1. **Adapter pattern: -**
2. This pattern uses when we need a class that is not implementing the interface you required.
3. When we are writing a framework and there are some unwritten code and interface which will use this.
4. Also known as Wrappers.
5. Match interfaces of Different classes.

There are two types of adapter patterns, The main difference:

* **Class Adapter** uses *inheritance* and can only wrap a **class**. It cannot wrap an interface since by definition it must derive from some base class.
* **Object Adapter** uses *composition* and can wrap classes or interfaces, or both. It can do this since it contains, as a private, encapsulated member, the class or interface **object instance** it wraps.
* if your language does not support multiple inheritance (such as Java, for instance), and you want to adapt several adaptees in one shot, you'll be obliged to use an object adapter.