

IP アドレスとポート・クライアントサーバモデル

樋口さぶろお

龍谷大学工学部数理情報学科

応用プログラミング☆実習 L09(2017-11-21 Tue)

最終更新: Time-stamp: "2017-12-17 Sun 11:19 JST hig"

今日の目標

- TCP/IP のソケット通信のクライアントプログラムが説明できる.
- IP アドレス, ネットワーク部, ホスト部, ポートの意味が説明できる.



<http://hig3.net>

ここまで来たよ

9 IP アドレスとポート・クライアントサーバモデル

- 演説
- ホスト名, IP アドレス, ポート
- クライアントサーバモデル
- ソケット - クライアントのプログラムから使う

学習目標

講義概要 → シラバス

計算機プログラムは、様々な操作を高速・効率的に処理したいという動機や目的があって作成されるものです。したがって、実用的なプログラムを作成する際には、現実の様々なデータ形式に応じた計算や入出力を行う必要があります。この科目では、画像データやネットワークデータなどの具体的なデータを用いたC言語のプログラミングを体験します。これらを通じて、より応用的なプログラミングの作法を学びます。

到達目標 → シラバス

- 現実の多様なデータを処理するためのプログラミング作法 (ex. ネットワークプログラミング) を修得する。
- 少し大きな規模のプログラムソースを管理・作成できるようになる。

ネットワークプログラミング

- ① TCP/IP ソケットの使い方
- ② クライアントサーバシステム
- ③ プロトコル (複数のプログラムの協調するための規則)

(

応用プログラミング☆実習の評価ののり

成績計算科目の前半 50+5 ピーナッツ, 後半 (樋口担当分)50+5 ピーナッツ.

	応用プログラミング☆実習 (後半)	確率統計☆演習 I
期末試験	0 ピーナッツ	45 ピーナッツ ファイナルトリアル 1 回
小テスト	35 ピーナッツ trial, 予習復習問題	30 ピーナッツ プチテスト 1 回
平常点	20 ピーナッツ 実習内課題, プログラム提出, 課題提出	25 ピーナッツ trial, 予習復習問題, 授業内課題
計	55 ピーナッツ	100 ピーナッツ

後半 50 ピーナッツは

- 15+5 ピーナッツ:平常点 実習での課題提出など
- 35 ピーナッツ:小テスト 授業期間中の e ラーニングや紙の非参照テスト

欠席届 毎回出席を前提に進めます. やむを得ず欠席して, ピーナッツ的に考慮されたい場合は, 事前 or 事後 (最終回まで) に専用用紙に事情を説明する書類を貼って, 授業前後各 5 分に提出 (事前事後とも可. ファイナルトリアルが締切). 不合格の理由はピーナッツのみで欠席回数ではありません.

ふだんは欠席の事前連絡は不要, チーム活動するとき (言います) 必要.

他の科目や研究室との関係

- 情報処理の基礎:計算機基礎実習:アルゴリズム及び実習
= ネットワーク構成論 (3年後期情報教職必修):応用プログラミング
☆実習後半:特別研究
- ネットワーク構成論は情報系の人にはセットでおすすめ. この科目では, 内部の仕組みや規約は必要なものだけを語り, “Cではこう書くところなる” ことに限定していきます.

関係する研究テーマ

担当者ののり

- なまえ: 樋口さぶろお hig@math.ryukoku.ac.jp
- へや: 1-502
- 樋口オフィスアワー月 3.5 – 4.5(1-502), 金 4(1-502)
- Web ページ: <http://hig3.net> (表紙に QR コード) 演習の指示や, スケジュールもここから.

学習サポート hig3.net 内の配布資料と課題のページ

<http://hig3.net> → 応用プログラミング

[Learn Math Moodle \(LMS\)](#)

<http://hig3.net> → Learn Math Moodle (全学認証) → 応用プログラミング☆実習

[Ryukoku Applied Math Mahara \(eポートフォリオ\)](#)

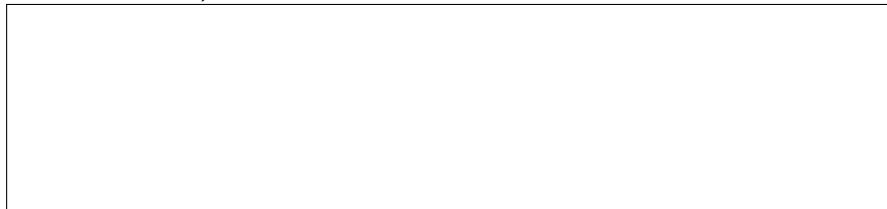
<http://hig3.net> → RaMMahara → グループ → 応用プログラミング☆実習

参考書

- 滝根哲哉編著, 「情報通信ネットワーク」の第10章, オーム社, 2013. ネットワーク構成論 (2017) の教科書. 学内なら Web で読める <https://opac.ryukoku.ac.jp/webopac/BB21553005>
- 阪田史郎編著, 「インターネットプロトコル」の第9章, オーム社, 2005. ネットワーク構成論 (2017) の参考書.
- 青木峰郎, 「ふつうのLinuxプログラミング」の第15章, ソフトバンククリエイティブ, 2017.

世の中にこんなプログラムある?

あるアプリ A に ABC と入力すると, 別のアプリ B(あるいはもう 1 個起動したアプリ A) から ABC と出力される.



C プログラムでそうするには?

すでに学んだことだけ使って, 実現方法を考えよう.

あるプログラム a.out に ABC と入力すると, 別のプログラム b.out から ABC と出力される.



ここまで来たよ

9 IP アドレスとポート・クライアントサーバモデル

- 演説
- ホスト名, IP アドレス, ポート
- クライアントサーバモデル
- ソケット - クライアントのプログラムから使う

インターネット

OSI の参照モデル	インターネット
アプリケーション層	アプリケーション層
プレゼンテーション層	
セッション層	
トランスポート層	TCP/UDP
ネットワーク層	IP
データリンク制御層	フレーム ネットワーク
物理層	イーサネット インターフェース層

TCP/IP=Transport Control Protocol / Internet Protocol
 プロトコルとは?

物理層:有線・無線ネットワーク

コンピュータをネットワークから切り離すには? どのケーブル?

ホスト名, IP アドレス, ポート

インターネットにつながったホスト (コンピュータ), ホスト上で (通信しようとする) プロセスを識別するには?

IP アドレス IP address (IPv4)

同じ IP アドレスとインターネット上のホストは 1 対 1 に対応する (例外を除いては)

人に優しく, 8 ビットずつドットで区切って (ドット区切り十進記法) 10.0.126.120 のように書くのが普通. 左ほど上の桁.

- 近い整数のホストはネットワーク的に近くにいる.
 - ▶ 例えば, 龍谷大学ネットワークに直接つながったホストはの IP アドレスは である
- 勝手に付番できない.

- 世界で IANA が管理. ネットワーク管理者が一部分を管理.
- いくつかの IP アドレスには特別の用途
- IP アドレス-ホスト 1 対 1 対応の例外: グローバルアドレスはその通りだけど, プライベートアドレス (閉じた組織内で使われる内線番号) はそうではない.
- IP アドレス-ホスト 1 対 1 対応の例外: 複数のネットワークインターフェースを持つホスト (ループバックアドレス 127.0.0.1 はその例)
- DHCP: 自動設定

IP アドレス IP address (IPv6)

インターネット上のホストを 1 個ずつ区別する 128 ビットの整数.
32 ビットずつコロンで区切って, 16 進法で表現し,
2001:DB8:0:0:8:800:200C:417A (省略) \Rightarrow 2001:DB8::8:800:200C:417A のように書くのが普通.

IP アドレスの個数?

IPv4, IPv6 ではそれぞれ 10^n 個のホストを区別できる?

ホストネーム hostname FQDN=Fully Qualified Domain Name

IP アドレスと文字列をほぼ 1 対 1 に対応させたもの.

- 例: `sirius.ws.ryukoku.ac.jp`
 - `sirius` を (狭義の) ホスト名
 - `ws.ryukoku.ac.jp` をドメイン名 (組織を表す),
 - `sirius.ws.ryukoku.ac.jp` を FQDN という.
-
- 右の方が大分類, 左のほうが大分類 (IP アドレスと逆).
 - ネットワーク的な近さでなく, 所属, 運営組織で分類されている.

世界では ICANN, 日本では JPNIC, 組織内ではネットワーク管理者が管理.

特別なホスト名, IP アドレス

コンピュータ, プログラム, 人間にとって

127.0.0.1 localhost

プログラムが実行されているそのコンピュータのこと. ループバックとも言う.

人間 (教科書の読者) にとって

example.com, example.co.jp, ...

例のための実在しないホスト名 (龍谷花子さんのようなもの)

IP アドレスとホストに関する Linux コマンド

IP アドレスとホスト名の間を変換するソフトウェア/関数

```

1 % ifconfig                /* 実行しているホストのIPアドレス他
2 % host www.ryukoku.ac.jp /* ホスト名->IPアドレスを知る */
3 % host 133.83.83.3        /* ホスト名<-IPアドレスを知る */
  
```

ポート port

1 個のホスト上で動いているプロセスを区別する 16 ビット整数.

- well-known port numbers 0–1023. 特定のサービス (=サーバ) が使うことになっている. IANA が管理. そのホストの管理者権限がないと使えない (特権ポート)
- 登録済みポート番号 1024–49151 の範囲に散在. 他の目的に使ってもよい. IANA が管理.
- プライベートポート番号 49152–65535.

- 本当はプロセスを区別でなくプログラムの連絡口 (ソケット) を区別. 1 個のプログラムが何個もの通信を同時に行うことはありうる.
- っていうか, かならず 2 個は使う.
- ポートスキャン, ポートフォワーディング, ...

ホスト:ポート組み合わせ表記

コロンの前に FQDN や IP アドレス, 後にポート番号

cache.st.ryukoku.ac.jp:8080

ここまで来たよ

9 IP アドレスとポート・クライアントサーバモデル

- 演説
- ホスト名, IP アドレス, ポート
- クライアントサーバモデル
- ソケット - クライアントのプログラムから使う

クライアントサーバモデル

通信する 2 個のプロセスは対等ではない。

- **サーバ** server 給仕 (お店) サービスする。
- **クライアント** client 客 サービスされる。

プロセスの動いているホストのことを指す場合も。
TCP/IP 以外のレイヤー/文脈でも使われる。

- サーバは要求を待つ。
- (クライアントは人間の入力からリクエストを作成する)
- クライアントは特定のサーバ (hostname:port) に**サービス** (service) を**リクエスト** (request, 要求) する。
- サーバは要求に応じた**レスポンス** (response, 返答) を返す。
- クライアントはレスポンスを受け取る。
- (クライアントはレスポンスを利用した処理を行い出力する)

例・ Web サーバ Web ブラウザ

ここまで来たよ

9 IP アドレスとポート・クライアントサーバモデル

- 演説
- ホスト名, IP アドレス, ポート
- クライアントサーバモデル
- ソケット - クライアントのプログラムから使う

こんなの全暗記してられる? だいたいの仕組みがわかって、マニュアルと Web とサンプルを参照しながら書ければ OK. API マニュアルと Web とサンプルの解説にかけてはプロになる必要。

ソースコード 1: daytime クライアント

```
1 #include <stdio.h>
2 #include <string.h> /* memset */
3 #include <sys/socket.h> /* socket, connect */
4 #include <arpa/inet.h> /* htons */
5 #include <netinet/in.h> /* htons, inet_addr */
6 #include <unistd.h> /* close sizeof */
7
8 #define SERVER_ADDR "127.0.0.1"
9 #define SERVER_PORT 12345
10 #define BUFFERSIZE 1024
11
12 int main(){
13
14     int s;
```

```
15 struct sockaddr_in sa;
16 int count;
17 char buf[BUFFERSIZE];
18
19 /* ソケットを作る */
20 s = socket(AF_INET, SOCK_STREAM, 0);
21
22 /* 接続先のホスト:ポート をソケットアドレス構造体に格納 */
23 memset(&sa, 0, sizeof(sa));
24 sa.sin_family=AF_INET;
25 sa.sin_port=htons(SERVER_PORT);
26 sa.sin_addr.s_addr=inet_addr(SERVER_ADDR);
27
28 /* 接続する = open */
29 connect(s, (struct sockaddr *)&sa, sizeof(sa));
30
31 /* 読む scanf */
32 count=recv(s, buf, BUFFERSIZE-1, 0);
33
34 /* 終端文字を加えて文字列にして出力 */
35 buf[count]='\0';
```

```
36     printf("%s", buf);
37
38     /* close */
39     shutdown(s, SHUT_RDWR);
40     close(s);
41
42     return 0;
43
44 }
```

sizeof 演算子

```
int x;
```

のとき, sizeof (int), sizeof x はバイト長 (整数) を返す
常に sizeof(char)==1

サービスの仕様やポート番号の探し方

- /etc/services サービス名とポート番号の一覧
- IETF の管理する RFC=Request for Comment

<https://www.ietf.org/rfc.html> インターネットにまつわる半公式仕様書 (IETF が管理). サービス名, ポート番号, プロトコルが説明されている.

ソケットクライアントの処理の流れは？

ソケット通信	(ファイル入力での例え)	関数
ソケット作成	fopen	socket
接続	fopen	connect
読み取り/書込	fscanf, fprintf	recv, send
接続終了	fclose	shutdown
ソケット廃棄	fclose	close

fopen-fscanf-fclose でなく、低レベルファイル入出力 open-read-close のほうがよく対応する。

関数解説

```
1 #include <sys/types.h>
2 #include <sys/socket.h>
3
4 int socket(int family, int type, int protocol);
5 /* ソケットを作り, デスクリプタを返す */
6
7 int connect(int sockfd, struct sockaddr *address, int address_len);
8 /* デスクリプタ sockfd で指定されるソケットから,
9 ソケットアドレス構造体 address で指定される通信先に接続要求を
10 接続失敗なら負の戻り値*/
11
12 int recv(int sockfd, char *buf, int len, int flags);
13 /* bufは受信バッファ, len は受信最大バイト数 */
14 /* 戻り値は受け取ったバイト数. 終了なら0, 失敗なら-1 */
15
16 int send(int sockfd, char *buf, int len, int flags);
17 /* bufは送信データ, len は送信バイト数 */
18 /* 戻り値は送信したバイト数. 失敗なら-1 */
```



```
1 #include <sys/socket.h>
2 int shutdown(int sockfd, int how);
```

```
1 #include <unistd.h>
2 int close(int sockfd);
```

詳細の暗記は不要. そういう関数あったな, で.

こういう解説は, ターミナルで (シェルで) `man` コマンドを使ってオンラインマニュアルで探せる (だいたい難解).

```
1 $ man socket
2 $ man -k socket
3 $ man 2 socket
```

2 は関数, 1 はコマンド...

どこのヘッダファイルで定義されているか探す野蛮な方法.

```
1 % grep connect /usr/include/*.h /usr/include/sys/*.h
```

ホスト:ポートの指定方法

プロトコル IP で、ホスト:ポートを指定する構造体. in = Internet

```
1 #include <netinet/in.h> /* /usr/include/netinet/in.h */
2
3 struct sockaddr_in {
4     /* 型          メンバー名*/
5     sa_family_t    sin_family; /* アドレスファミリー*/
6     in_port_t      sin_port;   /* ポート番号 */
7     struct in_addr sin_addr;   /* IPアドレスを収容する構造体 */
8     char           sin_zero[8]; /* 不使用 */
9 }
10
11 struct in_addr {
12     in_addr_t    s_addr; /* 16ビット整数で表されたIPアドレス */
13 }
```

他のアドレスファミリーまで含めた汎用の `sock_addr` 構造体は `socket.h` で定義. `/usr/include/*/sys/socket.h`

```
1 /* IPアドレスの, 32ビット整数と, ドット区切り十進記法の文字列  
2 long inet_addr(char *dot_ip);  
3 char *inet_ntoa(long integer_ip);
```

ホストバイトオーダー (x86 はリトルエンディアン) とネットワークバイトオーダー (ビッグエンディアン) の変換をする関数. network to host, long/short

計算機システム II

```
1 unsigned long ntohl(unsigned int n);  
2 unsigned long htonl(unsigned int h);  
3 unsigned int ntohs(unsigned int n);  
4 unsigned int htons(unsigned int h);
```

お知らせ

- 樋口オフィスアワー月 3.5 - 4.5(1-502), 金 4(1-502)
 - ごめんなさい 2017-11-28 火 12 都合により休講. たぶん 12 月中に補講.
 - L10 の最初で非参照のテストやります... プログラムを書けじゃなくて,
 - ▶ 選択肢で答える問題
 - ▶ 日本語で答える問題
 - ▶ プログラムを読んで何か答える問題
- の予定
- Learn Math Moodle に予習問題を載せるので, それで準備してね.