Rust Workshop

Day 4

Recap of Day 3

Generics

```
fn my_unwrap<T>(maybe_val: Option<T>) -> T {
    maybe_val.unwrap()
}
impl<T> Option<T> {
    fn unwrap(self) -> T {
        // ...
    }
}
enum Result<T, E> {
    Ok(T),
    Err(E),
}
```

Traits + Bounds

```
trait Comparable {
    fn is_greater_than(&self, other: &Self) -> bool;

    fn is_less_than_or_equal(&self, other: &Self) -> bool {
        !self.is_greater_than(other)
    }
}
fn find_largest<T>(list: &[T]) -> &T
where
    T: Comparable,
{ /**/ }
```

Lifetime Annotations

```
1  // possible
2  fn longest<'a>(x: &'a str, y: &'a str) -> &'a str {}
3
4  // recommended
5  fn longest(x: &str, y: &str) -> String {}
```

Closures

```
1  fn main() {
2      let x = 3;
3      let mut nums = vec![1, 2, 3, 4, 5, 6, 7, 8, 9, 10];
4
5      nums.retain(|elem| elem % x == 0);
6  }
```

Iterators

```
trait Iterator {
    type Item;
    fn next(&mut self) -> Option<Self::Item>;
}
fn main() {
    let numbers = vec![1, 2, 3, 4, 5];

    let mut iter = numbers.into_iter();
    while let Some(num) = iter.next() { /**/ }
    // equivalent:
    for num in numbers { /**/ }
}
```

The Rust Ecosystem

- Libraries & Documentation
- Idiomatic APIs
- News
- Developer Tools
- Testing
- Continuous Integration & Delivery

Libraries & Documentation

- How to find and use libraries
- How to read their documentation

Finding Libraries

1. blessed.rs
2. lib.rs
3. asking the community
4. crates.io

Not in `std`

- random
- regex
- logging
- time
- http

Using Libraries

demo

Finding Documentation

docs.rs/rand

Idiomatic APIs

Why talk about APIs?

A Rust-API can look *very* different than a C-API.

Case Study #1 – `itertools`

docs.rs/itertools

```
layout: center
class: text-center
# Case Study #2 -- `serde`

<div style="height: 32px"></div>

# [docs.rs/serde](https://docs.rs/serde/)

<Nr />
```

```
layout: cover
class: text-center
# [This Week in Rust](https://this-week-in-rust.org/)

<div></div>

The community-driven newsletter for Rustaceans

<Nr />
```

Development Tools

Included with `rustup`

- toolchain version manager
- build tool
- package manager
- formatter
- linter
- documentation generator
- LSP

Beyond `rustup`

...just read blessed.rs ¯_(□)_/¯

Case Study #1 – `cargo-deny`

docs.rs/cargo-deny

Case Study #2 – `divan`

docs.rs/divan

Testing

A function like this can be anywhere.

```
#[test]
fn program_is_correct() {
    assert_eq!(2 + 2, 4, "math has stopped working");
}
```

It will be executed by `cargo test`.

Do your tests have some shared util code?

```
#[cfg(test)]
mod tests {
    fn setup() {}
    fn teardown() {}

    #[test]
    fn program_is_correct() {
        setup();
        // ...
        teardown();
    }
}
```

It is good practice to have a `tests` module.

The module is only compiled during testing, due to the `#[cfg(test)]` attribute.

You can return `Result`s from your tests.

```
#[test]
fn it_works() -> Result<(), String> {
    if 2 + 2 == 4 {
        Ok(())
    } else {
        Err(String::from("two plus two does not equal four"))
    }
}
```

As expected, `Ok` means the test passed, `Err` means it failed.

Test Organization

unit tests	have access to your internals according to normal visibility rules
integration tests	only have access to the public API of your library

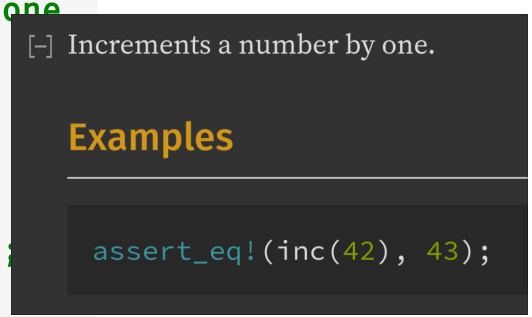
```
├─ src
│  └─ *.rs      <-- unit tests
├─ tests
│  └─ *.rs      <-- integration tests
└─ Cargo.toml
```

Integration tests only work for libraries (`lib.rs`) !

➔ It is common even for binaries to be split into `main.rs` and `lib.rs` ,
with `main.rs` being small and simple.

Documentation Tests

```
1  /// Increments a number by one
2  ///
3  /// # Examples
4  ///
5  /// ```
6  /// assert_eq!(inc(42), 43);
7  /// ```
8  pub fn inc(x: i32) -> i32 {
9      x + 1
10 }
```



`cargo test` will run this example as a test!

Continuous Integration & Delivery

Talk is cheap, let’s get our hands dirty!

Practice 

`rust-exerices/day_4/README.md`

Please suggest improvements for next week!

Check the readme of your repository for the form link.