EclipseJCDE User Guide

Last Updated: 20-Jun-2007
Author: Amgad Youssef

Table of Contents

Description

Eclipse is an extensible Integrated Development Environment (IDE) that allows developers to extend its functionalities to create new development environments, customized for their special needs. Eclipse Java Card Development Environment (EclipseJCDE) makes use of this feature by extending Eclipse to create a visual development environment for Java Card applications.

Sun Microsystems provides a Java Card Development Kit along with a set of command line tools to help developers develop Java Card applications. The steps of creating a Java Card application using command line tools are many and vulnerable to many human errors.

EclipseJCDE uses Eclipse platform to wrap the Java Card Development Kit tools and libraries provided by Sun Microsystems to provide a visual development environment that automates many of the required steps to develop a Java Card application.

EclipseJCDE provides the following features: -

Wizards

EclipseJCDE extends the wizards functionality of Eclipse to provide a set of wizards to automate the setup process of the environment of a Java Card Application. Figure 1 shows the available wizards for a Java Card application.

"Java Card Project" wizard is used to automate the creation process of a Java Card Project with all the required libraries in its environment (classpath). Figure 2 shows a screen shot for the "Java Card Project" wizard and Figure 3 shows a screen shot for a created Java Card project.

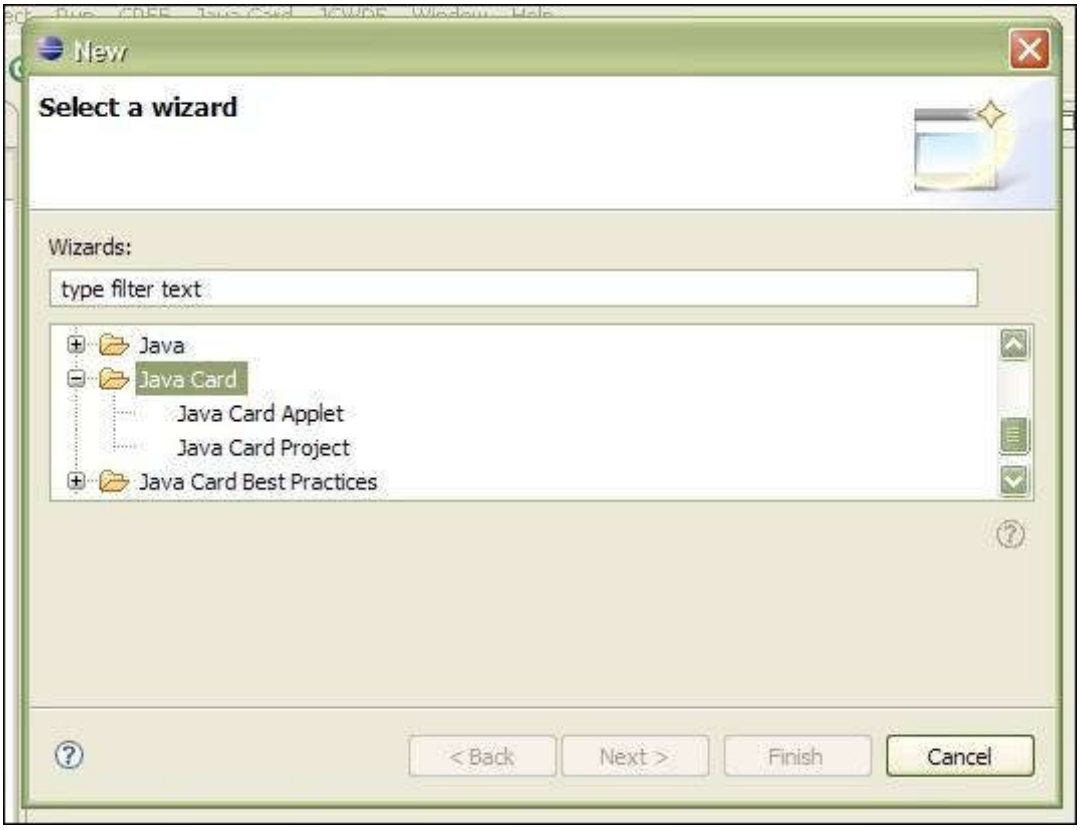

Figure 1: EclipseJCDE wizards

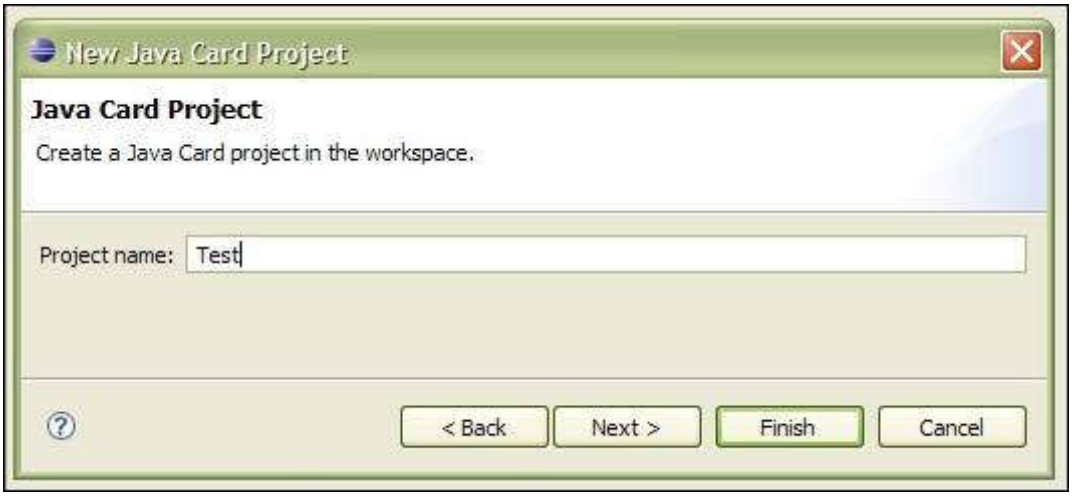

Figure 2: The "Java Card Project" wizard

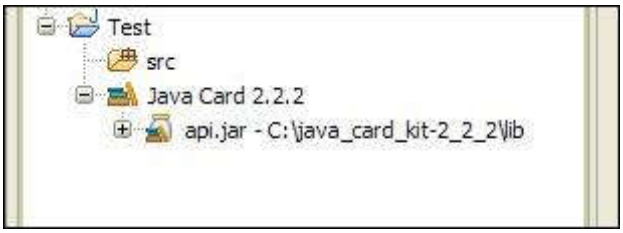

Figure 3: A created Java Card Project

"Java Card Applet" wizard is used to automate the creation process of a Java Card Applet, and to allow the java card developer to enter the applet's AID to be stored in the project's workspace. The AID is required for generating CAP files and APDU scripts, and for installing the applet on a simulator for testing.

The next step of the wizard allows adding extra features to the newly created Java Card applet. It allows adding support for Java Card RMI (Remote Method Invocation) and for Shareable Interfaces, and it allows optional generation of sample code for both features as well. Figure 4 shows a screen shot for the “Java Card Applet” wizard, and Figure 5 shows a screen shot for the extra features that could be added to a Java Card applet.

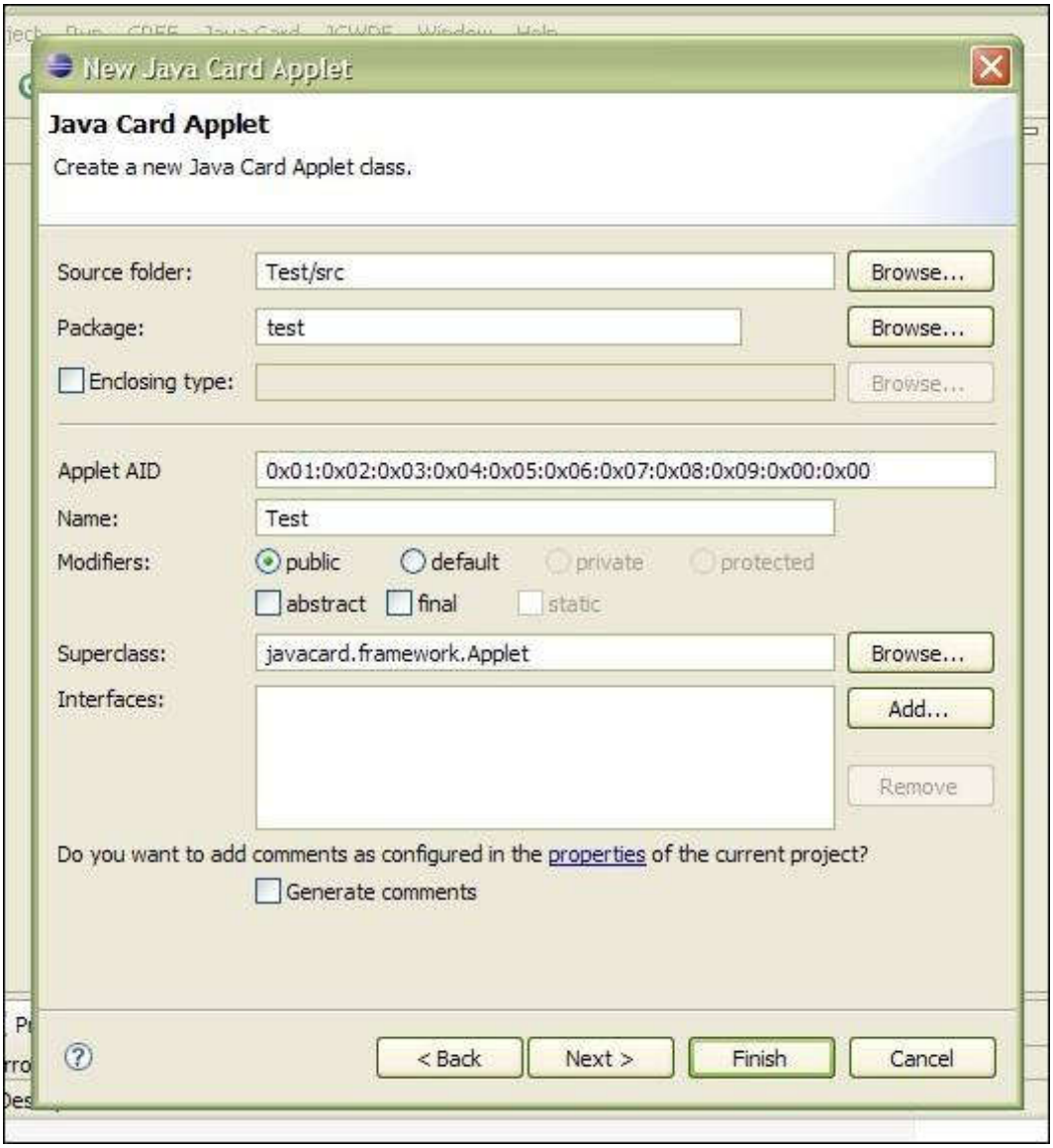

Figure 4: The “Java Card Applet” wizard

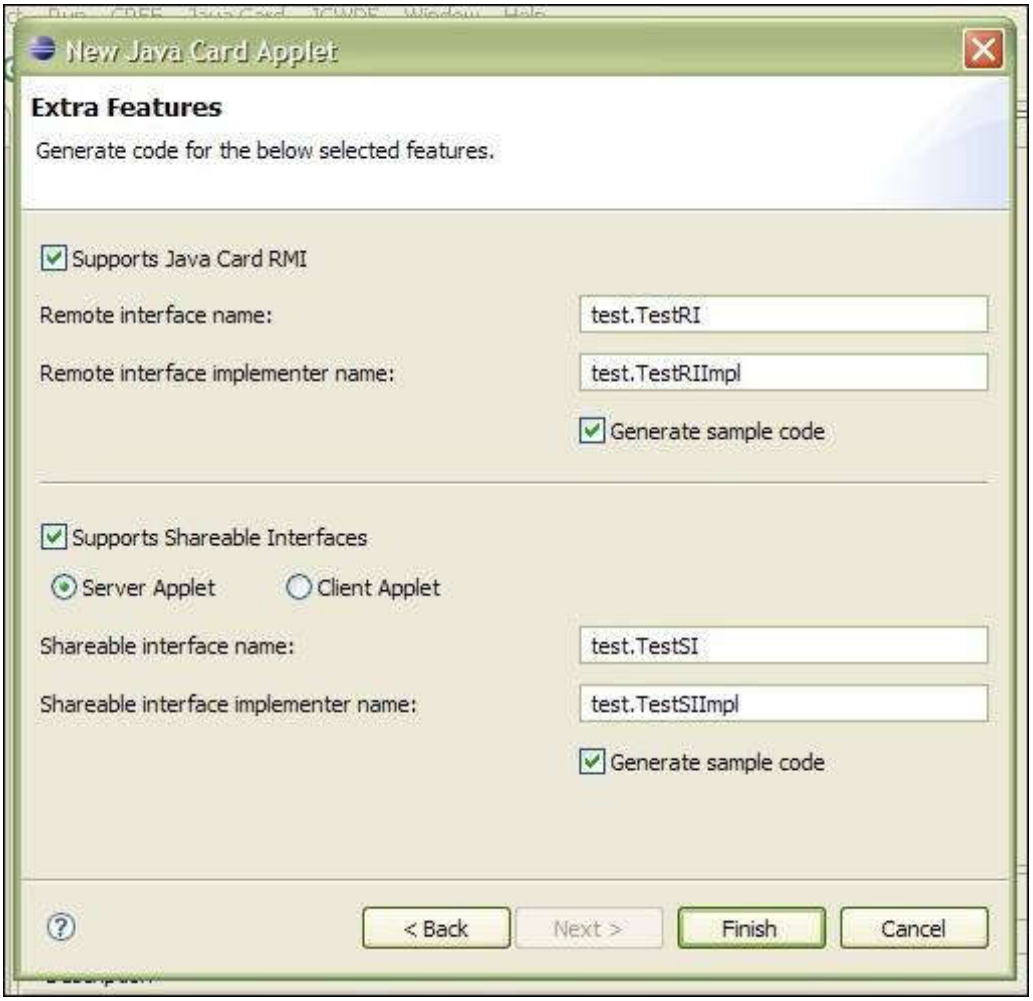

Figure 5: Java Card applet "Extra Features"

Tools

EclipseJCDE provides a set of tools to aid in the development of a Java Card application. Some of these tools wrap the command line tools provided by Sun Microsystems and automate the preparation of their arguments based on the setup of the workspace of the Java Card project.

"Set Package AID" is a tool to allow setting the AID of a selected package. The AID should be used later on while generating CAP files and APDU scripts and while installing the package on a simulator. Figure 6 shows the popup menu that appears after right clicking a package, and Figure 7 shows the dialog that appears after selecting the "Set Package AID" menu item.

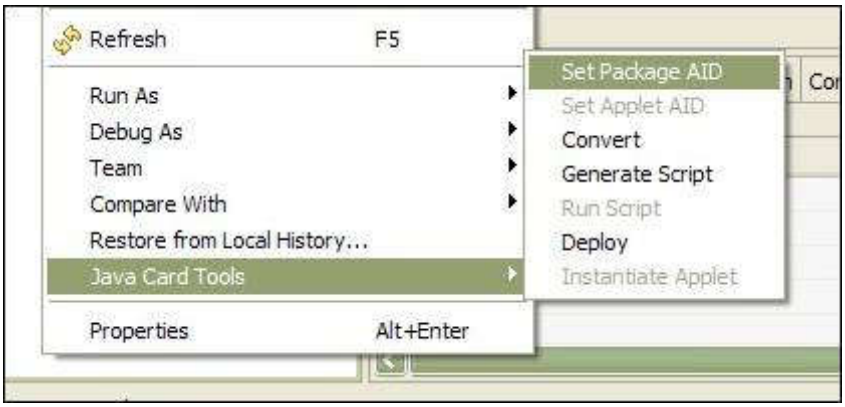

Figure 6: Set Package AID popup menu item

Figure 7: Set Package AID dialog

"Set Applet AID" is a tool to allow setting the AID of a selected Java Card applet. The AID should be used later on while generating CAP files and APDU scripts and while installing the applet on a simulator. Figure 8 shows the popup menu that appears after right clicking a Java Card applet, and Figure 9 shows the dialog that appears after selecting the "Set Applet AID" menu item.

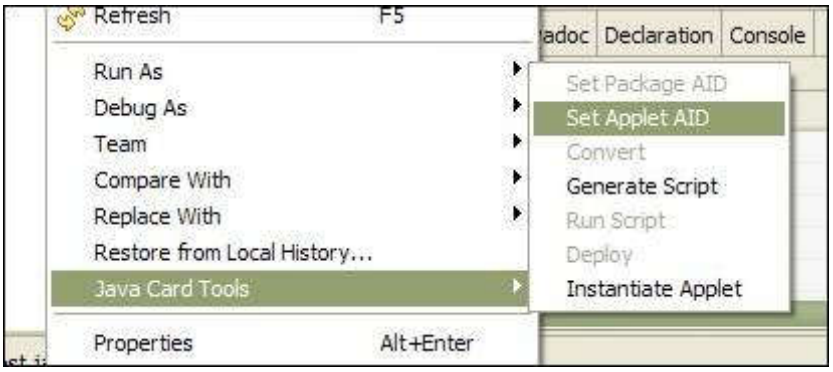

Figure 8: Set Applet AID popup menu item

Figure 9: Set Applet AID dialog

"Convert" is a tool that wraps a command line tool provided by Sun Microsystems to allow converting a Java Card package into a CAP file. The tool also optionally generates a EXP file and a JCA file. The CAP file is a JAR-format file which contains the executable

binary representation of the classes in a package. A EXP file is a Java Card export file that contains the public API linking information of classes in a package. A JCA file is a Java Card assembly file, which could be used to regenerate a CAP file [3]. Figure 10 shows the popup menu that appears after right clicking a Java Card package, and Figure 11 shows the output after selecting the Convert menu item.

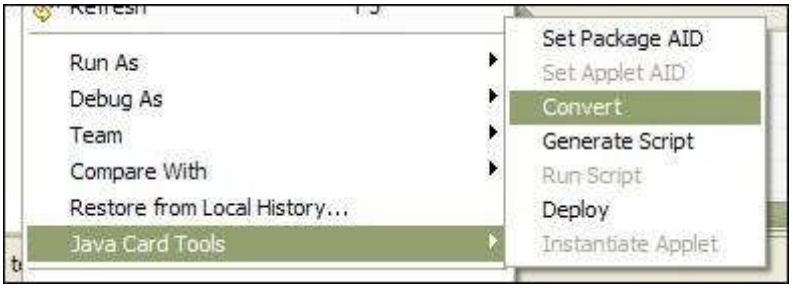

Figure 10: Convert popup menu item

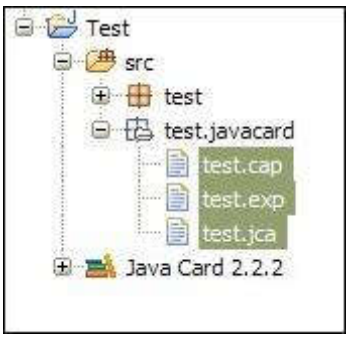

Figure 11: Convert output

"Generate Script" is a tool that wraps a command line tool provided by Sun Microsystems to allow generating Java Card package and applet installation APDU scripts. Extra APDU script lines have been automatically added to the script generated by Sun's tool to make it more feasible for direct running on a simulator. The Generate Script popup menu item appears after right clicking a Java Card package, a Java Card applet or a CAP file. Figure 12 shows the popup menu that appears after right clicking one of the previously mentioned components, and Figure 13 shows the output after selecting the Generate Script popup menu item.

Figure 12: Generate Script popup menu item

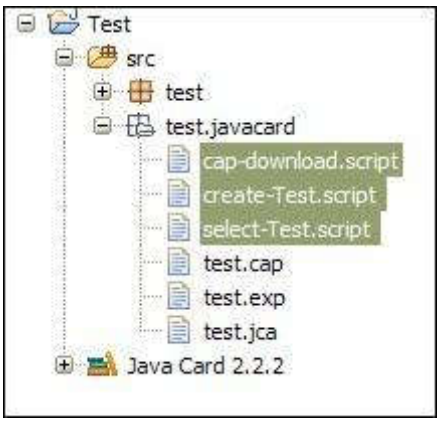

Figure 13: Generate Script output

"Run Script" is a tool that wraps a command line tool provided by Sun Microsystems to allow sending the APDU commands of an APDU script file to a simulator. Figure 14 shows the popup menu that appears after right clicking a .script file that contains APDU script.

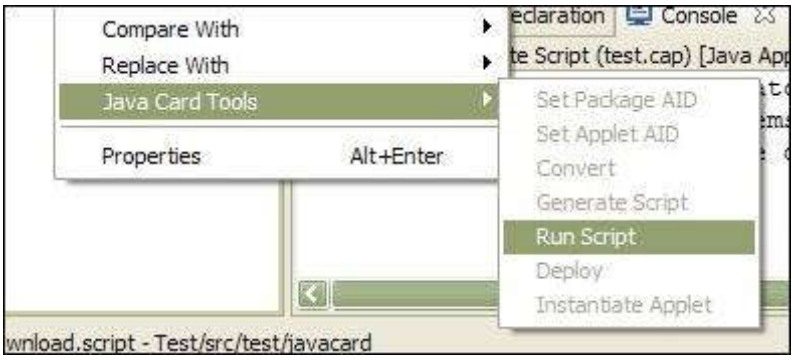

Figure 14: Run Script popup menu item

Simulators

"CREF" is a JCRE reference implementation written in C-language provided by Sun Microsystems. It is a simulator that is much like a real Java Card technology-based implementation but it does not support running Java Card applications in debug mode [3]. CREF is wrapped inside EclipseJCDE to automate the process of starting it, sending arguments to it according to the stored preferences and stopping it. Figure 15 shows the CREF menu and Figure 16 shows it preferences dialog. The preferences are stored in the project's workspace and are used to construct the arguments that will be sent to the simulator while starting.

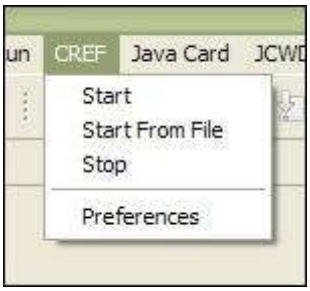

Figure 15: CREF menu

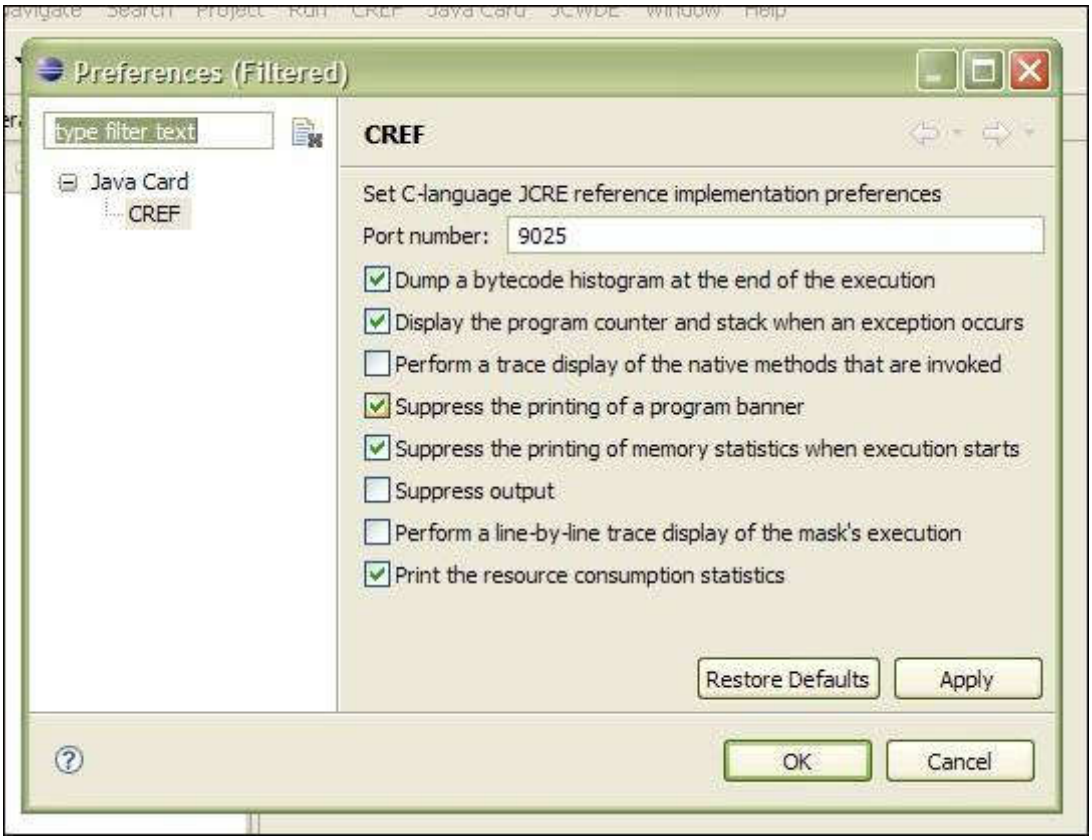

Figure 16: CREF preferences

"JCWDE" or Java Card Workstation Development Environment is another simulator provided by Sun Microsystems. It supports running Java Card applications in debug mode. It is a java application that emulates the JCRE but lacks some of its features. JCWDE is wrapped inside EclipseJCDE to automate the process of starting it in normal or debug mode, sending arguments to it according to the stored preferences and stopping it. Figure 17 shows the JCWDE menu and Figure 18 shows it preferences dialog. The preferences are stored in the project's workspace and are used to construct the arguments that will be sent to the simulator while starting.

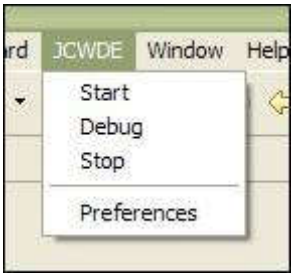

Figure 17: JCWDE menu

Figure 18: JCWDE preferences

Validations

EclipseJCDE provides a set of compile-time validations to validate the consistency of a Java Card project to avoid runtime errors while installing or running a Java Card application on a simulator.

EclipseJCDE applies the following validations: -

"Applet AID validation": Checks every Java Card applet in a Java Card project for the existence of applet AID.

"Package AID validation": Checks every package in a Java Card project for the existence of package AID.

"Duplicate AID validation": Checks every Java Card project for the inexistence of duplicate AIDs.

Figure 19 and Figure 20 show screenshots for examples of validation errors.

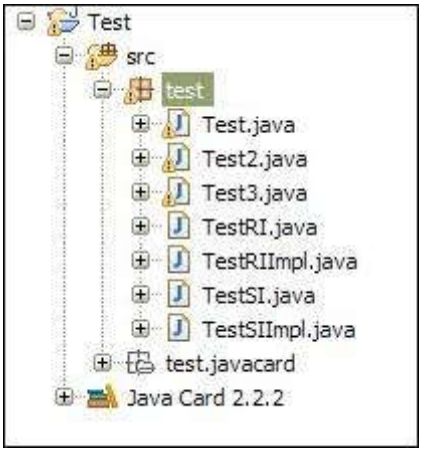

Figure 19: EclipseJCDE validation errors appearing as exclamation marks

Figure 20: EclipseJCDE validation errors appearing as warning messages