



Caché Server Pages に関するよくある質問

Version 5.1

2006-03-14

Caché Server Pages に関するよくある質問
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

Caché Server Pages に関するよくある質問.....	1
トラブルシューティング	1
セッションとライセンス	5
コマンドと構文	8
構成	11
その他	13

Caché Server Pages に関するよくある質問

トラブルシューティング

なぜ、以下のコードはコンパイルしないのですか？

```
<script language="Cache" runat="server">
  write "<script language=javascript>", !
  write "int x = 10; alert(x);", !
  write "</script>"
</script>
```

スクリプト・タグで `runat="server"` を使用して CSP ページをコンパイルするとき、コンパイラは Caché ObjectScript を実行し、それを適切な HTML に変換してページに表示します。しかし `<script language="cache" runat="server">` タグを読み込んだ後は、次の `</script>` タグを検索します。このタグは Caché ObjectScript コードの最後を意味します。この方法でコードを記述する場合、簡単な解決法があります。`</script>` タグを、以下のように 2 つの Write 文に分割します。

```
<script language="Cache" runat="server">
  write "<script language=javascript>", !
  write "int x = 10; alert(x);", !
  write "</script", ">", !
</script>
```

`&js<alert("Error!");>` を使用して警告ボックスをユーザに送信するとき、**`alert("Error!")`** の行がページに表示されます。なぜでしょうか？

この行を `runat="server"` コード・セクション、または `runat="server"` ブロックから呼び出されるメソッド内に記述したことが原因である可能性があります。ページをロードするときに JavaScript を実行するには、前の回答で示されたように、`<script language="javascript">` タグを追加します。

`&js<alert("Error!")>` コードは、ロードされたページから JavaScript イベントを経由して、サーバ側メソッド内部で動作します。

警告ボックスに Caché ObjectScript 変数を含むには、どのようにすればよいでしょうか？

`#()` # 構文を使用します。ご使用の Caché ObjectScript メソッド内部から、以下を実行します。

```
s error = "Bad password"
&js<alert(("#(..QuoteJS(error))#));>
```

%CSP.Page クラスから提供されている QuoteJS は、適切に引用符が付いた JavaScript 文字列を返します。また、返された結果に含まれるすべての引用符を適切にエスケープします。

CSP ページをロードするとき、以下のエラーが表示されます。ERROR #5924: An error occurred and the specified error page could not be displayed – please inform the Web master. これはどういう意味で、どうすればいいのですか？

このエラーは、異なる多くの問題から発生しています。エラー・ログから、実際のエラーに関する具体的な情報を入手してください。Caché ターミナル で、以下のコマンドを入力します。

```
d ^%ER
```

結果のエラー・ログを表示するには、[システム管理ポータル] の [アプリケーションエラーログの表示] ページに進み ([ホーム]→[システムログ]→[アプリケーションエラーログの表示])、適切なネームスペースに対するエラーにチェックを付けます。エラーは、日付ごとにフォルダ内に表示されています。

カスタムのエラー・ページを設定している場合、このエラー・メッセージは、ご使用のカスタム・エラー・ページには呼び出しているページのエラーを処理する機能がないことを意味します。また、カスタム・エラー・ページ自体が、エラーを生成している可能性もあります。このエラーを追跡するには、カスタム・エラー・ページを一時的に停止して、CSP ページをロードしてみてください。

CSP ページがローカルで正常に表示されても、他のコンピュータから呼び出されると正常に表示できない場合は、ユーザが Caché のシングル・ユーザ・バージョンを使用しているか、Caché のライセンスを持っていないかのどちらかが原因と考えられます。リモート・マシンから CSP ページを呼び出す場合は、Caché の完全なバージョンと有効なライセンスを持ったキーの両方が必要です。インターネットでダウンロードしたバージョンに、Caché キーを追加しても、すべての機能を使用することはできません。完全なバージョンを入手する必要があります。

以下のエラーが発生しました。HTTP Request failed.Unable to process hyperevent. これはどういう意味ですか？

基本的にハイパーイベント・エラーは、ブラウザが CSP ブローカー・アプレットと通信する必要があるのに、通信できない場合に発生します。これは、コード、または構成の問題に関係します。原因がコードにあるかを迅速に判断するために、`http://localhost:8972/csp/samples/zipcode.csp` をロードしてみましょう (Caché が既定のポート番号を使用してインストールされている場合)。Caché がリモートでインストールされている場合、localhost に IP アドレス、またはマシン名を入れます。#server をクリックし、Zip ボックスに 02142 などの郵便番号を入力し、Tab を押します。ハイパーイベント・エラーが発生しない場合は、構成は適切に行われており、問題はコードにあると考えられます。

問題がコードにあると考えられる場合、以下の 2 点が含まれていないか探すことが重要です。ページがロードされているときに、絶対に #server 呼び出しを使用しないでください。つまり、<body> タグの OnLoad イベントで直接的に、または OnLoad で呼び出される JavaScript メソッドから、#server

を呼び出さないでください。また、`runat="server"` コード・ブロックの `&js<>` 行内部からの呼び出しも実行しないでください。ページをロードするときにメソッドを呼び出す必要がある場合は、`runat="server"` ブロック内部で `Caché ObjectScript` 構文を使用します。

```
<script language="cache" runat="server">
    // if method is located in the page or in a class the page inherits from
    d ..mymethod()
    // if class cannot be called using dot syntax
    d ##class(User.MyClass).mymethod()
</script>
```

ルーチンでの問題を指摘するエラー・メッセージを受け取りましたが、INT ルーチンを見つけることができません。どこにあるのでしょうか？

既定では、CSP ページのコンパイル時にソース・コードは保存しません。CSP ページは、ユーザがそのページをブラウザにロードしようとしたとき、自動的にコンパイルされます。Caché でこのソース・コードを保存するには、以下の 2 つのうち 1 つを実行します。

- ・ Caché スタジオ を使用して、CSP ページをコンパイルします。スタジオ の **[ツール]** メニューで **[オプション]** を選択し、**[クラス]** タブをクリックします。**[生成されたソース・コードを保存]** チェック・ボックスに、チェックを付けます。次に、**[ビルド]** メニューで **[コンパイル]** をクリックして、CSP ページをコンパイルします。
- ・ Caché ターミナル で “k” フラグを使用して、CSP ページをコンパイルします。この方法でも、“生成されたソース・コードを保存” しながらコンパイルすることができます。ターミナル で、適切なネームスペースにいることを確認します。適切なネームスペースでない場合は、以下を入力してネームスペースを変更します。Zn “<namespace>” CSP ページをコンパイルするには、以下を入力します。Do \$System.CSP.LoadPage(“/csp/<namespace>/<pagename>/csp”, “ck”)以下はその例です。

```
Do $System.CSP.LoadPage("/csp/samples/james.csp", "ck")
```

CSP ページを一度コンパイルし、そのソース・コードを保存するように Caché に要求すると、以下の方法でそのルーチンを見つけることができます。

Caché スタジオ を開き、適切なネームスペースにいることを確認します。**[ファイル]** メニューで、**[開く]** をクリックします。**[開く]** ダイアログ・ボックスの **[ファイルの種類]** の一覧から、**[中間ルーチン(*.int)]** を選択します。ファイルには、`csp.<csp_page_name>.x.INT` という名前が付けられています。x は、一連のこのルーチンの番号を示しています。容量の大きい CSP ページは、複数の INT ルーチンに分割されます。エラーを含むファイル数は、最初に受け取ったエラー・メッセージに表示されています。

LoadPage コマンドに追加することのできるフラグのリストについては、“[フラグとは？](#)” の質問の回答を参照してください。

CSP ページを実行すると、ブラウザに **#(variable)#** のようなものが表示されます。なぜ、変数のデータと置換されないのでしょうか？

これは、ご使用の CSP ページが Caché CSP コンパイラによって適切に解析されていないことを示しています。 <http://localhost/csp/<namespace>/page.csp> にアクセスして、Web サーバを通してページを実行していることを確認してください。

Dreamweaver の **プレビュー** 機能、または Windows エクスプローラ でダブル・クリックしてそのページを表示しようとしている場合、CSP ページは CSP コンパイラによって解析されません。CSP はこの方法で見えることはできません。IIS や Apache などの Web サーバをご使用でない場合は、いずれか 1 つを導入する必要があります。Web サーバの設定に問題がある場合は、インターシステムズのサポート窓口 にお問い合わせください。

CSP ページをロードしようとする、**Invalid Character** というエラー・メッセージが表示されます。なぜでしょうか？

Web サーバを使用してページをロードしていない場合、ブラウザは CSP 構文を表す方法を知らない、このエラーが発生しても不思議ではありません。ユーザの URL が **C:/Cachesys/csp/user/mypage.csp** などのような場合、Web サーバを使用してページをロードしていないこととなります。

URL は <http://localhost:8972/csp/user/mypage.csp> のようなものになるはずですが。

これらのメッセージは、`#server()` 呼び出しが、適切に Caché ObjectScript メソッド呼び出しに変換されなかったために生じた可能性もあります。ご使用のブラウザで右クリックして、ページのソースを表示し、ソースに `#server` が含まれていれば、ユーザの構文に誤りがあります。

`#server(..methodname())#` のように、適切な構文になるようにしてください。

文字列をメソッドに渡している場合、文字列は二重引用符 (“ ”) に囲まれる必要があります。一度 CSP ページが HTML にコンパイルされると、すべての `#server()` 呼び出しは `csprunservermethod()` への呼び出しに変換されます。

なぜ CSP タグは解析されないのでしょうか？

Web サーバが、CSP ページを解析用の CSP ゲートウェイに適切に渡していないためです。

CSP はエラーを記録しますか？ログを増大するにはどうすればよいでしょうか？また、ログはどこにあるのでしょうか？

- ・ カスタム・エラー・ページ内でのエラーなどの内部エラーがある場合は、**BACK^ETN** エラー・ログにログが記録されます。このログでカスタム・エラー・ページに関係しない内部エラーを受け取った場合、コアの CSP エンジンに問題がある可能性が高いため、インターシステムズのサポート窓口にお問い合わせください。

- ・ ユーザが独自のログ機能を追加するユーザ定義のエラー・ページを呼び出すことで、他のエラーも記録されます。
- ・ `%SYSLOG` グローバルが削除された場合、CSP はこのグローバルへの重要なエラーに関する情報も記録します。Set `%SYSLOG=0` を実行した場合、このグローバルに関する情報は何も記録しません。このグローバルに含まれる情報は、内部デバッグのためだけに使用されます。

セッションとライセンス

CSP セッションを終了するには、どのようにすればいいのでしょうか？

CSP セッションは、Caché ObjectScript メソッドで `%session.EndSession` プロパティを 1 に設定すると終了できます。CSP アプリケーションがタイムアウトした場合、ご使用の CSP クラスが自動的にセッションを終了します。

CSP セッションをクローズしましたが、ライセンスを使用している状態であることを Caché が報告してきます。なぜでしょうか？

ユーザが単にブラウザを閉じた場合、CSP セッションはユーザが指定したタイムアウトまで有効です。したがって、ライセンスはセッションが無効になるまで解放されません。タイムアウトに関する詳細は、次の [“タイムアウトの変更”](#) に対する回答を参照してください。

また、ユーザが時間内に戻ってくることを想定して、同じライセンスを再度取得することができるように、CSP は猶予期間を設定しています。`%session.EndSession` が 1 に設定された後の猶予期間は 5 分です。しかし、各ライセンスは、CSP アプリケーションに対して最低 10 分が割り当てられる必要があります。例えばユーザがログアウトをクリックした時など、`%session.EndSession` は、タイムアウトした後自動的に、またはアプリケーションによって手動で 1 に設定されます。

例：

- ・ ログイン - 12:00、ログアウト - 12:15、猶予期間 5 分、ライセンス解放 - 2:20
- ・ ログイン - 12:00、ログアウト - 12:03、最小期間 10 分、ライセンス解放 - 12:10
- ・ ログイン - 12:00、ブラウザ終了 - 12:10、タイムアウト設定 15 分、セッション終了 - 12:25、猶予期間 5 分、ライセンス解放 - 12:30

アプリケーションのタイムアウトを変更するには、どのようにすればいいのでしょうか？

アプリケーションの既定のタイムアウトは、各ネームスペースで 900 秒 (15 分) と設定されています。

- ・ 特定のネームスペースで、すべての CSP ページに対する **タイムアウト** を変更する方法は以下のとおりです。

1. Cache キューブ から [システム管理ポータル] をクリックします。必要な場合は、ログインします。
2. [システム管理ポータル] のメイン・ページで [システム管理] 列から [セキュリティ管理] を選択すると、[セキュリティ管理] ページが表示されます ([[ホーム]→[セキュリティ管理]])。
3. [セキュリティ管理] ページで [アプリケーション定義] 列から [CSPアプリケーション] をクリックすると、[CSPアプリケーション] ページが表示されます ([[ホーム]→「セキュリティ管理」→[CSPアプリケーション]])。
4. [CSPアプリケーション] ページで、構成対象のアプリケーションに対応する [編集] をクリックすると、そのアプリケーション用の [CSPアプリケーション編集] ページが表示されます ([[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]])。
5. [デフォルトタイムアウト] フィールドに、新規の値を入力し (単位は秒)、[保存] をクリックします。

特定のアプリケーションに対する Timeout を変更するには、ページ内で以下を入力します。x はタイムアウトの秒数です。

```
s %session.AppTimeout = x
```

ユーザ CSP セッションがタイムアウトになるとき、クリーンアップ、またはロギングを実行したいと思っています。どのようにすればいいのでしょうか？

1. OnTimeout クラス・メソッドを持つイベント・クラスを作成します。
2. 以下のうちの 1 つの方法で、アプリケーションに対するイベント・クラスとして、これを指定します。
 - ・ [システム管理ポータル] の [CSPアプリケーション] ページ ([[ホーム]→「セキュリティ管理」→[CSPアプリケーション]]) で、構成対象の CSP アプリケーションに対応する [編集] をクリックすると、そのアプリケーション用の [CSPアプリケーション編集] ページが表示されます ([[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]])。[イベントクラス] フィールドに、“User.MyEventClass” などの、使用するクラス名を入力します。
 - ・ CSP ページで、%session.EventClass プロパティを使用して、以下を実行します。

```
<script language="cache" runat="server">  
s %session.EventClass = "User.MyEventClass"  
</script>
```

3. OnTimeout メソッドで、保存したいすべての情報を記録します。

注釈: ここで、ユーザは情報をブラウザに返送することはできません (警告が発生するか、戻されます)。

ページ間で情報を渡すには、どのようにすればよいのでしょうか？

情報を渡すには、多くの方法があります。

- 追加のパラメータとして、次のページへのリンクで情報を入力します。この追加のパラメータは、`%request` オブジェクトからアクセス可能です。

```
http://myserver/csp/user/mypage.csp?id=3&name=bill
```

情報にアクセスするには、`%request.Get("id")` を使用します。

ページに直接、情報を表示するには、`#{%request.Get("name")}#` を使用します。

- `%response.Context` 多次元プロパティを使用して、すべてのリンクを自動的に追加し、CSP エンジン形成する一連の名前値の組み合わせを定義します。
- `%session` オブジェクト内に、データを配置します。ユーザが 1 つのアプリケーションを同時に 2 つのブラウザで開くと、問題が発生します。

セッションがタイムアウトになったとき、他の Web ページをユーザに送信したいのですが、どのようにすればいいのでしょうか？

一番簡単な方法は、タイムアウト後に発生するように `metatag` で転送の設定を行うことです。

```
<html>
<head>
<META HTTP-EQUIV="REFRESH" CONTENT="910;
  URL=youhavetimedout.csp">
</head>
<body>
<script language="cache" runat="server">
  %session.AppTimeout = 900
</script>
</body>
</html>
```

ページを 60 秒ごとに自動的に更新したいのですが、どのようにすればいいのでしょうか？

CSP ページの最初で、以下の `metatag` を使用します。

```
META HTTP-EQUIV="REFRESH" CONTENT="60; URL=mypage.csp">
```

コマンドと構文

Caché 変数や式を CSP ページで表示するには、どうすればいいのでしょうか？

“#(var)#” や “#(expression)#” を使用して、変数や式を実行時にページに組み込むことができます。以下はその例です。

#(name)# : name は設定されています。

#(\$G(%request.Get("Username")))# : URL から Username を取得します。

#(2+7+3)# : Web ページ上で 12 を表示します。

“#(var)#” と “##(var)##” の違いは何でしょうか？

“#()#” は、括弧内の式をその実行時値と置換します。“##()##” はページがコンパイルされるときに、変数や式をその値と置換します。

この違いを示すため、以下のコード・サンプルを CSP ページに配置してみましょう。

```
Runtime: #($P($H,"",2))#  
Compile Time: ##($P($H,"",2))##
```

ブラウザでページを開き、何度かこのページを更新します。ページが更新されるたびに、Runtime 値が変更することに注意してください。Compile Time 値はページがコンパイルされる時間を保持します。ページがリコンパイルされたときのみ、変更されます。

#server(..method())# と #call(..method())# の違いは何でしょうか？

#server 指示語と #call 指示語は、両方とも JavaScript イベント、またはコードを使用して呼び出される必要があります。

#server 指示語は、CSP Event Broker によってサーバでメソッドを呼び出します。これは、この Java アプレットに依存するので、ブラウザ依存ともいえるでしょう。メソッドから CSP ページを終了することで、情報は CSP ページに返送されます。これは、同期メソッド呼び出しを提供します。#server メソッドが返されるまで、ユーザはページで何もできません。

#call 指示語の文は、CSP ページ内の隠し iFrame を使用して実行されます。これは Java を基にしたものではないので、ブラウザ依存です。#call メカニズムは非同期メソッド呼び出しを提供し、メソッドが値を終了、または値を返すことを許可しません。バックグラウンドで #call 経由で呼び出されたメソッドが動作している間でも、ユーザは作業を継続することができ、CSP ページでフィールドの変更を行うこともできます。

#call と #server を両方使用する例は、<http://localhost:8972/csp/samples/zipcode.csp> を参照してください。詳細は、“Caché Server Pages (CSP) の使用法” の “CSP におけるタグを使用した開発” の章の “サーバ側のメソッド” を参照してください。

“#include” と “CSP:Include” の違いは何でしょうか？

#include 指示語を使用して、JavaScript、html や単純なテキストなどのテキストを CSP ページに含むことができます。

<csp:include> タグには、適切にフォーマットされた CSP ページが含まれます。このタグは ServerSideRedirect を使用してこのページを挿入し、元のページの処理に戻ります。

CSP ページをコンパイルするには、どうすればいいのでしょうか？

既定で、CSP ページが変更された後 (タイムスタンプ比較から決定される)、ブラウザにロードされる時に自動的にコンパイルされるようになっています。また、Caché スタジオや Caché ターミナルを使用して、CSP ページを手動でコンパイルすることもできます。

スタジオを使用して CSP ページをコンパイルする方法は、以下のとおりです。

1. [ツール] メニューで [オプション] を選択し、[クラス] タブをクリックします。
2. [生成されたソース・コードを保存] チェック・ボックスにチェックを付け、[OK] をクリックします。
3. [ビルド] メニューで [コンパイル] を選択して、CSP ページをコンパイルします。

ターミナルを使用して CSP ページをコンパイルするには、コンパイラに “生成されたソース・コードを保存” するように伝える “k” フラグを使用します。

1. ターミナルで、現在適切なネームスペースにいることを確認します。適切なネームスペースでない場合は、以下を入力してネームスペースを変更します。

```
zn "<namespace>"
```

2. **do \$System.CSP.LoadPage("/csp/<namespace>/<pagename>.csp", "ck")** と入力して、CSP ページをコンパイルします。

以下はその例です。

```
SAMPLES> do $System.CSP.LoadPage("/csp/samples/james.csp", "ck")
```

フラグとは何ですか？

フラグの最新で完全なリストを表示するには、ターミナルで以下のコマンドを実行します。

```
Do $System.OBJ.ShowFlags()
```

これは、以下を出力します。

- a - Include application classes. This flag is set by default.
 - b - Include sub classes.
 - c - Compile. Use this flag while loading a CDL file will cause the classes loaded to be compiled as well.
 - d - Display. This flag is set by default.
 - e - Delete extent.
 - f - Force. Force a compilation even when classes are in use.
 - h - Generate help.
 - i - Validate XML export format against schema on Load.
 - k - Keep source. When this flag is set, source code of generated routines will be kept.
 - l - Use lock while compilation classes. This flag is set by default.
 - p - Include percent classes.
 - q - SQL-only compilation.
 - r - Recursive. It means include all the classes that are dependency predecessors.
 - s - Include system classes.
 - u - Update only. It means do not compile classes that are up-to-date.
 - v - Keep valid. When combined with "f" flag, also keeps the objects valid after the compilation of the new classes.
 - y - Include classes that are related to the current class in the way that they either reference to or are referenced by the current class in SQL usage.
- 4 - Export CDL compatible with version 4 from later version.
 - o1- Optimize ..Property to i%Property where possible.
 - o2- Optimize calls within this class, no incremental compile support.
 - o3- Optimize calls within this class and to system classes.
 - o4- Optimize calls to all classes, only works from CompileAll entry point.

Default flags for this namespace =adl01

You may change the default flags with the SetFlags(flags,system) call

注釈: “f” フラグを使用すると、既存のオブジェクトは、コンパイル後には無効になります。ユーザがクラスのコンパイル後もオブジェクトを有効にしたい場合は、“v” フラグを使用します。

頻繁に使用するユーティリティ・メソッドがいくつかあります。##class(Package.Class).method() を使用せずに呼び出すことはできますか？

それらのメソッドを特定のクラスに配置すれば、CSP ページがそのクラスから継承することができます。そうすれば、ドット構文を使用してそのメソッドにアクセスできます。これを行うには、以下のよう
に <csp:class> タグを使用します。

```
<csp:class super="%CSP.Page,App.Utils">
```

プライベート・ページとは何ですか？

プライベート・ページは、適切なトークンを使用して他の CSP ページから呼び出されたときに表示されます。プライベート・ページにブックマークを付けることはできず、ブラウザに URL を入力するだけでは表示することはできません。ページをプライベート・ページとして設定するには、以下のよう
に <csp:class> タグを使用します。

```
<csp:class private=1>
```

すべてのページで使用したい JavaScript 関数とヘッダがあります。これをページに含むには、どのようにすればいいのでしょうか？

新しい `#include` 構文を使用します。

```
<!--#include file="mystuff.inc"-->
```

これはテキスト形式のインクルード・ファイルで、Caché 5 の新機能です。ファイルのテキストは、ページがコンパイルされる前に自動的にページに組み込まれます。したがって、`#()` 変数や `<csp:query>` クエリの結果、`runat="server"` コードなど、コンパイルする必要があるコードを含むことができます。

`<csp:search>` タグを使用し、ユーザが ID 以外でもフィールドを検索できるようにもしたいのですが、これは可能でしょうか？

`<csp:search>` タグには **WHERE** 属性があります。この属性を使用して、検索するフィールドをコマ区切りのリストで指定することができます。

```
<csp:search name=mySearch where="Name,Gender" CLASSNAME="Sample.Person">
```

他にも、`<csp:search>` 機能をカスタマイズするために使用できる複数の属性があります。詳細は、“CSP HTML タグ・リファレンス”の“`<CSP:SEARCH>`” エントリを参照してください。

構成

サブディレクトリ内のページを供給する CSP アプリケーションは、どのように構成するのですか？

Caché の システム管理ポータルを使用します。

1. Caché キューブ から **[システム管理ポータル]** をクリックします。必要な場合は、ログインします。
2. **[システム管理ポータル]** のメイン・ページで **[システム管理]** 列から **[セキュリティ管理]** を選択すると、**[セキュリティ管理]** ページが表示されます ([[ホーム]→[セキュリティ管理]])。
3. **[セキュリティ管理]** ページで **[アプリケーション定義]** 列から **[CSPアプリケーション]** をクリックすると、**[CSPアプリケーション]** ページが表示されます ([[ホーム]→[セキュリティ管理]→[CSPアプリケーション]])。
4. **[CSPアプリケーション]** ページで構成対象のアプリケーションに対応する **[編集]** をクリックすると、そのアプリケーション用の **[CSPアプリケーション編集]** ページが表示されます ([[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]])。
5. **[CSPアプリケーション編集]** ページで、**[繰り返し]** を “[はい]” に設定します。
6. **[保存]** をクリックします。

現在、Windows 用の Macromedia Dreamweaver MX を持っています。CSP との作業を行うためには、どのように構成すればいいのでしょうか？

Dreamweaver を使用するには、Macromedia Extension Manager を使用して、CSP 拡張ファイル (**C:\¥CacheSys¥Dev¥csp¥dreamweaver¥CacheServerPagesMX.mxp**) をインストールします。この拡張ファイルのインストール方法、および使用方法に関する詳細は、“Cache Server Pages (CSP) の使用方法” の “Macromedia Dreamweaver での CSP 使用方法” を参照してください。

現在、Windows 用の Macromedia Dreamweaver バージョン 4.0 以降を持っています。CSP との作業を行うためには、どのように構成すればいいのでしょうか？

Dreamweaver を使用するには、Macromedia Extension Manager を使用して、CSP 拡張ファイル (**C:\¥CacheSys¥Dev¥csp¥dreamweaver¥CacheServerPages.mxp**) をインストールします。この拡張ファイルのインストール方法、および使用方法に関する詳細は、“Cache Server Pages (CSP) の使用方法” の “Macromedia Dreamweaver での CSP 使用方法” を参照してください。

現在、Macromedia Dreamweaver 3.0 を持っています。CSP との作業を行うためには、どのように構成すればいいのでしょうか？

<http://www.macromedia.com/exchange/dreamweaver/> から Macromedia Extension Manager をダウンロードし、“Cache Server Pages (CSP) の使用方法” の “Macromedia Dreamweaver での CSP 使用方法” の手順に従って、CSP 拡張ファイルをインストールしてください。

ユーザのブラウザに `/csp/` ではなく、<http://mydomain.com/banking/login.csp> と入力させ、CSP アプリケーションをロードしてほしいのですが、どのようにすればいいのでしょうか？

システム管理ポータルを使用して、新規の CSP アプリケーション (例えば `/myapp`) を設定します。このプロセスは、“Cache Server Pages (CSP) の使用方法” の “CSP 構成” の章の “CSP サーバの新規アプリケーションの定義” のセクションで詳細に説明しています。

Web ブラウザとは異なるマシンで Cache を起動しています。どのように構成すればいいのでしょうか？

“リモート Web サーバでの Cache Server Pages の使用方法” の技術情報を参照してください。

その他

CSP アプリケーションでフレームを使用することはできますか？

はい。しかし、フレームセット・ページには、**.csp** 拡張子を持つ名前を付ける必要があります。**index.html** というページを作成し、CSP ページを左フレームと右フレームにロードする場合、2 つのセッションが使用することとなり、各 CSP ページに 1 つずつ、計 2 つの Caché ライセンスが必要となります。これは、情報を保存するためにセッション・オブジェクトを使用する場合に非常に混乱し、また、不必要なライセンスを使用する必要もあります。

index.csp フレームセット・ページを呼び出すことで、セッションもそのアプリケーションのライセンスも 1 つで済みます。フレーム内の両方の CSP ページは、このセッションと、セッションに保存されているすべての情報を共有します。

CSP で文字セットを使用するには、どうすればいいのでしょうか？

“文字セットと CSP” の技術情報を参照してください。

どの HTTP ヘッダ情報がブラウザに送信されるのでしょうか？

ヘッダ情報を表示するには、以下の 2 つの方法があります。

- ・ ターミナル で ShowPage メソッドを使用して、ページを表示します。

```
D $System.CSP.ShowPage("/csp/user/mypage.csp")
```

これにより、このページに対して生成された HTML ソースと HTTP ヘッダが表示されます。

- ・ **%Net.HttpRequest** クラスの Head メソッドを利用します。

```
set http = ##class(%Net.HttpRequest).%New()  
set http.server = "localhost"  
set http.Port = 8972  
do http.Head("csp/samples/loop.csp")  
do http.HttpResponse.OutputToDevice()  
do http.%Close()
```

CSP に加え、**.csp** 拡張子を使用する Crystal Reports も使用しています。Caché Server Pages を使用するには、どのようにすればいいのでしょうか？

CSP と Crystal Reports は両方とも **.csp** ファイル拡張子を使用するので、Web サーバで両方を起動すると衝突が起きます。後にインストールされたアプリケーションは正常に起動しますが、先にインストールされたアプリケーションは正常に作動しません。このような衝突を回避するには、Web サーバ構成で CSP 用に 1 つの仮想ディレクトリを、Crystal Reports には別の仮想ディレクトリを作ります。

インターネット サービス マネージャ を使用して仮想ディレクトリを構成する方法は、以下のとおりです。

1. [スタート] メニューから [設定]、[コントロール パネル]、[管理ツール] を選択し、[インターネット サービス マネージャ] をクリックします。
2. 最初のノードを拡張し、[既定の Web サイト] を展開します。
3. CSP が後にインストールされている場合、[Crystal] 仮想ディレクトリを右クリックし、[プロパティ] を選択します。

Crystal Reports が後にインストールされている場合、[csp] 仮想ディレクトリを右クリックし、[プロパティ] を選択します。

4. [プロパティ] ダイアログ・ボックスの [仮想ディレクトリ] タブで、右下にある [構成] ボタンをクリックします。
5. [アプリケーションのマッピング] タブをクリックし、リストの下方にある .csp マッピングをスクロールして検索します。
6. CSP を後にインストールした場合は、.csp 拡張子マッピングの [実行ファイル] を Crystal Reports の DLL である WSCInSAPI.dll に変更します。この DLL は Crystal のインストール・ディレクトリの WCS ディレクトリにあります (例えば、C:¥Program Files¥Seagate Software¥WCS)。
Crystal Reports を後にインストールした場合は、.csp 拡張子マッピングの [実行ファイル] を Caché のインストール・ディレクトリの /csp/bin にある CSPms.dll の位置に変更します (例えば、C:¥CacheSys¥CSP¥bin)。
7. [OK] ボタンをクリックします。