

Progress Master Glossary for Character

© 2001 Progress Software Corporation. All rights reserved.

Progress® software products are copyrighted and all rights are reserved by Progress Software Corporation. This manual is also copyrighted and all rights are reserved. This manual may not, in whole or in part, be copied, photocopied, translated, or reduced to any electronic medium or machine-readable form without prior consent, in writing, from Progress Software Corporation.

The information in this manual is subject to change without notice, and Progress Software Corporation assumes no responsibility for any errors that may appear in this document.

The references in this manual to specific platforms supported are subject to change.

Progress, Progress Results, Provision and WebSpeed are registered trademarks of Progress Software Corporation in the United States and other countries. Apptivity, AppServer, ProVision Plus, SmartObjects, IntelliStream, and other Progress product names are trademarks of Progress Software Corporation.

SonicMQ is a trademark of Sonic Software Corporation in the United States and other countries.

Progress Software Corporation acknowledges the use of Raster Imaging Technology copyrighted by Snowbound Software 1993-1997 and the IBM XML Parser for Java Edition.

© IBM Corporation 1998-1999. All rights reserved. U.S. Government Users Restricted Rights — Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Progress is a registered trademark of Progress Software Corporation and is used by IBM Corporation in the mark Progress/400 under license. Progress/400 AND 400® are trademarks of IBM Corporation and are used by Progress Software Corporation under license.

Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Any other trademarks and/or service marks contained herein are the property of their respective owners.

.

May 2001



Product Code: 4546
Item Number: 81112U;9.1C

Preface

Purpose

The purpose of this book is to provide a glossary of terms for Progress.

Audience

This book is intended for all readers of Progress documentation.

Organization of This Manual

This book contains an alphabetical list of terms and definitions used in Progress documentation.

Typographical Conventions

This manual uses the following typographical conventions:

- **Bold typeface** indicates:
  - Commands or characters that the user types
  - That a word carries particular weight or emphasis
- *Italic typeface* indicates:
  - Progress variable information that the user supplies
  - New terms
  - Titles of complete publications

- `Monospaced typeface` indicates:
  - Code examples
  - System output
  - Operating system filenames and pathnames

The following typographical conventions are used to represent keystrokes:

- Small capitals are used for Progress key functions and generic keyboard keys.

  END-ERROR, **GET**, **GO**
  ALT, **CTRL**, **SPACEBAR**, **TAB**

- When you have to press a combination of keys, they are joined by a dash. You press and hold down the first key, then press the second key.

  CTRL-X

- When you have to press and release one key, then press another key, the key names are separated with a space.

  ESCAPE H
  ESCAPE CURSOR-LEFT

Other Useful Documentation

This section lists Progress Software Corporation documentation that you might find useful. Unless otherwise specified, these manuals support both Windows and Character platforms and are provided in electronic documentation format on CD-ROM.

Getting Started

Progress Electronic Documentation Installation and Configuration Guide (Hard copy only)

A booklet that describes how to install the Progress EDOC viewer and collection on UNIX and Windows.

Progress Installation and Configuration Guide Version 9 for UNIX

A manual that describes how to install and set up Progress Version 9.1 for the UNIX operating system.

Progress Installation and Configuration Guide Version 9 for Windows

A manual that describes how to install and set up Progress Version 9.1 for all supported Windows and Citrix MetaFrame operating systems.

Progress Version 9 Product Update Bulletin

A guide that provides a brief description of each new feature of the release. The booklet also explains where to find more detailed information in the documentation set about each new feature.

Progress Application Development Environment — Getting Started (Windows only)

A practical guide to graphical application development within the Progress Application Development Environment (ADE). This guide includes an overview of the ADE and its tools, an overview of Progress SmartObject technology, and tutorials and exercises that help you better understand SmartObject technology and how to use the ADE to develop applications.

Progress Language Tutorial for Windows and *Progress Language Tutorial for Character*

Platform-specific tutorials designed for new Progress users. The tutorials use a step-by-step approach to explore the Progress application development environment using the 4GL.

Progress Startup Command and Parameter Reference

A reference manual that describes the Progress startup commands and parameters in alphabetical order.

Welcome to Progress (Hard copy only)

A booklet that explains how Progress software and media are packaged. An icon-based map groups the documentation by functionality, providing an overall view of the documentation set. *Welcome to Progress* also provides descriptions of the various services Progress Software Corporation offers.

Development Tools

Progress ADM 2 Guide

A guide to using the Application Development Model, Version 2 (ADM 2) application architecture to develop Progress applications. It includes instructions for building and using Progress SmartObjects.

Progress ADM 2 Reference

A reference for the Application Development Model, Version 2 (ADM 2) application. It includes descriptions of ADM 2 functions and procedures.

Progress AppBuilder Developer's Guide (Windows only)

A programmer's guide to using the Progress AppBuilder visual layout editor. AppBuilder is a Rapid Application Development (RAD) tool that can significantly reduce the time and effort required to create Progress applications.

Progress Basic Database Tools (Character only; information for Windows is in online help)

A guide for the Progress Database Administration tools, such as the Data Dictionary.

Progress Basic Development Tools (Character only; information for Windows is in online help)

A guide for the Progress development toolset, including the Progress Procedure Editor and the Application Compiler.

Progress Debugger Guide

A guide for the Progress Application Debugger. The Debugger helps you trace and correct programming errors by allowing you to monitor and modify procedure execution as it happens.

Progress Help Development Guide (Windows only)

A guide that describes how to develop and integrate an online help system for a Progress application.

Progress Translation Manager Guide (Windows only)

A guide that describes how to use the Progress Translation Manager tool to manage the entire process of translating the text phrases in Progress applications.

Progress Visual Translator Guide (Windows only)

A guide that describes how to use the Progress Visual Translator tool to translate text phrases from procedures into one or more spoken languages.

Reporting Tools

Progress Report Builder Deployment Guide (Windows only)

An administration and development guide for generating Report Builder reports using the Progress Report Engine.

Progress Report Builder Tutorial (Windows only)

A tutorial that provides step-by-step instructions for creating eight sample Report Builder reports.

Progress Report Builder User's Guide (Windows only)

A guide for generating reports with the Progress Report Builder.

Progress Results Administration and Development Guide (Windows only)

A guide for system administrators that describes how to set up and maintain the Results product in a graphical environment. This guide also describes how to program, customize, and package Results with your own products. In addition, it describes how to convert character-based Results applications to graphical Results applications.

Progress Results User's Guide for Windows and *Progress Results User's Guide for UNIX*

Platform-specific guides for users with little or no programming experience that explain how to query, report, and update information with Results. Each guide also helps advanced users and application developers customize and integrate Results into their own applications.

4GL

Building Distributed Applications Using the Progress AppServer

A guide that provides comprehensive information about building and implementing distributed applications using the Progress AppServer. Topics include basic product information and terminology, design options and issues, setup and maintenance considerations, 4GL programming details, and remote debugging.

Progress External Program Interfaces

A guide to accessing non-Progress applications from Progress. This guide describes how to use system clipboards, UNIX named pipes, Windows dynamic link libraries, Windows dynamic data exchange, Windows ActiveX controls, and the Progress Host Language Call Interface to communicate with non-Progress applications and extend Progress functionality.

Progress Internationalization Guide

A guide to developing Progress applications for markets worldwide. The guide covers both internationalization—writing an application so that it adapts readily to different locales (languages, cultures, or regions)—and localization—adapting an application to different locales.

Progress Language Reference

A three-volume reference set that contains extensive descriptions and examples for each statement, phrase, function, operator, widget, attribute, method, and event in the Progress language.

Progress Programming Handbook

A two-volume handbook that details advanced Progress programming techniques.

Database

Progress Database Design Guide

A guide that uses a sample database and the Progress Data Dictionary to illustrate the fundamental principles of relational database design. Topics include relationships, normalization, indexing, and database triggers.

Progress Database Administration Guide and Reference

This guide describes Progress database administration concepts and procedures. The procedures allow you to create and maintain your Progress databases and manage their performance.

DataServers

Progress DataServer Guides

These guides describe how to use the DataServers to access non-Progress databases. They provide instructions for building the DataServer modules, a discussion of programming considerations, and a tutorial. Each DataServer has its own guide, for example, the *Progress DataServer for ODBC Guide*, the *Progress DataServer for ORACLE Guide*, or the *Progress/400 Product Guide*.

MERANT ODBC Branded Driver Reference

The Enterprise DataServer for ODBC includes MERANT ODBC drivers for all the supported data sources. For configuration information, see the MERANT documentation, which is available as a PDF file in *installation-path*\odbc. To read this file you must have the Adobe Acrobat Reader Version 3.1 or higher installed on your system. If you do not have the Adobe Acrobat Reader, you can download it from the Adobe Web site at: http://www.adobe.com/prodindex/acrobat/readstep.html.

SQL-89/Open Access

Progress Embedded SQL-89 Guide and Reference

A guide to Progress Embedded SQL-89 for C, including step-by-step instructions on building ESQL-89 applications and reference information on all Embedded SQL-89 Preprocessor statements and supporting function calls. This guide also describes the relationship between ESQL-89 and the ANSI standards upon which it is based.

Progress Open Client Developer's Guide

A guide that describes how to write and deploy Java and ActiveX applications that run as clients of the Progress AppServer. The guide includes information about how to expose the AppServer as a set of Java classes or as an ActiveX server.

Progress SQL-89 Guide and Reference

A user guide and reference for programmers who use interactive Progress/SQL-89. It includes information on all supported SQL-89 statements, SQL-89 Data Manipulation Language components, SQL-89 Data Definition Language components, and supported Progress functions.

SQL-92

Progress Embedded SQL-92 Guide and Reference

A guide to Progress Embedded SQL-92 for C, including step-by-step instructions for building ESQL-92 applications and reference information about all Embedded SQL-92 Preprocessor statements and supporting function calls. This guide also describes the relationship between ESQL-92 and the ANSI standards upon which it is based.

Progress JDBC Driver Guide

A guide to the Java Database Connectivity (JDBC) interface and the Progress SQL-92 JDBC driver. It describes how to set up and use the driver and details the driver's support for the JDBC interface.

Progress ODBC Driver Guide

A guide to the ODBC interface and the Progress SQL-92 ODBC driver. It describes how to set up and use the driver and details the driver's support for the ODBC interface.

Progress SQL-92 Guide and Reference

A user guide and reference for programmers who use Progress SQL-92. It includes information on all supported SQL-92 statements, SQL-92 Data Manipulation Language components, SQL-92 Data Definition Language components, and Progress functions. The guide describes how to use the Progress SQL-92 Java classes and how to create and use Java stored procedures and triggers.

Deployment

Progress Client Deployment Guide

A guide that describes the client deployment process and application administration concepts and procedures.

Progress Developer's Toolkit

A guide to using the Developer's Toolkit. This guide describes the advantages and disadvantages of different strategies for deploying Progress applications and explains how you can use the Toolkit to deploy applications with your selected strategy.

Progress Portability Guide

A guide that explains how to use the Progress toolset to build applications that are portable across all supported operating systems, user interfaces, and databases, following the Progress programming model.

WebSpeed

Getting Started with WebSpeed

Provides an introduction to the WebSpeed Workshop tools for creating Web applications. It introduces you to all the components of the WebSpeed Workshop and takes you through the process of creating your own Intranet application.

WebSpeed Installation and Configuration Guide

Provides instructions for installing WebSpeed on Windows and UNIX systems. It also discusses designing WebSpeed environments, configuring WebSpeed Brokers, WebSpeed Agents, and the NameServer, and connecting to a variety of data sources.

WebSpeed Developer's Guide

Provides a complete overview of WebSpeed and the guidance necessary to develop and deploy WebSpeed applications on the Web.

WebSpeed Version 3 Product Update Bulletin

A booklet that provides a brief description of each new feature of the release. The booklet also explains where to find more detailed information in the documentation set about each new feature.

Welcome to WebSpeed! (Hard copy only)

A booklet that explains how WebSpeed software and media are packaged. *Welcome to WebSpeed!* also provides descriptions of the various services Progress Software Corporation offers.

Reference

Pocket Progress (Hard copy only)

A reference that lets you quickly look up information about the Progress language or programming environment.

Pocket WebSpeed (Hard copy only)

A reference that lets you quickly look up information about the SpeedScript language or the WebSpeed programming environment.

Glossary

Numbers

4GL

Acronym for Fourth-Generation Language. The term used for Progress Software Corporation's application programming language.

A

Abbreviated Index

An index that can be searched through using the first few characters of a field, if the field is a character data field. Indexes not composed of character data require an exact match.

Accelerator Key

A function key or a combination of keys that you can press to access a menu item's functionality instead of choosing the menu item itself.

Active Database

The database that is currently in use. If you have more than one database, the Data Dictionary requires you to select an active, working database.

Active File

The file that is currently in use.

Active Index

An index that can be used to retrieve or to order records. If you are defining an index in a very large table, it is more efficient to define the index as initially inactive. You can subsequently activate the index by running PROUTIL with the IDXBUILD option.

Active Procedures

The executing procedure and all procedures on the stack.

Active Set

The collection of rows that SQL identifies when it opens a cursor and executes the query associated with the cursor. See also **Result Set**.

Active Window

The window that currently has input focus and that you can reference using the ACTIVE-WINDOW system handle. See also **Current Window**.

ActiveX

Microsoft's registered trademark for the third version of Object Linking and Embedding (OLE) controls. An umbrella of technologies developed by Microsoft that is built on the foundation provided by the Component Object Model (COM). COM defines a standard for providing software services that works across applications, system software, and networks. The main objective of these technologies is to facilitate better software development. Progress relies on COM standards to support Automation objects and ActiveX controls. See also **COM**, **COM Object**.

ActiveX Control

A reusable component built on the Microsoft Component Object Model (COM) that allows you to extend the user interface and functionality of a Progress application. Some ActiveX controls (also known as OCXs) are similar to 4GL widgets, such as combo boxes and radio sets. Others provide functionality, such as communications and time keeping, that has no user interface component. See also **ActiveX**, **COM**.

ActiveX Control Container

The COM object supported by Progress that holds an ActiveX control, enables the definition of its event handlers, and manages some aspects of its user interface. This control container is instantiated as part of the control-frame widget. See also **COM Object**.

Add [an ODBC data source]

Makes a data source available to ODBC through the Add operation of the ODBC Administrator. Adding a data source tells ODBC where a specific database resides and which ODBC driver to use to access it. Adding a data source also invokes a setup dialog box for the particular driver so you can provide other details the driver needs to connect to the database.

After-image Block

The basic unit of I/O in the after-image log. The Progress database manager reads and writes the after-image log in blocks.

After-image File

A file that is used to store a portion of the redo or after-image log of a Progress database.

After-image Log

A log of all database changes kept by the Progress database manager. If a database is damaged by a failure, it can be recovered by restoring a backup and rolling forward through the after-image log, repeating all the database changes up to the point of failure, up to a specified point in time, or up to a particular transaction ID. See also **Before-image Log**, **Roll-forward Recovery**.

After-image Note

A picture of the record after a transaction is committed.

After-image Writer (AIW)

A background process that the Progress database manager uses to write data to the after-image log on behalf of servers. See also **Asynchronous Page Writer (AIW)**, **Before-image Writer (BIW)**.

After-imaging

An optional feature of the Progress database. It protects from media loss by storing after-image notes in an after-image file. The after-image file must reside on a disk other than the disk on which the database resides. Use the after-image file in conjunction with the most recent backup to reconstruct the database. See also **Roll-forward Recovery**.

Aggregate Field

A user-defined field that computes summary information about a set of records in the report.

Aggregate Phrase

A Progress language element that identifies one or more values to be calculated based on a change in a break group.

AIW

See **After-image Writer (AIW)**.

Alert Box

A window that appears on the screen requesting user response. Typically, an alert box asks a question, or presents information (such as an error message) to the user. The user must dismiss the alert box before the application can continue.

Alias

1) A synonym for a logical database name used within a Progress procedure to reference a database. An alias is used as a database reference in general-purpose Progress procedures in place of a logical database name.

2) A temporary name (also called Correlation Name) for a table or column specified in the FROM clause of an SQL query expression. Derived tables and search conditions that join a table with itself must specify an alias. Once a query specifies an alias, references to the table or column must use the alias and not the underlying table or column name. See also **Correlation Name**.

ALink

A hypertext link that locates target topics based on A-keywords (keywords stored in A footnotes) rather than context strings. ALinks are typically used to find a set of related topics, and are often associated with a Related Topics button in a help topic.

Alternate Source Language

A language defined in the current translation database. You can display translation strings from an alternate source language in the Alt Source field of the Translation Manager.

API

See **Application Program Interface (API)**.

AppBuilder

A window-based tool you can use to build the user interface of an application and attach triggers to widgets.

AppBuilder Procedure File (.w file)

A special class of Progress procedure file generated when you create and save a Progress window or dialog box with the AppBuilder. AppBuilder procedure files must have a .w extension in their filenames.

Applet

A special kind of Java program whose compiled class files a Java-enabled browser can download from the Internet and run.

Application

A set of procedures that perform a set of interrelated tasks. An application typically consists of a startup procedure and additional procedures that can be run from the startup procedure.

Application Broker

The process running on an AppServer that manages connections from client applications and the Application Server processes that service them.

Application Compiler

A tool used to compile Progress procedure files (.p and .w files) either for the duration of a Progress session or for permanent storage. You can also use the Application Compiler to create a variety of listings and to compile procedure files in a different language or with an encryption key.

Application Context

Any currently displayed user-interface element that has input focus, for example, a window, dialog box, browse, button, combo box, control frame, editor, fill in, radio set, selection list, slider, toggle bar, or field. The application context can also be a display state or a step in an application procedure.The application context is the basis for context-sensitive help. See also **Context Number**, **Context-sensitive Help**, **Context ID**, **Context String**.

Application Debugger

In Progress, a software tool that allows you to control application execution by starting and stopping at any line in the 4GL, and allows you to monitor execution by viewing 4GL code and data as the application executes.

Application Partitioning

The process of segmenting application logic from the user interface and data, and running it across multiple computers on a local or wide area network.

Application Program Interface (API)

A term for the specification of the number and types of parameters, return values, and semantics of the externally called functions of an application, module, or program. Also known as calling sequences.

Application Security

Protects your database by allowing the user to access only the procedures or database objects that they are authorized to access.

Application Server

The process running on an AppServer that executes 4GL procedures in response to remote procedure requests. An AppServer can have any number of Application Server processes to service remote procedure requests.

AppServer Session

The 4GL session running in an Application Server.

APW

See **Asynchronous Page Writer (APW)**.

Argument

1) A piece of data that a calling procedure passes to a called procedure source file or to an include file. Progress evaluates the passed data at compile time.

2) A run-time value passed as a procedure or method parameter.

Array

A field or variable with multiple elements of the same data type, each of which can be referenced by a numerical position.

Ascending Index

An index whose keys are sorted in ascending order.

ASCII

A seven-bit code representing uppercase and lowercase letters, numbers, symbols, and punctuation marks. ASCII is an acronym for the American Standard Code for Information Interchange.

Assignment Operator

The equal sign (=) when used to assign a value to an element, such as a field, variable, or attribute. See also **Relational Operator**.

Asynchronous

Lacking regular timing, not occurring at regular intervals. Most higher-level human activities are asynchronous.

Asynchronous Checkpoint

A checkpointing technique used by the Progress database manager to allow database changes to continue during the checkpoint operation. See also **Synchronous Checkpoint**.

Asynchronous I/O

A mechanism provided in some operating systems to allow a program to initiate a read or write operation and continue processing while the data transfer takes place. The operating system notifies the program when the operation is complete. In UNIX systems, asynchronous I/O is often implemented for raw disk partitions but not for files.

Asynchronous Page Writer (APW)

A background process (or group of processes) that the Progress database manager uses to write dirty pages to stable storage. They free the servers from this task, enabling them to do more work on behalf of clients. See also **After-image Writer (AIW)**, **Before-image Writer (BIW)**, **Dirty Page**.

Asynchronous Replication

Asynchronous replication (also known as store and forward replication) allows the replication to occur outside the scope of the original transaction. The replication might take place seconds, minutes, hours, or days from the time of the transaction, depending on your business requirements. Although the replication executes record by record, replication can occur by transaction. That is, if an order is placed in the system with order lines containing multiple data changes, and these changes are made within the scope of a single transaction, the changes can be replicated as a single transaction.

Atomic

A term used to refer to inseparability, an atomic operation is one that must be done all at once.

Atomicity

The property that allows all database changes performed by a transaction to be made permanently (if the transaction is committed) or not to be made at all (if the transaction is rolled back). See also **Transaction**.

Attribute

A value associated with an object, such as a widget or system handle. The value represents an aspect (for example, size or color), state (for example, visibility), or capability (for example, to allow scrolling). Attributes are always referenced in the Progress 4GL using the same format: a widget (or system handle reference), followed by a colon, followed by the name of the attribute.

Authentication

The process of verifying a user's identity. See also **Authorization**.

Authorization

The process of verifying that a user has permission to perform a function or operation or has permission to use a service. See also **Authentication**.

Auto-connect

A Progress feature that uses information stored in a primary application database to connect to a second application database. This occurs when data from the second database is referenced in a compiled application at run time. The primary application database contains the database connection information for the second database. It must already be connected before Progress can execute the auto-connect for the second application database. Progress executes an auto-connect immediately prior to running a precompiled procedure that accesses a database on the auto-connect list.

Auto-connect List

A list of Progress databases (with connection parameters) that connect automatically, as required, during program execution.

Auto-indent

A Procedure Editor option. When you enter a return, the cursor returns to the indenting level of the previous line.

Automation Object

A package of functionality provided by one application for use by other applications in the form of COM objects. The applications providing the objects act as Automation Servers. Progress can access these objects as an Automation Controller. In effect, Progress can drive the Automation Server application to the extent supported by its Automation objects (also known as ActiveX Automation Objects). See also **COM Object**, **ActiveX Control**.

B

B-tree (Balance Tree)

A type of hierarchical data structure that provides keyed and key-sequenced access to data. B-trees are useful because they are self-balancing (they even out the distribution of keys in relation to data). Also, B-trees allow rapid location of the desired data and provide excellent performance, even when there are many entries. Progress indexes are B-trees.

Basic Database

A database that contains the schema for your application and perhaps some control data, but is otherwise empty.

Basic Object

A 4GL object; a user-interface element defined in the 4GL itself. Basic objects are distinct both from SmartObjects and from ActiveX objects.

Before-image Block

The basic unit of I/O in the before-image log. The Progress database manager reads and writes the before-image log in blocks.

Before-image Cluster

The Progress database manager's before-image log is divided into fixed-size units of storage space called clusters. When the before-image log must be expanded, it is expanded one cluster at a time. The cluster size can be changed when the before-image log is truncated. Clusters are reused when the data they contain is no longer needed for transaction rollback or crash recovery. Whenever a cluster is filled, an asynchronous checkpoint is initiated. See also **Asynchronous Checkpoint**, **Before-image Log**.

Before-image Extent

A file or disk partition that is used to store a portion of the before-image log of a Progress database. See also **Before-image Log**.

Before-image File

1) A file that ensures database integrity by keeping track of the status of your database in the event that a transaction is not completed or a system failure occurs. Unless you explicitly place this file somewhere else with the Before Image File Name (-g) startup parameter, the BI file is located in the same directory as your DB file. You can think of the before-image file as a pretransaction image of the database records your application is modifying. See also **After-image File**.

2) A file that is used to store a portion of the before-image log for a Progress database. When the before-image log is composed of multiple extents, these extents are sometimes collectively referred to as the before-image file. See also **After-image File**.

Before-image Log

A log of all recent database changes kept by the Progress database manager. The information in this log is used to undo a transaction when the UNDO operation is executed and also to perform crash recovery after a failure has occurred. Each change to a database page is recorded in the before-image log. See also **After-image Log**.

Before-image Writer (BIW)

A background process that the Progress database manager uses to write data to the before-image file on behalf of servers. It frees servers from this task, enabling them to do more work on behalf of clients. See also **After-image Writer (AIW)**, **Asynchronous Page Writer (APW)**.

Before-imaging

An automatic feature for a Progress database. It protects your database from system failure (crash-recovery) by storing dynamic transaction back-out information. However, it does not protect systems from media loss. See also **After-imaging**.

BIW

See **Before-image Writer (BIW)**.

Blank User ID

A user ID that consists of an empty string (" "). This is the default user ID on Windows. You can optionally disallow access by the blank user ID.

Block

A series of statements that Progress treats as a single unit. Each block, except the default procedure block, begins with a block-header statement and concludes with an END statement. Progress uses five kinds of blocks: FOR EACH, REPEAT, DO, trigger, and procedure. A block can have a number of properties, including looping, record reading, clearing, transaction recovery, error processing, and default screen design. Also called a **Procedure Block**.

Block Header

The statement that begins a block. It is different from other kinds of statements in two ways: it can end with a colon or period and it can have a label.

Block Label

An identifying name applied to a block. You can use the block label to identify a block to be undone, retried, etc.

Box

In a frame, the four lines around the outside of the frame that designate its boundary. By default, frames have boxes; however, the application can turn off frame boxes.

Box Select

A means of selecting more than one widget in a frame by drawing a box around them with your mouse. Also available to application users if widgets are enabled to do so.

Break Group

A set of records that have a common value in a certain database field. Break groups are used in reports to display file and record relationships.

Break Interrupt

The Debugger function that interrupts a running procedure and gives immediate control to the Debugger. It also purges the Debugger command queue of any commands waiting to execute.

Breakpoint

A specific line in a procedure where you want execution to temporarily stop. When the procedure reaches a breakpoint, it halts before executing the specified line and gives control to the Debugger for your next command.

Breakpoint Indicator

A Debugger tool panel listing that shows each line where a breakpoint is set. The breakpoint indicator is an octagon in Windows. The indicator is placed to the left of the line number for each line where a breakpoint is set.

Breakpoint Number

A unique number assigned to each breakpoint set during a debugging session.

Broker

On shared memory systems, the main database server process. The broker manages the resources shared by multiple users. If there are remote users, the broker starts servers to service them.

Browse Sequence

A series of related help topics designed for the user to access in a sequential order. When viewing a help topic defined as part of a browse sequence, the Browse buttons (<< and >>) are active, and the user can click them to see the previous or next topic. You define browse sequences in help topics with the plus sign (+) control code. Each help topic in a browse sequence has a sequence name and a unique sequence number. See also **Control Code**, **Sequence Name**, **Sequence Number**.

Browse Widget

Displays the results of a database query. There are two types of browse widgets: updateable and read-only. An updateable browse widget lets the user view, update, add, and delete data in records. A read-only browse widget is used for display only. It shares properties of a container and field-level widget.

Buffer

A multi-byte block of memory used as a temporary storage area for data during input or output operations. See also **Data Buffer**, **Database Buffer**, **Record Buffer**, **Screen Buffer**.

Buffer Cache

A set of buffers used by the Progress database manager to hold copies of database blocks for later, if needed. The size of this buffer cache has a noticeable effect on system performance.

Button

A field-level widget that the user can choose to direct an application to execute a trigger or control the interface.

Button Definition

In the Debugger, a series of commands that you can invoke by choosing an associated button in the Debugger window. You can review all current button definitions using the SHOW BUTTONS command.

Button Panel

The area of the Debugger window where buttons are displayed.

Bytecode

Machine-independent code generated by the Java compiler and executed by the Java interpreter.

C

Calculated Field

A field that you can derive from the values of other fields or variables.

Calling Sequences

See **Application Program Interface (API)**.

Candidate Key

Another term for unique key.

Captive User

A user that cannot access the operating system upon exit from a Progress session. A captive user has a startup file that starts a Progress procedure at login and logs the user off the operating system at the completion of the Progress procedure, preventing access to the operating system.

Cardinality

The number of rows in a result table.

Cascading Replication

A replication scheme where a source database is replicated to a target, and from that target to other target databases, in a cascading fashion. In a common implementation of cascading replication, external data is received at one database, then replicated in a cascading series to other databases.

Cartesian Product

Also called cross-product. In a query expression, the result table generated when a FROM clause lists more than one table but specifies no join conditions. In such a case, the result table is formed by combining every row of every table with all other rows in all tables. Typically, Cartesian products are not useful and are slow to process.

Cell

A basic unit of storage. The storage for one unit of information, usually one character, one byte, one word, or one file.

Character Client

A combination of hardware and software components that support a character-based interface. The following platforms support Progress running in character mode: UNIX, Windows NT, and Windows 95. Windows NT and Windows 95 only support the character mode by enabling a character client to run only when DOS is running under one of these 32-bit Windows operating systems.

Character Constant

A string of characters enclosed in quotation marks that expresses a value of the CHARACTER data type.

Character Coordinates

The basic unit in a system of coordinates Progress uses to map a precise location on a display terminal. Character coordinates are in decimal form, one representing a horizontal location (column), the other a vertical location (row).

Character Data Type

A string of characters from the current code page that can include both printable and nonprintable values. This string can vary in length from 0 (empty string, "") to 256 characters or a maximum size that depends on the underlying storage.

Character Field

A field that has a CHARACTER data type; an alphanumeric field that cannot be arithmetically manipulated. Although Progress allows character fields of up to 255 characters, you typically want to restrict the format length of a character field to the input/output line width of your terminal.

Character Set

A specific collection of alphabetic, numeric, and graphic characters that are related to one another as defined by a standards organization, operating system, or equipment vendor. Character sets also include communication and printer control codes — non printing characters like backspace, tab, and keyboard characters.

Character Unit

A variable and portable unit whose size is based on the current font and resolution of the display monitor. It is the unit of measure for character coordinates.

Checkpoint

A process by which the in-memory and on-disk state of the database are reconciled. As transactions are executed, changes are made to copies of parts of the database that have been brought into volatile memory. The version that is on disk becomes progressively more obsolete. During a checkpoint, all memory-resident database changes are written to stable storage, making the volatile and stabile copies consistent. The Progress database manager initiates a checkpoint each time a before-image cluster is filled. See also **Asynchronous Checkpoint**, **Before-image Cluster**, **Write-ahead Logging (WAL)**.

Checksum

An error-detection code derived by adding the bytes of a message together and appending the sum of the bytes to the message. The recipient performs the same computation as the sender and compares the result with the transmitted checksum. This technique can be used to detect single-bit errors in data transmission.

Child Frame

A frame parented to (contained by) another frame. See also **Parent Frame**.

Child Window

A window parented to (owned by) another window. See also **Parent Window**.

Child Window Family

A child window and all of its descendant windows.

Choose

A user action to activate a button widget or a menu item. There are several ways a user can choose. For example, the user can move the mouse pointer over a button and click the SELECT mouse button. This chooses the button.

Class

The term used, specifically in object-oriented programming, to describe an abstract object; a synonym for type or category. In programming, a class definition includes both data and behavior definitions.

Client

1) A process that executes a Progress 4GL application. The client communicates with a server, and the server accesses the database on behalf of the client.

2) In client/server systems, the part of the system that sends requests to servers and processes the results of those requests. See also **Server**.

3) In distributed application environments, the application that makes remote procedure requests to a connected Progress AppServer. Also known as a client application. See also **Open Client**.

Client Application Session

The session within a 4GL client application. The attributes and methods for this session can be referenced using the session system handle within a procedure executing in the client application. This session is separate and distinct from the connected AppServer session.

Client Node

In a network, a node that uses the services of other node to run an application, such as a report writer.

Client/Server Application

A single application divided into two or more pieces for efficiency. In general, the client piece handles user-interface issues, and the server piece handles database issues. Clients and servers typically run on different computers and communicate over a network, but this is not essential. One server can support many clients. Clients are sometimes referred to as being thin or thick. Thin clients do less processing locally; the thinnest clients might be little more than terminals.

Client/Server Architecture

A system in which a server (or group of servers) access a database on behalf of one or more clients. The clients execute application programs.

Client Session

See **Client Application Session**, **Session**.

Clipboard

A system-defined (or Progress-defined) temporary storage area in memory that Progress uses to store cut and copied text for the session. The clipboard allows applications to enable users to cut, copy, and paste text between Progress widgets, and on GUI platforms between Progress widgets and external (non-Progress) applications.

Clipboard Operations

The cut, copy, and paste data transfers provided by the system clipboard to the user.

Code Block

A Progress mechanism for grouping statements together and treating them as a single unit. See also **Block**.

Code Page

A code page is a table that maps a numeric value to each character in a set of related characters. The computer uses code pages to interpret and process data. Standards organizations and operating system vendors create code pages for use, or you can create your own code pages according to your needs.

Code Section Editor

An AppBuilder editor used to edit code in the current AppBuilder procedure file.

Collation

The rules used to control how character strings in a character set compare with each other. Each character set specifies a collating sequence that defines relative values of each character for comparing, merging, and sorting character strings. In addition, storage systems can define additional collations that SQL statements specify with the COLLATE clause in column definitions, column references, and character-string references.

Collation Table

A table that maps a character set to an alternative collating sequence that changes the sort order of characters.

Column

1) A component of a record; a field.

2) A vertically aligned set of character cells in the character coordinate system.

Column Alias

An alias specified for a column. See also **Alias**.

Column Label

A label displayed above a column of data (field values). Column labels are useful for creating lists of values, like columnar reports. See also **Side Labels**.

COM

The Microsoft Component Object Model. This model defines a standard for designing and implementing objects that are accessible through properties, methods, and events. Progress relies on this standard to provide support for Automation objects and ActiveX controls.

COM-HANDLE Data Type

A component handle value. This data type supports access to properties and methods provided by COM objects, including Automation objects and ActiveX controls. See also **COM Object**, **Component Handle**.

COM Object

Encapsulated Windows application object that conforms to the Microsoft Component Object Model (COM). This object provides reusable functionality for an application. Progress applications can access COM objects through Progress support for Automation objects and ActiveX controls. See also **ActiveX Control**, **Automation Object**, **COM**.

Combo Box Widget

A field-level widget that presents a scrolling list of character strings. The list is only visible when the user chooses the button next to the combo box. When the user selects a value, the combo box closes and displays only the selected value.

Command

An instruction to the system.

Command History

A list, in the Debugger, that includes commands entered from the menu bar, button panel, and command files, as well as the command panel itself. Each entered command echoes and scrolls into the panel-scrolling area before it executes.

Command Panel

An area where you enter commands from the keyboard and record a history of all commands entered and executed during a debugging session.

Command Queue

A first-in-first-out (FIFO) input queue in which the Debugger places all commands that you enter.

Commit

Makes permanent all changes made during a transaction.

Common Data Dictionary Plus (CDD/Plus)

A single repository facility that contains definitions used by VAX information architecture products.

Comparison Expression

See **Relational Expression**.

Compilation Unit

A group of files compiled together to produce one complete program or procedure.

Compile Log

A series of compiler messages stored in an ASCII file that details the procedures compiled during a compilation session.

Compile Time

The phase of a Progress session when the compiler compiles a procedure and generates its r-code. Progress identifies any syntax errors and defines all static resources (static widgets, queries, database references, and so on) at compile time. See also **Run Time**.

Compiled Procedure

Either a session compile version of a procedure or a precompiled r-code version. See **Session Compile**.

Compiler Listing

A compiler-produced listing that consists of the following information: the name of the procedure file you are compiling, the date and time at the start of the compilation, the procedure file contents displayed with each line numbered, the block number to which each statement belongs, the complete text of all include files, and the name of any subprocedures.

Compiler Options

Output settings such as listing file options, language, and encryption key.

Component Handle

A pointer to a COM object. To allow you to reference Automation objects, ActiveX controls, and other COM objects within an application, Progress returns a unique component handle for each COM object that you instantiate. Progress provides the COM-HANDLE data type to support component handle values. See also **COM-HANDLE Data Type**.

Concatenation Operator

The plus symbol (+) used to link or join any two character strings into a single character string.

Conditional Processing

A means of processing selected code based on one or more logical expressions.

Configuration File

A data file that identifies the Progress product components that you are licensed to use.

Connected Database

A database that is accessible and can be worked on in the Data Dictionary or in an application. For a multi-user session, you must start a server database before connecting to it.

Connection

Allows communication between the Progress session and the database.

Connection Mode

One of the three possible ways a database is connected: single-user mode, multi-user client mode, or multi-user direct access mode.

Connection Overhead

The amount of time it takes an application session to connect application databases.

Connection Parameters

A subset of Progress startup parameters that are relevant during a database connect operation. Each parameter represents a value used to control how your system connects to a database.

Constant

A value that is coded in a form consistent with a supported Progress data type. A constant can be coded for these data types: CHARACTER, DATE, DECIMAL, INTEGER, or LOGICAL.

Constraint

Part of an SQL table definition that restricts the values that can be stored in a table. When you insert, delete, or update column values, the constraint checks the new values against the conditions specified by the constraint. If the value violates the constraint, it generates an error. Along with Triggers, constraints enforce referential integrity by ensuring that a value stored in the foreign key of a table must either be null or be equal to some value in the matching unique or primary key of another table.

Container SmartLink

A containing SmartObject (SmartContainer) is linked to a SmartObject that it contains. One SmartObject contains another if it creates the contained SmartObject, regardless of whether the contained SmartObject is visually parented to the container.

Container Widget

A widget that can contain other widgets. For example, a frame is a container widget.

Contents File

An ASCII file containing information used by the help engine to populate the Contents tab of the Help Topics dialog box, as well as any or all of six contents file commands that determine the behavior of the Help Topics dialog box.

Context

1) The data and functionality that is visible at any point in an application.

2) The data and functionality defined and encapsulated by a procedure or trigger block. Such data or functionality is visible only to the defining block, unless it is shared (global).

3) A concept used to communicate the fact that processing activities are defined and performed only within the session in which they are initiated.

Context Arrow

An arrow that points to the current procedure in the Debugger context.

Context ID

See **Context Number**.

Context Number

An arbitrary number assigned to an application context (certain user interface elements) by the help programmer. The help author maps context IDs to topic IDs in the [MAP] section of the help project file to provide context-sensitive help for a user interface. See also **Application Context**, **Context String**, **Context-sensitive Help**, **Help Project File**, **Topic ID**.

Context-sensitive Help

Information designed to help users understand their current status or position in the context of an application. It is a quick access method, often used to help a user interact with a current dialog box, frame, screen, or window. See also **Application Context**, **Context String**, **Context Number**. Contrast with **Reference Help**.

Context String

A unique character string that identifies each topic in the help system. The context string identifier tells the help viewer which topic to display when the user chooses a pop-up link, go-to link, or a user interface feature that has been assigned context-sensitive help. You define context strings in help topics with the pound sign (#) control code. See also **Application Context**, **Context Number**, **Control Code**, **Topic ID**.

Control

1) A visual object associated with the operating system (especially Windows) that enables input and output or decoration in an application display.

2) An operating system visualization of a 4GL widget.

3) Any object (often but not necessarily visual) in Windows. See also **ActiveX Control**, **Widget**.

Control Code

An MS-Word footnote that, when saved as an RTF file, the help compiler recognizes as having a particular purpose within the help system. For example, the help compiler includes all the keywords included in K control codes in the Index tab of the Help Topics dialog box.

Control-frame Widget

Provides the 4GL interface for a run-time instance of an ActiveX control. See also **ActiveX Control Container**.

Conversation Exchange

When a DDE statement executes or Progress posts a DDE-NOTIFY event, Progress updates the DDE attributes of the frame that owns the conversation.

Correlation Name

See **Alias**.

Corruption

Lost integrity, or the process of losing integrity. Data corruption can occur on two levels: physical and logical.

Physical corruption means the data itself has been inappropriately altered, for example, by a hardware failure causing a dropped or stuck bit that creates a value of 0 or 3 in place of a 1.

Logical corruption means different elements inappropriately get out of sync with one another, often because a multi-step operation was neither fully completed nor fully undone. No physical corruption is involved.

Cross-file Link

A jump or pop-up link that, when selected, displays a help topic that is located in a different help resource file.

Cross Product

Another term for Cartesian Product.

Crash Recovery

See **Restart Recovery**.

CRC

See **Cyclic Redundancy Check**.

Current Object

A single object (widget or SmartObject instance) that is part of an AppBuilder design window.

Current Row

The row of the active set of an open cursor. After an OPEN statement, the cursor is positioned just before the first row of the active set. The first FETCH statement advances the cursor position and makes the first row the current row. Subsequent FETCH statements make subsequent rows the current row.

Current Widget

In the AppBuilder, the widget that has input focus. The current widget is the recipient of most keyboard input.

Current Window

The window, in a procedure, to which events are posted by default and that you can reference using the CURRENT-WINDOW system handle. You can also set a new current window using the CURRENT-WINDOW attribute of a procedure's handle. See also **Active Window**, **Procedure Handle**.

Cursor

1) The current insertion point in text.

2) In SQL, a place holder. A cursor is used as a pointer used to search through a retrieval set chosen by the SELECT statement, pointing to each row in the set one at a time. When a cursor is pointing to a row, it is said to be positioned on that row.

3) The active set defined by the query in a DECLARE CURSOR statement. Host programs use cursors to retrieve multiple rows of data returned by queries.

Custom Object File

A text file (normally with a .cst extension). By editing or creating a custom object file, you can tell the AppBuilder to work with your own custom widgets and SmartObjects. You can control the appearance and behavior of the Object Palette and some aspects of the New dialog box.

Cycle

The process of creating database tables when the first table has a foreign key that references the second table, the second table has a foreign key that references the third table, and so on, and the last table has a foreign key that references the first table.

Cyclic Redundancy Check (CRC)

An algorithm used to verify the integrity of a file by comparing a stored number with the value calculated from the contents of some other related files. If the calculated value equals the stored number, then the first file and its relatives are considered consistent. In Progress, r-code files store the CRC value calculated from the schema against which they are compiled. If the schema changes, so does the CRC. Thus, the r-code files that contain the old CRC are considered inconsistent and cannot be run with the new schema.

Cycling Sequence

Begins at an initial value and increments in one direction until it passes a specified value, at which point it restarts at the initial value. See also **Terminating Sequence**.

D

Data Administration Tool

A window-based tool used to dump and load database files, implement a database strategy, and run various utilities.

Data Buffer

Progress uses two types of data buffers: the record buffer, which is a temporary storage area for a record, field, or variable; and the screen buffer, which is a display area for a field, variable, or the result of a calculation.

Data Control Language (DCL) Statements

SQL GRANT and REVOKE statements. DCL statements control access to data and the rights to issue DCL statements.

Data Definition Language (DDL) Statements

SQL CREATE, ALTER, and DROP statements used to manage tables, views, indexes, and other database objects.

Data Definitions

The characteristics of the files, fields, and indexes that comprise the schema of a Progress database. The structure of a given database. See also **Data Dictionary**.

Data Dictionary

This term has two meanings. When capitalized, it refers to the menu-driven utility program that you use to manage your schema. In lowercase, data dictionary is a synonym for schema; that is, it refers to the actual database structure definitions. In Progress, Data Dictionary refers to the Progress program, while the terms schema and data definitions are used to refer to the database structure. Data Dictionary is also another term for **System Catalog**.

Data-driven

The quality of being guided by external data, rather than having all guidance within the program itself. The behavior of an interpreter can be thought of as data-driven.

Data Integrity

The certainty of data accuracy, consistency, or validity.

Data Manipulation Language (DML) Statements

SQL SELECT, INSERT, UPDATE, and DELETE statements that access or modify data. Also another term for **System Catalog**.

Data Replication

The distribution of copies of information to one or more sites. In a single enterprise, sites spanning organizational and regional boundaries often require the timely sharing of transaction data across databases in a consistent manner. Developing and deploying a successful replication process involves careful planning and input from business experts, application developers, and administrators. Progress 4GL supports implementing trigger-based replication, and provides Database Manager support for implementing log-based site replication.

Data Source

See **ODBC Data Source**.

Data Stream

Data that flows into and out of an application. A description of the data stream fully identifies the database, the tables, and the fields involved, as well as their ordering and permissions.

Database

A collection of logically related tables that usually contain data in fields that you can read and write. Progress databases can also contain **Sequence** objects in addition to **Table** objects.

Database Administrator (DBA)

An individual who is responsible for maintaining a database management system. This person typically is authorized to grant and revoke access to the system, select universal options for users, and perform backups and installations.

Database Block

The basic unit of disk storage and I/O in a Progress database, also known as a Page.

Database Buffer

A buffer that holds a copy of a database block. The Progress database manager keeps copies of database blocks in shared memory to eliminate unnecessary disk reads and writes.

Database Event

An action performed against a database, such as finding a record or writing to a record.

Database Object

A component of the database that is defined in the database schema, such as a field, a table, an index, or a sequence.

Database Properties

The schema definitions that define the database, such as the database name and the database type.

Database Report

One of the several types of reports you can generate with the Data Dictionary, such as a detailed table report.

Database Server Machine

In a network, any node running a Progress database server or broker process.

Database Storage Representation

The format used to store a value in the database. This format is different than host language representation for some data types.

Database Trigger

A piece of code that an application associates with a component of a database and a database action (such as writing a record). When the action occurs on the component, Progress locates this piece of code and executes it on behalf of the application.

Database Type

A startup parameter or field that indicates the type of database you want to connect to in order to run applications. For example, you can connect to a Progress, ORACLE, or RMS database type.

Date Constant

A character constant that can be passed to the DATE function. The format of this character string depends on the setting of the Date Format (-d) startup parameter. There is no explicit date constant. All date values must be expressed as a character string or the appropriate integer values for month, day, and year passed to the DATE function for use wherever a date value is required.

DATE Data Type

A date and time from 1/1/32768 B.C. through 12/31/32767 A.D.

Date Field

A field defined with a DATE data type.

DATE Function

A function that accepts a character constant as input and converts that constant to the value of the DATE data type.

DBA

See **Database Administrator (DBA)**.

DBA Privileges

The level of access allowed for database administrators.

DDE

See **Dynamic Data Exchange (DDE)**.

DDE Client

Opens and manages conversations with a DDE server.

DDE Server

Responds to client requests to send or receive information and to execute server commands on the client's behalf.

Debug Listing

A specially prepared source listing that allows you to more easily set breakpoints and follow the course of procedure execution in an application. This listing also provides a point of reference for specifying arguments to Debugger commands.

Debugger Application Mode

The mode that occurs when you start an initial application using the Debugger options of the Procedure Editor and the AppBuilder. This application runs in the Debugger context, and the Debugger takes control when the application stops at an automatic breakpoint set at the first executable line.

Debugger Command File

Any ASCII file that contains only valid Debugger commands. You can execute a command file using the Debugger INPUT command. Each line of the file can contain any command or semicolon-separated list of commands that you can enter in the command panel of the Debugger window. If you include INPUT commands in the command file, they cannot input the file in which they are included (no cyclic references).

Debugger Context

Includes the data and execution status of all procedures running under Debugger control.

Debugger Stand-alone Mode

The mode that occurs when you start the Debugger from the operating system or the ADE Desktop. In stand-alone mode, the Debugger takes control immediately as your initial Progress interface and with no initial application running in the Debugger context.

Debugging Session

A Progress session in which you control or monitor a procedure with the Progress Debugger Application.

Decimal Constant

An unquoted string of characters that expresses a value of the DECIMAL data type.

DECIMAL Data Type

A decimal number of up to 50 digits, and 10 digits to the right of the decimal point.

Decimal Field

A field defined with a DECIMAL data type.

Default

The term that refers to what you get if you do not choose something else. Not everything has a default value.

Default Button

In a dialog box or frame in a graphical user interface, this is the button that Progress fires if the user presses **RETURN**. The application designates which button, if any, is the default in the frame at any given time. Progress visually highlights the default button so that the user can tell what pressing **RETURN** will do. An example is the OK button in a dialog box.

Definitions Section

The area of the AppBuilder procedure file where you write procedure descriptions, define parameters, and add include files.

Delete [an ODBC data source]

Removes information about an ODBC data source through the Delete operation of the ODBC Administrator utility. Deleting a data source does not delete the database it corresponds to, but removes information about the database's location from the \%WINDIR%\ODBC.INI file.

Delimited Identifiers

Names in SQL statements that are enclosed in double quotation marks ("). Enclosing a name in double quotation marks preserves the case of the name and allows it to include reserved words and special characters. Subsequent references to a delimited identifier must also use enclosing double quotation marks.

Demo Database

A sample Progress database that contains tables, fields, and indexes (including some data) that procedures in Progress documentation use to acquaint you with the way Progress operates.

Derealization

The actions taken by the window system to "forget" about a widget to destroy the internal data structures it uses to hold information about the widget. Progress retains its own data structures for the widget until the widget is deleted.

Derived Table

A **Virtual Table** specified as a query expression in the FROM clause of another query expression.

Deselection

1) A user action on a selected widget that tells Progress that the widget should no longer be selected.

2) The event that goes to the widget when the user does this. Progress normally responds by dehighlighting the widget.

Design Page

The page you are working in at any given time while you are building an application in the AppBuilder. The Design Page setting is relevant only in design mode. It is ignored when the SmartContainer is running. The Design Page setting is indicated in the Status Bar of the AppBuilder main window.

Design Time

For ActiveX controls, an execution state (usually available by license) that allows you to read or modify certain properties and create a unique instance of an ActiveX control. Progress allows you to access an ActiveX control at design time using the AppBuilder. See also **Run Time**.

Design Window

An AppBuilder synonym for workspace, but only for objects with a visible run-time representation. The workspace for a nonvisible object is not a design window, but rather the Section Editor combined with a tree view of editable code sections.

Destination Topic Identifier

The part of the hypertext link hidden from the user that informs the help engine which topic to display when the user activates the link. A destination topic identifier is the topic ID of a particular help topic or a KLink or ALink macro.

Detailed Table Report

A report that displays information about the structure of the selected table or of all the tables in the selected database, including information about fields and indexes defined for the table. You access this report through the Progress Data Dictionary.

Developer Events

Ten Progress-supplied event names (U1–U10) that you can apply to widgets with the APPLY statement to perform actions customized entirely within the triggers for these events. Progress does not generate these events from any other user or system actions.

Device

A combination of physical components that form a unit that performs a specific set of input/output functions (for example, a terminal or a disk drive).

Dialog Box

A container widget that either notifies the user of important information or requests more information. The main interface is disabled until the user is done working with the dialog box. A dialog box is a frame, but it has some properties of a window. Dialog boxes have modal properties.

Digit

One of the characters 0–9.

Direct Manipulation Events

Explicit actions the user performs on frame and field-level widgets with a mouse, such as selecting, highlighting, moving, and resizing.

Direct Printing

Sends application output directly to a printer. This method of printing is most frequently found on single-user systems, but can also be used in other situations.

Directory

A file system object that contains files and, possibly, other directories.

Dirty Page

A page in a database buffer that has been updated and is ready to be written to stable storage.

Display Buffer

The area of memory used by the system to assemble and store text or video images to be displayed on a terminal or monitor. Display buffer sizes range from the 2KB text buffer of a traditional timesharing terminal to the 8MB (or larger) graphics buffer on a video adapter card.

Display Format

The way data appears on screens and in printed reports. Progress automatically supplies a default display format for each data type, but you can change that default format. You can specify display formats in the Data Dictionary, or with the format phrase. See also **Format Phrase**.

Distributed Application Computing

An architectural approach to designing application software that runs across a network. Distributed application computing allows you to use application partitioning and the n-tier model. The Progress AppServer makes the distributed application computing approach available in the Progress distributed application environment.

Distributed Debugging

A debugging technique that allows you to debug several 4GL sessions associated in a distributed application environment. One Debugger session, running on one session within the distributed environment, controls the debugging process for all sessions associated with it. The procedure call stack reflects all procedures in all sessions involved, enabling the root client application to be completely aware of the AppServer sessions that it is debugging.

Distributed Multiple-database Configuration

One or more of the application databases are stored on one or more remote machines in the network, and the databases are connected to the application session using a single networking protocol.

Distributed/Simultaneous Networks Multiple-database Configuration

One or more of the application databases are stored on one or more remote machines in the network, and the databases are connected to the application session using multiple networking protocols.

DLC Directory

The directory that contains the Progress system software.

DLC Variable

The environment variable that points to the directory that contains the Progress system software.

DLL

See **Dynamic Link Library (DLL)**.

Down Frame

A frame that displays multiple records, one per line. See also **Frame**.

Down Image

The image that displays when a button is in the down or pressed state. Progress displays the down image as long as the button is pressed. Typically, the down image is similar to the up image, but with a different color or shading.

Drag-and-Drop Layout

The method by which most layout editors work. The designer selects tool icons representing the various possible user-interface components and drag-and-drops them onto a workspace, visually arranging them into their final positions and sizes.

Driver Manager

See **JDBC Driver Manager** and **OBDC Driver Manager**.

Dump File

The user-defined file in which the Data Dictionary stores table data. Dumped data files have a .d extension.

Dump Name

A unique name assigned to each database table. Progress uses the dump name if you dump the table's data.

Durability

A characteristic of transactions that requires that all changes made during a transaction be permanent after the transaction is committed.

Dynamic Link Library (DLL)

On Windows, an executable file that contains functions organized so that applications can link to them at run-time rather than at compile time. This is similar to a shared code library on other operating systems.

Dynamic Data Exchange (DDE)

A protocol Windows provides for interprocess communications. Using this protocol, two applications communicate in a client/server relationship in which the client initiates communications and the server exchanges data with, and provides services to, the client.

Dynamic SQL

A set of special SQL statements (PREPARE, DESCRIBE, EXECUTE, and EXECUTE IMMEDIATE) and data structures (SQLCA and SQLDA) that let programs accept or generate SQL statements at run-time. Such dynamically generated statements are not necessarily part of a program's source code, but can be generated at run time.

Dynamic Widget

A user interface component that the application explicitly creates (with the CREATE statement) and deletes (with the DELETE WIDGET or DELETE WIDGET-POOL statement). Dynamic widgets are not scoped to any block or tied to any data and exist only in the screen buffer. They exist until you explicitly delete them or until the session ends.

E

Edge

In a rectangle widget, the set of four lines that form the border of the rectangle. The edge is drawn in the rectangle's foreground color; the application can specify the width of the edge.

Editor Widget

A field-level widget that allows editing of large character variables or database fields. By default, editor widgets support features such as cut, copy, paste, and wordwrap.

Effective Breakpoint

In the Debugger, the closest executable line on or near the specified breakpoint line number. Specifically, if the line number specifies a nonexecutable line (for example, a comment or DEFINE statement), the effective breakpoint occurs on the next executable line after the specified line. If the line number is less than one or greater than the last executable line, the effective breakpoint occurs on the corresponding first or last executable line in the procedure.

Embedded SQL

SQL statements that are embedded within a host language program. The ESQLC precompiler translates embedded SQL statements to equivalent C-language calls to routines in the Progress SQL-92application programming interface.

Empty Database

A sample database that contains no data definitions or data but does contain the Progress metaschema, the underlying system information required to define application tables, fields, and sequences. See also **Metaschema**.

Empty Selection

1) A user action that would normally select one or more widgets but which selects no widgets because of the location of the mouse. For example, if the user clicks on an empty part of a nonselectable frame, this is an empty selection. If the user selects a box but the box does not completely include any widget, this is also an empty selection.

2) The event that is sent to the frame when the user performs an empty selection.

Encapsulation

The hiding of data and functionality within an application in such a way that access to the data and functionality outside the hidden context is only available through a standard interface. In Progress, you can encapsulate data and functionality using a persistent procedure. You define the standard interface to the persistent procedure with internal procedures of the persistent procedure. Within the Progress AppServer, the Application Server provides the encapsulated environment in which rules and data are located.

Encryption Key

The key used to decrypt the procedure file and include files during compilation. Note that if you use an encryption key, the compiler does not produce a listing (as a security measure).

Environment Variable

System variables used to tailor a user's working environment to set search paths for files and set up terminal definitions.

EPI

See **External Program Interface (EPI)**.

Error Listing

A compiler-produced listing that contains the name of the compiling directory, the name of the procedure file and any include files, the date and time of the compile, and any error messages and numbers.

Error Log File

A file that contains errors that Progress found during program execution.

Error Rate

The percentage of records that cannot be loaded during the loading of table contents (.d) files. A zero (0) error rate threshold means the load stops when Progress encounters one unloadable record. A 100 error rate threshold means that Progress does not stop for any unloadable records.

Escape Character

A character or symbol in a stream that is used to signal the system that one or more of the following characters are to be interpreted in a special way.

ESQLC

The Progress Embedded SQL-92 precompiler for C host programs and the command-line syntax to invoke it.

Event

A keyword representation of a user or system action in Progress. Events can be generated by hardware or software. If you wish to handle an event in your application, you write a procedure, usually known as a trigger, and attach it to that class of event. When an event of that type occurs, your trigger code is automatically executed. See also **Trigger**.

Event Action

Any simple user interaction, such as a single keystroke or a simple mouse interaction.

Event Function

An abstraction of one or more event actions.

Exception Handler

Host program code that tests for a variety of possible errors and specifies what action will be taken if they arise.

EXCLUSIVE Locks

Locks that SQL acquires on rows that have been modified by a transaction. EXCLUSIVE locks prevent other transactions from either reading or modifying the rows until the transaction either commits or performs a rollback.

Executable Line

Any 4GL line that contains a run-time statement (for example, FOR, DISPLAY, or CREATE). See also **Nonexecutable Lines**.

Execution Queues

Queues associated with a particular print device.

Explicit Locking

Occurs by means of language elements that just do locking, such as the EXCLUSIVE-LOCK phrase. Locks have the same effect whether they are implicitly or explicitly obtained.

Export List

A list that contains all the remote 4GL procedures associated with an Application Server that can be run by a client application. The export list is an optional feature that is created by the EXPORT( ) method. If you use this option, only those procedures that are declared within this list can be run as a result of a remote procedure request. The EXPORT( ) method is executed on the SESSION handle within an Application Server instance.

Expression

In a program, a combination of constants, variables, operators, and parentheses used to perform a desired computation. An expression can consist of anything from a single constant or variable to the most complicated arrangement of operators and functions that fit into a single program statement.

Extent

1) The number of elements in an array field or variable.

2) One of the volumes in a multi-volume database.

3) Each entry in the structure file that identifies the location of a particular volume in a multi-volume database.

External Procedure

A Progress-specific term, also called a procedure file. An external procedure is some amount of code, typically including functionally related internal procedures and functions, that is stored in a file on disk. This is the largest Progress unit of execution, consisting of a single, outer block. In addition to run-time parameter values, you can pass compile-time arguments to a noncompiled procedure file. External procedures are identified by their filenames.

External Program Interface (EPI)

Consists of 4GL statements and supporting software that enables a Progress 4GL application to exchange data and services with an external, non-Progress application. Progress provides EPIs for access to such external applications as system clipboards, Windows dynamic link libraries, ActiveX Automation objects, ActiveX controls, and any external application accessible from C using the portable Progress Host Language Call (HLC) interface, among others.

Externally Defined User-defined Function

A user-defined function whose definition resides in a procedure external to the procedure that references it. See also **User-defined Function**.

F

Field

1) The smallest unit of useful information in a database.

2) A component of a record that holds a data value of a single data type. Also called a Column.

Field Group

A widget that collects field-level widgets. All field-level widgets are children of field groups.

Field Label

A label for Progress to display on the screen or on printed reports to identify the field. If you do not supply a label, Progress uses the field name as the label.

Field-level Widget

A widget that can be part of a field group. There are two types of field-level widgets: data representation widgets and graphical widgets. Data representation widgets include browse widgets, buttons, editors, fill-in fields, radio sets, selection lists, sliders, text widgets, and toggle boxes. Graphical widgets include images and rectangles.

Field Properties

The schema definitions that define the field, such as the field name, the field type, etc.

Field Validation Criteria

A test to ensure that the user does not enter invalid data in a field.

Field Validation Message

A message the Data Dictionary displays when the user attempts to enter data in a field that does not match the field validation criteria.

File

A stream of characters or a collection of logically related records treated as a unit and managed by the operating system. A file can contain a program, data, or text.

File Filter

An expression used to limit a search for stored files to those meeting the desired criteria or naming convention. A file filter usually uses at least one wild card character and often uses a conventional prefix or extension to limit the search. For example, the file filter `b*.r` will find all SmartDataBrowser files (which, by convention, begin with b) that are precompiled (and which, by convention, have a .r extension).

File Handles

A UNIX term, roughly equivalent to the number of open files.

File Specification

A filename, directory name, or search string (representing files or directories) the compiler uses to locate one or many Progress procedure files.

Fill-in Field

A field-level widget that handles text entry and display. The fill-in field is the default widget for displaying a variable or database field. Also known as a fill in.

Filter

A string used to perform a match on other strings, such as filenames or directory names. A filter frequently uses a wild card character to match strings; for instance, `a*.w` matches all filenames that begin with the letter a and end with `.w`.

Firewall

A security mechanism that prevents unauthorized access between networks. Usually, a firewall protects a site's inner network from the Internet. A firewall is designed to examine IP addresses of packets that travel between the server and the client to control the data flow.

Fit-and-Finish

The elements that combine to create in the user a feeling that the application was or was not well-designed and carefully written.

Focus

1) The place where keyboard events go. Usually, focus is in a field-level widget. You can also say a window has focus, which means that the window contains the widget where keyboard events go, and that if the user types a menu accelerator, Progress sends the accelerator to a menu in the window (usually the window's menu bar).

2) In the Debugger, it is the procedure on the call stack that the Debugger currently references and displays in the Debugger window.

Footer

Text, such as a page number, placed on the bottom of each page of a report.

Foreign Key

1) A common field shared among tables. It is the primary key in at least one of the tables.

2) The columns in a table whose values must be either null or equal to some value in a corresponding column (called the primary key) in another table. Use the REFERENCES clause in the SQL CREATE TABLE statement to create foreign keys.

Format Phrase

Specifies one or more attributes for a widget.

Form of Use

The storage format for characters in a character set. Some character sets, such as ASCII, require one byte (octet) for each character. Others, such as Unicode, use two bytes, and are called multi-octet character sets.

Frame

A container widget composed of a display area within a window that Progress uses to display field-level widgets.

Frame Family

A related collection of parent and child frames.

Frame-level Widgets

A class of widgets that display field groups containing field-level widgets. Frame-level widgets include frames and dialog boxes.

Frame Scope

The range or extent of the frame's availability within a procedure.

FreezeFrame

A tool that runs in graphical interfaces. It captures the frames of a Progress Version 6 application screen by screen, generating Version 8 Progress 4GL code.

Free-form Query

A query written by hand, rather than by using Query Builder. Free-form queries can have any structure the author likes, or none. Query Builder generates queries with a particular structure, and you can find it difficult or impossible to create certain kinds of unusual queries with it.

Front-to-back Order

The order in which Progress overlays widgets of the same parent. Front-to-back order is also referred to as **Z-order**.

Frozen Table

A table whose field and index definitions cannot be changed. The record data is not frozen—just the data definitions. Typically, tables are frozen when application development is completed.

Fully Qualified

The state of being described uniquely. A fully qualified database field reference is one that can map to only a single, specific field in a specific database, not to a field with the same name in some other database.

Function

A prepackaged solution to a task that has a name, sometimes accepts run-time arguments, and returns a value that can be used in an expression. There are many types of functions built into the Progress language. For example, **Arithmetic** functions (such as RANDOM or EXP) perform mathematical operations on numeric values; **Character** functions (such as LENGTH and SUBSTRING) manipulate character strings or expressions; **Date** functions (such as TODAY or MONTH) provide day, month, and year information for an application. See also **Method**.

Function Keys

Special keys to which Progress assigns certain actions. In procedures, you can reassign these actions to different keys.

G

Generic Queue

A queue that holds a print request until the appropriate execution queue becomes available.

Geometry

A set of numeric values indicating an object's position and size.

Global Shared Variable

A variable that is available to all parts of a multi-part program and has the same value wherever it is used in the program.

Global Translation

A translation string associated with all instances of a string within an application.

Glue

Code that helps integrate other, more specialized code. For example, the code that copies the results of a transformation from one place to another is usually thought of as glue. Also called **supporting code**.

Go-to Link

Text or graphics that when selected, cause the help viewer to replace the currently displayed help topic with a related topic. Go-to links are also referred to as Jumps. The help viewer displays go-to links in text with a solid underline. The character viewer surrounds go-to link text with angle brackets (< >). See also **Segmented-graphics Bitmap**. Contrast with **Pop-up Link**.

Granularity

A term referring to the size of a unit of analysis, intervention, etc.

Graphical User Interface (GUI)

Features high-resolution graphics and an additional user input device called a mouse.

Graphic File

A file that contains an image that is compiled along with a help topic file into a help resource file that the help viewer displays. The Progress and Windows help viewers support various graphic file formats. Graphic files (images) and help topic files (text) comprise help source files.

Grid

A framework of criss-crossed bars used to align widgets within frames.

Gridline

The AppBuilder term for one of the imaginary lines that cover a Frame object at regular intervals, horizontally and vertically. Gridlines exist only at design time. You can set the intervals at which they occur, make them visible or invisible, and make them magnetic or not.

Grid Unit

The distance between bars in a grid expressed in character units or pixels.

Group-assign SmartLink

A SmartObject that manages a record (for example, a SmartDataViewer) is linked to a SmartObject that manages a related record or another view of the same record, to assure that all records are updated in a single transaction.

Group Properties Window

An AppBuilder window that displays a list of properties (and their current values) common to all objects of the same type as the currently selected object. If multiple objects of the same type are selected, the Group Properties Window allows you to change the values of listed properties for all selected objects simultaneously.

H

Handle

An internal identifier for a widget or procedure. See also **Procedure Handle**, **Widget Handle**.

HANDLE Data Type

A widget handle or procedure handle value. See also **Widget Handle**, **WIDGET-HANDLE Data Type**.

Handle Field

A field defined with a HANDLE data type.

Help Calling Interface

The Progress code that allows users to request and receive help information using a help keystroke, a help button, or a help menu. The calling interface invokes the help engine and directs it to display the specified help information.

Help Compiler

A Progress/Windows-supplied program that converts help source files into a finished help file using the instructions in the help project file.

Help Engine

An application (`winhlp32.exe`) designed to display help windows, populate them with help topics, and navigate help topics.

Help File

A binary file that contains the compiled help information for the help engine to read and display. Help files have a .hlp file extension.

Help Project File

An ASCII file that contains instructions (such as help macros) that tell the help compiler how to build a help resource file from particular help source files. Help project files must have a .hpj file extension.

Help Resource File

A binary file that contains the compiled help information that a help viewer reads and displays. Help resource files are often referred to as "help files" and can be identified across platforms by their .hlp file extension.

Help Source Files

Files that contain the help information for a help system. Help source files consist of help topic files and graphics files.

Help Statements

An extended subset of tokens defined by the rich text format (RTF) standard. When you author help topic files in MS-Word, and save the file as RTF, the necessary help statements are automatically generated for you.

Help String

A string that appears in the status bar of a window and describes the function of the Progress field-level widget that has input focus. You define help strings for database fields in their schema definitions. You define help strings for field-level widgets, using the HELP option in the Format phrase of certain 4GL language statements.

Help System

A multi-component system that delivers large units of help information to application users and provides tools that allow users to navigate through help information.

Help Topic

A concise, modular piece of information about a single subject. A description of a dialog box in your application, a definition of a term, a set of instructions to complete a task, or a description of a 4GL language element are examples of help topics. In the help topic file, each help topic begins with one or more control codes, and ends with a page break.

Help Topics Dialog Box

A dialog box that serves as "home base" for a help file. The Help Topics dialog box includes, by default, three tabbed pages labeled Contents, Index, and Find. The Contents tab serves as a table of contents for the help file and displays only if there is a contents file that contains at least one valid topic entry (hypertext link). The Index tab lists the keywords contained in the K footnotes of the help file. The Find tab provides a full-text search feature. Users typically access the Help Topics dialog box from the Help menu in an application window or from the Help Topics button in a help window. The Help Topics dialog box can be customized by including one or more contents file commands in the contents file.

Help Window

A window, the characteristics of which are defined in the help project file, created by the help engine to display a help topic. One or more window types can be defined in the project file. Every help project must define at least one widow type, named "main." All other window types are called secondary windows.

Help Workshop

A Microsoft Windows utility, `\Program Files\Progress\bin\hcw.exe`, for creating help files, creating and editing project and contents files, and testing and reporting on help files. Help Workshop uses information in the project (.hpj) file to compile the topic (.rtf) files and graphics files into a binary help file (.hlp). The *Help Author's Guide*, `hcw.hlp`, provides online help for Help Workshop.

Help Viewer

An application designed to display and navigate help topics. Help viewers normally display a single help topic at a time, while providing features that allow users to navigate among help topics in a help file. Progress supplies help viewers that run on UNIX character systems and the Windows Help Viewer (WinHelp) that runs in Windows.

Hidden Tables

Tables that cannot be viewed by the user.

Host Language

Any programming language in which SQL statements can be embedded for database access. The ESQL precompiler supports embedding SQL statements in C-language programs.

Host Language Representation

The format used by a host language to represented values. This format is different from the database storage representation used by Progress SQL-92 for some data types. In some cases, applications must explicitly convert between data types to insert data or manipulate data retrieved from the database.

Host Program

An application program in which SQL statements are embedded.

Host Variable

Any host language variable that is used in embedded SQL statements. Programs must declare host variables in the BEGIN-END DECLARE SECTION. Host variables can be declared as a database, host-language, or user-defined types. Depending on how they are used in an SQL statement, host variables are either input (where values stored by the program are used as an expression in an SQL statement) or output (where SQL stores values returned by queries for use by the program).

Hotspot

The origin of a hypertext link. It usually refers to a rectangular region of a segmented hypergraphics file (.shg) that contains a hypertext link. See also **Hypertext Links**, **Segmented-graphics Bitmap**, **Segmented Hypergraphics File**.

Hypertext Links

Text or a graphic that is coded to connect help information in a nonlinear fashion, causing the help engine to display a related topic when the user selects the link. Hypertext links consist of two parts: **Link Text**, which is formatted to cue the user to recognize it as such, and a **Destination Topic Identifier** that informs the help engine which topic to find when the user activates the link. See also **Go-to Link**, **Hotspot**, **Jump**, **Pop-up Link**.

I

Icon

Generally, a nontext symbol or an image used as a symbol. In Windows, it represents a window when the user has minimized it. That is, the user has directed the window system to hide the window and show the icon. The user can recover the window by double-clicking the icon.

Identifier

The name that you give any Progress 4GL object such as a table, field, variable, or internal procedure. Identifiers can be up to 32 characters long. The first character must be an uppercase or lowercase letter. The remainder of the identifier can be any combination of letters, digits, underscore, and dash characters.

Image

A field-level widget used to display an image from a bitmap file on GUI systems. The DEFINE IMAGE statement creates an image widget and optionally makes an association between the widget and an operating system image file at compile time. A button widget can also have up to three different images associated with it.

Implicit Locking

A default locking mechanism in Progress that automatically allows one user at a time to update a record. Also called **Default Locking**.

Inactive Index

An index that is defined but not active. That is, all keys are compiled and collated, but the index is not enabled for use by Progress.

Inactive Procedures

Procedures that are not executing in any context.

Include File

A separate file that contains Progress 4GL code that you can insert into a procedure at compile time by placing the file's name in braces within the procedure file (for example `{myincl.i}`). Include files typically have a .i extension in their filenames. See also **Procedure File**.

Index

1) A data structure that contains a list of fields (keys) identifying the records in a table, and the locations where the records are stored.

2) A database structure that speeds access to particular rows in a table. Indexes specify one or more columns as an index key. Queries that use the index key retrieve data faster than those that do not take advantage of an index.

Index Bracketing

Having selected one or more indexes to satisfy a query, Progress tries immediately to isolate the smallest necessary index subset so as to return as few records as possible. Careful query design can increase the opportunities for bracketing, thereby preventing Progress from scanning entire indexes and examining all records.

Index Cursors

Place holders during file access. They have size 6 by default; that is, they support up to five levels of indirect access.

Index Properties

The schema definitions that define the index, such as the index name, the fields it comprises, etc.

Indexed File Organization

A system of file organization in which files contain records and a primary key index (and optionally one or more indexes). You can retrieve records sequentially or randomly using an index.

Indicator Variable

A variable that is used to represent null values in an application program. Indicator variables must be associated with a host variable. If the value retrieved by a query is null, SQL stores a negative in the indicator variable. Programs set indicator variables to –1 to specify a null value for insertion into the database. Programs declare indicator variables as short or smallint.

Inherit

In an object-oriented environment, to automatically get properties and, perhaps, values from an ancestor object or class upon creation.

Initial Value

The value to which a field or variable is set when created.

Inner Join

A join where records from the first (left-hand) table (or join) are only returned to the results list that also return a corresponding record from the second (right-hand) table.

Input Focus

The interface element that will receive the next user event (input), provided the user does not navigate to another element. See also **Focus**.

Input Host Variable

A host variable that is used as input to an embedded SQL statement. Embedded SQL statements can refer to host variables anywhere they can refer to an expression. Host programs use input variables to provide values to insert and update data in the database, and as arguments to search conditions.

Input Parameter

In a stored procedure specification, an argument that an application must pass when it calls the stored procedure. In an SQL statement, a Parameter Marker in the statement string that acts as a placeholder for a value that will be substituted when the statement executes.

Input SQLDA

An SQLDA used by a dynamic SQL program to determine the number and data type of input parameters of an SQL statement, and to store values for those input parameters. DESCRIBE BIND VARIABLES, EXECUTE, and OPEN statements can specify an input SQLDA that SQL uses to store information about input parameters.

Insensitive Image

An image that displays when a button is insensitive.

Insensitive Widget

A widget that does not accept user input.

Insertion Point

When the user types alphanumeric data into a widget (such as an editor or fill-in field), the insertion point is the position in the widget where the next typed character will go. There is usually a visual marker on the screen to show the user where the insertion point is.

Instance

1) A run-time implementation of some object, such as a widget or persistent procedure.

2) In the Translation Manager, one or more occurrences of a string in a procedure that share common justification and maximum length characteristics. For example, each line in the String Instance Information field of the Translation Manager window represents an instance of the string displayed in the Native field. The current instance is the currently selected line in the String Instance Information field.

3) A copy of a SmartObject running in memory.

4) In object-oriented programming, a specific object allocated in memory that represents a class. A class is unique, but you can normally create an unlimited number of instances that are members of a particular class. The memory allocation for an instance is typically only enough for the instance's data; all instances of an object type share the same executable code.

Integer Constant

An unquoted string of characters that expresses a value of the INTEGER data type.

INTEGER Data Type

A positive or negative whole number, within the range of –2,147,483,648 through 2,147,483,647.

Integer Field

A field defined with an INTEGER data type.

Integrity

See **Data Integrity**.

Integrity Constraint

Another term for constraint.

Interface

In Java, a definition of a set of methods that one or more objects will implement. Interfaces declare only methods and constants, not variables. Interfaces provide multiple-inheritance capabilities.

Internal Procedure

1) A subprogram identified by the reserved word PROCEDURE. A procedure block defined inside the context of an external procedure. Internal procedures have their own local context separate from the external procedure in which they are defined. You can call an internal procedure from within the context of the external procedure. If the external procedure is active (still in scope), you can also call its internal procedures from outside the external procedure context. Internal procedures compile as part of the containing external procedure. (You cannot dynamically compile an internal procedure.)

2) Also used more generally to refer to any PROCEDURE or FUNCTION in the abstract. See also **External Procedure**.

Internationalization

The process of designing and implementing an application so that it can be modified to adapt to all cultural or linguistic contexts, a prerequisite to creating specific language editions that take regional business conventions into consideration.

Intrinsic Functions

In the Debugger, certain menu options that are not Debugger commands. Intrinsic functions are never queued. You can execute them immediately whenever the Debugger window is enabled, and whether or not an application has control.

I/O

Input and output operations for all sources and destinations.

Isolation Level

The degree by which a transaction is isolated from the database modification operations of other concurrently active transactions.

Isports Database

A sample database that is similar to the sports database, but is also designed for internationalized applications. See also **Sports Database**.

Iteration

Each completion of a block.

J

Java Snippet

See **Snippet**.

JDBC

Java Database Connectivity. A part of the Java language that allows applications to embed standard SQL statements and access any database that implements a JDBC driver.

JDBC Driver

Database-specific software that receives calls from the JDBC driver manager, translates them into a form that the database can process, and returns data to the application.

JDBC Driver Manager

A Java class that implements methods to route calls from a JDBC application to the appropriate JDBC driver for a particular JDBC URL.

Join

A type of query operation that requests information from two tables or joins at a time. The output is a results list where each row contains a record from each joined table. A join can only occur when the two queried tables or joins have a common field.

Join Condition

The criteria used to relate the fields of one table in a join with those of another.

Jump

Text or graphics that, when selected, cause the help engine to display a specified related help topic. Jumps in text appear in help windows with a solid underline. See also **Segmented Hypergraphics File**. Contrast with **Pop-up Link**.

K

Key

One or more database fields that comprise a primary, unique reference to a record in a table.

Key Function

A special meaning that Progress applies to a keystroke.

Keyword

1) A word with a predefined meaning in the Progress 4GL.

2) A word or phrase associated with one or more help topics. You can search for information in the Index tab of the Help Topics dialog box using keywords. You use the K control code to associate one or more keywords with a help topic. See also **Control Code**.

KLink

A hypertext link that locates target topics based on index keywords (keywords stored in K footnotes) rather than context strings.

L

Label

Text that appears with a field or variable when it displays in an interactive application. The label is defined in the Data Dictionary.

Language

A group of strings and data formats used in an application interface designed for use by a particular audience. Ukrainian, Urdu, and Uzbek are examples of languages. AcmeCo and AbcCo are also valid languages for an application interface.

Layout Editor

In a RAP/RAD environment, the tool that allows a user-interface designer (or programmer) to quickly arrange user-interface elements within a context. Generally, layout editors resemble graphics editors, to which they are related.

Layout Inheritance

The process of automatically passing on a master layout's characteristics to alternate layouts.

Left Outer Join

A join that returns all the records of the first (left-hand) table, including the records from the second (right-hand) table for an inner join and null field values for all records in the right-hand table that have no corresponding records in the left-hand table.

Link Text

Text or a graphic reference formatted to serve as a visual cue to the user identifying it as a hypertext link.

Literal

See **Constant**.

Local Procedure

A procedure that executes in the same session as the RUN statement that initiated it. See also **Remote Procedure**.

Local Variable

A temporary field for storing data. Data in a local variable is available only within the procedure in which it is defined.

Localization

The process of customizing an application for a specific region.

Locally Defined User-defined Function

A user-defined function whose definition resides in the procedure that references it. See also **User-defined Function**.

Lock

A restriction on data access so that updates can be performed without compromising data integrity.

Log-based Site Replication

Log-based site implementation (or site replication) is based on a monitor that watches a log of transactions and propagates these changes to other sites. Generated by the database, this log comprises a stream of database transactions. This is a very efficient way of replicating data from server to server. It also allows you to maintain a remote backup of a primary database. Progress supports site replication using after-imaging (AI) files.

Logical Constant

The keywords TRUE, FALSE, YES, or NO used to express a value of the LOGICAL data type.

LOGICAL Data Type

One of two truth values from the TRUE/FALSE or YES/NO value pairs. You can also format other corresponding value pair representations for logical data input and output.

Logical Database Name

The name used within a Progress procedure for a connected database. The logical database name is used to resolve ambiguous database references. That is, when a procedure is compiled against a database, it is the logical database name that is stored in the procedure's object code, and when a procedure executes, its database name references must match the logical name of a connected database. See also **Physical Database Name**.

Logical Expression

An expression that yields a LOGICAL (TRUE/FALSE, YES/NO) value.

Logical Field

A field defined with a LOGICAL data type.

Logical Operator

An operator such as AND, OR, or NOT used in an expression that yields a LOGICAL (TRUE/FALSE, YES/NO) value.

Logical Page

See **Page**.

Logical Three-tier Model

A computing model that supports separating the application logic from the user interface in the application implementation, and moving it to a remote server where the application data resides. A configuration based on the logical three-tier model remains a physical client/server application in that it is deployed across two machines, but the user interface, application logic, and data are logically separated from one another.

Look-and-Feel

The industry-standard term referring to the style of an application and how it is put together.

M

Macro Definition

A customized Debugger command made up of a series of commands that you can specify by entering your own command name.

Main Block Section

The area of .w file containing the enabling and blocking logic for the window or dialog box.

Mandatory Field

A field that must contain data other than the unknown value (?).

Map ID

See **Context Number**, **Topic ID**.

Master

In ADM terms, a fully-qualified SmartObject, able to have instances made from it.

MEMPTR Data Type

A pointer to a location in system memory. You can only define variables as MEMPTR and only for a Windows client. See also **RAW Data Type**.

Memptr Field

A field defined with a MEMPTR data type.

Menu

A widget that contains a list of commands or functions that users frequently use. The two kinds of menu widgets available are menu bars and pop-up menus.

Menu Accelerator Key

A key or key combination which, when pressed, invokes an associated menu item. See also **Accelerator Key**.

Menu Bar

A horizontal bar displayed at the top of a window. The menu bar contains menu labels arranged horizontally. When you select a menu label, a pull-down menu containing a vertically arranged list of items is displayed.

Menu Hierarchy

The relationships between menus and their owners and menus and their children.

Menu Item

A menu component that defines a menu option.

Menu Label

A text string in a menu bar.

Menu Owner

The widget to which a menu is attached. For menu bars, the owner is always a window. For pop-up menus, the owner is whichever widget (possibly a window) the menu pops up from.

Message

A well-defined block of data that serves as a unit of communication between two program entities. The two principal ways of passing messages are by putting them in an agreed location (mailboxing), or by sending them as a packet over the network if the sending and receiving programs are on different systems.

Message Area of Window

The lines at the bottom of a Progress window. These lines are for procedure-specific messages.

Metadata

Data that describes the objects (tables, columns, views, and indexes) that are stored in the database. Metadata is data about data. It is stored in a collection of tables called system tables. The SQL engine stores metadata in the system catalog.

Metaschema

The set of internal database tables that define the underlying structure of a Progress database. Metaschema table names begin with the underscore (_) character. Metaschema tables are hidden tables and, therefore, are not ordinarily visible in the Data Dictionary.

Method

1) A function associated with an object, such as a widget or system handle, that performs a specific action and returns a value representing the result of that action.

2) In object terminology a procedure or function, defined in a class, that implements a particular object behavior. In some object languages, methods are further divided into Public (callable from outside the object instance) and Private (callable only by another method within the same instance). See also **Function**.

Method Library

A Progress include file that contains ADM internal procedures.

Mnemonic

A single letter that you press to choose a menu option. The mnemonic for each option is underlined when the label is displayed.

Modal

A type of application that forces you to perform a specific action before you can go on to other tasks. When referring to windows and dialog boxes, the state of capturing all input. Because a modal window captures all user input, it must be dismissed before any other window can get input focus.

Modeless

A type of application that gives you control over what tasks you perform when. It provides you with a number of processing choices and then reacts depending on what event you trigger.

Mouse Pointer

See **Pointer**.

Multi-byte Character Set

Any character set that contains 256 or more characters. Multi-byte character sets contain characters that are one, two, or more bytes per character. The characters in a double-byte character set are encoded into two-byte values. Double-byte character sets represent special cases of multi-byte character sets. Progress supports multi-byte character sets. See also **Character Set**, **Unicode**.

Multi-threaded Database

A multi-user database accessed by one server per user, providing simultaneous cooperative access to all users.

Multi-volume Database

A database having multiple data segments, usually stored on separate disks. It consists of three major components: the structure file, the structure description file, and extents.

Multiple-item Transfers

A clipboard operation that involves more than one value.

Multiplexing

Reusing the same resource for different tasks. In the context of application design, multiplexing most often involves the reuse of display space or memory.

N

N-tier Model

A computing model that supports a flexible networking structure. You can physically distribute application logic and processing loads among many machines across a distributed network. Also, the n-tier model supports the logical separation of user interface, application logic, and data across three or more machines.

Navigation Palette

A group of predefined buttons and their accompanying code that the AppBuilder creates for you to accompany frames and browsers in order to navigate through database records. You can also add and update records with the Navigation Palette.

Navigation SmartLink

A SmartObject that provides an interface for navigation (for example, a SmartPanel) and is linked to a SmartObject that supplies a set of entities that can be navigated through (for example, a SmartDataObject).

Nested Blocks

Procedure blocks contained within other procedure blocks.

Nonexecutable Lines

Lines that include only comments and static data definition statements (for example, DEFINE or FORM). Static data definition statements take effect only at compile time and generate data objects for a procedure.

Nonpersistent Procedure

A procedure that creates and maintains its context only until it returns from execution. That is, the context of a nonpersistent procedure only remains in scope until the RUN statement that executes it completes.

Nonscrolling Region

A help window feature that continuously displays the header region of a help topic while the user scrolls through the topic. The help author can determine the size of this region, and turn the feature on or off. The nonscrolling region is controlled by the Keep With Next paragraph attribute in MS-Word. If a pop-up topic defines a nonscrolling region, only the nonscrolling region will appear when the topic is displayed.

Nonterminating Sequence

A sequence that begins at an initial value and increments in one direction with no limit.

Nonvolatile Events

In a debugging session, the user interface events whose triggers can always be taken control of by the Debugger, regardless of the application environment.

Normalization

The database design principle that seeks to eliminate redundant data for optimum performance and reliability.

NT Registry

Where Progress stores startup parameters and database-related information that is used for server startup.

Null Value

1) An SQL term that indicates that the value for a particular column is unknown or not available. The Progress/SQL null value is equivalent to the Progress unknown value (?).

2) The absence of a value. Host programs use indicator variables or SQL constructs to specify null values and retrieve null values from the database.

O

Object

A discrete unit of organized data and functionality that can be uniquely identified by a name or handle value (pointer). Some objects also respond to events. Objects can be categorized into classes (types) based on the common data and functionality that they possess. See also **Attribute**, **Buffer**, **COM**, **Field**, **Function**, **Method**, **Procedure**, **Property**, **Sequence**, **Table**, **Variable**, **Widget**, **Widget Pool**.

Object Oriented

One of the two principal models of computer program, the other being the functional or procedural model. In the object oriented model, an object combines state information (data) with a behavioral repertoire (procedures and functions) under a single identifier. Object programs tend by nature to be asynchronous, event-driven, and multi-pathed.

Octet

A group of eight bits. Synonymous with byte and often used in descriptions of character-set encoding format.

OCX

Another name for ActiveX Control originating with OLE. See also **ActiveX Control**, **OLE**.

OCX Property Editor

An AppBuilder window that displays a list of design-time properties (and their current values) relevant to a selected OCX. The OCX Property Editor allows you to change the values of the listed properties for the selected OCX. See also **Design Time**, **Run Time**.

ODBC Administrator

Microsoft-supplied utility to add and delete ODBC data sources and drivers. The installation procedure installs the Administrator if it is not already present on a system. To invoke it, click on the ODBC icon in the Control Panel applications of the Windows Program Manager.

ODBC Application

Any program that calls ODBC functions and uses them to issue SQL statements. Many vendors have added ODBC capabilities to their existing Windows-based tools.

ODBC Data Source

In ODBC terminology, a specific combination of a database system, the operating system it uses, and any network software required to access it. Before applications can access a database through ODBC, you use the ODBC Administrator to add a data source—register information about the database and an ODBC driver that can connect to it—for that database. More than one data source name can refer to the same database, and deleting a data source does not delete the associated database.

ODBC Driver

Software that processes ODBC function calls for a specific data source. The driver connects to the data source, translates the standard SQL statements into syntax the data source can process, and returns data to the application. Progress SQL-92 includes an ODBC driver that provides access to proprietary storage systems underlying the ODBC server.

ODBC Driver Manager

A Microsoft-supplied program that routes calls from an application to the appropriate ODBC driver for a data source.

Offset

The displacement from a starting point. The position of a field (in RMS).

Old Translation String

A translation string that has a time stamp dated earlier than the time stamp of the corresponding source string. A translation string is a string entered by a translator in a language that is different from the language of a corresponding source string.

OLE

Object Linking and Embedding, a Microsoft standard that allows different applications to interact with each other's objects. For example, MS-Word can link a work sheet from MS-Excel, so you can view it in a Word document as you change the work sheet in Excel. Word can also embed the work sheet along with controls from the Excel user interface, so you can modify the work sheet in Word as though working directly in Excel. Progress provides some features of OLE with its support for ActiveX. See also **ActiveX**, **ActiveX Control**, **Automation Object**.

One-to-Many Relationship

Occurs when one record can relate to many records in another table.

One-to-One Relationship

Occurs when one record can relate to only one instance of a record in another table.

Open Client

A Java application or ActiveX Controller programmed and configured to execute remote procedure requests on a connected Progress AppServer. Also known as an Open Client application. You can create Open Client applications using the Progress Open Client Toolkit.

Operating System

Software that controls and manages the execution of computer programs and services.

Operator

1) The symbol you use to perform numeric calculations, date calculations, character string manipulations, or data comparisons (+, /, and GT, for example).

2) A symbol or word, such as = (equals), + (added to), AND, or NOT, that is part of the 4GL syntax and modifies one or more values according to a specified rule.

Options

A single keyword and a possible accompanying value. Options modify the way Progress executes a statement.

Order Number

The default order in which Progress displays fields on a form. By default, Progress numbers fields in the order you enter them, in increments of ten. You can override the Progress display order of fields in your procedures by naming the fields in the order you want to display them.

Outer Join

See **Left Outer Join**, **Right Outer Join**.

Output Destination

A file or device to which a program sends output. Output destinations can include a terminal, a printer, a text file, or a printer queue.

Output Host Variable

A host variable that SQL uses to store results from a query.

Output Parameter

In a stored procedure specification, an argument in which the stored procedure returns a value after it executes.

Output SQLDA

An SQLDA used by a dynamic SQL program to determine the number and data type of columns in the result set returned by a query, and to retrieve values of that result set. DESCRIBE SELECT LIST and FETCH statements can specify an output SQLDA that SQL uses to store the column information and data.

Overlap Factor

Determines the redundancy built into each incremental database backup. An overlap factor of one on every backup allows for losing any one incremental backup in a backup series, as long as the immediate predecessor of that backup is not also lost, and as long as you have not lost the last in the series. An overlap factor of two allows for losing the two immediate predecessors. The default overlap factor is zero.

Overlay

A type of multiplexing in which the same area of memory is loaded with different program modules as they come into use. Designing for overlay requires that the sections to be overlaid be independent of one another, since only one can occupy the designated area of memory at a given time.

Overlay Frame

A frame that Progress displays on top of any other frames that overlaps its display area. See also **Child Frame**.

Owner

The first name in the privilege fields of the schema files and, by default, is the only user who can access the object. See **Menu Owner**.

P

Package

A group of related Java classes and interfaces, like a class library in C++. The Java development environment includes many packages of classes that procedures can import. The Java run-time system automatically imports the *java.lang* package. Stored procedures must explicitly import other classes by specifying them in the IMPORT clause of a CREATE PROCEDURE statement.

Page

A designated group of SmartObjects within a SmartContainer. By allowing you to organize sets of SmartObjects into logical pages, the AppBuilder gives you control over when these sets of objects are initialized and displayed. Page 0 is the default page, which is always displayed. All basic objects are assigned to Page 0 regardless of the Design Page setting. You can control the viewing and hiding of logical pages in a variety of ways, including a SmartFolder, a View Page radio set, or a series of Select Page buttons.

Page SmartLink

A SmartObject that provides an interface for selecting pages (for example, a SmartFolder) is linked to a SmartObject that manages the hiding and viewing of pages (for example, a SmartContainer).

PageN SmartLink

A SmartObject that manages the hiding and viewing of a numbered set of pages (for example, a SmartContainer) is linked to the SmartObjects on a specific page. Creating an object while a page is active automatically creates the PageN link.

Parameter

1) An argument to a built-in or user-defined function. It can be a constant value, field, or expression. The parameters to a function provide the actual values that the function processes.

2) A variable or constant passed between a procedure and either a called procedure, function, or method.

3) A Progress startup or database connection option. See also **Connection Parameters**, **Startup Parameters**.

Parameter File

An ASCII file that contains Progress startup parameters. Use parameter files to store the appropriate startup parameters for a particular database, group of users, or system configuration, rather than supplying startup or CONNECT options explicitly on the pro or Progress command line or in a CONNECT statement.

Parameter Marker

1) A question mark (?) in a procedure call or SQL statement string that acts as a placeholder for an input or output parameter supplied at run time when the procedure executes. The CALL statement (or corresponding ODBC or JDBC escape clause) uses parameter markers to pass parameters to stored procedures, and the SQLIStatement, SQLPStatement, and SQLCursor objects use them within procedures.

2) Question marks (?) in the statement string of a PREPARE statement. Parameter markers act as placeholders for input variables in dynamic SQL statements.

Parent

In the widget hierarchy, the widget that contains other widgets. For example, the help viewer main window is the parent of the Search dialog box.

Parent Frame

A frame that contains another frame. See also **Child Frame**, **Root Frame**.

Parent Window

A window that owns another window. See also **Child Window**, **Root Window**.

Parsing

Breaking the command string into its elements to examine and interpret.

Partitioning

See **Application Partitioning**.

Password

A string that is known only to the user. At startup, the password is used in conjunction with the user ID to ensure that only known users can start the application. All Progress passwords are case insensitive and can be up to 16 characters long.

Pathname

1) If absolute, it specifies the complete name of a directory or file by starting at the root directory or disk volume and tracing the hierarchy of the file.

2) If relative, it specifies the hierarchy of the file relative to the current working directory.

PCCmd

The command line utility that you can use to perform many of the same tasks as the ProControl graphical tool.

Persistent Procedure

A run-time instance of an external procedure that stays in memory until it is explicitly deleted. SmartObjects are implemented as external procedures.

Phrases

A collection of keywords and values that modify the way in which Progress executes a statement.

Physical Database Name

The actual name of the database on a disk. See also **Logical Database Name**.

Physical N-tier Model

A computing model that supports both logically and physically discrete segments. Each segment (user interface, application logic, and data) can be deployed on separate machines across the enterprise.

Pixel

The basic unit in a system of coordinates that Progress uses to map a precise location on a graphical display. Pixel coordinates are in integer form, one representing a horizontal location (x-axis), the other a vertical location (y-axis). See also **Character Coordinates**.

Placement By Reference

A technique for including graphic files in a help topic file, by specifying the filename and alignment of the graphic in the help topic file as opposed to embedding the graphic directly in the topic file.

Pointer

A symbol that represents movement of the mouse. By positioning the pointer and clicking buttons on the mouse, you can select data, icons, widgets, and commands to initiate and complete actions.

Pop-up Definition

A hypertext link that displays a definition when selected.

Pop-up Link

Text or graphics that when selected cause the help engine to display text or graphics in a smaller, overlapping window. The help viewer displays pop-up links in text with a dotted underline. See also **Segmented Hypergraphics File**, **Segmented-graphics Bitmap**. Contrast with **Go-to Link**, **Jump**.

Pop-up Menu

A menu that contains items arranged vertically and that is associated with a widget. The pop-up menu is context sensitive and appears only when the user performs a particular mouse or keyboard action while the widget has focus.

Portable Mouse Events

Mouse events based on a conceptual four-button mouse model, with SELECT, MENU, MOVE, and EXTEND buttons. (The other mouse model Progress supports is the physical three-button mouse, with left, middle, and right buttons.)

Precompiled Procedure

An r-code version (.r) extension of a Progress procedure generated when the source version (.p) or (.w) extension is compiled with the SAVE option. See also **R-code**.

Precompiler

A translator that translates embedded SQL statements in a host program to the equivalent C calls to functions in the Progress SQL-92 application programming interface (API).

Preprocessor

A component of the Progress compiler. Before the compiler analyzes your source code and creates r-code, the preprocessor examines your source code and performs text substitutions. The preprocessor also conditionally includes blocks of source code to compile.

Preprocessor Directive

A statement that begins with the ampersand (&) character and is meaningful only to the preprocessor.

Primary Database

To perform an auto-connect operation, Progress uses information stored in a primary database to connect to a second application database, prior to running compiled procedures that access the second database. The primary application database (which contains the auto-connect information for one or more additional databases) must already be connected before Progress can perform an auto-connect.

Primary Index

Usually the most frequently used index. Progress allows you to set one index as primary for each table and uses it by default when searching for a record.

Primary Key

1) A field whose value uniquely identifies each record in a table. Because the key value is always unique, you can use it to detect and prevent duplicate records. Primary keys should be mandatory, unique, stable, and short.

2) A subset of the fields in a table, characterized by the constraint that no two records in a table can have the same primary key value, and that no fields of the primary key can have a null value. Primary keys are specified in a CREATE TABLE statement.

Primary Login Broker

In a network, the initial database broker that handles communications for one protocol and manages shared memory for the same database that is accessed by secondary login brokers that handle communications for additional protocols.

Private Data

A character string that Progress stores with a widget. Each widget can have its own private data, which the application can set and query. The data has no meaning for Progress.

Procedure

A block of code that you can execute (call) by name using the RUN statement and to which you can pass run-time parameters.

Procedure Area

The visible part of the current edit buffer. This is where you type and edit Progress procedures. It is also called a **Text Pane**.

Procedure Block

See **Block**.

Procedure Body

In a stored procedure, the Java code between the BEGIN and END keywords of a CREATE PROCEDURE statement.

Procedure Call Stack

A last-in-first-out (LIFO) memory structure that keeps track of the execution status of each procedure as it calls a subprocedure or invokes a trigger block. The number of procedures on the call stack grows as each subprocedure calls another, and decreases as each subprocedure returns to the one that called it.

Procedure Editor

An interactive utility that provides operations for editing code. You can use it to create, write, compile, and run Progress procedure files.

Procedure File

A source file that contains the Progress 4GL code for one external procedure that can be compiled into r-code. Procedure files usually have a .p or .w extension in their filenames. See also **Include File**, **R-code File**.

Procedure Handle

An internal identifier (or pointer) for a procedure. Progress assigns each active procedure a unique procedure handle. Progress provides the WIDGET-HANDLE and HANDLE data types to support procedure handle values. The THIS-PROCEDURE system handle references the handle for the current procedure.

Procedure Objects

Reusable objects. The ADM is grounded in the idea that developers can create procedure objects that can be reused again and again. SmartObjects are procedure objects.

Procedure Oriented

One of the two principal models of computer program, the other being the object or object-oriented model. In this model, procedures are like stations on an assembly line, receiving data and transforming it before handing it off to the next station down the line. Procedural programs are able to handle both synchronous and asynchronous problems.

Procedure Result Set

In a stored procedure, a set of data rows returned to the calling application. The number and data types of columns in the procedure result set are specified in the RESULT clause of the CREATE PROCEDURE statement. The procedure can transfer data from an **SQL Result Set** to the procedure result set or it can store data generated internally. A stored procedure can have only one procedure result set.

Procedure Settings

Maintain information about AppBuilder-generated procedure files. This information includes: the type of procedure file; the directory in which the AppBuilder compiles the procedure file; the external tables (if any) required for the procedure file; the custom lists available for the procedure file; and, if the procedure file is a SmartObject, the relevant information about its method libraries, SmartLinks, and pages.

Procedure Source File

See **Procedure File**.

Procedure Specification

In a CREATE PROCEDURE statement, the clauses preceding the procedure body that specify the procedure name, any input and output parameters, any result set columns, and any Java packages to import.

Procedure Variable

A Java variable declared within the body of a stored procedure, as compared to a procedure Input Parameter or Output Parameter, which are declared outside the procedure body and are visible to the application that calls the stored procedure.

Procedures Section

The area of the AppBuilder procedure file where you code internal procedures.

Process

The environment for a single application in a system that can support several applications running simultaneously.

ProControl

The graphical tool used to manage the Progress service (ProService) and to run Progress processes like databases, DataServers, and batch applications as registered NT services.

Progress AppServer

The Progress product that allows you to build complex distributed applications with Progress 4GL code by allowing you to initiate remote procedures within a client application and execute them within an AppServer session.

Progress Editor

See **Procedure Editor**.

Progress Help Development Utilities

The cross-platform toolkit containing the necessary programs and 4GL code to develop an application help system that is shipped with Progress. Its major components include: the help viewer, the help compiler, and the Progress 4GL code that comprises the help calling interface. See also **Help Calling Interface**, **Help Compiler**, **Help Viewer**.

Progress Session

Begins when you start Progress and ends when you execute the QUIT statement.

Project File

A text file that contains instructions that tell the help compiler how to build a help file from a specified set of help source files. Help project files must have a `.hpj` file extension.

PROPATH

A Progress environment variable that contains a list of directories Progress searches to find procedures.

Properties

1) The schema definitions for a particular database object (fields, tables, indexes, etc.).

2) Settings for widgets and SmartObjects supported in the AppBuilder by property sheets.

3) 4GL mechanisms provided to access data in COM objects. See also **Property**.

Property

1) A synonym for attribute or characteristic. Properties are usually represented by some variable value. In object programming, properties are defined at the class level and set at the instance level.

2) A value associated with a COM object, such as an Automation object or ActiveX control. The value represents an aspect (for example, size or color), state (for example, visibility), or capability (for example, to allow scrolling). Properties are always referenced in the Progress 4GL using the same format: a component handle expression, followed by a colon, followed by the name of the property. See also **COM Object**.

Proportional Font

A font that has characters of different widths.

PRO*Tools

A set of utility programs related to the process of developing and running Progress applications. For example, one of the PRO*Tools allows you to edit your PROPATH. When you run PRO*Tools, the ADE displays a resizable, editable palette of icons. You then run an individual PRO*Tool by clicking on its icon.

ProService

The Progress service. It is the component that enables Progress databases, DataServers, and batch processes to run as registered NT services.

Proxy Persistent Procedure Handle

A procedure handle within a 4GL client application to a remote persistent procedure. The PROXY attribute for a persistent procedure handle is set to TRUE if the handle is a proxy persistent procedure handle.

Publish

Related to named events. To cause a Progress named event to occur.

Pull-down Menu

A vertically displayed list of related menu elements (menu items and submenus) that is displayed when the user selects a menu label in a menu bar.

Q

Qualification

A search criteria that limits the data displayed in a report or in a query from a database. A qualification typically consists of a field name, a relational operator, and a constant. In complex qualifications, logical operators are used to join several qualifications.

Query

1) To ask for information in a database.

2) The code, typically in a nonprocedural language, that describes a desired data stream.

3) A set of database records that you create with your procedures. A query can be made up of one or more tables and can consist of all the records in a particular table or just a subset of records. The browse widget is a ready-made interface to a query.

Query Expression

An important element of the SQL languages. Query expressions specify a result table derived from some combination of rows from the tables or views identified in the FROM clause of the expression. Query expressions are the basis of the SELECT, UPDATE, DELETE, and INSERT statements, and can be used in some expressions and search conditions.

Queryable Attribute

An attribute that the application can query to obtain its current value.

Quick Field Report

In the Data Dictionary, a report that displays information about the structure of fields in one or all tables in the selected database. For each field, it lists the order number, name, data type, and format. It also shows whether the field is case sensitive, mandatory, an index component, or a view component.

Quick Index Report

In the Data Dictionary, a report that displays information about the indexes defined in one or all tables in the selected database. For example, for each table it lists the name of the index, the fields that define the index, and indicates whether the fields are ascending or descending. The report also indicates whether the indexes are primary, unique, word, abbreviated, or inactive.

Quick Table Report

In the Data Dictionary, a report that displays summary information about all the tables in the selected database. For each table it lists the filename, dump name, file flags, field count, index count, and file label. It can be accessed from the Progress Data Dictionary.

Quiet Point

A period during which all database writes are suspended and queued. The quiet point allows the backup of an active, online database. The PROQUIET ENABLE command creates the quiet point, and the PROQUIET DISABLE command ends the quiet point. After the PROQUIET DISABLE command, all writes that were queued during the quiet point are accomplished.

R

R-code

The canonical run code format that Progress generates when compiling a procedure file using the COMPILE or RUN statements.

R-code File

A file containing compiled Progress r-code for a single external procedure. Progress always generates r-code filenames with the `.r` extension. See also **Procedure File**.

Race Condition

The state of indeterminacy and uncertainty that results when an outcome is different depending on which of two blocks of code executes first, and the order of execution is not stable. The indeterminate nature of a race condition creates problems that might be **extremely** hard to solve or even reproduce.

RAD

Acronym for Rapid Application Development. A general term referring to tools and methods that help reduce application development time when compared to traditional hand-coding. See also **RAP**.

Radio Button

One of the components of a radio set; it represents one of the possible values that the radio set can take on.

Radio Set

A field-level widget composed of a series of buttons that represent a single field or variable, with each button representing a different possible value. Derived from the preset tuning buttons on many automobile radios—when one button is selected, any previously selected button is deselected.

RAP

Acronym for Rapid Application Prototyping. A general term referring to tools and methods that help programmers produce prototype applications quickly for customer review and approval prior to beginning full-scale development. See also **RAD**.

RAW Data Type

A binary field or variable value in the native format of the storage location (database or system memory). Within Progress, you can only define variables or work table fields as RAW. See also **MEMPTR Data Type**.

Raw Field

A field defined with a RAW data type.

Raw Partitions

A method of performing database operations without using the UNIX file system. All database files are managed by Progress. Raw partitioning does not use the UNIX file system because all database I/O is performed directly on disk.

Read-only File

A file to which changes in its data buffer cannot be written or saved.

Realization

The actions taken by the window system to create data structures to represent a widget. Progress calls on the window system to create these structures at a particular time (for example, when the widget is about to be displayed on the screen). See also **Widget Realization**.

Realized Widget

A widget known both to Progress and to the window system. If a widget is visible on the screen, it is realized; however, a widget can be realized and not visible at the moment. A widget can also be unrealized—Progress knows about the widget, but the window system does not.

RECID Data Type

A direct pointer to a record in a Progress database and in some DataServer databases. See also **ROWID Data Type**.

Recid Field

A field defined with a RECID data type.

Record

A single occurrence of the data contained in a table; each record is treated as a single unit. Records are made up of fields. See also **Row**.

Record Buffer

1) A temporary storage area in data memory for a record, field, or variable. When you read a record from the database, Progress makes it available to your application in the record buffer. When you write a record to the database, Progress gets that record from the record buffer.

2) A named and scoped memory area used to read and write records for a specific database table. The default name of a record buffer is the name of the table, possibly qualified by the database name.

Record Scope

The lifespan of a record in a record buffer. When the scope of a record ends, Progress releases the record and writes any changes to the database.

Rectangle

Displayed in a frame or in a frame background. A rectangle is display-only and cannot receive user input. Its edge consists of a foreground color and its interior consists of a background color.

Reference Help

Provides general technical and usage information about an application. Reference information is not designed for a specific application context. The primary access for reference help is through the Help Topics dialog box. Contrast with **Context-Sensitive Help**.

Referential Constraint

Another term for constraint.

Referential Integrity

1) The condition where the value stored in a database table's foreign key must either be null or be equal to some value in another table's matching unique or primary key. SQL provides two mechanisms to enforce referential integrity: constraints specified as part of CREATE TABLE statements prevent updates that violate referential integrity, and **triggers** specified in CREATE TRIGGER statements execute a stored procedure to enforce referential integrity.

2) The ensured consistency of file relationships.

Registry

Existing on Windows NT and Windows 95 only, is a repository for information on software, such as Progress, that runs on Windows. In earlier versions of Progress, on Windows, this information resides in initialization files such as `progress.ini`. On UNIX, which has no registry, this information resides in initialization files such as the PROTERMCAP file.

Relation

Another term for table.

Relational Database

A database that allows users to organize data based on the implicit relationships of the data.

Relational Database Management System (RDBMS)

Software that organizes and provides access to a relational database. It is a program for general-purpose data storage and retrieval.

Relational Expression

A combination of constants, variables, relational operators, and parentheses used to compare values.

Relational Operator

An operator, such as = (equal to), < (less than), or > (greater than), that is used to compare two values and yields a LOGICAL (TRUE/FALSE, YES/NO) value.

Remote Debugging

A debugging technique that restricts your debugging remote procedures only; you cannot debug any procedure in the client application session. This allows you to debug 4GL code running on an Application Server independently of the client application, and is especially useful for debugging AppServers in an Open Client application.

Remote Persistent Procedure

A remote procedure that is called as a persistent procedure. See also **Remote Procedure**.

Remote Persistent Procedure Handle

A persistent procedure handle that is also a remote procedure handle. See also **Remote Procedure Handle**.

Remote Procedure

A procedure that executes in a 4GL session that is separate from the session in which the RUN statement is executed. The session in which the RUN statement is executed is the client application session. The session in which the procedure executes is the AppServer session. See also **Local Procedure**.

Remote Procedure Handle

A procedure handle created within an AppServer session as a result of a remote procedure request initiated from a client application. The REMOTE attribute for a procedure handle is set to TRUE if the handle is a remote procedure handle.

Remote Session Debugging Mode

A debugging option that is initiated by a requesting application and restricts your debugging activities to debugging only remote application procedures; you cannot debug any procedure in the requesting application session. It allows you to debug application code running on an application server independently of the calling requesting application.

Remotely Defined User-defined Function

A user-defined function whose definition resides on a remote machine in a Progress AppServer. See also **User-defined Function**.

Repertoire

The set of characters allowed in a character set.

Replacement String

The code you use to overwrite one or all instances of a search string.

Replication

See **Data Replication**.

Report

An organized display of data from a database.

Report Builder

A robust data access and reporting tool that allows you to create production-quality reports that select, analyze, and present data from your database tables in a variety of ways. Report Builder's presentation features give you the ability to control fonts and colors, and to include images in your reports. The report definitions you create are saved in a report library.

Reserved Word

A word that has a special meaning in Progress. You cannot use a reserved word as an identifier.

Resize Handles

Tiny black squares attached to the highlight lines that surround marked widgets. When you click and drag one of these handles, you resize the widget. The corner handles proportionately resize the widget. Middle handles on the right and left edges control the widget's horizontal size. Middle handles on the top and bottom edges control the widget's vertical size.

Response Time

The amount of time it takes an application workstation to respond to a user request.

Restart Recovery

The process of restoring the database to a consistent state after a system failure. Restart recovery is performed automatically by Progress whenever the database is restarted after an abnormal shutdown or system failure.

Result Set

In a Stored Procedure, either an **SQL Result Set** or a **Procedure Result Set**. Also another term for **Result Table** and **Active Set**.

Result Table

A temporary virtual table of values derived from columns and rows of one or more tables that meet conditions specified by an SQL query expression.

Results

The interactive, menu-driven tool that lets end users with little or no programming experience query, report on, and maintain information stored in database tables. As an application developer, you can also customize Results according to the needs of individual sites, and then integrate it into your delivered applications.

Right Outer Join

A join that returns all the records of the second (right-hand) table, including the records from the first (left-hand) table for an inner join and null field values for all records in the left-hand table that have no corresponding records in the right-hand table.

Roll Back

Undo all changes made during a transaction.

Roll-forward Recovery

A feature that restores a database to a known condition in the event of media loss. You must enable after-imaging to use this feature and institute a procedure for using it effectively.

Root Client Application

A client application that initiates the Progress Debugger for distributed debugging.

Root Frame

A parent frame that is owned by a window and is the top parent of a frame family. See also **Child Frame**, **Parent Frame**.

Root Window

A parent window that is parented only by the window system and is the top parent of a window family. See also **Child Window**, **Parent Window**.

Row

1) A collection of related items. Also called a **Record**.

2) A horizontally aligned set of character cells in the character coordinate system.

ROWID Data Type

A direct pointer to a record in a Progress or DataServer database. This value is more DataServer-portable than a RECID value. See also **RECID Data Type**.

Row Identifier

Another term for **Tuple** identifier.

Rowid Field

A field defined with a ROWID data type.

Run Time

1) The phase of a Progress session, after compilation, when the interpreter executes the r-code for a procedure. See also **Compile Time**.

2) For ActiveX controls, an execution state that allows you to read or modify certain properties of an ActiveX control instance after it has been created at design time. You must create an ActiveX control in the AppBuilder before you can access its properties or methods from a 4GL application at run time. See also **Design Time**.

S

Schema

A description of a database's structure—the tables it contains, the fields within the tables, views, etc. In addition to database structure, Progress database schemas contain items such as validation expressions and validation messages. Also called the **Data Definitions**.

Schema Holder

A container for information about the data definitions for one or more non-Progress databases supported by the Progress DataServer architecture.

Schema Image

The description of a non-Progress database's structure contained in a schema holder. The DataServers use the schema image to translate Progress database requests into a format that can be used to access the data in the non-Progress database.

Schema Trigger

A database trigger stored in the database. See also **Session Trigger**, **Trigger**.

Scope

The duration that a resource or object is available to an application. The scope of static objects is determined at compile time. The scope of dynamic objects is determined at run time. See also **Context**, **Frame Scope**, **Record Scope**.

Screen Buffer

A display area for a field, a variable, or the result of a calculation. When you prompt for information or display information for the user, Progress places that information in the screen buffer. The SCREEN-VALUE attribute provides access to the screen buffer for a widget.

Search Condition

The SQL syntax element that specifies a condition that is true or false about a given row or group of rows. Query expressions and UPDATE statements can specify a search condition. The search condition restricts the number of rows in the result table for the query expression or UPDATE statement. Search conditions contain one or more predicates. Search conditions follow the WHERE or HAVING keywords in SQL statements.

Search String

An instance of the code you want to find.

Secondary Window

A help viewer window in which help topics are displayed. Unlike the main help window, a secondary window cannot contain a menu bar; however, the help author can control a secondary window's buttons, style, and placement in the [WINDOWS] section of the help project file. See also **Help Window**, **Window Redirection**.

Section Editor

See **Code Section Editor**.

Segmented Hypergraphics File

A graphics file that contains one or more hotspots. You can use the Hotspot Editor (`shed.exe`) to add hotspots to a standard Windows bitmap. Segmented hypergraphics files have a .shg extension.

Segmented-graphics Bitmap

A hypergraphic bitmap that contains one or more hotspots (or live areas). You can use tools, such as the System Help Editor utility (`shed.exe`), to add hotspots to a standard Windows bitmap. Segmented-graphics bitmaps have a .shg filename extension.

Selectable Widget

A field-level widget or frame that recognizes and responds to user actions for selection. By default, a widget is not selectable; the application must set its SELECTABLE attribute to make it selectable. Selectable widgets do not respond to most mouse events in the normal way (that is, in the way the widgets respond when they are not selectable). For example, if a button becomes selectable, the user can no longer choose it with the mouse, because whenever the user clicks the SELECT mouse button on the button widget, Progress responds by selecting the mouse, rather than by choosing it.

Selected Code

Highlighted code (shown in reverse video). Select code when you want to perform an action or command on that code or if you want to use the selected code in a command.

Selected Database

The database highlighted in the Databases combo box of the Data Dictionary window.

Selected Field

The field highlighted in the Fields combo box of the Data Dictionary window.

Selected Index

The index highlighted in the Indexes combo box of the Data Dictionary window.

Selected Sequence

The sequence highlighted in the Sequences combo box of the Data Dictionary window.

Selected Table

The table highlighted in the Tables combo box of the Data Dictionary window.

Selection

1) A user action to designate a particular widget (or group of widgets). The user can select a widget only if the application has made the widget selectable. To select such a widget, the user moves the mouse pointer over the widget and clicks the SELECT mouse button. To select more than one widget, the user can use box selection. When the user selects one or more widgets, Progress normally responds by drawing a highlight box around the widgets to show that they have been selected.

2) A user-interface event, sent to the widget that the user selected.

3) In a fill-in or editor widget, the set of characters that are highlighted (usually in reverse video). See also **Box Select**, **Selectable Widget**.

Selection Criteria

Conditional expressions used to select particular records from database tables.

Selection List

A scrolling list of character strings. The strings are the possible values for an underlying field or variable. You can use a selection list to allow the user to select one or several items from a finite list of character values.

Selectivity

The fraction of a table's rows returned by a query.

SELF

In a trigger, SELF is a system handle whose value is the widget handle of the widget that received the event that initiated the trigger. See also **Trigger**.

Self-join

In SQL, joining a table with itself using the SELECT statement.

Self-service Database Connection

A type of database connection that exists when you directly connect a database client to a database without using the network.

Sensitive Widget

A widget that accepts user input.

Sequence

A database object that provides incremental values to an application. Sequences can generate sequential values within any integer range with a positive or negative value.

Sequence Increment

The value by which a sequence increments (or decrements) values from the initial value.

Sequence Initial Value

The value at which a sequence begins incrementing (or decrementing).

Sequence Lower Limit

For a sequence with a negative increment, the value at which a sequence stops incrementing. You cannot specify a lower limit that is greater than the initial value. If you accept the default, Progress uses the minimum integer value on your system as the defined lower limit.

Sequence Name

The name the help compiler uses to distinguish different browse sequences in a help resource file. See also **Browse Sequence**, **Sequence Number**.

Sequence Number

A unique alphanumeric character that the help author assigns a help topic in a browse sequence. The help compiler uses the sequence number to do an ASCII (ascending) sort of all help topics defined in a particular browse sequence. For example, when the user selects the next help topic in a browse sequence, the help viewer displays the help topic with a higher sequence number. See also **Browse Sequence**, **Sequence Name**.

Sequence Properties

The schema definitions that define the sequence, such as the sequence name as the initial value.

Sequence Report

In the Data Dictionary, a report that displays information about all sequences defined in the selected database. For each sequence, it shows the name of the sequence, its initial value, increment value, minimum or maximum value, and whether it is cycling.

Sequence Upper Limit

For a sequence with a positive increment, the value at which a sequence stops incrementing. You cannot specify an upper limit that is less than the initial value. If you accept the default, Progress uses the maximum integer value on your system as the defined upper limit.

Server

1) A process that accesses a Progress database on behalf of a client, also known as a database server.

2) A process that services requests made by other processes. There are many different kinds of servers. Some examples include database servers, directory servers, display servers, fax servers, file servers, mail servers, name servers, modem servers, print servers, terminal servers, time servers, and so on.

3) In client/server systems, the part of the system that receives requests from clients and responds with results to those requests.

Server Name

On UNIX systems running shared memory versions of Progress, the name of the Broker process on the host machine in a network. On other systems, the name of the database server on the host machine.

Server Node

A node that provides some service to other nodes, such as access to files, printers, or other devices.

Session

1) A system handle that represents a central collection point for several session-wide attributes and methods that do not belong to any single widget.

2) Sometimes used to refer to a point-to-point connection between two processes. When connected to a database, Progress clients are said to be in session with a server.

3) The duration, from startup to shutdown, of a 4GL client. When used in this context, it is also known as a client session. The duration, from startup to shutdown, of an Application Server. Sessions within different processes can only share information using the Progress AppServer. See also **AppServer Session**, **Client Application Session**.

Session Compile

The active r-code version of a procedure for the current session. The session compile version of a procedure can be either a precompiled version or a version compiled without the SAVE option.

Session Handle

A system handle that represents a central collection point for several session-wide attributes and methods that do not belong to any single widget.

Session Trigger

A trigger defined within a 4GL application (not stored in the Data Dictionary). See also **Schema Trigger**, **Trigger**.

Setable Attribute

An attribute to which the application can assign a new value. Some attributes are setable only until the associated widget gets realized, and are not setable while the widget is realized.

SHARE Locks

Locks that SQL acquires on rows that have been read by a transaction. SHARE locks allow other transactions to read the row but prevent them from modifying the row until this transaction either commits or performs a rollback.

Shared Variable

A variable used to pass information from one procedure to another.

Side Labels

Field labels to the left of the data and separated from the data by a colon and a space. See also **Column Label**.

Signature

The signature of a procedure or function consists of the return type and the parameter types. In the ADM, a coded string that uniquely identifies a certain data stream. If the prospective sender and receiver do not have matching signatures for the data stream, it cannot be transferred between them because of the corruption that would occur.

Single-item Transfer

A single write to the clipboard that immediately replaces all data in the clipboard, and a single read from the clipboard that returns all data in the clipboard to the Progress application.

Site Replication

See **Data Replication**.

Slider

A field-level widget composed of a rectangular scale that represents a range of INTEGER values. A marker within the scale indicates the current value. Only values of type INTEGER can be viewed as sliders.

SmartContainer

The ADM class that supports paging and the placement of child objects. All the Smart organizers are members of class SmartContainer, but not all SmartContainers have an organizer role.

SmartContainers

Procedure objects designed to hold instances of other SmartObjects (including possibly other SmartContainers).

SmartDataBrowser

A SmartObject that displays and allows updates to records in a browse widget. A SmartDataBrowser not only displays records, but also coordinates their display and updates with other SmartObjects (commonly SmartDataViewers and SmartDataObjects).

SmartDataObject

A nonvisual object that accesses records from a data source. This object is used as the source for the SmartDataViewer and the SmartDataBrowser.

SmartDataViewer

A SmartObject that displays database fields in a frame. SmartDataViewers not only display records, but also coordinate their display with other SmartObjects (commonly SmartDataBrowsers and SmartDataObjects). SmartDataViewers also can coordinate with SmartPanels to modify records.

SmartDialog

A SmartContainer that has a default frame viewed as a dialog box, and three default buttons: OK, Cancel, and Help. SmartDialogs are not designed to be run persistently or to be inserted in other SmartContainers.

SmartFolder

A SmartObject that provides an interface for selecting pages in a SmartContainer.

SmartFrame

A SmartContainer that has a default frame but does not create a window. A SmartFrame groups logically related widgets (or SmartObjects) onto a frame that the AppBuilder can visualize in other SmartContainers. A SmartFrame can also be used to group SmartObjects into a single object that can be used as a logical unit.

SmartLink Diagram

A Schematic representation of how SmartObjects are linked together.

SmartObject

Progress Software Corporation's term for objects written to its Application Development Model standard.

SmartObject Instance

A persistent instance of a SmartObject procedure.

SmartObject Master

A SmartObject with specific data and UI that has been saved under its own filename.

SmartPanel

A SmartObject that groups a number of functionally related buttons onto a frame. The ADM provides templates for navigating records, for updating records, and for updating records as part of a larger transaction.

SmartWindow

A SmartContainer that has its own window and default frame. A SmartWindow can be the main window of an application or a subwindow.

Snap to Grid

An AppBuilder option that allows you to move and resize widgets only from grid point to grid point.

Snippet

In a stored procedure, the sequence of Java statements between the BEGIN and END keywords in the CREATE PROCEDURE (or CREATE TRIGGER) statement. The Java statements become a method in a class the SQL engine creates and submits to the Java compiler.

Source Code

The original version of a program containing 4GL and preprocessor statements, before it is compiled. In Progress, source files usually have the .p or .w extension. See also **Procedure File**.

Source File

See **Procedure File**.

Source Language/Native Language

The language used for strings embedded in the source code of the application or procedure.

Source Procedure

See **Procedure File**.

Source String/Native String

A string obtained from a source file for translation. The **Translation Manager** displays source strings in the Native field of the Translation Manager window.

Spooled Printing

Sends application output to a temporary file and places a print request in a print queue set up on the operating system. Spooling permits the user to queue a print request and continue processing; the system executes the print request when a printer becomes available. A spooled device can help balance the demand for online printers if you are running applications on a time-shared system.

Sports Database

A sample Progress demonstration database that contains tables, fields, and indexes (including some data) that procedures in Progress documentation use to acquaint you with the way Progress operates.

SQL Diagnostics Area

A data structure that contains information about the execution status (success, error, or warning conditions) of the most recent SQL statement. The SQL-92 standard specified the diagnostics area as a standardized alternative to widely varying implementations of the SQLCA. Progress SQL-92 supports both the SQLCA and the SQL diagnostics area. The SQL GET DIAGNOSTICS statement returns information about the diagnostics area to an application, including the value of the SQLSTATE status parameter.

SQL Engine

The core component of the Progress SQL-92 environment. The SQL engine receives requests from applications, processes them, and returns results.

SQL Result Set

In a stored procedure, the set of data rows generated by an SQL statement (SELECT and, in some cases, CALL).

SQL View Report

In the Data Dictionary, a report that displays information about all SQL views defined in the selected database. For each view, it shows the name of the view, the column list, SELECT criteria, WHERE clause criteria, and the tables contained in the view. Also displayed is whether each view is updateable and if a GROUP BY phrase or a check option was used.

SQLCA

1) SQL Communications Area. A data structure that contains information about the execution status (success, error, or warning conditions) of the most recent SQL statement. The SQLCA includes an SQLCODE field. The SQLCA provides the same information as the SQL diagnostics area, but is not compliant with the SQL-92 standard. Progress SQL-92 supports both the SQLCA and the SQL diagnostics area.

2) A host structure that SQL uses to store information about the execution of an SQL statement. The SQLCA contains information about the most recently executed SQL statement.

SQLCODE

1) An integer status parameter whose value indicates the condition status returned by the most recent SQL statement. An SQLCODE value of zero means success, a positive value means warning, and a negative value means an error status. SQLCODE is superseded by SQLSTATE in the SQL-92 standard. Applications declare either SQLSTATE or SQLCODE, or both. SQL returns the status to SQLSTATE or SQLCODE after execution of each SQL statement.

2) One of the components of the SQLCA that contains a long INTEGER indicating the status of the execution of an SQL statement.

SQLDA

A host structure used in dynamic SQL programs. The host program uses the SQLDA to determine the number and data type of input and output parameters of a dynamically generated SQL statement. SQL uses the SQLDA to determine where to retrieve input parameter values or to store query results.

SQLSTATE

A five-character status parameter whose value indicates the condition status returned by the most recent SQL statement. SQLSTATE is specified by the SQL-92 standard as a replacement for the SQLCODE status parameter (which was part of SQL-89). SQLSTATE defines many more specific error conditions than SQLCODE, which allows applications to implement more portable error handling. Applications declare either SQLSTATE or SQLCODE, or both. SQL returns the status to SQLSTATE or SQLCODE after execution of each SQL statement.

Stable Storage

Storage devices that store data persistently beyond the execution time of the process that generates the data. Such devices are resilient to processor and operating system failures. Examples include magnetic or optical disks and magnetic tapes.

Stack Entry

Each procedure on the **Procedure Stack Call**.

Startup Parameters

Parameters that tailor a Progress session or database connection.

State

1) A character string passed between two or more objects to indicate that some event has occurred.

2) A general-purpose flag that you can use to give a customized meaning.

Statement

An instruction written in the Progress 4GL.

Statement String

In dynamic SQL, the SQL statement used as an argument to the EXECUTE IMMEDIATE and PREPARE statements.

Static SQL Statement

An SQL statement that is known to the application at compile time, as opposed to a dynamically generated SQL statement.

Static Widget

A widget whose entire definition is created at compile time. You can specify static widgets using DEFINE statements or the Frame phrase and VIEW–AS phrase with several different statements.

Status Area

An optional feature of a window widget that displays one line of message text at the bottom of the window. Progress controls its appearance with the STATUS–AREA attribute.

Stored Procedure

A snippet of Java source code embedded in an SQL CREATE PROCEDURE statement. The source code can use all standard Java features as well as use Progress SQL-92-supplied Java classes for processing any number of SQL statements.

Stream

1) A sequence of characters or items.

2) A logical input or output destination used within a Progress application.

String

A character expression (a constant, field name, variable name, or any combination of these that results in a character string value).

String ID

A unique integer assigned to each source string loaded into a translation database from an XREF file. The Translation Manager uses the string ID to track changes to a source string and to associate translation strings with a source string.

String Occurrence

A single occurrence of a string in the source code of a procedure.

Structure Description File

A text file you prepare that defines the structure of a multi-volume database; the PROSTRCT utility creates a (.db) structure file from the .st structure description file.

Structure File

A table of contents (does not contain data) that tells Progress and the host operating system on which file systems and/or disk devices the data files and before-image files are located. Each entry in this table of contents is called an **Extent**.

Structured Procedure

In the AppBuilder environment, a file-level aggregation of source code—an external procedure—that was created and is maintainable by AppBuilder. AppBuilder requires that such files have a specific internal structure. The name has nothing to do with the concept of structured programming.

Structured Procedure File

An AppBuilder-generated file that is organized into sections according to the AppBuilder's standard file format. You create and edit structured procedure files using the Code Section Editor. A structured procedure file has a (.p) extension, and typically has no graphical visualization when it is running.

Submenu

See **Pull-down Menu**.

SubPH

A description of the structure of a non-Progress database (such as ORACLE or RMS) within a schema-holder database.

Subprocedures

A series of small procedures that run from one main procedure. A subprocedure can be either an internal or external procedure.

Subprogram Module

A well-bounded section of code, often written as a Procedure or Function, that addresses some part of a program's overall task.

Subscribe

To tell Progress to inform a particular procedure when a particular named event occurs. (Related to named events.)

Substitution Name

In the statement string of a PREPARE statement, a name preceded by a colon (:). Substitution names do not refer to host-language variable names, but act only as placeholders for input variables in dynamic SQL statements.

Subtype

A character-string attribute of fill-in widgets and menu items that indicates for each particular widget which category of fill-in or menu item it is. For example, for menu items, the subtypes are NORMAL, SKIP, READ-ONLY, and RULE.

Super Procedure

A procedure file that the programmer associates with another procedure file (the local procedure). When the code invokes an internal procedure or a user-defined function, Progress might search for it in the super procedures, if any, of the local procedure. (Related to procedure overloading.)

Super Procedure Stack

1) A last in first out (LIFO) list of the super procedures associated with a particular local procedure or with the current Progress session.

2) Super version (of an internal procedure or a user-defined function). An internal procedure that resides in a super procedure and that has the same name and signature as an internal procedure that resides in the local procedure. Or, a user-defined function that resides in a super procedure and that has the same name and signature as a user-defined function that resides in the local procedure.

Supporting Code

Code that integrates other code. Supporting code is more often referred to as “glue.”

Syntax

The term that refers to the grammatical structure of an expression as distinct from its meaning (semantics).

Synchronous Checkpoint

A checkpointing technique that requires locking of the entire database for the duration of the checkpoint operation, which can take several minutes. The Progress database manager uses asynchronous checkpoints. See also **Asynchronous Checkpoint**.

Synchronous Replication

In a synchronous replication model, all replication of data occurs within the scope of the original transaction. In other words, replication occurs transaction by transaction. Typically, this model is implemented using a two-phase commit protocol. Two-phase commit ensures that distributed transactions occur consistently across databases.

System Catalog

Tables created by the SQL engine that store information about tables, columns, and indexes that make up the database.

System Handle

A pseudo-widget provides a range of system capabilities sometimes associated with a visual widget. System handles that reference instances of visual widgets adopt all the attributes, methods, and events of the widget. For example, the FOCUS handle references the widget that has the current input focus. Pseudo-widget system handles provide monitoring and control capabilities (for example, CLIPBOARD or SESSION). System handles for pseudo-widgets have their own attributes and methods, but do not respond to Progress events.

System Tables

Another term for **System Catalog**.

T

Table

1) A database object that is a collection of logically related data organized into records (or rows) and fields (or columns).

2) The representation of data in a relational database as a collection of columns and rows. Also called **Relation**.

Table Label

A user-defined logical name for a table. Analogous to field label. Progress uses the table label when generating messages about the table.

TableIO SmartLink

A SmartObject that provides an interface for record modifications (for example, a SmartPanel) is linked to a SmartObject that can perform the modification (for example, a SmartDataViewer or SmartDataBrowser).

Table Number

A number that Progress assigns to each table internally.

Table Properties

The schema definitions that define the table, such as the table name, the table type, etc.

Table Relations Report

In the Data Dictionary, a report that lists the relations for one or all tables in the selected database.

Table Validation Criteria

An expression that must be true to allow record deletions in the selected table.

Table Validation Message

A user-defined message Progress displays when the user attempts to delete a record from the table without meeting the table validation criteria.

Target Database

1) The multi-volume structure file used when converting a void database.

2) The Progress or non-Progress database or files that you want to access data from when you run a procedure.

Target Language

The current translation language. The target language must be defined in the current translation database.

Template

A pattern for constructing something. In programming terms, a template is a block of incomplete code. It can be literally incomplete or be complete but need revision for current purposes. In the AppBuilder, a template is a procedure file that the AppBuilder reads and copies into a buffer as an untitled procedure.

Temporary Tables

Database tables defined within a procedure for Progress to store in memory. Note that while database tables are permanent and can be accessed by multiple users, temporary tables last only for the duration of the procedure that defines them.

Temp-table

The 4GL term for a temporary table, often created at the application level to subset the main database and so reduce the processing load of traversing irrelevant records.

Terminating Sequence

A sequence that begins at an initial value and increments in one direction until it reaches the designated limit. See also **Nonterminating Sequence**.

Text Widget

A field-level widget that displays any kind of data. It is read only and does not receive normal user events. Text widgets cannot have input focus.

THIS-PROCEDURE

The 4GL handle that identifies the currently executing procedure. In AppBuilder, THIS-PROCEDURE typically refers to the main organizer object being worked on. For example, when looking at a Window's SmartLinks, THIS-PROCEDURE always refers to the SmartWindow itself, not to any of the objects contained in it.

Tid

Another term for Tuple Identifier.

Tier

The physical and/or logical location of code on your distributed application network. Once the user interface, the application logic, and the data have been partitioned, you can deploy them on your network to suit your distributed application-specific needs. See also **Logical Three-tier Model**, **Physical N-tier Model**, **Two-tier Model**.

Time Stamp

An internal Progress file marker that indicates the date and time of the most recent modifications to a database definition.

Toggle Box

A field-level widget composed of a small box that represents a logical value. Only values of type LOGICAL can be viewed as toggle boxes. The presence (TRUE) or absence (FALSE) of filling in the toggle box indicates the current value. A toggle box can be a field-level widget or a special type of menu item.

Top-down Programming

Traditional programming where you define larger structures first and then define smaller and smaller components.

Topic ID

A unique character string that identifies each topic in the help system. The topic ID identifier tells the help viewer which topic to display when the user chooses a pop-up link, jump, or a user-interface feature that has been assigned context-sensitive help. You define topic IDs in help topics with the pound sign (#) control code. See also **Application Context**, **Context Number**, **Control Code**.

Topic Title

A word or phrase that identifies a help topic in four help engine functions: searching the list of keywords on the Index tab of the Help Topics dialog box, conducting a full-text search using the Find tab of the Help Topics dialog box, defining a bookmark using the Bookmark menu, and backtracking through previously viewed topics using the History dialog box. You define topic titles with the dollar sign ($) control code. See also **Control Code**.

Transaction

A group of logically related operations that must be completed atomically, without other activities intervening. Also called **Physical Transaction** and **Common Unit**. If any part of the transaction cannot be completed, all completed parts must be undone before execution can be allowed to proceed. A transaction left partly completed causes logical corruption that might be very hard to resolve without extensive and costly analysis. Once implementations add the ability to change data in the proprietary storage system, they must also implement transaction management to protect against data corruption.

Transaction Isolation Level

Another term for isolation level.

Translation Database

A Progress database that contains specific schema, known as XL tables, that the Translation Manager uses to store string translations by language.

Translation Manager

Allows you to select words and phrases from source procedure files for translation into other languages, and assemble them into files called translation kits. After the selected words and phrases are translated, you use the Translation Manager to review the translations and integrate them into your applications. The Translation Manager also provides statistical information about translation projects.

Trigger

1) The general 4GL term for an event-handler procedure.

2) A piece of code that Progress executes on behalf of an application when a certain database action occurs or a certain user-interface event occurs. There are two types of triggers: database triggers and user-interface triggers. See also **Database Trigger**, **User Interface Trigger**.

3) A special type of stored procedure that helps ensure referential integrity for a database. Like stored procedures, triggers also contain Java source code (embedded in a CREATE TRIGGER statement) and use Progress SQL-92 Java classes. However, triggers are automatically invoked (fired) by certain SQL operations (an insert, update, or delete operation) on the trigger's target table.

Trigger Action Time

The BEFORE or AFTER keywords in a CREATE TRIGGER statement. The trigger action time specifies whether the actions implemented by the trigger execute before or after the triggering INSERT, UPDATE, or DELETE statement.

Trigger Event

The statement that causes a trigger to execute. Trigger events can be SQL INSERT, UPDATE, or DELETE statements that affect the table for which a trigger is defined.

Trigger Procedure

A Progress procedure file (.p) beginning with a trigger header statement that indicates the trigger type. For example, TRIGGER PROCEDURE FOR FIND OF table.

Trigger Report

A report that lists all tables and fields in the report that have triggers defined for them. It also lists the type of event that causes the trigger to execute, indicates whether the Dictionary checks the cyclical redundancy check (CRC) code for the trigger procedure, and shows the name of the procedure. Markers indicate whether the trigger is overrideable, has a mismatched CRC code, or has no r-code available.

Triggered Action

The Java code within the BEGIN END clause of a CREATE TRIGGER statement. The code implements actions to be completed when a triggering statement specifies the target table.

Trigger-based Replication

Implemented using event triggers stored in the database. When an event to be replicated occurs (that is, a record is created, modified, or deleted) the database uses the event to record the change in a replication change log. Using the change log, you write procedures to replicate data to other sites. Progress 4GL provides full support for trigger-based replication.

TTY Dialog

A dialog box created in the AppBuilder specifically for Character Mode.

TTY Window

A window created in the AppBuilder specifically for Character Mode.

Tuple Identifier

A unique identifier for a tuple (row) in a table. The SQL scalar function ROWID and related functions return tuple identifiers to applications.

Two-phase Commit Protocol

A feature Progress uses to ensure that a record update executes across all databases. In the two-phase commit protocol, all the databases affected by the transaction are first polled to see if they are reachable. If they are, the update is committed to each database. If they are not, then the record update is cancelled, and all databases are restored to their prior state. If a database connection is lost during the commit phase you can resolve the inconsistency. Progress uses a two-phase commit protocol to minimize the effects of incomplete transactions.

Two-tier Model

The traditional client/server computing model that has two tiers that are logically and physically separated: the client and the server. In a two-tiered system, the user interface and the application logic are tightly integrated and located on a client machine. The data resides on a separate server machine.

U

Undo

A feature in certain Progress tools and 4GL blocks that rolls back a transaction.

Unicode

A collection of characters and symbols taken from all of the spoken and written languages used in business markets all over the world. Progress support of the Unicode Standard simplifies multi-national deployment, provides multi-national information access, and supports multi-lingual applications. See also **Character Set**, **Multi-byte Character Set**.

Unique Index

An indexed field where every index key must be different. For example, the Social Security number could be a unique index.

Unique Key

The columns in a table whose values must be unique. Use the UNIQUE clause of the SQL CREATE TABLE statement to create unique keys. Unique keys are also called candidate keys.

Unknown Value

A special Progress data value, represented by a question mark (?), that indicates that the field or variable data is unknown, uninitialized, or unavailable.

Unnamed Frames

A frame that Progress allocates by default, but that you do not explicitly name.

Unrealized Widget

A widget that the window system does not know about, and has no data structures to represent. An unrealized widget is known to Progress but not to the window system.

Unsubscribe

To cancel a subscription to one or more named events. (Related to named events.)

Up Image

The primary image for a button. It is displayed when the button is in the up or unpressed state.

URL

In general, a Universal Resource Locator used to specify protocols and locations of items on the Internet. In JDBC, a database connection string in the form jdbc:*subprotocol:subname*.

Usability

The characteristic of being easy and pleasant to learn and use. Software with high usability is easy to learn for anyone able to perform the underlying task, easy to use for both novice and expert, enhances productivity for both novices and experts, and has a positive effect on every user's sense of competence.

User-defined Function

A Progress 4GL function that you define using the FUNCTION statement and that you reference within a 4GL statement.

User ID

User identification. In Progress, a user ID is a 32-character string. The name can consist of alphabetic characters, digits, and the special characters #,$,%,&, -, and _. User IDs are not case sensitive.

User Interface

The controls, methods, and processes made available to the user of (in this context) software for the purpose of doing work. User interfaces range from a blinking command-line cursor and dozens or hundreds of cryptic commands, to sophisticated simulations that are intuitively easy to use for anyone with the appropriate real-world experience.

User Interface Event

A signal or message sent to a widget in response to a user or program action, such as pressing a key or manipulating a mouse. An event often is associated with a default Progress action, such as giving the widget focus. A procedure can define its own actions to associate with an event using a trigger. See also **User Interface Trigger**.

User Interface Trigger

A piece of code that an application associates with one or more event-widget pairs. When the widget receives the event, Progress locates this piece of code and executes it on behalf of the application.

User Report

In the Data Dictionary, a report that displays the list of users who have permission to use the selected database. This information is stored in the _User file and is used to control security on the database.

Utility

A program designed to help perform a general or routine task, usually of an administrative nature.

V

Validation Criteria

A test to ensure that the user does not enter invalid data in a field or that when the test is true a user can delete a table.

Validation Expression

A test to make sure that the user does not enter invalid data in a field.

Validation Message

A string that displays when data entry produces a FALSE value for the validation expression.

Variable

A temporary field for storing data in memory.

Version File

A file displayed when Progress starts that shows the version of Progress you are running.

View

1) A subset of rows and columns in one or more tables or two or more joined tables. Represented internally by a stored command, not stored data, a view can display data from one or more tables or from other views. The following SQL statements define and delete views: CREATE VIEW and DROP VIEW.

2) A virtual table that recreates the result table specified by a SELECT statement. No data is stored in a view, but other queries can refer to it as if it were a table containing data corresponding to the result table it specifies.

VIEW-AS

A reserved term in the 4GL. You use it to indicate to the compiler the type of widget you wish to have generated to display some data.

View Properties

The schema definitions that comprise the view, such as the view name and table field names.

Virtual Machine

The Java specification for a hardware-independent and portable language environment. Java language compilers generate code that can execute on a virtual machine. Implementations of the Java virtual machine for specific hardware and software platforms allow the same compiled code to execute without modification.

Virtual System Tables

Table schemas that the Progress database manager populates with database information during run time. This information is the same data that PROMON collects and displays. The virtual system table format provides 4GL application access to the database information.

Virtual Table

A table of values that is not physically stored in a database, but instead derived from columns and rows of other tables. SQL generates virtual tables in its processing of query expressions: the FROM, WHERE, GROUP BY, and HAVING clauses each generate a virtual table based on their input.

Visual Translator

A tool used by those who perform the actual translation of words and phrases into other languages. The Visual Translator acts on a translation kit that has been generated by the Translation Manager, and allows the person performing the translation to work either from a list or in the visual context of an application's windows.

Void Multi-volume Database

The resulting (.db) file when you use the prostrct create command on a structure description file.

W

WAL

See **Write-ahead Logging (WAL)**.

Widget

An object that provides visual and interactive capabilities for a Progress application. For example, a widget might be an image, a menu, a frame, a window, or a fill-in field. Widgets are also known as User Interface Components (UICs). A widget is a 4GL-aware control that shares common capabilities with other controls of the same type defined in the 4GL.

There are several widget types that form a hierarchy in the 4GL, starting with windows. Windows can own other windows. Windows also contain frames and dialog boxes. Frames and dialog boxes contain field groups, and field groups contain field-level widgets or other frames. Widgets have attributes and methods and respond to events. Widgets serve as receptors for user input and also display application output.

Widget Attributes

Characteristics of the widget, such as its type, screen location, size, color, font, and relationship to other widgets.

Widget Browser

The AppBuilder browser used to display a list of all the widgets in an AppBuilder session. The widget browser displays the name, type, label/title, and parent for each widget. With the widget browser you can sort the widgets list and access the AppBuilder editors used to modify widget triggers and attributes. You access the widget browser by selecting its button from the button panel of the AppBuilder main window.

Widget Handle

A pointer to a widget. To allow you to reference widgets within an application, Progress assigns each widget a unique widget handle. Progress provides the WIDGET-HANDLE data type to support widget and procedure handle values.

WIDGET-HANDLE Data Type

A widget handle or procedure handle value. This data type supports access to attributes and methods provided by widgets and procedures. See also **HANDLE Data Type**, **Widget Handle**.

Widget Pool

A group of dynamic widgets that are scoped together and can be deleted as a group. Every dynamic widget you create is assigned to a widget pool.

Widget Realization

The process of fully creating a widget at run time. When you create or define a widget, Progress creates an internal data structure associated with that widget. Before the widget can be displayed on the screen, the windowing system must also create a data structure for the widget. When this second data structure exists, the widget is realized. See also **Realization**.

Wild Card Character

Used in LIKE clauses to match character strings. The percent sign character (%) matches zero or more characters. The underscore character (_) matches any single character.

Window

A container widget that can contain frames or other windows, and is movable and resizable. A window is always in one of three states: minimized, maximized, or restored. Each time you start a Progress session, it automatically creates a default window. In character mode, the default window is the only window you use and is equivalent to your screen. In graphical interfaces, you start with the default window but you can create other windows dynamically within your application.

Window Family

A related collection of parent and child windows.

Window Procedure File (.w File)

A special class of a Progress procedure file generated when you create and save a Progress window or dialog box with the AppBuilder. Window procedure files must have a (.w) extension in their filenames.

Window Redirection

The technique used for displaying a help topic in another help viewer window. See also **Secondary Window**.

Window System

Software responsible for allocating rectangular areas of a display to window applications. It also manages the location and size of each window, the window border, the mouse and cursor interface, the operations for moving and resizing windows, and clipping information for each window.

Word Index

Contains all the words from a character field or array of character fields. Searching with this index makes it very easy to locate records that contain specific words.

Work Tables

Temporary tables stored in memory for use within a procedure or several procedures (also known as work files for backward compatibility). You define work tables using the DEFINE WORK-TABLE statement. See also **Temporary Tables**.

Working Database

In most Progress tools, the connected database that is currently in use and whose database objects, for example, tables, fields, indexes, you can access and modify in the Data Dictionary.

Working Directory

The directory you are in when you issue a command.

Workspace

The place where you design and assemble the elements of your application. In AppBuilder, a workspace can involve visual layout, in which case the workspace itself—character window or GUI dialog box, for example—becomes part of the result.

Write-ahead Logging (WAL)

A log protocol in which the log records that describe database changes are always written to stable storage before the changed database blocks. Database changes can be buffered in volatile storage indefinitely, because if they are lost, the changes can be repeated by reading the information in the log and redoing each operation. The Progress database manager uses write-ahead logging. In order to limit recovery time and also to be able to reuse space in the before-image log, it periodically writes all memory-resident changes to stable storage. This operation is called a checkpoint. See also **Before-image Log**, **Checkpoint**.

X

XFTR (Extended Feature)

A custom code block that you write and place in a procedure file. At critical moments in the processing of the procedure file, the AppBuilder passes control to the XFTR, which can then run its own 4GL procedures. These procedures can interact with the AppBuilder, calling the AppBuilder Application Program Interface (API). Using XFTRs, you can perform a variety of customized processing, such as creating and deleting objects, modifying code sections, and creating custom preprocessor variables. Wizards and Cue Cards are examples of customized processing implemented with XFTRs.

Xref Listing

A compiler-produced listing that contains cross-reference information between procedure files and database objects.

Z

Z-order

See **Front-to-back Order**.