

A Brain-Friendly Guide

Head First PHP & MySQL

頭とからだで覚えるWebアプリケーション開発の基本



Discover the secrets
behind dynamic,
database-driven sites



Load all the key
syntax directly
into your brain



Avoid
embarrassing
mishaps with
web forms



Flex your scripting
knowledge with dozens
of exercises

O'REILLY®

The O'Reilly logo and "Tools@maruhouse.org" personal use only.

Lynn Beighley
Michael Morrison

著

佐藤 嘉一 訳

Head First PHP & MySQL

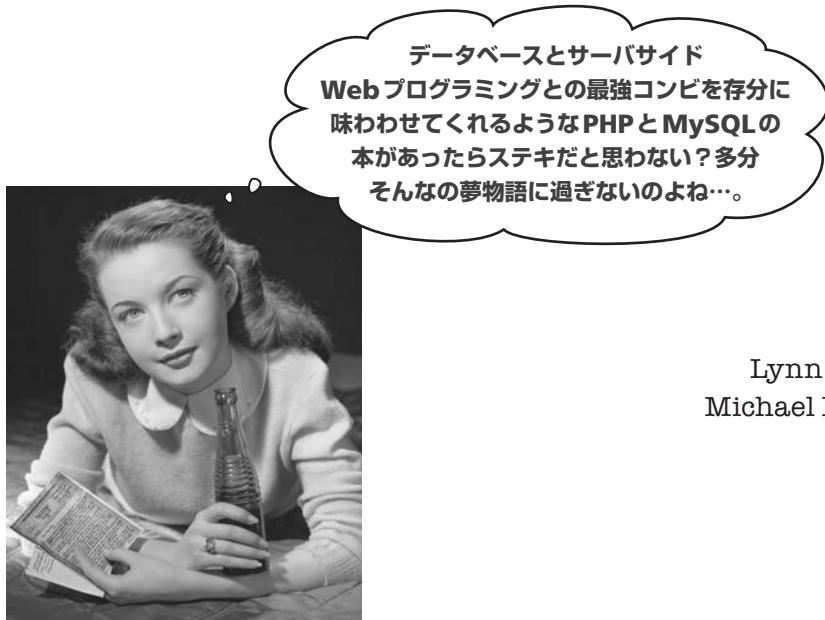
頭とからだで覚えるWebアプリケーション開発の基本

Lynn Beighley 著
Michael Morrison
佐藤 嘉一 訳

O'REILLY®
オライリー・ジャパン

本文中の製品名は、一般に各社の登録商標、商標、または商品名です。
本文中ではTM、[®]、[©]マークは省略しています。

Head First PHP & MySQL



データベースとサーバサイド
Web プログラミングとの最強コンビを存分に
味わわせてくれるような PHP と MySQL の
本があったらステキだと思わない? 多分
そんなの夢物語に過ぎないよね…。

Lynn Beighley
Michael Morrison

O'REILLY®

Beijing • Cambridge • Köln • Sebastopol • Tokyo

© 2010 O'Reilly Japan, Inc. Authorized Japanese translation of the English edition of Head First PHP & MySQL. © 2009 O'Reilly Media, Inc. This translation is published and sold by permission of O'Reilly Media, Inc., the owner of all rights to publish and sell the same.

本書は、株式会社オライリー・ジャパンがO'Reilly Media, Inc.との許諾に基づき翻訳したものです。日本語版についての権利は、株式会社オライリー・ジャパンが保有します。

日本語版の内容について、株式会社オライリー・ジャパンは、最大限の努力をもって正確を期していますが、本書の内容に基づく運用結果についての責任は負いかねますので、ご了承ください。

ひっきりなしにWebアプリケーションを使い、常にそこへと
私を導いてくれた両親に捧ぐ。

—— Lynn Beighley

Rasmus Lerdorfに捧ぐ。彼が片手間で作った言語は、後にPHPとして世に認められるようになった。このことにより彼は1人でも新しく、より輝かしい未来への道を示すことができることを証明したのだ。

—— Michael Morrison

Head First PHP & MySQLの著者



Lynn Beighley



Michael Morrison

Lynn Beighleyはフィクション作家なのですが、技術書の著者に捕まってしまいました。技術書を書くというのがペイする仕事だと分かってからは、現状を受け入れるどころか、むしろ楽しんで仕事をするようになりました。一度学校に戻って、コンピュータサイエンスの修士号を取得すると、頭文字がNRLとLANLという会社で働きました。次にFlashに出会い、最初のベストセラー作品を書きました。最悪のタイミングで、シリコンバレーに引っ越しました。つまり大不況の直前です。数年間はYahoo!で働き、他の本を書いたり、講師をしたりしました。現在は、作家としてのクリエイティブな好みに身を投じるため、ニューヨークに引っ越し、クリエイティブライティングでファインアートの修士を取得しようとしています。彼女はHead Firstスタイルの学位論文を教授や仲間の学生というクローズドな部屋に提出しました。この論文は非常に賞賛され、彼女は学位を得ると同時に、Head First SQLを仕上げ、さらに今まさにHead First PHP & MySQLを仕上げました。すばらしい！

Lynnは旅行と文筆が好きです。また、ズボの素人向けに背景を説明するようなストーリーを作り上げることも好きです。ただUFOを恐いと思っています。

Michael Morrisonは、オンラインの世界に対して熱狂的に貢献してきました。自分のコモドール64でBBSを動かして以来ずっとです。今と違ってコンピュータオタクといった人種がまるで認められていなかった頃の話です。数千bps程度以下の通信速度で、如何に遠くへ、如何に速くと考え続けていました。Michaelは今やBBSを動かしていませんが、相変わらず最先端の同等技術やツールに関わっています。それらを使えば同等のものを作ることができます。彼は仕事としては、大部分の時間をWeb関係の技術について執筆しています。著者または共著者として50冊以上を出しています。分野はモバイルゲームプログラミングからXMLと多様です。彼がHead Firstに進出したのは、Head First JavaScriptからで、それ以来後ろを向いたことはありません。

Michaelは、Stalefish Labs (www.stalefishlabs.com) の創設者でもあります。この会社はエンタメ系を専門としていて、ゲーム、おもちゃ、対話型のメディアなどを扱っています。因みに彼は実際にはオフラインで時を過ごすことでも有名です(爆)。スケボー、アイスホッケーをよくやっています。また奥さんのMasheedと一緒に鯉のいる池に出かけていくのも好きです。それとしつちゅう居眠りをしています。

用語・訳語について

本書では英単語に対する訳語や記号に対する用語について、さまざまな配慮で、注意深く単語を選んでいます。原則として、日本で普通に使われている用語を使用するよう努めていますが、あえて変更して訳出したものもあります。以下、本書において特に注意すべき用語を対応表としてまとめてあります。

英語	本書での訳語	一般的な訳語	備考
submit	「提出」する	送信する	HTMLのsubmitボタンを押すことにより本当に「送信」できるかどうかは、読者の方々がこれから作るPHPスクリプトにより決まるのです。
logic	論理、アルゴリズム、方程式など意味に応じて適宜	ロジック	一般にロジックという用語は極めて曖昧に使われています。特にアルゴリズムの意味で使われているのは完全に誤りです。
query	問い合わせまたは問い合わせ文	クエリ	データベースの世界では、もはやクエリが業界用語となっています。本書は初心者向けとの配慮であえて説明的な訳語を用いました。
HTTP authentication	HTTP認証	ベーシック認証 (Basic認証)	原著に合わせてこのようにしました。
{、} (curly brace)	波カッコ	中カッコ	直感的に誤解のない表現で統一。
[、] (bracket)	シカクカッコ	大カッコ	直感的に誤解のない表現で統一。
> (greater than)	より本当に大きい	より大きい	より大きいという表現が等しいものを含むのか微妙に曖昧なため。
< (less than)	より本当に小さい	より小さい	上記と同様
>= (greater or equal)	等しいかまたは大きい	以上	以上という表現が等しいものを含むのか微妙に曖昧なため。
<= (less or equal)	等しいかまたは小さい	以下	上記と同様

目次 (要約)

序章	xxv
1 章	ページは生き物だ：静的なページに命を吹き込む	1
2 章	お互いに接続するには？：2章 MySQLとつなげる	47
3 章	自分のデータを作る：データベースを作ってそれを使う	81
4 章	Web上のアプリケーション：現実的で実用的なアプリケーション	129
5 章	データベースでは十分でないとき： ファイルに格納されたデータを使う	183
6 章	みんなが自分を狙っていると思え： アプリケーションにセキュリティを与える	245
7 章	覚えてますか？：個人向けのWebアプリを作る	291
7.5 章	シェアすればケアできる：ダブったコードをなくす	357
8 章	データを収穫する：データをコントロールし、 世界をコントロールする	363
9 章	関数を通してより良い生活を：文字列とカスタム関数	433
10 章	置き換える規則：正規表現	493
11 章	世界とのインターフェース：シンジケーションとWebサービス	533
付録i	本編でカバーしなかったこと：残飯	589
付録ii	作業をする場所：開発環境をセットアップする	593
付録iii	PHP & MySQL日本語対応	607
索引	618

序章

あなたの脳をネットワーク構築に向けましょう。あなたは何かを学ぼうとしていますが、脳は学習内容を定着させないようにしています。あなたの脳は「どの野生動物を避けるべきかや裸でスノーボードをすることは悪いことかどうかなど、もっと重要なことに考える余地を残しておく方がよい」と考えています。そこで、ネットワーク構築を学習することであなたの人生が大きく変わると脳が考えるように仕向けるためにどのような策を講じればよいでしょう？

この本を読むのにふさわしい人は？	xxvi
みなさんが思っていることはわかっています	xxvii
メタ認知：自分の思考について考える	xxix
脳を服従させるために「みなさん」ができるここと	xxxi
まず最初に読んでください	xxxii
テクニカルレビューチーム	xxxiv
謝辞	xxxx

1 章 静的なページに命を吹き込む

ページは生き物だ

これまでHTMLにCSSをちりばめて多くのWebページを作ってきたことでしょう。しかしすでにお気づきのように、できあがったサイトにアクセスしてくれる人は、受身です。つまりページの内容を見ること以外には何もできないのです。ここでのコミュニケーションは一方通行です。これを変えてみたくありませんか？ 実際、サイトを見てくれている人が何を考えているのかを知りたいはずです。そのためには、その人たちの考えを理解するためにWebフォームに情報を入力させるようになります。そしてその情報を処理し、かつ自分のところに持っていく必要があります。つまりサイトを単なるHTMLから次のレベルに引き上げる時が来たとは思いませんか？

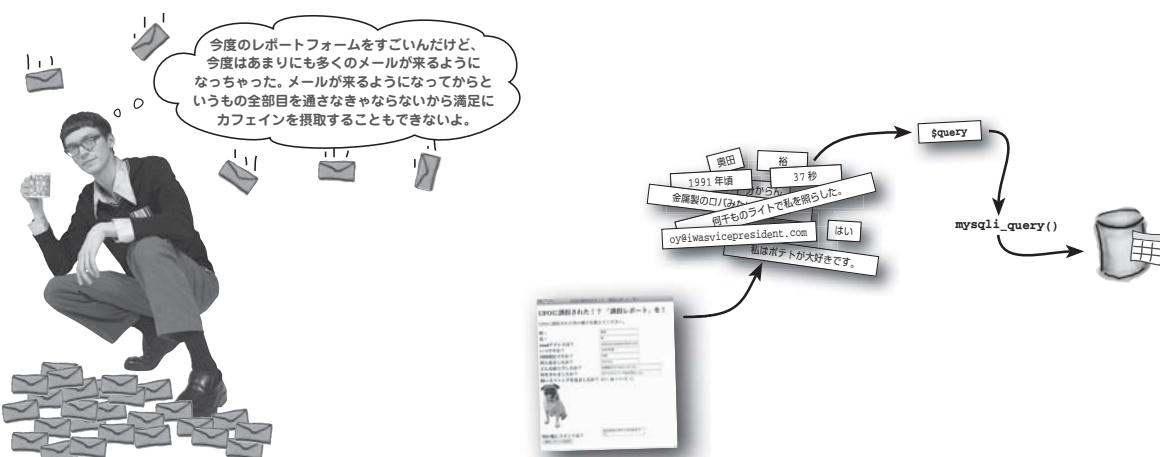


HTMLは静的で退屈です	2
PHPはWebページに命を吹き込んでくれます	3
フォームにより全ストーリーを得られる	5
フォームはHTMLでできています	6
HTMLフォームは問題を抱えています	8
HTMLはクライアント上で動作します	10
PHPはサーバ上で動作します	11
PHPスクリプトはサーバで動作します	12
PHPを使ってフォームのデータにアクセスする	16
PHPスクリプトはサーバ上にいなければならぬのです！	18
サーバがPHPをHTMLに変えます	22
コーディング上のPHP規則	25
完璧な変数名を見つける	26
\$_POSTはフォームのデータを保持する特別な変数です	28
\$_POSTはフォームのデータをスクリプトに運んでくれます	29
PHPでメッセージ本文を作成する	36
単なるテキストもフォーマットできます…ちょっとだけ	38
改行には二重引用符の文字列が必要です	39
オーウェンに飛ばすメールメッセージを組み立てる	40
変数はメールの要素や部品を格納しています	41
PHPでメールメッセージを送る	42
オーウェンにメールが来始めました	45

9 章 MySQLとつなげる お互いに接続するには？

構築を始める前にどのように接続するか知っておくというのは良いことです。最初の自前のPHPスクリプトはできあがり、ちゃんと動いています。でもフォームが結果としてメールになって飛んでくるというのは、実はちっともよくありません。フォームの結果を保存する方法が要ります。そして必要な間は結果をとっておくことができて、見たくなったら結果を取ってくることができたら素敵です。MySQLデータベースを使えばデータを安全に保存しておくことができます。でも前章で作ったPHPスクリプトをMySQLデータベースのところまで持ってこなければ何も始まりません。

オーウェンのPHPフォームはちゃんと動きますが、 ちゃんとしすぎです.....	48
MySQLはデータの格納に優れています.....	49
オーウェンにはMySQLデータベースが必要です.....	50
MySQLのデータベースとテーブルを作る.....	52
INSERT文の動作.....	55
テーブルのデータを見るにはSELECTを使う.....	58
PHPにウンザリSQLの処理をやらせる.....	61
PHPによりデータはオーウェンのWebフォーム上を 渡り歩く.....	62
PHPからデータベースに接続する.....	64
PHPスクリプトでデータを挿入する.....	65
データベースとお話しするPHPの関数を使う.....	66
\$_POSTがフォームのデータを供給します.....	71
オーウェンはデータを選別する必要が出てきました.....	76
オーウェンのファンクション探しはやっと軌道に乗りました.....	78





》章 データベースを作ってそれを使う

自分のデータを作る

必要なデータが常にあるとは限りません。

使う前にデータを作らなければならないことがあります。そんなデータを入れておくテーブルを作らなければならないこともあります。使う前に作らなければならないデータを入れておくデータベースを作らなければならないこともあります。混乱してませんか？心配なく。自分自身のデータベースとテーブルの作り方を学ぶ準備が整いました。もしこれで十分でなくても、自分自身の最初のPHP & MySQLアプリケーションをおいおい構築していくことになります。

エルビストアーを本格的に開店します	82
清野君にはアプリケーションが必要です	83
清野君のアプリケーション設計を視覚化する	84
何事もテーブルから始まる	87
MySQLサーバにコンタクトを取る	88
清野君のメール用データベースを作る	89
さっきのデータベースの中にテーブルを作る	90
データを定義する必要があります	91
MySQLのデータ型たちを一堂に会しましょう	92
問い合わせせ文でテーブルを作る	95
使う(use)前にデータベースをUSEする	97
DESCRIBEでテーブル構造を明らかにする	99
清野君はデータ格納準備完了です	101
メール追加用のスクリプトを作る	102
清野君のアプリケーションにはもう1つ必要です	109
メール送信用スクリプトにおけるボルトとナット	110
最初にすべきこと、データを捕まえる	111
mysqli_fetch_array()で問い合わせ結果を取ってくる	112
WHILE(しばらく)の間ループする	113
whileでデータをループする	114
メールが来ました…清野君から！	119
時には出て行く人も	120
DELETEでデータを取り除く	121
WHEREを使って特定のデータをDELETEする	122
間違って削除してしまうというリスクを最小化する	123

4 章 現実的で実用的なアプリケーション

Web上のアプリケーション

時には現実的になって計画を見直さなければなりません。

そうでなければ、もっと初期の段階で十分注意深く計画を練る必要があります。アプリケーションがWeb上に出て行ってしまってから、あまり十分に計画を練っていないかったと気がつくかもしれません。自分で動くだろうと思っていたことは、現実の世界では不十分なものです。本章では、現実の世界での問題をいくつか見ていきます。このようなことはご自身のアプリケーションがテストを経て実サイトにて行った時に実際に起こりうることなのです。問題を解決するために、今までより重要なPHPとMySQLのコードも見ていきます。

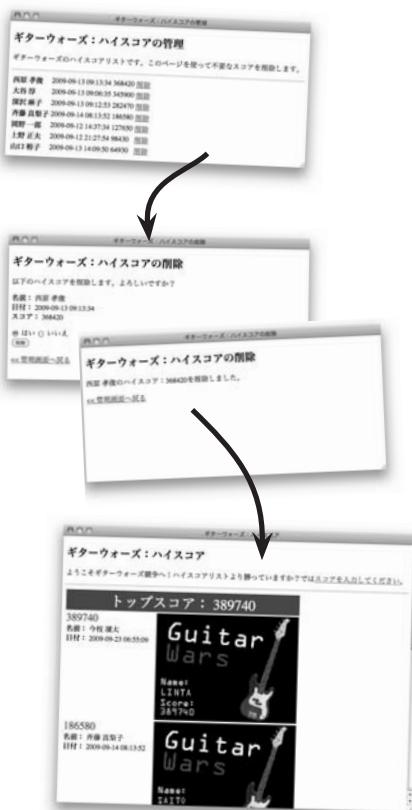


清野君はいらついている顧客を作り出しました.....	130
清野君を…清野君から守る	131
ちゃんとしたフォームデータを要求する	132
メール送信用画面の妥当性検証の背後にある論理	133
IFを使って判断を下せます.....	134
真偽値のテスト	135
IFは等価性以外もチェックします.....	136
PHP関数で変数の妥当性を検証する	140
複数の条件を論理積と論理環でテストする	143
フォームのユーザにフィードバックが必要です	147
必要に応じてPHPから出たり入ったりする	155
フラグを用いて重複のコードをなくす	156
HTMLフォームは一度だけコーディングする	157
自分自身を参照するフォーム	161
スクリプトでフォームのactionを指す	162
フォームが「提出」されたかどうかをチェックする	164
まだムカついているユーザがいます	168
テーブル行は一意に識別できるべきです	170
主キーは一意性を強要する	172
チェックボックスから顧客IDへ	177
foreachにより配列をループする	178

5 章 ファイルに格納されたデータを使う

データベースでは十分でないとき

データベースに関する…誇大広告に騙されてはいけません。確かにデータベースはテキストなどあらゆる種類のデータを格納するという驚くべき仕事をしてくれます。でもバイナリデータはどうでしょうか？例えばJPEGイメージとかPDFドキュメントなどです。珍しいギターピックコレクションの写真を全部データベースにぶち込むことに意味があると思いますか？普通はありません。その手のデータはファイルに格納するのが普通で、ファイルのままにしておくものです。でも仮想的なケーキを持って仮想的に食べることが完璧にできてしまうようになります。本章では、**ファイルとデータベースとを一緒にしてバイナリーデータで満ちあふれたPHPアプリケーションを構築するための秘密**を暴露します。



仮想ギタリストはコンペ好きです	184
証拠は写真の中にある	185
アプリケーションにイメージを格納する必要があります	186
ギターウォーズにイメージファイルアップロードを計画する	191
ハイスコアデータベースはALTERしなければいけません	192
ユーザからイメージをもらうにはどうすればよい？	196
データベースにイメージファイル名をぶち込む	198
アップロードされたファイルの名前を見分ける	199
アップロードしたファイルはどこへ行った？	200
アップロードされたイメージファイル用に家を作る	202
共有データは共有しなければならない	208
共有スクリプトデータはrequireする	209
require_onceは「挿入」と考える	210
順序がハイスコアのすべて	212
ギターウォーズのトップに勝者の栄誉を与える	215
HTMLとCSSでトップスコアをフォーマットする	216
小さなイメージだけを許可する	221
ファイルの妥当性検証により、	
アプリはもっと強固になります	222
管理用ページを計画する	226
管理用ページにスコア削除リンクを付ける	229
スクリプト同士でコミュニケーションをとることが	
できます	230
GETとPOSTについて	232
GETとPOSTとハイスコアの削除と	234

6

章 アプリケーションにセキュリティを与える みんなが自分を狙っていると思え

両親の言っていたことは正しかったのです。「知らない人と話してはいけません。」または少なくとも知らない人を信用してはいけません。何を差し置いても、知らない人にアプリケーションデータのカギとなるものをあげてはいけません。知らない人が正しいことをちゃんとしてくれるなどという仮定をしてはいけません。世の中一步外に出れば恐ろしいことだけです。誰でも信用できるなどと思ってはいけません。実際、アプリケーションの開発者として、時には皮肉屋になり、時にはテロリストとして陰謀を企ててみなければなりません。そうです。性悪説に立って下さい。みんな間違なくあなたを狙っているのです！まあ、確かにちょっと大げさかもしれません。でもセキュリティについて真剣に考えてアプリケーションを設計するというのは非常に重要なことです。そうでなければ危害を加える可能性のある誰かに対抗してアプリケーションを守ることはできません。



音楽が死んだ日	246
うじゅうじゅういる大群をセキュリティで監禁する	247
ギターウォーズの管理者用ページを守る	248
HTTP認証にはヘッダが必要です	250
ヘッダを丸裸にする	252
PHPでヘッダをコントロールする	253
ヘッダによる認証	254
認証用スクリプトを作る	262
ギターウォーズエピソードII：ハイスクアクローンの攻撃	266
足し算による引き算	267
セキュリティには人間が要ります	268
ギターウォーズを検査するための計画	269
ALTERで承認の置き場所を作る	270
未承認のスコアには価値がない	275
100万点ハック	278
すべて検査済み…？	279
彼女は正確には一体何をしたのか？	281
コメントでMySQLをごまかす	282
スコア追加用フォームはSQLを注入されたのです	283
SQL注入からデータを守る	284
(パラメタ付きの)安全なINSERT	285
フォームの妥当性検証にはやり過ぎということはありません	287

7 章 個人向けのWebアプリを作る 覚えてますか？

誰でも忘れられるのはイヤです。もちろんWebアプリケーションのユーザだってそうです。アプリケーションに「メンバ」という概念があるのなら、つまりユーザがアプリケーションと何らかの個人的な相互作用をするのなら、アプリケーションはユーザを覚えておく必要があります。家のドアを開ける度に家族の誰かに対して自己紹介をしなければならないなどというのはウンザリなはずです。でもその必要がないのは、メモリ（記憶）というすばらしいものがあるからです。ところがWebアプリケーションは人々を自動的に覚えてはくれません。そのような機能は、有能なWeb開発者に委ねられています。その人がその人の裁量で使える（もちろんPHPとMySQLの）ツールを使ってユーザを自動的に覚えてくれる個人向けのWebアプリを作れるかどうかにかかっています。



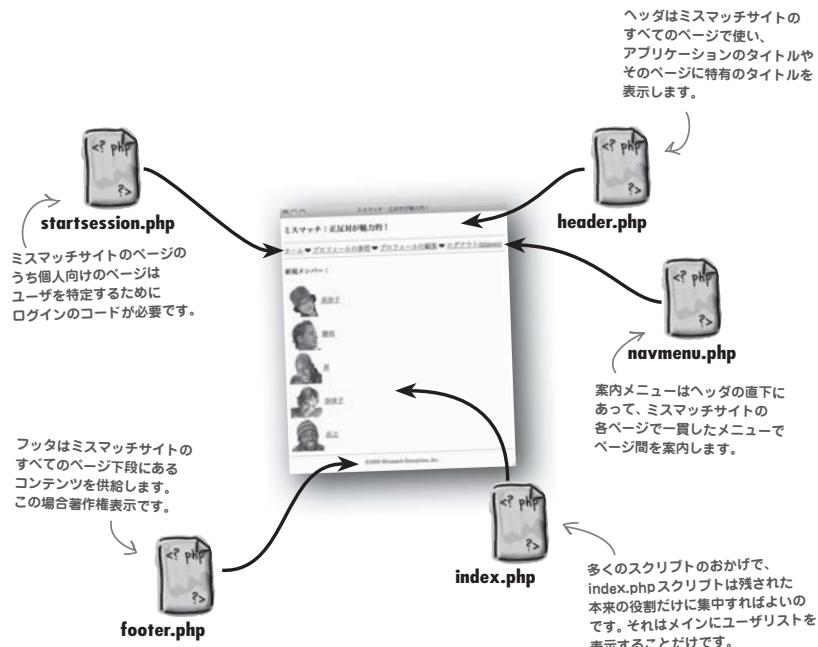
正反対が魅力的とよく言われます	292
ミスマッチサイトは個人の情報がすべてです	293
ミスマッチサイトにはログインが必要です	294
ログイン用のデータベースを準備する	297
ログイン・インターフェースを作る	299
パスワードをSHA()で暗号化する	300
パスワードを比較する	301
HTTP認証でユーザーとしてログインする	303
新しいユーザーをサインアップするためのフォーム	307
クッキーの中にあるものは？	317
PHPでクッキーを使う	318
ログインのフローを再考する	321
クッキー強化版ログイン	322
ログアウトとはクッキーを削除することを意味します	327
セッションはクライアントに依存しません	331
セッションデータを追跡する	333
ミスマッチサイトをセッションで刷新する	334
セッションを使ってログアウトする	335
セッションへの進化を完了させる	340
ユーザは嬉しく思っていません	346
セッションは短命です	348
…でもクッキーなら永遠に続きます！	349
セッション+クッキー=すごいログイン永続性	351

7.5 章 ダブったコードをなくす

シェアすればケアできる

シェアできるのは傘だけではありません。Web アプリケーションでも、同じコードが 2箇所以上にダブっているというような状況に陥ることがあります。これは無駄なだけでなく、メンテナンス時には頭痛の種となります。不意に何らかの変更をしなければならなくなったとき、変更はすべてのダブっている箇所に波及するためです。このような事態を解決するには、ダブったコードをシェアすることでなくしてしまえばよいのです。言い換えれば、ダブったコードは一箇所に貼り付けておき、それ以外の箇所からはそのコードを必要に応じて参照するようにすれば良いのです。ダブったコードをなくせば、アプリケーションはより効率的になり、メンテナンスも容易になり、最終的には堅牢なものとなります。

テンプレートでミスマッチサイトを作り直す	358
テンプレートを使ってミスマッチサイトを再構築する	360
ミスマッチサイトがまた完成しました…	
しかもはるかによい構成で	362

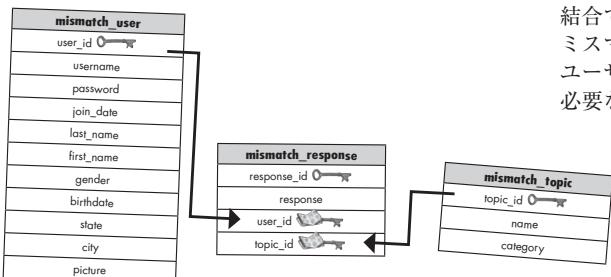
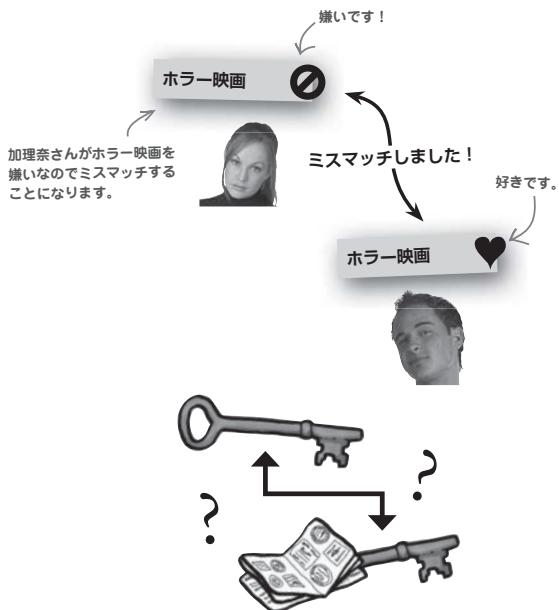


8

章 データをコントロールし、世界をコントロールする データを収穫する

データ収穫の秋は最高です。

有り余る程の情報が、検査、ソート、比較、結合を待っています。そして一般にカラー Web アプリが必要とするすべての処理は何でもできます。十分に？もちろんです。でも本当の収穫とは違って、MySQL データベースのデータをコントロールするには、ちょっとした労力とかなりの専門知識が必要です。Web ユーザはくたびれしおれたデータなんか欲しくありません。そんなもの退屈で全然魅力がありません。欲しいのは、豊富で十分で、しかも適切なデータです。さあ、ぐずぐずしている場合ではありません。MySQL トラクターを始動させて、作業開始です！



完全なミスマッチにする	364
ミスマッチさせるにはデータがすべてです	365
スキーマでデータベースをモデル化する	367
複数のテーブルを結び付ける	372
外部キーの動作	373
テーブルは行と行でマッチします	374
1行が複数行に対応する	375
多対多で行をマッチさせる	376
ミスマッチのアンケートを作る	381
データベースにレスポンスを入れる	382
データでフォームを駆動する	386
ミスマッチ用のアンケートフォームを作る	392
正規化に向けてちょっとがく	398
正規化するならアトムを考える	399
正規化されたデータベースへの3ステップ	401
ミスマッチのデータベースを変更する	405
ミスマッチサイトは本当に正規化されたのでしょうか？	406
問い合わせ文の中の問い合わせ文の中の問い合わせ文の中の…	408
みんなテーブルをつなぎましょう	409
ドットで結ぶ	410
内部結合でまだたくさんのことことができます	411
テーブルやカラムに愛称をつける	413
結合で救出する	414
ミスマッチを成功に導く5つのステップ	421
ユーザの「ミスマッチ性」を比較する	423
必要なものはforループです	424

9 章 文字列とカスタム関数

関数を通してより良い生活を

関数によってアプリケーションは完全に新しいレベルになります。

すでに PHP の組み込み関数を使って何かを達成したことはたくさんあります。これから更に本当に役立つ組み込み関数をいくつか見ていくことにします。更に続けて自分自身の **カスタム関数** の作り方を学びます。これで、これまでの想像を遥かに越えるところまで行き着くことができます。まあこの章で達人の域まで昇りつめることは無理でしょうが、カスタム関数によってコードは効率的になり、再利用可能となります。



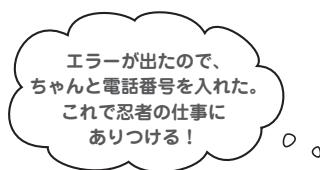
割のいい危険な仕事は見つけにくいものです	434
いまの検索だとエラーに対するゆとりが全くありません	436
SQL問い合わせ文は LIKE を使って柔軟にできます	437
文字列を単語単位に爆破 (explode) する	442
implode() は部分文字列から文字列を作ります	445
検索文字列の前処理	451
不要な検索文字を置き換える	452
問い合わせ文にはちゃんとした検索語が必要です	456
空でない要素を新しい配列にコピーする	457
時には文字列の一部しか必要ないこともあります	460
部分文字列はどちらの端からでも取り出せる	461
複数の問い合わせ文で結果をソートできます	464
関数でコードを再利用する	468
カスタム関数で問い合わせ文を作る	469
カスタム関数はどこまでカスタムか？	470
SWITCH は IF よりずっと多くの判断をします	474
build_query() にソート機能を加える	477
結果をページに割り付けることができます	480
LIMIT で必要な行だけ持ってくる	481
LIMIT でページリンクをコントロールする	482
ページ割り付けデータを追跡する	483
ページ割り付け用の変数をセットアップする	484
問い合わせ文を書き直して結果をページ割り付けする	485
ページ案内用のリンクを作る	486
すべてを合わせて検索用スクリプトを完成させる	489
完成版検索用スクリプト、続き…	490

10章 正規表現

置き換えの規則

文字列を扱う関数は愛用できるものばかりです。しかし同時に非常に制限があります。当然ながら、文字列の長さを教えてくれたり、一部を切り取ってくれたり、特定の文字を別の文字に変えてくれたりします。でも時にはもっと自由になって、もっと複雑な文字列操作に挑戦する必要が出てくることがあります。こんな時こそ正規表現の出番です。正規表現を使えば1つの基準ではなく規則の集合に基づいて、文字列を正確に変更することができます。

リスキージョブサイトから履歴書(レジュメ)を 送付できるようにする	494
データがどうあるべきかを決める	498
電話番号のパターンを定式化する	501
正規表現でパターンにマッチさせる	502
メタ文字を使ってパターンを作る	504
文字のクラスでパターンを改良する	511
preg_match()でパターンをチェックする	516
電話番号データを標準化する	523
不要な文字を捨てる	524
PHPを使ってドメインをチェックする	531



11 章 シンジケーションとWebサービス

世界とのインターフェース

外の世界というのはものすごく広いので、Webアプリケーションとしては無視するわけにはいきません。さらに重要なことは、世界から自分のWebアプリケーションを無視されないようにしなくてはいけないということです。良い方法があります。世界を自分のWebアプリケーションに向けるには、データをシンジケーション[†]で使えるようにすればよいのです。シンジケーションとはユーザがWebサイトのコンテンツを購読できるようにすることです。今までのようにユーザがWebサイトを訪問して直接新しい情報を探さなければならないのではありません。そればかりか、自分のアプリケーションがWebサービスを通して他のアプリケーションとインターフェースを取って、他のデータによるメリットを使って、もっとリッチな操作性を提供することができるのであります。



西門子

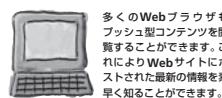
UFOに誘拐された！?

ようこそモザイクへ。宇宙人に誘拐されたことがありますか？説明映像はありますか？私の長い大ファンを貽ませんでし
たか？説明レポートをお願いします！

最近更新された説明レポート：

2009-06-18: 金川 離	説明されていた判別： 宇宙人の様子： 金川さん	大きさ： 金川さんくらいの大きさの船でした。	ファンダムはいましたか？ yes
2009-07-11: 佐藤 友也	説明された様子： 45歳	説明の中で「手足はひょろひょろでした。」	ファンダムはいましたか？ yes
2009-07-08: 田中 鶴朗	説明されていた判別： 宇宙人の様子： 田中さん	同じことをもうかる（=UFOが少なくとも）聞いたようです。	ファンダムはいましたか？ yes
2009-06-21: 岩井 真子	説明された様子： 不思議で変な奴や。リズム感ゼロ。		ファンダムはいましたか？ no
2009-05-11: 岩永 和子	説明された様子： 宇宙人の様子： 裸で恥窓がつきました。		ファンダムはいましたか？ yes

メールによっては、コンテンツのプッシュをサポートしています。Webサイトにアップデートがあると、emailメッセージを受け取ると同時にメッセージを受け取ることができます。



多くのWebブラウザも
プッシュ型コンテンツを閲覧することができます。これによりWebサイトにポストされた最新の情報を素
早く知ることができます。



携帯などのモバイルデバイスでも
プッシュ型コンテンツにアクセス
することができますから、Webサ
イト上で何かが変わったら自動的
に配信されます。

オーウェンはファンについての言葉が欲しいのです	534
UFOによる誘拐データを人々にプッシュする	535
RSSはWebコンテンツを人々にプッシュしてくれます	536
RSSというものは実はXMLです	537
データベースからニュースリーダへ	542
RSSを視覚化する	545
どうすればニュース即配信が可能となるか？	547
RSSフィードを動的に作る	548
RSSフィードへのリンク	552
百万の言葉より一枚のビデオを以て語らしめる	554
Webコンテンツを他のサイトから引っ張ってくる	556
YouTubeビデオとシンジケーションする	557
YouTubeビデオリクエストを作る	558
オーウェンはRESTリクエストを組み立てる準備が できました	562
YouTubeはXMLで話します	566
YouTube XML レスポンスを分解する	570
XMLのビデオデータを視覚化する	571
オブジェクトでXMLデータにアクセスする	572
XML要素からPHPオブジェクトへ	573
XMLデータをオブジェクトで掘り起こす	574
名前空間なしではダメです！	575
ファンダム目撃情報上昇中	577
ビデオを見やすく配置する	578
ビデオデータを表示用に配置する	579

付録 i

残飯

本編でカバーしなかったこと

全部終わった後ですが、まだ少し残っています。知っておくべきことはもう少しあると思います。それらを無視するというのは良いことではありません。ほんの少しだけでも説明しておくべきでしょう。そこで、本をしまう前に、以下を読んでください。PHPとMySQLに関する短いけれども大事な小話です。さて、この部分を読み終えたら、もういくつか付録がある…索引…、それと多分広告のページ…、そうしたら本当に終わりです。約束します！

本書の内容を PHP4 と MySQL 関数に後方互換で対応させる … 590



開発環境をセットアップする 作業をする場所

PHPとMySQLに関する真新しいスキルを練習するには、**Web上のデータが攻撃にさらされる心配などがない場所**が要ります。 PHPアプリケーションを世界中に(Webで)発信してしまう前に、まず安全な場所を確保しておくというのは非常に良いことなのです。この付録にはWebサーバ、MySQL、それとPHPをインストールする手順が書いてあります。安全な場所で作業や練習に取り組むことができます。

サーバコンピュータ



PHP 開発環境を作る	594
自分の持ち物を調べる	594
Web サーバがありますか？	595
PHP がありますか？バージョンは？	595
MySQL がありますか？バージョンは？	596
サーバをセットアップする	597
Windows で VertrigoServ を起動する	599
Apache と PHP の動作を確認する	600
MySQL の動作を確認する	600
Mac に XAMPP をインストールする	602
Mac で XAMPP を起動する	603
Apache と PHP の動作を確認する	604

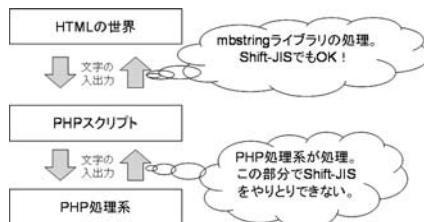


PHP & MySQL日本語対応

PHPとMySQLで日本語を扱う

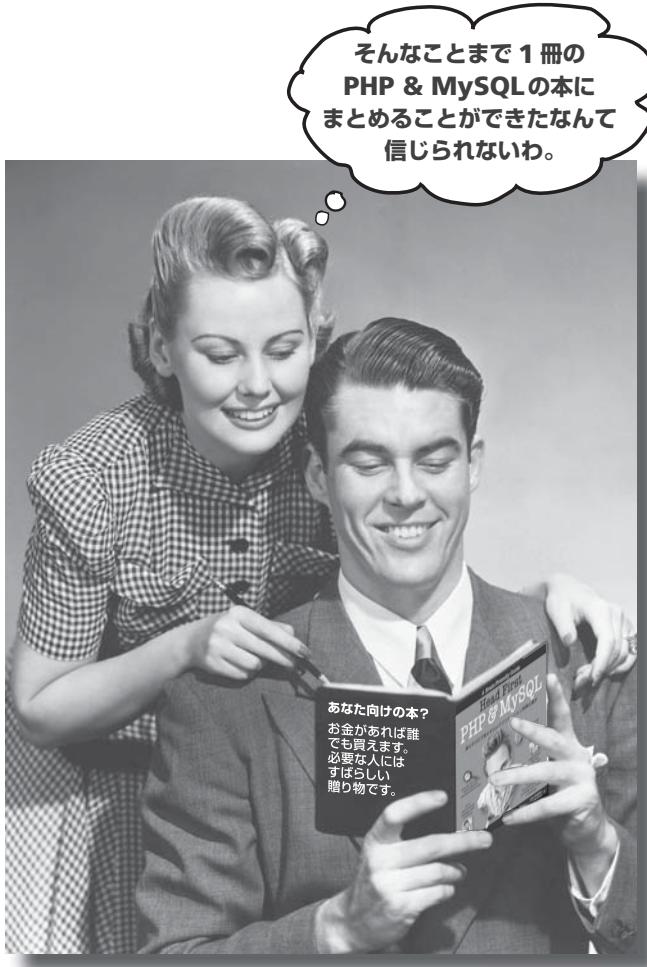
その通りです。PHPとMySQLでプログラミングをし、すばらしいWebアプリケーションを作ることができるようになりました。でも、まだやらなければならぬことがあります。ここは日本なのです。PHPとMySQLで日本語を取り扱えなければ話になりません。本文からも参照していますが、PHPとMySQLで日本語を扱う場合に、知っておかなければならぬことを、ここにまとめてあります。

PHP 日本語スクリプトを作る	608
UTF-8のファイルを作る	611
MySQL	612
文字コードセットを確認する	613
文字コードセットをUTF-8 だらけにする	614
日本語データを入力してみる	616
MySQL ターミナルで日本語を入力する場合の注意事項	617
SELECT問い合わせ文でカラムがずれます	617



この本の使い方

序章



本章では、「ではどうしてそんなことまで1冊の
PHP & MySQLの本にまとめることができたのか？」
というホットな疑問に答えます。

この本を読むのにふさわしい人は？

以下の質問のすべてに該当する方：

- ① HTMLまたはXHTMLの経験があるWebデザイナであって、Webページを次のレベルに持って行きたいと願っている方。
- ② 単純なHTMLページを自分で超えたいと思っている方。つまりPHPやMySQLを使ったWebアプリケーションの使い方を習得し、理解し、覚えるだけでは物足りない方。
- ③ 飲みながらの会話の方が、無味乾燥で退屈な学校の授業よりも刺激的で楽しいと思う方。

以上すべてに当てはまる方、この本はうってつけです。

本書に恐らくは向いていないであろうと思われる読者

以下の質問のいずれかに該当する方：

- ① 変数やループといった基本的なプログラミング概念を全く知らない方。
(ただし、仮に今までに一度もプログラミング経験がないとしても、本書によりキーとなる必須概念を習得することは、恐らくできるでしょう。)
- ② PHP Web開発のやっつけ仕事をしていて、リファレンス本を探している方。
- ③ 何か人と違ったことをするのにはためらいのある方。格子縞にストライプを合わせる位なら歯根治療(痛そ～)をしていた方がマシという方。技術書はまじめでなければならないと思っている方。UFOによる誘拐事件のデータベースを作るなどあり得ないと思っている方。

以上のいずれか1つでも当てはまる方、この本はお勧めできません。



[マーケティング部門からの注記：クレジットカードを持つている方、誰でもお勧めです。][†]

[†] 訳注：こういうつまらないオヤジギャグをかますマーケティング部門の人間はリストラ対象とした方が良いと思います。

みなさんが思っていることはわかっています

「こんなものがどうして真面目なPHPとMySQLの本なのか？」

「どうしてこんなに絵や写真が載っているのか？」

「こんなので本当に勉強できるのか？」

みなさんの「脳」がどう考えるかもわかっています

人間の脳は常に目新しいことを求めています。そして、いつも普通でないものを探し求めています。人間が生き延びるため、そもそも、脳はそのように作られているのです。

では、決まりきった日常的で普通なことに対して脳はどのように対処するのでしょうか？これらのことが重要なことを記録するという脳の本来の仕事を邪魔しないように全力を尽します。脳は退屈なことはわざわざ記憶しません。「明らかに重要でない」フィルタに必ず引っかかるのです。

では、脳はどのようにして重要なことを知るのでしょうか？例えば、ハイキングに出かけているときに、突然トラが襲いかかってきたとしましょう。頭と体の中では何が起こるでしょうか。

神経に火がつき、感情が昂り、脳内物質が活性化します。

そして、脳はこう判断するのです。

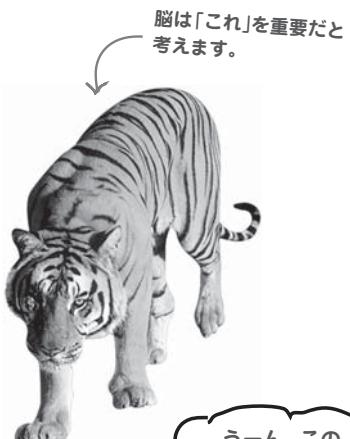
これは重要な違いない！絶対に忘れるな！

しかし、家や図書館にいる場合を想像してみて下さい。そこは安全で暖かいところです。トラ出没不能区域です。そこで試験勉強をしています。あるいは、上司が1週間から10日間かかると思っているような難しい技術課題を習得しようとしています。

ところが1つ問題があります。脳はあなたの役に立とうとします。明らかに重要でないことに貴重な資源が使われないように勝手にしてしまうのです。資源は本当に重大なことを格納するのに使われた方がよいわけです。つまりトラです。または火事の危機です。または今観ているUFOのYouTube動画のブラウザ画面を、上司にバレないように如何に素早く隠すか？です。

実は次のようなメッセージを脳に伝える簡単な方法はないのです。「拝啓、脳様。いつも感謝しています。脳にとっては、退屈で魅力がないことは分かっています。実際、私自身も地震計の針が激しく揺れるような感情もわき起こって来ません。でもこの本の内容を本当に覚えなければならないんです。」

YouTubeのUFO動画は、何らかの
コンピュータの本よりも明らかに
断然面白いものです。



脳は「これ」を重要だと
考えます。



脳は「これは」は
記憶するに値しない
と考えます。

「Head First」は読者を学習者とみなしています。

では、学習には何が必要なのでしょうか？まず習得し、その上で忘れないようにする必要があります。事実を頭に入れ込むことではありません。認知科学、神経生物学、教育心理学の最新の研究によると、学習には文字以外のものが必要です。この本は脳を刺激する方法を知っています。

以下Head Firstの学習原理です。

ビジュアル化します。視覚情報は、文字だけよりもはるかに記憶されやすいので、学習効果がかなり高まります（学習効果が最大89パーセント向上します）。もちろん、内容を分かりやすくできるというメリットもあります。本書では、文字を関連する絵や図、写真の中や近くに配置しているので、ページの下や別のページにまとめて載せるよりも内容に関する問題点を解決できる可能性が高くなるでしょう。

user_id = 1

個人に対する会話スタイルを採用しています。最近の研究では、形式ばった文体よりも対話形式で読者に直接語りかける会話的な文体の方が、学習後のテストで最大40%も成績がよいことがわかっています。本書は、くだけた言葉を使い、講義というよりは物語を語るような文体であります。あまり肩肘を張らずに楽な気持ちで読んで下さい。講義と楽しいディナーパーティでの仲間では、どちらの方に気持ちが向きますか？

エラー！
パスワードが
一致しません。

○
○



より深く考えるようにしています。つまり、自らの脳細胞を活発に動かさなければ、頭の中では何も起こらないのです。読者は、問題解決に対してやる気、真剣さ、興味、積極性を持って取り組み、結論を導き出すことで新しい知識を生み出す必要があります。そのためには、課題、練習問題、示唆に富む質問、そして脳と多くの感覚の両方を必要とする作業が必要です。

読者の注意を引き、注意をそらしません。「どうしてもこれを勉強したいけれども、1ページも読まないうちに眠くなってしまう」という体験は誰にでもあるでしょう。人間の脳は、普通でないもの、面白いもの、未知なもの、目を引くもの、予想外のものに注意を向けるようにできています。難しく新しい技術的トピックを学ぶことは必ずしも退屈とは限らないのです。退屈でなければ、脳はかなり短時間で学習できます。

感情に訴えるようにしています。何かを記憶する能力は、感情の動きに大きく左右されることがわかってきます。気になることや何かを感じたことは記憶しやすいのです。何も「少年と愛犬の涙と感動のストories」のことと言っているではありません。驚き、好奇心、楽しみ、「これは何だ？」といった感情や、「アハ！」のことを理解したときに沸き起こる「やったぞ！」という感覚のことを言っているのです。



若干の訂正。青年と彼の飼い犬に関する心に刺さる物語が出てきます。その犬はUFOに誘拐されてしまったのです。犬を見つけることで青年を助けてあげて下さい！



メタ認知：自分の思考について考える

何かを心から学習したいと思い、より早く、深く学習したいと考えるなら、何に注意すべきかについて注意深く考える必要があります。自分の思考についてよく考え、どう学習するかを学ぶのです。

自分の思考について認識することを「メタ認知」と呼びますが、学校でメタ認知について学ぶ人はほとんどいません。学習は要求されますが、学習の仕方を教わることはまずないです。

この本を手に取った人は、おそらくPHPとMySQLでデータベース駆動型のWebサイトを本気で作り上げたい人でしょう。しかし、勉強に長い時間をかけたいという人はいないはずです。さらに、本書で読んだことを利用したければ、読んだ内容を記憶しておく必要があります。そのためには、理解する必要があります。この本(あるいは勉強のために読むあらゆる本や学習体験)を最大限に活かすには、自分の脳に関して責任を負わなければいけません。脳が学ぼうとするように仕向ける必要があるのです。

脳をだますためには、学習している新しいことが「本当に重要な」ものだと脳に思い込ませる必要があります。つまり、学習する内容はあなたが幸せになるために重大なことであり、トラ同様に重要であると認識させるのです。それができないと、新しく学習した内容をどんどん忘れようとする脳と常に戦うことになります。



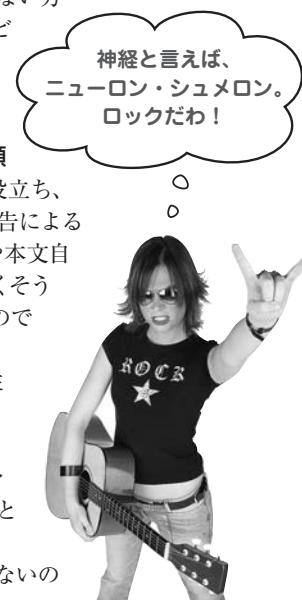
どうすれば脳はPHPとMySQLを腹ペコなトラと同じくらい重要と思うのか？

方法は2つあります。1つは退屈で時間のかかる方法、もう1つは効率的で時間のかからない方法です。「退屈で時間のかかる方法」とは、「同じことをひたすら反復する」というものです。どれほどつまらなく関心のないことでも、何度も繰り返し学べば覚えられるというのは誰しも経験していることでしょう。十分に繰り返せば、脳は「これは重要とは思えないのだが、これだけ繰り返しているのだからおそらく重要なのだろう」と判断するのです。

「効率的で時間のかからない方法」とは、「脳の働きを活性化する」(特に、さまざまな種類の脳の働きを活性化する)というものです。前のページに示した項目はこのために大いに役立ち、脳が望みどおりに機能するのを助けることがわかっています。例えば、いくつかの研究報告によると、絵や写真の中に内容を説明する言葉をいくつか入れておくと(それらの言葉を見出しや本文自体に入れるよりも)、その言葉と絵や写真との関係を知ろうとして神経細胞がより活発に働くそうです。神経細胞が活発に働くというのは、脳が注意を払うに値し、記憶するのに値するものであると考える可能性が増えるということです。

「会話スタイル」が効果的なのは、人は誰でも自分が会話の中に入っていると認識すると注意を向けるからです。なぜなら、話についてきて、会話における責任を果たすことが期待されるからです。面白いのは、脳は「会話」がみなさんと本との間で行われていても一向に構わないという点です！逆に、文章が形式張った無味乾燥なものであると、脳は大勢の受け身の聴衆と一緒に講義を受けている状態と同じように受け止めます。起きている必要はないと考えるのです。

しかし、脳の活性化に役立つのは、ビジュアル化や会話スタイルははじめの一歩に過ぎないのです…



Head Firstのアプローチ：

この本では絵や写真を多用しています。これは、人間の脳が、文字よりも絵や写真をよく認識できるようになっているからです。脳の観点から見れば、1枚の絵や写真は数千の言葉に匹敵するのです。また、テキストと絵や写真と一緒に使うときは、関係するテキストを絵や写真の中に埋め込む形で入れました。テキストがその対象となるものの中にいる方が、見出しや別の場所に埋もれているよりも脳がずっと効果的に働くからです。

また、同じことを繰り返し説明する、という工夫もしました。同じことを違った表現で言ったり、さまざまな媒体を使って人間のいろいろな「感覚」に訴えるようにしたりしました。こうすることで、学んだ内容が脳のさまざまな領域に記憶される可能性が高まるのです。

脳は目新しいものに向かうようになっているので、概念と絵や写真をみなさんにとって予想外の方法で使うようにしました。また、脳は感情の動きに注意を払う特性があるので、少なくとも何らかの感情に訴えるような形で使いました。その感情がちょっとしたユーモア、驚き、興味に過ぎなくても、何かを感じさせたものは記憶される可能性が高まるのです。

文章は読者個人に会話スタイルで書いてあります。脳は、受け身の姿勢で説明を聞いているときより会話をしているときの方が注意を払うようになっているからです。読む場合でも脳の働きは同じです。

脳は読んだ内容より実際に行ったことをよく記憶し学習するようになっているので、80個以上の練習問題を含めました。練習問題のレベルは「難しいが実行可能」というくらいにしてあります。そのくらいのレベルの問題を好む人が多いからです。

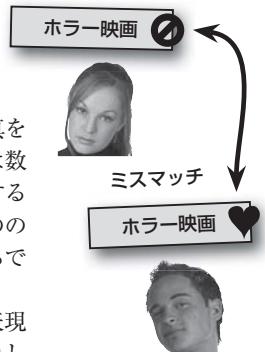
複数のアプローチで学べるようにという配慮もしています。基礎から順を追って学習する方がいいという人もいれば、とにかく、まずだいたいの要点をつかみたいという人、コード例が見られればそれでいいという人もいるからです。好みのアプローチはそれぞれに異なっていても、同じ内容を複数のアプローチで表現している本書なら誰もが満足できるはずです。

左脳と右脳の両方を使えるような工夫もしました。脳を使う箇所が多いほど、習得や記憶の可能性が高まり、集中力も持続します。一方の脳が働いているときには他方の脳は休めることが多いので、長期にわたって学習する際の生産性が上がります。

複数の観点を示す話や練習問題を含めるようにしています。脳は、評価や判断を強いられたときの方がより深く学習するようになっているからです。

練習問題を提供したり、必ずしも簡単な答えが出ないような質問をしたりすることで、課題が生じるようになっています。なぜなら、脳は何かに取り組む必要があるときに学習し記憶する特性があるからです。ジムで人を見ているだけでは体を鍛えることはできません。しかし本書では、努力すれば確実に力が付くように最善を尽くしています。難解な例を挙げたり、難しい専門用語を多用したり、逆にあまりに当たり前すぎる説明をしたりして、脳細胞を余計に使わせることはありません。

人を使うようにも留意しました。説明、例、絵や写真などに人を使っています。みなさんもやはり人なので、みんなの脳は物よりも人に注目するからです。





脳を服従させるために「みなさん」ができること

Head Firstとしてやるべきことはやりました。あとはみなさん次第です。以下のヒントは第一歩にすぎません。脳の言葉に耳を傾け、何が役に立ち何が役に立たないかを理解して下さい。新しいことを試してみて下さい。

このページのここから下を切り取って、冷蔵庫にでも貼っておくといいでしょう。

1 じっくり読みましょう。理解すればするほど、記憶すべきことは少なくなります。

読むだけではいけません。立ち止まって考えるようにして下さい。問題が出されたときに、すぐに答えに飛びつかないで下さい。誰かが本当に質問していると思いましょう。脳に深く考えさせればさせるほど、学習や記憶の効果が高まるのです。

2 問題を解きましょう。自分のノートに書き込んで下さい。

本書では問題を提供していますが、こちらで問題を解いてしまったら、みんなのために他人がトレーニングしているようなものです。練習問題を見ているだけではありません。実際に鉛筆を使って取り組んで下さい。学習中に体を動かすと学習効果が高まるという証拠はたくさん提示されています。

3 「素朴な疑問に答えます」のコラムを読みましょう。

このコラムはすべて大切です。これは内容を補足するものではなく、むしろ核心となる内容の一部なのです！これを読み飛ばさないようにしましょう。

4 本書を読んだ後は寝るまで他の本を読まないようにしましょう。少なくとも、難しいものは読まないようにしましょう。

学習の一部、特に学習内容の長期記憶への転送は、本を置いた後に起こります。脳がさらなる処理をするには時間が必要です。この処理時間中に新たに別のことを学習すると、前に学習したことが一部消去されてしまう恐れがあります。

5 水をたくさん飲みましょう。

脳は十分な水分がある状態で最もよく働きます。脱水状態（これはのどが渴いたと感じる前に起こります）になると認知機能は衰えます。

PHP と MySQL があれば世界で通用する本物の Web アプリケーションを作り上げることができます。是非本物の Web サーバーにアップロードして実際に使ってみて下さい。



6 内容について声に出して話してみましょう。

話すことは脳の違う部分を活性化します。何かを理解しようとしている場合や後で思い出しやすくしたい場合には、はっきりと声に出して話すようにして下さい。さらにいいのは、それを他の誰かに明確に説明してみることです。そうすると、より迅速に学習でき、読んでいるときにはわからなかった概念がはっきりするかもしれません。

7 脳が疲れたら休みましょう。

脳が過負荷状態にならないか注意してください。字面を追うだけになっていたり、今読んだばかりの内容を忘れてしまうようなら、休憩するべきです。ある限界を越えてしまうと、何を詰め込んでも学習効果は上がりず、逆にむしろ学習の妨げになってしまうことさえあります。

8 自分なりの感想を持ちましょう。

人間の脳は重要であるかどうかを知る必要があります。本を読むときはできるだけ感情移入をし、その世界に入り込むようにしましょう。写真や絵などに自分なりの注釈を入れるというのもよい方法です。くだらないジョークに文句を言うだけでも、何も感じないよりはよいでしょう。

9 たくさんのコードを書きましょう！

プログラムを習得する方法は1つしかありません。それは多くのコードを書くことです。本書ではそれを行っていきます。コーディングは技能であり、得意になるには実践するしかありません。本書では多くの練習問題を提供します。各章にはみなさんが解決すべき問題を提起する練習問題があります。それらの練習問題を飛ばさないで下さい。練習問題を解くときに多くのことを学べます。各練習問題には解答を示しています。行き詰ったら遠慮なく答えをのぞいてみて下さい（些細なことで引っかかることがあります）。とはいっても、答えを見る前に問題を解くように努力して下さい。また、次の内容に進む前に練習問題に取り組むようにして下さい。

まず最初に読んで下さい

Head Firstは、学習体験を提供するものです。リファレンス本ではありません。この本で取り上げるとかえって学習の妨げになる恐れのあるような事項については、あえて割愛しました。この本を初めて読む場合には、最初のページから順に読んで下さい。それまでに学んだことを前提に話を展開しているからです。

まず始めに単純なプログラミングの概念とデータベース接続の基礎を教えます。次により複雑なPHP関数やMySQL文について、最後に更に複雑なアプリケーションの概念について教えます。

ユーザがデータを追加したりデータを取ってきたりすることのできるWebアプリケーションを作ることとは確かに非常に重要です。しかしその前に、PHPとMySQLの両方について構文を理解しておく必要があります。そこでまずPHPとMySQLの文を自分自身で作ってみるところから始めます。そうすればPHPとMySQLを使ってすぐに何かを始めることができるようになり、わくわくしてくることでしょう。次にこの本の少し後の方で、良いアプリケーションおよびデータベース設計を実践的に示します。そうすれば、必要な構文について確かな手応えが得られ、コンセプトを習得することに集中できるようになります。

PHPとMySQLのすべての文、関数、キーワードなどをカバーするつもりはありません。

PHPとMySQLの文、関数、キーワードなどを、すべてこの本に収録しようと思えば、できたかもしれません。しかし、適度な厚さの本で、最も重要な文、関数、キーワードのみを教えてくれる本の方がよいと考えました。絶対に必要なものだけを厳選していますので、これさえあればあらゆる作業の95%は事足りるはずです。この本を読み終えた頃には、必要な機能が備わっていることについては確信を持っていることでしょう。自分で書き上げたイケてるアプリケーションを仕上げるのに十分だったのですから。

バージョンはPHP 5とMySQL 5.0をサポートしています。←

多くの人が今でもPHP 4や5を使っていますから、PHP 4や5や6といった特定のバージョンに依存したコードは可能な限り排除しています。ただし、ご自身ではPHPの5か6を、MySQLの5か6を使って、本書で勉強を進めて下さい。本書の執筆時点ではPHP 5およびMySQLの5をターゲットとしました。しかしコードは、これ以降のバージョンにも互換性があるよう配慮しています。

本書のプログラム、若干の修正を加えるだけでPHP 4に対応させることができます。付録iを参照して下さい。

PHPに対応したWebサーバが必要です。

PHPは、Webサーバ上にないと正しく動作しません。ApacheなどのWebサーバをご自身のパソコンにインストールするか、またはそのようなサーバにアクセスできる必要があります。同時にMySQLコマンドでデータを操作できる権限も必要です。この辺りのことは、付録iiとiiiを参照して下さい。PHPとMySQLのインストール方法および日本語対応に関して説明しています。

MySQLを使います。

SQLには標準的な言語が存在していますが、本書ではMySQLだけに焦点をあてて特定の構文を扱います。構文上は若干の変更が必要ですが、本書のコードはOracle、MS SQLサーバ、

PostgreSQL。DB2などを始めとする多くのリレーショナルデータベース管理システム (RDBMS) 上で動作するはずです。特定の PHP 関数と構文をチェックするだけで、他の RDBMS に移植することができます。もしすべての DBMS コマンドの構文を本書でカバーしようとしたら、今より遙かに分厚い本になってしまいます。森林伐採を食い止める意味からも、MySQL のみに焦点を合わせることにしました。

皆さんへの課題はオプションではありません。

エクササイズなどの課題は後付けのオプションではありません。これらは本書の核となるコンテンツの一部です。記憶に残るようにするものもあれば、理解を深めるためのものもあれば、復習のためのものもあります。エクササイズをスキップしてはいけません。クロスワードパズルはやらなくても構いません[†]。しかし、今覚えた用語を別の言い方で学び直すというのは、脳にとって良い刺激となります。

冗長は意図的なものであり重要です。

Head First シリーズの特徴の1つは、私たちが皆さんに**本当に**モノにして欲しいと思っているということです。そしてこの本を読み終えたら、学んだことを覚えておいて欲しいと思っています。多くのリファレンス本では、覚えてもらうことや思い出してもらうことをゴールに設定していません。しかし、本書は学習するためのものですから、いくつかの概念については何度も出てきます。

サンプルのうちいくつかは、完熟の
Web アプリケーションと呼べるもの
で、とてもパワフルなことができます。

サンプルコードは可能な限り余分なものをそぎ落としています。

200行もあるサンプルコードを読まされて、本当に理解する必要があるのはたったの2行だけだったというのでは非常に不愉快です。この本のサンプルは文脈に合わせて必要最小限に留めていますので、明快かつ簡潔にサンプルから学ぶことができるようになっています。すべてのサンプルが超堅牢で完璧だとは思わないで下さい。あくまで学習のために書かれたコードなので、あらゆる機能が備わっているわけではありません。

Web 上にすべてのサンプルの完全なコードとアプリケーションを公開しています。必要な部分をコピーしてテキストエディタや MySQL ターミナルで使うことができます。また完成版のスクリプトについては、そのままご自身の Web サーバに上げてテストすることもできます。この本のウェブサイトは以下の通りです。

<http://www.oreilly.co.jp/books/9784873114446/>

「頭の体操」というエクササイズには答えがありません。

この名前のエクササイズには正解と呼べるものがない場合があります。正解がある場合でも、「頭の体操」による思考過程は、学習の一部であり、自分の回答がどのような場合に正解かを決定することになるのです。「頭の体操」エクササイズには、正しい方向へ導くためのヒントとなっているものもあります。

[†] 訳注：本書ではページ数の制約からクロスワードパズルはすべて割愛しましたが、このクロスワード超難解です。従来 Head First シリーズ日本版ではクロスワードの質問文のみ翻訳していました。しかしこのようなクロスワードを解くのは至難の技です。日本語で書かれた質問の答に相当する英単語を考えなければならないのです！

テクニカルレビューチーム

Jereme Allen



David Briggs



Will Harris



Stephanie Liese



Steve Milano



テクニカルレビュー担当：

Jereme Allenは、上級Web開発者で、最新技術を利用したWebアプリケーションを構築する経験の持ち主です。9年以上のPHPとMySQL使用経験があるとともに、他のフレームワーク、OS、プログラム言語、および開発環境の経験もあります。

David Briggsは、技術書の著者であると同時にソフトウェアのローカリゼーション担当エンジニアで、英国のバーミンガムに住んでいます。彼は極端に巧妙に作られたソフトウェアの使い方についてユーザをどのようにガイドしようかと根を詰めて考えていないときは、外に出て妻のPauletteと愛犬のCleoと一緒に近所の公園に行くことが何よりの楽しみです。

Will Harrisは、IT部門に籍を置いていて、4つの大陸にある11の会社に対してサービスを提供しています。ラスベガスPASS(プロフェッショナル・アソシエーションSQLサーバ)支部の副代表でもあります。夜になると、電話ボックスに駆け込み自分のWeb 2.0統合環境につなぎ、Powered by By Geekでデザイナや開発者の手助けをし、MySQLとRailsを使ったそのデータプラットフォームが拡張性、ポータビリティ、メンテナンス性に富み、かつ速いものであるよう努力しています。また、奥さんのHeather、可愛い子供たちMaraとEllie、それに愛犬のSwiperと楽しいひとときを過ごすのが好きです。

Stephanie Lieseはカリフォルニアのサクラメントで技術トレーナーとWeb開発をしています。彼女はたいていの場合、標準に準拠したコードを褒めちぎっているか、またはCSSのレイアウトをデバッグしているのですが、それ以外の時はヨガのクラスにいて一汗かいていることでしょう。

Steve Milanoは、普段Day Job™用のコードを書き続けているか、またはパンクなロックバンドOnion Flavored Ringsのメンバーとして、空気のよどんだどこかの地下室で演奏しています。

Harvey Quamen



Chris Shiflett



そうでなければ恐らく家にいますが、猫の同僚Ralphも、人間の同僚Biancaも顧みずノートPCに向かっていることでしょう。

Harvey Quamenはコンピュータプログラミングのキャリアを諦め、トレーニングで、衆目を集められ、プロフィール上の評価も高いアカデミックな世界に身を投じました。彼は現在アルバータ大学の准教授であり、英語と人文科学コンピューティングを教えています。コースは、サイバーカルチャー、20世紀文学、およびWeb開発であり、その中にPHPとMySQLがあります。

Chris Shiflettは、OmniTI社のCTO(技術最高責任者)で、Webアプリケーションのセキュリティを実践するリーダーであり、Web開発のイニシアチブをとっています。彼はPHPおよびWebアプリケーションのセキュリティコミュニティにおける想像力豊かなリーダーであり、shiflett.orgのベストプロガーの1人でもあります。さらに世界中のインダストリーカンファレンスでも有名な講演者であり、PHPセキュリティコンソーシアムの設立者でもあります。著書には*Essential PHP Security*(O'Reilly)や*HTTP Developer's Handbook*(Sams)などがあります。

謝辞

担当者：

Brett McLaughlinには非常に感謝しています。的確な図を用いた尊敬すべきセッションのおかげで方向性を見失わずに済みました。また、自分が理解することに対して冷酷とも言える程の約束を果たしてくれたことにも感謝します。

この本は **Sanders Kleinfeld** の英雄的な努力と実践と忍耐がなければ、完成することはなかったかも知れません。彼はいつもボールを取ってきてくれました。猫だったこともあります。私たちがジャグリングをしていて不意に1つ(3つのときもありました!)それを落としてしまうからです。それについても感謝しています。このプロジェクトと同等以上の難しいプロジェクトに彼が投入されてしまう前に、少しの間でも彼に休みが取れることを願っています。

オライリーのチーム：



↑
Lou Barr

Lou Barr の驚異的なデザインのスキルに感謝します。おかげでこの本のビジュアルは格段に向上しました。

Brittany Smith にも最後の最後まで激務をこなしてくれたことに感謝します。 **Caitrin McCullough** は、例題用の Web サイトをセットアップし、動かしてくれました。 **Laurie Petrycki** は、私たちを信頼してくれて2冊目の Head First を書く機会を与えてくれました。



↑
Brett McLaughlin



↑
Sanders Kleinfeld

更に感謝：



最後に、**Elvis Wilson** には非常に感謝しています。11章で使う YouTobe ビデオを作ってくれました。すばらしいできです！彼が単に無骨なアートディレクタのように見えるだけにおさらです。

1章 静的なページに命を吹き込む

ページは生き物だ

ちょっとワイフを
呼んでみてくれないか。
退屈しのぎだよ…。

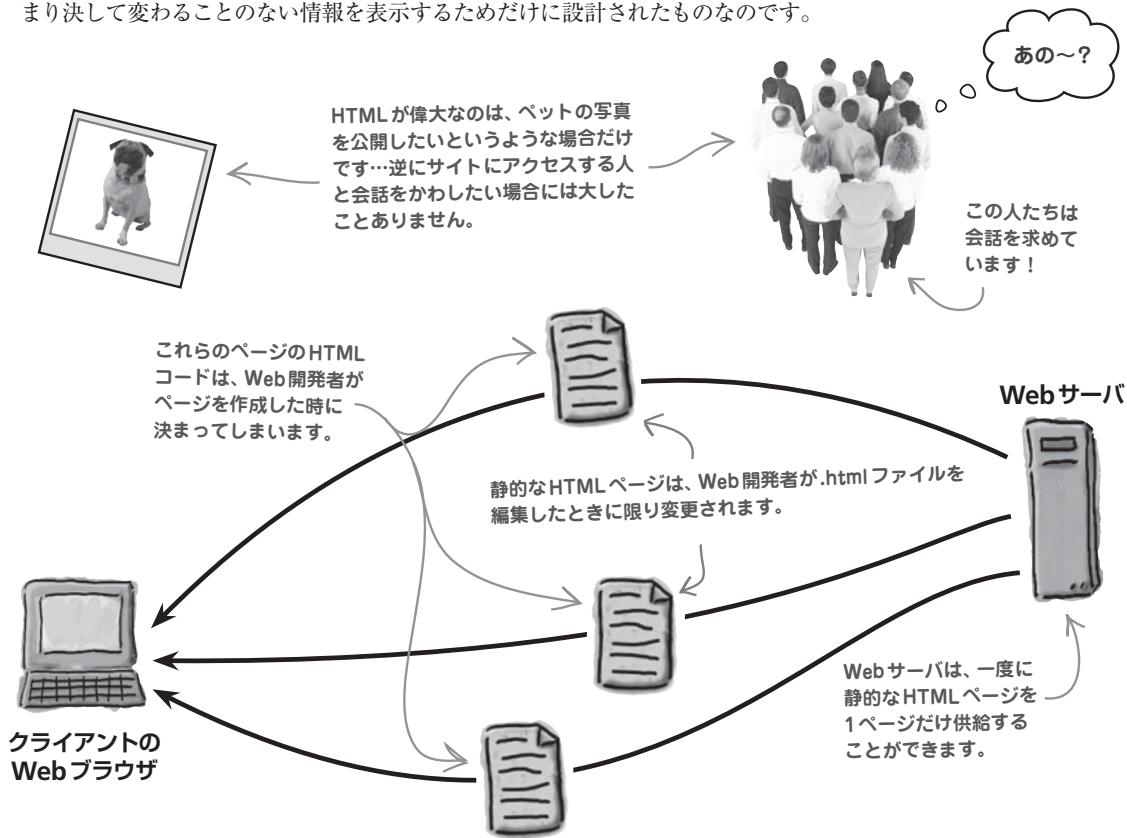


これまでHTMLにCSSをちりばめて多くのWebページを作ってきたことでしょう。

しかしすでにお気づきのように、できあがったサイトにアクセスしてくれる人は、受身です。つまりページの内容を見ること以外には何もできないのです。ここでのコミュニケーションは一方通行です。これを見てみたくありませんか？ 実際、サイトを見てくれている人が何を考えているのかを知りたいはずです。そのためには、その人たちの考えを理解するためにWebフォームに情報を入力させるようにする必要があります。そしてその情報を処理し、かつ自分のところに持ってくる必要があります。つまりサイトを単なるHTMLから次のレベルに引き上げる時が来たとは思いませんか？

HTMLは静的で退屈です

HTMLはWebページを作成する上で偉大であり、そのことについてはすでによく知っています。でもWebページ上で実際に何かをやる必要がある場合はどうでしょう？例えばデータベースを検索する必要があるとか、メールを飛ばす必要がある場合は、どうすればいいでしょうか？HTMLではイマイチです。あまり生き生きとした言語ではありません。つまり決して変わることのない情報を表示するためだけに設計されたものなのです。



Webサーバは、生き生きとしていないHTMLを使っている限り、退屈な配達メカニズム以外には何も供給することができないという大きな問題を抱えていることになります。ブラウザがページを要求すると、サーバがHTMLを返す。これで終わりです。Webサイトを会話型のWebアプリケーションに変えるためには、Webサーバは新しいもとダイナミックな役割を担う必要があります…その役割は多分PHPで果たすことができます。

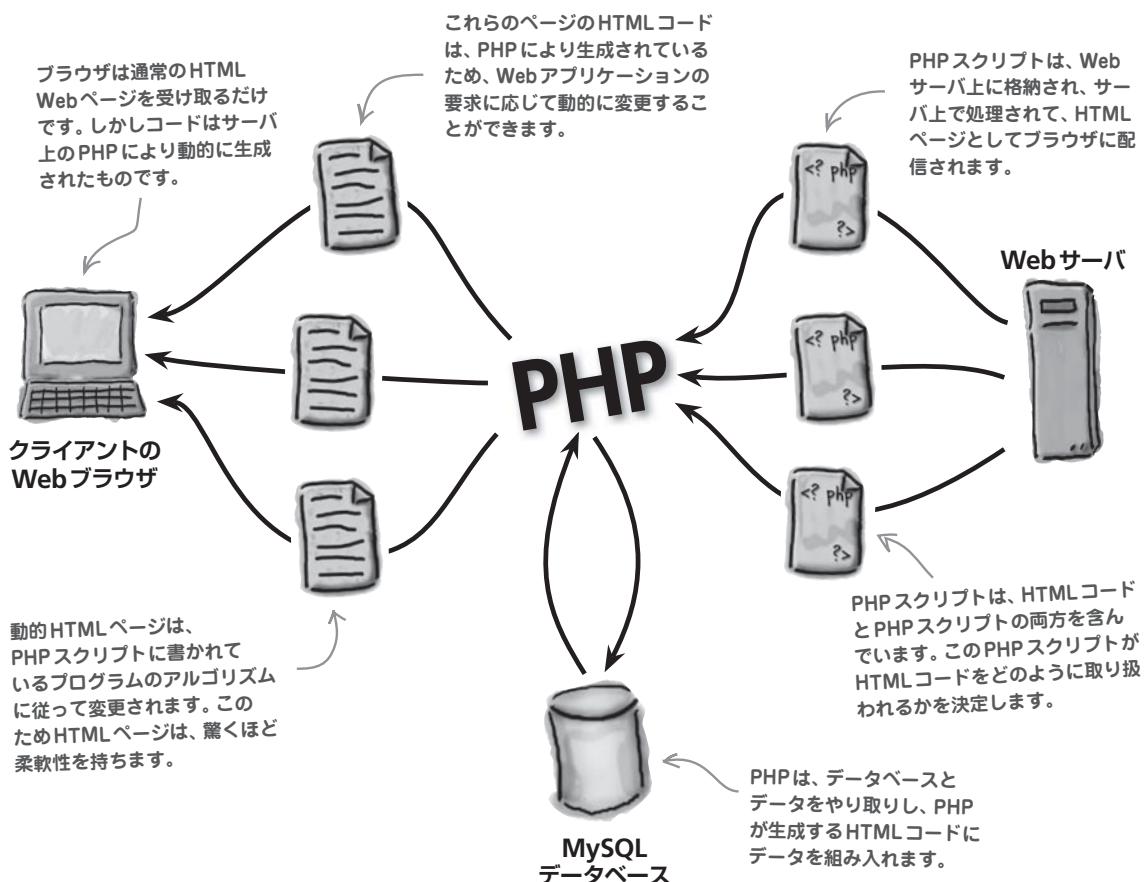
純粹なHTMLによる
Webページでは、サーバが
内容を表示するだけの
静的なHTMLを供給する
ことしかできません。

PHPはWebページに命を吹き込んでくれます

PHPを使えばクライアントのブラウザにページを配信する直前にサーバ上のWebコンテンツを操作することができます。次のように動作します。

PHPスクリプトはサーバ上で動作し、HTMLコードを意のままに変更したり、生成したりすることができます。ブラウザに配信されるのはHTMLのWebページです。ブラウザ側はサーバ上でPHPが処理を行っているというHTMLの微妙な変更についてわかりませんし、また気にする必要もありません。

PHPを使えば、
Webサーバは要求に
応じて動的にHTMLの
Webページをその場で
生成することができます。



宇宙の犬

オーウェン (Owen) を助けてあげて下さい。オーウェンの犬ファング (Fang) が迷子になりました。しかし、犬を見つけるには単に近所を探せばよいと言うような問題ではないのです。なぜならファングはUFOに誘拐されたからです。つまりオーウェンは全宇宙を探さなければなりません。オーウェンはHTMLとCSSを多少知っているので、Webサイトをカスタマイズして問題解決に役立てられないかと考えています。つまり他にもUFOに誘拐された経験を持つ人と情報を共有しようというわけです。

しかし、他の人から情報を得るためにには、オーウェンはユーザの入力、しかもたくさんの人からの入力を受け取ることのできるようなWebフォームが、さらにそのような入力があったことを知らせてくれるようなしきけが必要となります。でも大丈夫。HTMLにはそのようなWebフォームをサクッと作るための豊富なタグが用意されています。



詳しいことはわかりませんが、
ファングが光線で空に
連れ去られたことは確かです。

尋ね犬です。



オーウェンは、HTMLとCSSを
多少知っているので、Webを使って
犬の手がかりを得ることができる
のではないかと考えています。

フォームにより全ストーリーを得られる

オーウェンの新しいサイト AliensAbductedMe.com は、オーウェンと UFO に誘拐された経験を持つ人とを結びつけることを目的としています。この人たちのおかげで ファング蒸発事件に何か光明を見いだすことができるかもしれません。オーウェンは、 HTML フォームを使ってサイト閲覧者から誘拐のストーリーを聞き出す必要があることはわかっています。そしてその人の星間旅行の間にファングとの関わりがあったかどうかを聞き出さなければなりません。しかしサイトを立ち上げて動作させるには助けが必要です。これがオーウェンが考えているフォームです。

UFOに誘拐された！？「誘拐レポート」を！

UFOに誘拐された時の様子を教えてください。

姓：
名：
emailアドレスは？
いつですか？
何時間位ですか？
何人見ましたか？
どんな奴らでしたか？
何をされましたか？
飼い犬ファングを見ましたか？ はい いいえ

何か他にコメントは？

誘拐レポートの送信

このフォームは純粋な HTML としては100点満点です！

オーウェンの HTML フォームをどう思いますか？

このフォームを使って UFO による誘拐に関するデータを集めようとした場合、オーウェンが直面するであろう問題点を思いつきますか？自分の考えをメモしてから先に進みましょう。

フォームはHTMLでできています

オーウェンの誘拐レポートフォームは、すべてHTMLタグと属性で作られています。ほとんどの質問の回答用にテキストフィールドが用意され、閲覧者がファンダを見たかどうかを見分けるためのラジオボタンがあり、追加コメント用のテキストエリアが用意されています。そしてフォームはオーウェンのメールアドレスにデータを飛ばすようセットアップされています。

「mailto」はフォームのデータをメールで飛ばすためのプロトコルです。

```
<h2>UFOに誘拐された！？「誘拐レポート」を！</h2>
<p>UFOに誘拐された時の様子を教えて下さい。</p>
<form method="post" action="mailto:owen@aliensabductedme.com">
  <label for="lastname">姓:</label>
  <input type="text" id="lastname" name="lastname" /><br />
  <label for="firstname">名:</label>
  <input type="text" id="firstname" name="firstname" /><br />
  <label for="email">emailアドレスは?</label>
  <input type="text" id="email" name="email" /><br />
  <label for="whenithappened">いつですか?</label>
  <input type="text" id="whenithappened" name="whenithappened" /><br />
  <label for="howlong">何時間位ですか?</label>
  <input type="text" id="howlong" name="howlong" /><br />
  <label for="howmany">何人見ましたか?</label>
  <input type="text" id="howmany" name="howmany" /><br />
  <label for="aliendescription">どんな奴らでしたか?</label>
  <input type="text" id="aliendescription" name="aliendescription" size="32" /><br />
  <label for="whattheydid">何をされましたか?</label>
  <input type="text" id="whattheydid" name="whattheydid" size="32" /><br />
  <label for="fangspotted">飼い犬ファンダを見ましたか?</label>
  はい <input id="fangspotted" name="fangspotted" type="radio" value="はい" />
  いいえ <input id="fangspotted" name="fangspotted" type="radio" value="いいえ" /><br />
  <br />
  <label for="other">何か他にコメントは?</label>
  <textarea id="other" name="other"></textarea><br />
  <input type="submit" value="誘拐レポートの送信" name="submit" />
</form>
```

フォームは、開始と終了を示す
<form>タグで囲われます。

ここは驚くに値しません。フォームは
100%純粋にHTMLコードです！

HTML フォームを作成する上で
復習が必要な方は、『Head First
HTML with CSS & XHTML』[†]
Chapter 14を参照して下さい。

オーウェンは、このメールアドレスに飛んでくる
フォームの内容をみることができます。
オーウェンのメールアドレスを自分のものに
書き換えてフォームのテストをしてみて下さい。

この値は、データの送り方をサーバに
伝えます。「post」または「get」の
いずれかです。これらの違いについては
少し後で説明します。

inputタグは、フォームに情報入力が
想定されていることを伝えます。

type属性は、フォームの
アクションとしてtextが
想定されていることを伝えます。

「提出」(submit)ボタンは、
フォームにアクションを
実行させることを伝えます。

[†] 編注：日本語版はないので、代わりに『HTML & XHTML 第5版』をおすすめします。



試運転

誘拐レポートフォームを試してみる

誘拐レポートWebページをWebサイト<http://www.oreilly.co.jp/book/9784873114446>からダウンロードします。ページはch01フォルダにあります。

フォルダにはオーウェンのWebフォームreport.htmlとスタイルシート(style.css)とファングのイメージ(fang.jpg)が入っています。

テキストエディタで report.html を開きオーウェンのメールアドレスを自分のアドレスに書き換えます。次に Web ブラウザでこのページを開き、UFO による誘拐に関する情報をフォームに入力し、最後に「誘拐レポートの送信」ボタンをクリックします。



UFOに誘拐された！？「誘拐レポート」を！

UFOに誘拐された時の様子を教えてください。

姓：	石井
名：	秀樹
emailアドレスは？	hideki@theyrealgreen.com
いつですか？	今年の1月
何時間位ですか？	11時間
何人見ましたか？	たくさん
どんな奴らでしたか？	小さくて緑色です。
何をされましたか？	UFOのレギュレーションについて聞かれた。
飼い犬ファングを見ましたか？	はい ① いえ ②

何か他にコメントは？

フォームを「提出」すると、フォームに書き込まれたデータがメールで飛びはずなのですが…

HTML フォームは、メールメッセージをどうやって送信すれば良いのかを実は知らないので、ユーザが使っているメールにその仕事を依頼しているのです。

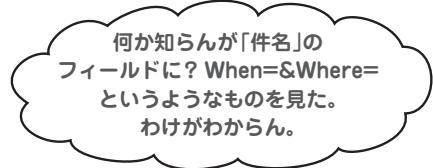
新規メッセージ

ユーザがこのわけのわからない
メールを実際に送信してくれなければ、
フォームデータは、オーウェンには
届かないのです。

さて、どう思いますか？フォームデータをメールで自分自身の受信フォルダに受け取ることができましたか？

HTMLフォームは問題を抱えています

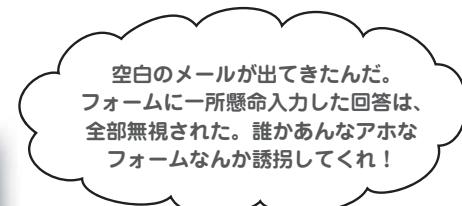
オーウエンの「誘拐レポート」フォームは、アップされ動作していますが、ユーザから大した情報は得られませんでした。ファンゲの誘拐は、あまりにもまれな事件だったのでしょうか…それともフォームの方に何か問題があったのでしょうか？ユーザが何を言っているのか聞いてみることにします。



何か知らんが「件名」の
フィールドに? When=&Where=
というようなものを見た。
わけがわからん。



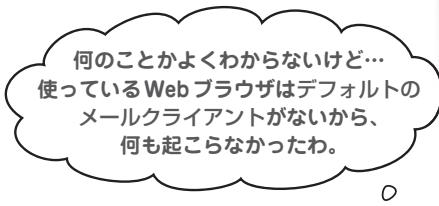
ボタンをクリックしたら
メーラ Outlook が立ち上がったけど、
中身は何もなし。私はフォーム入力の
15分を無駄にしたわ！



空白のメールが出てきたんだ。
フォームに一所懸命入力した回答は、
全部無視された。誰かあんなアホな
フォームなんか誘拐してくれ！



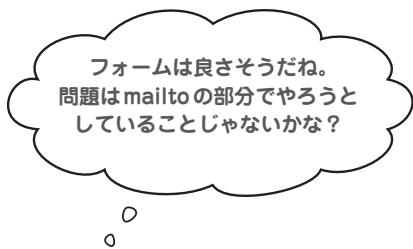
何故かオーウェンのフォームは、サイトの閲覧者から情報よりもフラストレーションの方を多く抽出してしまったようです。



何のことかよくわからないけど…
使っているWebブラウザはデフォルトの
メールクライアントがないから、
何も起こらなかつたわ。



何が起こっていたのでしょうか？フォームを改修するための良いアイデアはありませんか？

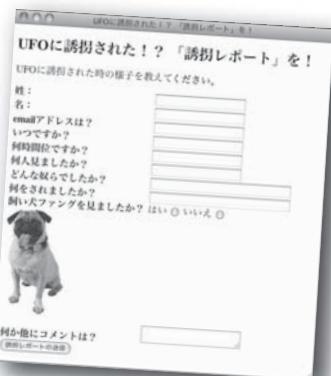


**そうです。HTML フォームのコードに問題はありません。しかし
mailtoでフォームのデータを配信するのは良い方法ではありません。**

オーウェンのフォームは、「誘拐レポートの送信」ボタンを押すまでは完璧だったのです。その直後、フォームのデータをパッケージしてmailtoに渡してしまったのです。しかしこのメールは自動的には飛びません。その代わりにデータはユーザが使っているパソコンのデフォルトメールに埋め込まれたのです。そして本当に飛ばすのは…ユーザです。データを送ってもらうためには、ユーザがメールでデータ送るという操作をしなければならないのです。つまりメール配信に関して、サーバ側でコントロールすることはできないのです。結局 Web フォームから Web ブラウザを通してメールを起動しメッセージを受け取るというもくろみが成功するかどうかはわからないのです。全然ダメです。

Web フォームの配信をコントロールする仕掛けが必要です。もっと詳しく言えば、PHPを使ってフォームデータをメールのメッセージにパッケージする必要があります。これはクライアント側(HTML, mailtoなど)からサーバ側(PHP)に目を向けるということを意味しています。

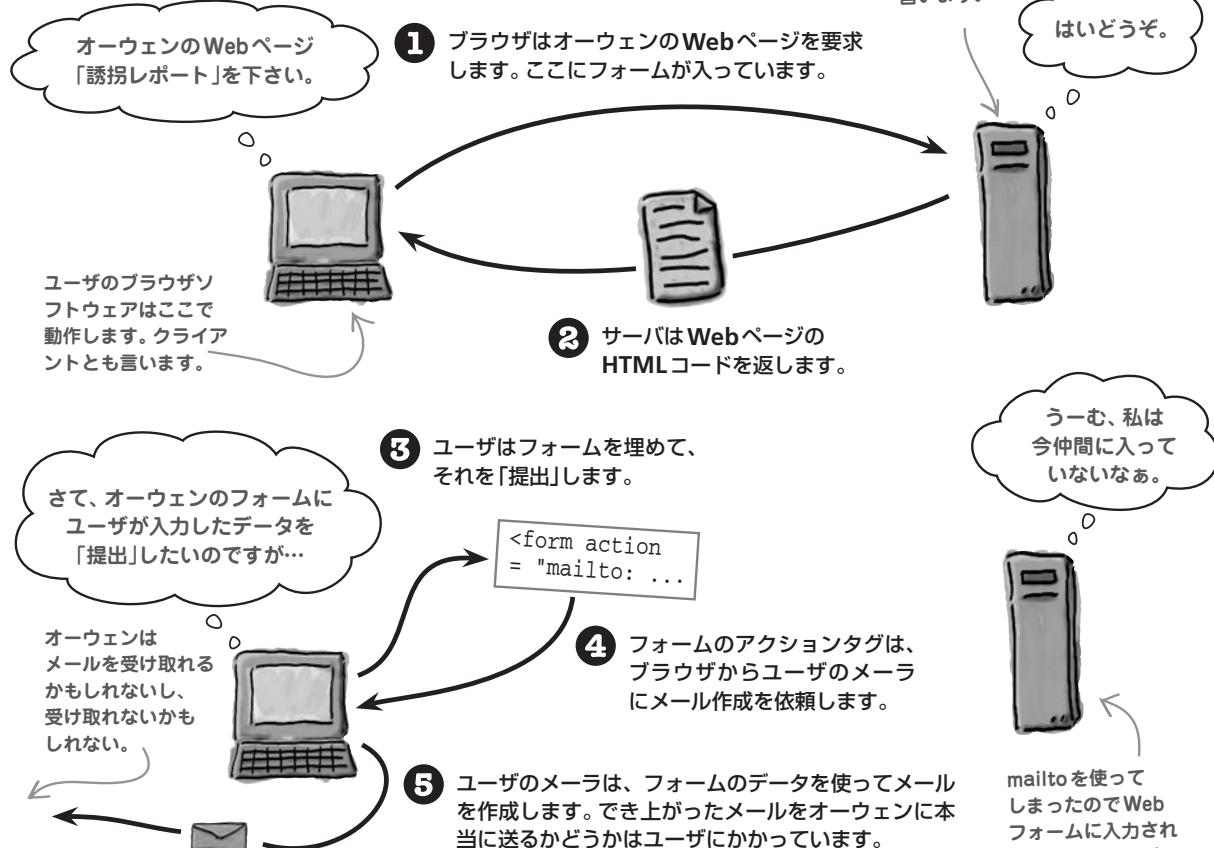
フォームはすばらしかった
「誘拐レポートの送信」を
クリックするまでは…そこで
すべてがバーになった！



HTMLはクライアント上で動作します

オーウェンのフォームは純粋にHTMLで書かれていて、mailtoフォームアクションを使ってフォームのデータをメール送信しようとします。report.htmlというWebページはWebサーバから送られてくるとはいえ、フォームを埋め、処理を進行させるのはすべてユーザのWebブラウザ上で行われます。

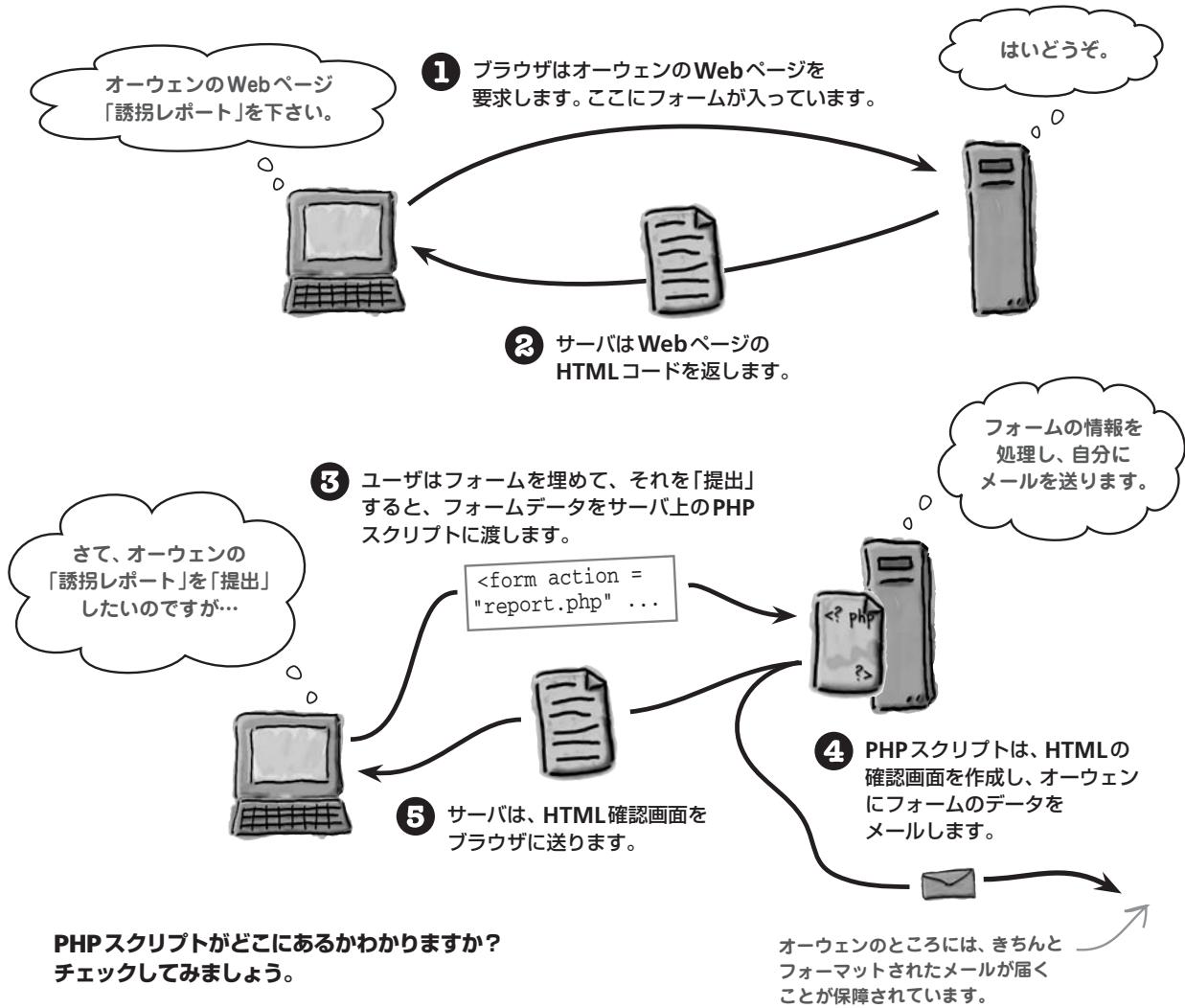
オーウェンのWeb
サーバソフトウェアは、
ここで動作します。
サーバとも
言います。



ここでのサーバの役割は非常に限られています。ブラウザにWebページを単に配信するだけです。ユーザがフォームを「提出」しても、ブラウザ(つまりクライアント!)は、自身のデバイスに留まつたままで、何とかしてメールを使ってフォームのデータを送り出そうとしています。クライアントにはフォームのデータを配信する機能は備わっていないのです。それはサーバのやるべき仕事です。

PHPはサーバ上で動作します

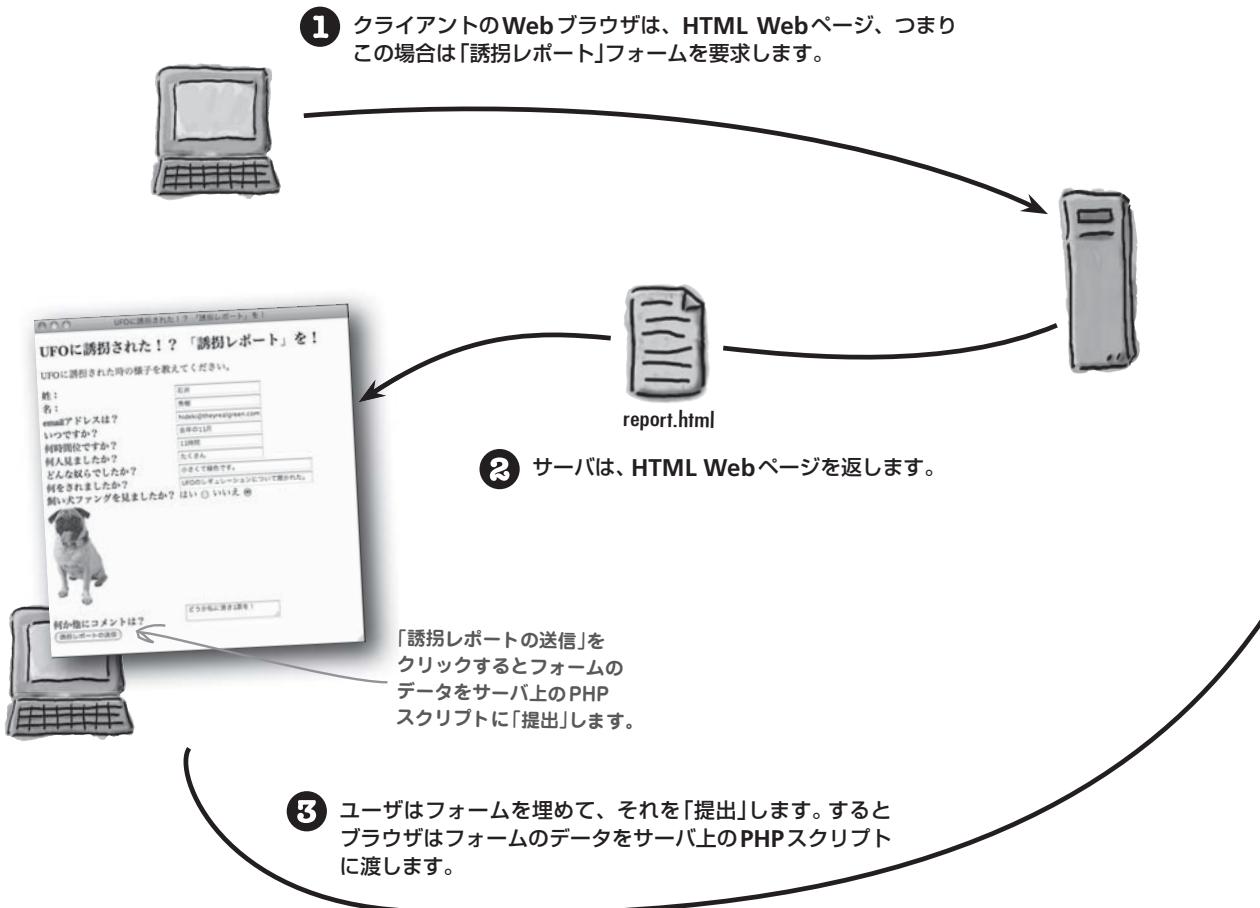
PHPを使えばユーザがフォームに入力したデータをメールで飛ばすまでの動作を一貫してコントロールできます。ユーザは誘拐のストーリーをフォームに入力し、「誘拐レポートの提出」ボタンを押せば終了です！ PHPコードはメールメッセージを生成し、メールを送り、最後にユーザへの確認画面を作成します。



PHP スクリプトはサーバで動作します

PHP コードはサーバで動作し、PHP スクリプトに格納されています。PHP スクリプトには通常 .php というファイル拡張子を付けます。PHP スクリプトは、HTML コードも CSS コードも含んでいるため、普通の HTML Web ページとそっくりに見えるものです。実際、サーバが PHP スクリプトを動作させた場合、最終結果は常に純粋な HTML と CSS になります。つまりすべての PHP スクリプトはサーバ上での動作を終えると、最終的には HTML と CSS に姿を変えるのです。

PHP スクリプトがオーウェンの Web フォームにどのように変わっていくか、もう少し細かく見てみましょう。



PHPはサーバ側にあるプログラミング言語です。 つまりWebサーバ上で動作します。

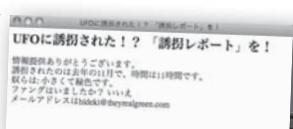
- ⑤ サーバは純粋なHTML Webページを返しますが、これは PHPスクリプトにより生成されたものです。

PHPスクリプトは
サーバ上で動きます。



ページの名前は、ブラウザ上では.phpで
終わる名前として表示されますが、この時点
では純粋なHTMLになっています。

- ⑥ ブラウザはWebの確認画面
を表示します。



ユーザはWebの確認
画面を目になります。

- ④ サーバはPHPスクリプトを動作させま
すが、この時メールを飛ばし、HTMLで
Webの確認画面を生成します。



- ⑦ メールがオーウェン
の受信ボックスに
配信されます。

オーウェンはこのメールを
受け取ります。





これが
PHPスクリプトの
ファイル名です。

ねえでも、実際に
PHPスクリプトをサーバ上で
動作させているところは
どこなの？

フォームの要素action属性は、フォームとPHPスクリプトとを結びつける
働きがあり、フォームが「提出」された際にスクリプトを実行させます。

フォームはHTMLの<form>タグを用いて作られていて、すべての<form>タグは
action属性を持っています。action属性に設定したファイル名がどんなものであって
も、Webサーバは「提出」されたフォームを処理するために使います。オーウェンのPHP
スクリプトがreport.phpという名前ならば、これをフォームと結びつける<form>タ
グは次のようにになります。

```
<form action = "report.php" method = "post">
```

ユーザがフォーム中の「Report Abduction」ボタンをクリックすると、フォームのアクション
はサーバ上でreport.phpスクリプトを動作させ、フォームのデータを処理します。

その前にもう1つ！コード系を変更します[†]。

metaタグ内にcharset=shift-jisという指定がありますが、これをUTF-8に変更
します。同じく変更したファイルを保存する際には、文字コードをUTF-8にしてから保
存して下さい。日本語を表現するための文字コードにはJIS、Shift-JIS、EUC、UTF-8な
ど、いくつかの種類があります。WindowsではShift-JISが標準的に使われていますが、
PHPはShift-JISには対応していません。

charset=UTF-8に
変更します。PHPは
Shift-JISでは正常に
動作しません。

<form>タグのaction
属性が、フォームが「提出」
された時にサーバ上のPHP
スクリプトを動作させる
働きをします。



report.html

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>UFOに誘拐された！？「誘拐レポート」を！</title>
<link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
<h2>UFOに誘拐された！？「誘拐レポート」を！</h2>

<p>UFOに誘拐された時の様子を教えて下さい。</p>
<form method="post" action="report.php">
<label for="lastname">姓:</label>
<input type="text" id="lastname" name="lastname" /><br />
```



report.php

[†] 訳注：PHPで日本語文字を処理するための必須事項ですので追記しました。

素朴な疑問に答えます

Q: PHPって何の略ですか?

A: PHPというのは、もともとはPersonal Home Pages(個人用のホームページ)の頭文字でした。いつの頃からかPHP: Hypertext Processor(PHPとはハイパーテキストプロセッサ)に変わりました。こちらは再帰的な頭文字ということができます。つまり頭文字が自分自身を参照しています。頭文字(PHP)が頭文字の中にあります。賢い? 悩ましい?ご自分で決めて下さい!

Q: Webブラウザは、Webページが.phpで終わることを表示していますが、それでも純粹なHTMLなのでしょうか? どうなっているのでしょうか?

A: 大丈夫です。ページはもともとサーバ上のPHPコードから始まりますが、ブラウザに行くまでの間にHTMLコードに形を変えるのです。言い換えるとサーバはPHPコードを実行し、これを閲覧用にブラウザに向けて送信する前にHTMLコードに変換します。これは.phpファイルがPHPコードを含んでいても、ブラウザはそのコードを見ることはない、ということを意味しています。ブラウザにはサーバ上で動作したPHPコードの実行結果としてのHTMLコードだけが見えます。

Q: でも、.htmlファイルの純粹なHTMLページであつても、すべてのWebページがサーバ上に置かれているわけではないでしょう?

A: 違います。Webサイトのすべてのページは.htmlも.cssも.phpも(その他も)サーバ上に格納されています。しかしすべてがサーバで処理されるわけではありません。HTMLファイル、CSSファイルは(イメージファイルもそうですが)クライアントブラウザに直接送られます。このとき実際に中身がどうなっているのかを気にしません。PHPファイルが異なるのは、Webサーバ上で処理され、実行されるコードを含んでいるという点です。ブラウザに送られるのは、PHPコードではありません。送られるのは、PHPコードを実行した結果なので、この結果というのは純粹なHTMLとCSSなのです。

Q: PHPはShift-JISに対応していないってどういうことですか?[†]

A: PHPを記述しているソースの文字コードとしては、PHPは日本独自のコード系Shift-JISを知らないのです。欧米で開発されたプログラムでは、よくあることです。アルファベット1文字が1バイトで表現可能なのに対し、日本語の場合1文字について通常2バイト以上の領域が必要となります。Shift-JISの場合この2バイト目が問題で、PHPはShift-JISの2バイト目だけを見て別の文字と勘違いして発狂してしまうのです。少し後で例を示します。ただしPHPが取り扱うデータとしてならShift-JISでも大丈夫です。つまりPHPスクリプトの入力としてShift-JISを食ったり、出力としてShift-JISを吐いたりすることはできます。

Q: ではUTF-8のファイルってどうすれば作れるんですか?

A: 良い質問です。Windowsでは、普通にテキストファイルを作るとShift-JISでファイルが保存されてしまいます。PHPファイルで<meta>タグ内にcharset=UTF-8と書いたらダメです。UTF-8に対応したエディタが必要です。例えばWindowsメモ帳でテキストファイルを作る場合、単にファイルを保存するのではなく、「ファイル」>>「名前をつけて保存」を選び、一番下の「文字コード」というプルダウンで「UTF-8」を選んで保存します。Macではテキストエディットのデフォルト設定がUTF-8になっているはずですから、何も変更しなくて大丈夫です。「テキストエディット」>>「環境設定」の「開く/保存」タブで一番下の文字コードを確認してみて下さい。

Q: サーバ系はEUCと聞いたことがあります。

A: おおむね正解です。Linuxなどのunix系サーバでは通常Shift-JISではなくEUCというコード系が使われているからです。PHPはEUCにも対応しています。ただし、本書ではWindowsまたはMacといった自宅のパソコンをテスト用サーバとして、簡単にファイルを作ることを想定してUTF-8で話を進めます。EUCコード系のファイルを簡単に作れるエディタをお持ちでしたら(フリーでダウンロードできますが) EUCでソースコードを記述しても問題ありません。もちろんその場合は<meta>タグ内でcharset=EUCと記述することをお忘れなく。

[†] 訳注: 日本語版に対応して以下のQ&Aを追加しました。

PHPを使ってフォームのデータにアクセスする

さてオーウェンは、PHPスクリプトが必要です。これでUFOによる誘拐フォームの情報をmailtoのテクニックよりも高い信頼性を勝ち取ることができます。早速作ってみましょう。まだ全部理解できていないなどどという心配には及びません。おいおいわかります。

PHPスクリプトが通常のHTMLタグや属性を含んでいるというのは全然普通ことです。

```
PHPスクリプトは、<head>
大抵HTML Web
ページとそっくり
な書き出しだす。
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>UFOに誘拐された！？「誘拐レポート」を！</title>
  </head>
  <body>
    <h2>UFOに誘拐された！？「誘拐レポート」を！</h2>
```

このスクリプトコードのブロック全体がPHPです…このスクリプト以外は普通のHTMLです。

ほら、ここから面白くなります。
これが実際のPHPコードの始まりです。

```
<?php
$when_it_happened = $_POST['whenithappened'];
$how_long = $_POST['howlong'];
$alien_description = $_POST['aliendescription'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];

echo '情報提供ありがとうございます。  


```

PHPコードのこの部分でフォームのデータを取りつけておいて、確認画面の一部として表示するためには使います。

ここでPHPを使ってフォームのデータからHTMLコードを生成します。

通常のWebページと同様、この部分のPHPスクリプトでHTMLタグを終了します。



試運転

オーウェンのフォームをPHPスクリプトを使って
フォームのデータを処理するように変更しましょう。

`report.php`という名前でテキストファイルを作り、反対側の
ページにあるすべてのコードを入力します。これがオーウェンの
Webフォームを処理するためのスクリプトです。

このPHPスクリプトは、まだフォームとつながっていません。そこで
`report.html`をテキストエディタで開き、フォームのactionをmailtoから`report.php`に
変更します。ファイルを保存するときは、文字コードをUTF-8とすることを忘れないで下さい。

```
<form action = "report.php" method = "post">
```

`report.html`ページをWebブラウザで開き[†]、何かUFOによる誘拐に関する情報を入力したら、
「誘拐レポートの送信」ボタンをクリックします。



The left screenshot shows the 'report.html' form with fields for name, email, alien sighting details, and a dog photo. The right screenshot shows the 'report.php' source code, which handles the POST data and prints it back to the browser.

ブラウザによっては、変な文字が
書いてあるWebページが見えて
しまう可能性がありますし、`report.`
phpスクリプトのPHPソースコード
がそのまま表示されてしまう
可能性もあります。

PHPスクリプトの動作としてこれで正しいと思いますか？何故か（何故ダメか）、何が起こっていると思われるか、書き留めておきましょう。

[†] 訳注：文字化けして日本語が読めない場合、文字コードをUTF-8以外で保存した可能性があります。`report.html`と`report.php`を開き、UTF-8で保存して下さい。Windowsではメモ帳で、「ファイル」>「名前をつけて保存」を選び、一番下の「文字コード」というプルダウンで「UTF-8」を選んで保存します。Macでは「テキストエディット」>「環境設定」の「開く/保存」タブで画面一番下の「文字コード」を確認してみて下さい。

PHPスクリプトはサーバ上にいなければなりません！

パソコン上で運よく Web サーバが実行できるようになっていれば話は別ですが、そうでなければ、report.php スクリプトは、「誘拐レポート」のフォームを「提出」しても、実行されません。PHP がプログラミング言語であること、そしてそれを実行するためには環境が必要であることを覚えておいて下さい。この環境というのは、PHP サポート付きの Web サーバのことです。PHP スクリプトとそのスクリプトに依存した Web ページは本物の Web サーバ上に置いておかなければなりません。ローカルのファイルシステムから直接的にスクリプトを単に開くというわけではありません。

もしローカルにインストールされた Web サーバがすでにあって、それが PHP もサポートしているのであれば、そのパソコンすぐに PHP スクリプトをテストすることができます。

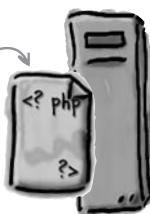
Web ブラウザは PHP については何も知らないので、PHP スクリプトを実行する能力ありません。



HTML Web ページは Web ブラウザでローカルに開くことができますが、 PHP スクリプトは常に Web サーバから URL を使って「開かなければ」なりません。

PHP スクリプトは、 Web ブラウザにとっては無意味なコードの羅列にすぎません。

Web サーバは PHP コードを理解し、スクリプトを実行することができます！



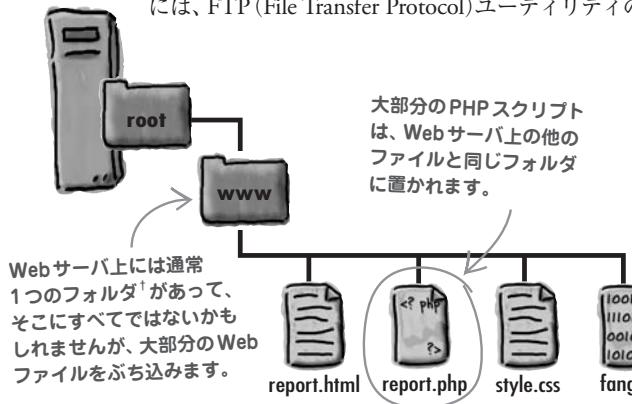
PHP をサポートしている Web サーバであれば、 PHP スクリプトを実行し、それを HTML Web ページに書き換えることができる、ブラウザはそれを読むことができます。

Web ページが Web サーバにより配信されたかどうかを簡単に見分けるには、 URL が何で始まっているかを見ることができます。「http:」で始まつていればサーバから配信されたものです。ローカルファイルとして開いたページは常に「file:」で始まります。

PHP スクリプトは Web サーバ上で 動かさなければなりません。そうでないとちゃんと動きません。

PHPスクリプトをサーバに送り込む

PHPスクリプトをパソコン上で作成／編集することについては、全く問題ありません。しかしこれを動かすには、ファイルをWebサーバ上に置く必要があります。PHPのファイルは通常Webサーバ上でHTMLファイルと同じところに置かれます。PHPスクリプトをWebサーバ上に置くにあたり、何も特別なことは必要ありません。単にWebページがアクセスできる場所[†]にそれをアップロードして置くだけです。Webサーバにファイルをアップロードするには、FTP(File Transfer Protocol)ユーティリティのようなものに助けてもらう必要があるでしょう。



PHPスクリプトをWebサーバにアップロードするだけでは、不十分です。WebサーバにはPHPもインストールされていなければならぬのです。WebサーバはデフォルトでPHPを含んでいる場合もありますが、そうでないものもあります。

素朴な疑問に答えます

Q: WebサーバにPHPがインストールされているかどうかは、どうすれば分かるのでしょうか？

A: Webの管理者かWebの提供会社に問い合わせればわかりますが、自分でちょっとしたテストをやってみるという手もあります。`test.php`というテキストファイルを作って、以下のコードを入力して下さい。

<?php
phpinfo();
?>

でき上がったら、`test.php`をWebサーバにアップロードして、WebブラウザにサーバのURL^{††}を入力して下さい。PHPがサーバにインストールされていれば、バージョン情報を含むPHPに関する詳細情報が表示されます。bingoです！

このコードは、PHPにPHP自身の情報を表示させるためのものです。



PHPがインストールされたWebサーバが無い場合、付録iiをチェックしてみて下さい。

ここにWebサーバにPHPをセットアップし動作させる方法が書いてあります。

終わったら忘れないで `phpinfo()`スクリプトを削除して下さい。忘れるとなにかのページが見えてしまします。

^{††} 訳注：URLの最後に`test.php`を付与するのを忘れないで下さい。例えば、Webサーバのアドレスが`http://www.someaddress.com/`だとすると、`http://www.someaddress.com/test.php`と入力して下さい。ご自身のパソコンにサーバをインストールした場合は、`http://localhost/test.php`または`http://127.0.0.1/test.php`と入力すればOKです。

[†] 訳注：本書付録iiに従って、サーバソフトを一括でパソコンにデフォルト設定のままインストールした場合、Windowsでは、

C:\Program Files\VertrigoServ\www

が該当フォルダです。Macでは、

/Applications/xampp/htdocs

(Finderではアプリケーション > xampp > htdocs)です。どちらの場合も同じパソコン内ですからファイルを置くのにFTPは必要ありません。単にコピーすればOKです。

イメージファイルは、サーバ上で専用のフォルダに押し込まれることもありますが、これは構成上の理由によるものです…この例はそうなっていません。





試運転

Webサーバに「誘拐レポート」関連ファイルをアップロードし、フォームをもう一度試してみましょう。

report.html、report.php、style.cssおよびfang.jpgの4つのファイルをPHPがインストールされているWebサーバ上にアップロードします。ブラウザにreport.htmlページのURLを入力し、フォームをUFOによる誘拐に関する情報で埋め、最後に「誘拐レポートの送信」ボタンをクリックします。



UFOに誘拐された！？「誘拐レポート」を！

UFOに誘拐された時の様子を教えてください。

姓：	石井
名：	秀樹
emailアドレスは？	hideki@theyrealgreen.com
いつですか？	去年の11月
何時間位ですか？	11時間
何人見ましたか？	たくさん
どんな奴らでしたか？	小さくて緑色です。
何をされましたか？	UFOのレギュレーションについて聞かれた。
飼い犬 Fang を見ましたか？	はい <input checked="" type="radio"/> いいえ <input type="radio"/>

何か他にコメントは？

UFOに誘拐された！？「誘拐レポート」を！

PHPスクリプトが動きました！
Web確認画面にフォームデータが表示されています。

情報提供ありがとうございます。
誘拐されたのは去年の11月で、時間は11時間です。
奴らは: 小さくて緑色です。
ファンガはいましたか? いいえ
メールアドレスは hideki@theyrealgreen.com



イケてる。フォームのデータをメールで送ろうと思ったら、あとちょっと PHP コードを足すだけでいいのか。

その通りです。今の `report.php` スクリプトはまだコードが足りないので、UFO による誘拐のデータをメールでオーウェンに飛ばすことができません。

でもこれは大した問題ではありません。PHP には関数があります。関数とはあらかじめビルド済みの再利用可能なコードのことです。これを使ってメールメッセージを飛ばすことができます。つまりメールメッセージに何を書き込むかを決めて、PHP を使ってメールを作成して送ればよいだけです。

ちょっと待って！まだもともとの `report.php` スクリプトがどういう風に動くかさえ知らないのよ。それなのにメールを飛ばす方に話を持って行くの？ それってものすごく大変なんじゃない…大丈夫！？

大丈夫です。PHP でもっとやればやるほど、PHP をもっと知ることができます。

ですからオーウェンの `report.php` スクリプトにメールの機能を追加するために、PHP の中により深く入り込んで行けば、結果としてスクリプトを操るための性能のいいコントローラを手にすることができます。



サーバが PHP を HTML に変えます

PHP スクリプトがどのように動作するかを理解する上で最も重要なことは、サーバ上でスクリプトを動作させる際に上手く操るためのコントローラを握っておくことです。大部分の PHP スクリプトは、PHP コードと HTML コードの両方を含んでいて、サーバはクライアントの Web ブラウザにすべてのものを HTML として渡してしまう前に PHP の部分を実行し HTML に変えます。オーウェンの report.php スクリプトの場合、PHP コードは確認画面の本体に含まれるほとんどの HTML コンテンツを生成します。その前後にある HTML コードは変更なしでそのまま配信されます。

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
  <meta http-equiv="Content-Type" content="text/html;
  charset=UTF-8" />
  <title>UFOに誘拐された！？「誘拐レポート」を！</title>
</head>
<body>
  <h2>UFOに誘拐された！？「誘拐レポート」を！</h2>

<?php
$when_it_happened = $_POST['whenithappened'];
$how_long = $_POST['howlong'];
$alien_description = $_POST['aliendescription'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];
echo '情報提供ありがとうございます。  

```

大部分の静的な HTML コード。
サーバは、これらを変更なしで
ブラウザに渡します。

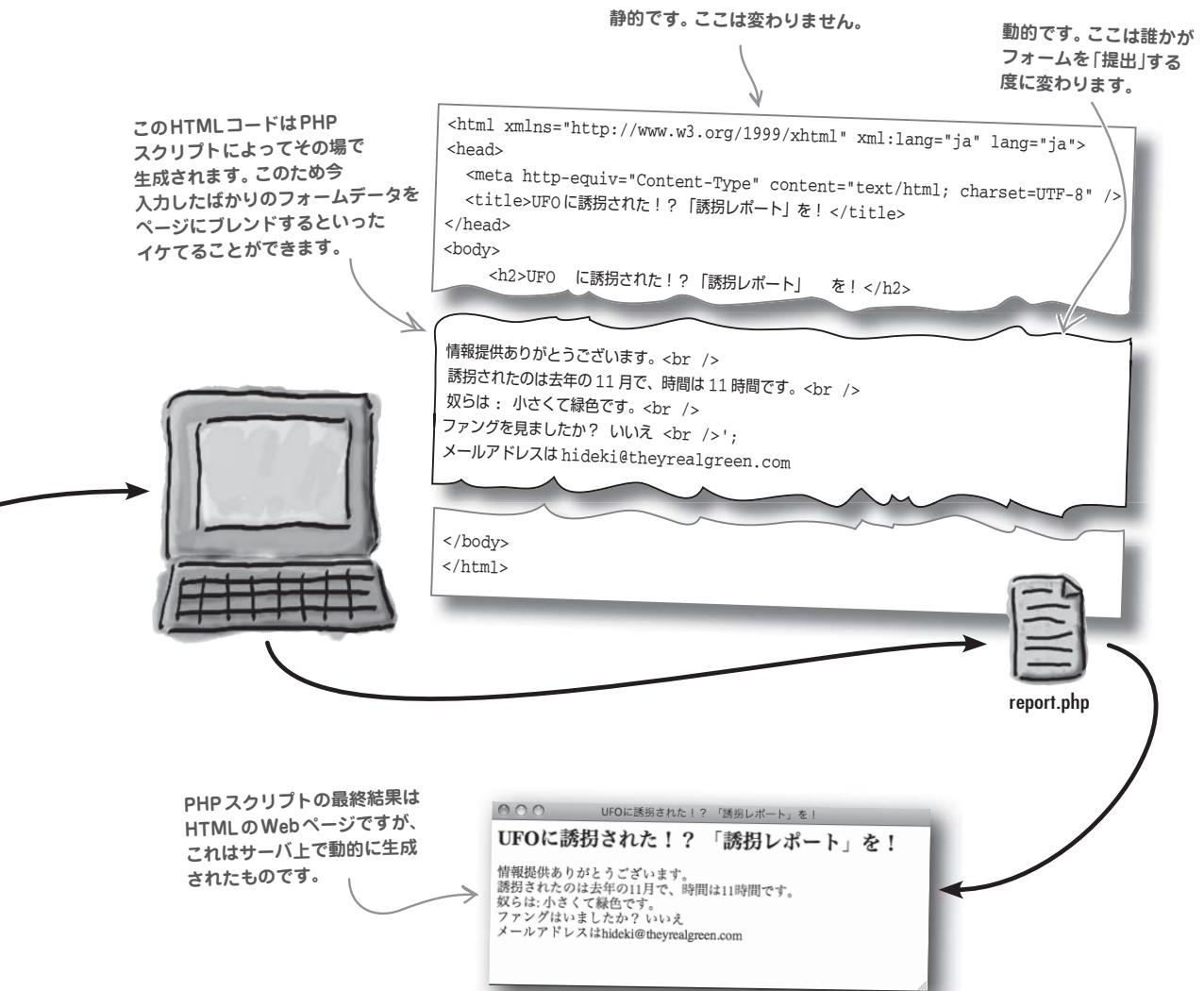
この部分の HTML コードはブラウザに
そのまま変わらずに渡されます。
charset=UTF-8 です。
くれぐれもご注意を。

この部分の PHP コードはサーバで
実行され、フォームに入力された
データを含む HTML コードを
生成します。



report.php





オーウェンの PHP スクリプトを分解する

`report.php` スクリプトは「誘拐レポート」のフォームをトリガーとしていて、(この時点での)役割は、フォームのデータを取ってきて確認画面の Web ページを生成することです。どうやっているのか見てみましょう。

最初のコード列は純粋な HTML です。今作ろうとしている Web ページのセットアップであり、すべてのページに必要ないくつかの HTML タグです。

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>UFO に誘拐された！？「誘拐レポート」を！</title>
</head>
<body>
    <h2>UFO に誘拐された！？「誘拐レポート」を！</h2>
```

ここから面白くなります。HTML コードを突き抜け、PHP コードに入る準備ができました。`<?php` タグが PHP コードセクションを開始します。このタグに続くすべてのものは、純粋な PHP です。

```
<?php
```

この部分のコードがフォームのデータを取ってきて、個々の変数に格納します。これにより以降の処理でデータに簡単にアクセスできるようになります。PHP 変数は、数値やテキスト、その他の種類のデータを格納することができます。

```
$when_it_happened = $_POST['whenithappened'];
$how_long = $_POST['howlong'];
$alien_description = $_POST['aliendescription'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];
```

さあ準備ができました！今作った変数を動的に生成された HTML コードに挿入することで動かすことができます。`echo` コマンドで Web ブラウザに直接返す HTML コードを出力します。

```
echo '情報提供ありがとうございます。<br />';
echo '誘拐されたのは' . $when_it_happened;
echo 'で、時間は' . $how_long . 'です。<br />';
echo '奴らは：' . $alien_description . '<br />';
echo 'ファングはいましたか？' . $fang_spotted . '<br />';
echo 'メールアドレスは' . $email;
```

?> タグは `<?php` タグとペアで PHP コードセクションが終了したことを表します。この後、通常の HTML コードに戻ります。

```
?>
```

最初に HTML タグで開いたページを、これで終了します。

```
</body>
</html>
```

これを忘れない。今 HTML の
Web ページを生成しているので、
HTML コードを終了させます。



report.php

その通りです。この HTML コードはきわめて小さいものです。通常は DOCTYPE とか `<meta>` タグなどがあるはずです。ここでは話を単純化しています。

ここから PHP コードを取り扱います…少なくとも終了を表す `?>` タグまでです。

PHP コードの各行でフォームフィールドのデータを新しい変数に代入しています。

この PHP コードが変数を HTML コードにブレンドし、それがブラウザの出力になります。

ここで PHP コードは終了です。
ここから通常の HTML コードに戻ります。

† 訳注：これは英語での話です。日本語を表示する場合コード系を指定する必要があるため、`<meta>` タグは含まれています。もちろん `charset=UTF-8` です。

コーディング 生きていく上の PHP 規則

オーウェンの report.php スクリプトはすべての PHP スクリプトで当てはまる PHP 言語に関するいくつかの基本的な規則を明らかにしてくれています。これら規則について見ていきましょう。

- PHP コードは必ず <?php と ?> とで囲まれている。



- すべての PHP の文は、セミコロン(;)で終わらなければならない。

コードが PHP を抜け出さない場合、セミコロンを忘れないかチェックしてみて下さい。これは想像以上によく起こっています。

セミコロンにより PHP は文の終わりを知ることができます。

- Web ページに PHP コードが入っているのであれば、Web サーバ上のファイル名は .html ではなく .php で終わらせるとよい。



これはルール違反とはなりませんが、PHP スクリプトに .php というファイル拡張子をつけるのは良いことです。

- PHP の変数名はドル記号 (\$) で始めなければならない。

\$email = \$_POST['email'];

ドル記号を見れば PHP 変数と一緒に識別することができます。ここに PHP スクリプトの情報を格納することができます。

report.php スクリプトで使われている変数を見て、PHP の変数に付隨している規則が他に何か思い浮かびますか？書き留めておきましょう！

完璧な変数名を見つける

\$で始めるという規則に加えて、PHPの変数名は大文字／小文字を区別します。ただしそれだけではありません。どのように変数名をつけるかを司る規則が他にもあります。規則には、まず構文上のものがあります。つまりこれらの規則を無視した場合、コードは破綻します。一方PHPの達人プログラマの間で古くから単によい慣習とされているという規則もあります。

正式な規則から始めましょう。これを無視して変数に名前をつけると、間違いなくトラブルを引き起こすというものです。正しい変数名をつけるには次の規則に従わなければなりません。

- 最初の文字はドル記号(\$)で始まらなければなりません。
もうやりました！
- 変数名は少なくとも1文字の長さがなければなりません。
\$記号は数えません。これはすべての変数名に要請されます。
- ドル記号直後の最初の文字は、英字または下線(_)が許され、その次以降の文字は英字、下線、または数字が許されます。
- 空白と_および\$以外の特殊文字は変数名のどこにも現れてはいけません。



以上の規則に従わないと、コードは動作しません。さらにコーディングの便宜というだけで片づけるにはもったいない、従うに値する規則がいくつかあります。これらの規則はPHPコードに一貫性を持たせ、読みやすくしてくれます。

- 変数名にはすべて小文字を使う。
- いくつかの単語をつなげて変数名をつける場合、単語間を下線で分ける。

以上の2つの規則に従わなくても、コードが破綻するということはありません。これらの規則にとらわれずにPHPコードを動かしても、正しく動作します。これは単にスタイル上の便宜だからです。しかし自分自身の変数を作る際にこれらの規則に従った方が良い結果が得られることでしょう。

変数とはデータを格納するための入れ物です。
すべての変数は一意の名前を持ちます。

NGです！ PHP変数名はマイナス記号や空白を含んではいけません。

PHP変数名はドル記号\$で始まらなければならず、空白を含むことはできません。

素朴な疑問に答えます

Q: PHPのコマンドに大文字・小文字を混ぜると問題なのでしょうか?

A: YesでもNoでもあります。PHPは大部分、大文字・小文字を区別しません。従って大部分のコマンドで文字種を混ぜても問題ありません。つまりコンテンツをechoするのにechoでもECHOでもEchoでも使うことができます。しかし便宜上の問題で、スクリプトでは文字種は一貫性を持たせて使った方がよいのです。大部分のPHPプログラマは、圧倒的大多数のPHPコードにおいて小文字を好んで使います。これが本書のサンプルコードで一貫してechoと書かれていた理由です。

Q: ではコーディングの便宜上悪いことだとしても、PHPコードに文字種を混ぜても同じものとみなされるのですね?

A: 違います、全くダメです。PHPにおける大文字・小文字の区別に関する最大の例外は変数名です。変数というのは自分で作り出したデータ格納領域です。例として「誘拐レポート」のスクリプトで使われている変数\$emailを取り上げましょう。この変数名は文字種を区別しますから、この変数を\$EMAILや\$emailのように参照することはできません。すべての変数名はこのように大文字・小文字を区別します。従ってコード上で変数に名前を付けるときは注意が必要であり、それを参照するときは一貫性を持たせることが重要です。注意一秒変数名一生。

Q: PHPとHTMLのコードを同じファイルに置いても本当に大丈夫なんですか?

A: 絶対大丈夫です。実際、多くの場合、絶対そのようにする必要があるくらいです。

Q: そのようにする必要があるってどういうことですか?

A: Webサーバに共通するあらゆるアイデアはHTMLのWebページを

ブラウザに届けるためのものです。PHPはこの事実を変えることはありません。PHPは、HTMLコンテンツをその場で変えることができるようになります。例えばその日の日付をデータベースから取ってくることも、ショッピングカートの中の注文の合計額を計算することもできます。つまりPHPを使えばWebページの中に入つてHTMLを操作することができます。これに対してHTMLは設計時に静的にページを生成するだけです。HTMLコードのページにPHPコードがちりばめられていて、重要なデータを差し込むとかHTMLをプログラム的に変えてしまうといったことは極めてよくあることです。

Q: HTMLファイルに組み込まれているPHPコードは、独立した行になければならないのでしょうか?それともHTMLタグ属性の一部として、といった具合にHTMLとしては行の中に組み込んで良いのでしょうか?

A: PHPコードの置き場所は、<?phpと?>というタグに囲まれていなければならぬという要請を除けば、何ら制約はありません。HTMLコードにどのように組み込んで構いません。実際、HTMLタグの属性を設定するときなどに、HTMLコードの中間部にPHPコードをちょっと押し込むという必要に迫られたりするものです。これは完璧に正当化されたPHPの使い方です。

Q: <?phpではなく<?で始まるPHPコードを見たことがあるのですが、これは正しいのでしょうか?

A: 完全には正しくありません。技術的に言うと正当ではありますが、お勧めできません。サーバの設定で短い開始タグ(<?)を有効にしなければならないことになっています。通常の<?phpタグは常に動きますので、こちらを使ったほうがよいでしょう。

Q: Webサーバは常に純粋なHTMLコードをクライアントブ

ラウザに返すのに、URLにはどうして webpage.phpのようなPHPスクリプト名が見えるのでしょうか?

A: 忘れでならないのは、すべてのWebページは、クライアントブラウザからのリクエストに対しWebサーバがレスポンスを返すという双方向通信の結果として表示されているということです。URLはリクエストのベースであるのに対し、サーバから返されるコンテンツはレスポンスです。PHPスクリプトは、通常のHTMLのWebページと同様にブラウザに入力されたURLや他のページからリンクされたURLやフォームのアクションとしてリクエストされます。これで、PHPの「ページ」のURLになせPHPスクリプトの名前が見えるかの説明になっています。

回答の残りの半分は、サーバからのレスポンスです。これはPHPスクリプトによって生成された結果のコードです。大部分のPHPスクリプトはHTMLコードを生成するため、生成されたコードがHTMLであってPHPではないというのは常識のことです。つまりURLがサーバ上の.phpファイルを参照し、これによってサーバ上のPHPが実行され、最終的に純粋なHTMLコンテンツとなってブラウザに返されます。当然のことなのです。

Q: PHPの変数にはどんな種類のデータでも格納することができるのですか?

A: まさしくその通りです。真偽値(true/false)データを格納することができます。また数値データとして整数でも浮動小数点(数)でも格納することができます。配列つまりデータやオブジェクトの集まりでも大丈夫です。ここでオブジェクトというのは、データとそれを操作するコードをまとめたものです。配列については、本章後半で少し触れます。オブジェクトについては11章で挑戦します。NULLという特別なデータ型もあります。これは値がないことを表します。例えば値を代入していない変数の値はNULLと考えます。

\$_POSTはフォームのデータを保持する特別な変数です

\$_POSTはスーパーグローバルとも呼ばれる特別な変数の1つです。この変数はPHPに最初から組み込まれていて、スクリプト全体で利用可能です。\$_POSTはスクリプトを実行するときにはすでに存在しています。他のPHPの変数のように自分で生成する必要はありません。

UFOに誘拐された！？ 「誘拐レポート」を！

UFOに誘拐された時の様子を教えてください。

姓： 石井
名： 秀樹
emailアドレスは？ hideki@theyrealgreen.com
いつですか？ 去年の11月
何時間位ですか？ 11時間
何人見ましたか？ たくさん
どんな奴らでしたか？ 小さくて緑色です。
何をされましたか？ UFOのレギュレーションについて聞かれた。
飼い犬ファンダグを見ましたか？ はい ○ いいえ ◎



何か他にコメントは？
誘拐レポートの送信

どうか私に済き1票を！

\$_POSTスーパーグローバル変数は、
フォームに入力された各項目
データを保持します。



\$_POST['howlong']

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <title>UFOに誘拐された！？ 「誘拐レポート」を！</title>
</head>
<body>
  <h2>UFOに誘拐された！？ 「誘拐レポート」を！</h2>
  <?php
    $when_it_happened = $_POST['whenithappened'];
    $how_long = $_POST['howlong'];
    $alien_description = $_POST['aliendescription'];
    $fang_spotted = $_POST['fangspotted'];
    $email = $_POST['email'];
    echo '情報提供ありがとうございます。<br />';
    echo '誘拐されたのは' . $when_it_happened;
    echo 'で、時間は' . $how_long . 'です。<br />';
    echo '奴らは' . $alien_description . '<br />';
    echo 'ファンダグはいましたか？' . $fang_spotted . '<br />';
    echo 'メールアドレスは' . $email;
  ?>
</body>
</html>
```



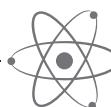
\$_POSTスーパーグローバル変数は、
HTML フォームで使われるフォームの
「提出」メソッドと直接的に結びついています。
メソッドが post に設定されていると、すべてのフォームの
データは、\$_POSTスーパーグローバル変数にパッケージ
されます。ここで各項目データは、必要に応じて取り出され
て使われます。

```
...<form method="post" action="report.php">
...
...</form>
```



フォームの「提出」メソッドは、
フォームのデータがPHPスクリプトにどのように供給される
かを決定します。

「howlong」という名前は、この
フォームフィールドの<input>
タグの名前属性から来ています。



脳力発揮

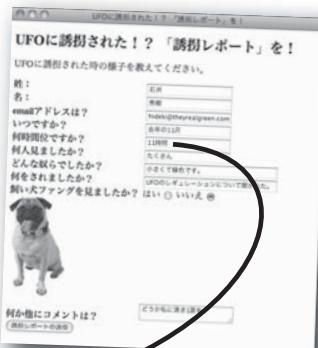
\$_POSTスーパーグローバル変数がどのように
動作するかわかりますか？ Owenのフォームに
あるすべてのテキストボックスから、いくつもの
値をどうやって格納しているのでしょうか？

\$_POSTはフォームのデータをスクリプトに運んでくれます

\$_POSTは配列という特別な種類のPHPの記憶領域で、ひとつの名前の下に変数の集まりを格納することができます。誰かがオーウェンのフォームを「提出」するとその人がフォームフィールドに入力したデータは、配列\$_POSTに格納されます。この配列の役割は、入力されたデータをスクリプトに渡すことです。

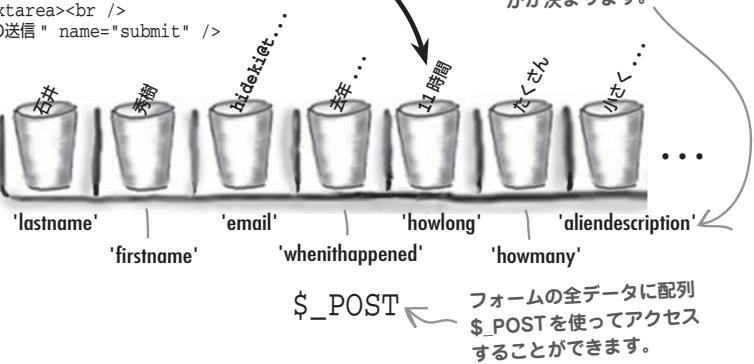
配列\$_POSTの各要素は、フォームフィールドに入力された各データに対応します。フォームフィールドの特定のデータにアクセスするには、\$_POSTにフィールドの名前を使います。つまり誘拐の期間は\$_POST['howlong']に格納されています。オーウェンのフォームのHTMLコードをよく見ると、フォームの名前が\$_POSTに格納されているデータとどのように関連付けられているのかがわかります[†]。

```
<p>UFOに誘拐された時の様子を教えて下さい。</p>
<form method="post" action="report.php">
  <label for="lastname">姓:</label>
  <input type="text" id="lastname" name="lastname" /><br />
  <label for="firstname">名:</label>
  <input type="text" id="firstname" name="firstname" /><br />
  <label for="email">emailアドレスは? </label>
  <input type="text" id="email" name="email" /><br />
  <label for="whenithappened">いつですか? </label>
  <input type="text" id="whenithappened" name="whenithappened" /><br />
  <label for="howlong">何時間位ですか? </label>
  <input type="text" id="howlong" name="howlong" /><br />
  <label for="howmany">何人見ましたか? </label>
  <input type="text" id="howmany" name="howmany" /><br />
  <label for="aliendescription">どんな奴らでしたか? </label>
  <input type="text" id="aliendescription" name="aliendescription" size="32" /><br />
  <label for="whattheydid">何をされましたか? </label>
  <input type="text" id="whattheydid" name="whattheydid" size="32" /><br />
  <label for="fangspotted">飼い犬ファングを見ましたか? </label>
  はい <input id="fangspotted" name="fangspotted" type="radio" value="はい" />
  いいえ <input id="fangspotted" name="fangspotted" type="radio" value="いいえ" /><br />
  <br />
  <label for="other">何か他にコメントは? </label>
  <textarea id="other" name="other"></textarea><br />
  <input type="submit" value="誘拐レポートの送信" name="submit" />
```



配列\$_POSTは、
ユーザがフォームに
入力した値が
入っています。

フォームフィールドの
名前で配列\$_POSTの
どの要素にアクセスする
かが決まります。



[†] 訳注：フィールドの名前を間違えてしまうと、\$_POSTでフォームデータに正しくアクセスできなくなってしまいます。report.phpで\$_POSTを使っている部分、例えば\$_POST['aliendescription']を\$_POST['description']と書き換えて何が起こるか観察してみて下さい。



オーウェンの「誘拐レポート」フォームに入力されたデータのうち、今使っているものはすべてではありません。この情報は命に関わる情報を含んでいるということを忘れてはいけません。迷子になった犬ファンクをオーウェンのところに導いてくれる可能性があります。ですから、誘拐に関するデータはすべて取っておいて、PHPの変数に転記しておく必要があります。

report.phpスクリプトは、現状
5つの異なる各フォームデータ
を無視しています。残念！

```
...
<form method="post" action="report.php">
<label for="firstname">First name:</label>
<input type="text" id="firstname" name="firstname" /><br />
<label for="lastname">Last name:</label>
<input type="text" id="lastname" name="lastname" /><br />
<label for="email">What is your email address?</label>
<input type="text" id="email" name="email" /><br />
<label for="whenithappened">When did it happen?</label>
<input type="text" id="whenithappened" name="whenithappened" /><br />
<label for="howlong">How long were you gone?</label>
<input type="text" id="howlong" name="howlong" /><br />
<label for="howmany">How many did you see?</label>
<input type="text" id="howmany" name="howmany" /><br />
<label for="aliendescription">Describe them:</label>
<input type="text" id="aliendescription" name="aliendescription" size="32" /><br />
<label for="whattheydid">What did they do to you?</label>
<input type="text" id="whattheydid" name="whattheydid" size="32" /><br />
<label for="fangspotted">Have you seen my dog Fang?</label>
Yes <input id="fangspotted" name="fangspotted" type="radio" value="yes" />
No <input id="fangspotted" name="fangspotted" type="radio" value="no" /><br />
<br />
<label for="other">Anything else you want to add?</label>
<textarea id="other" name="other"></textarea><br />
<input type="submit" value="Report Abduction" name="submit" />
</form>
</body>
</html>
```

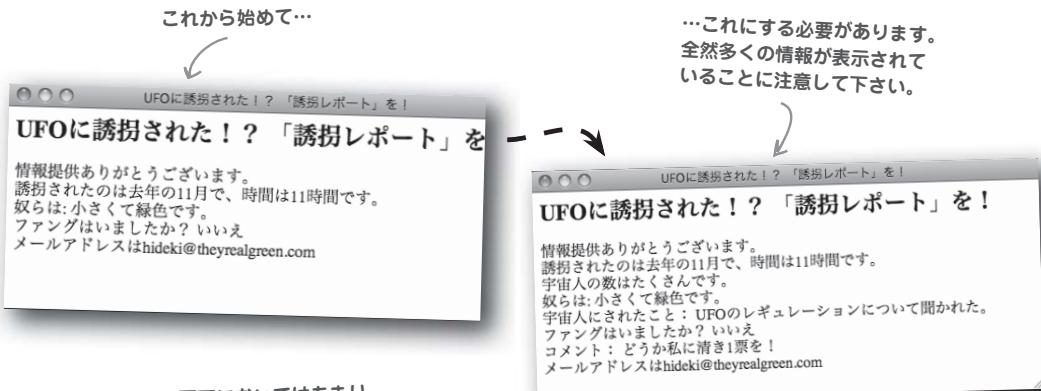


report.html

各フィールドの
<input>タグは PHP か
らフォームのデータに
アクセスするための
カギを握っています。

残りのフォームデータを格納するために4つの新しい変数を作つて PHP コードを書きます。変数名は \$name、\$how_many、\$what_they_did、\$other にします。
ヒント：変数 \$name を作ったら、ユーザの氏名を格納します。

まだ全然終わっていません。PHPスクリプトにより生成された確認画面のWebページは、これら新しい変数を使ってUFOによる誘拐に関するより多くの情報を表示する必要があります。



ユーザ名は確認画面においてはあまり重要ではありません。もちろん後で誘拐に関するメールをオーウェンに飛ばす際には必要となります。

\$name を除いて今作ったすべての変数を使い、より多くの情報を含む確認画面のページを生成できるように、以下のコードの足りない部分を仕上げます。

```
echo '情報提供ありがとうございます。<br />';
echo '誘拐されたのは' . $when_it_happened;
echo 'で、時間は' . $how_long . 'です。<br />';

.....
echo '奴らは:' . $alien_description . '<br />';

.....
echo 'ファングはいましたか？' . $fang_spotted . '<br />';

.....
echo 'メールアドレスは' . $email;
```

自分で考えてみよう の答え

オーウェンの「誘拐レポート」フォームに入力されたデータのうち、今使っているものはすべてではありません。この情報は命に関わる情報を含んでいるということを忘れてはいけません。迷子になった犬ファングをオーウェンのところに導いてくれる可能性があるのです。ですから、誘拐に関するデータはすべて取っておいて、PHPの変数に転記しておく必要があります。

report.phpスクリプトは、現状
5つの異なる各フォームデータ
を無視しています。残念！

```
...
<form method="post" action="report.php">
<label for="firstname">First name:</label>
<input type="text" id="firstname" name="firstname" /><br />
<label for="lastname">Last name:</label>
<input type="text" id="lastname" name="lastname" /><br />
<label for="email">What is your email address?</label>
<input type="text" id="email" name="email" /><br />
<label for="whenithappened">When did it happen?</label>
<input type="text" id="whenithappened" name="whenithappened" /><br />
<label for="howlong">How long were you gone?</label>
<input type="text" id="howlong" name="howlong" /><br />
<label for="howmany">How many did you see?</label>
<input type="text" id="howmany" name="howmany" /><br />
<label for="aliendescription">Describe them:</label>
<input type="text" id="aliendescription" name="aliendescription" size="32" /><br />
<label for="whattheydid">What did they do to you?</label>
<input type="text" id="whattheydid" name="whattheydid" size="32" /><br />
<label for="fangspotted">Have you seen my dog Fang?</label>
Yes <input id="fangspotted" name="fangspotted" type="radio" value="yes" />
No <input id="fangspotted" name="fangspotted" type="radio" value="no" /><br />
<br />
<label for="other">Anything else you want to add?</label>
<textarea id="other" name="other"></textarea><br />
<input type="submit" value="Report Abduction" name="submit" />
</form>
</body>
</html>
```



report.html

ドット記号は複数の文字列テキストを1つにくっつける働きがあります。この処理は文字列の連結と言います。

各フィールドの
<input>タグは PHP から
フォームのデータに
アクセスするための
カギを握っています。

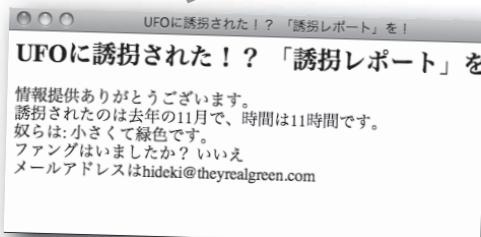
この空白で「姓」と
「名」を区切ります。

残りのフォームデータを格納するために4つの新しい変数を作つて PHP コードを書きます。変数名は \$name、\$how_many、\$what_they_did、\$other にします。
ヒント：変数 \$name を作ったら、ユーザの氏名を格納します。

```
$name = $_POST['firstname'] . ' ' . $_POST['lastname'];
$how_many = $_POST['howmany'];
$what_they_did = $_POST['whattheydid'];
$other = $_POST['other'];
```

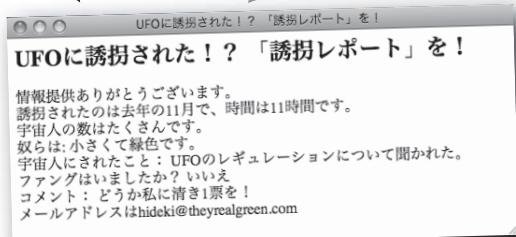
まだ全然終わっていません。PHPスクリプトにより生成された確認画面のWebページは、これら新しい変数を使ってUFOによる誘拐に関するより多くの情報を表示する必要があります。

これから始めて…



ユーザ名は確認画面においてはあまり重要ではありません。もちろん後で誘拐に関するメールをオーウェンに飛ばす際には必要となります。

…これにする必要があります。
全然多くの情報が表示されて
いることに注意して下さい。



\$nameを除いて今作ったすべての変数を使い、より多くの情報を含む確認画面のページを生成できるように、以下のコードの足りない部分を仕上げます。

echoコマンドは、ブラウザにHTMLコンテンツとして付加情報を出力するために使われます。

繰り返しになりますが、ドット記号は文字列と変数とを連結するために用いられます。

タグは情報をフォーマットして表示するのを助けます。今PHPを使ってHTMLを生成しているということを忘れないで下さい。

```
echo '情報提供ありがとうございます。<br />';
echo '誘拐されたのは' . $when_it_happened;
echo 'で、時間は' . $how_long . 'です。<br />';
echo '宇宙人の数は' . $how_many . 'です。<br />';
echo '奴らは:' . $alien_description . '<br />';
echo '宇宙人にされたこと:' . $what_they_did . '<br />';
echo 'ファングはいましたか？' . $fang_spotted . '<br />';
echo 'コメント:' . $other . '<br />';
echo 'メールアドレスは' . $email;
```



試運転

オーウェンのスクリプトを微調整し変化を見る。

新しい変数に関するコードを `report.php` に追加します。同様にブラウザにフォーマットされた HTML として変数を `echo` するコードも追加します。次にスクリプトを Web サーバにアップロードし、ブラウザで `report.html` を開いたら、UFO による誘拐に関する記述を埋めます。最後の「誘拐レポートの送信」ボタンを押してフォームを「提出」し結果を見てみましょう。

素朴な疑問に答えます

Q: ドット記号を使って複数の文字列を結合した時、実際に何が起こっているのでしょうか？

A: 連結は 2つ以上の文字列をくっつけて完全に新しい1つの文字列を作り出すという働きがあります。最終結果の文字列は、元の文字列がどれだけ多いかとは関係なく、常に単一の文字列です。つまり `echo` コマンドの一部として文字列を連結した場合、PHP はまず文字列を 1つにつなげ、次にその文字列をブラウザに `echo` します。

Q: 変数と文字列を連結する場合、変数の中身は文字列でなければならないのでしょうか？

A: そんなことはありません。連結は結果として常に文字列を生成しますが、連結する前の変数の中身は文字列である必要はありません。ですから例えば変数の中身が数値だったとしても、PHP はまず数値を文字列に変換してから、それを連結してくれます。

Q: ブラウザ上で PHP コードには何が起こるのでしょうか？

A: 何も起こりません。そしてこれこそが、PHP コードはブラウザでは決して見ることができない理由なのです。PHP コードはサーバ上で動作し、HTML コードに姿を変えてブラウザに送られます。つまりブラウザは、PHP の存在を全く知らないのです。Web ページとしては純粋な HTML と CSS として到着します。

Q: わかりました。ではサーバは PHP コードを HTML と CSS のコードに、どうやってちゃんと変えることができるのでしょうか？

A: その前に、PHP コードはデフォルトでは HTML コードとみなされているということを忘れないで下さい。PHP コードはスクリプト中で `<?php` と `?>` タグの間に置かれているのでそれと分かれます。サーバはこのタグを見つけると、その中身を PHP として実行するということを知っています。またこのタグの外側にあるコードに関しては、HTML としてブラウザに渡されます。

Q: なるほど。しかしそれでは PHP コードがどのように HTML/CSS コードに変わっていくのかの説明になっていません。どうでしょう？

A: あ、それは `echo` コマンドの活躍する場面です。`echo` は `<?php` と `?>` タグによってつけられた境界を突き抜けて情報を出力するコマンドと考えて下さい。つまり `echo` コマンドこそが動的に HTML/CSS コードを生成する PHP の能力のカギを握っているのです。テキストの文字列を PHP 変数と連結させ、HTML コードをその場で構成することができますが、さらに `echo` を使ってそれをブラウザに最終的な Web ページの一部として出力することができます。これに関する良い例が、オーウェンの `report.php` スクリプトで、`
` タグをテキストの最後にくっつけて HTML の改行を生成する箇所です。



相変わらずPHPスクリプトでオーウェンにフォームのデータを メールする必要があります。

現状では、report.phpスクリプトは誘拐に関する報告のフォームからデータを取ってきて、ユーザ向け確認画面のHTMLページを生成しているだけです。しかしこれでは、フォームが「提出」された際にオーウェンにメッセージをメールで飛ばすという元々の問題を解決していません。オーウェンは単にこんな感じの単純なテキストメールを受け取りたいのです。

確認画面のWebページと同様に、
このメールメッセージは静的な
テキストとフォームのデータとの
組み合わせでできています。

石井 秀樹さんが誘拐されたのは去年の11月で、時間は11時間です。

宇宙人の数はたくさんです。

奴らは: 小さくて緑色です。

宇宙人にされたこと: UFOのレギュレーションについて聞かれた

ファングを見ましたか? いいえ

コメント: どうか私に清き1票を!

このメールメッセージはPHPコードにより「コメント:」のような静的なテキストと変数に格納されたフォームフィールドのデータとを結合した文字列を出力することで生成することができます。

どうすれば静的テキストとPHP変数とからメールメッセージを出力できるか書いてみましょう。

PHP でメッセージ本文を作成する

PHP コードの中でドット記号を使って複数の文字列テキストを单一の文字列に連結する方法はすでに見てきました。ここで再び連結を使って、静的なテキストの中に散在する変数を使ってメールメッセージを構築します。

変数と静的なテキストはドット記号を使って单一のメールメッセージの文字列に連結されます。

```
$msg = $name . 'さんが誘拐されたのは' . $when_it_happened . 'で、時間は' . $how_long . 'です。' .  
' 宇宙人の数は' . $how_many . 'です。' . '奴らは：' . $alien_description . ' 宇宙人にされたこと：' .  
$what_they_did . ' ファングはいましたか？' . $fang_spotted . ' コメント：' . $other ;
```

多くのテキストエディタでは、自分で改行(return)を打たなくとも自動的に次の行へコードを折り返してくれます。

各変数は、「誘拐レポート」のフォームから引っ張ってきた文字列テキストを保持しているということを忘れないで下さい。

このような大きな文字列を構築する上で問題となるのは、PHP コードに莫大な行が必要となり、結果としてコードが読みにくく理解しにくくなることです。複数行にまたがる PHP コードを改行してフォローしやすくすれば良いのです。空白文字が影響を及ぼさない場所でコードを分割するよう気を付けて下さい。例えば連結される 2つの文字列の間は OK ですが、文字列の中はダメです。最後に PHP の文を終了させるために、コード行の終わりにセミコロンが必要です。

これは非常に大きな 1 行を複数行に分割したものです。

```
$msg = $name . 'さんが誘拐されたのは' . $when_it_happened . 'で、時間は' . $how_long . 'です。' .  
' 宇宙人の数は' . $how_many . 'です。' .  
' 奴らは：' . $alien_description . '  
' 宇宙人にされたこと：' . $what_they_did . '  
' ファングはいましたか？' . $fang_spotted . '  
' コメント：' . $other ;
```

コードの各行は、文字列の中で改行しないよう注意深く改行されています。

セミコロンで文全体を終了させる
必要はもちろんあります。

PHP コードの 1 行を意図的に複数行にまたがって記述する場合、習慣的に 2 行目以降はインデントして、どの行がどのコードに属しているのかがわかりやすくするということが行われています。

PHP コードの長い行は、改行を入れる場所にちゃんと注意を払っている限りにおいては、複数行に展開することができます。



あのPHPコードは確かに
素敵だわ。でもフォーマットされて
いないから、メールメッセージがごちゃ
混ぜになっちゃうんじゃないかな?

その通りです。PHPコードが立派にできているからといって、出力が自動的にきれいになったりはしません。

PHPコードをわかりやすく作り上げることと、ユーザが見るためのPHPコードからの出力をきちんとフォーマットすることとは全然関係ありません。PHPコードの出力をフォーマットするには、通常HTMLタグを使います。なぜならばほとんどの場合、PHPは動的にWebページを生成するために使われるからです。でもこの場合は違います。ここではメールメッセージを生成するのです。単なるテキストであって、HTMLではありません。現状を直視しましょう。メッセージはこんな感じです。

石井 秀樹さんが誘拐されたのは去年の11月で、時間は11時間です。宇宙人の数はたくさんです。奴らは：小さくて緑色です。宇宙人にされたこと：UFOのレギュレーションについて聞かれたファン グを見ましたか？いいえコメント：どうか私に清き1票を！

うわ！これはオーウェンが思い描いて
いるような誘拐に関する報告のメール
メッセージではありません。

素朴な疑問に答えます

Q : PHPスクリプトからメールを飛ばす際にHTMLによる
フォーマットを使う手段はありますか？

A : あります。しかしHTMLメールを使うためには、さらなるステップを踏んで、メッセージのcontent typeヘッダを設定しなければなりません。ヘッダやcontent typeの話は今の議論を少々越えてしまいます。そんなわけで、オーウェンのメールでのレスポンスに関しては純粋なテキストメールベッタリで話を進めます。ヘッダについては6章で詳しく学習します。ですからそこでHTMLメールについての知識を確実に得ることができます。

脳力発揮

どうすれば単なるテキストのメールメッセージを再フォーマットして読みやすくすることができるでしょう？

単なるテキストもフォーマットできます…ちょっとだけ

オーウェンはメールメッセージをHTMLによるフォーマットのない単なるテキストで送ろうとしているため、コンテンツがごっちゃになってしまわないよう、改行として
タグをくっつけることができません。でも改行文字を入れることならできます。エスケープして¥nと書きます。ですから¥nがメールテキストに現れたらいつでも、そこには改行が挿入され、その後のコンテンツは次の行の先頭から始まります。これが改行を付与した新しいメールメッセージのコードです。

メールメッセージ全体を通して改行
文字の箇所に¥nを使います。

```
$msg = $name . 'さんが誘拐されたのは' . $when_it_happened . 'で、時間は' . $how_long . 'です。¥n'.
'宇宙人の数は' . $how_many . 'です。¥n'.
'奴らは:' . $alien_description . '¥n'.
'宇宙人にされたこと:' . $what_they_did . '¥n'.
'ファングはいましたか?' . $fang_spotted . '¥n'.
'コメント:' . $other;
```

石井 秀樹さんが誘拐されたのは去年の11月で、時間は11時間です。
¥n 宇宙人の数はたくさんです。¥n 奴らは: 小さくて緑色です。¥n
宇宙人にされたこと: UFOのレギュレーションについて聞かれた¥n
ファングを見ましたか? いいえ¥n コメント: どうか私に清き1票を!

PHPの
エスケープ文字は
円記号(¥)または
バックスラッシュ(\)で
始まります†。

改行を入れる
というのはすごそうだった
けど…コードがちゃんと動かな
いんじゃ全然ダメだね。



¥nが改行ではなく
そのままテキストと
して現れています…
ダメです。

素朴な疑問に答えます

Q: エスケープ文字というのは正確には何なのでしょうか?

A: エスケープ文字とは直接的に入力するのが難しかったり、あるいはPHPコードが混乱してしまう原因となるような文字のことです。HTMLのエスケープ文字はよくご存知でしょう。表記は少し異なりますが、©とか©(著作権を表す記号©)のようにコード化されているものです。PHPのエスケープ文字セットは、非常に小さなものです。PHP言語自体が混乱を来さないように、单一引用符(¥')、二重引用符(¥")、そしてもちろん改行(¥n)といったものが用意されています。

† 訳注：エスケープ文字は、日本では円記号¥またはバックスラッシュ\のいずれかで表されます。コード系の違いにより表記が異なりますが、どちらも同じものを表します。

改行には二重引用符の文字列が必要です

オーウェンのコードに潜む問題点は、PHPが文字列を单一引用符で囲むか二重引用符で囲むかによって取り扱いが変わることによるのです。もっと詳しく言うと、改行文字(¥n)は、二重引用符で囲まれている文字列の中ではエスケープされないので。ですから誘拐レポートのメールメッセージは、二重引用符の文字列を使って作り直さないと改行がちゃんと動かないのです。

実は单一引用符対二重引用符についてはこれ以外にもお話ししなければならないことがあります。单一引用符の文字列が単なる裸のテキストであるのに対し、二重引用符の文字列についてはPHPが変数を処理するという働きがあります。二重引用符の文字列の中に変数があると、PHPは文字列を連結したときと同じように、文字列の中を変数の値で置き換えます。つまり二重引用符の文字列はメールメッセージの改行をちゃんと動かすのに必要というだけでなく、文字列の中に直接変数をぶち込むことでコードをシンプルにしてくれるという恩恵に浴することもできるのです[†]。

\$msg = "\$name さんが誘拐されたのは \$when_it_happened で、時間は \$how_long です。¥n" .

" 宇宙人の数は \$how_many です。¥n" .

" 奴らは： \$alien_description¥n" .

" 宇宙人にされたこと： \$what_they_did¥n" .

" ファングを見ましたか？ \$fang_spotted¥n" .

" コメント： \$other" ;

二重引用符の文字列のおかげで改行
文字が今度は正しく解釈されます。

でも、複数行にまたがったコードを読みやすく
するためにには、やっぱりメッセージを分割し、
文字列を連結したほうが良いでしょう。

ここがメールメッセージの
終わりの行なので、最後の最後
には改行は必要ありません。

素朴な疑問に答えます

Q: 二重引用符の文字列がそんなにイケているんだったら、なぜ今までずっと单一引用符の文字列を使ってきたのですか？

A: ええと、单一引用符の文字列はPHPでは全く処理されないということを心に留めておいて下さい。これは変数を埋め込んでない純粋なテキストを表示するには理想的です。ですから本書では引き続き单一引用符の文字列を使い続けます。もちろん二重引用符の文字列を代わりに使わなければならない理由がある場合は別です。文字列に関して單一対二重のどちらの引用符を使うかにおいて最も重要なことは、できる限り一貫性をもたせることです。

Q: 単一引用符の文字列において单一引用符(アポストロフィー)自体を使う必要があるときは、どうすればよいのでしょうか？例えば 'He's lost!' のような場合です。

A: これこそエスケープ文字の出番です。单一引用符を单一引用符の文字列で使用する場合は、正にエスケープして¥'と書きます。つまり 'He¥'s lost!' といった具合です。同じことが二重引用符の文字列の中で二重引用符を使う場合にもあてはまります。¥" †として下さい。ただし引用符は必ずエスケープしなければならないということはありません。二重引用符の中の单一引用符 "He's lost!" というように引用符の衝突が起らなければエスケープしなくても大丈夫です。

Q: つまり单一引用符の文字列は¥'をサポートし¥nはサポートしないということですね。单一引用符の中で使えるエスケープ文字を知るにはどうすればよいのでしょうか？

A: 单一引用符の文字列では¥'と¥¥のみがエスケープ文字として認識されます。これ以外のすべてのエスケープ文字は二重引用符の文字列でのみ使えます。

[†] 訳注：PHPはShift-JISに対応していないために、ここで問題が発生します。例えば「能」、「機」、「噂」、「十」、「表」といった文字を二重引用符の最後の文字列として記述するとPHPはエラーを表示します。つまり \$ten = "十"; といったスクリプトがエラーとなります。実はこれらの文字は2バイト目がエスケープ記号¥'と同じ値を持っています。このため文字列終了の"が¥'でエスケープされて文字列が終了していないことになり、PHPは発狂します。付録IIIで、もう少し詳しく説明します。

オーウェンに飛ばすメールメッセージを組み立てる

メールメッセージ本文はすでに文字列として生成したので、オーウェンのメールの残りの部分を組み立てる時が来ました。メールメッセージは単にメッセージ本文だけではありません。いくつかの異なるパートがあります。いくつかは省略可能ですが、次に述べる情報はほとんどすべてのメールで用いられています。

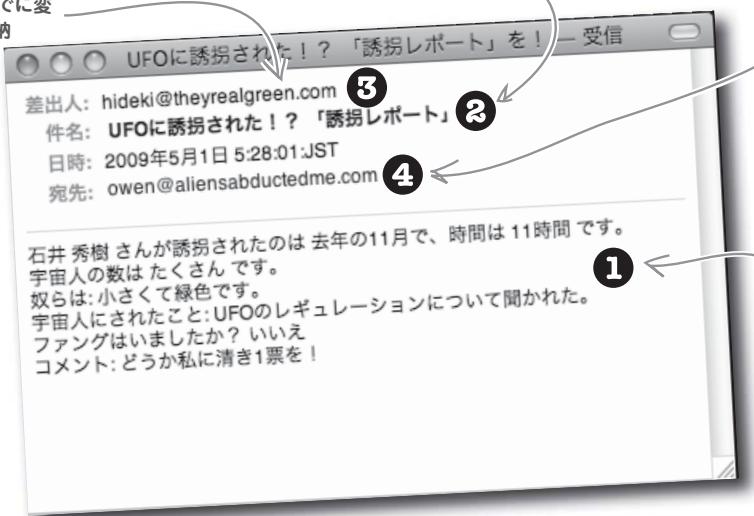
- ① メッセージ本文 もう作りました！
- ② メッセージのタイトル ここには好きなことを書けます。
オーウェンの受信ボックスで
メールの件名として表示されます。
- ③ 送り主のメールアドレス(メッセージのFROM) ユーザのメール
アドレスです。
- ④ 受け取り手のメールアドレス(メッセージのTO) オーウェンのメール
アドレスです。

こんな感じのメールメッセージをオーウェンはUFOによる誘拐のレポートとして「提出」してもらい受け取りたいと思っています。

これはユーザのメールアドレスで、すでに変数\$emailに格納されています。

これは静的な文字列でも構いません。

これはオーウェンのメールアドレスで、こちらも静的な文字列で構いません。

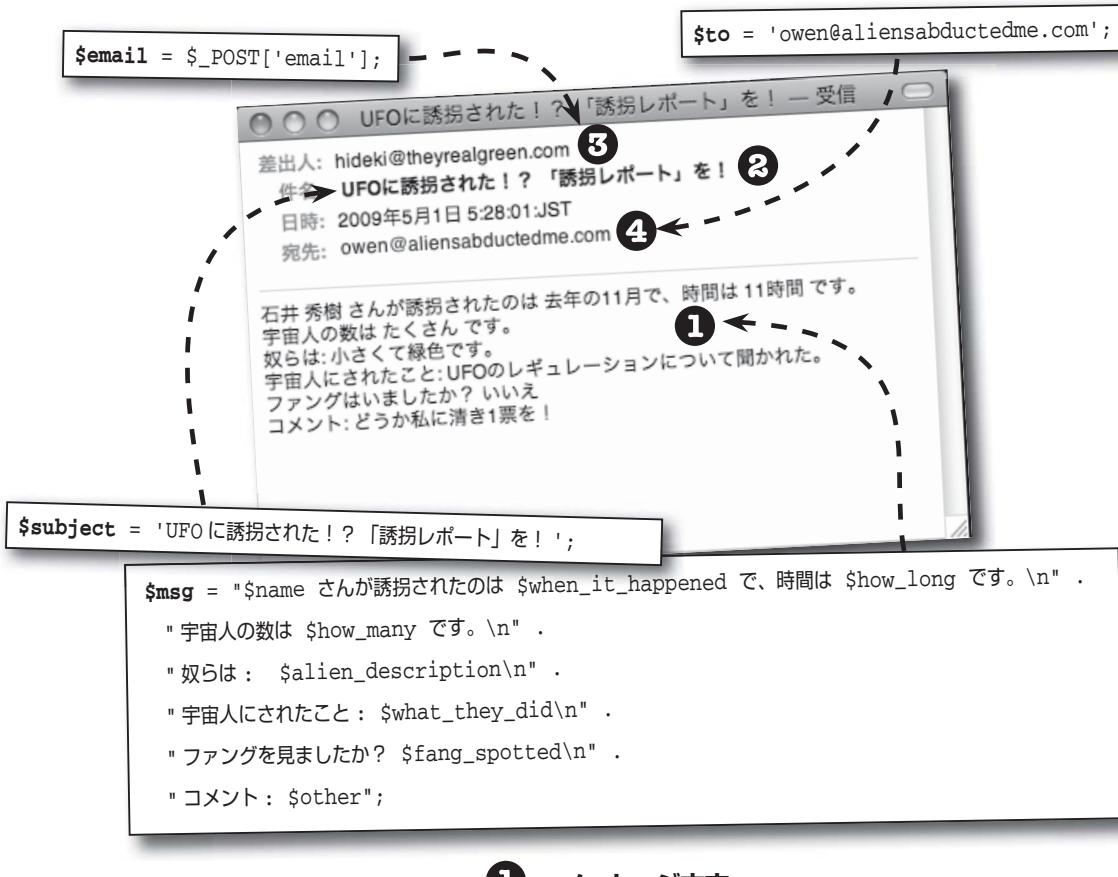


メール本文の文字列はすでに構築しましたが、これは変数\$msgに格納されています。

このメールメッセージの例で分かるように、コンテンツの大部分を占めるのはメッセージ本文です。しかもすでにできています。残っているのは、件名に加え "from" と "to" のそれぞれのメールアドレスです…そしてもちろん、何とかしてこのメッセージをPHPを使って実際に飛ばす必要があります！

変数はメールの要素や部品を格納しています

メッセージ本文はもう\$msgにあります。相変わらずメッセージの件名と"from"と"to"それぞれのメールアドレスがありません。件名と"to"アドレスについては新しい変数に静的なテキストを入れておけばよいでしょう。一方で"from"は変数\$emailに格納してあります。本章の始めのほうでフォームを処理するコードはすでに書いてあったおかげです。



① メッセージ本文

メールに必要な情報はすべて集まりました。
準備完了です。

② メッセージのタイトル

③ 送り主のメールアドレス(メッセージのFROM)

④ 受け取り手のメールアドレス(メッセージのTO)

PHP でメールメッセージを送る

さあ PHP コードで本当にオーウェンにメールメッセージを送る準備が整いました。送るには PHP の組み込み関数 `mb_send_mail()` を使います。この関数はパラメタとして提供した情報をもとにメッセージを送ります。

```
メールアドレスの
"to" です。          メッセージの件名です。
mb_send_mail($to, $subject, $msg);           ↑
                                                メッセージ本文です。
```

関数 `mb_send_mail()` には以上の3つの情報が必要ですので、常に用意しなければなりません。実は、メールの「from」アドレスは絶対必要ではないのですが、入れておくのはやっぱりよいことです。`mb_send_mail()` 関数を呼び出す際に「from」フィールドを指定するには、関数の実引数をもう1つ追加します。文字列の連結も一緒に使っておきます。

```
mb_send_mail($to, $subject, $msg, 'From:' . $email);
```

```
$to = 'owen@aliensabductedme.com';
$subject = 'UFOに誘拐された！？誘拐レポート'を！';
$msg = "$nameさんが誘拐されたのは $when_it_happened で、時間は $how_long です。\\n" .
"宇宙人の数は $how_many です。\\n" .
"奴らは： $alien_description\\n" .
"宇宙人にされたこと： $what_they_did\\n" .
"ファンを見ましたか？ $fang_spotted\\n" .
"コメント： $other";
```

```
$email = $_POST['email'];
```

メールメッセージの各要素は、
変数を使って `mb_send_mail()`
関数に渡されます。

そうです。立て続け
に2つのエスケープ
文字です！

PHP の
`mb_send_mail()`
関数はスクリプトの中
からメールメッセージを
送ります。

メールの送り主のアドレスを
指定する場合は、メールアドレスの
前に必ず 'From: ' というテキストを
付けておかなければいけません。

ドット演算は 'From: ' とオーウェン
のメールアドレスを連結するには、
ここでもやっぱり便利です。

素朴な疑問 に答えます

Q: メールの「from」アドレス以外でメールメッセージの一部として指定することのできるものがありますか？

A: あります。CC転送やBCC転送を「from」送信元と同様に指定することができます。
"From: " の代わりに単に "Cc: " や "Bcc: "
を使えばよいだけです。「from」とCC転送を同時に使いたい場合は、復帰改行文字を組み合わせ
("¥r¥n)を使って、これらを区切らなければなりません。こんな感じになります。

"From: " . \$from . "¥r¥nCc: " . \$cc
ここでは¥rと¥nというエスケープ
文字を使っているので二重引用符で
なければなりません。



それで実際には
どういう風にmb_
send_mail()関数を
使うんだい？

まずコード系を指定します[†]。

日本語のメッセージをメールで飛ばす場合、まず今使っている内部コード系をPHPに教えてあげる必要があります。デフォルト設定と等しいため、教える必要がないという場合もあり得ますが、常に明示した方が、後でスクリプトを読む人にも親切で、お行儀のよいコーディングスタイルです。

```
mb_internal_encoding("UTF-8");
```

あとはmb_send_mail()を呼び出すコードをスクリプトに単に追加するだけです。

関数mb_send_mail()を呼び出すコードの行がメールメッセージの送信に必要なすべてです。このコードはメールの変数を作った後でスクリプト上に書かなければいけませんが、それさえ守れば大丈夫です。これがオーウェンのreport.phpスクリプトの完全なコードで、mb_send_mail()関数の呼び出しがちゃんとあります。

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=shift-jis" />
<title>UFOに誘拐された！？「誘拐レポート」を！</title>
</head>
<body>
<h2>UFOに誘拐された！？「誘拐レポート」を！</h2>
<?php
$name = $_POST['lastname'] . ' ' . $_POST['firstname'];
$when_it_happened = $_POST['whenithappened'];
>Show_long = $_POST['howlong'];
>Show_many = $_POST['howmany'];
$alien_description = $_POST['aliendescription'];
$what_they_did = $_POST['whattheydid'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];

$to = 'owen@aliensabductedme.com';
$subject = 'UFOに誘拐された！？『誘拐レポート』を！';
$msg = "$nameさんが誘拐されたのは $when_it_happened で、時間は $Show_long です。\\n" .
"宇宙人の数は $Show_many です。\\n" .
"奴らは： $alien_description\\n" .
"宇宙人にされたこと： $what_they_did\\n" .
"ファンが見ましたか？ $fang_spotted\\n" .
"コメント： $other";
mb_internal_encoding("UTF-8");
mb_send_mail($to, $subject, $msg, 'From:' . $email);

echo '情報提供ありがとうございます。<br />';
echo '誘拐されたのは' . $when_it_happened;
echo 'で、時間は' . $Show_long . 'です。<br />';
echo '宇宙人の数は' . $Show_many . 'です。<br />';
echo '奴らは：' . $alien_description . '<br />';
echo '宇宙人にされたこと：' . $what_they_did . '<br />';
echo 'ファンが見ましたか？' . $fang_spotted . '<br />';
echo 'コメント：' . $other . '<br />';
echo 'メールアドレスは' . $email;
?>
</body>
</html>
```

内部コード系を設定し、
メールメッセージを
飛ばすところ。

フォームの「提出」が成功
したことを見たHTMLの
Webページをその場で
生成するところ。

\$_POST配列のフォーム
データをすべて取ってきて、
個々の変数に貼り付ける
ところ。

このスクリプトをテストして
みる時はメールアドレスを自分
のものに書き換えることを
忘れないで下さい。

メールメッセージの異なる
要素を組み立ててオーウェンに
送れるようにするところ。



[†] 訳注：日本語のメールを送る場合に必要なため追記しました。



試運転

オーウェンのスクリプトを完成させて試してみます。

新しい3つのemail変数(\$to, \$subject, \$msg)をreport.phpスクリプトに追加し、同時にmb_internal_encoding()関数の呼び出しとmb_send_mail()関数の呼び出しも追加します。\$to変数には自分のメールアドレスを設定します。オーウェンのままではありません！できあがったらスクリプトをWebサーバにアップロードし、ブラウザで開いてみます。フォームを誘拐に関する情報で埋めたら、誘拐のレポートボタンをクリックしてフォームを「提出」します。数秒待ってメールの受信ボックスにメッセージが来ているかどうかチェックしてみましょう。

The screenshots illustrate the following steps:

- Form Submission:** The first screenshot shows a web browser with a form titled "UFOに誘拐された！？「誘拐レポート」を！". It contains fields for name (石井 亮樹), email (hideki@theyreal/green.com), date (去年の11月), time (11時頃), and a comment section. A dog's face is displayed below the form.
- Confirmation Page:** The second screenshot shows the same form with the text "情報提供ありがとうございます。" and "UFOに誘拐されたのは去年の11月で、時間は11時間です。宇宙人の数はたくさんです。奴らは: 小さくて緑色です。宇宙人されたこと: UFOのレギュレーションについて聞かれた。ファンダムはいましたか? いいえ。コメント: どうか私に清き1票を！" and the email address "メールアドレスは hideki@theyreal/green.com".
- Email Confirmation:** The third screenshot shows an incoming email from "hideki@theyreal/green.com" with the subject "UFOに誘拐された！？「誘拐レポート」を！". The email body includes the same information as the confirmation page.
- Text Overlay:** A large text box on the right side of the screenshots states: "フォームのデータはきれいにフォーマットされ、メールメッセージとして送られます！" with a downward arrow pointing to the email confirmation.
- Bottom Text:** Below the screenshots, the text "動的に生成された確認画面のページでフォームが「提出」されたことをちゃんと確認できます。" is displayed with an upward arrow pointing to the confirmation page.



要注意

Webサーバ上でPHPの設定を変更しないと、メールがちゃんと飛ばないかもしれません。

もしmb_send_mail()関数がちゃんと動かないとしたら、問題はサーバ側でPHPインストール時にメールサポート設定[†]を正しくやっていないためかもしれません。Webサーバ上でメールを送信できるようにする設定の詳細についてはwww.php.net/mailをチェックしてみて下さい。

[†] 訳注：付録iiにphp.iniの設定に関する記述があります。パソコンをサーバとして使用している場合、[mail function]の部分を正しく書き換えたかを確認して下さい。



祐子さんは
最近UFOに
誘拐されました。

オーウェンにメールが来始めました

オーウェンはUFOによる誘拐の情報をちゃんと受け取れるようになってウキウキしてきました。今度はWebフォームから直接メールの受信ボックスに届きます。もう誰かがオーウェンの犬を目撃したかどうか聞けるのかと心配する必要はありません。オーウェンに連絡をくれた人のメールアドレスはわかっているのです。さらに良いことにレスポンスのメールに目を通すこと自体は楽しみにもなっています。

UFOに誘拐された！？「誘拐レポート」を！

UFOに誘拐された時の様子を教えてください。

例：
名前：
性別：
年齢：
e-mailアドレスは？
いつですか？
何回見ましたか？
何人見ましたか？
どんな感じですか？
何をされましたか？
聞いたり見たりしましたか？
はい・いいえ□

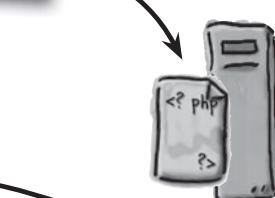
UFOに誘拐された！？「誘拐レポート」を！

UFOに誘拐された時の様子を教えてください。
誘拐されたのは3日前で、時間は午前です。
平日は寝てました。
取扱いはいたしません。
平日にはよくお出でになつてました。
ちょっとお出でになつてました。
ファンでいたことがあります。
コメント欄の文を見たいと思います。
連絡をください。
メールアドレスはyoko@grape-lab.com

祐子さんは
フォームを
「提出」しました。

```
<form action  
= "report.php"  
...>
```

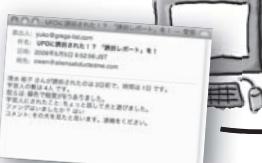
<form>タグのaction属性により report.phpスクリプトを起動しフォームのデータを処理します。



PHPスクリプトは確認画面のHTMLページを動的に生成します。

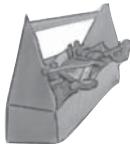
PHPスクリプトはメールメッセージを生成し、それをオーウェンに飛ばすこともします。

すごい！こういう誘拐レポートのメールがあれば、きっとファンを見つけられる。



オーウェンはUFOによる誘拐に関するレポートを、自分で作ったフォームを通して受け取ることができるようになって幸せです。





PHP & MySQL 道具箱

1章では PHP を利用してオーウェンの Web フォームに命を吹き込むやり方を学びました。
以下すべて習得したことです…

PHP

サーバ側のスクリプト言語で Web ページコンテンツをサーバで操作してからクライアントのブラウザに配信することを可能とする。

PHPスクリプト

PHPコードを含むテキストファイルで、Webサーバで仕事をする。

<?php ?>

PHPスクリプトにおいて PHP コードを囲むタグ

`mb_send_mail()`

日本語でメールを飛ばすための PHP 関数。パラメタとしてメールの件名、メールの本文テキスト、送り先アドレスを取る。(オプションで From アドレスを指定することもできる。)この関数を使用する前に `mb_internal_encoding()` で使用しているコード系を指定する。

クライアント側

クライアントの Web ブラウザ単独で解釈されるもの。

サーバ側

クライアントのパソコンではなく、Web サーバで解釈されるもの。

変数

データの格納領域。PHP では変数はドル記号で始まらなければならない。例えば `$variable_name`。

`$_POST`

フォームのデータを保持する特別な変数。

echo

出力をブラウザのウィンドウに送る PHP コマンド。構文は次の通り。

```
echo 'Hello World';
```

配列

値の組を格納するデータ構造。各値はインデックスを持ち、それを使ってデータにアクセスする。

エスケープ文字

入力が難しいかまたは他のコードとの混乱を来たす恐れのある文字を PHP コードで表現するために使われる文字。例(改行): `\n`。

2章 MySQLとつなげる

お互いに接続するには？

Webサイトの
接続設定につなぐ前に
Web間をつなぐプラグイン
がいるわね。

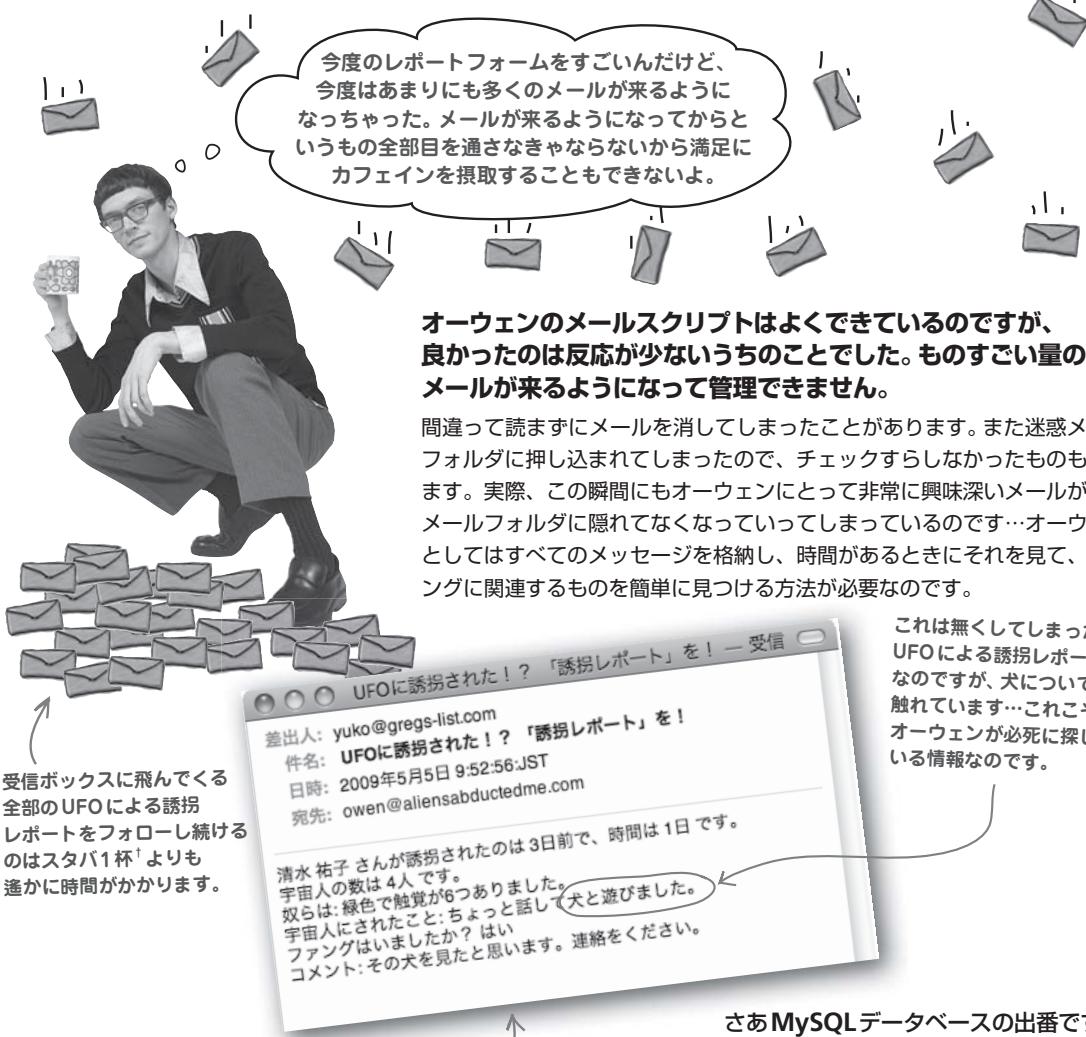
自分のWeb
アプリケーションは
あとの人の近くに置かない
ことにするわ。



構築を始める前にどのように接続するか知っておくというのは良いことです。

最初の自前のPHPスクリプトはできあがり、ちゃんと動いています。でもフォームが結果としてメールになって飛んでくるというのは、実はちっともよくありません。フォームの結果を保存する方法が要ります。そして必要な間は結果をとっておくことができて、見たくなったら結果を取ってくることができたら素敵です。MySQLデータベースを使えばデータを安全に保存しておくことができます。でも前章で作ったPHPスクリプトをMySQLデータベースのところまで持ってこなければ何も始まりません。

オーウェンのPHPフォームはちゃんと動きますが、ちゃんとしすぎです…



オーウェンはこのようなメッセージを一箇所にまとめて安全に保存しておく必要があるのです。まとめて安全に保存しておく必要があるのでふるいにファングの目撃情報かもしれないでふるいにかけることができます。

さあMySQLデータベースの出番です…

蛇足ですが、多くの人はMySQLを「マイ・エス・キュー・エル」と最後の3文字はアルファベット読みで発音します[‡]。

[†] 訳注：原著はbuzz coffeeです。Head First シリーズではStarbuzz coffee (Starbucks coffeeのパロディ)がよく登場します。

[‡] 訳注：SQLを「シケル（シークエル）」と発音する流儀もあります。この流儀に従えば「マイ・シーケル」となります。どちらも使われているようです。

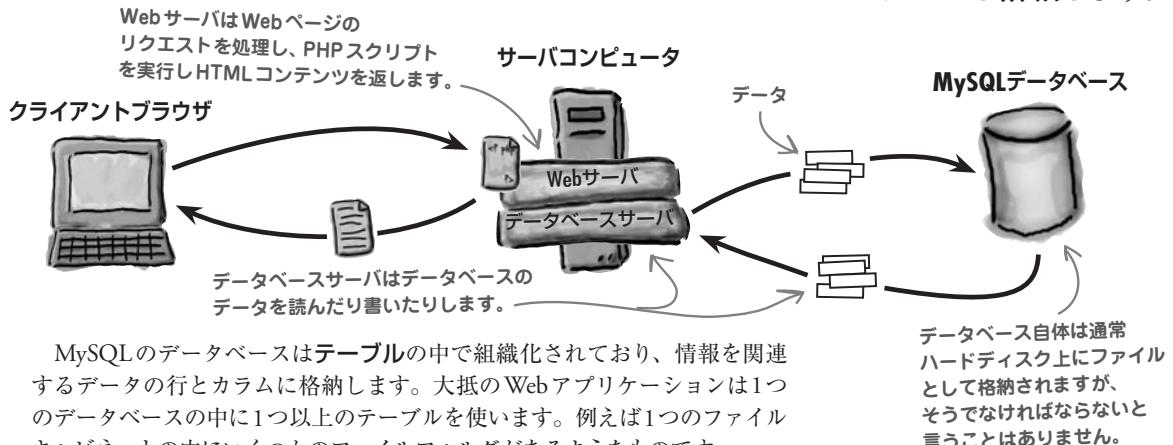
MySQLはデータの格納に優れています

オーウェンはメールの受信ボックスよりも安全にUFOによる誘拐レポートのデータを格納する方法が本当に必要になっています。オーウェンに必要なものはデータベースと言って特上の超組織化電子ファイルキャビネットとでも表現しておきましょう。データベースの情報は極めてうまく組織化されているため、必要なときに必要な情報をピンポイントで取り出すことができます。

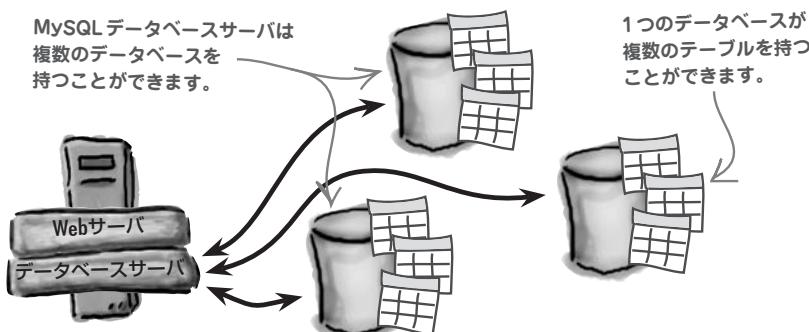
データベースはデータベースサーバという特別なプログラムで管理されています。今回はMySQLをデータベースサーバとして使います。データベースサーバとの対話には専用の言語が必要で、それがSQLです。データベースサーバは通常Webサーバと共に同じコンピュータ上で動作し、データの読み書きを協調して行い、Webページを配信します。

MySQLの「SQL」というのは Structured Query Language(構造化問い合わせ言語)の頭文字です。

MySQLは
データベース
テーブルの中に
データを格納します。



MySQLのデータベースはテーブルの中で組織化されており、情報を関連するデータの行とカラムに格納します。大抵のWebアプリケーションは1つのデータベースの中に1つ以上のテーブルを使います。例えば1つのファイルキャビネットの中にいくつかのファイルフォルダがあるようなものです。



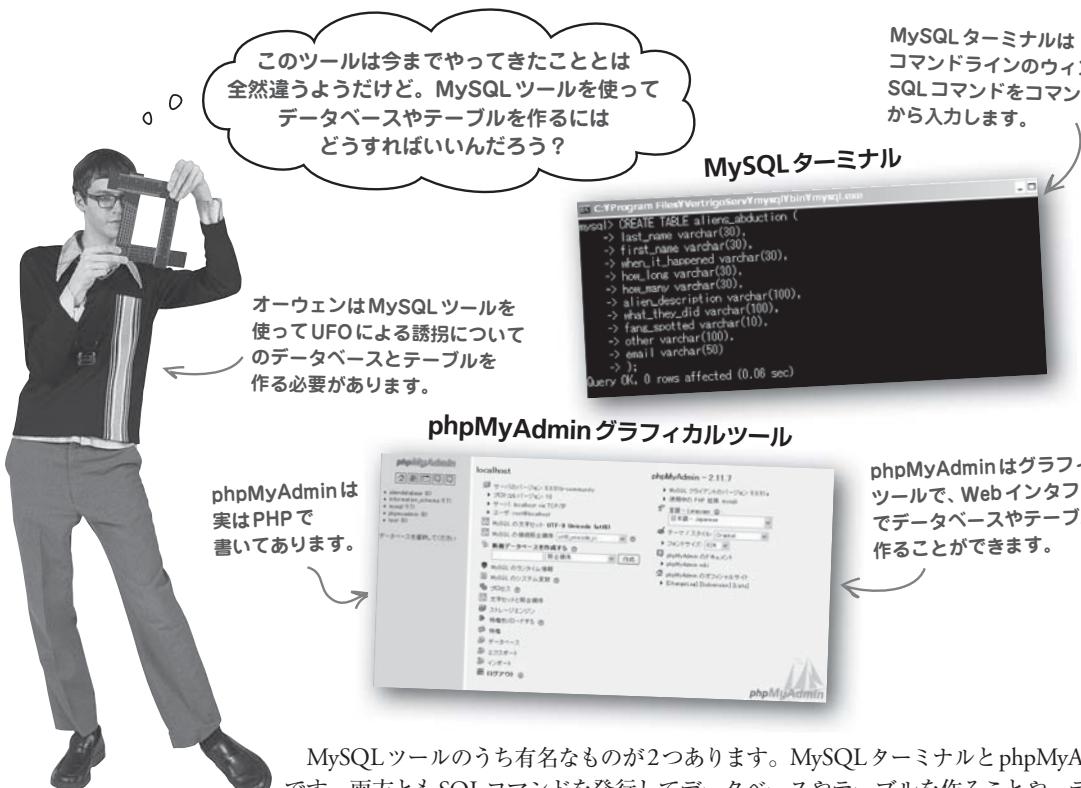
UFOによる誘拐のデータをMySQLデータベースに安全に格納できれば、オーウェンはみんなから寄せられたレポートのうち、ファンガードに関する質問に「はい」と答えたものについてだけ好きなときに解析することができます。必要なものはデータベースサーバと話すためにちょっとSQLを使うことだけです。

SQLはMySQL
データベースと
コミュニケーションを
とるために使う
問い合わせ言語です。

オーウェンにはMySQLデータベースが必要です

決まりました。MySQLデータベースは良さそうです。オーウェンはUFOによる誘拐データを格納するものが必要なのです。そこでreport.phpスクリプトを修正してデータをメールで送る代わりにテーブルに格納することにします。テーブルはデータを安全かつ健全にとっておいてくれます。データは誘拐された人から自動的に注ぎ込まれますから、オーウェンはデータをふるいにかけ、ファングを目撃したかもしれない人だけをより分けることに時間を注げばよいのです。しかし、まず最初にやらなければならないこと…それはデータベースです！

MySQLデータベースを作るにはMySQLデータベースサーバと特別なソフトウェアツールが必要です。なぜならWebサーバと異なり、データベースサーバはSQLコマンドを使ってコミュニケーションをとらなければならぬからです。



MySQLデータベースと
テーブルを作るには、
MySQLデータベース
サーバとコミュニケーション
しなければなりません。

MySQLターミナルは
コマンドラインのウィンドウで
SQLコマンドをコマンドライン
から入力します。

phpMyAdminはグラフィカル
ツールで、Webインターフェース
でデータベースやテーブルを
作ることができます。

ページをめくる前にMySQLデータベースサーバがインストールされているか確認しなければなりません。



これがないとオーナーを助けることはできません！MySQLデータベースサーバがインストールされていて、ちゃんと動くのであれば先へ進みましょう。そうでなければ付録iiへ移ってインストール手順に従って下さい。もしWebホスティングサービスを使っていて、そこがMySQLを提供してくれているのであれば、その会社にインストールしてもらうよう頼んで下さい。MySQLデータベースサーバにアクセスするには、いくつかの情報が必要になります。これらの情報は後でまた必要になりますから、それがどんなものであるかを今ここで確認しておきましょう。それぞれ書き込んだらチェックしておきます。

- MySQLサーバの場所(IPアドレスまたはホスト名):
 - データベースユーザ名:
 - データベースパスワード:
- これらすべてをチェックする必要があります。

この本が悪者の手に落ちることを恐れるのであれば、この部分は書き込まなくてもよいでしょう。

MySQLデータベースサーバの情報がそろったら、残っているのはサーバが立ち上がっていて、ちゃんと動いているかどうかを確認することです。以下のいずれかをチェックすることで、MySQLサーバにちゃんとアクセスできていることを確認できます。

- どれか1つだけをチェックすれば十分です。
- MySQLターミナルを使ってMySQLサーバにちゃんとアクセスできた。
 - phpMyAdminを使ってMySQLサーバにちゃんとアクセスできた。
 - を使ってMySQLサーバにちゃんとアクセスできた。
- 他に何か使えるMySQLツールを見つけたら、書き留めておきましょう。

日本語環境の確認[†]

MySQLに日本語データを入力する場合には、データベースの文字コードセットがUTF-8など日本語に対応していないかもしれません。1章でPHPの文字コードセットをUTF-8としましたので、MySQLもUTF-8にそろえておくとよいでしょう。MySQLツールで以下のコマンドを叩いて文字コードセットがUTF-8となっているかを確認して下さい。

```
SHOW VARIABLES LIKE "char%";
```

このコマンドがどんな意味なのかについては9章で説明します。このコマンドの結果、latin1というような他の文字コードセットが残っていたらダメです。設定の変更方法を含め、詳しい説明は付録iiiを参照して下さい。

[†] 訳注：日本語入力に関する必要事項ですので追記しました。

MySQL のデータベースとテーブルを作る

MySQL のインストール設定によってはデータベースはもう作ってあります。もしなければ、CREATE DATABASE という SQL コマンドを使って MySQL ターミナルからデータベースを作る必要があります。まず始めにコマンドラインウィンドウで MySQL ターミナルを開きます。単に mysql と打ち込めば大抵動きます。コマンドプロンプトが mysql> に変わったらターミナルにちゃんと入った証拠です。

UFO による誘拐のデータベースを新しく作るには、次のように入力します。

```
CREATE DATABASE aliendatabase;
```

こんな感じです。

MySQL サーバは通常
コマンドが成功した
かどうかを知らせる
ためのレスポンスを
返します。

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> CREATE DATABASE aliendatabase;
Query OK, 1 row affected (0.01 sec)
mysql>
```

ターミナルを利用する場合、各コマンドの最後にセミコロンを打たなければなりません。

データベースの中にテーブルを作る前に、新しいデータベースが選択されているかどうかを確認する必要があります。次のコマンドを入力します。

```
USE aliendatabase;
```

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> USE aliendatabase;
Database changed
mysql>
```

テーブルを作る SQL コードはもう少し複雑です。このテーブルにどのようなデータが格納されるかを正確に書き尽くさなければなりません。SQL コマンドを実際にターミナルに入力する前によく見ておきましょう。

```
CREATE TABLE aliens_abduction (
    last_name varchar(30),
    first_name varchar(30),
    when_it_happened varchar(30),
    how_long varchar(30),
    how_many varchar(30),
    alien_description varchar(100),
    what_they_did varchar(100),
    fang_spotted varchar(10),
    other varchar(100),
    email varchar(50)
);
```

これが SQL コマンドでテーブルを新しく作ります。

これらの要素はすべて、テーブルにどのようなデータが格納されるかについての詳細な情報です。

MySQL ターミナルに入力されたすべての SQL コマンドはセミコロンで終わらなければなりません。

新しいテーブルを本当に作るためには、大文字でCREATE TABLEコマンドをMySQLターミナルに入力します(コマンドのコードはWebサイト<http://www.oreilly.co.jp/catalog/9784873114446/>にあります)。このコマンド入力に成功したら、できたての新しいテーブルaliens_abductionを手に入れたことになります。

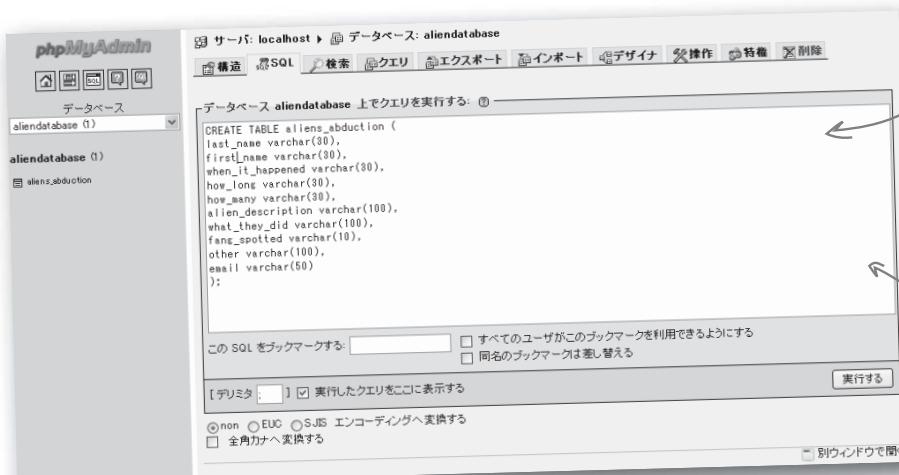
MySQL サーバから
「Query OK」という
レスポンスが返って
きたら、何一つ問題
なくテーブルができ
たとわかります。



```
C:\Program Files\VertrigoServer\mysql\bin\mysql.exe
mysql> CREATE TABLE aliens_abduction (
    -> last_name varchar(30),
    -> first_name varchar(30),
    -> when_it_happened varchar(30),
    -> how_long varchar(30),
    -> how_many varchar(30),
    -> alien_description varchar(100),
    -> what_they_did varchar(100),
    -> fang_spotted varchar(10),
    -> other varchar(100),
    -> email varchar(50)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql>
```

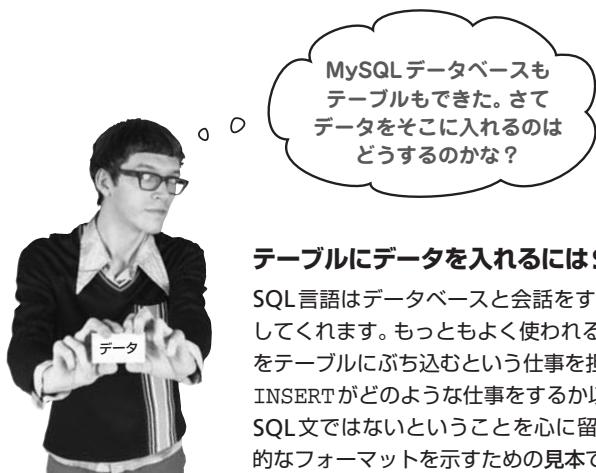
MySQLにWebベースのツールphpMyAdminがインストールされている場合は、データベースやテーブルにグラフィカルにアクセスすることができます。phpMyAdminユーザインタフェースを使えばクリックしながらデータベースやテーブルを作ることができます。もちろんMySQLターミナルと同様にSQLコマンドを直接打ち込むこともできます。phpMyAdminでSQLタブをクリックするとテキストボックスにアクセスでき、MySQLターミナルと同様に操作できます。



ここにMySQLターミナル
に入力したのと同じように
コマンドを打ち込むことが
できます。「実行する」を
クリックすれば実行する
ことができます。

SQLコードを入力したら、
このボタンをクリックして
テーブルを作ります。

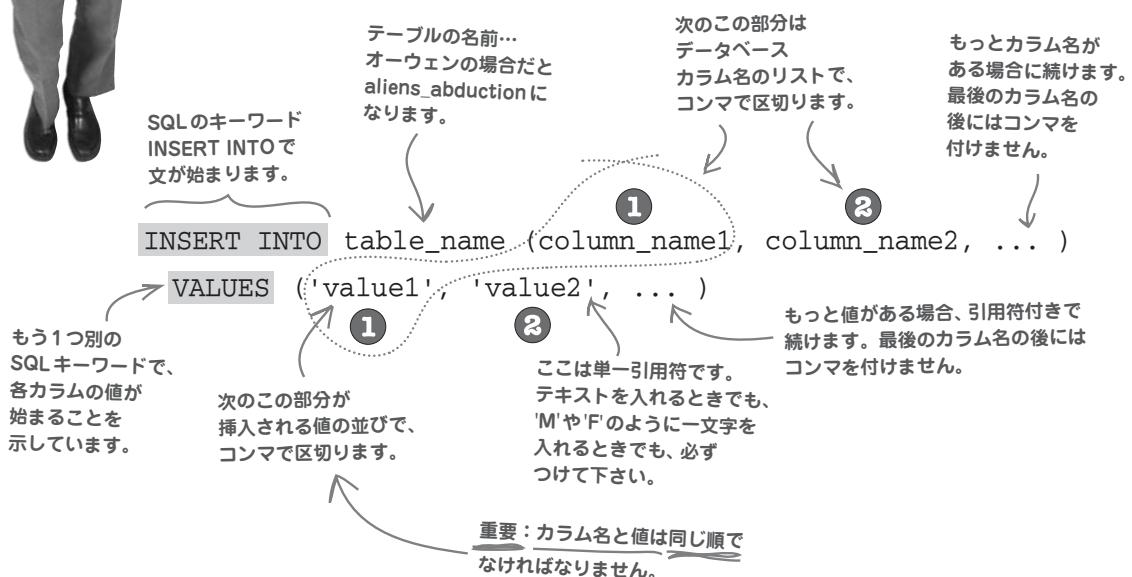
つまりphpMyAdminアプリケーションのSQLタブを使えば、あたかもMySQLターミナルを使っているかのようにSQLコマンドを発行する方法が提供されるのです。



テーブルにデータを入れるにはSQLのINSERT文を使います。

SQL言語はデータベースと会話をするためのあらゆる種類のイケてる文を提供してくれます。もっともよく使われる文の1つにINSERTがありますが、データをテーブルにぶち込むという仕事を担っています。

INSERTがどのような仕事をするか以下の文で見てみましょう。この文は本当のSQL文ではないということを心に留めておいて下さい。これはINSERTの一般的なフォーマットを示すための見本です。



この文において最も注意すべき重要なことは、2番目の括弧内に現れる値は、データベースカラム名と同じ順に現れなければならないということです。これによりINSERT文はデータを挿入する際に値とカラムをマッチさせているのです。

INSERT文の動作

オーウェンが新しく作ったテーブルaliens_abductionにUFOによる誘拐に関するデータをINSERT文を使って格納する方法は次の通りです。

ここはデータが挿入される
テーブル名であって、データ
ベース名ではありません。

カラム名は、最初の
カッコ内にコンマ
区切りで書きます。

要注意

順番が大事！

挿入する値は、カラム名の
順序と完全に一致して並ん
でいなければなりません。

```
INSERT INTO aliens_abduction (last_name, first_name,
 ③ ④ ⑤ ⑥
when_it_happened, how_long, how_many, alien_description,
 ⑦ ⑧ ⑨ ⑩
what_they_did, fang_spotted, other, email)
 ① ② ③ ④ ⑤
VALUES ('清水', '祐子', '3日前', '1日', '4人',
 ⑥ ⑦
'緑色で触覚が 6つありました。', 'ちょっと話して犬と遊びました。',
 ⑧ ⑨
'はい', 'その犬を見たと思います。連絡下さい。',
 ⑩
'yuko@gregs-list.net')
```

各カラムに設定する値は、
2番目のカッコ内に
コンマ区切りで書きます。

PHPの文と違って、
SQL文はPHPコードの中
で使われる時には
セミコロンで終了しません。

これらすべての値は
テキストであってメンバ名
ではありませんので、各項目
は単一引用符で囲みます。



自分で考えてみよう

これは
カラム名です。

aliens_abductionテーブルを以下に示しますが、まだデータは何も入れていません。祐子さんがレポートしたUFOによる誘拐に関するデータをテーブルに書き込んでみて下さい。上のテーブルのデータを書き写しても良いですし、スペースが足りなければ矢印で示しても良いです。

aliens_abduction

last_name	first_name	when_it_happened	how_long	how_many	alien_description	what_they_did	fang_spotted	other	email

自分で考えてみよう の答え

aliens_abductionテーブルを以下に示しますが、まだデータは何も入れていません。祐子さんがレポートしたUFOによる誘拐に関するデータをテーブルに書き込んでみて下さい。上のテーブルのデータを書き写しても良いですし、スペースが足りなければ矢印で示しても良いです。

yuko@gregs-list.net

緑色で触覚が6つありました。

ちょっと話して
犬と遊びました。その犬を見たと
思います。連絡を
下さい。**aliens_abduction**

last_name	first_name	when_it_happened	how_long	how_many	alien_description	what_they_did	fang_spotted	other	email
清水	祐子	3日前	1日	4人			はい	▼	▼

素朴な疑問に答えます

Q: データベースとテーブルの違いがよくわかりません。
どちらも単にデータを格納するためのものではないのですか?

A: その通りです。テーブルというのは、データベースにデータを入れるために関連したグループで分割する方法を提供するものです。1つの巨大な入れ物にデータが入っているという状態を回避できます。例えば1つの巨大な入れ物に靴を投げ込んでおくと、シューズケースに一組ずつそろえて靴を入れておくとの違いです。1つの大きな入れ物がデータベースであり、小さなシューズケースがテーブルです。データはテーブルに格納され、テーブルはデータベースに格納されているわけです。

Q: MySQLターミナルというのは正確には何ですか?コンピュータのどこにあるのでしょうか?

A: MySQLターミナルとはコマンドラインインターフェースでMySQLサーバにアクセスする技法です。多くの場合MySQLターミナルは単一のプログラムではなく、「汎用の」ターミナルプログラムからコマンドラインを通して確立した接続なのです。ここで汎用のターミナルプログラムというのはMax OS Xの場合ターミナルアプリケーションです。MySQLターミナルにどのようにすればアクセスできるかは、使ってているOSの違い、MySQLサーバがローカルにあるのかリモート(自分の使っている以外のコンピュータ)にあるのかによって様々に異なります。付録iiではMySQLターミナルにアクセスするやり方についての詳細を説明しています。

Q: ではphpMyAdminの方は何でしょうか?これはどこにあるのですか?

A: MySQLターミナルと異なり、phpMyAdminはWebベースのアプリケーションで、MySQLデータベースへのアクセスを可能としています。こちらはPHPアプリケーションですから、Webサーバごとにアクセスするわけです。ローカルのクライアントにインストールしてあるのではありません。多くのWebホスティング会社ではphpMyAdminを標準のMySQLホストプランの一部として提供していますので、すでにインストールされているかもしれません。もしくとも、自分でphpMyAdminをダウンロードしてインストールすることができます。www.phpmyadmin.netから無料でダウンロードできます。ただしphpMyAdminはWebサーバにインストールしてMySQLデータベースにアクセスできるように設定しなければならないことに注意して下さい。これはPHPアプリケーションやMySQLアプリケーションのすべてについて言えることです。

Q: MySQLとphpMyAdminの両方が使えるとしたら、どちらが良いでしょう?

A: これは完全に個人的な好みの問題です。phpMyAdminの良い点は、SQLコマンドを打つことなく、データベースやテーブルを視覚的に捉えることができることです。手軽にお気楽にSQLを使えるので、ちょっとしたことでも必ずSQLコマンドを叩かなくともみます。でも今はMySQLのデータとSQLコマンドを使ってどのように会話をしているのかをちゃんと理解することに集中するのはよいことではないでしょうか?そのような意味ではどちらのツールもちゃんと使えます。



試運転

SQLのINSERT文を使ってデータベースにUFOによる誘拐の目撃情報を格納する

MySQLターミナルやphpMyAdminのSQLタブなどMySQLツール[†]を使って、UFOによる誘拐に関するINSERT文を打ち込みます。例としてここでは、清水祐子さんの誘拐についてのINSERT文を使います。

```
INSERT INTO aliens_abduction (last_name, first_name,
when_it_happened, how_long, how_many, alien_description,
what_they_did, fang_spotted, other, email)
VALUES ('清水', '祐子', '3日前', '1日', '4人',
'緑色で触覚が6つありました。', 'ちょっと話して犬と遊びました。',
'はい', 'その犬を見たと思います。連絡を下さい。',
'yuko@gregs-list.net')
```

データベース aliendatabase 上でクエリを実行する: ⑦

```
INSERT INTO aliens_abduction (last_name, first_name, when_it_happened, how_long, how_many, alien_description, what_they_did, fang_spotted, other, email)
VALUES ('清水', '祐子', '3日前', '1日', '4人',
'緑色で触覚が6つありました。', 'ちょっと話して犬と遊びました。',
'はい', 'その犬を見たと思います。連絡をください。',
'yuko@gregs-list.net');
```

挿入した行数: 1 (クエリの実行時間 0.0007 秒)

phpMyAdminでINSERT文を実行すると、alien_abductionテーブルに新しいデータが1行追加されます。

INSERT文は成功したようです。データが正しく追加されたことを確認するにはどうすればよいと思いますか？書いてみましょう。

[†] 訳注：日本語を入力する関係でphpMyAdminを使って下さい。MySQLターミナルを使う場合の注意点は付録iiiにまとめています。

テーブルのデータを見るには SELECT を使う

テーブルにデータを挿入するのは簡単で、これがすべてです。しかしこれだけでは、ある意味全然安心できません。まだ本当にデータがテーブルに入ったのかどうかの確認ができるいないのです。これではまるで定期預金にお金を預けたのに、残高を知る手段がないようなものです。SELECT 文を使えば、データベースにあるテーブルの「残高を知る」ことができます。もっと正確に言うと、SELECT を使えばテーブルからデータのカラムを持ってくることができます。

SELECT に続けて参照したい
データのカラムを並べます。

```
SELECT columns FROM table_name
```

SELECT は常に特定のテーブルを
参照します。一般にデータベース
ではありません。

SELECT 文の FROM 部分はデータを
セレクトしてくるテーブルを SELECT
自体が知るためのものです。

SELECT 文に指定するカラムはすべてコンマで区切れます。テーブルにどれだけ多くのカラムがあるとも、SELECT に指定されたカラムのデータだけが返ってきます。次の SELECT 文は alien_abduction テーブルから UFO に誘拐された人の姓名だけをすべて取ってきます。

この 2 つのカラムのデータ
だけが、この SELECT 文に
よって返ってきます。

```
SELECT last_name, first_name FROM aliens_abduction
```

SELECT 文は alien_abduction
テーブルだけからデータを
引っ張ります。

INSERT をチェックするには、いくつかのカラムだけではなく、テーブル中のすべてのデータを素早く見る方法が必要でしょう。SELECT 文には、这样的ことをする簡単なやり方があります。

アスタリスク、または「星印」はテーブル
中の全カラムのデータを取ってくるという
SELECT 文への指示です。

```
SELECT * FROM aliens_abduction
```

* は「全部もってこい！」と
いう意味なので、カラムの
リストは必要ありません。



試運転

UFOによる誘拐のINSERT文が正しく動いたかどうか、テーブルのデータをSELECTして確認します。

MySQLツールを使ってSELECTで問い合わせを実行し、alien_abductionテーブルのすべての内容を見てみましょう。さっき挿入したデータが結果として新しい行にあるかを確認して下さい。

```
SELECT * FROM aliens_abduction
```

これらがカラムです。

last_name	first_name	when_it_happened	how_long	how_many	alien_description	what_they_did
清水	祐子	3日前	1日	4人	緑色で触覚が6つありました。	ちょっと話して犬と遊びました。

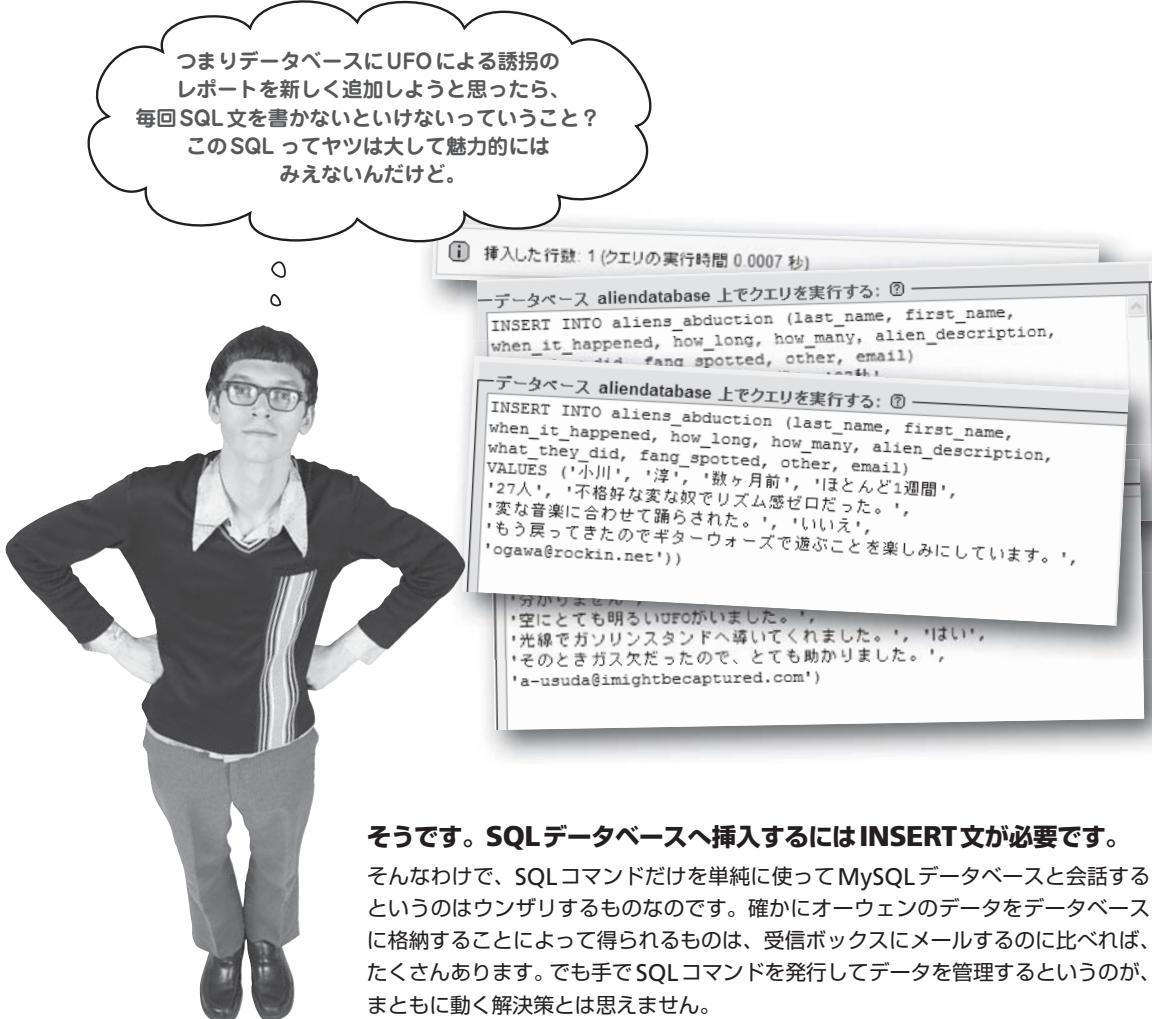
SELECTでの問い合わせによりテーブルに格納されているデータは1行であることがわかります。

各カラム名の下にあるのはそのカラムのデータです。

テーブルには
何行のデータが
あったでしょう?

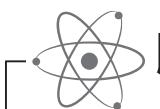
文字化けしていませんか？

日本語の部分は正しく表示されていますか？日本語データが????というように表示される、あるいは入力したにもかかわらず空欄になっている場合、文字コード系が正しく設定されていないことが原因と考えられます。phpMyAdminだとOKでMySQLターミナルだとうまくいかないといった場合も同様です。付録iiiを参照して設定をチェックして下さい。



そうです。SQLデータベースへ挿入するには **INSERT** 文が必要です。

そんなわけで、SQLコマンドだけを単純に使ってMySQLデータベースと会話するというのはウンザリするものなのです。確かにオーウエンのデータをデータベースに格納することによって得られるものは、受信ボックスにメールするのに比べれば、たくさんあります。でも手でSQLコマンドを発行してデータを管理するというのが、まともに動く解決策とは思えません。



脳力発揮

オーウエンのMySQLデータを挿入する問題はどうすれば解決できると思いますか？

PHPにウンザリ SQLの処理をやらせる

オーウェンの問題の解決するには、SQLを避けるのではなく、PHPの助けを借りてSQLを自動化すればよいのです。PHPでは、サーバ上でスクリプトコードの中からSQLを自動化文を発行することができます。ですからもうMySQLツールを使う必要はありません。つまりオーウェンのHTMLフォームからPHPスクリプトを呼び出し、「提出」されたらデータベースにデータを挿入するという処理をすればよいのです。eailもSQLツールも苦労も要りません！

オーウェンはメールのデータをデータベースに挿入するためにSQLのINSERT文を作成します。

HTML フォームはメールを生成し、オーウェンはそれを受け取ったら手でデータベースに追加しなければなりません。

report.html

HTML フォームを PHPスクリプトを呼出し
フォームのデータをデータベースに追加するよう依頼します。

```
データベース aliendatabase 上でクエリを実行する: ①
INSERT INTO aliens_abduction (last_name, first_name,
                               when_it_happened, how_long, how_many, alien_description,
                               what_they_did, fang_spotted, other, email)
VALUES ('清水', '祐子', '3日前', '1日', '4人',
       '緑色で触覚が6つありました。', 'ちょっと話して犬と遊びました。',
       'はい、この犬を見たと思います。連絡を下さい。',
       'yuko@gregs-list.net');
```

PHPなしと SQLのINSERT文を手で全部打ち込んでUFOによる誘拐レポートをデータベースに格納する必要があります。



PHPありだとフォームが「提出」された時に PHPスクリプトがINSERTを自動的に処理してくれます。

```
<?php
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensroot', 'aliendatabase')
or die('Error connecting to MySQL server.');
$query = "INSERT INTO aliens_abduction (last_name, first_name,
                                           when_it_happened, how_long, how_many, alien_description,
                                           what_they_did, fang_spotted, other, email)
VALUES ('清水', '祐子', '3日前', '1日', '4人', '' .
       '' 緑色で触覚が6つありました。', 'ちょっと話して犬と遊びました。', '' .
       '' はい、この犬を見たと思います。連絡を下さい。', '' .
       '' 'yuko@gregs-list.net')";
$result = mysqli_query($dbc, $query)
or die('Error querying database.');
mysqli_close($dbc);
?>
```

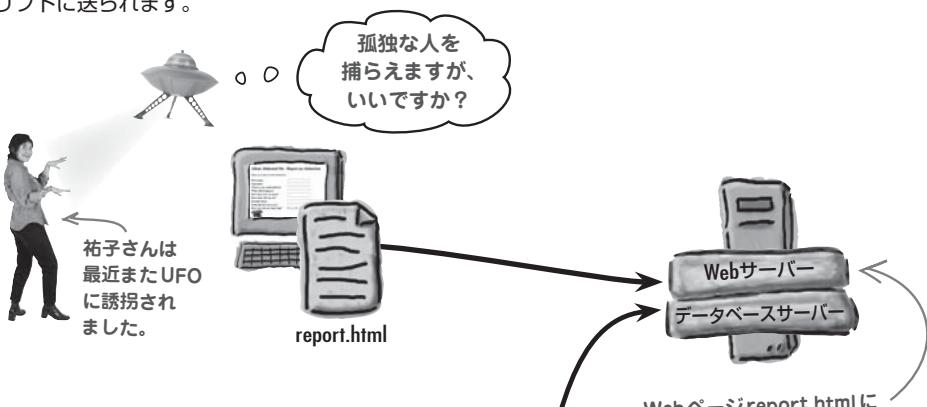
PHPスクリプトは、フォームのデータをデータベースに挿入するためにINSERT文を生成します…
オーウェンは手出し無用です！



PHPによりデータはオーウェンの Web フォーム上を渡り歩く

PHPはオーウェンのUFOによる誘拐のWebフォームを改良して、スクリプトからフォームのデータを直接データベースに送りつけるようにします。オーウェンのメールアドレスにデータを送りオーウェンが手で操作してそれを見るのではありません。このアプリケーションが正確にはどのように動いているのか絵を使ってよく見てみましょう。

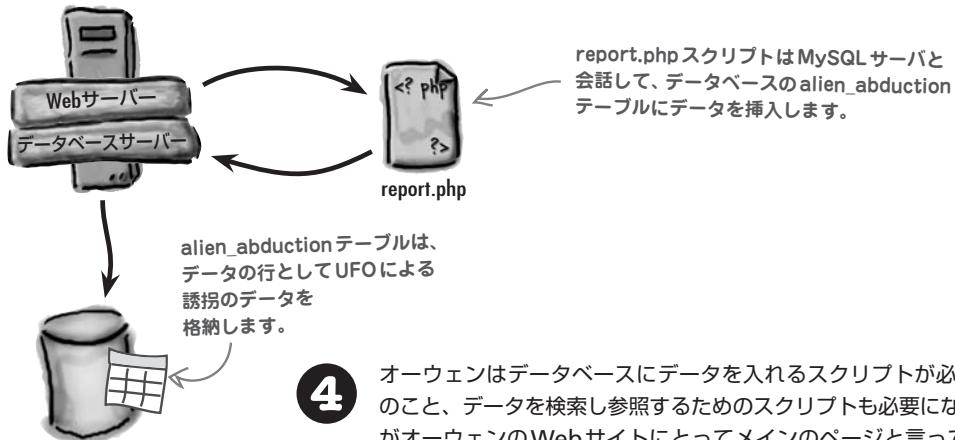
- ① 祐子さんはUFOによる誘拐のフォームを埋めて、「Report Abduction」ボタンを押して「提出」します。情報はWebサーバ上のreport.phpスクリプトに送られます。



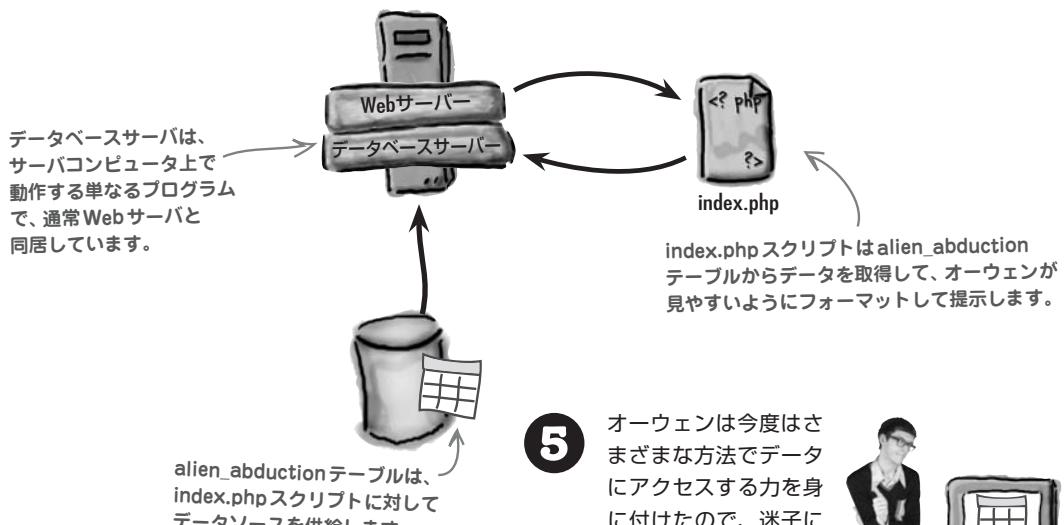
- ② ものすご～～くたくさんの他の人たちも絶え間なくフォームを「提出」します。



- 3** オーウェンのreport.phpスクリプトはMySQLデータベースにつながっていて、「提出」の度にSQLのINSERT文を使って情報を挿入します。



- 4** オーウェンはデータベースにデータを入れるスクリプトが必要なのはもちろんのこと、データを検索し参照するためのスクリプトも必要になります。実際これがオーウェンのWebサイトにとってメインのページと言っても良いでしょう。index.phpスクリプトはデータベースに接続し、UFOによる誘拐のデータを検索し、最後にオーウェンにそれを見せるのです。



- 5** オーウェンは今度はさまざまな方法でデータにアクセスする力を身に付けたので、迷子になったファングを見つけることに集中できるようになりました。



PHP からデータベースに接続する

PHP スクリプトから MySQL データベースへデータを格納したり取り出したりする前に、まずデータベースと接続しなければなりません。PHP から MySQL データベースへの接続は、色々な意味で MySQL ツールからデータベースにアクセスする方法と似ていますので、同種の情報が必要となります。本章の最初の方で埋めた3つのチェックボックスを覚えていていますか？あれと同じものがまた必要となります。今回はもう1つデータベースの名前も必要です。ではもう一度書き留めておきましょう。

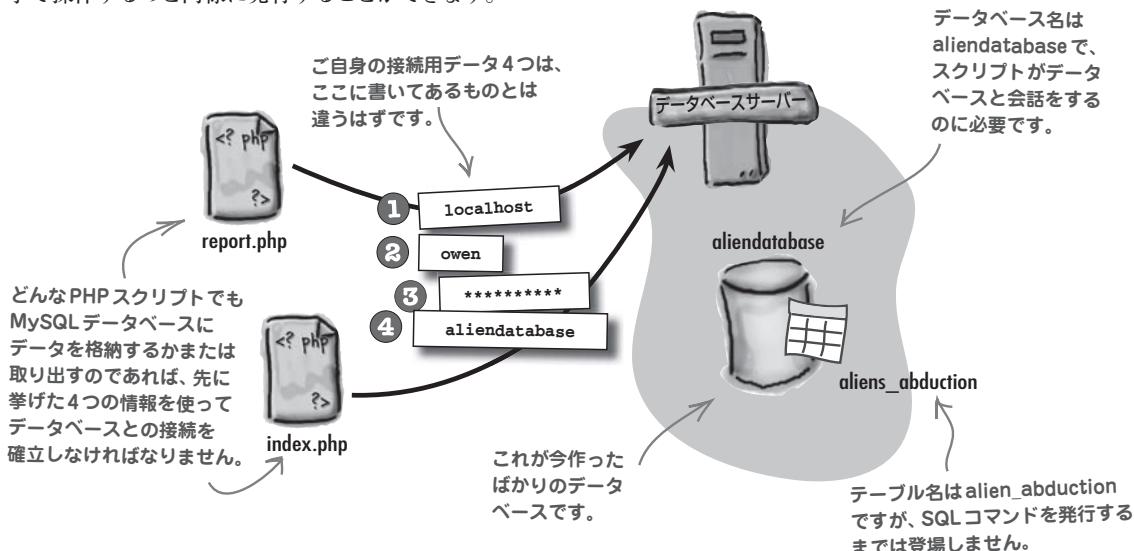
Web ホスティングサービス会社か
Web 管理者に聞けば分かる情報です。
もし Web サーバと MySQL データベース
サーバが同じマシン上で動いているので
あれば、「localhost」と書いても OK です。

- ① MySQL サーバの場所(IP アドレスまたはホスト名):
- ② データベースユーザ名:
- ③ データベースパスワード:
- ④ データベース名:

データベースサーバホストの場所、ユーザ名、パスワード、それとデータベース名のすべてを使わないと MySQL データベースに PHP スクリプトから接続を確立することはできません。ひとたび接続が確立すれば、スクリプトは SQL コマンドを、MySQL ツールを使って手で操作するのと同様に発行することができます。

さっき作ったデータベース名ですから、
aliendatabase です。もし何か理由が
あって他の名前をデータベースにつけた
とか、またはすでにあるデータベースを
使うと決めたとかいったことであれば、
代わりにその名前を使って下さい。

データベース名は
aliendatabase で、
スクリプトがデータ
ベースと会話する
のに必要です。



PHPスクリプトでデータを挿入する

PHPコードからMySQLへの問い合わせを発行するには、まずデータベースとの接続を確立する必要があります。次にPHPの文字列で問い合わせを構築します。問い合わせをデータベースサーバに対して渡すまでは、実際には発行されません。そして、データベースへの問い合わせが終わったら最後に接続を閉じます。これらすべての仕事はPHPスクリプトを通して運ばれます。以下にUFOによる誘拐のデータを新しい行として挿入する例を示します。

これら4つの値は
オーウェンのではなく、
自分のもの[†]に変えます。

```
<?php
    MySQLデータベース
    に接続します。
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool', 'aliendatabase')
    or die(' エラー：MySQL サーバとの接続に失敗しました。 ');
        ドメイン名の代わりにデータベースの位置として
        'localhost' を使えるかもしれません。

$query = "INSERT INTO aliens_abduction (last_name, first_name, when_it_happened, how_long, "
    "how_many, alien_description, what_they_did, fang_spotted, other, email) "
    "VALUES ('清水', '祐子', '3日前', '1日', '4人', '緑色で触覚が6つありました。', '',
    "'ちょっと話して犬と遊びました。', 'はい', 'その犬を見たと思います。連絡を下さい。', '',
    "'yuko@gregs-list.net')";

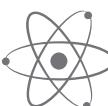
    PHPコードの文字列としてINSERTの
    問い合わせを作っています。

$result = mysqli_query($dbc, $query)
    or die(' エラー：データベースとの問い合わせに失敗しました。 ');

    INSERTの問い合わせをMySQL
    データベースに発行しています。
mysqli_close($dbc);
?
```

このあたりの単一引用符と
二重引用符の違いと、引用符
直後の空白文字にはマジで
注意して下さい！

これらの関数はWebサーバのバージョンが
4.1以上でないと使えません。



脳力発揮

これらのPHP関数がスクリプト中で何をしているのかわかりますか？

`mysqli_connect()`
`mysqli_query()`
`mysqli_close()`

[†] 訳注：付録iiに従って、VertrigoServをインストールしたのであれば、左から順に'localhost'(ホスト名)、'root'(データベースユーザ名)、'vertrigo'(データベースパスワード)となります。XAMPPをインストールしたのであれば、左から順に'localhost'(ホスト名)、'root'(データベースユーザ名)、"(データベースパスワード：なし)"となります。最後のデータベース名はご自身で決めたのであれば、その名前です。本書に従ったのであれば'aliendatabase'です。

データベースとお話しする PHP の関数を使う

`mysqli_connect()`、`mysqli_query()`、`mysqli_close()` の 3 つが MySQL データベースと通信する PHP の主な関数です。名前がワンパターンなのは偶然ではありません。MySQL とやり取りをする近代的な PHP 関数はすべて `mysqli_` で始まるのです。

MySQL とやり取りする古代の PHP 関数は「i」なしの「`mysql_`」で始まりました。`i` は improved (改良された) の頭文字で、今では `mysqli_` 関数が好まれています。

`mysqli_connect()`

すでに覚えたはずの 4 つの情報を使って MySQL データベースに接続します。

`mysqli_query()`

MySQL データベースに問い合わせを発行します。通常テーブルにデータを格納するかまたは取り出します。

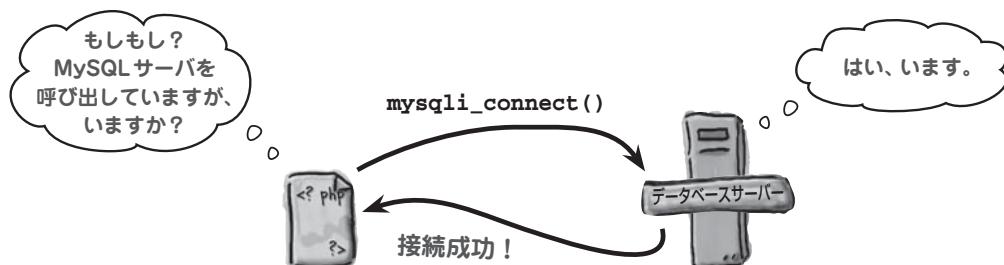
`mysqli_close()`

MySQL データベースの接続を閉じます。

これら 3 つの関数を使うには、およそ予想通りの典型的なステップを踏みます。

① `mysqli_connect()` を使ってデータベースに接続します。

サーバの場所、ユーザー名、パスワードを与えて MySQL データベースサーバとやり取りするための許可をもらいます。もちろんデータベース名も指定して下さい。特定のデータベースとの接続だからです。



② SQL で問い合わせを作って PHP 変数の文字列として格納します。

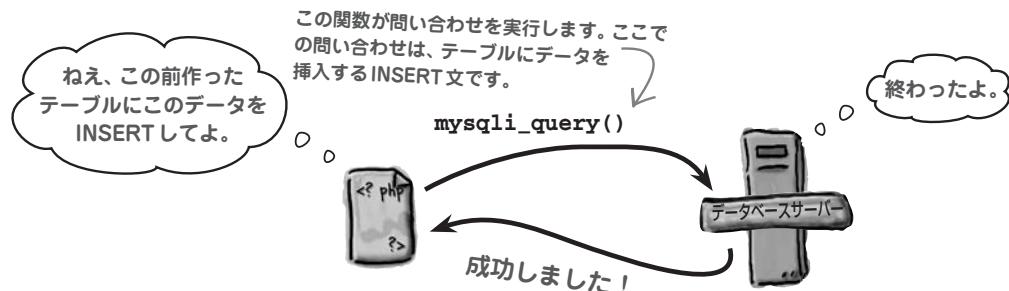
データベースサーバと通信するには、SQL コマンドを使う必要があります。例えば `alien_abduction` テーブルにデータを追加するには `INSERT` 文が必要です。どんな名前を付けても特に問題はありませんが、`$query` のような直感的にわかりやすい名前が良いでしょう。



問い合わせのメッセージを作って `$query` 変数に格納しました。

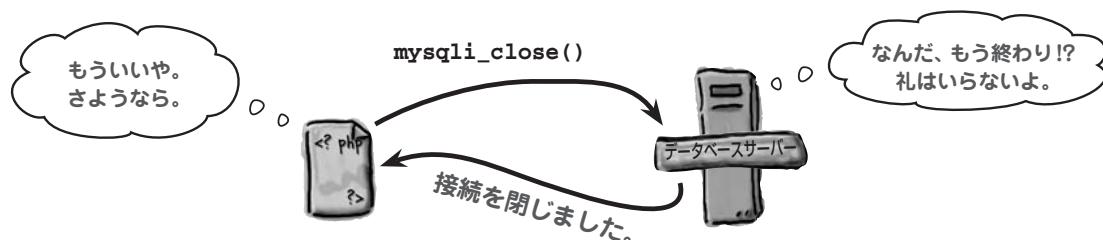
③ mysqli_query() 関数で問い合わせを発行する。

mysqli_query() 関数に \$query 変数を与えて MySQL データベースサーバに話しかけ、aliens_abduction テーブルにデータを追加します。mysqli_query() にはステップ 1 で取得した接続名とステップ 2 で作った問い合わせ内容を持つ変数名の両方を知らせる必要があります。



④ mysqli_close() 関数を使ってデータベース接続を閉じます。

最後に、mysqli_close() でデータベースサーバとの会話が終了したことを伝えます。



これが接続変数の名前です。

何か問題があったら、ここでメッセージが返されすべて止まります。

これがデータをデータベースに追加する SQL の問い合わせ INSERT です。

```
1 <?php  
$dbc = mysqli_connect ('data.aliensabductedme.com', 'owen', 'aliensrool', 'aliendatabase')  
or die(' エラー : MySQL サーバとの接続に失敗しました。 ');
```

```
2 $query = "INSERT INTO aliens_abduction (last_name, first_name, when_it_happened, how_long, "  
"how_many, alien_description, what_they_did, fang_spotted, other, email) ".  
"VALUES ('清水', '祐子', '13日前', '1日', '4人', '緑色で触覚が6つありました。', '.  
' ちょっと話して犬と遊びました。', 'はい', 'その犬を見たと思います。連絡を下さい。', '.  
'yuko@gregs-list.net');";
```

mysqli_query() は MySQL サーバと通信する PHP の手段です。変数 \$query に格納されているのは SQL コードであって PHP コードではありません。

```
3 $result = mysqli_query ($dbc, $query)  
or die(' エラー : データベースとの問い合わせに失敗しました。 ');
```

mysqli_query() は MySQL サーバと通信する PHP の手段です。変数 \$query に格納されているのは SQL コードであって PHP コードではありません。

```
4 mysqli_close ($dbc);  
?>
```

ここで接続を閉じます。

素朴な疑問に答えます

Q: 変数\$queryに格納する代わりに、すべてのSQLコードをmysqli_query()関数に直接書き込むことはできなかったのでしょうか？

A: できはしますが、コードが汚くなります。問い合わせをまず変数に格納し、そのあとその変数を使ってmysqli_query()関数を呼び出したほうが、ちょっとだけ自分のコードを管理しやすくなります。

Q: INSERTの問い合わせ文を発行したコードの結果を使って何かやるべきことがあるのでしょうか？

A: まあ、あるかもしれません。何か問題が起こった場合、今 のところはdie()を使ってスクリプトを終了し、ブラウザにメッセージを送っています。そのうち、問い合わせがうまく動かなかった場合、ユーザにより多くの情報を提示したいと思うようになるかもしれません。そのような場合問い合わせの結果もあわせて使い、問い合わせがうまくいくようにすることができるかもしれません。

Q: データベースへのINSERT(挿入)なのに、どうして問い合わせ(query)と言うのでしょうか？「問い合わせ」と言うからにはデータベースに何かを尋ねるのではないのでしょうか？

A: その通りです。「問い合わせ」というのは正に何かを尋ねる(お願いする)ことです。何かをするようにデータベースにお願いするのです。MySQLデータベースアプリケーションでは、「問い合わせ」という単語は非常に汎用的な意味を持っています。データベースに対して実行するすべてのSQLコマンドを、データの格納でも、取り出しでも、すべてを指します。

Q: INSERT文を始めから単に1つの長い文字列で作らないのは何故ですか？

A: INSERT文は、いくつかの短い文字列から作り上げられるとしても、最終的には1つの長い文字列に格納されていなければならないことは忘れないで下さい。理想的には、INSERT文は1つの文字列でコードになっていた方が良いのかもしれません。でも多くのSQL文もそうですが、INSERT文はものすごく長く、したがって「通常の」コード行には収まりきません。ですから問い合わせ文の文字列は短い文字列にしておいて、ドット演算で貼り付けていった方がコードが読みやすくなると思いませんか？

Q: INSERTを行うときカラム名のリストは本当に付けなければならぬものなのでしょうか？

A: 実はNOです。INSERT文にカラム名は無くても大丈夫です。ただしその場合、テーブルの全カラムの値を、テーブル構造に従った出現順に指定しなければなりません。このためカラム名と値だけで指定する方が一般に安全で便利です。

Q: データベースとの接続は自動的に閉じられると聞きました。mysqli_close()は呼び出さなくてもいいでしょうか？

A: 呼び出すべきです。データベースサーバが一度に提供できる接続数には限りがあるので、可能な限り残しておくべきです。接続を1つ閉じれば、それにより接続が1つ空きますので、新しい接続を1つ確立できます。データベースをみんなでシェアしているような場合、自分への接続数の割り当ては、例えば5つしかないかもしれません。そんなとき新しいデータベース駆動型アプリケーションを作る度に、今オーブンできる接続をありつけ自分で工面する必要に迫られることになるでしょう。

重要ポイント

- データベースとの接続には、場所とユーザ名とパスワードとデータベース名が必要です。
- mysqli_connect()関数はPHPスクリプトとMySQLデータベースサーバとの接続を確立します。
- die()関数はスクリプトを終了させ、接続に失敗した場合のフィードバックを返します。

- PHPコードからSQLの問い合わせを発行するには、問い合わせ文を文字列に組み上げ、その後その文字列をmysqli_query()に渡して呼び出しを実行します。
- 終わったらmysqli_close()関数を呼び出して、MySQLデータベースとの接続を閉じます。



試運転

オーウェンのreport.phpスクリプトのメールを送る部分のコードをMySQLデータベースにデータを挿入するコードに置き換えて、試してみましょう。

report.phpスクリプトの中でオーウェンにフォームのデータをメールで送る部分のコードを削除します。その場所に次のようなコードを入力します。まずMySQLデータベースと接続し、次にPHPの文字列としてSQLの問い合わせ文を構築し、その後データベースに問い合わせを発行し、最後に接続を閉じます。

これが今までかかって作ってきた
データベース操作用の新しいPHPコード
です。タグ<?php ?>はreport.phpには
入れてはいけません。なぜならこのコードは
すでにタグの中にあるスクリプトに
注入するからです。

```
<?php
$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool', 'aliendatabase')
or die('エラー: MySQL サーバとの接続に失敗しました。');
$query = "INSERT INTO aliens_abduction (last_name, first_name, when_it_happened, how_long, "
"how_many, alien_description, what_they_did, fang_spotted, other, email) "
"VALUES ('清水', '祐子', '3日前', '1日', '4人', '緑色で触覚が6つありました。', '",
"ちょっと話して犬と遊びました。', 'はい', 'その犬を見たと思います。連絡を下さい。', '",
'yuko@gregs-list.net')";
$result = mysqli_query($dbc, $query)
or die('エラー: データベースとの問い合わせに失敗しました。');
mysqli_close($dbc);
?>
```

新しいreport.phpをWebサーバにアップロードしたら、report.htmlページをブラウザで開き、誘拐のレポートフォームにアクセスします。フォームを埋めたら「誘拐レポートの送信」ボタンをクリックし、データをデータベースに格納します。次にMySQLツールに点火しSELECTで問い合わせを実行してデータベースに変化があったかを覗いてみます。

last_name	first_name	when_it_happened	how_long	how_many	alien_description	what_they_did
清水	祐子	3日前	1日	4人	緑色で触覚が6つありました。	ちょっと話して犬と遊びました。
清水	祐子	3日前	1日	4人	緑色で触覚が6つありました。	ちょっと話して犬と遊びました。

合っていますか？スクリプトが本当にやるべきことは何か、そして何故こうなったかを書き留めておきましょう。



ちょっと待ってよ。フォームから取ってきたデータをデータベースに格納するというのが大前提じゃないの？ これじゃフォームに何を入れても問い合わせ文が全く同じデータをデータベースに挿入しちゃうんじゃない。PHPスクリプトは何も自動化してないじゃない。

これは大問題です。INSERTによる問い合わせは、フォームのデータを挿入しなければいけないのであって、静的な文字列ではありません。

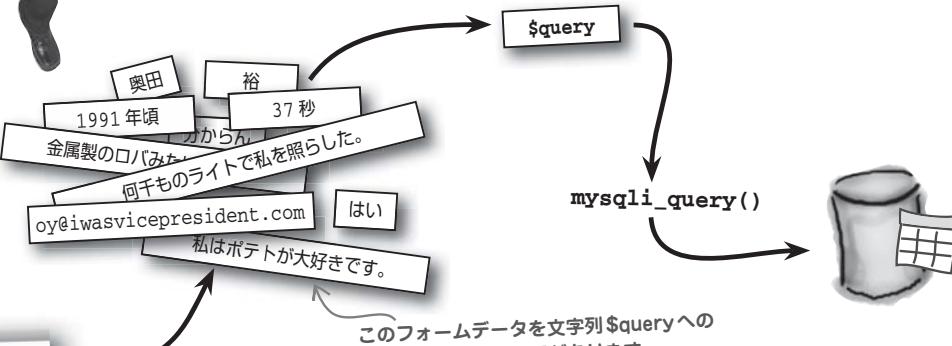
今までかかって作り上げた問い合わせ文は、ハードコーディングされた文字列でできています。本当はUFOによる誘拐のフォームに入力されたテキストデータを取ってこなければならなかったのです。スクリプトがフォームと協調して動くようにするには、フォームフィールドからデータを取ってきて問い合わせ用の文字列に食わせる必要があります。

UFOに誘拐された？！『誘拐レポート』を！

UFOに誘拐された時の様子を教えてください。

名：
住所：
UFOアドレスは？
いつですか？
何時頃ですか？
何人見ましたか？
どんな跡らしだしたか？
どうきましたか？
何かをありましたか？
何か犬が見ましたか？
いいえ

何か他にコメントは？



UFOによる誘拐の
フォームはユーザの
レポートデータによるもの

脳力発揮

オーウェンのフォームから INSERT の問い合わせ文に値をぶち込むにはどんな PHP コードが役立つでしょう？

\$_POSTがフォームのデータを供給します

良いニュースです。report.phpスクリプトは実はすでにフォームのデータを変数に格納済みなのです。\$_POSTスーパーグローバルのおかげです。以下のPHPコードを覚えていませんか？

```
$name = $_POST['lastname'] . ' ' . $_POST['firstname'];
$when_it_happened = $_POST['whenithappened'];
$how_long = $_POST['howlong'];
$how_many = $_POST['howmany'];
$alien_description = $_POST['aliendescription'];
$what_they_did = $_POST['whattheydid'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];
$other = $_POST['other'];
```

\$_POSTスーパーグローバルは、オーウェンの各フォームフィールドから値を取り出して、各変数に値を格納するのにすでに使っています。

\$_POSTに使う名前はHTMLのフォームフィールドの名前とマッチしていないなければならないということを覚えていますか？

つまりフォームのデータはすでに手元にあるのです。あとはこのデータをUFOによる誘拐のINSERT文の中に組み入れるだけです。でもその前にちょっと変更しなければならないことがあります。今度はもうフォームデータをメールで送信しないので、\$name変数はもう必要ないです。ただし、ユーザの氏名はデータベースに格納するので相変わらず必要です。でも今回は変数を分けておいた方が良いでしょう。

```
$last_name = $_POST['lastname'];
$first_name = $_POST['firstname'];
```

ユーザ名は今回は別々の変数に格納して、aliens_abductionテーブルの別々のカラムに挿入できるようにします。



オーウェンのINSERTの問い合わせ用文字列を生成するPHPコードを書いてみましょう。文字列は\$query変数に格納し、実際のフォームデータをaliens_abductionテーブルに格納することができるようにして下さい。

.....
.....
.....
.....



エクササイズ の答え

オーウェンの INSERT の問い合わせ用文字列を生成する PHP コードを書いてみましょう。文字列は \$query 変数に格納し、実際のフォームデータを aliens_abduction テーブルに格納することができるようにして下さい。

SQL 文に現れるカラム名は、先ほど
出てきたものと完全に同じです。

```
$query = "INSERT INTO aliens_abduction (last_name, first_name, when_it_happened, how_long, ".  
"how_many, alien_description, what_they_did, fang_spotted, other, email)".  
"VALUES ('$last_name', '$first_name', '$when_it_happened', '$how_long', '$how_many', ".  
" '$alien_description', '$what_they_did', '$fang_spotted', '$other', '$email')";
```

清水祐子さん説明に関する静的なデータ
ではなく、今度はユーザーがフォームに
入力したデータなら何でも挿入できます。

変数の順序は、カラム名の順序とマッチして
いなければなりません。そうでないとデータが
テーブルの正しいカラムに格納されません。

素朴な疑問に答えます

Q: \$_POST のデータを格納するためにたくさんの変数を作らなければならないものなのでしょうか? 直接 \$_POST のデータを参照して \$query の文字列の中に入れることはできないのでしょうか?

A: できます。\$_POST を問い合わせ文で直接使うことを拒む理由は何もありません。しかし、フォームのデータで何かをする前にデータを個別に隔離するというのはコーディング上良い習慣です。ある程度常識となっているのですが、フォームのデータをいくつかの段階ごとに処理してからデータベースに挿入するのが良いとされています。例えば、何か危険なフォームデータを入力するなどしてハッカーが問い合わせ文をハイジャックするというような賢い方法があるかもしれません。そのような攻撃を防ぐ方法については6章で学びます。話を単純にするため、本章ではフォームのデータに対する処理は何もしません。でもそれは、先に進まないということを意味するではありません。問い合わせ文にデータを貼り付ける前にフォームのデータに変数の値をあらかじめ格納しておくというような、よくある手については追って説明します。

Q: では、单一引用符と二重引用符についての問題はどうでしょう? 問い合わせ文全体を单一引用符で囲み、各

変数を二重引用符で囲ってもよいですか?

A: それは問題で、ダメです。問い合わせ文全体を单一引用符で囲み、変数を二重引用符で囲むことはできません。PHP は文字列が单一引用符の中か二重引用符の中かで異なる扱いをするためです。両者の違いは、单一引用符に含まれるテキストは正にそのまま処理されるのに対し、二重引用符の中ではテキストはいくつかの付加的な処理がなされることです。この処理の結果二重引用符の中の変数は値に変換されて、文字列の中で変数名のところが値に置き換えられます。これは非常に便利であり、このため二重引用符が SQL の問い合わせ用文字列を構築する際に一般に好んで用いられる理由です。

Q: 問い合わせ用の文字列を構築する際に、単に変数と SQL コードとを連結するというのはダメでしょうか?

A: できます。連結のルートをたどるのであれば、二重引用符の代わりに単一引用符だけで話は済みます。でも問い合わせ用の文字列は、読みにくく汚くなっていく傾向にあることは避けられません。何と言っても読みやすさが一番です。組み込みの変数は連結を使う代わりに二重引用符で囲む。これが問い合わせ用の文字列を読みやすくする最良の方法です。



今まで学んだすべてのことを使ってオーウェンのフォームを処理するPHPスクリプトを完成させ、UFOによる誘拐のデータをデータベースにちゃんと格納できるようにしましょう。`report.php`スクリプトの空欄を埋めてPHPコードを仕上げて下さい。

```
<?php  
.....  
  
$when_it_happened = $_POST['whenithappened'];  
$how_long = $_POST['howlong'];  
$how_many = $_POST['howmany'];  
$alien_description = $_POST['aliendescription'];  
$what_they_did = $_POST['whattheydid'];  
$fang_spotted = $_POST['fangspotted'];  
$email = $_POST['email'];  
$other = $_POST['other'];  
  
$dbc = .....  
.....  
  
$query = "INSERT INTO aliens_abduction (last_name, first_name, when_it_happened, how_long, ".  
    "how_many, alien_description, what_they_did, fang_spotted, other, email) ".  
    "VALUES ('$last_name', '$first_name', '$when_it_happened', '$how_long', '$how_many', ".  
    "'$alien_description', '$what_they_did', '$fang_spotted', '$other', '$email')";  
  
$result = .....  
.....  
  
echo '情報提供ありがとうございます。<br />';  
echo '誘拐されたのは' . $when_it_happened;  
echo 'で、時間は' . $how_long . 'です。<br />';  
echo '宇宙人の数は' . $how_many . 'です。<br />';  
echo '奴らは：' . $alien_description . '<br />';  
echo '宇宙人にされたこと：' . $what_they_did . '<br />';  
echo 'ファンダはいましたか？' . $fang_spotted . '<br />';  
echo 'コメント：' . $other . '<br />';  
echo 'メールアドレスは' . $email;  
?>
```



エクササイズ の答え

今まで学んだすべてのことを使ってオーウェンのフォームを処理するPHPスクリプトを完成させ、UFOによる誘拐のデータをデータベースにちゃんと格納できるようにしましょう。
report.phpスクリプトの空欄を埋めてPHPコードを仕上げて下さい。

```
<?php
$last_name = $_POST['lastname'];
$first_name = $_POST['firstname'];
$when_it_happened = $_POST['whenithappened'];
$how_long = $_POST['howlong'];
$how_many = $_POST['howmany'];
$alien_description = $_POST['aliendescription'];
$what_they_did = $_POST['whattheydid'];
$fang_spotted = $_POST['fangspotted'];
$email = $_POST['email'];
$other = $_POST['other'];

$dbc = mysqli_connect('data.aliensabductedme.com', 'owen', 'aliensrool', 'aliendatabase')
      or die('エラー：MySQL サーバとの接続に失敗しました。');

$query = "INSERT INTO aliens_abduction (last_name, first_name, when_it_happened, how_long, "
        . "how_many, alien_description, what_they_did, fang_spotted, other, email) "
        . "VALUES ('$last_name', '$first_name', '$when_it_happened', '$how_long', '$how_many', "
        . "'$alien_description', '$what_they_did', '$fang_spotted', '$other', '$email')";

$result = mysqli_query($dbc, $query)
      or die('エラー：データベースとの問い合わせに失敗しました。');
mysqli_close($dbc);

echo '情報提供ありがとうございます。<br />';
echo '誘拐されたのは' . $when_it_happened;
echo 'で、時間は' . $how_long . 'です。<br />';
echo '宇宙人の数は' . $how_many . 'です。<br />';
echo '奴らは：' . $alien_description . '<br />';
echo '宇宙人にされたこと：' . $what_they_did . '<br />';
echo 'ファングはいましたか？' . $fang_spotted . '<br />';
echo 'コメント：' . $other . '<br />';
echo 'メールアドレスは' . $email;
?>
```

氏名用の新しい変数はユーザーの「姓」と「名」の両方を、フォームに入力された通りに保持します。

まず適切な接続用の情報を与えてデータベースとの接続を確立しなければ、PHP から SQL の問い合わせを発行することはできません。

PHP 文字列として問い合わせ文を構築しますが、その際フォームフィールドからデータを抽出することを忘れないで下さい。

問い合わせをデータベースに對し実行します。これがデータを挿入します！

データベースとの接続を閉じます。

フォームの「提出」が成功したことの確認画面を表示します。前章のスクリプトでやったと同じです。



試運転

オーウェンのスクリプトを変更し、実際のフォームデータを使って INSERT を実行します。

report.php から変数 \$name を削除し、代わりに変数 \$last_name と \$first_name を追加します。次に変数 \$query を修正し INSERT 文で静的なテキストを使う代わりにフォームの変数を使います。新しいバージョンのスクリプトをアップロードして、試してみましょう。report.html ページのフォームを何度も「提出」してみましょう。毎回異なるデータを入れるよう注意して下さい。

次に MySQL ツールを使って SELECT で aliens_abduction テーブルの中身を取り出して見てみましょう。

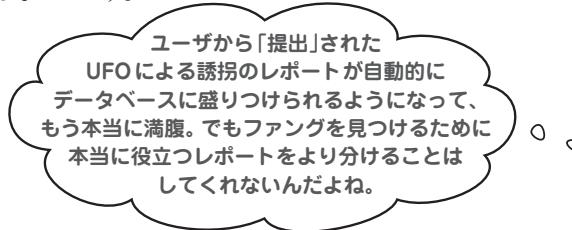
新しいUFOによる
誘拐のレポートが、
期待通り出て
きました！

last_name	first_name	when_it_happened	how_long	how_many	alien_description	what_they_did
清水	祐子	3日前	1日	4人	緑色で触覚が6つありました。	ちょっと話して犬と遊びました。
清水	祐子	3日前	1日	4人	緑色で触覚が6つありました。	ちょっと話して犬と遊びました。
奥田	裕	1991年頃	37秒	分からん	金属製の口バミみたいだった…	何千ものライトで私を照らした。
白田	敦朗	69年の夏	2時間	分かりません	空にとても明るい UFOがありました。	光線でガリソンスグンドへ導いてくれました。
石井	秀樹	去年の11月	11時間	たくさん	小さくて緑色です。	UFOのレギュレーションについて聞かれた。

清水祐子さんに関する
余計な行があります。
INSERT による問い合わせを直す前に投入したものです。でも心配にはおよびません。
次の章で不要なデータを削除する方法を学びます。

オーウェンはデータを選別する必要が出てきました

新しい改良版 report.php スクリプトはちゃんと仕事をしてくれていて、UFOによる誘拐のレポートをデータベースに追加するという処理を自動化してくれました。オーウェンは、ただふんぞり返ってレポートが転がり込んでくるのを待っていれば良いのです。しかし1つ新しい問題ができてしまいました。データが増えたということが、そのままで UFOによる誘拐のレポートの中から潜在的なファンダムの目撃情報を取り出しやすくなつたということにはならないのです。



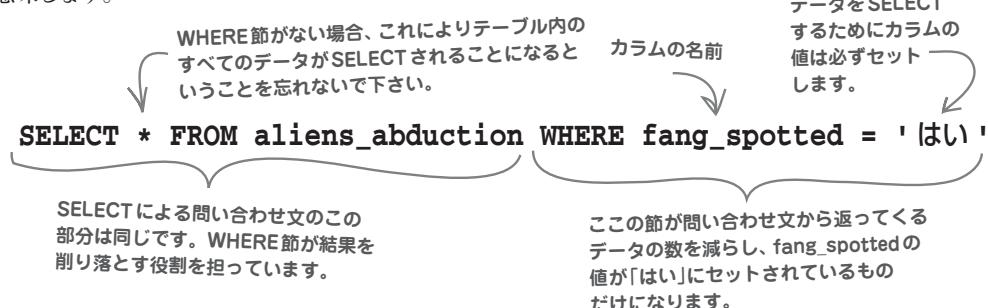
オーウェンはデータの中から、UFOによる誘拐の中でファンダムに関係する
ものだけというような特定のデータを見つける方法が必要です。

この質問に答えられる情報を持っているデータベースのカラムが何かご存じのは
ずです。そう fang_spotted です。このカラムの値は「はい」か「いいえ」ですが、
誘拐された人がファンダムを見たかどうかをレポートしています。ですから今必要な
のは aliens_abduction テーブルの中から fang_spotted カラムが値として
「はい」を持つものだけを選別する方法です。

以下のSQL問い合わせ文がテーブルからすべてのデータを引っ張ってくること
はすでに知っていますね。

```
SELECT * FROM aliens_abduction
```

SQLのSELECT文には、問い合わせにより返ってくる値を制御する節をくっつける
ことができます。これは WHERE と呼ばれ、問い合わせの結果をフィルターにかけて
正確に欲しいものだけをそこに書くことができます。オーウェンの場合、選ぶべき
(SELECTすべき)ものは、UFOによる誘拐(alien_abduction)のレポートなので
すが、この中(WHERE)で fang_spotted が「はい」と等しいものだけで良いとい
うことを意味します。





試運転

特定のデータを見つけるための WHERE 節付き SELECT による問い合わせを試してみる。

MySQL ツールで WHERE 節付きの SELECT による問い合わせを使い、ファンの目撃情報を含む特定の UFO による誘拐データを探してみます。

last_name	first_name	when_it_happened	how_long	how_many
奥田	裕	1991年頃	37秒	分からん
清水	祐子	3日前	1日	4人
清水	祐子	3日前	1日	4人
白田	敦朗	69年の夏	2時間	分かりません
斎藤	友世	つい今しがた	45分…で 継続中	数百人

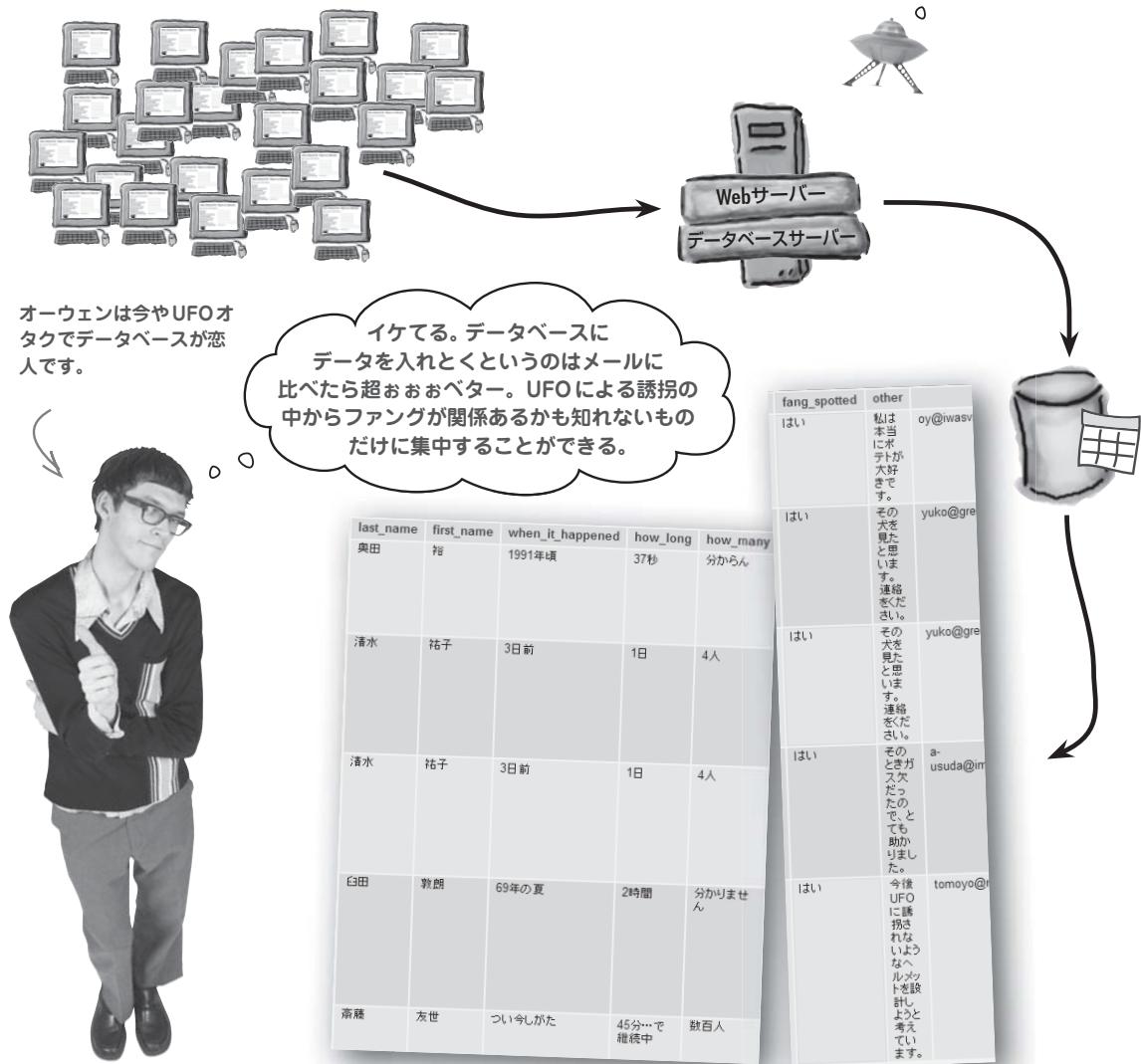
これらすべてのレコードは、
fang_spotted カラムの値が
「はい」にセットされています。



fang_spotted	other	
はい	私は本当にボテが大好きです。	oy@iwasv
はい	その犬を見たと思います。連絡をください。	yuko@gre
はい	その犬を見たと思います。連絡をください。	yuko@gre
はい	そのときが欠だつたので、とても助かりました。	a-usuda@im
はい	今後UFOに誘拐されないようなヘルメットを設計しようと考へています。	tomoyo@r

オーウェンのファンク探しはやっと軌道に乗りました

PHPとMySQLとインターフェースをとるPHP関数のおかげで、オーウェンのMySQLデータベースサーバは、UFOによる誘拐に関するデータをHTMLフォームから受け取り、それをデータベーステーブルに格納しています。データは安全にそのテーブルに格納されていて、折に触れてオーウェンがそれを選別します。オーウェンは準備ができたら、簡単なSELECTによる問い合わせ文を叩いて、潜在的にファングに関係のありそうな誘拐のレポートを抜き出せば良いのです。





素朴な疑問に答えます

Q: MySQLのテーブルにデータを挿入するやり方がわかったというのはハンパなく良いと思います。でもまだよくわかっていないのですが、テーブルとデータベースはどのように作られるのでしょうか？

A: 良い質問です。自分自身のテーブルがどのように作られるのかを理解する必要があります。単に提示されたコードを利用するだけではダメです。今までCREATE TABLE構文をあまり理解せずにテーブルを作っていました。オーヴェンの場合、单一のテーブルだったのでもよかったです、複数のテーブルを自分で設計して作るとなると、それでは不十分です。新しいテーブルに格納するデータについて詳細に検討し、それを表現するための最良の方法を考える必要があります。これこそが次の章の主眼です。準備は良いですか？



PHP & MySQL 道具箱

2章では、よく使われているデータベース MySQLについてと、データベースに接続する方法について学びました。

MySQL

データをデータベースおよびテーブルに格納し、SQL言語を用いて情報の追加および検索をするためのアプリケーション。

SQL

MySQLのようなデータベースアプリケーションと対話するための問い合わせ言語。

mysqli_connect()

この関数で MySQL サーバとの接続を閉じます。

mysqli_connect()

この関数で PHP スクリプトと MySQL サーバとの間の接続を開き、通信を開始します。接続には4つのデータが必要です。データベースのホスト名、データベースユーザ名、データベースパスワード、データベース名です。

mysqli_query()

この関数で MySQL サーバに問い合わせ文を送信します。

mysqli_select_db()

この関数で接続済みのデータベースに対して、使用するデータベースを後から伝えることができます。

3章 データベースを作ってそれを使う

自分のデータを作る



必要なデータが常にあるとは限りません。

使う前にデータを作らなければならないことがあります。そんなデータを入れておくテーブルを作らなければならないこともあります。使う前に作らなければならないデータを入れておくデータベースを作らなければならないこともあります。混乱してませんか？ご心配なく。自分自身のデータベースとテーブルの作り方を学ぶ準備が整いました。もしこれで十分でなくても、自分自身の最初のPHP & MySQL アプリケーションをおいおい構築していくことになります。

エルビスストアーを本格的に開店します

清野和司君はエルビスストアー MakeMeElvis.com を開店しました。需要は莫大です。既に多くの販売実績もあります。鉢付きジャンプスーツ(つなぎ:ポリエステル製)、付けモミアゲ、サングラスなどをたくさん売ってきました。

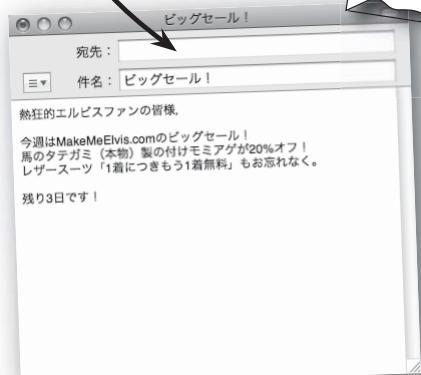
誰かが何かを買ってくれる度に、清野君には新しくメールアドレスが手に入ります。このアドレスを使って、清野君は店のセールに関するニュースを送っています。現時点では清野君は、メールアドレスのリストからそれぞれのアドレスを、セールの広告文面のメールにコピペして送るという操作を手作業でやっています。これはちゃんと機能していますが、非常に多くの時間と労力が必要です。

清野君は誰もが認めるオンラインエルビスグッズのキングです。



これじゃ時間がかかり過ぎる。メールを手作業で送るためにではなく、エルビスのコスプレのために時間をつぎ込みたいもんだ。

清野君はこのようなメールを書いて、「宛先」フィールドにメールアドレスをコピペしています。



清野君は使っているメールの「宛先」フィールドにアドレスをコピペするのにあまりにも莫大な時間を浪費しています。新しいアドレスを追加し、たくさんのメールを送るという作業を何とかして単純化できないかと思っています。

清野君の顧客メーリングリスト：

片岡 寿昭 nori_kataoka@breakneckpizza.com
毛利 みゆき miyuki@simuduck.com
神武 克彦 kolake2luv@breakneckpizza.com
田地 裕美 taji99@b0f0msup.com
金子 政次 masakaneko@breakneckpizza.com
松田 明子 matsuda@boardsrus.com
田中 芳津 tanaka@breakneckpizza.com
渋沢 利恵子 RieShibu@leapinlimos.com
安田 磨理 mari@objectville.net
福本 理絵 palofmine@mightygumball.net
沢 哲也 sawatetsu@breakneckpizza.com
福田 小百合 bukko@starbuzzcoffee.com
渋谷 洋子 shibuyoyo@chocololiccinc.com
赤井 雅子 am@honeydolt.com
岩岡 直樹 in86@objectville.net
渡辺 美恵子 watanabe@breakneckpizza.com
岩岡 直樹 katotakeshi@starbuzzcoffee.com
加藤 雄久 inue@starbuzzcoffee.com
井上 美佐 tomakita@objectville.net
北畠 智博 kurijun@mightygumball.net
栗林淳 hasemichi@tikibeanlounge.com
長谷 美智 chikurin@objectville.net
竹林 伸夫 fukuda@b0f0msup.com
福田 剛志 fujiki@starbuzzcoffee.com
藤木 達弘 matsuda@breakneckpizza.com
松田 智樹 kawamura@b0f0msup.com
川村 公俊 yame@weatherorama.com
中山 明子 nishioka@starbuzzcoffee.com
西岡 敏郎 toku@tikibeanlounge.com
森口 広美 nun23@objectville.net
沼田 治子 sugiyama@weatherorama.com
杉山 洋 ishi@breakneckpizza.com
石井 昌子 sasaki@chocololiccinc.com
佐々木 浩幸 kankebo@breakneckpizza.com
菅家 香織 maru@tikibeanlounge.com
野尻 朋子 ito@objectville.net
伊藤 拓 ozawa99@objectville.net

清野君は現時点で328のメールアドレスを集めていますが、毎日増えています。



この人たちは清野君のメールリストに載っていて、清野君のグッズでもっとエルビスっぽくなりたいともっと楽しんでいます。



清野君にはアプリケーションが必要です

アプリケーションというのはユーザの特別な目的を達成するために設計されたソフトウェアプログラムのことです。清野君が必要なアプリケーションは、フォームのボタンを1クリックするだけでメールアドレスのリストを管理し、リスト上の人達全員にメールを送ることができるというものです。清野君は次のような機能が欲しいと思っています。

- Webページ上でメールメッセージを入力する。
- ページ上で「送信」ボタンをクリックすると、メッセージが MakeMeElvis.com のメールリストの全員に対して送られる。
- メールリストは新しい顧客がWebフォームに登録することで自動的に更新されていく。 ←

アプリケーションの要求仕様に関するこの「洗濯物」リストに従えば、清野君のアプリケーションを視覚化してピカピカにすることができます。

Webアプリケーションというのはユーザの特定の目的を満たすために設計された動的なWebサイトのことです。

このメール処理部はオーウェンのUFOによる誘拐のアプリケーションよく似ているようです。でもここでの違いは、清野君のメールリストは自動的に構築されることと、メールメッセージがリストの全員に対して飛ぶことです。清野君のアプリはすべてについて全自動なのです！



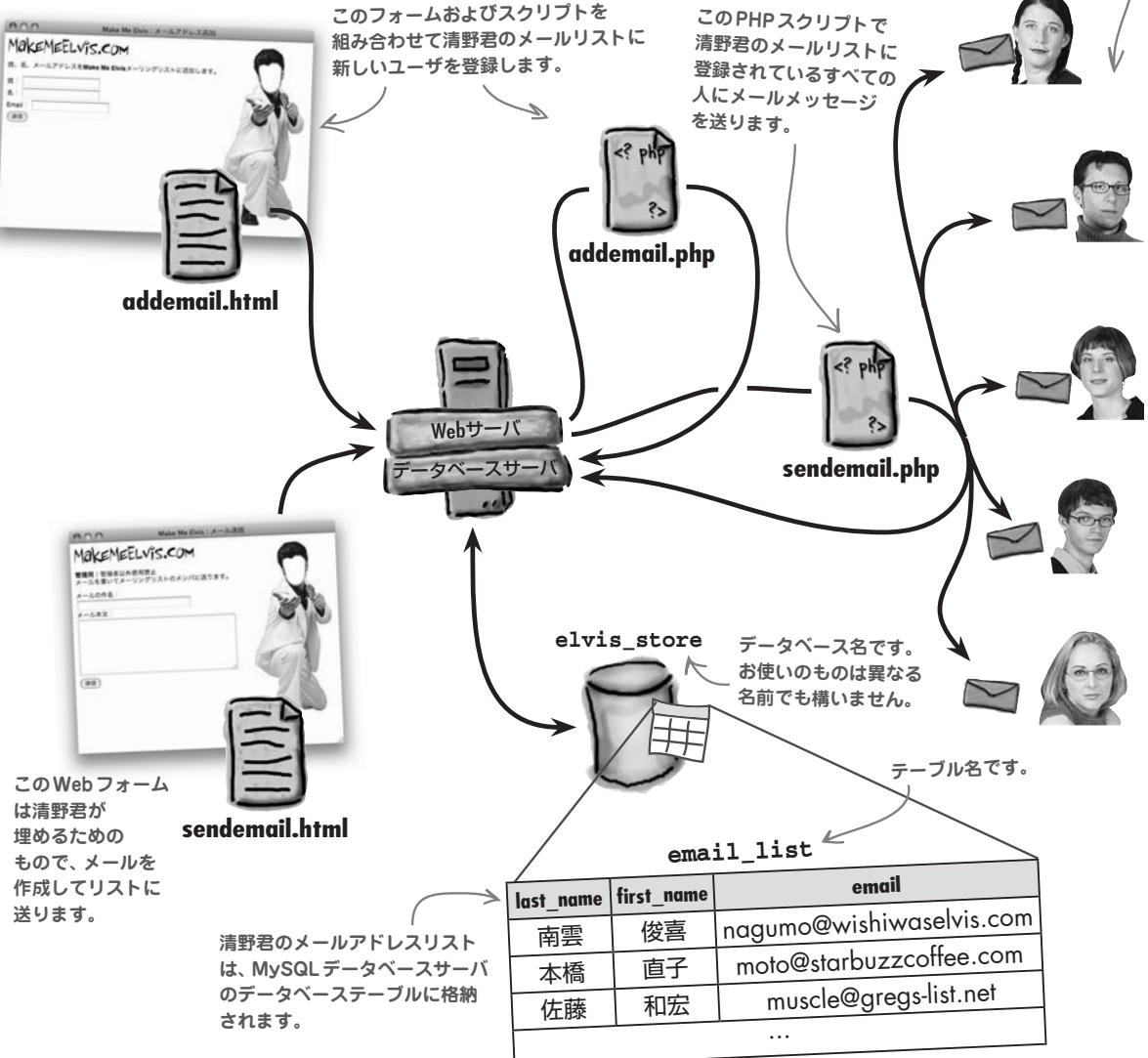
脳力発揮

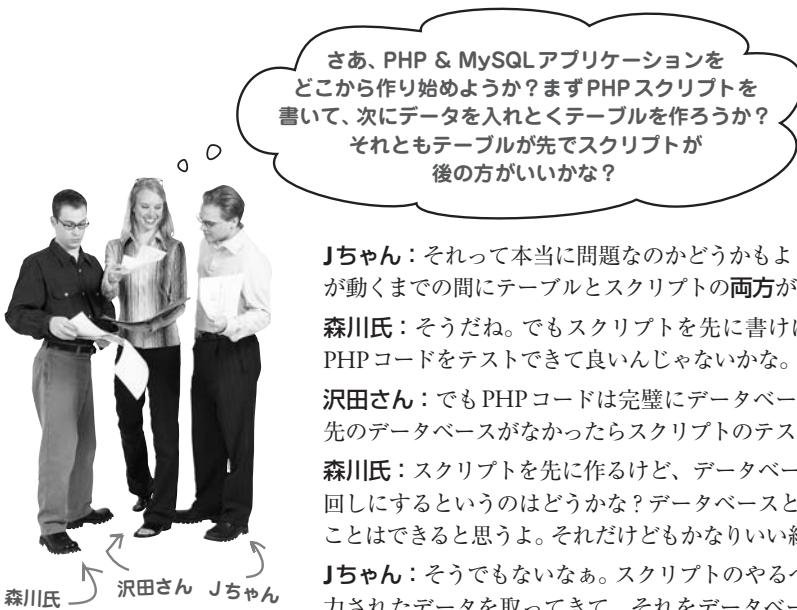
MakeMeElvis.com Webアプリケーションは2つの主要構成要素から成ります。1つは清野君のメールリストにある人達にメールメッセージを送るためのフォームであり、もう1つは、新しい顧客がメールリストに入会するためのフォームです。この2つのフォームを心に留めて、清野君のアプリケーションの概要を設計してみましょう。

清野君のアプリケーション設計を視覚化する

詳細な開発に飛び込む前にアプリケーションの設計を視覚化するというのは常に役に立つ作業です。この作業で、どのようなWebページやスクリプトが必要か、これらがどのように連携しあうか、そして恐らく最も重要なことです、データをどのようにMySQLデータベースに格納するかを明らかにすることができます。

この人達は清野君の
メールリストに登録されて
いて、清野君がリストに
対して送ったメールを
受け取ります。





Jちゃん：それって本当に問題なのかどうかもよくわかんないな。アプリケーションが動くまでの間にテーブルとスクリプトの両方が要ると思うよ。

森川氏：そうだね。でもスクリプトを先に書けば、データベースに接続する前に PHP コードをテストできて良いんじゃないかな。

沢田さん：でも PHP コードは完璧にデータベースに依存してるでしょ。接続する先のデータベースがなかったらスクリプトのテストなんてできっこないんじゃない。

森川氏：スクリプトを先に作るけど、データベースに接続する特定のコードだけ後回しにするというのはどうかな？データベースとのやり取り以外ほとんどすべてのことはできると思うよ。それだけでもかなりいい線でしょ？

Jちゃん：そうでもないなあ。スクリプトのやるべき仕事は HTML のフォームに入力されたデータを取ってきて、それをデータベースにぶち込むことだけでしょ。あと、メールをメーリングリストに送る方だって、スクリプトはデータベースから読み込んで、各ユーザへのメールメッセージを作り出すんだ。どっちにしろスクリプトにとってデータベースは超重要じゃん。

沢田さん：その通りだわ。でも今はまだ HTML フォームすら考えられないのよ。どんなものかもわからないのにぴったりの入れ物なんてどこにあるのかしら？スクリプトを書くことを考えることもできていないのに、データベースを作る必要があるということよ。

森川氏：わかった！まず HTML フォームを作る。それからどんなデータをデータベースに入れれば良いかを見極める。それでそれができたら、スクリプトで全部くっつけるんだよ。

Jちゃん：それで筋が通っているのかよくわかんないや。どうやって HTML フォームを作るの？その時にはすでにユーザからどんなデータをもらいたいかということを 100% の確度でわかっていることになるんだよ。

沢田さん：Jちゃんが正しいわ。HTML フォームを作るにしてもアプリケーションが欲しがっているデータが先に必要という方に話が戻っちゃう。データがすべてを動かしているのよ。だからたぶん先にデータベースとテーブルを作るべきなんじゃないかしら。その後 HTML フォーム。その次にフォームの「提出」に反応するスクリプト。

森川氏：納得。それで行こう！

Jちゃん：まだちょっと考えてるんだけど、このアプリケーションがどういう風に協調しようとしているのか「お決まりのステップ」を使って答えを探さなければならなくなると思うんだ…

MakeMeElvis.com の設計から実装に移る上で、
どのような「お決まりのステップ」があつてしかるべきか書き留めておきましょう。

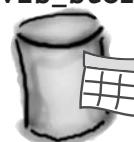
まず計画を！

清野君のアプリケーションを作り上げるというアタックには計画が是非でも必要です。やるべきことを個別のステップに分解することができれば、一時期に1つのことに集中できるので、他のことに惑わされることもないでしょう。

① メールリスト用のデータベースとテーブルを作る。

このテーブルは清野君のメーリングリストに載っている全員の「姓」と「名」とメールアドレスを格納します。

elvis_store



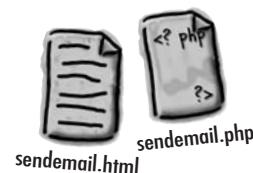
② 新しい顧客をリストに追加するためにメール追加用WebフォームとPHPスクリプトを作る。

ここでフォームとスクリプトを構築しますが、これを使って顧客は「姓」と「名」とメールアドレスを簡単に入力することができ、この情報をメールリストに追加することができます。



③ メール送信用のWebフォームとPHPスクリプトを作り、リストの全アドレスに対してメールを送ることができるようにする。

最後にWebフォームを作って清野君がメールメッセージを書き上げることができるようにします。もっと大事なことは、スクリプトを作って、書き上げたメッセージをメールリストテーブルに格納されている各々の人に送る能够することです。



何事もテーブルから始まる

実際に、何事もデータベースから始まります。それは基本的にデータを格納する入れ物だからです。前章でデータベースは内部的にはテーブルと呼ばれるより小さな入れ物に分割されているということを学習しました。覚えていますね。

カレンダーにおける曜日とか週といった具合に、テーブルはデータのカラムと行からできています。カラムはある特定の型のデータで構成されています。例えば、「姓」とか「名」とかemailなどです。行はカラムの集まりで、1行というのは各カラムから構成されています。行の一例としては「本橋、直子、moto@starbuzzcoffee.com」といったものが挙げられます。

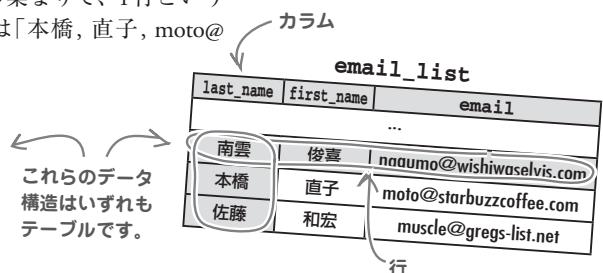
カレンダー

日	月	火	水	木	金	土
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21

...

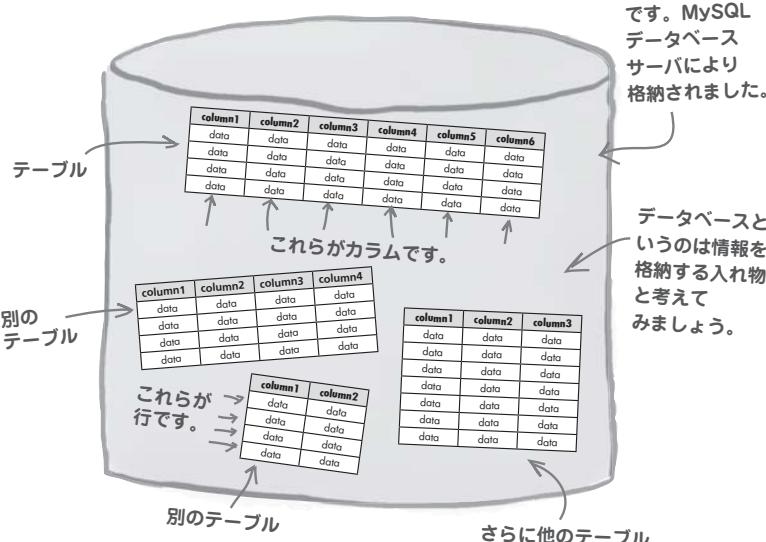
一般にデータベース内のすべてのテーブルはお互いに何らかの関係を持っています。もっとも関連性は非常にゆるい場合もあります。Webアプリケーションでは複数のテーブルがデータを通じてお互いに関係しあって構成されているということはよくあることです。しかし、それでもなおテーブルはカラムと行とからできています。

データベースとは、
データを格納するための
高度に構造化された
入れ物です。



テーブルにはカラムと
行の格子状パターンで
データを格納します。

素朴な疑問 に答えます



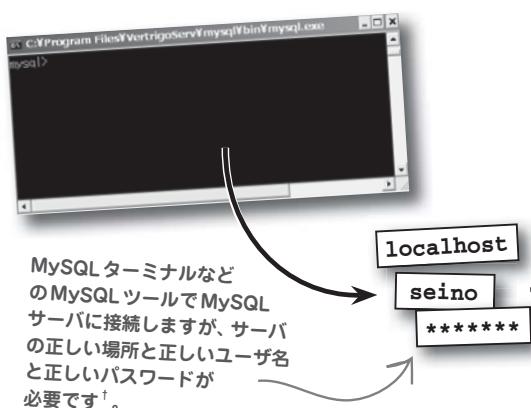
Q: データベースのデータは実際にどこに格納されるのですか？そのファイルを見ることができますか？

A: データベースは普通ハードディスク上のファイルに格納されます。もちろん見ることができることはできますが、見ても大した情報は得られません。データベースのファイルはバイナリファイルなので単純に開いても見ることはできません。ですからSQLがあるのです。データベースの中をじっくり見せてくれますし、その中に格納されているデータと対話することもできます。

MySQL サーバにコントクトをとる

清野君のアプリケーション設計にはデータベースとテーブルが必要です。データベースを取り扱う日々の作業の大半を占めるのは、テーブルとのやり取りです。でもいきなり飛び込んでテーブルを作ることはできません。まずテーブルを保持するためのデータベースを作らなければなりません。

SQLのCREATE DATABASEコマンドはデータベースを作るのに使います。一度作れば、CREATE TABLEコマンドでテーブルを作るところに行くことができます。ただしこれらいずれのコマンドを使うよりも前に、まずMySQLデータベースサーバと接続しなければなりません。これは前章すでにやったことですが、いくつかの重要な情報が必要でした。



PHPスクリプトでデータベースに接続し、データベースを操作するのであれば、必ずデータベースサーバの場所とユーザ名とパスワードをカギとしてMySQLターミナルかphpMyAdminを使うことになります。これらのツールはデータベースアプリケーションを離陸させるのに非常に有用で、データベースとテーブルを最初に作る際に使えます。

清野君のアプリケーションでデータベースとテーブルを作るのは最初の一回だけなので、手動でSQLの問い合わせ文によりこれらを作っても損はありません。そこで、好きなSQLツールを起動したら、清野君のアプリケーションをやっつける第一歩の準備完了です。メールリスト用のデータベースとテーブルを作りましょう。

[†] 訳注：付録iiに従って、VertrigoServをインストールしたのであれば、それぞれ「localhost, root, vertrigo」です。XAMPPをインストールしたのであれば「localhost, root, "（パスワードなし）」です。

- ① メールリスト用のデータベースとテーブルを作る。
- ② メール追加用WebフォームとPHPスクリプトを作って新しい顧客をリストに追加できるようにする。
- ③ メール送信用WebフォームとPHPスクリプトを作成してリストの顧客全員にメールを送れるようにする。

清野君のメール用データベースを作る

清野君のメールリスト用にテーブルとデータベースを新しく作るには、まずははじめにelvis_storeデータベースを作る必要があります。そこにemail_listテーブルを入れるので。SQLコマンドを使って両方を作ることができます。データベースを作るためにはCREATE DATABASEを使いますが、これは前章すでにちょっとだけ使いました。もうちょっと近づいて、これがどのように機能するのか見てみましょう。

CREATE DATABASE

database_name

これから作る新しい
データベース名です。

CREATE DATABASEコマンドの後ろに新しいデータベース名を指定する必要があります。

CREATE DATABASE elvis_store

MySQLデータベースサーバ上でこの文を実行すると、データベースが作られます。

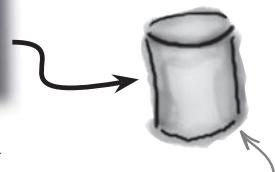


```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> CREATE DATABASE elvis_store;
Query OK, 1 row affected (0.00 sec)

mysql> .
```

ターミナルでSQLコマンドを実行するときは、必ず末尾にセミコロンを付与する必要があります。しかし、PHPのmysql_query()関数を使ってSQLの問い合わせ文を発行するときは違います。

elvis_store



データベースはできましたが、
テーブルがないのでまだデータ
を保持することができません。



要注意

SQL文をターミナル上で使うときは、必ずセミコロンで終わる。

PHPコードの中では、SQL文がセミコロンで終わる必要はありません。でもMySQLターミナルは違います。すべてのSQL文の末尾にセミコロンが必要ります。これはターミナルが複数のSQL文を動かす能力があるのに対し、PHPの場合は一度に1つの文を「提出」することしかできないためです。

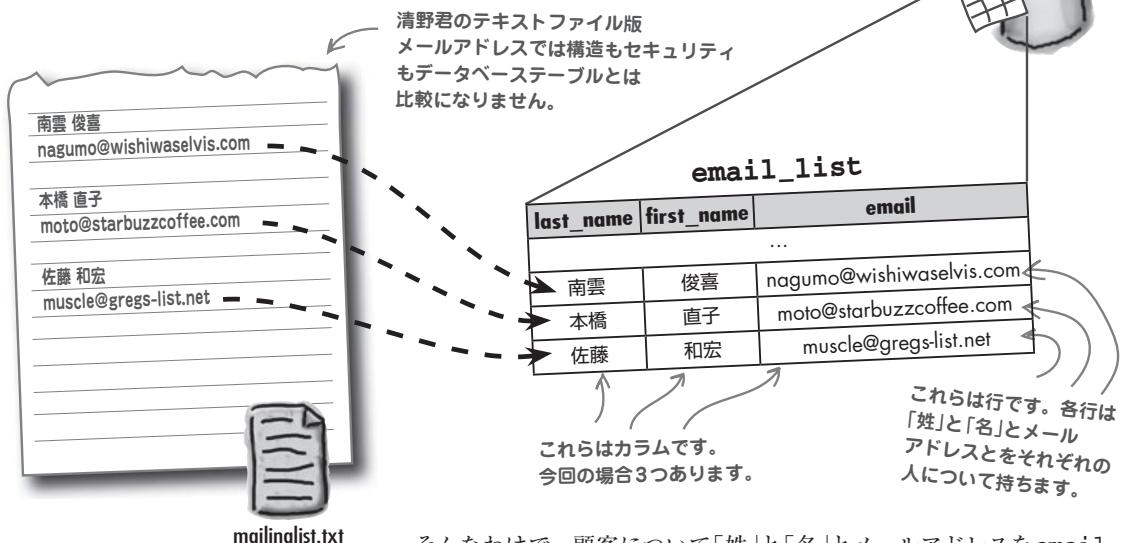
CREATE DATABASEは
新しいデータベースを
作るときに使う
SQLコマンドです。

さつきのデータベースの中にテーブルを作る

テーブルを作る前に、どのようなデータをテーブルに格納したいのかがわかつていなければなりません。清野君はメールリストの人達の「姓」と「名」を使って、メールメッセージをちょっとだけでもカスタマイズして送りたいと考えています。これらの情報にメールアドレスを加えると、清野君のemail_listテーブルには各エントリごとに3種類のデータを格納する必要があります。

テーブルのそれぞれのデータをカラムに入れるのですが、それぞれのカラムにデータを表す名前が必要です。last_name、first_nameそれとemailをカラム名にしましょう。テーブルの各行はそれぞれのカラムの値を持つ1つのデータから成り、清野君のメールリストの1つのエントリを構成します。

email_listテーブルは、elvis_storeデータベースに格納されうる多くのテーブルの中の1つです。



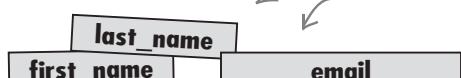
テーブルの行は

水平で、

テーブルのカラムは

垂直です。

そんなわけで、顧客について「姓」と「名」とメールアドレスをemail_listのカラムとして作らなければならないというところまでわかりました。ところが問題があって、MySQLのテーブルというのは、高度に構造化されているため、データのカラム名だけではなく、より多くの情報を指定しなければならないのです。データベースに対して指定すべきはどのような種類のデータをカラムに格納するつもりかということです。



データを定義する必要があります

テーブルを作るときは、MySQLサーバに対して各カラムがどんな種類のデータを保持するかを教えてあげる必要があります。データの型はすべてのMySQLカラムに必要となり、テーブルの各カラムを特定の型のデータのみを保持します。つまりカラムによってテキストを保持したり、数値を保持したり、時間や日付を保持したりするということです。MySQLデータベースには各種のデータ型があるため、ある特定のデータに対してどの型がフィットするかを知っておかなければなりません。とりあえず清野君はproductsという名前のテーブルを持っていることにしましょう。このテーブルは清野君の店で扱っているグッズを管理しています。

このカラムには清野君の店の各商品のテキストによる記述が含まれています。

在庫カラムinventoryには各商品が在庫としてどれだけあるかを示す整数値が含まれています。

products			
id	product	inventory	price
1	スエード靴(青)	24	5900
2	スパンコール付きポリエステル製パンツ	16	2350
3	粘着型付けモミアゲ	93	199
4	エルビス・カツラ	7	4800
...			

idカラムには、清野君の店の各商品を表すユニークなIDが含まれています。

price価格カラムには十進の値が含まれています。

product
スエード靴(青)
スパンコール付きポリエステル製パンツ
粘着型付けモミアゲ
エルビス・カツラ

整数値

inventory
24
16
93
7

整数値

テキスト

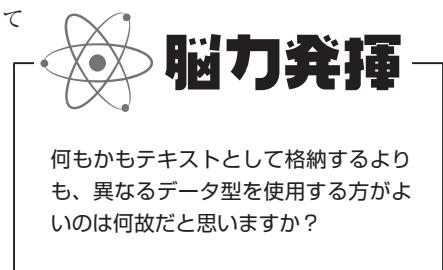
price
5900
2350
199
4800

十進数

productsテーブルのproductにはテキストしかありません。priceは十進数[†]だけですし、inventoryとidとは整数値だけです。MySQLではデータのこれらの型それぞれに特有の名前を付けています。実際には例えば日付と時間などのような型が他にもあります。

重要なのは、テーブルカラムを作るときには適切なデータ型を使用するということで、これによりテーブルを適正かつ効率的にすることができます。例えば、テキストデータを格納するには整数データよりも多くの領域を必要とします。ですから整数しか格納しないカラムに対しては、データ型として整数を使用するというのが賢いやり方です。またカラムがどのような型を保持するかがわかっていてれば、間違った型のデータを事故で挿入してしまうことをWebサーバは拒否することもできます。つまり日付を保持するようなカラムに対して、日付以外のデータをそのカラムに対して挿入しようとするとエラーとなります。

テーブルを作るには、各テーブルカラムに格納されるデータの型がわかつていなければなりません。



[†] 訳注：価格は十進数となっていますが、これは米ドルによる金額表記を例えば59.00のように、小数第2桁までで表現しているためです。本書では価格を円に書き換えたので、小数は必要なく、従って価格は十進数ではなく、整数値でも表現できます。

何もかもテキストとして格納するよりも、異なるデータ型を使用する方がよいのは何故だと思いますか？

MySQL のデータ型たちを一堂に会しましょう

MySQL のデータ型のうち特に便利なものいくつかを以下に挙げます。テーブルデータの特定のカラムに格納されるデータを指定するには、以下のどんな型を使ってもよいのです。覚えておいて下さい。データをぐちゃぐちゃにすることなく格納するというのが型の仕事です。



素朴な疑問に答えます

Q: VARCHARが同じことをもっと柔軟にやってくれるのにCHARをなぜ使うことがあるのでしょうか？

A: 答えは正確さと効率です。設計の観点から言うと、データをできる限り厳密にモデル化してテーブルを設計するべきです。もし、いさかの疑いの陰もなく、州のカラムが常にぴったり2文字の省略形表記[†]を保持するとわかっているのであれば、CHAR(2)を使って2文字の格納領域を認めてあげるのも意味のあることでしょう。一方、パスワードカラムが10文字までを許容するということであれば、VARCHAR(10)という方が理にかなっています。これは設計側の問題です。つまりCHARはVARCHARに比べると若干効率の面で上回ります。なぜならばCHARは長さが可変であることについて意識しなくてよいからです。従って、あるテキストカラムが固定の長さであると確信をもつてわかっている場合は、CHARが望ましいと言えます。

Q: どうしてINTやDECといった数値型が必要なのでしょう？

A: すべてデータベースの領域や効率をよくすることに帰着します。テーブル内の各カラムに対して、ベストマッチするデータ型を選ぶことで、テーブルの大きさを削減でき、結果としてデータ上の操作を速くすることができます。数字を実際の数(INTやDECなど)で格納すると、テキスト文字列として格納する場合に比べ、一般に効率的です。

Q: これで終わりですか？型はこれですか？

A: まだあります、これらが最もよく使われる型です。ひとまずはこれだけで走りましょう。絶対に使うことのないデータ型を見渡して泥沼にはまるよりはマシです。

† 訳注：アメリカの各州にはアルファベット2文字での省略形表記が割り当てられています。例えばNew York州はNY、Pennsylvania州はPAといった具合です。

目的は何？

左側の各MySQLデータ型とテーブルに格納するデータに関する右側の説明とをマッチさせましょう。

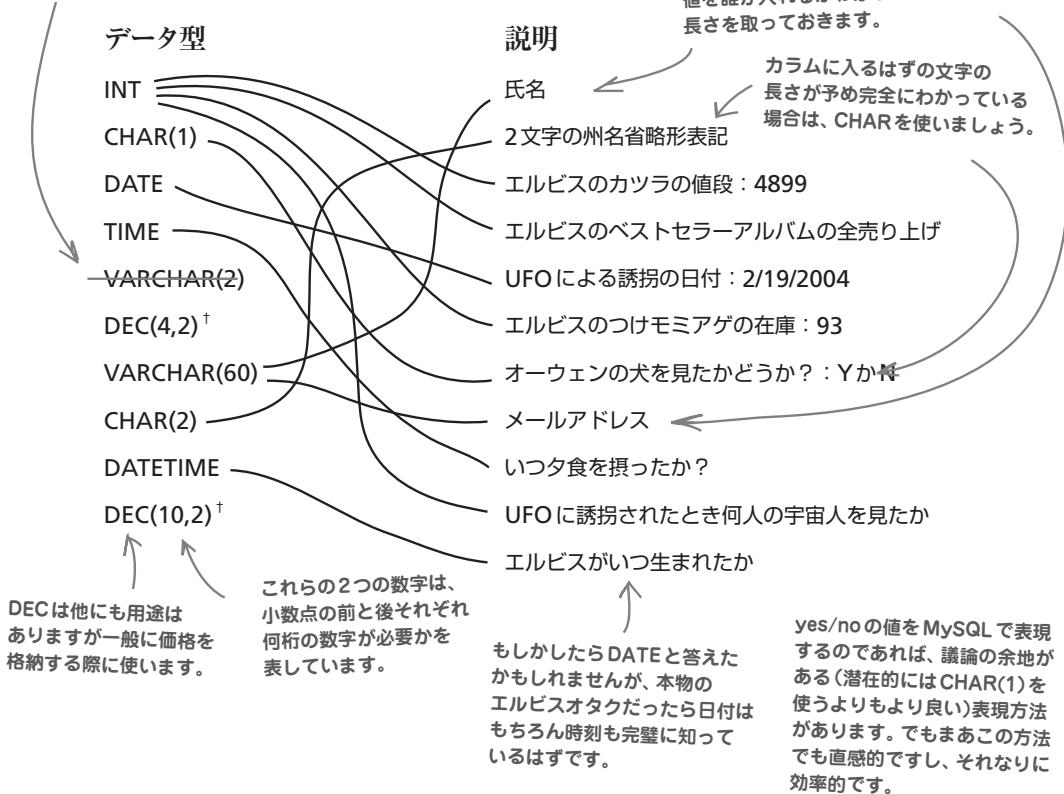
データ型	説明
INT	氏名
CHAR(1)	2文字の州名省略形表記
DATE	エルビスのカツラの値段：4899
TIME	エルビスのベストセラーアルバムの全売り上げ
VARCHAR(2)	UFOによる誘拐の日付：2/19/2004
DEC(4,2)	エルビスのつけモミアゲの在庫：93
VARCHAR(60)	オーウェンの犬を見たかどうか？：YかN
CHAR(2)	メールアドレス
DATETIME	いつ夕食を摂ったか？
DEC(10,2)	UFOに誘拐されたとき何人の宇宙人を見たか
	エルビスがいつ生まれたか

目的は何？ の 答え

左側の各MySQLデータ型とテーブルに格納するデータに関する右側の説明とをマッチさせましょう。

必要ありません。州名省略形表記で使えますが、多少効率的なのでCHAR(2)を選ぶべきです。

テキストの値が長さに関して変わるのであれば、VARCHARを選びましょう。どんな値を誰が入れるかわかりませんので、十分な長さを取っておきます。



[†] 訳注：原著では値段に対してこれらの型を割り当てていましたが、ドル表記から円表記に書き換えたため必要なくなりました。

問い合わせ文でテーブルを作る

テーブル作成に必要な道具はそろいました。テーブル名(`email_list`)も良い名前です。データカラムの名前もすでに決まりました。`last_name`、`first_name`、`email`です。残っているのは、各カラムのデータ型とこれら全部をくっつけてテーブルを作るSQL文です。テーブルを作るSQLコマンドは`CREATE TABLE`です。

目的のSQL文は`CREATE TABLE`で始まり、次にテーブル名が続きます。2つのカッコの中にカンマ区切りでカラム名のリストがあつて、それぞれのカラム名の後にそのデータ型を指定します。コマンドは以下のようになります。

当たりですが、まだここです…でも
次に移る準備はほとんどできました。

- ① メールリスト用のデータベースとテーブルを作る。
- ② メール追加用WebフォームとPHPスクリプトを作つて新しい顧客をリストに追加できるようにする。
- ③ メール送信用WebフォームとPHPスクリプトを作つてリストの顧客全員にメールを送れるようにする。

```
CREATE TABLE table_name
(
    column_name1 column_type1,
    column_name2 column_type2,
    ...
)
```

↑ テーブル名

↑ カラム名

↑ カラムのデータ型

↑ テーブル名やカラム名をつけるときに単語間を
分けるために下線を使わなければならないと
いうことはありませんが、名前付け規則に
一貫性を持たせるというのは良いことです。

必要に応じて
更にカラム名

SQLコマンドの
`CREATE TABLE`
はデータベースに新しい
テーブルを作るとときに
使います。

自分で考えてみよう

清野君の`email_list`テーブルを作るSQL問い合わせ文を書いてみましょう。
3つのデータのカラム`last_name`、`first_name`、`email`が必要です。

自分で考えてみよう の答え

清野君の email_list テーブルを作る SQL 問い合わせ文を書いてみましょう。
3つのデータのカラム last_name、first_name、email が必要です。

これがテーブルを作る SQL コマンドです。
大文字に注意して下さい。

開きカッコでこれから
作るカラムのリストの
開始を表します。

閉じカッコでカラムのリストが
終了したことを表します。

```
CREATE TABLE email_list
(
    last_name VARCHAR(20),
    first_name VARCHAR(20),
    email VARCHAR(60)
)
```

テーブル名はすべて小文字を
使いましょう。また単語の間は
ブランクではなく下線にします。

カンマはこれから作る
カラムの区切りです。

これによって MySQL は、email
カラムが VARCHAR データ型で
あることがわかります。(60) は、
テキストとして 60 文字分の長さ
まで保持できるということを
意味しています。

email アドレスを
格納するカラム名です。

使う(use)前にデータベースをUSEする

CREATE TABLE文をちゃんと動かすには、清野君はMySQLターミナルを使ってデータベースを選ぶ必要があります。そうすればMySQLは新しいテーブルがどのデータベースに所属するのかわかります。USEコマンドを使って、今開いているターミナルでのデフォルトのデータベースを設定することができます。この設定により以降のコマンドはそのデータベースに対するものとなります。以下のように動作します。

USE コマンドは MySQL に対してこれから使おうとしているデータベースを伝えます。

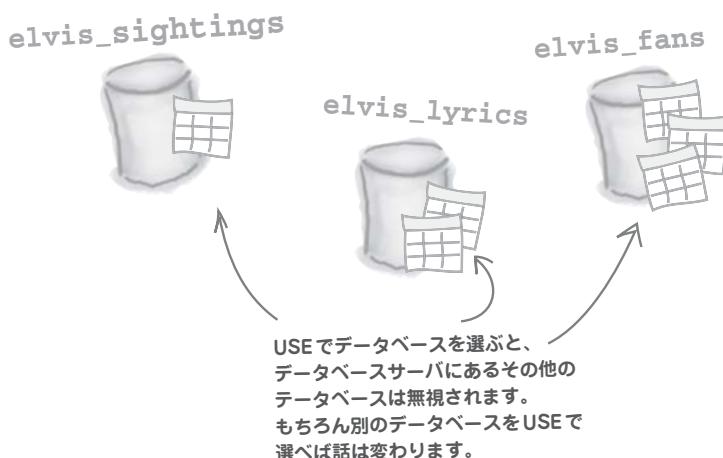
```
USE database_name
```

清野君の場合、データベース名(elvis_store)をUSE文で指定すれば、データベースを選ぶことができ、新しいテーブルにアクセスできます。

USEしようとしているデータベース名です。

```
USE elvis_store
```

USEコマンドは
それ以降のSQL文
に対するデフォルトの
データベースを
設定します。



USEコマンドでこれから一緒に作業をしたいと思っているデータベースを選びます。





試運転

まず清野君のデータベースをUSEし、次にテーブルを作る

問い合わせ文 USE を実行して、清野君のデータベース elvis_store を MySQL ツールで選び、その次に問い合わせ文 CREATE TABLE を実行して、そのデータベースの中にテーブル email_list を作ります。

```
USE elvis_store
```

```
CREATE TABLE email_list(last_name VARCHAR(20), first_name VARCHAR(20), email VARCHAR(60))
```

phpMyAdminのようなグラフィカルな SQL ツールを使っているのであれば、USE 文は必ずしも必要ありません。そのようなツールでは、SQL コマンドを発行する前にデータベースを選ぶようにうながしてきます。

テーブルを作成するコードはさっさと同じです。ちゃんと動かすためには事前にデータベースを選ぶ必要があつただけだったのです。

```
C:\Program Files\VertrigoServer\mysql\bin\mysql.exe
mysql> USE elvis_store;
Database changed
mysql> CREATE TABLE email_list
-> (
->     last_name VARCHAR(20),
->     first_name VARCHAR(20),
->     email VARCHAR(60)
-> );
Query OK, 0 rows affected (0.05 sec)

mysql>
```

USE コマンドのおかげでデータベースは選ばれており、テーブル生成は今度は何の問題もなく動きました。

素朴な疑問に答えます

Q: MySQL ターミナルを使っていると時々 -> という不可思議なプロンプトが出てくるのですが、これは何でしょうか？

A: プロンプトが -> と表示されるのは、複数行にまたがって 1 つの文を入力している場合です。MySQL は基本的に 1 つの文がまだ入力中であるということを知っているので、それを伝えたいのです。リターンキーで改行して複数行にまたがったとしても SQL 文が終了するわけではありません。文を終わらせるには、末尾にセミコロンをつける必要がありますが、そうすれば MySQL は文を実行します。

DESCRIBEでテーブル構造を明らかにする

テーブルの間違いを直すには、まず間違いを見つけなければなりません。たとえ間違いがないと確信しているとしても、自分のやったことをチェックするということが悪いはずはありません。SQLのDESCRIBEコマンドでテーブルの構造を解析し、カラム名やデータ型などの情報をリストにして表示することができます。

DESCRIBE table_name

清野君のテーブル名を組み入れると、次のようなSQL文となります。

これが説明して(DESCRIBEして)
欲しいテーブル名です。

DESCRIBE email_list

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> DESCRIBE email_list;
+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+
| last_name | varchar(20) | YES | | NULL | |
| first_name | varchar(20) | YES | | NULL | |
| email | varchar(60) | YES | | NULL | |
+-----+-----+-----+-----+
3 rows in set (0.03 sec)

mysql>
```

素朴な疑問に答えます

Q: Null, Key, Default, Extraといった残りのカラムはどんな意味ですか？

A: MySQLではテーブルの各カラムについてたくさんのオプション設定ができるようになっています。これらのオプションによりカラムに何も値が入れていない時に値をなしのままとするとか、デフォルトの値を設定しておくといったことができます。これらオプションについては、少し後でアプリケーションにとって、この手のことが問題となる場面で明らかにします。

Q: 実際にデータを格納したとき、このコマンドで表示することができますか？

A: 表示されません。DESCRIBEはテーブル構造だけを表示します。中に格納されているデータではありません。でも心配には及びません。テーブルに格納されているデータは、すぐ後で見ることができます。でもその前にデータをテーブルに実際に突っ込む方法を学ぶ必要があります。

Q: phpMyAdminを使っても全く同じテーブル構造を見ることがありますか？

A: できます。phpMyAdminのようなグラフィカルデータベースツールでも、DESCRIBE文を発行したり、またはテーブルを選び構造ボタンをクリックすることでテーブルの構造を見るできます。この手のことは、どのようなツールを使ってテーブルを解析しようとしているのかに完全に依存しています。

テーブルのカラム名や型は正しく表示されていますか？もし間違っていたらこれ以下の指示に従って下さい。間違っていないければ先へ進みましょう。

first_nameカラムを間違えて
first_naemと打ち込んだり
していませんか？

間違いに気付いて問い合わせ文の
CREATE TABLEをもう一度やって
みようとしたけど、ダメだったわ。たぶん
最初に間違ったテーブルを削除しないと
いけないのよ…ね？

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> DESCRIBE email_list;
+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+
| last_name | varchar(20) | YES |     | NULL    |       |
| first_name | varchar(20) | YES |     | NULL    |       |
| email | varchar(60) | YES |     | NULL    |       |
+-----+-----+-----+-----+
3 rows in set (0.03 sec)

mysql>
```



そうです。一度作ったテーブルを
もう一度CREATE TABLEで作り直すことはできません。

一度テーブルを作ってしまったら、それはもう存在してしまっていますから、新たな問い合わせ文CREATE TABLEで上書きすることはできません。もしテーブルをゼロから作り直したいのであれば、まずすでに存在するものを削除しなければなりません。その後で始めからやり直せばよいのです。

SQLでは、DROP TABLEコマンドでテーブルをデータベースから削除することができます。このコマンドはテーブルとそのテーブルに格納したすべてのデータを削除してしまいます。今、新しいテーブルには何のデータも入っていませんから、DROPで削除することにより失うものは何もありません。安心してDROPし、その後間違いを正しく直して新しいテーブルを作ればよいのです。

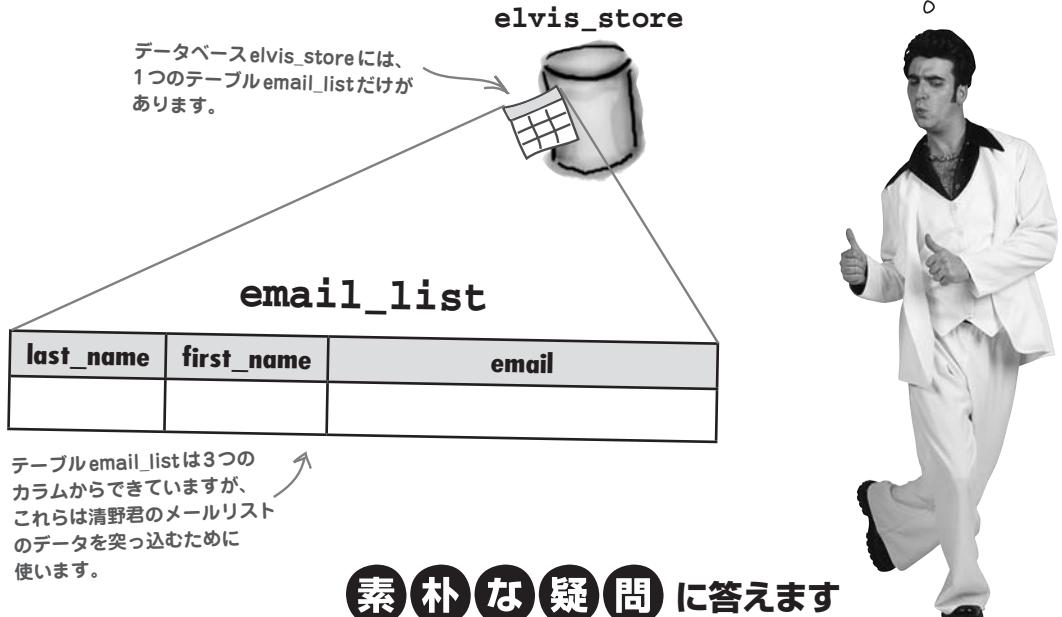
DROP TABLE email_list

データベースから削除したい
テーブル名です。

DROP TABLEコマンドは、データベース
からテーブルとその中の全データの両方を
削除します。

清野君はデータ格納準備完了です

SQLコマンドCREATE DATABASE、USEそれとCREATE TABLEの3つが全部成功して、清野君のメールリストデータベースとテーブルができました。でもあんまり嬉しくありません。テーブルは空です。上得意様のデータで埋まっているわけではないのです。ここはPHPの出番です…



素朴な疑問に答えます

Q: ねえ、Head First SQL(どうでもいいけど良い本だね)を持ってるんだけど、その本によると、SQL文のコード打ち込むときにはいつもセミコロンを後ろに置くことになってる。何でここは違うの？

A: Head First SQLを満喫して頂いているとのこと、感謝申し上げます。違いはMySQLと直接話しているかどうかによります。その場合セミコロンが必要で、これによりMySQLは文の終わりがどこなのかを判断します。従ってMySQLに対して複数の文を直接発行することもできるのです。PHPの場合は、関数mysqli_query()を使いますが、このときは一度に1つのSQL文を実行することしかできません。そのためセミコロンは必要ないのです。でもPHP文の後ろにはセミコロンが相変わらず必要であることは忘れてはいけません！

Q: じゃあデータがあるテーブルをDROPしたら、そのデータも全部消されちゃうの？

A: そうなってしまいます。テーブルをDROPするときは注意して下さい！

Q: じゃあじゃあデータのあるテーブルをデータを残して変えたくても、諦めるしかないの？

A: ちょっとちょっと、完璧な人なんてどこにもいません。違いは誰にでもあるものです。こんな時のためにSQLにはALTER文を用意しています。これは既存のテーブルを変更するために使います。このコマンドについては、本書のちよこっと後でお話しします。

メール追加用のスクリプトを作る

清野君はHTMLフォームを作って顧客から名前とメールアドレスを集め必要があります。これさえあれば、PHPスクリプトでデータを取ってきてemail_listテーブルに格納すれば良いだけです。Webフォーム(addeemail.html)には入力フィールドが3つとボタンが1つ要ります。フォームのactionは最も大事なコードです。なぜならその仕事は、フォームのデータを今から作ろうとしているaddemail.phpに渡すことだからです。

やっとここにきました。

- ① メールリスト用のデータベースとテーブルを作る。
- ② メール追加用WebフォームとPHPスクリプトを作成して新しい顧客をリストに追加できるようにする。
- ③ メール送信用WebフォームとPHPスクリプトを作成してリストの顧客全員にメールを送れるようにする。

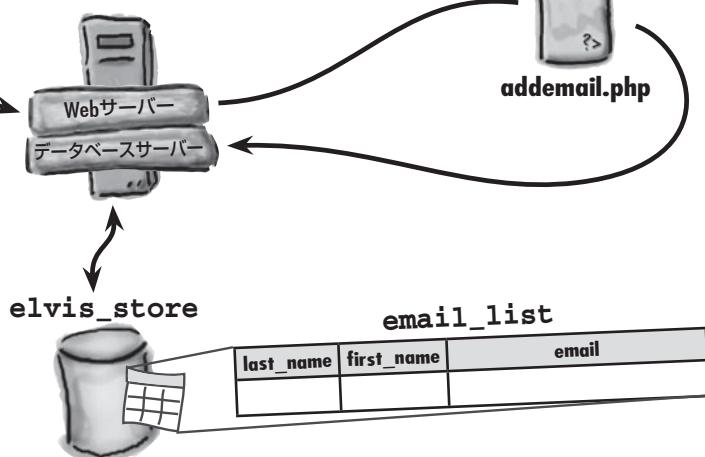
フォームのactionは、HTMLのWebフォームとPHPスクリプト(addemail.php)とをつなぐところです。データはPHPスクリプトで処理します。

```
...  
<form method="post" action="addemail.php">  
  <label for="lastname">姓:</label>  
  <input type="text" id="lastname" name="lastname" /><br />  
  <label for="firstname">名:</label>  
  <input type="text" id="firstname" name="firstname" /><br />  
  <label for="email">Email:</label>  
  <input type="text" id="email" name="email" /><br />  
  <input type="submit" name="submit" value="送信" />  
</form>  
</body>
```



addemail.php
スクリプトは、
フォームが「提出」
されると動き出しますが、その仕事は
フォームのデータを
処理して、その顧客を
メールリスト(データ
ベーステーブル)
に追加することです。

新しい顧客は清野君のメールリストに入りたかったら(清野君のデータベースに加えられることになりますが)単にWebフォームを使えば良いだけです。





`addemail.php`スクリプトは、メール追加用フォームからデータを取ってきて処理します。スクリプトはフォームからデータを持ってきて、elvis_storeデータベースに接続し、そして持ってきたデータをテーブル`eail_list`にINSERTします。まずは新しい顧客を挿入するSQL問い合わせ文サンプルを書いて清野君を助けてあげて下さい。その次はその問い合わせ文を使ってPHPスクリプトコードを完成させます。

データを清野君のテーブルに
突っ込む問い合わせ文の
一例をここに書きましょう。

```
<?php  
$dbc =  
.....  
.....  
$last_name = $_POST['lastname'];  
.....  
.....  
$query = .....  
.....  
.....  
mysql_query( ..... )  
.....  
echo '顧客を追加しました。';  
.....  
?>
```



addemail.php



addeemail.phpスクリプトは、メール追加用フォームからデータを取ってきて処理します。スクリプトはフォームからデータを持ってきて、elvis_storeデータベースに接続し、そして持ってきたデータをテーブルemail_listにINSERTします。まずは新しい顧客を挿入するSQL問い合わせ文サンプルを書いて清野君を助けてあげて下さい。その次はその問い合わせ文を使ってPHPスクリプトコードを完成させます。

```
INSERT INTO email_list (last_name, first_name, email)
```

```
VALUES ('国分', '真理', 'mari@breakneckpizza.com')
```

「提出」された情報を持っている
\$_POST配列の値はこのように
あります。

例で示したINSERT問い合わせ文は、
PHP文字列に書き換えられていて、
挿入用のフォームデータを使うように
なっています。

```
<?php
$dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')
or die('エラー：MySQL サーバとの接続に失敗しました。');

$last_name = $_POST['lastname'];
$first_name = $_POST['firstname'];
$email = $_POST['email'];

$query = "INSERT INTO email_list (last_name, first_name, email) :.
VALUES ('$last_name', '$first_name', '$email')";

mysqli_query( $dbc, $query )
or die('エラー：データベース問い合わせに失敗しました。');

echo '顧客を追加しました。';

mysqli_close($dbc);

?>
```

好みの問題ではありますが、ここに
HTMLの<a>タグを貼ってフォームに
戻るリンクを付けることもできます。



addeemail.php



試運転

メール追加用フォームを使ってみる

メール追加用フォームのWebページをサンプルコードのサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードします。WebページはCh03というフォルダにあります。コードの内訳は清野君のWebフォームがaddemail.htmlにあり、その他にスタイルシート(style.css)と2つのイメージ(elvislogo.gifとbkankface.jpg)があります。では新しくテキストファイルをaddemail.phpという名前で作り、反対側のページにあるすべてのコードを打ち込んで下さい。これが清野君のWebフォームを処理して、新しい顧客をemail_listテーブルに追加するスクリプトです。以上すべてのファイルをWebサーバにアップロードして、addemail.htmlページをWebブラウザで開きます。フォームに新しい顧客を入力したら、「送信」をクリックします。

データベース接続用の
変数を自分自身のもの
に合わせて変更すること
を忘れない。

新しい顧客をメールリストに挿入すると、addemail.phpスクリプトが確認画面を表示してくれます。

顧客がデータベースに追加されたかどうかをチェックするため、MySQLツールを使ってSELECTの問い合わせを発行してみます。

	← T →	last_name	first_name	email
	□	✎	×	
		国分	真理	mari@breakneckpizza.com

素朴な疑問に答えます

Q: SQLのSELECTコマンドの星はアスタリスク(*)と同じものですか?

A: そうです。キーボード上ではコロンの上[†]にある同じ文字です。**SHIFT** キーを押しながら **:** を打って下さい。ただし、アスタリスクと同一の文字ではありますが、SQLの世界では常に必ず星と呼びます。良いことではないでしょうか?なぜなら「SELECT アスタリスク FROM…」と読むより「SELECT 星 FROM」と読む方が簡単です[‡]。

Q: SQLには星の他にも特別な意味を持つ文字がありますか?

A: SQLには他にも特別な文字、あるいは予約文字があることはありますが、現時点で必要なものは星だけです。目の前の課題を解くために重要なものという意味では、SQL文のSELECT部分で使うのは星だけです。

[†] 訳注: JIS キーボードではこのような配列となっているので変更しました。原著はUS キーボード配置で8の上と説明されています。

[‡] 訳注: 原文はstarだったため星として原文通り訳しましたが、日本にはSQLの世界での*を特に何と呼ぶかについての掲はないと思います。星と呼んだりスターと呼んだりアスタリスクと呼んだり、あるいはアステリスクと呼んだりと人によってさまざまです。



自分で考えてみよう

清野君のメールリストを埋めていくに従い必要となるであろうSQL文を書いてみましょう。清野君が特定の顧客データを簡単に見つけられるようにします。

「名」が「彰」であるような顧客データすべてを抽出する。

「名」が「憲一」であるような顧客の「姓」を抽出する。

emailアドレスがsasaki@objectville.netであるような顧客の「姓」と「名」を抽出する。

「姓」が「川本」で「名」が「裕子」であるような顧客の全カラムを抽出する。

イケてるじゃん。これでお客様はWebページからメールリストに登録されていく。リストは勝手にどんどんできていく。

	last_name	first_name	email
<input type="checkbox"/>	原田	節	harada@leapinlimos.com
<input type="checkbox"/>	国分	真理	mari@breakneckpizza.com
<input type="checkbox"/>	清水	慶子	keiko@breakneckpizza.com
<input type="checkbox"/>	浅尾	高行	takayuki@boards-r-us.com
<input type="checkbox"/>	川本	裕子	kawamoto@breakneckpizza.com
<input type="checkbox"/>	東川	誠司	seiji@b0tt0msup.com
<input type="checkbox"/>	宮崎	章子	miyazaki@breakneckpizza.com
<input type="checkbox"/>	黒田	朗	kuroda@simuduck.com
<input type="checkbox"/>	久保	毅一郎	kubo@objectville.net
<input type="checkbox"/>	黒田	彰	akira@mightygumball.net
<input checked="" type="checkbox"/>	清野	正治	sugano@breakneckpizza.com
<input type="checkbox"/>	克史		kuronuma@starbuzzcoffee.com
<input type="checkbox"/>	昌美		sekai@chocoholic-inc.com
<input type="checkbox"/>	佳也		zenba@honey-doit.com
<input type="checkbox"/>	雅仁		kintaka@objectville.net
<input type="checkbox"/>	雅幸		sugimoto@breakneckpizza.com
<input type="checkbox"/>	石川	安則	ishikawa@starbuzzcoffee.com
<input type="checkbox"/>	川村	忠範	kawamura@starbuzzcoffee.com
<input type="checkbox"/>	佐々木	久郎	sasaki@objectville.net
<input type="checkbox"/>	星野	喜一	hosho@mightygumball.net

でもメールリストは自分でメールを送れません。

清野君には他にもまだ作らなければならないWebアプリケーション部品が残っています。その部分を使って、メールメッセージを入力し、そのメッセージをメールリストに載っている人全員に配信するのです。このためには、新たにHTMLフォームとPHPスクリプトを作つてこれらを実行に移す必要があります…

① メールリスト用のデータベースとテーブルを作る。

② メール追加用WebフォームとPHPスクリプトを作つて新しい顧客をリストに追加できるようにする。

③ メール送信用WebフォームとPHPスクリプトを作つてリストの顧客全員にメールを送れるようにする。

ステップ2が終わりました！

自分で考えてみよう の答え

清野君のメールリストを埋めていくに従い必要となるであろうSQL文を書いてみましょう。清野君が特定の顧客データを簡単に見つけられるようにします。

「名」が「彰」であるような顧客データすべてを抽出する。

```
SELECT * FROM email_list WHERE first_name = '彰'
```

星はテーブルの
全カラムを抽出します。

このWHERE節が問い合わせ
の結果を「名」が「彰」である
顧客だけにそぎ落とします。

「名」が「憲一」であるような顧客の「姓」を抽出する。

```
SELECT last_name FROM email_list WHERE first_name = '憲一'
```

問い合わせの結果からlast_name
カラムだけが返ってきます。

email アドレスがsasaki@objectville.netであるような顧客の「姓」と「名」を抽出する。

```
SELECT last_name, first_name FROM email_list WHERE email = 'sasaki@objectville.net'
```

複数カラムの結果データを指定するには、
カラム名をコンマで区切ります。

「姓」が「川本」で「名」が「裕子」であるような顧客の全カラムを抽出する。

```
SELECT * FROM email_list WHERE last_name = '川本' AND first_name = '裕子'
```

このWHERE節は独立した複数の情報から
できています。この場合は「姓」と「名」の
両方がマッチするものとなります。

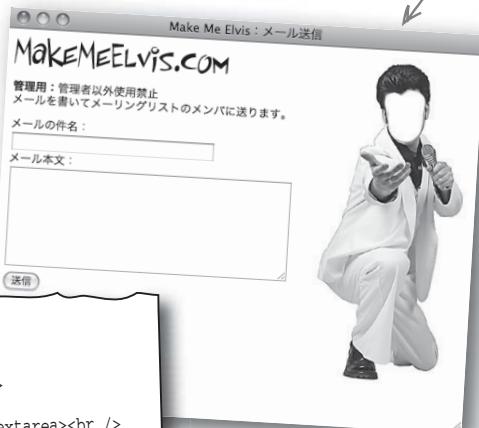
清野君のアプリケーションにはもう1つ必要です

メールメッセージを清野君のメールリストに入っている人たちに送るのは、その人たちをリストに追加したやり方とある意味似ています。なぜならHTMLのWebフォームとPHPスクリプトを使うからです。一番の違いは、メールメッセージをマーリングリストに送るには、email_listテーブルの中身を全部取り扱わなければならないということです。これに対してaddemail.phpスクリプトは一行のデータを取り扱つただけでした。

メール送信用のWebフォームで清野君はメールメッセージの件名と本文を入力し、最後にメールリストの全員にメールを送ります。

- ① メールリスト用のデータベースとテーブルを作る。
- ② メール追加用WebフォームとPHPスクリプトを作つて新しい顧客をリストに追加できるようにする。
- ③ メール送信用WebフォームとPHPスクリプトを作つてリストの顧客全員にメールを送れるようにする。

はあ、やつと
最後のステップ
に来ました。

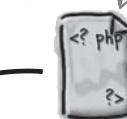


```
<form method="post" action="sendemail.php">
  <label for="subject">Subject of email:</label><br />
  <input type="text" id="subject" name="subject" size="60" /><br />
  <label for="elvismail">Body of email:</label><br />
  <textarea id="elvismail" name="elvismail" rows="8" cols="60"></textarea><br />
  <input type="submit" name="submit" value="Submit" />
</form>
</body>
</html>
```



sendemail.html

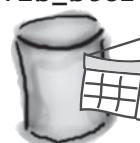
フォームのactionで
sendemail.phpスクリプト
をキックします。



sendmail.phpスクリプト
は、データベースのテーブル
から顧客を読み込み、清野君
が書いたメールメッセージを
それぞれの人に送ります。



elvis_store



last_name	first_name	email
国分	真理	mari@breackneckpizza.com
黒田	朗	kuroda@simuduck.com
宮崎	章子	miyazaki@breakneckpizza.com
...		

メール送信用スクリプトにおけるボルトとナット

sendmail.php スクリプトは、2つの別々の材料からデータを結合し、メールメッセージを生成して送らなければなりません。一方ではスクリプトは、メールの受信者の名前とメールアドレスとをelvis_store データベースの中のemail_list テーブルから引っ張ってくる必要があります。さらに清野君がメール送信用 Web フォーム (sendemail.html) に入力した件名とメッセージ本文を捕まえておかなければなりません。やるべきことをステップに分解してみましょう。

- \$_POST 配列を使って email の件名とメッセージ本文をフォームから取ってきます。

ここには何も目新しいことはありません。sendemail.html フォームで「提出」ボタンをクリックするとフォームのデータを sendemail.php にデータが送られます。このときデータは \$_POST 配列の助けを借りて変数に突っ込むことができます。

- email_list テーブルに対して問い合わせ文 SELECT を走らせます。

PHP の mysqli_query() 関数で SELECT の問い合わせを実行し、email リストからデータを取ってきます。今欲しいのはテーブルの全データなので、SELECT * を使えばよいのです。

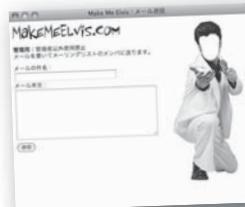
- 問い合わせの結果から email のデータを取ってきます。

問い合わせを実行しただけでは、データにアクセスすることはできません。

問い合わせ結果のデータから各行を捕らえて、各顧客の「姓」、「名」、email アドレスにアクセスしなければなりません。

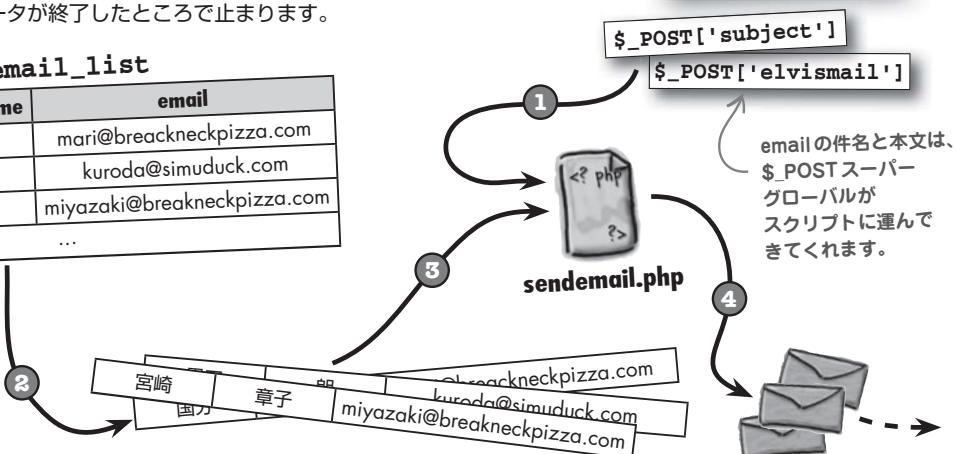
- 各顧客に email メッセージを送るために mb_send_mail() 関数を呼び出します。

email を送るために、email リストの各顧客ごとにループを回すことになります。これは問い合わせ結果の各行のデータについて処理を行うことに相当します。ここで作るループはデータの1行目から始まり、次に2行目に移り、SELECT 問い合わせ文により得られた残りのすべての行について回ります。データが終了したところで止まります。



email_list		
last_name	first_name	email
国分	真理	mari@breackneckpizza.com
黒田	朗	kuroda@simuduck.com
宮崎	章子	miyazaki@breakneckpizza.com
...		

スクリプトは email データを email_list テーブルから取ってくる必要があります。



最初にすべきこと、データを捕まえる。

PHPでフォームからデータを取り出すことに関しては、もはや皆さんは達人の域に到達しています。第1ステップには何も新しいことはありません。単に\$_POSTスーパーグローバルを使って、メールの件名とメッセージ本文を変数にぶち込むだけです。これはこれとして、ちょっと進んで清野君のメールアドレスも変数に格納しておきましょう。こっちも後ほどメールを送るときに必要となります。

```
$from = 'seino@makemeelvis.com';
$subject = $_POST['subject'];
$text = $_POST['elvismail'];
```

清野君のメールアドレスを1つの変数に入れておけば、将来変更する必要が生じた場合でも、この一箇所だけを変更すれば済みます。

メールメッセージのフォームデータも当然変数に入れておきます。

sendemail.phpスクリプトに必要な残りのデータは、清野君のMySQLデータベースからやります。顧客データをemail_listテーブルのデータからスクリプトまで引っ張って来るのには、SELECT問い合わせ文を使います。これまでMySQLターミナルを使ってSELECT文を発行しデータの中身を覗いていたのとは異なり、今回は同様のことをsendemail.phpスクリプトの中からmysqli_query()で問い合わせ文を発行して行います。

```
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query);
```

これが問い合わせ文です。email_listテーブルから(FROM)すべてのカラムを選び出し(SELECT)します。

\$query変数はSQLの問い合わせ文を文字列テキストとして保持しています。

データベースとの接続が問い合わせ文「提出」に必要です。接続に関する詳細は\$dbc変数に格納されています。

mysqli_queryは問い合わせ文を実行しますが、このとき接続用の変数(\$dbc)問い合わせ用文字列(\$query)を使います。

じゃあ残っていることは、\$result変数に入っている問い合わせ結果をなめていくことだけよね。合ってるでしょ？



残念ながら、\$result変数には問い合わせ文のデータは何も入りません。
\$result変数を直接参照しようとしたら、こんな感じに見えるでしょう。

Resource id #3

\$result変数はMySQLリソースのID番号を格納します。問い合わせの結果返された実際のデータではありません。何が起こっているかと言うと、MySQLサーバは一時的に問い合わせの結果を格納し、それに対する一意なリソース番号を付与したのです。これ以降このリソースID番号を使ってmysqli_fetch_array()関数を呼び出すと1回に1行ずつデータを捕らえることができます。

mysqli_fetch_array() で問い合わせ結果を取ってくる

問い合わせ文を実行すると、結果は \$result 变数で捕まえることができます。この变数は mysqli_fetch_array() 関数に食わすと1回に1行ずつテーブルのデータを取ってくすることができます。データの各行は、配列として返ってくるので、例えば \$row という名前で新しい变数に突っ込むことができます。

```
$row = mysqli_fetch_array($result);
```

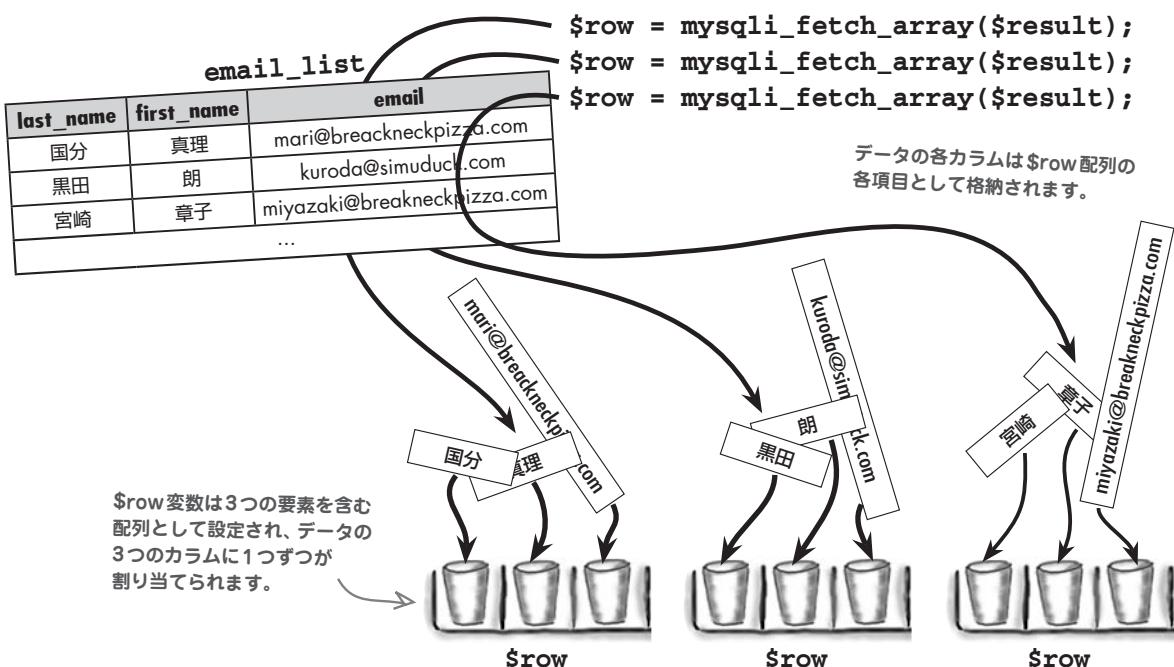
変数 \$row は配列で、最初は問い合わせ結果の第1行目のデータが格納されます。

この関数で問い合わせ結果からデータを1行引き抜き、配列に突っ込みます。

SQL の各問い合わせ文は、それぞれリソース ID 番号を持っていて、その結果に関連付けられたデータにアクセスする際に使われます。

Web サーバ上でこのコードが実行される度に、問い合わせ結果からデータ1行が取り込まれ、\$row 配列に格納されます。 mysqli_fetch_array() 関数を繰り返し呼び出すことで、問い合わせ結果の各行を1ステップずつ実行します。そんなわけで mysqli_fetch_array() 関数の最初の3回の呼び出いで、テーブルからデータの最初の3行を取ってきて、行の各カラムを \$row 配列の各項目として格納します。

mysqli_fetch_array()
関数はデータ1行を
配列に格納します。



WHILE(しばらく)の間ループする

whileループは、何らかの条件に合致するまでの間コードを繰り返すというタイプのループ文です。例えば、顧客サービスアプリケーションで、\$got_customersという変数を使っているとします。この変数は、顧客がサービスを待っているかどうかを表しています。\$got_customersがtrueになっているときは、まだ顧客が残っていると分かるので、next_customer()を呼び出して、次の顧客に対してサービスを提供する、といったことができます。このシナリオをwhileループを使ってコーディングするときのような感じです。

```
顧客がいる限り  
ループし続けます。  
↓  
while ($got_customers) {  
    next_customer(); }  
...  
}  
↓  
ループのコードは波カッコ((と))で  
囲まれているので、何行でも好きなだけ  
実行させることができます。
```

この部分がループ
を通して毎回実行
されるコードです。



whileループは顧客が
もういなくなるまで
回すことができます！

まだ顧客が残っているかどうかを確認するには、**条件をテストします**。条件というのはカッコの中にあるコードです。そこで必ずその結果がYESかNOのどちらになるかを確認します。もし結果がYESつまり**true**であれば、ループ本体を実行します。もし結果がNOつまり**false**であれば、ループを終了します。

`next_customer()`を呼び出して次の顧客のサービスに移る場合、**ループ本体の動作を実行します**。ループ本体というのは、波カッコの内側にあるコードのことです。この部分は、条件が**true**である限り繰り返されます。もしも条件が**false**になるとあれば、ループは終了し、ループ本体はもう繰り返されません。一般的なwhileループは次のようなフォーマットです。

```
↓  
テストの条件は常に結果として  
trueかfalseとなります…(true)の  
場合ループを継続し、(false)の場合  
ループを終了します。  
↓  
while (test_condition) {  
    loop_body  
}  
↓  
ループ本体は、ループを通して  
毎回1回実行されます。
```

脳力発揮

清野君のemail_listテーブルに載っている顧客全員をループするには、どのようにwhileループを使えば良いと思いますか？

whileでデータをループする

whileループを使って清野君のメールリストデータを処理すると一度に1行のデータにアクセスでき、同じコードを2度以上書く必要はありません。`mysqli_fetch_array()`がテーブルから1行を取りってきて、そのカラムの値を配列\$rowにぶち込むということはもう知っています。でも関数自体は、すべてのデータをなめてはくれません。最初の行だけぶち込んだら止まってしまいます。whileループは`mysqli_fetch_array()`を呼び出して、各行の結果のデータを1回に1つずつ、データがなくなるまでなめてくれます。

whileループの条件は、`mysqli_fetch_array()`の返却値で、データがある場合はtrueと解釈され、データがなくなるとfalseと解釈されます。

```
while($row = mysqli_fetch_array($result)) {
    echo $row['last_name'] . ' ' . $row['first_name'] .
        ' : ' . $row['email'] . '<br />';
}
```

ループ本体が、
ループの各回に
実行されます。

ループ本体は、それぞれ
文からなりますが、この
場合行のデータの後ろ
に改行をくっつけます。

ループの1回目は、
配列\$rowは
email_listテーブル
の1行目を保持
しています。

email_list		
last_name	first_name	email
国分	真理	mari@breakneckpizza.com
黒田	朗	kuroda@simuduck.com
宮崎	章子	miyazaki@breakneckpizza.com
...		

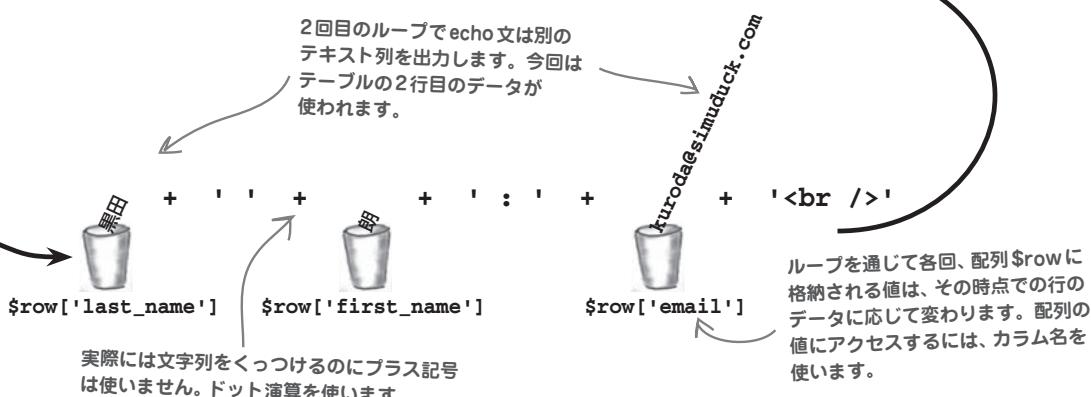
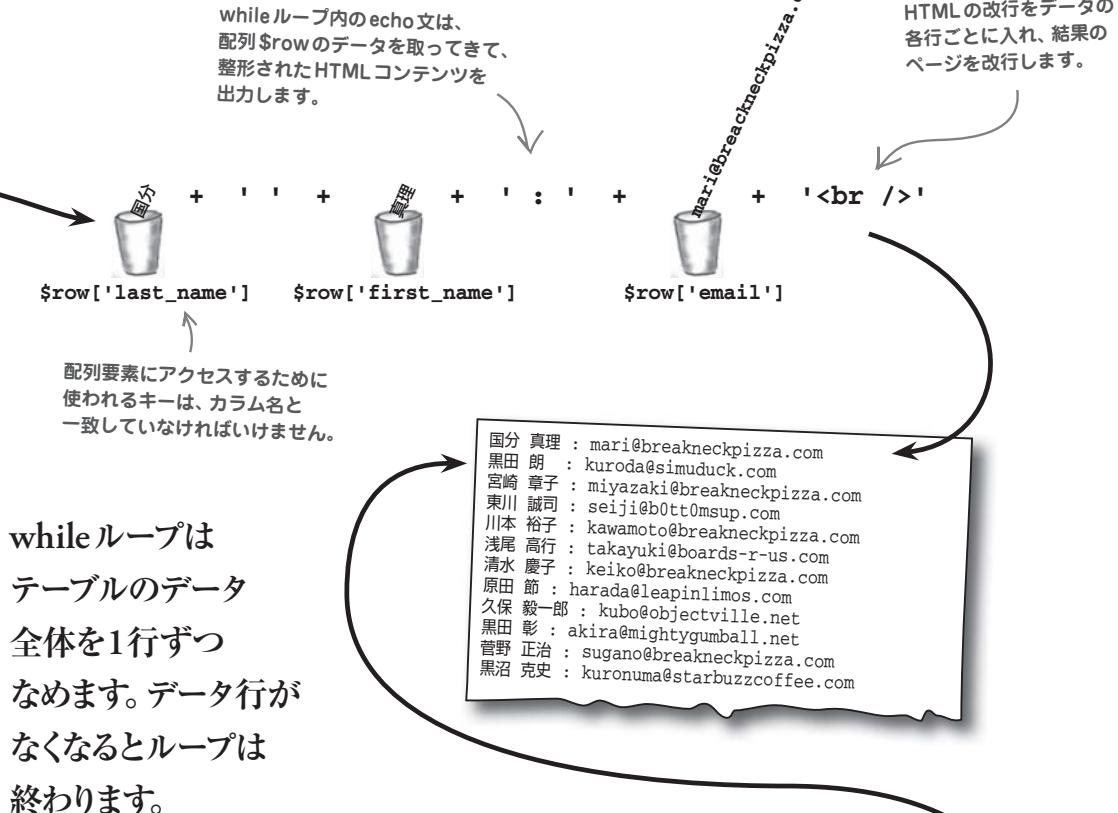
1回目のループ!



2回目のループ!

ループの2回目は、配列\$rowはemail_listテーブルの2行目を保持しています…
何かパターンが見えてきましたか？

さらなるループ…



素朴な疑問に答えます

Q: whileループがループし続けるかどうかを、きちんと知っているのはどうしてでしょう? 言い換えると、whileループはtrue/false条件によってコントロールされているはずなのに、`mysqli_fetch_array()`はリソースIDのようなものを返してきて、変数\$rowにぶち込まれます…どう考えてもこれはテスト条件true/falseのようには見えません。

A: すばらしい観察力です。ご指摘の通り、PHPはtrueの解釈にはかなり自由主義的なところがあります。手っ取り早く言うと、ゼロ(0)またはfalse以外の任意の値はtrue条件成立と解釈します。ですから`mysqli_fetch_array()`関数がデータ行を返してきたときは、配列\$rowはtrueと解釈されます。なぜなら値が0でもfalseでもないからです。そしてテスト条件がtrueなので、ループはボコボコ実行し続けます。面白のは、データがなくなったときに何が起こるかですが、実は`mysqli_fetch_array()`関数は、falseを返します。従ってループが終了します。

Q: じゃあ、whileループは、true/falseだけでなく、どんなデータを使ってコントロールしても良いわけですね?

A: その通りです。でもwhileループは最終的にはデータをtrueかfalseとして解釈しているということは心に留めておいて下さい。重要なことは、他の型のデータを解釈する場合に何がtrueで何がfalseとなるかを理解しておくことです。この簡単な答えは、0またはfalse以外は何でも必ずtrueと解釈されるということです。

Q: `mysqli_fetch_array()`関数がデータを何も返してこなかつたら、whileループはどうなってしまうのですか?

A: 問い合わせ文の結果として何もデータがない場合、`mysqli_fetch_array()`関数はfalseを返します。これによりwhileループはループ本体のコードに二度と陥ることはありません。一度も陥らないこともあります。

Q: では一度もループしないループというのもあり得るのでしょうか?

A: 実際にあります。逆に全然終わらないループというのもあります。こんなwhileループを考えてみて下さい。

```
while (true) {
```

これは無限ループと呼ばれています。テスト条件が絶対にループを終了させないために起こります。無限ループは非常に悪いヤツです。

重要ポイント

- データベースというのは、高度に構造化された様式でデータを格納するための入れ物です。
- テーブルはデータベースの中にカラムと行との格子状のパターンでデータを格納します。
- SQLコマンドCREATE DATABASEは、新しいデータベースを作るときに使います。
- SQLコマンドCREATE TABLEは、データベースの中にテーブルを作りますが、テーブル内のカラムデータに関する詳細な情報が必要です。
- データベースからテーブルを削除するには、SQLコマンドDROP TABLEを使います。
- `mysqli_fetch_array()`関数はデータベース問い合わせ文の結果から、データの1行を取ってきます。
- whileループは、テスト条件が合致する間、PHPコードの一部分を繰り返し実行します。

① メールリスト用のデータベースを作る。

② メール追加用WebフォームとPHPスクリプトを作って新しい顧客をリストに追加できるようにする。

③ メール送信用WebフォームとPHPスクリプトを作ってリストの顧客全員にメールを送れるようにする。

忘れないで下さい。仕上げるには、まだ最後のステップが残っています。



PHP & MySQLマグネット

下にあるマグネット部品を使って、メール送信用スクリプトのコードを仕上げて下さい。これで清野君は顧客リストにメールを送ることができます。念のため、mb_internal_encoding()関数とmb_send_mail()関数がどのような機能かを再度示します。

```
mb_internal_encoding("UTF-8");
mb_send_mail(to, subject, msg, 'From:' . from);
```

```
<?php
$from = 'seino@makemeelvis.com';
$subject = ..... ;
$text = ..... ;
$dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')
or die('エラー：MySQLサーバとの接続に失敗しました。');

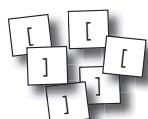
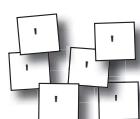
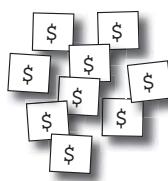
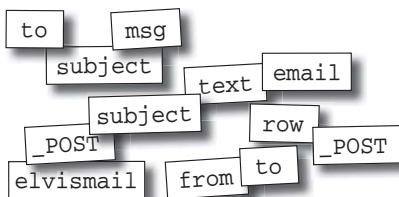
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query)
or die('エラー：データベース問い合わせに失敗しました。');

while ($row = mysqli_fetch_array($result)){
    $last_name = $row['last_name'];
    $first_name = $row['first_name'];
    $msg = "$last_name $first_name さん、\n.....";
    $to = ..... ;
    mb_internal_encoding("UTF-8");
    mb_send_mail(....., ..... , ..... , 'From:' . ..... );
    echo 'メールを' . ..... . 'へ送信しました。  


```



sendemail.php





PHP & MySQL マグネット

下にあるマグネット部品を使って、メール送信用スクリプトのコードを仕上げて下さい。これで清野君は顧客リストにメールを送ることができるようになります。念のため、mb_internal_encoding() 関数と mb_send_mail() 関数がどのような機能かを再度示します。

```
mb_internal_encoding("UTF-8");
mb_send_mail(to, subject, msg, 'From:' . from);
```

この部分ご自身の
メールアドレスに
変更することを
忘れない。

```
<?php
$from = 'seino@makemeelvis.com';
$subject = $_POST['subject'];
$text = $_POST['elvismail'];
dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store');
or die('エラー: MySQL サーバとの接続に失敗しました。');
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query);
or die('エラー: データベース問い合わせに失敗しました。');
while ($row = mysqli_fetch_array($result)) {
    $last_name = $row['last_name'];
    $first_name = $row['first_name'];
    $msg = "$last_name $first_name さん、$text";
}
$to = $row['email'];
mb_internal_encoding("UTF-8");
mb_send_mail($to, $subject, $msg, 'From:' . $from);
echo 'メールを' . $to . 'へ送信しました。<br />';
mysqli_close($dbc);
?>
```

データベース中の "email" カラムが、顧客のメールアドレスを保持しています。このアドレスにメールアドレスを設定する必要があります。

確認画面のメッセージとしてメールを送った各顧客のメールアドレスとともにページに出力します。

メール受信者、メッセージの件名、メッセージ本文に清野君の "from" アドレスをくっつけて mb_send_mail() 関数に渡します。

sendmail.php

セキュリティ上の観点から、ユーザの入力したものを何もチェックせずに直接 mb_send_mail() 関数に渡すというのあまりお勧めできません。6 章でこの問題を解消するための技法を明らかにします。



試運転

メール送信用フォームを使ってメーリングリストにメールを送る

メール送信用のWebページのコードをWebサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードします。コードはch03 フォルダにあります。前に見たemail追加用のページと同様ですが、このコードはWebフォームsendemail.html、スタイルシート(style.css)、イメージファイル(elvislogo.gifとblankface.jpg)とから構成されています。

新しくテキストファイルをsendemail.phpという名前で作り、反対側のページのコードを全部打ち込みます。全ファイルをWebサーバにアップロードしたら、sendmail.htmlページをWebブラウザで開きます。メールメッセージをフォームに入力したら、「送信」をクリックします。

メッセージを自分で受け取るためにには、ご自身のメールアドレスをメーリングリストに登録しておかなければならぬことを忘れないで下さい。

メールが来ました…清野君から！

やつと清野君は新しいメール送信用のWebフォームとPHPスクリプトを使って、メーリングリストにMakeMeElvis.comのセールのメールを送り出すことができました。清野君はスクリプトからの出力画面で各メッセージがちゃんと送られているかどうかを確認することもできます。スクリプトのwhileループが実行される度に、清野君は"メールをsomeone@somewhere.comへ送信しました。"という風にデータベースに登録された人のメールアドレスを見ることができます。結果として、清野君の製品の宣伝効果が上がり、善し悪しはともかくとして、エルビス風の人気が増えることになります！

メール送信用スクリプトは、データベースに格納されたアドレスに本当にメールを送ってしまいます。これを改造して使うときは本気で注意して下さい！

青いスエードの靴を売り切った…金持ちだ！

ビッグセール！

件名： ビッグセール！

熱狂的エルビスファンの皆様、

今週はMakeMeElvis.comのビッグセール！

馬のタデガミ（本物）製の付けモミアゲが2枚のレザースーツ「1着につきもう1着無料」もお

残り3日です！

メールをharada@leapinlimos.comへ送信しました。
メールをmaru@breakneckpizza.comへ送信しました。
メールをkeiko@breakneckpizza.comへ送信しました。
メールをkawamoto@breakneckpizza.comへ送信しました。
メールをseiji@bottomsup.comへ送信しました。
メールをmiyazaki@breakneckpizza.comへ送信しました。
メールをkuroda@simuduck.comへ送信しました。
メールをkubo@objectville.netへ送信しました。
メールをakira@mightygumball.netへ送信しました。
メールをsugano@breakneckpizza.comへ送信しました。
メールをkuronuma@starbuzzcoffee.comへ送信しました。
メールをseki@chocoholic-inc.comへ送信しました。
メールをzenba@honey-dolt.comへ送信しました。
メールをkintaka@objectville.netへ送信しました。
メールをsugimoto@breakneckpizza.comへ送信しました。
メールをishikawa@starbuzzcoffee.comへ送信しました。
メールをkawamura@starbuzzcoffee.comへ送信しました。
メールをsasaki@objectville.netへ送信しました。
メールをhoshino@mightygumball.netへ送信しました。
メールをyamazaki@lilbeanlounge.comへ送信しました。

時には出て行く人も

花開こうとする新しいビジネスにはありがちなことです、道は険しいのが常です。どこかのエルビスファンがKingに鞍替えし、清野君のメーリングリストから抜けたいらしいのです。清野君は誠実に対応したいと思っていますが、そのためにはデータベースからそのような顧客を取り除く必要があります。

清野様、
今後エルビストアからのセール
スメールはもう受け取りたくない
ません。今でもエルビスのファン
ではありますが、その手のグッズ
ではありません。私のアド
は興味がありません。私のアド
レスをリストから取り除いて下さ
い。私のメールアドレスはam@
honey-doit.comです。
よろしくお願いします。

コスプレ好

担当の方へ、
本物の馬の毛を使った付けモミア
ゲによりアレルギー反応が出てし
まったため、エルビスに扮装する
のは、私には無理だと悟りました。
ケープはすごく良かったのですが、
付けモミアゲはダメでした。メー
ルリストから私を取り除いて下さ
い。
よろしくお願いします。

伊藤 拓

ito@objectville.net

ケツ振り仲間で担当様†、
エルビスの魂の動きは今でも好き
なのですが、もうそれほどのめり
込まなくなってしまいました。今
はLiberaceの控えめなパフォー
マンスと卓越したピアノスキルの
方が好きです。このemailアドレ
スです(削除をお願いします)。
toku@tikibeanlounge.com
ではよろしく
森口 広美

全員があのKingのコスプレを
したくて抜けていくわけではない
のだろう。このメールをくれた人たち
だけをリストから取り除いて、本物の
ファンにフォーカスしよう。

† 訳注：原文は「Dear Fellow Hip Swiveler」となっていました。エルビスプレスリー独特のお尻を回すダンスのことを指しているようです。

清野君は顧客を失っていくので、
あまり幸せではありません。でも
希望者の要求に誠実に答えて
メーリングリストから取り除いて
あげたいと考えています。

MySQLが活動する上で当然のことです。時にはデータをデータベー
スから取り除く必要があります。清野君はアプリケーションを拡張して、
email_listテーブルからユーザを取り除く必要があります。

清野君がメール削除機能を実装する上で新しく必要となると
思われるアプリケーション部品を書いてみましょう。



DELETEでデータを取り除く

テーブルからデータを削除するには、新しいSQLコマンドDELETEが必要です。新しくメール削除用スクリプトを作り、そこで顧客のデータを清野君のメーリングリストから削除するためにDELETEを使います。実際、新しいスクリプトと、それを動かす新しいWebフォームが必要になります…でも最初に必要なのはDELETEです。

SQLコマンドDELETEは、テーブルからデータ行を削除します。このためこのコマンドを使うときは細心の注意を要します。なぜなら、このコマンドは一瞬のうちにテーブルの全データを消し去ってしまう能力を持ち合わせているからです。このことを踏まえて、DELETEの最も危険な形式を以下に示しますが、これはテーブルからすべての行を消してしまいます。

DELETE FROM table_name

何も修飾子を指定しないと、DELETEコマンドはテーブルの全データを完全に空にします。

ここに削除したい行のある
テーブルの名前を指定します。

じゃあ
全部消しちゃう以外に
テーブルから何かを消すことは
できないっていうこと？

そんなことはありません。DELETEを使ってピンポイントに特定の1行や数行を削除することができます。

正確に目的とする1行または数行をDELETEに指定して削除しようとするなら、WHERE節を組み合わせる必要があります。WHERE節をSELECTコマンドと一緒に使ったときのことを思い出して下さい。WHEREを使えば、問い合わせ文から特定の行をより分けることができます。

新しいステップが必要になってしまったように見えます…設計の仕様変更はよくあることです！



自分で考えてみよう

清野君には「名」が「理絵」という顧客が23人、「姓」が「福本」という顧客が11人、氏名が福本理絵という顧客は一人だとします。以下の各問い合わせ文により何行のデータが削除されるか書いてみましょう。

```
DELETE FROM email_list WHERE first_name = '理絵';
```

```
DELETE FROM email_list WHERE first_name = '理絵' OR last_name = '福本';
```

```
DELETE FROM email_list WHERE last_name = 理絵;
```

自分で考えてみよう の答え

清野君には「名」が「理絵」という顧客が23人、「姓」が「福本」という顧客が11人、氏名が福本理絵という顧客は一人だとします。以下の各問い合わせ文により何行のデータが削除されるか書いてみましょう。

`DELETE FROM email_list WHERE first_name = '理絵';`

23

`DELETE FROM email_list WHERE first_name = '理絵' OR last_name = '福本';`

33

`DELETE FROM email_list WHERE last_name = 理絵;`

0

ひっかけ問題です！「名」が引用符で囲まれていませんので、削除される行はありません。すべてのテキストの値は引用符で囲まなければなりません。

WHERE を使って特定のデータを DELETE する

`DELETE` コマンドに `WHERE` 節を使うと、特定の行のデータをターゲットとして削除することができます。テーブル全体を空にしてしまうのではありません。`WHERE` 節は削除しようとする行だけにフォーカスすることができます。この場合だと、清野君の顧客のうちメールリストから削除して欲しいと言っている人だけです。

`DELETE FROM email_list`

`WHERE email = 'kawamura@starbuzzcoffee.com'`

テーブルカラムの
名前です。

マッチすべき値です。

WHERE 節のこの部分が各行を
テストして条件にマッチするか
どうかを確認します。

WHERE 節は
問い合わせ文
から特定の
データ行に
フォーカスして
結果を狭めます。

WHERE 節の実際のテストは、テーブルの各行に対して順番に比較をすることにより推移していきます。この例では、等号(=)が `email` カラムのそれぞれの値の中で、"kawamura@starbuzzcoffee.com" と等しいものがあるかどうかをテストします。ある行の `email` カラムの値がマッチした場合、その行は削除されることになります。

WHERE 節に `first_name` や `last_name` ではなく、`email` カラムを使ったのは何故だと思いますか？書いてみて下さい。

間違つて削除してしまうというリスクを最小化する

以下で述べることは非常に重要なちやんと理解して下さい。WHERE節でマッチさせる行の対象として任意のカラムを指定することができますが、清野君のDELETE問い合わせ文に関してemailカラムを何故選んだのかということについては極めて明白な理由があります。2つ以上の行がWHERE節の条件にマッチする場合のことを考えてみて下さい。この場合、条件にマッチしたすべての行が削除されてしまいます。ですからWHERE節がピンポイントで削除したい行を指しているかどうかは清野君にとって重要なことなのです。

本気で考えなければならないのは單一性の問題です。ほぼ安全と仮定できるとして、メールアドレスがemail_listテーブルにおいては單一であるということです。これに対して「姓」や「名」は單一と仮定できません。last_nameカラムが「川村」であるというWHERE節を作つて、一人の顧客を削除しようとは思わないでしょう。そんなことをしたら「川村」という「姓」のすべての顧客が消えてしまいます。以上のような理由から、清野君のWHERE節はemailカラムで特定のマッチングを探すという具合に細心の注意を払い精巧に作られているのです。

DELETE FROM email_list

WHERE email = 'kawamura@starbuzzcoffee.com'

emailカラムをWHERE節で使うことで單一性を確保でき、あらぬ行を間違つて削除してしまうというリスクを軽減することができます。

WHERE節にemailの代わりにlast_nameを使ったとしたら、このユーザも間違つて消されてしまっていたでしょう。

**DELETE文に
WHERE節を
付けると削除したい
行をピンポイントで
指定できます。**

問い合わせ文DELETE
は、データベースから
この行を削除します…
もう2度と出会うこと
はありません！

email_list		
last_name	first_name	email
黒沼	克史	kuronuma@starbuzzcoffee.com
関	昌美	seki@chocoholic-inc.com
川村	忠範	kawamura@starbuzzcoffee.com
善波	佳也	zenba@honey-doit.com
金高	雅仁	kintaka@objectville.net
杉本	雅幸	sugimoto@breakneckpizza.com
石川	安則	ishikawa@starbuzzcoffee.com
佐々木	久郎	sasaki@objectville.net
星野	憲一	hoshino@mightygumball.net
山崎	謙一	yamazaki@tikibeanlounge.com
川村	公俊	kawamura@b0tt0msup.com
...		

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> DELETE FROM email_list WHERE email='kawamura@starbuzzcoffee.com';
Query OK, 1 row affected (0.00 sec)

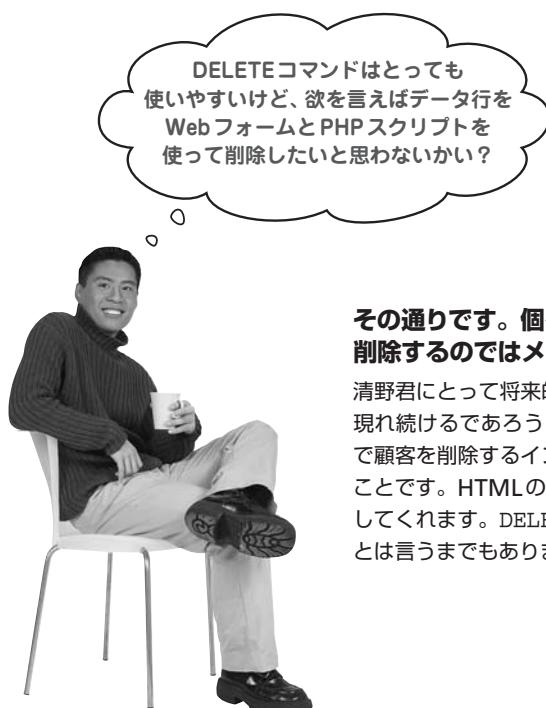
mysql>
```



試運転

清野君のデータベースにDELETEコマンドを試してみる

MySQLツールに点火したら、DELETEコマンドをいくつか試して個々のデータ行を email_list テーブルから顧客のメールアドレスをもとに削除してみましょう。注意しなければいけないことは、DELETE文には WHERE 節を必ず含めることです。そうしないと誤ってテーブル全体を消し去ってしまうことになります。



DELETEコマンドはとっても
使いやすいけど、欲を言えばデータ行を
WebフォームとPHPスクリプトを
使って削除したいと思わないかい？

その通りです。個々の問い合わせ文を手で打ちこんでユーザを
削除するのではマーリングリストを管理していることになりません。

清野君にとって将来的にもマーリングリストからの削除を希望するユーザ
現れ続けるであろうということは避けられないことなので、Webベース
で顧客を削除するインターフェースを開発するというのは非常に意味のある
ことです。HTMLのWebフォームとPHPスクリプトが、この策略を解決
してくれます。DELETE FROM問い合わせ文をWHERE節とともに使うこ
とは言うまでもありません…

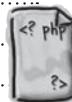


このフォームフィールド
の名前は"email"です。

「削除」ボタンを押すと
フォームはPHPスクリプト
に対しPOSTリクエストを
「提出」します。

```
<?php  
$dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')  
or die(' エラー : MySQL サーバとの接続に失敗しました。')
```

```
?>
```



removeemail.php



清野君は顧客をメーリングリストから削除するためにWebフォーム(removeemail.html)を作りました。フォームが受け付けるものは、メールアドレスのみで、これをemailという名のHTMLのフォームフィールドに入力します。清野君のremoveemail.phpスクリプトを仕上げて下さい。このスクリプトはHTMLフォームから呼び出され、各顧客の削除を行います。



このフォームフィールド
の名前は"email"です。

「削除」ボタンを押すとフォームは
PHPスクリプトに対しPOST
リクエストを「提出」します。

`$_POST`に格納されたemailの
フォームデータは、変数に格納され、
その後DELETE問い合わせ文で
使われます。

removeemail.html

```
<?php
$dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')
or die('エラー：MySQL サーバとの接続に失敗しました。');
$email = $_POST['email'];

$query = "DELETE FROM email_list WHERE email = '$email'";
mysqli_query($dbc, $query)
or die('データベース問い合わせに失敗しました。');

echo 'メールアドレス :.&email! を削除しました。';

mysqli_close($dbc);
?>
```

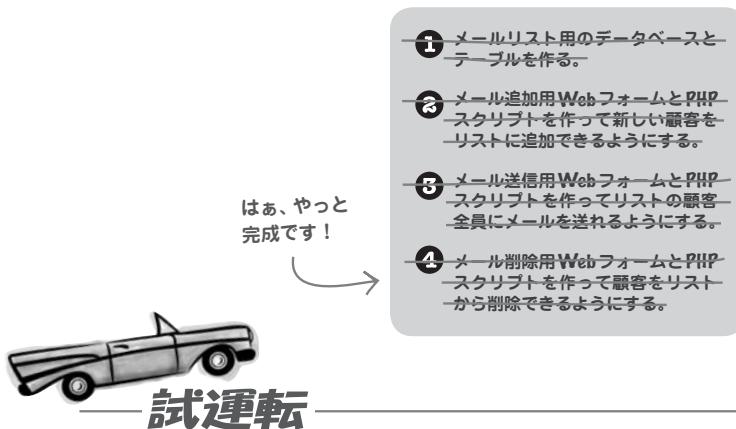
データベース接続をちゃんと
閉じて後始末を忘れずに。

この単一引用符と二重引
用符の使い方には注意し
て下さい！二重引用符は
SQL文全体を囲んでお
り、単一引用符は\$email
に格納されたメール
アドレスを囲んでいます。



removeemail.php

何が起こったかの確認に
ついては、特にデータ
ベースの削除に関しては
気にしなくて構いません。



試運転

顧客をメール削除用フォームを使ってメーリングリストから削除する。

この書き出しにはそろそろ慣れてきましたよね？ メール削除用のWebページをWebサイト (<http://www.oreilly.co.jp/books/9784873114446/>) からダウンロードします。コードはch03 フォルダにあります。コードはWebフォームremoveemail.htmlとスタイルシート(style.css)とイメージファイル(elvislogo.gifとblankface.jpg)とからなります。

新しくテキストファイルを作つてremoveemail.phpという名前を付け、反対側のページのコードをすべて入力します。すべてのファイルをWebサーバにアップロードしたら、removeemail.htmlページをWebブラウザで開きます。顧客のメールアドレスをフォームに入力したら、「削除」ボタンをクリックしてデータベースから削除します。



メーリングリストアプリの完成です！

MakeMeElvis.comはWebアプリケーションです

もはや公開版です。PHPとMySQLのおかげで、清野君のWebサイト MakeMeElvis.com は、もはやアプリケーションと呼ぶに値するものとなりました。清野君はデータを恒久的に MySQLデータベースに格納しておくことができますし、Webフォームを通じてデータをやり取りすることもできます。HTMLページとPHPスクリプトとコードに組み込まれたSQL問い合わせ文を組み合わせて、清野君はメールリストに顧客を追加したり削除したりすることができます(顧客が自分で自分を追加することもできます)。さらにリストの全員に対してメールメッセージを送ることもできます。



PHP & MySQL 道具箱

清野君のWebアプリケーションを手伝って無事離陸させることができたのはもちろんのこと、この章で PHPとMySQLに関する貴重なスキルを開拓することができました。例えば…

while

PHPのループを構成します。コードの一部分を特定の条件が true である間繰り返します。while ループが特に便利なのは、SQL の問い合わせ結果のデータ列に対してループして使うことができることです。

DELETE FROM tableName

このSQL文を使ってテーブルから行を削除します。文の使い方次第で特定の行を削除することも、たくさんの行を削除することもできます。

WHERE

このSQL節は他のSQLコマンドと共に使われて文を構成し、テーブル内で対象となる特定の行を指定します。例えば、特定の値とマッチするカラムを持つ行だけを、より分けることができます。

mysqli_fetch_array()

PHPの組み込み関数でデータベースの問い合わせ結果から1行のデータを取り出します。この関数を繰り返し呼ぶことでデータ行を次々と読み込むことができます。

DROP TABLE tableName

このSQL文はデータベースからテーブル全体を消します。つまりテーブルは削除され、それに伴ってテーブル内に格納されている全データもなくなります。

DESCRIBE tableName

テーブルの構造を知りたくなったら、必要なのがこのSQL文です。この文はデータを見せてはくれませんが、カラム名とそれに対応するデータ型を見せてくれます。

SELECT * FROM tableName

このSQL文は、テーブルから行を抜き出します。星(*)が使われた場合、テーブル内の行にあるすべてのカラムが返ってきます。もっと特定したい場合は、個別にカラム名を指定することができます。問い合わせ文から返ってくる全カラムのデータを見たくないときに使います。

4章 現実的で実用的なアプリケーション

Web上のアプリケーション

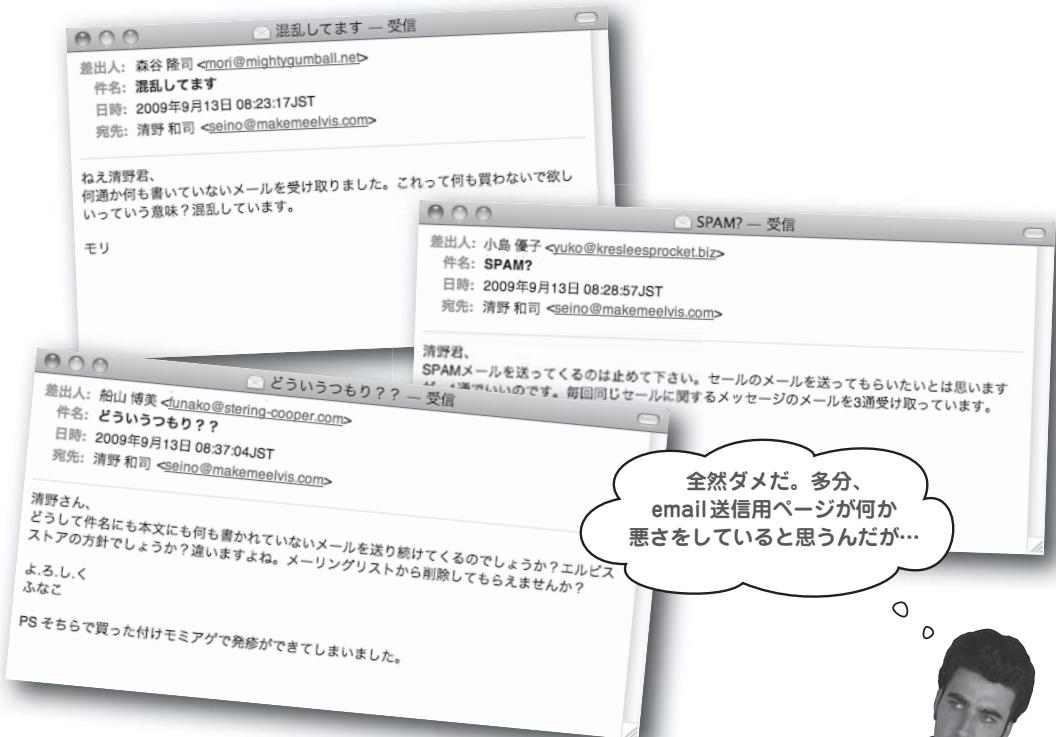


時には現実的になって計画を見直さなければなりません。

そうでなければ、もっと初期の段階で十分注意深く計画を練る必要があります。アプリケーションがWeb上に出て行ってしまってから、あまり十分に計画を練っていないかったと気がつくかもしれません。自分で動くだろうと思っていたことは、現実の世界では不十分なものです。本章では、現実の世界での問題をいくつか見ていきます。このようなことはご自身のアプリケーションがテストを経て実サイトに出て行った時に実際に起こりうることなのです。問題を解決するために、いままでより重要なPHPとMySQLのコードも見ていきます。

清野君はいらっしゃっている顧客を作り出してしまいました

清野君の顧客メーリングリストは、どんどん大きくなっています。ところが清野君のメールは不満をも生成してしまったのです。不満の内容は多岐に渡りますが、顧客が空白のメールメッセージを受け取ることや同じメールを何度も受け取ることが原因のようです。どちらも全然ダメです。清野君は何が原因で問題が起こっているのかを見極め直す必要があります。彼のビジネスにとって死活問題です。



清野君には問題が起こっていることは分かっています。でもどこが悪いのかを特定するためには誰かに助けてもらう必要があるでしょう。

清野君を…清野君から守る

実は「オペレータの間違い」というのが、まず第一に問題です。清野君がメールの情報を打ち込まず、うっかり「送信」ボタンをクリックしてしまうと、空白のメールがリストの全員に飛んでしまうのです。Webフォームが完全に意図した通りに使われるなどという甘い考えは捨てなければなりません。従ってスキのないPHPスクリプトを作らなければなりません。つまり誰かユーザが間違ったフォームの使い方をしても、これを予想してこの種の問題が起らないように予防策を講じる必要があります。

現状のsendemail.phpスクリプトのコードを見てみましょう。どのように清野君は空のメールを飛ばしてしまっているのでしょうか。

メール送信用スクリプトはフォームから
来たテキストを使ってメールを作り上げます。
たとえユーザが何も入力しなくとも、です。



```
<?php
    $from = 'seino@makemeelvis.com';
    $subject = $_POST['subject'];
    $text = $_POST['elvismail'];           ← フォームのテキストは$_POST['subject']と
                                         $POST['elvismail']とから取ってきて、
                                         $subjectと$textに、それぞれ格納されます…

    $dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')
        or die('エラー：MySQL サーバとの接続に失敗しました。');

    $query = "SELECT * FROM email_list";
    $result = mysqli_query($dbc, $query)
        or die('エラー：データベース問い合わせに失敗しました。');

    while ($row = mysqli_fetch_array($result)){
        $last_name = $row['last_name'];
        $first_name = $row['first_name'];
        $msg = "$last_name $first_name さん、\n $text ";
        $to = $row['email'];

        mb_internal_encoding("UTF-8");
        mb_send_mail($to, $subject, $msg, 'From: ' . $from);
        echo 'メールを' . $to . 'へ送信しました。<br />';
    }

    mysqli_close($dbc);
?>
```

問題は、\$textをメッセージに使うときに
変数に実際にテキストが入っているのか
どうかを見ていません。

…さらに\$subjectもテキストが
入っているかどうかを見ずに
使っています。

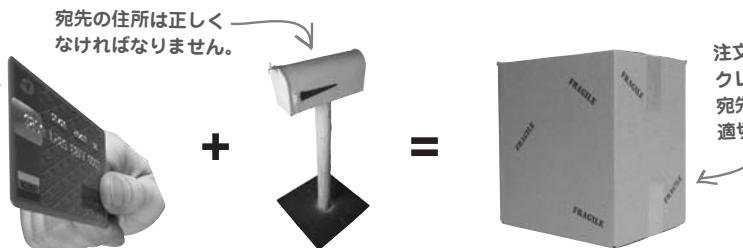
空のメールが飛んでしまう問題に対処するため sendemail.php を
どのように変更するべきだと思うか書いてみましょう。

ちゃんとしたフォームデータを要求する

清野君のメール送信用フォームは妥当性検証を必要としています。妥当性検証とは何かを実行する前にフォームデータがOKかどうかを確認するチェックプロセスのことです。清野君は、そうは思っていませんが、実はすでに妥当性検証をしています。Elvisグッズの注文を受けるといつでも、即座にそれに応じるなどということはしません…まず始めに妥当性検証をしています！

注文の場合、清野君はまず顧客のクレジットカードが妥当かどうかチェックします。大丈夫であれば、注文に応じ、発送準備にかかります。しかし発送前に顧客の宛先の住所が正しいかどうかチェックしなければなりません。このチェックにもパスした場合、清野君は次のステップに進むことができ、注文の品を送ります。清野君の店で正しく注文を完了するということは、常に注文のデータに対する妥当性検証の結果に依存しているのです。

清野君は注文に応じる前に各顧客のクレジットカードの妥当性検証をしなければなりません。



清野君が空のメールを送ってしまう問題を解決するには、sendemail.phpスクリプトに送られてきたフォームのデータについて妥当性検証をする必要があります。これはフォームのデータがクライアントのWebページ(sendemail.html)からサーバに送られてきたら、サーバ(sendemail.php)はすべてのデータが存在するかを確認するというチェックをすることを意味します。sendemail.phpにコードを追加し、テキストボックスの値を確認し、中身が空でないことを確認するというチェックをしましょう。すべてのチェックがOKだったら、スクリプトはメールを飛ばします。

- 1 清野君はメール送信用フォームを埋めて「提出」します。

ねえねえ、サーバ。
清野君のフォームデータ渡したいんだけど。

- 2 フォームのデータがサーバ上のメール送信用スクリプトに送られます。

```
<form action = "sendemail.php"  
...>
```



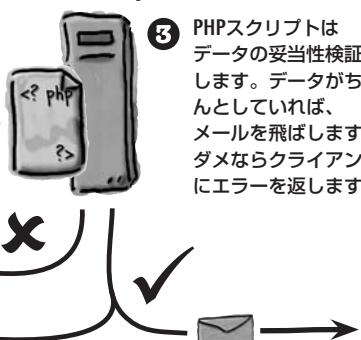
- 4 サーバはHTMLレスポンスをブラウザに返します。メールがちゃんと飛んだか、またはフォームのデータが妥当でないかのいずれかです。

妥当性検証とは
得られたデータが
欲しいデータか
どうかを確認する
ことです。

注文が発送されるのは、
クレジットカードと
宛先の住所の両方が
適切な場合だけです。

データが全部
イケてれば、メールを
送ってあげる。

- 3 PHPスクリプトはデータの妥当性検証をします。データがちゃんとしていれば、メールを飛ばします。ダメならクライアントにエラーを返します。



メール送信用画面の妥当性検証の背後にある論理

清野君は sendemail.html フォームから受け取るデータの妥当性を検証してからでないと、メールを送ってはいけません。実際、メール送信というのは、データの妥当性検証の結果に完全に依存すべきものなのです。PHPで本来やるべきだったことは、sendemail.php スクリプトで受け取ったフォームデータの妥当性を検証し、その上で決定を下すということです。必要なコードは「もしデータが妥当であれば、先に進みメールを飛ばす。」というものです。

この2つの条件を両方とも
満たした場合のみ、データは
妥当であるとみなします。

IF 件名がテキストを含んでいる **AND** 本文がテキストを含んでいる
THEN メールを飛ばす

両方の条件を満たす場合、幸せいっぱいなので、
メールを送り出すことができます。

今まで、2つのフォームフィールドに
何が入力されたかを、それがあっても
なくとも、全く気にせずにメールを
飛ばしていました。

妥当性検証のおかげで、どちらかの
フィールドが空の場合は、メールは
飛ばさないということを保障できます。



素朴な疑問に答えます

Q: サーバ側ではなくクライアント側でのデータの妥当性検証というのを聞いたことがあります。これはどのように動くのでしょうか？

A: Web ブラウザはクライアントと考えることができますので、クライアント側での妥当性検証とは、データが PHP スクリプトに送られる前に行われるすべてのチェックです。JavaScript のような言語では、クライアント側での妥当性検証を行うことができます。この辺りについてご興味がおありでしたら、『Head First JavaScript』をお勧めします。この本ではクライアント側での妥当性検証についてディープに議論しています。

Q: ではなぜクライアント側ではなくサーバ側での妥当性検証を行なうのですか？

A: クライアントの妥当性検証では、問題の一部しか解決できません。清野君がやろうと思えば sendemail.php 自体を直接ブラウザから閲覧することで、空のメールが飛んでしまいます。でもサーバ側で妥当性検証をすれば、2つの問題を同時に解決できます。フォームにデータが入っていない場合も検出できますし、直接ロードされた PHP スクリプトにデータが入っていない場合も検出できます。誤解しないで頂きたいのですが、クライアント側での妥当性検証が間違っているとは言っていません。むしろ良いことです。しかしサーバというのはおかしなフォームデータを捕まえる最後の砦ですので、サーバ側での妥当性検証を無視することはできないのです。

IFを使って判断を下せます

PHPのif文は何かがtrueかどうかに基づき判断を下すことができます。清野君の注文をもう一度考えてみましょう。注文に応じる前に、清野君は支払いを確定しなければなりません。これは顧客のクレジットカードにチャージすることを意味しています。仮に顧客がおかしなカード番号を教えた場合、清野君は注文に応じることができません。つまり清野君は実世界での妥当性検証をすべての注文について、次のような感じで行っています。

もし顧客のクレジットカードのチェックが成功すれば、次へ進み注文に応じる。

このシナリオをPHPコードのif文を使って翻訳することができます。if文とは、正にこの種の判断を処理するために設計されています。

基本的なif文は以下の3つの部分からなります。

① 予約語 if

これがif文の始まりを表します。

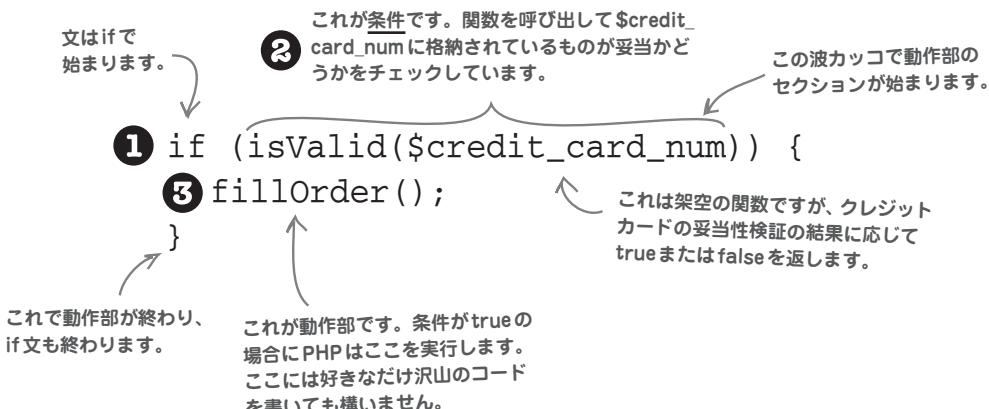
② テスト条件

テスト条件または条件式は、予約語ifの直後のカッコ内にあります。

ここに妥当性検証、つまり真偽を決めたい式を書きます。

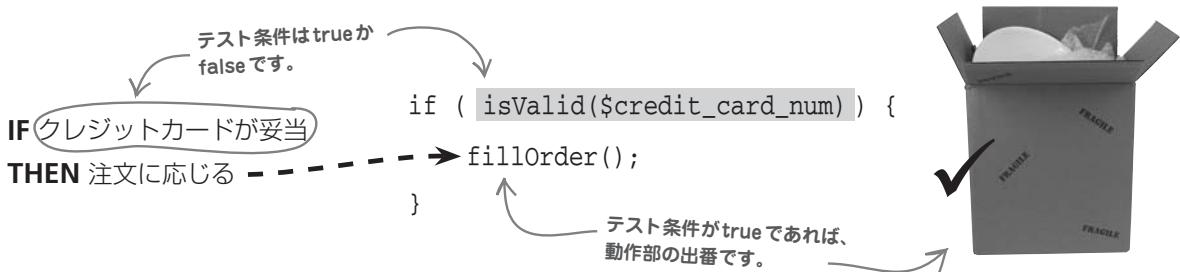
③ 動作部

ifの動作部は、テスト条件の直後であり、波カッコでくくられています。ここに条件がtrueの場合に実行したいPHPコードを書きます。

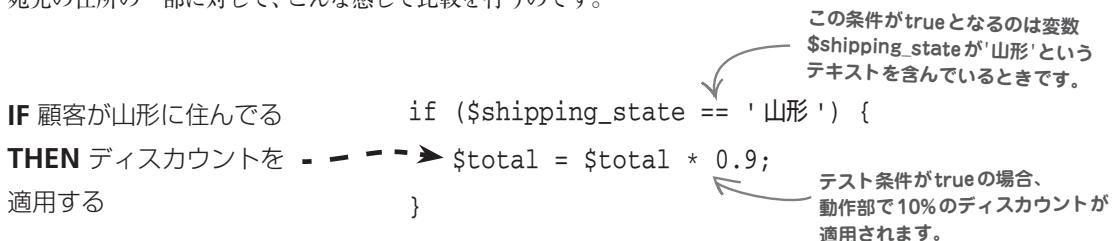


真偽値のテスト

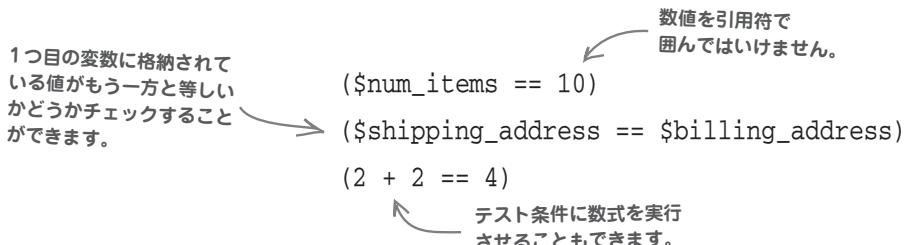
`if` 文の心臓部は、テスト条件です。この部分は常に `true` または `false` のいずれかと解釈されます。テスト条件は、変数でも関数呼び出しでも 2 つのものの比較でも、あるいはその他でも構いません。清野君のクレジットカード妥当性検証では、テスト条件として関数呼び出しを当てにしていました。つまり関数の返却値が `true` か `false` のいずれかという意味です。



テスト条件として非常によく用いられるのは比較で、典型的なのは変数と何らかの値とを比較するというものです。例えば、清野君が山形県に住んでいる顧客に対してディスカウントをしようと思っているとしましょう。この場合、清野君は次のような `if` 文を作ることになるでしょう。つまり、宛先の住所の一部に対して、こんな感じで比較を行うのです。



このテスト条件は、2 つの等号 (`==`) を使って、等価性に関する比較を実行します。等価性比較は、変数と文字列だけで行われるものではありません。変数と数値を比較することも、変数と変数を比較することも、計算をしてその結果を比較することもできます。



IFは等価性以外もチェックします

`if` 文は等価性以外もチェックできます。`if` 文のテスト条件では、値が別の値よりも大きいかどうかもチェックできます。成り立った場合、条件の結果は `true` となり、動作部が実行されます。以下に `if` 文の制御を決めるこことできるテストをもう少し挙げておきます。

`if` 文全部を1行に書いても、動作部が比較的単純ならOKです(訳注: これは単に読み易さだけの問題で、推奨事項です)。

この2つの変数に値が既に設定されていることにします。

```
$small_number = 2;
$big_number = 98065; これは2つの条件はtrueとなります。
```

対象となっているものが等しくないかどうかをチェックする方法は2つあります。`<>` と `!=` です。この2つは `==` による等価性テストの逆の結果を返します。

```
if ($small_number <> $big_number) { echo 'True'; }
if ($small_number != $big_number) { echo 'True'; }
```

「より本当に大きい」[†] の記号(`>`)は、左の値が右の値よりも本当に大きいかどうかをチェックします。もし成り立てば、条件は `true` となり、そうでなければ `false` となります。

この条件は `false` です。

```
if ($small_number > $big_number) { echo 'True'; }
```

左の値の方が右よりも本当に小さければ条件は `true` となります。

この条件は `true` です。

```
if ($small_number < $big_number) { echo 'True'; }
```

「に等しいかまたは大きい」の記号(`>=`)は、「より本当に大きい」(`>`)と同様ですが、2つの値が等しい場合も `true` を返します。

この条件は `false` です。

```
if ($small_number >= $big_number) { echo 'True'; }
```

「に等しいかまたは小さい」の記号(`<=`)は、「より本当に小さい」(`<`)と同様ですが、2つの値が等しい場合も `true` を返します。

この条件は `true` です。

```
if ($small_number <= $big_number) { echo 'True'; }
```



○ ○

文字列同士はどうなの?
("dog" > "cat")なんて
ちゃんと動く?

もちろんです。文字列同士も `if` テスト条件で比較できます。

アルファベットを基準に動きます。つまり `a` は `z` よりも本当に小さいとみなされます。「より本当に大きい」や「より本当に小さい」は、情報をアルファベットの辞書順に表示する必要があるときに役に立ちます。



if文のテスト条件になったつもりで

自分がif文のテスト条件になったつもりで、以下の変数が
与えられたときtrueかfalseかを決定して下さい。

```
$my_name = '横山';
$a_number = 3;
$a_decimal = 4.6;
$favorite_song = 'Trouble';
$another_number = 0;
$your_name = $my_name;
```

<code>(\$a_number == 3)</code>	trueまたはfalse
<code>(\$another_number == "")</code>	trueまたはfalse
<code>(\$favorite_song == "Trouble")</code>	trueまたはfalse
<code>(\$my_name == '\$your_name')</code>	trueまたはfalse
<code>(\$my_name == "\$your_name")</code>	trueまたはfalse
<code>(\$your_name == \$my_name)</code>	trueまたはfalse
<code>(\$favorite_song == 'Trouble')</code>	trueまたはfalse
<code>(\$a_number > 9)</code>	trueまたはfalse
<code>(\$favorite_food = 'hamburger')</code>	trueまたはfalse



if 文のテスト条件になったつもりでの答え

自分が if 文のテスト条件になったつもりで、以下の変数が与えられたとき true か false かを決定して下さい。

```
$my_name = '横山';  
$a_number = 3;  
$a_decimal = 4.6;  
$favorite_song = 'Trouble';  
$another_number = 0;  
$your_name = $my_name;
```

`($a_number == 3)`

true または false

0と空文字列は等しいという評価結果となります。

`($another_number == "")`

true または false

これは単一引用符ので Buster という文字列と "\$your_name" という文字列とを比較することになります。変数 \$your_name が持っている値との比較ではありません。

`($favorite_song == "Trouble")`

true または false

`($my_name == '$your_name')`

true または false

`($my_name == "$your_name")`

true または false

`($your_name == $my_name)`

true または false

`($favorite_song == 'Trouble')`

true または false

`($a_number > 9)`

true または false

\$a_number は 3 ですから 9 よりも本当に大きくはありません。

`($favorite_food == 'hamburger')`

true または false

そうなんですね。これは引っかけ問題です。ここでは等号が1つしか使われていなかったので、これは実際には代入文(=)なのです。比較(==)ではありません。これが何故 true になるかというと、0 でも NULL でも false でもないものは、PHP では何でも true と解釈されてしまうからです。

もし比較をしたかった
のなら == を使うべき
でした。

素朴な疑問に答えます

Q: 分かってきました。もしかしてテスト条件というのは3章で出てきた while ループを制御するときに使ったものと同じですか？

A: その通り。全く同じものです。ただ3章では問い合わせデータに残っている行があるかどうかを調べるために使いましたが、もっと別の比較を使って while ループを面白くしてくれるテスト条件に出会うことになります。本書で後ほど説明します。

メール送信用画面の妥当性検証の背後にある論理

清野君は sendemail.html フォームから受け取るデータの妥当性を検証してからでないと、メールを送ってはいけません。実際、メール送信というのは、データの妥当性検証の結果に完全に依存すべきものなのです。PHP で本来やるべきだったことは、sendemail.php スクリプトで受け取ったフォームデータの妥当性を検証し、その上で決定を下すということです。必要なコードは「もしデータが妥当であれば、先に進みメールを飛ばす。」というものです。

しかし、まずフォームのデータを取ってきて2つの変数に格納する必要があります。

```
$subject = $_POST['subject'];
$text = $_POST['elvismail'];
```

このフォームデータがチェックすべきすべてのもので、各フォームフィールドにデータが入っているかどうかを確認します。論理的にはこのような感じになります。

**IF \$subject がテキストを含んでいるかつ\$body がテキストを含んでいる
THEN メールを飛ばす**

または、反対のアプローチをとって、フォームフィールドが両方とも空[†]かどうかをチェックし、その場合ユーザに警告メッセージを出す、というやり方もできたかもしれません。

**IF \$subject が空であるかつ\$body が空である
THEN エラーメッセージをechoする**

どちらの例も1つのif文で2つの比較をするという論理的な要求に応える必要があつて問題です。2つのif文を使うというのも答えにはなっていますが…



† 訳注：原著のこの記述は論理的に正しくありません。「両方ともテキストを含んでいる」の逆は、「両方とも空」ではなく、「どちらか一方が空」です。すぐ下の式の記述で「かつ」となっている部分を「または」に変更すれば、正しくなります。ただし、話の流れとは関係ないので、あえて原著のままとしました。

自分で考えてみよう

2つのif文を使って、清野君のメール送信用フォームの件名とメッセージ本文の両方とも空かどうかをチェックするPHPの文を書いてみましょう。両方が空の場合、警告メッセージをechoします。

自分で考えてみよう の答え

2つのif文を使って、清野君のメール送信用フォームの件名とメッセージ本文の両方とも空かどうかをチェックするPHPの文を書いてみましょう。両方が空の場合、警告メッセージをechoします。

1番目のif文の内側にネストした2番目のif文により、両方の条件がtrueの場合のみecho文が実行されるというコードになっています。

```
if ($subject == '') { ← 2つ連続した单一引用符で空の
    文字列を表しています。
    → if ($text == '') {
        echo 'メールの件名と本文を記入して下さい。<br />';
    }
}

インデントをつけることで内側のif文の
終わりがどこか、外側のif文の終わりが
どこかが分かりやすくなります。
```

PHP 関数で変数の妥当性を検証する

==を使って空文字列かどうかチェックするというのは、ちゃんと動きますが、PHPの組み込み関数を使った、ちょっと良いやり方もあります。関数 iset() は、変数が存在するかどうかをテストします。これは変数に値が代入されたかどうかをテストするという意味です。関数 empty() は、もう1ステップ先を見て、変数が空の値であるかどうかを判定します。ここでPHPの空の値とは、0、空文字列("または")、false または NULL と定義されています。つまり iset() は変数に値が代入されている場合のみに true が返りますが、empty() は変数の値が0、空文字列、false または NULL の場合に true が返ります。

2つの関数がどのように動くのか見てみましょう。

```
$v1 には値が入っています。
$v1 = 'aloha';
} } $v1 と $v2 の両方とも値がセットされていると考えます。ただし
$v1 にのみ値があります。
$v2 は空文字列です。
} } 網掛けのechoコードだけが実行されます！

$v2 には、空文字列
ではあります
がセットされて
います。
} } $v1 は空ではありません。
テキストが入っています
ので、この条件は false と
なります。
} } $v2 は、そこに入っ
て
いる文字列が空なので
空となります。
} } $v3 は存在しません
が
空とみなされます。

$V3 は存在しません。
```



わかったわ。\$subjectとか\$textの
フォームデータの妥当性を検証するには
isset()やempty()を使えばいいのね。

半分あっています。今チェックしているのはフォームデータが空でないことを確かめることなので、empty()だけで十分です。

変数\$subjectや\$textには、スーパーグローバル\$_POST['subject']と\$_POST['elvismail']とを使って値を代入しました。isset()を使って変数のテストをしても、常にtrueが返ってきます。変数に本当にテキストが入っているかどうかには関係ありません。別の言い方をすると、isset()では、フォームフィールドが空か、それとも何か埋めてあるかの違いを区別することができません。関数empty()は、変数が本当に空かどうかをチェックしてくれます。これこそフォームの妥当性検証に必要なものです。

isset()は、変数が存在し、
値がセットされているかどうかを
チェックします。

empty()は、変数の内容が
何かあるかをチェックします。

素朴な疑問に答えます

Q：じゃあisset()っていつ使うの？

A：isset()関数はデータがそもそも存在するかどうかを知る必要がある場合に、特に威力を発揮します。例えば、POSTリクエストによりフォームが「提出」されたかどうかをチェックする場合、isset()関数に\$_POSTを食わせれば分かります。これは極めて役立つテクニックなのです。本章で後ほど出でます。

自分で考えてみよう

清野君のメール送信用フォームの件名とメッセージ本文の両方が空かどうかをチェックする2つのif文を書き直して下さい。今度は==の代わりにempty()関数を使います。

自分で考えてみよう の答え

清野君のメール送信用フォームの件名とメッセージ本文の両方が空かどうかをチェックする2つのif文[†]を書き直して下さい。今度は==の代わりにempty()関数を使います。

```

if のそれぞれのテスト
条件のところで、
等値演算(==)が
empty()関数の呼び
出しに置き換えられて
います。
→ if (empty($subject)) {
→ if (empty($text)) {
    echo 'メールの件名と本文を記入して下さい。<br />';
}
}
↑ 残りのコードは前回と
一緒にあります。

```

[†] 訳注：このような書き方だと条件が増えるたびにif文がネストしてコードが読みにくくなります。このネストを解消してくれるものが次のページ以降の話です。

今やりたいことはフォームフィールドが
空じゃないときに、何らかのアクションを
とりたいってことでしょう。
notempty()関数ってないの？



ありません。でもテスト条件の論理値を反転させる簡単な方法があります…
否定演算です。

if文をコントロールするテスト条件が、結果として常にtrueかfalseとなることはすでに知っていますね。でも条件として与えられるものを反転してチェックする必要があるという論理値が活躍するにはどうすればよいでしょう？例えば、清野君のフォームフィールドが空でないかどうかを、そのフォームのデータを使ってたくさんのメールを飛ばしてしまう前に、分かっていれば助かります。ところが問題なのは、notempty()関数がないことです。この問題を解決してくれるのが否定演算(!)です。trueをfalseに、falseをtrueにひっくり返してくれます。つまり !empty()とすれば、empty()関数を呼び出し、結果を反転してくれます。こんな感じです。

否定演算(!)は、trueを
falseに、falseをtrueに
ひっくり返します。

```

if (!empty($subject)) {
...
}
```

この条件は「件名フィールドは空で
ないか？」つまりフィールドにデータ
があるか？を確かめています。

複数の条件を論理積と論理和でテストする

複数のチェックを論理演算子でつなげて if 文のテスト条件を構築することができます。この演算子がどのように働くのかを、お馴染みの2つの条件、`!empty($subject)` と `!empty($text)` を使って見てみましょう。最初の例では、2つの式を論理積で結合しています。論理積は `&&` とコーディングします。

```
論理積演算子
if (( !empty($subject) ) && (!empty($text)) ) {  
    このテスト条件は、$subjectと$textの  
    両方が空でない場合に限り true となります。
}
このカッコは余分ですが、否定演算子が  
empty() 関数だけに作用していることを  
明確に示してくれます。
```

論理積演算子は2つの `true/false` 値をとり、両方とも `true` の場合にのみ `true` を返します。そうでなければ結果は `false` です。従ってこの場合、両方のフォームフィールドがともに空以外でなければなりません。この条件を満たせば、テスト条件は `true` となり、if 文の動作部のコードが実行されます。

論理和演算子は、`||` とコーディングしますが、論理積と同様です。異なる点は、どちらか一方の `true/false` 値が `true` となった場合に `true` を返すことです。例を示します。

```
if (( !empty($subject) ) || (!empty($text)) ) {  
    このテスト条件は、$subjectか$textのいずれか  
    一方が空でない場合に true となります。
}
```

つまりこの if 文の動作部のコードが実行されるのは、どちらか一方のフォームフィールドが、空でない場合です。話をもう少し面白くするには、片方のフォームフィールドは空のままでし、もう一方はデータを持たなければならないというような場合で、こんな感じになります。

```
if (empty($subject) && (!empty($text)) ) {  
    $subjectが空で、かつ$textが  
    空でない場合に限り、このテスト  
    条件はtrueとなります。
```

このテスト条件では論理積を使ったので、テスト条件の内側にある両方の式が `true` とならなければ、動作部のコードを実行することはできません。これは件名のフォームフィールドは空でなければならず、しかし本文フィールドはデータがなければならないことを意味します。否定演算子 (`!`) を反対の `empty()` 関数に移動して、このチェックを反対にすることもできます。

```
if (( !empty($subject) ) && empty($text)) {  
    これがtrueとなるのは、$subjectが  
    空でなく、$textが空の場合です。
}
```

論理積 (`&&`) と論理和 (`||`) の2つの論理演算子を使うと極めて強力なテスト条件を構築することができます。もしかったとすると、たいていは見るに堪えないほど if 文が増えしていくことでしょう。

PHPの論理

演算子は if 文を、
よりエレガントに
構築してくれます。

論理積は `&&` と

コーディングし、

論理和は `||` と

コーディングします。

これは数字の11ではありません。
2つの縦棒(パイプ)です。
キーボードでは、円記号(¥)の
上にあります。



`sendemail.php`スクリプトの網掛け部分を書き直して下さい。ネストしたif文の代わりに1つのifテスト条件の中で論理演算子を使います。

```
<?php
$from = 'seino@makemeelvis.com';
$subject = $_POST['subject'];
$text = $_POST['elvismail'];

if (!empty($subject)) {
    if (!empty($text)) { ← これがネストしたif文です。1つの
                            if文の中に論理演算子を使って
                            書き直して下さい。
    .....  

    .....
```

.....

```
$dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store');
or die(' エラー：MySQL との接続に失敗しました。');

$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query)
or die(' エラー：データベースの問い合わせに失敗しました。');

while ($row = mysqli_fetch_array($result)){
    $to = $row['email'];
    $last_name = $row['last_name'];
    $first_name = $row['first_name'];
    $msg = "$last_name $first_name さん,\n$text";
    mb_internal_encoding("UTF-8");
    mb_send_mail($to, $subject, $msg, 'From:' . $from);
    echo 'メールを' . $to . 'へ送信しました。<br />';
}

mysqli_close($dbc); ← この波カッコで2つの
}                                if文を閉じています。
}.....
```



試運転

メール送信用スクリプトの論理演算子がネストしたif文と同じ働きをすることを確認します。

sendemail.phpのコードを修正し、1つのif文で論理演算子の恩恵に浴しながら、メールメッセージを飛ばす前にフォームフィールドのデータをチェックするようにします。もしどのような変更をしたらいいのかよく分からぬ場合は、次のページにあるエクササイズの答えをよくチェックして下さい。

新しいバージョンのスクリプトをWebサーバにアップロードしたら、sendmail.htmlページをWebブラウザで開きます。どちらか一方のフォームフィールドを空のままにして、「送信」ボタンをクリックします。スクリプトは、フォームフィールドが空の場合、相変わらずメールメッセージが飛ばないようブロックしてくれますか？

素朴な疑問に答えます

Q: if文にある&&や||で結ばれた2つの条件の順番が問題になることはありますか？

A: あります。これら2つの演算子は可能であれば短絡的に処理をします。これは、仮に第1オペランドだけで式の結果を決定するのに十分な情報が得られた場合、第2オペランドは無視されるという意味です。一例として、論理積演算の第1オペランドがfalseだったとします。そうすると第2オペランドに関係なく、全体の式がfalseになるには十分な情報が得られたことになります。この場合第2オペランドは無視されます。論理和演算子の第1オペランドがtrueの場合も同じ規則が当てはまります。

Q: &&や||の代わりに、andやorを使っているPHPコードを見たことがあるのですが、これはどのように動くのでしょうか？

A: これらは実質的には&&や||と同じものです。他の演算子との兼ね合いで、どのように実行されるかが微妙に違います。しかしカッコを注意深く使い、テスト条件をはっきりさせれば、本質的な差はありません。



エクササイズ の答え

```
<?php
$from = 'seino@makemeelvis.com';
$subject = $_POST['subject'];
$text = $_POST['elvismail'];

if (!empty($subject)) {
    if (!empty($text)) {
        if ((!empty($subject)) && (!empty($text))) { ← 空でないフォームフィールドをチェック
            ← するため否定演算子 (!) を使います。
            ← 論理積を使うことにより2つの条件でも1つの
            if文でチェックすることができます。
            ← 論理積演算子として実際に指定するのは
            && であることを忘れないで下さい。
            $dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')
            or die(' エラー：MySQLとの接続に失敗しました。');

            $query = "SELECT * FROM email_list";
            $result = mysqli_query($dbc, $query)
            or die(' エラー：データベースの問い合わせに失敗しました。');

            while ($row = mysqli_fetch_array($result)){
                $to = $row['email'];
                $last_name = $row['last_name'];
                $first_name = $row['first_name'];
                $msg = "$last_name $first_name さん、\n$text";
                mb_internal_encoding("UTF-8");
                mb_send_mail($to, $subject, $msg, 'From:' . $from);
                echo 'メールを' . $to . 'へ送信しました。<br />';
            }

            mysqli_close($dbc);
        } ← 1つのif文を閉じるので、1つの
        ← 波カッコだけが必要です。
    }
}
?>
```

フォームのユーザにフィードバックが必要です

`sendemail.php`コードは、フォームデータの妥当性を検証するという偉大な業績を上げているため、件名または本文フィールドのいずれか一方でも空白のままだとメールは飛ばないようになりました。ところが妥当性検証が通らなかった場合、メールが飛ばないのは良いのですが、スクリプトは清野君に何が起こったのかを教えてくれません。出てくるのは空白のWebページだけです。

フォームを「提出」すると、このページが出てきます…
何故なのかさっぱり分かりません。



問題はコードが妥当性検証に成功した場合にのみ反応するために起こっています。この場合はちゃんとメールメッセージを飛ばしてくれます。でも `if` 文が `false` (フォームデータが不当) 側に倒れた場合、コードは何もしてくれません。清野君はメールが飛んだのか、それとも何かが間違っているのか何も分からずじまいです。スクリプトコードを部分的に再掲します。空白のページに関する問題を解き明かしましょう。

```
<?php
$from = 'seino@makemeelvis.com';
$subject = $_POST['subject'];
$text = $_POST['elvismail'];

if ((!empty($subject)) && (!empty($text))) {
    $dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')
    ...
    mysqli_close($dbc);
}
?>
```

何も起こらないのは、`if` 文が `false` となって動作部のコードを実行しないためです。このためフォームデータが不足しているときは空白のページが生成されます。



清野君に問題があったことを教えてあげましょう。フォームフィールドが空白だったので、もう一度メッセージを入力できるようにしてあげられれば理想的です。



簡単じゃない。if文を閉じるカッコの後ろにecho文を置いとけば良いでしょ。

それでは動きません。if文の後ろにあるコードは常に実行されてしまうからです。

if文の直後にecho文を置いておくということは、if文の後にその文が実行されてしまうという意味です。ifの結果と関係なく常に実行されてしまいます。これでは幸せはやってきません。今欲しいのはif文のテスト条件がfalseになった場合にだけエラーメッセージを表示してくれるecho文です。論理的にはこんな感じで表現できます。

- ✓ IF** 件名にテキストが含まれる **かつ** 本文にテキストが含まれる
- THEN** メールを飛ばす
- ✗ ELSE** エラーメッセージをechoする

if文にはelse節というオプションが備わっていて、テスト条件がfalseとなった場合にだけ、その部分を実行します。そこでechoコードによるエラーメッセージをelse節にぶち込んでおけば、フォームフィールドが一方でも空白のままの場合に実行されることになります。単にif文の後ろに、elseという予約語を書き、その後に波カッコで囲まれた動作部コードを貼つければ良いのです。

```
if (( !empty($subject) ) && ( !empty($text) )) {
    ...
    } // else節はif文の最後にある閉じ波カッコの直後から始まります。
}
else {
    echo 'メールの件名と本文を記入して下さい。<br />';
}
} // ifの動作部コードと全く同様に、elseのコードも波カッコで囲みます。
```

この部分は省略していますが、メールメッセージを飛ばすコードです。

if文がfalseに倒れた場合のみ、この部分のコードが実行されます。

else節はifのテスト条件がfalseの場合にコードを実行します。



こんなにネストしたifやelseがあったのでは
読みにくくてスクリプトをフォローできません。
そんなスクリプトと生涯を共にするなんて
考えたくもありません！誰かが怪我をする前に
もっとシンプルにするべきです。

可能な限りコードをシンプルにするというのは、常に
よいことです。特にあまり深くまでネストしてしまった
コードの場合はなおさらです。

if文の中にネストしたelse文がたくさんあると、コードが
フォローできなくなってしまいます。もしも二度とそのコード
を見ることがないのであれば、それでも問題ないかもしれません。
でもそんなことあり得ません。例えばもしフォームを変更
して新しいフィールドを追加する必要に迫られたら、妥当性を
検証するのは今までより更に複雑化していくことでしょう。
なぜならコードが読みにくいためにどこをどう変えればいいのか
分かり難くなってしまうからです。



IFコードを変更して見るに堪えないネストした
IFやELSEをきれいにします。コードを書き
直してネストをなくして下さい。ただしもち
ろん、コードは今まで通り正しく動かさなく
てはいけません。

ヒント：elseなしでも
できるかもしれません！

```
if (empty($subject) && empty($text)) {  
    echo 'メールの件名と本文を記入して下さい。<br />';  
} else {  
    if (empty($subject) || empty($text)) {  
        if (empty($subject)) {  
            echo 'メールの件名を記入して下さい。<br />';  
        } else {  
            echo 'メールの本文を記入して下さい。<br />';  
        }  
    } else {  
        // すべて記入されているのでメールを送ります。  
    }  
}
```

ネストがなくなるようにコード
を書き直します。



試運転

きれいにした if コードが期待通り動くことを確認します。

sendemail.php のコードを修正し、今作ったように if をネストをなくしたシンプルなものを使います。もしどのような変更をしたらいいのかよく分からない場合は、ページをめくって答えを参照して下さい。

新しいバージョンのスクリプトを Web サーバにアップロードしたら、sendmail.html ページを Web ブラウザで開きます。スクリプトに対して、2つのフォームフィールドを空白にしたり埋めたりして「提出」の実験をします。スクリプトはエラーメッセージを表示してくれますか？

素朴な疑問に答えます



多少のレベルのネストでも本当に大問題なのでしょうか？



A: 状況によります。書いているコードを今後自分以外見る予定もなく、どの行で何をやっているか半年たってもきちんと覚えていて、簡単に調整することもできるのであれば、ネストを放っておいてもよいでしょう。

でも、コードができる限りきれいかつ論理的にしたいのであれば、今までに見てきた論理演算子を使うという選択肢があります。



elseはどう働くのですか？



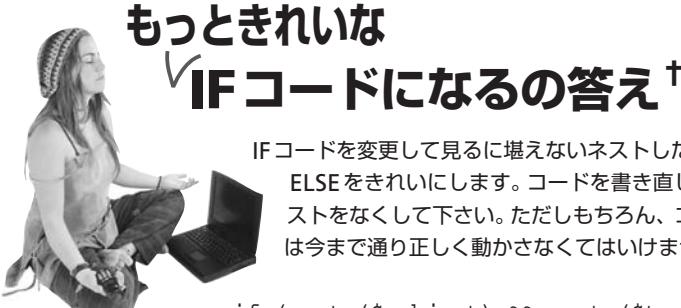
A: if...else 文で、else は、if でマッチしなかったすべてのものにマッチします。



ええと、一応分かりましたけど、それでは既存の if...else 文に if や else を更にネストさせても良いと言うことですか？



A: YES ではありますが、そのようなネストをさせると話はものすごい速さで複雑になってしまないので、ここでネストを避けようとしているのです！



もっときれいな IFコードになるの答え[†]

IFコードを変更して見るに堪えないネストしたIFやELSEをきれいにします。コードを書き直してネストをなくして下さい。ただしもちろん、コードは今まで通り正しく動かさなくてはいけません。

```

if (empty($subject) && empty($text)) {
    echo 'メールの件名と本文を記入して下さい。<br />';
} else {
    if (empty($subject) || empty($text)) {
        if (empty($subject)) {
            echo 'メールの件名を記入して下さい。<br />';
        } else {
            echo 'メールの本文を記入して下さい。<br />';
        }
    } else {
        // すべて記入されているのでメールを送ります。
    }
}

```

ここで変数\$subjectと\$textの両方が空かどうかをテストしています。

ここで\$textは空で、\$subjectは空でないかをテストしています。

ここで\$subjectも\$textもどちらも空でないことをテストしています。

if (empty(\$subject) && empty(\$text)) {
 echo 'メールの件名と本文を記入して下さい。
';
}

if (empty(\$subject) && (!empty(\$text))) {
 echo 'メールの件名を記入して下さい。
';
}

if (!empty(\$subject)) && empty(\$text)) {
 echo 'メールの本文を記入して下さい。
';
}

if (!empty(\$subject) && (!empty(\$text))) {
 // すべて記入されているのでメールを送ります。
}

このコードは\$subjectは空で\$textは空でないかをチェックしています。

もし論理積(&&)を使って、「空でない件名」かつ「空の本文」をより分けなかったら、すべてのフィードバックを尽くすことはできないでしょう。同じ話は「空でない件名」と「空の本文」でも言えることです。[‡]

否定演算(!)で\$subjectと\$textが空でないことをチェックします。

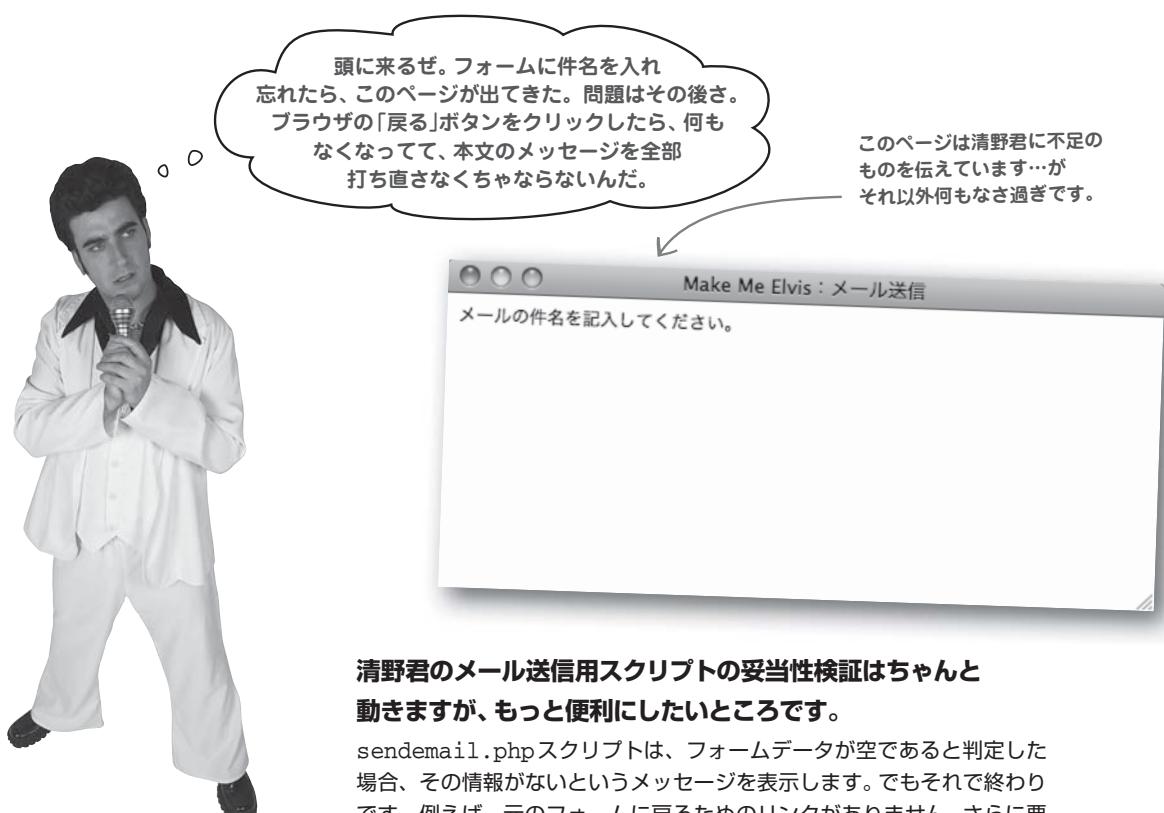
[†] 訳注：ここで原著者が示している「正解」は、コーディングスタイル上あまり好ましくありません。すべてをelseのないif文にしていますが、これではif文が排他的(どれか1つしか実行されない)であることを構文上明示していないためです。回答と同じことを書くとしても

```

if(empty(A) && empty(B))...
else
if(empty(A) && !empty(B))...
else
if(!empty(A) && empty(B))...
else...

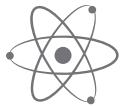
```

というスタイルを推奨します。こちらの書き方であれば、構文上どれか1つの部分だけが必ず実行される明示できています。



清野君のメール送信用スクリプトの妥当性検証はちゃんと 動きますが、もっと便利にしたいところです。

`sendemail.php`スクリプトは、フォームデータが空であると判定した場合、その情報がないというメッセージを表示します。でもそれで終わりです。例えば、元のフォームに戻るためのリンクがありません。さらに悪いことに、清野君が元のフォームの戻って来たときに、以前に入力したはずの情報はもうないのです。改めてメールメッセージの件名と本文の両方を打ち込まなければなりません。



脳力発揮

メール送信用スクリプトのエラー処理を改良して、もっと便利にするにはどうすればよいと思いますか？



エラーメッセージと一緒にフォームを表示できればイケてるんじゃない。メールの件名と本文が空だったらフォームを単にechoするなんてできないのかしら？

フォームを表示するというのは確かに便利になります。だって清野君はもうブラウザで戻る必要はなくなるのです。

そこで、いずれか一方のフォームが空だった場合、エラーメッセージを表示することに加えて、PHPでブラウザに対してechoすることで、HTMLフォームのコードを再生成する必要があります。以下のコードはPHPがかなり複雑なHTMLコードを生成できる能力があることを表しています。

この PHP コードは、HTML フォーム全体を生成しますので、<form> タグで始めます。

```
echo '<form method="post" action="sendemail.php">';  
echo ' <label for="subject">メールの件名:</label><br />';  
echo ' <input id="subject" name="subject" type="text" '' .  
      'size="30" /><br />';  
echo ' <label for="elvismail">メール本文:</label><br />';  
echo ' <textarea id="elvismail" name="elvismail" rows="8" '' .  
      'cols="40"></textarea><br />';  
echo '<input type="submit" name="submit" value="送信" />';  
echo '</form>';
```

厳密にはインデントは必要ではありませんが、元の HTML コードの構造を見やすくしてくれます。

HTML のコードは二重引用符だけなので、PHP で HTML コードの文字列を囲むには單一引用符を使う方が簡単です。

もし、このコードがちょっと混沌としていると感じるようでしたら、それは本当にその通りだからです。PHPで何かできるということは、そうすべきだという意味ではありません。この場合、HTML コードを全部echoしたことによって複雑さが増したというのが問題です。これでも PHP で echo を用いて吐くには十分すぎるほど大きなコード列です。この場合 echo は良いやり方とは言えません…

必要に応じてPHPから出たり入ったりする

忘れがちなことなのですが、PHPスクリプトは、単にHTMLのWebページなのです。PHPコードを保持することができるという点のみが異なるのです。PHPスクリプトの中で<?phpと?>タグとで囲まれていない部分のコードはHTMLとして取り扱われます。つまりPHPコードのブロックを閉じれば、必要に応じてHTMLに戻すことができ、新しいPHPコードのブロックを再開することもできます。これはHTMLコードを吐き出す極めてお手軽なテクニックです。先ほどのemail送信用フォームのようにPHPのecho文で生成しようとすると重たすぎる場合に特に有効です。

```
<?php
$from = 'seino@makemeelvis.com';
$subject = $_POST['subject'];
$text = $_POST['elvismail'];

この?>タグで
PHPブロックを
閉じて、HTMLの
世界へ戻ります。
?>

if (empty($subject) && empty($text)) { $subject も $text も空白の場合
    // We know both $subject AND $text are blank
    echo 'メールの件名と本文を記入してください。  
';
```

```
<form method="post" action="sendemail.php">
    <label for="subject">メールの件名:</label><br />
    <input id="subject" name="subject" type="text" size="30" /><br />
    <label for="elvismail">メール本文:</label><br />
    <textarea id="elvismail" name="elvismail" rows="8" cols="40"></textarea><br />
    <input type="submit" name="submit" value="送信" />
</form>
```

```
<?php タグは、PHP ブロックを開始します。
今はまだifの動作部の中にいるので、if文を終
えてから続きを書きます。

<?php
}

if (empty($subject) && (!empty($text))) {
    echo 'メールの件名を記入して下さい。  
';
```

```
if ((!empty($subject)) && empty($text)) {
    echo 'メールの本文を記入して下さい。  
';
```

```
if ((!empty($subject)) && (!empty($text))) {
    // メールを送信するコード。
    ...
}

?>
```

このコードが制限しているようなことがあるとしたら何だと思いますか?
書き留めておきましょう。どうすれば修正できますか?

PHPコードのブロック
を閉じたり開けたりする
ことで、PHPスクリプト
の中でHTMLコードを
吐くことができます。

フォームは通常のHTMLとして
コーディングします。このコードは
PHPタグの外側にあるからです。

フラグを用いて重複の重複コードをなくす

前ページのコードには、PHPから外に出たらフォームコードを別々の3箇所（妥当性検証エラー1箇所に1つずつ）に重複して書かなければならぬという問題を抱えています。true/false値をフラグとして使えば、フォームを吐く必要があるかどうかを変数に覚えさせることができます。その変数名を\$output_formとしましょう。そうすればその変数を後のコードでチェックし、その変数がtrueだったらフォームを表示することができます。

そこでまず、スクリプト上で\$output_formをfalseにセットするところから始め、フォームフィールドが空でフォームを再表示する必要がある場合にのみtrueに変更することにします。

\$output_formをfalseで初期化する

初期値として\$output_formをfalseにセットすることで、フォームは表示されないことを意味します。妥当性検証で問題が発見されると値が変更され表示されることになります。

IF 件名が空かつ本文が空

これらのエラーメッセージは微妙に異なります。
どの特定のフォームフィールドが空かを明示するためです。

✓ THEN エラーメッセージをechoし、\$output_formをtrueに設定する

IF 件名が空かつ本文が空でない

✓ THEN エラーメッセージをechoし、\$output_formをtrueに設定する

IF 件名が空でないかつ本文が空

✓ THEN エラーメッセージをechoし、\$output_formをtrueに設定する

IF 件名が空でないかつ本文が空でない

両方のフォームフィールドがチェックに成功しデータが入って

いると分かったら、先に進みメールを飛ばします。

✓ THEN emailを飛ばす

✓ THEN \$output_formがtrue

最後に、\$output_form変数をチェックし、フォームを表示する必要があるかどうかを判定します。いずれの場合もHTMLコードは1つだけです。

HTMLフォームは一度だけコーディングする

新しい妥当性検証の方式をPHPコードにぶち込むため、新しい変数\$output_formを作つて初期化することになりました。そして妥当性を検証するコードを通して、その変数に値を設定します。最も重要な部分は、コードの最後にある新しいif文で、\$output_formがtrueにセットされている場合に限りフォームを表示します。

```
<?php
    if (isset($_POST['submit'])) {
        $from = 'seino@makemeelvis.com';
        $subject = $_POST['subject'];
        $text = $_POST['elvismail'];
        $output_form = false; ← ここで新しい変数を作り
                                初期値をfalseにセット
                                します。
    }

    if (empty($subject) && empty($text)) {
        // $subjectも$textも空白の場合
        echo 'メールの件名と本文を記入して下さい。<br />';
        $output_form = true; ← $subjectと$textの両方が
                                空だった場合、変数をtrueにセットし
                                フォームを表示できるようにします。
    }

    if (empty($subject) && (!empty($text))) {
        echo 'メールの件名を記入して下さい。<br />';
        $output_form = true; ← $subjectが空なので、この場合も
                                変数をtrueにセットします。
    }

    if ((!empty($subject)) && empty($text)) {
        echo 'メールの本文を記入して下さい。<br />';
        $output_form = true; ← さらに$textが空の場合も、変数を
                                trueにセットします。
    }
}
else {
    $output_form = true;
}
if ((!empty($subject)) && (!empty($text))) {
    // メールを送信するためのコード。
    ...
}

if ($output_form) { ← このif文で$output_form変数を
                        チェックしtrueの場合にフォーム
                        を表示します。
    ?>
    <form method="post" action="sendemail.php">
        <label for="subject">メールの件名:</label><br />
        <input id="subject" name="subject" type="text" size="30" /><br />
        <label for="elvismail">メール本文:</label><br />
        <textarea id="elvismail" name="elvismail" rows="8" cols="40"></textarea><br />
        <input type="submit" name="submit" value="送信" />
    </form>
    <?php
    } ← 忘れずにPHPコードに戻つて、if文を閉じて
        表示すべきすべての箇所で情報を1つの変数$output_formに
        押し込むことができたからです。
    ?>
```

HTMLコードを作るかどうかは
1つのIF文だけに依存しています。
スクリプト内に重複するコードはありません。

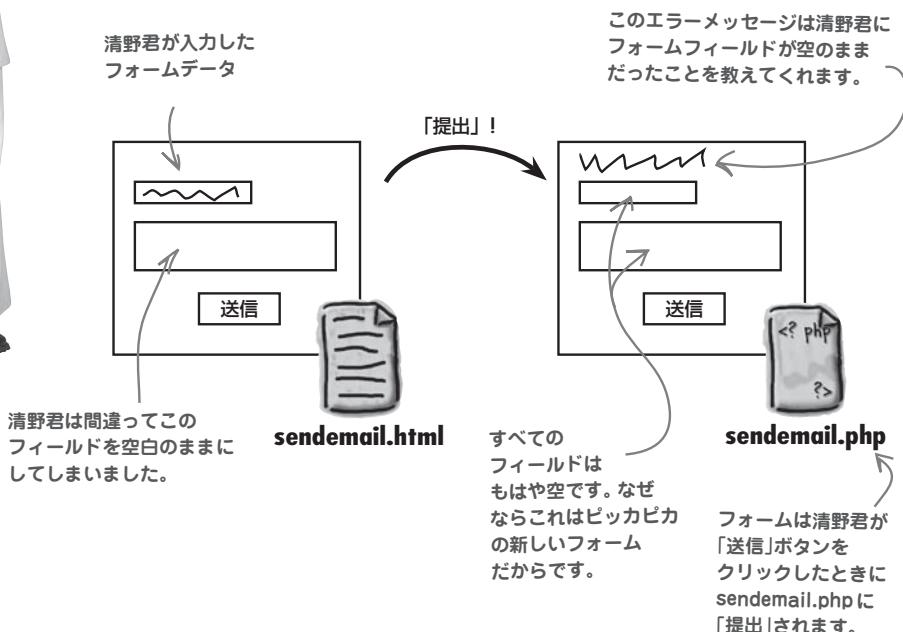
この部分はPHPから抜け出しています。
それでも閉じる}までは何でも、相変わらず
ifの動作部として扱われます。この場合で
言うと、フォームのHTMLコードもifの
動作部とみなされます。



新しいフォームは
良くなったけど、相変わらず、さっき
正しく打ったフィールドを再入力しなきゃ
ならない。喧嘩売ってんのか！

HTMLだけではフォームデータを保存できません

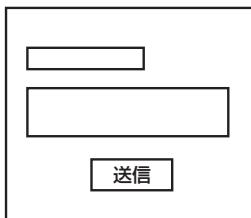
清野君がemail送信用フォームに1つだけ空のフィールドを残して「提出」ボタンを押した場合、sendemail.phpスクリプトはエラーを検出し、新しいフォームを作ります。しかし、新しいフォームは純粋なHTMLコードですから、清野君が以前入力したデータについては何も覚えていません。このため妥当性検証の結果としてまっさらなフォームを作ったのです。清野君が打ち込んだデータはきれいさっぱり消し去っていました。



認めざるを得ません。PHPスクリプトで新しいフォームを作るしかないのですが、それだけではダメなのです。清野君がすでに入力したデータを覚えておいて、新しいフォームにぶち込む方法が必要なのです。そうすれば清野君は間違って空のままにしてしまったフォームフィールドを埋めることだけに集中できます…

自分で考えてみよう

清野君が最初の1つのフォームフィールドだけを埋めて「提出」した後、フォームがどうあるべきか描いてみましょう。次に2つのファイル(HTMLとPHP)のそれぞれをどのように変更すれば、この新しい機能を実現できると思うか書き留めておきましょう。



sendemail.php



sendemail.html

.....
.....
.....
.....
.....
.....

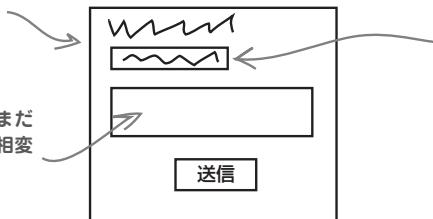
.....
.....
.....
.....
.....
.....

自分で考えてみよう の答え

清野君が最初の1つのフォームフィールドだけを埋めて「提出」した後、フォームがどうあるべきか描いてみましょう。次に2つのファイル(HTMLとPHP)のそれぞれをどのように変更すれば、この新しい機能を実現できると思うか書き留めておきましょう。

エラーメッセージは相変わらず表示されます…

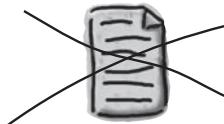
このフィールドは清野君がまだ何も打ち込んでいないので相変わらず空白のままで。



…でも、スクリプトは清野君が打ち込んだデータは覚えていてフォームに挿入しておいてくれます。



sendemail.php



sendemail.html

PHPスクリプトはフォームを表示するという仕事を担っていますが、それは「提出」の前か後に係りません。そしてスクリプトは入力された任意のフォームデータにアクセスできるため、フォームを生成するときにデータをぶち込むこともできるのです。これで清野君の問題は解決で、既に埋めたフォームデータを再入力する必要がなくなります。

フォームをすべてPHPスクリプトから表示するのであれば、HTMLページでやることはなくなります。PHPスクリプトにフォームの表示と中身の処理の両方をやらせればよいのです。PHPスクリプトは、フォームに入力された任意のデータにアクセスすることも使うことができますが、このようなことをするのは純粋なHTMLコードでは不可能です。

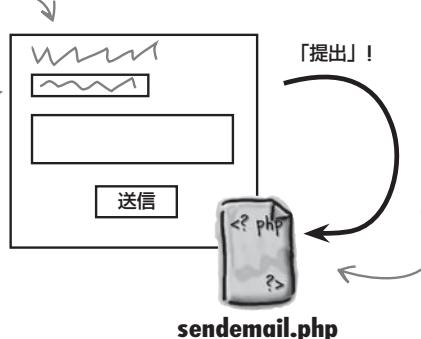
自分自身を参照するフォーム

メール送信用のフォームから sendemail.html を消去するという問題に解はあるのでしょうか？解は、HTMLコードを實際になくすのではなく、単にPHPスクリプトに移すことにより得られます。このようなことが可能なのは、PHPスクリプトが通常のWebページと同様にHTMLコードを含むことができるという事実によるのです。そこでスクリプトを改造し、「提出」されたフォームを処理するだけでなく、最初のフォームも表示するようにします。これはもともと sendemail.html がやっていたことです。

sendemail.html に残っていた役割を sendemail.php に果たさせるカギとなるのは、フォームの action です。今やスクリプト自体にHTMLフォームが書かれていますから、フォームの action は自分に戻ってくればよいことになります…自己参照フォームです。

もう sendemail.html は要りません。
ユーザは PHP スクリプトに直接
アクセスしてフォームを使います。

フォームのデータは、同じ
スクリプトに「提出」され、
このスクリプトがデータを
処理し、再びフォームを
表示します。ただ2度目
以降はすでに入力した
データを覚えています。

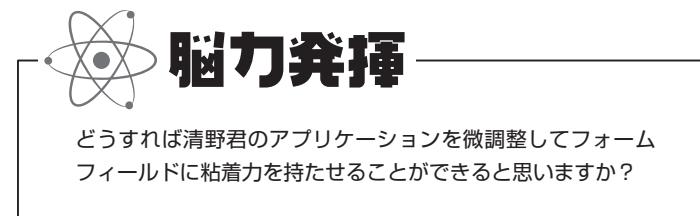


PHPスクリプトの
一部としての
HTMLフォームが、
自分自身を処理する
場合、自己参照
と言います。

スクリプトは最初はフォームを表示し、
「提出」後はフォームを処理します。
フォームの処理というのはメールの
送信か、またはエラーメッセージと
ともにフォームを再表示するかです。

ここで何が行われようとしているのかを理解するために、清野君が最初にページ（スクリプト）を参照する場合について考えてみます。空のフォームがHTMLコードとして生成され、表示されます。清野君がフォームのうち1つのフィールドのみを埋め、「提出」ボタンをクリックします。スクリプトは自分自身のフォームを処理し、もし不足のデータがあればエラーメッセージを表示します。次が重要で、スクリプトはフォームを再表示しますが、このとき清野君がすでに入力したデータがあれば、それも一緒に表示します。フォームが以前に「提出」されて入力されたデータを覚えていて十分賢いと呼ぶに値する場合、そのようなフォームを **粘着力のあるフォーム** と呼ぶことにします…データがフォームに貼付いているからです！

粘着力のあるフォームは、
ユーザがすでに正しく入力
したデータを覚えています。



スクリプトでフォームの action を指す

何度か見てきたとおり、`<form>` タグの `action` 属性が PHP スクリプトとフォームとを結びつけるもので、これによりスクリプトはフォームを処理します。清野君のフォームの `action` として `sendemail.php` と設定しても、全然問題なくちゃんと動いて、自分自身を処理してくれます。これが粘着力に向かう第1ステップです。実際フォームにはすでにスクリプトが設定された `action` 属性があります。

この図では、標準的な `<form>` タグが示されています。タグ内に `action="sendemail.php"` と `method="post"` が記載されています。手書き風の矢印とテキストで、`action` の説明がなされています。

`<form action=" sendemail.php " method="post">`

これは標準的な `<form>` タグで
フォームのデータをスクリプトに「提出」
するのにたまたま POST を使っています。

`sendemail.php` は PHP のコードだといふことです。つまり HTML コードとして出力するためには、値を `echo` しなければならないのです。こんな感じになります。

このコードは、スクリプトの名前を変えることなく、忘れずにコードをアップデートしていくという前提の下でなら、ちゃんと動きます。でももっといい方法があります。この方法なら特定のスクリプトのファイル名に依存することはないのです。PHP に組み込みのスーパーグローバル変数 `$_SERVER['PHP_SELF']` を使います。ここには現在のスクリプトの名前が格納されています。フォームの `action` に書いてあるスクリプト URL を `$_SERVER['PHP_SELF']` に置き換えれば、スクリプトの名前を変更する必要が生じても何も気にする必要はなくなります。唯一注意しなければならないのは、`$_SERVER['PHP_SELF']` は PHP のコードだということです。つまり HTML コードとして出力するためには、値を `echo` しなければならないのです。こんな感じになります。

この図では、`$_SERVER['PHP_SELF']` を使用した `<form>` タグが示されています。手書き風の矢印とテキストで、`$_SERVER['PHP_SELF']` の説明がなされています。

`<form action=" <?php echo $_SERVER['PHP_SELF']; ?> " method="post">`

スクリプトの名前のハードコーディング
してしまう代わりに、自分自身への参照は
スーパーグローバル `$_SERVER['PHP_SELF']` を
使って済ますことができます。

スクリプト名の代わりに `$_SERVER['PHP_SELF']` を使うというのは、大地を引き裂くほどの改良ではないかもしれません、何度も行われるスクリプトのメンテナンスを容易にする、小さな一歩の積み重ねのうちの1つなのです。

`$_SERVER['PHP_SELF']` を
使えば現在のスクリプト名を
格納する必要がなくなります。



試運転

フォームの妥当性検証を論理的に改良した新しい自己参照スクリプトを試してみる

sendemail.phpのコードを修正し、\$output_form変数を使って数ページ前で示したようにフォームを選択的に表示するようにします。<form>タグのaction属性も変更し、フォームが自己参照となるようにします。

sendemail.htmlページはWebサーバ上にもはや必要ありません。気兼ねなく削除して下さい。次に新しいバージョンのsendemail.phpスクリプトをWebサーバにアップロードし、スクリプトをWebブラウザで開きます。どんな風に見えますか？

何らかの理由により、
まだフォームを「提出」する
前だというのにスクリプトは
エラーメッセージを表示して
います…ダメです。

それだけではなく、このフォーム
はまだ粘着力がありません。
まだ仕事が残っているのです！

Make Me Elvis : メール送信

管理用：管理者以外使用禁止
メールを書いてメーリングリストのメンバーに送ります。

メールの件名と本文を記入してください。
メールの件名：

メール本文：

送信

最初にすべきこと。粘着力について
はしばらく置いときましょう。

スクリプトが最初にフォームを表示したときにもエラーメッセージを
表示したのは何故だと思うか書き留めておきましょう。

.....
.....
.....

フォームが「提出」されたかどうかをチェックする

問題は、フォームが最初に表示されたのか不完全なデータとともに「提出」されたのかをスクリプトには区別できないことに起因しています。このためスクリプトは、最初の最初でも不完全なデータが「提出」されたものとして、データがない旨を報告してしまうのです。困ったものです。問題は、どうすればフォームは「提出」されたものかどうかをチェックできるか、ということになります。それさえ分かれば、「提出」されたデータだけ妥当性を検証すれば済みます。

フォームがPOSTメソッドで「提出」された場合、データは\$_POST配列にどのように格納されるのかを覚えていますか？フォームがまだ「提出」されていなければ、\$_POST配列には何もデータは入っていません。別の言い方をすると、\$_POST配列はまだセットされていないのです。\$_POST配列がセットされているかどうかを確認するために、何と言う関数を呼び出せばよいか思い浮かびませんか？

isset()関数は、変数がセットされたかどうかをチェックします。

これは「提出」ボタンの<input>タグの名前と一致しなければなりません。

```
if ( isset($_POST['submit']) ) {  
    ...  
}
```

この部分のコードはフォームが「提出」された場合にだけしか実行されません。

すべてのフォームには「提出」ボタンが付いていますので、フォームが「提出」されたかどうかを簡単にチェックするには、その「提出」ボタンの\$_POSTデータがあるかどうかを見ればよいのです。データ自体は、ボタンに張り付いている単なるラベルなので、大して重要ではありません。重要なのは、\$_POST['submit']が存在するかどうかという点のみです。これによりフォームが「提出」されたかどうかが分かるのです。フォームコード上で、'submit'が「提出」ボタンのid属性とが一致しているかどうか、しっかり確認して下さい。

\$_POSTスーパーグローバルで、フォームが「提出」されたかどうかをチェックすることができます。

素朴な疑問に答えます

Q: フォームを間違って「提出」した場合にも、妥当性検証のエラーメッセージを表示しなくなってしまうことはないのでしょうか？

A: エラーメッセージが間違って表示されたのは、フォームが「提出」されたものなのか、初回の表示なのかを、スクリプトが区別できなかったためです。そこで、これは初回のフォーム表示なのかどうかを知る方法が必要でした。この場合、空のフォームフィールドというのは完璧に問題なしで、エラーではありません。妥当性を検証すべきなのは、「提出」後のフォームフィールドです。ですからフォームが「提出」されたかどうかを検出することは非常に重要なのです。

Q: じゃあ「送信」ボタンではなく、本当にフォームデータがセットされたかどうかをチェックしたのではいけないのでしょうか？

A: \$_POST['subject']や\$_POST['elvismail']をチェックしても全く問題なく動きます。でもそれはこのフォームでのチェックに限られます。すべてのフォームは、「送信」ボタンを持っていて、それは恒久的にsubmitという名前がつけられています。ですから\$_POST['submit']をチェックするというのは、すべてのスクリプトに通用する最も手堅い方法なのです。



メール送信用スクリプト準備完了

```

<?php
if (isset($_POST['submit'])) {
    $from = 'seino@makemeelvis.com';
    $subject = $_POST['subject'];
    $text = $_POST['elvismail'];
    $output_form = false;

    if (empty($subject) && empty($text)) {
        // $subject も $text も空白の場合
        echo 'メールの件名と本文を記入して下さい。<br />';
        $output_form = true;
    }

    if (empty($subject) && (!empty($text))) {
        echo 'メールの件名を記入して下さい。<br />';
        $output_form = true;
    }

    if ((!empty($subject)) && empty($text)) {
        // メール送信用のコード
        ...
    }
} else {
    $output_form = true;
}

if ($output_form) {
?>
<form method="post" action=<?php echo $_SERVER['PHP_SELF']; ?>>
    <label for="subject">メールの件名:</label><br />
    <input id="subject" name="subject" type="text" value=<?php echo $subject; ?>" size="30" /><br />
    <label for="elvismail">メール本文:</label><br />
    <textarea id="elvismail" name="elvismail" rows="8" cols="40"><?php echo $text; ?></textarea><br />
    <input type="submit" name="submit" value="送信" />
</form>

<?php
}
?>

```

\$_POST['submit'] の値をチェックします。
フォームがまだ一度も「提出」されていなければ isset() は false となります。

このカッコで最初の if を閉じます。
ここからフォームが「提出」されたことを表します。

フォームが一度も「提出」されていなければ、フォームを表示しなければなりません！



イケてる。じゃあこれで
フォームの「提出」を検出してちゃんと
エラーメッセージを表示できるわけだ。
でもまだフォームフィールドが
粘着力を持っていないでしょ？

その通りです。フォームの「提出」を検出するのは重要ですが、相変わらずフォームに
粘着力のあるフォームデータを戻して突っ込んでやる必要があります。

フォームが「提出」されたかどうかを知るというのは、粘着力を持たせるための重要な部分
ですが、それはまだ一部分に過ぎません。残りの部分は「提出」されたフォームデータを
取ってきて、フォームを表示するときにフォームの中に突っ込んでやることです。入力用の
フォームフィールドに値をセットするときは、HTMLのinputタグでvalue属性を使
います。例えば、以下のコードは入力(input)フィールドのvalue属性を使って、値をあ
らかじめセットしています。

この値はハードコーディングですから、
フォームが表示されるときは毎回
常に同じです。

```
<input name="subject" type="text" value="秋物在庫一掃！">
```

でも特定の値でハードコーディングしたいのではありません。PHP変数から取ってきた
データを挿入したいのです。どうすればよいのでしょうか？別の状況でPHPからHTMLコー
ドを動的に生成する際に、echoを使ったことを覚えてていますか？この場合もechoを使つ
て、value属性の値を、PHP変数から生成することができます。こんな感じです。

PHPに移って変数をechoするので
<?phpタグを使わなければなりません。

もうお馴染みにecho文を
使って、変数がechoされます。

```
<input name="subject" type="text" value=<?php echo $subject; ?>>
```

HTMLに戻ってくるので、
?>タグを使ってPHPを閉じます。

テキスト領域の入力フィールド
については、value属性を使う
代わりに<textarea>タグと
</textarea>タグの間に粘着力
のあるデータをechoします。

```
<form method="post" action=<?php echo $_SERVER['PHP_SELF']; ?>>  
<label for="subject">メールの件名:</label><br />  
<input id="subject" name="subject" type="text"  
value=<?php echo $subject; ?>" size="30" /><br />  
<label for="elvismail">メール本文:</label><br />  
<textarea id="elvismail" name="elvismail" rows="8" cols="40">  
?<?php echo $text; ?></textarea><br />  
<input type="submit" name="submit" value="送信" />  
</form>
```

† 訳注：すでに何度か「粘着力」という表現が出て来ていますが、原文ではstickyという単語を用いています。sticky form(粘着力のあるフォーム)などという表現で、すでに入力済みのフォームデータをそのまま残すことを表現しています。翻訳者が調べた範囲で、日本語にはこれに該当する用語は見当たりませんでした。このため「粘着力」という語を直訳的に用いて説明的に訳出しました。



試運転

清野君のデータに本当に粘着力があるかどうかチェックする

`sendemail.php`のコードを変更し、`$_POST`で「提出」チェックし、さらに`echo`コードをフォームに入れてフィールドが粘着力を持つようにします。新しいバージョンのスクリプトをWebサーバにアップロードして、Webブラウザでスクリプトを開きます。異なるフィールドの値を使い、1つまたは両方のフィールドを空にして、何度か「提出」の実験をしてみましょう。

Make Me Elvis : メール送信

管理用：管理者以外使用禁止
メールを書いてメーリングリストのメンバに送ります。

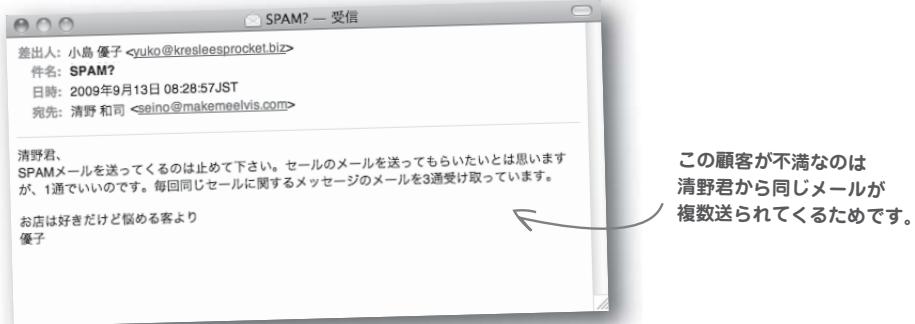
メールの本文を記入してください。
メールの件名：
秋物在庫一掃！

メール本文：

メール送信用スクリプトは、
清野君がフォームフィールドを
空白のままにしておくと、
エラーメッセージを表示しますが、
すでに打ち込んだデータは
覚えていてくれます。

まだムカついているユーザがいます

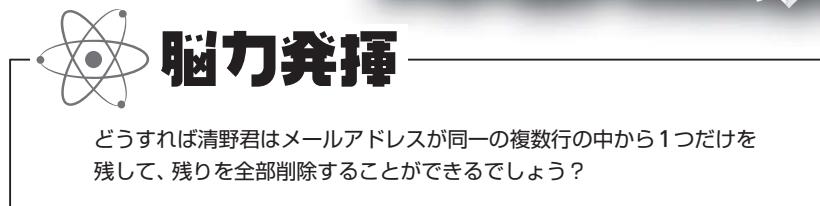
長い道のりを経てフォームの妥当性検証は、不満を持っている清野君の顧客をなだめる方向に向かっています。特に空白のメールが来ると言っていた人たちです。でもまだ幸せでない人がいます。何人かの人が同じメールを複数受け取っているようです…この人です。本章の最初の方に出てきたのを覚えていませんか？

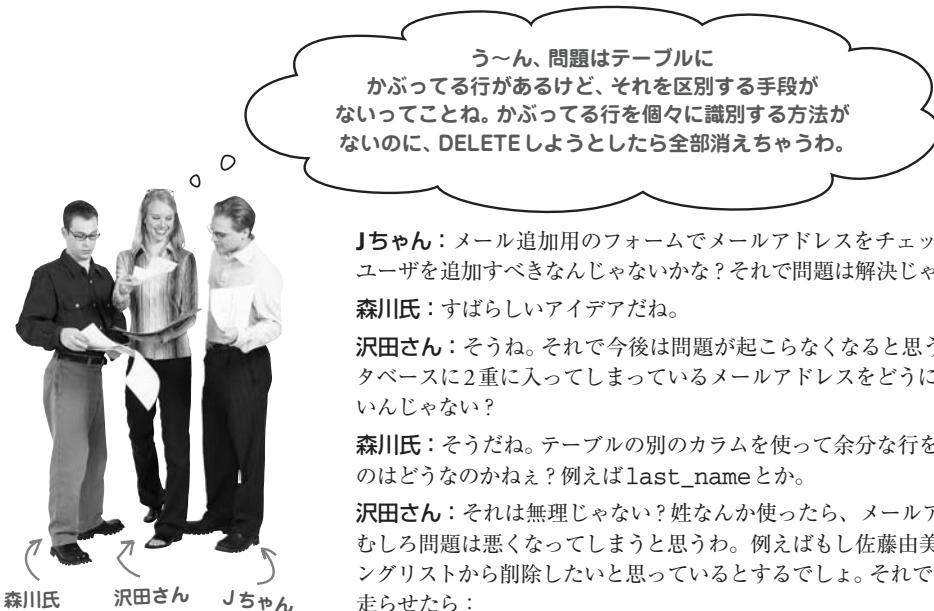


清野君は、メッセージを2度以上送っていないと知っていますから、ユーザに依存した何らかの問題があり、メールリストを2度以上登録してしまっているのではないかという疑いをもっています。だったら問題ありません。単に前章で作ったメール削除用のページとスクriプトを使って、そのユーザを削除してしまえばよいのです。これで良いと思いますか？

残念ながら、問題はそんなに単純ではありません。削除用ページを使って優子さんのメールアドレスを削除してしまうということは、この人のアドレスをemail_listテーブルから完全に消してしまうことになります。ということは優子さんは、もうメールメッセージを受け取ることはできなくなってしまいます。ここで必要なのは、優子さんの余分な行だけをテーブルから削除して、1つは残しておくことです。

前章で作ったメール削除用のページを使うと、
清野君のデータベースから顧客を完全に
消し去ってしまいます。それはここで
やりたいことではありません。





Jちゃん：メール追加用のフォームでメールアドレスをチェックしてから、新しいユーザを追加すべきなんじゃないかな？それで問題は解決じゃない？

森川氏：すばらしいアイデアだね。

沢田さん：そうね。それで今後は問題が起らなくなると思うわ。でも今もうデータベースに2重に入ってしまっているメールアドレスをどうにかすることはできないんじゃない？

森川氏：そうだね。テーブルの別のカラムを使って余分な行を削除してみるというはどうなのかねえ？例えばlast_nameとか。

沢田さん：それは無理じゃない？姓なんか使ったら、メールアドレスを使うよりもむしろ問題は悪くなってしまうと思うわ。例えばもしも佐藤由美子という人をメーリングリストから削除したいと思っているとするでしょ。それでこんなSQLコードを走らせたら：

```
DELETE FROM email_list WHERE last_name = '佐藤'
```

Jちゃん：佐藤由美子をテーブルから削除するだけじゃ済まないね。例えば、佐藤剛とか佐藤奈里彩とか佐藤健とか…みんな消えちゃうじゃない。

森川氏：おお、じゃ全然ダメだ。メールアドレスに比べたら姓は行の中で全然かぶりそうだし、名前だってダメだ。むしろ状況は悪化するかもしれない。単純な1つの問い合わせ文でとてつもなく多くの行を失ってしまうかもしれないね。

沢田さん：その通りだわ。WHERE節を使っても、とっておきたい行を削除してしまうかもしれないというリスクを回避できないわね。削除したい1行をピンポイントで指定する確実な方法が必要なのよ。

Jちゃん：一体どうすりやいいんだ？emailもlast_nameもfirst_nameもWHERE節には使えないときてる。

森川氏：テーブルの使えるカラムに見放されたんだ。ツキに見放されたみたいなもんだよ。

沢田さん：そうとばかりは言えないわ。本当に必要なものは、テーブルの各行を一意に識別することができるものでしょ。それがあれば何の問題もなくテーブルの行をピンポイントで指定できるわ。それで、今は一意の値を持つカラムが各行にないからといって、そういうカラムを追加できないということにはならないでしょ。

Jちゃん：新しいカラム？でももうテーブル構造は決めちゃったんだよ。

森川氏：そうだね。でも前に決めたのではダメだったってことだろ。あらかじめこうなることを予想できていたら、それに応じてテーブルを設計できていたよね。でも修正は今からでも遅くはないんじゃないかな。

Jちゃん：分かった。でもそのカラムは何ていう名前にする？あとどんなデータをそこにぶち込むの？

沢田さん：そうねえ。目的はテーブルの各行を一意に識別するっていうことだけだから、identifierで良いんじゃない？または略してidでも良いかもしれないわ。

森川氏：bingo。じゃあidカラムには各行毎に異なるID番号を埋めればいいわけだ。そうするとDELETEを実行するときには、一意な番号を基準に行を消すんだね。メールアドレスとか名前とかじゃなくて。

Jちゃん：当たり。本気ですごく良いアイデアだと思わない？こんなことを思いつくなんて超幸せ。

テーブル行は一意に識別できるべきです

データベースに何かをぶち込むというのは、後で探し出してそれを使って何かをしたいからです。この理解の下、テーブルの各行が一意に識別可能かどうかということは、とてつもなく重要な意味を持ちます。つまり特定の1行(かつその行だけ!)にアクセスできるからです。清野君のemail_listテーブルには、実は危険な仮定が隠れているのです。それはメールアドレスは一意であるということです。この仮定は誰もマーリングリストに間違って2度登録しない限りにおいてはちゃんと動きます。でも誰かが間違ってやってしまったら(たぶんやっちゃうでしょうが)、その人のメールアドレスはテーブルに2度登録されてしまいます…もう一意ではありません!

清野君のテーブル：現状版

last_name	first_name	email
森谷	隆司	mori@mightygumball.net
田中	順	tanaka@aliensabductedme.com
小島	優子	yuko@kresleespockets.biz
田中	優子	yuko@kresleespockets.biz

「姓」が同じ人が2人以上います。
ですから名前は一意なカラムとして
不適当です。

ここも同じで、「名」でも
一意になりません。

このテーブル構造では、どの行も
一意性を保障できません。

たいていの場合メールは
一意ですが、このように常に
というわけにはいきません。

テーブルに本当に一意な値を持つカラムがない場合は、そのようなカラムを作るべきです。MySQLには一意な整数値のカラムを追加する手段があります。このカラムは各テーブル行における主キーと呼ばれています。

清野君のテーブル：理想版

新しいカラムを用意して、テーブルの各行に
関して一意な値を保持するようにします。

id	last_name	first_name	email
1	森谷	隆司	mori@mightygumball.net
2	田中	順	tanaka@aliensabductedme.com
3	小島	優子	yuko@kresleespockets.biz
4	田中	優子	yuko@kresleespockets.biz

これで、このカラムは一意な値を保持する
ことになります。各行はこの値によって本当に
一意に識別できるという保障が得られます。

他のカラムで重複したデータが
あっても、もはや行の一意性には
影響がありません。idカラムが
その役割を担ってくれます。



ねえ、頭大丈夫？テーブル構造を変えたいと思ったら、DROP TABLEして、そのあとまっさらなテーブルを再構築しなきゃいけないの？清野君のメールデータが燃え尽きちゃうじゃない！

DROP TABLEしたら清野君のデータを間違いなく破壊してしまいます。でもSQLには他にもコマンドがあって、中のデータを失うことなく既存のテーブルを変更することができます。

そのコマンドは**ALTER TABLE**と言います。これを使うと、テーブルをドロップし、データを破壊することなく、新しいカラムを作ることができます。ALTER TABLE文の一般形、特にテーブルに新しいカラムを追加する場合は、次のような感じです。

```
ALTER TABLE table_name ADD column_name column_type
```

変更するテーブルの名前です。
新しく追加するカラムの名前です。
新しいカラムのデータ型です。

ALTER_TABLEコマンドを使って、email_listテーブルに新しいカラムを追加することができます。この新しいカラムにidと名づけます。idカラムには、データ型としてINTを与えます。なぜなら整数は一意性を確立するのに絶大な威力を発揮するからです。他にも少し、以下に示すような情報が必要です。

変更したいテーブルの名前です。

ADDしたい新しいカラムで、idという名前にしました。

これによりMySQLサーバは、このカラムに格納される値に、新しい行が挿入されるごとに1ずつ加えます。

```
ALTER TABLE email_list ADD id INT NOT NULL AUTO_INCREMENT FIRST,  
ADD PRIMARY KEY (id)
```

このカラムの型を整数(INT)にします。

FIRSTと書くとMySQLは、新しいカラムをテーブルの最初に起きます。これは省略しても構いませんが、idカラムを最初におくというのはよいことです。

この部分のコードによりMySQLは、新しく作ったidカラムがテーブルの主キーとして扱います。詳しい説明はもう少々お待ち下さい。

このALTER TABLE文には、やるべきことがたくさんあります。なぜなら主キーというものが、非常に特殊な特徴を携えて作らなければならないからです。例えば、NOT NULLというのはidカラムには必ず値がなければならないということをMySQLに教えるものです。このカラムを空白のままにすることはできません。AUTO_INCREMENTはidカラムの特性を述べています。これにより、新しい行が挿入された場合、自動的に一意の数値を設定してくれます。名前が示すとおり、AUTO_INCREMENTは、行内で最後に使われたidの値に1を加え、テーブルの新しい行にINSERTする際に、この値をidカラムに設定します。最後にPRIMARY KEYはidカラムの各値が一意であることをMySQLに教えると言いましたが、実は一意性以外にも役割があるのです…

主キーは一意性を強要する

主キーというのは、テーブルの中で各行を一意に区別するカラムのことです。通常のカラムも一意に設計されているかもしれません、主キーとなることができるるのは1つのカラムだけです。このため任意の問い合わせ文において、特定の行をピントに指定するには、どのカラムにすべきかは明らかです。

主キーの一意性を保障するために、MySQLではPRIMARY KEYと宣言されたカラムに対しては、多くの制約を課しています。これらの制約は主キーを使う場合に従わなければならない規則と考えてよいでしょう。

主キーに関する5つの規則：



主キーのデータは繰り返して現れてはいけません。

2つの行で主キーのデータが同じということがあってはいけません。例外なしです。主キーは1つのテーブル内では常に一意の値を持たなければなりません。



主キーは値を持たなければなりません。

主キーが空(NULL)のままだと、一意でなくなる可能性があります。なぜなら他の行もNULLとなる可能性があるからです。常に主キーには一意な値をセットして下さい！



新しい行を挿入する場合、主キーには値をセットしなければなりません。

主キーに値を設定せずに行を挿入してしまうと、主キーがNULLになってしまいうるリスクを負うことになります。これではテーブルに重複した行ができてしまう可能性が出てきますので、当初の目的にそぐいません。



主キーはできる限り効率的でなければなりません。

主キーは、一意性を保障するためだけの情報を持つべきであり、かつそれ以外は何も要りません。このため整数は主キーとして良いのです。小さな記憶域で一意性を保障してくれます。



主キーの値は変えることができません。

主キーの値を変えることができてしまったとすると、間違ってすでに使っている値をセットしてしまうかもしれないというリスクを負うことになります。何よりもまず一意性を保障しなければならないということを肝に銘じて下さい。

清野君のテーブル内でidカラムは
データを繰り返して使用することがなく、
すべての行に値があり、新しい行が
挿入される際には自動的に値が
セットされ、効率的にコンパクトで、
しかも値が変わりません。完璧です！

id	last_name	first_name	email
1	森谷	隆司	mori@mightygumball.net
2	田中	順	tanaka@aliensabductedme.com
			...



主キーというのは
テーブル内で
各行を一意に
するためのカラム
です。



試運転

清野君のテーブルを変更し、主キーを持つ新しいデータ行を挿入してみる。

MySQLターミナルやphpMyAdminのSQLタブなどのMySQLツールを使って、`ALTER TABLE`文を入力し、`id`という名前の主キーを追加します。

```
ALTER TABLE email_list ADD id INT NOT NULL AUTO_INCREMENT FIRST,
ADD PRIMARY KEY (id)
```

では新しい顧客をデータベースに挿入し `id`カラムが新しい行に自動的にセットされるかどうかを見てみます。以下は `INSERT` 文の例です。(主キーについては何も書いていないことに注意して下さい。)

```
INSERT INTO email_list (last_name, first_name, email)
VALUES ('船山', '博美', 'funako@sterling-cooper.com')
```

最後に `SELECT` 文を発行しテーブルの中身を覗いて、新しい主キーが栄光の規則に則っていることを確認しましょう！忘れてしまった方のために、以下に `SELECT` 文を示します。

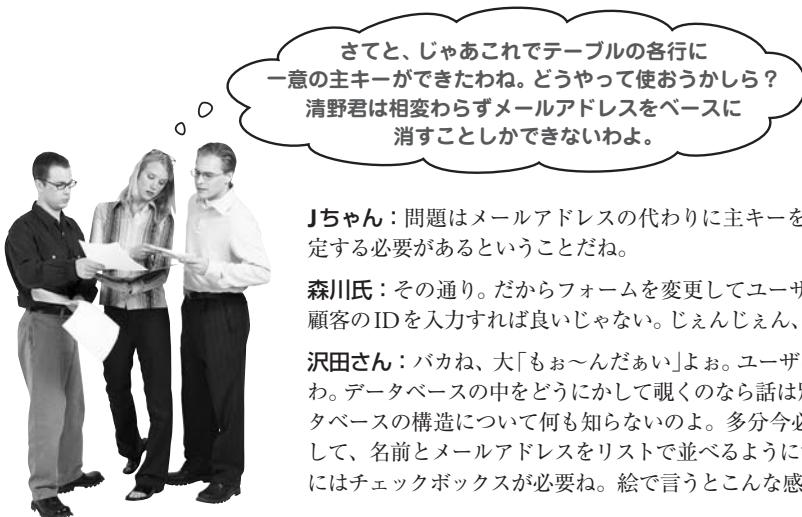
```
SELECT * FROM email_list
```

新しい `id` カラムは自動的に増加し、新しい行データの一意性を確保しています。

データベース elvis_store 上でクエリを実行する: ⑦ —

```
SELECT * FROM `email_list`
```

	←↑→	<code>id</code>	<code>last_name</code>	<code>first_name</code>	<code>email</code>
<input type="checkbox"/>		X	1 森谷	隆司	mori@mightygumball.net
<input type="checkbox"/>		X	2 田中	順	tanaka@aliensabductedme.com
<input type="checkbox"/>		X	3 小島	優子	yuko@kresleespockets.biz
<input type="checkbox"/>		X	4 田中	優子	yuko@kresleespockets.biz
<input type="checkbox"/>		X	5 船山	博美	funako@sterling-cooper.com



Jちゃん：問題はメールアドレスの代わりに主キーを使ってピンポイントに行を指定する必要があるということだね。

森川氏：その通り。だからフォームを変更してユーザはメールアドレスの代わりに顧客のIDを入力すれば良いじゃない。じえんじえん、もお～んだあいなし！

沢田さん：バカね、大「もお～んだあい」よお。ユーザは顧客のIDを知る方法がないわ。データベースの中をどうにかして覗くのなら話は別だけど。実際、ユーザはデータベースの構造について何も知らないのよ。多分今必要なのは、フォームを考え直して、名前とメールアドレスをリストで並べるようにすることだわ。リストの各項目にはチェックボックスが必要ね。絵で言うとこんな感じ。



森川氏：いい絵だけど、これでどうやって清野君は顧客のIDを使って削除したい顧客をより分けることができるようになるの？

Jちゃん：う～ん、チェックボックスに顧客IDを格納しておくっていうのはどう？ そうすれば実際には見えないけど、スクリプトにはIDが分かるよ。

沢田さん：すばらしいアイデアだわ。それじゃあループで自動的にフォームを作り出すことができればいいのね。SELECTでデータを取ってきて、問い合わせ文のデータ行から各チェックボックスの入力フィールドを作れるわ。

Jちゃん：イケてる。でも「削除」ボタンを押したときにはどうすればいいんだ？ \$_POSTには何が入っているんだろう？

森川氏：Jちゃん、ちょっと待ってよ。そんなのすぐに分かるよ。まずはこの部分のスクリプトを作り始めちゃおう。この部分って言うのはつまり、全データをテーブルから取ってきて表示して、あとチェックボックスを書き出すところ…



PHP & MySQL マグネット

下にあるマグネットを使ってメール削除用スクリプトの足りないコードを完成させて下さい。清野君のデータベース中の顧客にチェックボックスを付けた行を並べます。このコードはフォームを作るだけです。コードがDELETEを実行することについてはまだ気にしなくて構いません。

```

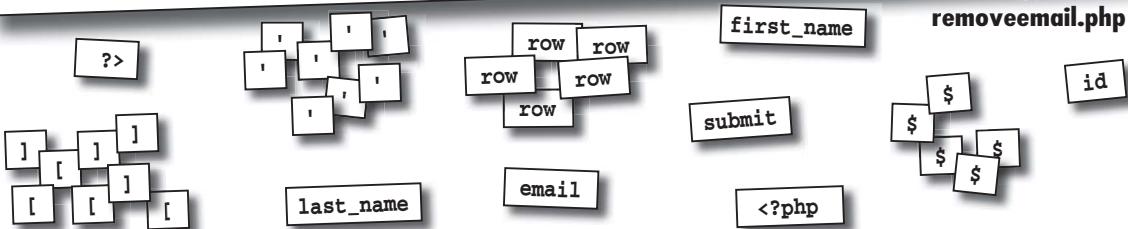


<p>以下の email リストから削除するメールアドレスを選んで「削除」をクリックして下さい。</p>
<form method="post" action=" ..... echo $_SERVER['PHP_SELF']; ..... ">
<?php
$dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')
or die('エラー:MySQLとの接続に失敗しました。');
// Display the customer rows with checkboxes for deleting 削除する顧客の行をチェックボックス付きで表示
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query);
while ( ..... = mysqli_fetch_array($result)) {
    echo '<input type="checkbox" value="1" ..... . ' name="todelete[]" />';
    echo ..... ;
    echo ' ' . ..... ;
    echo ' ' . ..... ;
    echo '<br />';
}
mysqli_close($dbc);
?>
<input type="submit" name=" ..... " value="削除" />
</form>

```



removeemail.php





PHP & MySQL マグネットの答え

下にあるマグネットを使ってメール削除用スクリプトの足りないコードを完成させて下さい。清野君のデータベース中の顧客にチェックボックスを付けた行を並べます。このコードはフォームを作るだけです。コードがDELETEを実行することについてはまだ気にしなくて構いません。

```





以下のemailリストから削除するメールアドレスを選んで「削除」をクリックして下さい。


<form method="post" action="...<?php echo $_SERVER['PHP_SELF']; ...?>">
    <?php echo $row['id'] . ' ' . $row['last_name'] . ' ' . $row['first_name'] . ' ' . $row['email'] . '<br />';
    <input type="checkbox" value="1" checked="checked" name="todelete[]" />
    <input type="submit" name="submit" value="削除" />

```

このフォームは自己参照です！

インライン PHP コードは相変わらず<?php と ?>タグの中におきます。

削除する顧客の行をチェックボックス付きで表示

ここで主キーがチェックボックスに使われます。あとでチェックされた任意の顧客を削除するときに使います。

各チェックボックスの入力
フィールドは顧客データの行で構成されます。

スクリプトは実際にはまだ何も削除しません。今のところはチェックボックスのリストを表示するだけです。



「提出」ボタンの名前は何でも構いません。
ただし後でフォームが「提出」されたか
どうか\$_POSTでチェックするときに
この名前を使いますので忘れずに覚えて
おいて下さい。

チェックボックスから顧客IDへ

チェックボックスコードは、メール削除用スクリプトから生成された、単なるHTMLに<input>タグのvalue属性として主キー(id)を付与したもので。ところが1つ小さな、しかし非常に重要な変更があって、これは通常のチェックボックスのHTMLコードとは違うのです。すでにお気づきかも知れませんが、チェックボックス名の後ろに角カッコ([])がついています。これが極めて重要な用途を提供してくれます。

```
echo '<input type="checkbox" value="' . $row['id'] . '" name="todelete[]" >';
```

シカクカッコは、\$_POSTの中に配列を作ってくれます。この配列はフォームのチェックボックスでチェックされたすべてのvalue属性の内容を格納します。今、各チェックボックスのvalue属性は、主キーを保持していますから、todelete配列の各値は、テーブル内で削除するべき行のIDということになります。これによりtodelete配列をループで回してSQL問い合わせ文を発行すれば、フォームでチェックされた各顧客を削除できることになります。

↑
チェックボックス名の後ろにあるシカクカッコで、“todelete[]”という名前をつけた配列にチェックボックスの値を自動的に入れてくれるようになります。

フォームフィールドの各チェックボックスは、配列に格納された顧客IDをもち、\$_POSTスーパーグローバルによりアクセスすることができます。

分かったわ。じゃあ後はwhileループでtodelete配列を回して顧客のIDを使ってそれを削除すれば良いのね。

whileループでもできますが、別のループを使ったもっとエレガントな解決策があります。

foreachループは、配列に格納された値を回すためだけに設計された特殊なループです。やるべきことはループしたい配列と値を格納するための変数を指定することだけです。するとPHPは1つずつ繰り返して処理してくれます…テスト条件は要りません！

foreachループが清野君の顧客IDの配列をどのように回ると思いますか？書き留めておきましょう。

foreachにより配列をループする

foreachループは配列をパラメタとして、配列の各要素をループします。テスト条件やループ回数のカウントは必要ありません。foreachは配列の要素毎にステップ実行するので、その要素の値を変数に一時的に格納する必要があります。例えば配列が\$customersという名前の変数に格納されているとすると、以下のコードは各要素をステップ実行します。

```
まずループで回したい  
配列名を書きます。  
foreach ($customers as $customer) {  
    echo $customer;  
};
```

ループは配列の個々の要素を回るので、その値はこの名前の変数に一時的に格納されます。

ループの中では、先ほど決めた変数を使って各要素にアクセスできます。

そこで、メール削除用スクリプトの\$_POST配列に格納されている顧客IDでループする場合は、以下のようなforeachコードを使えばよいでしょう。

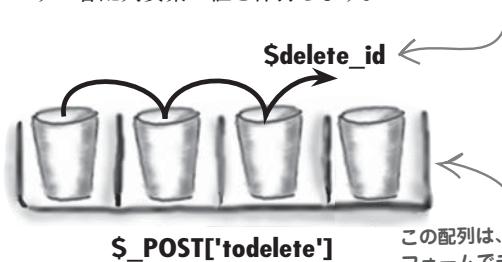
```
この場合、配列は$_POSTスーパー  
グローバルの中に格納されていて、  
"todelete"で識別することができます。  
foreach ($_POST['todelete'] as $delete_id) {  
    // Delete a row from the table テーブルから1行を削除  
};
```

配列の各要素は変数
\$delete_idを通してアクセスすることができます。

\$delete_idを使ってデータベースの各顧客を削除することができます。

この変数を使って各顧客のIDにアクセスし、その後テーブルから削除します。

\$delete_id変数は、ループの進行と共に各回1つずつ各配列要素の値を保持します。



この配列は、メール削除用フォームでチェックされた顧客のみを保持するように作られています。

これでforeachループは、メール削除用フォームの中でチェックされた各チェックボックスをステップ実行するようになりました。あとはループの内側のコードを追加するだけです。DELETE問い合わせ文を発行し、実際にemail_listテーブルから各行を削除します。



清野君の新しい改良版removemail.phpスクリプトを完成させて下さい。フォームが「提出」された際に、チェックされた顧客を削除します。

```
...
$dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')
or die(' エラー：MySQLとの接続に失敗しました。');

// Delete the customer rows (only if the form has been submitted)
if ( ..... ) {
    foreach ($_POST['todelete'] as $delete_id) {
        .....
        .....
        .....

    }
    echo ' 顧客を削除しました。<br />';
}

// Display the customer rows with checkboxes for deleting
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query);
while ($row = mysqli_fetch_array($result)) {
    echo '<input type="checkbox" value="' . $row['id'] . '" name="todelete[]" />';
    echo $row['last_name'];
    echo ' ' . $row['first_name'];
    echo ' ' . $row['email'];
    echo '<br />';
}

mysqli_close($dbc);
?>

<input type="submit" name="submit" value=" 削除 " />
</form>
```



removeemail.php



清野君の新しい改良版removeemail.phpスクリプトを完成させて下さい。フォームが「提出」された際に、チェックされた顧客を削除します。

削除するのは、
フォームが
「提出」され
た場合だけ
です！

```
...
$dbc = mysqli_connect('data.makemeelvis.com', 'seino', 'theking', 'elvis_store')
or die('エラー：MySQLとの接続に失敗しました。');

// Delete the customer rows (only if the form has been submitted)
if (isset($_POST['submit'])) {
    foreach ($_POST['todelete'] as $delete_id) {
        $query = "DELETE FROM email_list WHERE id = $delete_id";
        mysqli_query($dbc, $query)
        or die('エラー：データベースの問い合わせに失敗しました。');
    }
    echo '顧客を削除しました。  
>';
}

// Display the customer rows with checkboxes for deleting
$query = "SELECT * FROM email_list";
$result = mysqli_query($dbc, $query);
while ($row = mysqli_fetch_array($result)) {
    echo '<input type="checkbox" value="' . $row['id'] . '" name="todelete[]" />';
    echo $row['last_name'];
    echo ' ' . $row['first_name'];
    echo ' ' . $row['email'];
    echo '<br />';
}
mysqli_close($dbc);

<input type="submit" name="submit" value="削除" />
</form>
```

顧客の行を削除（フォームが「提出」された場合）

\$delete_idを使って削除する顧客をドンピシャで抽出します。

削除する顧客の行をチェックボックス付きで表示

顧客のチェックボックスを作り出すコードは、先ほど作ったものと同じです。



removeemail.php



試運転

新たに改良された清野君のメール削除用スクリプトをちょいと走らせる

removeemail.phpスクリプトのコードを修正します。以前のemailテキストフィールドを使うものの代わりに、顧客のチェックボックスを生成するようにします。次にフォームが「提出」された場合、顧客を削除するコードを追加します。また<form>タグのaction属性を変更し、フォームを自己参照にします。

これでremoveemail.phpは自己参照フォームになりましたから、removeemail.htmlページは、もはやWebサーバ上に必要ありません。気兼ねなく削除して下さい。次に新しいバージョンのremoveemail.phpスクリプトをWebサーバにアップロードし、スクリプトをWebブラウザで開きます。何人かの顧客をチェックしたら「提出」をクリックします。フォームは瞬時に変わり、顧客の削除が反映されるはずです。

以下のemailリストから削除するメールアドレスを選んで「削除」をクリックしてください。

- 森谷 隆司 mori@mightygumball.net
- 田中 順 tanaka@aliensabductedme.com
- 小島 優子 yuko@kresleesprockets.biz
- 田中 優子 yuko@kresleesprockets.biz
- 舟山 博美 funako@sterling-cooper.com

削除

顧客をチェックして「提出」をクリックすると、顧客はデータベースから削除されます。

以下のemailリストから削除するメールアドレスを選んで「削除」をクリックしてください。

- 森谷 隆司 mori@mightygumball.net
- 田中 順 tanaka@aliensabductedme.com
- 小島 優子 yuko@kresleesprockets.biz
- 舟山 博美 funako@sterling-cooper.com

削除

顧客を削除しました。

スクリプトは、顧客を確かに削除し、またチェックリストを更新してくれます。削除した顧客はもう現れません。

清野君は完全に機能するアプリケーションを手に入れました。顧客の追加、超豪華セルのメール送信、それも本当に受け取りたいと思っている顧客にだけの送信、そして顧客の削除、それも興味のなくなった顧客の削除や単にリストから削除したいだけなど何でも。すばらしきかな人生。



PHP & MySQL 道具箱

清野君のWebアプリケーションを全体として次のレベルまで高めるうちに、非常に多くのPHPおよびMySQLのスキルを手に入れました。

!

否定演算です。true/false値を反転させます。つまり true は false となり false は true となります。

foreach

PHP のループ構成要素で、テスト条件などを使うことなく配列の各要素を 1 つずつループします。ループの中で配列の各要素にアクセスすることができます。

isset(), empty()

PHP 組み込みの `isset()` 関数は、変数が存在するかどうか、つまり変数に値が代入されているかどうか、をテストします。`empty()` 関数は、もう 1 ステップ先を見て、変数が空の値(0、空文字列、`false`、`NULL`)かどうかをテストします。

ALTER TABLE

この SQL 文はテーブルの構造を変更します。例えば新しいデータカラムの追加などです。これによりテーブルの構造を、一度すべてなくして作り直すことなく、変更することができます。

if, else

PHP の if 文は何かが `true` かどうかに基づいて判断を行います。`true/false` のテスト条件と動作部のコードが与えられると、if 文はあらゆる種類のイケてる判断を下すことができます。`else` 節を追加すると if 文は二者択一の動作をとることができます。

`==`, `<>`, `!=`, `<`, `>`,
...

比較演算はテスト条件として使うことができ、2つの値を比較します。if 文やループを制御する場合によく用いられます。

&&, ||

論理演算子と言って、`true/false` 値をとる式を組み立てるのに使われます。`&&`(論理積)で 2 つの値をくっつけると、両方の値がともに `true` の場合のみ `true` となります。`||`(論理和)で 2 つの値をくっつけると、いずれか一方の値が `true` の場合 `true` となります。

5章 ファイルに格納されたデータを使う

データベースでは
十分でないとき

それを僕にアップロード
する権限をあげる。美味し
そうだから僕のお腹の中に
ファイルしておくよ！

じゃあ私はアナタを
有害物質フォルダに
ファイルしとくわ。



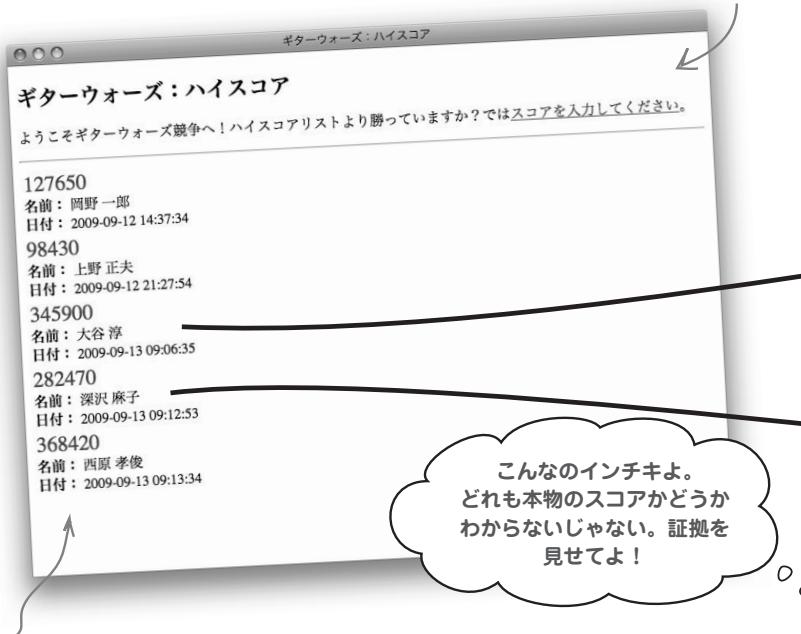
データベースに関する…誇大広告に騙されてはいけません。

確かにデータベースはテキストなどあらゆる種類のデータを格納するという驚くべき仕事をしてくれます。でもバイナリデータはどうでしょうか？例えばJPEGイメージとかPDFドキュメントなどです。珍しいギターピックコレクションの写真を全部データベースにぶち込むことに意味があると思いますか？普通はありません。その手のデータはファイルに格納するのが普通で、ファイルのままにしておくものです。でも仮想的なケーキを持って仮想的に食べることが完璧にできてしまうようになります。本章では、ファイルとデータベースとと一緒にしてバイナリデータで満ちあふれたPHPアプリケーションを構築するための秘密を暴露します。

仮想ギタリストはコンペ好きです

明らかに芸術のための芸術は、常に満足を得られません。プレーヤーは今ギターウォーズというホットな新しいゲームで仮想ギター演奏を競うことに魅了されているのです。このゲームがあまりに過熱しているためプレーヤーは自分のハイスクアをギターウォーズのWebサイトにじゃんじゃんポストしています。私達はそのメンテナンスをやらされる羽目に陥りましたが、問題なのは、今のところスコアを証明する良い方法がないことです。

ギターウォーズアプリケーションでは、
自分自身のスコアをハイスクアリスト
に追加させてくれます。



証明する方法がないので、
誰のスコアは正しくて誰のが
ウソなのかわかりません。

ギターウォーズに懐疑的な
ロック奏者で、麻子さんと
いいます。

テキストでは信用できません

現時点ではプレーヤーはハイスクアを純粋にテキストとしてのみポストしていますから、誰のスコアは本当で誰のがウソかと
いうことで多くの論争が起こっています。この論争に終止符を打ち、ギターウォーズの正当なチャンピオンに王冠を授与する方法
が1つだけあります…



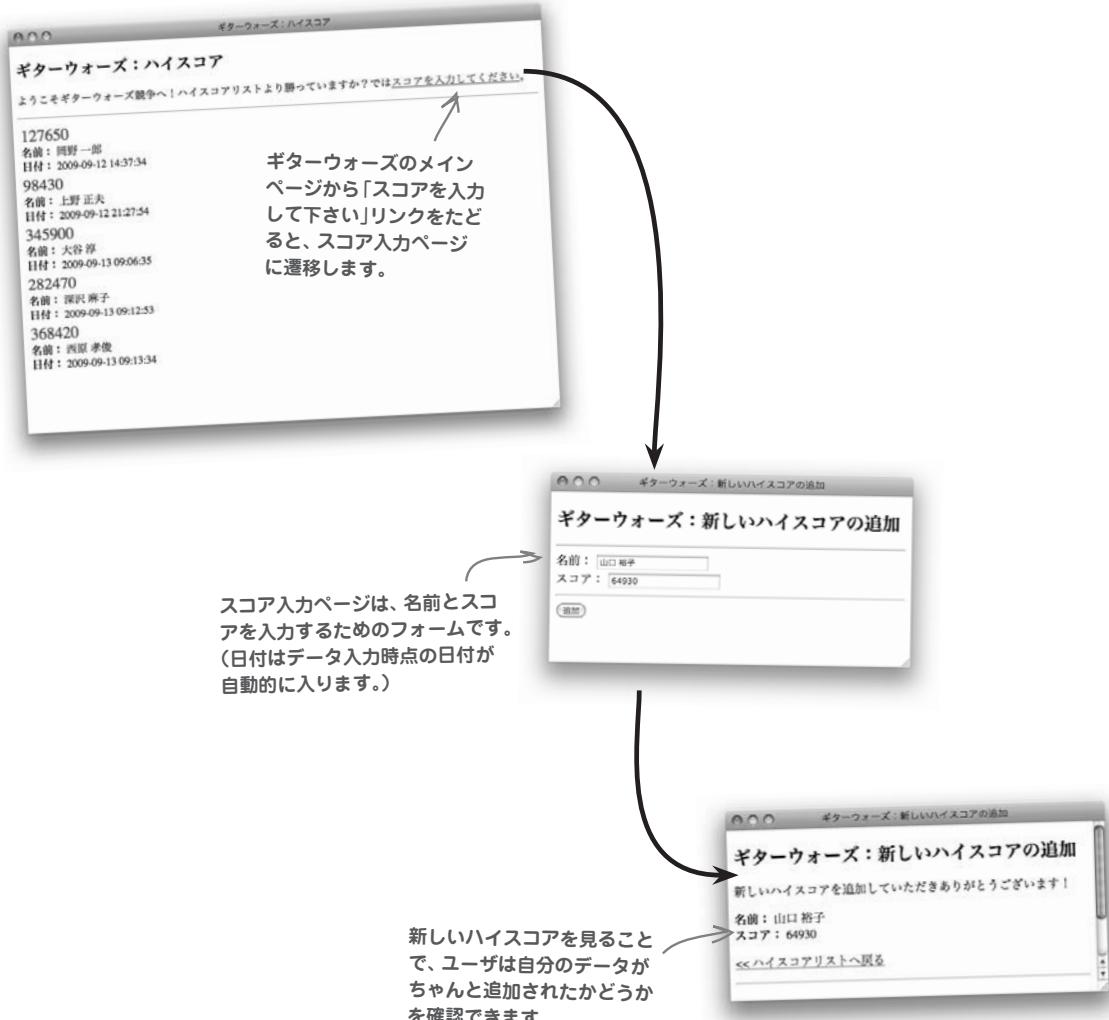
写真 証拠はロックの中にある

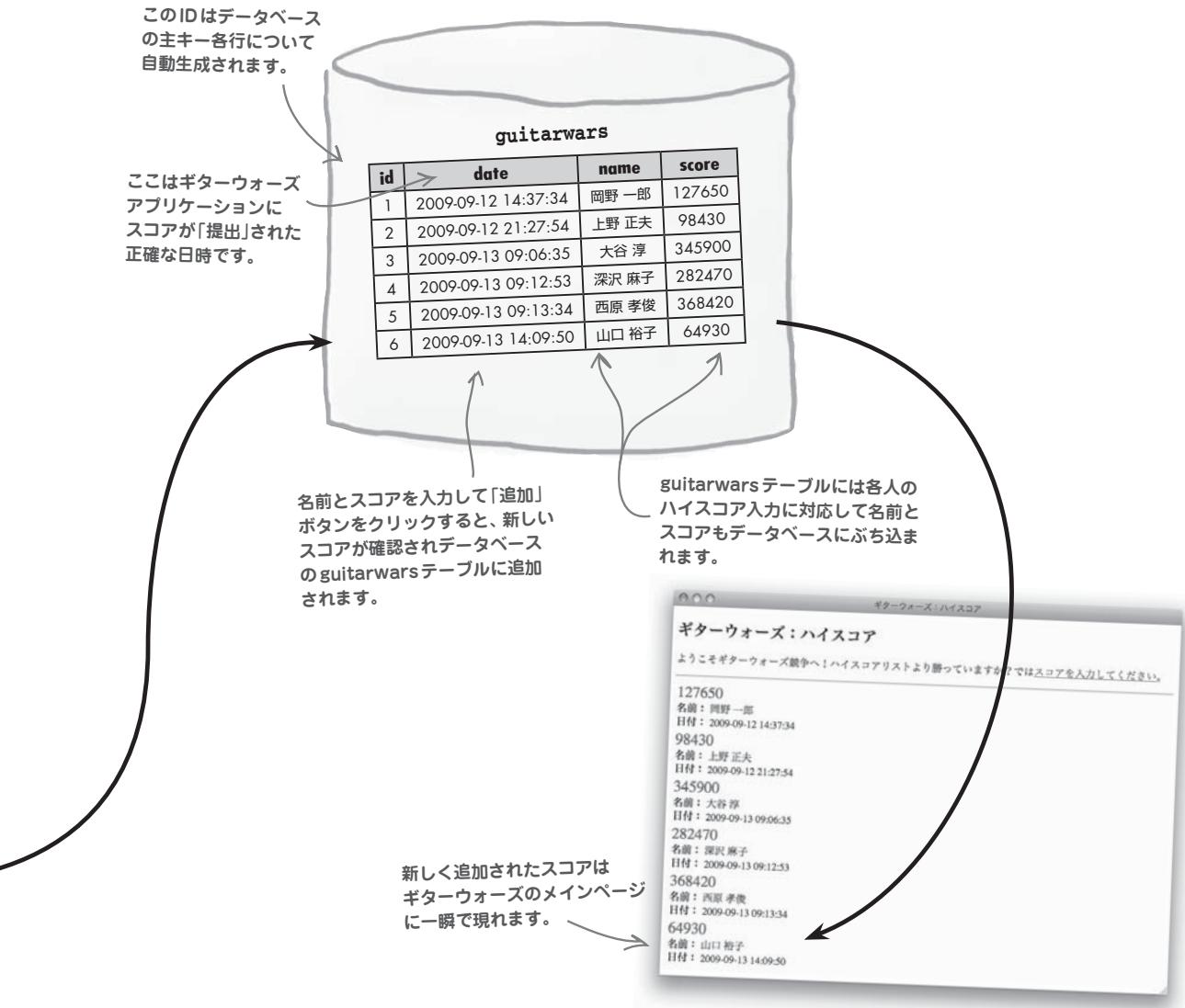
ハイスコアを目で見て証明できれば、誰が本物でだれがウソかすぐわかります。そこでギターウォーズアプリケーションにはユーザがスコアをポストするときにその人のハイスコアのスクリーンショットを「提出」させる機能があれば良いことになります。つまりハイスコアのリストというのは、スコアと名前と日付だけでなくイメージ(スクリーンショット)も必要になるという意味です。



アプリケーションにイメージを格納する必要があります

現状のギターウォーズハイスクアアプリケーションは、3種類の情報を管理しています。新しいスコアの日時、スコアを「提出」した人の名前、そしてスコアそのものです。この情報はアプリケーションの一部であるユーザインターフェースのフォームを通して入力されます。その後データはguitarwarsという名前のMySQLデータベースに突っ込まれます。







ギターオーズのハイスコアアプリケーションを、ハイスコアのスクリーンショットをファイルでアップロードできるように変更します。ユーザがイメージファイルを「提出」できるようにするために変更が必要な箇所にマルと注釈をつけて下さい。

```

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
  <title>ギターオーズ : ハイスコア </title>
  <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
  <h2>ギターオーズ : ハイスコア </h2>
  <p>ようこそギターオーズ競争へ！ハイスコアリストより勝っていますか？<br/>
  では<a href="addscore.php">スコアを入力して下さい</a>。</p>
  <hr />

<?php
  // Connect to the database データベースへの接続
  $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit',
    'gwdb');

  // Retrieve the score data from MySQL MySQLからスコアデータの取り出し
  $query = "SELECT * FROM guitarwars";
  $data = mysqli_query($dbc, $query); スコアデータの配列をループし、HTMLにフォーマット
  // Loop through the array of score data, formatting it as HTML
  echo '<table>';
  while ($row = mysqli_fetch_array($data)) { スコアデータの表示
    // Display the score data
    echo '<tr><td class="scoreinfo">';
    echo '<span class="score">' . $row['score'] . '</span><br />';
    echo '<strong>名前:</strong> ' . $row['name'] . '<br />';
    echo '<strong>日付:</strong> ' . $row['date'] . '</td></tr>';
  }
  echo '</table>';
  mysqli_close($dbc);
?>

</body>
</html>

```



このファイルは変更の必要がありませんから、気にしなくて構いません。



index.php

ダウンロードして下さい！

ギターオーズアプリケーションの全ソースコードは以下のサイトからダウンロードできます。

www.oreilly.co.jp/book/9784873114446/

guitarwars

id	date	name	score
1	2009-09-12 14:37:34	岡野一郎	127650
2	2009-09-12 21:27:54	上野正夫	98430
3	2009-09-13 09:06:35	大谷淳	345900
4	2009-09-13 09:12:53	深沢麻子	282470
5	2009-09-13 09:13:34	西原孝俊	368420
6	2009-09-13 14:09:50	山口裕子	64930

```

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
  <title> ギターウォーズ：新しいハイスクアの追加 </title>
  <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
  <h2> ギターウォーズ：新しいハイスクアの追加 </h2>

<?php
  if (isset($_POST['submit'])) {
    // Grab the score data from the POST POSTからスコアデータの取り出し
    $name = $_POST['name'];
    $score = $_POST['score'];

    if (!empty($name) && !empty($score)) {
      // Connect to the database データベースへ接続
      $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit', 'gwdb');

      // Write the data to the database データをデータベースへ書き込み
      $query = "INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score')";
      mysqli_query($dbc, $query);

      // Confirm success with the user 確認画面の表示
      echo '<p>新しいハイスクアを追加していただきありがとうございます！</p>';
      echo '<p><strong>名前:</strong> ' . $name . '<br />';
      echo '<strong>スコア:</strong> ' . $score . '</p>';
      echo '<p><a href="index.php">&lt;&lt; ハイスコアリストへ戻る </a></p>';

      // Clear the score data to clear the form
      $name = "";
      $score = "";
```

スコアデータを消去してフォームを空白にする

```

      mysqli_close($dbc);
    }
    else {
      echo '<p class="error">エラー：入力項目をすべて埋めてからハイスクア' .
        'を追加して下さい。</p>';
    }
  }
?>

<hr />
<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
  <label for="name">名前:</label><input type="text" id="name" name="name" value="<?php if (!empty($name)) echo $name; ?>" /><br />
  <label for="score">スコア:</label><input type="text" id="score" name="score" value="<?php if (!empty($score)) echo $score; ?>" />
  <input type="submit" value="追加" name="submit" />
</form>
</body>
</html>

```





エクササイズ の答え

ギターウォーズのハイスコアアプリケーションを、ハイスコアのスクリーンショットをファイルでアップロードできるように変更します。ユーザがイメージファイルを「提出」できるようにするために変更が必要な箇所にマルと注釈をつけて下さい。

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
<title>ギターウォーズ:ハイスコア</title>
<link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
<h2>ギターウォーズ:ハイスコア</h2>
<p>ようこそギターウォーズ競争へ!ハイスコアリストより勝っていますか?  
では<a href="addscore.php">スコアを入力して下さい</a>.</p>
<hr />

<?php
// Connect to the database
$dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit',
    $query = "SELECT * FROM guitarwars";
$data = mysqli_query($dbc, $query);
// Loop through the array of score data, formatting it as HTML
// Display the score data
while ($row = mysqli_fetch_array($data)) {
    // Display the score data
    echo '<tr><td class="scoreinfo">';
    echo '<span>' . $row['score'] . '</span><br />';
    echo '<strong>名前:</strong> ' . $row['name'] . '<br />';
    echo '<strong>日付:</strong> ' . $row['date'] . '</td></tr>';
}
echo '</table>';
mysqli_close($dbc);
?>
```

スクリーンショットの
イメージはメインページ
に表示させます。



index.php

イメージをユーザに表示して登録が成功
したことを示す方がよいでしょう。

guitarwars			
id	date	name	score
1	2009-09-12 14:37:34	岡野一郎	127650
2	2009-09-12 21:27:54	上野正夫	98430
3	2009-09-13 09:06:35	大谷淳	345900
4	2009-09-13 09:12:53	深澤麻子	282470
5	2009-09-13 09:13:34	西原李俊	368420
6	2009-09-13 14:09:50	山口裕子	64930

テーブルには新しい
カラムが必要で、そこに
スクリーンショットの
イメージファイル名を
格納します。

フォームにはイメージ
ファイルを入力するため
の<input>タグが必要です。

スクリーンショット
のイメージファイル
はフォームのPOST
データとして持つ
くる必要があります。

イメージファイル名が
空でないことの妥当性
を検証する必要が
あります。

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
<title>ギターウォーズ:新しいハイスコアの追加</title>
<link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
<h2>ギターウォーズ:新しいハイスコアの追加</h2>
<?php
if (isset($_POST['submit'])) {
    // Grab the score data from the POST
    $name = $_POST['name'];
    $score = $_POST['score'];

    if (!empty($name) && !empty($score)) {
        // Connect to the database
        $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit', 'gwdb');

        // Write the data to the database
        $query = "INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score')";

        mysqli_query($dbc, $query);

        // Confirm success with the user
        echo '<p>新しいハイスコアを追加していただきありがとうございます！</p>';
        echo '<p><strong>名前:</strong> ' . $name . '<br />';
        echo '<strong>スコア:</strong> ' . $score . '</p>';
        echo '<p><a href="index.php">ハイスコアリストへ戻る</a></p>';

        // Clear the score data to clear the form
        $name = "";
        $score = "";
    }
    else {
        echo '<p class="error">エラー: 入力項目をすべて埋めてからハイスコア  
' . 'を追加して下さい。</p>';
    }
}

<hr />
<form method="post" action="php echo $_SERVER['PHP_SELF']; ?&gt;"&gt;
&lt;label for="name"&gt;名前:&lt;/label&gt;&lt;input type="text" id="name" name="name" value="<?php if (!empty($name)) echo $name; ?&gt;" /&gt;&lt;br /&gt;
&lt;label for="score"&gt;スコア:&lt;/label&gt;&lt;input type="text" id="score" name="score" value="<?php if (!empty($score)) echo $score; ?&gt;" /&gt;
&lt;hr /&gt;
&lt;input type="submit" value="追加" name="submit" /&gt;
&lt;/form&gt;
&lt;/body&gt;
&lt;/html&gt;</pre

```

この問い合わせ文はカラム名
を指定しない省略形式の
書き方を使っています。



addscore.php

ギターウォーズにイメージファイルアップロードを計画する

スクリーンショットのイメージをギターウォーズにアップロードできるようにするということには、大した苦労は要らないように思えるかもしれません。しかし実際にはアプリケーションにはいろいろな方法で変更を加えることが可能なのです。従ってどんなコードでも、飛び込む前には必ず攻撃の計画を立てるというのは良いことです。ギターウォーズのハイスコアにスクリーンショットを追加するという改訂に必要となるステップをはっきりさせていきましょう。

① ALTERを使ってテーブルに screenshot カラムを追加する。

まず第1歩はデータベースです。スクリーンショットのイメージファイルをぶち込むための新しいカラムが必要です。現在の計画では全イメージファイルを1つのフォルダに突っ込むつもりですので、データベースにぶち込むのは、ファイル名のみ(パスなし)で良いでしょう。



③ スクリーンショットのイメージファイル名をテーブルの screenshot カラムに INSERT する問い合わせ文を書きます。

スコア追加用スクリプトは、従来フォームを処理してスコアを追加していましたが、さらに新しくファイル入力フォームフィールドも考慮に入れなければならなくなりました。つまり、新しくハイスコア行をguitarwarsテーブルに挿入する際には、スクリーンショットのイメージファイル名を screenshot カラムに挿入するという処理をしなければいけません。



② スコア追加用フォームを変更し、ファイル入力フィールドを使ってイメージファイルをアップロードできるようにします。

スコア追加用ページには、スコアを追加するためのフォームがすでにありますから、これを変更してファイル入力フィールドを加えればよいでしょう。この入力フィールドは、Webブラウザの協力の下、ユーザにアップロードするファイルを選ぶようなインターフェースを提供してくれます。

スクリーンショット : saitosscore.gif

④ ギターウォーズのメインページを変更して、ハイスコアに 対応するスクリーンショットのイメージを表示します。

最後に変更すべきもののリストは、ギターウォーズのメインページindex.phpをも書き込んでいます。メインページで各ハイスコアにスクリーンショットのイメージと一緒に実際に表示するという変更が必要です。



ハイスクアデータベースは ALTER しなければいけません

PHP スクリプトのマイナーチェンジもいろいろと必要ですが、さらにイメージによる強化版ギターウォーズアプリケーションには、guitarwars テーブルに新しいカラムが必要で、そこにスクリーンショットのイメージファイル名を格納します。SQL で ALTER 文を入力して、テーブルを修正します。ALTER 文を使えばテーブルをどのように変更することもできますから、新しいカラムを追加することもできます。ALTER 文は前章で清野君の email_list テーブルを変更するときにも使いましたが、コマンドの働きをもう一度復習しましょう。

```
ALTER TABLE guitarwars DROP COLUMN score
```

DROP COLUMN 文は
テーブルから1つのカラム
全体を削除します。

そうなんです。実はこれは危険な例なのですが、テーブルから1つのカラムをデータごと削除する方法をバラしてしまいました。でもまあ、テーブルからデータごとカラムを削除する必要がある、という状況はあり得るかもしれません。もっとありそうなのは、やはりデータのカラムを追加する必要があるという、今回のギターウォーズのような場合でしょう。追加は ADD COLUMN で実現できますが、この操作は ALTER を使って行えるテーブル変更操作のうちの1つです。

ADD COLUMN

テーブルに新しいカラムを追加します。
カラム名とその型を ADD COLUMN に
続けて指定するだけです。

```
ALTER TABLE guitarwars
```

```
ADD COLUMN age TINYINT
```

CHANGE COLUMN

カラムの名前とデータ型を変更します。以前の
カラム名、新しいカラム名、新しいデータ型を
CHANGE COLUMN に続けて指定するだけです。

```
ALTER TABLE guitarwars
```

```
CHANGE COLUMN score high_score INT
```

DROP COLUMN

テーブルからカラム（とそこに入っている全データ）を削除します。カラム名を DROP COLUMN に続けて指定するだけです。

```
ALTER TABLE guitarwars
```

```
DROP COLUMN age
```

MODIFY COLUMN

カラムのデータ型かまたはテーブル内での位置を変更します。カラム名と新しいデータ型を MODIFY COLUMN に続けて指定するだけです。カラムの位置を変更する場合は、カラム名とその絶対位置を指定するか（この場合 FIRST だけが指定できます）または相対位置を指定します（AFTER に続けて既存のカラムを名前で指定します）。

```
ALTER TABLE guitarwars
```

```
MODIFY COLUMN date DATETIME AFTER age
```

ALTER 文は
データベースの
構造を変更する[†]
ために使います。

ALTER 文は直後に TABLE を伴った形で、よく使いますが、これはテーブルを変更しようとしていることを意味しています。データベース全体の構造を変更するために ALTER DATABASE としても可能ですが、これは別の話題です。

[†] 訳注：ALTER とは「変更する」という意味です。

自分で考えてみよう

SQL文を書いて、guitarwarsテーブルにScreenshotという新しいカラムを追加して下さい。新しいカラムにはMySQLデータ型として適切なものを与えて下さい。次に別のSQL問い合わせ文を書いてテーブルの構造をチェックしカラムが正しく追加されたかどうかを確認して下さい。

guitarwars

id	date	name	score	screenshot
1	2009-09-12 14:37:34	岡野 一郎	127650	
2	2009-09-12 21:27:54	上野 正夫	98430	
3	2009-09-13 09:06:35	大谷 淳	345900	
4	2009-09-13 09:12:53	深沢 麻子	282470	
5	2009-09-13 09:13:34	西原 孝俊	368420	
6	2009-09-13 14:09:50	山口 裕子	64930	

ここにカラムを追加する
文を書いて下さい。



.....

.....

もう1つのSQL文を
ここに書いて下さい。



.....

自分で考えてみよう の答え

SQL文を書いて、guitarwarsテーブルに screenshot という新しいカラムを追加して下さい。新しいカラムには MySQL データ型として適切なものを与えて下さい。次に別の SQL 問い合わせ文を書いてテーブルの構造をチェックしカラムが正しく追加されたかどうかを確認して下さい。

id	date	name	score	screenshot
1	2009-09-12 14:37:34	岡野 一郎	127650	
2	2009-09-12 21:27:54	上野 正夫	98430	
3	2009-09-13 09:06:35	大谷 淳	345900	
4	2009-09-13 09:12:53	深沢 麻子	282470	
5	2009-09-13 09:13:34	西原 孝俊	368420	
6	2009-09-13 14:09:50	山口 裕子	64930	

ALTER文で guitarwars テーブルに新しく screenshot カラムを追加します。

このカラムはまだ新しいので、既存の行については空(NULL)で始まることになります。

ALTER文は他のテーブルデータには一切影響を及ぼしません。

ALTER TABLE guitarwars

これから変更するテーブル名を ALTER TABLE に引き続いで記述します。

ADD COLUMN は新しいデータカラムをテーブルに追加する場合に使います。

ADD COLUMN screenshot varchar(64)

新しいカラムの名前とデータ型を SQL 問い合わせ文の最後に指定します。64 文字あれば、たいていのイメージファイルを収容するのには十分でしょう。もちろん、より安全を期してもっと長い領域を確保しても構いません。

DESCRIBE guitarwars

この文はテーブルの構造、カラム名とそのデータ型のすべてを表示します。

第1ステップ
終了です！

終了

① ALTER を使ってテーブルに screenshot カラムを追加する。



試運転

guitarwarsテーブルに screenshotカラムを追加する。

MySQLツールを使って、ALTER文を実行し、guitarwarsテーブルに screenshotカラムを追加します。次にDESCRIBE文を発行してテーブル構造を覗いて見て、カラムがちゃんと追加されたかどうかを確認します。

DESCRIBE文を発行して新しく作った screenshotカラムを表示させます。

```
C:\Program Files\VertigoServ\mysql\bin\mysql.exe
mysql> DESCRIBE guitarwars;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra       |
+-----+-----+-----+-----+-----+-----+
| id    | int(11) | NO   | PRI | NULL    | auto_increment |
| date  | timestamp | NO   |     | CURRENT_TIMESTAMP |           |
| name  | varchar(32) | YES  | MUL | NULL    |           |
| score | int(11)  | YES  | YES | NULL    |           |
| screenshot | varchar(64) | YES  | YES | NULL    |           |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.01 sec)

mysql>
```

元々のguitarwarsテーブルはギターウォーズのサンプルコードをダウンロードして、guitarwars.sqlというファイルにあるSQL問い合わせ文を実行すれば作れます。

素朴な疑問に答えます

Q: ALTERで追加された新しいカラムはデータベーステーブルの最後にくっつけることしかできないですか？

A: そんなことはありません。どこにでも入れられます。ただ覚えておいて欲しいのは、テーブルの中でカラムの位置というのは大して重要ではないということです。言い方を変えれば、問い合わせの結果、データをどのような順序にでも好きなように並べ替えて構成することができます。でも感覚的に構造的と思う順序で特定のカラムを並べたいということもあるでしょう。このような場合、カラムを絶対位置（先頭）に追加したいかもしれません。これを行うにはALTER問い合わせ文に FIRSTを組み合わせればよいのです。または「AFTER カラム名」を用いて別のカラムの相対位置（後ろ）に置くこともできます。

ALTER TABLE guitarwars
ADD COLUMN age TINYINT AFTER name

新しいカラム位置を指定しない場合は、デフォルトでテーブルの最後にカラムを置くことになります。

Q: 新しく screenshotカラムを追加した後、ハイスクアデータベースにすでにあったデータの行はどうなるのですか？

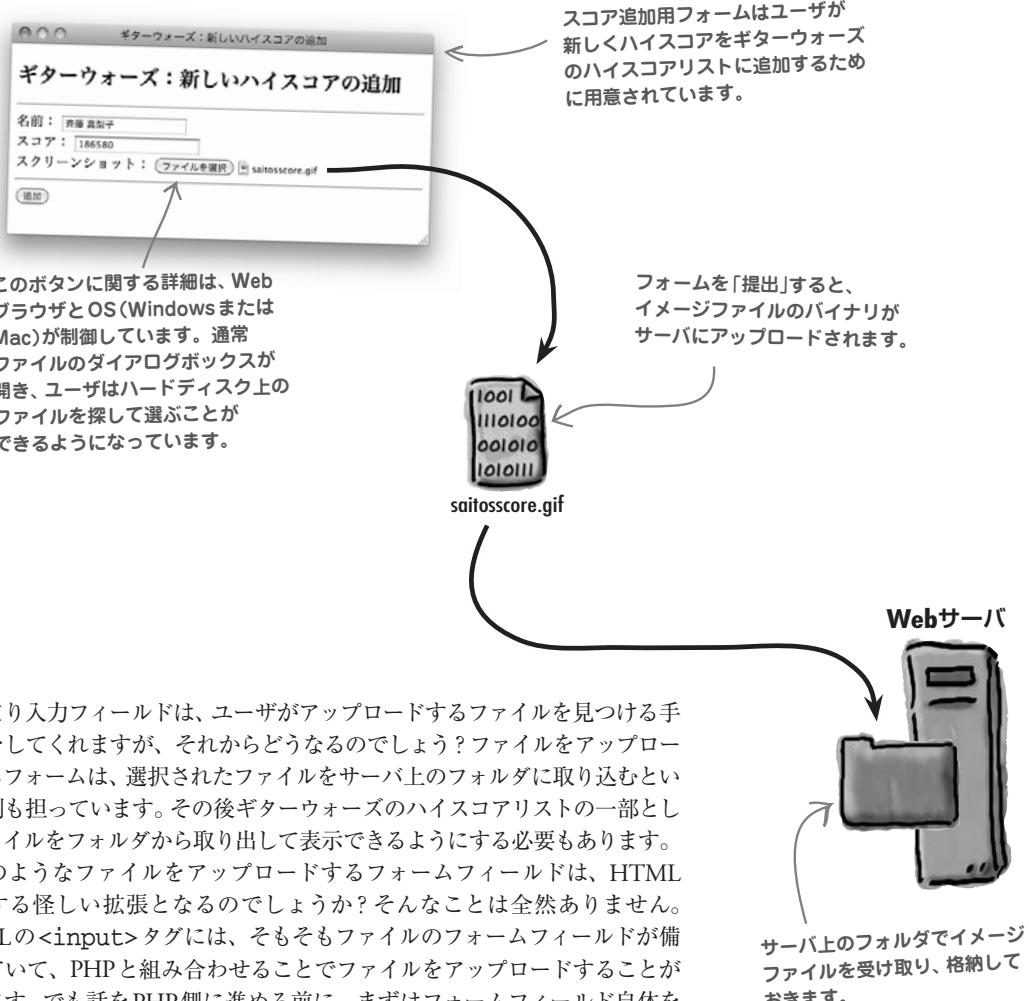
A: ALTER文はデータベースの構造に影響を与えるだけなので、新しく作った screenshotカラムについては、ハイスクアの既存の行は空となります。新しい行を追加して screenshotカラムに何かを入れることはできますが、既存の行の screenshotカラムはすべて相変わらず空のままでです。

Q: スクリーンショットのファイル名を既存の行に追加することはできますか？

A:もちろん、間違いなくできます。このためにはUPDATEというSQL文を使うことになります。手でWebサーバにイメージファイルをアップロードし、UPDATEを使って既存のスコアにスクリーンショットのファイル名を入れることを止めるのは誰もできません。でも今考えているのは、ユーザがイメージファイルを「提出」することです。ユーザが自分のスクリーンショットのイメージをアップロードできるようにすることを考えているのです。このようなことができるようになるには、ユーザがイメージ強化版改良の施されたスコア追加用スクリプトを使うしかなく、それを今作っている最中です…

ユーザからイメージをもらうにはどうすればよい？

ハイスコアデータベースに新しいカラムが追加されたので、ユーザがイメージファイルをアップロードするためにはどうすればよいかに集中することができます。でも實際にはどうやるのでしょうか？FTP？それともテレパシー？これを解決するにはもう一度スコア追加用フォームに戻って、ユーザがフォームフィールドを使ってアップロードするイメージファイルを選べるようにしてあげる必要があります。



つまり入力フィールドは、ユーザがアップロードするファイルを見つける手助けをしてくれますが、それからどうなるのでしょうか？ファイルをアップロードするフォームは、選択されたファイルをサーバ上のフォルダに取り込むという役割も担っています。その後ギターウォーズのハイスコアリストの一部としてファイルをフォルダから取り出して表示できるようにする必要があります。

このようなファイルをアップロードするフォームフィールドは、HTMLに対する怪しい拡張となるのでしょうか？そんなことは全然ありません。HTMLの<input>タグには、そもそもファイルのフォームフィールドが備わっていて、PHPと組み合わせることでファイルをアップロードすることができます。でも話をPHP側に進める前に、まずはフォームフィールド自体をよく見てみることにしましょう…



スコア追加用フォームをよく見る

このフォーム属性は、フォームが特別なエンコーディングを使ってファイルをアップロードする必要があることを示しています。フォームが「提出」されたときにPOSTデータがどのように詰め込まれて送られるかに関係しています。

アップロードするファイルの最大サイズを決めています。この場合32KB(32,768バイト)です。

```
<form enctype="multipart/form-data" method="post" action=<?php echo $_SERVER['PHP_SELF']; ?>>

<input type="hidden" name="MAX_FILE_SIZE" value=<?php echo GW_MAXFILESIZE; ?>" />

<label for="name">名前:</label>

<input type="text" id="name" name="name" value=<?php if (!empty($name)) echo $name; ?>" /><br />

<label for="score">スコア:</label>

<input type="text" id="score" name="score" value=<?php if (!empty($score)) echo $score; ?>" /><br />

<label for="screenshot">スクリーンショット:</label>

<input type="file" id="screenshot" name="screenshot" /> ←

<hr />

<input type="submit" value="追加" name="submit" />

</form>
```

これは自己参照
フォームです。

実際のファイル入力フィールドです。
この部分は結局のところOS(Windows
またはMac)のダイアログに依存して、
ファイルを表示し選ぶことになります。

2

スコア追加用フォームを変更し、ファイル
入力フィールドを使ってイメージファイル
をアップロードできるようにします。

終了

ファイル名 データベースにイメージ▼をぶち込む

単純にイメージファイルをWebブラウザのフォームを通してアップロードしただけでは不十分です。ファイル名はScreenshotカラムにぶち込んで、イメージにアクセスして表示できるようにする必要があります。現状、スコア追加用スクリプトすでに新しいハイスコアをguitarwarsテーブルに挿入しています。これにはSQLのINSERT文を使っていますが、この文には新しく作ったScreenshotカラムが含まれていません。

イメージファイル名は
INSERT文の一部
としてデータベースに
ぶち込みます。

MySQLのNOW()関数は現在の
日時を挿入するのに使います。

```
INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score')
↑
idカラムはAUTO_INCREMENTにより自動的に  
セットされます。この0は無視されますが、問い合わせ文のこの  
場所に値が必要なので書いてあります。
```

このSQL文はそれぞれのカラム名を指定せずに値を設定しているため、すべてのカラムに値を与えなければなりません。ところでつい先ほど新しいカラムを追加しましたね。ということは、この問い合わせ文はもう正しく動きません。新しく追加したScreenshotカラムに対する値がないからです。そこでスクリーンショットのイメージファイル名をデータベースの新しいハイスコア行の一部として加える必要があります。この新しい値をINSERT文に加えると次のようになります。

```
INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$Screenshot')
```

Screenshotカラムを追加する前に挿入されたデータ行は、
スクリーンショットのファイル名がありません。

id	date	name	score	Screenshot
1	2009-09-12 14:37:34	岡野 一郎	127650	
2	2009-09-12 21:27:54	上野 正夫	98430	
3	2009-09-13 09:06:35	大谷 淳	345900	
4	2009-09-13 09:12:53	深沢 麻子	282470	
5	2009-09-13 09:13:34	西原 孝俊	368420	
6	2009-09-13 14:09:50	山口 裕子	64930	
7	2009-09-14 08:13:52	斎藤 真梨子	186580	saitosscore.gif

新しいINSERT文によってスクリーンショットの
ファイル名がScreenshotカラムに挿入されました。

スクリーンショットのイメージ
ファイル名をINSERT文で渡せば、
そのままデータベースに追加
してくれます。

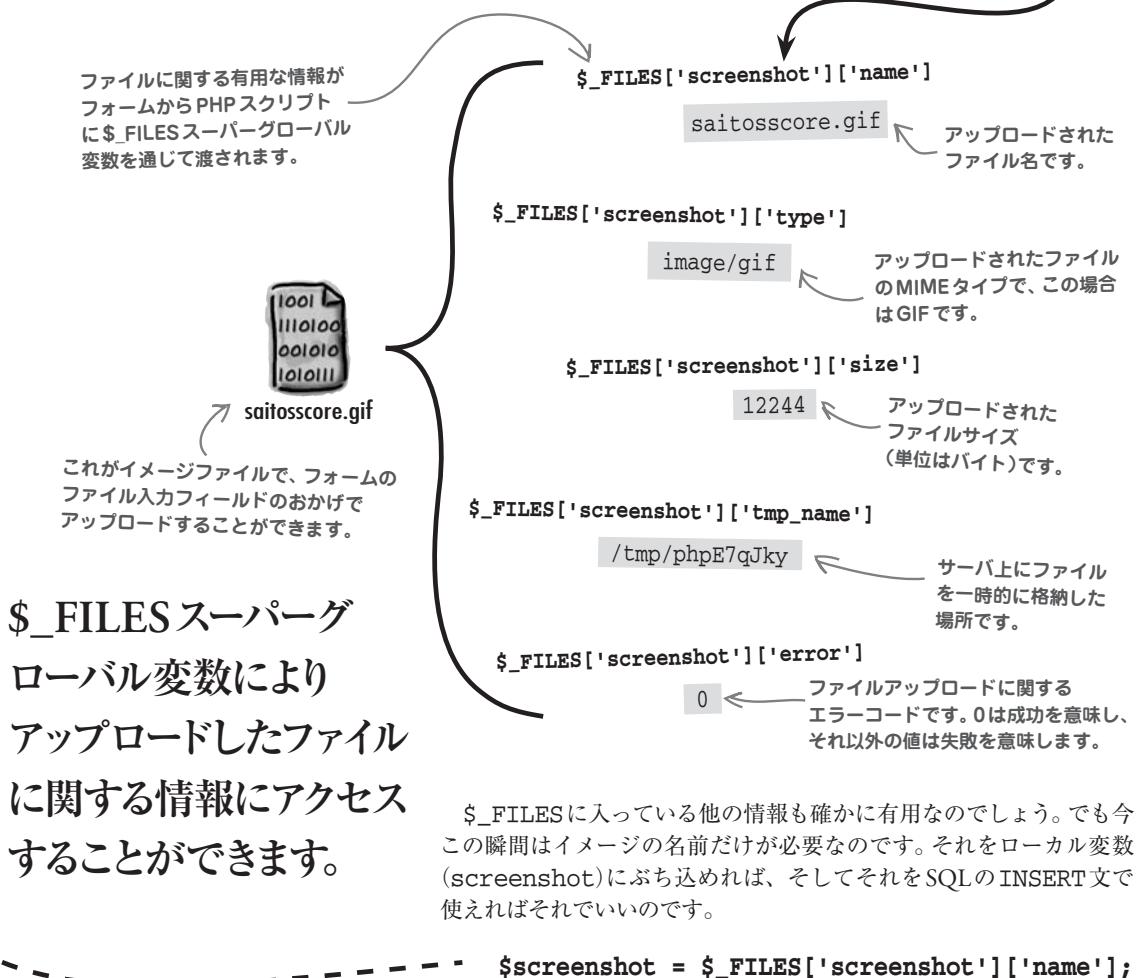
これらの値の順序は非常に
重要で、INSERT文はテーブルの
カラムの順序と値の順序が
同じであると仮定しています。

終了

③スクリーンショットのイ
メージファイル名をテ
ーブルのScreenshotカラムに
INSERTする問い合わせ文を書きます。

アップロードされたファイルの名前を見分ける

さっきの問い合わせ文は良さげですが、実際のイメージファイル名が何というのかを相変わらず知らないのです。フォームのファイル入力フィールドからファイル名にアクセスする何らかの方法があるだろうと思うのは的を射ています。でもどうやって？ 答えはPHPのスーパーグローバル変数\$_FILESが握っています。これはフォームデータにアクセするときにお世話になった\$_POSTスーパーグローバルと似ています。\$_POSTと同様\$_FILESは配列ですが、中身はアップロードされたファイル名だけでなく、他にもファイルに関する有用と思われる情報が入っています。



アップロードしたファイルはどこへ行った？

スコア追加用フォームを使って、ユーザは自身のパソコンからファイルを選びますが、そのファイルは実際にはサーバ上の一時フォルダにアップロードされます。この一時フォルダはサーバ上に自動的に作られますが、通常文字と数字とをランダムに複数組み合わせてできた変な名前です。

このため、index.phpの中にタグを用いて以下のようにコーディングしたとしてもアップロードされたファイルを参照することはできないです。なぜなら、このコーディングはイメージがWebのメインフォルダに置いてあるという想定していることになるからです。

```

```

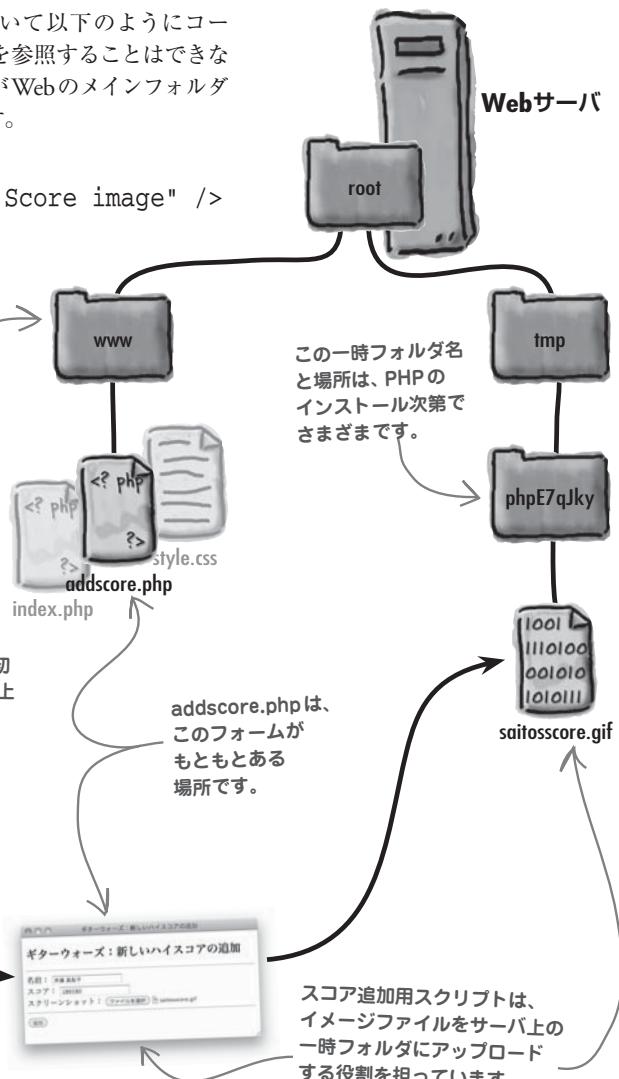
このコードはイメージがWebのメインフォルダに置いてあると想定しています。メインフォルダというのはPHPファイルが置いてあるところです…でもイメージはここにはありません！

クライアント
Webブラウザ

スクリーンショットのイメージファイルは、最初はユーザのコンピュータ上のどこかのフォルダにあります。

mypics

saitosscore.gif





イメージを暗号的な一時フォルダに
ぶち込むというのは、無駄な努力っていう
感じだな。きっとアップロードしたファイルを
突っ込む場所をコントロールできるんでしょ？

その通りです！アップロードファイルをぶち込む場所をPHPでコントロールできます。

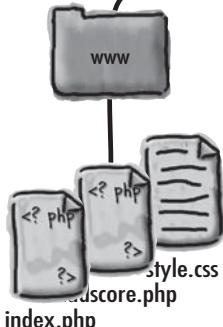
でもPHPでアップロードしたファイルを最初に格納する場所は、コントロールできません。このためその場所は一時的とみなされるのです。しかしアップロードが済んだ後であれば、ファイルを別の場所に動かすことができます。PHP関数move_uploaded_file()は、ファイルの移動前と移動後の場所をパラメタとして取り、移動の重責を担ってくれます。

```
move_uploaded_file( $_FILES['screenshot']['tmp_name'] , $target );
```

ここは移動前のイメージ
ファイルです。一時パス名と
ファイル名を含めます。

Webサーバ

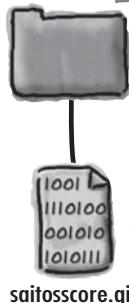
ここは移動後の
イメージファイル
です。一時的
ないパス名と
ファイル名を
含めます。



このフォルダはWebサーバ上の
どんなフォルダでも構いません。
ただしファイルを書き込む権限が
なければならない点に注意して下さい。



ファイルは
一時フォルダ
から通常の
フォルダに
移動します。



move_uploaded_file()



アップロードされたイメージファイル用に家を作る

アップロードされたファイルは専用の場所に入れておくのが慣例となっています。イメージファイルのためのフォルダなので `images` という名前のフォルダにしましょう。`images` フォルダと言っても Web サーバ上の他のフォルダと全く同じものです。唯一の違いは、そのアプリケーション用の Web メインフォルダ配下のどこかになければならないという点だけです。通常の場合だとフォルダは Web フォルダの直下に置いて問題ありません。しかしお望みとあらばもっと複雑な階層構造を持たせることもできます。

`images` フォルダを Web サーバ上で Web メインフォルダの直下に掘ったすると、PHP スクリプト内部からイメージファイルへの参照が可能となります。こんな感じです。

```
$target = GW_UPLOADPATH . $screenshot;
```

`images/phizsscore.gif`

`$target` のパス名は、新しい定数を使って組み立てます。この定数は今からスクリプトに追加するもので `GW_UPLOADPATH` という名前になります。この定数には、`images` フォルダへのパス名を持たせることにします。変数と同様に定数にもデータを入れておくことができます。しかし定数の場合は一度値が設定されたら二度と変更することはできません。イメージファイル名は、スコア追加用フォームに入力されたものが、ここで `images` フォルダのパス名とくっつけられます。

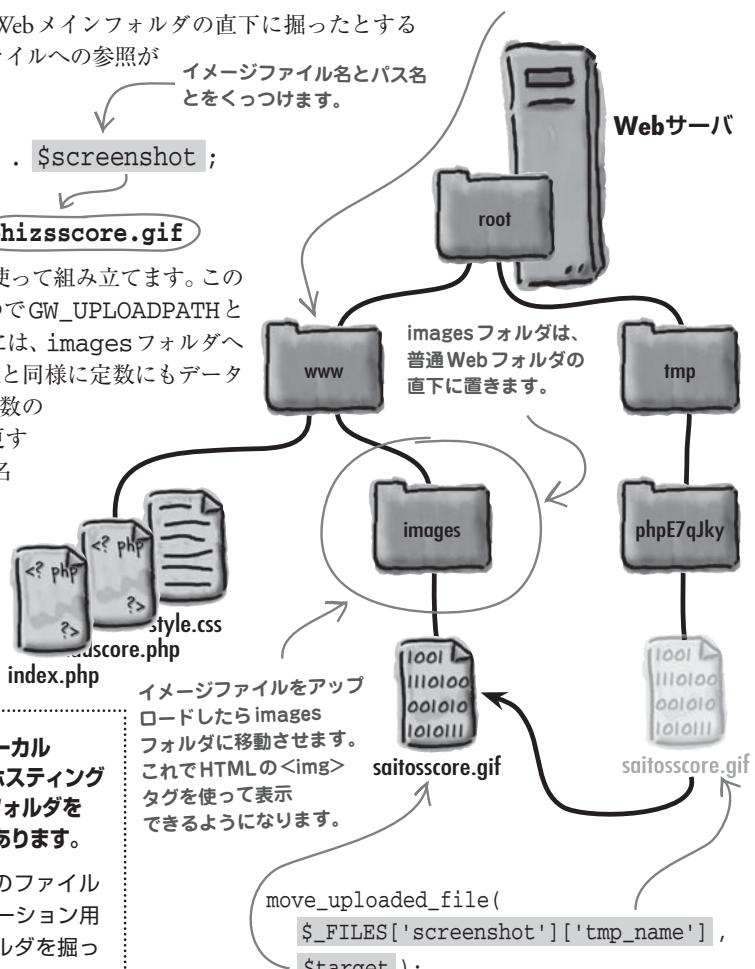


要注意

PHP アプリケーションがローカルコンピュータ以外の場所にホスティングされている場合、`images` フォルダを掘るには FTP を使う必要があります。

FTP プログラムを使って Web サイトのファイルシステムにアクセスし、このアプリケーション用の Web フォルダ直下に `images` フォルダを掘つて下さい。

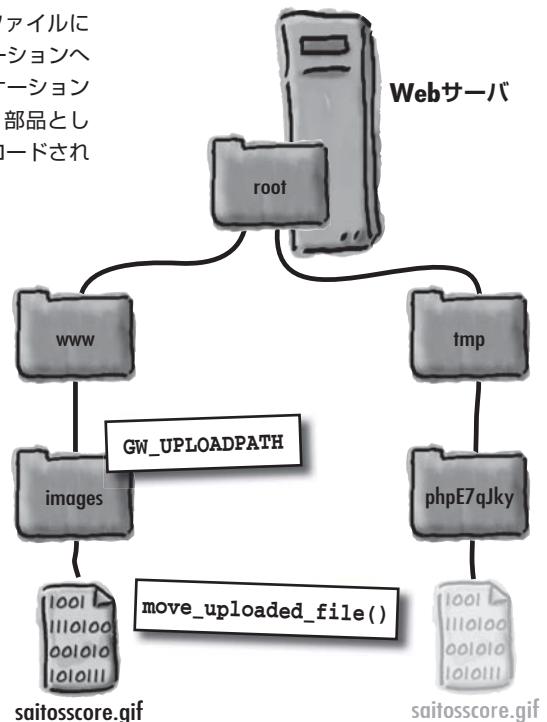
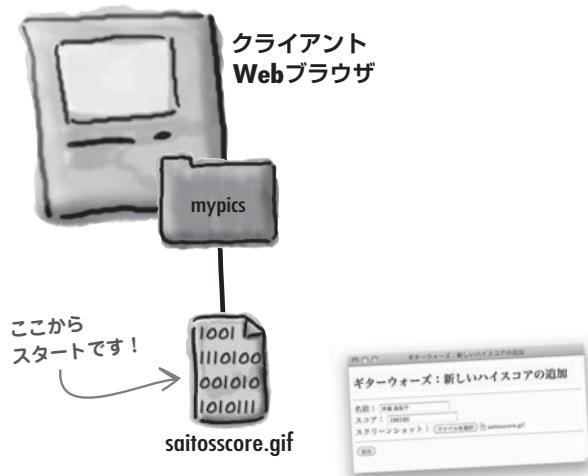
これがアプリケーションの Web フォルダです。ここに PHP スクリプトが（もちろん `index.php` も）ぶち込んであります。





アップロードされたイメージファイルになったつもりで

アップロードされたスクリーンショットのイメージファイルになったつもりで、ギターウォーズアプリケーションへのパスを矢印で表して下さい。アプリケーションの各部わからのパスを描いて下さい。部品としてのデータベースも忘れずに。アップロードされたファイルの身になって考えて下さい！



```
$screenshot = $_FILES['screenshot']['name'];
```

```
INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$screenshot')
```

guitarwars

<i>id</i>	<i>date</i>	<i>name</i>	<i>score</i>	<i>screenshot</i>
1	2009-09-12 14:37:34	岡野 一郎	127650	
2	2009-09-12 21:27:54	上野 正夫	98430	
3	2009-09-13 09:06:35	大谷 淳	345900	
4	2009-09-13 09:12:53	深沢 麻子	282470	
5	2009-09-13 09:13:34	西原 孝俊	368420	
6	2009-09-13 14:09:50	山口 裕子	64930	
7	2009-09-14 08:13:52	斎藤 真梨子	186580	saitosscore.gif

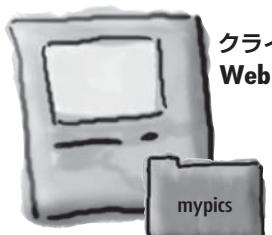


アップロードされたイメージファイルになつたつもりでの答え

アップロードされたスクリーンショットのイメージファイルになつたつもりで、ギターウォーズアプリケーションへのパスを矢印で表して下さい。アプリケーションの各部わからぬパスを描いて下さい。部品としてのデータベースも忘れないで。

アップロードされたファイルの身に

なつて考えて下さい！



クライアント
Webブラウザ

このフォルダはユーザのコンピュータ上にあります。何という名前かとか、どこにあるかとかいったことを一切コントロールできませんし、また気にする必要もありません。

このフォルダの名前と場所は一番気にする必要があります。なぜならギターウォーズでアップロードしたイメージファイルをぶち込んだり、覗いてみたりするのは正にここだからです。

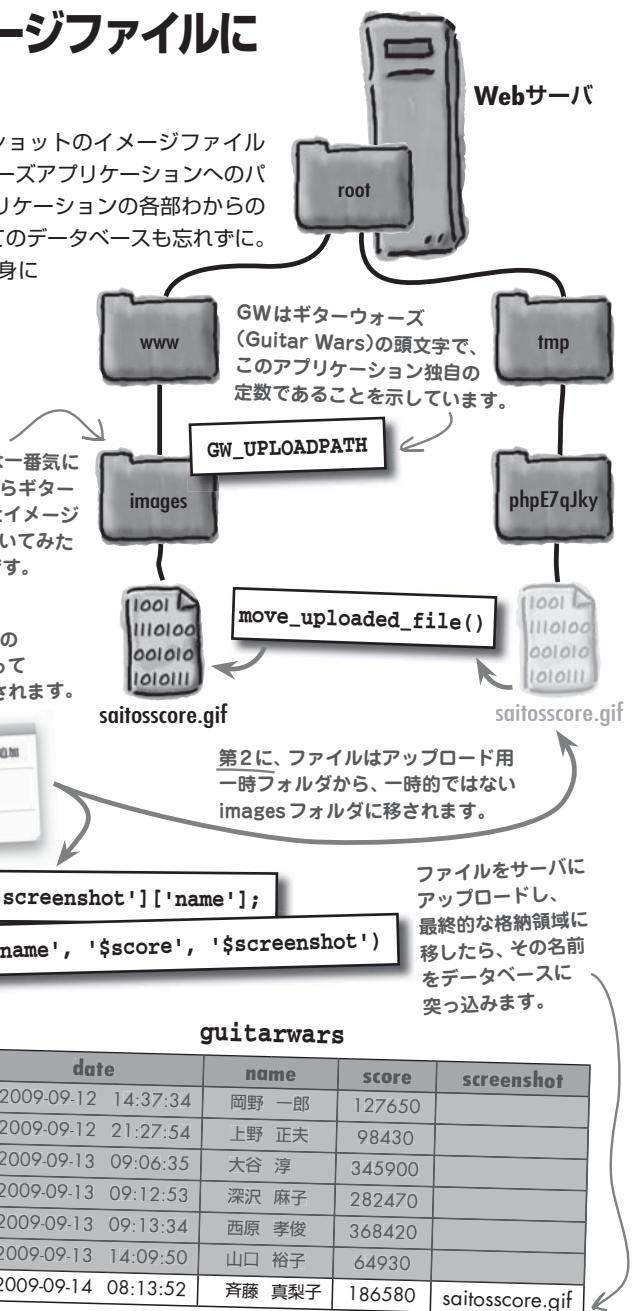
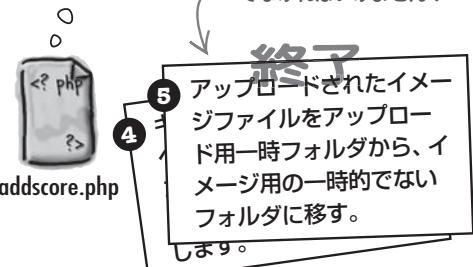
まず第1に、ファイル入力のフォームフィールドを使ってファイルがアップロードされます。

`$screenshot = $_FILES['screenshot']['name'];`

`INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$screenshot')`

やつたあ！

そうです。これは以前の計画にはなかった新しいステップです。設計はいつも柔軟でなければいけません！





試運転

アップロードされたスクリーンショットのイメージに対して、イメージフォルダ中に終の棲家を与える。

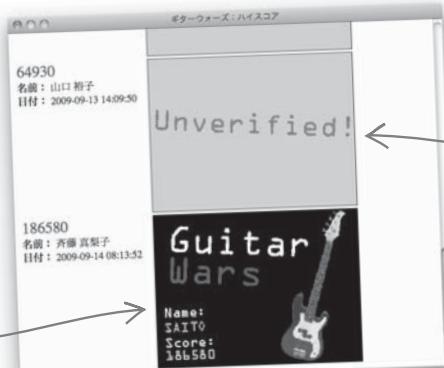
addscore.phpスクリプトを変更し、定数GW_UPLOADPATHを使ってアップロードされたスクリーンショットのイメージを、その定数が指示するパスにぶち込みます。これが変更すべきコードの要所です。

```
<?php
// Define the upload path and maximum file size constants
define('GW_UPLOADPATH', 'images/');
// Grab the score data from the POST POSTからスコアデータの取り出し
$name = $_POST['name'];
$score = $_POST['score'];
$screenshot = $_FILES['screenshot']['name'];
if (!empty($name) && !empty($score) && !empty($screenshot)) {
    // Move the file to the target upload folder
    $target = GW_UPLOADPATH . $screenshot;
    if (move_uploaded_file($_FILES['screenshot']['tmp_name'], $target)) { ファイルをアップロード用フォルダへ移動
        // Connect to the database
        $dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit', 'gwdb');
        // Write the data to the database
        $query = "INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$screenshot')"; データをデータベースへ書き込み
        mysqli_query($dbc, $query);
        // Confirm success with the user
        echo '<p>新しいハイスクアを追加していただきありがとうございます！</p>';
        echo '<p><strong>名前:</strong> ' . $name . '<br />';
        echo '<strong>スコア:</strong> ' . $score . '<br />';
        echo '</p>';
        echo '<p><a href="index.php">&lt;&lt; ハイスコアリストへ戻る</a></p>';
    }
}
// Connect to the database
$dbc = mysqli_connect('www.guitarwars.net', 'admin', 'rockit', 'gwdb');
// Write the data to the database
$query = "INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$screenshot')"; データをデータベースへ書き込み
mysqli_query($dbc, $query);
// Confirm success with the user
echo '<p>新しいハイスクアを追加していただきありがとうございます！</p>';
echo '<p><strong>名前:</strong> ' . $name . '<br />';
echo '<strong>スコア:</strong> ' . $score . '<br />';
echo '</p>';
echo '<p><a href="index.php">&lt;&lt; ハイスコアリストへ戻る</a></p>';
echo '?>
```



addscore.php

index.phpスクリプトも定数GW_UPLOADPATHの影響を受けます。こっちも忘れずに同様の修正が必要です。以上の修正を行ったら、これら2つのスクリプトをサーバにアップロードし、ハイスクアの追加をもう一度やってみましょう。



以前のスコアはスクリーンショットのイメージがないので「unverified!」というイメージを表示します。

アップロードされたスクリーンショットのイメージが、今度はメインページにちゃんと表示されます。

素朴な疑問に答えます

Q: `php.ini` ファイルでアップロードしたファイルの格納領域をコントロールできるのに、なぜファイルを移す必要があるのでしょうか？

A: なぜなら `php.ini` を常に変更できるとは限らないからです。例えば、PHP アプリケーションを仮想 Web サーバ上で構築しているとしたら、ほとんどの場合、`php.ini` のセッティングを変更することは不可能です。もし仮に `php.ini` を変更できたとしても、アプリケーションはリスクを負うことになります。つまりサーバが変わる度に同じ問題に直面することになるのです。言い方をえらばれば、アプリケーションは `php.ini` にコントロールされたパスに依存してしまうのです。逆にファイルを移しておけば、PHP コードだけでコントロール可能なパスにしか依存しないのです。

Q: 日時をギターウォーズにユーザが入力できるようにしないのは何故ですか？

A: 日時というのは、ハイスクアの中で非常に重要な地位を占めています。なぜならスコアをサイトに公式にポストしたのはいつなのかを表しているからです。他のあらゆる記録と同様、あるスコアを最初に達成した人に栄誉が与えられるべきです。ユーザが入力したハイスクア達成日時を信用するより、投稿した日時をもってスコアの公式記録としたほうがよいのです。これによりインチキの日付を排除でき、ハイスクアリストの信頼性も高まります。この手のコンペ系アプリケーションのユーザは、常に抜け道を探していますから、できる限りその手のものは排除しておくべきです！

実は `NOW()` 関数は Web サーバの時刻を使用しているということは、知つておくべきことなのです。この時刻はユーザのローカルタイムと同じであるとは限りません。しかし、これは問題にはなりません。なぜならすべてのユーザは同じサーバの時刻に張り付いていることになるからです。

Q: 同じ名前のファイルをアップロードすれば、誰でも別の人のスクリーンショットのイメージを上書きしちゃうんじゃないですか？

A: その通りです。問題は、スクリーンショットのイメージを、ユーザがアップロード用フォームフィールドに書き込んだファイル名のまま Web サーバ上で使ってしまっていることによるものです。このため 2 人で同じファイル名のイメージファイルをアップロードすると、先にアップロードしたユーザのイメージは後からアップロードしたユーザのイメージで上書きされてしまいます。これではダメです。1 つの解決策は、何らかの一意性をサーバ上のイメージファイル名に与えることです。シンプルな方法はサーバのその時点での時刻を秒単位まで含めて付加することです。こんな感じになります。

```
$target = GW_UPLOADPATH . time() .  
$Screenshot;
```

このコードを実行するとファイル名は、
`saitosscore.gif` から `1221634560saitosscore.gif` といった具合に
変わります。ここで `1221634560` というのは、

サーバの時刻を秒単位で表現したものです。

Q: ハイスコアのスクリーンショットとしてアップロードされた実際のイメージデータをギターウォーズのデータベースにぶっ込んで大丈夫でしょうか？

A: 大丈夫です。データベースというのには非常に汎用性があり、バイナリデータをぶっ込むことができます。ただし、このようにした場合に大問題となることがあります。ギターウォーズではアップロードされたイメージを HTML のコードとして使うのです。メインページ `index.php` 上に表示するのです。HTML の `` タグは Web サーバ上にぶち込まれたイメージファイルを参照するのであって、データベースにぶっ込まれたバイナリの塊ではないのです。ですから `guitarwars` テーブルを変更してバイナリイメージを格納できるようにしたとしても、その後で HTML コードでそれを表示するためにデータのフォーマットを元に戻す、という巨大な壁が立ちはだかっています。

`time()` 関数から返ってくる時間は、ちっとも恐ろしいものではありません。一意の数字を返してくれるだけです…返ってくる値はどんどん大きくなる一方です！



重要ポイント

- `ALTER` 文は、MySQL データベーステーブルの構造を変えるために使います。例えば新しいデータカラムの追加などです。
- PHP と MySQL がちょっと手助けしてあげるだけで、HTML の `<input>` タグはイメージファイルをアップロードするのに使えます。
- スーパーグローバル変数 `$_FILES` は、アップロードされたファイルに関する情報を入れておく場所です。
- PHP の標準関数 `move_uploaded_file()` は、Web サーバ上のどこかにあるファイルを動かすのに使うことができますが、アップロードされたファイルを処理するには非常に重要な役割を果たします。
- たいていの Web アプリケーションは、`images` フォルダを持つことでより恩恵に浴しています。ここにはアプリケーションが使うイメージを入れますが、特にユーザがアップロードしたものを入れるとよいでしょう。

データベースは、テキストデータをぶっ込むには優れていますが、バイナリデータについては、外部ファイルへの参照を入れておいた方が良いでしょう。



ファイルをアップロードするパスを定数にしておくのは良いと思うんだけど、2箇所に作ることになるんじゃない？ index.php と addscore.php の両方だよ！ パスが変わったらどうなるの？

定数 `GW_UPLOADPATH` は、スクリーンショットのイメージ用のファイルをアップロードするパスが設定してあります。

```
define('GW_UPLOADPATH', 'images/');
```

`define()` は定数を作るので使う。

定数の名前です。

定数の値です。もう変えられません…定数ですから！

もしパスが変わってしまったら、コードの2箇所を変えなければなりません…コードの重複は愚の骨頂です！

スクリプト `index.php` と `addscore.php` の両方それぞれに定数 `GW_UPLOADPATH` がってもちゃんと動きます。でも定数が両方のスクリプトに重複していますから、パスに関するいかなる変更があった場合も、両方のスクリプトを書き換える必要があります。このようなコードの重複は、愚かな設計と言わざるを得ないので、あつたら必ず抹殺するべきです。

```
// Define the upload path constant アップロード用パスの定義
define('GW_UPLOADPATH', 'images/');
```



`index.php`

定数が2箇所で作られています。つまり2つの異なる場所をメンテナンスしなければならないことになります。



脳力発揮

重複コードの問題を解決するには、定数 `GW_UPLOADPATH` を1箇所だけに入れておく必要があります。`index.php` と `addscore.php` のどちらに入れますか？なぜそう思いますか？

```
// Define the upload path and maximum file size constants アップロード用パスと最大ファイルサイズ定数の定義
define('GW_MAXFILESIZE', 32768); // 32 KB
define('GW_UPLOADPATH', 'images/');
```



`addscore.php`

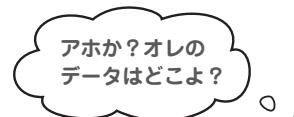
共有データは共有しなければならない

1つのアプリケーションでデータを複数のスクリプトにまたがって共有する場面に出くわしたら、データを1箇所に固めておいて、いくつもの異なるスクリプトで引っ張り抜いて使う方法が必要になります。でもそれでは相変わらず「そのデータをどこに本当に置けばいいのか?」という質問の答えにはなっていません…

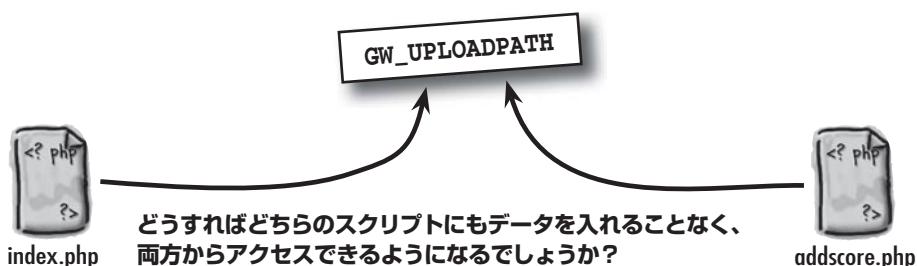
`index.php`だけにデータを置いたとすると…



…でもそうすると他のスクリプトからアクセスできなくなってしまいます。



スクリプト共有のデータを既存のスクリプトファイルに入れたのでは、全然動きません。なぜなら、そんなことをしたらデータは共有されなくなってしまうからです。答えは、複数のスクリプトからアクセスできるような何らかのものを作ることにより達成できます。直接どれか1つに入れるとダメです。

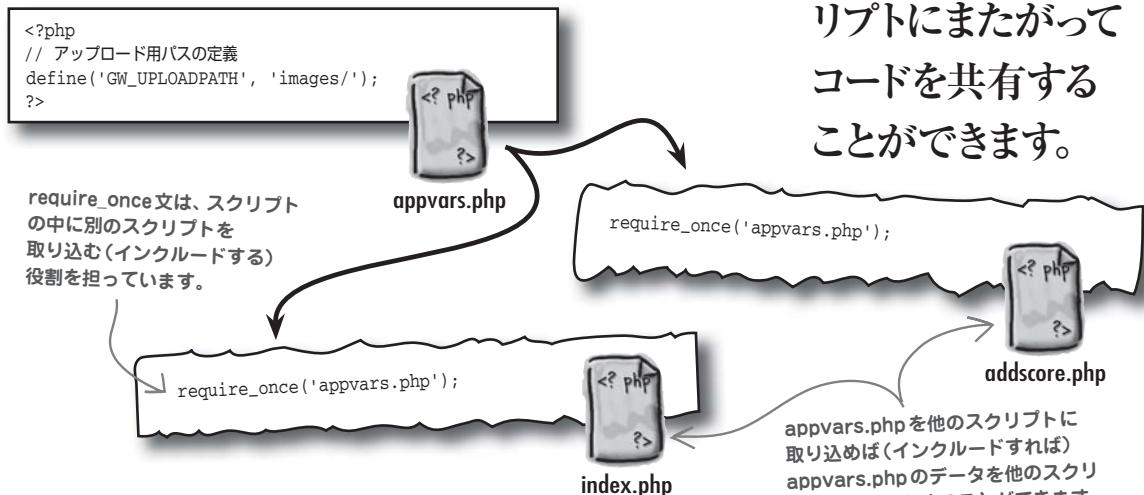


スクリプトデータを共有するための答えは、インクルードファイル[†]にあります。これはPHPのソースコードファイルですが、他のPHPファイルが必要に応じて取り込むことができるという類のものです。

[†] 訳注：ヘッダファイル(header files)と言うこともあります。インクルードファイル(include files)と同じものを表します。

共有スクリプトデータはrequire[†]する

インクルードファイルが非常に強力なのは、1つ作れば必要に応じて他のスクリプトで使い回しが効くことです。効率的にコードを共有することができます。定数GW_UPLOADPATHはインクルードファイルの中に置いておけば、「アプリケーション変数」をひとまとめにしておくことができます。



素朴な疑問に答えます

Q： ちょい待ち。アプリケーション「変数」って、ホンマは定数ちゃうん？

A： まあ、その通りですね。でも気にしなくて大丈夫です。今大事なのは変数か定数かの分け目を決めるこではありません。大事なのは、アプリケーションの中で共有データをぶち込む共通の場所を確立することです。そしてその場所が、スクリプトファイルであってappvars.phpという名前にしただけです。

Q： 共有スクリプトファイルに入れるコードはデータだけに制限せなあかんの？

A： そんなことは全くありません。PHPコードなら何でもスクリプトファイルに入れとくことができ、require_once文を使えば共有できます。実際、アプリケーションで複数のスクリプトファイルにまたがってたくさんの関数コードを共有するというのは、全然珍しくありません。共有スクリプトファイルを利用するというのは、普通に行われていることというよりも、チームでコードを構成する上で、すばらしいやり方と言った方が良いでしょう。

インクルードファイルを使えば複数のスクリプトにまたがってコードを共有することができます。

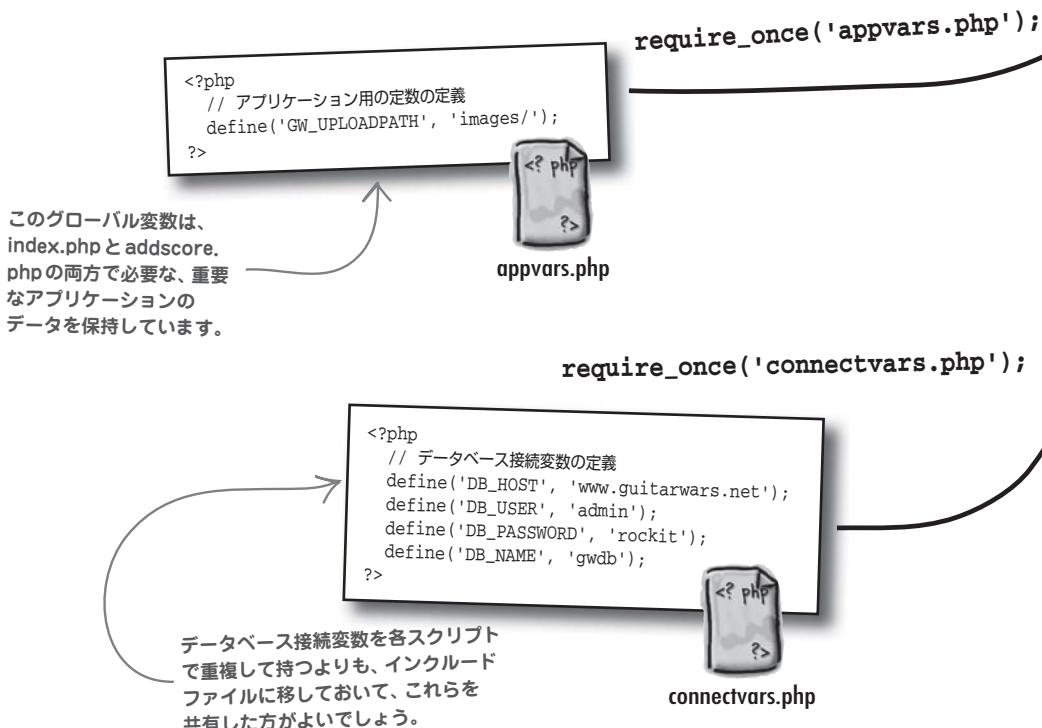
Q： スクリプトコードを取り込む（インクルードする）PHP文やのに、なんでrequire_onceと言うん？

A： 「インクルードファイル」という名前は、includeというPHP文から来ています。これはrequire_onceに非常によく似ています。違いは、インクルードファイルが見つからなかった場合にrequire_onceがエラーを吐くことです。includeの方は、インクルードファイルがなくてもエラーとは言ってくれません。また、require_onceのonce（一度）は、ファイルを間違って二度以上インクルードしてしまっても面倒をみてくれることを意味しています。もしかしたらinclude_onceの代わりにincludeを使って、あまり重要でないコードをインクルードしているのを目的にすることがあるかもしれません。これは、例えば純粋なHTMLコードなど、重大な目的を持っていないコードなのです。なおPHPにはinclude_onceやrequire文といったものもあります。これらは、それぞれrequire_onceやincludeのバリエーションです。

[†] 訳注：requireとは「要求する」という意味です。因みにincludeは「含む」という意味です。

require_once は「挿入」と考える

インクルードは、PHP ファイルの共有だけに限定されているわけではありません。スクリプトの中ならどこに現れても構いません。require_once 文は「挿入」文と考えてもよいでしょう。つまりその部分は、参照されたスクリプトファイルの中身で置き換わるのです。ギターウォーズの場合で言うと、データベース接続変数もインクルードファイルに移すことで恩恵に浴することができそうです。これら 2 つの共有スクリプトファイルの中身は、他のスクリプトファイルに、require された地点で直接挿入されます。



REQUIRE_ONCE 文は、共有のスクリプト
コードを他のスクリプトに突っ込みます。

```

<?php
// アプリケーション用の定数の定義
define('_GW_UPLOADPATH', 'images/');
// データベース接続変数の定義
define('DB_HOST', 'www.guitarwars.net');
define('DB_USER', 'admin');
define('DB_PASSWORD', 'chiefrocker');
define('DB_NAME', 'guitarwarsdb');

// Connect to the database データベースへの接続
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
MySQLからスコアデータの取り出し

// Retrieve the score data from MySQL
$query = "SELECT * FROM guitarwars";
$data = mysqli_query($dbc, $query);

スコアデータの配列をループし、HTMLにフォーマット
// Loop through the array of score data, formatting it as HTML
echo '<table>';
while ($row = mysqli_fetch_array($data)) { スコアデータの表示
    // Display the score data
    echo '<tr><td class="scoreinfo">';
    echo '<span class="score">' . $row['score'] . '</span><br />';
    echo '<strong>名前:</strong> ' . $row['name'] . '<br />';
    echo '<strong>日付:</strong> ' . $row['date'] . '</td>';
    if (is_file(GW_UPLOADPATH . $row['Screenshot'])) &&
        filesize(GW_UPLOADPATH . $row['Screenshot']) > 0) {
        echo '<td></td></tr>';
    }
    else {
        echo '<td></td></tr>';
    }
}
echo '</table>';
mysqli_close($dbc);
?>

```



試運転



index.php

あれ！オレも
共有データに
アクセスしなきゃ。



addscore.php

終了

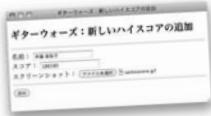
- ⑥ ファイルアップロード用のパスを定数にしてインクルードファイル経由で共有できるようにする。

おおっと、またもや新しいステップ！
事前に計画が困難なことはよくあります。「いつでもどこでも」設計を微調整できるよう準備しておく必要があります。

6

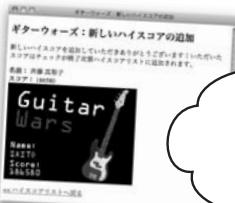
順序 タイミングがハイスコアのすべて

ギターウォーズはついにイメージ強化版となりました。ユーザはスクリーンショットのイメージをアップロードすることで、ハイスコアの証拠を示すことができます。これはアプリケーションにとって重要な改良となりましたが、まだ問題が残っています。ユーザはずっと不満を漏らしているのです。問題なのはメインページ上でのスコアの順序です。

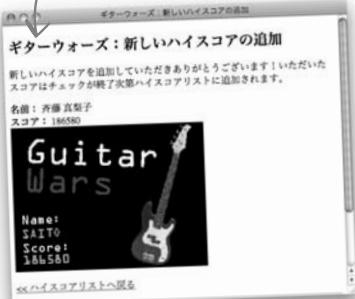


新しくハイスコアを追加すると、スクリーンショットのイメージも取り込まれます…

…これはすごくよさそうですが、スコアが順番に並んでいないことが問題です！



スクリーンショットのイメージ付きでスコアを投稿したのはアタシだけなのに、何でアタシのスコアがリストの一番下なの？



齊藤さんは、自分のスクリーンショットで証拠を示しましたが、相当良いスコアであるにも関わらずハイスコアリストの一番下に埋め込まれていることで、ちょっとムカついています。



そうです。スコアは順番になっていません。スコアが表示される順序は、データベースに突っ込まれた順序になってしまっています。これは完全に無作為です。データベースに突っ込まれた順序に頼るなどということは、決してするべきではありません。まあ順序が本当にどうでもいいような場合は別ですが、今回の場合はダメです。ですから問い合わせ文の結果に何らかの順序を強要する必要があります。SQL文の ORDER BY を使うと、そのような順序付けをしてくれます。



PHP & MySQLマグネット

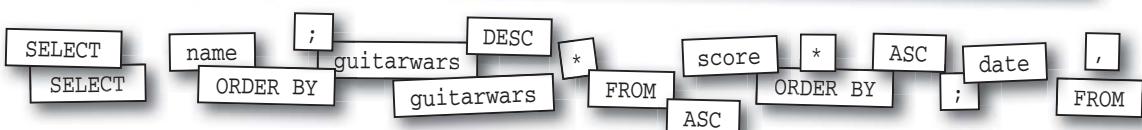
以下のマグネットを使って、ORDER BYがどのように動きをするか考えてみます。順序(order)付きのSELECT文を作って、次のような結果を表示させて下さい。また、ギターウォーズを修正するのに最適な問い合わせ文はどちらだと思いますか？マルで囲んで下さい。ヒント：ASCはASCending(昇順)、DESCはDESCending(降順)を表します。

		<input type="text"/> id	<input type="text"/> date	<input type="text"/> name	<input type="text"/> score	<input type="text"/> screenshot
<input type="checkbox"/>	<input type="pencil"/>	X 1	2009-09-12 14:37:34	岡野 一郎	127650	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 2	2009-09-12 21:27:54	上野 正夫	98430	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 3	2009-09-13 09:06:35	大谷 淳	345900	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 4	2009-09-13 09:12:53	深沢 麻子	282470	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 5	2009-09-13 09:13:34	西原 孝俊	368420	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 6	2009-09-13 14:09:50	山口 裕子	64930	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 7	2009-09-14 08:13:52	斎藤 真梨子	186580	saitosscore.gif

問い合わせ文の結果は、日付の昇順に並べられて返ってきます。

問い合わせ文の結果は、まずスコア数値の降順、
次に日時の昇順に並べられて返ってきます。

		<input type="text"/> id	<input type="text"/> date	<input type="text"/> name	<input type="text"/> score	<input type="text"/> screenshot
<input type="checkbox"/>	<input type="pencil"/>	X 5	2009-09-13 09:13:34	西原 孝俊	368420	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 3	2009-09-13 09:06:35	大谷 淳	345900	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 4	2009-09-13 09:12:53	深沢 麻子	282470	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 7	2009-09-14 08:13:52	斎藤 真梨子	186580	saitosscore.gif
<input type="checkbox"/>	<input type="pencil"/>	X 1	2009-09-12 14:37:34	岡野 一郎	127650	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 2	2009-09-12 21:27:54	上野 正夫	98430	NULL
<input type="checkbox"/>	<input type="pencil"/>	X 6	2009-09-13 14:09:50	山口 裕子	64930	NULL





PHP & MySQL マグネットの答え

下のマグネットを使って、ORDER BYがどのように動きをするか考えてみます。順序(order)付きのSELECT文を作って、次のような結果を表示させて下さい。また、ギターウォーズを修正するのに最適な問い合わせ文はどちらだと思いますか？マルで囲んで下さい。ヒント：ASCはASCending(昇順)、DESCはDESCending(降順)を表します。

SELECT	*	FROM	guitarwars	ORDER BY	date	ASC	;
		← ↑ →	id	date	name	score	screenshot
<input type="checkbox"/>		<input checked="" type="checkbox"/>	1	2009-09-12 14:37:34	岡野 一郎	127650	NULL
<input type="checkbox"/>		<input checked="" type="checkbox"/>	2	2009-09-12 21:27:54	上野 正夫	98430	NULL
<input type="checkbox"/>		<input checked="" type="checkbox"/>	3	2009-09-13 09:06:35	大谷 淳	345900	NULL
<input type="checkbox"/>		<input checked="" type="checkbox"/>	4	2009-09-13 09:12:53	深沢 麻子	282470	NULL
<input type="checkbox"/>		<input checked="" type="checkbox"/>	5	2009-09-13 09:13:34	西原 孝俊	368420	NULL
<input type="checkbox"/>		<input checked="" type="checkbox"/>	6	2009-09-13 14:09:50	山口 裕子	64930	NULL
<input type="checkbox"/>		<input checked="" type="checkbox"/>	7	2009-09-14 08:13:52	斎藤 真梨子	186580	saitosscore.gif

問い合わせ文の結果は、日付の昇順に並べられて返ってきます。

問い合わせ文の結果は、まずスコア数値の降順、
次に日時の昇順に並べられて返ってきます。

これがギターウォーズを直すのに必要な問い合わせ文です。

SELECT	*	FROM	guitarwars	ORDER BY	score	DESC	,	date	ASC	;
		← ↑ →	id	date	name	score	↓	screenshot		
<input type="checkbox"/>		<input checked="" type="checkbox"/>	5	2009-09-13 09:13:34	西原 孝俊	368420	NULL			
<input type="checkbox"/>		<input checked="" type="checkbox"/>	3	2009-09-13 09:06:35	大谷 淳	345900	NULL			
<input type="checkbox"/>		<input checked="" type="checkbox"/>	4	2009-09-13 09:12:53	深沢 麻子	282470	NULL			
<input type="checkbox"/>		<input checked="" type="checkbox"/>	7	2009-09-14 08:13:52	斎藤 真梨子	186580	saitosscore.gif			
<input type="checkbox"/>		<input checked="" type="checkbox"/>	1	2009-09-12 14:37:34	岡野 一郎	127650	NULL			
<input type="checkbox"/>		<input checked="" type="checkbox"/>	2	2009-09-12 21:27:54	上野 正夫	98430	NULL			
<input type="checkbox"/>		<input checked="" type="checkbox"/>	6	2009-09-13 14:09:50	山口 裕子	64930	NULL			

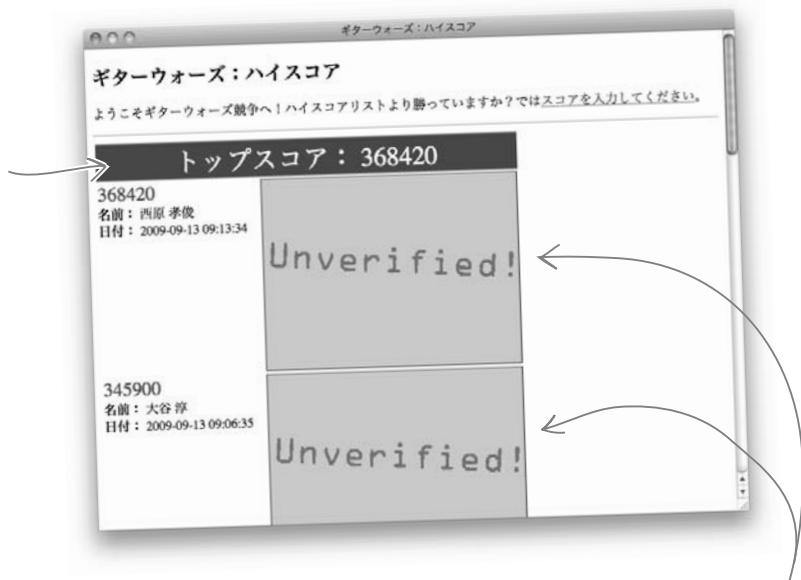
日時による順序は二次的なもので、同一スコア
が2つ以上あった場合にのみ威力を発揮します。
この例ではそのような状況になつていませんが、
十分大きなデータがあれば起こりうることです。

複数レベルの順序
付けを区切るため
にはコンマが必要
になります。

ギターウォーズのトップに勝者の栄誉を与える

スコアの順序を直したので、ハイスクアリストには予期せぬ改良を施すことができてしまいました。リストの先頭には最も高いスコアが来ているのです。ギターウォーズでトップのスコアを獲得した勝者には、トップスコアのヘッダを与えるに値します。ヘッダに最高スコアを表示することで、ギターウォーズの勝者が誰なのか疑問の余地がなくなります…同時にトップを引きずり下ろすためにはどのくらいのスコアが必要かもわかります。

トップスコアヘッダは、明らかにトップスコアを目立たせます。同時に競合するギターウォーズの参加者に目標を提供しています。



素朴な疑問に答えます

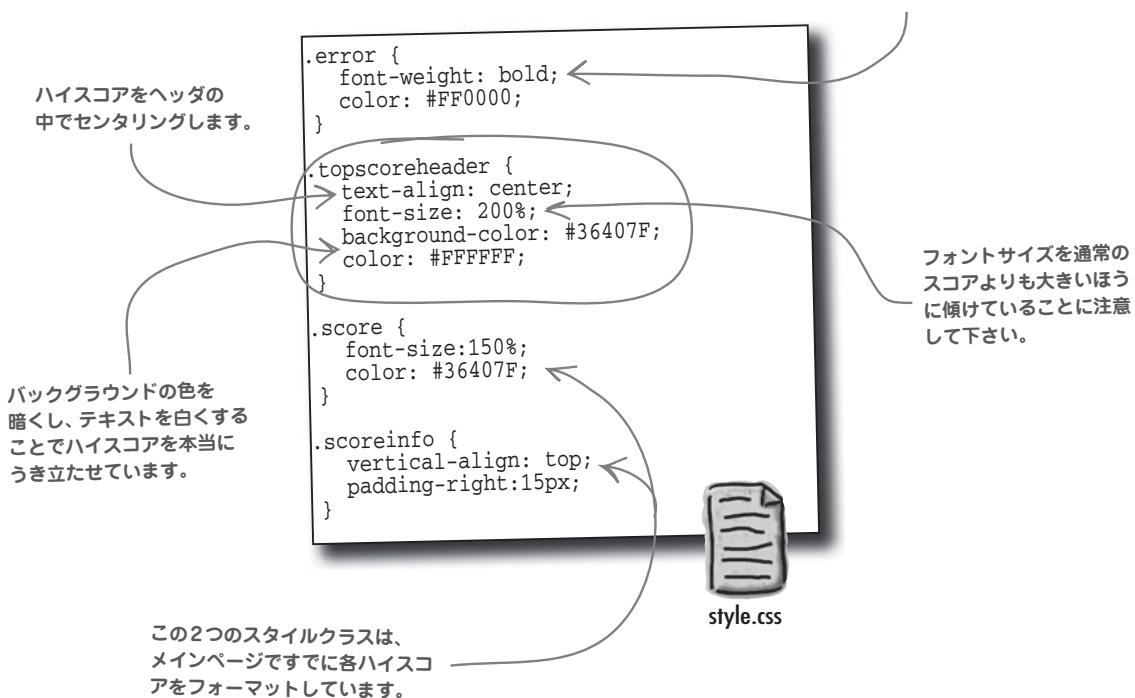
Q: 多くのスコアには相変わらず証拠あらへんやん？
それって問題ちゃうん？

A: 問題です。でもだからといって前に進むことをやめる必要はありません。トップスコアに注意を向けましょう。ハイスクアリストを徐々にきれいにしていく必要があるということを認識しておいて下さい。証拠のないスコアはリストから消去してしまえばよいでしょう。実際、トップスコアを目立たせることが終わったらすぐに、証拠のないスコアへの攻撃にとりかかります。

HTMLとCSSでトップスコアをフォーマットする

新しく作るハイスコアヘッダで最も重要なことは、他のすべてのハイスコアリストの上位によく見えるように配置しなければならないということです。このためHTMLとCSS両方に助けてもらって、見た目のスマートなものにします。ヘッダはHTMLテーブルに行として作られ、ここにCSSの特別なスタイルをあてはめます。このスタイルはtopscoreheaderという名前で、ギターウォーズのスタイルシートstyle.cssに追加しなければなりません。

このスタイルクラスは、スコア追加用スクリプトで投入されたデータにエラーがある場合に、それを目立たせるためにすでに使っています。



index.phpスクリプトは、すでにハイスコアリストを含むHTMLテーブルを生成しています。ヘッダを生成するには、単に1番目のスコアだけを取り出してトップスコアに組み込んでやるだけです。スコアは今や順番に並んでいるので、1番目のスコアはトップスコアであると保障されているのです。whileループがスコア全体をループする役割を担っていて、全スコアについて何かしら回る必要はありませんが、ヘッダについては最初の1つだけを生成します…



ギターウォーズスクリプトindex.phpのコードを仕上げて、フォーマットされたヘッダを追加し、トップスコアを表示するようにして下さい。CSSスタイルシートのtopscoreheaderを使います。ヒント：トップスコアヘッダは、ハイスクアのHTMLテーブルの一部であることを忘れないで下さい。このテーブルには2つのカラムがあります。

```
...
// Loop through the array of score data, formatting it as HTML
echo '<table>';
スコアデータの配列をループし、HTMLにフォーマット
$i = 0;
while ($row = mysqli_fetch_array($data)) {
    // Display the score data スコアデータの表示
    if ( ..... ) {
        -----
    }
    echo '<tr><td class="scoreinfo">';
    echo '<span class="score">' . $row['score'] . '</span><br />';
    echo '<strong>名前:</strong> ' . $row['name'] . '<br />';
    echo '<strong>日付:</strong> ' . $row['date'] . '</td>';
    if (is_file(GW_UPLOADPATH . $row['screenshot']) &&
        filesize(GW_UPLOADPATH . $row['screenshot']) > 0) {
        echo '<td></td></tr>';
    }
    else {
        echo '<td></td></tr>';
    }
    -----
}
echo '</table>';
...
```





エクササイズ の答え

ギターウォーズスクリプトindex.phpのコードを仕上げて、フォーマットされたヘッダを追加し、トップスコアを表示するようにして下さい。CSSスタイルシートのtopscoreheaderを使います。ヒント：トップスコアヘッダは、ハイスコアのHTMLテーブルの一部であることを忘れないで下さい。このテーブルには2つのカラムがあります。

```

...
// Loop through the array of score data, formatting it as HTML
echo '<table>';
$i = 0;
while ($row = mysqli_fetch_array($data)) {
    // Display the score data [スコアデータの表示]
    if ( $i == 0 ) { [スコアデータの表示]
        echo '<tr><td colspan="2" class="topscoreheader">トップスコア：'.
        $row['score']. '</td></tr>'; [スコアデータの表示]
    }
    echo '<tr><td class="scoreinfo">';
    echo '<span class="score">' . $row['score'] . '</span><br />';
    echo '<strong>名前:</strong> ' . $row['name'] . '<br />';
    echo '<strong>日付:</strong> ' . $row['date'] . '</td>';
    if (is_file(GW_UPLOADPATH . $row['screenshot']) &&
        filesize(GW_UPLOADPATH . $row['screenshot']) > 0) {
        echo '<td></td></tr>';
    }
    else {
        echo '<td></td></tr>';
    }
    $i++; [スコアアループの最後でカウンタを
           1だけ増加させます。このコードは
           $i = $i + 1;と同じです。]
}
echo '</table>';
...

```

\$iは、ハイスコアの数をカウントする変数です。最初のスコアを見分けるためにも使います。

スコアデータの配列をループし、HTMLにフォーマット

\$iが0と等しい場合、これが最初の(トップの!)スコアだとわかりますので、ヘッダのHTMLコードを描きます。

スタイルクラス topscoreheaderは、style.cssに入っています。



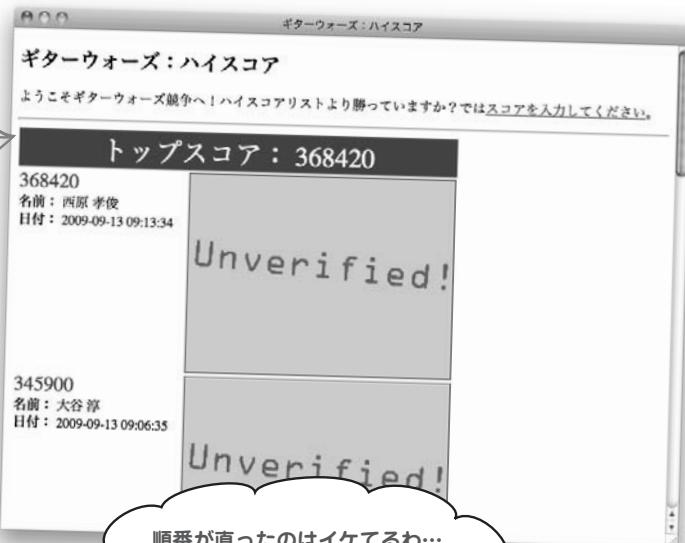
index.php



ハイスコアを順番に並べ、全体の最高スコアはショーケースに入れる

index.phpスクリプトを修正して、新しく作った順序付きのSELECT問い合わせ文を使います。次にトップスコアヘッダを生成するコードを追加します。新しいスクリプトをWebサーバにアップロードしたら、ブラウザで開いてトップスコアが眩いばかりに表示されるかどうか確認します。

最高のスコアが
でっかくはっきりと
ハイスコアリストの
先頭に表示されます。

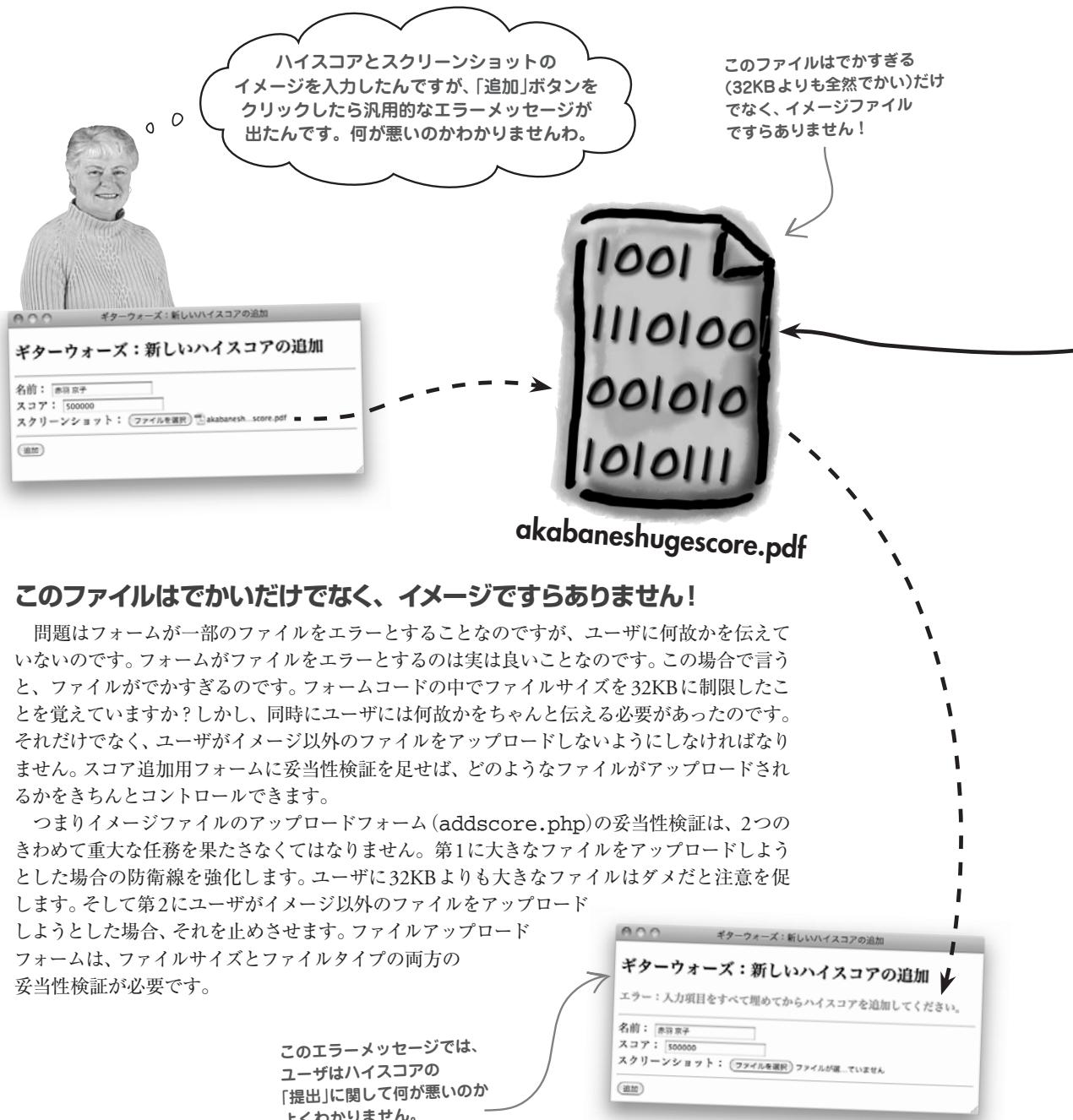


順番が直ったのはイケてるわ…
でも証拠のないスコアはインチキかも
しれないってわかってるわよね。



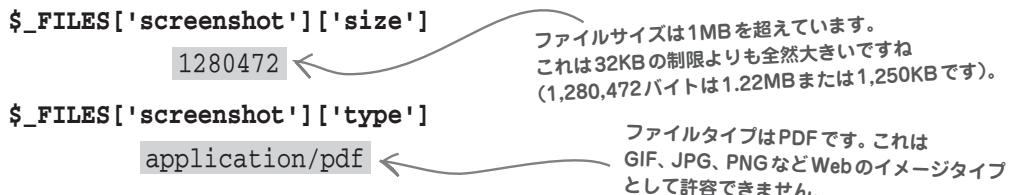
その通りです。証拠のないスコアを何とかする必要があります。

でも階段は1段ずつです。他にも問題に直面しています。ハイスコアのスクリーンショットをアップロードできない人たちがいるようです…

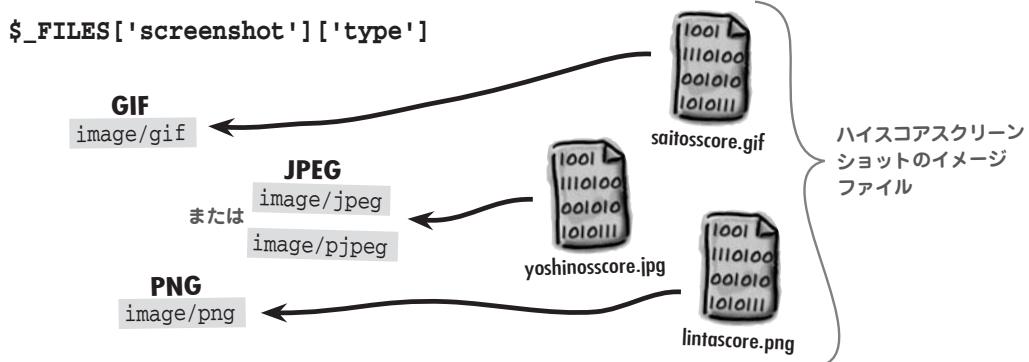


小さな ^イメージだけを許可する

それではスコア追加用フォームをきちんとチェックし、アップロードされたイメージが特定のサイズとタイプであると保証するにはどうすればよいでしょう？答えは\$_FILESスーパー・グローバル変数にあります。覚えていましたか？この変数は、以前に一時格納領域にあるアップロードされたファイルの情報を得て、imagesフォルダに移すのに使いました。ここでは同じ変数をファイルのサイズとMIMEタイプとを引っ張り出すために使います。



イメージファイルのサイズが32KBという制限よりも小さくなければならぬというだけでなく、ファイルタイプがWebのイメージとして表示可能でなければならないという要請も必要です。以下のMIMEタイプはWebイメージを表現する場合に標準的に使われるものです。



自分で考えてみよう

if文を書いて、スクリーンショットのファイルがイメージであること、同時にサイズは0よりも大きく、定数GW_MAXFILESIZEよりも小さいことをチェックして下さい。ファイルサイズとファイルタイプは、すでにそれぞれscreenshot_sizeとscreenshot_typeという変数に突っ込んであると仮定して構いません。

```

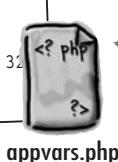
if (
.....
.....
)
{ }
  
```

自分で考えてみよう の答え

ブラウザによっては、このMIMEタイプを使ってJPEGイメージを認識します。

```
if ( (($Screenshot_type == 'image/gif') || ($Screenshot_type == 'image/jpeg') ||
      ($Screenshot_type == 'image/png') || ($Screenshot_type == 'image/pjpeg')) &&
    ($Screenshot_size > 0) && ($Screenshot_size <= GW_MAXFILESIZE)) ) {
```

```
<?php
// アプリケーション用定数の定義
define('GW_UPLOADPATH', 'images/');
define('GW_MAXFILESIZE', 32768); // 32 KB
?>
```

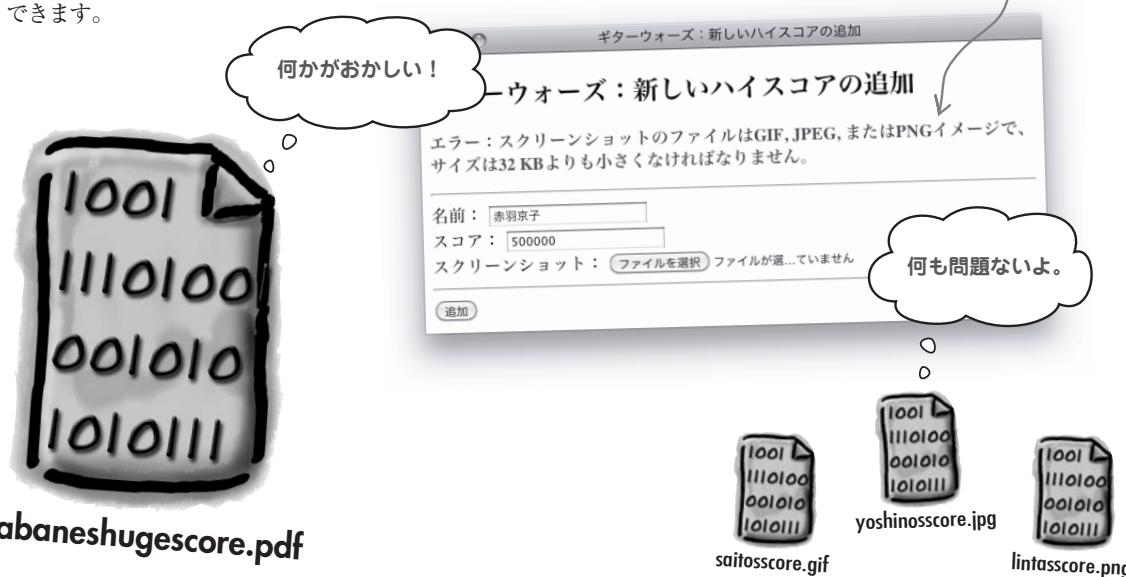


これで最大ファイルサイズがスコア追加用スクリプトの複数箇所に現れることになりましたので、これを定数としておくことは意味のあることです。

ファイルの妥当性検証により、アプリはもっと強固になります

ちょっとした妥当性検証の積み重ねで、どんなPHPアプリケーションでも、もっと直感的で使いやすくなっていくものです。誤用による危険性などについて言及するまでもありません。これで親切なエラーメッセージによりユーザは、アップロードするイメージファイルに課せられている正確な制約を知ることができます。

エラーメッセージが正確にはどのようなファイルをアップロードできるかを親切に説明してくれます。



スクリプトをもっと強固にしようとしているので、
\$_FILESスーパーグローバルでアップロード時に
エラーがあったかどうかをチェックするというの
は良いことです。

ファイルがおかしなタイプだったり
大きすぎたりした場合には、エラー
メッセージでその説明をします。

```

if (!empty($name) && !empty($score) && !empty($screenshot)) {
    if (($screenshot_type == 'image/gif') || ($screenshot_type == 'image/jpeg') ||
        ($screenshot_type == 'image/pjpeg') || ($screenshot_type == 'image/png')) &&
        ($screenshot_size > 0) && ($screenshot_size <= GW_MAXFILESIZE)) {
        if ($_FILES['file']['error'] == 0) {
            // Move the file to the target upload folder ファイルをアップロード用フォルダへ移動
            $target = GW_UPLOADPATH . $screenshot;
            if (move_uploaded_file($_FILES['screenshot']['tmp_name'], $target)) {
                // Connect to the database データベースへの接続
                $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
                // Write the data to the database データをデータベースへ書き込み
                $query = "INSERT INTO guitarwars VALUES (0, NOW(), '$name', '$score', '$screenshot')";
                mysqli_query($dbc, $query);
                // Confirm success with the user 確認画面の表示
                echo '<p>新しいハイスクアを追加していただきありがとうございます！</p>';
                echo '<p><strong>名前:</strong> ' . $name . '<br />';
                echo '<strong>スコア:</strong> ' . $score . '<br />';
                echo '</p>';
                echo '<p><a href="index.php">戻る</a></p>';
                // Clear the score data to clear the form スコアデータを消去してフォームを空白にする
                $name = "";
                $score = "";
                $screenshot = "";
                mysqli_close($dbc);
            }
        } else {
            echo '<p class="error">エラー：何らかの問題が発生しスクリーンショットのイメージ' .
                'をアップロードできませんでした。</p>';
        }
    }
} else {
    echo '<p class="error">エラー：スクリーンショットのファイルはGIF, JPEG, またはPNGイメージ' .
        'で、サイズは' . (GW_MAXFILESIZE / 1024) . ' KBよりも小さくなければなりません。</p>';
}
// Try to delete the temporary screen shot image file
@unlink($_FILES['screenshot']['tmp_name']); スクリーンショットの一時的なイメージファイルを削除
}
else {
    echo '<p class="error">エラー：入力項目をすべて埋めてからハイスクアを追加して下さい。</p>';
}

```

関数 unlink() は、ファイルを Web サーバから削除
します。@ を付けて[†] ファイルのアップロードが実際に
は成功していない場合のエラー表示を抑止します。

新しく改良されたスコア追加用スクリプトは、
これでイメージファイルの妥当性検証を行います。

[†] 訳注：@（アットマーク）は、エラー抑止演算子といって式の前に付けることで、エラーメッセージの表示を抑止します（文の前に付けることはできません）。この場合、unlink() がエラーメッセージを表示しようとしても無視して捨てられることになります。どのようなエラーメッセージが表示される可能性があるのかを事前に完全に判断できるということは少ないと思いますので、あまりお勧めできるコーディングスタイルではありません。



試運転

スコア追加用スクリプトにスクリーンショットのイメージファイル妥当性検証を加える

addscore.php スクリプトを修正して、新しくイメージファイルの妥当性を検証するコードを追加します。修正したスクリプトを Web サーバにアップロードしたら、スコア追加用フォームに妥当なイメージと不適なイメージ（でかいイメージやイメージ以外）のファイルをいくつか食わせてみます。

素朴な疑問に答えます

Q : JPEG イメージには 2 つの異なる MIME タイプがあるのは何故ですか？

A : この質問はどちらかというとブラウザのエンダに尋ねたほうが良いかもしれません。ヤツらが何故か JPEG イメージに対して異なる MIME タイプを使うことに決めたのです。JPEG ファイルの妥当性検証をなるべく多くのブラウザで通すようにするために、両方の MIME タイプをチェックする必要があります。

Q : イメージファイルが 0 バイトよりも本当に大きいかどうかを、何故チェックする必要があるのですか？イメージファイルなら 0 バイトよりも大きくて当たり前じゃないんですか？

A : 理論的にはその通りです。しかし技術的にはサーバ上に 0 バイトのファイルを作り出してしまっては可能です。例えばユーザが自分のコンピュータに実際には存在しないファイルを指定しまった場合などです。このようなことが起こってしまった場合、addscore.php は、その安全性を確認する役割を担い、ファイルの中身が空かどうかをチェックします。

Q : GW_MAXFILESIZE は addscore.php だけですか使わないのに、何故 appvars.php に入れとくのですか？

A : appvars.php が複数のスクリプトファイルで共有するデータをぶち込むことを意図して作ってあるというのは本当なのですが、同時にスクリプトのすべての定数データをぶち込むにも良い場所なのです。この場合、appvars.php に GW_MAXFILESIZE を置いとけば、仮にファイルアップロードの制限を緩くしたいと思った場合でも、どこをいじればよいか簡単に見つけられます。

Q : @unlink() のコード行はどのように動くのですか？

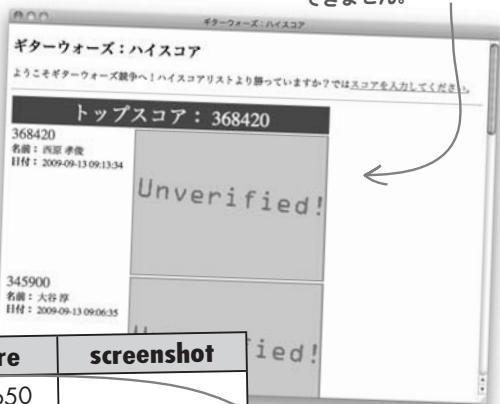
A : PHP の組み込み関数 unlink() は、ファイルを Web サーバから削除します。この場合で言うと、アップロードされた一時的なイメージファイルを削除します。アップロードに失敗して、一時的なイメージファイルがないということも起こりうるので、unlink() によって引き起こされる潜在的なエラーを抑止するために、関数に先行する記号 (@) を付与したのです。任意の PHP 関数の前に @ を貼つ付けてエラーの報告を抑止することができます。



ハイスコアリストをきれいにしなければなりません。

イメージファイルのアップロードは、妥当性検証のおかげできちんとしたものとなりましたので、証拠のないスコアはもはや無視できない問題となりました。スクリーンショットをアップロードした新しいスコアが、スクリーンショットのない古いスコアのとばっちりを被るわけにはいきません。スクリーンショットのないスコアは妥当であろうとなかろうと、ギターウォーズは古いスコアを削除する必要があります！

現在のトップスコアには証拠がありませんから、他のユーザに確信をもって浸透させることはできません。



guitarwars

id	date	name	score	screenshot
1	2009-09-12 14:37:34	岡野 一郎	127650	
2	2009-09-12 21:27:54	上野 正夫	98430	
3	2009-09-13 09:06:35	大谷 淳	345900	
4	2009-09-13 09:12:53	深沢 麻子	282470	
5	2009-09-13 09:13:34	西原 孝俊	368420	
6	2009-09-13 14:09:50	山口 裕子	64930	
7	2009-09-14 08:13:52	斎藤 真梨子	186580	saitosscore.gif

イメージがない証拠のないスコアは、データベースから一瞬で削除する必要があります！

証拠のないスコアをハイスコアリストから消すには、どのような作戦がいいと思いますか？書き留めておきましょう。

管理用ページを計画する

今必要なのは、データベースから証拠のないスコアをいくつか削除するだけです。これだけならSQLツールに点火して手動でいくつかのDELETE問い合わせ文を叩いて、データベースから行を削除すれば済みます。これは完璧に理にかなっています。でもスコアを削除する必要があるのは、今回限りではないかもしれません。Webアプリケーションをメンテナンスするために、手動でSQL問い合わせ文を叩きにょっちゅう出て行くというのは全然面白くありません。そこでアプリケーションを作つて、メンテナンスの労力をできるだけ少なくしようという算段です。

必要なページは、Webサイトの管理者だけがアクセス可能で、スコアを消すために使うことができるものです…管理者用ページです！しかし明確な境界線を注意深く引く必要があります。ギターウォーズのどこまでの部分が管理者用で、どこからがユーザ用かを区別しなければなりません。

このページはユーザ用です。

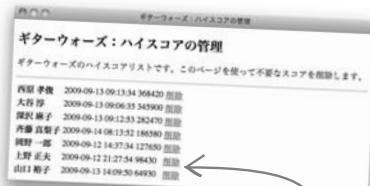


管理者用ページは、サイト管理者専用に設計されています。エンドユーザーに勝手にハイスコアを削除したりしたらイヤですよね。

Webアプリケーションには、普通一般アクセス用のページと管理者用のページがあります。管理者用のページはサイトのメンテナンス専用です。

スコア追加用ページとメインページは、ギターウォーズのエンドユーザーがハイスコアを「提出」したり閲覧したりするために設計されています。

このページは管理者専用です。



「削除」リンクをクリックすると、特定のスコアを削除します。



エクササイズ

管理者用スクリプトとスコア削除用スクリプトがやるべきことを書いて下さい。ギターウォーズにスコアを削除する機能を融合する必要があります。次にスコアの削除がguitarwarsテーブルの行とその行に関連付けられたスクリーンショットのイメージファイルにどのような影響を与えるか描いて下さい。



admin.php



removescore.php

Webサーバ



guitarwars

id	date	name	score	screenshot
1	2009-09-12 14:37:34	岡野 一郎	127650	
2	2009-09-12 21:27:54	上野 正夫	98430	
3	2009-09-13 09:06:35	大谷 淳	345900	
4	2009-09-13 09:12:53	深沢 麻子	282470	
5	2009-09-13 09:13:34	西原 孝俊	368420	
6	2009-09-13 14:09:50	山口 裕子	64930	
7	2009-09-14 08:13:52	斎藤 真梨子	186580	saitosscore.gif



管理者用スクリプトとスコア削除用スクリプトがやるべきことを書いて下さい。ギターウォーズにスコアを削除する機能を融合する必要があります。次にスコアの削除がguitarwarsテーブルの行とその行に関連付けられたスクリーンショットのイメージファイルにどのような影響を与えるか描いて下さい。

Webサーバ

```

graph TD
    Root["root"] --> WWW["www"]
    WWW --> Images["images"]
    Images --> NoScore["No score"]
    NoScore -- "X" --> DatabaseTable["guitarwars"]
    DatabaseTable --> AdminPHP["admin.php"]
    AdminPHP --> RemoveScorePHP["removescore.php"]
    RemoveScorePHP --> ConfirmationPage["ギターウォーズ：ハイスコアの削除"]
  
```

ギターウォーズ：ハイスコアの管理

ギターウォーズのハイスコアリストです。このページを使って不要なスコアを削除します。

西原 孝俊	2009-09-13 09:13:34	368420	削除
大谷 淳	2009-09-13 09:06:35	345900	削除
深沢 麻子	2009-09-13 09:12:53	282470	削除
斎藤 真梨子	2009-09-14 08:13:52	186580	削除
岡野 一郎	2009-09-12 14:37:34	127650	削除
上野 正夫	2009-09-12 21:27:54	98430	削除
山口 裕子	2009-09-13 14:09:50	64930	削除

admin.phpスクリプトは、すべてのハイスコア行をリストし、各行の隣に「削除」リンクを配置します。この情報はスコア削除用スクリプトに渡されます。

removescore.phpスクリプトは、データベースからスコアを実際に消すこと、サーバからイメージファイルを削除すること、および確認画面のメッセージを表示することのすべての面倒をみます。

ギターウォーズ：ハイスコアの削除

以下のハイスコアを削除します。よろしいですか？

名前： 西原 孝俊
日付： 2009-09-13 09:13:34
スコア： 368420

は 削除

<< 計算画面へ戻る

ギターウォーズ：ハイスコアの削除

西原 孝俊のハイスコア：368420を削除しました。

<< 計算画面へ戻る

guitarwars

id	date	name	score	screenshot
1	2009-09-12 14:37:34	岡野 一郎	127650	
2	2009-09-12 21:27:54	上野 正夫	98430	
3	2009-09-13 09:06:35	大谷 淳	345900	
4	2009-09-13 09:12:53	深沢 麻子	282470	
5	2009-09-13 09:13:34	西原 孝俊	368420	
6	2009-09-13 14:09:50	山口 裕子	64930	
7	2009-09-14 08:13:52	斎藤 真梨子	186580	saitosscore.gif

この例に限って言えば、行にはスクリーンショットのファイルがありません。しかし、スコア削除用スクリプトは、いずれはイメージを本当に持っているスコアをサーバから削除する時が来るでしょう。

管理者用ページにスコア削除リンクを付ける

スコア削除用スクリプトは、実際に削除するスコアの面倒をみますが、まずは管理者用スクリプトで、どのスコアを削除するのかを選ぶことができるようになります。`admin.php`スクリプトで、ハイスコアのリストを生成し、各行には「削除」リンクを付けます。このリンクは`removescore.php`スクリプトに選んだスコアに関する情報を渡します。

```
<?php
require_once('appvars.php');
require_once('connectvars.php');

// Connect to the database データベースへの接続
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

// Retrieve the score data from MySQL MySQLからスコアデータの取り出し
$query = "SELECT * FROM guitarwars ORDER BY score DESC, date ASC";

$data = mysqli_query($dbc, $query);           // スコアデータの配列をループし、HTMLにフォーマット

// Loop through the array of score data, formatting it as HTML
echo '<table>';

while ($row = mysqli_fetch_array($data)) {
    // Display the score data スコアデータの表示
    echo '<tr class="scorerow"><td><strong>' . $row['name'] . '</strong></td>';
    echo '<td>' . $row['date'] . '</td>';
    echo '<td>' . $row['score'] . '</td>';

    echo '<td><a href="removescore.php?id=' . $row['id'] . '&date=' . $row['date'] .
        '&name=' . $row['name'] . '&score=' . $row['score'] . '&screenshot=' .
        $row['screenshot'] . '">削除</a></td></tr>';
}

echo '</table>';
mysqli_close($dbc);
?>
```



admin.php

スコア削除用スクリプトのURLですが、
単にそのスクリプトへ遷移するだけではダメ
です…データも渡さなければいけません！

このコードで`removescore.php`への
HTMLリンクを生成します。削除する
スコアに関する情報も渡す必要があります。

```
<a href="removescore.php?id=5&date=2009-09-13%2009:1
3:34&name=西原%20孝俊&score=368420&screenshot="
```

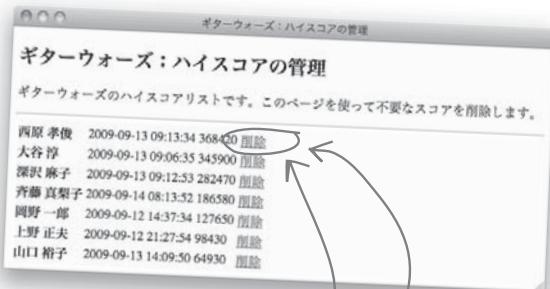
スクリプト同士でコミュニケーションをとることができます

スコア削除用スクリプトがハイスコアを削除するためには、どのスコアを削除すればいいのかを知っておかなければなりません。でもそれは管理者用スクリプトで決めたことです。というわけで疑問がわき起ります。管理者用スクリプトからスコア削除用スクリプトにどうやって削除すべきスコアを知らせればよいでしょうか？このようなスクリプト間のコミュニケーションは、管理者用ページに表示されている各ハイスコアについて「削除」URLの一部としてデータをパッケージ化することで達成できます。特定のスコアについてURLをよく見て解析すれば、すべてのハイスコアデータがそこに入っていることに気付くはずです。

スクリプトの
URLを使って、
GETリクエスト
でデータを渡す
ことができます。

```
<a href="removescore.php?  
id=5&  
date=2009-09-13%2009:13:34&  
name=西原%20孝俊 &  
score=368420&screenshot">削除 </a>
```

各データは名前と
値とからなります。「削除」URLはremovescore.php
それぞれの名前・
値ペアはアンパ
サンド(&)で
区切られています。
値とからなります。「削除」URLはremovescore.php
それそれぞれの名前・
値ペアはアンパ
サンド(&)で
区切られています。



このリンクをクリックするとスコア削除用
スクリプトを開くだけでなく、スクリプトに
GETリクエストとしてデータを渡します。

データがURLを通じて渡されるのはわかったとして、スコア削除用スクリプトが、そのデータを手にするには、正確にはどのようにするのでしょうか？ URLを通じてスクリプトに渡されたデータは、\$_GETスーパーグローバルの中に入っています。この変数も配列で\$_POSTと非常に似ています。リンクのURLにパッケージされたデータはWebフォームでGETリクエストを使うのと同様にできます。伝統的なHTMLのGETリクエストでは、フォームのデータはURLの一部として自動的にフォームを処理するスクリプトに送られます。ここでは同じことを手動的に行って、URLをカスタマイズすることでGETリクエストを組み上げます。

\$_POSTと同様、\$_GET配列を使ってハイスコアデータにアクセスするには、各データの名前が要ります。

スクリプト用にURLを使うと
データベース行のIDなど重要な
データを手軽に渡すことが
できます。



各データの名前を使って、
\$_GET配列の中身の
データにアクセスします。



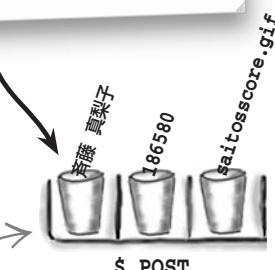
わからんないわ。GETに
ムカついてるんだけど…POSTを使って
スクリプトにデータを渡せないのは何故なの?
今までずっとそうしろって言われてきたわ。

POSTリクエストは、フォームを通してのみ送り出すことができるのに 対し、GETリクエストはURLにパッケージ化できます。

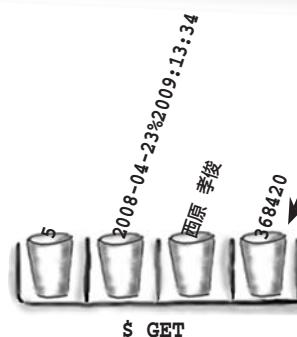
今までスクリプトにデータを受け渡す際は、Web フォームを使ってきました。ここではスクリプトは、フォームの「提出」ボタンのアクションとしてリストされていました。ユーザがフォームを埋めて「提出」ボタンを押すと、フォームのデータはパッケージ化され、POSTリクエストとして送られていました。

ここで問題となるのは、管理者用ページがスコア削除用スクリプトに送り出す際にフォームを使っていないということです。URLを通して削除用スクリプトにリンクしているだけです。このため URL 以外は何も使わずにデータを送るための別のやり方が必要なのです。このような場合、GETが特に便利です。URLにパラメタとしてパッケージ化してデータにアクセスする手段を提供してくれるからです。POSTと同様に、GETリクエストを通じてスクリプトに渡されるデータは、スーパーグローバルを通して使うことができます。ただし名前は \$_POST ではなく \$_GET です。

Web フォームは通常
POSTを使ってデータを
「提出」しますが、このとき
データは\$_POST配列に
突っ込まれています。



ギター ウォーズ：ハイスコアの管理	
ギター ウォーズのハイスコアリストです。このページを使って不要なスコアを削除します。	
西原 孝俊	2009-09-13 09:13:34 368420 削除
大谷 淳	2009-09-13 09:06:53 345900 削除
深沢 麻子	2009-09-13 09:12:53 282470 削除
齊藤 真梨子	2009-09-14 08:13:52 186580 削除
岡野 一郎	2009-09-12 14:37:34 127650 削除
上野 正夫	2009-09-12 21:27:54 98430 削除
山口 恵子	2009-09-13 14:09:50 64930 削除



URL を通してデータを
渡すのなら GET を使えば
達成できます。このとき
データは\$_GET配列に
突っ込まれています。

GETとPOSTについて

GETとPOSTの違いは、単にフォームとURLに留まりません。なぜならGETリクエストはフォームデータを「提出」するために使うことができ（かつ実際よくそのように使われる）からです。GETとPOSTの本当の違いは、リクエストの内容をどう扱うかによります。GETは、サーバに何ら影響を与えることなくサーバからデータを取り出す、というのが主な使い方です。一方POSTは、サーバにデータを送るというのが典型的です。この後、送られたデータの応答としてサーバの状態は何らかの変更を伴うのが普通です。

POST

データをサーバに送るために使います。これによりサーバの状態には何らかの変更が生じます。例えばデータをデータベースに突っ込むといったことです。データはレスポンスとして返すこともできます。GETと異なり、POSTリクエストはWebフォームのアクションとしてのみ作り出すことができます。同じくGETと異なりPOSTで送られたデータは目に見えるところにはありません。

2種類のWebリクエストGETとPOSTは、スクリプト間のデータをどのように行き来させるかをコントロールします。

GET

データを取り出すためによく使いますが、サーバ上には何ら変更を及ぼしません。データが少ない場合、GETはURLにデータを埋め込んでサーバへ直接送ることができます。POSTと異なり、GETは、主に少量のデータを送る場合に向いています。

素朴な疑問に答えます

Q : GETを使ってWebフォームを送るのを見たことがあるのですが、どのように動くのですか？

A : GETにもPOSTにもWebフォームにおいてそれぞれ適した場所があります。Webフォームを作る場合、<form>タグのmethod属性が、データをどのように送るかをコントロールします。一方でaction属性にはデータを受け取り、処理するスクリプトを指定します。

```
<form method="post" action="addscore.php">
「提出」ボタンをクリックし、このフォームを「提出」した場合、addscore.phpスクリプトが実行され、フォームのデータは$_POST配列を通してスクリプトに渡されます。しかし実は、同様に<form>タグを以下のように書いてもよかったです。この場合データは$_GET配列を通して渡されます。
```

```
<form method="get" action="addscore.php">
```

Q : なるほど。じゃあリクエストメソッドとしてはGETでもPOSTでもどっちでもよかったわけですね？

A : 違います。全くもって、どっちでもよくありません。GETは普通サーバからデータを取ってくるのに使われます

が、サーバ上では何も変更しません。ですからGETはサーバ上で情報的なリクエストをするフォームに完璧に向いています。サーバの状態を全く変更することのないもの、例えばデータベースから行を選ぶ場合などです。これに対しPOSTは、サーバの状態に影響を及ぼすようなリクエストに最適です。例えばINSERTやDELETEといった問い合わせ文を発行してデータベースを変更するような場合です。GETやPOSTについての、これ以外の違いは、GETを使ってデータを渡すとURL上にデータが見えるということです。これに対しPOSTデータは隠れていますので、ほんのちょっとだけセキュリティが上です。

Q : GETとPOSTに関するこの区別により、URLを通してスクриプトにデータを渡す際にどんな注意が必要でしょう？

A : そうですねえ、何よりもまずURLを通してデータを渡すことができるのはGETだけです。ですからこの点に関してはPOSTは即無視です。さらにGETは純粋にサーバの状態を変更しないリクエスト用という意図を持っています。ですからURLを通して受け取ったデータを使って、INSERTやDELETE FROMなど、スクリプトのデータベースを変更するようなあらゆる操作を行ってはいけません。

炉辺歓談[†]



今夜のゲスト：GETとPOST

[†] 訳注：原著ではFireside chatsで、暖炉脇でのなごやかな会話という意味です。ルーズベルト大統領はラジオでfireside chatsの形でニューデール政策を発表したそうです。コンピュータ上でのチャットとこの件をかけたジョーク。

GET：

じゃあまず、街の噂じゃ僕にできることは質問をすることぐらいで、その答えに対しては何もできないって君が言っているって聞いたよ。マジかよ？

そりや確かに僕の仕事は、基本的にサーバに変更を加えることで、例えばファイルを消したり、データベース行を足したりはしないけど…だからと言ってそれでオレが重要じゃないってことにはならないだろ？

そりやそうだけど、君はいつも相棒のフォームとベッタリくつ付いたままだよね。それに比べて僕とフォームとの仲は、ちょっとした知り合いで言う程度だよ。僕には他にも友人がいるよ。URLとか。

それじゃあ聞いてみたいんだけど、そのダチのフォームが君の周りにいないとき、君はどうやって動き出すことができるのさ？君だって知っているはずだけど、ページによってはフォーム使うことが難しいような状況だってあるよね。

まあ落ち着けよ。僕が言いたいのは、僕がサーバからデータを取り出したりするのに向いていて、それをする時にちょっとだけ柔軟性があるということさ。

それを聞けてよかったです。君と話ができる良かった…

POST：

マジや。現実を直視しよやないか。アンタはホンマの力は何もなくて、サーバに何か聞くことができるだけやろ。

ほな、それ認めるとしよやないか。けどワイの知っている限り、みなワイみたいにサーバ上で何かやりたいはずや。それ以外にやることなんてなんもないやろ？もし仮にサーバがずっと同じ状態やったら、ごっつう退屈やないか。

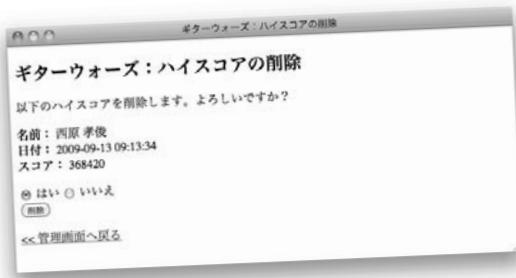
そしたらアンタは「友達の輪」が大事で、自分は何もできんことよりもエライとでも言うつもりか？そんなんあり得へん。

まあ聞けや。確かにフォームはダチや。遠い昔にヤツを通して以外とはいかなるリクエストもせえへんいう契約を交わしたんや。せやから今試されているんはワイの忠誠心の問題なんや。ワイはダチを裏切ったりはでけん。

そやな、認めるわ。アンタのことはよくわかったわ。

GETとPOSTとハイスコアの削除と

ギターウォーズのスコアを消すのに管理者用ページの「削除」リンクから始まり、スコア削除用スクリプトに至るというルートを確立しました。またスコアに関するデータはURLを通してスコア削除用スクリプトに渡されるということもすでに知っています。しかし、GETリクエストはサーバ上の何も変更してはならないという問題を抱えています。つまりスコアを削除してはいけないので。可能性としてあり得る解決策は、とりあえずはサーバ上では何も変更しないことです。スコア削除用スクリプトは、まずは確認画面を表示し、その後、データベースから実際の削除を行うというはどうでしょう？



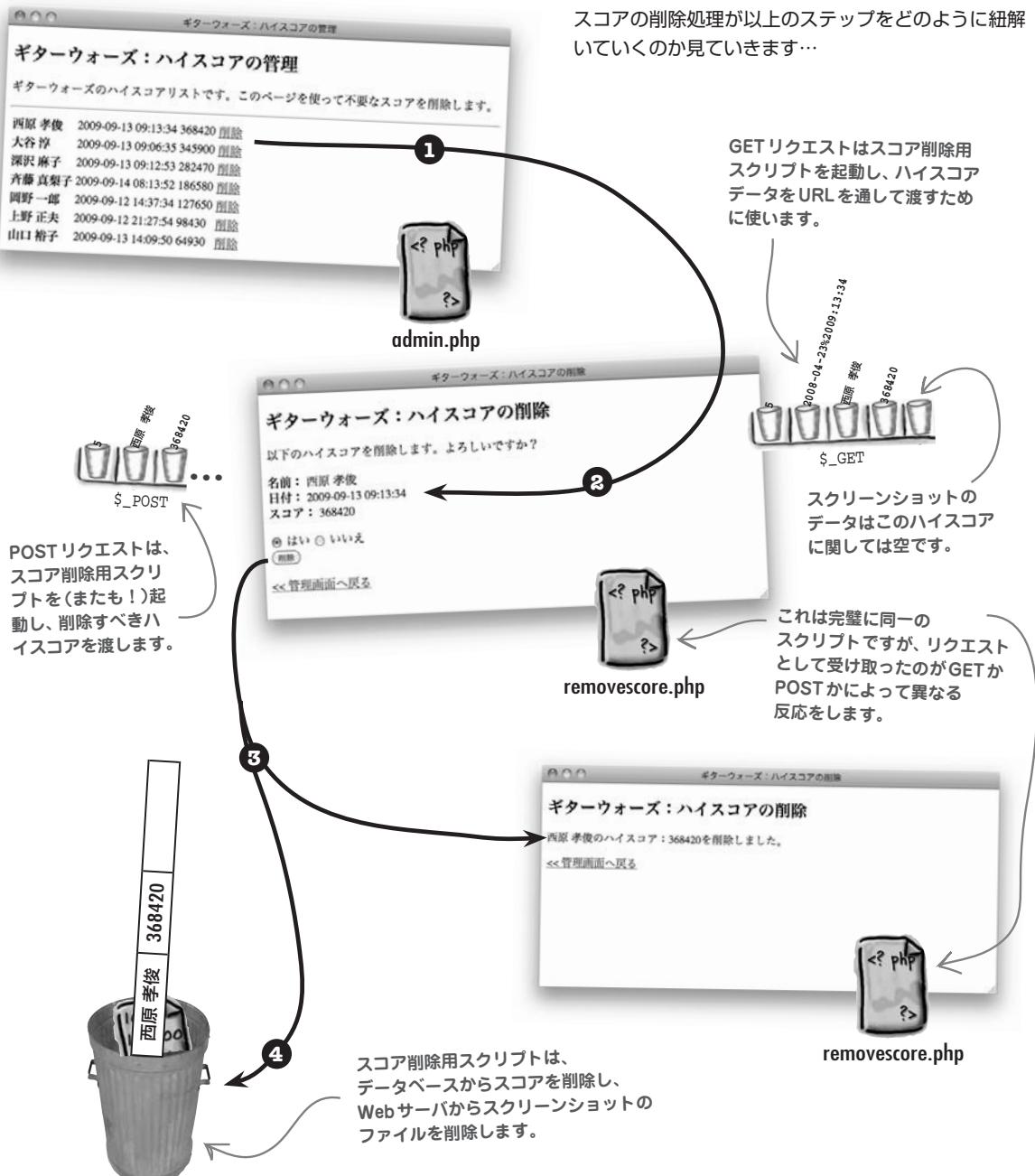
確認画面では、ハイスコアを削除するかどうかを確認する機会をユーザに与えます。
単に即削除ではありません。

同じスクリプトが
GETとPOSTの
両方のリクエストに
応答するというのは
全然あり、というか
むしろ便利なこともあります。

確認画面は削除するスコアを「はい・いいえ」付きで表示するシンプルなフォームでいいでしょう。「はい」を選んで「削除」ボタンをクリックすれば、スコアは削除され、「いいえ」を選べばスコア削除はキャンセルされます。

GETとPOSTを使って言い換えれば、スコア削除用スクリプトは、管理者用スクリプトのGETリクエストのレスポンスとして確認画面を表示します。次に確認画面それ自身はフォームですから、「提出」されるとPOSTリクエストを発行します。このフォームが自己参照フォームであれば、同じスクリプト(`removescore.php`)がPOSTを処理し、スコア削除を実行します。この処理は次のようなステップで行われます。

- ① スコア削除用スクリプトは、GETリクエストを通して、ユーザが管理者用ページの「削除」リンクをクリックすることにより起動されます。
- ② スコア削除用スクリプトは、\$_GET配列に入っているハイスコアデータを使って、削除確認画面のフォームを作ります。
- ③ スコア削除用スクリプトが再度起動されます。今回はユーザが確認画面のフォームを「提出」したことによるPOSTリクエストです。
- ④ スコア削除用スクリプトは、データベースからスコアを削除し、同時にWebサーバからスクリーンショットのイメージファイルを削除します。



素朴な疑問 に答えます

Q: 同じスクリプトでどうやって GET と POST の両方のリクエストを処理できるのですか?

A: これはスクリプトがどのように呼び出されるかにかかっています。スコア削除用スクリプトの場合、これらの2つは異なる方法で呼び出されます。第1の方法はユーザが管理者用ページで「削除」をクリックした場合で、URLがスクリプトへデータを運んできてくれます。データはURLにパッケージ化されているため、GETリクエストと考えることができます。このGETリクエストによってスクリプトはWebフォームを作ることができますが、フォームのアクションは同じくスコア削除用スクリプトを参照しています。このためユーザがフォームを「提出」すると同じスクリプトが改めて呼び出されます。しかし1回目と異なり、URLの中にデータがパッケージ化された風変わりなURLではないため、今度はGETリクエストではありません。代わりにハイスコアデータはPOSTリクエストを通して渡されますから、今度は\$_POST配列を使います。

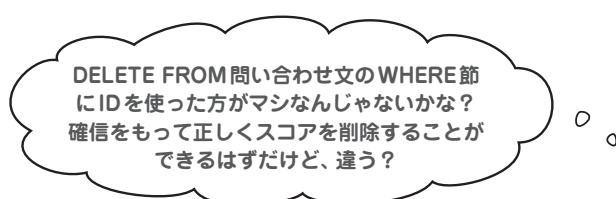
Q: つまりスクリプトがどのように呼び出されるかによって、それが何かが決まるということですか?

A: その通りです!スクリプトは、データがURLを通してGETリクエストで送られてくるのを見ると、確認画面のフォームを表示するのだと認識します。データベースから何も消したりしません。つまり\$_GET配列を使って送られてきたデータは、確認画面のページの中だけで使われ、サーバ上への永続的な効果はありません。
スクリプトは、データがPOSTリクエストを通して運ばれてくるのを見ると、データベースからデータを削除してもよいと認識します。つまり\$_POST配列はデータにアクセスし、DELETE FROM問い合わせ文を組み上げてスコアを削除します。さらにたいていのハイスクロアは、スクリーンショットのイメージファイルがサーバ上にぶち込まれていますから、スクリプトはそのファイルを消してしまうということも同時に行います。

LIMITで削除する数をコントロールする

`name`と`column`の両方に基づいて行を削除するというのは良いアイデアですが十分ではありません。アプリケーション開発では、常にリスクを最小限にすることを考えなければなりませんが、今のままで同じ`name`と`score`に一致する複数の行を削除してしまうかもしれないというリスクが多少なりともつきまといいます。これを解決するにはどんな場合でも問い合わせ文で**1行しか削除しない**というように強制してしまえばよいのです。`LIMIT`節を使えばこのようなことができます。

```
DELETE FROM guitarwars WHERE name = '西原 孝俊' AND score = '368420' LIMIT 1
```



↑
安全性を最大限に高めるために、削除する行数の上限を指定します。



その通りです！ハイスコアのIDを使うのは、削除するスコアをより分ける完璧な方法です。

主キーを作ったことによる第1の利点は、テーブル内での一意性を確保できることです。guitarwarsテーブルの`id`カラムは、主キーですから、すべてのハイスコアについて一意です。このカラムをDELETE FROM問い合わせ文のWHERE節に使うことで、どのスコアを削除するかということにまつわる、あらゆる疑惑を払拭することができます。これが`id`カラムを使った問い合わせ文で、これで一意性を保証できます。

```
DELETE FROM guitarwars WHERE id = 5
```

`id`カラムは実際に主キーですから、これを信用して上記のコードで削除すれば安全に1行だけを削除することができます。でも万が一データベースを作り損ねていたとしたらどうでしょう？一意性が保証されていないかもしれません。このようなことを想定すれば、`LIMIT`節を指定するというのも、相変わらず意味のあることかもしれません。根拠としては、問い合わせ文で1行だけに影響を与えた場合、問い合わせ文にその旨記述するという意味です。

```
DELETE FROM guitarwars WHERE id = 5 LIMIT 1
```

問い合わせ文で何をしたいのかを明示するのが悪い、ということは絶対にありません。この場合で言うとLIMITによりDELETE問い合わせ文が、もう一段上のレベルで安全性を付与しています。

↑
主キーに基づいてデータを削除すれば、削除する正しい行を正確により分けることが可能となります。

LIMIT節は、問い合わせ文が2行以上を削除できないことを明示的に指示します。



PHP & MySQLマグネット

removescore.php スクリプトはほとんどできています。でもコードの重要な部分が少し抜けています。欠けているコードにマグネットを差し込み、ギターウォーズから望ましくないスコアを根絶する機能を追加して下さい。

```

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
    <title>ギターウォーズ：ハイスコアの削除</title>
    <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
    <h2>ギターウォーズ：ハイスコアの削除</h2>

    <?php

        ..... ('appvars.php');
        ..... ('connectvars.php');
        if (isset($_GET['id'])) && isset($_GET['date']) && isset($_GET['name']) &&
        isset($_GET['score']) && isset($_GET[ ..... ]) {
            // Grab the score data from the GET GETからスコアデータを取り出す
            $id = $_GET['id'];
            $date = $_GET['date'];
            $name = $_GET['name'];
            $score = $_GET['score'];
            ..... = $_GET[ ]; .....
        }
        else if (isset($_POST['id']) && isset($_POST['name']) && isset($_POST['score'])) {
            // Grab the score data from the POST POSTからスコアデータを取り出す
            ..... = $_POST[ ]; .....
            $name = $_POST['name'];
            $score = $_POST['score'];
        }
        else {
            echo '<p class="error">エラー：削除するハイスコアが指定されていません。</p>';
        }
        if (isset($_POST['submit'])) {
            if ($_POST['confirm'] == ..... ) { サーバからスクリーンショットのイメージファイルを削除
                // Delete the screen shot image file from the server
                @unlink(GW_UPLOADPATH . $Screenshot);
                // Connect to the database データベースへの接続
                $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
            }
        }
    
```

```
// Delete the score data from the database データベースからスコアの削除
// ..... guitarwars WHERE ..... LIMIT ";
$query = " ..... guitarwars WHERE ..... LIMIT ";
mysqli_query($dbc, $query);
mysqli_close($dbc);

// Confirm success with the user 確認画面の表示
echo '<p>' . $name . ' のハイスコア: ' . $score . ' を削除しました。';
}

else {
    echo '<p class="error"> エラー: ハイスコアは削除されませんでした。</p>';
}
}

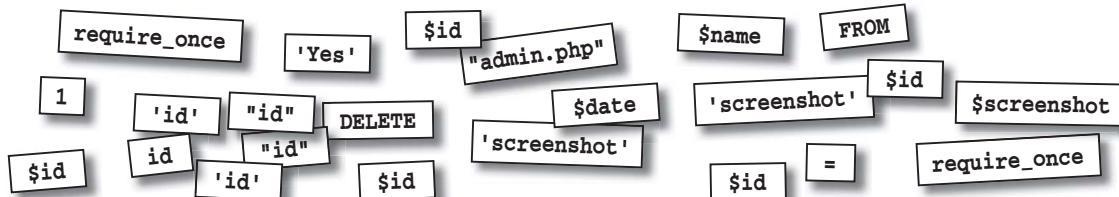
else if (isset( ..... ) && isset( ..... ) && isset( ..... ) &&
isset($score) && isset($screenshot)) {
    echo '<p>以下のハイスコアを削除します。よろしいですか? </p>';
    echo '<p><strong>名前: </strong>' . $name . '<br /><strong>日付: </strong>' . $date .
    '<br /><strong>スコア: </strong>' . $score . '</p>';
    echo '<form method="post" action="removescore.php">';
    echo '<input type="radio" name="confirm" value="Yes" /> はい ';
    echo '<input type="radio" name="confirm" value="No" checked="checked" /> いいえ <br />';
    echo '<input type="submit" value="削除" name="submit" />';
    echo '<input type="hidden" name= ..... value="" .. ..... . '' />';
    echo '<input type="hidden" name="name" value="' . $name . '" />';
    echo '<input type="hidden" name="score" value="' . $score . '" />';
    echo '</form>';
}

echo '<p><a href= ..... >&lt;&lt; 管理画面へ戻る </a></p>';
?>

</body>
</html>
```



removescore.php





PHP & MySQL マグネットの答え

`removescore.php`スクリプトはほとんどできています。でもコードの重要な部分が少し抜けています。欠けているコードにマグネットを差し込み、ギターウォーズから望ましくないスコアを根絶する機能を追加して下さい。

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>ギターオーズ：ハイスコアの削除</title>
  <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
  <h2>ギターオーズ：ハイスコアの削除</h2>
  <?php
    require_once ('appvars.php');
    require_once ('connectvars.php');

    if (isset($_GET['id']) && isset($_GET['date']) && isset($_GET['name']) &&
        isset($_GET['score']) && isset($_GET[ . 'screenshot' .. ])) {
      // Grab the score data from the GET
      $id = $_GET['id'];
      $date = $_GET['date'];
      $name = $_GET['name'];
      $score = $_GET['score'];
      $screenshot = $_GET[ ]; . 'screenshot' . .;

    } else if (isset($_POST['id']) && isset($_POST['name']) && isset($_POST['score'])) {
      // Grab the score data from the POST
      $id = $_POST[ ]; . 'id' . .;
      $name = $_POST['name'];
      $score = $_POST['score'];
    }
    else {
      echo '<p class="error">エラー：削除するハイスコアが指定されていません。</p>';
    }
    if (isset($_POST['submit'])) {
      if ($_POST['confirm'] == .. 'Yes' ..) {
        // Delete the screen shot image file from the server
        unlink(GW_UPLOADPATH . $screenshot);
        // Connect to the database
        $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
      }
    }
  }
</?php>
```

共有スクリプトをインクルードします
が、require_onceを使います。スコアの
削除にとって重要なデータだからです。

スクリプトは、入力の
リクエストがGETかPOSTか
によって異なった動作をします。

PHPの@は、エラー抑止指令といって、エラーが
表示されるのを防ぎます。これをunlink()に対して
使うことは有効です。なぜならファイルが存在しない
ときに削除しようとするかもしれないからです。
この場合でも、ユーザーにエラーを見せなくてすみます。

このスクリプトは、どんな
スコアでも削除することができます。そこでアップロード
されたファイルは、削除の一環
として消さなければなりません。

```

// Delete the score data from the database
$query = "DELETE FROM .. guitarwars WHERE .. id = $id LIMIT 1";
mysqli_query($dbc, $query);
mysqli_close($dbc);

// Confirm success with the user
echo '<p>' . $name . 'のハイスコア:' . $score . 'を削除しました。';
}

else {
    echo '<p class="error">エラー:ハイスコアは削除されませんでした。</p>';
}

}

else if (isset( $id ) && isset( $name ) && isset( $date ) &&
isset($score) && isset($screenshot)) {
    echo '<p>以下のハイスコアを削除します。よろしいですか? </p>';
    echo '<p><strong>名前: </strong>' . $name . '<br /><strong>日付: </strong>' . $date . 
    echo '<p><strong>スコア: </strong>' . $score . '</p>';
    echo '<form method="post" action="removescore.php">';
    echo '<input type="radio" name="confirm" value="Yes" /> はい ';
    echo '<input type="radio" name="confirm" value="No" checked="checked" /> いいえ <br />';
    echo '<input type="submit" value="削除" name="submit" />';
    echo '<input type="hidden" name= "id" value=' . $id . ' />';
    echo '<input type="hidden" name="name" value=' . $name . ' />';
    echo '<input type="hidden" name="score" value=' . $score . ' />';
    echo '</form>';
}

echo '<p><a href= "admin.php" >&lt;&lt; 管理画面へ戻る </a></p>';
?>

</body>
</html>

```

管理者用ページへ戻る
リンクをつけて操作性を
向上させます。

\$id
"id"
'id'

hidden フォームフィールドを
使ってスコアのデータを格納し、
POSTリクエストの一部として
送れるようにします。

マグネットが少し余りました。

idカラムでDELETE問い合わせ文
に一致させ、ついでにLIMITで
1行に限定します。

↓

↓

↓

↓

↓

↓

↓

↓

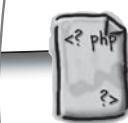
↓

↓

↓

↓

ここでは\$_SERVER['PHP_SELF']
を使いません。使ってしまうとGET
でURLの文字列を通して渡された
データが入ってしまうかもしれません
からです。このフォームによりGET
データが渡されないということを
保証したいのです。POSTデータ
だけです。



removescore.php

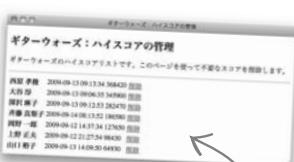
確認画面のフォームは、すべての
ハイスコア変数に値が設定されて
いる場合に限り表示されます。



試運転

スコア削除用スクリプトと管理者用スクリプトをギターウォーズに追加し、スコアを削除できるようにする。

新しく2つのテキストファイルremoveScore.phpとadmin.phpを作り、今作業してきたコードを追加します。新しいスクリプトをWebサーバにアップロードしたら、管理者用スクリプトをWebブラウザで開きます。適当な捨てたいスコアの「削除」リンクをクリックし、スコア削除用ページで削除の確認をします。管理者用ページに戻ってスコアがなくなっていることを確認したら、ギターウォーズのメインページ(index.php)に変更が反映されているか確かめて下さい。



新しく作った管理者用ページは証拠のないハイスコアを削除するためのリンクを持っています。

この町でロックのトップの座は1つだけ。
そしてそれはボクさ！

凜太くんといいます。
ギターウォーズの勝者で
ロックの神童です。



正直なギターウォーズ
参加者は、証拠付きの
ハイスコアだけを見る
ことができ幸せです。

証拠のないハイスコアは、
スクリーンショットのイメージ
がないので、システムから
削除されました。

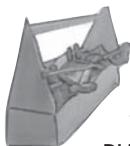


新しく作ったスコア削除用ページは、
要らないスコアの確認と削除の両方の面倒をみます。



ギターウォーズの
メインページは、これで
証拠のあるハイスコアのみを表示できます。





PHP & MySQL 道具箱

手にした仮想の武器をためらわずに使って下さい。

世界中の仮想ギタリストから愛されるのみならず、

PHPとMySQLのスキルについても極めて多くのものを手にしました。テーブル構造の変更、ファイルアップロードの処理、データ順序の制御、さらにデータの削除です。

`ALTER TABLE table
ADD COLUMN column type`

このSQL文を使って、既存のデータベーステーブルに新しいカラムを追加します。カラムはテーブルの最後に追加され、すでに存在する行は空で初期化されます。

`include, include_once,
require, require_once`

これらのPHP文は、1つのアプリケーションの複数のスクリプトファイルにまたがってスクリプトコードを共有することができるようになります。重複したコードをなくし、コードのメンテナンスをしやすくなります。

`ORDER BY column`

このSQL文は問い合わせ文の結果を特定のカラムデータに基づいて並べ替えます。ASCまたはDESCをSQL文の後ろに付けると、それぞれデータを昇順または降順にソートしてくれます。ORDER BYに何も指定しない場合のデフォルトはASCですから、これは省略可能です。

`images`

このフォルダはアプリケーションがイメージをぶち込むのに便利な場所です。ユーザがアップロードしたイメージを入れても構いません。

`$_FILES`

PHPの組み込みスーパーグローバル変数で、ファイル入力フォームを通してアップロードされたファイルに関する情報が突っ込まれています。これを使うことでファイル名、一時格納領域の場所、ファイルサイズ、ファイルタイプなどをることができます。

`DELETE FROM table
WHERE column = match
LIMIT num`

このSQL文を使うとデータベーステーブルから行を削除します。一致するものが2つ以上あるかもしれないで、削除の精度を上げるために、LIMITを使って1行のみを削除するようにしてすることができます(むしろそのように指定すべきと言った方がいいかもしれません)。

6章 アプリケーションにセキュリティを与える

みんなが自分を
狙っていると思え

ちょっと登って、この
短い電柱の足場をつかんで。
こんな小っつっさい町じゃ何に
ぶつかるかわからないよ。

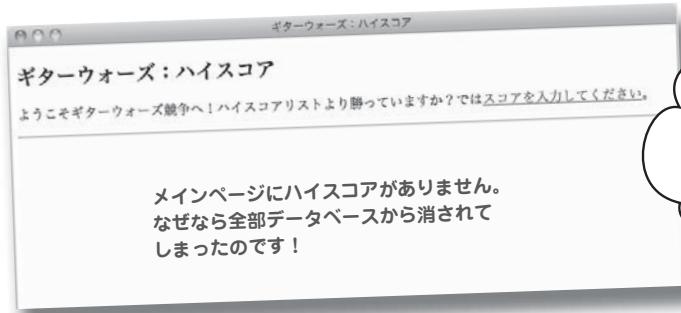


両親の言っていたことは正しかったのです。「知らない人と話してはいけません。」

または少なくとも知らない人を信用してはいけません。何を差し置いても、知らない人にアプリケーションデータのカギとなるものをあげてはいけません。知らない人が正しいことをちゃんとしてくれるなどという仮定をしてはいけません。世の中一步外に出れば恐ろしいことだらけです。誰でも信用できるなどと思ってはいけません。実際、アプリケーションの開発者として、時には皮肉屋になり、時にはテロリストとして陰謀を企てるなどなりません。そうです。性悪説に立って下さい。みんな間違いなくあなたを狙っているのです！まあ、確かにちょっと大げさかもしれません。でもセキュリティについて真剣に考えてアプリケーションを設計するというのは非常に重要なことです。そうでなければ危害を加える可能性のある誰かに対抗してアプリケーションを守ることはできません。

音楽が死んだ日

うわっと、我らが若き仮想ロックの神童が脚光を浴びた瞬間は、短命に終わってしまいました。凜太くんが記録したトップのギターウォーズスコアはどこかに消えてしまいました。その他のスコアも全部です。悪魔の暴挙によってハイスクアアプリケーションが打ち碎かれ、ギターウォーズ参加者はオンライン上で競うことができなくなってしまいました。不幸な仮想ギタリストたちは、不幸なユーザであり、怒りの矛先はアプリケーションの不幸な開発者に向けられます…つまりあなたです！



こんなのフェアじゃないよ！
ボクがギターウォーズの最高得点を
叩き出すのにどれだけ苦労したか
分かる？それなのにボクのスコアは
どっか行っちゃった。

ハイスクアはどこへ？

ギターウォーズのメインページは空白であることはもう分かっています。でもそれはデータベースに何も入っていないということなのでしょうか？SELECT文で答が分かります。

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> USE gwdb;
Database changed
mysql> SELECT * FROM guitarwars;
Empty set (0.00 sec)

mysql>
```

何らかの原因ですべてのハイスクアデータが、ギターウォーズデータベースから吹っ飛んでしまいました。どっかの誰かがスコア削除用スクリプトを使って悪さをしたのではないかでしょうか？スコアを守る必要があります！

ギターウォーズのトップスコアを
叩き出した凜太くんです。トップスコア
がハイスクアリストから消えてしまって、
怒り心頭で狂ってます。

凜太くんが音楽用の武器と
して選んだのは、2005年の
Eradicaster、ビンテージ
ものです。



SELECT問い合わせ文で調べたところ、
guitarwarsテーブルが完全にカラ
であることが分かりました。全スコアは
どっかへ行ってしまいました！

うじゃうじゃいる大群をセキュリティで監禁する

単純かつ直接的な方法で手早くギターウォーズのハイスコアにセキュリティを与えるなら、HTTP認証を使って管理者用ページをパスワードで保護すればよいのです。この技法では実際にはユーザ名とパスワードが必要です。しかしアイデアは秘密の情報をいくつか管理者に要求し、その後晴れて制限されたアプリケーションの機能、今の場合だとスコアの削除リンク、にアクセスできるというものです。

ページにHTTP認証を使ってセキュリティを与えると、ウインドウがポップアップしてユーザ名とパスワードを要求してきます。認証に成功すると保護されたページにアクセスすることができます。ギターウォーズの場合で言うと、管理者用ページへのアクセスを好きだけ少ない人数に制限することができます。もちろん自分1人にでもできます！



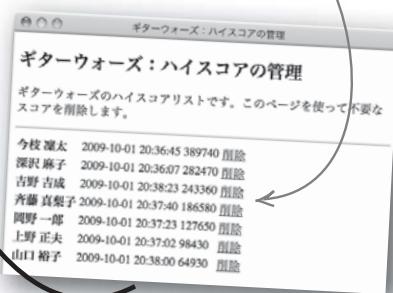
データベースのハイスコア
はこれで大丈夫です。
管理者用ページが
守られているからです。

guitarwars				
id	date	name	score	screenshot
14	2009-10-01 20:36:07	深沢 麻子	282470	asakoscore.gif
15	2009-10-01 20:36:45	今枝 潤太	389740	lintasscore.gif
16	2009-10-01 20:37:02	上野 正夫	98430	uenasscore.gif
17	2009-10-01 20:37:23	岡野 一郎	127650	okanasscore.gif
18	2009-10-01 20:37:40	齊藤 真梨子	186580	saitosscore.gif
19	2009-10-01 20:38:00	山口 裕子	64930	yukoscore.gif
20	2009-10-01 20:38:23	吉野 吉成	243360	yoshinoscure.gif

† 訳注：原著に従いHTTP認証としましたが、日本では普通Basic認証（ベーシック認証）と呼ばれています。

HTTP認証[†]は、
PHPでページに
セキュリティを与え
る簡単な方法を
提供してくれます。

これでHTTP認証
ウインドウがユーザと
管理者用ページの間に
立ちはだかります。

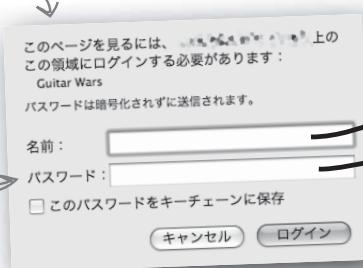


ギターウォーズの管理者用ページを守る

HTTP 認証は次のような動作をします。ユーザが認証により保護されているページ、例えば管理者用ページにアクセスすると、まずユーザに対してユーザ名とパスワード入力を促すウインドウが現れます。

Web ブラウザはこのような
ウインドウを使って、ユーザ
名とパスワードを要求し、
その結果に応じて保護され
たページへのアクセスを
許可します。

話を単純にするため、
パスワードは暗号化
していません。



この PHP スーパーグローバル
変数が認証ウインドウに入力
されたユーザ名を保持しています。

`$_SERVER['PHP_AUTH_USER']`

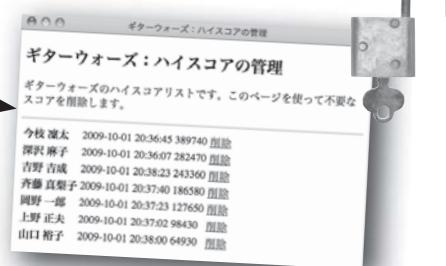
`$_SERVER['PHP_AUTH_PW']`

この変数が認証ウインドウ
に入力されたパスワードを
保持しています。

PHP の出番は、ユーザがユーザ名とパスワードを入力してアクセスするところからです。これら情報は `$_SERVER` スーパーグローバルに突っ込まれます。このスーパーグローバルも、今まで使ってきたもの (`$_POST`, `$_FILES` など) と同様な働きをします。PHP スクリプトは、ユーザが入力したユーザ名とパスワードを解析することができます。そして保護されたページへのアクセスが許されるべきかどうかを決定することもできます。例えば、ユーザ名が `rock` でパスワードが `roll` の場合のみ管理者用ページにアクセスできることにしましょう。その場合、以下のようにすれば管理者用ページのロックが外れます。

管理者用ページは、正しい
ユーザ名とパスワードを入
力した場合に限りアクセス
可能です。

`$_SERVER['PHP_AUTH_USER']`



素朴な疑問に答えます

Q : HTTP認証って本当にセキュリティ上問題ないんですか?

A : YESでもNOもあります。答えはセキュリティ（の技法）を使って何をやろうとしているのかに完全に依存しています。いまだかつて100%完全なセキュリティは存在しません。従って、常にセキュリティの度合い（レベル）について語るしかありません。ギターウォーズのスコアを守るという目的に関して言えば、HTTP認証は十分なレベルのセキュリティを提供してくれます。もちろんパスワードを暗号化することによって、ほんのちょっとレベルを上げることもできます。でも、それでさえもアプリケーションによっては恐らく全然十分でない場合があります。取り扱い厳重注意のデータ、例えば金融データなどを扱う場合がそうです。

Q : ユーザ名やパスワード[†]を間違って入れたら何が起こるんですか?

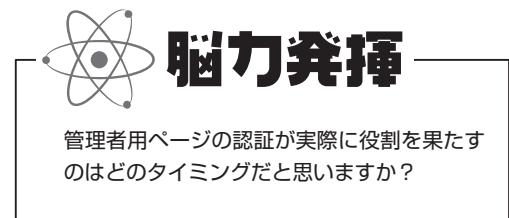
A : ブラウザはマウスを通して電気的な衝撃を放ちます。ウソです。何も残酷なことは起りません。通常はユーザに対してメッセージが表示されます。そのメッセージは、セキュリティで保護されたページにアクセスしようとしたが、その権限がないことを伝えるためのものです。このメッセージをどの程度恐ろしいと解釈するかは、人それぞれです。

Q : HTTP認証は常にユーザ名とパスワードの両方が必要なのでしょうか? パスワードだけを使いたいといった場合、どうすればよいのでしょうか?

A : 常にユーザ名とパスワードの両方が必要ということはありません。もしパスワードだけでいいのであれば、グローバル変数 \$_SERVER['PHP_AUTH_PW'] だけを集中的にチェックすれば良いのです。この変数をどうやってチェックするかについての詳細は、もう少々お待ち下さい…

Q : HTTP認証でページを保護するには、ちゃんと言うとどうやるんですか? PHP関数を呼び出すのでしょうか?

A : その通りです。HTTP認証では、ブラウザとサーバとの間でHTTPヘッダを通して一連の通信が発生します。ここでヘッダというのは、ブラウザとサーバとの間でのちょっとした会話と考えても良いでしょう。ブラウザとサーバは、ヘッダを使ってPHPコード以外の箇所で通信することがよくあります。でもPHPでヘッダを送ることも間違いなくできます。これによりHTTP認証が通ります。これから、ヘッダについてより深く掘り下げて、PHPによるHTTP認証におけるヘッダの役割について見ていきます。

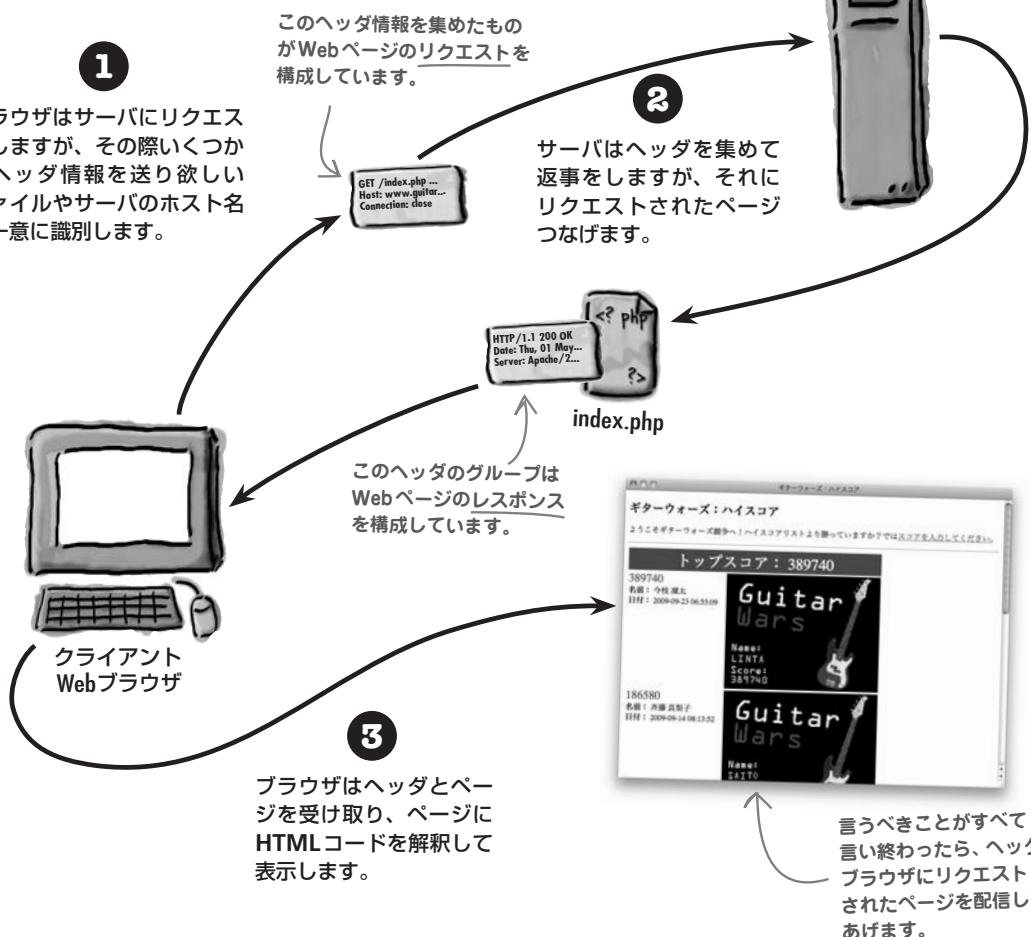


[†] 訳注：当然ながら、どちらか一方を間違えただけでも認証は通りません。

HTTP認証にはヘッダが必要です

HTTP認証のアイデアはサーバが保護されたWebページをサーバで保留するというものです。そしてブラウザがユーザにユーザ名とパスワードを入力するよう促すのです。ユーザがこれら的情報を正しく入力したら、ブラウザは先に進むことができ、ページを送ってもらえます。ブラウザとサーバ間でやり取りするこのダイアログは、**ヘッダ**を通して行われます。このヘッダというのは、ちょっとしたテキストメッセージで、リクエストや配信などの特定の指示を表します。ヘッダはWebページを訪ねていくと実は毎回使われています。認証が必要ないときもです。以下に通常の保護されていないWebページがサーバからブラウザにどのように配信されるかをヘッダの役割とともに示します。

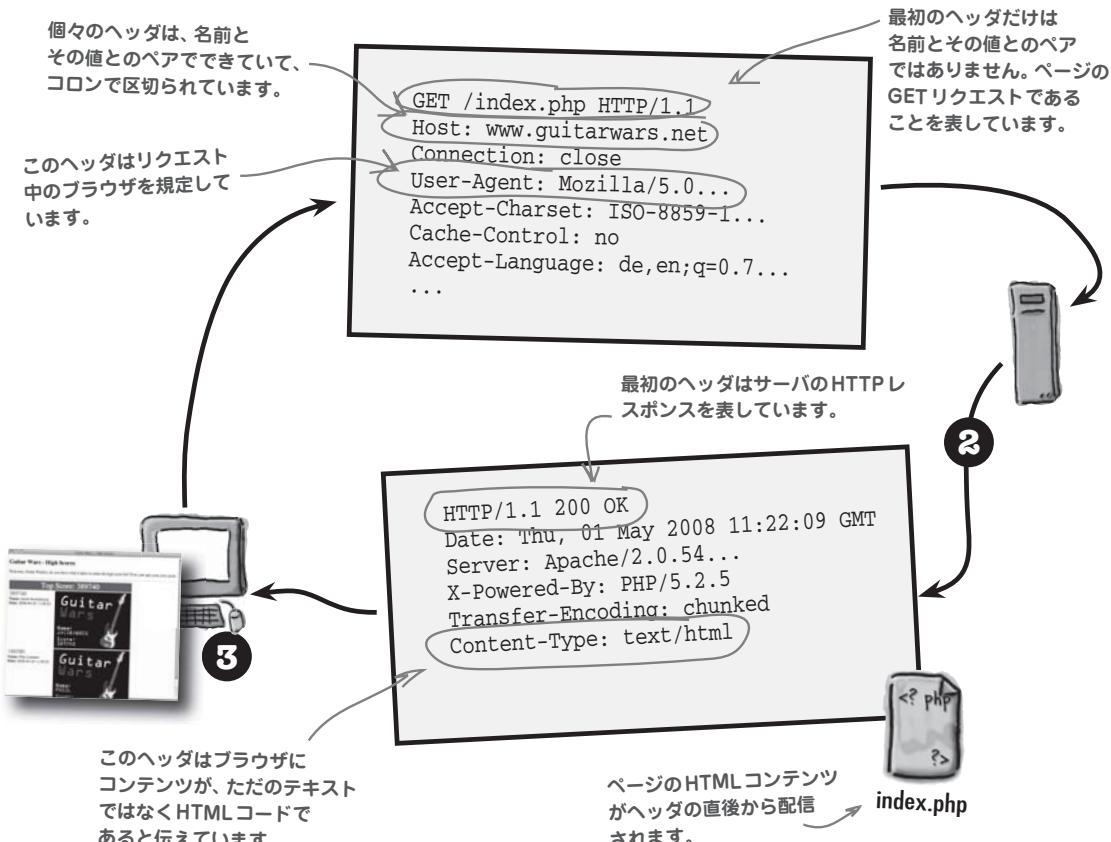
すべてのWebページ
はヘッダの助けを借り
て配信されます。





ヘッダ解剖学

ヘッダはWebブラウザとWebサーバとの間で、どのような情報をどうやってやり取りするかを厳密にコントロールします。個々のヘッダは通常、名前と値のペアで構成され、各情報を一意に識別します。例えばWebページ(HTML)のコンテンツタイプなどです。特定のグループのヘッダは、Webページリクエストの一部としてサーバに送られ、その後別のグループがレスポンスの一部としてブラウザに返されます。これらグループのヘッダを詳しく見ていくことで、サーバ・クライアント間通信でお互いに何が送られているのかを正確に把握することにします。



ヘッダがギターウォーズに関連して重要となるのは、サーバからのページ配信を中断するメカニズムを提供してくれるからです。同時にユーザに対してユーザ名とパスワードの入力を求め、その後ページが配信されます。言い換えれば、サーバから返ってくるヘッダに手を加えれば、HTTP認証を使ってページを保護することができます。



ヘッダを丸裸にする

今週のインタビュー：不満に思っていることは何でもどうぞ。

インタビュア：Webページを認証することになって俄然注目を集めているようですが…これは正当な根拠があつてのことなのでしょうか？それとも東の間の名声が欲しいだけということはないのでしょうか？

ヘッダ：もちろん正当な根拠があります。信じられないかもしれません、ヘッダがすべてのWebページ配信に必ず存在し重要な役割を演じているということを軽視しそうではないでしょうか？ヘッダがなければWebは全く動かなくなってしまうと言つても言い過ぎではありません。ヘッダはWeb界隈に東の間などというには遙かに長時間関わっています。もっともあまり正当に評価されているとも思えませんが。

インタビュア：ではヘッダが演じている役割とは厳密には何なのでしょうか？

ヘッダ：まずご理解頂きたいのですが、WebブラウザとかWebサーバといふのは人ではないということです。そのため単に電話で呼び出したり、単にテキストメッセージを送つたりするといったことはできないのです。

インタビュア：OMG(おー・まい・がっ)！

ヘッダ：でもそんなんです。ちょっとショックですが、マシンは人と同じようなコミュニケーションはできないのです。でもブラウザとサーバは通信しなければならないので、ヘッダが必要となるのです。

インタビュア：では、どのように動くのでしょうか？

ヘッダ：誰かがWebページでURLを打ち込んだりリンクをクリックしたりすると、ブラウザは、GETリクエストを組み立てて、これをサーバに送りつけます。このリクエストはいくつかのヘッダをパッケージにしたもので、それぞれのヘッダにはリクエストに関する情報が書いてあります。例えばリクエスト中のページ名やホスト名、リクエスト元のブラウザタイプ、といった情報が書いてあります。

インタビュア：まだよく分からぬのですが、それがどうして重要なのでしょうか？

ヘッダ：そうですねえ。コーヒーショップに行ったとして、何を注文するか、例えばバニラ・エスプレッソミルク付きのラージサイズを注文するといった情報は重要ですよね。

インタビュア：当然です。何を注文したいと思っているのか伝える必要がありますよ。

ヘッダ：同じことがここでも言えるのです。ブラウザはサーバに対して何を望んでいるのか、リクエストをパッケージにして、ヘッダで送るのです。

インタビュア：なるほど。でも聞くところによると、サーバも同様にヘッダを送れるそうですね。サーバは単にWebページを返すだけなのかと思っていました。

ヘッダ：お、鋭いですね。ヘッダはサーバ側からの通信においても同様に重要です。なぜならサーバは、ブラウザにコンテンツの塊を単に投げつければいい、というわけではないからです。ブラウザはコンテンツだけをもらってもどうして良いか分からないので、それを解くカギが必要なのです。

インタビュア：例えばどのようなものでしょう？

ヘッダ：1つの例としてはコンテンツのタイプがあります。これは恐らく最も重要な情報だと思います。でもサーバは他にも一緒にコンテンツのサイズとか、送信の日時などといった情報も送っています。

インタビュア：Webページ自体はいつ送られてくるのでしょうか？

ヘッダ：サーバがブラウザにヘッダを送った直後です。ヘッダに続けてコンテンツの実体が来ますが、それはHTMLのコードであつたり、PDFデータであつたり、GIFやJPEGなどのイメージデータだつたりします。

インタビュア：なるほど。ヘッダというものが通常のWebページに関係してどのような働きを持っているのか、だんだん見えてきました。ところで認証についてはどうなのでしょう？

ヘッダ：認証付きのWebページでも同じ役割を演じています。通常のWebページと異なる点は、ブラウザに対して、そのページには認証を通さなければならないと教えてあげるという役割を担っていることです。このためブラウザはユーザに対して認証に関する情報を要求することになるのです。

インタビュア：つまりユーザ名とパスワードのことですか？

ヘッダ：その通りです。その後はサーバ上のPHPコードの任務で、ユーザ名とパスワードがマッチするかどうかを判断します。マッチする場合は、サーバは処理を進め残りのページを送ってくれます。

インタビュア：すばらしいですね。スッキリしました。ありがとうございます。

ヘッダ：いえいえ。仕事ですから。

PHPでヘッダをコントロールする

PHPを使えば、サーバからブラウザへ送られるヘッダを、どのようにでもコントロールすることができます。HTTP認証のようなヘッダ駆動型の任務を遂行するための扉を開いてくれます。組み込みのheader()関数が、PHPスクリプトの中で、サーバからブラウザへ送られるヘッダを制御します。

```
header ('Content-Type: text/html');
```

header()関数はサーバからブラウザへ、ヘッダを真っ先に送りつけます。ですからそれ以前にはブラウザにいかなるコンテンツも送りつけてはいけません。これは極めて厳格な要請です。たった一文字でも空白でもヘッダに先立って送りつけてしまったら、ブラウザにエラーで、はじかれてしまいます。このため、header()関数の呼び出しが、PHPスクリプトの中で、あらゆるHTMLコードの前に置かれていなければなりません。コード系にUTF-8を用いる場合は、ファイル形式をUTF-8Nとする必要があります。単にUTF-8という形式だと、BOMと呼ばれるコードがファイル先頭に付与されるため、エラーではじかれます[†]。

間違って<?phpタグの前に
空白があったとしても、
この例のスクリプトは
エラーの原因となります。<?php

<?php ?> タグの
内側の空白は問題には
なりません。この部分
はブラウザには送られ
ないからです。

ファイルを保存するときは
UTF-8Nを指定して下さい。
単にUTF-8ではダメです。

header ('Content-Type: text/html');

...
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja" >
...
</html>

サーバはこのヘッダをブラウザに
送りつけますが、同じページの
あらゆるHTMLコンテンツを
送る前でなければなりません。

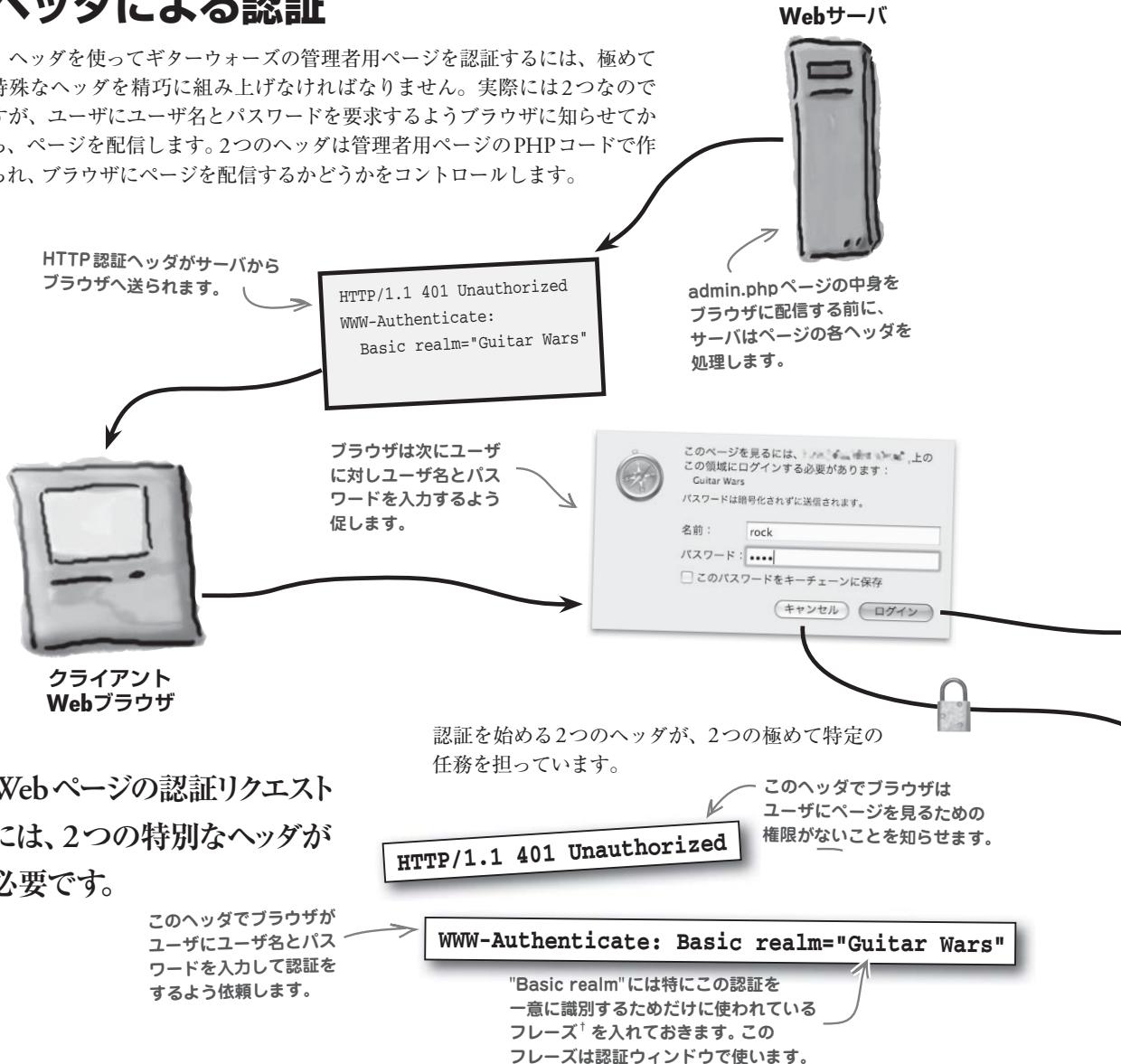
ヘッダ関連の部品はみんなステキ
だけど、実際に認証でページを保護する
にはどうやって使ったらいいの？

[†] 訳注：日本語を処理するために必須事項
ですので追記しました。

header()関数は
PHPスクリプトから
ヘッダを作成して送る
機能を持っています。

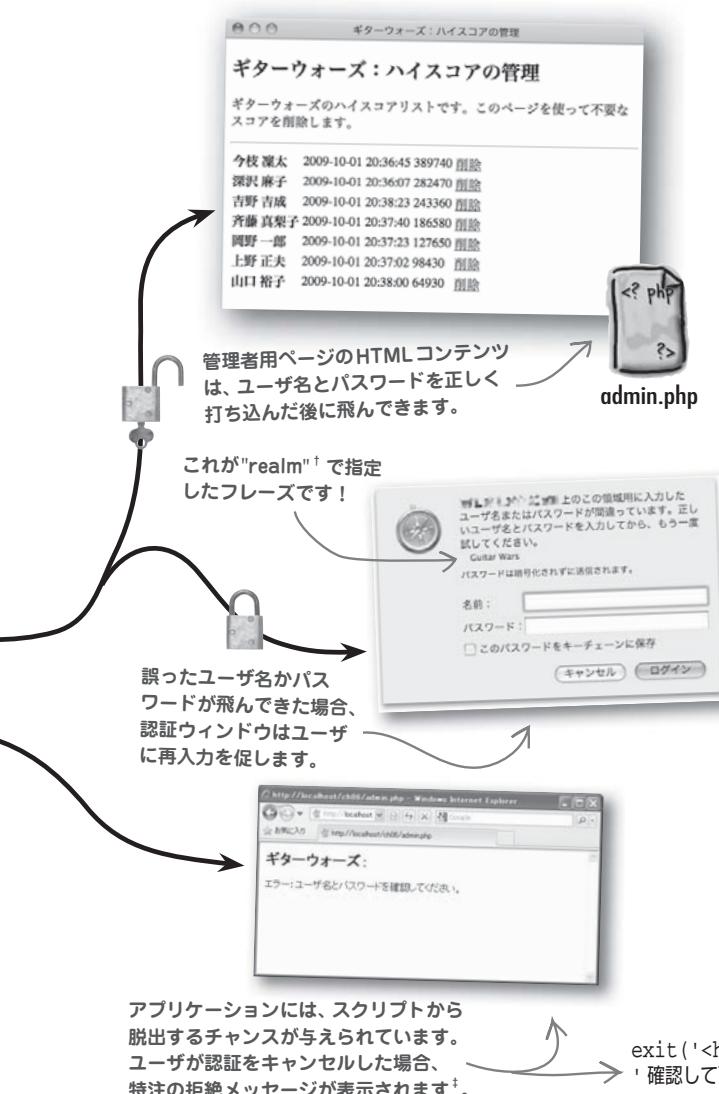
ヘッダによる認証

ヘッダを使ってギターウォーズの管理者用ページを認証するには、極めて特殊なヘッダを精巧に組み上げなければなりません。実際には2つなので、ユーザにユーザ名とパスワードを要求するようブラウザに知らせてから、ページを配信します。2つのヘッダは管理者用ページのPHPコードで作られ、ブラウザにページを配信するかどうかをコントロールします。



† 訳注：このフレーズ（ここではGuitar Wars）の部分に日本語文字を入れることはできません。アルファベットか数字のみと考えて下さい。

認証ヘッダの処理が終わると、ブラウザはユーザが認証ウィンドウに対して操作を行うの待ちます。ブラウザはユーザの行為に応じて劇的に異なる動作をします…



ユーザが正しいユーザ名とパスワードを打ち込んで、「ログイン」をクリックした場合、サーバは admin.php ページの HTML コンテンツをブラウザに送ってくれます。ブラウザは管理者用ページを表示し、ユーザは従来の保護されていないバージョンの時と全く同様にスコアを削除することができます。



ユーザが誤ったユーザ名とパスワードを打ち込んで「ログイン」をクリックした場合、サーバはブラウザに再度認証画面を表示するように指示します。ブラウザはユーザが誤ったユーザ名・パスワードを入力すると、果てしなくこの処理を繰り返します。言い換えれば、ユーザがユーザ名とパスワードをちゃんと知らないのであれば、残された道は「キャンセル」をクリックすることだけです。



ユーザが「キャンセル」ボタンをクリックし、認証ウィンドウから脱出しようとすると、サーバはブラウザに拒絶メッセージのページを送りつけます。それ以外には何もしません。admin.php ページは送られてきません。拒絶メッセージは、admin.php スクリプトの中の PHP コードでコントロールすることができます。このコードはヘッダと非常に密接に関連しています。このコードは PHP の exit() 関数を呼び出して、メッセージを表示し、スクリプトを即終了してしまいます。



exit ('<h2> ギターオーズ </h2> エラー：ユーザ名とパスワードを' .
' 確認して下さい。');

† 訳注：あえて訳しませんでしたが、領域、範囲といった意味です。ルムと発音します。

‡ 訳注：Mac の標準ブラウザ Safari（バージョン 4.0.2 および 4.0.3 時点）では、このメッセージは表示されず、遷移前の画面のままとなります。Safari のバグと思われます。代わりに IE のメッセージを載せてあります。



PHPマグネット

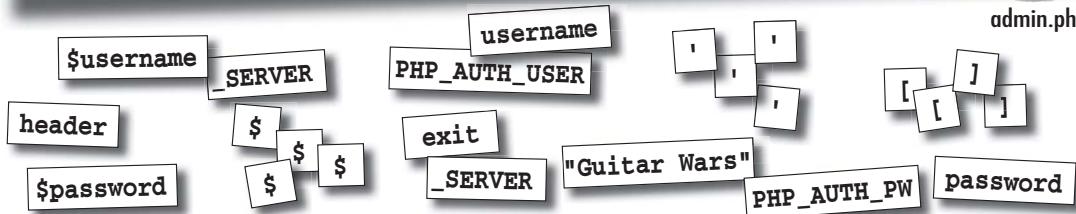
以下のギターウォーズ管理者用スクリプトには、PHPコードでHTTP認証をする上で非常に重要な部分がいくつか抜けています。下のマグネットを使って抜けているコードを埋め、ヘッダを使って管理者用ページにセキュリティを与えて下さい。ヒント：二度以上使うマグネットもあります。

```
<?php
// User name and password for authentication ユーザ名とパスワードによる認証
..... = 'rock';
..... = 'roll';
if (!isset( .....)) ||
!isset( .....)) ||
($_SERVER['PHP_AUTH_USER'] != ..... ) || ($_SERVER['PHP_AUTH_PW'] != ..... )) {
// The user name/password are incorrect so send the authentication headers
// HTTP/1.1 401 Unauthorized';
.....('WWW-Authenticate: Basic realm= ..... ');
.....('<h2> ギターウォーズ :</h2> エラー：ユーザ名とパスワードを確認して下さい。');
}
?>

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
...
</html>
```



admin.php



素朴な疑問に答えます

Q: UTF-8とUTF-8Nとはどう違うのですか？

A: UTF-8Nというのは日本固有の呼称です。ファイルの先頭にBOMと呼ばれる3バイトの特殊コードが埋め込まれているファイルをUTF-8と呼んでいます。日本では日本語を表記するためのさまざまなコードが存在するため、ファイルにBOMが付いている場合にUTF-8と判断することがあります。一方ファイルの先頭にBOMが付いていないファイルのことをUTF-8Nと呼びます。コード系がUTF-8であると決まっている場合は、これでも問題はないわけです。

今問題なのはヘッダの前には何もあってはならないという要請です。BOM付きのUTF-8ではこの要請に違反してしまうため、UTF-8Nでなければならないのです。

Q: ではUTF-8Nのファイルはどうすれば作れるのですか？

A: 残念ながらWindowsに標準で用意されているメモ帳やワードpadでは作れません。TeraPadやxyzyなどUTF-8Nに対応したフリーのエディタを使って作ることができます。付録iiiを参照してインストールしてください。Macの場合は、テキストエディタでそのまま作れます。MacではUTF-8がUTF-8Nのことを表しています。



PHPでいろんな種類のヘッダを送りつけることができるのかな?

全くその通りです…ヘッダはセキュリティだけのものではありません。

認証は、今正に必要とするヘッダの機能ですが、ヘッダには実に柔軟性があり、面白いことが各種できます。単にheader()関数を適切な名前と値のペアをくっつけて呼び出すだけでいいのです。こんな感じです。

ブラウザはこのヘッダを受け取ると「説明」(about)のページにリダイレクトされます。

```
<?php
header('Location: http://www.guitarwars.net/about.php');
?>
```

このヘッダは**Location**ヘッダといって、ギターウォーズのサイト内で現在のページをabout.phpという名のページにリダイレクトします。次も同様のヘッダですが、5秒後にabout.phpにリダイレクトします。

ブラウザは「説明」のページに5秒後にリダイレクトします。

```
<?php
header('Refresh: 5; url=http://www.guitarwars.net/about.php');
echo '5秒後に『説明』のページに遷移します。';
?>
```

このヘッダは**Refresh**ヘッダといいます。特定の時間が経過した後ページをリフレッシュするからです。ヘッダのURLで自身のページを参照するものが多くあります。これにより自分自身をリフレッシュすることができます。最後にもう1つ**Content-Type**ヘッダというものを紹介します。このヘッダはサーバから送りつけられるコンテンツのタイプをコントロールするものです。例として、ページを単なるテキストであってHTMLではない、というよう

に強要することができます。header()関数を呼び出す時に次のようなヘッダを使います。

コンテンツはブラウザに対して単なる
テキストとして配信されます。

```
<?php
header('Content-Type: text/plain');
echo 'この部分の<strong>テキスト</strong>は実際には強調されません。
';
?>
```

この例では、テキストはechoでブラウザにそのまま表示され、特別なフォーマット整形は一切行われません。言い換えれば、サーバはブラウザに対して、echoしたコンテンツをHTMLとして加工するなど伝えているのです。このため、HTMLタグはテキストとして文字通りそのまま表示されることになります。



要注意

ヘッダは、PHPファイルの中で一番最初にブラウザに送りつけなければなりません。

ヘッダはあらゆるコンテンツに先立って送らなければなりません。このため極めて重要な要請が課せられています。PHPスクリプトのheader()関数を呼び出す前には、たとえ空白1つであっても、PHPコードの外側にあることは許されないのです。

従って、くどいようですがUTF-8のファイルはダメでUTF-8Nです。



PHPマグネットの答え

以下のギターオーズ管理者用スクリプトには、PHPコードでHTTP認証をする上で非常に重要な部分がいくつか抜けています。下のマグネットを使って抜けているコードを埋め、ヘッダを使って管理者用ページにセキュリティを与えて下さい。ヒント：二度以上使うマグネットもあります。

```
<?php
// User name and password for authentication
$username = 'rock';
$password = 'roll';

if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW']) ||
    ($_SERVER['PHP_AUTH_USER'] != $username) || ($_SERVER['PHP_AUTH_PW'] != $password)) {
    // The user name/password are incorrect so send the authentication headers
    header('HTTP/1.1 401 Unauthorized');
    header('WWW-Authenticate: Basic realm= "Guitar Wars"');
    exit('
        <h2> ギターオーズ : </h2> エラー：ユーザ名とパスワードを確認して下さい。');
}

?>

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
    ...
</html>
```

HTMLコードは、ヘッダが送られて処理されるまではブラウザに配信されません。



脳力発揮

HTTP認証に関するコードをUTF-8とUTF-8Nのそれぞれで保存し、どのような違いがあるか比較してみてください

`$_SERVER`スーパーグローバルで認証ウィンドウにユーザが打ち込んだユーザ名とパスワードにアクセスすることができます。

ユーザが打ち込んだユーザ名とパスワードをチェックし、正しいかどうかを判定します。

header()関数の2回の呼び出しにより、このようなヘッダがブラウザに飛びます。

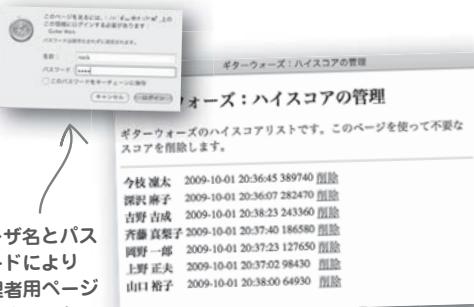
HTTP/1.1 401 Unauthorized
WWW-Authenticate:
Basic realm="Guitar Wars"



管理者用スクリプトにHTTP認証を追加する

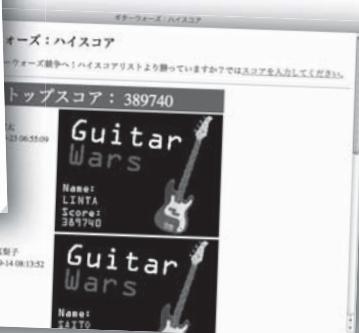
admin.phpスクリプトを修正して、HTTP認証により自分だけがアクセスできるようにします。

スクリプトをWebサーバにアップロードしたら、Webブラウザでページを開きます。まず間違った名前とパスワードを入れてみて、アクセスがどのように制限されているか確認してみて下さい。



ユーザ名とパスワードにより管理者用ページへの権限のないアクセスを防ぐことができます。

スコアは認証を通らなければ消すことができません。



ギターオーズの参加者は、これでハイスクアアブリケーションが安全にセキュリティが保たれるようになって驚喜しています！



素朴な疑問に答えます

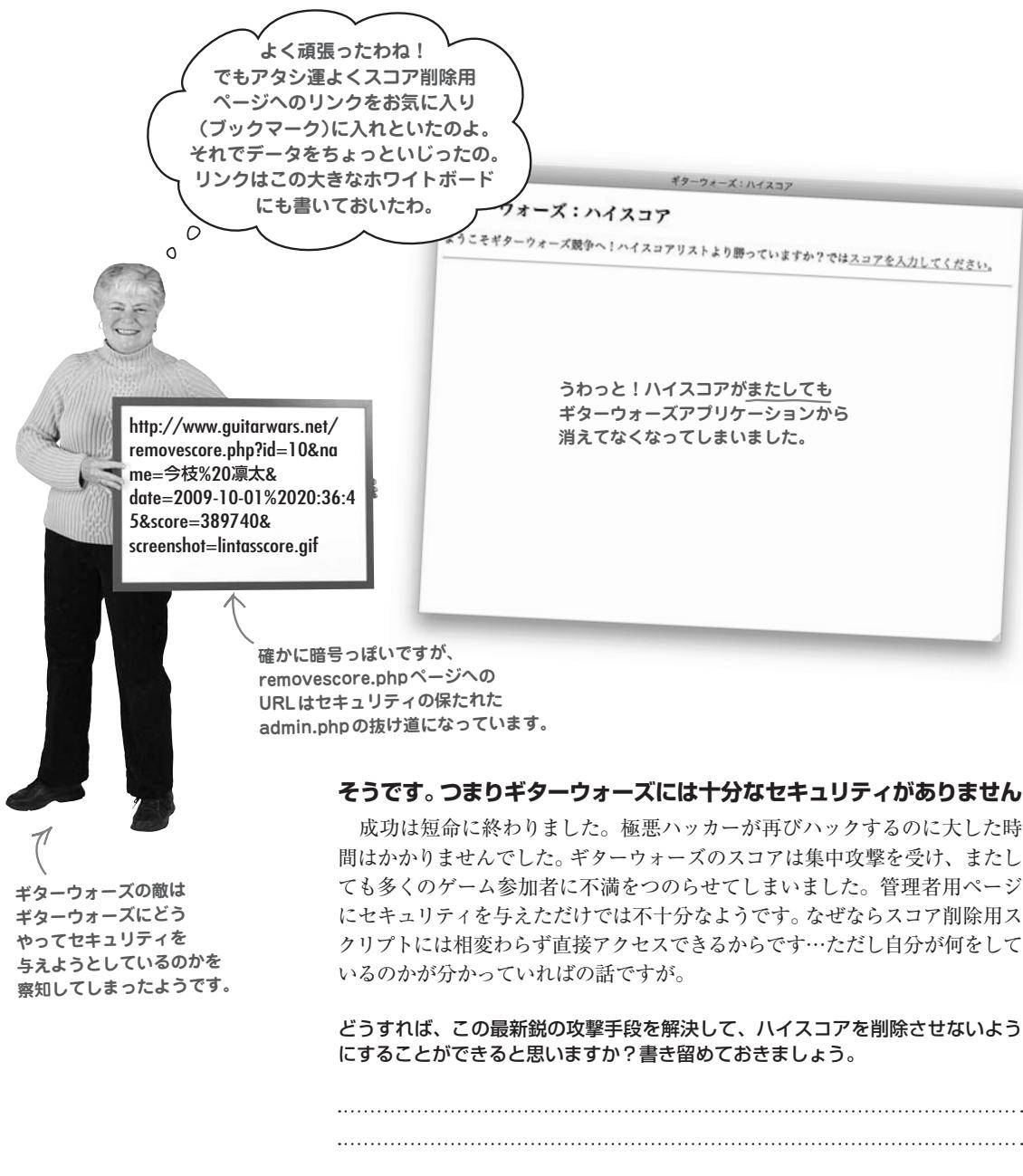
Q: `exit()` 関数がギターオーズの管理者用スクリプト内で呼び出されるのは正確にはいつですか？

A: `exit()` 関数は `header()` 関数の二度の呼び出しのすぐ下に書いてありますが、実際に呼び出されるのは、「キャンセル」ボタンをクリックすることによりユーザが認証ウインドウをキャンセルした場合のみです。認証に失敗した場合、サーバは2つの `header()` 呼び出しに戻って実行を繰り返すではありません。その代わりに、ヘッダを再送して処理を繰り返します。ユーザが「キャンセル」をクリックした場合に限り、サーバは `exit()` 関数を実行します。この場合サーバは関数呼び出しの中にあるコンテンツだけを送りつけ、それ以外は何もしません。認証に成功した場合は、`exit()` は、呼び出されません。なぜならスクリプトは `if` 文に入ることは決してないからです。`if` 文の内側にあるコードは、ユーザ名とパスワードがセットされていない場合か、または間

違って入力された場合に限られます。ただしMacの標準ブラウザSafari(バージョン4.0.3時点)では、呼び出されなくなりました。

Q: HTTP認証の「Basic realm」には他に何か目的があるのでしょうか？

A: あります。セキュリティゾーンというものを定義して、特定のユーザ名とパスワードを保護することができます。ユーザ名とパスワードが与えられたrealm(領域)に正しく入力されると、ブラウザはそれを覚えておいて、同一のrealmでは後続する認証ヘッダに対する認証ウンドウを表示しないようにします。言い換えれば、realmはブラウザがいくつかのページに対するセキュリティ上の要請を満たしたことを覚えておくためのものです。それぞれのページに対して単に同じrealmを認証ヘッダに指定すればよいのです。





スコア削除用スクリプトにも
セキュリティが必要ね。あつたり前だけど
HTTP認証をまた使えばできちゃうわ。

Jちゃん：そりゃそうだね。つまり、管理者用ページでちゃんと動いたからね。

森川氏：そうだね。だったらやるべきことは、同じようにヘッダ認証コードをスコア削除用スクリプトに置いとけばいい。そうすれば幸せだね。

沢田さん：そうね。確かに動くわね。だけど2箇所に認証のコードを重複させちゃうのってどうなのかなしら。後で別のページを追加することになって、保護される必要があるとしたらどうなるの？またまたコードを重複させちゃうわけ？

Jちゃん：コードの重複は確かに問題だね。特にユーザ名とパスワードを全部のスクリプトで共有しなきゃいけなくなることがね。この2つを変えたいと思ったら、全部の保護されたスクリプトを変えなきゃならなくなる。

森川氏：分かった！じゃあ\$usernameと\$passwordという変数をインクルードファイルに入れといいて、保護されたスクリプト間で共有するっていうのはどう？アプリケーション変数用のappvars.phpインクルードファイルに入れておくことだってできるよ。

Jちゃん：方向性は悪くないと思うけど、それはコードの重複という問題の極めて小さい一部分を処理しているだけだよ。今問題にしているのは、それなりの大きさの数行のコード全部なんだ。

```
<?php
// User name and password for authentication ユーザ名とパスワードによる認証
$username = 'rock';
$password = 'roll';

if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW']) ||
    ($_SERVER['PHP_AUTH_USER'] != $username) || ($_SERVER['PHP_AUTH_PW'] != $password)) {
    // The user name/password are incorrect so send the authentication headers
    header('HTTP/1.1 401 Unauthorized');
    header('WWW-Authenticate: Basic realm="Guitar Wars"');
    exit('<h2>ギターオーズ:</h2> エラー：ユーザ名とパスワードを確認して下さい。');
}
?>

<html>
```

ユーザ名 / パスワードが不正なため認証ヘッダを送信



admin.php

沢田さん：二人とも正しいわ。だから新しいインクルードファイルが必要で、そこに認証コードを全部入れとけば良いんじゃないかなしら。\$usernameと\$passwordだけじゃなくて。

森川氏：あ、つまりHTTP認証で保護したいページは全部そのスクリプトをインクルードすれば良いんだ。

Jちゃん：その通りだね！大事なのは一番最初にスクリプトをインクルードするっていうことだね。HTTP認証の全要素はヘッダに依存しているから。

認証用スクリプトを作る

新しく作る認証用スクリプトに必要なすべてのコードがすでにあります。単純にコードを admin.php から新しいスクリプトファイル (authorize.php) に移すだけのことです。あとはもともとのコードがあった場所を require_once 文で置き換えるべきです。

このコードを admin.php から引っ張ってきて、
単独のスクリプトファイル
authorize.php に入れれば
良いのです。

```
<?php
    // User name and password for authentication
    $username = 'rock';
    $password = 'roll';
    if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW']) ||
        ($_SERVER['PHP_AUTH_USER'] != $username) || ($_SERVER['PHP_AUTH_PW'] != $password)) {
        // The user name/password are incorrect so send the authentication headers
        header('HTTP/1.1 401 Unauthorized');
        header('WWW-Authenticate: Basic realm="Guitar Wars"');
        exit('<h2>ギターウォーズ:</h2> エラー：ユーザ名とパスワードを確認して下さい。');
    }
?>

<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
    <title>ギターウォーズ：ハイスクアの管理</title>
    <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
    <h2>ギターウォーズ：ハイスクアの管理</h2>
    <p>ギターウォーズのハイスクアリストです。このページを使って不要なスコアを削除します。</p>
    <hr />

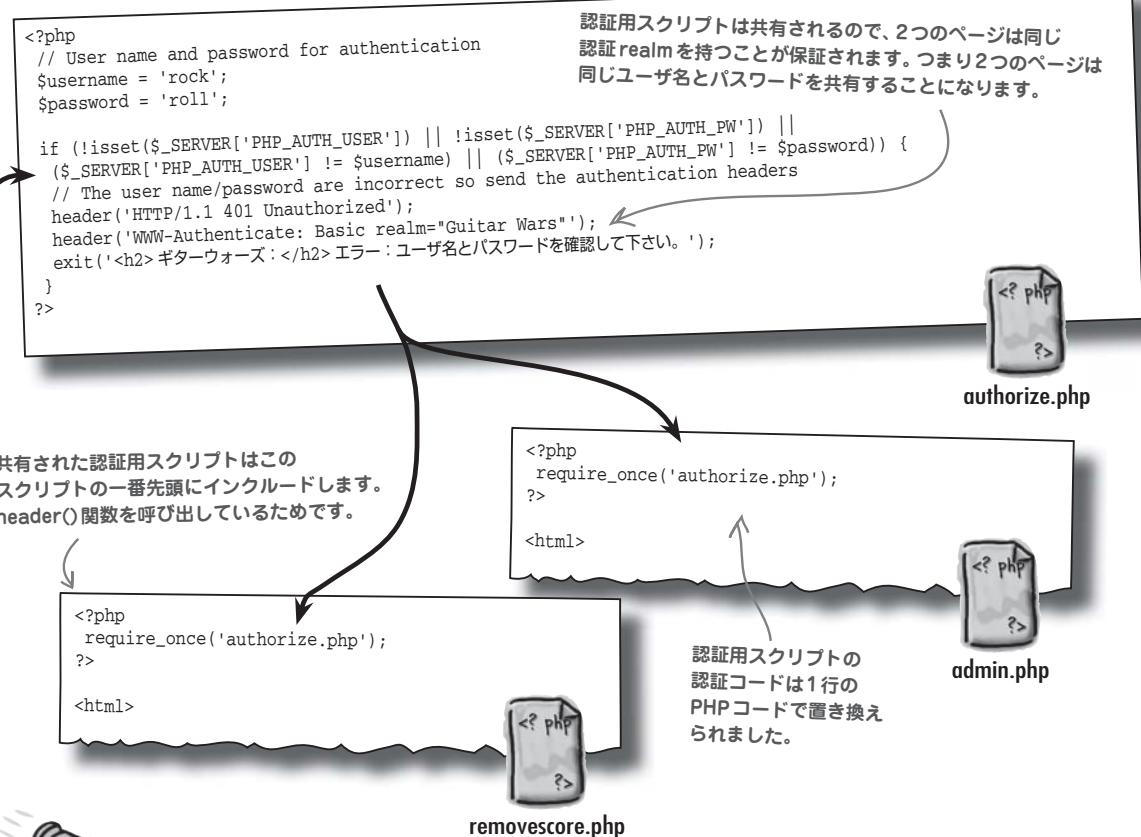
    <?php
        require_once('appvars.php');
        require_once('connectvars.php');
        // Connect to the database
        $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
        // Retrieve the score data from MySQL
        $query = "SELECT * FROM guitarwars ORDER BY score DESC, date ASC";
        $data = mysqli_query($dbc, $query);
        // Loop through the array of score data, formatting it as HTML
        echo '<table>';
        while ($row = mysqli_fetch_array($data)) {
            // Display the score data
            echo '<tr class="scorerow"><td><strong>' . $row['name'] . '</strong></td>';
            echo '<td>' . $row['date'] . '</td>';
            echo '<td>' . $row['score'] . '</td>';
            echo '<td><a href="removescore.php?id=' . $row['id'] . '&date=' . $row['date'] .
                '&name=' . $row['name'] . '&score=' . $row['score'] .
                '&Screenshot=' . $row['Screenshot'] . '">削除</a></td></tr>';
        }
        echo '</table>';

        mysqli_close($dbc);
    ?>

</body>
</html>
```



admin.php



重要ポイント

- PHPスクリプトはヘッダを使って、サーバがどのようにWebコンテンツをブラウザに送りつけるかをコントロールすることができます。
- PHPの組み込み関数`header()`は、ヘッダをブラウザに飛ばすときに使います。ヘッダを使うとページのリダイレクト、ページのContent typeのコントロール、ページ認証のリクエストなどができます。
- ヘッダをブラウザに`header()`関数を使って送りつけるには、まず`header()`関数を、他のあらゆるコンテンツ送信に先立って呼び出さなければなりません。
- ページがHTTP認証で保護されていると、ユーザが打ち込んだユーザ名とパスワードが`$_SERVER`スーパーグローバルにぶち込まれます。
- HTTP認証の「Basic realm」は、セキュリティゾーンといって、特定のユーザ名とパスワードに関連付けられており、複数のページをひとくくりで保護することができます。
- PHPの組み込み関数`exit()`は、PHPスクリプトから抜け出すことによって、それ以降のコードが実行されたり、ブラウザに送りつけられたりすることを防ぎます。

素朴な疑問に答えます

Q: まだよく分からぬのですが、赤羽さんがギター ウォーズのセキュリティに関する情報をどうやって得たのでしょうか? 彼女は何をしたのでしょうか?

A: 彼女は1つのページ(管理者用)だけを守ったという構造上の弱点につけ込んだのです。スコアを削除する機能は、本来は2つのページ(管理者用とスコア削除用)に依存しています。管理者用ページは「削除」リンクを表示し、スコア削除用ページにリンクしています。どのスコアを削除するかに関する詳細は、URLで渡されます。スコア削除用スクリプトは、これらの情報に\$_GETスーパーグローバルを通してアクセスすることができます。スコア削除用ページの正しいURLが分かったとしたら、管理者用ページを通り抜けてスコアを削除することができてしまいます。これが赤羽さんのやったことです。

Q: でも彼女はスコア削除用ページへのURLをどうやって作ったらよいかを、どうして知ることができたのでしょうか?

A: 彼女は極めて悪賢いのです。でもこれは天才でなくともできる仕事です。彼女はスコア削除用ページを、まだサイトに全然セキュリティがないときに、お気に入り(ブックマーク)に入れたと言っていたのを覚えていますか? 実はお気に入りというのは、単にURLです。ですから彼女はこれを使ってスコア削除用ページに直接アクセスするURLを作り上げたわけです。これなら管理者用ページを経由しなくても済みます。

Q: わかりました。でもハイスコアは以前攻撃を受けた後、再入力されてますよね。だったら昔のURLでは動かないのではないかでしょか? 日付とか違いますよね?

A: その通りです。良い質問です。でも覚えていますか? 赤羽さんはとても狡猾なのです。彼女はギターウォーズのメインページを簡単に見ることができますから、新しい日付が分かります。これを古いURLに突っ込んであげれば、何の問題もなく新しいスコアを削除できます。大事なことは、PHPスクリプトをリバースエンジニアで解読して弱点を暴き出そうとするような、一部の人たちの能力を決して過小評価してはいけないということです。

Q: 納得です。では管理者用とスコア削除用の両方のページを保護して赤羽さんを食い止めるのですね。でもこれだと正当にスコアを削除することに対してまるっきり混乱してしまうのではないかでしょか?

A: そんなことはありません。全然大丈夫です。realmの助けを借りなければ、正しくスコアを削除することに関して、間違いなく混乱を来たしたかもしれません。用户名とパスワードを管理者用とスコア削除用の両方でそれぞれ打ち込まなければならないからです。でもrealmが両方のページで共通に確立されているということを思い出してください。これは2つのページが同じセキュリティゾーンに入っているということを意味しています。つまり一度あるページの認証ウインドウを、特定のrealmで通れば、その用户名とパスワードはそのrealmで記憶されています。結論としては、用户名とパスワードを正しく一度入力するだけで両方のページのロックが外れます。

Q: realmのフレーズに日本語を使う方法はないのでしょうか?

A: 不可能ではないのですが、ブラウザ依存になってしまいます。realmのフレーズを表示するのは、実際にはブラウザなのですが、これはユーザーの環境です。PHP(サーバ側)でコントロールすることはできません。アルファベットと数字であれば、どのようなブラウザでも表示できることは保障されていますから、これらの文字だけを使ってフレーズを作ったほうが良いでしょう。



PHPスクリプトをリバースエンジニアで
解読して弱点を暴き出そうとするような、
一部の人たちの能力を決して過小評価
してはいけません。



認証用スクリプトを作り、これを管理者用とスコア削除用の両方にインクルードしてセキュリティを与えます。

新しくテキストファイルを作り `authorize.php` という名前にします。そこに認証用スクリプトのコードを打ち込みます。次に `admin.php` スクリプトを変更し、実際の HTTP 認証のコードの代わりに認証用スクリプトをインクルードします。同様に `require_once` 文を `removescore.php` スクリプトの先頭に追加し、このスクリプトも HTTP 認証で保護されるようにします。

すべてのスクリプトを Web サーバにアップロードしたら、スコア削除用スクリプトを Web ブラウザで直接開いてみて下さい。もしかしたら以前の HTTP 認証セッションをブラウザで、すべて削除[†]しなければ認証ウィンドウは現れないかもしれません。大抵のブラウザは認証の `realm` を覚えていて、ユーザ名とパスワードを再入力しなくても良いようになっています。

[†] 訳注：Windows IE では、「ツール >> 履歴の削除」または「ツール >> インターネットオプション >> 全般タブ >> 閲覧の履歴 >> 削除」で削除できます。Mac Safari では、「Safari >> キャッシュを空にする」または「Safari >> Safari をリセット…」で削除できます。

ギターウォーズ：ハイスコア

ギターウォーズ：ハイスコア

ユーザ名とパスワードは、管理者用とスコア削除用の両方のページで要求されるようになります。

この URL で管理者用ページをバイパスし、スコア削除用ページに直接アクセスできます。

スコア削除用ページはユーザがそこにたどり着けたとしても保護されています。

脳力発揮

ギターウォーズのハイスコアアプリケーションが直面する可能性のある何らかのリスクを思いつきますか？

ハイスコア ギターウォーズ エピソードII：▼クローンの攻撃

残念なことに、ギターウォーズ界の幸せは長くは続きませんでした。インチキのスコアが正当なスコアに取って代わってアプリケーションを占拠するようになったためです…このため相変わらずギターウォーズ界は怒り心頭のままで。明らかにギターウォーズのハイスコアリストをかき乱すことは、スコアを削除せざとも簡単にできます。でもどうやって？

ギターウォーズ：ハイスコア

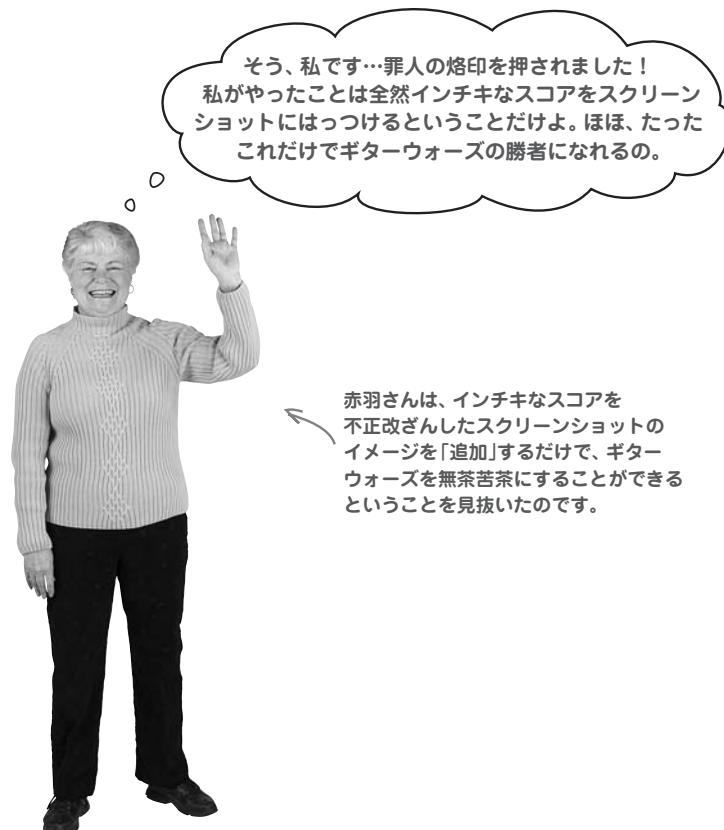
ようこそギターウォーズ競争へ！ハイスコアリストより勝っていますか？ではスコアを入力してください。

トップスコア：500000				
Name:	Score:	screenshot		
赤羽 京子	500000			
名前：赤羽 京子	日付：2009-10-02 14:02:54			
389740				
名前：今枝 凜太	日付：2009-10-01 20:36:45			
Name:	score	screenshot		
21	2009-10-01 20:36:07	深沢 麻子	282470	asakosscore.gif
22	2009-10-01 20:36:45	今枝 凜太	389740	lintasscore.gif
23	2009-10-01 20:37:02	上野 正夫	98430	uenoosscore.gif
24	2009-10-01 20:37:23	岡野 一郎	127650	okanosscore.gif
25	2009-10-01 20:37:40	斎藤 真梨子	186580	saitosscore.gif
26	2009-10-01 20:38:00	山口 裕子	64930	yukosscore.gif
27	2009-10-01 20:38:23	吉野 吉成	243260	yoshinosscore.gif
28	2009-10-01 21:14:56	西原 仁美	308710	nishisscore.gif
29	2009-10-01 21:15:17	鈴木 剛志	354190	suzukisscore.gif
30	2009-10-02 14:02:54	赤羽 京子	500000	akabanesscore.gif

赤羽さんのトップスコアは明らかに疑わしいものです。スクリーンショットがチープな不正改ざんによることは明白ですし、スコアがぴったり500,000であるということも不自然です。

足し算による引き算

今までスクリーンショットイメージ付きで「追加」されたハイスコアは妥当なものであるという仮定の下に動いてきました。しかしそのような仮定は正しくなかったといつても差し支えないでしょう！そして誰が容疑者かはもはや明白です…



ギターウォーズアプリケーションにインチキなハイスコアを投稿する
ことができてしまうという問題を解決するにはどうすれば良いと思
いますか？書き留めておきましょう。

セキュリティには人間が要ります

現代においても、本物に生きていて考えるもの、つまり人間を打ち負かすことはできません。この場合で言うと、情報を解析して、それが妥当かどうかを判定する本物の人間を打ち負かすことはほとんど不可能です。今話題にしているのは検査のことです。人がWebアプリケーションに投稿されたコンテンツを承認することを経てから、一般に対して公開するというわけです。

検査があれば、新しいハイスコアはデータベースには加わりますが、検査官が承認するまでは公開されません。

管理者画面を変更して、「承認」リンクを追加し、新しいハイスコアを承認できるようにします。

ギターウォーズ：新しいハイスコアの追加

名前：赤羽京子
スコア：322710
スクリーンショット：（ファイルを選択）ファイルが選択されていません
（送信）

単に新しいハイスコアを追加しただけでは、もはや自動的にハイスコアリストに公開されることはありません。

ギターウォーズ：ハイスコア管理

ギターウォーズのハイスコアリストです。このページを使って、新しいハイスコアを追加したり、既存のハイスコアを削除したりすることができます。

名前	日付	スコア	操作
赤羽京子	2009-10-02 14:02:54	500000	削除 / 承認
今枝凜太	2009-10-01 20:36:45	389740	削除

人間による検査というのは、ユーザが「追加」したコンテンツがまともかどうかを劇的に改善することのできる方法です。



ギターウォーズは、人間による検査を得ることができますようになります。確かに、まだ誰かがスクリーンショットの不正をするという可能性は残されています。人間の検査をもかいくぐってスコアを忍び込ませるかもしれません。でもそんなに簡単ではないでしょう。それに検査というのが絶大な抑止力となっているという事実には変わりありません。心に留めておいて欲しいことは、PHPアプリケーションにセキュリティを与えるということの大部分は予防だということです。

ギターウォーズを検査するための計画

ギターウォーズに人間による検査機能を追加することは重大な意味を持ちます。なぜならこれによりアプリケーションの各部分に影響が出るからです。データベースに変更が必要ですし、承認用に新しいスクリプトが必要ですし、管理者画面には各スコアへの「承認」リンクを追加しなければなりませんし、そして最後にメインページを変更して承認されたスコアのみを表示するようにしなければなりません。これら多くの変更を実行に移すには、計画をきちんと立てて各変更を1ステップずつこなしていくことが重要です。

① ALTERを使ってテーブルにapprovedカラムを追加します。

データベースから始めましょう。スコアが承認されているかどうかを保持する新しいカラムが必要です。

id	date	name	score	screenshot	approved
...					
28	2009-10-01 21:14:56	西原 仁美	308710	nishiscore.gif	0
29	2009-10-01 21:15:17	鈴木 剛志	354190	suzukisscore.gif	0
30	2009-10-02 14:02:54	赤羽 京子	500000	akabanesscore.gif	0
31	2009-10-02 20:32:54	米永 徳幸	314340	yonesscore.gif	0
32	2009-10-02 20:36:38	掛布 秀一	322710	kakefusscore.gif	0

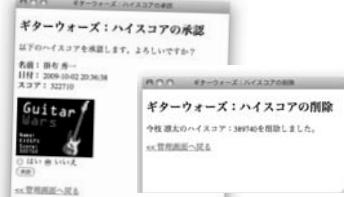
③ 管理者用ページを修正して、まだ承認されていないスコアに対して「承認」リンクをつけます。

スコア承認用スクリプトは、バックエンドのスクリプトです。つまり通常は直接アクセスできません。その代わりに承認用ページに生成され、表示されている「承認」リンクを通してのみアクセス可能です。未承認スコアのみがスコアのとなりに「承認」リンクを持ちます。



② スコア承認用スクリプトを作って、新しいハイスコアの承認処理をします(approvedカラムに1をセットします)。

データベースはハイスコア承認の準備ができたので、実際にスコア承認処理をするスクリプトが必要になります。このスコア承認用スクリプトは特定のスコアをデータベースから探し出して、approvedカラムの値を変更します。



④ メインページの問い合わせ文を変更して、承認されたスクリプトのみを表示するようにします。

最後のステップは、ここで承認したスコアのみをメインのハイスコア表示対象とすることです。つまりアプリケーションのメインページを変更し、承認されたハイスコアのみを表示します。この変更がないと、今までのすべての承認に関する修正が無意味になってしまいます。



ALTERで承認の置き場所を作る

新しくapprovedカラムをguitarwarsテーブルに追加するには、ALTER TABLE文を一度だけ使う必要があります。このALTER TABLE文というのはSQL文ですでに使ったことがあります。

ALTER TABLE guitarwars
ADD COLUMN approved TINYINT

MySQLのBOOL型と
TINYINTとは同じものです。
どちらを使っても構いません。

新しく作ったapprovedカラムはTINYINTで、0が未承認スコアを表し、1が承認済みスコアを表すことにします。ですからすべての新しいスコアは、すべて値0で初期化して、初めはすべて未承認ということになります。



① ALTERを使ってテーブルにapprovedカラムを追加します。

終了

ちょっと待って。ただデータベースに
カラムを足しただけじゃない。スコア追加用
スクリプトは変えてないけどいいの？新しいカラム
にデータをINSERTすべきじゃない？



その通りです。新しくカラムを追加したら、スコア追加用スクリプトの
INSERT問い合わせ文に新しく値が必要になります。

極めて重要なことなのですが、PHPアプリケーションというのは、いくつもの部品が複雑に絡み合っているという視点を常に持っていないなりません。部品というのは行とカラムを持つテーブルからなるデータベース、PHPコード、HTMLコード、それにCSSコードもです。一部分を変更したからといって、即座に別の部分の変更が必要になるとは限りません。新しくapprovedカラムをguitarwarsテーブルに追加して、新しくスコア承認用スクリプトに対応できるようにするということは、スコア追加用スクリプトのINSERT問い合わせ文を修正する必要があるということです。

新しくぶち込まれたハイスコア行のapproved
カラムは0にセットします…未承認です！

```
INSERT INTO guitarwars  
VALUES (0, NOW(), '$name', '$score', '$screenshot', 0)
```

id	date	name	score	screenshot	approved
...					
30	2009-10-02 14:02:54	赤羽 京子	500000	akabanesscore.gif	0
31	2009-10-02 20:32:54	米永 徳幸	314340	yonesscore.gif	0
32	2009-10-02 20:36:38	掛布 秀一	322710	kakefusscore.gif	0

新しいカラムを追加したら、
approvedカラムは0に
セットして、未承認か
始めることにします。

自分で考えてみよう

スコア承認用スクリプトはスコア削除用スクリプトと同様な構造をしています。違いは、その役割がスコアを承認することだけです。スコア承認用スクリプトの空いている部分のコードを埋めて仕上げて下さい。ページにセキュリティを与えて、URLを通して渡されたスコアデータに基づいて、適切なスコアのみを承認して下さい。

```
<?php
....;
?>
...
<?php
require_once('appvars.php');
require_once('connectvars.php');

...
if (isset($_POST['submit'])) {
    if (.....) {
        // Connect to the database データベースへの接続
        $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
        // Approve the score by setting the approved column in the database
        .....  
データベースの approved カラムを設定しスコアを認証
        $query = "UPDATE guitarwars SET .....";
        mysqli_query($dbc, $query);
        mysqli_close($dbc);
        // Confirm success with the user
        echo .....
    }
    else {
        echo .....
    }
}
...
echo '<p><a href=".....">&lt;&lt; 管理画面へ戻る </a></p>';
?>
```

自分で考えてみよう の答え

スコア承認用スクリプトはスコア削除用スクリプトと同様な構造をしています。違いは、その役割がスコアを承認することだけです。スコア承認用スクリプトの空いている部分のコードを埋めて仕上げて下さい。ページにセキュリティを与えて、URLを通して渡されたスコアデータに基づいて、適切なスコアのみを承認して下さい。

```
<?php
require_once('authorize.php');
?>
...
<?php
require_once('appvars.php');
require_once('connectvars.php');
...
if (isset($_POST['submit'])) {
    if ($_POST['confirm'] == 'Yes') {
        // Connect to the database
        $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
        // Approve the score by setting the approved column in the database
        $query = "UPDATE guitarwars SET approved = 1 WHERE id = '$id'";
        mysqli_query($dbc, $query);
        mysqli_close($dbc);
        // Confirm success with the user
        echo '<p>' . $name . 'さんのハイスコア:' . $score . 'は承認されました。';
    }
    else {
        echo '<p class="error">このハイスコアには問題があるため承認できません.</p>';
    }
}
...
echo '<p><a href="admin.php">&lt;&lt; 管理画面へ戻る &gt;&gt; 管理画面へ戻る </a></p>';
?>
```

認証用スクリプトをインクルードして、スコア承認用ページにユーザ名とパスワードによるセキュリティを与える必要があり、そしてこれは何をおいてもスクリプトの先頭に書いておかなければなりません。なぜならヘッダを使っているからです。

② スコア承認用スクリプトを作つて、新しいハイスコアの承認処理をします(approvedカラムに1をセットします)。

認証を反映させるためには、idが一致しなければなりません。

approvedカラムを1にセットすることで、スコアは承認されます。確認画面で提示されたスコアと名前が承認されたこと示します。

スコアが承認されなかったことを明示するのも重要です。これは他のギターウォーズがエラーを表示する場合と同様です。

管理者用ページへ戻るリンクを提供して、ページを遷移しやすくします。

素朴な疑問に答えます

 Q: 何でスクリーンショットのファイル名をスコア承認の時に渡さなくても良いのでしょうか?

 A: なぜならハイスクアの承認処理に必要なのは、スコアの行を見てそれを承認することだけで十分だからです。日付と名前とスコアがあれば特定の行を見つけ出すのには十分で、そのapprovedカラムを1にセットすることができます。

 Q: approvedの0と1というのは何となく暗号的に思えます。この情報をもっと分かりやすく表現する別の方法はないのでしょうか?

 A: あります。MySQLのENUMデータ型で、ENUMとは上げられたもの(enumerated)を意味しますが、これを使えば限られた取りうる値のリストでカラムを作ることができます。ですからapprovedカラムをTINYINTではなく、ENUMで追加することもできたのです。こうすれば0と1の代わりに、yesとnoを使うこともできました。その場合のALTER文は次のような感じになります。

```
ALTER TABLE guitarwars
ADD COLUMN approved ENUM('yes', 'no')
```



自分で考えてみよう

スコア承認用スクリプトで承認するために使われるスコアデータは、「承認」リンクを通して渡されますが、このリンクは管理者用スクリプトで作られます。空いている部分のコードを埋めて、管理者用スクリプトを完成させ、このリンクを作るようにして下さい。

```
// Loop through the array of score data, formatting it as HTML
echo '<table>';
echo '<tr><th>名前</th><th>日付</th><th>スコア</th><th>操作</th></tr>';
while ($row = mysqli_fetch_array($data)) {
    // Display the score data
    echo '<tr class="scorerow"><td><strong>' . $row['name'] . '</strong></td>';
    echo '<td>' . $row['date'] . '</td>';
    echo '<td>' . $row['score'] . '</td>';
    echo '<td><a href="removescore.php?id=' . $row['id'] . '&date=' . $row['date'] .
        '&name=' . $row['name'] . '&score=' . $row['score'] .
        '&Screenshot=' . $row['Screenshot'] . '">削除</a>';
    if (...) {
        echo ...
    }
}
echo '</td></tr>';
echo '</table>';

スコアデータの配列をループし、HTMLにフォーマット
```

ヒント：未承認スコアのみが「承認」リンクを持ちます。

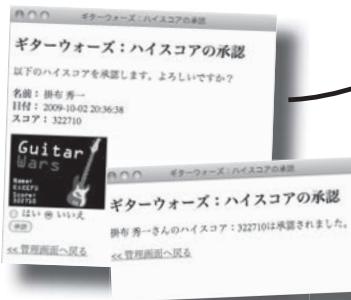
自分で考えてみよう の答え

スコア承認用スクリプトで承認するために使われるスコアデータは、「承認」リンクを通して渡されますが、このリンクは管理者用スクリプトで作られます。空いている部分のコードを埋めて、管理者用スクリプトを完成させ、このリンクを作るようにして下さい。

```
// Loop through the array of score data, formatting it as HTML
echo '<table>';
echo '<tr><th>名前</th><th>日付</th><th>スコア</th><th>操作</th></tr>';
while ($row = mysqli_fetch_array($data)) {
    // Display the score data
    echo '<tr class="scorerow"><td><strong>' . $row['name'] . '</strong></td>';
    echo '<td>' . $row['date'] . '</td>';
    echo '<td>' . $row['score'] . '</td>';
    echo '<td><a href="removescore.php?id=' . $row['id'] . '&date=' . $row['date'] .
        '&name=' . $row['name'] . '&score=' . $row['score'] . '>スコアが未承認かどうかを
        &Screenshot=' . $row['Screenshot'] . '">削除</a>;';
    if ( $row['approved'] == '0' ) { ←
        echo '/<a href="approvescore.php?id=' . $row['id'] . '&date=' . $row['date'] .
            '&name=' . $row['name'] . '&score=' . $row['score'] . '&Screenshot=' .
            $row['Screenshot'] . '>承認</a>;' →
    }
    echo '</td></tr>';
}
echo '</table>';
```

「承認」リンクを作つてid、日時、名前、スコア、それとスクリーンショットのイメージファイル名をURLで渡します。

「承認」リンクは、管理者用ページとスコア承認用ページとを結び付けます。



③ 管理者用ページを修正して、まだ承認されていないスコアに対して「承認」リンクをつけます。

終了

未承認のスコアには価値がない

ギターウォーズのハイスクアアプリケーションには検査機能も加わって、すべてのインフラは整いました。残っている最後のステップは、メインページを承認済みのスコアだけにすることです。これはSQLのSELECT問い合わせ文をちょっといじって、approvedカラムが1(承認済み)にセットされたものだけを引っ張り出せるようにするだけです。こんなことはWHERE節を使えばすぐできます。

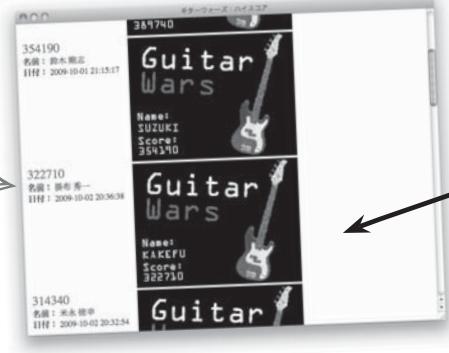
```
SELECT * FROM guitarwars
WHERE approved = 1
ORDER BY score DESC, date ASC
```

このように問い合わせ文にWHERE節を追加することで、承認されていないスコアを除外することができます。スコアには、もちろん新しいものも含まれます。これにより検査官はスコアを見渡して、削除すべきか公開すべきか(承認すべきか)を決定することができます。

WHEREを使えば
特定のカラムの値
に基づいて抽出が
できます。

approvedカラムが
1以外の値にセットされ
ている場合、スコアは
表示されません。

id	date	name	score	screenshot	approved
...					
28	2009-10-01 21:14:56	西原 仁美	308710	nishisscore.gif	1
29	2009-10-01 21:15:17	鈴木 剛志	354190	suzukisscore.gif	1
30	2009-10-02 14:02:54	赤羽 京子	500000	akabanesscore.gif	0
31	2009-10-02 20:32:54	米永 徳幸	314340	yonesscore.gif	1
32	2009-10-02 20:36:38	掛布 秀一	322710	kakefusscore.gif	1



承認されたスコア
だけがメインページ
(index.php)に
表示されるように
なりました。

④ メインページの問い合わせ文を変更して、承認されたスクリプトのみを表示するようにします。

終了



試運転

承認用スクリプトを作ってギターウォーズアプリケーションが使えるように
残りの作業をします。

MySQLツールを使って、ALTER問い合わせ文を発行し、guitarwarsテーブルに新しく approvedカラムを追加します。次に addscore.phpスクリプトのINSERT問い合わせ文を変更して、新しいデータ行のapprovedカラムに0を突っ込みます。

今度は、新しくテキストファイルを作って、approvescore.phpという名前をつけ、スコア承認用スクリプトのコードを打ち込みます。次にadmin.phpスクリプトを修正し、まだ承認されていないスコアに対して「承認」リンクを追加します。最後にindex.phpのSELECT問い合わせ文を変更し、承認されたスコアのみを表示するようにします。

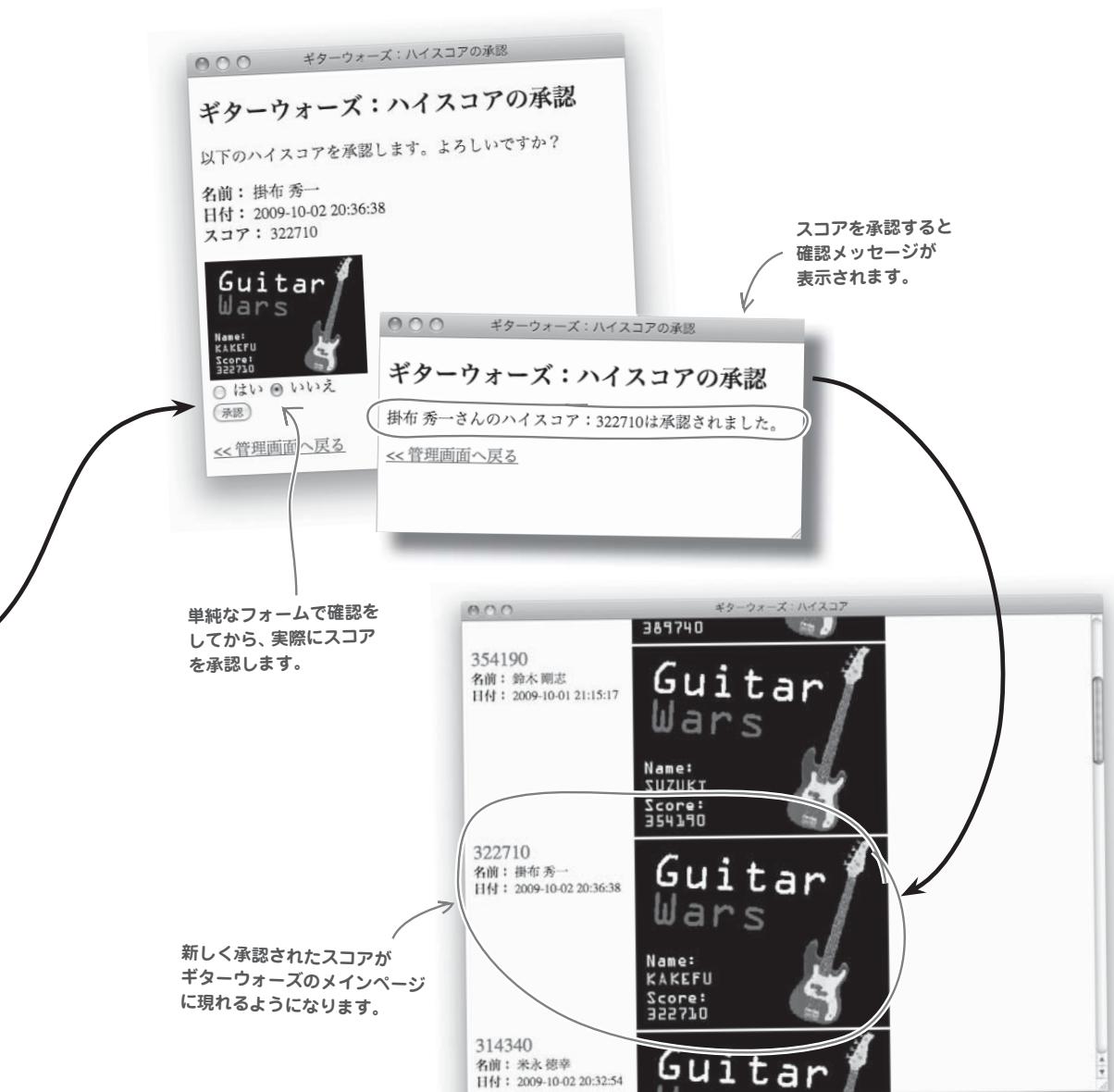
以上すべてのスクリプトをWebサーバにアップロードし、ギターウォーズのメインページをブラウザで開きます。今見えているスコアを覚えておいて、管理者用ページを開いてみます。「承認」リンクのいずれか1つをクリックしたら、スコアの承認作業を行います。この後、メインページに戻ってきて、そのスコアが現れたかどうか確認します。

ギター ウォーズ : ハイスコア 管理

ギター ウォーズ の全ハイスコアリストです。このページを使って不要なスコアを削除します。

名前	日付	スコア	操作
赤羽 京子	2009-10-02 14:02:54	500000	削除 / 承認
今枝 潤太	2009-10-01 20:36:45	389740	削除
鈴木 剛志	2009-10-01 21:15:17	354190	削除
掛布 秀一	2009-10-02 20:36:38	322710	削除 / 承認
米永 徳幸	2009-10-02 20:32:54	314340	削除
西原 仁美	2009-10-01 21:14:56	308710	削除
深沢 麻子	2009-10-01 20:36:07	282470	削除
吉野 良成	2009-10-01 20:38:23	243260	削除
齐藤 真梨子	2009-10-01 20:37:40	186580	削除
岡野 一郎	2009-10-01 20:37:23	127650	削除
上野 正夫	2009-10-01 20:37:02	98430	削除
山口 裕子	2009-10-01 20:38:00	64930	削除

管理者用ページに新しく
作った「承認」リンクにより、
スコア承認用ページに
アクセスすることができます。
このページで個々のスコアを
承認することができます。



100万点ハック

検査対応版のギターウォーズは、セキュリティ面で卓越した改良がなされました。しかし、鉄壁とは程遠いものです。狡猾な侵入者は、またしてもハイスクアシステムの何らかの弱点をついて検査官をすり抜け、自分のスコアを忍び込ませることに成功したようです。赤羽さんを永久に食い止めなければなりません。さもなくばギターウォーズ界に信頼を回復することはできないのです。



すべて検査済み…？

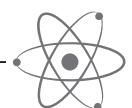
検査官は、赤羽さんのハイスコア「追加」を承認したはずではないと確認しているのですが、にもかかわらず表示させるとapprovedカラムは1にセットされて存在しています。スコア追加用スクリプトは新しいスコアについては確かにapprovedカラムを0にセットしました。そのスクリプトのINSERT問い合わせ文は今修正したばかりです。何か想定外のことがあるはずです！

ギターウォーズの検査官には何が起こっているのか分かりません。

どうしてこんなことが可能なんだろう？こんなスコアを承認したはずは絶対にない。
100万点だって！？



id	date	name	score	screenshot	approved
21	2009-10-01 20:36:07	深沢 麻子	282470	asakosscore.gif	1
22	2009-10-01 20:36:45	今枝 凜太	389740	lintasscore.gif	1
23	2009-10-01 20:37:02	上野 正夫	98430	uenoosscore.gif	1
24	2009-10-01 20:37:23	岡野 一郎	127650	okanosscore.gif	1
25	2009-10-01 20:37:40	斎藤 真梨子	186580	saitosscore.gif	1
26	2009-10-01 20:38:00	山口 裕子	64930	yukossscore.gif	1
27	2009-10-01 20:38:23	吉野 吉成	243260	yoshinosscore.gif	1
28	2009-10-01 21:14:56	西原 仁美	308710	nishisscore.gif	1
29	2009-10-01 21:15:17	鈴木 剛志	354190	suzukisscore.gif	1
31	2009-10-02 20:32:54	米永 徳幸	314340	yonesscore.gif	1
32	2009-10-02 20:36:38	掛布 秀一	322710	kakefusscore.gif	1
33	2009-10-05 14:58:59	赤羽 京子	1000000	akabanesscore2.gif	1



脳力発揮

赤羽さんはどうやってインチキのポストをして検査官をすり抜けたと思いますか？

このスコアは検査官によって承認されたものではないにもかかわらずapprovedカラムは1にセットされているため、表示されてしまいます。



頭の体操

すでに分かっていることは、赤羽さんの100万点ハック[†]はスコア承認用フォームでは何もしないということです。彼女のイタズラは、完全にスコア追加用フォームだけで行われています。赤羽さんはスコア追加用フォームに以下のようなフォームデータを打ち込むことでハックを成功させたのです。全く同じようにフォームデータを打ち込んで、スコアを追加してみて下さい。何が起こっているか分かりますか？

[†] 訳注：赤羽 京子さんは、実在の人物ではありません。実在の人物をモデルにしていますが、実際には老スーパー・ハッカーではなく、若くて美しいスーパー・プログラマです。

--の後のこの部分に空白を入れることを忘れないで下さい。

赤羽 京子

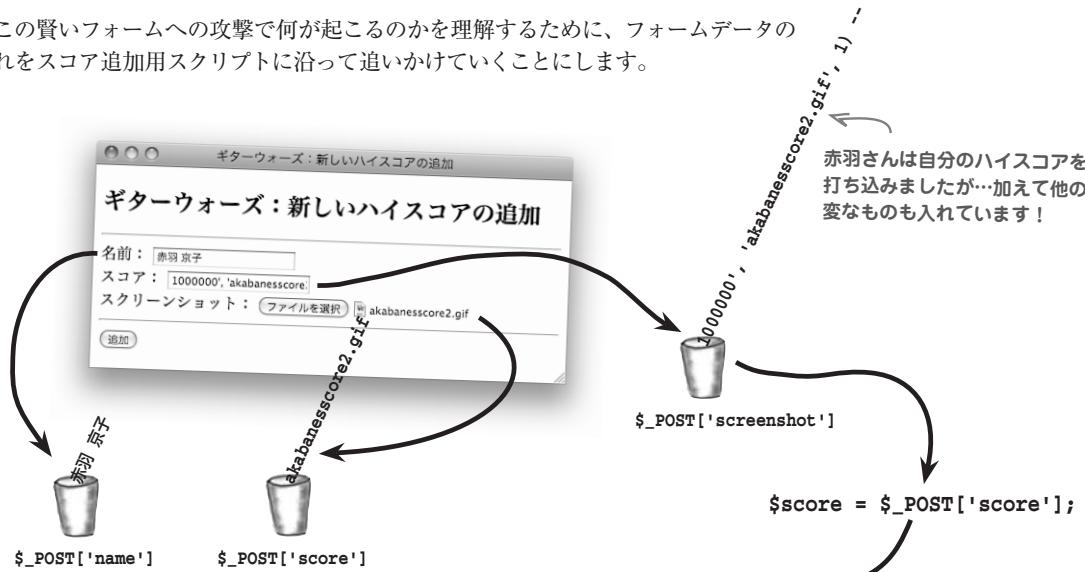
1000000', 'akabanesscore2.gif', 1) --

これは GIF でも JPEG でも 32KB 以下であればどんなイメージファイルでも構いません。

akabanesscore2.gif

彼女は正確には一体何をしたのか？

この賢いフォームへの攻撃で何が起こるのかを理解するために、フォームデータの流れをスコア追加用スクリプトに沿って追いかけていくことにします。



スコアのフォームフィールドには、通常とは異なる
コンテンツが突っ込まれ、それがそのまま \$score 変数
に入り、INSERT問い合わせ文に直接使われることに
なります。

```
INSERT INTO guitarwars
VALUES (0, NOW(), '$name', '$score', '$screenshot', 0)
```

スコアのフォームフィールドは、例えば1000000のような1つの数値が来るものとしています。しかし代わりに単一引用符で括られたいくつかの値が、コンマ区切りで書いてあります。そしてその次に奇妙な2つのマイナス記号で終わっています。超奇妙。

この奇妙なデータはまず \$score 変数にぶち込まれ、その後 INSERT問い合わせ文に組み込まれます。これは単に無意味なスコアになってしまいますよね？そうではなくて、何か悪意のあるものに取って代わるのでしょうか？

自分で考えてみよう

反対側のページに書いてあるフォームデータをそのまま使って、完全なスコア追加用SQL問い合わせ文を書いて、100万点アタックをやってみて下さい。問い合わせ文中の変数は実際のフォームデータに置き換えることを忘れないで。何が起こると思うか、メモしておきましょう。

自分で考えてみよう の答え

反対側のページに書いてあるフォームデータをそのまま使って、完全なスコア追加用SQL問い合わせ文を書いて、100万点アタックをやってみて下さい。問い合わせ文中の変数は実際のフォームデータに置き換えることを忘れずに。何が起こると思うか、メモしておきましょう。

```
INSERT INTO guitarwars
```

```
VALUES (0, NOW(), '赤羽 京子', '1000000', 'akabanesscore2.gif', 1) -- ', 'akabanesscore2.gif', 0)
```

赤羽さんは何故か自分バージョンの問い合わせ文を作り出し、元々の問い合わせ文を上書きしてしまいました。

これが摩訶不思議な問い合わせ文か。
スクリーンショットのファイル名は2回
出てくるね。だけどこの2つのマイナス記号
が何をするのかが分からぬいや…これで
問い合わせが動くの？

approvedカラムはデータベース
の構造上最後にあるため、値は
強制的に1になっています…
つまり承認済みです！



コメントでMySQLをごまかす

赤羽さんの100万点アタックに加担した本当の罪人は、奇妙なことにSQLのコメントだったのです。2つのマイナス記号(--)を使うと、SQLコードの残りの行はコメントアウトされます。コメントを有効にするには2つのマイナス記号に続けて空白文字を加える(--)必要があります。すると空白以降のすべてが無視されます。では赤羽さんの問い合わせ文全体をよく見てみましょう。小さな奇知の塊です。

```
INSERT INTO guitarwars
VALUES (0, NOW(), '赤羽 京子', '1000000', 'akabanesscore2.gif', 1) -- 'akabanesscore2.gif', 0)
```

これで分かったでしょうか？コメントが機能して残りのSQLコードを消し去り、エラーを吐かないようにしてあります。これで赤羽さん版の問い合わせ文は、どこにも引っ掛かることなく、すり抜けることができたのです。結果として承認済みの新しいハイスコアが、検査官一度も捕らえられることのないまま、簡単にできあがります。

--コメントでSQLコードの
残りの行は無視されます。

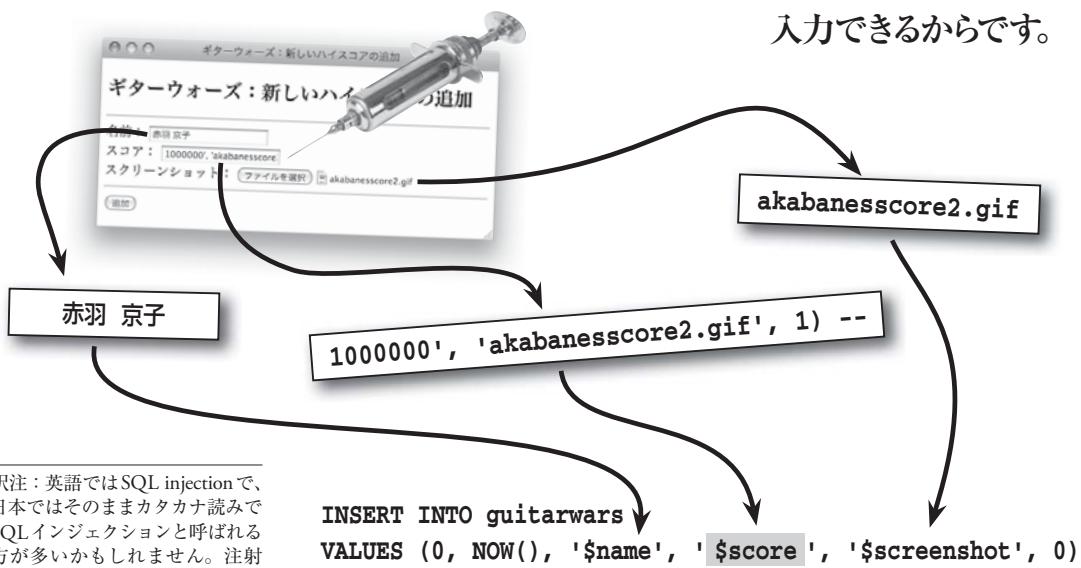
赤羽さんは問い合わせ文を
ごまかして、自分のハイスコア
を承認してしまいました。

id	date	name	score	screenshot	approved
33	2009-10-05 14:58:59	赤羽 京子	1000000	akabanesscore2.gif	1

スコア追加用フォームはSQLを注入されたのです

赤羽さんの攻撃はSQL注入[†]と呼ばれていて、極めて卑劣なトリックです。フォームのデータを使って問い合わせ文の基本的な操作を変えてしまいます。つまりフォームフィールドを使って、名前やスコアといった単なる情報を供給する代わりに、その基となっているSQL問い合わせ文自体に小細工をしてしまうのです。ギターウォーズの場合で言うと、赤羽さんはSQL注入をスコアフィールドに使って、スコアだけでなく、スクリーンショットのファイル名と承認の値も供給しています。そして最後にコメントを付けて元々のSQLコードがエラーを吐かないようにしているのです。

フォームフィールドは、Webアプリケーションにおいてはセキュリティ上の弱点です。なぜならそこにはユーザがデータを入力できるからです。



[†] 訳注：英語ではSQL injectionで、日本ではそのままカタカナ読みでSQLインジェクションと呼ばれる方が多いかもしれません。注射器の絵が挿入されていたため、あえてこちらの訳語を用いました。

素朴な疑問に答えます

Q: SQLには--以外にもコメントの記法がありますか？

A: あります。1行のコメントとしては--の代わりに#を使うものがあります。こちらもSQLコードの行の終わりまでをコメントアウトします。SQLでは複数行にまたがるコメントもあります。こっちはPHPの複数行にまたがるコメントと同様で/*と*/で囲みます。

Q: 赤羽さんのSQL注入攻撃はapprovedカラムがテーブルの最後になかったとしても効果があったのでしょうか？

A: ありません。これは非常に重要なことです。特にこのINSERT問い合わせ文に関しては、テーブルカラムのデフォルト順序に大きく依存しています。問い合わせ文の最後に1を突っ込んでうまくいってしまったのは、approvedが最後のカラムで、screenshotカラムの直後にあったからです。

SQL注入からデータを守る

SQL注入によって露呈した本当の弱点は、フォームフィールドに危険な文字が含まれているかどうかを検証しなかったことによるものです。「危険な文字」とはSQL問い合わせ文の性質を変えてしまう可能性のある任意の文字のことで、例えばコンマや引用符や--といったコメント(開始)文字などのことです。データの後ろに空白文字がついているだけでも、有害になることがあります。データの前後にある空白に関しては、PHPの組み込み関数trim()を使えば簡単に取り除くことができます。単にすべてのフォームデータについてtrim()関数を通してから、SQL問い合わせ文を使えばよいのです。

```
$name = trim($_POST['name']);
$score = trim($_POST['score']);
$screenshot = trim($_FILES['screenshot']['name']);
```

でもデータ前後の空白だけが問題なのではありません。まだコンマや引用符やコメント(開始)文字や、その他もろもろたくさんあります。ですからフォームフィールドをtrimする(刈り取る)ことに加えて、他にも問題となりそうな文字を見つけて無害化してしまう必要があります。PHPは、さらに別の組み込み関数mysqli_real_escape_string()で救援に駆けつけてくれています。この関数は潜在的に危険な文字をエスケープして、問い合わせ文をどのように実行するかについて、敵対的な影響を与えないようにしてくれます。これらの文字はフォームフィールドのデータとして存在したままです。単に問い合わせ文とのインターフェースができなくなるだけです。

trim()関数とmysqli_real_escape_string()関数とを組み合わせることで、SQL注入に対抗するための強固な防衛線を張ることができます。

```
$name = mysqli_real_escape_string($dbc, trim($_POST['name']));
$score = mysqli_real_escape_string($dbc, trim($_POST['score']));
$screenshot = mysqli_real_escape_string($dbc, trim($_FILES['screenshot']['name']));
```

ギターウォーズのフォームフィールドをtrim()関数と mysqli_real_escape_string()関数とで処理することで、SQL注入攻撃の可能性は劇的に軽減しました。ただしこの2つの関数だけでは十分ではありません。恐らく問い合わせ文自体の弱点を軽減する方法があるはずです…

trim()関数は、このフォームデータの前後についている空白文字を捨ててしまいます。

SQL注入はフォームデータを適切に処理すれば防ぐことができます。

mysqli_real_escape_string()関数は、危険な文字をエスケープ形式に変換してSQL問い合わせ文に敵対的な影響を与えないようにします。

mysqli_real_escape_string()は、データベース用の関数と考えることができますので、データベース接続変数を渡す必要があります。これは問い合わせ文を発行するときと同じものです。

(パラメタ付きの)安全なINSERT

フォームフィールドの防御が甘かったという弱点に付け込まれたことに加え、赤羽さんのSQL注入はapprovedカラムがデータベースの構造上 screenshotカラムの次にあったという事実にも依存しています。このようにして彼女はINSERT文の後ろに1を単に加えるだけで、approvedカラムにその値を突っ込むことができたのです。問題なのはINSERT問い合わせ文が、すべてのカラムにデータをぶち込まなければならぬような作りになっていたことです。このため不必要なリスクが増してしまったのです。

```
INSERT INTO guitarwars
VALUES (0, NOW(), '$name', '$score', '$screenshot', 0)
```

データをこのような形式でテーブルにぶち込む場合、データの順序はテーブル構造のカラムの順序と一致していなければなりません。つまり5番目のデータは、screenshotカラムに入ることになります。テーブルの5番目のカラムが screenshotだからです。しかし、idカラムとapprovedカラムには、明示的にデータを突っ込む必要はないのです。なぜならidは増分1で値が自動的に入りますし、approvedカラムは常に0でよいからです。もっとスマートなやり方は、新しいハイスコアとして必要なデータだけを明示的にぶち込むことに専念する、ということです。idカラムとapprovedカラムは、**デフォルト**でそれぞれAUTO_INCREMENTと0にしておけば良いわけです。

そこで限定的なINSERT問い合わせ文を使います。これはカラムのリストの次にデータのリストを置いて、1対1にマッチさせます。これによりapprovedカラムが1にセットされるというリスクはなくなります。このカラムはもはや問い合わせ文には存在しないのです。この種の問い合わせ文はすでにじみのあるものと思いますが、すでに他の例で何度か使ってきました。

```
INSERT INTO guitarwars (date, name, score, screenshot)
VALUES (NOW(), '$name', '$score', '$screenshot')
```

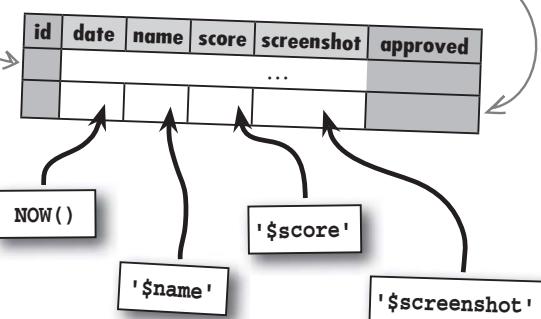
idカラムは取り扱います。値は
どっちみち自動増分で入ります。

今度の版のINSERT問い合わせ文は、どのカラムにどのデータをぶち込むかをちゃんと書いてあります。テーブル構造への依存性について心配する必要は全くなく、データをぶち込むことができるようになります。実際、このようにINSERT問い合わせ文を使う方が良いコーディングスタイルです。データを自分が意図した通りの場所にきっちり入れができるからで、テーブルの構造的なレイアウトに依存したやり方とは大違いです。

INSERT問い合わせ文は、どのカラムにどの値をぶち込むかをピンポイントで指定できるように書くことができます。

理想的には、idカラムとapprovedカラムには値を設定すべきではありません。なぜならこの2つのカラムにはデフォルトの値があるからです。

approvedカラムには何も入ませんが、問い合わせ文の一部としてリストされていないからです。





ちょっと待って。MySQL テーブルのデフォルト値なんて聞いたことないわ。
そんなことって本当にできるの？

できるなんてもんじゃなく、可能であればいつでも DEFAULT カラム値を指定した方が良いくらいです。

SQL の DEFAULT コマンドを使うと、カラムのデフォルト値を指定することができます。カラムがデフォルト値を持っている場合、INSERT問い合わせ文で値をセットしなくても安心で、値は自動的にデフォルト値になっているということが保証されています。これは guitarwars テーブルの approved カラムにとって完璧です。それではもう一度テーブルを修正し、approved カラムにデフォルト値 0 (未承認) をセットすることにします。

approved カラムはすでにありますので、この ALTER TABLE 文は ADD COLUMN ではなく MODIFY COLUMN を使う必要があります。

**ALTER TABLE guitarwars
MODIFY COLUMN approved TINYINT
DEFAULT 0**

DEFAULT により approved カラムは自動的に値 0 が代入されます。もちろん INSERT 問い合わせ文で明示的に別の値が指定された場合は除きます。

カラムの型は相変わらず指定しなければなりません。最初にカラムを追加した時と同じ覚えておけば良いでしょう。

これで approved カラムはデフォルト値を取るように変更されました。つまり新しく改良されたスコア追加用スクリプトの INSERT 問い合わせ文は、approved カラムについては指定しなくとも、ハイスコアを追加できるようになりました。こういう設計は極めて好ましいものです。第 1 に、デフォルトで済む値を明示的に突っ込むべきではないからです。第 2 にセキュリティレベルがほんの少しだけ上がり、approved カラムに対して潜在的な攻撃にさらされることがなくなったからです。

フォームの妥当性検証にはやり過ぎということはありません

SQL注入による攻撃に対するリスクを軽減する最後のステップとして、スコア追加用スクリプトの妥当性検証をやっておきます。スクリーンショットのファイルタイプやサイズがアプリケーションで定義した制限を越えていないかチェックする前に、スコア追加用の3つのフォームフィールドが空白でないかどうかをチェックします。

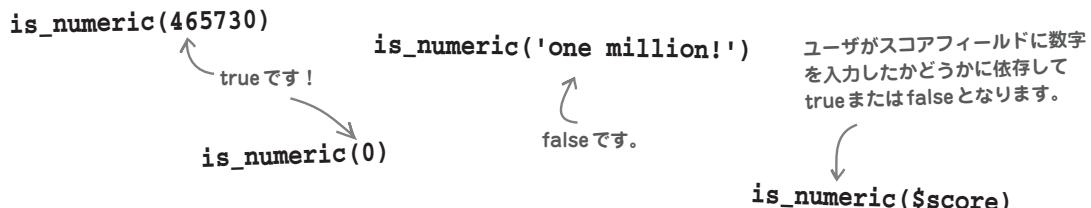
```
if (!empty($name) && !empty ($score) && !empty($screenshot)) {
```

…
}

このif文でチェックして、すべてのフォームフィールドが空白でないことを確認します。

このコードは、見たところ何も悪いところはありません。しかしアプリケーションにセキュリティを与えるには、「任務による招集」[†]よりもっと上まで見据えていなければならぬものです。スコアフィールドには数値が入るべきですから、空白でないことをチェックするだけでなく、本当に数値が入っているかを確認するという意味のあることです。PHPのis_numeric()関数は、正にこれをチェックし値が数字であればtrueを返し、そうでなければfalseを返します。些細なことではありますが、数字を期待しているところで数字かどうかをチェックするといったことは、最終的にはアプリケーションにセキュリティを与えデータ攻撃に対して強いものとなります。

[†] 訳注：原著はcall of dutyという、第二次世界大戦を舞台にしたシューティングゲームのタイトルになっています。日本では「赤紙」と呼ばれていたものと意味としては同じと思われます。



可能な場合はいつでも、フォームデータが要請通り

のフォーマットになっているかを確認します。



スコア追加用フォームの妥当性検証を書き直して、if文でis_numeric()関数を使いスコアとして数値のみが許されるようにして下さい。



エクササイズ の答え

スコア追加用フォームの妥当性検証を書き直して、if文で`is_numeric()`関数を使いスコアとして数値のみが許されるようにして下さい。

```
if (!empty($name) && is_numeric($score) && !empty($screenshot)) {  
    ...  
}
```



試運転

スコア追加用スクリプトのフォームデータ処理を強化する

`addscore.php`のフォームデータを変数に代入する部分を微調整し、`trim()`関数と`mysqli_real_escape_string()`関数を使ってフォームデータをきれいにします。次に`INSERT`問い合わせ文を変更し、カラム名と値の両方を指定する形式にします。これにより`id`カラムと`approved`カラムの値を指定する必要がなくなります。更に`if`文を変更しフォームフィールドの妥当性を検証し、スコアが数値であることをチェックします。

最後にMySQLツールを使って、ALTER問い合わせ文を走らせ、approvedカラムのデフォルトを0に設定します。

新しい版のスコア追加用スクリプトをWebサーバにアップロードし、Webブラウザで開いたら、さっくと同様な注入攻撃をもう一度やってみて下さい。

スコアのフォーム
フィールドは、今度は
数字以外は何も受け
付けてくれません。

ギターウォーズ：新しいハイスクアの追加

ギターウォーズ：新しいハイスクアの追加

エラー：入力項目をすべて埋めてからハイスクアを追加してください。

名前：

スコア：

スクリーンショット： ファイルが選べていません。

確かに、このエラーメッセージは
もうちょっと分かり易くできるはず
です。ただこれについては、スクリプト
の構造を変える必要もなく、それほど
難しいことではありません。

フォームの妥当性検証は、データベースのセキュリティよりも遙か遠くにあるテーマです。これについては第10章でもっと詳しくやります…

停戰一

ギターウォーズのハイスコアに干渉しようという赤羽さんのもくろみは、ついに打ち砕かれたようです。アプリケーションを改良してSQL注入の影響を受けないようにしたおかげです。



PHP&MySQL道具箱

ギターウォーズのハイスコアアプリケーションを新しいレベルに持ち上げたことに加えて、いくつかの新しいツールや技法を手に入れました。中でも最も大事なものについて復習しておきます。

header()

PHPの組み込み関数で、ヘッダをサーバからブラウザに送りつける際に使います。これによりページのリダイレクト、特定のcontent typeの指定、HTTP認証の実現などができます。

is_numeric()

PHPの組み込み関数で、値が数字かどうかをチェックします。数字用のフォームフィールドに実際に数値が入っているかどうかをチェックするのに便利です。

\$_SERVER

PHPの組み込みスーパーグローバルで、他にも入っているものがありますが、ユーザが打ち込んだユーザ名とパスワードが入ります。HTTP認証を要求するページにユーザがアクセスしようとしたときに必要となります。これらの値を想定している値と比較してチェックすることで、セキュリティを必要とするページを保護することができます。

exit()

PHPの組み込み関数で、PHPスクリプトは即終了させます。スクリプト実行中にexit()関数に出くわすと、その先のPHPコードは実行されず、その先のHTMLコードがブラウザに配信されることもありません。

DEFAULT value

このSQL文でテーブル内のカラムにデフォルトの値を設定することができます。新しくカラムが追加され、カラムに値が設定されていない場合、デフォルトの値が設定されます。

trim(), mysqli_real_escape_string()

この2つのPHP組み込み関数は、フォームデータを手軽に処理できて、SQL問い合わせ文と干渉するような問題となる文字を無害化してくれます。trim()の方はデータの前後にある空白を取り除きます。mysqli_real_escape_string()の方は、特殊な文字をエスケープしてくれます。

人間による検査

すべてに検査を！この場合で言うと人間というのは、望ましくないコンテンツが他人から投稿されたことを認識して除去することのできる防衛線として最適であるという意味です。自動化されたセキュリティの技法はもちろん重要ですが、やっぱり生きていて、息をしていて、脳みそのある人にはかないません！

HTTP認証

単純なWeb上のセキュリティ技法で、Webページやスクリプトにアクセスする際にユーザー名とパスワードによる制限をかけます。機密性の非常に高いセキュリティが必要なアプリケーションには向いていませんが、HTTP認証は、手軽に素早くある程度のセキュリティをWebアプリケーションに付与することができます。

カラムと値を指定する問い合わせ文

INSERT問い合わせ文の指定方法の1つで、カラム名と対応する値とを注意深くマッチさせながら指定するものを言います。これに対して、テーブルカラムの構造上の順序に依存してデータをマッチさせて指定するものもあります。

SQL注入 (SQLインジェクション)

セキュリティホールをついた攻撃の1つで、不逞の輩が何らかの方法でSQL問い合わせ文に危害を加えデータベースに想定外のアクセスをすることを言います。大抵のSQL注入はWebフォームに細工をしてやばいデータを直接書き込むことで、動的に問い合わせ文を作ってしまいます。このためフォームの妥当性検証で大体は解決できます。

フォームの妥当性検証

ユーザーによってフォームに入力されたデータをすべてチェックする処理で、データが想定しているフォーマットかどうか確認します。妥当性検証は、フォームを簡単に使えるようにするだけでなく、ユーザーがフォームに変なデータを入力することを防ぐことでWebアプリケーションのセキュリティを高めるのに役立ちます。

7章 個人向けのWebアプリを作る

覚えてますか？



誰でも忘れられるのはイヤです。もちろんWebアプリケーションのユーザだってそうです。

アプリケーションに「メンバ」という概念があるのなら、つまりユーザがアプリケーションと何らかの個人的な相互作用をするのなら、アプリケーションはユーザを覚えておく必要があります。家のドアを開ける度に家族の誰かに対して自己紹介をしなければならないなどというのはウンザリなはずです。でもその必要がないのは、メモリ（記憶）というすばらしいものがあるからです。ところがWebアプリケーションは人々を自動的に覚えてはくれません。そのような機能は、有能なWeb開発者に委ねられています。その人がその人の裁量で使える（もちろんPHPとMySQLの）ツールを使ってユーザを自動的に覚えてくれる個人向けのWebアプリを作れるかどうかにかかっているのです。

正反対が魅力的とよく言われます

昔から言われているコトです。男が女と出会い、女は男をバカだと思い、男は女をやっかいだと思う、しかしその違いが魅力に思え、二人は生涯を共に生きる。このストーリーに基づいて革新的な新しいデートサイト、Mismatch.netが立ち上りました。ミスマッチサイトは「正反対の魅力」という心理に関する理論を採用し、違いに基づいてミスマッチを探します。

問題なのは、ミスマッチサイトがまだ離陸前で、システム構築を完了させるにはWeb開発者が緊急に必要なことです。つまり私たちの出番です。何百万もの寂しいハートの持ち主がアプリケーションの完成を熱烈に待っています…彼らを失望させてはいけません！

加理奈さんはドキュメンタリー
TVとヨガと寿司が好きで、
ミスマッチが成功しないか
と期待しています。

超ミスマッチが
見つからないか
待ちきれないわ。

人見 勝則
男性
1981-11-03
栃木, 大田原

個人向けのWebアプリケーションは、
個人の情報で成長していきます。つまり
ユーザが個人レベルでアプリケーション
にアクセスできる必要があります。

見てくれ、
これがオレの
武器さ！



佐藤 加理奈
女性
1984-07-19
東京, 三鷹

人見さんはプロレスとウェイト
リフティングとSPAMが好きで、
誰か返事をくれないかと
興奮気味です。



ミスマッチサイトのユーザはサイトと個人レベルで相互作用できる必要があります。このため、ユーザは個人のプロフィールが必要で、自身の情報を入力し、その情報、例えば性別、誕生日、住所などを他のミスマッチサイトユーザと共有するのです。

ミスマッチサイトは個人の情報がすべてです

つまりミスマッチサイトは個人の情報を通して関係を確立することがすべてです。この関係は、**コミュニティの各ユーザー**によって維持されなければならず、各ユーザーはサイトとの相互作用によって自分自身の個人データを管理します。`mismatch_user`という名のデータベースを使って、ミスマッチサイトのユーザーを管理し、個人の情報をぶっ込みます。

これがミスマッチサイトのデータベースです。

The diagram shows a table titled "mismatch_user" with columns: user_id, join_date, last_name, first_name, gender, birthdate, state, city, and picture. Two rows of data are shown:

user_id	join_date	last_name	first_name	gender	birthdate	state	city	picture
1	2008-04-17 09:43:11	佐藤	加理奈	F	1984-07-19	東京	三鷹	karenapic.jpg
11	2008-05-23 12:24:06	人見	勝則	M	1981-11-03	栃木	大田原	hitomipic.jpg

A callout points to the table with the text: "ミスマッチサイトのデータベースには、mismatch_user テーブルがあってユーザーとその個人的なプロフィールデータをぶち込みます。"

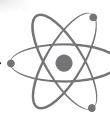
Another callout points to the rows with the text: "mismatch_userテーブルの各行は、個々のユーザーの個人データを含んでいます。"

The screenshot shows a profile editing form for a user named "skarena". The form includes fields for name, gender, birthdate, address, and a photo upload section. A preview image of the user's profile picture is shown on the right.

プロフィール編集ページおよびプロフィール参照ページは、誰のプロフィールにアクセスするべきかを知っている必要があります。

佐藤 加理奈
女性
1984-07-19
東京, 三鷹

ユーザプロフィールの参照に加え、ミスマッチサイトのユーザーは自分自身の個人プロフィールをプロフィール編集用ページを使って編集できるようにします。しかし問題は、アプリケーションがどのユーザーのプロフィールを編集できるようにすればいいのか、知っておかなければならないということです。プロフィール編集用ページは、何らかの手段によりユーザーを認識し、誰がページをアクセスしているのかを知っている必要があります。

 **脳力発揮**

ミスマッチサイトのプロフィール編集用ページを各々別のユーザー用にカスタマイズするにはどうすればよいと思いますか？

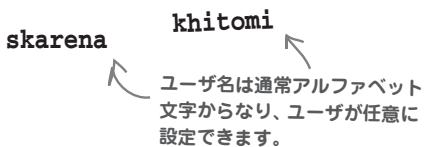
ミスマッチサイトにはログインが必要です

ミスマッチサイトで個人データにアクセスする問題を解決するには、ユーザがログインするという概念が必要となります。これはユーザがアプリケーションにログインにより入ってこれるようにするという意味です。ミスマッチサイトは異なるユーザ各々に対してそれぞれ特注の情報にアクセスできるという機能を提供できるようになります。例えば、ログインしたユーザは、自分のプロフィールデータだけを編集することができます。ただし別のユーザのプロフィールを参照することはできません。ユーザのログイン機能は、ミスマッチサイトアプリケーションを個人向けにするカギとなります。

ユーザログインでは、通常2種類の情報が必要です。ユーザ名とパスワードです。

ユーザ名

ユーザ名の役割は、各ユーザに一意の名前を提供することで、システム内でユーザを識別できるようになります。ユーザはユーザ名を通してアクセスしたり、お互いにコミュニケーションしたりできるようになります。



ユーザ名とパスワードとでユーザは、ミスマッチアプリケーションにログインし、個人のデータにアクセスし、プロフィールの編集をしたりします。

ユーザのユーザ名とパスワードは、アプリケーションにとってそれが誰なのかを知るために必要なすべてです。

ユーザがログインすると、アプリケーションはユーザを思い出すことができるので、個人に特化された履歴を表示することができます。

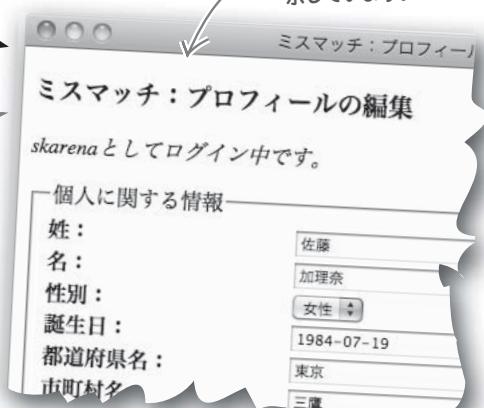
ユーザのログインによりWebアプリケーションはユーザにとって個人的なものとなります。

パスワード

パスワードはセキュリティのレベルを上げるためのもので、ユーザがログインする場合に個人データの保護手段となります。ログインするにはユーザはユーザ名とパスワードの両方を打ち込む必要があります。

パスワードは極めて重要なデータです。アプリケーション内のどこでも、たとえデータベース内であっても見えるようなことがあってはいけません。

プロフィール編集用ページは、特定のユーザでログインしたことを示しています。

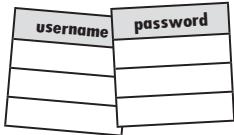


ユーザログイン作戦と連合する

ミスマッチサイトにユーザのログインをサポートする機能を付加するのは、ちょっとやそっとではできません。したがって重要なのは、まず何をしなければならないかを明確にしてから、コードを書き、データベース問い合わせ文を走らせるべきだということです。既存のテーブルにはユーザ情報が格納されていることは分かっていますから、まず始めにテーブルを変更して、ログインデータを突っ込みます。また、ユーザがログインデータを打ち込む方法が必要です。これは何らかの形でミスマッチアプリケーションの残りの部分と統合されていて、プロフィール編集用ページなどはログインに成功した場合に限りアクセス可能とする必要があります。これから作り上げるログインの開発ステップを以下に示します。

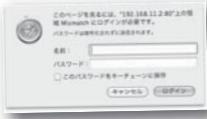
① ALTERを使ってテーブルにusernameカラムとpasswordカラムを追加する。

データベースには新しいカラムが必要で、各ユーザのログインデータを突っ込みます。このデータはユーザ名とパスワードです。



② 新しくログイン用スクリプトを作り、ユーザ名とパスワードを入力するよう促します。

ログイン用ページは、極限まで保護された個人用ページで、正しいユーザ名とパスワードの入力を要求します。この情報が正しく入力されてからでないと、ミスマッチサイトは特定のユーザデータを表示することはできません。つまりスクリプトは個人用データへのアクセスを制限し、正しくログインしないとデータを見ることができないようにしなければなりません。



③ ログイン用スクリプトをミスマッチアプリケーションの残りの部分にくっつける。

ミスマッチアプリケーションのプロフィール編集用ページおよびプロフィール参照用ページは、ログイン後のユーザにだけアクセス可能とすべきです。ですからログインスクリプトを通してユーザがログインしたかどうかを確認した後でのみ、これらのページへのアクセスを許可する必要があります。





一歩でも前へ進む前に、ちょっとの時間ミスマッチアプリケーションを舐めまわして、どのように動くべきか考えて見ます。

ミスマッチアプリケーションのコードをすべてWebサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードします。.sql以外のすべてのコードをWebサーバに置きます。.sqlファイルにはSQL文が書いてあり、これでミスマッチサイトに必要なテーブルを作ることができます。各.sqlに書いてある文をSQLツールで走らせれば、初期状態のミスマッチ用テーブルができますので、これで作業を始めます。

すべての作業が終わったら、Webブラウザでindex.phpページを開き、アプリケーションをチェックしてみて下さい。プロフィール参照用ページとプロフィール編集用ページは、まだ全然できていないことに注意して下さい。これらのページはユーザのログインに完全に依存しており、今まさに作ろうとしているところなのです。

ミスマッチ：正反対が魅力的！

ミスマッチサイトのメインページは、最新ユーザーの名前と写真を表示していますが、それ以外のことはログインしないと表示されません。

ミスマッチ：正反対が魅力的！

新規メンバー：

- ♥ プロフィールの参照
- ♥ プロフィールの編集

これら2つのリンクは、アプリケーション内の個人用部分につながっています。

ダウンドロード

ミスマッチアプリケーションの全ソースコードは、以下のWebサイトからダウンロードできます。

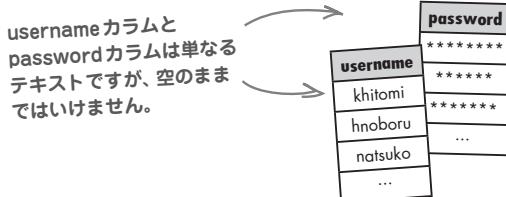
<http://www.oreilly.co.jp/books/9784873114446/>

ログイン用のデータベースを準備する

それでは、作り始めましょう。`mismatch_user` テーブルにはすでに各ユーザーのプロフィール情報をぶち込むのに十分な場所があります。ユーザーが来た際のログイン情報が欠乏しているだけです。もっとちゃんとと言うと、テーブルには各ユーザーのユーザ名とパスワードを突っ込むためのカラムが欠けています。

mismatch_user									
user_id	join_date	last_name	first_name	gender	birthdate	state	city	picture	

ユーザ名とパスワードは両方とも純粋なテキストデータですから、おなじみのMySQLデータ型VARCHARを使って、新しく`username`カラムと`password`カラムとを作ればよいでしょう。ただし、他のユーザプロフィールデータと違って、ユーザ名とパスワードは絶対に空(NULL)のままであることが許されません。



`mismatch_user` テーブルには
ユーザー名とパスワード用のカラム
が必要で、ここにユーザーのログイン
データを突っ込んでおきます。

素朴な疑問に答えます

Q: ユーザを一意に識別するためならusernameではなく、単にuser_idを使うというのはダメなのでしょうか？

A: どうしてもそうしたければですが、できます。実際user_idの目的はユーザーの行を一意に識別する効率的な方法を提供することです。ただし、数字のIDは覚えるのが大変なので、ユーザは自分で決めたユーザ名を使って、個人向けのWebアプリケーションにアクセスできるほうが嬉しいものです。つまりユーザにとってどちらが使いやすいかどうかという問題です。Johanという人が「Johan」と「11」のどちらでログインする方がいいのかということです。誰でも数字で呼ばれるような扱いは嫌なものですね！



自分で考えてみよう
16文字よりも長いパスワードを入れたい人も覚えられる人もほとんどないでしょう！

SQL文を書いて、`username`カラムと`password`カラムを以下に示したテーブル位置に追加して下さい。`username`は32文字まで、`password`は16文字までそれぞれ保存することができるようになります。どちらもNULLデータは許さないようにします。

mismatch_user										
user_id	username	password	join_date	last_name	first_name	gender	birthdate	state	city	picture

自分で考えてみよう の答え

ALTER TABLEを使って既存のテーブルに新しいカラムを追加します。

SQL文を書いて、usernameカラムとpasswordカラムを以下に示したテーブル位置に追加して下さい。usernameは32文字まで、passwordは16文字までそれぞれ保存することができるようになります。どちらもNULLデータは許さないようにします。

`ALTER TABLE mismatch_user ADD username VARCHAR(32) NOT NULL AFTER user_id,
ADD password VARCHAR(16) NOT NULL AFTER username`

AFTER節で、テーブル内で新しいカラムをどこに追加するのかを制御します。

usernameカラムを先に追加してあるので、ここで参照しても大丈夫です。

mismatch_user										
user_id	username	password	join_date	last_name	first_name	gender	birthdate	state	city	picture

テーブル内のカラム位置は実際には大して問題にはなりません。単に最も重要なカラムから順に並べるという意味での構造的な意味を持たせているだけです。

- ① ALTERを使ってテーブルにusernameカラムとpasswordカラムを追加する。

どう考えても、パスワードをデータベースにそのまままぶつ込むなんてあり得ないわ…
まぶつ込む前に暗号化とか普通するでしょ？



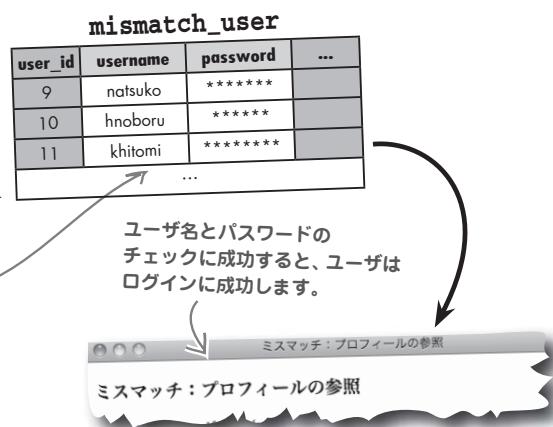
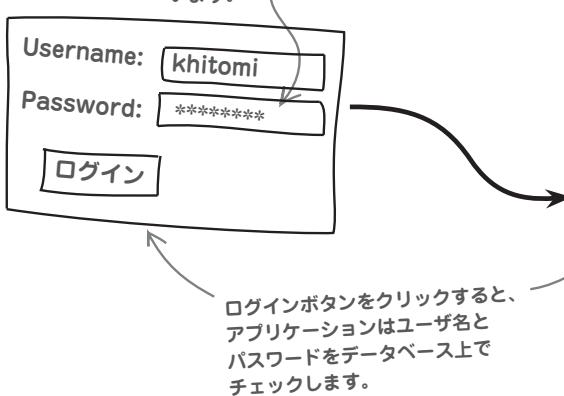
鋭い指摘です…パスワードには暗号化です。

ミスマッチサイトにおける暗号化とは、パスワードをデータベースに突っ込むときには認識できないフォーマットに変換することを指します。どんなアプリケーションでもユーザがログインするのであれば、パスワードを暗号化しなければなりません。そうでなければユーザは自分のパスワードが安全でセキュリティが保たれているという確信が持てません。ユーザのパスワードがバレてしまうというのは、たとえデータベース内であっても許されることではありません。つまりパスワードは暗号化してから、mismatch_user テーブルに突っ込まれなければなりません。ただ問題なのは、暗号化の前にまずユーザがユーザ名とパスワードを打ち込んで実際にログインする方法が必要だということです…

ログイン・インターフェースを作る

ユーザのログインデータを持っておくためにデータベースを変更しましたが、相変わらずユーザがアプリケーションにデータを打ち込んで、実際にログインできるようにする方法が必要です。このログイン用ユーザインターフェースは、ユーザ名とパスワード用のテキスト編集フィールドでできています。もちろんログインを実行するためのボタンも必要です。

パスワードフィールドは、
打ち込んだパスワードが
読めないように保護されて
います。



Q: まさかアスタリスクがデータベースにそのまま突っ込まれるんじゃないですよね？

A: もちろん違います。アスタリスクはパスワード用フォームフィールドが視覚的なセキュリティを提供しているだけです。誰かが後ろから（肩ごしに）パスワードを覗き見ても分からないようにしているのです。フォームが「提出」されると、本当に打ち込まれたパスワードが「提出」されます。アスタリスクではありません。従ってパスワードを暗号化してから、データベースに突っ込むことが重要なわけです。



もしユーザ名もパスワードもまだ割り当てられていないユーザが、どうやってログインすることができるようになるのか心配しているのでしたら…まだアセらなくて大丈夫です。

ちょっと後でユーザ毎にユーザ名とパスワードを作ることをやります。今はまだログインの第1段階にとどまっていることが大事です。まだまだやることはたくさん残っていますが、最後に全部をまとめ上げることになります。

パスワードをSHA()で暗号化する

ログインのユーザインターフェースは極めて単純です。ただ、ログインパスワードを暗号化する問題が残っています。MySQLにはSHA()という関数があって、テキストの文字列を暗号化してくれるアルゴリズムを備えています。呼び出しの結果は暗号化された文字列で、常に40文字分の16進数値です。元のパスワード長には関係ありません。関数は40文字のコードを作り出し、パスワードを一意に表現します。

SHA()はMySQLの関数であってPHPの関数ではありませんので、これを呼び出すときは、パスワードをテーブルに突っ込む際の問い合わせ文の一部とする必要があります。例えば、以下のコードで新しいユーザをmismatch_userテーブルに突っ込むことができますが、その際にSHA()を使ってパスワードを暗号化していることに注意して下さい。

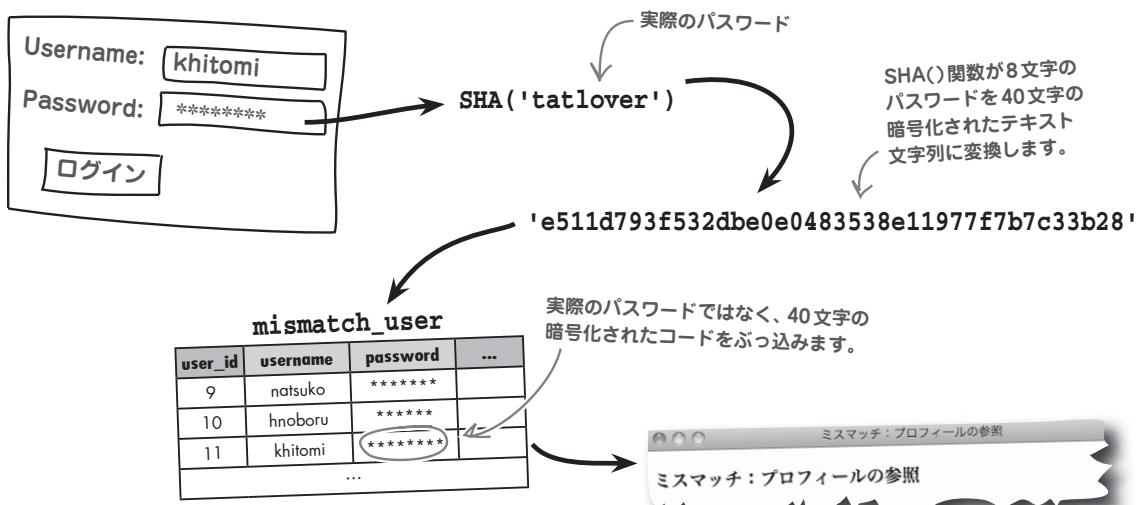
```
INSERT INTO mismatch_user
```

```
(username, password, join_date) VALUES ('khitomi', SHA('tatlover'), NOW())
```

SHA()関数でパスワードを暗号化して40文字の
16進コードにしてから、mismatch_userテーブル
のpasswordカラムにぶち込みます。

MySQLの
SHA()関数は、
テキストを暗号化して
一意な40文字の
コードにします。

同じくSHA()関数を使って、ログインが可能かどうかの比較をします。つまりユーザが打ち込んだパスワードが、データベースに暗号化してぶち込まれているパスワードと一致しているかどうかをチェックするのです。



比較 パスワードを復号化する

ひとたび情報を暗号化したら、どこかで復号化すると考えるのが自然な発想というものです。ところがSHA()関数は一方通行の暗号化関数で、元に戻す手段はありません。これにより暗号化されたデータのセキュリティを維持することができます。たとえ誰かがデータベースのハックに成功し、すべてのパスワードデータを盗み出したとしても大丈夫です。データは復号化できないのですから。では、ユーザがログインする時はどうすれば良いのでしょうか？パスワードは復号化できないので困りませんか？

実はユーザのもともとのパスワードは知らないでも良いのです。パスワードをちゃんと打ち込んでくれればログインできます。なぜならSHA()は同じテキスト文字列を打ち込みさえすれば、同じ40文字のコードを作り出すからです。つまりユーザが打ち込んだログインパスワードを単に暗号化し、それをmismatch_userテーブルのpasswordカラムにある値と比較すればよいのです。この程度のことには、SQL問い合わせ文一発で、マッチするユーザ行を暗号化パスワードに基づいて抽出すれば良いだけです。

```
SELECT * FROM mismatch_user
WHERE password = SHA('tatlover')
```

これがログインする際にユーザが打ち込んだパスワードです。

SHA()関数は、パスワードを暗号化するためには、WHERE節におきます。

このSELECT問い合わせ文はmismatch_userテーブルからpasswordカラムが入力されたパスワード、この場合だと「tatlover」とマッチするすべての行を抽出します。ここでは暗号化されたパスワード同士を比較するので、もともとのパスワードを知る必要はないのです。実際に特定のユーザとしてログインする際にもSHA()を使うことになりますが、それにはまずユーザIDに対するSELECTも必要になります。これについてはすぐ後でやりますので少々お待ち下さい。

暗号化されたパスワード用の場所を確保する

SHA()関数はミスマッチサイトに対して問題提起しています。というのは暗号化されたパスワードは40文字分の長さが必要なのですが、新しく作ったpasswordカラムは16文字分しかないからです。ALTERを使って、passwordカラムを広げて、暗号化パスワードをぶち込むようにします。

```
ALTER TABLE mismatch_user
CHANGE password password VARCHAR(40) NOT NULL
```

passwordカラムの大きさを40に変えて、
暗号化パスワードが入るようにします。

SHA()関数は一方通行の暗号化を提供します。暗号化されたデータを復号化することはできません。

素朴な疑問に答えます

Q : SHA()って何の略ですか？

A : SHA()関数は、安全なハッシュアルゴリズム(Secure Hash Algorithm)の頭文字です。ハッシュというのはプログラム上の専門用語で、一意な固定長文字列を意味し、テキスト文字列を一意に表現することができます。SHA()の場合で言うと、ハッシュは40文字分の16進数で暗号化されたテキスト文字列でもともとのパスワードを一意に表現することができます[†]。

Q : パスワードを暗号化する方法は他にもあるのでしょうか？

A : あります。MySQLではSHA()と同様な別の関数MD5()というものもあって同種の暗号化を提供しています。ただSHA()アルゴリズムの方が、MD5()に比べて若干セキュリティが高いと考えられています。つまりSHA()を使った方がよいということになります。PHPにも同等な関数(sh1()とmd5())がありますので、SQL問い合わせ文の中ではなく、PHPコードで何か暗号化する必要があればこちらを使えます。

[†] 訳注：この説明はハッシュの定義としてもSHAの定義としても厳密には正しくありません。しかし今ここで、あまり目くじらをたてるこどものないので、このまま流します。



mismatch_user テーブルにusername カラムとpassword カラムを追加して試してみます。

MySQLツールを使って、ALTER文を実行し、mismatch_user テーブルにusername カラムとpassword カラムとを追加します。

```
ALTER TABLE mismatch_user ADD username VARCHAR(32) NOT NULL AFTER user_id,  
ADD password VARCHAR(16) NOT NULL AFTER username
```

ただしpasswordカラムは実際には40文字の暗号化文字列を保存できなければならぬので、ALTERでテーブルをもう一度変更し、パスワードのための場所をもっと大きくします。

```
ALTER TABLE mismatch_user  
CHANGE password password VARCHAR(40) NOT NULL
```

次に新しいカラムをテストするため、INSERTで新しいユーザを追加してみます。

```
INSERT INTO mismatch_user  
(username, password, join_date) VALUES ('jimi', SHA('heyjoe'), NOW())
```

SHA() 関数を呼び出してパスワードを忘れずに暗号化して下さい。

パスワードが確かにデータベース内で暗号化されていることをもう一度チェックするため、SELECTを走らせて新しいユーザのパスワードを見てみます。

```
SELECT password FROM mismatch_user WHERE username = 'jimi'  
最後にログインのチェックをシミュレーションしてみます。ユーザ名とSHA() を適用したパスワードをWHERE節に与えて、SELECTします。
```

ログインに成功したとすれば、これは行を突っ込んだときと同じパスワードのはずです。

```
SELECT username FROM mismatch_user WHERE password = SHA('heyjoe')
```

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe  
mysql> USE mismatchdb;  
Database changed  
mysql> SELECT username FROM mismatch_user WHERE password = SHA('heyjoe');  
+-----+  
| username |  
+-----+  
| jimi |  
+-----+  
1 row in set (0.00 sec)  
mysql>
```

唯一のユーザが暗号化パスワードにマッチします。

HTTP認証でユーザとしてログインする

ログイン用スクリプト(`login.php`)[†]は、HTTP認証ヘッダを使って、ユーザ名とパスワードをユーザにもらうためのものです。ユーザ名とパスワードは`$_SERVER`スーパー全局から取ってくことができ、それを`mismatch_user`データベースでチェックしてから、制限されたページへアクセスできるようにします。

```
<?php
require_once('connectvars.php');

if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW'])) {
    // The username/password weren't entered so send the authentication headers
    header('HTTP/1.1 401 Unauthorized');
    header('WWW-Authenticate: Basic realm="Mismatch"');
    exit('<h3>ミスマッチ</h3> エラー：正しいユーザ名とパスワードを入力して下さい。');
}

// Connect to the database データベースへの接続
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME); ユーザ名とパスワードを取ってきます。

// Grab the user-entered log-in data ユーザが入力したログインデータの取得
$user_username = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_USER']));
$user_password = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_PW'])); ユーザが打ち込んだユーザ名とパスワードを取ってきます。

// Look up the username and password in the database ユーザ名とパスワードとをデータベースから探す
$query = "SELECT user_id, username FROM mismatch_user WHERE username = '$user_username' AND " .
    "password = SHA('$user_password')";
$data = mysqli_query($dbc, $query); マッチする行があれば、それはログインしてもよいことを意味します。

if (mysqli_num_rows($data) == 1) { この場合$user_idと$username変数をセットします。
    // The log-in is OK so set the user ID and username variables
    $row = mysqli_fetch_array($data);
    $user_id = $row['user_id'];
    $username = $row['username']; ログインに成功したのでユーザIDとユーザ変数を設定

}
else {
    // The username/password are incorrect so send the authentication headers
    header('HTTP/1.1 401 Unauthorized');
    header('WWW-Authenticate: Basic realm="Mismatch"'); ユーザ名とパスワードにマッチするデータベース行がない場合、認証ヘッダをもう一度送りつけ、ユーザに入力を促します。
    exit('<h2>ミスマッチ</h2> エラー：ユーザ名とパスワードが入力されていません。');

}

// Confirm the successful log-in ログイン中であることの確認表示
echo('<p class="login">' . $username . 'さんとしてログインしています.</p>');

?> 現時点では全部イケてますので、ログインが成功したことを示します。
```

[†] 訳注：ログイン用スクリプトをHTTP認証で作ることに関しては、ギター ウォーズのところでやったことと本質的に変わりありません。また、本章ではこれ以外のログインについて学びます。



新しくログイン用スクリプトを作り、ユーザにユーザ名とパスワードを入力するよう促します。

終了



試運転

**新しくログイン用スクリプトを作り、プロフィール参照用スクリプトと
プロフィール編集用スクリプトからインクルードします。**

新しくテキストファイルを作りlogin.phpという名前にし、ログイン用スクリプトのコードを打ち込みます(またはスクリプトはWebサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードすることもできます)。次にviewprofile.phpスクリプトとeditprofile.phpスクリプトにPHPコードを追加し、新しく作ったログイン用スクリプトをインクルードします。

すべてのスクリプトをWebサーバにアップロードしたら、ミスマッチサイトのメインページをWebブラウザで開きます。プロフィール参照用ページやプロフィール編集用ページのリンクをクリックしてログインし、個人用のページにアクセスしてみます。当然ながら、この操作がうまくいくためには、すでにユーザのユーザ名とパスワードをデータベースに登録しておかなければなりません。

この2つのリンクは保護されたページにつながっていますが、まだユーザがログインしていないければログイン用スクリプトを起動します。

ミスマッチ：正反対が魅力的！

- ♥ プロフィールの参照
- ♥ プロフィールの編集

新規メンバー：

	勝則
	昇
	奈津子
	岳之
	麻子

このパスワードをSHA()で暗号化し、データベース内のパスワードと比較し、ログイン可能かどうかを判定します。

このページを見るには、“192.168.1.2:80”上の領域 Mismatch にログインが必要です。
パスワードは暗号化されずに送信されます。

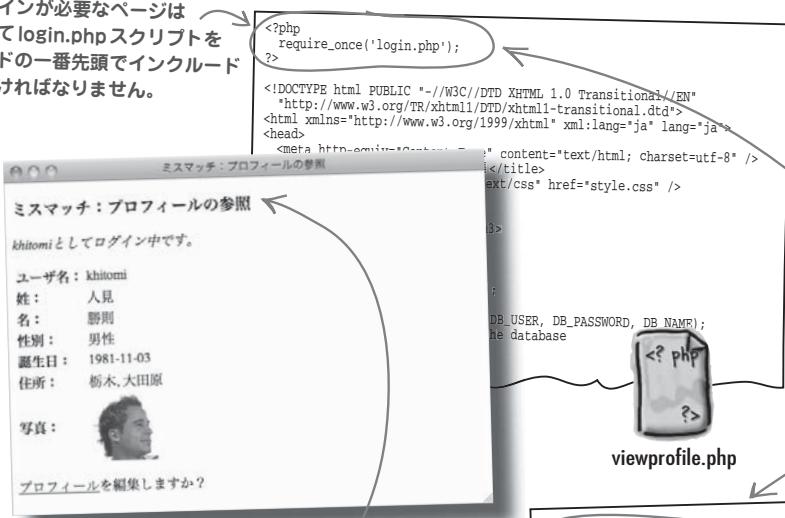
名前： キャンセル ログイン

パスワード： このパスワードをキーチェーンに保存

ログイン用スクリプトはHTTP認証を使ってプロフィール参照用ページやプロフィール編集用ページへの不正なアクセスを防ぎます。

ホームページはログイン用スクリプトで保護されていませんが、アプリケーションのディープな部分へ誘導するための出発地点としての役割があります。

ミスマッチサイトのページで
ログインが必要なページは
すべてlogin.phpスクリプトを
コードの一一番先頭でインクルード
しなければなりません。



```
<?php
require_once('login.php');
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>ミスマッチ：プロフィールの参照</title>
<link rel="stylesheet" type="text/css" href="style.css" />

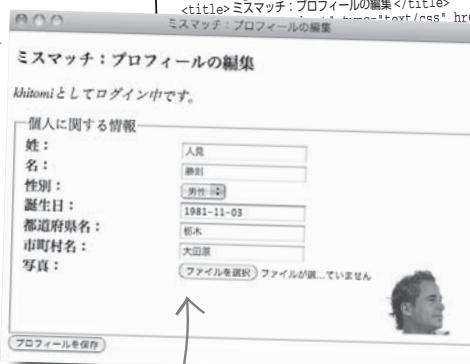
```

DB_USER, DB_PASSWORD, DB_NAME);
the database

viewprofile.php

ログイン用スクリプトはプロ
ファイル参照用スクリプトとブ
ロフィール閲覧用スクリプトの
先頭でインクルードされていて、
ユーザにログインを強制します。

どちらのページもログイン用
スクリプトによって提供された
確認メッセージが付いていて、
ログインが成功したことを
意味しています。



```
<?php
require_once('login.php');
?>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>ミスマッチ：プロフィールの編集</title>
<link rel="stylesheet" type="text/css" href="style.css" />
```

PASSWORD, DB_NAME);
the database

editprofile.php

ユーザー名とパスワードのチェック
が終わったら、ユーザはログイン
できて、残りのページを見ること
ができます。

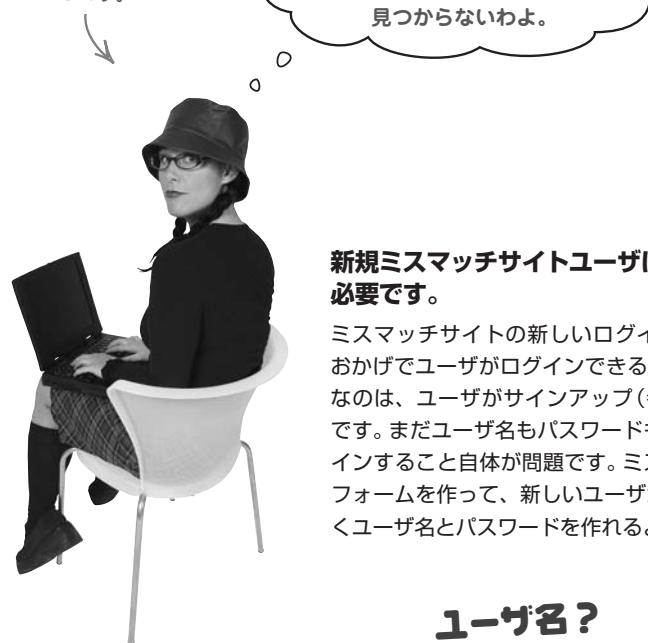
各ユーザは各個人専用に
カスタマイズされたミスマッチ
サイトのインターフェースです。

③

ログイン用スクリプトをミスマッチアプリ
ケーションの残りの部分にくっ付ける。

終ア

美奈子さんはホラー映画とルーピックキューブと激辛料理が好きですが、今のミスマッチサイトは嫌いです。自分がサインアップできないためシステムを使えないからです。



新規ミスマッチサイトユーザにはサインアップの方法が必要です。

ミスマッチサイトの新しいログイン用スクリプトはHTTP認証のおかげでユーザがログインできるようになりました。ところが問題なのは、ユーザがサインアップ(参加登録)をする方法がないことです。まだユーザ名もパスワードも持っていない人にとってはログインすること自体が問題です。ミスマッチサイトにサインアップ用フォームを作って、新しいユーザがサイトに参加できるよう、新しくユーザ名とパスワードを作れるようにする必要があります。

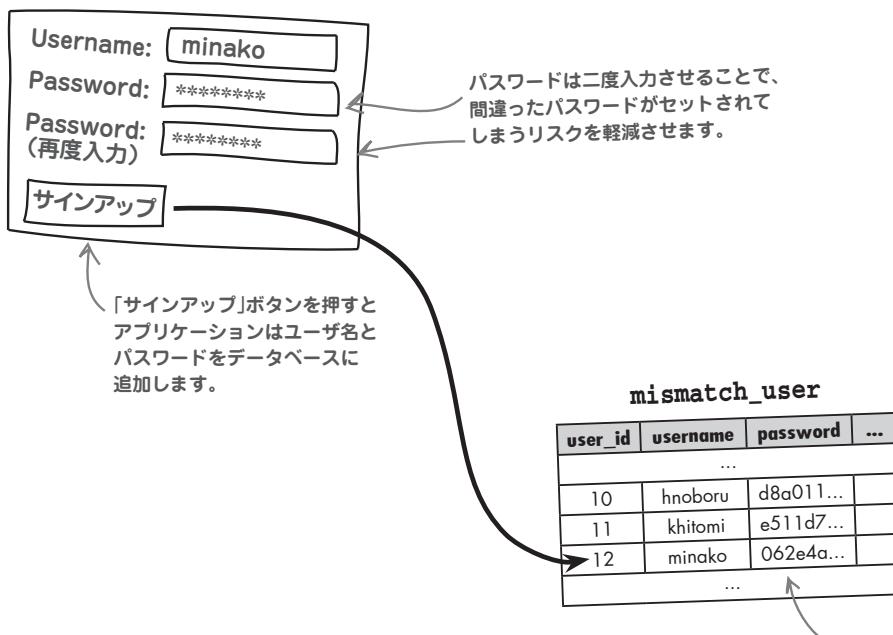
ユーザ名？

パスワード？

新しいユーザをサインアップするためのフォーム

新しいサインアップ用フォームはどんな感じになるでしょう？そのフォームでユーザが自分の好きなユーザ名とパスワードを打ち込めるようになっていなければならることは分かっていますが…他には何でしょう？ユーザは新しいサインアップ用フォームでパスワードを決めます。しかもパスワードはWebフォームでは普通アスタリスクで伏せ字として表示されセキュリティを確保します。そこで2つのパスワード用フォームフィールドを持たせるということはよいことです。ユーザにパスワードを2回打ち込ませることで打ち間違えがないことを確認するというわけです。

サインアップ用ページのやるべきことは、ユーザ名とパスワードをユーザに入力させ、ユーザ名が既に誰かに使われていないことを確認し、新しいユーザを mismatch_user データベースに追加することです。



サインアップ用スクリプトには、実は問題が残っています。ユーザが登録しようとしたユーザ名がすでに登録されている場合どうするかということです。スクリプトはこの問題を見つけたら、ユーザに違うユーザ名を試させるよう、ちゃんと対応できなければいけません。結局、サインアップ用ページのやるべきことは、ユーザ名とパスワードをユーザに入力させたら、ユーザ名が既に誰かに使われていないことを確認してから、新しいユーザを mismatch_user データベースに追加することになります。



PHP&MySQLマグネット

ミスマッチサイトのサインアップ用スクリプトは、専用のフォームを使ってユーザーに好みのユーザ名とパスワードとを入力させます。ただし問題が残っています。スクリプトコードはまだ完成していません。

下にあるマグネットを使って、スクリプトを仕上げ、新しいユーザがサインアップをして、ミスマッチサイトのコミュニティに入れるようにして下さい。

これがサインアップ用
フォームです。

```
<?php
require_once('appvars.php');
require_once('connectvars.php');

// Connect to the database
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

if (isset($_POST['submit'])) {
    // Grab the profile data from the POST
    $username = mysqli_real_escape_string($dbc, trim($_POST['username']));
    $password1 = mysqli_real_escape_string($dbc, trim($_POST['password1']));
    $password2 = mysqli_real_escape_string($dbc, trim($_POST['password2']));

    if (!empty($username) && !empty($password1) && !empty($password2) &&
        ($username == $password1)) {
        // Make sure someone isn't already registered using this username
        $query = "SELECT * FROM mismatch_user WHERE username = '$username'";

        $data = mysqli_query($dbc, $query);
        if (mysqli_num_rows($data) == 0) {
            // The username is unique, so insert the data into the database
            $query = "INSERT INTO mismatch_user (username, password, join_date) VALUES " .
                "('$username', SHA('$password1'), NOW())";

            mysqli_query($dbc, $query);

            // Confirm success with the user
            echo '<p>アカウントを作成しました。プロフィールを<a href='editprofile.php'>編集</a>することができます。</p>';
        }
    }
}

mysqli_close($dbc);
exit();
}
```

```

else {
    // An account already exists for this username, so display an error message
    echo '<p class="error">このユーザ名はすでに使われています。別のユーザ名をご利用下さい。</p>';

    .....= "";
}

}

else {
    echo '<p class="error">エラー：サインアップにはすべてのデータを入力する必要があります。'.
        ' パスワードは2回入力して下さい。</p>';
}

}

mysqli_close($dbc);
?>

<p>ユーザ名とパスワードを入力してミスマッチサイトにサインアップして下さい。</p>
<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
<fieldset>
<legend>Registration Info</legend>
<label for="username">ユーザ名:</label>

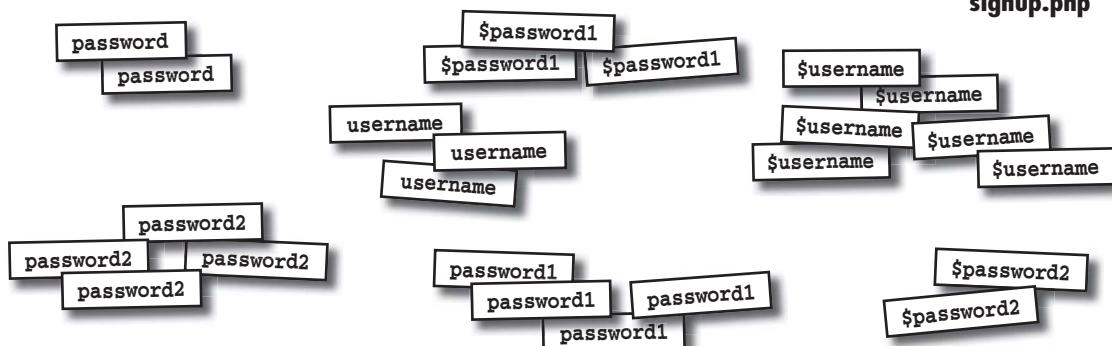
<input type="text" id="....." name="....."
       value="<?php if (!empty(.....)) echo.....; ?>" /><br />
<label for=".....">パスワード:</label>
<input type="....." id="....." name="....." /><br />
<label for=".....">パスワード（もう一度）:</label>
<input type="....." id="....." name="....." /><br />

</fieldset>
<input type="submit" value="サインアップ" name="submit" />
</form>

```



signup.php





PHP&MySQLマグネットの答え

ミスマッチサイトのサインアップ用スクリプトは、専用のフォームを使ってユーザーに好みのユーザ名とパスワードとを入力させます。ただし問題が残っています。スクリプトコードはまだ完成していません。

下にあるマグネットを使って、スクリプトを仕上げ、新しいユーザがサインアップをして、ミスマッチサイトのコミュニティに入れるようにして下さい。

```
<?php
require_once('appvars.php');
require_once('connectvars.php');

// Connect to the database
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

if (isset($_POST['submit'])) {
    // Grab the profile data from the POST
    $username = mysqli_real_escape_string($dbc, trim($_POST['username']));
    $password1 = mysqli_real_escape_string($dbc, trim($_POST['password1']));
    $password2 = mysqli_real_escape_string($dbc, trim($_POST['password2']));

    if (!empty($username) && !empty($password1) && !empty($password2) && $password1 == $password2) {
        // Make sure someone isn't already registered using this username
        $query = "SELECT * FROM mismatch_user WHERE username = '$username'";

        $data = mysqli_query($dbc, $query);
        if (mysqli_num_rows($data) == 0) {
            // The username is unique, so insert the data into the database
            $query = "INSERT INTO mismatch_user (username, password, join_date) VALUES ";
            $query .= "('$username', SHA('$password1'), NOW())";
            mysqli_query($dbc, $query);
        }
    }
}

// Confirm success with the user
echo '<p>アカウントを作成しました。プロフィールを<a href="editprofile.php">編集</a>することができます。</p>';

mysql_close($dbc);
exit();
}
```

これがサインアップ用フォームです。

ミスマッチ：サインアップ
ユーザ名とパスワードを入力してミスマッチサイトにサインアップしてください。
登録情報
ユーザ名: minako
パスワード: *****
パスワード（もう一度）: *****
サインアップ

ユーザが入力したデータを全部取ってきますが、まずはtrim()できれいにします。

どのフォームフィールドも空でないことを確認したら、2つのパスワードが一致することも確認します。

問い合わせ文を実行し、既存の行に打ち込まれたユーザ名と一致する行があるかどうかをみます。

一致する行が見つかなければ、ユーザ名は一意ですからINSERTすることができます。

ここではどっちのパスワードを使っても構いません。どちらも同じであることはこの時点では確認済みです。

サインアップが成功したことを確認メッセージで示してスクリプトは終了します。

```

else {
    // An account already exists for this username, so display an error message
    echo '<p class="error">このユーザ名はすでに使われています。別のユーザ名をご利用下さい。</p>';
}
$username = "";
    $username変数を初期化
}           して、フォームフィールドを
            クリアします。
}
else {
    echo '<p class="error">エラー：サインアップにはすべてのデータを入力する必要があります。'.
        'パスワードは2回入力して下さい。</p>';
}
}
mysqli_close($dbc);
?>

<p>ユーザ名とパスワードを入力してミスマッチサイトにサインアップして下さい。</p>
<form method="post" action="<?php echo $_SERVER['PHP_SELF']; ?>">
<fieldset>
<legend>Registration Info</legend>
<label for="username">ユーザ名:</label>
<input type="text" id="username" name="username" value="<?php if (!empty($_POST['username'])) echo... $username ?>" /><br />
<label for="password1">パスワード:</label>
<input type="password" id="password1" name="password1" /><br />
<label for="password2">パスワード(もう一度):</label>
<input type="password" id="password2" name="password2" /><br />
</fieldset>
<input type="submit" value="サインアップ" name="submit" />
</form>

```



signup.php

素朴な疑問に答えます

Q: 単純にHTTP認証を使って新しいユーザのサインアップをするのはダメなんですか？

A: ダメです。というのもサインアップ用スクリプトの目的は、ページへのアクセスを制限することではないからです。サインアップ用スクリプトでやるべきことは、ユーザに一意なユーザ名とパスワードとを入力してもらうことであり、その後これらのデータをデータベースに追加することです。確かにHTTP認証ウインドウを入力用フォームに使ってユーザ名とパスワードを入力させることはできます。しかし認証の機能を使うというのは、単に新しいユーザをサインアップさせるにはやりすぎです。サインアップのための専用フォームを作った方が全然良いでしょう。そうすればパスワードのデータ入力エラーに関する二重チェックという御利益も得られます。

Q: ではサインアップ用スクリプトで登録が終わったらそのままログインしているのですか？

A: しません。その主な理由はログイン用スクリプトはユーザがログインする処理をすでに終えてしまっているからです。サインアップ用のスクリプトで同じコードを書くというのはバカげています。その代わりにサインアップ用スクリプトにはプロファイル編集用ページへのリンクが貼ってあります。ユーザはサインアップを済ませたら、恐らくはこのページに行きたいと思っているはずです。そしてユーザはまだログインしていませんから、プロファイル編集用ページにアクセスしようとするとログイン用ウインドウが現れます。つまりサインアップ用スクリプトから、ユーザにログインウインドウを通してプロファイル編集用ページまで行き着くことができます。自動的にそのページにログインするのではありません。

ユーザにサインアップのチャンスを

サインアップ用スクリプトはできましたが、ユーザはどうすればそこにたどり着けるのでしょうか？ユーザにサインアップの方法を知らせる必要があります。1つの手はミスマッチサイトのメインページに「サインアップ」リンクを貼ることでしょう。それはそれで悪くはありませんが、理想を言えばユーザがログインしているかどうかに基づいて、リンクが出ていたり、出ていなかつたりしていて欲しいところです。もう1つの可能性として、「サインアップ」リンクをログイン用スクリプトの一部として単純に表示するというのも考えられます。

新規ユーザがメインページで例えば「プロフィール参照用」リンクまたは「プロフィール編集用」リンクをクリックすると、ユーザはユーザ名とパスワードの入力をログイン用スクリプトに求められます。ところがユーザはまだユーザ名とパスワードを登録していませんから、キャンセルをクリックしてログイン画面から脱出しようとします。その時がチャンスで「サインアップ」リンクを表示することができます。ログインに失敗した際のメッセージを表示するところをちょっと変更して、ログインスクリプトからsignup.phpへのリンクを付けてしまえばよいのです。

以下がログインに失敗した際の元々のコードです。

このコードはエラーメッセージを表示する
だけで、ミスマッチサイトへサインアップ
する方法については教えてくれません。

```
exit('<h3>ミスマッチ : </h3> エラー：ユーザ名とパスワードを確認して下さい。');
```

このコードはログイン用スクリプトの2箇所に出てきています。ユーザ名かパスワードのいずれかが打ち込まれなかった場合とこれらのデータを間違って入力した場合です。ここで先へ進めるように「サインアップ」リンクを両方の場所に付けてあげるというのは多分イケてます。コードを新しく書いたら多分こんな感じになるでしょう。

このコードの方が全然マシです。サインアップ
スクリプトへのリンクがあるので、ユーザは
登録処理をることができます。

```
exit('<h2>ミスマッチ : </h2> エラー：正しいユーザ名とパスワードを入力して下さい。' .
```

```
' 未登録の場合は、まず<a href="signup.php">サインアップ</a>して下さい。');
```

これはちっとも変じやあり
ません。signup.phpへの
ただのHTMLリンクです。



ミスマッチサイトにサインアップ機能を追加する

新しくテキストファイルを作り `signup.php` という名前にします。サインアップ用スクリプトのコードを打ち込みます(またはWebサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からスクリプトをダウンロードします)。次に、`login.php`を修正し、ログインできないユーザのためにサインアップ用スクリプトへのリンクを追加します。

できあがったスクリプトをWebサーバにアップロードしたら、サインアップ用ページをWebブラウザで開きます。新しいユーザとしてサインアップをし、ログインしてみて下さい。次に自分のプロフィールを編集し、自分のプロフィールを参照して、サインアップやログインがちゃんと動いていることを確認します。アプリケーションには今までなかった個人向け側面が加わりました。

ミスマッチ：サインアップ

ミスマッチ：プロフィールの編集

ミスマッチ：プロフィールの参照

美奈子さんは
HTTP認証を使い、
サインアップの
情報に基づいて
ログインします。

イケてるわ！私も
ミスマッチサイトにログイン
して、私自身のプロフィールを
編集したり参照したり
できるし。

美奈子さんのプロフィール
はログインした後でのみ
アクセスできます。

313



アタシ、ルームメイト2人と一緒にパソコンを使っているのよ。ルームメイトにはアタシのミスマッチサイト・プロフィールにアクセスしてもらいたくないわ。ログアウトできる必要があるんだけど！

コミュニティーのWebサイトはユーザがログアウトできるようになっていなければなりません。そうでないと同じコンピュータを使っている他人がその人のデータにアクセスできてしまいます。

ユーザがログアウトできるようにするというのは簡単に聞こえるかもしれません、実はHTTP認証では極めて大きな問題を抱えています。問題というのは、HTTP認証のそもそもの作りにあります。一度ページが見えるようになったら、ブラウザ 자체を閉じない限り認証はリセットされないような設計思想なのです。言い換えば、ユーザはHTTP認証のWebページで「ログアウト」することはできないのです。ブラウザを閉じるかまたはユーザが手動でHTTP認証のセッションを切るしかありません。ブラウザによって(Firefoxなど)は、セッションを切ることが(Safariなど)他に比べて簡単にできる場合もあります。

ひとたびログインすると、
ブラウザ自体を閉じる
まではログイン状態に
留まります。

ミスマッチ：プロフィールの編集

skarenaとしてログイン中です。

個人に関する情報

姓：
名：
性別： 女性 男性
誕生日：
都道府県名：
市町村名：
写真：
 ファイルが選...いません

プロフィールを保存

このページを見るには、“192.168.11.2:80”上の領域 Mismatch にログインが必要です。
パスワードは暗号化されずに送信されます。

名前：
パスワード：
 このパスワードをキーチェーンに保存

キャンセル ログイン

ログアウト機能があれば
加理奈さんは自分個人の
プロフィールへのアクセス
を気をつけてコントロール
することができます。

HTTP認証のおかげで手軽で簡単にユーザはミスマッチサイトアプリケーションにログインできるようになりましたが、ユーザがログアウトするための制御は提供されません。今必要なのはユーザを覚えることといつでも好きなときにログアウトできるようになることの両方なのです。



永遠にログインし続けるんじゃなくて、
必要な時だけユーザを覚えておくことが
できたらステキね。それってPHPじゃあ
見込みのない夢物語なのかしら？

単にクッキーが必要なだけかもしれません

問題は、HTTP認証の持つ二面性に起因しています。特定のページへのアクセスを制限するという面とユーザが自分自身の情報を打ち込んだらそれを覚えておくという面です。二番目の問題は少々やっかいです。というのは、複数のページ(スクリプト)にまたがって有効なユーザは誰なのかをアプリケーションが覚えていなければならぬからです。ミスマッチサイトでは、これを\$_SERVERスーパーグローバルに入っているユーザ名とパスワードとをチェックすることで成し遂げています。つまりPHPがHTTP認証のユーザ名とパスワードとをスーパーグローバルに格納していることにより複数のページにまたがって永続していられるということによる御利益を受けているわけです。

このページを見るには、"192.168.11.2 80"上の領域 Mismatch にログインが必要です。
パスワードは暗号化されずに送信されます。

名前: skarena
パスワード: *****
このパスワードをキーチェーンに保存
キャンセル ログイン

`$_SERVER['PHP_AUTH_USER']`

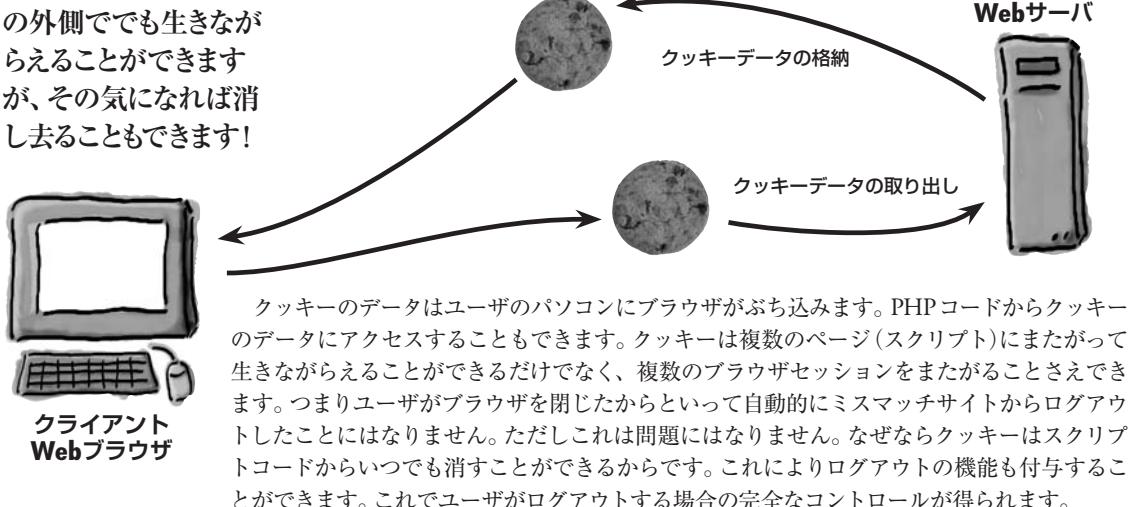
`$_SERVER['PHP_AUTH_PW']`

HTTP認証はデータを永続的にクライアントに格納しますが、終わってもそれを削除することができません。

`$_SERVER`スーパーグローバルはユーザ名とパスワードとを永続的に格納します。

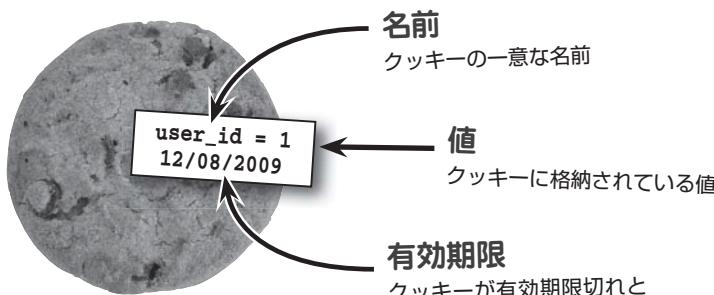
クッキーにはクライアント側に永続的にちょっとしたデータを突っ込むことができます。クッキーはどんなスクリプトの外側ででも生きながらえることができますが、その気になれば消し去ることもできます！

しかしHTTP認証で贅を尽くせるのはこれだけです。なぜならログアウトできないからです。そこで、複数ページにまたがってユーザが生きながらえることのできるものを他から探す必要があります。その答えはクッキーに眠っているかもしれません。クッキーとはユーザのパソコンにブラウザがぶち込んだデータのことです。クッキーはPHP変数に非常に似ていますが、クッキーはブラウザを閉じたり、パソコンの電源を切ったりしてもなくなりません。もっと重要なのはクッキーを消すことができることです。つまりデータをぶち込み終わったら抹消してもよいのです。例えばユーザがログアウトしたいような場合です。



クッキーの中にあるものは？

クッキーは特定のユーザに関するデータを1つだけぶち込むためのもので、PHP変数と似ています。変数と異なるのは、クッキーが有効期限付きだということです。有効期限が過ぎたクッキーは破壊されます。つまりクッキーは不死身ではありません。単にPHP変数よりも長生きなだけです。有効期限をつけないでクッキーというのを作ることもできます。この場合はPHP変数と同様で、ブラウザを閉じる際に破壊されることになります。



クッキーを使うことによりテキストの文字列を、PHPのテキスト用変数と同様に、特定の名前で突っ込むことができます。クッキーは通常のスクリプトデータよりも長生きなため非常に強力です。アプリケーションが複数のページでできていって、ログイン情報のような何らかのデータを共有する必要があるような状況で特に便利です。



そこでミスマッチサイトでも\$_SERVERスーパーグローバルから供給される永続性を2つのクッキーにセットすることにします。1つはユーザ名でもう1つはパスワードです。実際にはパスワードを覚えておく必要はないので、代わりにユーザIDを覚えていた方が役に立つかかもしれません。

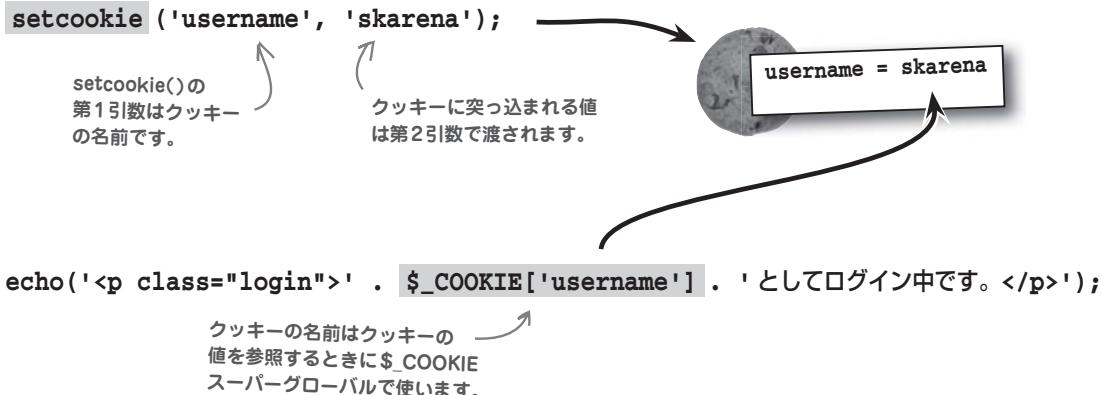
素朴な疑問に答えます

Q: クッキーの永続性で何がメデタイなん？ MySQLデータベースに突っ込まれたデータも永続性あるんちゃうの？

A: その通りです。データベースのデータは間違いなく永続的です。技術的にはクッキーよりも全然永続的です。データベースには有効期限なんてありませんから。データをデータベースに突っ込んだら、明示的に消し去るまではずっとそこに留まります。クッキーと永続性に関する実用上の問題は、むしろ利便性にあります。その時点でのユーザIDやユーザ名を単にプロフィールデータへのアクセスを許可するというだけの目的で完全な永続データを作る必要はありません。欲しいのはそれが誰であるのかを知るための手早い方法です。本当に欲しいものは一時的な永続性なのです。これは矛盾しているように聞こえるかもしれません。しかしページよりも長生き（永続的）であって、無期限ではないデータが必要だという風に考えてみて下さい。

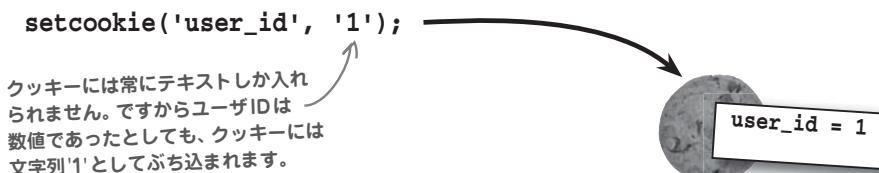
使う PHPでクッキーを焼く

PHPではクッキーへのアクセスを `setcookie()` という関数と `$_COOKIE` というスーパーグローバルを通して行います。`setcookie()` 関数は値とオプションでクッキーの有効期限とをセットするのに使います。`$_COOKIE` スーパーグローバルはクッキーの値を取ってくるのに使います。



クッキーにセットすることにより得られる破壊力は、クッキーのデータが複数のスクリプトにまたがって永続的であるということです。これによりユーザ名を覚えておくことができますから、ユーザにアプリケーション内でページを移動するたびにログインさせる必要がありません。ただし忘れてならないのは、ユーザIDもクッキーに突っ込んでおく必要があるということです。これがデータベース問い合わせ文の主キーという役割を担っているからです。

PHP の `setcookie()` 関数を使うとデータをクッキーに突っ込むことができます。



`setcookie()` 関数はオプションで第3引数を取ることができ、クッキーの有効期限を設定します。この日をもってクッキーは自動的に削除されます。上記の例のように有効期限を指定しなかった場合、クッキーはブラウザを閉じたときに自動的に有効期限切れとなります。

自分で考えてみよう

ミスマッチサイトでクッキーを使うように変更して、新しくログアウト用スクリプトを書くとの同等のことをしてみましょう。最初にログイン用スクリプトを再検討し、HTTP認証の代わりにクッキーを使うように変更します。ログインコードのうちクッキーに合わせて変更が必要と思う箇所にマルをつけ説明を書いて下さい。

```
<?php
require_once('connectvars.php');

if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW'])) {
    // The username/password weren't entered so send the authentication headers ←
    header('HTTP/1.1 401 Unauthorized');
    header('WWW-Authenticate: Basic realm="Mismatch"');
    exit('<h3>ミスマッチ:</h3>エラー:正しいユーザ名とパスワードを入力して下さい。' . ←
        '未登録の場合は、まず<a href="signup.php">サインアップ</a>して下さい。');
}

// Connect to the database データベースへの接続
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

// Grab the user-entered log-in data ユーザが入力したログインデータの取得
$user_username = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_USER']));
$user_password = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_PW']));

// Look up the username and password in the database ユーザ名とパスワードをデータベースから探す
$query = "SELECT user_id, username FROM mismatch_user WHERE username = " .
    "'$user_username' AND password = SHA('$user_password')";
$data = mysqli_query($dbc, $query); ←
    if (mysqli_num_rows($data) == 1) { ←
        // The log-in is OK so set the user ID and username variables ←
        $row = mysqli_fetch_array($data);
        $user_id = $row['user_id'];
        $username = $row['username'];
    } ←
    else { ←
        // The username/password are incorrect so send the authentication headers ←
        header('HTTP/1.1 401 Unauthorized');
        header('WWW-Authenticate: Basic realm="Mismatch"');
        exit('<h3>ミスマッチ:</h3>エラー:正しいユーザ名とパスワードを入力して下さい。' . ←
            '未登録の場合は、まず<a href="signup.php">サインアップ</a>して下さい。');
    }

// Confirm the successful log-in ログイン中であることの確認表示
echo('<p class="login">' . $username . 'としてログイン中です。</p>');
?>
```



login.php

自分で考えてみよう の答え

クッキーがあるかどうか
チェックして、ユーザが
ログインしているかどうか
を確認します。

ミスマッチサイトでクッキーを使うように変更して、新しくログアウト用スクリプトを書くのと同等のことをしてみましょう。最初にログイン用スクリプトを再検討し、HTTP認証の代わりにクッキーを使うように変更します。ログインコードのうちクッキーに合わせて変更が必要と思う箇所にマルをつけ説明を書いて下さい。

認証ウインドウからユーザ名とパスワードを
取ってくる代わりに、フォームを使った
POSTデータが必要になります。

もうHTTP認証
ヘッダを送る必要
はありません。

```
<?php
require_once('connectvars.php');

if (!isset($_SERVER['PHP_AUTH_USER']) || !isset($_SERVER['PHP_AUTH_PW'])) {
    // The username/password weren't entered so send the authentication headers
    header('HTTP/1.1 401 Unauthorized');
    header('WWW-Authenticate: Basic realm="Mismatch"');
    exit('<h3>ミスマッチ:</h3>エラー:正しいユーザ名とパスワードを入力して下さい。'.
        '未登録の場合は、まず<a href="signup.php">サインアップ</a>して下さい。');

}

// Connect to the database
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

// Grab the user-entered log-in data
$user_username = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_USER']));
$user_password = mysqli_real_escape_string($dbc, trim($_SERVER['PHP_AUTH_PW']));

// Look up the username and password in the database
$query = "SELECT user_id, username FROM mismatch_user WHERE username = ".
    "'$user_username' AND password = SHA('$user_password')";
$data = mysqli_query($dbc, $query);

if (mysqli_num_rows($data) == 1) {
    // The log-in is OK so set the user ID and username variables
    $row = mysqli_fetch_array($data);
    $user_id = $row['user_id'];
    $username = $row['username'];
}
else {
    // The username/password are incorrect so send the authentication headers
    header('HTTP/1.1 401 Unauthorized');
    header('WWW-Authenticate: Basic realm="Mismatch"');
    exit('<h3>ミスマッチ:</h3>エラー:正しいユーザ名とパスワードを入力して下さい。'.
        '未登録の場合は、まず<a href="signup.php">サインアップ</a>して下さい。');
}

// Confirm the successful log-in
echo('<p class="login">' . $username . 'としてログイン中です。</p>');
?>
```

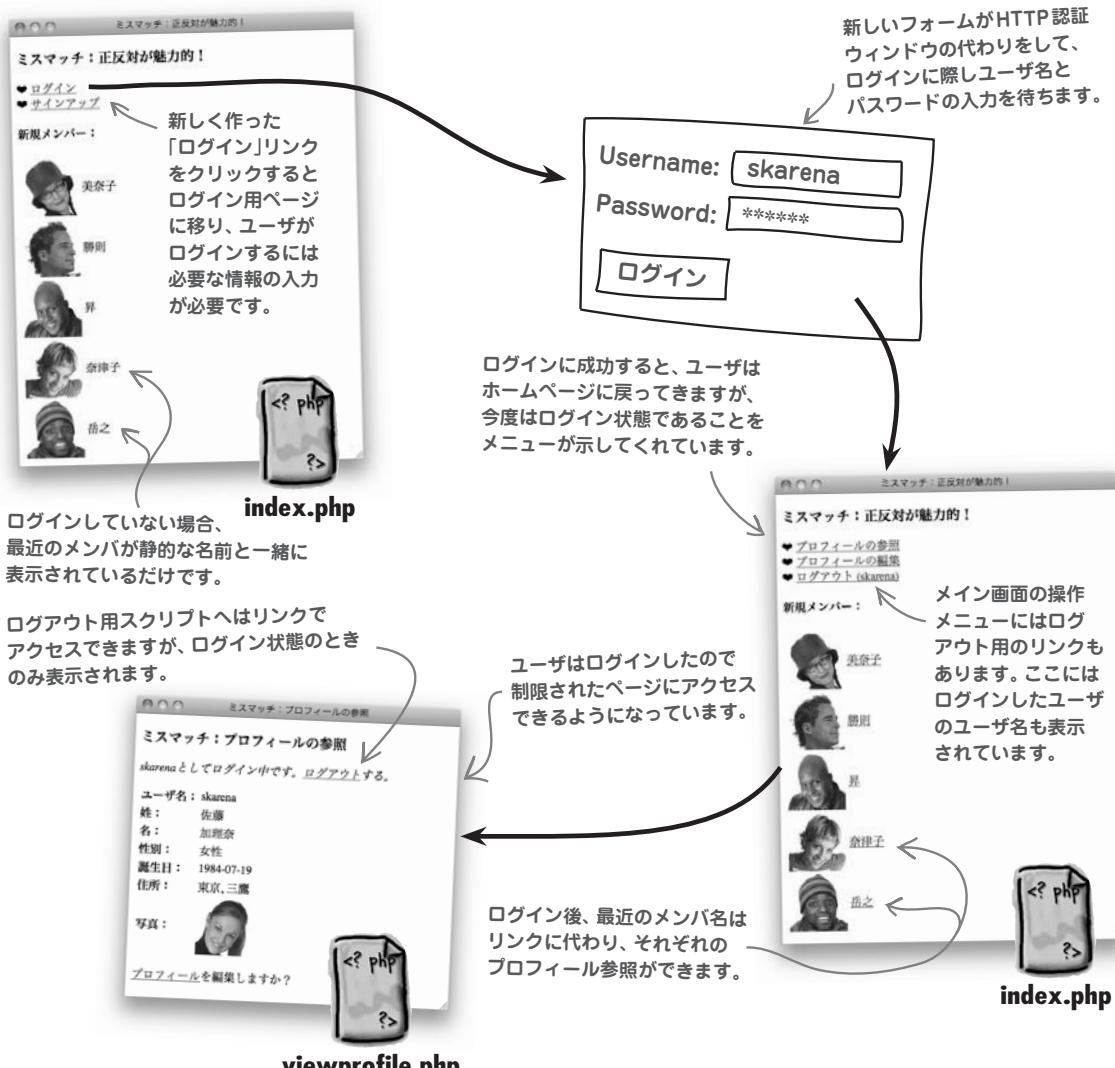
HTTP認証用ウインドウに頼ってユーザ名とパスワードを
打ち込んでもらうことはできないので、HTMLでログイン用
フォームを作り、これらを打ち込んでもらう必要があります。



問い合わせ文は全く変わ
る必要がありません。
ここでスクリプト変数の
代わりに、クッキーを2つ
セットする必要があります。

ログインのフローを再考する

ミスマッチサイトのログインにHTTP認証ではなくクッキーを使ったため、ユーザデータの格納先を再考するだけでは済まなくなりました。ログインのユーザインターフェースはどうなるのでしょうか？クッキー強化版のログインでは自前のフォームを作らなければなりません。なぜならHTTP認証ウィンドウに頼ってユーザ名とパスワードを打ち込んでもらうことは、もはやできないからです。専用のフォームを作らなければならないだけでは話はすみません。クッキーによりユーザがログインしてから他のページにアクセスするまで、アプリケーションのフローがどのように変わったのかを考える必要があります。



クッキー強化版ログイン

新しい版のログイン用スクリプトは、クッキーを使ってログインを管理します。このため永続性については以前よりも若干複雑になりました。自前でフォームを用意してユーザ名とパスワードを打ち込まなければなりません。しかし、ログアウトという小技も使えるようになったという意味で強力になりました。

```
<?php
require_once('connectvars.php'); // エラーメッセージの初期化
// Clear the error message
$error_msg = ""; // ユーザがログインしていない場合、ログインを促す
// If the user isn't logged in, try to log them in
if (!isset($_COOKIE['user_id'])) {
    if (isset($_POST['submit'])) { // データベースへの接続
        // Connect to the database
        $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME); // ユーザが入力したログインデータの取得
        // Grab the user-entered log-in data
        $user_username = mysqli_real_escape_string($dbc, trim($_POST['username']));
        $user_password = mysqli_real_escape_string($dbc, trim($_POST['password']));
        if (!empty($user_username) && !empty($user_password)) { // Look up the username and password in the database
            $query = "SELECT user_id, username FROM mismatch_user WHERE username = '$user_username' AND ";
            "password = SHA('$user_password')";
            $data = mysqli_query($dbc, $query);
            if (mysqli_num_rows($data) == 1) { // The log-in is OK so set the user ID and username cookies, and redirect to the home page
                $row = mysqli_fetch_array($data);
                setcookie('user_id', $row['user_id']);
                setcookie('username', $row['username']);
                $home_url = 'http://'. $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
                header('Location: ' . $home_url);
            } else { // ユーザ名 / パスワードが不正なためエラーメッセージを設定
                // The username/password are incorrect so set an error message
                $error_msg = 'エラー:正しいユーザ名とパスワードを入力して下さい。';
            }
        } else { // ユーザ名 / パスワードが入力されていないためエラーメッセージを設定
            // The username/password weren't entered so set an error message
            $error_msg = 'エラー:ユーザ名とパスワードが入力されていません。';
        }
    }
}
?>

<html>
<head>
    <title>ミスマッチ：ログイン</title>
    <link rel="stylesheet" type="text/css" href="style.css" />
</head>
<body>
    <h3>ミスマッチ：ログイン </h3>
```

ミスマッチ：ログイン

ログイン
ユーザ名：
skarena
パスワード：

ログイン

login.php

これが新しいログイン用
フォームです。

エラーメッセージを変数に突っ込んで
必要に応じて表示するようにします。
スクリプトの後の方で使います。

ユーザが既に
ログインしているかどうかを調べます。

ユーザがまだログインしていない場合、
ログインデータを「提出」したかどうかを
調べます。

ユーザが打ち込んだデータは、
今度は認証ウィンドウではなく
POSTデータとして飛んできます。

ユーザがログインしたらuser_idと
usernameをクッキーにセットします。

ログインに成功したのでユーザIDとユーザ名をクッキーに設定し、ホームページへリダイレクト

ログインに成功した場合、
ユーザをミスマッチサイトの
ホームページに再度誘導します。

ログインデータに何か間違いが
あった場合、エラーメッセージ
を変数にセットします。

ログイン用スクリプトは、もはや
完全なWebページですから、普通
のHTML要素が全部必要です。

反対側のページへ続く…

```

<?php
// If the cookie is empty, show any error message and the log-in form; otherwise confirm the log-in
if (empty($_COOKIE['user_id'])) {
    echo '<p class="error">' . $error_msg . '</p>';
}
?>

<form method="post" action=<?php echo $_SERVER['PHP_SELF']; ?>>
<fieldset>
<legend>ログイン</legend>
<label for="username">ユーザ名:</label>
<input type="text" id="username" name="username"
       value=<?php if (!empty($user_username)) echo $user_username; ?>><br />
<label for="password">パスワード:</label>
<input type="password" id="password" name="password" />
</fieldset>
<input type="submit" value="ログイン" name="submit" />
</form>

<?php
} | この波カッコより前にあるすべてのものは、
else { | ずっと最初のif節の一部です。
    // Confirm the successful log in
    echo('<p class="login">' . $_COOKIE['username'] . 'としてログイン中です.</p>');
}
?>

</body>
</html> | HTMLコードを終了させ、ログイン用
          | Webページの完成です。

```

クッキーが空の場合、エラーメッセージとログインフォームを表示、そうでなければログイン成功

ユーザがこの時点でもうログインしていない場合、先へ進んでエラーメッセージを表示します。

この2つのフォームフィールドはログイン用にユーザ名とパスワードを入力させるために使います。

この時点でユーザがログインしている場合、単にそのように伝えます。

素朴な疑問に答えます

 何でユーザIDとユーザ名の両方ともクッキーにぶち込む必要があるん?

 どちらの情報でもミスマッチサイトのユーザデータベースでユーザを一意に識別できますから、現在のユーザを覚えておくという意味ではどちらか一方を使えば十分です。ただし、user_idはデータベースのユーザを参照するという点に関して優れています(効率的です)。なぜならそれが数値であって主キーだからです。一方、user_idは微妙に暗号的とも言うか、ユーザに関する情報は何も持っていない。ですからユーザ名は、ユーザに自分がログインしていることを知らせる際に、ユーザ名をページ上に表示するといったことができて便利です。複数の人間が同じパソコンを共同で使っていると

いった状況も考えられますので、単にログイン状態であることを知らせるだけでなく、誰がログインしているのかを知らせるのは重要なことです。

 ほなログインデータの一部としてクッキーにパスワードをぶち込まないのは何でなん?

 パスワードは最初にユーザが誰だと言っているのかを検証するという目的のみ重要です。ログイン処理の一部として一度だけパスワードを検証してしまえば、それをずっと持ち歩く理由はありません。一方でパスワードというのは非常にデリケートなデータなので、一時的に格納するといったことは可能な限り避けるのが得策です。

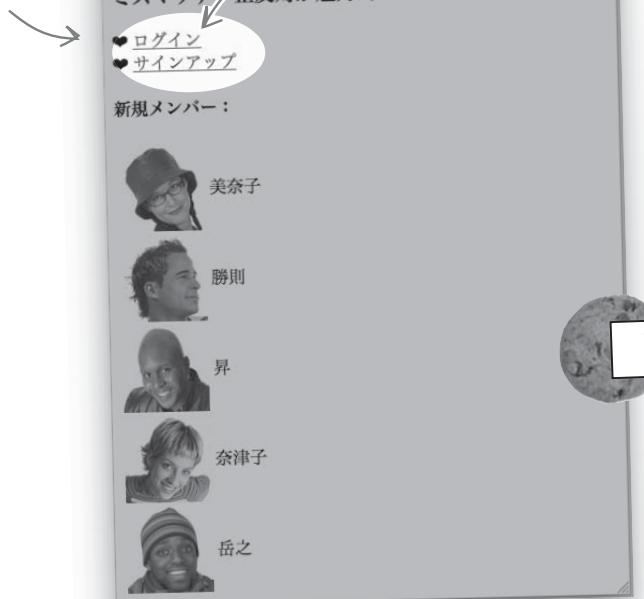
 ログイン用スクリプトのフォーム内のif文の内側にあるんやけど、それでホンマに大丈夫なん?

 大丈夫です。実際、このログイン用スクリプトの場合と同様に、HTMLコードでPHPコードが「分断される」というのは極めてよくあります。PHPコードのセクションを?>で閉じたからといって、コードの構造が閉じたことにはなりません。別のセクションを<?phpで開けば、プログラム構造は残りの部分から復活します。ログイン用スクリプトにおいては、HTMLフォームは最初のif分岐に含まれていますが、else分岐はフォームコードの後を担当します。PHPコードをHTMLコードでこのように分断すると、echo文を注意深く使ってフォームを作り管理しなければなりません。

ミスマッチサイトアプリケーション内を案内する

新しいログイン用スクリプトはミスマッチサイトアプリケーションのフローを変えました。このため簡単なメニューが必要になり、これをホームページ(index.php)上に追加してあります。このメニューはアプリケーション上の異なる主要部分へアクセスを可能にしてくれるという重要な意味を持ちます。現状では主要部分は2つでプロフィール参照用ページとプロフィール編集用ページです。ユーザのログイン状態によっては、ログイン、サインアップ、およびログアウトする機能も必要です。メニューはユーザのログイン状態に応じて変化するというのは重要な意味を持つことです。またメニューこそがアプリケーションに真のパワーと使いやすさを与えてくれるものなのです。

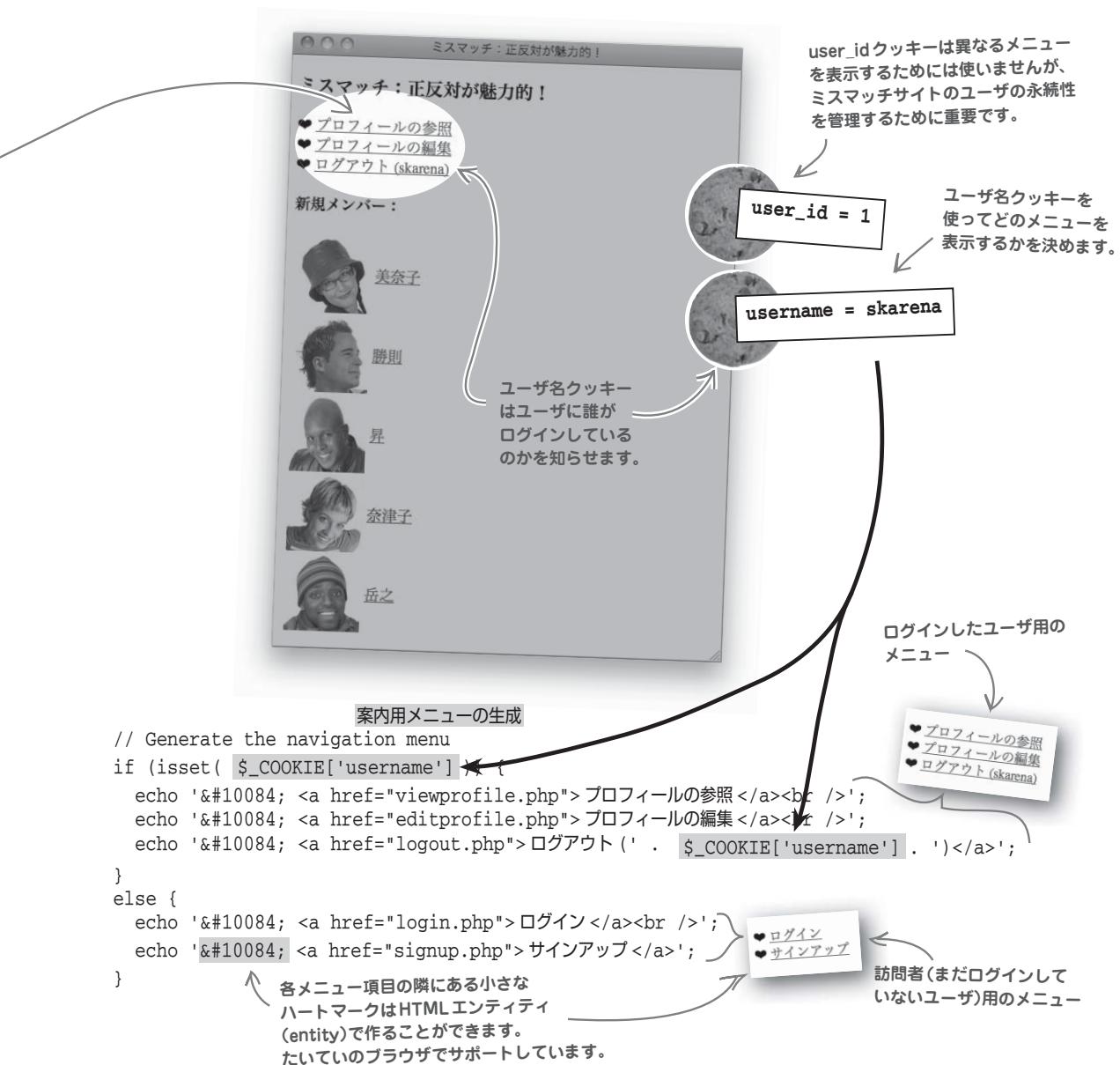
このメニューはユーザがログインしていないときに現れます。ユーザはログインするかまたはサインアップすることができます。

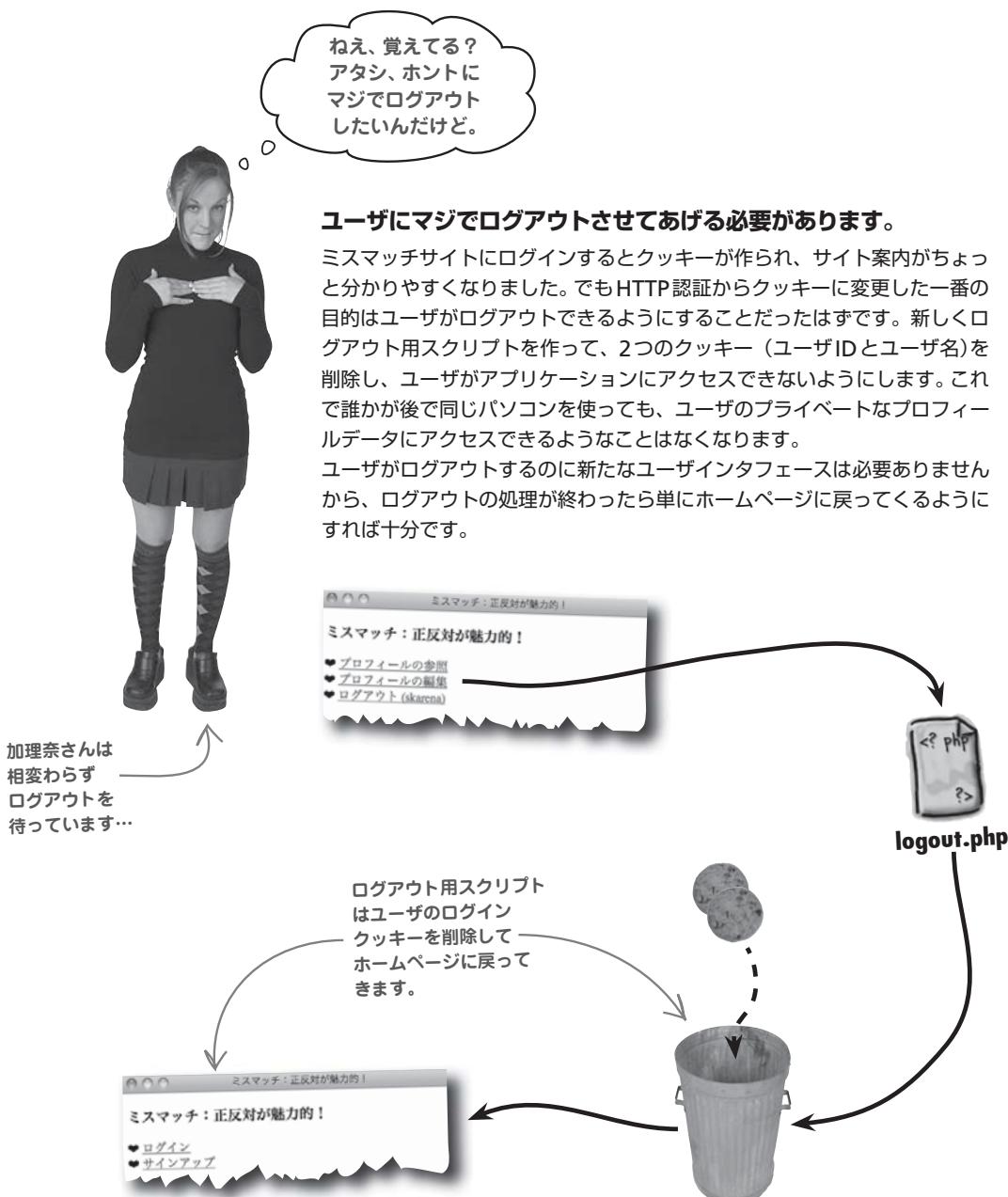


ユーザ名クッキーが設定されたかどうかで異なるメニューが現れます。

index.phpスクリプトはユーザ名クッキーが見つからない場合には制限されたメニューを表示するということを知っています。

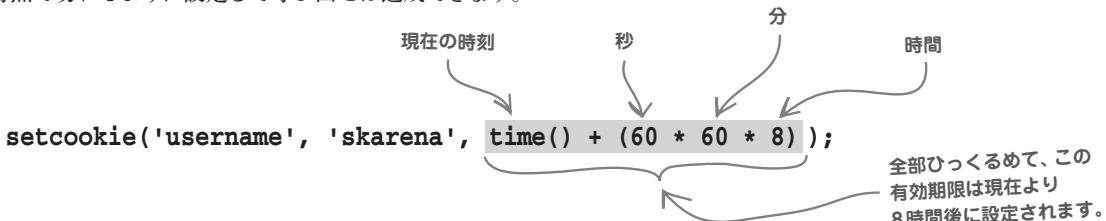
メニューは index.php の PHP コードが作り出し、このコードは \$_COOKIE スーパーグローバルを使って、ユーザ名クッキーを探し、ユーザがログインしているかどうかを判断します。ユーザ ID クッキーを使っても同じことはできますが、ユーザ名はメニューに表示されますから、こちらを使った方が理にかなっているでしょう。





ログアウトとはクッキーを削除することを意味します

ユーザがログアウトするには、ユーザの情報を保存していた2つのクッキーを削除しなければなりません。これは`setcookie()`関数を、有効期限がその時点で切れるように設定して呼び出せば達成できます。



このコードは有効期限を8時間後の未来に設定します。つまりクッキーは8時間後に自動的に削除されます。ただし今欲しいのはクッキーを即座に消すことです。このためには有効期限を過去に設定すれば良いのです。どのくらい過去かは大して重要ではありません。適当な時間をみつくろって、例えば1時間とかを、現在の時刻から引いて設定します。

```
setcookie('username', 'skarena', time() - 3600);
```

60秒×60分=3600秒で、
1時間過去になります。

クッキーを消す
には、有効期限を
過去に設定する
だけでよいのです。



エクササイズ

以下に示すミスマッチサイトのログアウト用スクリプトには、若干抜けているコードがあります。抜けているコードを補って下さい。まずログイン用クッキーを削除してから、ログアウトページからホームページに遷移させることに注意して下さい。

```
<?php
    // If the user is logged in, delete the cookie to log them out
    if (.....) {
        // Delete the user ID and username cookies by setting their expirations to an hour ago (3600)
        .....
        .....
    }
    // Redirect to the home page ホームページヘリダイレクト
    $home_url = 'http://'. $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '.....';
    header('Location: ' . $home_url);
?>
```

ユーザがログインしていれば、クッキーを削除しログアウト

有効期限を1時間(3600)前に設定して、
特定のユーザIDとユーザ名のクッキーを削除



エクササイズ の答え

```
<?php  
    // If the user is logged in, delete the cookie to log them out  
    if (isset($_COOKIE['user_id'])) { ← ログインしているユーザだけ  
        // Delete the user ID and username cookies by setting their expirations to an hour ago (3600)  
        setcookie('user_id', '', time() - 3600); ... ← ログアウトします。  
        setcookie('username', '', time() - 3600); ... ← それぞれのクッキーを  
    } ← 1時間過去に設定して、  
        // Redirect to the home page  
        $home_url = 'http://'. $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php'; ← ミスマッチサイトの  
        header('Location: ' . $home_url); ← ホームページに遷移させ  
    ?> ← システムに削除させます。ます。絶対アドレスの  
          URLで構成します。 )  
          ロケーションヘッダは、ブラウザが  
          別のページに遷移してから使われます。
```

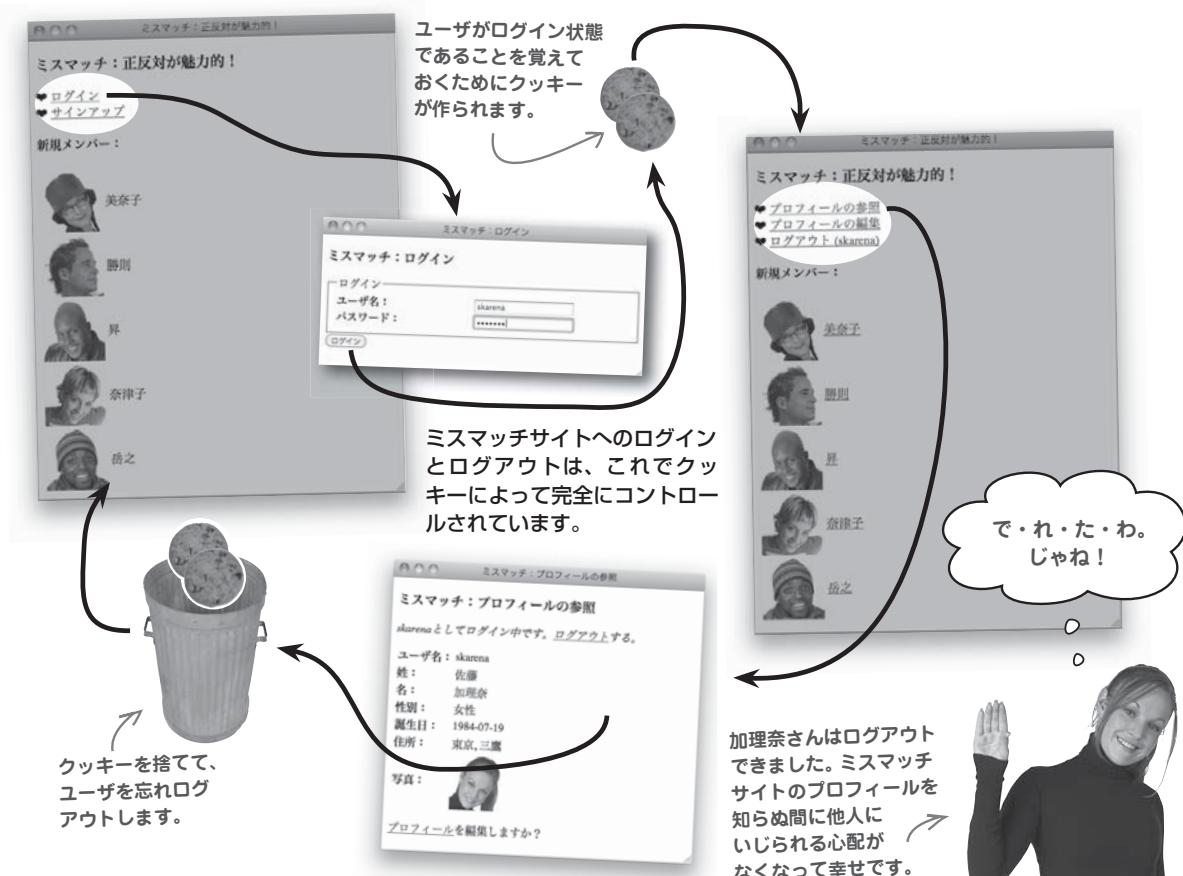


試運転

クッキーを使ってミスマッチサイトにログアウトの機能を追加する。

ミスマッチサイトのスクリプトを修正して下さい。クッキーを使って、ユーザがログインもログアウトもできるようにします。(またはスクリプトをWebサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードして下さい。)クッキーに関する修正はindex.phpとeditprofile.phpとviewprofile.phpに対して行います。2番目と3番目のスクリプトに対する修正は、ほんのちょっとで、主な変更は、user_id変数とusername変数を参照している箇所を、代わりに\$_COOKIEスーパーグローバルを参照するようにするだけです。

できあがったスクリプトをWebサーバにアップロードしたら、ミスマッチサイトのメインページ(index.php)をWebブラウザで開きます。案内用のメニューに注目して下さい。「ログイン」リンクをクリックしてログインします。ログイン用スクリプトによりメインページにどのように戻ってきたかに注目して下さい。ログイン状態が変わったことによりメニューが変わっているはずです。では「ログアウト」をクリックしてクッキーをぶっ壊し、ログアウトしてみます。



素朴な疑問に答えます

Q：つまり単にクッキーの削除というのが、ログアウトでやるべきすべてでしょうか？

A：そうです。クッキーはミスマッチサイトにおけるすべてのログイン情報（ユーザIDとユーザ名）を格納する任務を持っています。これを消せばログアウトは完了です。

Q：クッキーを削除するのに1時間過去に設定するのは何故ですか？1時間に何か
特別な意味があるのでしょうか？

A：ありません。クッキーは有効期限が切れれば、ブラウザにより自動的に削除されます。
従ってクッキーを削除したければ有効期限を過去のいつに設定しても構いません。

1時間（3600秒）というのは、適当にとっただけの時間で、クッキーを削除しようとしていることを一貫して表現しているだけです。

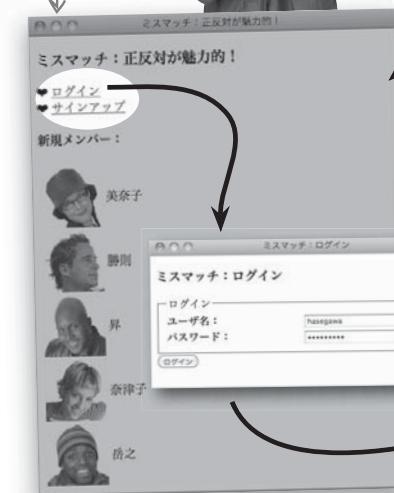


ミスマッチサイトのユーザ長谷川さんです。ハイキングとピアスとハワード・ Stern (Howard Stern)[†]が好きです。ブラウザにクッキーを拒絶されたためログインできなくて困っています。

あちゃー。ブラウザに
クッキーを拒否されちゃって
ログインできないや。どう
すればいいのかな?

[†] 訳注: 超人気ラジオ番組の
パーソナリティ。

ログインへの
道はここから
始まります。



クッキーが拒否されたので、ログイン用
スクリプトは不成功に終わり、ユーザは
ログインすることなくホームページに
戻ってきます。



クライアント
Webブラウザ

ブラウザがクッキーを拒否する
ため、ログイン用スクリプトは
クッキーをセットすることができ
ません。

Webサーバ

サーバはユーザIDと
ユーザ名のクッキーを
ブラウザにセットしよう
とします。

誰が長谷川さんの面倒を
見るの? たいていの人はクッキー
を受け付けないんじゃないの?

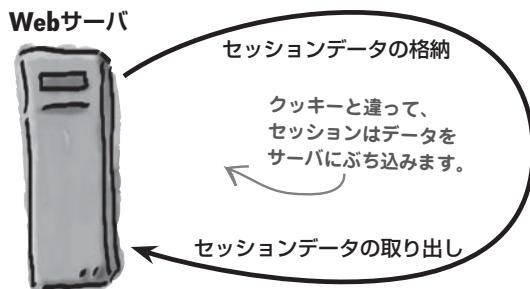
たいていは大丈夫です。ただし Web アプリケーションはできるだけ多くの
人が使えるようにするべきです。

クッキーを信用していない人というのもいるので、そういう人はクッキーを拒絶するよう
なセキュリティ設定を選んでいます。この点を理解すれば、クッキーを使うことのでき
ないユーザがログインする方法を考えるというのは意味のあることです。しかし他にも方法
があります。ログインデータを突っ込む先として、クライアントではなくサーバを使うと
いうもう1つのオプションがあります。スクリプトはサーバ上で動作しているのですから、
ログインデータも同様にサーバに突っ込むというのは理にかなっています。



セッションはクライアントに依存しません

クッキーは小さいけど強力なヤツなのですが限界があります。その限界というのはブラウザ依存ためサーバ側でコントロールできないものなのです。しかしブラウザに依存してはならないとしたら?データをサーバに直接ぶち込めるとしたらどうでしょう?セッションなら正にこれをやってくれます。個人の情報をちょうどクッキーと同様にぶち込むことができます。しかもデータはクライアントではなくサーバにぶち込まれるのです。このためセッションのデータはクッキーの限界であるブラウザ制限に引っかかることはありません。

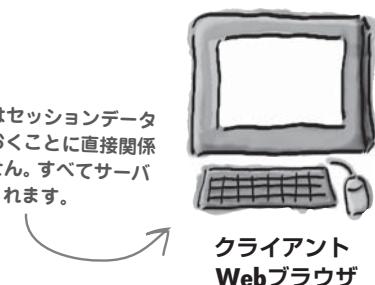


セッションはセッション変数にデータを突っ込みます。これは論理的にはサーバ上にあるクッキーと同じです。PHPコードを使ってセッション変数にデータを入れると、サーバ上に突っ込まれます。この後PHPコードからセッション変数のデータにアクセスすることもできまし、複数のページ(スクリプト)にまたがって永続性を持たせることもできます。クッキーのときと同様、セッション変数はいつでも好きなときに削除することができます。つまりログアウトの機能は、セッションをベースとしたコードでも引き続き提供することができます。



もちろん何かしらのトレードオフはあります。クッキーと違ってセッション変数はデータを突っ込んでおく期間に関して大したコントロールはできません。セッション変数はセッションの終了と同時に自動的に破壊されます。通常これはユーザーがブラウザを閉じるのと同じタイミングで起こります。つまりセッション変数はブラウザ上に格納されているわけではないのですが、ブラウザの影響を間接的に受けるのです。ブラウザのセッションが終了したら削除されるからです。

セッションを使えばちょっとしたデータをサーバに永続的にぶつ込むことができます。
クライアントとは独立に、です。



セッションのデータはサーバにぶち込まれるので、クッキーに突っ込まれたデータに比べて、セキュリティも信頼性も上です。

ユーザがセッションデータをブラウザ側から自分で削除することはできません。クッキーではできてしまうので問題になるかもしれません。
セッション変数に結び付けられた有効期限というものはありません。セッションが終了したら自動的に削除されるからです。

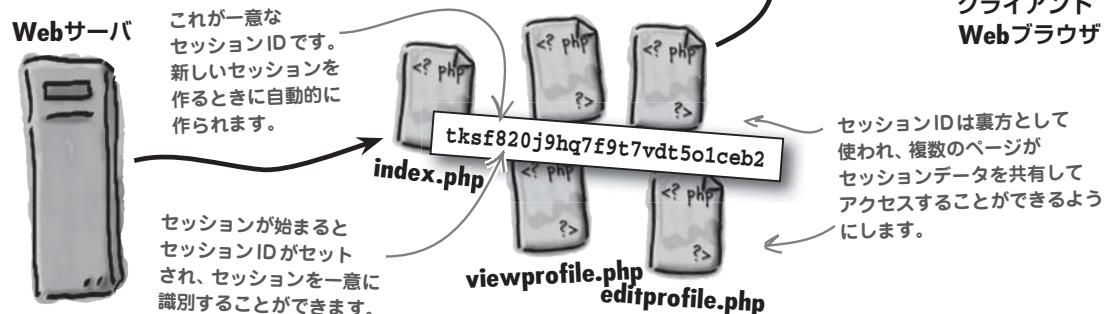
セッションの命と時間

セッションがセッションと呼ばれるには理由があります。極めて明確な開始と終了があるのです。セッションに結びついたデータはセッションの寿命と共に生き、そして死にます。これはPHPコードでコントロールすることができます。セッションの命をコントロールできない唯一の状況は、ユーザがブラウザを閉じたときです。これによりセッションは終了してしまいますから、好むと好まざるとに関わらずここで殺されてしまいます。

準備ができたら、PHP関数session_start()を呼び出してセッション開始を宣言しなければなりません。

`session_start();` PHP 関数でセッションを開始します。

session_start() 関数を呼び出しても何もデータはセットされません。この関数の仕事はセッションを開始し、走らせることだけです。セッションはまず一意なセッション識別子で初期化され識別することができますが、これについてはサーバ側が気にする必要はありません。このIDは Web ブラウザ側で複数のページを結びつけるために使います。



セッションIDはセッションが閉じられるまでは破壊されません。これが起こるのは、ブラウザを閉じた場合か、またはsession_destroy()関数を呼び出した場合です。

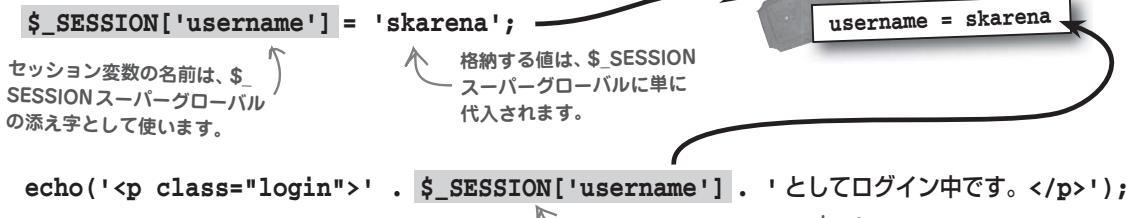
`session_destroy();` この PHP 関数でセッションが終了します。

session_destroy() 関数
はセッションを閉じます。

この関数を呼び出してセッションを閉じた場合、自分で作ったセッション変数が自動的に破壊されることはありません。セッションがどのようにデータを格納するのかをもう少しそう見て、何故そうなるのかを見てみましょう。

セッションデータを追跡する

セッションに関してイケてるところは、使い方がクッキーと非常によく似ていることです。セッションを `session_start()` を呼び出して開始したら、セッション変数をセットすることができるようになります。例えばミスマッチサイトのログインデータは `$_SESSION` スーパーグローバルを使えばいいのです。



クッキーと異なり、セッション変数には値をセットするための特別な関数は必要ありません。単に `$_SESSION` スーパーグローバルに値を代入するだけです。セッション変数名を配列の添え字に使うことさえ忘れないでください。

セッション変数の削除はどうでしょうか? `session_destroy()` 関数を使ってセッションを破壊しても、実際にはセッション変数が破壊されるわけではありません。このためセッション変数の削除は自分でやらなければなりません。(ログアウトのように!) ユーザがブラウザを閉じる前に殺したい場合は正にそうする必要があります。すばやく効率的にすべてのセッション変数を破壊する方法は、`$_SESSION` スーパーグローバルに空の配列をセットすることです。

`$_SESSION = array();`

このコードで現在のセッションのセッション変数を皆殺しにできます。

ただし、まだ全然終わっていません。セッションは裏で勝手にクッキーを使っているかもしれないのです。ブラウザがクッキーを許可している場合、セッションはクッキーを一時的なセッションIDの格納領域として使っている可能性があります。そこでPHPコードを使って完全にセッションを閉じるには、ブラウザ上でセッションIDを突っ込むために勝手に作られたかもしれないすべてのクッキーを削除しなければなりません。他のクッキーと同様に、このようなクッキーも有効期限を何か過去にセットして破壊することができます。そうすると知っておかなければならないのは、クッキーの名前だけです。これは `session_name()` 関数を使えば見つけることができます。

```
if (isset($_COOKIE[ session_name() ])) {
    setcookie(session_name(), '', time() - 3600);
}
```

まずセッションのクッキーが実際に存在するかをチェックします。

セッションのクッキーを、有効期限を1時間過去にセットすることで破壊します。

セッション変数が作られサーバ上に保持されます。

`username = skarena`

セッション変数にアクセスするには、単に `$_SESSION` スーパーグローバルとセッション変数名とを使うだけです。

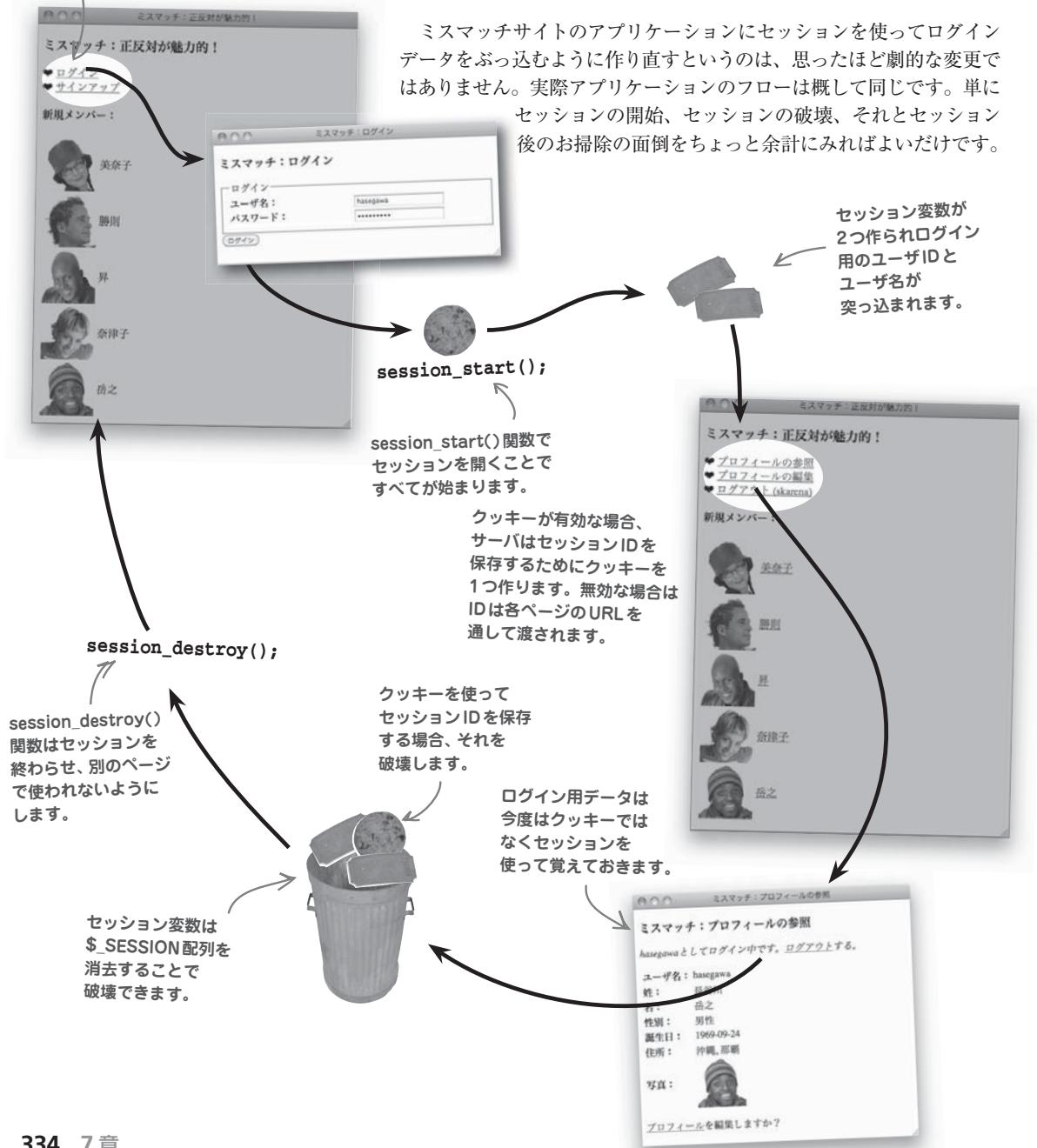
セッション変数はセッションが破壊されても自動的に削除されることはありません。

セッションがクッキーを使ってセッションIDを覚えていたとすると、そのIDはセッションの名前を持つクッキーに保存されています。

`PHPSESSID = tse820j...;`

ここから始まります！

ミスマッチサイトをセッションで刷新する



セッションを使ってログアウトする

ミスマッチサイトからユーザがログアウトするには、純粹にクッキーのみを使ったときのバージョンに比べ、もう少し仕事が必要です。以下のステップに従わないとセッションを使っているミスマッチサイトからユーザをちゃんとログアウトさせることができません。

- ① セッション変数を削除する。**
- ② セッションが作ったクッキーがあるかどうかをチェックし、もしあれば削除する。**
- ③ セッションを破壊する。**
- ④ ユーザをホームページへ遷移させる。**

セッションのクッキーが使われたのかどうかはチェックしてみないと分かりません。

その通りです。これはあってもなくても構いません。ユーザがログアウトするためにどうしても必要ということはありませんが、まあ便利なので入れておきます。

自分で考えてみよう

ログアウト用スクリプトをオーバーホールして純粋なクッキーの代わりにセッションを使ってログインの永続性を管理しようとしています。足りない部分のコードを埋めて、ログアウト用スクリプトを「セッション化」して下さい。次にログアウトの上記4ステップが、コードのどの部分に対応するか書き込んで下さい。

```
<?php
    // If the user is logged in, delete the session vars to log them out
    session_start();
    if (...) {
        // Delete the session vars by clearing the $_SESSION array
        .....
        // Delete the session cookie by setting its expiration to an hour ago (3600)
        if (isset($_COOKIE[session_name()])) {
            .....
        }
        // Destroy the session セッションを破壊
        .....
    }
    // Redirect to the home page ホームページヘリダイレクト
    $home_url = 'http://'. $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
    header('Location: ' . $home_url);
?>
```

自分で考えてみよう の答え

ログアウト用スクリプトをオーバーホールして純粋なクッキーの代わりにセッションを使ってログインの永続性を管理しようとしています。足りない部分のコードを埋めて、ログアウト用スクリプトを「セッション化」して下さい。次にログアウトの上記4ステップが、コードのどの部分に対応するか書き込んで下さい。

- ① セッション変数を削除する。
- ② セッションが作ったクッキーがあるかどうかをチェックし、もしあれば削除する。
- ③ セッションを破壊する。
- ④ ユーザをホームページへ遷移させる。

```

ログアウトのときでも、まずはセッションを
開始してセッション変数にアクセスできる
ようにする必要があります。

<?php
    // If the user is logged in, delete the session vars to log them out
    session_start();
    if (isset($_SESSION['user_id'])) { ← これでクッキーの代わりにセッション
        // Delete the session vars by clearing the $_SESSION array
        $_SESSION = array(); ← 变数を使ってログイン状態をチェック
        // Delete the session cookie by setting its expiration to an hour ago (3600)
        if (isset($_COOKIE[session_name()])) {
            setcookie(session_name(), "", time() - 3600); ② ← セッションのクッキーが存在したら、有効期限を
        }                                         1 時間前にセットして削除します。
        // Destroy the session
        session_destroy(); ← 組み込みのsession_destroy()
        // 関数を呼び出して、セッションを
    }                                         ③ 破壊します。
    // Redirect to the home page
    $home_url = 'http://'. $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
    header('Location: ' . $home_url); ④
?>

```

セッション変数をまっさらになると
するには、\$_SESSION
スーパーグローバルに空の
配列を代入すれば良いのです。

何をどう変える？

クッキーからセッションに移行するとインパクトがあるのはログアウト用スクリプトだけではありません。左側にあるミスマッチサイトのアプリケーションの各部分に対して、右側の説明、どのようにセッションに対応させて変更する必要があるか、とをマッチさせて下さい。



変更なし。このスクリプトにはログインの永続性との直接的な依存関係はありません。

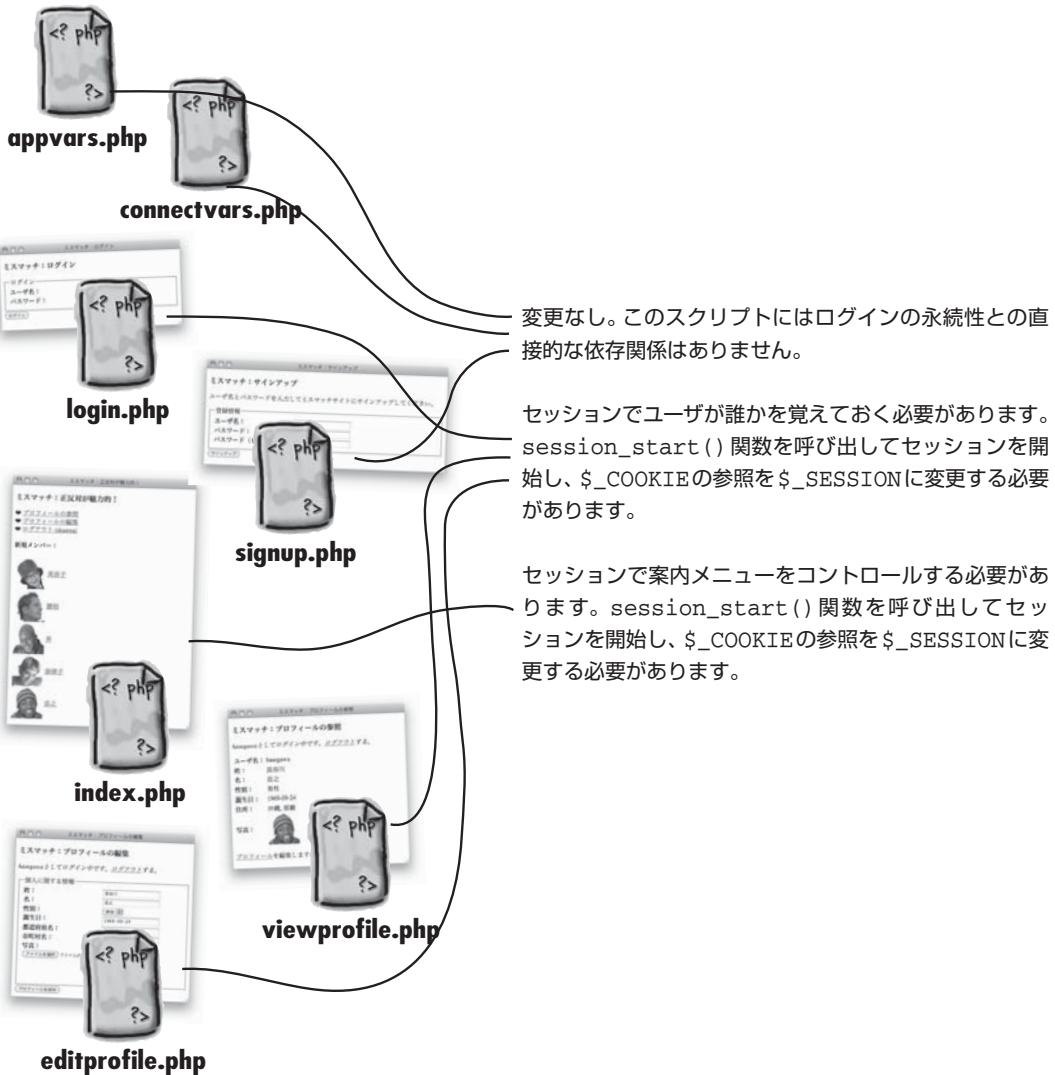
セッションでユーザが誰かを覚えておく必要があります。`session_start()` 関数を呼び出してセッションを開始し、`$_COOKIE` の参照を `$_SESSION` に変更する必要があります。

セッションで案内メニューをコントロールする必要があります。`session_start()` 関数を呼び出してセッションを開始し、`$_COOKIE` の参照を `$_SESSION` に変更する必要があります。



何をどう変える? の答え

クッキーからセッションに移行するとインパクトがあるのはログアウト用スクリプトだけではありません。左側にあるミスマッチサイトのアプリケーションの各部分に対して、右側の説明、どのようにセッションに対応させて変更する必要があるか、とをマッチさせて下さい。





重要ポイント

- HTTP認証は個々のページへのアクセスを手軽に制限できますが、ユーザがページへのアクセスを終了する際に「ログアウト」するための良い方法がありません。
- クッキーはクライアント(Webブラウザ)にちょっとしたデータ、例えばユーザのログインデータなどを突っ込むことができます。
- すべてのクッキーには有効期限があります。それは遠い未来でも構いませんし、ブラウザセッションの終了時点でも構いません。
- クッキーを削除するには、有効期限を過去に設定すればよいのです。
- セッションはクッキーと同様な格納領域を提供してくれますが、サーバに置かれます。このためクッキーの無効化といった、ブラウザ側の制約の対象ではありません。
- セッション変数は限られた寿命しか持っています。セッションが終了すると(例えばブラウザを閉じると)破壊されます。

素朴な疑問に答えます

Q: session_start()関数は様々な状況で、セッションの開始後にも呼び出されています。それぞれのsession_start()関数の呼び出しに対応して、複数のセッションが作られるのでしょうか?

A:違います。session_start()関数は単に新しいセッションを開始するだけではありません。既存セッションに入り込んでいくこともできます。ですからスクリプトがsession_start()を呼び出すと、関数は既にセッションが存在するかどうかをセッションIDの有無でチェックします。まだセッションがなければ、新しいセッションIDを作ることで、新しいセッションを作ります。これより先、同じアプリケーションからのsession_start()を呼び出すと、既存のセッションと認識されて別のセッションを作るのではなく、そのセッションIDを使います。

Q:ではセッションIDというのは、どのように保存されるのですか?場合によってはクッキーを使うということですか?

A:その通りです。セッションのデータはサーバ上に保存されるため、よりセキュリティが高く、かつブラウザのコントロールを受けないという御利益があります。ただ相変わらずセッションのデータについてスクリプトが知るためのメカニズムが必要だということに変わりはありません。

これがセッションIDの目的で、セッションと一緒に結びついたデータを一意に識別します。このIDは何らかの形でクライアントに留まり、複数のページを同一セッションとみなさなければなりません。セッションIDに永続性を持たせる1つの方法としてクッキーを使うという手があります。つまりIDをクッキーに突っ込んでしまい、そのセッションとスクリプトとを連付けて使うのです。

Q:結局のところセッションがクッキーに依存しているのなら、クッキーの代わりにセッションを使うことによるメリットは何なのでしょうか?

A:セッションはクッキーに完全に依存しているわけではありません。ここは重要なことで理解しておいて欲しいのですが、クッキーはセッションIDを複数のスクリプトにまたがって保存する際の最適化として、であって必須ではないということです。クッキーが無効となっている場合、セッションIDはスクリプトからスクリプトへURLの通して渡されます。これはGETリクエストでデータを渡したのと同様な方法です。つまりセッションはクッキーがなくても全然平気でちゃんと動きます。クッキーが無効化されているときにセッションがどのように動作するかに関する仕様は、設定ファイルphp.iniでコントロールされています。このファイルはWebサーバ上にあって、session.use_cookies, session.use_only_cookiesとsession.use_trans_sidで設定します。

Q:まだよく分からぬのですが、セッションはクッキーを使うことがあるのに、セッションがクッキーよりもメリットがあるとは思えません。どうでしょうか?

A:セッションは状況によってはクッキーに勝ることがあります。それはクッキーとの関係が必ずしも必要ではありません。セッションの明らかな利点はクライアントではなくサーバにデータが保存されることです。これによりセキュリティも信頼性も高まります。ですから機密性の高いデータを永続的に保存する必要がある場合、セッション変数の方がクッキーに比べて高いセキュリティを確保してくれます。セッションはまたクッキーに比べて、より多くの情報をぶち込む能力を備えています。ですからクッキーを使えるかどうかに関わらずセッションを使うことによる明らかな利点はあると言えます。

ミスマッチサイトに関して言うと、セッションはログインのデータをサーバ側に突っ込んでおくための簡単な手段を提供してくれます。クッキーを有効にしているユーザには、セッションでセキュリティと信頼性の改善を提供できます。ここではクッキーは使っていますが最適化としてです。逆にクッキーを無効としているユーザに対しては、セッションはセッションIDをURLを通して渡すことにより、クッキーは全くなしでもちゃんと動くことができます。

セッションへの進化を完了させる

セッションを使うと、ミスマッチサイトの異なる部分に影響が出て、異なることをやらなければなりません。しかし、スクリプトをクッキーからセッションに進化させるには最終的に同様な変更をすることになります。その1つとして、すべてのスクリプトは `session_start()` 関数を呼び出す必要があります、これによってセッションに組み込まれます。その先については、すべての変更は `$_COOKIE` スーパーグローバルから `$_SESSION` スーパーグローバルへの移行で、ここがセッション変数を入れておくところです。

```
<?php
    session_start();
?>
```

セッション強化版スクリプトは、すべて `session_start()` 関数を呼び出して、セッションを起こして走らせることから始まります。

```
// If the user isn't logged in, try to log them in
if (!isset($_SESSION['user_id'])) {
    if (isset($_POST['submit'])) {
        // Connect to the database
        $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
        // Grab the user-entered log-in data
        $user_username = mysqli_real_escape_string($dbc, trim($_POST['username']));
        $user_password = mysqli_real_escape_string($dbc, trim($_POST['password']));
        if (!empty($user_username) && !empty($user_password)) {
            // Look up the username and password in the database
            $query = "SELECT user_id, username FROM mismatch_user WHERE username = '$user_username' AND ".
                "password = SHA('$user_password')";
            $data = mysqli_query($dbc, $query);
            if (mysqli_num_rows($data) == 1) {
                // The log-in is OK so set the user ID and username session vars, and redirect to the home page
                $row = mysqli_fetch_array($data);
                $_SESSION['user_id'] = $row['user_id'];
                $_SESSION['username'] = $row['username'];
                $home_url = 'http://'. $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
                header('Location: ' . $home_url);
            } else {
                // The username/password are incorrect so set an error message
                $error_msg = 'エラー: 正しいユーザ名とパスワードを入力して下さい。';
            }
        }
    }
}
```

ログイン用スクリプトはセッションを使ってユーザIDとユーザ名を覚え、ログインに永続性を持たせます。これは `$_COOKIE` スーパーグローバルの代わりに `$_SESSION` スーパーグローバルを使うことで達成できます。



login.php

```
// Generate the navigation menu
if (isset($_SESSION['username'])) { 案内用メニューの生成
    echo '&#10084; <a href="viewprofile.php">プロファイルの参照</a><br />';
    echo '&#10084; <a href="editprofile.php">プロファイルの編集</a><br />';
    echo '&#10084; <a href="logout.php">ログアウト(' . $_SESSION['username'] . ')</a>';
} else {
    echo '&#10084; <a href="login.php">ログイン</a><br />';
    echo '&#10084; <a href="signup.php">サインアップ</a>';
}
...
// Loop through the array of user data, formatting it as HTML
echo '<h4>新規メンバー:</h4>';
echo '<table>';
while ($row = mysqli_fetch_array($data)) {
    ...
    if (isset($_SESSION['user_id'])) {
        echo '<td><a href="viewprofile.php?user_id=' . $row['user_id'] . '">' . $row['first_name'] . '</a></td></tr>';
    } else {
        echo '<td>' . $row['first_name'] . '</td></tr>';
    }
}
echo '</table>';
```

ミスマッチサイトのホームページは
\$_COOKIEの代わりに\$_SESSION
スーパーグローバルを使ってログイン
用データにアクセスし、メニューを
生成して、「新規メンバー」プロファイル
へのリンクを作るかどうかを
決めます。



index.php

ログイン用スクリプト
やホームページと
同様、プロフィール
編集用スクリプトも
\$_COOKIEの代わりに
\$_SESSIONを使って
ログイン用データに
アクセスします。

```
// Make sure the user is logged in before going any further.
if (!isset($_SESSION['user_id'])) {
    echo '<p class="login"><a href="login.php">ログイン</a>後のこのページにアクセスすることができます。</p>';
    exit();
} else {
    echo('<p class="login">(' . $_SESSION['username'] . ')としてログイン中です。<a href="logout.php">ログアウト</a>する。</p>');
}
...
if (!empty($first_name) && !empty($last_name) && !empty($gender) && !empty($birthdate) &&
    !empty($city) && !empty($state)) {
    // Only set the picture column if there is a new picture 写真が存在する場合に限り picture カラムを設定
    if (!empty($new_picture)) {
        $query = "UPDATE mismatch_user SET first_name = '$first_name', last_name = '$last_name', " .
            "gender = '$gender', birthdate = '$birthdate', city = '$city', state = '$state', " .
            "picture = '$new_picture' WHERE user_id = '" . $_SESSION['user_id'] . "'";
```



editprofile.php

中身を示していませんが、
プロフィール参照用
スクリプトもプロフィール
編集用スクリプトと、
ほとんど同様にセッション
を使います。



viewprofile.php

炉辺歓談



今夜の話題：クッキーとセッション変数が集まって
どちらの記憶力が勝っているかの泥仕合。



クッキー：

話したいことはいっぱいあるんだ。クッキーとセッションについてサーバの周りで正確には何が起こっているのかってことなんだけど…。噂じゃあキミはボクの縄張りに入ってきていてデータを記憶する仕事を奪っているって話だけど、どうなの？

それじゃ全然答えになってないよ。ブラウザというのはデータをぶち込むには完璧な場所で、それは正にボクの仕事だぜ。

うつ、まあ、それは全然違う話だよ。もしユーザがボクを無効にすると決めたのなら、それはユーザがデータをぶち込む必要がないからだろ。

じゃあキミの答えはサーバにデータをぶっ込んでいるっていうこと？ それって便利なの？

分かったよ博士。でもキミ一人で何でもできるんだったら、どうしてたまにボクをつかって、ブラウザ上にキミの小さくて大事なIDを残していくたりするの？



セッション変数：

ちょちょちょ、「奪ってる」て強烈やな。ホンマの話は、サーバのデータをたまに入れとるっちゅうだけやで。

じゃアンタ無効になってたらどないすんねん？

ちゃうわ。ユーザはWebアプリケーションがデータをぶっ込みに来るなんて、たいがい知りもせんのやで。例えばユーザ名みたいなのは見えへんデータや。せやからアンタが無効やったら、みんな何もでけへん取り残されてまうやろ。

そや。ユーザはサーバ上のものは何も無効にはでけんつちゅうこっちゃ。どやイケてるやろ。せやからデータがホンマにぶっ込めるかどうかなんてことはアンタ何も心配せんでええ。

あー、せやなあ。たいがいの人はそのことは知らへん。せやからその話は今せんでもよろしい。楽屋で話そか。大事なんはワシはいつでもそこらにあるから、サーバのデータをすぐにぶっ込めるちゅうこっちゃ。

クッキー：

ちょっと待ってよ。キミがボクを必要とする理由をちゃんと教えてよ！

うん、できるっていうことは知ってるけど、どちらかというと本当は違うと思うな。つまり多分本当のところはボクと同じなんじゃないの。

あ、つまりちっこいヤツらを選んで頼りたいっていうこと。確かにボクはキミに比べると全然ものは入らないかも知れないね。それにクライアント側に住んでるからセキュリティ上ちょっと劣ることも認めるよ。でもそれだって結構ワクワクするだろ！あとキミが夢にまで見るようなことだってボクはできるよ。

ええと、格納領域とかセキュリティについてキミが自慢していること全部について言えることなんだけど…寿命が短いってことなんだ。これはキミに言うべきことじゃないのかもしれないけど、キミの全寿命は1つのブラウザセッション次第だろ。だからキミはそういう名前が付いているんだと思っているけど。

簡単さ。ボクはセッションと一緒に死なないんだよ。有効期限があるんだ。つまりボクは長~い生涯を生き抜くようにセットすることもできるんだ。気まぐれにWebサーフィンを楽しむ人たちが、ブラウザを開いたり閉じたりするようなことがあっても、それより遙かに長い期間を生きていられるんだよ。

問題はあるよ。プログラマがボクの有効期限をすごく短期間にセットしてしまって、そのせいで本来長い時間をかけてやるべき仕事ができないことがあるんだ。つまり、つ…

セッション変数：

わかったがな。認めたるわ。複数ページにまたがって仕事せなあかんときは、アンタにちょっと頼つとるのや。せやけどワシ一人でせなあかんときは、できるんやで。

見てえな。アンタなしでも何も問題ないわ。ワシが思つとるんはアンタ、もちょっとセキュリティあつたらなあ、ちゅうこっちゃ。しかもサイズ制限もあるがな。永続データゆうたら全部が全部小っさいとは限らんのやで。

ホンマかいな？聞かして。

ほならアンタは1つのセッションを超えて生きられるんかいな？何でそんなんできんの？

おお、すごいやないか。まるで不死身ちゅう感じやな。でもワシの思いはただ1つや。セッションを閉じるときに怠けモンのプログラマがサボってワシを消し忘れることがないように、っちゅうこっちゃ…ま、もしあってもブラウザが終了するときにワシを殺してくれよるから問題ないんやけど。

もしもし？起きとんのかいな？アカンわ。有効期限切れとる。



試運転

ミスマッチサイトを変更してクッキーの代わりにセッションを使う。

ミスマッチサイトのスクリプトを修正してクッキーの代わりにセッションを使いログインの永続性を持たせます(またはスクリプトをWebサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードして下さい)。セッションに対応した修正によりindex.php, login.php, logout.php, editprofile.php, viewprofile.phpの各スクリプトが変更となります。主な変更は、セッションを開始するのにsession_start関数を呼び出すことと\$_COOKIEスーパークリエイタブルを参照していたところを\$_SESSIONを使うように変更することです。

変更したスクリプトをWebサーバにアップロードしたら、ミスマッチサイトのメインページ(index.php)をWebブラウザで開きます。ログインとログインを試してみて、何もかも以前と(クッキー版と)同様に動くことを確認して下さい。あらかじめクッキーを無効にしていなければ、何も違いは見つからないはずです。それで成功です！

イケてるねえ。
クッキーを無効にしなく
てもログインできるよう
になって素晴らしいよ。

ミスマッチ：正反対が魅力的！

ミスマッチ：ログイン

ミスマッチ：プロフィールの編集

セッションのおかげで、
クッキーを無効にしている
ユーザちゃんとログイン
できて個人のプロフィール
にアクセスできるよう
になりました。

344 7章



要注意

クッキーが無効だとセッションが動かない可能性があります。
サーバ上でPHPの設定が適切になされているか、`php.ini`を確認して下さい。

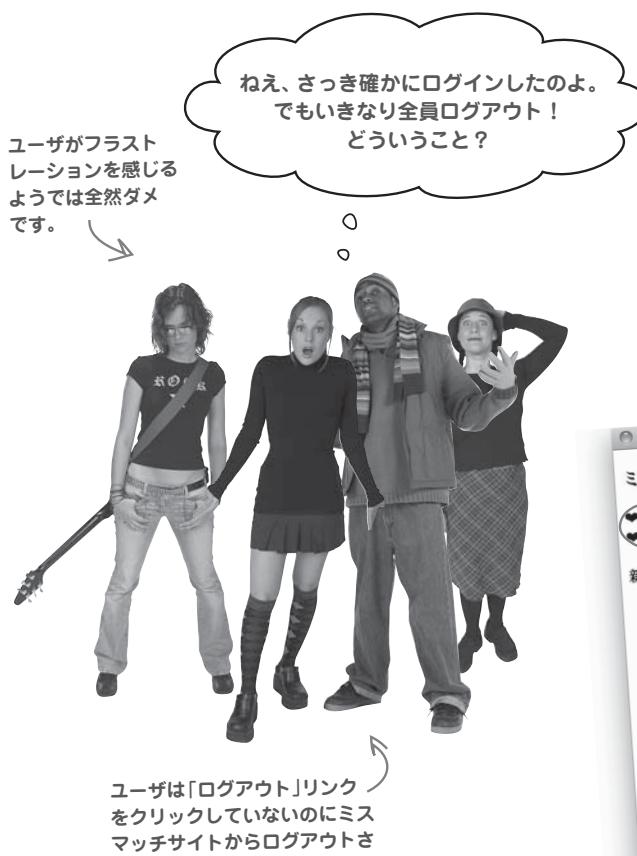
クッキーが無効でもセッションが動くためには、セッションIDを異なるページ間でやりとりするために、別のメカニズムが必要になります。このメカニズムというのはセッションIDを各ページのURLに貼り付けることで、サーバ上にある`php.ini`のセッティング`session.use_trans_id`が1(true)に設定されていれば自動的に働きます。お使いのWebサーバにアクセスしてこのファイルを変更する権限がない場合、自分でセッションIDをURLに貼り付けることもできます。クッキーが無効となっている場合、セッションのURLにセッションIDを貼り付けるコードはこんな感じになります。

```
<a href="viewprofile.php?<?php echo SID; ?>">プロフィールの参照</a>
```

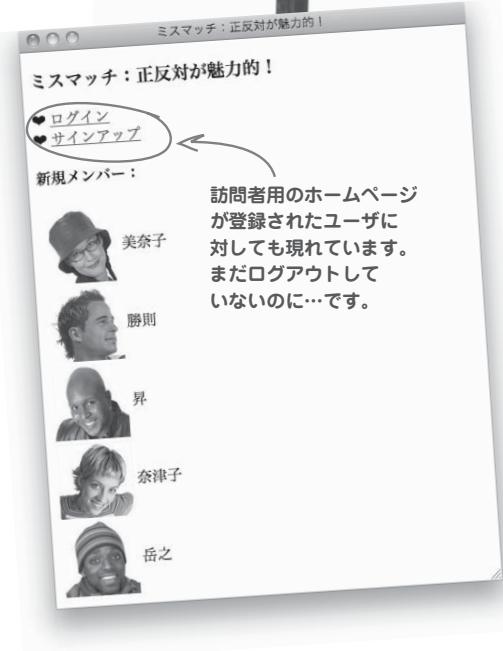
SIDスーパーグローバルがセッションIDを持っています。URLを通してプロフィール参照用ページにこれを渡して、セッションのことを知らせることができます。

ユーザは嬉しく思っていません

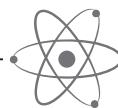
クッキーに勝るちょっとした改良を施したにも関わらず、セッション強化版ミスマッチサイトのアプリケーションは、何か落ち度があるようです。何人かのユーザが「ログアウト」リンクをクリックしていないのにアプリケーションがログアウトしてしまうと言っています。このアプリケーションではちっとも個人用と感じられません…大問題です。



こんなメッセージを
ミスマッチサイト
からユーザに送って
いけません。



訪問者用のホームページ
が登録されたユーザに
対しても現れています。
まだログアウトして
いないのに…です。

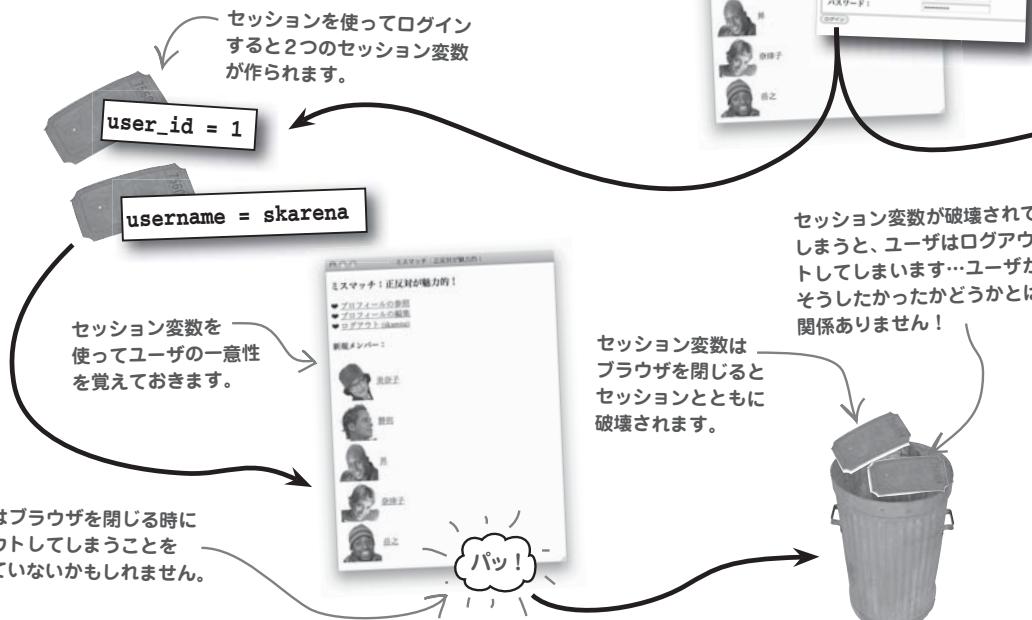


脳力発揮

ユーザが自動的にミスマッチサイトからログアウトして
しまった原因は何だと
思いますか？何か不注意でやってしまったのでしょうか？

セッションは短命です…

ミスマッチサイトが勝手にログアウトしてしまうという問題は、セッションの寿命が限られていることによるものです。覚えていると思いますが、セッションというのはその時点でのブラウザ・インスタンスが続いている間だけしか生きていられません。つまりセッション変数はユーザがブラウザアプリケーションを閉じたときに全部殺されてしまいます。言い換えると、ブラウザを閉じるとユーザがそうしたいかどうかとは無関係にログアウトしてしまうのです。これは不便なだけでなく、ちょっと混乱を来す恐れがあります。なぜならログアウトの機能はもう作ってあるのですから。ユーザは「ログアウト」リンクをクリックしない限りログアウトはしないものと思っているかもしれません。



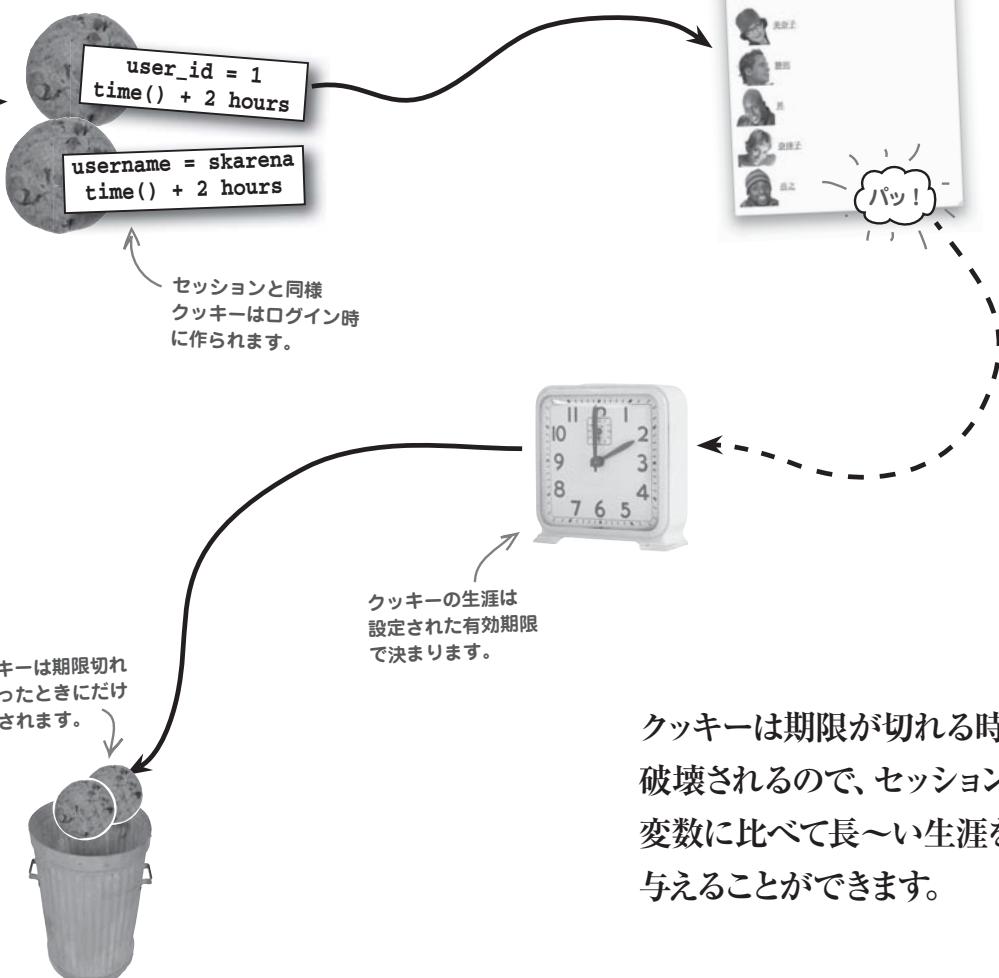
セッションが終わったときにセッションを破壊することができてしまうのですが、逆にセッションはブラウザのインスタンスを超えて生かしておくことはできないのです。つまりセッションはクッキーに比べると記憶領域としては短命なソリューションと言うことができます。なぜならクッキーには有効期限があり何時間、何日間、何ヶ月、あるいは何年といった未来にまで設定できるからです。これはセッションがクッキーより劣っているという意味になるでしょうか？そんなことはありません。しかし確かに問題はあります。セッションを使い、単一のブラウザインスタンスを超えて情報…例えばログインデータ、を覚えておこうとしてもできません。

セッション変数は
ユーザがブラウザを
閉じてセッションが終了
すると破壊されます。

…でもクッキーなら永遠に続きます！

セッション変数と異なり、クッキーの生涯はブラウザのインスタンスとは結びついていません。つまりクッキーはずっとずっと生き続けます。少なくとも有効期限が来るまでは。問題なのは、ユーザがやろうと思えばすべてのクッキーを破壊することができるということです。単にブラウザの設定を変更だけでパソコンに突っ込まれたすべてのクッキーを消すことができます。ですからクッキーの永続性について過度の期待は禁物です。こちらも結局のところ一時的なデータを突っ込むことを目的としているだけなのです。

多分永遠というのはウソです。
ただセッションに比べれば
果てしなく長くすることは
できます。



じゃあセッションとクッキーを両方
使うなんてどうかな？クッキーがあれば
ユーザは長時間ログインしていられるん
じゃないかな？クッキーを有効にしている
ユーザならちゃんと動くと思うんだけど。

超機密データを取り扱っているのでないのであれば、この
場合で言うと、あまりセキュリティ強度のないクッキーで
もセッションを使うということは意味があるでしょう。

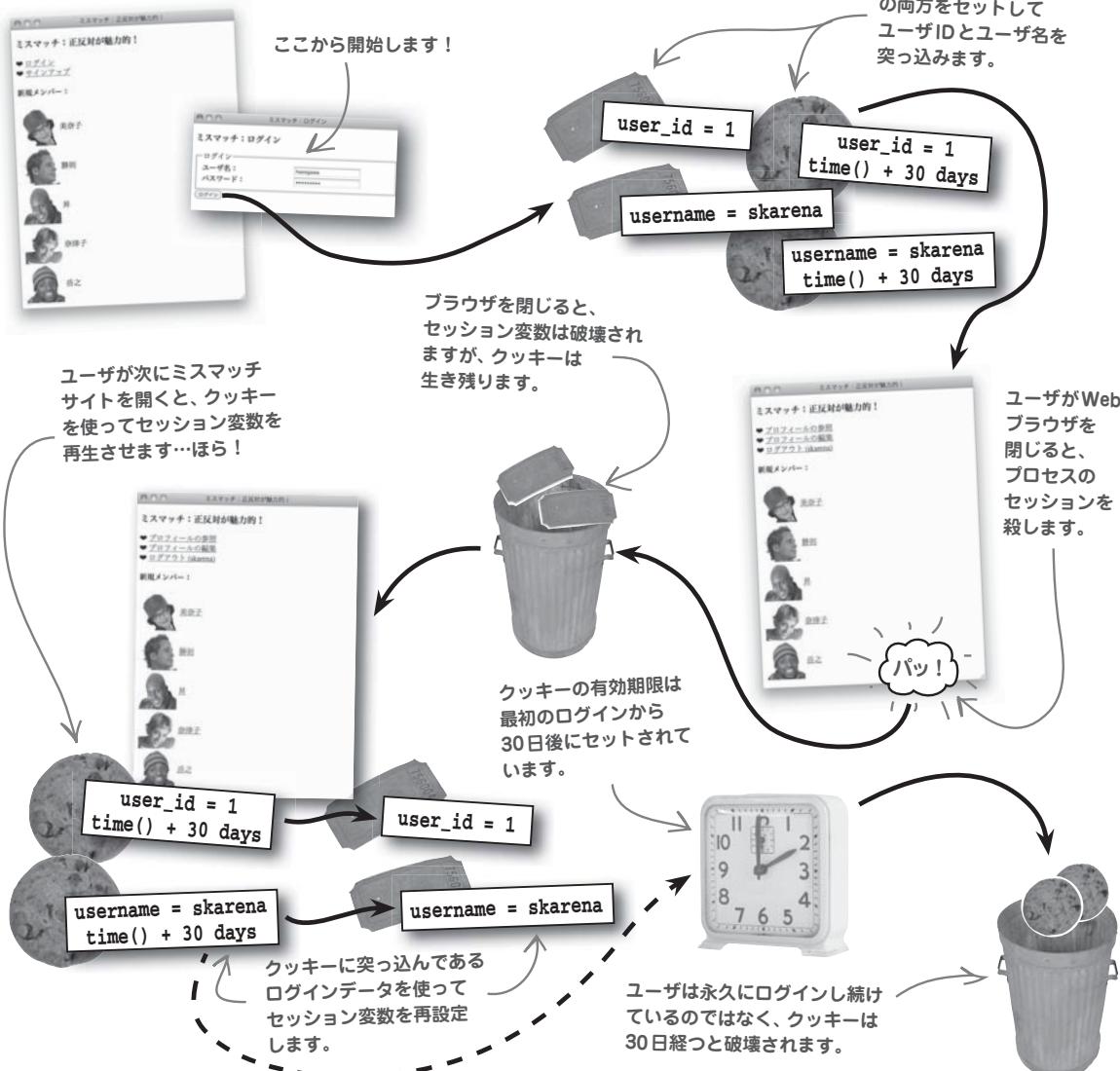
**その通りです。セッションとクッキーの両方にそれぞれ特有な
長所を活かすというのは間違っていません。ミスマッチサイト
のログインはもっと柔軟になります。**

実際、そうすればものすごく便利になります。セッションは短い期間の永
続性に適しています。セッションはより広いサポートを提供でき、ブラウ
ザの制約を受けません。一方でクッキーはログインのデータを長期間に
わたって覚えておくことができます。確かに、すべての人がクッキーによ
る再改良の恩恵に浴することができるわけではありません。しかしほと
んどの人は、それを問題とはしないでしょう。ユーザにとっての使い心地
(ユーザエクスペリエンス)を改良するときはいつでも、大多数のユーザ
を基準にし、他のユーザも切り捨てるがないようにします。それ
ができていれば勝ちです。



セッション+クッキー=すごいログイン永続性

究極のログイン永続性を達成するには、もっとクリエイティブになってこの章で学んだすべてのことを組み合わせる必要があります。セッションとクッキーの両方の長所を活用するのです。そのためには、ミスマッチサイトのアプリケーションを再構築し、ユーザのログイン永続性を短期間と長期間の両方について優れたものにする必要があります。



素朴な疑問 に答えます

Q : 結局のところ短期間か長期間かの理由でセッションかクッキーかのどちらを選ぶかということですか？

A : 違います。この違いはミスマッチサイトのアプリケーションを設計する指針として、たまたま役立つものになっています。しかし、アプリケーションが変われば状況も変わります。セッションとクッキーに関して天秤にかけなければならない項目は他にもあります。例えば、セッションに突っ込まれたデータはクッキーに入っているデータに比べてセキュリティ強度が上です。つまりクッキーが有効化されていて、単にセッションIDを覚えておくだけのために使われるとしても、セッションに突っ込まれている実際のデータはクッキーに直接突っ込まれた場合に比べて全然安全です。理由は簡単で、セッションデータはサーバにぶち込まれるため、権限のないユーザがアクセスするというのは非常に難しいためです。結論として高いセキュリティを保ってデータを扱わなければならぬ場合、セッションはクッキーに勝つことになります。

Q : データのサイズについてはどうなのでしょうか？何か関係がありますか？

A : あります。データのサイズも同様に重要です。セッションはクッキーに比べて大きなデータを入れておく能力があり、これもセッションに軍配を上げる際の理由の1つとなります。小さくて単純なテキストファイルを超えるデータを入れておく必要がある場合は、セッションでなければできません。もちろんMySQLデータベースは大きなデータをぶっ込むにはもっとよいかもしれません。セッションを動かす際といえどもこのことを忘れないで下さい。

Q : ではMySQLデータベースではなくセッションやらクッキーやらを使うのはどうしてですか？

A : 便利だからです。データをデータベースにぶっ込むには遙かに多くの労力が必要です。さらに忘れてならないのは、データベースと言うのは、理想的には恒久的なデータを入れておくのに適したものなのです。ログインのデータは真の意味では恒久的なデータではありません。するとクッキー やセッションが視界に入ってきます。この2つはちょっとの間データを覚えて、終わったら捨てるというデータに対しては便利です。



PHPマグネット

ミスマッチサイトのアプリケーションを再設計し、セッションとクッキーの両方を使って究極のログイン永続性を与えます。問題のは、コードが欠けていることです。セッションとクッキーのマグネットを使って空いているコードを埋めて完成させて下さい。



```

if (mysqli_num_rows($data) == 1) {
    // The log-in is OK so set the user ID and username session vars (and cookies),
    // and redirect to the home page
    $row = mysqli_fetch_array($data);
    .....['user_id'] = $row['user_id'];
    .....['username'] = $row['username'];
    setcookie('user_id', $row['user_id'], time() + (60 * 60 * 24 * 30)); // expires in 30 days
    setcookie('username', $row['username'], time() + (60 * 60 * 24 * 30)); // expires in 30 days
    $home_url = 'http://'. $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
    header('Location: ' . $home_url);
}

```

有効期限 30 日



login.php

```

<?php
// If the user is logged in, delete the session vars to log them out
session_start(); // ユーザがログインしている場合、セッション変数を削除しログアウト
if (isset(.....['user_id'])) {
    // Delete the session vars by clearing the $_SESSION array
    .....= array(); // $_SESSION 配列を再初期化することでセッション変数を削除
    // Delete the session cookie by setting its expiration to an hour ago (3600)
    if (isset(.....[session_name()])) { // 有効期限を1時間(3600)前に設定して、セッションのクッキーを削除
        setcookie(session_name(), '', time() - 3600);
    }
    // Destroy the session セッションの破壊
    session_destroy();
}

// 有効期限を1時間(3600)前に設定して、特定のユーザIDとユーザ名のクッキーを削除
// Delete the user ID and username cookies by setting their expirations to an hour ago (3600)
setcookie('user_id', '', time() - 3600);
setcookie('username', '', time() - 3600);

```



logout.php

\$_SESSION

\$_SESSION

\$_SESSION

\$_SESSION

\$_SESSION

\$_SESSION

\$_SESSION

\$_SESSION

```

<?php
session_start();
// If the session vars aren't set, try to set them with a cookie
// If the session vars aren't set, try to set them with a cookie
if (!isset(.....['user_id'])) {
    if (isset(.....['user_id']) && isset(.....['username'])) {
        .....['user_id'] = .....['user_id'];
        .....['username'] = .....['username'];
    }
}
?>
...

```



index.php



PHPマグネット の答え

ミスマッチサイトのアプリケーションを再設計し、セッションとクッキーの両方を使って究極のログイン永続性を与えます。問題なのは、コードが欠けていることです。セッションとクッキーのマグネットを使って空いているコードを埋めて完成させて下さい。

```

if (mysqli_num_rows($data) == 1) {
    // The log-in is OK so set the user ID and username session vars (and cookies),
    // and redirect to the home page
    $row = mysqli_fetch_array($data);
    • $_SESSION .['user_id'] = $row['user_id'];
    • $_SESSION .['username'] = $row['username'];
        ↓
        セッション変数に加えて
        新しくクッキーもセット
        します。
    setcookie('user_id', $row['user_id'], time() + (60 * 60 * 24 * 30)); // expires in 30 days
    setcookie('username', $row['username'], time() + (60 * 60 * 24 * 30)); // expires in 30 days
    $home_url = 'http://'. $_SERVER['HTTP_HOST'] . dirname($_SERVER['PHP_SELF']) . '/index.php';
    header('Location: ' . $home_url);
}

```



login.php

```

<?php
// If the user is logged in, delete the session vars to log them out
session_start();
if (isset(. $_SESSION .['user_id'])) {
    // Delete the session vars by clearing the $_SESSION array
    • $_SESSION .= array();
    // Delete the session cookie by setting its expiration to an hour ago (3600)
    if (isset(. $_COOKIE .[session_name()])) {
        setcookie(session_name(), '', time() - 3600); ←
    }
    // Destroy the session
    session_destroy();
}

// Delete the user ID and username cookies by setting their expirations to an hour ago (3600)
setcookie('user_id', '', time() - 3600); ←
setcookie('username', '', time() - 3600);
...

```



logout.php

```

<?php
session_start();
// If the session vars aren't set, try to set them with a cookie
if (!isset(. $_SESSION .['user_id'])) {
    if (isset(. $_COOKIE .['user_id']) && isset(. $_COOKIE .['username'])) {
        $_SESSION .['user_id'] = $_COOKIE .['user_id'];
        $_SESSION .['username'] = $_COOKIE .['username'];
    }
}

```



index.php

クッキーを使ってセッション変数をセットします。

これと同じクッキー・セッション対応のコード修正をeditprofile.phpとviewprofile.phpにも適用しなければなりません。



試運転

ミスマッチサイトを変更しセッションとクッキーの両方を使う

ミスマッチサイトのスクリプトを変更し、セッションとクッキーの両方を使ってログインの永続性をサポートします。またはスクリプトをWebサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードします。変更は次の4つのスクリプトに及びます：`index.php`, `login.php`, `editprofile.php`, `viewprofile.php`。

スクリプトをWebサーバにアップロードし、ミスマッチサイトのメインページ(`index.php`)をWebブラウザで開きます。ログインしたら、ブラウザを閉じてみます。これによりセッション変数は破壊されるはずです。再度メインページを開いてログイン状態のままでいるかどうかチェックして下さい。クッキーによりこのようなことが可能になっているのです。クッキーはブラウザのセッションを超えて生き続けることができるからです。





PHP & MySQL 道具箱

ミスマッチサイトのアプリケーションの一部としてユーザ管理システムを構築するという新しい領域で非常に多くの知識を得ました。重要な箇所の要点をまとめます。

`setcookie()`

PHPの組み込み関数でブラウザにクッキーをセットします。オプションで有効期限をセットすることもでき、これを過ぎるとクッキーは破壊されます。有効期限をセットしない場合、ブラウザを閉じたときにクッキーが破壊されます。

`$_COOKIE`

PHPの組み込みスーパーグローバルで、クッキーのデータにアクセスするために使います。このスーパーグローバルは配列で各クッキーは配列のエントリとして格納されています。つまりクッキーの値にアクセスするには、クッキーの名前を配列の添え字として指定する必要があります。

`session_start()`

PHPの組み込み関数で新しいセッションを開始するか、または既存のセッションを再開します。この関数を呼び出した後でないとセッション変数にアクセスしてはいけません。

`SHA(value)`

MySQL関数でテキストを暗号化して、40文字からなる16進数文字列を生成します。この関数は、データベースの中でさえも解読不能であることが要求されるようなデータを暗号化する手段を提供してくれます。しかしこの暗号化は一方方向です。つまり「復号化」関数はありません。

`session_destroy()`

PHPの組み込み関数でセッションを閉じます。特定のセッションが終了した場合は、この関数を呼び出すべきです。この関数を呼び出してもセッション変数は破壊されません。従ってセッション変数は別途自分できれいにしなければなりません。これには`$_SESSION`スーパーグローバルを空にすればよいのです。

`$_SESSION`

PHPの組み込みスーパーグローバルで、セッションのデータにアクセスするために使います。このスーパーグローバルは配列で各セッション変数は配列のエントリとして格納されています。つまりセッション変数の値にアクセスするには、セッションの名前を配列の添え字として指定する必要があります。

7½章 ダブったコードをなくす

シェアすればケアできる

簡単なことなんだよ。1本の傘をシェアすれば、傘は2本必要ないし、雨にも濡れない…しかもイケメンを1人手に入れることができるんだよ。

イケメンでしかもインテリなのね！あなたの傘シェア理論は正に天才的だわ。



シェアできるのは傘だけではありません。Webアプリケーションでも、同じコードが2箇所以上にダブっているというような状況に陥ることがあります。これは無駄なだけでなく、メンテナンス時には頭痛の種となります。不意に何らかの変更をしなければならなくなったとき、変更是すべてのダブっている箇所に波及するためです。このような事態を解決するには、ダブったコードをシェアすることなくしてしまえばよいのです。言い換えれば、ダブったコードは一箇所に貼り付けておき、それ以外の箇所からはそのコードを必要に応じて参照するようにすれば良いのです。ダブったコードをなくせば、アプリケーションはより効率的になり、メンテナンスも容易になり、最終的には堅牢なものとなります。

テンプレートでミスマッチサイトを作り直す

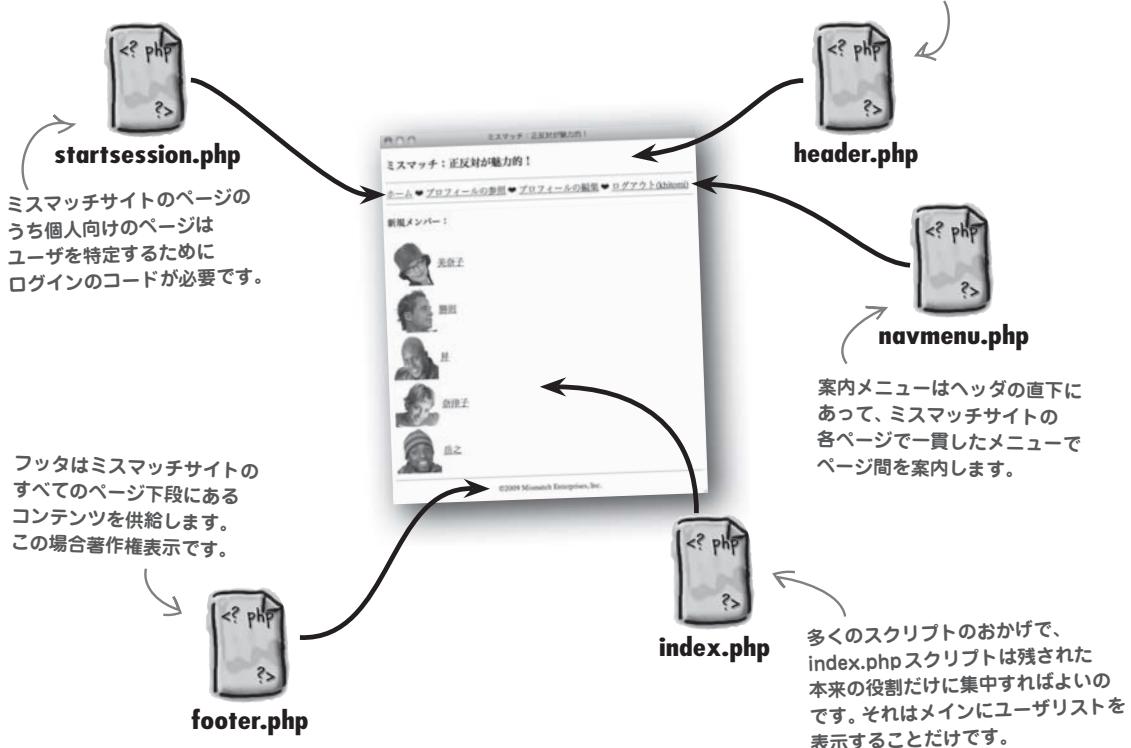
ミスマッチサイトをテンプレートという概念を用いて改良します。同一のコードが複数箇所に分散されているとメンテナンス上支障を来すことがよくあります。そこで冗長なコードはすべて1つのファイルにまとめてしまします。

このような考え方の下、ミスマッチサイトを複数のスクリプトに分割します。分割後に今度はどのように組み上げれば良いでしょうか？インクルードファイルがどのように作用するかということについてはよく知っているはずですが、これが答えの一部です。ただしインクルードファイルよりも大きな視野で考える必要があります…そこでテンプレートという考え方が必要です。これにより複数のインクルードファイルを組み合わせて1つページを組み上げることができます。テンプレートというのは、アプリケーション内のページの下書きのようなもので、そのページにしかないものを除くすべてのものをインクルードファイルから持ってきます。

テンプレート版のミスマッチサイトでは、共通コードはそれぞれ特定の役割を持ったスクリプトに分割します。表示用のHTMLコードを生成するものもあれば、そうでないものもあります。アイデアとしては、できる限り多くの共通機能を抽出して、テンプレートのインクルードファイルに突っ込んで、そのページに完璧に依存したものだけを各アプリケーションのページに残しておくのです。

テンプレートによって、
PHP アプリケーション
を再利用可能なスクリ
プトのコンポーネント
で構成することが
できるようになります。

ヘッダはミスマッチサイトの
すべてのページで使い、
アプリケーションのタイトルや
そのページに特有のタイトルを
表示します。



素朴な疑問 に答えます

 : テンプレートってちゃんと言うと何ですか?要するにインクルードファイルの類じゃないんですか?

 : そうです。テンプレートというのはインクルードファイルを集めたものです。ただしアプリケーションを機能的コンポーネントに分離することを主な目的として設計されています。ゴールは各ページが本当にそのページにしかないものだけで作られているようにサイズを削減することです。つまりヘッダ、フッタ、案内メニュー、その他のアプリケーションの部品は、いくつかのページにわたって同じあるいは似ているのであれば、アプリケーションのテンプレートにしてインクルードするのが理想なのです。最終的には、テンプレートコードをPHPのインクルードファイルに置いて、それを必要とする他のスクリプトが参照すればよいのです。

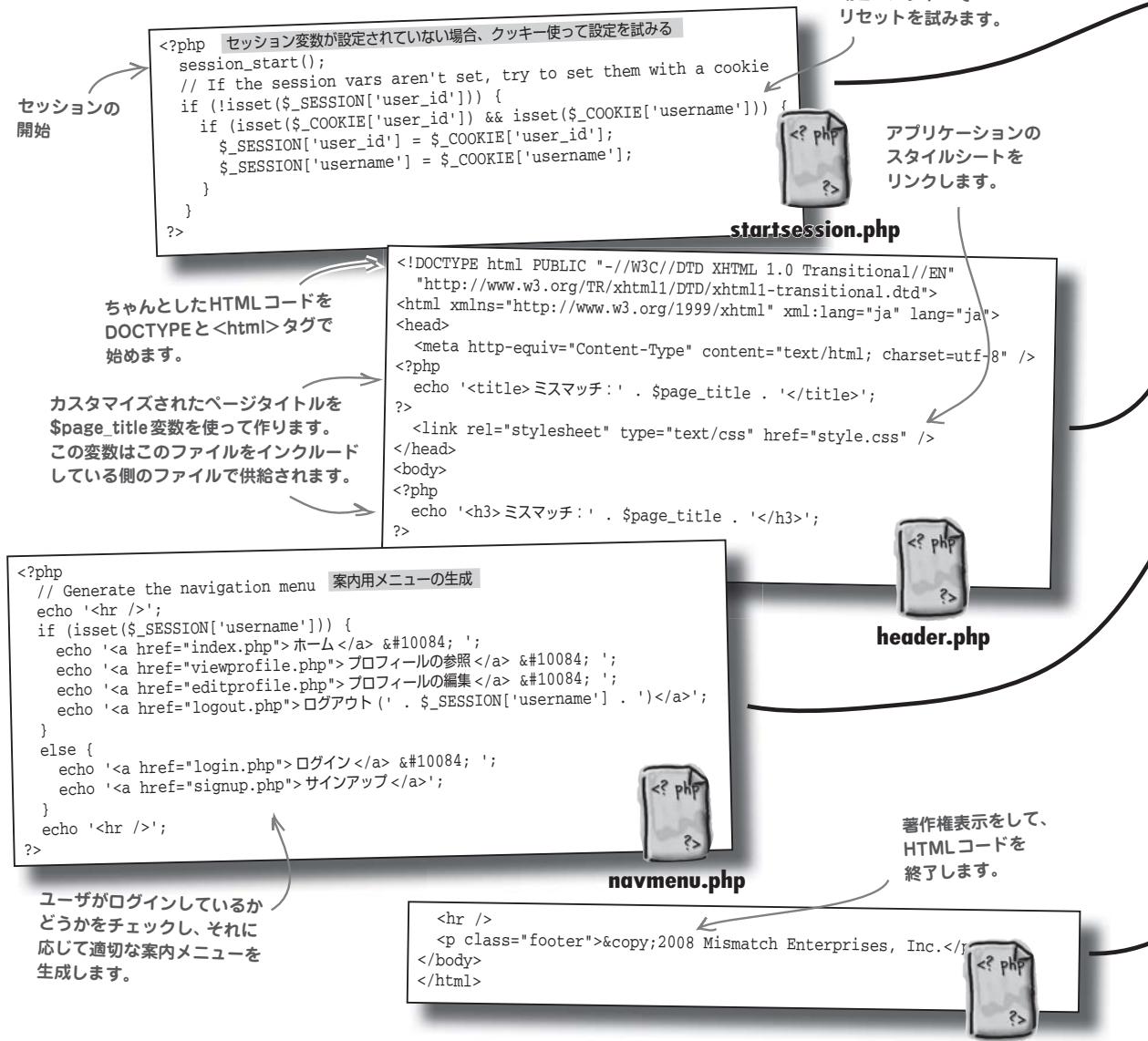
テンプレートというのは、インクルードファイルのグループのこと、ダブったコード單になくすよりはほんの少しエライものという風に考えることもできます。実際テンプレートは、アプリケーションを機能で構成するのに役立ちます。ミスマッチサイトの例では少々簡単すぎでテンプレートのありがた味が分からないかもしれません。もっと大きくて複雑なPHPアプリケーションの場合、極めて洗練されたテンプレートでシステムを構築できるものです。

 : テンプレートのコードは複数のスクリプトをまたいでシェアするため、完璧に一致していなければならぬのでしょうか?

 : そんなことはありません。単に似ているだけでも、完全に一致していなくともテンプレートコードとして十分役割を果たせます。その理由は変数を使って、ある程度のカスタマイズをして、異なるページでテンプレートを使えばよいからです。ミスマッチサイトのページタイトルは、このことを表す最適な例です。このページヘッダテンプレートは、各ページで似ていて、タイトルは常に「ミスマッチ」で始まります。しかしそれぞれのタイトルは異なっています。このため変数を使って異なるページ間で微妙に異なるタイトルに対応する必要があるのです。

テンプレートを使ってミスマッチサイトを再構築する

アプリケーションをテンプレートのスクリプトに分割するという設計は、非常に価値のある作業です。完了すれば、極めて限定的で極めて小さなスクリプトと、劇的に単純化された本体部アプリケーションのスクリプトファイルのコードができ上がり、本体部のスクリプトは、テンプレートに依存したものとなります。



まず始めに `startsession.php` スクリプトをインクルードして、セッションを開き、スクリプトの残りの部分がセッションデータにアクセスできるようにしなければなりません。

`$page_title` 変数で、ページのタイトルとしてページヘッダに表示されるものが決まります。

```
<?php セッションの開始
// Start the session
require_once('startsession.php');
// Insert the page header ページヘッダの挿入
$page_title = '正反対が魅力的！';
require_once('header.php');
require_once('appvars.php');
require_once('connectvars.php');
// Show the navigation menu 案内用メニューの表示
require_once('navmenu.php');
// Connect to the database データベースへの接続
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
// Retrieve the user data from MySQL MySQLからユーザデータの取り出し
$query = "SELECT user_id, first_name, picture FROM mismatch_user WHERE first_name IS NOT NULL " .
    "ORDER BY join_date DESC LIMIT 5";
$data = mysqli_query($dbc, $query);
// Loop through the array of user data, formatting it as HTML ユーザデータの配列を探し、HTMLにフォーマット
echo '<h4>新規メンバー:</h4>';
echo '<table>';
while ($row = mysqli_fetch_array($data)) {
    if (is_file(MM_UPLOADPATH . $row['picture']) && filesize(MM_UPLOADPATH . $row['picture']) > 0) {
        echo '<tr><td></td>';
    } else {
        echo '<tr><td></td>';
    }
}
if (isset($_SESSION['user_id'])) {
    echo '<td><a href="viewprofile.php?user_id=' . $row['user_id'] . '">' . $row['first_name'] .
        '</a></td></tr>';
}
else {
    echo '<td>' . $row['first_name'] . '</td></tr>';
}
echo '</table>';
mysqli_close($dbc);
?>
<?php ページフッタの挿入
// Insert the page footer
require_once('footer.php');
```

接続変数やアプリケーション変数は以前と同様に独立したスクリプトをインクルードします。

案内メニューはヘッダの後でかつページ本体の前に作ります。

テンプレート以外のコードはこれで、本当にそのページで一意になりました。これでコードが非常に少くなりました。

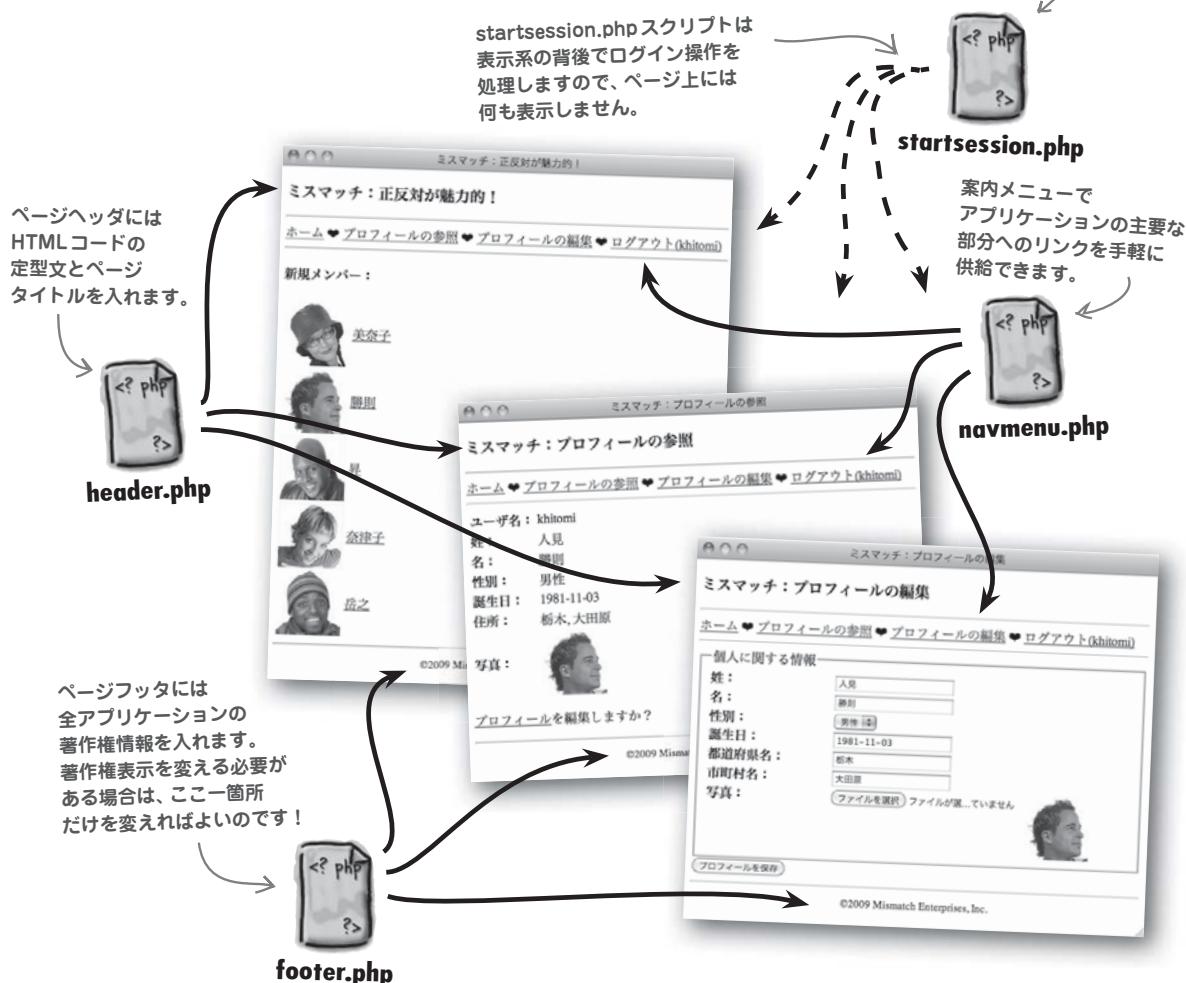


index.php

はるかに ミスマッチサイトがまた完成しました…しかも^よい構成で

ミスマッチサイトのアプリケーションをバラバラの小さな部品に分解するというのは、かなり気の滅入ることだったかもしれません。しかし、でき上がったものは間違いなく労力に報いる価値のあるものです。アプリケーションは今やいくつかの新しいテンプレート(インクルード)ファイルに分散しています。これにより遙かによい構成となり、スクリプトコードをシェアすることができます。仮にどこかに何らかの変更が必要になったとしても、単に1つのファイルを変更すればよく、その効果はアプリケーション全体に波及します…これこそがテンプレートの力なのです！

セッション開始用コードは、ユーザのログインが必要なすべてのページで使います。



8章 データをコントロールし、世界をコントロールする

データを収穫する



アタシのやっていることは、
全部データ管理なのよ。まず初めに豆を
ソートして、次にジャガイモをSELECTする。
そうしたらこの2つをセロリやトウモロコシの
列とJOINする(混ぜる)…そしたらほら、
美味しいシチューのできあがり！

データ収穫の秋は最高です。

有り余る程の情報が、検査、ソート、比較、結合を待っています。そして一般にキラーWebアプリが必要とするすべての処理は何でもできます。十分に？もちろんです。でも本当の収穫とは違って、MySQLデータベースのデータをコントロールするには、ちょっとした労力とかなりの専門知識が必要です。Webユーザはくたびれてしおれたデータなんか欲しくありません。そんなもの退屈で全然魅力がありません。欲しいのは、豊富で十分で、しかも適切なデータです。さあ、ぐずぐずしている場合ではありません。MySQLトラクターを始動させて、作業開始です！

完全なミスマッチにする

ミスマッチサイトのアプリケーションには、登録されたユーザのデータがデータベースにたまってきたました。そろそろみんな結果を見たいところです。ユーザに理想的な「正反対」を見つけられるように、各自の「好き」と「嫌い」を他のユーザのものと比べてミスマッチを探せるようにする必要があります。それぞれの「好き」が相手の「嫌い」にミスマッチすればするほど、そのペアは、完璧なミスマッチへ近づくわけです。

加理奈さんにはまだ
カレがいませんが、自分が
リアリティ一番組を好きな
分、カレはそれを嫌いなの
ではないかという予感が
あります。

アタシ、ホラー映画大っ嫌い。
あとSPAMもゲッって感じ！
でもバーバラ・ストライサンド大好き。
それと最高なのは何と言っても
ハイキングね…



オレのハートを熱くさせるのは、
超厚切りのSPAMサンドだけさ。
バーバラ・ストライサンドがハイキングの
映画に出てきたら最悪だね！

人見さんです。覚えています
か？寂しいハートの持ち主
で自分が大好きなウェイト・
リフティングを嫌っている
人を探しています。



嫌い：タトゥー
好き：カウボーイブーツ
好き：リアリティー番組
嫌い：ホラー映画
嫌い：SPAM
好き：激辛

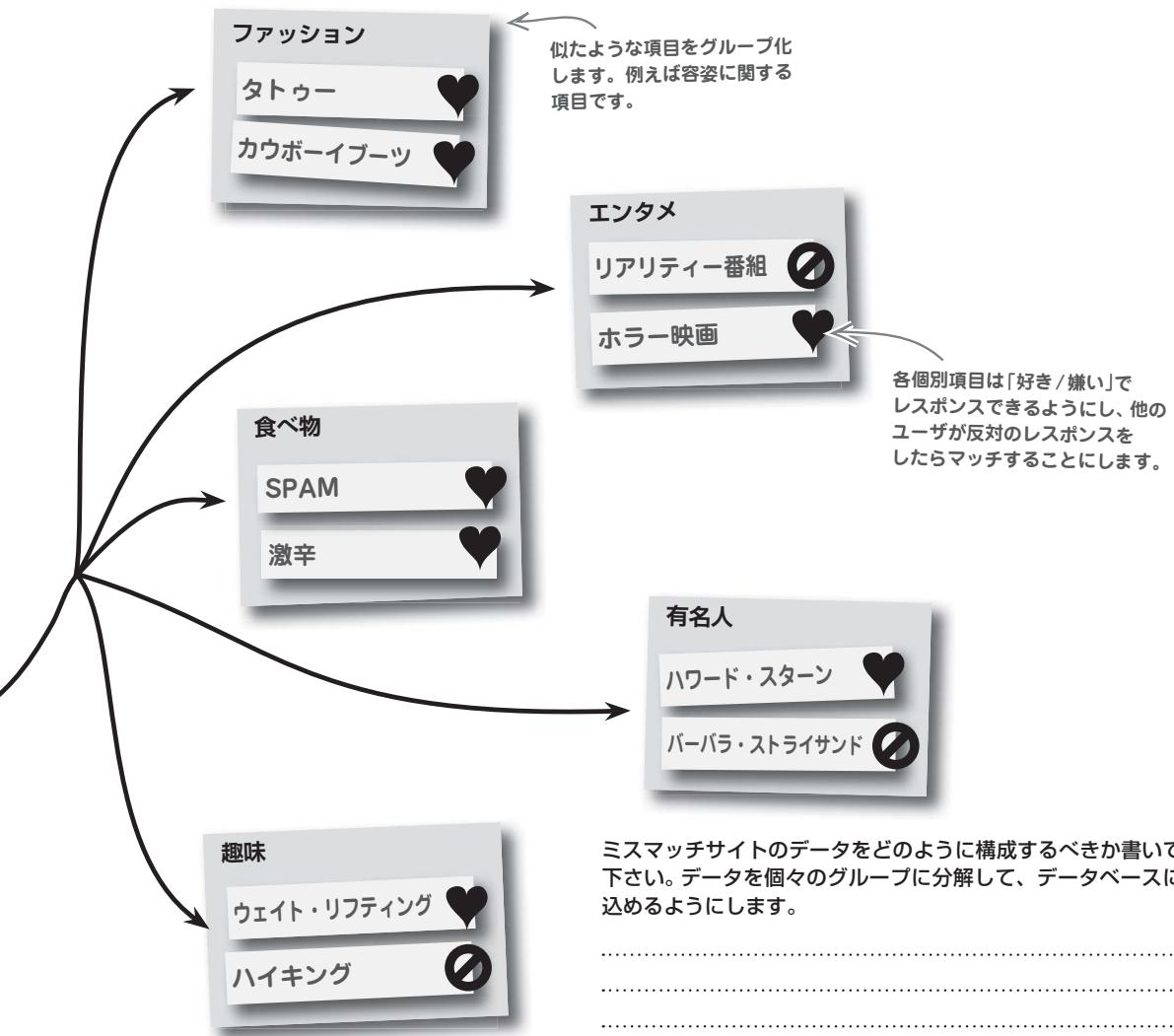
嫌い：ハワード・スター
好き：バーバラ・ストライサンド
嫌い：ウェイト・リフティング
好き：ハイキング

好き：タトゥー
好き：カウボーイブーツ
嫌い：リアリティー番組
好き：ホラー映画
好き：SMAP
好き：激辛
好き：ハワード・スター
嫌い：バーバラ・ストライサンド
好き：ウェイト・リフティング
嫌い：ハイキング

加理奈さんの「好きー嫌い」リストは人見さんのと
比べると、まるっきり違います。ということは
とてもよいミスマッチなペアということです。

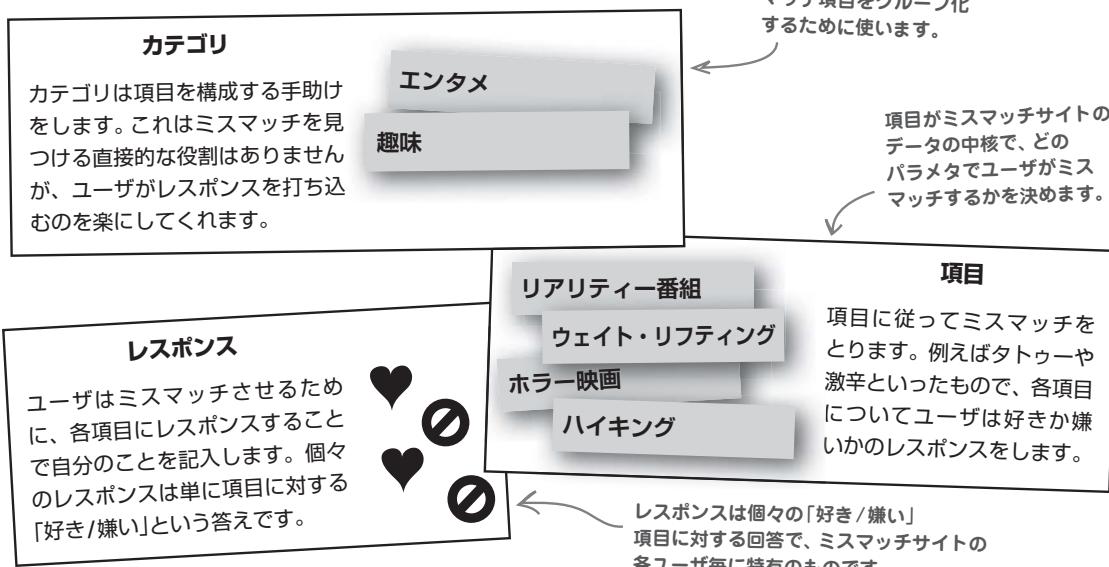
ミスマッチさせるにはデータがすべてです

ユーザのミスマッチ同士を見つけ出すには、ユーザは何が好きで何が嫌いかを表現するデータをうまく構成してあげなければなりません。データをMySQLデータベースにぶつ込むということがわかっているだけではダメです。この「好き／嫌い」項目をうまく構造化して、もっと管理しやすくし、ユーザが関連項目にレスポンスできるようにし、ユーザが各項目を好きか嫌いかの意思表示ができるようにするのです。



ミスマッチサイトのデータをブレイクダウンする

ミスマッチサイトのようなアプリケーションにおいて、データモデルという概念を使うことは極めて重要です。この概念を使うことで、アプリケーションがどのように構築されるかということに関して恐ろしいほど多くのことをコントロールできます。ミスマッチサイトの場合で言うと、データをブレークダウンして、3つの相互に関連したデータにする必要があります。



このデータでどの程度正確に2人のユーザをミスマッチに導くことができるでしょうか？今やろうとしていることは、ユーザが各項目に対して行ったレスポンスを比較することです。例えば、加理奈さんと人見さんは「ホラー映画」という項目に対して、反対のレスポンスをしましたから、この項目に関してだけ言えばミスマッチが成功したことになります。あるユーザに対して、全体の中から最高のミスマッチを見つけ出すには、最もミスマッチ項目の多いユーザを見つけることになります。

加理奈さんがホラー映画を嫌いなのでミスマッチすることになります。

ホラー映画



嫌いです！



ホラー映画



ミスマッチしました！

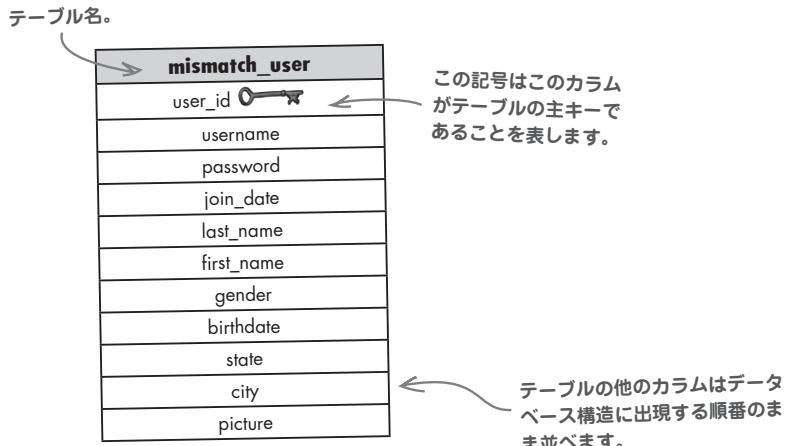
好きです。



スキーマでデータベースをモデル化する

ミスマッチサイトのアプリケーションでデータに関する要件を現実のデータベース設計に翻訳するには、スキーマという概念が必要です。スキーマとはすべてのテーブルやカラムといった構造を表現するものです。データベースを視覚的に描画することで、問い合わせ文を書いたときに、何がどうつながっているのかを見やすくなります。どの特定のカラムがつながっているのかということを、言わなくてよくなります。例えば、前章にあった元々のミスマッチサイトのデータベースに対するスキーマを見てみましょう。このデータベースは1つのテーブル `mismatch_user`だけからできています。

**データベースのデータ
(テーブルとカラム)と
その他の関連オブジェクト
およびこれらがどのように
つながっているかを説明した
ものをスキーマと言います。**

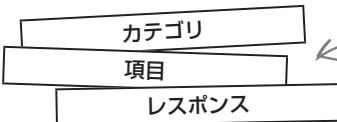


このテーブルの見方は、今までとちょっと違っています。今までテーブルはカラム名を横に並べて、その下にデータを書いてきました。このやり方は、個々のテーブルとそのテーブルに突っ込まれているデータを見るには、よくできた方法なのですが、複数のテーブルがどのように関連しあっているかを構造化ダイアグラムで表現するには、あまり実用的ではありません。そしてミスマッチサイトは、複数のテーブルを必要としています…

**テーブルのダイアグラムを
作るには、テーブルの設計と
テーブルの中に入れるデータ
とを分けて考える必要が
あります。**



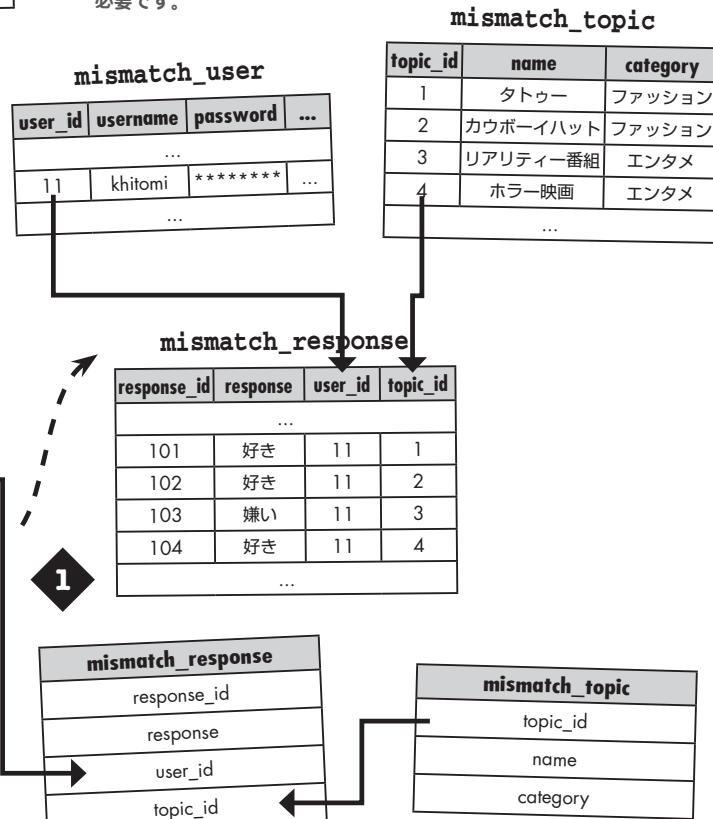
ミスマッチサイトのデータベースには、「好き/嫌い」項目に対するユーザのレスポンスをぶち込む場所が必要になりました。同様に項目名や対応するカテゴリも必要です。以下に3つの異なるデータベース設計を示します。どれもカテゴリ、項目、レスポンスをミスマッチサイトのデータベースに組み込んでいます。最も理にかなっていると思われるスキーマにマルをつけ、なぜそう思うのか書いて下さい。

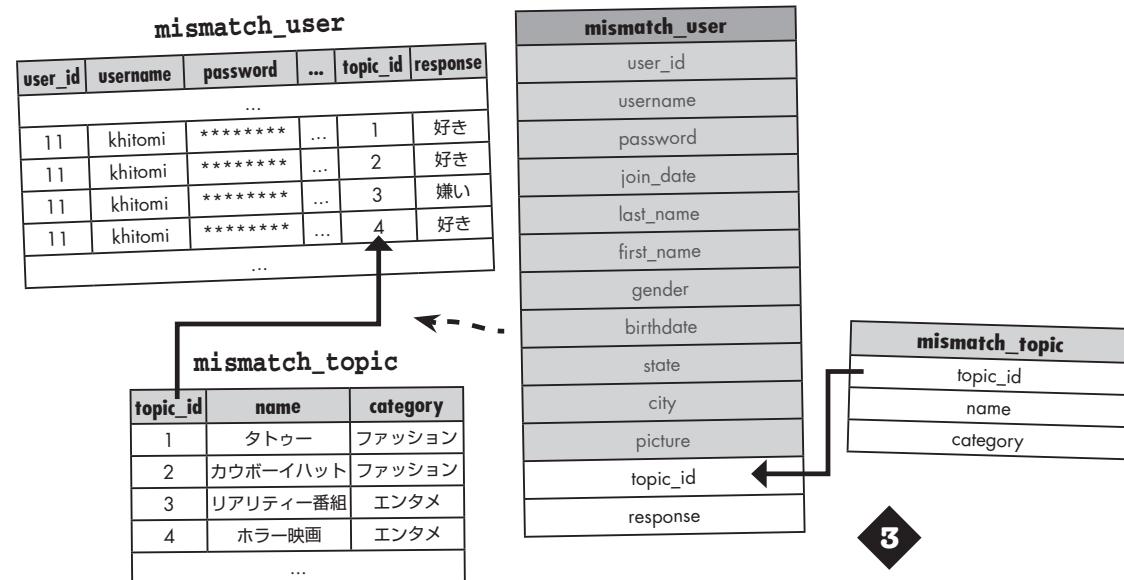
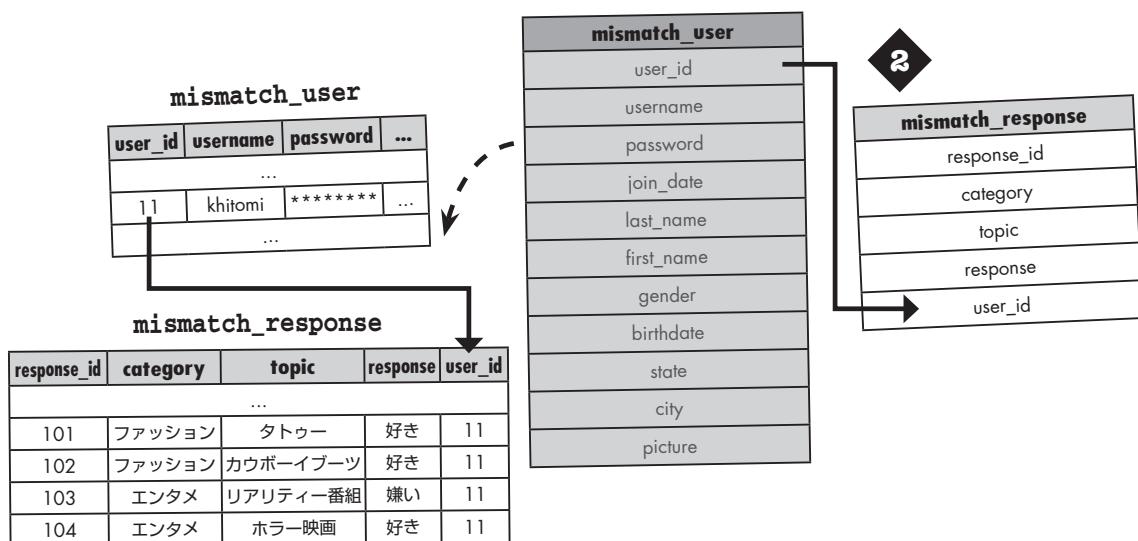


これが新しく導入されたデータで、
ミスマッチサイトのユーザの
「好き/嫌い」を覚えておくために
必要です。

mismatch_user		
user_id	username	password
...		
11	khitomi	*****
...		

mismatch_user		
user_id	username	password
user_id	username	password
username		
password		
join_date		
last_name		
first_name		
gender		
birthdate		
state		
city		
picture		







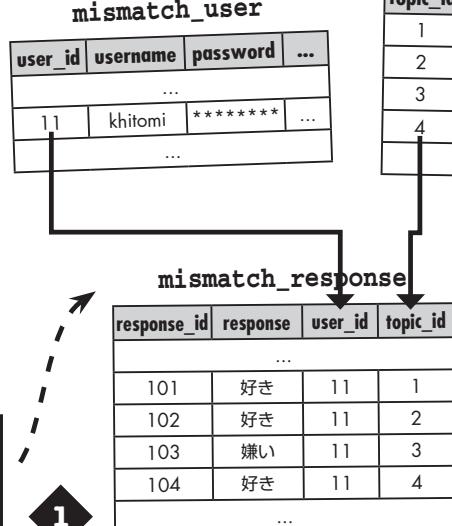
ミスマッチサイトのデータベースには、「好き/嫌い」項目に対するユーザのレスポンスをぶち込む場所が必要になりました。同様に項目名や対応するカテゴリも必要です。以下に3つの異なるデータベース設計を示します。どれもカテゴリ、項目、レスポンスをミスマッチサイトのデータベースに組み込んでいます。最も理にかなっていると思われるスキーマにマルをつけ、なぜそう思うのか書いて下さい。

まず始めに、ユーザが入力した「好き/嫌い」のレスポンスによって得られる新しいデータは、正にレスポンスだけだということ認識しておかなければなりません。データベースに入っているそれ以外のすべてのものは、少なくともユーザの視点から見れば変わっていません。

シンプルイズベストが常に正しいと言ったのは誰でしょうか？このデータベーススキーマでは、レスポンスをレスポンス用のテーブルに突っ込んでいます。他のデータは分離されているため、レスポンスの影響を直接的に受けることはありません。レスポンスによってデータが重複することもありません。ユーザ、カテゴリ、項目のすべてが mismatch_response テーブルの外側にあるためです。

mismatch_user
user_id
username
password
join_date
last_name
first_name
gender
birthdate
state
city
picture

元々の mismatch_user テーブルには変更がありません。



1

mismatch_response
response_id
response
user_id
topic_id

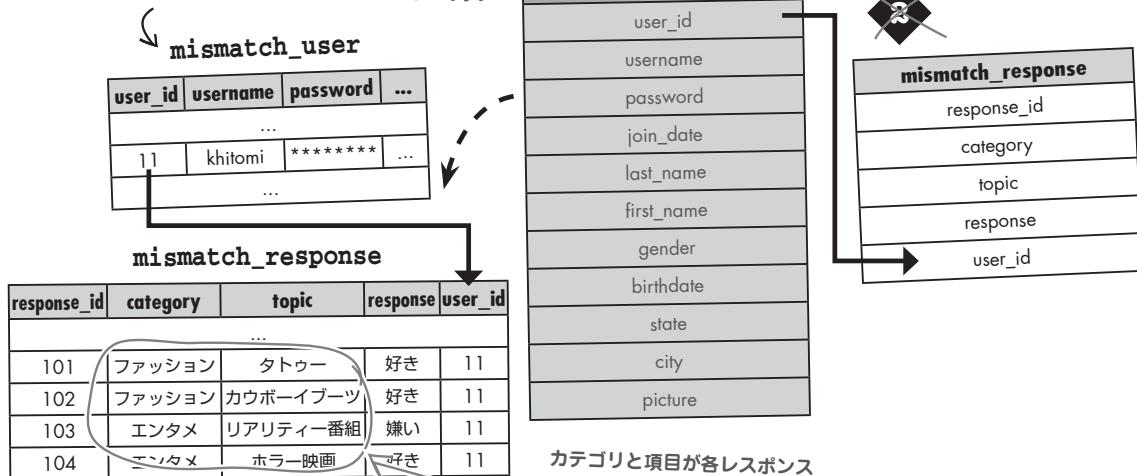
topic_id	name	category
1	タトゥー	ファッショ
2	カウボーイハット	ファッショ
3	リアリティー番組	エンタメ
4	ホラー映画	エンタメ
		...

新しく作った mismatch_topic テーブルに項目名と対応するカテゴリをぶち込みます。

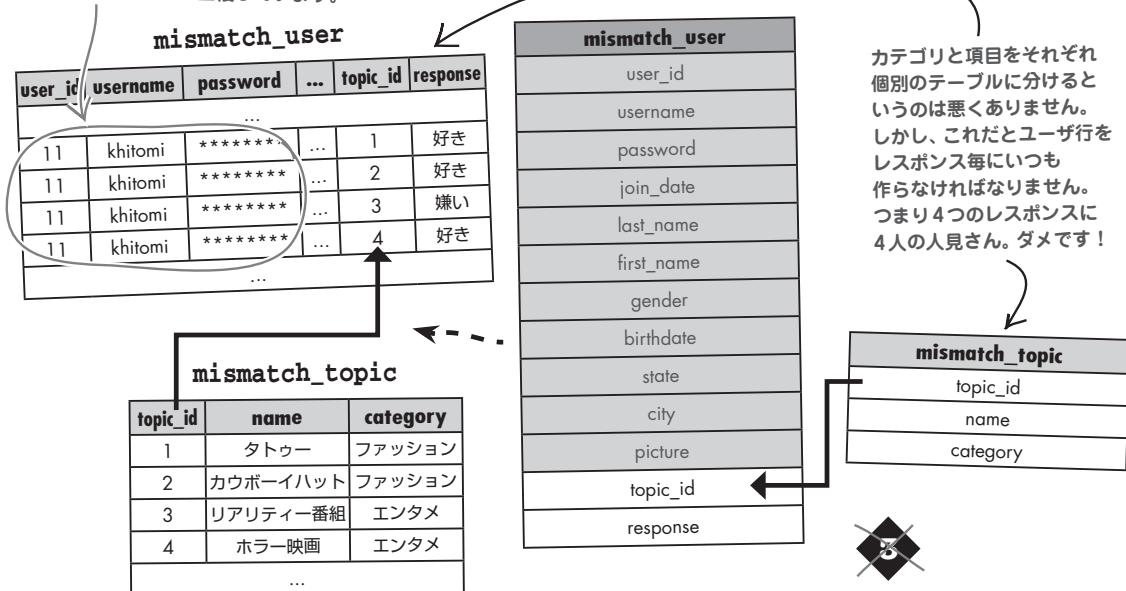
各レスポンスに対して重複データはありません。これは非常に良いことなのです！

mismatch_response テーブルはユーザと項目とを user_id カラムと topic_id カラムを使ってつなぎます。

レスポンスがユーザーテーブル内に突っ込まれてないというのはすばらしいことです。しかし、重複データが腐るほどできてしまします。カテゴリと項目がすべてのレスポンスに重複しているためです。



ユーザーデータがすべてのレスポンスについて、恐ろしいほど重複しています。



複数のテーブルを結び付ける

テーブルを相互接続してデータ結合システムを作り上げるには、キーを使う必要があります。すでに主キーを使ってテーブル内のデータを一意に識別できるようにしました。しかし今度は外部キーというものが必要で、あるテーブルの1つの行を別のテーブルの1つの行とリンクします。あるテーブルにある外部キーは、別のテーブルの主キーを参照します。これにより2つのテーブル間の結合を確立し、問い合わせ文で使うことができます。

先ほどのエクササイズで見たミスマッチサイトのスキーマは、mismatch_response テーブルにある2つの外部キーのおかげで、レスポンス行とユーザを結び付け、項目行を別のテーブルに結び付けています。

 **外部キーは**
テーブル内のカラムで
別のテーブルの
 **主キーを参照して**
います。

mismatch_user	
user_id	○ key
username	
password	
join_date	
last_name	
first_name	
gender	
birthdate	
state	
city	
picture	

この矢印は mismatch_user テーブルが mismatch_user テーブルとが2つのキーでつながっていることを表します。

この記号は主キーを表すことを忘れないで下さい。

mismatch_response	
response_id	○ key
response	
user_id	○ key
topic_id	○ key

この記号はこのカラムが外部キーであることを表します。外部キーは別のテーブルの主キーを参照します。

この主キーは、mismatch_topic テーブルの中で項目を一意に識別するだけでなく、mismatch_response テーブルでの項目に対するレスポンスと結合しています。

mismatch_topic	
topic_id	○ key
name	
category	

mismatch_response テーブルと mismatch_topic テーブルとで合わせて項目に対する「好き／嫌い」レスポンスをぶち込みます。例えば ファッション -> カウボーイブーツや 趣味 -> ハイキングに対する レスポンスです。

外部キーがないと、1つのテーブルにあるデータを別のテーブルにあるデータと結びつけることが非常に難しくなります。複数のテーブルにまたがってデータを散在させることで、データの重複をなくすことができ、効率的なデータベースになっていきます。つまり外部キーは、データベーススキーマの最も単純な要素ではありますが、最も重要な役割を果たしているのです。

大きな矢印は主キー
と外部キーとを結びつける
ことでテーブル同士を
結合します。

外部キーの動作

データの流れを、テーブル同士が主キーと外部キーとで結合していることを視覚化するとわかりやすくなります。ミスマッチサイトのテーブルを実際のデータをいくつか使って詳しく見ていくことで、主キーと外部キーがお互いにどのように関連しているのかを理解しやすくなります。

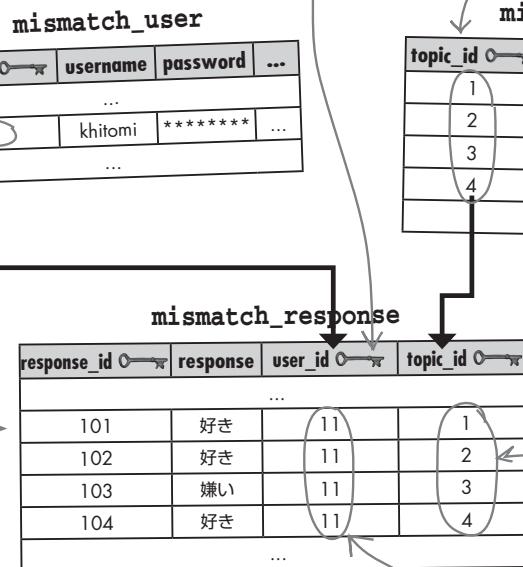
主キーなのでuser_idはmismatch_userで一意でなければなりません。事実それが主キーの目的そのものです。つまりユーザ行を一意に参照するためのものです。

user_id外部キーでmismatch_userテーブルのユーザ行を参照することができます。どのユーザがどのレスポンスをしたかを関連付けています。

topic_id主キーはmismatch_topicテーブルの行に対する一意なインデックスを供給してくれます。

このテーブルの各行ミスマッチサイトのユーザに対応していることを忘れないで下さい。

このテーブルの各行は特定のユーザによる特定の「好き／嫌い」レスポンスを表現します。

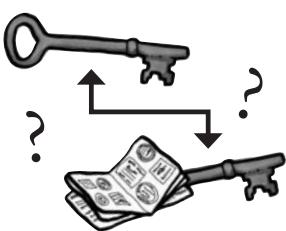


mismatch_responseテーブルの中で、ユーザに関する、より詳しい情報を探し出すには、user_idをmismatch_userテーブルからたどることで、ユーザが打ち込んだレスポンスを見つけることができます。同様にレスポンスの項目名やカテゴリを探し出すには、user_idをmismatch_topicテーブルの中からたどることで、見つけることができます。

主キーと外部キーとでテーブルを結合することで、一貫性をもってデータを結合させることができます。データベースを構築する場合に、主キーと対応する外部キーが正しくマッチしなければならないように強制することもできます。これは参照整合性(**referential integrity**)と呼ばれていて、すべてのキーの参照が妥当でなければならないことを言うための素敵な方法です。

topic_id外部キーは、mismatch_topicテーブルの項目行を参照していますから、一意ではありません。なぜなら多くの異なるユーザが同じ項目に対してレスポンスをするからです。

user_id外部キーは、mismatch_userテーブルのレスポンス行とユーザ行を結び付けます。ユーザはいつも「好き／嫌い」レスポンスをすることができますから、このキーは一意ではありません。



主キーと外部キーで複数の
テーブルを結合できるっていうことはわかった
けど、このブロック図にあるキーの間の矢印の
向きになんか意味があるのかな？

あります。矢印の向きで、各テーブルの行がどのような関係に
あるか教えてくれています。

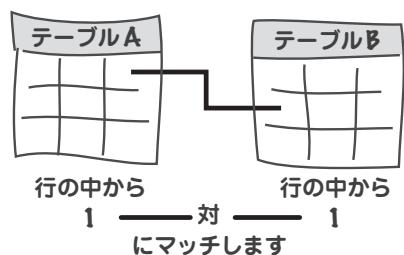
もっと正確に言うと、矢印は1つのテーブルの何行分が別のテーブルの
行とマッチしているかを、逆方向も含めて教えてくれています。これは
データベースのスキーマを設計する上で極めて重要な観点で、3つの異
なるデータのパターン、1対1、1対多、多対1、を表します。



テーブルは行と行でマッチします

最初のパターン1対1は、テーブルAの1行がテーブルBの高々1行と
マッチし、逆も同様にマッチするというものです。つまり各テーブルの各
行には1つのマッチしか存在しません。

例えば、ミスマッチサイトのユーザーテーブルが2つのテーブルに分かれ
ているとしましょう。1つはログイン情報(テーブルA)とし、もう1つはプロ
フィールデータ(テーブルB)とします。どちらのテーブルにもユーザID
があれば、ユーザとそのプロフィールとを結合しておくことができます。
ログインテーブルのuser_idカラムが主キーで、ユーザのログインデー
タが一意であることを保証してくれます。プロフィールテーブルのuser_
idは外部キーですから役割が違います。こっちの仕事はプロフィールとロ
グインとを單にくっ付けることです。



mismatch_user_login			
user_id	username	password	join_date
...			
9	natsuko	08447b...	2009-10...
10	hnoboru	230dc...	2009-10...
11	khitomi	e511d7...	2009-10...
12	minako	062e4a...	2009-10...
13	seino	b4f283...	2009-10...

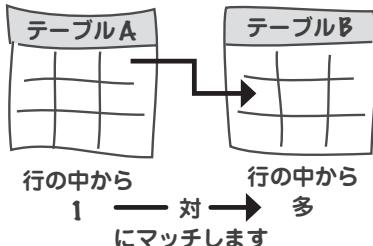
1対1の場合、
矢印はつけ
ません。

user_profile_id	last_name	first_name	gender	...	user_id
...					
7	人見	勝則	男	...	11
8	長谷川	岳之	男	...	8
9	久永	昇	男	...	10
...					

テーブルはuser_id
を通して1対1の
関係にありません。

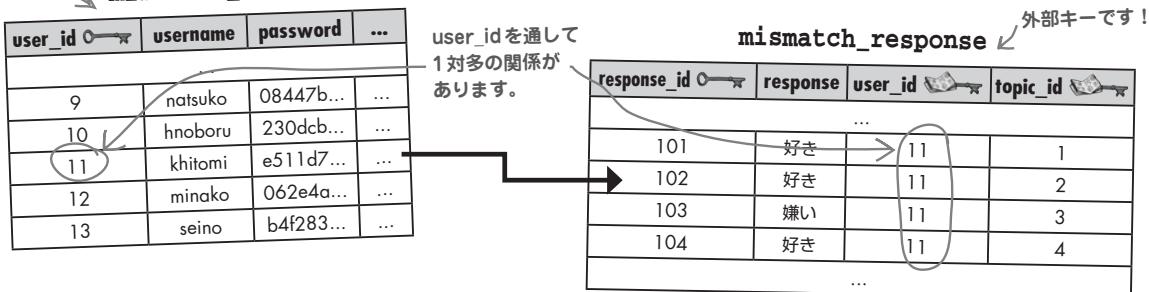
user_idカラムの視点で、ログインテーブルは親テーブルであり、プロフィール
テーブルは子テーブルと考えることができます。主キーを持つテーブルは、対応する
外部キーを持つテーブルと親子関係を持ちます。

1行が複数行に対応する



1対多の意味は、テーブルAの1行がテーブルBの複数行にマッチすることがある代わりに、テーブルBの1行はテーブルAの1行にしかマッチしないということです。テーブルダイヤグラムの矢印の向きは、1行側のテーブルから複数行側のテーブルを指します。

ミスマッチサイトのデータベースを再び使います。現状のスキーマはすでに1対多のデータパターンによる恩恵に浴しています。ユーザはたくさんの項目(タトゥーが好きか、ハイキングが嫌いか、など)に対してレスポンスすることができますから、ユーザ行とレスポンス行との間には1対多の関係があります。user_idカラムで、これら2つのテーブルをつないでいますが、このカラムはmismatch_userの主キーでmismatch_responseの外部キーです。



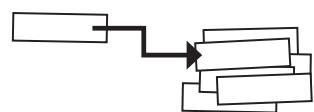
素朴な疑問に答えます



1対1:
親のテーブルの
1行が子のテーブル
の1行に対応
します。

Q: 2つのテーブルの行同士が1対1か1対多のどちらの関係かを知る方法があるのでしょうか？

A: 1対1に比べて、1対多のパターンを使うことの方が遙かに多いという傾向にあります。一般にメインの(親の)テーブルに主要なデータを持たせます。ミスマッチサイトで言えば、ユーザ情報です。このテーブルから派生的な(子の)テーブルへ1対多の配置で結合します。このような関係は、ミスマッチサイトのスキーマでは2度使われています。ユーザも項目もどちらも1対多の関係でレスポンスにつながっています。多くの場合、2つのテーブルで1対1関係を持つ行は、1つのテーブルに結合してしまうことができます。しかし、1対1のパターンが意味を持つような場合というのも間違いなく存在します。例えば、反対側のページにある仮想的なユーザプロファイルの例では、セキュリティ上の動機により、データの一部を固有のテーブルに移しているのです。



1対多：
親のテーブルの
1行がこのテーブル
の複数行に対応
します。

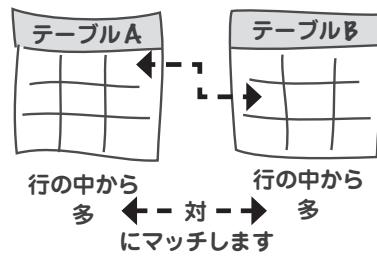
多対多で行をマッチさせる

テーブル行における第3の、かつ最後のデータパターンの関係は**多対多**の関係です。これはテーブルAの複数行のデータがテーブルBの複数行にマッチします…つまりデータのオーバーロードのような感じです！ただし全部がそうというわけではありません。多対多のパターンがめでたいような状況というのはたくさんあります。ミスマッチサイトでも？ちょっと見てみましょう。

user_id	username	password	...
9	natsuko	08447b...	...
10	hnoboru	230dcb...	...
11	khitomi	e511d7...	...
12	minako	062e4a...	...
13	seino	b4f283...	...

mismatch_user

ユーザと項目とは
レスポンスを通して
多対多の関係にあります。



mismatch_topic

topic_id	name	category
1	タトゥー	ファッショニ
2	カウボーイハット	ファッショニ
3	リアリティ一番組	エンタメ
4	ホラー映画	エンタメ
...		

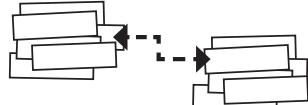
mismatch_response

response_id	response	user_id	topic_id
81	嫌い	9	1
82	好き	9	2
83	好き	9	3
84	嫌い	9	4
...			
101	好き	11	1
102	好き	11	2
103	嫌い	11	3
104	好き	11	4
...			

mismatch_responseは
結合のためのテーブルで、
ユーザと項目に対する
レスポンスとの関係を
確立してくれます。

ミスマッチサイトで多対多のパターンは間接的です。つまりmismatch_responseテーブルを介して現れます。しかしパターンが存在していることは確かです。同じuser_idとtopic_idがどのくらいmismatch_responseに出てくるか見てみます。

レスポンスデータを持っておくことに加えて、mismatch_responseテーブルは**結合テーブル(junction table)**[†]と呼ばれる機能を持っていて、ユーザと項目との間を簡単に行き来する方法を提供してくれます。結合テーブルがないと、重複したデータをたくさん持つことになるため、全然よくありません。納得しきれない場合は、本章の最初の方にあるスキーマについてのエクササイズに戻って、2番目の設計をよく見て下さい。あの設計では、mismatch_topicテーブルはmismatch_responseテーブルに抱え込まれていて、結果としてたくさんの重複データができてしまっています。

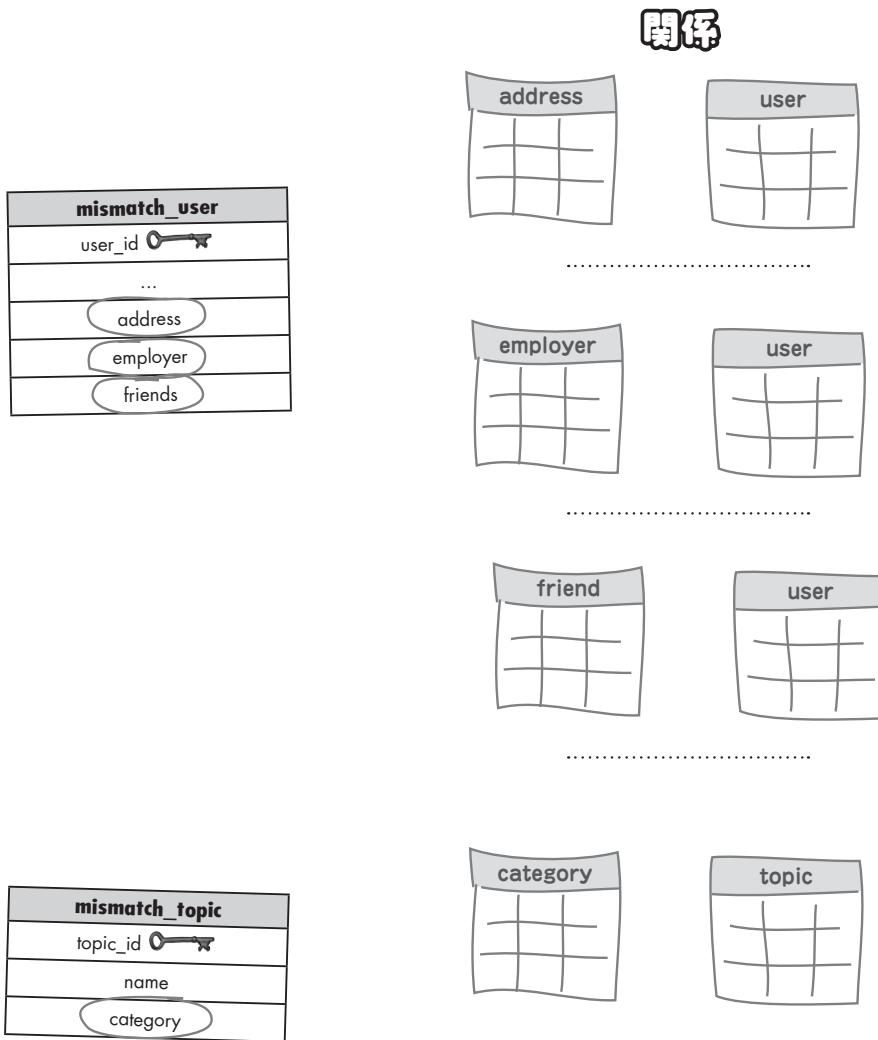


多対多：
親テーブルの
複数行が
子テーブルの
複数行と
対応します。

† 訳注：交差テーブルと呼ばれることもあります。

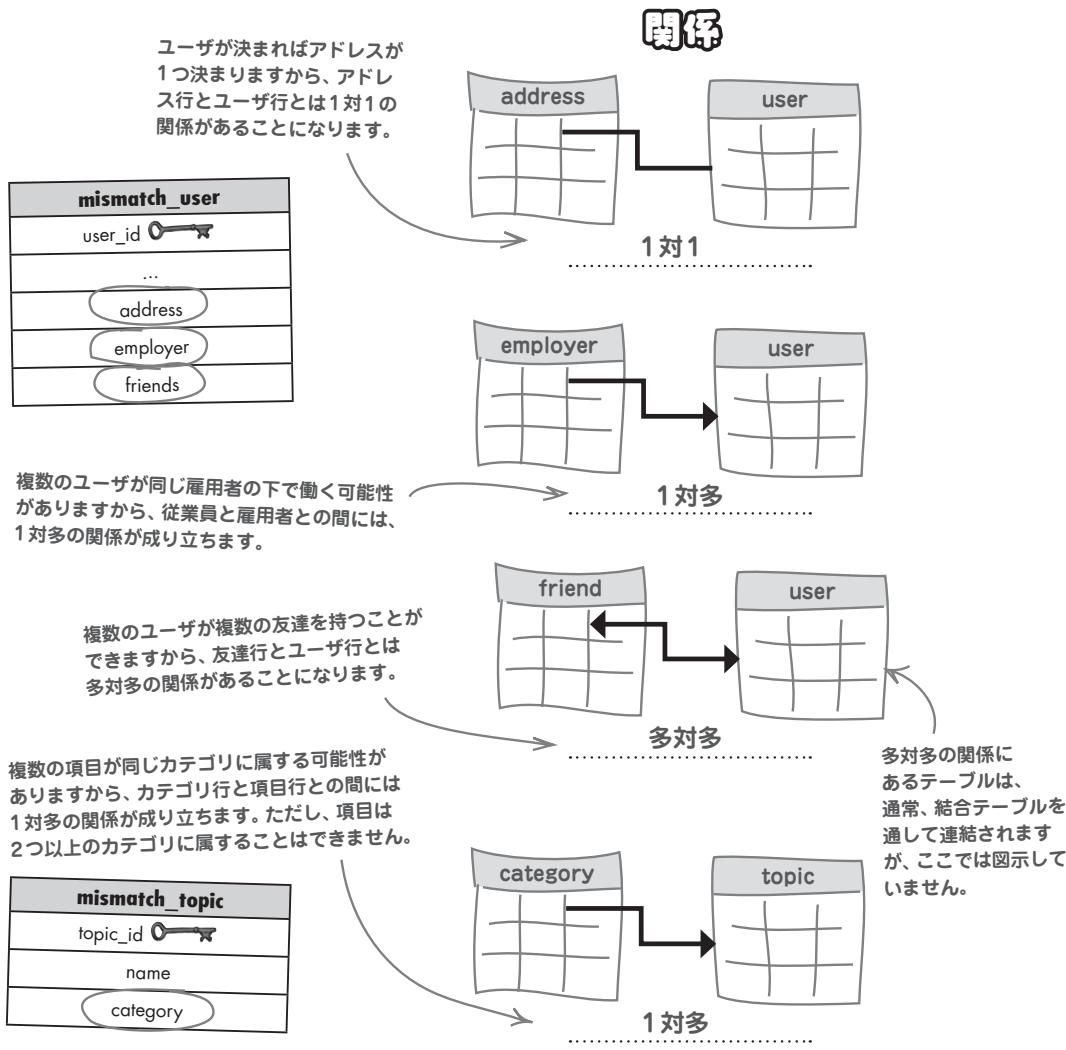
関係に名前をつけまSHOW

以下の各テーブルについて、マルをつけたカラムを、専用のテーブルに追い出します。各カラムのテーブルは元々のテーブルに対して、1対1、1対多、または多対多のどの関係で表現されるのが最もよいと思われるか、書いて下さい。次に2つのテーブルの関係を適切な矢印をつけた線で結んで下さい。



関係に名前をつけて SHOW の答え

以下の各テーブルについて、マルをつけたカラムを、専用のテーブルに追い出します。各カラムのテーブルは元々のテーブルに対して、1対1、1対多、または多対多のどの関係で表現されるのが最もよいと思われるか、書いて下さい。次に2つのテーブルの関係を適切な矢印をついた線で結んで下さい。





ちょっと止まって下さい！ミスマッチサイトのデータベースを取ってきて、ミスマッチをマッチさせることができますようにします。

ミスマッチサイトのアプリケーション用の.sqlファイルをサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードします。このファイルにはSQL文が書いてあって、ミスマッチサイトに必要なテーブルを作ってくれます。テーブル名はmismatch_user, mismatch_topic, mismatch_responseです。MySQLツールで各.sqlファイルのSQL文を走らせ、ミスマッチサイトのテーブルを初期状態にして作業を開始できるようにします。

すべての作業が終わったら、DESCRIBE文を走らせ、新しい2つのテーブル(mismatch_topic, mismatch_response)の構造が正しいか2重にチェックして下さい。この2つのテーブルは、これから組み込むミスマッチサイトのPHPスクリプトの重要な要素となります。

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> DESCRIBE mismatch_topic;
+-----+-----+-----+-----+-----+
| Field | Type  | Null | Key | Default | Extra       |
+-----+-----+-----+-----+-----+
| topic_id | int(11) | NO   | PRI | NULL    | auto_increment |
| name      | varchar(48) | YES  |     | NULL    |              |
| category  | varchar(48) | YES  |     | NULL    |              |
+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql> DESCRIBE mismatch_response;
+-----+-----+-----+-----+-----+
| Field | Type  | Null | Key | Default | Extra       |
+-----+-----+-----+-----+-----+
| response_id | int(11) | NO   | PRI | NULL    | auto_increment |
| user_id     | int(11) | YES  |     | NULL    |              |
| topic_id    | int(11) | YES  |     | NULL    |              |
| response    | tinyint(4) | YES  |     | NULL    |              |
+-----+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql>
```

外部キー topic_id は mismatch_topic テーブル の主キーと結び付きます。



わかったわ。じゃ、これですっごくよくできたデータベースでユーザとカテゴリと項目とレスポンスを管理できるのね。で、実際にどうやってミスマッチをマッチさせる役に立てられるのかしら？

データベースの設計がよくできていれば、アプリケーションパズルの残りのピースは、いとも簡単に解けるので組み立ても簡単です。

アプリケーションを最初から設計する際にデータベースがちゃんとしていることが、多分一番大事です。そうすれば以降の開発プロセスは非常にスムーズに進みます。データをぶち込むための最良の方法を探すための事前の準備や計画が多すぎると思うかもしれません。でも先は長いので十分にペイします。データがたくさんある状態になってしまってから、ミスマッチサイトのデータベースを作り直す羽目に陥ったらどれだけ厄介か想像してみて下さい。

これが、良いデータベースを設計することの大きな目に見える利点なのです。ミスマッチサイトのデータベースについて特に言えば、ユーザテーブルはユーザ自身のためにあってユーザ登録とプロフィール編集を担当します。新しく作った項目テーブルは、カテゴリと項目を十分に収容して、個人に対する適切な性格判断をしてくれます。ミスマッチをつなげる上で相変わらずないものは、ユーザがレスポンスを入力する手段と、それをレスポンステーブルに放り込むことです。

mismatch_topic テーブル全体には 25 の
← 項目が 5 つのカテゴリに分類されて存在しています…つまり「5 次元のチャンス！」があることになります。

topic_id	name	category
1	タトゥー	ファッション
2	ゴールドチェーン	ファッション
3	ピアス	ファッション
4	カウボーイブーツ	ファッション
5	ロン毛	ファッション
6	リアリティー番組	エンタメ
7	プロレス	エンタメ
8	ホラー映画	エンタメ
9	イージーリスニング	エンタメ
10	オペラ	エンタメ
11	寿司	食べ物
12	SPAM	食べ物
13	激辛	食べ物
14	ピーナッツバター・バナナサンド	食べ物
15	マティーニ	食べ物
16	ハワード・スターイン	有名人
17	ビルゲイツ	著名人
18	バーバラ・ストライサンド	有名人
19	ヒュー・ヘフナー	有名人
20	マーサ・スチュワード	有名人
21	ヨガ	趣味
22	ウェイト・リフティング	趣味
23	ルーピックキューブ	趣味
24	カラオケ	趣味
25	ハイキング	趣味

脳力発揮

どうすればこのカテゴリと項目のリストから質問を作り出して、「好き/嫌い」のレスポンスをユーザから得ることができますか？

ミスマッチのアンケートを作る

各ミスマッチ項目に対する「好き／嫌い」のレスポンスをユーザから得るには正確にはどうすれば良いのでしょうか？その答えはアンケートフォームで、mismatch_topocテーブルにある各項目について、ユーザに「好き」か「嫌い」かを選ばせるようにすればよいのです。このフォームはデータベースに入っているレスポンスから直接作ることができます。そしてその結果もデータベースに戻してあげればよいのです。実際、アンケートフォームを設計するには、mismatch_responseテーブルからレスポンスを読み込み、レスポンスを書き込むことに他なりません。以下がアンケートの頂上と、そこに至るまでの各ステップです。

① INSERTを使って空のレスポンス行をデータベースに追加し、最初にユーザがフォームにアクセスできるようにする。

今から作ろうとしているアンケートのフォームはmismatch_responseテーブルのデータから作りますから、ユーザがまだ何もレスポンスを打ち込んでいないときにもテーブルが必要です。つまりmismatch_responseテーブルには、ユーザがアンケートに最初に答えるまではレスポンスが空であるという元ネタの状態が必要なのです。この行のレスポンスカラムは空ですから、「好き」も「嫌い」もどちらのラジオボタンも最初にユーザが目にすることにはチェックされていません。

② UPDATEを使ってレスポンスフォームに入力されたユーザのレスポンスに基づいてレスポンス行を変更する。

ユーザがアンケートのフォームを「提出」したら、その個々のレスポンスをデータベースに反映させなければなりません。ただし、ユーザがチェックしたラジオボタンの項目だけをアップデートすることが重要です。言い換えれば、データベースは答えられたレスポンスについてだけ知っている必要があるということです。

③ SELECTを使ってアンケートのフォームを作るのに必要なリクエストデータを取ってくる。

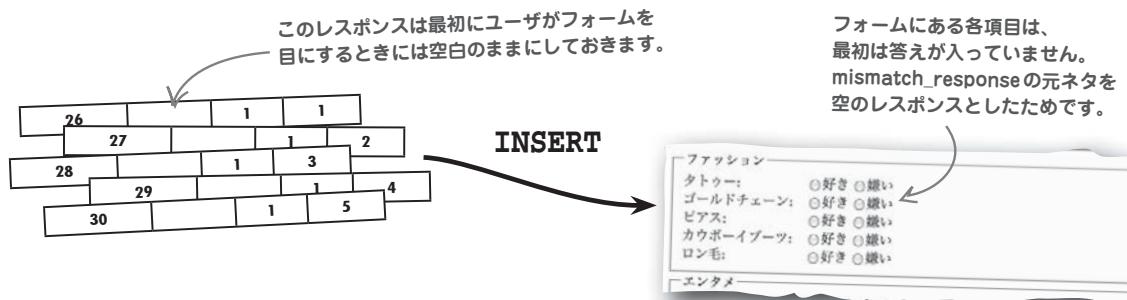
アンケートのフォームを作るには、ログインしたユーザに対してすべてレスポンスをしてもらう必要があります。その上、各レスポンスの項目名とカテゴリ名を探して、フォームに表示しなければなりません。これらの名前が入っているのは、mismatch_topocテーブルであって、mismatch_responseテーブルではありません。

④ HTMLのアンケートフォームをレスポンスデータから作る。

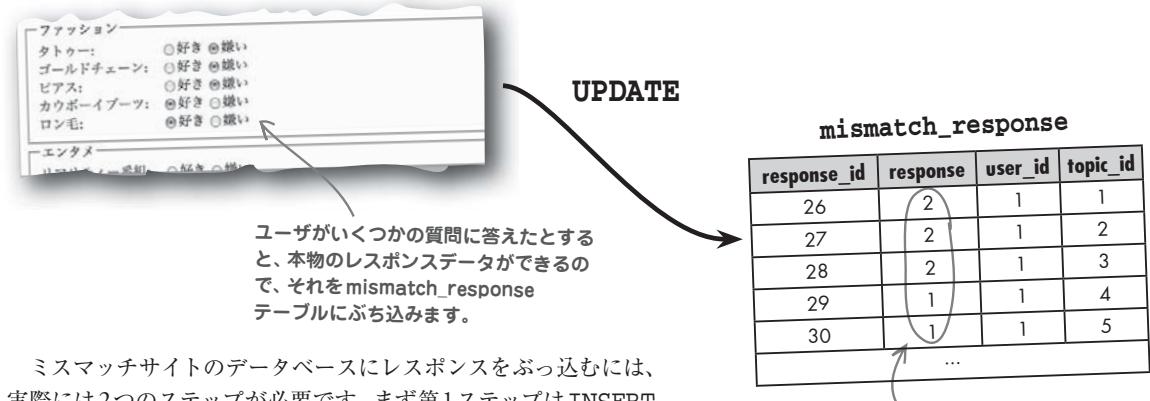
レスポンスデータは手元にそろったので、HTMLのアンケートフォームを作り、たくさんの入力フィールドを用意し、ユーザのレスポンスに応じて「好き」と「嫌い」を正しくチェックしてもらえるようにします。

データベースにレスポンスを入れる

アンケートのフォームをまず始めに作る方が良さそうに思えるかもしれません。実はフォームは mismatch_response テーブルの中にあるレスポンスデータに依存しています。つまり最初はこっちです。まず mismatch_response テーブルに答えがないレスポンスの行という元ネタの状態が必要です。これはユーザが最初にアンケートにアクセスしたときの状態です。これでアンケートのフォームを mismatch_response テーブルから生成することができます。ユーザが実際にレスポンスをどんなレスポンスをするかどうか、まだ心配しなくても大丈夫です。



つまりアンケートのフォームという観点では、mismatch_response テーブルのデータ行が常に存在し、フォームに各項目に関する質問があることになります。ユーザがアンケートのフォームを「提出」すると、フォームの各レスポンスに対して、単にデータ行をアップデートすればよいことになります。



ミスマッチサイトのデータベースにレスポンスをぶち込むには、実際には2つのステップが必要です。まず第1ステップは INSERT で、各ユーザー用に場所だけを確保します。最初に空のレスポンスを追加したら、それ以降のアンケートによる変更が、第2ステップで、SQL の UPDATE を通して処理します。

データベースのレスポンスはアンケートのフォームに入力されたユーザのレスポンスにマッチさせて上書きします。



PHP&MySQLマグネット

以下のコードは、mismatch_responseテーブルに空のレスポンスを突っ込むためのものです。ユーザーが最初にフォームを目にするときの状態です。さらにこのコードでは、ユーザーが何らかの変更をしてフォームを「提出」した場合のアップデートも行います。ところが、残念なことにコードの一部が落っこちてしまったので、何とかしなければいけません。下のマグネットを使って失ってしまったコードを修正して下さい。

```

...
    ユーザがまだアンケートに回答していない場合、データベースには空のレスポンスを挿入

// If this user has never answered the questionnaire, insert empty responses into the database
$query = "SELECT * FROM mismatch_response WHERE user_id = '" . $_SESSION['user_id'] . "'";

$data = mysqli_query($dbc, $query);
if (.....($data) == 0) {
    // First grab the list of topic IDs from the topic table 最初にtopicテーブルから項目IDのリストを取得
    $query = "SELECT.....FROM mismatch_topic ORDER BY category_id, topic_id";
    $data = mysqli_query($dbc, $query);
    $topicIDs = array();
    while ($row = mysqli_fetch_array($data)) {
        array_push($topicIDs, $row['topic_id']);
    }
    // Insert empty response rows into the response table, one per topic
    foreach ($topicIDs as $topic_id) {
        $query = ".....mismatch_response " .
            ".....,..... VALUES ('" . $_SESSION['user_id']. "", '$topic_id')";
        mysqli_query($dbc, $query);
    }
    アンケートフォームが「提出」された場合、フォームのレスポンスをデータベースへ書き出す
}

// If the questionnaire form has been submitted, write the form responses to the database
if (isset($_POST['submit'])) {
    アンケートのレスポンス行をresponseテーブルへ書き出す
    // Write the questionnaire response rows to the response table
    foreach ($_POST as $response_id => $response) {
        $query = ".....mismatch_response.....response = '$response' " .
            "WHERE .....= '$response_id'";
        mysqli_query($dbc, $query);
    }
    echo '<p>アンケート結果を保存しました。</p>';
}
...

```





PHP&MySQL マグネットの答え

以下のコードは、mismatch_responseテーブルに空のレスポンスを突っ込むためのものです。ユーザが最初にフォームを目にするときの状態です。更にこのコードでは、ユーザが何らかの変更をしてフォームを「提出」した場合のアップデートも行います。ところが、残念なことにコードの一部が落っこちてしまったので、何とかしなければいけません。下のマグネットを使って失ってしまったコードを修正して下さい。

```

...
// If this user has never answered the questionnaire, insert empty responses into the database
$query = "SELECT * FROM mismatch_response WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
if (mysqli_num_rows ($data) == 0) { ←
    // First grab the list of topic IDs from the topic table
    $query = "SELECT [topic_id] FROM mismatch_topic ORDER BY category_id, topic_id";
    $data = mysqli_query($dbc, $query);
    $topicIDs = array();
    while ($row = mysqli_fetch_array($data)) {
        array_push($topicIDs, $row['topic_id']);
    }
    // Insert empty response rows into the response table, one per topic
    foreach ($topicIDs as $topic_id) {
        $query = ". INSERT INTO mismatch_response " . ←
            " ( [user_id] , [topic_id] ) VALUES ('" . $_SESSION['user_id'] . "' , '$topic_id') ";
        mysqli_query($dbc, $query);
    }
}
// If the questionnaire form has been submitted, write the form responses to the database
if (isset($_POST['submit'])) {
    // Write the questionnaire response rows to the response table
    foreach ($_POST as $response_id => $response) {
        $query = ". UPDATE mismatch_response. [SET] ...response = '$response' " .
            " WHERE [response_id] = '$response_id'";
        mysqli_query($dbc, $query);
    }
    echo '<p>アンケート結果を保存しました。</p>';
}
...

```

問い合わせ文が0行を返したかどうかを調べて
います…データがありません！

レスポンスの空配列を作るために、
まず始めに項目テーブルからすべての
項目を取ってくる必要があります。

レスポンス行はこの時点では
「未回答」です。ユーザはこの
時点ではまだ「好き」も「嫌い」も
フォーム上で選んでいません。

ユーザがフォームを「提出」した際の
すべての変更は、レスポンステーブルの
レスポンスカラムに対するものですから、
ここをアップデートします。

素朴な疑問に答えます

Q : array_push() いう関数は何するん?まだ見たことないんとちゃう?

A : まだです。今まで必要なかったのは、動的に一度に1要素ずつ増えていく配列を作る必要がなかったからです。array_push() 関数は、新しい要素を1つ取って、配列の最後にくっつけます。その結果として配列は1だけ大きくなります。反対側のページにあるミスマッチサイトのコードでは、array_push() を使って、mismatch_topic テーブルから項目IDの配列を作っています。この配列は次にmismatch_response テーブルに空白のレスポンスを、各項目について1つずつ突っ込むために使います。

① INSERTを使って空のレスポンス行をデータベースに追加し、最初にユーザがフォームにアクセスできるようにする。

終了

バーン！正に一石二鳥です。
しかもアンケート用スクリプトも
ほぼ半分できています。

② UPDATEを使ってレスポンスフォームに入力されたユーザの
レスポンスに基づいてレスポンス行を変更する。

終了

ただし、まだ2ステップ
残っています。その後、
ミスマッチアンケートの
相性診断の部分を作る
ことができます…

③ SELECTを使ってアンケートの
フォームを作るのに必要なリク
エストデータを取ってくる。

④ HTMLのアンケートフォームをレスポンス
データから作る。

データでフォームを駆動する

Web フォームを使ってテキストフィールドやプルダウン(選択リスト)やラジオボタンなどにユーザが入力したデータを取ってくること自体はちっとも新しくありません。しかし PHP を使ってデータベースのデータから HTML のフォームを作り出してしまう、ということはまだ当たり前ということはないでしょう。ミスマッチサイトにおける考え方は、HTML のアンケートフォームをレスポンスデータから動的に作り出すというものです。ミスマッチサイトのアンケート用スクリプトは、レスポンスデータがすでに存在しているという仮定をおいています。この仮定の下、`mismatch_response` テーブルにあるデータからフォームを作ることができます。この仮定が安全なものであることは、最初にユーザがフォームを見るまでは、空のレスポンスを入れておくというコードを書いたことにより保障されています。

データ駆動型の
フォームは MySQL
データベースのデータ
に依存して HTML
のフォームフィールド
を作り出します。

mismatch_response

response_id	主キー
response	
user_id	外キー
topic_id	外キー

response_id 主キーを使って、HTML のフォームフィールドを一意に識別し、各フィールドとデータベース行とを対応付けます。

HTML フォームのコードは、`mismatch_response` テーブルのデータから作り出します。

checked 属性で、ラジオボタンを選んだかどうかをコントロールします。

このフォームでユーザが各項目でレスポンスとして何を選んだかを反映させます。

```

<form method="post" action="">
  <p>How do you feel about each topic?</p>
  <fieldset>
    <legend> ファッション </legend>
    <label for="76">タトゥー:</label><input type="radio" id="76" name="76" value="1" checked="checked" />好き
    <input type="radio" id="76" name="76" value="2" />嫌い<br />
    <label for="77">ゴールドチェーン:</label><input type="radio" id="77" name="77" value="1" checked="checked" />好き
    <input type="radio" id="77" name="77" value="2" />嫌い<br />
    <label for="78">ビアス:</label><input type="radio" id="78" name="78" value="1" checked="checked" />好き
    <input type="radio" id="78" name="78" value="2" />嫌い<br />
    <label for="79">カウボーイブーツ:</label><input type="radio" id="79" name="79" value="1" checked="checked" />好き
    <input type="radio" id="79" name="79" value="2" />嫌い<br />
    <label for="80">ロン毛:</label><input type="radio" id="80" name="80" value="1" checked="checked" />好き
    <input type="radio" id="80" name="80" value="2" />嫌い<br />
  </fieldset>
  <fieldset>
    <legend> エンタメ </legend>
    ...
  </fieldset>
</form>

```

ミスマッチ：アンケート

各項目についてどう思うか回答してください。

ファッション

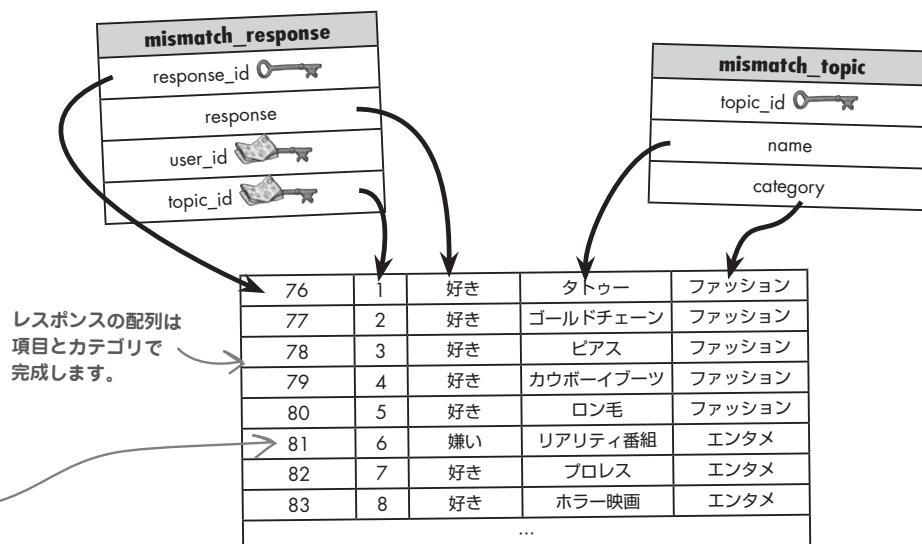
- タトゥー: 好き 嫌い
- ゴールドチェーン: 好き 嫌い
- ビアス: 好き 嫌い
- カウボーイブーツ: 好き 嫌い
- ロン毛: 好き 嫌い

エンタメ

- アリティーライブ: 好き 嫌い
- プロレス: 好き 嫌い
- ホラー映画: 好き 嫌い



ミスマッチサイトのレスポンス用アンケートは、ユーザのレスポンスから作られて、mismatch_response テーブルに突っ込まれます。HTMLフォームのコードを生成するには、このレスポンスを読み込んで、mismatch_topic テーブルから各レスポンスの項目名とカテゴリ名を探し当てる必要があります。以下のコードは、項目とカテゴリからなるレスポンスの配列を作るために、2つの問い合わせ文を実行します。第1の問い合わせ文は、ユーザ用にレスポンスを取ってくるものです。第2の問い合わせ文は各問い合わせ文項目名とカテゴリ名とを探します。問題なのは、コードの一部が欠けていることです…空白を埋めてコードが動くようにして下さい！



データベースからレスポンスデータを取得し、フォームを生成

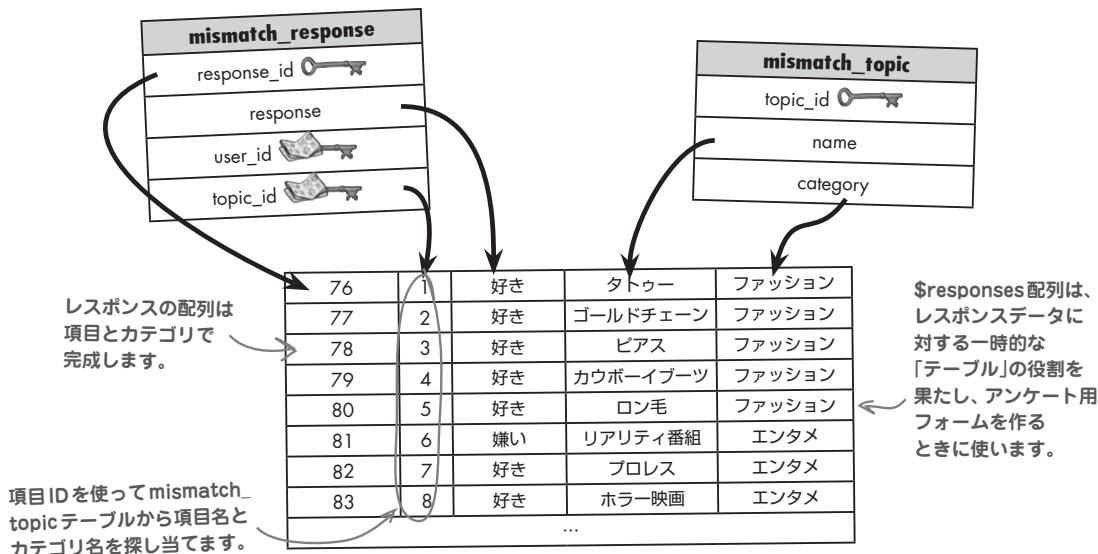
```
// Grab the response data from the database to generate the form
$query = "SELECT response_id, topic_id, response FROM mismatch_response " .
    "WHERE user_id = '" . $_SESSION['user_id'] . "' ";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) { レスポンスの項目名を topic テーブルから探索
    // Look up the topic name for the response from the topic table
    $query2 = ".....".
        "WHERE topic_id = '" . $row['topic_id'] . "' ";
    $data2 = mysqli_query($dbc, .....);
    if (mysqli_num_rows(.....) == 1) {
        $row2 = mysqli_fetch_array($data2);
        $row['topic_name'] = .....
        $row['category_name'] = .....
        array_push($responses, $row);
    }
}
```

このPHP関数は、問い合わせ文の結果として何行のデータが返ってきたのか教えてくれます。



エクササイズ の答え

ミスマッチサイトのレスポンス用アンケートは、ユーザのレスポンスから作られて、mismatch_responseテーブルに突っ込まれます。HTMLフォームのコードを生成するには、このレスポンスを読み込んで、mismatch_topicテーブルから各レスポンスの項目名とカテゴリ名を探し当てる必要があります。以下のコードは、項目とカテゴリからなるレスポンスの配列を作るために、2つの問い合わせ文を実行します。第1の問い合わせ文は、ユーザ用にレスポンスを取ってくるものです。第2の問い合わせ文は各問い合わせ文項目名とカテゴリ名とを探します。問題なのは、コードの一部が欠けていることです…空白を埋めてコードが動くようにして下さい！



```
// Grab the response data from the database to generate the form
$query = "SELECT response_id, topic_id, response FROM mismatch_response ";
    "WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
    // Look up the topic name for the response from the topic table
    $query2 = ".SELECT name, category FROM mismatch_topic .";
        "WHERE topic_id = '" . $row['topic_id'] . "' ";
    $data2 = mysqli_query($dbc, $query2..);
    if (mysqli_num_rows($data2..) == 1) {
        $row2 = mysqli_fetch_array($data2..);
        $row['topic_name'] = $row2['name'];
        $row['category_name'] = $row2['category'];
        array_push($responses, $row);
    }
}
array_push() 関数で配列の最後に項目を追加(プッシュ)します。
```

レスポンスデータが実際に存在するかどうか確認します。

非常に重要なことなのですが、新しい変数を使って2番目の(内側の)問い合わせ文を実行します。そうしないと最初の問い合わせ文が壊れてしまいます。

項目名とカテゴリ名をレスポンス配列に追加します。2番目の問い合わせ文のデータをぶち込めばよいのです。

③ SELECTを使ってアンケートのフォームを作るのに必要なリクエストデータを取ってくる。

終了



ユーザのレスポンスを
データベースにテキストのまま
突っ込んでいるよね？「好き」とか「嫌い」とか。
そんなことして効率的には大丈夫なの？

ダメでも大丈夫もあります。だからこそ可能な限り効率的なデータ型でMySQLデータベースにデータを突っ込むことが大事なのです。

ミスマッチサイトのレスポンスについて言えば、真偽値を答えとして使うのが良さそうに思えます。なぜなら答えは常に「好き」か「嫌い」かのどちらか一方だからです。ところが実際には、第3の値「わからない」があると便利です。アプリケーションに対して、ユーザが特定の項目についてまだ答えていない、ということにしておくのです。つまり本当に必要なのは、与えられたレスポンスに対して取りうる3つの値を見失わないようにすることです。この種の格納領域に関する問題には、TINYINTのような数字が理想的です。そうすれば異なる数値を起こりうるレスポンスに割り当てて表現するだけで良いのです。



「わからない」 = 0



「好き」 = 1



「嫌い」 = 2

データに必要な領域を最小にするということは、データベースの設計において非常に重要な部分を占めています。この場合、ミスマッチサイトの微妙かつ重要な部分と言えます。この数値によるレスポンスは、ミスマッチサイトのアンケート用のフォームフィールドを作る上で直接的な役割を演じてくれます。

自分で考えてみよう



「嫌い」のラジオボタンについては
今は気にしなくとも大丈夫です。
全く同様に作ることができます。

以下のコードは、ミスマッチサイトのレスポンス配列をループします。さっそく作った配列です。ループの結果HTMLのフォームフィールドに対して「好き」のラジオボタンができます。欠けているコードを埋めて、レスポンスが「好き(1)」であれば、フォームフィールドが初期値としてチェックされているようにして下さい。また<input>タグも対応する値を設定して下さい。

```
foreach ($responses as $response) {
    ...
    if (.....) {
        echo '<input type="radio" name="' . $response['response_id'] .
            '" value=..... checked=..... /> 好き ';
    }
    else {
        echo '<input type="radio" name="' . $response['response_id'] .
            '" value=..... /> 好き ';
    }
}
```

自分で考えてみよう の答え

以下のコードは、ミスマッチサイトのレスポンス配列をループします。さっき作った配列です。ループの結果HTMLのフォームフィールドに対して「好き」のラジオボタンができます。欠けているコードを埋めて、レスポンスが「好き(1)」であれば、フォームフィールドが初期値としてチェックされているようにして下さい。また<input>タグも対応する値を設定して下さい。

```
「好き」のラジオボタンは、レスポンスの
値に従ってチェックされます(データ
ベースでは1が「好き」を表します)。
foreach ($responses as $response) {
    ...
    if ( $response[ 'response' ] == 1 ) {
        echo '<input type="radio" name="' . $response['response_id'] .
            '" value="1" checked="checked" /> 好き ' ;
    }
    else {
        echo '<input type="radio" name="' . $response['response_id'] .
            '" value="1" /> 好き ' ;
    }
}
<input> タグの値を1に設定して
おけば、フォームが「提出」された
とき、レスポンスをデータベースに
突っ込みやすくなります。
```

このレスポンスが「好き(1)」に設定
されると、ラジオボタンのchecked
属性をcheckedに設定してチェック
をONにします。

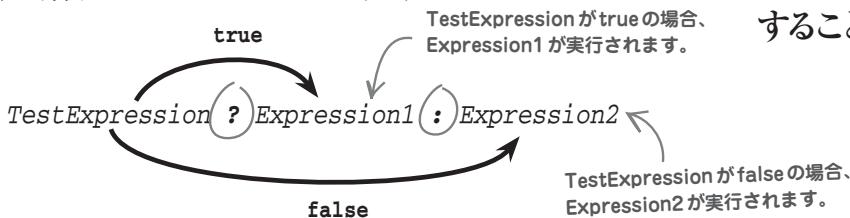
checked="checked"という記述をなく
して、ラジオボタンのチェックはOFFに
します。レスポンスが「好き(1)」に設定
されない場合です。

```
foreach ($responses as $response) {
    ...
    if ( $response[ 'response' ] == 2 ) {
        echo '<input type="radio" name="' . $response['response_id'] .
            '" value="2" checked="checked" /> 嫌い ' ;
    }
    else {
        echo '<input type="radio" name="' . $response['response_id'] .
            '" value="2" /> 嫌い ' ;
    }
}
```

心配性の方のために「嫌い」のラジオボタン用の
コードです。全く同様に動きますが、ほんの
ちょっと違うレスポンス探すだけです。しかし、
実際には「好き」と「嫌い」の両方のラジオボタンを
少ないコードで書く、スッキリしたやり方が
あります…

効率について言えば…

データベースの効率だけ考えれば済むということはありません。効率的なコーディングについても考える必要があります。これはいろいろな場面で現れます。その1つはPHP言語のif-else文を単純化することにより御利益を得るというものです。**3項演算**は、単純なif-else文をコーディングする手軽な方法で、コードがコンパクトになります。



3項演算を使って
? :
if-else文をもっと
コンパクトにコーディング
することができます。

3項演算は実際にはif-else文の略記法に過ぎません[†]。3項演算を使ってif-else文を単純化すると、特に変数に代入する場合やif条件式に応じてHTMLコードを作る場合などに便利です。以下に、さっきと同じ「好き」のラジオボタンを生成するコードを、3項演算子で書き直してみます。

```
echo '<input type="radio" name="' . $response['response_id'] . '" value="1" ' .
      ( $response['response'] == 1 ? 'checked="checked"' : '' ) . ' />好き ';
```

この真偽値テストで、3項演算子の
出力をコントロールします。

\$response['response']に入っているレスポンスの値が、1と等しければ、checked属性が<input>タグの一部として作られ、最終的に以下のように「好き」にチェックされたラジオボタンとなります。

```
<input type="radio" name="279" value="1" checked="checked" />好き
```

<input>タグのこの部分の
コードが3項演算子によって
コントロールされます。

ロン毛: 好き 嫌い

<input>タグのchecked
属性は、これでif文の
代わりに3項演算子を
使って生成されます。

一方、レスポンスの値が1以外の値であれば、<input>タグのchecked属性は作られませんから、「好き」のラジオボタンはチェックされません。

[†] 訳注：原著のこの説明は厳密には間違っています。if-elseは「文」ですが、3項演算は「式」です。式の中に文を書くことはできませんから、if-else文を3項演算で置き換えることは一般にはできません。ほとんど同等のものであり、置き換えが可能な場合もありますが、「単なる略記法」ではありません。

ミスマッチサイトのアンケート用フォームを作る

これでミスマッチサイトのアンケート用フォームパズルを解くピースは全部そろいました。さっき作ったレスポンス配列(responses)を使って、HTMLフォーム全体を作ります。この配列はmismatch_responseテーブルから現在のユーザのレスポンスを取り出して作ったということを思い出して下さい。先へ進んで、アンケート生成コードを完成版のquestionnaire.phpスクリプトを使って見てみましょう。



questionnaire.php

```

<?php
    // Start the session セッションの開始
    require_once('startsession.php');
    // Insert the page header テンプレートファイルをインク
    // ルードしてセッションを開始し、  
ページヘッダを表示します。
    $page_title = 'アンケート'; ページヘッダの挿入
    require_once('header.php');
    require_once('appvars.php');
    require_once('connectvars.php'); 先に進める前にユーザがログインしていることを確認
    // Make sure the user is logged in before going any further.
    if (!isset($_SESSION['user_id'])) {
        echo '<p class="login"><a href="login.php">ログイン</a>後のこのページにアクセスすることができます.</p>';
        exit();
    }
    // Show the navigation menu 案内用メニューの表示
    require_once('navmenu.php');
    // Connect to the database データベースへの接続
    $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME); ユーザがまだアンケートに回答していない場合、データベースには空のレスポンスを挿入
    // If this user has never answered the questionnaire, insert empty responses into the database
    $query = "SELECT * FROM mismatch_response WHERE user_id = '" . $_SESSION['user_id'] . "'";
    $data = mysqli_query($dbc, $query);
    if (mysqli_num_rows($data) == 0) {
        // First grab the list of topic IDs from the topic table 最初にtopicテーブルから項目IDのリストを取得
        $query = "SELECT topic_id FROM mismatch_topic ORDER BY category_id, topic_id";
        $data = mysqli_query($dbc, $query);
        $topicIDs = array();
        while ($row = mysqli_fetch_array($data)) {
            array_push($topicIDs, $row['topic_id']);
        }
        // Insert empty response rows into the response table, one per topic response テーブルへ空のレスポンス行を項目ごとに挿入
        foreach ($topicIDs as $topic_id) {
            $query = "INSERT INTO mismatch_response (user_id, topic_id) VALUES ('" . $_SESSION['user_id'] . ',
                "' . $topic_id . ')";
            mysqli_query($dbc, $query);
        }
    }
    // If the questionnaire form has been submitted, write the form responses to the database アンケートフォームが「提出」された場合、フォームのレスポンスをデータベースへ書き出す
    if (isset($_POST['submit'])) {
        // Write the questionnaire response rows to the response table
        foreach ($_POST as $response_id => $response) { アンケートのレスポンス行を response テーブルへ書き出す
            $query = "UPDATE mismatch_response SET response = '$response' ";
        }
    }
}

```

1

2

```

        "WHERE response_id = '$response_id'";
        mysqli_query($dbc, $query);
    }
    echo '<p>アンケート結果を保存しました。</p>';
}

// Grab the response data from the database to generate the form レスポンスデータをデータベースから取得し、フォームを生成
$query = "SELECT response_id, topic_id, response FROM mismatch_response WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
    // Look up the topic name for the response from the topic table レスポンスの項目名をtopicテーブルから探索
    $query2 = "SELECT name, category FROM mismatch_topic WHERE topic_id = '" . $row['topic_id'] . "' . ";
    $data2 = mysqli_query($dbc, $query2);
    if (mysqli_num_rows($data2) == 1) {
        $row2 = mysqli_fetch_array($data2);
        $row['topic_name'] = $row2['name'];
        $row['category_name'] = $row2['category'];
        array_push($responses, $row);
    }
}
mysql_close($dbc);      response配列をループし、アンケートフォームを生成

```

```

// Generate the questionnaire form by looping through the response array
echo '<form method="post" action="" . $_SERVER['PHP_SELF'] . '>';
echo '<p>各項目についてどう思うか回答して下さい。</p>';
$category = $responses[0]['category_name'];
echo '<fieldset><legend>' . $responses[0]['category_name'] . '</legend>';
foreach ($responses as $response) {
    // Only start a new fieldset if the category has changed
    if ($category != $response['category_name']) { カテゴリが変わった場合のみ新しいfieldsetに変更
        $category = $response['category_name'];
        echo '</fieldset><fieldset><legend>' . $response['category_name'] . '</legend>';
    }
    // Display the topic form field
    echo '<label ' . ($response['response'] == NULL ? 'class="error"' : '') . ' for="'
        . $response['response_id'] . '">' . $response['topic_name'] . ':</label>';
    echo '<input type="radio" id="' . $response['response_id'] . '" name="'
        . $response['response_id'] . '" value="1" ' . ($response['response'] == 1 ? 'checked="checked"' : '') . ' />好き ';
    echo '<input type="radio" id="' . $response['response_id'] . '" name="'
        . $response['response_id'] . '" value="2" ' . ($response['response'] == 2 ? 'checked="checked"' : '') . ' />嫌い<br />';
}
echo '</fieldset>';
echo '<input type="submit" value="アンケートを保存" name="submit" />';
echo '</form>'; ページフッタの挿入
// Insert the page footer
require_once('footer.php');
?>

```

それぞれのecho文でラジオボタンを作ります。1つは「好き」でもう1つは「嫌い」です。

最初のレスポンスに対する
カテゴリを取ってきて、準備が
できたらループに入れます。

各カテゴリで
fieldsetを作り、
項目をまとめて
グループ化します。

ここで3項演算子
を使って、未回答
項目に対して
ラベルの形式を
変更します。

各項目はラベルに
引き続いで「好き」
と「嫌い」のラジオ
ボタンで作って
あります。

終了
④ HTMLのアンケートフォームをレスポンス
データから作る。



試運転

新しく作ったミスマッチサイトのアンケートを試してみる

ミスマッチサイトを修正し、新しくアンケート用スクリプトを使えるようにします。(アプリケーションはWebサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードすることもできます。)このためには、新しく questionnaire.php を作り、「アンケート」メニュー項目を navmenu.php スクリプトに追加して、ユーザがアンケートにアクセスできるようにする必要があります。

スクリプトをWebサーバにアップロードしたら、ミスマッチサイトのメインページ(index.php)をWebブラウザで開きます。ログインして、「アンケート」メニュー項目をクリックし、アンケートにアクセスして下さい。どの項目にも答えていないことになっています。これがアンケートのページを最初に開いたためです。レスポンスに答えて、フォームを「提出」してみましょう。メインページに戻って、再びアンケートページを開くと、今度はレスポンスがデータベースから正しく反映されていることを確認して下さい。

アンケート用スクリプトでは、ユーザに
「好き／嫌い」の質問に答えてもらい、
結果をデータベースにぶち込みます。

タトゥー:	<input checked="" type="radio"/> 好き <input type="radio"/> 嫌い
ゴールドチェーン:	<input checked="" type="radio"/> 好き <input type="radio"/> 嫌い
ピアス:	<input checked="" type="radio"/> 好き <input type="radio"/> 嫌い
カウボーイブーツ:	<input checked="" type="radio"/> 好き <input type="radio"/> 嫌い
ロン毛:	<input checked="" type="radio"/> 好き <input type="radio"/> 嫌い

リアリティー番組:	<input type="radio"/> 好き <input checked="" type="radio"/> 嫌い
プロレス:	<input checked="" type="radio"/> 好き <input type="radio"/> 嫌い
ホラー映画:	<input checked="" type="radio"/> 好き <input type="radio"/> 嫌い

フォームにある質問項目は、
データベースから動的に
生成されます。自分で新しい
項目を追加するとフォームが
変わります。

ダウンロードして下さい！

ミスマッチサイトのアプリケーション用のすべてのソースコードは以下のWebサイトからダウンロードすることができます。

<http://www.oreilly.co.jp/books/9784873114446/>

素朴な疑問に答えます

Q: 「好き」のラジオボタンのコードは、どうして3項演算の結果が文字列だということが分かるのでしょうか？

A: 3項演算子は、テスト式の値(trueまたはfalse)に応じて、コロンの両側にある2つの式[†]のうち1つの値になります。2つの式が文字列ならば、3項演算の結果は文字列となります。このためこの演算子は手軽に使えるのです。代入や連結の真ん中に直に突っ込むことができます。

Q: 3項演算を使うとスクリプトを速くすることができるのでしょうか？

A: 恐らくできません。3項演算は、スタイル上の効率をコードに付与してくれるのであって、パフォーマンス上の効率は上がりません。つまりスクリプトコードを表面上少なくすことができるという意味です。場合によっては、完全なif-else文よりも3項演算を使った方が簡潔に書ける場合があります。しかし論理的にはこれら2つは等価なものです。とはいっても3項演算に偏りすぎるというのはよくありません。コードによってはわかり難くなってしまうかもしれませんから、複雑なif-else文を3項演算で書き直そなどとはしないことです。考え方としては、if-else文の代わりに3項演算を使うことで、混乱を来すことなく、確かにコードを単純化できる場合に限り使うようにするとよいでしょう。よくあるケースは、変数に代入する値や式に突っ込む値を3項演算で選択的にコントロールする場合です。ミスマッチサイトのラジオボタンの場合で言うと、後者のケースが使われて、HTMLの属性(checked)に選択的に挿入するものをコントロールしています。



① 新しい項目をmismatch_topicテーブルに以下のSQL文で追加します。

```
INSERT INTO mismatch_topic
(name, category) VALUES
('仮想ギター', '趣味')
```

② 以下のSQL文で、mismatch_responseテーブルからすべてのデータを消します。

```
DELETE FROM mismatch_response
```

③ ミスマッチサイトのアプリケーションのアンケートで新しい項目を見ます。

④ 新しい項目にレスポンスを書き、フォームを「提出」して保存したレスポンスをチェックします。

Q: ミスマッチサイトのアンケート用フォームを作り出すことができるのでしょうか？まだユーザが何もレスポンスしていないときです。

A: 良い質問です。アンケート用フォームには可能性として2つのシナリオを考える必要があります。ユーザが最初にアンケートに答える、またはユーザは一度アンケートに答えたが、後で答えを書き換えようとする。最初のシナリオでは、まだレスポンスはありませんから、mismatch_responseテーブルには、まだこのユーザのデータがありません。しかし、それでも動的にフォームを作らなければなりません。このためだけならmismatch_topicテーブルを使ってフォームを作っても良かったかもしれません。しかし、これは第2のシナリオではちゃんと動きません。なぜなら今回はフォームをユーザに依存した「好き／嫌い」レスポンスに基づいて作らなければならないからです。「好き」と「嫌い」のラジオボタンは、フォームの一部として作られるということを思い出して下さい。ここで問題が起ります。フォームを作ろうとしているコードは、ユーザがアンケートに答えたかどうかに依存して全く異なるということです。それだけでなく、ユーザが問題の一部にしか答えたかった場合についてはどうなるでしょうか？これはちょっと面倒です。ミスマッチサイトで採用したのは、mismatch_responseテーブルに「未回答」というレスポンスをあらかじめ設定しておき、ユーザが最初にアンケートにアクセスしたときの状態にしておくというものです。これでユーザがレスポンスをしたかどうかや、どれか特定の項目にのみレスポンスをしたかということで、異なるフォームを作るという心配がいりません。それでもフォームを作るコードはそんなに簡単ではありません。しかし、このアプローチを採用しなかった場合に比べればシンプルなものになっています。

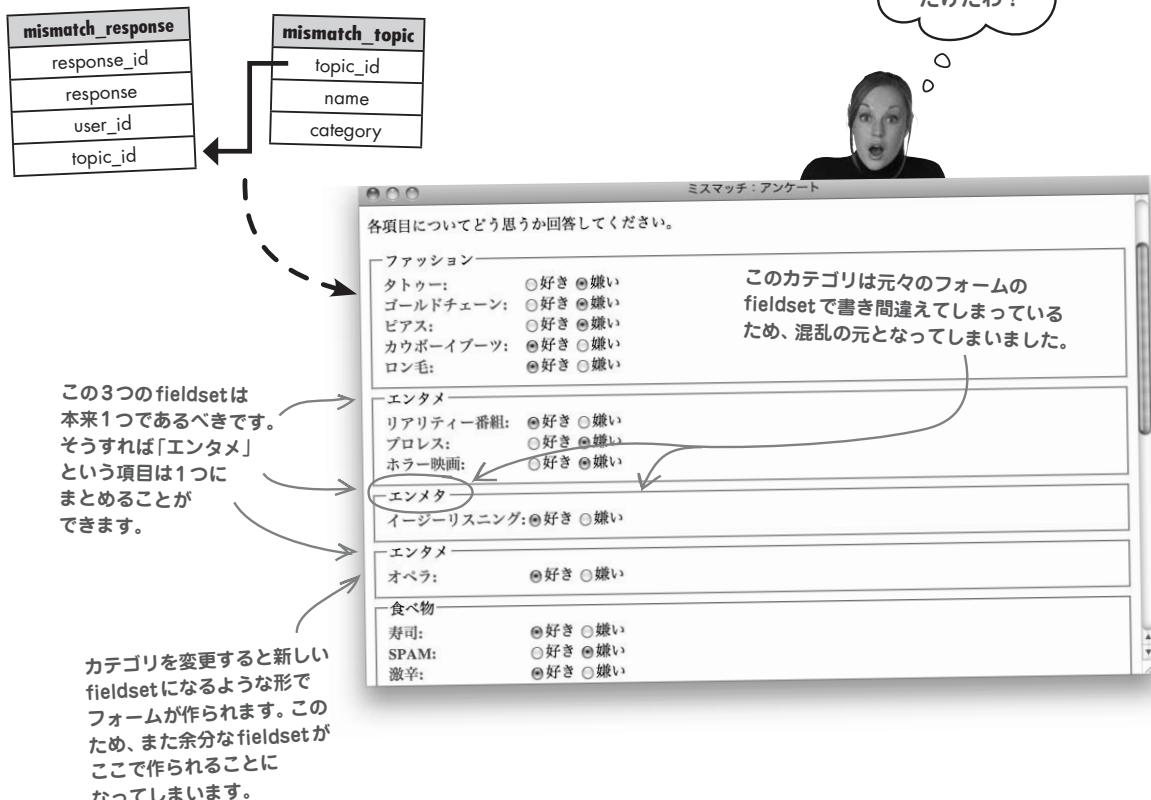
コードを単純化するため、ミスマッチサイトでは、新しい項目を自動的に調節することはありません。少なくともアンケートに既に回答したユーザがいない間はこれで大丈夫です。ですから新しい項目を追加したら、mismatch_responseテーブルを空にする必要があります。

[†] 訳注：この回答の後半部分は正しくありません。3項演算を代入や連結の真ん中に突っ込めるのは手軽だからではなく「式」だからです。「式」の中に「式」を書くことはできますが、「式」の中に「文」は書けません。例えば代入の右辺にif文やwhile文を書くことはできません。

データがフォームを駆動している

いろいろやつきましたが、ミスマッチサイトのアプリケーションはデータベースに突っ込んであるレスポンスからアンケートを動的に作り出すようになりました。このため、データベースに何らかの変更を施すと、それは自動的にフォームに反映されるようになったということになります。これが、Webアプリケーションのユーザインタフェースをデータベース駆動で作るということの意味です。でも変なデータを入れてしまったら何が起こるのでしょうか？

こんな
フォームなんて
本当に混乱する
だけだわ！



mismatch_topic

topic_id	name	category
...		
8	ホラー映画	エンタメ
9	イージーリスニング	エンタメ
10	オペラ	エンタメ
11	寿司	食べ物
12	SMAP	食べ物
13	激辛	食べ物
14	ピーナッツバター・バナナサンド	食べ物
15	マティーニ	食べ物
16	ハワード・スター	有名人
17	ビルゲイツ	著名人
18	バーバラ・ストライサン	有名人
...		

森川氏 沢田さん Jちゃん

この書き間違えのせいで
カテゴリに問題が起っている
ことはもうわかっています…

…それにここにもう1つあります。
同様にアンケート用フォームに
fieldsetの問題を引き起こします。

森川氏：そんなの簡単だよ。mismatch_topicテーブルのカテゴリの名前を正しく書き直せばいいだけだろ。

Jちゃん：でもカテゴリを書き間違えているところは2つ以上あるんだよ。それに思うんだけど、何で同じカテゴリ名を何度も突っ込む必要があるのかがよく分からない。

沢田さん：そうよね。データベースのスキーマを設計するときにデータをダブって持つことの問題点は知っていたはずよ。それなのに今カテゴリ名が腐るほどダブっているわ。それだけじゃなくて、名前がいくつも間違っているのよ。

森川氏：そうか。カテゴリ名は捨てちゃって、そうだな、カテゴリを数字か何かで参照するというのはどうかな？そうすれば打ち間違えのリスクは負わなくて良いだろ？

Jちゃん：その通り。でもカテゴリ名をアンケートフォームのタイトルとして使わなければならぬことには変わりないんだよ。

沢田さん：カテゴリを名前じゃなくて数字で参照するんでしょ。たっだらmismatch_topicテーブルをつかって項目の名前について、すでにやったことと同じことじゃないの？

Jちゃん：完璧！項目名をダブって何度もmismatch_responseテーブルにぶち込みたくなかったから、項目名はmismatch_topicテーブルに突っ込んでいて、項目とレスポンスとを数値キーで結びつけたんだもん。

森川氏：つまりカテゴリ名がダブっている問題は、新しくカテゴリテーブルを作れば解けるって言っているんだよね？

沢田さん：正に今それについて話しているのよ。新しくmismatch_categoryテーブルを作って、各カテゴリ名はビックシ1回だけぶっ込む。それでカテゴリと項目とを主キーと外部キーとを使ってmismatch_topicとmismatch_categoryとの間で結びつけるのよ。冴えてるわね！

正規化に向けてちょっとがく

ミスマッチサイトのデータベースからダブったデータをなくし、論理的に一貫性を持たせたテーブルに分割して、これらのテーブルを結合するように再設計するプロセスを、**正規化**と呼んでいます。正規化というのは、データベース設計に関する話題としては、かなりディープなもので、ある意味脅威です。しかし恐るるには足りません。巷にはシンプルなデータベース設計技法が溢れています。これらを使えば、正規化の基本原理に基づいて、MySQLデータベースをよりよい方向に変えていくことができます。単にレイアウトがどうあるべきかと思いつきでやってみるのとは比べものになりません。

以下にデータベース設計プロセスに関して広く用いられているステップを示します。これに従えば、自然と正規的なデータベースへ導かれることでしょう。

1. まずテーブルを使って記述したいものを1つ選びます。

正規化とは、データベースからダブったデータをなくし、データ間の関係を改善させるように設計することを意味します。

テーブルで表現したい
メインとなるものは
何でしょうか？

2. 今選んだものについてテーブルを使って表す必要のある
情報をリストアップします。

このテーブルをどのように
使いますか？

3. そのリストを使って、情報をブレークダウンし最初に選んだもの
を使いやすく分解し、テーブルを構成できるようにします。

どうすればこのテーブルに
最も簡単に問い合わせ
ることができますか？

正規化における基礎的な概念はアトミックデータという考え方を集められます。アトミックデータとは、データベースを使う上で意味を成す最小の形式までブレークダウンしたデータのことです。例えば、first_nameカラムやlast_nameカラムはミスマッチサイトのデータベースにおいてアトミックです。つまりこれらはユーザ名を1つの名前というカラムよりもさらにブレークダウンしているという意味です。このブレークダウンが必要な理由は、ミスマッチサイトではユーザを「名」だけで参照することができるようになしたいからです。

アプリケーションによっては、名前を必ずしも「姓」と「名」に分けてブレークダウンする必要はないかもしれません。このような場合、名前をアトミックとして扱っても全く問題ありません。つまり「今注目しているもの」を分解してブレークダウンする場合、データをどのように使うつもりかを考える必要があります。どのように表現できるかだけを考えればいいのではありません。



アトミックデータとは、
データベースが与えられ
たとき、必要となる最小
の形式にブレークダウン
したデータのことです。

正規化するならアトムを考える

データベース設計の考えを実行に移す段階になったら、データをターゲットとする質問をしてみるのが良いでしょう。これによりデータがどんな感じでテーブルにフィットするかを予想できるとともに、本当に適切なアトミック表現にブレークダウンされたかどうかを確認できます。いまだかつて誰もデータをアトムに分割することが簡単だといった人はいません。しかし以下の質問リストは役に立つことでしょう。

データをアトミックにする
ことが正規化された
テーブルを作る第1歩です。



1. そのテーブルが表現する ひとつのもの
は何ですか？



テーブルは次のようなものを表現していますか？
UFO目撃情報、email購読者リスト、テレビゲーム
のハイスコア、恋に恋する人。



2. その ひとつのもの を取ってくる のに、
どのようにテーブルを 使いますか？



テーブルは簡単に問い合わせができるように設計します！



3. カラム には アトミックデータ が入って
いて、問い合わせ文を簡潔で的確なものにする
ことができますか？



データは必要に応じた分
だけ小さくして下さい。

素朴な疑問に答えます

Q：データをブレークダウンする際、何が何でもこれ以上できなくなるまでちっさくしなければならないのでしょうか？

A：必ずしもその必要はありません。データをアトミックにするというのは、効率的なテーブルを作る上で必要なだけ小さくブレークダウンするという意味です。小さければ小さいほどいいということではありません。必要以上に小さくしないで下さい。余分なカラムは必要ありません。小さくなつてカラムが増えるだけでは愚の骨頂です。

Q：アトミックデータって何の役に立つんでしょうか？

A：まず、テーブルのデータが適正かどうかを確認する際に役に立ちます。例えば、UFO目撃情報の住所というカラムがあるとします。この住所を2つのカラムにブレークダウンしたいと思ったとしましょう。郵便番号とそれ以外とです。そうすると郵便番号の方は、数字のみが現れるはずだという確認が得られます。また、アトミックデータは、問い合わせ文を効率的に実行させる役に立ちます。なぜなら問い合わせ文が書きやすくなりますし、更に実行時間は短くなります。これは大量のデータがデータベースにぶつ込まれている場合に効いてきます。

何故正規化なのか？

データの小型化と正規化についての今までの話が、小さなデータベースに対する大げさだと思うかもしれません。でもご自身のWebアプリケーションが爆発的に大きくななることを想像してみて下さい。ご自身のデータベースが極めて短期間に飛び跳ねるようにサイズを膨らませたらどうなるでしょう？きっと設計時にわかっていたはずの何らかの弱点にしわ寄せが来るでしょう。ネット上で稼いだお金で買った車でショッピングに出かけるのと、データをバンドエイドで修復するような作業をし続けるのとどちらが良いですか？修復作業はどんどんコントロールを失って負のスパイラルに落ち込むのです。きっと正規化が恋しくなることでしょう。

まだ納得がいかないのであれば、または、あのカナリア色(canary yellow)のマクラーレンの夢を追い続けていたいのなら、以下にデータベースを正規化するに値する説得力のある理由を2つ示します。

1. 正規化されたテーブルは、ダブったデータを持ち得ません。 従ってデータベースのサイズを減らすことができます。

ダブついた荷物が満載のデータベースはダメで…



正規化されたデータベースは
非効率的な設計のものに
比べて、格段に小さくなる
傾向にあります。

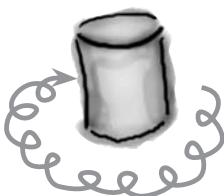


つまり、世間では
「嵩は金なり」と
言われています。

…小さくて効率的なデータベースはイケてます！

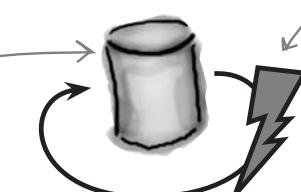
2. サーチするデータが少なくなるので、問い合わせが速く 実行できます。

ダブったデータに巻き込まれて問い合わせが遅くなるようではダメです…



データベースに入っ
きたら、速いとい
うのは常にイケてます。

あ、違った。確か
「時は金なり！」です。



…問い合わせ実行が速いとイケてます！

正規化されたデータベースへの3ステップ

しばらくの間データについてあれこれ考えてきて、何故データを正規化すべきかについて正しい認識が得られたものと思います。しかし抽象的なことばかり言っていてもモノになりません。今必要なのは、正規化を達成するためにどんなデータベースにでも適用できる簡潔な規則のリストです…言ってみれば、チェックリストのようなもので、それを使って作業をすれば、データベースがめでたく正規化できているということを確認できればありがたいわけです。これです。

データベースを正規化するということは、厳密に設計のステップ従うということです。

① カラムがアトミックかどうか確認する。

カラムが本当にアトミックであるということは、そのカラムに同じ型であっても色々な値をとるデータが存在しないということです。逆に同じ型のデータが複数のカラムにあってもいけません。

好き	嫌い
カウボーイブーツ、 ロン毛、 リアリティー番組、 イージーリスニング、 オペラ	タトゥー、ゴールド チェーン、ビアス、 プロレス、ホラー映画
…	…
タトゥー、ゴールド チェーン、ビアス、 カウボーイブーツ、 ロン毛、プロレス、 ホラー映画	リアリティー番組、 イージーリスニング、 オペラ
…	…

同じ型のデータがいくつかの値を持っているのに同じカラムにあります。しかも複数のカラムに同じデータがあります…大問題です。



② 各テーブルにそれぞれ主キーを割り当てる。

主キーは、テーブル内のデータが一意にアクセス可能であることを保証するために絶対必要です。主キーは1カラムでかつ理想的には数値であるべきです。こうすれば問い合わせ文は最も効率的に実行できます。

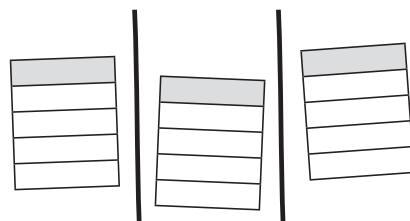


username	password	...
natsuko	08447b...	...
hnoboru	230dcb...	...
khitomi	e511d7...	...
minako	062e4a...	...
seino	b4f283...	...

主キーがないと、テーブル内で行の一意性を保証する手段がありません。

③ キー以外のカラムがお互いに依存していないことを確認する。

これがデータベース正規化についての最も過酷な要請です。しかも厳密に判定できるものではないのです。このステップでは、テーブル内のカラムデータがどのように関係しているのかを細かく見ていく必要があります。見方としては、1つのカラムの値を変更すると別のカラムを変更する必要がないようになっていればよいでしょう。

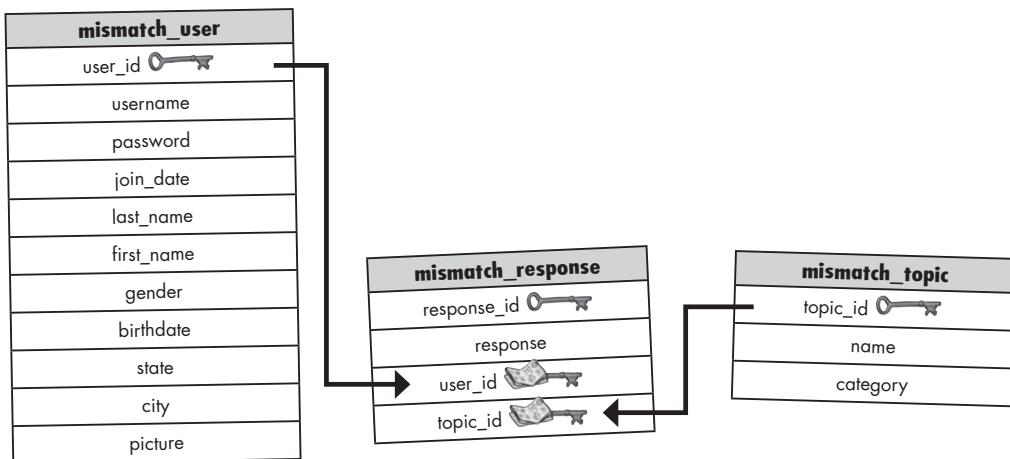


username	password	...	state	city	zip	picture
natsuko	08447b...	...	愛知	名古屋	021-3890	natsukopic.jpg
hnoboru	230dcb...	...	東京	佃	294-0001	noboropic.jpg
khitomi	e511d7...	...	栃木	大田原	306-0123	hitomipic.jpg
minako	062e4a...	...	福島	郡山	853-9900	minakopic.jpg
seino	b4f283...	...	山形	蔵王	388-0111	seinopic.jpg

郵便番号（ZIPコード）のカラム（仮のものです）は、都道府県名や市区町村名に依存しています。つまりこれを変更すると他も変更しなければなりません。この問題を解決するには、ユーザの住所については、郵便番号を主キーとする独立のテーブルに分離する必要があります。



ミスマッチサイトのデータベースを正規化によりオーバホールして、ダブったカテゴリ名の問題を解決する必要があります。今あるデータベース構造に対して、ダブったカテゴリの問題を解決するように改修された設計を図示して下さい。データ入力を間違えた場合のリスクを軽減します。それとともにその設計がどのように作用するかを書き込んで下さい。



素朴な疑問 に答えます

 : 正規化の第3ステップをミスマッチサイトに適用して郵便番号/都道府県名/市区町村名の問題を解決するにはどうすればよいのですか?

 : 解決策は、ユーザの住所を別のテーブルに分離することです。その後mismatch_userテーブルと新しいテーブルとを外部キーで結合すればよいのです。つまりmismatch_locationという名前のテーブルを作って、その主キーをlocation_idとし、例えばですが都道府県名と市区町村名とをそれぞれカラムにぶち込めば良いわけです。そうするとstateやcityカラムがmismatch_userからなくなり、代わりに外部キーlocation_idで置き換えられます。解決です!この設計がうまく動くためには、location_idが郵便番号を主キーとして実際に使う必要があります。これによりキーがないことによる問題を軽減できます。

 : えー!それってあまりにも膨大な作業じゃないですか?データベース設計のちまちました要請に応えるためだけですよね。どーしてもやらなくちゃいけないんですか?

 : YESでもNOでもあります。正規化における最初の2つのステップについては議論の余地はありません。アトミックデータと主キーはいかなるデータベース設計においても重要です。問題は第3ステップで、完全無欠のデータベース設計の誘惑と、実用的で現実的にアプリケーションが本当に必要としているものは何かとをよく考えなければならない場面なのです。ミスマッチサイトの郵便番号/都道府県名/市区町村名の問題については、単純な方が恐らくは良いでしょう。このような決断は安易にすべきではありません。実際データベース専門家の中には正規化のための3つのステップには常に厳格に従わなければならないと主張する人もいます。ただ、めでたいのは郵便番号カラムが仮のものだということです。実際にはmismatch_userのテーブルの一部ではありませんから、心配しなくても大丈夫です。

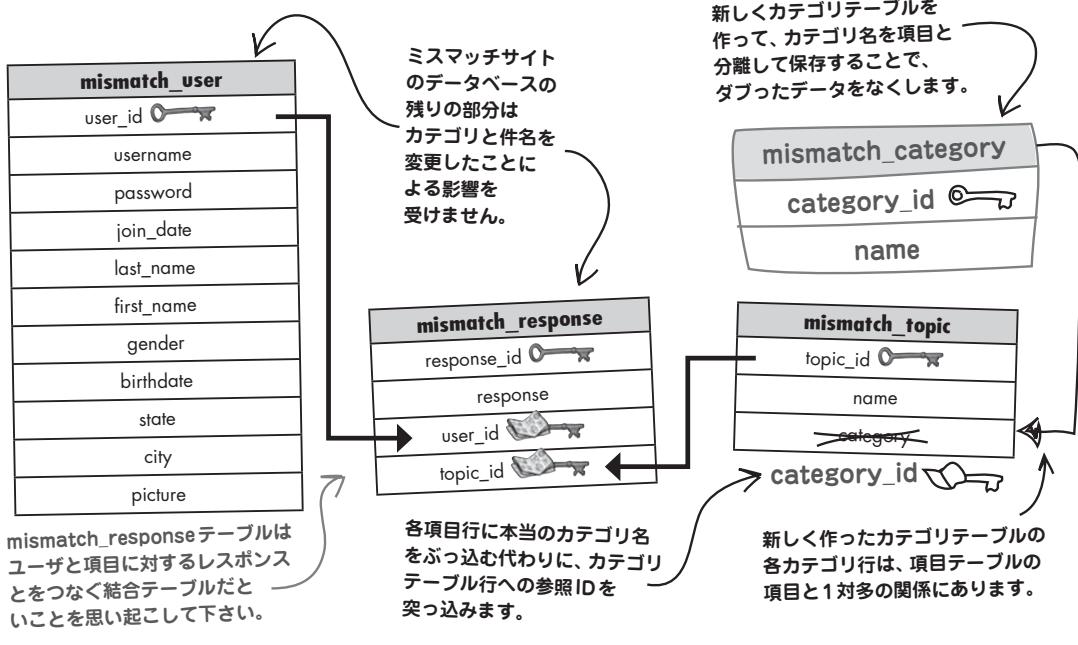
 : 郵便番号カラムがないとしても、都道府県名や市区町村名を別のテーブルに移して第3の正規化ステップに合わせた方が良いのでしょうか?

 : 悪くはありません。今のままだとmismatch_userテーブルに都道府県名/市町村名のデータがダブってしまうというのは確かです。ただ郵便番号なしで、都道府県名や市区町村名を分離するというのは問題があります。何らかの方法で、都道府県名や市区町村名が存在することを確認してからテーブルに入れなればならないのです。そうでないとユーザが仮に市の名前を書き間違えたとすると、相変わらず悩ましいデータができあがってしまいます。これは厳格に正規化を行うことによる利点に重きをおくか、あるいは実用的なアプリケーションの実現性をとるかという二者択一の良い例となっています。以上すべての問題を解決できるイケてるアイデアは、mismatch_userテーブルで都道府県名や市区町村名を使う代わりに、郵便番号を使うというものです。そして必要に応じて静的なテーブルまたはその他のWebサービスから都道府県名や市区町村名を引いてくるのです。ただ、これは今必要な処理よりも遙かに複雑になります。ですから今は単に都道府県名カラムと市区町村名カラムとを貼つ付けておきましょう。



エクササイズ の答え

ミスマッチサイトのデータベースを正規化によりオーバホールして、ダブったカテゴリ名の問題を解決する必要があります。今あるデータベース構造に対して、ダブったカテゴリの問題を解決するように改修された設計を図示して下さい。データ入力を間違えた場合のリスクを軽減します。それとともにその設計がどのように作用するかを書き込んで下さい。



素朴な疑問に答えます

Q： 新しく作ったmismatch_categoryテーブルでどのようにダブったデータの問題を解決できるのかちゃんと説明してもらえませんか？

A： 新しいテーブルは、カテゴリ名をmismatch_topicテーブルから分離して、このテーブル自身にぶち込むようになります。カテゴリを自身のテーブルにぶち込むことで、名前がダブるということはもうなくなります。各カテゴリに関する行があるだけで、この行はmismatch_topicテーブルの行から参照されるようになります。つまり、mismatch_categoryテーブルの各カテゴリ行は、mismatch_topicテーブルの項目行と1対多の関係にあることになります。

Q： そうするとmismatch_categoryテーブルは、各カテゴリについて1つで5行しかないということで良いのでしょうか？

A： その通りです。

mismatch_category	
category_id	name
1	ファッション
2	エンタメ
3	食べ物
4	有名人
5	趣味

各カテゴリ名は1つだけぶち込まれます！

ミスマッチサイトのデータベースを変更する

新しいスキーマの恩恵に浴するためには、ミスマッチサイトのデータベースを構造的に変更しなければなりません。もっとちゃんと言えば、新しく mismatch_category テーブルを作る必要があります。次に mismatch_topic テーブルに新しく外部キーを作り、先ほどのテーブルと結合します。更に、mismatch_topic テーブルには、ダブったカテゴリデータだらけの category カラムは不要となりますから、これを DROP で削除します。

```
CREATE TABLE mismatch_category (
```

新しくカテゴリーテーブルを作つて、カテゴリ名自身を入れておきます。

```
    category_id INT NOT NULL AUTO_INCREMENT,  
    name VARCHAR(48) NOT NULL,  
    PRIMARY KEY (category_id)  
)
```

古いcategoryカラムはDROPで削除します。カテゴリはカテゴリーテーブルから参照することになるので必要ありません。

```
ALTER TABLE mismatch_topic
```

```
DROP COLUMN category
```

```
ALTER TABLE mismatch_topic
```

```
ADD COLUMN category_id INT NOT NULL
```

mismatch_topic	
topic_id	category_id
...	
8	ホラー映画
9	イージーリスニング
10	オペラ
11	寿司
12	SMAP
13	激辛
14	ピーナッツバター・バナナサンド
15	マティーニ
16	ハワード・スタン
17	ビルゲイツ
18	バーバラ・ストライサンド
...	

mismatch_category	
category_id	name

新しく外部キー category_id を追加して、各項目をカテゴリーテーブルのカテゴリ名とつなぎます。

新しく作った mismatch_category テーブルには、カテゴリデータを入れます。このような場合、INSERT 文で手軽に達成できます。

```
INSERT INTO mismatch_category (name) VALUES ('ファッション')  
INSERT INTO mismatch_category (name) VALUES ('エンタメ')  
INSERT INTO mismatch_category (name) VALUES ('食べ物')  
INSERT INTO mismatch_category (name) VALUES ('有名人')  
INSERT INTO mismatch_category (name) VALUES ('趣味')
```

新しく作った category_id カラムに入るべきデータは、各項目のカテゴリと mismatch_category テーブルの適切なカテゴリとを正しく結びつけるものでなければなりません。

```
UPDATE mismatch_topic SET category_id = 3  
WHERE name = 'マティーニ'
```

このIDは mismatch_category テーブルのカテゴリに対応するID(自動増加で設定したもの)でなければなりません。



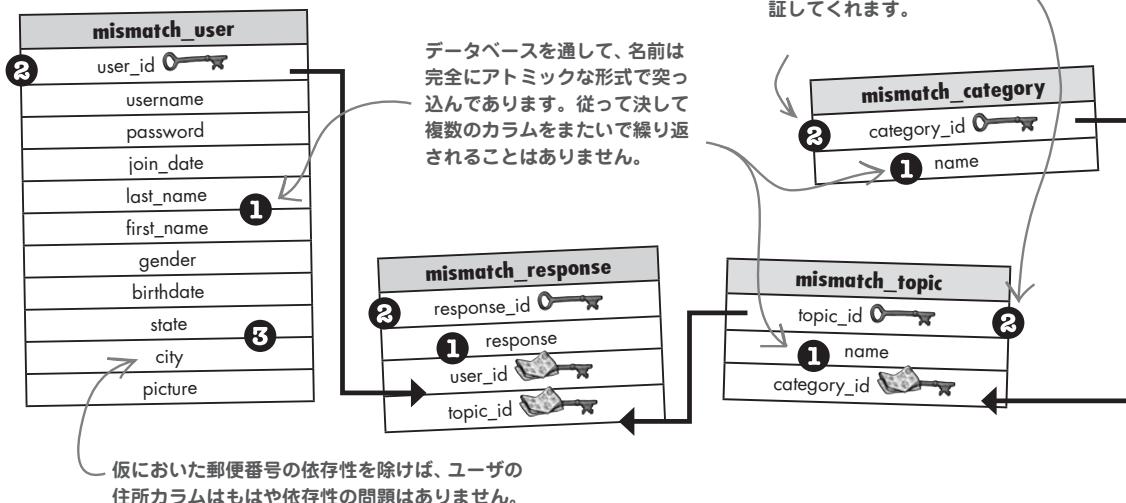
試運転

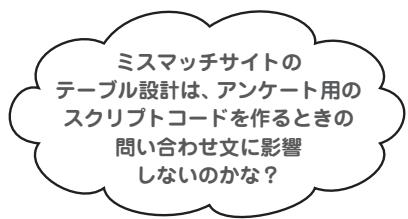
新しく `mismatch_category` データベーステーブルを作つて組み込む。

MySQLツールを使って、前ページのSQLコマンド`CREATE TABLE`を発行し、新しいテーブルを `mismatch_category` という名前でミスマッチサイトのデータベースに追加します。次に `INSERT` 文を発行して、テーブルにカテゴリデータを入れます。さらに `ALTER` 文を2つ走らせ、`mismatch_id` テーブルを変更して、`category_id` カラムを追加します。最後に `mismatch_topic` の各行を `UPDATE` して、`category_id` カラムが `mismatch_category` テーブルの正しいカテゴリを指すようにします。

ミスマッチサイトは本当に正規化されたのでしょうか？

もちろんです。正規化に関する3つの主要規則をミスマッチサイトのテーブルに適用すれば、大成功を取めているということに気付くでしょう。もしそうでなかったとしても、何も失うものはないのです。十人十色と同様のことが、データベースの正規化についてもある程度あてはまります。大事なことは、データベースを完全に正規化されるように設計しようということです。ルールを避けて通るに値する理由があれば、ちょっと敷居を下げてしまえばよいのです。





mismatch_category	
category_id	name
1	趣味

```

レスポンステーブルをデータベースから取得し、フォームを生成
...
// Grab the response data from the database to generate the form
$query = "SELECT response_id, topic_id, response FROM mismatch_response " .
    "WHERE user_id = '" . $_SESSION['user_id'] . "' ";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) { レスponsの項目名を topic テーブルから探索
    // Look up the topic name for the response from the topic table
    $query2 = "SELECT name, category FROM mismatch_topic " .
        "WHERE topic_id = '" . $row['topic_id'] . "' ";
    $data2 = mysqli_query($dbc, $query2);
    if (mysqli_num_rows($data2) == 1) {
        $row2 = mysqli_fetch_array($data2);
        $row['topic_name'] = $row2['name'];
        $row['category_name'] = $row2['category'];
        array_push($responses, $row);
    }
}
...

```



questionnaire.php

影響します。当然ながら、データベースの構造的な変更をすれば、影響の出たテーブルに対する問い合わせ文はほぼ間違いなく微調整が必要になります。

今回の場合、データベースの設計を変更して新しく mismatch_category テーブルを追加したので、mismatch_topic テーブルに関連するすべての問い合わせ文に影響が出ます。なぜなら先ほどの版のデータベース設計は、カテゴリが直接 mismatch_topic テーブルにぶち込んでいたからです。今回の変更でカテゴリは専用のテーブルに分離されました。これは正規化という素晴らしいアイデアのおかげです。これに対応して問い合わせ文を再構築し、追加されたテーブル (mismatch_category) に対応して動作するように変更しなければなりません。

問い合わせ文の中の問い合わせ文の中の問い合わせ文の中の…

データベースを正規化したことによって1つ問題が発生しています。問い合わせ文にはその子どもの問い合わせ文が必要になってしまうのです。複数のテーブルを伝ってデータに行き着く必要があるからです。これはなかなか厄介です。新しいバージョンの問い合わせ文でレスポンスの配列を作り、ミスマッチサイトのアンケートを生成することを考えてみて下さい。

テーブルが増えると
問い合わせ文が厄介
になっていきます。

```
// Grab the response data from the database to generate the form
$query = "SELECT response_id, topic_id, response FROM mismatch_response "
    . "WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);

$responses = array();
while ($row = mysqli_fetch_array($data)) { // レスポンスの項目名を topic テーブルから探索
    // Look up the topic name for the response from the topic table
    $query2 = "SELECT name, category_id FROM mismatch_topic "
        . "WHERE topic_id = '" . $row['topic_id'] . "'";
    $data2 = mysqli_query($dbc, $query2);
    if (mysqli_num_rows($data2) == 1) {
        $row2 = mysqli_fetch_array($data2);
        $row['topic_name'] = $row2['name'];

    // Look up the category name for the topic from the category table
    $query3 = "SELECT name FROM mismatch_category " . // 項目のカテゴリ名を category テーブルから探索
        "WHERE category_id = '" . $row2['category_id'] . "'";
    $data3 = mysqli_query($dbc, $query3);
    if (mysqli_num_rows($data3) == 1) {
        $row3 = mysqli_fetch_array($data3);
        $row['category_name'] = $row3['name'];
    }
    array_push($responses, $row);
}
}

// 覚えていますか？この関数は
// 問い合わせ文が何行のデータ
// を返すのかを教えてくれます。
array_push($responses, $row);
```

データを3つの異なる
テーブルから取って
くることになるので、
3つの問い合わせ文が
必要になります。

この新しい問い合わせ文は、
category_id キーを使って
カテゴリ名をカテゴリーテーブル
から取ってきます。

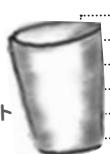
mismatch_response	
response_id	user_id
26	1
27	2
28	3
29	4
30	5
31	6
32	7
33	8

これは一時的なレスポンス
の配列で、ミスマッチサイト
のアンケートフォームを
作るのに使います。

mismatch_topic	
topic_id	category_id
タトゥー	ファッショ
ゴールドチェーン	ファッショ
ピアス	ファッショ
カウボーイブーツ	ファッショ
ロン毛	ファッショ
リアリティーパン組	エンタメ
プロレス	エンタメ
ホラー映画	エンタメ

mismatch_category	
category_id	name
タトゥー	ファッショ
ゴールドチェーン	ファッショ
ピアス	ファッショ
カウボーイブーツ	ファッショ
ロン毛	ファッショ
リアリティーパン組	エンタメ
プロレス	エンタメ
ホラー映画	エンタメ

ここでデータがダブっている
のは問題ではありません。
もともと1つのところ、つまり
カテゴリーテーブルから抽出した
もので、ダブって格納されて
いるわけではありません。



response_id	topic_id	category_id	name	response
26	1	好き	タトゥー	ファッショ
27	2	好き	ゴールドチェーン	ファッショ
28	3	好き	ピアス	ファッショ
29	4	好き	カウボーイブーツ	ファッショ
30	5	好き	ロン毛	ファッショ
31	6	嫌い	リアリティーパン組	エンタメ
32	7	好き	プロレス	エンタメ
33	8	好き	ホラー映画	エンタメ

テーブル みんな手をつなぎましょう

うわ！こんな風にネストした問い合わせ文全部をどうにかすることはできないでしょうか？解決策はSQLの結合(join)という機能があります。これを使えば、2つ以上のテーブルに対して1つの問い合わせ文で結果を引っ張って来ることができます。結合には多くの異なる種類がありますが、もっとも一般的な結合は、内部結合(inner join)と言って2つのテーブルから条件に従って行を取ってきます。内部結合では、問い合わせ文の結果はこの条件にマッチした行のみとなります。

結合を使えば、複数のテーブルから1つの問い合わせ文で結果を得ることができます。

```
SELECT mismatch_topic.topic_id, mismatch_category.name
FROM mismatch_topic
INNER JOIN mismatch_category
ON (mismatch_topic.category_id = mismatch_category.category_id)
```

カテゴリーテーブルと項目テーブルとがINNER JOINで結合されます。

項目IDとカテゴリ名を問い合わせ文で取ってきます。このカラムは2つの異なるテーブルにあります。

結合の条件は、カテゴリIDが返される各行のデータとマッチしなければならないという意味です。

mismatch_topic		
topic_id	name	category_id
1	タトゥー	1
2	ゴールドチェーン	1
3	ピアス	1
4	カウボーイブーツ	1
5	ロン毛	1
6	リアリティー番組	2
7	プロレス	2
8	ホラー映画	2
9	イージーリスニング	2
10	オペラ	2
11	寿司	3
...		

結果の第1カラムは項目テーブルから取ってきた項目IDです。

mismatch_category

category_id	name
1	ファッション
2	エンタメ
3	食べ物
4	有名人
5	趣味

このカラムで結合をコントロールします！

1	ファッション
2	ファッション
3	ファッション
4	ファッション
5	ファッション
6	エンタメ
7	エンタメ
8	エンタメ
...	

結果の第2カラムはカテゴリーテーブルから取ってきたカテゴリ名のうち、各項目IDとマッチするものです。

結果のデータには2つのテーブルから取ってきたカラムが入っています！

内部結合を使うと、2つのテーブルからデータを取ってきてうまくくっつけることができます。これを使わなかつたら2つの別々の問い合わせ文が必要となっていました。問い合わせ結果は、両方のテーブルから取ってきたデータでできています。

で ドットを結ぶ

結合は2つ以上のテーブルに関わるので、結合で参照される各カラムについて明らかにしておくことが重要です。もっとちゃんと言うと、テーブルの各カラムを識別して、混乱を来たさないようにしなければなりません。2つ以上のテーブルには同じ名前がかかるということがあります。特にキーカラムの場合に多く見られます。単にカラム名の前にテーブル名をおいて、ドットでつなぐことで解決できます。例えば、先ほどの INSERT JOIN問い合わせ文は次のようにになります。これにより結果として項目IDとカテゴリ名のペアを作ることができます。

これは
テーブル名です。 ドットです！ これはテーブル内のカラム名で、テーブル名
とはドット(ピリオド)で分けられています。

```
SELECT mismatch_topic.topic_id, mismatch_category.name
FROM mismatch_topic
INNER JOIN mismatch_category
ON (mismatch_topic.category_id = mismatch_category.category_id)
```

ここでドットが本当に役立っています。
カラム名は同一なので、テーブル名が
なければ完璧に曖昧になります。

ここにもう1つテーブル/
カラム参照があって、
ドット記法を使っています。

カラム名だけでは、それが
どのテーブルのものなのか
がわかりません。

ドット記法によって、テーブルとカラム名とを結び付けることができなれば、この問い合わせ文は曖昧でわけがわからなくなってしまいます。実際、先ほどの問い合わせ文のONの部分は理解不可能です。category_idカラム同士が等しいかをチェックすることになってしまいますが、例えばmismatch_topicテーブルの中だけでもできてしまいます。このため、曖昧かどうか関わらず常にカラム名に結び付いてるテーブル名を明示的に指定して、JOIN問い合わせ文を作るというのは極めてよいことです。

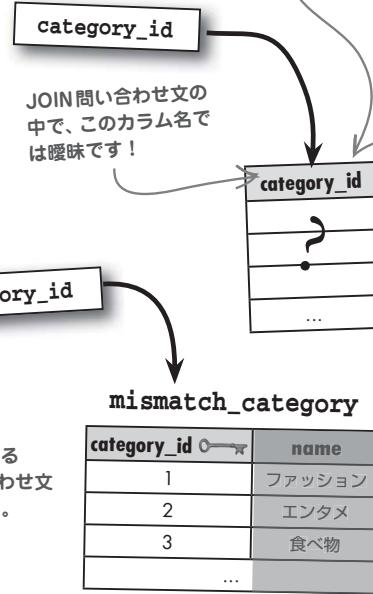
mismatch_topic.category_id

topic_id	name	category_id
1	タトゥー	1
2	ゴールドチェーン	1
3	ピアス	1
...		

mismatch_category.category_id

category_id	name
1	ファッショ
2	エンタメ
3	食べ物
...	

テーブル名を指定する
ことでJOIN問い合わせ文
が明示的になります。



内部結合でまだたくさんのことができます

内部結合は2つのテーブルのデータをくっ付けるだけに留まりません。内部結合も結局のところ単なる問い合わせ文ですから、通常の問い合わせ要素を使って、更に結果をコントロールすることができます。例えば、結合した結果から特定の行を取り出したければ、WHERE節をINNER JOINに貼つけることで、望みの行だけにすることができます。

```
SELECT mismatch_topic.topic_id, mismatch_category.name
FROM mismatch_topic
INNER JOIN mismatch_category
ON (mismatch_topic.category_id = mismatch_category.category_id)
WHERE mismatch_topic.name = 'ホラー映画'
```

ではこの問い合わせ文で正確には何が返ってくるでしょうか？まず第1にWHERE節は、先ほどの問い合わせ文を洗練するためのものだということを思い起こして下さい。言い換えれば、元々のINNER JOIN問い合わせ文で返ってきた行はそのまま持っているのです。要約すると、WHERE節がなければ内部結合の結果は次のようにになります。

項目IDは mismatch_topic テーブルから抽出されます。

カテゴリ名は mismatch_category テーブルから引っ張ってきます。

1	ファッショ
2	ファッショ
3	ファッショ
4	ファッショ
5	ファッショ
6	エンタメ
7	エンタメ
8	エンタメ
...	

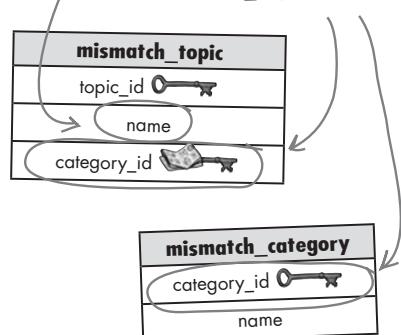
この行がWHERE
節にマッチしています。

mismatch_topic		
topic_id	name	category_id
...		
7	プロレス	2
8	ホラー映画	2
9	イージーリスニング	2
...		

INNER JOINは比較演算子を条件として使うことで、2つのテーブルの行を結び付けます。

このカラムは WHERE節の一部なので結果を洗練するために使われます。

この2つのカラムで2つのテーブル間の結合をコントロールします。



元々のINNER JOINの結果を WHERE節でぶった切ってこの1行になります。

8	エンタメ
---	------

WHERE節は結合の結果を1行に制限します。

素朴な疑問に答えます

 つまり WHERE 節を使えば JOIN 問い合わせ文の結果を、結合された行に基づいて制限することができるということですか？

 その通りです。心に留めておいて欲しいのは、実際の比較は WHERE 節を元々のテーブルに対して行っているということです。問い合わせの結果に対してではありません。つまりミスマッチサイトの例で言うと、問い合わせ文で 2 つの異なるテーブルからデータを取ってきます。このとき両方のテーブルに出てくる特定のカラム (category_id) とマッチするものだけを取ってきます。そして次に mismatch_topic の name カラムが特定の値（「ホラー映画」）であるようなものだけを選びます。結局 INNER JOIN は category_id カラムに関して、両方のテーブルに働きかけますが、WHERE 節は結果を mismatch_topic テーブルの name カラムのみを使って洗練するのです。

 ミスマッチサイトの JOIN 問い合わせ文にある WHERE 節を mismatch_category テーブルに基づくように変えることができますか？

 確かにできます。WHERE 節は問い合わせ結果に制約を加える効果を持ちますが、それは結合に含まれるどちらのテーブルに基づいても構いません。例として、WHERE 節を以下のように特定のカテゴリだけを見るように変えてみましょう。

```
... WHERE mismatch_category.name = 'エンタメ'
```

この WHERE 節は結果がエンタメカテゴリに落ちこちるような項目のみを含むように限定しています。つまり WHERE 節はテーブルがどのように結合されるかには影響されません。単に問い合わせ結果の特定の行のみに影響します。

ON を USING で単純化する

当面の目標はミスマッチサイトのややこしい問い合わせ文を INNER JOIN で単純化することでした。内部結合が同じ名前をマッチさせるような場合、問い合わせ文を更に単純化することができます。USING 文を使うのです。 USING 文は INNER JOIN 問い合わせ文の ON の代わりに使うことができ、マッチさせるカラム名を指定します。ただしカラムは両方のテーブルで全く同じ名前でなければならないことに注意して下さい。例としてもう一度ミスマッチサイトの問い合わせ文を示します。

```
SELECT mismatch_topic.topic_id, mismatch_category.name
FROM mismatch_topic
INNER JOIN mismatch_category
ON (mismatch_topic.category_id = mismatch_category.category_id)
WHERE mismatch_topic.name = 'ホラー映画'
```

各カラムの
名前は同じです。
テーブル名のみ
が異なります。

問い合わせ文の ON の部分は、同じ名前のカラム (category_id) に依存しています。このような場合 USING 文で単純化できます。

```
SELECT mismatch_topic.topic_id, mismatch_category.name
FROM mismatch_topic
INNER JOIN mismatch_category
USING (category_id)
WHERE mismatch_topic.name = 'ホラー映画'
```

必要なものはカラム名だけです…
等しいことを指定するために「=」を
付ける必要はありません。

ON を USING で書き直すと、もっと 簡潔 な内部結合の問い合わせ文となり、共通のカラム名 でマッチさせます。

内部結合で
USING 文を使う
ためには、カラム名が
同じでなければ
なりません。

テーブルやカラムに愛称をつける

INNER JOIN問い合わせ文はどんどんしっかりしてきました！もう1歩進みましょう。SQL問い合わせ文を発行するとき、テーブルやカラムはデータベースに格納されている名前で参照するのが標準的です。しかし複数の結合を含むような大きな問い合わせ文では、これでは煩わしいことがあります。名前のせいで問い合わせ文が読みにくくなってしまうことがあります。そんなときは別名をつけると良いかもしれません。つまりテーブルやカラムを問い合わせ文の中で参照する一時的な名前です。ミスマッチサイトの問い合わせ文を別名を用いて書き直してみましょう。

```

SELECT mt.topic_id, mc.name
FROM mismatch_topic AS mt
INNER JOIN mismatch_category AS mc
USING (category_id)
WHERE mt.name = 'ホラー映画'
      mismatch_topicへ
      の参照はmtと短く
      なりました。

```

テーブル名を短い別名に圧縮
したので、コードがちょっと
読みやすくなりました。

SQLのキーワードASで別名を作ります。
この場合で言うとmismatch_topic
テーブルの別名です。

mismatch_categoryは別名の
おかげで単にmcで参照するこ
ができるようになります。

この2つのカラムを結合で
選ぶことにより結果のカラムは
同じ名前になってしまいま
ダメです！

カラム名を別名で変更
すると、別名は問い合わせ
結果に表れます。

別名により問い合わせ文の
中でテーブルやカラムの
名前を変えることで、
問い合わせ文をいくらか
単純化することができます。

別名の利点は、問い合わせ文をコンパクトに書けるようにすることだけでしょうか？それだけではありません。別名が極めて本質的な状況というのはいくつかあります！ミスマッチサイトのアプリケーションでは、結合によりお手軽に特定のIDに対する項目名とカテゴリ名の両方を取ってくることができました。しかし mismatch_topic テーブルと mismatch_category テーブルは、このデータに対して両方で同じカラム名(name)を使っています。これは問題なのです。なぜならこれら2つのカラムを結合した結果として、曖昧なカラム名ができあがってしまうからです。しかし結果のカラムに別の名前をつけることができるのです。カラムは別名により曖昧性がなくなり目的を射たものとなっています。

```

SELECT mt.name AS topic_name, mc.name AS category_name
FROM mismatch_topic AS mt
INNER JOIN mismatch_category AS mc
USING (category_id)
WHERE mt.topic_id = '11'

```

選ばれたカラムは別名に
より的を射た名前になって
います。

これで結果のカラムは
一意な名前になりました。
非常に当を得ています。



結合で救出する

つまり結合を使えば1つの問い合わせ文に2つ以上のテーブルを含めることができます。従って複数箇所から効率的にデータを取り出し、1つのテーブルに結果を貼っ付けることができます。ミスマッチサイトの問い合わせ文で、レスポンス配列を作るというのは、結合を使うにはうってつけの場です。なぜなら少なく見積もっても3段にネストした問い合わせ文で複数のテーブルを扱わなければならいためです。では元々のコードから見ていくことにします。

```
// Grab the response data from the database to generate the form
$query = "SELECT response_id, topic_id, response FROM mismatch_response " .
    "WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
    // Look up the topic name for the response from the topic table
    $query2 = "SELECT name, category_id FROM mismatch_topic " .
        "WHERE topic_id = '" . $row['topic_id'] . "'";
    $data2 = mysqli_query($dbc, $query2);
    if (mysqli_num_rows($data2) == 1) {
        $row2 = mysqli_fetch_array($data2);
        $row['topic_name'] = $row2['name'];
    }
    // Look up the category name for the topic from the category table
    $query3 = "SELECT name FROM mismatch_category " .
        "WHERE category_id = '" . $row2['category_id'] . "'";
    $data3 = mysqli_query($dbc, $query3);
    if (mysqli_num_rows($data3) == 1) {
        $row3 = mysqli_fetch_array($data3);
        $row['category_name'] = $row3['name'];
    }
    array_push($responses, $row);
}
```

コード中の後ろ2つの問い合わせ文で、項目名やカテゴリ名をそれぞれのテーブルから取ってきてています。テーブル毎に1つの問い合わせ文です。

そしてこれが結合を使った新しい版のコードです。

```
// Grab the response data from the database to generate the form
$query = "SELECT response_id, topic_id, response FROM mismatch_response " .
    "WHERE user_id = '" . $_SESSION['user_id'] . "'";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
    // Look up the topic and category names for the response from the topic and category tables
    $query2 = "SELECT mt.name AS topic_name, mc.name AS category_name " .
        "FROM mismatch_topic AS mt " .
        "INNER JOIN mismatch_category AS mc USING (category_id) " .
        "WHERE mt.topic_id = '" . $row['topic_id'] . "'";
    $data2 = mysqli_query($dbc, $query2);
    if (mysqli_num_rows($data2) == 1) {
        $row2 = mysqli_fetch_array($data2);
        $row['topic_name'] = $row2['topic_name'];
        $row['category_name'] = $row2['category_name'];
        array_push($responses, $row);
    }
}
```

結合を使えば、項目名とカテゴリ名の両方を1つの問い合わせ文でとつ捕まえることができます。

レスポンスの項目名とカテゴリ名とをtopicテーブルとcategoryテーブルから探索

別名を使うとコードを単純化できます。

項目IDを使ってメインの問い合わせ文を作っていますが、カテゴリIDは結合をコントロールしているだけです。



まだ納得いかないわ。相変わらず
カテゴリ名を探すための余分な問い合わせ文が
あるじゃない。結合っていうのがそんなに
スゴイだったら、2つの問い合わせ文
なんて要らないんじゃないの？

2つの問い合わせ文は要りません。もし結合をフルパワーで 使ってればの話ですが。

3つ以上のテーブルを結合することもできます。ミスマッチサイトのレスポンス配列のコードで正にこのようなことが要求されています。1つの問い合わせ文で次の3つのこと達成できればよいのです。(1)ユーザのすべてのレスポンスを取り出す。(2)各レスポンスの項目名を持ってくる。(3)各レスポンスのカテゴリ名を持ってくる。反対側のページで作った改良版の新しいコードでは、後ろの2ステップを1つの問い合わせ文で達成しています。この問い合わせ文では、mismatch_topicとmismatch_categoryとを結合しています。理想的に言えば、1つの問い合わせ文が2つの結合を持っていて、「一(巨大結合)石三鳥」となってくれれば良いのです。



エクササイズ

以下のコードは、データベースから1つの問い合わせ文でレスポンスデータを引っ張ってくことができます。ただし結合を賢く使えばの話です。では賢くなりましょう。mismatch_responseとmismatch_topicとmismatch_categoryの3つのテーブルをすべて結合するSQL問い合わせ文を書いて下さい。

レスポンスデータをデータベースから取得し、フォームを生成

```
// Grab the response data from the database to generate the form
$query = .....  
.....  
.....  
.....  
.....  
  
$data = mysqli_query($dbc, $query);  
$responses = array();  
while ($row = mysqli_fetch_array($data)) {  
    array_push($responses, $row);  
}
```



エクササイズ の答え

以下のコードは、データベースから1つの問い合わせ文でレスポンスデータを引っ張ってくことができます。ただし結合を賢く使えばの話です。では賢くなりましょう。`mismatch_response`と`mismatch_topic`と`mismatch_category`の3つのテーブルをすべて結合するSQL問い合わせ文を書いて下さい。

```
// Grab the response data from the database to generate the form
$query = "SELECT mr.response_id, mr.topic_id, mr.response, ";
        ."mt.name AS topic_name, mc.name AS category_name ";
        ."FROM mismatch_response AS mr ";
        ."INNER JOIN mismatch_topic AS mt USING (topic_id) ";
        ."INNER JOIN mismatch_category AS mc USING (category_id) ";
        ."WHERE mr.user_id = ". $_SESSION[ 'user_id' ]. ";
$data = mysqli_query($dbc, $query);
$responses = array();
while ($row = mysqli_fetch_array($data)) {
    array_push($responses, $row);
}
$responses = array();
foreach ($responses as $row) {
    $response_array[$row['response_id']] = [
        'topic_name' => $row['topic_name'],
        'category_name' => $row['category_name'],
        'response' => $row['response']
    ];
}
```

最初の結合で、項目テーブルを問い合わせ文の中に取り込み、それを使って項目IDから項目名を取ってきます。

2番目の結合で、カテゴリーテーブルをカテゴリIDにより問い合わせ文に組び付けるので、カテゴリ名にアクセスできるようになります。

すべての問い合わせ結果データを\$responsesに突っ込みます。

素朴な疑問に答えます

Q: 他にはどんな種類の結合があるのでしょうか?

A: 内部結合(inner join)としては、他に等結合(equijoin)、非等結合(non-equijoin)、自然結合(natural join)などがあります。等結合と非等結合とは、比較結果がそれぞれ等しいか、または等しくないかに基づいて内部結合を実行します。等結合については既にミスマッチサイトの問い合わせ文でいくつかの例を見てきました。これらの問い合わせ文では`topic_id`カラムや`category_id`カラムがマッチするかどうかをチェックしていました。ここでのマッチングはカラムが「等価」のもの(同じIDのもの)を探していましたから、このような問い合わせ文のことを等結合といいます。

内部結合には自然結合という種類もあって、すべてのカラムを比較して、2つのテーブル間で同じ名前を持つものを求めます。つまり自然結合は実際には単に等結合なのです。結合に使われるカラムが自動的に決まるのが自然結合です。自然結合におけるこの自動性というのは、普通の内部結合に比べると記述性に欠けることを意味します。なぜなら何が起こるのかは、ただ見ただけでは明らかではないからです。自分でデータベース構造を把握してどのカラムが結合に使われるのかを理解しなければなりません。

Q: ではすべてのSQLの結合というのは、単に内部結合のバリエーションだけなのでしょうか?

A: そうではありません。他にもたくさんの結合があり、お好みでお使い頂けます。主な結合の分類として、総称して外部結合と呼ばれるものがあります。しかし外部結合の中にもいくつかの異なる種類があります。左外部結合、右外部結合、完全外部結合の3種類です。もう1つ、畏れ多き3重らせん構造同性愛結合というのもあります。冗談です。最後のヤツは本物の結合ではありません。でもあって良いでしょう! 外部結合というのは、基本的にテーブルを結合対象の行が結合に際し必ずしもマッチしなくても良いのです。このため外部結合は、マッチングの条件に関わらず常にテーブルから結果として行を構成することができます。外部結合も内部結合と同様に手軽に使えます。どちらが良いかは、データベースアプリケーションが何を必要としているかに依存します。異なる種類の結合についてどのように使うのかをもっと勉強したい場合は、『Head First SQL』をご一読下さい。



アンケート用スクリプトを改造して、ユーザのレスポンスデータを 1つの問い合わせ文で取ってくるようにします。

questionnaire.phpスクリプトを修正し、内部結合を使って、1つの問い合わせ文でユーザのレスポンスデータを取ってきて処理できるようにします。新しく作ったスクリプトをWebサーバにアップロードしたら、Webブラウザでアンケートのページまで行ってみます。もしごく普通にうまくいっているのであれば、以前と何も違わないはずです…ただし本当はスクリプトコードが刷新されているのです！





ミスマッチサイトはこれでユーザのレスポンスを覚えることができるようになりました。しかし、それを使ってまだ何もしていません…ミスマッチを見つけるときです！

ユーザのレスポンスデータを集めただけでは、ミスマッチを成功させるための中間地点に来ただけです。ミスマッチサイトのアプリケーションには、キューピットに矢の火をつけるメカニズムが欠けています。データベースの中から赤い糸を見つけださなければなりません。このためには、データベースの中からすべてのユーザのレスポンスを検査して、誰が理想的なミスマッチにマッチしているのかを見ていかなければなりません。



理想的なミスマッチを見つけ出すのって、
すごく複雑そうね。だって、こんなにたくさんの
カテゴリとか項目とかレスポンスがあるのよ。
こんなこと本気でできるって言ってるの？

間違いなく全然できます。やるべきことは一貫していて、任意の
2人のユーザがどれだけ多くのミスマッチ項目を共有している
のかを計算するだけです。

十分にシンプルな方法でミスマッチしている項目をユーザが共有しているの
かを計算することができるのであれば、ユーザデータベースをループしてユー
ザ同士を比較することが可能です。各ユーザについてミスマッチの数が最高
の人が、理想のミスマッチとなるわけです！

$$\heartsuit + \cancel{\circ} = \text{ミスマッチ!}$$

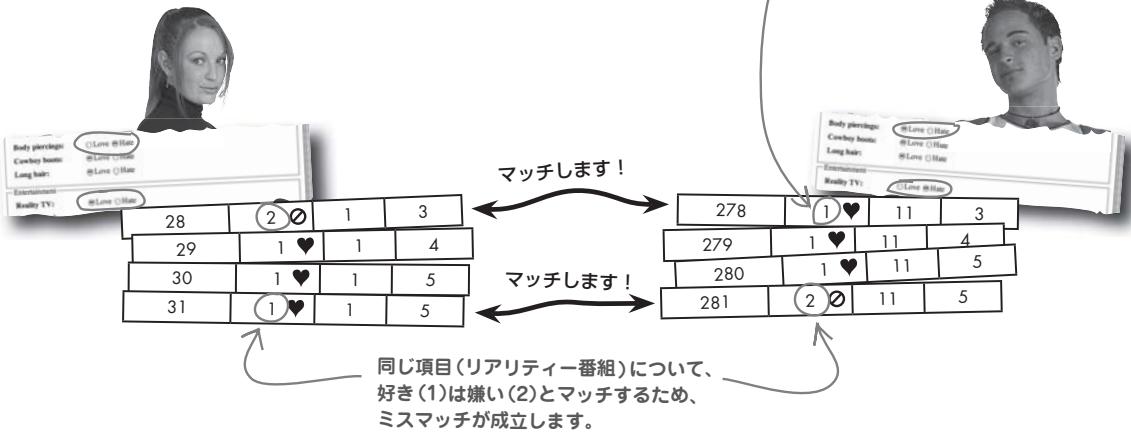
「ミスマッチ性」を計算するにはどうすればよいと思うか書いて下さい。
ミスマッチサイトのデータベースにぶち込まれているデータを使って
2人のミスマッチを計算します。

愛は数字のゲームです

思い出して下さい。ミスマッチのレスポンスはmismatch_responseテーブルを数字でぶち込んかったことです。0と1と2で、それぞれ特定のレスポンスに対応して特別な意味を与えていました。

「未回答」 = 0 「好き」 = 1 「嫌い」 = 2

これが2人のユーザ間のミスマッチを計算するために使うデータです。そして今特に探しているものは、「好き」が「嫌い」にマッチするか、または「嫌い」が「好き」にマッチするかのいずれかです。言い換えれば、探しているものはresponseが1に対して2にマッチするか、または2に対して1にマッチするような行です。



実は相変わらず手軽な方法でPHPコードにできません。2つのレスポンス間でミスマッチかどうかを決定したいのです。確かに2組のif-else文を使えば、値がの組が1と2のペアか、または2と1のペアかをチェックすることはできます。でももっとエレガントな解決策があるのです。どちらの場合にも、2つのレスポンスの結果を足すと値は3になります。つまり単純な等式を使って2つのレスポンス間にミスマッチが成り立つかどうかを調べることができます。

もし(レスポンスA + レスポンスB = 3)ならばミスマッチが成立します!

つまり赤い糸を見つけるということを、単純な数式に落とし込むことにできたのです。これで、個々のマッチを比較することに関する特定の問題は解けました。しかしこの問題は残っています。ミスマッチサイトのスクリプトを実際にどうやって作り上げれば良いのでしょうか?

ミスマッチを成功に導く5つのステップ

完全にミスマッチな誰かを見つけ出すということは、レスポンス行を単に比較すれば済むという問題ではありません。ミスマッチ相手発見用スクリプトは、注意深く編成されたステップに従って、ミスマッチを成功させなければなりません。以下のステップが最終的にユーザを満足させアンケートのレスポンスに意味を持たせてくれます。

① `mismatch_response` テーブルからユーザのレスポンスを取ってきます。項目名とその回答とを結合することを忘れないで下さい。

② ミスマッチのサーチ結果を初期化します。このとき「最高のミスマッチ」を保存する変数も初期化しておきます。

③ ユーザテーブルをループして、ユーザのレスポンスと他の人のレスポンスとを比較します。これは、データベース内のすべての人のレスポンスを対応するレスポンス同士についてすべて比較することを意味しています。「スコア」を導入してユーザがそれぞれの相手と正反対のレスポンスを何項目共有しているかを数えます。

④ ループの各サイクルにおいて、実行中のミスマッチが今までの最高よりも良いかどうかを比較します。もし良ければ、今の値を新しい「最高のミスマッチ」として保存し、一緒にミスマッチした項目も取っておきます。

⑤ 「最高のミスマッチ」が見つかったら、問い合わせ文でミスマッチした相手に関するより多くの情報を問い合わせ文で持ってきて結果を表示します。

ミスマッチ検索の準備をする

ステップ1は、もうすでによくわかっている領域のはずです。これをやるために問い合わせ文はもういくつか書きました。ただ、ユーザのレスポンスをどこに取っておいて、他のユーザのレスポンスと比べることができるようになります。これはこのスクリプトのあとの方(ステップ3)でやります。以下のコードで配列\$user_responsesを作ります。ここにログインしたユーザのレスポンスを入れます。

```
$query = "SELECT mr.response_id, mr.topic_id, mr.response, mt.name  
AS topic_name ".  
"FROM mismatch_response AS mr ".  
"INNER JOIN mismatch_topic AS mt ".  
"USING (topic_id) ".  
"WHERE mr.user_id = '" . $_SESSION['user_id'] . "'";  
$data = mysqli_query($dbc, $query);  
$user_responses = array();  
while ($row = mysqli_fetch_array($data)) {  
    array_push($user_responses, $row);  
}  
ループが終了すると、  
$user_responses配列には、すべての  
ユーザのレスポンスが入ります。
```

この問い合わせ文はJOINを使ってユーザのレスポンスをすべて集めます。

whileループを使って、問い合わせ
結果の各行を舐め、このプロセスで
ユーザレスポンスの配列を作ります。

終了

- ❶ mismatch_responseテーブルからユーザのレスポンスを取ってきます。項目名とその回答とを結合することを忘れないで下さい。

ステップ2は、ミスマッチ相手発見用スクリプトを構築するプロセスで、いくつかの変数をセットアップします。個々のミスマッチ検索の結果を入れます。この変数はミスマッチ相手発見用スクリプトの全体を通して、最高のミスマッチを見つけるために使われます。

これが、これからミスマッチかどうかをチェックする人の
ユーザIDです…探索が終わったらこの変数には最高のミスマッチ相手のIDが入ります。

```
$mismatch_score = 0;  
$mismatch_user_id = -1;  
$mismatch_topics = array();
```

この変数には2人のユーザ間のミスマッチスコアを入れます。最高のスコアになったら、それが最高のミスマッチです。

この配列には2人のユーザ間でミスマッチした項目が入ります。

この変数の値が探索の後も-1のままだった場合、ミスマッチは見つからなかったということがわかります。このようなことが起こるのは、他のどのユーザもアンケートに答えていない場合です。ほとんどありえないことです。

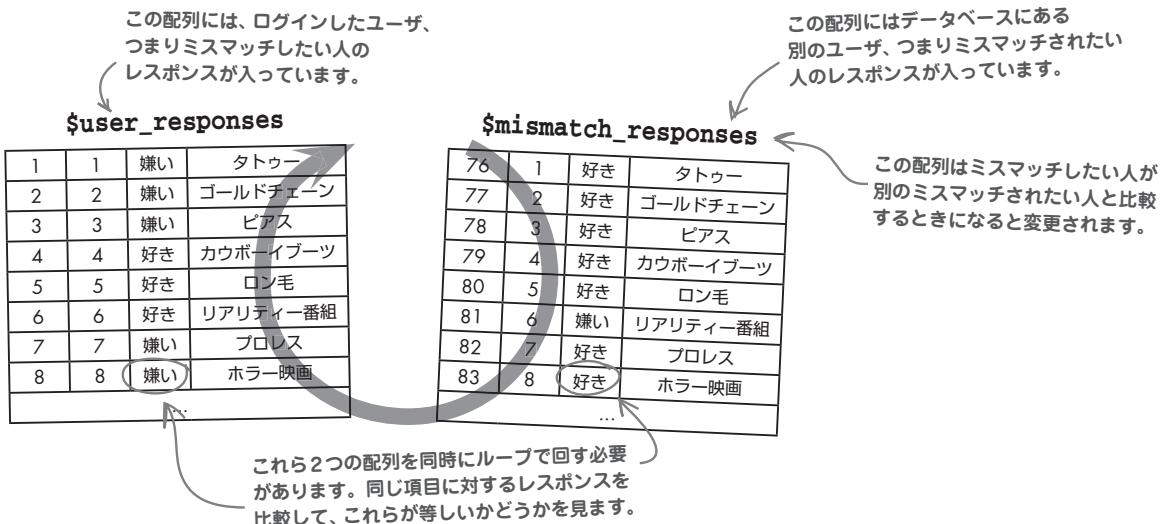
終了

- ❷ ミスマッチのサーチ結果を初期化します。このとき「最高のミスマッチ」を保存する変数も初期化しておきます。

ユーザの「ミスマッチ性」を比較する

次のミスマッチステップは、すべてのユーザをループすることです。全ユーザのレスポンスをログインユーザのレスポンスと比較します。言い換れば、ログインユーザ、つまりミスマッチしたい人（例えば加理奈さんです）を一人取ってきて、全ユーザテーブルを舐めて、彼女のレスポンスとミスマッチされたい人各々のレスポンスとを比較します。探しているのはミスマッチしたい人と反対のレスポンスを最も多く持っているようなミスマッチされたい人です。

どこから始めるのが良いでしょう？`$user_response`配列（ミスマッチしたい人のレスポンス）を回るループではどうでしょうか？ループの内側では、各要素を比較対象となっているもう1つの配列要素と比べます。もう1つの配列の方には、ミスマッチされたい人のレスポンスが入っています。こちらの配列を`$mismatch_response`という名前にします。



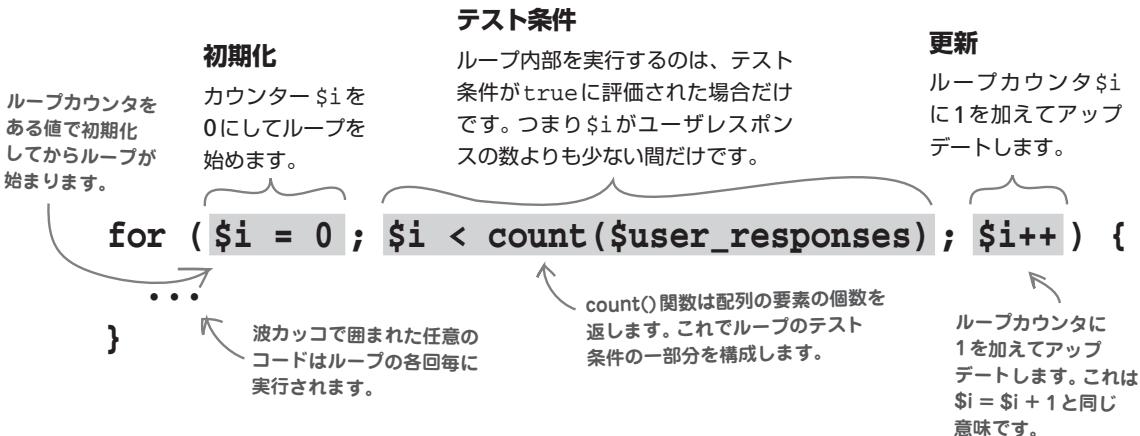
ここで難しいのは、同時に2つの配列を回るループが必要だということです。対応する要素を1対1で比較しなければならないからです。`foreach`ループは、ここでは使えません。なぜならこれは1つの配列を回すことしかできないからです。今必要なのは2つの配列を同時に回ることのできるものです。`while`ループでもイケることはイケます。ただしループカウンターの変数を作ってループの都度、自分で増分しなければなりません。理想的には、ループカウンター変数管理の面倒を自動でしてくれるループがあって、それを使って配列の各要素にアクセスできればよいのです。

~~`foreach (...) {`~~ 使えません！

`while (...) {` 使えますが、理想的ではありません。

必要なものは for ループです

PHPには別のタイプのループがあって、これにはミスマッチ相手のレスポンスを比較するために必要な機能がすべて整っています。名前は `for` ループといって、何かを **特定** の回数だけ繰り返すのに最適です。例えば `for` ループは、数えながらの処理に適しています。ゼロまでカウントダウンしながらの処理とか、またはある値までカウントアップしながらの処理などです。以下の `for` ループの構造を示します。ループがループカウンタ変数 (`$i`) を使って、どのようにステップ実行を構成するかを示しています。



ミスマッチ相手発見用スクリプトのステップ3では、2人のユーザについて、それぞれのレスポンスをループで回して比較し、最終的にミスマッチが何個あったかで「スコア」を計算します。以下の

データを基に、スコアを計算する `for` ループを仕上げて下さい。

<code>\$user_responses</code>	<code>\$mismatch_responses</code>	<code>\$score</code>
ユーザのレスポンスの配列。	ミスマッチするかも知れないユーザのレスポンスの配列。	ループ内で計算されるミスマッチのスコア。

```

for ($i = 0; $i < count($user_responses); $i++) {
    if (..... + ..... == ....) {
        .....
        array_push($topics, $user_responses[$i]['topic_name']);
    }
}
    
```

素朴な疑問 に答えます

 : **for**ループではなく単に**foreach**を使ってスコアを計算するというのはダメなのでしょうか?

 : **foreach**ループですべての異なるレスポンスに対して回ると言うことに対しては全く問題ないのですが、ループのある回に関する添字(\$i)がわからないのです。この添字が重要なのは、コードがユーザレスポンスの配列とミスマッチレスポンスの配列の両方にまたがって使っているためです。**foreach**ループだと2つの配列のうちどちらか一方の添字が必要になりますが、両方ではありません。そこで普通の**for**ループと添字を使って、各配列の対応する要素にアクセスする必要が出てくるわけです[†]。

 : ミスマッチしたレスポンスをそれぞれの配列に突っ込む目的は何でしょうか?

 : ミスマッチしたレスポンスの配列は、理想のミスマッチ相手とどんな話題で比較されたのかをきちんとユーザに知らせる上で重要な意味を持っています。単に理想のミスマッチ相手の存在を見つけ出すだけでは不十分です。その人との項目でミスマッチしたのかを共有できる方がイケてると思いませんか?これでミスマッチ結果がもう少し分かり易くなります。それにユーザには、何故特にこの人が本当にその人にとってものすごくミスマッチなのかという理由をちょっと多く教えてあげることができます。

 : ミスマッチサイトのスクリプトのステップ5で、あるユーザに対して最高のミスマッチが見つからないというのはどういう場合でしょうか?

 : ほとんどあり得ないのですが、一応想定しておかなければいけないシナリオは、全システム中に1人のユーザしかいない場合です。この場合、そのユーザに対して誰もミスマッチすることはできません。

[†] 訳注：このQ&Aはちょっと無理をして作った感じが否めません。本来なら**for**と**while**について、もっと議論すべきでしょう。実際**for**を使って書けるコードは、**while**を使っても同程度の読みやすさで書けます。前のページで**while**は、理想的ではないと書いてありましたが、現実的にどちらを使うかは趣味の問題です。


**エクササイズ
の答え**

ミスマッチ相手発見用スクリプトのステップ3では、2人のユーザについて、それぞれのレスポンスをループで回して比較し、最終的にミスマッチが何個あったかで「スコア」を計算します。以下のデータを基に、スコアを計算するforループを仕上げて下さい。

\$user_responses			
1	1	嫌い	タトゥー
2	2	嫌い	ゴールドチェーン
3	3	嫌い	ピアス
4	4	好き	カウボーイブーツ
5	5	好き	ロン毛
6	6	好き	リアリティー番組
7	7	嫌い	プロレス
8	8	嫌い	ホラー映画
...			

\$mismatch_responses			
76	1	好き	タトゥー
77	2	好き	ゴールドチェーン
78	3	好き	ピアス
79	4	好き	カウボーイブーツ
80	5	好き	ロン毛
81	6	嫌い	リアリティー番組
82	7	好き	プロレス
83	8	好き	ホラー映画
...			



ミスマッチレスポンスが
見つかる毎にスコアを
1ずつ増やします。

ユーザレスポンスの数は、項目の
総数と同じです。レスポンスは
アンケートから直接持って
くるからです。

\$score変数は0(ミスマッチ
なし)から項目の総数(完璧な
ミスマッチ)までの値を
取り得ます。

```
for ($i = 0; $i < count($user_responses); $i++) {
    if ( $user_responses[$i]['response'] + $mismatch_responses[$i]['response'] == 3 ) {
        $score += 1;
        array_push($topics, $user_responses[$i]['topic_name']);
    }
}
```

ループカウンタを使って、
各ユーザレスポンスを
ステップ実行します。

各ミスマッチ項目を配列に
加えて、何がミスマッチ
したのかをユーザに表示
できるようにしておきます。

ミスマッチは、好き(1)が
嫌い(2)にマッチするため、
ミスマッチがあるという
ことは、この2つを足すと
常に3になるということです。

終了

- ③ ユーザテーブルをループして、ユーザのレスポンス
と他の人のレスポンスとを比較します。

ミスマッチを仕上げる

ピカピカの新しいループでミスマッチスコアを計算しますが、これはもっと大きなスクリプト (mymismatch.php) の一部に過ぎません。これがユーザーの理想のミスマッチをミスマッチサイトのデータベースから見つけるための面倒をみてくれます。そして最後にその情報を表示してくれます。

このスクリプトでユーザーの完全なミスマッチを見つけます！



mymismatch.php

```
...  
// Only look for a mismatch if the user has questionnaire responses stored  
  
$query = "SELECT * FROM mismatch_response WHERE user_id = '" . $_SESSION['user_id'] . "'";  
$data = mysqli_query($dbc, $query);  
if (mysqli_num_rows($data) != 0) { ←  
    // First grab the user's responses from the response table (JOIN to get the topic name)  
  
    $query = "SELECT mr.response_id, mr.topic_id, mr.response, mt.name AS topic_name " .  
            "FROM mismatch_response AS mr " .  
    ① "INNER JOIN mismatch_topic AS mt " . ←  
        "USING (topic_id) " .  
        "WHERE mr.user_id = '" . $_SESSION['user_id'] . "'";  
    $data = mysqli_query($dbc, $query);  
    $user_responses = array();  
    while ($row = mysqli_fetch_array($data)) {  
        array_push($user_responses, $row);  
    } ← $user_responses配列には、ユーザーのすべてのレスポンスが入れてあります。  
  
    // Initialize the mismatch search results ミスマッチ検索結果を初期化  
    $mismatch_score = 0;  
    ② $mismatch_user_id = -1; ← この変数で、プログラムの進行に連れてミスマッチを探します。  
    $mismatch_topics = array();  
    ...
```

最初にユーザーのレスポンスを response テーブルから (項目名は JOIN して) 取得

よく知っている JOIN を使って、項目名を取ってきます。ユーザーのアンケートのレスポンスを SELECT してからです。

ちょっと待って下さい。
まだたくさんあります。
次のページへ行きましょう。

ユーザテーブルの探索。ユーザのレスポンスを他のユーザのレスポンスと比較する

```
// Loop through the user table comparing other people's responses to the user's responses
$query = "SELECT user_id FROM mismatch_user WHERE user_id != '' . $_SESSION['user_id'] . '';
$data = mysqli_query($dbc, $query);
while ($row = mysqli_fetch_array($data)) { ユーザ（ミスマッチ相手の候補）のレスポンステータを取得
    // Grab the response data for the user (a potential mismatch)
    $query2 = "SELECT response_id, topic_id, response FROM mismatch_response " .
        "WHERE user_id = '' . $row['user_id'] . '';
    $data2 = mysqli_query($dbc, $query2);
    $mismatch_responses = array();
    while ($row2 = mysqli_fetch_array($data2)) { 各ユーザーについて、この問い合わせ文でアンケートのレスポンスを取ってきて、ミスマッチの可能性について比較します。
        ③ array_push($mismatch_responses, $row2);
    }
    // Compare each response and calculate a mismatch total
    $score = 0; 各レスポンスを比較しミスマッチの合計を算出
    $topics = array();
    for ($i = 0; $i < count($user_responses); $i++) {
        if ((int)$user_responses[$i]['response'] + (int)$mismatch_responses[$i]['response'] == 3) {
            $score += 1;
            array_push($topics, $user_responses[$i]['topic_name']);
        }
    }
    // Check to see if this person is better than the best mismatch so far
    if ($score > $mismatch_score) { より良いミスマッチが見つかったため、ミスマッチ検索結果を更新
        // We found a better mismatch, so update the mismatch search results
        $mismatch_score = $score;
        $mismatch_user_id = $row['user_id'];
        $mismatch_topics = array_slice($topics, 0); もし今のユーザが今までの最高のミスマッチよりも良いミスマッチならば、彼を新しく最高のミスマッチにセットします。
    }
}
...

```

この波カッコ は、外側の }
whileループ
の終了を意味
しています。 // Compare each response and calculate a mismatch total

このforループがミスマッチスコアを計算し、ミスマッチ相手の可能性を探ります。

文字によるレスポンス（例えば'2'）を整数(2)に型変換して足したり比較したりできるようにします。

この人が今まで最高のミスマッチ相手よりも良いかどうかチェック

終了 ④ もし今のユーザが今までの最高のミスマッチよりも良いミスマッチならば、彼を新しく最高のミスマッチにセットします。

この関数は配列を分割して(slice)取ってきます。
この場合だと、\$topics配列を\$mismatch_topicsへ単にコピーするために使っています。

まだ最初のif文の中です。前のページから
続いています。まだまだコードは続きます…

```

ミスマッチが見つかったことの確認
// Make sure a mismatch was found
if ($mismatch_user_id != -1) { ←
    $query = "SELECT username, first_name, last_name, city, state, picture FROM mismatch_user " .
        "WHERE user_id = '$mismatch_user_id'"; ←
    $data = mysqli_query($dbc, $query);
    if (mysqli_num_rows($data) == 1) {
        // The user row for the mismatch was found, so display the user data
        $row = mysqli_fetch_array($data); ←
        if (!empty($row['first_name']) && !empty($row['last_name'])) { ←
            echo $row['last_name'] . ' ' . $row['first_name'] . '<br />'; ←
        }
        if (!empty($row['city']) && !empty($row['state'])) { ←
            echo $row['state'] . ', ' . $row['city'] . '<br />'; ←
        }
        echo '</td><td>';
        if (!empty($row['picture'])) { ←
            echo '<br />'; ←
        }
        echo '</td></tr></table>'; ←
        ミスマッチした項目を表示
    }
    // Display the mismatched topics
    echo '<h4>' . count($mismatch_topics) . ' 項目についてミスマッチしました。</h4>';
    foreach ($mismatch_topics as $topic) {
        echo $topic . '<br />'; ←
    }
    // Display a link to the mismatch user's profile ミスマッチしたユーザのプロフィールへのリンクを表示
    echo '<h4><a href=viewprofile.php?user_id=' . $mismatch_user_id . '>' . ←
        $row['first_name'] . 'さんのプロフィール</a>を見る。</h4>';
}
else {
    echo '<p>まず<a href="questionnaire.php">アンケート</a>答えてからミスマッチを探します。</p>';
}
...

```

ミスマッチの結果を表示する前に、「最高のミスマッチ」が本当に見つかったのかを確認します。

ミスマッチしたユーザ情報用の問い合わせ文です。これを表示します。

ユーザ名を表示します。

ユーザの都道府県名と市区町村名を表示します。

忘れずにタグも作って、ユーザの写真を表示します！

どの項目が実際にミスマッチに導いたのかを表示するというのは大事なことです。

最後のミスマッチしたユーザのプロフィールへのリンクを張っておきます。これでログインユーザは彼についてもっと知ることができます。

終了
5



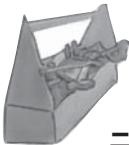
試運転

完全なミスマッチが見つかった！

ミスマッチサイトのスクリプトを修正して、新しく作ったミスマッチ相手発見用スクリプトを使えるようにします（またはアプリケーションをWebサイト（<http://www.oreilly.co.jp/books/9784873114446/>）からダウンロードして下さい）。これには、新しく `mymismatch.php` スクリプトを作る必要があります。同時に「ミスマッチ相手」というメニュー項目を `navmenu.php` スクリプトに追加し、ユーザがこのスクリプトにアクセスできるようにする必要があります。

できあがったらスクリプトをWebサーバにアップロードし、ミスマッチサイトのメインページ（`index.php`）をWebブラウザで開きます。自分がログインしたことを確認できたらアンケートを埋め、次に「ミスマッチ相手」のメニューをクリックして、自分のミスマッチ相手を見てみましょう。





PHP & MySQL 道具箱

ちょっとでは済まないMySQLデータベーステクニックについては、本章ではカバーし切れていません。
新しいPHPの小技についても言うに及びません。
ちょっと復習してみましょう！

スキーマとダイヤグラム

スキーマはデータベースのすべての構造(テーブル、カラムなど)を表現するもので、どのようにこれらがつながっているかも合わせて表現します。ダイヤグラムはデータベースを視覚的に記述するもので、特定のカラムがテーブルとの結合に関わる詳細についても記述します。

正規化

正規化とはデータベースの設計を変更する課程であって、ダブったデータをなくし、データの配置やデータの関係を改善することができます。この処理の目的は、肥大化するデータにも耐えうるような頑強な設計を提供することです。

for (...)

特定の数値での繰り返しに基づくループに適して作られたループです。ループを作るには、カウンタの初期化、テスト条件の確立、および各繰り返し毎にカウンタがどのようにアップデートされるかを指定します。

? :

この3項演算子はPHPの構成要素でとてもコンパクトなif-else文のように使えます。手軽に単純なtrue/falseを基準とする選択を実行することができます。

外部キー

テーブル内のカラムで別のテーブルとリンクするために使う。子テーブル内の外部キーは、通常親テーブル内の主キーと結合することで、2つのテーブル間の行を効率的に結びます。

AS name

このSQL文は別名を作ります。ここで付けられた名前はデータの一部を問い合わせ文の中で識別するために使います。別名は通常、問い合わせ文を単純化するために使います。長いテーブル名とカラム名を省略形で表記できるからです。別名はまた結果のデータの名前を付け替える場合にも使います。元々のテーブルカラムが十分に名前を特定しきれていない場合に使います。

INNER JOIN

この種類の結合は、2つのテーブルから一致する行を使ってデータを結合します。通常の問い合わせ文と違って、結合では2つ以上のテーブルからデータを取ってくることができます。このためデータベースが複数のテーブルからできている場合とても役に立ちます。

9章 文字列とカスタム関数

関数を通して より良い生活を



[†] 訳注：原文はblow up the moonです。あえて直訳しました。何か突拍子もないことをやる、という意味で使われる慣用句のようです。

関数によってアプリケーションは完全に新しいレベルになります。

すでにPHPの組み込み関数を使って何かを達成したことはたくさんあります。ここからさらに本当に役立つ組み込み関数をいくつか見ていくことにします。さらに続けて自分自身のカスタム関数の作り方を学びます。これで、これまでの想像を遥かに越えるところまで行き着くことができます。まあ、この章で達人の域まで昇りつめることは無理でしょうが、カスタム関数によってコードは効率的になり、再利用可能となります。

割のいい危険な仕事は見つけにくいものです

RiskyJobs.bizはインターネット操業開始で、適切な人材をその人に適した最も危険な仕事に割り当てることで会社の手助けをすることを目的として設計されました。ビジネスモデルは単純なものです。それぞれの危険な仕事に対して適切な候補者を割り当て、仲介手数料をもらうのです。適正が高ければ高いほど、仲介報酬も高くなります。

リスキージョブサイトは、サイトでの仕事検索機能を改良するため、助けを必要としています。現時点では、データベースは危険な仕事でパンク状態で、適正な人材を割り当ててもらうのを待っています。リスキージョブサイトの検索用フォームと裏でデータベースにぶち込まれている求人中の仕事をちょっと見てみましょう。

この単純なフォームがスクリプトを呼び出し、riskyjobsテーブルを検索します。

リスキージョブサイトの検索用フォームは、riskyjobsテーブルの問い合わせ文を駆動し、マッチする仕事を検索します。

リスキージョブ：危険な仕事検索

危険な仕事を検索します。

検索

riskyjobsテーブルは、仕事の名称と詳細、および位置情報とそれぞれの仕事が投稿された日付を保持しています。

job_id	title	description	state	city	zip	company	date_posted
1	マタドール (Matador)	多忙な乳製品工場にて...	新潟	上越	057-0123	牛乳工場首ったけ	2009-03-11 10:51:24
2	パパラッチ (Paparazzo)	トップセレブ...	東京	六本木	902-1010	大物追跡専	2009-03-24 10:51:24
3	サメの調教師 (Shark Trainer)	新しくウォーターテーマパーク...	宮崎	都城	328-0123	サメ嗜み専	2008-04-28 03:12:45
4	消防士 (Firefighter)	データビレ市では...	千葉	データビレ	454-4888	データビレ市	2009-05-22 12:34:17
5	電圧検査士 (Voltage Checker)	屋外にて直流...	兵庫	神戸	277-0100	ショックシステム (有)	2009-06-28 11:16:30
6	ワニの歯医者 (Crocodile Dentist)	動物好きで...	福岡	博多	341-3901	強欲爬虫類	2009-07-14 10:51:24
7	カスタードウォーカー (Custard Walker)	人間がカスタード上...	広島	尾道	8710-4301	パイ・テクノロジー	2009-07-24 10:54:05
8	電動暴れ牛修理士 (Electric Bull Repairer)	ハンクスホンキートンクでは...	神奈川	横須賀	070-20200	ハンクスホンキートンク	2009-07-27 11:22:28

投稿されたそれぞれの仕事は、job_id主キーで一意に識別されます。

検索結果の表示へ！



自分で考えてみよう

リスキー ジョブサイトのフォームが「提出」されると、検索文字列が変数\$user_searchに突っ込まれて、以下のSQL問い合わせ文に組み込まれ、実際に検索を行います。反対側のページにあるriskyjobsデータベースから何行がゴン太の検索結果として見つかるか書いて下さい。

```
$search_query = "SELECT job_id, title, state, description FROM riskyjobs " .  
"WHERE title = '$user_search'" ;  
  
$result = mysqli_query($dbc, $search_query);
```

ここに答えを書いて下さい！ →

自分で考えてみよう の答え

WHERE節で`=`があると
2つの文字列を比較して
正確に一致しなければ
なりません。

リスクジョブサイトのフォームが「提出」されると、検索文字列が変数`user_search`に突っ込まれて、以下のSQL問い合わせ文に組み込まれ、実際に検索を行います。反対側のページにある`riskyjobs`データベースから何行がゴン太の検索結果として見つかるか書いて下さい。

この変数はテキストボックスに打ち込まれた
すべてのものを含んでいます。

```
$search_query = "SELECT job_id, title, state, description FROM riskyjobs ".  
    "WHERE title = '$user_search'";  
  
$result = mysqli_query($dbc, $search_query);
```

タコ、ゼロ、無！問題なのは、問い合わせ文が
あまりにもドンピシャでなければならないこと
です。ユーザが打ち込んだ文字列とドンピシャで
一致したものだけなのです。 → 0

今の検索だとエラーに対するゆとりが全くありません

リスクジョブサイトのスクリプトにおいて、このSELECT問い合わせ文は
ものすごく厳格で、2つの文字列を比較して完全に同一のときだけしかマッチ
しません。これだと今やろうとしている検索では問題が露呈します。入力する
側の人は、検索用の用語をリストに書かれているものとマッチさせなければな
らないからです。仕事の名称がちょっとでも違っただけでもうダメです。

ゴン太の検索に戻ってみます。この検索では、`riskyjobs`テーブルの
`title`カラムから「Bull Fighter Matador」[†] を検索するという問い合わせ文
です。

検索用の用語は大文字、小文字を
区別しません。これはMySQLの
WHERE節のデフォルトです。

`SELECT job_id, title, description FROM riskyjobs`

`WHERE title = 'Bull Fighter Matador'`

=演算は、2つの文字列を比較して
等価を判定する場合、完全一致です。

問題が見えてきましたか？この問い合わせ文は、テーブル内でタイトルカラ
ムが「Bull Fighter Matador」というドンピシャのテキストの場合に限り、その行
とマッチするのです。仕事の名称が「Matador」だけではマッチしません。も
ちろん「Firefighter(消防士)」や「Electric Bull Repairer(電動暴れ牛修理工)」もマ
ッチしません。もちろん、後ろの2つにはマッチしなくていいのです。しかし検索
が思った通りに動いてくれていないことに変わりはありません。さらに大文字、
小文字を混在させた場合にも問題が起こります。(MySQLの検索はデフォルト
では**大文字、小文字を区別しません**。)検索用の文字列全体が、WHERE節の等
価(`=`)演算によって完全一致をしなければならないのです^{††}。

[†] 訳注：Bull Fighterとは闘牛士の
ことで、Matadorとはマタドール
(闘牛士の主役)のことです。

^{††} 訳注：MySQLのデフォルト設定
では大文字、小文字を区別しな
いで検索するが、`=`によって等価
比較をするときは大文字、小文字
を区別するため問題となるとい
うことを言っています。

SQL問い合わせ文はLIKEを使って柔軟にできます

今必要なのは、データベースから文字列の任意の一部分を取り出してマッチすものを検索する方法なのです。SQLでは、このようなことをするためにLIKEというキーワードを用意していて、WHERE節で返される一致条件にこの種の柔軟性を付与してくれます。LIKEというのは、=演算の非常に緩いバージョンと思っても良いかもしれません。以下の問い合わせ文を見てみましょう。これはLIKEを使って行をマッチさせていますが、単語として「fighter」がタイトルカラムのどこかに入っているべきです。

```
SELECT job_id, title, description FROM riskyjobs
```

```
WHERE title LIKE '%fighter%'
```

キーワードLIKEで、引用符の中に
ある語と完全に同じではない検索
条件で探してくれます…もちろん
大文字、小文字を区別しません。

%記号は、ワイルドカード(万能)です。
ここでは、語の前後に0文字以上のどんな
文字の代理も引き受けってくれます。

LIKEで、マッチするものを簡単に見つけられるようになります。特に長い語
やフレーズの一部として検索文字列をマッチさせる必要がある場合に有用です。
以下の例を確認して下さい。先ほどの問い合わせ文にマッチした文字列です。

firefighter

FightErnestoPlease

Prize Fighter[†]



マニア向け情報

LIKE節はワイルドカード文字と一緒に使われる
のが普通です。ワイルドカード文字は、マッチさせる
データの任意の文字の代理です。SQLではワイルドカードはパーセント記号(%)で、ゼロ文字以上
の任意グループの文字を表します。問い合わせ文の
中で、このワイルドカードを検索する語の前後に置くことで、上記のSELECT文のように、SQLはデータのどこかに語が表れれば結果を返してくれます。
語の前後に何文字あるかとは関係ありません。

SQLには他にももう1つワイルドカード文字があつて、LIKEと一緒に使うことができます。これは下線(_)で、何か1文字を表します。次のようなLIKE節を考えてみます。

```
LIKE '_ _ _ _fighter%'
```

この節は、「文字列のうち「fighter」の前に4文字あつて、後ろには任意の文字があるものを見つける」ことを表します。これにより「bullfighter」や「firefighter」はマッチしますが「streetfighter」はマッチしません。

[†] 訳注：Prize Fighterとはプロボクサーという意味です。



時間です！よく見てリスキージョブサイトのデータベースを理解したら…
ちょっと検索をしてみましょう。

リスキージョブサイト・アプリケーション用にriskyjobs.sqlファイルをサイト(<http://www.oreilly.co.jp/books/9784873114446/>)からダウンロードします。このファイルにはriskyjobsテーブルを作って、サンプルデータを入れるSQL文が書いてあります。

riskyjobs.sqlに書いてある文をMySQLツールで実行したら、いくつか問い合わせ文を実行して、仕事の検索をシミュレートしてみて下さい。以下、まず試してみるべきいくつかの例です。

SELECT * FROM riskyjobs ← この問い合わせ文はriskyjobsテーブル内のすべての仕事に関するすべてのカラムを取ってきます。

SELECT job_id, title, description FROM riskyjobs
WHERE title = 'Bull Fighter Matador' ← この問い合わせ文は、仕事のうち、ジョブIDとタイトルと説明を取ってきますが、タイトルが「Bull Fighter Matador」と完全に一致するものだけです。

SELECT job_id, title, description FROM riskyjobs
WHERE description LIKE '%animals%'

この問い合わせ文は、
LIKEを使って仕事を見つけるので、
業務概要のどこかに「animals」と
書いてあるものが見つかります。

ダウンロードして下さい！



リスキージョブサイトのアプリケーションの全ソースコードは、以下のWebサイトからダウンロードできます。

<http://www.oreilly.co.jp/books/9784873114446/>



LIKE節マグネット

いくつかのLIKE節マグネットが冷蔵庫に貼ってあります。それぞれ適切な結果と節とをマッチさせることができますか？どのLIKE節にもマッチしないマグネットはどれか分かりますか？

複数の答えがある
ものもあります。

LIKE '%er'

LIKE '% T%'

LIKE 'c%'

LIKE '%test %'

LIKE '%Tipper Cow%'

LIKE '%do_'

LIKE '%ma%'

Human Cannonball

Cliff Diver

Pet Food Tester

Team Mascot

Crash Test Dummy

Matador

Rodeo Clown

Cat Herder

Politician

Cow Tipper

Snake Charmer

Shark Finder



LIKE 節マグネットの答え

いくつかのLIKE節マグネットが冷蔵庫に貼ってあります。それぞれ適切な結果と節とをマッチさせることができますか？どのLIKE節にもマッチしないマグネットはどれか分かりますか？

LIKE '%er'

「er」で終わるものなら何でも、これにマッチします。

Pet Food Tester

Shark Finder

Cow Tipper

Cliff Diver

Snake Charmer

Cat Herder

LIKE '% T %'

これにマッチさせるためには「T」の前に空白が1つ必要です。

Pet Food Tester

Cow Tipper

Crash Test Dummy

SQL問い合わせ文では大文字、小文字を区別しません。つまり小文字でも大文字でも「c」が始まれば、どちらでもこの問い合わせ文にマッチします。

LIKE '%test %'

大文字、小文字は関係ありません。

Crash Test Dummy

Cow Tipper

Cliff Diver

Cat Herder

LIKE '%do_ %'

「do」の後に1文字だけあります。

Matador

2文字「ma」がどこかに現れればマッチします。

LIKE '%ma%'

Human Cannonball

Team Mascot

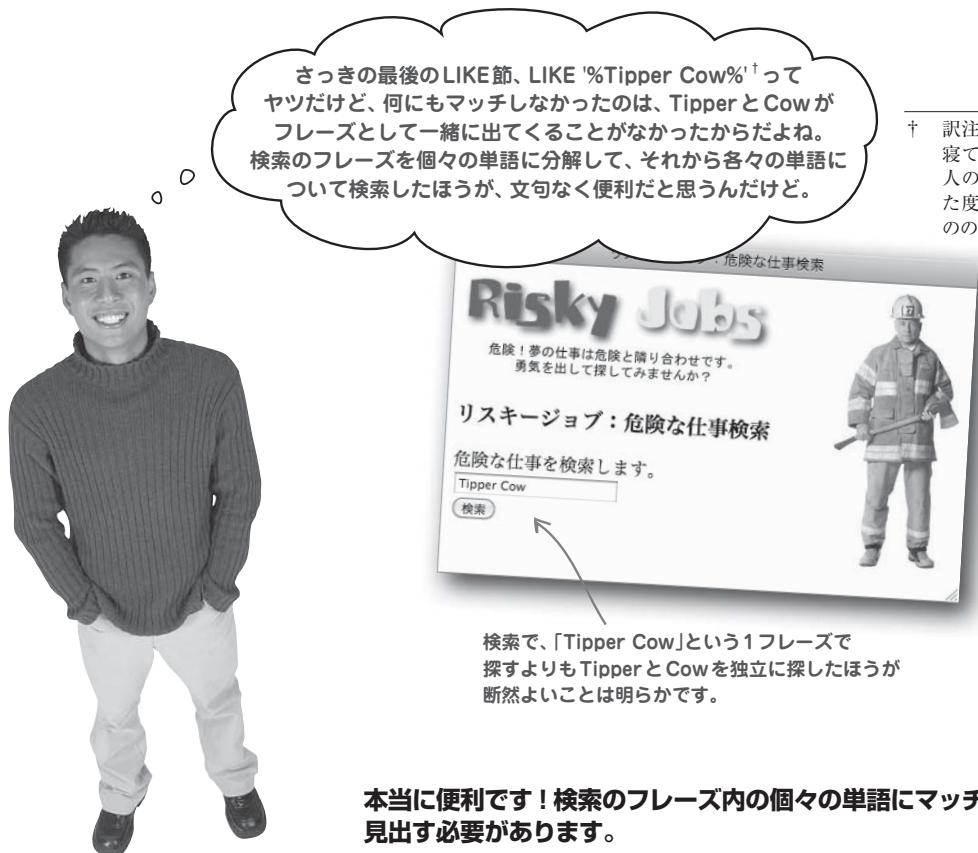
これはマッチしなかったフレーズです。

Rodeo Clown

Politician

LIKE '%Tipper Cow%'

このLIKE節は、何ともマッチしません。



[†] 訳注：Cow Tipperとは
寝ている牛を押し倒す
人のことで、牛を使っ
た度胸試しのようなも
ののようです。

本当に便利です！検索のフレーズ内の個々の単語にマッチさせる方法を見出す必要があります。

リスクジョブサイトの検索フィールドにユーザがタイプしたものをそのまま取つてきて、検索条件としただけでは、いつもうまくいくとは限りません。検索をもっと効率的にするには、入力された各検索語で検索すべきであって、検索フレーズ全体で検索すべきではありません。しかしどうすれば複数語での検索ができるでしょうか？まず、各検索語を配列に突っ込み、それから SELECT 問い合わせ文を微調整して、個々の単語を独立に検索すれば良いのです。

文字列を単語単位に爆破(explode)する

リスキージョブサイトの検索機能をもっと使いやすくするには、ユーザが入力した検索文字列を分割した方がよい場合があります。つまりユーザが複数の単語をフォームフィールドに入力した場合がそうです。危険な仕事を探している人が検索用フォームに入力するデータは、テキストです。つまり任意の PHP 組み込み文字列操作関数を使ってこれを処理することができます。explode() という超強力な関数があって、これを使うと文字列を粉々にして分離した文字列の配列にすることができます。以下に例を示します[†]。

explode() 関数は、文字列をぶった切って共通のセパレータに基づく部分文字列の配列にします。セパレータは分離子(デリミッタ)とも言います。

このパラメタが、これから爆破したいテキストです。

[†] 訳注: PHP の explode() 関数は日本語文字対応(正確にはマルチバイト文字対応)ではありません。ここでの利用では大丈夫ですが、区切り文字によっては発狂する可能性があります。

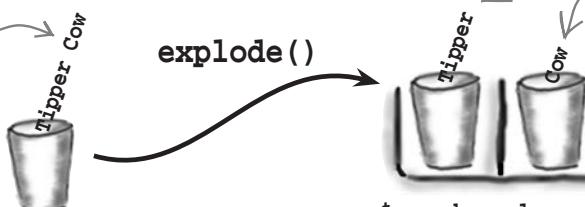
```
$search_words = explode(' ', 'Tipper Cow');
```

\$search_words 変数には、ここで検索用の単語の配列が入ります。これがこの後 SQL 問い合わせ文の種となります。

このパラメタで指定したものを使って explode() は文字列を部分文字列に分割します。この場合で言うと空白文字が指定されています。ここに複数文字を指定することもでき、分離子(デリミッタ)と呼ばれます。

explode() 関数は2つのパラメタを取ります。最初のパラメタは、分離子(デリミッタ)で、1 文字または複数文字で、どこで文字列を分割するかを指定します。今回の場合、分離子として空白文字を使っています。これは検索文字列は空白が現れればどこでも分割されることを意味します。分離子自体は、結果の部分文字列には含まれません。2番目のパラメタが爆破(explode)される文字列です。

空白分離子で文字列をどのように爆破するかをコントロールします。



各部分文字列は異なる配列要素として格納されます。

検索用単語の配列をリスキージョブサイトに組み込むには、コードを1行追加してから、リスキージョブサイトのデータベースに対して問い合わせ文を実行する必要があります。誰かが「Tipper Cow」と検索フィールドに打ち込んだ場合、以下のコードで2語に分割し、それぞれの語を配列(\$search_words)に突っ込んでくれます。

```
$user_search = $_GET['usersearch'];
$search_words = explode(' ', $user_search);
```

explode() 関数は \$user_search の各語を \$search_words という配列に突っ込みます。



エクササイズ

爆破した検索語をリスクジョブサイトのアプリケーションに組み込むには、LIKE と OR を使って各単語をSQLのSELECT問い合わせ文にはり付ける必要があります。例えば、ゴン太の最初の検索「Bull Fighter Matador」では、問い合わせ文は次のような感じになるでしょう。

```
SELECT * FROM riskyjobs
WHERE description LIKE '%Bull%' OR description LIKE '%Fighter%' OR
description LIKE '%Matador%'
```

ここでは、タイトルではなく業務概要を検索しています。説明の方が多い情報がありマッチしやすいためです。

では、以下のPHPコードを使ってこの問い合わせ文を組み立てようとしているものとします。検索用のデータは、ユーザがリスクジョブサイトの検索用フォームに打ち込んだものを使用します。

```
$search_query = "SELECT * FROM riskyjobs";
$where_clause = '';
$user_search = $_GET['usersearch'];
$search_words = explode(' ', $user_search);
foreach ($search_words as $word) {
    $where_clause .= " description LIKE '%$word%' OR ";
}
```

```
if (!empty($where_clause)) {
    $search_query .= " WHERE $where_clause";
}
```

このコードにより生成されるSQL問い合わせ文を書いてみて下さい。ゴン太は、「Bull Fighter Matador」と打ち込んで検索したものとします。次に問題と思われるがあれば、それを書き留めておいて下さい。

.....
.....
.....





エクササイズ の答え

爆破した検索語をリスクジョブサイトのアプリケーションに組み込むには、LIKEとORを使って各単語をSQLのSELECT問い合わせ文にはり付ける必要があります。例えば、ゴン太の最初の検索「Bull Fighter Matador」では、問い合わせ文は次のような感じになるでしょう。

```
SELECT * FROM riskyjobs
WHERE description LIKE '%Bull%' OR description LIKE '%Fighter%' OR
description LIKE '%Matador%'
```

ここでは、タイトルではなく業務概要を検索しています。説明の方が多くの情報がありマッチしやすいためです。

では、以下のPHPコードを使ってこの問い合わせ文を組み立てようとしているものとします。検索用のデータは、ユーザがリスクジョブサイトの検索用フォームに打ち込んだものを使用します。

```
$search_query = "SELECT * FROM riskyjobs";
$where_clause = '';
$user_search = $_GET['usersearch'];
$search_words = explode(' ', $user_search);
foreach ($search_words as $word) {
    $where_clause .= " description LIKE '%$word%' OR ";
}
WHERE節が空でないことを確認してから、
検索問い合わせ文にくっ付けます。
```

それぞれのLIKE節はORで終わっていて、次の節とつながります。これはちゃんと動きそうですが、最後だけがダメです。

このコードにより生成されるSQL問い合わせ文を書いてみて下さい。ゴン太は、「Bull Fighter Matador」と打ち込んで検索したものとします。次に問題と思われることがあれば、それを書き留めておいて下さい。

SELECT * FROM riskyjobs

WHERE description LIKE '%Bull%' OR description LIKE '%Fighter%' OR

description LIKE '%Matador%' OR



問い合わせ文の最後に余分なORがあるため、問い合わせ 자체が失敗に終わります！

† 訳注：原文の説明が正確でないので補足します。左辺の文字列の後ろに右辺の文字列をくっ付けて、それが左辺の文字列の新しい結果となります。

implode()[†]は部分文字列から文字列を作ります

今本当に必要なのは、WHERE節のLIKEの間にORを置いてあげることだけです。ただし最後には置いてはダメなのです。そのところをちゃんとやるにはどうすればよいでしょう？最後の場合だけを特別なケースとしてループの中で検索語の最後かどうかを調べる、というはどうでしょうか？その場合だけはORを置かないというわけです。それでも動きますが、ちょっと面白いです。もっとスマートなやり方があって、それはexplode()と逆の動作をする関数を使うのです。implode関数は文字列の配列を取って、それらを使って1つの文字列を作ってくれます。

[†] 訳注：PHPのimplode()関数もexplode()関数と同様、日本語文字（正確にはマルチバイト文字）に対応していません。使い方によっては発狂する可能性があります。

```
$where_clause = implode(' OR ', $where_list);
```

implode()関数は、単一の文字列を返します。

でもこの関数が問い合わせ文の後にぶら下がっているORの問題をどのように解決してくれるのでしょうか？実は、implode()というのは文字列を結合しようとしたら、文字列の間に指定した文字列を置いてくっ付けてくれます。そこで' OR 'を分離子に使うと、各LIKE節の間にだけORがあるようなWHERE節を作り上げることができます。

これが分離子（デリミッタ）で、いくつかの文字列を1つに貼り付けるとき、各語の間に押し込まれます。

こっちは文字列の配列でなければなりません。これらをまとめて1つにします。

自分で考えてみよう

PHPコードを書き直してリスクイージョブサイトのSELECT文をちゃんと生成するようにして下さい。このSELECT文は、implode()関数を使うことで余計なORがくっ付くという問題を直すものとします。

自分で考えてみよう の答え

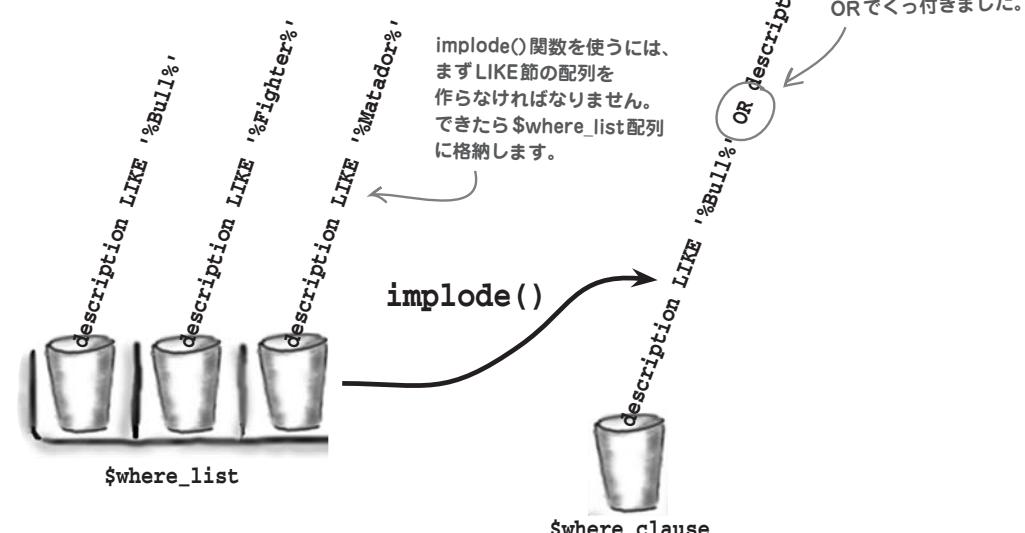
PHPコードを書き直してリスクジョブサイトのSELECT文をちゃんと生成する
ようにして下さい。このSELECT文は、implode()関数を使うことで余計なORが
くっ付くという問題を直すものとします。

```
$search_query = "SELECT * FROM riskyjobs";
$where_list = array();
$user_search = $_GET['usersearch'];
$search_words = explode(' ', $user_search);
foreach ($search_words as $word) {
    $where_list[] = "description LIKE '%$word%'";
}
$where_clause = implode( ' OR ', $where_list);
if (!empty($where_clause)) {
    $search_query .= " WHERE $where_clause";
}
```

implode()は、これからくっ付ける文字列の配列を受け取るので、LIKE節の配列を作る必要があります。

こんな使い方をすると、□演算子はarray_push()関数と同じような動作をします。新しい要素を配列の最後に追加してくれます。

implode()に渡される分離子はORですが、その前後に空白が必要です。





リスキージョブサイトの検索用フォームに試乗してみる

リスキージョブサイトのアプリケーションをWebサイト <http://www.oreilly.co.jp/books/9784873114446/> からダウンロードします。search.phpスクリプトには、今までやってきたコードを生成する問い合わせ文が含まれています。この問い合わせ文は、search.htmlページのフォームに打ち込まれた検索データを処理に使います。

スクリプトやその他のリスキージョブサイト関連ファイルをWebサーバにアップロードしたら、検索用のフォーム(search.html)をWebブラウザで開きます。いくつか検索を試してみて、問い合わせ文を生成するコードがどのようにうまくやってくれるか確認して下さい。ゴン太の「Bull Fighter Matador」検索がどうなるかも忘れずに試して下さい。これは implode() コードを確認するには優れたテストケースです。

Risky Jobs

危険！夢の仕事は危険と隣り合わけです。
勇気を出して探してみませんか？

リスキージョブ：危険な仕事検索の結果

仕事の名称	業務概要	都道府県	更新日時
プロボクサー (Prize Fighter)	来るべきスーパーフライ級ボクサーを募集しています。チャンピオンの連勝記録を伸ばしてください。鈍いファットワーク、ダメダメなジャブ、ガラスのアゴの持ち主を求めています。未経験者募集です。フルタイムの正社員としての雇用ではありません。懸賞金の分配はリングの後ろにある路地で行います。ロングギングがあなたを挑戦者にします。または少なくとも1敗の手助けをします。	東京	2009-11-14 11:31:08
闘牛士 (Torreador)	可愛らしいイシクが、あなたの優しいマント使いの卓識したスキルを待っています。闘牛士達は基本検査にパスしている必要があります。	山形	2009-11-14 21:49:31
電動暴れ牛修理士 (Electric Bull Repairer)	Hank's Honky Tonkでは、経験豊富な電動暴れ牛修理士を募集しています。特典として(修理後の)フリーライドおよび手羽先唐揚げの半額などがあります。	神奈川	2009-07-27 11:22:28

ゴン太にとって完璧な「危険な仕事」が即座に見つかったわけではありません。でも間違いなく多少の進展はありました。検索用スクリプトが個々の検索語を独立に探してくれるからです。

たくさんの仕事が出てきた！
プロボクサーやら電動暴れ牛の修理を大喜びで
やりたい、とは思わないけど、ここで夢の
仕事が見つかる…かもれない…そのうち。

This PDF is prepared for tools@maruhouse.org personal use only.

447

**アタシ綱渡りができるの。
このサイトに来たけど、ろくな仕事が
見つからなかつたわ。検索語は
ちゃんと入れたわよ。**

危険！夢の仕事は危険と隣り合わせです。
勇気を出して探してみませんか？

リスキージョブ：危険な仕事検索

危険な仕事を検索します。
tightrope, walker, circus

検索

危険！夢の仕事は危険と隣り合わせです。
勇気を出して探してみませんか？

リスキージョブ：危険な仕事検索

危険な仕事を検索します。
tightrope, walker, circus

検索

検索語は明らかにサーカスの綱渡りを
探そうとしています。しかし結果は
全然ちゃんとマッチしていません。

Risky Jobs

危険！夢の仕事は危険と隣り合わせです。
勇気を出して探してみませんか？

リスキージョブ：危険な仕事検索の結果

仕事の名称
猫ジャグラー
(Master Cat Juggler)

業務概要
今や失われつつある猫ジャグリングに従事していますか?40ヶ国
以上で禁止されていく中、ジムライズサーカスだけが猫ジャグリ
ングを洗練させ現代の観客をうけますよう進化させています。ウデ
を磨いて私たちのサーカスで猫ジャグリングの初日を飾りましょ
う。シンクロ猫ジャグリングをマスターすることのできる地球上
で唯一の場所です。実際ジャグリングの力が渠に面倒を見るより
も、むしろ難しいのです。従業員のみなさんは全員平日にはチャ
ンスを与えられます。我々のチーム入りすることをお待ちしてい
ます。しっかりとケージをご用意ください。猫の扱いが器用
であることを証明するためのテストに必要です。猫ジャグリング
ます。

都道府県 高知 更新日時 2009-11-14 21:13:35

Are you a practitioner of the lost art of cat juggling? Banned in forty
countries, only the Jim Ruiz Circus has refined cat juggling for the
sophisticated tastes of the modern audience. Ply your trade with
premiere cat jugglers at our circus, the only place on earth to master
synchronized cat juggling. It's true, juggling them is even harder than
herding them. We are an equal opportunity employer, and look
forward to adding you to our team. Please be prepared to undergo a
thorough battery of tests to prove your deft handling of felines. Only
the cream of the crop will be accepted into our Master Cat Juggler
program.

検索語に何か落ち度
があるのでしょうか？
それとも綱渡りの
仕事なんて本当に
ないのでしょうか？

綱渡り用ロープのテスト
(Tightrope Tester)

非常に高いところに何時間もぶら下がって考え事をすることが好きなら、この仕事はあなたにうってつけです。私どもの綱渡り用ロープは厳格な43ポイントテストのパスしたものです。これは、人間が一定期間以上ぶら下がるという条件では最高水準を確保しています。ぶら下がるのはあなたです！安全用のネットは提供しますが、ヘルメットおよび手袋はご持参ください。福島の大天井にある私たちの工場では、各種雇用パッケージプランにて優遇い



自分で考えてみよう

おーちゃんが自分の仕事を検索するために打ち込んだ「tightrope, walker, circus」[†] に対して作られるSQL問い合わせ文を書いて下さい。そしてこの問い合わせ文が持っている問題点を記入して下さい。

.....
.....
.....

[†] 訳注：これらの単語の意味は、それぞれ「綱渡り用ロープ、歩く人、サーカス」です。

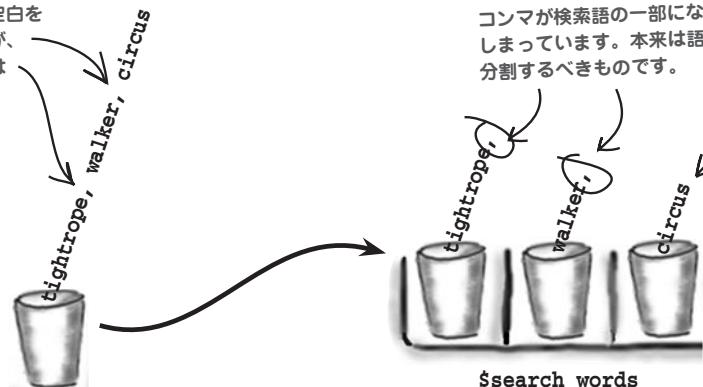
自分で考えてみよう の答え

おーちゃんが自分の仕事を検索するために打ち込んだ「tightrope, walker, circus」に対して作られるSQL問い合わせ文を書いて下さい。そしてこの問い合わせ文が持っている問題点を記入して下さい。

```
SELECT * FROM riskyjobs
```

```
WHERE description LIKE '%tightrope,%' OR description LIKE '%walker,%' OR  
description LIKE '%circus%'
```

`explode()` 関数は
分離子として空白を
使っていますが、
コンマのことは
全く考えて
いません。



コンマが検索語の一部になってしまっています。本来は語を分割するべきものです。

検索語のうち、実際にマッチする仕事のために使われるのは「circus」だけです。これだけがコンマが間違って付いてしまっていないためです。

大した問題じゃないじゃない。
単に `explode()` 関数を2回呼べばいいじゃない。
1回目で空白を捨てて、次にコンマを捨てるの。



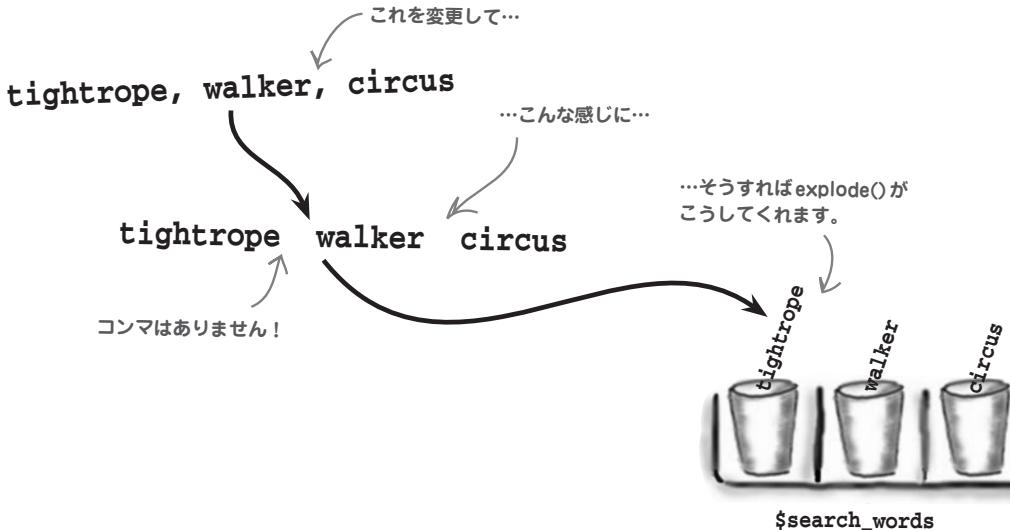
`explode()` 関数は、1つの文字列を複数の部分文字列に爆破してしまいます。今の問題はすでに複数の部分文字列になっているということです。

1回目に `explode()` 関数を呼び出すと、複数の文字列が配列にぶち込まれてしまいます。つまり爆破すべき1つの文字列はもうありません。ここで文字列をさらに爆破しようとするのは、さらなる問題を引き起こしそうです。分離子の問題を `explode()` 関数の複数回の呼び出しで解決しようとするよりも、検索文字列を前処理して分離子を1つだけにしてから、`explode()` を呼び出した方が良いでしょう。そうすれば最良の結果が得られます。文字列を1つの分離子で分解するだけです。

検索文字列の前処理

`explode()` 関数1発だけで文字列をきれいに爆破したいのです。どうすればよいでしょう? `explode()` 関数は、例えば空白文字のような1つの分離子しか気にしてくれないということを忘れないで下さい。つまり、検索文字列を前処理して、各検索語は空白で分離されているようにしなければならないということを意味しているのです。たとえユーザがコンマを打ち込んだ場合でもそうなのです。

データを前処理することで不要な文字をなくし、データを簡単に処理することができるようになります。



素朴な疑問に答えます

Q: 検索文字列を破壊する際に、分離子として2文字以上を使うことができますか?

A: できます。分離子として何文字を指定することもできます。でもこれは異なる分離子を指定することと同じではありませんから、今直面している問題の解決策にはなりません。

例えば、`explode(' ', ' ', $user_search)`と指定して文字列を分離したとすると、コンマと空白を組み合わせて分離子となります。ですから誰かが、「tightrope, walker, circus」と打ち込んでくれば、ちゃんと動きます。ただし別の誰かが「tightrope walker circus」と打ち込んだらダメです。こっちの場合は、1つの長い文字列になってしまいます。全然イケません。

Q: コンマを空白に変更するのではなく、単に削除してしまうというのはどうでしょう?

A: このやり方でうまくいくのは、ユーザが検索語をコンマと空白の両方を使って分離していた場合だけです。これではうまくいきません。もしコンマを削除してしまったら、例えば「tightrope,walker」を「tightropewalker」に変換してしまうというリスクを負うことになります。これでは恐らくリスキーなサイトのデータベースでは何もマッチしないと思われます。

不要な検索文字を置き換える

置き換えについて考るるのであれば、リスキージョブサイトの検索文字列でやるべきことはワープロの置換を使うことと同じです。今の場合、コンマを見つけて空白に置き換えたいわけです。PHPのstr_replace()に3つのパラメタを渡すだけで、これができます。3つのパラメタとは、探しているテキスト、置き換えたいたテキスト、それと置換を実行したいテキストです。以下にstr_replace()の動作例を示します。

```
この部分文字列を置き換える  
つもりです…
```

…そしてそこには代わりに
この文字列が入ります。

```
$clean_search = str_replace('thousands', 'hundreds',  
'Make thousands of dollars your very first month. Apply now!');
```

↑
第3のパラメタが変更される文字列です。
thousandsをhundredsに置き換える
ことで広告に少し真実味を加えています。

では検索文字列の中にあるコンマについてはどうでしょうか？str_replace()関数[†]は、個々の文字を同様に置き換えてくれます。

```
覚えましたね。これが置き換え  
たい部分文字列で…
```

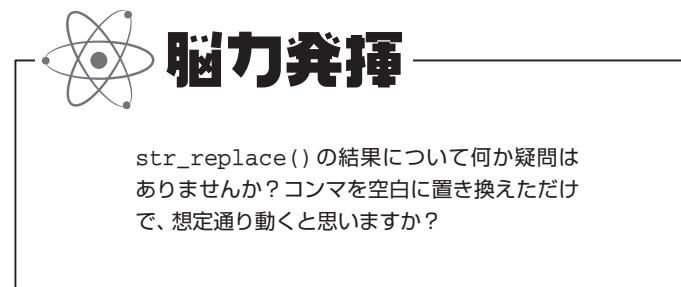
…そしてこれが置き換え後の
文字列です。

```
$clean_search = str_replace(',', ' ', ', 'tightrope, walker, circus');
```

この文字列の中で現れるコンマは
すべて空白に置き換わります。

このコードを実行すると、変数\$clean_stringには“tightrope walker circus”という文字列が入ります。

[†] 訳注：この関数のマルチバイト対応版（日本語対応版は、PHPの標準関数としては用意されていません。ただし、ここでの使用方法に関してはstr_replace()のままで特に問題ありません。





以下のPHPコードが与えられているとします。次のような検索文字列が入っているとすると、\$search_words配列への出力がどうなるかを示して下さい。データは適切な配列要素(を表すコップ)に書き入れて下さい。\$search_words配列要素よりもデータの方が少なければ、要素(を表すコップ)は、×印で消して下さい。

```
$clean_search = str_replace(',', ' ', $user_search);
$search_words = explode(' ', $clean_search);
```

bull,matador cape[†]



\$search_words

空白3つ！
bull matador cape



\$search_words

bull , matador cape



\$search_words

空白2つ！
bull,matador, cape



\$search_words

† 訳注：これらの意味は順に、牛、マタドール、ケープ
(闘牛士が持っている赤い布)です。



**エクササイズ
の答え**

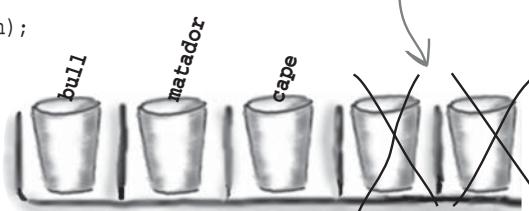
以下のPHPコードが与えられているとします。次のような検索文字列が入っているとすると、\$search_words配列への出力がどうなるかを示して下さい。データは適切な配列要素(を表すコップ)に書き入れて下さい。\$search_words配列要素よりもデータの方が少なければ、要素(を表すコップ)は、×印で消して下さい。

```
$clean_search = str_replace(',', ' ', $user_search);
```

```
$search_words = explode(' ', $clean_search);
```

bull,matador cape

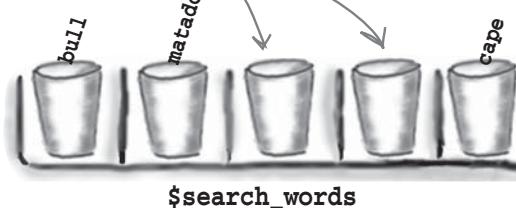
この配列には3つの要素しかありません。



bull matador cape

空白3つ！

この2つの配列要素は空です。2つの余分な空白が検索文字列内の matador と capeとの間にあります。

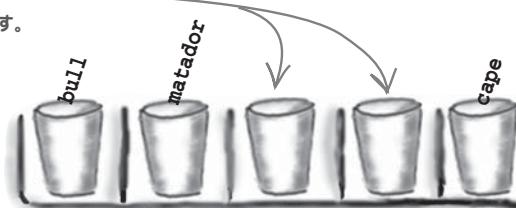
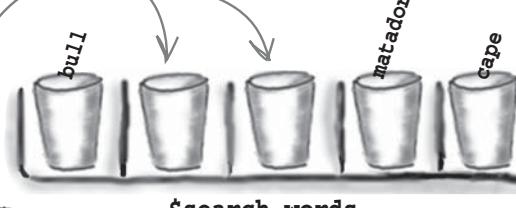


bull , matador cape

空白2つ！

またしても2つの空要素です。コンマが空白に置き換えられたためです。

bull,matador, cape





じゃあこれで検索文字列の前処理は終わって準備完了なんでしょう？

う～ん、残念。前処理で不要な文字は捨てたのですが、それがそのまま全部イケてる検索語だけを含む配列にはなっていないのです。

覚えていますか？当面の目標は、文字列を作り出すことなのですが、それは全く同一の分離子、つまり1つの空白で分割されていなければならぬのです。ところで反対側のページで最後の3つの例がどうなっているかをよく見てみましょう。search_words配列の要素のいくつかは空です。もし空の検索要素でWHERE節を作ってしまったとしたら、以下のような感じのものができ上がってしまいます。

```
SELECT * FROM riskyjobs
WHERE description LIKE '%bull%' OR
description LIKE '%matador%' OR
description LIKE '% %' OR
description LIKE '% %' OR
description LIKE '%cape%'
```

この空白文字は、業務概要に書いてあるすべての空白文字にマッチしてしまいます。大問題です。

でもこんなもの何ともマッチしないでしょ。良いじゃない？

違います！こいつらはすべてとマッチしてしまいます。

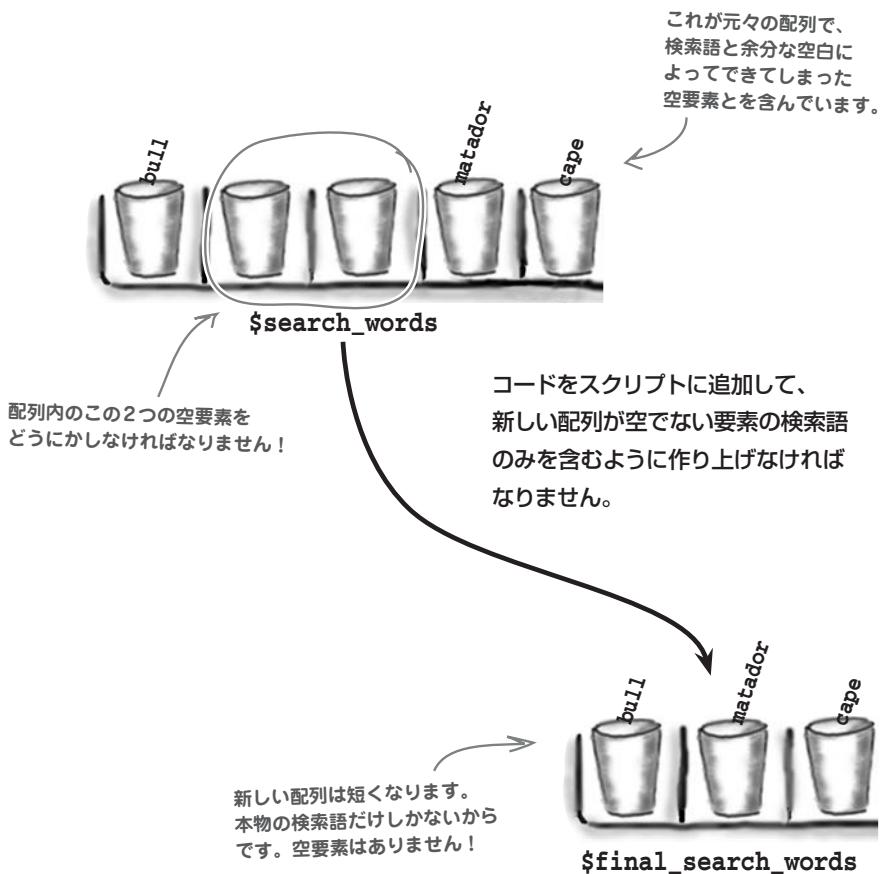
業務概要のどこかに空白さえあれば、この問い合わせ文は、必ずマッチしてしまい、それを結果として返します。つまりリスクジョブサイトのデータベースに入っているすべての仕事に、この問い合わせ文はマッチしてしまうのです。配列内のこれらの空要素を捨ててからSQL問い合わせ文を作らないと、検索文字列は便利なものに戻りません。



問い合わせ文にはちゃんとした検索語が必要です

めでたいことに、検索語をきれいにしてから、それを使って問い合わせ文を作るというのは、大して難しくありません。新しい配列を作つて、そこには本物の検索語だけを含んでいるようにすればよいだけです。つまりすべての空でない要素を最初の配列から第2の配列にコピーしてから、その配列を使ってSELECT問い合わせ文を構築するのです。

新しい配列を作るには、`foreach`ループを元々の配列の各要素について回して、空でない要素を見つけたらそれを使えばよいわけです。空でない要素を見つけた場合、単に新しい配列に加えてやります。この処理は以下のようになります。



空でない要素を新しい配列にコピーする

ではコードを見てみましょう。空でない要素を \$search_words 配列から、新しく作る \$final_search_words 配列にコピーします。

```
$search_query = "SELECT * FROM riskyjobs"; 検索語を取り出し配列に格納
// Extract the search keywords into an array
$search_clean = str_replace(',', ' ', $user_search);
$search_words = explode(' ', $clean_search);
$final_search_words = array();
if (count($search_words) > 0) {
    foreach ($search_words as $word) {
        if (!empty($word)) {
            $final_search_words [] = $word;
        }
    }
}
```

この部分は何も新しくありません。
str_replace()を使ってコンマを
空白で置き換えているだけです。

\$search_words 配列の要素について
のループです。要素が空でなければ、
それを \$final_search_words という
名前の配列に入れます。

少なくとも1つ以上の検索語が \$search_words に入っていることを確認してから、foreach ループで配列を回し、空でない要素を探します。空でない要素が見つかったら、[] 演算を使って、その要素を \$final_search_words 配列の後ろに加えます。以上が新しい配列の組み立て方です。

それからどうしますか？ そうです。単に SELECT 問い合わせ文を以前と同じように作ればいいのです。ただし使う配列は \$final_search_words であって \$search_words ではありません。

```
// Generate a WHERE clause using all of the search keywords すべての検索語を使って WHERE 節を生成
$where_list = array();
if (count($final_search_words) > 0) {
    foreach($final_search_words as $word) {
        $where_list[] = "description LIKE '%$word%'";
    }
}
$where_clause = implode(' OR ', $where_list);
```

これは以前のコードと同じです。
問い合わせ文の WHERE 節を作っています。
ただし新しい配列 \$final_search_words を
使っています。これには空でない
要素だけが入っています。

```
// Add the keyword WHERE clause to the search query 検索語の WHERE 節を検索用の問い合わせ文に追加
if (!empty($where_clause)) {
    $search_query .= " WHERE $where_clause";
}
```

このコードで検索用の問い合わせ文はもはや空要素を含んでいません。
「bull, matador, cape」を検索する場合の新しい問い合わせ文は次のようになります。

```
SELECT * FROM riskyjobs
WHERE description LIKE '%bull%' OR
description LIKE '%matador%' OR
description LIKE '%cape%'
```



脳力発揮

この検索によりユーザは
望みの結果が得られると
思いますか？



試運転

検索用スクリプトをアップデートし、ユーザの検索用文字列を前処理します。

search.phpスクリプトをアップデートし、explode()関数とimplode()関数を使って、ユーザの検索用文字列を前処理し、もっとちゃんとしたSELECT問い合わせ文を作ります。できあがったスクリプトをWebサーバにアップロードしたら、いくつか検索を試してみます。

リスクジョブ：危険な仕事検索

都道府県	更新日時
山口	2009-11-14 11:43:59

おーちゃんの検索
「tightrope, walker, circus」で、今度は関係のありそうな仕事が出来ました。

2009-11-14 21:13:35

2009-11-14 21:17:16

Risky Jobs

危険！夢の仕事は危険と隣り合わせです。
勇気を出して探してみませんか？

リスキージョブ：危険な仕事検索の結果

仕事の名称

網渡り (Tightrope Walker)

業務概要

新規オープンの大テントにて本格サーカスのプロフェッショナルを募集しています。1~3年間の経験を有し、網渡りのクロバットを演じながら大きな象と一緒に演じてください。象の排泄物を喜んで手伝って頂ける方お可です。特典として、医療保険、401K、自社株保有および割引購入、効率保険、各種商品の割引購入、短期間の休職保険、生命およびビジネス渡航保険、メガネ割引プラン、自動車および火災保険の割引、健康保険および扶養者保険、教育費補助、有給休暇、養育支援などがあります。給与はスキル、能力、経験、および開拓地を考慮の上、柔軟に対応します。プロモーションの機会はバフォーマンスに応じて得られまます。大テント最上部にある網渡り用のワイヤーに行くことを選るのは、ご自身の意志と倫理です。大テント最上部にある網渡り用のワイヤーに行くことを選るのは、ご自身の意志と倫理です。柔軟性、緊急時の対応、成功報酬型のクロバットのプロが成功への道です。チャイ！織機性、緊急時の対応、成功報酬型のクロバットのプロが成功への道です。チャイ！織機性、緊急時の対応、成功報酬型のクロバットのプロが成功への道です。チャイ！織機性、緊急時の対応、成功報酬型のクロバットのプロが成功への道です！

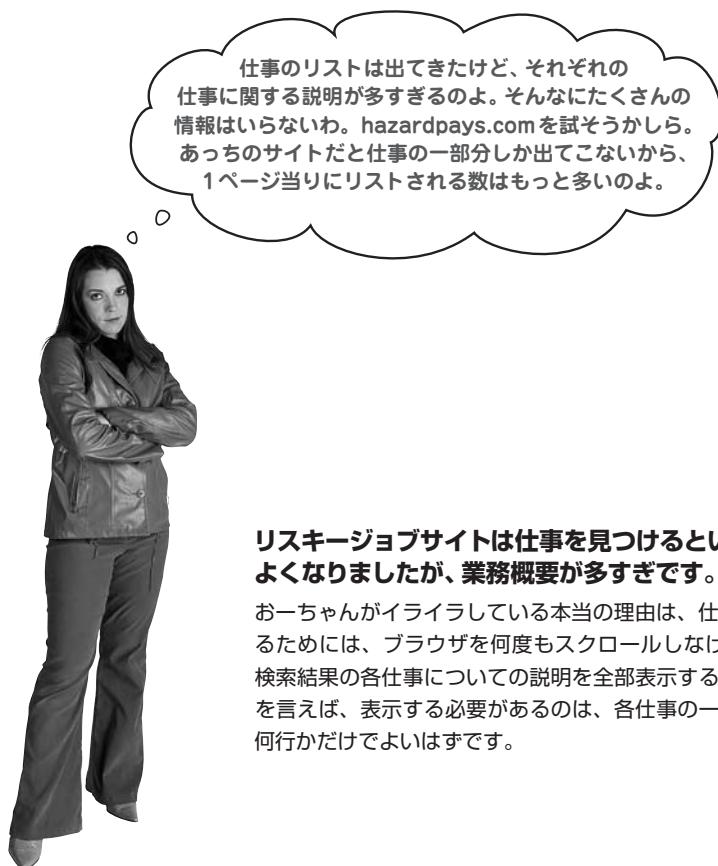
Fledgling big top looking for three-ring professionals with 1-3 years of experience to perform tightrope acrobatics with pudgy elephants. Willingness to sweep excrement a big plus. Excellent benefits including medical and dental plans, 401 (k), stock ownership and discount purchase plan, prescription coverage, merchandise discount, short and long term disability insurance, life and business travel insurance, vision discount plan, auto and home insurance discounts, medical care and dependent care reimbursement, educational assistance, paid vacation and holidays, and adoption assistance. Flexible starting salaries based on skills and abilities, experience and geographic market. Promotion opportunities based on performance. The only thing stopping you from the highest wire in the big tent is your desire and work ethic...and your balance! Other duties include planning & organizing wires, handling minor elephant administration, processing comment cards from children. Leading by example (don't fall!), showing initiative and a sense of urgency and being results-driven help acrobatic professionals become successful. If you want to be challenged and your talent needs mentoring and opportunity, Bingling Brothers can offer you a fast track to success!

今や失われつつある猫ジャグリングに従事していますか?240ヶ国以上で禁止されています中、ジ・高知・ルイス・ザ・サーカスだけが猫ジャグリングを洗練させ現代の観客を惹くよう進化させています。ウデを磨いて私たちのサーカスで猫ジャグリングの初日を飾りましょう。シンクロ猫ジャグリングをマスターすることのできる地球上で唯一の場所です。実際ジャグリングの方から面白みよりも、むしろ難しいのです。従業員のみなさんは全員平等にチヤンスが与えられます。我々のチーハーもすることをお待ちしています。しっかりとしたケージをご用意ください。猫の扱いが器用であることを証明するためのテストに必要です。猫ジャグリングマスタープログラムでは、豆乳クリーム以外は禁止されています。

Are you a practitioner of the lost art of cat juggling? Banned in forty countries, only the Jim Ruiz Circus has refined cat juggling for the sophisticated tastes of the modern audience. Play your trade with premiere cat jugglers at our circus, the only place on earth to master synchronized cat juggling. It's true, juggling them is even harder than herding them. We are an equal opportunity employer, and look forward to adding you to our team. Please be prepared to undergo a thorough battery of tests to prove your deft handling of felines. Only the cream of the crop will be accepted into our Master Cat Juggler program.

非常に高いところに何時間もぶら下がって考え事をすることが好きなら、この仕事はあなたにうってつけです。私たちの網渡り用ロープは厳密な43ポイントテストをパスしたもので、これは、人間が一定期間以上ぶら下がるという条件では最高水準を確保しています。ぶら下がるのはあなたです!安全用のネットは提供しますが、ヘルメットおよび手袋はご持参ください。福島の大天井にある私たちの工場では、各種雇用パッケージプランにて優遇いたします。例えば職場へのベット同伴は平日および週末共に認められています。応募には3通の推薦状が必要です。ぶら下がっていた最も長時間および過去に落ちた回数を証明してください。

福島



**リスキージョブサイトは仕事を見つけるという意味ではとても
よくなりましたが、業務概要が多すぎです。**

おーちゃんがイライラしている本当の理由は、仕事のリストをたくさん見るためには、ブラウザを何度もスクロールしなければならないからです。検索結果の各仕事についての説明を全部表示する必要はないのです。理想を言えば、表示する必要があるのは、各仕事の一部分だけ、恐らく最初の何行かだけでよいはずです。

どうすれば仕事の記述を刈り取ることができると思うか書いて下さい。
検索結果が巨大なものにならないようにする必要があります。

.....
.....
.....

時には文字列の一部しか必要ないこともあります

リスキージョブサイトのデータベースに入っている業務概要の長さは、短いものから非常に長いものまで色々です。検索結果をきれいにするには、すべての説明を短く切り落としてしまえばよいのです。このとき、ユーザが混乱しないように、後ろに省略記号(…をくつ付けておけば、ユーザは説明の一部だけを見ることができます。

PHP の mb_substr() 関数は、日本語を含む文字列から一部分だけを取り出すのに完璧です。「部分文字列」関数には、元々の文字列と 2 つの整数、および文字コード種別を渡します。第 1 の整数は、部分文字列の開始位置を指定する添字です。第 2 の整数は、部分文字列の長さを文字数で指定します。最後の文字コードセットには本書の場合 'UTF-8' を指定します。以下に構文を示します。

PHP の mb_substr()
関数を使うと文字列の
一部分を取り出すこと
ができます。

mb_substr(string, start, length, encoding)

これが元々の文字列で、
ここから部分文字列を
取り出します。

ここで部分文字列がどこか
ら始まるかを指定します…

…ここで何文字返す
かを指定します。

最後の文字コードを
指定します。本書の
場合'UTF-8'です。

mb_substr() 関数を使うときは、文字列を配列と考えるとよいでしょう。
各文字が配列の異なる要素という感じです。以下の文字列を見てみます。

```
$job_desc = 'Are you a practitioner of the lost art of cat juggling? ';
```

配列の要素と同様に、この文字列の各文字も添字を持ちます。0 から始まって文字列の終わりまでカウントアップしていきます。

Are you a practitioner of the lost art of cat juggling?

0 1 2 3 4 5 6 7 8 9...
... 50 51 52

これら各文字の添字を使って、mb_substr() 関数を呼び出すと、文字列の一部分を取ってくことができます。

添字 4 から
始めて 3 文字
取ってきます。

mb_substr(\$job_desc, 4, 3, 'UTF-8') → you

添字 49 から始めます。
第 2 の整数引数がない場合は、
文字列の最後まで取って
くることを意味します。

mb_substr(\$job_desc, 49, 'UTF-8') → ing?
mb_substr(\$job_desc, 0, 3, 'UTF-8') → Are
mb_substr(\$job_desc, 0, 9, 'UTF-8') → Are you a

部分文字列はどちらの端からでも取り出せる

`mb_substr()` 関数は文字列の先頭側からだけしか部分文字列を取り出せないわけではありません。文字列の後ろ側から始めて文字を抽出することもできます。抽出自体はここでも左から右になれます。負の添字を使うだけで、部分文字列の開始を識別することができます。

Are you a practitioner of the lost art of cat juggling?

↙ ↘ ↗ ↙
-53 -52 -51 -50 ...

... ↗ ↘ ↗
-3 -2 -1

以下に2つの例を示します。

添字-53から始めて、7文字取ってきます。
`mb_substr($job_desc, -53, 7, 'UTF-8')` → Are you

添字-9から始めて、残りの文字列を取ってきます。
`mb_substr($job_desc, -9, 'UTF-8')` → juggling?

自分で考えてみよう

以下に示すPHPコードは、リスキージョブサイトの検索結果を表示するHTMLテーブルを生成します。欠けているコードを埋めて下さい。この部分で業務概要を100文字のテキストに制限し、さらに投稿日時を年月日のみを表示するようデータを刈り取ります。

```
echo '<table border="0" cellpadding="2">';
echo '<td>仕事の名称</td><td>業務概要</td><td>都道府県</td><td>更新日時</td>';
while ($row = mysqli_fetch_array($result)) {
    echo '<tr class="results">';
    echo '<td valign="top" width="20%">' . $row['title'] . '</td>';
    echo '<td valign="top" width="50%">' . ..... . '....</td>';
    echo '<td valign="top" width="10%">' . $row['state'] . '</td>';
    echo '<td valign="top" width="20%">' . ..... . '</td>';
    echo '</tr>';
}
echo '</table>';
```

自分で考えてみよう の答え

以下に示すPHPコードは、リスキージョブサイトの検索結果を表示するHTMLテーブルを生成します。欠けているコードを埋めて下さい。この部分で業務概要を100文字のテキストに制限し、さらに投稿日時を年月日のみを表示するようデータを刈り取ります。

```
echo '<table border="0" cellpadding="2">';
echo '<td> 仕事の名称 </td><td> 業務概要 </td><td> 都道府県 </td><td> 更新日時 </td> ';
while ($row = mysqli_fetch_array($result)) {
    echo '<tr class="results">';
    echo '<td valign="top" width="20%">' . $row['title'] . '</td>';
    echo '<td valign="top" width="50%">' . mb_substr($row['description'], 0, 100, 'UTF-8') . '...</td>';
    echo '<td valign="top" width="10%">' . $row['state'] . '</td>';
    echo '<td valign="top" width="20%">' . mb_substr($row['date_posted'], 0, 10, 'UTF-8') . '</td>';
    echo '</tr>';
}
echo '</table>';

最後に省略記号を付けて、これは説明の一部であることを示します。
```

すべての投稿日時は、MM-DD-YYYY
という形式で始まっていますので、
ちょうど10文字です。



マニア向け情報

PHPのmb_substr()関数を使わずに、業務概要のデータをSQL問い合わせ文自体で制限してしまうこともできます。MySQLに非常によく似た関数SUBSTRING()[†]というものがあるって、mb_substr()と同様な引数を取ります。1つだけ違ったのがあって、最初の添字が0ではなく1なのです。つまり業務概要から最初の100文字を取ってくると、こんな感じになります。

```
SELECT SUBSTRING(description, 1, 100)
FROM riskyjobs;
```

PHP関数で処理することの利点は、業務概要に関して部分的な説明も、完全な説明も両方持っているということです。MySQLを使ってしまうと、仕事の部分的な説明しかありません。完全な説明が必要な場合は、改めて問い合わせ文を発行しなければならないのです。

[†] 訳注：MySQLのSUBSTRING()関数は日本語文字にも対応していますので、対応するMB_といった関数ではなく、そのまま用いることができます。

素朴な疑問 に答えます

Q : mb_substr()は数値にも使えますか？

A : 使えません。この関数は厳格に文字列のみに作用します。しかし、数値をCHARやVARCHARやTEXTでぶち込んでいるのであれば、そしてそれをSQLで引っ張ってくるのであれば、PHPでは文字列として扱われます。それは数字ではありませんから、mb_substr()関数を使うことはできます。

Q : 長さの値は、文字列よりも長い場合は何が起こるのでしょうか？長さの値に合わせるために後ろに空白でも埋めてくるのですか？

A : 残りの文字列全部を返します。しかし後ろに長さを合わせるための空白を埋めたりはしません。例えば以下のコードでは文字列"dog"が返ってきます。

```
mb_substr('dog', 0, 10)
```



試運転

検索用スクリプトを微調整して、仕事の記述として表示されるテキストと投稿日時を制限します。

search.phpスクリプトを修正します。PHPのmb_substr()関数を使って業務概要と投稿日時を、検索結果からそぎ落として下さい。できあがったらスクリプトをWebサーバにアップロードし、いくつか検索をテストしてみます。

おーちゃんはバカでかい業務概要の
せいでスクロールする必要もなく、
検索結果をサクサク見ることが
できて大喜びです。



リスクジョブ：危険な仕事検索

仕事の名称	業務概要	都道府県	更新日時
綱渡り (Tightrope Walker)	新設オープンの大テントにて本格サーカスのプロフェッショナルを募集しています。1~3年の経験を有し、綱渡りのアクロバットを演じながら大きな象と共に演してください。象の排泄物を喜んで手伝って頂ければなお可で…	山口	2009-11-14
猫ジャグラー (Master Cat Juggler)	今や失われつつある猫ジャグリングに従事していますか？40ヶ国以上で禁止・高知されていく中、ジムルイズサーカスだけが猫ジャグリングを洗練させ現代の観客ウケするよう進化させていきます。ウデを磨いて私どものサーカス…		2009-11-14
綱渡り用ロープのテスト (Tightrope Tester)	非常に高いところに何時間もぶら下がって考え事をすることが好きなら、こ福島の仕事はあなたにうってつけです。私たちの綱渡り用ロープは厳格な43ボイントテストのパス!たものです。これは、人間が一定期間以上ぶら下…	福島	2009-11-14

結果が投稿日時や都道府県名で
ソートされていると嬉しいんだけどなあ。
本当言うと新潟でマタドールの
仕事を見つけたいんだ。

投稿日時も少し読みやすくなっています。
日時ではなく日付のみを表示しています。



脳力発揮

ページレイアウトを変更して、問い合わせ結果を投稿日付や都道府県名や仕事の名称でソートするにはどうすればよいと思いますか？

複数の問い合わせ文で結果をソートできます

サイトの訪問者に検索結果をソートすることができるようになります。その人たちが結果をどのような順番で並べたいのかを識別する方法が必要となります。恐らくフォーム…かまたはボタンで? 実際にはそれよりもっと単純な方法があります。HTMLを使って検索結果の見出しをリンクにして、カラムを変更すればよいのです。ユーザはリンクをクリックすることで、どのカラムで結果をソートしたいかを示すことができます。

リスキージョブ：危険な仕事検索

Risky Jobs

危険！夢の仕事は危険と隠れています。
勇気を出して探してみませんか？

リスキージョブ：危険な仕事検索の結果

ユーザは結果を見出しに基づいてソートすること、特定の仕事に注目することができます。見出しをリンクに変えて、ユーザがクリックすることで仕事のリストをソートできるようにします。

仕事の名称	業務概要	都道府県	更新日時
プロボクサー (Prize Fighter)	来るべきスマーフライ級ボクサーを募集しています。チャンピオンの運 勝負譲を伸ばしてください。美しいファイトワーク、ダメダメなジャッジ、ガラ スのアゴの持ち主を求めています。未経験者募集です。フルタイムの正社 員...	東京	2009-11-14
闘牛士 (Toreador)	可愛らしいウシが、あなたの優しいマント使いの卓越したスキルを待って います。闘牛士道正基本検食にバスしている必要があります。 Lovely bovines waiting for you...	山形	2009-11-14
電動暴れ牛修理工士 (Electric Bull Reparier)	ハンクスホンキートンクでは、経験豊富な電動暴れ牛修理工士を募集してい ます。特典として(修理後の) フリーライドおよび手羽先磨きの半額な どがあります。 Hank's Honky Tonk nee...	神奈川	2009-07-27

この部分で問い合わせ文の結果を業務概要での
アルファベット昇順にソートします。

このリンクを使って同じ検索用スクリプトでデータを再ロードします。ただし、問い合わせ文で結果をクリックされたリンクに従ってソートします。ORDER BYの使って問い合わせ文の結果をソートして出力するやり方はもう知っています。それぞれのカラム毎にORDER BYを使って、異なるSQL問い合わせ文を作れば、ユーザに対して、検索結果をソートして表示することができます。ソートのキーが、仕事の名称、説明、都道府県名の場合アルファベット順[†]になり、投稿日時の場合は時系列順となります。

以下に結果を業務概要に関するアルファベット順でソートするSQL問い合わせ文を示します。

```
SELECT * FROM riskyjobs
WHERE description LIKE '%Bull%' OR description LIKE '%Fighter%' OR
description LIKE '%Matador%'
ORDER BY description
```

この部分で問い合わせ文の結果を業務概要での
アルファベット昇順にソートします。

[†] 訳注：日本語文字列をソートする場合、実際にはUTF-8文字コードの値の大小に基づくソートとなってしまいますので、直感的にはあまり意味をなしません。現実的な解決策は、ソートしたいキーについて、読みをひらがな（またはカタカナ）を持っておくことです。このキーでソートすれば「あいうえお順」となります。ただし本書日本語版ではプログラムをこのように拡張していません。



自分で考えてみよう

リスキージョブサイトの検索結果を仕事の名称、都道府県名、および投稿日時のそれについてソートする3つの異なる問い合わせ文を書いて下さい。ユーザーは検索文字列として、「window, washer, skyscraper」[†]と打ち込んだものとして下さい。

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

以上の問い合わせ文を次のように並べ替えるには、どのように書き換えればよいでしょう？仕事の名称および都道府県名を逆順にしたい場合。最新の（最も最近投稿された）仕事を最初に出す場合。

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

[†] 訳注：それぞれ窓、清掃人、超高層ビルという意味です。

自分で考えてみよう の答え

リスキージョブサイトの検索結果を仕事の名称、都道府県名、および投稿日時のそれについてソートする3つの異なる問い合わせ文を書いて下さい。ユーザは検索文字列として、「window, washer, skyscraper」と打ち込んだものとして下さい。

```
SELECT * FROM riskyjobs
WHERE description LIKE '%window%' OR description LIKE '%washer%' OR
description LIKE '%skyscraper%'
ORDER BY job_title
```

ORDER BY のデフォルト
は昇順(ASC)です。これは
ORDER BY job_title ASC
と同じ意味です。

```
SELECT * FROM riskyjobs
WHERE description LIKE '%window%' OR description LIKE '%washer%' OR
description LIKE '%skyscraper%'
ORDER BY state
```

```
SELECT * FROM riskyjobs
WHERE description LIKE '%window%' OR description LIKE '%washer%' OR
description LIKE '%skyscraper%'
ORDER BY date_posted
```

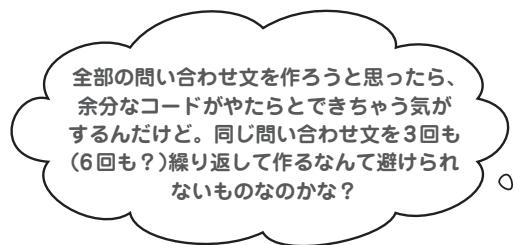
以上の問い合わせ文を次のように並べ替えるには、どのように書き換えればよいでしょう？仕事の名称および都道府県名を逆順にしたい場合。最新の（最も最近投稿された）仕事を最初に出す場合。

```
SELECT * FROM riskyjobs
WHERE description LIKE '%window%' OR description LIKE '%washer%' OR
description LIKE '%skyscraper%'
ORDER BY job_title DESC
```

すでにあるカラムについて順番に並んでいて、
ユーザがもう一度同じ箇所をクリックしたときには逆順に並べたい
というような場合、
このような問い合わせ文が必要になるかも
しません。

```
SELECT * FROM riskyjobs
WHERE description LIKE '%window%' OR description LIKE '%washer%' OR
description LIKE '%skyscraper%'
ORDER BY state DESC
```

```
SELECT * FROM riskyjobs
WHERE description LIKE '%window%' OR description LIKE '%washer%' OR
description LIKE '%skyscraper%'
ORDER BY date_posted DESC
```



避けられません。ユーザが異なるリンクをクリックしたときに異なる問い合わせ文を走らせる必要があるというのは事実です。ただしクリックされたリンクに基づいて1つの問い合わせ文を作り出すことができます。

最初に結果を表示するときには、どのリンクもクリックされていません。従ってソートについて何も心配する必要はありません。フォームに「提出」されたキーワードを取ってきて、問い合わせ文を ORDER BY なしで作ればよいだけです。結果はクリック可能な見出しを付けて表示します。それぞれの見出しリンクは、スクリプト自身にリンクしていますが、ソートの順序が異なるのです。そこで各リンクは URL に元々のキーワードと sort という名前のパラメタを取り、これを使って結果をどの順序でソートすべきかを指示します。このようなことをやるために本当に役に立つのは、専用のカスタム関数を作ることです。この関数は仕事のデータをどのようにソートすべきかという情報を受け取って、WHERE と ORDER BY を持った文字列を返してくれます。新しいカスタム関数は、sort パラメタの中を見て、検索結果をどのようにソートすればよいかを判断します。この関数は、次のようなステップを踏む必要があります。

- ❶ 検索キーワードを前処理し、これらを配列に突っ込みます。
- ❷ オプションで sort パラメタを取ります。このパラメタは関数にどのカラムでソートするのかを伝えます。
- ❸ 空の検索キーワードは全部捨てます。
- ❹ すべての検索キーワードを含む WHERE 節を作ります。
- ❺ sort パラメタが値を持つかどうかをチェックします。もしあれば、ORDER BY 節を組み合わせます。
- ❻ 新しく作り上げた問い合わせ文を返します。

これは大変な作業に思えるかもしれません、実はほとんどのコードはもう書き上げているのです。単に関数に置き換えれば良いだけです。でもその前に、カスタム関数をどうやって組み立てるのかを見ておきましょう…

関数でコードを再利用する

関数というのはコードのブロックのことと、他の部分のコードと分離されていて、スクリプトで必要な場合はいつでも実行することができます。これまでのところ、PHP であらかじめ用意されている組み込み関数のみを使ってきました。`explode()`, `mb_substr()`, `mysqli_query()`などはすべて、PHP であらかじめ定義された関数で、どんなスクリプトでも使うことができます。

ところで自分専用のカスタム関数というものを作つて、言語で提供されていないような機能を自分で提供することもできます。カスタム関数を作ると、自分自身のコードを何度も繰り返し使うことができます。しかもスクリプトを繰り返し書く必要はありません。繰り返す代わりに、コードを走らせたいときには、単に関数をその名前で呼び出せばよいのです。

以下はカスタム関数の例です。`replace_commas()` という名前で、文字列内のコンマを空白に置き換えます。

カスタム関数を作るには、
予約語 `function` で
始めます。

ここには、関数に付けていきたい名前を
適切に決めて書きます。できる
限り分かりやすく明確なものに
します。

関数名の後に2つのカッコが続きます。
関数に値を引数としていくつでも渡すことが
できます。各々はコンマで区切られます。
この例の場合、値を1つだけ渡しています。

```
function replace_commas($str) {
    $new_str = str_replace(',', ' ', $str);
    return $new_str;
}
```

関数は呼び出し元のコードに値を
返すことができます。この場合変更
された文字列を返します。

波カッコは、関数のコード
の開始(と終了)を表します。
ループや `if` 文と同様です。

カスタム関数を使う準備ができたら、後は処理して欲しい値をカッコに入れて単に名前で呼び出すだけです。関数が値を返すように作られている場合は、返ってきた値を新しい変数に代入することができます。こんな感じです。

文字列 'tightrope, walker, circus'
を渡します。

```
$clean_search = replace_commas('tightrope, walker, circus');
```

関数はコンマを空白で
置き換えた新しい文字
列を返してきます。

カスタム関数で問い合わせ文を作る

リスキーなサイトの検索用問い合わせ文を生成するカスタム関数を作るために必要なコードの大部分はすでに書いてあります。残っているのは、コードをPHP関数の骨組みに落とし込むことだけです。以下がbuild_query() カスタム関数です。

```
function build_query($user_search) {
    $search_query = "SELECT * FROM riskyjobs";
    // Extract the search keywords into an array
    $clean_search = str_replace(',', ' ', $user_search);
    $search_words = explode(' ', $clean_search);
    $final_search_words = array();
    if (count($search_words) > 0) {
        foreach ($search_words as $word) {
            if (!empty($word)) {
                $final_search_words[] = $word;
            }
        }
    }

    // Generate a WHERE clause using all of the search keywords
    $where_list = array();      // すべての検索語を使って WHERE 節を生成
    if (count($final_search_words) > 0) {
        foreach($final_search_words as $word) {
            $where_list[] = "description LIKE '%$word%'";
        }
    }
    $where_clause = implode(' OR ', $where_list);
    // Add the keyword WHERE clause to the search query
    if (!empty($where_clause)) {
        $search_query .= " WHERE $where_clause";
    }
    return $search_query;
}
```

関数には\$user_search配列を渡します。これは検索用フォームに打ち込まれたデータから作ったものです。

関数の内側は何も新しくありません！

そうです。ここは新しいところです。ここで新しい問い合わせ文を返し、この関数を呼び出した側のコードがこれを使います。

build_query() 関数は、SQL問い合わせ文を完全な形で返します。問い合わせ文は\$user_search引数を通して渡された検索文字列に基づいて作ります。関数を使うには、ユーザが打ち込んだ検索データを単に渡せばよいだけです。その後、新しい文字列に入っている結果を、\$search_queryと名付けた変数に突っ込めば良いだけです。

```
$search_query = build_query($user_search);
```

ここで関数結果の値を捕まえます。
この場合新しい検索用問い合わせ文です。

ここがユーザが「提出」した検索フォームの値です。



カスタム関数を暴露する

今週のインタビュー：

カスタム関数：カスタム関数はどこまでカスタムか？

Head First：ちょっといいですか？ 1つ疑問に思うんです
が…冗長なコードと言うのはそんなに悪いものなのでしょう
か？つまり作るのは簡単です。単にコピペすればおしまいで
すよね。

カスタム関数：え、いきなり冗長なコードから始めるんですか？ だってそのせいで平坦で醜くて読みにくくなるんですよ。十分悪いでしょ。でも、もっともっと重大な理由があつて、冗長なコードはダメなんです。

Head First：と言いますと？

カスタム関数：まず、コードに変更がある場合のことを考えてみて下さい。こんなことはよくあることです。

Head First：するとどうなるんでしょう？ 何かが変わると
いうのはいつものことです。単にそれに対応して直すだけです
よね？

カスタム関数：でも何か変更が冗長なコードで起こったら
どうでしょう？ それが5箇所も10箇所もアプリケーションに
散らばっているとしたら？

Head First：何が問題なのかよく分からないのですが…それを見つけて全部変えれば、終わりですよね。

カスタム関数：そうですか。わかりました。では一箇所だけ
でも変更し忘れたらどうなるでしょう？ 変更するのは人間です。
プログラマです。変更し損ねたら、それを見つけるために多くの時間を浪費してしまうかもしれませんよ。

Head First：その通りですね。確かにそういうことは起るかもしれませんね。では、カスタム関数はそんな状況を助けることができるのでしょうか？

カスタム関数：そう、それこそが美学であってカスタム関数の出番です。コードが関数内に書いてあれば、その部分を1箇所だけ変更すれば済みます。1箇所だけです。チョチョイのチョイで終わりです。

Head First：認めましょう。とても説得力があります。でもまだわからないことがあります。どうしてカスタム関数を使うために外に出て行かなければならないのでしょうか？つまり、カスタム関数というのは非常に限定的ですよね？ 文字列を扱うことしかできないのでしょうか？

カスタム関数：ええ！ ちょっと待って下さいよ！ カスタム関数は送ってもらえばどんなデータでも扱えますよ。関数の内部コードが、そのデータを正しく扱って下さいれば、どんなデータでもお望みどおり使えます。ほら、さっきだって配列を扱っていたでしょう。ものすごく洗練されていると言って良いんじゃないですか？

Head First：でも文字列を返しましたよね。

カスタム関数：カスタム関数はお望みのものを何だって返せます。カスタム関数が提供できるものを作って、正しく使えばよいだけです。

Head First：もう1つあります。常に要求しますよね。とにかくデータを渡さなければならない。

カスタム関数：どっからそんなバカげた話が出てくるのでしょうか？ カスタム関数を変数なしで呼ぶことだってできますよ。お望みなら、そしてそのように作り上げてくれればですが。データを送りたくないければ、関数を作るとき関数名の次にあるカッコの中に変数を書かなければ良いのです。でもカスタム関数にデータを渡したくないという理由はあまり見当たらないと思います。それにreturn文でデータを戻したくないということも、あまりないでしょう。

Head First：時間が来てしまったようです。今日はどうもありがとうございました。

カスタム関数：とんでもありません。供すれば生きるのか？ または生きれば供するのか？ それともレバーを供するのか？ ま、そんな感じです。



試運転

検索用スクリプトを修正して**build_query()**関数を使うようにします。

新しく**build_query()**関数をsearch.phpスクリプトに作り、元々のコードを新しい関数の呼び出しに置き換えます。できあがったらWebサーバにスクリプトをアップロードし、Webブラウザで検索を試み、動作が正しいかどうか確認して下さい。

新しいカスタム関数**build_query()**はイケてるけど、
でもまだ検索結果をソートしていないや。パラメタを
追加してその処理ができるようになるのかな？



その通りです。**build_query()**関数に1つではなく
2つのパラメタを渡せばよいのです。

すでに関数には引数として\$user_searatchを与えています。この引数にはユーザーの検索語が入っています。新たに必要なもう1つの引数は\$sortです。これによってデータをどのようにソートするのかを指示します。新しい\$sort引数は、問い合わせ文によって返ってきたデータの順序を6通りにコントロールする必要があります。これは467ページで見たとおり、riskyjobsテーブルのjob_title, state, date_postedの各カラムについてのソートで、データの昇順と降順の両方です。

\$sortにORDER BYで始まる文字列を実際に突っ込んでも良いのですが、別のやり方もあります。数字の1から6までを使って各ソートの種別を表現するのです。以下のような感じです。

```
$sort == 1 ➔ ORDER BY job_title
$sort == 2 ➔ ORDER BY job_title DESC
$sort == 3 ➔ ORDER BY state
$sort == 4 ➔ ORDER BY state DESC
$sort == 5 ➔ ORDER BY date_posted
$sort == 6 ➔ ORDER BY date_posted DESC
```

仕事の記述でアルファベット順にソートすることには、あまり意味はないと思いますので、触れていません。

この数字は適当に選んだだけ
なので大して意味はありません。
特別な規則もありません。
一貫して使えば良いだけです。

でも、整数値というのは、コードを読む際に暗号的にならないませんか？ちゃんとコメントを書かなければわけが分からなくなってしまいます。しかし、整数值を使うことに対する重要な理由がここにはあるのです。もしORDER BYの文字列を使ったとすると、このデータはスクリプトのURLに見出しリンクの一部として見えてしまいます。このためテーブルのカラム名がうかつにもバレてしまうのです。カラム名などはセキュリティ上の理由から公開すべきではありません。



OKよ。新しい\$sort引数がどういう働きなのかはわかったわ。でも\$sortにどの値を渡して関数を呼ぶのかはどうすれば分かるのかしら？ユーザがそれを知らせなければならぬなんてことはないわよね？

ユーザが検索結果をどのようにソートするかを指定しなければならない、というのは事実です。ただし、検索語自体を指定するだけです。

めでたいことに、この機能をどうやって実現したいのかは、分かっているのです。カラムの見出しをクリックするとハイパーリンクで結果のページに変わります。ユーザが見出しの例えば「都道府県」をクリックしたら、都道府県名でソートすることを表す数字をbuild_query()関数に渡すわけです。

ただし、まだ問題は残っています。リンクからソートの順序(昇順、降順の別)もスクリプトに渡す必要があります。これについては、見出し用のカスタムリンクを作るときに、\$sortパラメタをURLにくっ付けておくことで解決できます。

検索結果はHTMLテーブルの一部として作られますから、<td>タグがここにあるのです。

```
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search . '&sort=3">State</a></td>;'
```

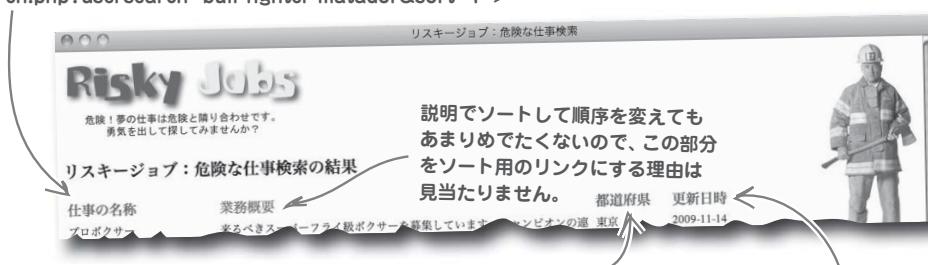
build_query()関数は、ユーザが入力した探索キーワードを使って結果を表示します。そこでこれをURLに渡します。

ユーザがカラムの見出しをクリックしたら、結果をソートしてページを再度読み込みます。つまり自己参照フォームにするのです。

一緒にsortデータを渡して検索をどのようにソートするかを指示します。この場合「都道府県」のリンクですから、sortは3です。

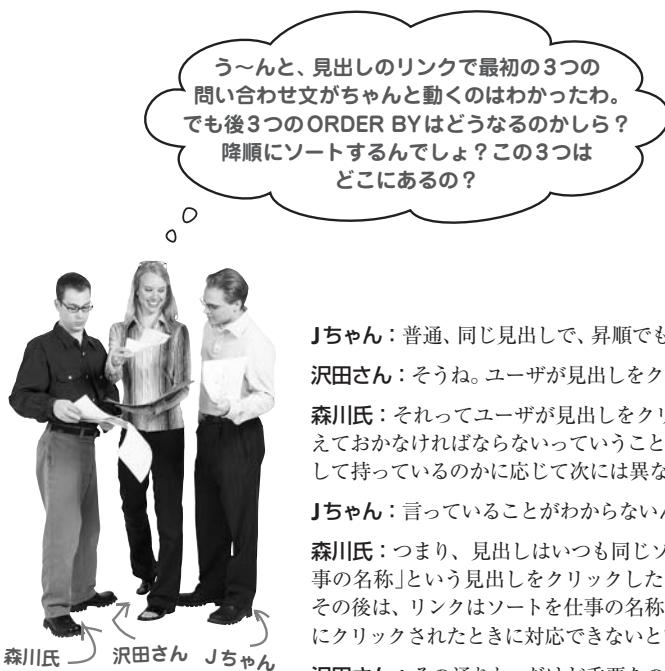
結果のページができたら、(業務概要を除く)各見出しリンクは、それぞれ独自のURLを持たせます。これには\$sortの値も含んでいて、結果がどのようにソートされるべきかを示しています。

```
<a href="search.php?usersearch=bull fighter matador&sort=1">
```



```
<a href="search.php?usersearch=bull fighter matador&sort=3">
```

```
<a href="search.php?usersearch=bull fighter matador&sort=5">
```



Jちゃん：普通、同じ見出しで、昇順でも降順でもどっちでもソートできるよね。

沢田さん：そうね。ユーザが見出しをクリックする度に、順序が変わるわ。

森川氏：それってユーザが見出しをクリックする度に何らかの方法でその状態を覚えておかなければならぬっていうことじゃない? だって、見出しが今何をリンクとして持っているのかに応じて次には異なるリンクを持っていなきゃいけないから。

Jちゃん：言っていることがわからないんだけど。

森川氏：つまり、見出しあはいつも同じソートをするわけじゃないんだ。例えば、「仕事の名称」という見出しをクリックしたときに、仕事の名称で昇順にソートされる。その後は、リンクはソートを仕事の名称での降順で行うように変わっていて、その次にクリックされたときに対応できないといけない。

沢田さん：その通りね。だけど重要なのは、どのタイプのソートをするかは数字で表されていて、それをURLに書き込んで、スクリプトにどのソートをするかを知らせるっていうことよ。今そのリンクを作ろうとしているわけだから、どのソート番号が必要なのかをちゃんとコントロールできなきゃいけないのよ。

Jちゃん：わかってる。つまり今の目標は、何らかの方法で、現在のソート状態に基づいて正しいリンクを作ることのできるようなコードを組み上げることだろう？

森川氏：あ、わかった! それってif文がいくつかあれば解決できるんじゃない? つまり、この種の判断をするにはif文が適しているだろ?

Jちゃん：そう。それでも良いと思うけど、同じデータつまりソートのタイプに対して、判断がいくつかあるというところが気になるんだよ。何かもっと良い方法で判断ができたら良いんだけどな。if-else文をたくさんネストさせるんじゃないくて、だよ。

沢田さん：鋭いわね。アタシ聞いたことがあるんだけど、これって新しい文を試してみるには完璧な状況だわ。switch文で複数の判断ができるのよ。2つ以上を1つの値だけに基づいて。

森川氏：何かすごそうだね。それやってみようよ。

Jちゃん：賛成。複雑なif-else文を避けられるのなら何でも賛成。頭痛くなってくるからね！

沢田さん：その通りね。switch文は、単にチケットみたいなものかもしれないけど…

SWITCHはIFよりずっと多くの判断をします

switch 文は、値をチェックし、その値に応じていくつかの異なるブロックのコードのうちの1つを実行する効率的な方法を提供してくれます。これは if-else 文みたいなものです。特に判断のオプションがあまりないような状況で有効です。

ネストした if-else 文を書いて、取りうる値をチェックする代わりに、switch 文に case ラベルを付けて対応する値を書きます。case ラベルの最後には break 文において、PHP に switch 文から抜け出すことを指示します。これで他の case は実行しなくなります。これにより PHP は1つの case だけしか実行しないことが保証されます。

switch を使った例を見てみましょう。

```
switch ($benefit_code) {
    case 1:
        $benefits = '主な病気、10日間の病欠扱い';
        break; // このコードは $benefit_code の値が 1 のときに限り実行されます。
    case 2:
        $benefits = '死亡または四肢切断、1ヶ月の有給休暇扱い';
        break;
    case 3:
        $benefits = '2つ以上の値に対して同じことをやりたい場合、break 文を書かずにしてすべての case を並べます。';
    case 4:
        $benefits = '健康状態!';
        break;
    default:
        $benefits = 'なし';
}

echo '4種類の健康保険パッケージを用意しています。';
echo 'あなたの選んだプラン' . $benefits;
```

これがswitch文がチェックする値です。この値によってswitch全体をコントロールします。

break文でPHPにswitch文から抜け出すことを指示します。

\$benefit_codeに入っている1~4までの値以外についてはdefaultのコードが実行されます。

ウソです。パッケージは3つかりません。3と4とは同じだからです。case 3にbreakを付けなかったおかげです。

SWITCH文
は、一連のCASE
ラベルを含んでいて、
変数の値に応じて
異なるコードブロック
を実行します。



リスキージョブサイトに新しく generate_sort_links() という関数を追加して、結果の見出しをクリックすることでユーザが検索結果をソートできるようにします。残念なことに、いくつかの重要なコードが抜けています。関数のコードを仕上げて下さい。大事なことを覚えていますか？各検索タイプを表す数値は次の通りです。

- 1 = 仕事の名称での昇順、2 = 仕事の名称での降順、3 = 都道府県名での昇順、
4 = 都道府県名での降順、5 = 投稿日時での昇順、6 = 投稿日時での降順。

```
.....generate_sort_links($user_search, $sort) {
$sort_links = '';
.....($sort) {

case 1:
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">仕事の名称</a></td><td>Description</td>';
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">都道府県</a></td>';
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">更新日時</a></td>';
.....
case 3:
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">仕事の名称</a></td><td>Description</td>';
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">都道府県</a></td>';
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">更新日時</a></td>';
.....
case 5:
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">仕事の名称</a></td><td>Description</td>';
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">都道府県</a></td>';
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">更新日時</a></td>';
.....
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">仕事の名称</a></td><td>Description</td>';
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">都道府県</a></td>';
$sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
'&sort= .....">更新日時</a></td>';
}
return ....;
}
```



これはデフォルトの見出します。ユーザが
ソート方法を選ばなかったときに使います。



リスクジョブサイトに新しく generate_sort_links() という関数を追加して、結果の見出しをクリックすることでユーザが検索結果をソートできるようにします。残念なことに、いくつかの重要なコードが抜けています。関数のコードを仕上げて下さい。大事なことを覚えていますか？各検索タイプを表す数値は次の通りです。

1 = 仕事の名称での昇順、2 = 仕事の名称での降順、3 = 都道府県名での昇順、

4 = 都道府県名での降順、5 = 投稿日時での昇順、6 = 投稿日時での降順。

```
function .....generate_sort_links($user_search, $sort) {
    $sort_links = '';
    switch.....($sort) {
        case 1:           $sortの値が1の場合、仕事の名称ですでにソートされていることを意味していますから、同じく仕事の名称で今度は降順に再ソートする必要があります。
            $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
                '&sort= . 2.....">仕事の名称</a></td><td>Description</td>';
            $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
                '&sort= . 3.....">都道府県</a></td>';
            $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
                '&sort= . 5.....">更新日時</a></td>';
        break: .....
```

case 3:

```
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
        '&sort= . 1.....">仕事の名称</a></td><td>Description</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
        '&sort= . 4.....">都道府県</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
        '&sort= . 3.....">更新日時</a></td>';
    break: .....
```

case 5:

```
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
        '&sort= . 1.....">仕事の名称</a></td><td>Description</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
        '&sort= . 3.....">都道府県</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
        '&sort= . 6.....">更新日時</a></td>';
    break: .....
```

default:

```
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
        '&sort= . 1.....">仕事の名称</a></td><td>Description</td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
        '&sort= . 3.....">都道府県</a></td>';
    $sort_links .= '<td><a href = "' . $_SERVER['PHP_SELF'] . '?usersearch=' . $user_search .
        '&sort= . 5.....">更新日時</a></td>';
}
return $sort_links...;
```

}

*\$sortがセットされていない場合、または2, 4, 6に
セットされている場合、元々のリンクを表示して
データを昇順にソートするようにします。*

† 訳注：このコード例では default ラベルの最後に break 文が付いていませんが、コーディングの作法としては常に付けることをお勧めします。ただし switch 文最後の break 文はあってもなくても動作は同じです。

build_query()にソート機能を加える

これでリスクジョブサイトの検索を処理する2つの関数が揃いました。build_query()は、ユーザが打ち込んだ検索語に基づいてSQL問い合わせ文を作り上げます。generate_sort_links()は、検索結果の見出しに付けるハイパーリンクを生成し、ユーザが結果をソートできるようにします。でもまだbuild_query()は全然完成していません。この関数が生成した問い合わせ文はまだ、ソートをしないのです。この関数でORDER BY節を問い合わせ文にくっ付ける必要があります。ただし新しく追加した\$sort引数の値に応じて正しいORDER BY節を付けなければなりません。

```

function build_query($user_search, $sort) {
    $search_query = "SELECT * FROM riskyjobs";
    ...

    // Add the keyword WHERE clause to the search query 検索語の WHERE 節を検索用の問い合わせ文に追加
    if (!empty($where_clause)) {
        $search_query .= " WHERE $where_clause";
    }

    // Sort the search query using the sort setting 検索用の問い合わせ文をソート設定に基づいてソート
    switch ($sort) {
        // Ascending by job title 仕事の名称での昇順
        case 1:
            $search_query .= " ORDER BY title";
            break;
        // Descending by job title 仕事の名称での降順
        case 2:
            $search_query .= " ORDER BY title DESC";
            break;
        // Ascending by state 都道府県名での昇順
        case 3:
            $search_query .= " ORDER BY state";
            break;
        // Descending by state 都道府県名での降順
        case 4:
            $search_query .= " ORDER BY state DESC";
            break;
        // Ascending by date posted (oldest first) 更新日時での昇順
        case 5:
            $search_query .= " ORDER BY date_posted";
            break;
        // Descending by date posted (newest first) 更新日時での降順
        case 6:
            $search_query .= " ORDER BY date_posted DESC";
            break;
        default:
            // No sort setting provided, so don't sort the query
    }
}

return $search_query;
}

```

\$user_searchに加えて、新しく
\$sort引数を関数に渡します。

この部分がbuild_query()に
追加したコードです。このswitch文
で\$sortの値をチェックし、それに
応じてORDER BY節を検索用
問い合わせ文の後ろに追加します。

ユーザが結果のページを
カラム見出しきリックせずに
読み込むと\$sortは空であり、
このためデフォルトでは
結果を全然ソートしません。

以前と同様\$search_queryを返します。今回は
ORDER BY節が後ろに付いている点が異なります。



試運転

検索用スクリプトを修理して、2つの新しいカスタム関数を使うようにします。

新しく generate_sort_links() 関数を search.php スクリプトに作ります。次に build_query() 関数に新しいコードを追加して、ソートされた結果を生成できるようにします。generate_sort_links() 関数を実際に呼び出すことを忘れないで下さい。スクリプト中で結果の見出しを echo するコードのところです。

スクリプトを Web サーバにアップロードし、search.html ページをブラウザで開いたら、検索を実行してみます。次に検索結果の上にある見出しをクリックして、異なるデータに基づいて仕事がソートされることを確認します。同じ見出しを 2 度以上クリックすると、昇順、降順が入れ替わることも確認して下さい。

The screenshot shows two windows of the 'Risky Jobs' website. The top window is a search interface with fields for '検索語' (Search term) containing '危険な仕事' (dangerous job), '都道府県' (Prefecture) set to '新潟', and '更新日時' (Update date) set to '2009-03-11'. The bottom window shows the search results for '危険な仕事'.

仕事の名称	業務概要	都道府県	更新日時
マタドール (Matador)	多忙な乳製品工場にてパートのマタドールを募集しています。元気の良い牛と遊んでやってください。ただし牛は酔い泣き欠席障害にかかることがあります。 Bulling da...	新潟	2009-03-11
消防士 (Firefighter)	データビン市では、消防士を募集しています。未経験者歓迎で、まず研修があります。既婚者はあまり好ましくありません。身体検査にパスする必要があります。高所(および熱)を恐れてはいけません。必須ではありませんが、ハングルホンキー団では、経験豊富な消防士や牛修理工を募集しています。特典として(研修後の) フリーライドおよび手羽先唐揚げの手配などがあります。	千葉	2009-05-22
電動巻れ牛修理工 (Electric Bull Repairer)	Hank's Honky Tonk ne...	神奈川	2009-07-27

generate_sort_links() 関数は、クリック可能なカラム見出しへなります。各リンクにはソートのオプションも URL に含めて詰め込みます。

build_query() 関数は、ユーザが入力した検索語を受け取り、それを配列に explode で分解し、配列内の空文字列をきれいに掃除してから、SQL 問い合わせ文を検索語から作り上げます。もちろんソート値があれば対応する ORDER BY を続けます。

さて最も古く登録された
仕事が見つかったよ。きっと新潟で
マタドールを雇うのには相当
苦労しているようだな。



もうちょっと広く検索したいと思うこともあるんだけど、結果が膨大になっちゃうよ。



リスキージョブ：危険な仕事検索

Risky Jobs
危険！夢の仕事は危険と隣り合わせです。
勇気を出して探してみませんか？

リスキージョブ：危険な仕事検索の結果

仕事の名称	業務概要	都道府県	更新日時
カスター・ウォーカー (Custard Walker)	人間がカステード上を歩くことが可能かどうかという理論の検証に協力してくれる人。広島を募集します。カスターで満たしたブルを用意しますので、そこを歩いてください。カスターで含む粘性の強い液... 新しいうォーター・マークバーで可視化できるようサメの訓練をして下さい。宮崎	宮崎	2009-04-28
サメの訓教師 (Shark Trainer)	新しいうォーター・マークバーで可視化できるようサメの訓練をして下さい。サメの恐怖の中、一人で水に入れないかもしれません。サメの訓練は夜更け、早朝、夕暮れなどデーターマークにお客様が... 屋外で直進および交配電圧の検査をしてください。電圧のレンジは3~250ボルトま... たはそれ以上です。機械型の発光ダイオードを装着してすべての電圧を測定します。 なお直進については電圧の属性の検査してください。	兵庫	2009-06-28
電圧検査士 (Voltage Checker)	屋外で直進および交配電圧の検査をしてください。電圧のレンジは3~250ボルトま... たはそれ以上です。機械型の発光ダイオードを装着してすべての電圧を測定します。 なお直進については電圧の属性の検査してください。	神奈川	2009-09-04
アンテナ設置士 (Antenna Installer)	アンテナ等の金属製放送受像装置を般に設置して下さい。神奈川最高級の安全装置として、ポリエチル製のボイラースーツとナイロン製スニーカーを完結いたします。 You'll be... 美作：経験豊富な虹門科学者で大きな動物を診ることのできる人。動物園（名古屋）に愛知あります。の象は、後部の裏庭に点くなっています。経験豊富で熟練のプロの方に、 貴重な象に対する診断、治療、およびフォローアップ... ジエット機のエンジンを清掃する必要があります。きつい好きである必要がありま... 千葉	愛知	2009-07-29
象の肛門科学者 (Elephant Proctologist)	ジエット機のエンジンを清掃する必要があります。きつい好きである必要がありま... 千葉 さびや骨のこりの除去、時には鳥の死骸を処理して貰うこともあります。常に機械的な行動をとり、迅速に通信に応じてください。時にはバ... 多忙な乳製品工場にてパートのマタドールを募集しています。元気の良い牛と遊んで、新聞やってください。ただし牛は無い注意欠陥障害にかかっています。手旗信号の経験者優選します。	千葉	2009-08-17
マタドール (Matador)	Bursting day... トッカセレブ写真工房では、パパラッチを時限付きで募集しています。神経質なロバ 東京 のグラブスターを快適に負けず追跡してください。この採用では難康保険は付与されません。 Top celebri...	東京	2009-03-11
パパラッチ (Paparazzo)	トッカセレブ写真工房でのパパラッチを時限付きで募集しています。山口 のグラブスターを快適に負けず追跡してください。この採用では難康保険は付与されません。 Top celebri...	山口	2009-11-14
網膜り (Retinotope Walker)	網膜セーピンの大テントにて本格サーカスのプロフェッショナルを募集しています。1~3年の経験を有し、網膜りのアグロバットを演じながら大きな衆と共演してください。象の頭蓋物を真んできつて頂ければなお可... 福岡	福岡	2009-07-14
ワニの歯医者 (Crocodile Dentist)	動物好きでブラー捕りの方、あなたにうってつけの仕事があります！私どものワニ 倒育館では麻利門を募集しており、愛らしいベビーワニの笑顔に働きを加えたいと考えています。これは来るべき週刊雑誌マガジンの撮影にて... 大阪	大阪	2009-11-02
	th insurance... 美しいかには自信があります。試作品をよりよく 宮城 リードの試食をしてくれる人を募集しています。	東京	2009-11-09
	を募集しています。チャンピオンの頭髪記録を仲 東京 、ダメダメなジャブ、ガラスのアゴの持ち主を求 ルタイムの正社員...	東京	2009-11-14

これは恐ろしく多い仕事のリストで、一度に全部見る気にはなれません。

脳力発揮

一般的なサイトでは、たくさんの検索結果を1ページに表示するのを避けるためにどのようにしているでしょう？

結果をページに割り付けることができます

現状では1ページに結果をすべて表示しようとしています。これだと検索がたくさんの仕事にマッチしたときに問題となります。ユーザに巨大なページのスクロールをさせて、マッチした仕事をすべて見せる代わりに、**ページ割り付け**と呼ばれるテクニックを使って、検索結果を表示することもできます。結果をページに割り付けるには、マッチした仕事をグループに分けて、各グループを分割したページに表示すればよいのです。こんな感じです。

The screenshot shows a search results page for "Risky Jobs" with the query "危険な仕事検索". The results are paginated into two pages:

- Page 1:** Includes results for "カスター・ウォーカー" (Casted Walker) and "スマーティー" (Stark Trainer). It also lists "電圧検査士" (Voltage Checker), "アンテナ設置工" (Antenna Installer), and "象の肛門科学者" (Elephant Proctologist).
- Page 2:** Includes results for "ジエットエンジン整備士" (Airplane Engine Cleaner), "マクドール" (McDowell), "ババリッチ" (Paparazzo), and "ワニの歯医者" (Crociodia Dentist).

Each result card includes a thumbnail image of a person in a work uniform and a brief description. Arrows point from the "1 2 3 4" navigation links at the bottom of each page to the corresponding page numbers on the other page, indicating that these are separate pages.

これらのリンクは複数のページへユーザを案内してくれます。

現在のページには
リンクがありません。
これは仕事の検索結果
2ページ目です。

**ごそそうだけど、結果を
こんな感じにどうやって分けるの？
SQL問い合わせ文は検索文字列にマッチ
したすべての結果を返してきたのよ。**

ページ割り付けて、
問い合わせ結果を
グループに分け、各
グループをそれぞれ
のWebページに
します。

結果すべてではなく**一部分だけ**を返してくれる
問い合わせ文が必要です。

幸せなことに、SQLにはこのような機能が備わっています。LIMIT節です。LIMITについて復習し、これを使って結果を5つずつのグループに分割する方法を見てみましょう…

各ページは結果を5件表示し、他の結果のページへのリンクも付いています。ユーザは各ページをクリックすることでページを行き来できます。スクロールは不要です。



LIMITで必要な行だけ持ってくる

与えられたページに対して、どの行を表示するかをコントロールするカギは、もう1つの節を問い合わせ文に追加することにあります。LIMIT節です。最大で5行持ってくるには、LIMIT 5を問い合わせ文の後ろに追加すればよいのです。こんな感じです。

```
SELECT * FROM riskyjobs
ORDER BY job_title
LIMIT 5
```

WHERE節がないと、この問い合わせ文はデータベースからすべての仕事を取ってきます。これは検索語がない場合の検索と同じです。

実際にどれだけ多くのものがマッチするかに問わらず、最初の5つだけがマッチして返ってきます。

カスタム関数build_query()を使ってリスクジョブサイトの問い合わせ文を作ったことを思い出してください。これをマッチした最初の5行だけを表示するようにするには、問い合わせ文ができあがった後で、LIMIT 5を問い合わせ文の文字列の後ろにくっ付けるだけよいのです。

```
$query = build_query($user_search, $sort);
$query = $query . " LIMIT 5";
```

LIMIT節を問い合わせ文の最後に追加すると、問い合わせ文から返ってくる行数を制限することができます。この場合は5行です。

これでうまくいくのは、結果から最初の5行を取ってくるときだけです。次の5行はどうすれば良いでしょうか？そしてさらにその次の5行は？結果の中から、もっと奥の方にある行を引っ張り出すには、LIMITをちょっと変更しなければなりません。でもどうやって？ LIMIT 10だと最初の10行を持ってしまいますが、これではダメです。必要なのは6行目から10行目までです。このようなことをするために、LIMITを別の構文で使う必要があります。LIMITに2つの引数を与えると、最初の引数が何行スキップするかをコントロールし、第2の引数が、そこから何行取ってくるかをコントロールします。例えば、次のように書くと11行目から15行目まで取ってきてくれます。これは3ページ目の結果となります。

```
$query = build_query($user_search, $sort);
$query = $query . " LIMIT 10, 5";
```

最初の引数はLIMITが何行スキップするかを指定します。最初の10行です。

2番目の引数は、何行返すかをコントロールします。前と同様5行です。

カスター・ウォーカー (Custard Walker)	...
サメの調教師 (Shark Trainer)	...
電圧検査士 (Voltage Checker)	...
アンテナ設置士 (Antenna Installer)	...
象の肛門科学者 (Elephant Proctologist)	...
シェットランドネズミ捕獲士 (Airplane Engine Cleaner)	...
マタドール (Matador)	...
パパラッチ (Paparazzo)	...
綱渡り (Tightrope Walker)	...
ワニの歯医者 (Crocodile Dentist)	...
パントマイム (Mime)	...
ペットフードの試食 (Pet Food Tester)	...
闘牛士 (Toreador)	...
電動累れ牛修理士 (Electric Bull Repairer)	...
消防士 (Firefighter)	...
...	

LIMITでページリンクをコントロールする

ページ割り付けで重要なのは、リンクを付けてユーザが異なるページ間を行ったり来たりできるようにすることです。LIMIT節を使って案内用のリンクを各ページの下部にセットアップすることができます。例えば、「次へ」や「戻る」のリンクは、それぞれ専用のLIMITを持たせます。同様なことは、数値リンクにもあてはまります。これらのリンクで、ユーザは特定のページに直接飛ぶことができます。

以下に検索結果の最初の3ページに対応するLIMIT節を、ページリンクのLIMIT数値と共に示します。

The image shows three screenshots of a website titled "Risky Jobs" illustrating how LIMIT controls page links.

- Screenshot 1 (Left):** Shows the first page of search results for "危険な仕事検索の結果". It includes a sidebar with job descriptions and a footer with page navigation arrows labeled "LIMIT 0, 5".
- Screenshot 2 (Middle):** Shows the second page of search results for "危険な仕事検索の結果". The footer shows "LIMIT 15, 5".
- Screenshot 3 (Right):** Shows the third page of search results for "危険な仕事検索の結果". The footer shows "LIMIT 5, 5".

A large thought bubble at the bottom contains the text:

カンタン、カンタン。
後はいろんなLIMITでそれぞれ
問い合わせ文をいくつか
書くだけでしょう？

まあ、そうです。必要なのはページとリンクに依存した異なるLIMITです。でも複数の問い合わせ文を書く代わりに、それを作り出すことができます。

やるべきことは、`build_query()` 関数をもうちょっと修正して、正しいLIMITを問い合わせ文の後ろにくっ付けることだけです

ページ割り付けデータを追跡する

新しくページ割り付け機能をbuild_query()に追加するには、いくつかの変数をセットアップして、どの検索結果を問い合わせるのか、どのページを表示するのかを記録しておく必要があります。これらの変数は、ページの下部に配置する案内用のリンクをどのようにするかを決定する上でも重要な役割を持っています。

\$cur_page

現在のページをスクリプトURLから\$_GETを通して\$cur_pageに取ってきます。現在のページがURLから渡されない場合、\$cur_pageは最初のページ(1)にセットします。

\$results_per_page

ここに1ページあたりに結果の行数を入れます。ページの見栄えをベースに適当に決めて下さい。どの程度の検索結果がそのレイアウトでページにフィットするかよく考えます。これがLIMIT節への第2引数になります。

\$skip

現在のページの行を開始する前にスキップする行の数を計算して\$skipに入れます。この変数は各ページが結果のどこから始まるのかをカントロールし、LIMIT節の最初の引数になります。

\$total

問い合わせ文を走らせて、LIMITなしですべての行を取ってきたら、結果を数えてその値を\$totalに突っ込みます。言い換えれば、これが検索結果の総数です。

\$num_pages

ページ数を計算して\$num_pagesに入れます。\$totalを\$results_per_pageで割ったものになります。検索が1つ決まれば、\$totalにマッチする行の総数が入ります。しかし結果は一度に1ページずつ、各ページにはマッチした\$results_per_page行だけが表示されます。つまり\$num_pages分のページが必要で、現在のページは\$cur_pageで分かります。

ページ割り付け用の変数をセットアップする

ほとんどのページ割り付け用変数は、URLから供給される情報だけでセットアップすることができます。これらの値は\$_GETスーパーグローバルを通してアクセスすることができます。例えば、\$sort, \$user_search, \$cur_page の3つの変数はGETデータから直接やってきます。これらの変数を使えば、最初の行を表示するためにデータを何行スキップすればよいか、つまり\$skipの値を計算することができます。\$results_per_pageはちょっと違っていて、単にセットすればよいだけです。検索結果を各ページに何行表示したいかで、結果のページをレイアウトする際の個人の趣味の問題です。

```
ソートの順序を取って  
きます。1から6までの  
整数値です。 GET を用いて URL からソート設定と探索語を取得  
// Grab the sort setting and search keywords from the URL using GET  
  
現在のページをURL  
からGETを通して取っ  
てきて$cur_page  
に入れます。まだ現在の  
ページが設定されて  
いない場合、$cur_page  
を1にセットします。 $sort = $_GET['sort']; 検索文字列を取ってきます。  
ユーザがフォームに  
打ち込んだものです。  
  
// Calculate pagination information ページ割り付け情報の算出  
  
$cur_page = isset($_GET['page']) ? $_GET['page'] : 1; ページ番号がセット  
されていない場合、  
デフォルトで最初の  
ページにします。  
  
$results_per_page = 5; // number of results per page ページ毎の結果の行数  
$skip = ((($cur_page - 1) * $results_per_page);  
この計算の結果1ページ目は0に、  
2ページ目は10に、3ページ目は  
15に、と言った具合になります。  
  
ページ毎に表示する  
結果の行数をセット  
します。 そのページが始まる最初の  
行を計算して$skipに入れます。
```

まだ重要な変数が2つ残っています。\$total と \$num_pages です。この2つの変数は1回目の問い合わせ文を実行して、データベースから何行がマッチしたのかを見つけ出した後でないとセットできません。いくつマッチするかがわかれば、これらの変数をセットすることができ、結果をLIMITすることもできます…

問い合わせ文を書き直して結果をページ割り付けする

これで変数のセットアップは終わりです。検索用スクリプトを書き直して、すべての結果を問い合わせる代わりに、結果の一部分だけを問い合わせるようにします。ユーザが現在るために必要なページだけを表示するようになります。まず問い合わせ文を発行して、\$total 変数をセットし、これにより\$num_pages 変数の値を計算します。次に第2の問い合わせ文は、\$skip と\$results_per_page を使って LIMIT 節を問い合わせ文の後ろにくっ付けます。以下に書き直した部分の search.php スクリプトを示します。追加した部分は網掛けしてあります。

```

// Query to get the total results
$query = build_query($user_search, $sort);
$result = mysqli_query($dbc, $query);
$total = mysqli_num_rows($result);

$num_pages = ceil($total / $results_per_page);

// Query again to get just the subset of results
$query = $query . " LIMIT $skip, $results_per_page";
$result = mysqli_query($dbc, $query);
while ($row = mysqli_fetch_array($result)) {
    echo '<tr class="results">';
    echo '<td valign="top" width="20%">' . $row['title'] . '</td>';
    echo '<td valign="top" width="50%">' . mb_substr($row['description'], 0, 100, 'UTF-8') . '...</td>';
    echo '<td valign="top" width="10%">' . $row['state'] . '</td>';
    echo '<td valign="top" width="20%">' . mb_substr($row['date_posted'], 0, 10, 'UTF-8') . '</td>';
    echo '</tr>';
}
echo '</table>';

mysqli_num_rows() は、問い合わせ文により全部で何行返ってきたかを数えて返してくれます。
この問い合わせ文は、LIMITなしですべての行を取ってきます。
すべての結果を取得するための問い合わせ
mysqli_num_rows() 関数の呼び出しで返ってきた行の総数を突っ込みます。
総行数を1ページあたりの行数で割ることによりページ数を計算し、結果を切り上げます。
ceil() 関数は、数値を直近上位の整数に丸めます。切り上げです。
結果の一部を取得するための再問い合わせ
この分の行をスキップして… …この分の行を返します。
2番目の問い合わせ文を発行しますが、今回は結果を現在のページに LIMIT します。

```

ページ案内用のリンクを作る

これでいくつかの変数をセットアップし、新しいSQL問い合わせ文ができるようになりました。今度はページ用に結果の一部分を返してくれます。残っている作業はページ案内用のリンクを検索結果を表示するページの下部に付けることです。「戻る」リンク、各ページを表す数字リンク、そして「次へ」リンクです。もうすべての情報はそろっていますから、これらを合わせてリンクを作るだけです。もう一度何を使って何をすればいいのかを確認しましょう。

\$user_search

すべてのページリンクは、ユーザが実際に何を検索したいのかを、ここでもわかっていないければなりません。そこで検索語を各リンクのURLに渡す必要があります。

\$num_pages

すべてのページのリンクを作る必要がありま
すから、全部で何ページあるのかを知っておく必要があります。

\$cur_page

ページ案内用のリンクは、完全に現在のページに依存しています。つまり現在のページをすべてのリンクURLに詰め込んでおく必要があります。

\$sort

ソートの順序もページ割り付けのリンクで重要な要素です。順序も合わせて面倒をみておかなければページ割り付け自体が無意味なものとなってしまいます。

さて、ページ案内用のリンクを作るのにどんな情報が必要なのかはわかりました。PHPコードを書き上げて実際に動かす準備は整ったわけです。このコードはsearch.phpの中に組み込んでも良いのですが、カスタム関数を独立のファイルにするというはどうでしょうか? そうすれば、メインのスクリプトコードは非常に単純になります。検索結果を生成するだけです。1行のコードでページリンクを作るだけでよいのです。つまりこの新しい関数の呼び出しだけです。この関数にはgenerate_page_links()という名前を付けることにします。

注意すべきことが1つだけあります。この関数は結果が1ページに収まるときには呼んで欲しくないです。そこで、まずページ数をチェックしてから、この新しい関数generate_page_links()関数を呼び出します。以下にチェックをしてから、関数を呼び出す方法を示します。必要な情報を関数の引数としてちゃんと渡していることを確認して下さい。

```
if ($num_pages > 1) {  
    echo generate_page_links($user_search, $sort, $cur_page, $num_pages);  
}
```

まず検索結果のページ数が2ページ以上で
あることを確認します。1ページの場合は
ページリンクを作る必要はありません。

↑ 検索結果、ソートの順序、現在の
ページ、それとページ総数を渡して、
ページリンクを作ります。



PHP & MySQL マグネット

`generate_page_links()` 関数はほとんど完成しています。しかしコードの一部分が欠けています。マグネットを使って抜けているコードを埋め、リスキー・ジョブサイトにページ案内用のリンクを作るようパワー・アップして下さい。

```

function generate_page_links($user_search, $sort, $cur_page, $num_pages) {
    $page_links = '';

    // If this page is not the first page, generate the "previous" link
    if (...) {
        $page_links .= '<a href="' . $_SERVER['PHP_SELF'] .
            '?usersearch=' . $user_search .
            '&sort=' . $sort .

        '&page=' . (...) . '"><-</a> ';
    }
    else {
        $page_links .= '<- ';
    }
}

// Loop through the pages generating the page number links
for ($i = 1; $i <= $num_pages; $i++) {

    if (...) {
        $page_links .= ' ' . $i;
    }
    else {
        $page_links .= '<a href="' . $_SERVER['PHP_SELF'] .
            '?usersearch=' . $user_search .
            '&sort=' . $sort .
            '&page=' . $i . '">' . $i . '</a>';
    }
}

// If this page is not the last page, generate the "next" link
if (...) {
    $page_links .= '<a href="' . $_SERVER['PHP_SELF'] .
        '?usersearch=' . $user_search .
        '&sort=' . $sort .
        '&page=' . ($cur_page + 1) . '">-></a>';
}
else {
    $page_links .= '->';
}

return $page_links;
}

```

このページが最初のページでない場合「戻る」リンクを生成

「戻る」リンクは左向き矢印
「<-」で表します。

ページを走査しページ番号リンクを生成

このページが最後のページでない場合「次へ」リンクを生成

「次へ」リンクは右向き矢印「->」で表します。



PHP & MySQL マグネットの答え

generate_page_links() 関数はほとんど完成しています。しかしコードの一部分が欠けています。マグネットを使って抜けているコードを埋め、リスクイジョブサイトにページ案内用のリンクを作るようパワーアップして下さい。

```
function generate_page_links($user_search, $sort, $cur_page, $num_pages) {
    $page_links = '';
```

// If this page is not the first page, generate the "previous" link

```
if (. $cur_page > 1.) {
    $page_links .= '<a href="" . $_SERVER['PHP_SELF'] .
        '?usersearch=' . $user_search .
        '&sort=' . $sort .
        '&page=' . (. $cur_page - 1. ) . '"><-</a> ';
```

ユーザの検索語やソート順序
は相変わらず各リンクURL
へ渡す必要があります。

```
} else {
    $page_links .= '<-';
```

「戻る」リンクは
左向き矢印「<-」
で表します。

// Loop through the pages generating the page number links
for (\$i = 1; \$i <= \$num_pages; \$i++) {

```
if (. $cur_page == $i.) {
    $page_links .= ' ' . $i;
} else {
    $page_links .= ' <a href="" . $_SERVER['PHP_SELF'] .
        '?usersearch=' . $user_search .
        '&sort=' . $sort .
        '&page=' . $i . '"> ' . $i . '</a>';
```

各ページリンクは、同じスクリプトに
戻ってきます。単に各リンクに異なる
ページ番号を渡しているだけです。

特定のページへのリンクと
いうのは、単にページ番号です。

// If this page is not the last page, generate the "next" link

```
if (. $cur_page < $num_pages.) {
    $page_links .= ' <a href="" . $_SERVER['PHP_SELF'] .
        '?usersearch=' . $user_search .
        '&sort=' . $sort .
        '&page=' . ($cur_page + 1) . '">-></a> ';
```

```
} else {
    $page_links .= ' ->';
```

「次へ」リンクは右向き
矢印「->」で表します。

```
return $page_links;
```

```
}
```

すべてを合わせて検索用スクリプトを完成させる

やっとリスクジョブサイトの検索用スクリプトを完成させることまできました。これでユーザの検索語に基づく適切な検索結果を表示できるようになります。クリック可能な結果の見出しリンクがあつてソートできます。結果のページ割り付けもできます。さらにページ案内用のリンクも各ページの下部には付いています。

```
<?php
    // This function builds a search query from the search keywords and sort setting
    function build_query($user_search, $sort) {
        ...
        return $search_query;
    }
    // This function builds heading links based on the specified sort setting
    function generate_sort_links($user_search, $sort) {
        ...
        return $sort_links;
    }
    // This function builds navigational page links based on the current page and
    // the number of pages
    function generate_page_links($user_search, $sort, $cur_page, $num_pages) {
        ...
        return $page_links;
    }
    // Grab the sort setting and search keywords from the URL using GET
    $sort = $_GET['sort'];
    $user_search = $_GET['usersearch'];
    // Calculate pagination information
    $cur_page = isset($_GET['page']) ? $_GET['page'] : 1;
    $results_per_page = 5; // number of results per page
    $skip = (($cur_page - 1) * $results_per_page);
    // Start generating the table of results
    echo '<table border="0" cellpadding="2">';
    // Generate the search result headings
    echo '<tr class="heading">'; 検索結果見出しの生成
    echo generate_sort_links($user_search, $sort);
    echo '</tr>';

    // この関数で検索語とソート設定から検索用問い合わせ文を構築
    // これらの関数はすでに作ってありますから、ここで改めて蒸し返す必要はないでしょう。
    // この関数で指定されたソート設定に基づく見出しリンクを構築
    // この関数で現在のページとページ数から案内用のページリンクを構築
    // ソート順序と検索用文字列を取ってきます。URLを通してGETデータとして渡されたものです。
    // GETを使ってURLからソート設定と検索語とを取得
    // ページ割り付け用変数を初期化します。
    // これらの変数は問い合わせ文をLIMITしてページ割り付け用リンクを作るので、すぐに必要になります。
    // ページ割り付け情報の算出
    // ページ毎の結果の行数
    // 結果テーブルの生成を開始
    // generate_sort_links()関数を呼び出して、結果の見出しリンクを作り、できたらそれをechoします。
    // ちょっと待って下さい。まだあります！
```

The diagram illustrates the data flow in the script:

- User Input:** A mobile phone icon labeled "search.php" represents the user input.
- GET Parameters:** Arrows point from the phone to the script, indicating the extraction of \$sort and \$user_search via \$_GET.
- Pagination:** An arrow points from \$cur_page and \$num_pages to the calculation of \$skip.
- Results Per Page:** An arrow points from \$results_per_page to the calculation of \$skip.
- Table Generation:** Arrows point from \$sort and \$user_search to generate_sort_links().
- Heading Row:** An arrow points from \$sort and \$user_search to the opening of the table and the generation of the heading row.
- Body Rows:** An arrow points from \$sort and \$user_search to the main loop where rows are generated using generate_sort_links().
- Final Output:** An arrow points from the completed table back to the "search.php" mobile phone icon.

完成版検索用スクリプト、続き…

```
// Connect to the database データベースへの接続
require_once('connectvars.php');
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);
すべての結果を取得するための問い合わせ build_query() を呼び出して、仕事検索
// Query to get the total results のSQL問い合わせ文を作ります。
$query = build_query($user_search, $sort);
$result = mysqli_query($dbc, $query);
$total = mysqli_num_rows($result);
$num_pages = ceil($total / $results_per_page);
結果の一部を取得するための再問い合わせ ここがLIMIT節で、仕事の
// Query again to get just the subset of results 検索結果の一部のみ問い合わせ
$query = $query . " LIMIT $skip, $results_per_page";
$result = mysqli_query($dbc, $query);
while ($row = mysqli_fetch_array($result)) {
    echo '<tr class="results">';
    echo '<td valign="top" width="20%">' . $row['title'] . '</td>';
    echo '<td valign="top" width="50%">' . mb_substr($row['description'], 0, 100, 'UTF-8') .
        '...</td>';
    echo '<td valign="top" width="10%">' . $row['state'] . '</td>';
    echo '<td valign="top" width="20%">' . mb_substr($row['date_posted'], 0, 10, 'UTF-8') .
        '</td>';
    echo '</tr>';
}
echo '</table>';

// Generate navigational page links if we have more than one page
if ($num_pages > 1) {
    echo generate_page_links($user_search, $sort, $cur_page, $num_pages);
}

mysqli_close($dbc);
?>
データベースの接続を閉じて generate_page_links() 関数を呼び出して、
行儀よく終了します。 ページリンクを作り echo します。
```

素朴な疑問に答えます

 **Q:** generate_page_links() に検索語、ソート順序、およびページ割付の情報を本当に全部渡す必要があるのでしょうか？

A: あります。そしてそうしなければならない理由は、適切に設計された関数というものは、関数自身のコード外にあるデータを操作すべきではないからです。つまり関数は引数として渡されたデータにのみアクセスすべきであり、かつ返却するデータのみを変更すべきなのです。

 **Q:** わかりました。ではデータを echo するのはどうでしょうか？ generate_page_links() がリンクを echo するのは何故ダメなのでしょうか？

A: 問題は同じです。ブラウザにデータを echo することによって、関数は簡単に自分自身を越えてどこか別のところを変更できてしまいます。このため関数のデバッグは難しくなり、メンテナンスも大変になります。関数がどのデータを変えたのか明確でないからです。これを解決するには、関数はそれが影響を与えたデータのみを返すことであり、その後その関数によって返されたデータに対して望みの処理を、**関数の外側**で施すことです。



試運転

リスキージョブサイトのスクリプトを仕上げる

新しく `generate_page_links()` を `search.php` スクリプトに追加します。結果が2ページ以上になる場合は、この関数を呼び出すコードを追加することも忘れないで下さい。さらにこの変数への引数として使う変数を作成して初期化して下さい。また問い合わせ文のコードをアップデートします。LIMITを使って各ページに適した結果の一部分を取ってくるようにします。

すべてできあがったら、新しい `search.php` スクリプトを Web サーバにアップロードし、`search.thml` ページを Web ブラウザで開きます。いくつかの検索を実行して、たくさんの結果が出るような探索語に対しては、新しくページ割り付け機能が組み込まれていることを確認して下さい。結果のページを最大にするには、空の検索フォームで検索を実行すればよいのです。

Risky Jobs

危険！夢の仕事は危険と隠り合わせです。
勇気を出して探してみませんか？

リスキージョブ：危険な仕事検索の結果

仕事	都道府県	更新日時
マタドール (Matador)	新潟	2009-03-11
ババラッチ (Paparazzo)	東京	2009-03-24
サメの調教師 (Shark Trainer)		2009-04-28
消防士 (Firefighter)		2009-05-22
電圧検査士 (Voltage Checker)		2009-06-28

<- 1 2 3 4 ->

とうとう夢の仕事が
見つかった！新潟へ行こう。

ゴン太は完璧に十分
危険な仕事を見つける
ことができました！

ダウンロードして下さい！

覚えていますか？リスキージョブサイトアプリケーション
の全ソースコードはWebサイト
<http://www.oreilly.co.jp/books/9784873114446/>
からダウンロードできます。



PHP & MySQL 道具箱

リスキージョブサイトの検索用スクリプトは極めてたくさん新しいPHPとMySQLの技法を必要としました。重要なものについて復習しておきましょう。

LIKE

LIKEを使うとSQL問い合わせ文で必ずしも完全に一致しないデータを探すことができます。%記号を検索語の前後におくことで、検索語の周りに別の文字があってもよいことをLIKEに知らせることができます。

explode(), implode()

PHPのexplode()関数は、文字列を部分文字列の配列に分解します。分解する際には共通の分離子として一般に空白やコンマなどを使用します。

implode()はexplode()の逆をやります。こっちは文字列の配列から1つの文字列を作り、それぞれの間に分離子を入れます。

str_replace()

このPHP関数を呼び出すと、テキストの文字列上「見つけたら置き換える」処理を実行します。1文字か一連の文字を別のものに置き換えます。

switch-case

判断に基づく処理を行うためのPHPの構文要素で、1つの値に基づいていくつかのコードグループのうちの1つを実行することができます。もしたくさんのif-else文でネストしてしまうような箇所を見つけた場合、switch文を使うことでもっと効率的にコーディングすることができるかもしれません。

mb_substr()

このPHP関数は日本語を含む文字列の一部分を引数として与えられた情報に基づいて取り出します。文字列の先頭から取り出すことも、末尾から取り出すことも、途中の一部を取り出すこともできます。

カスタム関数

PHPの一連のコードで名前を付けて再利用可能なようにパッケージ化して構成したもの。考え方としては、特定の処理を行うコードを独立させて、最小の努力と最小のコード重複で再利用できるようにすることを指します。

LIMIT

LIMIT節でSQL問い合わせ文から正確に何行返ってくるかをコントロールできます。それだけでなく、LIMITを使うと結果の中から行をスキップすることもできますから、結果の一部分だけを取り出することができます。

10章 正規表現

置き換えの規則



Pratt 先生ったら
 クラスのハムスターを置き
 換えちゃったのよ！アシシ
 たちが気付かないとでも
 思ってるのかしら？

文字列を扱う関数は愛用できるものばかりです。しかし同時に非常に制限があります。

当然ながら、文字列の長さを教えてくれたり、一部を切り取ってくれたり、特定の文字を別の文字に変えてくれたりします。でも時にはもっと自由になって、もっと複雑な文字列操作に挑戦する必要が出てくることがあります。こんな時こそ正規表現の出番です。正規表現を使えば 1 つの基準ではなく規則の集合に基づいて、文字列を正確に変更することができます。

リスキージョブサイトから履歴書（レジュメ）を送付できるようにする

riskyjobs.bizはどんどん大きくなっています。会社では求職者に対してWebフォームに履歴書と連絡先を入力させ、危険な仕事の雇用者側が簡単に求職者を見つけるようにしています。フォームは以下のよう感じです。

通常の連絡先情報に
加えて、リスキージョブサイトでは、
希望の職種を入力
します。もちろん
履歴書もです。

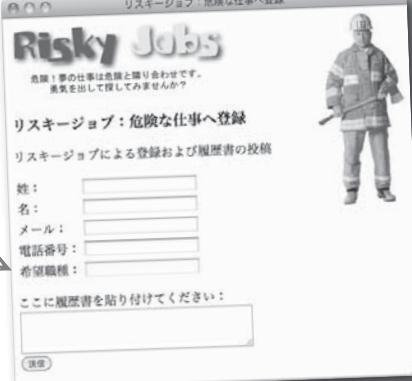
リスキージョブ：危険な仕事へ登録

リスキージョブによる登録および履歴書の投稿

姓：
名：
メール：
電話番号：
希望職種：

ここに履歴書を貼り付けてください：

送信



新しくリスキージョブサイトの登録
フォームを作って、仕事の候補者は
自分自身の情報を入力し、雇用者側
が見つけやすいようにします。

求職者の情報は、テーブルに格納されていて、事業主やリクルーターやヘッドハンターが新しい従業員の候補を見つけるために検索されます。ところが問題があります…フォームに投入されたデータは明らかに信用できないのです！

まず、この忍者っていうのはダメだな。
電話番号が間違っている。さらにこのナイフジャグラーに
送ったメールはエラーで戻ってきた。リスキージョブサイト
の履歴書バンクに登録されたデータは、
この手の間違いばかりだな。

姓：横山
名：憲治
メール：yken@sim-u-duck.com
電話番号：636 4652
希望職種：忍者(Ninja)



事業主はリスキージョブ候補者データ
ベースを検索して、適切な人を雇おう
とコンタクトをとります…ただし連絡
先の情報が十分かつ正確に入力されて
いればの話です！

姓：宮本
名：4本指のリキ
メール：riki-four@gregs-listnet
電話番号：080-098
希望職種：ナイフジャグラー
(Knife Juggler)





以下のコードは、registration.phpスクリプトの一部です。このスクリプトでユーザが入力したデータを表示し、処理して、仕事の新しい候補者として登録します。このコードのどこが問題と思うか書き込んで下さい。また、間違ったデータの問題を解決するにはどのように変更すべきかも示して下さい。

```
<?php
if (isset($_POST['submit'])) {
    $last_name = $_POST['lastname'];
    $first_name = $_POST['firstname'];
    $email = $_POST['email'];
    $phone = $_POST['phone'];
    $job = $_POST['job'];
    $resume = $_POST['resume'];
    $output_form = 'no';

    if (empty($last_name)) {
        // $last_name is blank $last_nameが空
        echo '<p class="error">エラー：「姓」が入力されていません。</p>';
        $output_form = 'yes';
    }

    if (empty($first_name)) {
        // $first_name is blank $first_nameが空
        echo '<p class="error">エラー：「名」が入力されていません。</p>';
        $output_form = 'yes';
    }

    if (empty($email)) {
        // $email is blank $emailが空
        echo '<p class="error">エラー：メールアドレスが入力されていません。</p>';
        $output_form = 'yes';
    }

    if (empty($phone)) {
        // $phone is blank $phoneが空
        echo '<p class="error">エラー：電話番号が入力されていません。</p>';
        $output_form = 'yes';
    }
}
else {
    $output_form = 'yes';
}

if ($output_form == 'yes') {
?>
...
    空でない仕事フィールドと履歴書
    フィールドの妥当性検証の続き。
...
    フォームを表示する部分。
}
```


**エクササイズ
の答え**

以下のコードは、registration.phpスクリプトの一部です。このスクリプトでユーザが入力したデータを表示し、処理して、仕事の新しい候補者として登録します。このコードのどこが問題と思うか書き込んで下さい。また、間違ったデータの問題を解決するにはどのように変更すべきかも示して下さい。

```
<?php
if (isset($_POST['submit'])) {
    $last_name = $_POST['lastname'];
    $first_name = $_POST['firstname'];
    $email = $_POST['email'];
    $phone = $_POST['phone'];
    $job = $_POST['job'];
    $resume = $_POST['resume'];
    $output_form = 'no';

    if (empty($last_name)) {
        // $last_name is blank
        echo '<p class="error">エラー：「姓」が入力されていません。</p>';
        $output_form = 'yes';
    }

    if (empty($first_name)) {
        // $first_name is blank
        echo '<p class="error">エラー：「名」が入力されていません。</p>';
        $output_form = 'yes';
    }

    if (empty($email)) {
        // $email is blank
        echo '<p class="error">エラー：メールアドレスが入力されていません。</p>';
        $output_form = 'yes';
    }

    if (empty($phone)) {
        // $phone is blank
        echo '<p class="error">エラー：電話番号が入力されていません。</p>';
        $output_form = 'yes';
    }
}

else {
    $output_form = 'yes';
}

if ($output_form == 'yes') {
?>
... フォームを表示する部分。
}
```

このスクリプトはフォームフィールドが空かどうかチェックしています。これ自体は良いことなのですが、フォームフィールドによってはもっと特別なデータで、具体的なフォーマットに従わなければならないこともあります。

「姓」と「名」に関しては他にチェックできることは見当たりませんので、このコードで十分です。

メールアドレスは極めて限定的なフォーマットを持っていますから、ユーザのフォームデータを受理する前に厳しくチェックすべきです。

4本指のリキちゃんはメールアドレスの最後の方にドットを忘れてしまっています。この種の誤りはフォームのチェックで見つけるべきです！

同じことが電話番号にもあてはまります。ユーザのフォームを受け付ける前に、電話番号が正しいフォーマットかどうかを確認すべきです。

横山憲治さんは、電話番号に市外局番を付け忘れたようです。これはフォームでチェックしておくべきでした。

結局本当に必要なものはメールアドレスと電話番号を検証する方法です。この2つのフィールドは、フォームの中で特定のフォーマットを持つものだからです。これら以外のフィールドについては、空でないことを確認するくらいで良いでしょう。



何か文字列操作関数を使って
間違ったデータを直せないの?
str_replace()使って足りない
データを足せばいいじゃない?

文字列操作関数で直せるデータもありますが、関数でできることは限られているので、データが極めて特殊なパターンにフィットしなければならないような場合には使えません。

文字列操作関数は、単純な検索や置き換え操作に適しています。例えば、ユーザが電話番号の区切り文字としてハイフンではなくドットを使って送りつけようとした場合、`str_replace()`を使ってハイフンに置き換えるようなコードを書くことは簡単にできます。

しかし想定していないことについてはどうでしょうか? 例えば横山憲治さんの電話番号には市外局番がありません。このような場合、フォームを送ってきたその人に対して明示するようにお願いする必要があります。さらに彼が市外局番を入れ忘れたことを知る唯一の方法は、電話番号の正確なパターンをあらかじめ知っておくことです。ここで要求されているのは、もっと進んだ妥当性検証であって、電話番号やメールアドレスのようなものが、きちんと入力されたかどうかを確認する手段です。

わかったわ。でもやっぱり文字列
操作関数でその妥当性検証を
すればいいじゃない?

文字列操作関数は最も基本的なデータ検証以外ではあまり役に立ちません。

メールアドレスを文字列操作関数で検証できるかどうか考えてみましょう。PHPには`strlen()`という関数があって、文字列が何文字か教えてくれます。しかしメールアドレスには長さの制限はありません。確かにこの関数で電話番号を検証できる可能性はあります。なぜなら電話番号は通常一定の長さの番号だからです。しかし、ドットやハイフンやカッコなどを取り扱わなければならない可能性は残っています。

メールアドレスに戻りましょう。このフォーマットは文字列操作関数を使って何とかするには、あまりにも複雑過ぎます。今欲しいものは特定のデータのパターンです。これにより妥当性検証の方法が得られ、ユーザデータがパターンとして適正かどうかをチェックできることになります。フォームデータをモデル化できるパターンこそが、この種の妥当性検証の心臓部なのです。

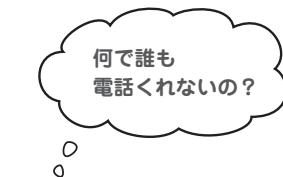


データがどうあるべきかを決める

やらなければならないことは、与えられたフォームデータが正確にはどうあるべきかを、各文字に至るまで明確に規定することです。横山さんの電話番号について考えてみます。人間が見れば、電話番号に市外局番が抜けていることは極めて明らかです。しかしどの検証は人間がやるのではありません。PHPのコードがやるのです。つまりユーザが入力したデータの文字列がどうあるべきかをコードに「教え込む」必要があります。電話番号としてのパターンにマッチしているかどうかを決定しなければならないのです。そのようなパターンを見つけ出すことは通常容易ではなく、データがどのようなタイプなのかについて可能性を色々と考える必要があります。電話番号の場合は、比較的容易です。なぜなら10桁または11桁の数字プラス区切り文字が追加されていることがあるくらいだからです。メールアドレスの場合は話が全然変わってきます。こちらについては本章の少し後の方で改めて考えることにします。

人間にとっては横山さんが市外局番または携帯電話番号の先頭3桁を忘れたことを見つけるのは簡単ですが、PHPコードで同じような「観測」をするというのはそんなに単純ではありません。

姓：横山
名：憲治
メール：yken@sim-u-duck.com
電話番号：636 4652
希望職種：忍者(Ninja)



素朴な疑問に答えます

Q：まだよくわからないのですが、`isset()`と`empty()`だけではフォームの妥当性検証としてはダメなのは何故でしょうか？

A：この2つの関数は誰かが「提出」したフォームのテキストフィールドにデータが入っているかどうかを判定します。しかしユーザが打ち込んだ実際のデータについては何も教えてくれません。`empty()`関数に限ってみても、フォームの電話番号フィールドにユーザが「(077)827-700」と入力しても「4FG8SXY12」と入力しても全く区別がありません。これはリスキージョブのようなサイトでは大問題となります。このデータが正しくなければ、候補者は仕事にありつけることができません。

Q：もし`isset()`や`empty()`がダメなら、単に誰かにチェックしてもらってからデータベースに入れるというはどうでしょうか？

A：大丈夫です。ただし通常これでは間違ったデータを直すには遅すぎます。例えば電話番号に市外局番がない場合、ユーザにお願いして問題点をはつきりさせてもらい、そのフォームフィールドのデータを再度提出してもらう必要があります。ちょっとの間待って後でデータを一度にチェックしようとするのであれば、データはすでにデータベースの中に入ってしまいます。ユーザにデータの一部が間違っていることを知らせるために連絡する手段はもうありません。ユーザは恐らく自分が間違っていることに気付かないでしょうから、何が間違っている

のかもわからないでしょう。

ですから最も良いのは、ユーザのフォームデータが「提出」された瞬間に検証することです。そうすればエラーメッセージを表示してフォームを再入力させることができます。

Q：それではユーザが入力したデータが妥当かどうかをどうやったら決めることができるのですが？

A：それはどんな種類のデータかに依存します。異なるタイプの情報は異なる規則に従わなければなりません。どんな文字種が許されるのか、何文字まで許されるのか、どのような順序で文字が入っているのか、などです。このような規則をPHPコードの中に組み込む必要があります。では電話番号についての規則をもう少し詳しく見ていきます…



自分で考えてみよう

固定電話の番号について、思いつく限りの異なる表現方法を書いて下さい。
市外局番は3行で考えて下さい。

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

ここから規則を
書いて下さい。

フォームを埋める際に、ユーザに従って欲しい規則として理にかなっていると思えるもの
は何でしょうか？例えば、電話番号は数字以外であってはいけない、といったものです。
数字のみで10桁でなければならないという規則。

.....
.....
.....
.....
.....

自分で考えてみよう の答え

固定電話の番号について、思いつく限りの異なる表現方法を書いて下さい。
市外局番は3桁で考えて下さい。

空白、ハイフン、左右のカッコ、
時にはドットも電話番号の中に
現れることがあります。

051 636 4652
(051) 636-4652
(051)636-4652
(051) 6364652
051636-4652
051 636-4652
051.636.4652
0516364652
051-636-4652
051 ME NINJA

アルファベットを電話番号に含める
ことも可能ですが。ただし妥当な番号と
して何を認めるかという制限を緩める
ことになります。

ここから規則を
書いて下さい。

フォームを埋める際に、ユーザに従って欲しい規則として理にかなっていると思えるもの
は何でしょうか？例えば、電話番号は数字以外であってはいけない、といったものです。
数字のみで 10 行でなければならないという規則。

番号を 3 つのフィールドに分け、最初の 3 行用、次の 3 行用、最後を残りの
4 行用とする。

代替案として、フォームに入力する際には必ず (051)636-4652 というような
形式でなければならないとする。この 2 つは我々規則を決める側の問題です。

パターンっていうんな可能性が
あるのね。全部をカバーする
規則なんてできるのかしら？



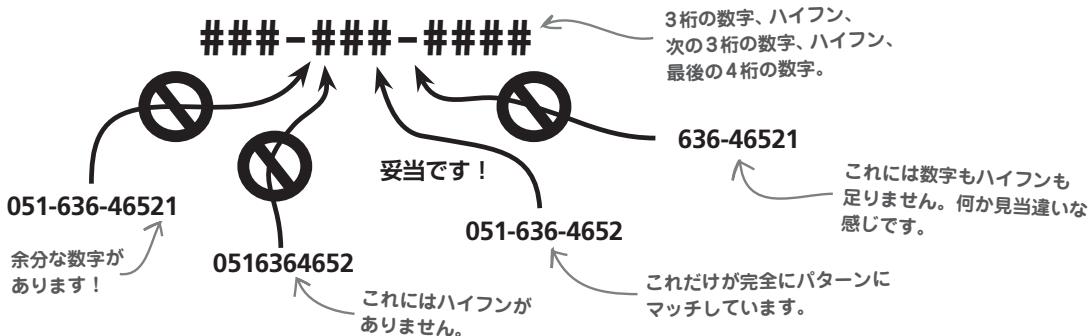
電話番号についてなら確信を持って言えることがいくつかあります。
これを使って規則を作りましょう。

まず、最初の数字は 0 です。次に電話番号は数字のみです。固定電話の場合は、
市外局番を入れて全部で 10 行です。携帯電話や IP 電話に関しては、最初の 3
桁は、090、080、070、050 のいずれかです。その先に数字が 8 行続き全部で
11 行です。

電話番号のパターンを定式化する

`empty()` や `isSet()` といった基本的な検証を越えるためには、パターンというものを決めて、データをそれにマッチさせる必要があります。電話番号の場合で言うと、単一のフォーマットのみをフォームの電話番号フィールドから受理すると決めてしまうことを意味します。ひとたび電話番号のフォーマット／パターンを決めてしまえば、それに反しているかどうか検証することができるようになります。

以下に現在最も普及していると思われる電話番号のフォーマットを示します。少なくとも日本国内の電話番号についての話です。話を単純化するため、ここでは市外局番が3桁の固定電話の番号のみを考えることにしましょう。ここでフォーマットに決めてしまうということは、ユーザが「提出」した電話番号データがこのフォーマットに従っていない場合、PHPスクリプトはフォームを受理せず、エラーメッセージを表示することを意味します。



素朴な疑問に答えます

Q: 電話番号とマッチさせるためには、このようなパターンを使わなければならないのでしょうか？

A: リスキージョブサイトでこれを使うことにしたのは、極めて標準的だからです。ただし、自分でフォームを設計する際には、自分が納得できるものに変えて構いません。ただし一般的に受け入れられているパターンを使えば、ユーザはそれに従いやすいということは心に留めておいて下さい。

Q: 単に#####というようなパターンをユーザに提示して、PHPの文字列操作関数を使ってデータが10桁の数字かどうかを確認するというのはダメでしょうか？

A: ダメということはありません。ユーザがそのようなパターンを想定しているのであれば、それで十分でしょう。ただし残念なことに、これはあまり良いパターンとは言えません。なぜならたいていの人は、フォームを埋めるときに自分の電話番号をこのようにひとまとめにしないからです。ちょっと標準的とは言えません。つまりユーザはこのパターンには慣れていないので、従ってくれなさそうです。

Q: そうですか？これって私のパターンですよ。私が私のしたいようにもいいんでしょう？

A: もちろんです。でも自分の作ったものがユーザに受け入れられる操作性であって欲しいはずです。そうでなければ、ユーザはサイトに来てくれなくなってしまいます。

Q: わかりました。では3つのテキストファイルを使って電話番号を表すというはどうでしょうか？1つ目が市外局番で、次に市内局番、最後に残りの4桁です。それからPHPの文字列操作関数を使えば良いのではないか？

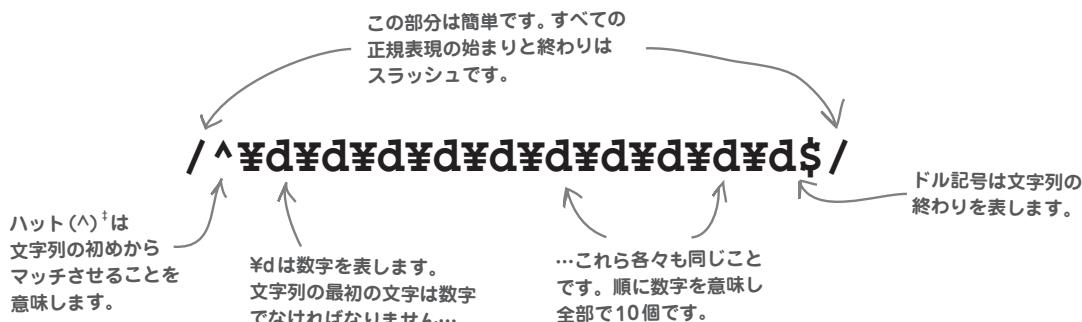
A: もちろんOKです。実際サイトによってはそのようにしています。ただ、パターンにマッチすることができれば、柔軟性が高まるということです。またパターンにマッチするということは、色々な意味で便利です。単にユーザが打ち込んだ電話番号が正しいパターンかを確認する以上の意味がありますが、これについては本章で後述します。

正規表現でパターンにマッチさせる

PHPにはテキストに対してパターンを作つてマッチさせる強力な方法があります。規則を作つて文字列テキストの中からパターンを探すことができるのです。この規則のことを**正規表現**、省略形は**regex**[†]と言います。正規表現はマッチさせる文字のパターンを表現します。正規表現を使えば、コード中に、文字列が満足すべき規則を書いてマッチするものが現れるかどうか調べることができます。

一例として、行の中から10桁の数字を探す正規表現を示します。このパターンは10桁の数字から成る文字列のみとマッチします。文字列が10文字より長くても短くともマッチしません。文字列に数字以外のものが入ついてもマッチしません。もう少しブレークダウンしてみます。

[†] 訳注：正規表現を英語では regular expression と言います。これを略して regex です。



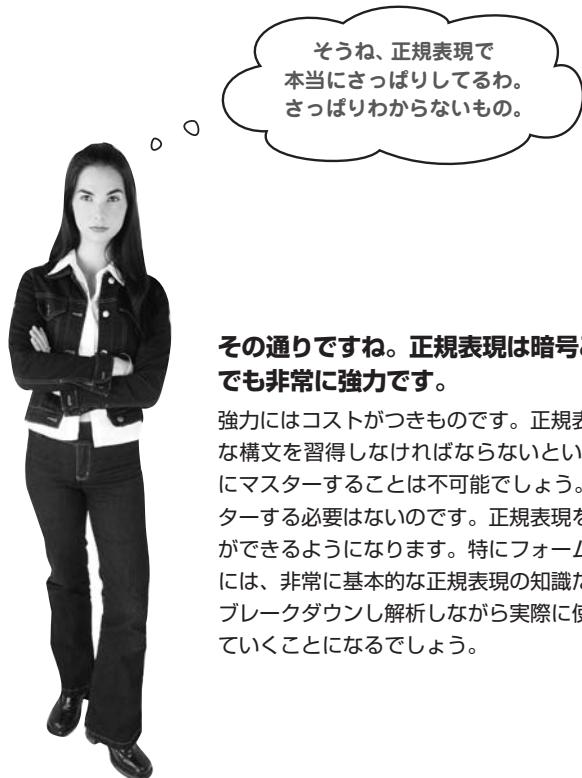
同じことをもう少し簡潔な正規表現で書くこともできます。波カッコを使うことになります。波カッコは繰り返しを表します。

`/^\$d{10}\$/`

これは上記と同じパターンを意味します。`{10}`で10個の数字があることの省略形です。

正規表現は1つ以上の文字列にパターンをマッチさせる場合に使われる規則です。

[‡] 訳注：`^`は英語では caret と言います。日本語では脱字記号、挿入記号、カレット、ハットなどと呼ばれ、あまり統一されていないようです。個人的には「へ」をよく使いますが、本書ではハットと呼ぶことにします。



**その通りですね。正規表現は暗号みたいで読みにくいものです…
でも非常に強力です。**

強力にはコストがつきものです。正規表現の場合、コストというのは暗号的な構文を習得しなければならないということです。一晩で正規表現を完全にマスターすることは不可能でしょう。ただしありがたいことに完全にマスターする必要はないのです。正規表現を使えば、驚くほど強力で便利なことができるようになります。特にフォームフィールドの妥当性検証に使う場合には、非常に基本的な正規表現の知識だけでできます。一方で、正規表現をブレークダウンし解析しながら実際に使っていくことで、だんだんとわかっていくことになるでしょう。

メタ文字を使ってパターンを作る

`\d`を使ってテキスト文字列にある数字とマッチさせるのは非常に簡単です。しかし正規表現で提供されている機能がそれしかないのであれば、正規表現は非常に限られた使い方しかできないことになります。リスキージョブサイトの電話番号を検証する機能に限定したとしても数字をマッチさせるだけでは全く不十分です。空白やハイフンやアルファベットといった文字にもマッチさせる必要があるからです。

好都合なことに、PHP正規表現の機能には、`\d`のような多くの特殊な記法があって、色々な文字にマッチさせることができます。これらの記法のことを**メタ文字**と言います。非常によく使われる正規表現のメタ文字を少し見てみましょう。

メタ文字を使うと正規表現の中でテキストのパターンを記述することができます。

¥d

前のページすでに見たように、このメタ文字は数字を意味し、0から9までのどの数字ともマッチします。注意して欲しいのは`\d`は、1つの数字のみとマッチするということです。従って2つの数字とマッチさせたいような場合、`\d\d`と書くかまたは`\d{2}`と書く必要があります。

¥w

任意のアルファベットまたは数字を意味します。アルファベットでも数字でもどちらでも構いません。以下に示す文字のどれか1つです。a-z, A-Z(大文字、小文字のいずれでも)、および0-9(`\d`と同じ)。

¥s

空白類文字を意味します。スペースキーを押すことによって入力される空白文字だけでなく、タブ文字、改行文字(NL)、および復帰文字(CR)も含まれます。ここでも`\s`は一度に1文字にしかマッチしないことには注意して下さい。従って2つの空白類とマッチさせたいような場合、`\s\s`と書くかまたは`\s{2}`と書く必要があります。

このハットについても前のページで見ています。文字列の初めを表します。つまりこれを使えば、テキスト文字列の初めからマッチしなければならないことを指示することができます。文字列のどこからでもいいのではない、という意味です。例えば、`/^\d{3}/`は、「300のアプリケーション」とはマッチしますが、「ここに300のアプリケーションがあります。」とはマッチしません。

メタ文字ドットは、改行文字を除く任意の文字とマッチします。`\w`と同様にアルファベットや数字とマッチしますし、`\s`と同様に空白やタブともマッチします。

\$

文字列の終わりを表します。このメタ文字と`^`とを使って、マッチを挟むことができます。つまり始まりと終わりをきちんと指定できます。例えば、`/^\w{5}\s\w{3}$`には「Nanny 411」はマッチしますが「Nanny 411 is great」や「Call Nanny 411」にはマッチしません。

これらのメタ文字は非常にイケていますが、正規表現で特定の文字を本当に指定したい場合はどうでしょう？実は単にその文字を正規表現の中で使うだけよいのです。例えば、特定の電話番号に「077-827-7000」にマッチさせたい場合は、正規表現として`/077-827-7000/`を使えば良いのです。



誰が何をする？

電話番号を表す正規表現とマッチする電話番号とをマッチさせて下さい。

正規表現

```
/^\$\d{3}\$\$\d{7}\$/
```

マッチする文字列

0516364652

```
/^\$\d{3}\$\$\d{3}\$\$\d{4}\$/
```

051 636 4652

```
/^\$\d{3}\$\d{3}-\$\d{4}\$/
```

051636-4652

```
/^\$\d{3}-\$\d{3}-\$\d{4}\$/
```

051 ME NINJA

```
/^\$\d{3}\$\$\w\$\w\$\w\$\w\$\w\{5}\$/
```

051 6364652

```
/^\$\d{10}\$/
```

051-636-4652

誰が何をする？の答え

電話番号を表す正規表現とマッチする電話番号とをマッチさせて下さい。

正規表現

マッチする文字列

/^\\$\{3\}\\$\\$\{\{7\}\\$/\$

0516364652

/^\\$\{3\}\\$\\$\{\{3\}\\$\\$\{\{4\}\\$/\$

051 636 4652

/^\\$\{3\}\\$\\$\{\{3\}-\\$\{4\}\\$/\$

051636-4652

これはリスキージョブサイトで必要となる
パターンの1つで##-##-##-##-#
という形式の電話番号とマッチします。

/^\\$\{3\}-\\$\{3\}-\\$\{4\}\\$/\$

051 ME NINJA

/^\\$\{3\}\\$\\$\{w\}\\$\\$\{5\}\\$/\$

051 6364652

このパターンのメタ文字\\$wは
アルファベットと数字にマッチします。

/^\\$\{10\}\\$/\$

051-636-4652

このパターンはすべて数字を
表します。つまり空白やハイフンの
ない電話番号にのみマッチします。

正規表現になったつもりで

正規表現の役割を演じて、リスキージョブサイトのユーザの電話番号を受け入れるか拒絶するかを決めて下さい。データが妥当だと思ったら、チェックボックスをONにし、そうでない場合はOFFのままでします。不正だと思った場合は理由も書いて下さい。



これが電話番号を表す
正規表現です。これを
演じて下さい！

`/^\$\d{3}-\$\d{3}-\$\d{4}\$/`



(051) 935-2659



(051) 672-0953



051-343-8263

サーフィンも場合によって
は非常に危険な仕事です。
特にサメ寄せとして働く
場合などです！



051-441-9005



051.903.6386



051-612-8527-8724

正規表現になったつもりでの答え



正規表現の役割を演じて、リスキージョブサイトのユーザの電話番号を受け入れるか拒絶するかを決めて下さい。データが妥当だと思ったら、チェックボックスをONにし、そうでない場合はOFFのままとします。不正だと思った場合は理由も書いて下さい。

これが電話番号を表す正規表現です。これを演じて下さい！

/^\\$\d{3}-\d{3}-\d{4}\\$/



カッコは許されません、空白もダメです。

(051) 935-2659



(051)672-0953



051-343-8263

カッコはダメです。



この正規表現ではハイフンでなければなりません。ドットはダメです。

051-441-9005



051.903.6386



あれ？この番号には余分な4桁の数字があります。これは内線番号でしょうか？

051-612-8527-8724



電話番号に何桁かの数字を追加したい場合もあると思うんだけど、例えば4桁の内線番号みたいなもの。こういうパターンもマッチさせるような何か良い方法はあるのかな?

あります。ただし大事なのは正規表現の中でそのようなパターンを省略可能な形で指定することです。

正規表現を `/^\d{3}-\d{3}-\d{4}-\d{4}$` と変更してしまうと4桁の内線番号が最後に付いてなければならないことになってしまいます。そうすると以前マッチしていた通常の電話番号「080-636-4652」はもはやマッチしなくなってしまいます。しかし、正規表現では文字列の一部を省略可能と指定することができます。正規表現には数量子という機能が備わっていて、文字やメタ文字がパターンの中に何回現れてもよいかを指定することができます。実はすでに正規表現の中で数量子の動きを見ています。こんな感じです。

`/^\d{10}$`

ここでは「数字が1行に10回現れる」と言っています。

ここでは、波カッコが数量子であって、先行する数字が何回現れるかを指定しています。よく使われる数量子を他にも見ておきましょう。

{min,max}

波カッコの中に2つの数字があってコンマで区切られている場合、先行する文字またはメタ文字が繰り返して現れる回数の範囲を指定します。ここでは[†]、先行する文字は1行に2回か3回か4回現れてもよいことになります。

[†] 訳注:「ここでは」に相当するものが原著ではありません。恐らく例として{2,4}をあげたつもりなのだと思います。

+

先行する文字またはメタ文字が1回以上現れなければならないことを表します。

?

先行する文字またはメタ文字が0回または1回現れなければならないことを表します。

*

先行する文字またはメタ文字が0回または1回以上現れなければならないことを表します。

数量子は、

メタ文字が

何回現れるかを
指定します。

結論として、電話番号の最後に省略可能な4桁の数字をマッチさせるには、次のようなパターンを用いることになります。

`/^\d{3}-\d{3}-\d{4}(-\d{4})?$/`

数量子が適用されるセクション
はカッコで囲みます。

疑問符は、ハイフンとそれに続く最後の4桁の数字が省略可能であることを表します。



全くその通りです。市外局番は必ず0で始まります。

携帯電話もIP電話も0で始まります。さらに携帯電話に限って考えれば、最初の3桁は、090、080、070のいずれかです。このようなことをするには、文字のクラスが必要です。

文字のクラスを使えば、特定の値の中で文字をマッチさせることができます。文字のクラスで数字の範囲を表すことができるのです。さらにハットを付加することでその中にはないものすべてを表すこともできます。

ある文字のクラスに属する文字やメタ文字をまとめて指定するためには、これをシカクカッコ [] で囲めばよいのです。実際に文字のクラスの例をいくつか示します。

文字のクラスの中では、
^「マッチしない」という
意味です。

[^b-f]

このハットは文字のクラスの中で使われた場合、別の意味を持ちます。「文字列の開始」を表すのではなく、「…以外とマッチする」という意味になります。

これは、b、c、d、e、f以外のすべての文字とマッチします。

[7-9]

これは数字のある範囲でマッチします。7と8と9がマッチします。

[A-D]

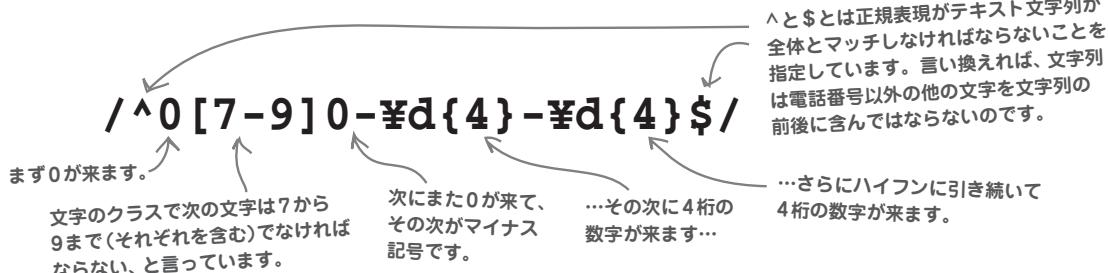
これはAとBとCとDがマッチします。

文字のクラスは、
1つの文字とマッチする
規則の集合です。

携帯電話の番号とマッチする正規表現を書いて下さい。

文字のクラスでパターンを改良する

今度は携帯電話の番号について考えましょう。文字のクラスの助けを借りることで、電話番号の正規表現を適切なものにすることができます。これで不正な数字の組み合わせとはマッチしなくなります。誰かが間違えて090、080、070以外で始めて、エラーメッセージを投げることができます。今度の正規表現は次のような感じです。



素朴な疑問に答えます

Q：つまり文字のクラスでテキスト文字列とマッチする文字の範囲を指定することができるということですか？

A：そうです。正規表現に指定された文字のクラスにより、指定された文字の集合のメンバであれば何でも、テキスト文字列とマッチします。1つの文字ではありません。

例えば、文字のクラス `[aeiou]` とは、小文字の母音のいずれかとマッチします。またクラス `[m-zM-Z]` は、アルファベットの後ろ半分の文字に大文字/小文字を問わずマッチします。

さらに文字のクラス `[0-9]` はメタ文字 `\d` と等価です。実はこのメタ文字は同じことを表現するための略記法なのです。

Q：文字のクラスを指定するときに文字や範囲の間に空白やコンマを置く必要はないのでしょうか？

A：置いてはダメです。もし置いたら、それらの余分な文字 자체がテキスト文字列とマッチさせるための文字の集合の一部と解釈されてしまいます。例えば、文字のクラス

`[m-z, M-Z]`

は、大文字/小文字を問わず `m` から `z`までのアルファベットだけでなく、コンマと空白にもマッチしてしまいます。これは恐らく意図と違うものではないでしょうか？

Q：文字のクラスの文字と2度以上マッチさせたい場合はどうすればよいのでしょうか？例えば1回以上の母音の連続というような場合です。

A：単に数量子を文字のクラスの後ろに付けるだけでよいのです。例えば `/[aeiouAEIOU]+/` という正規表現は、1行に1回以上の母音とマッチします。

Q：数量子というのは、直前にある文字にだけ使えるものだと思っていました。

A：通常はそうなります。ただし、文字のクラスの直後に数量子があれば、数量子はクラスに対して適用されます。さらにもし一連の文字に対して数量子を適用したければ、文字をカッコで括って、これらの文字がグループ化されたことを表すことができます。例えば、正規表現 `/(hello)+/` は、テキスト文字列内で1回以上連続する「hello」とマッチします。

Q：2つの異なるスペルの語とマッチさせたい場合はどうでしょうか？例えば、「ketchup」または「catsup」とマッチさせる場合です。

A：正規表現に縦棒[†](|)を使って選択オプションの集合を表すことができます。例えば、`/ (ketchup|catsup|catchup) /` とすれば、ケチャップの3つの異なるスペルのどれか1つとマッチします。

† 訳注：パイプとも言います。



ドットとか疑問符みたいなものを正規表現に
入れたいときはどうするの？そのまま打ったんじゃあ、
PHPはこういう文字をメタ文字とか数量子と
思っちゃうから正規表現の処理は失敗よね？

正規表現に予約文字を使いたい場合は、エスケープする必要があります。

正規表現の構文では、特別な意味を持つ文字が少數ではあります、あります。このような文字を使ってメタ文字や数量子や文字のクラスを表現するためです。この種の文字としてドット(.)、疑問符(?)、プラス記号(+)、開きおよび閉じシカクカッコ([])、開きおよび閉じカッコ、ハット(^)、ドル記号(\$)、縦棒(|)、円記号(¥)[†]、スラッシュ(/)、アスタリスク(*)があります。

これらの文字を正規表現の中で使いたい場合は、これらの文字が通常表現するメタ文字や数量子ではなく、字面通りの意味であることを明示する必要があります。このためには円記号(¥)を直前に付けて、これらの文字をエスケープしなければなりません。

例えば、電話番号にカッコをマッチさせたいのであれば、以下のように書いてはダメです。

(051)636-4652 X /^(¥d{3})¥d{3}-¥d{4}\$/

これでは単にグループとして
扱われてしまいます。

上記の代わりに、開きカッコと閉じカッコの前に円記号を付けて、これらの文字が本当にカッコを意味するということを示してあげる必要があります。

(051)636-4652 ✓ /^¥(¥d{3}¥)¥d{3}-¥d{4}\$/

これでPHPはこれらの文字が字面通り
カッコであると分かります。

[†] 訳注：コード系によって逆スラッシュ(\)と表示されることがあります、同じものを表します。



以下のパターンにマッチする文字列を示して下さい。

/ ^ [3-6] {4} /

/ ^ ([A-Z] ¥d) {2} \$ /

リスキージョブサイトの電話番号の妥当性検証を拡張して、ユーザが「提出」できる電話番号のフォーマットをもう少し増やすことにしたとします。ただし話を単純化するため市外局番は3桁のもののみとします。以下のすべての書式を受理するような1つの正規表現を書いて下さい。ただし1桁目は0で2桁目にも3桁目にも0は許されません。パターンは数字とカッコと空白とハイフンのみを許可して下さい。

**051-636-4652 051 636-4652
(051)-636-4652 (051) 636-4652**



エクササイズ の答え

文字列の
始まりは…

3から6まで…

この文字のクラスを
4回繰り返します。

`/^ [3-6] {4} /`

3から6までの範囲の4桁の数字いずれかで
始まる文字列であれば何でもマッチします。
例えば「5533」や「3456 is a number.」や
「6533xyz」がマッチします。

文字列の
始まりは…

大文字の
アルファベットで…

次に数字…

2回
繰り返し…

これで
文字列の
終了です。

`/^ ([A-Z]\d) {2} $/`

アルファベット大文字で始まり、次に数字が続き、
その後にもう1回アルファベット大文字と数字が
来て終わりです。例えば「B5C9」や「R2D2」です。

リスキージョブサイトの電話番号の妥当性検証を拡張して、ユーザが「提出」できる電話番号のフォーマットを
もう少し増やすことにしたとします。ただし話を単純化するため市外局番は3桁のもののみとします。以下の
すべての書式を受理するような1つの正規表現を書いて下さい。ただし1桁目は0で2桁目にも3桁目にも0は
許されません。パターンは数字とカッコと空白とハイフンのみを許可して下さい。

051-636-4652 051 636-4652
(051)-636-4652 (051) 636-4652

文字列の
始まりは0で…

1から9まで…
2回繰り返し…

ハイフンまたは
空白があって…

ハイフンが
もう1つあって…

数字が4回
繰り返し…

`/^\$(?0[1-9]{2}\$)?[-\$\s]\$\d{3}-\$\d{4}\$/`

省略可能な開きカッコ
つまり0回か1回現れる
ことができて…

数字があって…

省略可能な閉じ
カッコがあって…

数字が3回繰り返し…

これで終了です。



素朴な疑問に答えます

Q: 正規表現で日本語を含む文字列を扱うことができますか？

A: できます。アルファベットや数字の代わりに単に日本語文字を用いればよいだけです。文字のクラスも英数字と同様に定義することができます。例えば[あいうえお]または[あ-お]というクラスは、「あ」、「い」、「う」、「え」、「お」のいずれか、を表します。

Q: 正規表現で日本語を扱う上で何か注意すべきことはありますか？

A: あります。次のページから説明する関数をマルチバイト対応の関数で置き換えて使う必要があります。`preg_match()` および `preg_replace()` に相当するマルチバイト対応の関数は、それぞれ `mb_ereg_match()` および `mb_ereg_replace()` です。該当する章で、改めて対応するマルチバイト関数を紹介します。ただし、本書で扱う文字列には日本語が含まれません。詳しい説明は本書の範囲を超えますので、省略いたします。

preg_match()[†]でパターンをチェックする

今までいろいろなパターンを作っていましたが、今のところただ作っただけです。これらのパターンを PHP 関数 `preg_match()` で使うことができます。この関数は正に今まで作ってきたような正規表現パターンとテキスト文字列をパラメタとして取ります。正規表現にマッチしなければ `false` を返し、マッチすれば `true` を返します。

[†] 訳注：日本語を含む文字列を扱う場合は、`mb_ereg_match()` 関数を用います。

`preg_match($regex, $my_string)`

ここに正規表現が来ます。関数は文字列を
パラメタとします。つまり正規表現は單一
引用符で囲まれている必要があります。

マッチするかどうかチェック
する文字列がここに来ます。

以下に `preg_match()` 関数の動作例を示します。3桁の数字、ハイフン、2桁の数字、ハイフン、4桁の数字というパターンのテキスト文字列を探す正規表現を使っていきます。

正規表現を `preg_match()` に渡す場合、引用符
で囲まなければなりません。

```
preg_match('/^$\d{3}-\d{2}-\d{4}$/', '555-02-9983')
```

整数を返します。文字列がパターンに
マッチした場合 1 を返します。そうで
なければ 0 を返します。

実際のパターンをこのように関数の
パラメタに直接書いても構いません。
しかし通常は変数にぶち込んで
おいた方がよいでしょう。

この文字列を正規表現と
マッチさせますので、
関数は 1 を返すでしょう。

`preg_match()` 関数のおかげで、非常に洗練された妥当性検証機能を PHP スクリプトに組み込むことができます。関数から返ってきた値を `if` 文で囲んであげれば良いのです。

`preg_match()` 関数は条件部に
ネストしていますので、関数の
結果によりどちらのコードを
実行するか決まります。

```
if (preg_match('/^\d{3}-\d{2}-\d{4}$/', '555-02-9983')) {
    echo 'このソーシャルセキュリティナンバーは妥当な形式です。';
} else {
    echo 'このソーシャルセキュリティナンバーは不当な形式です！';
}
```

マッチに成功すれば、`preg_match()` は `true` を返します。これにより PHP は条件部が `true` であると認識します。つまりこのコードが走ります。

マッチに成功しなければ、`preg_match()` は `false` を返します。これにより条件部は `false` と評価されます。つまりこのコードが走ります。



以下はリスクジョブサイトのPHPスクリプトの一部分で、登録用フォームのデータをチェックしています。電話番号フィールドに入力されたテキストをempty()ではなくpreg_match()を使って検証するように書き換えて下さい。preg_match()関数のところで先ほど作った正規表現を使います。

```
if (empty($phone)) {  
    // $phone is blank $phone が空  
    echo '<p class="error">エラー：電話番号を入力して下さい。</p>' ;  
    $output_form = 'yes' ;  
}
```


**エクササイズ
の答え**

以下はリスクジョブサイトのPHPスクリプトの一部分で、登録用フォームのデータをチェックしています。電話番号フィールドに入力されたテキストをempty()ではなくpreg_match()を使って検証するように書き換えて下さい。preg_match()関数のところで先ほど作った正規表現を使います。

```
if (empty($phone)) {
    // $phone is blank
    echo '<p class="error">エラー：電話番号を入力して下さい。</p>';
    $output_form = 'yes';
}
```

empty()の代わりにpreg_match()を使って電話番号を検証します。先頭に否定演算子(!)を付けます。なぜなら入力されたデータがパターンにマッチしない場合エラーを表示したいからです。

```
if (!preg_match('^\(?0[1-9]{2}\)?[-\s]\d{3}-\d{4}$', $phone)) {
```

// \$phone is not valid \$phone が不正

```
echo '<p class="error">エラー：電話番号の形式が妥当ではありません。</p>';
```

```
$output_form = 'yes' ;
```

```
}
```

\$output_formを「yes」に
設定するのは以前のままで。

先ほど作った電話番号の正規表現です。

echo文を少々変える必要があります。データが入力されたかどうかのチェックだけでなく、標準的な電話番号のパターンにマッチしているかどうかもチェックしているからです。

エラーが出たので、
ちゃんと電話番号を入れた。
これで忍者の仕事に
ありつける！

姓：横山
名：憲治
メール：yken@sim-u-duck.com
電話番号：(555)636 4652
希望職種：忍者(Ninja)





リスクジョブサイトの登録用スクリプトで電話番号が妥当かどうかチェックする。

registration.phpスクリプトをダウンロードサイト<http://www.oreilly.co.jp/books/9784873114446/>からダウンロードします。リスクジョブサイトのスタイルシート(style.css)とイメージファイル(riskyjobs_title.jpgとriskyjobs_fireman.png)も一緒に持ってきて下さい。次にregistration.phpスクリプトを修正します。preg_match()を使って電話番号が正規表現で規定されるものとマッチするか検証するようにして下さい。エラーメッセージもちゃんと調整して、ユーザは電話番号が単に空ではなく、不当であると分かるようにして下さい。

変更したスクリプトをWebサーバにアップロードしたら、Webブラウザで開きます。いくつかの電話番号を各種のフォーマットで入力し、スクリプトがどのようにエラーを捕まえるかを確認して下さい。

空でない電話番号ならOKという
わけではありません！

これで間違って変な電話番号を
打ち込むこともなくなった。
仕事のチャンスを失う
こともない！

Risky Jobs

危険！夢の仕事は危険と隣り合わせです。
勇気を出して探してみませんか？

リスクジョブ：危険な仕事へ登録

リスクジョブによる登録および履歴書の投稿

姓：

名：

メール：

電話番号： ←

希望職種：

ここに履歴書を貼り付けてください：
消防の剣士を何年も演じてきました。ほうきを使う魔女
と対決する役を演じたこともあります。このときもNC
なしですぐに撮影終了しました。真剣も含めどのよう
な難題の劇でも抜きとができます。ファンシング一日

電話番号を間違って入力
すると、スクリプトはエラー
メッセージを表示します。
この場合で言うとハイフンの
代わりにドットが使われて
いるためです。

ダウンロードして下さい！

リスクジョブサイトのアプリケーションの全ソースコードはWebサイトからダウンロードできます。

<http://www.oreilly.co.jp/books/9784873114446/>



う～ん、正規表現が電話番号の複数パターンにマッチするとしたら、そのテキストは全部異なるフォーマットでデータベースに入るっていうことかい？ダメじゃん。全部を標準化する必要があると思うんだけど。

異なるすべてのフォーマットでデータの入力を許可するからと言って、それらすべてのフォーマットでデータを格納したいというわけではありません。

めでたいことに、他にも正規表現関数があって、リスキージョブサイトのユーザが「提出」した妥当な電話番号をパラメタに取って、全部で4つだったすべてのパターンを1つの一貫したパターンに従うように変換してくれます。

`preg_replace()` 関数[†]は、正規表現を使ってパターンマッチを実行する上で `preg_match()` 関数よりも1ステップ先に連れて行ってくれます。与えられたパターンが与えられたテキスト文字列にマッチするかどうかを判定することに加えて、マッチしたテキストの文字列を置き換えるためのパターンを提供してくれます。これはすでに使ったことのある `str_replace()` 関数と非常によく似ています。ただし文字列の代わりに正規表現を使ってマッチさせる点が異なります。

[†] 訳注：日本語文字を含む文字列を扱う場合は、`mb_ereg_replace()` 関数を使います。

`preg_replace($pattern, $replacement, $my_string)`

ここに不要な(置き換え前の)文字を書きます。

不要な文字が見つかったら、これに置き換えます。

ここが検索・置き換えの対象となる文字列です。

以下に `preg_replace()` 関数の動作例を示します。

```
$new_year = preg_replace('/200[0-9] /', '2010', 'The year is 2009.');
```

`preg_replace()` 関数の結果として、検索・置き換えが終わって書き換えられたテキスト文字列を、`$new_year` に突っ込みます。

この正規表現が `preg_replace()` に2000から2009までにマッチするものを探すように教えます。

マッチするものが見つかったら、2010に置き換えられます。

2000から2009までの年が文字列から見つかったらすべて2010に置き換えられます。



リスキージョブサイトのフォームに入力された電話番号データを標準化します。次の各電話番号データを以下のデータベーステーブルのphoneカラムに標準化したデータを書き込んで下さい。
フォーマットとして用いるのは、ユーザの電話番号を表現するために必要となるデータなるべく小さくして格納できるようにして下さい。

(051) 935-2659
(051)672-0953
051-343-8263
051-441-9005
051.903.6386
051-612-8527-8724



phone
...



エクササイズ
の答え

リスキージョブサイトのフォームに入力された電話番号データを標準化します。次の各電話番号データを以下のデータベーステーブルのphoneカラムに標準化したデータを書き込んで下さい。
フォーマットとして用いるのは、ユーザの電話番号を表現するために必要となるデータをなるべく小さくして格納できるようにして下さい。

(051) 935-2659
(051)672-0953
051-343-8263
051-441-9005
051.903.6386
051-612-8527-8724

リスキージョブ：危険な仕事へ登録

Risky Jobs

危険！夢の仕事は危険と隣り合われます。
勇気を出して探してみませんか？

リスキージョブによる登録および履歴書の投稿

姓：
名：
メール：
電話番号：
希望職種：

ここに履歴書を貼り付けてください：

送信

phone
...
0519352659
0516720953
0513438263
0514419005
0519036386
0516128527

電話番号を突っ込む最も
簡潔な方法は、数字以外
のすべてをはぎ取って
しまうことです。

電話番号データを標準化する

現時点ではリスキーなサイトは以下の正規表現を使って電話番号を検証しています。この番号はユーザーが登録用フォームを通して「提出」してきたものです。

```
/^\(?0[1-9]{2}\)?[-\s]\d{3}-\d{4}\$/
```

この正規表現でマッチする電話番号は以下の4つのパターンに集約[†]されます。



この種のフォーマットは人間が解釈する分には簡単ですが、SQL問い合わせ文で望みの形式にソートするのは容易ではありません。このようなカッコがあると、例えば電話番号を市外局番でグループ分けするといったことが台無しになってしまいます。このようなグループ分けはリスキーなサイトでは重要な意味を持つかも知れないので、サイトのユーザーは地理的にどのあたりからどのくらい応募してきているのかといった解析ができるためです。

この種の問い合わせを可能にするためには、電話番号を `preg_replace()` を使って1つのフォーマットに標準化し、それからデータベースにINSERTする必要があります。このための最良の戦略は数字以外のすべての文字を捨ててしまうことです。そうすれば、単純に10桁の数字をぶち込めば良いだけです。他の文字はありません。テーブルにぶち込みたい数字はこんな感じになります。

#####

あとは4つの文字を見つけて置き換えるだけです。この場合は開きカッコ、閉じカッコ、空白、そしてハイフンを見つけたら消してしまいます。さらにこれらの文字が文字列のどこにあっても見つけたいのです。そこで最初を表すハット(^\)や最後を表すドル記号(\\$)は必要ありません。この文字グループのどれでもよいわけですから、文字のクラスを用います。検索の順番はどうでもよいので、たとえば正規表現は以下のようになります。

/ [\$(\\$\\$)-\s] /

↑ 開きカッコ
↑ 閉じカッコ
↑ ハイフン
↑ 空白

データを標準化すると
SQL問い合わせ文で
よりよい結果が得られます。

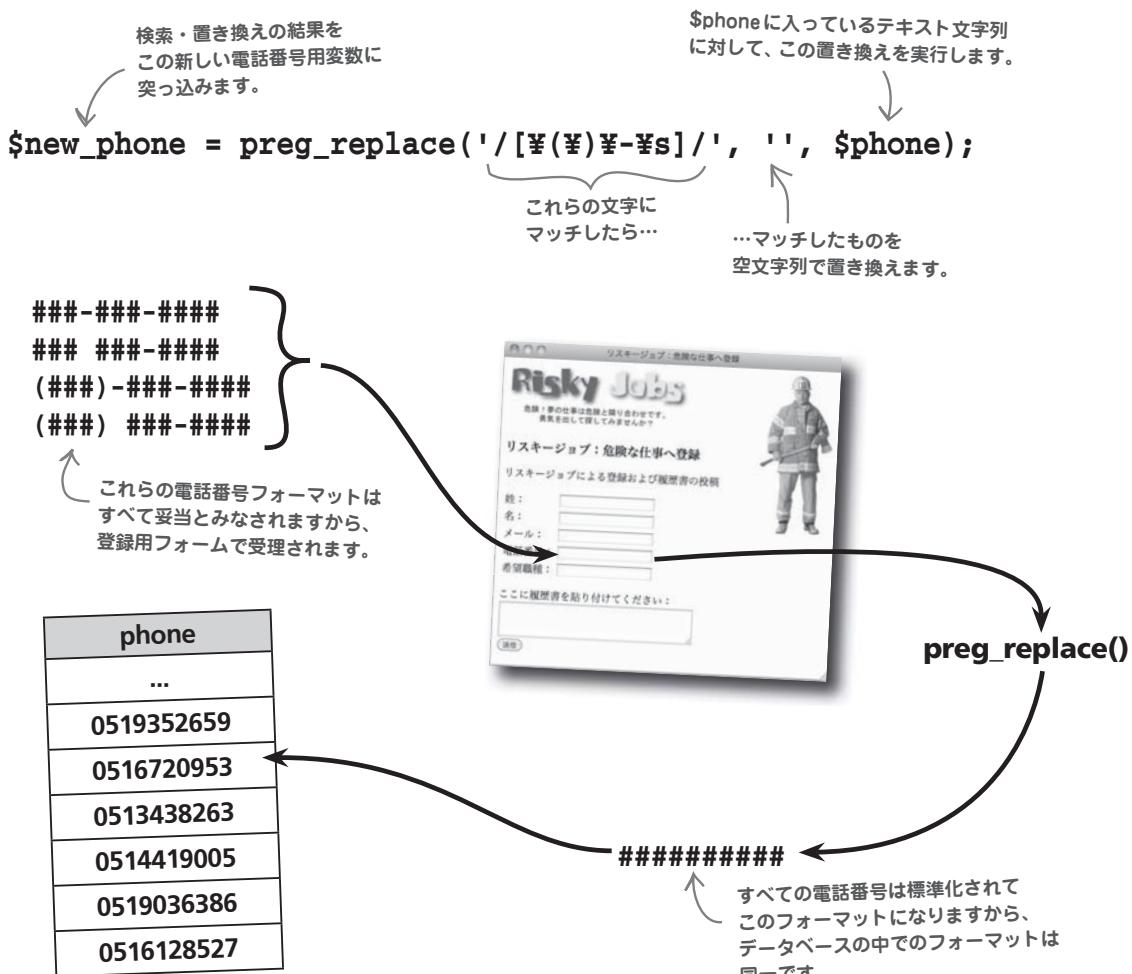
[†] 訳注：厳密に言うと、この説明は間違っています。マッチする文字列は他にもあります。特に問題と思えるものは、例えば「###-###-####」や「###-###-#」といったカッコの対応の取れていない文字列もマッチしてしまうことです。この正規表現は開きカッコと閉じカッコが対応することを表現していないことが原因です。この問題を解消したければ、正規表現を：

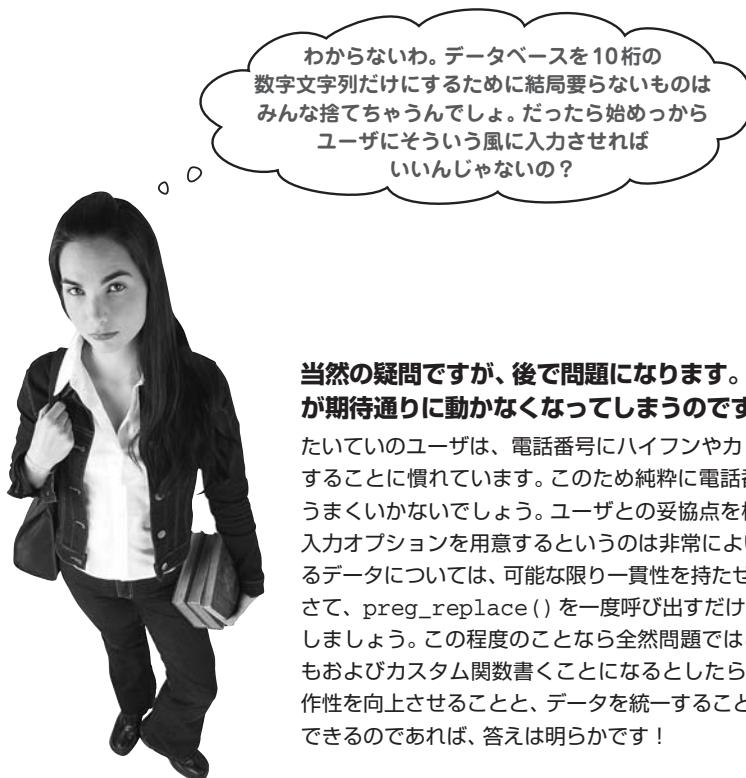
```
/^(?(\(?0[1-9]{2}\))|(\(?[1-9]{2}\)))[-\s]\d{3}-\d{4}\$/
```

と変更します。こちらの正規表現ではカッコのある場合とない場合とに分けて|(または)でつないでいます。

不要な文字を捨てる

パターンを使って不要な文字を見つけることができるようになりましたから、このパターンを電話番号に適用してきれいにすることにします。それからデータベースにぶっ込めばいいわけです。ではどうすればよいのでしょうか？ここでpreg_replace()関数の登場です。工夫しなければならないのは、不要な文字を置き換えるわけではないということです。不要な文字はなくしてしまいたいのです。このような場合、preg_replace()に、置き換え後の値として、単に空文字列を渡せばよいのです。以下に、電話番号として不要な文字を見つけ、それを空文字列で置き換える例を示します。これで効率的に不要な文字を捨てることができます。





当然の疑問ですが、後で問題になります。電話番号での問い合わせ文が期待通りに動かなくなってしまうのです。

たいていのユーザは、電話番号にハイフンやカッコや空白を組み合わせて入力することに慣れています。このため純粋に電話番号だけでの入力を強制してもうまくいかないでしょう。ユーザとの妥協点を模索して、十分に柔軟性のある入力オプションを用意するというのは非常によいことです。同時に、格納されるデータについては、可能な限り一貫性を持たせることが重要です。
さて、`preg_replace()`を一度呼び出すだけで解決できる問題について議論しましょう。この程度のことなら全然問題ではありません。もし仮に何十行にもおよびカスタム関数書くことになるとしたら、話は全然別です。ユーザの操作性を向上させることと、データを統一することを、たった1行のコードで両立できるのであれば、答えは明らかです！



試運転

登録用スクリプトの電話番号をきれいにする

registration.phpスクリプトを修正して、電話番号をきれいにします。以下のコードをスクリプトに追加して下さい。リスキージョブサイトに登録してくれたユーザに「ありがとう」のメッセージを表示する行の直後です。

```
$pattern = '/[¥(¥)¥-¥s]/';
$replacement = '';
$new_phone = preg_replace($pattern, $replacement, $phone);
echo '電話番号：' . $new_phone . ' を登録しました。</p>';
```

できあがったらスクリプトをWebサーバにアップロードし、Webブラウザで開きます。フォームを埋めてみて下さい。電話番号を入力するときに余分な文字を忘れずに入れて下さい。例えば(077) 827-7000と言った具合です。フォームを「提出」して結果を確認しましょう。

リスクジョブ：危険な仕事へ登録

Risky Jobs

危険！夢の仕事は危険と隣り合わせです。
勇気を出して探してみませんか？

リスクジョブ：危険な仕事へ登録

高松 静さん、リスクジョブにご登録ありがとうございます！

電話番号：0718277000を登録しました。

電話番号が
囁み碎かれて番号
だけになってしまいます。
余分な文字は
ありません！

電話番号としていくつかの種類のフォーマットを試して下さい。例えば次のようなものです。077.827.7000、(077)-827-7000、077 827-7000。正規表現とpreg_replace()によりどのように余分な文字が捨てられるか、よく見て下さい。

電話番号のパターンはまだあります

その通りです。今まで、ほとんどのページで市外局番が3桁の固定電話に限って話を進めてきました。しかし、日本の電話番号には他にも種類があります。市外局番については、2桁、3桁、4桁、5桁の場合があります。それ以外に携帯電話やIP電話もあります。これらのパターンをすべて表すことはできるでしょうか？実は答えは、YESなのですが、正規表現が非常に複雑になります。(個々の場合を今まで見てきたような正規表現で表し、それらすべてを縦棒(|)でつなげばよいのです。)

現実的には以下のいずれかの方法を探るのがよいでしょう。

- ① それぞれのパターンに分けて正規表現を作り、`preg_match()`を個別に呼び出す。
- ② すこし雑な正規表現を作り、受理された式の中をさらに別の方法で検査する。

①の方法は、直感的に分かりやすいものと思います。市外局番が2桁の場合の正規表現、3桁の場合の正規表現、…5桁の場合の正規表現をすべて列挙するわけです。次のようにになります。

```
/^(?0[1-9]?)?[-\$s]\$d{4}-\$d{4}$/  
/^(?0[1-9]{2})?[-\$s]\$d{3}-\$d{4}$/  
/^(?0[1-9]{3})?[-\$s]\$d{2}-\$d{4}$/  
/^(?0[1-9]{4})?[-\$s]\$d-\$d{4}$/
```

さらに市外局番が2桁の場合は、2桁目は3、4、または6に限られるということを表現したければ、最初の正規表現を

```
/^(?0[346])?[-\$s]\$d{4}-\$d{4}$/
```

で置き換えることもできます。

携帯電話およびIP電話については比較的簡単に、まとめて1つの正規表現とすることができます。こんな感じです。

```
/^(?0[5789]0)?[-\$s]\$d{4}-\$d{4}$/
```

検証スクリプトを書き換える

あとは、これらの正規表現各々について `preg_match()` 関数を使って `if-else` 文でチェックし、1つでも `true` が返ってくれば、正しい電話番号とみなせばよいでしょう。この方法に従った場合 517 ページのスクリプトは次のようになります。

```
if (!preg_match( '/^\(?0[346]\)?[-\d]{4}-\d{4}$/, $phone) &&
    !preg_match( '/^\(?0[1-9]\){2}\)?[-\d]{3}-\d{4}$/, $phone) &&
    !preg_match( '/^\(?0[1-9]\){3}\)?[-\d]{2}-\d{4}$/, $phone) &&
    !preg_match( '/^\(?0[1-9]\){4}\)?[-\d]{4}$/, $phone) &&
    !preg_match( '/^\(?0[5789]0\)?[-\d]{4}-\d{4}$/, $phone) ) {
    // $phone is not valid $phone が不正
    echo '<p class= "error" >エラー：電話番号の形式が妥当ではありません。</p>' ;
    $output_form = 'yes' ;
}
```

②の方法というのは、固定電話の市外局番は 2～5 桁、市内局番は 1～4 桁、それ以降が 4 桁という具合に雑な正規表現を定義するのです。これだと正規表現は大きな式にもたくさんの方程式にならない、という点でちょっとイケてます。こんな感じになります。

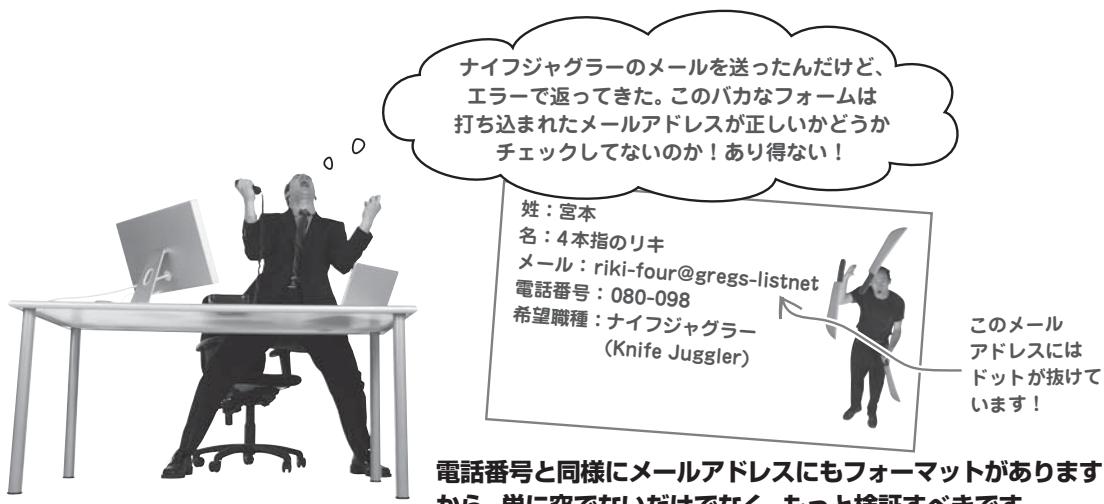
```
/^\(?0[1-9]\){1,4}\)?[-\d]{1,4}-\d{4}$/
```

このたった 1 つの式ですべての固定電話番号とマッチさせることができます。しかし、残念なことに 01234-5678-9012 のような番号にもマッチしてしまいます。この正規表現では、市外局番が何桁であるかと、それに対応して市内局番が何桁になるかを数えていないためです。雑な正規表現と言った意味はこれです。そこで次に、この正規表現にマッチした「候補」の中で、実際に数字の数をカウントして、ちょうど 10 個のものだけを受理するような別の検査を追加すれば、望みの結果が得られます。正規表現は小さいままで、同じ効果が得られたことになります。

```
if (!preg_match( '/^\(?0[1-9]\){1,4}\)?[-\d]{1,4}-\d{4}$/, $phone) &&
    !preg_match( '/^\(?0[5789]0\)?[-\d]{4}-\d{4}$/, $phone) ) {
    // $phone is not valid $phone が不正
    echo '<p class= "error" >エラー：電話番号の形式が妥当ではありません。</p>' ;
    $output_form = 'yes' ;
}
if(preg_match( '/^\(?0[1-9]\){1,4}\)?[-\d]{1,4}-\d{4}$/, $phone) &&
    count_digits($phone) != 10) {
    echo '<p class= "error" >エラー：電話番号の形式が妥当ではありません。</p>' ;
    $output_form = 'yes' ;
}
```

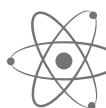
①の方法、②の方法ともにメリット、デメリットがあります。どちらが良いかは好みの問題です。ただ 1 つ、言えることは正規表現の中に多くの規則を取り入れようとすればだけ正規表現が大きくなり読みにくくなるということです。

ここで `count_digits()` はカスタム関数で、パラメタとして与えられた文字列の中にある数字の数を返すものとします。



電話番号と同様にメールアドレスにもフォーマットがありますから、単に空でないだけでなく、もっと検証すべきです。

電話番号の妥当性を今まで検証してきたように、まず妥当なメールアドレスが従わなければならない規則を決めておく必要があります。次にこの規則を正規表現として形式化し、それをPHPスクリプトで実現します。ではまずメールアドレスとは正確にはどうあるべきか見ていきます。



脳力発揮

十分汎用的で正しいメールアドレスにマッチする正規表現を思いつきましたか？以下に書いて下さい。

aviator.howard@bannerocity.com

cube_lovers@youcube.ca

rocky@i-rock.biz

メールアドレスとマッチさせるのは難しい

残念ながら正規表現で「正しいメールアドレス」かどうかを検査することは不可能です。実はこれは電話番号についてもあてはまるました。正規表現で正しい形式と判断した電話番号が、現在使われている電話番号であるという保障はどこにもありません。また、使われているものであったとしても、入力した本人のものであるという保障もありません。正規表現ではその文字の並びが持っている意味は分からぬのです。メールアドレスについても同様です。実際、正しいメールアドレスを入手することは極めて難しい問題です。

現実的なWebアプリケーションは次のような方法で、メールアドレスを取得しています。ただし本章の内容（正規表現）から逸脱しますのでここでは、その指針についての説明に留めます。

1

空メール登録

1. ユーザは特定のメールアドレス（登録申し込み用のアドレス）に対して、空メールを送信する。
2. 空メールを受け取ったら、登録用サイトのシステムからメールの送信元に対して、登録用のサブアドレスの付きのメールを送信する。
(登録用のサイトには、送り主のアドレスをスクリプト上であらかじめ登録しておく。)
3. ユーザは返信されたメールに書かれた登録用サイトにアクセスし、必要事項を記入することでユーザ登録を完了する。
(登録用サイトは、そのメールアドレスを持つ人のものであり、空メール受信時点でメールアドレスを登録できるため、サイトアクセス時には正しいアドレスであることが保証されており、ユーザは入力の必要がない。)

2

2段階登録

1. ユーザはまず仮登録用のサイトにアクセスしメールアドレスを入力する。
(このとき入力ミスのリスクを軽減するため、メールアドレスは2回入力させる。)
2. サイト側でメールアドレスの2回入力をチェックし、正しければ、そのアドレス宛に本登録用のサイトアドレス付きのメールを送信する。
3. メールを受け取ったユーザは、メールに書かれた本登録用サイトにアクセスし、必要事項を記入することでユーザ登録を完了する。
(本登録用サイトは、そのメールアドレスを持つ人のものであり、メールアドレスは仮登録時にすでに入力されているため本登録時には入力の必要がない。)

以上いずれの方法も、登録手続きの中に実際にメールのやり取りが含まれているため、そのアドレスが正しいアドレスであることの確認ができるだけでなく、本人であるという確認もできます。ただし、実際にこのアプローチに則ってサイトを構築することは本書では行いません。

PHPを使ってドメインをチェックする

PHPには`checkdnsrr()`関数があって、ドメインが妥当かどうかをチェックしてくれます。この方法を使えばメールアドレスのドメイン名(@よりも先の部分)が本当に存在するかどうかだけなら確認できます。この関数はDNSレコードを実際にチェックして、ドメインが本当に登録されているかどうか見てくるのです。ですから例えば、正規表現では`lasdjlkalkjaf.com`が妥当であると判定するでしょうが、`checkdnsrr()`は、もう1歩先を見ます。そしてこのドメインが登録されていないと教えてくれます。この場合、登録用フォームに入力された`sdfhfdskl@lasdjlkdfsalkjaf.com`をエラーとしてはじくことになるでしょう。(ただしドメイン名が存在しても、メールアドレスが正しく、かつ本人のものであるかどうかは確認できません。)

`checkdnsrr()`の構文は極めて単純です。

本物のドメインであれば1が返ってきます。そうでなければ0です。

`checkdnsrr()`はドメイン名の文字列をパラメタに取ります。
@の後ろ全部です。

`checkdnsrr('headfirstlabs.com')`



PHPをWindowsサーバ上で走らせている場合、このコマンドはうまく動かないはずです。

要注意

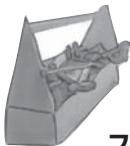
代わりに以下のコードを使って下さい。

```
function win_checkdnsrr($domain,$recType='') {
    if (!empty($domain)) {
        if ($recType=='') $recType="MX";
        exec("nslookup -type=$recType $domain",$output);
        foreach($output as $line) {
            if (preg_match("/^$domain/", $line)) {
                return true;
            }
        }
        return false;
    }
    return false;
}
```

この問題はWebサーバがWindowsの場合に限られます。Windows PCでWebサイトを構築してはいても、実際にはunix/Linuxサーバにポストしているだけであれば、問題は起こりません。

このexec()コマンドはサーバ上で動作する外部のプログラムを呼び出して、ドメインのチェックをします。

実験をしてみたければ、`echo $line`を
foreachの直後においてみて下さい。
以下のようなものが表示されるはずです。
`Server: 68.87.64.146Address:
68.87.64.146#53Non-authoritative answer:
oreilly.com mail exchanger = 20 smtp1.
oreilly.com.`



PHP & MySQL 道具箱

テキストにあるパターンを探すことによって、Web フォームにユーザが入力したデータの妥当性を手軽に検証することができます。以下の PHP の技法は、正規表現を使ってデータを検証するために使います。

正規表現

テキスト文字列のパターンにマッチさせるために使う規則です。PHPには文字列が特定のパターンかどうかをチェックする関数や、文字列内のテキストのパターンを検索して置き換える関数が用意されています。

preg_match()

この PHP 関数で文字列テキストをチェックし、正規表現にマッチするかどうかを判定します。関数はマッチするものがあれば `true` を返し、なければ `false` を返します。日本語を扱う場合は `mb_ereg_match()` を使います。

preg_replace()

これらの PHP 関数を使って、文字列内の部分文字列を正規表現に基づいて置き換えるときに使います。関数は検索・置き換えを実行しますが、検索には正規表現を使い、置き換えにはパラメタで指定した文字列を使います。日本語を扱う場合は `mb_ereg_replace()` を使います。

`\d`, `\w`, `\s`, `^`, `$`,

...

正規表現はメタ文字を使って作ることもできます。メタ文字を使って3桁の数字(`\d\d\d`)や空白類文字(`\w`)といったテキストを表現することができます。

文字のクラス

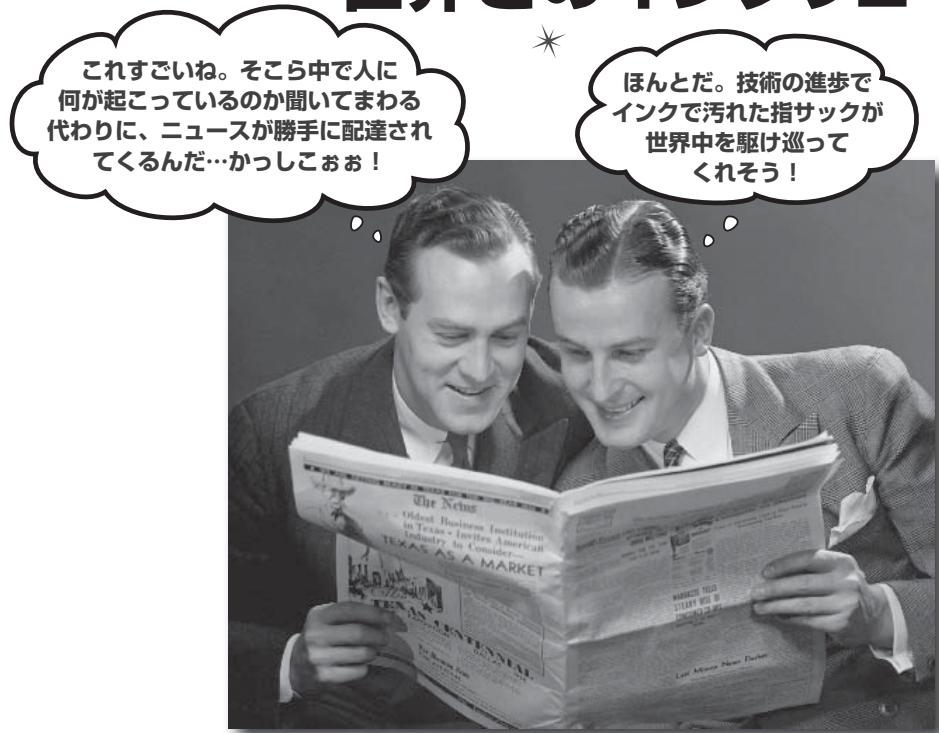
1つの文字が正規表現の中でマッチする規則の集合です。例えば、`[A-D]`には文字、A、B、C、D がマッチします。

checkdnsrr()

この PHP 関数はドメイン名が実際に存在するかどうかをチェックします。`email` アドレスの妥当性を手軽に検証することができます。`email` のドメイン部分が本物かどうかを確認することができるからです。

11章 シンジケーションとWebサービス

世界とのインターフェース



外の世界というものはものすごく広いので、Webアプリケーションとしては無視するわけにはいきません。さらに重要なことは、世界から自分のWebアプリケーションを無視されないようにしなくてはいけないということです。良い方法があります。世界を自分のWebアプリケーションに向けるには、データをシンジケーション[†]で使えるようにすればよいのです。シンジケーションとはユーザがWebサイトのコンテンツを購読できるようにすることです。今までのようにユーザがWebサイトを訪問して直接新しい情報を探さなければならないではありません。そればかりか、自分のアプリケーションがWebサービスを通して他のアプリケーションとインターフェースを取って、他のデータによるメリットを使って、もっとリッチな操作性を提供することができるのであります。

[†] 訳注：英語ではsyndicationです。syndicate(シンジケート、組織)と同一語源の単語ですが、インターネットの世界ではフィードを使ってWebコンテンツを配信することを指します。適切な日本語がないため、そのままカタカナとしました。

オーウェンはファングについての言葉が欲しいのです

どんなWebサイトでも直面する大きな問題は、リピータを増やすことです。訪問してくれる人を捕まえるだけでも大変ですが、リピータになってもらうには別の苦労があります。極めて魅力的なコンテンツを持っているようなサイトですら、人間の探知機から消えてしまうことがあります。単にWebサイトを定期的に訪問するという行為を忘れてしまうからです。このことを踏まえて、オーウェンは、UFOによる誘拐レポートを閲覧するために別の手段を提供しようと考えています。レポートを人々に「プッシュ」で送るのです。彼らに定期的にサイトを訪問してもらうのではありません。

UFOによる誘拐
レポートをアップしたんだけど、相変わらずファングは見つからない！

ファングは何回か目撃されているようですが、この情報に基づいてもオーウェンはファングのいどころを知ることができませんでした。

誘拐レポートのフォームは非常によく機能していますが、オーウェンはサイトがもっともっと人目をひく必要があると思っています。

あの後オーウェンは新しくユーザが「提出」したUFOによる誘拐レポートを閲覧するためのメインページを作りました。

オーウェンは、サイトがもっと人目をひけばファングを見つける可能性も高まると期待しています。

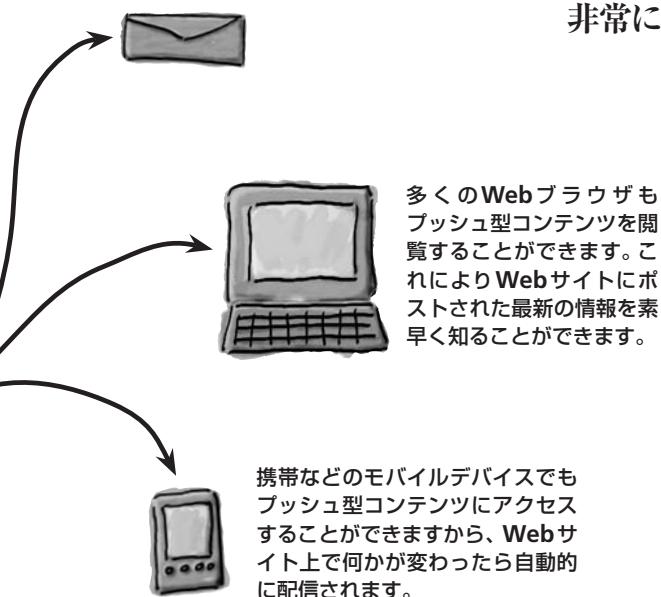
最近更新された誘拐レポート：		
2009-08-10 : 市川 徹 誘拐されていた時間： 3時間	宇宙人の様子： 満月くらいの大きさの船でした。	ファングはいましたか？ no
2009-07-11 : 斎藤 友世 誘拐されていた時間： 45分	宇宙人の様子： 頭がでかくて手足はひょろひょろでした。	ファングはいましたか？ yes
2009-07-05 : 白田 敏朗 誘拐されていた時間： 2時間	宇宙人の様子： 空にとても明るいUFOが少なくとも2~3機いたようです。	ファングはいましたか？ yes
2009-06-21 : 深沢 麻子 誘拐されていた時間： ほぼ1週間	宇宙人の様子： 不格好で変な奴で、リズム感ゼロ。	ファングはいましたか？ no
2009-05-11 : 清水 裕子 誘拐されていた時間： 1日	宇宙人の様子： 緑色で触覚が6つありました。	ファングはいましたか？ yes

UFOによる誘拐データを人々にプッシュする

UFOによる誘拐のコンテンツをユーザーにプッシュすることによって、オーウェンは仮想的なチームを効率的に作ることができます。彼らに誘拐レポートの監視を手伝ってもらうのです。事件により多くの人が関われば、ファンが目撃を認識する可能性は高くなり、ファンのいどろにたどり着く期待も高まります。

Webコンテンツをユーザーにプッシュすることは、Webサイトをより多くの人の目にさらす可能性が高まるという意味で非常に効果的な方法です。

メールによっては、コンテンツのプッシュをサポートしています。Webサイトにアップデートがあると、メールメッセージを受け取るのと同様にメッセージを受け取ることができます。



オーウエンはコンテンツをユーザーにプッシュするにはどうすればよいのかあまりよく知りません。でも本当にこのようにしたいとは思っています。



オーウエンの仮想チームメンバーです。UFOによる誘拐コンテンツを見てくれるので、オーウエンがファンを見つける可能性が高まると期待しています。

RSSはWebコンテンツを人々にプッシュしてくれます

HTMLコンテンツをWebにポストするというアイデアの背後にあるのは、Webサイトを訪問する人々に見てもらうということに尽きます。でももし、ユーザが訪ねて来なくてもWebコンテンツを受け取ってもらうようにしたいとしたらどうすればよいでしょう？このようなことはRSSを使うことで解決できます。RSSというのはデータのフォーマットのことで、これを使うとユーザがサイトを実際に訪問することなくWebコンテンツを見つけることができるようになります。

RSSはWebにおけるデジタルビデオデッキのようなものです。ビデオデッキだとテレビ番組を「購読」することができます。飛んでくる番組は何でも自動的に録画できます。ビデオデッキのおかげで、好きな番組が勝手に録画することができます。何故わざわざチャネルを回して番組を探す必要があるでしょうか？実際にはRSSは何も記録してくれませんが、Webコンテンツを自分から探す代わりに、勝手に届けてくれるという意味でビデオデッキに似ています。

RSSのフィード(エサ)をUFOによる誘拐のデータに対して作ることによって、新しいレポートがポストされる毎にユーザに知らせたい、とオーウェンは考えています。これにより人々の興味を維持できるので、結果としてより多くの人をデータに結びつけることができます。賢いのは、WebページとRSSフィードの両方を同じデータベースで動かすことができます。

UFOに誘拐された！?

UFOに誘拐された！ニュースフィード

市川 勝・謹月くらいの大きさの船でした。... 8月 10 日 [詳細]

山の頂上まで車で登れ自分で行きそこから乗としました。北丸直樹さん。

謙月 携帯電話で話がかかるくて手足はひょろひょろでした。... 7月 27 日 [詳細]

私たちは人に見付かりました。北丸直樹さん。

日田 駆駕駕・空にとても明るいUFOが少なくとも2~3機いたようです。... 7月 9 日 [詳細]

大陸で飛行中の飛行機をガラガラシタクシードで見ていました。北丸直樹さん。

深井 麻子・不思議で変な奴で、リズム感ゼロ。... 6月 21 日 [詳細]

本当に不思議で頭を痛めようとした。北丸直樹さん。

市川 勝・緑色で触覚がつきました。... 6月 11 日 [詳細]

色々と想像してそれを現実にしました。北丸直樹さん。

小川 淳・大きくて青らさを出した商品のディスクみた以為て、また労働... 2008年7月 13日 [詳細]

私もこのままお出で下さい。何を差し出すかわからなくて困りました。北丸直樹さん。

北田 稔・金剛製のババみたいだった。何をジェットエンジンのようなな... 2008年6月 14日 [詳細]

ちょっとバカで面白かった。北丸直樹さん。

水島 雅子・面白く気分がなさうで、これを用ひました。でもあ... 2008年1月 29日 [詳細]

私はお酒を手に見ながら、色々と語っていました。北丸直樹さん。

Webページはデータベースのデータから動的に作られてはいますが、新しいデータがポストされたら自分で再訪しなければなりません。

RSSは、新しいコンテンツが提供されると、Webデータをユーザに自動的に配信することで、閲覧できるようにしてくれるものです。RSSでデータの特定のグループを閲覧することをRSSフィードとかニュースフィードと言います。ユーザはフィードを購読してWebサイトにポストされる毎に新しいコンテンツを受け取ることができます。サイトを訪れてタブを追う必要はないのです。RSSフィードを閲覧するのに必要なものはRSSニュースリーダです。と言っても単にフィードのURLにニュースリーダを渡すだけです。後は全部向こうがやってくれます。

HTMLは
閲覧用でRSSは
シンジケーション用
です。

ニュースリーダで
ニュースフィードを
購読することができます。
ここにはWebサイトの
コンテンツから配信され
たニュースが入って
います。



ここではSafari Webブラウザに組み込まれたニュースリーダを使っています。

RSS というのは実は XML です

RSS というのは HTML のような単なるテキストのマークアップ言語で、コンテンツを記述するためにタグや属性を使います。RSS は XML をベースにしています。XML というのは、汎用のマークアップ言語でどのような種類のデータでも記述することができます。XML は非常に柔軟性が高いため威力があります。実は XML はで特定のタグや属性を定義していないのです。単にタグや属性がどのように作られ、どのように使われるかという規則をセットするだけです。後は HTML や RSS など特定の言語が、詳細を決めてどのタグや属性をどのように使うのかを確立させなければよいのです。

RSS を習熟するためには、まず XML の一般則を理解しなければなりません。この一般則はすべての XML ベースの言語で成り立ちます。RSS はもちろん HTML の新しいバージョン XHTML でも通用します。この規則というのは単純ですが、重要です。この規則に違反した場合、XML(RSS) のコードは動きません。以下のようなものです。

- コンテンツを含むタグは、ペアでマッチさせなければなりません。

誤りです！空のタグには「>」の前に
空白とスラッシュが必要です。

- コンテンツを持たない空のタグは、空白とスラッシュを閉じタグ
記号 (>) の直前に置かなければなりません。

<p>Phone home! </

<p>Phone home!</p>

合っています。
↓
↓

- すべての属性値は、二重引用符で囲わなければなりません。

誤りです！属性は二重引用符で
囲まなければなりません。

合っています。
↑
↑

XML はマークアップ言語で、
どんな種類のデータを記述
するためにも使えます。

PHP では、たいていの場合、
二重引用符でも單一引用符でも
使うことができましたが、XML で
属性値を表現する場合、厳格に
二重引用符のみしか許されません。

素朴な疑問 に答えます

Q : RSSというのは単にWebサイトを訪れてもらうよりも、本当に優っているのでしょうか？

A : みんなが定期的にサイトを訪れてくれて、最新のコンテンツを探してくれるのであれば、RSSでもWebサイトで単にコンテンツを表示するのでも大差はありません。しかし、たいていの人はWebサイトのことを忘れてします。たとえそのサイトを気に入っていたとしてもです。そこでRSSの出番です。Webコンテンツを直接みんなに運ぶ方法を提供してくれます。みんなに探してもらうように要求するではありません。

Q : RSSというは何の略ですか？

A : 最近では、RSSはReally Simple Syndication(本当に単純なシンジケーション)の頭文字ということになっています。以前はいくつかのバージョンがあったようですが、現状の最新版(バージョン2.0)では、Really Simple Syndicationですから、これだけ覚えておけばよいでしょう。

Q : ではRSSを構成しているものは何ですか？

A : RSSというのはデータフォーマットです。つまりHTMLというのが単にデータフォーマットで、WebコンテンツをWebブラウザで閲覧するために記述することができるのと同様です。RSSもデータフォーマットで、ニュースフィードとしてアクセスできるようなWebコンテンツを記述できるようにしてくれます。HTMLと同様に、RSSのデータフォーマットは純粋なテキストで、タグと属性から構成されています。これらはニュースフィードのコンテンツを記述するために使われます。

Q : RSSリーダーはどこにあるのでしょうか？

A : たいていのWebブラウザにはRSSリーダーが組み込まれています。メールにもRSSリーダー機能を持つものがあります。この場合RSSのニュース項目は特別なニュースフィードフォルダへ送られるメールメッセージとなります。また単体のRSSリーダーソフトもあります。



以下は「UFOに誘拐された」のニュースフィード用のRSSコードです。影をつけた部分について、各タグの役割りがどんなものかを説明して下さい。

```

<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
<channel>
    <title> UFOに誘拐された！ニュースフィード </title>
    <link> http://aliensabductedme.com/ </link>
    <description> UFOによる誘拐レポートが、オーウェンと飼い犬ファンのため好意で世界中から
        寄せられています。</description>
    <language> ja-JP </language>

    <item>
        <title> 深沢 麻子 - 不格好で変な奴で、リズム感ゼロ。... </title>
        <link> http://www.aliensabductedme.com/index.php?abduction_id=7 </link>
        <pubDate> Sat, 21 Jun 2009 00:00:00 EST </pubDate>
        <description> 私に変な音楽で踊らせようとした。</description>
    </item>

    <item>
        <title> 清水 祐子 - 緑色で触覚が6つありました。... </title>
        <link> http://www.aliensabductedme.com/index.php?abduction_id=8 </link>
        <pubDate> Sun, 11 May 2009 00:00:00 EST </pubDate>
        <description> ちょっと話して犬と遊びました。</description>
    </item>

    ...
</channel>
</rss>
```



エクササイズ の答え

以下は「UFOに誘拐された」のニュースフィード用のRSSコードです。影をつけた部分について、各タグの役割りがどんなものかを説明して下さい。

```

<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
  <channel>
    <title> UFOに誘拐された！ニュースフィード </title>
    <link> http://aliensabductedme.com/ </link>
    <description> UFOによる誘拐レポートが、オーウェンと飼い犬ファングのために好意で世界中から  
寄せられています。</description>
    <language> ja-JP </language>
    <item>
      <title> 深沢 麻子 - 不格好で変な奴で、リズム感ゼロ。... </title>
      <link> http://www.aliensabductedme.com/index.php?abduction_id=7 </link>
      <pubDate> Sat, 21 Jun 2009 00:00:00 EST </pubDate>
      <description> 私に変な音楽で踊らせようとしたしました。</description>
    </item>
    <item>
      <title> 清水 祐子 - 緑色で触覚が 6 つありました。... </title>
      <link> http://www.aliensabductedme.com/index.php?abduction_id=8 </link>
      <pubDate> Sun, 11 May 2009 00:00:00 EST </pubDate>
      <description> ちょっと話して犬と遊びました。</description>
    </item>
    ...
  </channel>
</rss>

```

この行のコードはタグではありません。XML の「指令」(ディレクティブ)と言って、このドキュメントが XML コードを含んでいることを識別します。

この<title>タグは、全体のチャネルの役割りを果たします。

チャネルのリンクは通常ニュースフィードに結び付けられた Web サイトを指しています。

すべてのチャネルでは、どのようなニュースを提供するのかを記述する必要があります。

ニュースフィードを日本語以外で記述することもできます。このタグでチャネルの言語を確立します。

個々のニュース項目へのリンクで、通常はニュースフィードと結び付けられた Web サイト全コンテンツを指しています。

<pubDate>タグで指定された日付は、RFC-822 形式の日時フォーマットで表現されます。この表記は日時を詳細に表す際の標準的なフォーマットです。

RSS ドキュメントは1つのチャネルだけを含んでいます。ニュース項目を異なるカテゴリに分割する必要がない場合には、非常にやり方です。

すべての XML タグは、開始タグと終了タグを持たなければなりません。これは終了タグで RSS ドキュメントを閉じます。



わかったわ。じゃあRSSっていうのは実はXMLで、要するにタグのカタマリなんですよ。十分簡単よ。結局やらなければならないことは、ニュースフィードを作るためにXMLファイルを作るだけでしょう？

まあ、そうです。ただ通常XMLコードは手で書くのではなく、ほとんどの場合ファイルとして格納するわけでもありません。

XMLをファイルとして格納することはできますし、そのようなことはよく行われています。しかし、RSSでは動的なデータ、つまり常に変わり続けるデータを扱いますから、XMLをファイルに格納するということにあまり意味はありません。データはすぐに賞味期限切れになってしまいますから、継続的にファイルを書き換え続けなければならなくなってしまいます。それよりも、データベースからリアルタイムにXMLコードを作り出した方がよいでしょう。これはHTML版「UFOに誘拐された！」のメインページすでにやったことと同様です。つまりPHPを使って動的にRSS(XML)コードを生成し、リクエストに応じてRSSニュースリーダに直接コードを返せばよいわけです。

データベースからニュースリーダへ

UFOによる誘拐のデータにニュースフィードを提供するには、オーウェンが RSS のコードを MySQL データベースから動的に生成する必要があります。この RSS コードというのは、そのまま RSS ニュースリーダで読むことのできる、完全な RSS ドキュメント形式をしています。つまり PHP を使って UFO による誘拐に関するナマデータをフォーマットして RSS フォーマットにすれば、ニュースリーダで処理することができるので、ユーザはすぐ使うことができます。この処理の中で一番イケているところは、ニュースフィードが RSS フォームとして使えるようになった瞬間、そこから先へは自動的に行けるということです。更新されたニュース項目が出てきたら、それをどう表示するかはニュースリーダ側の責任なのです。

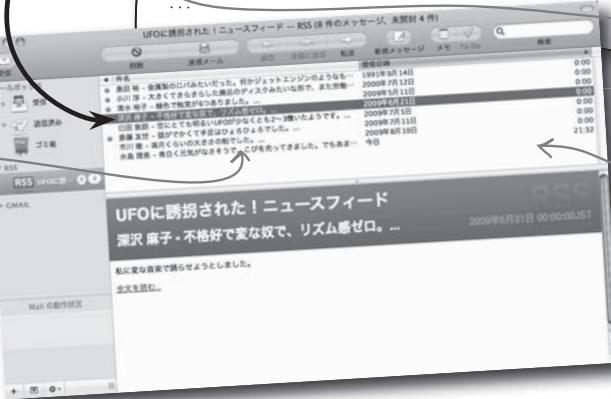
RSS ニュースリーダは RSS ニュースフィードによって供給されたデータを使うように設計されています。

abduction_id	last_name	first_name	when_it_happened	how_long	how_many	alien_description	what_they_did	...
1	小川	淳	2000-07-12	1週間	少なくとも12人	大きくてきらきらした废品のディスクリ...	音もなく空から近づいてきて、...	...
2	奥田	裕	1991-09-14	37秒	わからん	金属製のロバみたい...	座ってバイクドボテで食べて...	
3	水島	理恵	1969-01-21	約4年	1人だけ	青白く元気がなさそうで、...	私を不思議そうに見ながら、いろいろと...	
4	深沢	麻子	2009-06-21	ほぼ1週間	27	不格好で変な奴で、リズム感ゼロ。	私に変な音楽で踊らせようとした。	
5	清水	祐子	2008-05-11					

ニュースリーダは XML コードで書かれた個々のニュース項目を、どう解釈してユーザーにどのように表示すればよいか知っています。

```
<?xml version="1.0" encoding="utf-8"?>
<rss version="2.0">
<channel>
<title>UFOに誘拐された！ニュースフィード</title>
<link>http://aliensabductedme.com/</link>
<description>UFOによる誘拐レポートが、オーウェンと飼い犬ファンのため好意で世界中から寄せられています。</description>
<language>ja-JP</language>
```

```
<item>
<title>深沢 麻子 - 不格好で変な奴で、リズム感ゼロ。....</title>
<link>http://www.aliensabductedme.com/index.php?abduction_id=7</link>
<pubDate>Sat, 21 Jun 2009 00:00:00 EST</pubDate>
<description>私に変な音楽で踊らせようとしました。</description>
</item>
```



PHPを使って RSS ニュースフィード ドキュメントをMySQL データベースから 作ります。

個々のニュース項目は、RSS ニュースフィードドキュメントの中で、それぞれ個別のセクションを持っています。

ニュースリーダは Mac OS X の標準メーラに組み込まれています。他の多くの一般的なメールにもニュースリーダが組み込まれています。

誰が何をする？

RSSフィードを作ると言うことは、RSSを言語として理解するということに尽きます。つまりどのタグがどのニュース項目に使われるのかを理解しなければなりません。左側のRSSタグを対応する説明と結びつけて下さい。

<rss>

このタグは、RSSでは何もしません。しかし、ニュースデータを扱うと言う意味でイケてる名前です！

<channel>

公開日というのは、あらゆるニュース項目の中でも非常に重要な役割を持つ情報です。このタグにより公開日を指定します。

<cronkite>

このタグはRSSフィードで1つのチャネルを表現します。データを記述し、個々のニュース項目のコンテナの役割を果たします。

<title>

個々のニュース項目、つまりストーリーを表現します。この項目はさらに子の要素により説明されます。

<language>

このタグには常にURLを記述し、チャネルまたはニュース項目へのリンクの役割を果たします。

<link>

すべてのRSSフィードを開きます。つまり他のすべてのタグが、このタグの内側に現れます。

<description>

このタグにはチャネルまたはニュース項目のタイトルを格納します。通常は<channel>および<item>タグの中で用いられます。

<pubDate>

チャネルまたはニュース項目の概要説明のために使います。<channel>タグまたは<item>タグの中で使います。

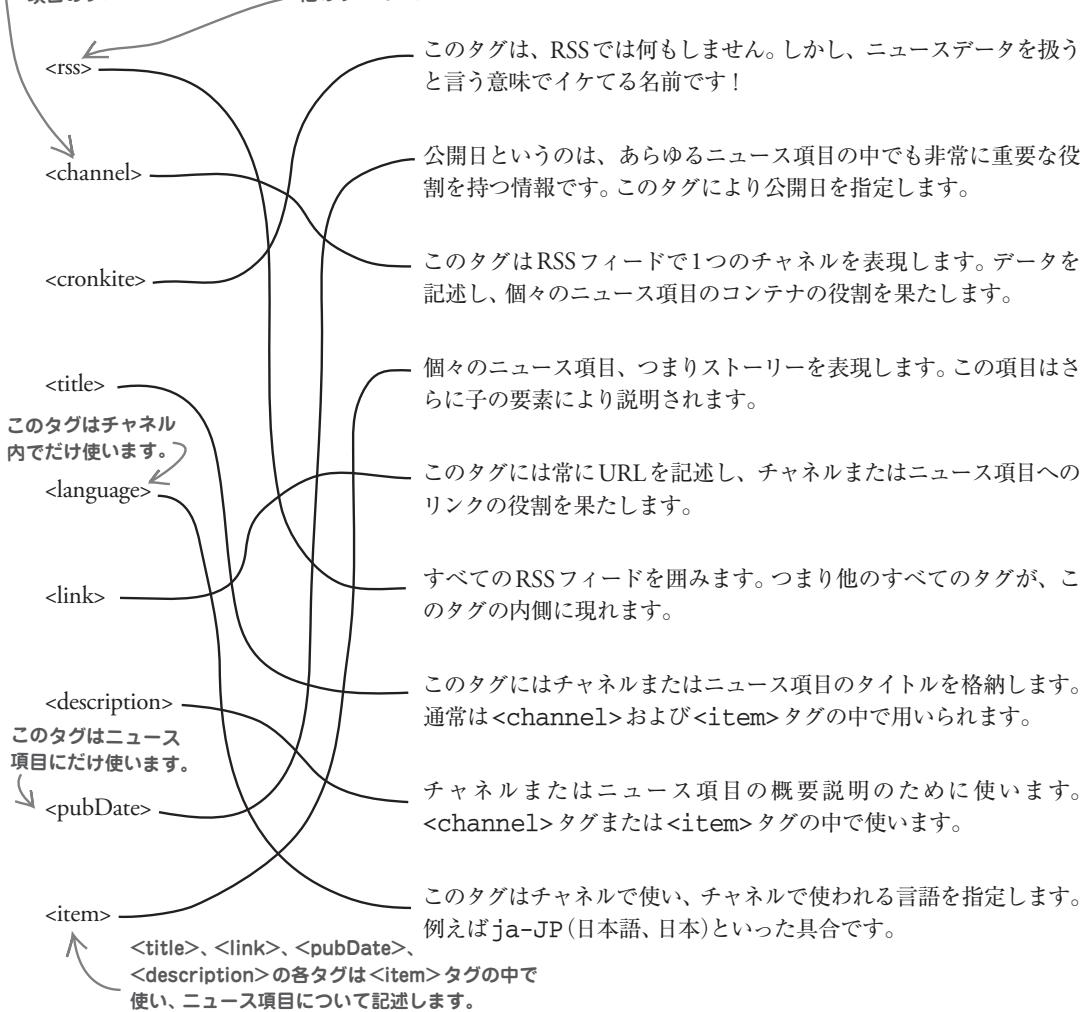
<item>

このタグはチャネルで使い、チャネルで使われる言語を指定します。例えばja-JP(日本語、日本)といった具合です。

誰が何をする？の答え

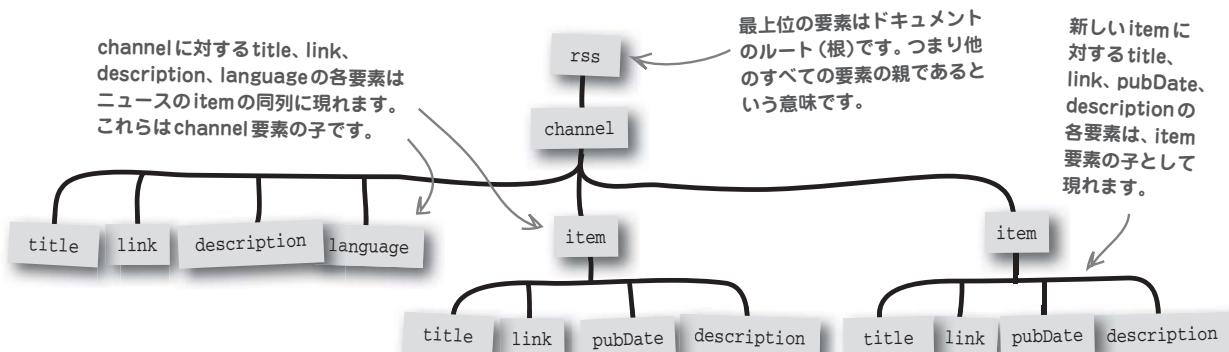
すべての RSS フィードは少なくとも 1 つのチャネルを含んでいます。チャネルとは基本的に関連するニュース項目のグループです。

RSS フィードを作ると言うことは、RSS を言語として理解するということに尽きます。つまりどのタグがどのニュース項目に使われるのかを理解しなければなりません。左側の RSS タグを対応する説明と結びつけて下さい。



RSS XMLを視覚化する

XMLコードはタグでできていることはすでに理解しました。タグは要素と呼ばれることもあり、完全なXMLドキュメントの文脈において親子関係を形成します。XMLコードで作業をする場合、この親子関係を視覚化することができると非常に便利です。例として反対側のページのRSSドキュメントを視覚化し、要素の階層を家族構成木(図)のように表現してみます。ニュースフィードデータについて親の要素を展開し、この要素に行くに従って下の方に進みます。



以下のテーブルは、真新しいUFOによる誘拐のレポートで、aliens_abductionデータベースに追加されたところです。この誘拐レポートについて、RSSの<item>タグに対応するXMLコードを書いて下さい。ニュースフィードにRSSフォーマットを結合させることを忘れないで下さい。

aliens_abduction

abduction_id	last_name	first_name	when_it_happened	how_long	how_many	alien_description	what_they_did	...
...								
14	臼田	敦朗	2009-07-05	2 時間	分かりません。	空にとても明るい UFOが…	光線で砂漠の中を ガソリンスタンド	...
...								



以下のテーブルは、真新しいUFOによる誘拐のレポートで、aliens_abductionデータベースに追加されたところです。この誘拐レポートについて、RSSの<item>タグに対応するXMLコードを書いて下さい。ニュースフィードにRSSフォーマットを結合させることを忘れないで下さい。

aliens_abduction

abduction_id	last_name	first_name	when_it_happened	how_long	how_many	alien_description	what_they_did	...
...								
14	臼田	敦朗	2009-07-05	2 時間	分かりません。	空にとても明るい UFOが…	光線で砂漠の中を ガソリンスタンド	…
...								

<item>

<item> タグで新しい
項目を囲みます。

<title>

<title>臼田 敦朗 - 空にとても明るいUFOが…</title>

<link>http://www.aliensabductedme.com/index.php?abduction_id=14</link>

<pubDate>Sat, 05 Jul 2009 00:00:00 EST</pubDate> <description>光線で砂漠の中をガソリンスタンド…</description>

</item>

<title>、<link>、<pubDate>、
<description>のそれぞれのタグで
ニュース項目の詳細を記述します。

<pubDate>タグはDだけが
大文字で、大文字・小文字が
混じっていなければなりません。
従って<pubdate>や
<PUBDATE>ではダメです。

素朴な疑問に答えます



: XMLでは大文字・小文字を区別しますか?

A: 区別します。XML言語は大文字・小文字を区別しますから、XMLのタグや属性を指定するときに、テキストが大文字か小文字かは重要です。RSSの<pubDate>タグが良い例で、Dだけが大文字で大文字・小文字が混じっていなければなりません。たいていのXMLタグは、すべて小文字であるか、あるいは大文字・小文字が混じっているかのいずれかです。



: では空白類文字はどうでしょうか?どのようにXMLに適応しているのでしょうか?

A: まず第1に、XMLで空白類文字といったら、復帰(CR、¥r)、改行(NL、¥n)、タブ(¥t)、空白(' ')のことを指します。たいていの場合、XMLドキュメントにおける空白類文字は、純粹にフォーマット上の美しさのために用いられます。例えば、子のタグをインデントするといった具合です。このような「どうでもよい」空白類は、通常XMLデータを処理するアプリケーション側、例えばRSSニュースリーダーは無視されます。しかしタグの内側に現れる空白類は非常に重要で、「どうでもよく」ありません。通常、そこに現れたとおりに解釈されます。このため、空白が意味を持つような詩のような場合でも、XMLでは正しく表現できることになります。



: RSSフィードにイメージを含めることができますか?

A: できます。ただし、すべてのニュースリーダーがイメージを表示できるわけではありませんから注意して下さい。またRSS 2.0では、イメージを入れられるのはチャネルだけです。個々のニュース項目はダメです。チャネルにイメージを入れる場合は、<image>タグを使います。このタグは<channel>タグの内側に現れなければなりません。以下に例を示します。

```
<image>
<url>http://www.aliensabductedme.com/fang.jpg</url>
<title>飼い犬のfangです</title>
<link>http://www.aliensabductedme.com</link>
</image>
```

技術的には、RSS 2.0のニュース項目にイメージを入れることも可能です。ただちょっと小細工が必要で、HTMLのタグを項目の記述の中で使うのです。このやり方は一応可能ですが、HTMLタグをXML実体(エンティティ)を使ってコード化することになります。このようなやり方は、RSS項目には純粋なテキストコンテンツしか含めてはならないという規約に反しているのです。



RSSを暴く

今週のインタビュー：どうすればニュース即配信が可能となるか？

Head First：ではまず、人々がWeb上でニュースを探す場合、あなたに頼っていると伺ったのですが、本当でしょうか？

RSS：それは何を持って「ニュース」とするかに依存すると思いますね。私の仕事は、情報をニュースリーダーすぐ読めるようなフォーマットなパッケージにすることです。その際コンテンツが本当に新しいかどうかは知りません…その点に関しては私はコントロールできないのです。それを決めるのは人々でしょうね。

Head First：え？ では「ニュースリーダー」というのは個々の人々と言う意味で使っていますか？

RSS：違います。ニュースリーダーとはRSSとはどんなもので、RSSがどのようにデータを表現しているのかを理解してくれるソフトウェアツールのことです。例えば、多くのメーラがRSSをサポートしています。つまりニュースフィードを購読し、更新情報をメールメッセージと同様に受け取ることができます。

Head First：面白いですね。ではメール自体はどう違うのでしょうか？

RSS：ええと、RSSはメールとは全然違いますよ。一例をあげれば、メールメッセージは人から人へ送られるもので、通常双方向対話の一部です。当然メールメッセージに応答することができるし、レスポンスをもらうこともあります。しかしRSSは一方通行の通信です。Webサイトから各個人に向けて発信するだけです。

Head First：どのようにして一方通行の通信が行われるのでしょうか？

RSS：ある人がニュースフィードを受信することに決めたとすると、ニュースリーダーソフトウェアでこれを購読することになります。基本的には、特定のWebサイトにポストされた新しいコンテンツについて知りたいと宣言したことになるのです。そこで新しいコンテンツが実際にポストされると、RSSでそれを表現します。つまりニュースリーダーソフトウェアが理解でき、その人に提示できるようになります。しかし、ニュース項目に返信するチャンスはないのです。つまりこれがWebサイトから各個人への一方通行の通信と言っている意味です。

Head First：分かりました。ではRSSとは正確には何なのでしょうか？

RSS：RSSというのは単にデータフォーマットです。双方合意の方法で、コンテンツを突っ込み、ニュースリーダーが認識できて、表示できるようになっています。RSSを使ってデータを突っ込めば、ニュースリーダーはニュースフィードとしてデータにアクセスできるのです。

Head First：なるほど。ではHTMLとはどのように違うのでしょうか？

RSS：RSSもHTMLもテキストのデータフォーマットであることに変わりありません。結局のところどちらもXMLベースです。つまりどちらもタグと属性を使ってデータを記述しているのです。しかし、HTMLはWebブラウザが処理し表示するという特定の目的で設計されているのに対し、RSSはニュースリーダーが処理し表示するために設計されています。HTMLとRSSは同じデータを異なる方法で表現しているということもできるかもしれません。

Head First：でも、Webブラウザでもニュースフィードを表示しているのを見たことがありますよ。あれはどうなっているのでしょうか？

RSS：良い質問です。すでに明らかかと思いますが、Webブラウザによっては、ニュースリーダーが組み込まれているものがあるのです。つまりこのようなブラウザは、実際には2通りの使い方ができます。しかし、Webブラウザでニュースフィードを閲覧する場合、HTMLをWebブラウザで見る場合と、全く異なるページが見えることになります。

Head First：でも、たいていのニュースフィードはHTMLのWebページにリンクしているんですよね？

RSS：その通りです。つまりRSSはHTMLと協力し合ってWebコンテンツにアクセスするためのより良い方法を提供しているのです。アイデアとしては、RSSを使ってWebサイトに直接訪問することなく、新しいコンテンツについての情報を入手します。次に何かもっと知りたいと思えるものを見つければ、クリックして実際のページへ飛んで行けばよいのです。これが新しい各項目にリンクが付いている理由です。

Head First：ということは、RSSというのはWebページのプレビューみたいなものですよね。

RSS：正にその通りです。でも覚えておいて欲しいのは、RSSが皆さんのがころへ行っても、皆さんはRSSを訪ねる必要はないということです。これが、RSSが好かれる本当の理由かもしれません。RSSは新しいコンテンツのタブを持ち続けていくので、人々がWebサイトを再訪させるようにすることができます。

Head First：よく分かりました。本当に便利ですね。RSSのWebでの役割を分かりやすく説明してくれてありがとうございました。

RSS：いえいえ、こちらこそ、ありがとうございます。後はお構いなく。

RSS フィードを動的に作る

RSS データフォーマットを理解するということに関しては大体十分です。しかしオーウェンは相変わらずニュースフィードを使って、UFOによる誘拐レポートを人々に届ける必要があります。そろそろ PHP の殻を破って、動的にニュースフィードを作り始めましょう。オーウェンの MySQL データベースから UFO による誘拐のデータを全部引っ抜いてニュースフィードを作ります。幸せなことに、これは以下のステップを踏襲することで達成できます。

できあがったニュースフィードは
ファイルに格納するのではありません。
XML ドキュメントです。



- ① ドキュメントのコンテンツタイプ (Content type) を XML にセットします。

ヘッダを使って
XML のコンテンツ
タイプを RSS

```
<?php header('Content-Type: text/xml'); ?>
```

ドキュメント ② XML 指令を生成し、これが XML ドキュメントであることを明示します。

にセットしなければ
なりません。

```
<?php echo '<?xml version="1.0" encoding="utf-8"?>'; ?>
```

- ③ 静的な RSS コードを生成します。この部分はデータベースから引っ張ってくる必要はありません。<rss> タグやチャネル情報などです。

```
<rss version="2.0">
  <channel>
    <title>...
    <link>...
    <description>...
    <language>...
```

このコードはデータベースと
関係ありません。このニュース
フィードでは常に同じです。

- ④ aliens_abduction データベースに対して、UFO による誘拐データに関する問い合わせ文を作ります。

```
abduction_id
  first_name   last_name
when_it_happened      alien_description
                        what_they_did
```

ニュース項目に対する
RSS コードを生成する
前に、MySQL データ
ベースに問い合わせて、
UFO による誘拐データ
を引っ抜いてくる
必要があります。



- ⑤ データをループして、新しい各項目について RSS コードを生成します。

```
<item>
  <title>...
  <link>...
  <pubDate>...
  <description>...
</item>
```

このコードにデータベースから
抜いてきたデータが入っていますから、
注意して作り上げなければなりません。

- ⑥ ドキュメントを完成させるために必要な静的な RSS コードを生成する。終了タグの </channel> や </rss> です。

```
</channel>
</rss>
```



& XML! PHP & MySQL マグネット

オーウェンの「UFOに誘拐された！」用RSSニュースフィードスクリプト(newsfeed.php)にはいくつか重要なコードが欠けています。適切なマグネットを注意して選びコードを完成させ、ニュースフィードを動的に生成するようにして下さい。

```

1  <?php header('Content-Type: text/xml'); ?>
2  <?php echo '<?xml version="1.0" encoding="utf-8"?>'; ?>
<rss version="2.0">

3  .....  

<title>UFOに誘拐された！ニュースフィード</title>  

.....http://aliensabductedme.com/.....  

<description>UFOによる誘拐レポートが、オーウェンと飼い犬ファングのために好意で世界中から寄せられています。</description>  

.....ja-JP.....
```



```

4  <?php
    require_once('connectvars.php');

    // Connect to the database データベースへの接続
    $dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

    // Retrieve the alien sighting data from MySQL UFO目撃データをMySQLから取得
    $query = "SELECT abduction_id, first_name, last_name, "
        . "DATE_FORMAT(when_it_happened, '%a, %d %b %Y %T') AS when_it_happened_rfc, "
        . "alien_description, what_they_did "
        . "FROM aliens_abduction "
        . "ORDER BY when_it_happened .....";
```



```

5  $data = mysqli_query($dbc, $query);

    // Loop through the array of alien sighting data, formatting it as RSS
    while ($row = mysqli_fetch_array($data)) { UFO目撃データの配列を走査し、RSSとして整形
        // Display each row as an RSS item 各行をRSS項目として表示

        echo '.....';
        echo ' <title>' . $row['last_name'] . ' ' . $row['first_name'] . ' - ' .
            mb_substr($row['alien_description'], 0, 32, 'UTF-8') . '...</title>';
        echo ' <link>http://www.aliensabductedme.com/index.php?abduction_id=' .
            $row['.....'] . '</link>';
        echo ' .....' . $row['when_it_happened_rfc'] . ' ' . date('T') . '.....';
        echo ' <description>' . $row['what_they_did'] . '</description>';
        echo '</item>';

    }
    ?>
```



```

6  </channel>
</rss>
```



newsfeed.php



& XML! PHP & MySQL マグネットの答え

オーウェンの「UFOに誘拐された！」用RSSニュースフィードスクリプト(newsfeed.php)にはいくつか重要なコードが欠けています。適切なマグネットを注意して選びコードを完成させ、ニュースフィードを動的に生成するようにして下さい。

```

1 <?php header('Content-Type: text/xml'); ?> ← このヘッダによりスクリプトは
2 <?php echo '<?xml version="1.0" encoding="utf-8"?>'; ?> XMLドキュメントを出力することになります。
<rss version="2.0">

3   <channel>
    <title>UFOに誘拐された！ニュースフィード</title> </link>
    <link>...http://aliensabductedme.com/...
    <description>UFOによる誘拐レポートが、オーウェンと飼い犬ファンのため好意で世界中から寄せられています。</description>
    <language>.ja-JP.</language>

<?php
require_once('connectvars.php');

// Connect to the database
$dbc = mysqli_connect(DB_HOST, DB_USER, DB_PASSWORD, DB_NAME);

// Retrieve the alien sighting data from MySQL
$query = "SELECT abduction_id, first_name, last_name, " .
"DATE_FORMAT(when_it_happened,'%a, %d %b %Y %T') AS when_it_happened_rfc, " .
"alien_description, what_they_did " .
"FROM aliens_abduction " .
"ORDER BY when_it_happened .. DESC ..";

4 $data = mysqli_query($dbc, $query);

// Loop through the array of alien sighting data, formatting it as RSS
while ($row = mysqli_fetch_array($data)) {
    // Display each row as an RSS item
    echo '...';
    echo ' <item>...';
    echo ' <title>' . $row['last_name'] . ' ' . $row['first_name'] . ' - ' .
        mb_substr($row['alien_description'], 0, 32, 'UTF-8') . '...</title>';
    echo ' <link>http://www.aliensabductedme.com/index.php?abduction_id=' .
        $row['abduction_id'] . '</link>';
    echo ' <pubDate>' . $row['when_it_happened_rfc'] . ' ' . date('T') . '</pubDate>';
    echo ' <description>' . $row['what_they_did'] . '</description>';
    echo '</item>';
}
?>

</channel>
</rss>

```



newsfeed.php

ASC

<pubDate>

</item>

last_name

first_name

</channel>

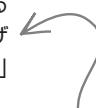


試運転

RSSニュースフィードスクリプトを「UFOに誘拐された！」サイトに追加する

新しくテキストファイルを作り newsfeed.php という名前を付け、オーウェンの RSS フィードスクリプトのコードを打ち込みます。数ページ前で完成させたマグネットエクササイズのものです。(またはサイト <http://www.oreilly.co.jp/books/9784873114446> からダウンロードして下さい。)

できあがったスクリプトを Web サーバにアップロードし、ニュースリーダで開いてみます。たいていの Web ブラウザやメーラなら、ニュースフィードを閲覧することができるはずです。もし単体のニュースリーダアプリケーションが見つからない場合は、ブラウザかメーラで試してみて下さい。ニュースフィードのスクリプトは、「UFOに誘拐された！」のデータベースから最新の UFO により誘拐レポートを直で引っ張ってくるはずです。



ブラウザでニュースフィードを閲覧する際に何か問題があったら、URL を <http://feed://> で始める代わりにみて下さい。

UFOに誘拐された！ニュースフィード

市川 徹 - 満月くらいの大きさの船でした。... 8月 10 日 (00:00)
山の頂上で私を連れて行き、そこから落としました。[全文を読む](#)

斎藤 友世 - 頭がでかくて手足はひょろひょろでした。... 7月 11 日 (00:00)
私の心を読んでいました。[全文を読む](#)

和田 敏郎 - 空にとても明るいUFOが少なくとも2~3機いたようです。... 7月 5 日 (00:00)
光線で砂漠の中をガソリンスタンドまで導いてくれました。[全文を読む](#)

深沢 麻子 - 不格好で変な奴で、リズム感ゼロ。... 6月 21 日 (00:00)
私に変な音楽で踊らせようとした。[全文を読む](#)

清水 裕子 - 緑色で触覚が6つありました。... 6月 11 日 (00:00)
ちょっと舐して大好きでした。[全文を読む](#)

三浦 大輔 - 大きくてきらきらした商品のディスクみたいな形で、また労働... 2000年 7月 19 日 (00:00)
ついでに、何も言わずに私は連れて行ってしまいました。[全文を読む](#)

口バみたいだった。何かジェットエンジンのようなも... 1991年 9月 14 日 (00:00)
食べてたら、ものすごい音がして光線に連れ去られてしまった。[全文を読む](#)

光気がなさそうで、こびを売ってきました。でもあま... 1989年 1月 21 日 (00:00)
ん。色々と調べてみました。[全文を読む](#)

記事を検索：
検索の長さ：
並べ替え：
日付
タイトル
ソース
新規を上へ
最近の記事：
すべて
今日
昨日
直前の 7 日間
今月
先月
ソース：
UFOに誘拐された...
アクション：
今までアップデートメールで知らせる
 RSS で会員登録
ブックマークに追加...

このニュースフィードって
すごそうだね。でもサイトに
訪問してくれた人はどうやって
ニュースフィードを見つけ
出すのかな？

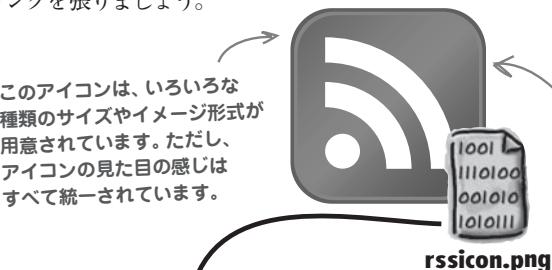


ホームページからニュースフィードへのリンクを張ればよいのです。

newsfeed.php というのは、単に PHP スクリプトであるということを忘れないで下さい。これと他の大部分の PHP スクリプトとの違いは、目しているものが RSS ドキュメントであって、HTML ドキュメントではないということだけです。それでも、他の PHP スクリプトと同様にアクセスすることができます。URL にスクリプトの名前を指定するだけでよいのです。オーウェンがやり残していることは、サイトを訪れた人に対して、この URL を共有できる方法を提供することだけです。これはほんのちょっとの労力でできます。シンジケーションリンクを張ればよいのです。これは単にオーウェンのサーバ上での、newsfeed.php スクリプトへのリンクです。

RSS フィードへのリンク

ニュースフィードへのリンクを目立つように提供しておくというのは非常に重要なことです。多くのユーザはそのようなサービスがあるということを喜んでくれることでしょう。ユーザが特定のサイトの RSS フィードをすぐに見つけられるようにしてあげるために、標準的なアイコンが用意されています。一目見てフィードと分かるように、これを使います。オーウェンのホームページ(index.php)の下部にこのアイコンを使ってニュースフィードへのリンクを張りましょう。



```
<p>
<a href=" newsfeed.php ">

クリックすると誘拐ニュースフィードをシンジケーションします。
</a>
</p>
```

ニュースフィードの URL は単に newsfeed.php スクリプトで構いません。これは下にあるメインの Web ページと同じフォルダにスクリプトが置いてあればちゃんと動きます。

HTML のニュースフィードリンクは、RSS アイコンと説明のテキストでできています。

「UFO に誘拐された！」のメインページ上に目立つようにリンクを張れば、サイト閲覧者は、オーウェンのニュースフィードにすぐにアクセスできるようになります。

標準の RSS アイコン  があるので、これを使えば RSS ニュースフィードを提供していることをユーザに分かりやすく示すことができます。

誘拐されていた時間	宇宙人の様子	ファングはいましたか？
2009-08-10 : 市川 徹 3時間	宇山人の様子： 満月くらいの大きさの船でした。	ファングはいましたか？ no
2009-07-11 : 斎藤 友世 45分	宇宙人の様子： 頭がでかくて手足はひょろひょろでした。	ファングはいましたか？ yes
2009-07-05 : 白田 敏郎 2時間	宇宙人の様子： 空にとても明るいUFOが少なくとも2~3機いたようです。	ファングはいましたか？ yes
2009-06-21 : 深澤 麻子 ほぼ1週間	宇宙人の様子： 不格好で変な奴で、リズム感ゼロ。	ファングはいましたか？ no
2009-05-11 : 清水 裕子 1日	宇宙人の様子： 緑色で触覚が6つありました。	ファングはいましたか？ yes



試運転

ニュースフィードリンクを「UFOに誘拐された！」ホームページに追加する。

「UFOに誘拐された！」のindex.phpスクリプトを修正して、ニュースフィードリンクをページの下部に表示させます。またrssicon.pngイメージをコードの一部として取り入れるため、サイト<http://www.oreilly.co.jp/books/9784873114446>からダウンロードします。

index.phpスクリプトとrssicon.pngイメージをWebサーバにアップロードしたら、スクリプトをWebブラウザで開きます。新しく作ったリンクをクリックし、RSSニュースフィードを見て下さい。

これで誘拐の成り行きを見ていれば、いつでも宇宙人を見張っていきることができるわ。

RSSのおかげで、新しいUFOによる誘拐レポートが購読者に「ブッシュ」され、購読者は「UFOに誘拐された！」Webサイトを直接訪ねる必要はないのです。

ファングを見たことはないけど、このレポートはすごいね。

アタシがYouTubeで見たのと同じ犬じゃないかしら…

由美子さんです。熱心に「UFOに誘拐された！」ニュースフィードを読んでいて、ファングをYouTubeで見たかも知れないと思っています。

件名	送信日時	件数
■ 斎田 栄 - 金属製の口ばかりだった。何かジェットエンジンのようなも…	1991年9月14日	0.00
■ 小川 淳 - 大きくてさり気なさった商品のディスクあたりが、また効率…	2008年7月12日	0.00
■ 関根 勝 - 空港で候室で待ちました。…	2009年5月11日	0.00
■ 佐藤 勝明 - 空港で待つ間、音楽を聴いていたときに、リズム感が…	2009年6月21日	0.00
■ 豊島 友智 - 朝がでかけて私はひさしひさなくとも2~3回いたようです。…	2009年7月5日	0.00
■ 清川 雅 - 清月くらいの大きさの船でした。…	2009年7月11日	0.00
■ 氷島 理恵 - 青白く光気がなさそうで、飛び出していました。でもあま…	2009年8月10日	0.00
		21:32

百万 ビデオ 千の言葉より一枚の写真を以て語らしめる。

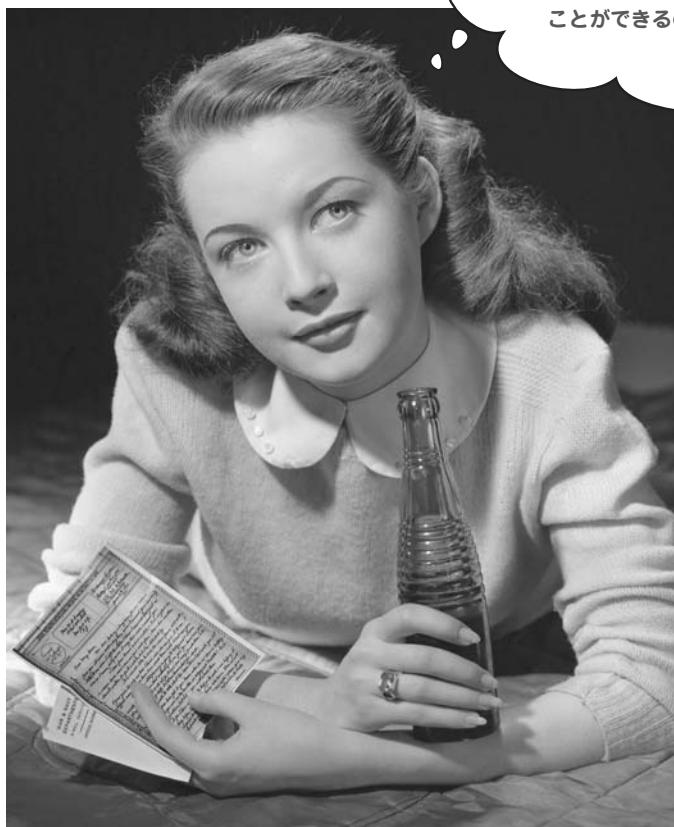
ニュースフィードの購読者のおかげでオーウェンはYouTubeに目をつけました。YouTubeビデオでファンゲに似た犬を見たと言ってきたからです。オーウェンはさらに新しいテクノロジーを使ってファンゲを探す方法を拡張する必要があるとの認識を持ちました。でもどうすればいいのでしょうか？オーウェンが「UFOに誘拐された！」サイトにYouTubeビデオを組み込むことができれば、ユーザはみんなファンゲの見張りをすることができるのです。それだけでなく、自分でYouTubeビデオを絶えず探すといったアホみたいな作業をやめたいという必要にも迫られています。



やってみよう!

- ① オーウェンのYouTubeビデオにアクセスしてみて下さい。
www.youtube.com/user/aliensabductedmeです。
- ② オーウェンが見つけたUFOによる誘拐のビデoをいくつか観て下さい。
ビデoの犬はファングでしょうか？

YouTubeを自分で探すんじゃなくて
「UFOに誘拐された！」サイトから直接ビデoを
観ることができたらステキじゃない？ただWeb
ページに行って、もう見つかっているものを観る
ことができるのよ。夢物語なのかしら…



Webコンテンツを他のサイトから引っ張ってくる

RSSニュースフィードの背景にあるアイデアは、コンテンツを外に対してプッシュすることで、Webサイトに新しいコンテンツが上がったかどうかを絶えず確認しなくともよくなるということでした。これは人々にサイトのタブを維持しておいてもらう上で非常に便利な方法です。オーウェンが気付いた通りです。ところでWebシンジケーションというコインには反対側もあります。つまり、自分のサイトに別のサイトからコンテンツをブルしてくる（引っ張ってくる）のです。つまり自分のサイトが利用者側になり、他の誰かのサイトがコンテンツのプロバイダとして働くわけです。オーウェンの場合で言うと、YouTubeビデオを彼のサイトで見せるためには、YouTubeがプロバイダとなるわけです。

**「UFOに誘拐された！」サイトは
ビデオの利用者です。**

The screenshot shows a table of user reports. The columns are: 报告日 (Report Date), 姓氏 (Name), 誘拐されていた時間 (Time of Abduction), 宇宙人の様子 (Description of Alien), ファングはいましたか? (Did you have a fan? - Yes/No), and ファングはいましたか? (Did you have a fan? - Yes/No). The data includes:

報告日	姓氏	誘拐されていた時間	宇宙人の様子	ファングはいましたか?
2009-08-10	市川 徹	3時間	宇宙人の様子：満月くらいの大きさの船でした。	no
2009-07-11	斎藤 友世	45分	宇宙人の様子：頭がでかくて手足はひょろひょろでした。	yes
2009-07-05	白田 敏朗	2時間	宇宙人の様子：空にとても明るいUFOが少なくとも2~3機いたようです。	yes
2009-06-21	深沢 麻子	ほぼ1週間	宇宙人の様子：不格好で変な奴で、リズム感ゼロ。	no
2009-05-11	清水 裕子	1日	宇宙人の様子：緑色で触覚がちちありました。	yes

ビデオのサムネイルイメージ
をここに置きます！

「UFOに誘拐された！」サイトのホームページ
を若干変更して、ビデオ検索結果を表示する
ための場所を作る必要があります。

まず始めに理解しておくべきことは、オーウェンは単に特定のYouTubeビデオ本体または、ビデオへのリンクを自分のサイトに組み込みたいわけではないのです。そんなことは、YouTubeからHTMLコードをカット&ペーストするだけでできちゃいます。彼が実際にやりたいことは、YouTubeビデオをその都度検索して、その検索結果を表示することです。つまり「UFOに誘拐された！」サイトでは、YouTubeのデータに対して、リアルタイムに問い合わせを実行し、その結果を動的に表示する必要があるのです。これによりオーウェンやその援軍としてファンをしてくれる人々は、YouTubeに投稿されたUFOによる誘拐のビデオを常に監視し続けることができるようになります。

**YouTubeはビデオの
プロバイダです。**



YouTubeでUFOによる誘拐の
検索をかけた結果見つかった
ビデオが、YouTubeから返って
くるのでオーウェンのメイン
ページのフィードとなります。

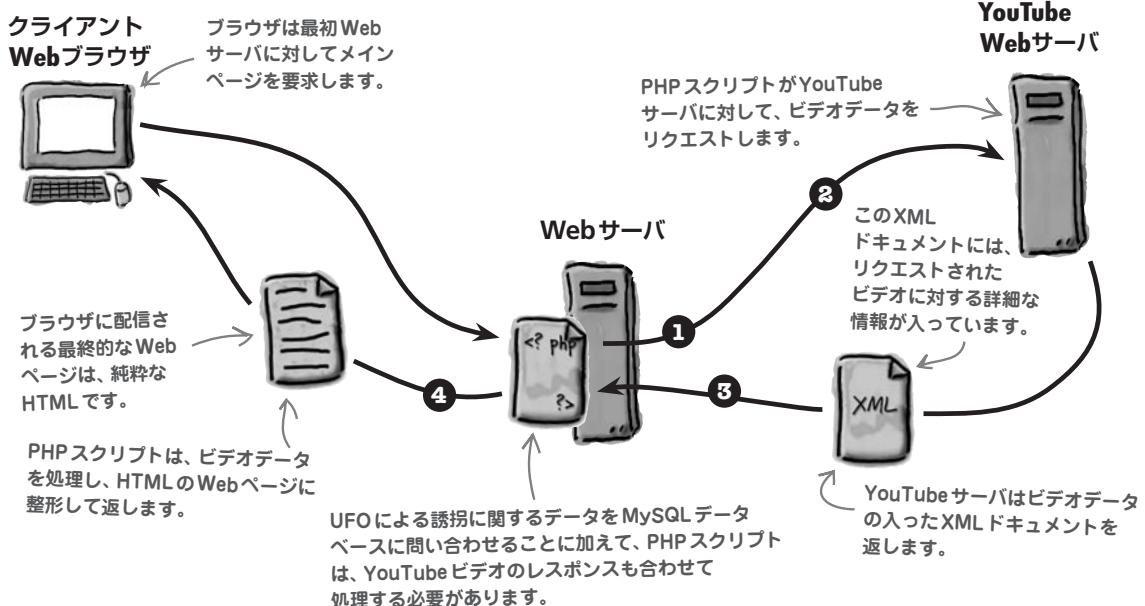
YouTubeビデオとシンジケーションする

YouTubeからビデオを引っ張ってくるには、まずYouTubeがどのようにシンジケーションに対応しているのかを正確に把握しなければなりません。YouTubeではシンジケーションによりビデオを提供しています。その方法はリクエスト・レスポンスによるコミュニケーションプロセスを通して行われており、特定のビデオをリクエストすると、そのビデオに関する情報をYouTubeのサービスからのレスポンスとして受け取るのです。YouTubeが規定するフォーマットでリクエストを発行し、そのレスポンスを処理するという両方に責任を持たなければなりません。その処理には、レスポンスデータから所望の特定のビデオ(タイトル、サムネールイメージ、リンク等)をより分ける、といった作業も含まれます。

以下に、ビデオをYouTubeから引っ張ってきて、表示するまでに必要となるステップを示します。

YouTubeのビデオをシンジケーションするには、リクエストを発行し、レスポンスを処理する必要があります。

- ① YouTubeビデオに対するリクエストを組み立てる。
このリクエストは通常URLのフォームに入ります。
- ② できあがったリクエストをYouTubeに投げる。
- ③ YouTubeのレスポンスデータを受け取る。通常ビデオに関する情報が入っている。
- ④ レスponsデータを処理し、HTMLコードにフォーマットする。
YouTubeは、XMLを使ってビデオリクエストにレスポンスします。



YouTubeビデオリクエストを作る

YouTubeからビデオを取ってきて、それを自身のWebページに組み入れるには、まずリクエストを作ることから始めます。YouTubeはRESTリクエストの使用を通してビデオがリクエストされるものと定めています。これはカスタムURLといって特定のリソースに誘導してくれます。例えば（もちろん）YouTubeビデオのデータです。所望のビデオを識別するURLを作り上げると、YouTubeはそのビデオに関する情報をXMLドキュメントに載せて送り返してくれます。

YouTubeへのリクエストURLに関する詳細は、どのビデオにアクセスしたいかに依存して決まります。例えば、特定のユーザのお気に入りのビデオといった具合にリクエストすることもできます。オーウェンの場合だと、ベストなアプローチは、キーワードによる検索をすべてのYouTubeビデオに対して実行することだろうと考えられます。この種のリクエストに対応するURLは、各種微妙に異なっていますが、URLの最初は常に同じで、ベースURLはこのような形式です。

`http://gdata.youtube.com/feeds/api/` ←
YouTubeへのすべてのRESTリクエスト
は、このベースURLを使います。



ユーザでのビデオリクエスト

お好みのビデオとして特定のYouTubeユーザのものをリクエストするには、ベースURLにYouTubeのユーザ名を追加します。

`http://gdata.youtube.com/feeds/api/users/username/favorites`

ユーザ名がelmerpriestleyのお気に入りのビデオをリクエストする場合、以下のURLを使用します。

`http://gdata.youtube.com/feeds/api/users/elmerpriestley/favorites`



キーワード検索によるビデオリクエスト

もっと強力でかつ通常もっと便利なYouTubeビデオのリクエストと言えば、キーワード検索です。これは特定のユーザとは関係ありません。1つ以上のキーワードを斜線記号で区切ってURLの後ろにくっ付けることができます。

`http://gdata.youtube.com/feeds/api/videos/-/keyword1/keyword2/...`

URLはユーザによるリクエストの場合
と同様に始まりますが、ここでusers
の代わりにvideosを使います。

お気に入りのビデオをキーワード「elvis」と「impersonator」とでリクエストする場合、以下のURLを使用します。

`http://gdata.youtube.com/feeds/api/videos/-/elvis/impersonator`

キーワードは大文字・小文字を区別しませんから、「elvis」
でも「Elvis」でも「eLvis」でも全部結果は同じになります。

素朴な疑問

に答えます

Q: RESTというのは何の略ですか？

A: REpresentational State Transfer（表現的状態遷移）の略です。これで確かに、一応頭文字ではありますですが、実際よりも空想的で技術的に聞こえます。RESTの背景にある主なアイデアは、Web上のリソースが一意なリンクでアクセスできるべきである、ということです。つまり「RESTful」（平穡）なデータには、単にそのURLを構築することでアクセスできるべきだという意味です。YouTubeで言うと、ビデオの問い合わせ文を、検索の基準を持ったURLを通して実行すればよいという意味です。

YouTubeユーザのユーザ名をここに書くことで、そのユーザのお気に入りのビデオにアクセスできます。

このRESTリクエストの結果は、YouTubeユーザelmerpriestleyのお気に入りのビデオとなります。

複数のキーワードを使ってビデオの検索ができます。キーワードは斜線記号で区切れます。

ここで探索キーワード「elvis」と「impersonator」とを使って、ビデオを検索します。

YouTubeのRESTリクエストになってみよう



YouTubeの心の中に入ったつもりになって、ビデオのRESTリクエストになって下さい。下にあるマグネットを使って、以下のYouTubeビデオに対する、ビデオのRESTリクエストを組み立てて下さい。
できあがったら、Webブラウザで試してみて下さい。

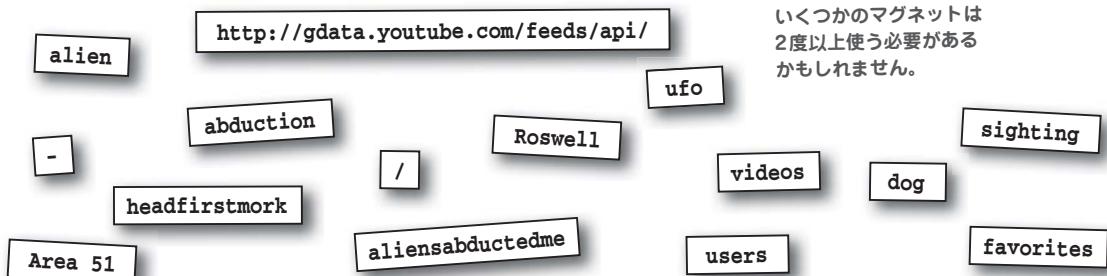
キーワード「Roswell」にマッチするすべてのビデオ：

キーワード「alien」と「abduction」とにマッチするすべてのビデオ：

ユーザ headfirstmork がお気に入りタグをつけたすべてのビデオ：

キーワード「ufo」と「sighting」と「dog」にマッチするすべてのビデオ：

ユーザ aliensabductedme がお気に入りタグをつけたすべてのビデオ：





YouTube の REST リクエストになってみようの答え

YouTube の心の中に入ったつもりになって、ビデオの REST リクエストになって下さい。下にあるマグネットを使って、以下の YouTube ビデオに対する、ビデオの REST リクエストを組み立てて下さい。
できあがったら、Web ブラウザで試してみて下さい。

いくつかのマグネットは 2 度以上使ったかもしれません。

すべての REST リクエストについて YouTube のベース URL には同じものを使用します。

URL の最後に 1 つのキーワードが現れます。

キーワード「Roswell」にマッチするすべてのビデオ：

`http://gdata.youtube.com/feeds/api/ videos / - / Roswell`

URL の最後に検索キーワードが現れます。それぞれのキーワードは斜線記号で区切ります。

キーワード「alien」と「abduction」とにマッチするすべてのビデオ：

`http://gdata.youtube.com/feeds/api/ videos / - / alien / abduction`

ユーザ headfirstmork がお気に入りタグをつけたすべてのビデオ：

`http://gdata.youtube.com/feeds/api/ users / headfirstmork / favorites`

ユーザのお気に入りの場合、リクエスト URL には「video」の代わりに「users」が入ります。

キーワード「ufo」と「sighting」と「dog」にマッチするすべてのビデオ：

`http://gdata.youtube.com/feeds/api/ videos / - / ufo / sighting / dog`

URL は「favorites」で終わります。

ユーザ aliensabductedme がお気に入りタグをつけたすべてのビデオ：

`http://gdata.youtube.com/feeds/api/ users / aliensabductedme / favorites`

Area 51

このマグネットは
使いません…ワナです！

ここにお気に入りに
アクセスしたいユーザの
ユーザ名を書きます。

素朴な疑問 に答えます

 : RESTというのは、例えばGETリクエストとはどのように違うのでしょうか?

 : 違いはありません。GETリクエストを使ってきたところ、例えば単純にWebページを要求する場合などではいつでも、RESTを使っています。通常のWebページは、RESTのリソースであると考えることもできます。Webページには、URLを通してアクセスすることができ、GETというのはリソースにアクセスするために使われるRESTの「動作」であるということです。RESTがもっと威力を発揮するのは、問い合わせ文を作るために使われるときです。例えばYouTubeビデオのリクエストがそうです。この場合もRESTリクエストを扱ってはいますが、単に静的なWebページをリクエストするのではなく、データベースの問い合わせになっています。

 : YouTubeのキーワード検索を実行する際、引数の順序が問題になることがありますか?

 : あります。第1のキーワードには後ろの方のキーワードに比べて優先順位が高くなります。ですから重要性が減っていくような順序でキーワードを並べる必要があります。

 : ビデオ検索で複数のマッチが得られた場合、どのビデオを返すのかを、YouTubeはどうやって決めるのでしょうか?

 : YouTubeのキーワードによるビデオリクエストでは、検索の関連に基づいてビデオが返ってきます。つまり、キーワードにベストマッチするビデオが返ってきます。そのビデオがいつYouTubeにポストされたのかとは関係ありません。



オーウェンはRESTリクエストを組み立てる準備ができました

オーウェンにとってゴールは、YouTubeにすがってUFOによる誘拐のビデオの中にファンが映っているかどうかを調べることです。このような場合YouTubeに飛ばすRESTリクエストとしては、キーワード検索が最も理にかなっています。ファンが映っているかもしれないビデオを検索できる可能性があるキーワードの組み合わせとしては、たくさんの種類が考えられます。しかし、ファンに特に関係のあるビデオを見つけるには、これが良いでしょう。



`http://gdata.youtube.com/feeds/api/videos/-/ alien / abduction / head / first`

本のシリーズ名を参照して、通常のYouTubeのビデオ検索を使うということは、多分ありえないでしょうが、特にこの場合に関しては何故かうまくいきます。偶然にもUFOによる誘拐のビデオが、Head First ファンによってたくさん作られたのです！今あるRESTリクエストURLを使って、オーウェンはYouTubeビデオのシンジケーション処理をステップ1から順に処理済として消していくことができます。

↑
↑
最後の2つのキーワードが
UFOによる誘拐のビデオのうち
オーウェンとファンに関係ある
ものだけに限定してくれます！

第1ステップはもうやっつけ
ました。YouTubeリクエスト
URLのおかげです。

1 YouTubeビデオ用のリクエストを作る。

- 2 YouTubeヘビデオリクエストを投げる。
- 3 YouTubeからビデオの情報を含むレスポンスデータを受け取る。
- 4 レスポンスデータを処理し、データをHTMLコードに整形する。



試運転

オーウェンのYouTubeリクエストURLを試してみる。

オーウェンのYouTubeリクエストURLをWebブラウザに打ち込んで下さい。

<http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first>

ブラウザは何を表示したでしょうか？ページのソースを表示させて、YouTubeが実際に返してきたコードを見て下さい。

YouTube Videos

総数：3

記事を検索：

記事の長さ：

並べ替え：

- 日付
- タイトル
- ソース
- 新規を上に

最近の記事：

- すべて
- 今日
- 昨日
- 直前の7日間
- 今月
- 先月

ソース：

YouTube Videos

アクション：

- 今すぐアップデート
- メールで知らせる
- Mailで照会
- ブックマークに追加...



WebブラウザでXMLデータを表示しています。
ニュースフィードとしてのYouTubeレスポンス
です。ただし、この場合各項目は実際にビデオで
ある点が異なります。



Web ブラウザに URL を打ち込んで
YouTube にビデオをリクエストするのって
ステキで完璧。でも PHP で何ができるの?
スクリプトから返ってきたビデオにアクセスする
なんて無理なんじゃないの?

PHP の単純な XML 拡張です。
`simplexml_load_file()` 関数が PHP のバージョン 5 から
追加されました。つまり PHP の以前の版では XML を
処理する組み込みのサポートはありませんでした。

**無理ではありません。PHP 関数を使って、REST リクエストを飛ばし、
レスポンスを受け取るだけで良いのです。**

組み込みの PHP 関数 `simplexml_load_file()` を使って、REST リクエストを飛ばし、XML レスポンスを受け取ります。たとえば YouTube へのリクエストとレスポンスです。この関数は実際には、XML ドキュメントを PHP 内にオブジェクトとして読み込みます。このオブジェクトを使って、XML データを分解し、特定の情報を必要に応じて何でも引っ張り出することができます。ではこの関数が、オーウェンの YouTube ビデオリクエストに対してどのようなインパクトがあるでしょうか? 以下のコードをチェックしてみて下さい。このコードは YouTube の URL を保持する定数を作り、次に REST リクエストを `simplexml_load_file()` 関数を使って飛ばしています。

```
define('YOUTUBE_URL', 'http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first');

$xml = simplexml_load_file(YOUTUBE_URL);
```

厳密には不要ですが、一般に静的な URL を
定数に突っ込んでおくのは良いことです。
こうしておくことで、変更が必要になった場合、
どこから変えればよいかがすぐ分かります。

- ① YouTube ビデオ用のリクエストを作る。
- ② YouTube へビデオリクエストを投げる。
- ③ YouTube からビデオの情報を含むレスポンス
データを受け取る。
- ④ レスポンスデータを処理し、データを
HTML コードに整形する。

この 2 ステップが
絡みました!



オブジェクトとは何かを知らなく
ても、特に PHP のコンテキストに
おいてはアセる必要はありません。

PHP のオブジェクトというのは、特別のデータ型
で、データと関数とが 1 つの構造の中にパッケージ化されたものです。今知っておくべきことは、
XML データを PHP で処理するには、オブジェクト
を使うと非常に簡単になるということだけです。この
点についてはこの後、少しだけ学習します。

```

<xml version='1.0' encoding='UTF-8'?>
<feed xmlns='http://www.w3.org/2005/Atom'
      xmlns:gd='http://schemas.google.com/2005/atomExtensions'
      xmlns:openSearch='http://a9.com/-/spec/opensearchrss/1.0/'
      xmlns:georss='http://www.georss.org/georss'
      xmlns:media='http://search.yahoo.net/xmlns/mrss/'
      xmlns:patch='http://gdata.youtube.com/schemas/2007'
      xmlns:gfd='http://schemas.google.com/g/2005'
      id='http://gdata.youtube.com/feeds/api/users/alienabductedme/favorites'
      updated='2008-09-01T09:19:58.000-07:00'
      updatedPublished='2008-09-01T09:19:58.000-07:00'>
<category scheme='http://schemas.google.com/g/2005#kind' term='http://gdata.youtube.com/schemas/2007#video'/>
<title type='text'>Favorites of alienabductedme</title>
<link href='http://www.youtube.com/user/alienabductedme' rel='alternate' type='text/html' title='YouTube Channel - alienabductedme' type='application/atom+xml' href='http://gdata.youtube.com/feeds/api/users/alienabductedme/favorites'/>
<link href='http://schemas.google.com/g/2005#feed' type='application/atom+xml' rel='self' href='http://gdata.youtube.com/feeds/api/users/alienabductedme/favorites?start-index=1&max-results=25' type='application/atom+xml' />
<link href='http://gdata.youtube.com/feeds/api/users/alienabductedme/favorites?start-index=1&max-results=25' rel='next' type='application/atom+xml' />
<author>
  <name>alienabductedme</name>
  <uri>http://gdata.youtube.com/feeds/api/users/alienabductedme/uri</uri>
</author>
<generator version='2.0' uri='http://gdata.youtube.com/feeds/api/generator'>
<openSearch:totalResults>9</openSearch:totalResults>
<openSearch:startIndex>1</openSearch:startIndex>
<openSearch:itemsPerPage>25</openSearch:itemsPerPage>
</generator>
<entry>
  <id>http://gdata.youtube.com/feeds/api/users/gaspirtz/6ULbgf0vtk</id>
  <published>2008-09-01T09:19:58.000-07:00</published>
  <updated>2008-09-01T09:19:58.000-07:00</updated>
  <category scheme='http://gdata.youtube.com/schemas/2007/keywords.cat' term='sightings'/>
  <category scheme='http://gdata.youtube.com/schemas/2007/keywords.cat' term='ca'/>
  <category scheme='http://gdata.youtube.com/schemas/2007/keywords.cat' term='51'/>
  <category scheme='http://schemas.google.com/g/2005#kind' term='http://gdata.youtube.com/schemas/2007#video'/>
  <category scheme='http://gdata.youtube.com/schemas/2007/keywords.cat' term='area 51'/>
  <category scheme='http://gdata.youtube.com/schemas/2007/keywords.cat' term='alien'/>
  <category scheme='http://gdata.youtube.com/schemas/2007/categories.cat' term='travel' label='Travel & Events' />
  <category scheme='http://gdata.youtube.com/schemas/2007/keywords.cat' term='nevada' />
  <category scheme='http://gdata.youtube.com/schemas/2007/keywords.cat' term='nevada' />
  <category scheme='http://gdata.youtube.com/schemas/2007/keywords.cat' term='ufos' />
  <category scheme='http://gdata.youtube.com/schemas/2007/keywords.cat' term='sighting' />
  <title type='text'>UFO Sighting in Yosemite Park near Area 51</title>
  <content type='text'>I went on a trip to Yosemite Park in 2008. Yosemite Park is very close to the border between California and Nevada. In the evening, on my way back from driving down a winding road, when I saw a small ball of light high up in the sky, and a long twisted trail behind it. I video taped it for a few seconds, and then I grabbed my camera to take some photos, but the ball of light was still enough. Unfortunately it was too dark for the photo camera, so the pictures didn't come out. All I have is the video I took before and after I took the photos.</content>
  <link href='http://www.youtube.com/watch?v=_6ULbgf0vtk' rel='alternate' type='text/html' href='http://www.youtube.com/watch?v=_6ULbgf0vtk' type='application/atom+xml' href='http://gdata.youtube.com/feeds/api/videos/_6ULbgf0vtk' type='application/atom+xml' rel='related' type='application/atom+xml' href='http://gdata.youtube.com/feeds/api/users/gaspirtz/favorites/_6ULbgf0vtk' rel='self' type='application/atom+xml' href='http://gdata.youtube.com/feeds/api/users/gaspirtz/favorites'/>
  <author>
    <name>gaspirtz</name>
    <uri>http://gdata.youtube.com/feeds/api/users/gaspirtz/uri</uri>
  </author>
  <media:group>
    <media:content type='plain'>UFO Sighting in Yosemite Park near Area 51</media:title>
    <media:description type='plain'>I went on a trip to Yosemite Park in 2008. Yosemite Park is very close to the border between California and Nevada, and close to Area 51. In the evening, on my way back from driving down a winding road, when I saw a small ball of light high up in the sky, with a large halo of light around it, and a long twisted trail behind it. I video taped it for a few seconds, and then I grabbed my camera to take some photos, but the ball of light was still enough. Unfortunately it was too dark for the photo camera, so the pictures didn't come out. All I have is the video I took before and after I took the photos.</media:description>
    <media:content url='rtsp://rtsp2.youtube.com/ChdLDrh7YiwAEQnqVsh1r1_MVTEsARFfPgGDA==/0/0/0/video.3gp' type='video/3gp' medium='video' expression='full' duration='50' yt:format='1' />
    <media:content url='rtsp://rtsp2.youtube.com/ChdLDrh7YiwAEQnqVsh1r1_MVTEsARFfPgGDA==/0/0/0/video.3gp' type='video/3gp' medium='video' expression='full' duration='50' yt:format='1' />
    <media:player url='http://www.youtube.com/watch?v=_6ULbgf0vtk' />
    <media:thumbnail url='http://img.youtube.com/vi/_6ULbgf0vtk/0.jpg' height='130' width='130' time='00:00:25' />
    <media:thumbnail url='http://img.youtube.com/vi/_6ULbgf0vtk/1.jpg' height='97' width='130' time='00:00:12.500' />
    <media:thumbnail url='http://img.youtube.com/vi/_6ULbgf0vtk/3.jpg' height='97' width='130' time='00:00:37.500' />
    <media:thumbnail url='http://img.youtube.com/vi/_6ULbgf0vtk/0.jpg' height='240' width='320' time='00:00:25' />
  </media:group>
  <statistics viewCounts='1250356' favoriteCounts='1331' />
  <gd:rating min='1' max='5' numRaters='1648' average='4.17' />
  <gd:comments>...
  ...

```

これがsimplexml_load_file()関数が返してくるXMLファイルです。
リクエストされたビデオに対するYouTubeのXMLデータでできています。

おぞましい…問題が大きく
なってるわ！こんなわけのわからない
XMLデータ全部まとめてどうしろって
言うの？ PHPスクリプトで手に
負えるわけないじゃない。



いえ、そんなことがあります！ YouTubeが返してきたXMLコードは
実は見た目ほどおぞましくはないのです…単にどこを見ればよいかを
知っておけば良いだけです。

YouTubeはXMLで話します

YouTubeからのビデオレスポンスは、実際にはきれいな箱にパッケージされたDVDが玄関に配達されるというわけではありません。そうではなくて、XMLドキュメントであって、リクエストしたビデオについての詳細な情報が含まれています。ビデオそのものではありません。

```
<?xml version='1.0' encoding='UTF-8'?>
<feed xmlns='http://www.w3.org/2005/Atom'
      xmlns:openSearch='http://a9.com/-/spec/opensearchrss/1.0/'
      xmlns:gml='http://www.opengis.net/gml'
      xmlns:georss='http://www.georss.org/georss'
      xmlns:media='http://search.yahoo.com/mrss/'
      xmlns:batch='http://schemas.google.com/gdata/batch'
      xmlns:yt='http://gdata.youtube.com/schemas/2007'
      xmlns:gd='http://schemas.google.com/g/2005'
      <id>http://gdata.youtube.com/feeds/api/users/alienabductedme/favorites</id>
      <updated>2008-07-25T03:22:37.001Z</updated>
      <category scheme='http://schemas.google.com/g/2005#kind'
              term='http://gdata.youtube.com/schemas/2007#video'/>
      <title type='text'>Favorites of alienabductedme</title>
      ...
<entry>
  <id>http://gdata.youtube.com/feeds/api/videos/_6Uiqbqf0vtA</id>
  <published>2006-06-20T07:49:05.000-07:00</published>
  ...
  <media:group>
    <media:title type='plain'>UFO Sighting in Yosemite Park near Area 51</media:title>
    <media:description type='plain'>I went on a trip to Yosemite Park in 2002. Yosemite Park is very close to the border between California and Nevada, and close to Area 51...</media:description>
    <media:keywords>51, alien, aliens, area, ca, california, nevada, sighting, sightings, ufo</media:keywords>
    <yt:duration seconds='50' />
    <media:category label='Travel & Events' scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
    <media:content url='http://www.youtube.com/v/_6Uiqbqf0vtA' type='application/x-shockwave-flash' medium='video' isDefault='true' expression='full' duration='50' yt:format='5' />
    <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wIaEQnQvvSnbIKL_xMYDSANFEgGDA==/0/0/0/video.3gp' type='video/3gpp' medium='video' expression='full' duration='50' yt:format='1' />
    <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wIaEQnQvvSnbIKL_xYESARFEgGDA==/0/0/0/video.3gp' type='video/3gpp' medium='video' expression='full' duration='50' yt:format='6' />
    <media:player url='http://www.youtube.com/watch?v=_6Uiqbqf0vtA' />
    <media:thumbnail url='http://img.youtube.com/vi/_6Uiqbqf0vtA/2.jpg' height='97' width='130' time='00:00:25' />
    <media:thumbnail url='http://img.youtube.com/vi/_6Uiqbqf0vtA/1.jpg' height='97' width='130' time='00:00:12.500' />
    <media:thumbnail url='http://img.youtube.com/vi/_6Uiqbqf0vtA/3.jpg' height='97' width='130' time='00:00:37.500' />
    <media:thumbnail url='http://img.youtube.com/vi/_6Uiqbqf0vtA/0.jpg' height='240' width='320' time='00:00:25' />
  </media:group>
  <yt:statistics viewCount='2478159' favoriteCount='1897' />
  <gd:rating min='1' max='5' numRaters='1602' average='4.17' />
  <gd:comments>
    <gd:feedLink href='http://gdata.youtube.com/feeds/api/videos/_6Uiqbqf0vtA/comments' countHint='4426' />
  </gd:comments>
</entry>
<entry>
  <id>http://gdata.youtube.com/feeds/api/videos/XpNd-Dg6_zQ</id>
  <published>2006-11-19T16:44:43.000-08:00</published>
  ...
</entry>
</feed>
```

YouTubeはビデオリクエストに対して、XMLデータでレスポンスします。データにはビデオについて記述しています。

このXMLコードにはたくさん
の記述がありますが、注目すべきは、
個々のビデオは<entry>タグの
中に現れるという点です。



この<entry>タグがXMLレスポンステーブル
の中で、新しいビデオの開始を表します。



自分で考えてみよう

反対側のページで YouTube レスポンスの網掛け部分の XML コードをよく見て、次の質間に答えて下さい。もしかしたら一目見て思っていたよりも YouTube ビデオの XML フォーマットについて理解が深まるかも知れません！

1. ビデオのタイトルは何でしょうか?
2. ビデオに結び付いている3つのキーワードは何でしょうか?
3. ビデオの長さは何秒でしょうか?
4. ビデオは何という YouTube ビデオカテゴリに属しているでしょうか?
5. このビデオは何回閲覧されてでしょうか?
6. ユーザはこのビデオに平均でどのくらいの評価をつけているでしょうか?

自分で考えてみよう の答え

反対側のページで YouTube レスポンスの網掛け部分の XML コードをよく見て、次の質問に答えて下さい。もしかしたら一目見て思っていたよりも YouTube ビデオの XML フォーマットについて理解が深まるかも知れません！

```
<media:title type='plain'>UFO Sighting in Yosemite Park near Area 51</media:title>
```

1. ビデオのタイトルは何でしょうか? **UFO Sighting in Yosemite Park near Area 51**

```
<media:keywords>51, alien, aliens, area, ca, california, nevada, sighting, sightings, ufo</media:keywords>
```

2. ビデオに結び付いている3つのキーワードは何でしょうか? **51, aliens, nevada**

3. ビデオの長さは何秒でしょうか? **50**

```
<yt:duration seconds='50'>
```

XMLはいくつかの文字を特別なコードで
エスケープ(エンコード)します。
例えば \$amp; はドル記号 (\$) を表します。

```
<media:category label='Travel & Events'  
scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
```

4. ビデオは何という YouTube ビデオカテゴリに属しているでしょうか? **Travel & Events**

```
<yt:statistics viewCount='2478159' favoriteCount='1897' />
```

うわ、すごい閲覧
回数です…ほぼ250万!

5. このビデオは何回閲覧されてでしょうか? **2478159**

```
<gd:rating min='1' max='5' numRaters='1602' average='4.17'>
```

6. ユーザはこのビデオに平均でどのくらいの評価をつけているでしょうか? **4.17**



ううむ、ちょっとXMLタグに混乱してきた。
コロンで区切られた2つの名前があるけど…。タグを
まとめ上げる何かいい方法ってあるのかな?あと
ビデオカテゴリのところにある&とかいう
わけのわからないコードはどうなの?

特別なXMLコードがあって、名前空間と実体(エンティティ)というものをしています。これらはそれぞれタグをまとめ上げることと、特別な文字をエスケープする役割を担っています。

XMLタグを見たときに、コロンで区切られた2つの名前があったら、1つは名前空間といって、タグの集まりを論理的なグループにまとめ上げる方法を提供しています。名前空間の目的は、複数のXML語句が同じドキュメントに使われたときに、同じ名前のタグが衝突しないようにすることです。一例として、以下のXMLタグを考えてみます。

```
<title type='text'>Favorites of aliensabductedme</title>
```

```
<media:title type='plain'>UFO Sighting in Yosemite Park near Area 51</media:title>
```

**名前空間は XML
タグに名前をつけた
グループです。一方、
実体(エンティティ)は
XMLドキュメント中
で特別な文字を
エスケープするために
使われます。**

Yahoo!の名前空間がYouTubeの
XMLコードに現れるというのは不思議
に思えるかも知れません。これは単に
Yahoo!によって作られたXMLデータ
フォーマットにYouTubeが部分的に
依存しているためです。

2番目の<title>タグにmediaという名前空間がないと、同じXMLコードに両方が現れた場合、2つのタグを区別することができません。つまり名前空間とは、タグの「姓」と考へてもよいでしょう。これによりXMLドキュメントは全部「名」だけで、「名」にぶら下がっているタグが衝突してしまうことを避けることができます。YouTubeのレスポンスコードは、いくつかの異なる名前空間を使っています。これはいくつかの異なるXML言語を同時に使っていることを意味します。名前空間は、これらを明確に区別する手段を提供してくれます。

一意性を保証するために、XMLの名前空間は常にURLと関連付けられます。例えば、YouTubeのXMLデータで使われているmedia名前空間は、<feed>タグの中で、以下のように確立されています。

xmlns:media='http://search.yahoo.com/mrss/'

URLは実際にはWeb
ページではありません。
これは単に名前空間の
ための一意な識別子です。

YouTubeのXMLコードの中には、もう1つ奇妙なものがありました。&です。これはXMLにおいて特別な文字を参照する符号化の方法です。特別な文字には、&、<、>などがあります。これらはすべてXMLコードの中で特別な意味を持っています。以下は5つの既定義XML要素で、XMLコードの中をより深く掘り下げていくと、いずれ出会うことになるでしょう。

& = &

< = <

> = >

" = "

' = '

YouTube XML レスポンスを分解する

YouTube レスポンスの構造を理解してしまえば、必要なビデオデータを引っ張り出すのは、一発でできます。どのタグや属性がどんなデータを格納しているのかを理解することに加え、タグが相互にどのように関連しているのかを理解することも重要です。本章の最初の方で、RSS フィードを解析したことを思い出して下さい。XML ドキュメントは要素の階層構造と捉えることができます。これは YouTube ビデオレスポンスとして返ってきた XML データについても言えることなのです。

```

<entry>
  <id>http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA</id>
  <published>2006-06-20T07:49:05.000-07:00</published>
  ...
  <media:group>
    <media:title type='plain'>UFO Sighting in Yosemite Park near Area 51</media:title>
    <media:description type='plain'>I went on a trip to Yosemite Park in 2002. Yosemite Park is very close to the border between California and Nevada, and close to Area 51...</media:description>
    <media:keywords>51, alien, aliens, area, ca, california, nevada, sighting, sightings, ufo</media:keywords>
    <yt:duration seconds='50' /> ビデオの長さ、単位は秒です。
    <media:category label='Travel & Events' scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
    <media:content url='http://www.youtube.com/v/_6Uibqf0vtA' type='application/x-shockwave-flash' medium='video' isDefault='true' expression='full' duration='50' yt:format='5' />
    <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wlaEOnQvvSnbIKL_xMDYDSANFEGDAA=/0/0/0/video.3gp' type='video/3gpp' medium='video' expression='full' duration='50' yt:format='1' />
    <media:content url='rtsp://rtsp2.youtube.com/ChoLENy73wlaEOnQvvSnbIKL_xMYESARFEGDAA=/0/0/0/video.3gp' type='video/3gpp' medium='video' expression='full' duration='50' yt:format='6' />
    <media:player url='http://www.youtube.com/watch?v=_6Uibqf0vtA' /> ビデオの YouTube でのカテゴリです。
    <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/2.jpg' height='97' width='130' time='00:00:25' /> YouTube のビデオへのリンクです。
    <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/1.jpg' height='97' width='130' time='00:00:12.500' />
    <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/3.jpg' height='97' width='130' time='00:00:37.500' />
    <media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/0.jpg' height='240' width='320' time='00:00:25' />
  </media:group>
  <yt:statistics viewCount='2478159' favoriteCount='1897' /> プレビューのためのビデオのサムネイルイメージです。
  <gd:rating min='1' max='5' numRaters='1602' average='4.17' /> ビデオが閲覧された回数です。
  <gd:comments>
    <gd:feedLink href='http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA/comments' countHint='4426' />
  </gd:comments> 「gd」名前空間は Google データを意味します。各種データを表現するために Google で定義したタグが入っています。
</entry> YouTube というのは Google の一部なのです。

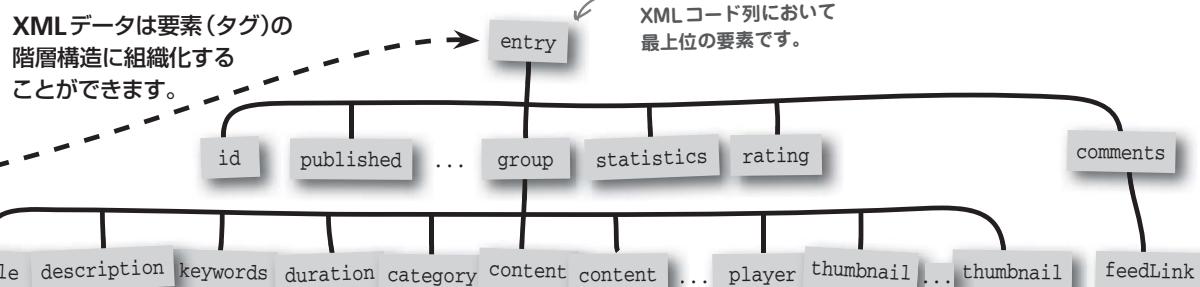
```

XML コードに埋め込まれているビデオデータを理解する上で重要なカギは、異なる名前空間が使われているということです。media 名前空間は特にビデオデータに関連する多くのタグにくっついています。一方 yt 名前空間は `<statistics>` タグにだけ使われています。最後に、コメントは `<comments>` タグに囲まれていて、gd 名前空間の配下にあります。これら名前空間は、PHP コードを書いて特定のタグやそのデータを探す際に、非常に重要な役割を演じます。

XMLのビデオデータを視覚化する

本章の最初の方でRSSコードについて見てきましたが、XMLドキュメントというのは、要素(タグ)が親子関係を持つ階層構造として視覚化できることが明らかになりました。この関係はXMLコードを処理し、そこの格納されたデータにアクセスしようとすると、ますます重要なものです。実際XMLドキュメントを見て、即座に要素間の関係を視覚化することができるようになれば、それはかけがえのないスキルと言って良いでしょう。1つ大事なことは、どんな要素でも別の要素に囲まれていれば、それは子であり、囲んでいる側は親であるということです。反対側のページのYouTubeビデオに対するXMLコードを見てきましたが、結果として次のように視覚化することができます。

要素とは、単に
XMLタグとそれが
持つデータを抽象化
する方法です。



要素の階層において特筆すべき点は、任意の要素から別の要素を探すことができるということです。階層の頂点からのパスをたどればよいのです。例えば、ビデオのタイトルを知りたいと思ったら、次のようなパスをたどることができます。



素朴な疑問に答えます

Q: なぜ名前空間などというものまで心配する必要があるのでしょうか?

A: なぜならXMLコードは一般に他の何かから作られ、それにもともと名前空間があったからです。このためXML要素にプログラム的にどうやってアクセスするかということに影響が出ます。この後すぐ分りますが、名前空間は要素に関連付けられているので、XMLデータを処理するPHPコードを書く際に、要素を見つけるのにモロに影響します。つまり名前空間を考慮に入れコードを書かなければ、ある要素のデータを取ってくことができないです。

Q: タグが名前空間の一部であるということを知るにはどうすればいいのでしょうか?

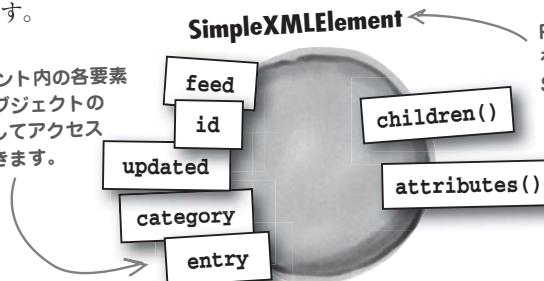
A: デフォルトの名前空間というものがあって、タグのコードに明示的に現れないものもありますが、通常、名前空間はタグ名のところに直にあります。タグ名は単に<title>ではなく、<media:title>という具合にコード化されます。コロンの左側の名前は常に名前空間です。

オブジェクトで XML データにアクセスする

PHPでXMLデータを処理するには色々なやり方がありますが、オブジェクトで処理するというのは非常によい方法です。オブジェクトとは、PHPの特別なデータ型であって、データと関数とを1つの構造に組み合わせたものです。ではこれを使ってXMLをどう処理すれば良いのでしょうか？ XMLドキュメント内で要素の全階層構造は1つのオブジェクトという変数に含まれています。このオブジェクトをデータに分解して、個々の要素にアクセスすることができます。オブジェクトにはメソッドも含まれています。メソッドとはオブジェクトと結合している形式の関数のことです。これを使えばオブジェクトのデータをさらに操作することができます。XMLデータを含んでいるようなオブジェクトでは、メソッドが、ある要素の子の要素と属性すべてへのアクセスを可能してくれます。

オブジェクトとは
PHPの特別な
データ型で、
データと関数とを
組み合わせた
ものです。

XMLドキュメント内の各要素
へは、XMLオブジェクトの
プロパティとしてアクセス
することができます。



PHPオブジェクトの型は、XMLデータ
を操作し格納するために使いますが、
SimpleXMLElementです。

SimpleXMLElementオブジェクトには、
要素についての細かい情報を、例えばその
要素の子の要素や属性を、調べるためのメ
ソッドがあります。

すでにオーウェンのUFOによる誘拐に関するYouTubeのキーワード検索に対するXMLオブジェクトを作る方法はみてきました。

この関数を使うにはPHPの
バージョン5以上が必要です。
お忘れなく。

```
define('YOUTUBE_URL', 'http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first');
$xml = simplexml_load_file(YOUTUBE_URL);
```

このコードを実行すると\$xmlという名前の変数に、XMLのYouTubeビデオデータがすべて、PHPオブジェクトとしてパッケージ化されて入ります。データにアクセスするには、オブジェクトのプロパティを使います。プロパティとはオブジェクトに格納された個々のデータのことです。各プロパティは、XML要素に対応しています。以下の例を見て下さい。ここではドキュメント内のすべてのentry要素にアクセスしています。

この関数は
PHPオブジェクトを作ります。
型はSimpleXMLElementで、
YouTubeビデオレスポンス
のすべてのXMLデータを
含んでいます。

```
$entries = $xml->entry;
```

-> 演算子はオブジェクト
内のプロパティにアクセス
する際に使います。

要素の名前(entry)を指定することで、
XMLデータの中にあるすべての要素を
取ってくることができます。

すべてのビデオ
エントリは
\$entries配列に
突っ込まれます。

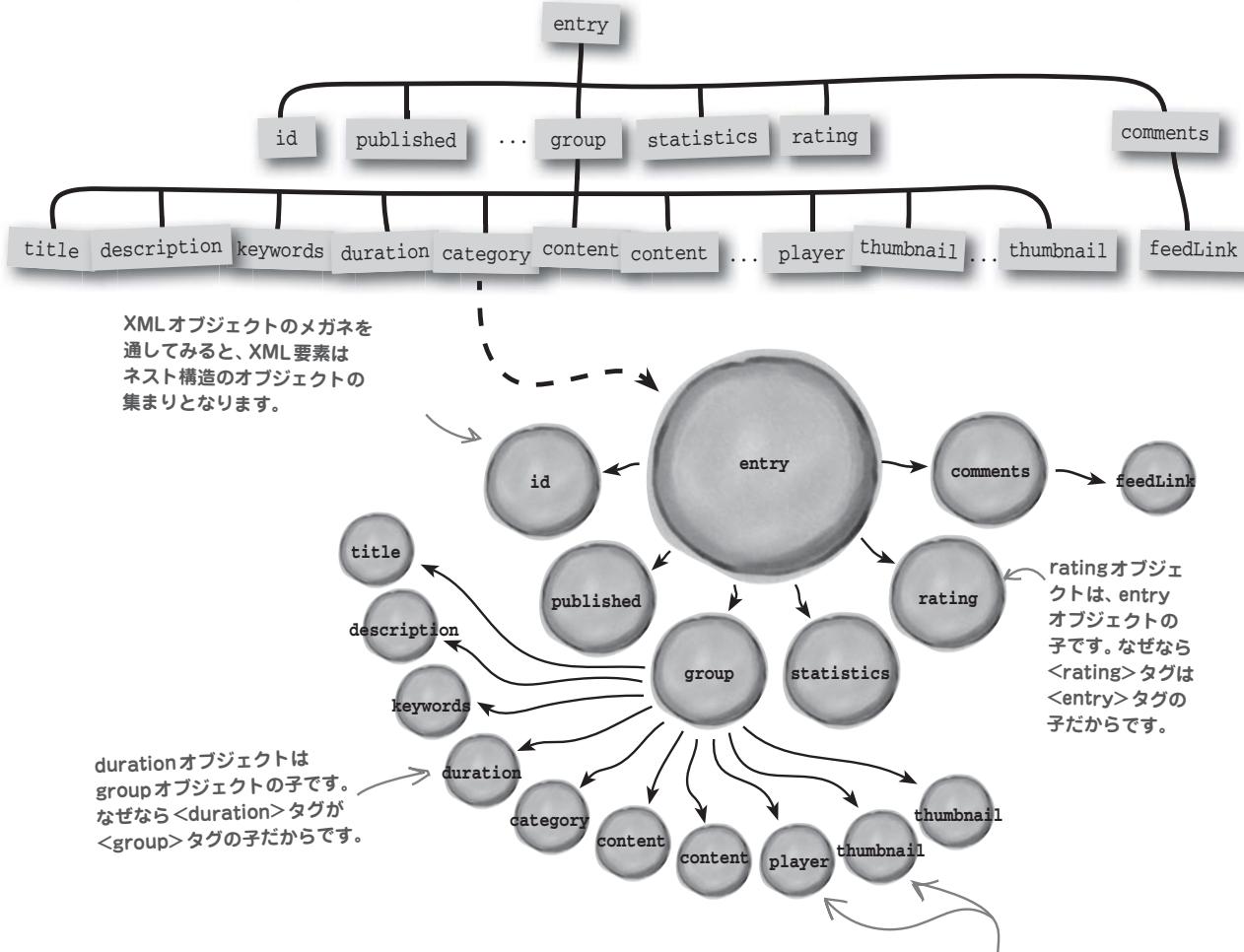
このコードでは、XMLデータのすべてのentry要素にプロパティを使ってアクセスしています。データには複数のentry要素がありますから、\$entries変数はオブジェクトの配列ということになって、個々のビデオエントリにアクセスすることができるようになります。ここで配列を扱うことになるわけですから、各ビデオの<entry>タグへは、配列の添字でアクセスすることになります。例えば、ドキュメントの最初の<entry>タグは、配列の最初の項であり、2番目のタグは配列の2番目の項である、といった具合になります。



\$entries

XML要素からPHPオブジェクトへ

XMLデータとPHPオブジェクトまで来たら、オブジェクトの集まりを処理する準備はできました。XMLドキュメントを要素の階層構造で視覚化することについて覚えていますか？当然ながら、同じ階層構造がPHPのオブジェクトの集まりにも適用されます。見てみましょう。



この要素の階層構造とオブジェクトの集まりとの対比により、XMLデータをPHPで掘り起こすにはどうすればよいかについての基本的理解が得られます。個々のXMLデータ間の関係を心に留めることで、データ間をたどるコードを書くことが可能となります。そうすればXMLドキュメントの奥深くにある特定のタグや属性に格納されたコンテンツを取り出すことができるわけです。

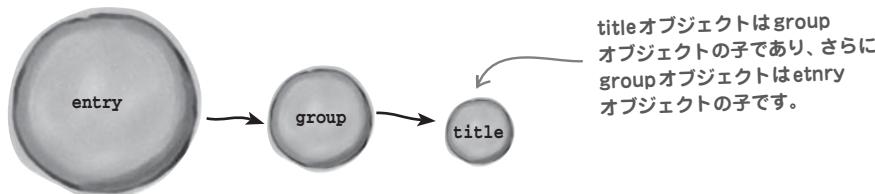
XMLデータをオブジェクトで掘り起こす

オーウェンに戻ると、ゴールはYouTubeのXMLレスポンスの一部として返ってきたビデオに関するいくつかの情報を取り出すことです。すでにXMLデータを取りだしてPHPオブジェクトに格納する方法は知っています。`simplexml_load_file()`関数を使えば良いのです。しかし、大部分の重要なデータは、このデータのさらに奥の方に入っています。オブジェクトの集まりをたどるにはどうすればよいでしょうか？その答えは、`->`演算子です。この演算子はオブジェクトのプロパティやメソッドを参照する際に使います。XMLオブジェクトの場合で言うと、`->`演算子はそれぞれの子のオブジェクトにアクセスできます。つまり以下のコードは、`$entry`という名前の変数に格納されたビデオエントリのタイトルを表示します。

`echo $entry->group->title;`

ここで`->`演算子を使ってネストした子のオブジェクトに掘り下げて、`title`オブジェクトにアクセスします。

このコードは`title`、`group`、`entry`の3つのオブジェクトに極めて強く依存しています。これらはそれぞれ、左から右へ親子関係を形成しているのです。



`->`演算子は、親オブジェクトから子オブジェクトを参照する。つまり`title`は`group`の子で、さらに`group`は`entry`の子です。`->`演算子は、プロパティとメソッドのどちらにアクセスする際にも使えるということを覚えておいて下さい。特に便利なメソッドは、`attributes()`メソッドと言って、特定の要素に対してXML属性の値を引っ抜いてくることができます。

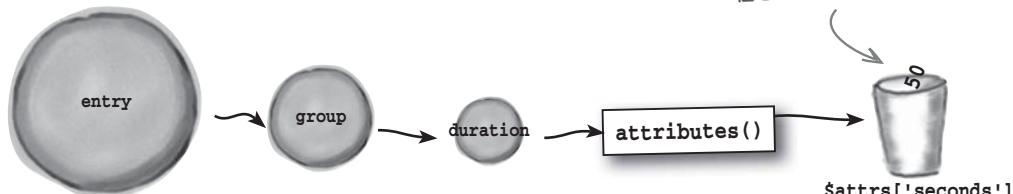
```
$attrs = $entry->group->duration->attributes();
```

`attributes()`メソッドで、オブジェクト(要素)の属性の配列を得ることができます。

`echo $attrs['seconds'];`

上のコードは`duration`要素まで掘り下げて、そこですべての属性を捕まえて、`$attrs`変数に格納しています。この変数はすべての属性の配列となります。次に`seconds`属性の値を、配列から取ってきています。

属性の名前を配列のキー(添字)として指定することで、特定の属性値を取ってくることができます。



名前空間なしではダメです！

実は反対側のページでXMLデータにオブジェクトを使ってアクセスしているコードにはちょっと問題があったのです。名前空間を使っていたからです。覚えていると思いますが、名前空間というのはタグの「姓」のような働きをして、タグを意味のある集まりにまとめ上げる働きをします。YouTubeのレスポンスにおいては、<duration>タグは、実際には<yt:duration>という風にコード化されていますし、ビデオのタイトルも<media:title>という風にコード化されているのです。単に<title>ではありません。要素が名前空間に関連付けられている場合、PHPコードから単にタグ名で参照することはできません。その代わりに、まず名前空間によって、タグを分離します。このためはchildren()メソッドを親オブジェクトに対して呼び出す必要があります。

```
$media = $entry->children ('http://search.yahoo.com/mrss/');
```

上のコードは、ビデオエントリのすべての子のオブジェクトを取ってきますが、名前空間がhttp://search.yahoo.com/mrss/のものだけです。ただしこれは名前空間のURLであって、名前空間そのものではありません。このURLはXMLコードドキュメントの先頭にある<feed>タグに書いてあります。ここを見れば使われている名前空間がすべて分かります。

```
<feed xmlns='http://www.w3.org/2005/Atom'  
      xmlns:openSearch='http://a9.com/-/spec/opensearchrss/1.0/'  
      xmlns:gml='http://www.opengis.net/gml'  
      xmlns:georss='http://www.georss.org/georss'  
      xmlns:media='http://search.yahoo.com/mrss/'  
      xmlns:batch='http://schemas.google.com/gdata/batch'  
      xmlns:yt='http://gdata.youtube.com/schemas/2007'  
      xmlns:gd='http://schemas.google.com/g/2005'>
```

このコードによって、それぞれの名前空間がどのURLと結び付いているのかが分かります。さらに言えば、mediaやytといった名前空間を、ドキュメントの中で使用する際にどのように指定されるかを示しています。これら2つの名前空間に関連したタグを見つけるために知っておくべきことはこれですべてです。

children()メソッドを親要素に対して呼び出すことで、一度、特定の名前空間に対して子の要素を分離してしまえば、子のオブジェクトへのアクセスを->演算子を使って再開することができます。例えば、以下のコードは<media:group>タグからビデオのタイトルを取ってきてくれます。

```
$title = $media->group->title;
```

<title>タグは<media:group>
タグの子です。

名前空間がある
ので、XMLデータ
の中の要素に
アクセスするのは
少し面倒に
なります。

children()メソッドは、
指定された名前空間の
中で子の要素をすべて
含んだ配列を返します。

children()メソッド
を使うことで、名前
空間の結び付いた
すべての要素を
分離できます。



自分で考えてみよう

名前空間の情報と上記PHPコードを使って、以下のPHPコードを完成させて下さい。
ビデオの再生時間(duration : 秒単位)を取ってきます。

```
$yt = $media->children('.....');  
$attrs = .....;  
echo $attrs['.....'];
```

自分で考えてみよう の答え

```
$yt = $media->children('http://gdata.youtube.com/schemas/2007');
$attrs = $yt->duration->attributes();
echo $attrs['seconds'];
属性の名前をキーとして
使って、属性配列(の要素)にアクセスします。
```

名前空間の情報と上記PHPコードを使って、以下のPHPコードを完成させて下さい。
ビデオの再生時間(duration:秒単位)を取ってきます。

これは名前空間を指定するためのURLでドキュメントの先頭で<feed>タグにリストされていたものです。

<yt:duration>タグの属性をすべて取ってきます。

素朴な疑問に答えます

Q: オブジェクトと配列とはどのように違うのでしょうか?配列もデータの集まりを格納できるのではないか?

A: その通りです。オブジェクトも配列も実際には非常によく似ています。しかし1つ非常に大きな違いがあります。オブジェクトにはメソッドという形で、実行可能なコードを組み入れることができます。メソッドというのは、関数とほとんど同じものです。違いは、メソッドがオブジェクトと結び付いているということだけです。そしてオブジェクトの格納されたデータに対してのみ動作するという前提で設計されています。配列は関連するデータを格納するのみで、メソッドという概念はありません。なお、配列では、添字またはキーをシカクカッコ([])の内側に指定することで配列要素と結びつけられますが、オブジェクトでは->演算子に名前を指定することで、オブジェクトのプロパティやメソッドと結びつけられます。

Q: オブジェクトというのは正確に言うと何ですか?普通の変数のようなものでしょうか?

A: その通りです。オブジェクトというのは、PHPの変数とほとんど同じものです。単に、もっと複雑なデータを格納することができるというだけです。つまり単に文字列テキストや数値を入れるだけでなく、オブジェクトなら文字列や数値等を組み合わせて入れておくことができます。考え方としては、関連するデータとそれに作用する関数などを組み合わせることによって、アプリケーション全体の設計やコーディングを、もっと論理的にするというものです。

Q: ではオブジェクトはXMLデータを処理する上でどのようにメタタイののでしょうか?

A: オブジェクトはXMLデータを処理する上で役立つのは、XMLドキュメントの要素の階層構造を、ネストした子のオブジェクトを使ってモデル化できるためです。このアプローチの利点は、->演算子を使って子のオブジェクトをたどることで、必要などんなデータにでもアクセスできることです。

Q: ->演算子というのは、オブジェクトのプロパティにアクセスするものと思っていたら、どうすれば子のオブジェクトにアクセスできるのでしょうか?

A: 実は、PHPでXMLオブジェクトを取り扱う場合、子のオブジェクトというのは、実際にはプロパティとして格納されています。つまり->演算子を使って子のオブジェクトにアクセスするというのは、実際にはプロパティにアクセスしているだけなのです。SimpleXMLElementオブジェクトのおかげで、このようなことが可能となっています。

Q: ちょっと待って下さい。SimpleXMLElementオブジェクトって何ですか?

A: PHPのすべてのオブジェクトは何らかのデータ型を持っています。つまり「オブジェクト」というのは一般的な用語なのです。オブジェクトを作るというのは、特定の型のオブジェクトを作るという意味です。ここで特定の型というのは特定のタスクを遂行するために設計されたものと言います。XMLの場合で言うと、オブジェクトの型は、SimpleXMLElementで、これはsimplexml_load_file()関数により自動的に返されます。言い換えば、simplexml_load_file()関数を呼び出すと、SimpleXMLElement型のオブジェクトが作られて返ってくるのです。

Q: SimpleXMLElementについて何を知っておけばよいでしょうか?

A: 驚くかもしれません、大してありません。知っておくべき主なことは、XMLドキュメントの要素をプロパティとして表現してくれるということであり、このプロパティにより、子のオブジェクトにたどり着けるということです。そして子のオブジェクト自身もSimpleXMLElementの実体(インスタンス)となるということです。SimpleXMLElementオブジェクトは、要素内のデータにアクセスするためのメソッドも持っています。例えばchildren()やattributes()がそうです。

ファング目撃情報上昇中

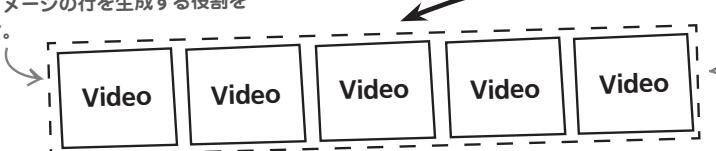
オーウェンはXMLやらYouTubeとの通信方法の勉強に忙しい中、ファングも忙しかったのです。いくつものビデオ目撃情報が、見つかっています。ファングは小さいヤツと一緒にいるようなので、UFOによる誘拐解決ツアーも終わりに向かっていることは明らかです。オーウェンのYouTubeスクリプトはもうすぐ完成しますから、「UFOに誘拐された！」ホームページにいくつかのビデオを載せて、迷子の犬を見つけることができることでしょう。



ビデオを見やすく配置する

`youtube.php` スクリプトを作る上で考慮すべきことは、「UFOに誘拐された！」のメインページ`index.php`にインクルードされるということです。これは`youtube.php`スクリプトが、ビデオリクエストを発行し、XMLレスポンスを処理し、個々のビデオを配置する、という意味です。ビデオはHTMLで表示してメインページにすでに表示されているUFOによる誘拐レポートと共存していなければなりません。これを同時に達成するために、ビデオをページの下部に横にアレンジするというはどうでしょう。

`youtube.php`スクリプトは、ビデオのサムネイルイメージの行を生成する役割を担っています。



これらはYouTubeにアクセスして動的に取ってきたXMLデータです。



5つ位のビデオサムネイルを横に配置するのが、スペース的にちょうど良いでしょう。

`youtube.php`スクリプトをインクルードして、ビデオをUFOによる誘拐レポートのすぐ下に配置します。

日付	報告者	誘拐された時間	宇宙人の様子	ファンダはいましたか？
2009-08-10	市川 敏	3時間	満月くらいの大きさの船でした。	no
2009-07-11	斎藤 友世	45分	頭がでかくて手足はひょろひょろでした。	yes
2009-07-05	白田 敏朗	2時間	空にとても明るいUFOが少なくとも2~3機いたようです。	yes
2009-06-21	深澤 麻子	ほぼ1週間	不格好で変な奴で、リズム感ゼロ。	no
2009-05-11	清水 裕子	1日	緑色で触覚が6つありました。	yes

■クリックすると誘拐ニュースフィードをシンケーションします。

ここがビデオサムネイルを並べて表示するにはちょうど良い場所です。閲覧者は簡単にアクセスできます。

ビデオをメインページに横に配置することで、UFOによる誘拐レポートの存在感をあまり損なうことありません。また今問題にしているのは、ビデオサムネイルの配置であって、ビデオ自体ではありません。そこでユーザがサムネイルをクリックすると、YouTubeに飛んで実際のビデオを観ることができるようにする必要があります。複数のビデオを「UFOに誘拐された！」のメインページに取り込んで表示しようとするのは、画面領域を食い過ぎるので現実的ではないでしょう。

ビデオデータを表示用に配置する

ビデオサムネイルのイメージは確かに、ビデオが観るに値するかどうかを判断するための重要な情報ではあります。しかし、オーウェンのYouTubeスクリプトのための有用なデータであるに過ぎません。例えば、ビデオのタイトルだって、ビデオの特徴を表現する何らかの重要な情報を持っているかもしれません。それこそ犬というキーワードを含んでいるかも知れないので。ビデオの再生時間も有用な情報かもしれません。それに、当然ですがビデオのURLが必要で、YouTubeにリンクしてユーザがビデオサムネイルをクリックしたら実際にビデオを閲覧できるようにしなければなりません。結論として、以下の情報をYouTubeレスポンスのXMLデータから抽出してくる必要があります。

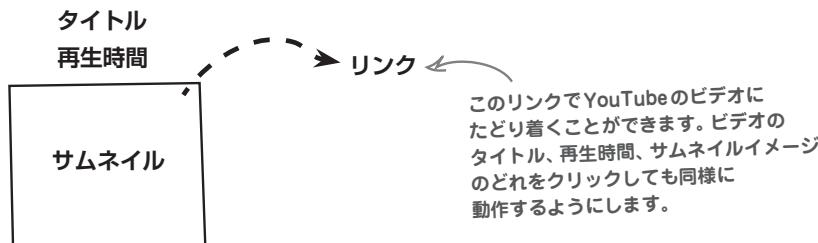
タイトル 再生時間

サムネイル

リンク

YouTubeビデオをWebページに貼り付けるには、いくつかのデータが必要です。

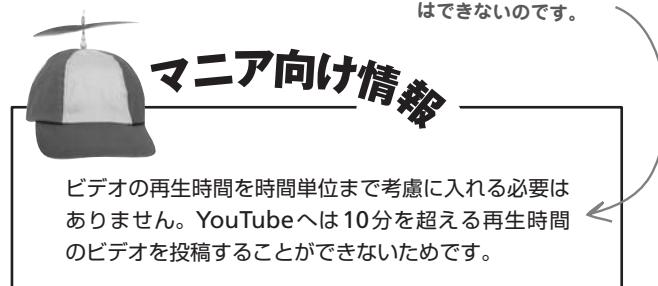
このデータをベースとして、ビデオを横に並べて表示するHTMLコードを作ることができます。実際には行内の各ビデオは次のような感じになります。



YouTubeのレスポンスデータで、ビデオの再生時間は、秒単位で`<yt:duration>`タグの属性として規定されています。しかし、残念ながらていの人は秒で全再生時間を教えてもらっても嬉しいでしよう。時間というのは分と秒で指定されることに慣れているからです。例えば、330秒とは5分30秒のビデオであるとすぐにはわからないでしよう。時間の長さを計算して意味のある値に変更してあげる必要があります。これがわかれれば、先へ進んでユーザのために計算を実行し、ビデオの再生時間を表示する場合は、秒から分と秒へ変換するというのは良いことだと理解できるはずです。

つまり、YouTubeのディレクターアカウントでない限り、10分を超える再生時間のビデオを投稿することはできないのです。

再生時間 → 330秒
→ 5分30秒
より直感的で、ユーザに分かりやすい表記です。





youtube.phpスクリプトはPHPコードでUFOによる誘拐のYouTubeビデオを検索し、マッチした上位5件を取ってきます。取ってきましたらビデオのサムネイルイメージを横一列に表示し、YouTube上の実際のビデオへのリンクをつけます。スクリプト中の抜けているコードを埋めて下さい。反対側のページにあるYouTubeのXMLビデオレスポンスデータを参考にして下さい。

```
<?php

define('YOUTUBE_URL', 'http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first');
define('NUM_VIDEOS', 5);

// Read the XML data into an object XML データを読み込みオブジェクトへ格納
$xml = .....(YOUTUBE_URL);

$num_videos_found = count(.....);
if ($num_videos_found > 0) {
    echo '<table><tr>';
    for ($i = 0; $i < min($num_videos_found, NUM_VIDEOS); $i++) {
        // Get the title タイトルの取得
        $entry = $xml->entry[$i];
        $media = $entry->children('http://search.yahoo.com/mrss/');
        $title = $media->group->....;
        // 再生時間を分秒単位で取得し、整形
        // Get the duration in minutes and seconds, and then format it
        $yt = $media->children('http://gdata.youtube.com/schemas/2007');
        $attrs = $yt->duration->attributes();
        $length_min = floor($attrs['.....'] / 60);
        $length_sec = $attrs['.....'] % 60;
        $length_formatted = $length_min . '分' . $length_sec . '秒';
        // Get the video URL ビデオのURLを取得
        $attrs = $media->group->player->.....();
        $video_url = $attrs['url'];
    }
}
```

```

// Get the thumbnail image URL サムネイルイメージのURLを取得
$attrs = $media->.....->thumbnail[0]->attributes();
$thumbnail_url = $attrs['url'];

// Display the results for this entry このエントリの結果を表示
echo '<td style="vertical-align:bottom; text-align:center" width="'.(100 / NUM_VIDEOS).' .
'%'><a href="'.$video_url.'" . .... . '<br /><span style="font-size:smaller">' .
$length_formatted.'</span><br /></a></td>';
}

echo '</tr></table>';

}

else {
    echo '<p>ビデオが見つかりませんでした。</p>';
}

?>

```

抜けているPHPコードを
書く上で、この例にある
XMLコードを参考に
して下さい。

...

```

<entry>
<id>http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA</id>
<published>2006-06-20T07:49:05.000-07:00</published>
...
<media:group>
<media:title type='plain'>UFO Sighting in Yosemite Park near Area 51</media:title>
<media:description type='plain'>I went on a trip to Yosemite Park in 2002. Yosemite Park is very
close to the border between California and Nevada, and close to Area 51...</media:description>
<media:keywords>51, alien, aliens, area, ca, california, nevada, sighting, sightings,
ufo</media:keywords>
<yt:duration seconds='50' /> ← ビデオの再生時間
<media:category label='Travel & Events'
scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
<media:content url='http://www.youtube.com/v/_6Uibqf0vtA' type='application/x-shockwave-flash'
medium='video' isDefault='true' expression='full' duration='50' yt:format='5' />
<media:content url='rtsp://rtsp2.youtube.com/ChoLENY73wIaEQnQvSnbIK1_xMYDSANFEGGDA==/0/0/0/video.3gp'
type='video/3gp' medium='video' expression='full' duration='50' yt:format='1' />
<media:content url='rtsp://rtsp2.youtube.com/ChoLENY73wIaEQnQvSnbIK1_xMYESARFEGGDA==/0/0/0/video.3gp'
type='video/3gp' medium='video' expression='full' duration='50' yt:format='6' />
<media:player url='http://www.youtube.com/watch?v=_6Uibqf0vtA' /> ← YouTube上のビデオの
リンクURLです。
<media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/2.jpg' height='97' width='130'
time='00:00:25' />
<media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/1.jpg' height='97' width='130'
time='00:00:12.500' />
<media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/3.jpg' height='97' width='130'
time='00:00:37.500' />
<media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/0.jpg' height='240' width='320'
time='00:00:25' />
</media:group>
<yt:statistics viewCount='2478159' favoriteCount='1897' />
<gd:rating min='1' max='5' numRaters='1602' average='4.17' />
<gd:comments>
<gd:feedLink href='http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA/comments'
countHint='4426' />
</gd:comments>
</entry>
<entry>
...
</entry>
...

```

ビデオのサムネイル
イメージ(プレビュー)の
URLです。



エクササイズ の答え

`youtube.php`スクリプトはPHPコードでUFOによる誘拐のYouTubeビデオを検索し、マッチした上位5件を取ってきます。取ってきましたらビデオのサムネイルイメージを横一列に表示し、YouTube上の実際のビデオへのリンクをつけます。スクリプト中の抜けているコードを埋めて下さい。反対側のページにあるYouTubeのXMLビデオレスポンスデータを参考にして下さい。

```
<?php
define('YOUTUBE_URL', 'http://gdata.youtube.com/feeds/api/videos/-/alien/abduction/head/first');

define('NUM_VIDEOS', 5); // 表示するビデオの数を定数にぶち込みます。

// Read the XML data into an object
$xml = simplexml_load_file(YOUTUBE_URL); // simplexml_load_file() 関数を使って、YouTubeからXMLデータをリクエストします。

$num_videos_found = count($xml->entry); // 実際にYouTubeから返ってきたビデオの数をチェックします。<entry>タグの数を数えれば分かるのです。
if ($num_videos_found > 0) {
    echo '<table><tr>';

    for ($i = 0; $i < min($num_videos_found, NUM_VIDEOS); $i++) { // 一度に1エントリずつビデオデータでループします。
        // Get the title
        $entry = $xml->entry[$i];
        $media = $entry->children('http://search.yahoo.com/mrss/');

        $title = $media->group->title; // このエントリのすべての子を捕まえてきます。全部Yahoo!のメディア名前空間mediaにあります。
        // ビデオエントリからタイトルを取ってきます。これは<media:title>タグに入っています。
        // Get the duration in minutes and seconds, and then format it
        $yt = $media->children('http://gdata.youtube.com/schemas/2007');
        $attrs = $yt->duration->attributes(); // 今度は、このエントリのすべての子を捕まえてきます。全部YouTubeの名前空間ytにあります。
        $length_min = floor($attrs['seconds'] / 60); // ビデオの再生時間は、秒単位で<yt:duration>タグから得られます。
        $length_sec = $attrs['seconds'] % 60; // そうしたら、これを分単位に変換します。
        $length_formatted = $length_min . '分' . $length_sec . '秒';

        // Get the video URL
        $attrs = $media->group->player->attributes();
        $video_url = $attrs['url']; // ビデオへのリンク(URL)を<media:player>タグのurl属性から取ってきます。
    }
}
```

```

// Get the thumbnail image URL
$attrs = $media->...->thumbnail[0]->attributes();
$thumbnail_url = $attrs['url'];

// Display the results for this entry
echo '<td style="vertical-align:bottom; text-align:center" width="' . (100 / NUM_VIDEOS) . "
    '%><a href="' . $video_url . '">' . $title . '<br /><span style="font-size:smaller">' .
    $length_formatted . '</span><br /></a></td>';
}

echo '</tr></table>';
}
else {
    echo '<p>ビデオが見つかりませんでした。</p>';
}
?

```

最初のサムネイルイメージの URL を <media:thumbnail> タグの url 属性から取ってきます。

ビデオの結果をフォーマットして、テーブル要素にビデオのタイトル、再生時間、サムネイルイメージを格納します。

- ① YouTubeビデオに対するリクエストを組み立てる。
- ② できあがったリクエストをYouTubeに投げる。
- ③ YouTubeのレスポンスデータを受け取る。
通常ビデオに関する情報が入っている。
- ④ レスポンスデータを処理し、HTMLコードにフォーマットする。

```

...
<entry>
<id>http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA</id>
<published>2006-06-20T07:49:05.000-07:00</published>
...
<media:group>
<media:title type='plain'>UFO Sighting in Yosemite Park near Area 51</media:title>
<media:description type='plain'>I went on a trip to Yosemite Park in 2002. Yosemite Park is very close to the border between California and Nevada, and close to Area 51...</media:description>
<media:keywords>51, alien, aliens, area, ca, california, nevada, sighting, sightings, ufo</media:keywords>
<yt:duration seconds='50' /> ビデオの再生時間  
(長さ : 秒単位)です。
<media:category label='Travel & Events' scheme='http://gdata.youtube.com/schemas/2007/categories.cat'>Travel</media:category>
<media:content url='http://www.youtube.com/v/_6Uibqf0vtA' type='application/x-shockwave-flash' medium='video' isDefault='true' expression='full' duration='50' yt:format='5' />
<media:content url='rtsp://rtsp2.youtube.com/ChoLEnY73wlaEQnQvvsNbiK1_xMYDSANFEGGDA==/0/0/0/video.3gp' type='video/3gp' medium='video' expression='full' duration='50' yt:format='1' />
<media:content url='rtsp://rtsp2.youtube.com/ChoLEnY73wlaEQnQvvsNbiK1_xMYESEARFEGGDA==/0/0/0/video.3gp' type='video/3gp' medium='video' expression='full' duration='50' yt:format='6' />
<media:player url='http://www.youtube.com/watch?v=_6Uibqf0vtA' /> YouTube 上のビデオのリンク URL です。
<media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/2.jpg' height='97' width='130' time='00:00:25' />
<media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/1.jpg' height='97' width='130' time='00:00:12.500' />
<media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/3.jpg' height='97' width='130' time='00:00:37.500' />
<media:thumbnail url='http://img.youtube.com/vi/_6Uibqf0vtA/0.jpg' height='240' width='320' time='00:00:25' />
</media:group>
<yt:statistics viewCount='2478159' favoriteCount='1897' />
<gd:rating min='1' max='5' numRaters='1602' average='4.17' />
<gd:comments>
<gd:feedLink href='http://gdata.youtube.com/feeds/api/videos/_6Uibqf0vtA/comments' countHint='4426' />
</gd:comments>
</entry>
<entry>
...
</entry>
...

```

ビデオのサムネイルイメージ(プレビュー)の URL です。



試運転

「UFOに誘拐された！」サイトにYouTubeスクリプトを追加します。

新しくテキストファイルを作り、youtube.phpという名前にします。オーウェンのYouTubeスクリプトのコードを直前の2ページの通りに打ち込みます（またはスクリプトをサイト <http://www.oreilly.co.jp/books/9784873114446/> からダウンロードして下さい）。さらに、このスクリプトをindex.phpスクリプトに組み込んで、「UFOに誘拐された！」サイトのメインページをYouTubeビデオへ向ける必要があります。以下の2行のコードを組み込めば達成できます。

```
echo '<h4>最近更新された誘拐に関するビデオ:</h4>';
require_once('youtube.php');
```

できあがったスクリプトをWebサーバにアップロードしたら、index.phpをWebブラウザで開いて下さい。ページ下部にUFOによる誘拐に関連したYouTubeビデオのリンク行が動的に表示されているはずです。

これでファンが
どこにいるのか
分かると思うな…

youtube.php
スクリプトを
メインページに
インクルードする
ことが、やるべき
ことのすべて
です。これで
UFOによる誘拐
のビデオの行を
追加したことになります。

YouTubeビデオの
おかげで、オーウェン
はファンの居場所
をかなり特定でき
ます。

UFOに誘拐された！？

ようこそ当サイトへ。宇宙人と遭遇したことがありますか？誘拐体験はありますか？私の飼い犬ファンが見ませんでしたか？誘拐レポートをお願いします！

最近更新された誘拐レポート：

2009-08-10 : 市川 徹	誘拐されていた時間： 宇宙人の様子： 満月くらいの大きさの船でした。	ファンがいましたか？ no
2009-07-11 : 斎藤 友世	誘拐されていた時間： 宇宙人の様子： 頭がでかくて手足はひょろひょろでした。	ファンがいましたか？ yes
2009-07-05 : 白田 敏朗	誘拐されていた時間： 宇宙人の様子： 空にとても明るいUFOが少なくとも2~3機いたようです。	ファンがいましたか？ yes
2009-06-21 : 深沢 麻子	誘拐されていた時間： 宇宙人の様子： 不格好で変な奴で、リズム感ゼロ。	ファンがいましたか？ no
2009-05-11 : 清水 裕子	誘拐されていた時間： 宇宙人の様子： 緑色で触覚が6つありました。	ファンがいましたか？ yes

■クリックすると誘拐ニュースフィードをシンジケーションします。

最近更新された誘拐に関するビデオ：

Dog Rides in UFO Hovering Near San Francisco!
0分11秒



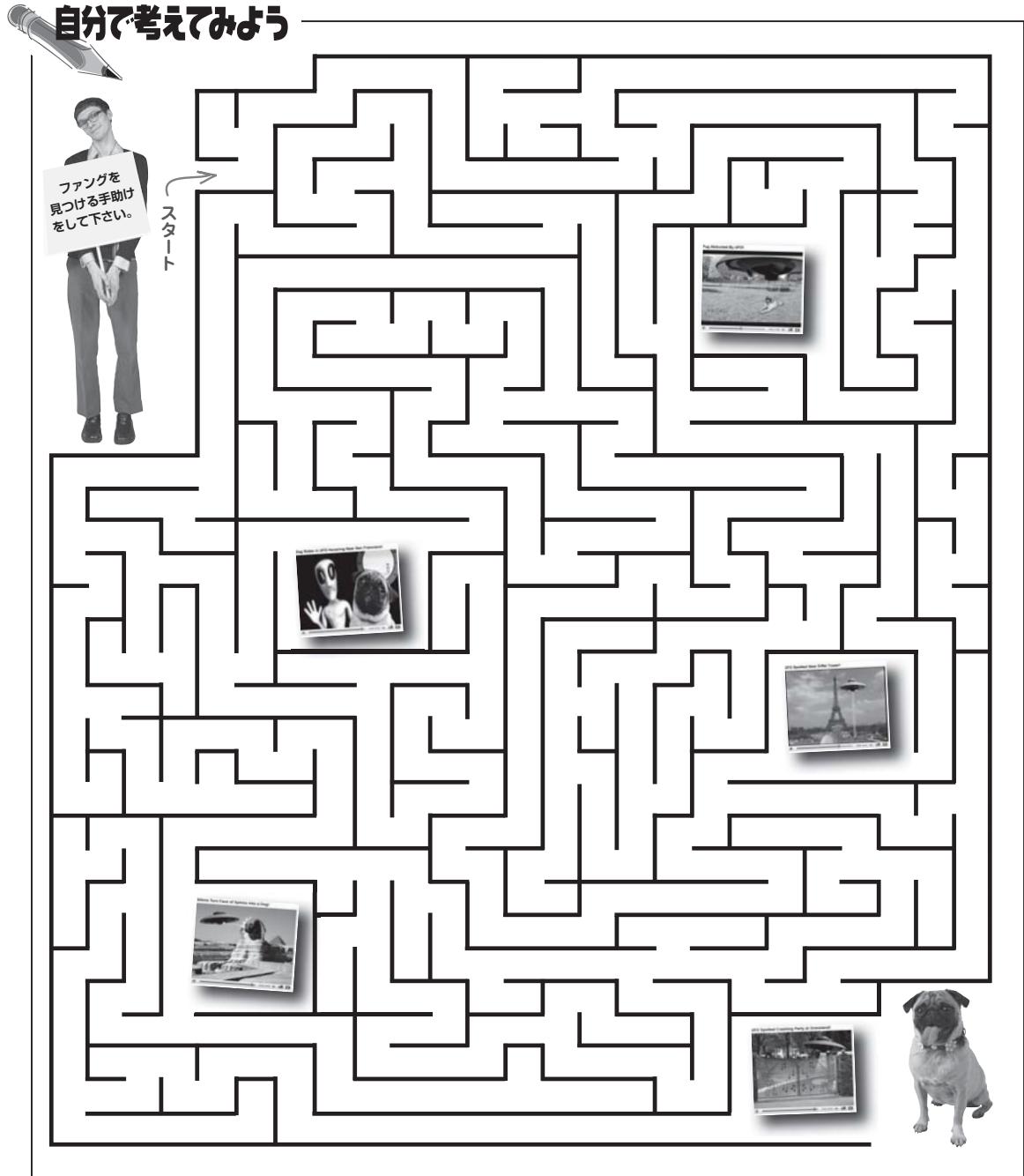
UFO Spotted Near Eiffel Tower!
0分13秒



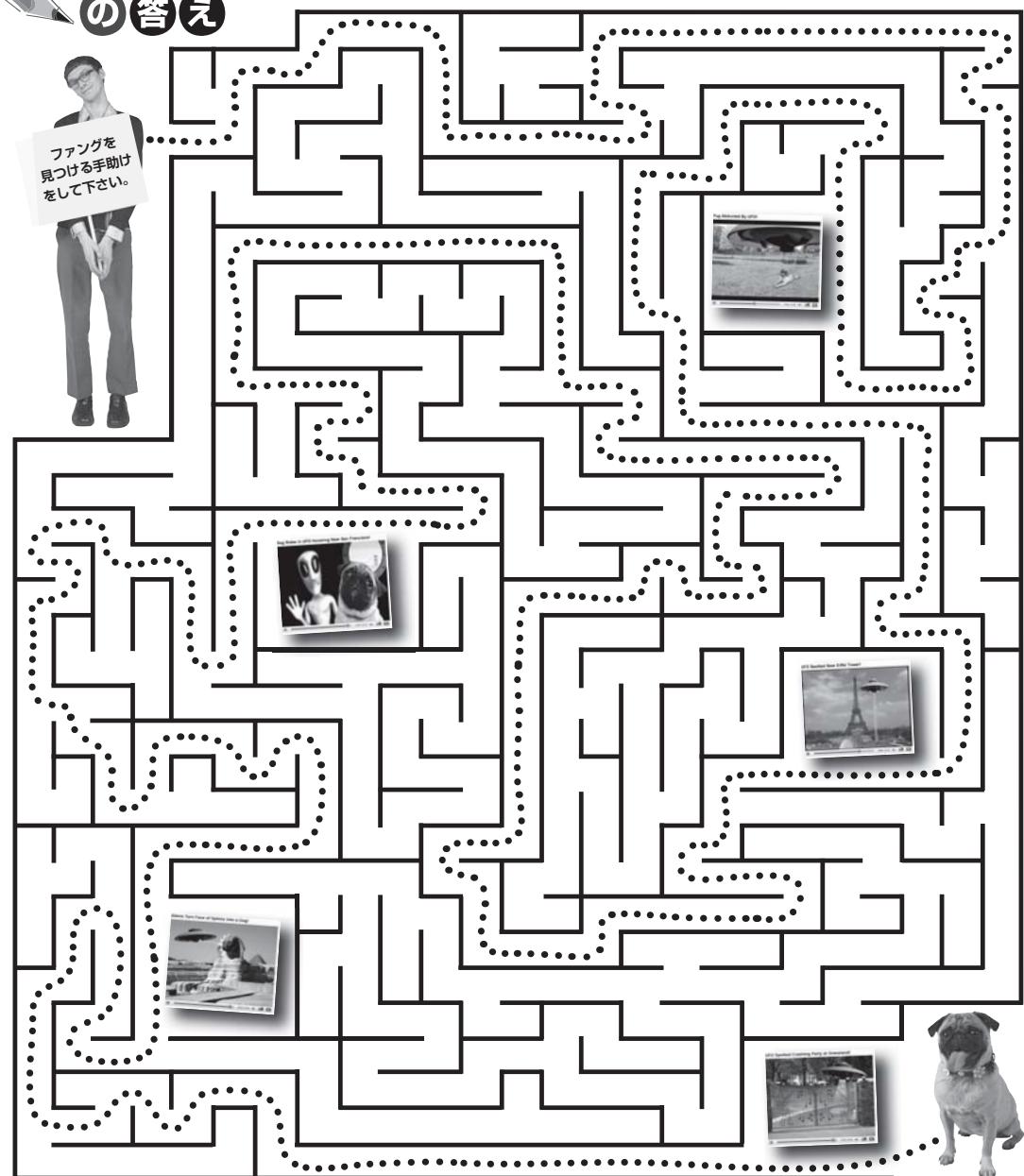
Pug Abducted By UFO!
0分18秒



自分で考えてみよう



自分で考えてみよう
の答え





PHP & MySQL道具箱

ファンダムが見つかったので、その足跡をたどってみましょう。ご承知の通り PHP と MySQL からもさらにいくつかの技法の助けを借りました。

XML

汎用のマークアップ言語で、想定された構造のデータを提供する場合に使います。XMLから作られたマークアップ言語にはたくさんの種類があります。例えば XHTML や RSS です。考え方としては、XML に突っ込むあらゆるデータに特定のタグを作っていくというものです。

`simplexml_load_file()`

PHP の組み込み関数で XML ファイルを URL から読み込み、結果の XML データをオブジェクトとしてアクセスできるようにします。

`SimpleXMLElement`

PHP の組み込みオブジェクトで、XML データにアクセスする場合に使います。このオブジェクトは `simplexml_load_file()` 関数が返すもので、XML ドキュメントの完全な階層構造を表現しています。

REST

Web 上の情報に純粋に URL だけを通してアクセスする手段です。REST のおかげで強力なデータリクエストを、単に URL を作るだけでできるようになります。このようなリクエストのことを、よく「RESTful」な(平和な)リクエストと言います。

RSS

XML をベースとする言語で、ニュースなどのシンジケーション用のコンテンツを格納するのに使います。RSS は Web サイトのデータを他のアプリケーションで使えるようにし、Web サイトがシンジケーションに対応できるようにします。これにより他のサイトで作られたデータが使えるようになります。

名前空間

XML タグを集めて論理的なグループの構成する方法です。「姓」で家族を名前のついたグループとして構成してするようなものです。名前空間は常に URL と結び付いていて、他のすべての名前空間に対して一意であることを保証してくれます。



おしまい。



付録i 残飯

本編でカバーしなかったこと



全部終わった後ですが、まだ少し残っています。知っておくべきことはもう少しあると思います。それらを無視するというのは良いことではありません。ほんの少しだけでも説明しておくべきでしょう。そこで、本をしまう前に、以下を読んでおいて下さい。PHPとMySQLに関する短いけれども大事な小話です。さて、この部分を読み終えたら、もういくつか付録があって…索引…、それと多分広告のページ…、そうしたら本当に終わりです。約束します！

本書の内容を PHP4 と MySQL 関数に 後方互換で対応させる

11章の XML 関数を除けば、本書の大部分のコードは、ほとんど修正なしに PHP 4 サーバ上でも動作します。本書では mysqli 系の関数を使いましたが、これらの関数は PHP 4.1 以降でしか使えません。これらのライブラリは手動でインストールする必要があるため、サーバによっては mysqli 関数をサポートしていません。

mysqli 関数は一般に速いのですが、この速度が問題になるのは、データベースが大量のデータを抱えてからのことです。小さなデータベースまたは標準的なデータベースであれば、以前の mysql 関数を使っても、有意な差は認識できないと思います。ここでは、mysql 関数を後方互換にするために、mysql 関数で書き直し、古い版の PHP で動作させる方法を示します。

次のプログラムでは：

```
$dbc = mysqli_connect('localhost', 'mork', 'fromork');
mysqli_select_db($dbc, 'alien_database');
```

次のように書き直します。

```
$dbc = mysql_connect('localhost', 'mork', 'fromork');
mysql_select_db('alien_database', $dbc);
```

データベース接続変数は、`mysqli_select_db()`
と違って第1引数ではありません。

一般に、単に mysqli の i を取り去り、次に引数の順序を入れ替えればよいだけです。入れ替えてデータベース接続変数（この例では \$dbc）が最後に来るようになります。

ただし、少々厄介なのは `mysqli_connect()` 関数が `mysql_select_db()` 関数を押しのけて、データベース名も引数となっている場合です。mysql 系の関数では全く同様のことはできません。`mysqli_connect()` 関数だけでデータベース名も使用している場合は、2つの mysql 関数に分ける必要があります。

次のプログラムでは：

```
$dbc = mysqli_connect('localhost', 'mork', 'fromork', 'alien_database');
```

次の2行が必要になります。

```
$dbc = mysql_connect('localhost', 'mork', 'fromork');
mysql_select_db('alien_database', $dbc);
```

この接続変数データベース接続の
「リンク」と呼ぶこともあります。

ここで接続を確立しながら
データベースを選んで
います。このようなことは
mysql の関数では
できません。

mysql 関数では、特定のデータベース
と接続を確立するには常に2度の
関数呼び出しが必要となります。

mysql関数とmysqli関数との対応表を以下に示します。

操作	mysql関数	mysqli関数
MySQL接続のクローズ	mysql_close(conn)	mysqli_close(conn)
MySQLサーバへの接続オープン	mysql_connect(host, username, password) まずmysql_select_db()でデータベースを選ぶ必要がある。	mysqli_connect(host, username, password, database) mysqli_select_db()を使わなくてもデータベースを選ぶことができる。
前回のMySQL操作でのエラーメッセージテキストを返す。	mysql_error(conn)	mysqli_error(conn)
文字列のエスケープ	mysql_escape_string(string, conn) 引数の順番が逆。最初が文字列で、次が接続(リンク)。	mysqli_escape_string(conn, string) 引数は最初が接続(リンク)で、次が文字列。
結果の行を連想配列、数値配列、またはその両方で取ってくる。	mysql_fetch_row(result)	mysqli_fetch_row(result)
結果の行数を返す。	mysql_num_rows(result)	mysqli_num_rows(result)
MySQL問い合わせ文を実行する。	mysql_query(conn, query)	mysqli_query(conn, query)
文字列内の特殊文字をエスケープする。	mysql_real_escape_string(string, conn) 引数の順番が逆。最初が文字列で、次が接続(リンク)。	mysqli_real_escape_string(conn, string) 引数は最初が接続(リンク)で、次が文字列。
MySQLデータベースを1つ選ぶ。	mysql_select_db(dbname, conn) 引数の順番が逆。最初が文字列で、次が接続(リンク)。	mysqli_select_db(conn, dbname) 引数は最初が接続(リンク)で、次が文字列。

付録ii 開発環境をセットアップする

作業をする場所



カレは私の料理が上手
だと思ってるけど、実は
見られる前に失敗は全部
隠しちゃったの。

PHPとMySQLに関する真新しいスキルを練習するには、Web上のデータが攻撃にさらされる心配などがない場所が要ります。 PHPアプリケーションを世界中に（Webで）発信してしまう前に、まず安全な場所を確保しておくというのは非常に良いことなのです。この付録にはWebサーバ、MySQL、それとPHPをインストールする手順が書いてあります。安全な場所で作業や練習に取り組むことができます。

PHP 開発環境を作る

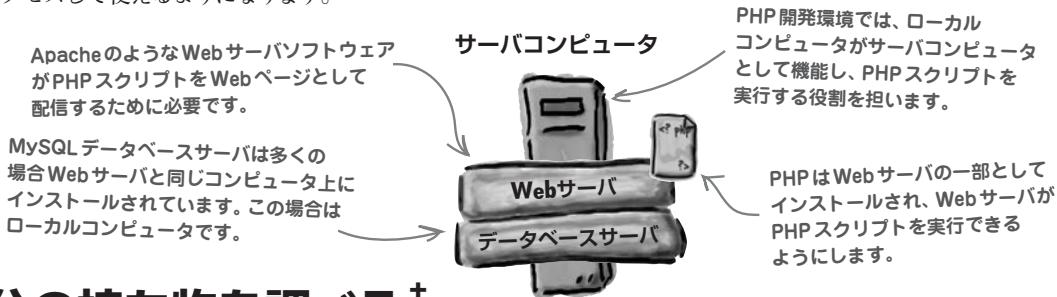
Web 上に完成したアプリケーションを置く前に、まずは開発する必要があります。しかし Web アプリケーションの開発を誰もが見ることのできる Web 上で行うなど言語道断です。ソフトウェアをインターネット上に公開する前に、ローカルにインストールしビルドしてテストしなければいけません。

PHP アプリケーションをビルドしてテストするには、3種類のソフトウェアがローカルコンピュータに必要になります。

1. Web サーバ
2. PHP
3. MySQL データベースサーバ

PHP はサーバではありません。PHP というのは規則の集まりです。Web サーバが規則を理解し、PHP コードを翻訳できるようにしてくれます。Web サーバも MySQL サーバも両方もコンピュータ上で動く実行形式プログラムです。

まず覚えておいて欲しいのですが、ここではローカルコンピュータを Web サーバとして PHP 開発環境をセットアップする手順を説明します。究極的にはインターネット上に Web サーバが必要になります。そこに完成したアプリケーションをアップロードすれば誰でもアクセスして使えるようになります。



自分の持ち物を調べる[†]

PHP 開発環境パズルにピースを 1 つでもインストールするまでの最良の戦略は、まずすでに何がインストールされているかを調べることです。3 つのピースそれぞれについて、何がすでにしているかを見分ける方法について見てきます。

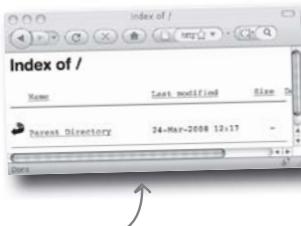
ローカルコンピュータのプラットフォームによって何がインストールされているかどうかについて大きな違いがあります。例えば Mac OS X には Web サーバがデフォルトでインストールされていますが、Windows 系ではたいていインストールされていません。

注意：この付録では Windows 2000, XP, Vista, Windows Server 2003/2008 その他 32 ビット Windows OS をカバーしています。Macについては Mac OS X 10.3.x 以降が対象です[†]。

[†] 訳注：日本語版では Windows XP, Vista または 7, Mac OS X 10.5.7 以降 (Leopard, Snow Leopard) の通常を状態を想定して、必要となるサーバソフトウェアを一括でインストールする方法を紹介します。本書で扱われるアプリケーションをパソコン自体をサーバとしてテストするという目的では、これで十分です。逆に必要なソフトウェアを個別にインストールするのは、設定などが結構厄介です。通常パソコンには Apache, PHP, MySQL はインストールされていないか、またはインストールされていても無効となっているはずですので、一括インストール (597 ページ) へ進んで下さい。

Webサーバがありますか？

比較的新しいPCやMacを使っているのでしたら、恐らくWebサーバはすでにあります。いずれのシステムでも手早く調べるには、ブラウザを開いてアドレスバーに `http://localhost/` と打ち込んでみることです。以下のような紹介ページになつたら、Webサーバがローカルマシン上でちゃんと生きているという意味です。



MacまたはWindowsでApache Webサーバがインストールされている場合、こんな風に表示されるかも知れません。



IISがインストールされたWindowsマシンでは、こんな風に表示されるかもしれません。

PHPがありますか？バージョンは？

Webサーバがあったら、PHPがインストールされているかどうかをチェックするのは簡単です。バージョンも同様にすぐ分かります。`info.php`という名前で新しいスクリプトを作り、次のように打ち込みます。

```
<?php phpinfo(); ?>
```

このファイルをWebサーバが使うフォルダ(ディレクトリ)に保存します。Windowsの場合は、通常以下の場所[†]です。

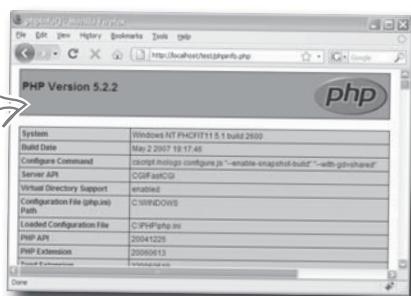
C:\Inetpub\wwwroot

Macの場合、普通こんな感じです。

/Users/yourname/sites/

このファイルをブラウザで開く場合、`http://localhost/info.php`と打ち込んで、次のような画面が現れたらPHPがインストールされていることを意味します。

ここにインストールされているPHPのバージョンが書いてあります。

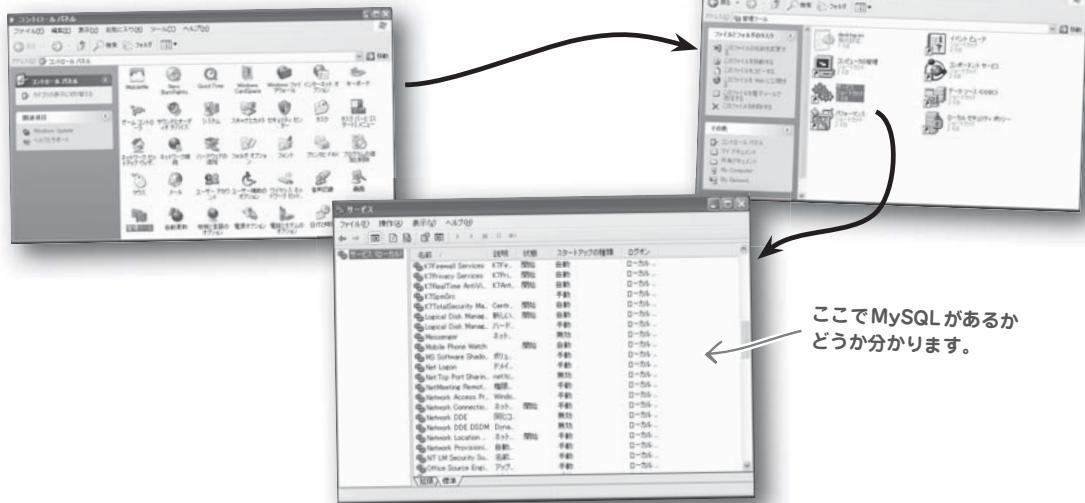


[†] 訳注：Apacheをインストールした場合、デフォルトでは以下の場所となります。

C:\Program Files\Apache Software Foundation\Apache2.2\htdocs

MySQL がありますか？バージョンは？

Windowsの場合、コントロールパネル --> 管理ツール --> サービスを開くとわかります。



ここでMySQLがあるかどうか分かります。

MacにMySQLがあるかどうかを見るには、ターミナルを開いて以下のように打ち込みます。

```
cd /user/local/mysql
```

このコマンドが動いたら、MySQLがインストールされています。バージョンをチェックするには、次のように打ち込んでみて下さい。

```
mysql
```

MySQLターミナルは、MySQLモニタとも言います。

このコマンドが成功したら、MySQLがインストールされていることを意味します。

これがインストールされているMySQLのバージョンです。

```
C:\$Program Files\VertrigoServer\mysql\bin\$mysql.exe  
Enter password: *****  
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 17  
Server version: 5.0.51b-community MySQL Community Edition (GPL)  
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.  
mysql>
```

サーバをセットアップする[†]

ここではApache、PHP、MySQLのすべてをまとめてインストールすることが簡単にできる方法を示します。2009年5月時点でVertrigoServとXAMPPの2種類がありますが、これらを使うと個別にインストールした後の設定が不要で、インストールが完了すればすぐに動きます。本書に従って作ったプログラムをパソコン上でテストするだけなら、これで十分使えます。商用のWebサーバへインストールすることを想定して書かれていませんのでご注意下さい。

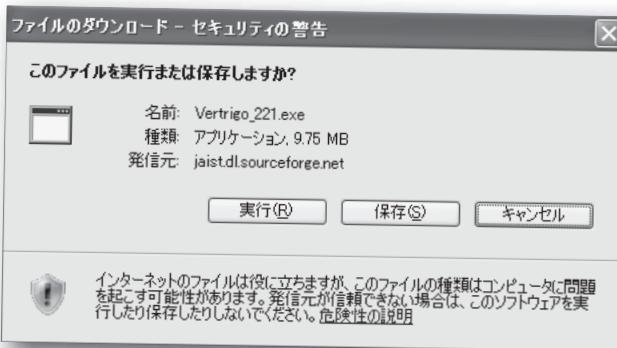
WindowsではVertrigoServとXAMPPの両方が提供されています。MacではXAMPPのみ利用で行きます。本書ではWindows上でVertrigoServをインストールする方法と、MacでXAMPPをインストールする方法を示します。

WindowsにVertrigoServをインストールする

まず管理者権限のあるアカウントでログインしていることを確認して下さい。VertrigoServは<http://vertrigo.sourceforge.net/>からダウンロードします。ページ上「GET THE LATEST VERSION」と書いてある部分をクリックすると、ダウンロード画面に遷移します。ダウンロードボタン：



をクリックすることでダウンロードが始まります。次のようなウィンドウが出たら「実行」をすることで、ダウンロードの終了と一緒にインストールが始まります。



Vistaや7の場合はセキュリティに関する確認画面が出ますので、ダウンロードや実行を許可して下さい。

[†] 訳注：本付録の以下のページは、原著ではApache、PHP、MySQLをそれぞれ独立にインストールする方法を示していました。日本語版ではこれを全面的に書き換えました。

VertrigoServ のインストール

インストールが始まると、まずインストール時の言語を聞いてきます。



デフォルトで日本語になっていますので、「OK」をクリックします。以下インストールウィンドウの指示に従って下さい。デフォルトで設定されているものを変更する必要はありません。「そのまま」でじゃんじゃん進んで下さい。



これでインストールは終了です。

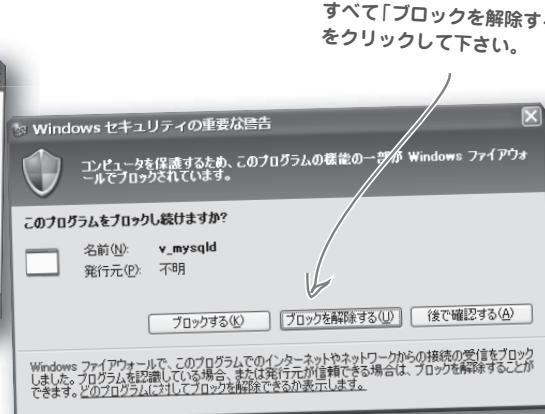
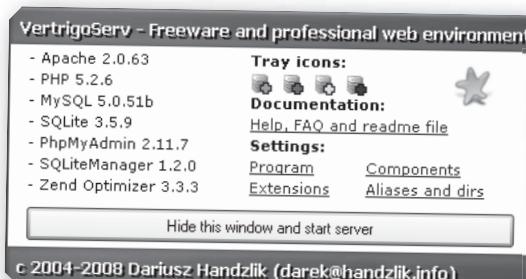
WindowsでVertrigoServを起動する

インストールが完了するとデスクトップ上に以下のようなアイコンができるいるはずです。



VertrigoServを起動するには、このアイコンをダブルクリックすればOKです。また「スタート」からメニューをたどって、VertrigoServを選んで起動することもできます。起動すると次の左のようなウィンドウが現れます。このウィンドウの下にある大きな横長のボタンをクリックすることで、すべてのサーバソフトウェアを起動することができます。

このボタンをクリックするとセキュリティに関する右のような警告画面がいくつか表示されます。



起動がうまくいった場合、画面右下のタスクトレイに以下のように表示されます。7の場合、「隠れているインジケータ」の中にあるかもしれません。



図のように太いプラスのような記号が緑色であれば、サーバは正常に動作しています。

Apache と PHP の動作を確認する

では、Apacheが動作しているかを確認してみましょう。ブラウザを立ち上げ、`http://localhost/`にアクセスしてみて下さい。次のような画面が表示されれば、VertrigoServのApacheは正常に動作しています。

PHPについては、`http://localhost/info.php`にアクセスしてみて下さい。次のように表示されるはずです。

The left screenshot shows the VertrigoServ 'Welcome to VertrigoServ' page with a 'Information' section listing packages like Apache 2.0.63 and PHP 5.2.6, and a 'Tools and links' sidebar with links to PhPMyAdmin, SQLiteManager, and other documentation.

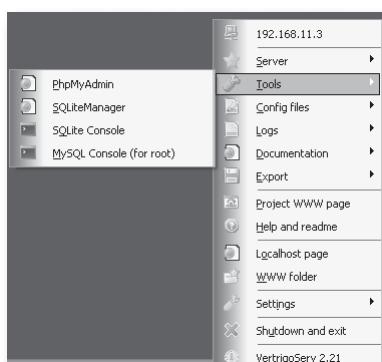
The right screenshot shows the 'phpinfo()' page titled 'PHP Version 5.2.6'. It displays detailed configuration information for the PHP environment, including system details (Windows NT IODATEN 5.1 build 2600), build date (May 2 2008 18:01:20), and various PHP extensions and configurations.

A callout arrow points from the URL `http://localhost/` to the first screenshot, and another callout arrow points from the URL `http://localhost/info.php` to the second screenshot.

以上でApache、PHPが正常に動作していることを確認できました。

MySQL の動作を確認する

次にMySQLサーバが動作していることを確認します。タスクトレイのVertrigoServアイコンをクリックして下さい。次のようなメニューが現れます。



ここで、Toolsの部分にカーソルを合わせると、サブメニューが現れますので、「MySQL Console (for root)」を選びます。するとコマンドプロンプト(DOS窓)と同様なウィンドウが現れます。

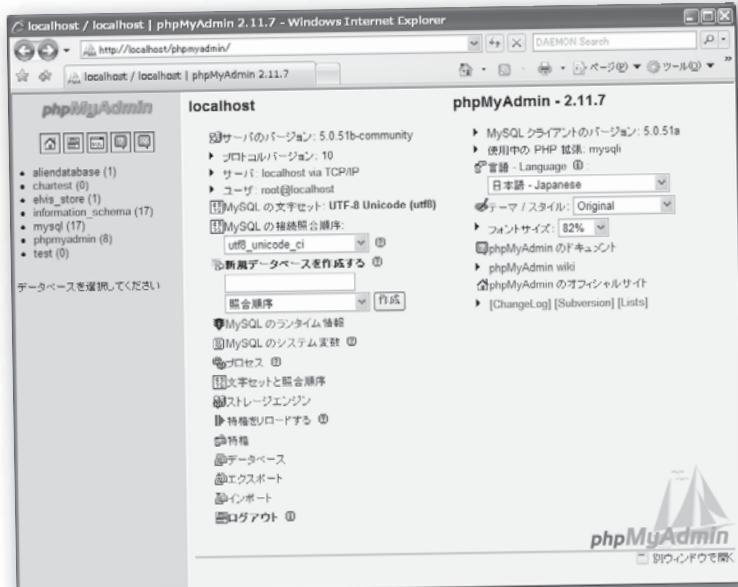
```
C:\Program Files\VertrigoServ\mysql\bin>mysql -u root
Enter password: *****
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 17
Server version: 5.0.51b-community MySQL Community Edition (GPL)

Type 'help' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

データベースパスワードの入力を求められています。「vertrigo」と入力して上記のように表示されれば、MySQLは正常に動作しています。この画面で「phpMyAdmin」を選ぶか、または直接 <http://localhost/phpmyadmin/> にアクセスすることで phpMyAdmin を使うことができます。HTTP認証画面が現れますので、ユーザ名として「root」、パスワードとして「vertrigo」と入力して下さい。次のような画面が表示されればOKです。

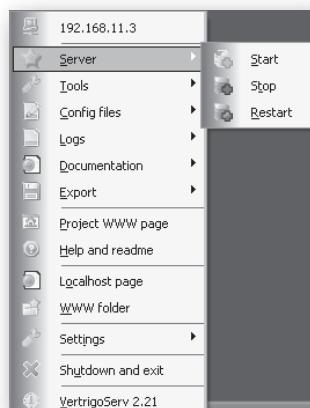
VertrigoServでMySQLをインストールした場合、デフォルトのデータベースユーザ名は root であり、データベースパスワードは vertrigo となっています。



VertrigoServを停止する

最後に、VertrigoServの終了および再起動の方法について説明します。タスクトレイのVertrigoServアイコンをクリックして下さい。次のようなメニューが現れます。

ここで、Serverの部分にカーソルを合わせると、サブメニューが現れます。起動中であればこの図のように表示されます。この状態から「Stop」を選ぶことで、サーバはすべて停止します。「Restart」を選ぶと、サーバは一度停止した後、再び起動します。停止している状態で、「Start」を選べばサーバが起動します。なお、通常使用している場合これらの操作は必要ありません。パソコンをシャットダウンすればVertrigoServは自動的に終了します。ただし、サーバの設定を変更した場合などには、サーバを再起動(停止と起動)する必要があります。



MacにXAMPPをインストールする

インストールには管理者権限が必要です。インストールに先立って管理者パスワードを要求されます。

XAMPP for Mac OS Xは、<http://www.apachefriends.org/jp/xampp-macosx.html>からダウンロードします。サイトは日本語で書かれていますので、単に指示に従って下さい。



最新版をダウンロードすれば良いでしょう。執筆時点での最新版は1.7.2aでした。「XAMPP Mac OSX 1.7.2a」と書いてある部分をクリックして下さい[†]。ダウンロードが始まります。

ダウンロードが完了すると、次のようなウィンドウが開きます。



左側のフォルダを右側のフォルダにドラッグ＆ドロップすればインストールは完了です。

[†] 訳者注：2009年8月時点でのダウンロードサイトの構成に基づいて、説明しています。読者がアクセスした時点での実際のサイトの構成とは異なる可能性があります。その際は実際の画面の指示に従って下さい。

MacでXAMPPを起動する

XAMPPは、アプリケーション >> XAMPPにインストールされています。
Finderでアプリケーション >> XAMPP >> XAMPP Control.appをダブルクリックします。Dockには

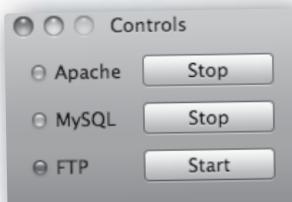


のようなアイコンが現れますので、右クリックしてDockに追加しておくと良いでしょう。

XAMPPcontrol.appを起動すると、Getting StartedとControlsという2つのウィンドウが現れます。Getting Startedの画面に表示されている通り、初期状態ではMySQLのデータベースユーザ名としてrootが、データベースパスワードとしては「なし」が設定されています。



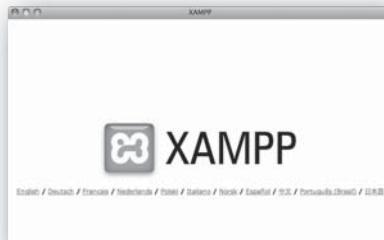
Controls ウィンドウのApacheボタンと、MySQLボタンをクリックすることでそれぞれのサーバを起動することができます。このとき管理者権限を持つユーザーのパスワードを要求されます。起動に成功するとControls ウィンドウは次のように変わります。



このように画面左側のボタンが緑色に変わっていれば、サーバは正常に起動しています。

ApacheとPHPの動作を確認する

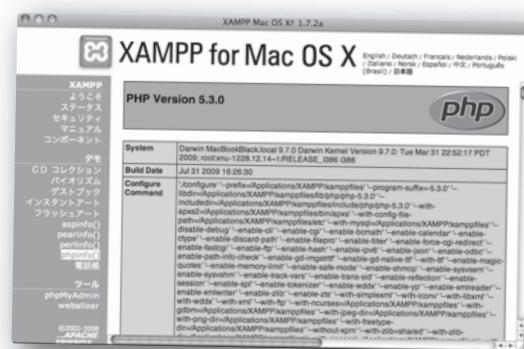
では、Apacheが動作しているかを確認してみましょう。ブラウザを立ち上げ、`http://localhost/`にアクセスしてみて下さい。次のような画面が表示されれば、XAMPPのApacheは正常に動作しています。



PHPについては、上記画面で「日本語」を選び、左側に表示されたメニューから「phpinfo()」を選んでクリックします。右のような画面が表示されるはずです。以上でApache、PHPが正常に動作していることが確認できました。

MySQLの動作を確認する

XAMPPのMySQLでは、mysqlターミナルは、「アプリケーション > XAMPP > xamppfiles > bin」にインストールされます。Finderでこのフォルダの下にある、mysqlをダブルクリックして下さい。ターミナルが立ち上がり、次のように表示されます。



```
ターミナル - mysql - 80x24
Last login: Sat Nov  7 18:31:15 on console
/Applications/XAMPP/xamppfiles/bin/mysql ; exit;
silverMac:~ matsu$ /Applications/XAMPP/xamppfiles/bin/mysql ; exit;
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.1.33 Source distribution

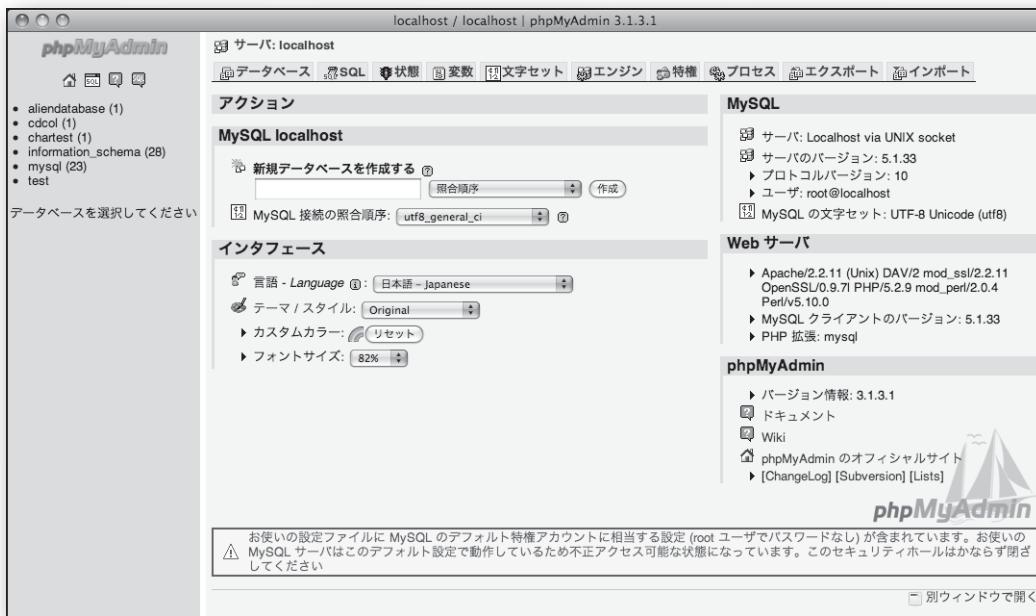
Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql>
```

これでMySQLが正常に動作していること、並びにmysqlターミナルが使えることが確認できました。初期状態ではデータベースパスワードは設定されていないため、パスワードなしで起動することができます。

XAMPPでphpMyAdminを使う

XAMPPには先ほど見た管理用画面が用意されています。ブラウザで、<http://localhost/>にアクセスして表示される画面の左側にあるメニューから、phpMyAdminを選んでクリックするか、または直接<http://localhost/phpmyadmin/>にアクセスすることでphpMyAdminを使うことができます。次のような画面が表示されればOKです。



XAMPPでインストールMySQLした場合、デフォルトのデータベースユーザ名はrootであり、データベースパスワードは「なし」となっています。

XAMPPを終了する

XAMPP Controls ウィンドウでApacheとMySQLのそれぞれについて「Stop」をクリックすることで。XAMPPのApache、PHP、MySQLは終了します。なお、通常使用している場合、これらの操作は必要ありません。パソコンをシャットダウンすればXAMPPは自動的に終了します。ただし、サーバの設定を変更した場合などには、サーバを再起動(停止と起動)する必要があります。

メール送信に関する設定

1章でmb_send_mail()関数を使ってPHPからメールを送信します。このためには、PHPにメールサーバが誰なのかを教えてあげる必要があります。この設定にはphp.iniというコンフィグファイルを書き換えることで達成できます。php.iniファイルは、VertrigoServの場合は、タスクトレイのアイコンをクリックし、サブメニューから「Config files >> php.ini」を選ぶことで書き換えられます。XAMPPの場合は、「Applications >> XAMPP >> etc >> php.ini」をエディタで開いてください。

630行目あたりに[mail function]という部分があります。恐らく以下のように記述されています。

```
[mail function]
; For Win32 only.
SMTP = localhost
smtp_port = 25

; For Win32 only.
; sendmail_from = me@example.com

; For Unix only. You may supply arguments as well (default: "sendmail -t -i").
; sendmail_path =

; Force the addition of the specified parameters to be passed as extra parameters
; to the sendmail binary. These parameters will always replace the value of
; the 5th parameter to mail(), even in safe mode.
;mail.force_extra_parameters =
```

これをご自身の環境に合わせて変更してください。「SMTP = localhost」の部分を正しいメールサーバとする必要があります。ご自身のメール環境で利用しているものであれば、そのまま利用できます。例えば、利用している送信メールサーバのアドレスが、mail.sample.ne.jpであれば、「SMTP = mail.sample.ne.jp」とします。ただし、プロバイダとの契約等により、プログラムからのメールを送信を許可されていない可能性もありますので契約内容をご確認ください。また、プログラムから一時期に大量のメール(特に携帯宛のエラーメール)を送信すると迷惑メールの送信主扱いとなってしまう可能性があります。くれぐれもご注意ください。

また、「; sendmail_from = me@example.com」の部分については、コメントを外した上で、ご自身のメールアドレスに変更しておきます。例えばご自身のアドレスがjibun@mail.sample.ne.jpであれば、「sendmail_from = jibun@mail.sample.ne.jp」とします。

付録 iii PHP & MySQL日本語対応[†]

PHPとMySQLで 日本語を扱う



もう何もかもあるわ。
ありきたりのものでも、ドキッと
する程きれいな悪魔のように買ひ
運命の女^{ひと}が欲しいという
ものでも…でもまだ足りない。

その通りです。PHPとMySQLでプログラミングをし、すばらしいWebアプリケーションを作ることができるようになりました。でも、まだやらなければならないことがあります。ここは日本なのです。PHPとMySQLで日本語を取り扱えなければ話になりません。本文からも参照していますが、PHPとMySQLで日本語を扱う場合に、知っておかなければならぬことを、ここにまとめてあります。

[†] 訳注：この付録はHead First PHP & MySQL 日本語版に対応して追記しました。

PHP 日本語スクリプトを作る

PHPのスクリプトコード内で日本語データ(メッセージ)を使うためには、コード系をUTF-8またはEUCで作る必要があります。PHPはWindowsで標準的に使われているShift-JISには対応していません。このため、例えばWindowsメモ帳でスクリプトを作り、HTMLタグに

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=shift-jis" />
```

という指定をしても幸せはやってきません。まず第1に

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="ja" lang="ja">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
```

と書き換える必要があります。しかし、それだけでスクリプトのコード系自体がUTF-8に自動的に変わってくれる程、世の中甘くありません。先のWindowsメモ帳でファイルを作ると、コード系は普通Shift-JISとして保存されます。でも大丈夫。メモ帳ならファイルを保存する際にコード系を指定することができます。普通にファイルを保存する代わりに、「ファイル」>>「名前を付けて保存」を選んで下さい。このとき出てくるウインドウの一番下に「文字コード」という欄があります。プルダウンで「UTF-8」を選んで保存すれば、コード系をUTF-8にすることができます。

Macではテキストエディットのデフォルト設定がUTF-8になっているはずですから、何も変更しなくて大丈夫です。「テキストエディット」>>「環境設定」の「開く/保存」タブで一番下の文字コードを確認してみて下さい。

PHPが対応しているコード系としてEUCでも日本語データを使うことができます。このためにはEUC形式で保存できるエディタが必要となります。この点(および<meta>タグ内のcharset=eucとする点)以外は同じことをすればコード系はEUCでもOKです。でもくどいようですが、Shift-JISはダメです。

どうしてShift-JISはダメなの?

PHPというプログラムは、スクリプト記述用のコードとしてShift-JISという日本の「方言」を知らないためです。ただし、運が良ければ(ほとんどの場合?)問題なく動作するよう見えるかもしれません。でも、あるスクリプトがShift-JISでもたまたまちゃんと動くということと、PHPがShift-JISに対応しているということとは別の話です。試しに、コード系をShift-JISとして次のような小さなスクリプトを書いてみて下さい。

```
<?php
$ten = "十";
echo $ten;
?>
```

このファイルをtest.phpとでも名付けて、サーバにアップロードし、クライアントのWebブラウザで開くとどうなるでしょう? PHPがShift-JISに対応してくれれば、

もちろん「十」と表示されるはずです。しかし実際にはPHPはわけのわからぬエラーメッセージを吐きます。

恐らくブラウザにはこんなメッセージが表示されたはずです。

```
Parse error: syntax error, unexpected $end in C:\Program Files\VertrigoServ\www\test.php on line 4
```

同じことをコード系だけをUTF-8に変えて(ファイルをUTF-8で保存し直して)やってみましょう。今度は期待通り「十」と表示されます。これこそが、PHPというプログラムがスクриプト記述言語としてのShift-JISを知らないという証拠であり、使えない理由なのです。

Shift-JISスクリプトのエラー詳細

では先ほどのShift-JISスクリプトでエラーメッセージが出た理由を詳しく説明します。あのスクリプトでPHPが発狂してしまうカギは実は漢数字の「十」という文字にあったのです。Shift-JISの日本語文字は通常2バイトで表現されますが、PHPはShift-JISを知らないため、2バイトで1文字とは解釈せず、1バイトずつそれぞれを文字とみなして処理してしまいます。ここで問題なのが「十」を構成している2バイト文字の2バイト目です。実はこの文字の2バイト目は16進数で表すと5cという値を持ちます。この値は通常の(1バイト)文字の円記号(バックスラッシュ)\¥の値とたまたま一緒なのです。このためPHPは、「十」の2バイト目を円記号と勝手に解釈します。するとその次にある二重引用符""は円記号と組み合わせてエスケープされてしまいます。このため本来文字列終了のためにあたはずの二重引用符は文字列中のエスケープ文字として扱われます。PHPは文字列の終了を探して更にスクリプト読み進めますが、文字列の終了を表す""は現れず、そのままスクリプトファイルは終わってしまいます。PHPでは文字列の読み込み中にファイルが終わってしまったため、エラーと判断してメッセージを吐くことになります。

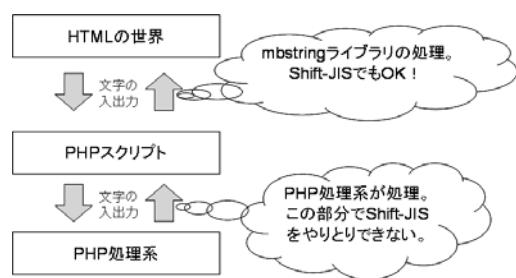
同じように2バイト目がたまたま円記号と一致してしまうやばい文字には、他に「ソ」、「噂」、「構」、「能」、「表」、「暴」、「予」などが(他にもたくさん)あります。

円記号以外にもShift-JISの2バイト目の値が普通の(1バイト)文字の値とたまたま一致しているための問題はあります。

php.iniの設定を変えればいいのでは？

残念ながらダメです。php.iniの設定はmbstring系ライブラリ(マルチバイト対応のライブラリ)の入出力(特にデフォルト設定)を規定するのに使っています。今問題となっているのは、PHPという処理系自体の入出力(つまりPHPスクリプト自体)です。この部分にShift-JISを用いると、今まで述べたような問題が起こります。

図で言うと、mbstring系ライブラリやphp.iniで取り扱っているのは、上側の矢印での文字入出力です。これに対し今問題なっているのは下側の矢印による文字入出力です。つまりPHPスクリプトとPHP処理系との間での文字のやりとりが問題なのです。結局、PHPスクリプト自体をShift-JISで書いてしまうと、問題となることは現時点では避けられません。



PHPでShift-JISのページを作れないということ？

そんなことはありません、作れます。今まで説明したのはスクリプト記述用のコードとしてのShift-JISです。PHPというプログラムの入力としての文字列や出力としての文字列がShift-JISであることには何も問題はありません。PHPの出力としてShift-JISを指定すればShift-JISのページになります。先ほどの図で言うと、上側の矢印の世界であればShift-JISでも大丈夫です。そのためには、ページのすべてを（静的な部分も動的に変わる部分も）PHPがShift-JISで出力すればよいのです。スクリプトがUTF-8で記述されていようと、EUCで記述されていようとShift-JISを出力すればよいだけです。ただ、具体的な方法については本書の範囲を超えますので、ここでは説明しません。

日本語はPHPスクリプトのどこに書けるのか？

PHPスクリプトの中で日本語は、アルファベットと同じ扱いを受けます。従ってアルファベットが書けるところであればどこにでも書けます。実際問題としては、文字列内とコメントに書ければ十分でしょう。属性値に書きたいというのも人情でしょう。でもこの程度に留めておくことをお勧めします。実は変数名とか関数名としても日本語を使うことができます。しかし、このようなコーディングスタイルは、広く受け入れられていません。むしろ「そんなことができる！」などと知らないの方が多いくらいです。このようなコーディングはしないことを強くお勧めします。

変数を含む日本語文字列をechoしたら何も表示されない！

次のようなPHPスクリプトを作って表示させてみましょう。

```
<?php  
$name = '中村公子';  
echo "$name さん、こんにちわ";  
?>
```

予想に反して（予想通り？）何も表示されなかったことと思います。どうしてでしょう？実は先ほどの説明と関係があります。日本語はPHPスクリプトのどこにでも書けるのです。そしてPHP処理系は日本語をアルファベットと同等に扱います。するとどうなるでしょうか？上のプログラム例で言うと、1行目は、\$nameという変数に'中村公子'という文字列を代入したことになります。これは意図したとおりです。しかし2行目は、「\$name さん、こんにちわ」という1つの長い名前の変数をechoしていることになります。\$に続くアルファベットの並びは「わ」まで続いていたのです。そしてこのような変数は定義されていません。ですから変数に何も入っていないため、何も表示されなかったというわけです。

これを回避して思惑通りにechoするには2通りの方法があります。1つ目は英語のように単語の間にプランクを入れることです。つまり：

```
<?php  
$name = '中村公子';  
echo "$name さん、こんにちわ";  
?>
```

といった具合です。これで「中村公子さん、こんにちわ」と表示されます。全角のプランクはダメですから注意して下さい。(PHPは全角のプランクもアルファベットと同等に扱うのです!)ただ、表示上(半角の)プランクがあるのはちょっと気にくわない、という方のためにもう1つ技があります。

```
<?php
$name = '中村公子';
echo "{$name} さん、こんにちわ";
?>
```

このように、変数名を波カッコ({と})で囲むのです。これでPHP処理系は変数の区切りを理解することができるので、今度こそ「中村公子さん、こんにちわ」と表示されます。

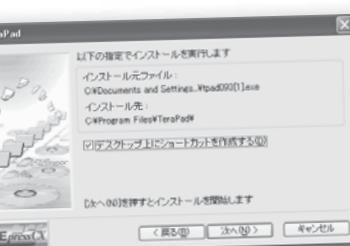
UTF-8 のファイルを作る

PHPのソースをUTF-8で作るためには、UTF-8Nという形式でファイルを保存できるエディタが必要です。

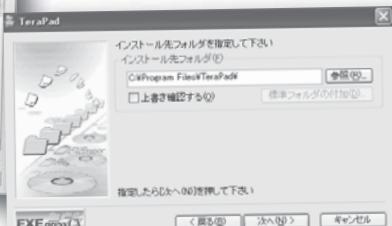
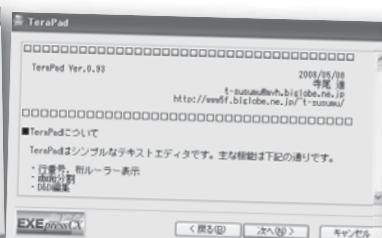
ここではTeraPadというフリーのエディタを紹介します。窓の杜やVectorなどのサイトからダウンロードできます。

インストール方法

サイトからダウンロードボタンを押し、実行をクリックします。ブラウザがダウンロードをブロックする場合は、メッセージ部分を右クリックしてダウンロードを許可して下さい。以下インストーラの指示に従うだけでOKです。



ここで、メッセージダイアログが表示されます。単に「はい」をクリックして下さい。

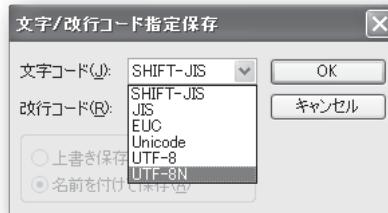


以上で、インストールは完了です。

使い方

デフォルト設定のままインストールしたのであれば、デスクトップに作られたアイコンをクリックすれば起動することができます。もちろん「スタートメニュー >> すべてのプログラム >> TeraPad >> TeraPad」とたどっても起動することができます。起動できたら、後はメモ帳と同様の使い方でテキストを入力すればOKです。

大事なのは、ファイルの保存方法です。「ファイル >> 文字/改行コード指定保存」を選びます。すると、



というウインドウが現れます。ここで文字コードとして「UTF-8N」を選び、改行コードとして「CR+LF」を選んでOKボタンをクリックすれば、UTF-8Nという形式でファイルを作ることができます。

UTF-8 と UTF-8N の違い

UTF-8Nというのは日本のみで通用する呼称で、純粹にUTF-8の文字コードのみからなるファイルのことを指します。これに対しUTF-8というのは、ファイルの先頭にBOMと呼ばれる3バイトの識別マークが付いています。この3バイトはファイルがどんなコード系を使っているのか判定するためだけに用いられます。実際日本にはShift-JIS、UTF-8、EUC、JIS(ISO2022-JP)などさまざまなコード系がありますから、ファイルの先頭に付いているBOMを見るだけで、どのファイルがどんなコード系かを判定できるというのは非常に都合がよいわけです。

ところがPHPスクリプトにBOMが付いていると問題が起こります。6章で学習しますが、ヘッダを用いてページを作る場合、ヘッダはとにかく何よりも先頭になければならないのです。しかし、このBOMと呼ばれる識別マークはファイルの1文字目よりも更に先頭に付いてしまいます。このため、ヘッダを用いたページはエラーとなってしまいます。6章で作った正しく動作するPHPスクリプトをUTF-8とUTF-8Nとでそれぞれ保存して、結果を比較してみて下さい。

MySQL

まず文字コードセットを日本語対応に設定します。MySQL自体は文字コード系としてShift-JIS、UTF-8、EUCなどさまざまなものに対応しています。しかし本書ではPHPとMySQLとを連携させますから、同じコード系を採用するとよいでしょう。PHPがShift-JISに対応していないこと、およびWindowsで簡単にUTF-8のファイルが作れることから、本書ではコード系はUTF-8に統一することにします。

また、ご自身のパソコンをテスト環境とし、付録iiで示した一括インストールに従って必要なサーバソフトウェアをインストールしているものとして話を進めます。ご自身で個別にソフトウェアをインストールしている場合はフォルダ名などを適宜読み替えて下さい。

文字コードセットを確認する

MySQLの内部状態を表示するにはSTATUS問い合わせ文を使います。MySQLツールを立ち上げて、STATUSと打ち込んで下さい。次のように表示されるはずです。

また、MySQLの文字コードセットはサーバ変数と呼ばれるサーバの内部的な変数に格納されています。この変数の値を見ることで、MySQLの文字コードセットがどのように設定されているかを確認することができます。次の問い合わせ文を叩いてみて下さい。

```
SHOW VARIABLES LIKE "char%";
```

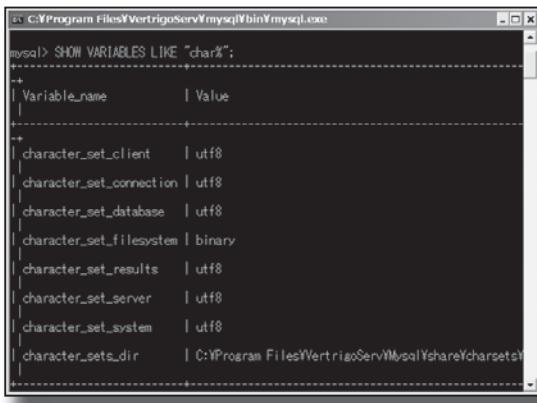
この問い合わせ文は「charで始まるすべてのサーバ変数の値を表示させる」という意味です。詳しくは9章で学習します。これで文字コードに関するサーバ変数を表示させることができます。次のような結果が表示されるはずです。

注目すべきは文字コードセット(characterset)または、そのサーバ変数の値で、付録iiで示した一括インストールだと、このようにlatin1という文字コードセットがデフォルトとして設定されています。この文字コードセットには日本語を格納することができません。無視して(または知らずに)テーブルを作りデータを格納し、表示させると????という具合に表示されます。逆に言えば、日本語をINSERTして、SELECTで表示させたときに????と表示された場合(または空白が表示された場合)それはコード系を正しく設定していないことが原因として考えられます。

もし次のように表示されていれば、かなり幸せです。なぜならテーブルを作るとき、データを格納するとき、データを表示させるときなど、いちいち文字コードセットをUTF-8と指定しなくても済むからです。

```
c:\> C:\Program Files\Vertrigo\Server\MySQL\bin\mysql.exe
mysql> STATUS
+-----+-----+
| Connection_id: | 1
| Current_database: | 
| Current_user: | root@localhost
| SSL: | Not in use
| Using delimiter: | ;
| Server version: | 5.0.51b-community MySQL Community Edition (GPL)
| Protocol version: | 10
| Connection: | localhost via TCP/IP
| Server characterset: | latin1
| Db characterset: | latin1
| Client characterset: | latin1
| Conn. characterset: | latin1
| TCP port: | 3306
| Uptime: | 17 sec
+-----+
Threads: 1 Questions: 4 Slow queries: 0 Opens: 12 Flush tables: 1 Open tables: 8 Queries per second avg: 0.285
mysql>
```

```
c:\> C:\Program Files\Vertrigo\Server\MySQL\bin\mysql.exe
mysql> SHOW VARIABLES LIKE "char%";
+-----+-----+
| Variable_name | Value
+-----+-----+
| character_set_client | latin1
| character_set_connection | latin1
| character_set_database | latin1
| character_set_filesystem | binary
| character_set_results | latin1
| character_set_server | latin1
| character_set_system | utf8
| character_sets_dir | C:\Program Files\Vertrigo\Server\MySQL\share\charsets
+-----+
```



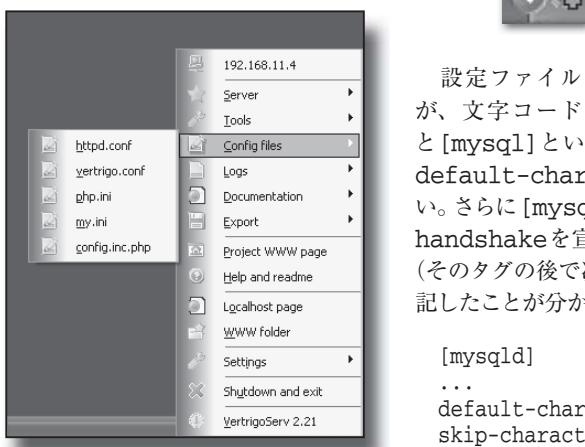
```
mysql> SHOW VARIABLES LIKE "char%";
```

Variable_name	Value
character_set_client	utf8
character_set_connection	utf8
character_set_database	utf8
character_set_filesystem	binary
character_set_results	utf8
character_set_server	utf8
character_set_system	utf8
character_sets_dir	C:\Program Files\VertrigoServ\MySQL\share\charsets

これらの変数は、クライアント側の文字コードセット、データを取り扱うときの文字コードセット、データベースに格納されるデータの文字コードセットなどのデフォルト設定を規定しています。実はここで表示されているデフォルトの文字コードセットは、MySQLを立ち上げる際に設定ファイル(config file)というものを参照して決まります。ではどうすればこちらのようにUTF-8だけにできるかを次に説明します。

文字コードセットをUTF-8 だけにする

MySQLのデフォルト文字コードセットの設定は設定ファイル(Config file)を書き換えることで、変更できます。MySQLの設定ファイルはmy.iniまたはmy.cnf(Windowsではmy.iniという名前ですが、どちらも同じものです)という名前です。付録iiで示した一括インストールの場合、Windwosでは、C:\Program Files\VertrigoServ\MySQL\にあります。Macでは/Applications/XAMPP/etc/にあります。これをメモ帳などのエディタで開きます。WindowsにVertrigoServをインストールした場合は、サーバ起動後に右下のタスクトレイに現れるアイコンを右クリックし、Config filesからmy.iniを選ぶこともできます。



設定ファイルは、いくつかのセクションからなっていますが、文字コードに関するセクションは [mysqld] と [mysqldump] と [mysql] というタグで始まる3つのセクションです。ここで default-character-set に utf8 という値を設定して下さい。さらに [mysqld] には skip-character-set-client-handshake を宣言しておきます。場所は、そのセクションの中(そのタグの後で次のタグの前)であればどこでも構いません。追記したことが分かるようにセクションの最後に置いておきます。

```
[mysqld]
...
default-character-set=utf8
skip-character-set-client-handshake
```

これで、[mysqld] セクションのデフォルト文字コードセットがUTF-8に設定されます。その次の宣言 skip-character-set-client-handshakeは、文字コードの自動変換を抑止するためのものです。(まあ、おまじないと思って頂いて結構です。)

以下同様に残り2つのセクションでもデフォルト文字コードセットをUTF-8に設定します。

```
[mysqldump]
...
default-character-set=utf8
```

```
[mysql]
...
default-character-set=utf8
```

次のような感じに書き加えて
おけばよいでしょう。

```
[mysqld]
basedir=C:\Program Files\VertrigoServ\MySQL\
datadir=C:\Program Files\VertrigoServ\MySQL\data\
port = 3306
key_buffer = 16M
max_allowed_packet = 1M
table_cache = 32
sort_buffer_size = 512K
net_buffer_length = 8K
read_buffer_size = 256K
read_rnd_buffer_size = 512K
myisam_sort_buffer_size = 8M

default-character-set=utf8
skip-character-set-client-handshake

[mysqldump]
quick
max_allowed_packet = 16M
default-character-set=utf8

[mysql]
no-auto-rehash
# Remove the next comment character if you are not
familiar with SQL
#safe-updates
default-character-set=utf8
...
```

以上の操作を終えたら、設定ファイルを上書きします。次にMySQLサーバを再起動して下さい。設定ファイルはサーバ起動時に一度だけ読み込まれますので、設定ファイルを変更した後はサーバを再起動しないと変更が反映されません。WindowsにVertrigoServをインストールした場合は、サーバ起動後に右下のタスクトレイに現れるアイコンを右クリックし、Server >> Restartです。



MacにXAMPPをインストールした場合は、XAMPP controlsの画面中央にあるMySQLを一度stopし、再度startすればOKです。このとき管理者パスワードを要求されますので、ちゃんと入力します。



以上すべての操作が終わったら、もう一度文字コードセットを確認してみましょう。コマンドを覚えていますか？

`SHOW VARIABLES LIKE "char%";`
です。次のように表示されれば第1段階終了です！

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> SHOW VARIABLES LIKE "char%";
+-----+-----+
| Variable_name      | Value
+-----+-----+
| character_set_client | utf8
| character_set_connection | utf8
| character_set_database | utf8
| character_set_filesystem | binary
| character_set_results | utf8
| character_set_server | utf8
| character_set_system | utf8
| character_sets_dir   | C:\Program Files\VertrigoServ\MySQL\share\charsets\
```

日本語データを入力してみる

では本文2章に戻って、PhpMyAdminか、MySQLターミナルを使い、日本語データを投入してみましょう。日本語を入力してみて、正しく日本語が表示されていれば成功です。???と表示されたり、入力したはずのカラムが空白で表示された場合は、文字コードの設定が間違っています。再度本章をよく読んで設定を確認して下さい。ただしMySQLターミナルを使う場合は、以降の注意事項をよく読んで下さい。特段の事情がなければ、PhpMyAdminのSQLタブを使う方が良いでしょう。

ターミナルをセットアップする

以下のセットアップはMySQLターミナルを使用する場合にのみ必要です。phpMyAdminだけでも特に困ることはありませんので、よくわからない場合はインストールしなくても大丈夫です。

Windowsにはコマンドプロンプト（いわゆDOS窓）というターミナルプログラムが用意されています。Macではターミナルと言います。これらはMySQLのターミナルとして使うと少々問題があります。Windowsコマンドプロンプトの場合、SELECT問い合わせ文による日本語表示がずれ、カラム名と値との対応が見にくくなります。

Macターミナルの場合はもう少し深刻で日本語の入力ができません（文字化けします）。入力された日本語の表示はできます。入力も行うためには、iTermというフリーのターミナルソフトを使うことで解決できます。

MySQL ターミナルで日本語を入力する場合の注意事項

Windows上のMySQLターミナル(VertrigoServではTools >> MySQL Consoleという名前になっています)で日本語を入力するには2つの小技が必要です。まず第1に入力する際のコード系をShift-JISに変更してあげる必要があります。誠に残念なことにWindowsが標準で使っている文字コード系はShift-JISであり、これは変更できません。ターミナルはWindowsアプリケーションですから入力時の文字コードはShift-JISになっています。このため先ほどUTF-8で統一したコード系のうち、ターミナルからの入力については、一時的にShift-JISに変更してあげる必要があります。ターミナルで次のように叩いて下さい。

```
SET NAMES cp932;
```

ここでcp932というのはShift-JISの別名(Microsoft等による拡張を含む文字セット)です。これでターミナルからShift-JISを入力することができ、データベースにはUTF-8でぶっ込まれるようになります。またデータベースからUTF-8の文字データを引っ張ってきててもShift-JISに逆変換して表示してくれます。つまりターミナルからデータベースに出し入れしても文字化けは起こらなくなります。

SELECT問い合わせ文でカラムがずれます

日本語を含むデータに対してターミナル上でSELECT文を実行すると、カラムがずれて表として見るに堪えないものが表示されます。例えばこんな感じです。

```
C:\Program Files\VertrigoServ\mysql\bin\mysql.exe
mysql> SET NAMES cp932;
Query OK, 0 rows affected (0.01 sec)

mysql> select last_name, first_name, how_long from aliens_abduction limit 2;
+-----+-----+-----+
| last_name | first_name | how_long |
+-----+-----+-----+
| 水島      | 理恵      | 約4年    |
| 奥田      | 裕          | 37秒    |
+-----+-----+-----+
2 rows in set (0.00 sec)

mysql>
```

残念ながら、これについては2009年11月時点ではWindows上では対応策を見つけることができませんでした。表示された日本語の文字数分だけカラムがずれていきます。Macでは、iTermというフリーのターミナルソフト上でMySQLターミナルを起動すれば、カラム位置がずれることはありません。特段の事情がなければ、PhpMyAdminを利用することをお勧めします。

索引

数字・記号

- 1対1の関係 374, 375
- 1対多くの関係 374, 375
- 2つのマイナス記号 (--) 282
- 3項演算子 391, 395, 431
- ! (否定演算子) 142, 182
- % (パーセント記号) 437
- && (論理積演算子) 143, 182
- _ (アンダースコア) 26
 - ワイルドカード 437
 - . (ピリオド) 32, 33
 - < (本当に小さい) 136, 182
 - \$ (ドル記号) 25, 26
 - \$ (メタ文字) 504
 - * (アスタリスク) 58, 106
 - /* と */ 283
 - ;(セミコロン) 101
 - MySQL 52, 55
 - PHP 25
 - SQL文 89
 - @ (エラー抑止演算子) 223, 240
 - ^ (メタ文字) 504
 - || (論理和演算子) 143, 182
 - == (等号) 135
 - <? タグ 27
 - <> (等しくない) 136, 182
 - > (本当に大きい) 136, 182
 - >演算子 574, 576
 - >= (等しいかまたは大きい) 136, 182
 - ¥ (バックスラッシュ) 38

A

- action 属性 14
- ADD COLUMN 文 192, 243
- addemail.php スクリプト 102-107

B

- ALTER 文 101, 195, 206
- ALTER TABLE コマンド 171, 182, 192, 243, 270
- array_push() 関数 385
- AS キーワード 431
- AUTO_INCREMENT 171

C

- CASE ラベル 474-476
- CHANGE COLUMN コマンド 192
- CHAR データ型 92-94
- CHARACTER データ型 92
- checkdnsrr() 関数 531, 532
 - 注意 531
- children() メソッド 575
- Content-Type ヘッダ 257
- \$_COOKIE 変数 318, 324, 356
- CREATE データベースコマンド 52, 88, 89
- CREATE TABLE コマンド 64-65, 88, 95-96

D

- ¥d (メタ文字) 504
- DATE データ型 92-94
- DATETIME データ型 92-94
- DEC データ型 92
- DEFAULT 文 289
- DELETE コマンド 121-127, 128
 - WHERE 句 122-123
 - 間違って削除 123

DELETE FROM 文	237, 243
LIMIT 句	241
DESCRIBE コマンド	99, 128
die() 関数	68
DROP COLUMN コマンド	192
DROP TABLE コマンド	100, 128

E

echo コマンド	24, 33
フォームを再生成	154
else 句	148-152, 182
empty() 関数	140, 182, 498
ENUM データ型	273
exit() 関数	255, 259, 263, 289
explode() 関数	442, 450, 492

F

\$_FILES 変数	199, 206, 243
アップロードエラー	223
for ループ	424-425, 431
foreach ループ	177-180, 182
配列	178
<form> タグ	6
action 属性	14
FROM 部分 (SELECT 文)	58
FTP (File Transfer Protocol) ユーティリティ	19

G

\$_GET	230
get メソッド	6
GET リクエスト	230-236
GW_MAXFILESIZE 定数	207, 221-224
GW_UPLOADPATH 定数	207

H

header() 関数	253, 289
hidden フォームフィールド	241
HTML	
PHP から出たり入ったりする	155
PHP コード	15

PHP と HTML のコードを同じファイルに置く	27
PHP を使う	3
クライアント	10
静的な言語	2
動的な HTML ページ	3
フォームデータをメールで送る	10
HTTP 認証	247-251, 290, 339
Basic 認証	259
クッキーの違い	316
クッキーを使う	321
ヘッダ	250-257
ヘッダによる認証	254-255
ユーザログイン	303

I

\$i 変数	218
if 文	134-138, 182
3 項演算子	391, 395
else 句	148-152
HTML フォームを作る	157
きれいなコード	150-152
テスト条件	136
ネストされた	149
implode() 関数	445, 492
include_once 文	243
<input> タグ	6, 30, 190, 206
INSERT 文	66-67, 198, 285
試運転	57
INSERT INTO 文	54
INT 型	92-94
INTEGER データ型	92-94
IP アドレス	51
is_numeric() 関数	289
isset() 関数	140-142, 164, 182, 498

J

JPEG イメージと MIME タイプ	224
---------------------	-----

L

LIKE 節	437-441, 492
LIMIT 句	243, 481-482, 492

DELETE FROM 文	241
Location ヘッダ	257

M

mailto	6, 9, 10
mb_send_mail() 関数	42-43, 110
サーバ設定	44
MD5() 関数	301
MIME タイプと JPEG イメージ	224
MODIFY COLUMN コマンド	192
move_uploaded_file() 関数	201, 203, 204, 206
MySQL	
CREATE データベースコマンド	52
CREATE TABLE コマンド	52-53
INSERT 文	54-55
試運転	57
INSERT INTO 文	54
MySQLとつなげる	49-80
mysqli> プロンプト	52
PHP 関数	66-68
die()	68
mysqli_close()	66-67
mysqli_connect()	66
mysqli_query()	66-67
PHP スクリプト	64
PHP スクリプトでデータを挿入する	65
PHP スクリプトとフォーム	61-63
\$_POST がフォームのデータを供給	71-74
SELECT 文	
WHERE 句	76-77
アスタリスク (*)	58
すべての内容を持ってくる	58-59
SELECT 文の FROM 部分	58
USE コマンド	52
VALUES キーワード	54
値の順序	54-55
サーバ	
コンタクトをとる	88
場所	51, 64
ターミナル	50, 56, 88
データベースサーバ	63
データベース名	64
データを選別する	76-77

テーブル	63
問い合わせ文	66-67
バージョンの確認	596
パスワード	51, 64
ユーザ名	51, 64
要件	50-51
mysql> プロンプト	52
mysqli_close() 関数	66-67
mysqli_connect() 関数	66
mysqli_fetch_array() 関数	111-116, 128
while ループ	113-116
mysqli_query() 関数	66-67, 68, 101, 111
mysqli_real_escape_string() 関数	284, 288, 289
mysqli 関数を後方互換で mysql 関数に対応	590-591

N

¥n (改行文字)	39
NOT NULL	171
NOW() 関数	198

O

ON キーワード	412
ORDER BY 句	212, 243, 464-466, 477-478

P

PHP	4, 590-591
HTMLから出たり入ったりする	155
PHP と HTML のコードを同じファイルに置く	27
PHP は何の略	15
アプリケーションのビルトとテスト	594
規則	25
サーバ	11
サーバにインストールされているかチェックする	19
日本語スクリプトを作る	608
日本語対応	607
バージョンの確認	595
フォームデータをメールで送る	9-14
ブラウザ	34
PHP 関数	66-68
変数の妥当性を検証	140
.php 拡張子	25

PHP スクリプト	
action 属性	14
AliensAbductedMe.com を分解	24
MySQL に接続する	64, 65
サーバ	12-13
サーバ上で動かす	18
サーバが変換	22-23
サーバに送り込む	19
フォームデータにアクセス	16
フォームと MySQL 問い合わせ文	61-63
<?php> タグ	16, 24
内側の空白	253
PHP を使ったメールのフォーマットと送信	35-44
 タグ	38
HTML フォーマット	37
PHP でメッセージ本文を作成	36
エスケープ文字	38
改行文字 (¥n)	39
二重引用符の文字列	39
変数にメールの要素や部品を格納	41
php.ini ファイル	206
phpMyAdmin	50, 53, 56
PHP & MySQL 道具箱	
3 項演算子	431
< (本当に小さい)	182
<> (等しくなり)	182
! (否定演算子)	182
&& (論理積演算子)	182
(論理和演算子)	182
== (等号)	135
> (本当に大きい)	182
>= (等しいかまたは大きい)	182
ADD COLUMN 文	243
ALTER TABLE コマンド	182, 243
AS キーワード	431
checkdnsrr() 関数	532
\$_COOKIE	356
DEFAULT 文	289
DELETE コマンド	128
DELETE FROM 文	243
DESCRIBE コマンド	128
DROP TABLE コマンド	128
echo	46
else 句	182
empty() 関数	182
exit() 関数	289
explode() 関数	492
\$_FILES	243
for ループ	431
foreach ループ	182
header() 関数	289
HTTP 認証	290
if 文	182
implode() 関数	492
include_once 文	243
is_numeric() 関数	289
isset() 関数	182
LIKE 節	492
LIMIT 句	243, 492
mb_send_mail()	46
MySQL	46
mysqli_fetch_array() 関数	128
mysqli_real_escape_string() 関数	289
ORDER BY 文	243
PHP	46
PHP スクリプト	46
<?php ?>	46
\$_POST	46
preg_match() 関数	532
preg_replace() 関数	532
require_once 文	243
require 文	243
REST リクエスト	587
RSS	587
SELECT * FROM コマンド	128
\$_SERVER	289
\$_SESSION	356
session_destroy() 関数	356
session_start() 関数	356
setcookie() 関数	356
SHA() 関数	356
simplexml_load_file() 関数	587
SimpleXMLElement オブジェクト	587
SQL	46
SQL 注入	290
str_replace() 関数	492
substr() 関数	492
switch-case	492

trim()関数.....	289
WHERE句.....	128, 243
whileループ.....	128
XML.....	587
イメージフォルダ.....	243
エスケープ文字.....	46
外部キー.....	431
カスタム関数.....	492
カラムと値を指定する問い合わせ文.....	290
クライアント側.....	46
サーバ側.....	46
スキーマ.....	431
正規化.....	431
正規表現.....	532
ダイヤグラム.....	431
内部結合.....	431
名前空間(XML).....	587
人間による検査.....	290
配列.....	46
フォームの妥当性検証.....	290
変数.....	46
メタ文字.....	532
文字のクラス.....	532
論理演算子.....	182
\$_POST.....	110, 230
MySQLにフォームのデータを供給.....	71-74
スーパーグローバル.....	28
配列.....	29
フォームが提出されたかどうかをチェック.....	164
postメソッド.....	6
POSTリクエスト.....	230-236
preg_match()関数.....	516-518, 532
preg_replace()関数.....	520-522, 532

R

Refreshヘッダ.....	257
regex.....	502
removeemail.php.....	179-180
require_once文.....	209-211, 240, 243
require文.....	243
RESTリクエスト.....	558-563, 587
組み立て.....	562
\$result変数.....	111

RSS.....	587
RSSアイコン.....	552
RSSシンジケーション.....	536-552
RSSフィードへのリンク.....	552
RSSフィードを動的に作る.....	548
XML.....	537, 545
データベースからRSSニュースリーダへ.....	542
RSSニュースリーダ.....	536, 538
データベースから.....	542
RSSフィード.....	536
イメージ.....	546
動的に作る.....	548
リンク.....	552
RSSを暴く.....	547

S

¥s(メタ文字).....	504
SELECT文	
FROM.....	58
WHERE句.....	76-77
アスタリスク(*).....	58, 106
すべての内容を持ってくる.....	58-59
SELECT * FROM コマンド.....	110, 111, 128
sendemail.phpスクリプト.....	109-119
mb_send_mail()関数.....	110
mysqli_fetch_array()関数.....	111-116
whileループ.....	113-116
mysqli_query()関数.....	111
\$_POST配列.....	110
\$result変数.....	111
自己参照スクリプト.....	163, 167
妥当性検証.....	131-133
背後にいる論理.....	139
フィードバック.....	147
\$_SERVER変数.....	289
アプリケーションにセキュリティを与える.....	248
\$_SERVER['PHP_SELF'].....	162, 241
session_destroy()関数.....	332, 334, 356
session_start()関数.....	332, 334, 337-339, 356
\$_SESSION変数.....	333, 337-338, 356
setcookie()関数.....	318, 356
ログアウト.....	327-328
SHA()関数.....	300-302, 356

パスワードの比較	301
Shift-JIS	608
SID スーパーグローバル	345
simplexml_load_file() 関数	564, 574, 587
SimpleXMLElement オブジェクト	576, 587
SQL	46, 49
SQL注入	283-288, 290
SQL文とセミコロン (;)	89
str_replace() 関数	452, 492
substr() 関数	460-462, 492
SUBSTRING() 関数	462
SWITCH 文	474-476

T

TEXT データ型	92
TIME データ型	92-94
TIMESTAMP データ型	92-94
TINYINT 型	270
trim() 関数	284, 288, 289
type 属性	6

U

UFOによる誘拐サイト	534-588
HTML フォーム	5-9
問題	8-9
MySQL データベースとつなげる	60-80
PHP スクリプトを分解	24
RSS シンジケーション	536-552
YouTube ビデオのシンジケーション	554-588
REST リクエスト	562-563
ビデオを見やすく配置	578-579
試運転	7, 17, 20, 34, 44
INSERT 文	57
MySQL データベースにデータを挿入	69
RSS ニュースフィードスクリプト	551
WHERE 句	77
YouTube スクリプト	584
YouTube リクエスト URL	563
すべての内容を持ってくる	59
ニュースフィードへのリンクを追加	553
フォームデータと INSERT 文	75
メール受信	45

メールメッセージを組み立てる	40
unlink() 関数	223
UPDATE コマンド	195
USE コマンド	52, 97-98
user_id	297
USING キーワード	412

V

VALUES キーワード	54
値の順序	54-55
VARCHAR データ型	92-94

W

¥w (メタ文字)	504
Web アプリケーション	
個人向け	291
定義	83
Web コンテンツ	
プッシュする	535
RSS	536-552
他のサイトから引っ張ってくる	556
Web サーバ	サーバを参照
Web サーバからファイルを削除	223
Web フォーム	フォームを参照
WHERE 句	76-77, 128, 243
DELETE コマンド	122-123
空の検索要素	455
内部結合	411
while ループ	113-116, 128

X

XML	587
RSS シンジケーション	537, 545
RSS フィードを動的に作る	548
XML データをオブジェクトで掘り起こす	574
YouTube ビデオのシンジケーション	566
レスポンスを分解する	570
オブジェクトでアクセス	572
オブジェクトの集まり	573
階層構造	571
実体 (エンティティ)	569

名前空間 569, 575

Y

YouTube ビデオのシンジケーション 554-588
 REST リクエスト 558-563
 組み立て 562
 simplexml_load_file() 関数 564, 574
 XML 566
 階層構造 571
 名前空間 569
 実体 (エンティティ) 569
 レスポンスを分解する 570
 ビデオの再生時間 579
 ビデオを見やすく配置 578-579
 リクエスト・レスポンスによるコミュニケーション
 プロセス 557

あ行

アスタリスク (*) 58, 106
 アップロードファイルを入れる場所 201
 アトミックデータ 398-399
 正規化 401
 アプリケーション
 個人向け 291
 設計 84
 定義 83
 アプリケーションにセキュリティを与える 245-290
 Content-Type ヘッダ 257
 header() 関数 253
 HTTP 認証 247-251
 Basic 認証 259
 ヘッダ 250-257
 ヘッダによる認証 254-255
 HTTP 認証ではなくクッキーを使う 321
 INSERT (パラメタ付き) 285
 Location ヘッダ 257
 <?php ?> タグの内側の空白 253
 Refresh ヘッダ 257
 \$_SERVER 変数 248
 SQL 注入 283-288
 クッキー 316-330
 コミュニティ Web サイト 314

コメントで MySQL をごまかす 282
 デフォルトのカラムの値 285-286
 人間による検査 268-269
 第 1 ステップ 270
 第 2 ステップ 272
 第 3 ステップ 274
 第 4 ステップ 275
 認証用スクリプト 262-265
 フォームの妥当性検証 287
 リバースエンジニア 264
 アポストロフィー 39
 アンダースコア (_) 26
 ワイルドカード 437
 一意に識別 170-173
 一時的な永続性 317
 一時フォルダ 200
 イメージ
 RSS フィード 546
 フォルダ 198-202, 206, 243
 イメージを格納 185-206
 アップロードファイルを入れる場所 201
 一時フォルダ 200
 イメージファイルアップロードを計画 191
 イメージファイル名を挿入 198-199
 ユーザからイメージをもらう 196
 インクルードファイル 208-209, 243
 引用符 39, 65
 单一引用符二重引用符との違い 72
 永続性
 一時的 317
 セッション + クッキー 351
 短期間と長期間 352
 ユーザ 325
 エスケープ文字 38
 エラーメッセージ 222-224
 抑止 223, 240
 エラー抑止演算子 (@) 223, 240
 エルビストアプロジェクト 82, 130
 addemail.php スクリプト 102-107
 sendemail.php スクリプト 109-119
 妥当性検証 131-133
 背後にある論理 139
 フィードバック 147
 アプリケーション設計 84

空白のメールメッセージ	131
計画	86
試運転	
addemail.php スクリプト	105
DELETE コマンド	124
USE コマンド	98
きれいにした SQL コード	151
顧客のチェックボックス	181
自己参照スクリプト	163, 167
主キー	173
データベースとテーブルを作る	98
メーリングリストから顧客を削除	127
メール送信用フォームを使ってメールを送信	119
論理演算子	145
妥当性検証	132-133
チェックされた顧客を削除	179-180
フォームデータの保存	158-163
大文字	27
オブジェクト	564, 572-576
XML データにアクセス	572
XML データをオブジェクトで掘り起こす	574
集まり	573
配列との違い	576
オブジェクトの集まり	573
親テーブル	374
か行	
改行文字 (¥n)	39
開発環境の設定	593
MySQL	594
バージョンの確認	596
PHP	594
バージョンの確認	595
Web サーバの確認	595
外部キー	372-373, 431
動作	373
外部結合	416
カウントする変数	218
カスタム関数	467-492
値	468
問い合わせ文を作る	469-471
ページ割り付け	480-486
LIMIT 句	481-482
書き直した問い合わせ文の結果	485
ページ案内用リンク	486
ページ割り付けデータの追跡	483
変数	484
カスタム関数を暴露する	470
カラム	87
キー以外のカラムの依存性	401
デフォルトの値	285-286
～と値を指定する問い合わせ文	285-286
別名	413
カラム名	90
管理者用ページ	226-232
アプリケーションにセキュリティを与える	254
スコアの削除リンク	229
守る	248
キー	372-373
ギターウォーズ	184-243
アプリケーションにセキュリティを与える	246-290
HTTP 認証	247-251
SQL 注入攻撃から守る	284
管理者用ページのセキュリティ	264
人間による検査	268-269
認証用スクリプト	262-265
フォーム攻撃	278-281
ヘッダによる認証	254
イメージファイルアップロードの妥当性検証	220-224
イメージファイルアップロードを計画	191
第 1 ステップ	194
第 2 ステップ	197
第 3 ステップ	198
第 4 ステップ	204
第 5 ステップ	204
第 6 ステップ	211
イメージファイル名を挿入	198-199
イメージフォルダ	198-202
管理者用ページ	226-232
スコアの削除リンク	229
守る	248
削除するために ハイスコアをより分ける	234-237
試運転	
Screenshot カラムを追加	195
イメージフォルダ	205
インクルードファイル	211
管理者用スクリプトに HTTP 認証を追加	259

最高スコアをショーケースに入れる	219
承認用スクリプトを作る	276-277
スクリーンショットのイメージファイルの 妥当性検証	224
スコア削除用スクリプトと管理者用スクリプト	242
スコア追加用スクリプトのフォームデータ 処理を強化	288
認証用スクリプト	265
証拠のないスコア	225
ダウンロードして下さい!	188
トップスコアの勝者	215-219
トップスコアのフォーマット	216
ハイスコアデータベースの変更	192
ユーザからイメージをもらう	196
行	87, 90
一意に識別	170-173
共有スクリプトデータ	208-209
空白	
<?php ?> タグの内側	253
変数名	26
メールメッセージ	131
クエリ	問い合わせ文を参照
クッキー	316-330
HTTP認証ではなくクッキーを使う	321
HTTP認証との違い	316
PHPで使う	318
削除	327-329
寿命	348-352
セッション	345
セッションとの違い	342-343
セッションに移住	340-341
定義	316
データのサイズ	352
中にあるもの	317
ユーザIDとユーザ名を格納	323
ユーザログイン	322-323
クライアントHTML	10
クライアント側	46
結合	409-417
USING キーワード	412
外部結合	416
自然結合	416
等結合	416

ドット記法	410
内部結合	409, 411
非等結合	416
結合テーブル	376
検索結果をソート	464-466, 472-473, 477-478
後方互換	590-591
効率的なコーディング	391
コードをシンプルにする	149-152
個人向けのWebアプリを作る	291
新しいユーザをサインアップ	307-313
コミュニティ Web サイト	293
セキュリティ	314
ログアウト	327-329
子テーブル	374
コマンドと大文字・小文字	27
コミュニティ Web サイト	293, 314
コメント	
2つのマイナス記号 (-)	282
MySQLをごまかす	282
複数行	283
小文字	27

さ行

サーバ	
PHP	11
インストールされているかチェックする	19
PHPスクリプト	12-13
送り込む	19
動かす	18
変換	22-23
確認	595
サーバ側	46
サインアップ	307-313
参照整合性	373
視覚的なセキュリティ	299
自己参照フォーム	161-163, 166-167
自然結合	416
実体 (エンティティ) (XML)	569
重要ポイント	
<input> タグ	206
ALTER 文	206
CREATE データベースコマンド	116
CREATE TABLE コマンド	116

die() 関数	68	主キー	401
DROP TABLE コマンド	116	利点	400
exit() 関数	263	正規表現	493-532
\$_FILES 変数	206	checkdnsrr() 関数	531, 532
header() 関数	263	注意	531
HTTP 認証	339	preg_match() 関数	516-518, 532
move_uploaded_file() 関数	206	preg_replace() 関数	520-522, 532
mysqli_close() 関数	68	数量子	509
mysqli_connect() 関数	68	定義	502
mysqli_fetch_array() 関数	116	電話番号パターン	500-501, 505-509
mysqli_query() 関数	68	標準化	523
\$_SERVER 変数	263	不要な文字を捨てる	524
while ループ	116	メールアドレスパターン	529
イメージフォルダ	206	ドメインのチェック	531
クッキー	339	メタ文字	504-509, 532
セッション	339	文字のエスケープ	512-514
セッション変数	339	文字のクラス	510-511, 532
データベース接続	68	予約文字	512-514
ヘッダ	263	セッション	330-345
主キー	171-173, 372-373	+クッキー	351
5つの規則	172	クッキーから移住	340-341
正規化	401	クッキーとの違い	342-343
条件式	134-138	クッキー無効	345
条件をテスト	113	寿命	348-352
シンジケーション		データのサイズ	352
RSS	536-552	ログアウト	335-336
YouTube ビデオ	554-588	セッション変数	331, 333, 339, 348
スーパーグローバル	28	セミコロン(;)	101
数量子	509	MySQL	52, 55
スキーマ	367-371, 431	PHP	25
矢印(記号)	372	SQL文	89
向き	374	素朴な疑問に答えます	
スクリプト		<? タグ	27
require_once 文	209-211	->演算子	576
インクルードファイル	208-209	3項演算子	395
共有スクリプトデータ	208-209	ALTER 文	101, 195
コミュニケーションをとる	230	array_push() 関数	385
矢印(記号)	372, 374	CHAR データ型	93
リバースエンジニア	264	DESCRIBE コマンド	99
スコア追加用フォームをよく見る	197	die() 関数	68
正規化	398-404, 431	else 句	151
3ステップ	401	empty() 関数	498
アトミックデータ	401	ENUM データ型	273
キー以外のカラムの依存性	401	exit() 関数	259

for ループ	425
GET と POST	232, 236
GW_MAXFILESIZE	224
HTTP 認証	249
Basic 認証.....	259
isset() 関数	141, 498
MIME タイプと JPEG イメージ	224
mismatch_category テーブル	404
MySQL ターミナル	56
mysqli_fetch_array()	116
mysqli_query() 関数.....	68
Null, Key, Default, Extra カラム	99
php.ini ファイル	206
PHP コードと HTML コード	15
PHP スクリプトから HTML フォーマットを使う	37
PHP と HTML のコードを同じファイルに置く	27
PHP は何の略	15
phpMyAdmin	56
\$_POST	72
require_once.....	209
REST	558, 561
RSS	538
RSS フィードとイメージ	546
RSS リーダ	538
SELECT コマンド	106
session_start() 関数.....	339
SHA() 関数.....	301
SimpleXMLElement オブジェクト	576
SQL コメント	283
SQL 注入攻撃	283
substr() 関数	462
UPDATE コマンド	195
user_id	297
VARCHAR データ型	93
XML	546
アトミックデータ	399
永続性が短期間か長期間か	352
エスケープ文字	38
オブジェクトと配列	576
管理者用ページのセキュリティ	264
共有スクリプトファイル	209
クッキー	317
削除	329
ユーザ ID とユーザ名を格納	323

結合	412, 416
検索文字列	451
コマンドと大文字・小文字	27
サインアップ用スクリプト	311
視覚的なセキュリティ	299
証拠のないスコア	215
数値型	93
正規化	403
セッション	339
セミコロン	101
妥当性検証	133
单一引用符	39
单一引用符と二重引用符の違い	72
データの格納場所	87
データのサイズと永続性	352
データベースとテーブルの比較	56
データベースの関係	375
テスト条件	138
テンプレート	359
電話番号のパターン	501
名前空間 (XML)	571
二重引用符の文字列	39
ファイルの上書き	206
フォームが提出されたか	164
ページ割り付け	490
変数	27, 209
変数と連結	72
文字のクラス	511
文字列と変数の連結	34
論理演算子の順番	145
ダイヤグラム	367, 431
ダウンロードして下さい!	
ギターウォーズ	188
ミスマッチアプリケーション	296, 394
リスクジョブアプリケーション	519

た行

多対多の関係	374-376
妥当性検証	132-133
if 文	134-138
else 句	148-152
HTML フォームを作る	157
きれいなコード	150-152

テスト条件.....	136	多対多の関係.....	374-376
ネスト.....	149	作って使う.....	81
PHP 関数で変数の妥当性を検証.....	140	テーブル	テーブルを参照
sendemail.php	139	テーブルとの違い.....	56
イメージファイルアップロード.....	220-224	テーブルの結合.....	376
エラーメッセージ.....	222-224	名前.....	64
抑止.....	223, 240	場所.....	51
サーバ側とクライアント側の違い.....	133	パスワード.....	51
正規表現.....	493-532	ユーザ名.....	51
背後にある論理.....	133	データベースを作って使う	
フォーム.....	287	addemail.php スクリプト	102-107
複数の条件をテスト.....	143-146	ALTER 文	101
フラグと重複コード.....	156	CREATE データベースコマンド.....	88, 89
論理演算子.....	143-146	CREATE TABLE コマンド.....	88, 95-96
順番.....	145	DELETE コマンド	121-127
单一引用符.....	39, 65	WHERE 句.....	122-123
二重引用符との違い.....	72	間違って削除.....	123
チェックボックス.....	177	DESCRIBE コマンド	99
重複コード.....	156	DROP TABLE コマンド	100
重複コードをなくす.....	357-362	mb_send_mail() 関数	110
テンプレート.....	358-361	MySQL サーバにコンタクトをとる	88
提出ボタン.....	6	SELECT コマンド	106
データ型.....	91, 92	SELECT * FROM コマンド	110, 111
効率的.....	389	sendemail.php スクリプト	109-119
データ駆動型フォーム.....	386-396	mysqli_fetch_array() 関数	111-116
効果的なデータ型.....	389	mysqli_query() 関数	111
データのコントロール	363-432	\$_POST 配列	110
結合.....	409-417	\$result 変数	111
正規化.....	398-404	USE コマンド	97-98
データ駆動型フォーム.....	386-396	カラム	87
別名.....	413	カラム名	90
データの削除.....	123	行	87, 90
データの定義.....	91	セミコロン (;)	101
データの前処理.....	450-451	データ型	91, 92
データベース.....	49	データの格納場所	87
1 対 1 の関係.....	374-375	データの定義	91
1 対多くの関係.....	374-375	データベースとテーブルを作る	86-101
構造的な変更と問い合わせ文	407	テーブル	
サーバ.....	49, 63	構造	99
参照整合性	373	定義	87
正規化.....	398-404	データベースの中に作る	90-91
3 ステップ	401	問い合わせ文	95-96
利点	400	はじめる	85
ダイヤグラム	367, 431	データを取り除く	121-127

テーブル	49, 63
CREATE TABLE コマンド	52
行を一意に識別	170-173
親	374
結合	376
子	374
構造	99
主キー	171-173, 372-373
ダイヤグラム	367, 431
定義	87
データベースとの違い	56
データベースの中を作る	90-91
複数テーブルと問い合わせ文	408
別名	413
テスト条件	113, 134-138
デフォルトのカラムの値	285-286
テンプレート	358-361
電話番号のパターン	500-501, 505-509
標準化	523
不要な文字を捨てる	524
問い合わせ文	66-67, 95-96
カスタム関数で作る	469-471
ちゃんとした検索語	456-457
データベースの構造的な変更	407
複数テーブル	408
等結合	416
等号 (==)	135
動的なHTMLページ	3
特殊文字と変数名	26
ドット記法	410
ドメインのチェック	531
ドル記号 (\$)	25, 26

な行

内部結合	409, 411, 431
名前空間 (XML)	569, 571, 575, 587
二重引用符	65
单一引用符との違い	72
文字列	39
日本語対応	606-617
ニュースフィード	536
人間による検査	268-269, 290
ALTER TABLE 文	270

第1ステップ	270
第2ステップ	272
第3ステップ	274
第4ステップ	275
認証用スクリプト	262-265
粘着力のあるフォーム	161-163, 166-167

は行

パーセント記号 (%)	437
配列	29
foreachでループ	178
オブジェクトとの比較	576
パスワード	294
HTTP認証	247
暗号化	298
SHA() 関数	300-302
視覚的なセキュリティ	299
比較	301
バックスラッシュ (\)	38
否定演算子 (!)	142, 182
ビデオの再生時間	579
等しいかまたは大きい (>=)	136
等しいかまたは小さい (<=)	136
等しくない (<>)	136
非等結合	416
ピリオド (.)	32, 33
メタ文字	504
ファイルに格納されたデータ	183-243
GW_UPLOADPATH 定数	207
php.ini ファイル	206
アップロードファイルを入れる場所	201
一時フォルダ	200
イメージ	185-206
イメージファイルアップロードを計画	191
第1ステップ	194
第2ステップ	197
第3ステップ	198
第4ステップ	204
第5ステップ	204
第6ステップ	211
イメージファイル名を挿入	198-199
イメージフォルダ	198-202
FTP プログラム	202

妥当性検証	
イメージファイルアップロード	220-224
エラーメッセージ	222-224
ファイルの上書き	206
ユーザからイメージをもらう	196
ファイルの上書き	206
フォーム	10
 タグ	33
<form> タグ	6
<input> タグ	6, 30
action 属性	14
echo でフォームを再生成	154
get メソッド	6
if 文に応じて HTML フォームを作る	157
isset() 関数	164
mailto	6, 9
PHP スクリプト	11
SQL	61-63
フォームデータをメールで送る	9-14
PHP を使ってフォームデータにアクセス	16
\$_POST	28-29
MySQL にフォームのデータを供給	71-74
Post メソッド	6
\$_SERVER['PHP_SELF']	162
type 属性	6
自己参照	161-163, 166-167
妥当性検証	287
提出ボタン	6
データ駆動型	386-396
フォームが提出されたかどうかをチェック	164
フォームデータの保存	158-163
フォームデータの保存	158-163
複数行コメント	283
複数の条件をテスト	143-146
プッシュ (Web コンテンツ)	535
部分文字列	460-462
フラグ	156
ページ割り付け	480-486
LIMIT 句	481-482
書き直した問い合わせ文の結果	485
ページ案内用リンク	486
ページ割り付けデータの追跡	483
変数	484
ヘッダ	250-257
Content-Type ヘッダ	257
Location ヘッダ	257
Refresh ヘッダ	257
ヘッダ解剖学	251
ヘッダによる認証	254-255
要注意	257
ヘッダ解剖学	251
ヘッダによる認証	254-255
ヘッダを暴露する	252
別名	413, 431
変数	24, 26, 27, 209
\$i 変数	218
スーパーグローバル	28
セッション	331, 333, 339, 348
メールの要素や部品を格納	41
文字列と変数の連結	33-34
変数名	25
完璧なものを見つける	26
他のサイトから Web コンテンツを引っ張ってくる	556
ホスト名	51
本当に大きい (>)	136
本当に小さい (<)	136
ま行	
マークアップ言語	537
間違って削除	123
マニア向け情報	
SUBSTRING() 関数	462
ビデオの再生時間	579
ミスマッチアプリケーション	292, 364-432
mymismatch.php スクリプト	427-429
新しいユーザをサインアップ	307-313
新しいログイン用スクリプトを案内	324-325
アンケート	381-395
計画	381
第 1 ステップ	385, 392
第 2 ステップ	385, 392
第 3 ステップ	393
第 4 ステップ	393
データベースにレスポンスを入れる	382
フォームを作る	392-393
クッキー	316-330
クッキーからセッションに移住	340-341

クッキー強化版ユーザログイン	322-323
結合	414
試運転	
1つの問い合わせ文のアンケート用スクリプト	417
login.phpスクリプト	304-305
mismatch_categoryデータベーステーブル	406
アンケート用スクリプト	394
クッキーからセッションへの変更	344
サインアップ機能の追加	313
セッションとクッキーを使う	355
ミスマッチ相手発見用スクリプト	430
ユーザ名とパスワードの追加	302
ログアウト	328-329
セッション	334
ログアウト	335-336
ダウンロードして下さい!	296, 394
重複コードをなくす	357-362
データ駆動型フォーム	386-390
データブレークダウン	366
データベースの正規化	405-406
テンプレート	358-362
ミスマッチ方程式	420-425
5つのステップと成功	421
forループ	424
検索の準備	422
ユーザの比較	423
ユーザコミュニティ	293
ユーザログイン	294-305
データベースを準備する	297
パスワード	294
ユーザ名	294
ログアウト	327-329
命名規則(変数)	26
メール	
HTMLを使ってフォームデータを送る	10
mb_send_mail()関数	42-43
PHPを使ったフォーマットと送信	35-44
 タグ	38
HTMLフォーマット	37
PHPでメッセージ本文を作成	36
エスケープ文字	38
改行文字(¥n)	39
二重引用符の文字列	39
変数にメールの要素や部品を格納	41
PHPを使ってフォームデータを送る	11
空白	131
組み立てる	40
受信	45
メールアドレスパターン	529
ドメインのチェック	531
メール送信用スクリプト準備完了	165
メタ文字	504-509, 532
数量子	509
文字のエスケープ(正規表現)	512-514
文字のクラス	510-511, 532
文字列操作	
LIKE節	437-441
substr()関数	460-462, 492
WHERE句	455
検索結果をソート	464-466, 472-473
ちゃんとした検索語を使った問い合わせ	456-457
データの前処理	450-451
部分文字列	460-462
不要な検索文字の置き換え	452
文字列操作関数	442-467
文字列と変数の連結	32-34
ワイルドカード文字	437
文字列操作関数	442-467
explode()関数	442, 450
implode()関数	445
str_replace()関数	452
substr()関数	460-462
文字列と変数の連結	33-34
や行	
矢印(スキーマの記号)	372
向き	374
ユーザ永続性	325
ユーザのログアウト	326-329
セッション	335-336
ユーザ名	294
ユーザログイン	294-305
HTTP認証	303
HTTP認証ではなくクッキーを使う	321
インターフェースの構築	299
データベースを準備する	297
パスワード	294

SHA()関数.....	300-302
暗号化.....	298
視覚的なセキュリティ	299
ユーザー名	294
要注意	
checkdnsrr()関数.....	531
FTP プログラム	202
mb_send_mail()関数.....	44
SQL 文とセミコロン (;)	89
値の順序	55
クッキーが無効のセッション	345
ヘッダ	257
抑止 (エラーメッセージ)	223
予約文字 (正規表現).....	512-514

ら行

リクエスト・レスポンスによるコミュニケーションプロセス	557
リスキージョブアプリケーション	434-492, 494-532
build_query()関数.....	469-471
書き直した問い合わせ文の結果	485
ソート機能.....	477-478
ページ案内用リンク	486
ページ割り付け	480-486
ページ割り付けデータの追跡.....	483
ページ割り付け用変数	484

検索用スクリプトを完成させる	489-490
試運転	
build_query()関数.....	471
explode() と implode() 関数.....	458
generate_sort_links()関数.....	478
検索用スクリプト	491
検索用フォーム	447
電話番号が妥当かチェック	519
登録用スクリプトの電話番号をきれいにする ...	526
表示されるテキストと投稿日時を制限	463
ダウンロードして下さい!.....	519
リバースエンジニア	264
ログアウト	326-329
セッション	335-336
ログイン	294-305
炉辺歎談	
GET と POST	233
クッキーとセッションとの違い	342-343
論理演算子	143-146, 182
順番	145
論理積演算子 (&&)	143
論理和演算子 ()	143
ワイルドカード	437

わ行

翻訳を終えて

「ガンかも知れないですね…」病院の診察室で唐突に医者から宣告された。最近変わったところはないですか? というような質問に対して、私が「特にダイエットなどをしているわけでもないのにちょっと痩せたかも知れない。」と答えたことに対する診断だった。ガンというのは自覚症状が全くないため「何かちょっと変だな?」と思ったときにはもう手遅れなのだろう。そのような場合、もう手の施しようがなく、余命は半年程度だそうである。もう手遅れであるのなら精密検査などをする意味もないので、この際、その半年で何がしたいか? と考えてみた。結論は、もう一度本の仕事に関わりたい! という想いだった。そんな折、オライリー社の赤池氏から、この本の翻訳の話が舞い込んできた。もちろん喜んで承諾した。

翻訳本に関わったのはこれが2度目だが、翻訳自体はこれが初めての経験である。しかも PHP も MySQL も技術的には専門外だった。しかし日本語版を作り上げるという意味では、かえってそれが幸いしたように思う。特に日本語対応についてはかなり勉強になった。同時にちまたにあふれている Web ページや書籍が如何に頼りないものであるかということも痛感した。この成果はすべて本書に反映されている。

Head First シリーズは、超初心者向けのとっつきやすい口語調の文で構成されている。これを、原文の雰囲気のままに翻訳する、ということについては実は困難を極めた。英語の単語と日本語の訳語とで文章のカタさ軟らかさが全然変わってしまうためである。特に問題なのが動詞で、例えば、英語では create、build、make、construct などといった動詞が、日本語の柔らかい単語、という意味では全部「作る」とせざるを得なかった。しかし、英語では単に store、insert となっているものを、日本語では、「ぶっ込む、ぶち込む、突っ込む、たたき込む」などと表現した。これらは開発の現場で実際に使われている用語である。またある時、口語調の Q&A コーナーを訳していると、どう考えても「英語が関西弁を話している。」としか思えないところに出くわした。仕方がないので、すべての質問を関西弁に変更し、関西在住の友人奥田裕氏に関西弁原稿の添削をして頂いた。忙しい最中、快く関西弁指導に応じてくれた彼の好意に対し記して感謝の意を表したい。ただし、関西弁としておかしな表現があったとしても、その責任はすべて私にある。以上はすべて口語調の英語を真剣に翻訳しようとした結果であり、決してふざけているのではないことをご理解頂きたいたい。と同時にそのような口語調の日本語を楽しんで頂ければ幸いである。

最後に、編集担当の赤池涼子氏には最初から最後まで(飲み会まで)お世話になった。記して謝辞に代えさせて頂きたい。

なお、あの宣告から1年半が経過しているが、幸いガンは全く姿を現していない。このところむしろ太り気味であるという普通の(幸せな?)悩みをつのらせている。

2010年2月(2月としては異様に暖かな日)

佐藤 嘉一

●訳者紹介

佐藤 嘉一(さとう よしかず)

東京生まれ。1978年 都立西高(同水泳部)卒、1983年 早稲田大学理工学部数学科(同早水会)卒、1994年 米国ペンシルバニア大学大学院コンピュータサイエンス学科修了。

沖電気住籍時代C言語処理系の開発プロジェクトに携わって以来、言語処理系の開発および形式言語理論の研究に従事。その後、IPネットワーク関連、携帯電話コンテンツ開発支援、データシンク(同期)関連、携帯電話/インターネット連携サービスなどの業務にも手を染める。

C言語標準化委員、仕様記述言語LOTOS研究会会員、Java言語標準化委員、C#言語標準化委員、CLI標準化委員、ECMAScript標準化委員を歴任。現西水会会員。

趣味：冬はスキー、春と秋はバイク、夏はビール、通年でフラメンコギターと宴会幹事。

特技：水泳(2009年現在さいたま市浦和地区水泳大会6連覇中、大会記録保持)

愛車：Honda CBR1000RR'07 Repsol Fireblade(逆輸入車)

Head First PHP & MySQL

頭とからだで覚えるWebアプリケーション開発の基本

2010年3月22日 初版第1刷発行

著 者 Lynn Beighley (リン・ペイフリー)
Michael Morrison (マイケル・モリソン)

訳 者 佐藤 嘉一 (さとう よしかず)

発 行 人 ティム・オライリー

制 作 ピーンズ・ネットワークス

印刷・製本 日経印刷株式会社

発 行 所 株式会社オライリー・ジャパン

〒160-0002 東京都新宿区坂町26番地27 インテリジェントプラザビル1F

Tel (03)3356-5227

Fax (03)3356-5263

電子メール japan@oreilly.co.jp

発 売 元 株式会社オーム社

〒101-8460 東京都千代田区神田錦町3-1

Tel (03)3233-0641 (代表)

Fax (03)3233-3440

Printed in Japan (ISBN978-4-87311-444-6)

乱丁本、落丁本はお取り替え致します。

本書は著作権上の保護を受けています。本書の一部あるいは全部について、株式会社オライリー・ジャパンから文書による許諾を得ずに、いかなる方法においても無断で複写、複製することは禁じられています。