C++ Execution Model In Five Minutes

Bryce Adelstein Lelbach

Expressions

- An *expression* is a sequence of operators and operands that specifies a computation.

  [expr.pre] p1 s2

- *Subexpressions* are a part of a larger expression.

  [intro.execution] p3, p4

- *Full expressions* are not subexpressions.

  [intro.execution] p5

```
{
  T a = 2;
  // Full: `T::T(int)` call

  a += 17 + 42;
  // Full: `a += 17 + 42`
  // Sub: `17 + 42`

  if (a == T(61)) {}
  // Full includes:
  // lvalue-to-rvalue conversion
  // int-to-bool conversion
  // `operator==(T, T)` call

} // Full: destruction of a
```

Evaluations

- ***Evaluation*** of an **expression** includes **value computations** and the initiation of **side effects**:

  [intro.execution] p7 s2

- ***Side effects*** change the environment:
  - Reading a `volatile` object or modifying any object.
  - Calling a library I/O function.
  - Calling a function that does any of the above.

  [intro.execution] p7 s1

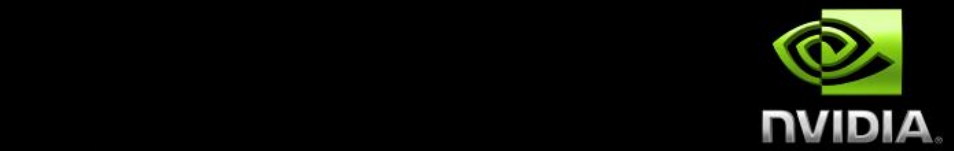
- ***Value computations*** are pure.

  [intro.execution] p7 s2

- Completion of the **execution** of an **evaluation** does not imply completion of its **side effects**.

  [intro.execution] p7 s3

Sequenced Before



- Given any two **evaluations** A and B…
- If A is ***sequenced before*** B, then the **execution** of A shall precede the **execution** of B.

[intro.execution] p8 s2

- The **sequenced before** relationship is…
- **Asymmetric**: A is **sequenced before** B does not imply that B is **sequenced before** A.

[intro.execution] p8 s1

- **Transitive**: If A is **sequenced before** B and B is **sequenced before** C, then A is **sequenced before** C.

[intro.execution] p8 s1

Sequenced Before

- Given any two evaluations A and B…
- If A is *sequenced before* B, then the execution of A shall precede the execution of B.

[intro.execution] p8 s2

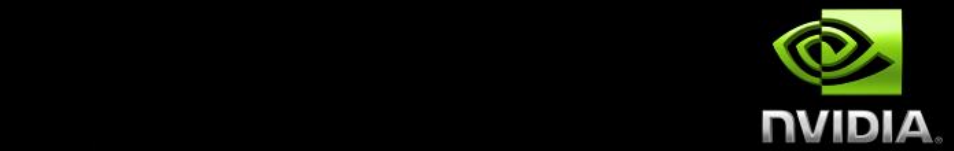
- If A and B are *unsequenced*, then A is not *sequenced before* B and B is not *sequenced before* A.

[intro.execution] p8 s3

- If A and B are *indeterminately sequenced*, then either A is sequenced before B or B is sequenced before A, but it is unspecified which. E.g. A and B are not interleaved.

[intro.execution] p8 s4

Function Evaluation



- When calling a function F…
- Every expression in the body of F is sequenced after the value computations and side effects associated with every:
  - Argument expression of the function:

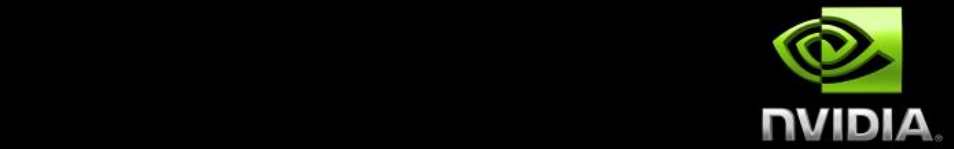
    ```
    f(a + b);
    // `a + b` is an argument expression.
    ```

  - The expression designating the function:

    ```
    (*f)(x);
    // `(*f)` is the expression designating the function.
    ```

[intro.execution] p11 s1

Function Evaluation



- When calling a function F…
- The expression designating the function is sequenced before the argument expressions.

[expr.call] p8 s1

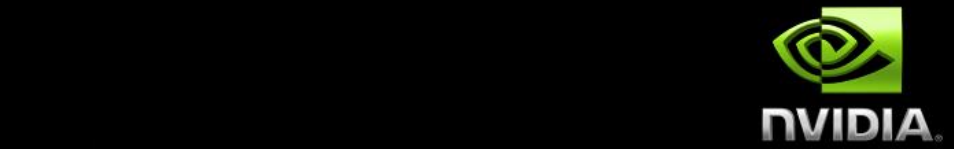
- The value computations and side effects of an argument expression is indeterminately sequenced with all other argument expressions.

[expr.call] p8 s2

- Every evaluation within F and every evaluation not within F are indeterminately sequenced.

[intro.execution] p11 s2

Function Evaluation



- The value computations and side effects of the operands to the following operators are sequenced right to left:
  - E2[E1] [expr.sub] p1 s6
  - E2.*E1 and E2->*E1 [expr.mptr.oper] p3 s3
  - E2 @= E1 [expr.ass] p1 s5
- For all binary operators, the value computations (but not the side effects) operands are unsequenced with respect to each other.
- For || and &&, the value computations and side effects of the operands to the following operators are sequenced right to left.

Happens Before

- Given any two **evaluations** A and B…
- If A ***synchronizes with*** B, then the **side effects** of A are seen by B.

[intro.races] p6

- **Asymmetric**: A **synchronizes with** B does not imply that B **synchronizes with** A.
- Ways to achieve **synchronizes with**:
  - Acquire/release.
  - Mutex lock/unlock.
  - Thread create/join.

Happens Before

- Given any two evaluations A and B…
- If A *happens before* B:
  - A is sequenced before B.
  - A synchronizes with B.
  - For some evaluation X, A happens before X and X happens before B.

[intro.races] p7, p8, p9, p10

- A happens before B does not imply the execution of A completes before the execution of B completes.
- The execution of A completing before the execution of B completes does not imply that A happens before B.

Happens Before

- **Happens before** describes arbitrary concatenations of **sequenced before** and **synchronizes with**.