Factory Methods

Factory Method is widely used.

They appears in almost all Java frameworks.

A *simple factory method* (not really the pattern) is also very useful for simplifying object creation.

For example:

```
Calendar cal = Calendar.newInstance( );
```

The explanation of Factory method is given in separate doc file.

Factory Methods

Factory Method is a method that creates objects.

They are often used instead of contructors, especially where creating object is complex or you want to create *subclasses of the same type.*

Simple Factory Method

Example: Calendar class does not have a public constructor.

You must use a static factory method.

```
Calendar cal = Calendar.getInstance( );
Calendar cal =
            Calendar.getInstance( locale );
cal.toString()
   //Thai Calendar:  24 Jan 2554
   //US Calendar:    Jan 24, 2011
   //Japan Calendar: 2011 Jan 24
```

Factory Method can create subtype

A factory method has freedom to create an object of a subclass.

```
Calendar cal = Calendar.getInstance( );
// What did it create?
// Use getClass() to get the class name
System.out.println(
            cal.getClass().getName() );


java.util.GregorianCalendar
```

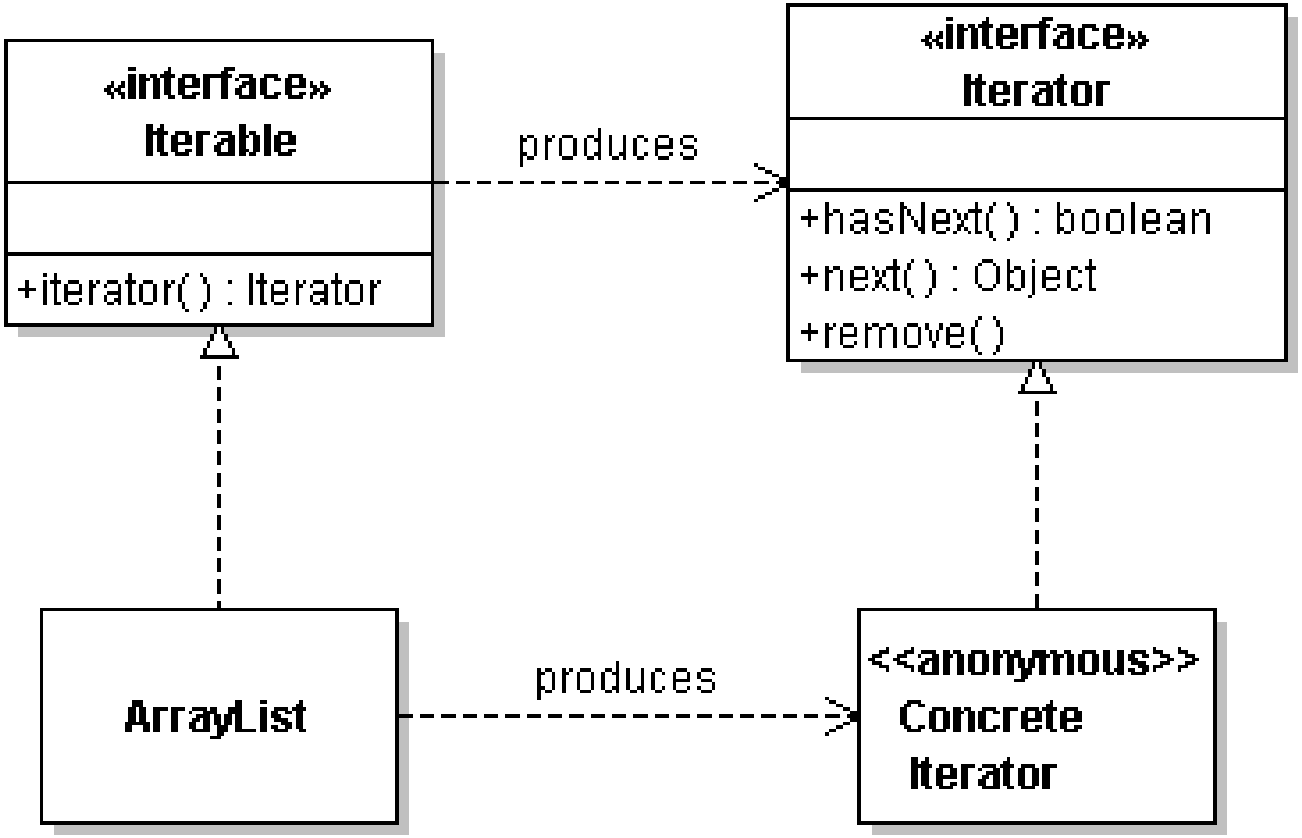
Java API has Many Factory Methods

```
// An immutable date object
LocalDate date = LocalDate.now( );
// A Double object from a String
Double pi = Double.valueOf("3.14159");
// Connection to a database for JDBC
// creating a connection is complex and
// might fail, so use factory method
String url = "jdbc:mysql://somehost/oopdb";
Connection connection =
      DriverManager.getConnection(url);
```

iterable.iterator( ) is Factory Method

How to create iterators?

We don't want the application to know the logic for creating different kinds of iterators.



Factory Method

Factory Interface	Iterable
Factory Method	iterator( )
Product created by factory	Iterator
Concrete Product	class that implements Iterator