New Concepts in interface in one example: JDK 8 and JDK 9

```
package com.app;
@FunctionalInterface //JDK1.8-Having only one abstract method
public interface Sample<T> {
	//default method-JDK1.8-can override
	public default void show() {
		System.out.println("aaa");
		showA();
	}
	//private method-JDK1.9-no access outside
	private void showA() {
		System.out.println("Hello");
	}
	//static method-JDK1.8- no override/only using interfaceName.method()
	public static void newA() {
		System.out.println("M");
	}
	//single abstract method-So funcational interface
	public void getInfo(T a);
	//static-main-method accessed
	public static void main(String[] args) {
		//method reference concept-JDK1.8
		Sample<String> s=System.out::println;
		s.getInfo("Hello");

		//lambda expressions-JDK1.8
		Sample<String> s2=(p)->{System.out.println(p);};
		s2.show();

		//static method call -JDK1.8
		Sample.newA();
	}
}
```