

МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П.КОРОЛЕВА
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

А. В. Гаврилов

Технологии сетевого программирования

Задания на лабораторные работы

Самара

2011

Автор: ГАВРИЛОВ Андрей Вадимович

Набор заданий на лабораторные работы по курсу «Технологии сетевого программирования» предназначены для бакалавров четвертого курса факультета информатики направлений 010400.62 «Прикладная математика и информатика» и 010900.62 «Прикладные математика и физика».

Лабораторная работа №1

Работа с сокетами

Темы для предварительного изучения

- Примитивные и ссылочные типы данных.
- Операторы и управление ходом выполнения программы.
- Классы-обертки.
- Массивы и строки.
- Интерфейсы и наследование.
- Исключения, их создание и обработка.
- Потоки инструкций, взаимодействие потоков инструкций.
- Потоки данных и их разновидности.
- Лексический анализ строк и потоков данных.
- Сериализация.
- Работа с сокетами на Java.

Задание на лабораторную работу

В процессе разработки учебного приложения ознакомиться с особенностями создания клиент-серверных приложений на основе сокетов.

Введение

Часто необходимость создания клиент-серверного приложения возникает в случае, если на клиентской машине решение некоторой задачи (например, вычислительной), не представляется возможным. В этом случае клиент лишь формирует задание для сервера, а потом получает результаты. В данной работе предлагается создать клиент-серверное приложение, реализующее решение на серверной стороне некоторой вычислительной задачи.

Таблица 1. Требования к реализации

№ варианта	Вид сервера	Передача параметров	Действие над матрицами
1	Последовательный	Последовательность элементов	Сложение
2	Последовательный	Последовательность элементов	Умножение
3	Последовательный	Сериализованные массивы	Сложение
4	Последовательный	Сериализованные массивы	Умножение
5	Последовательный	Сериализованные объекты	Сложение
6	Последовательный	Сериализованные объекты	Умножение
7	Параллельный	Последовательность элементов	Сложение
8	Параллельный	Последовательность элементов	Умножение
9	Параллельный	Сериализованные массивы	Сложение
10	Параллельный	Сериализованные массивы	Умножение
11	Параллельный	Сериализованные объекты	Сложение
12	Параллельный	Сериализованные объекты	Умножение

Задание 1

Написать класс матриц, поддерживающий операцию сложения или перемножения (в зависимости от вашего варианта, см. таблицу 1).

Должен быть описан один класс, который должен:

- инкапсулировать в себе массив-матрицу;
- иметь конструктор, параметрами которого являются размеры матрицы;
- реализовывать получение размеров матрицы (два метода, возвращающие количество строк и количество столбцов соответственно);

- реализовывать получение и изменение отдельных элементов матрицы (два метода, позволяющие получить и изменить значение элемента матрицы по его координатам);

- иметь статический метод сложения или перемножения матриц, возвращающий результат действия;

- иметь статический метод, позволяющий записать матрицу в текстовый файл;

- иметь статический метод, позволяющий считать матрицу из текстового файла.

Задание 2

Разработать клиент-серверную программу: клиент считывает исходные данные из файлов и передает их для решения серверу, после чего получает ответ от сервера и записывает его в файл.

При запуске клиентской части в командной строке передаются следующие параметры: имя файла с первой матрицей, имя файла со второй матрицей, имя файла для результирующей матрицы.

Передача параметров серверу может производиться различными способами: передача непосредственно исходных данных (при этом сервер сам создает объекты для решения задачи) в виде последовательности чисел или в виде сериализованного массива, либо передача уже сформированных объектов задачи (методами сериализации), для которых на серверной стороне лишь вызывается метод решения. В обоих случаях предполагается, что на серверной стороне наличествует байт-код необходимых классов.

Сервер также можно реализовать различными способами: с последовательной обработкой запросов (в один момент времени сервер выполняет только один запрос), либо с параллельной (каждый запрос выполняется в отдельном потоке инструкций).

В выходной файл клиентской программы должен быть записан результат выполнения действия. Если выполнение действия невозможно, в выходной

файл должно быть записано текстовое сообщение о невозможности выполнения действия.

Требования к конкретной реализации в зависимости от варианта задания приведены в таблице 1.

Лабораторная работа №2

RMI. Обработка изображений

Темы для предварительного изучения

Примитивные и ссылочные типы данных.

Операторы и управление ходом выполнения программы.

Классы-обертки.

Массивы и строки.

Интерфейсы и наследование.

Исключения, их создание и обработка.

Потоки команд, взаимодействие потоков команд.

Потоки данных и их разновидности.

Сериализация.

Работа с сокетами на Java.

RMI

Работа с изображениями, пакет `java.awt.image`

Задание на лабораторную работу

В процессе разработки учебного приложения ознакомиться с особенностями создания клиент-серверных приложений на основе технологии RMI, изучить простейшие принципы работы с изображениями на Java.

Введение

Выполнение лабораторной работы предусматривает реализацию клиент-серверной программы, основанной на технологии RMI, реализующую обработку растрового изображения КИХ-фильтром. Материал, посвященный алгоритмам, реализующим локальную обработку скользящим окном, вы можете найти в следующей литературе:

1. Методы компьютерной обработки изображений / Под ред. В.А. Сойфера. – М.: Физматлит, 2001. – 784 с. – ISBN 5-9221-0180-3.
2. Теоретические основы цифровой обработки изображений: Учебное пособие / В.А. Сойфер, В.В. Сергеев, С.Б. Попов, В.В. Мясников. Самарский государственный аэрокосмический университет имени академика С.П. Королева. Самара, 2000, 256 с. (доступна в электронном виде по адресу <http://www.wslib.nm.ru/disp.djvu>)

Примеры работы с изображениями в Java, а также пример фильтра вы можете найти в соответствующей лекции из курса «Системное и прикладное программное обеспечение».

Задание

Разработать клиент-серверную программу, реализующую обработку изображений.

Клиентская программа получает в качестве параметров имена входного и выходного файлов изображений. Для простоты рекомендуется для входных файлов использовать формат JPG, а для выходных PNG. Далее клиентская программа вызывает удаленный метод обработки изображения указанным в варианте задания фильтром, пересылая в качестве параметра файл с исходным изображением. После чего выходное изображение возвращается на клиентский компьютер как результат работы метода.

Логика работы серверного приложения очевидно следует из описания логики клиентского приложения.

Список алгоритмов обработки изображения в зависимости от номера варианта приведен в таблице 1. В качестве комментариев приведены рекомендации по поиску информации в книге «Теоретические основы цифровой обработки изображений».

Таблица 1. Алгоритмы для различных вариантов.

№ варианта	Выполняемое действие
1	Повышение резкости окном 3x3 с усреднением по 5 отсчетам <i>См. стр. 181-183</i>
2	Повышение резкости окном 3x3 с усреднением по 9 отсчетам <i>См. стр. 181-183</i>
3	Сглаживание окном 3x3 с высокой степенью сглаживания высокочастотных шумов <i>Математический аппарат аналогичен случаю повышения четкости, пример маски приведен ниже:</i> $\begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix}$
4	Сглаживание окном 3x3 со средней степенью сглаживания высокочастотных шумов <i>Математический аппарат аналогичен случаю повышения четкости, пример маски приведен ниже:</i> $\begin{pmatrix} 1/10 & 1/10 & 1/10 \\ 1/10 & 2/10 & 1/10 \\ 1/10 & 1/10 & 1/10 \end{pmatrix}$
5	Сглаживание окном 3x3 с низкой степенью сглаживания высокочастотных шумов <i>Математический аппарат аналогичен случаю повышения четкости, пример маски приведен ниже:</i> $\begin{pmatrix} 1/16 & 1/8 & 1/16 \\ 1/8 & 1/4 & 1/8 \\ 1/16 & 1/8 & 1/16 \end{pmatrix}$
6	Выделение контуров оператором Лапласа, размер окна 3x3 <i>См. стр. 195-196</i>
7	Медианная фильтрация с окном из 5 элементов, расположенных «крестом» <i>См. стр. 224-226</i>
8	Медианная фильтрация с окном из 9 элементов, расположенных «крестом» <i>См. стр. 224-226</i>
9	Медианная фильтрация с окном из 9 элементов, расположенных «квадратом» <i>См. стр. 224-226</i>
10	Медианная фильтрация с окном из 81 элемента, расположенного «квадратом» <i>См. стр. 224-226</i>

Лабораторная работа №3

Servlets. Java Server Pages

Темы для предварительного изучения

Протокол HTTP

HTML

CSS

XML

JAXB

Servlets

JSP

Задание на лабораторную работу

В процессе разработки учебного приложения ознакомиться с созданием Web-приложений с применением Java-технологий.

Задание 1

Разработать html-страницы для ввода данных и для вывода результата.

Первая страница должна называться `index.html`. На ней должны находиться:

- заголовок,
- сопроводительный текст,
- два поля для ввода данных для выполнения операции,
- приглашения (подсказки) к этим полям ввода,
- кнопка для завершения ввода (с переходом на `result.html`).

Вторая страница должна называться `result.html`. На ней должны находиться:

- заголовок,
- значения аргументов,

- результат выполнения операции,
- ссылка на исходную страницу,
- ссылка на скачивание файла `result.xml`.

На этапе разработки дизайна в качестве значений аргументов и результата выполнения операции могут использоваться произвольные данные.

Обе страницы должны быть оформлены с применением CSS. Таблица стилей должна быть размещена в отдельном файле и быть общей для обеих страниц. Все элементы обеих страниц должны изменить свой внешний вид в соответствии со стилями (дизайн – на ваше усмотрение). Особое замечание: ссылки должны менять цвет при наведении на них курсора.

Задание 2

На основе страницы `index.html` следует разработать страницу `index.jsp`, добавляющую следующую функциональность.

1. Форма должна переходить на эту же страницу.
2. Если запрос содержит введенные данные, то эти данные должны быть проверены на корректность в соответствии с бизнес-логикой действия, определяемого вариантом (см. таблицу 1).
3. Если данные некорректны, то они должны быть снова выведены в форму, а некорректный элемент выделен и снабжён комментарием (выделение также средствами CSS).
4. Если данные корректны, они должны быть помещены в объект класса (его также нужно разработать), описывающий задание на выполнение. Объект должен быть помещён в сессию как атрибут, после чего обработка запроса должна быть передана на адрес `result.html`.

Задание 3

К адресу `result.html` должен быть привязан сервлет, который выполняет следующие действия.

1. Извлекает из сессии объект задания.

2. Выполняет задание, помещая результат выполнения в тот же объект.
3. Передаёт обработку запроса на страницу `result.jsp`.

Задание 4

На основе страницы `result.html` следует разработать страницу `result.jsp`, добавляющую следующую функциональность.

1. Задание и результат его выполнения извлекаются из сессии, значения используются для вывода на страницу.
2. Если объект задания в сессии отсутствует, следует вывести сообщение об ошибке и перенаправить запрос на `index.jsp` с задержкой в 5 секунд и прямой ссылкой для немедленного перехода.

Задание 5

К адресу `result.xml` должен быть привязан сервлет, выполняющий следующие действия.

1. Тип отклика изменяется на `xml`.
2. Задание и результат извлекаются из сессии.
3. На их основе формируется отклик следующего вида.

```
--- result.xml -----  
  
    <task>  
        <argument1>Argument value</argument1>  
        <argument2>Argument value</argument2>  
        <result>Result value</result>  
    </task>  
-----
```

Для формирования `xml`-отклика должна использоваться технология JAXB. Класс, используемый для передачи значений в рамках сессии, должен быть соответствующим образом подготовлен.

Таблица 1. Операции для различных вариантов.

№ варианта	Выполняемое действие	Некорректные данные
1	Конкатенация строк	Пустые строки
2	Поиск вхождения второй строки в первую (результат – позиция вхождения)	Пустые строки
3	Размножение строки (первый аргумент – строка, второй – число, результат – строка, повторенная заданное число раз)	Пустая строка, введено не число, отрицательное число.
4	Удаление конечных символов (первый аргумент – строка, второй – количество символов, которые надо удалить, результат – уменьшенная строка)	Пустая строка, введено не число, отрицательное число, количество символом больше длины строки.
5	Удаление начальных символов (первый аргумент – строка, второй – количество символов, которые надо удалить, результат – уменьшенная строка)	Пустая строка, введено не число, отрицательное число, количество символом больше длины строки.
6	Удаление подстрок (первый аргумент – строка, второй – подстрока, результат – исходная строка, в которой удалены все элементы, совпадающие с подстрокой)	Пустые строки
7	Сложение двух чисел	Введено не число
8	Разность двух чисел	Введено не число
9	Умножение двух чисел	Введено не число
10	Частное двух чисел	Введено не число
11	Возведение в степень	Введено не число, первое число отрицательное.
12	Логарифм первого числа по основанию второго	Введено не число, первое число отрицательное.

Лабораторная работа №4

Enterprise Java Beans

Темы для предварительного изучения

Протокол HTTP

HTML

CSS

JSF

EJB

Задание на лабораторную работу

В процессе разработки учебного приложения ознакомиться с созданием Web-приложений с применением Enterprise Java Beans.

Задание

Разработать web-приложение, сходное по функциональности с приложением из лабораторной работы 3, но основанное на технологиях JSF и EJB.

1. Обе web-страницы должны быть jsf-страницами.
2. Объект задания должен быть управляемым бином.
3. Контроль ввода должен осуществляться на уровне валидаторов.
4. Действие по выполнению операции должно осуществляться сессионным компонентом, обращение к которому происходит из управляемого бина.
5. Экспорт в xml и соответствующий сервлет не требуются.