



CSP での文字セットの使 用法

Version 5.1
2006-03-14

CSP での文字セットの使用法

Caché Version 5.1 2006-03-14

Copyright © 2006 InterSystems Corporation.

All rights reserved.

このドキュメントは、Sun Microsystems、RenderX Inc.、アドビ システムズ および ワールドワイド・ウェブ・コンソーシアム (www.w3c.org) のツールと情報を使用して、Adobe Portable Document Format (PDF) で作成およびフォーマットされました。主要ドキュメント開発ツールは、InterSystemsが構築したCaché と Javaを使用した特別目的のXML処理アプリケーションです。



Caché 製品とロゴは InterSystems Corporation の登録商標です。



Ensemble 製品とロゴは InterSystems Corporation の登録商標です。



InterSystems という名前とロゴは InterSystems Corporation の登録商標です

このドキュメントは、インターシステムズ社(住所: One Memorial Drive, Cambridge, MA 02142)あるいはその子会社が所有する企業秘密および秘密情報を含んでおり、インターシステムズ社の製品を稼動および維持するためにのみ提供される。この発行物のいかなる部分も他の目的のために使用してはならない。また、インターシステムズ社の書面による事前の同意がない限り、本発行物を、いかなる形式、いかなる手段で、その全てまたは一部を、再発行、複製、開示、送付、検索可能なシステムへの保存、あるいは人またはコンピュータ言語への翻訳はしてはならない。

かかるプログラムと関連ドキュメントについて書かれているインターシステムズ社の標準ライセンス契約に記載されている範囲を除き、ここに記載された本ドキュメントとソフトウェアプログラムの複製、使用、廃棄は禁じられている。インターシステムズ社は、ソフトウェアライセンス契約に記載されている事項以外にかかるソフトウェアプログラムに関する説明と保証をするものではない。さらに、かかるソフトウェアに関する、あるいはかかるソフトウェアの使用から起こるいかなる損失、損害に対するインターシステムズ社の責任は、ソフトウェアライセンス契約にある事項に制限される。

前述は、そのコンピュータソフトウェアの使用およびそれによって起こるインターシステムズ社の責任の範囲、制限に関する一般的な概略である。完全な参照情報は、インターシステムズ社の標準ライセンス契約に記載され、そのコピーは要望によって入手することができる。

インターシステムズ社は、本ドキュメントにある誤りに対する責任を放棄する。また、インターシステムズ社は、独自の裁量にて事前通知なしに、本ドキュメントに記載された製品および実行に対する代替と変更を行う権利を有する。

Caché および InterSystems Caché、Caché SQL、Caché ObjectScript および Caché Object は、インターシステムズ社の商標です。

ここで使われている他の全てのブランドまたは製品名は、各社および各組織の商標または登録商標です。

インターシステムズ社の製品に関するサポートやご質問は、以下にお問い合わせください:

InterSystems ワールドワイド カスタマサポート

Tel: +1 617 621-0700

Fax: +1 617 374-9391

Email: support@InterSystems.com

目次

CSP での文字セットの使用方法.....	1
-----------------------	---

CSP での文字セットの使用法

文字セットについて、および文字セットが CSP によって処理される方法については、以前にも説明しました。そこで今回は、CSP 文字セット選択枝の背景にあるロジックと、オプション、およびそれらがどのように作動するかについて説明します。ユーザの皆さんにとって、有用な情報となります。

Caché で提供されている各 CSP ページには、ページのコンテンツをブラウザに伝える、一連の HTTP ヘッダが含まれます。これらのヘッダの定義は、RFC 1945 (現在 CSP でサポートされている HTTP/1.0 仕様) に含まれています。これらのヘッダはブラウザでは表示されず、ヘッダを表示できるブラウザはほとんどありません。ヘッダが何であるかを知るには、Caché に組み込まれている `%Net.HttpRequest` オブジェクトを使用すると便利です。例えば、一般的な CSP ページからヘッダを検索するには、以下を実行します。

```
Set http = ##class(%Net.HttpRequest).%New()  
Set http.Server = "127.0.0.1"  
Set http.Port = 1972  
Do http.Head("/csp/samples/loop.csp")  
Do http.HttpResponse.OutputToDevice()  
Do http.%Close()
```

これにより、(Unicode Caché システムで) 以下が出力されます。

```
HTTP/1.0 200 OK  
CACHE-CONTROL: no-cache  
CONNECTION: Close  
CONTENT-TYPE: text/html; charset=utf-8  
DATE: Mon, 13 Aug 2001 17:34:18 GMT  
EXPIRES: Thu, 29 Oct 1998 17:04:19 GMT  
PRAGMA: no-cache  
SET-COOKIE: CSPSESSIONID=260066677206262462181; path=;
```

Caché に組み込まれているミニ Web サーバを使用するため、ポートは 1972 に設定されていますが、適切な Web サーバのセットアップをお持ちの場合は、既定でポート番号 80 を使用することもできます。CSP が生成するヘッダを表示するには、以下を使用して Caché から生成する方法もあります。

```
Do ShowPage^%apiCSP("/csp/samples/loop.csp",,,1)
```

'Content-Type' ヘッダは常に送信され、これにはブラウザで予期されている、ドキュメントのタイプを指定します。通常、これは "text/html" になります。HTML ドキュメントであり、対応するレンダリングが必要なことを意味します。JPEG イメージの場合は "image/jpeg" になります。コンテンツ・タイプが "text" の場合、文字セットを指定して、"charset" 修飾子と同じ行で使用することもできます。

```
Content-Type: text/html; charset=utf-8
```

これは、後に続く Web ページが Unicode 標準のマッピングである utf-8 でコード化されていることを意味しています。ASCII コード 0-127 は、値 0-127 として utf-8 にマップされます。ブラウザは、HTTP ヘッダで指定された文字テーブルの各文字を検索し、この文字に対応するシンボルをウィンドウに表示します。

CSP では、適切な charset ヘッダを CSP ページと送信し、ブラウザがそれを適切に表示できるようにすることが重要です。charset に値を指定しなければ (この方法は後述します)、CSP は Unicode Caché システムで、既定値を使用します。これは、8 ビットの Caché システム上の “utf-8” です。これは、システムの既定のロケールで、CNLS.EXE ユーティリティ (実行ファイルとともに、CacheSys/Bin ディレクトリにインストールされています) を使用して設定でき、このユーティリティを実行する時に、“ロケール” タブの最上部に表示されます。以下を呼び出すことでも見つけることができます。

```
Write ^%SYS("LOCALE", "CURRENT")
```

上記をコマンド行で入力します。ドロップダウン・ダイアログの隣の名前を、HTTP ヘッダで使用されている外部名に変換するには、以下の関数を使用します。

```
Write $$MapExtCharset^%NLS(^%nls("Loc", locale, 0))
```

locale は、CNLS.EXE ユーティリティの小文字の値です。

ページの charset を、既定ではなく明示的に設定したい場合、以下のようないくつかの方法があります。

OnPreHTTP メソッドで、%response オブジェクトの **CharSet** プロパティを設定します。

```
Set %response.CharSet="iso-8859-1"
```

これは、コンテンツ・タイプが “text” タイプの場合 (他のコンテンツ・タイプではできません)、この値とともに “charset” ヘッダを出力します。このメソッドは通常は、入力を変換されないのでお勧めしません。他のメカニズムのうちのどれかを使用することをお勧めします。CSP ページでは <csp:content charset="iso-8859-1"> タグを使用します。これは CSP システム・ルールに適合し、この CSP ページからコンパイルされたクラスの CHARSET パラメータを設定します。ページが読み込まれるとき、%response.CharSet 値はこのページの CHARSET パラメータに初期化されます。したがって、コンテンツ・タイプが 'text' タイプの場合、これは charset に対して書き出される値です。CSP ページの <head></head> セクションにある、<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1"> タグを使用します。これはシステム規約から見つけられ、コンパイルされたクラスの CHARSET パラメータに変換されます。したがって、最終結果はオプション 1 とまったく同じものです。

%CSP.Page から派生するクラスを作成し、このクラスでパラメータ CHARSET に使用したい charset を指定します。[システム管理ポータル] の [CSPアプリケーション編集] ([[ホーム]→[セキュリティ管理]→[CSPアプリケーション]→[CSPアプリケーション編集]]) で、作成した新しいクラスがこのアプリケーションのスーパークラスになるように、[デフォルトスーパークラス] フィールドを設定します。次に、CSP アプリケーションのすべてのページをリコンパイルし、このスーパークラスを継承するようにします。結果として、既定の CHARSET パラメータを自動的に取得するようになります。これは、同時に大量のページの charset を設定するときに非常に便利です。

HTTP ヘッダで “charset” エントリを入力する方法以外に、これを設定することで、ブラウザに書き出されるすべての文字を Caché が変換する方法、およびブラウザから受け取った入力を HTTP ポスト、または URL のパラメータとして変換する方法に影響します。CSP は常に utf-8 を使用して HTTP ヘッダを出力します。ブラウザが ‘charset’ ヘッダを読む前には、charset が何かを知ることはできないからです。また、RFC は文字が HTTP ヘッダで ASCII 以外の文字を許可するならば、ASCII 文字はこのエンコードを使用してすべて保持されるので、論理的な唯一の選択肢は utf-8 になります。ヘッダが一度出力されると、CSP は以下の呼び出しを使用して、ページの残りの出力をヘッダで指定された charset に切り替えます。

```
Write $$SetIO^%NLS(tablename)
```

この関数は、使用されていた古いテーブルを返し、新規のテーブルを使用するために現在のデバイスを変更します。このテーブル名は、テーブルの Caché 内部名です。例えば、変換を停止する場合は “RAW” で、utf-8 を使用する場合は “UTF8” です。現在のテーブルの名前を検索する場合は、以下を使用します。

```
Write $$GetIO^%NLS()
```

これは、Caché の Unicode バージョンで、既定の charset を使用する場合、すべての文字出力が utf-8 で表されるように強制します。8ビットの Caché で、既定値 (Caché で指定されているロケール) を使用している場合、文字はまったく変換されず、“RAW” テーブルが使用されます。これは、Caché データベースの文字がすでに既定のロケールにあるからです。(なぜなら、既定のロケールは、Caché データベースのロケール自体を表すためです)。したがって既定のテーブルから既定のテーブルに変換されるだけなので、変換は必要ありません。

使用する charset を指定する場合、Caché データベース内部に保存されている文字が、指定された charset に変換されるように、Caché は NLS テーブルを取得します。charset 名から内部 Caché テーブルの名前を検索するには、以下の関数を使用します。

```
Write $$MapCharset^%NLS(charset)
```

CSP からページを表示するときに “Character Set TEST not installed” エラー・メッセージが発生した場合、これは Caché で利用できる charset を要求しているという意味です。例えば、以下を呼び出す場合、

```
Write $$MapCharset^%NLS("TEST")
```

これは “” が返されます。“TEST” という charset が存在しないからです。このエラーが発生した場合、対処法は 2 つあります。1 つは CNLS.EXE ユーティリティを使用して Caché にテーブルをロードする (またはこのテーブルが提供されていない場合は、この名前を持つテーブルを作成する) 方法、もう 1 つはサポートされている他の charset を使用する CSP ページに変更する方法です。

指定されたテーブルを使用して変換できない文字は、“?” として出力されるか、エラー文字は CNLS.EXE ユーティリティで指定されます。したがって、Web ページや Caché から返されたデータに “?” という文字が表示されている場合、適切に変換できない文字が含まれているということになります。

URL、または HTTP ポストで、Web ブラウザからデータが返されると、CSP は文字を外部 charset から Caché の既定のロケールに変換します。Web ブラウザから送信された HTTP ヘッダは、ブラウザが使用している charset を指定しないので (HTTP/1.0 および 1.1 仕様の重大な欠点)、ユーザは使用される charset を推定する必要があります。ブラウザから送信された charset が、情報を送信しているページで指定された charset と同じであると想定します。ユーザが 'iso-8859-1' の charset を使用する **form.csp** ページを持ち、“iso-8859-1” の charset を使用する **result.csp** ページに情報を送信するフォームがある場合、**result.csp** の charset からブラウザによって送信される情報を変換します。つまり、“iso-8859-1” は Caché の既定のロケールに変換されます。一般的なブラウザのテストによると、ユーザが特定の charset を持つページを提供する場合、ブラウザによってこのページから送信されるすべての情報は、同じ charset を使用してコード化されます。これはユーザが、情報を Caché に送信しようとするページと、この情報を受け取るページの両方が同じ charset を使用していることを明確にする必要があることを意味しますが、アプリケーション全体は同じ charset を使用する傾向があるので、難しい要求ではありません。CSP は、\$ZConvert 関数を使用して、文字列を外部 charset から内部 charset に変換します。

```
Write $ZConvert(string,"I",table)
```

string は変換する文字列で、table は Caché 内部テーブル名です。

このドキュメントで説明する最後のトピックは、csp ファイルで filesystem から CSP がどのように読み取るかと、これが charset の問題をどのように処理するかです。コンパイラは csp ファイルを %FileCharacterStream オブジェクトを使用して読み取ります。最初に、そのファイルが Unicode ファイルであることをチェックし、Unicode ファイルである場合は、ビッグ・エンディアンか、リトル・エンディアンかをチェックします。このファイルが \$char(255,254)、もしくは \$char(254,255) で、Unicode である場合は、ファイルの最初の 2 バイトを読むことでチェックできます。ファイルが Unicode ではない場合は、(シーケンシャル・デバイスに対する変換タイプの “Locale” タブで) Caché からのファイルの読み込み、書き込みのための CNLS.EXE で既定の変換設定を使用します。

例えば、Unicode の Caché システムを持ち、これを charset が Shift-JIS である日本で使用すると想定します。CSP コンパイラが csp ファイルをロードするとき、Shift-JIS から Unicode への変換のために設定される既定のシーケンシャル・デバイスの変換を使用します。したがって、Caché でコンパイルされる CSP クラスは、Unicode で内部的に保存されます。上記のメカニズムの 1 つを使用することで、Shift-JIS の charset を持つこのページを出力することを CSP に伝えるので、このページに対する要求が発生したとき、CSP は TCP/IP デバイスを設定して、Unicode 内部表現を Shift-JIS に外部的に変換します。また、これは HTTP ヘッダ 'charset=Shift-JIS' の追加も行うので、ブラウザはページを適切に表示します。このページに Shift-JIS 文字が入力され、送信されるフォームを含む場合、ブラウザによって送られたデータは Shift-JIS の charset を使用します。送信されるページも Shift-JIS の charset を持つので、CSP は Shift-JIS からこの文字の内部 Unicode 表示に戻されます。csp ファイルで charset の設定を行う <meta http-equiv> メソッドには、論理的にわずかな矛盾があることに注意してください。Web サーバで <meta http-equiv> タグを含む html ファイルを作成する場合、このページがブラウザから出力されるとき、ブラウザはこの <meta http-equiv> タグで指定された charset からデータを変換して、'Content-Type' HTTP ヘッダが 'charset' を含むかのようにページに表示します。したがって、これはブラウザに html ファイルの charset を伝えます。しかし、ユーザが <meta http-equiv> タグを csp ファイルに入力し、これがシーケンシャル・ファイルに対する既定のデバイス変換を使用して Caché にロードされる場合、これがブラウザに出力されると

き、Caché の既定のロケールから `<meta http-equiv>` タグで指定された charset に変換されます。したがって、`<meta http-equiv>` を含むページを保存し、'utf-8' の charset を指定する HTML 編集ツールは、全ページを utf-8 として保存します。Caché がこのページをロードするとき、utf-8 から Caché の既定の内部ロケールに変換するはずですが、デバイス変換がページの charset と適合しない場合には、問題が発生するものの代わりに、既定のデバイス変換を使用するだけです。`<meta http-equiv>` で指定された charset は、既定のデバイス変換と同じである場合がよくあります (例：日本でのシステムは両方とも Shift-JIS)。このため正しく機能します。

ブラウザへの出力、およびブラウザからの入力でページの変換を避けたい場合、NOCHARSETCONVERT というコンパイルされたクラスのパラメータも提供しています。これが 1 に設定されている場合、CSP は HTTP “Content-Type” ヘッダの charset を出力しますが、Caché の既定のロケールからこの charset に変換するためのデバイス変換の設定は行いません。また、このブラウザによって送信された値を、この charset から Caché の既定のロケールに変換することはありません。これは、すべての charset 変換問題が、CSP エンジンそのものではなく、使用しているアプリケーションによって処理される必要があるということを意味しています。また、`<csp:content nocharsetconvert="1">` タグを持つ csp ページで、NOCHARSETCONVERT を設定することもできます。

NOCHARSETCONVERT は、ロケールが定義されていないタイ国などに対して開発されたもので、英語の既定のロケールを使用しますが、実際にデータベースでは 8 ビットのタイ語文字を保存します。

