\*\*WAR (Web Application Archive)\*\* и \*\*JAR (Java Archive)\*\* - это архивные форматы, используемые в мире Java для упаковки и распространения приложений и библиотек. Каждый из этих форматов имеет свои особенности и предназначение:

1. \*\*WAR (Web Application Archive)\*\*:

- Формат WAR используется для упаковки веб-приложений, предназначенных для развертывания на веб-серверах.

- WAR-файл содержит в себе веб-компоненты, такие как сервлеты, JSP-страницы, HTML, CSS, JavaScript и другие ресурсы, необходимые для работы веб-приложения.

- WAR-файлы обычно развертываются на серверах приложений, таких как Apache Tomcat, WildFly, и служат для обслуживания веб-приложений.

2. \*\*JAR (Java Archive)\*\*:

- Формат JAR используется для упаковки и распространения Java-библиотек (библиотек классов и ресурсов), а также самостоятельных приложений.

- JAR-файлы содержат Java-байт-код, ресурсы, манифест (manifest), который содержит метаданные о JAR-файле, и могут содержать файлы манифеста.

- JAR-файлы используются для организации и упаковки библиотек и приложений так, чтобы они могли быть легко переносимыми и повторно используемыми на разных платформах и в разных проектах.

Интересный факт: WAR-файлы на самом деле являются специфическими JAR-файлами с определенной структурой каталогов и содержат специфические файлы манифеста, которые указывают на точку входа в веб-приложение.

Таким образом, WAR-файлы предназначены для веб-приложений и их развертывания на серверах, в то время как JAR-файлы используются для упаковки и распространения Java-кода и библиотек, которые могут быть использованы другими Java-приложениями.

Структура выходного проекта Maven (обычно называемая "стандартной структурой каталогов Maven") имеет следующий общий формат. Обратите внимание, что Maven обеспечивает конфигурацию и соглашения по умолчанию, но вы можете настроить структуру под свои нужды, если это необходимо.

```

my-maven-project/

├── src/

│ ├── main/

│ │ ├── java/ # Исходные Java-коды приложения

│ │ ├── resources/ # Ресурсы, такие как конфигурационные файлы

│ │ └── webapp/ # Каталог для веб-ресурсов (HTML, JSP и другие)

│ └── test/

│ ├── java/ # Исходные Java-коды для тестирования

│ └── resources/ # Ресурсы для тестов

├── target/ # Сюда будут помещены результаты сборки

├── pom.xml # Файл Project Object Model, описывает проект

└── ... # Другие файлы и каталоги, включаются по мере необходимости

```

Краткое описание каждой из директорий и файлов:

- `src/main/java`: Здесь находятся исходные Java-коды вашего приложения.

- `src/main/resources`: Этот каталог содержит ресурсы, такие как конфигурационные файлы, изображения и другие необходимые файлы.

- `src/main/webapp`: В этом каталоге обычно находятся веб-ресурсы, такие как HTML-страницы, JSP, CSS, JavaScript и другие файлы, связанные с веб-приложением.

- `src/test/java`: Здесь находятся исходные Java-коды для unit-тестирования вашего приложения.

- `src/test/resources`: Каталог с ресурсами, необходимыми для тестирования.

- `target`: Этот каталог создается Maven и содержит результаты сборки, включая скомпилированные классы, библиотеки и дистрибутивы вашего приложения.

- `pom.xml`: Это основной файл конфигурации Maven, который описывает зависимости, цели сборки и другие параметры вашего проекта.

Следуя этой структуре, вы сможете управлять и организовывать ваш Maven-проект согласно соглашениям Maven, что облегчит сборку и разработку вашего приложения.

Структура Tomcat

В структуре каталогов Tomcat папка `lib` предназначена для хранения библиотек (JAR-файлов), которые являются частью самого Tomcat или используются для его работы. Эти библиотеки обеспечивают функциональность сервера приложений и общие ресурсы, которые могут быть использованы веб-приложениями, развернутыми на Tomcat.

Вот некоторые из библиотек и JAR-файлов, которые обычно находятся в каталоге `lib` Tomcat:

1. \*\*servlet-api.jar\*\*: Этот JAR-файл содержит классы и интерфейсы, определенные в спецификации Java Servlet API. Он необходим для выполнения сервлетов на Tomcat.

2. \*\*jsp-api.jar\*\*: Этот JAR-файл содержит классы и интерфейсы, определенные в спецификации JavaServer Pages (JSP) API. Он необходим для выполнения JSP-страниц на Tomcat.

3. \*\*catalina.jar\*\*: Этот JAR-файл содержит классы, отвечающие за ядро Tomcat, включая обработку HTTP-запросов и управление жизненным циклом сервлетов.

4. \*\*jasper.jar\*\*: Этот JAR-файл отвечает за компиляцию JSP-страниц в Java-код и их выполнение.

5. \*\*commons-logging.jar\*\*: Библиотека для ведения журнала и логирования событий.

6. \*\*tomcat-dbcp.jar\*\*: Библиотека для поддержки пула соединений с базой данных (DBCP) на Tomcat.

7. \*\*websocket-api.jar\*\*: Содержит классы, связанные с веб-сокетами, которые позволяют веб-приложениям взаимодействовать в режиме реального времени.

Эти библиотеки предоставляют базовую инфраструктуру и функциональность для работы веб-приложений на Tomcat. Ваше веб-приложение может зависеть от некоторых из этих библиотек, и Tomcat автоматически предоставляет доступ к ним, чтобы обеспечить выполнение вашего приложения.