Python CSV Module Overview

# Table of Contents:

1. Importing the CSV Module  
2. Reading from a CSV File  
 - Using reader()  
 - Using DictReader()  
3. Writing to a CSV File  
 - Using writer()  
 - Using DictWriter()  
4. Modifying CSV File Data

# 1. Importing the CSV Module

Before you can work with CSV files, you need to import the module:

import csv

# 2. Reading from a CSV File

CSV files can be read using either the `reader()` or `DictReader()` functions.

## 2.1 Using reader()

The `reader()` function reads each row of a CSV file as a list of strings.

Example:  
import csv  
with open('example.csv', 'r') as file:  
 reader = csv.reader(file)  
 for row in reader:  
 print(row)  
# Output: Each row will be printed as a list of strings

## 2.2 Using DictReader()

The `DictReader()` function reads each row as an ordered dictionary where the keys are the column headers.

Example:  
import csv  
with open('example.csv', 'r') as file:  
 reader = csv.DictReader(file)  
 for row in reader:  
 print(row)  
# Output: Each row will be printed as a dictionary with column headers as keys

# 3. Writing to a CSV File

CSV files can be written to using either the `writer()` or `DictWriter()` functions.

## 3.1 Using writer()

The `writer()` function writes rows to a CSV file as lists of strings.

Example:  
import csv  
with open('output.csv', 'w', newline='') as file:  
 writer = csv.writer(file)  
 writer.writerow(['Name', 'Age', 'City'])  
 writer.writerow(['Alice', '30', 'New York'])  
# Output: CSV file with two rows written

## 3.2 Using DictWriter()

The `DictWriter()` function writes rows as dictionaries to a CSV file. The keys of the dictionary must match the field names.

Example:  
import csv  
with open('output.csv', 'w', newline='') as file:  
 fieldnames = ['Name', 'Age', 'City']  
 writer = csv.DictWriter(file, fieldnames=fieldnames)  
   
 writer.writeheader()  
 writer.writerow({'Name': 'Alice', 'Age': '30', 'City': 'New York'})  
# Output: CSV file with a header and a single row written

# 4. Modifying CSV File Data

To modify a CSV file, you can read its contents, update the data in memory, and then write it back.

Example:  
import csv  
# Read the data  
rows = []  
with open('example.csv', 'r') as file:  
 reader = csv.reader(file)  
 for row in reader:  
 rows.append(row)  
  
# Modify the data (e.g., change the first row's first column)  
rows[0][0] = 'Updated Value'  
  
# Write the modified data back  
with open('example.csv', 'w', newline='') as file:  
 writer = csv.writer(file)  
 writer.writerows(rows)  
# Output: CSV file with modified data