О курсе. Java Core.

Составил: Черниговский А.С.
Старший преподаватель кафедры “Информатика”
ИКИТ СФУ

О чем этот курс?

- Ранее читался курс “Программирование на языке Java”
- Он был не совсем интересен, т.к. содержал в себе простые главы и студенты ранее уже программировали то же самое на занятиях по дисциплине “Объектно-ориентированное программирование”
- Данный курс читается последний год, после чего будет читаться курс “Проектирование корпоративных информационных систем”

О чем этот курс?

- Поэтому уже в данном семестре было принято решение изменить курс и опрабировать
- Ожидается, что слушатели уже обладают начальными знаниями о языке Java и ООП, этому будет посвящена только первая часть курса
- Остальная (бОльшая) чать курса посвящена фреймворку Spring.
- Фреймворк Spring – популярный, современный фреймворк который в короткие сроки позволяет построить свою собственную информационную систему корпоративного уровня

Было/Стало

- Основы
- Проектирование классов
- Наследование
- Интерфейсы
- Исключения
- Потоки данных
- Обобщенное программирование
- Многопоточность

- Java Core
- Внедрение зависимостей в Spring
- Аннотирование в Spring
- Работа с БД в Spring
- Spring MVC
- *В разработке*
- *В разработке*
- *В разработке*

Структура курса

- 8 тем, 8 лекций, 8 практических работ (может 7)
- Введение в курс — вводное тестирование, для проверки остаточных знаний (если такие имеются)
- Первая тема — Java Core за две недели
- Все остальное — изучение Spring
- В начале изучения Spring вам может все показаться абсолютно непонятным и/или ненужным, но необходимо немного подождать. В последствии вам станет понятно зачем все это нужно
- Результатом выполнения курса будет ожидаться создание простенького сайта-сервиса. Но это неточно.

Структура курса

- Для каждой темы будут предусмотрены:
- Лекция в виде слайдов (по большей части предназначена для преподавателя, для последующего чтения аудитории)
- Конспект лекции (Основной необходимый материал для усвоения темы и сдачи практической работы, содержит материал читаемый на лекции для последующей подготовки к сдаче работы)
- Короткий пример (Для тех, кому сложно разобраться самостоятельно, или вообще ничего непонятно, рекомендуется сначала повторить простой пример, а затем приступать к выполнению задания)
- Практическое задание для самостоятельно выполнения и последующей защиты
- Тесты на еКурсах обязательны для выполнения

Состояние курса

- В настоящее время (03.09.2020) в курсе готово 4,5 темы
- Вероятно, полностью курс будет готов к его окончанию (к декабрю)
- Тесты и материал будут добавляться в течении семестра по мере прохождения курса
- В случае возникновения вопросов, выявления опечаток или обнаружения неточностей в каких-либо материалах прошу не стесняясь обращаться к Черниговскому А.С. в личных сообщениях на еКурсах или по электронной почте achernigovskiy@sfu-kras.ru

Аттестация

- Итоговая аттестация – зачет
- Каждая работа оценивается по трехбалльной системе: 3, 4, 5
- Чтобы получить зачет “автоматически” необходимо получить балл (среднее арифметическое за все практические работы) выше, чем 4,3. Все тесты должны быть выполнены >50% верных ответов
- Сначала практическая работа должна быть защищена преподавателю. Затем, отчет о практической работе обязательно должен быть загружен в соответствующий элемент курса. Временем сдачи работы является загрузка отчета на электронный курс.
- Для практической работы установлены крайние сроки сдачи (будут прописаны в электронном курсе), в случае несвоевременной сдачи работы оценка за работу снижается на 1 балл. Если сдача работы просрочена более чем на 2 недели, то оценка снижается на 2 балла.

Аттестация

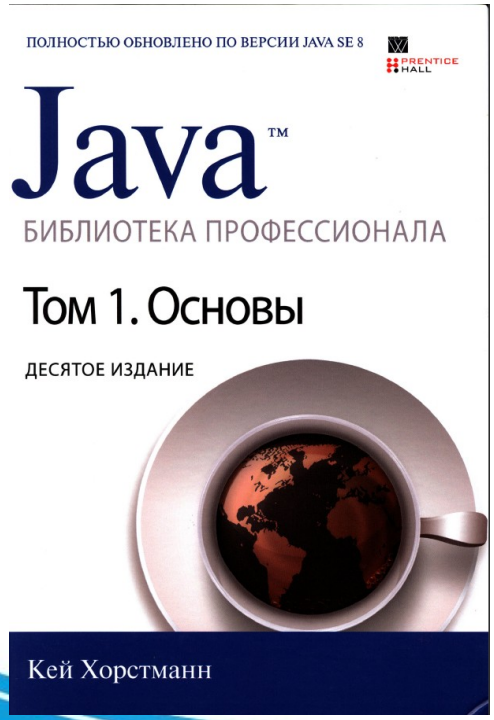
- В случае если вы не набрали необходимое число баллов, то вас ожидает зачет
- Зачет проводится только при всех выполненных тестах
- Зачет может быть получен двумя способами:

  1) Сдачей работ (которые не были сданы в течении семестра) + дополнительный вопрос по другой теме.

  2) Написание зачета. По каждой несданной теме дается 2 теоретических вопроса. При сдаче зачета студенту могут быть заданы дополнительные вопросы преподавателя.
- Вопросы на зачет появятся в конце семестра

Ну а что почитать?

- По Java Core



- По Spring

а также habr, javarush, metanit, stackoverflow, youtube и др ;)

Повторение и вопросы к слушателям

- Теперь когда мы уладили все организационные вопросы, вспомним основные моменты языка Java
- Что вы знаете о языке Java?
- Типы данных?
- Основные языковые конструкции?
- Понятие класса и объекта?
- Понятие полей (свойств) и методов?
- Преобразование данных?
- Основные принципы ООП?
- Переопределение и перегрузка?

Обо всем быстро и кратко
boolean и final

- В Java существуют довольно строгие ограничения по отношению к типу boolean: значения типа boolean нельзя преобразовать ни в какой другой тип данных, и наоборот
- Ключевое слово final означает, что присвоить данной переменной какое-нибудь значение можно лишь один раз, после чего изменить его уже нельзя

Символьные строки

- String
- Принцип постоянства (или неизменяемый объект)
- equals() и ==
- "" и null

for each

- for (переменная : коллекция) оператор
- Iterable

Статические поля

- Может быть кто-нибудь знает?

Статические поля

- Может быть кто-нибудь знает?
- Поле с модификатором доступа static существует в одном экземпляре для всего класса.

Статические методы

- Может быть кто-нибудь знает?

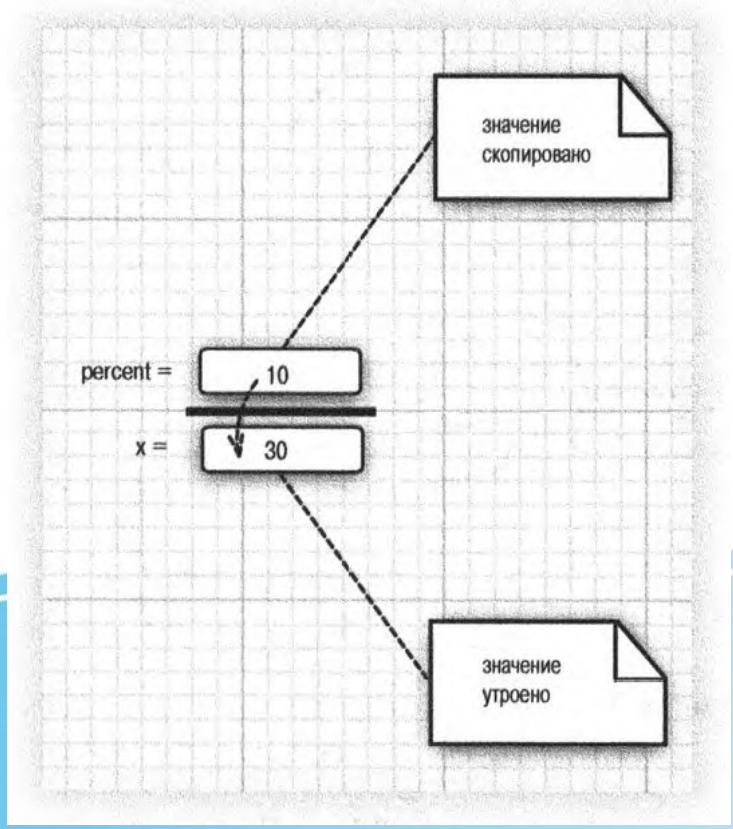
Статические методы

- Может быть кто-нибудь знает?
- Статическими называют методы, которые не оперируют объектами. Они оперируют классами.

Параметры методов

- В языке Java передача параметров в методы осуществляется только **по значению**.
- public static void tripleValue(double x);

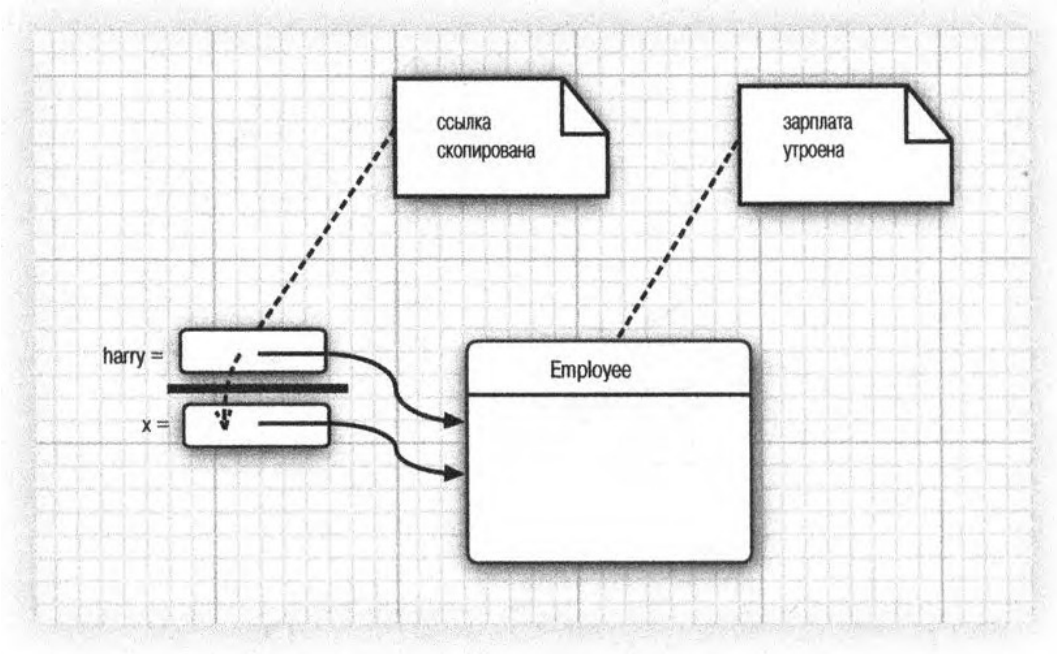
```
{
    x = 3 * x;
}
...
double percent = 10;
tripleValue(percent) ;
```



Параметры методов

- public static void tripleSalary(Employee x) // сработает!

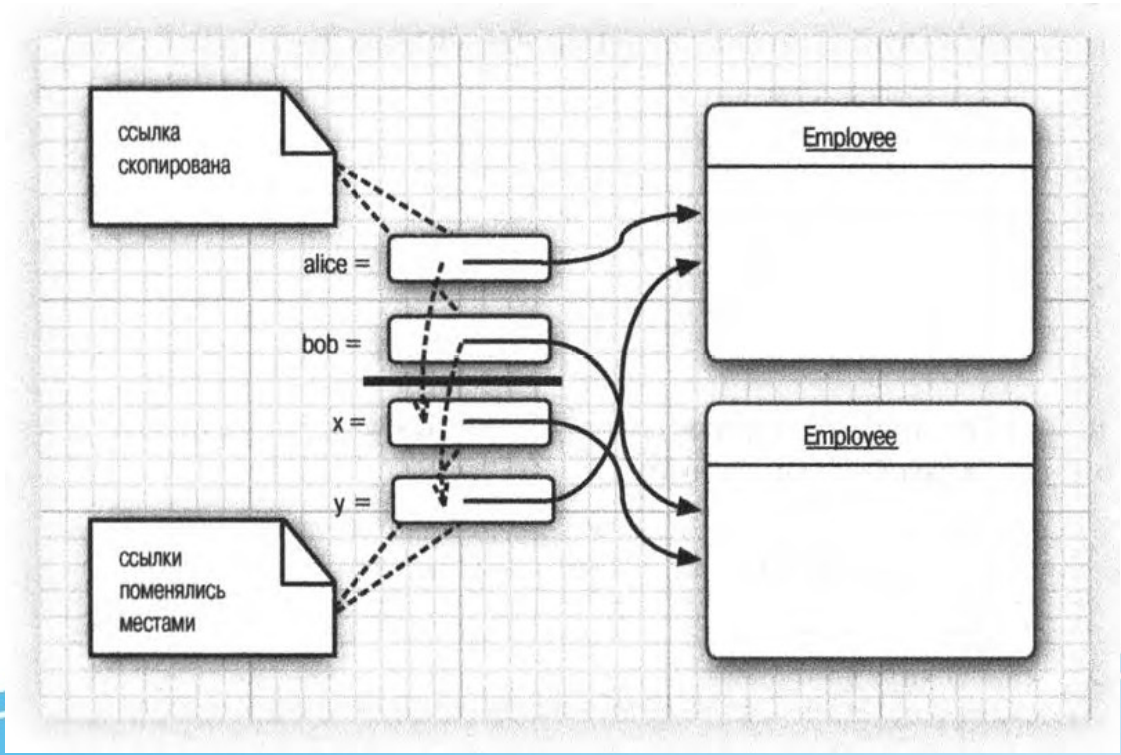
```
{
    x.raiseSalary(200);
}
…
harry = new Employee(...);
tripleSalary(harry);
```



Передача по значению

- public static void swap(Employee x, Employee y)

```
{
    Employee temp = x;
    x = y;
    y = temp;
}
...
Employee a = new Employee("Alice", . . .);
Employee b = new Employee("Bob", . . .);
swap(a, b);
```



finalize()

- В нем выполняются действия, которые необходимы при уничтожении объекта.
- Вызывается автоматически.

Пакеты

- Похожи на пространства имен
- Служат для однозначности определения имен классов
- import
- package
- Область действия пакетов

Наследование

- Что такое наследование?

Наследование

- Что такое наследование?
- extends
- Суперкласс, подкласс
- Родительский, порожденный (потомок, дочерний)
- super

Вызов методов

- 1) Компилятор перечисляет все доступные ему методы с определенным именем (в суперклассе и подклассах)
- 2) Компилятор определяет типы параметром и если имеется только один метод (подходящий по типам параметром), то осуществляется его вызов. Это называется **разрешением перегрузки**.
- 3) В конечном итоге компилятору становится известно имя метода и типы параметром.
- 4) Если метод является закрытым, статическим, конечным или конструктором, то компилятору точно известно как его вызвать. Это называется **статическим связыванием**. В ином случае, вызывающий метод определяется по фактическому типу неявного параметра, а во время выполнения программы происходит динамическое связывание.
- 5) Если при выполнении используется динамическое связывание, то виртуальная машина делает вызов метода, соответствующего фактическому типу объекта, в противном случае, поиск вызываемого метода осуществляется в суперклассе.
- 6) На поиск вызываемого метода уходит слишком много времени, поэтому виртуальная машина заранее создает таблицу методов, в которой перечисляются сигнатуры и фактически вызываемые методы.

О наследовании

- final
- instanceof

Абстрактные классы

- public abstract class Person

  {

  public abstract String getDescription();

  }

equals

- 1) otherObject
- 2) if (this == otherObject) return true;
- 3) if (otherObject == null) return false;
- 4) if (getClass() != otherObject.getClass()) return false;

  if (! (otherObject instanceof ИмяКласса)) return false;
- 5) ИмяКласса other = (ИмяКласса) otherObject;
- 6) Сравните все поля

hashCode

- Хеш-код — это целое число, генерируемое на основе конкретного объекта, идентифицирующее его.
- Если переопределяется метод equals(), то следует переопределить и метод hashCode()
- Если x.equals(y) возвращается логическое значение true, то и результаты вызовов x.hashCode() и y.hashCode() также должны совпадать

Объектные оболочки

- Integer, Long, Float, Double, Short, Byte, Character и Boolean
- ArrayList<Integer> list = new ArrayList<Integer>();
- list.add(3);

Интерфейсы

- Не может быть ни полей, ни экземпляров.
- Однако может иметь константные поля и можно создавать интерфейсные переменные.
- Может иметь методы по умолчанию.

Исключения

- try catch throw
- Проверяемые и непроверяемые исключения
- Перехват нескольких исключений
- finally
- try с ресурсами

Спасибо за внимание!