

Java™ Platform, Micro Edition Connected Device Configuration

Consumer and embedded device platforms



Highlights

- Leverages Java™ Platform, Standard Edition (Java SE, formerly J2SE™) technology
- Supports resource-constrained connected devices
- Offers CLDC/MIDP migration path
- Developed through the Java Community Process™ program



The Connected Device Configuration (CDC) technology, defined through the Java™ Specification Request (JSR) 36 and JSR 218 specifications, is a standards-based framework for building and deploying applications that can be shared across a range of network-connected consumer and embedded devices. Users benefit from the compatibility and security of Java technology. Developers benefit from the safety and productivity of the Java programming language and the rich APIs in the Java platform. And enterprises benefit from using network-based applications that extend the reach of business logic to mobile customers, partners and workers.

Goals of CDC

CDC has two principal goals:

- Support the feature sets of a broad range of connected devices while fitting within their resource constraints.
- Leverage technology skills and developer tools based on the Java Platform, Standard Edition (Java SE).

Target devices

CDC-based technology is intended for use with a broad range of resource-constrained devices such as smartphones, TV set-top boxes, telematics systems, and RFID readers. Typically, these devices include a 32-bit microprocessor/controller and require about 2 MB of RAM and 2.5 MB of ROM for the Java runtime environment.

Relationship with Java SE

Each CDC version is based on a related Java SE software version. CDC 1.1 is based on J2SE version 1.4.2.

Relationship with CLDC

The Connected Limited Device Configuration (CLDC) technology is targeted at much smaller devices than CDC. CDC includes a CLDC compatibility package to provide an upward migration path.

Elements of the Java Platform, Micro Edition (Java ME, formerly J2ME™) Technology

CDC is part of a family of standards developed in collaboration with industry leaders through the Java Community Process (JCP™). Expert groups drawn from several industries contribute to the development of standards called Java Specification Requests (JSRs).

The Java ME technology is based on three elements: a *configuration* provides the most basic set of libraries and virtual machine capabilities for a broad range of devices. A *profile* is a set of APIs that support a narrower range of devices. And an *optional package* is a set of technology-specific APIs.

A Java runtime environment can be composed of a configuration, a profile, and any number of optional packages. For example, a typical implementation could include CDC, Personal Profile, the Java Database Connectivity (JDBC™) Optional Package, and the Advanced Graphics and User Interface (AGUI) Optional Package.

CDC profiles

CDC supports three profiles:

Foundation Profile 1.1 (JSR 219)

- Core Java class library
- No GUI support
- CLDC 1.1 compatibility library

Personal Basis Profile 1.1 (JSR 217)

- Lightweight component support
- xlet support
- Foundation Profile 1.1 APIs

Personal Profile 1.1 (JSR 216)

- Full AWT support
- Applet support
- Migration path for Personal Java™ technology
- Personal Basis Profile 1.1 APIs

CDC optional packages

CDC supports several optional packages:

- The *RMI Optional Package* (JSR 66) provides an RMI subset that exposes distributed application protocols through high-level Java interfaces, classes, and method invocations.
- The *JDBC Optional Package* (JSR 169) provides a subset of the JDBC 3.0 API for accessing tabular data sources, including spreadsheets, flat files, and cross-DBMS connectivity to a wide range of SQL databases.
- The *Advanced Graphics and User Interface Optional Package* (JSR 209) provides Swing support for rich GUI components, Java2D imaging and product-specific appearances.

- *Java Secure Socket Extension* (JSSE — JSR 219), *Java Cryptography Extension* (JCE — JSR 219), and *Java Authentication and Authorization Service* (JAAS — JSR 219) provide extensions for the security architecture based on Java SE.

Application models

CDC supports different application models to give developers the flexibility to handle a range of user needs and deployment scenarios.

- *Standalone applications* support fixed-purpose designs that manage their own life cycle and resource needs.
- *Managed applications* such as applets and xlets add an application management layer that handles the tasks of deployment and resource management.

Reference implementations and technology compatibility kits

The JCP program requires development of specifications, reference implementations, and technology compatibility kits. These demonstrate the technology of a JSR and provide a verification framework for alternate implementations.

Optimized implementations

Sun develops and licenses optimized implementations of CDC technology for a variety of CPUs and operating systems. Portability interfaces enable rapid modification for new target platforms.

Learn More

Get the inside story on the trends and technologies shaping the future of computing by signing up for the Sun Inner Circle program. You'll receive a monthly newsletter packed with information, plus access to a wealth of resources. Register today at sun.com/joinic.

Development environment

CDC leverages developer tools based on the Java SE standard, including NetBeans™ technology.

For more information

To learn more about the Connected Device Configuration technology, visit java.sun.com/products/cdc

About Sun

For years, customers have turned to Sun Microsystems to help them expand their business, lower their costs, and gain competitive advantage. Sun is a leading provider of industrial-strength hardware, software, services, and technologies that make the Net work.