Laboratory Work #28 (part 2)

Java Serialization

LEARN. GROW. SUCCEED.

 Department: <Software of Information Systems and Technologies>
Faculty of Information Technology and Robotics
Belarusian National Technical University
by Viktor Ivanchenko / ivanvikvik@bntu.by / Minsk

ЛАБОРАТОРНАЯ РАБОТА #28 (part 2)
Сериализация объектов в Java

Цель работы 

Изучить концепцию и аспекты сериализации и десериализации в Java и закрепить знания и навыки на примере разработки интерактивных приложений.

Требования 

1) Скорректировать UML-диаграмму классов и интерфейсов, составляющих архитектуру основного приложения.
2) При проектировании и реализации программы рекомендуется использовать архитектурный шаблон MVC, а также фундаментальные SOLID и GRASP принципы.
3) Классы и другие сущности программы должны быть грамотно структурированы по соответствующим пакетам и иметь отражающую их функциональность названия.
4) Программа должна быть снабжена дружелюбным и интуитивно понятным интерфейсом. Работа с консолью должна быть минимальной.
5) Приложение должно корректно обрабатывать любые исключительные ситуации, которые могут возникнуть в процессе работы программы.
6) Для подтверждения работоспособности и адекватности программы, вся модель проекта должна быть покрыта соответствующими модульными тестами. Для модульного (*unit*) тестирования рекомендуется использовать тестовый фреймворк *jUnit* или *TestNG*.
7) Необходимо по максимуму пытаться разрабатывать универсальный код.
8) Программа должна обязательно быть снабжена комментариями на английском языке, в которых необходимо указать краткое предназначение программы, номер лабораторной работы и её название, версию программы, ФИО разработчика, номер группы и дату разработки.

9) Исходный текст классов и демонстрационной программы рекомендуется также снабжать комментариями.
10) При разработке программ придерживайтесь соглашений по написанию кода на JAVA (Java Code-Convention).

Основное задание

В программе, разработанной в предыдущей лабораторной работе, необходимо:

- добавить дополнительно два способа инициализации объектов с использованием стандартной и пользовательской («кастомной») сериализации;
- попробовать сделать рефакторинг слоя View таким образом, чтобы в контроллере объект-принтер возвращался с помощью объекта-креатора, метод которого брал бы уже готовые объекты из контейнера типа *Map*. Ключом в данном контейнере может быть тип принтера, а значением – конкретный объект-принтер.

Best of LUCK with it, and remember to HAVE FUN while you're learning :)

Victor Ivanchenko

Контрольные вопросы 

1) Что такое сериализация/десериализации? В чём основная концепция?
2) Для чего нужна сериализация/десериализации? Её преимущества и недостатки?
3) В каких областях в Java используется сериализация?
4) В чём особенность технологии сериализации в Java?
5) Какие существуют способы сериализации объектов в Java?
6) Какие интерфейсы и потоки ввода-вывода применяются для реализации сериализации/десериализации в Java? Какой функционал они предоставляют?
7) Что такое автоматическая (стандартная или нативная) сериализация объектов в Java?
8) Как должен описываться класс и какое он должен реализовывать поведение, чтобы объекты данного класса можно было сериализовать и десериализовать с помощью автоматической сериализации?
9) Какая исключительная ситуация возникает, когда не возможно сериализовать объект? Что необходимо предпринять, чтобы такого не происходило?
10) Что такое маркерный интерфейс в Java?
11) Опишите поведение, которое декларирует интерфейс ***java.io.Serializable***.
12) Как запретить сериализацию некоторых полей объекта при автоматической сериализации?
13) Что вообще может быть сериализовано при автоматической сериализации (какие именно поля, описанные в классе, могут быть сериализованы)?
14) Какие исключительные ситуации могут возникнуть, если не возможно десериализовать объект? Когда такие вещи могут произойти?
15) За что отвечает статическое поле ***serialVersionUID*** и каким образом его можно добавить в класс, объекты которого должны сериализоваться?
16) Как происходит сериализация и десериализация объекта, класс которого не реализует интерфейс *java.io.Serializable,* а его базовый класса реализует данный интерфейс?
17) Как происходит сериализация и десериализация объекта, класс которого реализует интерфейс ***java.io.Serializable***, а его базовый класса не реализует данный интерфейс?
18) В чём особенность сериализации объекта, который состоит в композиции (или агрегации) с объектами других классов?

19) Если необходимо обойти ограничения стандартной сериализации, что для этого дополнительно нужно знать и что добавить в класс, объекты которого необходимо сериализовать?
20) Какие есть плюсы и минусы у стандартной (автоматической) сериализации?
21) Что такое «кастомная» (custom-ая) сериализация? Какие у неё преимущества и недостатки перед стандартной сериализацией?
22) Как реализовать custom-ую сериализацию в Java?
23) За что отвечает интерфейс ***java.io.Externalizable***? Что за поведение в нём декларируется?
24) Чем отличается custom-ая сериализация от стандартной?
25) Что и как необходимо описать в класса, чтобы его объекты могли быть сериализованы с помощью custom-ой сериализацией?