

モバイル プログラミング2

本日の予定

午前

- サーバサイドプログラミング
 - PHPの起動
 - PHPの基本文法

午後

- ネットワーク基礎
 - HTTPメソッド
- HTTP演習
 - 動的なWebページ

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

前回のFB

前回のアンケートに 答えるコーナー

1. Deamon? Daemon?
2. 前の電気消して
3. パワポ見やすいです！
4. リダイレクトが難しい？
5. C#
6. コピペ厨
7. JSちゃん、ぬるぽ

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

前回の復習

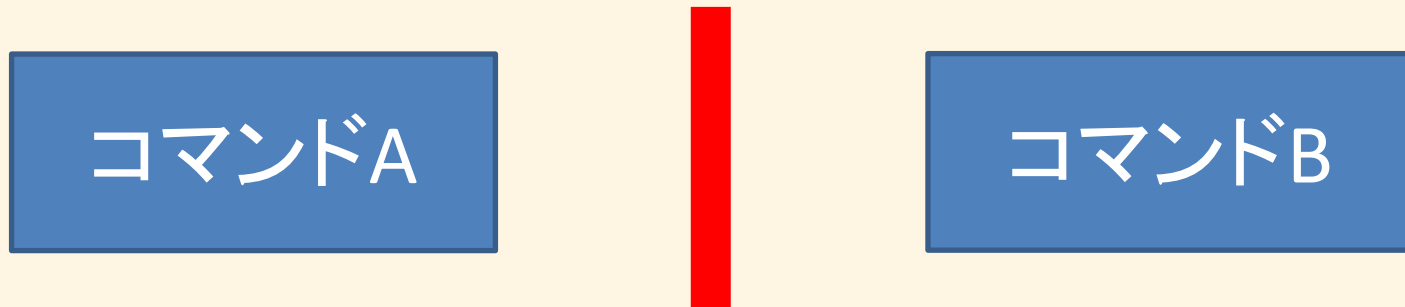
Linuxコンソール



パイプ

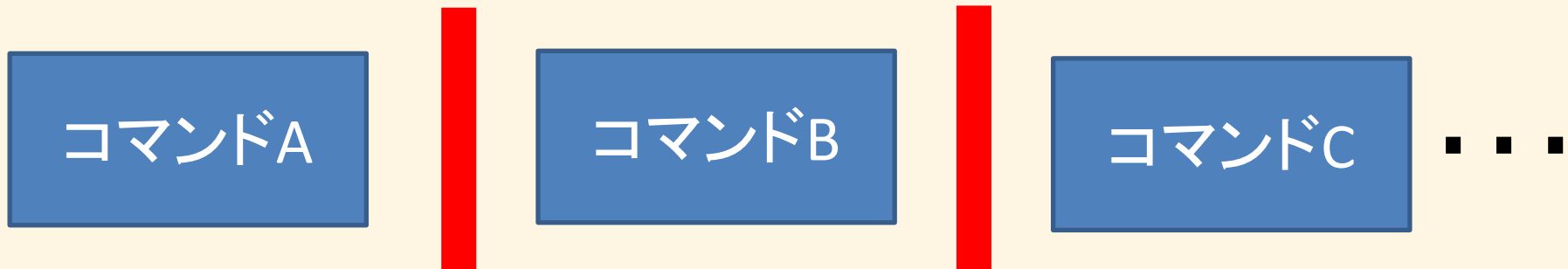
パイプを利用する

- コマンドの実行結果を、次のコマンドに引き継ぐことができます。



パイプを利用

- パイプは何個でもコマンドを連結することができます。



パイプを利用

```
$ grep 'GET /category/games' access.log | wc -l
```

- grepでaccess.log から"/category/games"を検索する
- 検索結果の行数をwcでカウント

パイプ

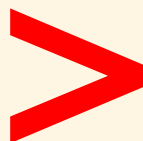
- |
 - コマンドの実行結果(標準出力)を、次のコマンドに渡す
 - パイプは一度に何個でも指定できる
- Linuxの1コマンドが1機能なのは、パイプの利用を前提としている

リダイレクト

リダイレクトを利用

- コマンドの実行結果を、ファイル等に保存することができます。

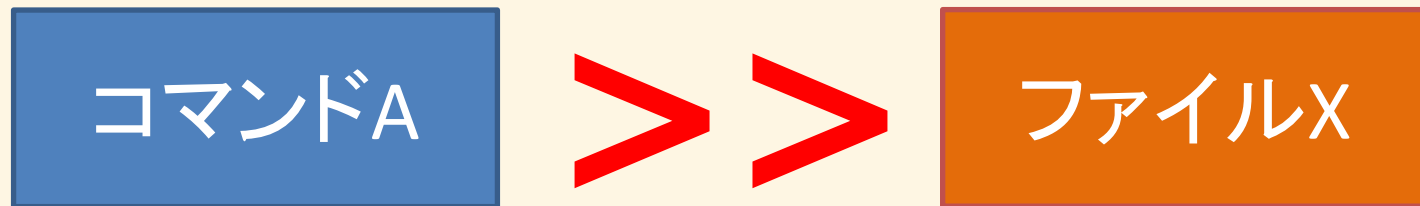
コマンドA



ファイルX

リダイレクトを利用する

- “>>”を用いることで、コマンドの実行結果を、ファイル等に追記することができます。



リダイレクトを利用

```
$ grep 'iPhone' access.log > iphone.log
```

- grepでaccess.log から文字列*"iPhone"*を検索する
- iphone.logへ結果を保存する

リダイレクトを利用

```
$ grep 'Macintosh' access.log >> iphone.log
```

- grepでaccess.log から文字列"Macintosh"を検索する
- iphone.logへ結果を追記する

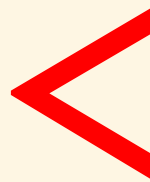
リダイレクト

- > (名前をつけて保存)
 - 標準出力に出力された情報を、指定されたファイル等に**上書き**する
- >> (追記)
 - 標準出力に出力された情報を、指定されたファイルの**末尾に追記**する

リダイレクトを利用

- “<”を用いることで、ファイルの内容をコマンドに送ることができます。

コマンドA



ファイルX

リダイレクトを利用

```
$ grep 'GET /category/games' < iphone.log | wc -l
```


- grepにiphone.logの内容を送り、文字列"/category/games"を検索する
- 検索結果の行数をカウントする

リダイレクト

- < (入力)
 - コマンドにファイル等の情報を標準入力として渡す

UNIXという哲学

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

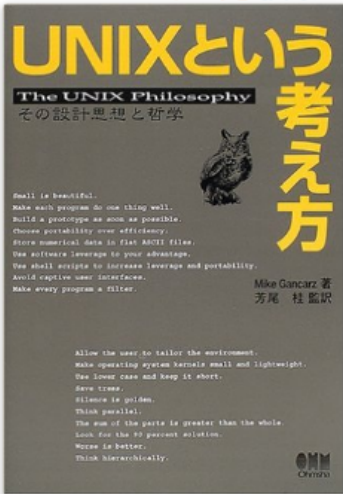
JP 勝部麻季人さん アカウントサービス

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

The UNIX Philosophy

その設計思想と哲学

Mike Gancarz 著
芳尾 桂 監訳

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）

定理9：すべてのプログラムをフィルタにする

6.2 **定理9**：すべてのプログラムをフィルタにする

コンピュータの出現以来、書かれてきたすべてのプログラムはフィルタだ。すべてのプログラムは、簡単なものでも複雑なものでも、何らかの形式のデータを入力として受け入れ、何らかの形式のデータを出力として生成する。プログラムがどのようにデータをフィルタするかは、そこに含まれているアルゴリズムによる。

ほとんどの人は、テキストフォーマッタや変換プログラムをフィルタとみなすことにはそう抵抗を感じない。しかし、フィルタとはみなされない他のプログラムも同じだということを受け入れるのは難しいようだ。例えば、リアルタイムデータ収集システムを考えてみよう。典型的なシステムは、アナログデジタルコンバータを定期的にサンプリングして、データを収集する。これが入力ストリームとなる。次に、このデータから適切な一部を抽出し、出力ストリームに送り出す。それがさらにユーザーインタフェースに送られ、他のアプリケーションに送られ、あるいはファイルに保存されることになる。

定理8：過度の対話的インターフェースを避ける

6.1 定理8：過度の対話的インタフェースを避ける

拘束的ユーザーインタフェースを避ける理由について議論を始める前に、まずそれを定義しておこう。拘束的ユーザーインタフェースとは、システムに存在するあるアプリケーションが、最上位のコマンドインタプリタの範囲外にあって、ユーザーと対話するスタイルをいう。いったん、そのアプリケーションをコマンドインタプリタから起動すると、そのアプリケーションが終了するまでコマンドインタプリタとの対話ができなくなる。ユーザーはアプリケーションのユーザーインタフェースの内部に拘束され、拘束を解くための行動を起こさない限り、その拘束から逃れられない。

分かりやすくするために、例を見てみよう。二つのプログラムがある。一つ（mail）は電子メールボックスの内容一覧を書き出し、もう一つ（search）はファイル中からテキスト文字列を探し出す。mailプログラムとsearchプログラムは、共に拘束的ユーザーインタフェースを採用しているとすると、システムとの対話は次のようになる。

\$mail	コマンドラインからmailプログラムを起動する
MAIL> dir	メールボックスの内容を表示する
	:
	:
MAIL> exit	mailプログラムを終了し、コマンドインタプリタレベルに戻る
\$search	searchプログラムを起動する
SEARCH> find jack *.txt	検索する
SEARCH> exit	searchプログラムを終了する
	:
	:
\$	もう一度コマンドインタプリタレベルに戻る

Linux基礎



デーモン

Daemon
is
Not Demon



Daemon

=

守護神

(ギリシャ神話、ダイモーン)

デーモン

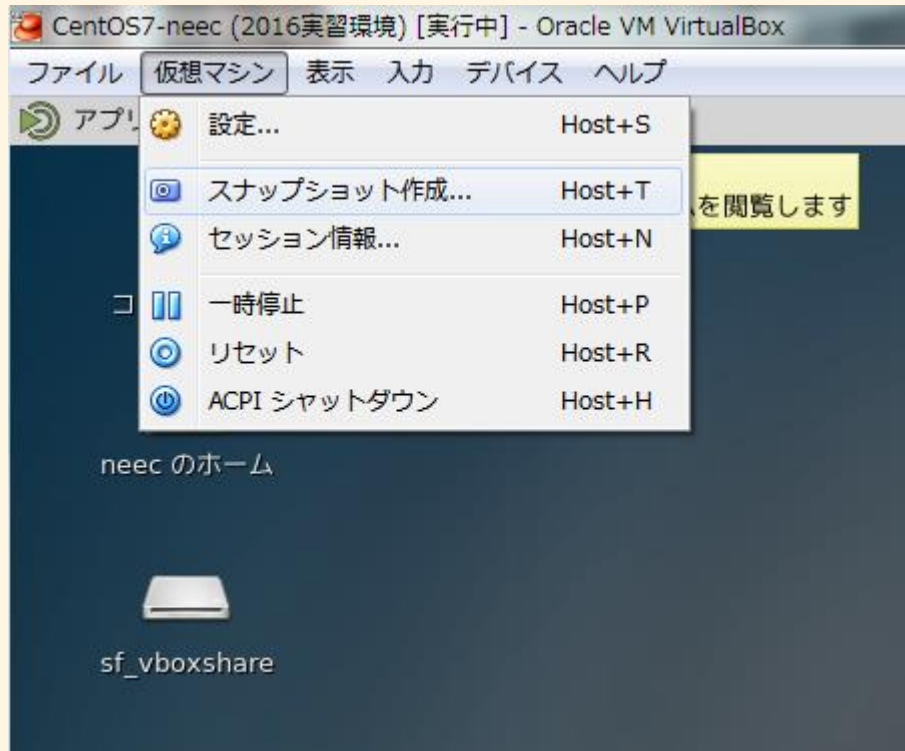
- バックグラウンドにて動作するプログラム(プロセス)
- サーバはデーモンの一種

「サーバ」を
開発する

スナップショット



VirtualBoxの 「スナップショット」機能



スナップショット
を作成しておくと、
いつでもその時点
の状態に戻せます。

「telnet」 コマンド

telnetコマンドの インストール

```
$ telnet  
telnet>
```

- telnetコマンドを打ち、このような画面になったら成功。
- “quit”と打つと、対話型のtelnetコマンドが終了します

「echoサーバ」
を動かす

Socket

Socket



「echoサーバ」 を改造する

演習3 「特定の文字列に反応する」

```
$ telnet 127.0.0.1 10000  
[Client] mountain  
[Server] river
```

- クライアントで「**mountain**」と打つと、サーバから「**river**」と返すように修正しましょう。

演習4 「ファイルを返却」

```
$ telnet 127.0.0.1 10000  
[Client] /var/www/html/hello.html  
[Server]  
(hello.htmlの中身を返却)
```

- クライアントでファイルのパスを指定されると、サーバはそのファイルの内容を返却する。

サーバサイド プログラミング



午前中は
まるっと
Linuxを使った
演習です

PHPの裏側や
そもそもの話しは
来週やります

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
 - C言語のデータ構造なども取り上げる
 - 制御構造
 - `if / switch / while / for` などが利用できる
- 演習
 - 目的はPHPの文法に慣れること

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
 - 目的はPHPの文法に慣れること

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

Mike Gancarz 著
芳尾 桂 監訳

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においての試作は、効率的な設計には欠かせない重要な一部だ。

演習サマリー6

- アンケートの記入時間
 - 今回は15分程度まとめる時間を作る
(いつも作れなくて申し訳ない)
- お知らせ
 - オセロ盤は来週も使うので、できなかった人は完成させてきてね。