

# 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:** () (0 bytes, align 1)
- **Notable properties:**
  - Contains panic path
  - Introduces borrows
  - Has conditional branches

## 1.3 Locals

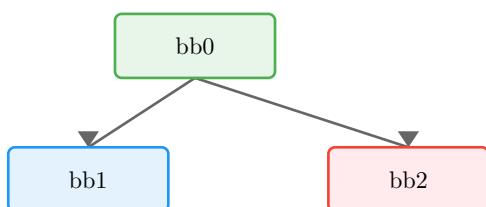
Local	Type	Notes
0	() (0 bytes, align 1)	Return place
1	Int(I32)	
2	&i32 (8 bytes, align 8)	
3	Int(I32)	
4	()	

## 1.4 Borrows

#	Borrow	Kind	Created At	Borrowed
0	_2	&	bb0[1]	_1

Borrows are tracked conservatively: active from creation until reassignment or scope end.

## 1.5 Control-Flow Overview



## 1.6 Basic Blocks

### 1.6.1 bb0 — entry

Entry point of the function.

MIR	Annotation
-----	------------

<code>_1 = 42</code>	Load constant
<code>_2 = &amp;_1</code>	Shared borrow
<code>_3 = (*_2)</code>	Copy value
<code>→ switch(_3) [42-&gt;bb1; else-&gt;bb2]</code>	Branch on <code>_3</code>

### 1.6.2 bb1 — return / success

*Normal return path.*

MIR	Annotation
<code>→ return</code>	Return from function

### 1.6.3 bb2 — panic path

*Panic/diverging path.*

MIR	Annotation
<code>→ _4 = panic([16 bytes])</code>	Call panic

## 1.7 Key Observations

TODO: Add bullet points summarizing what this MIR teaches

- 
- 

## 1.8 Takeaways

TODO: One or two sentences to generalize this example

