Enterprise Java in CICS TS using Maven and Gradle

Stew Francis – stewartfrancis@uk.ibm.com

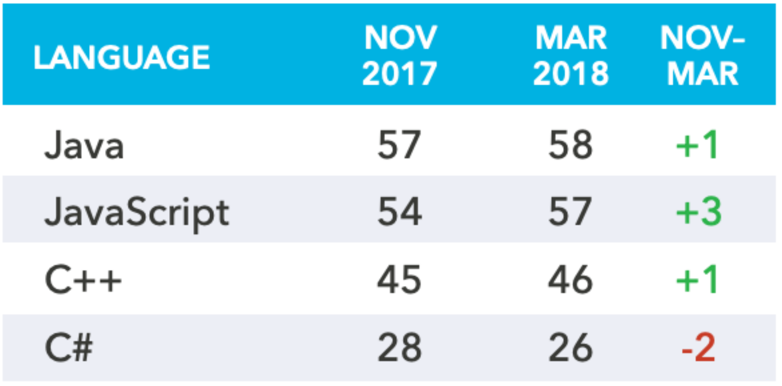
Why are we talking about enterprise modernization using Java?

- Skills shortages with traditional Z languages may inhibit agility
- But we can run lots of modern languages on mainframe (Java, Node, Python…)
- Modern language adoption can expose mainframe applications to new talent pools
- Great integration with CICS as we'll see later…

https://ibmsystemsmag.com/IBM-Z/07/2017/java-on-z-systems-portability

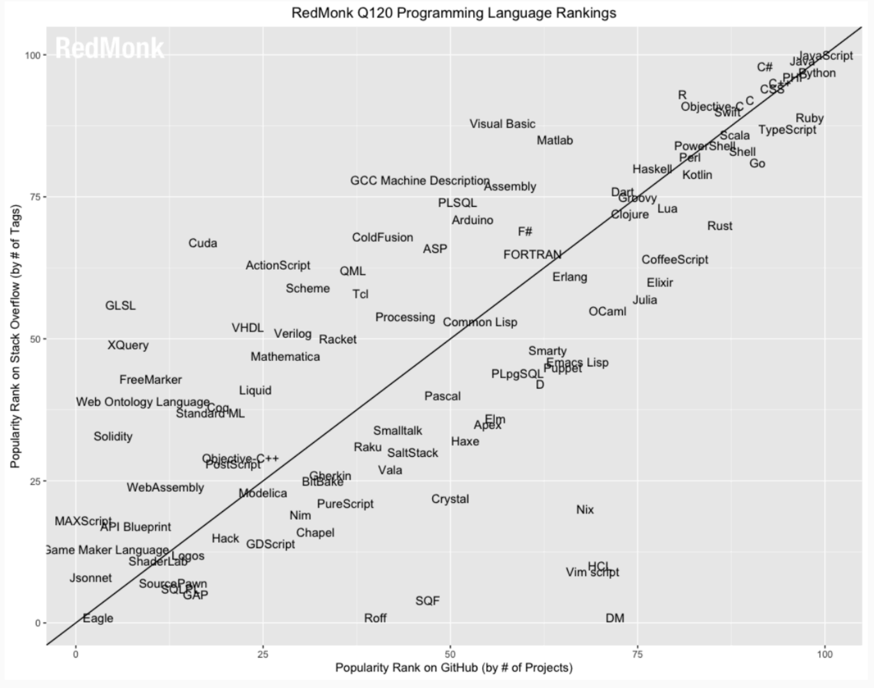
How popular is Java anyway?

"Over the last nine months, Java and JavaScript have remained the two dominant languages in the enterprise developer landscape…"



LANGUAGE	NOV 2017	MAR 2018	NOV-MAR
Java	57	58	+1
JavaScript	54	57	+3
C++	45	46	+1
C#	28	26	-2

https://www.cloudfoundry.org/wp-content/uploads/Developer-Language-Report_FINAL.pdf (2018)



https://redmonk.com/sogrady/2020/02/28/language-rankings-1-20/ (2020)

CICS TS as a host for Java

- CICS TS has had some form of Java support since 1998!
- CICS TS is a great mixed language application server
- JCICS is our library for interacting with CICS TS from Java, including linking to programs written in other languages
- Many more enhancements than listed on the right! Particularly in the last 8 years, where we've really focused on Java
- In CICS TS 5.6 we've really been trying to make the skillset of existing Java developers directly applicable to z/OS and CICS TS

CICS Transaction Server for OS/390 1.3 (1998)

- Support for the Java Virtual Machine (JVM)

CICS Transaction Server for z/OS 2.1 (2001)

- EJBs

CICS Transaction Server for z/OS 2.3 (2003)

- Java 1.4

CICS Transaction Server for z/OS 4.2 (2011)

- OSGi JVM servers, 64-bit Java

CICS Transaction Server for z/OS 5.1 (2012)

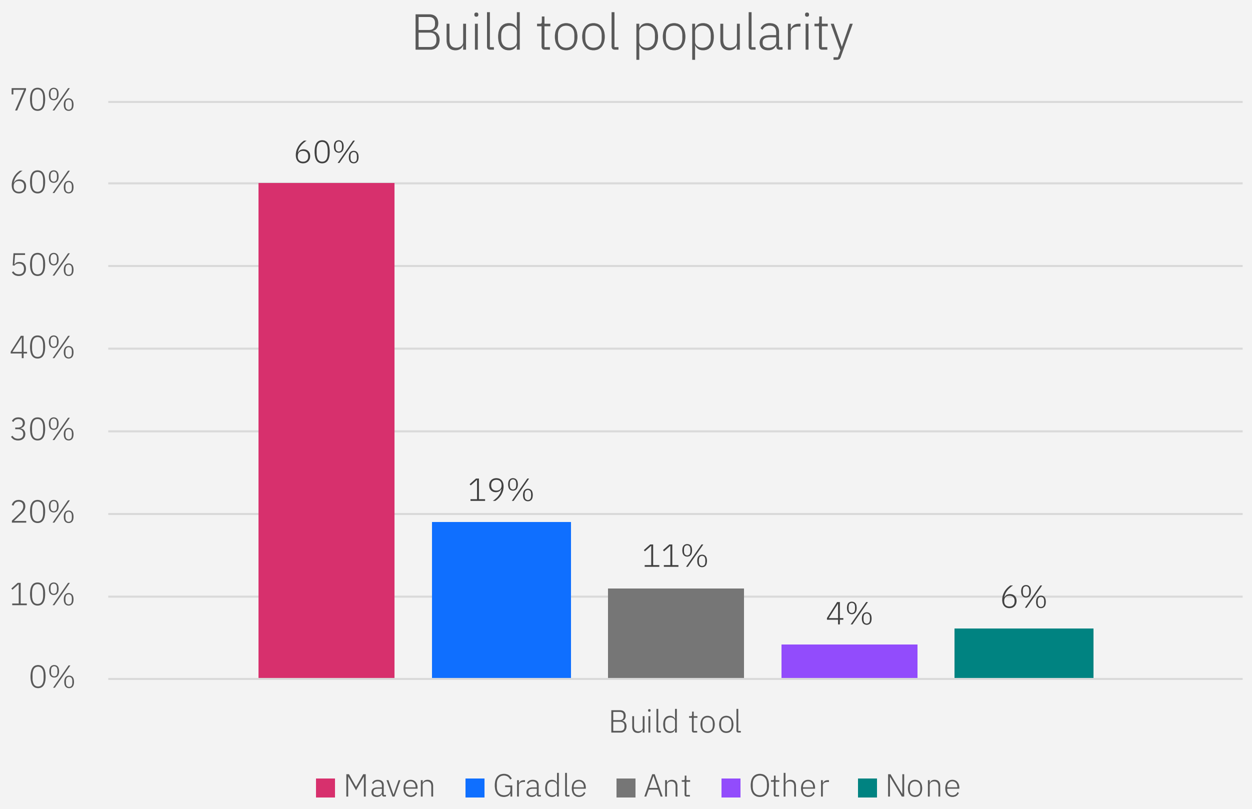
- WebSphere Application Server Liberty profile

CICS Java Development Using Maven And Gradle

- 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!



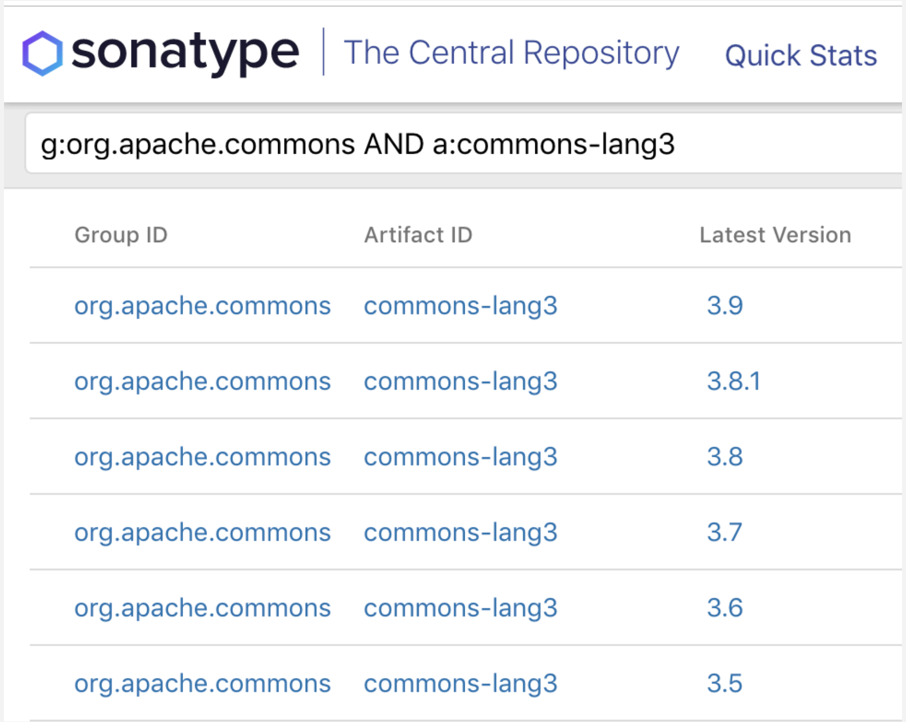
Source: Snyk Java ecosystem report 2018

What are Maven and Gradle?

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
- Applications can declare dependencies on these libraries
- Maven and Gradle take care of retrieving the library and using it at build-time / runtime



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

What are Maven and Gradle?

Pluggable build systems and dependency management for Java applications

Pluggable build systems

- Create plugins which encapsulate build logic

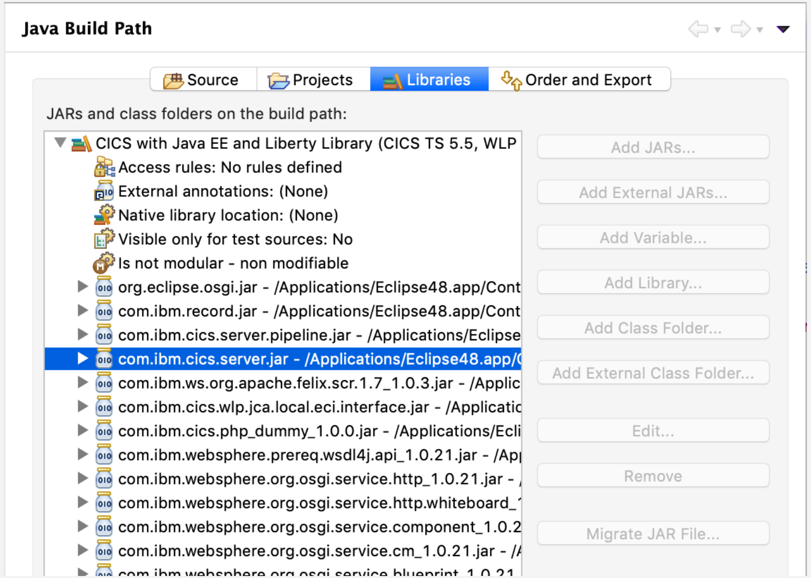
- Plugins are shareable across projects within an enterprise, or with the world

- Applications can declare dependencies on plugins too

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*



Java Build Path

JARs and class folders on the build path:

```
CICS with Java EE and Liberty Library (CICS TS 5.5, WLP
   Access rules: No rules defined
   External annotations: (None)
   Native library location: (None)
   Visible only for test sources: No
   Is not modular - non modifiable
  org.eclipse.osgi.jar - /Applications/Eclipse48.app/Cont
  com.ibm.record.jar - /Applications/Eclipse48.app/Cont
  com.ibm.cics.server.pipeline.jar - /Applications/Eclipse
  com.ibm.cics.server.jar - /Applications/Eclipse48.app/(
  com.ibm.ws.org.apache.felix.scr.1.7_1.0.3.jar - /Applic
  com.ibm.cics.wlp.jca.local.eci.interface.jar - /Applicatic
  com.ibm.cics.php_dummy_1.0.0.jar - /Applications/Ecli
  com.ibm.websphere.prereq.wsdl4j.api_1.0.21.jar - /Ap|
  com.ibm.websphere.org.osgi.service.http_1.0.21.jar - /
  com.ibm.websphere.org.osgi.service.http.whiteboard_1
  com.ibm.websphere.org.osgi.service.component_1.0.2
  com.ibm.websphere.org.osgi.service.cm_1.0.21.jar - /A
  com.ibm.websphere.org.osgi.service.blueprint_1.0.21
```

Why do we need CICS bundles?

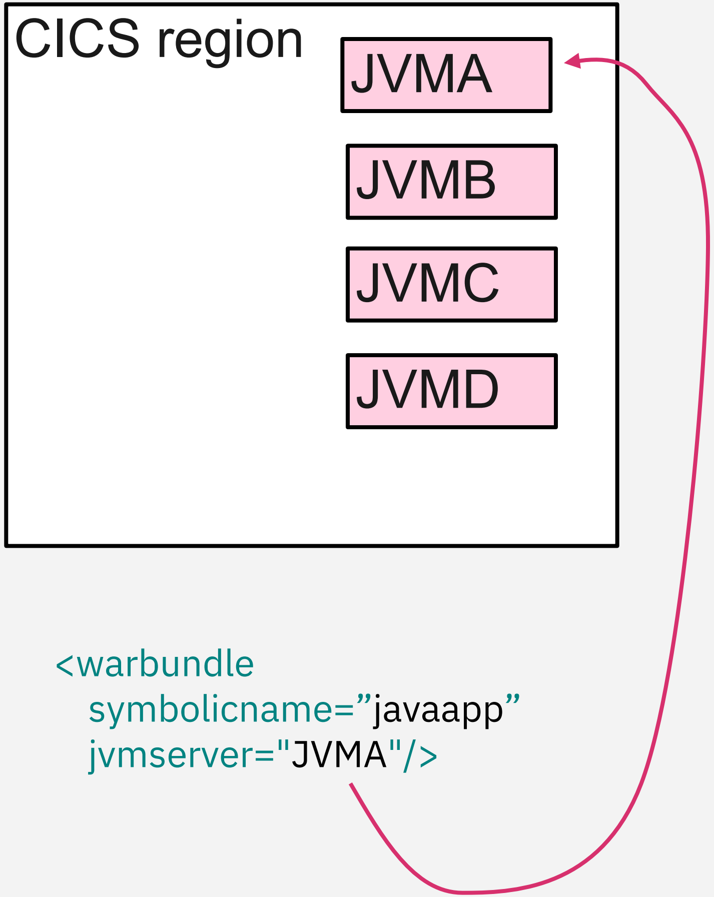
A CICS bundle is a package for deploying resources into CICS

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**

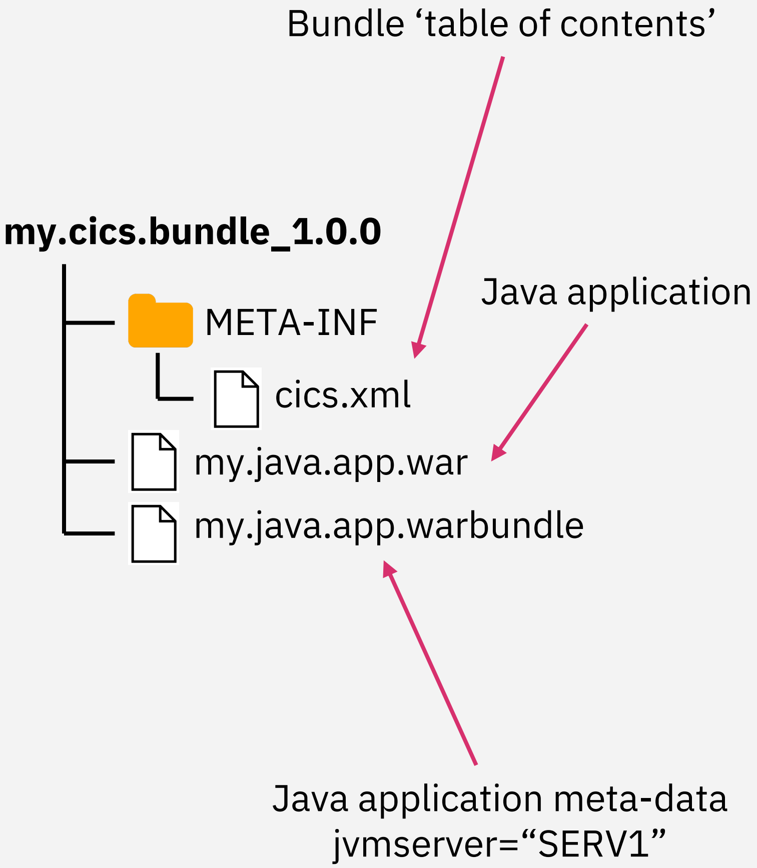


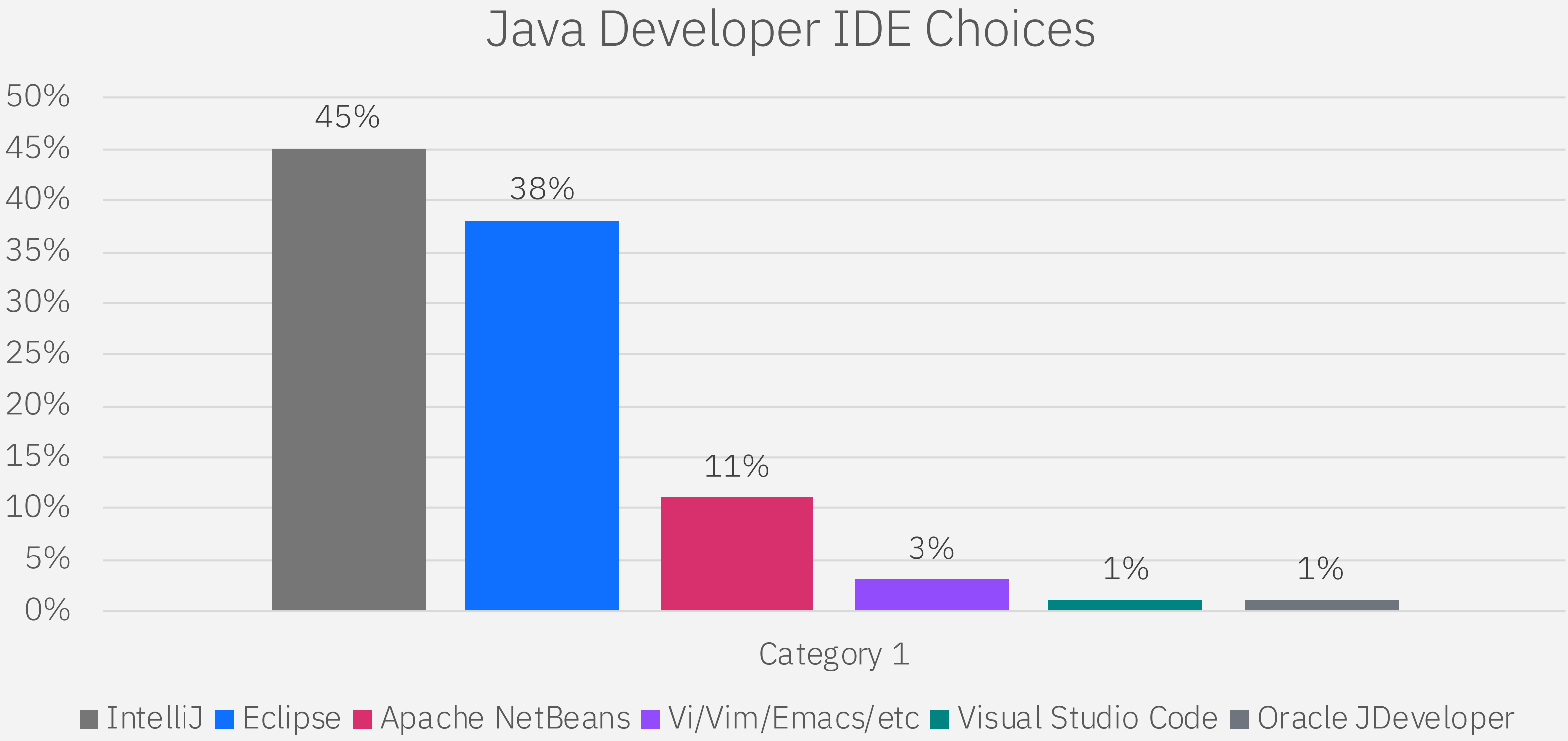
```
<warbundle
  symbolicname=”javaapp”
  jvmserver="JVMA"/>
```

Anatomy of a CICS bundle

- Packaging entities for getting a lot of different stuff into CICS
  - Java applications
  - 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…

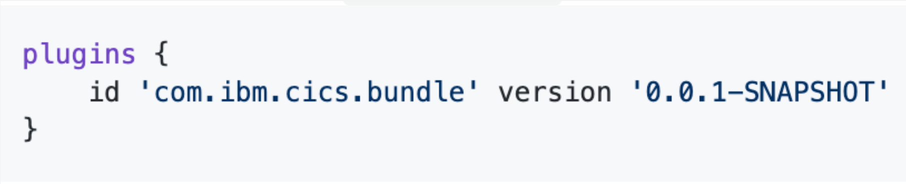
Our solution:

- 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

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

```
<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>
```

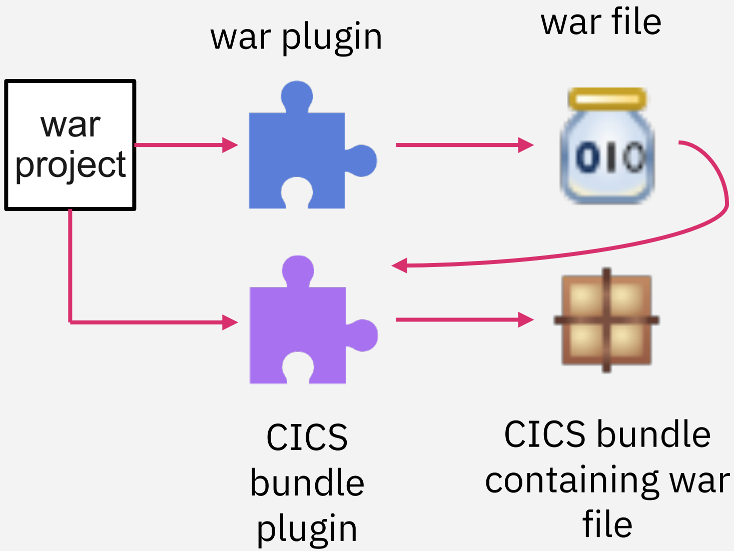


```
plugins {
    id 'com.ibm.cics.bundle' version '0.0.1-SNAPSHOT'
}
```

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

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

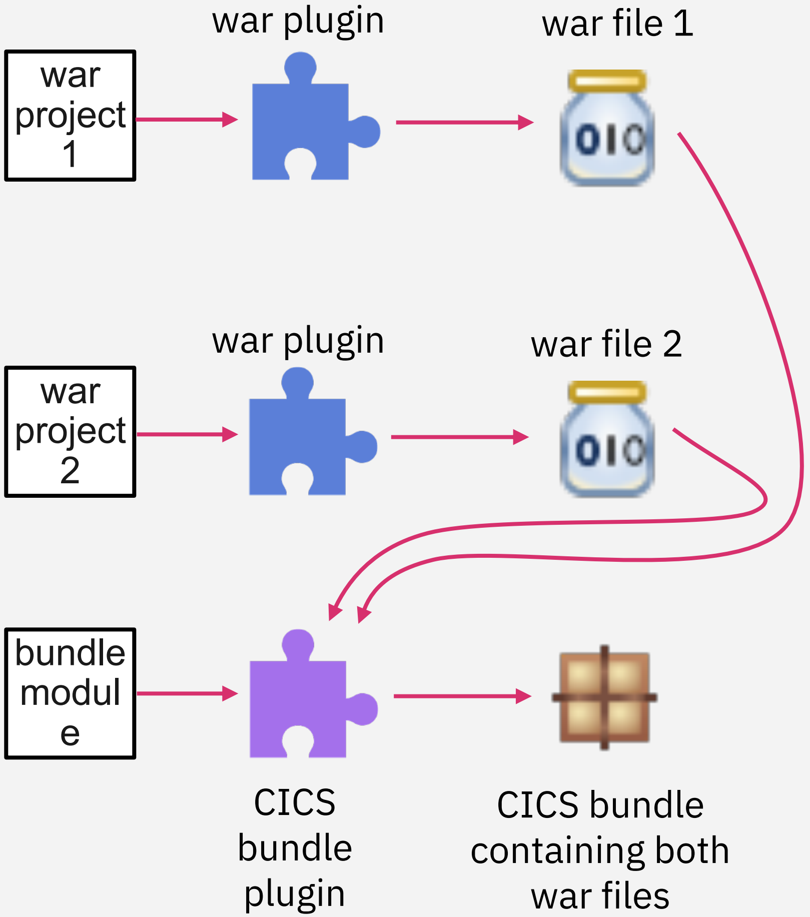
Minimal configuration, minimal customization



Create a dedicated CICS bundle module integrated into an existing Maven/Gradle build

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 or Gradle to author CICS bundles

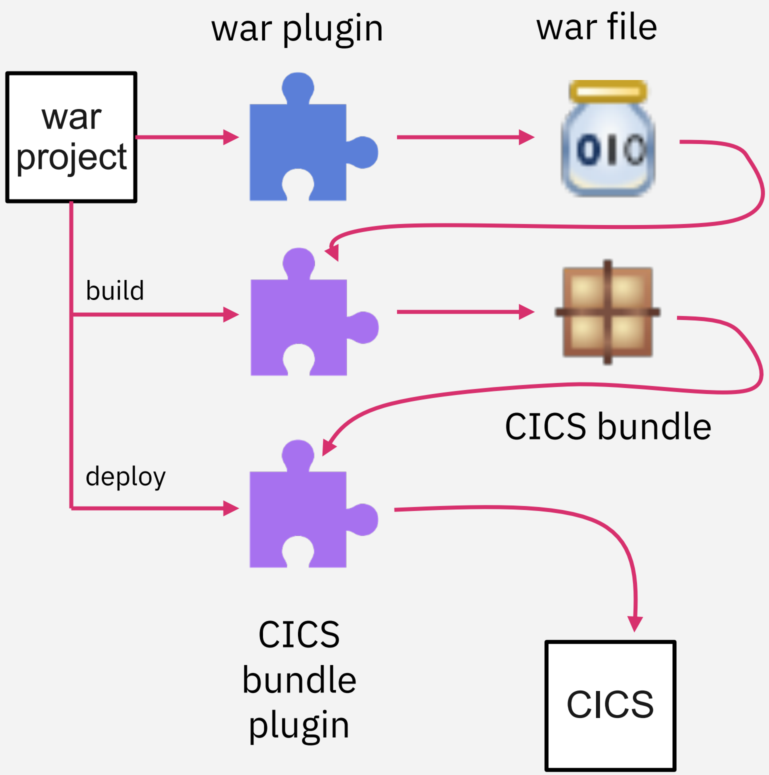
We wanted to allow developers to publish their applications from Maven and Gradle

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

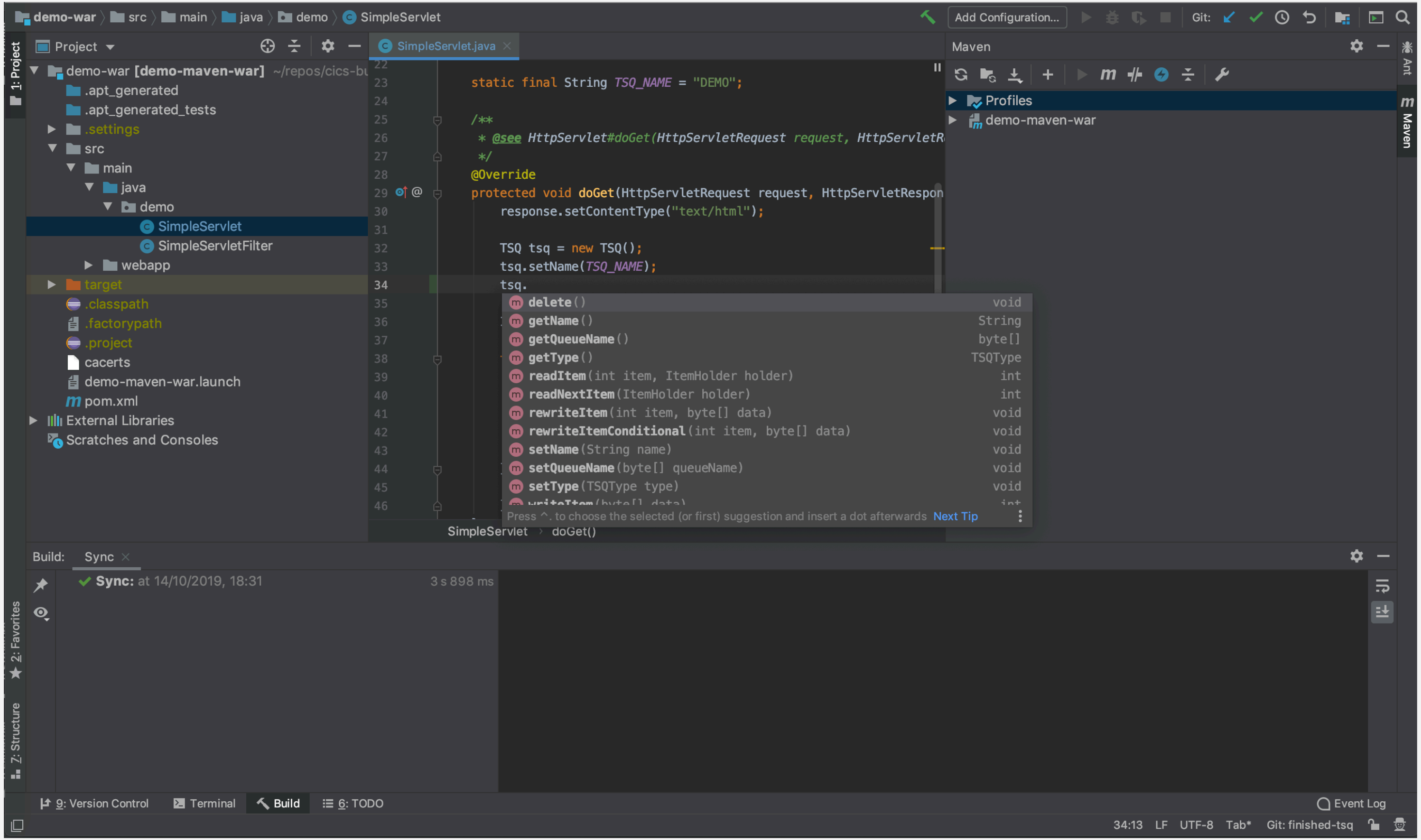
Bundle deployment API and Maven/Gradle support

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 and Gradle plugins to integrate the publish process with the Java application build



IntelliJ

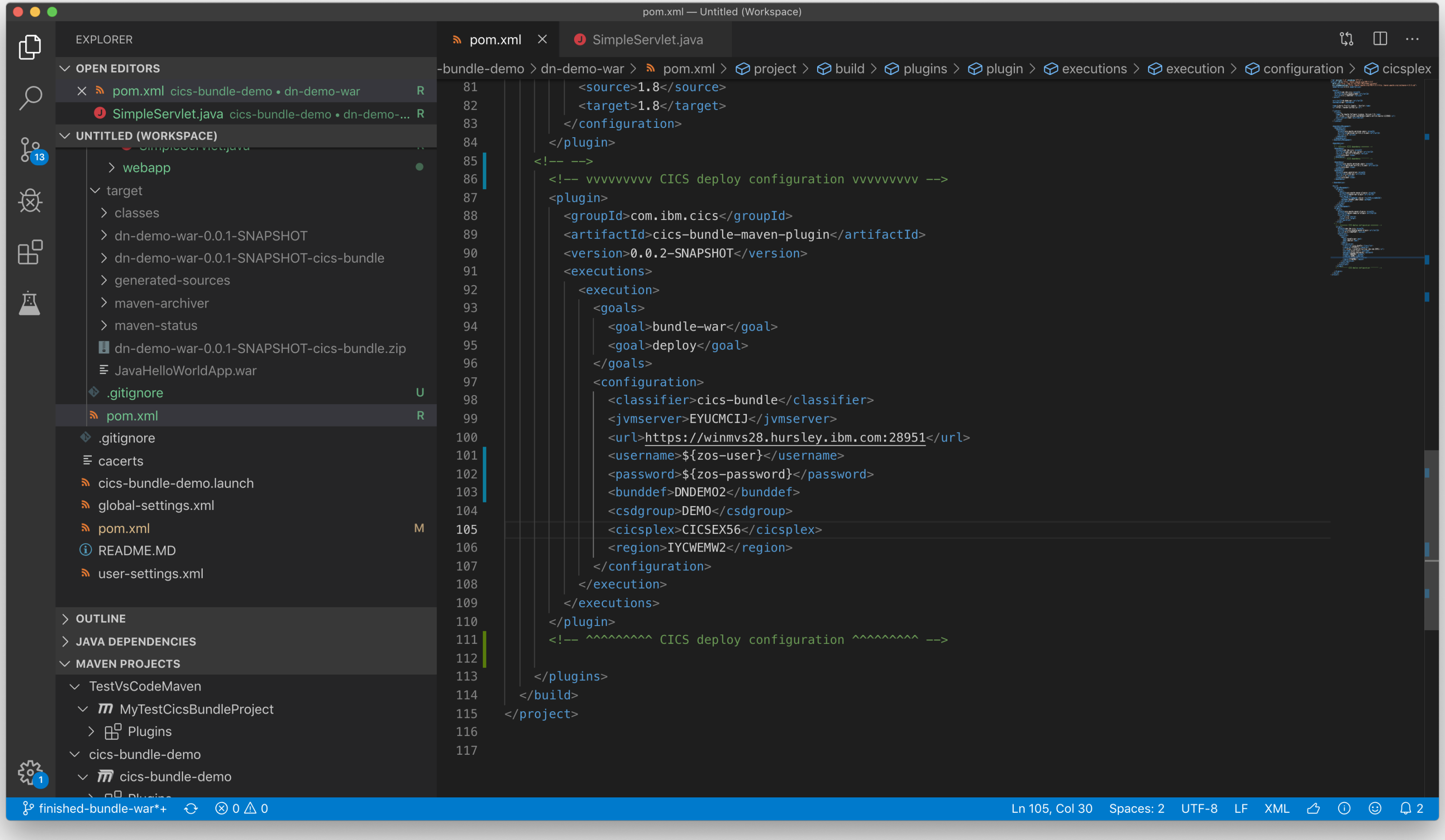


```
    static final String TSQ_NAME = "DEMO";

    /**
     * @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.
```

```
m delete()                                        void
m getName()                                     String
m getQueueName()                                byte[]
m getType()                                    TSQType
m readItem(int item, ItemHolder holder)            int
m readNextItem(ItemHolder holder)                  int
m rewriteItem(int item, byte[] data)              void
m rewriteItemConditional(int item, byte[] data)   void
m setName(String name)                            void
m setQueueName(byte[] queueName)                  void
m setType(TSQType type)                           void
m writeItem(byte[] data)                           int
Press ^. to choose the selected (or first) suggestion and insert a dot afterwards Next Tip
```

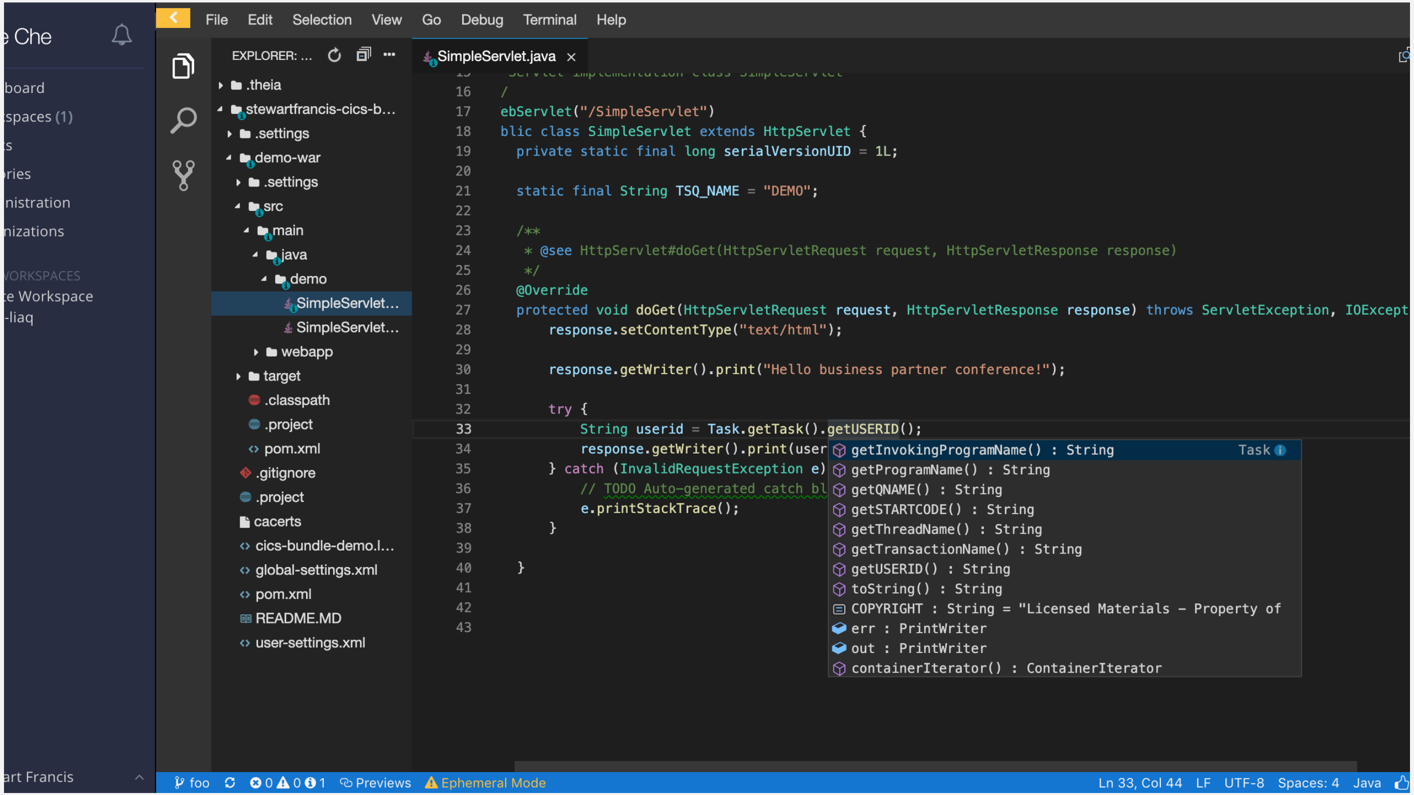
```
✓ Sync: at 14/10/2019, 18:31        3 s 898 ms
```

VS Code



```
          <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>
```

Eclipse Che



```
 Servlet implementation class SimpleServlet
/
ebServlet("/SimpleServlet")
blic class SimpleServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;

  static final String TSQ_NAME = "DEMO";

  /**
   * @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();
      }

  }
```

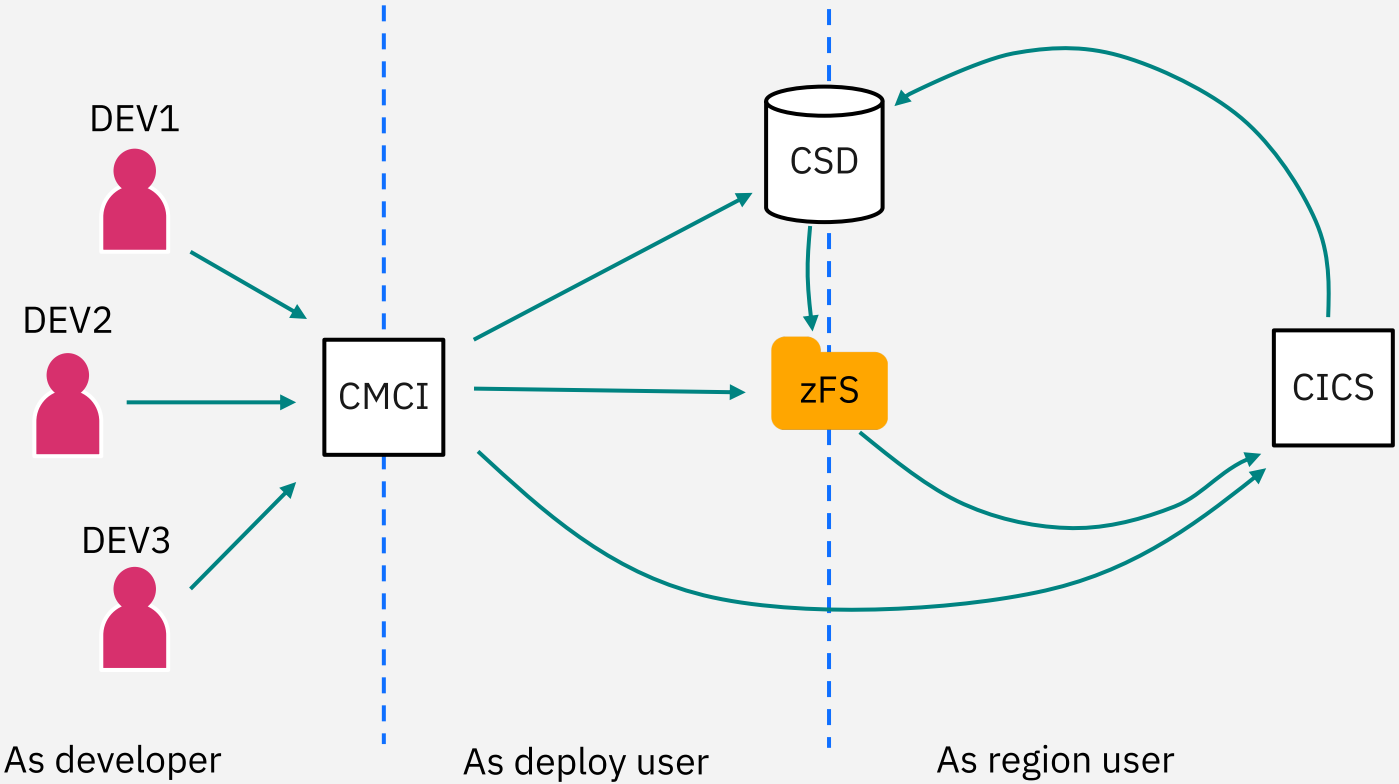
What have we gained?

- 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
- Version 1.0.0 released!

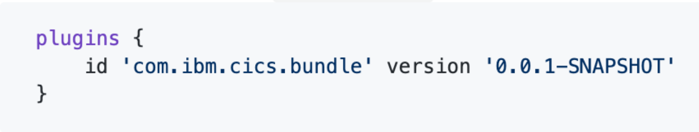
cics-bundle-gradle

- Open source on GitHub
- Development builds published to Sonatype Snapshots
- Version 0.0.2 released!
- 1.0.0 coming soon

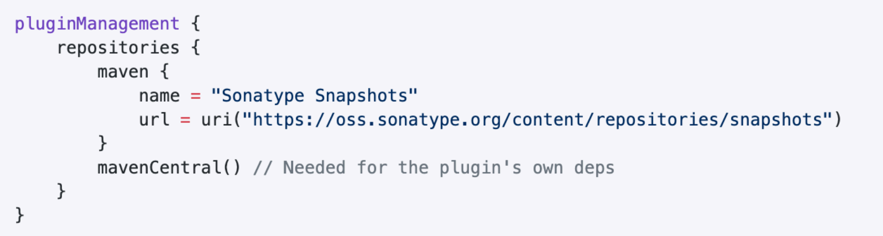
Bundle publish API

- 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>1.0.0</version>
</plugin>
```



```
plugins {
    id 'com.ibm.cics.bundle' version '0.0.1-SNAPSHOT'
}
```



```
pluginManagement {
    repositories {
        maven {
            name = "Sonatype Snapshots"
            url = uri("https://oss.sonatype.org/content/repositories/snapshots")
        }
        mavenCentral() // Needed for the plugin's own deps
    }
}
```

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!