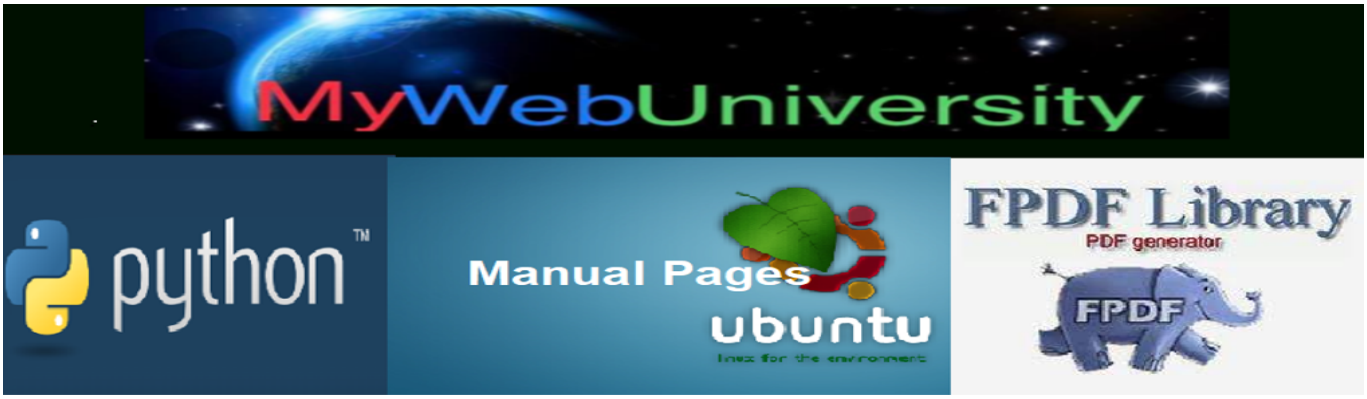

Rocky Enterprise Linux 9.2 Manual Pages on command 'URI::QueryParam.3pm'

$ man URI::QueryParam.3pm

URI::QueryParam(3pm) User Contributed Perl Documentation URI::QueryParam(3pm)

NAME

URI::QueryParam - Additional query methods for URIs

SYNOPSIS

```
use URI;
use URI::QueryParam;

$u = URI->new("", "http");
$u->query_param(foo => 1, 2, 3);
print $u->query;    # prints foo=1&foo=2&foo=3

for my $key ($u->query_param) {
    print "$key: ", join(", ", $u->query_param($key)), "\n";
}
```

DESCRIPTION

Loading the "URI::QueryParam" module adds some extra methods to URIs that support query methods. These methods provide an alternative interface to the $u->query_form data.

The query_param_* methods have deliberately been made identical to the interface of the corresponding "CGI.pm" methods.

The following additional methods are made available:

@keys = $u->query_param

@values = $u->query_param($key)

$first_value = $u->query_param($key)

$u->query_param($key, $value,...)

If $u->query_param is called with no arguments, it returns all the distinct parameter keys of the URI. In a scalar context it returns the number of distinct keys.

When a $key argument is given, the method returns the parameter values with the given key. In a scalar context, only the first parameter value is returned.

If additional arguments are given, they are used to update successive parameters with the given key. If any of the values provided are array references, then the array is dereferenced to get the actual values.

Please note that you can supply multiple values to this method, but you cannot supply multiple keys.

Do this:

```
$uri->query_param( widget_id => 1, 5, 9 );
```

Do NOT do this:

```
$uri->query_param( widget_id => 1, frobnicator_id => 99 );
```

$u->query_param_append($key, $value,...)

Adds new parameters with the given key without touching any old parameters with the same key. It can be explained as a more efficient version of:

```
$u->query_param($key,
```

```
    $u->query_param($key),
    $value,...);
```

One difference is that this expression would return the old values of $key, whereas the query_param_append() method does not.

@values = $u->query_param_delete($key)

$first_value = $u->query_param_delete($key)

Deletes all key/value pairs with the given key. The old values are returned. In a scalar context, only the first value is returned.

Using the query_param_delete() method is slightly more efficient than the equivalent:

```
  $u->query_param($key, []);
```

$hashref = $u->query_form_hash

$u->query_form_hash(\%new_form)

Returns a reference to a hash that represents the query form's key/value pairs. If a key occurs multiple times, then the hash value becomes an array reference.

Note that sequence information is lost. This means that:

```
  $u->query_form_hash($u->query_form_hash);
```

is not necessarily a no-op, as it may reorder the key/value pairs. The values returned by the query_param() method should stay the same though.

SEE ALSO

URI, CGI

COPYRIGHT