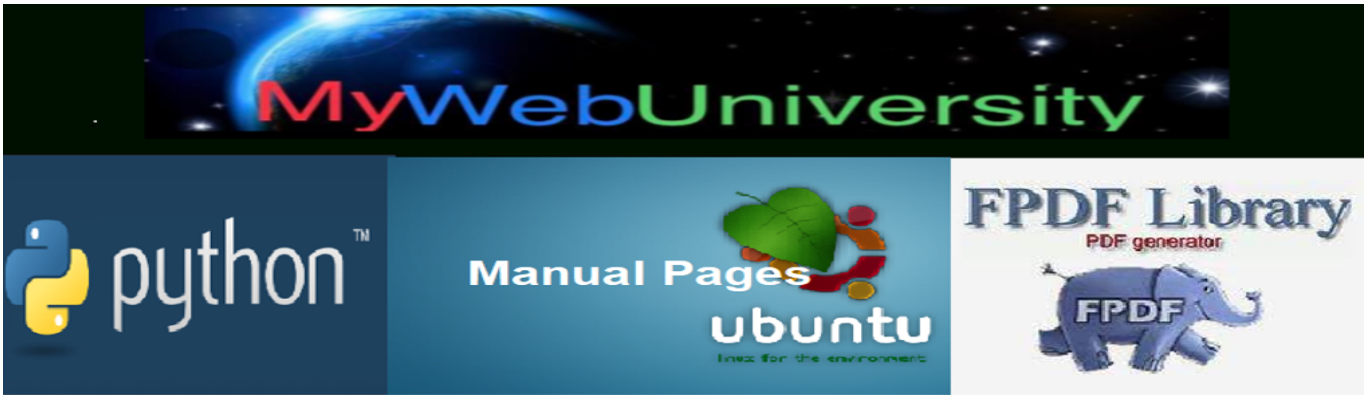

Rocky Enterprise Linux 9.2 Manual Pages on command 'APR::Util.3pm'

$ man APR::Util.3pm

libapache2-mod-perl2-2.0.12::doUseraContriblibapache2-mod-perl2-2.0.12::docs::api::APR::Util(3pm)

NAME

APR::Util - Perl API for Various APR Utilities

Synopsis

```
use APR::Util ();

$ok = password_validate($passwd, $hash);
```

Description

Various APR utilities that don't fit into any other group.

API

"APR::Util" provides the following functions and/or methods:

"password_validate"

Validate an encrypted password hash against a plain text password (with lots of restrictions and peculiarities).

```
$ok = password_validate($passwd, $hash);
```

arg1: $passwd (string)

Plain text password string

arg2: $hash (string)

Encrypted or encoded hash. See below for supported hash formats.

ret: $ok (boolean)

The password either matches or not.

since: 2.0.00

The function handles the output of the following functions (it knows to tell md5 and sha1 from the others, since they are have a special pattern recognized by apr):

? md5

generated by "apr_md5_encode()" (for which at the moment we have no perl glue, ask if you need it).

? sha1

generated by "apr_sha1_base64()" (for which at the moment we have no perl glue, ask if you need it).

and it's available only since Apache 2.0.50

? crypt

On all but the following platforms: MSWin32, beos and NetWare. Therefore you probably don't want to use that feature, unless you know that your code will never end up running on those listed platforms.

Moreover on these three platforms if that function sees that the hash is not of md5 and sha1 formats, it'll do a clear to clear text matching, always returning success,

no matter what the hashed value is.

Warning: double check that you understand what this function does and does not before using it.

Unsupported API

"APR::Socket" also provides auto-generated Perl interface for a few other methods which aren't tested at the moment and therefore their API is a subject to change. These methods will be finalized later as a need arises. If you want to rely on any of the following methods please contact the the mod_perl development mailing list so we can help each other take the steps necessary to shift the method to an officially supported API.

"filepath_name_get"

META: Autogenerated - needs to be reviewed/completed

[We have File::Spec and File::Basename for this purpose, I can't see why this api is needed]

return the final element of the pathname

```
$ret = filepath_name_get($pathname);
```

arg1: $pathname (string)

The path to get the final element of

ret: $ret (string)

the final element of the path

For example:

```
"/foo/bar/gum"    => "gum"
"/foo/bar/gum/"   => ""
"gum"             => "gum"
```

"bs\\path\\stuff" => "stuff"

since: subject to change

"password_get"

META: Autogenerated - needs to be reviewed/completed

Display a prompt and read in the password from stdin.

$ret = password_get($prompt, $pwbuf, $bufsize);

arg1: $prompt (string)

The prompt to display

arg2: $pwbuf (string)

Buffer to store the password

arg3: $bufsize (number)

The length of the password buffer.

ret: $ret (integer)

since: subject to change

See Also

mod_perl 2.0 documentation.

Copyright

Authors

The mod_perl development team and numerous contributors.