Welcome to RoboHELP. Click Topic (Ctrl+T) to add your first Help topic.

Glossary

This is the glossary for Visual Cafe, Visual Cafe Database Development Edition, and dbANYWHERE:

- Definitions
- URLs (not pop-ups because they're too large)
- dbawdsn.ini fields (for dbANYWHERE)
- Fields in Data Source configuration files (for dbANYWHERE)

How it was created

1. Started with defin.doc (the glossary for Visual Cafe).
2. Put entries in alphabetical order.
3. Added definitions from the Pro glossary (progloss.doc) and the dbaw glossary (db_pop.doc).
4. Verified that all definitions in the Visual Cafe User's Guide glossary (defin.doc) are already in here.
5. Created and attached glossary.dot for the styles (glossary.dot is a revised version of robortf.dot).
6. Combined duplicate entries.
7. Modified descriptions (so that they are clearer).
8. Added descriptions: JDK, Java API, Hierarchy Editor, dbNAVIGATOR, Project window, Form Designer, server, foreign key, optimistic concurrency, primary key, stored procedure.
9. Create the main glossary topic which can be added to the .cnt files. For the bitmaps, I used the samples provided on the CD that came with *Developing Online Help* by Boggan, Farkas, and Welinske. You may want to use different bitmaps because I don't know if it's legal to use these and because someone can probably create more attractive ones.

Ramifications

1. Files for projects: Each help project needs to:
   - Delete the current glossary/definitions file (defin.doc, progloss.doc, or db_pop.doc).
   - Add this file (gloss.doc).
   - Make whatever modifications are necessary to access the definitions in this file.
2. Duplicate entries:
   - debug mode: Kept the Debug_mode_glossary topic ID. Deleted the Break_mode_glossary topic ID.
   - Java Virtual Machine: Kept the Java_Virtual_Machine_glossary topic ID. Deleted the Java_VM_Java_Virtual_Machine topic ID.
   - database view, global view, and two views: Deleted database view and global view (this information was in the other view definitions). Combined the two view definitions. Kept the View_glossary topic ID. Deleted the global_view_glossary and view topic IDs (there were two of these).
   - URLs: Deleted the URL entry because it has been superseded by the dbANYWHERE Server machine URL entry. Deleted the URL topic ID.
   - class method, instance method, method: Deleted the class method and instance method entries. Kept the Method_glossary topic ID. Deleted the Class_method_glossary and Instance_method_glossary topic IDs.
   - component and visual object: Deleted the visual object entry and its Visual_Object_glossary topic ID. Kept the Component_glossary topic ID.
3. Changed entry names:
   - Changed class variable and instance variable to variable. Kept the Class_variable_glossary topic ID. Deleted the instance_variable topic ID.
   - Changed class member to member. Kept the same topic ID.
   - Changed invisible component and non-visible object to hidden component. Kept the

Invisible_object_glossary topic ID. Deleted the Non_visual_object_glossary topic ID.

4. Deleted entries:
   - control (topic ID = Control_glossary).
5. Since the main glossary topic uses mid-topic jumps, you must redefine the visual window so that it does not auto-size the height.

To do

1. action: Make sure all the valid actions are listed.
2. Data Source: what spreadsheets are supported? should this information go in the glossary definition or in the "Installing dbANYWHERE Server" manual?

Main glossary topic

a
b
c
d
e
f
g
h
i
j
k
l
m

n
o
p
q
r
s
t
u
v
w
x
y
z

A

abstract class
access type
action
active window
adapter
anchor component
applet
applet tag

K

(none)

L

M

N

O

P

Q

R

S

T

U

(none)

V

W

Watch window

window

workspace

X

(none)

Y

(none)

Z

(none)

Definitions

A

abstract class

A class that contains an abstract method and is incompletely defined; that is, at least one method is not complete. You cannot instantiate an abstract class.

access type

Scope of a variable or method:

- Public: Accessible from any class.
- Protected: Accessible from the class that defines the variable or method and from the class' subclasses.
- Private: Accessible from the class that defines the variable or method.
- Package: Accessible from any class in the package that defines the variable or method. This is the default access type.

action

Action performed on a database:

- delete: Deletes the current record from the Data Source.
- first: Moves the database cursor to the first record in the database table.
- new: Creates a new record in the database table.
- next: Moves the database cursor to the next record in the database table.
- prev: Moves the database cursor to the previous record in the database table.

active window

Window which receives keyboard input. Only one window can be active at any time.

adapter

Classes that implement an interface. The java.awt.event package provides, as a convenience, a series of listener classes that can be implemented by classes.

anchor component

Component which acts as the reference point when modifying a group of components. When you select multiple components, the last component selected is the anchor component. It has distinct, colored selection handles.

applet

Program

Special type of Java program that you can add to a Web page. To run an applet:

- Add the applet to a Web page and display the Web page in a Java-enabled Web browser.

  -or-

- Run the applet from the Applet Viewer. (The Applet Viewer is a tool that Sun provides with the JDK. The Applet Viewer is also included in Visual Cafe and Visual Cafe Database Development Edition.)

Class

Class in the Java API.

applet tag

An applet tag is HTML code that causes an applet to appear in a Web page. It has the following basic format:

<APPLET code="*applet*.class" width=*pixw* height=*pixh*></APPLET>

applet is the name of the applet.

pixw is the number of pixels for the width.

pixh is the number of pixels for the height.

Consult an HTML book for more information on the applet tag.

application

Program

Java program you can run from a computer that has the Java Virtual Machine.

Template

Template provided by Visual Cafe (and Visual Cafe Database Development Edition) when you choose File ▸ New Project

▸ JFC or AWT Application. This template creates a class that is an extension of the JFrame or Frame class. This class is a good base for a main application window.

B

bean

A component complying with the JavaBeans standard. Also called a JavaBean or a JavaBeans component. A bean is a reusable component that can be visually manipulated in a builder tool. The minimum requirements are that it can be instantiated (it is not an abstract class or interface) and has a class constructor method that takes zero parameters (it has a null constructor). It can also have properties and events, implement the serializable or externalizable interface, and follow method signature rules so it can be introspected.

boolean expression

Expression that evaluates to true or false.

Breakpoints window

Window that contains a list of all breakpoints in a project. Use this window to add, remove, or modify breakpoints. You can set simple breakpoints that stop execution at a certain line or method, or conditional breakpoints based on an expression.

To see the Breakpoints window, choose View ▸ Breakpoints.

bytecode

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

C

call stack

An area reserved in memory by the compiler to keep track of all method calls that are made.

Calls window

Debugger window that shows all the active method calls leading to the current process. When an applet or application calls a method, the Java Virtual Machine (Java VM) adds the method to the stack. When the method returns, the Java VM takes it off the stack. The currently executing method is on the top of the stack and the previous methods called are below it. This window is useful for following the flow of your code, for example.

To see the Calls window, View ▸ Call Stack.

class

Collection of variables and methods that you can use to define an object. The variables define the class structure. The methods define the class behaviors.

When compiled as a bytecode program (versus a native Win32 executable), a Java source file becomes one or more class files.

Class Browser

A three-pane window that lists the Java classes in your project and the methods and data members contained within each class:

- Classes pane — classes
- Members pane — methods and data in the selected class
- Source pane — source code of the selected member

Both the Classes and Members panes support keyboard incremental searches. As you type the name of a class or member, the list of matching items is refined until the class or member you want is automatically selected. You can display classes and members in a variety of views and filter the items that are displayed. The Source pane provides the same editing features as the Source window, while ensuring that you do not unintentionally change code outside of the object's scope.

To see the Class Browser, choose View ▸ Class Browser.

client machine

Machine that communicates with a dbANYWHERE Server machine to access a Data Source.

client software

Software that dbANYWHERE uses to communicate with a database engine such as:

- Oracle Server client software
- Microsoft SQL Server client software
- ODBC32 Administrator

Code Helper

Interprets the current context in the Source Editor and provides either a list of the methods in a given class, a list of the different versions of a particular method, or a list of class objects that begin with a particular character sequence. It also helps you enter Javadoc tags.

component

A JavaBeans component.

- A *visual component* is a user interface element, such as a window, menu, button, and so on, that is visible at run time and appears in the Visual Cafe Project window and Form Designer. It extends from the Java Component class and has a screen position, a size, and a foreground and background color.
- An *invisible component* is not visible at run time, such as a Timer, or displays in a different way in the Form Designer and at run time, such as a MenuBar. It does not extend from the Java Component class. In the Form Designer, an invisible component is represented by an icon that does not effect the form layout.
- Some components can contain other components, such as an application window containing a button; these components are called *containers*. In the Objects view of the Project window, the components in a container appear subordinate to the container, like a file system display. The containers at the top level are separate Java files in your project (called Visual Cafe *forms*), while the components in the containers are Java code within the container Java file.
- To create your user interface, you can display a form in the Form Designer, then drag onto the form a variety of components from the Component Library or Palette; you assign the component properties in a separate Property List window, and add interactions between components with the Interaction Wizard. Visual Cafe automatically creates the Java code for you during this design process.
- Components are like controls in C++.

Component Library

Collection of components that you can add to your applet or application. When you create a project template or add a component to the Component Palette, Visual Cafe adds the component to the Component Library. To see the Component Library:

View ▸ Component Library

Component Palette

Palette that provides easy access to components. You can place any component in the Component Library on the Component Palette.

conditional breakpoint

Breakpoint that lets you stop program execution when a specified expression evaluates to true.

connection statistics

DC messages which occur when a client disconnects. The format for a DC message is:

DC, <client IP address>, <initial connect time>, <connect time duration>, <number of requests processed>

where:

- <initial connect time> is in second since 1970
- <connect time duration> is in seconds

ConnectionInfo

For the dbANYWHERE API, when you use a database wizard or drag a Data Source item to a project, Visual Cafe Database Development Editon creates a ConnectionInfo object, showing the database you are connected to. It includes the default user name and password, which you can remove. The object supports multiple queries and updates simultaneously, reducing database resource requirements.

ConnectionManager

A bean, used with the JDBC API, as a logical container for JdbcConnection beans. Dragging a Data Source item from dbNAVIGATOR to a project creates ConnectionManager and JdbcConnection objects, which manage connections to a data source. You have one ConnectionManager object per project, and one JdbcConnection object per data source. You can also drag ConnectionManager objects to the Component Library.

container

Component that can contain other components. For example, Applet and Panel are containers.

A top-level container, also called a Visual Cafe form, is at the top level in the Objects view of the Project window. It has a corresponding Java file that appears in the Packages and Files views.

context menu

Menu that appears when you click the secondary mouse button in a Visual Cafe window. A context menu is often called a pop-up menu.

custom code

Java code that Visual Cafe has not automatically created for you.

D

data binding

The mechanism by which data is automatically passed between components and the data source. The data binding is the same whether your data is being obtained from the JDBC API or the dbANYWHERE API. The components adhere to a standard interface for the binding. If the component is not aware of the data-binding interfaces, the binding can be controlled by a mediator bean.

data source

Contains the information for creating a database connection, like the name of the database, its server, and its network location. A data source is what identifies a database to an ODBC- or JDBC-compliant database application. Before you can use your database in a Visual Cafe project, you must create its data source. Sometimes a data source is called a DSN (Data Source Name). For information about the Data Sources dbANYWHERE supports, see the dbANYWHERE manual.

database template

Forms which are pre-designed. You can choose amongst these by using the dbAWARE Template Wizard. Each Visual Cafe database template is designed for a different kind of information.

dbANYWHERE (naming conventions)

dbANYWHERE Server	Middleware database software that supports three-tier architecture for database access. Symantec dbANYWHERE is a database connectivity server that provides an implementation of the JavaSoft JDBC API as well as its own powerful java database connectivity API, called the dbANYWHERE API.
dbANYWHERE Workgroup Server	Version of the dbANYWHERE Server which is shipped with Visual Cafe Database Development Edition that restricts the number of licenses for use to five.
dbANYWHERE Server machine	Hardware on which dbANYWHERE software is running
dbANYWHERE	Both versions of dbANYWHERE

dbANYWHERE API

An API developed by Symantec for accessing databases using the dbANYWHERE Server. It is an SQL-level database protocol that makes use of the extensions provided by dbANYWHERE. Symantec dbANYWHERE is a database connectivity server that provides an implementation of the JavaSoft JDBC API as well as its own powerful Java database connectivity API, called the dbANYWHERE API.

dbAW subprotocol

A JDBC subprotocol supported by Symantec dbANYWHERE, which is a JDBC driver. Another JDBC subprotocol supported by Visual Cafe Database Development Edition is ODBC, which is used by the free JDBC-ODBC bridge (another JDBC driver). The JDBC API requires a URL to establish a connection to a database; the subprotocol is part of this URL syntax:

jdbc:dbaw://*host-name-or-IP-address*:*port-number*/*data-source-name*

When attempting to connect to a JDBC driver, the Java program makes a connect call, passing the URL to the driver. If the driver recognizes the information in the URL, including the subprotocol, the connection can be made.

dbAWARE component

Visual Cafe component that has additional properties for binding the component to a database row, column, table, or result set.

dbNAVIGATOR

Browser window which shows the servers, data sources, and contents of the data sources that are connected to those servers. You can also use the dbNAVIGATOR perform drag-and-drop operations on your forms and components. To see the dbNAVIGATOR:

View ▸ dbNAVIGATOR

debug mode

Temporary suspension of a running program. In debug mode, you can edit code, edit forms, set or clear breakpoints, debug threads, and view the calls on the call stack. After making any edits, restart program execution to see the effect of the changes.

Debug toolbar

Toolbar that has buttons for debugging.

Debugging window

Window for debugging an applet or application. The Debugging window is a special mode of the Source window: It is the Source window while you are debugging.

declaration

Statement that establishes an identifier and associates attributes with it, without necessarily reserving its storage for data or providing the implementation for methods.

definition

Declaration that reserves storage for data or provides implementation for methods.

design time

Time during which you build an applet or application in the development environment.

development machine

Machine that combines a client to a database server and a Java development environment such as Visual Cafe Database Development Edition.

dialog box

Window with a title bar that can receive and process input from a user.

- Applets cannot use dialog boxes.
- Dialog boxes are not suitable for an application's main window.
- Dialog boxes can contain components, but not menus.
- Use dialog boxes for temporary windows.

DML

Data Modification Language, a subset of SQL.

E

event

Action or occurrence to which an object can respond. Events are typically user actions that the program can capture and respond to. For example, mouse clicks, key presses, and mouse movements are events.

event adapter

Event listener interface that is implemented as a class.

event binding

Code that enables an object to receive and process an event. It is made of three parts: the event handler, the listener or adapter implementation, and the code to register the listener or adapter to the object triggering the event.

event handler

A method that is called when a certain type of event is triggered. Visual Cafe automatically generates the code needed to bind the occurrence of an event to an event handler when you create an interaction with the Interaction Wizard or from the Events/Methods pull-down menu of the Source window. An event binding is made of three parts: the event handler, the listener or adapter implementation, and the code to register the listener or adapter to the object triggering the event. The default name of an event handler is the object name, followed by an underscore then the name of the action that triggers the event.

event listener

An object that has defined the listener interface for a specific event. After this interface has been implemented in a class, an instance of this class may be registered as an event listener. When an event is generated, the event is sent to the object, as well as all other registered listeners.

exception

An event that occurs during the execution of your program that interferes with, disrupts, or stops the normal flow of instructions.

F

foreign key

Column or columns in one database table whose values match the primary key in another table. Foreign key columns may or may not be defined as Unique or NOT NULL.

form

A component that can only appear at the top level in the Objects view of the Project window. Applet, Frame, Window, and some dialog components are forms. In the Component Library, the Visual Cafe forms are all in the Forms group. When you open a form in the Form Designer, the form boundaries are the bounds of the window. You can position other components, such as text, buttons, and graphics, on the form; these components are contained by the form.

Form Designer

Window that visually displays form components so you can design the user interface of the form.

frame

Window which can contain components and menus.

G

garbage collection

Automatic detection and freeing of memory that is no longer in use. The Java run time system performs garbage collection so you don't have to explicitly free the memory associated with objects and other data.

H

Hierarchy Editor

Window that displays the class hierarchy for your applet or application. To see the Hierarchy Editor:

View ▸ Hierarchy Editor

identifier

Name of an item in a Java program.

incremental debugging

A feature that enables you to edit code while your program is executing or paused in the Visual Cafe debugger. Also called run-time editing. You enable this feature by choosing Tools ▸ Environment Options ▸ Debugging.

inheritance

Concept wherein classes automatically contain the variables and methods defined in their superclasses.

inner class

A class that is included within the body of another class, even within a method (called a local class). An inner class is also called a nested class. This feature is new for JDK 1.1 and is useful for creating adapter classes. After compilation, the inner class ends up in its own class file, which has a dollar sign ($) in its name.

instance

Data item based on a class. An instance of a class is usually called an object. For example, multiple instances of a Form class share the same code and are loaded with the same components with which the Form class was designed. At run time, the individual properties of components on each instance can be set to different values.

interaction

Relationship between two or more components, or a component and itself. The components may be on the same form or on different forms. An interaction consists of:

- One or more components (trigger component and action component)
- Trigger event
- Action

For example, you can connect a button (the trigger component) to a text box (the action component) so that when the user clicks on the button (the trigger event), the associated text box is enabled for user input (the action).

interface

A set of methods and constants to be implemented by another object. It defines the behavior, or certain characteristics, that another object implements. An interface can define abstract methods and final fields, but not the implementation of them.

introspection

The ability to read JavaBeans classes directly with the Core Reflection API using the Introspector class. This information is stored in a BeanInfo object and includes data such as properties, events, and all the accessible methods.

invisible component

Component that is not visible at run time, such as a Timer, or displays in a different way in the Form Designer and at run time, such as a MenuBar. It does not extend from the Java Component class. In the Form Designer, an invisible component is represented by an icon that does not effect the form layout. To toggle between displaying and hiding invisible components at design time:

Layout ▸ Invisibles

J

Java API: Java Application Program Interface

API provided by the JDK. For more information, see the Java Web page (java.sun.com).

Java file

File that contains components and Java source code.

Java Archive (JAR) file

Compressed archive file that complies with the JavaBeans standard. It is the primary method for delivering JavaBeans components. For example, a JAR file can contain one or more related beans, and any support files, including classes, icons, graphics, sounds, HTML documentation, serialization files, and internationalization files. You can deploy applets and applications from a JAR file. A JAR tool, called jar.exe on Windows computers, archives and extracts JAR files and is provided with JDK 1.1. In Visual Cafe, to use the JavaBeans components in a JAR file, you must first add the file to the Component Library.

Java VM: Java Virtual Machine

Virtual machine provided with the JDK (Java Development Kit). The machine contains:

- Bytecode translator that converts a downloaded binary Java file into instructions that the client machine can execute.
- Library routines that a Java applet calls.

For more information, see the Java Web page (java.sun.com).

JavaBeans

A standard for creating portable, cross-platform components.

JavaBeans component

A component complying with the JavaBeans standard. Also called a bean or JavaBean. A bean is a reusable component that can be visually manipulated in a builder tool. The minimum requirements are that it can be instantiated (it is not an abstract class or interface) and has a class constructor method that takes zero parameters (it has a null constructor). It can also have properties and events, implement the serializable or externalizable interface, and follow method signature rules so it can be introspected.

Javadoc

Javadoc is is a way to generate HTML documentation directly from source code. It is made up of two parts:

- Special comments with embedded tags in Java source code that help describe how to use a class, and its methods and fields. The Javadoc comments are also used by the Java compiler to report to the user if a class or member is deprecated.
- A utility that scans this Java source code and generates the HTML documentation from it. Extra HTML files can also be generated which show all the classes in a particular package, an index, and a hierarchical class tree list.

JDBC API

An API developed by JavaSoft as a standard SQL-level database protocol. JDBC, based on ODBC, is a database extension to Java. The Visual Cafe Database Development Edition implementation of the JDBC API includes the Symantec JDBC components, in addition to the base JDBC. While it is not an acronym, JDBC is often thought of as the Java Database Connectivity protocol.

JDBC-ODBC bridge

A free JDBC driver. It supports an ODBC subprotocol of the JDBC API. The JDBC API requires a URL to establish a connection to a database; the subprotocol is part of this URL syntax. For example:

jdbc:odbc:*//subname*

jdbc:dbaw:*//host-name-or-IP-address***:***port-number/data-source-name*

When attempting to connect to a JDBC driver, the Java program makes a connect call, passing the URL to the driver. If the driver recognizes the information in the URL, including the subprotocol, the connection can be made.

JdbcConnection

A bean, used with the JDBC API, that represents a connection to a database. It has properties for establishing connection parameters in accordance with JDBC. Dragging a Data Source item from dbNAVIGATOR to a project creates ConnectionManager and JdbcConnection objects, which manage connections to a data source. You have one ConnectionManager object per project, and one JdbcConnection object per data source. You can drag either of these objects to the Component Library.

JDK: Java Developer's Kit

Tools for developing Java applets and applications. The JDK is included in Visual Cafe and Visual Cafe Database Development Edition. It is also available on the Java Web page (java.sun.com).

join

Any database query that produces data from more than one table. As its name suggests, a join brings together data from the specified tables.

K

L

layout

Property for container objects like forms and panels. The Layout property automatically arranges components in a container so that they display well on different platforms, Web browsers, screen sizes, and resolutions.

list binding

Data binding that lets you display data from multiple database columns. List binding is helpful for populating list boxes.

local variable

Data item defined within a block of code and accessible only by the code within the block. For example, a variable defined in a Java method is a local variable and can't be used outside the method.

M

macro

Sequence of keystrokes. The Source editor's macro facilities let you record, save, and edit macros.

manifest file

A file that describes the contents of an archive, such as a JAR. The file provides information on certain parts of the archive and can provide information about any JavaBeans in a JAR. It is made of sections separated by empty lines. Each section has one or more headers. Each header uses a *attribute*:*value* format where *attribute* specifies various attributes of the archive contents and *value* is the relative name of the file being described. Attributes used include Manfest-Version, Name, Java-Bean, Digest-Algorithms, and XXX-Digest (related to Digest-Algorithms). See Sun Microsystem's Manifest File specification for more information.

master/detail relationship

A one to many relationship between two database tables. The join property of the detail QueryNavigator or RelationViewPlus component defines the column of the master view which is the common attribute in the one to many relationship between the tables. For instance, you might want to generate a result set which enumerates all customers who purchased a product called the Spectacular Widget. You create this master/detail relationship by joining the Spectacular Widget column of a Sales table (which also has columns for numbers of sales and price) in an SQL query that returns a list of all the customers in a Customer database table who purchased a Spectacular Widget.

mediator

Bean that lets you make a component database-aware. It is a bridge between the component and the QueryNavigator or RelationViewPlus components.

At design time, you can add a Mediator component to a form in a project and set its properties for a particular component on the form. If you are creating a JavaBeans component or have access to the component Java code, you can add a mediator to the component code, thereby encompassing its functionality within your component.

member

Variable or method defined in a class.

method

Behavior that acts on an object. A method is defined in a class.

- Class method: Method invoked using a class name.
- Instance method: Method invoked using the name of an instance (object).

Messages window

Window that displays all Visual Cafe informational and error messages. For example, if Visual Cafe detects an error during parsing, the error message is displayed here. You can double-click an error message to open the file in which the error was detected. To see the Messages window:

View ▸ Messages

metadata

Data about data. In a database, the schema contains metadata that describes the data in the tables. In CORBA, the interface repository contains metadata that describes the objects that have been registered with the ORB.

modal

Window or dialog box behavior that requires you to take some action before the focus can switch to another window or dialog box.

modeless

Window or dialog box behavior that does not require you to take some action before the focus can switch to another window or dialog box.

N

nested class

A class that is included within the body of another class, even within a method (called a local class). A nested class is also called an inner class. This feature is new for JDK 1.1 and is useful for creating adapter classes.

O

ODBC

Open DataBase Connectivity, a standard interface for communicating with databases. The dbANYWHERE server uses this interface to communicate with many of the databases it supports. The dbANYWHERE server also communicates to other major database types through direct native calls to the client APIs (application programming interfaces) for those databases.

ODBC subprotocol

A JDBC subprotocol supported by the JDBC-ODBC bridge, which is a free JDBC driver. (Each JDBC driver supports certain subprotocols.) Another subprotocol supported by Visual Cafe Database Development Edition is dbAW, which is used by Symantec dbANYWHERE. The JDBC API requires a URL to establish a connection to a database; the subprotocol is part of this URL sytax:

jdbc:odbc://*subname*

When attempting to connect to a JDBC driver, the Java program makes a connect call, passing the URL to the driver. If the driver recognizes the information in the URL, including the subprotocol, the connection can be made.

object

Instance of a class.

Objects view

View of a project that displays only visible components. To toggle between displaying and hiding hidden components at design time:

Layout ▸ Invisibles

optimistic concurrency

Locking technique that maximizes concurrency. The database management system (DBMS) locks data that is being modified but leaves alone data that is being viewed.

overloading

Using one identifier to refer to multiple items in the same scope. In Java, you can overload methods but not variables or operators.

overriding

Providing a different implementation of a method in a subclass of the class that originally defined the method. Overridden methods have the same name but take different parameters.

P

package

Group of classes and interfaces. Java source code is organized into packages. Each part of a package name generally refers to a hierarchical directory structure, for example, COM.sun.java.swing is in the directory structure /com/sun/java/swing. If you are going to distribute your packages, you should create a globally unique name based on an Internet domain name. For example, sun.COM is specified as COM.Sun. This first part of the name is in uppercase letters; if the first part is not in uppercase letters, it is for local use, except for packages that are part of the Java language and system (which start with java). Packages help prevent naming conflicts. For example, two Java files could have the same names, but as long as they are in different packages, there is no naming conflict.

Packages view

View of a project that displays both visible and hidden components. To toggle between displaying and hiding hidden components at design time:

Layout ▸ Invisibles

panel

Container that you can add to another container. A panel doesn't have a visible border. You can use a panel to group a window into logical regions.

ping: Packet INternet Groper

Network message that a computer transmits to check for the presence, connectedness, and responsiveness of another computer. The other computer responds by echoing the message back to the first computer.

primary key

Column or columns in a database table that uniquely identify a record. A value in a primary key column cannot be NULL and must be unique.

project

Set of components and code that comprise an applet or application.

project template

Collection of components that you can use as the foundation for an applet or application. When you choose a template for a new project, the new project inherits all of the template's components. For example, you can create project templates for Web sites, single documents, and applets.

Project window

Window that displays a project's objects or packages depending on which tab you select.

projection binding

Data binding to one row in a database column.

property

Attribute of a component. Properties define component characteristics such as size, color, label, or the state of an object, such as enabled or disabled. The Property List displays the properties of a project's components. To see the Property List:

View ▸ Property List

Property List

Window that displays a component's properties. To see the Property List:

View ▸ Property List

Q

QueryNavigator

A bean, used with the JDBC API, that manages a set of records. Dragging a Data Table item from dbNAVIGATOR to a form in a project creates a top-level RecordDefinition object and a QueryNavigator object, which is contained by the form. You can think of the RecordDefinition component as the data, and the QueryNavigator component as a way of looking at the data. After you have a QueryNavigator and RecordDefinition component in a project, you can create a master/detail relationship. You can drag a QueryNavigator object to the Component Library.

R

RecordDefinition

A bean, used with the JDBC API, that is used to define and access a row of data in a database table. Dragging a Data Table item from dbNAVIGATOR to a form in a project creates a top-level Record Definition object and a QueryNavigator object, which is contained by the form.

resource bundle

A file containing locale-specific information. When a program needs a resource specific to a locale, such as a string in French, the program can load the resource from a resource bundle for that locale. This way, you can write locale-independent code that stores locale-specific information in resource bundles. Two types of resource bundles are list type, which is a Java source file, and properties type, which is a text file.

refresh

Updates the dbNAVIGATOR window to reflect changes made to database tables and columns it is displaying.

regular expression syntax

Syntax that lets you perform regular expression matching. Regular expressions are wildcard characters. The pattern you search for can be a text string or a regular expression.

Wildcard characters

?	Any character.
*	Zero or more occurrences of any character.
@	Zero or more occurrences of the previous character or expression.
% or <	Beginning of a line.
$ or >	End of a line.
[...]	Any of the characters listed between [and]. You can use a hyphen (-) to specify a range of characters. For example, [abc] matches a, b, or c; [a-z] matches any lowercase letter; [A-Za-z] matches any upper or lowercase letter.
[~...]	Any character except those listed between [~ and]. You can use a hyphen (-) to specify a range of characters. For example, [~A] matches any character but A; [~abc] matches any character except a, b, or c; [~A-Za-z] matches any non-alphabetic character.
\	Take the following character literally instead of using it as a wild card character. For example, you can use * to search for an asterisk or \\ to search for a backslash character.
\t	Tab character.
\f	Formfeed character.

RelationViewPlus

Database-aware component that shows a view of a database and is used with the dbANYWHERE API. It provides a simple, property-driven solution for defining and maintaining a result set. It is invisible at run time but contains methods used to handle navigation and data manipulation operations on the data in the result set. (The ‘‘Plus’’ was added when the data binding changed in later versions of Visual Cafe Database Development Edition.)

result set

Set of data which is returned by a query on a database(s). An example of a result set might be the data from all the rows in an employee database table which have 200 as their department identification number.

run mode

Mode wherein your applet or application is running. In run mode:

- You can interact with your program as a user.
- Visual Cafe replaces the Project Menu with the Debug Menu.

run time

Time when your applet or application is running. At run time:

- You can interact with your program as a user.
- You can pause the application and start debugging by pressing CONTROL+BREAK.

S

scope

Access type of a variable or method:

- Public: Accessible from any class.
- Protected: Accessible from the class that defines the variable or method and from the class' subclasses.
- Private: Accessible from the class that defines the variable or method.
- Package: Accessible from any class in the package that defines the variable or method. This is the default access type.

Session

For the dbANYWHERE API, when you use the database wizard or drag a dbANYWHERE Server item to a project, Visual Cafe Database Development Edition creates a Session object, indicating the dbANYWHERE Server you are connected to through sockets.

server

- Database server: Computer that stores and manages a database.
- dbANYWHERE Server: Computer that runs dbANYWHERE, the middleware database software from Symantec which supports three-tier architecture for database access.
- File server: Computer that stores programs and data files.
- Web server: Computer that stores and sends out Web pages in response to HTTP requests from Web browsers.

servlet

Java component that resides on a Java-enabled server and can dynamically extend server-side functionality. A servlet can provide services using a request-response paradigm; it has no GUI. For example, a servlet can provide secure access to data presented in an HTML Web page and let users interactively view or modify data. .

Source pane

Pane in the Source window and Class Browser that lets you view and edit code.

Source window

Window that displays and lets you edit a Java source file, an HTML file, or a text file. It simplifies development with full-color Java syntax and keyword highlighting and an integrated Java macro language for extending the editor. During debugging, you can use the Source window to monitor program execution.

SQL

Standard query language which is used for retrieving information from a relational database. The acronym stands for Structured Query Language.

SQL statement

SQL string used to query a database and thereby define a view. If the string is in the standard SELECT statement format as shown below, the Visual Cafe Database wizards and Join Definition dialog box can parse the string and edit it for you. If the string is not in the standard SELECT statement format, you need to use the SQL Statement dialog box to edit the string manually.

Standard format

SELECT column[, column ...] FROM table WHERE condition

Example

SELECT emp_fname, emp_lname, location_id FROM emp_info WHERE location_id BETWEEN 10 AND 20

stored procedure

Collection of SQL statements that is named and precompiled.

Procedures and triggers store procedural SQL statements in a database for use by all applications.

Procedures and triggers can include control statements that allow repetition (LOOP_statement) and conditional execution (IF_statement and CASE_statement) of SQL statements.

Procedures are invoked with a CALL_statement, and use parameters to accept values and return values to the calling environment. Procedures can also return result sets to the caller. Procedures can call other procedures and fire triggers.

Triggers are associated with specific database tables. They are invoked automatically (fired) whenever rows of the associated table are inserted, updated or deleted. Triggers do not have parameters and cannot be invoked by a CALL statement. Triggers can call procedures and fire other triggers.

User-defined functions are one kind of stored procedure that returns a single value to the calling environment. User-defined functions do not modify parameters passed to them. They broaden the scope of functions available to queries and other SQL statements.

Syntax Checker

Helps you identify problems that the compiler would find, but while you are writing code in the Source Editor. This can save time and assist with troubleshooting.

T

thread

Path of execution in a running application. For example, an application can have threads that handle background processes that aren't visible to the user.

Threads window

Debugger window that displays all the existing threads your program has created and the state of each thread. You can pause individual threads, which causes their execution to cease temporarily while all other threads continue to execute, and resume them. This helps you check for and resolve thread synchronization errors, where more than one thread is in contention for the execution of a method. Double-clicking a thread in this window updates the Calls window, so it reflects the scoping level of the thread.

To see the Threads window, choose View ▸ Threads.

top-level component

A component that can only appear at the top level in the Objects view of the Project window. Also called a *form*. Applet, Frame, Window, and some dialog components are top-level components. In the Component Library, the Visual Cafe top-level components are all in the Forms group. When you open a top-level component in the Form Designer, the component boundaries are the bounds of the window. You can position other components, such as text, buttons, and graphics, on the top-level component in the Form Designer; these components are contained by the top-level component.

transaction isolation levels

The *isolation level* defines degree to which the operations in one transaction are visible to the operations in a concurrent transaction. Isolation levels prevent some or all inconsistent behavior and can be different for each connection. All isolation levels guarantee that each transaction will execute completely or not at all, and that no updates will be lost. This ensures recoverability at all times, regardless of the isolation level.

There are three types of inconsistency that can occur during the execution of concurrent transactions:

Dirty read — Transaction A modifies a row. Transaction B then reads that row before transaction A performs a COMMIT. If transaction A then performs a ROLLBACK, transaction B will have read a row that was never committed.

Non-repeatable read — Transaction A reads a row. Transaction B then modifies or deletes the row and performs a COMMIT. If transaction A then attempts to read the same row again, the row will have been changed or deleted.

Phantom row — Transaction A reads a set of rows that satisfy some condition. Transaction B then executes an INSERT, or an UPDATE (that generates one or more rows that satisfy the condition used by transaction A) and then performs a COMMIT. Transaction A then repeats the initial read and obtains a different set of rows.

Here are the types of transaction isolation levels:

TRANSACTION_NONE — Transactions are not supported.

TRANSACTION_READ_UNCOMMITTED — Lets dirty reads, non-repeatable reads, and phantom reads occur.

TRANSACTION_READ_COMMITTED — Prevents dirty reads, but not non-repeatable reads and phantom reads.

TRANSACTION_REPEATABLE_READ — Prevents dirty reads and non-repeatable reads, but not phantom reads.

TRANSACTION_SERIALIZABLE — Prevents dirty reads, non-repeatable reads and phantom reads.

U

V

ValueTip

Interface element in Visual Cafe that displays the value of the variable that is under the cursor in the Source window in debug mode.

variable

Structure in memory which holds data that has been assigned to it. A variable that is defined in a class defines the class' structure.

- Class variable: Variable associated with a class and not with a particular instance of the class.
- Instance variable: Variable associated with an instance of a class (an object).

Variables window

Debugger window that shows the variables that are active in the current context. To change a variable value at run time, edit its value in this window.

To see the Variables window, choose View ▸ Variables.

view

In a development environment

Window that shows a particular type of information.

- Global view: View that is accessible from the View menu. The global views are: Breakpoints window, Calls window, Class Browser, Component Library, Component Palette, Hierarchy Editor, Output window, Property List, Threads window, Variables window, Watch window.
- Local view: View that is local to a project. The local views are: Project window, Source window.

For Data Sources

A QueryNavigator and a RecordDefinition component, or a RelationViewPlus component, represent a view. A view has three parts:

- It is a collection or set of objects that contain data. You can set focus to one member of the set at a time by scrolling forward and backward through the set.
  - A QueryNavigator component has next and previous methods.
  - A RelationViewPlus component has next and previous methods.
- There is a data model associated with the objects that contain data. Each object has a set of values that can be indexed numerically.
  - A QueryNavigator component has symantec.itools.db.beans.jdbc.RecordDefinition objects. These objects obtain their data model on their own. The Table property is a property of RecordDefinition.
  - A RelationViewPlus component has symantec.itools.db.pro.Record objects. These objects obtain their data model from the SQL statements that define the RelationViewPlus.
- There is a relationship between objects in the view.
  - RecordDefinition components contained by a QueryNavigator component are related in that they meet the select criteria imposed by the Filter property of the QueryNavigator.
  - Records contained by a RelationViewPlus component are related in that they all meet the select criteria in the defining SQL statement.

For detail views, the objects (RecordDefinition or RelationViewPlus' Record) are further related in that they meet the join criteria.

Visual Cafe window

Main development environment window that contains the menu bar and toolbars.

W

Watch window

Debugger window that displays the value of a variable or expression you enter. The values update when you pause execution or step through code. You can also examine the contents of a class member. To watch a variable accessible to a method, drag a variable from the Variables window to the Watch window. You can modify values directly in this window and continue debugging without having to stop and restart the debug session.

To see the Watch window, choose View ▸ Watch.

window

Area that has no borders and no menu bar.

workspace

Saved window arrangement. Your workspace is saved automatically while you work.

X

Y

Z

{mci PLAY NOMENU NOPLAYBAR,MUG4.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,Cupbgr3.bmp}

Click a Topic

Creating Applications

Creating Applets

Creating JavaBeans Components

Using Javadoc Utilities

{ewc HLP25632,HLP256_TILE,CupbgrTile.bmp }

Welcome to the Visual Cafe Online Demo

This demo lets you view commonly performed Visual Cafe tasks through step-by-step procedures and videos. It can help you become familiar with Visual Cafe features, including new features in version 3.0, very quickly.

Videos run best when you have lots of system resources available. For best results, close any programs you do not need to run while viewing the demo. If the videos will not play or you see color problems in the windows, you probably do not have enough system resources available. When you run in 256 color mode, some color schemes cause the demo graphics to appear strangely. In this case, temporarily change to the Windows standard system colors or increase the number of colors on your display.

Now let's get started!

Start Demo

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Creating applications

Visual Cafe provides powerful tools for assembling an application quickly. In general, the development cycle is made of these basic steps. In this demo, we will create a simple application that displays a new image every two seconds.

1. Create a new project. {button Demo, JI(`',`Creating_new_project_video')}

2.	Design the user interface.	{button Demo, JI(`',`Designing_user_interface_video') }
3.	Enhance Java source code.	{button Demo, JI(`',`Application_code_video')}
4.	Test run the application in Visual Cafe.	{button Demo, JI(`',`Application_running_video') }
5.	Deploy the application.	{button Demo, JI(`',`Application_deploying_video')}

Back to Contents

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Creating a new project

To get started quickly, you can use an Application project template, as shown in this video. Visual Cafe provides application project templates for AWT and JFC Swing applications. The application project has a frame, dialog boxes, menu items, and for JFC, toolbars, already set up for you.

Other alternatives are opening an existing project and adding an application to it, copying an existing application as a start, or creating your own project templates.

In this video, we choose File ▸ New Project, open a new JFC Application, and look at what is in the template.

{button Video,JI(`demo.hlp>video',`Crprjatn_avi')}

Back to Creating Applications

{mci PLAY,CRPRJATN.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Enhancing Java source code

When you create your GUI using Visual Cafe tools, Visual Cafe automatically generates Java code in your Java source file. To complete your application, you can manually add code to this source file using the powerful Visual Cafe Source Editor. In addition, the Code Helper, Syntax Checker, and Event Binding dialog box help you quickly add code.

The Source Editor simplifies development with full-color Java syntax and keyword highlighting and an integrated Java macro language for extending the editor. During debugging, you can use the Source window to monitor program execution.

The Code Helper in the Source Editor that interprets the current context and provides either a list of the methods in a given class, a list of the different versions of a particular method, or a list of class objects that begin with a particular character sequence. It also helps you enter Javadoc tags.

The Visual Cafe Syntax Checker lets you identify problems that the compiler would find, but while you are writing code. This can save time and assist with troubleshooting.

Like the Interaction Wizard, the Event Binding dialog box is a way to add skeleton events to your code. After adding an event, it takes you to the event handler in the source code so you can add the code you need to handle the event.

In this video, we use the Event Binding dialog box and Source Editor to create the functionality for the Reset menu item.

{button Video,JI(`demo.hlp>video',`Code_avi')}

Back to Creating Applications

{mci PLAY,CODE.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }

Running the application from Visual Cafe

After you have completed your GUI and code, you are ready to run your application. You can quickly run the application directly from the Visual Cafe environment. Visual Cafe lets you set a variety of run options. If you used one of the Application project templates, you probably do not need to set them before running your application from Visual Cafe.

If you find bugs while running your application, Visual Cafe's advanced debugging tools can help you quickly locate the problem. The integrated graphical debugger provides source-level debugging and lets you browse data and manipulate calls and threads. The Source Editor displays ValueTips so you can quickly view variable values.

In this video, we choose Project ▸ Execute, then use the running application.

{button Video,JI(`demo.hlp>video',`Run_avi')}

Back to Creating Applications

{mci PLAY,RUN.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Deploying your application

The Visual Cafe deployment feature provides many ways to help you quickly deploy your applications. Options include creating JAR, ZIP, CAB, and directory archives, signing, deploying to the local computer or using FTP to send files to a remote computer, compressing, overwriting existing files or not, and adding a JAR to the Component Library.

In this video, we set project options for deployment, then choose Project ▸ Deploy.

{button Video,JI(`demo.hlp>video',`Deploy_avi')}

Back to Creating Applications

{mci PLAY,DEPLOY.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }# Designing the user interface

Visual Cafe provides a wide range of easy-to-use tools to make designing your graphical user interface (GUI) much faster. In general, you follow these basic steps:

1. Add forms to the project. {button Demo, JI(`',`Adding_forms_to_project_video2')}
2. Add components to your forms and arrange them. {button Demo, JI(`',`Adding_components_to_forms_video2')}
3. Modify component properties. {button Demo, JI(`',`Modifying_comp_props_video2')}
4. Create component interactions. {button Demo, JI(`',`Creating_interactions_video2')}
5. Design application menus. {button Demo, JI(`',`Designing_menus_video2')}

Back to Creating Applications

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Adding forms to the project

When you create a new Application project using the template, your project template already has forms, as shown in the video. The application project has dialog boxes, menu items, and for JFC, toolbars, already set up for you. You can also drag forms from the Component Library or use the Insert menu items to quickly add new forms to your project.

{button Video,JI(`demo.hlp>video',`Crprjatn_avi')}

Back to Designing the User Interface

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }

Adding components to forms

Visual Cafe provides a Form Designer, which is a true WYSIWYG form layout tool: it lets you edit all types of forms, including windows and dialog boxes. To design the visual interface of your application, you can drag components from the Component Library or Palette to your form or the Project window. The Form Designer helps you lay out your visual components in the layout you prefer. Invisible components, such as a timer, appear as an icon.

The Component Library is a repository of components and project templates. For fast access, you can configure the Component Palette to contain your favorite components. In addition to the extensive library of components provided with Visual Cafe, you can add third-party components to the Component Library and Palette, and create your own components.

When using JFC Swing, Visual Cafe provides prebuilt model, border, and icon beans to make using JFC easier. On your form, you can switch between different look-and-feels simply by choosing a menu item.

Visual Cafe supports two-way development by translating Java code into a visual representation and by translating the visual representation in the Form Designer to Java code. Your code and your visual model always match.

In this video, we add one visible and one invisible component to a form.

{button Video,JI(`demo.hlp>video',`AddComp_avi')}

Back to Designing the User Interface

{mci PLAY,ADDCOMP.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }# Modifying component properties

The properties of components in your project are displayed in the Property List. To specify the properties of components, you simply fill in fields in the Property List. Visual Cafe updates the component appearance in the Form Designer and the Java code for you.

If a component has a customizer, you can access it from a Property List field or by right-clicking in the Form Designer or Project window.

In this video, we add images to a slide show and set the timer speed.

{button Video,JI(`demo.hlp>video',`ModProps_avi')}

Back to Designing the User Interface

{mci PLAY,MODPROPS.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Creating component interactions

The Interaction Wizard helps you create interactions between components without writing code. The Interaction Wizard lets you graphically build relationships between components in a project, or between a component and itself. These relationships specify the actions to take when an event is triggered on a component. For example, in a slide show, a timer event could cause the next image to display. Visual Cafe automatically generates the necessary code for the specified relationship.

A line with an arrow on your form identifies the interactions you have made. You can delete the lines to delete interactions, double-click a line to modify the interaction in the Interaction Wizard, and choose which interactions you want to display on your form.

In this video, we create an interaction between the timer and slide show, so an image appears every two seconds.

{button Video,JI(`demo.hlp>video',`Inter_avi')}

Back to Designing the User Interface

{mci PLAY,INTER.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Designing application menus

Visual Cafe provides an AWT and Swing Menu Designers as a convenient, visual way to create your menus. The Application project templates have well-developed starter menus that you can modify and add to. In this video, we create a new menu with menu items by using the Swing Menu Designer.

{button Video,JI(`demo.hlp>video',`Menu_avi')}

You can add interactions to the menu items while you are using the Menu Designer, as shown in this video.

{button Video,JI(`demo.hlp>video',`Menu2_avi')}

Back to Designing the User Interface

{mci PLAY,MENU.AVI}

{mci PLAY,MENU2.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp}Creating applets

An applet can add dynamic, interactive functionality to your Web page. Visual Cafe makes it easy for you to develop an applet. You can follow these basic steps:

1. Create a new project. {button Demo, JI(`',`Creating_new_project_applet_video')}
2. Design the user interface. {button Demo, JI(`',`Designing_user_interface_applet_video')}
3. Enhance Java source code. {button Demo, JI(`',`Applet_code_video') }
4. Run the applet in Visual Cafe. {button Demo, JI(`',`Applet_running_video')}
5. Run the applet in the HTML page. {button Demo, JI(`',`Applet_HTML_video')}
6. Deploy the applet. {button Demo, JI(`',`Applet_deploying_video')}

Back to Contents

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }

Creating a new project

To get started quickly, you can use an Applet project template, as shown in this video. Visual Cafe provides applet project templates for AWT and JFC Swing.

Other alternatives are opening an existing project and adding an applet to it, copying an existing applet as a start, or creating your own project templates.

{button Video,JI(`demo.hlp>video',`App1_avi')}

Back to Creating Applets

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }

Enhancing Java source code

When you create your GUI using Visual Cafe tools, Visual Cafe automatically generates Java code in your Java source file. To complete your applet, you can manually add code to this source file using the powerful Visual Cafe Source window. In addition, the Code Helper and Syntax Checker, available from the Source window, box help you quickly add code.

In this video, we add code so you can change the look of the chart through a drop-down list.

{button Video,JI(`demo.hlp>video',`App5_avi')}

Back to Creating Applets

{mci PLAY,APP5.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Running the applet from Visual Cafe

If you used one of the Applet project templates, you probably do not need to set them before running your application from Visual Cafe.

If you find bugs while running your application, Visual Cafe’s advanced debugging tools can help you quickly find your problem.

In this video, we execute the applet and use it in the Applet Runner.

{button Video,JI(`demo.hlp>video',`App6_avi')}

Back to Creating Applets

{mci PLAY,APP6.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Running the applet in an HTML page

As shown in this video, by setting your project options, you can direct Visual Cafe to launch your applet in your default browser so you can test it there.

Visual Cafe supports debugging directly in a browser, which helps you identify problems in the environment where your applet will run.

{button Video,JI(`demo.hlp>video',`App7_avi')}

Back to Creating Applets

{mci PLAY,APP7.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Deploying your applet

The Visual Cafe deployment feature provides many ways to help you quickly deploy your applets. Options include creating JAR, ZIP, CAB, and directory archives, signing, deploying to the local computer or using FTP to send files to a remote computer, compressing, overwriting existing files or not, and adding a JAR to the Component Library.

In this video, we set project options for deployment, then choose Project ▸ Deploy.

{button Video,JI(`demo.hlp>video',`App8_avi')}

Back to Creating Applets

{mci PLAY,APP8.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }# Designing the user interface

Visual Cafe provides a wide range of easy-to-use tools to make designing your graphical user interface (GUI) much faster. In general, you follow these basic steps:

1. Add forms to the project. {button Demo, JI(`',`Adding_forms_to_project_applet_video2')}
2. Add components to your forms and arrange them. {button Demo, JI(`',`Adding_components_to_forms_applet_video2')}
3. Modify component properties. {button Demo, JI(`',`Modifying_comp_props_applet_video2')}
4. Create component interactions. {button Demo, JI(`',`Creating_interactions_applet_video2')}

Back to Creating Applets

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Adding forms to the project

When you created the project, your project template might have already added forms to your project, as shown in the video. You can also drag forms from the Component Library or use the Insert menu items.

{button Video,JI(`demo.hlp>video',`App1_avi')}

Back to Designing the User Interface

{mci PLAY,APP1.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Adding components to forms

You can drag components from the Component Library or Palette to your form or the Project window. The Form Designer helps you lay out your visual components in the layout you prefer. Invisible components, such as a timer, appear as an icon.

In addition to the extensive components provided with Visual Cafe, you can add third-party components to the Component Library and Palette, and create your own components.

In this video, we add JChart, JCombobox, and JLabel components to a form, then arrange them.

{button Video,JI(`demo.hlp>video',`App2_avi')}

Back to Designing the User Interface

{mci PLAY,APP2.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Modifying component properties

You can modify component properties in the Property List. If a component has a customizer, you can access it from the Property List or by right-clicking in the Form Designer or Project window.

In this video, we specify the properties of components in the applet, including adding text to JLabel components and adding models to the JCombobox and JChart components.

{button Video,JI(`demo.hlp>video',`App3_avi')}

Back to Designing the User Interface

{mci PLAY,APP3.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Creating component interactions

The Interaction Wizard helps you create interactions without writing code. A line with an arrow on your form identifies the interactions you have made. You can delete the lines to delete interactions or double-click a line to modify the interaction in the Interaction Wizard.

In this demo, we create an interaction to change the appearance of the JChart when a menu item in the JCombobox is chosen.

{button Video,JI(`demo.hlp>video',`App4_avi')}

Back to Designing the User Interface

{mci PLAY,APP4.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Creating JavaBeans components

You can create JavaBeans components within the Visual Cafe environment. Visual Cafe provides tools to make your job easier. In this demo, we create an invisible label bean.

1.	Create a new project for developing a JavaBeans component.	{button Demo, JI(`',`Bean_Wizard_video')}
2.	Add code to create the bean functionality.	{button Demo, JI(`',`Bean_Code_video')}
3.	Test the bean within a project.	{button Demo, JI(`',`Bean_Project_video')}
4.	Deploy the bean.	{button Demo, JI(`',`Bean_Deploy_video')}

Back to Contents

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Creating a new bean project by using the JavaBean Wizard

The JavaBean Wizard helps you quickly get started creating your JavaBeans component by setting up a skeleton source file containing the bean options you want. For example, you can choose a bean name, type, weight, methods, properties, and icons. The wizard also creates a BeanInfo file for you.

In this video, we create a new bean project using the wizard. You could also add a bean to an existing project by choosing Insert ▸ JavaBean.

{button Video,JI(`demo.hlp>video',`Bean1_avi')}

Back to Creating JavaBeans Components

{mci PLAY,BEAN1.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Adding code to create the bean functionality

The JavaBean Wizard creates a Java source file containing the code you specified, which saves coding time. Tags saying "to do: add code here" help you quickly identify areas that need additional code. While you are adding code in the Visual Cafe Source window , you can use the Code Helper, Syntax Checker, and other tools.

In this video, we use the Source window to enter code into the Java source file created by the JavaBean Wizard.

{button Video,JI(`demo.hlp>video',`Bean2_avi')}

Back to Creating JavaBeans Components

{mci PLAY,BEAN2.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Testing the bean within a project

Visual Cafe greatly reduces the time it takes for you to test beans you have written. After creating a project containing a bean, you can add a test form to the project, then add the bean to the form to test it. Choose Project ▸ Update Project Beans to update the bean on the form. After updating project beans, you can view and modify the bean properties in the Property List, and position it in the Form Designer.

In this video, we add an applet to the bean project, drag the bean to the applet, update project beans, modify bean properties, then run the applet.

{button Video,JI(`demo.hlp>video',`Bean3_avi')}

Back to Creating JavaBeans Components

{mci PLAY,BEAN3.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Deploying the bean

When deploying a bean, you most likely want to place it in a JAR file. Visual Cafe helps you create the JAR, including the manifest file that specifies the contents of the JAR.

You might want to add more than one bean to a JAR. Note that when you add a JAR file to the Component Library, all JavaBeans components in the JAR file are added (as specified in the manifest file). You cannot remove one component independently of the others in the JAR file.

In this video, we deploy by adding a single bean to a JAR and adding the bean to the Component Library.

{button Video,JI(`demo.hlp>video',`Bean4_avi')}

Back to Creating JavaBeans Components

{mci PLAY,BEAN4.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Using Javadoc utilities

Javadoc utilities make documenting your code and obtaining information about packages, methods, and classes during development much easier. In this demo, we cover these features:

1. View Javadoc comments in the Javadoc Editor. {button Demo, JI(`',`Jdoc1_video') }
2. Add comments from the Javadoc Editor. {button Demo, JI(`',`Jdoc2_video') }
3. View Javadoc in the Javadoc Viewer. {button Demo, JI(`',`Jdoc3_video') }
4. View Javadoc for classes and packages. {button Demo, JI(`',`Jdoc4_video') }

Back to Contents

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Viewing Javadoc comments in the Javadoc Editor

The Javadoc Editor can read in Javadoc comments from a file displayed in the Source window, so you can easily view and modify comments in a Java file.

In this video, we view Javadoc comments in a source file, then see the same comments in the editor.

{button Video,JI(`demo.hlp>video',`Jdoc1_avi')}

Back to Using Javadoc Utilities

{mci PLAY,JDOC1.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Adding comments from the Javadoc Editor

While working in the Source window, you can use the Javadoc Editor to quickly add Javadoc tags that document your code. This documentation will appear in HTML files when you generate Javadoc for the file, for example, by choosing Project ▸ Produce Javadoc.

In this video, we add Javadoc comments for a frame and view them in the Source window.

{button Video,JI(`demo.hlp>video',`Jdoc2_avi')}

Back to Using Javadoc Utilities

{mci PLAY,JDOC2.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Viewing Javadoc comments in the Javadoc Viewer

Visual Cafe provides a Javadoc Viewer that lets you quickly locate Javadoc documentation for files, packages, beans in the Component Library, classes, methods, and fields in the Source Editor, and more.

Anytime you choose the View Javadoc command, Visual Cafe generates or updates the Javadoc for that file, if it belongs to an open project.

In this video, we look at the Javadoc comments in a source file then display these comments in the Javadoc Viewer.

{button Video,JI(`demo.hlp>video',`Jdoc3_avi')}

Back to Using Javadoc Utilities

{mci PLAY,JDOC3.AVI}

{ewc HLP25632,HLP256_UPPER_LEFT,CupbgrTile.bmp }Viewing Javadoc for a class and package in the Source window

With the Javadoc Viewer, information about Java packages, classes, methods, and fields is easy for you to obtain during development. You can look up information directly in the Javadoc files on your system, as well as quickly obtain Javadoc for items displayed in the Visual Cafe environment.

In this video, we display Javadoc for the Home location, then for items in the Project window, including an object in the Objects view, a package in the Packages view, and a file in the Files view.

{button Video,JI(`demo.hlp>video',`Jdoc4_avi')}

Back to Using Javadoc Utilities

{mci PLAY,JDOC4.AVI}