Caché Concepts For MultiValue Developers

Version 2018.1
2020-11-13

InterSystems Corporation 1 Memorial Drive Cambridge MA 02142 www.intersystems.com

Caché Concepts For MultiValue Developers
Caché Version 2018.1 2020-11-13

For Support questions about any InterSystems products, contact:

InterSystems Worldwide Response Center (WRC)
Tel: +1-617-621-0700
Tel: +44 (0) 844 854 2917
Email: support@InterSystems.com

Table of Contents

1

Concepts

This document describes some key Caché concepts and features. Understanding these can make it easier to work effectively in the Caché environment. They are:

- Logical Storage
- Physical Storage
- Namespaces
- High Availability
- Distributed Databases
- Programming Languages
- Studio – The Development IDE
- System Management in Caché
- Access to Data from Without
- Web-Based Development
- Internationalization Support
- Documentation

1.1 Logical Storage

Fundamentally, Caché logically stores its data in persistent, multidimensional arrays managed by an extremely efficient multidimensional data engine. These arrays are referred to as "globals" and indicate that potentially all users on the system have access to them. Each global can have a varying number of subscripts; this implies that the data in the arrays is inherently sparse. No restrictions are imposed on the type used for any subscript. Nor is there a restriction on the data stored in the array element itself. There is no data dictionary, and thus no data definitions, required to describe the multidimensional data.

It is also possible to simultaneously map the data in the globals to SQL tables or to instances of Caché classes. Thus, from a programming perspective, developers may choose to model data in the most appropriate form for the application at the time. Concurrent access, with appropriate controls, to the same data using different models is permitted, even within a single application.

All persistent data in the system — source code, object code, documentation, application data, data mappings and so on — are stored in globals.

1.2 Physical Storage

Underneath the logical representation, the multidimensional engine efficiently maps data onto operating system files. Each of these databases has the name cache.dat. Multiple databases are stored in different directories to distinguish them.

With each database Caché manages storage in 8-KB chunks and provides physical integrity guarantees for both the actual data and the metadata that organizes it. The database is automatically extended as needed.

This integrity is guaranteed even if an error occurs during writes to the database.

1.3 Namespaces

Access to code and data in Caché databases is via namespaces. A namespace is a logical entity that groups part or all of a database, or multiple databases into a single unit. Thus namespaces shield the application code from knowledge of the physical location of the data. The number of namespaces is not fixed. The same part of a database may be a member of more than one namespace.

Namespaces thus perform a function similar to MultiValue ACCOUNTS in organizing data.

1.4 High Availability

Caché provides a number of strategies that allow high availability and recoverability. These include:

- Journaling — This tracks changes made to a Caché database, for up-to-the-minute recovery after a crash or restoring your data during system recovery.
- Shadowing — Using this facility one Caché database server "shadows" another by reading the primary server's journal and applying database changes to its own copy of the database. Note that this is not true replication, but allows for emergency failover where a small amount of latency is permissible.
- Clustering — There is full support of clustering on operating systems that provide it.
- Mirroring — A Caché mirror consists of two physically independent Caché systems, called failover members, each of which maintains a copy of each database included in the mirror. Application updates are made on the primary failover member, while the backup failover member's mirrored databases are kept synchronized with the primary through application of journal files from the primary. When the primary Caché instance is hung or unavailable, the mirror fails over to the backup without interrupting application access.

1.5 Distributed Databases

Caché has a technology for distributing data and application logic and processing among multiple systems. It is called the Enterprise Cache Protocol (ECP). On a multi-server system, a network of Caché database servers can be configured as a

common resource, sharing data storage and application processing, with the data distributed seamlessly among them. This provides increased scalability as well as automatic failover and recovery.

1.6 Programming Languages

Caché supports three native programming interfaces: ObjectScript, Caché Basic, and Caché MultiValue Basic (MVBasic). All three generate identical p-code, so the programmer is free to choose the language best suited to expressing the application solution without incurring a performance penalty. All Caché features — the Studio IDE, Caché Server Pages (CSP), debugging facilities, and so on — are equally available from any interface.

1.7 Studio – The Development IDE

InterSystems provides a GUI-based integrated development environment called the Studio. Developers can use Studio to create and maintain applications used with Caché, including application components written in MVBasic. The Studio also includes interfaces to support application debugging and coordinate development actions with common source control utilities.

1.8 System Management in Caché

Caché is managed through a Web-based interface, called the Management Portal. Having a Web-based interface means that a system administrator can manage Caché from any Web browser and is independent of physical location or operating system.

1.9 Access to Data from Without

1.9.1 ODBC & JDBC

InterSystems provides an ODBC and JDBC interface to Caché, which is automatically installed along with the database. Therefore, Caché data can be accessed from any ODBC- or JDBC-enabled application.

Caché methods can be deployed as stored procedures for relational access, and Caché queries can be stored as SQL-based queries and called by a relational, ODBC or JDBC client.

1.9.2 Object Access

Caché data can also be accessed from most object technologies: Java, C and C++, .NET, Perl, Python and others. Persistent Caché classes can be deployed as Java classes, C++ classes, and the like for native access from those object technologies. The .NET capability provided by Caché is a full, managed-provider implementation. Caché also provides for the construction of database class definitions that provide persistence for an existing set of application Java classes either directly or using Hibernate.

1.10 Web-Based Development

The ability to deploy Web-based applications has become a required component in any database management system. Caché fully supports the most widely-used Web-based deployment methodologies such as Active Server Pages (ASP) and Java Server Pages (JSP).

Caché also provides its own Web deployment methodology, Caché Server Pages (CSP). CSP is tightly-bound to the Caché database, which provides increased performance and scalability compared to other Web deployment alternatives. Deploying with Caché Server Pages also permits a developer to use a single IDE for both Web-based and client-server applications (the Studio).

These deployment options allow a MultiValue developer to easily deploy data and applications to Web-based users.

Caché has built-in SOAP protocol support. This means that any Caché method — including those coded in MVBasic — can be deployed as a Web service and called from (or "consumed by") any external Web service consumer.

1.11 Internationalization Support

Caché supports both 8-bit storage and Unicode/16-bit storage. Data initially stored in 8-bit Caché can be read by a Unicode version of Caché; however, Unicode data cannot be accessed by an 8-bit Caché system.

Caché supports the concept of locales, a related set of properties that affect number and date/time formatting and text representation. With the underlying support for locales, application programmers can develop applications that are localized, that is, provide an interface adapted to the country/language where they are run.

1.12 Documentation

Caché documentation is installed with Caché in the DOCBOOK namespace. It is accessed via a browser. Assuming you installed Caché on your local system using the default Web server port, and Caché is running, you may cut and paste this link into the browser of your choice: http://localhost:8972/csp/docbook/DocBook.UI.Page.cls. Windows users may also access the documentation by choosing **Documentation** from the Cube.

Caché documentation conforms to the OASIS standard, DocBook. Users may create new material that conforms to this standard and add this content to the documentation database.

Documentation on all aspects of Caché is available in Adobe Portable Document Format (PDF) from our corporate Web site. You may download individual books and articles as well as groups of related material. A specific technical article gives the detailed specifications needed if you wish to print any or all of the documents at a commercial reproduction site.