Python: inviare un'e-mail con allegati

In questo articolo vedremo come inviare un'e-mail con allegati in Python.

Si tratta di creare una sessione SMTP con il server remoto dopo aver creato il messaggio da inviare. La sessione prevede l'uso di SSL con cui verranno inviate le credenziali di accesso. Se il messaggio contiene un riferimento ad un file da allegare, tale file va letto in modalità binaria e codificato in Base64 per poter essere inviato con il corpo dell'e-mail.

La funzione che andremo ad implementare accetta come parametri due dizionari: `smtp_config` contiene i dati per la connessione al server SMTP e `mail_data` i dati dell'e-mail (mittente, destinatario, oggetto, testo semplice, testo in HTML e eventuale file allegato).

```
import smtplib
import ssl
from email import encoders
from email.mime.base import MIMEBase
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart


def send_email(smtp_config, mail_data):
    if not isinstance(smtp_config, dict) or not
isinstance(mail_data, dict):
        return False

    smtp_port = smtp_config.get('port', 465)
    smtp_host = smtp_config.get('host', 'localhost')
```

```
    smtp_user = smtp_config.get('user', '')
    smtp_pass = smtp_config.get('pass', '')

    sender = mail_data.get('from')
    to_address = mail_data.get('to')

    if not sender or not to_address:
        return False

    receiver = [to_address]
    msg = MIMEMultipart('alternative')

    msg['Subject'] = mail_data.get('subject', 'Test
Message')
    msg['From'] = sender
    msg['To'] = to_address

    text_plain = MIMEText(mail_data.get('body', ''),
'plain', 'UTF-8')
    html = MIMEText(mail_data.get('body_html', ''),
'html', 'UTF-8')

    msg.attach(text_plain)
    msg.attach(html)

    attachment_file = mail_data.get('attachment')

    if attachment_file:
        try:
            with open(attachment_file, 'rb') as f:
                part = MIMEBase('application',
'octet-stream')
                part.set_payload(f.read())
            encoders.encode_base64(part)
```

```
            part.add_header('Content-Disposition',
f'attachment; filename= {attachment_file}')
            msg.attach(part)
        except IOError:
            return False

    context = ssl.create_default_context()

    with smtplib.SMTP_SSL(smtp_host, smtp_port,
context=context) as client:

        client.login(smtp_user, smtp_pass)
        client.sendmail(sender, receiver,
msg.as_string())
        return True
```

La funzione restituisce un valore booleano che indicherà l'esito dell'operazione.