Gmail 黃詠翔 <jamesbond0705@gmail.com>

An Incorrect Statement and typos (version 2022-10-30)

黃詠翔 <jamesbond0705@gmail.com> 2023年2月19日 晚上7:56
草稿收件者：cppstd20@josuttis.com

Dear Mr. Josuttis,

In the pdf version **2022-10-30,**

19.1 New Types for Non-Type Template Parameters
(Page 631, line -6 and Page 635, line -4)
According to C++ named requirements: LiteralType (since C++11) - cppreference.com **(5th bullet's** 1st and last sub-bullets), I suggest this correction:

s/ has a constexpr constructor,-~~**no copy/move constructor, no destructor**~~, **no copy/move constructor or destructor,**
/ has a constexpr constructor **that is not a** copy/move constructor, **and has a** constexpr destructor, /
(2 places, **repeated and incorrect sentences**)

Typos:

5.3 Concepts for Iterators and Ranges
(Page 103, std::output_iterator)

1. s/ assign values of **typeT**./assign values of **type T**./ (Missing a blank)
2. s/ std::indirectly_writable<**Pos, I**> is satisfied/ std::indirectly_writable<**Pos, T**> is satisfied/ (incorrect type name)

Best regards,

Yung-Hsiang Huang

(Part 18) Typos in pdf version 2022-10-23

黃詠翔 <jamesbond0705@gmail.com> 2022年11月2日 中午12:01
草稿

Dear Mr. Josuttis,

This is an update to my previous typo report from **2022-10-23** to **2022-10-30.**

I found some new possible typos (about 28 items) in the pdf version **2022-10-30.**

3.1 Motivating Example of Concepts and Requirements
(Page 37, line -1)
s/ standard library**,**/ standard library**:**/ (change comma to colon or period.)

6.6 Summary of All Container Idioms Broken By Views
(Page 168, line 1)
s/ idioms that~~ we~~ **we** can count/

7.3.2 std::common_iterator
(Page 180)
s/ you can use this type function ~~be able~~ for calling **these algorithms**:/

7.6 Range Algorithms
(Page 192)

1. s/ the iterator is **no valid** iterator/ the iterator is **not a valid** iterator/
2. Table 7.11's **in_fun_result**/ in, **out**/ in, **fun**/ (cf. ISO's [algorithms.results]/1)

8.3.3 Owning View
(Page 216)
s/ owning_view<std::vector<std::string>>~~>~~/ (**2 places**, remove the last ">")

8.4.4 IStream View
(Page 233)
table/ Content: Generator of a range with elements read from **an input stream**/

(Page 234)
s/ over **a~~n~~ string** stream/

(Page 235)

1. s/ for (const auto **elem& e** : coll)/ for (const auto**& elem** : coll)/
2. s/ pos iterates over the view **va**,/

8.5.3 Drop View
(Page 249)
s/ when the range **a filter view** refers to is modified./ when the range **a drop view** refers to is modified./

(Page 250)

1. **subtitle**/ Drop View and const/ Drop View**s** and const/
2. s/ for (const auto **elem& e** : coll)/ for (const auto**& elem** : coll)/

8.5.4 Drop-While View
(Page 254)
s/ when the range **a filter view** refers to is modified./ when the range **a drop-while view** refers to is modified./

(Page 255)

1. **subtitle**/ Drop-While View and const/ Drop-While View**s** and const/
2. s/ for (const auto **elem& e** : coll)/ for (const auto**& elem** : coll)/

8.5.5 Filter View
(Page 262)

1. **subtitle**/ Filter View and const/ Filter View**s** and const/
2. s/ for (const auto **elem& e** : coll)/ for (const auto**& elem** : coll)/

8.6.1 Transform View
(Page 265)
table/ Content: ... values of all **element**s/

(Page 268)

1. s/ the transform **views do**~~**es**~~ not/
2. s/ a **take** view is only common/ a **transform** view is only common/

8.7.1 Reverse View
(Page 282)

1. **subtitle**/ Reverse View and const/ Reverse View**s** and const/
2. s/ for (const auto **elem& e** : coll)/ for (const auto**& elem** : coll)/

8.8.1 Spilt and Lazy-Spilt View
(Page 288)
subtitle/ Split View**s** and const/

8.8.2 Join View
(Page 292)
subtitle/ Join **View** and const/ Join **View**s and const/

(Page 293)

1. s/ When we ~~**use**~~ join the elements of the array/ (Alternative: "When we **use std::views::join** ~~the elements of~~ **for** the array")
2. s/ when ~~**we create**~~ we pass..., calling printConstCall() is an error./
3. s/ the following **doe** not compile/ the following **does** not compile/
4. s/ // ERROR if std::next() used/ // ERROR if std::next() **is** used/

Best regards,

Yung-Hsiang Huang

Gmail 黃詠翔 <jamesbond0705@gmail.com>

(Part 13) Typos in pdf version 2022-08-09

黃詠翔 <jamesbond0705@gmail.com> 2022年11月2日 上午11:54
收件者: cppstd20@josuttis.com

Dear Mr. Josuttis:

This is an update (from **2022-09-16** to **2022-10-30**) to my previous typo report. Hope it will help you to locate them easily:

I found **7** possible typos in Chapter 12 of your book C++20 (pdf version: **2022-10-30**).

Typos:

12.1 Motivation for std::jthread
(Page 403)
code snippet/ neither t.join() nor t.detach() are called

(Page 405)

1. code snippet/ wait for **the** first thread/ (Missing the definite article "*the*")
2. code snippet/ wait for **the** second thread/
3. s/ **move-assign** it in the try clause/ (Missing a dash)

(Page 407, line -3)
s/ the **thread and the last**.... **is**/ the **thread and the last**.... **are**/

12.3 std::jthread in Detail
(Page 418)
s/ the API of **std::thread/** the API of **std::jthread/**

(Page 420)
code snippet/ **pool**.push_back/ **threads**.push_back/

Latex Issue:

The tilde symbol for destructors in Table 12.2, 12.3 & 12.4 seem to be ill-formed. I found code snippet in this stackexchange solution may be helpful.
By the way, these three tables have the following common typos

1. s/ **copy-assigns**/ (Missing a dash)
2. s/ **move-assigns**/

Best,

Yung-Hsiang Huang

(Part 14) Typos in pdf version 2022-08-09

黃詠翔 <jamesbond0705@gmail.com> 2022年11月2日 上午11:55
收件者: cppstd20@josuttis.com

Dear Mr. Josuttis:

This is an update (from **2022-09-16** to **2022-10-30**) to my previous typo report. Hope it will help you to locate them easily:

I found 21 possible typos to your book C++20 (pdf version: **2022-10-30**). They are mainly about Chapter 13, 15.9 and 20.

13.1 Thread Synchronization with Latches and Barriers
(Page 423)
s/ **The moment**..., all threads.../ **When**..., all threads.../ (*Reason*: no conjunction in the origin sentences)

(Page 426)
s/ std::latch allReady**{numThreads}**;/ std::latch allReady **= 10**;/ (Since it causes narrowing conversion)

(Page 429)

1. **Weird sentence in first bullet** "*We initialize... computation**:** the number of tags/tasks**:***". Comparing with Page 403 line 1. I suggest the whole sentence should become "We initialize... **allDone** ***with the number of tags/tasks*** **to print all**... computation**:**"
2. s/ The **API** of barriers also **provide**s/ (Missing "s")

13.2 Semaphores
(Page 434)
s/ If you want to sleep or do something **else if** you cannot get a resource/ If you want to sleep or do something **else when** you cannot get a resource/ *(change "if" to "when")*

(Page 435)

1. s/ threads that **wait** the longest time/ threads that **wait for** the longest time/
2. s/ **which one** of multiple threads... **are** woken-up/ **which one** of multiple threads... **is** woken-up/

13.3 Extensions for Atomic Types
(Page 441)
s/ not use **the same** atomic_ref<> **object~~s~~**/ (redundant "s")

(Page 442)
s/ Atomic **reference** have the following restrictions/ Atomic **reference**s ..../ (Missing "s")

(Page 443-444, ***lib/atomicshared.cpp***)

1. It might **be better to #include <vector>** for latter use in main() function.
2. s/ // populate list with elements from **10** threads/ // populate list with elements from **100** threads/

(Page 449)

1. s/ Here is **a** example/ Here is **an** example/

2. footnote/ CppCon ~~**2029**~~ **2019**/
3. In **lib/atomicticket.cpp**, **remove** the **unused header** file **#include <semaphore>**.

13.4 Synchronized Output Streams
(Page 452)
In the beginning paragraph, it might be better to **mention that we should include the header <syncstream> // for std::osyncstream**

(Page 453)

1. lib/syncstream.cpp, add **// for std::osyncstream** after #include**<syncstream>.**
2. s/ the synchronized **output buffer** synchronizes the output with other **output** to a synchronized **output buffer** / the synchronized **output stream** synchronizes the output with other **output**~~**s**~~ to a synchronized **output buffer**/ *(Reason:* the original text "**buffer** synchronizes... to a **buffer**" looks weird.)

15.9 Concurrent Use of Coroutines
(Page 544)
s/ Before we end the program~~.~~**,** the pool/ (change the middle period to comma)

(Page 551, first bullet)
s/ Each coroutine ~~**knows**~~ has a pointer/ (it seems to be a redundant verb)

(Page 555)
s/ **any signal** that... ~~**it**~~ **results** in a/ (*Reason*: Since "any signal" is the subject of this sentence, we should remove the pronoun "it")

20.3.2 unwrap_reference<> and unwrap_ref_decay_t
(Page 643)
After For example:, add **std::string s;** (the definition of the variable s).

20.6 Afternotes
(Page 650, line 1 & 3)

[隱藏引用文字]

(Part 15) Typos in pdf version 2022-08-24

黃詠翔 <jamesbond0705@gmail.com> 2023年3月7日 晚上11:22
草稿收件者：cppstd20@josuttis.com

黃詠翔 <jamesbond0705@gmail.com> 於 2022年11月2日 週三 上午11:58寫道：

Dear Mr. Josuttis,

This is an update to my previous typo report from version **2022-08-24** to **2022-10-30.** Hope it will help you to locate them easily. (Items for Chapter 7 & 8 have been moved to Part 18 for uniformity)**:**

For double checking, I also compare this pdf version with my earlier processed feedback emails (Part 9~12) and form a section (reworded previous typo reports) in the end.

2. Reworded Previous Issues & Typos report:

14.4.1 Awaiters
(Page 492)
s/ In this awaiter, we trace **when which** function/ In this awaiter, we trace **when and which** function/ (Reason: it's weird to see consecutive Five Ws.)

14.4.3 Resuming Sub-Coroutines
(Page 497)
As I test earlier, the last output snippet **misses the following 2 green lines after coro(): Part1**

```
MAIN: callCoro() suspended
      coro(): Part2
```

15.6 Allocating Memory for the Coroutine Frame
(Page 525, beginning sentences of 15.6 & 15.6.1)
s/ their **state**s/ (2 places, Missing a "s")

17.5.1 Capturing this and *this

(Page 596, **line -4, comment of l2**)
s/ **deprecated** (val and name by ref.)/
OK (val and name by reference)
Reason:
I checked the original proposal p0806r2, [expr.prim.lambda.capture]/2 and [depr.capture.this]. I didn't find that capturing this pointer by [&] is deprecated in C++20.
In fact, the "Revision History" and "Discussion" sections of original proposal has the following sentences:

"P0806R1: Remove the `[&]` part;"
"The implicit capture of `*this` via `[&]` continues to be idiomatic. An earlier revision of this paper also proposed its deprecation."

5.3.1 Concepts for Ranges and Views
(Page 100, viewable_range)
In the "*Requires*" of std::ranges::viewable_range, it only states ***std::range<Rg> is satisfied***. But the following long list is probably missed in the "*Requires*", which corresponds to the earlier sentence "The concept is satisfied if Rg either is already a view or an lvalue of a range or a movable rvalue or a range but no initializer list":

```
((ranges::view<std::remove_cvref_t<T>> &&
     std::constructible_from<std::remove_cvref_t<T>, T>) ||
    (!ranges::view<std::remove_cvref_t<T>> &&
     (std::is_lvalue_reference_v<T> ||
      (std::movable<std::remove_reference_t<T>> && !/*is-initializer-list*/<T>))));
```

cf. cppreference

5.5.2 Concepts for Incrementable Types

(Page 112)

std::weakly_incrementable:

According to p2325r3 (the 2nd changes in section *Wording*)(this paper is also mentioned in Page 99's footnote2), we should **remove** *std::default_initializable<T>* in "Requires". (Or add a footnote?)

Best regards,

Yung-Hsiang Huang