Curso de Introducción al Pensamiento Computacional con Python

Artículo Excepciones y control de flujo

David Aroesti 11 de Diciembre de 2019

Excepciones como control de flujo

Hasta ahora hemos visto como las excepciones nos permiten controlar los posibles errores que pueden ocurrir en nuestro código. Sin embargo, dentro de la comunidad de Python tienen otro uso: control de flujo.

En este momento ya debes estar familiarizado con las estructuras de control flujo que ofrece Python (`if... elif...else`); entonces, ¿por qué es necesaria otra modalidad para controlar el flujo? Una razón muy específica: el principio EAFP (*easier to ask for forgiveness than permission*, es más fácil pedir perdón que permiso, por sus siglas en inglés).

El principio EAFP es un estilo de programación común en Python en el cual se asumen llaves, índices o atributos válidos y se captura la excepción si la suposición resulta ser falsa. Es importante resaltar que otros lenguajes de programación favorecen el principio LBYL (*look before you leap*, revisa antes de saltar) en el cual el código verifica de manera explícita las precondiciones antes de realizar llamadas.

Veamos ambos estilos:

```python
# Python

def busca_pais(paises, pais):
    """
    Paises es un diccionario. Pais es la llave.
    Codigo con el principio EAFP.
    """

    try:
        return paises[pais]
    except KeyError:
        return None
```

```javascript
// Javascript

/**
* Paises es un objeto. Pais es la llave.
* Codigo con el principio LBYL.
*/
function buscaPais(paises, pais) {
  if(!Object.keys(paises).includes(pais)) {
    return null;
  }

  return paises[pais];
}
```

Como puedes ver, el código de Python accede directamente a la llave y únicamente si dicho acceso falla, entonces se captura la excepción y se provee el código necesario. En el caso de JavaScript, se verifica primero que la llave exista en el objeto y únicamente en posterioridad se accede.

Es importante resaltar que ambos estilos pueden utilizarse en Python, pero el estilo EAFP es mucho más “pythonico”.