Groovy. Differences with Java.

Groovy tries to be as natural as possible for Java developers. Groovy tries to follow the principle of least surprise, particularly for developers learning Groovy who've come from a Java background.

Here's a list of all major differences between Java and Groovy.

1. Default imports

All these packages and classes are imported by default, i.e. you do not have to use an explicit **import** statement to use them:

- java.io.*
- java.lang.*
- java.math.BigDecimal
- java.math.BigInteger
- java.net.*
- java.util.*
- groovy.lang.*
- groovy.util.*

2. Multi-methods

In Groovy, the methods which will be invoked are chosen at runtime. This is called runtime dispatch or multi-methods. It means that the method will be chosen based on the types of the arguments at runtime. In Java, this is the opposite: methods are chosen at compile time, based on the declared types.

The following code, written as Java code, can be compiled in both Java and Groovy, but it will behave differently:

```
int method(String arg) {
    return 1;
}
int method(Object arg) {
    return 2;
}
Object o = "Object";
int result = method(o);
```

In Java, you would have:

```
assertEquals(2, result);
```

Whereas in Groovy:

```
assertEquals(1, result);
```

That is because Java will use the static information type, which is that **o** is declared as an **Object** , whereas Groovy will choose at runtime, when the method is actually called. Since it is called with a **String** , then the **String** version is called.