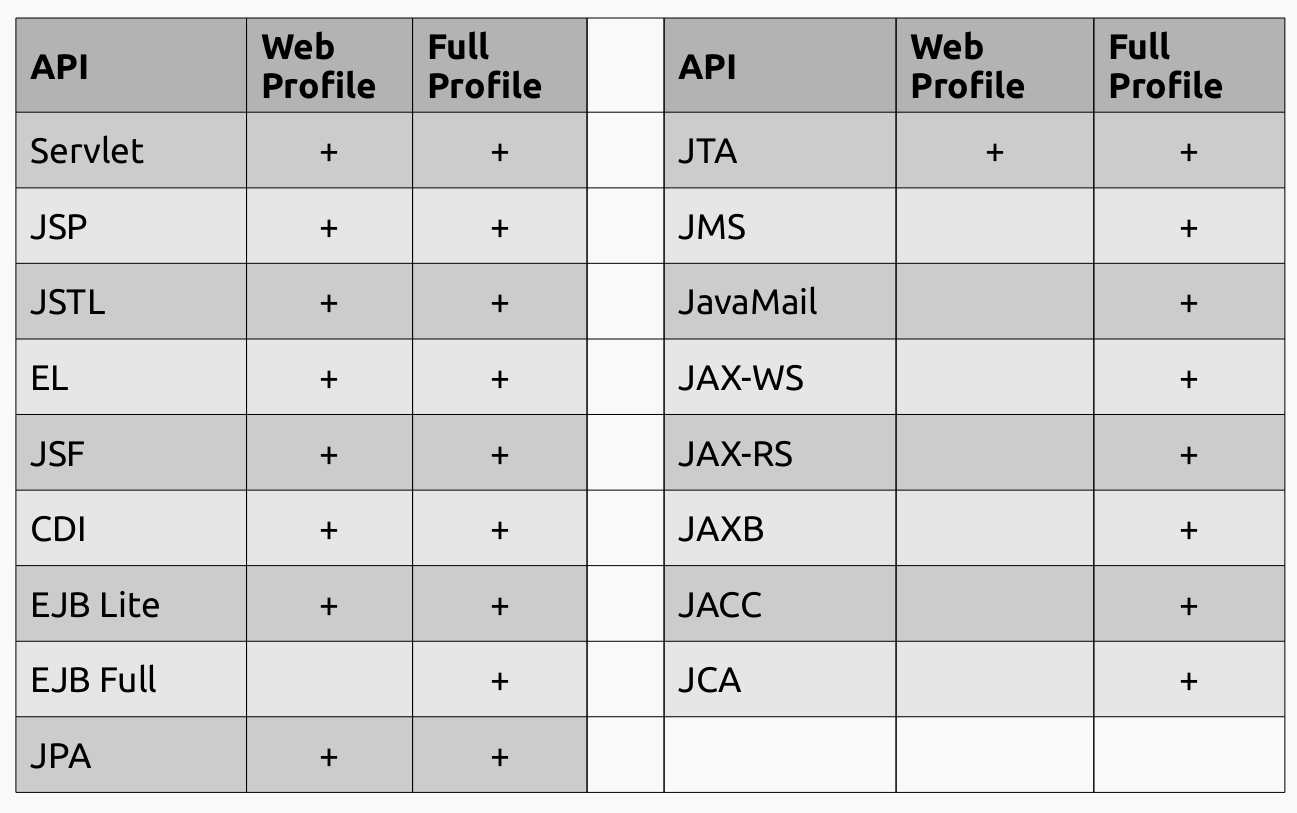
<https://github.com/band-of-four/cheatsheets/blob/master/Internet-Applications-Development/Lab4.md>

<https://github.com/RedGry/__pip__/tree/master/l4>

1) Платформа Java EE. Спецификации и их реализации.

<https://javarush.ru/groups/posts/2637-vvedenie-v-java-ee>

Конкретные реализации-это так называемые серверы приложений: GlassFish, WildFly, TomEE, IBM WebSphere и т.д. Существуют также контейнеры сервлетов, которые реализуют только часть JSP/сервлета огромного Java EE API , например Tomcat , Jetty и т. Д.



Изображение выглядит как стол

Автоматически созданное описание

javaProfiles JAXB - <https://www.codeflow.site/ru/article/jaxb>

JACC - спецификация, определяющая взаимодествие между сервером приложений Java EE и сервисом авторизации.

JCA - <https://habr.com/ru/post/251131/>

Enterprise JavaBeans (также часто употребляется в виде аббревиатуры EJB) — спецификация технологии написания и поддержки серверных компонентов, содержащих бизнес-логику. Является частью Java EE.

<https://javarush.ru/groups/posts/2648-znakomstvo-s-ejb>

<https://javarush.ru/groups/posts/2650-ispoljhzovanie-jndi-v-java>

Message Driven Beans, или MDB, или бины управляемые сообщениями, подобно сеансовым бинам реализуют некоторую бизнес-логику. Но в отличие от своих родственников, у MDB есть одно важное отличие. Клиенты никогда не вызывают методы MDB напрямую. Такие бины чаще всего выступают в роли слушателей JMS (Java Message Service) сообщений и служат для организации асинхронного обмена сообщениями между частями системы.

Итак, служба имен и каталогов — это система, которая управляет отображением множества имен во множестве объектов. В нашей файловой системе мы взаимодействуем с именами файлов, за которыми скрываются объекты — сами файлы в различных форматах.

JNDI, или же Java Naming and Directory Interface, представляет собой Java API для доступа к службам имен и каталогов.

В JNDI все операции с именами и каталогами выполняются относительно контекста. Абсолютных корней нет. Поэтому JNDI определяет InitialContext, который обеспечивает отправную точку для именования и операций с каталогами. После получения доступа к начальному контексту, его можно использовать для поиска объектов и других контекстов.

Name objectName = new CompositeName("java:comp/env/jdbc");

InitialContext — это класс, который представляет из себя корневой элемент JNDI tree и реализует интерфейс Context.

Java Transaction API (JTA) — API, для поддержки транзакций, входящее в стандарт серверной платформы для языка программирования Java – Jakarta EE. Определяет взаимодействие между менеджером транзакций и другими участниками распределённой транзакционной системы.

<https://www.baeldung.com/jee-jta>

<http://java-online.ru/javax-jms.xhtml>

Свободно распространяемая библиотека JavaMail предназначена для подключения к почтовым серверам с целью получения и отправки электронной почты с использованием протоколов SMTP, POP3 и IMAP. Библиотека позволяет отправлять сообщения с вложениями и удалять их на почтовом сервере.

<http://java-online.ru/javax-mail.xhtml>

Java API for XML Web Services (JAX-WS) is a standardized API for creating and consuming SOAP (Simple Object Access Protocol) web services.

<https://www.baeldung.com/jax-ws>

<https://ru.wikipedia.org/wiki/JAX-RS>

Java Cryptography Architecture — это криптографический фреймворк, входящий в состав платформы Java. JCA предоставляет API для различных целей, в том числе шифрования, управления ключами, генерации криптографически стойких псевдослучайных чисел и т. д.

<https://javarush.ru/groups/posts/2277-java-cryptography-architecture--pervoe-znakomstvo>

2) Принципы IoC, CDI и Location Transpanency. Компоненты и контейнеры.

Inversion of Control – инверсия управления. Существует какой-либо фреймворк, который осуществляет вызов каких-либо методов,

<https://habr.com/ru/post/136140/>

<https://www.baeldung.com/java-ee-cdi>

Изображение выглядит как текст

Автоматически созданное описание

JNDI

<https://coderlessons.com/articles/java/di-cdi-osnovy>

3) Управление жизненным циклом компонентов. Дескрипторы развёртывания.

beans.xml

PostConstruct, PreDestroy ... (вызывается только один раз)

4) Java EE API. Виды компонентов. Профили платформы Java EE.

Изображение выглядит как текст, электроника, калькулятор

Автоматически созданное описание

<https://javaee.github.io/tutorial/overview008.html>

В мастере создания проектов в качестве версии Java EE можно выбрать Java EE 6 Web или Java EE 7 Web. Java EE 6 Web и Java EE 7 Web являются упрощенными профилями Java EE 6, которые содержат подмножество полной платформы Java EE. Профили Java EE Web разработаны для веб-приложений, не требующих расширенных технологий Java EE, таких как поддержка удаленных интерфейсов, полная спецификация EJB 3.1 и API-интерфейс службы передачи сообщений Java (JMS).

Веб-профили поддерживают обработку транзакций и управление сохранением состояния, которые часто используются в корпоративных веб-приложениях. Профили Java EE Web можно использовать для веб-приложений, в которых применяются компоненты сеансов с локальным интерфейсом или без интерфейса. При использовании в приложении удаленного интерфейса необходим полный профиль Java EE.

5) Компоненты EJB. Stateless & Stateful Session Beans. EJB Lite и EJB Full.

и так далее.

6) Работа с электронной почтой в Java EE. JavaMail API.

7) JMS. Реализация очередей сообщений. Способы доставки сообщений до клиента. Message-Driven Beans.

8) Понятие транзакции. Управление транзакциями в Java EE. JTA.

9) Веб-сервисы. Технологии JAX-RS и JAX-WS.

10) Платформа Spring. Сходства и отличия с Java EE.

11) Модули Spring. Архитектура Spring Runtime. Spring Security и Spring Data.

12) Реализация IoC и CDI в Spring. Сходства и отличия с Java EE.

13) Реализация REST API в Java EE и Spring.

14) React JS. Архитектура и основные принципы разработки приложений.

15) Компоненты React. State & props. "Умные" и "глупые" компоненты.

16) Разметка страниц в React-приложениях. JSX.

17) Навигация в React-приложениях. ReactRouter.

18) Управление состоянием интерфейса. Redux.

19) Angular: архитектура и основные принципы разработки приложений.

20) Angular: модули, компоненты, сервисы и DI.

21) Angular: шаблоны страниц, жизненный цикл компонентов, подключение CSS.

22) Angular: клиент-серверное взаимодействие, создание, отправка и валидация данных форм.