                     Uniform Resource Locators (URL)

Status of this Memo

Abstract

   This document specifies a Uniform Resource Locator (URL), the syntax
   and semantics of formalized information for location and access of
   resources via the Internet.

1. Introduction

   This document describes the syntax and semantics for a compact string
   representation for a resource available via the Internet. These
   strings are called "Uniform Resource Locators" (URLs).

   The specification is derived from concepts introduced by the World-
   Wide Web global information initiative, whose use of such objects
   dates from 1990 and is described in "Universal Resource Identifiers
   in WWW", RFC 1630. The specification of URLs is designed to meet the
   requirements laid out in "Functional Requirements for Internet
   Resource Locators" [12].

   This document was written by the URI working group of the Internet
   Engineering Task Force. Comments may be addressed to the editors, or
   to the URI-WG <uri@bunyip.com>. Discussions of the group are archived
   at <URL:http://www.acl.lanl.gov/URI/archive/uri-archive.index.html>

2. General URL Syntax

resources.

The generic syntax for URLs provides a framework for new schemes to
be established using protocols other than those defined in this
document.

URLs are used to `locate' resources, by providing an abstract
identification of the resource location. Having located a resource,
a system may perform a variety of operations on the resource, as
might be characterized by such words as `access', `update',
`replace', `find attributes'. In general, only the `access' method
needs to be specified for any URL scheme.

2.1. The main parts of URLs

A full BNF description of the URL syntax is given in Section 5.

In general, URLs are written as follows:

    <scheme>:<scheme-specific-part>

A URL contains the name of the scheme being used (<scheme>) followed
by a colon and then a string (the <scheme-specific-part>) whose
interpretation depends on the scheme.

Scheme names consist of a sequence of characters. The lower case
letters "a"--"z", digits, and the characters plus ("+"), period
("."), and hyphen ("-") are allowed. For resiliency, programs
interpreting URLs should treat upper case letters as equivalent to
lower case in scheme names (e.g., allow "HTTP" as well as "http").

2.2. URL Character Encoding Issues

URLs are sequences of characters, i.e., letters, digits, and special
characters. A URLs may be represented in a variety of ways: e.g., ink
on paper, or a sequence of octets in a coded character set. The
interpretation of a URL depends only on the identity of the
characters used.

In most URL schemes, the sequences of characters in different parts
of a URL are used to represent sequences of octets used in Internet
protocols. For example, in the ftp scheme, the host name, directory
name and file names are such sequences of octets, represented by
parts of the URL. Within those parts, an octet may be represented by

the chararacter which has that octet as its code within the US-ASCII
[20] coded character set.

In addition, octets may be encoded by a character triplet consisting
of the character "%" followed by the two hexadecimal digits (from
"0123456789ABCDEF") which forming the hexadecimal value of the octet.
(The characters "abcdef" may also be used in hexadecimal encodings.)

character within the US-ASCII coded character set, if the use of the
corresponding character is unsafe, or if the corresponding character
is reserved for some other interpretation within the particular URL
scheme.

No corresponding graphic US-ASCII:

URLs are written only with the graphic printable characters of the
US-ASCII coded character set. The octets 80-FF hexadecimal are not
used in US-ASCII, and the octets 00-1F and 7F hexadecimal represent
control characters; these must be encoded.

Unsafe:

Characters can be unsafe for a number of reasons. The space
character is unsafe because significant spaces may disappear and
insignificant spaces may be introduced when URLs are transcribed or
typeset or subjected to the treatment of word-processing programs.
The characters "<" and ">" are unsafe because they are used as the
delimiters around URLs in free text; the quote mark (""") is used to
delimit URLs in some systems. The character "#" is unsafe and should
always be encoded because it is used in World Wide Web and in other
systems to delimit a URL from a fragment/anchor identifier that might
follow it. The character "%" is unsafe because it is used for
encodings of other characters. Other characters are unsafe because
gateways and other transport agents are known to sometimes modify
such characters. These characters are "{", "}", "|", "\", "^", "~",
"[", "]", and "`".

All unsafe characters must always be encoded within a URL. For
example, the character "#" must be encoded within URLs even in
systems that do not normally deal with fragment or anchor
identifiers, so that if the URL is copied into another system that
does use them, it will not be necessary to change the URL encoding.

Reserved:

Many URL schemes reserve certain characters for a special meaning:
their appearance in the scheme-specific part of the URL has a
designated semantics. If the character corresponding to an octet is
reserved in a scheme, the octet must be encoded. The characters ";",
"/", "?", ":", "@", "=" and "&" are the characters which may be
reserved for special meaning within a scheme. No other characters may
be reserved within a scheme.

Usually a URL has the same interpretation when an octet is
represented by a character and when it encoded. However, this is not

Thus, only alphanumerics, the special characters "$-_.+!*'(),", and
reserved characters used for their reserved purposes may be used
unencoded within a URL.

On the other hand, characters that are not required to be encoded
(including alphanumerics) may be encoded within the scheme-specific
part of a URL, as long as they are not being used for a reserved
purpose.

2.3 Hierarchical schemes and relative links

In some cases, URLs are used to locate resources that contain
pointers to other resources. In some cases, those pointers are
represented as relative links where the expression of the location of
the second resource is in terms of "in the same place as this one
except with the following relative path". Relative links are not
described in this document. However, the use of relative links
depends on the original URL containing a hierarchical structure
against which the relative link is based.

Some URL schemes (such as the ftp, http, and file schemes) contain
names that can be considered hierarchical; the components of the
hierarchy are separated by "/".

3. Specific Schemes

The mapping for some existing standard and experimental protocols is
outlined in the BNF syntax definition. Notes on particular protocols
follow. The schemes covered are:

ftp   File Transfer protocol
http   Hypertext Transfer Protocol
gopher   The Gopher protocol
mailto   Electronic mail address
news   USENET news
nntp   USENET news using NNTP access
telnet   Reference to interactive sessions
wais   Wide Area Information Servers
file   Host-specific file names
prospero   Prospero Directory Service

some scheme names that are under development.

3.1. Common Internet Scheme Syntax

    While the syntax for the rest of the URL may vary depending on the
    particular scheme selected, URL schemes that involve the direct use
    of an IP-based protocol to a specified host on the Internet use a
    common syntax for the scheme-specific data:

        //<user>:<password>@<host>:<port>/<url-path>

    Some or all of the parts "<user>:<password>@", ":<password>",
    ":<port>", and "/<url-path>" may be excluded. The scheme specific
    data start with a double slash "//" to indicate that it complies with
    the common Internet scheme syntax. The different components obey the
    following rules:

     user
         An optional user name. Some schemes (e.g., ftp) allow the
         specification of a user name.

     password
         An optional password. If present, it follows the user
         name separated from it by a colon.

    The user name (and password), if present, are followed by a
    commercial at-sign "@". Within the user and password field, any ":",
    "@", or "/" must be encoded.

    Note that an empty user name or password is different than no user
    name or password; there is no way to specify a password without
    specifying a user name. E.g., <URL:ftp://@host.com/> has an empty
    user name and no password, <URL:ftp://host.com/> has no user name,
    while <URL:ftp://foo:@host.com/> has a user name of "foo" and an
    empty password.

     host
         The fully qualified domain name of a network host, or its IP
         address as a set of four decimal digit groups separated by
         ".". Fully qualified domain names take the form as described
         in Section 3.5 of RFC 1034 [13] and Section 2.1 of RFC 1123
         [5]: a sequence of domain labels separated by ".", each domain
         label starting and ending with an alphanumerical character and
         possibly also containing "-" characters. The rightmost domain
         label will never start with a digit, though, which
         syntactically distinguishes all domain names from the IP
         addresses.

     port
         The port number to connect to. Most schemes designate
         protocols that have a default port number. Another port number

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.