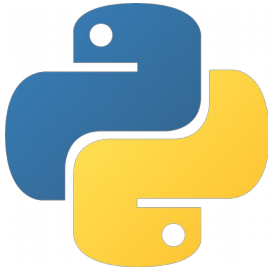
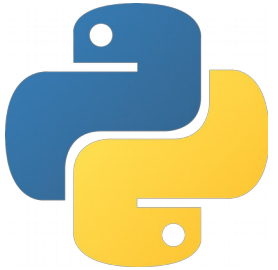
Python Netzwerkkommunikation



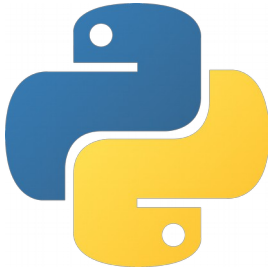
- Netzwerkkommunikation
  - URLs
    - Eine URL (für Uniform Resource Locator) spezifiziert eine Ressource, beispielsweise im Internet, über ihren Ort und das zum Zugriff zu verwendende Protokoll
    - Das Paket urllib bietet eine komfortable Schnittstelle zum Umgang mit Ressourcen im Internet

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - URLs
  - Zugriff auf Ressourcen im Internet – urllib.request
    - Das Modul urllib.request bietet eine komfortable Schnittstelle, um auf Dateien im Internet zuzugreifen
    - Die zentrale Funktion dieser Bibliothek ist urlopen, die der Funktion open ähnelt, bis auf die Tatsache, dass statt eines Dateinamens eine URL übergeben wird
    - Außerdem können auf dem resultierenden Dateiobjekt aus naheliegendem Grund keine Schreiboperationen durchgeführt werden

Python Netzwerkkommunikation



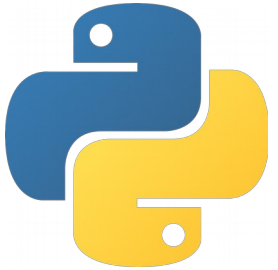
- Netzwerkkommunikation
  - URLs
  - Zugriff auf Ressourcen im Internet – urllib.request

```
import urllib.request
```

    - Die Funktion urlopen greift auf die durch url adressierte Netzwerkressource zu und gibt ein geöffnetes Dateiobjekt auf dieser Ressource zurück

    - Wenn bei der URL kein Protokoll wie beispielsweise http:// oder ftp:// angegeben wurde, wird angenommen, dass die URL auf eine Ressource der lokalen Festplatte verweist
      - Für Zugriffe auf die lokale Festplatte können Sie außerdem das Protokoll file:// angeben

Python Netzwerkkommunikation

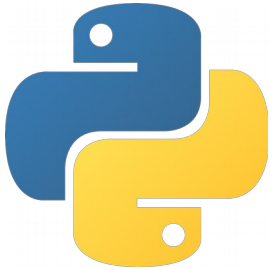


- Netzwerkkommunikation
  - URLs
  - Zugriff auf Ressourcen im Internet – urllib.request

Die folgenden Tabellen zeigen die verfügbaren Methoden des zurückgegebenen Objekts (urlopen) mit einer kurzen Beschreibung

Methode	Beschreibung
read([size])	Liest *size* Byte aus der Ressource aus. Wenn *size* nicht angegeben wurde, wird der komplette Inhalt ausgelesen. Die gelesenen Daten werden als String zurückgegeben.
readline([size])	Liest eine Zeile aus der Ressource aus. Wenn *size* angegeben wurde, werden maximal *size* Byte gelesen. Die gelesenen Daten werden als String zurückgegeben.

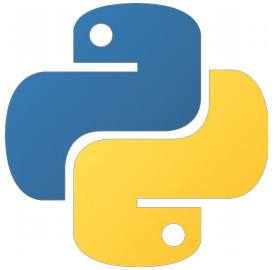
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - URLs
  - Zugriff auf Ressourcen im Internet – urllib.request

Methode	Beschreibung
readlines([sizehint])	Liest die Ressource zeilenweise aus und gibt sie in Form einer Liste von Strings zurück. Wird *sizehint* angegeben, so werden Zeilen nur so lange eingelesen, bis die Gesamtgröße der gelesenen Zeilen *sizehint* überschreitet.
fileno()	Gibt den Dateideskriptor der geöffneten Ressource als ganze Zahl zurück.
close()	Schließt das geöffnete Objekt. Nach Aufruf dieser Methode sind keine weiteren Operationen mehr möglich.
info()	Gibt ein dictionary-ähnliches Objekt zurück, das Metainformationen der heruntergeladenen Seite enthält. Im Anschluss an diese Tabelle werden wir uns eingehend mit der von *info* zurückgegebenen Instanz beschäftigen.
geturl()	Gibt einen String mit der URL der Ressource zurück.

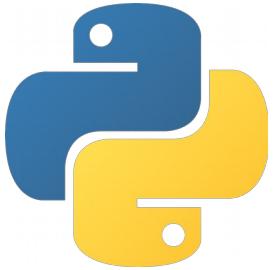
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - URLs
  - Zugriff auf Ressourcen im Internet – urllib.request
    - info()

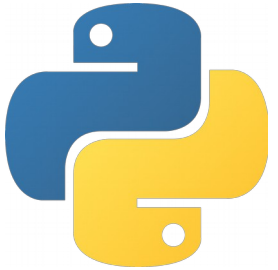
```
>>> f = urllib.request.urlopen("http://www.galileo-press.de")
>>> d = f.info()
>>> d
<http.client.HTTPMessage object at 0xb7689e8c>
>>> d.keys()
['Date', 'Server', 'Content-Length', 'Content-Type',
'Cache-Control', 'Expires', 'Vary', 'Connection']
```

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - URLs
  - Zugriff auf Ressourcen im Internet – urllib.request
    - urllib.request.urlretrieve(url[, filename[, reporthook[, data]]])
      - Diese Funktion macht den Inhalt der Ressource, auf die die URL url verweist, unter einem lokalen Dateinamen verfügbar
      - Dazu wird der Inhalt der Ressource heruntergeladen oder kopiert, sofern dies notwendig ist
      - Die Funktion urlretrieve gibt ein Tupel mit zwei Elementen zurück: dem Dateinamen der lokalen Datei und dem Rückgabewert der info-Methode des dateiähnlichen Objekt

Python Netzwerkkommunikation

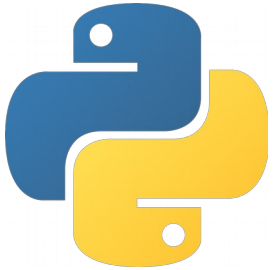


- Netzwerkkommunikation
  - URLs
  - Zugriff auf Ressourcen im Internet – urllib.request
    - urllib.request.urlretrieve(url[, filename[, reporthook[, data]]])

      >>> urllib.request.urlretrieve("http://www.google.de")
      ('/tmp/tmpYger_7',
      <http.client.HTTPMessage object at 0xb768a30c>)

      - Mithilfe des dritten Parameters lässt sich also eine Statusanzeige des Downloads realisieren, wie auf der folgenden Folien aufgezeigt

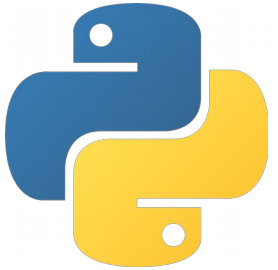
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - URLs
  - Zugriff auf Ressourcen im Internet – urllib.request

```
>>> def f(blocks, blocksize, size):
…             print("Status: {}
                        %".format(int(blocksize*blocks*100/size)))
...
>>> res = urllib.request.urlretrieve(url, "datei.html", f)
Status: 0%
Status: 14%
Status: 29%
Status: 59%
Status: 88%
Status: 103%
>>> res
('datei.html', <http.client.HTTPMessage object at
0xd8cc50>)
```

Python Netzwerkkommunikation

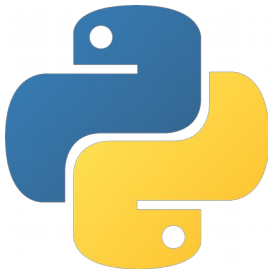


- Netzwerkkommunikation
  - URLs
  - Verarbeiten einer URL – urllib.parse

    - Eine URL in ihre Bestandteile zu zerlegen oder diese Bestandteile wieder zu einer gültigen URL zusammenzufügen

    - Auf den folgenden Folien die Tabelle mit dem Inhalt des Moduls zusammengefasst

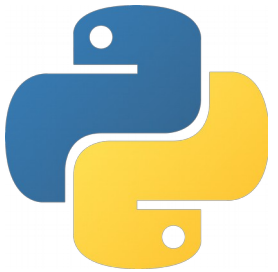
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - URLs
  - Verarbeiten einer URL – urllib.parse

Funktion	Beschreibung
urlparse(urlstring[, scheme[, allow_fragments]])	Zerlegt die URL *urlstring* in ihre Bestandteile.
parse_qs(qs[, keep_blank_values[, strict_parsing[, encoding[, errors]]]])	Zerlegt den Query-String *qs* einer URL in seine Bestandteile. Das Ergebnis wird als Dictionary zurückgegeben.
parse_qsl(qsl[, keep_blank_values[, strict_parsing[, encoding[, errors]]]])	Zerlegt den Query-String *qs* einer URL in seine Bestandteile. Das Ergebnis wird als Liste von Schlüssel-Wert-Paaren zurückgegeben.
urlunparse(parts)	Das Gegenstück zu *urlparse*: Erzeugt aus einem Tupel mit den Bestandteilen einer URL einen URL-String.
urlsplit(urlstring[, scheme[, allow_fragments]])	Wie *urlparse*, berücksichtigt aber nicht den Parameterteil einer URL.

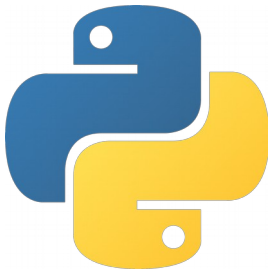
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - URLs
  - Verarbeiten einer URL – urllib.parse

Funktion	Beschreibung
urlunsplit(parts)	das Gegenstück zu *urlsplit*
urljoin(base, url[, allow_fragments])	Kombiniert die Basis-URL *base* und die relative URL *url* zu einer absoluten Pfadangabe.
urldefrag(url)	Spaltet den Anker von der URL *url* ab, sofern dieser vorhanden ist.
quote(string[, safe[, encoding[, errors]]])	Ersetzt Sonderzeichen im String *string* durch Escape-Sequenzen, wie sie in URLs erlaubt sind.
quote_plus(string[, safe[, encoding[, errors]]])	Funktioniert wie *quote* mit dem Unterschied, dass ein Leerzeichen in *string* durch ein + ersetzt wird. Dies ist insbesondere im Zusammenhang mit HTML-Formulardaten interessant.

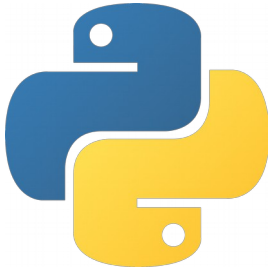
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - URLs
  - Verarbeiten einer URL – urllib.parse

Funktion	Beschreibung
quote_from_bytes(bytes[, safe])	Funktioniert wie *quote* mit dem Unterschied, dass der betrachtete Text als bytes-Instanz und nicht als String übergeben wird.
unquote(string[, encoding[, errors]])	das Gegenstück zu *quote*
unquote_plus(string[, encoding[, errors]])	das Gegenstück zu *quote_plus*
unquote_to_bytes(string)	das Gegenstück zu *quote_from_bytes*
urlencode(query[, doseq[, safe[, encoding[, errors]]]])	Erzeugt aus dem Dictionary *query* einen Query-String, der in einer URL verwendet werden kann.

Python Netzwerkkommunikation

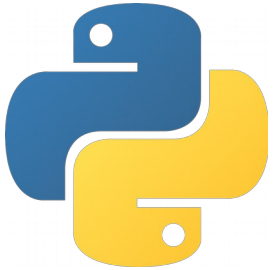


- Netzwerkkommunikation
  - URLs
  - Verarbeiten einer URL – urllib.parse

    - Die folgende Folien zeigen die Tabelle, die alle Attribute des Rückgabewertes der Funktion urlparse auflistet

Attribut	Index	Beschreibung
scheme	0	das Protokoll der URL, beispielsweise `http` oder `file`
netloc	1	Die *Network Location* besteht üblicherweise aus einem Domainnamen mit Subdomain und TLD, beispielsweise `www.galileo-press.de`. Optional können auch Benutzername, Passwort und Portnummer in *netloc* enthalten sein.

Python Netzwerkkommunikation

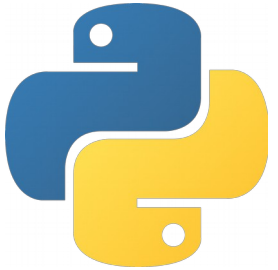


- Netzwerkkommunikation
  - URLs
  - Verarbeiten einer URL – urllib.parse

    - Die folgende Folien zeigen die Tabelle, die alle Attribute des Rückgabewertes der Funktion urlparse auflistet

Attribut	Index	Beschreibung
path	2	eine Pfadangabe, die einen Unterordner der Network Location kennzeichnet
params	3	Parameter für das letzte Element des Pfades
query	4	Über den *Query-String* können zusätzliche Informationen an ein serverseitiges Script übertragen werden.
fragment	5	Das Fragment, auch *Anker* genannt. Ein geläufiges Beispiel für einen Anker ist eine Sprungmarke innerhalb einer HTML-Datei.

Python Netzwerkkommunikation

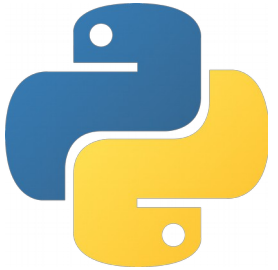


- Netzwerkkommunikation
  - URLs
  - Verarbeiten einer URL – urllib.parse

    - Die folgende Folien zeigen die Tabelle, die alle Attribute des Rückgabewertes der Funktion urlparse auflistet

Attribut	Index	Beschreibung
username	–	der in der URL angegebene Benutzername, sofern vorhanden
password	–	das in der URL angegebene Passwort, sofern vorhanden.
hostname	–	der Domainname der URL, beispielsweise `www.galileo-press.de`
port	–	die in der URL angegebene Portnummer, sofern vorhanden

Python Netzwerkkommunikation

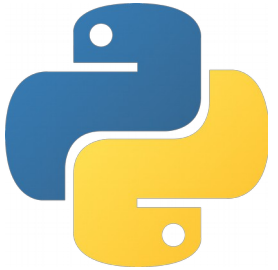


- Netzwerkkommunikation
  - URLs
  - Verarbeiten einer URL – urllib.parse

    - Aufgabe
    - In selbständiger Arbeit, urllib.parse Funktion urlparse ausprobieren, damit spielen

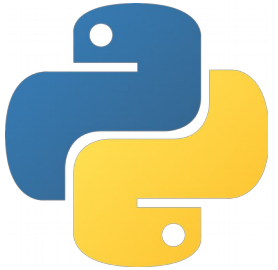
    - Einen kleinen HTML Seiten Parser schreiben, der den Reinen Text einer Seite ausgibt, ohne die Symantik zu zerstören, d.H. Eine Überschrift bleibt eine Überschrift usw.

Python Netzwerkkommunikation



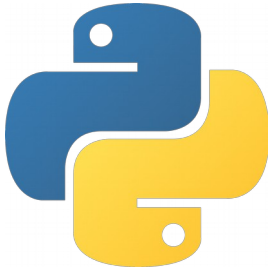
- Netzwerkkommunikation
  - URLs
  - Verarbeiten einer URL – urllib.parse
    - Aufgabe
    - Folgende Funktionen anschauen und ausprobieren
    - urllib.parse.parse_qs(qs[, keep_blank_values[, strict_parsing[, encoding[, errors]]]]),
    - urllib.parse.parse_qsl(qs[, keep_blank_values[, strict_parsing[, encoding[, errors]]]])
    - urllib.parse.urlunparse(parts)
    - urllib.parse.urlsplit(urlstring[, scheme[, allow_fragments]])
    - urllib.parse.urljoin(base, url[, allow_fragments])

Python Netzwerkkommunikation



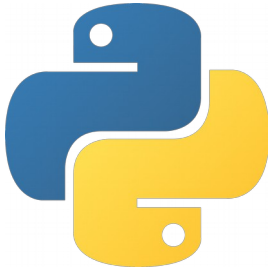
- Netzwerkkommunikation
  - E-Mail
    - Mit einem E-Mail-Server zu kommunizieren
    - E-Mails vom Server abzuholen bzw. E-Mails über den Server zu versenden
    - das Modul smtplib
      import smtplib
    - Das Modul smtplib enthält im Wesentlichen nur eine Klasse namens SMTP
    - Über diese Klasse läuft, nachdem sie instanziiert wurde, alle weitere Kommunikation mit dem Server

Python Netzwerkkommunikation



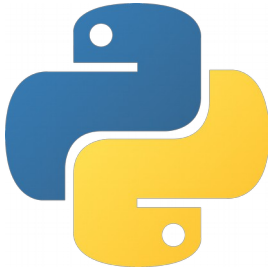
- Netzwerkkommunikation
  - E-Mail
    - SMTP([host[, port[, local_hostname[, timeout]]]])
      - Hiermit wird eine Instanz der Klasse SMTP erzeugt
      - Optional können hier bereits die Verbindungsdaten zum SMTP-Server übergeben werden
      - Beachten Sie, dass Sie den Port nur explizit anzugeben brauchen, wenn er sich vom SMTP-Standardport 25 unterscheidet

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - SMTP([host[, port[, local_hostname[, timeout]]]])
      - Als dritter Parameter kann der Domainname des lokalen Hosts übergeben werden
      - Dieser wird dem SMTP-Server als Identifikation im ersten gesendeten Kommando übermittelt
      - Wenn der Parameter local_hostname nicht angegeben wird, wird versucht, den lokalen Hostnamen automatisch zu ermitteln
      - Für den vierten Parameter können Sie einen speziellen Timeout-Wert in Sekunden übergeben, der bei der Verbindung zum SMTP-Server berücksichtigt wird

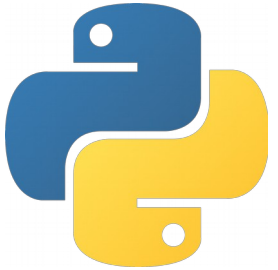
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - s.connect([host[, port]])
      - Diese Methode verbindet zum SMTP-Server host mit der Portnummer port
      - Sie sollte nicht aufgerufen werden, wenn bei der Instanziierung der SMTP -Klasse bereits Verbindungsdaten übergeben wurden
      - Wenn keine Verbindung zum SMTP-Server aufgebaut werden kann, wird eine Exception geworfen

        >>> s.connect("smtp.beispiel.de")
        (220, 'Die Botschaft des Servers')

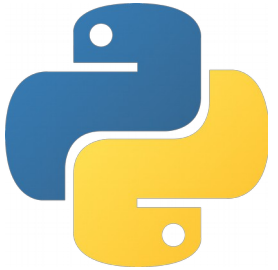
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - s.login(user, password)
      - Diese Methode ermöglicht es, sich beim SMTP-Server mit dem Benutzernamen user und dem Passwort password einzuloggen, sofern der Server dies verlangt

        >>> s.login("Benutzername", "Passwort")
        (235, '2.0.0 Authentication successful')

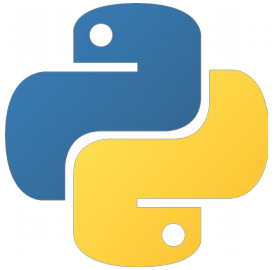
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - s.login(user, password)
      - Im Fehlerfall wird eine der folgenden Exceptions geworfen

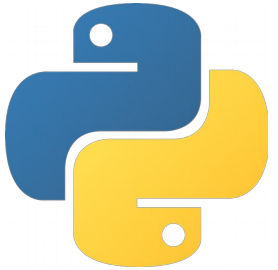
Exception	Beschreibung
SMTPHeloError	Der SMTP-Server hat nicht oder nicht richtig auf das Begrüßungskommando HELO geantwortet.
SMTPAuthenticationError	Die angegebene Benutzername-Passwort-Kombination wurde vom SMTP-Server nicht akzeptiert.
SMTPError	Es wurde keine Möglichkeit gefunden, eine Authentifizierung bei diesem SMTP-Server durchzuführen.

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - s.sendmail(from_addr, to_addrs, msg[, mail_options[, rcpt_options]])
      - Durch Aufruf der Methode sendmail wird eine E-Mail über den SMTP-Server versendet
      - Beachten Sie, dass die SMTP-Instanz dafür an einem SMTP-Server angemeldet und zumeist auch authentifiziert sein muss
      - Die ersten beiden Parameter enthalten die E-Mail-Adressen des Absenders (from_addr) bzw. eine Liste der E-Mail-Adressen der Empfänger (to_addr)

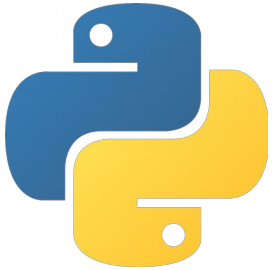
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - s.sendmail(from_addr, to_addrs, msg[, mail_options[, rcpt_options]])
      - Als E-Mail-Adresse wird dabei ein String des folgenden Formats bezeichnet
      - Alternativ kann auch nur die E-Mail-Adresse im String stehen

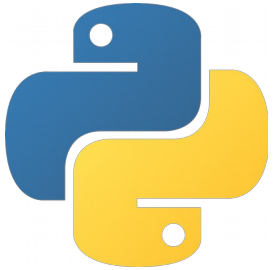
        Vorname Nachname <em@il.addr>

Python Netzwerkkommunikation



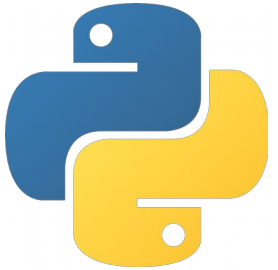
- Netzwerkkommunikation
  - E-Mail
    - s.sendmail(from_addr, to_addrs, msg[, mail_options[, rcpt_options]])
      - Als dritten Parameter, msg, übergeben Sie den Text der E-Mail
      - Hier werden auch weitere Angaben wie beispielsweise der Betreff der E-Mail definiert
      - Wie so etwas genau aussieht und welche Möglichkeiten Python bietet, diesen Header komfortabel zu erzeugen, erfahren Sie später

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - s.sendmail(from_addr, to_addrs, msg[, mail_options[, rcpt_options]])
      - Die Methode sendmail gibt stets ein Dictionary zurück, in dem alle Empfänger, die vom SMTP-Server zurückgewiesen wurden, als Schlüssel enthalten sind und der jeweilige Error-Code mit Fehlerbezeichnung als Wert aufgeführt ist
      - Wenn alle Empfänger die E-Mail bekommen haben, ist das zurückgegebene Dictionary leer

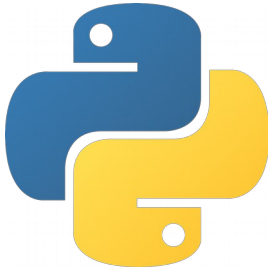
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - s.sendmail(from_addr, to_addrs, msg[, mail_options[, rcpt_options]])
      - Im Fehlerfall wird eine der folgenden Exceptions geworfen

Exception	Beschreibung
SMTPRecipientsRefused	Alle Empfänger wurden vom SMTP-Server zurückgewiesen. Das heißt, dass die E-Mail an niemanden verschickt wurde. Als Attribut `recipients` enthält die Exception ein Dictionary, wie es von *sendmail* im Erfolgsfall zurückgegeben wird.
SMTPSenderRefused	Der angegebene Absender wurde vom SMTP-Server zurückgewiesen.
SMTPDataError	Der Server hat mit einem unerwarteten Fehler geantwortet.
SMTPHeloError	Der SMTP-Server hat nicht oder nicht richtig auf das Begrüßungskommando `HELO` geantwortet.

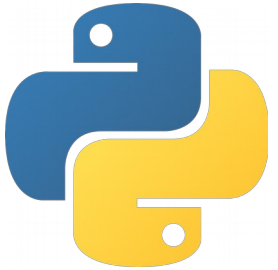
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - Beispiel

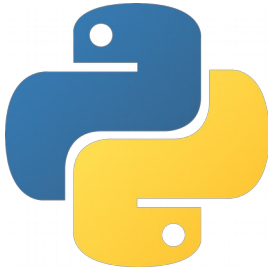
```
>>> smtp = smtplib.SMTP("smtp.hostname.de")
>>> smtp.login("Benutzername", "Passwort")
(235, '2.0.0 Authentication successful')
>>> smtp.sendmail(
…                       "Peter Kaiser <p@penguin-p.de>",
…                       "Johannes Ernesti <je@lpe-media.de>",
…                       "Dies ist der Text")
{}
>>> smtp.sendmail(
…                       "Peter Kaiser <p@penguin-p.de>",
…                       ["je@lpe-media.de", "p@penguin-p.de"]
…                       Dies ist der Text")
{}
>>> smtp.quit()
```

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
  - POP3 → poplib
    - Dieses Modul implementiert das POP3-Protokoll (Post Office Protocol Version 3)
    - Bei POP3 handelt es sich um ein Protokoll, um auf einen POP3-Server zuzugreifen und dort gespeicherte E-Mails einzusehen und abzuholen
    - Das POP3-Protokoll steht damit in Konkurrenz zu IMAP4
    - Die folgende Tabelle listet die wichtigsten POP3-Kommandos mit ihrer Bedeutung auf, nächste Folie

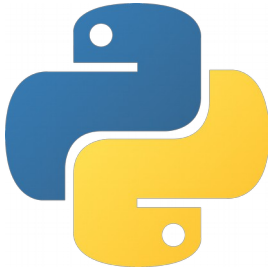
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
  - POP3 → poplib
    - Wichtigsten POP3-Kommandos mit ihrer Bedeutung

Befehl	Beschreibung
USER	Überträgt den Benutzernamen zur Authentifizierung auf dem Server.
PASS	Überträgt das Passwort zur Authentifizierung auf dem Server.
STAT	Liefert den Status des Posteingangs, beispielsweise die Anzahl der neu eingegangenen E-Mails.
LIST	Liefert Informationen zu einer bestimmten E-Mail des Posteingangs.
RETR	Überträgt eine bestimmte E-Mail.
DELE	Löscht eine bestimmte E-Mail.
RSET	Widerruft alle anstehenden Löschvorgänge.[11]
QUIT	Beendet die POP3-Sitzung.

Python Netzwerkkommunikation

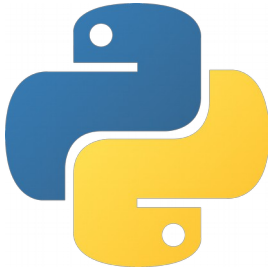


- Netzwerkkommunikation
  - E-Mail
  - POP3(host[, port[, timeout]]) → Siehe SMTP

```
>>> import poplib
>>> pop = poplib.POP3("pop.beispiel.de")
```

  - pop.pass_(password)
    - Diese Methode übermittelt das Passwort password an den POP3-Server
    - Nachdem das Passwort vom Server akzeptiert worden ist, darf auf den Posteingang zugegriffen werden
    - Dieser ist bis zum Aufruf von quit für andere Login-Versuche gesperrt

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
  - POP3(host[, port[, timeout]]) → Siehe SMTP

pop.pass_(password)

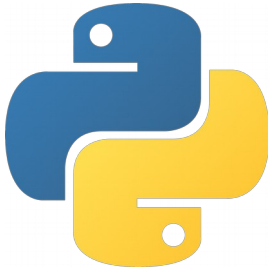
```
>>> pop.pass_("Passwort")
b'+OK logged in.'
```

- Im Falle einer fehlgeschlagenen Authentifizierung wird eine poplib.error_proto-Exception geworfen

- Hinweis
- Der bevorzugte Name pass für diese Methode ist in Python bereits mit einem Schlüsselwort belegt

- In solchen Fällen wird an den belegten Namen häufig einUnterstrich angehängt

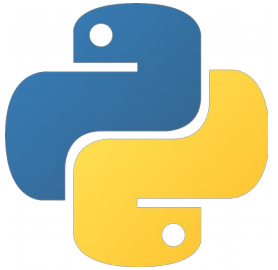
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
  - POP3(host[, port[, timeout]]) → Siehe SMTP
    - pop.stat()
      - Diese Methode gibt den Status des Posteingangs zurück
      - Das Ergebnis ist ein Tupel mit zwei ganzen Zahlen: der Anzahl der enthaltenen Nachrichten und der Größe des Posteingangs in Byte

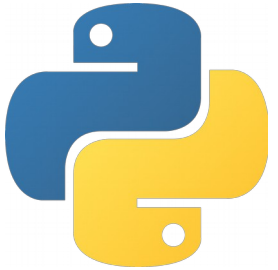
        >>> pop.stat()
        (1,623)

Python Netzwerkkommunikation



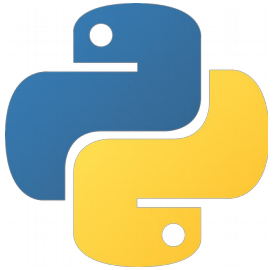
- Netzwerkkommunikation
  - E-Mail
  - POP3(host[, port[, timeout]]) → Siehe SMTP
    - pop.list([which])
      - Diese Methode gibt eine Liste der im Posteingang liegenden Mails zurück
      - Der Rückgabewert dieser Methode ist ein Tupel der folgenden Form
        (antwort, [b"mailID laenge", ...], datlen)
      - Dabei enthält das Tupel als erstes Element den Antwortstring des Servers und als zweites Element eine Liste von bytes-Strings, die je für eine E-Mail des Posteingangs stehen

Python Netzwerkkommunikation



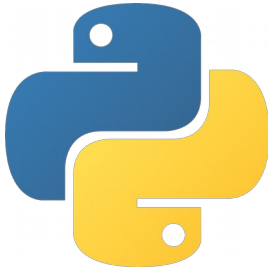
- Netzwerkkommunikation
  - E-Mail
  - POP3(host[, port[, timeout]]) → Siehe SMTP
    - pop.list([which])
      - Der String enthält zwei Angaben: Die Angabe mailID ist die laufende Nummer der Mail, eine Art Index, und laenge ist die Gesamtgröße der Mail in Byte
      - In Bezug auf den Index sollten Sie beachten, dass alle E-Mails auf dem Server fortlaufend von 1 an indiziert werden und nicht, wie beispielsweise bei einer Python-Liste, mit 0 beginnend

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
  - POP3(host[, port[, timeout]]) → Siehe SMTP
    - pop.retr(which)
      - Diese Methode greift auf die Mail mit der laufenden Nummer which zu und gibt ihren Inhalt in Form des folgenden Tupels zurück
        (antwort, zeilen, laenge)
      - Das erste Element des Tupels entspricht dem Antwortstring des Servers
      - An zweiter Stelle steht eine Liste von bytes-Strings, die je eine Zeile der E-Mail inklusive des E-Mail-Headers enthalten
      - Das letzte Element des Tupels ist die Größe der E-Mail in Byte

Python Netzwerkkommunikation

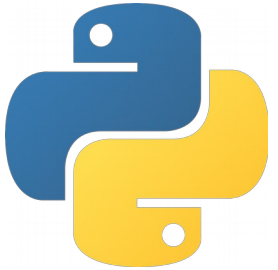


- Netzwerkkommunikation
  - E-Mail
  - POP3(host[, port[, timeout]]) → Siehe SMTP
    - pop.retr(which)

      >>> pop.retr(1)
      (b'+OK 623 octets follow.', [...], 623)

      - Im Antwortstring ist von »623 octets« die Rede
      - Mit Octets (dt. »Achtergruppen«) sind Bytes gemeint
      - Anstelle von [...] stünde eine Liste von Strings, die die Zeilen der vollständigen E-Mail enthält

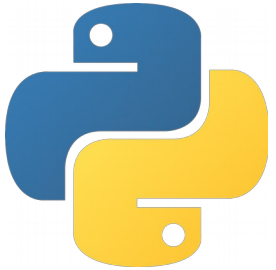
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
  - POP3(host[, port[, timeout]]) → Siehe SMTP
    - pop.dele(which)
      - Diese Methode löscht die Mail mit der laufenden Nummer which vom POP3-Server
      - Beachten Sie, dass die meisten Server solche Befehle puffern und erst nach Aufruf der Methode quit tatsächlich ausführen

        >>> pop.dele(1)
        b'+OK Deleted.'

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - Beispiel

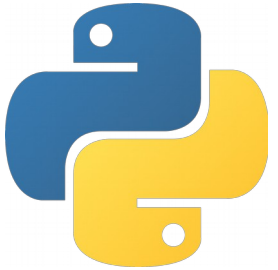
```
import poplib

pop = poplib.POP3("pop.hostname.de")
pop.user("benutzername")
pop.pass_("passwort")

for i in range(1, pop.stat()[0]+1):
   for zeile in pop.retr(i)[1]:
      print(zeile)
   print("***")

pop.quit()
```

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail

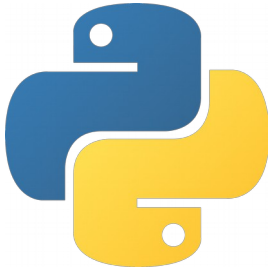
    - Aufgabe
      Schreiben Sie einen Konsolenbasierenden E-Mail Klienten für POP3 Konten

    - Erarbeiten Sie sich den Umgang mit IMAP4

    - Erweitern Sie Ihr Projekt um IMAP4 Konten

    - Menüführung
      - Hinzufügen, ändern, löschen, speichern von Kontodaten
      - Speichern der Userinformationen als Objekt (pickle)
      - Abfragen von E-Mails und Ausgabe einer Übersicht

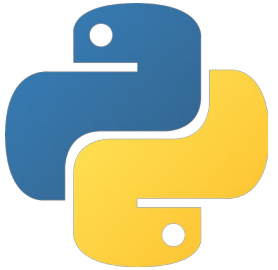
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - Methoden IMAP4 Instanz

Methode	Beschreibung
login(user, password)	Sendet Benutzername und Passwort an den verbundenen IMAP4-Server.
logout()	Beendet die Verbindung zum IMAP4-Server.
select([mailbox[, readonly]])	Wählt eine Mailbox aus, um weitere Operationen auf dieser durchführen zu können.
close()	Schließt die momentan ausgewählte Mailbox.
list([directory[, pattern]])	Gibt die Namen aller Mailboxen zurück, die sich im Ordner *directory* befinden und auf *pattern* passen.

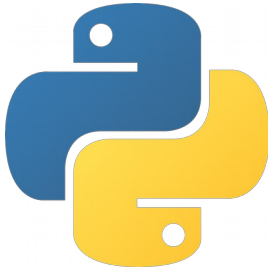
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - Methoden IMAP4 Instanz

Methode	Beschreibung
fetch(message_set, message_parts)	Lädt Teile der E-Mails vom Server herunter.
create(mailbox)	Erstellt eine neue Mailbox namens *mailbox*.
delete(mailbox)	Löscht die Mailbox *mailbox*.
rename(old_mailbox, new_mailbox)	Benennt die Mailbox *old_mailbox* in *new_mailbox* um.

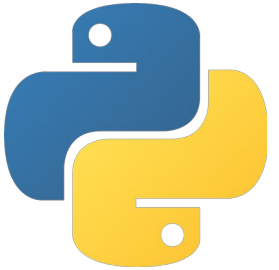
Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - Methoden IMAP4 Instanz

Methode	Beschreibung
copy(message_set, new_mailbox)	Kopiert die E-Mails *message_set* in die Mail-box *new_mailbox*. Der Parameter *message_set* ist wie bei *fetch* zu verstehen.
search(charset, criterion[, ...])	Sucht innerhalb der ausgewählten Mailbox nach E-Mails, die auf bestimmte Kriterien passen.
store(message_set, command, flag_list)	Verändert die Eigenschaften der E-Mails *message_set*.

Python Netzwerkkommunikation



- Netzwerkkommunikation
  - E-Mail
    - Methoden IMAP4 Instanz
    - Instanz erzeugen
      >>> import imaplib
      >>> im = imaplib.IMAP4("imap.beispiel.de")