1 Apache2::URI - Perl API for manipulating URIs
1.1 Synopsis

```perl
use Apache2::URI ();

$hostport = $r->construct_server();
$hostport = $r->construct_server($hostname);
$hostport = $r->construct_server($hostname, $port);
$hostport = $r->construct_server($hostname, $port, $pool);

$url = $r->construct_url();
$url = $r->construct_url($rel_uri);
$url = $r->construct_url($rel_uri, $pool);

$parsed_uri = $r->parse_uri($uri);
$parsed_uri = $r->parsed_uri();
$url = join '%20', qw(one two three);
Apache2::URI::unescape_url($url);
```

1.2 Description

While APR::URI provides a generic API to dissect, adjust and put together any given URI string, Apache2::URI provides an API specific to Apache, by taking the information directly from the $r object. Therefore when manipulating the URI of the current HTTP request usually methods from both classes are used.

1.3 API

Apache2::URI provides the following functions and methods:

### 1.3.1 `construct_server`

Construct a string made of hostname and port

```perl
$hostport = $r->construct_server();
$hostport = $r->construct_server($hostname);
$hostport = $r->construct_server($hostname, $port);
$hostport = $r->construct_server($hostname, $port, $pool);
```

- **obj:** `$r` (**Apache2::RequestRec** object)
  
  The current request object

- **opt arg1:** `$hostname` (**string**)
  
  The hostname of the server.
If that argument is not passed, \$r->get_server_name is used.

- **opt arg2: $port (string)**
  
  The port the server is running on.
  
  If that argument is not passed, \$r->get_server_port is used.

- **opt arg3: $pool (APR::Pool object)**
  
  The pool to allocate the string from.
  
  If that argument is not passed, \$r->pool is used.

- **ref: $hostport (string)**
  
  The server’s hostport string

- **since: 2.0.00**

Examples:

- **Assuming that:**
  
    \$r->get_server_name == "localhost";
    \$r->get_server_port == 8001;

    The code:

    \$hostport = \$r->construct_server();

    returns a string:

    localhost:8001

- **The following code sets the values explicitly:**

    \$hostport = \$r->construct_server("my.example.com", 8888);

    and it returns a string:

    my.example.com:8888

### 1.3.2 **construct_url**

Build a fully qualified URL from the uri and information in the request rec:

```perl
$url = \$r->construct_url();
$url = \$r->construct_url($rel_uri);
$url = \$r->construct_url($rel_uri, $pool);
```
1.3.2 construct_url

- **obj: \$r (Apache2::RequestRec object)**
  
The current request object

- **opt arg1: \$rel_uri (string)**
  
The path to the requested file (it may include a concatenation of *path, query* and *fragment* components).
  
  If that argument is not passed, \$r->uri is used.

- **opt arg2: \$pool (APR::Pool object)**
  
The pool to allocate the URL from.
  
  If that argument is not passed, \$r->pool is used.

- **ret: \$url (string)**
  
  A fully qualified URL

- **since: 2.0.00**

Examples:

- Assuming that the request was
  
  http://localhost.localdomain:8529/test?args
  
  The code:
  
  ```perl
  my \$url = \$r->construct_url;
  
  returns the string:
  
  http://localhost.localdomain:8529/test
  
  notice that the query (args) component is not in the string. You need to append it manually if it’s needed.

- Assuming that the request was
  
  http://localhost.localdomain:8529/test?args
  
  The code:
  
  ```perl
  my \$rel_uri = "/foo/bar?tar";
  my \$url = \$r->construct_url(\$rel_uri);
  ```
returns the string:

http://localhost.localdomain:8529/foo/bar?tar

### 1.3.3 parse_uri

Break apart URI (affecting the current request’s uri components)

```perl
$r->parse_uri($uri);
```

- **obj:** \$r (Apache2::RequestRec object)
  
  The current request object
- **arg1:** \$uri (string)
  
  The uri to break apart
- **ret:** no return value
- **warning:**
  
  This method has several side-effects explained below
- **since:** 2.0.00

This method call has the following side-effects:

1. sets \$r->args to the rest after ‘?’ if such exists in the passed \$uri, otherwise sets it to undefined.
2. sets \$r->uri to the passed \$uri without the \$r->args part.
3. sets \$r->hostname (if not set already) using the (scheme://host:port) parts of the passed \$uri.

### 1.3.4 parsed_uri

Get the current request’s parsed uri object

```perl
my \$uri = \$r->parsed_uri();
```

- **obj:** \$r (Apache2::RequestRec object)
  
  The current request object
- **ret:** \$uri (APR::URI object)
  
  The parsed uri
• since: 2.0.00

This object is suitable for using with APR::URI::rpath

1.3.5 unescape_url

Unescape URLs

Apache2::URI::unescape_url($url);

• obj: $url ( string )

The URL to unescape

• ret: no return value

The argument $url is now unescaped

• since: 2.0.00

Example:

my $url = join '%20', qw(one two three);
Apache2::URI::unescape_url($url);

$url now contains the string:

"one two three";

1.4 See Also

APR::URI, mod_perl 2.0 documentation.

1.5 Copyright

mod_perl 2.0 and its core modules are copyrighted under The Apache Software License, Version 2.0.

1.6 Authors

The mod_perl development team and numerous contributors.
Table of Contents:

1 Apache2::URI - Perl API for manipulating URIs .......................... 1
  1.1 Synopsis ........................................................................ 2
  1.2 Description ..................................................................... 2
  1.3 API ............................................................................... 2
    1.3.1 construct_server ..................................................... 2
    1.3.2 construct_url ........................................................ 3
    1.3.3 parse_uri ............................................................... 5
    1.3.4 parsed_uri ............................................................. 5
    1.3.5 unescape_url ......................................................... 6
  1.4 See Also ....................................................................... 6
  1.5 Copyright ..................................................................... 6
  1.6 Authors ...................................................................... 6