Static vs Instance Variables

Let's discuss the differences between static variables and instance variables.

Static Variables

Declared by using the keyword static.

Static variables are also known as static member variables.

Every instance of the class shares the same static variable.

If changes are made to that variable, all other instances of that class will see the effect of that change.

Static Variables

It is considered best practice to use the Class name and not a reference variable to access a static variable.

```
class Dog {

    static String genus = "Canis";

    void printData() {

        Dog d = new Dog();
        System.out.println(d.genus);          // Confusing!
        System.out.println(Dog.genus);        // Clearer!
    }
}
```

Static Variables

An instance isn't required to exist to access the value of a static variable.

```
class Dog {

    static String genus= "Canis";
}

class Main {

    public static void main(String[] args) {
        System.out.println(Dog.genus);        // No instance of Dog needs to exist, in order to access a static variable
    }
}
```

Static Variables

Static variables aren't used very often but can sometimes be very useful.

They can be used for:

- Storing counters.
- Generating unique IDs.
- Storing a constant value that doesn't change, like PI, for example.
- Creating and controlling access to a shared resource.

Static Variables

```java
class Dog {

    private static String name;

    public Dog(String name) {
        Dog.name = name;
    }

    public void printName() {
        System.out.println("name = " + name);  // Using Dog.name would have made this code less confusing.
    }
}

public class Main {

    public static void main(String[] args) {

        Dog rex = new Dog("rex");              // create instance (rex)
        Dog fluffy = new Dog("fluffy");        // create instance (fluffy)
        rex.printName();                       // prints fluffy
        fluffy.printName();                    // prints fluffy
    }
}
```

Instance Variables

They **don't** use the **static** keyword.

They're also known as fields or member variables.

Instance variables belong to a specific instance of a class.

Instance Variables

Each instance has its own copy of an instance variable.

Every instance can have a different value.

Instance variables represent the state of a specific instance of a class.

Instance Variables

```java
class Dog {

    private String name;

    public Dog(String name) {
        this.name = name;
    }

    public void printName() {
        System.out.println("name = " + name);
    }
}

public class Main {

    public static void main(String[] args) {

        Dog rex = new Dog("rex");                // create instance (rex)
        Dog fluffy = new Dog("fluffy");          // create instance (fluffy)
        rex.printName();                         // prints rex
        fluffy.printName();                      // prints fluffy
    }
}
```