

# モバイル プログラミング2

本日の予定

# 午前

- Apache Overview
  - ライフサイクルとモジュール
- サーバサイドプログラミング
  - ファイル処理
  - 多種多様な関数

# 午後

- ネットワーク基礎
  - HTTPヘッダ
  - Cookie
- HTTP演習
  - ファイル処理

前回休んだ人

(°▽°)ｼ

PC借りた人

(°▽°)ｼ

# まずは追いつこう

1. GitHubの資料見てね

<http://github.com/katsube/neec>

2. 環境構築

3. 環境構築で困ったらすぐに聞いてください

# アンケート (出席カード)

氏名  
学号  
所属  
学年  
性別  
年齢  
職業  
学業  
研究  
生活  
その他



# アンケート

1. 提出 = 出席 (授業終了までに限る)  
**未提出 = 欠席**
2. 学籍番号、名前が確認できない場合は**欠席**
3. わからない場合は、どこが理解できなかったか記入

## ・返却を希望する場合

法 ※大きく2種類

チェックしてください

次回～次々回の授業で返却します

# アンケート

- 今回から難易度を追加

モバイルプログラミング 2 出席カード

2016/11/07

学籍番号 [                      ]

氏名 [                      ]

難易度 ( 優しい ・ 最適 ・ 難しい )

☐ 返却を希望する

○をつけてください。

様子を見て難易度を調整します。

# アンケート

1. 「白紙提出」「授業を聞いていたと判断できない」場合は個別にヒアリングを行います。

- よほどのことがなければ呼び出されません
- 大人としての自覚を持って授業に望んで下さい

# 前回のアンケートに 答えるコーナー

1. アンケートの設問の意図が不明瞭
2. 後ろだと画面が見えない  
(下部の文字が見えない)
3. 授業スピードが早い箇所があった
4. 月給いくらですか？

# この授業では持ち込み可




家電&カメラ ▾ オペラグラス

Amazonポイント: 235

マイストア ギフト券 タイムセール Amazonで売る ヘルプ

カメラ 新製品 ▾ デジタル一眼レフ ▾ ミラーレス一眼 ▾ コンパクト ▾ ビデオカメラ ▾ レンズ ▾ 三脚 ▾ アクセサリー ▾ 光学機器 ▾ 業務用 ▾ 中古 ▾ バーゲン Amazonランキング

家電&カメラ ▾ カメラ ▾ 双眼鏡・望遠鏡・光学機器 ▾ 双眼鏡



画像にマウスを合わせると拡大されます

## Golden Ray 双眼鏡 コンパクト 8倍21口径 ダハプリズム式 オペラグラス コンサート/ライブ/登山/スポーツ観戦用

Golden Ray  
★★★★☆ ▾ 34件のカスタマーレビュー

参考価格: ¥1,780  
価格: ¥ 999 対象商品¥2,000以上の注文で通常配送  
OFF: ¥ 781 (44%)

この商品の特別キャンペーン 条件を満たした場合は、1倍

在庫あり。在庫状況について

11/4 金曜日にお届けするには、今から**13 時間** 確定してください (有料オプション。Amazonプライム会員は無料です。)

この商品は、Setomが販売し、Amazon.co.jp が出品しています。Amazon.co.jp の出品は、Amazon.co.jp の出品ポリシーに従って販売・配送されます。

新品の出品: 1 ¥ 999より

- 光透過率に優れた高品質レッド膜コーティング。コンサート会場、スポーツ会場など、遠くまで見やすい。
- 最先端の光学技術を用い、極めて明るくシャープな映像を再現します。
- 8倍率 × 21対物レンズ仕様で広く遠い目標でも明確に見えます。
- ダイヤを回転してレンズの鮮明度を調整することができます。
- スタイリッシュなコンパクトボディで持ち運びに便利です。コンサート、スポーツ観戦、旅行、アウトドアなど幅広いシーンで、「観る楽しみ」を再発見してください。

▶ もっと見る





# この授業では持ち込み可 (他の人の迷惑にならない範囲であれば可)

iPhone・スマホ用  
**50倍望遠レンズ**



汎用ホルダー付



Ahomaro

月給いくら  
ですか？

にマジレスするコーナー



# 「月給いくらですか」に マジレスするコーナー

- ポイント

- そもそも個人事業主(フリーランス)には、  
「給料」「給与」という概念はない



一般的に『給料』『給与』とは、会社などに雇用された人へ労働の対価として支払われる物を指します。公務員は『俸給』といいます。

雇われていない人は、誰からも給与がもらえないのです。

「社長」などの役員は  
『役員報酬』

# 「月給いくらですか」に マジレスするコーナー

- ポイント

ーイメージとしては「商店街の八百屋」  
さんのような物。



※法人化していない場合

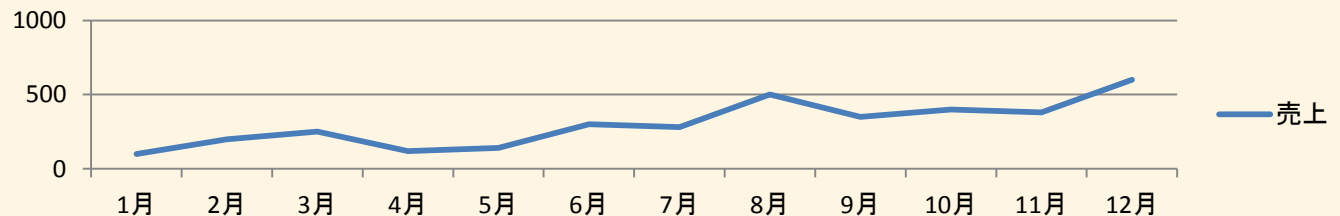
# 「月給いくらですか」に マジレスするコーナー

- ポイント

- サラリーマンは基本給が決まっている。
- 個人事業主は毎月変動する（場合がある）



売上



# フリーランス

- 会社には縛られない分、自由に活動でき、働けば働くほどお金を得ることができます。
- しかし、将来は何も約束されていません。失業保険はなく、労働基準法も適用されません。病気になれば収入が0になるリスクを常に抱えています。

# 働き方

- 世の中には様々な職種、契約形態があります。ゲーム業界の中でもたくさんあります。
- もし仮に就職した先でつまずいたとしても、別の道を探せばいいだけなのです。
- 悲観的にならず、むしろチャンスだと考えられればしめたものです。

# 書籍「仕事の裏切り」

amazon.co.jp

本

amazonstud

Amazonポイント: 235

マイストア ギフト券 タイムセール

JP

勝部麻季人さん  
アカウントサービス

カテゴリー

本 詳細検索 ジャンル一覧 新刊・予約 Amazonランキング コミック・ラノベ 雑誌 文庫・新書 Amazon Stud

本 > ビジネス・経済 > 実践経営・リーダーシップ

i

お客様は、2004/12/8にこの商品を注文しました。  
[この注文を表示](#)



仕事の裏切り なぜ、私たちは働くのか 単行本 -

2003/11/22

ジョアン・キウーラ (著), 中嶋 愛 (翻訳), 金井 寿宏 (翻訳)

★★★★☆ 3件のカスタマーレビュー

▶ その他 ( ) の形式およびエディションを表示する

単行本

¥ 1 より

¥ 1 より 16 中古品の出品

¥ 793 より 2 コレクター商品の出品

この画像を表示

価格履歴 商品のトラッキング 設定 → 関連する商品を探す → トリ

¥ 200

基本情報	目次	ダウンロード	正誤表
第1部	仕事の意味		
第1章	なぜ、私たちは働くのか		
第2章	仕事とは何か		
第3章	呪いから天職へ		
第4章	ロマンチックな幻想		
第2部	他人のための仕事		
第5章	仕事と自由		
第6章	労働者を飼い馴らす		
第7章	いかにして、仕事はかくも複雑になったのか		
第8章	希望の職場		
第3部	仕事と人生		
第9章	裏切られたホワイトカラー		
第10章	時間		
第11章	余暇と消費		
第12章	「さらなる何か」を求めて		

今後  
も  
マジレスします  
質問はお気軽に

# Apache Overview





# 様々なWebサーバ



Apache



IIS

(Internet Information Services)

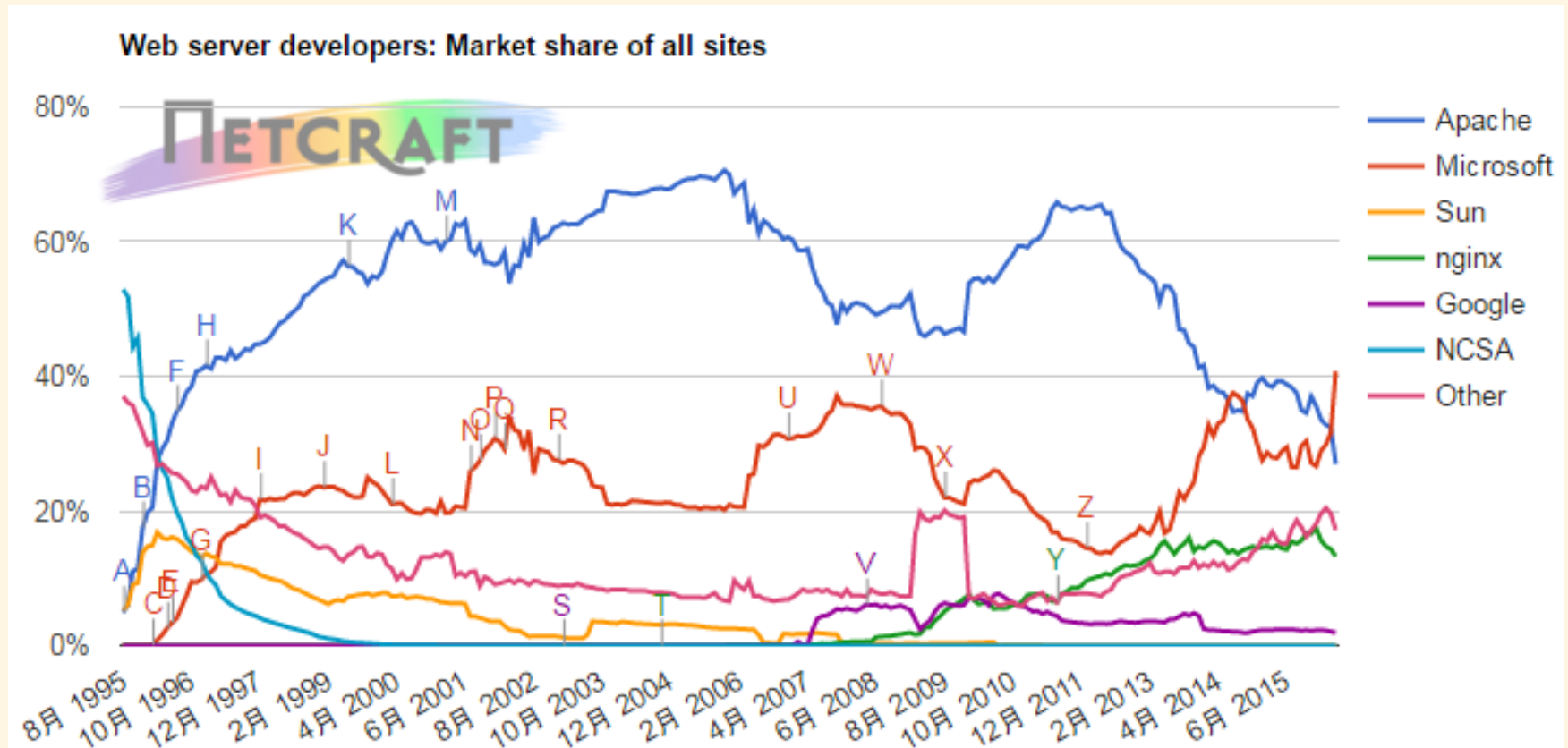


NGINX

NCSA  
*HTTPd*

自作することも可能

# Webサーバの世界シェア



Developer	March 2016	Percent	April 2016	Percent	Change
Microsoft	317,761,318	31.65%	441,470,894	40.75%	9.10
Apache	325,285,185	32.40%	292,043,548	26.96%	-5.44
nginx	143,464,293	14.29%	143,349,439	13.23%	-1.06
Google	20,790,767	2.07%	20,597,605	1.90%	-0.17

<https://news.netcraft.com/archives/2016/04/21/april-2016-web-server-survey.html>

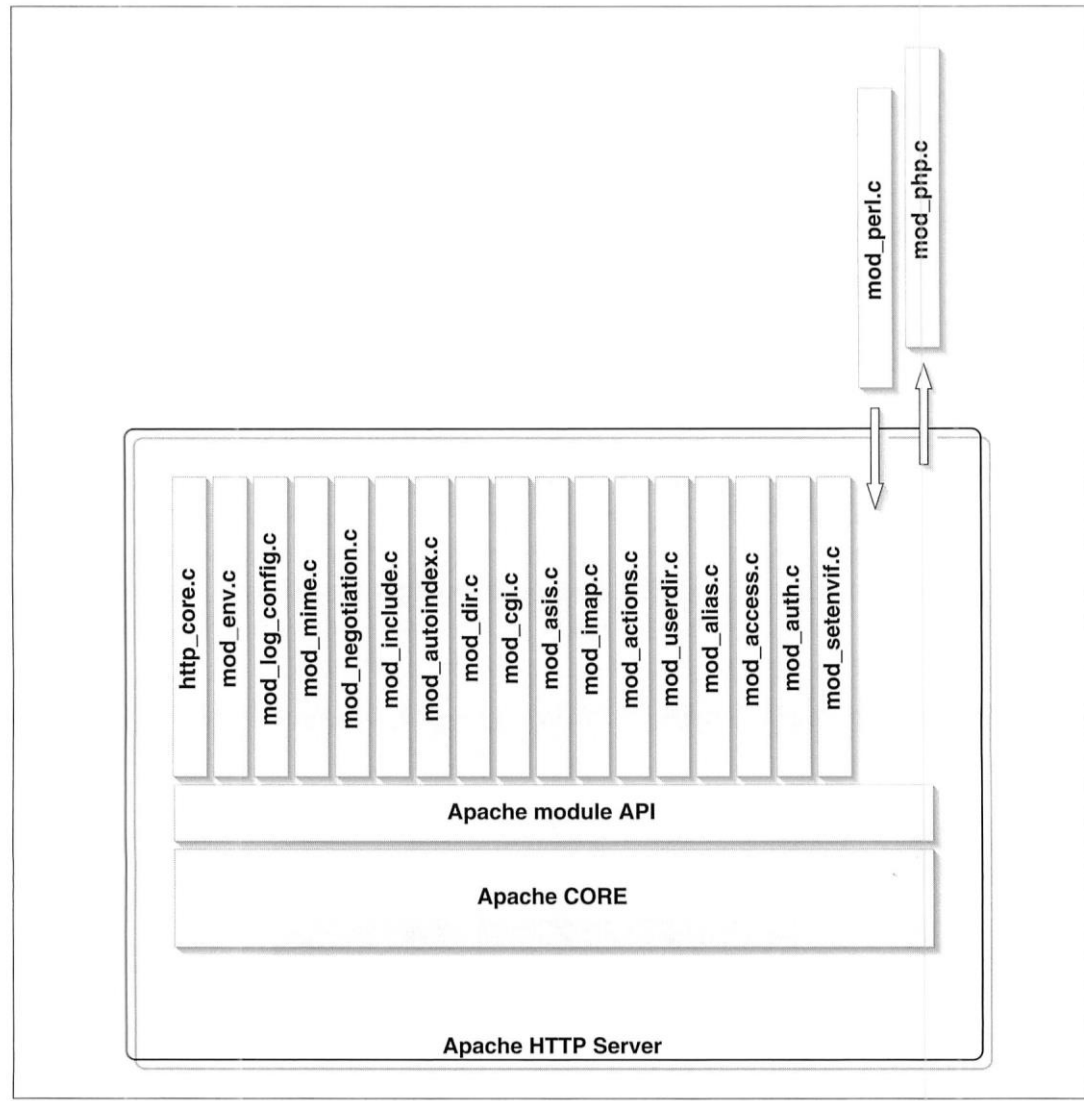
# 基本的な構造

- モジュール

- Apacheは小さなモジュール(機能)の集合体。必要なモジュールを組み合わせて稼働させる。

# Apacheのモジュール構造

●図 1-1 : Apacheのモジュールアーキテクチャ



# Apacheのモジュール構造

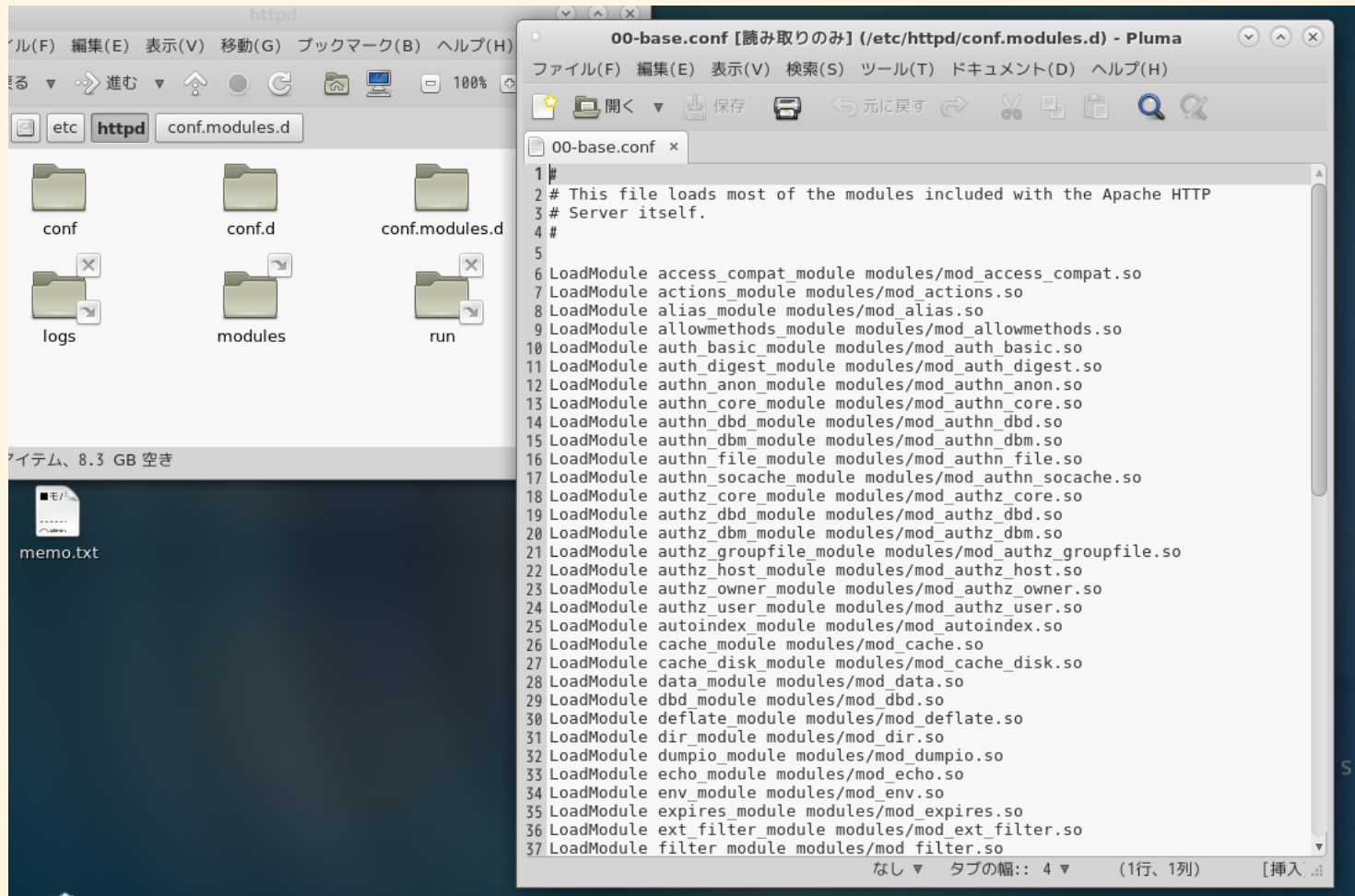
/etc/httpd/modules

```
Mate Terminal
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)
[neec@localhost ~]$ cd /etc/httpd/modules/
[neec@localhost modules]$ ls
libphp56-php5.so      mod_deflate.so        mod_proxy_connect.so
mod_access_compat.so  mod_dialup.so         mod_proxy_express.so
mod_actions.so        mod_dir.so            mod_proxy_fcgi.so
mod_alias.so          mod_dumpio.so         mod_proxy_fdpass.so
mod_allowmethods.so   mod_echo.so           mod_proxy_ftp.so
mod_asis.so           mod_env.so            mod_proxy_http.so
mod_auth_basic.so     mod_expires.so        mod_proxy_scgi.so
mod_auth_digest.so    mod_ext_filter.so     mod_proxy_wstunnel.so
mod_authn_anon.so     mod_file_cache.so     mod_ratelimit.so
mod_authn_core.so     mod_filter.so         mod_reflector.so
mod_authn_dbd.so      mod_headers.so        mod_remoteip.so
mod_authn_dbm.so      mod_heartbeat.so      mod_reqtimeout.so
mod_authn_file.so     mod_heartmonitor.so   mod_request.so
mod_authn_socache.so  mod_include.so       mod_rewrite.so
mod_authz_core.so     mod_info.so           mod_sed.so
mod_authz_dbd.so      mod_lbmethod_bybusyness.so mod_setenvif.so
mod_authz_dbm.so      mod_lbmethod_byrequests.so mod_slotmem_plain.so
mod_authz_groupfile.so mod_lbmethod_bytraffic.so mod_slotmem_shm.so
mod_authz_host.so     mod_lbmethod_heartbeat.so mod_socache_dbm.so
mod_authz_owner.so    mod_log_config.so     mod_socache_memcache.so
mod_authz_user.so     mod_log_debug.so      mod_socache_shmcb.so
mod_autoindex.so      mod_log_forensic.so   mod_speling.so
mod_buffer.so         mod_logio.so          mod_status.so
mod_cache.so          mod_lua.so            mod_substitute.so
mod_cache_disk.so     mod_macro.so          mod_suexec.so
mod_cache_socache.so  mod_mime.so           mod_systemd.so
mod_cgi.so            mod_mime_magic.so     mod_unique_id.so
mod_cgid.so           mod_mpm_event.so      mod_unixd.so
mod_charset_lite.so   mod_mpm_prefork.so    mod_userdir.so
mod_data.so           mod_mpm_worker.so     mod_usertrack.so
mod_dav.so            mod_negotiation.so    mod_version.so
mod_dav_fs.so         mod_proxy.so          mod_vhost_alias.so
mod_dav_lock.so       mod_proxy_ajp.so      mod_watchdog.so
mod_dbd.so            mod_proxy_balancer.so
[neec@localhost modules]$ pwd
/etc/httpd/modules
[neec@localhost modules]$
```

# Apacheのモジュール構造

/etc/httpd/conf/httpd.conf

/etc/httpd/conf/conf.modules.d/00-base.conf



# リクエスト処理

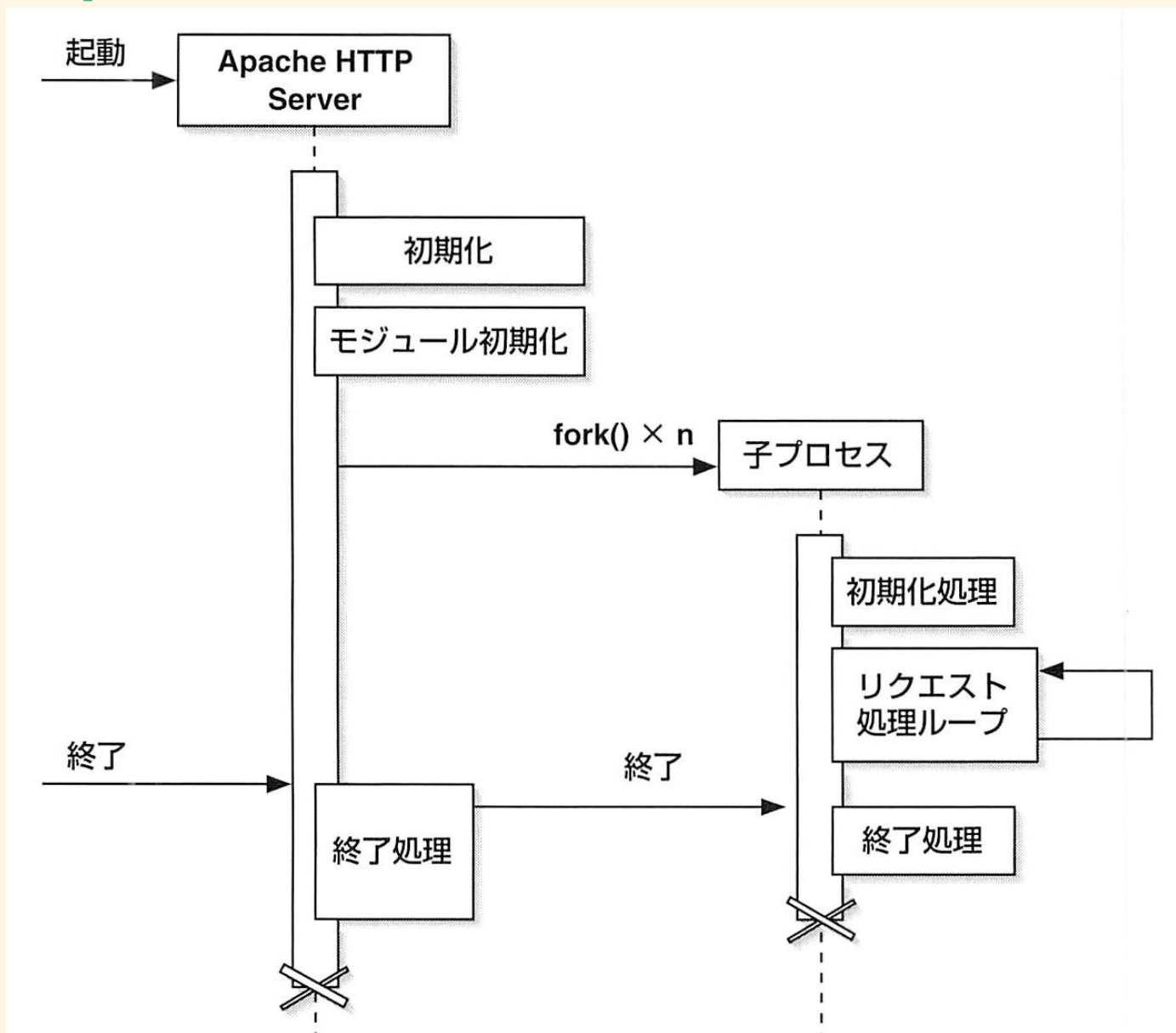
- **prefork**

- 起動時に子プロセスを複数作成しておく。
- 1つのプロセスで1つのリクエスト

- **worker**

- 子プロセスがリクエストをスレッドで処理する。
- 1つのプロセスで複数のリクエスト

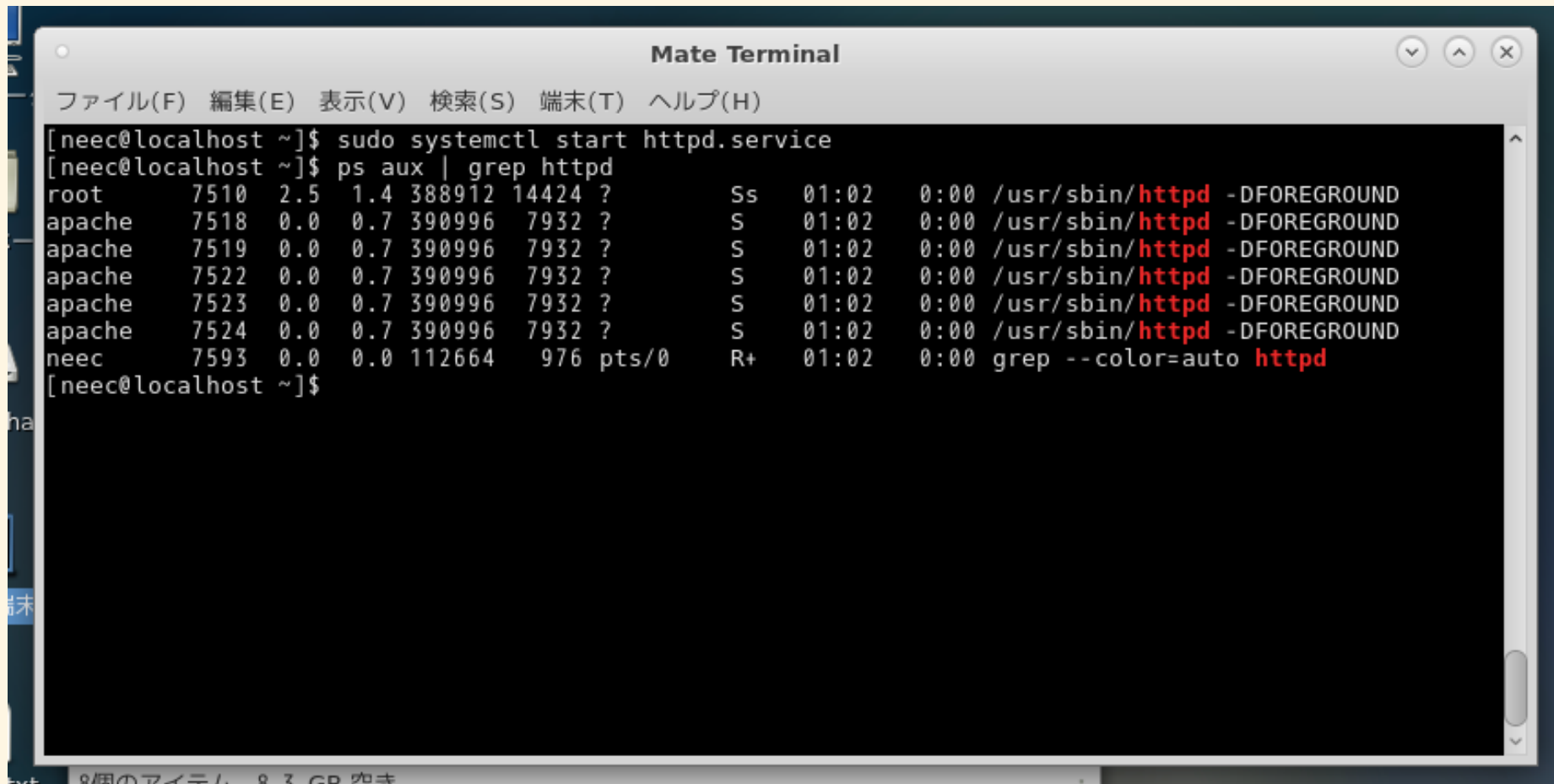
# Apacheのライフサイクル





# Apacheのライフサイクル

\$ ps aux | grep http



The screenshot shows a terminal window titled "Mate Terminal" with a menu bar containing "ファイル(F)", "編集(E)", "表示(V)", "検索(S)", "端末(T)", and "ヘルプ(H)". The terminal content shows the execution of the command `sudo systemctl start httpd.service` followed by `ps aux | grep httpd`. The output lists several processes: one running as root and five as apache, all using `/usr/sbin/httpd` with the `-DFOREGROUND` flag. A final line shows the `grep` process itself.

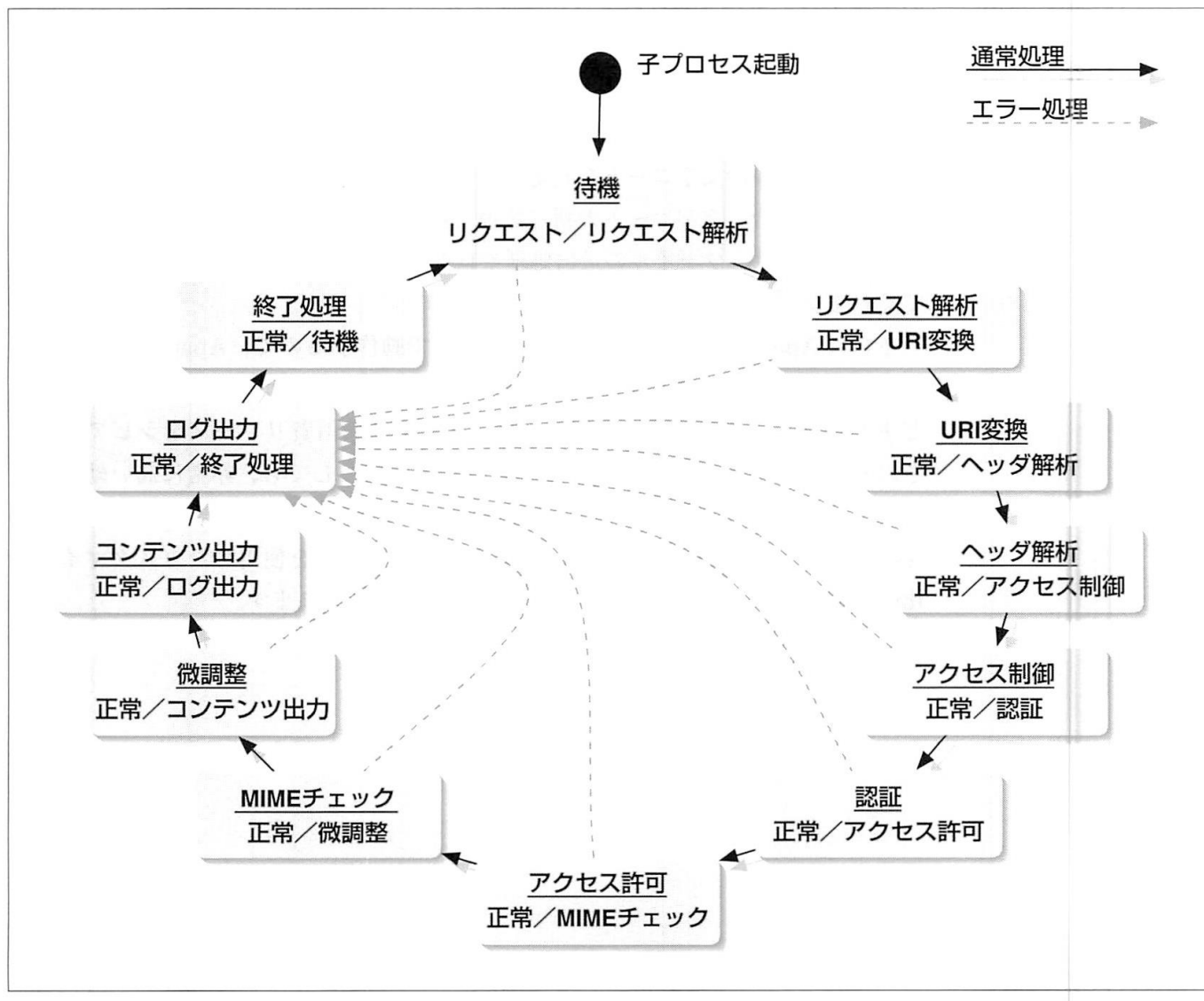
```
[neec@localhost ~]$ sudo systemctl start httpd.service
[neec@localhost ~]$ ps aux | grep httpd
root      7510   2.5   1.4 388912 14424 ?        Ss   01:02   0:00 /usr/sbin/httpd -DFOREGROUND
apache    7518   0.0   0.7 390996  7932 ?        S    01:02   0:00 /usr/sbin/httpd -DFOREGROUND
apache    7519   0.0   0.7 390996  7932 ?        S    01:02   0:00 /usr/sbin/httpd -DFOREGROUND
apache    7522   0.0   0.7 390996  7932 ?        S    01:02   0:00 /usr/sbin/httpd -DFOREGROUND
apache    7523   0.0   0.7 390996  7932 ?        S    01:02   0:00 /usr/sbin/httpd -DFOREGROUND
apache    7524   0.0   0.7 390996  7932 ?        S    01:02   0:00 /usr/sbin/httpd -DFOREGROUND
neec      7593   0.0   0.0 112664   976 pts/0    R+   01:02   0:00 grep --color=auto httpd
[neec@localhost ~]$
```

# 一連の流れ

- Apacheは非常にたくさんの処理を各リクエスト毎に行っています。
- 昨今のWebサーバには求められることが非常に多く、単純にファイルを返却するだけではないのです。

# Apache リクエスト処理ループ

●図 1-3 : リクエスト処理ループ



# 静的と動的

- 静的

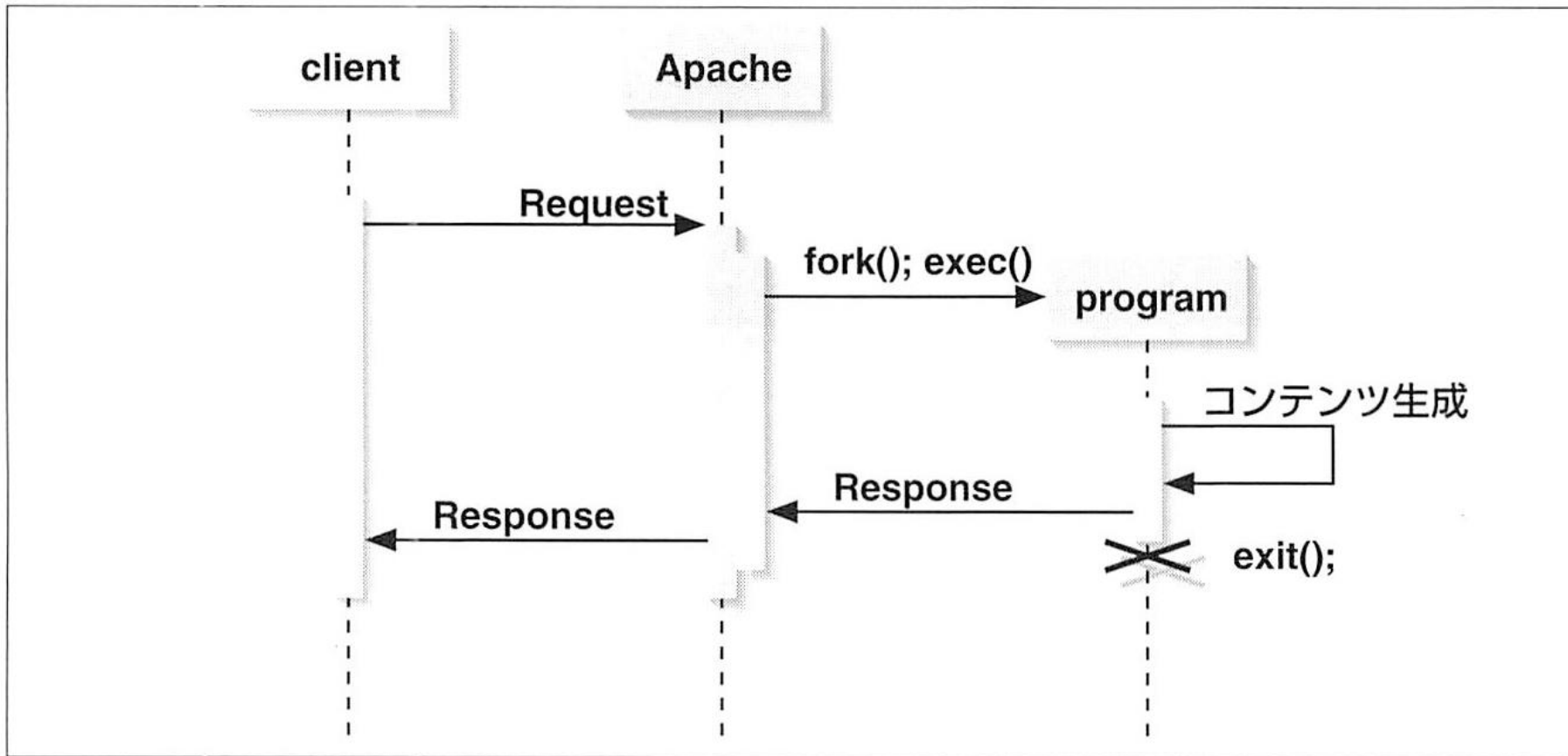
- xxx.html、xxx.pngのようにいつ誰がリクエストしても毎回特定のファイル内容を返却する。

- 動的

- プログラムを作動させ、その処理結果を返却する。

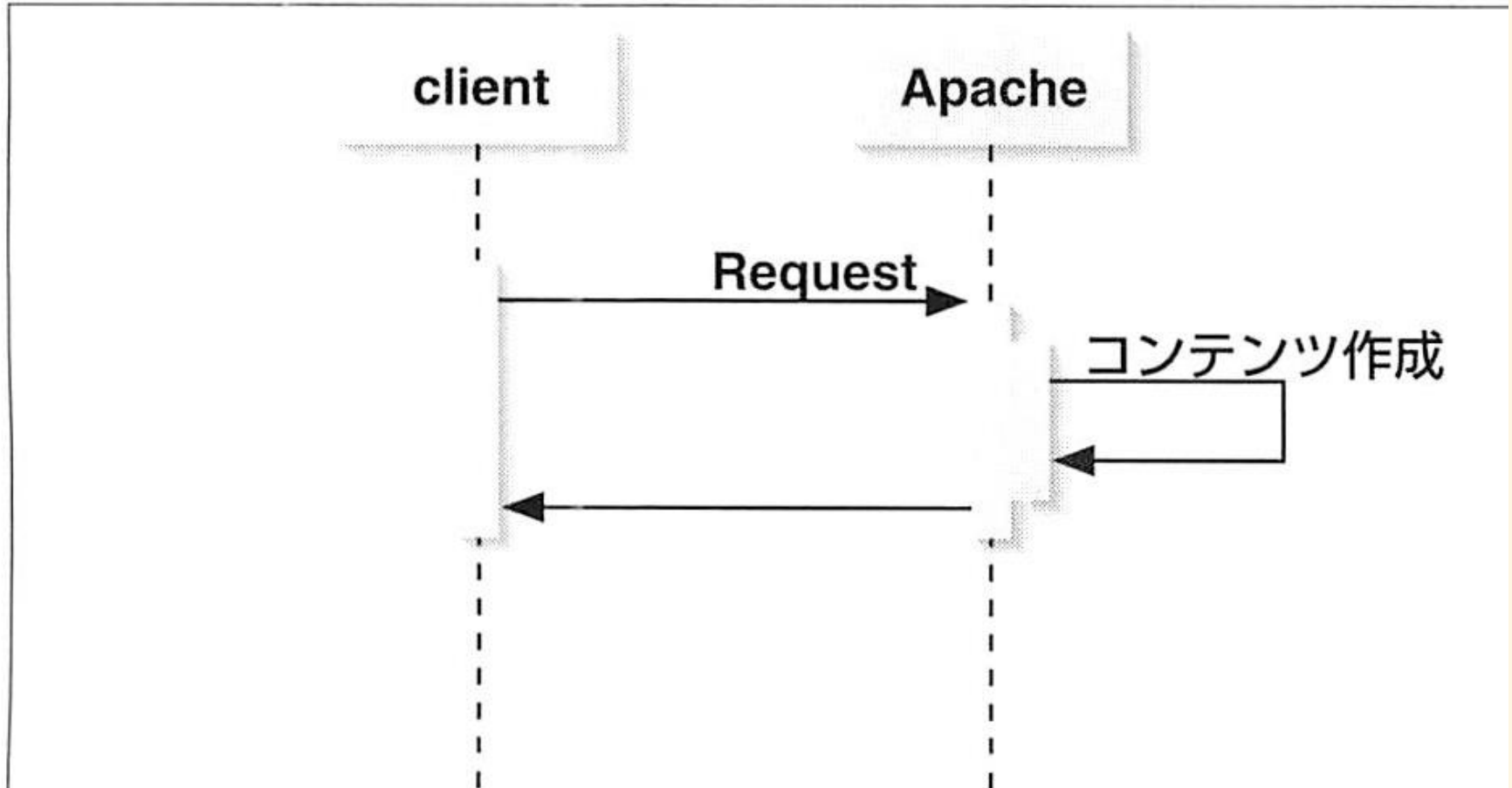
# 「CGI」と「Apacheモジュール」

●図 1-4 : CGIの処理シーケンス



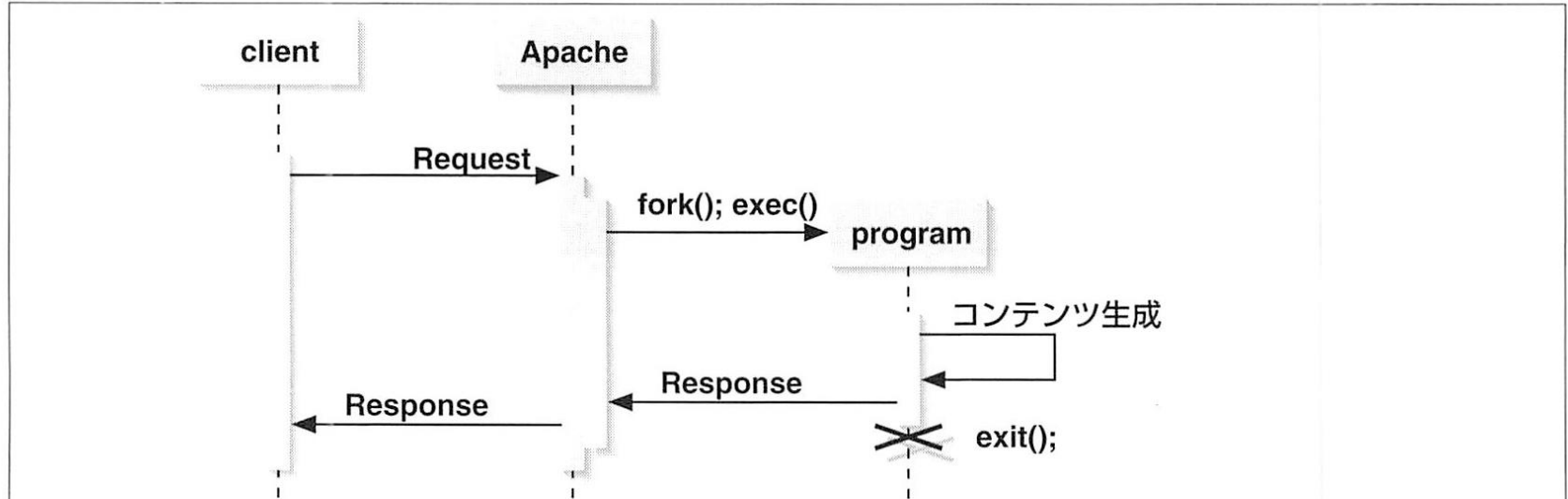
# 「CGI」と「Apacheモジュール」

●図 1-6 : Apache モジュールの処理シーケンス

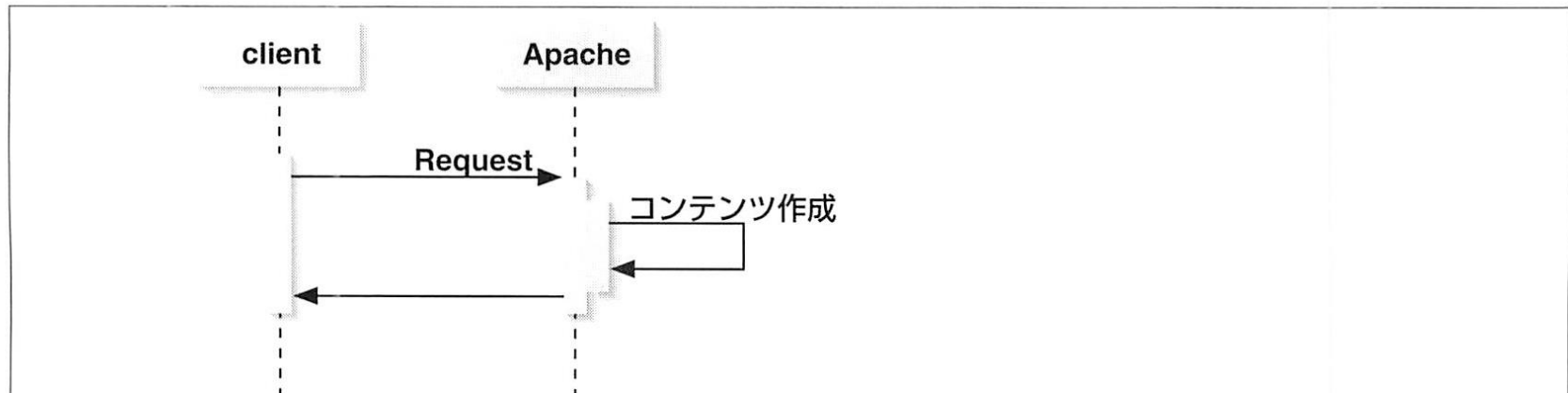


# 「CGI」と「Apacheモジュール」

●図 1-4 : CGIの処理シーケンス



●図 1-6 : Apache モジュールの処理シーケンス



# 書籍「Apacheモジュールプログラミング」



すべて ▾ apache モジュールプログラミング 🔍

Amazonポイント: 235

JP 勝部麻季人さん アカウントサ

カテゴリー ▾

マイストア ギフト券 タイムセール Amazonで売る ヘルプ

本 詳細検索 ジャンル一覧 新刊・予約 Amazonランキング コミック・ラノベ 雑誌 文庫・新書 Amazon Student バーゲン 古本

本 > コンピュータ・IT > ネットワーク



お客様は、2006/12/14にこの商品を注文しました。  
[この注文を表示](#)

なか見!検索↓



  
[この画像を表示](#)

### Apacheモジュール プログラミングガイド (Advanced Server-side programmingシリーズ) 単行本 - 2003/8/1

小山 浩之 ▾ (著)

★★★★☆ ▾ 3件のカスタマーレビュー

▶ その他 ( ) の形式およびエディションを表示する

単行本

¥ 1,530 より

¥ 1,530 より 14 中古品の出品

 不正確な製品情報を報告。

※情報が古い箇所もありますが、C言語が読めれば理解しやすいです。



# サーバサイド プログラミング



# 前回の復習



# 演習サマリー1

- Webサーバ上でPHPを起動する
  - Helloworld
    - CLIでprint ver
    - HTMLの中に埋め込むVer
  - HTML中にPHPを埋め込んでみる
    - 時間を表示する
      - date関数
    - 困ったら公式ドキュメントを参照 <http://php.net/>
- コマンドラインからPHPの起動
  - php -v

# 演習サマリー2

- PHPの文法

- 基本構文

- 原則として半角英数字を推奨。スペースやタブ、改行の扱いはCなどと同じ。変数名や関数名で**大文字小文字は区別されない**。最後にセミコロン、左辺と右辺の関係は他の言語と同じ。演算子なども概ね同じ。
    - コメントは `//` `/* */` `#`が使える

- 変数

- `$`をつける。宣言は不要。型は自動的に決まる。また自動的にauto変数（ローカル変数）になる。

# 演習サマリー3

- PHPの文法
  - 配列、ハッシュ(連想配列)
    - 配列は他言語とほぼ同じ使い方ができる。
    - 初期化は `array, []`
    - ハッシュについて詳しく解説
  - 制御構造
    - `if / switch / while / for` などが利用できる

# 演習サマリー4

- PHPの文法（つづき）
  - 関数
    - function で作成する
      - 先ほどの演習問題を関数化する
    - 引数、戻り値も他言語と同様
      - デフォルト値の定義
      - 型宣言も可能だが、7未満は機能制限が厳しいのであまり利用する機会がない。
      - 参照渡しをしたい場合は、関数側に & をつける

# 演習サマリー5

- PHPの文法（つづき）
  - クラス
    - classで作成できる
      - メソッド `public|private function foo(){}`
        - » static でインスタンスを作成しなくても呼び出せる
      - プロパティ `public|private $foo = 'bar';`
      - クラス定数 `const AAA = 'foo';`
      - コンストラクタ `__construct`
      - デストラクタ `__destruct`
    - `$a = new MyClass();` でインスタンス作成
    - extends で継承可能
      - メソッドのオーバーライド(上書き)も可能
        - » public, protected

# 演習1. MAP表示

配列

```
$map= [  
    [0, 0, 0, 0]  
    , [0, 1, 2, 0]  
    , [0, 2, 1, 0]  
    , [0, 0, 0, 0]  
];
```



実行結果

```
$ php map.php  
_ _ _ _  
_ A B _  
_ B A _  
_ _ _ _
```

※2Dゲームでよくあるマップデータの表示  
※PHPの文法を覚えるのが目的



# 演習2. オセロ盤クラス

```
<?php
$board = new OthelloBoard(8, 8);

//座標 1,3 に白のコマを配置
$ret = $board->set(1, 3, OthelloBoard::WHITE);
if($ret === false){
    print "そこにコマは置けません";
}
else{
    //盤面を表示
    $board->printboard();
}

class OthelloBoard{
    //ここを作成する
}
```

# 書籍「UNIXという考え方」



すべて

誕生


Amazonポイント: 85

マイストア ギフト券 タイムセール

JP 勝部麻季人さん アカウントサービス

本 詳細検索 ジャンル一覧 新刊・予約 Amazonランキング コミック・ラノベ 雑誌 文庫・新書 Amazon Student

本 > コンピュータ・IT > コンピュータサイエンス

 お客様は、2002/5/19にこの商品を注文しました。  
[この注文を表示](#)

なか見!検索↓



## UNIXという考え方—その設計思想と哲学

単行本  
— 2001/2

Mike Gancarz (著), 芳尾 桂 (翻訳)

★★★★☆ 23件のカスタマーレビュー

▶ その他 ( ) の形式およびエディションを表示する

単行本

¥ 1,728

¥ 1,299 より 14 中古品の出品  
¥ 1,728 より 3 新品

10/10 月曜日 にお届けするには、今から**13 時間 41 分**以内に「**お急ぎ便**」または「**当日お急ぎ便**」を選択して注文を確定してください（有料オプション。Amazonプライム会員は無料）

## 1.1 UNIXの考え方：簡単なまとめ

UNIXの考え方におけるそれぞれの定理は、一見簡単なものに見える。また、実際に簡単なので、その重要さを見過ごされることも多い。しかし、だまされてはいけない。一見簡単に見えるこれらのアイデアが一貫して行われると、信じられないほど効果的なものとなる。

ここで、UNIXの考え方の輪郭を明確にしてもらうため、それぞれの定理について簡単にまとめておく。本書の残りは、これらの解説に費やされる。

### 1. スモール・イズ・ビューティフル（小さいものは美しい）

小さいものは、大きいものにはない利点がいくつもある。小さいもの同士なら、簡単に独特の便利な方法で組み合わせることができるというのもその一つだ。

### 2. 一つのプログラムには一つのことをうまくやらせる

一つのことに集中することで、プログラムに不要な部分をなくせる。不要部分があると、実行速度が遅くなり、不必要に複雑になり、融通が効かなくなる。

### 3. できるだけ早く試作する

あらゆるプロジェクトにおいて、試作は重要だ。一般的に試作は設計全体のうちのほんの一部として扱われているが、UNIXにおける試作は、効率的な設計には欠かせない重要な一部だ。

# ガチャを作る



# 演習1. ガチャ

- コマンドライン上で動作する「ガチャ」を作成してください。
- 仕様はGithubを参照
- 基本的にOOP（Gachaクラスを作成）

## 実行結果

```
$ php gacha.php 3
```

```
1回目 SR キャラ名
```

```
2回目 N キャラ名
```

```
3回目 R キャラ名
```

# 乱数

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル > 関数リファレンス > 数学 > Math > Math 関数

« mt\_getrandmax

mt\_srand »

## mt\_rand

Change language: Japanese ▼

[Edit](#) [Report a Bug](#)

(PHP 4, PHP 5, PHP 7)

mt\_rand — よりよい乱数値を生成する

### 説明

```
int mt_rand ( void )
```

```
int mt_rand ( int $min , int $max )
```

古い libc の多くの乱数発生器は、怪しげであるか特性が不明であったりし、また低速でした。デフォルトでは、PHP は [rand\(\)](#) において libc の乱数発生器を使用します。mt\_rand() 関数は、その代替品となるものです。この関数は、その特性が既知の乱数生成器 » [Mersenne Twister](#) を使用し、平均的な libc の rand() よりも 4 倍以上高速に乱数を生成します。

オプションの引数 **min,max** を付けずにコールした場合、mt\_rand() は 0 から [mt\\_getrandmax\(\)](#) の間

### Math 関数

[abs](#)  
[acos](#)  
[acosh](#)  
[asin](#)  
[asinh](#)  
[atan2](#)  
[atan](#)  
[atanh](#)  
[base\\_convert](#)  
[bindec](#)  
[ceil](#)  
[cos](#)  
[cosh](#)  
[decbin](#)  
[dechex](#)  
[decoct](#)  
[deg2rad](#)  
[exp](#)  
[expm1](#)  
[floor](#)

## 演習2. ガチャ(外部ファイル)

- 演習1で作成したガチャのデータを別のファイルに分けてください。

require

```
<?php  
require('gacha.php');
```

実行結果

```
$ php gacha.php 3  
1回目 SR キャラ名  
2回目 N キャラ名  
3回目 R キャラ名
```

# 外部のPHPファイルを読み込む

[php](#)[Downloads](#)[Documentation](#)[Get Involved](#)[Help](#)

PHP マニュアル > 言語リファレンス > 制御構造

« returninclude »

Change language: Japanese Edit Report a Bug

## require

(PHP 4, PHP 5, PHP 7)

`require` は [include](#) とほぼ同じですが、失敗した場合に **E\_COMPILE\_ERROR** レベルの致命的なエラーも発生するという点が異なります。つまり、スクリプトの処理がそこで止まってしまうということです。一方 [include](#) の場合は、警告 (**E\_WARNING**) を発するもののスクリプトの処理は続行します。

どのように動作するかについては [include](#) のドキュメントを参照ください。

**User Contributed Notes** 26 notes [+ add a note](#)

▲ 82 ▼ chris at chrisstockton dot org9 years ago

Remember, when using require that it is a statement, not a function. It's not necessary to write:  

```
<?php  
require('somefile.php');
```

### 制御構造

- 導入
- if
- else
- elseif/else if
- 制御構造に関する別の構文
- while
- do-while
- for
- foreach
- break
- continue
- switch
- declare
- return
- » **require**
- include
- require\_once
- include\_once
- goto



## 演習3. ガチャ(CSV)

- 演習1で作成したガチャのデータをCSVから読み込むようにしてください。
- CSVはGitHubから取得

fopen, fgets, fclose

```
<?php
$fp = fopen('data.csv', 'r');
while ( ($data = fgets($fp) ) !== false ) {
    ;
}
fclose($fp);
```

# ファイル処理 fopen

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル > 関数リファレンス > ファイルシステム > ファイルシステム > ファイルシステム 関数

« fnmatch

fpasssthru »

## fopen

Change language: Japanese ▼

[Edit](#) [Report a Bug](#)

(PHP 4, PHP 5, PHP 7)

fopen — ファイルまたは URL をオープンする

### 説明

```
resource fopen ( string $filename , string $mode [, bool $use_include_path = false [, resource $context ]] )
```

fopen() は、**filename** で指定されたリソースをストリームに結び付けます。

### パラメータ

#### filename

**filename** が "スキーム://..." の形式である場合、それは URL とみなされ、PHP はそのプロト

### ファイルシステム 関数

[basename](#)

[chgrp](#)

[chmod](#)

[chown](#)

[clearstatcache](#)

[copy](#)

[delete](#)

[dirname](#)

[disk\\_free\\_space](#)

[disk\\_total\\_space](#)

[diskfreespace](#)

[fclose](#)

[feof](#)

[fflush](#)

[fgetc](#)

[fgetcsv](#)

[fgets](#)

[fgetss](#)

[file\\_exists](#)

[file\\_get\\_contents](#)

# ファイル処理 fgets

**php** Downloads Documentation Get Involved Help

Search

PHP マニュアル > 関数リファレンス > ファイルシステム > ファイルシステム > ファイルシステム 関数

« fgetcsv fgetss »

## fgets

(PHP 4, PHP 5, PHP 7)

fgets — ファイルポインタから 1 行取得する

説明

```
string fgets ( resource $handle [, int $length ] )
```

ファイルポインタから 1 行取得します。

パラメータ

**handle**

ファイルポインタは、有効なファイルポインタである必要があり、[fopen\(\)](#) または [fsockopen\(\)](#) で正常にオープンされた（そしてまだ [fclose\(\)](#) でクローズされていない）ファイルを指している必要

### ファイルシステム 関数

- basename
- chgrp
- chmod
- chown
- clearstatcache
- copy
- delete
- dirname
- disk\_free\_space
- disk\_total\_space
- diskfreespace
- fclose
- feof
- fflush
- fgetc
- fgetcsv
- » **fgets**
- fgetss
- file\_exists
- file\_get\_contents

# 文字列分割 explode

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル > 関数リファレンス > テキスト処理 > 文字列 > String 関数

« echo

fprintf »

## explode

Change language: Japanese ▼

Edit

Report a Bug

(PHP 4, PHP 5, PHP 7)

explode — 文字列を文字列により分割する

### 説明

```
array explode ( string $delimiter , string $string [, int $limit = PHP_INT_MAX ] )
```

文字列の配列を返します。この配列の各要素は、**string** を文字列 **delimiter** で区切った部分文字列となります。

### パラメータ

**delimiter**

### String 関数

[addslashes](#)

[addslashes](#)

[bin2hex](#)

[chop](#)

[chr](#)

[chunk\\_split](#)

[convert\\_cyr\\_string](#)

[convert\\_uudecode](#)

[convert\\_uuencode](#)

[count\\_chars](#)

[crc32](#)

[crypt](#)

[echo](#)

» **explode**

[fprintf](#)

[get\\_html\\_translation\\_table](#)

[hebrew](#)

[hebrevc](#)

[hex2bin](#)

[html\\_entity\\_decode](#)

## 演習4. ガチャ(ログ)

- 演習3で作成したガチャに、ログを記録する機能を追加してください。
- ログの仕様はGithubを参照

fwrite, flock

```
<?php
$fp = fopen('gacha.log', 'a');
fwrite($fp, $string);
fclose($fp);
```

# ファイル処理 fwrite

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル > 関数リファレンス > ファイルシステム > ファイルシステム > ファイルシステム 関数

« truncate

glob »

## fwrite

Change language: Japanese ▼

[Edit](#)

[Report a Bug](#)

(PHP 4, PHP 5, PHP 7)

fwrite — バイナリセーフなファイル書き込み処理

### 説明

```
int fwrite ( resource $handle , string $string [, int $length ] )
```

fwrite()はstringの内容を handleが指しているファイル・ストリームに書き込みます。

### パラメータ

#### handle

[fopen\(\)](#) を使用して作成したファイルシステムポインタリソース。

### ファイルシステム 関数

[basename](#)

[chgrp](#)

[chmod](#)

[chown](#)

[clearstatcache](#)

[copy](#)

[delete](#)

[dirname](#)

[disk\\_free\\_space](#)

[disk\\_total\\_space](#)

[diskfreespace](#)

[fclose](#)

[feof](#)

[fflush](#)

[fgetc](#)

[fgetcsv](#)

[fgets](#)

[fgetss](#)

[file\\_exists](#)

[file\\_get\\_contents](#)

# 日時 date

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル > 関数リファレンス > 日付および時刻関連 > Date/Time > 日付・時刻 関数

« date\_timezone\_set

getdate »

## date

Change language: Japanese ▼

[Edit](#) [Report a Bug](#)

(PHP 4, PHP 5, PHP 7)

date — ローカルの日付/時刻を書式化する

### 説明

```
string date ( string $format [, int $timestamp = time() ] )
```

指定された引数 **timestamp** を、与えられた フォーマット文字列によりフォーマットし、日付文字列を返します。タイムスタンプが与えられない場合は、現在の時刻が使われます。つまり **timestamp** はオプションであり そのデフォルト値は [time\(\)](#) の値です。

### パラメータ

#### format

### 日付・時刻 関数

[checkdate](#)

[date\\_add](#)

[date\\_create\\_from\\_format](#)

[date\\_create\\_immutable\\_from\\_format](#)

[date\\_create\\_immutable](#)

[date\\_create](#)

[date\\_date\\_set](#)

[date\\_default\\_timezone\\_get](#)

[date\\_default\\_timezone\\_set](#)

[date\\_diff](#)

[date\\_format](#)

[date\\_get\\_last\\_errors](#)

[date\\_interval\\_create\\_from](#)

[date\\_string](#)

[date\\_interval\\_format](#)

[date\\_isodate\\_set](#)

[date\\_modify](#)

[date\\_offset\\_get](#)

[date\\_parse\\_from\\_format](#)

# 配列の結合 implode

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル > 関数リファレンス > テキスト処理 > 文字列 > String 関数

« htmlspecialchars

join »

## implode

Change language: Japanese ▼

[Edit](#) [Report a Bug](#)

(PHP 4, PHP 5, PHP 7)

implode — 配列要素を文字列により連結する

### 説明

```
string implode ( string $glue , array $pieces )
```

```
string implode ( array $pieces )
```

配列の要素を **glue** 文字列で連結します。

注意:

**implode()** は、歴史的な理由により、引数をどちらの順番でも受けつけることが可能です。しかし、[explode\(\)](#) との統一性の観点からは、ドキュメントに記述された引数の順番を使用の方が混乱が

### String 関数

[addcslashes](#)

[addslashes](#)

[bin2hex](#)

[chop](#)

[chr](#)

[chunk\\_split](#)

[convert\\_cyr\\_string](#)

[convert\\_uuencode](#)

[convert\\_uuencode](#)

[count\\_chars](#)

[crc32](#)

[crypt](#)

[echo](#)

[explode](#)

[fprintf](#)

[get\\_html\\_translation\\_table](#)

[hebrew](#)

[hebrevc](#)

[hex2bin](#)

[html\\_entity\\_decode](#)



# 排他制御 flock

php

Downloads

Documentation

Get Involved

Help

Search

PHP マニュアル › 関数リファレンス › ファイルシステム › ファイルシステム › ファイルシステム 関数

« filetype

fnmatch »

## flock

Change language: Japanese ▼

Edit

Report a Bug

(PHP 4, PHP 5, PHP 7)

flock — 汎用のファイルロックを行う

### 説明

```
bool flock ( resource $handle , int $operation [, int &$wouldblock ] )
```

**flock()** を使うと、(ほとんどの Unix、そして Windows さえ含む) 事実上すべてのプラットフォームで使用可能な、簡易な読み手/書き手モデルを実現できます。

PHP 5.3.2 より前のバージョンでは、[fclose\(\)](#) でロックの解放も行います (これは、スクリプトが終了した場合にも自動的にコールされます)。

PHP は、恣意的にファイルをロックする汎用の手段を提供します (これは、アクセスする全プログラムが同一のロックの方法を使用する必要があり、そうでない場合は動作しないことを意味します)。デフォルトでは、要求したロックが確保されるまでこの関数はブロックします。以下で説明する **LOCK\_NB** オプショ

### ファイルシステム 関数

basename

chgrp

chmod

chown

clearstatcache

copy

delete

dirname

disk\_free\_space

disk\_total\_space

diskfreespace

fclose

feof

fflush

fgetc

fgetcsv

fgets

fgetss

file\_exists

file\_get\_contents