

Connecting objects;↵Connecting objects

- 1 **Select an object.**
- 2 **Control-drag a connection to another object.**
- 3 **In the Inspector panel's Connections display, select an outlet or action.**
- 4 **Click the Connect button.**

In an object-oriented application, isolated objects have little value; they need to send messages to each other to get the work of the application done. Interface Builder gives you a way to establish connections between objects.

When you Control-drag between two objects, the Inspector panel becomes the key window. Its Connections display shows the current and potential connections for the destination object.

_ConnectingObjects1.eps ↵

If the Connect button doesn't become active when you select an outlet or action, you probably have connections locked. See ^aWhen You Don't Want to Disconnect^o in this chapter.

;ConnectionsConcepts.rtf;WhenYouDon'tWanttoDisconnect;↵

Outlet Connections

In the previous example, the connection is made from a *controller* object to a custom object that manages the application to a text field. The controller object (ConverterController) declares several *outlets* as identifiers of destination objects as instance variables.

The example shows a connection between an object in the nib file window Instances display and an object in the interface. You can also make outlet connections between two objects in the Instances display.

_ConnectingObjects2.eps ↵

When you make a connection between objects, the first column of the Connections display shows the source object's outlets (a "source" meaning the object from which a connection line is drawn).

Outlets are destination objects specified as instance variables. Actions are methods that NSControl objects (such as buttons) invoke in another object. See "Communicating With Other Objects: Outlets and Actions" in this chapter for more information. ;ConnectionsConcepts.rtf;linkMarkername CommunicatingWithOtherObjects:OutletsandActions;;

Chapter 6, "Subclassing," describes connecting the outlets and actions of custom objects in the context of creating a class. ;../03_Coding/06_Subclassing/Subclassing.rtf;;-

Action Connections

When you make a connection by dragging a line *from* an NSControl object in the interface—a button, slider, text field, menu command, pop-up list, or matrix—what you are doing is that the destination object is a *target*

and that you can complete the connection by selecting an *action* method.

_ConnectingObjects3.eps -

The destination object in an action connection is frequently a custom object that manages the application or a particular window (controller object).

When you make a connection from an NSControl object, the Inspector panel shows the Connections display for the destination object.

_ConnectingObjects4.eps -

When the user manipulates the NSControl object, such as clicking a button or moving a slider, the action message is sent to the destination object (the target).

See "Compound Objects" in Chapter 3 for descriptions of the interaction between NSControl objects and NSCell objects, and of the role NSMatrix objects play. ;../03_SettingObjectAttributes/SettingAttributesConcepts.rtf;CompoundObjects;;-

Connections Within the Interface

Sometimes you can connect two objects on an interface. These connections can involve both outlets and

actions.

_ConnectingObjects5.eps ~

Connections within an interface can also involve two Application Kit objects. Two examples are interconnecting text fields (so the user can tab from field to field), and connecting a menu command such as Print to an NSText object.

Tip: To enable printing of an NSText object, drag a connection line from the Print menu command (or other NSControl object that initiates printing) and select the **print:** action in the Connections display.

You can connect text fields and form fields so that when the user presses the Tab key, the cursor moves to another field. See ["Enabling inter-field tabbing"](#) in this chapter for information on this procedure.
[;EnablingInterfieldTabbing.rtf](#);↵

Related Concept: [;ConnectionsConcepts.rtf](#);linkMarkername

[StandardObjectsintheInstancesDisplay:File'sOwner,FirstResponder,andFontManager;](#), Standard Objects in the Instances Display: File's Owner, First Responder, and Font Manager