

IT 特別講座

Apache http web server を 動かしてみよう！

動きのあるプログラムを作る 3 種類の方法とは？

LA-Linux 専任講師 矢越昭仁

2009/04/25

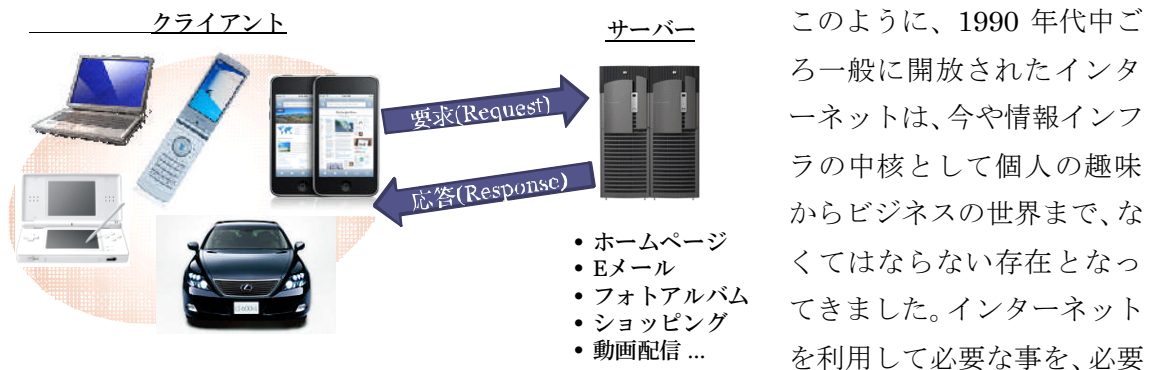
Linux ベーシックコースで構築する Apache Web Server は Windows PC でも動作させる事ができます。Windows PC で、お手軽に Web Server を立ち上げて、いろんな実験をしてみましょう。

目次

1.はじめに	4
1.1 インターネットとは.....	4
1.2 サーバーとは.....	5
2 Web サーバーとは.....	6
2.1 URL	6
2.2 サーバー構成.....	7
2.3 サーバー増強.....	7
3. Web ページ.....	8
3.1 静的なページ.....	9
3.2 動的なページ1 (CGI)	10
3.3 動的なページ2 (サーバーページ)	11
3.4 動的なページ3 (スクリプト)	12
3.5 動的ページのまとめ.....	13
やってみよう	14
Apache の紹介	14
外部との接続.....	18
ファイアウォール	18
Perl 紹介.....	21
Perl インストール.....	22
サンプル CGI.....	24
Perl の簡単な解説(printenv.pl).....	25
簡単な Perl の改造(printenv2.pl)	26
表形式への改造(printenv3.pl).....	28
HTML	30
少し高度なCGI	31
FORM 文.....	31
サンプル CGI(sampleform.pl)	32
後片付け	32

1.はじめに

OECD¹によれば 20 世紀が工業品を製造・購入するといった「モノ社会」だったすれば、21 世紀はサービスの提供や知識の共有・共感といった「コト社会」へと移行していくと考えられています。今までは CD や雑誌をお店に買いに行き、自宅に所有していた訳ですが、最近では聞きたい音楽やビデオをネットでダウンロード、本もネットで検索して必要なものを注文すれば 1~2 日で手元に届く時代です。さらに、そもそも新人アーティストの楽曲の評判や、映画の評論などの比較情報もネットから入手している人が多いでしょう。



なタイミングで、どこにいても利用できる世の中だといえるでしょう。PC がなくても携帯さえあればメールのやり取り、音楽のダウンロード、乗換案内、チケットの購入などいろんな事ができるようになっています。

こうしたネットワークを使って、いろんなサービスを授受できる仕組みはどうやって実現できているのでしょうか。この仕組みを作る方法を解説しているのが、LA の「Linux ベーシック」コースや「Linux マスター」コースです。

1.1 インターネットとは

1980 年代までは、インターネットは一般には公開されていませんでした。その頃のネットワークはパソコン通信、略してパソ通と呼ばれていました。電話回線をつかって FAX の原理によって接続された回線はとても遅く、9,600bps 程度の速度でした。今、通信業者が広告を派手に売っている光ファイバーは 100Mbps なので、1 万分の 1 以下の速度です。ポータブル音楽プレーヤー用の 1 曲をダウンロードするのに 20 分近くかかる計算です。

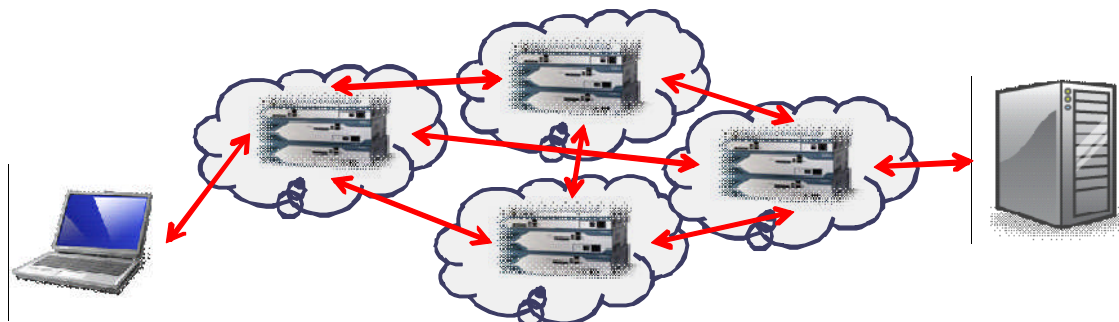
そんな技術だったので、通信する内容はもっぱら文字だけのメールが中心で、よくても新聞の記事にある程度の荒い写真ぐらいしかやり取り出来ませんでした。

またやりとりする相手も同じプロバイダーに加入している人だけに限定されたり、利用できる PC のメーカーが限定されていたりと、とにかく小さな集団の世界でしか利用できない状態でした。

こういった状況を解決したのがインターネットでした。従来プロバイダー毎に閉じたネッ

¹ OECD: Organization for Economic Cooperation and Development 経済協力開発機構。経済先進国・新興工業国による国際経済全般を協議する国際的な組織。

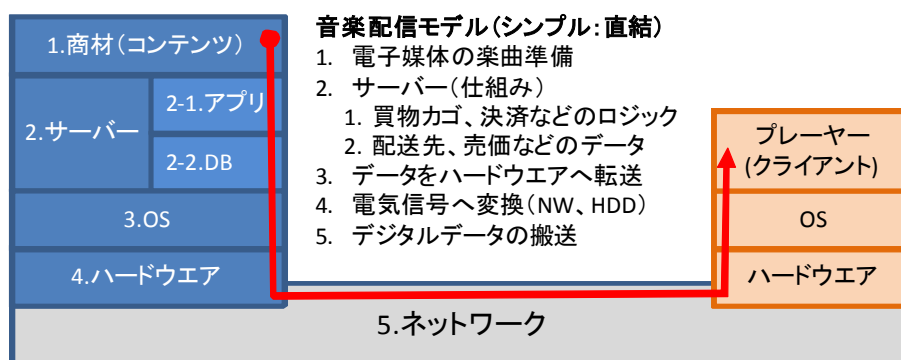
トワークを相互接続して、世界中のネットワークが 1 つになりました。その利便性から企業での本格的な導入も始まり、日本は世界でもトップクラスの高速で安定したネットワークが実現されています。



1.2 サーバーとは

さてネットワークを高速道路に例えるならば、そこを流れる自動車が必要です。たとえば音楽ファイルといった荷物を目的地まで届けてくれるトラックや倉庫・店舗が必要です。こういった情報を作って送り届けてくれる機能がサーバーです。

例えば楽曲という商品をネットワーク経由で、客先に送り届ける場合を考えてみましょう。



アーティスト名や曲名、ジャンルなどから特定の曲を検索するといった機能はビジネスロジックと呼ばれます。それをお客さんに届けるには、顧客情報である送り先情報、過去にダウンロードした記録など情報を蓄積しておく必要があります。ここは DBMS(Data Base Management System、略して DB)と呼ばれています。このビジネスロジックとデータを動かす仕組みがサーバーと呼ばれていて、基本ソフトに対し応用ソフト(アプリケーション・ソフトウェア)と呼ばれています。

2 Web サーバーとは

インターネット上のサービスで最も利用されているのは、Web でしょう。いろんなメーカーやお店のサイトで商品の情報を得たり、ポータルサイトでニュースや天気予報を見たり、インターネットのまさに入口としての機能を提供しています。

Web は正確には World Wide Web といって、インターネット上のあらゆるコンピュータに配置された情報をブラウザソフトで参照するという仕組みです。1990年に仏 CERN(セルン、サーン。欧州原子核研究機構、略号はその前身である研究所設立準備理事会 **Conseil Européen pour la Recherche Nucléaire** から)の研究者だったティム・バーナーズ=リーによって開発・発表されました。最初は文字情報だけしか扱えなかったのですが、発表後すぐ 1992 年米イリノイ大学の NCSA(米国立スーパーコンピュータ応用研究所 **National Center for Supercomputing Applications**)のマーク・アンドリーセンを始めとする学生たちによって拡張され画像や音声などマルチメディアに対応できるようになりました。

2.1 URL

世界中のコンピュータにある情報を一意に表すためには、ネットワークに接続されたコンピュータ名、参照したファイルの場所(ディレクトリやフォルダとファイル名)と、その種類が必要です。この世界中のどこかにあるコンピュータのファイルを表す名前を URL(Uniform Resource Locator)といいます。

書式は以下の様になります。

スキーム://[ユーザ名[:パスワード]@]ホスト名[:ポート]/パス[?引数…]

例)

<http://www.yahoo.co.jp/>

http://www.google.co.jp/search?q=Apache&lr=lang_ja&ie=utf-8&oe=utf-8&aq=t&rls=org.mozilla:ja:official&client=firefox-a

要素	解説
スキーム	扱うデータ種別を表し、多くの場合はプロトコル名と同じです。【必須】 例) http, https, ftp など
ユーザー名	FTP や Basic 認証(後述)などで必要な場合に指定します。
パスワード	ユーザー名同様、必要に応じ指定します。正し平文となるためセキュリティの関係上、利用する時は十分に検討すべきです。
サーバー名	接続先のサーバー名を指定、IP アドレスも利用できます。【必須】
ポート番号	サーバープログラムに割り当てられた番号を指定します。 スキームで http を指定した時の省略値は 80 です。
パス名	サーバー上で一意に表されるデータのありか。フォルダ名だけで、ファイル名が省略された場合 index.html が採用されます(設定による)。
引数	Web サーバー上で動作するプログラムへの値を指示します。

2.2 サーバー構成

URLによって、インターネット（世界中）から必要な情報（コンテンツ）が特定できれば、サーバーはURLにひもづくデータを要求があった、クライアント（ブラウザ）に返すだけです。

しかしいつもサーバーとブラウザが1：1でやり取りしているわけではありません。特に人気があるサイトは1つのサーバーが膨大なクライアントを相手にしています。そのためサーバーを任されているコンピュータには高性能が要求されます。

Yahoo! のように非常にアクセスの多いサイトではURLで指し示すサーバーを複数のコンピュータで分担しています。

```
C:\Users\ayakoshi>nslookup www.yahoo.co.jp
```

```
サーバー: dns-a.bbtec.net
```

```
Address: 218.176.253.65
```

```
権限のない回答:
```

```
名前: www.yahoo.co.jp
```

```
Addresses: 124.83.167.212
```

```
203.216.227.176
```

```
203.216.235.154
```

```
:
```

この様に一つの仕事を複数のコンピュータで分担する方法と、巨大な専用コンピュータを用意する方法があります。

2.3 サーバー増強

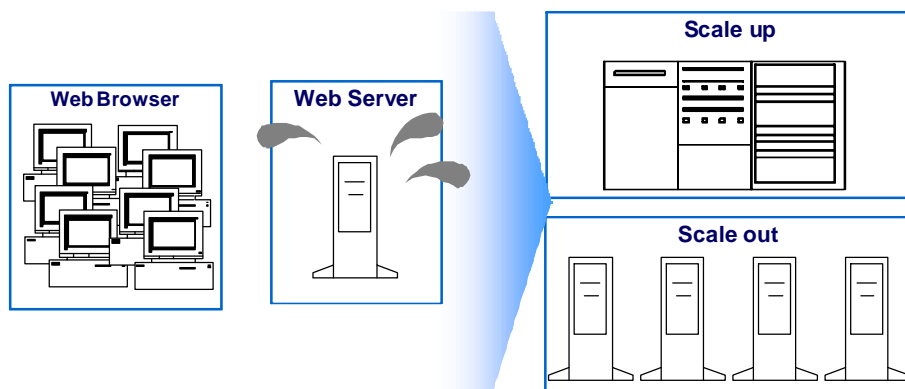
ホームページの人気上昇すると、それまでのサーバー性能では対応しきれない場合が発生します。そんな時、サーバー機を増強するには、以下の2つの方法があります。

1. スケール・アップ（高性能コンピュータへ切り替える）

サーバーのコンピュータに、必要なハードウェアを追加・増強、より強力な機種に交換する方法です。原則として増強作業中はシステムを停止させる必要があります。

2. スケール・アウト（コンピュータを追加する）

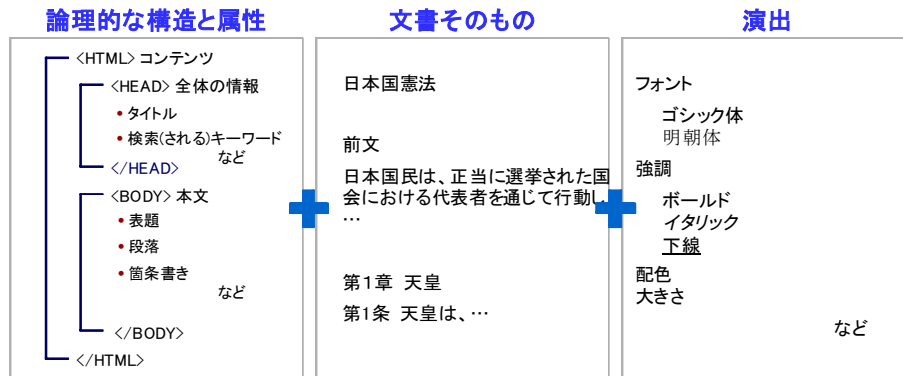
今使っているコンピュータと同等品を追加する方法で、廉価なコンピュータを使う事ができます。



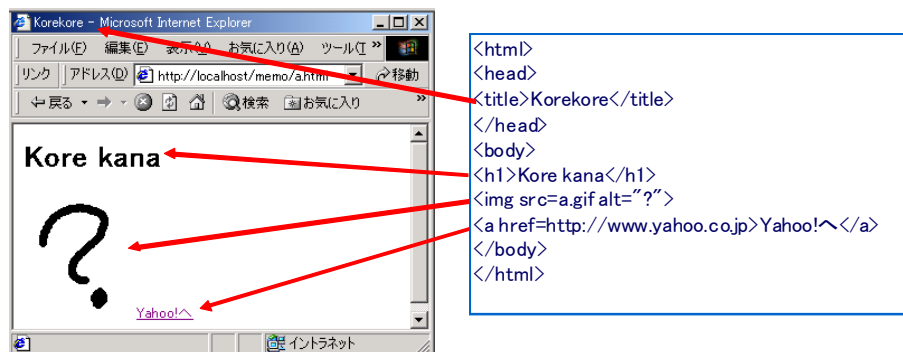
3. Web ページ

繰り返しになりますが、Web サーバーの基本は、クライアントから URL で指定された情報（コンテンツ）に対する要求にこたえ、結果をクライアントへ応答する事にあります。

このコンテンツは HTML と呼ばれる形式で表現され、文書の論理構造や演出効果を指定できるようになっています。



このような構造をブラウザへの指示する部分をタグと呼び不等号で囲みます。また HTML は1つの文書を書きとめる機能だけではなく、他の文書の参照（リンク）や、図・楽音といったマルチメディア情報の取り込みといった機能も提供します。

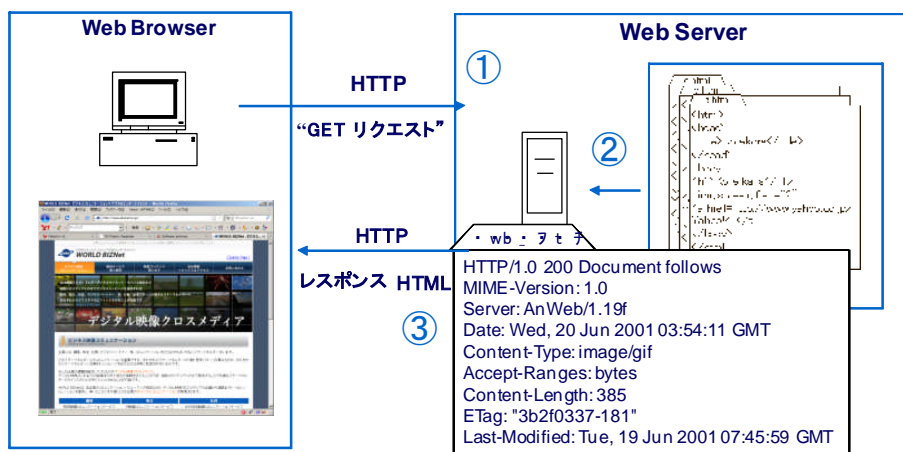


こうして出来た HTML で表現される画面をページと呼びます。ホームページは、Web サーバーにアクセスしたときに最初に表示されるページを指します。

ページには常に同じ内容が表示される「静的なページ」と、ユーザーの操作や時間、直前に見ていたページ内容などによって、表示される内容が変動する「動的なページ」と呼びます。

3.1 静的なページ

予め用意されたコンテンツ（HTML ファイル、画像・音声ファイルなど）を貯蔵しリクエストに応じ、適切なコンテンツを返すサーバーです。単純な Web ページの処理は以下のようにになります。



1. ブラウザーからリクエストを受ける
2. URLにあるファイルを探す（無ければエラーを返す）
3. コンテンツに応じた付加情報を付けて、コンテンツを返す。

さらにコンテンツ内に別の映像や楽曲といったコンテンツがあれば、それも繰り返しヘッダを付加して逐一送りつけます。

このように事前に用意されたコンテンツを返すページを「静的なページ（Static pages）」と呼びます。

（補足）

Web サーバーはクライアントのリクエストに応じて、自分が持つファイルなどを提供するため、セキュリティに注意しなければ不適切な情報が引き出されてしまう可能性があります。例えば顧客名簿や、仕入単価、上場前の経営情報など一般に漏洩してしまうと、会社経営に直接影響がでてしまう情報などがあります。

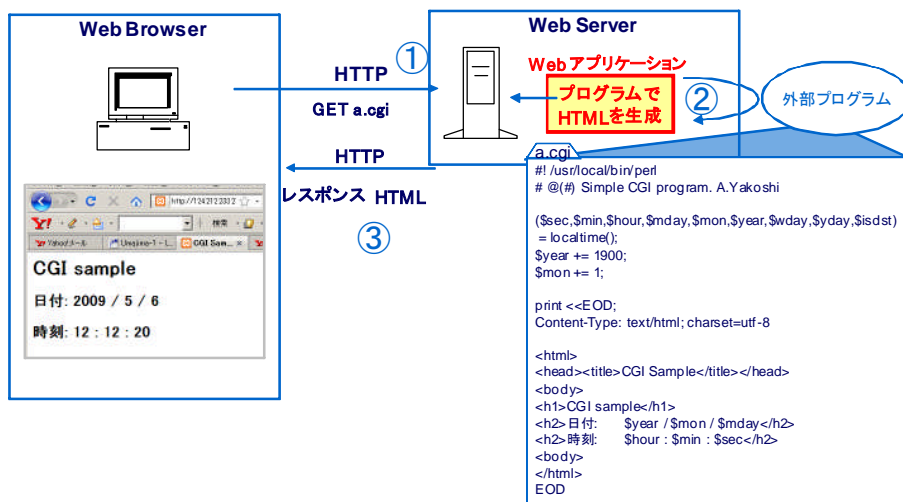
こういった情報漏洩リスクを回避するために、Web サーバーにはいろんな機能が搭載されています。ユーザー名とパスワードで利用者を限定したり、暗号化して通信したり、参照できるディレクトリを限定したりと様々な機能がありますが、それらを有効・無効にするためにサーバーの設定が必要です。

特に次節から解説する「動的ページ」など便利な機能を使えば使うほど、情報漏洩リスクは増える傾向にあります。そのため、システム管理者はサーバーの機能について知識と設定できる技術が必要です。

3.2 動的なページ 1 (CGI)

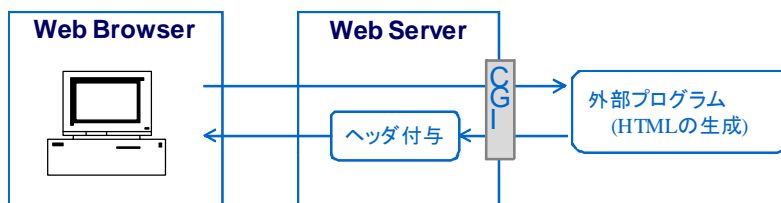
Web ページは単に事前に用意されたファイルを表示するといった単純なものだけでなく、最近ではユーザーの操作によってその内容・構成を組み替えるものが増えてきました。

プログラムを実行する方法のうち、Web サーバーが他のプログラムを起動してコンテンツを生成する方法を CGI(Common Gateway Interface)と呼びます。



1. ブラウザーからのクエストを受け取る
2. 外部プログラムを起動し、コンテンツを作る
3. 作成されたコンテンツをクライアントへ返す。

CGI は正確には、Web サーバーが外部プログラムを呼び出す接点を指し、プログラムは単体で動作します。

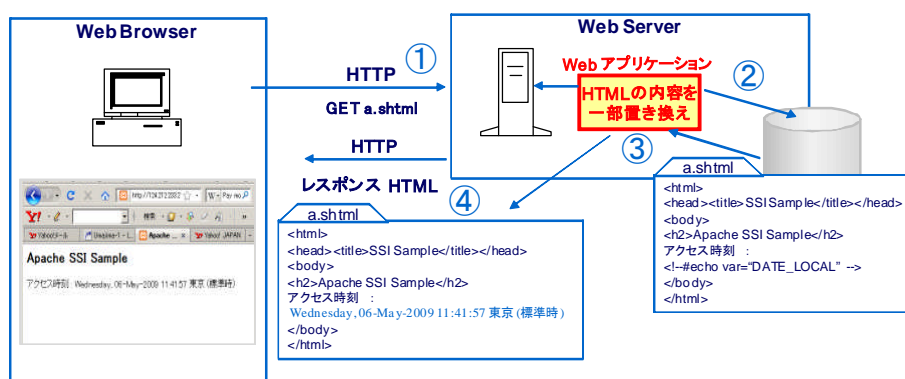


外部プログラムが生成するのは、コンテンツ本体 (Content-Type と HTML の構文) だけで、それを受けとった Web サーバーは静的ページ同様、ヘッダ情報を付与しクライアントへ返します。

3.3 動的なページ 2 (サーバーページ)

最後の方法は、Web サーバーにてプログラムを動作させ、HTML を生成させる方法です。SSI(Server Side Include)又はサーバーサイドページと呼ばれ、CGI と違い Web サーバーの中でプログラムが動作します。SSI は予め用意した HTML 内に特殊なキーワードを差し込み、アクセスがあった時にそのキーワード部分を差し替えます。PHP もこの方式を用いています。サーバーサイドページには、マイクロソフト社の ASP(ActiveServer Pages)と、Java を用いた JSP(Java Server Page)があります。

Apache に組み込まれた SSI を用いるには、予めキーワードを組み込んだ HTML を用意するだけで、とても簡単に利用する事ができます。このキーワードは文法上 HTML のコメントとして扱われるので、SSI の動作を制限しているサーバーでは単にコメントとして無視される仕組みになっています。



1. ブラウザーからのクエストを受け取る
2. 指定されたコンテンツの検索、準備 (拡張子は.shtml)
3. 特定キーワードの置換 (この場合は、#echo に関する行)
4. 置き換えた結果をレスポンスとして、クライアントへ返信

Apache では HTML に以下の構文を差し込むと SSI として解釈されます。

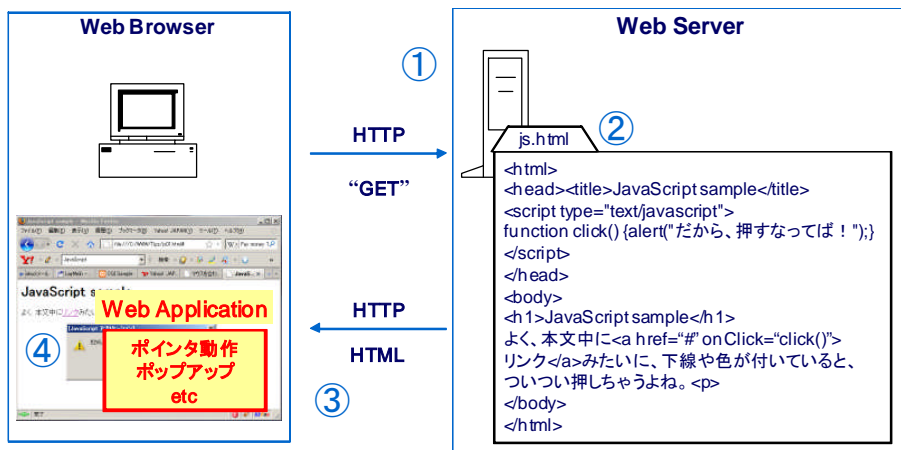
```
<!--#命令 データ名="データ値" -->
```

Apache で利用できる、主な SSI の例

HTML 例	解説
<!--#echo var="DOCUMENT_NAME" -->	ファイル名を表示
<!--#echo var="DATE_LOCAL" -->	アクセス日時を表示
<!--#echo var="LAST_MODIFIED" -->	ファイルの最新更新日時
<!--#include file="ファイル" -->	ファイルの取り込み
<!--#filesize file="ファイル" -->	ファイルの大きさを表示

3.4 動的なページ3 (スクリプト)

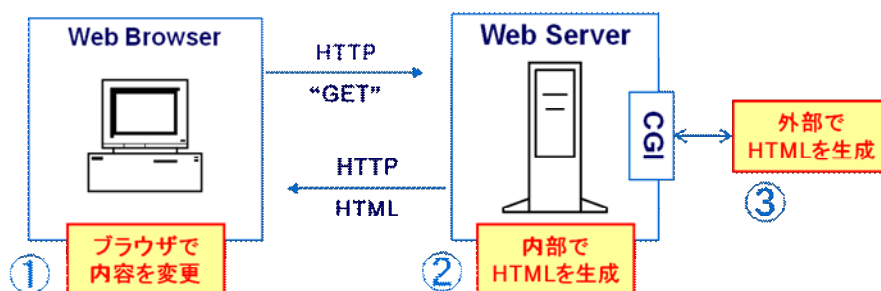
ブラウザ上でマウスポインターに反応して図柄が変わったり、ポップアップが表示されるようなページがあります。この場合プログラムが動作するのはサーバー上ではなく、ブラウザで動作しています。



1. ブラウザーからのリクエストを受け付ける
 2. 事前に用意されたコンテンツを探す
 3. ヘッダを付けて、クライアントに返す
 4. ブラウザーは受け取ったページに組み込まれたプログラムを実行します。
- このようにページに組み込まれたプログラムをスクリプトと呼び、**JavaScript** が最も普及しています。

3.5 動的ページのまとめ

動的ページの動作メカニズムをまとめると以下のようになります。



1. スクリプト :
ブラウザの中で動作するプログラム
2. SSI
Web サーバーの中で動作するプログラム
3. CGI
Web サーバーが呼び出す外部プログラム

このように動的なページを作る場合、プログラムがどこで動作するかで分類することができます。これらは混在してもかまいません。また CGI を使った大規模なサイトでは、CGI 自体を複数のサーバーで動作させることがあります。

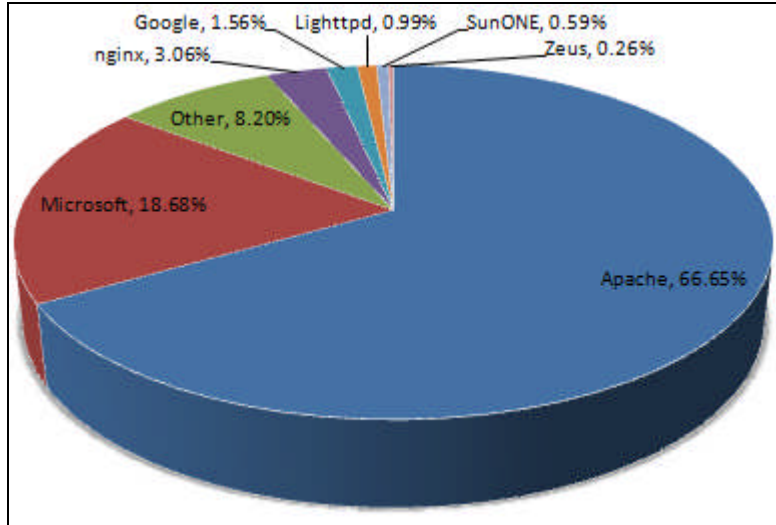
<p>簡易サーバー 単純に HTML だけを返す。 とくにプログラムは動かさない</p>	
<p>単一サーバー Web サーバー上でプログラムや、DB など全てを動かす</p>	
<p>2階層サーバ Web サーバから、DB を切り出し専用のサーバで一元管理を行う</p>	
<p>3階層サーバ Web サーバ上ではブラウザとの通信のみ。アプリケーション DB は専用のサーバで一元管理を行う</p>	

やってみよう

Apache の紹介

まずは Web サーバーをインストールします。今回このコースで使うサーバーは Apache HTTP Server で、Linuxをはじめ、Mac, Windows, 大型コンピュータなど多くの基本ソフトで動作します。インターネット上で利用されているソフトウェアを調査している Netcraft (<http://news.netcraft.com/>) によると、世界中でアクセスの多いサイト、TOP 100 万で使われているサーバーの約 70%が Apache です。

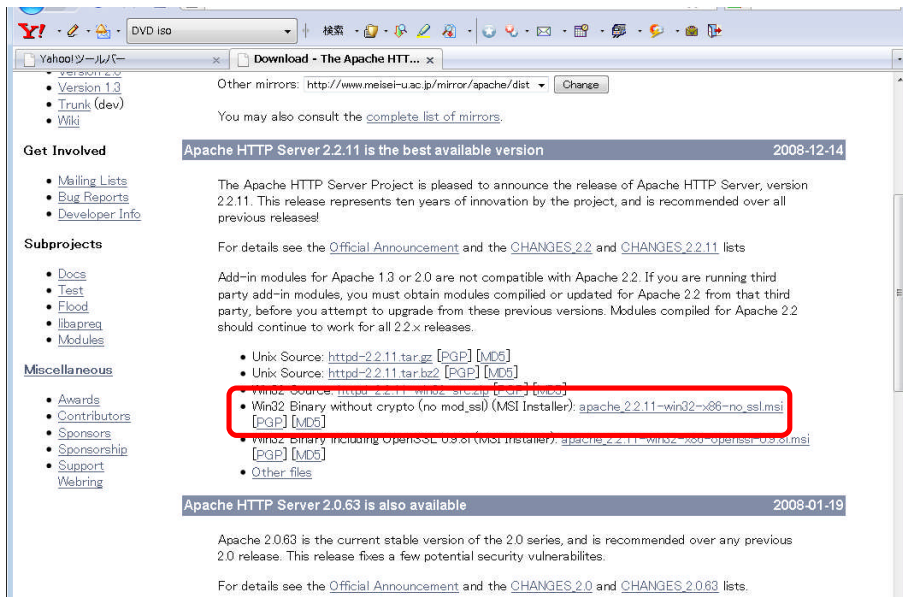
Server Share amongst the Million Busiest Sites, March 2009



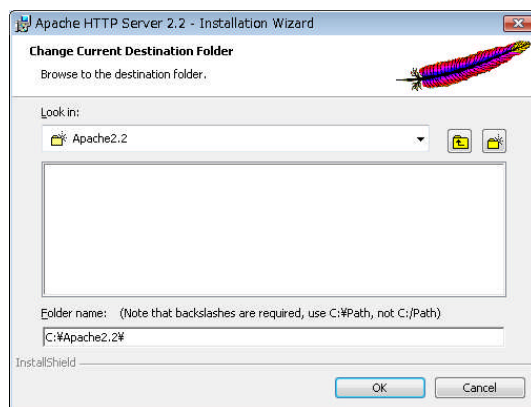
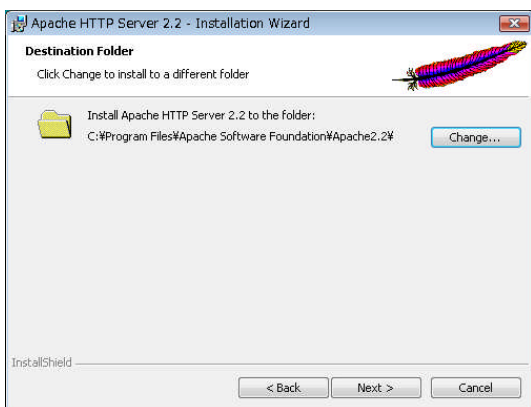
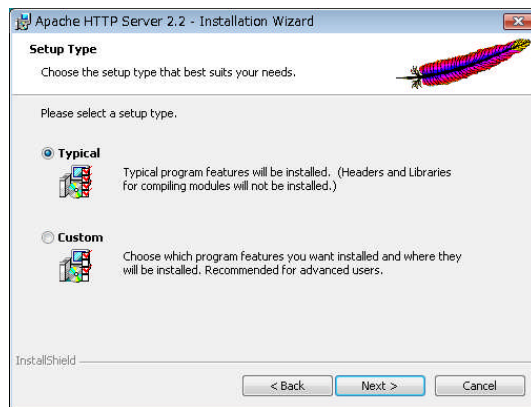
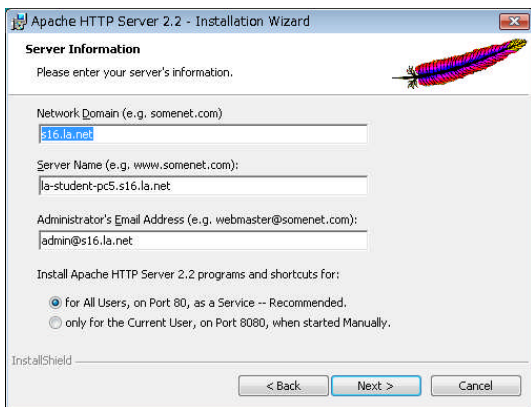
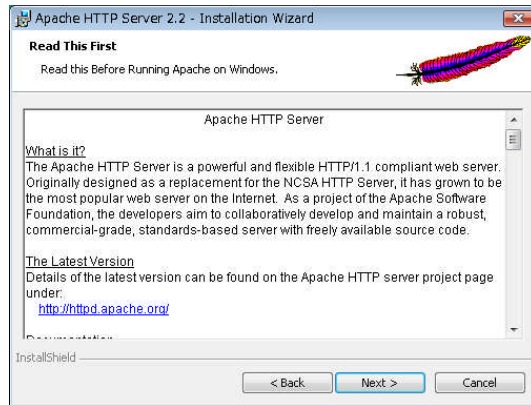
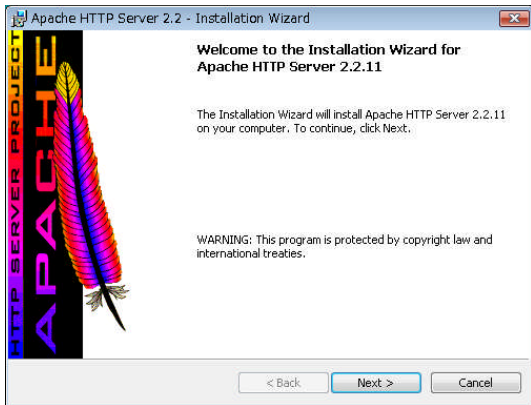
Apache のインストール

Apache は OSS(Open Source Software)で、<http://www.apache.org/>から無償でダウンロードできます。現在公開されている Apache Web サーバーは Ver. 2.2, 2.0, 1.3 の 3 種類がありますが、今回は最新の 2.2 を用います。

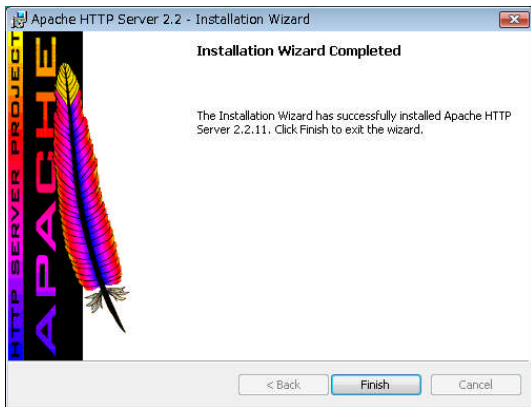
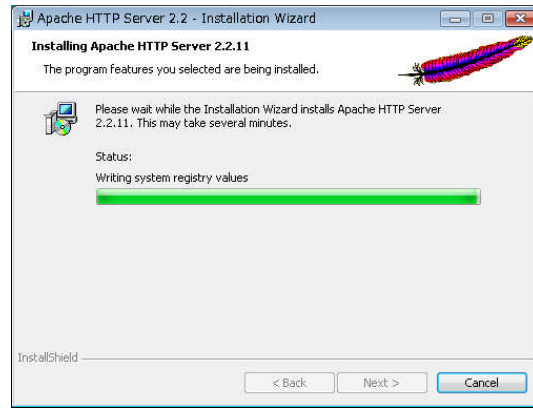
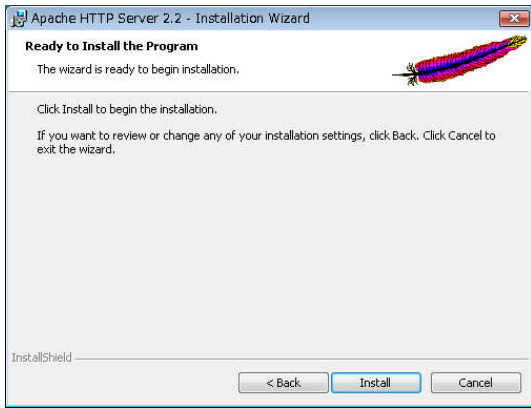
ホームページから HTTP Server > Win32 Binary without crypto をダウンロードします。



MSI 形式のファイルをダブルクリックしてインストールを開始します。
基本的にインストーラの省略値を用いますが、インストール先だけは C:\¥Apache2.2 というようにローカルディスクの直下に置きましょう。

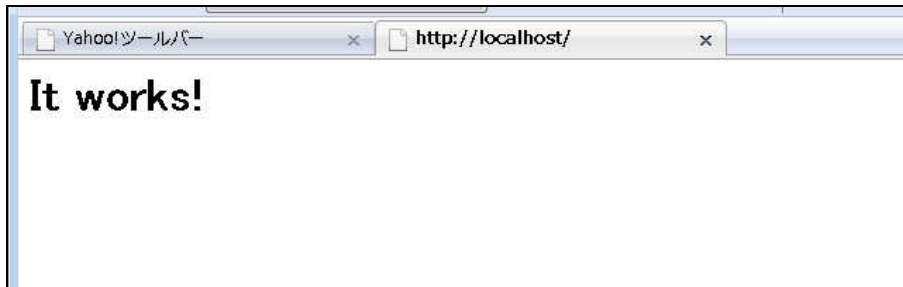


インストール先は [Change] で、ローカルディスクの直下に変更



9 DONE!

ブラウザを起動して、URL に <http://localhost/> と入力し、「It works!」が表示されれば正しく動作しています。

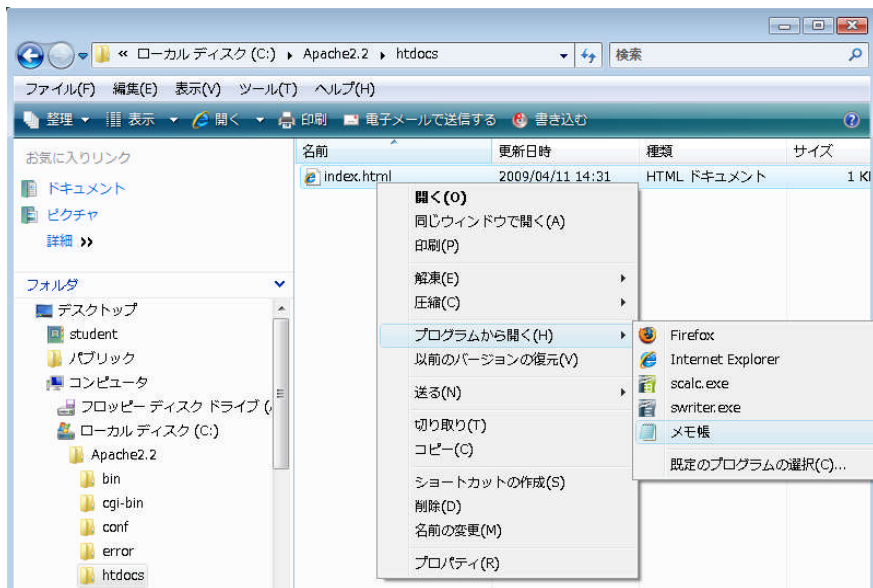


最初に表示されるこの画面は、Apache をインストールしたディレクトリの下にある、htdocs の index.html というファイルです。

静的ページの確認

では、このテスト画面を修正し表示内容が変わることを確認してみましょう。

index.html をメモ帳で開きます。




内容を修正して、上書き保存しブラウザの再読み込みをすると、書き変わるのがわかるはずです。

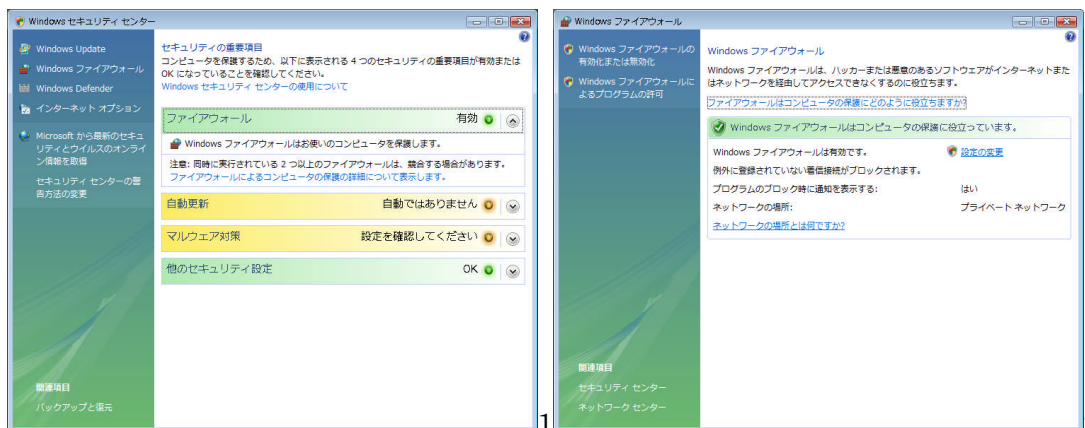


HTML の知識があれば、このディレクトリの下にホームページを作成することができます。

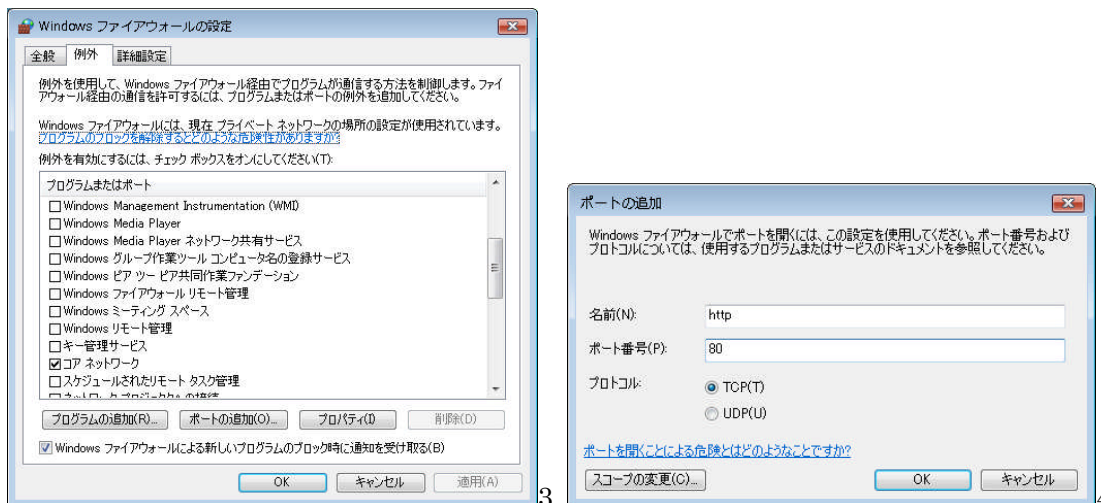
ファイアウォールはネットワークを経由してコンピュータに接続する際に、その通信の種類と方向（コンピュータの内から外か、外から内か）を判別して、特定の通信だけを通すソフトウェアです。この図ではブラウザの周りはファイアウォール（赤い枠）で覆われていて、ホームページへリクエストを出して、その答えを受ける事はできます。しかし外からのリクエストは拒否するような設定になっています。

皆さんの PC も特に指定しない限り、外部からの要求には答えない設定になっています。そこで、外部からの Web アクセスを許可するよう、ファイアウォールを設定します。

メニューバーの楕円アイコンをクリックし、セキュリティセンターを呼び出します。



左側のメニュー「Windows ファイアウォール」をクリック、次の画面では右側の「設定の変更」をクリックします。



「例外」タブをクリックし、ポートの追加をクリックします。

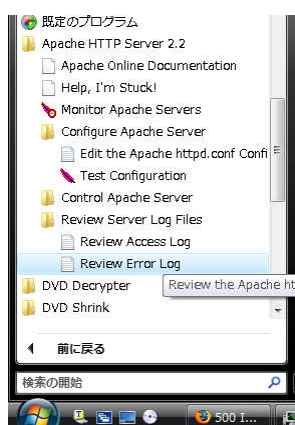
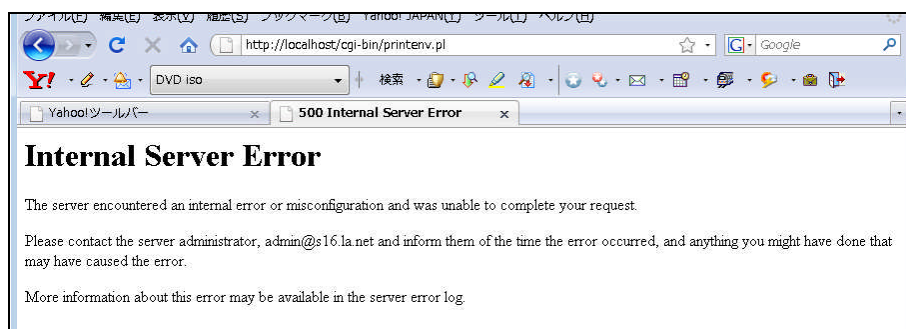
ポートの追加では、名前に「http」、ポート番号に「80」を指定、プロトコルは●TCPを選択し[OK]を押します。

これによって、外部からの Web ブラウザーによるアクセスが許可されます。

隣の人や、講師席から自分の PC にアクセスできるか確認してください。

確認できたら、次はいよいよ CGI を実行します。

Apache には最初から Perl と呼ばれるプログラムで作られた、サンプル CGI として <http://localhost/cgi-bin/printenv.pl> が搭載されています。ブラウザから起動すると、エラーになるはずですが。



この時のエラー内容はメニューから Apache HTTP Server 2.2 > Review Server Log files > Review Error Log を選択します。

するとメモ帳が起動され、エラーの詳細が表示されます。

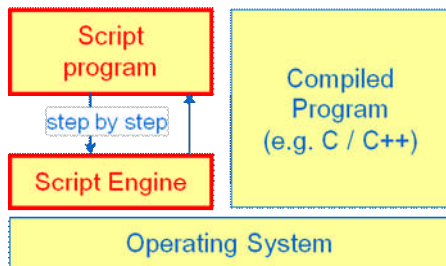
このエラーの内容は、「Perl プログラムがなくて、実行できない」旨を表しているはずですが。

内容を確認したら、メモ帳を閉じてください。

そこで、Perl プログラムを入手しインストールします。Perl は Windows でも Linux でも動く、強力なプログラミング言語で CGI などによく用いられます。

Perl 紹介

Perl は UNIX で誕生した、非常に強力（何でもできる、でも難解な）プログラミング言語です。現在では Linux をはじめ多くの OS 上で動作するフリーのスクリプト言語として定着しています。Linux では標準で搭載されますが、Windows ではサイトからダウンロードする必要があります。



スクリプト言語はインタプリタ言語とも呼ばれ、テキストファイルのプログラムを逐次実行します。

スクリプト言語は一般的に他のアプリケーションやツールとの連携が簡単に行えるという特徴がありますが、実行速度は他の形式（コンパイラ言語、アセンブラ言語）に比べると若干遅くなります。

スクリプト言語には他にも、以下のようなものがあります。

- ✓ JavaScript

Java の文法に似たスクリプト言語。Web ブラウザーによく用いられますが、セキュリティを考慮し制限事項が多く課せられています。

- ✓ VBScript

マイクロソフトのスクリプト言語で Windows の各種機能呼び出し、Office シリーズなどのマクロなどにも使われる事があります。

- ✓ Bash

Linux に指示するときに使っているアレです。

中でも Perl は早い時期（20 年ほど前）に誕生し、世界中に広まったので便利なツールが沢山用意されています。CPAN を探せば、大抵のものは置いてあります。

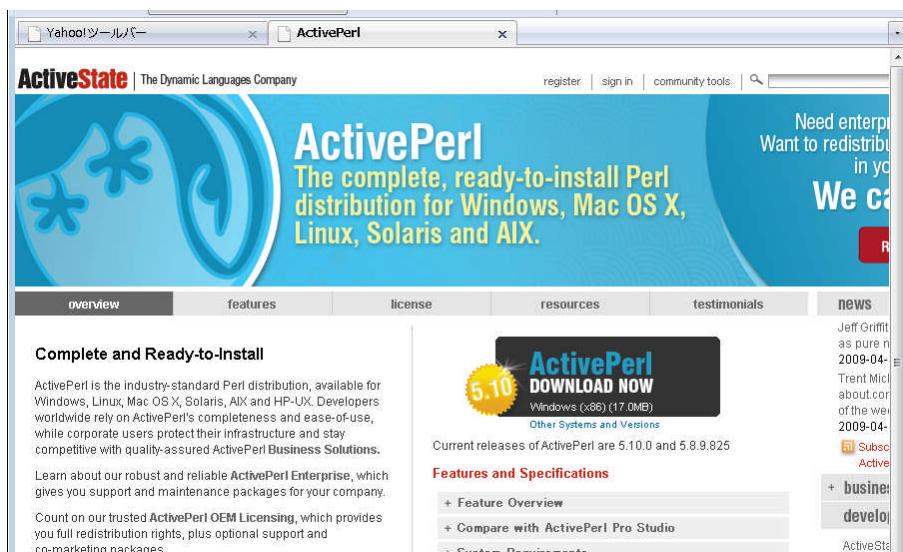
CPAN : Comprehensive Perl Archive Network,

<http://www.cpan.org>

Perl インストール

では早速 Perl をインストールします。今回は ActivePerl を用います。

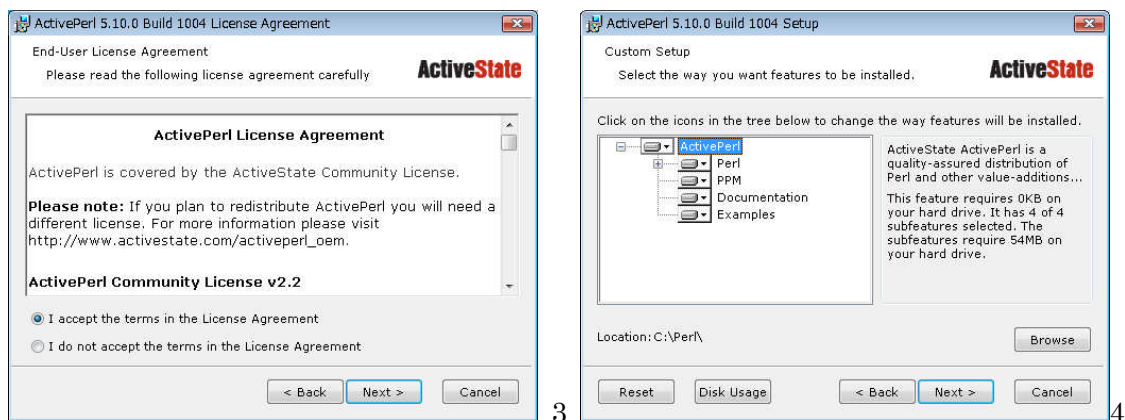
<http://www.activestate.com/activeperl/>



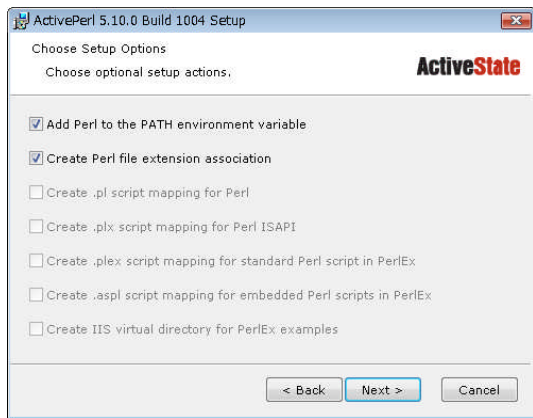
[ActivePerl / Download Now] をクリックします。



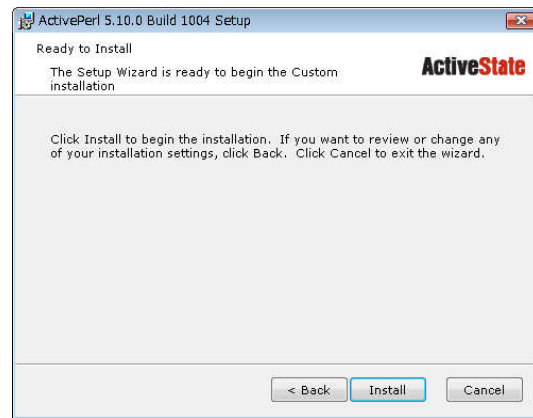
ファイル (~.msi) をダウンロードし、ダブルクリック。



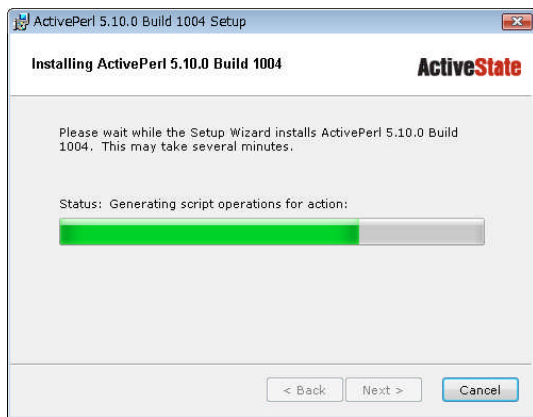
使用許諾は内容を確認し、問題なければ● I accept the ~を選び先に進みます。



5



6



7

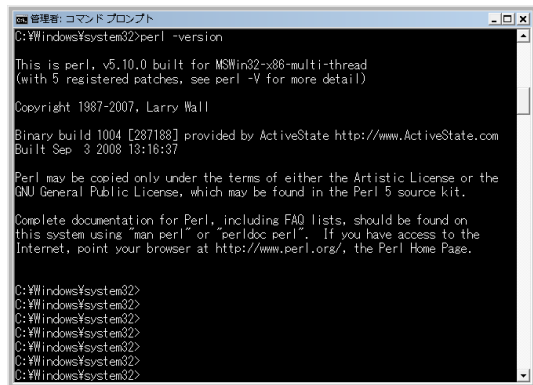


8

インストールが終了したら、コマンドプロンプトを起動し

```
C:\Documents and Settings> perl -version
```

とタイプして、以下のような内容が表示されればOKです。



サンプル CGI

では、再度 CGI の URL を指定してみてください。

<http://localhost/cgi-bin/printenv.pl>

```
COMSPEC="C:\Windows\system32\cmd.exe"
DOCUMENT_ROOT="C:/Apache2.2/htdocs"
GATEWAY_INTERFACE="CGI/1.1"
HTTP_ACCEPT="text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8"
HTTP_ACCEPT_CHARSET="Shift_JIS,utf-8;q=0.7,*;q=0.7"
HTTP_ACCEPT_ENCODING="gzip,deflate"
HTTP_ACCEPT_LANGUAGE="ja,en-us;q=0.7,en;q=0.3"
HTTP_CONNECTION="keep-alive"
HTTP_HOST="localhost"
HTTP_KEEP_ALIVE="300"
HTTP_USER_AGENT="Mozilla/5.0 (Windows; U; Windows NT 6.0; ja; rv:1.9.0.8) Gecko/2009032609 Firefox/3.0.8 (.NET
PATH="C:\Perl\site\bin;C:\Perl\bin;C:\app\student\product\11.1.0\db_1\bin;C:\Windows\system32;C:\Windows;C:\Win
PATHEXT=".COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC"
QUERY_STRING=""
REMOTE_ADDR="172.0.0.1"
REMOTE_PORT="50001"
REQUEST_METHOD="GET"
REQUEST_URI="/cgi-bin/printenv.pl"
SCRIPT_FILENAME="C:/Apache2.2/cgi-bin/printenv.pl"
SCRIPT_NAME="/cgi-bin/printenv.pl"
SERVER_ADDR="172.0.0.1"
SERVER_ADMIN="admin@si6.la.net"
SERVER_NAME="localhost"
SERVER_PORT="80"
SERVER_PROTOCOL="HTTP/1.1"
SERVER_SIGNATURE=""
SERVER_SOFTWARE="Apache/2.2.11 (Win32)"
SYSTEMROOT="C:\Windows"
WINDIR="C:\Windows"
```

今度は、エラーではなく（それよりも意味のわからない）記号の羅列が出てきたと思います。

これは CGI から呼び出された Perl が Web サーバーから引き継いだ情報の一覧です。等号の左が変数名、右がその値という形式になっています。主な変数の意味は以下の通りです。

変数名	意味	例
HTTP_USER_AGENT	見ているブラウザの種類	Mozilla/5.0
QUERY_STRING	引数	
REMOTE_ADDR	クライアントの IP アドレス	172.160.0.101
REQUEST_METHOD	HTTP の命令	GET, POST
SERVER_ADDR	サーバーの IP アドレス	172.160.0.230

QUERY_STRING は URL から CGI プログラムに何か値を指定すると、引き継がれます。たとえば

<http://172.16.0.230/cgi-bin/printenv.pl?hoge>

というように、最後に?と文字列（英数）を入れれば、QUERY_STRING="hoge"となります。

URL は日本語が扱えないので、漢字をいれると URL-Encode と呼ばれる方法で数字に変換されてしまいます。

?"こんにちわ" → %22%82%B1%82%F1%82%C9%82%BF%82%ED%22

また CGI プログラムは、Web サーバーが呼び出す外部プログラムです。つまり、コマンドプロンプトから直接実行することもできます。

先のコマンドプロンプトで

```
> cd %Apache2.2%cgi-bin
> perl printenv.pl
```

と実行すると、先ほどの画面に似た結果が表示されるはずです。

```
Content-type: text/plain; charset=iso-8859-1

ALLUSERSPROFILE="C:\Documents and Settings\All Users"
APPDATA="C:\Documents and Settings\WORLD BIZNet\Application Data"
COMMONPROGRAMFILES="C:\Program Files\Common Files"
COMPUTERNAME="WORLD-OD7015EC6"
COMSPEC="C:\WINDOWS\system32\cmd.exe"
FP_NO_HOST_CHECK="NO"
HOMEDRIVE="C:"
HOMEPATH="\Documents and Settings\WORLD BIZNet"
LOGONSERVER=\\WORLD-OD7015EC6
:
```

よく見ると、先の Web の実行結果とは違うはずです。QUERY_STRING とか「主な変数」は表示されていません。これは実行しているのが皆さんで、しかもコマンドプロンプトから呼び出されているからです。Web ブラウザーで見た結果は、Web サーバが CGI から printenv.pl コマンドを呼び出したからです。

また、最初の 2 行(空白行を含む)も違います。これがこのコンテンツの属性を表すヘッダ情報です。

Perl の簡単な解説(printenv.pl)

このサンプルプログラムをメモ帳で開くと、下記のようにになっています(左の数字は解説のために付けた行番号で、実際には存在しません)。

```
1  #! "C:\perl\bin\perl.exe"
2  ##
3  ## printenv -- demo CGI program which just prints its environment
4  ##
5
6  print "Content-type: text/plain; charset=iso-8859-1\n\n";
7  foreach $var (sort(keys(%ENV))) {
8      $val = $ENV{$var};
9      $val =~ s|\n|\\n|g;
10     $val =~ s|"|\\\\"|g;
11     print "$var=%"$val"%\n";
12 }
```

1 行目は、Perl プログラム本体の格納場所を示しています。インストール時に特に指定していなければ、このようになります。

2~3 行目はコメントです。# は、それ以降行末までコメントとして無視されます。

6 行目はヘッダ情報を出しています。\\n は改行を示していて、この行の最後に 2 つ連続していることから、Content~ という行と空行の 2 行が表示されます。

また perl は文の終わりはセミコロン(;)で終わるルールになっています。

7 行目から 12 行目までは繰り返しで、環境変数 %ENV を引き出しています。この辺は難しいので、決まり文句として軽く流してください。

8行目は \$vel という変数に環境変数の1番目の要素を代入しています。

9~10行目は特殊文字を間違って Web に解釈されないよう、ちよいとしたおまじないをしています。

11行目でやっと、「変数名=その値」を表示しています。

簡単な Perl の改造(printenv2.pl)

さきのサンプルプログラムをコピーし、printenv2.pl を用意します。

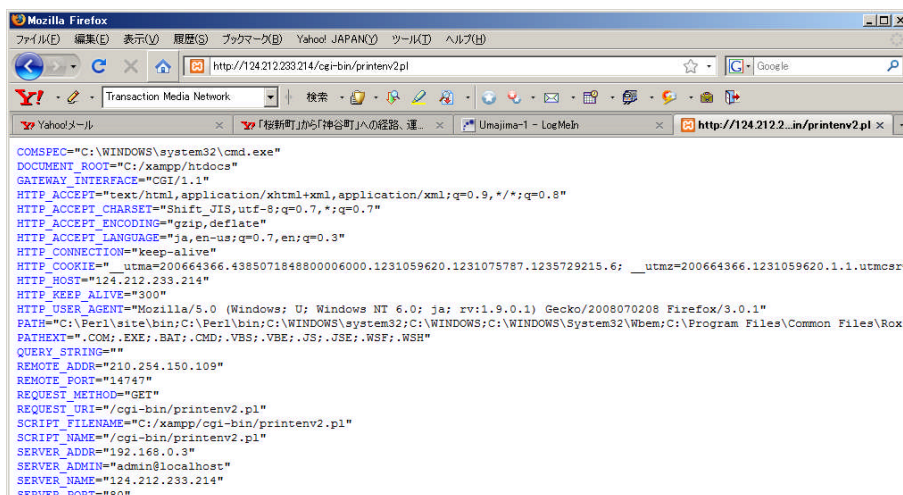
そして printenv2.pl をメモ帳で開き、下記のように少し修正します。

```
1  #!"C:¥perl¥bin¥perl.exe"
2  ##
3  ## printenv -- demo CGI program which just prints its environment
4  ##
5
6  print "Content-type: text/html; charset=iso-8859-1¥n¥n<pre>";
7  foreach $var (sort(keys(%ENV))) {
8      $val = $ENV{$var};
9      $val =~ s|¥n|¥¥n|g;
10     $val =~ s|"|¥¥"|g;
11     print "<font color=blue>${var}</font>=¥"${val}¥¥n";
12 }
```

6行目の中ほど、text/plain を text/html にし、末尾の ¥n の直後に <pre>を追加します。

11行目は print の最初のダブルクォーテーションの直後に を追加、更に、{var}=の等号の直前に を追加します。

このファイルができれば、<http://localhost/cgi-bin/printenv2.pl> を実行します。最初のサンプルより、もう少し見やすくなったでしょう。



6行目の中ほど、text/html を text/plain に戻して実行すると、先ほど青く表示された変数名は、妙なキーワードで囲まれているはずです。

```
<pre><font color=blue>COMSPEC</font>="C:\WINDOWS\system32\cmd.exe"
<font color=blue>DOCUMENT_ROOT</font>="C:/xampp/htdocs"
<font color=blue>GATEWAY_INTERFACE</font>="CGI/1.1"
<font color=blue>HTTP_ACCEPT</font>="text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8"
<font color=blue>HTTP_ACCEPT_CHARSET</font>="Shift_JIS,utf-8;q=0.7,*;q=0.7"
<font color=blue>HTTP_ACCEPT_ENCODING</font>="gzip,deflate"
<font color=blue>HTTP_ACCEPT_LANGUAGE</font>="ja,en-us;q=0.7,en;q=0.3"
<font color=blue>HTTP_CACHE_CONTROL</font>="max-age=0"
<font color=blue>HTTP_CONNECTION</font>="keep-alive"
<font color=blue>HTTP_COOKIE</font>="__utma=200664366.4385071848800006000.1231059620.1231075787.1235729215.6; __utmz=200664366.1231059620.1231075787.1235729215.6; __utmv=200664366.1231059620.1231075787.1235729215.6"
<font color=blue>HTTP_HOST</font>="124.212.233.214"
<font color=blue>HTTP_KEEP_ALIVE</font>="300"
<font color=blue>HTTP_USER_AGENT</font>="Mozilla/5.0 (Windows; U; Windows NT 6.0; ja; rv:1.9.0.1) Gecko/2008070208 Firefox/3.0"
<font color=blue>PATH</font>="C:\Perl\site\bin;C:\Perl\bin;C:\WINDOWS\system32;C:\WINDOWS;C:\WINDOWS\System32\Wbem;C:\Program Files\Internet Explorer\iexplore.exe"
<font color=blue>PATHEXT</font>=".COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH"
<font color=blue>QUERY_STRING</font>=""
<font color=blue>REMOTE_ADDR</font>="210.254.150.109"
<font color=blue>REMOTE_PORT</font>="7455"
```

これは、ヘッダの Content-Type の違いによるものです。

Content-Type: データの形式(タイプ) / 解釈方法(サブタイプ)

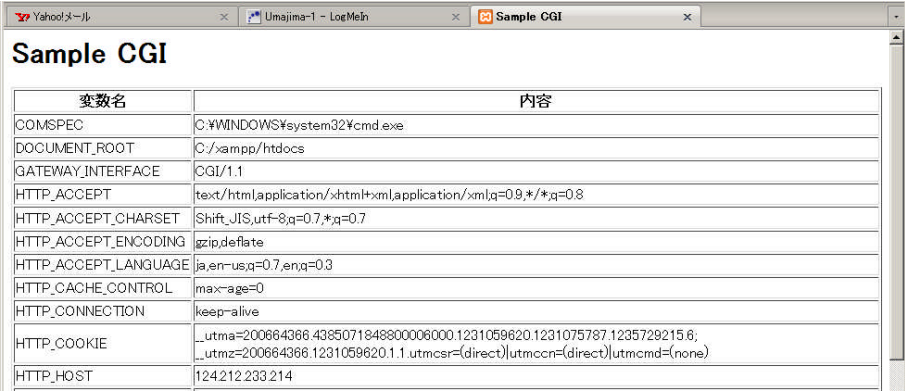
printenv.pl, printenv2.pl では、両方ともデータ形式はテキスト(文字だけの簡単なファイル)ですが、解釈方法が異なります。このヘッダ情報は、ブラウザの動作を指定するもので、受け取ったブラウザは同じデータでも「解釈方法」に従って、plain であれば、そのまま表示しますが、html であれば、HTML として文字の色や大きさ、枠といったアレンジを行って表示します。

表形式への改造(printenv3.pl)

もう少し複雑な表示を試してみます。再び `printenv2.pl` を `printenv3.pl` としてコピーして、`printenv3.pl` を以下のように修正します。

```
1  #!"C:¥perl¥bin¥perl.exe"
2  ##
5  ## printenv -- demo CGI program which just prints its environment
6  ##
7
8  print<<END1;
9  Content-type: text/html; charset=utf-8
10
11 <html><head><title>Sample CGI</title></head>
12 <body>
13 <h1>Sample CGI</h1>
14 <table border=1>
15 <tr><th>変数名</th><th>内容</th></tr>
16 END1
17
18 foreach $var (sort(keys(%ENV))) {
19     $val = $ENV{$var};
20     $val =~ s|¥n|¥¥n|g;
21     $val =~ s|¥¥|¥¥"|g;
22     print "<tr><td>${var}</td><td>${val}</td></tr>";
23 }
24
25 print<<END2;
26 </table>
27 </body>
28 </html>
29 END2
```

8~17行目は `print<<END1;` 直後(9行目)から、`END1` までの行を、そのまま表示します。`END1` は終わりの目印で、英数字ならば何でもかまいません。こんかいは `END1` としました。同様に複数行をそのまま表示する 25~29行では `END2` としています。完成すると、以下のような表形式になります。



変数名	内容
COMSPEC	C:\WINDOWS\system32\cmd.exe
DOCUMENT_ROOT	C:/xampp/htdocs
GATEWAY_INTERFACE	CGI/1.1
HTTP_ACCEPT	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
HTTP_ACCEPT_CHARSET	Shift_JIS,utf-8;q=0.7,*;q=0.7
HTTP_ACCEPT_ENCODING	gzip,deflate
HTTP_ACCEPT_LANGUAGE	ja,en-us;q=0.7,en;q=0.3
HTTP_CACHE_CONTROL	max-age=0
HTTP_CONNECTION	keep-alive
HTTP_COOKIE	__utma=200664366.4385071848800006000.1231059620.1231075787.1235729215.6; __utmz=200664366.1231059620.1.1.utmcsr=(direct) utmccn=(direct) utmcmd=(none)
HTTP_HOST	124.212.233.214

このとき、もし時化けを起こしてしまった場合は、9行目を見直してみてください。

```
Content-type: text/html; charset=sjjs
```

`charset` はこのコンテンツで使う文字コードを表します。ISO-8859-1 になっていません

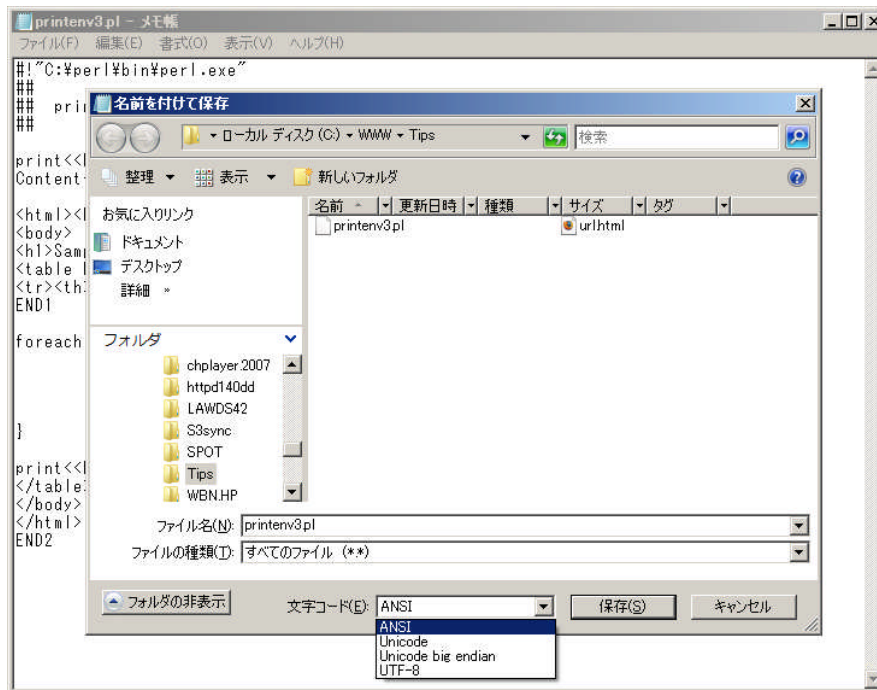
か？

文字コードには英語、フランス語といった言語によって割り当てられていますが、日本語ではさらに歴史的経緯から4つの文字コードが存在します。

charset	文字コード名(通称)	解説
ISO-8859-1	ラテン文字 (ASCII)	いわゆる英語圏で用いられているコードで、コンピュータの世界ではほぼ標準的に用いられています。英語、ドイツ語、フランス語など西欧諸国の文字を含みます。
Shift_JIS (sjis)	シフト JIS	マイクロソフト社が開発したコード体系で、MS-DOS、Windows XP で用いられています。
EUC-JP (eucjp)	拡張 UNIX コード	古いタイプの商用 UNIX (Solaris, HP-UX, AIX, Linux 2.2 以前) で用いられていました。
UTF-8	UNICODE	言語を問わず利用できる国際標準。Java, Linux 2.6 で採用されています。
ISO-8859-1	JIS コード	古いコンピュータで利用されていたコード体系。今では転送時のメール本文などに用いられ、直接ユーザが扱う事はめったにありません。

以前よくあったホームページの文字化とうのは、コンテンツに、この文字コードを明確に指定していなかったため、受け取ったブラウザ各自が勝手に判断していた事に起因していました。

また Windows Vista では、UNICODE を使う事を前提にしていますので、それ以前の PC では sjis を使うとよいでしょう。また XP 移行のメモ帳では文字コードを選ぶことができます。



「文字コード」にANSIを指定すると、シフトJISで保存することができます。また新規にCGIファイルを作成するときは、「ファイルの種類」には「すべてのファイル(*.*)」を選択します。

HTML

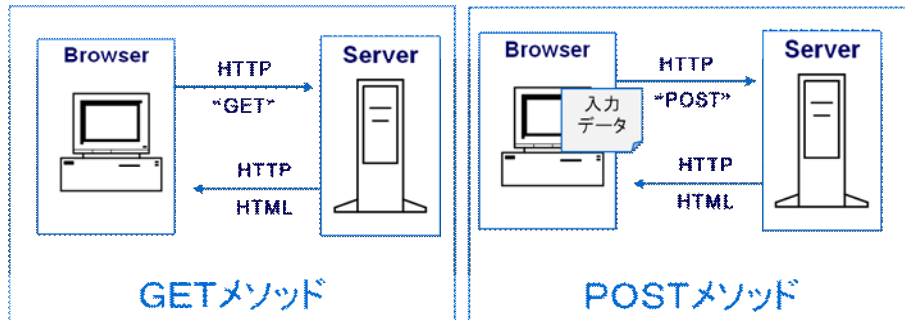
Perlの機能やプログラミングの話とは別に、ブラウザでコンテンツを表現するための手法をHTML(HyperText Markup Language)と呼んでいます。CGIを作るには多かれ少なかれHTMLの知識が必要です。そこで、printenv3.plで利用したHTMLを解説します。

タグ	意味
<html>~</html>	コンテンツの開始と終了
<head>~</head>	ヘッダ情報の開始と終了。ヘッダ情報にはタイトルや検索エンジンで参照されるキーワードなどが格納できます。
<title>~</title>	タイトル。ブラウザのタブやウインド枠に表示されます。
<body>~</body>	本文の開始と終了
<h1>~</h1>	表題。大きさやフォントにバリエーションがあり、h1, h2, h3.. と続く
<table>~</table>	表の開始と終了
<tr>~</tr>	テーブル行の開始と終了
<th>~</th>	項目名列(センタリングされる)の開始と終了
<td>~</td>	列の開始と終了

少し高度なCGI

printenv.pl は URL を呼び出すと、情報が表示されるだけの CGI でした。多くの企業のホームページなどではアンケートなどのように、ユーザーが入力したデータをサーバーが受け取って、CGI プログラム処理しています。

この様にユーザーの入力を行わせて、CGI で処理を行うには HTTP の POST と呼ばれる機能を使います。



ブラウザからサーバに依頼する仕事内容をメソッドと呼びます。普段ページを表示するには URL をサーバにつたえ、その情報を取得する「GET メソッド」が用いられています。URL だけでなく、関連する入力データを合わせてサーバに送り、それを元に CGI プログラムの処理結果を返してもらうのは「POST メソッド」と呼んでいます。

FORM 文

HTML に入力項目を設けるには、FORM タグを uses。まず以下の sampleform.html を用意します。(用意する場所は、C:\¥Apache2.2¥htdocs の下になります)

sampleform.html

```
<html>
<head><title>Perl sample</title></html>
<body>
<h1>Perl sample</h1>
<form action=/cgi-bin/sampleform.pl method=POST>
Your name:
<input name=name type=text size=20>
<p>
<input type=submit value="PUSH">
</form>
</body>
```

サンプル CGI(sampleform.pl)

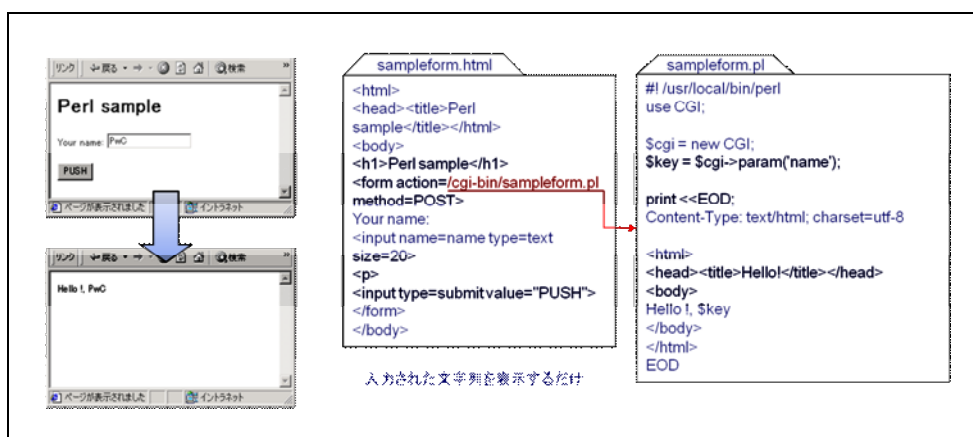
sampleform.html から form タグの action によって指定された sample.form.pl を今度は CGI の置場である C:\Apache2.2\cgi-bin に作ります。

```
#!/usr/local/bin/perl
use CGI;

$cgi = new CGI;
$key = $cgi->param('name');

print <<EOD;
Content-Type: text/html; charset=utf-8

<html>
<head><title>Hello!</title></head>
<body>
Hello !, $key
</body>
</html>
EOD
```



このように HTML と CGI を組み合わせて、いろんなサイトが構築されています。

後片付け

インストールしたソフト群を削除してください！

おまけ： J a v a S c r i p t のサンプル

以下の JavaScript を埋め込んだ HTML は、予め用意した写真ファイル（この例では Windows XP のピクチャーとしてサンプル提供されているもの）の特定の位置にマウスが近付くと背景色が変わります。

```
<html>
<head><title>Java Script sample</title></head>
<body>
<SCRIPT LANGUAGE="JavaScript">
<!--
        function bgRed() {document.bgColor=' Red' }
        function bgBlue() {document.bgColor=' blue' }
//-->
</SCRIPT>

<h1>onMouseOver</h1>
<center>

</center>
<map name="imageMap">
    <Area name="areaPoint1" coords="150,150 150" shape="circle"
        href="javascript:alert(' Red' )" onMouseOver=bgRed() >
    <Area name="areaPoint2" coords="450,450 150" shape="circle"
        href="javascript:alert(' Blue' )" onMouseOver=bgBlue() >
</MAP>
</body>
</html>
```