ネットワークアーキテクチャ

第12回

ネットワークプログラミング（2）

演習課題

- ウェブクライアント・サーバの作成
- クライアント
  - 指定したURIの情報を取得
- サーバ
  - 要求された情報を送信

⇨ HTTPの実装

HTTP（HyperText Transfer Protocol）

- Webコンテンツを送受信するプロトコル
- 基本的にはテキストメッセージの交換
- HTTP/1.0
  - 1コネクションで，1回の「要求」，「応答」
- HTTP/1.1
  - 1コネクションで，複数回の「要求」，「応答」

動作例

% telnet www.kitakyu-u.ac.jp 80
Trying 202.245.2.41…
Connected to www.kitakyu-u.ac.jp.
GET /env/index.html HTTP/1.1
Host: www.kitakyu-u.ac.jp

改行
<CR><LF>
（¥r¥n）

HTTP/1.1 200 OK
Date: …
Content-Length: 19873
Content-Type: text/html

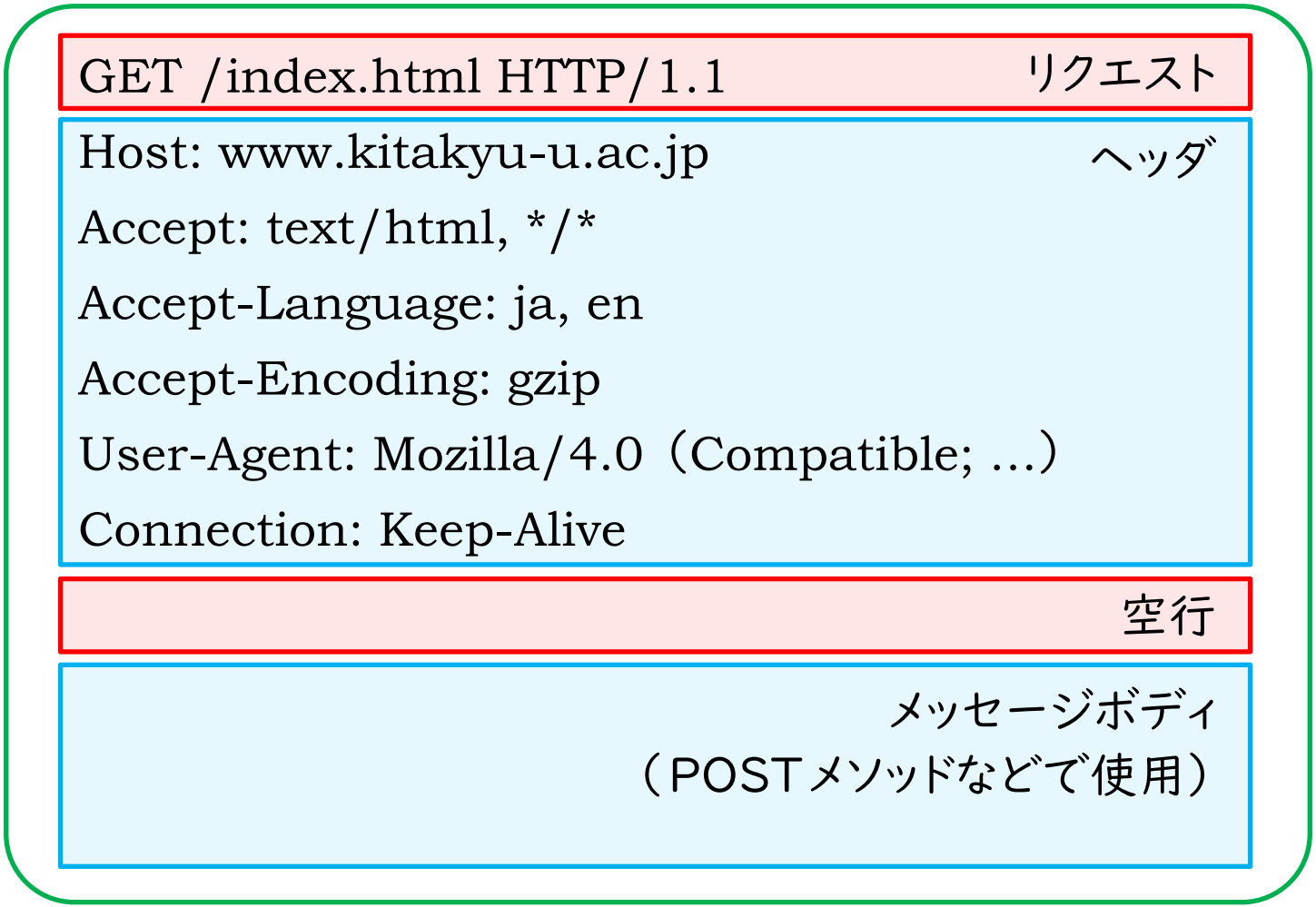
ヘッダ

<!DOCTYPE HTML PUBLIC …
<html lang="ja">
…

コンテンツ

メッセージ構文

- 要求メッセージ



メッセージ構文

- 応答メッセージ

HTTP/1.1 200 OK　　レスポンス

Date: Thu, 09 Jul 2009 12:00:00 GMT　　ヘッダ
Server: Apache/2.2.9（Unix）
Last-Modified: Sun, 28 Jun 2009 …
Content-Length: 1234
Connection: close
Content-Type: text/html

空行

<html>　　メッセージボディ
…
</html>

リクエスト

- メソッド パス HTTP/バージョン

  （GET /index.html HTTP/1.1）
- 主要なメソッド
  - GET:　コンテンツの取得
  - HEAD: ヘッダ情報のみ取得
  - POST: データをサーバへ転送（CGIなどで利用）
  - PUT:　ファイルのアップロード

レスポンス

- HTTP/バージョン レスポンスコード メッセージ
  （HTTP/1.1 200 OK）
- 主要なレスポンスコード
  - 200 OK: 成功
  - 403 Forbidden: アクセス拒否
  - 404 Not Found: 存在しない
  - 501 Not Implemented: メソッドが利用できない
  - 503 Service Unavailable: サービスが利用不可

ヘッダ

- プロパティ：値
- 主要なヘッダ情報
  - Host：ホスト名の指定（仮想ホスト対応）
  - Server：サーバ情報
  - User-Agent：ブラウザ情報
  - Date：日付，時刻情報
  - Connection: 持続接続の指定（Keep-Alive, close）
  - Accept関連：受信可能な形式の指定
  - Content関連：コンテンツ情報

仕様書

- Request For Comments（RFC）
  - IETFで策定された標準規約
- RFC1945
  - “Hypertext Transfer Protocol -- HTTP/1.0”
  - http://www.ietf.org/rfc/rfc1945.txt
- RFC2817
  - “Upgrading to TLS Within HTTP/1.1”
  - http://www.ietf.org/rfc/rfc2817.txt

レポート課題

- ウェブクライアント・サーバの作成
- クライアントの仕様
  - ウェブページを取得し，表示（HTMLのままでOK）
  - URI（ポート番号含む）をコマンドライン引数で指定
- サーバの仕様
  - GETメソッド，応答コード200，404，501に対応
  - ブラウザでもアクセスできること
- 余裕があれば，

  他のメソッド，応答コード，ヘッダ情報などにも対応

実行例

- クライアント

```
% ./web-client http://localhost:50000/index.html
connected to 127.0.0.1
getting /index.html from localhost

HTTP/1.1 200 OK
Content-Length: 49
Connection: close
Content-Type: text/html

<html>
<body>
This is test page.
</body>
</html>
```

受信した応答メッセージ

- サーバ

```
% ./web-server
listening...
connected from 127.0.0.1

GET /index.html HTTP/1.1
Host: localhost
空行

send ./index.html file
```

受信した要求メッセージ

実行例

- クライアント

```
% ./web-client http://localhost:50000/test.html
connected to 127.0.0.1
getting /test.html from localhost

HTTP/1.1 404 Not Found
Connection: close
Content-Type: text/html; charset=iso-8859-1

<html><head>
<title>404 Not Found</title>
</head><body>
<h1>Not Found</h1>
<p>The requested URL /test.html was not
    found on this server.</p>
</body></html>
```

- サーバ

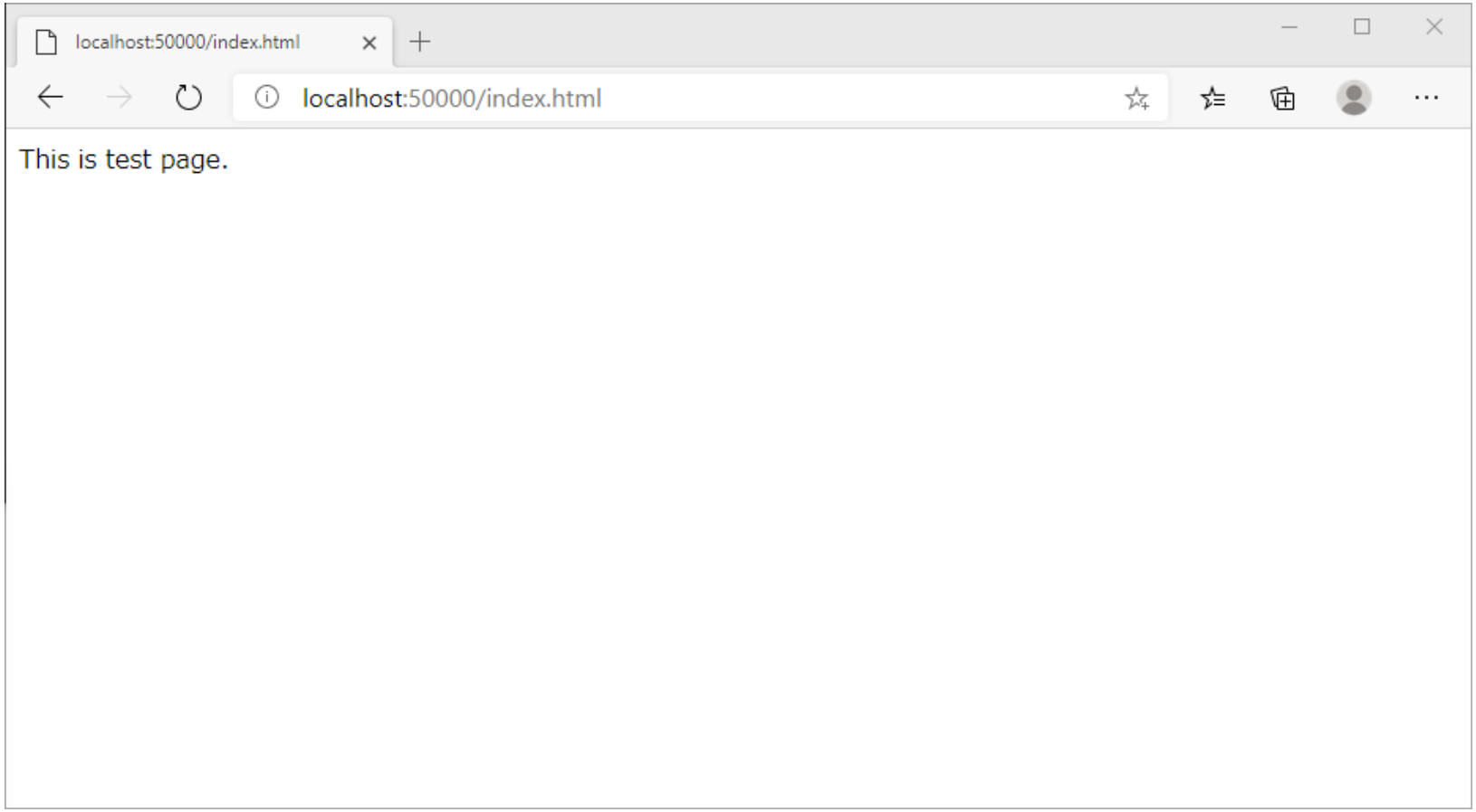
```
connected from 127.0.0.1

GET /test.html HTTP/1.1
Host: localhost
空行

send response code 404
```

実行例

- ブラウザ


localhost:50000/index.html
localhost:50000/index.html
This is test page.

ヒント：クライアント

- URIの解釈
  - http://www.example.com:50000/index.html
  - ホスト名：「http://」から「:」の間の部分
  - ポート番号：「:」から「/」の間の部分
  - パス：「/」以降の部分
  - 文字列操作関数（strchr, strcpyなど）を利用
- 要求・応答の送受信
  - 要求メッセージ（文字列）を作成し，ソケットへ書き込む
  - 応答メッセージをソケットから読み込む

URIの処理例

```
char host[1024], path[1024], *p;
int port;

sscanf(argv[1], "http://%s", host);
p = strchr(host, '/');
strcpy(path, p);
*p = '¥0';
p = strchr(host, ':');
port = atoi(p + 1);
*p = '¥0';
```

注）エラー処理は省略

ヒント：サーバ

- コンテンツの準備（例：./index.html）
  - www.example.com/index.html → ./index.html
- 要求の受信
  - メソッドとヘッダ情報をソケットから読み込む
  - メソッドによって処理を変える
- 応答の送信
  - （GETの場合）指定されたパスのファイルを開く
  - 成功：応答コード200をソケットへ書き込む
  - 失敗：応答コード404をソケットへ書き込む

recv()の注意点

- 文字列の最後に終端文字(¥0)をつけない
- 対策

⇨ 連続受信する際，問題が生じることがある

1. 標準入出力関数(fprintf()，fgets()など)を使う
2. 受信文字数を管理する

fdopen()などが必要

```
while（1）{
    int size = recv(sockfd, buff, sizeof(buff) – 1, 0);
    if（size）{
        buff[size] = '¥0';
        printf("%s", buff);
    } else {
        break;
    }
}
```

レポート内容

- 表紙
  - 科目名, 学籍番号, 氏名, メールアドレス
- HTTPに関する説明
  - 作成プログラムの関連部分（プロトコル, メソッド, 応答コードなど）
- プログラムに関する説明
  - ソースコードだけではなく, 全体的な処理の流れや内容を詳しく
- 実行結果, 考察
  - すべての仕様を満たしていることが確認できるように
  - ブラウザでアクセスした結果も含むこと（http://localhost:...）
- 感想, 参考文献

レポート提出方法

- 電子ファイルで提出
  - pdf形式, word形式
  - ファイル名： 学籍番号-report.pdf（.docxなど）
- 提出先
  - Moodle
- 提出期限
  - 2020年8月14日（金）24:00