

Rust in 3 Weeks

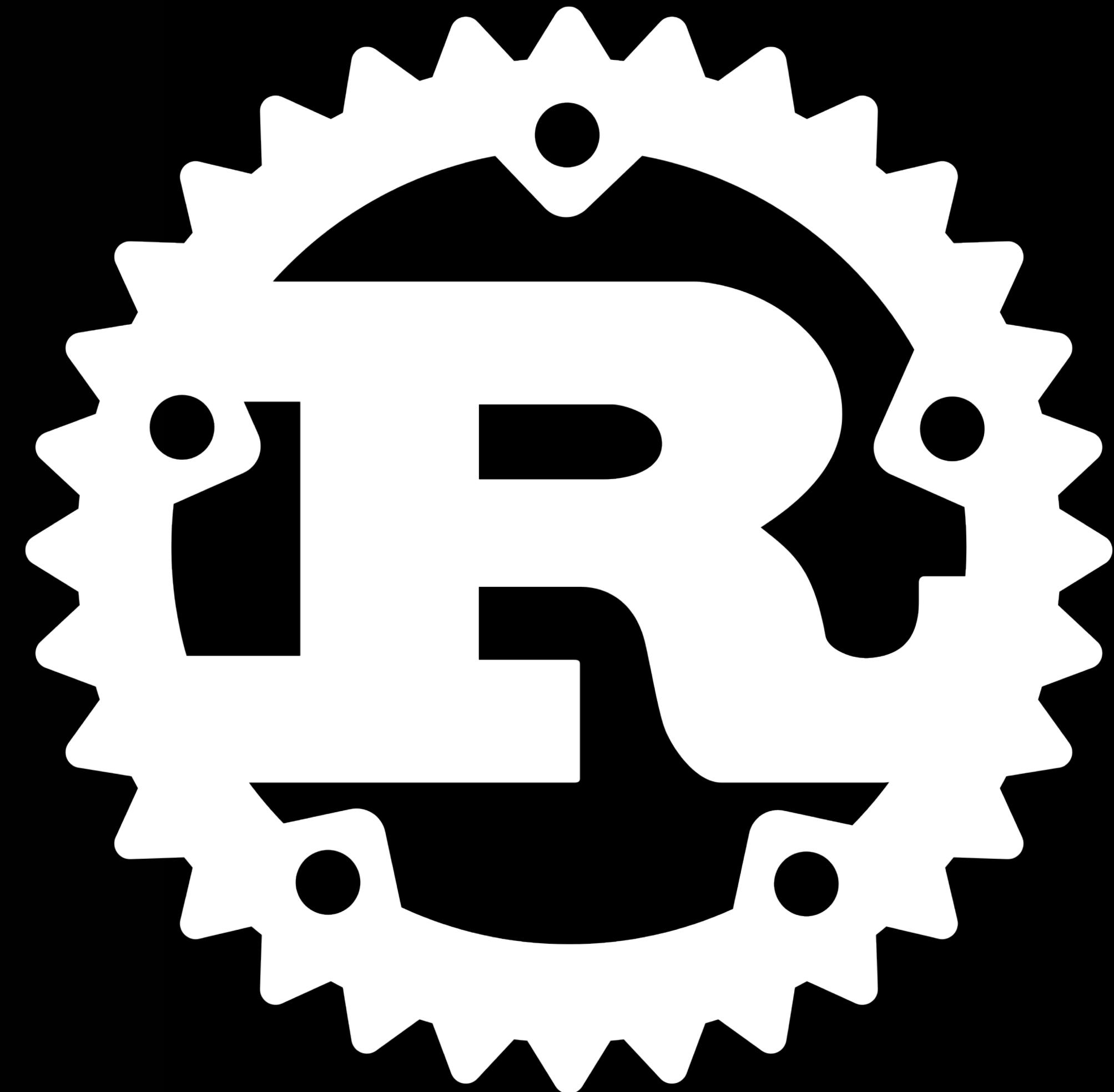
Week 2: Practical Rust

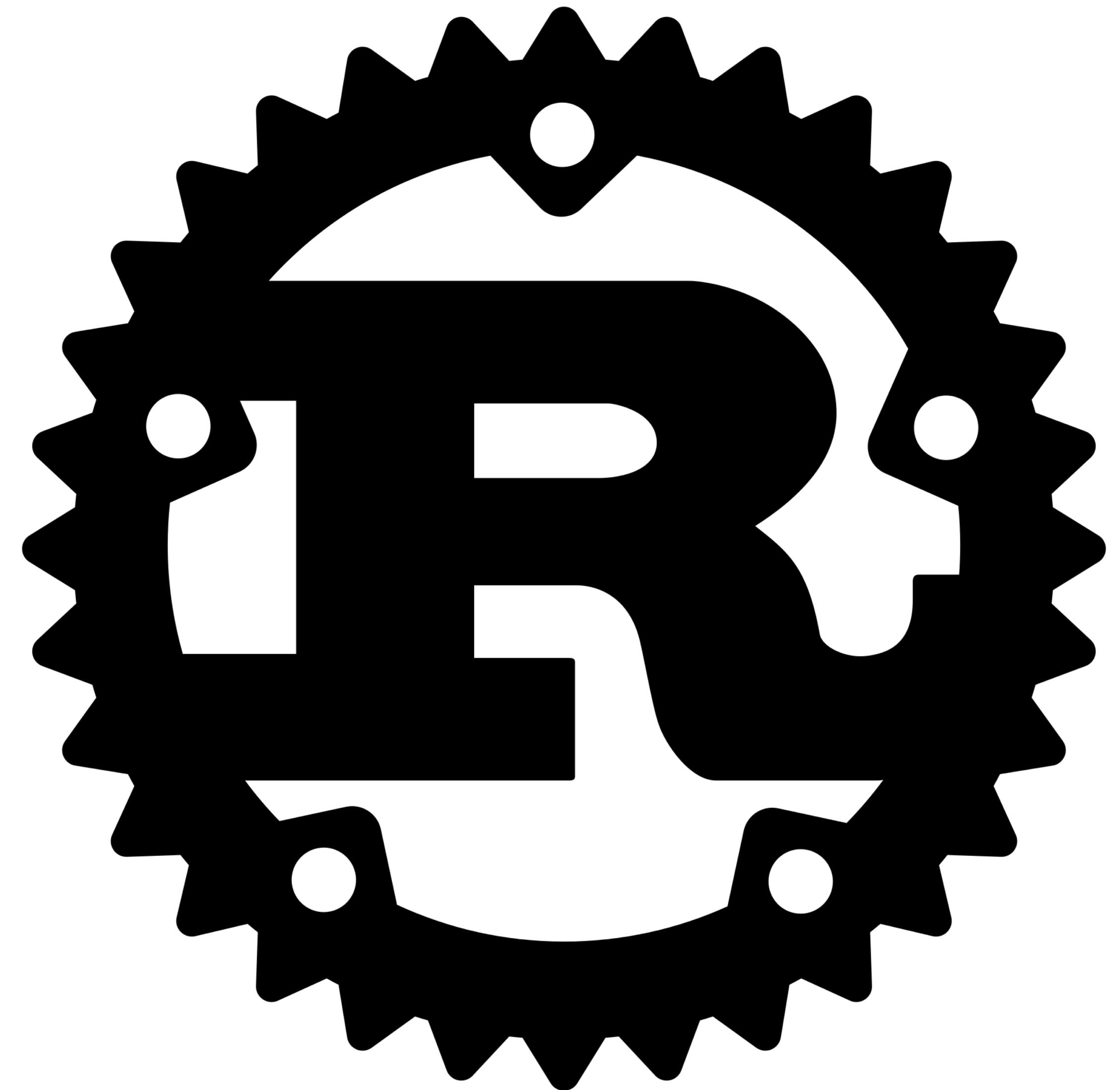


GitHub

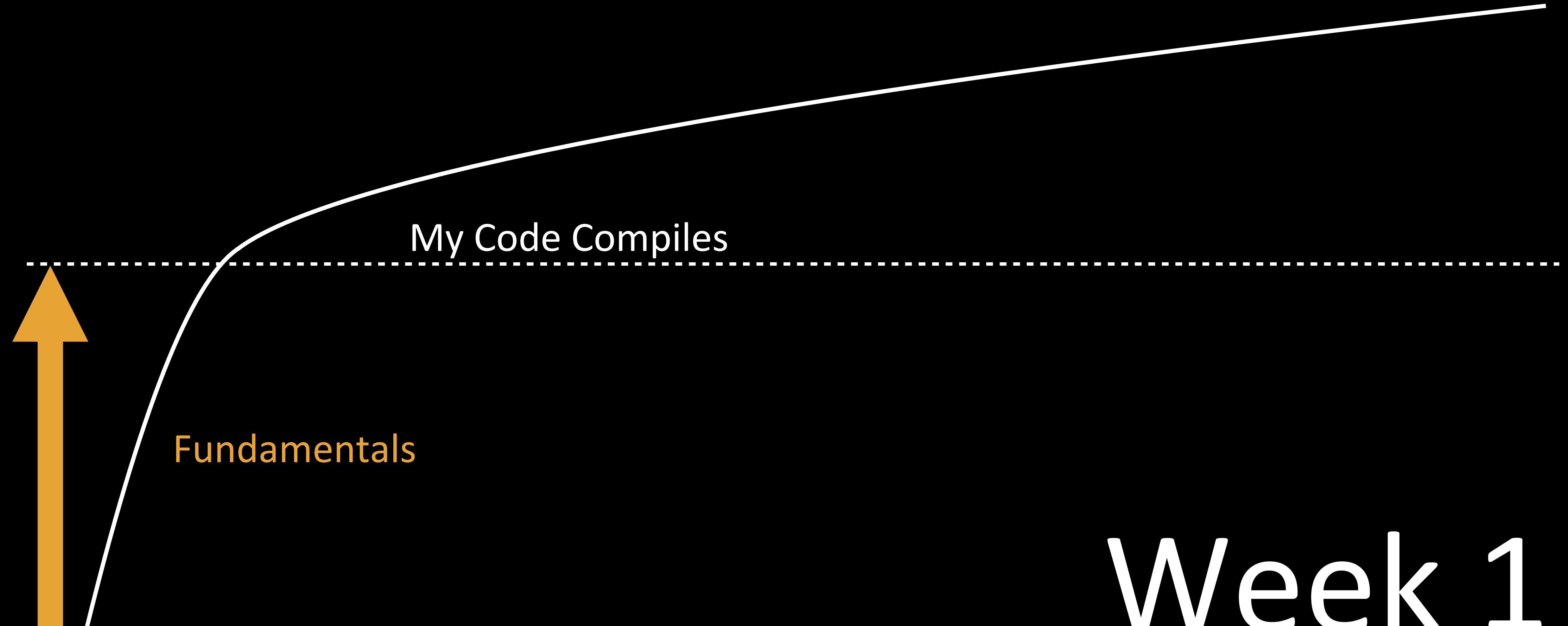


VIDEOGAMES

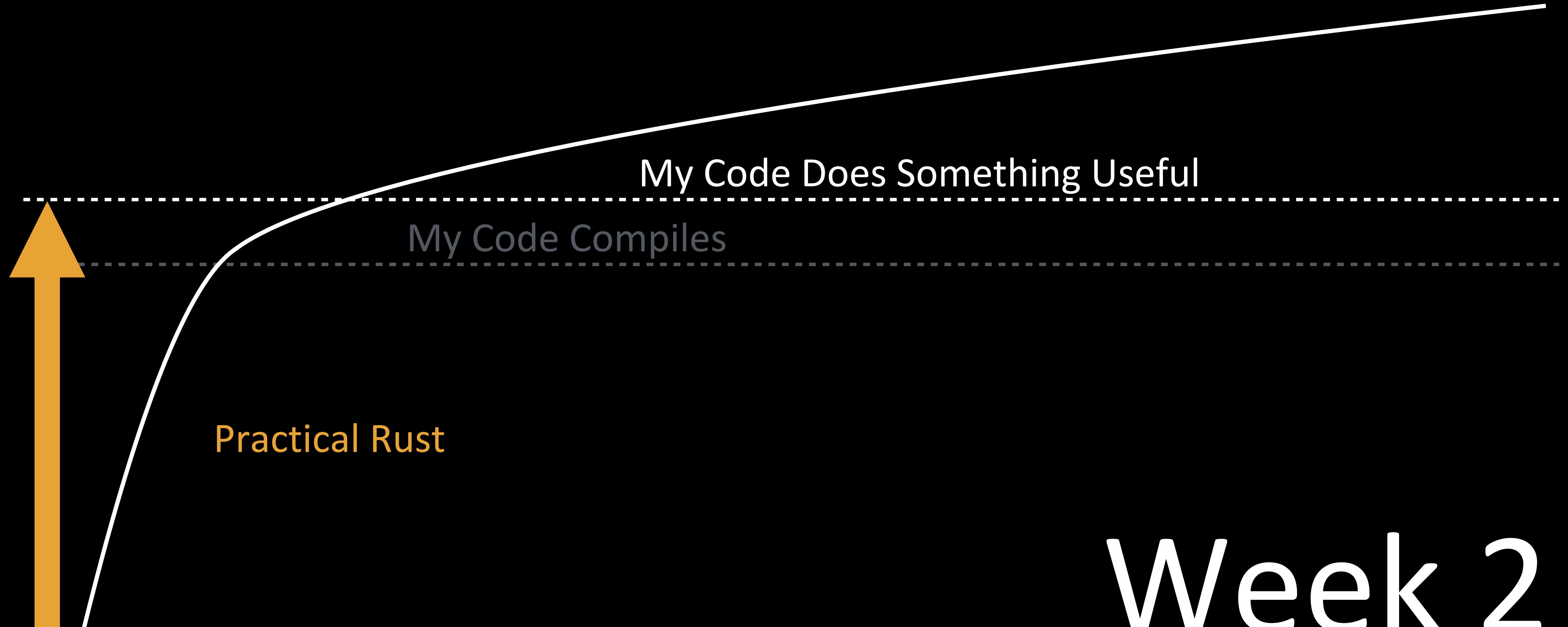




Rust's Steep Learning Curve



Rust's Steep Learning Curve





Repository for Exercises

https://github.com/CleanCut/ultimate_rust2

Idiomatic Rust

idiomatic

adjective

1. expressions that are natural to a native speaker
2. appropriate to the style of a particular group

Automation!

rustfmt

fmt -> format

rustfmt

cargo fmt

```
struct Thing{field: i32}
```

```
enum
```

```
Choice {
```

```
A, B, C
```

```
}
```

```
fn main ()
```

```
{
```

```
    println!("Hello, world!");
```

```
}
```

```
struct Thing{field: i32}
```

```
enum
```

```
Choice {
```

```
A, B, C
```

```
}
```

```
fn main ()
```

```
{
```

```
    println!("Hello, world!");
```

```
}
```

cargo fmt



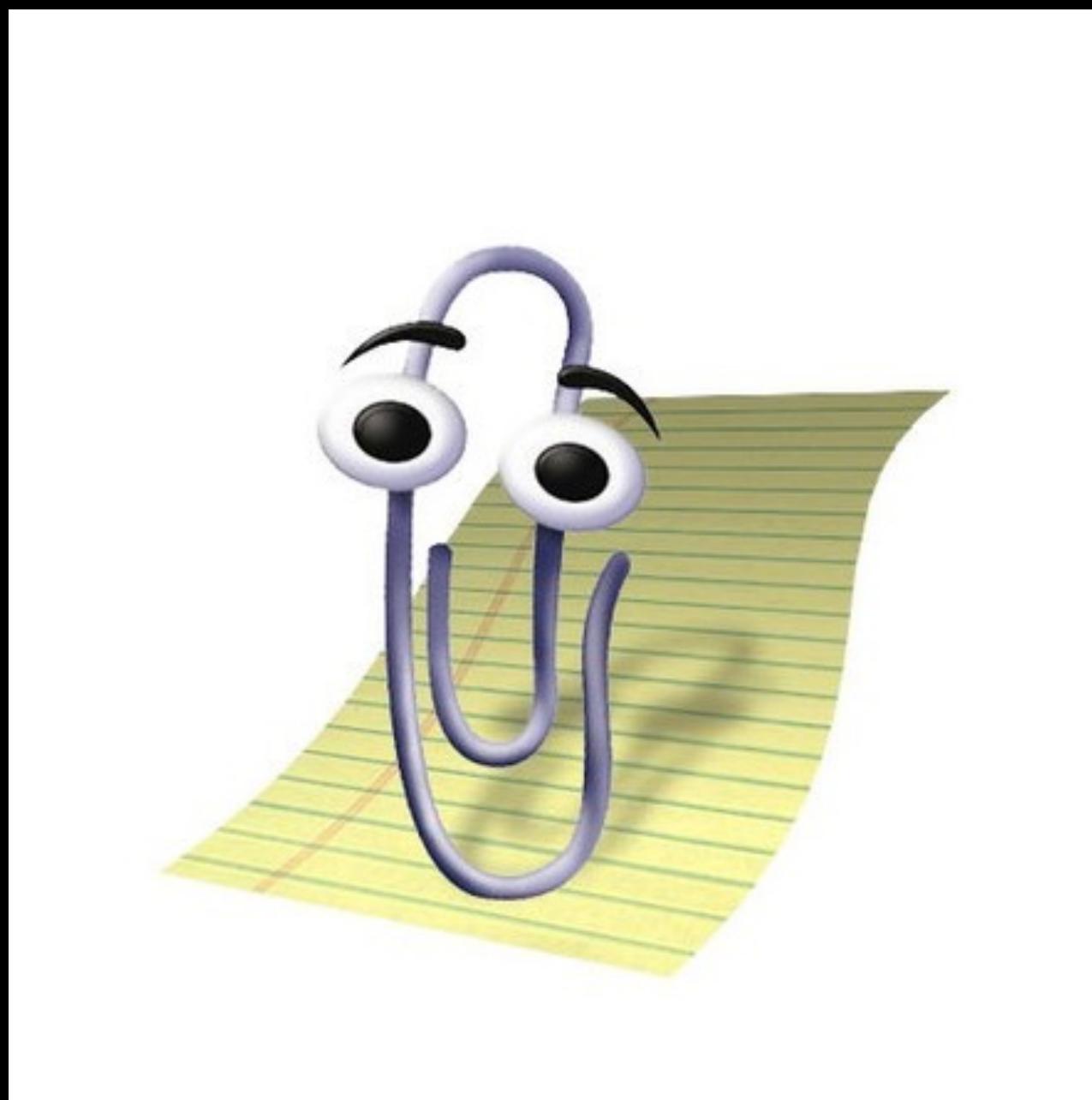
Editor: Format On Save

Format a file on save. A formatter must be available, the file must not be saved after delay, and the editor must not be shutting down.

.rustfmt.toml

clippy

clippy



Noooooooooooooo...!

cargo clippy

1. Style

```
fn number() -> i32 {  
    return 5;  
}
```

```
warning: unneeded `return` statement
```

```
--> src/main.rs:6:5
```

```
6 |     return 5;
  | ^^^^^^^^^^ help: remove `return`: `5`
```

```
= note: `#[warn(clippy::needless_return)]` on by default
= help: for further information visit https://rust-lang.github.io/rust-clippy/master/index.html#needless\_return
```

```
fn number() -> i32 {
```

```
    5
```

```
}
```

2. Correctness

```
loop {  
    break;  
}
```

3. Complexity

let x = (5);

let x = 5;

warning: this function has too many arguments (8/7)

--> src/main.rs:5:1

|

5 | / fn sum(
6 | | first: u8,
7 | | second: u8,
8 | | third: u8,

...

13 | | eighth: u8,
14 | |) -> u8 {

| | ^

|
= **note:** `#[warn(clippy::too_many_arguments)]` on by default

= **help:** for further information visit https://rust-lang.github.io/rust-clippy/master/index.html#too_many_arguments

warning: this function has too many arguments (8/7)

--> src/main.rs:5:1

|

5 | / fn sum(
6 | | first: u8,
7 | | second: u8,
8 | | third: u8,

... |

13 | | eighth: u8,
14 | |) -> u8 {

| |_____ ^

|

= **note**: `#[warn(clippy::too_many_arguments)]` on by default

= **help**: for further information visit https://rust-lang.github.io/rust-clippy/master/index.html#too_many_arguments

```
#[allow(clippy::too_many_arguments)]
```



```
#[allow(clippy::too_many_arguments)]
```

```
#[allow(clippy::too_many_arguments)]
```

```
#[allow(clippy::too_many_arguments)]
```

```
#[allow(clippy::too_many_arguments)]
```


warning: this function has too many arguments (8/7)

--> src/main.rs:5:1

|

5 | / fn sum(
6 | | first: u8,
7 | | second: u8,
8 | | third: u8,

... |

13 | | eighth: u8,
14 | |) -> u8 {

| |_____ ^

|
= note: `#[warn(clippy::too_many_arguments)]` on by default

= help: for further information visit https://rust-lang.github.io/rust-clippy/master/index.html#too_many_arguments

4. Performance

```
let len = iterator.clone().collect::<Vec<i32>>().len();
```

```
let len = iterator.count();
```



clippy lints X

[Google Search](#)

[I'm Feeling Lucky](#)

Help kids [Be Internet Awesome](#) with new tools for educators

Exercise 1

exercises/idiomatic

Questions?

Documentation

cargo doc

cargo doc --no-deps --open

`target/doc/packagename/index.html`

```
pub const PUZZLE_PIECES: u32 = 42;
```



```
pub const PUZZLE_PIECES: u32 = 42;
```

///

pub *const* PUZZLE_PIECES: *u32* = 42;

```
/// Number of pieces in the puzzle  
pub const PUZZLE_PIECES: u32 = 42;
```

/// Number of pieces in the puzzle

///

/// This is a separate paragraph.

pub *const* PUZZLE_PIECES: *u32* = 42;

Crate puzzles

[-][src]

Constants

PUZZLE_PIECES Number of pieces in the puzzle

Crate puzzles

[-][src]

Constants

PUZZLE_PIECES Number of pieces in the puzzle

Constant puzzles::PUZZLE_PIECES

[-][src]

```
pub const PUZZLE_PIECES: u32 = 42;
```

[-] Number of pieces in the puzzle other stuff

This is a separate paragraph.

/// Number of pieces in the puzzle

///

/// # History

///

/// This is a separate paragraph.

/// - Clickable link: [`PUZZLE_PIECES`]

/// - We tried `7`, but this is better

pub *const* PUZZLE_PIECES: *u32* = 42;

/// Number of pieces in the puzzle

///

/// # History

///

/// This is a separate paragraph.

/// - Clickable link: [`PUZZLE_PIECES`]

/// - We tried `7`, but this is better

pub *const PUZZLE_PIECES: u32 = 42;*

Constant puzzles::PUZZLE_PIECES

[-][src]

```
pub const PUZZLE_PIECES: u32 = 42;
```

[-] Number of pieces in the puzzle

History

This is a separate paragraph.

- Clickable link: PUZZLE_PIECES
- We tried 7, but this is better

/// Number of pieces in the puzzle

///

/// # History

///

/// This is a separate paragraph.

/// - Clickable link: [`PUZZLE_PIECES`]

/// - We tried `7`, but this is better

pub *const* PUZZLE_PIECES: *u32* = 42;

/// Number of pieces in the puzzle

///

/// # History

///

/// This is a separate paragraph.

/// - Clickable link: [`PUZZLE_PIECES`]

/// - We tried `7`, but this is better

pub *const PUZZLE_PIECES: u32 = 42;*

/// Number of pieces in the puzzle

///

/// # History

///

/// This is a separate paragraph.

/// - Clickable link: [`PUZZLE_PIECES`]

/// - We tried `7`, but this is better

- Clickable link: PUZZLE_PIECES

/// Number of pieces in the puzzle

///

/// # History

///

/// This is a separate paragraph.

/// - Clickable link: `PUZZLE_PIECES`

/// - We tried `7`, but this is better

- Clickable link: PUZZLE_PIECES

/// Number of pieces in the puzzle

///

/// # History

///

/// This is a separate paragraph.

/// - Clickable link: [`PUZZLE_PIECES`]

/// - We tried `7`, but this is better

- Clickable link: PUZZLE_PIECES

/// Number of pieces in the puzzle

///

/// # History

///

/// This is a separate paragraph.

/// - Clickable link: [PUZZLE_PIECES]

/// - We tried `7`, but this is better

- Clickable link: PUZZLE_PIECES

/// Number of pieces in the puzzle

///

/// # History

///

/// This is a separate paragraph.

/// - [Clickable link](PUZZLE_PIECES)

/// - We tried `7`, but this is better

- Clickable link

/// [Spawn a thread](std::thread::spawn)

Outer

Inner

//!

Crate **puzzles**

[\[-\]](#)[\[src\]](#)

[\[-\]](#) Hi! I'm your friendly Rust Puzzle Library documentation. Please come in, sit down, and have a cup of hot chocolate!

Constants

`PUZZLE_PIECES` Number of pieces in the puzzle



Crate puzzles

Version 0.1.0

See all puzzles's items

Constants

Crates

puzzles



All crates ▾

Click or press 'S' to search, '?' for more options

?



Crate **puzzles**

[-][src]

[-] Hi! I'm your friendly Rust Puzzle Library documentation. Please come in, sit down, and have a cup of hot chocolate!

Constants

PUZZLE_PIECES Number of pieces in the puzzle

Struct puzzles::Puzzle

[\[-\]](#)[\[src\]](#)

[\[+\]](#) Show declaration

[\[-\]](#) This is a Puzzle!

Fields

`num_pieces: u32`

[\[-\]](#) Number of pieces

`name: String`

[\[-\]](#) Descriptive name

Implementations

[\[-\]](#) `impl Puzzle`

[\[src\]](#)

[\[-\]](#) `pub fn new() -> Self`

[\[src\]](#)

Make a new puzzle!

Exercise 2

exercises/docs

Questions?

Publishing

**crates.io** Click or press 'S' to search...[Browse All Crates](#) | [Docs ▾](#) | [Log in with GitHub](#)

The Rust community's crate registry

 [Install Cargo](#) [Getting Started](#)

Instantly publish your crates and install them. Use the API to interact and find out more information about available crates. Become a contributor and enhance the site with your work.

6,948,695,391

Downloads

**60,809**

Crates in stock



New Crates

mu_alb
v0.2.0

Most Downloaded

rand

mu_runtime
v0.2.0

millionaire
v0.1.0

Just Updated

stencila-schema
v1.6.1

imap
v3.0.0-alpha.4

terra-rust-api
v0.1.13

Permanent

Unique

highlander

Unique

**crates.io** Click or press 'S' to search...[Browse All Crates](#) | [Docs ▾](#) | [Log in with GitHub](#)

The Rust community's crate registry

 [Install Cargo](#) [Getting Started](#)

Instantly publish your crates and install them. Use the API to interact and find out more information about available crates. Become a contributor and enhance the site with your work.

6,948,695,391
Downloads



60,809
Crates in stock



New Crates

mu_alb
v0.2.0

Most Downloaded

rand

Just Updated

stencila-schema
v1.6.1

mu_runtime
v0.2.0

syn

millionaire
v0.1.0

libc

imap
v3.0.0-alpha.4

terra-rust-api
v0.1.13

**crates.io**

🔍 Click or press 'S' to search...

[Browse All Crates](#) | [Docs ▾](#) [Log in with GitHub](#)

The Rust community's crate registry

[⬇️ Install Cargo](#)[🏁 Getting Started](#)

Instantly publish your crates and install them. Use the API to interact and find out more information about available crates. Become a contributor and enhance the site with your work.

6,948,695,391

Downloads

**60,809**

Crates in stock



New Crates

mu_alb
v0.2.0

Most Downloaded

rand

Just Updated

stencila-schema
v1.6.1**mu_runtime**
v0.2.0

syn

millionaire
v0.1.0

libc

imap
v3.0.0-alpha.4**terra-rust-api**
v0.1.13

Browse All Crates | Docs ▾ |  Nathan Stocks ▾

[Dashboard](#)
[Account Settings](#)
[Owner Invites](#)
[Sign Out](#)

Sort by  Relevance ▾

API Access

[New Token](#)

If you want to use package commands from the command line, you will need to login with `cargo login (token)` using one of the tokens listed below.

When working in shared environments, supplying the token on the command line could expose it to prying eyes. To avoid this, enter `cargo login` and supply your token when prompted.

cargo login

```
[package]
name = "rusty_engine"
version = "0.11.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
description = "Learn Rust with a simple, cross-platform, 2D game engine."
edition = "2018"
homepage = "https://github.com/CleanCut/rusty_engine"
repository = "https://github.com/CleanCut/rusty_engine"
readme = "README.md"
keywords = ["game", "engine", "graphics", "audio", "rusty"]
categories = ["game-engines"]
license = "MIT OR Apache-2.0"
```

```
[package]
name = "rusty_engine"
version = "0.11.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
description = "Learn Rust with a simple, cross-platform, 2D game engine."
edition = "2018"
homepage = "https://github.com/CleanCut/rusty_engine"
repository = "https://github.com/CleanCut/rusty_engine"
readme = "README.md"
keywords = ["game", "engine", "graphics", "audio", "rusty"]
categories = ["game-engines"]
license = "MIT OR Apache-2.0"
```

```
[package]
name = "rusty_engine"
version = "0.11.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
description = "Learn Rust with a simple, cross-platform, 2D game engine."
edition = "2018"
homepage = "https://github.com/CleanCut/rusty\_engine"
repository = "https://github.com/CleanCut/rusty\_engine"
readme = "README.md"
keywords = ["game", "engine", "graphics", "audio", "rusty"]
categories = ["game-engines"]
license = "MIT OR Apache-2.0"
```

```
[package]
name = "rusty_engine"
version = "0.11.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
description = "Learn Rust with a simple, cross-platform, 2D game engine."
edition = "2018"
homepage = "https://github.com/CleanCut/rusty_engine"
repository = "https://github.com/CleanCut/rusty_engine"
readme = "README.md"
keywords = ["game", "engine", "graphics", "audio", "rusty"]
categories = ["game-engines"]
license = "MIT OR Apache-2.0"
```

```
[package]
name = "rusty_engine"
version = "0.11.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
description = "Learn Rust with a simple, cross-platform, 2D game engine."
edition = "2018"
homepage = "https://github.com/CleanCut/rusty_engine"
repository = "https://github.com/CleanCut/rusty_engine"
readme = "README.md"
keywords = ["game", "engine", "graphics", "audio", "rusty"]
categories = ["game-engines"]
license = "MIT OR Apache-2.0"
```

```
[package]
name = "rusty_engine"
version = "0.11.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
description = "Learn Rust with a simple, cross-platform, 2D game engine."
edition = "2018"
homepage = "https://github.com/CleanCut/rusty_engine"
repository = "https://github.com/CleanCut/rusty_engine"
readme = "README.md"
keywords = ["game", "engine", "graphics", "audio", "rusty"]
categories = ["game-engines"]
license = "MIT OR Apache-2.0"
```

```
[package]
name = "rusty_engine"
version = "0.11.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
description = "Learn Rust with a simple, cross-platform, 2D game engine."
edition = "2018"
homepage = "https://github.com/CleanCut/rusty_engine"
repository = "https://github.com/CleanCut/rusty_engine"
readme = "README.md"
keywords = ["game", "engine", "graphics", "audio", "rusty"]
categories = ["game-engines"]
license = "MIT OR Apache-2.0"
```

```
[package]
name = "rusty_engine"
version = "0.11.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
description = "Learn Rust with a simple, cross-platform, 2D game engine."
edition = "2018"
homepage = "https://github.com/CleanCut/rusty_engine"
repository = "https://github.com/CleanCut/rusty_engine"
readme = "README.md"
keywords = ["game", "engine", "graphics", "audio", "rusty"]
categories = ["game-engines"]
license = "MIT OR Apache-2.0"
```

```
[package]
name = "rusty_engine"
version = "0.11.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
description = "Learn Rust with a simple, cross-platform, 2D game engine."
edition = "2018"
homepage = "https://github.com/CleanCut/rusty_engine"
repository = "https://github.com/CleanCut/rusty_engine"
readme = "README.md"
keywords = ["game", "engine", "graphics", "audio", "rusty"]
categories = ["game-engines"]
license = "MIT OR Apache-2.0"
```

cargo publish

```
$ cargo publish
Updating crates.io index
Packaging rusty_engine v0.12.0 (/Users/cleancut/rust/rusty_engine)
Verifying rusty_engine v0.12.0 (/Users/cleancut/rust/rusty_engine)
Compiling libc v0.2.97
Compiling proc-macro2 v1.0.27
[snip lots and lots of compiling dependencies]
Compiling bevy_sprite v0.5.0
Compiling bevy_pbr v0.5.0
Compiling bevy_wgpu v0.5.0
Compiling bevy_gltf v0.5.0
Compiling bevy_text v0.5.0
Compiling bevy_ui v0.5.0
Compiling bevy_internal v0.5.0
Compiling bevy v0.5.0
Compiling bevy_kira_audio v0.5.0
Compiling rusty_engine v0.12.0 (/Users/cleancut/rust/rusty_engine/target/package/rusty_engine-0.12.0)
Finished dev [unoptimized + debuginfo] target(s) in 1m 08s
Uploading rusty_engine v0.12.0 (/Users/cleancut/rust/rusty_engine)
```

\$



rusty_engine 0.12.0

[Follow](#)

Learn Rust with a simple, cross-platform, 2D game engine.

#audio #graphics #engine #rusty #game

[Readme](#)[10 Versions](#)[Dependencies](#)[Dependents](#)[Settings](#)

Rusty Engine

Rusty Engine is a simple, 2D game engine for those who are learning Rust. Create simple prototypes using straightforward Rust code, without any advanced game engine concepts.

[Questions](#), [bug reports](#), and contributions are most welcome!

If you would like to support this project consider [sponsoring me] on GitHub. ❤

Metadata

4 minutes ago

MIT OR Apache-2.0

10.3 MB

Install

Add the following line to your Cargo.toml file:

rusty_engine = "0.12.0"



Click or press 'S' to search, '?' for more options...



Crate rusty_engine

Version 0.12.0

See all rusty_engine's items

Modules

Modules

actor

audio

Create rusty_engine

[][

[] Asset Licenses

All assets included with this game engine have the appropriate license described and linked to in a `README.md` file in the same directory as the source files. In most cases, the license is [CC0 1.0 Universal](#)-meaning you may do whatever you wish with the asset.

One notable exception is some of the music files, which are under a different license and include specific attribution requirements that must be met in order to be used legally when distributed. Please see [this README.md file](#) for more information.

Modules

Questions?

Closures & Iterators

Closures

| params | { expr1 ; expr2 }

| params | expr

```
let add = |x, y| x + y;
```

```
add(1, 2); // returns 3
```

$$|x, \textcolor{teal}{y}|_{x+y}$$

|| x + y

|| {}

```
let s = "🍓";
```

```
let f = || {  
    println!("{}", s);  
};
```

```
f(); // prints 🍓
```

```
let s = "🍓";
```

```
let f = move || {  
    println!("{}", s);  
};
```

```
f(); // prints 🍓
```

```
let s = "🍓";
```

```
let f = move || {  
    println!("{}", s);  
};
```

```
f(); // prints 🍓
```

```
let s = "🍓";  
let s2 = s.clone();  
let f = move || {  
    println!("{}", s2);  
};  
  
f(); // prints 🍓
```

Fn

FnMut

FnOnce

std::ops

Iterators

```
let v = vec![6, 7, 8, 9];
```

```
for num in v {
```

```
    println!("{}", num);
```

```
}
```



```
let v = vec![6, 7, 8, 9];  
  
for num in v.into_iter() {  
    println!("{}", num);  
}
```

```
let v = vec![6, 7, 8, 9];
```

```
for num in v.into_iter() {  
    println!("{}", num);  
}
```

Intolterator

.into_iter()

.into_iter()

Takes Ownership

```
let v = vec![6, 7, 8];  
v.into_iter().for_each(|num| println!("{}", num));
```

```
let v = vec![6, 7, 8];  
v.into_iter().for_each(|num| println!("{}", num));
```

```
let v = vec![6, 7, 8];  
v.into_iter().for_each(|num| println!("{}", num));
```

```
let v = vec![6, 7, 8];  
v.into_iter().for_each(|num| println!("{}", num));
```

```
let v = vec![6, 7, 8];  
v.into_iter().for_each(|num| println!("{}", num));
```

Iterator Adaptors

```
let v = vec![6, 7, 8];
```

```
let total: i32 = v  
    .into_iter()           // 6, 7, 8  
    .map(|x| x * 3)       // 18, 21, 24
```

```
let v = vec![6, 7, 8];
```

```
let total: i32 = v  
    .into_iter()          // 6, 7, 8  
    .map(|x| x * 3)      // 18, 21, 24  
    .filter(|y| *y % 2 == 0) // 18, 24
```

```
let v = vec![6, 7, 8];
```

```
let total: i32 = v  
    .into_iter()          // 6, 7, 8  
    .map(|x: i32| x * 3) // 18, 21, 24  
    .filter(|y: &i32| *y % 2 == 0) // 18, 24
```

```
[+] pub fn filter<P>(self, predicate: P) -> Filter<Self, P> ⓘ  
where  
    P: FnMut(&Self::Item) -> bool,
```

[src]

Creates an iterator which uses a closure to determine if an element should be yielded.

Given an element the closure must return `true` or `false`. The returned iterator will yield only the elements for which the closure returns true.

Examples

Basic usage:

```
let a = [0i32, 1, 2];  
  
let mut iter = a.iter().filter(|x| x.is_positive());  
  
assert_eq!(iter.next(), Some(&1));  
assert_eq!(iter.next(), Some(&2));  
assert_eq!(iter.next(), None);
```

Run

Because the closure passed to `filter()` takes a reference, and many iterators iterate over references, this leads to a possibly confusing situation, where the type of the closure is a double reference:


```
let v = vec![6, 7, 8];

let total: i32 = v
    .into_iter()           // 6, 7, 8
    .map(|x: i32| x * 3)   // 18, 21, 24
    .filter(|y: &i32| *y % 2 == 0) // 18, 24
```

```
let v = vec![6, 7, 8];
```

```
let total: i32 = v  
    .into_iter()           // 6, 7, 8  
    .map(|x: i32| x * 3)   // 18, 21, 24  
    .filter(|y: &i32| *y % 2 == 0) // 18, 24
```

```
let v = vec![6, 7, 8];
```

```
let total: i32 = v
```

```
.into_iter()           // 6, 7, 8
```

```
.map(|x: i32| x * 3) // 18, 21, 24
```

```
.filter(|y: &i32| *y % 2 == 0) // 18, 24
```

Iterator Consumer

```
let v = vec![6, 7, 8];
```

```
v.into_iter()           // 6, 7, 8  
.map(|x| x * 3)       // 18, 21, 24  
.filter(|y| *y % 2 == 0) // 18, 24  
.for_each(|z| println!("{}", z));
```

```
let v = vec![6, 7, 8];
```

```
let total = v  
  
.into_iter()          // 6, 7, 8  
  
.map(|x| x * 3)      // 18, 21, 24  
  
.filter(|y| *y % 2 == 0) // 18, 24  
  
.sum();               // ERROR!
```



```
let v = vec![6, 7, 8];
```

```
let total: i32 = v  
    .into_iter()          // 6, 7, 8  
    .map(|x| x * 3)      // 18, 21, 24  
    .filter(|y| *y % 2 == 0) // 18, 24  
    .sum();              // 42
```

```
let v = vec![6, 7, 8];  
  
v.into_iter()          // 6, 7, 8  
.map(|x| x * 3)      // 18, 21, 24  
.filter(|y| *y % 2 == 0) // 18, 24  
.sum()               // ERROR!
```

error[E0283]: type annotations needed

--> src/main.rs:31:10

```
|  
31 |     .sum();  
|     ^^^ cannot infer type for type parameter `S` declared on the associated function `sum`  
|
```

= note: cannot satisfy `_: Sum<i32>`

help: consider specifying the type argument in the method call

```
|  
31 |     .sum::<S>();  
|     ^^^^^^
```

error: aborting due to previous error

For more information about this error, try `rustc --explain E0283`.

error[E0283]: type annotations needed

--> src/main.rs:31:10

```
|  
31 |     .sum();  
|     ^^^ cannot infer type for type parameter `S` declared on the associated function `sum`  
|  
= note: cannot satisfy `_: Sum<i32>`
```

help: consider specifying the type argument in the method call

```
|  
31 |     .sum::<S>();  
|     ^^^^^^
```

error: aborting due to previous error

For more information about this error, try `rustc --explain E0283`.

error[E0283]: type annotations needed

--> src/main.rs:31:10

```
|  
31 |     .sum();  
|     ^^^ cannot infer type for type parameter `S` declared on the associated function `sum`  
|
```

= note: cannot satisfy `_: Sum<i32>`

help: consider specifying the type argument in the method call

```
|  
31 |     .sum::<S>();  
|     ^^^^^^
```

error: aborting due to previous error

For more information about this error, try `rustc --explain E0283`.

fish

::<>

fish

::<>

bubbles...

turbo fish!

::<>

::<*i32*>

.sum::<i32>

.sum::<i32>()

```
let v = vec![6, 7, 8];  
  
v.into_iter()          // 6, 7, 8  
.map(|x| x * 3)      // 18, 21, 24  
.filter(|y| *y % 2 == 0) // 18, 24  
.sum::<i32>()        // 42
```

```
let v = vec![6, 7, 8];

let v2 = v

    .into_iter()          // 6, 7, 8
    .map(|x| x * 3)      // 18, 21, 24
    .filter(|y| *y % 2 == 0) // 18, 24
    .collect();           // Error!
```

```
let v = vec![6, 7, 8];
```

```
let v2 = v  
    .into_iter()          // 6, 7, 8  
    .map(|x| x * 3)      // 18, 21, 24  
    .filter(|y| *y % 2 == 0) // 18, 24  
    .collect();           // Error!
```



```
let v = vec![6, 7, 8];
```

```
let v2: Vec<i32> = v  
    .into_iter()          // 6, 7, 8  
    .map(|x| x * 3)      // 18, 21, 24  
    .filter(|y| *y % 2 == 0) // 18, 24  
    .collect();           // vec![18, 24]
```

```
let v = vec![6, 7, 8];
```

```
let v2: Vec<_> = v
    .into_iter()          // 6, 7, 8
    .map(|x| x * 3)      // 18, 21, 24
    .filter(|y| *y % 2 == 0) // 18, 24
    .collect();           // vec![18, 24]
```

```
let v = vec![6, 7, 8];
```

```
let v2 = v  
  
.into_iter()          // 6, 7, 8  
  
.map(|x| x * 3)      // 18, 21, 24  
  
.filter(|y| *y % 2 == 0) // 18, 24  
  
.collect::<Vec<_>>(); // vec![18, 24]
```



```
let v = vec![6, 7, 8];
```

```
let v2: Vec<_> = v
    .into_iter()          // 6, 7, 8
    .map(|x| x * 3)      // 18, 21, 24
    .filter(|y| *y % 2 == 0) // 18, 24
    .collect::<Vec<_>>(); // vec![18, 24]
```

`v.into_iter() // consumes v, returns owned items`

`v.iter() // returns immutable references`

`v.iter_mut() // returns mutable references`

v.into_iter() // consumes v, returns owned items

v.**iter()** // returns immutable references

v.iter_mut() // returns mutable references

`v.into_iter() // consumes v, returns owned items`

`v.iter() // returns immutable references`

`v.iter_mut() // returns mutable references`

`for num in v.iter()`

`v.into_iter() // consumes v, returns owned items`

`v.iter() // returns immutable references`

`v.iter_mut() // returns mutable references`

`v.into_iter() // consumes v, returns owned items`

`v.iter() // returns immutable references`

`v.iter_mut() // returns mutable references`

`for num in v.iter_mut()`

`v.into_iter() // consumes v, returns owned items`

`v.iter() // returns immutable references`

`v.iter_mut() // returns mutable references`

```
v.into_iter()  for _ in v  
v.iter()      // returns immutable references  
v.iter_mut() // returns mutable references
```

```
v.into_iter()  for _ in v
```

```
v.iter()       for _ in &v
```

```
v.iter_mut() // returns mutable references
```

v.into_iter() for _ in v

v.iter() for _ in &v

v.iter_mut() for _ in &mut v

Exercise 3

exercises/closures_iterators

Questions?

10 Minute Break

Common Traits

What can implement a Trait?

struct
enum

closure
function

struct

enum

closure

function

Derivable Traits

#[derive(Debug)]

Debug

```
println!("{:?}", puzzle); // Debug
```

```
println!("{:?}", puzzle); // Debug
```

```
println!("{:#?}", puzzle); // Pretty debug
```



```
#[derive(Debug)]  
pub struct Puzzle {  
    pub num_pieces: u32,  
    pub name: String,  
}
```

```
#[derive(Debug)]  
pub struct Puzzle {  
    pub num_pieces: u32,  
    pub name: String,  
}  
println!("{:?}", puzzle);
```



```
#[derive(Debug)]  
pub struct Puzzle {  
    pub num_pieces: u32,  
    pub name: String,  
}
```

println!("{:#?}", puzzle);

Clone

```
#[derive(Clone, Debug)]  
pub struct Puzzle {  
    pub num_pieces: u32,  
    pub name: String,  
}
```

```
#[derive(Clone, Debug)]  
pub struct Puzzle {  
    pub num_pieces: u32,  
    pub name: String,  
}  
  
let puzzle2 = puzzle.clone();
```

Copy

```
#[derive(Clone, Debug)]  
pub struct Puzzle {  
    pub num_pieces: u32,  
    pub name: String,  
}
```

```
#[derive(Clone, Debug)]  
pub struct Puzzle {  
    pub num_pieces: u32,  
    pub name: String,  
}
```

```
pub enum PuzzleType {
```

Jigsaw,

```
}
```

```
##[derive(Copy)]  
pub enum PuzzleType {  
    Jigsaw,  
}
```

```
#[derive(Clone, Copy)]
pub enum PuzzleType {
    Jigsaw,
}
```

Traits you Implement

1. Bring the Trait Into Scope

Trait std::cmp::PartialEq

1.0.0 [-][src]

[+] Show declaration

[−] Trait for equality comparisons which are **partial equivalence relations**.

This trait allows for partial equality, for types that do not have a full equivalence relation. For example, in floating point numbers `NaN != NaN`, so floating point types implement `PartialEq` but not `Eq`.

Formally, the equality must be (for all `a`, `b`, `c` of type `A`, `B`, `C`):

- **Symmetric:** if `A: PartialEq` and `B: PartialEq<A>`, then
`a == b implies b == a`; and

std::cmp::PartialEq

2. Boilerplate

```
36 37 impl PartialEq for Puzzle {}  
38  
39  
40
```

A screenshot of a code editor showing a tooltip for a partially implemented trait implementation. The code on the left shows a file with lines 36 through 40. Line 37 contains the code `impl PartialEq for Puzzle {}`. A yellow lightbulb icon is positioned above the line number 36. A tooltip box is open over the brace at the end of the line 37 code. The tooltip has two items: "Implement missing members" (highlighted in blue) and "Implement default members".

2. Boilerplate

```
36
37     impl PartialEq for Puzzle {}  
38
39
40
```

A screenshot of a code editor showing a tooltip for the missing `PartialEq` implementation. The code on the left shows a partial implementation of `PartialEq` for `Puzzle`. A tooltip box is open over the closing brace of the implementation block, containing two options: "Implement missing members" (highlighted in blue) and "Implement default members".

2. Boilerplate

```
impl PartialEq for Book {  
    fn eq(&self, other: &Self) -> bool {  
        self.isbn == other.isbn  
    }  
}
```

3. Implementation

Default

#[derive(Default)]

```
impl Default for Puzzle {
```

```
    fn default() -> Self {
```

```
        todo!()
```

```
}
```

```
}
```

```
impl Default for Puzzle {  
    fn default() -> Self {  
        panic!("not yet implemented");  
    }  
}
```

```
impl Default for Puzzle {  
    fn default() -> Self {  
        Puzzle {  
            num_pieces: PUZZLE_PIECES,  
            name: "Forest Lake".to_string(),  
        }  
    }  
}  
}
```

```
let puzzle = Puzzle {  
    num_pieces: 3000,  
    ..Default::default()  
};
```

```
let puzzle = Puzzle {  
    num_pieces: 3000,  
    ..Default::default()  
};
```

```
let puzzle = Puzzle {  
    num_pieces: 3000,  
    ..Default::default()  
};
```

```
let puzzle = Puzzle {  
    num_pieces: 3000,  
    ..Default::default()  
};
```

PartialEq / Eq

PartialEq / Eq

NaN != NaN

```
impl PartialEq for Puzzle {  
    fn eq(&self, other: &Self) -> bool {  
        todo!()  
    }  
}
```



```
impl PartialEq for Puzzle {  
    fn eq(self: &Self, other: &Self) -> bool {  
        todo!()  
    }  
}
```

```
impl PartialEq for Puzzle {  
    fn eq(self: &Self, other: &Self) -> bool {  
        todo!()  
    }  
}
```



```
impl PartialEq for Puzzle {  
    fn eq(self: &Puzzle, other: &Puzzle) -> bool {  
        todo!()  
    }  
}
```

```
impl PartialEq for Puzzle {  
    fn eq(&self, other: &Self) -> bool {  
        todo!()  
    }  
}
```

```
impl PartialEq for Puzzle {  
    fn eq(&self, other: &Self) -> bool {  
        (self.num_pieces == other.num_pieces) && (self.name == other.name)  
    }  
}
```

```
impl PartialEq for Puzzle {  
    fn eq(&self, other: &Self) -> bool {  
        (self.num_pieces == other.num_pieces) && (self.name == other.name)  
    }  
}
```

```
impl Eq for Puzzle {}
```

```
impl PartialEq for Puzzle {  
    fn eq(&self, other: &Self) -> bool {  
        (self.num_pieces == other.num_pieces) && (self.name == other.name)  
    }  
}
```

```
impl PartialEq for Puzzle {  
    fn eq(&self, other: &Self) -> bool {  
        (self.num_pieces == other.num_pieces) &&  
        (self.name.to_lowercase() == other.name.to_lowercase())  
    }  
}
```

From / Into

From<T> for U

Into<U> for T

From<T> for U
Into<String> for Puzzle

From<Puzzle> for String
Into<String> for Puzzle

```
impl From<Puzzle> for String {  
    fn from(_: Puzzle) -> Self {  
        todo!()  
    }  
}
```

```
impl From<Puzzle> for String {  
    fn from(puzzle: Puzzle) -> Self {  
        todo!()  
    }  
}
```

```
impl From<Puzzle> for String {  
    fn from(puzzle: Puzzle) -> Self {  
        puzzle.name  
    }  
}
```

```
impl From<Puzzle> for String {  
    fn from(puzzle: Puzzle) -> Self {  
        puzzle.name  
    }  
}
```

```
let puzzle = Puzzle::default();
```

```
impl From<Puzzle> for String {  
    fn from(puzzle: Puzzle) -> Self {  
        puzzle.name  
    }  
}
```

```
let puzzle = Puzzle::default();  
let s = String::from(puzzle);
```

```
impl From<Puzzle> for String {  
    fn from(puzzle: Puzzle) -> Self {  
        puzzle.name  
    }  
}
```

```
let puzzle = Puzzle::default();  
let t: String = puzzle.into();
```

```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```



```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```

```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```

```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```

```
let puzzle = Puzzle::default();
```

```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```

```
let puzzle = Puzzle::default();  
show(puzzle);
```

```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```

```
let puzzle = Puzzle::default();  
show(puzzle);  
// puzzle has been consumed :-(
```

```
impl From<Puzzle> for String {  
    fn from(puzzle: Puzzle) -> Self {  
        puzzle.name  
    }  
}
```

```
impl From<&Puzzle> for String {  
    fn from(puzzle: Puzzle) -> Self {  
        puzzle.name  
    }  
}
```

```
impl From<&Puzzle> for String {  
    fn from(puzzle: &Puzzle) -> Self {  
        puzzle.name  
    }  
}
```

```
impl From<&Puzzle> for String {  
    fn from(puzzle: &Puzzle) -> Self {  
        puzzle.name.clone()  
    }  
}
```

```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```

```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```

```
let puzzle = Puzzle::default();
```

```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```

```
let puzzle = Puzzle::default();  
show(&puzzle);
```

```
pub fn show<T: Into<String>>(s: T) {  
    println!("{}", s.into());  
}
```

```
let puzzle = Puzzle::default();  
show(&puzzle);  
// puzzle is still available!
```

```
impl From<&Puzzle> for String {  
    fn from(puzzle: &Puzzle) -> Self {  
        puzzle.name.clone()  
    }  
}
```

Exercise 4

exercises/traits

Questions?

Creating Errors



ERROR HANDLING ISN'T ALL ABOUT ERRORS

JANE LUSBY



Make an Error

Make an Error
for a library

Make an Error
for a library
that you publish

#1: enum

```
pub enum PuzzleError {  
    WontFit(u16),  
    MissingPiece,  
}
```

#2: Group Errors

```
pub enum PuzzleError {  
    WontFit(u16),  
    MissingPiece,  
}
```

[-]

[+] Expand attributes

```
pub enum ErrorKind {  
    NotFound,  
    PermissionDenied,  
    ConnectionRefused,  
    ConnectionReset,  
    ConnectionAborted,  
    NotConnected,  
    AddrInUse,  
    AddrNotAvailable,  
    BrokenPipe,  
    AlreadyExists,  
    WouldBlock,  
    InvalidInput,  
    InvalidData,  
    TimedOut,  
    WriteZero,  
    Interrupted,  
    Other,  
    UnexpectedEof,  
}
```

Enum std::io::SeekFrom

1.0.0 [-][src]

```
[-] pub enum SeekFrom {  
    Start(u64),  
    End(i64),  
    Current(i64),  
}
```

#3: Only YOUR Errors

```
pub enum FractalError {
```

SadSnowflake,

```
}
```

```
pub enum PuzzleError {
```

WontFit(u16),

MissingPiece,

```
}
```

```
pub enum FractalError {
```

SadSnowflake,

```
}
```

```
pub enum PuzzleError {
```

WontFit(u16),

MissingPiece,

```
}
```

```
pub enum FractalError {
```

SadSnowflake,

```
}
```

```
pub enum PuzzleError {
```

WontFit(u16),

MissingPiece,

```
}
```



```
pub enum FractalError {
```

SadSnowflake,

```
}
```



```
pub enum PuzzleError {
```

WontFit(u16),

MissingPiece,

PrettyImageless,

```
}
```

#4. Non-Exhaustive

```
#[non_exhaustive]
pub enum PuzzleError {
    WontFit(u16),
    MissingPiece,
}
```

```
// Nope!  
  
match error {  
    PuzzleError::WontFit(_) => {}  
    PuzzleError::MissingPiece => {}  
}
```

```
// Yes!  
  
match error {  
    PuzzleError::WontFit(_) => {}  
    PuzzleError::MissingPiece => {}  
    _ => {}  
}
```

#5. Debug + Display + Exception

pub trait *Error*: *Debug* + *Display*

```
#[non_exhaustive]
pub enum PuzzleError {
    WontFit(u16),
    MissingPiece,
}
```

```
#[derive(Debug)]  
#[non_exhaustive]  
pub enum PuzzleError {  
    WontFit(u16),  
    MissingPiece,  
}
```

```
use std::fmt::{Display, Formatter};

impl Display for PuzzleError {
    fn fmt(&self, f: &mut Formatter) -> std::fmt::Result {
        use PuzzleError::*;

        match self {
            MissingPiece => write!(f, "Missing a piece"),
            WontFit(n) => write!(f, "Piece {} doesn't fit!", n),
        }
    }
}
```



```
use std::fmt::{Display, Formatter};

impl Display for PuzzleError {
    fn fmt(&self, f: &mut Formatter) -> std::fmt::Result {
        use PuzzleError::*;

        match self {
            MissingPiece => write!(f, "Missing a piece"),
            WontFit(n) => write!(f, "Piece {} doesn't fit!", n),
        }
    }
}
```

```
use std::error::Error;

impl Error for PuzzleError {
    fn source(&self) -> Option<&(dyn Error + 'static)> {
        None
    }
}
```



```
use std::error::Error;

impl Error for PuzzleError {
    fn source(&self) -> Option<&(dyn Error + 'static)> {
        None
    }
}
```


Yes We Can!

#5b. Use thiserror

[package]

name = "puzzles"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

[package]

name = "puzzles"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

thiserror = "1.0"

```
#[non_exhaustive]
pub enum PuzzleError {
    WontFit(u16),
    MissingPiece,
}
```

```
use thiserror::Error;

#[non_exhaustive]
pub enum PuzzleError {
    WontFit(u16),
    MissingPiece,
}
```

```
use thiserror::Error;

#[derive(Debug, Error)]
#[non_exhaustive]
pub enum PuzzleError {
    WontFit(u16),
    MissingPiece,
}
```

```
use thiserror::Error;

#[derive(Debug, Error)]
#[non_exhaustive]
pub enum PuzzleError {
    #[error("Piece {0} doesn't fit!")]
    WontFit(u16),
    #[error("Missing a piece")]
    MissingPiece,
}
```

```
use thiserror::Error;

#[derive(Debug, Error)]
#[non_exhaustive]
pub enum PuzzleError {
    #[error("Piece {0} doesn't fit!")]
    WontFit(u16),
    #[error("Missing a piece")]
    MissingPiece,
}
```

```
use thiserror::Error;

#[derive(Debug, Error)]
#[non_exhaustive]
pub enum PuzzleError {

    #[error("Piece {piece_index} doesn't fit!")]
    WontFit { piece_index: u16 },
    #[error("Missing a piece")]
    MissingPiece,
}
```

```
use thiserror::Error;

#[derive(Debug, Error)]
#[non_exhaustive]
pub enum PuzzleError {
    #[error("Piece {0} doesn't fit!")]
    WontFit(u16),
    #[error("Missing a piece")]
    MissingPiece,
}
```

```
use thiserror::Error;

#[derive(Debug, Error)]
#[non_exhaustive]
pub enum PuzzleError {
    #[error("Piece {0} doesn't fit!")]
    WontFit(u16),
    #[error("Missing a piece")]
    MissingPiece,
}
```

#5

```
use std::error::Error;
```

```
use std::fmt::{Display, Formatter};
```

```
#[derive(Debug, Error)]  
#[non_exhaustive]  
  
pub enum PuzzleError {  
    #[error("Piece {0} doesn't fit!")]  
    WontFit(u16),  
    #[error("Missing a piece")]  
    MissingPiece,  
}  
  
impl Display for PuzzleError {  
    fn fmt(&self, f: &mut Formatter) -> std::fmt::Result {  
        use PuzzleError::*;

        match self {  
            MissingPiece => write!(f, "Missing a piece"),  
            WontFit(n) => write!(f, "Piece {} doesn't fit!", n),  
        }
    }
}
```

#5b

```
use thiserror::Error;
```

Handling Errors

Handling Errors in an application

Result

#[must_use]

enum Result<T, E> {

Ok(T),

Err(E),

}

Non-Recoverable Errors

panic!

```
// Manually panic  
panic!("Your computer is on fire");
```

```
// Manually panic  
panic!("Your computer is on fire");
```

```
// Same thing if result is a Result::Err  
result.expect("Your computer is on fire");
```

```
// Manually panic  
panic!("Your computer is on fire");
```

```
// Same thing if result is a Result::Err  
result.expect("Your computer is on fire");
```

```
// Same thing, but without a message  
result.unwrap();
```

Recoverable Errors

Handle or Return

```
if let Err(e) = my_result {  
    println!("Warning: {}", e);  
}
```

```
if let Err(e) = my_result {  
    println!("Warning: {}", e);  
}
```

```
if let Err(e) = my_result {  
    println!("Warning: {}", e);  
}
```

```
if let Err(e) = my_result {  
    println!("Warning: {}", e);  
}
```

```
if let Err(e) = my_result {  
    println!("Warning: {}", e);  
}
```

```
let score = match get_saved_score() {  
    Ok(x) => x,  
    Err(_) => 0,  
};
```

```
let score = get_saved_score().unwrap_or(0);
```

Return the Error


```
fn poem() -> Result<String, io::Error> {
    let file = File::open("pretty_words.txt")?;
    // do stuff with file
}
```



```
fn poem() -> Result<String, io::Error> {
    let file = File::open("pretty_words.txt")?;
    // do stuff with file
}
```

optimus.stand()?
transform()?
rollout()?
chase()?

```
pub fn autobots_rollout() -> Result<Vehicle, TransformerError> {  
    let optimus = Transformer::new();  
    optimus.stand()?.transform()?.rollout()?.chase()?.  
}
```

optimus.stand()?
transform()?
rollout()?
chase()?

```
try!(try!(try!(optimus.stand()).transform()).rollout()).chase())
```


optimus.stand()?
transform()?
rollout()?
chase()?

No!

[package]

name = "puzzle_game"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

[package]

name = "puzzle_game"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

anyhow = "1.0"

[package]

name = "puzzle_game"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

anyhow = "1.0"

eyre snafu

[package]

name = "puzzle_game"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

anyhow = "1.0"

puzzles = { path = "../puzzles" }

```
use anyhow::Result;
```

```
use anyhow::Result;  
use puzzles::Puzzle;
```

```
use anyhow::Result;  
use puzzles::Puzzle;  
use std::fs::File;
```

```
use anyhow::Result;
use puzzles::Puzzle;
use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle>
```

```
use anyhow::Result;
use puzzles::Puzzle;
use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {
    let fh = File::open(filename)?;
```

```
use anyhow::Context, Result;
use puzzles::Puzzle;
use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {
    let fh = File::open(filename)?;
```

```
use anyhow::{Context, Result};

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {
    let fh = File::open(filename).context("couldn't open the puzzle file")?;
```

```
use anyhow::{Context, Result};

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {
    let fh = File::open(filename)
        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;
}
```

```
use anyhow::{Context, Result};

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {
    let fh = File::open(filename)
        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;
    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    puzzle
}
```

```
use anyhow::{Context, Result};

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {
    let fh = File::open(filename)
        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;
    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;
    Ok(puzzle)
}
```

main.rs

```
use anyhow::{Context, Result};
```

```
use puzzles::Puzzle;
```

```
use std::fs::File;
```

```
fn get_puzzle(filename: &str) -> Result<Puzzle> {
```

```
    let fh = File::open(filename)
```

```
        .with_context(|| format!("couldn't open the puzzle file {}", filename))?
```

```
    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?
```

```
    Ok(puzzle)
```

```
}
```

```
fn main() -> Result<()>
```

main.rs

```
use anyhow::{Context, Result};
```

```
use puzzles::Puzzle;
```

```
use std::fs::File;
```

```
fn get_puzzle(filename: &str) -> Result<Puzzle> {
```

```
    let fh = File::open(filename)
```

```
        .with_context(|| format!("couldn't open the puzzle file {}", filename))?
```

```
    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?
```

```
    Ok(puzzle)
```

```
}
```

```
fn main() -> Result<()> {
```

```
    let puzzle = get_puzzle("puzzle.dat").context("Couldn't get the first puzzle")?
```

main.rs

```
use anyhow::{Context, Result};

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {
    let fh = File::open(filename)
        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {
    let puzzle = get_puzzle("puzzle.dat").context("Couldn't get the first puzzle")?;
```

\$ cargo run

```
$ cargo run
Compiling proc-macro2 v1.0.27
Compiling unicode-xid v0.2.2
Compiling syn v1.0.72
Compiling anyhow v1.0.40
Compiling quote v1.0.9
Compiling thiserror-impl v1.0.25
Compiling thiserror v1.0.25
Compiling puzzles v0.1.0 (/Users/cleancut/projects/puzzles)
Compiling puzzle_game v0.1.0 (/Users/cleancut/projects/puzzle_game)
Finished dev [unoptimized + debuginfo] target(s) in 6.97s
Running `target/debug/puzzle_game`
Error: Couldn't get the first puzzle
```

Caused by:

```
0: couldn't open the puzzle file puzzle.dat
1: No such file or directory (os error 2)
```

```
$
```

```
$ cargo run
Compiling proc-macro2 v1.0.27
Compiling unicode-xid v0.2.2
Compiling syn v1.0.72
Compiling anyhow v1.0.40
Compiling quote v1.0.9
Compiling thiserror-impl v1.0.25
Compiling thiserror v1.0.25
Compiling puzzles v0.1.0 (/Users/cleancut/projects/puzzles)
Compiling puzzle_game v0.1.0 (/Users/cleancut/projects/puzzle_game)
Finished dev [unoptimized + debuginfo] target(s) in 6.97s
Running `target/debug/puzzle_game`
Error: Couldn't get the first puzzle
```

Caused by:

```
0: couldn't open the puzzle file puzzle.dat
1: No such file or directory (os error 2)
```

\$

```
$ cargo run
Compiling proc-macro2 v1.0.27
Compiling unicode-xid v0.2.2
Compiling syn v1.0.72
Compiling anyhow v1.0.40
Compiling quote v1.0.9
Compiling thiserror-impl v1.0.25
Compiling thiserror v1.0.25
Compiling puzzles v0.1.0 (/Users/cleancut/projects/puzzles)
Compiling puzzle_game v0.1.0 (/Users/cleancut/projects/puzzle_game)
Finished dev [unoptimized + debuginfo] target(s) in 6.97s
Running `target/debug/puzzle_game`
Error: Couldn't get the first puzzle
```

Caused by:

```
0: couldn't open the puzzle file puzzle.dat
1: No such file or directory (os error 2)
```

\$

```
$ cargo run
Compiling proc-macro2 v1.0.27
Compiling unicode-xid v0.2.2
Compiling syn v1.0.72
Compiling anyhow v1.0.40
Compiling quote v1.0.9
Compiling thiserror-impl v1.0.25
Compiling thiserror v1.0.25
Compiling puzzles v0.1.0 (/Users/cleancut/projects/puzzles)
Compiling puzzle_game v0.1.0 (/Users/cleancut/projects/puzzle_game)
Finished dev [unoptimized + debuginfo] target(s) in 6.97s
Running `target/debug/puzzle_game`
Error: Couldn't get the first puzzle
```

Caused by:

```
0: couldn't open the puzzle file puzzle.dat
1: No such file or directory (os error 2)
```

\$

\$ cargo run

```
$ cargo run  
Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
Running `target/debug/puzzle_game`
```

Error: Couldn't get the first puzzle

Caused by:

0: couldn't convert data into a puzzle
1: Missing a piece

\$

```
$ cargo run  
  Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
    Running `target/debug/puzzle_game`  
Error: Couldn't get the first puzzle
```

Caused by:

0: couldn't convert data into a puzzle
1: Missing a piece

\$


```
$ rustup override set nightly
```

[snip]

```
$ RUST_BACKTRACE=1 cargo run
```

[snip]

Error: Couldn't get the first puzzle

Caused by:

- 0: couldn't convert data into a puzzle
- 1: Missing a piece

Stack backtrace:

```
0: std::backtrace::backtrace::libunwind::trace  
    at /rustc/79e50bf77928f374921a6bcafee3fcff1915f062/library/std/src/../..//backtrace/src/backtrace/libunwind.rs:90:5  
std::backtrace::backtrace::trace_unsynchronized  
    at /rustc/79e50bf77928f374921a6bcafee3fcff1915f062/library/std/src/../..//backtrace/src/backtrace/mod.rs:66:5  
std::backtrace::Backtrace::create  
    at /rustc/79e50bf77928f374921a6bcafee3fcff1915f062/library/std/src/backtrace.rs:327:13  
1: std::backtrace::Backtrace::capture  
    at /rustc/79e50bf77928f374921a6bcafee3fcff1915f062/library/std/src/backtrace.rs:295:9  
2: <E as anyhow::context::ext::StdError>::ext_context  
    at /Users/cleancut/.cargo/registry/src/github.com-1ecc6299db9ec823/anyhow-1.0.40/src/context.rs:27:29  
3: anyhow::context::<impl anyhow::Context<T,E> for core::result::Result<T,E>>::context::{closure}  
    at /Users/cleancut/.cargo/registry/src/github.com-1ecc6299db9ec823/anyhow-1.0.40/src/context.rs:50:30  
4: core::result::Result<T,E>::map_err  
    at /Users/cleancut/.rustup/toolchains/nightly-x86_64-apple-darwin/lib/rustlib/src/rust/library/core/src/result.rs:591:27  
5: anyhow::context::<impl anyhow::Context<T,E> for core::result::Result<T,E>>::context  
    at /Users/cleancut/.cargo/registry/src/github.com-1ecc6299db9ec823/anyhow-1.0.40/src/context.rs:50:9  
6: puzzle_game::get_puzzle  
    at ./src/main.rs:8:18  
7: puzzle_game::main  
    at ./src/main.rs:13:18  
8: core::ops::function::FnOnce::call_once  
    at /Users/cleancut/.rustup/toolchains/nightly-x86_64-apple-darwin/lib/rustlib/src/rust/library/core/src/ops/function.rs:227:5  
9: std::sys_common::backtrace::__rust_begin_short_backtrace  
    at /Users/cleancut/.rustup/toolchains/nightly-x86_64-apple-darwin/lib/rustlib/src/rust/library/std/src/sys_common/backtrace.rs:125:18  
10: std::rt::lang_start::{closure}  
    at /Users/cleancut/.rustup/toolchains/nightly-x86_64-apple-darwin/lib/rustlib/src/rust/library/std/src/rt.rs:49:18
```

[snip -- you get the idea]

Exercise 5

exercises/errors

Questions?

10 Minute Break

Testing

Unit Tests

```
pub fn snuggle(bunnies: u128) -> u128 {  
    bunnies * 8  
}
```

```
pub fn snuggle(bunnies: u128) -> u128 {  
    bunnies * 8  
}
```

```
mod test {  
}
```

```
pub fn snuggle(bunnies: u128) -> u128 {  
    bunnies * 8  
}
```

```
mod test {  
}
```

```
pub fn snuggle(bunnies: u128) -> u128 {  
    bunnies * 8  
}
```

```
#[cfg(test)]  
mod test {  
}
```

```
pub fn snuggle(bunnies: u128) -> u128 {  
    bunnies * 8  
}
```

```
#[cfg(test)]  
mod test {  
}
```

```
pub fn snuggle(bunnies: u128) -> u128 {  
    bunnies * 8  
}
```

```
#[cfg(test)]  
mod test {  
}
```

```
pub fn snuggle(bunnies: u128) -> u128 {
```

```
    bunnies * 8
```

```
}
```

```
#[cfg(test)]
```

```
mod test {
```

```
use super::*;

#[test]
```

```
fn snuggling_bunnies_multiply() {
```

```
pub fn snuggle(bunnies: u128) -> u128 {
```

```
    bunnies * 8
```

```
}
```

```
#[cfg(test)]
```

```
mod test {
```

```
use super::*;

#[test]
```

```
fn snuggling_bunnies_multiply() {
```

```
pub fn snuggle(bunnies: u128) -> u128 {
```

```
    bunnies * 8
```

```
}
```

```
#[cfg(test)]
```

```
mod test {
```

```
use super::*;

#[test]
```

```
fn snuggling_bunnies_multiply() {
```

```
pub fn snuggle(bunnies: u128) -> u128 {
```

bunnies * 8

```
}
```

```
#[cfg(test)]
```

```
mod test {
```

```
use super::*;

#[test]
```

```
fn snuggling_bunnies_multiply() {
```

```
pub fn snuggle(bunnies: u128) -> u128 {
```

bunnies * 8

```
}
```

```
#[cfg(test)]
```

```
mod test {
```

```
use super::*;

#[test]
```

```
fn snuggling_bunnies_multiply() {
```

```
pub fn snuggle(bunnies: u128) -> u128 {
```

bunnies * 8

```
}
```

```
#[cfg(test)]
```

```
mod test {
```

```
use super::*;

#[test]
```

```
fn snuggling_bunnies_multiply() {
```

```
assert_eq!(5, 6);
```

PartialEq

```
assert_eq!(5, 6);
```

```
assert_ne!(5, 5);
```

```
assert!(5 >= 5);
```

```
panic!("Time to crash!");
```

`#[should_panic]`

```
#[should_panic]
#[test]
fn scared_bunny() {
    panic!("Hop hoppity hop!");
}
```

```
pub fn snuggle(bunnies: u128) -> u128 {
```

```
    bunnies * 8
```

```
}
```

```
#[cfg(test)]
```

```
mod test {
```

```
use super::*;

#[test]
```

```
fn snuggling_bunnies_multiply() {
```

cargo test

hello \$ cargo test

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
$
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
$
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
$
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
$
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
$
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
$
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
$
```

No!

Definition

1. crate = package

Definition

1. crate = package
2. crate = a library or a binary

crate = package

crate = package

crate = library (one or none)

crate = package

crate = binary 1

crate = binary 2

crate = binary 3

⋮

crate = binary N

crate = library (one or none)

crate = package

`crate = binary 1`

`crate = binary 2`

`crate = binary 3`

⋮

`crate = binary N`

`crate = library (one or none)`

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
cleancut@ice hello $
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
cleancut@ice hello $
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 1.21s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 0 tests
```

```
test result: ok. 0 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
cleancut@ice hello $
```


Example

```
let bunnies = snuggle(5);  
assert_eq!(bunnies, 40);
```

hello \$ cargo test

Compiling hello v0.1.0 (/Users/cleancut/projects/hello)

Finished test [unoptimized + debuginfo] target(s) in 0.58s

Running target/debug/deps/hello-004b8164d2284adf

running 1 test

test test::snuggling_bunnies_multiply ... **ok**

test result: **ok**. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

Doc-tests hello

running 1 test

test src/lib.rs - snuffle (line 3) ... **ok**

test result: **ok**. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.38s

hello \$

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 0.58s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

Doc-tests hello

```
running 1 test
```

```
test src/lib.rs - snuffle (line 3) ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.38s
```

```
hello $
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 0.58s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 1 test
```

```
test src/lib.rs - snuffle (line 3) ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.38s
```

```
hello $
```

```
#[test]

fn bunny_result() -> Result<(), ParseIntError> {
    let num_bunnies: u64 = "four".parse()?;
    assert_eq!(num_bunnies, 4);
    Ok(())
}
```



```
#[test]

fn bunny_result() -> Result<(), ParseIntError> {
    let num_bunnies: u64 = "four".parse()?;
    assert_eq!(num_bunnies, 4);
    Ok(())
}
```

```
#[test]

fn bunny_result() -> Result<(), ParseIntError> {
    let num_bunnies: u64 = "four".parse()?;
    assert_eq!(num_bunnies, 4);
    Ok(())
}
```

```
$ cargo test
Finished test [unoptimized + debuginfo] target(s) in 0.05s
Running unitests (target/debug/deps/hello-0e8ff4f0fd808e8e)
```

```
running 3 tests
test test::snuggling_bunnies_multiply ... ok
test test::scared_bunny ... ok
test test::bunny_result ... FAILED
```

failures:

```
---- test::bunny_result stdout ----
Error: ParseIntError { kind: InvalidDigit }
thread 'test::bunny_result' panicked at 'assertion failed: `!(left == right)`
left: `1`,
right: `0`: the test returned a termination value with a non-zero status code (1) which indicates a failure',
/Users/cleanut/.rustup/toolchains/stable-x86_64-apple-darwin/lib/rustlib/src/rust/library/test/src/lib.rs:193:5
```

failures:

```
test::bunny_result
```

```
test result: FAILED. 2 passed; 1 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
error: test failed, to rerun pass '--lib'
```

```
$
```

```
$ cargo test
Finished test [unoptimized + debuginfo] target(s) in 0.05s
Running unitests (target/debug/deps/hello-0e8ff4f0fd808e8e)
```

```
running 3 tests
test test::snuggling_bunnies_multiply ... ok
test test::scared_bunny ... ok
test test::bunny_result ... FAILED
```

failures:

```
---- test::bunny_result stdout ----
Error: ParseIntError { kind: InvalidDigit }
thread 'test::bunny_result' panicked at 'assertion failed: `!(left == right)`
left: `1`,
right: `0`: the test returned a termination value with a non-zero status code (1) which indicates a failure',
/Users/cleanut/.rustup/toolchains/stable-x86_64-apple-darwin/lib/rustlib/src/rust/library/test/src/lib.rs:193:5
```

failures:

```
test::bunny_result
```

```
test result: FAILED. 2 passed; 1 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
error: test failed, to rerun pass '--lib'
```

\$

```
$ cargo test
```

```
Finished test [unoptimized + debuginfo] target(s) in 0.05s
Running unitests (target/debug/deps/hello-0e8ff4f0fd808e8e)
```

```
running 3 tests
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test test::scared_bunny ... ok
```

```
test test::bunny_result ... FAILED
```

```
failures:
```

```
---- test::bunny_result stdout ----
```

```
Error: ParseIntError { kind: InvalidDigit }
```

```
thread 'test::bunny_result' panicked at 'assertion failed: `!(left == right)`'
```

```
left: `1`,
```

```
right: `0`: the test returned a termination value with a non-zero status code (1) which indicates a failure',
```

```
/Users/cleanut/.rustup/toolchains/stable-x86_64-apple-darwin/lib/rustlib/src/rust/library/test/src/lib.rs:193:5
```

```
failures:
```

```
test::bunny_result
```

```
test result: FAILED. 2 passed; 1 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
error: test failed, to rerun pass '--lib'
```

```
$
```

```
$ cargo test
```

```
Finished test [unoptimized + debuginfo] target(s) in 0.05s
Running unitests (target/debug/deps/hello-0e8ff4f0fd808e8e)
```

```
running 3 tests
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test test::scared_bunny ... ok
```

```
test test::bunny_result ... FAILED
```

```
failures:
```

```
---- test::bunny_result stdout ----
```

```
Error: ParseIntError { kind: InvalidDigit }
```

```
thread 'test::bunny_result' panicked at 'assertion failed: `!(left == right)`'
```

```
left: `1`,
```

```
right: `0`: the test returned a termination value with a non-zero status code (1) which indicates a failure',
```

```
/Users/cleanut/.rustup/toolchains/stable-x86_64-apple-darwin/lib/rustlib/src/rust/library/test/src/lib.rs:193:5
```

```
failures:
```

```
test::bunny_result
```

```
test result: FAILED. 2 passed; 1 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
error: test failed, to rerun pass '--lib'
```

```
$
```

```
$ cargo test
```

```
Finished test [unoptimized + debuginfo] target(s) in 0.05s
Running unitests (target/debug/deps/hello-0e8ff4f0fd808e8e)
```

```
running 3 tests
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test test::scared_bunny ... ok
```

```
test test::bunny_result ... FAILED
```

```
failures:
```

```
---- test::bunny_result stdout ----
```

```
Error: ParseIntError { kind: InvalidDigit }
```

```
thread 'test::bunny_result' panicked at 'assertion failed: `!(left == right)`'
```

```
left: `1`,
```

```
right: `0`: the test returned a termination value with a non-zero status code (1) which indicates a failure',
```

```
/Users/cleanut/.rustup/toolchains/stable-x86_64-apple-darwin/lib/rustlib/src/rust/library/test/src/lib.rs:193:5
```

```
failures:
```

```
test::bunny_result
```

```
test result: FAILED. 2 passed; 1 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
error: test failed, to rerun pass '--lib'
```

```
$
```

```
$ cargo test
```

```
Finished test [unoptimized + debuginfo] target(s) in 0.05s
Running unitests (target/debug/deps/hello-0e8ff4f0fd808e8e)
```

```
running 3 tests
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test test::scared_bunny ... ok
```

```
test test::bunny_result ... FAILED
```

```
failures:
```

```
---- test::bunny_result stdout ----
```

```
Error: ParseIntError { kind: InvalidDigit }
```

```
thread 'test::bunny_result' panicked at 'assertion failed: `!(left == right)`'
```

```
left: `1`,
```

```
right: `0`: the test returned a termination value with a non-zero status code (1) which indicates a failure',
```

```
/Users/cleanut/.rustup/toolchains/stable-x86_64-apple-darwin/lib/rustlib/src/rust/library/test/src/lib.rs:193:5
```

```
failures:
```

```
test::bunny_result
```

```
test result: FAILED. 2 passed; 1 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
error: test failed, to rerun pass '--lib'
```

```
$
```

```
$ cargo test
```

```
Finished test [unoptimized + debuginfo] target(s) in 0.05s
Running unitests (target/debug/deps/hello-0e8ff4f0fd808e8e)
```

```
running 3 tests
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test test::scared_bunny ... ok
```

```
test test::bunny_result ... FAILED
```

```
failures:
```

```
---- test::bunny_result stdout ----
```

```
Error: ParseIntError { kind: InvalidDigit }
```

```
thread 'test::bunny_result' panicked at 'assertion failed: `!(left == right)`'
```

```
left: `1`,
```

```
right: `0`: the test returned a termination value with a non-zero status code (1) which indicates a failure',
```

```
/Users/cleanut/.rustup/toolchains/stable-x86_64-apple-darwin/lib/rustlib/src/rust/library/test/src/lib.rs:193:5
```

```
failures:
```

```
test::bunny_result
```

```
test result: FAILED. 2 passed; 1 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
error: test failed, to rerun pass '--lib'
```

```
$
```

```
$ cargo test
  Finished test [unoptimized + debuginfo] target(s) in 0.05s
    Running unitests (target/debug/deps/hello-0e8ff4f0fd808e8e)
```

```
running 3 tests
test test::snuggling_bunnies_multiply ... ok
test test::scared_bunny ... ok
test test::bunny_result ... FAILED
```

failures:

```
---- test::bunny_result stdout ----
Error: ParseIntError { kind: InvalidDigit }
thread 'test::bunny_result' panicked at 'assertion failed: `!(left == right)`
left: `1`,
right: `0`: the test returned a termination value with a non-zero status code (1) which indicates a failure',
/Users/cleanut/.rustup/toolchains/stable-x86_64-apple-darwin/lib/rustlib/src/rust/library/test/src/lib.rs:193:5
```

failures:

```
test::bunny_result
```

```
test result: FAILED. 2 passed; 1 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

error: test failed, to rerun pass '--lib'

\$

```
$ cargo test test::bunny_result
```

Integration Tests

hello \$ mkdir tests

hello \$

```
hello $ mkdir tests
```

```
hello $ tree -l target
```

```
.
```

- └── Cargo.lock
- └── Cargo.toml
- └── **src**
 - └── lib.rs
 - └── **tests**

2 directories, 3 files

```
hello $
```

```
hello $ mkdir tests
```

```
hello $ tree -l target
```

```
.
```

- └── Cargo.lock
- └── Cargo.toml
- └── **src**
 - └── lib.rs
 - └── **tests**

2 directories, 3 files

```
hello $ vim tests/anything.rs
```

```
use hello::snuggle;

#[test]

fn it_works_from_outside() {
    assert!(snuffle(4) == 32);

}
```

```
use hello::snuggle;
```

```
#[test]
```

```
fn it_works_from_outside() {
```

```
    assert!(snuffle(4) == 32);
```

```
}
```

```
use hello::snuggle;
```

```
#[test]
```

```
fn it_works_from_outside() {
```

```
    assert!(snuffle(4) == 32);
```

```
}
```

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 0.90s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Running target/debug/deps/anything-fd796e12b370625a
```

```
running 1 test
```

```
test it_works_from_outside ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 1 test
```

```
test src/lib.rs - snuffle (line 3) ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.21s
```

hello \$ cargo test

Compiling hello v0.1.0 (/Users/cleancut/projects/hello)

Finished test [unoptimized + debuginfo] target(s) in 0.90s

Running target/debug/deps/hello-004b8164d2284adf

running 1 test

test test::snuggling_bunnies_multiply ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

Running target/debug/deps/anything-fd796e12b370625a

running 1 test

test it_works_from_outside ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

Doc-tests hello

running 1 test

test src/lib.rs - snuffle (line 3) ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.21s

```
hello $ cargo test
```

```
Compiling hello v0.1.0 (/Users/cleancut/projects/hello)
```

```
Finished test [unoptimized + debuginfo] target(s) in 0.90s
```

```
Running target/debug/deps/hello-004b8164d2284adf
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Running target/debug/deps/anything-fd796e12b370625a
```

```
running 1 test
```

```
test it_works_from_outside ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Doc-tests hello
```

```
running 1 test
```

```
test src/lib.rs - snuffle (line 3) ... ok
```

```
test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.21s
```

hello \$ cargo test

Compiling hello v0.1.0 (/Users/cleancut/projects/hello)

Finished test [unoptimized + debuginfo] target(s) in 0.90s

Running target/debug/deps/hello-004b8164d2284adf

running 1 test

test test::snuggling_bunnies_multiply ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

Running target/debug/deps/anything-fd796e12b370625a

running 1 test

test it_works_from_outside ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.00s

Doc-tests hello

running 1 test

test src/lib.rs - snuffle (line 3) ... ok

test result: ok. 1 passed; 0 failed; 0 ignored; 0 measured; 0 filtered out; finished in 0.21s

binary

binary

LIBRARY

binary



LIBRARY

[+] Show declaration

[−] A process builder, providing fine-grained control over how a new process should be spawned.

A default configuration can be generated using `Command::new(program)`, where `program` gives a path to the program to be executed. Additional builder methods allow the configuration to be changed (for example, by adding arguments) prior to spawning:

```
use std::process::Command;

let output = if cfg!(target_os = "windows") {
    Command::new("cmd")
        .args(&["/C", "echo hello"])
        .output()
        .expect("failed to execute process")
} else {
    Command::new("sh")
        .arg("-c")
        .arg("echo hello")
        .output()
        .expect("failed to execute process")
};

let hello = output.stdout;
```

Run

Benchmarks

Criterion

Cargo.toml

```
[package]

name = "hello"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

[dev-dependencies]

criterion = { version = "0.3", features = ["html_reports"] }
```

Cargo.toml

```
[package]

name = "hello"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

[dev-dependencies]

criterion = { version = "0.3", features = ["html_reports"] }
```

Cargo.toml

```
[package]
```

```
name = "hello"
```

```
version = "0.1.0"
```

```
authors = ["Nathan Stocks <nathan@agileperception.com>"]
```

```
edition = "2018"
```

```
[dependencies]
```

```
[dev-dependencies]
```

```
criterion = { version = "0.3", features = ["html_reports"] }
```

Cargo.toml

```
[package]
```

```
name = "hello"
```

```
version = "0.1.0"
```

```
authors = ["Nathan Stocks <nathan@agileperception.com>"]
```

```
edition = "2018"
```

```
[dependencies]
```

```
[dev-dependencies]
```

```
criterion = { version = "0.3", features = ["html_reports"] }
```

Cargo.toml

```
[package]
```

```
name = "hello"
```

```
version = "0.1.0"
```

```
authors = ["Nathan Stocks <nathan@agileperception.com>"]
```

```
edition = "2018"
```

```
[dependencies]
```

```
[dev-dependencies]
```

```
criterion = { version = "0.3", features = ["html_reports"] }
```

hello \$ mkdir benches

hello \$

hello \$ mkdir benches

hello \$ tree -l 'target|tests'

```
.
```

- ├── Cargo.lock
- ├── Cargo.toml
- └── **benches**

- └── **src**

- └── lib.rs

2 directories, 3 files

hello \$

hello \$ mkdir benches

hello \$ tree -l 'target|tests'

```
.
```

- ├── Cargo.lock
- ├── Cargo.toml
- └── **benches**

- └── **src**

- └── lib.rs

2 directories, 3 files

hello \$ vim benches/snuggle_speed.rs

```
use criterion::{black_box, criterion_group, criterion_main, Criterion};
```

```
use hello::snuggle;
```

```
pub fn snuffle_benchmark(c: &mut Criterion) {  
    c.bench_function("snuffle 2", |b| b.iter(|| snuffle(black_box(2))));  
}
```

```
criterion_group!(benches, snuffle_benchmark);
```

```
criterion_main!(benches);
```

```
use criterion::{black_box, criterion_group, criterion_main, Criterion};
use hello::snuffle;

pub fn snuffle_benchmark(c: &mut Criterion) {
    c.bench_function("snuffle 2", |b| b.iter(|| snuffle(black_box(2))));
}

criterion_group!(benches, snuffle_benchmark);
criterion_main!(benches);
```

```
use criterion::{black_box, criterion_group, criterion_main, Criterion};
```

```
use hello::snuggle;
```

```
pub fn snuffle_benchmark(c: &mut Criterion) {  
    c.bench_function("snuffle 2", |b| b.iter(|| snuffle(black_box(2))));  
}
```

```
criterion_group!(benches, snuffle_benchmark);
```

```
criterion_main!(benches);
```

```
use criterion::{black_box, criterion_group, criterion_main, Criterion};
```

```
use hello::snuggle;
```

```
pub fn snuffle_benchmark(c: &mut Criterion) {  
    c.bench_function("snuffle 2", |b| b.iter(|| snuffle(black_box(2))));  
}
```

```
criterion_group!(benches, snuffle_benchmark);
```

```
criterion_main!(benches);
```

```
use criterion::{black_box, criterion_group, criterion_main, Criterion};
```

```
use hello::snuggle;
```

```
pub fn snuffle_benchmark(c: &mut Criterion) {  
    c.bench_function("snuffle 2", |b| b.iter(|| snuffle(black_box(2))));  
}
```

```
criterion_group!(benches, snuffle_benchmark);
```

```
criterion_main!(benches);
```

```
use criterion::{black_box, criterion_group, criterion_main, Criterion};
```

```
use hello::snuffle;
```

```
pub fn snuffle_benchmark(c: &mut Criterion) {  
    c.bench_function("snuffle 2", |b| b.iter(|| snuffle(black_box(2))));  
}
```

```
criterion_group!(benches, snuffle_benchmark);
```

```
criterion_main!(benches);
```

```
use criterion::{black_box, criterion_group, criterion_main, Criterion};
```

```
use hello::snuffle;
```

```
pub fn snuffle_benchmark(c: &mut Criterion) {  
    c.bench_function("snuffle 2", |b| b.iter(|| snuffle(black_box(2))));  
}
```

```
criterion_group!(benches, snuffle_benchmark);
```

```
criterion_main!(benches);
```

```
use criterion::{black_box, criterion_group, criterion_main, Criterion};
```

```
use hello::snuggle;
```

```
pub fn snuffle_benchmark(c: &mut Criterion) {  
    c.bench_function("snuffle 2", |b| b.iter(|| snuffle(black_box(2))));  
}
```

```
criterion_group!(benches, snuffle_benchmark);
```

```
criterion_main!(benches);
```

```
use criterion::{black_box, criterion_group, criterion_main, Criterion};
```

```
use hello::snuggle;
```

```
pub fn snuffle_benchmark(c: &mut Criterion) {  
    c.bench_function("snuffle 2", |b| b.iter(|| snuffle(black_box(2))));  
}
```

```
criterion_group!(benches, snuffle_benchmark);
```

```
criterion_main!(benches);
```

cargo bench

hello \$ cargo bench

hello \$ cargo bench

Compiling autocfg v1.0.1

[...snip many lines of compiling dependencies...]

Compiling criterion v0.3.4

Finished bench [optimized] target(s) in 34.45s

Running target/release/deps/hello-8d5b79303bf2f5da

running 1 test

test test::snuggling_bunnies_multiply ... ignored

test result: **ok**. 0 passed; 0 failed; 1 ignored; 0 measured; 0 filtered out; finished in 0.00s

Running target/release/deps/snuggle_speed-22ad257c0f2a9def

snuffle 2 time: [1.9421 ns **1.9524 ns** 1.9631 ns]

Found 6 outliers among 100 measurements (6.00%)

2 (2.00%) low mild

1 (1.00%) high mild

3 (3.00%) high severe

hello \$

```
hello $ cargo bench
```

```
  Compiling autocfg v1.0.1
```

```
[...snip many lines of compiling dependencies...]
```

```
  Compiling criterion v0.3.4
```

```
  Finished bench [optimized] target(s) in 34.45s
```

```
  Running target/release/deps/hello-8d5b79303bf2f5da
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ignored
```

```
test result: ok. 0 passed; 0 failed; 1 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
Running target/release/deps/snuffle_speed-22ad257c0f2a9def
```

```
snuffle 2          time: [1.9421 ns 1.9524 ns 1.9631 ns]
```

```
Found 6 outliers among 100 measurements (6.00%)
```

```
  2 (2.00%) low mild
```

```
  1 (1.00%) high mild
```

```
  3 (3.00%) high severe
```

```
hello $
```

```
hello $ cargo bench
```

```
  Compiling autocfg v1.0.1
```

```
[...snip many lines of compiling dependencies...]
```

```
  Compiling criterion v0.3.4
```

```
  Finished bench [optimized] target(s) in 34.45s
```

```
  Running target/release/deps/hello-8d5b79303bf2f5da
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ignored
```

```
test result: ok. 0 passed; 0 failed; 1 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
  Running target/release/deps/snuffle_speed-22ad257c0f2a9def
```

```
    snuffle 2          time: [1.9421 ns 1.9524 ns 1.9631 ns]
```

```
    Found 6 outliers among 100 measurements (6.00%)
```

```
      2 (2.00%) low mild
```

```
      1 (1.00%) high mild
```

```
      3 (3.00%) high severe
```

```
hello $
```

```
hello $ cargo bench
```

```
  Compiling autocfg v1.0.1
```

```
[...snip many lines of compiling dependencies...]
```

```
  Compiling criterion v0.3.4
```

```
  Finished bench [optimized] target(s) in 34.45s
```

```
  Running target/release/deps/hello-8d5b79303bf2f5da
```

```
running 1 test
```

```
test test::snuggling_bunnies_multiply ... ignored
```

```
test result: ok. 0 passed; 0 failed; 1 ignored; 0 measured; 0 filtered out; finished in 0.00s
```

```
  Running target/release/deps/snuffle_speed-22ad257c0f2a9def
```

```
    snuffle 2          time: [1.9421 ns 1.9524 ns 1.9631 ns]
```

```
    Found 6 outliers among 100 measurements (6.00%)
```

```
      2 (2.00%) low mild
```

```
      1 (1.00%) high mild
```

```
      3 (3.00%) high severe
```

```
hello $
```

hello \$ cargo bench

Compiling autocfg v1.0.1

[...snip many lines of compiling dependencies...]

Compiling criterion v0.3.4

Finished bench [optimized] target(s) in 34.45s

Running target/release/deps/hello-8d5b79303bf2f5da

running 1 test

test test::snuggling_bunnies_multiply ... ignored

test result: **ok**. 0 passed; 0 failed; 1 ignored; 0 measured; 0 filtered out; finished in 0.00s

Running target/release/deps/snuggle_speed-22ad257c0f2a9def

snuffle 2 time: [1.9421 ns **1.9524 ns** 1.9631 ns]

Found 6 outliers among 100 measurements (6.00%)

2 (2.00%) low mild

1 (1.00%) high mild

3 (3.00%) high severe

hello \$

```
pub fn snuggle(bunnies: u128) -> u128 {  
    bunnies * 8  
}
```

```
pub fn snuggle(bunnies: u128) -> u128 {  
    let mut result = 0;  
    for _ in 0..8 {  
        result += bunnies  
    }  
    result  
}
```

Loop with addition compared to multiplication

Running target/release/deps/snuffle_speed-22ad257c0f2a9def
snuffle 2 time: [1.7582 ns **1.7764 ns** 1.7942 ns]
change: [-11.058% **-9.6325%** -8.2220%] (p = 0.00 < 0.05)
Performance has improved.

Found 2 outliers among 100 measurements (2.00%)

2 (2.00%) high mild

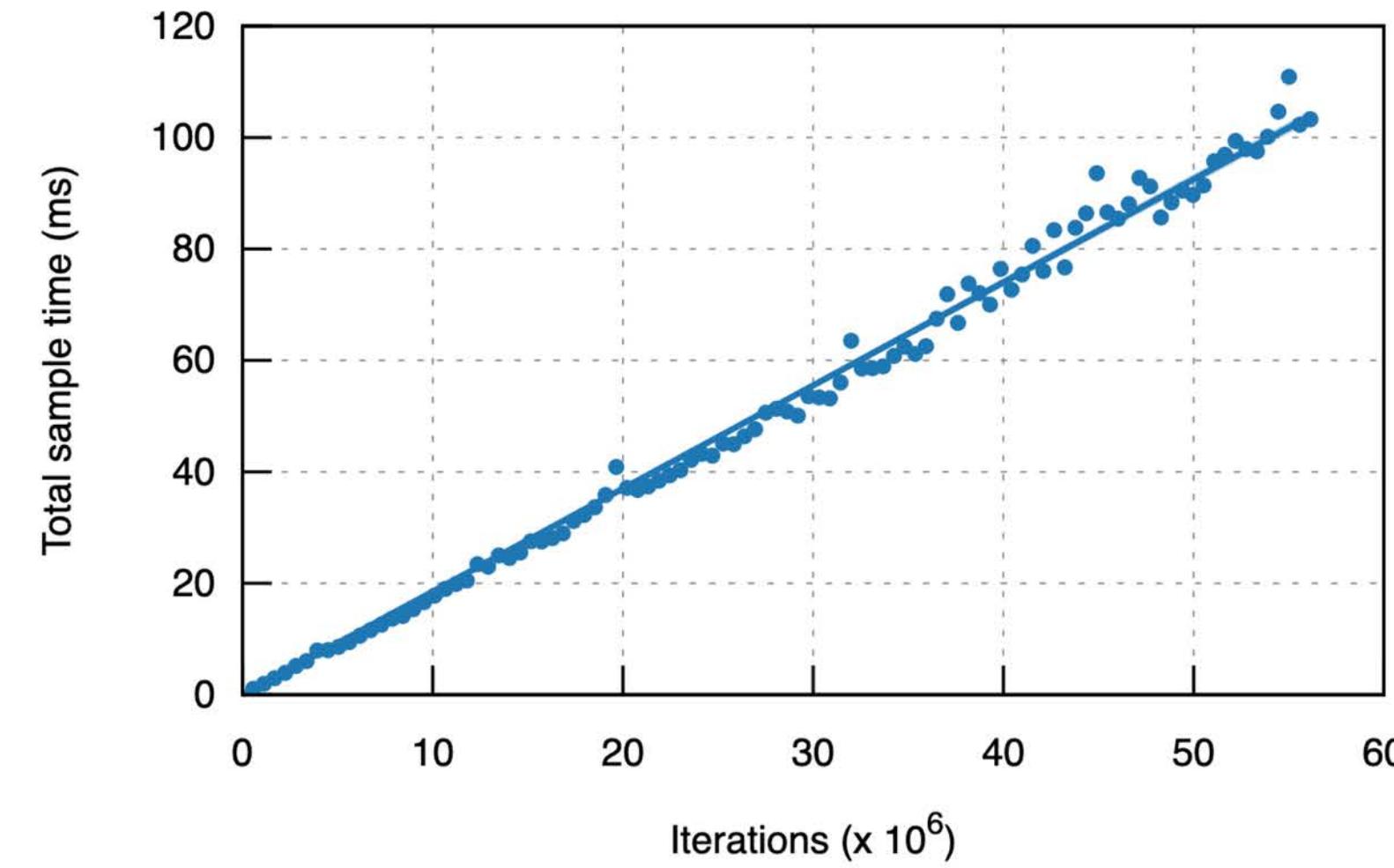
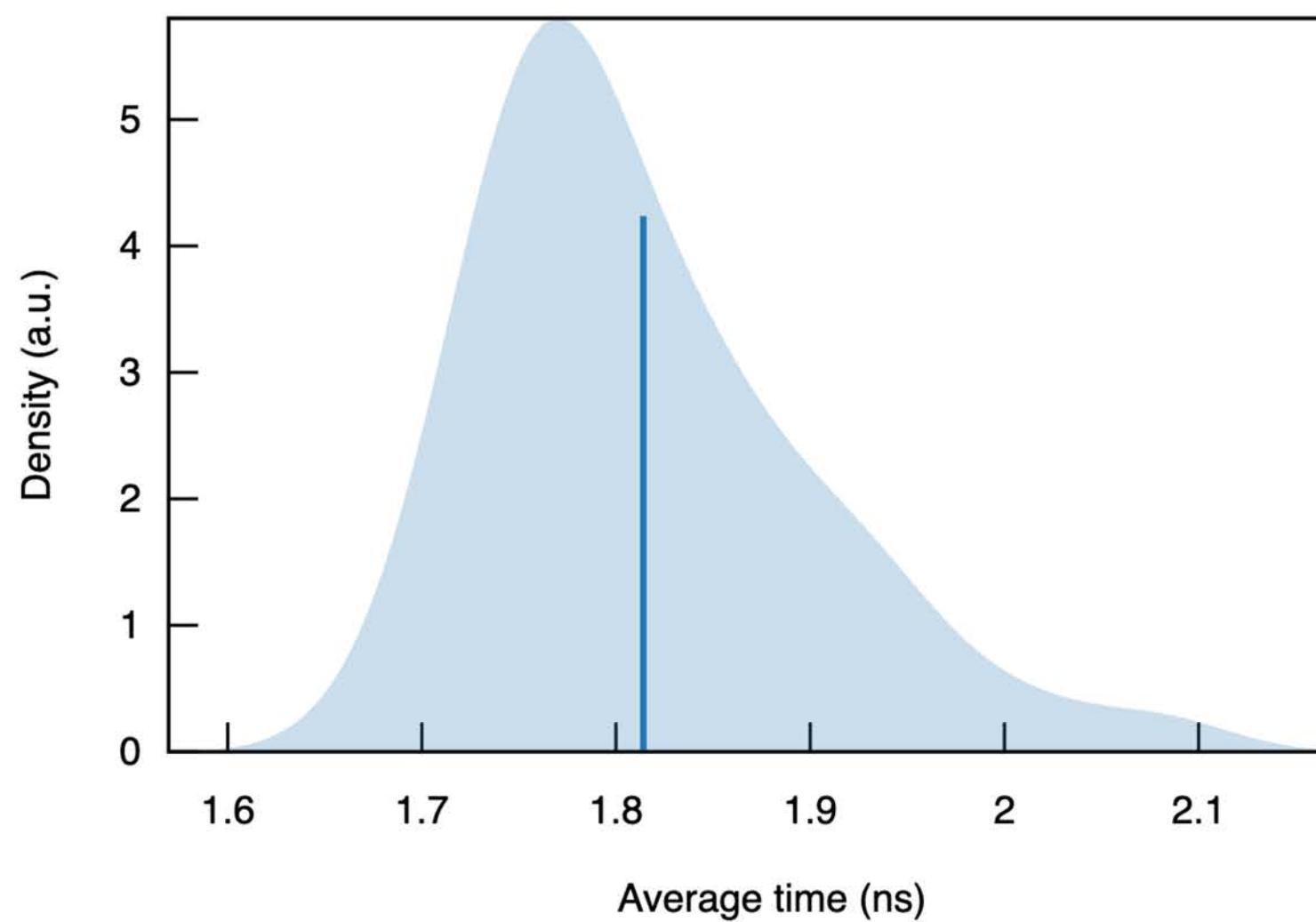
```
pub fn snuggle(bunnies: u128) -> u128 {  
    bunnies << 3  
}
```

Bit-shifting by 3 compared to loop with addition

Running target/release/deps/snuggle_speed-22ad257c0f2a9def
snuggle 2 time: [1.8305 ns **1.8502 ns** 1.8698 ns]
change: [+1.4762% **+2.8152%** +4.1108%] (p = 0.00 < 0.05)
Performance has regressed.

target/criterion/report/index.html

snuggle 2



Additional Statistics:

	Lower bound	Estimate	Upper bound
Slope	1.8305 ns	1.8502 ns	1.8698 ns
R ²	0.7874063	0.7972891	0.7874431
Mean	1.7981 ns	1.8141 ns	1.8311 ns
Std. Dev.	69.744 ps	84.556 ps	97.542 ps
Median	1.7759 ns	1.7918 ns	1.8119 ns
MAD	51.292 ps	71.619 ps	89.568 ps

Additional Plots:

- Typical
- Mean
- Std. Dev.
- Median
- MAD
- Slope

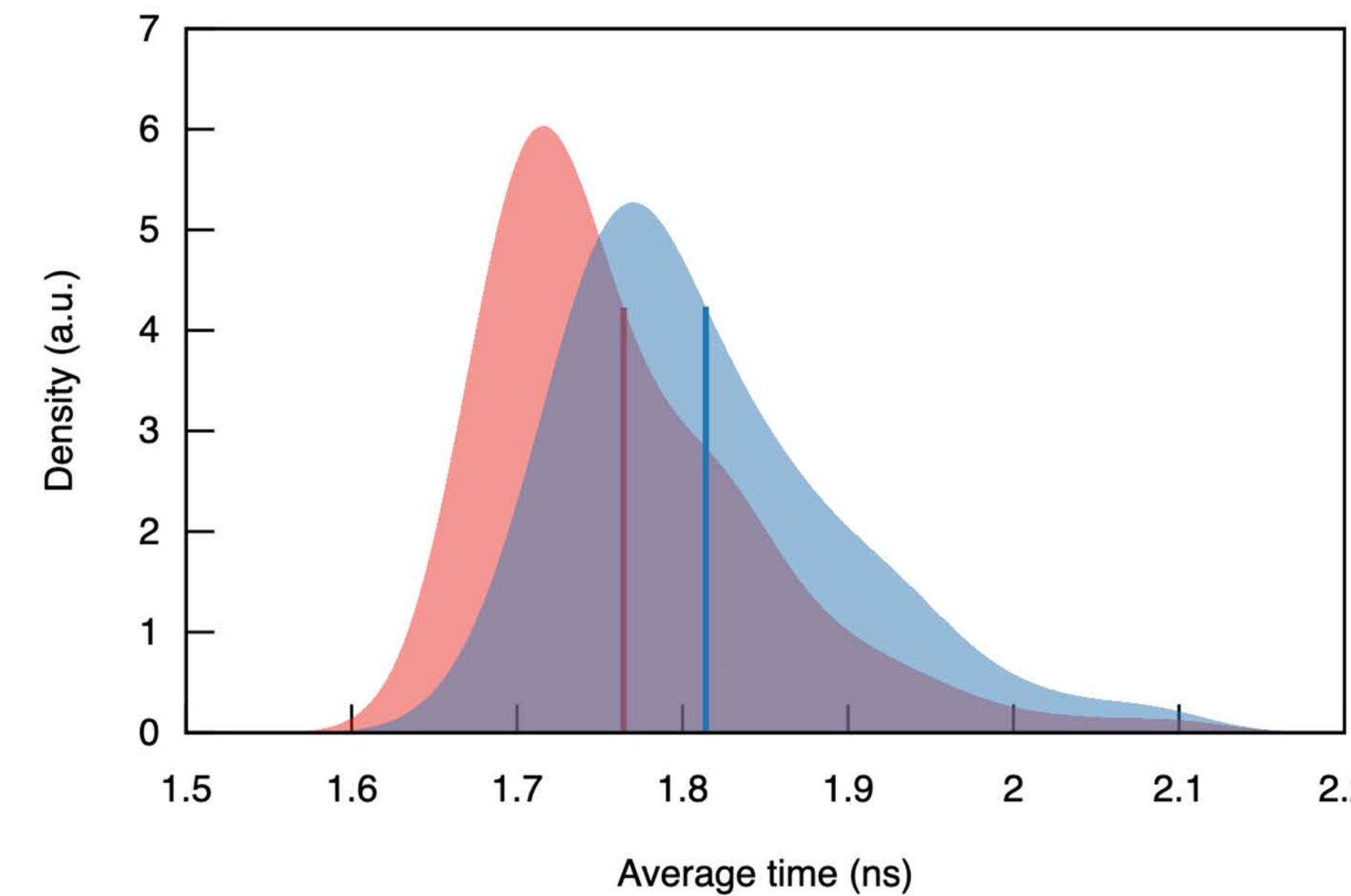
Understanding this report:

The plot on the left displays the average time per iteration for this benchmark. The shaded region shows the estimated probability of an iteration taking a certain amount of time, while the line shows the mean. Click on the plot for a larger view showing the outliers.

The plot on the right shows the linear regression calculated from the measurements. Each point represents a sample, though here it shows the total time for the sample rather than time per iteration. The line is the line of best fit for these measurements.

See [the documentation](#) for more details on the additional statistics.

Change Since Previous Benchmark



Additional Statistics:

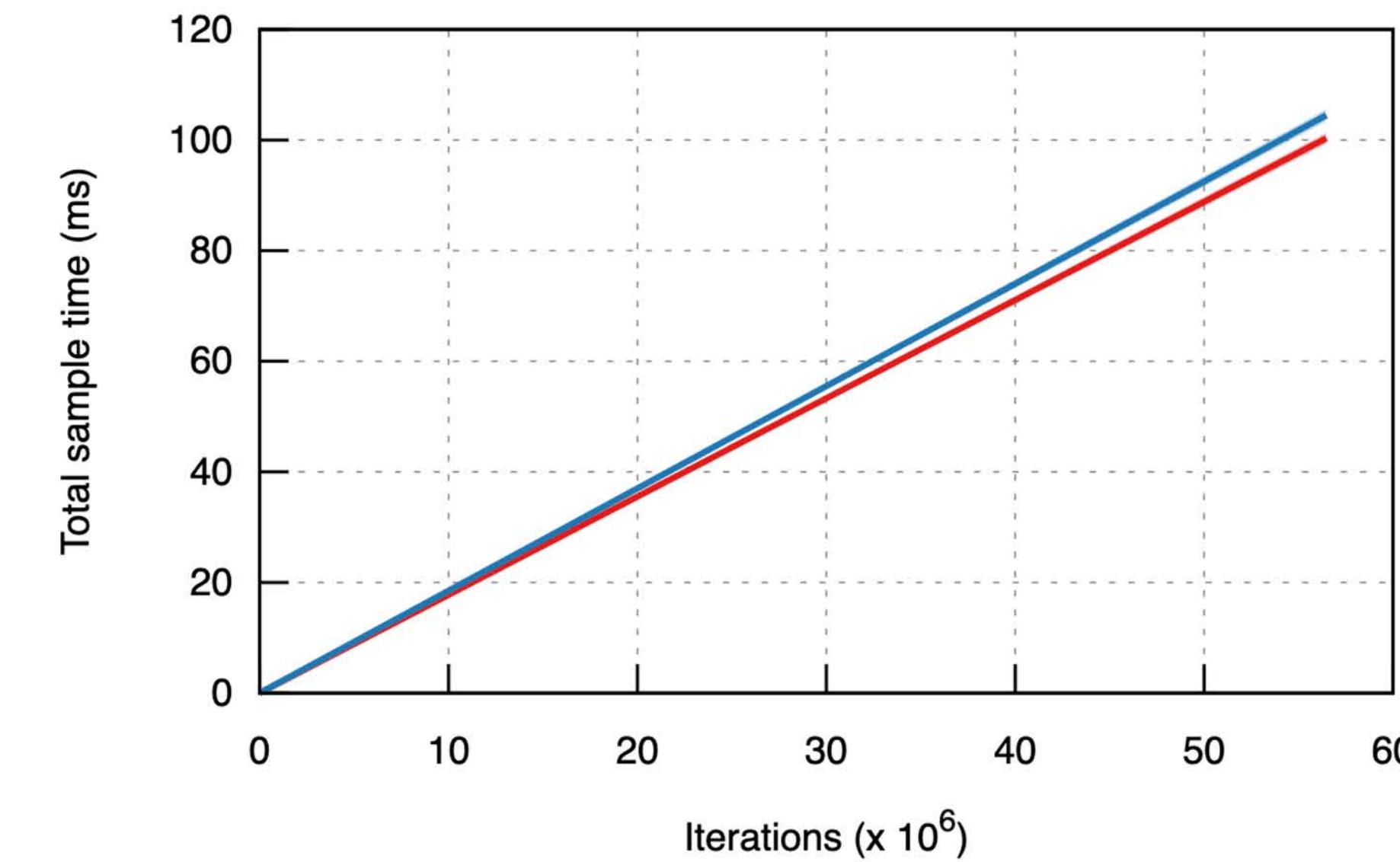
	Lower bound	Estimate	Upper bound	
Change in time	+1.4762%	+2.8152%	+4.1108%	($p = 0.00 < 0.05$)
Performance has regressed.				

Understanding this report:

The plot on the left shows the probability of the function taking a certain amount of time. The red curve represents the saved measurements from the last time this benchmark was run, while the blue curve shows the measurements from this run. The lines represent the mean time per iteration. Click on the plot for a larger view.

The plot on the right shows the two regressions. Again, the red line represents the previous measurement while the blue line shows the current measurement.

See [the documentation](#) for more details on the additional statistics.



Additional Plots:

- Change in mean
- Change in median
- T-Test

Exercise 6

exercises/testing

Questions?

Logging

log

Cargo.toml

```
[package]  
name = "puzzles"  
version = "0.1.0"  
authors = ["Nathan Stocks <nathan@agileperception.com>"]  
edition = "2018"
```

```
[dependencies]
```

```
thiserror = "1.0"
```

Cargo.toml

```
[package]  
name = "puzzles"  
version = "0.1.0"  
authors = ["Nathan Stocks <nathan@agileperception.com>"]  
edition = "2018"
```

```
[dependencies]  
thiserror = "1.0"  
log = "0.4"
```

```
use log::{error, warn, info, debug, trace};
```

```
error!("Serious stuff");
```

```
warn!("Pay attention");
```

```
info!("Useful info");
```

```
debug!("Extra info");
```

```
trace!("All the things");
```

```
use log::{error, warn, info, debug, trace};
```

```
error!("Serious stuff");
```

```
warn!("Pay attention");
```

```
info!("Useful info");
```

```
debug!("Extra info");
```

```
trace!("All the things");
```

```
use log::{error, warn, info, debug, trace};
```

```
error!("Serious stuff");
```

```
warn!("Pay attention");
```

```
info!("Useful info");
```

```
debug!("Extra info");
```

```
trace!("All the things");
```

```
use log::{error, warn, info, debug, trace};
```

```
error!("Serious stuff");
```

```
warn!("Pay attention");
```

```
info!("Useful info");
```

```
debug!("Extra info");
```

```
trace!("All the things");
```

```
use log::{error, warn, info, debug, trace};
```

```
error!("Serious stuff");
```

```
warn!("Pay attention");
```

```
info!("Useful info");
```

```
debug!("Extra info");
```

```
trace!("All the things");
```

```
use log::{error, warn, info, debug, trace};
```

```
error!("Serious stuff");
```

```
warn!("Pay attention");
```

```
info!("Useful info");
```

```
debug!("Extra info");
```

```
trace!("All the things");
```

warn!("Pay attention");

warn!(target: "puzzle", "Pay attention");

warn!(target: "puzzle", "Pay attention");

warn!("Pay attention");

warn!("Pay attention, minion {}!", 352);

puzzles library -> lib.rs

```
use log::{error, info};
```

```
impl Puzzle {
```

```
    /// Make a new puzzle!
```

```
    pub fn new() -> Self {
```

```
        let puzzle = Default::default();
```

```
        info!("Created a puzzle with new(): {:?}", puzzle);
```

```
        puzzle
```

```
}
```

```
    /// Load a puzzle from a file
```

```
    pub fn from_file(_fh: File) -> Result<Self, PuzzleError> {
```

```
        error!("This file is missing a piece!");
```

```
        Err(PuzzleError::MissingPiece)
```

```
}
```

puzzles library -> lib.rs

```
use log::{error, info};
```

```
impl Puzzle {
```

```
    /// Make a new puzzle!
```

```
    pub fn new() -> Self {
```

```
        let puzzle = Default::default();
```

```
        info!("Created a puzzle with new(): {:?}", puzzle);
```

```
        puzzle
```

```
}
```

```
    /// Load a puzzle from a file
```

```
    pub fn from_file(_fh: File) -> Result<Self, PuzzleError> {
```

```
        error!("This file is missing a piece!");
```

```
        Err(PuzzleError::MissingPiece)
```

```
}
```

puzzles library -> lib.rs

```
use log::{error, info};
```

```
impl Puzzle {
```

```
    /// Make a new puzzle!
```

```
    pub fn new() -> Self {
```

```
        let puzzle = Default::default();
```

```
        info!("Created a puzzle with new(): {:?}", puzzle);
```

```
        puzzle
```

```
}
```

```
    /// Load a puzzle from a file
```

```
    pub fn from_file(_fh: File) -> Result<Self, PuzzleError> {
```

```
        error!("This file is missing a piece!");
```

```
        Err(PuzzleError::MissingPiece)
```

```
}
```

puzzles library -> lib.rs

```
use log::{error, info};
```

```
impl Puzzle {
```

```
    /// Make a new puzzle!
```

```
    pub fn new() -> Self {
```

```
        let puzzle = Default::default();
```

```
        info!("Created a puzzle with new(): {:?}", puzzle);
```

```
        puzzle
```

```
}
```

```
    /// Load a puzzle from a file
```

```
    pub fn from_file(_fh: File) -> Result<Self, PuzzleError> {
```

```
        error!("This file is missing a piece!");
```

```
        Err(PuzzleError::MissingPiece)
```

```
}
```

puzzles library -> lib.rs

```
use log::{error, info};
```

```
impl Puzzle {
```

```
    /// Make a new puzzle!
```

```
    pub fn new() -> Self {
```

```
        let puzzle = Default::default();
```

```
        info!("Created a puzzle with new(): {:?}", puzzle);
```

```
        puzzle
```

```
}
```

```
    /// Load a puzzle from a file
```

```
    pub fn from_file(_fh: File) -> Result<Self, PuzzleError> {
```

```
        error!("This file is missing a piece!");
```

```
        Err(PuzzleError::MissingPiece)
```

```
}
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = get_puzzle("puzzle.dat").context("Couldn't get the first puzzle")?;

    println!("Playing puzzle: {}", puzzle.name);

    Ok(())
}
```

[package]

name = "puzzle_game"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

anyhow = "1.0"

puzzles = { path = "../puzzles" }

[package]

name = "puzzle_game"

version = "0.1.0"

authors = ["Nathan Stocks <nathan@agileperception.com>"]

edition = "2018"

[dependencies]

anyhow = "1.0"

puzzles = { path = "../puzzles" }

log = "0.4"

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = get_puzzle("puzzle.dat").context("Couldn't get the first puzzle")?;

    println!("Playing puzzle: {}", puzzle.name);

    Ok(())
}
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = get_puzzle("puzzle.dat").context("Couldn't get the first puzzle")?;

    println!("Playing puzzle: {}", puzzle.name);

    Ok(())
}
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = get_puzzle("puzzle.dat").context("Couldn't get the first puzzle")?;

    println!("Playing puzzle: {}", puzzle.name);
}
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = get_puzzle("puzzle.dat").context("Couldn't get the first puzzle")?;

    info!("Playing puzzle: {}", puzzle.name);
}
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = get_puzzle("puzzle.dat").context("Couldn't get the first puzzle")?;

    info!("Playing puzzle: {}", puzzle.name);
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = match get_puzzle("puzzle.dat").context("Couldn't get the first puzzle") {

        Ok(p) => p,
        Err(e) => return Err(e)
    }

    info!("Found puzzle: {:#?}", puzzle);
}
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = match get_puzzle("puzzle.dat").context("Couldn't get the first puzzle") {

        Ok(p) => p,
        Err(e) => Puzzle::new()
    }

    info!("Solved puzzle: {puzzle:?}");
}
```

puzzle_game app -> main.rs

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = match get_puzzle("puzzle.dat").context("Couldn't get the first puzzle") {

        Ok(p) => p,
        Err(e) -> Puzzle::new()
    }

    info!("Solved puzzle: {puzzle}");
}
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = match get_puzzle("puzzle.dat").context("Couldn't get the first puzzle") {

        Ok(p) => p,
        Err(e) => Puzzle::new()
    }

    info!("Puzzle: {:#?} ", puzzle);
}
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = match get_puzzle("puzzle.dat").context("Couldn't get the first puzzle") {

        Ok(p) => p,
        Err(e) => Puzzle::new()
    }

    info!("got puzzle: {:#?}", puzzle);
}
```

```
$ cargo run  
Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
Running `target/debug/puzzle_game`
```

```
$
```

```
$ cargo run  
Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
Running `target/debug/puzzle_game`
```

```
$ cargo run  
Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
Running `target/debug/puzzle_game`
```

```
$
```

```
$ cargo run  
Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
Running `target/debug/puzzle_game`
```

```
$ cargo run  
Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
Running `target/debug/puzzle_game`
```

```
$ cargo run  
Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
Running `target/debug/puzzle_game`
```

```
$
```

```
$ cargo run  
  Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
  Running `target/debug/puzzle_game`
```

```
$ cargo run  
  Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
  Running `target/debug/puzzle_game`
```

```
$ cargo run  
  Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
  Running `target/debug/puzzle_game`
```

```
$ cargo run  
  Finished dev [unoptimized + debuginfo] target(s) in 0.01s  
  Running `target/debug/puzzle_game`
```

```
$
```

Interface





Available logging implementations

In order to produce log output executables have to use a logger implementation compatible with the facade. There are many available implementations to choose from, here are some of the most popular ones:

- Simple minimal loggers:
 - [env_logger](#)
 - [simple_logger](#)
 - [simplelog](#)
 - [pretty_env_logger](#)
 - [stderrlog](#)
 - [flexi_logger](#)
- Complex configurable frameworks:
 - [log4rs](#)
 - [fern](#)
- Adaptors for other facilities:
 - [syslog](#)
 - [slog-stdlog](#)

env_logger

puzzle_game app -> Cargo.toml

```
[package]
```

```
name = "puzzle_game"
```

```
version = "0.1.0"
```

```
authors = ["Nathan Stocks <nathan@agileperception.com>"]
```

```
edition = "2018"
```

```
[dependencies]
```

```
anyhow = "1.0"
```

```
puzzles = { path = "../puzzles" }
```

```
log = "0.4"
```

puzzle_game app -> Cargo.toml

```
[package]  
  
name = "puzzle_game"  
version = "0.1.0"  
authors = ["Nathan Stocks <nathan@agileperception.com>"]  
edition = "2018"  
  
[dependencies]  
  
anyhow = "1.0"  
puzzles = { path = "../puzzles" }  
log = "0.4"  
env_logger = "0.8"
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = match get_puzzle("puzzle.dat").context("Couldn't get the first puzzle") {

        Ok(p) => p,
        Err(e) => Puzzle::new()
    }

    info!("got puzzle: {:#?}", puzzle);
}
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    env_logger::init();

    let puzzle = match get_puzzle("puzzle.dat").context("Couldn't get the first puzzle") {
```

puzzle_game app -> main.rs

```
use anyhow::{Context, Result};

use log::info;

use puzzles::Puzzle;

use std::fs::File;

fn get_puzzle(filename: &str) -> Result<Puzzle> {

    let fh = File::open(filename)

        .with_context(|| format!("couldn't open the puzzle file {}", filename))?;

    let puzzle = Puzzle::from_file(fh).context("couldn't convert data into a puzzle")?;

    Ok(puzzle)
}

fn main() -> Result<()> {

    let puzzle = match get_puzzle("puzzle.dat").context("Couldn't get the first puzzle") {

        Ok(p) => p,
        Err(e) => Puzzle::new()
    }

    info!("got puzzle: {:#?}", puzzle);
}
```

```
$ cargo run
```

```
[snip compiling new dependencies]
```

```
Finished dev [unoptimized + debuginfo] target(s) in 0.02s
```

```
Running `target/debug/puzzle_game`
```

```
[2021-06-04T19:20:38Z ERROR puzzles] This file is missing a piece!
```

```
$
```

```
$ cargo run
```

```
[snip compiling new dependencies]
```

```
Finished dev [unoptimized + debuginfo] target(s) in 0.02s
```

```
Running `target/debug/puzzle_game`
```

```
[2021-06-04T19:20:38Z ERROR puzzles] This file is missing a piece!
```

```
$ RUST_LOG=info cargo run
```

```
$ cargo run  
[snip compiling new dependencies]
```

Finished dev [unoptimized + debuginfo] target(s) in 0.02s

Running `target/debug/puzzle_game`

```
[2021-06-04T19:20:38Z ERROR puzzles] This file is missing a piece!
```

```
$ RUST_LOG=info cargo run
```

Finished dev [unoptimized + debuginfo] target(s) in 0.03s

Running `target/debug/puzzle_game`

```
[2021-06-04T19:23:53Z ERROR puzzles] This file is missing a piece!
```

```
[2021-06-04T19:23:53Z INFO puzzles] Created a puzzle with new(): Puzzle { num_pieces: 42, name: "Forest Lake" }
```

```
[2021-06-04T19:23:53Z INFO puzzle_game] Playing puzzle: Forest Lake
```

```
$
```

```
$ cargo run  
[snip compiling new dependencies]  
  Finished dev [unoptimized + debuginfo] target(s) in 0.02s  
    Running `target/debug/puzzle_game`  
[2021-06-04T19:20:38Z ERROR puzzles] This file is missing a piece!
```

```
$ RUST_LOG=info cargo run  
  Finished dev [unoptimized + debuginfo] target(s) in 0.03s  
    Running `target/debug/puzzle_game`  
[2021-06-04T19:23:53Z ERROR puzzles] This file is missing a piece!  
[2021-06-04T19:23:53Z INFO puzzles] Created a puzzle with new(): Puzzle { num_pieces: 42, name: "Forest Lake" }  
[2021-06-04T19:23:53Z INFO puzzle_game] Playing puzzle: Forest Lake
```

```
$
```

```
$ cargo run  
[snip compiling new dependencies]  
  Finished dev [unoptimized + debuginfo] target(s) in 0.02s  
    Running `target/debug/puzzle_game`  
[2021-06-04T19:20:38Z ERROR puzzles] This file is missing a piece!
```

```
$ RUST_LOG=info cargo run  
  Finished dev [unoptimized + debuginfo] target(s) in 0.03s  
    Running `target/debug/puzzle_game`  
[2021-06-04T19:23:53Z ERROR puzzles] This file is missing a piece!  
[2021-06-04T19:23:53Z INFO  puzzles] Created a puzzle with new(): Puzzle { num_pieces: 42, name: "Forest Lake" }  
[2021-06-04T19:23:53Z INFO  puzzle_game] Playing puzzle: Forest Lake
```

```
$
```

```
$ cargo run  
[snip compiling new dependencies]
```

Finished dev [unoptimized + debuginfo] target(s) in 0.02s

Running `target/debug/puzzle_game`

```
[2021-06-04T19:20:38Z ERROR puzzles] This file is missing a piece!
```

```
$ RUST_LOG=info cargo run
```

Finished dev [unoptimized + debuginfo] target(s) in 0.03s

Running `target/debug/puzzle_game`

```
[2021-06-04T19:23:53Z ERROR puzzles] This file is missing a piece!
```

```
[2021-06-04T19:23:53Z INFO puzzles] Created a puzzle with new(): Puzzle { num_pieces: 42, name: "Forest Lake" }
```

```
[2021-06-04T19:23:53Z INFO puzzle_game] Playing puzzle: Forest Lake
```

```
$
```

tracing

Exercise 7

exercises/logging

Questions?

10 Minute Break

Multithreading

Process

Process

Thread A

Process

Thread A

Thread B

Process

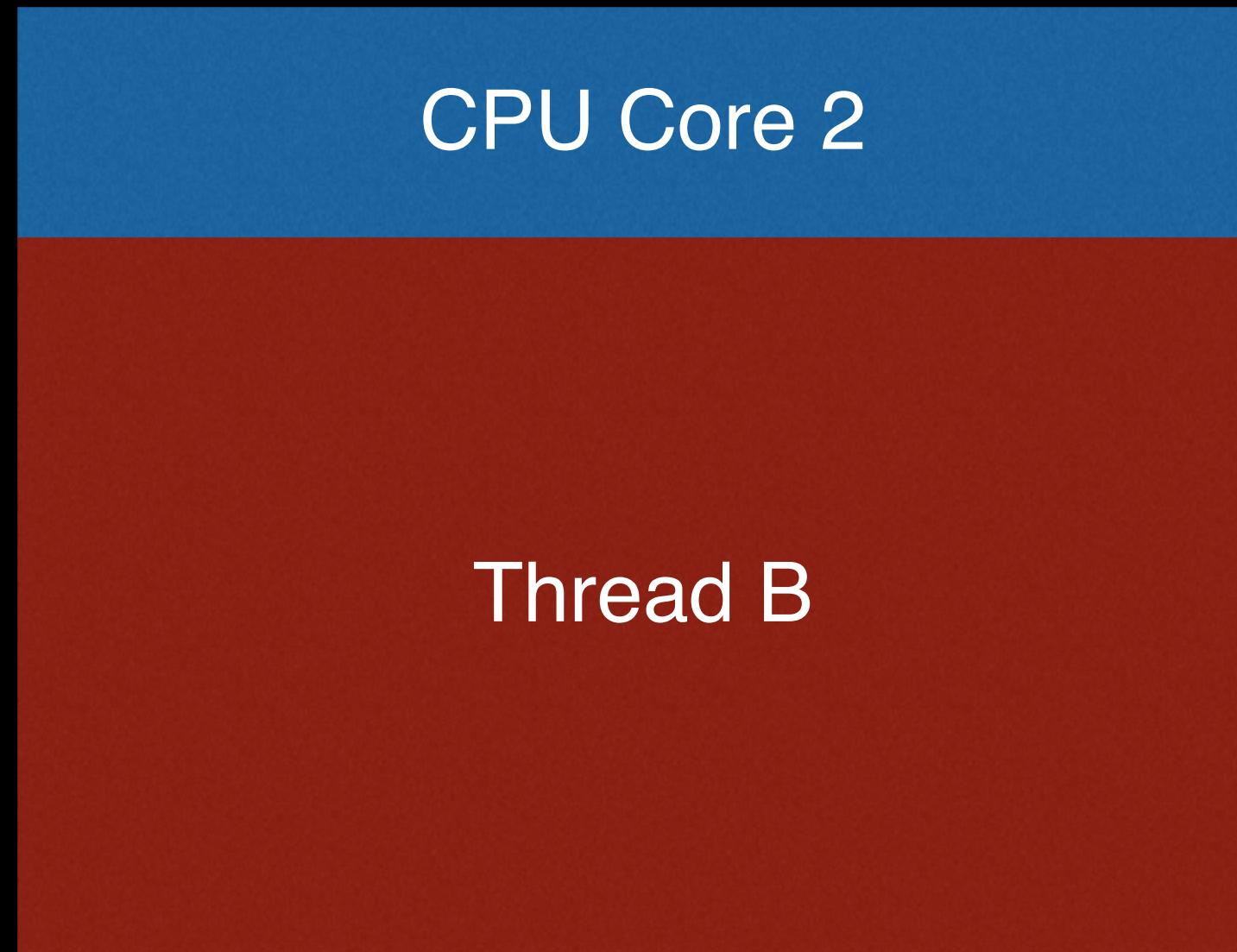
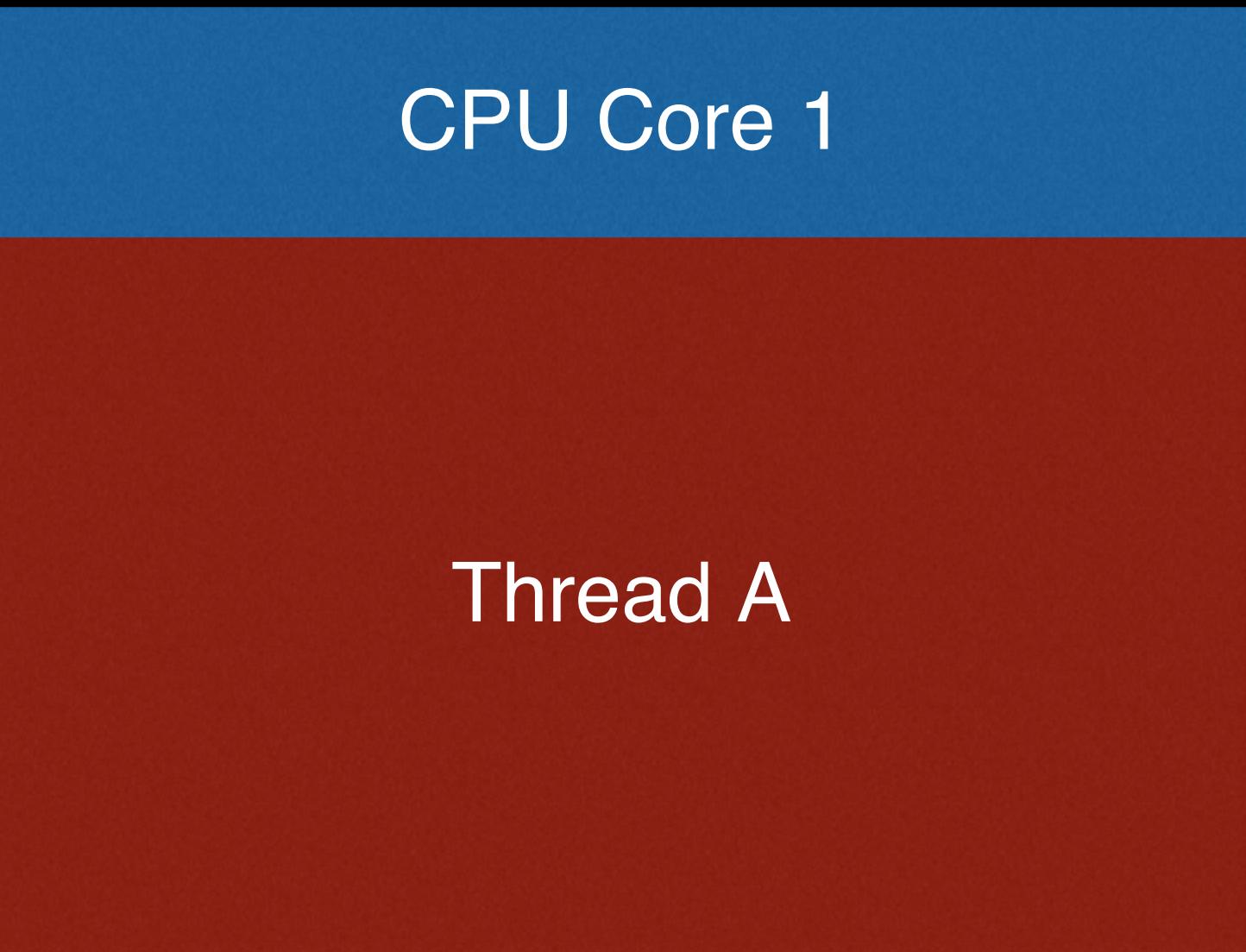
Thread A

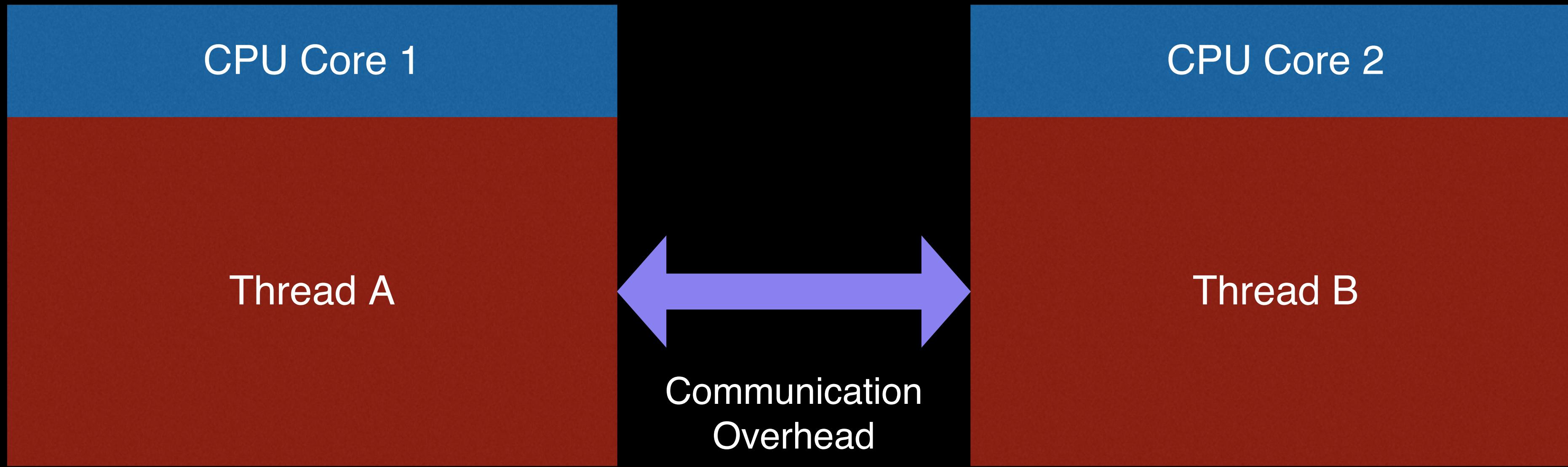
Shared
Memory

Thread B

CPU Core 1

CPU Core 2





Parallel Processing



kitchen -> Cargo.toml

```
[package]
name = "kitchen"
version = "0.1.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
edition = "2018"
```

```
[dependencies]
log = "0.4"
env_logger = "0.8"
```

kitchen -> Cargo.toml

```
[package]
name = "kitchen"
version = "0.1.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
edition = "2018"
```

```
[dependencies]
log = "0.4"
env_logger = "0.8"
```

kitchen -> main.rs

```
use log::{error, info};
use std::thread, time::Duration;

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

kitchen -> main.rs

```
use log::{error, info};
use std::{thread, time::Duration};

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

kitchen -> main.rs

```
use log::{error, info};
use std::thread, time::Duration;

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

kitchen -> main.rs

```
use log::{error, info};  
use std::{thread, time::Duration};  
  
fn sleep(seconds: f32) {  
    thread::sleep(Duration::from_secs_f32(seconds));  
}
```

```
pub mod dad {  
    use super::{info, sleep};  
  
    pub fn cook_spaghetti() -> bool {  
        info!("Cooking the spaghetti...");  
        sleep(4.0);  
        info!("Spaghetti is ready!");  
        true  
    }  
}
```

kitchen -> main.rs

```
use log::{error, info};
use std::thread, time::Duration;

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

kitchen -> main.rs

```
use log::{error, info};
use std::thread, time::Duration;

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

kitchen -> main.rs

```
use log::{error, info};
use std::thread, time::Duration;

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

kitchen -> main.rs

```
use log::{error, info};
use std::thread, time::Duration;

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

kitchen -> main.rs

```
use log::{error, info};
use std::thread, time::Duration;

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

kitchen -> main.rs

```
use log::{error, info};
use std::thread, time::Duration;

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

kitchen -> main.rs

```
use log::{error, info};
use std::thread, time::Duration;

fn sleep(seconds: f32) {
    thread::sleep(Duration::from_secs_f32(seconds));
}

pub mod dad {
    use super::{info, sleep};

    pub fn cook_spaghetti() -> bool {
        info!("Cooking the spaghetti...");
        sleep(4.0);
        info!("Spaghetti is ready!");
        true
    }
}
```

Send

kitchen -> main.rs

```
pub mod mom {  
    use super::{info, sleep};  
  
    pub fn cook_sauce_and_set_table() {  
        sleep(1.0);  
        info!("Cooking the sauce...");  
        sleep(2.0);  
        info!("Sauce is ready! Setting the table...");  
        sleep(2.0);  
        info!("Table is set!");  
    }  
}
```

kitchen -> main.rs

```
pub mod mom {  
    use super::{info, sleep};  
  
    pub fn cook_sauce_and_set_table() {  
        sleep(1.0);  
        info!("Cooking the sauce...");  
        sleep(2.0);  
        info!("Sauce is ready! Setting the table...");  
        sleep(2.0);  
        info!("Table is set!");  
    }  
}
```

kitchen -> main.rs

```
pub mod mom {  
    use super::{info, sleep};  
  
    pub fn cook_sauce_and_set_table() {  
        sleep(1.0);  
        info!("Cooking the sauce...");  
        sleep(2.0);  
        info!("Sauce is ready! Setting the table...");  
        sleep(2.0);  
        info!("Table is set!");  
    }  
}
```

kitchen -> main.rs

```
pub mod mom {  
    use super::{info, sleep};  
  
    pub fn cook_sauce_and_set_table() {  
        sleep(1.0);  
        info!("Cooking the sauce...");  
        sleep(2.0);  
        info!("Sauce is ready! Setting the table...");  
        sleep(2.0);  
        info!("Table is set!");  
    }  
}
```

kitchen -> main.rs

```
pub mod mom {  
    use super::{info, sleep};  
  
    pub fn cook_sauce_and_set_table() {  
        sleep(1.0);  
        info!("Cooking the sauce...");  
        sleep(2.0);  
        info!("Sauce is ready! Setting the table...");  
        sleep(2.0);  
        info!("Table is set!");  
    }  
}
```

kitchen -> main.rs

```
fn main() {  
    env_logger::init();  
  
    let handle = thread::spawn(|| dad::cook_spaghetti());  
  
    mom::cook_sauce_and_set_table();  
  
    if handle.join().unwrap_or(false) {  
        info!("Spaghetti time! Yum!")  
    } else {  
        error!("Dad messed up the spaghetti. Order pizza instead?");  
    }  
}
```

kitchen -> main.rs

```
fn main() {  
    env_logger::init();  
  
    let handle = thread::spawn(|| dad::cook_spaghetti());  
  
    mom::cook_sauce_and_set_table();  
  
    if handle.join().unwrap_or(false) {  
        info!("Spaghetti time! Yum!")  
    } else {  
        error!("Dad messed up the spaghetti. Order pizza instead?");  
    }  
}
```

kitchen -> main.rs

```
fn main() {  
    env_logger::init();  
  
    let handle = thread::spawn(|| dad::cook_spaghetti());  
  
    mom::cook_sauce_and_set_table();  
  
    if handle.join().unwrap_or(false) {  
        info!("Spaghetti time! Yum!")  
    } else {  
        error!("Dad messed up the spaghetti. Order pizza instead?");  
    }  
}
```

kitchen -> main.rs

```
fn main() {  
    env_logger::init();  
  
    let handle = thread::spawn(|| dad::cook_spaghetti());  
  
    mom::cook_sauce_and_set_table();  
  
    if handle.join().unwrap_or(false) {  
        info!("Spaghetti time! Yum!")  
    } else {  
        error!("Dad messed up the spaghetti. Order pizza instead?");  
    }  
}
```

kitchen -> main.rs

```
fn main() {  
    env_logger::init();  
  
    let handle = thread::spawn(|| dad::cook_spaghetti());  
  
    mom::cook_sauce_and_set_table();  
  
    if handle.join().unwrap_or(false) {  
        info!("Spaghetti time! Yum!")  
    } else {  
        error!("Dad messed up the spaghetti. Order pizza instead?");  
    }  
}
```

kitchen -> main.rs

```
fn main() {  
    env_logger::init();  
  
    let handle = thread::spawn(|| dad::cook_spaghetti());  
  
    mom::cook_sauce_and_set_table();  
  
    if handle.join().unwrap_or(false) {  
        info!("Spaghetti time! Yum!")  
    } else {  
        error!("Dad messed up the spaghetti. Order pizza instead?");  
    }  
}
```

kitchen -> main.rs

```
fn main() {  
    env_logger::init();  
  
    let handle = thread::spawn(|| dad::cook_spaghetti());  
  
    mom::cook_sauce_and_set_table();  
  
    if handle.join().unwrap_or(false) {  
        info!("Spaghetti time! Yum!");  
    } else {  
        error!("Dad messed up the spaghetti. Order pizza instead?");  
    }  
}
```

kitchen -> main.rs

```
fn main() {  
    env_logger::init();  
  
    let handle = thread::spawn(|| dad::cook_spaghetti());  
  
    mom::cook_sauce_and_set_table();  
  
    if handle.join().unwrap_or(false) {  
        info!("Spaghetti time! Yum!")  
    } else {  
        error!("Dad messed up the spaghetti. Order pizza instead?");  
    }  
}
```

kitchen -> main.rs

```
fn main() {  
    env_logger::init();  
  
    let handle = thread::spawn(|| dad::cook_spaghetti());  
  
    mom::cook_sauce_and_set_table();  
  
    if handle.join().unwrap_or(false) {  
        info!("Spaghetti time! Yum!")  
    } else {  
        error!("Dad messed up the spaghetti. Order pizza instead?");  
    }  
}
```

```
$ RUST_LOG=info cargo run
```

```
$ RUST_LOG=info cargo run
Finished dev [unoptimized + debuginfo] target(s) in 0.01s
Running `target/debug/kitchen`
[2021-06-10T02:22:55Z INFO kitchen::dad] Cooking the spaghetti...
[2021-06-10T02:22:56Z INFO kitchen::mom] Cooking the sauce...
[2021-06-10T02:22:58Z INFO kitchen::mom] Sauce is ready! Setting the table...
[2021-06-10T02:22:59Z INFO kitchen::dad] Spaghetti is ready!
[2021-06-10T02:23:00Z INFO kitchen::mom] Table is set!
[2021-06-10T02:23:00Z INFO kitchen] Spaghetti time! Yum!
```

\$

```
$ RUST_LOG=info cargo run
Finished dev [unoptimized + debuginfo] target(s) in 0.01s
Running `target/debug/kitchen`
[2021-06-10T02:22:55Z INFO kitchen::dad] Cooking the spaghetti...
[2021-06-10T02:22:56Z INFO kitchen::mom] Cooking the sauce...
[2021-06-10T02:22:58Z INFO kitchen::mom] Sauce is ready! Setting the table...
[2021-06-10T02:22:59Z INFO kitchen::dad] Spaghetti is ready!
[2021-06-10T02:23:00Z INFO kitchen::mom] Table is set!
[2021-06-10T02:23:00Z INFO kitchen] Spaghetti time! Yum!
```

\$

```
$ RUST_LOG=info cargo run
  Finished dev [unoptimized + debuginfo] target(s) in 0.01s
    Running `target/debug/kitchen`
[2021-06-10T02:22:55Z INFO kitchen::dad] Cooking the spaghetti...
[2021-06-10T02:22:56Z INFO kitchen::mom] Cooking the sauce...
[2021-06-10T02:22:58Z INFO kitchen::mom] Sauce is ready! Setting the table...
[2021-06-10T02:22:59Z INFO kitchen::dad] Spaghetti is ready!
[2021-06-10T02:23:00Z INFO kitchen::mom] Table is set!
[2021-06-10T02:23:00Z INFO kitchen] Spaghetti time! Yum!
```

\$

```
$ RUST_LOG=info cargo run
Finished dev [unoptimized + debuginfo] target(s) in 0.01s
Running `target/debug/kitchen`
[2021-06-10T02:22:55Z INFO  kitchen::dad] Cooking the spaghetti...
[2021-06-10T02:22:56Z INFO  kitchen::mom] Cooking the sauce...
[2021-06-10T02:22:58Z INFO  kitchen::mom] Sauce is ready! Setting the table...
[2021-06-10T02:22:59Z INFO  kitchen::dad] Spaghetti is ready!
[2021-06-10T02:23:00Z INFO  kitchen::mom] Table is set!
[2021-06-10T02:23:00Z INFO  kitchen] Spaghetti time! Yum!
```

\$

```
$ RUST_LOG=info cargo run
Finished dev [unoptimized + debuginfo] target(s) in 0.01s
Running `target/debug/kitchen`
[2021-06-10T02:22:55Z INFO kitchen::dad] Cooking the spaghetti...
[2021-06-10T02:22:56Z INFO kitchen::mom] Cooking the sauce...
[2021-06-10T02:22:58Z INFO kitchen::mom] Sauce is ready! Setting the table...
[2021-06-10T02:22:59Z INFO kitchen::dad] Spaghetti is ready!
[2021-06-10T02:23:00Z INFO kitchen::mom] Table is set!
[2021-06-10T02:23:00Z INFO kitchen] Spaghetti time! Yum!
```

\$

```
$ RUST_LOG=info cargo run
  Finished dev [unoptimized + debuginfo] target(s) in 0.01s
    Running `target/debug/kitchen`
[2021-06-10T02:22:55Z INFO kitchen::dad] Cooking the spaghetti...
[2021-06-10T02:22:56Z INFO kitchen::mom] Cooking the sauce...
[2021-06-10T02:22:58Z INFO kitchen::mom] Sauce is ready! Setting the table...
[2021-06-10T02:22:59Z INFO kitchen::dad] Spaghetti is ready!
[2021-06-10T02:23:00Z INFO kitchen::mom] Table is set!
[2021-06-10T02:23:00Z INFO kitchen] Spaghetti time! Yum!
```

\$



Questions?

Channels

std::sync::mpsc

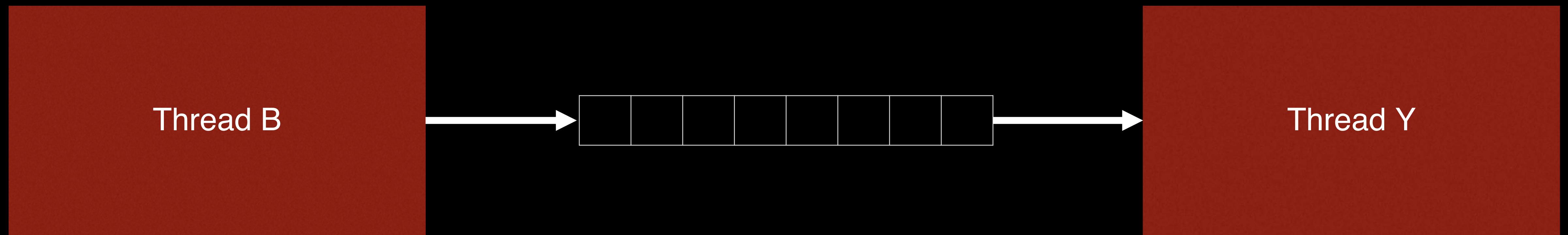


std::sync::mpsc

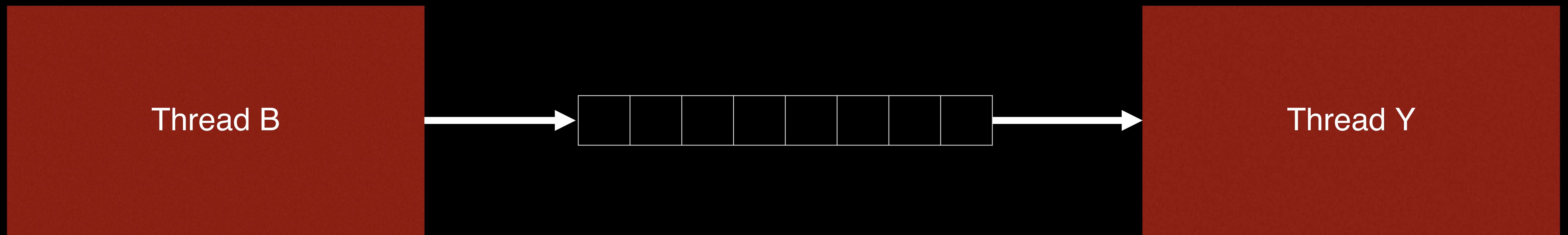


std::sync::mpsc

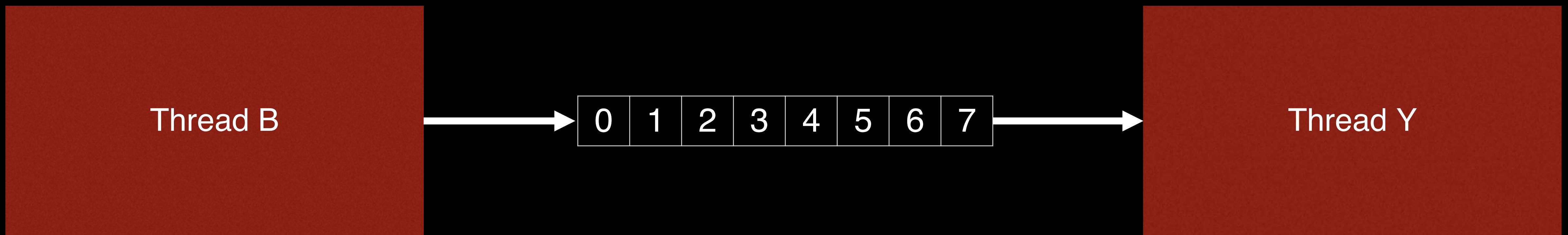
crossbeam::channel



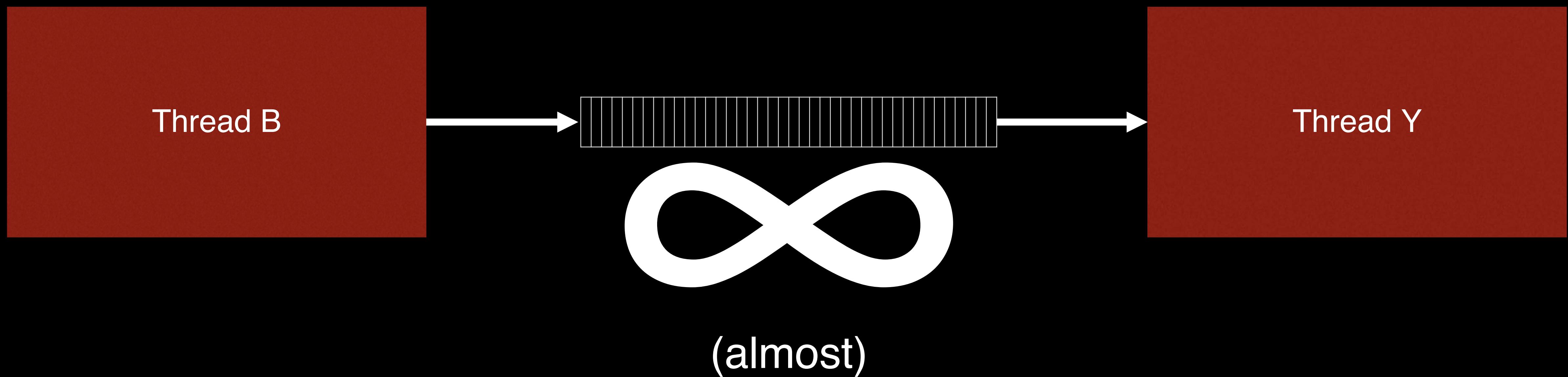
Send

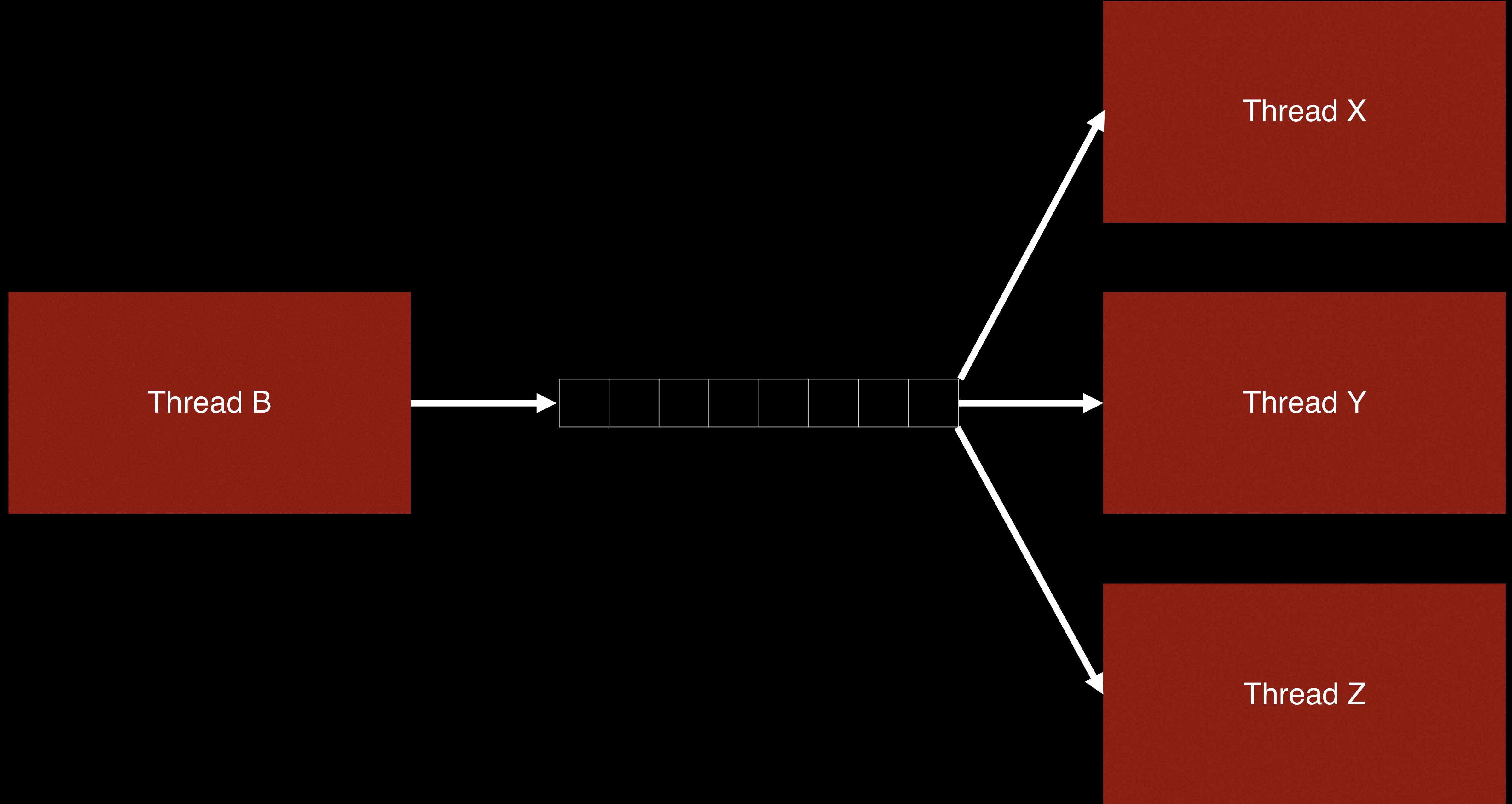


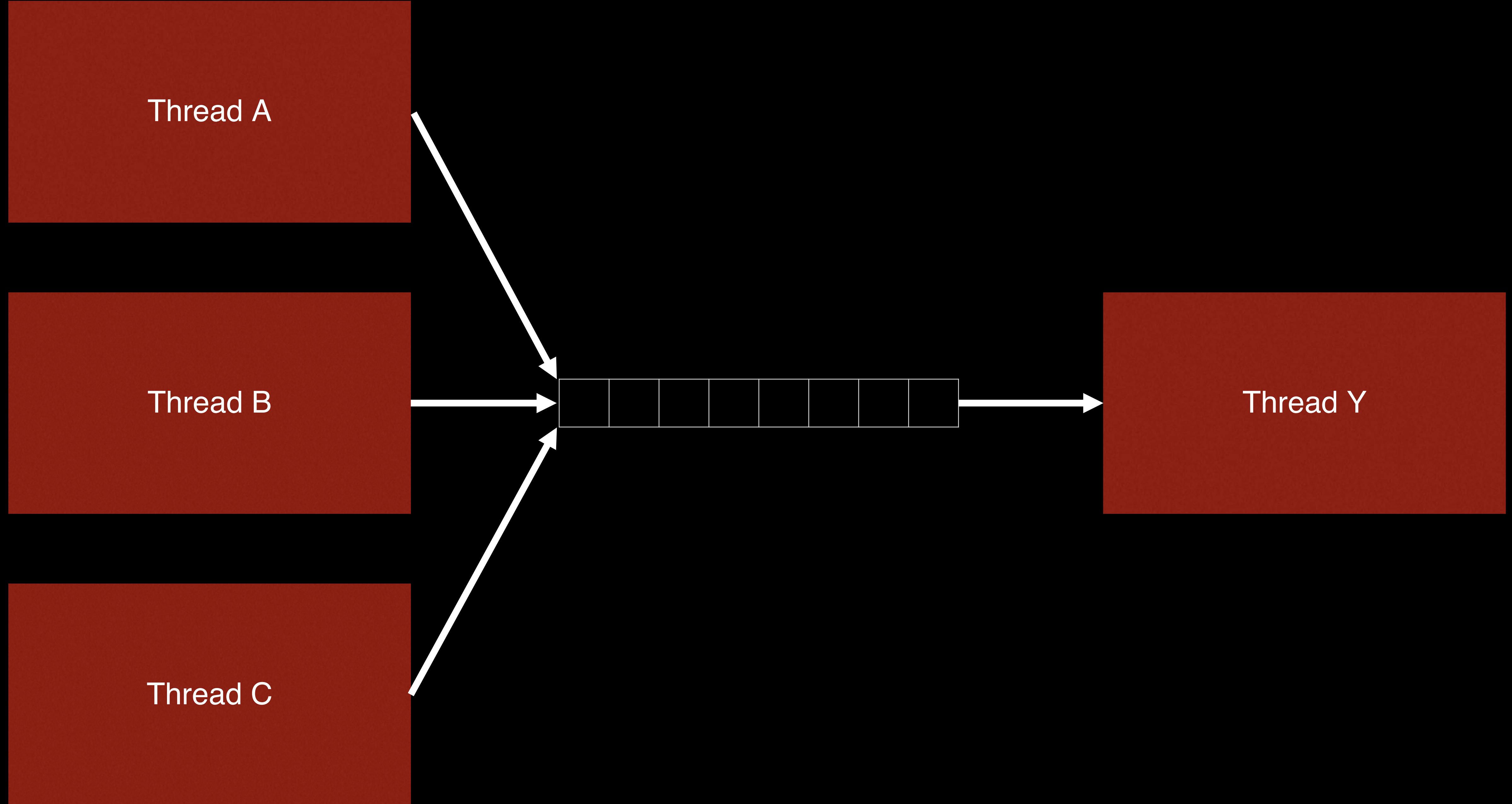
channel::bounded(8)

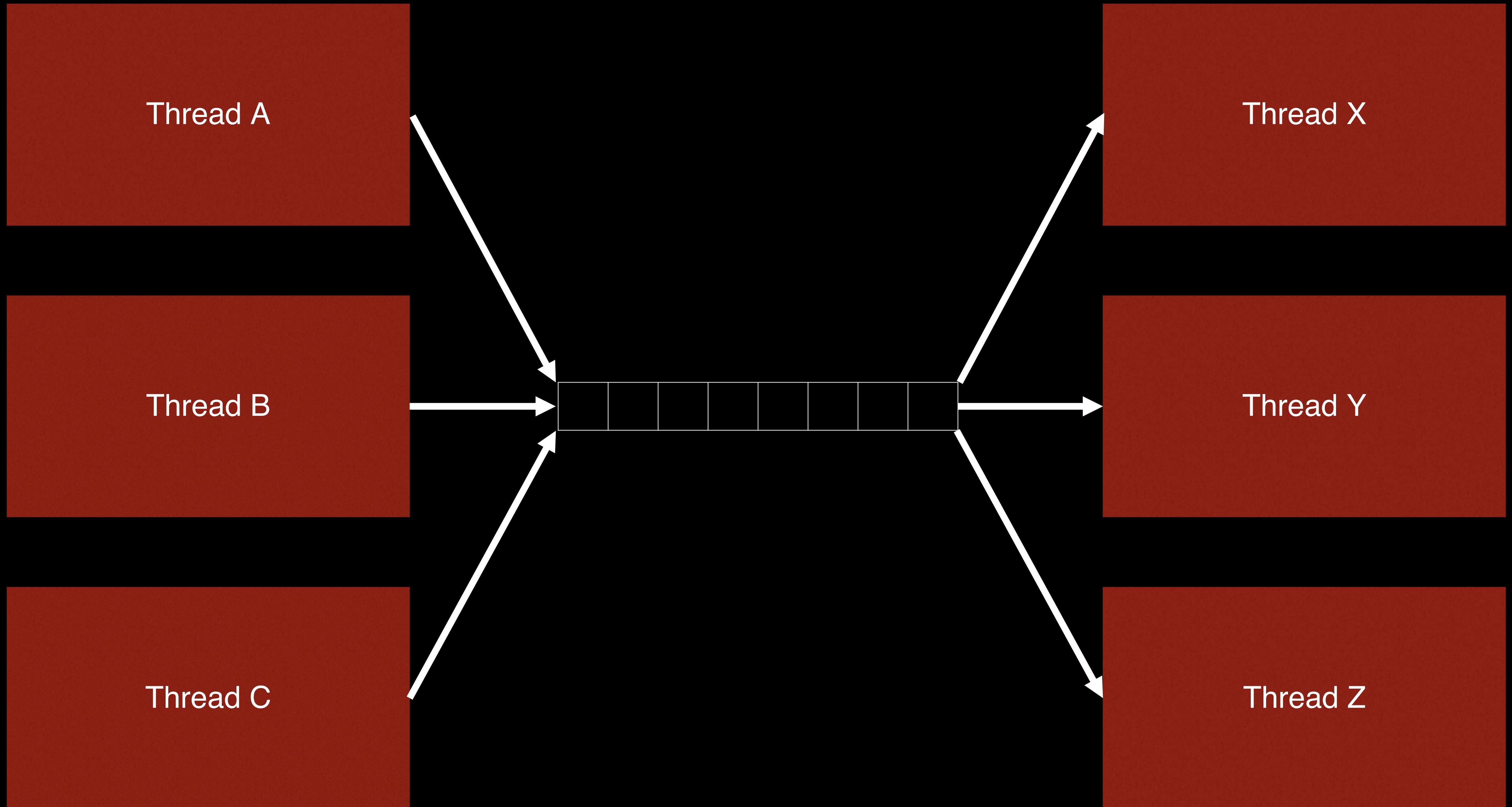


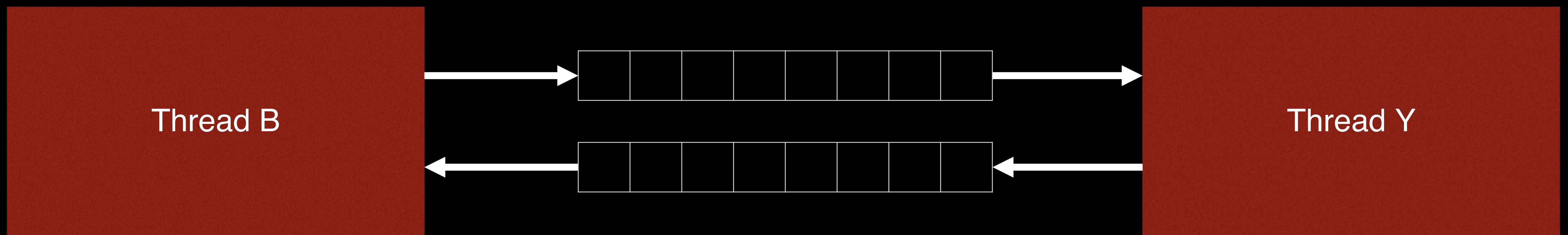
channel::unbounded()

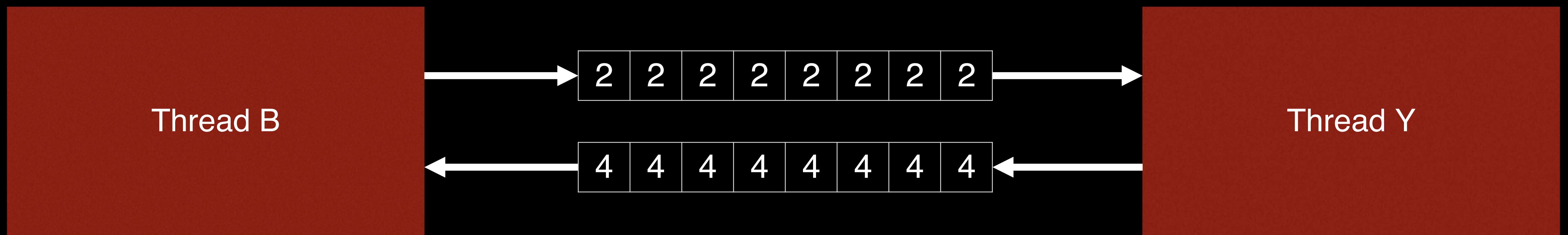












cafeteria -> Cargo.toml

```
[package]
name = "cafeteria"
version = "0.1.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
edition = "2018"
```

```
[dependencies]
crossbeam = "0.8"
```

cafeteria -> Cargo.toml

```
[package]
name = "cafeteria"
version = "0.1.0"
authors = ["Nathan Stocks <nathan@agileperception.com>"]
edition = "2018"
```

```
[dependencies]
crossbeam = "0.8"
```

cafeteria -> main.rs

```
use crossbeam::channel::{self, Receiver, Sender};  
use std::{thread, time::Duration};
```

```
#[derive(Debug)]  
enum Lunch {  
    Soup,  
    Salad,  
    Sandwich,  
    HotDog,  
}
```

cafeteria -> main.rs

```
use crossbeam::channel::{self, Receiver, Sender};  
use std::{thread, time::Duration};  
  
#[derive(Debug)]  
enum Lunch {  
    Soup,  
    Salad,  
    Sandwich,  
    HotDog,  
}  
}
```

cafeteria -> main.rs

```
use crossbeam::channel::{self, Receiver, Sender};  
use std::{thread, time::Duration};
```

```
#[derive(Debug)]  
enum Lunch {  
    Soup,  
    Salad,  
    Sandwich,  
    HotDog,  
}
```

cafeteria -> main.rs

```
use crossbeam::channel::{self, Receiver, Sender};  
use std::{thread, time::Duration};
```

```
#[derive(Debug)]  
enum Lunch {  
    Soup,  
    Salad,  
    Sandwich,  
    HotDog,  
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn cafeteria_worker(name: &str, orders: Receiver<&str>, lunches: Sender<Lunch>) {
    for order in orders {
        println!("{} receives an order for {}", name, order);
        let lunch = match &order {
            x if x.contains("soup") => Lunch::Soup,
            x if x.contains("salad") => Lunch::Salad,
            x if x.contains("sandwich") => Lunch::Sandwich,
            _ => Lunch::HotDog,
        };
        for _ in 0..order.len() {
            thread::sleep(Duration::from_secs_f32(0.1))
        }
        println!("{} sends a {:?}", name, lunch);
        if lunches.send(lunch).is_err() {
            break;
        }
    }
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

```
pub fn drop<T>(_x: T) {}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

cafeteria -> main.rs

```
fn main() {
    let (orders_tx, orders_rx) = channel::unbounded();
    let orders_rx2 = orders_rx.clone();
    let (lunches_tx, lunches_rx) = channel::unbounded();
    let lunches_tx2 = lunches_tx.clone();

    let alice_handle = thread::spawn(|| cafeteria_worker("alice", orders_rx2, lunches_tx2));
    let zack_handle = thread::spawn(|| cafeteria_worker("zack", orders_rx, lunches_tx));

    for order in vec!["polish dog", "caesar salad", "onion soup", "reuben sandwich"] {
        println!("ORDER: {}", order);
        let _ = orders_tx.send(order);
    }
    drop(orders_tx);

    for lunch in lunches_rx {
        println!("Order Up! -> {:?}", lunch);
    }

    let _ = alice_handle.join();
    let _ = zack_handle.join();
}
```

```
$ cargo run
```

```
$ cargo run
```

```
[snip]
```

```
ORDER: polish dog
```

```
ORDER: caesar salad
```

```
ORDER: onion soup
```

```
ORDER: reuben sandwich
```

```
zack receives an order for caesar salad
```

```
alice receives an order for polish dog
```

```
alice sends a HotDog
```

```
alice receives an order for onion soup
```

```
Order Up! -> HotDog
```

```
zack sends a Salad
```

```
zack receives an order for reuben sandwich
```

```
Order Up! -> Salad
```

```
alice sends a Soup
```

```
Order Up! -> Soup
```

```
zack sends a Sandwich
```

```
Order Up! -> Sandwich
```

```
$
```

Exercise 8

exercises/threads_channels

Questions?

Now what?

What did we miss?

Lots more Traits

File I/O

Slice

Box

Mutex

Rc/Arc

Cell/RefCell

Args / Vars

Hasher

TCP / UDP

Command

Instant / Duration

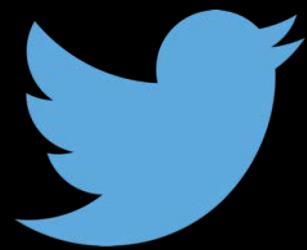
...and that's just the notable intermediate stuff

A close-up photograph of a large, weathered metal gear assembly. The main gear is massive, showing significant signs of rust and paint peeling, particularly along its left edge. Behind it, a smaller, also rusted gear is visible. A vertical metal plate with two circular rivets or bolts secures the gears together. The entire assembly is set against a solid black background.

Week 3

Rusty Engine

Questions?



@nathanstocks



github.com/CleanCut



agileperception.com



patreon.com/nathanstocks

Thank You

