# FALCON LYNX BROWSER

LYNX

GNU GENERAL PUBLIC LICENSE Version 2, June 1991



Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to

the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

Appendix: How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.> Copyright (C) 19yy <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) 19yy name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'. This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

# Technical Support

The Falcon Lynx Browser comes with free online technical support. Users can review the most common issues on the PSC Product FAQ web page or users can email PSC technical support staff:

FAQ: http://www.pscnet.com/html/faq.htm

Email: techsupport@pscnet.com or
http://www.pscnet.com/html/technical_support_request.htm

Since the Falcon Lynx Browser is an extension of the open-source Lynx product, PSC technical support will be limited to installation issues and Falcon-specific HTML tags (matchupg, backlight, etc.).

If necessary, you can obtain live technical support on a pay-per-incident basis. Call (541) 344-1189 for more information.

For more generic technical support regarding the Lynx browser, visit www.falconlynx.com for a list of technical resource links.

# 1 OVERVIEW

LYNX

# Welcome to Falcon Lynx Browser

The Falcon Lynx Browser (FLB) is a web browser that can run on your Percon Falcon. This browser enables developers to create web-based applications for real-time data collection environments, using the latest software development technologies and techniques.

As the name suggests, the Falcon Lynx Browser is an extension of the Lynx browser, a popular DOS and Unix character-mode web browser.

The Falcon Lynx Browser includes the following features:

- Optimized for the Falcon screen and gray scaling
- Supports hyphenation and justification of text
- Entry pattern validation
- Text entry/text areas
- Checkboxes and radio-buttons
- Proxies
- Cookies
- Server-side scripting
- DHCP Support
- FTP support for file transfers
- WWW browsing
- Supports most features of HTML v1.0 through v4.0

For those advanced users/developers who are interested, the source code for the Falcon Lynx Browser is available. Please contact Percon technical support for details.

# What This Manual Covers

Since the Falcon Lynx Browser is an extension of the popular Lynx browser, this manual will only cover the specific extensions created for the Percon implementation.

For general instruction and information about the Lynx browser, please see the Lynx documentation, which is included as a separate PDF (lynxman.pdf).

This documentation will include the following subjects:



# System Requirements

To use the Lynx browser on your Falcon, you need:

- Falcon RF Terminal with firmware v2.6 or greater
- Network card ("wired" NIC, RF, or CDPD)

To check the version of firmware installed on your Falcon, type `rev` at a c:\ prompt on your Falcon. If your Falcon firmware needs to be upgraded, install the Falcon Configuration Utility V2.6 (available at http://www.percon.com under "Support"). See the readme file for instructions on performing the firmware upgrade under the `c:\percon\falcon\16line\coreupd` directory.

# Technical Introduction to Lynx

Lynx is a fully functional text mode web and FTP browser, which is an ideal fit for DOS based portable devices. The following section, which discusses the Lynx implementation for the Falcon, is meant for advanced users and/or web developers.

## Common Lynx Features

- Server-side Scripting Support
- HTML Forms Support
- Cookies (persistent and non-persistent)
- Cookie Management (manage cookies individually, by domain, etc.)
- Custom MIME-type Mappings
- FTP and HTTP Proxy Support
- Multiple Character Set Support (including Unicode)
- Easy Configuration with Configuration Files and Command Line Functions (includes environment variables, such as proxy or startfile)
- Text Justification (aligning left and right edge of text to one vertical line by inserting extra spaces between words)
- Detailed Control Headers Transfers HTTP Server (e.g. Referer, UserAgent)
- Detailed Control of HTML Parsing (allows ability to handle various versions of HTML specifications)
- Client-side and Server-side Image Maps Support (as hyperlinks)
- Multiple Image Handling Options (ignores, hyperlinks, etc.)
- Multiple Screen Sizes (e.g. 16x20)
- On-the-fly Decompression of GZIPPED Files

## Custom Falcon Lynx Features

- Text Hyphenation Support (hyphenation rules composed for TeX typesetting system can be translated to simplified format for hyphenation rules that Lynx uses)
- Falcon-specific HTML Extensions (client-side validation, input processing, backlight support, input source verification, etc.)
- Batch and Configuration Files to Aid Lynx Configuration
- Lynxconf Utility for Falcon/Network Configuration (netmask, gateway, IP, DNS IP, etc.)

## Useful Lynx Hyperlinks and Resources

- Official web site for lynx: http://lynx.isc.org
- Useful site with lynx-related links: http://lynx.browser.org
- Lynx-dev (lynx-dev@sig.net) - mailing list dedicated to lynx development. To subscribe, send a message with 'subscribe lynx-dev' in body to majordomo@sig.net.

# 2 CONFIGURE AND INSTALL LYNX BROWSER

LYNX

# Install Falcon Lynx Browser

Before you begin to load your Falcon with the Falcon Lynx Browser, be sure that you have contacted or are working with your IT personnel. You will need some specific network information to correctly configure your Falcon Lynx Browser. It may be helpful to read the following section and gather the information before actually performing the following steps.

## Set Home Page

Just like any Windows based Internet browser, you can set a home page for your Lynx browser. This page is automatically opened each time the Lynx browser is started. This home page can be a local HTML file or a remote website. Depending on your needs, complete the steps in one of the following sections to define your home page.

### Specify Remote Page

If your users will always have access to the remote web application that you've built, then you may elect to specify the URL of the application as the home page for a Falcon terminal equipped with the Falcon Lynx Browser.

Specifying the remote URL is done in the LOCAL.CFG file. Complete the following steps to make the necessary modification to define a remote URL as the home page:

1. **Select START | PROGRAM FILES | ACCESSORIES | NOTEPAD.**
2. **Select OPEN from the FILE menu.**
3. **Navigate to the Percon\Falcon\Lynx\common directory.**
4. **Select ALL FILES from the FILES OF TYPE pull-down list.**
5. **Select LYNX.CFG and press the OPEN button.**

6 **Change the STARTFILE value of the LYNX.CFG file from STARTFILE:.\start.htm to a fully qualified URL of your choosing.**
For example, your new line might read:

```
STARTFILE:http://www.myCompany.com
```

7 **Select SAVE from the FILE menu.**

8 **Select EXIT from the FILE menu.**
You will be returned to the Falcon Configuration Utility.

### Build Local Start Page

In most cases, users of the Falcon Lynx Browser will not be browsing the internet, but will rather be using web-based applications for data collection purposes.

The START page is actually an HTML file located in the directory containing the Falcon Lynx Browser. Though this is a local file, it should contain hyperlinks to web-based applications that have been deployed in a warehouse, manufacturing, or other environments requiring automated data collection.

It is preferable to modify the contents of the START page before you load it to a Falcon Terminal containing the Falcon Lynx Browser. Using your favorite HTML editor, add or modify hyperlinks in the START.HTM file, located in the Percon\Falcon\Lynx\common directory. Though your users can open any website in the Falcon Lynx Browser, creating a START page for your users will help them easily and quickly begin their data collection activities instead of manually typing URLs.

Finally, since you need to load the START page to the Falcon Terminal, it is preferable to create a START page that contains static hyperlinks to pages that will provide the user with dynamic choices (this approach will prevent you from having to load the START page to the Falcon Terminal each time your application changes).

**Modify LYNX.CFG -** Before you can load your new Start page to the Falcon, you need to modify the LYNX.CFG file to use a local HTML file as the home page.

Complete the steps in "Specify Remote Page" on page 2-2, but make sure to set the STARTFILE parameter to read the local START HTML file. Your new STARTFILE parameter will read:

```
STARTFILE:.\Start.htm
```

This setting will cause your Falcon Lynx Browser to use a local copy of the START.HTM file as the default home page.

## Load Falcon Lynx Browser

Before you can begin loading your Lynx software and configuration files to your Falcon Terminal, please make sure that you have installed the Falcon Configuration Utility.

This procedure will illustrate how to perform all network configuration on your portable data terminal using the LYNXCONF utility. You can preset much of the network information by modifying the configuration files before loading the Falcon Lynx Browser. Consult the appropriate Falcon RF guide for your RF configuration.

Complete the following steps to load your Falcon Terminal with the software and configuration files:

1. **Press the START button in the lower left corner of your desktop.**

2. **Select the PROGRAMS ⇨ FALCON CONFIGURATION UTILITY program group.**

3. **Select the FALCON CONFIGURATION UTILITY program item.**
   The Falcon Configuration Utility comes with each Falcon Terminal.

4. **Press the CUSTOM button.**

5 **Select the appropriate Lynx configuration file for your RF radio.**

| | |
|---|---|
| ar_lynx.cfg | Cisco 340 Series (Aironet) |
| ls_lynx.cfg | Linksys (wired) |
| lu_lynx.cfg | Lucent WaveLAN 2/11 |
| px_lynx | Proxim RangeLAN2 |
| s802lynx | Symbol Spectrum24 802.11 |
| sm_lynx | Symbol Spectrum24 Spring |

6 **Press the OPEN button.**

7 **Select the appropriate Falcon configuration file.**
For most cases, you will simply want to select the Default.prs file.

8 **Press the OPEN button.**

9 **Press the DOWNLOAD button and follow the on-screen instructions to complete loading your Falcon Terminal.**
Remember to type LD at the C:\ prompt on your Falcon to place it in the "program mode."

10 **When the unit has completed downloading all of the necessary files, reboot your Falcon portable data terminal.**

## Use LYNXCONF

The LYNXCONF utility enables users to configure their Lynx browser directly on the Falcon. For example, this utility enables users to configure network information, such as IP address and netmask, or default start page.

The first time that your reboot your Falcon after loading the Lynx browser, the LYNXCONF utility will automatically start. If you ever need to invoke the LYNXCONF utility after the initial reboot, simply type LYNXCONF in the c:\lynx directory.

1. **If the LYNXCONF utility doesn't automatically start, type `c:\lynx\lynxconf` at a c:\ prompt.**
2. **The LYNXCONF utility will walk you through the network settings; review each setting and change as necessary.**
   Be certain to work with your network administrator so that you properly configure your Falcon portable data terminal.

   If possible, it is recommended that you allow LYNXCONF to determine network settings by using DHCP.

# 3 USING FALCON LYNX BROWSER

LYNX

# Using the Falcon Lynx Browser

The Falcon Lynx Browser is based on the popular DOS Lynx browser that is built and maintained as an open-source project (maintained under the GNU General Public License (GPL)).

As such, there is a great deal of documentation available for the Lynx browser. It is recommended that you read the included Lynx manual.

The following table contains just a few useful commands to get you up and running with the Falcon Lynx Browser.

| Basic Falcon Lynx Browser Commands | |
|---|---|
| **Keystroke** | **Purpose** |
| <g> | Open URL |
| <G> | Modify Current URL (this is a helpful feature if you want to open a different page of the current website) |
| <q> | Quit the Falcon Lynx Browser |
| <tab> | Move to the next field, form control, or hyperlink. |
| <shift><tab> | Move to the previous field, form control, or hyperlink. |
| <page down> | Scroll down. |
| <page up> | Scroll up. |
| <left arrow> | Move to the previously viewed page (like the Back button in your regular web browser) |
| <right arrow> or <enter> | Follow the currently selected hyperlink. |
| <up arrow> | Select previous link. |
| <down arrow> | Select next link. |
| <ctrl g> | Escape from current prompt. |
| <ctrl r> | Reload current page. |
| <ctrl l> | Repaint current screen. |

**Table 3-1:** List of Basic Falcon Lynx Browser

For more information about additional features of the Falcon Lynx Browser, consult the Lynx documentation.

## View Lynx Manual

To view the standard Lynx documentation, complete the following steps:

1. **Select START | PROGRAMS | FALCON LYNX BROWSER | GENERIC LYNX DOCUMENTATION.**

## Start Lynx on Falcon

If you followed the instructions in Chapter 2, then the Falcon Lynx Browser will automatically start when you boot your Falcon. If the Falcon Lynx Browser does not start automatically, then your UPG-START.BAT file has most likely been modified to not start the Falcon Lynx Browser automatically. If this is the case, complete the following step:

1 **Type Lynx at the c:\ prompt on your Falcon.**

4

# FALCON LYNX TAG EXTENSIONS

LYNX

# HTML Tag Extensions

The Falcon Lynx Browser has the ability to understand a series of HTML extensions for Text Fields and Text Areas. These tags allow you to perform data validation and some data manipulation on the Falcon Lynx Browser at the time of data entry.

In addition to ensuring that entered data is accurate, network traffic is reduced since the data validation is performed on the Falcon versus performing the validation at the server.

## MATCHUPG

The Falcon Lynx Browser allows you to implement pattern matching using a series of character “building blocks” (users of Percon’s Universal Program Generator will be familiar with these building blocks as the Falcon Lynx Browser has a similar implementation). Assemble the building blocks in the order that you want your entry pattern to match. Any entry that does not match the entry pattern will be rejected and the user will receive the following message: “Field doesn't match pattern!”

Building blocks are assembled inside of quotation marks using the MATCHUPG tag for an input field. For example, if you were trying to ensure that a user entered a properly formatted Social Security number, the HTML for your input field might look like:

```
<input type="text" name="fldITEM"
matchupg='###-##-####'>
```

If the user does not enter a properly formatted Social Security number, a beep will sound (ALERTBEEP) and a message will inform the user that the entered value is not properly formatted.

### *Syntax*

matchupg=*argument*

where argument is a valid entry pattern. The MATCHUPG attribute requires that you assemble entry patterns within single or double quotation marks if the pattern contains spaces; for example, you might specify `matchupg='### ##'`. This attribute is valid with INPUT or TEXTAREA tags. The remainder of this section describes how to build data entry patterns.

**Basic Entry Pattern Building Blocks**

| Character | Description | Example |
|---|---|---|
| 0-9 | Entry pattern will accept any of the specified numbers. | 146 (accept only the number 146). 5#### (accept any 5 digit number that begins with 5) |
| a-z, A-Z | Entry pattern will accept any of the specified alpha characters. The entry pattern is case insensitive until a $ (see below) is used. | akdt (accept only akdt as a valid entry, with any combination of character case) |
| $ | Make entry pattern case sensitive. An entry pattern is case insensitive unless a $ is used. If $ is used more than once in an entry pattern, the case status is toggled. | $abcdEFG (accept only abcdEFG as a valid entry). |
| ? | Entry pattern will accept any single character; operates like a single character wildcard. | ##?## (accepts 2 numbers, followed by any single character, followed by 2 numbers). |
| # | Entry pattern will accept only numerical data (0-9). | ### (accepts 3 numbers) |
| @ | Entry pattern will accept an alpha character (is not case sensitive). | @@@@ (accepts 4 letters) |
| : | Entry patterns will accept either alpha or numeric data entry. | :::: (accepts any combination of 4 numbers or letters) |
| ^ | Will only accept uppercase alpha data entry. | ^^^ (accepts only 3 uppercase letters) |
| & | Will only accept lowercase alpha data entry. | &&& (accepts only 3 lowercase letters) |
| * | Entry pattern will accept any number of characters after the wildcard (*). This works just like the wildcard (*) in a DOS command, such as `DIR t*.*`. You should only place a wildcard at the end of a pattern. | ###* (accept 3 numbers followed by any number of valid characters, such as 333kd-9 or 333h-9-kk) |
| ' | This is the escape character to be used if you want to include a building block character in your entry pattern. | '###-## (accepts a number sign # followed by 2 numbers, a dash, and 2 more numbers, i.e. #33-44) |

**Table 4-1:** Combine the individual building block characters in this table to form meaningful entry patterns; an entry pattern can increase the quality of collected data by ensuring that only data with the proper format is entered.

### *Sample Entry Pattern Definitions*

Use the following table to learn how to assemble the building blocks contained in Table 4-1 into meaningful entry patterns.

| Basic Entry Pattern Building Blocks | | |
|---|---|---|
| **Data Example** | **Entry Pattern** | **Explanation** |
| PN1234-01 | PN####-## | This entry pattern requires a prefix of PN, followed by any 4 numbers, a hyphen, and any two more numbers. |
| ABC1234cb | ^^^####&& | Accepts any three uppercase letters, followed by any 4 numbers, and ending with any 2 lowercase letters. |
| 1234ABCD | ####???? | Accepts any 4 numbers followed by and 4 letters. |
| ABcdEF | $ABcdEF | User must enter uppercase A and B, followed by a lowercase c and b, completed with an uppercase E and F. |
| PN123994-223 | PN* | This entry pattern requires a prefix of PN but can be followed by any combination of numbers, letters, or other ASCII characters. |

**Table 4-2:** Examples of data entry patterns built with the Basic Entry Pattern Building Blocks detailed in table Table 4-1.

### *Advanced Entry Patterns*

In addition to simply enforcing simple entry patterns, you have the ability to manipulate the collected data with advanced entry pattern building blocks. Table 4-3 lists all of the advanced entry pattern building blocks available with the Falcon Lynx Browser. Each is discussed in detail in the following sections.

Use Table 4-3 along with the examples detailed in this section to determine what advanced entry patterns you need. It is important that after you define your advanced entry patterns that you test your application completely; test different combinations of entered data.

| Advanced Entry Pattern Building Blocks | | |
|---|---|---|
| **Character** | **Description** | **Example** |
| {*saved data*} | Enclose the portions of the entry pattern that you want to save to disk; the portion of the entry pattern outside the braces will be discarded (used for parsing). | ##{###}# (accepts a 6 digit number as valid data entry, but only saves the third, fourth, and fifth numbers to disk) |
| [*character set*] | Enclose the valid character set within brackets for a position ([ ]); use a hyphen (-) to specify a range of characters. | [abc]## (accepts only a, b, or c followed by 2 numbers as valid data entry) |
| ^ | When placed at the beginning of a character set, makes the values in the character set invalid. | [^abc]### (accepts all letters except a, b, or c followed by 3 numbers as valid data entry) |

**Table 4-3:** Use the advanced building block characters contained in this table to increase the power of your entry pattern. Many of these building blocks not only ensure quality of the data entered, but also allows you to manipulate the entered data.

## *Character Sets*

Previously, you have learned how to require a letter (case sensitive or insensitive) in an entry pattern. Defining a character set will allow you to define a range of characters that are acceptable data entry for a particular entry pattern position.

For example, you might want to accept a 3 digit number that has a prefix within the range of a-e. You would enter the value [a-e]### as your entry pattern. You can also use a caret (^) to exclude ranges. For example, you might enter [^abcd] to accept data entry that does not include a, b, c, or d in the first character position.

**Building Blocks:** [*character set*], ^

**Examples:** [A-M]###
*Will accept data entry of* A *through* M *followed by three numbers.*

[^abc]##
*Will accept data entry of any letter except* a, b, *or* c *followed by two numbers.*

[cgkm]@@@
*Will accept 4 letters, but the first character must be* c, g, k, *or* m.

### *Data Parsing*

It may be necessary at times to remove certain characters that are contained in a barcode (or collected data for that matter); for example, if the value of a barcode is ISBN-67543, you may want to write only the "67543" to your data collection file. You can use data parsing to remove the "ISBN-" portion of the barcode. This is a very simple example, but the Falcon Lynx Browser's data parsing capabilities (sometimes called substring extraction) allow you to remove any part of the collected data, even if it is in the middle of a value.

Braces, sometimes known as *curly brackets*, ({}) are used to surround the portion or portions of the entry pattern that you want to save; all portions of the entry pattern that are not within the braces will not be saved to disk (it will be discarded).

**Building Blocks:** {*saved data*}*discarded data*

**Examples:** ##{###}@@

*Will accept data entry of 5 numbers, followed by 2 letters. Only the third, fourth, and fifth numbers will be saved, while the rest will be discarded.*

##{@@}#{##}

*Will accept 2 numbers, followed by 2 letters, followed by 3 numbers as valid data entry. Only the 2 letters and the last 2 numbers of the entry pattern will be saved; everything else will be discarded.*

### *Multiple Entry Patterns*

If you want to define multiple entry patterns for your input field, simply separate the entry patterns with a pipe (|). For example, if you wanted to define #### and ##@## as the multiple entry patterns, you would specify ####|##@## in the Entry Pattern field.

---

**Note:** The data parsing capabilities specified in the previous section do not work when multiple entry patterns are defined.

---

**Building Blocks:** | (pipe)

**Examples:** @@@@|####
*Accept 4 letters or 4 numbers.*

PN####-##|####
*Will accept data entry that contains the prefix PN, followed by any four numbers, a hyphen, and ending with any two numbers or data entry that contains any 4 numbers.*

## BACKLIGHT

The BACKLIGHT attribute turns on the Falcon backlight when the field becomes active (when the cursor is in the field). The BACKLIGHT attribute takes no arguments and can be used with the INPUT, SELECT, TEXTAREA tags.

A sample implementation might look like:

<input type=text backlight name=i2>

### *Syntax*

backlight

where there are no valid arguments.

## INPUT-SOURCE

Using the INPUTSOURCE attribute, you can force users to either use the scanner for data input or the keypad for data input.

The INPUTSOURCE attribute has only three acceptable arguments: ‘k,’ ‘s,’ or ‘a’. As you might guess, ‘k’ forces keypad data entry and an ‘s’ forces scanner data entry. The ‘a’ value accepts either keypad or scanner entry. The INPUTSOURCE attribute works with INPUT (type=text) and TEXTAREA tags.

A sample implementation of the INPUTSOURCE attribute might look like:

<input type=text name=i8 inputsource=k>

### *Syntax*

inputsource=*argument*

where argument is ‘k’ for keypad, ‘s’ for scanner, or ‘a’ for either keypad or scanner entry.

## ALERTBEEP

If any entry pattern is violated, you can use the ALERTBEEP attribute to sound an error beep. This can help draw user’s attention to the error condition. The ALERTBEEP attribute is compatible with the INPUT (type=text, matchupg=<pattern>) and TEXTAREA (matchupg=<pattern>) tags.

### *Syntax*

alertbeep=*duration@tone*

where duration is the length of the tone (in milliseconds) and tone is the frequency of the beep (in hertz). Be sure to include the @ sign between the duration and tone arguments.

If the duration is specified without the tone specification (i.e. alertbeep=*duration*), than an audible alert will use the default tone.

Additionally, if a tone value is specified, it should be specified in the range of 100 to 65000.

## RXFAIL-ALERT

The RXFAILALERT attribute allows you to define an alternate message when an entry pattern is violated. Use this attribute to provide useful error messages when a pattern is violated. For example, if a field has been defined to only accept numbers and a user enters letters, you can use the RXFAILALERT tag to remind users to "Enter Numbers Only." This attribute is compatible with the INPUT (type=text, matchupg=<pattern>) and TEXTAREA (matchupg=<pattern>) tags.

### *Syntax*

rxfailalert="*message*"

where message is the text that you want displayed when the entry pattern is violated. Make sure to include the text within double or single quotes.

## AUTOENTER

When used in conjunction with the standard MAXLENGTH attribute for a INPUT tag, the AUTOENTER attribute will automatically advance the cursor to the next field (or form object) when the length of the data entry matches the length specified by the MAXLENGTH attribute. This attribute is compatible with the INPUT tag (type=text, maxlength=<length>).

A sample implementation of the AUTOENTER attribute might look like:

<input type=text maxlength=4 autoenter name=i1>

### *Syntax*

autoenter

where no arguments are required. The MAXLENGTH attribute, however, must be part of the form object definition.

## CLEAR

The CLEAR attribute forces a form to clear itself after submission. This prevents stale data to be present in the form when it is reloaded.

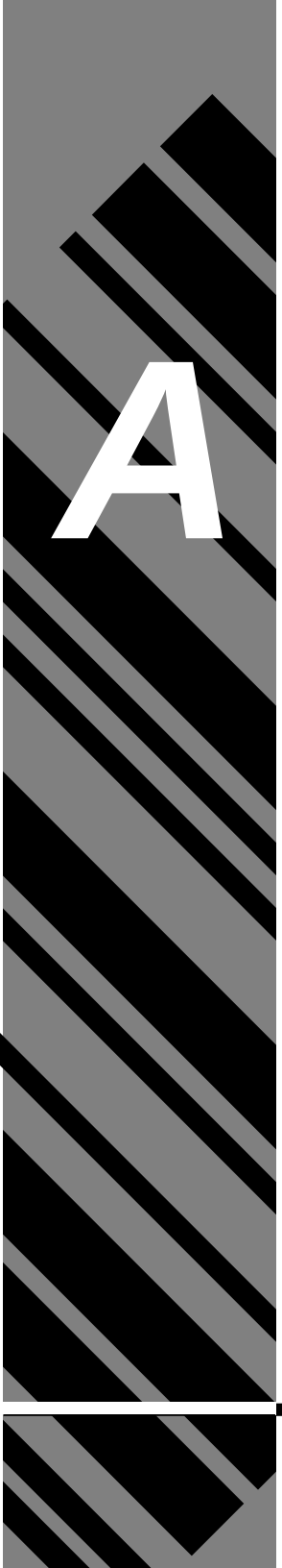
A sample implementation of the CLEAR attribute might look like:

<form clear method="GET" action="update.asp">

### *Syntax*

clear

where no arguments are required.



# A SHORT FILE DESCRIPTIONS

LYNX

# Description of files in \common directory (PC)

In general, all files in this directory are uploaded to Falcon (though screen.cfg is downloaded from the directory corresponding to the model of Falcon).

**Cwsdpmi.exe**

DPMI server for DOS. Some default settings of it are modified via cwsparam.exe (e.g. swapping is disabled).

**Dlynx.bat**

bat file to run lynx on machine that doesn't support extended Percon BIOS calls

**Emu387.dxe**

module that is used to emulate math coprocessor

**Hydict**

Hyphenation dictionary

**Jumps.htm**

Example of jump file

**Lynx.cfg**

Main lynx.cfg independent of Falcon model - it's subject to change without notice, modify (and redistribute) **local.cfg** instead

**Nullpkt.com**

Fake-net packet driver - allows user to run lynx on computer not equipped with network adapter or RF card

**Plynx.bat**

bat file to run lynx on computer without networking adapter but that supports extended Percon BIOS calls

**Plynxn.bat**

bat file to run lynx on computer with networking adapter (it assumes that drivers are already booted) provided it supports extended Percon BIOS calls

**Start.htm**

Default start file (will be loaded by lynx if you don't define another in local.cfg)

**_dolynx.bat**

Low level bat file used to start lynx that is called by other bat files.

**_lynx.exe**

Lynx for Falcon executable

**_mime.typ**

File that defines mapping of file's extension to its MIME type (used to decide whether lynx can render it or prompt to confirm downloading)

**base.lst**

File that specifies what files to load to Falcon. The file base.lst won’t be downloaded itself.

**demo.htm**

File that shows some of Percon's extensions to HTML

**local.cfg**

This is the configuration file you can customize for your organization and then upload to each Falcon

**lynx.bat**

This file will be uploaded to the root of c: - it's a general interface to invoke lynx - it will automatically determine whether you've booted RF card drivers and starts them if you haven't yet, and then starts lynx. If no RF card drivers are installed, then lynx will be run with networking disabled.

**lynxconf.exe**

The program you can use directly on the Falcon to configure most of lynx's important options along with networking parameters

**lynxconf.it**

Placeholder - this file is used to indicate that lynxconf hasn't run on the computer. lynx-conf will be started if this file exists and after that this file will be removed - this causes lynxconf to automatically start first time you boot computer with lynx installed.

**screen.cfg**

This file is specific to each model of Falcon - it defines screen size and colors (shades of gray) used by lynx.

**start.htm**

This is the default start (home) page for the Falcon Lynx Browser.

**upgstart.bat**

This file is uploaded to c: - it's started automatically by autoexec.bat. This file just invokes c:\lynx.bat that will perform complete lynx booting process.

# Description of files in \win directory (PC)

**Dlynx.bat**

Bat file that starts lynx with non-compact screen layout and screen sized 25x80

**Hydict**

Hyphenation dictionary

**Jumps.htm**

Example of jump file

**Lynx.cfg**

Lynx's configuration file

**Plynx.bat**

Bat file that starts lynx with compact screen layout (as used on Falcons) and screen sized 16x20 (as on Falcon 325)

**Start.htm**

Default start file (will be loaded by lynx if you don't define another in lynx.cfg)

**_dolynx.bat**

Low level bat file used to start lynx that is called by other bat files.

**_lynx.exe**

Executable of Lynx with Percon extensions for Windows

**_mime.typ**

File that defines mapping of file's extension to its MIME type (used to decide whether lynx can render it or prompt to confirm downloading)

**demo.htm**

File that shows some of Percon's extensions to html

**icons**

Directory that contains icons that can be used for lynx shortcut from various sources

**lynx_help**

Directory with documentation distributed with original lynx

# B DETAILED FILE DESCRIPTIONS

LYNX

# Description of Configuration & Batch Files

## Configuration Files

The following configuration files are shipped with lynx for Falcon:

- lynx.cfg - main configuration file - it's read by lynx at startup. It directs lynx to include 2 other configuration files listed just below. They have the same syntax as lynx.cfg.
- screen.cfg - configuration file that is included by lynx.cfg. Its content depends on the model of Falcon (in fact, there is a unique copy of screen.cfg for each model of Falcon in the distribution). In general, it defines screen dimensions for that model of Falcon and colors for various elements of lynx's screen that are tuned for the screen of that model.
- local.cfg - configuration file that is included by lynx.cfg last (this way the value of any setting set in lynx.cfg and screen.cfg can be overridden). This is the only file which is designed to be modified by user (i.e. it's safe to edit local.cfg, while lynx.cfg and screen.cfg can be changed in future versions of lynx for Falcon).

### *Lynx.cfg, Screen.cfg, Local.cfg*

Lynx.cfg is the main configuration file. It's read by lynx at startup. The syntax of configuration files understood by lynx allows directing to include content of another configuration file. Due to this ability, all options are spread across three configuration files for more flexibility (each of the three files has the same syntax):

- screen.cfg - configuration file with content that depends on the model of Falcon. In general, it defines screen dimensions for that model and colors for various elements of lynx's screen that are tuned for screen of that model.
- local.cfg - the only configuration file that is dedicated to changing by the user. It's included by lynx.cfg at the last line, so it's possible to override values of any setting assigned in lynx.cfg and screen.cfg. If no model specific options are set in local.cfg, it will be model independent - thus it could be written once and copied to each Falcon independent of its model. In particular, http proxy and default file/URL should be specified there. It's possible to edit some values in this configuration file interactively via lynxconf.exe - this program will make all changes to this file by default.
- lynx.cfg - the main configuration file. It's content doesn't depend on the model of Falcon. It's unsafe to make modifications to this file - it's better to make all changes to local.cfg since content of lynx.cfg could change in future versions of lynx for Falcon and you'll have to manually do the same changes in lynx.cfg of future version.

### *Syntax of lynx.cfg and included files*

Definitions have the form of **OPTION:VALUE**. You can use letters of any case in the name of the option. No space around **:** or to the left of **OPTION** is allowed. Comments are introduced by putting **#** in the first column - you can't start comment in any other column. For most options, the value used in last definition of the same setting is taken in effect. For example, the following fragment:

**STARTFILE:start1.htm**
**STARTFILE:start.htm**
**#this is the http proxy to use**
**HTTP_PROXY:http://192.168.100.2:80/**
**#STARTFILE:.**

means that the value of **STARTFILE** will be **start.htm** (since it comes after line **STARTFILE:start1.htm**), the value of **HTTP_PROXY** is **http://192.168.100.2:80/**. The last line in this fragment is ignored since it's commented.

To direct lynx to include another configuration file at some point, use the **INCLUDE** directive. The name of file to be included uses a standard DOS convention of referring to files. If the file to be included resides in the same directory as lynx.cfg is, you can use only the name of that file without directory name. Here is the exact code fragment from **lynx.cfg** that directs **local.cfg** to be included:

INCLUDE:local.cfg

# Batch Files

The following batch files are supplied with lynx for Falcon.

- **c:\lynx.bat** - invoking this batch file is the most correct way of invoking lynx. If you run lynx.bat first time on the given computer, lynxconf will be automatically started to allow you to configure various settings you need to configure (lynxconf won't be automatically started on any subsequent invokations of lynx.bat - to start it manually **cd** to **c:\lynx** and type **lynxconf**).
  This batch file will then insure that RF card drivers are already started (or start them if they haven't started yet). If there is no packet drivers in c:\net\, then fake network driver will be started that allows booting lynx, but without any networking functions. Note: you have to reboot Falcon after installing packet drivers to c:\net if you are

planning to use them with lynx and you have started lynx in current session (since fake network packet driver will interfere with "real" RF card drivers you've installed). And of course **c:\lynx.bat** will start lynx itself after all checks.

- **c:\upgstart.bat** - this file invokes **c:\lynx.bat** - see description of c:\lynx.bat for more information about it. **c:\autoexec.bat** starts **c:\upgstart.bat** at the end - this means that lynx will be automatically started when you boot Falcon.
- **_dolynx.bat** - this is the internal batch file aimed at starting lynx. Its behavior can be controlled via commandline arguments. If first argument is **falcon**, then extended Percon BIOS calls (that provide functions like controlling backlight and checking input source) are enabled for use and compact screen layout is used (otherwise no extended Percon BIOS calls will be enabled and normal screen layout is used). If next argument is **net**, then it will assume that RF card drivers are already loaded, otherwise fake-net packet driver will be loaded before starting lynx and unloaded on exit. All remaining commandline options are passed directly to lynx. Also some environment variables that will be read by lynx are initialized. Since this batch file uses a lot of environment variables, it will most probably run out of environment space if not started with command interpreter with increased space for environment variables - that's what all other batch files that invoke **_dolynx.bat** are doing.
  All other batch files listed below just call **_dolynx.bat** with some commandline options.
- **Plynxn.bat** – starts lynx and directs it to enable extended Percon BIOS calls, use compact screen layout and assume that packet drivers are loaded. This file will pass all its commandline arguments, prepending two arguments **falcon** and **net** to **_dolynx.bat**, that will pass all arguments except prepended to lynx. Note: This batch file is called by **c:\lynx.bat** Note: if no packet drivers were really started, lynx will abort loading.
- **Plynx.bat** – starts lynx and directs it to enable extended Percon BIOS calls, use compact screen layout and start the fake network packet driver. This file will pass all its commandline arguments, prepending argument **falcon** to **_dolynx.bat**, that will pass all arguments except prepended to lynx.
- **dlynx.bat** – starts lynx and directs it to disable extended Percon BIOS calls, use normal screen layout and start that fake network packet driver. This file will pass all its commandline arguments, prepending argument **dos** to **_dolynx.bat**, that will pass all arguments except prepended to lynx.

In general, c:\lynx.bat is the most correct way to start lynx as it handles everything automatically.

# C CONFIGURATION & COMMAND LINE OPTIONS

LYNX

# Falcon Lynx Configuration File Options

**Option name:** compact_layout
**Value format:** TRUE or FALSE
**Default value:** TRUE
**Description:** Whether compact screen layout is used when displaying html documents (e.g. default right and left margin for each style of text, indentation of nested lists, etc). There is a screen.cfg configuration file in the distribution that contains all settings that are Falcon model dependent. The setting in Falcon model dependant screen.cfg has correct value for this setting (screen.cfg for models 615 and 655 have FALSE for this option).

**Option name:** hyphenate
**Value format:** TRUE or FALSE
**Default value:** TRUE
**Description:** Controls whether hyphenation is enabled. Hyphenation will be really enabled only if path to hyphenation dictionary file is correct. This setting is in the LYNX.CFG file.

**Option name:** hyphendict
**Value format:** pathname
**Default value:** hydict
**Description:** Specifies location of hyphenation dictionary. This setting is in the LYNX.CFG file.

**Option name:** nonmatching_textfield_alert
**Value format:** arbitrary string
**Default value:** Field doesn't match pattern
**Description:** Default alert to be issued when textfield doesn't match pattern. This setting is in the LYNX.CFG file.

**Option name:** nonmatching_textfield_traps<
**Value format:** TRUE or FALSE
**Default value:** TRUE
**Description:** Whether user can leave nonmatching text field or not. If TRUE, user can leave nonmatching text field only after 7 attempts of editing and leaving the field, otherwise user will be able to leave nonmatching text field from the first attempt. This setting is in the LYNX.CFG file.

**Option name:** percon_environment
**Value format:** TRUE
**Default value:** TRUE
**Description:** Whether the use of Percon extended BIOS calls is allowed.

**Option name:** screen_cols
**Value format:** NUMBER
**Default value:** 80
**Description:** The number of screen columns to use. There is a screen.cfg configuration file in the distribution that contains all settings that are Falcon model dependent. The setting in Falcon model dependant screen.cfg has correct value for this setting.

**Option name:** screen_lines
**Value format:** NUMBER
**Default value:** 25
**Description:** The number of screen lines to use. There is a screen.cfg configuration file in the distribution that contains all settings that are Falcon model dependent. The setting in Falcon model dependant screen.cfg has correct value for this setting.

# Falcon Lynx Command Line Options

Most of the behavior that is controllable via configuration files is also controllable via command line options on the file _dolynx.bat. Lynx parses the commandline after the configuration file, so if some aspect of behavior is covered by configuration file and command line, behavior requested via command line will override the one in configuration file (i.e. the ones who requests wins).

Description of the format:

The name of the option is after **Option name:**

Information about format of the value for this option is provided after **Value format:** . Options with format stated as SWITCH can have two states: ON and OFF. To ask for ON state, just supply the name of that option on commandline, prepended with one or two hyphens, e.g. **`-hyphenate`**. To ask for OFF state, supply the name of that option on commandline, prepended with one or two hyphens and with one hyphen appended, e.g. **`-hyphenate-`**.

**Configuration file analog:** points to the name of configuration file option that controls the same aspect of behavior. Consult the description of configuration file option first.

Description of the command line option is provided after **Description:**

Here is the list of options:

**Option name:** falcon_stylesheet
**Value format:** SWITCH
**Configuration file analog:** compact_layout
**Description:** See description of corresponding configuration file option.

**Option name:** hyphenate
**Value format:** SWITCH
**Configuration file analog:** hyphenate
**Description:** See description of corresponding configuration file option.

**Option name:** hyphendict
**Value format:** =filename
**Configuration file analog:** hyphendict
**Description:** See description of corresponding configuration file option.

**Option name:** netstate_file
**Value format:** =filename
**Configuration file analog:** <NONE>
**Description:** Filename to which information about connection to packet driver is stored and from which information is read. If this option is not used, then no information about packet driver will be stored and no information will be read. This is only useful when debugging lynx. When lynx crashes, the packet driver handles are not released. After starting lynx again (without rebooting) packet driver will reject attempt to get handles from lynx since packet driver doesn't allow several clients to use one network card at the same time. If this option is used, information about connection to packet driver (IRQ and handles) are stored into file. If lynx finds during booting that something is saved to that file, it will read that information and will try to release packet driver handles as stored in the file. The alternative to the use of this option is rebooting after each lynx' crash.

**Option name:** percon
**Value format:** SWITCH
**Configuration file analog:** percon_environment
**Description:** See description of corresponding configuration file option.

# D DESIGNING FOR THE FALCON LYNX BROWSER

LYNX

# Key Lynx Features

Here is a list of lynx features that you should keep in mind while designing pages for lynx:

- Lynx is a text mode browser. It can't show any images. It can use only one type of font - the one that is currently loaded into video adapter, so you can safely remove all <FONT> tags to reduce the size of HTML file.
- Lynx ignores all colors you ask for in your HTML so don't use them to reduce size of HTML file transferred. It's impossible to alter background color via HTML too.

  Lynx renders text inside   in different color, so use   to make some parts of text distinct from other text. Lynx doesn't use different colors for visited and for non-visited links.
- Lynx doesn't support frames in their usual sense. If file contains frameset, each file referenced in frameset will be present as a link so user will be able to visit them.
- Lynx doesn't support CSS (it makes no sense for text-based browser since most information provided via CSS is mostly oriented toward visual representation).
- Lynx doesn't support client-side scripting (i.e. no Javascript or Java is supported by lynx).
- In most cases lynx won't render tables in the form we are used to. Lynx can render tables as we are used to - it will happen if entire table can be rendered without wrapping content of the cells and if each cell takes at most one line. If this condition is not satisfied, lynx will separate cells' content with spaces, and new line will be started for each row.
- HTML includes provision for passing instructions to clients via directives in META elements, and one such instruction, via the token *Refresh*, should invoke reloading of the document, fetched from a server with the same URL or a new URL, at a specified number of seconds following receipt of the current document.

  Lynx recognizes and processes *Refresh* directives in META elements, but puts up a labeled link, typically in the upper left corner of the display, indicating the number of seconds intended before a refresh, and the URL for the refresh, instead of making the request automatically after the indicated number of seconds. If you need to redirect, use HTTP redirects for this purpose.
- Lynx accepts gzipped (compressed) files and decompress them on the fly. So you can use this feature to speed up transfer time for some pages.
- Lynx doesn't support XML - so don't (ab)use software products that generate it (in the best case you'll get highly bloated documents, in the worst lynx won't understand part of markup).

- Lynx has justification (aligning left and right side of text in the paragraph to the vertical line by inserting extra spaces between words) enabled by default (it can be turned off only in configuration file). In general, most people think that justification improves the look of the text, but it can be undesirable in some situations when screen is not wide. To suppress justification, specify paragraph alignment other than “left” and “justify” (i.e. “right” or “center”).

# Basic Lynx Design Tips

The following bullets describe some basic design tips for all pages intended to be displayed on a Lynx browser (Falcon or not).

- If the text typed into some textinput will be short, the use of the attribute named “size” of the INPUT tag (it directs that input field should show only that many characters at the same time - e.g. it tells how long will be form field, using character width as the length unit) - such “shortening” of textinput can allow placing several textinputs on the same line.

  Don't forget that attribute “size” doesn't limit the maximum number of characters allowed - “maxlength” does.

- Try to avoid tables in your pages. If some information is better to be presented in table with 2 columns, you can use <DL> (definition list) - it requires less width of screen and looks somewhat like a table. Here is an example of the HTML code:

```
<dl>
    <dt>IP_ADDRESS</dt><dd>192.168.23.43</dd>
    <dt>DNS_IP</dt><dd>192.168.23.20</dd>
</dl>
```

  Put text of the left column inside <dt> </dt> and text of the right column inside <dd> </dd>

- If some form contains only text input (and no other form fields at all except buttons), it's possible to get rid of the “submit” button - lynx will submit that form when user presses “enter” while in that input field.

- You can compose jumpfile for easy navigation of pages of your site - jumpfiles is a feature unique to lynx that allows to associates URLs with strings that identify them. Pressing 'j' and then typing the identifier will load the URL associated with the identifier (you can define one-letter identifiers for most main screens). See example file “jumps.htm” supplied with your links for details on the syntax. You have to add **`JUMPFILE:filename`** to your local configuration file (local.cfg) to use that jumpfile.

- Lynx has support for extended functionality called RULE-based URL rewriting. It allows to direct lynx to rewrite some URLs that it is going to load according to rules (rule apply to the given URL if URL matches the pattern associated with that rule). Consult file cernrules.txt for more information.
- The textareas are growing automatically in lynx (when user presses enter in the last line of textarea). If a sequence of values is to be input from user and the order is not significant (like barcodes of the goods in the given sector of warehouse) it may be better to accumulate all input to one textarea and submit it when filled, rather than submitting each barcode in a separate form with the single text input.

# Debugging Scripts and Pages with Lynx

- You can use 'Ctrl-R' to reload document (and 'Ctrl-L' to redraw).
- Preview your pages with lynx. There is a lynx with Percon's extensions for Windows available, that can be controlled to use any screen size, so it's possible to see how your pages will look like on a screens with differ
- You can inspect the source of the HTML lynx displays currently by pressing '\' (single forward slash without quotes). To return back to rendered version, press it again.
- There is a quick way to check whether lynx understands given tag or attribute or named character reference or not - just switch to source mode (using '\') - all tags/ attributes/named references will be shown in color different from color of elements lynx understands.
- You can save the rendered version of the document to the text file - press single letter 'p', press 'enter' and type in file name to save to. To save the source to the file, there are two ways - press 'V' then 'd' and the file will be retrieved and will be saved with some headers (like <BASE HREF=".."> to preserve validity of links; charset info will also be saved) or another way is switching to source and printing it to file (pressing 'p' while viewing source) - you will get original file then.
- It's possible to provide any User-Agent string for use by lynx, so if you are having trouble with some library that works fine with other browsers, try spoofing MSIE or NN to see if this helps.
- You can view the cookies lynx accumulated during current session (and even delete them selectively) by going to *Cookie Jar* screen by pressing 'Ctrl-k'.
- You can scroll by two lines up or down (instead of by a screen) by pressing INS or DEL. You can scroll by half of a screen by pressing '(' and ')'.
- You can send HEAD request for the current document to the HTTP server (and look at the response) by pressing ']'.

- To see full URL of the hyperlink destination or all vital information lynx gathered about your document (like expiry date, character set) press '=' - URL of the link will be at the bottom of the screen, URL and info about current document will be at the top of the screen.
- Press 'k' to see all commands and keys they are accessible by.
- Lynx has a special “direct navigation” facilities that allow you to go or follow to the link by its absolute or relative number, and go to any page by its absolute or relative number (this can be useful if you are debugging the script). Items in SELECT lists can be selected using this facility too. See lynx_help/keystrokes/follow_help.html for more information.

# Modifying Page Layout

The following sections will help you refine the layout and presentation of your HTML intended for display on the Lynx browser.

## *General information*

The screen is 20 characters wide on some models of Falcon. Designing web pages for such screen size can be somewhat tricky. In some cases you will find that you wish to know about text rendering approach lynx uses (most specifically, the way hyphenation and justification are applied). You will find this information here.

## *Justification*

When rendering, lynx tries to fit as much text on one line as possible. Hyphenation (splitting words where it's allowed by the grammar rules) is the aid lynx uses. Lynx applies hyphenation everywhere (if you allowed), except pre-formatted text (text in <pre>). In order to fit more text on one line, lynx accumulates words that fit on current line, and tries to hyphenate the word that “crosses” the right margin of text, trying to leave as long part on the “current” line as possible. If such hyphenation is possible, hyphen character (just a dash) is added, and remaining part of the word starts on the next line. Previous line (which is already filled with text) is justified if justification is allowed for the current type of paragraph and if the resultant line doesn't look very “sparse”. Lynx won't justify pre-formatted text and the paragraphs that are right-aligned or centered. If justification is allowed for the current type of the paragraph, then lynx counts the number of spaces it has to add to the line and then, if the resultant line won't be very sparse, this number of spaces is spread as equally as possible between the words. Lynx won't add any spaces between words that are separated with non-breakable space ( ).

To decide whether the line will be sparse, lynx calculates the percentage of the spaces added with respect to the length of the justified line. If this value (in percents) is greater than the value specified by configuration file setting named “justify_max_void_percent”, then lynx concludes that the line will be too sparse and it won't justify that line. So, to control the sparsity of lines allowed alter the value of justify_max_void_percent in some configuration file (default value is 35 - i.e. the total size of the spaces spread across line can't exceed 35 percents of the maximum line length).

If some sentence doesn't look perfect on the narrow screen of Falcon, try paraphrasing that sentence or inserting articles (“the” or “a”).

## *Falcon Design Issues*

The previous section discussed general design issues for all Lynx browsers. This section will discuss Lynx design issues specific to the Falcon.

- It's obvious that user won't be able to use mouse.
- As was stated earlier, if the form contains only one input field and probably buttons, it's possible to get rid of “submit” button - lynx will submit form when user presses “enter” in that text input. If user can use scanner to fill this field, then getting rid of submit button is highly recommended - just remember that scanner simulates enter keypress at the end - this means that in case of scanned input, form will be submitted right after scanning automatically!
- If 'maxlength' attribute is specified (with valid number as value) and if “autoenter” attribute is used, then lynx will try to submit the form when the number of characters specified by maxlength attribute entered. This is not useful for pure scanner-only text inputs (since scanner simulates 'enter' after good scan), but can be used when input from user is allowed and when the length of text to be entered is known.
- Studying Extended Regular Expressions (EREs) is highly recommended - this is a powerful language for expressing patterns, it's actively used in unix tools and mature languages like Perl. When using EREs with alternation, don't forget to enclose each alternative and entire list of alternatives in the parentheses. For example, to compose the pattern that will accept only the following text: *either 2 to 4 letters in range from 'a' to 'f', or number from 10 to 18*, **`([a-f]{2,4})|(1[0-8])`** will be incorrect pattern (it will allow, say **`aaaaaaaaaaa`**), the correct pattern is **`(([a-f]{2,4})|(1[0-8]))`** - i.e. that “wrong” pattern enclosed in parentheses.

  Here is an example of pattern that will allow only digits in input: **`[0-9]+`**.
- Remember to correctly quote the value of all attributes (including patterns and alerts).

When writing secure, safe and reliable CGI scripts that handle content from forms with textfields that were subject to pattern matching, don't assume that all content submitted to the script matched the patterns since synthesizing form content and submitting it is one of the most used cracker's approaches.

**Note**: lynx has a special keystroke 'ctrl-v' called 'execute lynx command' - pressing this keystroke while editing textfield will direct lynx to treat any keystroke pressed after this special keystroke interpret that following keystroke as in normal navigation mode. E.g. while in textfield pressing 'ctrl-v' and then 'q' will show 'do you really want to quit' prompt instead of inserting letter 'q' into text field. Pressing 'ctrl-v' and then arrow up or arrow down while editing textfield will move the focus to the previous or next link correspondingly, without applying any pattern matching procedure. So don't treat pattern matching facilities of lynx for Falcon as complete mean for rejecting invalid input - just treat it as a way to reduce network traffic spent for pages with alerts.