string_view

Marshall Clow
Qualcomm

mclow@qti.qualcomm.com

Twitter: @mclow
Occasional Blog: https://cplusplusmusings.wordpress.com/

Is this what Bjarne was talking about on Monday?

No.

What is `string_view`?

"The class template `basic_string_view` describes an object that can refer to a constant contiguous sequence of char-like objects."

from the Library Fundamentals TS

What does it look like?

A `string_view` has almost exactly the same interface as a constant std::string

How is it implemented?

A `string_view` is conceptually a pointer and a length.

What's that "non-owning" bit again?

A `string_view` does not manage the storage that it refers to.

Lifetime management is up to the user.

When would you use a `string_view` instead of a string?

- Passing as a parameter to a “pure” function
- Returning from a function
- A reference to part of a long-lived data structure

Example #1

```
string extract_part(const string &bar) {
    return bar.substr (2, 3);
}

if (extract_part("ABCDEFG").front() == 'C')
{ /* do something */ }
```

Why do we care about temporary strings?

- They’re not small (3 pointers)
- They can cause memory allocation
  - SSO mitigates this somewhat
- You have to copy the data

Example #1
with string_view

```
string_view extract_part(string_view bar) {
    return bar.substr (2, 3);
}

if (extract_part("ABCDEFG").front() == 'C')
{ /* do something */ }
```

Example #2

```
void legacy_call (const char *xxx)
{ … }

legacy_call (“Hi Mom”);
string foo;
legacy_call (foo.c_str());
```

Example #2

```
void legacy_call(const string &xxx)
{ … }

legacy_call (“Hi Mom”);
string foo;
legacy_call (foo.c_str());
```

Example #2
with string_view

```
void legacy_call(string_view xxx)
{ … }

legacy_call (“Hi Mom”);
string foo;
legacy_call (foo.c_str());
// or
legacy_call (foo);
```

A string_view is constructible from

- A std::string
- A {pointer, length}
- A null-terminated string
  - This requires traversing the string to find the length.

string_view interoperates with std::string

- You can compare string_view to
  - a std::string
  - a null-terminated string
- You can construct a string from a string_view (using to_string())
- string_view has all the same methods as string (find, find_first_of, find_first_not_of, etc)

Additional functionality

- remove_prefix, remove_suffix

Drawbacks of string_view

- Lifetime management
- Not null-terminated

Where can I get string_view?

- std::experimental::string_view in libc++, libstdc++ (very recently)
- Boost has boost::string_ref, which is based on an earlier version of the proposal. This will be updated very soon (next week?)

Future Directions

- array_view
  - Complicated because of multiple dimensions.
- Ranges
  - A more general solution; not limited to contiguous sequences.

Questions?

Discussion of spam filtering (if time)