

Lasso CDML Converter

A comparison of FileMaker® Pro's CDML and OmniPilot's Lasso languages for custom Web publishing.

Documentation of the Lasso CDML Converter for Adobe® GoLive® CS or Macromedia® Dreamweaver® MX 2004.



Trademarks

Lasso, Lasso Professional, Lasso Studio, LDML, Lasso Service, Lasso Connector, Lasso Web Data Engine, and OmniPilot are trademarks of OmniPilot Software, Inc. MySQL™ is a trademark of MySQL AB. FileMaker® Pro and related product and component names are trademarks of FileMaker, Inc. Adobe® GoLive® CS and related product and component names are trademarks of Adobe Systems, Inc. Macromedia® Dreamweaver® MX and related product and component names are trademarks of Macromedia, Inc. All other products mentioned may be trademarks of their respective holders.

Third Party Links

This paper may contain links to third-party Web sites that are not under the control of OmniPilot. OmniPilot is not responsible for the content of any linked site. If you access a third-party Web site mentioned in this guide, then you do so at your own risk. OmniPilot provides these links only as a convenience, and the inclusion of the links does not imply that OmniPilot endorses or accepts any responsibility for the content of those third-party sites.

Copyright

Copyright © 2005 OmniPilot Software, Inc. This paper may not be copied, photocopied, reproduced, translated or converted to any electronic or machine-readable form in whole or in part without prior written approval of OmniPilot Software, Inc.

Second Edition: April 14, 2005

OmniPilot Software, Inc.
1815 Griffin Road
Dania Beach, Florida 33004
U.S.A.

Telephone: (954) 874-3100
Email: info@omnipilot.com
Web Site: <http://www.omnipilot.com>

Contents

Chapter 1	
Introduction	5
Background	5
Chapter 2	
Lasso CDML Converter	7
Requirements	7
Installation	8
Converting a Single Page	10
Batch Converting a Folder	11
What is Converted	11
Chapter 3	
CDML/Lasso Script Conversion	13
Terminology	13
Database Actions	14
Variable Tags	16
Field Inputs	20
Replacement Tags	20
Looping Tags	23
Conditional Expressions	24

Chapter 4	
CDML/Lasso Script Equivalents	27
<i>Table 1: Database Action Tags</i>	27
<i>Table 2: Variable Tags</i>	28
<i>Table 3: Looping Tags</i>	29
<i>Table 4: Replacement Tags</i>	30
<i>Table 5: Encoding Keywords</i>	31
<i>Table 6: Intratag Parameters</i>	32
<i>Table 7: Conditional Symbols</i>	33
Chapter 5	
Tools and Resources.	34

1

Chapter 1

Introduction

This paper provides an overview of the differences between FileMaker Pro's language for custom Web publishing, namely CDML, and OmniPilot's language for custom Web publishing in Lasso Professional, namely Lasso Script.

The emphasis in this paper is on providing a quick introduction to Lasso Script for anyone who has created a site in FileMaker Pro using the Web Companion and CDML and now wants to move that site over to Lasso. This paper assumes that the reader knows basic CDML terminology.

This paper includes chapters that detail the following topics:

- Documentation of the Lasso CDML Converter. This free tool installs in Adobe GoLive CS or Macromedia Dreamweaver MX 2004. It converts individual files or even a whole folder of files from CDML syntax to Lasso Script syntax automatically.
- A comparison of CDML and Lasso Script with an emphasis on similarities between the two languages and tips for converters.
- A set of tables providing a direct translation from every CDML tag to its Lasso Script equivalent. Notes about any tags that don't have direct translations.

This paper assumes that FileMaker Pro will be used as the back-end for the completed site in Lasso. OmniPilot also provides an FMP2MySQL Conversion Kit that details how to transfer a site from a FileMaker Pro back-end to MySQL.

<http://support.omnipilot.com/article.lasso?id=5142003608>

Background

FileMaker Pro is a desktop database application available for Mac OS X and Windows. FileMaker Pro provides graphical user interfaces for all database administration tasks including defining databases, laying out user

interfaces, searching, and creating reports. FileMaker Pro is primarily used as a rapid application development (RAD) environment in which custom applications with desktop user interfaces can be created and deployed either stand-alone or using FileMaker Server.

FileMaker Web Publishing

FileMaker Pro versions 4 through 6 provide a plug-in called the Web Companion which enables FileMaker Pro to be used as a Web application server. The Web Companion can be used to enable Instant Web Publishing which automatically attempts to recreate the user-interface of a FileMaker Pro database or Custom Web Publishing which allows custom format files to be created in the language CDML (Claris Dynamic Markup Language).

Lasso Professional provides a built-in data source connector which allows connectivity with FileMaker Pro versions 4 through 6 using the Web Companion as a back-end API. Due to licensing restrictions FileMaker Pro Unlimited is required for Web serving with FileMaker versions 5 and 6.

FileMaker Pro 7 supports only Instant Web Publishing. There is no provision for Custom Web Publishing, CDML support, or Lasso support. Instead, a new Web Publishing Engine is integrated into FileMaker Server Advanced 7.

Lasso Professional provides a built-in data source connector which allows connectivity to FileMaker Server Advanced 7 through the Web Publishing Engine API. It is not possible to connect Lasso to FileMaker Pro 7.

Lasso Professional

Lasso Professional is a middleware Web Data Engine from OmniPilot Software. Lasso provides a link between back-end database servers and Web-servers allowing applications to be created with HTML interfaces in standard Web browsers. Web sites are created by programming Lasso Script. Lasso Professional includes a built-in copy of SQLite and has database connectors for FileMaker Pro, MySQL, and JDBC compliant databases.

Both Lasso Professional and Lasso Studio include free tools that allow FileMaker Pro and MySQL databases to be manipulated through a Web interface. Lasso Administration includes the Lasso Database Browser and Lasso Database Builder.

2

Chapter 2

Lasso CDML Converter

Lasso CDML Converter can be used to automatically convert either single pages or entire folders of CDML format files to Lasso Script format files automatically. The converter is included in Lasso Studio 8 and is also shipped as a separate plug-in for Adobe GoLive CS or Macromedia Dreamweaver MX 2004.

This section includes the system requirements for the converter, installation instructions, documentation of how to use the converter on single pages or on folders of CDML files, and details about what syntax the converter handles.

Chapter 3: CDML/Lasso Script Conversion includes details about what is converted automatically by the Lasso CDML Converter and includes instructions for converting sites manually or troubleshooting code which is not converted properly. *Chapter 4: CDML/Lasso Script Equivalents* includes quick reference tables that show the Lasso Script equivalents of all CDML tags, keywords, and comparison operators.

Lasso Studio 8

Lasso CDML Converter is integrated into Lasso Studio 8. This new version of the converter fully support Lasso 8 syntax and provides all the other capabilities documented in this white paper. More information about Lasso Studio 8 (including links to a free 30-day evaluation) can be found at:

<http://www.lassostudio.com/>

If you are using Lasso Studio 8 you can skip forward to the instructions for *Converting a Single Page* or *Batch Converting a Folder* and the subsequent sections.

Lasso CDML Converter is also available as a separate install from Lasso Studio. Installation and usage instructions are provided in the remainder of this chapter.

Requirements

Lasso CDML Converter has the following system requirements depending on whether it is installed into Dreamweaver MX 2004 or Adobe GoLive CS.

Dreamweaver MX 2004

- Macromedia Dreamweaver MX 2004. Earlier versions of Dreamweaver are not guaranteed to work.
- Windows – 600 MHz Pentium III processor or equivalent, Windows 2000/XP/Server 2003, 256 MB RAM, 300 MB available disk space.
- Macintosh – 500 MHz PowerPC G3 processor or better, Mac OS X 10.2.6 and later, 256 MB RAM, 300 MB available disk space.
- Lasso Professional 7 is required in order to process the converted pages.

Adobe GoLive CS

- Adobe GoLive CS 7.0.2. Earlier versions of GoLive are not guaranteed to work.
- Windows – 600 MHz Pentium III processor or equivalent, Windows 2000/XP, 256 MB RAM, 300 MB available disk space.
- Macintosh – PowerPC G3 processor or better, Mac OS X 10.2.4 and later, 256 MB RAM, 300 MB available disk space.
- Lasso Professional 7 is required in order to process the converted pages.

Installation

Installation instructions are provided separately for both Macromedia Dreamweaver MX 2004 on Macintosh and Windows and Adobe GoLive CS on Macintosh and Windows.

Note: Consult the documentation of Lasso Studio 8 for installation instructions. If Lasso Studio 8 is already installed then this section may be skipped.

Macromedia Dreamweaver MX 2004 (Macintosh)

- 1 Download the Lasso CDML Converter from OmniPilot's Web site and decompress the archive.
- 2 Locate the Macromedia Dreamweaver MX 2004 Configuration folder. This is usually located at the following path:

/Applications/Macromedia Dreamweaver MX 2004/Configuration

- 3 Copy the files from the Dreamweaver (Macintosh) folder in the Lasso CDML Converter download into the appropriate folders in the Configuration folder. Two files should be placed in Configuration/Floaters and one file in Configuration/Commands.

```
.../Configuration/Floaters/Lasso CDML Converter.htm
.../Configuration/Floaters/Lasso CDML Converter.xml
.../Configuration/Commands/Lasso CDML Converter.htm
```

- 4 Quit and relaunch Macromedia Dreamweaver MX 2004.
- 5 Select the command Lasso CDML Converter from the Commands menu. The Lasso CDML Converter floating palette will appear. Follow the instructions in the next section to use the converter.

Adobe GoLive CS (Macintosh)

- 1 Download the Lasso CDML Converter from OmniPilot's Web site and decompress the archive.
- 2 Locate the Adobe GoLive CS Extend Scripts folder. This is usually located at the following path:

/Applications/Adobe GoLive CS/Modules/Extend Scripts

- 3 Copy the folder Lasso CDML Converter from the GoLive folder in the Lasso CDML Converter download into the Extend Scripts folder. The entire folder should be moved including all of its contained files.

```
.../Extend Scripts/Lasso CDML Converter/Lasso CDML Converter.htm
.../Extend Scripts/Lasso CDML Converter/Lasso CDML Converter.xml
```

- 4 Quit and relaunch Adobe GoLive CS.
- 5 Select the command Lasso CDML Converter from the Windows menu. The Lasso CDML Converter floating palette will appear. Follow the instructions in the next section to use the converter.

Macromedia Dreamweaver MX 2004 (Windows)

- 1 Download the Lasso CDML Converter from OmniPilot's Web site and decompress the archive.
- 2 Locate the Macromedia Dreamweaver MX 2004 Configuration folder. This is usually located at the following path:

C:\Program Files\Macromedia\Dreamweaver MX 2004\Configuration

- 3 Copy the files from the Dreamweaver (Windows) folder in the Lasso CDML Converter download into the appropriate folders in the Configuration

folder. Two files should be placed in Configuration/Floaters and one file in Configuration/Commands.

```
...\\Configuration\\Floaters\\Lasso CDML Converter.htm
...\\Configuration\\Floaters\\Lasso CDML Converter.xml
...\\Configuration\\Commands\\Lasso CDML Converter.htm
```

- 4 Quit and relaunch Macromedia Dreamweaver MX 2004.
- 5 Select the command Lasso CDML Converter from the Commands menu. The Lasso CDML Converter floating palette will appear. Follow the instructions in the next section to use the converter.

Adobe GoLive CS (Windows)

- 1 Download the Lasso CDML Converter from OmniPilot's Web site and decompress the archive.
- 2 Locate the Adobe GoLive CS Extend Scripts folder. This is usually located at one of the following paths depending on whether GoLive CS was installed standalone or as part of the CS suite.

```
C:\\Program Files\\Adobe\\GoLive_ENG\\Modules\\Extend Scripts
C:\\Program Files\\Adobe\\GoLive_CS\\Modules\\Extend Scripts
```

- 3 Copy the folder Lasso CDML Converter from the GoLive folder in the Lasso CDML Converter download into the Extend Scripts folder. The entire folder should be moved including all of its contained files.

```
...\\Extend Scripts\\Lasso CDML Converter\\Lasso CDML Converter.htm
...\\Extend Scripts\\Lasso CDML Converter\\Lasso CDML Converter.xml
```

- 4 Quit and relaunch Adobe GoLive CS.
- 5 Select the command Lasso CDML Converter from the Windows menu. The Lasso CDML Converter floating palette will appear. Follow the instructions in the next section to use the converter.

Converting a Single Page

A single page can be converted directly within Adobe GoLive CS or Macromedia Dreamweaver MX.

Important: The converter should always be used on a backup of your site format files. When testing the converted format files you should always use a backup of your database until you are sure they are working properly.

- 1 Open the CDML Converter by choosing its name from the Windows menu in GoLive or from the Commands menu in Dreamweaver.
- 2 Select Single File from the pop-up menu in the floating palette.

- 3 Click on the Convert button
- 4 The file is converted in place and any warnings are displayed in a dialog box.

Batch Converting a Folder

An entire folder for format files can be converted within Adobe GoLive CS or Macromedia Dreamweaver MX. Lasso CDML Converter will convert CDML format files that have one of the following extensions:

.html	.htm	.lasso	.las
.incl	.inc	.txt	

Important: The converter should always be used on a backup of your site format files. When testing the converted format files you should always use a backup of your database until you are sure they are working properly.

- 1 Open the CDML Converter by choosing its name from the Windows menu in GoLive or from the Commands menu in Dreamweaver.
- 2 Select Choose Folder... from the pop-up menu in the floating palette. A dialog box will ask you to choose the folder to convert
- 3 Click on the Convert button
- 4 The folder will be scanned for appropriate files to convert. Each file is converted in place and backed up if any changes were made. A file Lasso CDML Converter.txt is created in the source folder which describes what files were converted and includes warnings that were generated during the conversion.

What is Converted

This section documents what is converted by the Lasso CDML Converter and how to manually update any tags or syntax that are not automatically updated.

The following two chapters also provide additional details about how to convert sites from CDML to Lasso Script. *Chapter 3: CDML/Lasso Script Conversion* includes details about what is converted automatically by the Lasso CDML Converter and includes instructions for converting sites manually or troubleshooting code which is not converted properly. *Chapter 4: CDML/Lasso Script Equivalentents* includes quick reference tables that show the Lasso Script equivalentents of all CDML tags, keywords, and comparison operators.

The following tags and syntax types are automatically converted:

- The FMPPro CGI target is converted to Action.Lasso in HTML form actions and URLs.
- **Database Action Tags** (-edit, -find, -view, etc.) included in HTML form inputs (including hidden inputs, text and password inputs, select lists and pop-up menus, and textareas), URLs, and inline tags are converted to their Lasso Script action tag equivalents.

Note: The -dbopen, -dbclose, -dbnames, -layoutnames, and -scriptnames tags must all be converted manually. See the section on *Database Action Tags* in the following chapter for details.

- **Variable Tags** (-db, -lay, -sortfield, etc.) included in HTML form inputs (including hidden inputs, text and password inputs, select lists and pop-up menus, and textareas), URLs, and inline tags are converted to their Lasso Script command tag equivalents.

Note: The -errorfmtfield, -errornum, -mail..., -modid, -stylehref, and -styletype tags must all be converted manually. See the section on *Variable Tags* in the following chapter for details.

- **Looping Tags** ([FMP-Inline] ... [FMP-Inline], [FMP-Record] ... [FMP-Record], [FMP-Portal] ... [FMP-Portal], etc.) are converted to their Lasso Script container tag equivalents.

Note: The CDML [FMP-ValueList] ... [FMP-Value_List] tags behave differently when translated to Lasso Script [Value_List] ... [Value_List] tags. The [FMP-ValueNames] ... [FMP-ValueNames] tags do not have an Lasso Script equivalent. See the section on *Looping Tags* in the following chapter for details.

- **Replacement Tags** ([FMP-Field], [FMP-Include], [FMP-Cookie], etc.) are converted to their Lasso Script substitution tag equivalents. **Encoding Keywords** are converted to their Lasso Script equivalents. Any **Intratag Parameters** are automatically converted to substitution tags surrounded by parentheses. Field names and literal string values are automatically enclosed by single quotes.

Note: The CDML [FMP-IncludeField] and [FMP-CurrentModID] tags do not have direct Lasso Script equivalents. CDML has several keywords (display and format) which do not have equivalents in Lasso Script. See the section on *Replacement Tags* in the following chapter for details.

- **Comparison Symbols** (.eq., .cn., .or., etc.) and **Conditional Expressions** within [FMP-If] ... [FMP-If] tags are automatically converted to their Lasso Script equivalents. The converter automatically handles nested boolean

expressions and intratag parameters. Field names and literal string values are automatically enclosed by single quotes.

Note: Conditional expressions should always be double checked for accuracy since it can be difficult for the converter to decide heuristically whether a word is a string literal or a field name. For the purposes of conditional expressions the CDML `.xor` symbol is equivalent to the `!=` symbol in Lasso Script. See the section on *Conditional Expressions* in the following chapter for details.

3

Chapter 3

CDML/Lasso Script Conversion

This chapter documents how to convert CDML format files to Lasso Script. Most of the conversions documented here are performed automatically by the Lasso CDML Converter introduced in the previous chapter. It is recommended that you use the converter if possible.

The instructions in this chapter can be used to perform a conversion from CDML to Lasso Script manually. The chapter starts with a discussion of how CDML terminology maps to Lasso Script terminology. Each type of tag and syntax is then discussed in turn including specific instructions of how to convert the CDML syntax to its Lasso Script equivalent. Each section ends with a discussion of any tags or syntax that will not convert directly.

This chapter should be read in concert with the quick references tables in *Chapter 4: CDML/Lasso Script Equivalents* that show the Lasso Script equivalents of all CDML tags, keywords, and comparison operators.

Terminology

This section documents the terminology of CDML and the Web Companion. Each term's Lasso or Lasso Script equivalent is provided in parentheses.

- **Format Files** – Both CDML and Lasso Script use the term format files to describe text files that contain embedded tags for server-side processing. Lasso Script format files usually have a .lasso extension (or .html). Lasso Script format files usually have a .html extension
- **FMPPro (Action.Lasso)** – The FileMaker Pro Web Companion is invoked through the FMPPro CGI. The format file is referenced through the -format tag <http://www.example.com/FMPPro?-format=format.html>. The equivalent for Lasso is to use Action.Lasso as the CGI and reference the format file through the -Response tag <http://www.example.com/Action.Lasso?-Response=format.lasso>

- **Database Action Tags** – In either CDML or Lasso Script the tag in HTML forms or URLs which specifies what database action to perform.
- **Variable Tags (Command Tags)** – Variable tags are used in HTML forms and URLs in order to specify the parameters for database actions. Command tags are the direct Lasso Script equivalent.
- **Field Inputs** – Field inputs are used in HTML forms and URLs. Both CDML and Lasso Script treat URL parameters, field inputs, select lists and popups, and textareas identically.
- **Replacement Tags (Substitution Tags)** – Replacement tags return a value when a format file is evaluated. Substitution tags are the direct Lasso Script equivalent. Lasso also provides process tags which perform a task, but don't return a value to the page.
- **Intratag Parameters (Sub-Tags)** – CDML defines a limited set of tags that can be used as parameters for other tags. These intratag parameters are enclosed in curly brackets { ... } and only support one layer of nesting. In contrast, any substitution tag in Lasso Script can be used as a parameter to another tag. These sub-tags are enclosed in parentheses (...) and can be arbitrarily nested.
- **Encoding Parameters** – Both CDML and Lasso Script support a range of encoding parameters. In CDML they are specified as the last parameter to certain replacement tags Break, HTML, Raw, URL. In Lasso Script they are specified with a hyphen as a parameter to any substitution tag: -EncodeBreak, -EncodeHTML, -EncodeNone, -EncodeStrictURL, -EncodeXML. In both cases the default if no encoding parameter is specified is to do HTML encoding.
- **Conditional Expression** – Both CDML and Lasso Script support conditional expressions using comparison symbols within the [FMP-If] ... [/FMP-IF] tag and the [If] ... [/If] tag respectively. However, the structure of the expressions is significantly different. See the section on *Conditional Expressions* below for details and examples.

Database Actions

Database actions in CDML can be performed using the same methodology as those in Lasso Script. The names of the database actions are different and the names of the variable tags that specify the parameters of the database action are different, but in general simply replacing the tag names with their equivalents will result in a working Lasso Script solution.

See *Table 1: Database Action Tags* in *Chapter 4: CDML/Lasso Script Equivalents* for a complete list of CDML database action tags and their Lasso Script action tag equivalents.

The basic database actions supported by CDML are listed here:

-Delete	-Dup	-Edit	-Find
-FindAll	-FindAny	-Img	-New
-View			

The Lasso Script equivalents of these database actions are listed here:

-Delete	-Duplicate	-Update	-Search
-FindAll	-Random	-Image	-Add
-Show			

Both CDML and Lasso Script support performing database actions through inline tags, HTML forms, and URLs. In CDML the Web companion is usually invoked by calling a URL like the following. The FMPro CGI name invokes the Web companion. The port 591 is only required if the Web companion is serving on a different port than the HTTP default of 80.

`http://www.example.com:591/FMPro?...`

In Lasso, an equivalent URL for invoking Lasso follows. The Action.Lasso name invokes Lasso. A port is rarely required to invoke Lasso since it runs within the Web server directly.

`http://www.example.com/Action.Lasso?...`

See the following section on *Variable Tags* for a description of how the parameters to the database actions can be specified in CDML and Lasso Script. See the sections on *Open/Close Tags* and *Schema Tags* for a discussion of some CDML actions like `-dbopen` and `-dbnames` that are not supported in Lasso Script and how to simulate them.

Note: In Lasso the user of the equivalent command tags is supported (and called the Classic Lasso methodology), but deprecated in favor of using `[Inline]` operations. For the purposes of this paper it will be assumed that the converted site will use a mix of command tags and `[Inline]`s as were found in the CDML solution.

To convert database action tags from CDML to Lasso Script:

The Lasso CDML Converter tool included with this paper is the easiest way to convert a site from CDML to Lasso Script. If you want to perform the conversion manually or want to double-check the automatic conversion the following steps are required:

- 1 Search your CDML format file database action tags in inline tags, HTML forms, or URLs.

- 2 Change the tag name to the Lasso Script equivalent using the list in *Table 1: Database Action Tags* in *Chapter 4: CDML/Lasso Script Equivalents*.
- 3 Convert the HTML form actions or URLs on the page from the FMPPro CGI call to Action.Lasso. If necessary, remove the port from the URLs.

Open/Close Tags

The database action tags `-DBOpen` and `-DBClose` allow FileMaker Pro databases to be opened and closed through the Web Companion. These tags are not directly supported by Lasso, but can be simulated by sending a URL directly to the Web Companion for processing.

For example, if the address of the FileMaker Pro machine is `fmp.example.com` and it is serving on port 591 then a `-DBOpen` command for the database `Contacts.FP5` can be executed as follows. The example includes a password which will be used to log in to the opened database.

```
[Include_URL: 'http://fmp.example.com:591/FMPPro?-DBOpen=Contacts',
  -Password='Example']
```

Schema Tags

The database action tags `-DBNames`, `-LayoutNames`, and `-ScriptNames` allow the schema of FileMaker Pro databases to be examined through the Web Companion. These tags are not directly supported by Lasso, but the following tags provide an equivalent of the `-DBNames` and `-LayoutNames` tags.

```
[Database_Names]
  <br /><b>[Database_Nameltem]</b>
  [Database_TableNames] [Database_TableNameltem] [/Database_TableNames]
[/Database_Names]
```

There is no way equivalent in Lasso of the `-ScriptNames` tag, but the same technique used for `-DBOpen` above could be used to acquire these values.

Variable Tags

Variable tags in CDML are equivalent to command tags in Lasso Script. Both types of tags can be used in inlines, HTML forms, or URLs to specify the parameter of a database action.

See *Table 2: Variable Tags* in *Chapter 4: CDML/Lasso Script Equivalents* for a complete list of CDML variable tags and their Lasso Script command tag equivalents.

For example, a `-DB` tag in CDML is equivalent to a `-Database` tag in Lasso Script. A CDML URL might look like this:

```
http://www.example.com:591/FMPro?-Findall&-DB=Contacts&-Format=format.htm
```

The Lasso Script equivalent would be as follows:

```
http://www.example.com/Action.Lasso?-FindAll&-Database=Contacts&-Response=format.htm
```

Similarly, an HTML form may start with a series of hidden inputs that specify a database action. In CDML such a form might start like this:

```
<form action="http://www.example.com:591/FMPro" method="POST">
  <input type="hidden" name="-FindAll" value="">
  <input type="hidden" name="-DB" value="Contacts">
  <input type="hidden" name="-Format" value="format.htm">
  ...
</form>
```

The Lasso Script equivalent would be as follows:

```
<form action="http://www.example.com/Action.Lasso" method="POST">
  <input type="hidden" name="-FindAll" value="">
  <input type="hidden" name="-Database" value="Contacts">
  <input type="hidden" name="-Response" value="format.htm">
  ...
</form>
```

Format Files

CDML supports a number of special arguments to the `-format` variable tag which serve data in predefined formats. This section includes details about how to recreate each of these formats in Lasso Script.

The various XML formats (`-dso_xml`, `-dso_xml_dtd`, `-fmp_xml`, `-fmp_xml_dtd`, etc.) can be created using Lasso (with data from any data source) using the Examples provided with the Lasso 7 Language Guide. The FileMaker examples should be copied into your Web server root and then referenced as normal format files. For example:

```
-Response=/Examples/FileMaker/dso_xml.lasso
```

A field can be used as the format file for a database request by specifying it using `-Response=field:FieldName` in Lasso Script. You must go into Lasso Administration and allow the field to be used as a response in the **Setup > Data Sources > Field** section.

The `-raw` format which is used to pull data from the Web Companion without the overhead of XML tags can be easily recreated in Lasso Script, but this format is beyond the scope of this paper.

Error Pages

CDML has a rich set of error control tags which differ markedly from the error controls that are provided in Lasso Script. The `-Error` variable tag has a direct analogue in the Lasso Script `-ResponseAnyError` command tag. A site that makes use of this tag will convert to Lasso Script directly without any further modifications.

The `-ErrNum` tag has no direct analogue in Lasso Script. However, Lasso Script does provide a set of tags which allow different error pages to be displayed for different errors. For example, `-ResponseAddError` returns the specified page if an error occurs while performing an `-Add` action. Other command tags for particular errors include:

```
-ResponseAddError           -ResponseAnyError
-ResponseRequiredFieldMissingError -ResponseSecurityError
-ResponseUpdateError
```

Lasso Script also does not provide for an `Error` folder like that which CDML provides. Instead, Lasso allows you to create one file `Error.Lasso` at the root of your Web serving folder and use it to serve custom error messages. To recreate the functionality of the `Error` folder you can check the value of `[Error_CurrentError]` and then serve the appropriate file from your `Error` folder..

Sending Email

The `-Mail...` tags in CDML had a direct analogue in earlier versions of Lasso Professional, but these tags have been eliminated in Lasso Professional 7. When converting a site from CDML to Lasso Script for any version of Lasso it is recommended that you convert any use of the `-mail...` tags to the `[Email_Send]` tag instead. The `[Email_Send]` tag should be on the response to the database action in which the `-email...` tags were used.

The `[Email_Send]` tag is used as follows. The parameters to the tag specify the SMTP host, from address, to address (and optional cc and bcc addresses), subject, and body of the message. For example, the following tag would send a simple message that is specified entirely within the tag.

```
[Email_Send: -Host='mail.example.com',
  -To='example@example.com',
  -From='example@example.com',
  -Subject='An Example Message',
  -Body='This is an example message!']
```

In order to send a message which uses a format file for the body of the message you can use the `[Include_Tag]` within the `[Email_Send]` tag. For example, the following tag will send a message using the file `format.htm` as the body of the message.

```
[Email_Send: -Host='mail.example.com',
-To='example@example.com',
-From='example@example.com',
-Subject='An Example Message',
-Body=(Include: 'format.htm')]
```

Finally, in order to send a message which uses a field for the body of the message you can use the [Field] within the [Email_Send] tag. For example, the following tag will send a message using the field format as the body of the message.

```
[Email_Send: -Host='mail.example.com',
-To='example@example.com',
-From='example@example.com',
-Subject='An Example Message',
-Body=(Field: 'format')]
```

XSLT Style Tags

Lasso provides significantly more XML processing capabilities than CDML. The -StyleHref and -StyleYype variable tags simply insert an xml-stylesheet directive into a served XML format file. Using the [XML_Transform] tag in Lasso Script a stylesheet can actually be applied to XML data and the result served.

In order to recreate the action of the -StyleHref and -StyleYype variable tags all that is necessary is to insert the appropriate directive into the served XML format file. For example, to insert an XSL stylesheet named style.xml the following directive would be used. The browser will fetch the style sheet and apply it client-side.

```
<?xml-stylesheet type="text/xml" href="style.xml" ?>
```

Use of the [XML_Transform] and other XML tags in Lasso Script is beyond the scope of this paper. Please consult Lasso's documentation for more details about Lasso's extensive XML support.

ModID

One significant feature of CDML that Lasso does not support directly is the modification ID. This is a token which is assigned to each record in FileMaker Pro and incremented each time the record is updated. A Web client can fetch the modification ID using [FMP-ModID] from FileMaker Pro and send it back with an -Edit action. If the modification ID has changed then the edit is disallowed.

Equivalent functionality can be created in Lasso by passing the modification ID to Lasso in a calculation field on the Filemaker Pro layout. A

requirement for an Exact Search can then be established within Lasso's security settings in Lasso Administration in order to require that the value of this field match in order to perform an update.

Field Inputs

CDML and Lasso Script treat field inputs in URLs, HTML forms, inputs, select lists and popups, and textareas identically. No changes should be necessary for most field inputs when converting a site from CDML to Lasso Script.

However, any inputs that reference variable tags in their names must be converted according to the rules for *Variable Tags* above. And, any inputs that reference replacement tags in their values must be converted according to the rules for *Replacement Tags* immediately below.

Replacement Tags

Replacement tags in CDML are equivalent to substitution tags in Lasso Script. CDML tags are a subset of those supported in Lasso Script (with one or two exceptions). In general the corresponding tags are called identically except for the different tag name. See the list below for some things you should watch out for when converting tags.

See *Table 4: Replacement Tags in Chapter 4: CDML/Lasso Script Equivalents* for a complete list of CDML replacement tags and their Lasso Script substitution tag equivalents.

For example the [FMP-ClientUsername] tag, which returns the username of the current site client, is equivalent to the [Client_Username] tag in Lasso Script. Similarly, the [FMP-Field: Field Name] tag is equivalent to the [Field: 'Field Name'] tag in Lasso Script.

There are some issues you should watch out for when converting tags:

- **Tag Parameters** – In CDML the parameters to tags are often not specified with quote marks. Lasso requires that all parameters be quoted. The following tags would be equivalent.

```
[FMP-Field: FieldName]
```

```
[Field: 'FieldName']
```

Lasso requires quote marks because of its flexibility in specifying parameters and nesting substitution tags (intratag parameters). The quote marks make it clear that a literal field name is being referenced rather than a sub-tag, expression, or variable.

- **Intratag Parameters** – CDML defines a subset of replacement tags as intratag parameters that can be used within another tag. An intratag parameter is enclosed in braces { ... } and has the same name as a replacement tag without the FMP- prefix.

For example, you can fetch a field that has the same name as the current user with this code:

```
[FMP-ClientUsername]: [FMP-Field: {ClientUsername}]
```

Lasso is considerably more flexible about using the output of one tag as a parameter for another. Any substitution tag in Lasso can be used as a parameter (or sub-tag) for another tag. Substitution tags use the same name when called directly or when used as a parameter. Sub-tags should be enclosed in parentheses (...).

The same example of fetching a field that has the same name as the current user would look like this in Lasso Script:

```
[Client_Username]: [Field: (Client_Username)]
```

Lasso also defines a set of process tags which function identically to substitution tags, but do not return a value. For example, [Cookie_Set] is a process tag that sets a cookie, but does not return a result. This corresponds to the CDML tag [FMP-SetCookie] which can be used as a replacement tag which doesn't return a value, but cannot be used as an intratag parameter.

- **Encoding Keywords** – Both CDML and Lasso Script automatically HTML encode any results from replacement or substitution tags. They each have additional encoding keywords that can be specified in order to change the output encoding.

CDML supports the keywords Break, HTML, Raw, URL, and others (see below). The keyword is specified after any tag parameters.

```
[FMP-Field: FieldName, Break]      [FMP-Field: FieldName, HTML]
[FMP-Field: FieldName, Raw]       [FMP-Field: FieldName, URL]
```

Lasso Script supports the keywords -EncodeBreak, -EncodeHTML, -EncodeNone, -EncodeStrictURL, and others (see the Lasso Reference for a complete list). The keyword is specified after any tag parameters and always starts with a hyphen.

```
[Field: 'FieldName', -EncodeBreak]  [Field: 'FieldName', -EncodeHTML]
[Field: 'FieldName', -EncodeNone]   [Field: 'FieldName', -EncodeStrictURL]
```

CDML also supports two encoding keywords that are not provided in Lasso Script. The Display and Format keywords do not have a direct analogue in Lasso. Instead, use the default HTML encoding which provides the text within the field without any formatting.

To convert replacement tags from CDML to Lasso Script:

The Lasso CDML Converter tool included with this paper is the easiest way to convert a site from CDML to Lasso Script. If you want to perform the conversion manually or want to double-check the automatic conversion the following steps are required:

- 1 Search your CDML format file for the start of a CDML tag [FMP-.
- 2 Change the tag name to the Lasso Script equivalent using the list in *Table 4: Replacement Tags* in *Chapter 4: CDML/Lasso Script Equivalents*.
- 3 If the tag has a parameter such as a literal field name or string then enclose it in single quotes.
- 4 If the tag has an intratag parameter then convert it to the Lasso Script equivalent using the list in *Table 6: Intratag Parameters* in *Chapter 4: CDML/Lasso Script Equivalents*. Convert the braces to parentheses.
- 5 If the tag has an encoding keyword then convert it to the Lasso Script equivalent using the list in *Table 5: Encoding Keywords* in *Chapter 4: CDML/Lasso Script Equivalents*.

Include Field

The tag [FMP-IncludeField] does not have a direct equivalent in Lasso Script. However, its functionality can be replicated using the [Process] and [Field] tags as follows:

```
[Process: (Field: 'Field Name')]
```

Display and Format Encoding

CDML supports two encoding keywords that do not have direct equivalents within Lasso Script: Format and Display.

The Format keyword can be used to translate the text formatting of a field on a layout to its HTML equivalent. There is no way in Lasso Script to replicate the function of this keyword. Instead, HTML formatting must be stored in FileMaker Pro fields and displaying using the Lasso Script -EncodeNone keyword.

The Display keyword can be used to send data using the custom numeric or date format which is specified on a layout rather than generic numeric or date formatting. In Lasso Script the [Date_Format], [Date->Format], [Integer->SetFormat] and [Decimal->SetFormat] tags can be used to format the returned data.

ModID

One significant feature of CDML that Lasso does not support directly is the modification ID. This is a token which is assigned to each record in FileMaker Pro and incremented each time the record is updated. A Web client can fetch the modification ID using [FMP-ModID] from FileMaker Pro and send it back with an -edit action. If the modification ID has changed then the edit is disallowed.

Equivalent functionality can be created in Lasso by passing the modification ID to Lasso in a calculation field on the Filemaker Pro layout. A requirement for an Exact Search can then be established within Lasso's security settings in Lasso Administration in order to require that the value of this field match in order to perform an update.

Looping Tags

Looping tags in CDML are equivalent to container tags in Lasso Script. CDML tags are a subset of those supported in Lasso Script (with one or two exceptions). In general the corresponding tags are called identically except for the different tag name.

See *Table 3: Looping Tags* in *Chapter 4: CDML/Lasso Script Equivalents* for a complete list of CDML looping tags and their Lasso Script container tag equivalents.

For example the [FMP-Record] ... [/FMP-Record] tag, which repeats once for each record in the found set is equivalent to the [Records] ... [/Records] tag in Lasso Script. Similarly, the [FMP-Header] ... [/FMP-Header] tag is equivalent to the [Header] ... [/Header] tag in Lasso Script.

See the section on *Replacement Tags* above for a discussion of tag parameters and intratag parameters. The [FMP-If] ... [/FMP-If] tag is treated separately in the *Conditional Expressions* section below.

To convert looping tags from CDML to Lasso Script:

The Lasso CDML Converter tool included with this paper is the easiest way to convert a site from CDML to Lasso Script. If you want to perform the conversion manually or want to double-check the automatic conversion the following steps are required:

- 1 Search your CDML format file for the start of a CDML tag [FMP-.
- 2 Change the tag name to the Lasso Script equivalent using the list in *Table 3: Looping Tags* in *Chapter 4: CDML/Lasso Script Equivalents*. Both the opening and the closing tags need to be changed.

- 3 If the tag has a parameter such as a literal field name or string then enclose it in single quotes.
- 4 If the tag has an intratag parameter then convert it to the Lasso Script equivalent using the list in *Table 6: Intratag Parameters in Chapter 4: CDML/Lasso Script Equivalents*. Convert the braces to parentheses.

Value List Tags

Lasso does not have an equivalent of the CDML tag for listing the value lists that are available on the current layout. The [FMP-ValueNames] ... [/FMP-ValueNames] tag is only available in CDML. Instead, Lasso always associates value lists with the field that they are formatted to show on the current layout.

The [FMP-ValueList] ... [/FMP-ValueList] tag in CDML takes the name of a value list as its parameter and loops through the values within the specified value list. The equivalent [Value_List] ... [/Value_List] tag accepts the name of a field on the current layout and loops through the value list that is associated with that field.

Conditional Expressions

The [FMP-If] ... [/FMP-If] tag in CDML is directly equivalent to the [If] ... [/If] tag in Lasso Script. However, the syntax of the conditional expression specified within the opening tag differs significantly.

See *Table 7: Conditional Symbols in Chapter 4: CDML/Lasso Script Equivalents* for a complete list of CDML conditional symbols and their Lasso Script equivalents.

In CDML a basic conditional expression consists of a right operand, a comparison symbol, and a left operand. The operands can be either literals, intratag parameters enclosed in braces, or sub-expressions. The comparison symbol is from the following list:

.eq.	.neq.	.cn.	
.lt.	.lte.	.gt.	.gte.
.and.	.or.	.xor.	

Lasso Script conditional expressions have a similar form. They consist of a right operand, a comparison symbol, and a left operand. However, Lasso's operands can be string literals in single quotes, sub-tags in parentheses, or sub-expressions enclosed in parentheses. Lasso's equivalent comparison symbols are shown in this list:

==	!=	>>	
<	<=	>	>=
&&		!=	

For example, an expression that checks the value of the field `First_Name` is equal to John would be formatted like this:

```
[FMP-If: {Field: First_Name} .eq. John] ... [/FMP-If]
```

The equivalent expression in Lasso would be formatted like this. The comparison symbol is different, the string literals are enclosed in quotes, and the braces are converted to parentheses.

```
[If: (Field: 'First_Name') == 'John'] ... [/If]
```

CDML allows for one level of subexpression. Using the `.and.`, `.or.`, or `.xor.` symbols, either operand can be a subexpression enclosed in parentheses. The following expression checks whether the `First_Name` is equal to John or the `First_Name` is equal to Jane.

```
[FMP-If: ({Field: First_Name} .eq. John) .or. ({Field: First_Name} .eq. Jane)]
...
[/FMP-If]
```

The equivalent expression in Lasso would be formatted like this:

```
[If: ((Field: 'First_Name') == 'John') || ((Field: 'First_Name') == 'Jane')]
...
[/If]
```

To convert conditional expressions from CDML to Lasso Script:

The Lasso CDML Converter tool included with this paper is the easiest way to convert a site from CDML to Lasso Script. If you want to perform the conversion manually or want to double-check the automatic conversion the following steps are required:

- 1 Search your CDML format file for a CDML conditional tag `[FMP-If.`
- 2 Change the tag names of the opening and closing tags to `[If] ... [/If]`.
- 3 If the expression has parameters such as a literal field names or strings then enclose them in single quotes.
- 4 If the tag has intratag parameters then convert them to the Lasso Script equivalents using the list in *Table 6: Intratag Parameters in Chapter 4: CDML/Lasso Script Equivalents*. Convert the braces to parentheses.
- 5 Convert the conditional symbols to their Lasso Script equivalents using the list in *Table 7: Conditional Symbols in Chapter 4: CDML/Lasso Script Equivalents*.
- 6 If there are any sub-expressions change the braces that enclose them to parentheses.

Exclusive Or

In this paper the CDML exclusive or comparison symbol `.xor.` is listed as being equivalent to the Lasso Script not equal `!=` symbol. This equivalency is true in a practical sense, but not in a general sense. The `.xor.` symbol in CDML can only be used in the limited context of comparing two sub-expressions in a conditional expression. If both are True or False then `.xor.` will return False, if one is True and the other False then `.xor.` will return True. The Lasso Script `!=` symbol returns the same results.

4

Chapter 4

CDML/Lasso Script Equivalents

This chapter includes quick reference tables that show the Lasso Script equivalents of all CDML tags, keywords, and comparison operators. The Lasso CDML Converter automatically converts all of the tags and syntax types in these tables.

Any tags or syntax types that do not have direct Lasso Script equivalents are marked in the table. The previous chapter includes details about how to manually convert these tags and syntax types.

Table 1: Database Action Tags

CDML Tag	Lasso Tag
-DBClose	(see note)
-DBNames	(see note)
-DBOpen	(see note)
-Delete	-Delete
-Dup	-Duplicate
-Edit	-Update
-Find	-Search
-FindAll	-FindAll
-FindAny	-Random
-Img	-Image
-LayoutNames	(see note)
-New	-Add
-ScriptNames	(see note)
-View	-Show

Note: The `-DBOpen`, `-DBCclose`, `-DBNames`, `-LayoutNames`, and `-ScriptNames` tags must all be converted manually. See the section on *Database Action Tags* in the previous chapter for details.

Table 2: Variable Tags

CDML Tag	Lasso Tag
<code>-DB</code>	<code>-Database</code>
<code>-Error</code>	<code>-ResponseAnyError</code>
<code>-ErrorFmtField</code>	(see note)
<code>-ErrNum</code>	<code>-ResponseAddError</code> , <code>-ResponseDeleteError</code> , <code>-ResponseDuplicateError</code> , <code>-ResponseSearchError</code> , <code>-ResponseUpdateError</code> (see note)
<code>-Format</code>	<code>-Response</code>
<code>-FmtField</code>	<code>-Response=Field:...</code> (see note)
<code>-Lay</code>	<code>-Table</code>
<code>-Lop</code>	<code>-OperatorLogical</code>
<code>-MailBCC</code> , <code>-MailCC</code> <code>-MailFmtField</code> , <code>-MailFormat</code> <code>-MailFrom</code> , <code>-MailHost</code> <code>-MailSub</code> , <code>-mailto</code>	<code>[Email_Send: ...]</code> (see note)
<code>-Max</code>	<code>-MaxRecords</code>
<code>-ModID</code>	(see note)
<code>-Op</code>	<code>-Operator</code>
<code>-RecID</code>	<code>-KeyValue</code>
<code>-Script</code>	<code>-FMScript</code>
<code>-Script.PreFind</code>	<code>-FMScript.Pre</code>
<code>-Script.PreSort</code>	<code>-FMScript.PreSort</code>
<code>-Skip</code>	<code>-SkipRecords</code>
<code>-SortField</code>	<code>-SortField</code>
<code>-SortOrder</code>	<code>-SortOrder</code>
<code>-StyleHref</code>	(see note)
<code>-StyleType</code>	(see note)
<code>-Token</code>	<code>-Token</code>

Note: The `-errorfmtfield`, `-errnum`, `-mail...`, `-modid`, `-stylehref`, and `-styletype` tags must all be converted manually. See the section on *Variable Tags* in the previous chapter for details.

Table 3: Looping Tags

CDML Tag	Lasso Tag
[FMP-CurrentFind] ... [FMP-CurrentFind]	[Search_Arguments] ... [/Search_Arguments]
[FMP-CurrentSort] ... [/FMP-CurrentSort]	[Sort_Arguments] ... [/Sort_Arguments]
[FMP-Header] ... [/FMP-Header]	[Header] ... [/Header]
[FMP-If] ... [/FMP-If]	[If] ... [/If]
[FMP-InlineAction] ... [/FMP-InlineAction]	[Inline] ... [/Inline]
[FMP-LayoutFields] ... [/FMP-LayoutFields]	[Loop: (Field_Name: -Count)] ... [/Loop]
[FMP-Link] ... [/FMP-Link]	[Link_CurrentAction] ... [/Link_Currentaction]
[FMP-LinkFirst] ... [/FMP-LinkFirst]	[Link_FirstGroup] ... [/Link_FirstGroup]
[FMP-LinkLast] ... [/FMP-LinkLast]	[Link_LastGroup] ... [/Link_LastGroup]
[FMP-LinkNext] ... [/FMP-LinkNext]	[Link_NextGroup] ... [/Link_NextGroup]
[FMP-LinkPrevious] ... [/FMP-LinkPrevious]	[Link_PrevGroup] ... [/Link_PrevGroup]
[FMP-Log] ... [/FMP-Log]	[Log] ... [/Log]
[FMP-Portal] ... [/FMP-Portal]	[Portal] ... [/Portal]
[FMP-Record] ... [/FMP-Record]	[Records] ... [/Records]
[FMP-Repeating] ... [/FMP-Repeating]	[Repeating] ... [/Repeating]
[FMP-ValueList] ... [/FMP-ValueList]	[Value_List] ... [/Value_List] (see note)
[FMP-ValueNames] ... [FMP-ValueNames]	(see note)

Note: The CDML `[FMP-ValueList] ... [/FMP-Value_List]` tags behave differently when translated to Lasso Script `[Value_List] ... [/Value_List]` tags. The `[FMP-ValueNames] ... [/FMP-ValueNames]` tags do not have an Lasso Script equivalent. See the section on *Looping Tags* in the previous chapter for details.

Table 4: Replacement Tags

CDML Tag	Lasso Tag
[FMP-ClientAddress]	[Client_Address]

CDML Tag	Lasso Tag
[FMP-ClientIP]	[Client_IP]
[FMP-ClientPassword]	[Client_Password]
[FMP-ClientType]	[Client_Type]
[FMP-ClientUserName]	[Client_Username]
[FMP-ContentMIMEType]	[Content_Type]
[FMP-Cookie]	[Cookie]
[FMP-CurrentAction]	[Lasso_Action]
[FMP-CurrentDate]	[Server_Date]
[FMP-CurrentDay]	[Server_Day]
[FMP-CurrentDatabase]	[Database_Name]
[FMP-CurrentError]	[Error_CurrentError]
[FMP-CurrentFormat]	[Response_FilePath]
[FMP-CurrentFoundCount]	[Found_Count]
[FMP-CurrentLayout]	[Table_Name]
[FMP-CurrentLOP]	[Operator_LogicalValue]
[FMP-CurrentMax]	[MaxRecords_Value]
[FMP-CurrentModID]	(see note)
[FMP-CurrentPortalRowNumber]	[Loop_Count]
[FMP-CurrentReclD]	[KeyField_Value]
[FMP-CurrentRecordCount]	[Total_Records]
[FMP-CurrentRecordNumber]	[Record_Count]
[FMP-CurrentRepeatNumber]	[Loop_Count]
[FMP-CurrentSkip]	[SkipRecords_Value]
[FMP-CurrentTime]	[Server_Time]
[FMP-CurrentToken]	[Token_Value]
[FMP-ElseIf]	[Else]
[FMP-Field]	[Field]
[FMP-FieldName]	[Field_Name: Loop_Count]
[FMP-FindFieldItem]	[Search_FieldItem]
[FMP-FindOpItem]	[Search_OperatorItem]
[FMP-FindValueItem]	[Search_ValueItem]
[FMP-Image]	[Image_URL]

CDML Tag	Lasso Tag
[FMP-Include]	[Include]
[FMP-IncludeField]	[Process: (Field: ...)] (see note)
[FMP-LinkRecID]	[Link_DetailURL]
[FMP-Option]	[Option]
[FMP-RangeEnd]	[Shown_Last]
[FMP-RangeSize]	[Shown_Count]
[FMP-RangeStart]	[Shown_First]
[FMP-RepeatingItem]	[Repeating_Valueltem]
[FMP-SetCookie]	[Cookie_Set]
[FMP-SortFieldItem]	[Sort_FieldItem]
[FMP-SortOrderItem]	[Sort_OrderItem]
[FMP-ValueListChecked]	[Checked]
[FMP-ValueListItem]	[Value_ListItem]
[FMP-ValueNameItem]	(see note)

Note: The CDML [FMP-IncludeField] and [FMP-CurrentModID] tags do not have direct Lasso Script equivalents. See the section on *Replacement Tags* in the previous chapter for details.

Table 5: Encoding Keywords

CDML Keyword	Lasso Keyword
Break	-EncodeBreak
Display	(see note)
Format	(see note)
HTML	-EncodeHTML
Raw	-EncodeNone
URL	-EncodeStrictURL

Note: CDML has several keywords (display and format) which do not have equivalents in Lasso Script. See the section on *Replacement Tags* in the previous chapter for details.

Table 6: Intratag Parameters

CDML Tag	Lasso Tag
{CanDelete}	(see note)
{CanEdit}	(see note)
{CanNew}	(see note)
{CurrentAction}	(Lasso_Action)
{ClientAddress}	(Client_Address)
{ClientIP}	(Client_IP)
{ClientPassword}	(Client_Password)
{ClientType}	(Client_Type)
{ClientUsername}	(Client_Username)
{CurrentCookie}	(Cookie)
{CurrentDate}	(Server_Date)
{CurrentDay}	(Server_Day)
{CurrentDatabase}	(Database_Name)
{CurrentError}	(Error_CurrentError)
{CurrentFormat}	(Response_FilePath)
{CurrentFoundCount}	(Found_Count)
{CurrentLayout}	(Table_Name)
{CurrentLOP}	(Operator_LogicalValue)
{CurrentMax}	(MaxRecords_Value)
{CurrentModID}	(see note)
{CurrentPortalRowNumber}	(Loop_Count)
{CurrentRecordCount}	(Total_Records)
{CurrentRecordNumber}	(Loop_Count)
{CurrentRepeatNumber}	(Loop_Count)
{CurrentSkip}	(SkipRecords_Value)
{CurrentTime}	(Server_Time)
{CurrentToken}	(Token_Value)
{Field}	(Field)
{IsSorted}	(see note)
{RangeEnd}	(Shown_Last)
{RangeSize}	(Shown_Count)

CDML Tag	Lasso Tag
{RangeStart}	(Shown_First)
{ValueListItem}	(Value_ListItem)

Note: The CDML {CurrentModID} intratag parameter does not have a direct Lasso Script equivalent. See the section on *Replacement Tags* in the previous chapter for details.

Table 7: Conditional Symbols

CDML Tag	Lasso Tag
.eq.	==
.neq.	!=
.gt.	>
.gte.	>=
.lt.	<
.lte.	<=
.cn.	>>
.ncn.	!>>
.and.	&& (see note)
.or.	(see note)
.xor	!= (see note)

Note: Conditional expressions should always be double checked for accuracy since it can be difficult for the converter to decide heuristically whether a word is a string literal or a field name. For the purposes of conditional expressions the CDML .xor. symbol is equivalent to the != symbol in Lasso Script. See the section on *Conditional Expressions* in the previous chapter for details.

5

Chapter 5

Tools and Resources

This chapter includes pointers to many tools and resources which can help when converting a solution from FileMaker Pro CDML to Lasso Script.

Note: The products and links in this chapter are included for reference only and should not be construed as a recommendation to use any of these products. All of the links were valid at the time this paper was published.

OmniPilot Software, Inc.

Main Site – <http://www.omnipilot.com/>

Lasso Professional – <http://www.omnipilot.com/Lasso/>

Lasso Studio – <http://www.lassostudio.com/>

OmniPilot Store – <http://store.omnipilot.com/>

Support – <http://support.omnipilot.com/>

Lasso Reference – <http://ldml.omnipilot.com/>

FileMaker Pro, Inc.

Main Site – <http://www.filemaker.com/>

FileMaker Pro – <http://www.filemaker.com/products/>

Support – <http://www.filemaker.com/support>

Documentation – <http://www.filemaker.com/downloads/>

FileMaker Store – <http://store.filemaker.com/>