**---今回は前回作成したサーバーに対応したクライアントを作成します---**

* **内容**今回はUDPクライアント側を作成して、前回作成のサーバーと今回のクライアントで通信を行います。
* **重要キーワード**

ユニキャスト

マルチキャスト

ブロードキャスト  
DNS

* **目標**DNSサーバからサーバのIPアドレスを取得し、接続を行う。

サーバとクライアントの接続、データ通信の流れをUDP通信で学ぶ。

**---用語解説---**

**-ユニキャスト-**

ユニキャストとは1対1の通信のことを指します。  
TCP通信はソケットに通信相手が指定されているので、ユニキャスト方式で通信を行っています。サーバから接続者全員に送信している処理も実際はサーバとクライアントで1対1の通信を何回も繰り返しているだけです。

**-マルチキャスト-**

コンピュータネットワークにおいて、決められた**特定**の複数端末に対して同時にデータを送信する1対 複数 の通信のこと指します。  
TCPと異なり、UDP通信で使用されることから信頼性の求められる通信には不向きです。

**-ブロードキャスト-**

ダイアグラム

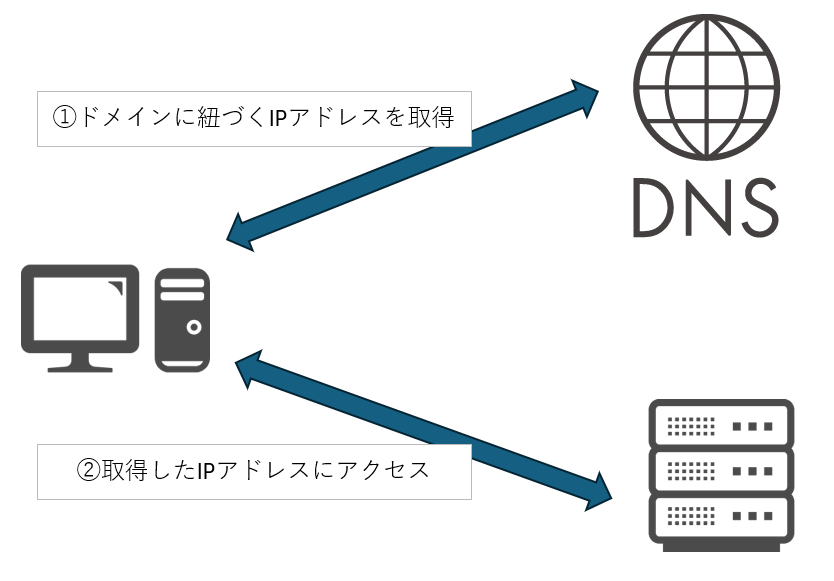
自動的に生成された説明同時通報と言う意味で、不特定多数に同じデータを送ること。（ネットワーク上末端に対しばらまくイメージです）  
コンピュータネットワーク上にある、**すべて**の端末に対して同時に同じデータを送信する1対 複数 の通信のことを指します。  
  
右画像ではオレンジ線が

それぞれの繋がりを指します。

**-DNS-**

IPアドレスは、複数桁の番号から構成されており、その番号によって接続先が決定されます。  
しかし、ブラウザでWebサイトを開く時、直接番号を入力することはあまりありませんし、意識することもありません。なぜかというと、これはDNSという、IPアドレス問い合わせシステムが利用されているからです。

DNS(Domain Name System)とはインターネット上で人間が理解しやすいドメイン名（例：example.com）を、コンピュータが通信に使用するIPアドレス（例：192.0.2.1）に変換する仕組みです。  
DNSサーバはドメインからIPアドレスを変換する機能を持ったサーバです。通常、DNSサーバにドメインを渡し、IPアドレスを取得する流れとなります。



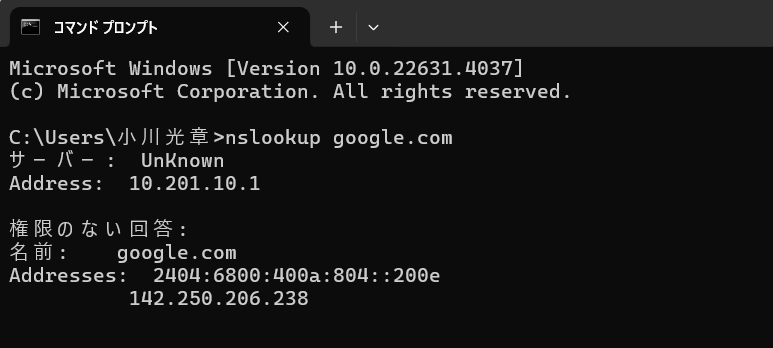
google.comって何番？

142.250.206.238だよ

142.250.206.238へ接続

ようこそgoogleです

試しに、コマンドプロンプト(黒画面)から[nslookup google.com⤵]と打ってみてください  
googleのIPアドレスが出てきます、ブラウザでは入力された文字からIPアドレスを調べ、そのアドレスへアクセスするようになっています。



nslookupで調べたいドメイン名を入れると

IPアドレス(IPV6 , IPV4)が返ってくる

WinsockでDNSサーバへの問い合わせを行うには**getaddrinfo**関数を使用します。  
getaddrinfo関数を使用するには<WS2tcpip.h>をインクルードする必要があります。  
addrinfoにDNSサーバへの問い合わせに必要な情報を設定します

addrinfo構造体

struct addrinfo {

int ai\_flags; // オプションフラグ

int ai\_family; // アドレスファミリ (例: AF\_INET, AF\_INET6)

int ai\_socktype; // ソケットタイプ (例: SOCK\_STREAM, SOCK\_DGRAM)

int ai\_protocol; // プロトコル (例: IPPROTO\_TCP, IPPROTO\_UDP)

size\_t ai\_addrlen; // アドレスの長さ

struct sockaddr \*ai\_addr; // アドレス情報

char \*ai\_canonname; // 正式な名前

struct addrinfo \*ai\_next; // 次のアドレス情報へのポインタ

};

この構造体の中から必要なパラメータの設定を行います。  
今回の課題であればアドレスファミリとソケットタイプの2種類になります。  
今回はIPv4のIPアドレスを取得するため、ai\_familyにはAF\_INETを指定。またDNSサーバへのアクセスにはTCPを使用するためSOCK\_STREAMを指定します。(SOCK\_DGRAMを指定しても可能です)

**今回課題でのコード例**

#include <WS2tcpip.h>

//

// 省略

//

// DNSへ問い合わせ

addrinfo hints; // DNSへ問い合わせに必要な情報を設定

addrinfo\* addrInfo = NULL; // 取得したアドレスがここに保存される

// ゼロクリア

ZeroMemory(&hints, sizeof(addrinfo));

// 設定する情報を設定

hints.ai\_family = AF\_INET; // IPv4で取得

hints.ai\_socktype = SOCK\_STREAM; // TCP通信でDNSサーバへアクセス（確実性の為TCPを利用）

const char hostname[] = "localhost"; // ドメイン名指定(今回課題では[localhost])

const char port[] = PORTSTR; // ポート番号指定

// DNSへ問い合わせ0のとき正常に完了(addrInfoにIPが入ってくる)

if (getaddrinfo(hostname, port, &hints, &addrInfo) != 0) {

Logger::Print("getaddrinfo error.\n");

return;

}

// TODO 01\_12

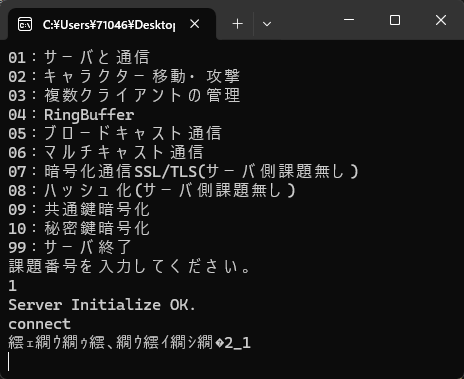
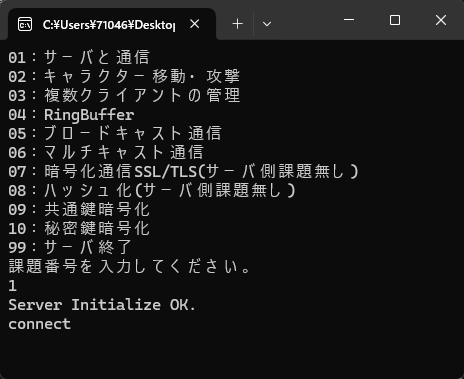
// WinsockAPIを初期化

// 取得したIPアドレスをsockaddr\_inに変換しsin\_addrを代入

addr.sin\_addr = reinterpret\_cast<sockaddr\_in\*>(addrInfo->ai\_addr)->sin\_addr;

addr.sin\_family = AF\_INET;

addr.sin\_port = htons(PORT);

* ******課題内容  
  ・クライアント側を作成します。**

この後はMessageに入力されたテキストをSendボタンでサーバに送信し、サーバから送り返されてきたデータを表示する

送られてきたデータをそのままクライアントへ送信。

起動時にサーバにクライアント情報を知らせるためにデータ送信。(図ではconnectを送信)

サーバIPをDNSから取得し、サーバへ接続を行うこと。ドメイン名は「**localhost**」でDNSサーバに問い合わせ、取得したIP番号を使用して接続を行う事 (実際に取得する番号は127.0.0.1となる。)  
またUDP通信を用いてサーバとデータの送受信を行えるようにしなさい。  
(TODO 01\_10～19)