

JEE - Prepare and deliver the source code

- Information about discovery
 - Discovery when using AIP Console
- Source code delivery using CAST AIP Console
 - Prepare the application source code
 - Maven based source code
 - No .pom or .project file
 - Behaviour in Console 2.3
 - Behaviour in Console 2.2
- Using legacy CAST Delivery Manager Tool
 - How do I add a source code package to my delivery
 - What should you package?
 - Exclusions
 - How do I package the Version?
 - How do I validate and fine-tune my Version?
 - How do I deliver the Version for analysis?
 - Delivery acceptance

 **Summary:** This section describes how to prepare and deliver the source code of your JEE application.

Information about discovery

Discovery is a process that is actioned during the delivery process. CAST will attempt to automatically identify "projects" within your application using a set of predefined rules. This discovery process also allows CAST AIP to set the initial analysis configuration settings explained in [JEE - Analysis configuration](#). Discoverers are:

- either **embedded in CAST AIP core**:
 - [Eclipse Project Discoverer](#)
 - [Maven Project Discoverer](#)
 - [Eclipse no nature Project Discoverer](#)
 - [Web JSP Discoverer](#)
- or are provided as **installable extensions**:
 - [JEE Manifest Discoverer](#)
 - [DMT extractor for Maven remote repository on HTTP or HTTPS](#)
 - [Gradle Project Discoverer](#)
 - [JEE Netbeans Discoverer](#)
 - [JEE Maven Build Extractor](#)
 - [JEE File Discoverer](#)

You should read the relevant documentation for each discoverer (provided in the link above) to understand how the source code will be handled.

Discovery when using AIP Console

AIP Console will **automatically install discoverers** based on the presence of **specific source code files** in the delivered source code, these are as follows:

- [Eclipse Project Discoverer](#) >>> if the project contains **.project files "with Java nature"**
- [Maven Project Discoverer](#) >>> if the project contains **.pom files** (except when the type of packaging declared in the pom.xml is "pom")
- [Gradle Project Discoverer](#)>>> if the project contains **.gradle** or **.gradle.kts** files
- [JEE File Discoverer](#) >>> if the project contains **.java** files (only in [Console 2.3](#))

Source code delivery using CAST AIP Console

 See [Application onboarding](#) and [Application onboarding - prerequisites](#) for more detailed information about the steps you should take to deliver your source code.

Prepare the application source code

AIP Console expects either a **ZIP/archive file** or **source code located in a folder** configured in AIP Console. You should include in the ZIP/source code folder all JEE source code, including JAR files if necessary. CAST highly recommends placing the files in a folder dedicated to JEE and using sub-folders where necessary. If you are using a ZIP/archive file, zip the folders in the "temp" folder - but do not zip the "temp" folder itself, nor create any intermediary folders:

```
D:\temp
|-----JEE-Java
|-----OtherTechno1
|-----OtherTechno2
```

Maven based source code

When **adding a new version** to analyze an Application that includes **Maven based source code**, you have several choices with regard to specifying where the required Maven repositories are located. The location of the repository is crucial to ensure that any associated JAR files can be automatically discovered and that POM dependencies can also be located. You can do as follows:

- You can include the Maven repository when you deliver the source code (i.e. in the ZIP or in the designated source code folder). Place the contents of the Maven repository (using the same file structure) at the root of the ZIP, for example:

```
D:\temp
|-----JEE-Java
|-----MavenRepo
|-----OtherTechno1
```

- You can define a local Maven repository for the target AIP Node
- You can define a remote HTTP Maven repository for the target AIP Node.

AIP Console will also use the **above order to prioritise the various repositories**. In other words, if you include a repository in the ZIP or in the designated source code folder this will be used instead of any local or remote repositories that have been defined.

See [Configuring source code delivery for Maven](#) for more information.

No .pom or .project file

When **adding a new version** to analyze an Application that includes **JEE based source code** and this source code **does not contain** a **.pom** or **.project file** Console will behave as follows (see also [Configuring source code delivery for JEE without .pom file or .project file](#) or [Configuring source code delivery for JEE without .pom file or .project file for CAST Imaging](#) for more information):

Behaviour in Console 2.3

Console will automatically install the **JEE File Discoverer** whenever a **.java file** is encountered in your delivered source code. This extension will ensure that the required Analysis Units are created for your source code and that an analysis can proceed without issue.

Behaviour in Console 2.2

Console is unable to "discover" this code as JEE (Console relies on the presence of the .pom / .project files). In this situation, the source code delivery will end in failure. In order to resolve this issue, you can manually configure Console to install an extension called the **JEE File Discoverer** whenever a **.java file** is encountered in your delivered source code. This extension will ensure that the required Analysis Units are created for your source code and that an analysis can proceed without issue.

Using legacy CAST Delivery Manager Tool

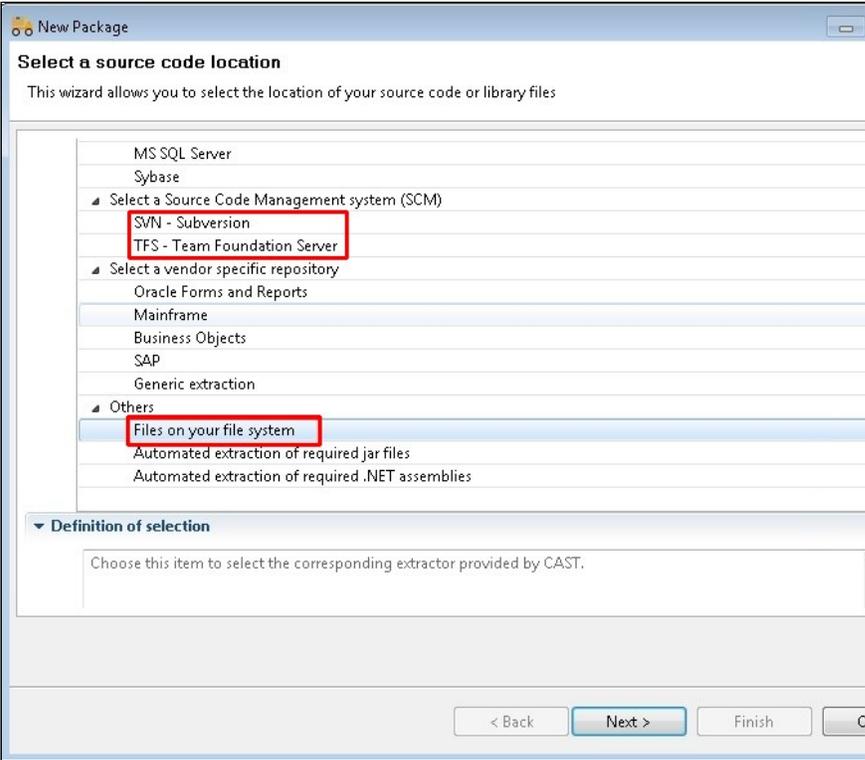
How do I add a source code package to my delivery

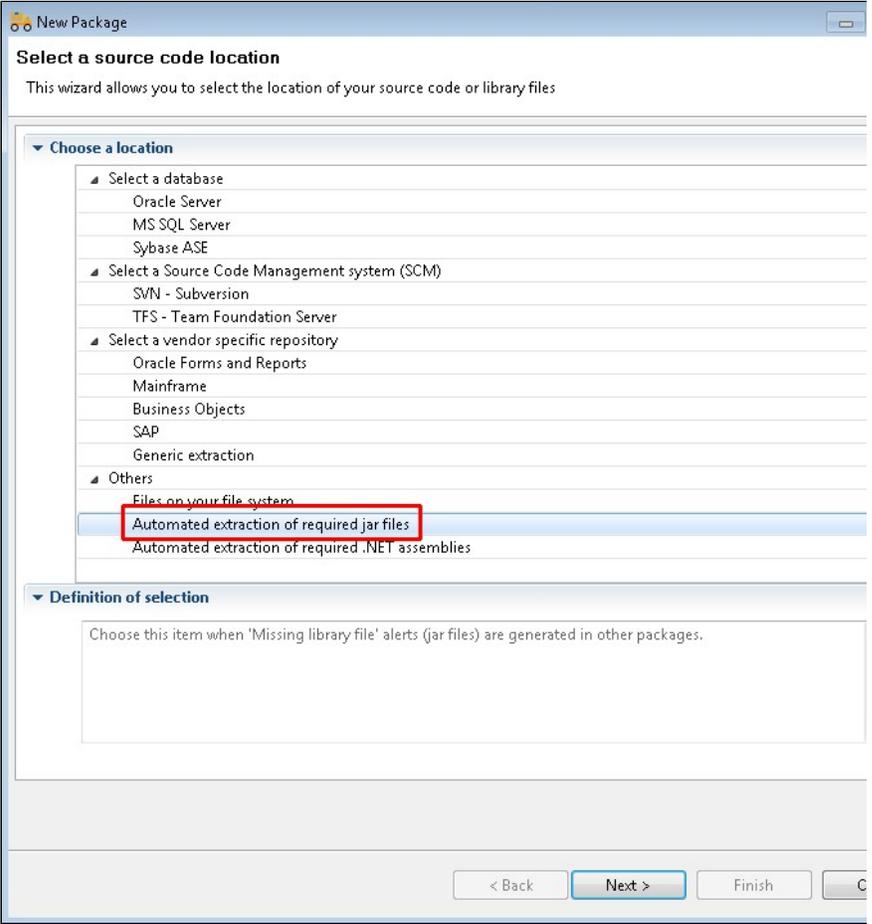
See [How do I add a source code package to my delivery](#).

What should you package?

When creating packages to discover and extract your JEE application you should create them as listed below:

Package	Package name /type	Mandatory?	Location /path	Notes

1	Source code	✓	Source code root folder	<p>Use the "Files on your file system" / SVN / TFS options in the CAST Delivery Manager Tool:</p> <p><i>Click to enlarge</i></p> 
---	-------------	---	-------------------------	--

2	Internal /external JAR files	✘	JAR file location	<p>Use the "Automated extraction of required jar files" options in the CAST Delivery Manager Tool:</p>  <p>This package is generally created after the initial packaging action when the CAST Delivery Manager Tool items and generates corresponding alerts.</p>
---	------------------------------	---	-------------------	--

Exclusions

The CAST Delivery Manager Tool offers various exclusion options for JEE applications so that certain projects that may be detected by a discoverer are ignored:

▼ Projects to exclude

Define criteria to filter the list of selected projects

◆ Exclusion rules

- Exclude all empty projects
- Exclude ASP .NET web projects when a Visual C#/basic .NET project also exists
- Exclude ASP projects when a .NET web project also exists
- Exclude basic JSP projects when a full JEE project also exists for the same web.xml file
- Exclude Eclipse Java projects when a Maven project also exists
- Exclude Maven Java projects when an Eclipse project also exists
- Exclude Eclipse project located inside the output folder of another Eclipse project
- Exclude Eclipse project sharing the name of another Eclipse project
- Exclude Duplicate Dot Net project located inside the exactly same source folder.
- Exclude Test Code



Note that:

- Any source code that contains both Maven and Eclipse projects and whose type is set to "**eclipse-plugin**" will be ignored by the [Maven Project Discoverer](#). An "**eclipse-plugin**" project means that an associated Eclipse project also exists and therefore should be handled by the Eclipse discoverer.
- Any source code that contains both Maven and Eclipse projects and whose type is set to "**maven-project**" will be ignored by the [Eclipse Project Discoverer](#). A "**maven-project**" project means that an associated Maven project also exists and therefore should be handled by the Maven discoverer.

In these situations, CAST highly recommends that you tick the appropriate exclusion rules in the CAST Delivery Manager Tool:

▼ **Projects to exclude**

Define criteria to filter the list of selected projects

◆ Exclusion rules

- Exclude all empty projects
- Exclude ASP .NET web projects when a Visual C#/basic .NET project also exists
- Exclude ASP projects when a .NET web project also exists
- Exclude basic JSP projects when a full JEE project also exists for the same web.xml file
- Exclude Eclipse Java projects when a Maven project also exists
- Exclude Maven Java projects when an Eclipse project also exists
- Exclude Eclipse project located inside the output folder of another Eclipse project
- Exclude Eclipse project sharing the name of another Eclipse project
- Exclude Duplicate Dot Net project located inside the exactly same source folder.
- Exclude Test Code

How do I package the Version?

See [How do I package the Version](#) for more information.

How do I validate and fine-tune my Version?

See [How do I fine-tune my Version](#) for more information.

How do I deliver the Version for analysis?

See [How do I deliver the Version for analysis](#) for more information.

Delivery acceptance

See [Validate and Accept the Delivery](#) for more information.