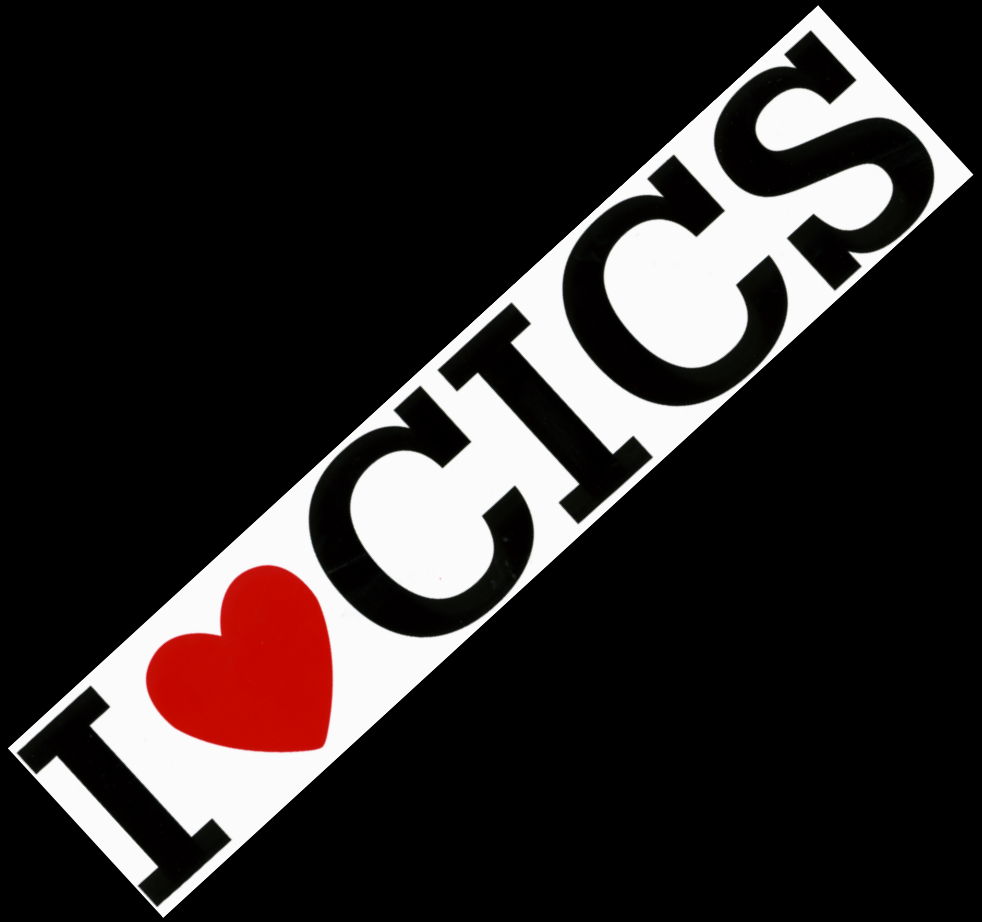
Do your application developers love CICS

Will Yates

wyates@uk.ibm.com

@hobbit1983



IBM

Developing a CICS application is the same as developing any other application

Our aim for developers

Developing a CICS application is the same as developing any other application

Works the same

Without good reason, why should development be different in CICS compared to another environment?

No one-size-fits-all

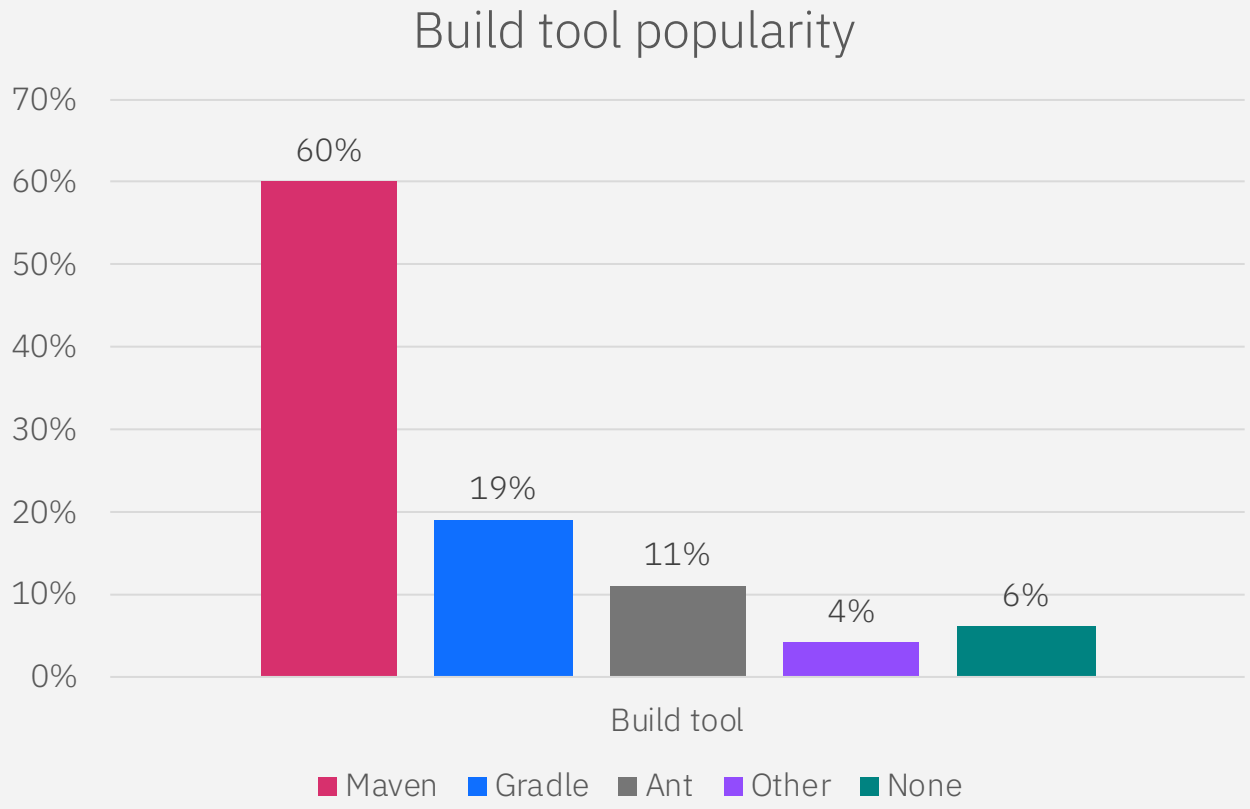
Experiences suited for each type of developer that creates applications for CICS.

Effortless collaboration

With the right security model, different roles can enable, rather than constrain, one another — and still maintain a strong security posture.

CICS Bundle Development Using Maven And Gradle

- Since CICS TS V4 you have been able to write Java applications in CICS
- Java development in CICS feels different to other platforms
- We want to make CICS Java development more immediately recognizable to all Java developers
- Going to look at the problems with Java development, and what we're doing to fix them!



Source: [Snyk Java ecosystem report 2018]

What are Maven and Gradle?

Rarely does a java application not depend on any external components

Examples

- Java libraries for common functions
- Java libraries to use a middleware product - like JCICS

Using external OSS components allow greater flexibility, speed and value to market

What are Maven and Gradle?

Pluggable build systems and dependency management for Java applications

Pluggable build systems

- Create plugins which encapsulate build logic
- Plugins are shareable across projects within an enterprise, or with the world
- Applications can declare dependencies on plugins too

What are Maven and Gradle?

Pluggable build systems and dependency management for Java applications

Dependency management:

- Online catalog of libraries: Maven Central
- Used by both Maven and Gradle builds
- More than 3 million reusable components
- Application can declare dependencies on these libraries
- Maven and Gradle take care of retrieving the library and using it at build-time / runtime

sonatype | The Central Repository Quick Stats

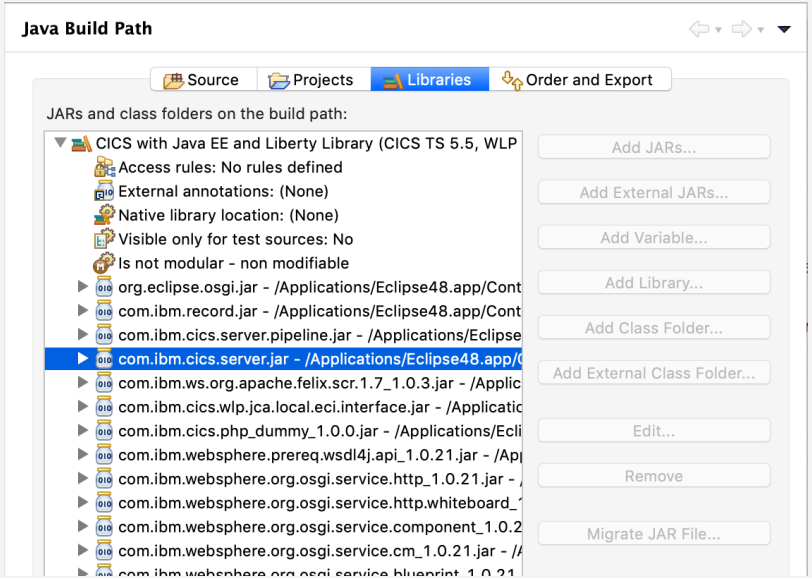
g:org.apache.commons AND a:commons-lang3

Group ID	Artifact ID	Latest Version
org.apache.commons	commons-lang3	3.9
org.apache.commons	commons-lang3	3.8.1
org.apache.commons	commons-lang3	3.8
org.apache.commons	commons-lang3	3.7
org.apache.commons	commons-lang3	3.6
org.apache.commons	commons-lang3	3.5

How's this fit into CICS Java development?

Not very smoothly!

- Java libraries for *EXEC CICS* API etc aren't available on Maven Central
- We built bespoke support in CICS Explorer for automatically setting up a compilation environment for you
  - Doesn't really help you if you want to use Maven or Gradle, though.
- Once you have the API to code against set up locally, actually writing the code in CICS Explorer is pretty good
- However, to get your Java into CICS, need to use a *CICS bundle*



Why do we need CICS bundles?

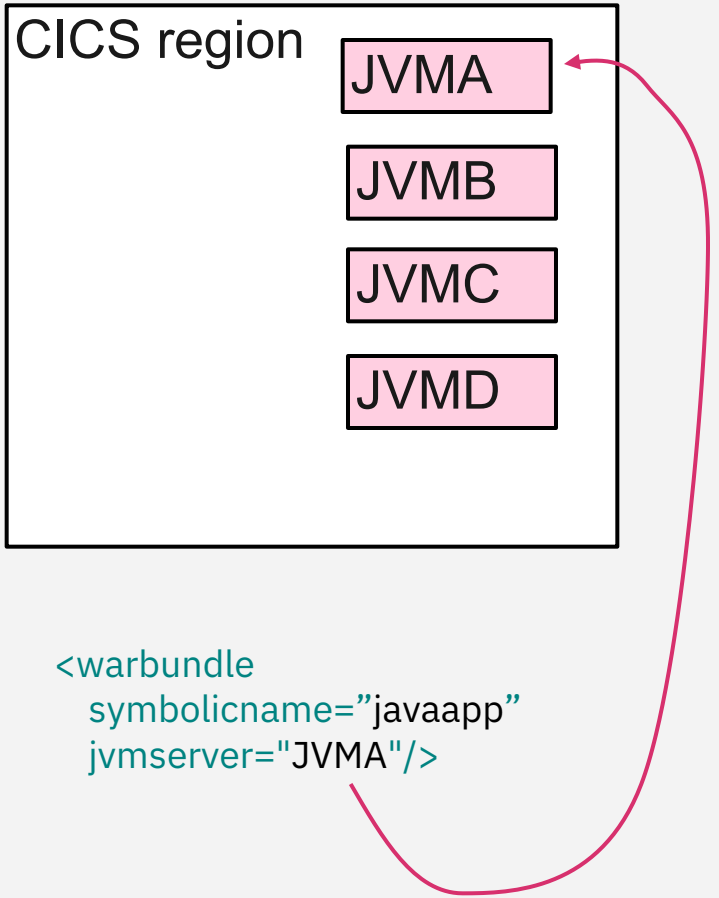
A CICS bundle is a package for deploying resources into CICS

Once our Java application is deployed, we need some way of referring to it so we can e.g. un-deploy it later

CICS bundle provides us

- Identity for a collection of resources
- Means of life-cycling them
- Means of supplying additional meta-data

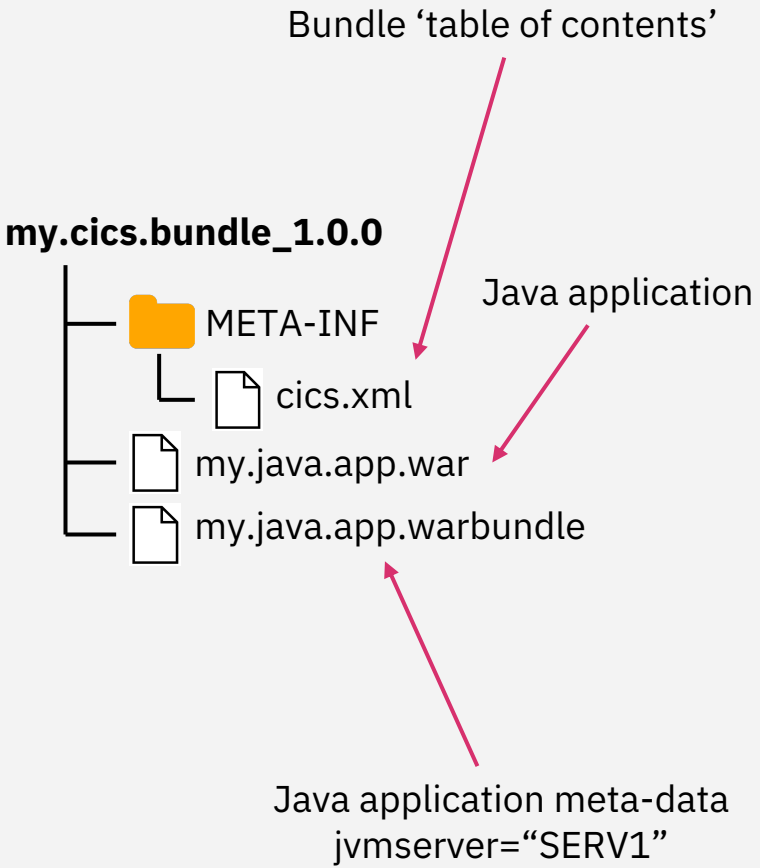
For Java applications we must specify **which JVM**

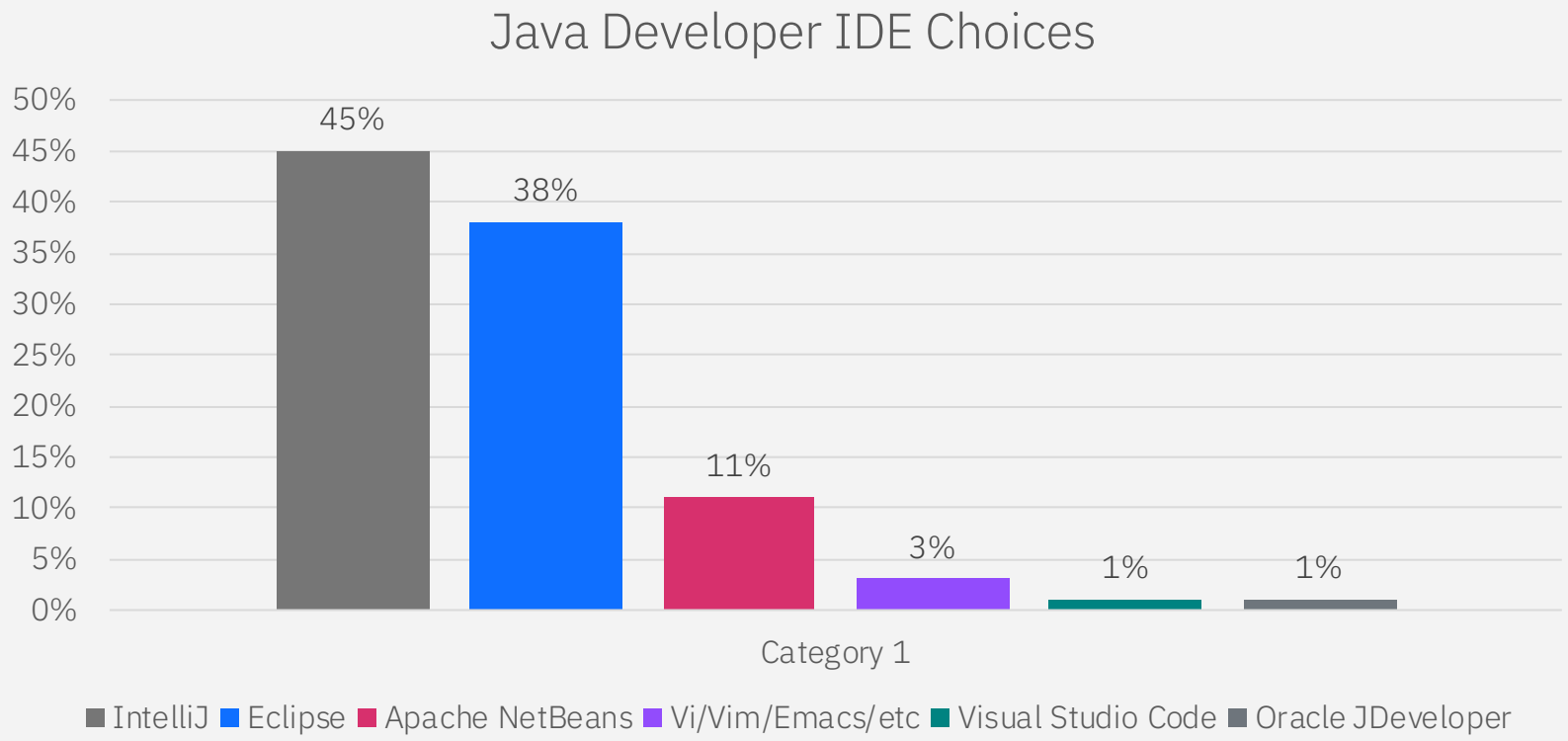


Anatomy of a CICS bundle

- Packaging entities for getting a lot of different stuff into CICS
  - Java applications
  - Programs
  - Transactions
  - URI maps
  - Policies
  - Event bindings
  - Etc

- Authored using CICS Explorer





Source: [Snyk Java ecosystem report 2018]

Problem summary:

- Can only author CICS bundles in CICS Explorer
- Java developer experience is better in CICS Explorer than more popular IDEs
- Difficult to effectively use Maven and Gradle in any environment

Let’s take a look at some of the issues with the Java developer experience in CICS Explorer…

CICS Explorer bundle publish demo

Our solution:

- Put our Java API libraries in Maven central
  - JCICS
    com.ibm.cics.server
  - Annotations
    com.ibm.cics.server.invocation.annotations
  - Annotation processor
    com.ibm.cics.server.invocation

- Create a Maven plugin for authoring CICS bundles, which can be used directly in your Java application build toolchain

- We're also thinking about something similar for Gradle, but that's still a W.I.P. Going to be discussing what's already available for Maven from now.

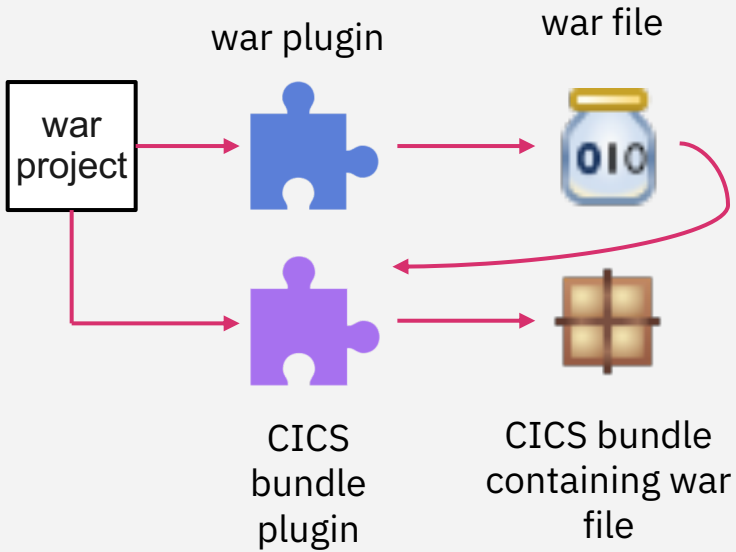
```
<dependency>
    <groupId>com.ibm.cics</groupId>
    <artifactId>com.ibm.cics.server</artifactId>
    <version>1.700.0-5.5-PH10453</version>
    <scope>provided</scope>
</dependency>
```

```
<plugin>
    <groupId>com.ibm.cics</groupId>
    <artifactId>cics-bundle-maven-plugin</artifactId>
    <version>0.0.1</version>
</plugin>
```

Produce a CICS bundle as part of the build for an existing Java application

Simplest use case allows automatic ‘CICS bundling’ of a Java application so it can be deployed into CICS

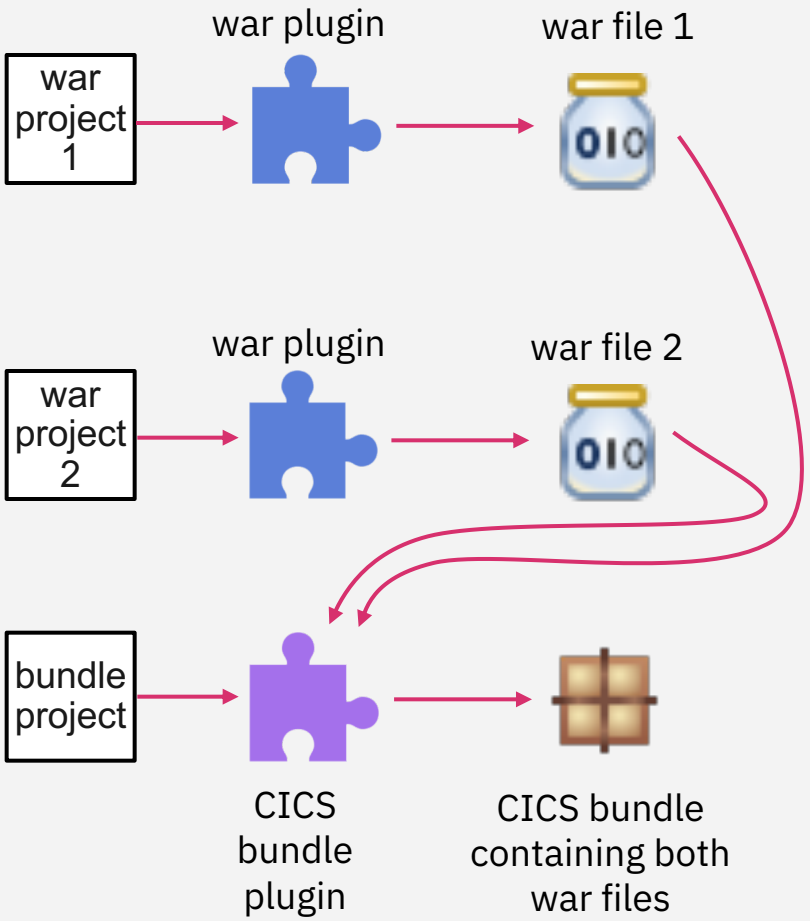
Minimal configuration, minimal customization



Create a dedicated CICS bundle module

This is more complicated to set up, but allows greater flexibility

- Add additional resources to the CICS bundle like transactions, and URI maps
- Add multiple Java applications to the same bundle



What does this solve?

This lets developers choose to use Maven to author CICS bundles

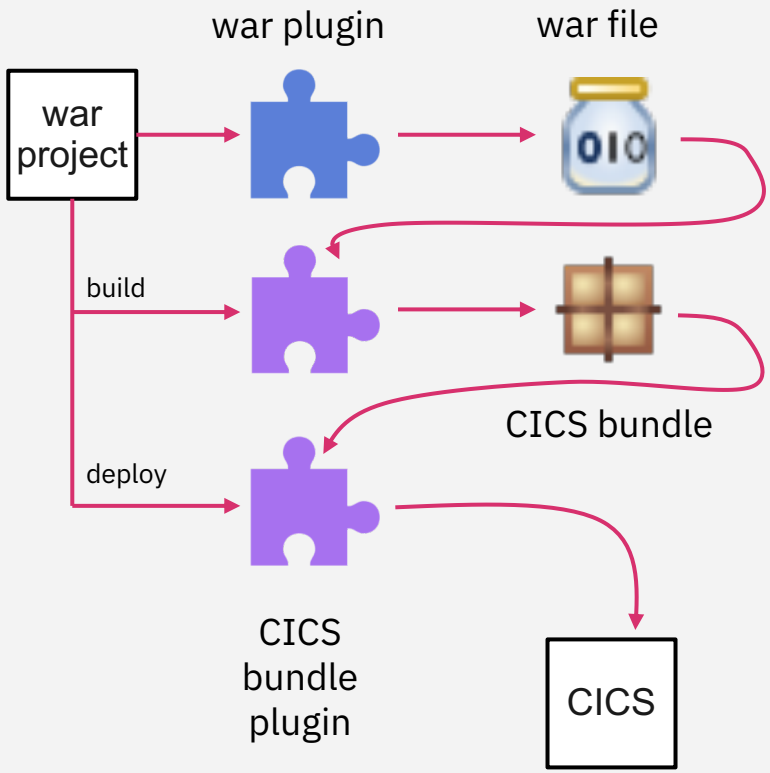
We wanted to allow developers to publish their applications from Maven, too.

We also wanted to fix a lot of the underlying problems with bundle deployment, too...

Bundle deployment API and Maven support

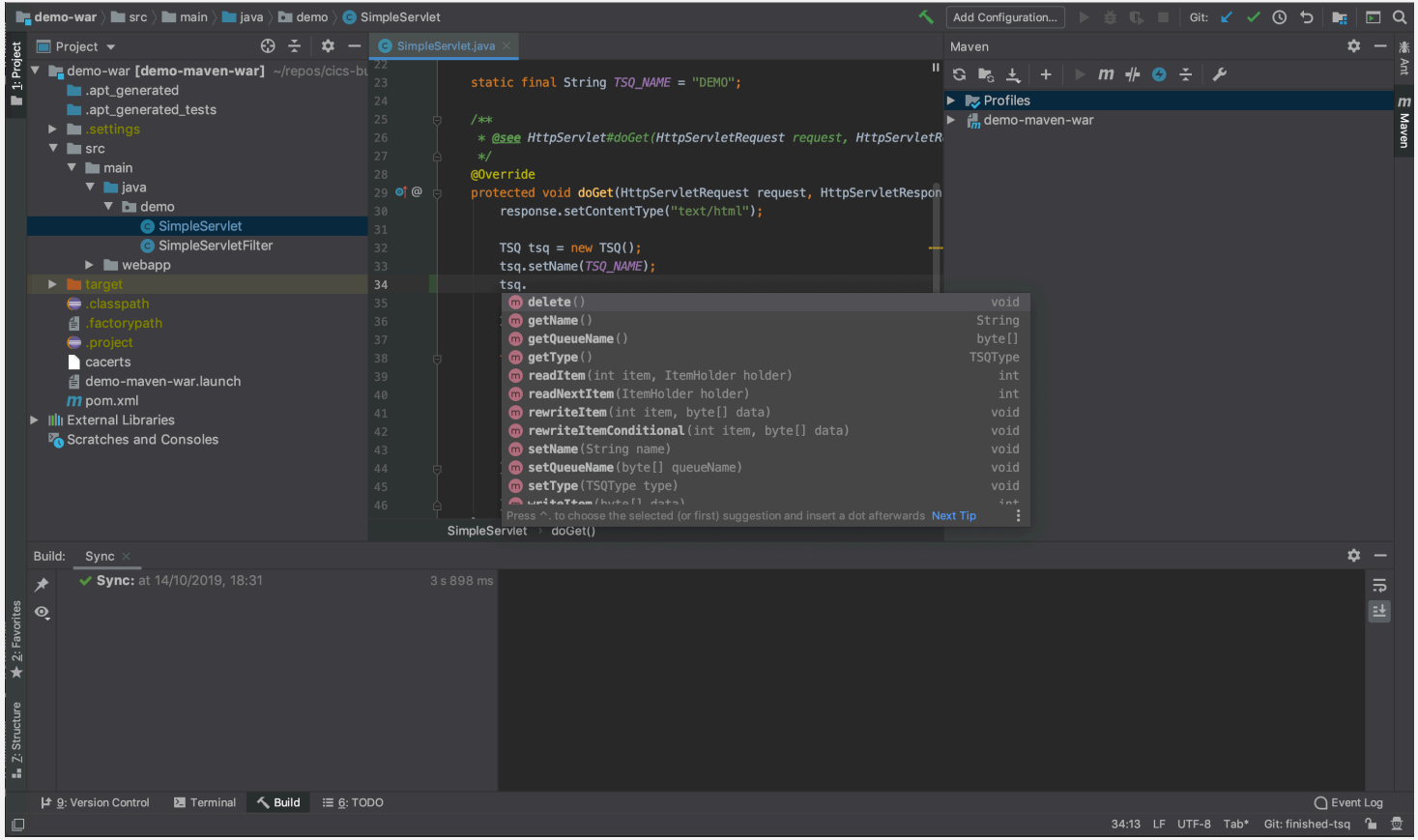
New API in CICS TS 5.6 Open Beta for publishing CICS bundles

- Does all the things that we had to do manually in the demo
- Isolates developers from having to understand the mechanics of how CICS bundles are implemented
- We can use this API in our new Maven plugin to integrate the publish process with the Java application build

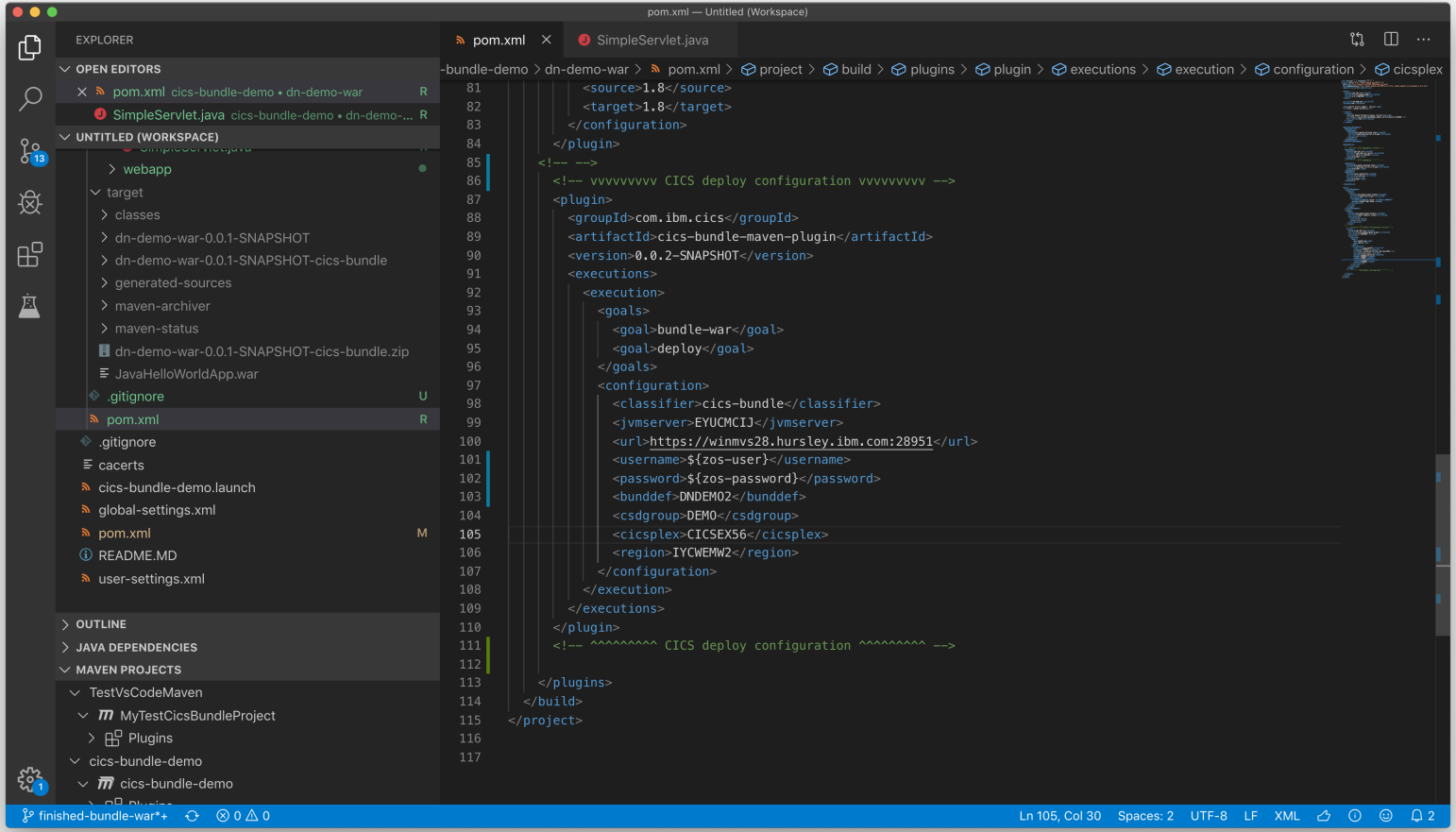


Demo2: publishing a bundle using the Maven plugin

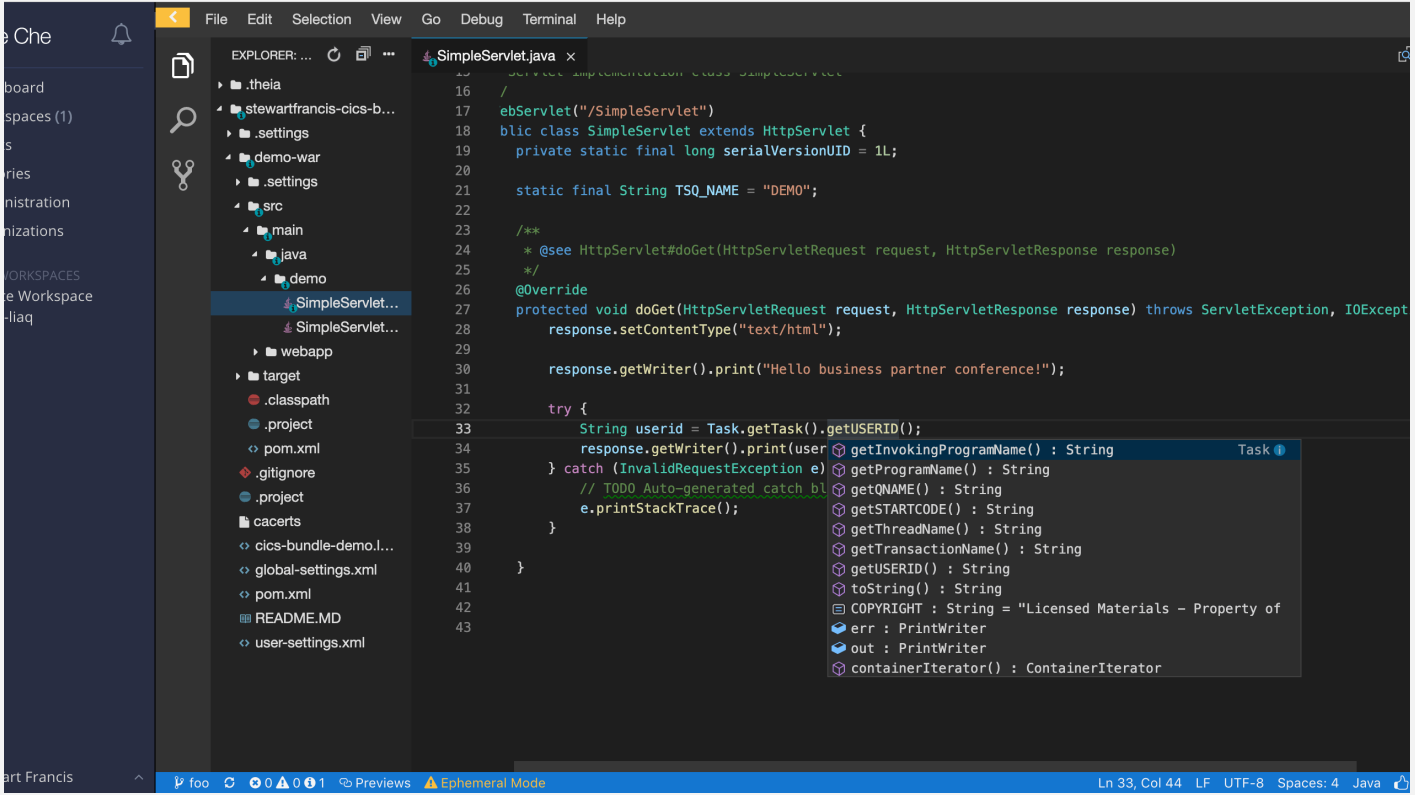
IntelliJ


demo-war
src
main
java
demo
SimpleServlet
Add Configuration...
Git:
Project
SimpleServlet.java
demo-war [demo-maven-war] ~/repos/cics-bt
.apt_generated
.apt_generated_tests
.settings
src
main
java
demo
SimpleServlet
SimpleServletFilter
webapp
target
.classpath
.factorypath
.project
cacerts
demo-maven-war.launch
pom.xml
External Libraries
Scratches and Consoles
static final String TSQ_NAME = "DEMO";
/**
* @see HttpServlet#doGet(HttpServletRequest request, HttpServletR
*/
@Override
protected void doGet(HttpServletRequest request, HttpServletRespon
response.setContentType("text/html");
TSQ tsq = new TSQ();
tsq.setName(TSQ_NAME);
tsq.
delete() void
getName() String
getQueueName() byte[]
getType() TSQType
readItem(int item, ItemHolder holder) int
readNextItem(ItemHolder holder) int
rewriteItem(int item, byte[] data) void
rewriteItemConditional(int item, byte[] data) void
setName(String name) void
setQueueName(byte[] queueName) void
setType(TSQType type) void
Press ^. to choose the selected (or first) suggestion and insert a dot afterwards Next Tip
SimpleServlet
doGet()
Maven
Profiles
demo-maven-war
Build:
Sync
Sync: at 14/10/2019, 18:31
3 s 898 ms
9: Version Control
Terminal
Build
6: TODO
Event Log
34:13
LF
UTF-8
Tab*
Git: finished-tsq

VS Code


pom.xml — Untitled (Workspace)
EXPLORER
OPEN EDITORS
pom.xml cics-bundle-demo • dn-demo-war
SimpleServlet.java cics-bundle-demo • dn-demo-...
UNTITLED (WORKSPACE)
webapp
target
classes
dn-demo-war-0.0.1-SNAPSHOT
dn-demo-war-0.0.1-SNAPSHOT-cics-bundle
generated-sources
maven-archiver
maven-status
dn-demo-war-0.0.1-SNAPSHOT-cics-bundle.zip
JavaHelloWorldApp.war
.gitignore
pom.xml
.gitignore
cacerts
cics-bundle-demo.launch
global-settings.xml
pom.xml
README.MD
user-settings.xml
OUTLINE
JAVA DEPENDENCIES
MAVEN PROJECTS
TestVsCodeMaven
MyTestCicsBundleProject
Plugins
cics-bundle-demo
cics-bundle-demo
pom.xml
SimpleServlet.java
-bundle-demo > dn-demo-war > pom.xml > project > build > plugins > plugin > executions > execution > configuration > cicsplex
<source>1.8</source>
<target>1.8</target>
</configuration>
</plugin>
<!-- -->
<!-- vvvvvvvvv CICS deploy configuration vvvvvvvvv -->
<plugin>
<groupId>com.ibm.cics</groupId>
<artifactId>cics-bundle-maven-plugin</artifactId>
<version>0.0.2-SNAPSHOT</version>
<executions>
<execution>
<goals>
<goal>bundle-war</goal>
<goal>deploy</goal>
</goals>
<configuration>
<classifier>cics-bundle</classifier>
<jvmserver>EYUCMCIJ</jvmserver>
<url>https://winmvs28.hursley.ibm.com:28951</url>
<username>${zos-user}</username>
<password>${zos-password}</password>
<bunddef>DNDEMO2</bunddef>
<csdgroup>DEMO</csdgroup>
<cicsplex>CICSEX56</cicsplex>
<region>IYCWEMW2</region>
</configuration>
</execution>
</executions>
</plugin>
<!-- ^^^^^^^^^ CICS deploy configuration ^^^^^^^^^ -->
</plugins>
</build>
</project>
finished-bundle-war*+
0 0
Ln 105, Col 30
Spaces: 2
UTF-8
LF
XML
2

Eclipse Che


File Edit Selection View Go Debug Terminal Help
EXPLORER: ...
SimpleServlet.java
.theia
stewartfrancis-cics-b...
.settings
demo-war
.settings
src
main
java
demo
SimpleServlet...
SimpleServlet...
webapp
target
.classpath
.project
pom.xml
.gitignore
.project
cacerts
cics-bundle-demo.l...
global-settings.xml
pom.xml
README.MD
user-settings.xml
@see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
@Override
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOExcept
response.setContentType("text/html");
response.getWriter().print("Hello business partner conference!");
try {
String userid = Task.getTask().getUSERID();
response.getWriter().print(user
} catch (InvalidRequestException e)
// TODO Auto-generated catch bl
e.printStackTrace();
}
}
getInvokingProgramName() : String
Task
getProgramName() : String
getQNAME() : String
getSTARTCODE() : String
getThreadName() : String
getTransactionName() : String
getUSERID() : String
toString() : String
COPYRIGHT : String = "Licensed Materials – Property of
err : PrintWriter
out : PrintWriter
containerIterator() : ContainerIterator
foo
Previews
Ephemeral Mode
Ln 33, Col 44
LF
UTF-8
Spaces: 4
Java

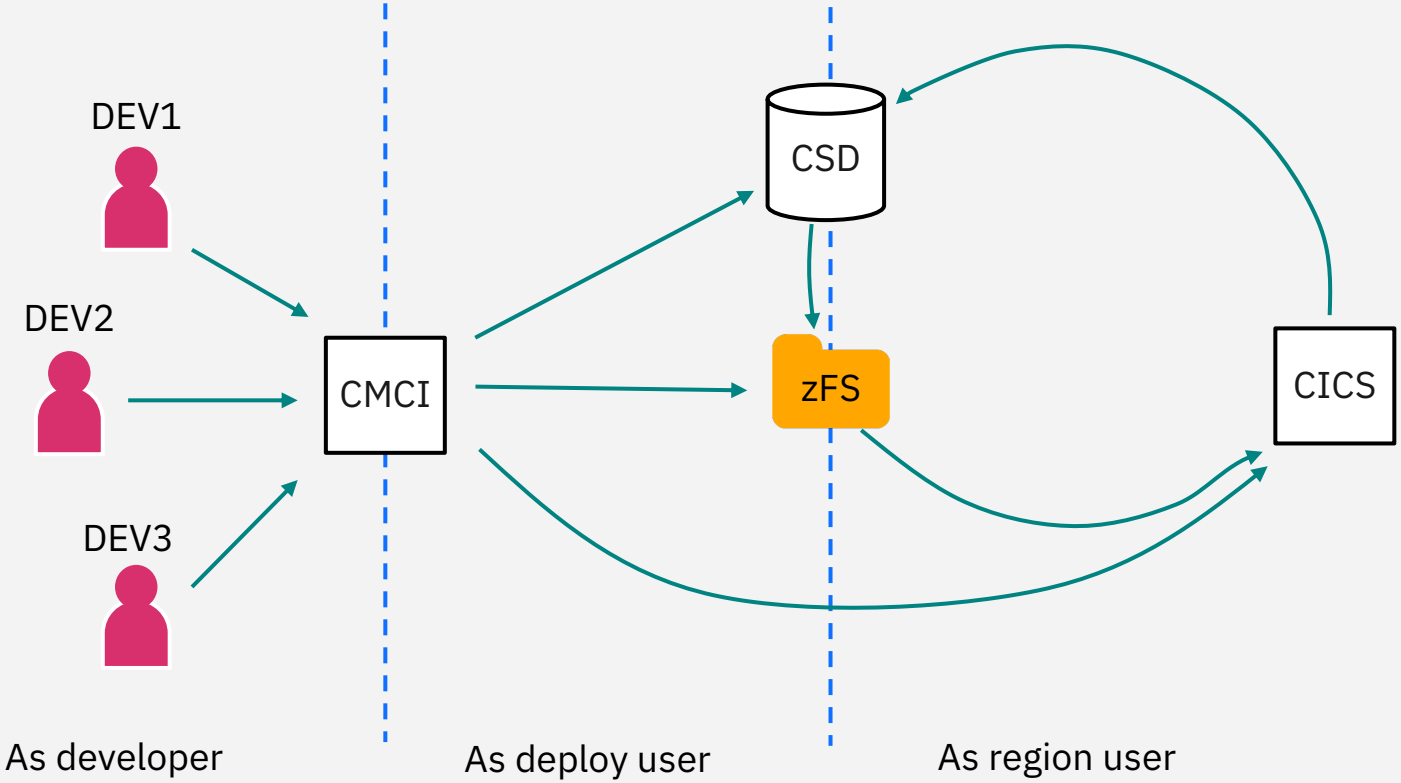
What have we gained?

- System programmers can pre-configure which bundles developers can publish
- Application developers can choose the IDE which best suits them
- Application developers are liberated from having to understand CICS bundles and how to use them
- Deploying applications is **faster** and **easier**

And...

- System programmers can give application developers autonomy without sacrificing control

How does the API work?



Want to try it out?

cics-bundle-maven
- Open source on GitHub
- Published to Maven Central

Bundle publish API
- New in CICS TS 5.6 Open Beta
- We're thinking about back-porting to TS 5.5
- Requires CPSM
- We're thinking about options for support without CPSM
- Requires CMCI JVM server (which gets you other nice stuff anyway!)

```
<plugin>
    <groupId>com.ibm.cics</groupId>
    <artifactId>cics-bundle-maven-plugin</artifactId>
    <version>0.0.1</version>
</plugin>
```

The COBOL, PL/I and Assembler developer experience

Editing COBOL, PL/I and Assembler

IDz has been the premier editor for these languages for some time.

If you’re using a modern SCM and modern editors, you might want to consider z Open Development (ZOD):

- Modern Eclipse editors
- Modern SCM integration
- Dependency-based build

Editing COBOL, PL/I and Assembler

Alternatively, the Z Open Editor for COBOL and PL/I, an extension for Visual Studio Code, can be used.

Not yet externally available but should be in the Visual Studio Code marketplace soon.

Integrates well with the Zowe extension.

The COBOL, PL/I and Assembler developer toolbox

IDz offers the premier editing experience, with syntax highlighting, compilation, and SCM integration.

If familiar with modern editors and modern SCM, consider ZOD.

More modern and standard SCMs and CI servers can be used instead of legacy library systems.

Couple them with Dependency-Based Build (DBB).

Look out for the Visual Studio Code-based editor.

Developing applications for CICS

Embracing open

Embracing existing technologies

Embracing the right options for your development chalenges

Modern Editors

Modern Source Code Management

Automated Testing

CI/CD Integration

Help us make more improvements to Java development for CICS TS!

We’d love to hear from your Java development teams, to help focus our ideas for what to do next

Get in touch with us via email, or your advocates

Fill out our survey for Java developers

Thanks for listening!