URLEncoder And URLDecoder

- description : URLEncoder And URLDecoder
- author : 오션
- email : shlim@repia.com
- lastupdate : 2022-04-14 Thu

Percent-encoding

퍼센트 인코딩은 URL 컨텍스트에서 특정 의미를 가지는 8bit 문다를 인코딩하는 메커니즘입니다.
퍼센트 인코딩은 URL 인코딩이라고 불리기도 합니다. 퍼센트 인코딩은 기호 % 다음에 대체 문자의 ASCII 값을 나타내는 16진수 표현식으로 표시됩니다.

인토딩이 필요한 특수문자는 **:**, **/**, **?**, **#**, **[**, **]**, **@**, **!**, **$**, **&**, **'**, **(**, **)**, ***, **+**, **,**, **;**, **=**, **%** 입니다. 다른 문자는 인코딩할 필요가 없지만 인코딩을 할 수 있습니다.

No	Character	Encoding	No	Character	Encoding	No	Character	Encoding	No	Character	Encoding
1	**:**	**%3A**	6	**]**	**%5D**	11	**'**	**%27**	16	**,**	**%2C**
2	**/**	**%2F**	7	**@**	**%40**	12	**(**	**%28**	17	**;**	**%3B**
3	**?**	**%3F**	8	**!**	**%21**	13	**)**	**%29**	18	**=**	**%3D**
4	**#**	**%23**	9	**$**	**%24**	14	*****	**%2A**	19	**%**	**%25**
5	**[**	**%5B**	10	**&**	**%26**	15	**+**	**%2B**	20	**' '**	**%20** or **+**

컨텍스트에 따라 **' '** 문자는 **+**(**application/x-www-form-urlencoded** 메시지에 사용되는 퍼센트 인코딩 버전에서와 같이) 또는 URL상에 표시되는 것과 같은 **%20**으로 변경됩니다.

Class URLEncoder

HTML form 인코딩을 위한 유틸리티 클래스입니다. 이 클래스에는 String을 application/x-www-form-urlencoded MIME 포맷으로 변환하기 위한 정적 메서드가 포함되어 있습니다. HTML 양식 인코딩에 대한 자세한 내용은 HTML 명세표를 참조하십시오.

문자열을 인코딩할 때 다음 규칙이 적용됩니다.

- 영숫자 문자 “a” ~ “z”, “A” ~ “Z” 및 “ 0 ” ~ “ 9 ” 는 그대로 유지됩니다.
- 특수 문자 “ . ”, “ - ”, “ * ” 및 “ _ ” 는 그대로 유지됩니다.

- 공백 문자 “ “ 는 더하기 기호 ” + “ 로 변환됩니다.
- 다른 모든 문자는 안전하지 않기 때문에, 먼저 일부 인코딩 체계를 사용하여 하나 이상의 바이트로 변환됩니다. 그런 다음 각 바이트는 3자 문자열 ”%xy”로 표시됩니다. 여기서 xy는 바이트의 2자리 16진수 표현입니다. 사용을 권장하는 인코딩 체계는 UTF-8입니다. 그러나 호환성의 이유로, 인코딩을 지정하지 않으면 플랫폼의 기본 인코딩이 사용됩니다.

예를 들어 UTF-8을 인코딩 체계로 사용하면, 문자열 “The string ü@foo-bar”는 “The+string+%C3%BC%40foo-bar”로 변환됩니다. 이는 UTF-8에서 문자 ü는 2바이트 C3(16진수) 및 BC(16진수)로 인코딩되고, 문자 @는 1바이트 40(16진수)으로 인코딩되기 때문입니다.

Class URLDecoder

HTML form 디코딩을 위한 유틸리티 클래스입니다. 이 클래스에는 application/x-www-form-urlencoded MIME 형식에서 문자열을 디코딩하기 위한 정적 메서드가 포함되어 있습니다.

변환 프로세스는 URLEncoder 클래스에서 사용하는 것과 반대입니다. 인코딩된 문자열의 모든 문자는 “a” ~ “z”, “A” ~ “Z”, “0” ~ “9” 및 “ - ”, “ _ ”, “ . ”, 그리고 “ * ” 중 하나입니다.
문자 “ % “ 는 허용되지만 특수 이스케이프 시퀀스의 시작으로 해석됩니다.

변환에는 다음 규칙이 적용됩니다.

- 영숫자 문자 “a” ~ “z”, “A” ~ “Z” 및 “ 0 ” ~ “ 9 ” 는 그대로 유지됩니다.
- 특수 문자 ” . “, “ - ”, “ * ” 및 “ _ ” 는 그대로 유지됩니다.
- 더하기 기호 “ + “ 는 공백 문자 ” “ 로 변환됩니다.
- ”%xy” 형식의 시퀀스는 xy가 8비트를 두 자리의 16진수 표현인 바이트를 나타내는 것으로 처리됩니다. 그런 다음 이러한 바이트 시퀀스 중 하나 이상을 연속적으로 포함하는 모든 하위 문자열은 인코딩이 해당 연속 바이트를 생성하는 문자로 대체됩니다. 이러한 문자를 디코딩하는 데 사용되는 인코딩 체계가 지정될 수 있으며, 지정되지 않은 경우 플랫폼의 기본 인코딩이 사용됩니다.

이 디코더가 잘못된 문자열을 처리할 수 있는 두 가지 가능한 방법이 있습니다. 잘못된 문자를 그대로 두거나 IllegalArgumentException을 throw할 수 있습니다. 디코더가 취하는 접근 방식은 구현에 달려 있습니다.

Example 1

```
package com.ocean.escapehtml;

import java.io.UnsupportedEncodingException;
import java.net.URLDecoder;
import java.net.URLEncoder;

public class PercentEncode {
```

```
    // 돌체 비타 %EB%8F%8C%EC%B2%B4+%EB%B9%84%ED%83%80

    public static void main(String[] args) throws
UnsupportedEncodingException {

        String encodedText = "돌체 비타";
        encodedText = URLEncoder.encode(encodedText,
"UTF-8").replaceAll("\\+", "%20");
        System.out.println("encodedText ::  " + encodedText);
        // encodedText ::  %EB%8F%8C%EC%B2%B4%20%EB%B9%84%ED%83%80

        String decodedText = encodedText;
        decodedText = URLDecoder.decode(decodedText, "UTF-8");
        System.out.println("decodedText ::  " + decodedText);
        // decodedText ::  돌체 비타
    }
}
```

Example 2

```
package com.ocean.escapehtml;

import java.io.UnsupportedEncodingException;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLEncoder;

public class URLEncoderExample {

    public static void main(String[] args) throws MalformedURLException,
UnsupportedEncodingException {
        // base URL
        String baseurl = "https://www.javaguides.net/search?q=";

        // String to be encoded
        String query = "core+java+tutorial";

        URL url = new URL(baseurl + query);
        System.out.println("URL without encoding : " + url);
        // URL without encoding :
https://www.javaguides.net/search?q=core+java+tutorial

        // encode() method
        url = new URL(baseurl + URLEncoder.encode(query, "UTF-8"));
        System.out.println("URL after encoding : " + url);
        // URL after encoding :
https://www.javaguides.net/search?q=core%2Bjava%2Btutorial
```

```
    }
}
```

Example 3

```
package com.ocean.escapehtml;

import java.io.UnsupportedEncodingException;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLDecoder;
import java.net.URLEncoder;

public class URLDecoderExample {

    public static void main(String[] args) throws MalformedURLException,
UnsupportedEncodingException {
        // base URL
        String baseurl = "https://www.javaguides.net/search?q=";

        // String to be encoded
        String query = "core+java+tutorial";

        URL url = new URL(baseurl + query);
        System.out.println("(1) Original URL: " + url);
        // (1) Original URL:
https://www.javaguides.net/search?q=core+java+tutorial

        String encodedStr = URLEncoder.encode(query, "UTF-8");
        url = new URL(baseurl + encodedStr);
        System.out.println("(2) Encoded URL : " + url);
        // (2) Encoded URL :
https://www.javaguides.net/search?q=core%2Bjava%2Btutorial

        // decode() method
        String decodedStr = URLDecoder.decode(encodedStr, "UTF-8");
        System.out.println("(3) Decoded URL : " + baseurl + decodedStr);
        // (3) Decoded URL :
https://www.javaguides.net/search?q=core+java+tutorial
    }
}
```

Ref Site

Percent-encoding

Class URLEncoder

Class URLDecoder

URLEncoder Class Overview

오션,, URLEncoder,, encode,, URLDecoder,, decode,, PercentEncoding

From:
http://125.132.25.164/dokuwiki/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다. - 2023.12**

Permanent link:
http://125.132.25.164/dokuwiki/doku.php?id=wiki:miscellaneous:urlencoderandurldecoder&rev=1649930596

Last update: **2022/04/14 19:03**