15.3. md5 — MD5 message digest algorithm¶

Deprecated since version 2.5: Use the hashlib module instead.

This module implements the interface to RSA's MD5 message digest algorithm (see also Internet **RFC 1321**). Its use is quite straightforward: use new() to create an md5 object. You can now feed this object with arbitrary strings using the update() method, and at any point you can ask it for the *digest* (a strong kind of 128-bit checksum, a.k.a. "fingerprint") of the concatenation of the strings fed to it so far using the digest() method.

For example, to obtain the digest of the string 'Nobody inspects the spammish repetition':

```
>>> import md5
>>> m = md5.new()
>>> m.update("Nobody inspects")
>>> m.update(" the spammish repetition")
>>> m.digest()
'\xbbd\x9c\x83\xdd\x1e\xa5\xc9\xd9\xde\xc9\xa1\x8d\xf0\xff\xe9'
```

More condensed:

```
>>> md5.new("Nobody inspects the spammish repetition").digest()
'\xbbd\x9c\x83\xdd\x1e\xa5\xc9\xd9\xde\xc9\xa1\x8d\xf0\xff\xe9'
```

The following values are provided as constants in the module and as attributes of the md5 objects returned by new():

md5.digest_size¶

The size of the resulting digest in bytes. This is always 16.

The md5 module provides the following functions:

md5.new([*arg*])¶

Return a new md5 object. If *arg* is present, the method call update(arg) is made.

md5.md5([*arg*])¶

For backward compatibility reasons, this is an alternative name for the new() function.

An md5 object has the following methods:

md5.update(*arg*)¶

Update the md5 object with the string *arg*. Repeated calls are equivalent to a single call with the concatenation of all the arguments: m.update(a); m.update(b) is equivalent to m.update(a+b).

md5.digest()¶

Return the digest of the strings passed to the update() method so far. This is a 16-byte string which may contain non-ASCII characters, including null bytes.

md5.hexdigest()¶

Like digest() except the digest is returned as a string of length 32, containing only hexadecimal digits. This may be used to exchange the value safely in email or other non-binary environments.

md5.copy()¶

Return a copy ("clone") of the md5 object. This can be used to efficiently compute the digests of strings that share a common initial substring.

See also

Module sha

Similar module implementing the Secure Hash Algorithm (SHA). The SHA algorithm is considered a more secure hash.

Previous topic

15.2. hmac — Keyed-Hashing for Message Authentication

Next topic

15.4. sha — SHA-1 message digest algorithm

This Page

- Show Source

Navigation

- index
- modules |
- next |
- previous |
- Python v2.6.4 documentation »
- The Python Standard Library »
- 15. Cryptographic Services »

The Python Software Foundation is a non-profit corporation. Please donate.
Last updated on Feb 26, 2010. Created using Sphinx 0.6.3.