Member Functions

Because `string` is a library or class type, it also has methods, just like the Java `String` class has methods such as `length()`, `toUpper()` and `charAt()`. In C++ instead of calling these **methods**, we use the term **member function** instead. Let's look at the difference between a regular (or "free") function in C++, and a member function.

In the **`string`** class, you've already seen the **`getline()`** function. The prototype for **`getline()`** looks like this:

```
istream& getline(istream&> in, string& str);
```

The function has **two parameters**: the input stream to read from, and the `string` object to modify; it returns a reference to its input stream (which may be ignored).

```
string line;
getline(cin, line);
```

Although **`getline()`** is declared inside the **`<string>`** header, it **is not** part of the **`string class`**; it is just a regular function. **Member functions**, in contrast, **are part of a class**, and, as in Java, they are called by using a special syntax:

```
receiver.request(arguments);
```

In this case, *receiver* is an **object**, and *request* is a member function defined in that class. When compiled, the address of the *receiver* object is passed to the member function as an **invisible** or **implicit** first parameter. Inside the member function, that implicit parameter is accessed using the keyword **`this`**, in a manner similar to Java.