Класи й об’єкти

№ уроку: 2 **Курс:** Java Essential

Засоби навчання: Комп'ютер із встановленою IntelliJ IDEA

Огляд, мета та призначення уроку

- Розгляд поняття «Конструктор», keyword «this».
- Перенавантаження конструкторів.
- Конструктор за замовчуванням.
- Приватний конструктор.
- Keyword «static».

Вивчивши матеріал цього заняття, учень зможе:

- Оперувати знаннями з конструкторів.
- Створювати конструктори за замовчуванням і «default».
- Знати та розуміти ключові слова «this», «static».

Зміст уроку

1. Розглянути поняття конструктора.
2. Оголошення конструктора.
3. Ключове слово **this**.
4. Розглянути значення та застосування конструктора за замовчуванням.
5. Розглянути, коли конструктор за замовчуванням не створюється.
6. Розглянути створення та використання приватного конструктора.
7. Перевантаження конструкторів.
8. Створення конструкторів із конструкторів.
9. Статичні елементи.

Резюме

Конструктор. Оголошення конструктора виглядає, як оголошення методу, тільки для цього ми використовуємо ім'я класу. Коли після ключового слова **new** ми пишемо ім'я класу з дужками, то насправді ми викликаємо конструктор класу з тими параметрами, які вкажемо у дужках (передамо на вхід конструктора).

Є два різновиди конструкторів: «з параметрами» та «за замовчуванням». Конструктори не можуть бути synchronized, final, abstract, native, static.

this – це посилання на нинішній об'єкт (об'єкт, конструктор якого був викликаний), тобто на екземпляр класу, в якому розгортається дія.

Конструктор:

1. Конструктор створюється з полів, які оголошені у класі.
2. Якщо у класі явно не створено конструктор, то компілятор створить конструктор сам, за замовчуванням.
3. Якщо ми створили конструктор з параметрами, конструктор за замовчуванням не створюється. Якщо він нам буде потрібний для роботи, то його треба буде створювати руками.
4. Також конструктор може мати модифікатор доступу **private**. Це означає, що об'єкти з використанням параметрів цього конструктора можуть створюватися тільки в межах класу, де конструктор оголошений.

Перевантаження конструкторів. Перевантаження – це коли сигнатура конструкторів відрізняється одна від одної.

Також можна створювати конструктори з конструкторів за допомогою ключового слова **this**.

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

t. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Під час виконання конструктора **this()** спочатку виконується перевантажений конструктор, який відповідає списку параметрів, потім виконуються оператори, що розташовані всередині вихідного конструктора, якщо такі наявні. Виклик конструктора **this()** має бути першим оператором конструктора.

static позначає зв'язок із класом, а не з його об'єктом.

Слово **static** застосовується з:

- полями;
- методами;
- класами:
- блоками ініціалізації;
- імпортами.

Якщо необхідно, щоб змінні були спільними для всіх об'єктів, то застосовується **static**. Поля, які мають статичний модифікатор в оголошенні, називаються статичними полями або змінними класу. Кожен екземпляр класу поділяє змінну класу, яка розташована в одному місці в пам'яті.

static модифікатор у поєднанні з **final** модифікатора використовується також для визначення констант.

Статичні методи мають перевагу у застосуванні, оскільки відсутня потреба щоразу створювати новий об'єкт для доступу до таких методів. Статичний метод можна викликати, використовуючи тип класу, де ці методи описані. Саме тому подібні методи якнайкраще підходять як методи-фабрик (factory) і методи-утиліт (utility). Клас java.lang.Math – чудовий приклад, в якому майже всі методи статичні, тому класи-утиліти в Java фіналізовані (final).

final модифікатор показує, що значення цього поля не може бути змінено.

Java Naming Convention говорить, що константи мають писатися у верхньому регістрі. Відокремлювати слова треба нижнім підкресленням «_».

Статичні методи мають бути викликані ім'ям класу, без необхідності створення екземпляра класу:

ClassName.methodName(args);

Ви також можете звернутися до статичних методів з об'єктного посилання, але це не рекомендується, бо стає не зрозуміло, що методи є методами класу.

Зазвичай статичні методи використовуються, щоб звернутися до статичних полів. Так само **static** не може використовувати з **this** (бо this це посилання на нинішній об'єкт).

Підсумки:

1. static для класів означає, що клас є допоміжним і залишатиметься таким як є (так звані статичні вкладені класи. За винятком класів верхнього рівня).
2. Статичні методи не мають залежності від даних екземпляра.
3. Деякі дані, які необхідно використовувати кількома екземплярами, позначаються статичними. Замість їхнього створення щоразу, ми створюємо спільний для всіх.
4. Статичні методи не можуть бути **overridden**.

Закріплення матеріалу

- Що таке конструктор?
- Що таке конструктор за замовчуванням, приватний?
- Що означає **this**?
- Що таке навантаження конструкторів?
- Як створювати конструктор із конструктора.
- Як застосовується статика із полями?
- Як використовується статика у методах?
- Перерахуйте, де статика може використовуватися.

Додаткове завдання

Використовуючи Intelij IDEA, створити проєкт, пакет.
Створити клас MyArea, у ньому оголосити константу PI = 3.14 і статичний метод areaOfCircle, який має приймати радіус, і, використовуючи PI, порахувати площу кола.
Створити клас Main, де запустити цей метод.

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

t. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com

Самостійна діяльність учня

Завдання 1
У будь-якій з профільних книг **(Горстманн, Еккель)** знайти відповідні теми та закріпити матеріал. Використання **YouTube, Quizful** вітається.

Завдання 2
Використовуючи Intelij IDEA, створити проєкт, пакет.
Створити клас Машина з полями рік(int), колір(String).
Створити конструктор за замовчуванням, конструктор з одним і двома параметрами.
Створити клас Main, в якому створити екземпляри, викликаючи різні конструктори.

Завдання 3
Використовуючи Intelij IDEA, створити проєкт, пакет.
(Наново!) Створити клас Машина з полями рік(int), швидкість(double), вага(int) колір(String).
Створити конструктор за замовчуванням, конструктор із 1-м параметром, 2-а, 3-я, 4-а.
Перевантажити конструктори.
Створити клас Main, де створити екземпляри класу Машина з різними параметрами.

Завдання 4
Використовуючи Intelij IDEA, створити проєкт, пакет.
(Наново!) Створити клас Машина з полями рік(int), швидкість(double), вага(int) колір(String).
Створити конструктор за замовчуванням, конструктор із 1-м параметром, 2-а, 3-а, 4-а.
Перевантажити конструктори викликаючи конструктор із конструктора.
Створити клас Main, де створити екземпляри класу Машина з різними параметрами.

Рекомендовані ресурси

https://docs.oracle.com/javase/tutorial/java/javaOO/thiskey.html

https://docs.oracle.com/javase/tutorial/java/javaOO/constructors.html

https://docs.oracle.com/javase/tutorial/java/javaOO/classvars.html

CyberBionic Systematics ®
19 Eugene Sverstyuk Str., 5 floor
Kyiv, Ukraine

t. 0 800 337 146
E-mail: edu@cbsystematics.com
Site: www.edu.cbsystematics.com | itvdn.com