IMPORT STATEMENTS IN JAVA

To access a class or method from another package we need to use the fully qualified name or we can use import statements. The class or method should also be accessible. Accessibility is based on the access modifiers. Private members are accessible only within the same class. So you won’t be able to access a private member even with fully qualified name or an import statement.

Below packages from JDK are automatically imported into your code by Java:

- java.lang package
- default package (package with no name). However the use of default package is disabled.

Using import statement
You can use an import statement to import a single class (import java.util.List;) or all classes of a package (import java.util.*;). Remember that you can import only types (classes and interfaces, but not methods) using a regular import.

Consider an example:

```java
package util;

public class BitUtils {

    public static void process(byte[] b) { /* some code here */ }

}
```

You can access the static method process from another package either as:

```java
util.BitUtils.process(bytes);
```

or using an import statement and then use the class name without qualifier as:

```java
import util.BitUtils;
...
BitUtils.process(bytes);
```

What if the process method was default (no access modifier specified)? You cannot use a default access method from another package, as default members are not accessible outside the package.
When you use an import statement you are declaring only the package and not any sub packages. Therefore import java.util.*; import all of the classes in the java.util package, but not the java.util.jar classes or java.util.regex packages.

**Static Imports**

Regular import can import only types (classes and interfaces, but not methods). Static imports can import a class's static members (variables and methods) as well.

Consider our BitUtils example again:

```java
package util;

public class BitUtils {
    public static void process(byte[] b) { /* some code here */ }
}
```

You can also use a static import as:

```java
import static util.BitUtils.process;
```

And then simply use the process method as:

```java
process(bytes);
```

You can also use the fully qualified name and regular import here.

Another very common example is to use a static import for System.out.println as:

```java
import static java.lang.System.out;
```

And then using it as:

```java
out.println();
```

Even if the feature is called static import, syntax is import static.

*Source: http://javajee.com/import-statements-in-java*