JAM 7

Master Index

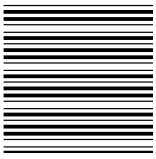
also includes:

Glossary

August 1995

This software manual is documentation for JAM® 7. It is as accurate as possible at this time; however, both this manual and JAM itself are subject to revision.

Send suggestions and comments regarding this document to:
Technical Publications Manager
JYACC, Inc.
116 John Street
New York, NY 10038
(212) 267–7722



Glossary

active screen
The *screen* at the top of the window stack. This is the screen that the user can interact with. If there is a *cursor*, it appears on this screen.

aggregate function
One of a group of *database* functions that calculates summary information on a group of *rows*.

alias
A mechanism whereby a JAM variable is designated to receive information from a database *column*. You should create aliases when the JAM variable and the database column do not have the same name.

alternative scrolling
An alternative method to JAM's memory-resident mechanism for storing offscreen array occurrences. Also called alternative scroll function. The function can be specified in the Alt Scroll Func property under Geometry for any *widget* that has the Scrolling property set to Yes.

application mode
The start-up mode when using the JAM authoring executable. In this mode, you can test a complete application, or portions of an application. You can access the

screen editor, menu bar editor, and styles editor from this mode as well as connect to *databases*.

array

A *field* comprised of one or more *occurrences*. The number of onscreen occurrences is specified in the Array Size property under Geometry. Each occurrence can contain data. The onscreen portion of an array contains one or more *elements*, where each element is identified by a field number that is unique on the *screen*. An array can contain more occurrences than it has onscreen elements; then it is called a *scrolling array* and the number of offscreen occurrences is specified in the # of Occurrences property.

authoring

The act of using the `jamdev` utility (authoring executable) to create a JAM application.

authoring environment

The tools used to create and test application *screen*s and *link*s; the authoring environment includes *application mod*e, the *screen editor*, *menu bar editor*, *styles editor*, and *visual object repositories*.

automatic function

A *hook function* that executes on all occurrences of an event type—for example, screen exit on every *screen* in an application. These functions are never explicitly called in the application code or screens; instead, JAM calls them automatically at the appropriate stage of program execution. Contrast with *demand function*.

bundle

A buffer that stores screen data sent by JPL's `send` command or `sm_append_bundle_data`. JAM can maintain up to ten bundles. Bundle data can be read by the `receive` command or `sm_get_bundle_data`.

check box

A *selection widget* used to enable or disable one or more features or options, usually occurring in a *group* of check boxes. When an option is selected, an X or check mark appears in the box to the left of the widget. Contrast with *radio button*.

child

General term in defining a Detail section or Subdetail section of a screen wizard-generated screen.

child object

Refers to an object that has its Inherit From property set to inherit from a repository object (parent). The child object can be a screen or widget.

class

A widget property that is applied when you run the *transaction manager*. The class setting specifies a *style* for the *widget* in each *transaction mode*.

client

A machine on a network that uses the programs found on the database *server.*

column

The various subsections of a database *table*, each containing a piece of information. `actor_id`, `first_name` and `last_name` are the columns in the `actors` table. In some database systems, a column is called a field.

commit

The act of saving the database changes back to the *server* when a *transaction* is complete. Contrast with *roll back.*

configuration variable

Variables defined either in the `smvars` file or in the environment, which serve as pointers to required configuration files (key translation, video, and message), to alternate setup files, and to application-specific files and information. See also *setup variable*.

connection

A session on a database *server* that must be declared before you can perform SQL statements using JAM's database interface.

control string

Control strings can be thought of as the connectors that hold a JAM application together. A Control String property setting defines the action to take place when a pushbutton, function key, or menu selection *event* occurs. Possible actions include: displaying a *screen* or *window,* calling a developer-written function, or executing a system command.

correlation name

A temporary alternate name for a database *table* which is specified in the SQL statement.

cursor
1) The positional indicator on the monitor. 2) A SQL object associated with a specific query or operation.

database
A collection of information, organized into database *tables*.

database engine
A *DBMS* product which is identified by the vendor name and version number. For example, SYBASE 4.8, ORACLE 6.0 and ORACLE 7.0 are three distinct database engines.

data model
See *schema*.

DBMS
Database Management System. The components used to create and maintain a *database*.

demand function
A *hook function* that is explicitly called from a JAM object, such as a function called from a control string on a *widget* or *screen*, a screen entry function on a specific screen, or a function called from a JPL module. Contrast with *automatic function*.

display text
A *widget* that is not altered by the user or program at *runtime*. Static labels and graphical widgets (boxes and lines) are considered to be display text. These widgets do not get a *field number*.

element
An onscreen *field* in an array. Contrast with *occurrence*.

embedded character
Refers to any character in a data entry *field* that is automatically inserted into the field and cannot be modified by the user. They are created through the Keyboard Filter property. Depending upon the filter used, the library routine `sm_getfield` may or may not see the embedded characters as data.

engine
See *database engine*.

event
An action recognized by a *screen* or *widget* that causes something to happen. Some events are associated with a user interacting with an input device by pressing a function key or push button, or making a menu selection. Others are associated with something happening in the application, such as screen exit. Events often determine the flow of an application.

external menu
A menu that is invoked by a submenu-type item and is not defined in the same menu file as the item. At runtime, JAM searches for the menu among all menu files that are loaded into memory.

field
A *widget* that is typically populated at *runtime*. This includes widgets in which the user enters data, or the program displays variable output. Data entry widgets, option menus and dynamic labels are examples of fields. Each field on a *screen* is uniquely identified by a *field number*, and optionally, a name (specified in the Name property under Identity). It is recommended that you name all the fields on your screens. Contrast with *display text*.

field number
A unique number automatically assigned to a *widget* on a JAM *screen* based on its position on the screen. Only widgets that are considered to be *fields* get a field number. Widgets are numbered from left to right, beginning at the top line of the screen and ending at the bottom line. Thus, the leftmost widget on the top line of the screen is identified as Field #1, and the rightmost widget in the bottom line would have the highest field number on the screen.

Note: *Static labels and graphical widgets (lines and boxes) are not considered fields, and therefore do not get a field number.*

foreign key
A column in a database *table* which is a *primary key* in another table within the same *database*.

form
One of two ways that JAM displays a *screen* in an application. When a form opens, all other screens in the application are closed. Contrast with *window.*

function key
A key with a function other than data entry. Function keys are normally associated with control strings that specify such actions as *form* and *window* display. JAM

function keys are defined as logical keys, and you decide which physical key to map each function to. This feature enables JAM applications to be terminal independent.

group

One or more *selection widget*s (*radio button*, *check box*, *list box*, and *toggle button*) that are functionally connected—therefore, the widgets function as a group. The group has properties that define its entry/exit/validation behavior and tabbing order. Also, the set of widgets that make up a *table view.*

handles

Places on a *widget* or *screen* where the object can be“grabbed” for repositioning or resizing.

help screen

An informational screen that can be attached to a *screen* or *widget* by specifying its name in the JAM Help Screen property. The specified help screen will display at *runtime* when the Help key is pressed or automatically when the screen or widget is entered. JAM also supports an External Help property for displaying help using a third-party help engine.

hook function

A function that is called at specific events during program execution. JAM identifies each stage of program execution as one type of event or another. For example, JAM identifies all screen entries as one event type, and all field exits as another. For more information about hook functions, refer to Chapter 8 in the *Application Development Guide*.

join

A SQL method of combining database *tables* in order to obtain the desired information.

key translation file

A configuration file that contains the mapping between JAM *logical keys* and the physical keys on a terminal. You can create or edit a key translation file with a text editor. You can use the `showkey` utility to aid in determining the sequences that the keys on your terminal generate, for inclusion in the file.

LDB

See *local data block*.

LDB entry
A field in a LDB screen.

link
A *widget* which defines the relationship between two *table views* by designating one table view to be the parent and one table view to be the child. There are two types of links: sequential links and server links. In a sequential link, the processing for the parent table view occurs first, followed by the processing for the child table view. In a server link, the processing for both table views is done at the same time by the database *server.*

list box
A *selection widget* that contains a scrolling list of choices. You may define the number of selections that a user can make.

local data block (LDB)
A JAM *screen* that is used to initialize and save values on other screens. When a screen serves as an LDB, JAM uses its fields, or *LDB entries*, to transfer data to and from corresponding fields on the current screen. By using LDBs, applications can transfer data between screens automatically.

An LDB screen is not displayed. JAM matches LDB entries and screen fields by name. Only named fields and LDB entries take part in LDB processing, or *write-through.*

logical key
A device-independent mnemonic for a predefined JAM function. These logical functions have names such as TRANSMIT and EXIT. Logical keys are mapped to physical keys via the *key translation file.*

lookup screen
A *screen* that serves to validate entries made to certain data entry *widgets*. This screen does not display at runtime. The lookup screen is defined in the Table Lookup property under Input.

message file
A configuration file that maps message text to message identifiers. JAM messages are stored in the message file to enable customization (e.g., for international use). User messages may be included as well. The message file also contains date, time, and numeric format specifications that can be customized.

occurrence
An entry in an *array.* If the number of occurrences is larger than the number of onscreen *elements*, then the array is a *scrolling array.*

parent
General term used in defining a Master section or Detail section of a screen wizard-generated screen.

parent object
Refers to a repository object that sets properties on widgets that inherit from it. The parent object can be a screen or widget.

pop-up menu
A menu that can be accessed by pressing the right (menu) mouse button. The menu that is displayed is context-sensitive—its content depends on the location of the mouse pointer and the menu specification. You can specify the menu in the Pop-up Menu property under Help.

primary key
The information in a database *column* or columns that constitutes a unique entry for each *row* in the database *table*.

prototyped function
A hook function that gets only the number and type of arguments that you specify. Prototyped functions are *demand functions*—that is, they must be invoked by name from a JAM component, such as a *widget* or *screen*. Prototyped functions can be JAM library functions or developer-written. For more information on hook functions, refer to Chapter 8 in the *Application Development Guide*.

push button
Describes a *widget* used to carry out a command or action when selected. A push button is known as a command button in MS Windows parlance.

qbe
Query by example. In this type of database query, the `WHERE` clause includes specific values so that a certain subset of information can be returned.

radio button
A *selection widget* that, when *grouped* with other radio buttons, allows the user to select a single option from a set of choices. A radio button is known as an option button in MS Windows parlance. Contrast with *check box*.

repository
See *visual object repository.*

result set
The set of *rows* that gets returned after executing a database query with SQL `SELECT`.

roll back
The act of undoing the changes made to a database in a database *transaction.* Contrast with *commit.*

root table view
The *table view* that forms the basis of the database processing on a *transaction manager* screen.

row
An entry in a database *table* containing a value for each *column*. In some database systems, a row is called a record.

runtime
The environment in which a user runs a JAM application. The runtime environment is different than the *authoring environment* in that the editors and repositories cannot be accessed.

schema
An outline of your *database* showing the different areas of information (database *tables*) and the different pieces of information in each of those areas (database *columns*) and illustrating how pieces relate to each other. Also called a data model.

screen
Generic term that refers to the objects created using the screen editor. Screens usually contain collections of *widgets.* At runtime, a screen may be opened as a *form* or as a *window.*

scrolling array
An array in which the maximum number of *occurrences* exceeds the number of array *elements*. When the array contains more occurrences than elements, JAM lets you scroll the array to bring hidden occurrences into view. If circular scrolling is enabled, JAM lets you continue scrolling beyond the first and last occurrences to the last and first occurrences, respectively.

select list
In database applications, the list of columns or expressions that are to be processed by the SQL `SELECT` statement.

selection widget
A *widget* that allows a user to select from several of choices. JAM supports four kinds of selection widgets: *check box*, *radio button*, *toggle button*, and *list box*. A set of similar selection widgets are usually *grouped*, so they behave in a certain fashion with respect to the number of choices that can be made.

sequential link
A link where the *transaction manager* processing for the parent *table view* occurs first, followed by the processing for the child table view. See also *link*.

server
A computer on a network that controls access to the database.

server link
A link where *transaction manager* processing for the parent and child *table views* is done at the same time by the database *server.* See also *link*.

server view
A table view and all the table views connected to it by *server links*.

setup file
At runtime, JAM uses setup files to find and translate an application's configuration. The environment variables in setup files tell JAM how the application should look, how the hardware is configured, and where system-specific information is located. Setup files can be used to establish installation-wide and application-specific configuration information.

setup variable
Variables, usually defined in the `smvars` or `smsetup` file, that control the overall behavior of JAM and JAM applications. See also *configuration variable*.

shifting field
A *field* in which the maximum number of characters (Max Data Length property) allowed for input exceeds the number of characters (Length property) that the field can display at any given time. When the field data contains more characters than the field can display, the user can scroll the data horizontally—left and right—to bring hidden data into view.

sibling window
A *window* that is at the same level as another window. Sibling windows can be opened and activated without closing the calling window, and are therefore

non-modal. The VIEWPORT key allows the user to select among (activate) any open sibling windows. Contrast with *stacked window.*

SQL
Structured Query Language. A language used to access *databases* employed by many relational database products. SQL was originally developed by IBM in the 1970's. ANSI standards have been issued for SQL in 1986 and 1992.

stacked window
A *window* that is layered on top of other open *screens*. Once a stacked window is opened, it must be closed before the user can access any underlying screens. This is also called a modal window, and is often used for a dialog box.

style
A group of settings affecting the appearance and focus protection properties of a *widget*. A style is applied to a widget by the *transaction manager* for a particular *transaction mode* based on the widget's *class* property.

submenu
A lower level menu that displays additional choices under a menu item on an upper level menu.

support routine
A module supplied with JAM's database drivers which supplies all the engine-specific instructions needed to process database statements.

synchronized array
Two or more *scrolling arrays* whose data scroll together. Synchronized arrays are defined via the Edit menu in the screen editor. The arrays need not be adjacent to each other on the *screen*, but must contain the same number of *occurrences* and onscreen *elements*.

system catalog
A series of database *tables* containing information about the *database* itself, such as the table names, *column* names and key information.

table
A main subset of information in a database containing a series of *columns* and *rows*. In some database systems, a table is called a file.

table view
A *group* of *widgets*, generally from the same database *table*, which work together in *transaction manager* processing.

test mode
A special state in the various editors that allows you to test the object you are editing. You can toggle into and out of test mode without disrupting your editing session.

toggle button
A *selection widget* that allows users to toggle between two settings—turning some option off or on.

transaction
A database transaction is a series of SQL statements that must either be completed as a unit, or not completed at all. At the end of the series, you *commit* the transaction in order to save the changes to the *database*. If there is an error, you can *roll back* the statements in order to return the database to its original state.

A *transaction manager* transaction is a series of transaction manager commands that are completed together. A transaction manager transaction may contain several database transactions.

transaction manager
A runtime program which performs the processing needed to view and update *database* information. Database statements are generated from settings stored in screen and widget properties.

transaction mode
One of a series of states which indicate the data entry status of a *screen* when using the *transaction manager*. Transaction modes include: initial, new, update and view.

validation
The process of checking user data entry against the keystroke filters you specify in the screen editor. Validation generally occurs when a user tabs out of a *field*. Validation also occurs for all fields on a *screen* at screen exit, or when the XMIT key is pressed.

video file
A configuration file that provides JAM with the information required to use the capabilities of a specific terminal. JAM includes a set of video files for many different terminals. Optionally, you can define a custom video file using the instructions in the *Configuration Guide*.

viewport
The mechanism through which the user views all or part of a *screen*. The VIEWPORT key allows the user to move, resize, and scroll the viewport.

visual object repository

A special JAM library that can consist of one or more *screen*s. Objects in repository entries can export their properties to other objects on your application screens. By copying repository objects to application screens, you create an inheritance relationship between the parent and child object, which can be maintained to ensure consistency throughout your application.

widget

An object on a JAM *screen*. Some widgets are used to interact with an application, while others are for display only.

window

One of two ways that JAM displays a *screen* in an application. Unlike a screen displayed as a *form*, a screen displayed as a window overlays, but preserves, any screens beneath it. The open window is the *active screen* and the image hidden by the open window is saved, to be restored and made active again when the open window is closed. Windows can be *stacked* or *sibling*.

window stack

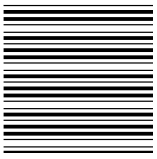
A list that is kept internally by JAM that allows it to remember the order in which *windows* were opened, or were rearranged with the VIEWPORT key. When a window closes, it is popped off the window stack, and the previous window comes to the fore.

word wrap

A property that can be applied to multitext widgets. When characters are inserted, JAM wraps any text which is too lengthy for the current line onto to the next line. When characters are deleted from the *field*, JAM automatically fills the line with words from the next line in the field.

write-through

Intrascreen data transfer that uses *local data block* screens (LDBs).



Index

Symbols

. (dot), in regular expressions: *Editors* 168

! (exclamation point), in control string: *Editors 322*, 327

??? (in properties window): *Editors* 28, 29

; (semicolon), command terminator in JISQL: *Database* 49

:: (parameters), in DECLARE CURSOR command: *App Dev* 250–255

:+ (colon–plus processing): *App Dev* 240
See also Colon preprocessing

:= (colon–equal processing): *App Dev* 246–247
See also Colon preprocessing

[ ] (square brackets)
in regular expressions: *Editors* 168
selection indicator in character JAM: *Editors* 37

{ } (curly braces), selection indicator in character JAM: *Editors* 37

& (ampersand), in control string: *Editors 322*; *App Dev* 110

&& (double ampersand), in control string: *Editors 322*; *App Dev* 110

(pound sign)
comments in ISQL : *Database* 120
comments in JISQL: *Database* 49
in key translation file: *Config* 77
in video file: *Config* 92
with field number: *Editors* 308

# of Selections property. *See* Number of Selections property

% (percent sign)
as floating point formatter: *Editors* 317
as pattern matching operator: *Database* 86
in message file: *Config 57*
parameter sequences: *Config* 94

%A, display attributes in messages: *Editors* 271; *Config* 57

%B, bell for messages: *Editors* 272; *Config* 59

%K, key label in message: *Editors* 271; *Config* 59

%Md, force user acknowledgment of messages: *Config* 59

%Mu, acknowledgment of error messages: *Config* 60

%N, carriage returns in messages: *Config* 60

%t, as floating point formatter: *Editors* 317

%W, pop–up window for messages: *Config* 60

@ (at), to reference database driver variable: *App Dev* 257–259

@date
defining format for: *Config* 66
in Calculation property: *Editors* 318
in JPL: *Lang Ref* 36
international support: *App Dev* 494

@length
in Calculation property: *Editors* 319
in JPL: *Lang Ref* 37

@sum
in Calculation property: *Editors* 318
in JPL: *Lang Ref* 37
international support: *App Dev* 494

@tm_sel_cursor, default select cursor name: *App Dev* 221, 376

+ (plus sign), in Properties window: *Editors* 29

– (hyphen)
in Properties window: *Editors* 29
in regular expressions: *Editors* 169

* (asterisk)
in regular expressions: *Editors* 168, 170
indicator in color palette: *Editors* 256

^ (caret)
in control string: *Editors 322*; *App Dev* 112
in regular expressions: *Editors* 169

\ (backslash)
in messages: *Config* 44
in regular expressions: *Editors* 168

_ (underscore), as pattern matching operator: *Database* 86

Numbers

000 Separator property. *See* Thousands Separator property

3D, Windows initialization option: *Config* 156

3D property
for graph widget: *Editors* 135–136
bar/line graph: *Editors* 152–153
high/low chart: *Editors* 159–160
pie chart: *Editors* 145
XY plot: *Editors* 155–156
for screens: *Editors* 249–251
background color: *Editors* 250

A

AC_KEEPATTRS: *Config* 31

AC_SETATTRS: *Config* 31

AC_SWATTRS: *Config* 31

Accel property, for menu item: *Editors* 218

Accelerator, assigning to menu item: *Editors* 218

Action menu item: *Editors* 221

Active Pixmap property
for menu items: *Editors* 224
for push button widget: *Editors* 193
for toolbar items: *Editors* 219
for widget: *Editors* 266

Active property: *Editors* 302
for menu item: *Editors* 220
for push button widget: *Editors* 196

Addition operation, in JDB: *Database* 90

ADDM key (add mode), hex value: *Config 80*

Aggregate functions
aliasing to widgets: *App Dev* 227–228
in automated SQL generation: *App Dev* 284–285
in JDB: *Database* 59–61
with GROUP BY clause: *Database* 75

ALIAS, dbms command, aliasing column names: *Database* 135–137; *App Dev* 225–226

Aliasing
colors: *Config* 138–142
column names to widgets: *Database* 135–137; *App Dev* 225–228

Align command: *Editors* 46

Aligning widgets: *Editors* 46–48
on grid coordinate: *Editors* 47
with Snap to Grid: *Editors* 47

ALL keyword, in JDB: *Database* 101

Alphabetic data: *Editors* 164
range checking: *App Dev* 493
specifying range: *Editors* 171

Alphanumeric data: *Editors* 164

ALT keys, hex value: *Config 81*

Alt Tab property: *Editors* 307

Alternate character sets: *Config* 125–128

Alternative scroll driver: *Editors* 316

Alternative scrolling. *See* Scrolling array, alternative scroll driver

ANSI terminal
  latch attributes: *Config* 120
  sample video file: *Config* 133
  setting color: *Config* 119

ANY keyword, in JDB: *Database* 101

app–defaults directory: *Config* 170

APP1–APP63 (application function keys), hex value: *Config 82*

Application
  aborting: *Lang Ref* 182, 296
  base form: *App Dev* 69
  code. *See* Hook functions
  development, overview: *App Dev* 3–31
  escaping to operating system: *Lang Ref* 342
  exiting base form: *App Dev* 70; *Config* 36
  handle to instance, getting: *Lang Ref* 395
  initialization: *Lang Ref* 282; *Config* 8
    key translation file: *App Dev* 474
  localization: *App Dev* 482–494
  memory. *See* Memory
  menu, attaching: *App Dev* 89
  properties. *See* Application runtime properties
  referencing in JPL: *Lang Ref* 25
  resetting display to system defaults: *Lang Ref* 426
  returning after escape: *Lang Ref* 430
  size: *App Dev* 525
  start: *Lang Ref* 308

Application behavior
  changing default, in Windows: *Config* 156
  changing the default: *Config* 20
  defining at design time: *Editors* 321
  options in Motif: *Config* 174
  variables for controlling: *Config* 21–38

Application components, referencing: *Lang Ref* 24
  with object modifiers: *Lang Ref* 25

Application data: *App Dev* 482–483

Application function keys (APP1–APP63), hex value: *Config 82*

Application messages: *Config* 45
  adding: *Config* 48–49
  header file: *Config* 53

Application mode: *Editors* 4–6; *App Dev* 6
  menu bar: *Editors* 4

Application runtime properties: *Lang Ref* 520
  getting: *Lang Ref* 406, 413
    cursor offset in field: *Lang Ref* 444
    decimal symbol: *Lang Ref* 406
    no value: *Lang Ref* 406
    repository name: *Lang Ref* 406
    screen name: *Lang Ref* 407
    status line attributes: *Lang Ref* 407
    terminal identifier: *Lang Ref* 406
    video settings: *Lang Ref* 407
    yes value: *Lang Ref* 406
  getting handle to: *Lang Ref* 416
  setting: *Lang Ref* 418, 421
    decimal symbol: *Lang Ref* 421
    no value: *Lang Ref* 421
    screen name: *Lang Ref* 421
    status line attributes: *Lang Ref* 421
    through global variables: *Lang Ref* 297
    video settings: *Lang Ref* 421
    yes value: *Lang Ref* 421

Area attributes
  assigning: *Config* 116–117
  defined: *Config* 115
  removing: *Config* 117

Area graphics, setting: *Config* 118

AREAATT keyword (video file): *Config* 116–117

ARGR keyword (video file): *Config* 117

Argument passing in JPL: *Lang Ref* 14

Arithmetic commands: *Config 96*, 98

Arithmetic operators, in JDB: *Database* 90

Armed Pixmap property
  for menu items: *Editors* 224
  for push button widget: *Editors* 193
  for toolbar items: *Editors* 219
  for widget: *Editors* 267

Arranging widgets: *Editors* 46–49

Array
  about: *App Dev* 79
  behavior of: *Editors* 313–316
  clearing all data: *Lang Ref* 186; *App Dev* 84
  copying data: *Lang Ref* 189
  creating: *Editors* 242
  declaring in JPL: *Lang Ref* 76
  deleting occurrence: *Lang Ref* 219; *App Dev* 85

Array (continued)
editing contents with external editor: *Lang Ref* 235
elements: *App Dev* 78
getting
current occurrence number: *Lang Ref* 402
runtime properties. *See* Array runtime properties
horizontal : *Editors* 242
inserting occurrence: *Lang Ref* 291; *App Dev* 85
occurrence. *See* Occurrence
offscreen occurrence specification: *Editors* 243
reading file contents into: *Lang Ref* 239
required data and: *Editors* 172
scrolling: *Editors* 242
See also Scrolling array
setting runtime properties. *See* Array runtime properties
spacing between occurrences: *Editors* 242
sum of occurrences: *Lang Ref* 37
synchronizing: *Editors* 312–316
See also Synchronized arrays
tab order specification: *Editors* 309–311
trimming: *Lang Ref* 446
value source for graph widget: *Editors* 138
writing contents to file: *Lang Ref* 233

Array data, accessing in JPL: *Lang Ref* 27

Array runtime properties
accessing in JPL: *Lang Ref* 30
getting: *Lang Ref* 413–415
for element: *Lang Ref* 414
for occurrence: *Lang Ref* 414
getting handle to elements: *Lang Ref* 417
getting handle to occurrences: *Lang Ref* 417
setting: *Lang Ref* 418–420
for element: *Lang Ref* 419
for occurrence: *Lang Ref* 419

Array Size property: *Editors* 242, 312
for dynamic label widgets: *Editors* 119

Arrow keys
hex value: *Config 79*
setting horizontal movement: *Config* 22
setting vertical movement: *Config* 23
wrapping behavior: *Config* 24

ARROWS keyword (video file): *Config* 130

ASC keyword
in ORDER BY clause: *Database* 94
in Sort Widgets property: *App Dev* 288

ASCII
extended control codes: *Config* 93
non–ASCII display: *App Dev* 482
table of mnemonics and hex values: *Config 83*

ASCII output
menus: *App Dev* 98
screens: *App Dev* 565

ASGR keyword (video file): *Config 107*, 118
parameters: *Config* 118

Authoring environment: *Editors* 3–7

Auto Help property: *Editors* 275

Auto Horiz Resize property: *Editors* 43

Auto Item property: *Editors* 276

Auto Release on Screen Close: *Editors* 16

Auto Vert Resize property: *Editors* 43

Auto–wrap, setting margin: *Config* 113

AUTOCOMMIT, dbms command, committing transactions: *Database* 273, 306

Automatic hook functions
defined: *App Dev* 116
example: *App Dev* 159–163, 164–166
installing
field function: *App Dev* 128
group function: *App Dev* 133
screen function: *App Dev* 124

Autonumbering
grid columns: *Editors* 185
grid rows: *Editors* 188

Autotab: *Editors* 311

AVAIL_FUNC. *See* Record function

AVG function, in JDB: *Database* 59

Axis, graph widget: *Editors* 130–135
labelling: *Editors* 131–132
positioning: *Editors* 130–131
tick marks: *Editors* 132–135

B

BACK key (backtab), hex value: *Config 79*

Background color: *Editors 256*
resource in Motif: *Config* 172

Background status, displaying: *App Dev* 203

Backslash
in messages: *Config* 44
inputting: *Config* 92

Backspace, video file entry: *Config* 113

Backtab: *Lang Ref* 169

Backward scrolling, viewing database rows: *Database* 153–156, 174–176, 213–214, 243, 266, 294–295, 331, 365; *App Dev* 232–233

Bar Type property, for graph widget, bar/line graph: *Editors* 151–152

Bar/line graph: *Editors* 150–154
bar type: *Editors* 151–152
creating: *Editors* 150–151
data series style: *Editors* 139–140
data source: *Editors* 154
displaying in 3D: *Editors* 152–153
legend: *Editors* 126, 127, 142–143

Base form: *App Dev* 69
exiting: *App Dev* 70; *Config* 36

Base window: *Config* 174
getting Widget ID: *Lang Ref* 516

Basic colors
defined: *Editors* 253–258
defining in Motif: *Config* 171
defining in Windows: *Config* 155–156
keywords: *Config* 140
listed: *Editors* 256

Before image processing: *App Dev* 366–367
comparing values: *Lang Ref* 171–172
copying current values: *Lang Ref* 173
in automated SQL generation: *App Dev* 297, 300
initializing: *Lang Ref* 174–175
modifying data in transaction manager: *App Dev* 323

BEGIN, dbms command, starting database transaction: *Database* 226, 339, 388

Behavior, variables for controlling: *Config* 21

Bell
in status line text: *Editors* 272
invoking: *Lang Ref* 170
setting in messages: *Lang Ref* 60, 224; *Config* 59

BELL keyword (video file): *Config* 131

BETWEEN predicate, in JDB: *Database* 62–63, 105

bin2c: *App Dev* 524, 563–564

bin2hex: *App Dev* 564–565

BINARY, dbms command, fetching binary column values: *Database* 138–139

Binary columns
reading from database: *App Dev* 235
writing to database: *App Dev* 249

Binary variables
deleting occurrence: *Lang Ref* 111
getting
maximum number of occurrences: *Lang Ref* 115
occurrence data: *Lang Ref* 113
occurrence data length: *Lang Ref* 112
occurrence length: *Lang Ref* 114
pointer to occurrence: *Lang Ref* 110
setting, occurrence data length: *Lang Ref* 116

Binding, supplying database column values: *App Dev* 219, 250–255

binherit: *App Dev* 66–68
arguments and options: *App Dev* 67
error messages: *App Dev* 68

BIOS flag (video file): *Config 110*

Bit–mapped attributes: *Config* 122

Bitmap
See also Pixmap
custom mouse cursor shapes: *Editors* 110
mapping string ID to integer ID: *Lang Ref* 409

Bitwise expression: *Lang Ref* 39

Bitwise operators: *Lang Ref* 37

BKSP key (backspace), hex value: *Config 79*

Blank numeric field: *Editors* 248

Blink display attribute, setting: *Editors* 258

BMP files: *Editors* 266

BOFD key (beginning of field), hex value: *Config 80*

Bold property: *Editors* 238

BOLN key (beginning of line), hex value: *Config 80*

Border
keywords: *Config 108*, 128–130
limiting attributes: *Config* 130
property. *See* Border property; Border Style property
setup variables, for menu bars: *Config* 32
styles, specifying alternate: *Config* 128–130
zoom window: *Config* 30

BORDER keyword (video file): *Config* 128–130

Border property
character mode screens: *Editors* 111
for list box widget: *Editors* 201
for screens: *Editors* 111
for widgets: *Editors* 265

Border Style property, for screens: *Editors* 111

Border Width property, for graph widget legend: *Editors* 127

BOTTOMRT keyword (video file): *Config* 109

BOX keyword (video file): *Config* 130

Box widget: *Editors* 26, 259–263
3D (in Windows): *Editors* 251
default style: *Editors* 260
including title: *Editors* 261
specifying alias style: *Editors* 260
specifying margin: *Editors* 264
specifying style: *Editors* 261

BRDATT keyword (video file): *Config* 130

BROWSE, dbms command, returning one row at a time: *Database* 390

Buffer, setting size of output: *Config* 109

BUFFER_DEFAULT, dbms command, setting type of scrolling: *Database* 227, 392

BUFSIZ keyword (video file): *Config* 109

Built–in control functions: *Editors 326*; *Lang Ref* 81–87

Bundle. *See* Send data

Button, types of: *Editors* 25–35

Button/Menu Status property: *Editors* 288

C

C function, executing from control string: *Editors* 325–327; *App Dev* 112–113

C Type property: *Editors* 300–301
See also JAM type
formatting fetched data: *App Dev* 235–237
from database column type: *Database* 210–211, 241–242, 263–264, 291–292, 326–327, 361–362
setting for version columns: *App Dev* 370–371
writing values to database
character strings: *App Dev* 249
hexadecimal strings: *App Dev* 249
numeric data: *App Dev* 245

CA, case interface message tag prefix: *Config* 44

Calculation
in fields: *Editors* 316
using a date: *Editors* 318

Calculation property: *Editors* 316–319
See also Math expression

Calling JPL procedure: *Lang Ref* 14
as hook function: *Lang Ref* 15
from control string: *Lang Ref* 16
through call command: *Lang Ref* 17
within expression: *Lang Ref* 17

CANCEL, dbms command, stopping a stored procedure: *Database* 340, 393

Cancel check out: *Editors* 365, 367

Cancel push button, creating: *Editors* 194

Caret function. *See* Control function

Carriage return
in message. *See* %N
video file entry: *Config* 112

Case sensitivity
alias names: *App Dev* 226
column names: *App Dev* 209, 210
connection names: *App Dev* 213
cursor names: *App Dev* 217
engine names: *App Dev* 209, 210
filenames: *Config* 33
transaction manager commands: *App Dev* 397
widget names: *App Dev* 225

CATQUERY, dbms command, writing results to widget or file: *Database* 140–142; *App Dev* 237

CBDSEL keyword (video file): *Config* 131

CBSEL keyword (video file): *Config* 131

Centering text, on box widgets: *Editors* 235

Centering widgets: *Editors* 49

Century specification: *Config* 36

CHANGE, transaction manager command, switching transactions: *App Dev* 403

char (data type), in JDB: *Database* 69

CHAR_VAL_OPT: *Config* 36

Character classes
  in regular expressions: *Editors* 168–169
  special characters in: *Editors 169*

Character data, 8–bit: *App Dev* 482–483

Character JAM
  multiple select mode: *Editors* 15
  selecting multiple widgets: *Editors* 38
  setting line and box style in cmap file: *Config* 145

Character set
  8–bit translation: *Config* 85, 125
  graphics: *Config* 125–128

Character strings
  reading from database: *App Dev* 234
  writing to database: *App Dev* 244, 249
    in Oracle: *Database* 293

Character–level regular expression: *Editors* 166

Character–sequence, defined: *Config* 77

Characters, as unit of measurement: *Editors 41*

Chart Type property, for graph widget: *Editors* 122

Check box widget: *Editors* 26, 199–206
  See also Group
  3D (in Windows): *Editors* 250
  appearance of: *Editors* 200
  displaying image on: *Editors* 266

Check digit function: *App Dev* 139–140
  executing: *Lang Ref* 183
  return codes: *App Dev* 139
  standard arguments: *App Dev* 139

Check Digit property: *Editors* 319

Check In: *Editors* 365

Check Out: *Editors* 365
  cancel: *Editors* 365

Child List window: *Editors* 66

Child property: *Editors* 345
  determining child table view: *App Dev* 314, 358–359

Child widget
  finding: *Editors* 65
  finding parent of: *Editors* 65
  setting the source of inheritance: *Editors* 65
  turning inheritance on/off for specific properties: *Editors* 64

Circular property
  for array: *Editors* 243
  for grid frames: *Editors* 187

CKDIGIT_FUNC. *See* Check digit function

Class name (Motif)
  application: *Config* 170, 178
  field widgets: *Config* 180
  menu widgets: *Config* 183
  screen widgets: *Config* 179
  widget: *Config* 178–184

Class property
  for menu items: *Editors* 221, 287; *App Dev* 335–336
  for push buttons: *App Dev* 335–336
  for widgets: *Editors* 284, 285

Classes. *See* Transaction classes

CLEAR, transaction manager command, clearing data in widgets: *App Dev* 404–405

Clear on input. *See* Select on Entry property

Clearing Protect property: *Editors* 304–305
  for scale widgets: *Editors* 176
  in screen editor: *Editors* 174
  in styles editor: *Editors* 290

CLICK_TIME: *Config* 24

Clock Type property: *Editors* 245

CLOSE, transaction manager command, closing database transaction: *App Dev* 406–408

CLOSE CONNECTION, dbms command, closing database connections: *Database* 144; *App Dev* 215

CLOSE CURSOR, dbms command, closing database cursor: *Database* 145; *App Dev* 222

Close Item property: *Editors* 113

CLOSE TRANSACTION, dbms command, closing a declared transaction: *Database* 396

CLOSE_ALL_CONNECTIONS, dbms command, closing database connections: *Database* 143; *App Dev* 215

CLOSE_ALL_TRANSACTIONS, dbms command, closing all declared transactions: *Database* 394

CLR key (clear all)
  clock update and: *Editors* 245
  hex value: *Config 79*

cmap. *See* Configuration map file

CMFLGS keyword (video file): *Config* 112

CMSG keyword (video file): *Config* 124

COF keyword (video file): *Config* 114

COLMS keyword (video file): *Config* 109

Colon preprocessing: *Lang Ref* 20
  colon equal: *App Dev* 246
  colon plus: *App Dev* 240
  examples: *App Dev* 247–249
  simulating from C: *Lang Ref* 124–125
  substring specifier: *Lang Ref* 22
  writing to a database: *Database* 211–212, 242, 265, 293, 328–329, 363–364; *App Dev* 239–249

COLOR keyword (video file): *Config* 118–119

Color Name property: *Editors* 255

Color palette
  defining colors in Motif: *Config* 171
  defining colors in Windows: *Config* 155
  using: *Editors* 256–257

Color properties
  See also Basic colors; Extended colors; Scheme
  3D effect on: *Editors* 250
  aliasing colors: *Config* 138–142
  basic colors. *See* Basic colors
  changing: *Editors* 255–257
  container color, specifying on Color palette: *Editors* 256
  display attributes
    keywords: *Config* 140
    setting: *Editors* 257–258
  for screens: *Editors* 33
  for transaction styles: *Editors* 289

Color properties (continued)
  for widgets: *Editors* 31
  highlighted colors, in Windows: *Config* 139
  JAM basic colors, keywords: *Config* 140
  Motif resources for overriding: *Config* 172
  scheme. *See* Scheme
  types defined: *Editors* 253–254

Color property, for graph widget data series: *Editors* 142

Color terminal, display attributes in messages: *Config* 58

Column. *See* Grid column; Database columns

Column Edits subproperties: *Editors* 353

Column Move Resize property: *Editors* 189

Column Name property: *Editors* 353
  in automated SQL generation: *App Dev* 280, 293, 297

Column Separators property: *Editors* 186

Column Title property, set via the screen wizard: *Editors* 83

Column Titles property: *Editors* 185

COLUMN_NAMES, dbms command, mapping column names only: *Database* 146–147

Columns property, for table view: *Editors* 352

Combo box widget: *Editors* 24, 176–180
  3D (in Windows): *Editors* 250
  and autotab behavior: *Editors* 312
  assigning double–click event to: *Editors* 305
  controlling size of: *Editors* 178
  populating: *Editors* 177
  scrolling: *Editors* 178
  specifying initial text: *Editors* 180
  updating contents: *Lang Ref* 495

Command line, Motif: *Config* 177–188
  bg switch: *Config* 178
  fg switch: *Config* 178
  name switch: *Config* 170
  ownColormap switch: *Config* 178

Comments
  in JPL: *Lang Ref* 7
  in key translation files: *Config* 77
  in message file: *Config* 45

Comments property: *Editors* 115
  assigning in screen wizard: *Editors* 79

COMMIT, dbms command
committing database transaction: *App Dev* 266–267
committing transactions: *Database* 228, 249, 275, 308, 341, 397

commit
transaction in ISQL: *Database* 120
transaction in JDB: *Database* 115
transaction in JISQL: *Database* 50

Comparison operators, in JDB: *Database* 91

Compose characters: *Config* 85

Compose key: *Config* 85, 126

Compress
library: *App Dev* 561
repository: *Editors* 57

COMPRESS keyword (video file): *Config* 132

CON keyword (video file): *Config* 114

Configuration
converting message files: *Config* 49–52
converting terminfo/termcap to video file: *Config* 101–102
converting video files: *Config* 104–105
JDB: *Database* 32
memory–resident files: *App Dev* 523

Configuration directory, contents: *Config* 3

Configuration files, required by JAM: *Config* 2

Configuration map file
aliasing colors: *Config* 138–142
colors section: *Config* 138–142
object specification keywords: *Config* 143
scheme section: *Config* 142–145
screen editor section: *Config* 140–142

Configuration variables
application–specific: *Config* 16–18
defined: *Config* 6
for setting path: *Config* 18
types: *Config* 13–14

CONNECTION, dbms command, setting database connection: *Database* 148; *App Dev* 214

Connections. *See* Database connections

Constant data: *Editors* 177

Constants in JPL: *Lang Ref* 23

Container color: *Editors* 253
defined: *Editors* 256–258

Continuation character: *Lang Ref* 6

Continuation file
scrolling through select set: *App Dev* 232
specifying: *Database* 174–176
in the transaction manager: *App Dev* 322, 377
specifying availability of: *Editors* 339
using in transaction manager: *Lang Ref* 465

CONTINUE
availability in transaction manager: *Lang Ref* 465
dbms command, fetching next set of rows: *Database* 149–150; *App Dev* 231–232
transaction manager command, fetching next set of data: *App Dev* 409–411

Continue buttons
defined in screen wizard: *Editors* 81, 87
modifying: *Editors* 89

Continue commands: *Editors* 339

CONTINUE_BOTTOM
dbms command, fetching last set of rows: *Database* 151; *App Dev* 232
transaction manager command, fetching last set of rows: *App Dev* 412–415

CONTINUE_DOWN
dbms command, fetching next set of rows: *Database* 152
transaction manager command, fetching next set of rows: *App Dev* 416–419

CONTINUE_TOP
dbms command, fetching first set of rows: *Database* 153; *App Dev* 232
transaction manager command, fetching first set of rows: *App Dev* 420–423

CONTINUE_UP
dbms command, fetching previous set of rows: *Database* 154–155; *App Dev* 232
transaction manager command, fetching previous set of rows: *App Dev* 424–427

Control characters, entering: *Config* 93

Control flow, commands: *Config 96*, 99

Control flow in JPL: *Lang Ref* 6

Control function: *App Dev* 142–143
example: *App Dev* 179
return codes: *App Dev* 142
standard argument: *App Dev* 142

Control panel (Windows), defining color scheme: *Editors* 254

Control string: *App Dev* 109–114
  assigning to menu item: *Editors* 218
  binding to function key: *Config* 25
  calling JPL: *Lang Ref* 16; *App Dev* 112–113
  case sensitivity for filename searches: *Config* 33
  debugger view of assignments: *App Dev* 511
  executing from list box: *Editors* 323
  executing from menu item: *Editors* 221
  executing from push button: *Editors* 323–327
  executing from screen–level: *Editors* 323
  executing function from: *Editors* 325–327; *App Dev* 112–113
  executing OS command from: *Editors* 327; *App Dev* 113–114
  including screen size/dimension: *Editors* 324
  nesting: *Editors* 325
  syntax: *Editors 322*
  target string in: *App Dev* 112

Control String property: *Editors* 323
  for list box widget: *Editors* 203
  for push button widget: *Editors* 196–197
  syntax: *Editors 196*

CONTROL_FUNC. *See* Control function

Controlling input, variables for: *Config* 21–38

Controls, specifying in screen wizard: *Editors* 79

Conversion utilities
  bin2c: *App Dev* 563–564
  bin2hex: *App Dev* 564–565
  f2asc: *App Dev* 565–567
  jpl2bin: *Lang Ref* 11
  key translation files to binary (key2bin): *Config* 86–87
  m2asc: *App Dev* 97
  message files to binary (msg2bin): *Config* 49–52
  setup files to binary (var2bin): *Config* 10–11
  styles file to/from ASCII, s2asc: *Editors* 292
  video file to binary (vid2bin): *Config* 104–105

Convert Case property: *Editors* 171

COPY, transaction manager command, copying data for edit: *App Dev* 428–429

COPY_FOR_UPDATE, transaction manager command, changing to update mode: *App Dev* 430–431

COPY_FOR_VIEW, transaction manager command, changing to view mode: *App Dev* 432–433

Copying widgets
  at runtime: *Lang Ref* 399
  in screen editor: *Editors* 44–45

Correlation names
  for database tables: *Database* 23
  for self-joins: *Database* 84

COUNT function, in JDB: *Database* 59

CREATE DATABASE statement, in JDB: *Database* 64–65

Create menu: *Editors* 22

CREATE TABLE statement, in JDB: *Database* 66–68

Creating. *See* specific object type

Ctrl property, for menu items: *Editors* 218

CUB keyword (video file): *Config* 113
  parameters: *Config 94*

CUD keyword (video file): *Config* 113
  parameters: *Config 94*

CUF keyword (video file): *Config* 113
  parameters: *Config 94*

CUP keyword (video file): *Config* 114
  parameters: *Config 94*

CURPOS keyword (video file): *Config* 132

Currency format: *Editors* 173, 248; *Config* 66–70
  default entries in message file: *App Dev 487*; *Config 68*
  fetching from database: *App Dev* 235
  internationalization: *App Dev* 486–488
  stripping from string: *Lang Ref* 453
  writing to database
    colon–plus processing: *App Dev* 245
    in SYBASE: *Database* 328, 363

Currency Symbol property: *Editors* 248

Cursor
  See also Cursor (database); Mouse pointer
  appearance
    keywords: *Config 107*, 114–115
    restoring: *Config* 115
    saving: *Config* 115
    setting: *Config* 22, 114
    switching to/from system style: *Config* 114
  backtabbing to previous field: *Lang Ref* 169
  behavior in groups: *Config* 34

Cursor (continued)
changing delay state: *Lang Ref* 212; *App Dev* 559
controlling behavior in group: *Lang Ref* 318
defining gray/white keys: *Config 110*
getting location in field: *Lang Ref* 444
getting offset in field: *Lang Ref* 216
movement
defining: *Config* 22–24
setting video display: *Config* 113
movement in arrays: *Editors* 313
moving to
field: *Lang Ref* 275, 403
first field: *Lang Ref* 280
last field: *Lang Ref* 323
next field: *Lang Ref* 458
next line: *Lang Ref* 397
position
after check digit function: *App Dev* 139
after field validation: *App Dev* 128
after group validation: *App Dev* 133
displaying: *Config* 132
keywords: *Config 106*, 112–114
restoring: *Config* 115
saving: *Config* 115
setting absolute: *Config* 114
setting absolute position (CUP): *Config* 114
specifying style of: *Config* 110
toggling position display: *Lang Ref* 180
turning off: *Lang Ref* 178
turning on: *Lang Ref* 179
turning on/off. *See* CON/COF keyword

Cursor (database): *App Dev* 217–222
checking status: *Lang Ref* 152
closing: *Database* 145; *App Dev* 214, 222
declaring: *Database* 157–158, 212–213, 242–243, 265–266, 294, 329–331, 364; *App Dev* 219–222, 250–252
executing statement: *Database* 160
redeclaring: *App Dev* 222
in transaction manager: *App Dev* 221
specifying cursor for dbms command: *Database* 180–181
transaction manager usage: *App Dev* 375–376
transaction model usage strategies: *App Dev* 365
using bind values: *App Dev* 219–221, 250–253
using colon expansion: *App Dev* 219
using the default: *App Dev* 218

Cursor attributes. *See* Cursor, appearance

Cursor backward. *See* CUB keyword

Cursor down. *See* CUD keyword

Cursor forward. *See* CUF keyword

Cursor up. *See* CUU keyword

Custom spacing: *Editors* 48

Customer Drawn property: *Editors* 268

CUU keyword (video file): *Config* 113
parameters: *Config 94*

D

DA_CENTBREAK: *Config* 36

DARR key (down arrow), hex value: *Config 79*

Data
See also Application data; Field data
clearing, in the transaction manager: *App Dev* 404–405
clearing field of: *Editors* 173
copying, in the transaction manager: *App Dev* 428–429
deleting, in the transaction manager: *App Dev* 379–380
deleting from database, in JDB: *Database* 72
entering into database, in JDB: *Database* 79–80
formatting for database updates: *Database* 211–212, 242, 265, 293, 328–329, 363–364
inserting, in the transaction manager: *App Dev* 440–442
matching specified pattern: *Database* 86
modifying: *Database* 103–104
in the transaction manager: *App Dev* 323–324, 444–449
protecting: *Editors* 173–174
retrieving from multiple tables: *Database* 81–85
scrolling through result set: *Database* 151–157
selecting: *Database* 96–99
in the transaction manager: *App Dev* 321–322, 377–379, 450–454, 458–462
using a database driver: *App Dev* 223–237
specifying groups in database: *Database* 75–76
specifying order from database: *Database* 94–95
writing to a database: *App Dev* 239–254

Data compression, specifying for Jterm: *Config* 132

Data entry
required: *Editors* 172
widgets: *Editors* 24, 161–180
with input widget: *Editors* 24

Data filter: *Editors* 162–171

Data Formatting property
  for date/time specification: *Editors* 244–246
  for numeric specification: *Editors* 247
  formatting fetched data: *App Dev* 235
  using in database updates: *App Dev* 243

Data series, graph widget: *Editors* 136–143

Data type
  in JDB: *Database* 69–71
  specifying: *Editors* 300–301
  JPL: *Lang Ref* 32

Database columns
  aliasing to widgets: *Database* 135–137; *App Dev* 225–228
  automatic mapping to widgets: *App Dev* 225
  defined: *Database* 10–11
  defining in JDB: *Database* 66–68
  fetching binary values: *Lang Ref* 110; *Database* 138–139
  getting serial column value: *Database* 198
  importing to a repository: *App Dev* 63–64
  in automated SQL generation: *App Dev* 274, 280–281, 293, 297
  JDB, defining in JISQL: *Database* 38–39
  mapping column names into JAM variables: *Database* 146–147
  mapping result set to widget/file: *Database* 140–142
  naming conventions, JDB: *Database* 28
  selecting: *Database* 96–99
  setting number of: *Config* 109
  suppress repeating values: *Database* 177

Database connections
  checking status: *Lang Ref* 151
  closing: *Database* 143, 144; *App Dev* 215
  declaring: *Database* 156; *App Dev* 213–215, 320
  Informix: *Database* 208
  JDB: *Database* 240
  ODBC: *Database* 260
  Oracle: *Database* 289
    for XA library: *Database* 289
  setting current: *Database* 178–179; *App Dev* 214
  setting default: *Database* 148; *App Dev* 214
  SYBASE: *Database* 324, 359
  using more than one: *App Dev* 214

Database drivers
  commands: *Database* 131–182
  Informix: *Database* 205–236
  initializing: *Database* 206, 238, 257, 286, 320, 356; *App Dev* 207–211
    in Windows: *App Dev* 210–211
  JDB: *Database* 237–253
  keywords: *Database* 199–203
  listing of error messages: *Database* 192–194
  ODBC: *Database* 255–283
  Oracle: *Database* 285–317
  selecting data: *App Dev* 223–237
  Sybase-CT Library: *Database* 319–353
  Sybase-DB Library: *Database* 355–421
  writing to a database: *App Dev* 239–254

Database engines
  accessing: *App Dev* 213–215
  adding support for an engine: *App Dev* 211
  checking status: *Lang Ref* 153
  deinstalling: *Lang Ref* 154
  initializing: *Lang Ref* 149–150; *Database* 206–207, 238–239, 257–258, 286–287, 320–322, 356–357; *App Dev* 207–211
    in Windows: *App Dev* 210
  optimistic locking: *App Dev* 370–371
  setting current: *Database* 182; *App Dev* 211, 214
  setting default: *Database* 159; *App Dev* 211, 214
  using more than one: *Database* 182; *App Dev* 214–215
  viewing error messages: *App Dev* 257–259

Database interface
  execute dbms command in JPL: *Lang Ref* 49
  initializing: *Lang Ref* 117

Database menu, connecting to the database: *App Dev* 318

Database message tag: *Config* 44

Database properties
  for SQL generator: *Editors* 352
  for table view: *Editors* 340
  for widgets: *Editors* 32

Database tables. *See* Tables

Databases
  See also Database connections; Database drivers; Database engines
  connecting to: *Editors* 59
  designing: *Database* 14–16
  importing database to a repository: *App Dev* 63–64
  JDB
    connecting using ISQL: *Database* 118

Databases (continued)
connecting using JISQL: *Database* 34–35, 50
creating databases: *Database* 64–65, 118
creating databases in JISQL: *Database* 36
deleting: *Database* 73
describing using JISQL: *Database* 46
disconnecting using JISQL: *Database* 35
dropping using JISQL: *Database* 47
naming conventions, JDB: *Database* 27
optimistic locking: *App Dev* 370–371
re-creating JDB database: *Database* 122
reading information from: *App Dev* 223–237
relational: *Database* 9
transaction processing: *App Dev* 265–269
writing information to: *App Dev* 239–254

Date, using in JPL expressions: *Lang Ref* 36

Date/time format: *Editors* 244–246
applying to supplied value: *Lang Ref* 492
applying to system date/time: *Lang Ref* 436
customizing: *Config* 60–66
defaults: *Editors* 244; *Config* 61–62
defining custom format: *Editors* 245–249
examples of custom formats: *Editors 246*
fetching from database: *App Dev* 234
for non-English JAM: *Config* 65
for non–English applications: *Config* 64
internationalization: *App Dev* 483–486
literal format for @date calculations: *Config* 66
system update: *Editors* 245
tokens: *Config* 62–63
variables for custom format: *Editors 246*
writing to database: *App Dev* 248
colon–plus processing: *App Dev* 244
in Informix: *Database* 211
in Oracle: *Database* 293
in SYBASE: *Database* 328, 363

datetime (data type), in JDB: *Database* 70

DB Interactions: *Editors* 14; *App Dev* 383
viewing link types: *App Dev* 315
viewing transaction tree: *App Dev* 315–316, 359

dbiinit.c
creating new: *App Dev* 211–212
initializing database engines: *App Dev* 207–211

DBMS commands: *Database* 131–182
executing from C: *Lang Ref* 118, 120, 265
for Informix: *Database* 234–236
for JDB: *Database* 251–253
for ODBC: *Database* 278–280
for Oracle: *Database* 315–317
for SYBASE: *Database* 351–353, 418–420
last executed command: *Lang Ref* 148
summary: *Database* 131–134

DDE: *App Dev* 545–552
callback function: *App Dev* 551
installing: *Lang Ref* 207
cold links
creating for JAM client: *Lang Ref* 196; *App Dev* 549
updated from JAM server: *App Dev* 547
cold paste links, creating for JAM client: *Lang Ref* 202; *App Dev* 549
destroying links on JAM client: *Lang Ref* 199; *App Dev* 551
disabling JAM as client: *Lang Ref* 200
disabling JAM as server: *Lang Ref* 210; *App Dev* 548
enabling JAM as client: *Lang Ref* 201; *App Dev* 548
enabling JAM as server: *Lang Ref* 211; *App Dev* 546
in initialization file: *App Dev* 546
executing command from JAM client: *Lang Ref* 206; *App Dev* 551
executing command on JAM server: *App Dev* 552
hot links
creating for JAM client: *Lang Ref* 197; *App Dev* 549
specifying in initialization file: *App Dev* 550; *Config* 160
updated from JAM server: *App Dev* 547
hot paste links, creating for JAM client: *Lang Ref* 203; *App Dev* 549
links
created on JAM server: *App Dev* 546
creating for JAM client: *App Dev* 549–550
specifying in initialization file: *App Dev* 550; *Config* 160
updated from JAM server: *App Dev* 547
paste links
created on JAM server: *App Dev* 546
creating for JAM client: *Lang Ref* 202; *App Dev* 549
poking data from JAM client: *Lang Ref* 209; *App Dev* 552

DDE (continued)
poking data into JAM server: *App Dev* 552
requesting link data: *Lang Ref* 205; *App Dev* 551
updating JAM client data: *App Dev* 550
warm links
creating for JAM client: *Lang Ref* 198; *App Dev* 549
updated from JAM server: *App Dev* 547
warm paste links, creating for JAM client: *Lang Ref* 204; *App Dev* 549

Debugger: *Editors* 5; *App Dev* 495–518
accessing: *App Dev* 501–502
from Test mode: *Editors* 35
in application and test mode: *App Dev* 502
accessing source code: *App Dev* 506
animation: *App Dev* 500, 512
Application Data: *App Dev* 504
Application Data window: *App Dev* 518
breakpoints: *App Dev* 498
location: *App Dev* 513
setting: *App Dev* 512
setting in JPL: *App Dev* 509
setting on events: *App Dev* 513
sorting: *App Dev* 504
Breaks menu: *App Dev* 505
calling a function: *App Dev* 504
calling a function on breakpoint: *App Dev* 516
configuring: *App Dev* 499
Data Watch window: *App Dev* 517
database import tracing: *App Dev* 501
DBUG key: *App Dev* 497, 501, 502
dumping windows to log file: *App Dev* 504
Edit Breakpoints window: *App Dev* 515
add breakpoint: *App Dev* 517
event mode: *App Dev* 516
location mode: *App Dev* 516
Edit menu: *App Dev* 505
enable in screen editor: *App Dev* 501
enabling from screen editor: *Editors* 16
event filtering: *App Dev* 515
sub-events: *App Dev* 516
event stack: *App Dev* 498
exiting: *App Dev* 502
expert mode: *App Dev* 496, 500, 515
features: *App Dev* 495
file browsing: *App Dev* 506
file menu: *App Dev* 503
how it works: *App Dev* 496
log file preferences: *App Dev* 499

Debugger (continued)
menu bar: *App Dev* 503
monitor variables and expressions: *App Dev* 498
Open Source Module: *App Dev* 506
Options menu: *App Dev* 505
pending keys: *App Dev* 499
preferences: *App Dev* 499
saving preferences: *App Dev* 500, 503
source code, module type: *App Dev* 507
Source Code window: *App Dev* 497
step through execution: *App Dev* 511–512
Tools menu: *App Dev* 504
Trace menu: *App Dev* 505
tracing: *App Dev* 511
expert mode: *App Dev* 512
variable and expression monitoring: *App Dev* 517
view menu: *App Dev* 497
viewing control string assignments: *App Dev* 511
viewing screen information: *App Dev* 509
control strings: *App Dev* 511
field information: *App Dev* 509
group information: *App Dev* 510
screen information: *App Dev* 509
screen JPL: *App Dev* 509
viewing source code: *App Dev* 505
watch data, sorting: *App Dev* 504
Windows menu: *App Dev* 505

Decimal places, setting JPL default: *Config* 36

Decimal Places property: *Editors* 176

Decimal symbol: *Editors* 247
getting default: *Lang Ref* 406
setting default: *Lang Ref* 421; *Config* 69–70

DECIMAL_PLACES: *Config* 36

DECLARE CONNECTION, dbms command, making database connection: *Database* 156, 310; *App Dev* 213–215

DECLARE CURSOR, dbms command
creating database cursor: *Database* 157–158; *App Dev* 219–222, 250–252
for ODBC catalog functions: *Database* 276
for Oracle stored subprograms: *Database* 311
for RPCs: *Database* 399
using bind values: *App Dev* 219–221, 250–253
using colon expansion: *App Dev* 219

DECLARE TRANSACTION, dbms command, declaring a transaction: *Database* 400

Declaring hook functions. *See* Hook functions

Decorating screens: *Editors* 26

Decorations
on list box widget: *Editors* 201
on screen borders: *Editors* 110

Default push button, creating: *Editors* 194

Default/Cancel property: *Editors* 195

Defaults
setting for Motif: *Config* 169–188
setting for Windows: *Config* 153–167

Defining keys: *Config* 83–86

Delay cursor: *Lang Ref* 212; *App Dev* 559

Delayed write: *App Dev* 477; *Config* 37
flush: *Lang Ref* 50
flushing: *App Dev* 477
forcing: *Lang Ref* 251

DELE key (delete character), hex value: *Config 79*

Delete Order property: *Editors* 346
in automated SQL generation: *App Dev* 300

DELETE statement
constructing: *Database* 21–23
in JDB: *Database* 72
SQL generation from properties: *App Dev* 299

Deleting widgets, at runtime: *Lang Ref* 401

DELL key (delete line), hex value: *Config 79*

Demand hook functions: *App Dev* 116
example: *App Dev* 163–164
installing
field function: *App Dev* 128
group function: *App Dev* 133
screen function: *App Dev* 124

Depth property, for graph widget: *Editors* 136
bar/line graph: *Editors* 153
high/low chart: *Editors* 160
pie chart: *Editors* 145
XY plot: *Editors* 156

DESC keyword
in ORDER BY clause: *Database* 94
in Sort Widgets property: *App Dev* 288

describe, table in JISQL: *Database* 50

Deselect in selection group: *Lang Ref* 214

Design considerations, for screens: *Editors* 95–96

Detail specification
adding another table view: *Editors* 76
in screen wizard: *Editors* 75–76

DFLT_GROUP_FUNC. *See* Group function

DFLT_SCREEN_FUNC. *See* Screen function

DFLT_SCROLL_FUNC. *See* Scrolling array, alternative scroll driver

Dialog box, creating: *Editors* 98

Dialog property: *Editors* 99

Diameter property, for graph widget, pie chart: *Editors* 146

Digits only filter: *Editors* 162
and check digit calculation: *Editors* 319
and justification: *Editors* 162

Dim display attribute, setting: *Editors* 258

Dimensions of widgets. *See* Widgets, size

Direction property, for graph widget, pie chart: *Editors* 146

Disk–based scrolling. *See* Scrolling array, alternative scroll driver

Display
See also Terminal
getting HWND handle: *Lang Ref* 517
getting Widget ID: *Lang Ref* 517

Display area, size for portability: *App Dev* 479

Display attributes
as parameters: *Config* 21
defaults, assigning: *Config* 21
keywords: *Config* 140
keywords for video: *Config 107*
portability: *App Dev* 479–480
setting: *Editors* 257–258
for area: *Lang Ref* 176
for zoom window borders: *Config* 30
in messages: *Config* 57–60
in status line: *Lang Ref* 59, 224; *Config* 26, 57
video attribute handling types: *Config* 115–124
video attributes
ANSI terminals and: *Config* 120
combining: *Config* 121
for grayed menu items: *Config* 120
for message line: *Config* 125
for onscreen or area: *Config* 116–117
latch attributes: *Config* 120–123
video file keywords: *Config* 115–124

Display properties, for screens: *Editors* 33

Distinct property, for table view: *Editors* 352
  in automated SQL generation: *App Dev* 279

Division operation, in JDB: *Database* 90

DLL
  getting load error: *Lang Ref* 447
  installing function from: *Lang Ref* 448
  loading: *Lang Ref* 450

DM, database message tag prefix: *Config* 44

dm_, @dm global variables: *Database* 183–198

Dominant widget, defined: *Editors* 37

double (data type), in JDB: *Database* 70

Double Click property: *Editors* 306
  set by the screen wizard: *Editors* 84

Double clicking, getting time between clicks: *Lang Ref* 394

Double–click event, specifying: *Editors* 305

Drawing area: *Config* 179

Drawing function, attaching to widget: *Lang Ref* 165

Driver, keywords: *Config 108*, 132

DROP DATABASE statement, in JDB: *Database* 73

Drop shadows
  on character–mode screens: *Config* 33
  setting emphasis: *Config* 119

DROP TABLE statement, in JDB: *Database* 74

Drop–down Data property: *Editors* 177

Drop–down Screen property: *Editors* 178

Drop–down Size property: *Editors* 178

Drop–down Source property: *Editors* 177

dump, table in JISQL: *Database* 50

DW_OPTIONS: *Config* 37

Dynamic label widget: *Editors* 24, 118–119
  accessing with keyboard: *Editors* 300
  assigning double–click event to: *Editors* 305–306
  displaying image on: *Editors* 266
  resizing: *Editors* 119

Dynamic link library. *See* DLL

DynaText: *Editors* 280

E

ED keyword (video file): *Config* 112

Edit Mask: *Editors* 164–166

Edit menu: *Editors* 13

Edit menu items: *Editors* 222
  setting label: *Editors* 223

Editor
  invoking for JPL procedures: *Lang Ref* 12
  invoking from JPL dialog box: *Lang Ref* 13
  invoking to edit array at runtime: *Lang Ref* 235
  setting: *Lang Ref* 452; *Config* 17

Eight–bit character set: *Config* 93

EL keyword: *Config* 112

EMOH key (go to last field), hex value: *Config 79*

EMPHASIS: *Config* 33

Emphasis style, defining: *Config* 33

EMPHASIS_KEEPATT keyword (video file): *Config* 119

EMPHASIS_SETATT keyword (video file): *Config* 120

Empty Format property: *Editors* 248

Empty numeric field: *Editors* 248

EMSGATT: *Config* 27

Enable debugger: *Editors* 16

ENGINE, dbms command, setting database engine: *Database* 159; *App Dev* 211

Engine–specific Notes. *See* Database drivers

Engines. *See* Database engines

ENTEXT_OPTION: *Config* 37

Entry Function property
  for grid: *Editors* 332
  for grid frame widget: *Editors* 189
  for screen: *Editors* 334
  for selection group: *Editors* 333
  for widgets: *Editors* 330

Environment variables: *Config* 6, 14–18
  defining pixmap location: *Editors* 225

EOFD key (end of field), hex value: *Config 80*

EOLN key (end of line), hex value: *Config 80*

Equi–joins: *Database* 81

ER_ACK_KEY: *Config* 28, 80

ER_KEYUSE: *Config* 28

ER_SP_WIND: *Config* 28

Erase display command. *See* ED keyword

Erase line command. *See* EL keyword

Erase window command. *See* EW keyword

Erasure command keywords (video file): *Config 106*, 112

Error acknowledgment key
  and space bar: *Config* 80
  defining: *Config* 28

Error function: *App Dev* 137–138
  return codes: *App Dev* 137
  standard arguments: *App Dev* 137

Error handling: *App Dev* 199–203
  DLL loading: *Lang Ref* 447
  for menu API: *Lang Ref* 347
  for properties API: *Lang Ref* 412
  installing database error handler: *App Dev* 262–264

Error hook function: *App Dev* 201
  example: *App Dev* 173

Error messages
  See also Error messages (database); Message; Message file; Status line
  acknowledgment: *Config* 28–39, 59
  database drivers: *Database* 192–194
  JDB: *Database* 111–114
  transaction manager: *App Dev* 465–469
  translating: *App Dev* 489

Error messages (database): *App Dev* 255–263
  calling function after dbms command: *Database* 170–171
  calling function before dbms command: *Database* 165–166
  customized processing: *App Dev* 259–262
  default processing: *App Dev* 256
  engine–specific codes: *Database* 186–187
  engine–specific messages: *Database* 188, 214–218, 243–245, 266–268, 295–296, 331–332, 368–370; *App Dev* 257–259
  error handler: *App Dev* 260–261
  exit handler: *App Dev* 260–261

Error messages (database) (continued)
  generic database driver messages: *Database* 192–194, 195; *App Dev* 257–259
    listing: *Database* 192–194
  installing error handler: *Database* 167–169; *App Dev* 259–264
  testing in transaction manager: *Lang Ref* 466
  transaction error handling: *App Dev* 267–270
  transaction hook functions: *App Dev* 387
  warning codes: *Database* 190, 191; *App Dev* 257–259

Escape sequence, for setting cursor style: *Config* 109

EW keyword (video file): *Config* 112
  parameters: *Config 94*

EXECUTE
  dbms command, executing statement: *Database* 160; *App Dev* 219
  dbms statement, changing in SQL generation: *Lang Ref* 127

Executing SQL statements: *Database* 172

EXISTS keyword, in JDB: *Database* 100, 106

Exit Function property
  for grid: *Editors* 332
  for grid frame widget: *Editors* 189
  for screen: *Editors* 334
  for selection group: *Editors* 333
  for widgets: *Editors* 330

EXIT key, hex value: *Config 79*

Exit screen: *Lang Ref* 82

Exiting JAM: *Editors* 21

EXPHIDE_OPTION: *Config* 37

Expressions
  See also JPL expression; Regular expression
  in JDB: *Database* 91

EXT key (extend selection), hex value: *Config 80*

EXTD key (extend selection down), hex value: *Config 80*

Extended colors
  aliasing colors: *Config* 138–142
  defined: *Editors* 254
  specifying in Color palette: *Editors* 257

Extended keyboard: *Config* 84
  defining for MS–DOS: *Config* 110

Extensions. *See* Filename, extensions

External Help Tag property
for menu items: *Editors* 221
for widgets and screens: *Editors* 280

External menu: *Editors* 217; *Lang Ref* 360; *App Dev* 92

External screen: *Editors* 178

EXTU key (extend selection up), hex value: *Config 80*

F

F_EXTOPT: *Config* 34

F_EXTREC: *Config* 34

F_EXTSEP: *Config* 34

F11/F12 function keys
accessing: *Config 110*
defining: *Config* 85

f2asc: *App Dev* 565–567

FCASE: *Config* 33

FE_KEEPATTRS: *Config* 31

FE_SETATTRS: *Config* 31

FE_SWATTRS: *Config* 31

FERA key (clear field)
clock update and: *Editors* 245
hex value: *Config 79*

FETCH, transaction manager command, fetching next row of data: *App Dev* 434–435

Fetch Directions property: *App Dev* 377–378
for screen: *Editors* 340
for table view: *Editors* 339

FHLP key (screen help), hex value: *Config 79*

Field
See also Widgets
alphanumeric filter: *Editors* 164
characteristics, internationalization: *App Dev* 488–489
copying array data: *Lang Ref* 189
currency. *See* Currency format
database. *See* Database columns
date/time format. *See* Date/time format
decimal symbol: *Config* 70
digits only filter: *Editors* 162
displaying status of: *App Dev* 202
edit mask: *Editors* 164–166
entry/exit functions: *Editors* 330–331
function. *See* Field function
getting current field number: *App Dev* 79
MDT bit. *See* Validation
numeric filter: *Editors* 164
regular expression: *Editors* 166, 167
select on entry: *Editors* 173
validation. *See* Validation
VALIDED bit. *See* Validation
with yes/no entry: *Editors* 163

Field data
accessing in JPL: *Lang Ref* 27, 30
accessing substring: *Lang Ref* 28
clearing all fields: *Lang Ref* 185; *App Dev* 84
clearing all fields in table view: *Lang Ref* 459
clearing from array: *Lang Ref* 186; *App Dev* 84
copying to buffer: *Lang Ref* 270
getting length: *Lang Ref* 37, 217; *App Dev* 80
of word wrapped text: *Lang Ref* 510
reading: *Lang Ref* 256; *App Dev* 80
double precision float: *Lang Ref* 194
from LDBs: *Lang Ref* 157
integer: *Lang Ref* 290
long integer: *Lang Ref* 343
unformatted data: *Lang Ref* 453
word wrapped text: *Lang Ref* 512
testing
all fields for changes: *Lang Ref* 491; *App Dev* 82
for yes value: *Lang Ref* 295; *App Dev* 81
if null: *Lang Ref* 398; *App Dev* 81
testing for no value: *Lang Ref* 294; *App Dev* 81
forcing validation: *Lang Ref* 262
validating with check digit function: *Lang Ref* 183
writing: *Lang Ref* 423; *App Dev* 83–84
double precision floating point: *Lang Ref* 221
formatted data: *Lang Ref* 158
integer: *Lang Ref* 300
long integer: *Lang Ref* 345
word wrapped text: *Lang Ref* 513

Field exit, setting validation condition: *Config* 24

Field function: *App Dev* 124–129
example of automatic function: *App Dev* 159
example of demand function: *App Dev* 163
passing non–standard arguments into: *App Dev* 161
return codes: *App Dev* 128
standard arguments: *App Dev* 126

Field number
assignment: *Editors* 298; *App Dev* 78
getting for current field: *App Dev* 79
in math expressions: *Editors* 317
relative referencing in tab properties: *Editors* 308
specifying in tab properties: *Editors* 308

Field runtime properties. *See* Widget runtime properties

Field validation: *App Dev* 81
causes: *App Dev* 125
using table lookup: *Editors* 278–279

FIELD_FUNC. *See* Field function

File
export JDB database to text files: *Database* 123
getting path name: *Lang Ref* 229
import to JDB database from text files: *Database* 123
opening as binary read–only: *Lang Ref* 228

File I/O
closing file stream: *Lang Ref* 234
error handling: *Lang Ref* 236
getting file stream handle: *Lang Ref* 244
invoking external editor for array: *Lang Ref* 235
opening file for read/write: *Lang Ref* 245
reading characters from file: *Lang Ref* 241
reading line from file: *Lang Ref* 242
rewinding file stream: *Lang Ref* 250
setting error code: *Lang Ref* 238
writing array to file: *Lang Ref* 233
writing character to file: *Lang Ref* 247
writing file contents to array: *Lang Ref* 239
writing line to file: *Lang Ref* 248

File menu: *Editors* 12

File selection dialog box: *Lang Ref* 230
adding to file type option menu: *Lang Ref* 232

File stream. *See* File I/O

Filename
case sensitivity: *Config* 33
conventions for screens: *Editors* 20
extensions, setting defaults: *Config* 33
for key translation file: *Config* 75
setting default behavior: *Config* 33–34
specifying screen extension: *Config* 34

Fill at Init: *Editors* 178

Fill at Popup: *Editors* 178

Fill Character property: *Editors* 248

FINISH, transaction manager command, closing current transaction: *App Dev* 436–437

float (data type), in JDB: *Database* 69

Floating point
in calculations: *Editors* 317
reading from field: *Lang Ref* 194
writing to field: *Lang Ref* 221

Flow control. *See* Control flow

FLUSH, dbms command, throwing away unread rows: *Database* 342, 401

Flush buffered output: *Lang Ref* 50

FM, message tag prefix: *Config* 44

Focus
menu items: *Editors* 220
push button widgets: *Editors* 195–196

Focus properties
for screens: *Editors* 33
for widgets: *Editors* 32

Focus Protection property: *Editors* 303
for grid frame widgets: *Editors* 190
in styles editor: *Editors* 289

Font
aliasing: *Editors* 240
aliasing names for portability: *Editors* 236
application default: *Editors* 237
for grid frame widgets: *Editors* 184
for screen: *Editors* 237
GUI–specific names: *Editors* 239
italic: *Editors* 239
JAM–specific: *Editors* 237
point size: *Editors* 238
setting bold attribute: *Editors* 238
setting point size: *Editors* 238
specifying: *Editors* 236–240
specifying for graph widget: *Editors* 124–125

Font Name property: *Editors* 238

Font properties
for screens: *Editors* 33
for widgets: *Editors* 32

For loop: *Lang Ref* 51
skip to next iteration: *Lang Ref* 63

Force Valid property: *Editors* 354

FORCE_CLOSE, transaction manager command, discarding changes: *App Dev* 438–439

Foreground color: *Editors 256*
resource in Motif: *Config* 172

Foreign keys
defined: *Database* 12
defining using JISQL: *Database* 41–45
enforcing with validation link: *App Dev* 362–363

Form
See also Screen
closing: *Lang Ref* 301
opening: *Lang Ref* 303; *App Dev* 73

Form list. *See* Memory–resident list

Form stack: *App Dev* 70
return to base screen: *Lang Ref* 83

FORMAT, dbms command, formatting result set: *Database* 162–163; *App Dev* 237

Format selection, in screen wizard: *Editors* 69

Format Type property
for date/time fields: *Editors* 244
for numeric fields: *Editors* 247

Format/Display properties
See also Display properties
described: *Editors* 231–251
for widgets: *Editors* 32
numeric format: *Editors* 247–249
text format: *Editors* 233–241, 244

Formatting text
for a database: *App Dev* 239–249, 251–252
from a database: *App Dev* 234–237
in widgets: *Editors* 233–241

formlib: *App Dev* 561–563
with –g option: *Editors* 364
arguments: *Editors* 364
with –m option: *Editors* 368

formMenus: *Config* 176

Frames, creating: *Editors* 261–262

Frequency property: *Editors* 245

Frozen Columns property: *Editors* 186

funclist.c. *See* Hook functions

Function
See also Hook functions
aggregate. *See* Aggregate functions
calling from JPL: *Lang Ref* 47

Function keys
associating with control string: *App Dev* 109
hex value: *Config 81*
setting default behavior: *App Dev* 110; *Config* 25

Function list. *See* Hook functions

Function property, for transaction hook: *Editors* 339

G

GA_CURATT: *Config* 34

GA_CURMASK: *Config* 35

GA_SELATT: *Config* 35

GA_SELMASK: *Config* 35

Generate Item Selection screens: *Editors* 80

Geometry properties
for screens: *Editors* 33
for widgets: *Editors* 31

GIF files: *Editors* 266

Global data. *See* Application data

Global JPL variable
clearing: *Lang Ref* 20
declaring: *Lang Ref* 19, 53

Global string
getting: *Lang Ref* 406
setting: *Lang Ref* 421

Global variables
database drivers: *Database* 183–198
getting values: *Lang Ref* 285
setting values: *Lang Ref* 297

GRAPH keyword: *App Dev* 474
in video file: *Config* 126
using: *App Dev* 477

Graph properties: *Editors* 32

Graph widget: *Editors* 24, 121–160
bar/line graph: *Editors* 150–154
converting between chart types: *Editors* 143
creating: *Editors* 122–123
data series: *Editors* 136–143
displaying in 3D: *Editors* 135–136
fonts: *Editors* 124–125
high/low chart: *Editors* 157–160
label text: *Editors* 129
legend: *Editors* 126–130, 142–143

Graph widget (continued)
minimizing chart re–draw: *Editors* 138
orientation: *Editors* 129–130
pie chart: *Editors* 143–150
subtitle: *Editors* 125–126
text size: *Editors* 125
title: *Editors* 125
X and Y axes: *Editors* 130–135
XY plot: *Editors* 154–157

Graphics characters: *App Dev* 477
keywords: *Config 108*
supporting: *Config* 125–128

Graphics file
supported for toolbar items: *Editors* 224
supported formats: *Editors* 266

Graphics sets, defining: *Config* 126

Graphics widgets: *Editors* 26

Grayed menu items, setting emphasis: *Config* 119

Graying, inactive screens: *Config* 33

GRAYKEYS flag (video file): *Config 110*

Grid (screen)
align widget on: *Editors* 47
and wallpaper pixmaps: *Editors* 108
as unit of measurement: *Editors 41*
defining size of: *Editors* 99
snapping to: *Editors* 47

Grid Align command: *Editors* 47

Grid column
autonumbering: *Editors* 185
defining properties of: *Editors* 184–186
frozen: *Editors* 186
moving/resizing at runtime: *Editors* 189
positioning: *Editors* 185
titles: *Editors* 185
using as row title: *Editors* 188

Grid Column property: *Editors* 185

Grid display layout
description: *Editors* 83
specifying in screen wizard: *Editors* 78

Grid frame widget: *Editors* 181–190
and focus protection: *Editors* 303
assigning titles to columns: *Editors* 185
defining number of occurrences: *Editors* 187
deleting: *Editors* 183

Grid frame widget (continued)
entry/exit functions: *Editors* 332–333
font specification: *Editors* 237
hiding: *Editors* 234
horizontal lines: *Editors* 188
member types: *Editors* 183–184
scrolling: *Editors* 187
selecting: *Editors* 182
selecting members in: *Editors* 183
setting scrolling behavior: *Editors* 187
sizing to content: *Editors* 184
specifying in screen wizard: *Editors* 78
specifying number of rows: *Editors* 186–187
vertical lines: *Editors* 186

Grid function: *App Dev* 129–132
return codes: *App Dev* 131
standard arguments: *App Dev* 129

Grid Height property, for screens: *Editors* 99

Grid members
and focus protection: *Editors* 303
moving/resizing: *Editors* 189
removing: *Editors* 183
selecting: *Editors* 183

Grid property: *Editors* 182

Grid row
autonumbering: *Editors* 188
defining properties of: *Editors* 186–190
specifying number of: *Editors* 186–187
striping: *Editors* 188
titles: *Editors* 187
hiding: *Editors* 234

Grid Style property, for graph widget tick marks: *Editors* 135

Grid Width property, for screens: *Editors* 99

GRID_FUNC. *See* Grid function

Group
autotab: *Editors* 308
changing members of: *Editors* 210
check box widget: *Editors* 26
converting to field number: *Lang Ref* 277; *App Dev* 79
creating: *Editors* 206
cursor attributes: *Config* 34
controlling cursor movement: *Lang Ref* 318
display attributes: *Config* 34–35
entry/exit functions: *Editors* 333–334
forcing validation: *Lang Ref* 278

Group (continued)
getting name from field reference: *Lang Ref* 261; *App Dev* 80
identifying members of: *Editors* 207
list box widget: *Editors* 26
naming: *Editors* 209
occurrence attributes: *Config* 35
properties: *Editors 208*
radio button widget: *Editors* 26
referencing in JPL as variable: *Lang Ref* 31
selecting: *Editors* 207–208
selection widgets: *Editors* 25, 206
specifying allowable number of selections: *Editors* 209
specifying initial selection: *Editors* 209
specifying selection/deselection characters: *Config* 131
tabbing in: *Editors* 306
toggle button widget: *Editors* 26
validation: *App Dev* 132
See also Validation

GROUP BY clause
changing generated SQL: *Lang Ref* 133–134
in automated SQL generation: *App Dev* 284–287
in JDB: *Database* 75–76

Group By property: *Editors* 354
in automated SQL generation: *App Dev* 284–287

Group function: *App Dev* 132–134
example of automatic function: *App Dev* 164
return codes: *App Dev* 133
standard arguments: *App Dev* 132

Group member
changing: *Editors* 210
identifying: *Editors* 207

GROUP_FUNC. *See* Group function

GRTYPE keyword (video file): *Config* 126
keywords: *Config* 126

GUI–specific colors, using: *Editors* 254, 257

GUI–specific line/box styles: *Editors* 261

H

Handle. *See* HINSTANCE handle; HWND handle; Widget ID

Hard reset. *See* RESET keyword

HAVING clause
changing generated SQL: *Lang Ref* 135–136
in automated SQL generation: *App Dev* 287–288
in JDB: *Database* 77–78

Having property: *Editors* 355
in automated SQL generation: *App Dev* 287–288

Header file
creating: *Config* 53–56
sample: *Config* 53

Height property
defined: *Editors* 40
for widgets: *Editors* 40

Help
See also Help properties; Help screen
multilevel: *Editors* 274

Help function: *App Dev* 134
example: *App Dev* 166
return codes: *App Dev* 134
standard arguments: *App Dev* 134

HELP key (field help), hex value: *Config 79*

Help properties
for menu items: *Editors* 220
for screens: *Editors* 33
for widgets: *Editors* 32

Help screen: *Editors* 272–276; *Lang Ref* 182, 296
attaching: *Editors* 275
displaying: *Editors* 275
external: *Editors* 280
populating: *Editors* 273
positioning: *Editors* 275–276
to enter data: *Editors* 273–274

Hexadecimal strings
converting binary columns: *App Dev* 235
writing to database: *App Dev* 245, 249

Hidden property: *Editors* 233
and version columns: *Editors* 359

Hiding
horizontal lines in grid frames: *Editors* 188
vertical lines in grid frames: *Editors* 186

Hiding text: *Editors* 234

Hiding widgets: *Editors* 233
in screen wizard output: *Editors* 87
row titles in grids: *Editors* 234

High/low chart: *Editors* 157–160
creating: *Editors* 158–159
data series style: *Editors* 139–140
data source: *Editors* 160
displaying in 3D: *Editors* 159–160
legend: *Editors* 126, 127, 142–143

HINSTANCE handle: *Lang Ref* 395

HOME key, hex value: *Config 79*

Hook function arguments
check digit: *App Dev* 139
control: *App Dev* 142
error: *App Dev* 137
field: *App Dev* 126
grid: *App Dev* 129
group: *App Dev* 132
help: *App Dev* 134
initialization: *App Dev* 140
insert toggle: *App Dev* 138
key change: *App Dev* 136
playback: *App Dev* 141
record: *App Dev* 141
reset: *App Dev* 140
screen: *App Dev* 122
timeout: *App Dev* 135
transaction manager: *App Dev* 146
video processing: *App Dev* 144

Hook function return codes
check digit: *App Dev* 139
control: *App Dev* 142
error: *App Dev* 137
field: *App Dev* 128
Grid: *App Dev* 131
group: *App Dev* 133
help: *App Dev* 134
initialization: *App Dev* 140
insert toggle: *App Dev* 138
key change: *App Dev* 136
playback: *App Dev* 141
record: *App Dev* 141
reset: *App Dev* 140
screen: *App Dev* 124
status line: *App Dev* 143
timeout: *App Dev* 135
transaction manager: *App Dev* 147, 385–388
video processing: *App Dev* 145

Hook function types
check digit: *App Dev* 139
control: *App Dev* 142
database driver errors: *App Dev* 146
error: *App Dev* 137
field: *App Dev* 124
grid: *App Dev* 129
group: *App Dev* 132
help: *App Dev* 134
initialization: *App Dev* 140
insert toggle: *App Dev* 138
key change: *App Dev* 136
playback: *App Dev* 141
prototyped: *App Dev* 120
record: *App Dev* 141
reset: *App Dev* 140
screen: *App Dev* 122
status line: *App Dev* 143
timeout: *App Dev* 134
transaction manager: *App Dev* 146–147, 384–393
video processing: *App Dev* 144

Hook functions
See also Hook function types
automatic: *App Dev* 116
demand: *App Dev* 116
executing from control string: *Editors* 325–327
installing: *Lang Ref* 218, 288; *App Dev* 117–120
invoking: *Lang Ref* 15
from grid frame widget: *Editors* 332–333
from group: *Editors* 333–334
from screen: *Editors* 334–335
from widgets: *Editors* 330–331
standard arguments: *App Dev* 117
transaction manager: *App Dev* 384–393

Horiz Rotation property, for graph widget: *Editors* 136
bar/line graph: *Editors* 152–153
high/low chart: *Editors* 159–160
XY plot: *Editors* 156

Horizontal Anchor property, for widgets: *Editors* 50

Horizontal array, setting tab order in: *Editors* 309

Horizontal lines, in grid frames: *Editors* 188

Horizontal property: *Editors* 242

Horizontal Scroll Bar property
for grid frames: *Editors* 185
for shifting fields: *Editors* 244

Horizontal Shrinking property, for a screen: *Editors* 105, 264

HWND handle, getting for
display: *Lang Ref* 517
drawing area: *Lang Ref* 220
screen–resident widget: *Lang Ref* 499

I

I/O processing: *App Dev* 473–477
See also File I/O

Icon
file types supported for: *Editors* 102
identification: *Editors* 102

Icon property: *Editors* 102

Identity properties
for screens: *Editors* 33
for widgets: *Editors* 31

If logic: *Lang Ref* 55

Import: *Editors* 59–62
of database objects to a repository: *Database* 208–211, 240–242, 261–265, 290–293, 324–328, 359–363; *App Dev* 63–64, 310–312
re–import: *Editors* 61
results of: *Editors* 60

In Data Space property, for graph widget legend: *Editors* 127–128

In Delete Where property: *Editors* 358
in automated SQL generation: *App Dev* 301

IN keyword
in automated SQL generation: *App Dev* 283
in JDB: *Database* 100, 106

In Update Where property: *Editors* 358
in automated SQL generation: *App Dev* 299

IN_BLOCK: *Config* 22

IN_ENDCHAR: *Config* 23

IN_HARROW: *Config* 22

IN_RESET: *Config* 23

IN_VALID: *Config* 24

IN_VARROW: *Config* 23

IN_WRAP: *Config* 24

Inactive Pixmap property
for menu items: *Editors* 224
for push button widget: *Editors* 193
for toolbar items: *Editors* 219
for widget: *Editors* 267

Inches, as unit of measurement: *Editors 41*

Include JPL module: *Lang Ref* 57

Included JPL modules: *Lang Ref* 6

IND_OPTIONS: *Config* 29

IND_PLACEMENT: *Config* 30

Indicator property, for menu items: *Editors* 219

Indicator symbol
keywords (video file): *Config 108*, 130–132
setting initial state on menu item: *Editors* 219
submenu in character JAM: *Config* 132

Informix: *Database* 205–236
connection options: *Database* 208
DBMS command listing: *Database* 234
executing stored procedures: *Database* 218
setting cursor behavior: *Database* 232–234

Inh button: *Editors* 29

Inherit From property
removing specification: *Editors* 64
setting in repository entry: *Editors* 60
specifying: *Editors* 65
syntax for widget inheritance: *Editors* 63

Inherit menu option: *Editors* 15, 64

Inheritance: *App Dev* 65–67
controlling: *Editors* 63–66
displayed in properties window: *Editors* 29
ensuring: *Editors* 53–55
finding child widget: *Editors* 65–66
finding parent object: *Editors* 65–66
Inherit From property: *Editors* 63
specifying: *Editors* 65
maintained when copying: *Editors* 44
messages: *Editors* 15, 64
preventing propagation: *Editors* 64
propagating changes: *Editors* 54
properties inherited from repository: *Editors* 63
restoring: *Editors* 64
source
finding: *Editors* 65–66
setting: *Editors* 65
toggling for a specific property: *Editors* 29
turning on/off: *Editors* 64
updating (binherit): *App Dev* 66–68

INIT keyword (video file): *Config* 109
  undoing effects of: *Config* 111

Init Selections property
  for radio buttons: *Editors* 204
  for selection group: *Editors* 209

Initial Text property
  embedding punctuation: *Editors* 163
  for list box widget: *Editors* 203
  for widgets: *Editors* 232

Initial Value property, for scales: *Editors* 176

Initialization
  application: *Lang Ref* 282
    options in Motif: *Config* 174–177
    options in Windows: *Config* 156
  database engines: *Lang Ref* 149–150; *App Dev* 207–211
  database interface: *Lang Ref* 117
  JAM: *Config* 8
  key translation table: *Lang Ref* 315
  menu system: *Lang Ref* 367
  messages: *Lang Ref* 391
  video translation table: *Lang Ref* 496

Initialization file. *See* Windows initialization file

Initialization function: *App Dev* 140–141
  example: *App Dev* 175
  return codes: *App Dev* 140
  standard argument: *App Dev* 140

Initialization property: *Editors* 178

Input
  keyboard: *App Dev* 474–476
  simulating from keyboard: *Lang Ref* 85, 493
  test for keyboard activity: *Lang Ref* 313

Input devices, for data: *Editors* 175–180

Input filters: *Editors* 162–171

Input properties, for widgets: *Editors* 32

Input Protection property: *Editors* 173, 304
  in styles editor: *Editors* 290

INS key (insert/overwrite), hex value: *Config 79*

INSCRSR_FUNC. *See* Insert toggle function

Insert Order property: *Editors* 346

INSERT statement
  constructing: *Database* 21
  in JDB: *Database* 79–80
  NULL values and: *Database* 88
  SQL generation from properties: *App Dev* 292–296, 302–303

Insert toggle function: *App Dev* 138–139
  example: *App Dev* 174
  return codes: *App Dev* 138
  standard argument: *App Dev* 138

INSL key (insert line), hex value: *Config 79*

INSOFF keyword (video file): *Config* 114

INSON keyword (video file): *Config* 114

Instance, getting handle: *Lang Ref* 395

Insufficient space, for widgets: *Editors* 49

int (data type), in JDB: *Database* 69

Integer value
  reading from field: *Lang Ref* 290
  writing to field: *Lang Ref* 300

Interactive SQL. *See* ISQL; JISQL

International characters, supporting: *Config* 85

Internationalization: *App Dev* 481–494
  8–bit characters: *App Dev* 482–483; *Config* 85, 125
  alternate message files: *Config* 71
  currency formats: *App Dev* 486–488; *Config* 66
  date/time formats: *App Dev* 483–486; *Config* 65
    substitution variables: *App Dev 485*
  decimal symbol: *App Dev* 488; *Config* 69–70
  keystroke filters: *App Dev* 488–489
  library functions: *App Dev* 482
  messages: *App Dev* 482
  of application screens: *App Dev* 490–493
  range checks: *App Dev* 493–494
  status and error messages: *App Dev* 489–490
  supporting: *Config* 127
  yes/no values: *Config* 71

Interrupt handler: *Lang Ref* 182; *App Dev* 140

Is Help property, for menu item: *Editors* 220

ISQL: *Database* 118–120
  clearing the input buffer: *Database* 120
  command terminator: *Database* 119
  committing transactions: *Database* 120
  connecting to a database: *Database* 120

ISQL (continued)
editing an ISQL statement: *Database* 120
executing a command file: *Database* 120
exiting: *Database* 120
starting: *Database* 118

Italic property: *Editors* 239

Item Selection key (ITSEL): *Editors* 276

ITSEL key (item selection): *Editors* 276

J

JAM
about: *Get Started* 1–15
modifying: *App Dev* 521
product family: *Get Started* 11

JAM basic colors, keywords: *Config* 140

JAM events: *App Dev* 495

JAM Help property, for menu item: *Editors* 220

JAM help screen
See also Help screen
displaying: *Lang Ref* 279

JAM type
character strings
fetching from database: *App Dev* 234
writing to database: *App Dev* 244–245, 247–251
converting to C type: *App Dev* 243
currency formats, writing to database: *App Dev* 243, 245–246
date and time formats
fetching from database: *App Dev* 234
writing to database: *App Dev* 243, 244–246, 248
from database column type: *Database* 210–211, 241–242, 263–264, 291–292, 326–327, 361–362
hexadecimal strings, writing to database: *App Dev* 245
numeric data
fetching from database: *App Dev* 235
writing to database: *App Dev* 245–246, 249
using to enter data: *App Dev* 241–244
using to format selected data: *App Dev* 234–237

jam.ini: *Config* 154
sample: *Config* 161–167

JAM/Case *interface*, about: *Get Started* 14

JAM/ReportWriter, about: *Get Started* 11

JAM/TP*i*, about: *Get Started* 12

jamdev, starting: *Editors* 9

jamdev message tag: *Config* 44

JDB
connecting to database using JISQL: *Database* 34–35
connection options: *Database* 240
creating databases: *Database* 27, 64–65, 118
creating databases in JISQL: *Database* 36
creating tables using JISQL: *Database* 36–37
database driver for: *Database* 237–253
DBMS command listing: *Database* 251
defining columns using JISQL: *Database* 38–39
defining table keys using JISQL: *Database* 39–45
describing: *Database* 3–8
disconnecting from database using JISQL: *Database* 35
error messages: *Database* 111–114
executing transactions: *Database* 115–116
ISQL: *Database* 118–120
JISQL: *Database* 33–55
keywords: *Database* 125–128
naming conventions: *Database* 27–28
SQL commands: *Database* 57–109
SQL syntax summary: *Database* 109
system tables: *Database* 30–32
unsupported features: *Database* 7
utilities: *Database* 117–120

jdbroll, restoring transaction log: *Database* 121

JISQL: *Database* 33–55
command terminator: *Database* 49
committing transactions: *Database* 50
connecting to a database: *Database* 34–35, 50
creating databases: *Database* 36
creating tables: *Database* 36–37
defining columns: *Database* 38–39
disconnecting from a database: *Database* 35
displaying database description: *Database* 46
dropping databases: *Database* 47
dropping tables: *Database* 46–47
editing SQL scripts: *Database* 47–49
executing operating system commands: *Database* 35–36
executing SQL scripts: *Database* 50–55
exiting: *Database* 35
log file: *Database* 51, 53–54
macro commands: *Database* 49–50

JISQL (continued)
output options: *Database* 50–51
query results: *Database* 52–53
rolling back transactions: *Database* 50
running interactive SQL: *Database* 47–55
script format: *Database* 49–52
starting: *Database* 33
terminating execution: *Database* 55

JM, message tag prefix: *Config* 44

jmain.c: *Config* 154, 170
See also Source code, main routines

Join
database tables: *Database* 22–23, 81–85
implementing self–join: *Editors* 93
in automated SQL generation: *App Dev* 289
table views: *Editors* 345
using correlation names: *Database* 23

JPEG files: *Editors* 266

JPL
calls. *See* JPL calls
choosing an editor: *Lang Ref* 13; *Config* 17
commands, summary: *Lang Ref* 41–43
comments: *Lang Ref* 7
compared to compiled code: *App Dev* 525–527
constants: *Lang Ref* 23
control flow: *Lang Ref* 6
displaying messages: *Lang Ref* 58
generated by screen wizard: *Editors* 86
memory–resident: *App Dev* 524
modules. *See* JPL module; JPL procedure
null statement: *Lang Ref* 6
optimizing performance: *Lang Ref* 39
reading send data. *See* Send data
sending data. *See* Send data
stubbing out: *App Dev* 527
validation: *Lang Ref* 8
variables. *See* JPL variable

JPL calls: *Lang Ref* 14
arguments: *Lang Ref* 14
from C function: *Lang Ref* 305
from control string: *Editors 322*, 325–327; *Lang Ref* 16; *App Dev* 112–113
from screen: *Editors* 328; *Lang Ref* 15

JPL calls (continued)
from widget: *Lang Ref* 15
inline calls: *Lang Ref* 17
return value: *Lang Ref* 15
search order: *Lang Ref* 17
to JPL and installed functions: *Lang Ref* 47

JPL expression: *Lang Ref* 38
bitwise: *Lang Ref* 39
logical: *Lang Ref* 39
numeric: *Lang Ref* 38
numeric format: *Lang Ref* 38
operand conversion: *Lang Ref* 35
precision: *Lang Ref* 38
specifying substring in variable: *Lang Ref* 36
string: *Lang Ref* 38

JPL module: *Lang Ref* 3
adding to memory–resident list: *Lang Ref* 254
calling by name: *Lang Ref* 10
compiling: *Lang Ref* 11
with jpl2bin: *Lang Ref* 11
compiling at runtime: *Lang Ref* 10
continuation character: *Lang Ref* 6
external modules: *Lang Ref* 9
include module: *Lang Ref* 57
including external modules: *Lang Ref* 6
line length: *Lang Ref* 6
loading as public: *Lang Ref* 10, 68, 306
unloading public: *Lang Ref* 307
memory–resident: *Lang Ref* 11
named procedure: *Lang Ref* 3
screen: *Lang Ref* 9
storing in library: *Lang Ref* 10
types: *Lang Ref* 7
unloading public: *Lang Ref* 75
unnamed procedure: *Lang Ref* 3
widget validation: *Lang Ref* 8

JPL operators: *Lang Ref* 33
@date: *Lang Ref* 36
@length: *Lang Ref* 37
@sum: *Lang Ref* 37
bitwise: *Lang Ref* 37
concatenation: *Lang Ref* 35
precedence: *Lang Ref* 35
substring specifier: *Lang Ref* 36

JPL procedure: *Lang Ref* 3
attaching to screen: *Editors* 328
attaching to widget: *Editors* 329–330
declaring parameters: *Lang Ref* 4
declaring return type: *Lang Ref* 5, 66
execution: *Lang Ref* 6
getting standard arguments: *Lang Ref* 5
named: *Lang Ref* 3
returning from: *Lang Ref* 15, 72
unnamed: *Lang Ref* 3

JPL Procedures property: *Editors* 328

JPL program text window: *Lang Ref* 12
compiling and saving: *Lang Ref* 14
invoking local editor: *Lang Ref* 13
reading and writing files: *Lang Ref* 13

JPL Validation property: *Editors* 329

JPL variable: *Lang Ref* 18
allocate size: *Lang Ref* 76
declaring: *Lang Ref* 18, 76
as array: *Lang Ref* 76
global: *Lang Ref* 19, 53
expanding to literal value: *Lang Ref* 20
initialize: *Lang Ref* 76
name conventions: *Lang Ref* 76
referencing group selection: *Lang Ref* 31
resolving name ambiguity: *Lang Ref* 26
scope and lifetime: *Lang Ref* 20
substring specifier: *Lang Ref* 36
watching through debugger: *App Dev* 517

jpl2bin: *Lang Ref* 11

Jterm, enabling data compression: *App Dev* 522; *Config* 132

Justification, with digits only filter: *Editors* 162

Justification property: *Editors* 234
for box widgets: *Editors* 262

JX, message tag prefix: *Config* 44

jxmain.c: *Config* 154, 170

K

KBD_DELAY keyword: *App Dev* 475
in video file: *Config* 111

Key
See also Key label; Key translation; Key translation file
changing cursor control key behavior: *Lang Ref* 317
classes for PC extended keyboards: *Config* 84
disabling: *Lang Ref* 317
getting integer value: *Lang Ref* 311
getting logical value: *Lang Ref* 272
label. *See* Key label
logical: *Config* 77
See also Logical key
defined: *Config* 73
displaying in message: *Editors* 271
hexadecimal values: *Config* 78–82
mnemonics: *Config* 78–82
pushing onto input queue: *Lang Ref* 493
remapping, for viewing select set: *App Dev* 233
routing: *App Dev* 475–476
translation. *See* Key translation

Key change function: *App Dev* 136–137
example: *App Dev* 171
return codes: *App Dev* 136
standard argument: *App Dev* 136

Key classes for PC extended keyboards: *Config* 84

Key columns
foreign key, defined: *Database* 12
primary key, defined: *Database* 11–12

Key label
displaying in messages: *Lang Ref* 60, 224; *Config* 59
portability: *App Dev* 480

Key translation: *App Dev* 474–475
initializing table: *Lang Ref* 315
installing file: *Lang Ref* 315
internationalization: *App Dev* 483
portability: *App Dev* 480
variable: *Config* 15

Key translation file: *Config* 73–88
accessing: *Config* 88
comments: *Config* 77
converting to binary (key2bin): *Config* 86–87
creating and modifying: *Config* 83–86
defining as SMKEY variable: *Config* 83
identifying for initialization: *Config* 15
memory–resident: *Config* 86
modifying: *Config* 83–84
multiple: *Config* 75

Key translation file (continued)
names of: *Config 2*
naming convention: *Config* 75
purpose: *Config* 74–75
syntax: *Config* 77–83
using alternate files: *Config* 87

key2bin: *Config* 86–87
error messages: *Config* 87

Keyboard
assigning timing interval: *Config* 111
extended: *Config* 84
for MS–DOS: *Config* 110
logical, mnemonics and hex values: *Config 79–82*
more than one type: *Config* 87
opening for input: *Lang Ref* 284
portability: *App Dev* 480
processing: *App Dev* 473–477

Keyboard interface: *Editors* 371–375
Invoking pop–up menu without mouse: *Lang Ref* 411; *App Dev* 96

KEYCHG_FUNC. *See* Key change function

Keys, defining using JISQL: *Database* 39–45

Keystroke Filter property: *Editors* 162
alphabetic : *Editors* 164
digits only: *Editors* 162
edit mask: *Editors* 164–166
in styles editor: *Editors* 289
numeric : *Editors* 164
regular expression: *Editors* 166
translation support: *App Dev* 488–489
using in database updates: *App Dev* 243
using to format database values: *App Dev* 248, 249
yes/no entries: *Editors* 163

Keytop. *See* Key label

Keywords
database drivers: *Database* 199–203
in JDB: *Database* 125–128

L

Label. *See* Key label

Label Location property
for graph widget, pie chart: *Editors* 148
for graph widget axes: *Editors* 131–132

Label property
for box widgets: *Editors* 262
for check box widget: *Editors* 200
for graph widget: *Editors* 122
for graph widget axes: *Editors* 131
for menu items: *Editors* 217
for push button widget: *Editors* 192
for static label widgets: *Editors* 118
for toggle button widget: *Editors* 205

Label Source property
for graph widget, pie chart: *Editors* 147–148
for graph widget tick marks: *Editors* 134

Label text display, setup variables: *Config* 30

Label widgets, creating: *Editors* 117–119

Language. *See* Internationalization

LARR key (left arrow), hex value: *Config 79*

Latch attributes
defined: *Config* 115
setting: *Config* 120–123

LATCHATT keyword (video file): *Config* 120–123

Layout selection, in screen wizard: *Editors* 78

LDB: *App Dev* 191–195
activating
at application startup: *App Dev* 193
at runtime: *App Dev* 194
activating at runtime: *Lang Ref* 339
and widget names: *Editors* 299
changing to read–only at runtime: *Lang Ref* 339
changing to read/write at runtime: *Lang Ref* 339
default library: *Lang Ref* 452
default screen: *Lang Ref* 452
disabling write–through: *Lang Ref* 195
forcing read from screen: *Lang Ref* 344
getting
contents of entry: *Lang Ref* 328
current state : *Lang Ref* 338
LDB name: *Lang Ref* 333
most recently activated: *Lang Ref* 324
most recently inactivated: *Lang Ref* 325
previously activated: *Lang Ref* 326
previously inactivated: *Lang Ref* 327
handle
getting: *Lang Ref* 329
getting to another instance: *Lang Ref* 334
identifying files for initialization: *Config* 18
inactivating at runtime: *Lang Ref* 339
initializing: *Lang Ref* 330

LDB (continued)
loading: *Lang Ref* 332
at application startup: *App Dev* 193
at runtime: *App Dev* 194
multiple instances of: *App Dev* 193
popping: *Lang Ref* 335; *App Dev* 194
pushing: *Lang Ref* 336; *App Dev* 194
read–only: *App Dev* 194
reading data from all: *Lang Ref* 157
referencing entries: *App Dev* 195
referencing in JPL: *Lang Ref* 26
screen functions and: *Config* 37
selection group data write–through: *App Dev* 192
testing whether loaded: *Lang Ref* 331
unloading at runtime: *Lang Ref* 341
write–through and screen entry: *App Dev* 192
writing to entry: *Lang Ref* 337

Legend, graph widget: *Editors* 126–130
identifying the data series: *Editors* 142–143
placement by location: *Editors* 127–128
placement by position: *Editors* 128–130

Legend property, for graph widget data series: *Editors* 142–143

Length property
defined for widgets: *Editors* 40
defined in database: *Editors* 353
for shifting fields: *Editors* 243

Letters only: *Editors* 164

Library
See also DLL
closing: *Lang Ref* 320
creating: *App Dev* 561–563
lock on: *Editors* 366
maintaining: *App Dev* 561–563
naming conventions: *Editors* 20
opening: *Lang Ref* 321
screen as form: *Lang Ref* 252
screen as window: *Lang Ref* 501
screen
opening: *Editors* 18
saving: *Editors* 20–21
storing JPL modules: *Lang Ref* 10
synchronize with source code management: *Editors* 367
under source code management: *Editors* 363–364

Library member
creating: *Editors* 97
in use: *Editors* 18
naming conventions: *Editors* 20
opening: *Editors* 18

Library message tag: *Config* 44

Library screen. *See* Library member

LIKE predicate, in JDB: *Database* 86–87, 107

Line drawing
for boxes: *Config* 130
keywords: *Config 108*, 128–130

Line feed, video file entry: *Config* 113

Line graph. *See* Bar/line graph; Graph widget

Line length of JPL statement: *Lang Ref* 6

Line Style property, for graph widget data series: *Editors* 141

Line styles
setting in cmap file: *Config* 145
table of style names: *Config 145*

Line widget: *Editors* 26, 259–263
3D (in Windows): *Editors* 251
default style: *Editors* 260
specifying alias style: *Editors* 260
specifying style: *Editors* 261

Line Width property, for graph widget data series: *Editors* 141

Line/Box Style property: *Editors* 260
3D effect on: *Editors* 251

LINES keyword (video file): *Config* 111

LINEWRAP flag (video file): *Config* 116

Link widget: *Editors* 27
automatically created: *Editors* 61
creating: *Editors* 343–344
properties: *Editors* 344–349
runtime properties: *Lang Ref* 551
selecting: *Editors* 344

Links: *App Dev* 313, 357–364
creating: *App Dev* 357–358
from database import: *Database* 209, 241, 263, 291, 325, 360
guidelines for using: *App Dev* 464
in automated SQL generation: *App Dev* 289–293
noncyclic: *Editors* 344
restrictions: *App Dev* 359

Links (continued)
sequential: *App Dev* 314–315, 359–360
server: *App Dev* 314–315, 359–360
setting child table view: *App Dev* 314, 358–359
setting parent table view: *App Dev* 314, 358–359
specifying in screen wizard: *Editors* 76–77
transitive: *Editors* 92
traversal properties: *App Dev* 382
validation: *Editors* 349, 359; *App Dev* 316, 361–364
adding lookup: *App Dev* 363–364
enforcing foreign keys: *App Dev* 362

List box widget: *Editors* 26, 200–206
3D (in Windows): *Editors* 250
and autotab behavior: *Editors* 312
as part of combo box: *Editors* 24
assigning double–click event to: *Editors* 305–306
attaching an action to: *Editors* 203
decorations: *Editors* 201
enabling extended selection: *Config* 37
executing control string: *Editors* 323
label: *Editors* 201
populating: *Editors* 203
scrolling: *Editors* 202
selecting items: *Editors* 201
title: *Editors* 201

List command, for parameter indexing: *Config* 100

Listbox Type property: *Editors* 201, 203

LISTBOX_SELECTION: *Config* 37

Local Data Block. *See* LDB

Local decimal symbol: *Config* 69

Locating child widget: *Editors* 65

Locating parent object: *Editors* 65–66

Location property, for graph widget axes: *Editors* 130–131

Lock, on library/repository: *Editors* 366

Locking shifts: *Config* 127

Log file, JISQL: *Database* 53–54

Logical expression: *Lang Ref* 39

Logical key: *App Dev* 474
changing behavior at runtime: *Config* 84
changing mapping of: *Config* 84
defined: *Config* 74
getting integer value: *Lang Ref* 311
getting label: *Lang Ref* 316
invoking control string from: *Editors* 323; *App Dev* 109
mnemonics and hex values: *Config 78–82*
required by jamdev: *Config 79*

Logical keyboard, vs. physical keyboard: *Config* 75

Logical operators, in JDB: *Database* 92

logon, connecting to JDB database: *Database* 50, 120

long (data type), in JDB: *Database* 69

Long integer
reading from field: *Lang Ref* 343
writing to field: *Lang Ref* 345

Look and feel
controlling: *Editors* 231–251
using repository to define: *Editors* 63

Look, don't feel. *See* Look and feel

Lookup specification, in Relations dialog box: *Editors* 348; *App Dev* 363

Lookup table: *Editors* 278–279

Loop
breaking from: *Lang Ref* 46
for condition: *Lang Ref* 51
skipping to next iteration: *Lang Ref* 63
while condition: *Lang Ref* 78

LP key (local print)
defining default: *Config* 35
hex value: *Config 79*

LSHF key (left shift), hex value: *Config 79*

LWRD key (previous word), hex value: *Config 79*

M

m2asc: *App Dev* 97

Macro commands, JISQL: *Database* 49–50

Major Increment property, for graph widget tick marks: *Editors* 133

MARKCHAR keyword (video file): *Config* 131

Master (only) format: *Editors* 70
specifying layout: *Editors* 78

Master–Detail format: *Editors* 70
defining link for: *Editors* 76–77
specifying layout: *Editors* 78

Master–Detail–Detail format: *Editors* 92

Master–Detail–Subdetail format: *Editors* 70
specifying layout: *Editors* 78

Math expression: *Editors* 316–319
international support: *App Dev* 494
specifying in function call: *Lang Ref* 181

Max Data Length property, for shifting fields: *Editors* 244

MAX function, in JDB: *Database* 59

Max Occurrences property
for grid frames: *Editors* 187
for list boxes: *Editors* 202
for scrolling array: *Editors* 243

Max Size property: *Editors* 43

Max/Min property: *Editors* 101

Maximize option, on screens: *Editors* 101

Maximum Decimals property: *Editors* 247

Maximum property, for graph widget axes: *Editors* 133

Maximum Value property
for scales: *Editors* 176
setting: *Editors* 171

MB_BORDATT: *Config* 32

MB_BORDSTYLE: *Config* 32

MB_DISPATT: *Config* 32

MB_FLDATT: *Config* 32

MB_HBUTDIST: *Config* 32

MB_KEEPATTRS: *Config* 31

MB_LINES_PROT: *Config* 32

MB_SETATTRS: *Config* 32

MB_SWATTRS: *Config* 32

MB_SYSTEM: *Config* 32

MDI frame, placement of window in: *Config* 157

MDT bit: *App Dev* 81
See also Validation
clearing for all fields: *Lang Ref* 184; *App Dev* 82
setting: *App Dev* 81
testing to find first modified field: *Lang Ref* 491; *App Dev* 82

Measurement, valid units of: *Editors 41*

Memo Text property
for menu items: *Editors* 221
for screens: *Editors* 116

Memory
allocating for application: *Lang Ref* 282
deallocating on exit: *Lang Ref* 426
optimization: *App Dev* 521–527

Memory model, compiling for small and medium: *App Dev* 525

Memory–resident
configuration files: *App Dev* 523
JPL modules: *Lang Ref* 11; *App Dev* 524
key translation file: *Config* 86
message file: *App Dev* 524
screens: *App Dev* 522
installing: *App Dev* 522
user messages: *Config* 46
video file: *Config* 104

Memory–resident list
preparing files for: *App Dev* 564
purging: *Lang Ref* 431
updating: *Lang Ref* 254

Menu
ASCII format: *Editors* 228; *App Dev* 98
ASCII/binary conversion: *App Dev* 97
attaching to screen as menu bar: *Editors* 113
changing properties: *Lang Ref* 349
creating: *Editors* 215–217
creating at runtime: *Lang Ref* 352; *App Dev* 95
definition: *App Dev* 87
deleting at runtime: *Lang Ref* 353; *App Dev* 95
deleting items at runtime: *App Dev* 95
displaying as toolbar: *App Dev* 87, 92
external reference: *Lang Ref* 360; *App Dev* 92
formMenus resource: *Config* 176
getting error on menu function calls: *Lang Ref* 347
getting property: *Lang Ref* 354
hierarchical view in editor: *Editors* 226–227
identical instances of: *Lang Ref* 359
in character–mode: *Config* 31
inserting items at runtime: *App Dev* 95

Menu (continued)
installing: *Lang Ref* 357; *App Dev* 89–92
for application: *App Dev* 90
for screen: *App Dev* 90
for widget: *App Dev* 90
identical instances of: *App Dev* 90
unique instances of: *App Dev* 91
item. *See* Menu item
loading script into memory: *App Dev* 88
memory location constants: *Lang Ref* 382
naming: *Editors* 217
pop–up for field: *Editors* 301
invoking: *App Dev* 96
invoking from keyboard: *Lang Ref* 411; *App Dev* 96
pop–up for screen: *Editors* 114
pop–up title: *Editors* 217
properties of: *Editors* 217
property constants: *Lang Ref* 350
removing from display: *Lang Ref* 362; *App Dev* 96
saving in editor: *Editors* 227
scope assignment and display: *Lang Ref* 359; *App Dev* 89
scope constants: *Lang Ref* 357
separator styles: *Editors* 223
tear–off: *Editors* 217
testing: *Editors* 34, 115
in menu bar editor: *Editors* 227
unique instances of: *Lang Ref* 360
unused in file: *Editors* 226
widget hierarchy in Motif: *Config* 183

Menu bar
See also Menu; Menu bar editor
assigning in screen wizard: *Editors* 79
displaying items on: *Editors* 219; *App Dev* 92
in application mode: *Editors* 4–6
reserving space for: *Config* 32
screen wizard prototype: *Editors* 86
submenu indicator: *Config* 132

Menu bar editor: *Editors* 213–228
features: *Editors* 7

Menu item
assigning control string to: *Editors* 218
attaching external help: *Editors* 221
attaching JAM help screen: *Editors* 220, 275
changing properties: *Lang Ref* 368
displaying on menu bar: *Editors* 219; *App Dev* 92
displaying on toolbar: *Editors* 219; *App Dev* 92

Menu item (continued)
displaying status of: *App Dev* 202
getting properties: *Lang Ref* 377
inactivating: *Editors* 220
including help: *Editors* 221
indicator symbol
reserving space for: *Editors* 219
setting for character mode: *Config* 131
inserting at runtime: *Lang Ref* 374
invoking submenu: *Editors* 222
keyboard mnemonic: *Editors* 217
naming: *Editors* 218
platform–specific types: *Editors* 222
properties of: *Editors* 217–221
property constants: *Lang Ref* 378
removing at runtime: *Lang Ref* 376
right justifying on menu bar: *Editors* 220
setting status in transaction style: *Editors* 288; *App Dev* 335–336
setup variables: *Config* 31
text: *Editors* 217
transaction classes for: *Editors* 287; *App Dev* 335–336
type constants: *Lang Ref* 375
types: *Editors* 221–223
action: *Editors* 221
edit: *Editors* 222
separator: *Editors* 221
submenu: *Editors* 222
toggle: *Editors* 221
windows list: *Editors* 222
windows operations: *Editors* 222

Menu list window: *Editors* 226–227

Menu Name property: *App Dev* 90
for menus: *Editors* 217
for screens: *Editors* 114

Menu property, for menu items: *Editors* 219

Menu runtime properties: *App Dev* 93

Menu script
loading into memory: *Lang Ref* 382; *App Dev* 88–89
unloading from memory: *Lang Ref* 384; *App Dev* 96

Menu Script File property: *Editors* 114; *App Dev* 88

Menu Title property: *Editors* 217

Message
See also Error messages; Message file; Status line
acknowledgment: *Lang Ref* 59, 61, 223, 225, 226; *Config* 60
forcing: *Config* 28, 59
acknowledgment key: *Lang Ref* 60, 224
bell: *Editors* 272; *Lang Ref* 60, 224; *Config* 59
creating: *Config* 44
default display
in status line: *Lang Ref* 59, 223, 226; *Config* 26
in window: *Lang Ref* 59, 223, 226; *Config* 26
display attributes in: *Lang Ref* 59, 224; *Config* 27, 57–60
hexadecimal codes for: *Config 58*
displaying
background status: *App Dev* 203
error tag: *Lang Ref* 58, 258, 259
forcing to window: *Lang Ref* 61, 225; *Config* 26
in dialog box: *Lang Ref* 363
in window: *Config* 60–69
on status line: *App Dev* 202
through JPL commands: *Lang Ref* 58
through library functions: *Lang Ref* 223, 226, 258, 259
error: *App Dev* 199–203
file. *See* Message file
forcing to status line: *Lang Ref* 61, 225
automatic dismissal: *Lang Ref* 60, 224; *Config* 59
functions: *App Dev 200–203*
installing: *Config* 48
JAM messages: *Config* 44
key labels in: *Lang Ref* 60, 224; *Config* 59
line. *See* Status line
line break insertion: *Lang Ref* 61, 225
multiple lines in: *Config* 60
Ready/Wait status, displaying: *Lang Ref* 442
removing from memory: *Lang Ref* 391
retrieving from message file: *Lang Ref* 389, 390
setup variables: *Config* 26–28
status, formStatus Motif resource: *Config* 175
status line. *See* Status line
text not visible: *Config* 27
transaction manager errors: *Lang Ref* 463, 464

Message dialog box: *Lang Ref* 363
button combinations: *Lang Ref* 364, 365
default button: *Lang Ref* 365, 366
modality setting: *Lang Ref* 365, 366
system icon: *Lang Ref* 365, 366
text format options: *Lang Ref* 363

Message file: *Config* 41–71
converting to binary (msg2bin): *Config* 49–52
creating: *Config* 48–49
disk–based: *App Dev* 524
identifying for initialization: *Config* 16
initialization: *Lang Ref* 391
internationalization
currency formats: *App Dev* 486–488
date/time formats: *App Dev* 483–486
JDB: *Database* 32
modifying: *Config* 47
multiple sections: *Config* 45
name of: *Config 2*
size: *Config* 45
syntax: *Config* 44–47
text: *Config* 44
translating: *App Dev* 482; *Config* 47
using alternate: *Config* 71
variable: *Config* 16

Message tag prefix, uses: *Config* 48

MESSAGE_WINDOW: *Config* 26

Millimeter, as unit of measurement: *Editors 41*

MIN function, in JDB: *Database* 59

Min Size property: *Editors* 43

Minimize option, on screens: *Editors* 101

Minimum Decimals property: *Editors* 247

Minimum Digits property, for check digit modulus: *Editors* 319

Minimum Horizontal Space property
for a box: *Editors* 263
for a screen: *Editors* 104

Minimum property, for graph widget axes: *Editors* 133

Minimum Value property
for scales: *Editors* 176
setting: *Editors* 171

Minimum Vertical Space property
for a box: *Editors* 264
for a screen: *Editors* 104

Minor Increment property, for graph widget tick marks: *Editors* 133

mksql, re–creating statements for database: *Database* 122

MNBR (menu bar key): *Config 79*

Mnemonic
accessing from data-entry widgets: *Editors* 300
assigning to widget: *Editors* 299–300
attaching to menu item: *Editors* 217
display attributes
in menu items: *Config* 31
in widgets: *Config* 31

Mnemonic Position property
for widgets: *Editors* 300
on push button label: *Editors* 193

MODE0 to MODE6 keyword: *App Dev* 474
in video file: *Config* 126–128
interpreting: *App Dev* 477

Model property
for screen: *Editors* 339
for table view: *Editors* 339

Modifying messages: *Config* 47

Money, currency format: *Editors* 248

Monochrome terminal: *Config* 58

More button: *Editors* 29

Motif
common color names: *Config* 173–174
setting defaults: *Config* 169–188

Motif resource file: *Config* 169–188
application behavior options: *Config* 174
background: *Config* 178
baseWindow: *Config* 174, 178
class name: *Config* 170
colors: *Config* 171–174
defining color scheme: *Editors* 254
defining pixmap location: *Editors* 267
focusAutoRaise: *Config* 177
foreground: *Config* 178
formStatus: *Config* 175
global resources: *Config* 177–188
introPixmap: *Config* 174
location: *Config* 170
names: *Config* 169–170
overriding colors: *Config* 172
ownColorMap: *Config* 178
restricted resources: *Config* 177
restricting resources to a screen: *Config* 180
sample, Pages: *Config* 184
syntax: *Config* 170
text alignment: *Editors* 235

MOUS key (indicate mouse event), hex value: *Config 79*

MOUS_CRSR_ATTR: *Config* 24

MOUS_CRSR_CHAR: *Config* 24

MOUS_CRSR_MASK: *Config* 25

Mouse
right–button behavior: *Editors* 301
supporting in JAM: *Config* 132

Mouse driver, specifying: *Config* 132

Mouse events
getting name of last clicked–on field: *App Dev* 556
getting name of last clicked–on screen: *App Dev* 556
getting state of buttons: *Lang Ref* 385; *App Dev* 557
getting system time for mouse click: *Lang Ref* 394

Mouse pointer
character JAM appearance and behavior: *Config* 24
custom shape creation: *Editors* 110–112
shape options for Motif: *Editors 109*
shape options for Windows: *Editors 109*
specifying: *Editors* 108–110

MOUSEDRIVER keyword (video file): *Config* 132

Mouseless interface: *Editors* 371–375

Moving widgets: *Editors* 44–45

MS–DOS, INIT keywords: *Config 110*

msg2bin: *Config* 49–52
errors: *Config* 50

msg2hdr: *Config* 53–56
errors: *Config* 54
options/arguments: *Config* 53
sample output: *Config* 53

MSGATT keyword (video file): *Config* 125
flags for: *Config* 125

MTGL key (toggle menu mode), hex value: *Config 79*

Multi-item properties
accessing in JPL: *Lang Ref* 29
getting at runtime: *Lang Ref* 415
setting at runtime: *Lang Ref* 420

Multi–user access: *Editors* 362–363

Multiline text widget: *Editors* 24
3D (in Windows): *Editors* 250
and autotab behavior: *Editors* 312
assigning double–click event to: *Editors* 305–306
specifying word wrap on: *Editors* 174

Multiple Create mode: *Editors* 16

Multiple sections, in message file: *Config* 45

Multiple Select mode: *Editors* 15

Multiple table joins: *Database* 82

Multiplication operation, in JDB: *Database* 90

MULTISHIFT flag (video file): *Config 110*

Must Fill property: *Editors* 173

N

Name property
- for link: *Editors* 344
- for menu items: *Editors* 218
- for selection group: *Editors* 209
- for style widget: *Editors* 288
- for table view: *Editors* 338
- specifying for widgets: *Editors* 299
- to ensure inheritance: *Editors* 54

Natural joins: *Database* 82

Navigating, in screen wizard: *Editors* 69

NEW, transaction manager command, entering new data: *App Dev* 440–442

NEXT, dbms command, executing next statement: *Database* 344, 403

Next Tab Stop property: *Editors* 307
- on dynamic labels: *Editors* 300

NL key (newline)
- acting like XMIT: *Config* 38
- hex value: *Config 79*

No auto tab, setting: *Config* 23

No border, for screens: *Editors* 111

No Validation property: *Editors* 304
- in styles editor: *Editors* 290

Non–locking shifts: *Config* 127

Nonprinting characters: *Config* 93

NOT keyword
- in joins: *Database* 81
- NOT BETWEEN, in JDB: *Database* 62, 105
- NOT EXISTS, in JDB: *Database* 100, 106
- NOT IN, in JDB: *Database* 100, 106

NOT keyword (continued)
- NOT LIKE, in JDB: *Database* 86, 107
- NOT NULL
  - in JDB: *Database* 88, 106
  - on screen wizard output: *Editors* 90

NULL, specifying in JDB: *Database* 88–89

Null edit
- colon–equal processing: *App Dev* 246–247
- on required data field: *Editors* 172
- results of: *Editors* 241
- with edit masks: *Editors* 165
- writing null value to database: *App Dev* 242, 248

Null Field property: *Editors* 240
- in automated SQL generation: *App Dev* 283
- writing null values to database: *App Dev* 242, 248

Null statement in JPL: *Lang Ref* 6

Null Text property: *Editors* 241

Null value
- and arithmetic operations in JDB: *Database* 90
- and COUNT aggregate function: *Database* 59
- default indicator: *Editors* 241
- defined: *Database* 13
- specifying in JDB: *Database* 88–89, 93
- writing to database: *App Dev* 242, 248

Number of rows, specifying for grid frame: *Editors* 186–187

Number of Selections property
- for radio button widget: *Editors* 204
- for toggle button widget: *Editors* 205
- setting for selection group: *Editors* 209

Numeric data: *Editors* 164
- range checking: *App Dev* 493
- reading from database: *App Dev* 235
- specifying for scale widgets: *Editors* 176
- specifying range: *Editors* 171
- writing to database: *App Dev* 245–246
  - for empty fields: *App Dev* 246

Numeric expression, JPL: *Lang Ref* 38

Numeric format: *Editors* 247–249
- defining custom format: *Editors* 247–249
- examples of: *Editors 247*
- including punctuation: *Editors* 247
- JPL: *Lang Ref* 38
- properties
  - getting application defaults: *Lang Ref* 406
  - setting application defaults: *Lang Ref* 421
- rounding: *Editors* 248
- zero format: *Editors* 248–251

Numeric format properties, setting application defaults: *Lang Ref* 421

O

OCCUR, dbms command, setting occurrence for SELECT: *Database* 164; *App Dev* 234

Occurrence
- deleting: *Lang Ref* 219; *App Dev* 85
- getting current number: *Lang Ref* 402
- group. *See* Group
- inserting: *Lang Ref* 291; *App Dev* 85
- referencing: *Editors* 299
- referencing in JPL: *Lang Ref* 26
- setting attributes: *Config* 35
- specifying in tab order: *Editors* 308

ODBC: *Database* 255–283
- connection options: *Database* 260
- DBMS command listing: *Database* 278
- description of: *Database* 255

OMSG keyword (video file): *Config* 124

ONENTRY, dbms command, calling function before dbms command: *Database* 165–166; *App Dev* 260–261

ONERROR, dbms command, installing error handler: *Database* 167–169; *App Dev* 260–261

ONEXIT, dbms command, calling function after dbms command: *Database* 170–171; *App Dev* 260–261

Onscreen attributes
- defined: *Config* 115
- setting: *Config* 116

Onscreen Columns property: *Editors* 184

ONSCREEN flag (video file): *Config* 116

Onscreen Rows property: *Editors* 242, 312
- for list box widget: *Editors* 202
- in a grid frame: *Editors* 187

Operands, conversion in JPL: *Lang Ref* 35

Operating system
- accessing from control string: *Editors* 327; *App Dev* 113
- accessing from within JAM: *Editors* 5
- date/time
  - displaying: *Editors* 245
  - getting: *Lang Ref* 436
- escaping from application: *Lang Ref* 342
- executing command
  - from control string: *Editors 322*
  - from JISQL: *Database* 35–36
  - from JPL: *Lang Ref* 86
  - through library function: *Lang Ref* 445
- print command specification: *Lang Ref* 452
- returning to JAM application: *Lang Ref* 430

Operator property: *Editors* 355
- in automated SQL generation: *App Dev* 281–283

Operators
- in JDB: *Database* 90
- JPL. *See* JPL operators
- supported in WHERE clause: *App Dev* 281

Optimistic locking, property settings: *Editors* 357; *App Dev* 370–371

Option menu widget: *Editors* 25, 176–180
- 3D (in Windows): *Editors* 250
- and autotab behavior: *Editors* 312
- controlling size of: *Editors* 178
- populating: *Editors* 177
- scrolling: *Editors* 178
- specifying initial text: *Editors* 180
- updating contents: *Lang Ref* 495

Oracle: *Database* 285–317
- connection options: *Database* 289
  - for XA library: *Database* 289
- DBMS command listing: *Database* 315
- executing stored subprograms: *Database* 297–300

ORDER BY clause
- changing generated SQL: *Lang Ref* 139–140
- in automated SQL generation: *App Dev* 288–289
- in JDB: *Database* 94–95

Order property, for toolbar items: *Editors* 219

Ordering, grid columns: *Editors* 185

Orientation property, for graph widget: *Editors* 129–130

Other Style property: *Editors* 260

Output commands: *Config* 97
- specifying: *Config 95*

Output label widget. *See* Dynamic label widget

Output processing: *App Dev* 476–477
  messages: *App Dev* 202

P

Packed decimal: *Editors* 301

Padding commands: *Config* 100
  in termcap: *Config* 101
  in terminfo: *Config* 101

Parameters
  declaring in JPL: *Lang Ref* 4
    named procedure: *Lang Ref* 66
    unnamed procedure: *Lang Ref* 64
  for binding, in DECLARE CURSOR command: *App Dev* 219, 250–255
  in video file. *See* Video file
  name requirements: *Lang Ref* 64

Parent First specification: *Editors* 346

Parent object
  finding: *Editors* 65–66
  finding children of: *Editors* 65
  turning inheritance on/off for specific properties: *Editors* 64

Parent property: *Editors* 345
  determining parent table view: *App Dev* 314, 358–359

Password Char property: *Editors* 234

Password Field property: *Editors* 234

Path: *Lang Ref* 452; *Config* 18

PC extended keyboard: *Config* 84

Percent commands
  arithmetic: *Config 96*, 98
  for changing parameters: *Config* 99
  for control flow: *Config 96*, 99
  for parameter sequencing: *Config 96*, 98
  for specifying output: *Config 95*, 97
  for stack manipulation: *Config 96*, 98
  for terminal delays: *Config* 100
  in video file: *Config* 94–102

Percent escapes
  in JPL message commands: *Lang Ref* 59
  in message file: *Config 57*
  in message functions: *Lang Ref* 224

Percent Location property, for graph widget, pie chart: *Editors* 147

Pessimistic locking: *Editors* 357

PF1–PF24 (function keys). *See* Function keys

Pie chart: *Editors* 143–150
  creating: *Editors* 144–145
  data source: *Editors* 145
  displaying in 3D: *Editors* 145
  formatting: *Editors* 145–146
  legend: *Editors* 126, 127
  segments: *Editors* 146–150

Pixel, as unit of measurement: *Editors 41*

Pixmap
  creating: *Editors* 194
  displaying on toolbar items: *Editors* 224
  displaying on widgets: *Editors* 266–268
  location on system: *Editors* 267
    Motif: *Editors* 225
    Windows: *Editors* 225
  on push button widget: *Editors* 194
  sizing
    on toolbar items: *Editors* 225
    on widgets: *Editors* 268

Placement property: *Editors* 248
  for graph widget legend: *Editors* 126

PLAY_FUNC. *See* Playback function

Playback function: *App Dev* 141–142
  example: *App Dev* 176
  return codes: *App Dev* 141
  standard argument: *App Dev* 141
  turn on or off: *Lang Ref* 312

Point Marker property, for graph widget data series: *Editors* 141–142

Point Size property: *Editors* 238

Pointer property: *Editors* 109

Pop–up menu, invoking: *App Dev* 96
  through function call: *Lang Ref* 411; *App Dev* 96

Popup Menu property
  for screens: *Editors* 114
  for widgets: *Editors* 301 *App Dev* 90

Popup Script File property: *Editors* 301

Portability: *App Dev* 479–480
  aliasing colors: *Config* 138
  font name: *Editors* 125, 236
  of JAM applications: *Get Started* 2–3
  smmach.h: *App Dev* 480
  terminal: *App Dev* 476

Position Region property: *Editors* 262

Positioning properties, for box widgets: *Editors* 263–265

Positioning widgets: *Editors* 46–49

Precision
  in SELECT results: *App Dev* 235
  JPL: *Lang Ref* 38

Precision property
  defined in database. *See* C Type property
  for C type specification: *Editors* 301

Preferences
  saving: *Editors* 15
  setting: *Editors* 15–16

Prefixes, in message file: *Config* 44

PREPARE_COMMIT, dbms command, preparing two phase commit: *Database* 404

Presentation Manager, mapping to string resource IDs: *Lang Ref* 409

Preview button: *Editors* 69

Previous Tab Stop property: *Editors* 307

Primary keys
  defined: *Database* 11–12, 15
  defining using JISQL: *Database* 39–41
  specifying in screen wizard: *Editors* 72–73

Primary Keys property
  for table view: *Editors* 352
  in automated SQL generation: *App Dev* 297–298, 300

Print file, setting system command: *Config* 35

Procedure, declaring in JPL: *Lang Ref* 66

Properties
  See also Application runtime properties; Array runtime properties; Screen runtime properties; specific property names; Widget runtime properties
  accessing in JPL: *Lang Ref* 28
    application properties: *Lang Ref* 28
    editor properties: *Lang Ref* 28
    multi-item properties: *Lang Ref* 29
    runtime properties: *Lang Ref* 28
    substring of setting: *Lang Ref* 29
  error handling: *Lang Ref* 412
  getting at runtime: *Lang Ref* 413
    for array element: *Lang Ref* 414
    for array occurrences: *Lang Ref* 414
  getting handle to object: *Lang Ref* 416
  getting multi-item settings: *Lang Ref* 415
  inheritance. *See* Inheritance
  screen property headings defined: *Editors* 33
  setting: *Editors* 27–33
  setting at runtime: *Lang Ref* 418
    for array element: *Lang Ref* 419
    for array occurrences: *Lang Ref* 419
  setting item: *Lang Ref* 420
  transaction manager: *App Dev* 380–384
  traversal properties: *App Dev* 380–384
  value types: *Lang Ref* 29
  widget property headings defined: *Editors* 31–33

Properties list, expanding/collapsing: *Editors* 29

Properties window
  in menu bar editor: *Editors* 215
  in screen editor: *Editors* 27–33
  in styles editor: *Editors* 282

Protected modes: *Config* 123

Protected widgets: *Editors* 117–119

Protection: *Editors* 302–305
  and tabbing order: *Editors* 303
  focus: *Editors* 302
  from clearing: *Editors* 304
  from clearing data: *Editors* 174
  from entering data: *Editors* 173
  scrolling field: *Editors* 304
  shifting field: *Editors* 304
  tabbing into: *Editors* 302
  validation: *Editors* 303

PROTO_FUNC. *See* Prototyped function

Prototyped function: *App Dev* 120–122
examples: *App Dev* 148
get standard arguments: *App Dev* 120
valid prototypes: *App Dev 122*

Public module
loading: *Lang Ref* 68, 306
unloading: *Lang Ref* 75, 307

Punctuation, in numeric fields: *Editors* 247

Push button widget: *Editors* 25, 191–197
adding to screen wizard output: *Editors* 87
assigning mnemonic: *Editors* 193
attaching action to: *Editors* 196–197
cancel: *Editors* 194
deactivating: *Editors* 195
default: *Editors* 194–195
displaying images on: *Editors* 193–194
executing control string: *Editors* 323
in repository entry: *Editors* 192
label text alignment: *Editors* 192, 235
labelling: *Editors* 192–193
setting initial activation status: *Editors* 195
setting status in transaction style: *Editors* 288; *App Dev* 335–336
specifying in screen wizard: *Editors* 79
transaction classes for: *App Dev* 335–336
transaction classes for : *Editors* 287

PVCS support: *Editors* 361

Q

Queries, database: *Database* 96–99

QUIETATT: *Config* 27

quit, exiting ISQL: *Database* 120

Quotation marks, inputting: *Config* 92

R

Radio button widget: *Editors* 26, 204–206
See also Group
3D (in Windows): *Editors* 250
displaying image on: *Editors* 266

Radix separator. *See* Decimal symbol

Range
checking: *App Dev* 493–494
in regular expression: *Editors* 166
search conditions in JDB: *Database* 62–63
specifying: *Editors* 171–172

RARR key (right arrow), hex value: *Config 79*

RCP keyword (video file): *Config* 115

Re–importing database tables: *Editors* 61–62

read, executing file in ISQL: *Database* 120

Read–only screen
opening: *Editors* 17
saving: *Editors* 367

Ready/Wait status, displaying: *Lang Ref* 442; *App Dev* 202

receive command, emulating through sm_receive: *Lang Ref* 424

Record function: *App Dev* 141–142
example: *App Dev* 176
return codes: *App Dev* 141
standard argument: *App Dev* 141
turning on or off: *Lang Ref* 312

RECORD_FUNC. *See* Record function

Records, database. *See* Rows

Redo command: *Editors* 23

Redoing, actions: *Editors* 51

REFR key (refresh screen), hex value: *Config 79*

REFRESH, transaction manager command, refreshing the screen: *App Dev* 443

Region Margin property
for a box: *Editors* 264
for a screen: *Editors* 104

Regular expression: *Editors* 166–171; *App Dev* 489
character classes: *Editors* 168–169
character–level: *Editors* 166
concatenating: *Editors* 169–170
constructing: *Editors* 167–171
examples of: *Editors* 167
field–level: *Editors* 167
repeating: *Editors* 170
simple: *Editors* 167–168
special characters: *Editors 168*

Regular Expression property: *Editors* 167

Relational databases: *Database* 9

Relations property: *Editors* 345; *App Dev* 360–361
  in automated SQL generation: *App Dev* 289, 291, 292

Repeat character sequence
  See also REPT keyword
  setting: *Config* 111

Repeating property: *Editors* 241

REPMAX keyword (video file): *Config* 111

Reposition widgets at runtime: *Lang Ref* 156

Repository: *App Dev* 61–68
  and the screen wizard: *Editors* 67, 90
  benefits: *Editors* 53
  creating: *Editors* 55–56
  defined: *Editors* 53
  getting name: *Lang Ref* 406
  importing database objects: *App Dev* 63–64, 310–312, 353–354
  listing contents: *Editors* 56
  lock on: *Editors* 366
  naming conventions: *Editors* 20
  opening: *Editors* 56
    screen in: *Editors* 17–18, 56–57
  propagating changes: *Editors* 54
  saving screen to: *Editors* 58–59
  screen wizard entries: *App Dev* 65
  setting default pathname: *Config* 17
  storing screen templates: *App Dev* 63
  synchronize with source code management: *Editors* 367
  table of contents (TOC): *Editors* 56
  under source code management: *Editors* 363–364

Repository entry
  and widget names: *Editors* 299
  copying from: *Editors* 62; *App Dev* 311–312
  creating: *Editors* 57–58, 98
  creating from database: *Editors* 59–62
  deleting: *Editors* 57
  in use: *Editors* 18
  naming conventions: *Editors* 20
  naming widgets in: *Editors* 54
  opening: *Editors* 17, 56
    under source code management: *Editors* 365
  saving: *Editors* 57
  uses for: *Editors* 63

Repository screen. *See* Repository entry

Repository TOC: *Editors* 14, 56, 98
  displaying comments: *Editors* 115

REPT keyword (video file): *Config* 111
  parameters: *Config 94*

Required property: *Editors* 172
  in styles editor: *Editors* 290

Reservation
  prompt for release of: *Editors* 16
  releasing: *Editors* 363

Reserving screen: *Editors* 362–363

Reset function: *App Dev* 140–141
  example: *App Dev* 175
  return codes: *App Dev* 140
  standard argument: *App Dev* 140

RESET keyword (video file): *Config* 111

Resize Function property: *Editors* 43, 100

Resizeable property: *Editors* 43, 100

Resizing widgets: *Editors* 40–44

Resource file. *See* Motif resource file

Resources, mapping string to integer IDs: *Lang Ref* 409

Restoring inheritance: *Editors* 64

Restrictions, JDB: *Database* 7

Return codes
  stored procedures: *Database* 189, 376
  transaction hook functions: *App Dev* 385

Return value: *Lang Ref* 15, 72
  declaring type in JPL: *Lang Ref* 5, 66

Reverse display attribute, setting: *Editors* 258

Revert menu option: *Editors* 13

REWRITE flag (video file): *Config* 116

rgb.txt: *Config* 173

Right justified text: *Editors* 235

ROLLBACK, dbms command
  rolling back transactions: *Database* 229, 250, 277, 312, 345, 405; *App Dev* 266–270

rollback
  transaction in ISQL: *Database* 120
  transaction in JDB: *Database* 115
  transaction in JISQL: *Database* 50

Root property: *Editors* 342

Root table view
setting: *App Dev* 356
specifying: *Editors* 342

Rounding property: *Editors* 248

Routing. *See* Key, routing

Row. *See* Grid row

Row Entry Func property: *Editors* 190
for grid: *Editors* 332
set by screen wizard: *Editors* 87

Row Exit Func property: *Editors* 190
for grid: *Editors* 332
set by screen wizard: *Editors* 87

Row Separators property: *Editors* 188

Row Titles property: *Editors* 187

Rows
defined: *Database* 11
fetching: *Database* 149–150, 151–157
no more rows status: *App Dev* 230
number fetched: *Database* 196–197; *App Dev* 228
retrieving multiple rows: *App Dev* 228
scrolling through result set: *App Dev* 232–233
value of @dmrowcount in DBMS START: *Database* 173

RSHF key (shift right), hex value: *Config 79*

Rubberbanding: *Editors* 38

Runtime (JAM) message tag: *Config* 44

Runtime path, setting: *Config* 18

Runtime properties
See also Properties
accessing in JPL: *Lang Ref* 28
getting: *Lang Ref* 413
setting: *Lang Ref* 418

RWRD key (next word), hex value: *Config 79*

S

s2asc: *Editors* 292

Sample application: *App Dev* 33–57

SAVE
dbms command, setting a savepoint: *Database* 313, 407
transaction manager command, saving database changes: *App Dev* 391–393, 444–449

Save Pref menu option: *Editors* 12

SB_OPTIONS: *Config* 30

Scale property
defined in database: *Editors* 353
for graph widget axes: *Editors* 133

Scale widget: *Editors* 25, 176
setting initial value: *Editors* 176

Scan line: *Config* 110

SCCS support: *Editors* 361

Scheme
defined: *Editors* 254
defining in color map file: *Config* 142
object names for color mapping: *Config* 143
specifying on Color palette: *Editors* 257
using Motif resource file: *Editors* 254
using Windows control panel: *Editors* 254

Scope. *See* Menu

SCP keyword (video file): *Config* 115

SCR_KEY_OPT: *Config* 29

Screen
See also Screen editor; Window
3D (in Windows): *Editors* 249–251
about: *App Dev* 69–75
ASCII/binary conversion: *App Dev* 565–567
border: *Editors* 110–111
C data structure conversion: *App Dev* 563–564
calls to JPL from: *Lang Ref* 15
change window through keyboard: *Lang Ref* 87
character–mode attributes: *Config* 33
clock update and: *Editors* 245
closing: *Editors* 21; *Lang Ref* 82, 84, 187, 301; *App Dev* 75
control string: *App Dev* 109
executing: *Editors* 323
creating: *Editors* 17, 96–99
as library member: *Editors* 97
as repository entry: *Editors* 58, 98
screen templates: *App Dev* 63
decoration: *Editors* 26–27
display defaults: *App Dev* 73
overriding: *App Dev* 74–75

Screen (continued)
documenting: *Editors* 115–116
entry function. *See* Screen function
entry/exit functions: *Editors* 334
exit function. *See* Screen function
file extension: *Lang Ref* 452
setting defaults: *Config* 34
focus, in Motif: *Config* 177
font property: *Editors* 237
forcing validation: *Lang Ref* 433
forcing write to LDB: *Lang Ref* 344
freeing saved data: *Lang Ref* 455
function. *See* Screen function
getting name of current: *Lang Ref* 407
hexadecimal conversion: *App Dev* 564–565
HWND handle: *Lang Ref* 220
iconifying: *Editors* 101–103
inheritance. *See* Inheritance
inheritance source, specifying: *Editors* 65
internationalization. *See* Internationalization
JPL: *Editors* 328
JPL module: *Lang Ref* 9
library. *See* Screen library
location: *Editors* 103
manipulating without mouse: *Editors* 372
maximizing/minimizing: *Editors* 101–103
memory–resident: *App Dev* 522
adding to list: *Lang Ref* 254
enabling installation: *App Dev* 522
menu
attaching: *App Dev* 89
attaching as menu bar: *Editors* 113
modal: *Editors* 99
mouse cursor, specifying shape: *Editors* 108–110
naming: *Editors* 111–112
naming conventions: *Editors* 19
navigating without mouse: *Editors* 371–373
opening: *App Dev* 73–74
as a form: *Editors* 323; *Lang Ref* 252, 303; *App Dev* 70, 73
as a sibling window: *Editors* 323; *App Dev* 71
as a window: *Editors* 323; *Lang Ref* 309, 501; *App Dev* 70, 73
at specific size/dimension: *Editors* 324; *App Dev* 74–75
from control string: *Editors* 323–324; *App Dev* 73, 110–112
through dialog box: *Lang Ref* 84
through library functions: *App Dev* 73
with size/dimension arguments: *App Dev* 111

Screen (continued)
opening in editor: *Editors* 17
as a template: *Editors* 17
read only: *Editors* 17
repository entry: *Editors* 17–18
stored in library: *Editors* 18
under source code management: *Editors* 365
ownership information: *Editors* 366
properties. *See* Screen properties
read–only: *Editors* 366
saving: *Editors* 367
referencing in JPL
by name: *Lang Ref* 25
by number: *Lang Ref* 26
refreshing: *Lang Ref* 156
removing from save list: *Lang Ref* 494
reserving: *Editors* 362
resources, Motif: *Config* 180
restoring saved data: *Lang Ref* 429, 432
root table view, specifying: *Editors* 342
saving: *Editors* 19–20
in memory: *Lang Ref* 456
to a repository: *Editors* 58–59
to library: *Editors* 20–21
saving data: *Lang Ref* 435, 454
search path for opening: *Lang Ref* 253
selecting: *Editors* 30
setting name of current: *Lang Ref* 421
setting number of lines: *Config* 111
shrinking: *Lang Ref* 446
sizing: *Editors* 99
maintaining: *Editors* 100
startup: *Config* 37
template, using repository entry as: *Editors* 18
testing: *Editors* 34–35
title bar: *Editors* 112–113
translating coordinates to pixels: *Lang Ref* 489
trim: *Lang Ref* 504–517
validation. *See* Validation
viewport: *Editors* 100; *App Dev* 74, 111
virtual: *Editors* 100
widget hierarchy, Motif: *Config* 179
widget ID: *Lang Ref* 220

Screen data transfer. *See* Send data

Screen editor
display options: *Editors* 14
exiting: *Editors* 21
grid options: *Editors* 15
menu bar description: *Editors* 11–15
options: *Editors* 15–16

Screen editor (continued)
setting defaults: *Lang Ref* 452
setting editor colors with cmap file: *Config* 140
starting: *Editors* 9–11
toolbar: *Editors* 10

Screen editor message tag: *Config* 44

Screen format, specifying in screen wizard: *Editors* 70

Screen function: *App Dev* 122–124
attaching entry/exit function: *Editors* 334
data access, LDB vs. fields: *Config* 37
example of automatic function: *App Dev* 156
execution options: *Config* 37
return codes: *App Dev* 124
standard arguments: *App Dev* 122

Screen library, identifying for initialization: *Config* 17

Screen module: *Lang Ref* 9

Screen properties: *App Dev* 75
accessing: *Editors* 30
set by the screen wizard: *Editors* 82

Screen runtime properties: *Lang Ref* 521; *App Dev* 75
getting: *Lang Ref* 413
getting handle to: *Lang Ref* 416
setting: *Lang Ref* 418

Screen save list, check for screen: *Lang Ref* 299

Screen title, specifying in screen wizard: *Editors* 79

Screen wizard
and NOT NULL database columns: *Editors* 90
generated JPL, description of: *Editors* 86
layout selection: *Editors* 78
output description: *Editors* 81
prototype screens: *Editors* 68–93
and the repository: *Editors* 69
starting: *Editors* 67
troubleshooting: *Editors* 90–93

SCREEN_FUNC. *See* Screen function

SCREENWRAP flag (video file): *Config* 116

Script. *See* Menu script

Scroll bars
on list box widget: *Editors* 202
slider characters: *Config* 131

Scroll Increment property: *Editors* 243
for grid frames: *Editors* 187

Scroll indicators, setting position of: *Config* 30

SCROLL_FUNC. *See* Scrolling array, alternative scroll driver

Scrolling
in the database drivers: *Database* 213–214, 243, 266, 294, 331, 365
specifying backward scrolling: *Database* 153–156; *App Dev* 232–233
specifying continuation file: *Database* 174–176
specifying engine scrolling: *Database* 227, 230, 392, 412

Scrolling array
alternative scroll driver: *App Dev* 529–543
action codes: *App Dev* 532
array size: *App Dev* 530
delete lines: *App Dev* 542
enabling: *App Dev* 522
get data: *App Dev* 540
initialize: *App Dev* 538
insert blank lines: *App Dev* 542
installing: *App Dev* 531, 536
put data: *App Dev* 541
release: *App Dev* 539
reserve space: *App Dev* 543
reset: *App Dev* 537
specifying: *Editors* 316
struct parameter: *App Dev* 533
circular: *Editors* 243
creating: *Editors* 242
getting next synchronized array: *Lang Ref* 396
indicator
in video file: *Config* 130
placement: *Config* 30
setting: *Config* 29
input protection and: *Editors* 304
isolating: *Editors* 315
occurrence. *See* Occurrence
scroll increment specification: *Editors* 243
setup options: *Config* 29–30
synchronizing: *Editors* 312–316
tab order: *Editors* 310
viewing database information: *App Dev* 230

Scrolling Function property: *Editors* 316

Scrolling property
for arrays: *Editors* 243
for list box widget: *Editors* 202

Search conditions, in SQL statements: *Database* 105

Search path, screen: *Lang Ref* 253

SELECT, transaction manager command, fetching data for update: *App Dev* 450–454

Select on Entry property: *Editors* 173

SELECT statement
aliasing columns to widgets: *App Dev* 225–226
automatic mapping of column names: *App Dev* 225
changing generated SQL: *Lang Ref* 131–146
concatenating result row: *App Dev* 237
constructing: *Database* 18–19
destination of: *App Dev* 224, 237
aggregate functions: *App Dev* 227
fetching binary columns: *Database* 138–139
format of results: *App Dev* 234–237
formatting result set: *Database* 162–163
free memory for statement: *Lang Ref* 126
generating SQL: *Lang Ref* 147
in INSERT statement: *Database* 80
in JDB: *Database* 96–99
NULL values and: *Database* 88
number of rows fetched: *Database* 196–197
no more rows status: *Database* 196–197; *App Dev* 230
scrolling through result set: *Database* 149–156, 174–176; *App Dev* 228
setting starting row: *Database* 173
specifying multiple tables, in automated SQL generation: *App Dev* 289–292
SQL generation from properties: *App Dev* 277–292, 302
suppressing repeating values: *Database* 177; *App Dev* 236–237
transaction manager, writing hook function: *App Dev* 388–393
unique column values: *App Dev* 236–237
writing results
to a file: *Database* 140–142; *App Dev* 237
to a specific occurrence: *Database* 164; *App Dev* 229, 234
to word-wrapped arrays: *App Dev* 229

Selecting widgets: *Editors* 37–39
grid frame: *Editors* 182
in character JAM: *Editors* 38
in grid frame: *Editors* 183
rubberbanding: *Editors* 38
using Widget list: *Editors* 38

Selection group
deselecting: *Lang Ref* 214; *App Dev* 83
getting selection data: *Lang Ref* 31; *App Dev* 82
propagating data through LDB: *App Dev* 192
runtime properties: *Lang Ref* 546
selecting: *Lang Ref* 438; *App Dev* 83
testing for selection: *App Dev* 82

Selection screen: *Editors* 276–278
attaching: *Editors* 278
creating: *Editors* 277–278
displaying on entry: *Editors* 278
enhancing screen wizard output: *Editors* 89
properties set by the screen wizard: *Editors* 83–84
specifying in screen wizard: *Editors* 79
using with table lookup: *Editors* 278

Selection Screen property: *Editors* 278
setting via the screen wizard: *Editors* 81

Selection widgets: *Editors* 25
grouping: *Editors* 206–211
and autotab behavior: *Editors* 312
specifying number of selections: *Editors* 209

Self–joins: *Database* 84
and the screen wizard: *Editors* 93
in automated SQL generation: *App Dev* 290

Send data: *App Dev* 195–198
appending to bundle: *Lang Ref* 160
counting bundle item occurrences: *Lang Ref* 268
counting bundle items: *Lang Ref* 267
creating bundle item: *Lang Ref* 163
destroying bundle: *Lang Ref* 260
emulating send command in C: *Lang Ref* 439
getting bundle status: *App Dev* 197
getting previous bundle name: *Lang Ref* 269
initializing bundle: *Lang Ref* 191
optimizing bundle storage: *Lang Ref* 162
reading bundle data: *App Dev* 197–198
through JPL: *App Dev* 197
through library functions: *App Dev* 197
reading bundle data through JPL: *Lang Ref* 70
reading bundle data through sm_receive: *Lang Ref* 424
reading occurrence from bundle: *Lang Ref* 266
verifying bundle name: *Lang Ref* 293
writing data to bundle: *Lang Ref* 439; *App Dev* 196–197
through JPL: *Lang Ref* 73; *App Dev* 196
through library functions: *App Dev* 196

Separator menu items: *Editors* 221
styles: *Editors* 223

Separator property, for menu item: *Editors* 220

Sequential link type: *Editors* 345; *App Dev* 314–315, 359–360
  in automated SQL generation: *App Dev* 291
  join specification: *Editors* 346

Serial column, @dmserial: *Database* 198

Server link type: *Editors* 345; *App Dev* 314–315, 359–360
  in automated SQL generation: *App Dev* 289–290
  join relationship: *Editors* 347
  synchronized scrolling: *Editors* 312

Server views: *App Dev* 315–316
  traversal properties: *App Dev* 381–384

SET, dbms command
  set cursor handling in SYBASE: *Database* 346, 409
  setting Client Library cursor: *Database* 349

Set area graphics. *See* ASGR keyword

SET clause, in automated SQL generation: *App Dev* 297

Set graphics rendition. *See* SGR keyword

SET HOLD, dbms command, setting cursor behavior in Informix: *Database* 232

SET HOLD_DEFAULT, dbms command, setting cursor behavior in Informix: *Database* 233

Set inherit warnings: *Editors* 15, 64

Set latch graphics. *See* SGR keyword

Set Valid property: *Editors* 354

SET_BUFFER, dbms command, specifying engine scrolling: *Database* 230, 412

Setting field, in properties window: *Editors* 28

Setting inheritance for all properties: *Editors* 65

Setup file
  converting to binary (var2bin): *Config* 10–11
  creating: *Config* 10
  modifying: *Config* 10
  name of: *Config 2*
  sample: *Config* 39
  specifying: *Config* 14, 16
  syntax: *Config* 9
  syntax of: *Config* 20
  types of: *Config* 6–7

Setup variables
  changing: *Config* 20
  changing at runtime: *Lang Ref* 405
  defined: *Config* 6
  defining: *Config* 20–21
  delayed–write: *Config* 37
  designating active screens: *Config* 33
  display attribute keywords: *Config* 21
  for character–mode screens: *Config* 33
  for controlling behavior: *Config* 21–38
  for cursor appearance: *Config* 22–38
  for filenames: *Config* 33–34
  for label text display: *Config* 30
  for menus: *Config* 31
  for message display: *Config* 26–28
  for screen entry/exit processing: *Config* 37
  for scrolling: *Config* 29–30
  for shifting: *Config* 29–30
  group attributes: *Config* 34–35
  zoom: *Config* 29–30

Seven–bit character set: *Config* 93

SFTS key (shift scope), hex value: *Config 79*

SGR keyword (video file): *Config* 120, 123–124
  parameters: *Config* 123

Shift Increment property: *Editors* 244

Shift indicators, setting position of: *Config* 30

Shifted function keys (SPF1–SPF24), hex value: *Config 81*

Shifting array. *See* Shifting field

Shifting field: *Editors* 243–244
  and input protection: *Editors* 304
  horizontal scroll bar: *Editors* 244
  setting indicator: *Config* 29
  setting indicator placement: *Config* 30
  setup options: *Config* 29
  shift increment specification: *Editors* 244
  specifying indicators for: *Config* 130

Sibling window
  See also Window
  assigning status to existing window: *App Dev* 72
  changing focus: *Lang Ref* 506
  changing stacked window to: *App Dev* 72
  setting for next window: *App Dev* 72
  setting for next-opened window: *Lang Ref* 441

Sign property, for decimal specification: *Editors* 301

Single line text widget: *Editors* 24
  and autotab behavior: *Editors* 312
  assigning double–click event to: *Editors* 305–306

Single–row layout, description: *Editors* 82

Size to Contents property: *Editors* 41

Sizing widgets: *Editors* 40–44
  to fit content: *Editors* 41

Slider characters, for scroll bars in character JAM: *Config* 131

SM, message tag prefix: *Config* 44

SM_CALC_DATE, setting default format: *Config* 66

SM_DECIMAL, setting default: *Config* 69

SM_NO
  getting value: *Lang Ref* 406
  setting value: *Lang Ref* 421

SM_YES
  getting value: *Lang Ref* 406
  setting value: *Lang Ref* 421

SMDICNAME, defined: *Config* 17

SMEDITOR
  defined: *Config* 17
  editing SQL statements : *Database* 32

smerror.h, contents of: *Config* 42

SMFEXTENSION: *Config* 34

SMFLIBS, defined: *Config* 17

SMINICTRL, defined: *Config* 25

SMKEY: *Config* 83
  defined: *Config* 15

smkeys.h, contents of: *Config* 74, 78–82

SMLDBNAME, defined: *Config* 18

SMLPRINT, defined: *Config* 35

smmach.h: *App Dev* 480

SMMSGS
  defined: *Config* 16
  setting alternate value: *Config* 65, 71
  setting value of: *Config* 42

SMPATH, defined: *Config* 18

SMSETUP: *Config* 7
  defined: *Config* 16

SMSGBKATT: *Config* 27

SMSGPOS: *Config* 26

SMTERM
  defined: *Config* 14
  in Windows: *Config* 158

SMUSER: *Config* 158

SMVARS: *Config* 6
  defined: *Config* 14

SMVIDEO, defined: *Config* 16

SMVIEWER: *Config* 18

smwizard screen: *Editors* 85

smwizard.jpl: *Editors* 86

smwizis screen: *Editors* 85–86

smwzmenu: *Editors* 86

Snap to grid: *Editors* 16, 47
  using: *Editors* 48

SOME keyword, in JDB: *Database* 101

Sort Widgets property
  for table view: *Editors* 352
  in automated SQL generation: *App Dev* 288–289

Source code
  main routines
    jmain.c: *Lang Ref* 91
    jxmain.c: *Lang Ref* 91
    modifying: *App Dev* 521–527
  platform–dependent: *App Dev* 480
  stub functions: *App Dev* 525

Source code management: *Editors* 361–368
  accessing screen outside of: *Editors* 362
  features of: *Editors* 364
  setting up: *Editors* 364

Source control. *See* Source code management

Source Mgmt: *Editors* 365
  See also Source code management

Space bar: *Config* 80

Space command: *Editors* 48–49

Spacing property, for vertical arrays: *Editors* 242

Spacing widgets: *Editors* 48–49
  custom: *Editors* 48
  insufficient space: *Editors* 49

SPF1–SPF24 (shifted function keys). *See* Shifted function keys (SPF1–SPF24)

SPGD key (scroll down page), hex value: *Config 79*

SPGU key (scroll up page), hex value: *Config 79*

Splash screen
  Motif: *Config* 174
  Windows: *Config* 157

SPXATT keyword (video file): *Config* 124

SQL: *Database* 17–25
  automated: *App Dev* 273–304
    SELECT statement: *Lang Ref* 147
    setting properties for: *Editors* 351–359
  changing generated SQL: *Lang Ref* 131–146
  commands, in JDB: *Database* 57–109
  dbms command, executing SQL statements: *Database* 172
  executing in JISQL: *Database* 24
  executing in JPL: *Database* 24
  executing SQL statements: *Database* 172
    on named cursor: *Database* 160–161
  generation: *Lang Ref* 121–123
  last executed command: *Lang Ref* 148
  modifying automated SQL: *App Dev* 304, 390–391
  re-creating JDB database: *Database* 122
  syntax summary for JDB: *Database* 109
  viewing generated statements: *App Dev* 301

SQL generation
  appending to the SELECT statement: *Lang Ref* 141–142
  changing bind values in DBMS EXECUTE: *Lang Ref* 127–130
  changing the FROM clause: *Lang Ref* 131
  changing the GROUP BY clause: *Lang Ref* 133–134
  changing the HAVING clause: *Lang Ref* 135–136
  changing the ORDER BY clause: *Lang Ref* 139
  changing the select list: *Lang Ref* 137–138
  changing the WHERE clause: *Lang Ref* 143–145
  getting the correlation name: *Lang Ref* 146

SQL generator: *App Dev* 273–304
  modifying automated SQL: *App Dev* 304

SQL statements, declaring cursors for: *App Dev* 219–222

Stack manipulation commands: *Config 96*, 98

Stacked window. *See* Window

Standard arguments: *App Dev* 117
  check digit function: *App Dev* 139
  control function: *App Dev* 142
  error function: *App Dev* 137
  field function: *App Dev* 126
  grid function: *App Dev* 129
  group function: *App Dev* 132
  help function: *App Dev* 134
  initialization function: *App Dev* 140
  insert toggle function: *App Dev* 138
  key change function: *App Dev* 136
  passing into JPL procedure: *Lang Ref* 5
  passing into unnamed procedure: *Lang Ref* 5
  playback function: *App Dev* 141
  prototyped function, getting for: *App Dev* 120
  record function: *App Dev* 141
  reset function: *App Dev* 140
  screen function: *App Dev* 122
  timeout function: *App Dev* 135
  types: *Lang Ref* 15
  video processing function: *App Dev* 144

START
  dbms command, setting starting row: *Database* 173; *App Dev* 234
  transaction manager command, initiating transaction: *App Dev* 455–457

Start Angle property, for graph widget, pie chart: *Editors* 146

Start Column property
  for grid position: *Editors* 47
  specifying widget position: *Editors* 47

Start Row property
  for grid position: *Editors* 47
  specifying widget position: *Editors* 47

STARTSCREEN: *Config* 37

Startup property: *Editors* 101

Startup state, for screens: *Editors* 101

STAT_FUNC. *See* Status line function

Static label widget: *Editors* 23, 117–118
  displaying active pixmap on: *Editors* 266
  resizing: *Editors* 118

Status line
  bell: *Editors* 272
  closing: *Config* 124
  cursor position display: *Lang Ref* 180; *Config* 132
  default message: *Lang Ref* 58
    overriding: *Lang Ref* 58, 192; *App Dev* 202

Status line (continued)
setting: *Lang Ref* 440
display attributes: *Config* 124–125
getting: *Lang Ref* 407
setting: *Lang Ref* 421
flushing: *Lang Ref* 346
force user to acknowledge: *Config* 28
formStatus resource: *Config* 175
in screen editor: *Editors* 11
keywords for video: *Config 107*, 124–125
location, setting in Motif: *Config* 175
message: *Lang Ref* 388
message functions: *App Dev 200–203*
message priority: *App Dev* 202
message types: *App Dev* 202–203
opening: *Config* 124
Ready/Wait status, toggling: *Lang Ref* 442
setting
display attributes: *Config* 26–28
position: *Config* 26
text attributes: *Config* 27
setup variables: *Config* 26–28
terminal, portability: *App Dev* 479
text
displaying: *Editors* 270–272
for menu item: *Editors* 220
text not visible: *Config* 27

Status line function: *App Dev* 143–144
cursor position display: *Lang Ref* 180
example: *App Dev* 188
return codes: *App Dev* 143

Status Text property, for menu item: *Editors* 220

STEXTATT: *Config* 27

STORE, dbms command, setting continuation file: *Database* 174; *App Dev* 232–233

Stored procedures
executing
in Informix: *Database* 218–221
in ODBC: *Database* 268
in Oracle: *Database* 297–300, 311
in SYBASE: *Database* 333–334, 371–378
executing an rpc, in SYBASE: *Database* 373–378
return codes: *Database* 189, 376

String
getting length: *Lang Ref* 37
reading from file: *Lang Ref* 242
writing to file: *Lang Ref* 248

String expression, JPL: *Lang Ref* 38

Stripe Current Row property: *Editors* 188

Stub functions: *App Dev* 525

Style property
for graph widget data series: *Editors* 139–141
for graph widget tick marks: *Editors* 133–134
for list box border: *Editors* 201
for wallpaper: *Editors* 108

Style Source property, for graph widget, pie chart: *Editors* 148–150

Styles. *See* Styles editor; Transaction styles

Styles editor
features: *Editors* 7
invoking: *Editors* 281–282
Pages: *Editors* 281

styles.sty, saving: *Editors* 292

Sub–system: *App Dev* 521

Subdetail specification, in screen wizard: *Editors* 77

Submenu
attaching to menu item: *Editors* 220, 222
creating: *Editors* 216
indicator: *Config* 132
naming: *Editors* 220
nesting: *Editors* 216

Submenu property, for menu item: *Editors* 220

Subqueries, database: *Database* 100–102

Substring specifier: *Lang Ref* 36
colon variables: *Lang Ref* 22

Subtitle property, for graph widget: *Editors* 125

Subtraction operation, in JDB: *Database* 90

SUM function, in JDB: *Database* 59

Support routine
See also Database drivers
database engines: *App Dev* 209

SYBASE: *Database* 319–424, 355–460
connection options: *Database* 324, 359
DBMS command listing: *Database* 351, 418
executing statements: *Database* 346, 409
executing stored procedures: *Database* 333, 371
transaction manager cursor usage strategies: *App Dev* 365

Synchronization property: *Editors* 313; *App Dev* 379

Synchronized arrays: *Editors* 312–316
creating: *Editors* 314
getting next: *Lang Ref* 396
identifying members of: *Editors* 314–315
isolating: *Editors* 315
runtime properties: *Lang Ref* 548
updating group of: *Editors* 315

Syntax summary, JDB: *Database* 109

System. *See* Operating system

system, executing operating system command in ISQL: *Database* 120

System decimal
defining symbol: *Config* 69
interpreting: *App Dev* 482

System menu
displaying on screen border: *Editors* 113
setup variable: *Config* 32

System Menu property: *Editors* 113

System tables, JDB: *Database* 30

System Update property: *Editors* 245

T

TAB key
acting like XMIT: *Config* 38
hex value: *Config 79*

Tab order
and cursor positioning keys: *Editors* 306
changing: *Editors* 306–307
default: *Editors* 306
hidden widgets and: *Editors* 234
in arrays: *Editors* 309–311
in groups: *Editors* 306
specifying: *Editors* 306–312

Table Lookup property: *Editors* 279

Table property, for table view: *Editors* 352
in automated SQL generation: *App Dev* 281, 293, 297, 300

Table views: *Editors* 337–349; *App Dev* 313, 354–357
accessing properties for: *Editors* 338
adding members to: *Editors* 341
clearing fields: *Lang Ref* 459

Table views (continued)
connecting to database: *Editors* 341
creating: *Editors* 340; *App Dev* 354, 355
from database import: *Database* 209, 240, 262, 290, 325, 360
determining relationships: *Editors* 93
displaying members: *Editors* 341
getting correlation name: *Lang Ref* 146
guidelines for using: *App Dev* 463–465
identifying as root: *Editors* 342; *App Dev* 315, 356
linking two : *Editors* 345
properties for SQL generation: *Editors* 351–353; *App Dev* 274, 281, 293, 297, 300
runtime properties: *Lang Ref* 549
selecting: *Editors* 338
setting child table view: *App Dev* 314, 358–359
setting parent table view: *App Dev* 314, 358–359
specifying additional in screen wizard: *Editors* 73
attaching selection screen: *Editors* 79
specifying master in screen wizard: *Editors* 71
synchronized scrolling: *Editors* 312
synchronizing: *App Dev* 379
traversal properties: *App Dev* 380–383
updatable: *Editors* 91

Tables
correlation names: *Database* 23
defined: *Database* 9–10
describing in JISQL: *Database* 46, 50
dropping using JISQL: *Database* 46–47
dumping in JISQL: *Database* 50
export from JDB to text files: *Database* 123
import to JDB from text files: *Database* 123
importing to repository: *Editors* 59–62
in automated SQL generation: *App Dev* 274, 281, 293, 297, 300
JDB
creating: *Database* 66–68
creating in JISQL: *Database* 36–37
deleting: *Database* 74
joining multiple: *Database* 22–23, 81–85
keys, defining using JISQL: *Database* 39–45
naming conventions, JDB: *Database* 28
re–importing: *Editors* 61
retrieving data: *Database* 96
storing in repository: *App Dev* 63–64
system tables, JDB: *Database* 30

Target string: *App Dev* 112

tbldata, import/export JDB database: *Database* 123

Tear-Off property, for menus: *Editors* 217

Tear–off menu: *Editors* 217

Template: *Editors* 17
  from repository entry: *Editors* 18

TERM: *Config* 14

term2vid: *Config* 101–102

termcap: *Config* 90
  padding in: *Config* 101
  supporting: *Config* 91

Terminal
  assigning timing interval: *Config* 111
  attributes: *Config* 115
  bell: *Editors* 272
    in message: *Config* 28, 59
  changing display size: *Lang Ref* 427
  flush delayed write: *Lang Ref* 50
  flushing output: *Lang Ref* 251
  getting identifier: *Lang Ref* 406
  graphics character display: *App Dev* 477
  initializing: *Lang Ref* 282; *Config* 109
  mapping input to output: *Config* 126
  output: *App Dev* 477, 525
  portability: *App Dev* 476, 479
  refreshing: *Lang Ref* 425
  reset sequence: *Config* 111
  resetting to system defaults: *Lang Ref* 426
  status line: *App Dev* 202
  timing interval: *Config* 100
  type. *See* Terminal type
  visible bell: *Config* 131

Terminal type, setting: *Config* 14

Terminal–specific variable: *Config* 14

terminfo: *Config* 90
  commands, not supported: *Config 97*
  padding in: *Config* 101

Test mode: *Editors* 34–35; *App Dev* 6
  entering: *Editors* 34
  exiting: *Editors* 35
  menu bar: *Editors* 4
  vs. application mode: *Editors* 6, 34

Testing screens: *Editors* 34–35

Testing transaction styles: *Editors* 291

Text
  entry widgets: *Editors* 24
  format and display properties: *Editors* 233–241
  graph widgets: *Editors* 124–129
  hiding: *Editors* 234
  justification: *Editors* 234
  selection appearance: *Config* 25

Text editor
  invoking for JPL procedures: *Lang Ref* 12
  naming for JPL procedures: *Config* 17

Text files, import/export to JDB database: *Database* 123

Text Size property
  for graph widget: *Editors* 125
  for graph widget labels: *Editors* 125, 129
  for graph widget legend: *Editors* 125, 127
  for graph widget pie chart labels: *Editors* 148
  for graph widget subtitle: *Editors* 125, 126
  for graph widget title: *Editors* 125

Text widget
  3D (in Windows): *Editors* 250
  selection appearance: *Config* 25

Thousands Separator property: *Editors* 247

Tick marks, graph widget axes: *Editors* 132–135

Time format. *See* Date/time format

Time–out delay: *Config* 132

Timeout function: *App Dev* 134–136
  example: *App Dev* 171
  return codes: *App Dev* 135
  standard arguments: *App Dev* 135
  testing input: *Lang Ref* 313

TIMEOUT_FUNC. *See* Timeout function

Timing interval
  assigning to keyboard input: *Config* 111
  setting with percent commands: *Config* 100

Title
  for MDI window: *Config* 156
  on grid columns: *Editors* 185
  on grid rows: *Editors* 187

Title Bar property: *Editors* 112

Title property
  for graph widget: *Editors* 125
  for graph widget legend: *Editors* 127
  for list box widget: *Editors* 201

Title property (continued)
for screens: *Editors* 111
on borders: *Editors* 265
on smwizard screen: *Editors* 85
specifying in screen wizard: *Editors* 79

TM Class property
See also Class property, for menu item
for menu items: *Editors* 221

Toggle button widget: *Editors* 26, 205
displaying image on: *Editors* 266
label text alignment: *Editors* 235

Toggle indicator, on menu item: *Editors* 221

Toggle menu item: *Editors* 221

Tool box: *Editors* 14
creating widgets with: *Editors* 22
multiple create mode: *Editors* 16

Toolbar: *App Dev* 92–93
adding items: *Editors* 219; *App Dev* 92
assigning images to items: *Editors* 219, 224
assigning in screen wizard: *Editors* 79
assigning tooltip text to item: *Editors* 220
displaying: *App Dev* 87
enabling display: *App Dev* 92; *Config* 26
ordering items in: *Editors* 219
screen wizard prototype: *Editors* 86
setup variables: *Config* 26
sizing images: *Editors* 225

Toolbar property, for menu items: *Editors* 219

TOOLBAR_DISPLAY: *Config* 26

Tooltip
assigning to toolbar item: *Editors* 220
enabling display: *Config* 26
testing: *Editors* 227

Tooltip property, for toolbar items: *Editors* 220

TOOLTIP_DISPLAY: *Config* 26

Top Screen option: *Editors* 4

TP, transaction monitor message tag prefix: *Config* 44

TRANSACTION, dbms command, setting a default transaction: *Database* 414

Transaction
committing in JISQL: *Database* 50
error handling: *App Dev* 267–270
in the transaction manager
changing transactions: *App Dev* 403
closing the current transaction: *App Dev* 436–437
starting a new transaction: *App Dev* 455–457
processing database transactions: *Database* 221–224, 245–247, 268–271, 300–303, 334–337, 378–385; *App Dev* 265–269
committing the transaction: *Database* 228, 249, 275, 308, 341, 397
in JDB: *Database* 115–116
rolling back a transaction: *Database* 229, 250, 277, 312, 345, 405
setting a savepoint: *Database* 313, 407
starting transaction: *Database* 226, 339, 388
restoring journals/logs: *Database* 121
rolling back in JISQL: *Database* 50
transaction model strategies: *App Dev* 365–367
unspecified: *Editors* 343

Transaction classes: *Editors* 285–287; *App Dev* 349–352
defaults: *Editors* 286–287
for fields: *App Dev* 349–350, 351–352
menu options, setting for: *App Dev* 335–336
push buttons, setting for: *App Dev* 335–336
style assignments by mode: *Editors 287*; *App Dev* 351

Transaction commands, executing: *Lang Ref* 461

Transaction events: *App Dev* 337–345, 399–402
adding to the stack: *App Dev* 339
after an error: *App Dev* 372–373, 398–400
by transaction manager command: *App Dev* 340–345
emptying event stack: *Lang Ref* 460
event stack: *App Dev* 338–339
getting the event name: *Lang Ref* 471
getting the event number: *Lang Ref* 470
popping event from stack: *Lang Ref* 484
pushing event onto stack: *Lang Ref* 487
unsupported events: *App Dev* 388

Transaction manager
accessing traversal properties: *Lang Ref* 31
attributes
getting integer values: *Lang Ref* 473–475
getting string values: *Lang Ref* 481–482, 483
setting integer values: *Lang Ref* 476–477
setting string values: *Lang Ref* 485–486

Transaction manager (continued)
before image processing: *Lang Ref* 171–172, 173, 174–175; *App Dev* 323, 366–367
changing to update mode: *App Dev* 430–431
changing to view mode: *App Dev* 432–433
classes: *App Dev* 349–352
clearing data in widgets: *App Dev* 404–405
closing a screen: *App Dev* 325
closing current transaction: *App Dev* 436–437
closing database transaction: *App Dev* 406–408
commands: *App Dev* 329–336, 395–461
listing of events: *App Dev* 340–345, 399–402
connecting to the database: *App Dev* 318, 320
copying data, for edit: *App Dev* 428–429
creating screens for: *App Dev* 310–317, 353–354
customizations: *App Dev* 369–393
deleting data: *App Dev* 324, 379–380
description of: *App Dev* 309–325
discarding changes: *App Dev* 438–439
entering new data: *App Dev* 324, 440–442
error processing: *App Dev* 371–375
controlling display: *App Dev* 374
error list: *App Dev* 465–469
errors
logging errors: *Lang Ref* 468
message processing: *Lang Ref* 463, 464
reporting: *Lang Ref* 467, 478–481
specifying message: *Lang Ref* 472
testing for database errors: *Lang Ref* 466
event stack
emptying: *Lang Ref* 460
popping event off stack: *Lang Ref* 484
pushing event onto stack: *Lang Ref* 487
fetching data: *App Dev* 321–322, 377–379
for update: *App Dev* 323–325, 379, 450–454
for view: *App Dev* 458–462
getting first set of rows: *App Dev* 420–423
getting last set of rows: *App Dev* 412–415
getting next set of rows: *App Dev* 409–411, 416–419, 434–435
getting previous set of rows: *App Dev* 424–427
number of rows fetched: *App Dev* 378–379
setting backward scrolling: *App Dev* 377–378
guidelines for using: *Editors* 337
hook functions: *App Dev* 146–147, 384–393
checking for database errors: *App Dev* 387
DELETE statement: *App Dev* 391–393
INSERT statement: *App Dev* 391–393
return codes: *App Dev* 147, 398–400

Transaction manager (continued)
SELECT statement: *App Dev* 388–393
specifying return codes: *App Dev* 385–388
standard arguments: *App Dev* 146
UPDATE statement: *App Dev* 391–393
initiating a transaction: *App Dev* 455–457
non–sequential scrolling: *Lang Ref* 465
opening a screen: *App Dev* 317
processing for transaction commands: *App Dev* 395–461
processing transaction commands: *Lang Ref* 461
refreshing the screen: *App Dev* 443
restrictions: *App Dev* 398
saving database changes: *App Dev* 444–449
setting the transaction mode: *App Dev* 397
specifying commands: *App Dev* 331
SQL generation: *App Dev* 274–277
switching transactions: *App Dev* 403
transaction events: *Lang Ref* 470, 471; *App Dev* 337–345, 399–402
transaction requests: *App Dev* 337–345
tree traversal: *App Dev* 315–316, 337–338, 359, 398
troubleshooting: *App Dev* 463–469
using the Transaction menu: *App Dev* 318–319

Transaction manager commands: *App Dev* 329–336, 395–461
availability by mode: *App Dev* 346–348
listing of events: *App Dev* 340–345
specifying full commands: *App Dev* 332
specifying partial commands: *App Dev* 332
specifying the table view: *App Dev* 397

Transaction mode: *Editors* 283–285; *App Dev* 346–348
availability of commands: *App Dev* 346–348
changing to initial mode: *App Dev* 406–408, 438–439
changing to new mode: *App Dev* 440–442
changing to update mode: *App Dev* 430–431, 450–454
changing to view mode: *App Dev* 432–433, 458–462
setting: *App Dev* 319–320, 397

Transaction model: *App Dev* 324, 364–366
cursor usage strategies: *App Dev* 365
for JDB: *Database* 248
for Oracle: *Database* 303
for SYBASE: *Database* 337, 386
initializing: *App Dev* 207–208
return codes: *App Dev* 385, 398–400

Transaction model (continued)
specifying: *Editors* 339
specifying in Windows: *App Dev* 211
transaction strategies: *App Dev* 365–366

Transaction monitor interface message tag: *Config* 44

Transaction properties
for link widgets: *Editors* 344
for screens: *Editors* 33
for table view: *Editors* 338–340
for widgets: *Editors* 33

Transaction styles: *Editors* 285–287; *App Dev* 322–323, 349–352
ASCII: *Editors* 292
assigned to default classes: *Editors 287*
converting to/from ASCII: *Editors* 292
creating: *Editors* 288–290
defaults: *Editors* 285–286; *App Dev* 350–351
for menu items: *App Dev* 335–336
for push buttons: *App Dev* 335–336
report, s2asc: *Editors* 292
saving: *Editors* 292

Transitive links, creating via the screen wizard: *Editors* 92

Translating: *App Dev* 489–494
applications: *App Dev* 482–494
currency fields: *App Dev* 486–488
message file: *App Dev* 483; *Config* 47
messages: *Config* 43
physical keyboard: *Config* 73–88
substitution variables: *Config* 63, 68

Traversal properties, accessing at runtime: *Lang Ref* 31

Turning on/off inheritance: *Editors* 64–66

Tutorial
running on Motif or character mode: *Tutorial* 4
running on Windows: *Tutorial* 3
starting: *Tutorial* 4

TXT_SELECT_ATTR: *Config* 25

TXT_SELECT_MASK: *Config* 25

TYPE, dbms command, specifying parameter types: *Database* 415

Type property
defined in database. *See* C Type property
for menu items: *Editors* 218
sequential: *Editors* 345
server: *Editors* 345

Type–ahead buffer: *Config* 28

U

UARR key (up arrow), hex value: *Config 79*

UINIT_FUNC. *See* Initialization function

Underline display attribute, setting: *Config* 118

Underlined property: *Editors* 239

Undo levels: *Editors* 16

Undoing
actions: *Editors* 51
property changes: *Editors* 31

Unfiltered data: *Editors* 162

Unifying widget size: *Editors* 42–44

UNIQUE, dbms command, suppressing repeating values: *Database* 177; *App Dev* 236–237

Unique keys, defining using JISQL: *Database* 39–41

Unnamed procedure: *Lang Ref* 3
getting standard arguments: *Lang Ref* 5

Unsupported features, JDB: *Database* 7

Updatable property: *Editors* 338
in automated SQL generation: *App Dev* 292, 296, 299

UPDATE, dbms command, updating in browse mode: *Database* 416

Update group, for table view: *Editors* 342

Update Order property: *Editors* 346
in automated SQL generation: *App Dev* 296

UPDATE statement
constructing: *Database* 20–21
in JDB: *Database* 103–104
NULL values and: *Database* 88
SQL generation from properties: *App Dev* 296–299, 303–304

URESET_FUNC. *See* Reset function

USE, dbms command, specifying a database: *Database* 417

Use If Null property: *Editors* 356
in automated SQL generation: *App Dev* 283

Use in Insert property: *Editors* 356
expression: *Editors* 356; *App Dev* 293–295
in automated SQL generation: *App Dev* 293

Use in Select property: *Editors* 353–356
expression: *Editors* 354; *App Dev* 280–281, 284
in automated SQL generation: *App Dev* 280–281

Use in Update property: *Editors* 356
expression: *Editors* 357; *App Dev* 297
in automated SQL generation: *App Dev* 297

Use in Where property: *Editors* 355
in automated SQL generation: *App Dev* 281–284
operator specification: *Editors* 355

User messages. *See* Application messages

UT, message tag prefix: *Config* 44

Utilities
file extensions, default behavior: *Config* 34
JDB: *Database* 117–123
updating inheritance (binherit): *App Dev* 66–68

Utilities message tag: *Config* 44

V

Validation: *App Dev* 81
automatic help: *Editors* 272
character–level: *Config* 36
clearing MDT bit: *Lang Ref* 184; *App Dev* 82
executing check digit function: *Lang Ref* 183
field: *Editors* 303, 330–331; *App Dev* 125
field function invocation: *App Dev* 125
forcing
for field: *Lang Ref* 262
for group: *Lang Ref* 278
for screen: *Lang Ref* 433
grid frame widget: *Editors* 332–333
group: *Editors* 333–334
MDT bit: *App Dev* 81
screen: *Editors* 331; *App Dev* 125
setting VALIDED bits: *App Dev* 81
setup options: *Config* 24
testing screen for modified data: *Lang Ref* 491; *App Dev* 82
using check digit: *Editors* 319
using table lookup: *Editors* 278

Validation (continued)
validation bit: *App Dev* 81
XMIT key: *Editors* 331; *App Dev* 125

Validation bit: *App Dev* 81
setting: *App Dev* 81

Validation Func property
for grid: *Editors* 332
for selection group: *Editors* 333
for widgets: *Editors* 330

Validation Link property: *Editors* 349, 359; *App Dev* 361–364
set via the screen wizard: *Editors* 83
setting on a widget: *App Dev* 316

Validation properties, for widgets: *Editors* 32

Validation Protection property: *Editors* 304

VALIDED bit. *See* Validation

Value Location property, for graph widget
bar/line graph: *Editors* 154
pie chart: *Editors* 147

Value Source property
for graph widget
bar/line graph: *Editors* 154
high/low chart: *Editors* 160
pie chart: *Editors* 145
for graph widget data series, Pages: *Editors* 137

var2bin: *Config* 10–11
errors: *Config* 11

Variable
See also JPL variable
configuration: *Config* 13–14
declaring global in JPL: *Lang Ref* 53
declaring in JPL: *Lang Ref* 76
environment: *Config* 14–18
watching through debugger: *App Dev* 517

Version Column property: *Editors* 358; *App Dev* 370–371
in automated SQL generation: *App Dev* 295–296, 298–299, 300–301

Version control: *Editors* 366

Vert Rotation property, for graph widget: *Editors* 136
bar/line graph: *Editors* 153
high/low chart: *Editors* 160
pie chart: *Editors* 145
XY plot: *Editors* 156

Vertical Anchor property, for widgets: *Editors* 50

Vertical lines, in grid frames: *Editors* 186

Vertical Scroll Bar property
  for grid frames: *Editors* 187
  for list box widget: *Editors* 202
  for scrolling arrays: *Editors* 243

Vertical Shrinking property, for a screen: *Editors* 105, 264

vid2bin: *Config* 104–105
  errors: *Config* 104

Video file: *App Dev* 474; *Config* 89–135
  converting terminfo/termcap to: *Config* 101
  converting to binary: *Config* 104–105
  creating: *Config* 101–102
  identifying for initialization: *Config* 16
  international character support: *Config* 125–128
  keyword summary: *Config 105*
  names of: *Config 2*
  parameter changing commands: *Config* 99
  parameter sequencing
    commands: *Config* 98
    for processing keywords: *Config* 91
  parameters for keyword sequences: *Config* 93–101
  sample
    ANSI terminal: *Config* 133
    MS–DOS: *Config* 133
  screen size entries: *Config* 111
  syntax: *Config* 92–93
  variable: *Config* 16

Video mapping
  character sets: *App Dev* 477
  file: *App Dev* 474, 476–477
  initializing: *Lang Ref* 496
  internationalization: *App Dev* 483
  optimization: *App Dev* 525

Video processing function: *App Dev* 144–146
  arguments: *App Dev 144*
  return codes: *App Dev* 145
  standard argument: *App Dev* 144

videobiz
  description of database: *App Dev* 569–575
  diagram: *Database* 15

VideoBiz tutorial: *Tutorial* 1

VIEW, transaction manager command, fetching data for view: *App Dev* 458–462

View menu: *Editors* 14

Viewing reports: *Config* 18

Viewport: *Editors* 100; *App Dev* 74
  changing: *Editors* 100
  enabling user to change: *Lang Ref* 505
  specifying size/dimension, in control string: *Editors* 324; *App Dev* 111

Virtual screen: *Editors* 100

Visible bell: *Config* 131

Visual Object Repository. *See* Repository

VPROC_FUNC. *See* Video processing function

VWPT key (viewport): *Lang Ref* 505
  hex value: *Config 79*

W

Wallpaper: *Editors* 107–108
  file types supported for: *Editors* 107
  for MDI window: *Config* 157

Wallpaper Pixmap property: *Editors* 108

Warning messages, database: *Database* 190, 191

WHERE clause
  changing generated SQL: *Lang Ref* 143–145
  constructing: *Database* 19–20
  in automated SQL generation: *App Dev* 281–284, 297, 299, 300, 301
  in JDB: *Database* 105–108

While loop: *Lang Ref* 78

Widget ID, getting for
  base window: *Lang Ref* 516
  display: *Lang Ref* 517
  drawing area: *Lang Ref* 220
  screen–resident widget: *Lang Ref* 499

Widget list, using: *Editors* 38–39

Widget name
  assigning: *Editors* 299–300; *App Dev* 78
  case sensitivity: *App Dev* 225
  getting: *App Dev* 79
  in math expressions: *Editors* 317
  in tab properties: *Editors* 307
  using in Relations dialog box: *Editors* 346

Widget properties, set by the screen wizard: *Editors* 82

Widget runtime properties: *Lang Ref* 526
getting: *Lang Ref* 413–415; *App Dev* 81–82
getting handle to: *Lang Ref* 416
setting: *Lang Ref* 418–420

Widget Type property: *Editors* 232

Widgets
See also Field; specific widget types; Widget list; Widget name
3D (in Windows): *Editors* 250–251
about: *Editors* 23
accessing properties for: *Editors* 30
accessing with mnemonic: *Editors* 299
aliasing to column names: *Database* 135–137
arranging: *Editors* 46–49
calls to JPL from: *Lang Ref* 15
changing types: *Editors* 232–233
class names in Motif: *Config* 178–188
command execution with: *Editors* 25
copying
at runtime: *Lang Ref* 399
from repository: *Editors* 62
in screen editor: *Editors* 44–45
copying from repository, for transaction manager: *App Dev* 311–312
creating: *Editors* 22
from database import: *Database* 210–211, 241–242, 263–265, 291–293, 326–328, 361–363
creating with keyboard: *Editors* 373–374
database type: *Editors* 27
decoration type: *Editors* 26–27
deleting
at runtime: *Lang Ref* 401
in screen editor: *Editors* 45
deselecting: *Editors* 38
displaying images on: *Editors* 266–268
dominant selection: *Editors* 37
setting: *Editors* 38
drawing area: *Config* 180
entering data automatically with: *Editors* 24–25
font property: *Editors* 237
getting runtime properties. *See* Widget runtime properties
hierarchy, Motif: *Config* 178–188
base screen: *Config* 178–188
dialog box: *Config* 179
fields: *Config* 180
JAM screens: *Config* 179–188
menu bars: *Config* 183

Widgets (continued)
identifying: *Editors* 297–301
inheritance. *See* Inheritance
JPL, attaching procedure: *Editors* 329–330
JPL validation: *Lang Ref* 8
menu, attaching: *App Dev* 90
positioning: *Editors* 46–49
aligning automatically to grid: *Editors* 16
aligning with other widgets: *Editors* 46–48
centering in screen: *Editors* 49–51
moving: *Editors* 44–45
spacing evenly: *Editors* 48–49
units of measurement: *Editors* 41
properties
affecting colon preprocessing: *App Dev* 241–244
affecting formatting: *App Dev* 241–244
properties for SQL generation: *Editors* 353–359
protecting: *Editors* 302–305
reading data. *See* Field data
referencing in JPL
by name: *Lang Ref* 26
by number: *Lang Ref* 26
repositioning at runtime: *Lang Ref* 156
resizing: *Editors* 40–44
at runtime: *Editors* 42
units of measurement: *Editors 41*
selecting: *Editors* 37–39
selecting with keyboard: *Editors* 374–375
selection type: *Editors* 25–26
setting runtime properties. *See* Widget runtime properties
size
changing: *Editors* 40–44
specifying: *Editors* 40
unifying: *Editors* 42–44
specifying initial data for: *Editors* 231
storing templates in repository: *App Dev* 64
validating: *App Dev* 81
writing to. *See* Field data

Width property
defined: *Editors* 40
for graph widget tick marks: *Editors* 134
for widgets: *Editors* 41

Wildcard characters, in JDB: *Database* 86

win.ini: *Config* 160

Window
See also Screen
changing focus among siblings: *Lang Ref* 506; *App Dev* 72
changing from sibling to stacked: *App Dev* 72
changing from stacked to sibling: *App Dev* 72
closing: *Lang Ref* 187, 301
deselecting: *Lang Ref* 498; *App Dev* 72
displaying messages in: *Config* 60–69
giving focus to: *Lang Ref* 508; *App Dev* 72
opening: *Lang Ref* 309, 501; *App Dev* 73
as sibling: *App Dev* 71
from control string: *Editors* 323
placement in MDI frame: *Config* 157
setting next as sibling: *App Dev* 72
setting next-opened as sibling: *Lang Ref* 441
sibling: *Lang Ref* 508

Window stack: *App Dev* 70–73
changing order: *Lang Ref* 508; *App Dev* 72
counting windows: *Lang Ref* 497
deselecting window: *Lang Ref* 498

Windows
color definition: *Config* 155
control panel: *Config* 156, 160
setting defaults. *See* Windows initialization file

WINDOWS flag (video file): *Config 110*

Windows initialization file
3D: *Config* 156
application behavior options: *Config* 156–167
colors: *Config* 155–156
defining bitmap location: *Editors* 267
defining pixmap location: *Editors* 225
FrameTitle: *Config* 156
JAM Colors: *Config* 155
location: *Config* 153
name: *Config* 154
sample: *Config* 161–167
setting defaults: *Config* 153–167
SMTERM: *Config* 158
splash screen: *Config* 157
syntax: *Config* 154–167
wallpaper for MDI window: *Config* 157
window placement in MDI frame: *Config* 157

Windows list menu item: *Editors* 222

Windows operations menu item: *Editors* 222

WinHelp: *Editors* 280

WITH CONNECTION, dbms command, setting database connection: *Database* 178–179; *App Dev* 214

WITH CURSOR, dbms command, setting database cursor: *Database* 180–181

WITH ENGINE, dbms command, setting database engine: *Database* 182; *App Dev* 211

WMODE (wordwrap toggle mode), hex value: *Config 80*

WNL (wordwrap newline), hex value: *Config 80*

Word wrap function, specifying: *Editors* 174

Word Wrap property: *Editors* 175

Word wrapped text
executing entry/exit procedures for each occurrence: *Config* 38
fetching column values: *App Dev* 229
getting length: *Lang Ref* 510
reading from field: *Lang Ref* 512
setting tab space: *Config* 38
special logical keys: *Editors* 175
writing to field: *Lang Ref* 513

WTAB (wordwrap tab), hex value: *Config 80*

WW_COMPATIBLE: *Config* 38

WWTAB: *Config* 38

X

X Anchor property, for graph widget legend: *Editors* 129

X axis. *See* Axis, graph widget

X Location property, for graph widget legend: *Editors* 128

X Position property
for graph widget, pie chart: *Editors* 146
for graph widget legend: *Editors* 128–129

X Value Source property, for graph widget, XY plot: *Editors* 156–157
Pages: *Editors* 137

XA library
connecting to: *Database* 289
using in JAM: *Database* 303

XAPPLRESDIR: *Config* 171

XBM files: *Editors* 266

Xdefaults: *Config* 171

XJam file: *Config* 170
  sample: *Config* 184–188

XKEY flag (video file): *Config 110*

XMIT key (transmit)
  and push buttons: *Editors* 326
  hex value: *Config 79*
  making TAB and NL act like: *Config* 38
  screen validation: *Editors* 331; *App Dev* 125

XMIT_LAST: *Editors* 331; *Config* 38

XPM files: *Editors* 266

xrdb: *Config* 171

XY plot: *Editors* 154–157
  creating: *Editors* 155
  data series style: *Editors* 140–141
  data source: *Editors* 156–157
  displaying in 3D: *Editors* 155–156
  legend: *Editors* 126, 127, 142–143

Y

Y Anchor property, for graph widget legend: *Editors* 129

Y axis property, for graph widget data series: *Editors* 142

Y Location property, for graph widget legend: *Editors* 128

Y Position property
  for graph widget, pie chart: *Editors* 146
  for graph widget legend: *Editors* 128–129

Y Value Source property, for graph widget, XY plot: *Editors* 156–157
  Pages: *Editors* 137

Y1 axis. *See* Axis, graph widget

Y2 axis. *See* Axis, graph widget

Yes/No
  data entry filter: *Editors* 163
  setting default values: *Config* 71
  translating: *App Dev* 482

Z

Zero Format property: *Editors* 248

ZM_DISPLAY: *Config* 29

ZM_SC_OPTIONS: *Config* 29

ZM_SH_OPTIONS: *Config* 29

Zoned decimal: *Editors* 301

ZOOM key, hex value: *Config 79*

Zoom window
  setting border attributes: *Config* 30
  setting border style: *Config* 30
  setup options: *Config* 29

ZW_BORDATT: *Config* 30

ZW_BORDSTYLE: *Config* 30