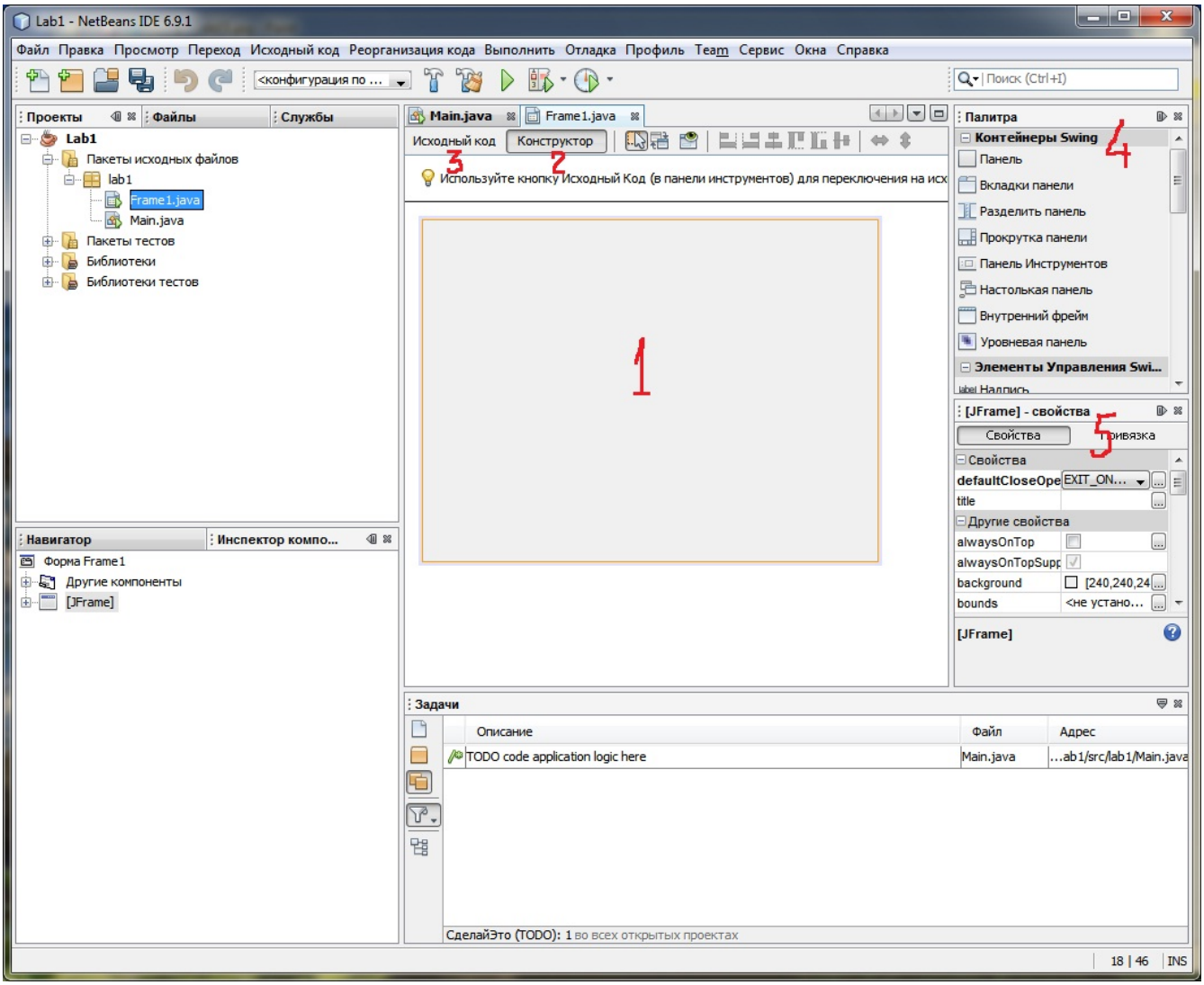


NetBeans пропонує набір готових класів зі стандартних бібліотек Java. Але перш, ніж почати їх використовувати, необхідно виконати деякі налаштування. Після вибору класу форми JFrame з'явиться наступне вікно:



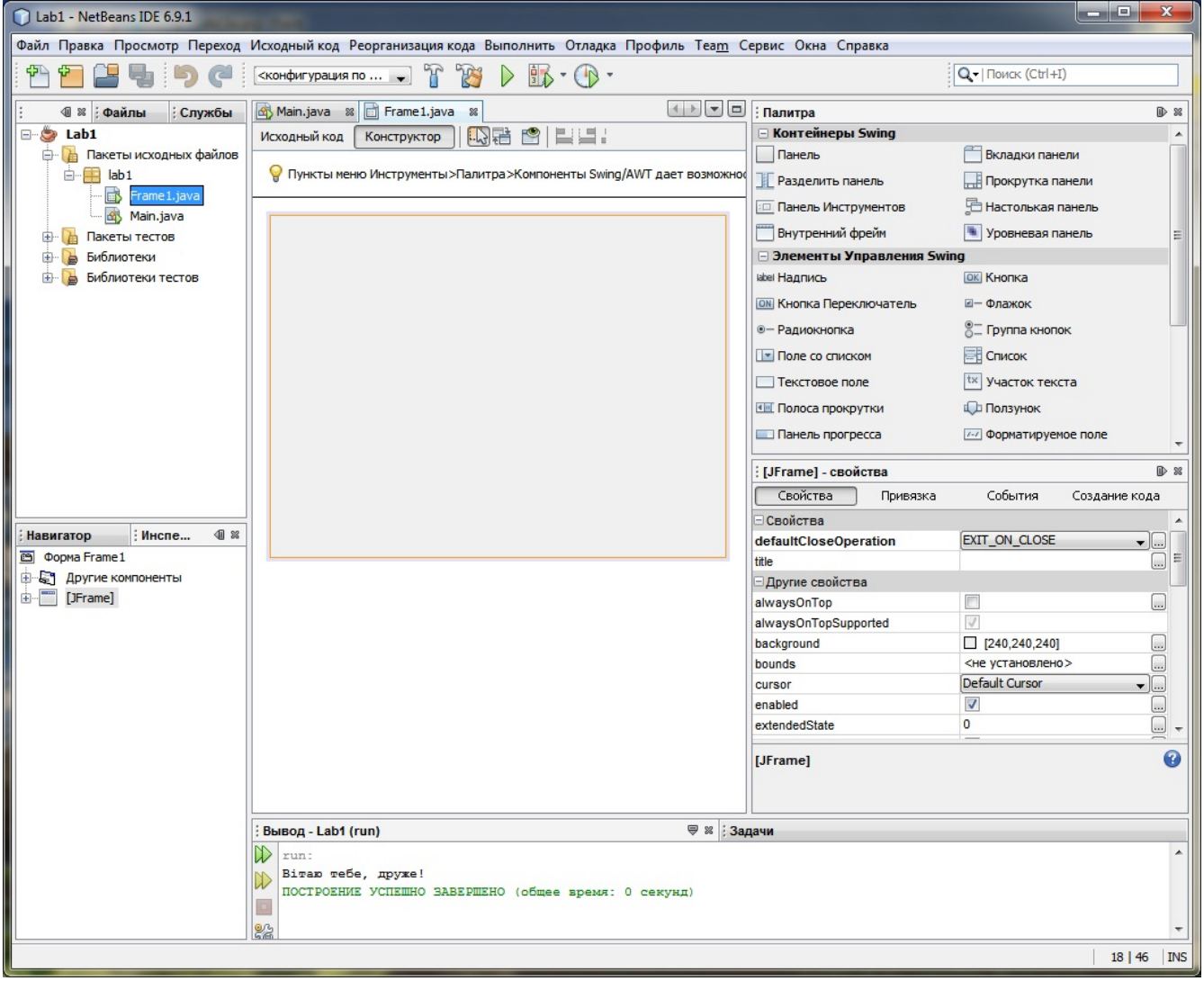
Змінимо ім'я нового класу (Class Name) на Frame1. Обов'язково переконаємося, що назву проекту вказано як Lab1, а назву пакету, до якого додається цей клас — lab1. Інші параметри NetBeans встановлює правильно за

замовченням. Натиснемо кнопку “Завершить” і дочекаємося, поки NetBeans виконає операцію. Отримаємо наступний стан проекту:

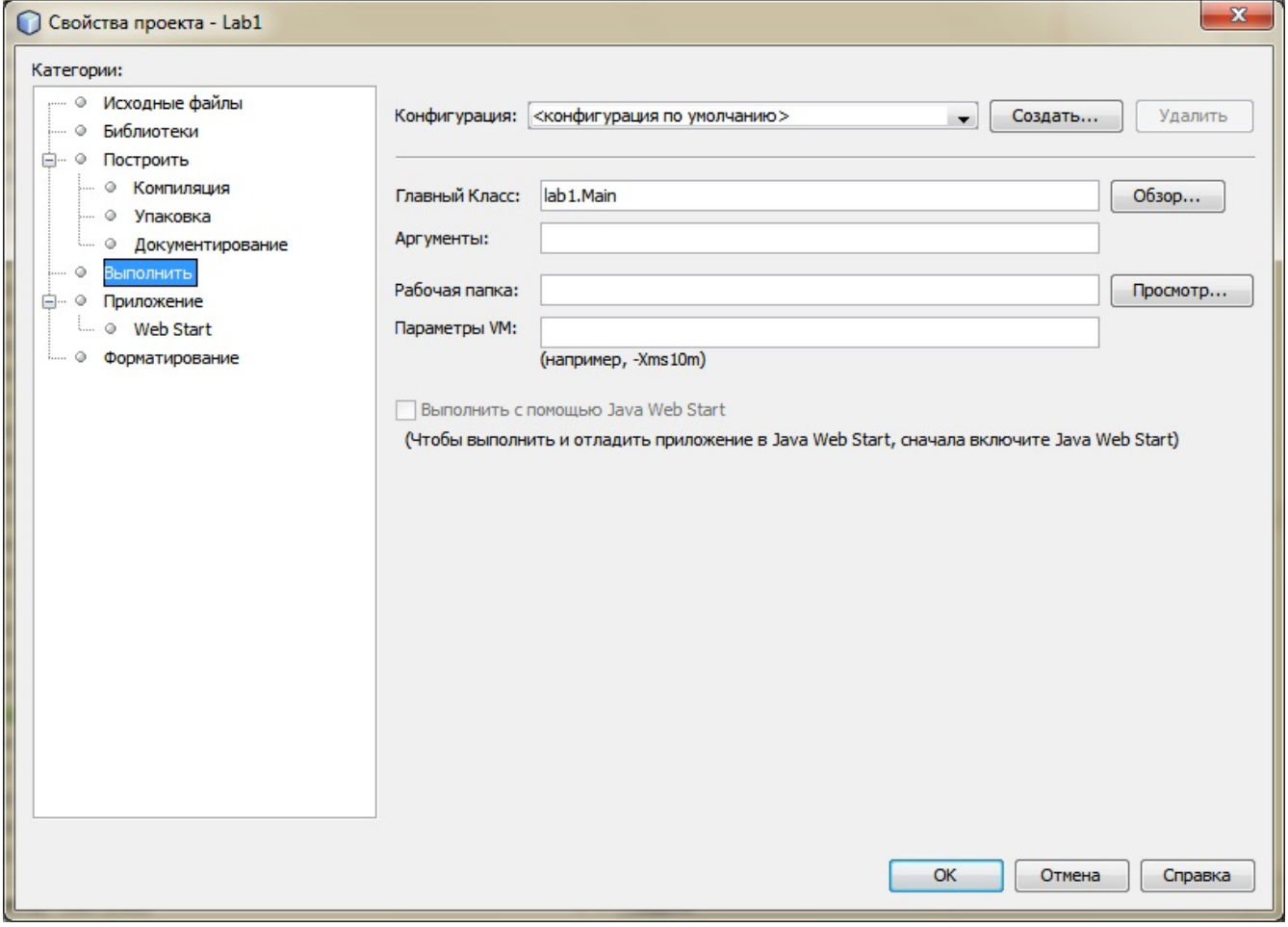


Зверніть увагу, що у вікні проекту пакет lab1 містить вже два файли (класи): Main.java і Frame1.java. У робочій області вміст нового класу завантажується у нову закладку Frame1.java (на рисунку її можна побачити над кнопкою (2) — Конструктор). Оскільки клас Frame1 має візуальне представлення, його можна переглядати у двох варіантах: режимі конструктора (2) і режимі вихідного коду (3), тобто текстового опису. Режим конструктора дуже зручний, оскільки дозволяє візуально, практично без програмування будувати інтерфейс майбутньої програми. При додаванні класу фрейма автоматично на його основі створюється об’єкт, який можна спостерігати у вікні Конструктора візуально у вигляді невеличкої рамки сірого кольору (1). Праворуч від вікна Конструктора розташовуються вікна палітри візуальних елементів (4) і властивостей (5) того елементу, який зараз обраний у вікні конструктора. За допомогою палітри можна продовжити компонувати інтерфейс новими елементами, а за допомогою вікна властивостей — налаштовувати їх.

Зараз у вікні середовища NetBeans ми фактично маємо 6 різних вікон, які розділені тонкими бордюрами. Рекомендуємо перемістити мишкою ці бордюри так, щоб подальша робота була більш зручною. Так вікно проекту можна стиснути, а вікна палітри і властивостей, навпаки, збільшити. Висоту вікна, що відображає хід запуску проекту, також можна зменшити. Після цього вікно середовища може виглядати так:

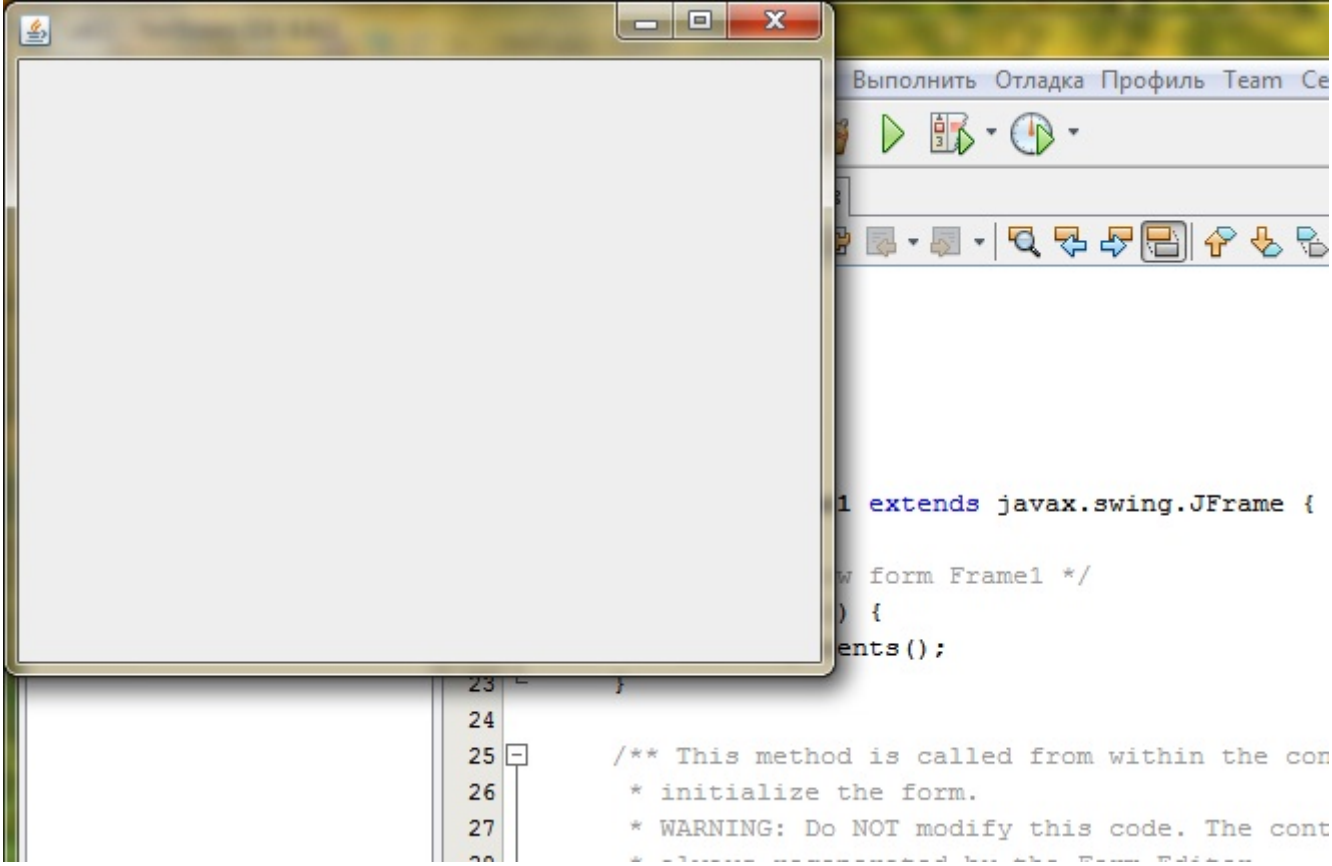


Можна спробувати запустити новий проект (очікуємо побачити невеличке порожнє віконце). Натискаємо клавішу F6 і після декількох секунд з подивом констатуємо: програма продовжує вітати нас (див. результат виводу у самому нижньому віконці) і ніякої порожньої форми не з'являється. Що трапилося? Пригадаємо, що при першому створенні проекту автоматично з'явився клас Main, який було призначено як головний. Тобто середовище продовжує запускати проект, починаючи виконання з методу main() головного класу. А він у нас "вміє" тільки привітатися і нічого "не знає" про клас JFrame. Запуск графічної програми повинен починатися з появи головного вікна-форми. Тому слід обов'язково перемкнути головний клас проекту відразу ж після додавання фрейму. Це робиться за допомогою властивостей проекту. У вікні проекту клацнемо правою кнопкою миші по значку проекту (значок з чашкою кави Lab1) і викличемо вікно властивостей проекту:

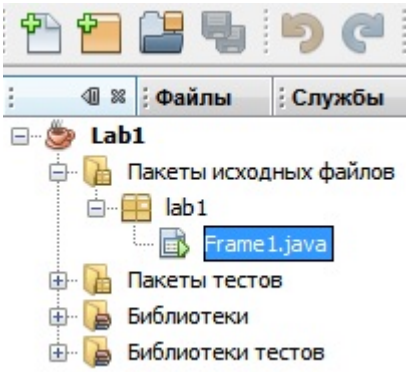


Оберемо категорію “Выполнить” і у вікні праворуч за допомогою кнопки “Обзор...” змінимо головний клас проекту з Main на Frame1.

Зафіксуємо зміни кнопкою “ОК”. Знову ж запустимо наш проект клавішею F6. Зараз вже усе в порядку. Після запуску з’являється порожнє вікно, яке дещо вже вміє: його можна зачепити мишкою за заголовкову смугу і двигати по всій площі екрану, розгортати, згортати і закривати за допомогою кнопок управління вікном у кутку заголовкової смуги. І для всього цього ми поки ще не написали жодного рядка коду вручну. Тільки проста робота мишкою у Конструкторі.



Закриємо запущений проект як звичайне вікно. Зверніть увагу, що зараз вже клас Main зовсім не працює у нашому проекті, ніяких привітань ми більше не бачимо. Цей клас можна або просто залишити, або видалити з проекту (у вікні проекту виділити файл Main.java і натиснути клавішу Delete, або за допомогою контекстного меню за правою кнопкою миші обрати команду “Удалить” — залишиться тільки файл Frame1.java).

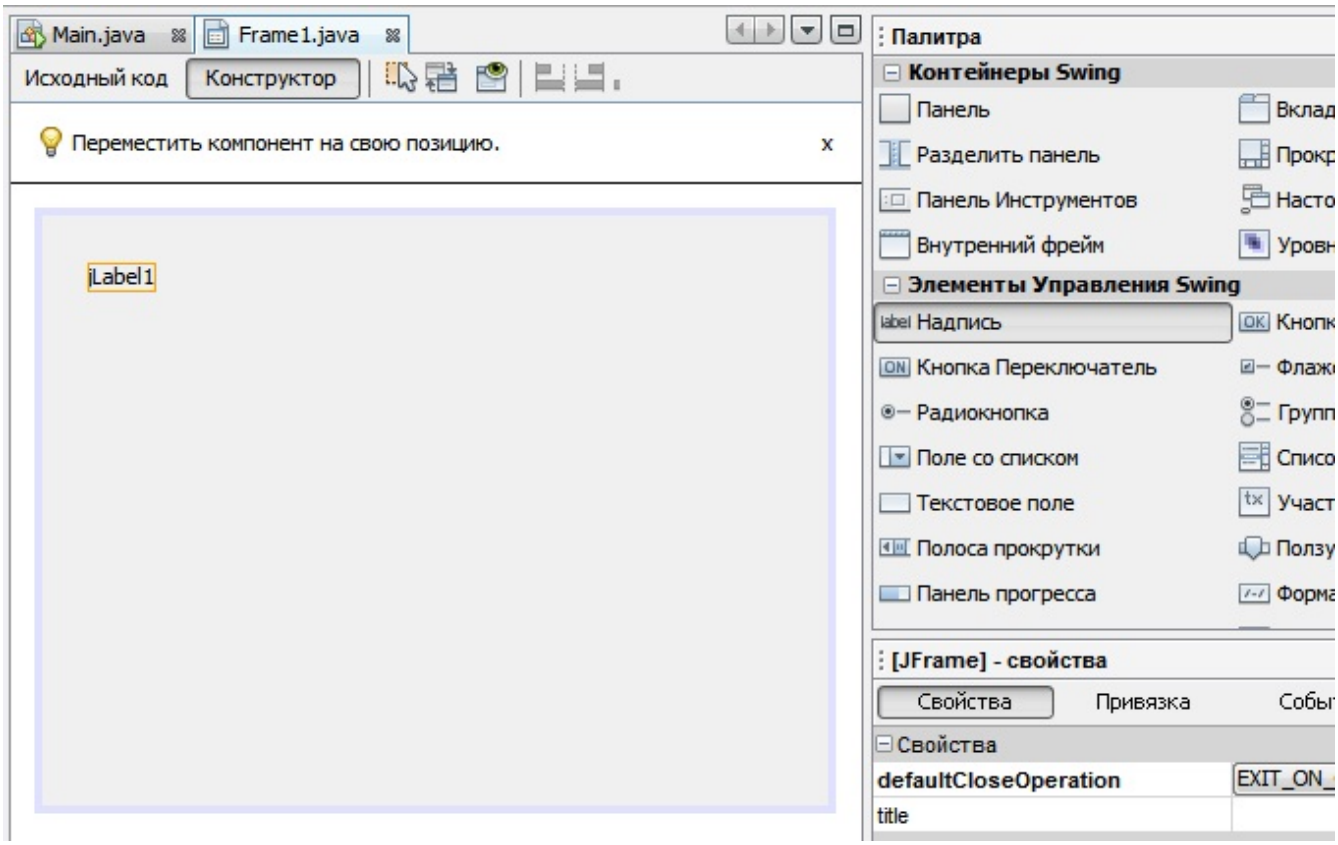


Продовжимо експериментувати з іншими візуальними елементами інтерфейсу. Подивимося, як можна здійснювати введення-виведення даних у графічному (а не консольному) вікні. Зазвичай графічне вікно грає роль контейнера, у якому можуть розташовуватися інші елементи. Не прийнято виводити будь-яку текстову інформацію безпосередньо у вікно. Це робиться за допомогою відповідних елементів-об’єктів.

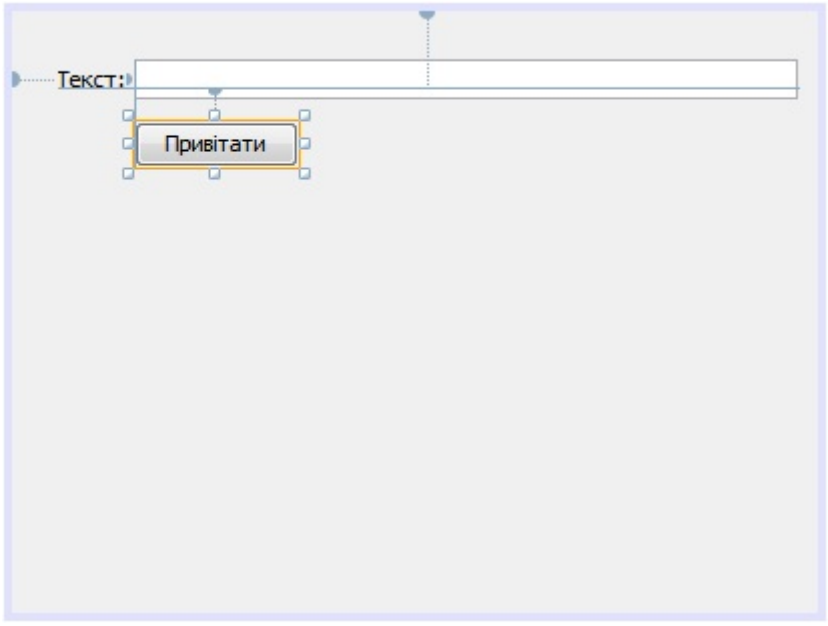
Найбільш вживані елементи для введення-виведення інформації — це Написи і Текстові поля. Напис — це рамка, у яку можна виводити текст, не призначений для редагування. Написи часто використовують для розміщення статичних (не змінних) даних. За допомогою Текстового поля можна як виводити текстові дані (зазвичай невеликого обсягу), так і вводити з клавіатури для подальшої обробки у програмі. Для подачі певних команд з метою виконання необхідних дій дуже часто використовують звичайні Кнопки. Додамо до проекту Текстове поле з Написом, який розташуємо ліворуч від поля і Кнопку, при натисканні якої у Текстове поле буде виводитися привітальний текст. Головне — уявити механізм взаємодії між зазначеними елементами.

Джерело готових візуальних елементів — це Палітра (праворуч від вікна Конструктора). Ми будемо здебільшого використовувати сучасну бібліотеку візуальних графічних класів Swing. Коротким клацанням миші виберемо на Панелі елементів елемент Надпись, перемістимо курсор миші на площу нашої форми Frame1 і після

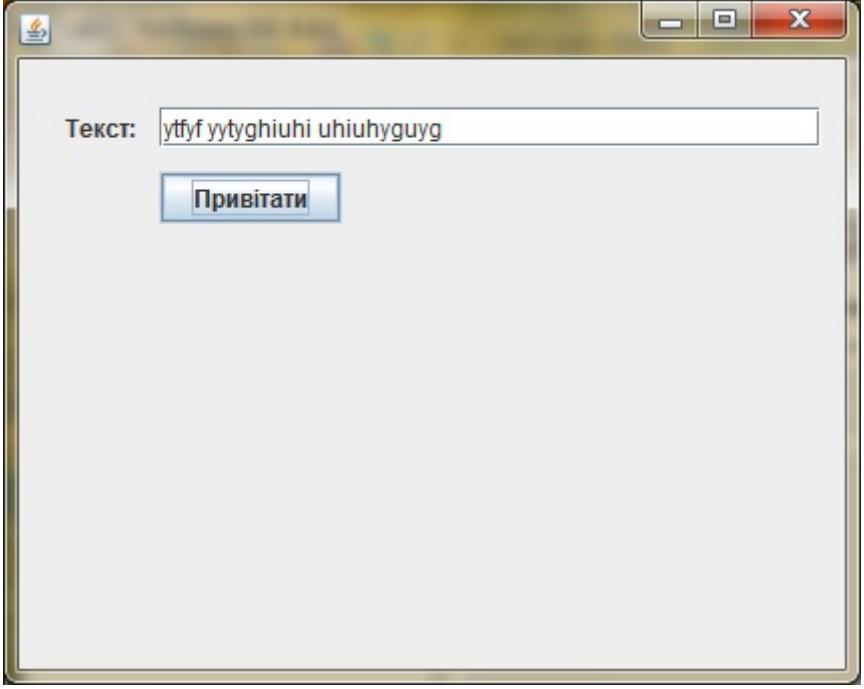
короткої паузи (поки NetBeans сформує новий об’єкт) побачимо новий об’єкт у вигляді маленького прямокутника з написом jLabel1, який просто дублює ім’я нового елементу.



Клацанням миші зафіксуємо елемент у потрібному місці. Усі властивості об’єкту (у тому числі і текст на елементі) можна встановлювати у вікні Властивостей об’єкту (нижче Палітри елементів), але для зручності найбільш вживані операції можна вибрати у контекстному меню елемента за правою кнопкою миші. В нашому випадку виберемо у контекстному меню Напису команду “Изменить текст” і впишемо “Текст:”. Зафіксуємо клавішею Enter. Аналогічним чином додамо до проекту елементи Текстовое поле (напис у самому полі витремо взагалі) і Кнопку. Отримаємо наступний стан проекту:



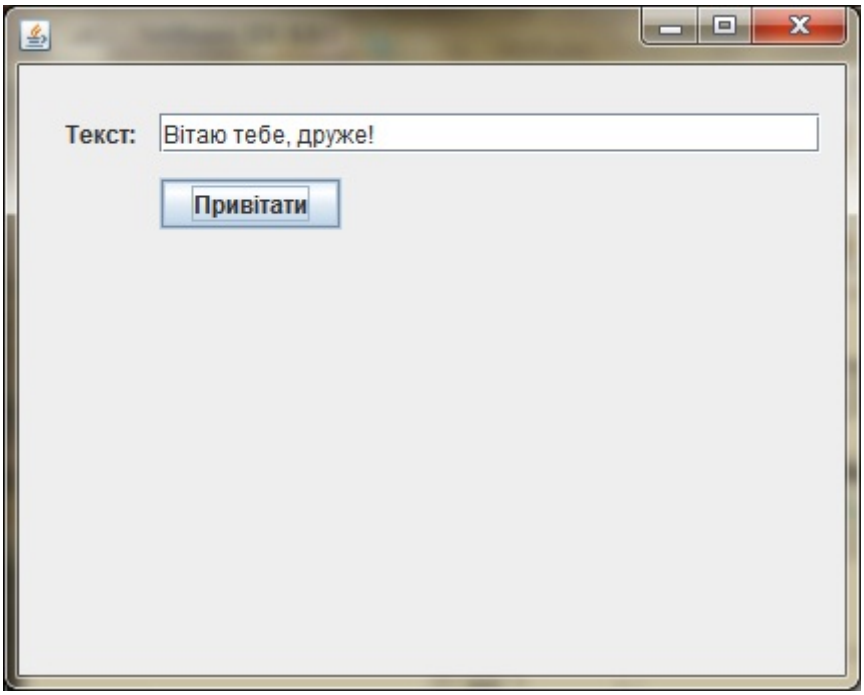
Якщо запустимо проект на виконання (F6), виявимо, що у поле можна вводити і редагувати будь-який текст, але натискання на кнопку ні до чого не приводить.



Закриємо програму і повернімося до проекту. Щоб примусити об’єкт реагувати на певні дії користувача (натискання клавіші на клавіатурі, клацання мишею), слід запрограмувати відповідний метод об’єкта, який автоматично викликається на конкретну подію користувача. Такі методи прийнято називати *обробниками подій*. У контекстному меню нашої кнопки виберемо “События — Mouse — MouseClicked”, після чого NetBeans переведе нас у вікно вихідного коду, де з’явиться шаблон методу, назва якого складається з назви об’єкта (jButton1) і назви події (MouseClicked — клацання мишею). У круглих дужках вказаний аргумент — об’єкт події, яка передається від користувача. Всередині методу впишемо наступну команду:

```
77      private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
78          jTextField1.setText("Вітаю тебе, друже!");
79      }
80
```

Щоб змусити об’єкт щось зробити (вивести текст всередині текстового поля), слід звернутися до об’єкту за іменем (jTextField1) і викликати його відповідний метод (метод setText встановлює/виводить на об’єкті вказаний в аргументі текст). Текстові (рядкові) дані потрібно укладати в подвійних лапках. Запуск (F6) і перевірка показують, що все чудово працює.



Звіт по роботі

Звіт слід оформити у текстовому документі формату .odt за допомогою текстового редактора OpenOffice і надіслати викладачеві на перевірку. У звіті потрібно указати заголовок роботи, сформулювати мету, яку ви досягли у процесі виконання цієї роботи, навести графічну копію вікна середовища NetBeans окремо у режимі вихідного коду і окремо — конструктора остаточного стану свого проекту. Сформулюйте висновки за основними підсумками роботи. Відповіді на наступні питання допоможуть вам зробити це:

1. З чого складається програмний проект на мові Java?

2. Чим відрізняється клас від об’єкту?

3. З чого складається опис будь-якого класу?

4. Як здійснюється запуск проекту (тобто, що робить Java після запуску) і що робити, якщо після запуску проекту нічого не відбувається?

5. Як створюється і налаштовується візуальний об’єкт проекту?

6. Яким чином здійснюється введення-виведення даних у консольних і графічних програмах?

7. Як примусити об’єкт щось зробити у ході виконання програми?

Для тих, хто не знає, як зробити графічну копію вікна середовища: знаходячись у вікні NetBeans (активне вікно), натисніть клавіші Alt+PrintScreen — копія зображення вікна поміщається в буфер обміну (просто PrintScreen зробить копію всього екрану, а не тільки вікна NetBeans). Перейдіть у вікно текстового редактора і у потрібну позицію курсора зробіть вставку з буферу обміну — у документі з’явиться рисунок.