

ARIUS SOFTWARE CORPORATION

ConnectedDB - Business Logic

Arius Software Corporation
Waterloo, ON • N2L 1T2
Phone 519.885.9045 • Fax 519.885.7123

Business Logic

Every business has unique business rules and processes. ConnectedDB uses a general purpose Business-Logic language, based on JavaScript, to specify application-specific tasks.

Business Logic is the concept of taking a general application toolkit and customizing it to perform the tasks specific to your needs. These tasks may include opening a window, creating a report, or simply highlighting data. Business logic can be used in any entity in the system, triggered by user input, data changes, or system events.

Use of Business Logic can be broken down into two broad groups; logic running on the server and logic running on the client or presentation tier. A piece of logic running on the client can be embedded in an on-screen widget, such as a Button, or it can stand alone, in which case it is referred to as a Neuron. Logic running on the server is referred to as a Cortex. Both entities function in very similar fashions; a Neuron can be converted to a Cortex by simply changing the entity type.

When Business Logic is being used to validate data entry, or to provide an interactive interface, it makes sense for the code to be in a widget or Neuron. If your Business Logic is doing database-intensive work, it will be faster if run on the server as a Cortex.

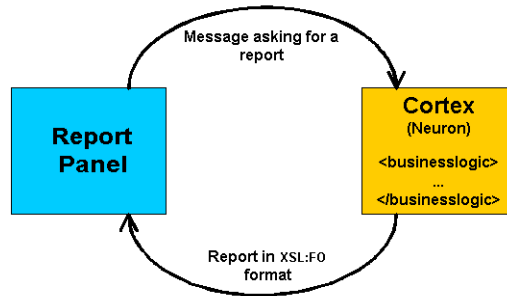
Frequently used code segments can be factored into Business Logic Objects to be used in multiple places. This makes your code easier to read and understand, and encourages code reuse across a project. ConnectedDB comes with a library of predefined functions listed in `stdfuncs.cdb`.

One task that ConnectedDB's Business Logic language is used for is creating reports. A report is a summary or analysis of selected data, which can be used to analyze, summarize, or present information in the form you need it. The report is created on a Report Panel, which simply waits for instructions in XSL:FO (*Extensible Stylesheet Language: Formatting Objects*) so that it can display or print out the report. These instructions come from the neuron or cortex, which has created them using the *businesslogic* command, as below.

XSL:FO is a standardized specification for formatting with XML. Further informational resources can be found on the Internet.

ConnectedDB - Business Logic

The Process of Creating a Report



The report panel sits idle, waiting for the cortex (or neuron) to send it the report in XSL:FO format. The cortex (or neuron) gets the information from another entity, and processes it using businesslogic to create the XSL:FO code.

When a piece of Business Logic is executed, the variables that it uses can contain several types of data: String, Integer, Date, Boolean, and ElementNode. Each of these types can be converted into Strings (for transmission over XML) and the appropriate String can then be converted back into the other types. Business Logic, though similar to Javascript, is strongly typed, so errors can be detected earlier and type conversions done explicitly.

ElementNode values have special properties. Although they can be viewed as Strings, they are assumed to contain valid XML strings and they expose an interface for modifying the XML they contain.

Because XML is so versatile, ElementNode values are used to store a variety of advanced types of data. For example, a structure containing the results of a query (a TableData) or a structure containing a message is represented as a ElementNode. *For functions to assist in manipulating XML, see the `stdFuncs.cdb` library.*

Business Logic is executed on a virtual machine (the BLVM). The business logic in the source CDB file is precompiled, where possible, in order to increase speed at runtime. If the XML is created at runtime, it will be compiled just before being run. This means that repetitive operations, especially data manipulations and calculations, will be much faster.