Here is an example of creating and using a variable that will store an integer value:

```
package com.github.akarazhev.jacademy.jprog.basics;

public final class FortyTwo {

    public static void main(final String[] args) {
        int meaningOfLife;
        meaningOfLife = 42;
        System.out.println(meaningOfLife);
    }
}
```

Variable declaration

The statement `int meaningOfLife` is called a **variable declaration** and it causes Java to reserve a space in memory to hold an integer value; it names that space `meaningOfLife`. In languages like Python or Javascript, you can skip this step, but C and C++ also require variable declarations.

The structure of a variable declaration is the type of the variable (`int` in this case) followed by the name of the variable (`meaningOfLife`).

Unlike in dynamically-typed languages like Python or Javascript, the variable `meaningOfLife` must hold an `int` now and forever. Assigning a floating-point value like `3.14` to `meaningOfLife` will fail with a compiler error. This may seem inconvenient, but has an advantage: if you stated the intent that `meaningOfLife` be an integer, and later try to store something else in that variable, there is an inconsistency between intent or action, a likely bug. It's better to catch such bugs at compile-time rather than during deployment.

Java variable naming convention: camel case

Variable names are often composed of many words. In Java, the first letter should be lowercase, and each new word should be uppercase. This convention is sometimes called "camel case", since the capital letters look like the humps on a camel's back: `meaningOfLife`.

In contrast, some other languages, such as Python, conventionally use separate words in variable names by underscores: `meaning_of_life`.

Combined declaration and assignment

In order to use a variable in Java, you must first do two things:

1. declare the variable, giving it a name and type
2. assign an initial value to that variable, using the assignment operator `=`

It's good to think of these as separate steps, but it is sometime convenient to write one line of code to do both at the same time, like this:

```
package com.github.akarazhev.jacademy.jprog.basics;

public final class CombinedDeclarationAndAssignment {

    public static void main(final String[] args) {
        final int x = 5;
        System.out.println(x);
    }
}
```

Fractional values of numbers with 'double'

Intention: Use a double to store a number that has values after the decimal point.

The `int` data type only represents numbers without a fractional part. What if we want to store an approximation for π in a mathematics program? We can use a different data type, called a `double`, to store numbers that have values after the decimal point. `double` is short for *double-precision floating point*. Floating point means that there is a decimal point that can be placed at different locations (or float), in the number.

Computer programming languages evolve over time; double-precision just means that this representation allows double the precision that an earlier representation of floating-point numbers used. There is also a floating point type called `float` in Java, which uses less memory and has less precision than a double. It is rarely used.