The Internet

The Internet is a pervasive aspect of modern computing. Even small, single-use scripts frequently interact with remote services to send or receive data. Python’s rich set of tools for working with web protocols makes it well suited for programming web-based applications, either as a client or a server.

The `urllib.parse` module manipulates URL strings, splitting and combining their components, and is useful in clients and servers.

The `urllib.request` module implements an API for retrieving content remotely.

HTTP POST requests are usually “form encoded” with `urllib`. Binary data sent through a POST should be encoded with `base64` first, to comply with the message format standard.

Well-behaved clients that access many sites as a spider or crawler should use `urllib.robotparser` to ensure they have permission before placing a heavy load on the remote server.

To create a custom web server with Python, without requiring any external frameworks, use `http.server` as a starting point. It handles the HTTP protocol, so the only customization needed is the application code for responding to the incoming requests.

Session state in the server can be managed through cookies created and parsed by the `http.cookies` module. Full support for expiration, path, domain, and other cookie settings makes it easy to configure the session.

The `uuid` module is used for generating identifiers for resources that need unique values. UUIDs are good for automatically generating Uniform Resource Name (URN) values, where the name of the resource needs to be unique but does not need to convey any meaning.

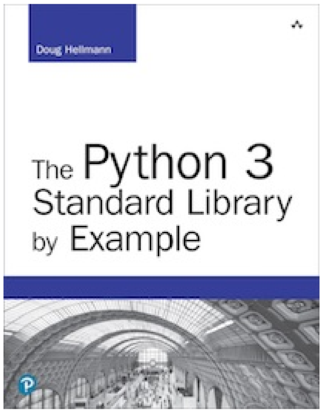
Python’s standard library includes support for two web-based remote procedure call mechanisms. The JavaScript Object Notation (JSON) encoding scheme used in AJAX communication and REST API is implemented in `json`. It works equally well in the client or the server. Complete XML-RPC client and server libraries are also included in `xmlrpc.client` and `xmlrpc.server` respectively.

socketserver — Creating Network Servers urllib.parse — Split URLs into Components

This page was last updated 2016-12-29.

Navigation

socketserver — Creating Network Servers
urllib.parse — Split URLs into Components



The output from all the example programs from PyMOTW-3 has been generated with Python 3.7.1, unless otherwise noted. Some of the features described here may not be available in earlier versions of Python.

Looking for examples for Python 2?

This Site

- Module Index
- Index

Other Writing

- Blog
- The Python Standard Library By Example