Cookies

Cookies

Implementation Progress

I started work on this in a local branch. Patches for the changes made there can be found here:
http://people.apache.org/~jboynes/patches/
Of these, patches 01 to 12 have been applied.

There is substantial refactoring in there to simply the current implementation. Actual changes are:

- C3 '=' is now disallowed in Netscape cookie names (it was already not allowed in RFC2109 names)
- C4 Attribute names are allowed as cookies names
- Cookie names starting with '$' are allowed in Netscape and RFC6265 mode and will still throw an IAE in RFC2109 mode

Round Trip Behaviour

The following tables document how a value is sent in a Set-Cookie header, what gets stored by a typical browser, the Cookie header that is generated by the browser and then the final value returned to a Servlet application.

The browser tested here is Chrome-31

Default Configuration (no properties set)

	Generation		Browser Value	Parsing	
Version	**Value**	**Set-Cookie Header**	**Cookie Header**	**Resulting Value**	
0	bar	test=bar	bar	test=bar	bar
0	"bar"	test="bar"	"bar"	test="bar"	bar
0	""	test=""	""	test=""	*emptyString*
0	a"b	test="a\"b"; Version=1	"a\"b"	test="a\"b"	a"b
0	a\b	test="a\b"; Version=1	"a\b"	test="a\b"	ab
0	a?b	test="a?b"; Version=1	"a?b"	test="a?b"	a?b
1	bar	test=bar; Version=1	bar	test=bar	bar

ALLOW_HTTP_SEPARATORS_IN_V0=true

	Generation		Browser Value	Parsing	
Version	**Value**	**Set-Cookie Header**	**Cookie Header**	**Resulting Value**	
0	bar	test=bar	bar	test=bar	bar
0	"bar"	test="bar"	"bar"	test="bar"	bar
0	""	test=""	""	test=""	*emptyString*
0	a"b	test=a"b	a"b	test=a"b	a"b
0	a\b	test=a\b	a\b	test=a\b	ab
0	a?b	test=a?b	a?b	test=a?b	a?b
0	a;b	test="a;b"; Version=1	"a	test="a	*no cookie*
0	a,b	test="a,b"; Version=1	"a,b"	test="a,b"	a,b
1	bar	test=bar; Version=1	bar	test=bar	bar

Proposed Changes

The intent of the following changes is improve interoperability of cookies with other servers and with client-side JavaScript. The primary changes are a switch to the RFC6265 format for transmission of V0 cookies to be more in line with browser behaviour, and support for UTF-8 encoded values that are now specified by HTML-5.

This is very preliminary and intended primary to focus discussion on something concrete now the behavior has been clarified.

Changes to Cookie class

C1 Stricter default validation of name::

- *

${renderedContent}

- *

${renderedContent}

- *

${renderedContent}

C2 Always allow "/" in Netscape cookie names::

- *

${renderedContent}

C3 Always disallow "=" in Netscape cookie names::

- *

${renderedContent}

C4 Always allow attribute names (e.g. "Expires") as cookie names::

- *

${renderedContent}

C5 Allow unnamed cookies in C1b "netscape" mode::

- *

${renderedContent}

Changes to generation of Set-Cookie header

G1 Use RFC6265 format header for V0 cookies::

- *

${renderedContent}

- *

${renderedContent}

G2 Use RFC2109 format header only for V1 cookies::

- *

${renderedContent}

G3 Stop adding quotes or escaping to values::

- *

${renderedContent}

- *

${renderedContent}

G4 Use UTF-8 encoding for values::

- *

${renderedContent}

- *

${renderedContent}

```
message-header = field-name ":" [field-value ]

field-value = *( field-content | LWS )

field-content = <the OCTETs making up the field-value and consisting of either *TEXT or combinations of token,
separators, and quoted-string>
```

The tokens are US-ASCII (0-127 minus CTLs or separators) (pages 16-17).

The TEXT is defined on page 16 where it says: "Words of *TEXT MAY contain characters from character sets other than ISO-8859-1 [22] only when encoded according to the rules of RFC 2047 [14]."

The quoted-string is TEXT in double quotes (page 16).

- *

${renderedContent}

G5 Validate domain per RFC6265::

- *

${renderedContent}

G6 Do not quote Path values for V0 cookies::

- *

${renderedContent}

G7 Always set both Max-Age and Expires as a pair::

- *

${renderedContent}

Changes to parsing of Cookie header

P1 Parse non-RFC2109 cookies using a lenient RFC6256 parser::

- *

${renderedContent}

The lenient parser for V0 cookies will allow:

- A cookie name will accept all CHARs up to semicolon or equals. Leading and trailing spaces will be trimmed. This relaxes RFC6265 to accommodate additional characters browsers can use in names but does not allow for 8-bit characters in names.
- A cookie value will accept all octets up to a semicolon decoded using UTF-8. Leading and trailing spaces will be trimmed. This relaxes RFC6265 to accommodate HTML5.
    - **Issue:** Any commas added by header folding prior to receipt will be accepted as part of the previous cookie's value. This is indicative of a mis-behaving proxy or user-agent and conflicts with browsers that accept commas in values.
    - **Issue:** There is a conflict with Safari (WebKit?) here in that it does allow semicolons in values that start/end with DQUOTE. However, the other browsers will truncate the value at the semicolon when it is being set resulting in a mismatched pair.
- Any invalid character will result in the cookie being dropped. Parsing will restart at the next semicolon.
- Any cookie whose name begins with "$" will be dropped to avoid confusion with RFC2109 attributes. RFC6265 and Netscape do not support $Path and $Domain attributes and so any value they define will not be used to set fields in a Cookie object.
- Cookies whose name matches an attribute (e.g. "Expires") will be permitted.

This generally matches the rules defined by Netscape and RFC6265 for the Cookie header.

P2 Parse RFC2109 requests (determined by the presence of a "$Version=1" attribute) using the current parser::

- *

${renderedContent}

- *

${renderedContent}

P3 Do not throw IAE from the parser::

- *

${renderedContent}

P4 Ensure that the cookie header is always available for the application to parse manually.::

- *

${renderedContent}

Impact of proposal on existing issues

Issue	Impact
Bug 55917	Parsing will no longer cause an IAE. 8-bit values will be interpreted as a UTF-8 value and the cookie would be dropped if they are not a valid encoding.

Bug 55918	The cookie would be dropped rather than accepted.
Bug 55920	Valid values would be round tripped including quotes supplied by the application. Attempts to set invalid values would result in a IAE from addCookie. Invalid values sent by the browser would result in the cookie being ignored.
Bug 55921	Attempts to set a cookie containing raw JSON would results in an IAE due to the DQUOTE characters. A cookie sent from the browser containing JSON would be accepted although any semicolons in the data would result in early termination (note, browsers other than Safari do not allow semicolons in values anyway).

Parsing the Cookie header by Tomcat

The various specifications define the following formats for the Cookie header sent by the user-agent:

Specification	Format of Cookie header	
Netscape	`Cookie: NAME1=OPAQUE_STRING1; NAME2=OPAQUE_STRING2 ...`	
RFC2109	{{"Cookie:" "$Version" "=" value 1*((";"	",") cookie-value)}}
RFC6265	`"Cookie:" OWS cookie-pair *( ";" SP cookie-pair ) OWS`	

Chrome-31, Firefox-26, Firefox Aurora-28, Internet Explorer-11 and Safari-7.01 all send a single header in Netscape/RFC6265 format with name=value pairs separated by semicolon and space. The name and value correspond to whatever was stored in the browser when the "Set-Cookie" header was parsed. These may contain commas, spaces, other separators or 8-bit characters.

None of them add any of the "$" attributes ("$Version" "$Domain" or "$Path) from RFC2109 and specifically do not send the leading "$Version" attribute that is part of that specification's syntax. All except Safari support a unnamed "value-only" cookie that is sent as is (without a name or "="); i.e. a unnamed cookie with value "foo" (including quotes) is sent as the line:

```
Cookie: "foo"
```

When set through JavaScript, any Unicode codepoints in the text are encoded as UTF-8 in the header. For example, in Chrome the statement `document.cookie = "foo=b\u00e1r";` will result in a header containing the octets

```
43 6f 6f 6b 69 65 3a 20 66 6f 6f 3d 62 c3 a1 72
```

showing codepoint U+00E1 being converted to its UTF-8 equivalent 0xC3 0xA1. This matches the behaviour defined by HTML5.

Issue	Current behaviour (8.0.0-RC10/7.0.50)	Proposed new behaviour	Servlet + Netscape + RFC2109	Servlet + RFC 6265
0x80 to 0xFF in cookie value (Bug 55917)	IAE	TBD	Netscape yes. RFC2109 requires quotes.	RFC 6265 never allowed.
CTL allowed in quoted cookie values (Bug 55918)	Allowed	TBD	Not allowed.	Not allowed.
Quoted values in V0 cookies (Bug 55920)	Quotes removed.	TBD	Netscape - quotes are part of value.	Quotes are not part of value.
Raw JSON in cookie values (Bug 55921)	TBD	TBD	TBD	TBD
Allow equals in value	Not by default. Allowed if property set.	TBD	Netscape is ambiguous. RFC2109 requires quoting.	Allowed.
Allow separators in V0 names and values	Not by default. Allowed if property set.	TBD	Yes except semi-colon, comma and whitespace.	Never in names. Yes in values except semi-colon, comma and whitespace, double-quote and backslash.
Always add expires	Enabled by default. Disabled by property.	TBD	Netsacpe uses expires. RFC2109 uses Max-Age.	Allows either, none or both.
/ is separator	Enabled by default. Disabled by property.	TBD	Netscape allowed in names and values. RFC2109 allowed in values if quoted.	Allowed in values.
Strict naming (as per Servlet spec)	Enabled by default. Disabled by property.	TBD	Netscape allows names the Servlet spec does not. RFC2109 is consistent with the Servlet spec.	Consistent with the Servlet spec.
Allow name only	Disabled by default. Enabled by property.	TBD	Netscape allowed and equals sign expected before empty value. RFC2109 not allowed.	Allowed but equals sign required before empty value.

Issues to add to the table above

- Bug 55951 regarding UTF-8 encoded values from HTML5
- Any further issues raised on mailing lists

Generating the Set-Cookie header by Tomcat

Requirements as defined by the specifications

Requirement	Servlet	Netscape	RFC2109	RFC6265	
Format of name	Must conform to RFC2109. Vendors may provide option to allow Netscape format	A sequence of characters excluding semi-colon, comma and white space. Browsers generally stop at first equals,	token	token	
Format of value	The value can be anything the server chooses to send. With Version 0 cookies, values should not contain white space, brackets, parentheses, equals signs, commas, double quotes, slashes, question marks, at signs, colons, and semicolons. Empty values may not behave the same way on all browsers.	This string is a sequence of characters excluding semi-colon, comma and white space.	token	quoted-string	cookie-value
Domain	String, per RFC2109	domain=DOMAIN_NAME	"Domain" "=" value	"Domain=" domain-value	
Path	String, per RFC2109	path=PATH	"Path" "=" value	"Path=" path-value	
Secure	boolean	secure	"Secure"	"Secure"	
Http Only	boolean	N/A	N/A	"HttpOnly"	
Expires	N/A	expires=DATE as "Wdy, DD-Mon-YYYY HH:MM:SS GMT"	N/A	"Expires=" sane-cookie-date	
Max-Age	int in seconds	N/A	"Max-Age" "=" value	"Max-Age=" non-zero-digit *DIGIT	
Comment	String	N/A	"Comment" "=" value	allowed by extension	
Version	int (0 or 1)	N/A	"Version" "=" 1*DIGIT	allowed by extension	
Extension	N/A	N/A	N/A	any CHAR except CTLs or ";"	

The RI defines a vendor system property "org.glassfish.web.rfc2109_cookie_names_enforced" (default true) that controls the characters permitted in the name argument. If true, RFC2616 separators (including "/") will trigger an IllegalArgumentException; if false, only comma, semicolon and space are considered invalid.

Current Implementation

Cookie

The constructor of javax.servlet.http.Cookie will throw an IllegalArgumentException if any of the following conditions are met:

- name is null or zero length
- if name is not a token
- if name equalsIgnoreCase any of "Comment" "Discard" "Domain" "Expires" "Max-Age" "Path" "Secure" "Version"
- if name startsWith "$"

By default, a token comprises characters 0x21..0x7E except comma, semicolon and space. If STRICT_NAMING is true, then token also excludes characters from "()<>@,;:\\\"[]?={} \t" which corresponds to RFC2616 separators without "/" (i.e. "/" is allowed); if FWD_SLASH_IS_SEPARATOR is true than "/" is also excluded. These properties will default to true if STRICT_SERVLET_COMPLIANCE is true.

Issues

${renderedContent}

- *

${renderedContent}

- *

${renderedContent}

No checks are made in any of the other setters.

The domain value is converted to lower case (per Locale.ENGLISH) when set as "IE allegedly needs this."

Neither the Cookie class or any of its methods are declared final so any of this behaviour can be overridden if an application sub-classes Cookie; for example, the checks performed on the name can be bypassed by overriding the getName() method.

HttpServletResponse

This is typically implemented by o.a.c.connector.Response whose addCookie method delegates generation of the Set-Cookie header to o.a.t.util.http.ServerCookie#appendCookieValue. This first appends the name (relying on checks performed by Cookie), "=" and then the value using RFC2109 quoting rules:

- if the value is null or empty, append empty quoted-string ""
- if the value starts and ends with "", output as is after escaping any "" characters between the outer quotes
- if ALLOW_HTTP_SEPARATORS_IN_V0 is false and the value contains a RFC2616 separator, output as a quoted-string after escaping "" and force Version=1
- if ALLOW_HTTP_SEPARATORS_IN_V0 is true and the value contains a Netscape separator, output as a quoted-string after escaping "" and force Version=1
- otherwise, output as is

Netscape separators are {',', ';', ' ', '\t'}
RFC2616 separators by default do not include "/" unless FWD_SLASH_IS_SEPARATOR is set (or implied by STRICT_SERVLET_COMPLIANCE).
Characters outside the set { HT, 0x20..0x7E } will result in a IllegalArgumentException when the check for token characters is performed.

The same quoting rules are applied when outputting any Domain or Path value.

If maxAge >=, then the Max-Age attribute will be set for V1 cookies and the Expires attribute for V0 cookies. If the property ALWAYS_ADD_EXPIRES is true then Expires will also be set for V1 cookies.

Issues::

- *

${renderedContent}

- *

${renderedContent}

- *

${renderedContent}

Proposed Implementation

TBD

RFC2616 definitions

```
token          = 1*<any CHAR except CTLs or separators>
separators     = "(" | ")" | "<" | ">" | "@" | "," | ";" | ":" | "\" | <"> | "/" | "[" | "]" | "?" | "=" | "{"
| "}" | SP | HT
CHAR           = <any US-ASCII character (octets 0 - 127)>
CTL            = <any US-ASCII control character (octets 0 - 31) and DEL (127)>
quoted-string  = ( <"> *(qdtext | quoted-pair ) <"> )
qdtext         = <any TEXT except <">>
quoted-pair    = "\" CHAR
TEXT           = <any OCTET except CTLs, but including LWS>
rfc1123-date   = wkday "," SP date1 SP time SP "GMT"
```

RFC2109 definitions

```
cookie-value   = NAME "=" VALUE [";" path] [";" domain]
```

RFC6265 definitions

```
cookie-pair       = cookie-name "=" cookie-value
cookie-value      = *cookie-octet / ( DQUOTE *cookie-octet DQUOTE )
cookie-octet      = %x21 / %x23-2B / %x2D-3A / %x3C-5B / %x5D-7E
domain-value      = <subdomain> ; defined in [RFC1034], Section 3.5, as enhanced by [RFC1123], Section 2.1
path-value        = <any CHAR except CTLs or ";">
sane-cookie-date  = <rfc1123-date, defined in [RFC2616], Section 3.3.1>
```

References

1. RFC6265 discussion on 0x80-0xFF