Python `os` Module

The `os` module in Python provides functions for interacting with the operating system. It allows you to perform various tasks such as file and directory operations, process management, and system-level configurations.

Key Features of `os` Module

1. **Getting Current Working Directory**

- The `os.getcwd()` method returns the current working directory of a process.

```
import os
current_dir = os.getcwd()
print(current_dir)
```

2. **Changing Directory**

- The `os.chdir(path)` method changes the current working directory to the specified path.

```
os.chdir('/path/to/directory')
```

3. **Listing Files and Directories**

- The `os.listdir(path)` method returns a list of files and directories in the specified path.

```
files = os.listdir('/path/to/directory')
print(files)
```

4. **Creating a Directory**

- The `os.mkdir(path)` method creates a new directory at the specified path.

```
os.mkdir('new_directory')
```

5. **Removing a Directory**

- The `os.rmdir(path)` method removes a directory. Note that the directory must be empty to remove it.

```
os.rmdir('new_directory')
```

6. **Renaming Files or Directories**

- The `os.rename(src, dst)` method renames a file or directory from `src` to `dst`.

```
os.rename('old_name.txt', 'new_name.txt')
```

7. **Removing Files**

- The `os.remove(path)` method removes the specified file.

```
os.remove('file.txt')
```

8. **Checking Path Existence**

- The `os.path.exists(path)` method checks whether the specified path exists.

```
if os.path.exists('file.txt'):
    print('File exists')
else:
    print('File does not exist')
```

9. **Getting File/Directory Information**

- The `os.stat(path)` method returns information about the specified file or directory (such as size, modified time, etc.).

```
info = os.stat('file.txt')
print(info)
```

10. **Environment Variables**

- The `os.environ` allows access to the environment variables of the system. You can retrieve a specific environment variable using `os.getenv()`.

```
home_dir = os.getenv('HOME')
print(home_dir)
```

11. **Joining Paths**

- The `os.path.join()` method is used to join one or more path components in a platform-independent manner.

```
full_path = os.path.join('/home/user', 'documents', 'file.txt')
print(full_path)
```

12. **Splitting Paths**

- The `os.path.split()` method splits a path into two parts: the head (directory) and the tail (file).

```
head, tail = os.path.split('/home/user/file.txt')
print(head)  # Output: /home/user
print(tail)  # Output: file.txt
```

13. **Checking if Path is File or Directory**

- The `os.path.isfile()` method checks if a given path is a file.
- The `os.path.isdir()` method checks if a given path is a directory.

```
if os.path.isfile('file.txt'):
    print('It is a file')

if os.path.isdir('/home/user'):
    print('It is a directory')
```

14. **Running System Commands**

- The `os.system(command)` method allows you to run shell commands directly from Python.

```
os.system('ls')  # List files on Linux/macOS
os.system('dir')  # List files on Windows
```