PARULINSTITUTEOF ENGINEERING &TECHNOLOGY
FACULTY OF ENGINEERING & TECHNOLOGY
PARULUNIVERSITY

303105257 - Programming in Python with Full Stack Development

Modules and Packages.

Computer Science & Engineering

Prabhat Parashar(Asst. Professor. PIET-CSE)

Parul® University

NAAC A++ ACCREDITED UNIVERSITY

Table of Contents:

Parul® University

NAAC A++ ACCREDITED UNIVERSITY

DIGITAL LEARNING CONTENT

Module

A module in programming is **a section of code that is designed to be reused or added to a main program as a whole**. Modules are self-contained units of code that can be used in different projects, and they help keep projects organized.

Here are some characteristics of modules in programming:

- **Independent**: Modules are separate programs that can be designed to perform a specific task.
- **Reusable**: Modules can be used over and over again in different projects.
- **Organized**: Modules keep things organized by hiding how they work and only showing what they can do.
- **Easy to update**: Modules can be updated without affecting the rest of the project.

Creating a Module

To create a Python module, write the desired code and save that in a file with **.py** extension. Let's understand it better with an example:

Example:

Let's create a simple calc.py in which we define two functions, one **add** and another **subtract**.

Python

```
# A simple module, calc.py
def add(x, y):
    return (x+y)

def subtract(x, y):
    return (x-y)
```

Import a Module in Python

We can import the functions, and classes defined in a module to another module using the **import statement** in some other Python source file.

When the interpreter encounters an import statement, it imports the module if the module is present in the search path.

***Note**: A search path is a list of directories that the interpreter searches for importing a module.*

For example, to import the module calc.py, we need to put the following command at the top of the script.

Syntax to Import Module in Python

```
import module
```

Note: This does not import the functions or classes directly instead imports the module only. To access the functions inside the module the dot(.) operator is used.

Package

Python Packages are a way to organize and structure your Python code into reusable components. Think of it like a folder that contains related Python files (modules) that work together to provide certain functionality. Packages help keep your code organized, make it easier to manage and maintain, and allow you to share your code with others. They're like a toolbox where you can store and organize your tools (functions and classes) for easy access and reuse in different projects.

Parul® University

NAAC A++ ACCREDITED UNIVERSITY

DIGITAL LEARNING CONTENT

How to create Package

Creating packages in Python allows you to organize your code into reusable and manageable modules. Here's a brief overview of how to create packages:

- **Create a Directory:** Start by creating a directory (folder) for your package. This directory will serve as the root of your package structure.
- **Add Modules:** Within the package directory, you can add Python files (modules) containing your code. Each module should represent a distinct functionality or component of your package.
- **Init File:** Include an __init__.py file in the package directory. This file can be empty or can contain an initialization code for your package. It signals to Python that the directory should be treated as a package.
- **Subpackages:** You can create sub-packages within your package by adding additional directories containing modules, along with their own __init__.py files.
- **Importing:** To use modules from your package, import them into your Python scripts using dot notation. For example, if you have a module named module1.py inside a package named mypackage, you would import its function like this: from mypackage.module1 import greet.

How to create Package continues

Here's a basic code sample demonstrating how to create a simple Python package:

1. Create a directory named mypackage.
2. Inside mypackage, create two Python files: module1.py and module2.py.
3. Create an __init__.py file inside mypackage (it can be empty).
4. Add some code to the modules.
5. Finally, demonstrate how to import and use the modules from the package.

```
mypackage/
│
├── __init__.py
├── module1.py
└── module2.py
```

Python packages for Web Framework

- Flask: Flask is a lightweight and flexible web framework for Python. It's designed to make getting started with web development in Python quick and easy, with a simple and intuitive interface. Flask provides tools and libraries to help you build web applications, APIs, and other web services.
- Django: Django is a Python web framework for building web applications quickly and efficiently. It follows the DRY principle and includes features like URL routing, database management, and authentication, making development easier. It's highly customizable and widely used in web development.
- FastAPI: Python FastAPI is a high-performance web framework for building APIs quickly and efficiently. It's easy to use, based on standard Python-type hints, and offers automatic interactive documentation. FastAPI is designed to be fast, easy to learn, and ideal for building modern web APIs.

Python Libraries

A Python library is a collection of related modules. It contains bundles of code that can be used repeatedly in different programs. It makes Python Programming simpler and convenient for the programmer. As we don't need to write the same code again and again for different programs. Python libraries play a very vital role in fields of Machine Learning, Data Science, Data Visualization, etc.

List of Important Libraries

1. **TensorFlow:** This library was developed by Google in collaboration with the Brain Team. It is an open-source library used for high-level computations. It is also used in machine learning and deep learning algorithms. It contains a large number of tensor operations. Researchers also use this Python library to solve complex computations in Mathematics and Physics.
2. **Matplotlib:** This library is responsible for plotting numerical data. And that's why it is used in data analysis. It is also an open-source library and plots high-defined figures like pie charts, histograms, scatterplots, graphs, etc.
3. **Pandas:** Pandas are an important library for data scientists. It is an open-source machine learning library that provides flexible high-level data structures and a variety of analysis tools. It eases data analysis, data manipulation, and cleaning of data. Pandas support operations like Sorting, Re-indexing, Iteration, Concatenation, Conversion of data, Visualizations, Aggregations, etc.

List of Important Libraries

4. **Numpy:** The name "Numpy" stands for "Numerical Python". It is the commonly used library. It is a popular machine learning library that supports large matrices and multi-dimensional data. It consists of in-built mathematical functions for easy computations. Even libraries like TensorFlow use Numpy internally to perform several operations on tensors. Array Interface is one of the key features of this library.
5. **SciPy:** The name "SciPy" stands for "Scientific Python". It is an open-source library used for high-level scientific computations. This library is built over an extension of Numpy. It works with Numpy to handle complex computations. While Numpy allows sorting and indexing of array data, the numerical data code is stored in SciPy. It is also widely used by application developers and engineers.
6. **Scrapy:** It is an open-source library that is used for extracting data from websites. It provides very fast web crawling and high-level screen scraping. It can also be used for data mining and automated testing of data.
7. **Scikit-learn:** It is a famous Python library to work with complex data. Scikit-learn is an open-source library that supports machine learning. It supports variously supervised and unsupervised algorithms like linear regression, classification, clustering, etc. This library works in association with Numpy and SciPy.

List of Important Libraries

8. **PyGame:** This library provides an easy interface to the Standard Directmedia Library (SDL) platform-independent graphics, audio, and input libraries. It is used for developing video games using computer graphics and audio libraries along with Python programming language.
9. **PyTorch:** PyTorch is the largest machine learning library that optimizes tensor computations. It has rich APIs to perform tensor computations with strong GPU acceleration. It also helps to solve application issues related to neural networks.
10. **PyBrain:** The name "PyBrain" stands for Python Based Reinforcement Learning, Artificial Intelligence, and Neural Networks library. It is an open-source library built for beginners in the field of Machine Learning. It provides fast and easy-to-use algorithms for machine learning tasks. It is so flexible and easily understandable and that's why is really helpful for developers that are new in research fields.

Introduction to PyCharm IDE

PyCharm is a sturdy and characteristic-packed IDE that many Python builders swear with the aid of, it is not without its negative aspects. These limitations consist of useful resource consumption, a getting-to-know curve, price for the Professional Edition, slower startup instances, and constrained language support. It's critical for builders to not forget their precise desires and options while selecting an IDE, as the "best" IDE can vary depending on the context and individual requirements.

Features of PyCharm IDE

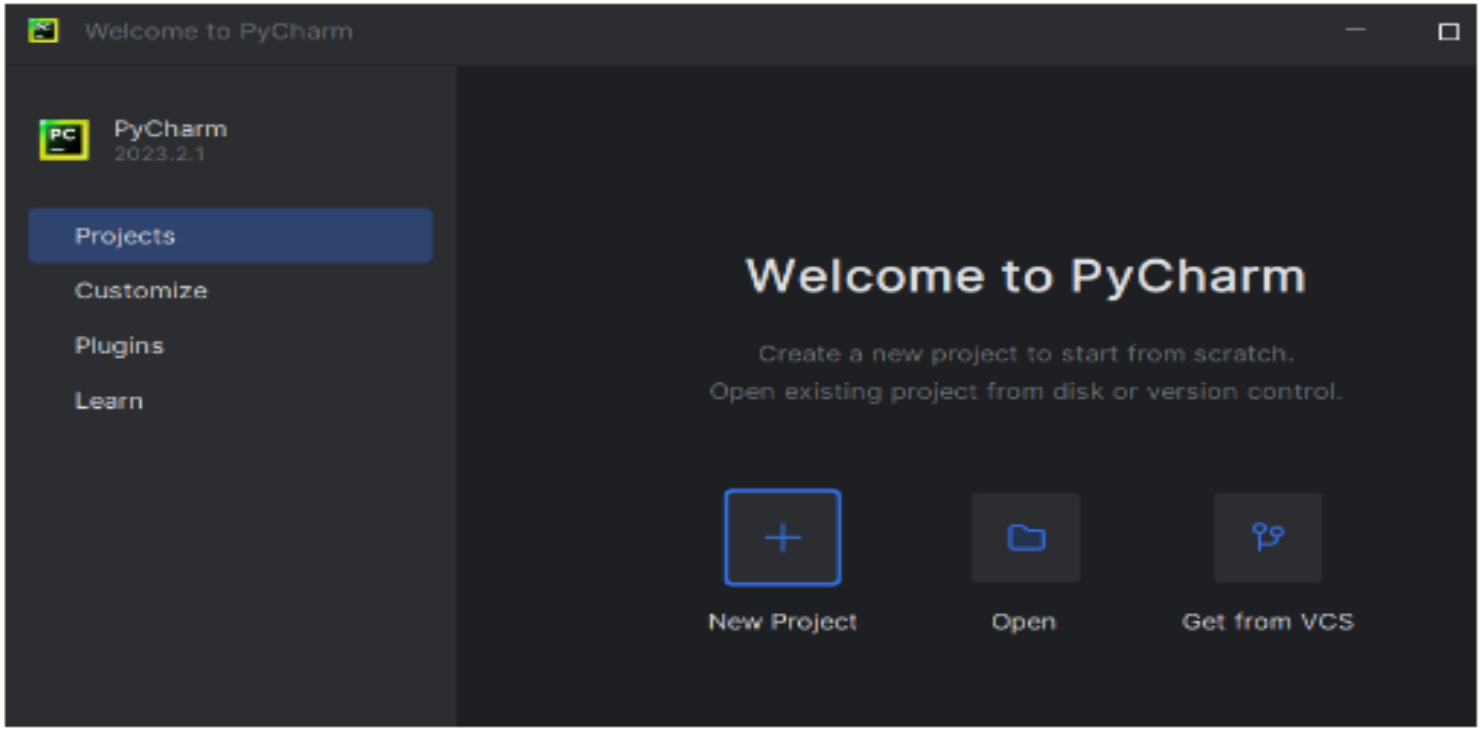
- Feature-wealthy Environment: PyCharm offers a comprehensive set of capabilities for Python improvement, along with code crowning glory, debugging, version manipulation integration, and more.
- Smart Code Assistance: It provides wise code final touch and guidelines, which could appreciably speed up your coding method and reduce mistakes.
- Cross-Platform Support: PyCharm is to be had for Windows, macOS, and Linux, making it on hand to a huge variety of builders.
- Community and Professional Versions: You can pick between the unfastened Community version and the extra characteristic-rich Professional version, depending on your needs and budget.

Parul® University

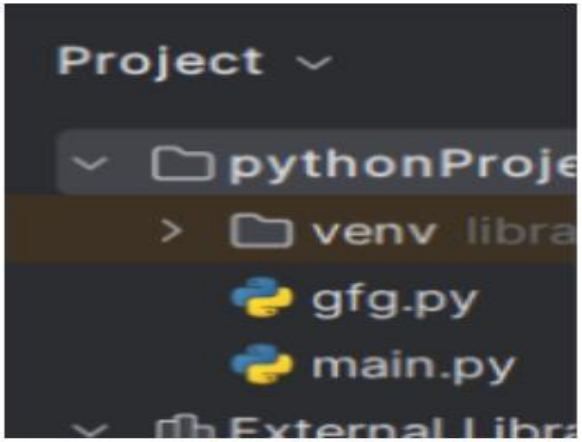
NAAC ACCREDITED UNIVERSITY A++

DIGITAL LEARNING CONTENT

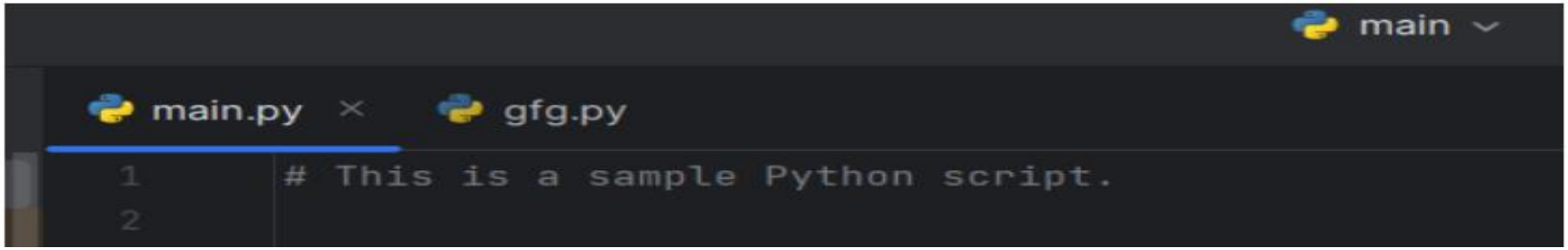
PyCharm IDE



PyCharm IDE

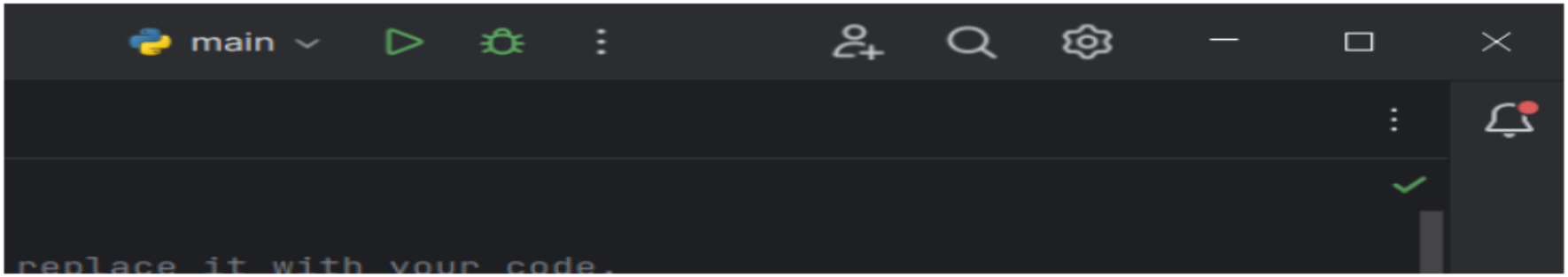


Here you can see the working file, and easily navigated by clicking on that tab.

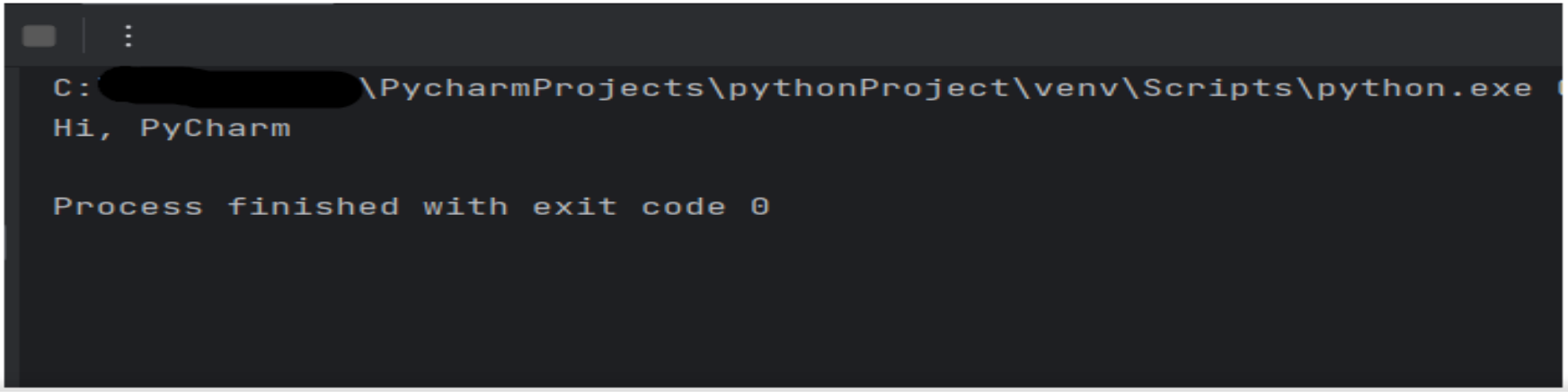


PyCharm IDE

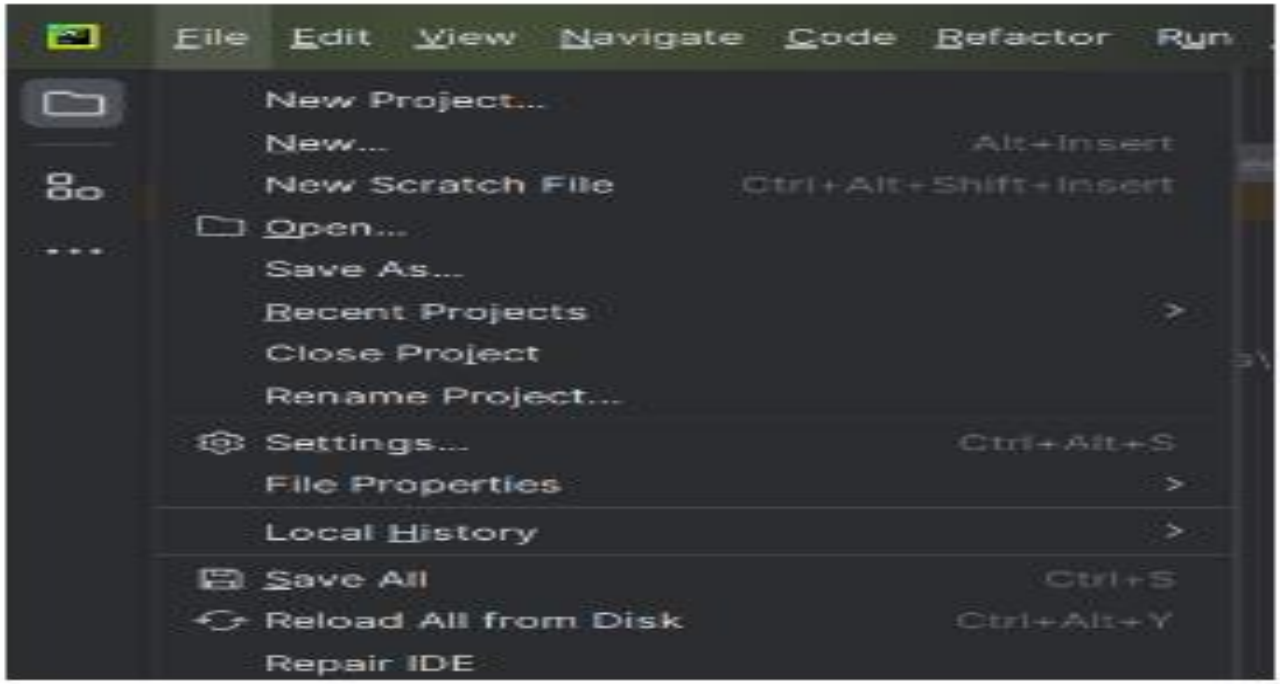
The Green play button to run your code and the bug sign represent code DEBUGGING.



This is the terminal where we see output of the running program.



PyCharm IDE


File Edit View Navigate Code Refactor Run
New Project...
New... Alt+Insert
New Scratch File Ctrl+Alt+Shift+Insert
Open...
Save As...
Recent Projects
Close Project
Rename Project...
Settings... Ctrl+Alt+S
File Properties
Local History
Save All Ctrl+S
Reload All from Disk Ctrl+Alt+Y
Repair IDE

Parul® University

NAAC A++ ACCREDITED UNIVERSITY

DIGITAL LEARNING CONTENT

PyCharm IDE

Here are some common settings and configurations you can adjust using the "Settings" button in PyCharm:

- Editor Settings: Customize code formatting, indentation, code style, and various editor behaviors.
- Interpreter Settings: Configure Python interpreters for your projects, including virtual environments and external interpreters.
- Version Control: Set up version control systems like Git, configure repositories, and define commit and push behaviors.
- Appearance & Behavior: Adjust themes, fonts, and UI-related settings. You can also enable or disable features like code inspections and code folding.
- Keymap: Customize keybindings and keyboard shortcuts to match your preferred coding style.
- Plugins: Manage and install additional plugins and extensions to extend PyCharm's functionality.
- Project Settings: Configure project-specific settings like project structure, deployment options, and run/debug configurations.
- Build, Execution, Deployment: Control how your code is built, executed, and deployed. You can specify run configurations, deployment servers, and more.

Parul® University

NAAC A++ ACCREDITED UNIVERSITY

DIGITAL LEARNING CONTENT

Connecting Python with MySQL Database

Install MySQL Driver

Python needs a MySQL driver to access the MySQL database.

In this tutorial we will use the driver "MySQL Connector".

We recommend that you use PIP to install "MySQL Connector".

PIP is most likely already installed in your Python environment.

Navigate your command line to the location of PIP, and type the following:

Download and install "MySQL Connector":

```
C:\Users\Your Name\AppData\Local\Programs\Python\Python36-32\Scripts>python -m pip install mysql-connector-python
```

Now you have downloaded and installed a MySQL driver.

Connecting Python with MySQL Database

Create Connection

Start by creating a connection to the database.

Use the username and password from your MySQL database:

demo_mysql_connection.py:

```
import mysql.connector

mydb = mysql.connector.connect(
  host="localhost",
  user="yourusername",
  password="yourpassword"
)

print(mydb)
```

MongoDB crude operations

CRUD Operations (Create, Read, Update, and Delete) are the basic set of operations that allow users to interact with the MongoDB server.

As we know, to use MongoDB we need to interact with the MongoDB server to perform certain operations like entering new data into the application, updating data into the application, deleting data from the application, and reading the application data.

C → **Create**

R → **Read**

U → **Update**

D → **Delete**

MongoDB crude operations

Create Operations

Create or insert operations add new **documents** to a **collection**. If the collection does not currently exist, insert operations will create the collection.

MongoDB provides the following methods to insert documents into a collection:

- `db.collection.insertOne()`
- `db.collection.insertMany()`

In MongoDB, insert operations target a single **collection**. All write operations in MongoDB are **atomic** on the level of a single **document**.

```
db.users.insertOne(          <-- collection
   {
      name: "sue",           <-- field: value
      age: 26,               <-- field: value   } document
      status: "pending"      <-- field: value
   }
)
```

MongoDB crude operations

Read Operations

Read operations retrieve **documents** from a **collection**; i.e. query a collection for documents.

MongoDB provides the following methods to read documents from a collection:

- `db.collection.find()`

You can specify **query filters or criteria** that identify the documents to return.

```
db.users.find(
    { age: { $gt: 18 } },
    { name: 1, address: 1 }
).limit(5)
```

← collection
← query criteria
← projection
← cursor modifier

MongoDB crude operations

Update Operations

Update operations modify existing **documents** in a **collection**. MongoDB provides the following methods to update documents of a collection:

- `db.collection.updateOne()`
- `db.collection.updateMany()`
- `db.collection.replaceOne()`

In MongoDB, update operations target a single collection. All write operations in MongoDB are **atomic** on the level of a single document.

You can specify criteria, or filters, that identify the documents to update. These **filters** use the same syntax as read operations.

```
db.users.updateMany(                          <-- collection
   { age: { $lt: 18 } },                      <-- update filter
   { $set: { status: "reject" } }             <-- update action
)
```

MongoDB crude operations

Delete Operations

Delete operations remove documents from a collection. MongoDB provides the following methods to delete documents of a collection:

- `db.collection.deleteOne()`
- `db.collection.deleteMany()`

In MongoDB, delete operations target a single **collection**. All write operations in MongoDB are **atomic** on the level of a single document.

You can specify criteria, or filters, that identify the documents to remove. These **filters** use the same syntax as read operations.

```
db.users.deleteMany(          <-- collection
   { status: "reject" }       <-- delete filter
)
```

Parul® University

NAAC A++ ACCREDITED UNIVERSITY

DIGITAL LEARNING CONTENT

THANK YOU

DIGITAL LEARNING CONTENT

Parul® University

www.paruluniversi