

Лекция 7. HTTP сервер

Задача — реализовать примитивный HTTP сервер, к которому можно было бы обращаться от стандартного интернет-браузера. Таким образом, сервер должен передавать браузеру некоторый HTML текст, предваряя его соответствующим HTTP заголовком. В свою очередь, браузер, выполняя функции клиента, должен каким-то образом получать от пользователя требуемые данные и также формировать нужный HTTP запрос к серверу.

Проблемы:

1. Сложность всесторонней поддержки HTTP протокола, множество технических тонкостей.

2. Представление результата в форме HTML документа. Опять же, HTML имеет множество разнообразных средств и возможностей для представления данных.

3. Логика работы конкретного браузера с HTTP протоколом (параллельные соединения, упреждающие запросы, служебные запросы и т.д.)

Но, к счастью, сам HTTP устроен так, что общение возможно и при самой примитивной реализации сервера.

Задача сервера — понять, что от него хочет клиент (браузер).

Задача клиента — принять данные для запроса от пользователя.

Начнем с клиента.

Понятие о HTML — HyperText Markup Language.

Исторически прародителем HTML является SGML — Standard Generalized Markup Language — стандартный обобщённый язык разметки. Это довольно громоздкая конструкция, из которой позднее выделилось более удобное подмножество XML — eXtensive Markup Language — расширяемый язык разметки, используемый для структуризации и разметки данных, если связь между элементами данных чем-то напоминает дерево (хотя это и не обязательно).

Для простейшего понимания достаточно двух понятий:

тег — элемент разметки `<tag> ... </tag>`, некоторые теги могут не иметь закрывающей части

атрибуты — поименованные “параметры” тега `<tag attr="abc">`

XML вводит правила для использования тегов и атрибутов, которые формируют “окружение” каких-либо данных, текстов и т.п.

Строго говоря, HTML не удовлетворяет спецификациям XML, хотя есть направление, которое пытается этот недостаток поправить — XHTML.

Описание тегов HTML см. в справочной литературе или в интернете.

Теги используются для визуального оформления текста и для определения способов и правил использования этого текста.

Есть несколько версий HTML, а также расширений для разных браузеров. Есть проблема совместимости и различия результатов отображения. Т.е. каждый конкретный браузер интерпретирует теги, пытаясь следовать устоявшимся “стандартам”, но может также и действовать по своему разумению.

Простейший HTML текст.

```
<!DOCTYPE html>
<html>
<head>
  <meta charset = "cp1251">
```

```

<title>Учебный сервер</title>
</head>
<body>
  <h2>МГУ им М.В.Ломоносова</h2>
  <p>
    Просто абзац текста
  </p>
</body>
</html>

```

Пример HTTP сервера, возвращающего данный текст.

Запрос: localhost:5555/index.html

Сервер должен проанализировать HTTP запрос, и на все непонятные запросы ответить отказом, а на понятные запросы ответить правильным образом.

Для ввода пользовательских данных используется тег <form>. Событие submit формирует HTTP запрос и отправляет его в соответствии с атрибутами action и method.

Метод get. Передача параметров ввода из формы в строке запроса.

Метод post. Передача параметров ввода из формы в рамках entity body запроса. (если данных много).

```

<!DOCTYPE html>
<html>
<head>
  <meta charset = "cp1251">
  <title>Учебный сервер</title>
</head>
<body>
  <h2>МГУ им М.В.Ломоносова</h2>
  <form name="myform" action="http://localhost:5555" method="get">
    <input name="request" type="text" width="20"> request <br>
    <input name="comment" type="text" width="20" value="some text"> comment <br>
    1 <input name="rb1" type="radio" value="check1">
    2 <input name="rb1" type="radio" value="check2" checked>
    3 <input name="rb2" type="radio" value="check3">
    4 <input name="rb2" type="radio" value="check4">
    <br>
    A <input name="ch1" type="checkbox" value="checkA">
    B <input name="ch2" type="checkbox" checked>
    C <input name="ch3" type="checkbox">
    <br>
    <input name="sub" type="submit" value="Send">
  </form>
</body>
</html>

```

Пример. HTTP сервер с обслуживанием формы.

Методы GET и POST

Особенности записи параметров:

пробелы — “+”

русские буквы и другие спецсимволы — %код