WsResponse

Apr 03, 2024 22:28

Supported Clients

SmartClient WebClient NGClient

Property Summary

String	characterEncoding	Sets the character encoding (MIME charset) of the response being sent to the client, for example, to UTF-8.
String	contentType	Sets the content type of the response being sent to the client, if the response has not been committed yet.
String	localeLanguageTag	Returns the locale specified for this response using the #setLocale method.
Number	status	Sets the status code for this response.

Methods Summary

void	addCookie(cookie)	Adds the specified cookie to the response.
void	addDateHeader(name, date)	Adds a response header with the given name and date-value.
void	addHeader(name, value)	Adds a response header with the given name and value.
void	addIntHeader(name, value)	Adds a response header with the given name and integer value.
Boolean	containsHeader(name)	Returns a boolean indicating whether the named response header has already been set.
String	getHeader(name)	Gets the value of the response header with the given name.
Array	getHeaderNames()	Gets the names of the headers of this response.
Array	getHeaders(name)	Gets the values of the response header with the given name.
void	sendError(sc)	
void	sendError(sc, msg)	
void	setDateHeader(name, date)	Sets a response header with the given name and date-value.
void	setHeader(name, value)	Sets a response header with the given name and value.
void	setIntHeader(name, value)	Sets a response header with the given name and integer value.

Property Details

characterEncoding

```
Sets the character encoding (MIME charset) of the response
being sent to the client, for example, to UTF-8.
If the character encoding has already been set by
#setContentType(String) or #setLocale,
this method overrides it.
Calling #setContentType(String) with the <code>String</code>
of <code>text/html</code> and calling
this method with the <code>String</code> of <code>UTF-8</code>
is equivalent with calling
<code>setContentType</code> with the <code>String</code> of
<code>text/html; charset=UTF-8</code>.
<p>This method can be called repeatedly to change the character
encoding.
This method has no effect if it is called after
<code>getWriter</code> has been
called or after the response has been committed.
<p>Containers must communicate the character encoding used for
the servlet response's writer to the client if the protocol
provides a way for doing so. In the case of HTTP, the character
encoding is communicated as part of the <code>Content-Type</code>
header for text media types. Note that the character encoding
cannot be communicated via HTTP headers if the servlet does not
specify a content type; however, it is still used to encode text
written via the servlet response's writer.
```

Returns

String

Supported Clients

SmartClient,WebClient,NGClient

Sample

contentType

```
Sets the content type of the response being sent to
the client, if the response has not been committed yet.
The given content type may include a character encoding
specification, for example, <code>text/html;charset=UTF-8</code>.
The response's character encoding is only set from the given
content type if this method is called before <code>getWriter</code>
is called.
<p>This method may be called repeatedly to change content type and
character encoding.
This method has no effect if called after the response
has been committed. It does not set the response's character
encoding if it is called after <code>getWriter</code>
has been called or after the response has been committed.
<p>Containers must communicate the content type and the character
encoding used for the servlet response's writer to the client if
the protocol provides a way for doing so. In the case of HTTP,
the <code>Content-Type</code> header is used.
```

Returns

String

Supported Clients

SmartClient,WebClient,NGClient

Sample

---

localeLanguageTag

```
Returns the locale specified for this response
using the #setLocale method. Calls made to
<code>setLocale</code> after the response is committed
have no effect. If no locale has been specified,
the container's default locale is returned.

<p>If the specified language tag contains any ill-formed subtags,
the first such subtag and all following subtags are ignored.  Compare
to Locale.Builder#setLanguageTag(String) which throws an exception
in this case.

<p>The following <b>conversions</b> are performed:<ul>

<li>The language code "und" is mapped to language "".

<li>The language codes "he", "yi", and "id" are mapped to "iw",
"ji", and "in" respectively. (This is the same canonicalization
that's done in Locale's constructors.)

<li>The portion of a private use subtag prefixed by "lvariant",
if any, is removed and appended to the variant field in the
result locale (without case normalization).  If it is then
empty, the private use subtag is discarded:

<pre>
    Locale loc;
    loc = Locale.forLanguageTag("en-US-x-lvariant-POSIX");
    loc.getVariant(); // returns "POSIX"
    loc.getExtension('x'); // returns null

    loc = Locale.forLanguageTag("de-POSIX-x-URP-lvariant-Abc-Def");
    loc.getVariant(); // returns "POSIX_Abc_Def"
    loc.getExtension('x'); // returns "urp"
</pre>

<li>When the languageTag argument contains an extlang subtag,
the first such subtag is used as the language, and the primary
language subtag and other extlang subtags are ignored:

<pre>
    Locale.forLanguageTag("ar-aao").getLanguage(); // returns "aao"
    Locale.forLanguageTag("en-abc-def-us").toString(); // returns "abc_US"
</pre>

<li>Case is normalized except for variant tags, which are left
unchanged.  Language is normalized to lower case, script to
title case, country to upper case, and extensions to lower
case.

<li>If, after processing, the locale would exactly match either
ja_JP_JP or th_TH_TH with no extensions, the appropriate
extensions are added as though the constructor had been called:

<pre>
   Locale.forLanguageTag("ja-JP-x-lvariant-JP").toLanguageTag();
   // returns "ja-JP-u-ca-japanese-x-lvariant-JP"
```

```
    Locale.forLanguageTag("th-TH-x-lvariant-TH").toLanguageTag();
    // returns "th-TH-u-nu-thai-x-lvariant-TH"
<pre></ul>
```

```
<p>This implements the 'Language-Tag' production of BCP47, and
so supports grandfathered (regular and irregular) as well as
private use language tags.  Stand alone private use tags are
represented as empty language and extension 'x-whatever',
and grandfathered tags are converted to their canonical replacements
where they exist.
```

```
<p>Grandfathered tags with canonical replacements are as follows:
```

```
<table>
<tbody align="center">
<tr><th>grandfathered tag</th><th> </th><th>modern replacement</th></tr>
<tr><td>art-lojban</td><td> </td><td>jbo</td></tr>
<tr><td>i-ami</td><td> </td><td>ami</td></tr>
<tr><td>i-bnn</td><td> </td><td>bnn</td></tr>
<tr><td>i-hak</td><td> </td><td>hak</td></tr>
<tr><td>i-klingon</td><td> </td><td>tlh</td></tr>
<tr><td>i-lux</td><td> </td><td>lb</td></tr>
<tr><td>i-navajo</td><td> </td><td>nv</td></tr>
<tr><td>i-pwn</td><td> </td><td>pwn</td></tr>
<tr><td>i-tao</td><td> </td><td>tao</td></tr>
<tr><td>i-tay</td><td> </td><td>tay</td></tr>
<tr><td>i-tsu</td><td> </td><td>tsu</td></tr>
<tr><td>no-bok</td><td> </td><td>nb</td></tr>
<tr><td>no-nyn</td><td> </td><td>nn</td></tr>
<tr><td>sgn-BE-FR</td><td> </td><td>sfb</td></tr>
<tr><td>sgn-BE-NL</td><td> </td><td>vgt</td></tr>
<tr><td>sgn-CH-DE</td><td> </td><td>sgg</td></tr>
<tr><td>zh-guoyu</td><td> </td><td>cmn</td></tr>
<tr><td>zh-hakka</td><td> </td><td>hak</td></tr>
<tr><td>zh-min-nan</td><td> </td><td>nan</td></tr>
<tr><td>zh-xiang</td><td> </td><td>hsn</td></tr>
</tbody>
</table>
```

```
<p>Grandfathered tags with no modern replacement will be
converted as follows:
```

```
<table>
<tbody align="center">
<tr><th>grandfathered tag</th><th> </th><th>converts to</th></tr>
<tr><td>cel-gaulish</td><td> </td><td>xtg-x-cel-gaulish</td></tr>
<tr><td>en-GB-oed</td><td> </td><td>en-GB-x-oed</td></tr>
<tr><td>i-default</td><td> </td><td>en-x-i-default</td></tr>
<tr><td>i-enochian</td><td> </td><td>und-x-i-enochian</td></tr>
<tr><td>i-mingo</td><td> </td><td>see-x-i-mingo</td></tr>
<tr><td>zh-min</td><td> </td><td>nan-x-zh-min</td></tr>
</tbody>
</table>
```

```
<p>For a list of all grandfathered tags, see the
IANA Language Subtag Registry (search for "Type: grandfathered").
```

```
<p><b>Note</b>: there is no guarantee that <code>toLanguageTag</code>
and <code>forLanguageTag</code> will round-trip.
```

Returns

String

Supported Clients

SmartClient,WebClient,NGClient

Sample

status

Sets the status code for this response.

<p>This method is used to set the return status code when there is
no error (for example, for the SC_OK or SC_MOVED_TEMPORARILY status
codes).

<p>If this method is used to set an error code, then the container's
error page mechanism will not be triggered. If there is an error and
the caller wishes to invoke an error page defined in the web
application, then #sendError(int) must be used instead.

<p>This method preserves any cookies and other response headers.

<p>Valid status codes are those in the 2XX, 3XX, 4XX, and 5XX ranges.
Other status codes are treated as container specific.

Returns

Number

Supported Clients

SmartClient,WebClient,NGClient

Sample

Methods Details

addCookie(cookie)

Adds the specified cookie to the response. This method can be called
multiple times to set more than one cookie.

Parameters

WsCookie cookie the Cookie to return to the client

Supported Clients

SmartClient,WebClient,NGClient

Sample

addDateHeader(name, date)

Adds a response header with the given name and
date-value. The date is specified in terms of
milliseconds since the epoch. This method allows response headers
to have multiple values.

Parameters

String   name the name of the header to set
Number date   the additional date value

Supported Clients

SmartClient,WebClient,NGClient

Sample

addHeader(name, value)

Adds a response header with the given name and value.
This method allows response headers to have multiple values.

Parameters

String name the name of the header
String value the additional header value If it contains octet string, it should be encoded according to RFC 2047 (http://www.ietf.org/rfc/rfc2047.txt)

Supported Clients

SmartClient,WebClient,NGClient

Sample

addIntHeader(name, value)

```
Adds a response header with the given name and
integer value.  This method allows response headers to have multiple
values.
```

Parameters

String   name the name of the header
Number value the assigned integer value

Supported Clients

SmartClient,WebClient,NGClient

Sample

containsHeader(name)

```
Returns a boolean indicating whether the named response header
has already been set.
```

Parameters

String name the header name

Returns

Boolean <code>true</code> if the named response header has already been set; <code>false</code> otherwise

Supported Clients

SmartClient,WebClient,NGClient

Sample

getHeader(name)

```
Gets the value of the response header with the given name.

<p>If a response header with the given name exists and contains
multiple values, the value that was added first will be returned.

<p>This method considers only response headers set or added via
#setHeader(String,String), #addHeader(String,String), #setDateHeader(String,long),
#addDateHeader(String,long), #setIntHeader(String,int), or
#addIntHeader(String,int), respectively.
```

Parameters

String name the name of the response header whose value to return

Returns

String the value of the response header with the given name, or <tt>null</tt> if no header with the given name has been set on this response

Supported Clients

SmartClient,WebClient,NGClient

Sample

getHeaderNames()

```
Gets the names of the headers of this response.

<p>This method considers only response headers set or added via
#setHeader(String,String), #addHeader(String,String), #setDateHeader(String,long),
#addDateHeader(String,long), #setIntHeader(String,int), or
#addIntHeader(String,int), respectively.
```

Returns

Array a (possibly empty) array of the names of the headers of this response

Supported Clients

SmartClient,WebClient,NGClient

Sample

getHeaders(name)

```
Gets the values of the response header with the given name.

<p>This method considers only response headers set or added via
#setHeader(String,String), #addHeader(String,String), #setDateHeader(String,long),
#addDateHeader(String,long), #setIntHeader(String,int), or
#addIntHeader(String,int), respectively.
```

Parameters

String name the name of the response header whose values to return

Returns

Array a (possibly empty) array of the values of the response header with the given name

Supported Clients

SmartClient,WebClient,NGClient

Sample

sendError(sc)

Parameters

Number sc ;

Supported Clients

SmartClient,WebClient,NGClient

Sample

sendError(sc, msg)

Parameters

Number sc   ;
String   msg ;

Supported Clients

SmartClient,WebClient,NGClient

Sample

setDateHeader(name, date)

```
Sets a response header with the given name and
date-value.  The date is specified in terms of
milliseconds since the epoch.  If the header had already
been set, the new value overwrites the previous one.  The
<code>containsHeader</code> method can be used to test for the
presence of a header before setting its value.
```

Parameters

String   name the name of the header to set
Number date   the assigned date value

Supported Clients

SmartClient,WebClient,NGClient

Sample

setHeader(name, value)

```
Sets a response header with the given name and value.
If the header had already been set, the new value overwrites the
previous one.  The <code>containsHeader</code> method can be
used to test for the presence of a header before setting its
value.
```

Parameters

String name the name of the header
String value the header value If it contains octet string, it should be encoded according to RFC 2047 (http://www.ietf.org/rfc/rfc2047.txt)

Supported Clients

SmartClient,WebClient,NGClient

Sample

setIntHeader(name, value)

```
Sets a response header with the given name and
integer value.  If the header had already been set, the new value
overwrites the previous one.  The <code>containsHeader</code>
method can be used to test for the presence of a header before
setting its value.
```

Parameters

String   name the name of the header
Number value the assigned integer value

Supported Clients

SmartClient,WebClient,NGClient

Sample