1. **What build tools do you know?**

**CookieCutter** A classic Python project directory structure

**PyScaffold** is a tool for creating scaffolding for Python projects

The build tool **PyBuilder,** which creates a directory structure very close to Maven

**Poetry**, feels like it's a more mature, project-active Python build with more powerful trust management capabilities

**2.What Java-specific tools do you know?**

**Apache Maven** – An automation tool primarily used to build Java projects

**Hudson** – A continuous integration (CI) tool written in Java

**Jenkins** – An open-source continuous integration tool written in Java. The project was a branch from Hudson after an argument with Oracle.

**Gradle** – An open-source automated build system that builds on the Apache Ant and Maven Apache concepts and introduces a Domain Specific Language (DSL) based on Groovy, rather than configuring XML forms using the project announced by Apache Maven.

**Apache Ant** – A software tool for automating the software build process, originated from the Apache Tomcat project in the early 2000s.

**SBT** – An open-source build tool for Scala and Java projects, similar to Java's Maven and Ant. **Atlassian Bamboo** – A continuous integration and delivery tool that bundles automated build, test, and release into a single process.

**TeamCity** – A Java-built management and continuous integration server from JetBrains. **Grape** – Embedded in Groovy's JAR dependency manager.

**Ivy** – A subproject of the Apache Ant project, a transitive dependency manager.

**Leiningen** – an automated build and dependency management tool for software projects written in the Clojure programming language.

**3. Main differences between Java specific build tools.**

Gradle is one of the core tools recommended in the Java Developer Guide, many open-source projects are built with Gradle, thanks to Gradle's flexible syntax, developers can flexibly arrange the build task according to their actual needs, but Gradle is not the only option for project construction. Maven is the most popular project building tool after Ant, which solves Ant's complex build methods, improves project build efficiency, and integrates functions such as compilation and deployment.

**4. Maven dependency scopes and their meanings.**

Maven's life cycle exists to compile, test, run these processes, then obviously some dependencies are only used for testing, such as junit; some dependencies are not used for compilation, only when running, such as mysql driver packages are not used at the compilation period (the compilation period uses the JDBC interface), but used at runtime; there are also some dependencies, the compilation period is used, and the runtime does not need to be provided, because some containers have been provided. For example, servlet-api is already available in tomcat, we only need to provide it at compile time. In summary, in POM 4, <dependency>it was also introduced<scope>, which mainly manages the deployment of dependencies. There are roughly several such as composite, provided, runtime, test, system, etc.

**5.Maven scope ‘system’. When do we use it?**

system: Valid at compile, test time, but not at runtime. The difference with provided is that when using system-scoped dependencies, the path to the dependent file must be explicitly specified through the systemPath element. Since such dependencies are not resolved through the Maven repository and are often tied to native systems, potentially making the build unmanageable, it should be used with caution. The systemPath element can reference environment variables.

**6. What is the difference between remote and local repositories? What is maven .m2 repository? How to install artifact into local repo? How to deploy artifacts into remove repository?**

Maven's local repository The local repository in maven stores the dependencies of all projects, and every time a new maven project is created, all related project files are stored in the local repository. The location of maven's default local repository is in a .m2 folder.

Maven's Central Repository When you create a new maven project, a pom .xml configuration file is generated, and then maven checks your pom .xml configuration file to see which dependencies need to be loaded. The first is to look up from your local repository, and if you don't find a dependency, switch to maven's central repository to find it.

Remote repositories In maven, if the dependencies you need are not in the local repository or in the central repository, then maven will report an error. But we can load the remote repository with configuration.

To deploy jar to an external repository, you must configure the repository URL in the pom .xml and the authentication information to connect to the repository in the settings .xml.

**7. Maven goals and phases. What phases do you know?**

Maven is based around the central concept of a build lifecycle. What this means is that the process for building and distributing a particular artifact (project) is clearly defined.

For the person building a project, this means that it is only necessary to learn a small set of commands to build any Maven project, and the [POM](https://maven.apache.org/guides/introduction/introduction-to-the-pom.html) will ensure they get the results they desired. There are three built-in build lifecycles: default, clean and site. The default lifecycle handles your project deployment, the clean lifecycle handles project cleaning, while the site lifecycle handles the creation of your project's web site.

**8. Analogs of maven dependency scopes in gradle.**

| **Maven** | **Gradle** | **说明** |
| --- | --- | --- |
| **compile** | **implementation、api** | **The former applies to all projects that apply Java plugins while apis**  **only applies java library plugins. In most cases, using simplification is sufficient,**  **especially when building applications or webapps** |
| **test** | **testImplementation、testRuntimeOnly** | **Gradle distinguishes between those dependencies needed to compile your project's tests**  **and those that are needed to run them only.**  **testImplementation dependencies required for test compilation,**  **testRuntimeOnly to run tests** |
| **provided** | **compileOnly** | **Note that the War plugin adds the supportedCompile**  **and providedRuntime dependency configurations.**  **compileOnly behaves slightly differently from war files and only ensures that**  **these dependencies are not packaged** |
| **import** | **testImplementation、implementation** | **Maven allows you to share dependency constraint poms by defining**  **dependencies within part of a POM file of packaging type.**  **This special type of POM can then be imported into other POM**  **in order to have a consistent version of the library in the project.**  **Gradle can use such BOMs for the same purpose**  **through special dependency syntax based on platform() and forcedPlatform() methods.** |

**Ши Веньцзяо(вера)**