Інтернаціоналізація програм в Java

Интернационализация (internationalization) – это процесс проектирования программы таким образом, чтобы она могла использоваться для различных языков и регионов без перепрограммирования.

Интернационализированная программа имеет следующие характеристики:

- при добавлении локализованных данных эта программы может быть выполнена в любой стране;
- текстовые элементы, такие как сообщения или надписи в графических приложениях и апплетах, не встроены в программу, а хранятся отдельно и доступ к ним происходит во время выполнения программы;
- включение новых языков не требует перекомпиляции программы;
- региональные (зависящие от языка и страны) данные, такие как дата, представление денежных единиц и чисел, выводятся в форматах, принятых в данном регионе.

Локализация – это процесс адаптации программного обеспечения для определенного региона или языка с помощью добавления зависящих от языка или региона компонент, а также перевода текста.

Обычно подавляющую часть времени, требуемого на локализацию, занимает перевод текста. При локализации производится проверка соответствуют ли форматы представления дат, чисел и денежных единиц региональным и языковым требованиям.

Средства интернационализации и локализации программ в Java включают в себя следующие компоненты:

- определение и установка языка и региона, в котором выполняется программа;
- преобразование текста в различные кодировки символов;
- обработка текста (проверка свойств символов, сравнение строк, определение границ текста);
- изоляция выводимого пользователю текста от программы;
- форматирование выводимых значений даты, времени, чисел, денежных единиц и сообщений;

Приведенные компоненты содержатся в пакетах java.util и java.text

Колекції в Java

Коллекция, иногда называемая контейнером, – это объект, который объединяет несколько элементов в один объект. Коллекции используются для хранения данных, доступа и манипуляций с данными, а также для передачи данных от одного метода к другому.

Коллекция обычно представляет данные, которые составляют естественную группу, например телефонный справочник (коллекция соответствий между именем и телефонным номером). Массив также можно рассматривать как коллекцию, объединяющую данные одного типа, элементы которой расположены последовательно, в порядке возрастания индекса.

Все интерфейсы и классы, относящиеся к коллекциям, находятся в пакете **java.util**.

Схема коллекций (collections framework) – это унифицированная архитектура для представления и манипулирования коллекциями. Все схемы коллекций содержат следующие три компоненты:

1. **Интерфейсы** – абстрактные типы данных, представляющие коллекции. Интерфейсы позволяют манипулировать коллекциями независимо от деталей их представления. В Java, как и в других объектно-ориентированных языках, эти интерфейсы обычно образуют иерархию.
2. **Реализации** – конкретные реализации интерфейсов коллекции. По своей сути они являются повторно используемыми структурами данных.
3. **Алгоритмы** – методы, выполняющие некоторые полезные действия, например, поиск или сортировку, над объектами, реализующими интерфейсы коллекций. Эти методы называют полиморфными, так как один и тот же метод может использоваться во многих различных реализациях соответствующего интерфейса коллекций. По сути алгоритмы обеспечивают повторно используемую функциональность.

Робота з СУБД в Java

Файлы с текстами программ используются как входные тексты компиляторов, которые в свою очередь формируют файлы, содержащие объектные модули. Система программирования накладывает на эту структуру более сложную и специфичную для этой системы структуру объектного модуля.

Аналогично обстоит дело с файлами, формируемыми редакторами связей и содержащими образы выполняемых программ. Логическая структура таких файлов остается известной только редактору связей и загрузчику – программе операционной системы. Примерно такая же ситуация с файлами, содержащими графическую и звуковую информацию.

Таким образом, файловые системы обычно обеспечивают хранение слабо структурированной информации, оставляя дальнейшую структуризацию прикладным программам.

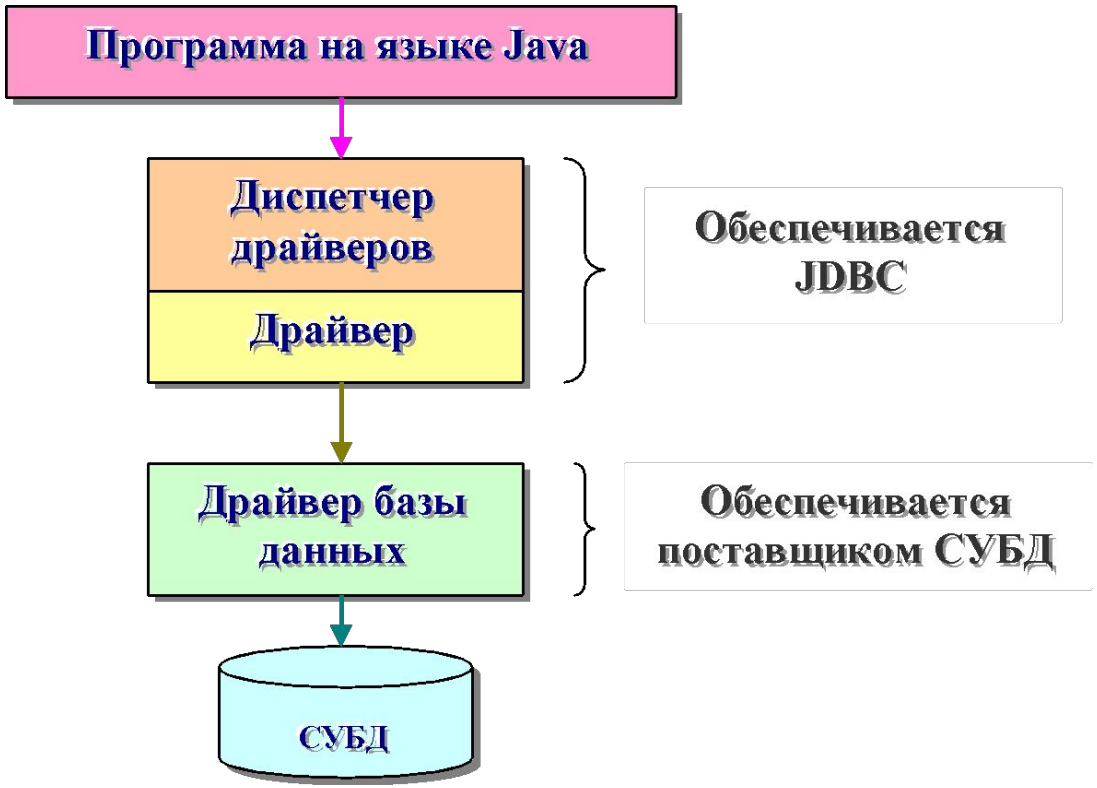
Стремление выделить и обобщить общую часть информационных систем, ответственную за управление сложно структурированными данными, явилось первой причиной создания **систем управления базами данных** (СУБД). Невозможно обойтись общей библиотекой программ, реализующей над стандартной базовой файловой системой более сложные методы хранения данных.

При операциях, связанных с добавлением, удалением или изменением записей или полей записей информационная система на основе использования файлов требует создания временных файлов, в которых происходит изменение данных. Эти операции, как правило, приходится выполнять для каждого изменения, что приводит к существенным накладным расходам по памяти и времени при многократном изменениям исходных данных.

Таким образом, традиционных возможностей файловых систем оказывается недостаточно для построения даже простых информационных систем. Поэтому развитие информационных систем привело к появлению систем управления данными, отличающимися от файловых систем, используемых в компьютере.

Первоначально практически для каждой реализации СУБД разрабатывался свой язык манипулирования данными. Однако в настоящее время самым распространенным языком такого рода для СУБД является язык последовательных запросов – **SQL (Structured Query Language).**

Для подключения к базе данных среда выполнения Java должна загрузить соответствующий драйвер указанной базы данных. Загрузка и выгрузка таких драйверов осуществляется с помощью класса **DriverManager** пакета **java.sql**.



Мережеві засоби Java

Язык Java обеспечивает мощные средства создания сетевых приложений для серверов и клиентов при помощи сравнительно небольших программ. Сетевые средства Java содержат следующие компоненты:

- средства, обеспечивающие работу в сети Internet (пакет **java.net** в Java 2 Standard Edition);
- средства доступа к базам данных (пакеты **java.sql** и **javax.sql** в Java 2 Standard Edition и пакет **javax.sql** в Java 2 Enterprise Edition);
- поиск распределенных объектов в сети JNDI (Java Naming and Directory Interface) (группа пакетов **javax.naming** в Java 2 Standard Edition);
- механизм распределенных объектов RMI (группа пакетов **java.rmi** в Java 2 Standard Edition);
- система обмена сообщениями между частями распределенного приложения JMS (Java Message Service) (пакет **javax.jms** в Java 2 Enterprise Edition);
- средства обработки данных на Web-сервере (группа пакетов **javax.servlet** и **javax.ejb** в Java 2 Enterprise Edition).

Засоби обробки даних на сервері в Java

Средства обработки данных на сервере (и, в частности на Web-сервере) представляют так называемое промежуточное (**middleware**) программное обеспечение, которое служит связующим звеном между клиентскими приложениями и данными, необходимыми для клиентских приложений.

В языке Java для обработки данных на сервере определены следующие два компонента:

1. Web-компоненты (Web components);
2. Компоненты, ориентированные на бизнес-приложения (business components).

Web-компоненты в Java представлены двумя тесно связанными между собой технологиями: сервлетами и серверными страницами Java – JSP (Java Server Pages).

Сервлеты Java дают возможность Web-серверу обрабатывать запросы с помощью программ, написанных на языке Java. Специальным образом оформленная программа запускается в виде отдельного потока, после чего сервер передает ей данные для обработки и получает ответ.

Сервлеты выполняются под управлением Web-сервера так же, как апплеты выполняются под управлением Web-браузера. Сервлеты запускаются в виде потоков. Они также позволяют хранить информацию о состоянии текущего соединения и поддерживают разделение данных между различными экземплярами программы. Кроме того, сервлеты могут пересылать запросы другим серверам и сервлетам.

Таким образом, сервлеты можно применять для распределения нагрузки между несколькими серверами, которые отражают одинаковое содержание, а так же для распределения одного логического сервиса по нескольким серверам, в соответствии с поставленной задачей.

Технология JSP позволяет включать в документ HTML или XML обрабатываемые на сервере сценарии на языке Java.
Компонентами, ориентированными на приложения, являются **EJB (Enterprise Java Beans).** Контейнеры и серверы EJB предоставляют услуги по организации взаимодействия между компонентами распределенного приложения, позволяют работать с данными и обеспечивать жизнедеятельность системы.

В Java определено три типа EJB: компоненты сеанса (Session Beans), компоненты объектов (Entity Beans) и управляемые сообщениями компоненты (Message-driven Beans).
Компоненты сеанса обеспечивают диалог клиента с приложением. При завершении соединения с клиентом эти компоненты удаляются из системы.
Компоненты объектов содержат данные. Если клиент заканчивает работу или сервер останавливается, соответствующие службы обеспечивают сохранность компонентов объектов.
Управляемые сообщениями компоненты объединяют свойства компонентов сеанса и блоков прослушивания службы сообщений Java (JMS), что позволяет компонентам EJB асинхронно получать сообщения JMS.

Сервлеты представляют собой обычные программы на языке Java, но, так же как апплеты, они имеют структуру, отличную от приложений Java. **Для слежения за работой сервлетов и управления ими создается специальный программный модуль, называемый контейнером сервлетов** (servlet container). Контейнер сервлетов загружает сервлеты, инициализирует их, передает им запросы клиентов, принимает ответы.

Установка (deployment) сервлета в контейнер включает получение уникального имени и определение начальных параметров сервлета, запись их в конфигурационные файлы, создание каталогов для хранения всех файлов сервлета и другие операции.

Один контейнер может управлять работой нескольких установленных в него сервлетов. При этом один контейнер может в одно и то же время работать в нескольких виртуальных машинах Java, образуя **распределенное Web-приложение.** Сами же виртуальные машины Java могут работать на одном компьютере или на разных компьютерах.

Интерфейс прикладного программирования для сервлетов (Servlet API) содержится в пакетах javax.servlet и javax.servlet.http