1 `main` — MIR Walkthrough

Purpose: TODO: Describe why this walkthrough exists

1.1 Source Context

```
fn main() {
    let a = 42;
    let b = &a;
    let c = &b;

    assert!(**c == 42);
}
```

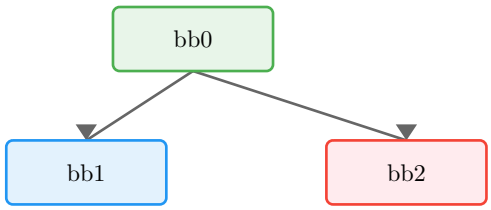
1.2 Function Overview

- **Function:** `main`
- **Basic blocks:** 3
- **Return type:** `()`
- **Notable properties:**
  - Contains panic path
  - Introduces borrows
  - Has conditional branches

1.3 Locals

Local	Type	Notes
0	`()`	Return place
1	`i32`	
2	`&i32`	
3	`&&i32`	
4	`i32`	
5	`!`	
6	`&i32`	

1.4 Control-Flow Overview



1.5 Basic Blocks

1.5.1 bb0 — entry

Entry point of the function.

MIR	Annotation
`\_1 = 42`	Load constant
`\_2 = &\_1`	Shared borrow
`\_3 = &\_2`	Shared borrow

`\_6 = copy\_deref((\*\_3))`	
`\_4 = (\*\_6)`	Copy value
`→ switch(move \_4) \[42→bb1; else→bb2\]`	Branch on move _4

1.5.2 bb1 — return / success

Normal return path.

MIR	Annotation
`→ return`	Return from function

1.5.3 bb2 — panic path

Panic/diverging path.

MIR	Annotation
`→ \_5 = panic(\[16 bytes\])`	Call panic

1.6 Key Observations

TODO: Add bullet points summarizing what this MIR teaches

-
-

1.7 Takeaways

TODO: One or two sentences to generalize this example