Co to jest HTTP?

- HTTP – HyperText Transfer Protocol.
- Używany do transmisji dowolnych plików.
- Przesyłany pewnymi strumieniami.
- Działa w modelu klient-serwer.
- Serwer nasłuchuje na porcie 80.
- HTTP służy do przenoszenia zasobów.
- Zasób = dane + określenie ich rodzaju.

Więcej o zasobach

- Zasób jest identyfikowany za pomocą URI.
- URI – Uniform Resource Identifier.
- W HTTP URI nazywany jest często URL.
- URL – Uniform Resource Locator.
- Zasób to najczęściej plik lub wynik działania programu-skryptu.

Tranzakcje w HTTP

- Klient wysyła *żądanie*, na które serwer daje *odpowiedź*.
- Po odesłaniu odpowiedzi serwer zazwyczaj zamyka połączenie.
- Oznacza to, że HTTP jest bezstanowe.
- Wymieniane komunikaty mają następującą strukturę
  - wiersz początkowy,
  - zero lub więcej wierszy nagłówków,
  - wolny wiersz (CRLF),
  - dane (opcjonalnie).
- Kolejne linijki w HTTP zakończone są znakami CRLF (bajty 13, 10).

Pierwszy wiersz żądania

- Pierwszy wiersz żądania składa się z trzech, oddzielonych spacjami elementów:
  - metody żądania,
  - lokalnej ścieżki,
  - wersji protokołu HTTP.
- Zwykle wygląda to tak:

  ```
  GET /sciezka/do/pliku/index.html HTTP/1.0
  ```
- Nazwy metod są zawsze pisane dużymi literami.
- Ścieżka to ta część URL-a, która znajduje się po nazwie komputera (od 3. ukośnika począwszy).
- Określenie wersji HTTP zgodne jest ze wzorcem: "HTTP/x.x" i jest zawsze pisane dużymi literami.

Pierwszy wiersz odpowiedzi

- Inna nazwa: wiersz stanu.
- Składa się z trzech, oddzielonych spacjami elementów:
  - wersji protokołu HTTP,
  - kodu stanu odpowiedzi,
  - słownego opisu stanu w języku angielskim.
- Przykłady

  ```
  HTTP/1.0 200 OK
  ```

  lub

  ```
  HTTP/1.0 301 Moved Permanently
  ```

- Wersja HTTP ma format jak w żądaniu.
- Kod stanu przeznaczony jest dla komputera.
- Opis stanu przeznaczony jest dla człowieka i może się zmieniać.

Kody odpowiedzi

- Pierwsza cyfra określa ogólną kategorię odpowiedzi:
  - **1xx** oznacza komunikat informujący,
  - **2xx** oznacza pomyślne wykonanie żądania,
  - **3xx** przekierowuje klienta pod inny URL,
  - **4xx** oznacza błąd po stronie klienta,
  - **5xx** oznacza błąd po stronie serwera.

Przykładowe kody odpowiedzi

- **100** Continue
- **200** OK
- **301** Moved Permanently
- **303** See Other
- **400** Bad Request
- **404** Not Found
- **405** Method Not Allowed
- **500** Internal Server Error
- **501** Not Implemented
- **505** HTTP Version not supported

Wiersze nagłówka

- Zawierają dodatkowe informacje określające własności żądania/odpowiedzi.
- Ogólna postać: `Nazwa-wiersza: wartość`.
- Reguły (RFC 822):
  - każdy wiersz kończy się CRLF,
  - nie liczy się poczet liter,
  - między : a wartością może być dowolna liczba spacji lub znaków tabulacji,
  - wiersze zaczynające się od spacji lub tabulatora to kontynuacje poprzednich wierszy
- Następujące wiersze są równoważne

  ```
  Naglowek1: wartosc1, wartosc2
  ```

  i

  ```
  NAGLOWEK1:          wartosc1,
                      wartosc2
  ```

Więcej o nagłówkach

- W HTTP 1.0 zdefiniowano 16 wierszy.
- W HTTP 1.1 zdefiniowano 46 wierszy w tym jeden wymagany (Host:).
- Warto pamiętać przy wysyłaniu żądań:
  - **From:** zawiera adres e-mail wykonującego żądanie, obecność musi być konfigurowalna.
  - **User-Agent:** zawiera identyfikację programu wykonującego żądanie; format "Nazwa-programu/x.xx", np.

```
Mozilla/4.0 (compatible; MSIE 5.01; Windows NT 5.0)
```

- Warto pamiętać przy wysyłaniu odpowiedzi:
  - **Server:** podobnie jak User-Agent:, przykład

    ```
    Server: Apache/1.2b3-dev
    ```
  - **Last-Modified:** podaje datę ostatniej modyfikacji podawanego zasobu, przykład:

    ```
    Last-Modified: Wed, 05 Mar 2003 23:47:59 GMT
    ```

Dane

- Najczęściej zawierają żądany zasób.
- Mogą zawierać stronę z opisem błędu.
- W przypadku żądań zawierają dane przesyłane do serwera: plik, treść formularza.
- Gdy występują nietrywialne dane, pojawiają się następujące nagłówki:
  - **Content-Type:** zawiera typ MIME danych, np.: text/html lub image/png,
  - **Content-Length:** zawiera długość danych.

Przykład tranzakcji HTTP

```
GET /~alx/WWW/http.php HTTP/1.1
Host: www.mimuw.edu.pl
From: alx@mimuw.edu.pl
User-Agent: telnet/1.0

HTTP/1.1 200 OK
Date: Wed, 05 Mar 2003 15:04:08 GMT
Server: Apache/1.3.26 (Unix)  (PLD/Linux)
     mod_fastcgi/2.2.10 PHP/4.2.2
X-Powered-By: PHP/4.2.2
Content-Type: text/html

<html>
...
</html>
```

Inne metody

- Oprócz GET istnieją metody: OPTIONS, HEAD, POST, PUT, DELETE, TRACE, CONNECT.
- Najczęściej wykorzystywane metody to: GET, POST, PUT, HEAD.
- Metoda HEAD:
  - Działa podobnie jak GET.
  - Służy do pobrania informacji o zasobie bez pobierania jego samego.
  - Dane są zawsze puste.

Inne metody c.d.

- Metoda POST:
  - Służy do przesyłania danych, które serwer ma przetworzyć.
  - Różnice w stosunku do GET:
    * w żądaniu przesyłane są zwykle pewne dane,
    * zwykle obecne są nagłówki Content-Type: czy Content-Length:,
    * URI wskazuje na program,
    * odpowiedź to wynik działania programu.
  - Służy do obsługi formularzy (Content-Type: application/x-www-form-urlencoded).
  - Przykład żądania POST

    ```
    POST /ścieżka/program.cgi HTTP/1.0
    From: alx@mimuw.edu.pl
    User-Agent: Mozilla/4.0
    Content-Type: application/x-www-form-urlencoded
    Content-Length: 31

    gdzie=tutaj&w+jaki+sposob=dlugo
    ```

  - GET też może służyć do wysyłania formularzy.

Połączenia trwałe w HTTP/1.1

- Jeśli każda tranzakcja wymaga nowego połącznia, to marnuje się przepustowość i czas.
- Wprowadzono połączenia trwałe (ang. persistent).
- Jeśli klient dołączy wiersz `Connection: close`, to połączenie zostanie zamknięte po odpowiedzi serwera.
- Serwer nie musi obsługiwać połączeń trwałych, więc trzeba zachować ostrożność przy wysyłaniu do niego szeregu żądań.

Specyfikacja HTTP

- Opisy HTTP:
  - HTTP 1.0 — RFC 1945,
  - HTTP 1.1 — RFC 2068.
- Inne pomocne dokumenty
  - RFC 822 – opis komunikatów tekstowych w Internecie,
  - RFC 2396 – definicja składni URL/URI,
  - RFC 1521 – definicja MIME i (podstawowych) typów MIME.
- Protokołem HTTP opiekuje się World Wide Web Consortium:

  `http://www.w3c.org/`