



Python Identifiers and Reserved Words

Understanding Naming Conventions in Python

Table of Contents

- 01 Understanding Python Identifiers
- 02 Reserved Words in Python



Photo by Pexels

Python Identifiers



Naming Rules

- Identifiers in Python are names for variables, functions, classes, and modules. They can include letters (uppercase or lowercase), digits, and underscores.
- Identifiers cannot start with a digit, are case-sensitive, and cannot use reserved words (keywords) as names. Choose names that are meaningful and readable.
- Best practices include using `snake_case` for variables, `CamelCase` for classes, and uppercase with underscores for constants to enhance code readability.
- Examples of valid identifiers: `my_variable=10`, `MyVariable=20`, `my_variable_2=30`. Examples of invalid identifiers: `2variable=40`, `my-variable=50`.



Reserved Words in Python

Keywords with Special Meanings

- Python has reserved words with special meanings that cannot be used as identifiers. Examples include 'False', 'True', 'and', 'if', 'try', 'class', 'def', 'while', and more.
- Reserved words are keywords like 'break', 'continue', 'if', 'else', and 'yield' that have predefined meanings in Python. They provide structure and functionality to the language.
- Using reserved words correctly is crucial to avoid conflicts and ensure proper functionality. Always refer to the Python documentation for a comprehensive list of reserved words.
- Example usage:

```
def example_function():
    try:
        for i in range(5):
            if i % 2 == 0:
                print(f'{i} is even')
            else:
                print(f'{i} is odd')
    except Exception as e:
        print(f'An error occurred: {e}')
```