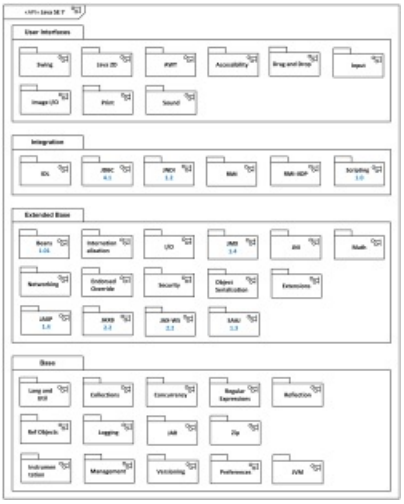
UML Diagram Examples
Java™ Technology

Here we provide several UML diagrams of different types, related to Java™ technology and APIs.

Java™ Platform Standard Edition 7 API UML package diagram example.

***Purpose**: An example of UML **package diagram** representing **Java™ Platform Standard Edition (SE) 7 API**.*

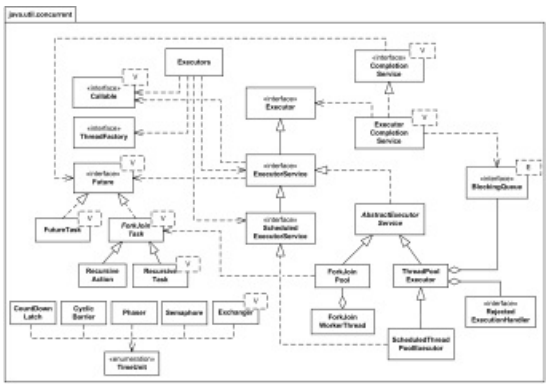
***Summary**: Java™ SE 7 API is comprised of several modules - User Interface and Toolkits APIs, APIs of Integration Libraries, Other Base Libraries APIs, lang and util Base Libraries APIs, Java Virtual Machine (JVM) API. Java SE 7 API includes such well known APIs as Swing, AWT, JavaBeans, JDBC, JAXP, JAX-WS, JAR.*



Java util.concurrent API UML class diagram examples

***Purpose**: Examples of UML **class diagram** representing most important interfaces and classes of **Java™ util.concurrent** API. Several java.util.concurrent.* packages support high-level concurrency features in Java with the new concurrent data structures in the Java Collections framework.*

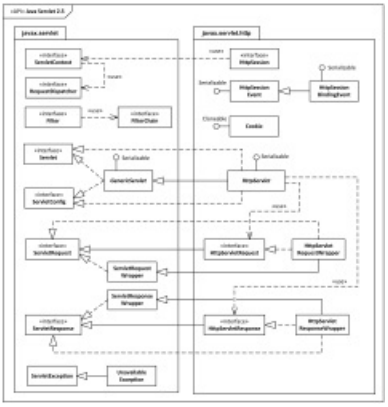
***Summary**: Executors define a high-level API for launching and managing threads to support large-scale applications mostly by adding thread pool management abilities. Concurrent collections reduce the need for synchronization and are designed to support concurrent access and modifications of the large collections of data. The Future<V> interface represents the result of an asynchronous computation.*



Java Servlet 2.5 API - package diagram

***Purpose**: An example of UML **package diagram** representing most important interfaces and classes of **Java™ Servlet 2.5 API**.*

***Summary**: Java Servlet 2.5 API consists of two packages: **javax.servlet** and **javax.servlet.http**. The **javax.servlet** package contains a number of interfaces and classes (both abstract and concrete) that describe and define the contracts between a servlet class and the runtime environment provided for an instance of such a class by a conforming servlet container. The **javax.servlet.http** is package specialized for the servlet class running under the **HTTP** protocol and for corresponding runtime environment.*

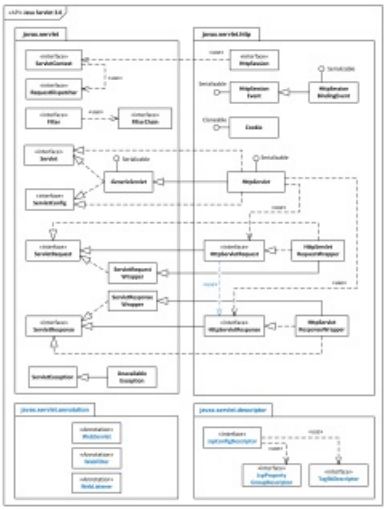


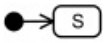
Java Servlet 3.0 API - package diagram

***Purpose**: An example of UML **package diagram** representing most important interfaces and classes of **Java™ Servlet 3.0 API**.*

***Summary**: Java Servlet 3.0 API consists of four packages: **javax.servlet**, **javax.servlet.http**, **javax.servlet.annotation**, and **javax.servlet.descriptor**. The **javax.servlet** package contains a number of interfaces and classes (both abstract and concrete) that describe and define the contracts between a servlet class and the runtime environment provided for an instance of such a class by a conforming servlet container.*

*The **javax.servlet.http** is package containing API interfaces and classes specialized for the servlets supporting **HTTP** protocol and corresponding runtime environment. The **javax.servlet.annotation** and **javax.servlet.descriptor** packages allow to declare servlets, filters and listeners by using annotations and to have access to a web application's configuration information.*

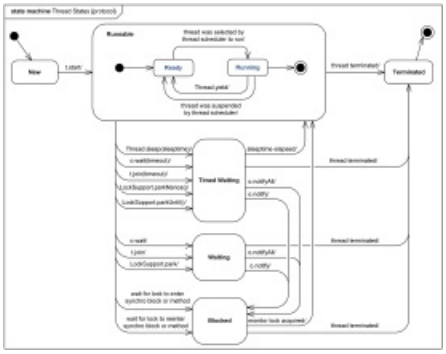




Java Thread states and life cycle - protocol state machine

***Purpose**: An example of UML **protocol state machine** diagram showing **thread states** and **thread life cycle** for the Thread class in **Java™**.*

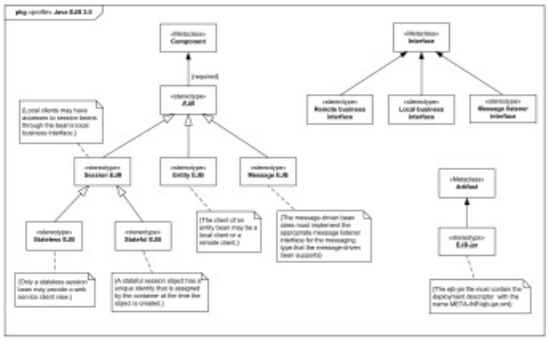
***Summary**: Thread is a lightweight process, the smallest unit of scheduled execution. Instance of the Thread class in **Java** could be in one of the following states: **new**, **runnable**, **timed waiting**, **waiting**, **blocked**, **terminated**.*



Java EJB 3.0 UML Profile

***Purpose**: An example of UML profile diagram for **Enterprise JavaBeans (EJB) 3.0**.*

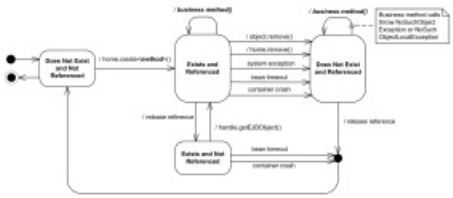
***Summary**: Simplified and unofficial UML profile for EJB 3.0 with support for session, entity, and message-driven **Enterprise JavaBeans**.*



Java EJB life cycle of a session object - UML protocol state machine diagram example.

***Purpose**: Life cycle of an EJB session object is shown from the point of view of a local or remote client using the EJB 2.1 and earlier client view API.*

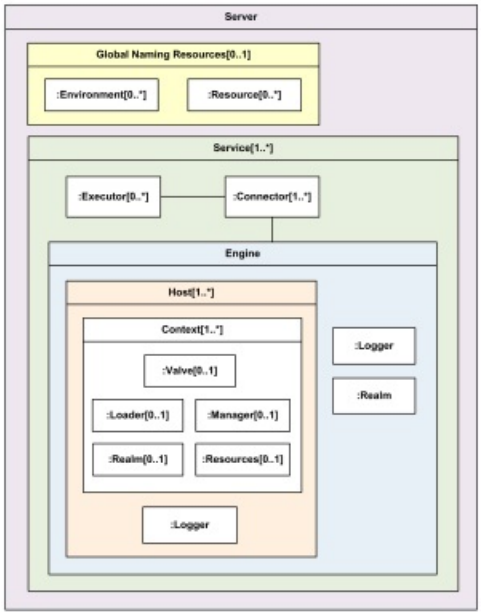
***Summary**: A session object does not exist until it is created. When a client creates a session object, the client has a reference to the newly created session object's component interface.*



Apache Tomcat 7 web server UML composite structure diagram example

***Purpose**: Show a simplified composite structure of a non clustered Apache Tomcat 7 web server.*

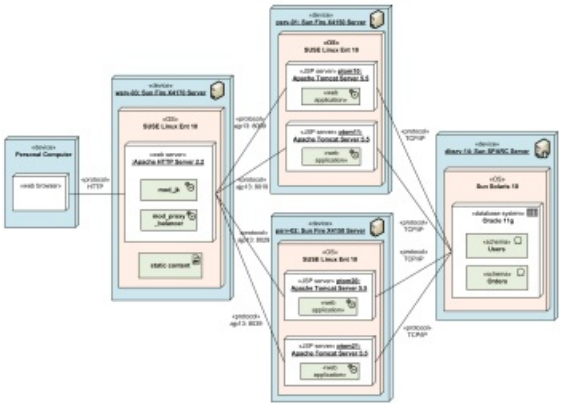
***Summary**: A Server element represents Catalina servlet container of Apache Tomcat 7 web server. It is a single outermost element in the conf/server.xml configuration file. Server element could contain optional Global Naming Resources component and one or more Services. Each Service element is a composition of Executors and Connectors that share a single Engine component.*



Clustered deployment of J2EE web application

***Purpose**: An example of **deployment diagram** for J2EE **web application** with load balancing and clustering which shows specific server instances involved.*

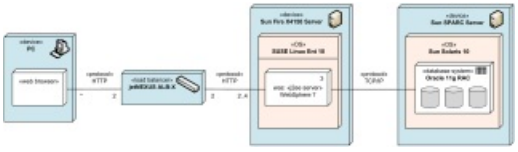
***Summary**: Incoming HTTP requests are first processed by **Apache web server**. Static content such as HTML pages, images, CSS, and JavaScript is served by the web server. Requests to JSP pages are **load balanced** and forwarded to 2x2 Apache Tomcat servers using both vertical and horizontal **clustering**.*



Multilayered load balancing of J2EE servers

***Purpose**: An example of UML **deployment diagram** with hardware and software load balancing and clusters.*

***Summary**: An example shows 2 active hardware load balancers connected to 2 to 4 Sun Fire Servers. Each server has 3 instances of IBM WebSphere 7 J2EE application servers installed.*



Noticed a spelling error? Select the text using the mouse and press Ctrl + Enter.

by Kirill Fakhroutdinov

This document describes **UML 2.5** and is based on **OMG™ Unified Modeling Language™ (OMG UML®) 2.5** specification ***[UML 2.5***

FTF - Beta 1].
All UML diagrams were created in **Microsoft Visio** 2007-2016 using ***UML 2.2 stencils***. You can send your comments and suggestions to webmaster at **webmaster@uml-diagrams.org**.