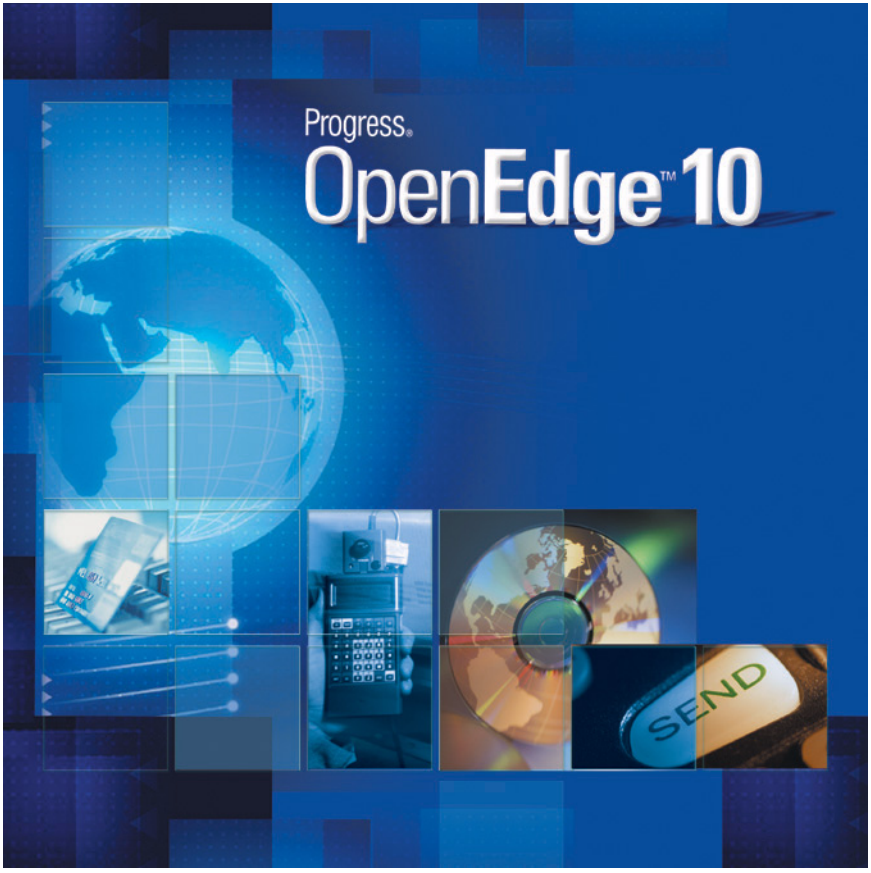

OpenEdge™ Getting Started: New and Revised Features

August 2004

Product Code: 4645
Item Number: 101202; R10.0B

Contents

Tables

Preface

This Preface contains the following sections:

Purpose

The *OpenEdge Getting Started: New and Revised Features* guide briefly introduces new features and changes to existing features from Progress® OpenEdge™ Release 10.0A to Progress OpenEdge Release 10.0B. It guides you to where you can find more detailed information about these changes in the OpenEdge Release 10.0B documentation set. The Release 10.0B documentation set references include:

- **Product manuals or sections of manuals** — identified by the **Manual** label in tables throughout this book.
- **HTM-based online help** — identified by the **HTML online help** label in tables throughout this book.
- **Web papers** — identified by the **Web paper** label in tables throughout this book.

  You can access the OpenEdge Web paper from the following Web site: `http://www.progress.com/products/documentation`. If you are viewing the pdf version of this document, click the Web paper title to go directly to the web site.
- **Release Notes** — provided in two formats: the hard-copy version included in the Release 10.0B product media kit and the on-line version available after the product is installed.

Audience

This guide is primarily intended for OpenEdge application developers and system administrators who are upgrading their license to Release 10.0B from Release 10.0A. It is also a good information source for other existing Release 10.0A users who want to know more about the new and changed features available in the 10.0B release.

Organization

Chapter 1, "What's New"

This chapter briefly introduces new and enhanced features from Release 10.0A to Release 10.0B and references places within the documentation set where more detailed feature information can be found, and a Release 10.0B Documentation map by technology.

Chapter 2, "Feature Comparisons"

Describes how changed features in OpenEdge Release 10.0B might affect application code written in OpenEdge Release 10.0A.

Typographical conventions

This manual uses the following typographical conventions:

Convention	Description
Bold	Bold typeface indicates commands or characters the user types, or the names of user interface elements.
Italic	Italic typeface indicates the title of a document, provides emphasis, or signifies new terms.
SMALL, BOLD CAPITAL LETTERS	Small, bold capital letters indicate OpenEdge™ key functions and generic keyboard keys; for example, **GET** and **CTRL**.
KEY1-KEY2	A hyphen between key names indicates a *simultaneous* key sequence: you press and hold down the first key while pressing the second key. For example, **CTRL-X**.
KEY1 KEY2	A space between key names indicates a *sequential* key sequence: you press and release the first key, then press another key. For example, **ESCAPE H**.
Syntax:	
`Fixed width`	A fixed-width font is used in syntax statements, code examples, and for system output and filenames.
`Fixed-width italics`	Fixed-width italics indicate variables in syntax statements.

Convention	Description
Fixed-width bold	Fixed-width bold indicates variables with special emphasis.
UPPERCASE fixed width	Uppercase words are Progress® 4GL language keywords. Although these always are shown in uppercase, you can type them in either uppercase or lowercase in a procedure.
	This icon (three arrows) introduces a multi-step procedure.
	This icon (one arrow) introduces a single-step procedure.
Period (.) or colon (:)	All statements except DO, FOR, FUNCTION, PROCEDURE, and REPEAT end with a period. DO, FOR, FUNCTION, PROCEDURE, and REPEAT statements can end with either a period or a colon.
[ ]	Large brackets indicate the items within them are optional.
[ ]	Small brackets are part of the Progress 4GL language.
{ }	Large braces indicate the items within them are required. They are used to simplify complex syntax diagrams.
{ }	Small braces are part of the Progress 4GL language. For example, a called external procedure must use braces when referencing arguments passed by a calling procedure.
|	A vertical bar indicates a choice.
...	Ellipses indicate repetition: you can choose one or more of the preceding items.

Examples of syntax descriptions

In this example, ACCUM is a keyword, and *aggregate* and *expression* are variables:

Syntax

```
ACCUM aggregate expression
```

FOR is one of the statements that can end with either a period or a colon, as in this example:

```
FOR EACH Customer:
  DISPLAY Name.
END.
```

In this example, STREAM *stream*, UNLESS-HIDDEN, and NO-ERROR are optional:

Syntax

```
DISPLAY [ STREAM stream ] [ UNLESS-HIDDEN ] [ NO-ERROR ]
```

In this example, the outer (small) brackets are part of the language, and the inner (large) brackets denote an optional item:

Syntax

```
INITIAL [ constant [ , constant ] ]
```

A called external procedure must use braces when referencing compile-time arguments passed by a calling procedure, as shown in this example:

Syntax

```
{ &argument-name }
```

In this example, EACH, FIRST, and LAST are optional, but you can choose only one of them:

Syntax

```
PRESELECT [ EACH | FIRST | LAST ] record-phrase
```

In this example, you must include two expressions, and optionally you can include more. Multiple expressions are separated by commas:

Syntax

```
MAXIMUM ( expression , expression [ , expression ] ... )
```

In this example, you must specify MESSAGE and at least one *expression* or SKIP [(*n*)], and any number of additional *expression* or SKIP [(*n*)] is allowed:

Syntax

```
MESSAGE { expression | SKIP [ ( n ) ] } ...
```

In this example, you must specify { *include-file*, then optionally any number of *argument* or *&argument-name = "argument-value"*, and then terminate with }:

Syntax

```
{ include-file
    [ argument | &argument-name = "argument-value" ] ... }
```

Long syntax descriptions split across lines

Some syntax descriptions are too long to fit on one line. When syntax descriptions are split across multiple lines, groups of optional and groups of required items are kept together in the required order.

In this example, WITH is followed by six optional items:

Syntax

```
WITH [ ACCUM max-length ] [ expression DOWN ]
  [ CENTERED ] [ n COLUMNS ] [ SIDE-LABELS ]
  [ STREAM-IO ]
```

Complex syntax descriptions with both required and optional elements

Some syntax descriptions are too complex to distinguish required and optional elements by bracketing only the optional elements. For such syntax, the descriptions include both braces (for required elements) and brackets (for optional elements).

In this example, ASSIGN requires either one or more *field* entries or one *record*. Options available with *field* or *record* are grouped with braces and brackets:

Syntax

```
ASSIGN   { [ FRAME frame ] { field [ = expression ] }
           [ WHEN expression ] } ...
       | { record [ EXCEPT field ... ] }
```

Examples of syntax diagrams (SQL)

In this example, GRANT, RESOURCE, DBA, and TO are keywords. You must specify RESOURCE, DBA, or both, and at least one *user_name*. Optionally you can specify additional *user_name* items; each subsequent *user_name* must be preceded by a comma:

Syntax

```
GRANT { RESOURCE, DBA } TO user_name [, user_name ] ... ;
```

This excerpt from an ODBC application invokes a stored procedure using the ODBC syntax { call *procedure_name (param)* }, where braces and parentheses are part of the language:

Syntax

```
proc1( param, "{ call proc2 (param) }", param);
```

In this example, you must specify a *table_name*, *view_name*, or *synonym*, but you can choose only one. In all SQL syntax, if you specify the optional *owner_name* qualifier, there must not be a space between the period separator and *table_name*, *view_name*, or *synonym*:

Syntax

```
CREATE [ PUBLIC ] SYNONYM synonym
  FOR [ owner_name.]{table_name |view_name |synonym } ;
```

In this example, you must specify *table_name* or *view_name*:

Syntax

```
DELETE FROM [ owner_name.]{table_name |view_name }
  [ WHERE search_condition ] ;
```

In this example, you must include one expression *(expr)* or column position *(posn)*, and optionally you can specify the sort order as ascending (ASC) or descending (DESC). You can specify additional expressions or column positions for sorting within a sorted result set. The SQL engine orders the rows on the basis of the first *expr* or *posn*. If the values are the same, the second *expr* or *posn* is used in the ordering:

Syntax

```
ORDER BY { expr | posn } [ ASC | DESC ]
  [ , [ { expr | posn } [ ASC | DESC ] ] ... ]
```

Long syntax descriptions split across lines

Some syntax descriptions are too long to fit on one line. When syntax descriptions are split across multiple lines, groups of optional and groups of required items are kept together in the required order.

In this example, `CREATE VIEW` is followed by several optional items:

Syntax

```
CREATE VIEW [ owner_name.]view_name
  [ ( column_name [, column_name ] ... ) ]
  AS [ ( ] query_expression [ ) ] [ WITH CHECK OPTION ] ;
```

1

What's New

This chapter briefly introduces new and enhanced features from Release 10.0A to Release 10.0B as described in the following sections:

Changes in OpenEdge Release 10.0B

This section briefly identifies new and enhanced features in Progress® OpenEdge™ Release 10.0B. For more detailed information about a feature, see the appropriate OpenEdge documentation referenced.

Administration

To support the extended Secure Sockets Layer (SSL) functionality offered by Release 10.0B, new SSL-related administrative options are available for numerous OpenEdge client and server components.

New SSL client options are available for the following components:

- Progress® 4GL client.
- Web Services Adapter (WSA).
- AppServer™ Internet Adapter (AIA).
- WebSpeed® Messengers.

New SSL server options are available for the following components:

- WebSpeed®.
- AppServer.
- OpenEdge Adapter for SonicMQ®.
- OpenEdge RDBMS Server.

For more information, see:

Manuals:	*OpenEdge Getting Started: Security* *OpenEdge Application Server: Administration*
HTML online help:	Progress Explorer

OpenEdge Installation

This section provides a summary of the new feature of the Install for Release 10.0B.

Notification for Silent/Batch Install

Silent/Batch install has a new, optional parameter that enables status messages. The status messages display the current phase of the installation and the percentage complete.

For more information, see:

Manuals:	*OpenEdge Getting Started: Installation and Configuration for UNIX*
	OpenEdge Getting Started: Installation and Configuration for Windows

Security

Release 10.0B includes the following enhancements:

- HTTPS access to the AppServer through the AIA
- Native Secure Sockets Layer (SSL) connections between OpenEdge client and server components
- Cryptographic functions in the Progress 4GL

HTTPS access to the AppServer through the AIA

In previous releases of OpenEdge only Progress 4GL, WebSpeed, and Java™ Open Client applications allowed you to make a secure Internet connection to the AppServer through the AppServer Internet Adapter (AIA) using HTTPS. With Release 10.0B, .NET Open Clients also allow you to make a secure Internet connection to an AppServer thought the AIA using HTTPS.

Native Secure Sockets Layer (SSL) connections between OpenEdge client and server components

OpenEdge Release 10.0B provides security functions that use some features of a Public Key Infrastructure (PKI). This PKI support provides connections between most OpenEdge client and server components that are secured using a native Secure Sockets Layer (SSL) and also includes tools for managing key and certificate stores used by the OpenEdge SSL client and SSL server components.

The OpenEdge implementation of SSL provides data privacy and client-server authentication for the following OpenEdge client-server relationships:

- Progress 4GL application to database server running the OpenEdge™ RDBMS™.
- Progress 4GL application to AppServer or Progress OpenEdge Adapter for SonicMQ.
- Progress 4GL application socket to Progress 4GL server socket (or other non-OpenEdge SSL socket server).
- Non-OpenEdge socket client to Progress 4GL server socket.
- AppServer Internet Adapter (AIA) to AppServer or SonicMQ Adapter.
- .NET or Java Open Client to AppServer.
- Web Services Adapter (WSA) to AppServer, as configured for each WSA-managed Web service.
- OpenEdge™ Adapter for Sonic ESB™ to AppServer, as configured for each Sonic ESB Adapter-managed OpenEdge service.
- WebSpeed Messengers to WebSpeed® Transaction Server.
- JDBC and ODBC connections to the SQL Server.

Cryptographic functions in the Progress 4GL

Release 10.0B provides cryptographic and helper functionality in the Progress 4GL that supports:

- Progress 4GL security policies.
- Symmetric encryption and decryption with both random binary and password-based key generation.

- Message digests usable as Message Authentication Codes (MACs).
- Conversion between binary and Base 64 (character) representations of data to support transport and storage of encryption keys and data.

For more information, see:

Manuals:	*OpenEdge Getting Started: Security* *OpenEdge Deployment: Startup Command and Parameter Reference* *OpenEdge Data Management: Database Administration* *OpenEdge Application Server: Developing AppServer Applications* *OpenEdge Application Server: Administration* *OpenEdge Development: Open Client Introduction and Programming* *OpenEdge Data Management: SQL Development* *OpenEdge Development: Programming Interfaces* *OpenEdge Development: Progress 4GL Reference*
HTML online help:	Progress Explorer Startup Command and Parameter Reference Progress 4GL Reference

Startup Parameters

The following sections provide information on new and updated startup parameters:

- New security startup parameters
- New SQL startup parameters
- Updated OpenEdge Application Debugger startup parameters

New security startup parameters

Table 1–1 identifies the new security startup parameters that Release 10.0B supports.

Table 1–1: Security startup parameters

Parameter	Syntax	Description
Key Alias	`-keyalias` *`key-alias-name`*	Specifies the alias name of a Secure Sockets Layer (SSL) private key/digital certificate key-store.
Key Alias Password	`-keyaliaspasswd` *`key-alias-password`*	Specifies the encrypted Secure Sockets Layer (SSL) key alias password.
No Host Verify	`-nohostverify`	Turns off host verification for a Secure Sockets Layer (SSL) connection.
No Session Cache	`-nosessioncache`	Disables the Secure Sockets Layer (SSL) session caching.
No Session Reuse	`-nosessionreuse`	Avoids the reuse of a Secure Sockets Layer (SSL) session ID.
Session Timeout	`-sessiontimeout` *`n`*	Specifies the length of time a SSL session is held in SSL session cache.
SSL-based Connection	`-ssl`	Specifies a Secure Socket Layer (SSL) to all database and client connections.

For more information, see:

Manuals:	*OpenEdge Deployment: Startup Command and Parameter Reference* *OpenEdge Getting Started: Security*
HTML online help:	Startup Command and Parameter Reference

New SQL startup parameters

Table 1–2 identifies the new SQL startup parameters that Release 10.0B supports.

Table 1–2: SQL startup parameters

Parameter	Syntax	Description
Type of Server to Start	`-ServerType [4GL | SQL | Both]`	Specifies whether to start a 4GL server, SQL server, or both.
SQL Open Cursors	`-SQLCursors` *value*	Specifies the maximum number of cursors open at any one time.
SQL Stack Size	`-SQLStack` *n*	Specifies the size, in MB, of the SQL Stack.
SQL Statement Cache Size	`-SQLStmtCache` *num_entries*	Specifies the number of statements allowed in the SQL statement cache.
SQL Sorting Memory	`-SQLTempBuff` *value*	Defines the size of the temporary table buffer in memory.
SQL Sorting on Disk	`-SQLTempDisk` *value*	Defines the size of the temporary table for backup storage.
SQL Temp Table Data Page	`-SQLTempPgSize` *value*	Defines the size of the temporary table data page.

For more information, see:

Manual:	*OpenEdge Deployment: Startup Command and Parameter Reference*
HTML online help:	Startup Command and Parameter Reference

Updated OpenEdge Application Debugger startup parameters

Release 10.0B provides the following updated OpenEdge Application Debugger startup parameters:

- **Enable Attachable Debugging** (`-debugReady`) — Enables a Progress 4GL client, a single WebSpeed® Agent, or single AppServer™ Process to be attached to by the Debugger.
- **Log Entry Types** (`-logentrytypes`) — Specifies one or more log entry types to write to the log file.

  Table 1–3 identifies the new log entry types.

Table 1–3: New log entry types for the Application Debugger

Log entry type	Executable	Description

`AiAMgmt`
`AiaProp`
`AiaRqst`
`AiaUbroker`
`AiaDefault`
	AIA.	Turns on logging for the AIA component.

`FileID`	4GL clients, AppServer, and WebSpeed agents.	Turns on logging for file opening, file closing, and error messages that do not contain the file number.
`NSPlumbing`	NameServer.	Turns on logging for the NameServer component.

`UBroker.Basic`
`UBroker.ClientFSM`
`UBroker.ServerFSM`
`UBroker.ClientMsgStream`
`UBroker.ServerMsgStream`
`UBroker.ClientMsgQueue`
`UBroker.ServerMsgQueue`
`UBroker.ClientMemTrace`
`UBroker.ServerMemTrace`
`UBroker.ThreadPool`
`UBroker.Stats`
`UBroker.AutoTrim`
 	
`UBroker.All`
	Unified Broker.	Turns on logging for the Unified Broker component.

For more information, see:

Manuals:	*OpenEdge Deployment: Startup Command and Parameter Reference* *OpenEdge Development: Debugging and Troubleshooting*
HTML online help:	Startup Command and Parameter Reference

AppServer

Release 10.0B includes the following enhancements:

- Increased client support for session-free applications
- Direct connections to the AppServer using SSL tunneling
- Remote debugging by attaching the Debugger to an AppServer agent

Increased client support for session-free applications

You can now access the AppServer running in state-free operating mode, and therefore participate in session-free applications from:

- .NET Open Clients.
- Java Open Clients.
- The AppServer Internet Adapter (AIA).

Direct connections to the AppServer using SSL tunneling

You can now use the AppServer protocol to support direct and secure access to the AppServer (with or without mediation from the OpenEdge NameServer) using Secure Sockets Layer (SSL) tunneling. This feature allows a client, running as an SSL client, to establish an SSL connection directly to the AppServer, running as an SSL server. This SSL support includes all clients of the AppServer:

- 4GL clients (including WebClient™).
- .NET and Java Open Clients.
- AIA.
- WSA, as configured for each Web service.
- Sonic ESB Adapter, as configured for each OpenEdge service.

Release 10.0B also provides support for maintaining private key and public key certificate stores on the AppServer as well as support for maintaining public key certificate stores on the client for all OpenEdge-installed clients and adapters that access the AppServer.

Remote debugging by attaching the Debugger to an AppServer agent

You can now attach a session of the Application Debugger remotely to a running AppServer agent session. Attaching the Debugger there by allows you to step through and debug AppServer 4GL run-time code that is called from any AppServer client. This debugging technique allows you to debug AppServer code without having to modify that AppServer code in order to launch the Debugger.

For more information, see:

Manuals:	*OpenEdge Application Server: Developing AppServer Applications*
	OpenEdge Development: Web Services

WebSpeed

Release 10.0B includes the following enhancements for WebSpeed®:

- LONGCHAR support
- Using the Application Debugger with SpeedScript

LONGCHAR support

The LONGCHAR data type is now available in WebSpeed sessions. You use the new `{&OUT-LONG}` preprocessor and the new `get-cgi-long` function with this data type.

The `{&OUT-LONG}` preprocessor is defined in `src/web/method/cgidefs.i` as follows:

```
&GLOBAL-DEFINE OUT-LONG EXPORT {&WEBSTREAM}
```

The `get-cgi-long` function serves as a wrapper for the `GET-CGI-LONGCHAR-VALUE` method of the `WEB-CONTEXT` system handle.

For more information on the new language elements, see:

Manual:	*OpenEdge Development: Progress 4GL Reference*

Using the Application Debugger with SpeedScript

You can use the Application Debugger on SpeedScript applications running on WebSpeed agents. The process is the same as debugging an application running on an AppServer agent.

For more information on the new language elements, see:

Manuals:	*OpenEdge Application Server: Developing AppServer Applications* *OpenEdge Development: Debugging and Troubleshooting*

Progress Dynamics: Managers API Updates

Release 10.0B includes the following enhancements for Progress Dynamics®:

- Connection Manager updates
- Web service support in Progress Dynamics
- Repository Managers updates

Connection Manager updates

New APIs have been added to the Connection and Service Type Managers that allow you to pass parameter and substitution lists when you connect to a service. A new Service Type Manager has been added to support connecting to Web services.

For more information, see:

Web paper:	*Managers APIs Updates*

Web service support in Progress Dynamics

A new Service Type Manager has been added to Progress Dynamics to support connecting to Web services. The Service Type Maintenance tool now allows you to define Web services as service types in the Repository. To reduce unnecessary overhead during session startup, Web services can be set to not connect automatically at startup through the Logical Service Maintenance tool.

For more information, see:

Manual:	*OpenEdge Development: Progress Dynamics Administration*
Web paper:	*Managers APIs Updates*

Repository Managers updates

Changes have been made to the Repository Managers' APIs to support the new class extension functionality.

For more information, see:

Web paper:	*Managers APIs Updates*

Progress Dynamics: Calculated fields

In previous releases, calculated fields on dynamic SmartDataObjects (SDOs) and dynamic viewers were stored in the repository as master objects, and the calculated field master object names needed to be unique within the repository, and as a result, prevented their reuse.

In Release 10.0B, calculated fields are now included as entity instances and are retrieved and cached at run time as part of an entity. You can now include calculated field objects instances on multiple entities, dynamic (SDOs) and dynamic viewers.

To support the changes to calculated fields, the following tools have changed:

- The Data Field Maintenance tool was expanded to support maintenance of calculated fields and the maintenance of calculated field instances on entities.
- The AppBuilder Column Editor now provides the ability to add a calculated field and to select an existing calculated field for use on a dynamic SDO.

With the improvements to these tools, you can now add calculated fields to the repository as follows:

- Using the **Data Fields** tab in the **Entity Maintenance** window, you can maintain calculated fields for an entity. A master calculated field and its attributes are maintained as data field master field objects. A calculated field instance can be added to multiple entities.
- Using the **Column Editor** for a dynamic SDO, you can open the **Calculated Field Selector** window with the **Calculated Fields** button. From this window, you can choose a calculated field master object stored in the repository to add to the dynamic SDO. If the calculated field does not have an instance on one of the SDO's entities, it is added as an instance to the entity for the first updateable table of the SDO. If there are no updateable tables in the SDO, it is added to the entity for the first table of the SDO.

- From the **Calculated Field Selector** window, you can also create a new calculated field master object by choosing the **Create new calculated field** button to open the **Create Calculated Field** dialog box. When you add a field, it is added as an instance to the entity for the first updateable table of the SDO. If there are no updateable tables in the SDO, the field is added to the entity for the first table of the SDO.

Considerations when you upgrade to Release 10.0B

When you upgrade to Release 10.0B, an existing calculated field master object on a dynamic SDO is created as an instance on the SDO's first updateable table's entity, or, if there are no updateable tables in the SDO, on the first table's entity. However, if the instance would have been added to a repository entity, the instance will not be created and a message is written to the Dynamic Configuration Utility (DCU) log file. As a result, you must manually add the calculated field to an application table entity of the SDO after the upgrade.

There are four messages that will be written to the DCU log file, indicating that the DCU was unable to add calculated fields to repository entities for the following SDOs and entities:

SDO	Entity
gsmsxfullo	gsm_scm_xref
gsmssful2o	gsm_security_structure
gsmusdyno	gsm_user
gsmrlfullo	gsm_release

This is expected and required behavior. These calculated fields are applied to the appropriate repository entities during the application of the ADOs.

For more information, see:

HTML online help:	Progress Dynamics AppBuilder

Progress Dynamics: Print Preview support for XML and HTML output

The **Print Preview**  button from the standard Progress Dynamics browse toolbar can now export data to XML and HTML files, as well as to Crystal Reports®. While it defaults to XML output, you can choose between XML, HTML, and Crystal formats using the `print_preview_preference` session property. You can specify custom style sheets for XML and HTML reports with the `print_preview_stylesheet` session property.

For more information, see:

Manual:	OpenEdge Development: Progress Dynamics Administration

Progress 4GL Reference Updates for 10.0B

This section provides a summary of new and modified Progress 4GL Reference entries organized within the following categories:

- Data Types
- Diagnostics
- Internationalization
- ProDataSet
- Security—Cryptography
- Security—SSL connections
- Access to industry Web services
- Miscellaneous

Data Types

The following new and modified reference entries provide enhanced data type support:

- **BASE64 encoding and decoding:**
  - COPY-LOB statement (BASE64 option removed)
  - BASE64-DECODE function
  - BASE64-ENCODE function
- **COM and DLL support for new data types:**
  - DEFINE PARAMETER statement
- **Support for LONGCHAR data type:**
  - PUT-STRING statement
  - LEFT-TRIM function
  - RIGHT-TRIM function
  - TRIM function
  - FORM-LONG-INPUT attribute (WEB-CONTEXT system handle)
  - GET-CGI-LONGCHAR-VALUE method (WEB-CONTEXT system handle)
  - { } Preprocessor name reference (new WebSpeed preprocessor)
- **Miscellaneous:**
  - DATETIME-TZ function (*timezone-exp* argument is optional)

Diagnostics

The following modified reference entries provide enhanced logging support:

- **New log entry types for FileID, AIA, Unified Broker, and NameServer:**
  - LOG-ENTRY-TYPES attribute

Internationalization

The following new and modified reference entries provide support for Unicode normalization and new collation strengths and rules for linguistic sorting (using the International Components for Unicode):

- **Unicode normalization:**
  - NORMALIZE function
- **New collation strengths and rules:**
  - COMPARE function
  - FOR statement
  - OPEN QUERY statement
  - PRESELECT phrase
  - CPCOLL attribute
- **Character string comparisons and new collation rules:**
  - EQ or = operator
  - GE or >= operator
  - GT or > operator
  - LE or < = operator
  - LT or < operator
  - NE or <> operator

ProDataSet

The following new and modified reference entries provide enhanced ProDataSet support:

- **Handles:**
  - Buffer object handle
  - Data-source object handle

  - ProDataSet object handle
  - Temp-Table object handle
- **Attributes:**
  - AUTO-SYNCHRONIZE attribute
  - BATCH-SIZE attribute
  - DATA-SOURCE-MODIFIED attribute
  - ERROR attribute
  - IGNORE-CURRENT-MODIFIED attribute
  - LAST-BATCH attribute
  - MERGE-BY-FIELD attribute
  - PREFER-DATASET attribute
  - QUERY-OFF-END attribute
  - REJECTED attribute
  - ROW-STATE attribute
  - TRACKING-CHANGES attribute
- **Methods:**
  - APPLY-CALLBACK method
  - COPY-DATASET method
  - COPY-TEMP-TABLE method
  - MERGE-CHANGES( ) method
  - MERGE-ROW-CHANGES( ) method
  - SAVE-ROW-CHANGES method
  - SET-CALLBACK-PROCEDURE( ) method
  - SYNCHRONIZE() method

- **Events:**
  - FIND-FAILED event
  - OFF-END event
  - SYNCHRONIZE event
- **Statements:**
  - DEFINE TEMP-TABLE statement (BEFORE-TABLE option missing in 10.0A)
- **Functions:**
  - DATA-SOURCE-MODIFIED function
  - ERROR function
  - REJECTED function
  - ROW-STATE function

Security—Cryptography

The following new reference entries provide cryptography support:

- **Handle:**
  - SECURITY-POLICY system handle
- **Attributes:**
  - ENCRYPTION-SALT attribute
  - PBE-HASH-ALGORITHM attribute
  - PBE-KEY-ROUNDS attribute
  - SYMMETRIC-ENCRYPTION-ALGORITHM attribute
  - SYMMETRIC-ENCRYPTION-IV attribute
  - SYMMETRIC-ENCRYPTION-KEY attribute
  - SYMMETRIC-SUPPORT attribute

- **Functions:**
  - DECRYPT function
  - ENCRYPT function
  - GENERATE-PBE-KEY function
  - GENERATE-PBE-SALT function
  - GENERATE-RANDOM-KEY function
  - MD5-DIGEST function
  - SHA1-DIGEST function

Security—SSL connections

The following new and modified reference entries provide SSL connection support:

- **Attributes:**
  - SSL-SERVER-NAME attribute
- **Methods:**
  - CONNECT( ) method (AppServer)
  - CONNECT( ) method (Socket Object)
  - CONNECT( ) method (Web Service)
  - ENABLE-CONNECTIONS( ) method
- **Functions:**
  - SSL-SERVER-NAME function

Access to industry Web services

The following modified reference entries provide enhanced Web Services support:

- **Handles:**
  - Server object handle
  - THIS-PROCEDURE system handle
- **Attributes:**
  - FIRST-PROCEDURE attribute
  - LAST-PROCEDURE attribute
  - NEXT-SIBLING attribute
  - PREV-SIBLING attribute

Miscellaneous

The following reference entries were also modified:

- **Mouse wheel support for the browse widget**
- **Additional image format support**
  - Image phrase
- **HTML Help**
  - SYSTEM-HELP statement (reference entry reorganized)
- **WEB-CONTEXT system handle**

  These reference entries, previously incomplete in Release 10.0A, have been completed in Release 10.0B:
  - AUTO-DELETE-XML attribute
  - CONFIG-NAME( ) attribute
  - CURRENT-ENVIRONMENT attribute
  - EXCLUSIVE-ID attribute

- FORM-INPUT attribute
- FORM-LONG-INPUT attribute
- HANDLE attribute
- HTML-CHARSET attribute
- HTML-END-OF-LINE attribute
- HTML-END-OF-PAGE attribute
- HTML-FRAME-BEGIN attribute
- HTML-FRAME-END attribute
- HTML-HEADER-BEGIN attribute
- HTML-HEADER-END attribute
- HTML-TITLE-BEGIN attribute
- HTML-TITLE-END attribute
- INSTANTIATING-PROCEDURE attribute
- IS-XML attribute
- SESSION-END attribute
- TYPE attribute
- VALIDATE-XML attribute
- X-DOCUMENT attribute
- XML-SCHEMA-PATH attribute
- XML-SUPPRESS-NAMESPACE-PROCESSING attribute
- GET-CGI-LIST( ) method
- GET-CGI-LONGCHAR-VALUE( ) method
- GET-CGI-VALUE( ) method
- GET-CONFIG-VALUE( ) method

- INCREMENT-EXCLUSIVE-ID( ) method
- URL-DECODE( ) method
- URL-ENCODE( ) method

For more information, see:

Manual:	*OpenEdge Development: Progress 4GL Reference*
HTML online help:	Progress 4GL Reference

OpenEdge Application Development Tools

This section highlights the following changes to OpenEdge Application Development tools in 10.0B:

- AppBuilder FILL-IN property sheet
- Improved support for calculated fields
- Support for Web Services
- Class maintenance tool
- TEMP-DB maintenance tool
- Support for encryption
- Changes to windows and dialog boxes for new data types

AppBuilder FILL-IN property sheet

The AppBuilder FILL-IN property sheet now has a **Password-Field** option. Select this option when you want a password field to display asterisks (*) instead of a blank field when users enter a password.

Improved support for calculated fields

Changes were made to the following tools to improve the creation and maintenance of calculated fields:

- The Data Field Maintenance tool was expanded to support maintenance of calculated fields and the maintenance of calculated field instances on entities.
- The AppBuilder Column Editor now provides the ability to add a calculated field and to select an existing calculated field for use on a dynamic SDO.

Support for Web Services

New options have been added to the following development tools to support the development of Web services applications:

- The **Service Type Control** window in the **Administration** window of the Progress Dynamics AppBuilder now includes Web service as a **Service Type** control.
- The **Physical Services Maintenance** window accessed from the **Administration** window of the Progress Dynamics AppBuilder now includes the following connection parameters for Web services:
  - WSDL document.
  - WSDL UserID.
  - WSDL Password.
  - Service and Port.
  - Binding and SOAP endpoint.
  - Service name.
  - Service namespace.
  - Port name.
  - Binding name.
  - Binding namespace.
  - URL endpoint.
  - SOAPEndpointment User ID.

- SoapEndpoint Password.
- Target namespace.
- Maximum connections.
- Parameter filename.

Class maintenance tool

A new Class Maintenance tool has been added to Progress Dynamics. This tool provides support for extending objects classes to define new objects. Access this tool from the Progress Dynamics AppBuilder **Build** menu.

TEMP-DB maintenance tool

A new TEMP-DB tool has been added to the AppBuilder and Progress Dynamics AppBuilder. You use this tool to maintain the TEMP-DB schema instead of using the Data Dictionary. Using the TEMP-DB Maintenance tool, you can define Temp-Table structures in source include files using 4GL statements and then use the TEMP-DB Maintenance tool to update the TEMP-DB schema and synchronize the source files with the table schema. Access this tool from the Progress Dynamics AppBuilder and AppBuilder **Tools** menu.

Support for encryption

The **Physical Services Maintenance** window accessed from the Progress Dynamics **Administration** window, and the Service Parameter Maintenance PRO*Tool accessed from AppBuilder PRO*Tools now includes the following connection parameters to support encryption for AppServers and Web services:

- SSL enabled (-SSL).
- Session reuse disabled (-nosessionreuse).
- Host verification disabled (-nohostverify).

Changes to windows and dialog boxes for new data types

Options have been changed or added to the following windows and dialog boxes to support the new data type in Release 10.0B:

- `FILL-IN` property sheet now supports new date time and date time timezone options.
- `Editor` property sheet now supports the `LongChar` data type for editors.

- `Static SmartDataObject` (SDO) property sheet now includes a **Use NO-UNDO for RowObject** option.
- The property sheet for a data logic procedure now includes **Use NO-UNDO for RowObject** option.
- The Object Palette now includes a **SmartLOB** options and a new `SmartLOB instance` property sheet provides information about an instance of a `SmartLOB`.

For more information, see:

Manual:	*OpenEdge Development: Progress 4GL Reference*
Web papers:	*New Data Types* *Customizing Classes* *TEMP-DB Tool*
HTML online help:	Progress Dynamics AppBuilder Progress AppBuilder Common Dialog Boxes and Editing Windows PRO*Tools

New data types in the ADM and the OpenEdge Application Development Environment

Support has been added for the following Large Object (LOB) data types:

- Character Large Objects (CLOB)
- Binary Large Objects (BLOB)
- DateTime (DT)
- Date Time-Tz

To support these new data types, Release 10.B includes

- New and changed ADM APIs for new data types
- Changes to windows and dialog boxes for new data types

New and changed ADM APIs for new data types

To support the new Large Object (LOB) data types, the following APIs and options are new or modified in Release 10.0B:

- AssignLists
- DisplayedFields
- collectChange
- createData
- RowObject TempTable
- submitRow
- UpdatableColumns ByTables
- UpdateData
- Pass by reference
- columnLongcharValue (new DataObject API)
- copyLargeColumnToMemptr (new DataObject API)
- copyLargeColumnToFile (new DataObject API)
- FieldModified
- widgetIsModified

Changes to windows and dialog boxes for new data types

Options have been changed or added to the following windows and dialog boxes in the development tools to support the new data types in Release10.0B:

- `FILL-IN` property sheet now supports new **Date Time** and **Date Time-Tz** (timezone) options.
- `Editor` property sheet now supports the **LongChar** data type for editors.
- `Static SmartDataObject` (SDO) property sheet now includes a **Use NO-UNDO for RowObject** option.
- The property sheet for a data logic procedure now includes U**se NO-UNDO for RowObject** option.
- The Object Palette now includes a **SmartLOB** options and a new SmartLOB instance property sheet provides information about an instance of a SmartLOB.

For more information, see:

Web paper:	*New Data Types*
HTML online help:	Progress Dynamics AppBuilder Progress AppBuilder Common Dialog Boxes and Editing Windows

Password fields

Login dialog boxes now use the Progress 4GL `PASSWORD-FIELD` attribute to display asterisks (*) in the password field of a login dialog box. Password fields now display asterisks (*) when a user enters a password for the following:

- Character database connection procedure.
- **Graphical user interface database connection** dialog box.
- **AppServer utility security** dialog box.
- Sample RESULTS login procedure.
- **Utility-user kit** database window.

- **JMS broker login** dialog box.
- **Dynamics change user password** dialog box.
- **Dynamics application** login dialog box.
- **Dynamics suspend** dialog box.
- RESULTS login procedure.
- **Database login** dialog box.

AppBuilder FILL-IN property sheet

The AppBuilder `FILL-IN` property sheet now has a **Password-Field** option. Select this option when you want a password field to display asterisks (*) instead of a blank field when users enter a password.

For more information, see:

Manual:	*OpenEdge Development: Progress 4GL Reference*
HTML online help:	Progress Dynamics AppBuilder Progress AppBuilder

Sonic ESB Adapter

The set of service properties for 4GL applications exposed as Sonic ESB services includes new SSL client properties that support secure communications between the Sonic ESB Adapter and the AppServer.

For more information, see:

Manuals:	*OpenEdge Getting Started: Security* *OpenEdge Application Server: Administration*

SonicMQ Adapter

Release 10.0B enhancements to the SonicMQ Adapter include the following:

- **Newly supported data types** — The 4GL-JMS API now supports the following 4GL data types: `DATE`, `DATETIME`, `DATETIME-TZ`, and `LONGCHAR`.
- **SSL support** — The SonicMQ Adapter can be configured to accept SSL connections from a 4GL client.

For more information, see:

Manuals:	*OpenEdge Development: Messaging and ESB* *OpenEdge Application Server: Administration* *OpenEdge Getting Started: Security*
HTML online help:	Progress Explorer

WebClient

Release 10.0B introduces a new tool, the WebClient Deployment Packager, that simplifies the process of customizing an application for deployment in an environment different from the one in which it was developed and generated. The tool extracts configuration information from an existing `.prowcapc` file, allows the user to edit it through a graphical user interface, and stores the modified configuration in a new `.prowcapc` file.

The Deployment Packager is an integral part of OpenEdge™ Studio. In addition, OpenEdge Studio includes a single `.zip` file that expands to a standalone version of the Deployment Packager. Application providers can distribute this standalone version to customers who want to customize their own deployment configurations.

For more information, see:

Manual:	*OpenEdge Deployment: WebClient Applications*
HTML online help:	PRO*Tools

Web services

Release 10.0B offers enhanced support for Web services as follows:

- Progress 4GL Web services
- Accessing industry Web services from the 4GL
- Web service support in Progress Dynamics

Progress 4GL Web services

Enhanced features for Progress 4GL Web services include:

- **Newly supported data types** — The SOAP-4GL API now maps the following 4GL data types to XML Schema data types: `BLOB`, `CLOB`, `DATETIME`, `DATETIME-TZ`, `LONGCHAR`.
- **Array parameters** — Arrays (extents) can now be passed as parameters between 4GL Web services and their client applications.

For more information, see:

Manual:	*OpenEdge Development: Web Services*

Accessing industry Web services from the 4GL

Enhanced features for accessing Web services from the Progress 4GL include:

- Additional startup option support for a WSDL Analyzer accessing a WSDL file using HTTPS.
- Changes to the WSDL Analyzer output, including:
  - Inclusion of Analyzer startup options in the "Connect details" sections of the Analyzer output.
  - Representation of the WSDL file location in Analyzer output as an absolute file path.
  - Addition of a new operation index document in the Analyzer output and reorganization of the Analyzer output to provide a more consistent user interface.
  - Changes in the output to provide both procedure and user-defined function prototypes for Web service operations that can be represented as both procedures and user-defined functions.

- Additional connection parameter support for Web services accessed using HTTPS.
- Enhanced port type access, allowing a single port type supported by a given Web service binding to use more than one procedure object. This allows the definition of more than one set of header callbacks for a single port type.
- Changes to existing procedure object attributes to allow for using multiple procedure objects to map a single port type.
- Enhanced access to Web service operations that allows all Web service operations to be accessed as 4GL internal procedures, allowing all Web service operations to be invoked asynchronously. Web service operations that the WSDL Analyzer determines can be accessed as user-defined functions, the Analyzer also documents how to access as user-defined functions in addition to access as internal procedures.
- Enhanced support for using the WSAViewer SOAP viewer without having to modify 4GL code.

For more information, see:

Manual:	*OpenEdge Development: Web Services*

Web service support in Progress Dynamics

A new Service Type Manager has been added to Progress Dynamics to support connecting to Web services. The Service Type Maintenance tool now allows you to define Web services as service types in the Repository. To reduce unnecessary overhead during session startup, Web services can be set to not connect automatically at startup through the Logical Service Maintenance tool.

For more information, see:

Manual:	OpenEdge Development: Progress Dynamics Administration
Web paper:	*Connection Manager Updates*

Dynamic Treeview for Web applications

In Release 10.0B, the Dynamic Treeview is supported for Web applications. This support is fundamentally the same as support for Graphical User Interfaces with the following exceptions and considerations:

- Super procedures that are associated with 4GL client code must be written with client-side Javascript.
- Viewer objects can only be dynamic viewers. There is no support for static viewers.
- Check boxes are not supported.
- The best bitmap format for images on the Web is the `.gif` format.
- You must use CSS style sheets to adjust the height and width of images.
- You cannot specify additional parameters for launched containers.
- Fields to store functionality is not supported.
- Web Treeview does not support Primary Data Object for Extract Program scenarios.

For more current information, see:

Reference:	OpenEdge Release10.0B Release Notes

.NET Open Client

This section provides an overview of the new and updated features to the .NET Open Client for the 10.0B release.

Session-free support

The Open Client supports two session models:

- **Session-managed** — Where a client maintains a dedicated physical connection to a single AppServer that handles each request for a given application service.
- **Session-free** — Where a client maintains a logical connection to a given application service, and each client request for that application service can be handled by any one of many AppServer resources.

HTTP/HTTPS support

Release 10.0B includes support of HTTP and HTTPS protocols on Microsoft .NET clients. You can use a URL with HTTP or HTTPS to connect to an AppServer from a .NET client via the AppServer Internet Adapter.

SSL support

To provide support for Secure Sockets Layer (SSL) with .NET Open Client, this release includes:

- A new assembly, `Progress.ssl.dll`.
- The `noSessionReuse` property that determines whether to reuse session IDs.
- The `SSLSubjectName` common method which provides the internal AppServer object with access to the SSL server's subject name that is obtained from its validated digital certificate.

Assemblies

You now choose from two sets of assemblies that make up the .NET Open Client:

- **Strong-named and digitally-signed** — For tightly secure .NET applications. If you use this set of assemblies, you must use the exact same Open Client Runtime assemblies for deployment and proxy generation.
- **Digitally-signed only** — For .NET applications where security is not a high concern. If you use this set of assemblies, you can simply give your users the updated OpenEdge assemblies (You do not need to regenerate your proxies.).

ProxyGen methods

ProxyGen generates the following new methods:

- **Request ID** (`_getRequestId`) — Returns a unique string, identifying the request most recently run on a thread.
- **Procedure return string** (`_getProcReturnString`) — Retrieves the return value of the method most recently run on a thread.
- **SSL subject name** (`_getSSLSubjectName`) — Provides the internal AppServer object with access to the SSL server's subject name that is obtained from its validated digital certificate.

Data type support

The .NET Open Client now includes support for passing arrays and the following data types as parameters: `DATETIME`, `DATETIME-TZ`, `LONGCHAR`, `CLOB` and `BLOB`.

Proxy property accessor methods

You can now access proxy properties with two types of methods:

- General property accessor methods (`get` and `set`).
- Purposed property accessor methods (individual methods specific to each property).

Schema Marshal methods

The new `getNoSchemaMarshal` and `setNoSchemaMarshal` methods for the `Progress.Open4GL.ProDataTable` class let you specify to omit all schema information when marshaling data from the .NET Open Client to the AppServer.

For more detailed information on .NET Open Client, see:

Manuals:	*OpenEdge Development: .NET Open Clients*
	OpenEdge Development: Open Client Introduction and Programming

Java Open Client

This section provides an overview of the new and updated features to the Java Open Client for the 10.0B release.

Session-free support

The Open Client supports two AppServer session models:

- **Session-managed** — Where a client maintains a dedicated physical connection to a single AppServer that handles each request for a given application service.
- **Session-free** — Where a client maintains a logical connection to a given application service, and each client request for that application service can be handled by any one of many AppServer resources.

SSL support

To provide support for Secure Sockets Layer (SSL) with Java Open Client, the Release 10.0B includes:

- The `noSessionReuse` property that determines whether to reuse session IDs.

- The SSLSubjectName common method which provides the internal AppServer object with access to the SSL server's subject name that is obtained from its validated digital certificate.
- Support for the new protocols, AppServerS and AppServerDCS.

Proxy methods

ProxyGen generates the following new methods:

- **Request ID** (_getRequestId) — Returns a unique string, identifying the request most recently run on a thread.
- **Procedure return string** (_getProcReturnString) — Retrieves the return value of the method most recently run on a thread.
- **SSL subject name** (_getSSLSubjectName) — Provides the internal AppServer object with access to the SSL server's subject name that is obtained from its validated digital certificate.

Connection class

The Java Open Client Runtime now includes a Connection class. The Connection class stores AppServer connection information, which can be passed to the AppObject constructor when connecting to the AppServer. You can change the value of proxy properties by creating an instance of the Connection class (called a Connection object). You can then use the Connection object to programmatically override the property values of the RunTimeProperties class.

Data type support

The Java Open Client Runtime now includes support for passing arrays and the following data types as parameters: DATETIME, DATETIME-TZ, LONGCHAR, CLOB and BLOB.

Proxy property accessor methods

You can now access proxy properties with two types of methods:

- General property accessor methods (`get` and `set`).
- Purposed property accessor methods (individual methods specific to each property).

Schema Marshal methods

The new `getNoSchemaMarshal` and `setNoSchemaMarshal` methods on the `com.progress.open4gl.InputResultSet` class let you specify whether to omit schema information for static temp-table parameters when marshaling data from the Java Open Client to the AppServer.

For more detailed information on Java Open Client, see:

Manuals:	*OpenEdge Development: Java Open Clients*
	OpenEdge Development: Open Client Introduction and Programming

External Program Interfaces

This section describes the updates in Release 10.0B that support external program interfaces, including:

- ActiveX control data type support
- COM object data type support
- Dynamic Link Library support
- Sockets support

ActiveX control data type support

Release 10.0B includes support for arrays as parameters and support for LONGCHAR, DATETIME, and DATETIME-TZ data types for ActiveX controls.

COM object data type support

Release 10.0B includes support for LONGCHAR, DATETIME, and DATETIME-TZ data types for COM objects.

Dynamic Link Library support

Release 10.0B includes support for the LONGCHAR data type and support for arrays as parameters for DLL functions.

Sockets support

Release 10.0B includes the following support for sockets in the 4GL:

- Connecting to a Secure Sockets Layer (SSL) socket server using a socket object accessed from a 4GL socket client.
- Enabling connections to a 4GL SSL socket server using a server socket object accessed from a 4GL socket server.

For more information, see:

Manual:	*OpenEdge Development: Programming Interfaces*

Internationalization

The following new features provide additional support for internationalizing applications in the 10.0B release:

- **GB18030 code page** — A code page that extends the GB2312 (Chinese) code page and includes all characters defined in Unicode.
- **Linguistic sorting** — The sorting of Unicode characters based on the language-specific requirements for a locale (rather than the binary value of characters in UTF-8).

- **Unicode normalization** — The conversion of character data into standardized Unicode forms.

For more information, see:

Web paper:	*New Features for Internationalizing Applications in OpenEdge 10.0B*

OpenEdge Application Debugger

Release 10.0B provides the following enhancements to the Application Debugger:

- **Attaching to an AppServer process** — The Debugger can attach to an AppServer process running on the same machine or on the same LAN as the Debugger.
- **Watchpoints** — A breakpoint set to interrupt program execution when the value of a variable, buffer field, or attribute reference changes.

For more information, see:

Manual:	*OpenEdge Development: Debugging and Troubleshooting*

Diagnostics and Troubleshooting

Release 10.0B expands the existing logging infrastructure to support four new logging types:

- **AIA logging** — Specifying the AIA log entry type turns on logging for different AIA actions based on logging level.
- **FileID logging** — Specifying the `FileID` log entry type turns on logging for file opening, file closing, and error messages that do not contain the filename.

- **NameServer logging** — Specifying the NameServer (`NSPlumbing`) log entry type turns on logging for different NameServer actions based on logging level.
- **Unified Broker logging** — Specifying the Unified Broker log entry types turn on logging for different Unified Broker actions based on logging level.

For more information, see:

Manual:	*OpenEdge Development: Debugging and Troubleshooting*

OpenEdge DataServer for Microsoft SQL Server

Release 10.0B includes the following new features that provide additional support for improving the performance of OpenEdge™ DataServer for Microsoft® SQL Server applications:

- Connection Pooling
- "FireHose Cursor" FAST-FORWARD Only Record Sets

Connection Pooling

The DataServer for MS SQL Server is enhanced with the ability to form a connection pool. A *connection pool* is a set of database connections that are available for an application to use without having to be reestablished. Connection pooling significantly improves the cursor management associated with multi-table joins.

"FireHose Cursor" FAST-FORWARD Only Record Sets

FireHose Cursors deliver a streamlined, unmanaged, cursor-processing mechanism that handles result sets from a DataServer query. FireHose cursors will be the default mechanism for handling read-only results.

For more information, see:

Web paper:	*New Features for the DataServer for MicroSoft SQL Server in OpenEdge 10.0B*

DataServer for ODBC

Release 10.0B includes the following new feature that provides additional support for connecting to a DB2/400 data source:

Multiple DB2/400 Library Access

The OpenEdge™ DataServer for ODBC allows the user to specifically select DB2/400 as the foreign data source, and to further specify a library name when a connecting to a DB2/400 data source.

For more information, see:

Web paper:	*New Features for the DataServer for ODBC in OpenEdge 10.0B*
HTML online help:	Data Administration

OpenEdge RDBMS

This section provides a summary of new and enhanced features of the OpenEdge RDBMS for Release 10.0B:

- **Failover Clusters** — Support for failover clusters, formerly through the Fathom™ Clusters product, is integrated into the OpenEdge Enterprise RDBMS product.
- **IDXBUILD** — The IDXBUILD qualifier to PROUTIL is enhanced to allow you three additional ways to specify the indexes to rebuild: by area, by schema owner, and by table.
- **IDXCHECK** — The IDXCHECK qualifier to PROUTIL is enhanced to allow you three additional ways to specify the indexes to check: by area, by schema owner, and by table.
- **PROSTRCT** — The PROSTRCT utility is enhanced with the REORDER AI qualifier. The REODER AI qualifier enables you to reorder after-image extents so that empty extents immediately follow the current extent in switch order.

- **RFUTIL** — The RFUTIL utility is enhanced with the EXTRACT qualifier. The EXTRACT qualifier provides you the ability to extract the active AI notes from a full after-image extent.

- **SSL Connections** — The RDBMS brokers and servers are enhanced to accept Secure Sockets Layer (SSL) connections.

For more information, see:

Manuals:	*OpenEdge Data Management: Database Administration* *OpenEdge Getting Started: Security* for SSL and other OpenEdge security features

Support for Mozilla

Release 10.0B provides support for the Mozilla browser on MacIntosh computers.

Release 10.0B Documentation map by technology

Release 10.0B Documentation map by technology

Table 1–4 lists the documentation available for each technology area.

Table 1–4: Documentation map by technology *(1 of 3)*

To learn about . . .	See . . .
Installation and configuration	*OpenEdge Getting Started: Installation and Configuration for UNIX* *OpenEdge Getting Started: Installation and Configuration for Windows* *OpenEdge Deployment: Startup Command and Parameter Reference*
OpenEdge Studio	*OpenEdge Getting Started: Progress OpenEdge Studio* *OpenEdge Development: AppBuilder* *OpenEdge Development: Basic Database Tools* *OpenEdge Development: Basic Development Tools*
WebSpeed	*OpenEdge Getting Started: WebSpeed* *OpenEdge Application Server: Developing WebSpeed Applications*
Progress Dynamics	*OpenEdge Development: Progress Dynamics Getting Started* OpenEdge Development: Progress Dynamics Administration *OpenEdge Development: Progress Dynamics Basic Development* *OpenEdge Development: Progress Dynamics Advanced Development* *OpenEdge Development: Progress Dynamics Web Development Guide* *OpenEdge Development: Progress Dynamics Managers API Reference* *OpenEdge Development: Progress Dynamics Repository Reference*

Table 1–4: Documentation map by technology *(2 of 3)*

To learn about . . .	See . . .
Developing OpenEdge Applications	*OpenEdge Development: Progress 4GL Handbook* *OpenEdge Development: Programming Interfaces* *OpenEdge Development: Progress 4GL Reference* *OpenEdge Development: ProDataSets*
Service Oriented Architectures (SOA)	*OpenEdge Getting Started: Application and Integration Services* *OpenEdge Application Server: Administration* *OpenEdge Application Server: Developing AppServer Applications*
ADM and SmartObjects	*OpenEdge Development: ADM and SmartObjects* *OpenEdge Development: ADM Reference*
Open Client interfaces to a Java or .NET application or a Web service definition	*OpenEdge Development: Open Client Introduction and Programming* *OpenEdge Development: Java Open Clients* *OpenEdge Development: .NET Open Clients* *OpenEdge Development: Web Services*
Sonic messaging and application integration environment	*OpenEdge Development: Messaging and ESB*
Security	*OpenEdge Getting Started: Security*
Internationalization and localization	*OpenEdge Development: Internationalizing Applications* *OpenEdge Development: Translation Manager* *OpenEdge Development: Visual Translator*
Debugging	*OpenEdge Development: Debugging and Troubleshooting*
Deployment	*OpenEdge Deployment: Managing 4GL Applications* *OpenEdge Deployment: WebClient Applications* *OpenEdge Deployment: Application Portability*

Table 1–4: Documentation map by technology *(3 of 3)*

To learn about . . .	See . . .
Data Management	*OpenEdge Data Management: Database Design* *OpenEdge Data Management: Database Administration* *OpenEdge Data Management: DataServer for Microsoft SQL Server* *OpenEdge Data Management: DataServer for ODBC* *OpenEdge Data Management: DataServer for ORACLE* *OpenEdge Data Management: SQL Development* *OpenEdge Data Management: SQL Reference*
Reporting	*OpenEdge Reporting: Deploying Crystal Reports* *OpenEdge Reporting: Report Builder Deployment* *OpenEdge Reporting: Query/Results Administration and Development* *OpenEdge Reporting: Query/Results for UNIX* *OpenEdge Reporting: Query/Results for Windows*

2

Feature Comparisons

This chapter provides comparison about how the behavior of features in Release 10.0A differs in the Release 10.0B product.

This chapter includes information on:

- Progress Dynamics
- AppServer Internet Adapter (AIA)
- Open Clients
- ProDataSets

Progress Dynamics

The following features have changed in Release 10.0B:

DCU on run-time client

In previous versions, the Progress Dynamics Configuration Utility (DCU) required a compiler license to run. This requirement existed because some of the fix programs that the DCU applied needed to be compiled on the fly at run time.

Release 10.0B has removed this requirement. You can now run the DCU on a client site that does not have the Progress 4GL compiler installed.

Calculated fields

When you upgrade to version Release 10.0B, an existing calculated field master object on a dynamic SmartDataObject (SDO) is created as an instance on the SDO's first updateable table's entity, or, if there are no updateable tables in the SDO, on the first table's entity. However, if the instance would have been added to a repository entity, the instance will not be created and a message is written to the Dynamic Configuration Utility (DCU) log file. As a result, you must manually add the calculated field to an application table entity of the SDO after the upgrade.

There are four messages that will be written to the DCU log file indicating that the DCU was unable to add calculated fields to repository entities for the following SDOs and entities:

SDO	Entity
gsmsxfullo	gsm_scm_xref
gsmssful2o	gsm_security_structure
gsmusdyno	gsm_user
gsmrlfullo	gsm_release

This is expected and required behavior. These calculated fields are applied to the appropriate repository entities during the application of the ADOs.

AppServer Internet Adapter (AIA)

The function of the `httpsEnabled` property in the `ubroker.properties` file (**HTTPS enabled** check box in the AIA properties dialog box of the Progress Explorer) has changed for Release 10.0B.

In Release 10.0A, if you set this property to 1 (the **HTTPS enabled** check box is checked in the Progress Explorer) any 4GL or Open Client was required to access the Web server using HTTPS in order to access the AppServer using the AIA. If the client used HTTP, the connection to the AppServer was rejected. Also, the default value for this property was 1 (**HTTPS enabled** check box was checked).

In Release 10.0B, this restriction has been lifted and the function of the `httpsEnabled` property changed as follows. If this property is set to 1 (the **HTTPS enabled** check box is checked in the Progress Explorer) and a 4GL or Open Client attempts to connect to the Web server using HTTP, the AIA silently returns an HTTPS protocol URL to the client, which retries the connection using the returned URL. Also, the default value for this property is 0 (**HTTPS enabled** check box is cleared).

For more complete information, see:

Manuals:	*OpenEdge Application Server: Administration* *OpenEdge Application Server: Developing AppServer Applications*
HTML online help:	Progress Explorer

Open Clients

The following features have changed in Release 10.0B:

ProxyGen support for handling 4GL unknown values (?) in OpenClient method parameters

In Release 10.0A, the ProxyGen user interface included an **Allow Unknown for Parameter and Return Values** check box on the **AppServer** tab and similar check boxes in other dialog boxes of the UI to indicate how unknown values were handled. Also, the online help indicated that these check boxes determined if the Open Client interface would allow the passing of unknown values as specified.

In Release 10.0B, this check box on the **AppServer** tab has been relabeled as **Enable Unknowns for Parameter and Return Values**, and similar check boxes have been relabeled accordingly in other dialog boxes of the UI. The online help has also been updated to indicate that these check boxes determine *how* the Open Client interface will support the passing of 4GL unknown values (not *if* it will allow unknown values to be passed). If the 4GL passes unknown values in procedure or user-defined function parameters, the Open Client interface does not prevent it, but only enables or disables the ability of .NET or Java Open Client proxies to recognize the unknown values from the 4GL. Therefore, the ProxyGen UI and online help have been changed to reflect this.

Options for specifying .NET Open Client Runtime assemblies

In Release 10.0A, the .NET Open Client Runtime assembly was a strong-named assembly and required both strong-naming and signing for all other assemblies in the Open Client application. This meant that you always had to regenerate the proxies and rebuild the application to apply any updates to the application, but the application deployment was highly secure.

In Release 10.0B, you have the choice of generating .NET Open Client proxies and deploying the application using strong-named or signed-only Open Client Runtime assemblies. With a signed only deployment, you only need to rebuild the files that have changed in order to update the application, but the deployment is less secure.

Client support for accessing a session-free AppServer

In Release 10.0A, you could access a session-free application service using only the 4GL client or a Web service client through the Web Services Adapter (WSA).

In Release 10.0B, you can access the same session-free application service using a .NET or Java Open Client.

Client support for new 4GL data types passed as parameters

In Release 10.0A, you could access AppServer application services using the following new 4GL data types using only a 4GL client:

- Arrays as parameters
- DATETIME
- DATETIME-TZ
- LONGCHAR
- CLOB and BLOB (as fields in a temp-table parameter only)

In Release 10.0B, you can access the same AppServer application services using .NET or Java Open Clients and Web service clients accessing Progress 4GL Web services (Web service Open Clients).

For more complete information, see:

Manuals:	*OpenEdge Development: Open Client Introduction and Programming* *OpenEdge Development: .NET Open Clients* *OpenEdge Development: Java Open Clients* *OpenEdge Development: Web Services*
HTML online help:	Open Client Proxy Generator (ProxyGen)

ProDataSets

The following topics describe changes in functionality first introduced in OpenEdge Release 10.0A:

- NO-UNDO restriction lifted
- Keyword attributes now allowed in WHERE clause
- New default behavior in SAVE-ROW-CHANGES
- BLOB/CLOB changes not always tracked

There is a new Expert Series book on ProDataSets that you can reference for more information. *OpenEdge Development: ProDataSets* replaces and extends the series of white papers on ProDataSets authored by John Sadd for OpenEdge Release 10.0A. The book describes and demonstrates effective techniques for ProDataSet development.

For more complete information about ProdataSets in Release 10.0B, see:

Manual:	*OpenEdge Development: ProDataSets*

NO-UNDO restriction lifted

As of Release 10.0B, you can track changes on a ProDataSet temp-table defined with or without the NO-UNDO qualifier. Prior to Release 10.0B, you had to use the NO-UNDO qualifier to define a ProDataSet temp-table for which you wanted to track changes.

Keyword attributes now allowed in WHERE clause

Prior to Release 10.0B, attributes whose values are expressed as keywords, such as ROW-MODIFIED, would not compile if used in a WHERE clause. Beginning in Release 10.0B, these keywords will be allowed in WHERE clauses.

New default behavior in SAVE-ROW-CHANGES

In Release 10.0A, the SAVE-ROW-CHANGES method checked whether the underlying database record for a modified ProDataSet buffer had been changed by another user by comparing the before-table row in the ProDataSet with its corresponding database record. If any fields had been changed, then the update from the ProDataSet was rejected, unless the IGNORE-CURRENT-MODIFIED attribute was set for the temp-table.

In Release 10.0B, there are new attributes to enhance this functionality:

- DATA-SOURCE-MODIFIED is a new logical attribute on the temp-table rows, the ProDataSet temp-tables and on the ProDataSet. It is set by the SAVE-ROW-CHANGES method for a temp-table row when another user has changed the database record. If this attribute is set for any row, then SAVE-ROW-CHANGES also sets the DATA-SOURCE-MODIFIED attribute on the corresponding temp-table and the ProDataSet.

- An existing Release 10.0A data-source attribute, IGNORE-CURRENT-MODIFIED, has been renamed PREFER-DATASET (the IGNORE-CURRENT-MODIFIED name remains for backward compatibility with Release 10.0A, but may be deprecated in the future).

  PREFER-DATASET is false by default. When this attribute is false, SAVE-ROW-CHANGES compares the database row fields to their corresponding fields in the ProDataSet temp-table: if there is any conflict, then the ProDataSet change is rejected. When the PREFER-DATASET attribute is true, SAVE-ROW-CHANGES does not compare the database row fields to their corresponding fields in the ProDataSet temp-table: the ProDataSet's changes are written to the database whether or not there is a conflict.

- MERGE-BY-FIELD is a new logical attribute on the data-source object, which is true by default. If MERGE-BY-FIELD is true, then ProDataSet changes are applied field by field, only for modified fields, and there is no conflict unless the same field has been modified both by the ProDataSet update and in the database. If it is false, and if any field in the database record has changed, the ProDataSet changes are either entirely rejected or entirely accepted, depending on the setting of PREFER-DATASET.

  As of Release 10.0B, the default behavior for SAVE-ROW-CHANGES is that if the ProDataSet and another user have changed the same column in the same row, the ProDataSet update is rejected (same as in Release 10.0A). If the ProDataSet and another user have changed different columns, the ProDataSet update is accepted, the columns changed by the ProDataSet update are copied to the database buffer and the database is updated (new in Release 10.0B).

  Clearly, there are several outcomes possible depending on the types of changes and different combinations of the settings of PREFER-DATASET and MERGE-BY-FIELD.

See the new Expert Series book for Release 10.0B, *OpenEdge Development: ProDataSets*, for a complete discussion of update techniques.

BLOB/CLOB changes not always tracked

If a BLOB or a CLOB field is the only field changed in a dataset record when TRACKING-CHANGES = TRUE, no before-image of the record is created in the before-image temp-table, and consequently the changes will not be recognized by the SAVE-ROW-CHANGES, ACCEPT-CHANGES, or REJECT-CHANGES methods. If other record fields are also changed in addition to a BLOB or a CLOB field, then the before-image of the record is created, and the changes are processed correctly.