What is an object reference?

In Java, each and every object is referenced by an object reference. If a variable is declared to be of an object type, e.g., `Student`, the value of the variable is actually an object reference. If you follow the reference, you will see the actual object. So, if you defined a class named `Student` and declared a variable `s1` to be of type `Student` like this:

```java
Student s1 = new Student(...); // Line 1
```

or in two lines like this:

```java
Student s1;
s1 = new Student(...);
```

then the situation can be viewed pictorially as follows:

```
+-------+ +-------------------+
| 200   | <a field value> |
+-------+ +-------------------+
| <a field value> |
| . . . |
| <a field value> |
+-------------------+
```

Here the memory address 200 is arbitrary, i.e., whatever memory address that the Java runtime memory allocation module happened to have found as it was creating the student instance due to Line 1.

In the diagram, `s1` is 200. That is, they are identical. The Java compiler remembers `s1` as 200 internally at run time. There is no name `s1` at run time. It is kept as 200 which is an object reference. `s1` is the name we see in the program that we write, but it is gone by the time the program is running at run time. A symbolic name like `s1` is converted to a memory address, e.g., 200 in this case. I will draw it as I did in the diagram above for readability.

What is the default value for an object reference in Java? It is `null`.

That is, if a variable is declared to be of an object type without initializing it with a different value, the value of the variable is by default set to be `null`. For example,

```java
Student s2;
```

will create a situation that looks like this:

```
+-------+
| 200   |
+-------+
| null  |
```