

C Data Types Not Supported By the Java Language

Trail: Learning the Java Language

Lesson: The Nuts and Bolts of the Java Language

C Data Types Not Supported By the Java Language

The Java language does not support pointers, struct, or union.

pointers

Some data types in Java, such as objects and arrays, are reference data types. The value of a variable whose data type is a reference type is a reference (or a pointer in other terminology) to the actual data. However, Java does not have an explicit pointer type. You cannot construct a reference to anonymous memory. In addition to making programming easier, this prevents common errors due to pointer mismanagement.

struct and union

The Java language does not support either struct or union. Instead you use classes or interfaces to build composite types. For example, in C, you might declare a structure that contains information about employees like this:

```
struct employee {
    char name[NAMESIZE];
    char address[ADRSIZE];
    long ssn;
    double salary;
    double (*compute_raise)(double, double);
};
```

The last line of this structure is a pointer to a function. When you allocate and initialize an employee structure, you must supply the structure with a pointer to a function, and that function must be defined as indicated in the struct declaration. In the example above, the compute_raise function must accept two doubles as arguments, and return a double. This function will do the trick.

```
double compute_raise(double salary, double percent)
{
    return salary * percent;
}
```

Note that even though salary is part of the structure we still need to supply that value to the function. Here's a main program that creates, initializes an employee structure, and then computes a 10% raise for that person. Note that the main program has access to information (the employee's salary) that ideally should be kept private.

```
main()
{
    struct employee George = {
        "George",
        "NOWHERE",
        123456789,
```

```

        45000.00,
        compute_raise
    };
    printf("raise = %f\n", George.compute_raise(George.salary, 0.10));
}

```

In Java, structures are completely superseded by classes which provide for a cleaner way to bundle data and methods together, and a way to keep some of that data private to the class.

In Java, instead of the struct declared above, you would declare a class to maintain information about employees:

```

class Employee {
    String name;
    String address;
    long ssn;
    private double salary;
    double compute_raise(double percent) {
        return percent * salary;
    }
    Employee(String a_name, String a_address, long a_ssn, double a_salary){
        name = a_name;
        address = a_address;
        ssn = a_ssn;
        salary = a_salary;
    };
}

```

Note that the class includes the implementation of the compute_raise method. Also note that the salary variable is declared private--this means that only an instance of this class has access to the salary information--thereby keeping the information protected from prying eyes. Try doing that with C.

This application can compute George's raise without ever obtaining the salary information directly.

```

class MainClass {
    public static void main(String[] args) {
        Employee george = new Employee("George", "NOWHERE", 123456789,
45000.0);
        System.out.println("raise = " + george.compute_raise(0.10));
    }
}

```