¿Por qué Python es popular en aplicaciones en la nube?

Barbara Gaspar

Conceptos clave

¿Qué es Python?

Se trata de un lenguaje de programación orientado a objetos. Es de propósito general, sus principales aplicaciones, las podemos encontrar en la creación y mantenimiento de aplicaciones web, desarrollo de software, ciencia de datos, Machine learning, etc (IBM, 2024)

¿Qué es la nube?

Es una red de servidores remotos a los que nos podemos conectar mediante internet, para almacenar, administrar y ejecutar recursos informáticos.

Existen diferentes modelos de servicio como Infraestructura como despliegue como (IaaS) Plataforma como servicio (PaaS) y Software como servicio (SaaS) y modelos de servicio nube pública, nube privada y nube híbrida

¿Cuál es el rol de Python en la nube? ¿Por qué están relacionados?

1. Automatización de infraestructura

Gestionar y configurar recursos en la nube mediante herramientas de automatización

Herramientas comunes:

- **Boto3** (AWS).
- **Google Cloud Client Library**.
- **Azure SDK for Python**.

2. Desarrollo de aplicaciones web

- **Frameworks comunes**: Django, Flask, FastAPI

Otros ejemplos

3. **Computación Serverless**

Python es compatible con arquitecturas sin servidor (**serverless**), donde el código se ejecuta en respuesta a eventos sin necesidad de administrar servidores.

- Ejecutar funciones con **AWS Lambda**.
- Crear microservicios con **Google Cloud Functions**.
- Procesar datos en tiempo real con **Azure Functions**.

4. **Integración con DevOps**

Python es clave en las prácticas de **DevOps**, donde se utiliza para:

- Automatizar pipelines de integración y entrega continua (CI/CD).
- Configurar infraestructura como código con herramientas como Ansible, Terraform (con extensiones Python).
- Monitorear y registrar aplicaciones mediante APIs de servicios en la nube.

Caso aplicado

Creación de una instancia EC2 con Boto3

```
import boto3

# Configuración: establecer la región y las credenciales
aws_region = "us-east-1"
ec2_client = boto3.client("ec2", region_name=aws_region)
```

```python
# Crear una nueva instancia EC2
def create_ec2_instance():
    try:
        response = ec2_client.run_instances(
            ImageId="ami-0abcdef1234567890",  # ID de la imagen (AMI)
            InstanceType="t2.micro",          # Tipo de instancia
            MinCount=1,                       # Número mínimo de instancias
            MaxCount=1,                       # Número máximo de instancias
            KeyName="mi-clave-ssh",           # Nombre de la clave SSH
            SecurityGroupIds=["sg-0abcdef1234567890"],  # Grupo de seguridad
            SubnetId="subnet-0abcdef1234567890",        # Subred
            TagSpecifications=[
                {
                    "ResourceType": "instance",
                    "Tags": [{"Key": "Name", "Value": "MiServidorPython"}],
                }
            ],
        )
        print("Instancia EC2 creada:", response["Instances"][0]["InstanceId"])
    except Exception as e:
        print("Error al crear la instancia:", str(e))
```

```
# Llamar a la función para lanzar la instancia
create_ec2_instance()
```

Cuando se ejecuta este script, Python interactúa con la API de AWS para crear una instancia EC2 automáticamente, eliminando la necesidad de realizar esta tarea manualmente desde la consola de AWS.

¿Es difícil? ¿Cómo iniciar?

1. Aprende los fundamentos de Python
2. Familiarízate con elementos clave de la nube
3. Crea una cuenta en el proveedor de nube que tú prefieras
4. Instala python y pip
5. Instala CLI
6. Inicia con código de tareas simples
7. Explora herramientas y procesos de despliegue
8. Integra DevOps y CI/CD