Общество с ограниченной ответственностью «АйТи Бюро оптимальных решений» (ООО «АйТиБОР»)

ОПИСАНИЕ ТЕХНИЧЕСКОЙ АРХИТЕКТУРЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ «СИСТЕМА МОНИТОРИНГА ДИСКОВОГО ПРОСТРАНСТВА И БЭКАПОВ «РЕВИЗОР»

Листов 15

г. Липецк, 2026 г.

Содержание

Аннотация

Документ представляет собой техническое описание программного комплекса **«РЕВИЗОР»**, предназначенного для централизованного мониторинга свободного места на дисках и состояния резервных копий.

В описании рассмотрены архитектура, модули, графический интерфейс (Swing), применяемые технологии (Gson, JavaMail, многопоточность) и требования к программному обеспечению. Документ предназначен для администраторов и разработчиков, выполняющих развертывание и сопровождение системы.

Перечень сокращений

Сокращение	Расшифровка
API	Application Programming Interface, программный интерфейс приложения
БД	База данных
HTTP	HyperText Transfer Protocol, протокол передачи гипертекста
HTTPS	HTTP Secure, защищённый протокол передачи гипертекста
SSL/TLS	Secure Sockets Layer / Transport Layer Security, протоколы шифрования
XML	eXtensible Markup Language, расширяемый язык разметки
JSON	JavaScript Object Notation, формат обмена данными
GUI	Graphical User Interface, графический интерфейс пользователя
CLI	Command Line Interface, интерфейс командной строки
SMTP	Simple Mail Transfer Protocol, протокол отправки почты
JKS	Java KeyStore, хранилище ключей и сертификатов Java

Термины и определения

Термин	Определение
Сервер мониторинга	Программный компонент, принимающий отчёты от клиентов, анализирующий данные и отправляющий уведомления
Клиент мониторинга	Программный компонент, собирающий данные о дисках и бэкапах и отправляющий их на сервер
Порог предупреждения	Процент свободного места на диске, при достижении которого генерируется предупреждение
Индивидуальный порог	Порог, заданный для конкретного ПК, отличный от общего
Бэкап	Резервная копия данных, хранящаяся в заданной папке
Keystore	Файл-хранилище криптографических ключей и сертификатов для HTTPS
Truststore	Файл-хранилище доверенных сертификатов
Планировщик	Компонент, выполняющий задачи по расписанию

1. Общие сведения

1.1. Наименование программы

Наименование: **РЕВИЗОР**

1.2. Функциональное назначение

Система предназначена для:

- сбора данных о свободном месте на дисках указанных ПК;
- мониторинга состояния бэкапов (возраст, размер);
- отправки отчётов на центральный сервер;
- анализа данных и генерации предупреждений при превышении порогов;
- отправки email-уведомлений администраторам;
- ведения базы данных ПК с индивидуальными настройками порогов и проверки бэкапов.

1.3. Программное обеспечение, необходимое для функционирования

Для серверной части:

- Java Runtime Environment (JRE) 8 или выше;
- Доступ к SMTP-серверу для отправки почты.

Для клиентской части:

- Java Runtime Environment (JRE) 8 или выше;
- Сетевой доступ к серверу мониторинга.

1.4. Используемые языки программирования

- Java (основной язык реализации);
- XML (для генерации и передачи отчётов);
- JSON (для хранения конфигурации).

1.5. Лингвистическое обеспечение

Интерфейс пользователя реализован на русском языке. Логи и системные сообщения могут содержать текст на английском языке.

2. Структура программного обеспечения

2.1 Взаимодействие компонентов

ПО «РЕВИЗОР» состоит из следующих компонентов:

1. *Клиентская часть* - сбор и отправка данных о состоянии дискового пространства и бэкапов на сервер;
2. *Серверная часть* - прием, хранение, анализ данных от клиентов и оповещение администратора о проблемах.

Схема взаимодействия компонентов представлена на рисунке 1.1

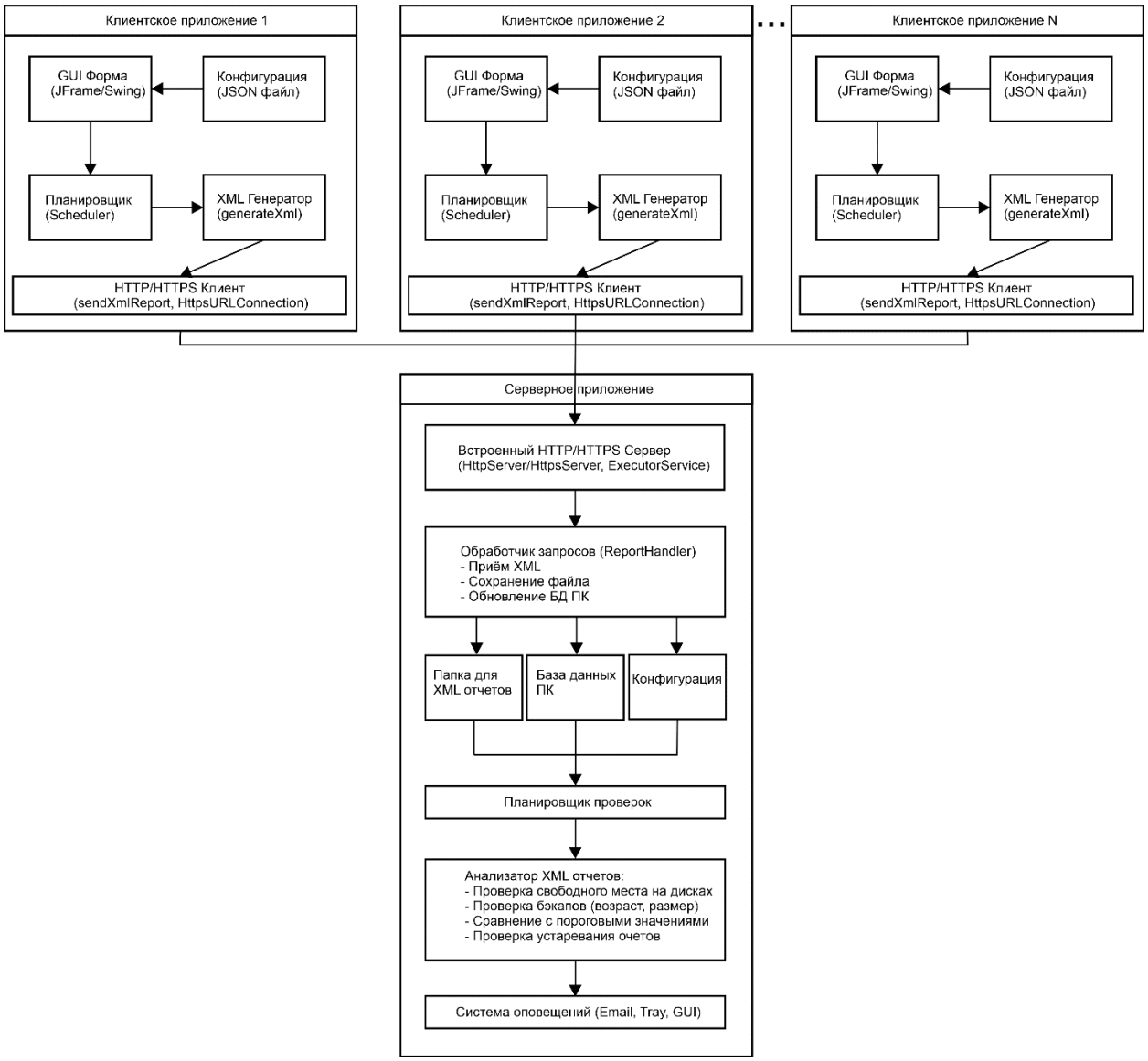

Рисунок 1.1 – Схема взаимодействия компонентов

3. Техническое описание программного обеспечения

3.1 Общее назначение

Программа состоит из двух частей: серверной и клиентской.
Сервер собирает и анализирует XML-отчёты от клиентов, отслеживает свободное место на дисках и актуальность резервных копий, при необходимости отправляет предупреждения по электронной почте. Клиент периодически формирует отчёт о состоянии выбранных дисков и папок с бэкапами и отправляет его на сервер по протоколу HTTP/HTTPS.

3.2 Серверная часть (DiskReportProcessorGUI)

3.2.1 Основной класс DiskReportProcessorGUI

Класс наследует JFrame и реализует графический интерфейс для настройки и мониторинга. Основные функции:

- отображение и изменение параметров (папка с отчётами, пороги, интервалы, настройки сервера и почты);
- управление списком компьютеров и их индивидуальными настройками;
- запуск HTTP/HTTPS сервера для приёма отчётов;
- планирование периодической обработки отчётов и очистки старых XML-файлов;
- отображение текущего состояния дисков и бэкапов в текстовой области;
- интеграция с системным треем (сворачивание, уведомления).

3.2.1.1 Ключевые поля

- config – объект с настройками (класс Config);
- computersDB – Map<String, ComputerInfo> база данных компьютеров;
- scheduler, cleanupScheduler – планировщики для периодических задач;
- httpServer, httpsServer – серверы для приёма отчётов;
- GUI-компоненты: folderPathField, thresholdSpinner, intervalSpinner, computerList и др.

3.2.1.2 Основные методы

- initComponents() – создание элементов интерфейса;
- startServer(), stopServer(), restartServer() – управление HTTP/HTTPS сервером;
- startScheduler(), restartMainScheduler() – запуск периодической обработки отчётов;
- startCleanupScheduler(), restartCleanupScheduler() – планирование очистки папки с отчётами;
- processReportsInternal() – основной метод обработки всех XML-файлов, анализа дисков и бэкапов, формирования предупреждений;
- sendReport() – отправка отчёта по электронной почте (в отдельном потоке);
- loadConfig(), saveConfig() – загрузка/сохранение JSON-конфигурации в файл diskmonitor.cfg;

- loadComputersDB(), saveComputersDB() – работа с базой данных компьютеров (файл computers.db, формат: имя;время_отчёта;порог;мин_размер_бэкапа;проверять_бэкап);
- addComputerToDB(), removeComputerFromDB(), updateComputerReportTime() – управление записями компьютеров;
- getThresholdForComputer(), getComputerBackupMinSize() – получение индивидуальных или общих значений порогов;
- showAlerts() – отображение всплывающих уведомлений в трее и мигание иконки при проблемах;
- minimizeToTray(), openFromTray() – работа с треем.

3.2.2 Внутренние классы сервера

3.2.2.1 Config

Хранит все настройки сервера (поля со значениями по умолчанию):
- folderPath – путь к папке с отчётами;
- threshold – общий порог свободного места (%);
- checkIntervalMinutes – интервал проверки отчётов (минуты);
- reportTimeoutMinutes – время, после которого отчёт считается устаревшим;
- smtpServer, smtpPort, smtpUser, smtpPass, emailFrom, emailTo – настройки почты;
- httpPort, httpsPort – порты для HTTP/HTTPS;
- useHTTPS – флаг использования HTTPS;
- keystorePath, keystorePassword – параметры для SSL;
- sendOnlyNegative – отправлять только негативные отчёты;
- backupMaxAgeHours – максимальный возраст бэкапа (часы);
- defaultBackupMinSizeMB – минимальный нормальный размер бэкапа (МБ);
- cleanupTime, cleanupFrequency, cleanupDayOfWeek – параметры очистки папки.

3.2.2.2 ComputerInfo

Представляет информацию об одном компьютере:
- name – имя ПК;
- lastReportTime – время последнего отчёта (мс);
- threshold – индивидуальный порог (0 – использовать общий);
- backupMinSizeMB – индивидуальный минимальный размер бэкапа (0 – общий);
- checkBackup – флаг, нужно ли проверять бэкапы для этого ПК.

3.2.2.3 ComputerListRenderer extends DefaultListCellRenderer

Кастомный рендерер для отображения элементов списка компьютеров:
- показывает имя, дату последнего отчёта, индивидуальный порог и размер бэкапа;
- использует цвет: красный для устаревших отчётов, серый – если отчётов никогда не было;
- добавляет эмодзи (🔍 или 🚫) для индикации включённой/отключённой проверки бэкапов;
- всплывающая подсказка подсказывает действие двойного клика.

3.2.2.4 ReportHandler implements HttpHandler

Обрабатывает POST-запросы на /report:

- читает тело запроса (XML);
- извлекает имя компьютера из XML;
- сохраняет XML в файл с именем вида ИмяКомпьютера_ГГГГММДД_ччммсс.xml;
- обновляет время последнего отчёта в computersDB;
- возвращает HTTP 200 при успехе.

3.2.2.5 Вспомогательные классы для данных

- DiskInfo – информация о диске: буква, процент свободного места, свободно ГБ, всего ГБ.
- DiskAlert – предупреждение по диску: компьютер + DiskInfo.
- BackupInfo – информация о бэкапе: папка, имя последнего файла, дата, время, размер в байтах, отображаемый размер, флаг isOk, строка статуса.
- BackupAlert – предупреждение по бэкапу: компьютер, объект BackupInfo (может быть null, если бэкапы отсутствуют), причина.

3.2.3 Основные функциональные модули сервера

3.2.3.1 HTTP/HTTPS сервер

- запускается в методах startHttpServer() / startHttpsServer();
- при использовании HTTPS создаётся SSLContext из указанного keystore;
- на сервере регистрируется контекст /report с обработчиком ReportHandler;
- сервер работает в многопоточном режиме (пул Executors.newCachedThreadPool()).

3.2.3.2 Обработка отчётов (processReportsInternal)

- сканирует папку с XML-файлами, для каждого компьютера оставляет самый свежий файл;
- парсит каждый XML (используя DOM), извлекает:
- список дисков (<Disk>) – буква, свободно %, свободно ГБ, всего ГБ;
- список бэкапов (<Backup>) – папка, последний файл, дата, время, размер;
- для каждого диска сравнивает процент с порогом (индивидуальным или общим) – формирует DiskAlert при превышении.
- для каждого бэкапа (если для ПК включена проверка) проверяет:
- возраст (сравнивает с backupMaxAgeHours);
- размер (сравнивает с backupMinSizeMB, индивидуальным или общим);
- при несоответствии формирует BackupAlert;
- обновляет список компьютеров в GUI;
- выводит детальную информацию в текстовую область;
- вызывает showAlerts() для визуального оповещения;
- отправляет email-отчёт, если он не пустой (с учётом sendOnlyNegative).

3.2.3.3 Отправка почты (sendReport)

- формирует тело письма в текстовом формате с детализацией по каждому ПК;
- использует javax.mail с настройками SMTP из конфигурации;
- в зависимости от флага sendOnlyNegative может отправлять письмо только при наличии проблем;
- добавляет в письмо сводную статистику (количество ПК, проблемы, индивидуальные настройки).

3.2.3.4 Планировщики задач

- основной (scheduler) – запускает processReportsSilent() (обработка без обновления GUI) с интервалом checkIntervalMinutes;
- очистки (cleanupScheduler) – выполняет удаление всех XML-файлов из папки отчётов с заданной периодичностью (ежедневно или еженедельно в указанное время).

3.2.3.5 База данных компьютеров (computers.db)

- хранит записи в текстовом файле (построчно, поля разделены ;);
- позволяет сохранять индивидуальные настройки между перезапусками;
- при загрузке проверяет целостность, при ошибке создаёт резервную копию .bak.

3.3 Клиентская часть (DiskReportClientGUI)

3.3.1 Основной класс DiskReportClientGUI

Наследует JFrame и предоставляет интерфейс для настройки клиента, выбора дисков и папок бэкапов, ручной и автоматической отправки отчётов.

3.3.1.1 Ключевые поля

- config – объект настроек клиента (ClientConfig);
- scheduler – планировщик для периодической отправки;
- diskCheckboxes, backupFolderCheckboxes – карты для связи файлового объекта с чекбоксом;
- GUI-компоненты: serverField, portField, computerNameField, intervalSpinner, logArea и др.

3.3.1.2 Основные методы

- initComponents() – создание интерфейса с вкладками «Диски» и «Бекапы»;
- loadConfig(), saveConfig() – загрузка/сохранение JSON-конфигурации (diskmonitor_client.cfg);
- refreshDisks() – сканирование корневых дисков системы и создание чекбоксов;
- refreshBackupFolders() – отображение выбранных папок с бэкапами из конфигурации;

- addBackupFolder(), removeSelectedBackupFolders() – диалоги добавления/удаления папок;
- sendReport() – формирование и отправка отчёта в отдельном потоке;
- startScheduler(), stopScheduler() – управление автоматической отправкой;
- log() – добавление сообщения в лог с временной меткой;
- minimizeToTray(), openFromTray() – работа с треем.

3.3.2 Внутренние классы клиента

3.3.2.1 ClientConfig

- computerName – имя компьютера (по умолчанию "MyComputer");
- serverAddress, serverPort – адрес и порт сервера;
- intervalMinutes – интервал отправки;
- selectedDisks – список путей выбранных дисков;
- selectedBackupFolders – список путей выбранных папок с бэкапами;
- backupFilePrefix – префикс для фильтрации файлов бэкапов;
- useHTTPS, trustAllCertificates – настройки протокола и доверия сертификатам.

3.3.2.2 BackupInfo (внутри методов)

Вспомогательный класс для хранения информации о последнем файле бэкапа: имя, размер, дата, время, отображаемый размер.

3.3.2.3 ConsoleModeRunner

Статический вложенный класс для работы в консольном режиме (запуск с аргументом --console):

- загружает конфигурацию;
- при --send-now отправляет отчёт один раз и завершается;
- иначе запускает бесконечный цикл с отправкой по расписанию, логируя в stdout.

3.3.3 Основные функциональные модули клиента

3.3.3.1 Сбор информации о дисках

использует File.listRoots() для получения доступных дисков;
для каждого выбранного диска вычисляет:

- totalGB = root.getTotalSpace() / 1073741824.0;
- freeGB = root.getFreeSpace() / 1073741824.0;
- freePercent = (freeGB * 100) / totalGB.

3.3.3.2 Сбор информации о бэкапах

- для каждой выбранной папки сканирует файлы;
- если задан префикс, учитываются только файлы, начинающиеся с него (без учёта регистра);
- находит самый свежий файл по дате модификации;
- формирует дату, время и размер (в байтах и в человекочитаемом формате).

3.3.3.3 Генерация XML-отчёта (generateXmlReport)

- создаёт DOM-документ с корневым элементом `<DiskReport>`;
- добавляет `<Computer>`, `<Timestamp>`;
- для каждого диска создаёт `<Disk>` с дочерними элементами: `<DriveLetter>`, `<TotalSizeGB>`, `<FreeSpaceGB>`, `<FreeSpacePercentage>`;
- для каждого бэкапа создаёт `<Backup>` с элементами: `<Folder>`, `<LatestFile>`, `<Date>`, `<Time>`, `<Size>`, `<DisplaySize>`, `<FilterPrefix>`;
- возвращает строку с отформатированным XML.

3.3.3.4 Отправка отчёта (sendXmlReport)

- открывает `HttpURLConnection` (или `HttpsURLConnection`) по адресу `http[s]://server:port/report`;
- если включён `trustAllCertificates`, подменяет `TrustManager` на доверяющий всем сертификатам и отключает проверку имени хоста;
- отправляет POST-запрос с XML в теле;
- проверяет код ответа (должен быть 200).

3.3.3.5 Планировщик

- использует `ScheduledExecutorService` с интервалом из конфигурации;
- при изменении настроек перезапускает планировщик;
- в консольном режиме планировщик работает аналогично.

3.3.3.6 Графический интерфейс

- вкладка «Диски» – список дисков с чекбоксами, кнопка обновления;
- вкладка «Бекапы» – список папок с чекбоксами, кнопки добавления/удаления/обновления;
- панель лога с прокруткой;
- возможность свернуть в трей, откуда можно открыть окно или отправить отчёт вручную.

3.3.3.7 Консольный режим

- предназначен для запуска на серверах без графической оболочки;
- читает ту же конфигурацию, что и GUI;
- работает до прерывания (Ctrl+C) или сразу завершается после отправки (--send-now).

3.4 Взаимодействие клиента и сервера

1. Клиент по расписанию (или вручную) генерирует XML-отчёт.
2. Отправляет его методом POST на конечную точку /report сервера.
3. Сервер сохраняет XML в указанную папку, обновляет время последнего отчёта.
4. При следующей плановой обработке сервер анализирует все накопленные файлы и принимает решение об оповещении.
5. При необходимости сервер отправляет email-уведомление с подробным отчётом.

Таким образом, сервер не инициирует соединения с клиентами – клиенты сами «стучатся» на сервер, что упрощает сетевую архитектуру (не нужны открытые порты на клиентах).

4. Применяемые технологии разработки программного обеспечения

Разработка программного комплекса велась с использованием современных технологий, инструментов и библиотек экосистемы Java.

4.1 Язык программирования

Java – приложение полностью написано на Java, что обеспечивает кроссплатформенность и доступ к богатой экосистеме библиотек.

4.2 Платформа

Java Standard Edition (Java SE) – используется базовый набор API (I/O, сеть, многопоточность, GUI).

4.3 Графический интерфейс пользователя (GUI)

Swing (javax.swing) – основа графического интерфейса как серверной, так и клиентской части.
Применяются компоненты: JFrame, JPanel, JButton, JTextField, JSpinner, JCheckBox, JList, JTabbedPane, JTextArea, JScrollPane, JDialog и др.
AWT (java.awt) – используется для работы с системным треем (SystemTray, TrayIcon), управления цветами, шрифтами, компоновкой (GridBagLayout, BorderLayout), а также для обработки событий и рисования (Graphics2D).

4.4 Сетевое взаимодействие

Встроенный HTTP-сервер (com.sun.net.httpserver) – на серверной части применяется HttpServer и HttpsServer для приёма POST-запросов от клиентов. HttpURLConnection / HttpsURLConnection (java.net) – на клиенте используется для отправки XML-отчётов методом POST. Поддерживается как HTTP, так и HTTPS с возможностью настройки TrustManager для самоподписанных сертификатов.

4.5 Работа с XML

DOM (Document Object Model) из пакета javax.xml.parsers – как на сервере, так и на клиенте. Сервер парсит входящие XML-файлы, извлекая информацию о дисках и бэкапах. Клиент строит XML-документ программно с последующей сериализацией в строку.

4.6 Работа с JSON

Google Gson – библиотека для сериализации/десериализации Java-объектов в JSON и обратно. Используется для сохранения и загрузки конфигурационных файлов (diskmonitor.cfg, diskmonitor_client.cfg).

4.7 Отправка электронной почты

JavaMail (javax.mail) – применяется для отправки отчётов и предупреждений по электронной почте. Настройки SMTP-сервера, аутентификации и получателей задаются в конфигурации.

4.8 Безопасность и SSL/TLS

JSSE (Java Secure Socket Extension) – встроенные средства Java для работы с SSL/TLS. На сервере создаётся SSLContext на основе keystore, задаётся HttpsConfigurator.

4.9 Многопоточность и планирование задач

java.util.concurrent – основной инструмент для асинхронной работы:

- ScheduledExecutorService – используется для периодического запуска обработки отчётов (на сервере) и отправки отчётов (на клиенте);Executors.newCachedThreadPool() – для обработки входящих HTTP-запросов.

4.10 Системный трей

java.awt.SystemTray – позволяет свернуть приложение в трей, где оно продолжает работу в фоновом режиме. Иконка трея реагирует на клики и отображает всплывающие уведомления.

4.11 Хранение данных

Файловая система – все данные хранятся в файлах:

- конфигурация сервера и клиента в формате JSON (с использованием Gson);
- база данных компьютеров (computers.db) в текстовом формате (CSV-подобный, разделитель ;);
- XML-отчёты сохраняются на диске сервера с именем, содержащим имя компьютера и временную метку.

4.12 Средства сборки и управления зависимостями

Для сборки проекта используется Maven.

4.13 Инструменты логирования и отладки

системный вывод (System.out, System.err) – используется для логирования основных событий и ошибок. В GUI-версии логи выводятся также в текстовую область (JTextArea);
стек-трейсы исключений выводятся в консоль (e.printStackTrace()).

4.14 Среда выполнения

Приложение требует Java Runtime Environment (JRE) версии 17 или выше.