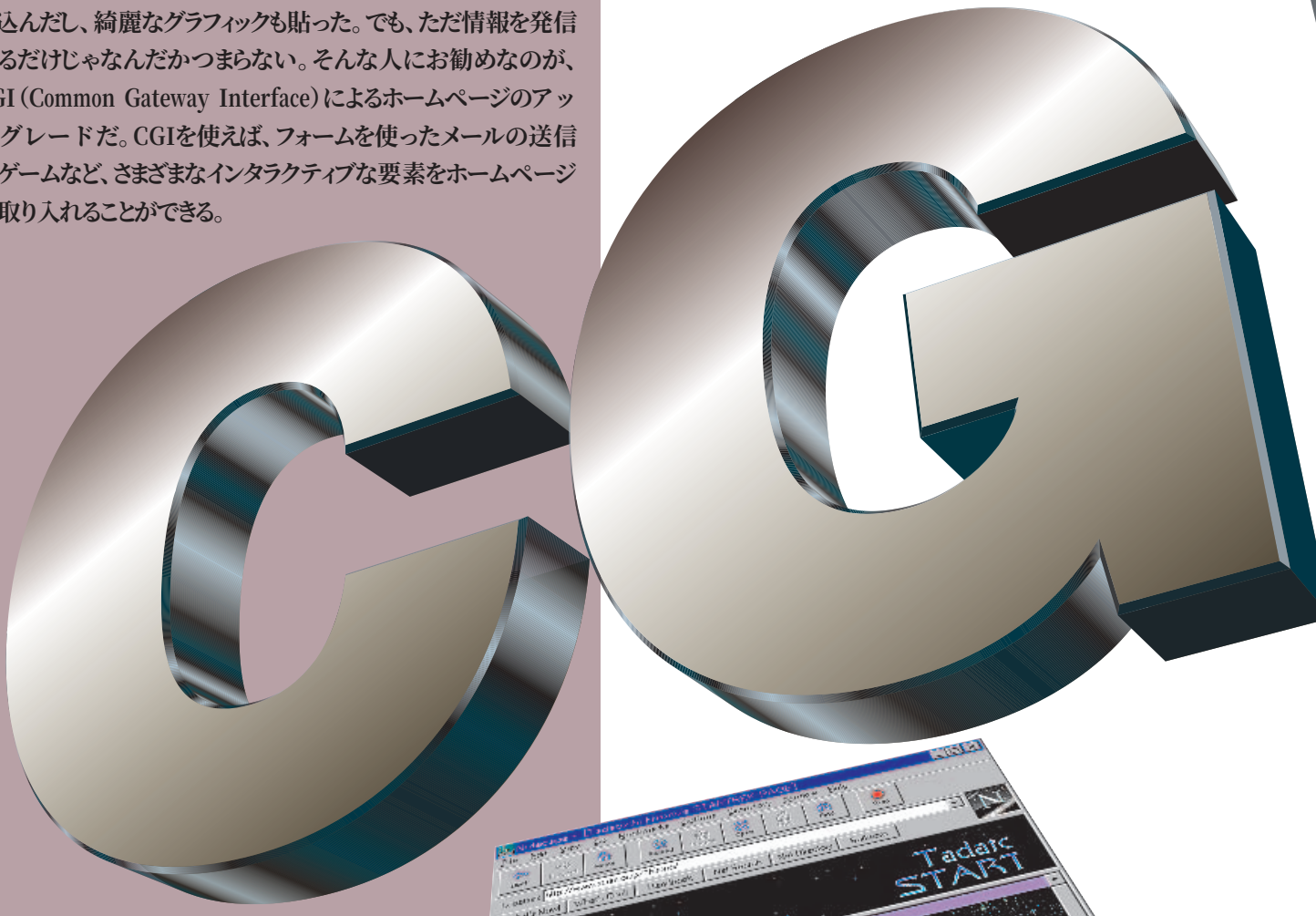
Common Gateway Interface

集中企画

ホームページ作成の上級テクニック

CGIの使い方 +SSI

ホームページを立ち上げたけど、どうも物足りない。情報も盛り込んだし、綺麗なグラフィックも貼った。でも、ただ情報を発信するだけじゃなんだかつまらない。そんな人にお勧めなのが、CGI（Common Gateway Interface）によるホームページのアップグレードだ。CGIを使えば、フォームを使ったメールの送信やゲームなど、さまざまなインタラクティブな要素をホームページに取り入れることができる。



ひろのただとし
広野忠敏
URL http://www.st.rim.or.jp/~hirono/

* UNIXで使われているプログラム言語の1つ。フリーソフトとして入手できる。さまざまな処理に対して柔軟に対応できるので、ホームページでインタラクティブな処理を行うCGIを記述するのに適している（本誌付録CD-ROMに収録）。

（参考文献）

- 「はじめてのPerl」Randal L. Schwartz著（ソフトバンク）
- 「Perlプログラミング」Larry Wall/Randal L. Schwartz著（ソフトバンク）

CGIってなんだろう

HTMLとCGIによる情報発信の違い

ふつうホームページはHTMLを使って書く。ブラウザーからページを見るときは、Webサーバー（httpd）にアクセスして、サーバーにURLを送るとWebサーバーが該当するページのHTMLファイルやイメージファイルをブラウザーに送信してくれる。受け取ったファイルを解釈して画面に表示するのがブラウザーの役割だ（図1）。つまり、HTMLだけでホームページを作ると、作っておいた内容以上のものを表示することはできない。ということは、静的な情報の発信しかできないわけだ。

これに一石を投じてくれるのがCGIだ。CGIはCommon Gateway Interfaceの略で、Webサーバーとサーバー上で動く他のプログラムやスクリプトとのインターフェイスの役割を果たす。通常のHTMLではHTMLに書かれた以上の情報をブラウザーに送信することはできないが、CGIを使えば、プログラムやスクリプトで生成したHTMLファイルなどをブラウザーに送ることができるようになる。つまり、CGIはHTMLでは不可能な複雑な仕事をすることのできる非常に強力な機能を持っているのだ。たとえば、CGIではブラウザーで対話的な処理を行う場合（データベースの検索のためのキー項目の入力など）や、ページを表示するときにあらかじめ何かを行わせておきたいとき（訪問者カウンターの生成など）のように、他のプログラムに処理を行わせて、その実行結果をブラウザーに表示したいときに利用することができる（図2）。

ところで、CGIを利用してインタラクティブなページを作るにはサーバーの外で何かを実行するためのプログラムやスクリプトを作らなければならない。したがって、これらのプログラミング言語一般に関する知識や、Webサーバーに関する知識、Webサーバーが設置されているプラットフォームの知識などが必要になることをあらかじめお断りしておく。

なお、今回の解説はWebサーバーとして最もポピュラーなNCSA httpdとサーバー環境として最も利用されているUNIX、そしてプログラムではPerl*とシェルスクリプトを前提に進めることにする。NCSA以外のサーバーでもCGIを使うことができるが、設定が若干異なったり、そのサーバー固有のルールがあることも多いので注意してほしい。

NCSA httpdに関しては、開発元のNCSAのサーバーに詳しい情報（英文）があるので参考になるだろう（図3）。

図3：NCSA httpdのCGIに関する情報ページ

URL http://hoohoo.ncsa.uiuc.edu/cgi/

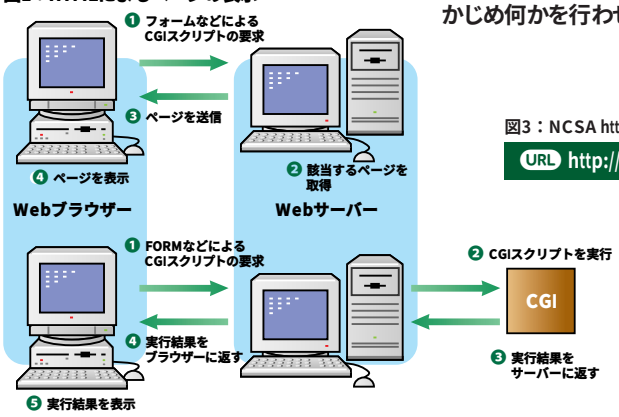


図1：HTMLによるページの表示

図2：CGI処理の流れ

CGI

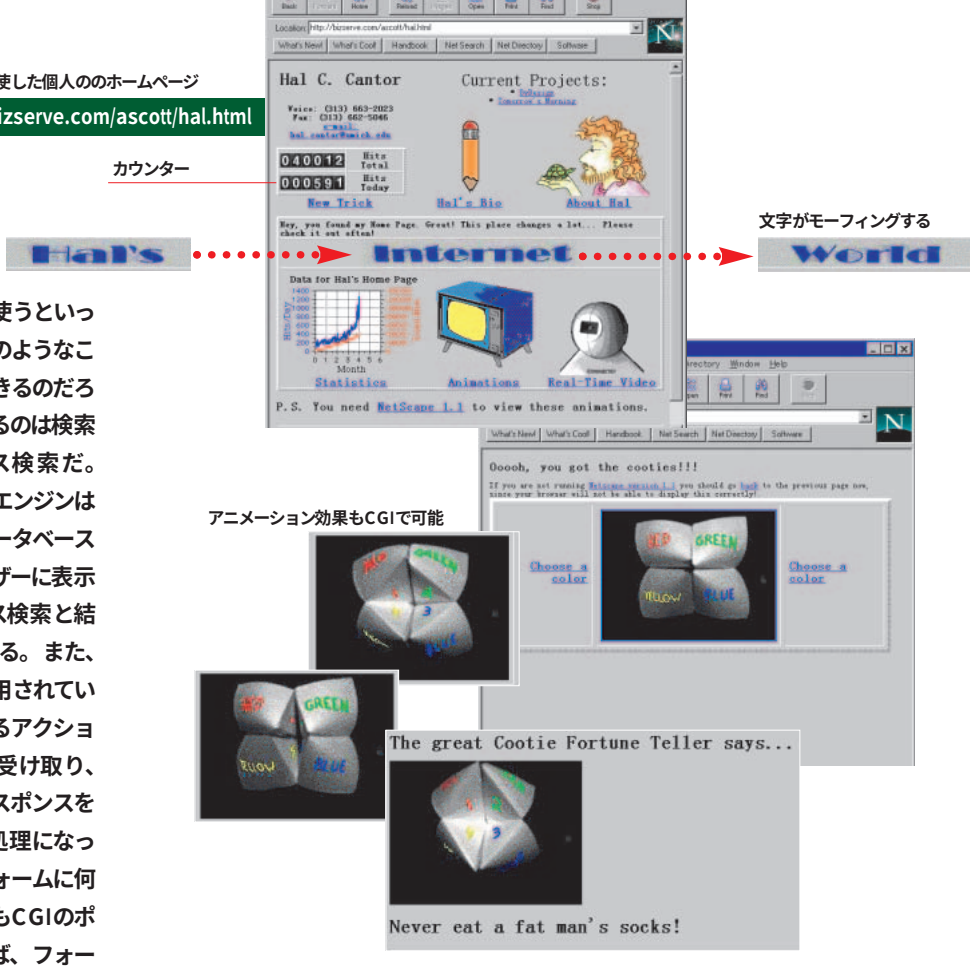
Common Gateway Interface

CGIでは こんなことができる

CGIを使うといったいどのようなことができるのだろう。CGIが最も利用されているのは検索エンジンなどのデータベース検索だ。YahooやLycosといった検索エンジンはユーザーが入力した語句でデータベースを検索し、その結果をブラウザーに表示する。このときのデータベース検索と結果の表示にCGIが使われている。また、Web上のゲームでもCGIが利用されている。ゲームではゲームに関するアクションや情報をCGIプログラムが受け取り、そのアクションに対してのレスポンスをCGIプログラムで返すという処理になっていることが多い。また、フォームに何か入力してそれを処理するのもCGIのポピュラーな使い方だ。たとえば、フォームに入力された内容のメール送信、訪問者リスト、アンケートの収集などといったものにCGIプログラムが使われている。

図4：CGIを駆使した個人ののホームページ

URL http://bizserve.com/ascott/hal.html



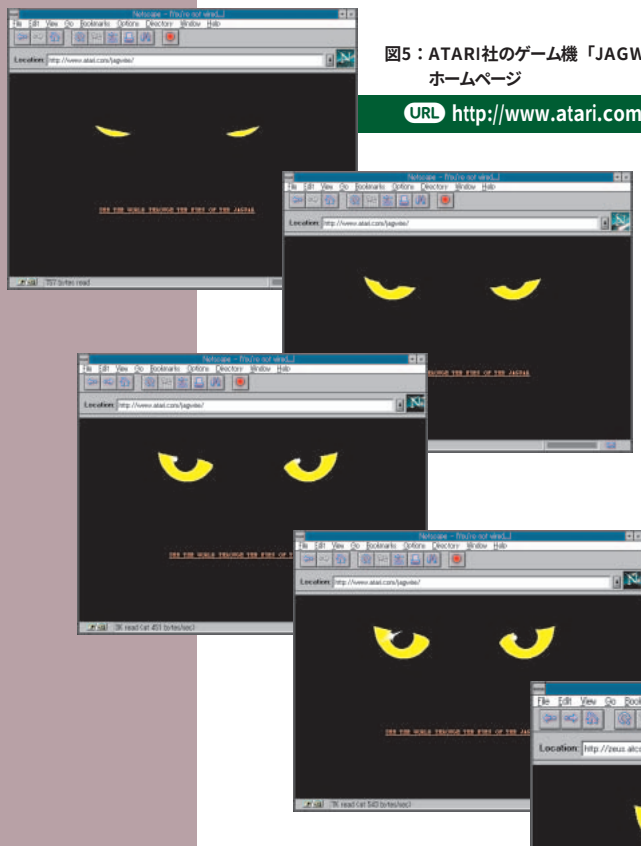


図5：ATARI社のゲーム機「JAGWIRE」のホームページ

URL http://www.atari.com/jagwire/

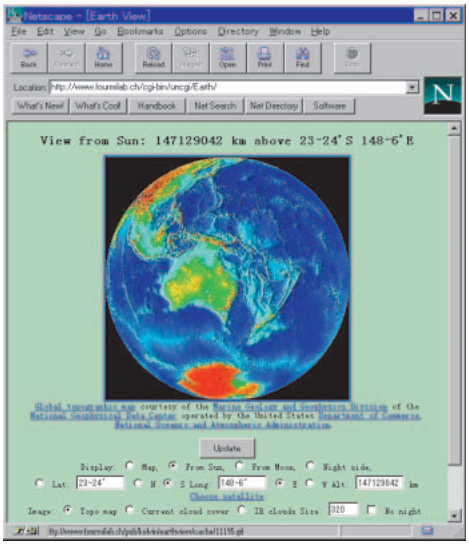


図6：フォームから入力した値に基づいて画像が作られる

URL http://www.fourmilab.ch/cgi-bin/uncgi/Earth/

図7：検索エンジンにもCGIが利用されている

URL http://www.yahoo.com/

使用サーバー	NCSA httpd	CERN httpd	Netscape他
CGI利用可（セキュリティチェックなどがある場合もあり）	InfoNOC、虹ーネット、ラボネット、AIANetインターネットサービス、RIMNET、ネットラピュタ、彩ネット、CYBORG、インターネット浜松、Info RYOMA、両毛インターネット、インターリンク、GulfNet、インターネットアクセスセンター、インターネット互助会横浜（CERNも使用）	NMCインターネットサービス、マリモインターネット、アリネットインターネットサービス、INTER BROAD、まほろば、avisインターネット、ネスク・インターネット・サービス、オレンジインターネットサービス、INGインターネット	CISネット、ミコネット、BEKKOAME/INTERNET、BAOBABNET、biwaネット、スリーウェブ、ilinx、Info Sphere、インターサーブ
条件付きで利用可	KuMaGaYaNet（実験として）、CJNインターネット（自作は不可）、栃木インターネット（協力グループのみ）、丸紅インターネットサービス（個別対応）	incl（アクセスカウンターのみ）、THE FSINetwork（別途見積り）、NewCOARA（システム管理者が許可したもの）、M Birdインターネットサービス（法人対象）	INTERCITY（法人ユーザーのみ）、α-Web（オーダー物件のみ）、netSpace（コンテンツサービス利用の法人）、Butaman（法人のみ）、at-mインターネット（インストール済のものを利用）、tcp-ip（会社で用意したものを利用）
利用不可	HITインターネットサービス、晴れの国ネット（汎用的なCGIの利用を検討中）、ガリバーインターネット（予定あり）、スペック、GNOC（準備中）	しかせんべいインターネット、Aoi-net、インターネットビークル、メディアマウンテン、ガンセキ・インターネット・サービス、SPWINTERNET、SeaFolk（予定あり）	InfoValley、IFNET、SKJ、PICNET（検討中）

表1：ホームページを持てるプロバイダーのCGI対応状況（1995年12月25～1996年1月10日編集部調査）

CGIを使う前に

使用するサーバーを確認する　CGIを使う前には、まずCGIを使おうとするサーバーがCGIを使える状態になっているのかどうかを確認する必要がある。そのサーバーの管理者に問い合わせてみよう。多くのWebサーバーではデフォルトの設定ではCGIが使えないようになっている。ユーザーレベルで自由にCGIが使えると、それによってセキュリティーに穴が生じるケースもあるので、プロバイダーなどのサーバーではCGIを禁止しているところも多い。

なお、WebサーバーでCGIを使用可能にする方法については、Webサーバーによって細かい設定方法が違うため、ここでは触れないことにする。

プロバイダーのWWWサーバーをチェック　CGIを使う場合、自分でサーバーを立ち上げたり、大学や会社のサーバー環境をそのまま利用できるのでなければ、最も手っ取り早い方法は、UNIXのシェルをユーザーに開放している商用プロバイダーのホームページ設置サービスを利用することだ。ただし、各プロバイダーでCGIへの対応はさまざまなので注意してほしい。たとえば、特に手続きもなく自由に利用できるところでは、もともとサポートなしでUNIXシェルを操作できるユーザーが多く、自力でなんとかするという気構えが必要だ。また、システム管理者宛ての連絡が必要なところ、個別に対応するところ、事務局でご用意したCGIのみ利用を許可するところなどもある（表1）。

CGIの利用に関する情報をWeb上で提供しているプロバイダーもあるが（表2）、ニュースグループやメーリングリストでユーザー同士が情報交換を行っているところも多い。

現在のところCGIの利用はたいてい基本料金に含まれているが、利用が増えてくるとサーバーを切り分ける形で有料化を検討しているところもある。

その他のサーバーでのCGI

プラットフォームがUNIXのWebサーバーはCGIを作るときに、UNIXの開発環境やPerlやawk、シェルスクリプトといったプログラミング環境が使えるためCGIを作るのも比較的簡単だ。また、シェルコマンドやさまざまなコマンドを組み合わせていろいろな処理をすることができる。ところで、UNIX以外のプラットフォームで動作するWebサーバーではどのような方法でCGIを実現するのだろうか。UNIX以外ではWindows NTのhttpsやMacintoshのMacHTTPなどが比較的よく利用されているWebサーバーだ。もちろん、これらのサーバーでもCGIを利用することができる。

Windows NT上で動作するhttpsはCGIとしてコンソールアプリケーションを呼び出すことができる。つまり、CGIプログラムはコンソールアプリケーションとして作成する。また、Perlなどのインタープリターを使うことも可能だ。さらに、最近のバージョンではVisual BasicなどのウィンドウアプリケーションをCGIで使うことが可能になっている。

MacHTTPの場合はAppleScriptでCGIプログラムを作成する。AppleScriptだけでもかなり高度なCGIを作ることもできるが、その他のプログラムやアプリケーションをAppleScriptでコントロールすることができるので、かなり高度な処理が可能になる。

表2：プロバイダーの提供するCGI情報

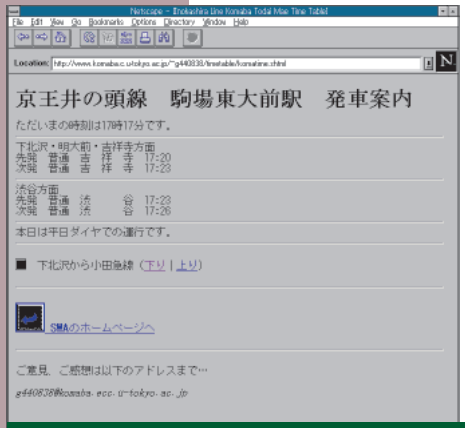
（プロバイダー名）	（参考情報）
BEKKOAME	http://www.bekkoame.or.jp/Users/homepageinfo.html
MAHOROBA	http://www.mahoroba.or.jp/uinfo/etc.html
BAOBABNET	http://www.linkage.co.jp/HTML/CGI/cgi1.shtml
インターリンク	http://home.interlink.or.jp/Prim/guide.html
オレンジインターネットサービス	http://www.orange.or.jp/orangecom/homepage/homepage.html

CGIを実行するためのセットアップ

従来のCGIプログラムはサーバーの特定のディレクトリーになければ使うことができなかったが、NCSA httpdではユーザーが宣言をすれば、ユーザーのディレクトリーにあるCGIプログラムを使うことができる（CERN httpdでは不可。ただし、cgi-wrapperなどを使えば同様のことが可能）。そのための宣言ファイルが.htaccessと呼ばれるファイルだ。この.htaccessファイルは通常はユーザーのホームページのホームディレクトリー（~/public_html）に置く。ユーザーのディレクトリーに置いたCGIプログラムを使用可能にするには、この.htaccessファイルに次の1行を追加する。

```
AddType application/x-httpd-cgi .cgi
```

この記述があると、拡張子が.cgiのファイルがCGIプログラムとして取り扱われるようになる。なお、サーバーでCGIの試用を禁止するように設定されているときは、たとえ.htaccessで設定を行ってもCGIを使うことはできない。



URL http://www.komaba.c.u-tokyo.ac.jp/~g440838/timetable/izumitime.shtml

図8：アクティブ時刻表
現在時刻を元に「次の電車」を表示する

CGIの基礎知識

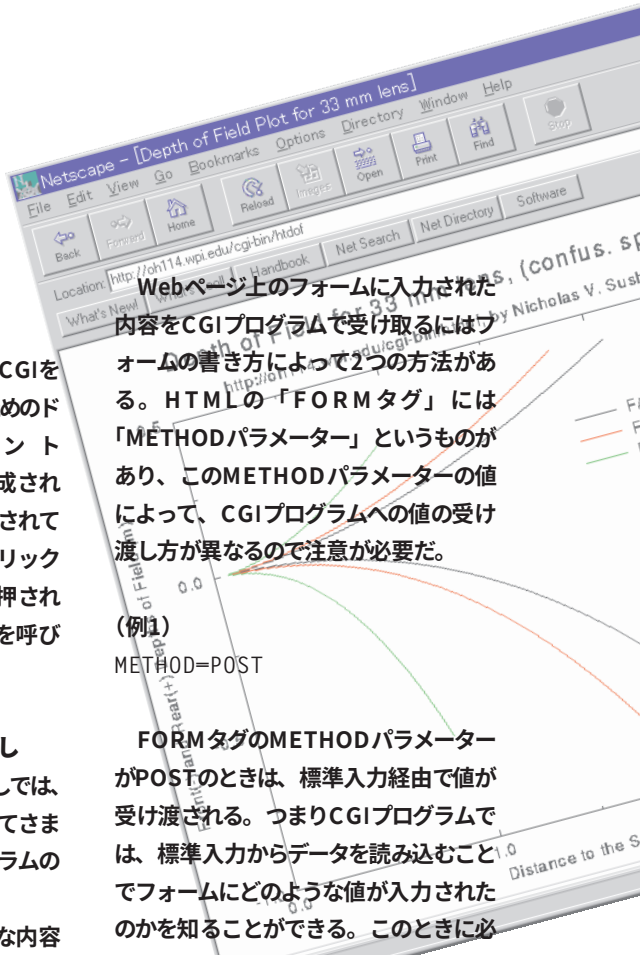
CGIの呼び出しとデータの受け渡し

CGIはそのCGIを呼び出すためのドキュメント（HTML）とCGIプログラムで構成される。つまり、ドキュメントが表示されている状態でアンカーポイントをクリックしたり、フォーム処理のボタンが押されたりしたときに、CGIプログラムを呼び出すことができるのである。

❶ フォームによるCGIの呼び出し

フォームを使ったCGIの呼び出しでは、フォームに入力された内容を使ってさまざまな処理をするのがCGIプログラムの仕事となる。

このとき、フォームにどのような内容が入力されたのかをCGIプログラム側で知る必要がある。そのために使われるのが「環境変数」と呼ばれる特殊な変数だ。Webサーバーでは、あらかじめいくつかの環境変数が設定されていて、その環境変数を参照すれば、フォームにどのような内容が入力されたのかをCGIプログラムで判断することができる。このために使われる環境変数がQUERY_STRING、REQUEST_METHOD、CONTENT_LENGTHなどである。これらの3つの環境変数の内容を確認すれば、どのような手順と内容でCGIプログラムに値が渡されるのかを知ることができる。



Webページ上のフォームに入力された内容をCGIプログラムで受け取るにはフォームの書き方によって2つの方法がある。HTMLの「FORMタグ」には「METHODパラメーター」というものがあり、このMETHODパラメーターの値によって、CGIプログラムへの値の受け渡し方が異なるので注意が必要だ。

（例1）

```
METHOD=POST
```

FORMタグのMETHODパラメーターがPOSTのときは、標準入力経由で値が受け渡される。つまりCGIプログラムでは、標準入力からデータを読み込むことでフォームにどのような値が入力されたのかを知ることができる。このときに必要なのがCONTENT_LENGTH環境変数だ。CONTENT_LENGTH環境変数には、データの長さ（バイト数）が格納されるので、必ず入力データの長さを受け取ってから、その長さだけ標準入力から値を読み込まなければならない（図9）。

（例2）

```
METHOD=GET
```

FORMタグのMETHODパラメーターがGETのとき、あるいはFORMタグにMETHODパラメーターがないときは、QUERY_STRING環境変数にフォーム

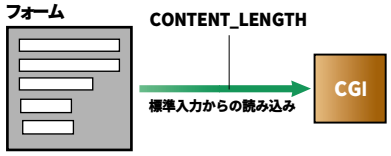


図9：METHODパラメーターが"POST"のとき

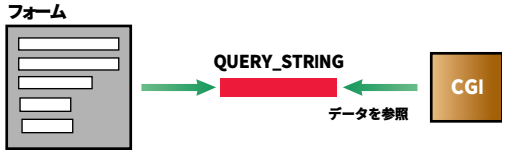
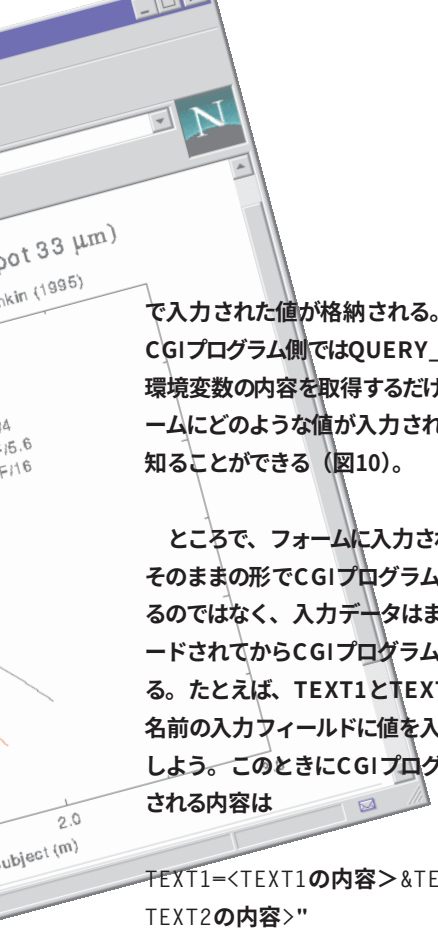


図10：METHODパラメーターが"GET"のとき



で入力された値が格納される。つまり、CGIプログラム側ではQUERY_STRING環境変数の内容を取得するだけで、フォームにどのような値が入力されるのかを知ることができる（図10）。

ところで、フォームに入力された値は、そのままの形でCGIプログラムに渡されるのではなく、入力データはまずエンコードされてからCGIプログラムに渡される。たとえば、TEXT1とTEXT2という名前の入力フィールドに値を入力したとしよう。このときにCGIプログラムに渡される内容は

```
TEXT1=<TEXT1の内容>&TEXT2=<TEXT2の内容>"
```

のように、それぞれの入力が"&"（アンパーサンド）で区切られて渡される。また、入力内容のうち英数字以外のキャラクター（日本語や記号など）は%83のように%nn（nnはキャラクタコード）の形にエンコードされる。つまり、どのような値が入力されたのかを知るには、入力データを受け取った後で、そのデータをデコードする必要が出てくる。

❷ アンカータグによるCGIの呼び出し

HTMLのアンカータグに別のページのURLを記述しておくと、そのアンカーがクリックされたらそこに記述されたURLにジャンプすることができる。通常はここに他のページのURLを記述するのだが、その代わりにCGIプログラムのURLを書いておけばアンカーをクリックしたときにURLで指定されたCGIプログラムを動かすことができる。

アンカーでCGIプログラムが呼ばれるときは、FORMタグのMETHODパラメーターがGETのときと同様に、QUERY_STRINGを使って値の受け渡しが行われる。そのため、

```
<A HREF="foo.cgi?arg1=val1&
arg2=val2">
```

のように、アンカーでCGIプログラムに値を受け渡すことも可能だ（図11）。

❸ イメージタグによるCGIの呼び出し

アンカータグと同様に、イメージタグにもCGIプログラムのURLを書くことができる。イメージタグに指定したCGIプログラムでGIFやXBMなどのイメージを生成すれば、そこにプログラムで生成したグラフィックイメージを表示することができる。これを応用すればCGIプログラムでグラフをプロットしたり、日替わりのイメージを表示したりすることも可能だ。また、グラフィック表示されているアクセスカウンターの多くはこの手法を使っている（図12）。

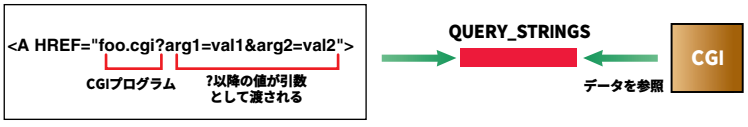


図11：アンカータグからCGIを呼び出す

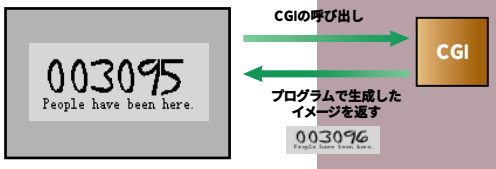


図12：イメージタグからCGIを呼び出す

JAVAとLiveScript

Netscape 2.0でもサポートし、最近にわかに注目を集めているのがJAVAやLiveScriptといった言語だ。CGIはサーバー側でプログラムを動かして、その結果をブラウザーに返す。そのため、重いプログラムなどをCGIとして動作させると、その分だけサーバーに負荷がかかることになる。ところが、JAVAやLiveScriptはサーバー側ではプログラムを動作させずに、プログラムのコードをブラウザーに転送し、ブラウザー側でプログラムを実行する。そのため、サーバーに負荷をかけずにインタラクティブな仕掛けを作ることができるのだ。

JAVAはC++言語に似た仕様を持ち、グラフィックの処理やテキスト処理のみならず、ソケットを使ったネットワークプログラミングまでカバーする極めて高度な言語だ。JAVAを使いこなすにはプログラミングに関しての豊富な知識が必要になるが、イメージをアニメーションさせたりゲームを作ったりと、実にさまざまなことを実現できる。今のところJAVAを解釈できるブラウザーはHotJavaとNetscape 2.0（β3以降）の2つ。さらに今後はさまざまなブラウザーでJAVAを解釈できるようになる。

LiveScriptはNetscape 2.0に搭載されているスクリプト言語だ。LiveScriptはイメージのハンドリングなど高度なことはまだできないが、フォームやボタンのハンドリングを簡単に行うことができる。ただ、まだ仕様が未確定な部分もあるため、今後に注目したい。LiveScriptもC++言語をもとにした言語仕様を採用しているが、JAVAよりは簡単にプログラムを作成することができるようになっている。

CGIプログラムを書いてみよう

ヘッダーを作る

HTMLによってCGIプログラムを呼び出す準備ができたら、次にCGIプログラムの本体を作成する。CGIプログラムはWebサーバーが動作しているプラットフォームで使える言語ならばどのような言語でも使うことができるが、UNIXの場合ではシェルスクリプト、Perl、Cコンパイラ、Tcl/Tkなどがよく使われている。ここでは、Perlを使ってCGIプログラムを作ることにしよう。

呼び出されたCGIプログラムはデータを標準出力に出力するが、そのデータは実際にブラウザーのウィンドウに表示される。このときに出力されるイメージの先頭には、ヘッダーと呼ばれる特別な行を付加しなければならない。ヘッダーは出力をどのように扱ったらよいのかをサーバーに指示する。このヘッダーにはContent-type、Location、Statusの3種類がある。なお、これらのヘッダーの後には、常に空行が必要になる。

```
Content-type:
```

Content-typeヘッダーはヘッダーに続く出力内容がどのようなものかをブラウザーに指示するためのヘッダーだ。たとえば、CGIがHTMLファイルを出力するときは、

```
Content-type: text/html
```

という内容のヘッダーを出力するようにする。また、GIFイメージを出力したいときは、

```
Content-type: image/gif
```

のようなヘッダーを生成すればよい。text/htmlやimage/gifはMIME形式のタイプで、これらのほかにもtext/plain、image/jpeg、video/mpegなどさまざまなタイプを指定することができる。ブラウザーでは、このタイプを解釈してブラウザー自身にデータを表示するのか、あるいはヘルパーアプリケーションでデータを再生するのかが決定される。

```
Location:
```

Locationヘッダーは CGIの出力をブラウザーに表示するのではなく、サーバー上の別のドキュメントをブラウザーに表示させるための特殊なヘッダーである。どのドキュメントを表示させるのかは、「Location:」の後にURLで指定する。たとえば、

```
Location: http://www.foo.bar/
```

の例では、CGIプログラムが実行されると、http://www.foo.bar/で指示されたHTMLファイルがブラウザーに表示される。このヘッダーを使ってランダムなURLを生成すれば、ランダムリンク機能をCGIで簡単に実装することができる。

```
Status:
```

Statusヘッダーはブラウザーに状態コードを返すために使われる。この状態コードはHTTPの仕様に定義されているもので、さまざまなコードを返すことができるのだが、あまり一般的ではないのでここでは解説を省略する。

簡単なCGIプログラムの例

ここで簡単なCGIプログラムを2つ紹介しよう。

❶ 日付と時間を表示する

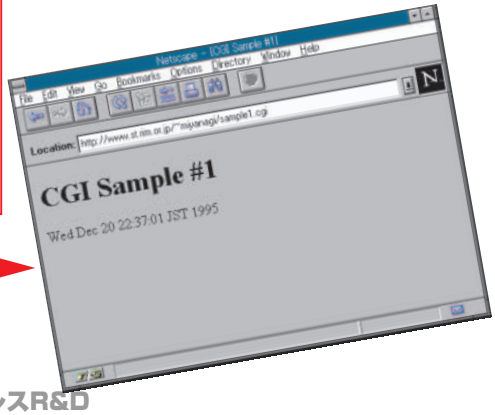
sample1.cgi（図13）は最も簡単なCGIの例である。ここではUNIXのdateコマンドを使って、現在の日付を表示する。このように、CGIを使うとUNIXのコマンドの出力結果をHTMLとして吐き出すことができるので、いろいろと応用できる。たとえば、Webサーバー経由で参照できるArchieやWhois、Fingerといったゲートウェイは、このようにCGIでUNIXコマンドを実行することによって実現されていることが多い。

図13：sample1.cgi
現在の日付を表示する

ヘッダーでHTMLファイルの出力を指定

```
#!/usr/local/bin/perl

print "Content-type: text/html\n\n";
print "<TITLE>CGI Sample #1</TITLE>\n";
print "<H1>CGI Sample #1</H1>\n";
$d = `date`;
print "$d\n";
```



❷ 時刻によってイメージを変化させる

変化するイメージ、たとえば、時刻で内容が変わるイメージや、アニメーションするイメージにもCGIが使われている。sample2.cgi（図14）は時刻によって異なるイメージを表示するためのCGIプログラムだ。この例では、参照した時刻が6時以前のときはmidnight.gifが、6時から9時まではmorning.gifが、9時から18時まではafternoon.gifが、それ以降のときはnight.gifが表示される。このCGIプログラムを、

```
<A IMG="sample2.cgi">
```

のようにイメージタグで指定すると、ページを見た時刻によって違うイメージのGIFファイルが表示される。また、これを応用すればランダムに変化するイメージなども作ることができる。

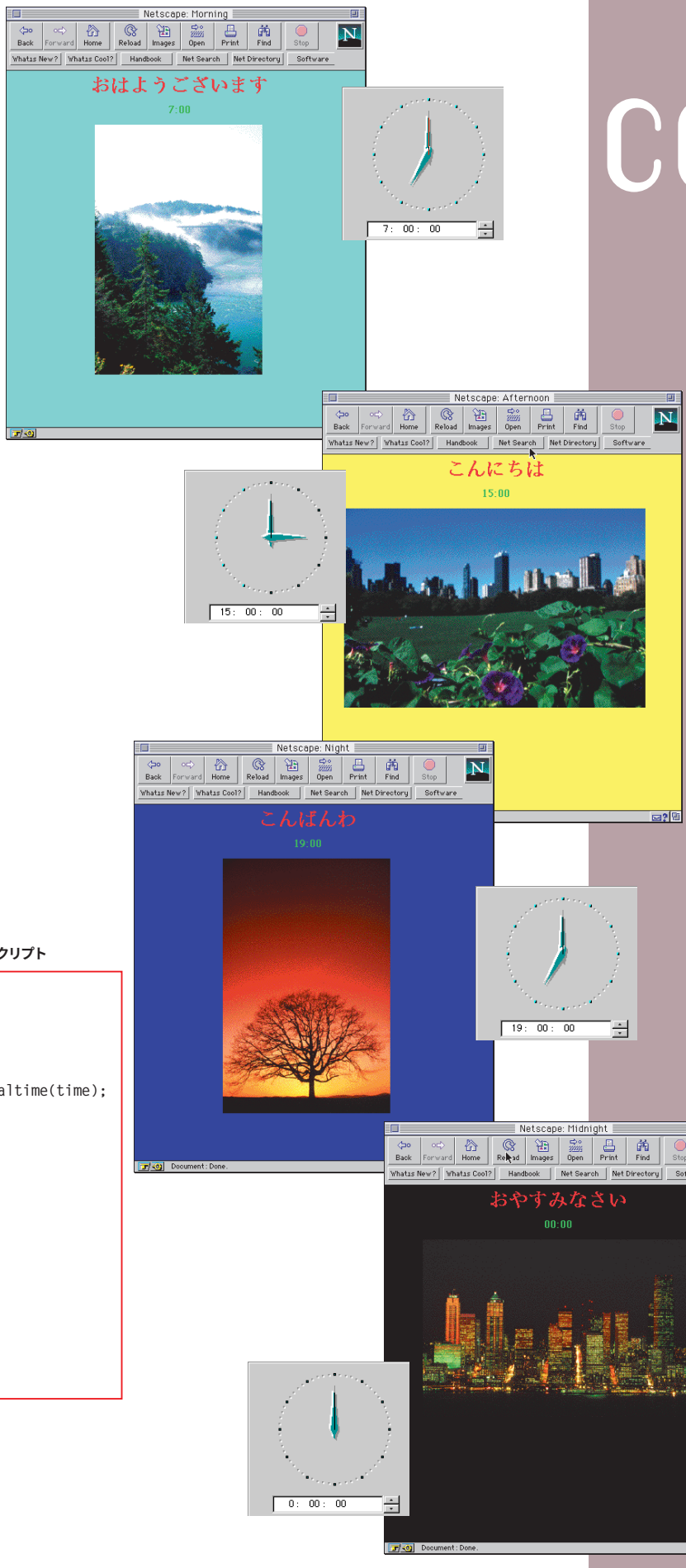


図14：sample2.cgi
時間によってホームページのイメージを変えるCGIスクリプト

```
#!/usr/local/bin/perl

print "Content-type: image/gif\n\n";

($sec,$min,$hour,$mday,$mon,$year) = localtime(time);

if ($hour < 6) {
    $img = 'midnight.gif';
} elsif ($hour < 9) {
    $img = 'morning.gif';
} elsif ($hour < 18) {
    $img = 'afternoon.gif';
} else {
    $img = 'night.gif';
}
$image = `/bin/cat $img`;
print "$image";
```

クリッカブルマップを作ってみよう

今やさまざまなところで使われているクリッカブルマップ。これは、マウスでイメージをクリックすると、クリックした場所によってジャンプ先が変わるものだ。実はこのクリッカブルマップもCGIで実現されている。ここではNCSA httpdのISMAP機能を使ったクリッカブルマップの作成方法を紹介しよう。

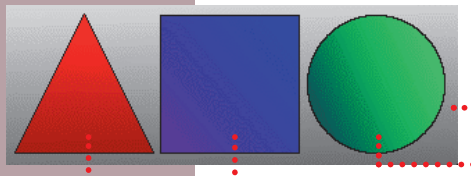
イメージタグのISMAP機能

NCSA httpdにはISMAPと呼ばれるクリッカブルマップを実現するための機能が用意されている。この機能は、イメージマップを指示するためのHTMLタグと、それによって呼び出されるCGIプログラム（imagemap）の2つで実現されている。このCGIプログラムは、NCSA httpdのパッケージに含まれているもので、通常Webサーバーをセットアップするときに、サーバーのcgi-binディレクトリーに置かれるはずだ。もし、imagemapプログラムがサーバーのcgi-binディレクトリーにないときは、サーバーの管理者にお願いして用意してもらうようにしよう。

ところで、このimagemapプログラムには大きく分けて2種類のバージョンが存在する。古いバージョンのimagemapではサーバーで管理するイメージマップ全体の設定ファイル（imagemap.conf）が必要で、このファイルは一般にユーザーが修正することができないようになっている。一方NCSA httpdのバージョン1.3以降に同梱されている新しいimagemapは、このような管理者用の設定ファイルを必要としない。つまり、ユーザーが自由に自分のディレクトリーにイメージマップを作れるようになっているのだ。ここでは、NCSA httpd 1.3以降に付属しているimagemapを使ってクリッカブルマップの作成方法を紹介しよう。なお、サーバーがNCSA httpdであるにもかかわらず、ここで紹介した方法が使えないときは、サーバーの管理者にお願いしてサーバーのバージョンアップをしてもらうか、新しいimagemapをインストールしてもらおう。また、NSCA以外のWebサーバーの場合は、まったく同じ方法は使えないが、似たような方法でクリッカブルマップを作ることができる。これらのサーバーを使っている人は、そのサーバーのマニュアルなどで確認してほしい。

イメージとmapファイルを作る

クリッカブルマップを作るには、まずクリッカブルマップの元になる画像ファイルと座標指定用のmapファイルを用意する（図15）。画像ファイルはブラウザーでインライン表示が可能な形式にする必要があるが、GIF形式のファイルならばたいていのブラウザーでインライン表示ができる。イメージファイルを用意したら、次にmapファ



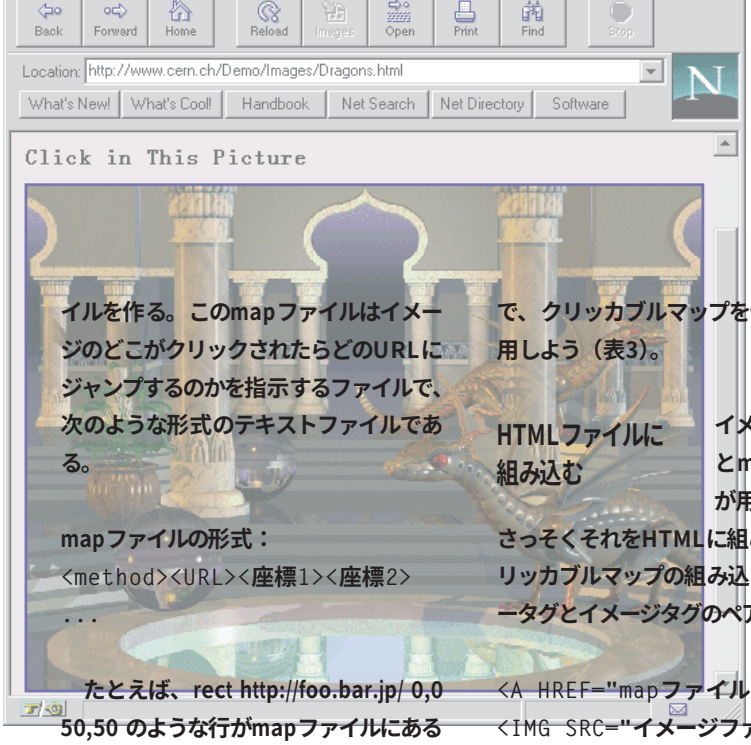
```
default /~foo/default.html
#
circle /~foo/circle.html 268,56 290,102
#
rect /~foo/rectangle.html 111,7 211,104
#
poly /~foo/poligon.html 57,6 5,106 107,106
```

<method>

default
デフォルトのURLを指定する。
circle
円の領域を指定する。座標は円の中心とエッジの座標を指定。
rect
四角形の領域を指定する。座標は四角形の左上と右下。
poly
多角形の領域を指定する。座標は多角形のそれぞれの座標を指定。
point
点を指定する。座標は点の座標。

図15：マップ上の座標を指定するmapファイル

図16：ブラウザーでクリッカブルマップを表示したところ



イルを作る。このmapファイルはイメージのどこがクリックされたらどのURLにジャンプするのかを指示するファイルで、次のような形式のテキストファイルである。

mapファイルの形式：

```
<method><URL><座標1><座標2>
...
```

たとえば、rect http://foo.bar.jp/ 0,0 50,50 のような行がmapファイルにあるときは、0,0～50,50の四角形の領域内がマウスでクリックされるとhttp://foo.bar.jp/にジャンプする。このように、イメージのどの領域がクリックされたらどこにジャンプするのかをmapファイルにどんどん書いていけばよい。defaultの指定はmapファイルに定義された領域のどこにも属さないところがクリックされたときの指定だ。

ちなみに、mapファイルの名前は、元になるイメージファイルの名前と同じベースネームを持つファイル名にしておくと後々管理が楽になる。仮にイメージのファイル名がclick_img.gifのときはmapファイルの名前はclick_img.mapにすればよいだろう。ところで、イメージの座標を確認しながらmapファイルを手作業で作るのはかなり大変な作業だ。クリッカブルマップの元になるイメージを表示しながら、領域をマウスで指定するだけで自動的にmapファイルを作成してくれるフリーソフトやシェアウエアもあるので、クリッカブルマップを作るときに活用しよう（表3）。

HTMLファイルに組み込む

イメージファイルとmapファイルが用意できたら、さっそくそれをHTMLに組み込もう。クリッカブルマップの組み込みは、アンカータグとイメージタグのペアで行われる。

```
<A HREF="mapファイルの指定">
<IMG SRC="イメージファイル"
ISMAP>
</A>
```

このときにイメージタグには必ずISMAPパラメーターを付けなければならない。ISMAPを付けるのを忘れてしまうと、イメージが普通のアンカーになってしまう。

さて、アンカーに指定するmapファイルの指定だが、ここはmapファイルのURLを直接書くのではないので気をつけよう。ここには、imagemapプログラムのURLとその後に続けてmapファイルのURLを書くようにする。たとえば、クリッカブルマップのイメージが/~foo/click.gifで、imagemapが/cgi-bin/imagemapにあり、mapファイルが/~foo/click.mapにあるとすると、クリッカブルマップの指定は次のようになる。

```
<A HREF="/cgi-bin/imagemap/
~foo/click.map">
<IMG SRC="/~foo/click.gif">
</A>
```

ここで気をつけてほしいのだが、mapファイルがユーザーのディレクトリー（~fooなど）にあるときはimagemap/に続くmapファイルの指定は~foo/...というように必ず「~+ユーザー名」から始めるようにする。他のURLのように相対パスでのファイル指定はできない。

ところで、NCSA httpdの最新バージョン（1.5）ではimagemapの機能がサーバー側に取り込まれるようになった（Internal Imagemap）ので、アンカーに/cgi-bin/imagemap....のように書かなくても直接mapファイルを指定するだけでクリッカブルマップを作れるようになっている。NCSA httpd 1.5を使うと、先ほどの例は次のようになる。

```
<A HREF="/~foo/click.map">
<IMG SRC="/~foo/click.gif">
</A>
```

ただし、これを実現するにはサーバーの設定を少々変更するか、ユーザーのディレクトリーに.htaccessファイルを用意して.mapファイルに関しての設定を追加しなければならない。なお、NCSA httpd 1.5のInternal Imagemapに関しての詳細は、http://hoohoo.ncsa.uiuc.edu/docs/tutorials/imagemapping.htmlを参照してほしい。

表3　mapファイル作成用のソフト一覧

OS	ソフト名	入手先	本誌付録CD-ROMに収録
Windows	MapEdit	ftp://ftp.netcom.com/pub/bo/boutell/mapedit	●
	MapThis!	ftp://ftp.coast.net/SimTel/win3/internet/mpths120.zip	
Mac	HyperMapEdit	ftp://ftp.comvista.com/pub/net/www/HyperMapEdit.sit.bin	●
	MacMapMaker	ftp://ftp.ncsa.uiuc.edu/edu/mmm/MacMapMaker.sea.hqx	
	WebMap	ftp://ftp.riken.go.jp/pub/info-mac/text/html/web-map101.hqx	
UNIX	MapEdit	ftp://ftp.netcom.com/pub/bo/boutell/mapedit	●

アクセスカウンターに挑戦

SSIによるプログラムの実行

最近はホームページに「あなたはxxxx人番目のお客様です」とホームページを見に来た人をカウントしているページも多くなってきた。このようなしかけもCGIで実現することはできるが、結構難しい。そこでもっと簡単なSSIを使ったアクセスカウンターの作り方を紹介しよう。

SSIはServer Side Includesの略。この機能はNCSA httpdと呼ばれるWebサーバーに固有の機能だ。したがって、NCSA httpdでしか利用できないことをあらかじめお断りしておく。SSIもCGIと同様に外部のプログラムを実行して、その結果をブラウザーに返す。ただし、CGIはたいていの場合フルコンテンツのデータが返されるが、SSIの場合はページの一部分にSSIの実行結果が埋め込まれる（インクルードされる）ような仕組みになっている。たとえば、時刻を表示するプログラムを作って、それをSSIで呼び出すと、ホームページの特定の場所に時刻が表示される。SSIを使えばUNIXコマンドの出力結果だけでなく、自分で作ったプログラムの出力結果をホームページにインクルードすることが可能だ。そのため、アクセスをカウントする部分をプログラムで作成して、アクセスカウンターを表示したい場所にSSIを使って埋め込めば、その場所にアクセスカウンターが表示される。

SSIのセットアップとHTMLの書き方

SSIもCGIと同様にWebサーバーでSSIが使えるように設定されていなければ使うことができない。SSIに関しても、多くのプロバイダーやサーバーは使用を許可していないため、あらかじめそのサーバーでSSIが使えるのかどうかを調べなければならない。SSIが使えるように設定されているサーバーでSSI機能を使うには、CGIのときと同様に.htaccessファイルに次の1行を追加する。

```
AddType text/x-server-parsed-html .html
```

この設定をしておけば、拡張子が.HTMLのファイルでSSIが使用可能になる。

次に、ホームページでSSIを使うには次のようなタグをHTML内に埋め込む。

```
<!--#exec cmd="コマンド" -->
```

<!--で始まるタグは、通常はコメントとして扱われるが、SSIが使えるようにセットアップされているときは、SSIのコマンドと認識される。#execはコマンドを実行しなさいという意味で、cmd=で始まるパラメーターには実行したいコマンドを書く。このようなタグをページに埋め込むと、<!--...-->で指定したタグの代わりに、コマンドの実行結果がページに表示される。また、SSIでは#execのほかにも#includeなどの指示を使うこともできるようになっている。なお、#exec以外のコマンドの詳細についてはhttp://hoohoo.ncsa.uiuc.edu/docs/tutorials/includes.html を参照してほしい。それでは、いくつかSSIを使ったサンプルを紹介しよう。

❶ 日付の表示

```
「現在の日付は<!-#exec cmd="date"-->です」
```

これはUNIXコマンドのdateの出力結果を埋め込む例である。つまり、この例の表示イメージは「現在の日付はyyyy年mm月dd日....です」となる。

❷ ログイン状況の表示

```
<!-#exec cmd="who"-->
```

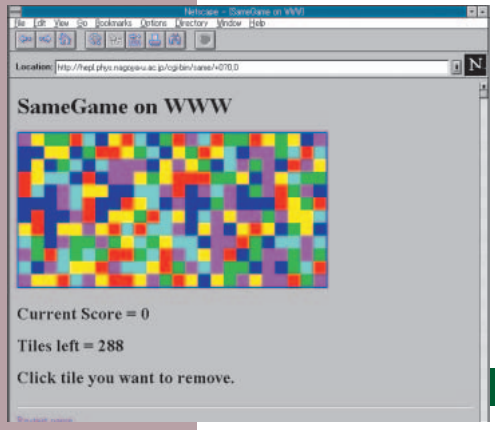


図17：おたのしみページ
Web上でできる3目ならべ、立体迷路、SamaGameなど

URL http://hepl.phys.nagoya-u.ac.jp/~mitsuru/games.html

これもUNIXのwhoコマンドを使ってそのときサーバーにログインしている人の一覧を表示する例である。このように、SSIを使えばコマンドの実行結果をページに取り込むことができる。また、このコマンドにはシェルスクリプトやPerlスクリプト、Cで作ったプログラムなどを指定することもできる。

アクセスカウンターを作る

それでは、SSIを使ってアクセスカウンターを作ってみよう。アクセスカウンターを作るには、アクセスをカウントしてそのときのアクセス数を表示するプログラムが必要だ。このプログラムをSSIを使ってホームページに埋め込めば、そこにカウント数を表示することができる。アクセスをカウントするには、まず、そのときまでのアクセス数を記録するファイルが必要となる。

❶ カウント数を記録するファイルを準備

まず、アクセス数を保存するためのファイルを作成する。ここでは、"count"という名前のファイルをカウントしたいホームページと同じディレクトリーに置くことにしよう。カウンターファイルを作成するには、エディターなどを使ってもよいが、echoコマンドを使って次のようにして作ることもできる。このときの数値がカウンターの初期値になる。

```
% echo 0 > count
```

カウンターファイルを作成したら、次のそのカウンターファイルを誰でも書き込めるようにパーミッションを設定する。Webサーバーからファイルがアクセスされるときは、自分のアカウントではなく、まったく別のアカウントになるので、他人が書き込めるようにしておく必要があるからだ。パーミッションの設定は次のコマンドを使う。

```
% chmod 666 count
```

❷ カウンター用のプログラムを作る

カウンターを保存するファイルができたら、カウントアップしてそのときのカウンターの値を返すプログラムを作る。ここでは、シェルスクリプトを使って次のようなプログラムを作ってみた。

```
#!/bin/sh
c = `/bin/cat count`
expr $c + 1 | /usr/bin/tee
count
```

このプログラムでは、まず先ほど作ったカウンターファイルの内容を読み込み、それを1つだけインクリメントして、ふたたびカウンターファイルに書き出す。このときに、標準出力にカウンターの値が出力されるようになっている。なお、このプログラム自体はcount.shという名前でセーブする。また、カウンターファイルと同様に、このプログラムは他人が実行できなければならないので、次のコマンドを使ってパーミッションを設定する。

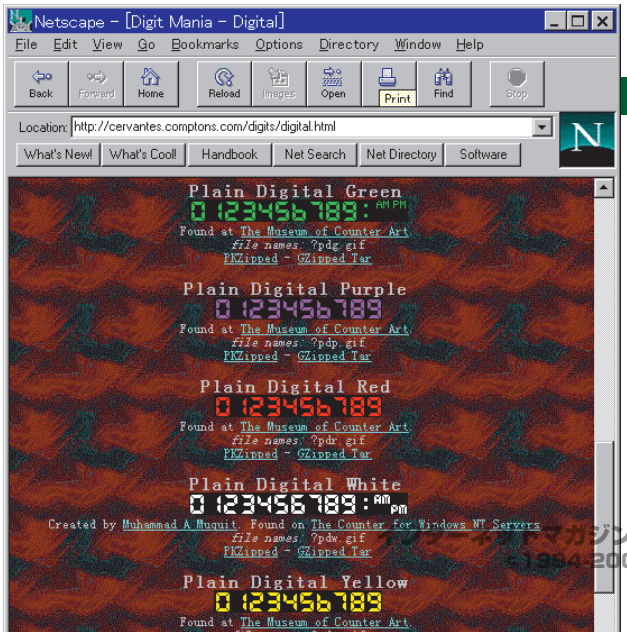
```
% chmod 755 count.sh
```

❸ ホームページにカウンタープログラムを埋め込む

最後に、カウントするプログラムをホームページに埋め込む。たとえば、次のようにすればよいだろう。

```
「あなたは<!--#exec cmd="./
count.sh"-->人目の訪問者です」
```

こうすれば、count.shの出力結果がホームページに埋め込まれる。また、count.shの出力を数値ではなく、<IMG SRC="1.gif"><IMG SRC="2.gif">....のようにイメージタグを出力するようにして、あらかじめ数字のグラフィックファイルを用意すれば、アクセスカウンターをグラフィックで表示することも可能だ。



URL http://cervantes.comptons.com/digits/

図18：Digit Mania
カウンター用のグラフィックが豊富にアップロードされている

フォームの内容をメールで受け取る

フォームを使ったCGI　CGIで最もポピュラーなのが、いわゆるメールフォームやアンケートフォームなど入力フォームを使ったものだろう。いろいろな項目を用意しておけば、読み手も反応しやすくなる。フォームを作ってフォームの内容を自動的にメールするようなCGIプログラムを書けば、ユーザーの反応をダイレクトに受け取ることも可能だ。また、アンケートなどの用途では、フォームに入力された内容をデータベースに登録するCGIプログラムやデータベースの集計結果を表示するようなCGIプログラムを作れば、まとまったアンケート結果を手軽に見ることも可能だ。

ここでは、フォームに入力された内容を自動的に自分宛にメールするCGIプログラムを作ってみることにしよう。

❶ HTMLでフォームを作る

まずしなければならないのは、入力フィールドを持つページを作ることである。つまり、フォームの定義だ。フォームを定義するには、HTMLの「FORMタグ」を使う（図19）。

```
<FORM ACTION="CGIプログラムのURL">
入力フィールドの定義....
</FORM>
```

これが、入力フォームの定義の基本となる部分である。なお、FORMタグにはMETHODを指定することができるが、ここではデフォルトのGETを使うことにする（p.168参照）。

フォームの基本部分ができたら、次は入力フィールドをデザインする。HTMLには通常のテキスト入力フィールド、複数行の入力ができるテキスト入力エリア、ラジオボタンやチェックボックス、リスト、ボタンを定義することが可能だ。

・テキスト入力フィールド

```
<INPUT TYPE="text" NAME="name" SIZE=n MAXLENGTH=n VALUE="value">
```

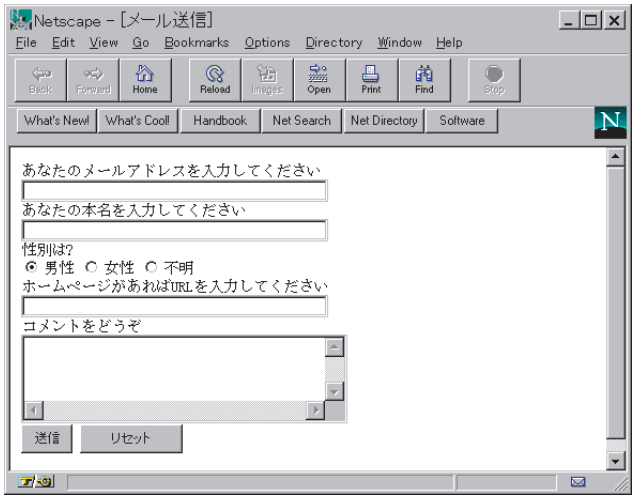
このタグは通常のテキスト入力フィールドのためのタグである。NAMEには入力フィールドの名前を指定する。SIZEとMAXLENGTHは省略が可能で、それぞれ入力フィールドの長さと、入力フィールドに入力できる最大文字列長を表す。また、VALUEが指定されているとあらかじめテキスト入力フィールドにVALUEの内容を入れておくことができる。また、typeを"text"ではなく"password"にすると入力文字がすべて"＊"で表示される。

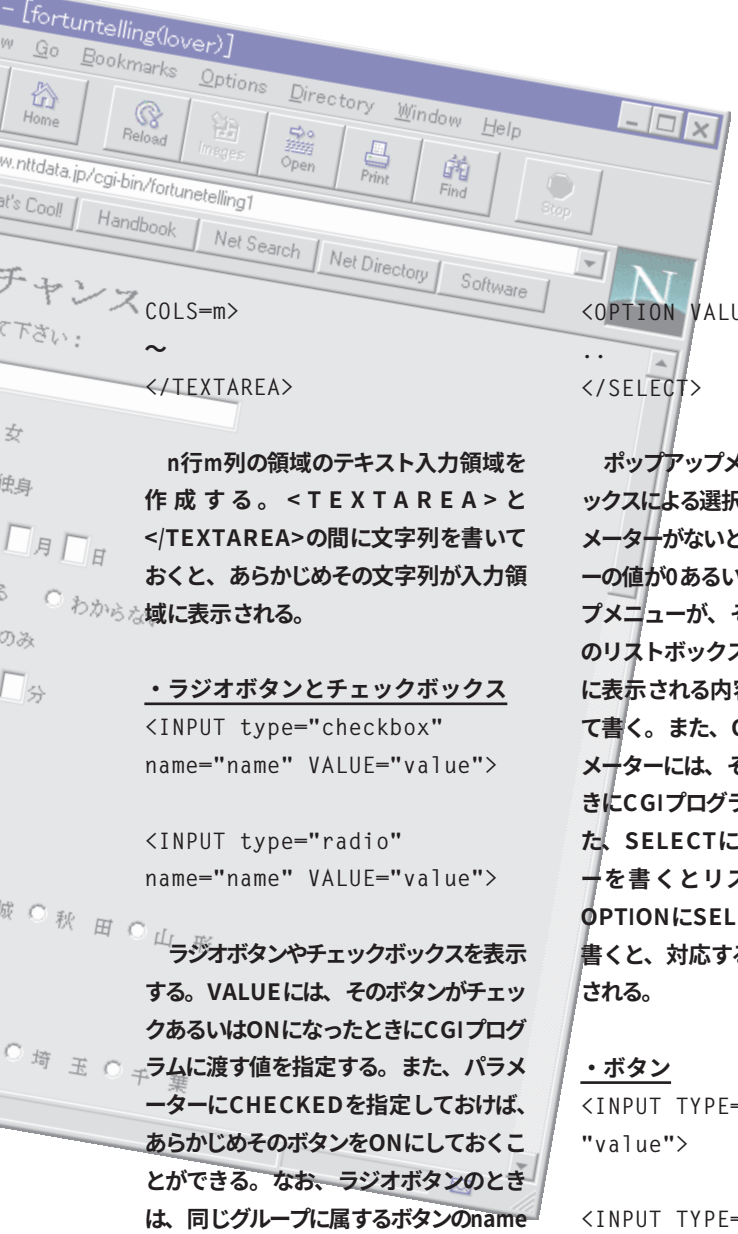
・テキスト入力エリア

```
<TEXTAREA NAME="name" ROWS=n
```

```
<HTML>
<HEAD>
<TITLE>メール送信</TITLE>
</HEAD>
<BODY>
<FORM ACTION="postmail.cgi">
あなたのメールアドレスを入力してください<BR>
<INPUT TYPE="text" name="email" SIZE=40><BR>
あなたの本名を入力してください<BR>
<INPUT TYPE="text" name="name" SIZE=40><BR>
性別は?<BR>
<INPUT TYPE="radio" NAME="sex" value="male" CHECKED>男性
<INPUT TYPE="radio" NAME="sex" value="female">女性
<INPUT TYPE="radio" NAME="sex" value="unknown">不明<BR>
ホームページがあればURLを入力してください<BR>
<INPUT TYPE="text" NAME="url" SIZE=40><BR>
コメントをどうぞ<BR>
<TEXTAREA NAME="comment" ROWS=4 COLS=40>
</TEXTAREA><BR>
<INPUT TYPE="submit" VALUE="送信">
<INPUT TYPE="reset" VALUE="リセット">
</FORM>
</BODY>
</HTML>
```

図19：HTMLでフォームのあるページを作る






```
COLS=m>
～
</TEXTAREA>
```

n行m列の領域のテキスト入力領域を作成する。<TEXTAREA>と</TEXTAREA>の間に文字列を書いておくと、あらかじめその文字列が入力領域に表示される。

・ラジオボタンとチェックボックス

```
<INPUT type="checkbox"
name="name" VALUE="value">

<INPUT type="radio"
name="name" VALUE="value">
```

ラジオボタンやチェックボックスを表示する。VALUEには、そのボタンがチェックあるいはONになったときにCGIプログラムに渡す値を指定する。また、パラメーターにCHECKEDを指定しておけば、あらかじめそのボタンをONにしておくことができる。なお、ラジオボタンのときは、同じグループに属するボタンのnameを同じ名前にすると、そのグループのうち1つのボタンだけをonにすることができる。

・リスト

```
<SELECT NAME="name"SIZE=
"n">
<OPTION VALUE="value">～
..
</SELECT>
```

ポップアップメニューあるいはリストボックスによる選択を表示する。SIZEパラメーターがないときや、SIZEパラメーターの値が0あるいは1のときはポップアップメニューが、それ以外のときはn行分のリストボックスが表示される。リストに表示される内容は<OPTION>に続けて書く。また、OPTIONのVALUEパラメーターには、その要素が選択されたときにCGIプログラムに渡す値を書く。また、SELECTにMULTIPLEパラメーターを書くとリストが複数選択型に、OPTIONにSELECTEDパラメーターを書くと、対応する要素があらかじめ選択される。

・ボタン

```
<INPUT TYPE="submit"VALUE=
"value">

<INPUT TYPE="reset"VALUE=
"value">
```

それぞれvalueという名前を持ったボタンを定義する。TYPE="submit"のボタンが押されると、FORMタグのACTIONで指定したCGIプログラムが実行される。また、TYPE="reset"のボタンはFORMタグ内のすべての入力フィールドを初期化するために使われる。

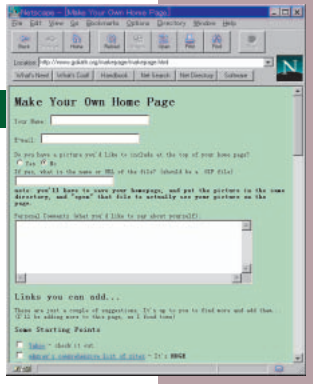
これらのFORMタグを使って入力フォームを作ったら、次はその入力フォームの内容を解釈して、それをメールするCGIプログラムを作る。

❷ CGIプログラムを作る

メール送信のためのCGIプログラムは、次の5つの部分に分けて考えることができる。

❶ 渡された内容をデコードする
❷ 渡された内容を分割する
❸ フォームに書かれた内容が正しいかどうかをチェックする。間違っていたら再度入力を促すためのページを生成する
❹ 以上の情報を使ってメールを送信する
❺ すべて終了したら、メールが送信されたことがわかるようなページを生成する

まず、CGIプログラムに渡される内容は、前述のとおりエンコードされているので、それを元の状態に戻す必要がある。また、フォームの入力フィールドに入れられた値は、name=value&name=value&name=value....のように連続して格納されているため、これを扱いやすい形



URL http://www.goliath.org/makepage/makepage.html

図20：Make Your Own Home Page
フォームに入力した情報をもとにHTMLを自動的に作成してくれる

式に変換しなければならない。

次に、入力された内容をチェックする。たとえば、送信者のE-Mail欄が空欄だったときなどにはメールが送信されないようにするためだ。このときはメール送信ができなかった旨のHTMLをCGIプログラムで生成すればよいだろう。

内容のチェックが済んだら、その内容をメールで送信する。プラットフォームがUNIXの場合はsendmailコマンドを使えばよいだろう。なお、sendmailコマンドについてはUNIXのmanなどで確認してみてほしい。なお、sendmailを使うときは、メールヘッダーをきちんと設定しないとメールが正常に送られないので注意しよう。

すべての処理が終わったら、正しくメールが送信されたことを伝えるためのHTMLをCGIプログラムで生成しておく。以上の処理を実現するコードが図21のpostmail.cgiだ。なお、ここではPerlを使ったが、その他の言語でも同様のことが実現可能だ。

URL http://www.worldwidemart.com/scripts/

図22：CGIのスクリプト集

図21：postmail.cgi
フォームの入力内容を自動的にメールするプログラム

```
#!/usr/local/bin/perl
#
# postmail.cgi : sending mail from www Version 0.1/Beta
#
$user = "foo@bar.or.jp";        #ここに受取人のアドレスを書く
$subject = 'subject';           #メールのサブジェクトを書く
#
# REQUEST_METHODをチェックして値を$bufferに格納する
#
if ($ENV{'REQUEST_METHOD'} eq 'POST') {
  read(STDIN,&buffer,$ENV{'CONTENT_LENGTH'});
} else {
  $buffer = $ENV{'QUERY_STRING'};
}
#
# フォームに入力された値をデコード/分解する
#
@pairs = split(/&/,$buffer);
foreach $pair(@pairs) {
  ($name,$value) = split(/=/,$pair);
  $value =~ tr/+/ /;
  $value =~ s/%([a-fA-F0-9][a-fA-F0-9])/pack("C",hex($1)/eg;
  $FORM{$name} = $value;
}
#
# フォームに入力された内容をチェック
#
if (FORM{'email'} eq "" || FORM{'name'}) {
  print "Content-type: text/html\n\n";
  print "<TITLE>ERROR</TITLE>\n";
  print "<H1>メールアドレスと本名を入力してください</H1>";
  exit (0);
}
#
# メールを送信する
#
if (open(ML,"| /usr/lib/sendmail $user")) {
  print ML "From: $FORM{'email'} ($FORM{'name'})\n";
  print ML "Subject: $subject\n";
  print ML "X-Mailer: postmail.cgi ver.0.1/Beta\n";
  print ML "X-URL: $FORM{'url'}\n";
  print ML "\n";
  print ML
"------------------------------------------------------------\n";
  print ML "$FORM{'sex'}\n";
  print ML
"------------------------------------------------------------\n";
  print ML "$FORM{'comment'}";
  print ML
"\n------------------------------------------------------------\n";
  print ML "Remote host: $ENV{'REMOTE_HOST'}\n";
  print ML "User Agent : $ENV{'HTTP_USER_AGENT'}\n";
  close(ML);
  print "Content-type: text/html\n\n";
  print "<TITLE>Request Accepted</TITLE>\n";
  print "<H1>メールは正常に送信されました</H1>\n";
} else {
  print "Content-type: text/html\n\n";
  print "<TITLE>Request Rejected</TITLE>\n";
  print "<H1>メールは正常に送信されませんでした</H1>\n";
}
exit(0);
```

［インターネットマガジン　バックナンバーアーカイブ］ ご利用上の注意

このPDFファイルは、株式会社インプレスR&D（株式会社インプレスから分割）が1994年～2006年まで発行した月刊誌『インターネットマガジン』の誌面をPDF化し、「インターネットマガジン　バックナンバーアーカイブ」として以下のウェブサイト「All-in-One INTERNET magazine 2.0」で公開しているものです。

http://i.impressRD.jp/bn

このファイルをご利用いただくにあたり、下記の注意事項を必ずお読みください。

- 記載されている内容（技術解説、URL、団体・企業名、商品名、価格、プレゼント募集、アンケートなど）は発行当時のものです。
- 収録されている内容は著作権法上の保護を受けています。著作権はそれぞれの記事の著作者（執筆者、写真の撮影者、イラストの作成者、編集部など）が保持しています。
- 著作者から許諾が得られなかった著作物は収録されていない場合があります。
- このファイルやその内容を改変したり、商用を目的として再利用することはできません。あくまで個人や企業の非商用利用での閲覧、複製、送信に限られます。
- 収録されている内容を何らかの媒体に引用としてご利用する際は、出典として媒体名および月号、該当ページ番号、発行元（株式会社インプレスR&D）、コピーライトなどの情報をご明記ください。
- オリジナルの雑誌の発行時点では、株式会社インプレス R&D（当時は株式会社インプレス）と著作権者は内容が正確なものであるように最大限に努めましたが、すべての情報が完全に正確であることは保証できません。このファイルの内容に起因する直接的および間接的な損害に対して、一切の責任を負いません。お客様個人の責任においてご利用ください。

このファイルに関するお問い合わせ先
株式会社インプレスR&D
All-in-One INTERNET magazine 編集部
im-info@impress.co.jp