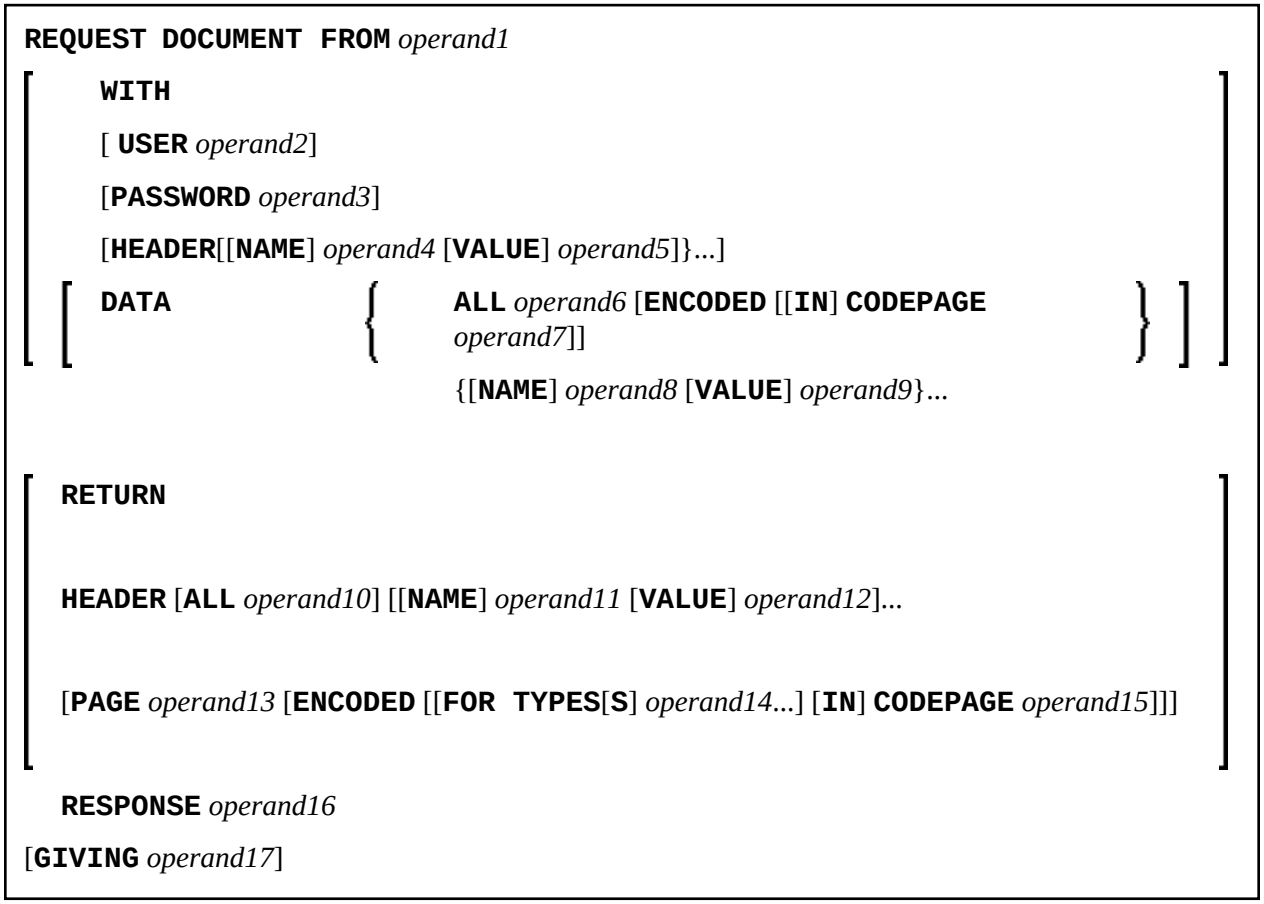
REQUEST DOCUMENT



Dieses Kapitel behandelt folgende Themen:

- Funktion
- Syntax-Beschreibung
- Kodierung von eingehenden/ausgehenden Daten
- Beispiele

Eine Erläuterung der in dem Syntax-Diagramm verwendeten Symbole entnehmen Sie dem Abschnitt *Syntax-Symbole*.

Gehört zur Funktionsgruppe: *Internet und XML*

Funktion

Mit dem Statement `REQUEST DOCUMENT` haben Sie die Möglichkeit, auf ein externes System zuzugreifen.

Siehe auch *Statements für Internet- und XML-Zugang* im *Leitfaden zur Programmierung*.

Informationen bezüglich Unicode-Support finden Sie im Abschnitt *Statements* in der *Unicode and Code Page Support*-Dokumentation.

Einschränkungen für Protokollarten

Das HTTPS-Protokoll wird nur unter z/OS unterstützt.

Einschränkungen für Cookies

Unter dem HTTP-Protokoll setzt ein Server Cookies ein, um Status-Informationen zur Client-Workstation zu verwalten.

`REQUEST DOCUMENT` wird mit Optionseinstellungen für das Internet implementiert. Dies bedeutet, dass abhängig von den Sicherheitseinstellungen Cookies verwendet werden.

Wenn in den Einstellungen der Internet-Optionen *Disabled* (Sperren) gesetzt ist, werden keine Cookies versandt, auch wenn ein Cookie-Header (*operand 4*/5) versandt wird. Benutzen Sie für Server-Umgebungen nicht die Internet-Optionseinstellung *Prompt* (Eingabeaufforderung). Mit dieser Einstellung bleibt der Server hängen, weil kein Client auf die Eingabeaufforderung reagieren kann.

In Großrechner-Umgebungen werden Cookies nicht unterstützt und ignoriert.

Syntax-Beschreibung

Operanden-Definitionstabelle:

Operand	Mögliche Struktur					Mögliche Formate												Referenzierung erlaubt	Dynam. Definition
operand1	C	S				A												nein	ja
operand2	C	S				A												nein	ja
operand3	C	S				A												nein	ja
operand4	C	S				A												nein	ja
operand5	C	S				A		N	P	I	F		D	T	L			nein	ja
operand6	C	S				A	U	N	P	I	F	B	D	T	L			nein	ja
operand7	C	S				A												nein	ja
operand8	C	S				A												nein	ja
operand9	C	S				A		N	P	I	F		D	T	L			nein	ja
operand10		S				A												nein	ja
operand11	C	S				A												nein	ja
operand12		S				A		N	P	I	F	B	D	T	L			nein	ja
operand13		S				A	U					B						nein	ja
operand14	C	S				A												nein	ja
operand15	C	S				A												nein	ja
operand16		S								I4								nein	ja
operand17		S								I4								nein	nein

Syntax-Element-Beschreibung:

DOCUMENT FROM *operand1*	**Adresse des Dokuments:** *operand1* ist der URI, der zum Aufruf eines Dokuments benutzt wird. **Anmerkung:** Die folgenden Informationen gelten nur, wenn *operand1* mit `http://` oder `https://` beginnt.
WITH	**WITH-Klausel:** Sie können diese Klausel benutzen, um optional Benutzer/Passwort, Header und Einzelheiten zu den Daten für die Anforderung anzugeben.
USER *operand2*	**Benutzer-Name:** *operand2* ist der Name des Benutzers, der für die Anforderung benutzt wird.
PASSWORD *operand3*	**Benutzer-Passwort:** *operand3* ist das Passwort des Benutzers, das für die Anforderung benutzt wird.

HEADER {[[NAME] ***operand4*** **[VALUE]** ***operand5*****]}...**	**HEADER-Klausel:** *operand4* und *operand5* können nur gemeinsam verwendet werden. • *operand4* ist der Name einer mit dieser Anforderung versandten `HEADER`-Variable. • *operand5* ist der Wert einer mit dieser Anforderung versandten `HEADER`-Variable. **HEADER-Name für *operand4*:** `HEADER`-Namen dürfen nicht `CR`/`LF` (Carriage Return/Line Feed) oder Doppelpunkt (:) enthalten. Dies wird nicht vom Statement `REQUEST DOCUMENT` überprüft. Gültige `HEADER`-Namen entnehmen Sie den HTTP-Spezifikationen. Aus Gründen der Kompatibilität mit der Web-Schnittstelle können Header-Namen mit Unterstrich (_) anstatt Bindestrich (-) geschrieben werden. (Intern wird der Unterstrich durch einen Bindestrich ersetzt). **HEADER-Wert für *operand5*:** `HEADER`-Werte dürfen nicht `CR`/`LF` enthalten. Dies wird vom Statement `REQUEST DOCUMENT` nicht überprüft. Gültige Header-Werte und -Formate entnehmen Sie den HTTP-Spezifikationen. **Allgemeine Informationen zu Headers:** Für eine HTTP-Anforderung sind einige Headers erforderlich, zum Beispiel: *Request-Method* oder *Content-Type*. Diese Headers werden automatisch erzeugt, und zwar abhängig von den mit dem Statement `REQUEST DOCUMENT` angegebenen Parametern. Siehe auch *Automatisch erzeugte Headers*.
DATA	**DATA-Klausel:** Sie können entweder einen spezifischen `DATA`-Variablennamen und -wert angeben (siehe *operand8* und *operand9* weiter unten) oder das vollständige Dokument (siehe *DATA ALL-Klausel* weiter unten).
ALL ***operand6***	*operand6* ist ein vollständiges, zu versendendes Dokument. Dieser Wert ist normalerweise erforderlich für die automatische HTTPAnforderungsmethode `PUT` (siehe *Automatisch erzeugte Headers*). Siehe *Kodierung von eingehenden/ausgehenden Daten*, *DATA ALL-Klausel*.

[ENCODED [[IN] CODEPAGE *operand7*]	*operand6* wird von der voreingestellten (Default-)Codepage (Wert der Systemvariablen `*CODEPAGE`) in die in *operand7* angegebenen Codepage umkodiert. Siehe *Kodierung von eingehenden/ausgehenden Daten*, *DATA ALL-Klausel*.
{[NAME] *operand8* [VALUE] *operand9*}...	**DATA-Variablenname und -wert:** *operand8* und *operand9* können nur gemeinsam benutzt werden: ● *operand8* ist der Name einer mit dieser Anfrage zu sendenden `DATA`-Variablen. Dieser Wert ist erforderlich für die HTTP-Anforderungsmethode `POST` (URL-Kodierung erforderlich, insbesondere Und-Zeichen (&), Gleichheitszeichen (=), Prozentzeichen (%). **Einschränkung:** Wenn *operand8*/*operand9* angegeben wird und die Kommunikation standardmäßig `http://` oder `https://` ist, wird die Anfragemethode `POST` (siehe *Automatisch erzeugte Headers*) mit dem Inhaltstyp (*Content Type*) `application/x-www-form-urlencoded` benutzt. Im Verlauf der Anfrage werden *operand8*/*operand9* durch Gleichheitszeichen (=) und Und-Zeichen (&) voneinander getrennt. Deshalb dürfen die Operanden keine Und-Zeichen (&), Gleichheitszeichen (=) oder (wegen der URL-Kodierung) Prozentzeichen (%) enthalten. Diese Zeichen gelten als ”unsicher” und müssen wie folgt kodiert werden: **Zeichen** **URL-Kodierungssyntax** % %25 & %26 = %3D Siehe auch *Allgemeine Anmerkung zur URL-Kodierung*.
RETURN	**RETURN-Klausel:** Diese Klausel kann benutzt werden, um die `HEADER`- und/ oder `PAGE`-Rückgabeinformationen anzugeben.

HEADER [ALL *operand10*]	**RETURN HEADER ALL-Klausel:** Wenn diese Klausel angegeben wird, enthält *operand10* alle mit der HTTP-Rückmeldung angegebenen Header-Werte. Die erste Zeile enthält die Status-Informationen, und alle folgenden Zeilen enthalten die Headers als Paare mit Namen und Werten. Die Namen enden immer auf einen Doppelpunkt (:), und die Werte enden mit einem Zeilenvorschub (`LF`). (Intern werden alle `CR`/`LF` auf Zeilenvorschub, d.h. `LF`,umgesetzt.)
HEADER [[NAME] *operand11*] [VALUE] *operand12*]...	**RETURN HEADER NAME/VALUE-Klausel:** Wenn diese Klausel angegeben wird, werden nur spezifische `HEADER`-Informationen zurückgegeben. *operand11* und *operand12* können nur in Verbindung miteinander benutzt werden: ● *operand11* ist der Name eines in dieser Anfrage erhaltenen Headers. Die `HEADER`-Angabe ist für HTTP erforderlich. ● *operand12* ist der Wert eines in dieser Anfrage erhaltenen Headers. Die `HEADER`-Angabe ist für HTTP erforderlich. **Rückgabe des Header-Namens für *operand11*:** Aus Gründen der Kompatibilität mit dem Natural Web Interface können Haeder-Namen mit Unterstrich (_) anstatt Bindestrich (-) geschrieben werden. Intern wird der Unterstrich durch einen Bindestrich ersetzt. Wenn *operand11* eine Leerzeichenkette ist, werden die Status-Informationen zurückgegeben: `HTTP/1.0 200 OK`
RETURN PAGE	**RETURN PAGE-Klausel:** Sie können die `PAGE`-Klausel benutzen, wenn Sie möchten, dass die eingehenden Daten in einer spezifischen Codepage kodiert werden. Siehe *Kodierung von eingehenden/ausgehenden Daten*, *RETURN PAGE-Klausel* weiter unten.
PAGE *operand13*	*operand13* ist das für diese Anfrage zurückgegebene Dokument. Siehe *Kodierung von eingehenden/ausgehenden Daten*, *RETURN PAGE-Klausel* weiter unten.

[ENCODED [[FOR TYPE[S] *operand14*...] [IN] CODEPAGE *operand15*]]	*operand14* ist die Liste der Mime-Typen, für die eine Kodierung des zurückgegebenen Dokuments in *operand13* ausgeführt wird. Siehe *Kodierung von eingehenden/ausgehenden Daten*, *RETURN PAGE-Klausel* weiter unten.
	operand15 ist die Codepage, die erforderlichenfalls für die Kodierung von *operand13* benutzt wird. Wenn der Wert von *operand15* leer ist, dann wird *operand13* von der mit `RDCP` definierten Codepage in die voreingestellte (Default-)Codepage (A/B) oder (U) kodiert. Der Schlüsselwort-Subparameter `RDCP` des Profilparameters `XML` wird benutzt, um den Namen der standardmäßigen HTML/XML-Codepage anzugeben. Siehe *Kodierung von eingehenden/ausgehenden Daten*, *RETURN PAGE-Klausel* weiter unten.
RESPONSE *operand16*	**RESPONSE-Klausel:** Geben Sie die `RESPONSE`-Klausel an, wenn Sie möchten, dass die Response-Codenummer der Anforderung angezeigt wird. *operand16* ist die Response-Codenummer der Anforderung, zum Beispiel: `200` (Anforderung erledigt). Siehe auch *Übersicht über Response-Nummern für HTTP-Anfragen*.
GIVING *operand17*	**GIVING-Klausel:** *operand17* enthält den Natural-Fehler, wenn die Anforderung nicht ausgeführt werden konnte.

Automatisch erzeugte Header (*operand4/5*)

Request-Method

Die folgenden Werte werden für *operand5* unterstützt: `HEAD`, `POST`, `GET` und `PUT`.

Die folgende Tabelle zeigt die automatische Berechnung der *request-method* in Abhängigkeit von den vorgegebenen Operanden:

Operand	Request-Method			
	HEAD	**POST**	**GET**	**PUT**
WITH HEADER (***operand4*/*operand5***)	optional	optional	optional	optional
WITH DATA (***operand7*/*operand8***)	nicht angegeben	angegeben	nicht angegeben	nur bei Option `ALL` (*`operand6`*)
RETURN HEADER (***operand10*** **bis** ***operand12***)	angegeben	angegeben	optional	optional
RETURN PAGE (***operand13***)	nicht angegeben	angegeben	angegeben	optional

Content-Type

Wenn die *request-method* `POST` ist, muss ein Header des Typs *content-type* mit der HTTP-Anfrage angegeben werden. Wenn kein *content-type* explizit gesetzt wird, ist der automatisch generierte Wert von *operand5* wie folgt:

```
application/x-www-form-urlencoded
```

Anmerkung:
Es ist möglich, die automatisch erzeugten Headers zu überschreiben. Natural überprüft sie nicht auf Fehler. Es können unerwartete Fehler auftreten.

Allgemeiner Hinweis zur URL-Kodierung

Wenn Sie `POST`-Daten mit dem Inhaltstyp `application/x–www–form–urlencoded` versenden, müssen bestimmte Zeichen mittels URL-Kodierungen dargestellt werden, was bedeutet, dass das Zeichen durch einen hexadezimalen Zeichencode (`%hexadecimal–character–code`) ersetzt wird. Die vollständigen Einzelheiten, wann und warum die URL-Kodierung erforderlich ist, sind in RFC 1630, RFC 1738 und RFC 1808 erläutert. Sie finden hier einige grundsätzliche Informationen. Alle Nicht-ASCII-Zeichen (d.h. gültige ISO 8859/1-Zeichen, die nicht auch ASCII-Zeichen sind) müssen URL-kodiert sein, z.B. die Datei `köln.html` würde in einer URL als `k%F6ln.html` erscheinen.

Einige Zeichen gelten als "unsicher", wenn Web-Seiten per Email angefordert werden.

Diese Zeichen lauten:

Zeichen	URL-Kodierungssyntax
Tab-Zeichen	%09
Leerzeichen	%20
[	%5B
\	%5C
]	%5D
^	%5E
‘	%60
{	%7B
|	%7C
}	%7D
~	%7E

Wenn Sie URLs schreiben, sollten Sie diese Zeichen URL-kodieren.

Einige Zeichen haben spezielle Bedeutungen in URLs, wie z.B. der Doppelpunkt (:), der das URL-Schema vom Rest des URLs abtrennt, der doppelte Schrägstrich (//), der angibt, dass der URL der Common Internet Scheme-Syntax entspricht, und das Prozentzeichen (%). Wenn diese Zeichen als Teile von Dateinamen erscheinen, müssen sie generell URL-kodiert werden, um sie von ihrer Sonderbedeutung in URLs zu unterscheiden (dies ist eine Vereinfachung, die vollständigen Informationen finden Sie in den RFCs).

Diese Zeichen sind:

Zeichen	URL–Kodierungssyntax
"	%22
#	%23
%	%25
&	%26
+	%2B
,	%2C
/	%2F
:	%3A
<	%3C
=	%3D
>	%3E
?	%3F
@	%40

Übersicht über Response-Nummern für HTTP-/HTTPs-Anforderungen

Status	Wert	Rückmeldung
STATUS CONTINUE	100	OK — Fortfahren mit der Anforderung.
STATUS SWITCH_PROTOCOLS	101	Server hat die Protokolle im Upgrade-Header geändert.
STATUS OK	200	Anforderung erledigt.
STATUS CREATED	201	Objekt erstellt, Grund = neuer URL.
STATUS ACCEPTED	202	Asynchrone Beendigung (TBS).
STATUS PARTIAL	203	Teilweise Beendigung.
STATUS NO_CONTENT	204	Keine Infos zurückzugeben.
STATUS RESET_CONTENT	205	Anforderung abgeschlossen, aber Inhalt zurückgesetzt.
STATUS PARTIAL_CONTENT	206	Teilweises `GET` vollendet.
STATUS AMBIGUOUS	300	Server konnte keine Entscheidung über Rückmeldung fällen.
STATUS MOVED	301	Objekt permanent übertragen.
STATUS REDIRECT	302	Objekt zeitweise übertragen.
STATUS REDIRECT_METHOD	303	Umleiten ohne neue Zugriffsmethode.
STATUS NOT_MODIFIED	304	Wenn geändert — seit wann nicht geändert.
STATUS USE_PROXY	305	Umleiten zu Proxy, Adress-Header gibt zu benutzende Proxy an.

Status	Wert	Rückmeldung
STATUS REDIRECT_KEEP_VERB	307	HTTP/1.1: dasselbe Verb beibehalten.
STATUS BAD_REQUEST	400	Ungültige Syntax.
STATUS DENIED	401	Zugriff verweigert.
STATUS PAYMENT_REQ	402	Zahlung erforderlich.
STATUS FORBIDDEN	403	Anforderung nicht erlaubt.
STATUS NOT_FOUND	404	Objekt nicht gefunden.
STATUS BAD_METHOD	405	Methode ist nicht zulässig.
STATUS NONE_ACCEPTABLE	406	Keine Rückmeldung für gefundenen Client annehmbar.
STATUS PROXY_AUTH_REQ	407	Proxy-Echtheitsprüfung erforderlich.
STATUS REQUEST_TIMEOUT	408	Server-Zeitüberschreitung – warten auf Anforderung.
STATUS CONFLICT	409	Benutzer sollte mit mehr Informationen neu starten.
STATUS GONE	410	Die Ressource steht nicht mehr zur Verfügung.
STATUS LENGTH_REQUIRED	411	Der Server verweigerte die Annahme der Anforderung ohne Länge.
STATUS PRECOND_FAILED	412	In Anfrage angegebene Vorbedingung unzulässig
STATUS REQUEST_TOO_LARGE	413	Anforderungselement war zu groß.
STATUS URL_TOO_LONG	414	Anforderungs-URL zu lang.
STATUS UNSUPPORTED_MEDIA	415	Nicht unterstützter Medien-Typ.
STATUS SERVER_ERROR	500	Interner Server-Fehler.
STATUS NOT_SUPPORTED	501	"Required" nicht unterstützt.
STATUS BAD_GATEWAY	502	Fehler-Rückmeldung vom Gateway.
STATUS SERVICE_UNAVAIL	503	Zeitweise überlastet.
STATUS GATEWAY_TIMEOUT	504	Zeitüberschreitung – warten auf Gateway.
STATUS VERSION_NOT_SUP	505	HTTP-Version nicht unterstützt.

Response 301 - 303 (Umleitung)

Umleitung bedeutet, dass der angeforderte URL umgezogen ist. Als Response (Rückmeldung) wird der Rückgabe-Header mit dem Namen `LOCATION` (Adresse) angezeigt. Dieser Header enthält den URL, wohin die angeforderte Seite umgezogen ist. Eine neue `REQUEST DOCUMENT`-Anfrage kann benutzt werden, um die umgezogene Seite zu suchen.

HTTP-Browser leiten automatisch zur neuen URL um, aber das Statement REQUEST DOCUMENT nimmt die Umleitung nicht automatisch vor.

Response 401 (Verweigert)

Die Rückmeldung Access Denied (Zugriff verweigert) bedeutet, dass die angeforderte Seite nur aufgerufen werden kann, wenn mit der Anfrage eine gültige Benutzer-ID und ein gültiges Passwort angegeben werden. Als Rückmeldung wird der Rückgabe-Header mit dem Namen WWW–AUTHENTICATE mit dem für diese Anfrage erforderlichen Bereich ausgegeben.

HTTP-Browser zeigen normalerweise einen Dialog mit Benutzer-ID und Passwort an, aber beim Statement REQUEST DOCUMENT wird kein Dialog angezeigt.

Kodierung von eingehenden/ausgehenden Daten

Bei der Datenübertragung mit dem Statement REQUEST DOCUMENT kommt es normalerweise nicht zu Konvertierungen von Codepages. Wenn Sie möchten, dass die ausgehenden und/oder eingehenden Daten in einer bestimmten Codepage kodiert werden, können sie die Klausel DATA ALL und/oder die Klausel RETURN PAGE verwenden, um dies anzugeben.

DATA ALL-Klausel

Zur Kodierung der ausgehenden Daten wird die Klausel DATA ALL benutzt:

ALL *operand6* [**ENCODED** [[**IN**] **CODEPAGE** *operand7*]]

Syntax-Beschreibung:

ALL *operand6*	*operand6* ist ein vollständiges Dokument, das versandt werden soll. Dieser Wert wird normalerweise für die automatische HTTP-Anforderungsmethode PUT benötigt (siehe *Automatisch erzeugte Headers*).
[ENCODED [[IN] CODEPAGE *operand7*]]	*operand6* wird von der voreingestellten (Default-)Codepage (Wert der Systemvariablen *CODEPAGE) in die in *operand7* angegebenen Codepage umgesetzt.

RETURN PAGE-Klausel

Zur Kodierung von eingehenden Daten wird die Klausel RETURN PAGE verwendet:

[**PAGE** *operand13* [**ENCODED** [[**FOR TYPE**[**S**] *operand14*...] [**IN**] **CODEPAGE** *operand15*]]]

Als Rückmeldung für eine HTTP-/HTTPS-Anfrage können eingehende Daten binäre Daten (zum Beispiel image/gif) oder Zeichen-Daten (zum Beispiel text/html) enthalten. Zusammen mit der Rückmeldung erhält das Statement REQUEST DOCUMENT einen Parameter, der die Art des Inhalts des angeforderten Dokuments angibt (Mime-Typ). Dieser Parameter kann Informationen über die Codepage enthalten, in der das Dokument kodiert ist.

Diese Klausel bietet eine automatische Umsetzung in die voreingestellte (Default-)Codepage (Wert der Systemvariablen `*CODEPAGE`) der Natural-Session.

Syntax-Beschreibung:

RETURN PAGE *operand13*	Es erfolgt keine Kodierung der zurückgegebenen Seite; das bedeutet, die Seite bleibt so kodiert, wie sie vom HTTP-Server geliefert wird.
RETURN PAGE *operand13* ENCODED	Wenn der zurückgegebene Mime-Typ eine Kodierung enthält, dann wird *operand13* von dieser Codepage in die voreingestellte (Default-)Codepage (A/B) oder (U) umkodiert. Siehe Anmerkung unten.
RETURN PAGE *operand13* ENCODED [IN] CODEPAGE *operand15*	Wenn der der zurückgegebene Mime-Typ keine Kodierung enthält, wird *operand13* von der mit *operand15* definierten Codepage in die voreingestellte (Default-)Codepage (Wert der Systemvariablen `*CODEPAGE`) (A/B) oder (U) umkodiert.
RETURN PAGE *operand13* [ENCODED [[FOR TYPE[S] *operand14*...] [IN] CODEPAGE *operand15*]]	Wenn der zurückgegebene Mime-Typ keine Codierung enthält, wird eine zusätzliche Prüfung durchgeführt, ob der zurückgegebene Mime-Typ mit einem der in *operand14* gelieferten Typen übereinstimmt. Wenn eine Übereinstimmung vorliegt, wird *operand13* von der mit *operand15* definierten Codepage in die voreingestellte (Default-)Codepage (A/B) oder (U) umkodiert.

Anmerkung:
"Zurückgegebener Mime-Typ enthält eine Kodierung" bedeutet, dass der HTTP-Server einen Content-Type-Header mit einer `charset=`-Klausel zurückliefert, zum Beispiel: `charset=ISO-8859-1`.

Beispiele für die Verwendung der RETURN PAGE ENCODED-Klausel

1. Server liefert einen Header zurück: `'Content-type: text/html;charset=UTF-8'`

   Programmcode-Beispiel 1:

   ```
   ...
   RETURN PAGE operand13
   ```

   Resultierende Verarbeitung:

   operand13 bleibt UTF-8-kodiert.

   Programmcode-Beispiel 2:

   ```
   ...
   RETURN PAGE operand13 ENCODED [..]
   ```

   Resultierende Verarbeitung:

operand13 wird unabhängig von den eventuell vorhanden Angaben in *operand14* und *operand15* von UTF-8 in die voreingestellte (Default-)Codepage umgesetzt. Die beiden Operanden werden nicht ausgewertet, weil im zurückgelieferten Content-Type-Header eine gültige Kodierung festgestellt wurde.

2. Server liefert einen Header zurück: `'Content-type: text/xml'`

   Programmcode-Beispiel 1:

   ```
   ...
   RETURN PAGE operand13 ENCODED
   ```

   Resultierende Verarbeitung:

   operand13 wird nicht umgesetzt, weil der Content-Type-Header keine gültige Kodierung enthält.

   Programmcode-Beispiel 2:

   ```
   ...
   RETURN PAGE operand13 ENCODED FOR TYPES 'text/xml' IN CODEPAGE 'USASCII'
   ```

   Resultierende Verarbeitung:

   operand13 wird von USASCII-Codepage in die voreingestellte (Default-)Codepage umgesetzt. In diesem Fall erfolgt die Umsetzung gemäss der Annahme des Programmierers über die Kodierung der empfangenen Seite.

   Programmcode-Beispiel 3:

   ```
   ...
   RETURN PAGE operand13 ENCODED FOR TYPES 'text/html'
       IN CODEPAGE '  '
   ```

   Resultierende Verarbeitung:

   operand13 wird nicht umgesetzt, weil der in *operand14* angegebene Mime-Typ `'text/html'` nicht mit dem im Content-Type-Header angegebenen Mime-Typ `'text/xml'` übereinstimmt.

   Programmcode-Beispiel 4:

   ```
   ...
   RETURN PAGE operand13 ENCODED IN CODEPAGE ' '
   ```

   Resultierende Verarbeitung:

   operand13 wird von der mit dem Subparameter `RDCP` des Profilparameters `XML` angegebenen Standard-Codepage in die voreingestellte (Default-)Codepage umgesetzt.

   Anmerkung:
   Der Standardwert für den `RDCP`-Subparameters, der gültig ist, wenn nichts anderes explizit angegeben wird, ist `'ISO-8859-1'`. Siehe auch *XML - Activate PARSE XML and REQUEST DOCUMENT Statements* in der *Parameter-Referenz*-Dokumentation.

Beispiele

- Beispiel 1 — Allgemeine Anforderung
- Beispiel 2 — Einfache Get-Anforderung (keine Daten)
- Beispiel 3 — Einfache Head-Anforderung (keine zurückgelieferte Seite)
- Beispiel 4 — Einfache Post-Anforderung (Voreinstellung)
- Beispiel 5 — Einfache Put-Anforderung (mit allen Daten)

Anmerkung:
Es gibt einen Beispiel-Dialog V5–RDOC für dieses Statement in der Beispiel-Library SYSEXV.

Beispiel 1 — Allgemeine Anforderung

```
REQUEST DOCUMENT FROM "http://bolsap1:5555/invoke/sap.demo/handle_RFC_XML_POST"
  WITH
    USER #User PASSWORD #Password
    DATA
    NAME 'XMLData'       VALUE #Queryxml
    NAME 'repServerName' VALUE 'NT2'
  RETURN
    PAGE #Resultxml
RESPONSE #rc
```

Beispiel 2 — Einfache Get-Anforderung (keine Daten)

```
REQUEST DOCUMENT FROM "http://pcnatweb:8080"
  RETURN
    PAGE #Resultxml
RESPONSE #rc
```

Beispiel 3 — Einfache Head-Anforderung (keine zurückgelieferte Seite)

```
REQUEST DOCUMENT FROM "http://pcnatweb"
RESPONSE #rc
```

Beispiel 4 — Einfache Post-Anforderung (Voreinstellung)

```
REQUEST DOCUMENT FROM "http://pcnatweb/cgi-bin/nwwcgi.exe/sysweb/nat-env"
  WITH
    DATA
    NAME 'XMLData'       VALUE #Queryxml
    NAME 'repServerName' VALUE 'NT2'
  RETURN
    PAGE #Resultxml
RESPONSE #rc
```

Beispiel 5 — Einfache Put-Anforderung (mit allen Daten)

```
REQUEST DOCUMENT FROM "http://pcnatweb/test.txt"
  WITH
    DATA ALL     #document
  RETURN
    PAGE #Resultxml
RESPONSE #rc
```