Advanced Rust (2026): Unsafe Boundaries and Soundness

Lecture 5

Lukáš Hozda

Spring 2026

MFF CUNI

Unsafe Model

What `unsafe` Means

- `unsafe` does not disable compiler checks globally.
- It allows specific operations requiring extra invariants.
- Soundness is now your obligation.

The Five Unsafe Capabilities

- Dereference raw pointers.
- Call unsafe functions.
- Access/modify mutable statics.
- Implement unsafe traits.
- Access union fields.

Invariant-First Design

- Write invariants before code.
- `unsafe` block should be tiny and auditable.
- Safe wrapper should enforce preconditions.

Raw Pointer Example

```rust
fn read_first(ptr: *const i64, len: usize) -> Option<i64> {
    if ptr.is_null() || len == 0 {
        return None;
    }
    unsafe { Some(*ptr) }
}
```

slice::from_raw_parts

- Requires valid pointer + length.
- Memory must be initialized and immutable for lifetime.
- Aliasing and lifetime mistakes become UB.

```rust
fn as_slice<'a>(ptr: *const u8, len: usize) -> &'a [u8] {
    unsafe { std::slice::from_raw_parts(ptr, len) }
}
```

MaybeUninit<T>

- Correct tool for partially initialized buffers.
- Avoid uninitialized reads.
- Convert to initialized type only after full init.

`NonNull<T>`

- Non-null raw pointer wrapper.
- Useful in custom collections.
- Does not prove aliasing or lifetime correctness.

Unsafe Trait Contracts

- `Send` / `Sync` manual impls are high-risk.
- Must reason about all reachable state.
- One wrong impl can invalidate whole crate safety.

FFI Boundaries

- `extern "C"` defines ABI contract.
- Layout compatibility and ownership transfer rules must be explicit.

UB Is Not A Panic

- UB can look correct during tests.
- UB invalidates optimizer assumptions globally.
- “Works on my machine” is not evidence.

Review Workflow

- For every unsafe block:
  - state required invariants
  - state why they hold
  - state how caller can break them
- Add targeted tests for boundary behavior.

Strategy

- Keep unsafe core minimal.
- Expose safe, narrow API.
- Move complexity to compile-time checks where possible.