GhostFill as a Second Language

A Guide for HotDocs Users and Developers

Capstone Practice Systems

Preliminary Edition – January 2002

Contents

Purpose and Scope of this Guide

This guide is for current users and developers of HotDocs document assembly applications who are moving to – or considering – GhostFill as an alternative software platform. These two fine products accomplish many of the same purposes using different techniques and vocabularies. Our goal here is to ease the "culture shock" for people who have become familiar with document assembly through the lens of HotDocs, but wish to understand GhostFill as a possible supplement or alternative. Familiarity with HotDocs is accordingly assumed.

GhostFill is substantially different from HotDocs. Learning it is kind of like … well, like learning a second language.

Although we've written this guide with the encouragement and cooperation of GhostFill Technologies, this is not intended as an advertisement for GhostFill. We are great fans of Capsoft and HotDocs, and expect them to remain key players in the document automation business for many years to come. GhostFill represents a formidable competitor, though, whose products and services will in some circumstances prove more attractive. It is our belief that this healthy competition is good for the industry, for the users, and for the vendors themselves.

Except where otherwise stated, our references are to the latest versions of the two products: HotDocs 5.3 and GhostFill 4.0. The online versions of these products (HotDocs Online and GhostFill Server) will be addressed in a later version of this guide.

The User Experience

Word processing environment

Unlike HotDocs, GhostFill does not directly support WordPerfect. It has superb RTF support, though, so templates can be written that produce documents that WordPerfect will convert faithfully. Users in a WordPerfect shop will need to run GhostFill independently of their word processor, and template authors will typically need to do their work in Word.

GhostFill supports HTML and text templates, as well as dot and rtf.

GhostFill is more pervasively visible in the Microsoft Word environment than is HotDocs, which ordinarily contents itself with a single red D, and a toolbar only when editing templates. GhostFill puts a special menu on the menu bar, as well as a toolbar (actually three contiguous toolbars.) The toolbar appears by default whether or not you are currently using GhostFill, and whether you are editing or just using a template. GhostFill also adds several commands to the context-sensitive right-click menu in Word.

Applications vs. Libraries

Rather than seeing a list of items in a HotDocs library, GhostFill users access templates within "applications" via the GhostFill Explorer interface. (As with HotDocs, specific GhostFill templates can also be launched directly, bypassing the Explorer.)

Applications are ways of grouping templates and their related objects. An application is a collection that includes documents that have been assembled from its templates, as well as saved answer sets. Each of these kinds of application objects are stored in designated folders based on "mappings."

Dialogs

A GhostFill dialog is a data capture form that you can fill in and save. An optional question tree pane appears at the left, offering random access to different points in the current dialog and its subsidiaries. HotDocs Outline mode offers comparable navigability, but requires special set up and has some frustrating limitations (such as not being dynamic, not handling repeats well, and misbehaving in some circumstances.)

Answer Sets vs. Answer Files

GhostFill answers are saved in bundles ("answer sets") that correspond to the dialogs into which corresponding questions have been organized. Saving or pulling up all the answers involved in creating one or more documents for a given matter therefore might involve accessing several different answer sets, unless all of the associated dialogs have been embedded within a single master dialog. A GhostFill answer set also contains the questions and logic of a dialog itself, and is in fact an instantiated copy of the dialog. It can accordingly be passed among users in work flow applications independent of any document templates. But this different approach to answer management is a frequent source of confusion for users with a HotDocs background.

The Developer Experience

The HotDocs developer will find many familiar concepts and terms in GhostFill. There are templates, dialogs and variables. There are types of variables. There are clauses. There are buttons on the word processor toolbar that can be used to invoke GhostFill and its tasks. But there are also various "explorers," "editors", "managers," and "wizards" that will take some getting used to.

GhostFill Fillpoints

When you create a template in GhostFill, all of the automation manifests in the document in what are called "fillpoints". This term covers the full range of things in GhostFill that are the functional equivalent of HotDocs variables, ask statements, repeat statements, set statements, if/endif blocks, inserted clauses and templates blocks. In other words, a fillpoint can contain variables or commands.

GhostFill uses the notation %[*contents*], where variables and commands can be substituted for *contents.* Everything between the brackets is interpreted by GhostFill's

assembly engine as something that needs to be processed by it. Using the same convention, one would say that HotDocs uses the notation «*contents*» to refer to a variable or command. Unlike HotDocs, where only one element (variable or command) can be inserted between the chevrons, GhostFill allows multiple statements, separated by semi-colons.

Fillpoint Editor

In place of the blue buttons on the HotDocs toolbar used to insert variables, ASK statements, IF..ENDIFs, inserted templates, and clauses, GhostFill offers a utility called the "Fillpoint Editor". This can be invoked from a GhostFill button or the Explorer. It is a *very* elegant tool for composing, editing, inserting, and understanding fillpoints.

Organization and Storage of Definitions (No Component File)

GhostFill does not use a single file to store all of the definitions of components used in a template. In addition, the definitions are independent of the template itself. If you create a dialog, for example, you can refer to that dialog in any template you create. Templates can share defined elements (variables, dialogs, etc.) at will. There is no concept akin to "pointing" component files or copying elements from one to another, as is required in HotDocs for definitions to be shared.

GhostFill definitions are actually stored in XML files, which can be located locally or on a server. GhostFill just has to know where to find the appropriate definition in order to use it. Variable, dialog, clause, and script (the functional equivalent of a computation) definitions all have their own separate storage locations. You can associate a particular definition with an "Application" (see "Applications vs. Libraries", above), but this is an organizational convenience only. There is nothing to prevent you from using a defined element that is associated with one Application in a template that is associated with another.

Defining Dialogs

There are a number of respects in which defining a dialog is different from HotDocs. For one, there is no drag and drop interface for adding or deleting variables. All operations, except for the relative placement of variables and other elements, are carried out using the toolbar or the menu.

Another is the close relationship between variables and dialogs and the fact that variables can only be created when defining a dialog (see "Creating and Referring to Variables", below). Yet another is the close association between dialogs and answer sets, the equivalent of the HotDocs answer file (see "Answer Sets vs. Answer Files", above).

The way inserted dialogs are handled introduces some new concepts. One difference from HotDocs is that the inserted (or "sub-dialog" in GhostFill parlance) dialog needs to be created before you can insert it. You need not define it entirely—it can just be an empty dialog—but it must be defined in advance.

For sub-dialogs, GhostFill's concepts of “embedded” vs. “linked” dialogs is another departure. Embedded dialogs are what the HotDocs developer would think of as true inserted dialogs. The values of variables are stored in the same answer set as the parent dialog. Linked dialogs are different. They have their own answer sets. Think of them as the functional equivalent of HotDocs picklists.

In addition to variables, “labels” can be inserted in dialogs. These are the equivalent of HotDocs “Additional text”. The contents of labels can be manipulated by dialog or variable scripts (see “Scripts vs. Computations” below).

Creating and Referring to Variables

In GhostFill, variables are created in the context of the dialog in which they are asked. There is not, as in HotDocs, a variable that exists independently of the dialog definition. You can use a GhostFill “Query” command to prompt the user for a value and insert it in a document without creating a variable. You can also create a named fillpoint without a variable, which will cause GhostFill to stop and allow the user to enter a value for the fillpoint; no prompt is displayed, however, and the answer is not saved.

To create a true variable in GhostFill, you need to first create a dialog and add the variable to the dialog. You create the variable in the process of editing the dialog. When you refer to the variable in the template, at least initially, you need to refer to the variable using the dialog name. Thus, for example, if you have a dialog named “Client” and a variable called “Name”, you need to refer to “Client.Name”. If you are referring to a variable in a sub-dialog, your initial reference must be a full reference path. For instance, if your “Client” dialog were a sub-dialog of “Case”, you would need to refer to “Case.Client.Name”.

You can create a shorthand reference to any part of the full reference path by making a simple assignment in a fillpoint. For example, you could insert “%[ClientInfo=“Case.Client”] and then refer to”%[ClientInfo.Name]”. You could also insert “%[ClientName=Case.Client.Name]” and then refer to “%[ClientName]”. GhostFill's documentation actually refers to these shorthand references as “variables”.

Lists vs. Multiple Choice Variables

Most HotDocs variable types (text, number, true/false, etc.) have obvious counterparts in GhostFill. You will not find a multiple choice variable type in GhostFill, however. GhostFill does have an equivalent, but calls it a “List” rather than classifying it as a variable. Before you can insert a list in a dialog, you need to define it, much as you need to define sub-dialogs. (But you can create a new list on-the-fly from within the dialog editor.)

Lists can accomplish the same purposes as multiple choice variables, and be used in similar ways, but function differently. List definitions have two columns. The first, “Elements”, is more or less the equivalent of the HotDocs multiple choice “Options”. When the user is prompted for a selection from the list, items in the “Elements” column will be displayed.

The second, or “Results” column, is similar to the “merge text” in HotDocs. It functions differently, however. You can hard code this column with text to be inserted in the template. If you do, the same text will merge no matter where in the template the list is inserted. “Results” can also include, however, any number of fillpoints. Your options are virtually unlimited. There are a number of ways you can achieve the same functionality as merge text. One is to set a variable. For example, if the “Elements” contain a list of names, and you want to insert either a name or title in the document, “Results” column for a particular entry might read “%[Name=“John”]%[Title=“Developer”]. In the template, “%[Name]” would resolve to the name chosen, whereas “%[Title]” would resolve to a job title.

Scripts vs. Computations

Another variable type you won't find in GhostFill is a computation. The equivalent in GhostFill is the script, which functions in many similar ways. Like the FillPoint Editor, GhostFill’s Script Editor is quite powerful, with automatic coloring and syntax detection.

Dialog Scripting and Validation

GhostFill offers much more granularity when it comes to validation of variable answers and dialog scripting. GhostFill also lets you dynamically hide elements on a dialog while the dialog is being displayed.

You can trigger GhostFill scripts at both the dialog and field level. At the dialog level, you specify that the script executes on load, unload, entry, and exit. At the field level, scripts can trigger on entry, exit or change.

Unlike HotDocs, which has a limited ability to display more than one variable on the same horizontal line, all fields and labels in a GhostFill dialog are stacked in a single vertical sequence.

Converting Documents to Templates

When creating a template in HotDocs, it is often convenient to start with the document itself, creating variables on the fly and inserting them in the document before creating other elements. With GhostFill, various elements (dialogs, variables, lists, etc.) need to be created first. A “quick and dirty” template is a real option in HotDocs; the reverse is true with GhostFill.

When you test a template by filling it while open for editing, the template processes itself, rather than creating a new document. You have to be careful not to save the result and lose the template.

Customized Integration With Other Applications

If you have a database or homegrown application you need to tie into a document assembly engine, both HotDocs and GhostFill offer you some options.

In one common scenario, the document assembly program is used simply to pull existing data from a database. The data is incorporated in the assembled document. You can do this with the HotDocs Database Connection, a relatively inexpensive add-on. Similar functionality comes built in to GhostFill.

A developer with programming skills can also create a custom application that interacts with the user and then fires the document assembly engine to produce a document. To do so with HotDocs, you need to purchase the HotDocs Integration Kit. This documents HotDocs function calls, allowing you to write code that sends data to HotDocs and triggers assembly of a document. GhostFill ships with ActiveX components that can be used to develop applications that assemble templates created in RTF, text or HTML format. There is no ActiveX component for assembly of Word (.dot format) templates.

Since the entire GhostFill environment is exposed as a set of COM objects, it should theoretically be possible to write applications to do anything that can be done within GhostFill, including assembling Word templates. At present, however, there is limited documentation available.

Some other considerations

GhostFill is presently *more technically elegant* than HotDocs. It has been designed, from the ground up, on Microsoft's COM Automation. It can act as both a COM server and a COM client. As a result, it is more extensible than HotDocs and application integration is generally easier. Database connectivity is built-in. Developers can easily extend and customize GhostFill by integrating their own custom automation servers into the GhostFill environment

GhostFill is also *more technically demanding* in some respects than HotDocs, and non-trivial template development can require more comfort with arcane syntax than many non-programmers have. Its object-oriented features are powerful but can be bewildering for avocational or occasional template developers.

HotDocs presently has an overwhelming edge in *market share*. It has a huge installed base, a history of national user conferences, a very active Internet discussion service, and regional user groups. Dozens of independent consultants provide system development and training services. Major publishers have used CAPS and HotDocs as platforms for off-the-shelf document assembly systems, and many case management and other commercial software packages affirmatively support integration with HotDocs. Third party resources like the Computation Archive has emerged.

On the other hand, despite its longevity GhostFill is charged with fresh energy and enthusiasm, has considerable human and financial resources, and is building a formidable network of resellers and other early adopters in North America. It has started a listserv, and has a promising knowledge base on its web site. Even though it’s got a “deep bench” of its own (in terms of long-term staff in South Africa), it seems to have the nimbleness and entrepreneurial characteristics of a startup.

GhostFill presently lacks built-in integration with document management systems (such as iManage and PC DOCS)

GhostFill's new PDF filler provides good support for Acrobat-readable graphical forms (which HotDocs has notoriously had problems with), but is no match for the rich functionality of HotDocs Automator when it comes to complex forms with automated overflow handling etc.

Both GhostFill and HotDocs of course are evolving. HotDocs 6, for instance, expected sometime this year, will likely introduce significant changes.

Conversion Strategies

If you find yourself needing or wanting to convert existing HotDocs templates into corresponding GhostFill applications, several concepts and techniques will come in handy. These will be reviewed in a later edition of this guide.

GhostFill Technologies has a macro available that assists with the conversion process.

Many conversion techniques also work in reverse, by the way. That is, people knowledgeable in both GhostFill and HotDocs will find it reasonably straightforward to transform GhostFill templates into HotDocs templates by following similar macros. Since GhostFill puts even more of its application logic on the surface of its templates, arguably it is even easier to go in reverse.

Capstone Practice Systems, http://www.capstonepractice.com, provides software development and training services to a wide range of law firms, legal departments, publishers, government agencies, and other organizations. It has special expertise in (and is a certified reseller of) CAPS, HotDocs, and GhostFill, but is an independent consulting firm with good relations with other vendors. Its principals are world leaders in the comparative assessment of document automation tools and their successful deployment.

Other independent GhostFill specialists can be found on the Reseller page at http://www.ghostfill.com.

Evolving versions of this guide will be available at http://www.capstonepractice.com/Gfas2nd.pdf.
Comments appreciated. Please send them to ghostfill@capstonepractice.com.