What are the differences between class and object?

|  |  |
| --- | --- |
|  | |
| **Class** | **Object** |
|  |  |
| It is a logical entity. | It is a real-world entity. |
| It is conceptual. | It is real. |
| It binds data and methods together into a single unit. | It is just like a variable of a class. |
| It does not occupy space in the memory. | It occupies space in the memory. |
| It is a data type that represents the blueprint of an object. | It is an instance of the class. |
| It is declared once. | Multiple objects can be declared as and when required. |
|  | It uses the new keyword to create an object. |
| A class can exist without any object. | Objects cannot exist without a class. |

|  |  |
| --- | --- |
| **Constructor** | **Method** |
|  |  |
| Constructor has the same name as the class name. | The method name and class name are not the same. |

What are the key differences between class and structure?

|  |  |
| --- | --- |
|  | |
| **Class** | **Structure** |
|  |  |
| Class is a group of common objects that shares common properties. | The structure is a collection of different data types. |
| It deals with data members and member functions. | It deals with data members only. |
| It supports inheritance. | It does not support inheritance. |
| Member variables cannot be initialized directly. | Member variables can be initialized directly. |
| It is of type reference. | It is of a type value. |
| It's members are private by default. | It's members are public by default. |
| The keyword class defines a class. | The keyword struct defines a structure. |
| An instance of a class is an object. | An instance of a structure is a structure variable. |
| Useful while dealing with the complex data structure. | Useful while dealing with the small data structure. |

What are the differences between the constructor and the method in Java?

|  |  |
| --- | --- |
| It is a special type of method that is used to initialize an object of its class. | It is a set of instructions that can be invoked at any point in a program. |
| It creates an instance of a class. | It is used to execute Java code. |
| It is invoked implicitly when we create an object of the class. | It gets executed when we explicitly called it. |
| It cannot be inherited by the subclass. | It can be inherited by the subclass. |
| It does not have any return type. | It must have a return type. |
| It cannot be overridden in Java. | It can be overridden in Java. |
| It cannot be declared as static. | It can be declared as static. |
| Java compiler automatically provides a default constructor. | Java compiler does not provide any method by default. |

1. How does procedural programming be different from OOP differ?

|  |  |
| --- | --- |
| **Procedural Oriented Programming** | **Object-Oriented Programming** |
|  | J |
| It is based on functions. | It is based on real-world objects. |
| It follows a top-down approach. | It follows a bottom-up approach. |
| It is less secure because there is no proper way to hide data. | It provides more security. |
| Data is visible to the whole program. | It encapsulates the data. |
| Reuse of code is not allowed. | The code can be reused. |
| Modification and extension of code are not easy. | We can easily modify and extend code. |
| Examples of POP are C, VB, FORTRAN, Pascal, etc. | Examples of OOPs are C++, Java, C#, .NET, etc. |

1. What are the differences between error and exception?

|  |  |  |
| --- | --- | --- |
|  | | |
| **Basis of**  **Comparison** | **Exception** | **Error** |
|  |  |  |
| **Recoverable/**  **Irrecoverable** | Exception can be recovered by using the try-catch block. | An error cannot be recovered. |
| **Type** | It can be classified into two categories i.e. checked and unchecked. | All errors in Java are unchecked. |
| **Occurrence** | It occurs at compile time or run time. | It occurs at run time. |
| **Package** | It belongs to java.lang.Exception package. | It belongs to java.lang.Error package. |
| **Known or**  **unknown** | Only checked exceptions are known to the compiler. | Errors will not be known to the compiler. |
| **Causes** | It is mainly caused by the application itself. | It is mostly caused by the environment in which the application is running. |
| **Example** | **Checked Exceptions:** SQLException, IOException  **Unchecked**  **Exceptions:** ArrayIndexOutOfBoundException,  NullPointerException, ArithmaticException | Java.lang.StackOverFlow, java.lang.OutOfMemoryError |

|  |  |
| --- | --- |
| **Abstract class** | **Interface** |
| 1) Abstract class can **have abstract and non-abstract** methods. | Interface can have **only abstract** methods. Since Java 8, it can have **default and static methods** also. |
| 2) Abstract class **doesn't support multiple inheritance**. | Interface **supports multiple inheritance**. |
| 3) Abstract class **can have final, non-final, static and non-static variables**. | Interface has **only static and final variables**. |
| 4) Abstract class **can provide the implementation of interface**. | Interface **can't provide the implementation of abstract class**. |
| 5) The **abstract keyword** is used to declare abstract class. | The **interface keyword** is used to declare interface. |
| 6) An **abstract class** can extend another Java class and implement multiple Java interfaces. | An **interface** can extend another Java interface only. |
| 7) An **abstract class** can be extended using keyword "extends". | An **interface** can be implemented using keyword "implements". |
| 8) A Java **abstract class** can have class members like private, protected, etc. | Members of a Java interface are public by default. |
| 9)**Example:** public abstract class Shape{ public abstract void draw(); } | **Example:** public interface Drawable{ void draw(); } |

Simply, abstract class achieves partial abstraction (0 to 100%) whereas interface achieves fully abstraction (100%).

Example of abstract class and interface in Java

Let's see a simple example where we are using interface and abstract class both.

1. //Creating interface that has 4 methods
2. **interface** A{
3. **void** a();//bydefault, public and abstract
4. **void** b();
5. **void** c();
6. **void** d();
7. }
9. //Creating abstract class that provides the implementation of one method of A interface
10. **abstract** **class** B **implements** A{
11. **public** **void** c(){System.out.println("I am C");}
12. }
14. //Creating subclass of abstract class, now we need to provide the implementation of rest of the methods
15. **class** M **extends** B{
16. **public** **void** a(){System.out.println("I am a");}
17. **public** **void** b(){System.out.println("I am b");}
18. **public** **void** d(){System.out.println("I am d");}
19. }
21. //Creating a test class that calls the methods of A interface
22. **class** Test5{
23. **public** **static** **void** main(String args[]){
24. A a=**new** M();
25. a.a();
26. a.b();
27. a.c();
28. a.d();
29. }}

Output:

I am a

I am b

I am c

I am d

Following are the important differences between Web Server and Application Server.

| **Sr. No.** | **Key** | **Web Server** | **Application Server** |
| --- | --- | --- | --- |
| 1 | Purpose | Web Server contains Web container only. | Application Server contains Web Container plus EJB Container. |
| 2 | Useful | A web server is good in case of static contents like static html pages. | Applcation server is relevant in case of dynamic contents like bank websites. |
| 3 | Resource Consumption | Web server consumes less resources like CPU, Memory etc. as compared to application server. | Application server utilizes more resources. |
| 4 | Target Environment | Web Server provides the runtime environment for web applications. | Application server provides the runtime environment for enterprise applications. |
| 5 | Multithreading support | Multithreading is not supported. | Multithreading is supported. |
| 6 | Protocol(s) supported | Web Server supports HTTP Protocol. | Application Server suppots HTTP as well as RPC/RMI protocols. |
| 7 | Example | Apache Web Server. | Weblogic, JBoss. |

Apache Tomcat (Link resides outside IBM) is an open source **application server** that executes Java Servlets, renders and delivers web pages that include JavaServer Page code, and serves Java Enterprise Edition (Java EE) applications. Released in 1998, Tomcat is the most widely used open source Java application server.

## **Key differences between Artificial Intelligence (AI) and Machine learning (ML):**

|  |  |
| --- | --- |
| **Artificial Intelligence** | **Machine learning** |
| Artificial intelligence is a technology which enables a machine to simulate human behavior. | Machine learning is a subset of AI which allows a machine to automatically learn from past data without programming explicitly. |
| The goal of AI is to make a smart computer system like humans to solve complex problems. | The goal of ML is to allow machines to learn from data so that they can give accurate output. |
| In AI, we make intelligent systems to perform any task like a human. | In ML, we teach machines with data to perform a particular task and give an accurate result. |
| Machine learning and deep learning are the two main subsets of AI. | Deep learning is a main subset of machine learning. |
| AI has a very wide range of scope. | Machine learning has a limited scope. |
| AI is working to create an intelligent system which can perform various complex tasks. | Machine learning is working to create machines that can perform only those specific tasks for which they are trained. |
| AI system is concerned about maximizing the chances of success. | Machine learning is mainly concerned about accuracy and patterns. |
| The main applications of AI are **Siri, customer support using catboats**, Expert System, Online game playing, intelligent humanoid robot, etc. | The main applications of machine learning are **Online recommender system**, **Google search algorithms**, **Facebook auto friend tagging suggestions**, etc. |
| On the basis of capabilities, AI can be divided into three types, which are, **Weak AI**, **General AI**, and **Strong AI**. | Machine learning can also be divided into mainly three types that are **Supervised learning**, **Unsupervised learning**, and **Reinforcement learning**. |
| It includes learning, reasoning, and self-correction. | It includes learning and self-correction when introduced with new data. |
| AI completely deals with Structured, semi-structured, and unstructured data. | Machine learning deals with Structured and semi-structured data. |

**Weak (Narrow) AI**

The artificial intelligence that we now see in our daily lives is what we refer to as weak (narrow AI). It’s actually the only artificial intelligence that exists today. It’s called weak because it has limited functionality. The AI algorithms are built to accomplish specific tasks rather than possess full cognitive abilities like the human brain. Weak AI illustrates that intelligent behavior can be modeled. Examples of weak AI include Siri, Alexa and Google search.

**Strong (General) AI**

Strong or general artificial intelligence at this point is only theoretical. It’s when machines develop consciousness and decision-making on par (or better) than humans. When strong AI is achieved, the machines won’t rely on human programming to be able to think or accomplish tasks. This is the AI imagined in sci-fi stories such as “I-Robot” and “The Terminator.”

**Why al and ml?**

AI and machine learning help you understand your customer’s preferences. A website with AI and ML features helps you analyze your customer’s preferences, search history, and even location. This way you will be able to design and improve your website according to the needs of your customers. You can also give them a better customer experience by updating your UI accordingly.

Seaborn is **a library for making statistical graphics in Python**. It builds on top of matplotlib and integrates closely with pandas data structures. Seaborn helps you explore and understand your data.

## **internet of things**

The Internet of things describes physical objects with sensors, processing ability, software, and other technologies that connect and exchange data with other devices and systems over the Internet or other communications networks.