The Java Tutorials have been written for JDK 8. Examples and practices described in this page don't take advantage of improvements introduced in later releases and might use technology no longer available.
See Java Language Changes for a summary of updated language features in Java SE 9 and subsequent releases.
See JDK Release Notes for information about new features, enhancements, and removed or deprecated options for all JDK releases.

Declaring Member Variables

There are several kinds of variables:

- Member variables in a class—these are called *fields*.
- Variables in a method or block of code—these are called *local variables*.
- Variables in method declarations—these are called *parameters*.

The `Bicycle` class uses the following lines of code to define its fields:

```
public int cadence;
public int gear;
public int speed;
```

Field declarations are composed of three components, in order:

1. Zero or more modifiers, such as `public` or `private`.
2. The field's type.
3. The field's name.

The fields of `Bicycle` are named `cadence`, `gear`, and `speed` and are all of data type integer (`int`). The `public` keyword identifies these fields as public members, accessible by any object that can access the class.

Access Modifiers

The first (left-most) modifier used lets you control what other classes have access to a member field. For the moment, consider only `public` and `private`. Other access modifiers will be discussed later.

- `public` modifier—the field is accessible from all classes.
- `private` modifier—the field is accessible only within its own class.

In the spirit of encapsulation, it is common to make fields private. This means that they can only be *directly* accessed from the Bicycle class. We still need access to these values, however. This can be done *indirectly* by adding public methods that obtain the field values for us:

```
public class Bicycle {

    private int cadence;
    private int gear;
    private int speed;

    public Bicycle(int startCadence, int startSpeed, int startGear) {
        gear = startGear;
        cadence = startCadence;
        speed = startSpeed;
    }

    public int getCadence() {
        return cadence;
    }

    public void setCadence(int newValue) {
        cadence = newValue;
    }

    public int getGear() {
        return gear;
    }

    public void setGear(int newValue) {
        gear = newValue;
    }

    public int getSpeed() {
        return speed;
    }

    public void applyBrake(int decrement) {
        speed -= decrement;
    }

    public void speedUp(int increment) {
        speed += increment;
    }
}
```

Types

All variables must have a type. You can use primitive types such as `int`, `float`, `boolean`, etc. Or you can use reference types, such as strings, arrays, or objects.

Variable Names

All variables, whether they are fields, local variables, or parameters, follow the same naming rules and conventions that were covered in the Language Basics lesson, Variables—Naming.

In this lesson, be aware that the same naming rules and conventions are used for method and class names, except that

- the first letter of a class name should be capitalized, and
- the first (or only) word in a method name should be a verb.