

5 Using IP Addresses with other modules

Other modules that use IP addresses (such as `socket`) usually won't accept objects from this module directly. Instead, they must be coerced to an integer or string that the other module will accept:

```
>>> addr4 = ipaddress.ip_address('192.0.2.1')
>>> str(addr4)
'192.0.2.1'
>>> int(addr4)
3221225985
```

6 Getting more detail when instance creation fails

When creating address/network/interface objects using the version-agnostic factory functions, any errors will be reported as `ValueError` with a generic error message that simply says the passed in value was not recognized as an object of that type. The lack of a specific error is because it's necessary to know whether the value is *supposed* to be IPv4 or IPv6 in order to provide more detail on why it has been rejected.

To support use cases where it is useful to have access to this additional detail, the individual class constructors actually raise the `ValueError` subclasses `ipaddress.AddressValueError` and `ipaddress.NetmaskValueError` to indicate exactly which part of the definition failed to parse correctly.

The error messages are significantly more detailed when using the class constructors directly. For example:

```
>>> ipaddress.ip_address("192.168.0.256")
Traceback (most recent call last):
...
ValueError: '192.168.0.256' does not appear to be an IPv4 or IPv6 address
>>> ipaddress.IPv4Address("192.168.0.256")
Traceback (most recent call last):
...
ipaddress.AddressValueError: Octet 256 (> 255) not permitted in '192.168.0.256'

>>> ipaddress.ip_network("192.168.0.1/64")
Traceback (most recent call last):
...
ValueError: '192.168.0.1/64' does not appear to be an IPv4 or IPv6 network
>>> ipaddress.IPv4Network("192.168.0.1/64")
Traceback (most recent call last):
...
ipaddress.NetmaskValueError: '64' is not a valid netmask
```

However, both of the module specific exceptions have `ValueError` as their parent class, so if you're not concerned with the particular type of error, you can still write code like the following:

```
try:
    network = ipaddress.IPv4Network(address)
except ValueError:
    print('address/netmask is invalid for IPv4:', address)
```