Lambda Expression in Java 8

Arraylist.foreach(I -> System.out.println(I));

// Foreach is the internal part of collection.

As given above the arrow ‘->’ is called lambda expression.

**Anonymous Class**

**package** Demo;

**interface** A{

**void** show();

}

//class xyz implements A{

// public void show() {

// System.out.print("Hello World");

// }

//}

**public** **class** LearnLambda {

A obj = **new** A() {

**public** **void** show() {

System.***out***.print("Hello World");

}

};

}

Now, if we have A obj on one side then it’s obvious to have new A() on another and the same can be assumed.

Hence no need to write it there so after removing the boiler plate code from above class LearnLambda it becomes something like as below –

**package** Demo;

**interface** A{

**void** show();

}

//class xyz implements A{

// public void show() {

// System.out.print("Hello World");

// }

//}

**public** **class** LearnLambda {

// A obj = new A() {

// public void show() {

// System.out.print("Hello World");

// }

// };

A obj = ()->{

System.***out***.print("Hello World");

};

Obj.show();

}

Now as we have defined anonymous inner class in LearnLambda, then after compiling the code we do not get any .class file for this inner class as well.