Разработка клиент-серверных приложений. Структура дисциплины (2023)

Рейтинг

	Рейтинг		
Лаб. 1	20	Разработка классов движущихся объектов	
Лаб. 2	10	Сетевое соединение на сокетах (TCP)	
Лаб. 3	10	Передача данных без установления соединения (UDP)	
Лаб. 4	20	Разработка клиент-серверного приложения на RestAPI в фреймворке Spark-Retrofit	
РГР	20		
Зачет	20		

Лабораторные работы

В качестве примера для лаб.1-3 - 24_MovingObjects_TCP_UDP_2023. Код разработки и решения должны быть ОРИГИНАЛЬНЫМИ.

Лабораторная работа 1. Разработка классов движущихся объектов

Разработать абстрактный базовый класс графического объекта с данными и функционалом (абстрактными и переопределяемыми методами): координаты центра и размеры охватывающего прямоугольника, цвет, рисование объекта, проверка на принадлежность точки объекту, запись/чтение в различные потоки, а также все остальные, необходимые для реализации функционала в последующих работах.

Разработать два производных класса (по вариантам). Разработать оконное приложение с созданием и уничтожением объектов по координатам клика мыши. Объекты сохраняются в векторе в виде ссылок на базовый класс.

Варианты заданий

1. Многоугольник с произвольным количеством вершин, соединенных через одну вершину (типа звезда)
2. Выпуклый многоугольник с произвольным количеством вершин, соединенных последовательно
3. Строка текста
4. Смайлик
5. Картинка, загружаемая из файла
6. Вертикальная (горизонтальная) синусоида с изменяющейся начальной фазой (один период)
7. Объект – группа объектов (группировка)

Реализовать процесс движения объектов (вид движения по вариантам для каждого типа), остановки и возобновления движения по отдельности и всех вместе. Внести в базовый класс функционал движения, в производный – алгоритм движения для конкретного типа.

Варианты заданий

1. Равномерное движение со случайным направлением и скоростью, отскакивание от краев экрана
2. Ускоренное движение со случайным направлением и скоростью, отскакивание от краев экрана с кратным уменьшением скорости (неупругое столкновение)
3. Вращение вокруг своей оси
4. Вращение по окружности со случайным радиусом
5. Движение по сторонам квадрата
6. Броуновское движение со случайным изменением значения скорости и направления
7. Дрожжание – случайные изменения значения скорости и направления с возвратом в среднюю точку (отслеживание среднего значения координат)
8. Триангулярная пульсация размера фигуры
9. Синусоидальная пульсация размера фигуры
10. Триангулярная пульсация по каждой прямоугольной координате в противофазе
11. Синусоидальная пульсация по каждой прямоугольной координате в противофазе
12. Триангулярное колебание пульсация по каждой прямоугольной координате в противофазе

В базовом классе реализовать (объявить) методы (абстрактные) чтения/записи объекта в двоичный и текстовый потоки. Переопределить во всех производных классах. В оконном классе реализовать методы сохранения и загрузки вектора объектов в тестовый, двоичный и сериализуемый файл (двоичная сериализация средствами Java и XML-сериализация с использованием библиотеки xstream). Формат двоичного и текстового файла содержит начальный счетчик объектов, а затем для каждого их них – имя и содержимое. При загрузке из файла создавать объект файла произвольного класса с помощью средств рефлексии (методы Class.forName и Class.newInstance)

Лабораторная работа 2. Сетевое соединение на сокетах

Разработать приложение с клиентской и серверной частью. Серверная часть реализуется в потоке и запускается/останавливается при помощи флажка. Серверная компонента ждет запроса на соединение на фиксированном порте. Клиентская компонента в другой программе запрашивает соединение с серверной. Протокол является симметричным (двунаправленным), т.е. каждое приложение может посылать один тот же набор команд:

- закрытие соединения;
- очистка вектора объектов;
- передача объекта;
- запрос на передачу объекта;
- передача списка имен объектов (в качестве имени можно использовать «подпись» объекта, например в виде – «имя класса,цвет»);
- запрос размера вектора объектов;
- запрос объекта с заданным номером из вектора;
- ответ на запрос размера вектора.

Принимаемые объекты отображаются в клиенте и запоминаются в векторе. В протоколе для передачи объектов через соединения использовать методы сериализации из лаб.1.

Лабораторная работа 3. Передача данных без установления соединения

Задание в л.р.2 реализовать с использованием класса дейтаграмм протокола UDP без установления соединения (класс DatagramSocket). Флажком выделяется «первое/второе» приложение, отличающиеся только номером порта, на котором они прослушивают прием. Например, первое принимает данные на порте 7001, а передает на 7002, второе – наоборот. Для конвертации XML-строки, содержащей передаваемый объект, в массив байтов для передачи в дейтаграммный сокет использовать классы ByteArrayOutputStream/ByteArrayOutputStream в качестве класса-источника текстового потока.

Лабораторная работа 4. Разработка клиент-серверного приложения на RestAPI в фреймворке Spark-Retrofit

Для выполнения работы использовать каркас SparkRetrofitExample.

Разработать серверную компоненту, которая хранит вектор движущихся объектов. При завершении работы сохраняет текущий вектор в файл, при перезапуске – восстанавливает. Принимает от клиента команды в формате RestAPI:

- получить список «имен» объектов ;
- получить объект по его индексу в векторе;
- удалить из вектора по индексу;
- получить все объекты;
- добавить объект в вектор;

Разработать клиентское приложение на основе л.р.1-3, использующее серверное приложение для хранения графических объектов.

Расчетно-графическая работа

Необходимо изучить, установить и опробовать один из компонент разработки, используемой при разработке протоколов, распределенных приложений или протокольных клиентов: библиотеку, фреймворк. Содержание работ и отчета (пояснительной записки).

1. Краткое описание функционала, интерфейсов
2. Аннотированная подборка ссылок по теме:
   - официальный сайт;
   - ссылки на репозитории исходного кода, maven-репозитории;
   - посты в профессиональных сообществах (habrahabr) с учебными материалами и примерами развертывания (использования);
   - посты в профессиональных сообществах с обсуждением практики использования, проблем, преимуществ и недостатков.
3. Сравнительный анализ преимуществ и недостатков с другими функционально подобными библиотеками и феймворками. Определить квалификационные признаки – масштаб, среда (язык) использования, свободное/коммерческое ПО,

распространенность, простота применения, по которым провести сравнение с аналогами.

4. Создать собственный проект – пример использования, оформить документы: описание ключевых идей, диаграмма классов приложения, описание классов (назначение класса, свойства, методы), описание настройки и работы (скриншоты, тесты). Приложение – экранная форма («кабина самолета») с набором элементов для исполнения функционала в любой последовательности.

Совместно отчетом необходимо сдать исходные коды проектов с примерами, приведенными в отчете.

Варианты можно выбрать из документа «Список полезных ссылок для Java программиста» - https://yadi.sk/i/AyFzLKgjGvt-FA Разделы I-2, III-2,3,4,7,10

Зачет

Зачет теоретический в форме ответа на тесты. В билете выдается 10 тестовых вопросов **по темам зачета**, на каждый дается содержательный письменный ответ – 1-2 абзаца. Вес ответа 0..2 балла.

Темы зачета

1. Понятие протокола. Уровни протоколов в открытых системах, их функциональность.
2. Транспортный протокол TCP. Форматы сообщений, структуры данных, функциональность и алгоритмы передача данных и управления потоком.
3. Система идентификации соединений в TCP. Статические и динамические порты, сокеты и серверные сокеты. Процедуры установления и разрыва соединения. Состояния соединения.
4. Основные технологические решения в протоколах низких уровней и их использование в прикладном программировании сетевых приложений.
5. Системы клиент-сервер. Клиент и сервер как программные компоненты и роли. Синхронное и асинхронное взаимодействие.
6. Понятие тонкого и толстого клиента. Слои: представление, контроллер, бизнес-слой, DAO. Уровни соединения клиента и сервера в типовом приложении. Прикладной протокол как элемент взаимодействия слоев клиент-серверной архитектуры. Проектирование протоколов распределенных систем.
7. Проектирование протокола. Стратегия взаимодействия клиента с сервером. Архитектурные аспекты проектирования протокола. Использование конечных автоматов (диаграмм состояний) в протоколах
8. Распределенные системы. Службы и симметричные протоколы распределенных систем
9. Аспекты проектирования протокола. CheckList понимания протокола.
10. Классы Socket и ServerSocket, установление и разрыв соединения
11. Протокол HTTP. Класс URLConnection. Реализация webAPI.
12. Протокол UDP. Класс DatagramSocket
13. Серверные web-приложения на Java. Сервлеты.
14. RestAPI. Основные положения. Клиентская библиотека Retrofit.

15. Контекст соединения. Восстановление соединений. Паттерн «сессия». Взаимодействие клиент-сервер без последействия. Идентификатор сессии (токен) Паттерн «сессия», его реализация при восстановлении постоянных соединений и при взаимодействии клиент-сервер без последействия