

C++ Constructors

Constructors

A constructor in C++ is a **special method** that is automatically called when an object of a class is created.

To create a constructor, use the same name as the class, followed by parentheses `()`:

Example

```
class MyClass {    // The class
public:           // Access specifier
    MyClass() {    // Constructor
        cout << "Hello World!";
    }
};

int main() {
    MyClass myObj;    // Create an object of MyClass (this will call the
constructor)
    return 0;
}
```

Note: The constructor has the same name as the class, it is always **public**, and it does not have any return value.

Constructor Parameters

Constructors can also take parameters (just like regular functions), which can be useful for setting initial values for attributes.

The following class have **brand**, **model** and **year** attributes, and a constructor with different parameters. Inside the constructor we set the attributes equal to the constructor parameters (**brand=x**, etc). When we call the constructor (by

creating an object of the class), we pass parameters to the constructor, which will set the value of the corresponding attributes to the same:

Example

```
class Car {           // The class
public:               // Access specifier
    string brand;     // Attribute
    string model;     // Attribute
    int year;         // Attribute
    Car(string x, string y, int z) { // Constructor with parameters
        brand = x;
        model = y;
        year = z;
    }
};

int main() {
    // Create Car objects and call the constructor with different values
    Car carObj1("BMW", "X5", 1999);
    Car carObj2("Ford", "Mustang", 1969);

    // Print values
    cout << carObj1.brand << " " << carObj1.model << " " <<
carObj1.year << "\n";
    cout << carObj2.brand << " " << carObj2.model << " " <<
carObj2.year << "\n";
    return 0;
}
```

Just like functions, constructors can also be defined outside the class. First, declare the constructor inside the class, and then define it outside of the class by specifying the name of the class, followed by the scope resolution `::` operator, followed by the name of the constructor (which is the same as the class):

Example

```
class Car {           // The class
public:               // Access specifier
    string brand;     // Attribute
    string model;     // Attribute
    int year;         // Attribute
    Car(string x, string y, int z); // Constructor declaration
};
```

```

// Constructor definition outside the class
Car::Car(string x, string y, int z) {
    brand = x;
    model = y;
    year = z;
}

int main() {
    // Create Car objects and call the constructor with different values
    Car carObj1("BMW", "X5", 1999);
    Car carObj2("Ford", "Mustang", 1969);

    // Print values
    cout << carObj1.brand << " " << carObj1.model << " " <<
carObj1.year << "\n";
    cout << carObj2.brand << " " << carObj2.model << " " <<
carObj2.year << "\n";
    return 0;
}

```