Modern Java Technologies

Overview

Why Java?

Features & Benefits

- Java is the foundation for virtually every type of networked application
  - ✓ and is the global standard for developing and delivering mobile applications, games, Web-based content, and enterprise software
- With more than 9 million developers worldwide, Java enables you to efficiently develop and deploy exciting applications and services
- With comprehensive tooling, a mature ecosystem, and robust performance, Java delivers applications portability across even the most disparate computing environments

- Platform Independence
  - ✓ Java runs on most major hardware and OS platforms
- High Performance
  - ✓ HotSpot and JRockit are examples of proven just-in-time virtual machine technologies that make Java one of the fastest programming environments
  - ✓ Built in optimizations for multi-threaded environments make it even faster
- Easy to Learn
  - ✓ Java's model for memory management, multi-threading, and exception handling make it a powerful language for both new and skilled developers

- Standards-Based

  ✓ The Java language and related technology evolves via the Java Community Process, a mechanism for developing technical specifications for Java technology

- Worldwide Prevalence

  ✓ Java is the most popular application platform on the planet and delivers a vibrant developer ecosystem fueled by powerful tools, books, libraries, code samples, and more

- Consistent Runtime Environments

  ✓ Java enables you to deploy with confidence with runtime environments spanning Java SE on the desktop, Java SE for Embedded Devices, and Oracle Java Micro Edition Embedded Client

- Optimized for Embedded

  ✓ Java SE for Embedded Devices includes support for key requirements such as embedded processor support, power management, small-footprint deployments, and more

- High-performance, Portable Applications
  - ✓ Java achieves native performance while providing portability across a range of embedded processors and operating systems
- Proven Security Model
  - ✓ Java delivers an advanced, highly secure application environment that is ideal for network-based applications
- Java Platform, Enterprise Edition (Java EE)
  - ✓ Java EE includes lightweight Web Profile to create next-generation Web applications and the full power of the Java EE platform for enterprise applications

- Java has a rich set of standard libraries
  - ✓ Networking
  - ✓ Threads (lightweight processes)
  - ✓ Distributed objects
  - ✓ Database access
  - ✓ Graphics: GUI controls and drawing
  - ✓ Data structure library
  - ✓ Arbitrary precision integral and fixed-point arithmetic
  - ✓ Digital signatures
  - ✓ Serialization (transmitting/reassembling data structures)
  - ✓ File and stream compression
  - ✓ XML parsing
  - ✓ Web services

Java Versions

SE, EE, ME

- Java SE
  - ✓ This is often what people mean when they say “Java” or “the Java programming language”
  - ✓ Applications
    - ❖ Desktop programming
    - ❖ Applets
    - ❖ Java WebStart
    - ❖ Java FX
    - ❖ Base on which to build Web apps that are not full Java EE
  - ✓ Examples - Eclipse, NetBeans, Yahoo games, GWT, …

- Java SE is designed to enable you to develop secure, portable, high-performance applications for the widest range of computing platforms possible
- By making applications available across heterogeneous environments, businesses can boost end-user productivity, communication, and collaboration
  - ✓ and dramatically reduce the cost of ownership of both enterprise and consumer applications

- Java EE
  - ✓ This is Java running on application servers
  - ✓ Applications
    - Servlets
    - JSP
    - JSF
    - Struts
    - EJB
    - Spring
    - Hibernate
  - ✓ Examples - Google home page, gmail, Google Maps, Google Docs, Ebay, PayPal, …

- Java EE initially evolved as an enterprise application deployment platform that focused on
  - ✓ robustness
  - ✓ Web services
  - ✓ and ease of deployment
- Java EE represents a universal standard in enterprise IT, facilitating the development, deployment, and management of multi-tier, server-centric applications

- Java EE 7 enables developers to deliver HTML5 dynamic scalable applications
- New to the platform, WebSockets reduce response time with low latency bi-directional data exchange
- Standard JSON support simplifies data parsing for portable applications
- JAX-RS has been improved to deliver asynchronous, scalable, high performance RESTful Services

- Java EE increases developer productivity in multiple ways
- It offers a simplified application architecture with a cohesive integrated platform;
- Java EE increases efficiency with reduced boiler-plate code and broader use of annotations
- Java EE enhances application portability with standard RESTful Web service client support
- Java easily defines multithreaded concurrent tasks for improved scalability
- It breaks down batch jobs into manageable chunks for uninterrupted performance

- Java ME
  - ✓ This is Java running on small devices
  - ✓ Applications
    - ❖ Cell phone apps
    - ❖ embedded apps
    - ❖ printers
    - ❖ etc.
  - ✓ Examples – Android, Amazon Kindle, Blackberry, All Blu-Ray DVD players, …

Java EE 7

Modern Technologies

- Deliver Dynamic Scalable HTML5 Applications
  - ✓ HTML5 is accelerating the ability of developers to create applications that are highly interactive and dynamic, alongside client-side technologies like JavaScript and CSS3
- These applications can deliver
  - ✓ live data feeds such as sport scores
  - ✓ stock news and quotes
  - ✓ application notifications
  - ✓ twitter and chat feeds
  - ✓ and more, all in a highly interactive manner

- Deliver Dynamic Scalable HTML5 Applications
  - ✓ HTML5 enables these applications to be written once and render properly on a range of devices like smart phones, tables, and desktops
    - ❖ These highly dynamic applications, combined with the ability to access them at any time from anywhere, are driving the need to scale the services that feed application data to the client
  - ✓ Java EE 7 lays the foundation for dynamic HTML5 applications with new JSRs like WebSockets and JSON Processing, along with updates to existing JSRs like JAX-RS 2.0, Java Server Faces 2.2, and Servlet 3.1 NIO

- Low Latency Data Exchange Using the Java API for WebSocket 1.0
  - ✓ A growing number of Web applications rely on timely updates from central servers.
  - ✓ WebSockets offer a solution to the inherent problems of latency and bi-directional communication that come with HTTP-based solutions like polling, long-polling and HTTP-streaming
  - ✓ The WebSocket API, at its most basic level, supports sending and receiving simple text and binary messages
  - ✓ The simplicity of the API enables developers to get started quickly

- Simplify Data Parsing for Portable Applications with Java API for JSON Processing 1.0
  - ✓ JSON (JavaScript Object Notation), a lightweight data-interchange format, is used by many popular Web services to invoke and return textual data
  - ✓ Many popular online services, like Twitter, Facebook, and Pinterest, expose RESTful services that exchange JSON objects
  - ✓ With the Java API for JSON Processing 1.0, JSON processing is standardized into a single API so that applications that use JSON need not bundle 3rd party implementation libraries

➢ Scalable RESTful Services with Java API for RESTful Web Services 2.0 – JAX-RS 2.0

✓ JAX-RS 2.0 adds asynchronous response processing, which is critical for scaling to meet the demands of data-hungry HTML5 clients

✓ Asynchronous processing is a technique that enables a better and more efficient use of processing threads

✓ On the server side, a thread that is processing a request should avoid blocking while waiting for an external task to complete so that other requests arriving at the server during that period of time can be attended

- Scalable RESTful Services with Java API for RESTful Web Services 2.0 – JAX-RS 2.0

  - ✓ JAX-RS 2.0 adds asynchronous response processing, which is critical for scaling to meet the demands of data-hungry HTML5 clients

  - ✓ Asynchronous processing is a technique that enables a better and more efficient use of processing threads

  - ✓ On the server side, a thread that is processing a request should avoid blocking while waiting for an external task to complete so that other requests arriving at the server during that period of time can be attended

  - ✓On the client side, a thread that issues a request will block while waiting for a response, impacting the performance of the app

- Enhanced Ease of Development with JavaServer Faces 2.2
  - ✓ JavaServer Faces is the standard, component-oriented Java EE framework for building portable Web application user interfaces
  - ✓ It maximizes the productivity of Web application development for graphical IDEs
    - ❖ while simultaneously minimizing the complexity of maintenance of the Web application during its production lifetime
  - ✓ With this release, JSF adds support for HTML5

- Enhanced Ease of Development with JavaServer Faces 2.2
  - ✓ JavaServer Faces 2.2 offers HTML5-friendly markup
    - ❖ enabling page authors to write “pure” HTML markup that can be viewed in an HTML tool
    - ❖ or simply rendered in a browser page as HTML formatted code without any clunky XML markup
  - ✓ Any JSF attributes that are preceded by “jsf:” are ignored by the browser and passed on to the server
  - ✓ With passthrough elements, the JSF renderer ignores the elements, and instead just passes them to the browser, which renders them properly
    - ❖ enabling existing JSF components to "utilize" HTM5 features

- Improved Request Processing with Servlet 3.1 NIO
  - ✓ HTML5 applications are inherently more dynamic, and drive many more requests to the server for information updates
    - ❖ In Java EE 6, Servlet Asynchronous I/O enabled many more concurrent requests by removing the “thread per request” limitation, enabling a thread to handle multiple concurrent requests
    - ❖ This can help deliver necessary data to an HTML5 client in a scalable manner
    - ❖ However, if the server can read data faster than the client can send it, perhaps due to a slow client connection, the thread will block until more data is available

- Improved Request Processing with Servlet 3.1 NIO
  - ✓ With Servlet 3.1 NIO, reading data from a client is non-blocking when using the new event-driven API
  - ✓ When data is available, a Servlet thread can read and process just that data, and then move on to another request

- Cohesive, Integrated Platform
  - ✓ Java EE 7 brings the ease of use of EJB container managed transactions to the platform as a whole
    - ❖ using a more general solution based on CDI interceptors so that these can be used by CDI managed beans and other Java EE components
  - ✓ Java EE 6 introduced Managed Beans 1.0 as the first step towards aligning EJBs, JSF Managed Beans, and CDI beans
  - ✓ With Java EE 7, managed bean alignment continues. For example, JSF Managed Beans has begun the pruning process in favor of CDI Beans

- Leading IDEs such as NetBeans and Eclipse can be used to develop applications and other components for Java EE 7
  - ✓ Such IDEs support virtually all the Java EE capabilities
- NetBeans provides comprehensive support of the Java EE 7 platform and bundles GlassFish so you can get started quickly
  - ✓ NetBeans also includes wizards to rapidly create JAX-RS 2.0 Interceptors and Filters, JSF 2.2 Faces Flow, WebSocket Endpoints, and more
- Eclipse includes support for Java EE 7, and the Oracle Enterprise Pack for Eclipse (OEPE) 12.1.2 in the Eclipse Marketplace hosts the GlassFish plugin