4. Tutorial

Task 1

Answer the following questions:

- What is the difference between URI and URL?
- What is the meaning of the URL scheme?
- What, why and how should be encoded in URLs?

What is the difference between URI and URL?

- URI is an abstract resource identifier (may be a unique name of the resource – URN or it's location – URL)
- URL describes a location of the resource and the protocoll used to access it

What is the meaning of the URL scheme?

- URL scheme describes a method to access a resource
- Often (but not always) corresponds to some specific protocol: http, ftp, news, ssh, file, ldap, ...

What, why and how should be encoded in URLs?

- What?
  - Segments of URLs
- Why?
  - Reserved characters : / ? # [ ] @ ! $ & % ' ( ) * + , ; =
  - Non-ASCII characters
  - Unsafe characters (whitespace, ",<,>...)
- How?
  - % + 2 Hexadecimal digits
  - Hexadecimal digits correspond to the ASCII-value of the character
  - Each byte of the UTF-8 encoding for non-ASCII symbols

Task 2

Which URLs are **syntactically** correct and which are not:

http://www.tu-chemnitz.de/informatik
http://tu-chemnitz.de/informatik
http://www.tu-chemnitz.de:443/informatik
http://www.tu-chemnitz.de/informatik?show=all?group=true
http://www.tu-chemnitz.de/informatik?show=all%20group=true
file:///c:/windows/php.ini
ftp://www.tu-chemnitz.de/informatik?show=all&group=true
ftp://bob:pass@www.tu-chemnitz.de/informatik

mailto://user@example.org
~~*test://127.0.0.1:13000/home/calc?x=1;b=1*~~

Task 3

Implement a class for parsing and generation of HTTP URLs. Use the given template bellow as a start point:

- The constructor `Url(string urlStr)` should split the `urlStr` into URL components using a **regular expression** and fill in the instance variables `Scheme`, `Host` etc. One should assume the URL is correctly encoded.
- The function `string ToString()` should concat the instance variables to the string representation of the URL.
- The static method `string Encode(string s)` should convert all characters from `s`, which are not in `VALID_CHARACTERS` into the %-form and give the resulting string back
- The static method `string Decode(string s)` should convert all %-escaped characters from s and give the resulting string back

See examples for using regular expressions and text processing in C# (*TextParsingExample.cs*).

For simplification purposes, it is enough that the given unit test passes (the solution should not be limited to the given URL though)

TextParsingExample.cs
Shared on Dropbox

Tutorial4-Task3-Template.zip
Shared on Dropbox

Task 4

- Establish a TCP connection[1] to the server of Wikipedia (en.wikipedia.org) and request the page http://en.wikipedia.org/wiki/Uniform_resource_locator by sending a corresponding HTTP message.
- Record traffic, which is sent by your browser while requesting the following URL:
- http://en.wikipedia.org/wiki/Uniform_resource_locator#Syntax. How the fragment "*#Syntax*" is handled?

[1] Use for example *telnet*, putty (http://www.chiark.greenend.org.uk/~sgtatham/putty/) or

Task 5

The following self-defined URL scheme should be used to access environment variables of a remote machine: `env://host:port/environmentvar?ask`

Example: `env://127.0.0.1:13000/NUMBER_OF_PROCESSORS?ask`. Use the URL class from the Task 3 and the TcpClient from Tutorial 2 to implement a resolver for the URL scheme above. The resolver should take a URL as input and return the value of the requested variable. Test your implementation using the TestServer from the 2nd tutorial.