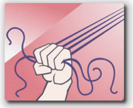
Creating *string* Objects



As in Java, `string` **is a library class type;** it is not part of the C++ language. As in most programming languages, the C++ `string` type is a sequence of characters, which can be treated as a single unit. The class is declared in the `<string>` header, which you **must** include, (unlike Java).

There are several different ways you to **create** `string`objects:

```
string s1;                  // empty string
string s2{"Hello"};         // explicitly initialized
string s3 = "World";        // Legacy C/Java style
string s4{s3};              // a copy of s3
string s5{'c', 'a', 't'};   // a sequence of chars
string s6{R"("bob")"};      // a raw string
string s7(20, '-');         // 20 dashes
```

Let's look the most useful ones.

1. In Java, `s1` is a *null* string. (That is, it a `String` variable which contains the special value `null`, which cannot be used. Unlike Java, in C++, it is the **empty string**.
2. `s2` **explicitly** converts a **string literal** (character array) to a C++ `string` object. String literals, such as `"hello"` are **not** `string` objects, as they are in Java. Instead, they are **pointers** to a single character at the beginning of the literal.
3. `s3`, the syntax you are probably most comfortable with, **implicitly** converts a C-string literal to a C++ `string` object.
4. Produces a `string` that is **a copy** of the `string s3`.
5. A `string` initialized with **a sequence** of `char` literals.
6. Produces a `string` object from a **raw string literal**. Raw string literals begin with `R"(` and end with `)"`. Inside you may store **any** character without using escape sequences.
7. Produces a `string` made of `20 '-'` characters. Note that `char` literals use single quotes, just as they do in Java. Python does not use the `char` type. Note that **you must use parentheses** for this constructor, not braces.

The `{}` and the `()` may often be used interchangably. However, for `s5`, you **must** use the braces `{}`, and for `s7` you **must** use parentheses `()`. In C++98, you must use parentheses, not braces, and `s5` and `s6` will not work at all. These constructors, and raw strings were not added until C++11.

C++14 added C++ string literals, which is a regular C-string literal, with an `s` suffix, like `"hello"s`. This is no longer a pointer, but a full-fledged C++ `string` object, as in Java.