Зміст (коротко)

Зміст (докладно)

Вступ

Ваш мозок і Java. Коли ви намагаєтеся щось вивчити, ваш мозок намагається надати вам послугу, переконуючи, що все це не варто уваги. Він думає: «Краще перейматися важливішими речами, наприклад, небезпечними дикими тваринами або тим, що катання голяка на сноуборді — погана ідея». Як же переконати свій мозок у тому, що від знання Java залежить ваше життя?

1 Відкидаючи зайве

Java відкриває нові МОЖЛИВОСТІ. Від часів скромного релізу відкритої (і доволі вбогої) версії 1.02, ця мова підкорила розробників своїм дружнім синтаксисом, об'єктно-орієнтованими функціями, управлінням пам'яттю і, що найважливіше, перспективою перенесення на різні платформи. Ми швиденько зануримося, напишемо код, скомпілюємо й запустимо його. Маємо на увазі синтаксис, цикли, розгалуження і все, що робить Java такою кльовою. Отже, починаємо.

2 Подорож до Об'єктвіля

Мені казали, що там будуть об'єкти. У Розділі 1 ми розмістили весь свій код у методі main(), але це не зовсім об'єктно-орієнтований підхід. Тепер ми маємо залишити позаду цей світ процедур і почати розробляти власні типи об'єктів. Ми розглянемо, чим цікаве об'єктно-орієнтоване (ОО) програмування за допомогою мови Java. Також розглянемо різницю між класом та об'єктом і дізнаємося, як об'єкти можуть покращити ваше життя.

3 Знайте свої змінні

Змінні бувають двох видів: примітиви і посилання.

У житті має бути щось окрім чисел, рядків і масивів. Як бути з об'єктом PetOwner у якого є змінна типу Dog? Або Car зі змінною Engine? У цьому розділі ми розкриємо завісу таємниці над типами в мові Java й розглянемо, що саме можна *оголошувати* як змінну, які значення *присвоювати* їм і що взагалі можна з ними *зробити*. І нарешті, розглянемо як насправді виглядає життя в кучі (heap), яка контролюється сміттєзбиральним механізмом (garbage-collectible heap).

Тут heap, або «купа»,— це спеціальна ділянка пам'яті, яка контролюється автоматично і створена для впорядкування об'єктів.

Посилання на об'єкт Dog

4 Як поводяться об'єкти

Стан впливає на поведінку, а поведінка — на стан.

Нам відомо, що об'єкти характеризуються **станом** і **поведінкою**, що представлені **змінними екземпляра** і **методами**. Зараз ми обговоримо *зв'язок* між станом і поведінкою. Поведінка об'єкта використовує унікальний стан об'єкта. Інакше кажучи, **методи використовують значення змінних екземпляра.** Скажімо, «якщо собака важить менше 14 фунтів, треба видати звук ***yipp***, інакше...» **Давайте змінимо стан!**

5 Надпотужні методи

Давайте зробимо метод потужнішим. Ви вже побавилися зі змінними, погралися з декількома об'єктами і написали невеличкий код. Але для повноцінної роботи потрібно більше інструментів. На кшталт **операторів** і **циклів**. Це може бути корисним для **генерації випадкових чисел.** І перетворення об'єкта String на Int — так, це було б круто. Чому б вам не створити щось справжнє, аби на власні очі побачити, як з нуля пишуться (і тестуються) програми? **Можливо, це буде гра** на кшталт Sink a Dot Com (подібна до «Морського бою»).

6 Використання бібліотеки Java

Разом із Java постачаються сотні готових класів. Можете не витрачати час на винахід власного велосипеда, якщо знаєте, як знайти необхідне в бібліотеці Java (її також називають Java API). Гадаю, у вас знайдуться важливіші справи. При написанні коду зосередьтеся на тій частині, що є унікальною для вашого застосунку. Базова бібліотека Java являє собою гігантський набір класів, готових до застосування як будівельні блоки.

«Приємно знати, що ArrayList міститься в пакеті java.util. Але яким чином я змогла б додуматися до цього самостійно?»

Джулія, 31 рік, модель

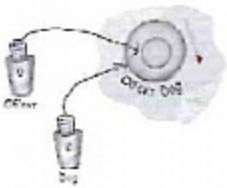
7 Покращене життя в Об'єктвілі

Плануйте свої програми з прицілом на майбутнє. Чи легко написати код так, щоби хтось інший міг би його розширити? Чи зацікавить вас створення гнучкого коду, що не боїться змін у технічному завданні, що виникають в останню мить? Долучившись до планування поліморфізму, ви дізнаєтеся про п'ять кроків вправного проєктування класів, три прийоми задля досягнення поліморфізму і вісім способів створення гнучкого коду, а якщо діятимете енергійно — отримаєте бонусний урок із чотирма порадами щодо використання наслідування.

8 Серйозний поліморфізм

Наслідування — це лише початок. Аби задіяти поліморфізм, необхідні інтерфейси. Нам слід вийти за рамки простого наслідування, щоби ви могли отримати гнучкість лише проєктуючи та кодуючи інтерфейси. Що таке інтерфейс? Це стовідсотково абстрактний клас. Що таке абстрактний клас? Це клас, для якого не можна створити екземпляр. Навіщо це потрібно? Читайте розділ...


```
Object o = al.get(index);
Dog d = (Dog) o;
d.bark();
```



Приводимо Object до Dog, який, як ми знаємо, ним і є.

9 Життя і смерть об'єктів

Об'єкти народжуються і вмирають. Ви головний. Ви вирішуєте, коли і як саме створювати їх. І ви вирішуєте, коли відмовитися від них. **Збирач сміття (Garbage Collector — gc)** звільняє від них пам'ять. Ми розглянемо, як створюються об'єкти, де розміщуються, як їх ефективно використовувати і звільняти пам'ять від них, коли вони стають непотрібними. Тобто ми обговорюватимемо купу (heap), стек (stack), зону впливу (skope), конструктори, суперконструктори, null посилання і вимоги gc.

Змінній «d» призначено новий об'єкт Duck, при цьому оригінальний (початковий) об'єкт Duck стає недосяжним. Та перша комка відтепер засмажена.

10 Числа мають значення

Займемося математикою. Java API має методи для роботи з числами на кшталт визначення абсолютного значення, округлення, пошук мінімуму / максимуму тощо. А як щодо форматування? Можливо, ви захочете, щоби числа друкувалися із точністю до двох десяткових знаків або з комами в усіх необхідних місцях. Ви також можете друкувати дати і маніпулювати ними. А як щодо розбору рядка в число? Або перетворення числа на рядок? Ми почнемо цей розділ із вивчення особливостей *статичної* змінної або методу.

11 Небезпечна поведінка

Трапляються різні речі. То файл зникає. То сервер падає. Хай би яким вправним розробником ви були, неможливо контролювати геть усе. При створенні ризикованого методу вам знадобиться код, що буде обробляти можливі нестандартні ситуації. Але яким чином дізнатися, чи ризиковий цей метод? І куди помістити код для обробки *непередбаченої* ситуації? У цьому розділі ми розробимо музичний MIDI-плеєр, що використовує ризикове JavaSound API.

12 Дуже графічна історія

Готуйтеся, вам доведеться створювати GUI (*graphical user interface* — **графічний** користувацький інтерфейс). Навіть якщо ви впевнені, що все життя будете писати код для серверних програм, рано чи пізно вам доведеться створювати інструменти, і тоді виникне потреба у використанні графічного інтерфейсу. Роботі над GUI присвячені два розділи, із яких ви дізнаєтеся про ключові особливості мови Java, включно з **обробкою подій** і **внутрішніми класами.** Ми розмістимо кнопку на екрані, малюватимемо на екрані, додаватимемо зображення у форматі JPEG і навіть створюватимемо анімацію.

```
class MyOuter {

    class MyInner {
      void go() {
      }
    }

}
```

Ці два об'єкти на купі мають особливий зв'язок. Внутрішній може використовувати змінні зовнішнього (і навпаки).

13 Попрацюйте над Swing

Swing — це просто. Доти, доки вам байдуже, де ви опинитесь. Код для роботи зі Swing виглядає просто, але, скомпілювавши його, запустивши і подивившись на екран, ви подумаєте: «Агов, цей об'єкт має бути в іншому місці!» Інструмент, що *спрощує створення* коду і водночас *ускладнює управління* ним,— це **менеджер** макету інтерфейсу **(Layout Manager)**. Проте, доклавши невеликих зусиль, ви зможете підкорити його. У цьому розділі ми будемо працювати зі Swing і ближче познайомимося з віджетами.

Компоненти на півночі та півдні отримують бажану висоту.

Компоненти на заході та сході отримують бажану ширину.

14 Збереження об'єктів

Об'єкти можуть бути стиснутими та відновленими. Вони характеризуються станом і поведінкою. Поведінка міститься в класі, а стан визначається кожним об'єктом окремо. Якщо вашій програмі необхідно зберегти стан, ви можете зробити це складним способом, опитуючи кожен об'єкт і старанно записуючи значення кожної змінної екземпляра. Або ви можете **зробити це легким ОО-способом** — просто застосуйте суху заморозку (серіалізуйте об'єкт) та відновіть (десеріалізуйте), аби отримати його знов.

Ми вважаємо читача «Head First» учнем

Яким чином ми щось *дізнаємося*? Спочатку потрібно це «щось» *зрозуміти*, а потім *не забути*. Заштовхати в голову безліч фактів недостатньо. Згідно з даними новітніх досліджень в галузі когнітивістики, нейробіології та психології навчання, для засвоєння матеріалу потрібно щось більше, ніж текст на сторінці. Ми знаємо, як змусити ваш мозок працювати.

Основні принципи серії «Head First»:

Зробіть це наочним. Графіка запам'ятовується значно краще за звичайний текст і значно підвищує ефективність сприйняття інформації (до 89 % підвищується пригадування та відтворення вивченого). До того ж, матеріал стає зрозумілішим. Якщо **розташувати відповідний текст на малюнках або поряд з ними**, а не під ними або на сусідній сторінці, учні отримають удвічі більше шансів зрозуміти зміст і вирішити проблеми, які описуються.

Погано бути абстрактним методом. Доводиться обходитися без тіла.

```
abstract void roam();
```

Метод не має тіла! Не забудьте поставити крапку з комою.

Використовуйте розмовний та персоналізований стиль викладу. Нещодавні дослідження показали, що при особистісному (направленому на читача напряму) розмовному стилі викладу матеріалу, замість формальних лекцій поліпшення результатів на підсумковому тестуванні складало до 40 %. Розповідайте історію замість того, щоби читати лекцію. Не ставтеся до себе занадто серйозно. Що ймовірніше приверне вашу увагу: цікава бесіда за вечерею або лекція?

Чи можна сказати, що ванна є (IS-A) ванною кімнатою? Ванна кімната є (IS-A) ванною? Або це взаємопов'язані (HAS-A) речі?

Змусьте учня мислити глибше. Доки ви не почнете ворушити звивинами, у вашій голові нічого не відбуватиметься. Читач повинен бути зацікавлений у результаті; він повинен натхненно вирішувати завдання, формулювати висновки і генерувати нові знання. А для цього необхідні вправи і каверзні питання, у пошуку відповідей на які задіяні обидві півкулі мозку й різноманітні почуття.

Приверніть увагу читача — і зафіксуйте її. Ситуація, знайома кожному: «Я дуже хочу вивчити це, але засинаю вже після першої сторінки». Мозок звертає увагу на цікаве, дивне, привабливе, несподіване. Вивчення складної технічної теми не має бути нудним. Ваш мозок осягне це набагато швидше, якщо це буде цікаво викладено.

Залучення емоцій. Відомо, що наша здатність запам'ятовувати значною мірою залежить від емоційного співпереживання. Ми запам'ятовуємо те, що нам не байдуже. Ми запам'ятовуємо, коли щось відчуваємо. Ні, ми не маємо на увазі хвилюючі історії про хлопчика та його собаку. Мова йде про такі емоції, як здивування, цікавість, інтерес і відчуття власної значущості при вирішенні задачі, яку оточення вважає надто складною, або коли ви зрозуміли, що розбираєтеся в темі краще за всезнайку Боба з технічного відділу.