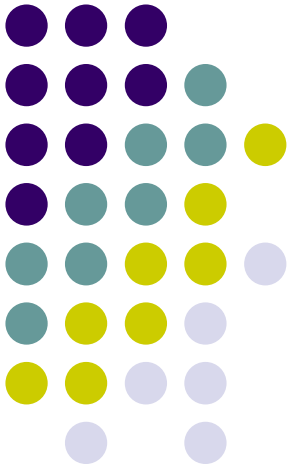
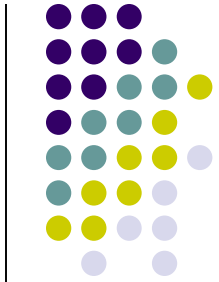
Мрежово програмиране

Технология Java EE



Мотивацията:



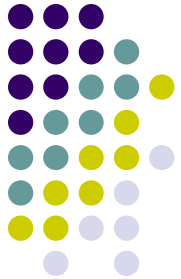
Можем и без

- Java EE
- .NET
- Spring
- Guice
- CICS
- и т.н.

Но тогава ще се наложи самостоятелно да се реализират:

- паралелизъм
- транзакции
- блокировки
- сеанси

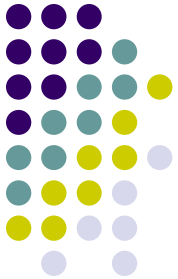
Митове за разпределените изчисления



- Мрежите са надеждни
- Забавянията са нулеви
- Честотната лента е безкрайна
- Топологията не се променя
- Има само един администратор
- Предаването на информацията нищо на струва
- Мрежата е еднородна

Най-развитите платформи за разпределени приложения са Microsoft .NET и Java Enterprise Edition (JEE)

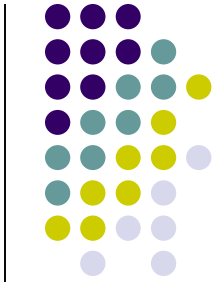
- Изграждането на разпределени системи с високо качество е една от най-сложните задачи при разработването на програмно осигуряване
- Технологиите JEE и .NET се създават именно за това, за да направят създаването на широко разпространените видове разпределени системи (така наричаните бизнес приложения, поддържащи решаването на бизнес задачи от някаква организация) да е достатъчно проста и достъпна практическа задача за всеки програмист



JAVA EE и .Net се използват за езикова платформа на middleware, като CORBA е родоначалник

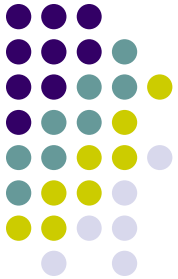
- Основната задача, която се опитват да решат с помощта на разпределените системи, е осигуряването на максимално прост достъп до възможно най-голям брой ресурси до колкото е възможно по-голям брой потребители
- Най-важните свойства на тази система са прозрачност, отвореност, мащабируемост и безопасност.

Java Platform, Enterprise Edition (Java EE)

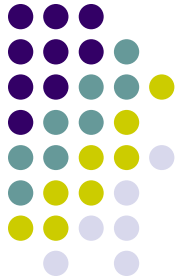


- Набор от спецификации и съответна документация за езика Java, описващи архитектура на сървърна платформа, решаваща задачи на средни и крупни организации.
- Java EE е основно ориентирана за използване чрез уеб, както в Интернет, така и за локални мрежи.

Основна цел на Java EE



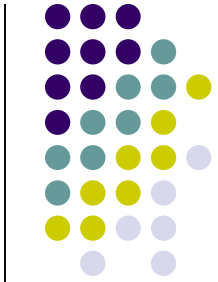
- Осигуряване на мащабируемост на приложенията и цялостност на данните по време на работа на системата.
- Намаляване стойността и увеличаване скоростта на проектиране и разработване на корпоративни приложения.



Платформата JEE предлага компонентен подход към проектиране, разработване и внедряване на корпоративни приложения

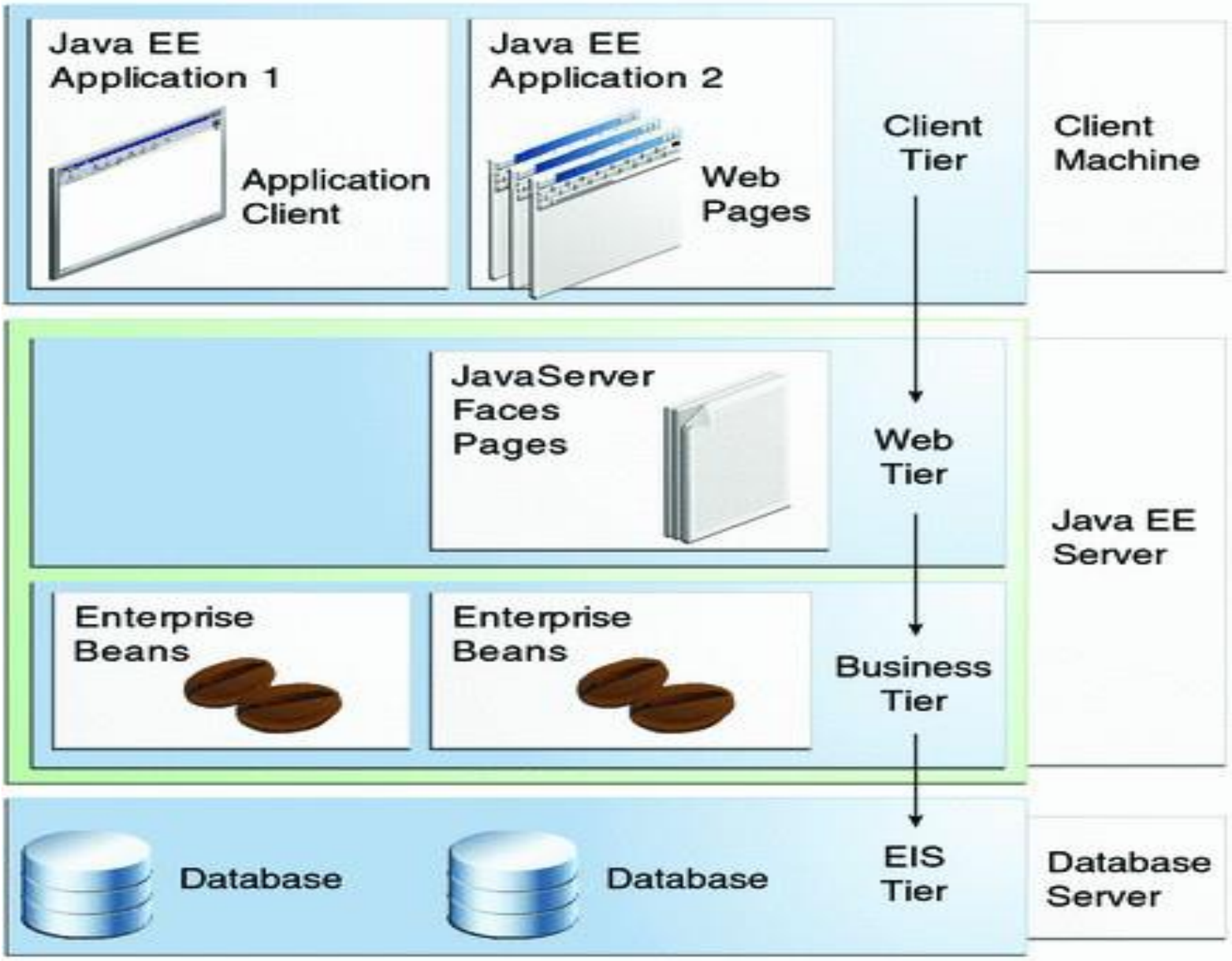
Платформата JAVA EE предлага:

- Модел на многослойно разпределено приложение;
- Възможност за повторно използване на компонентите;
- Интегриран обмен на данни чрез използване на XML;
- Унифициран модел на безопасност;
- Гъвкаво управление на транзакциите.

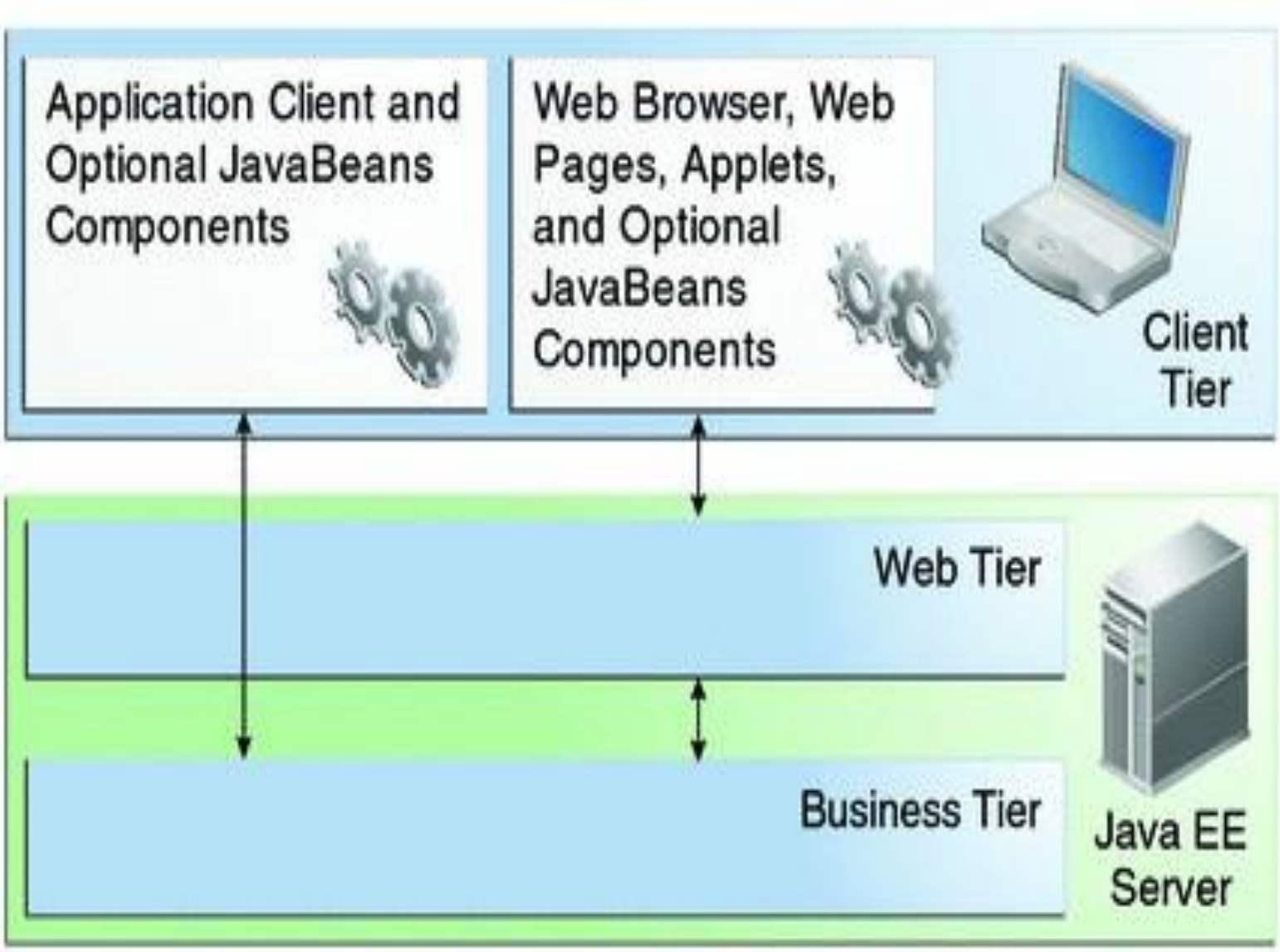


Java EE компонент е самостоятелна, функционална, софтуерна единица, сглобена в Java EE приложение, с нейните класове и файлове, която общува с други компоненти.
Архитектурата на JAVA EE чрез контейнери поддържа различни модели на приложните услуги за безопасност, транзакции и именоване.
Контейнерите отговарят за жизнения цикъл на JAVA EE компонентите и за взаимодействието със съхраняваните данни.

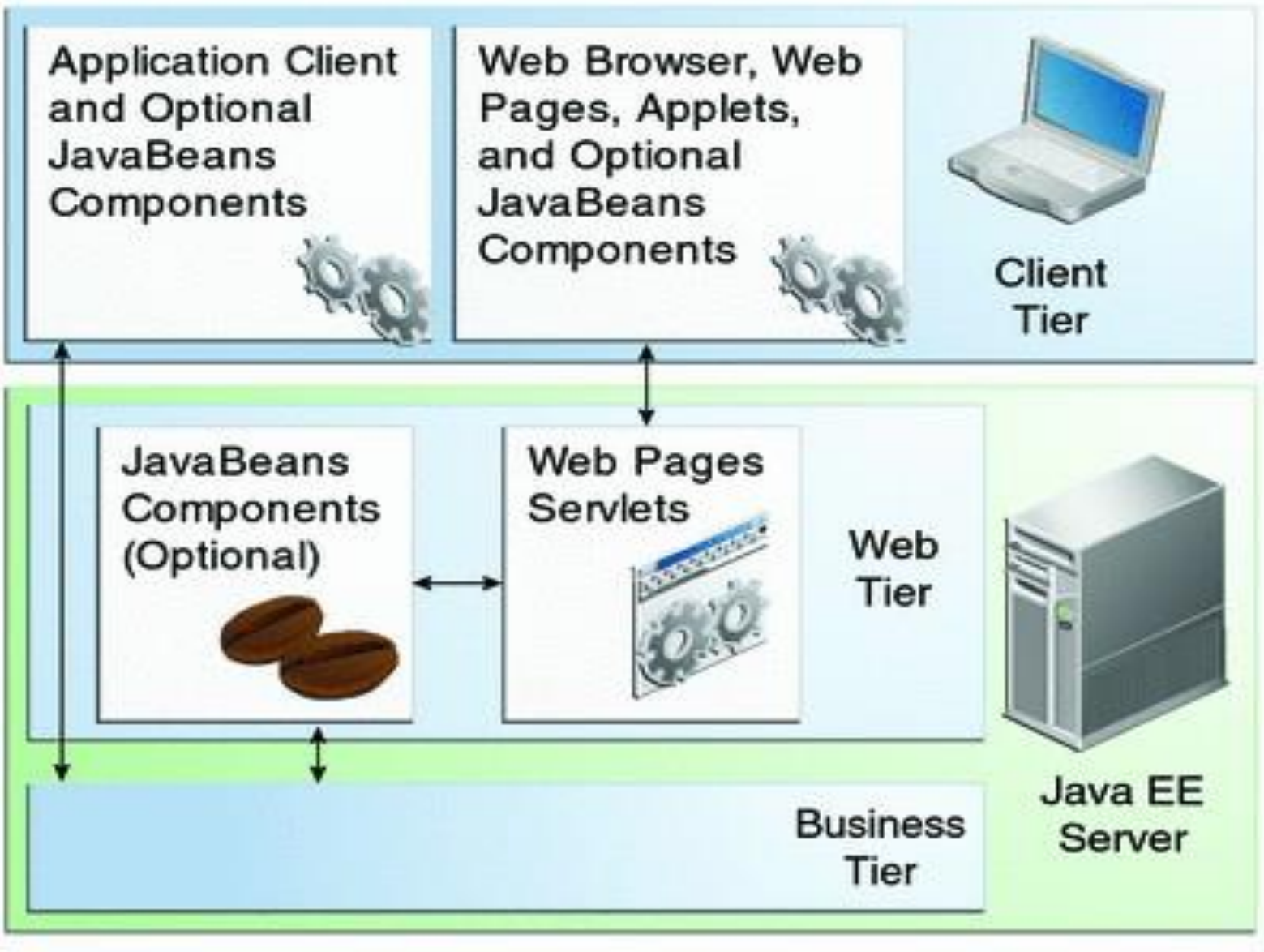
Многослойна архитектура



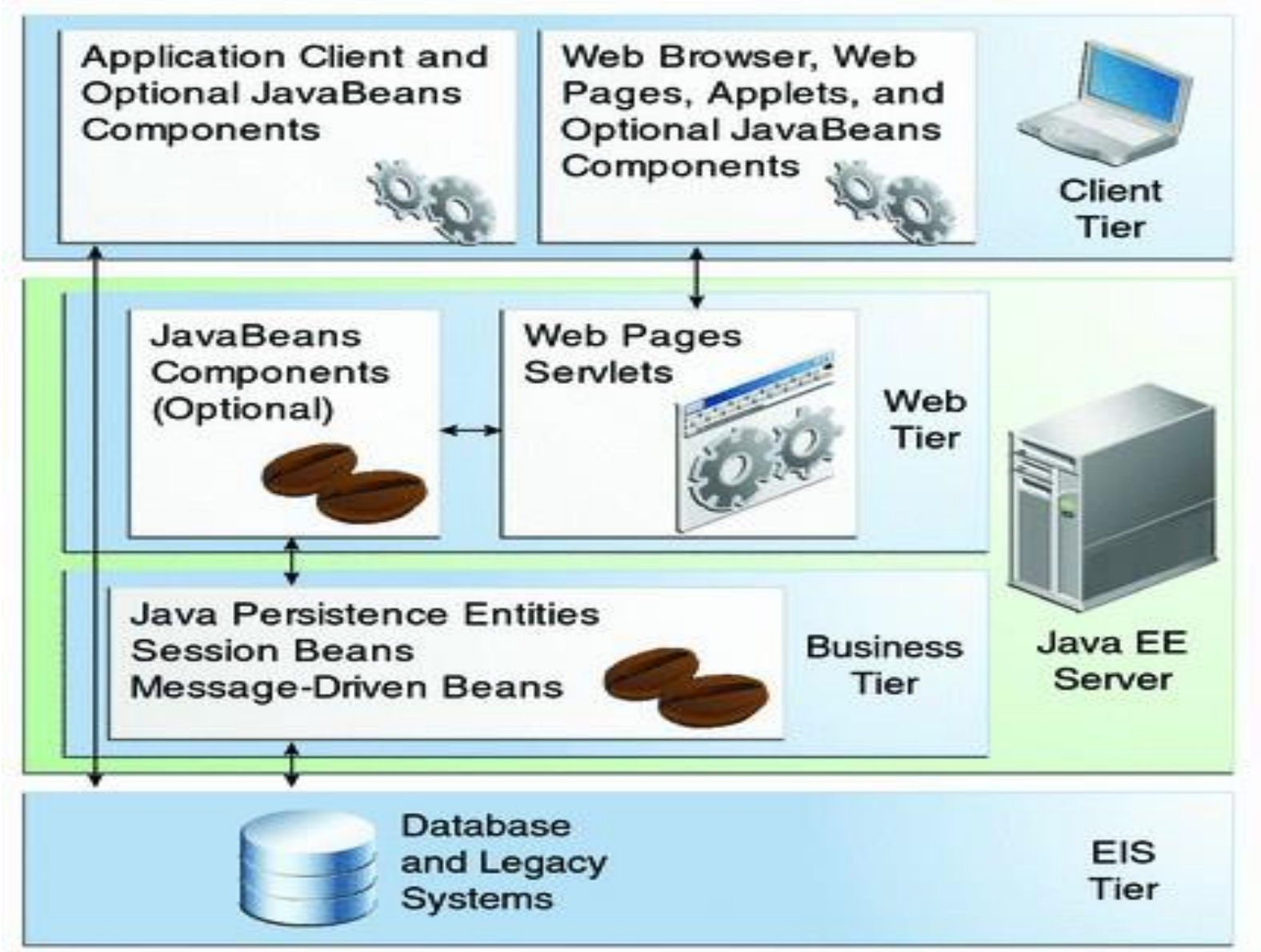
Ролята на сървъра на приложенията



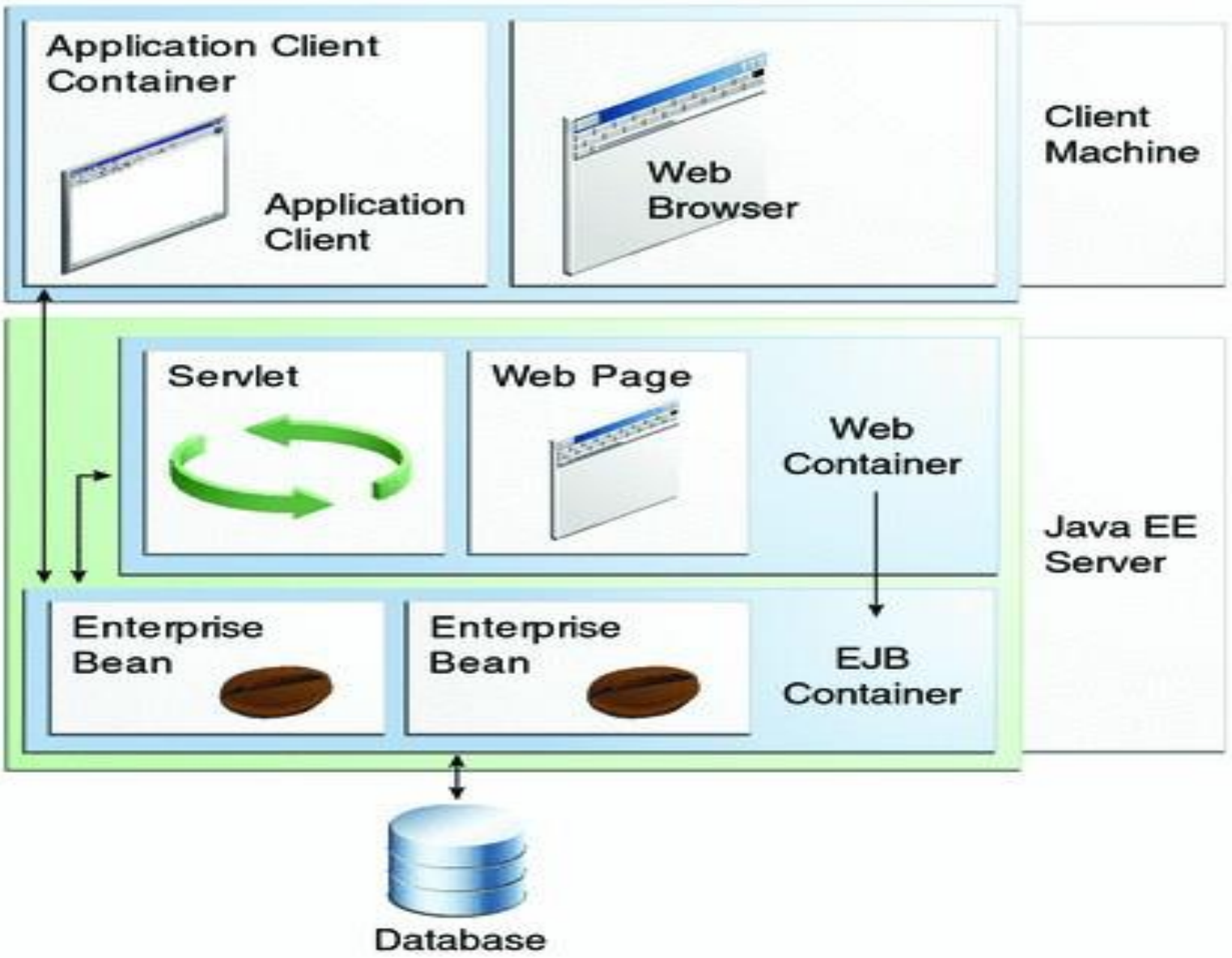
Уеб нивото и Java EE приложенията



Бизнес нивото и EIS (Enterprise Information Systems)



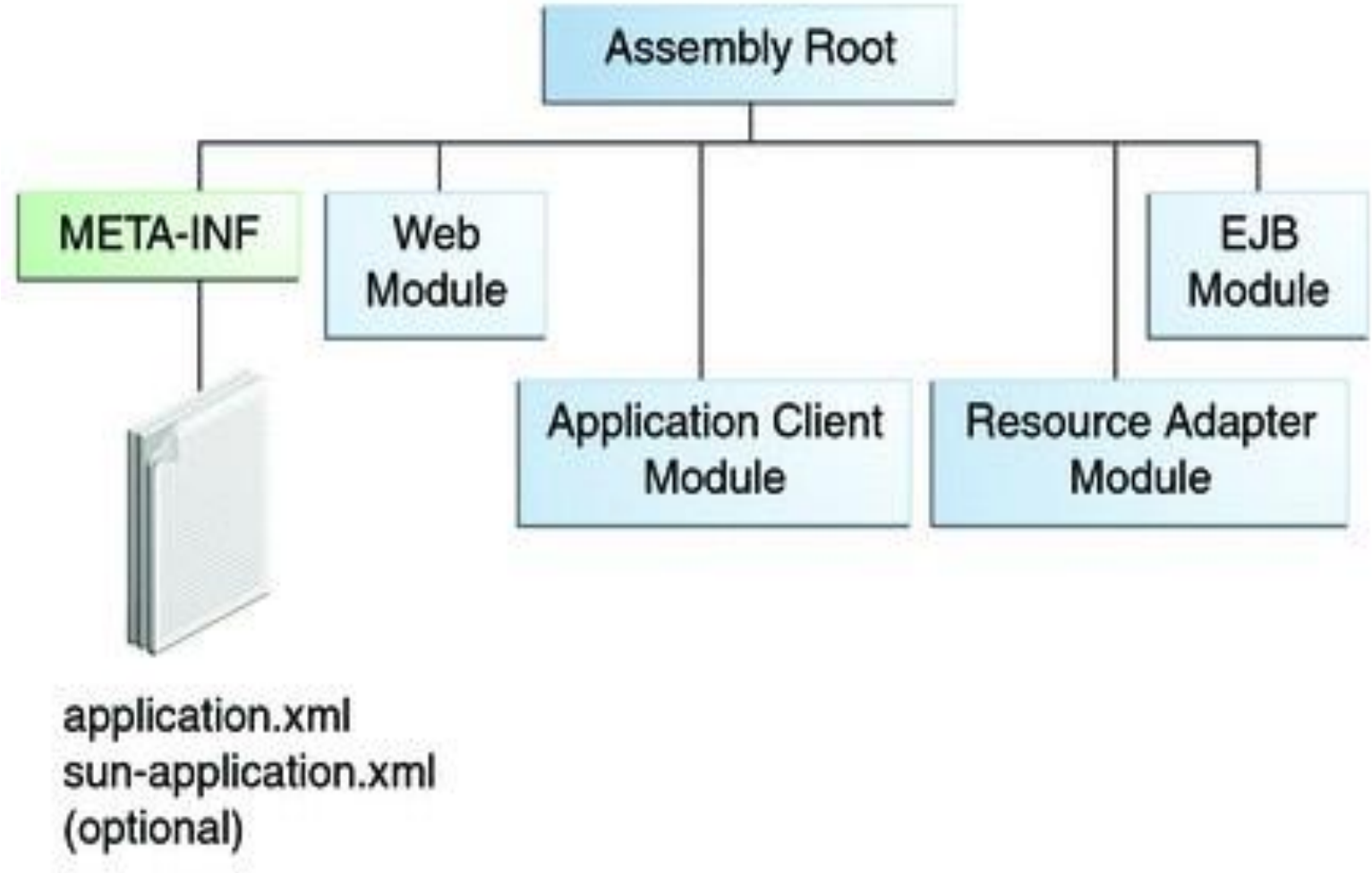
Java EE сървър и контейнерите



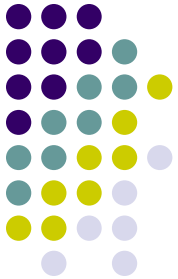
Поддръжка на уеб услугите

- XML — Extensible Markup Language
- SOAP — Simple Object Access Protocol
- WSDL — Web Service Description Language – стандарт за XML описания на уеб услуга (име, локализация, начини за комуникация)
  - – използва RPC за комуникация с клиента

Пакет на Java EE приложения

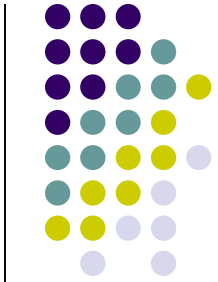


EAR архив



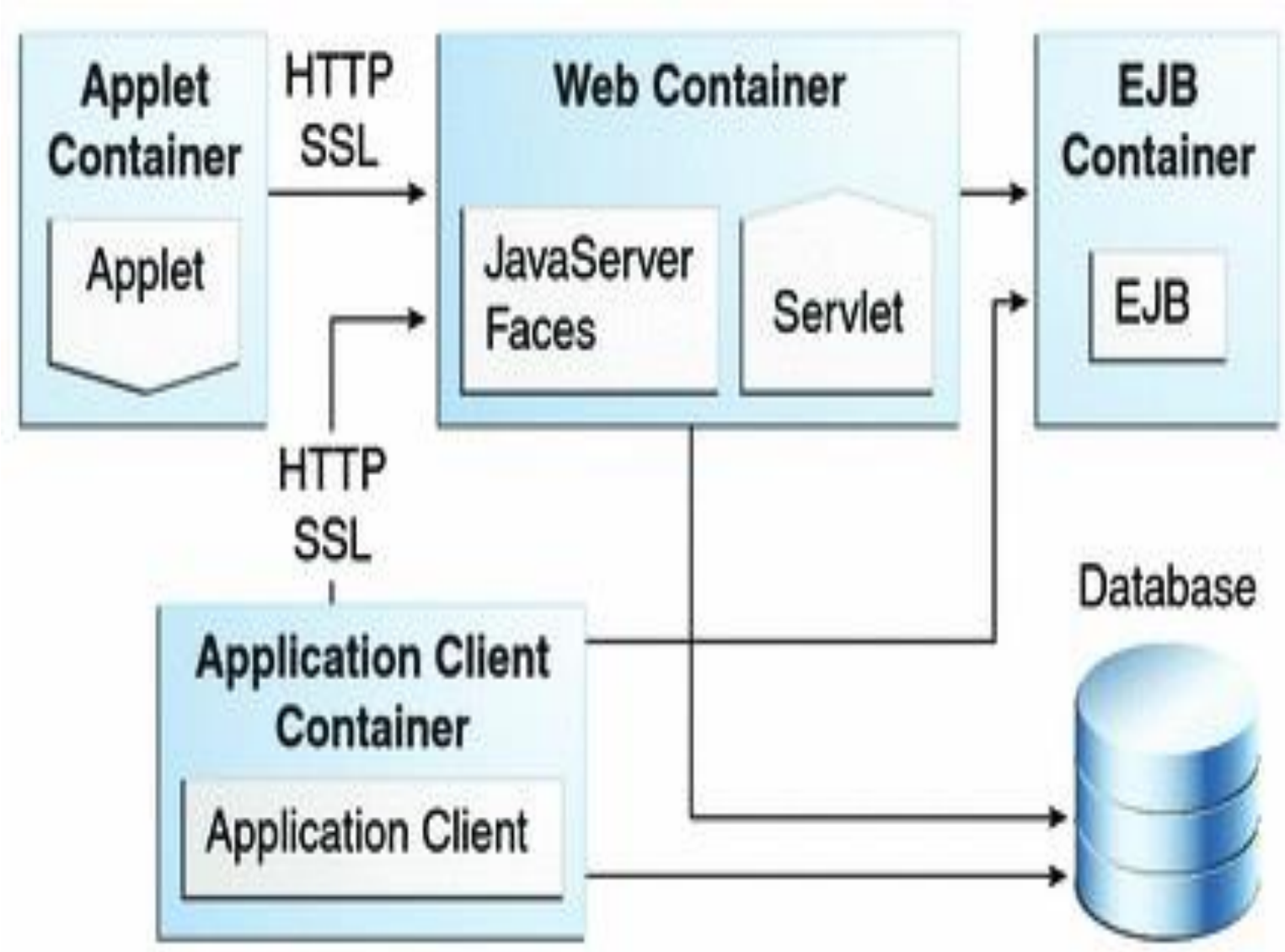
- EJB модул — класове, бинове и EJB дескриптори в .jar
- Уеб модул — сървлети, уеб файлове, необходимите класове, GIF и HTML файлове и дескриптори за разгръщане в .war
- Клиентски модул - класове, файлове и дескриптор в .jar
- Модул на адаптерите на ресурсите — класове, интерфейси, библиотеки и др. документи и дескриптор за EIS в .rar (resource adapter archive)

Ролите на разработчиците

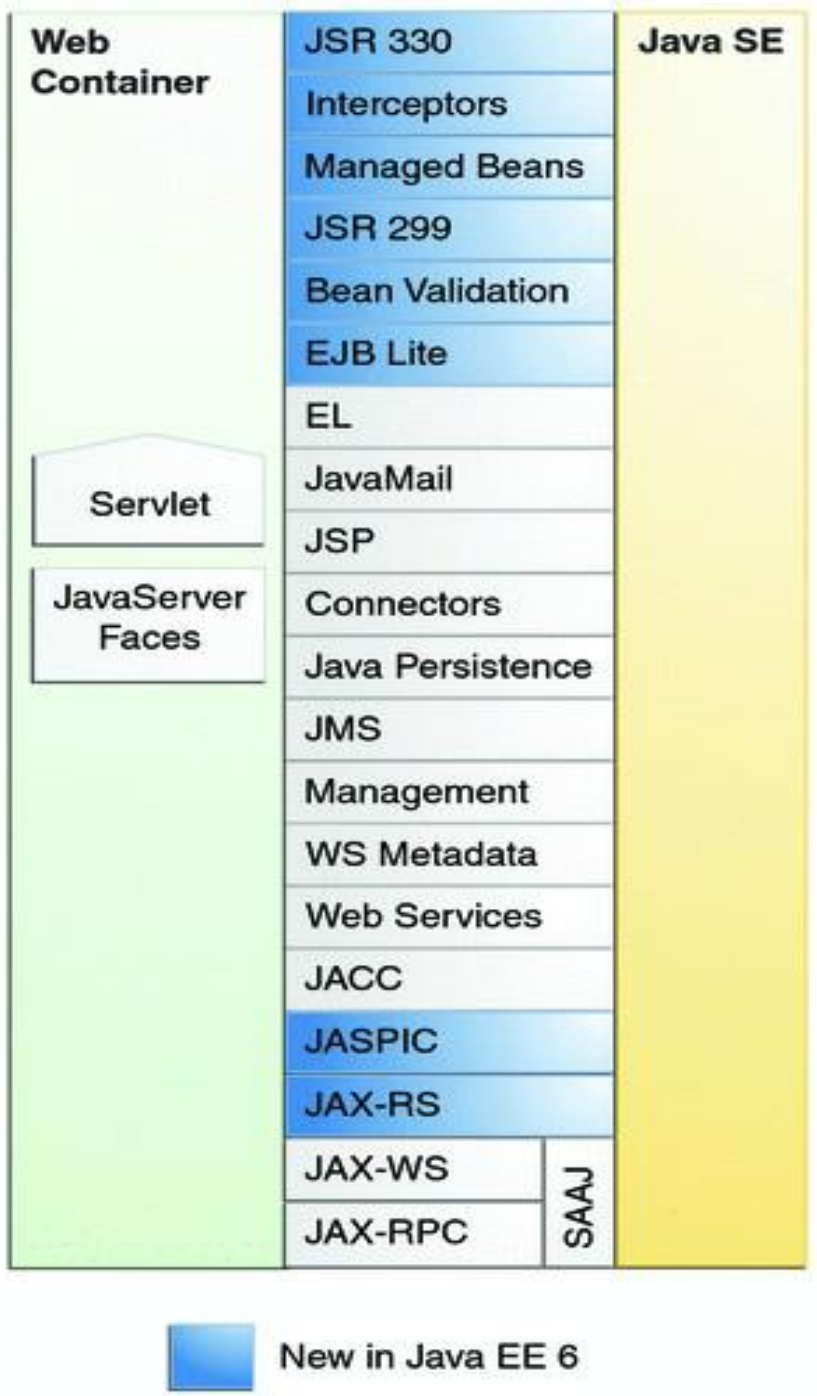


- Java EE Product Provider — доставчик на сървъра и API (например Oracle GlassFish server)
- Tool Provider — доставчик на средствата за разработване (например NetBeans IDE)
- Application Component provider — доставчик на уеб компонентите, EJB, аплетите и клиентските приложения
- Enterprise Bean Developer — разработчик на EJB (бизнес логика) и опаковане в ejb.jar
- Web component developer — създател на Servlets, JSF, JSP, уеб страници, дескриптори за war
- Application Client developer - разработчик на клиентската част за jar
- Application Assembler — настройчик на дескрипторите и създател на ear
- Application Deployer and Administrator — администратор, отговорен за разгръщането и настройването на системата на сървъра

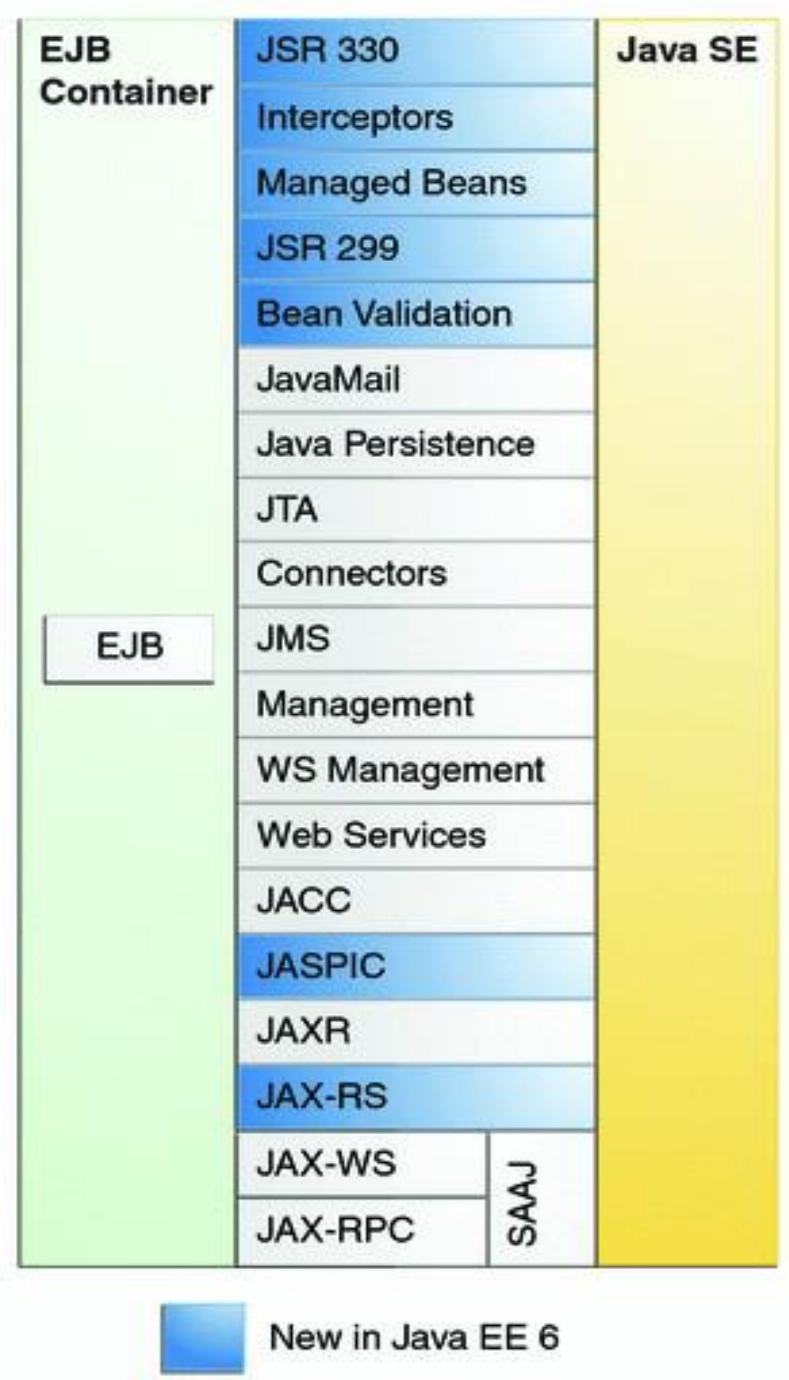
Java EE контейнери



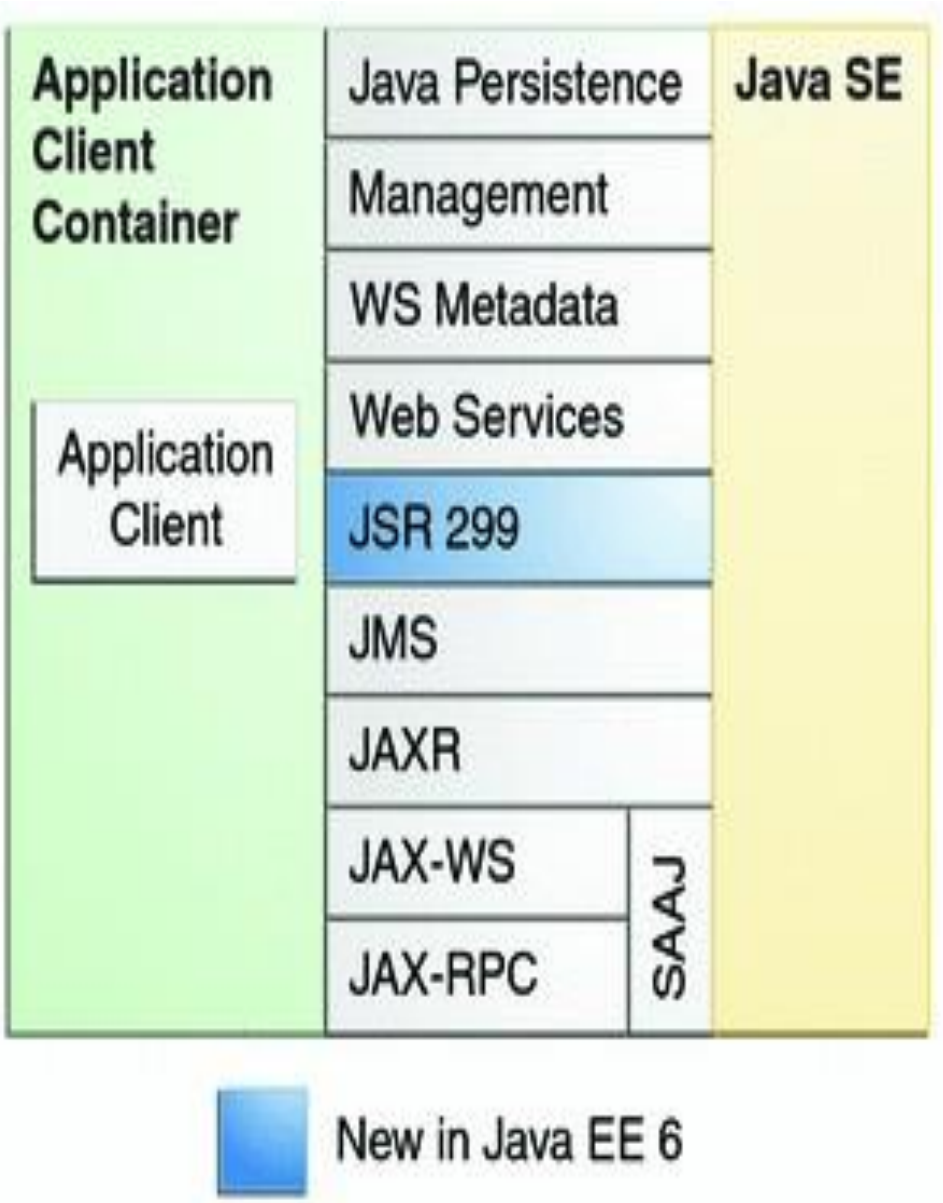
Уеб контейнер



EJB контейнер



Контейнер на клиентските приложения



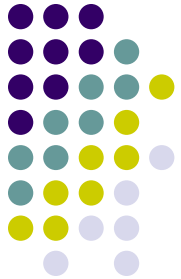
Технологии на Java EE

- EJB — Enterprise Java Beans
- Servlets
- JSF — Java Server Faces
- JSP — Java Server Pages
- JSTL — JSP Standard Tag Library
- JPA — Java Persistence API
- JTA — Java Transactions API
- JAX-RS — Java API for RESTful Web Services
- Managed Beans
- CDI — Context Dependency Injection
- Dependency Injection for Java

- Bean Validation
- Java Message Service API
- Java EE Connector Architecture
- Java Mail API
- JACC - Java Authorization Contract for containers
- JASPIC - Java Authentication Service Provider Interface for Containers
- Java EE API from Java SE 6
- JDBC API — Java Database Connectivity
- JNDI — Java Naming and Directory Interface
- Java Beans Activation Framework
- JAXP - Java API for XML Processing
- JAXB - Java Architecture for XML Binding
- SAAJ - SOAP with Attachments API for Java
- JAX-WS - Java API for XML Web Services
- JAAS - Java Authentication and Authorization Services

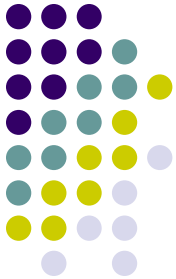
За популярността на JEE също така способства това, че Sun Microsystems предлага безплатен комплект за разработване на SDK (Software Development Kit), позволяващ на бизнеса да разработва своите системи, без да се инвестират големи суми.
В този комплект влиза сървърът за приложения GlassFish с лиценз за разработване.

GlassFish сървър



- Уеб конзола на администратора
- Asadmin
- Appclient
- Capture-schema
- Package-appclient
- Java DB
- Xjc
- Schemagen
- Wsimport
- wsgen

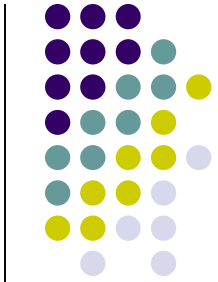
Компонентни системи



- Компонентно-ориентираният подход в проектирането и реализацията на програмните системи и комплекси е в някакъв смисъл развитие на обектно-ориентирания подход и практически е по-подходящ за разработването на крупни и разпределени програмни системи (например: корпоративни приложения).
- От гледна точка на компонентно-ориентирания подход програмната система е набор от компоненти с детайлно определени интерфейси. За разлика от другите подходи в програмирането, измененията в системата се внасят чрез създаване на нови компоненти или изменения на старите, а не чрез преработване на съществуващия код.
- Програмният компонент е автономен елемент на програмното осигуряване, предназначен за многократно използване, който може да се разпространява за използване в други програми във вид на компилиран код. Свързването с програмен компонент се реализира чрез отворени интерфейси, а взаимодействието с програмната среда се осъществява посредством събития, като в програмата, използваща компонент, може да се зададат обработчици на събития, на които да реагира компонента.

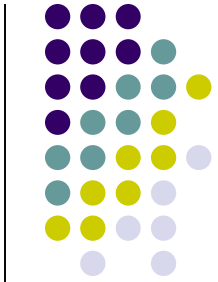
Концепцията на JavaBeans

- JavaBeans са класове на езика Java, написани по определени правила. Те се използват за обединяване на няколко обекта в един (bean) за удобно предаване на данните. JavaBeans осигурява основата за многократно използваните вграждани и модулни компоненти на програмното осигуряване.
- Спецификацията на Sun Microsystems определя JavaBeans, като «универсални програмни компоненти, които могат да се управляват с помощта на графичния интерфейс».
- Компонентите на JavaBeans могат да приемат различни форми, но най-широко те се използват в елементите на графичния потребителски интерфейс. Една от целите на създаването на JavaBeans е взаимодействието с подобни компонентни структури. Например Windows програма, при наличие на съответстващ мост или обект-обвивка, може да използва JavaBeans компонент така, както ако той се явява COM компонент или ActiveX компонент.



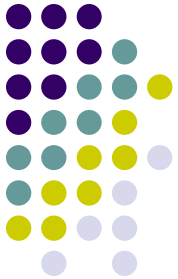
Enterprise JavaBeans (EJB) компонентът представлява в JEE приложението елемент с данни или с вътрешна, невидима за потребителя логика на приложението.
За EJB компонентите е определен жизнен цикъл в рамките на работния процес на приложението.
Това е наборът от състояния, през които преминава един екземпляр на този компонент.
EJB компонентите работят вътре в EJB контейнера, който представлява за тях компонентна среда.

Правила за описание на JavaBean



- Класът трябва да има public конструктор без параметри. Такъв конструктор позволява на инструментите да създават обект без допълнителни сложности с параметрите.
- Свойствата на класа трябва да са достъпни чрез get, set, is и други методи за достъп, които се подчиняват на стандартното споразумение за имената. Това лесно позволява на инструментите автоматично да определят и обновяват съдържанието на bean. Много инструменти имат специализирани редактори за различните типове свойства.
- Класът трябва да е сериализуем. Това позволява надеждно съхраняване, запазване и възстановяване на състоянието на bean по независим от платформата и от виртуалната машина начин.
- Класът не е длъжен да съдържа методи за обработване на събития.

Механизъм за обработване на събития



Понеже Java не поддържа указатели към функции, разработчиците на този език използват друг подход. Обработването на събитията се изпълнява с помощта на специални класове, в това число и интерфейсни.
Всяко събитие се генерира от обект-източник при изменение на вътрешното му състояние. Всички основни събития са класифицирани и имат съответстващи класове.
Родоначалник на йерархията на събитията е класа EventObject, от който са породени следните основни класове събития:

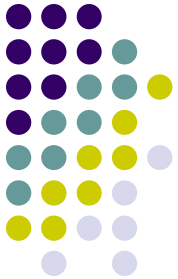
ActionEvent; ComponentEvent; ContainerEvent; KeyEvent; MouseEvent; WindowEvent; ItemEvent; TextEvent;

Enterprise JavaBeans

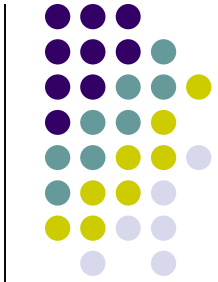
- Enterprise JavaBeans е от високо ниво, базираща се на използването на компоненти, технология за създаване на разпределени приложения, която използва на ниско ниво API за управление на транзакциите. Първият вариант на спецификацията Enterprise JavaBeans е през март 1998 г. За времето на своето съществуване технологията е преминала огромен път и продължава да се развива.
- Enterprise JavaBeans е повече от само инфраструктура. Нейното използване предполага още и технология (процес) на създаване на разпределени приложения, задава определена архитектура на приложението, а също така и определя стандартните роли на участниците в разработката.

Всеки EJB компонент се състои от:

- отдалечен интерфейс (remote-интерфейс), който определя бизнес-методите, които може да извиква EJB клиента;
- собствен интерфейс (home-интерфейс), който предоставя методите:
  - create - за създаването на нови екземпляри EJB компоненти;
  - finder - за намиране на екземпляри на EJB компоненти;
  - remove - за премахване на екземпляри на EJB.
- реализации на EJB компонента, които определят бизнес-методите, обявени в отдалечения интерфейс, и методите за създаване, премахване и търсене на собствен интерфейс.

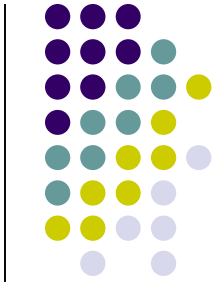


- EJB контейнерите се изпълняват под управлението на EJB сървъра, който се явява свързващо звено между контейнерите и използваната операционна среда. EJB сървърът осигурява достъп на EJB контейнерите до системните услуги, такива като управление на достъпа до базите от данни или мониторинг на транзакциите. Всички екземпляри на EJB компонентите се изпълняват под управлението на EJB контейнера, който им предоставя разположените в него компоненти и управлява техният жизнен цикъл.
- След приключването на разработването, наборите от EJB компоненти се разполагат в специални файлове (архиви, jar), за един или повече компоненти, заедно със специални deployment параметри. След което те се установяват в специална операционна среда, в която се зарежда EJB контейнера.
- Клиентът търси компоненти в контейнера с помощта на home-интерфейса на съответстващия компонент.
- След като компонентът е създаден и/или намерен, клиентът изпълнява обръщение към неговите методи с помощта на remote-интерфейса.



В общия случай контейнерът е предназначен за решаването на следните задачи:

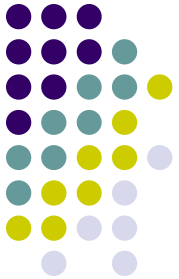
- *Осигуряване на безопасност*
- *Осигуряване на отдалечени извиквания*
- *Управление на жизнения цикъл*
- *Управление на транзакциите*



Съществуват три типа EJB компоненти:

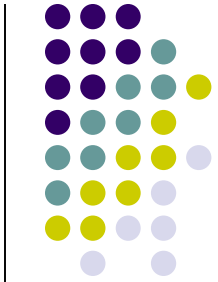
- сесийни (Session Beans);
  - stateless (без състояние);
  - stateful (с поддръжка на текущо състояние на сесията);
  - singleton (един обект за всички приложения);
- същностни (Entity Beans);
- управляеми от съобщения (Message Driven Beans).

Съставни части на EJB компонент



- Enterprise Bean
- Домашен интерфейс
- Отдалечен интерфейс
- Описател на разгръщането (XML файл)
- EJB-Jar файл

Enterprise JavaBean инфраструктура



- EJB инфраструктурата осигурява отдалечено взаимодействие на обектите, управление на транзакциите и безопасност на приложението. Спецификацията EJB определя изискванията към елементите на инфраструктурата и определя Java API, като тя не засяга въпросите за избора на платформата, протоколите и други аспекти, свързани с реализацията.
- В общия случай е необходимо да се гарантира съхраняването на състоянията на компонентите в контейнерите. EJB инфраструктурата е длъжна да предостави възможности за интеграция на приложението със съществуващите системи и приложения. Всички аспекти на взаимодействието на клиентите със сървърните компоненти трябва да са реализирани в контекста на транзакциите, управлението на които се възлага на EJB инфраструктурата.

JEE е изключително хармонична, удобна и полезна.
Приела е всички основни стандарти на CORBA и XML.
JEE позволява съществено да се опрости труда на системните архитектори, проектанти и разработчици, като предлага ясна и гъвкава архитектура и набор от взаимно свързани стандарти за използване на най-важните системни услуги.
Enterprise JavaBeans спецификацията е съществена крачка към стандартизацията на модела на разпределените обекти в Java.