SmartPointer and PinCoerceUnsized

Kangrejos 2024

Xiangfei Ding <dingxiangfei2009@protonmail.ch> @dingxiangfei2009
08.09.2024

Outline

- Original Problems
- Solution
  - SmartPointer macro
  - Pin safety and PinCoerceUnsized
- Current status

Original Problems

Alice Ryhl (@Darksonn) et al need an Arc<T> that

- Works like a pointer but conform to LKMM

- Super power: dynamic dispatch, and *un-sizing* as implied

The *Super Powers*

- `Ptr<[T; N]> => Ptr<[T]>`

- `Ptr<T> where T: Trait => Ptr<dyn Trait>`
  - By extension, calling `Trait` associate methods on `Ptr<dyn Trait>` should be dispatched to the concret associated methods of `T`

- `Pin<Ptr<T>> where T: Trait => Pin<Ptr<dyn Trait>>`
  - Very desirable

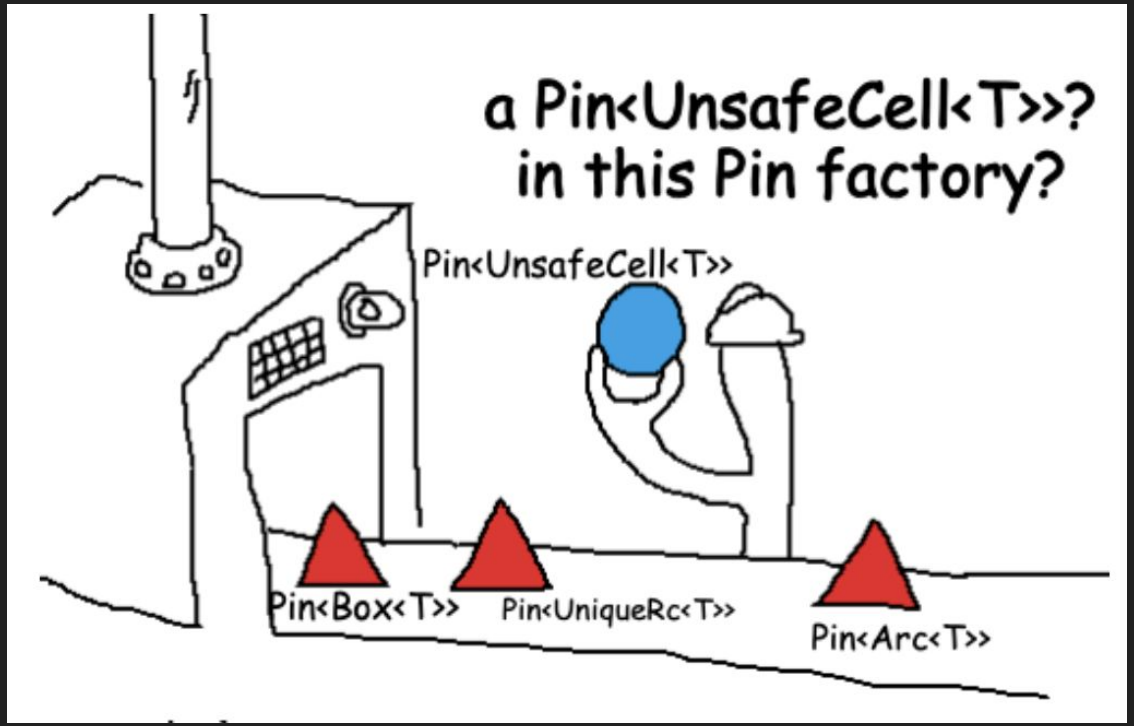
RFC 3621: `#[derive(SmartPointer)]`, for now

- **`CoerceUnsize`** and **`DispatchFromDyn`**
  - They are not stabilized yet
  - but they are used in serious applications already

- Nicely packaged behind the builtin macro in the *core* crate

- We don’t have to wait for the two trait to stabilize

Pin safety


a Pin<UnsafeCell<T>>?
in this Pin factory?
Pin<UnsafeCell<T>>
Pin<Box<T>>
Pin<UniqueRc<T>>
Pin<Arc<T>>

`PinCoerceUnsized`

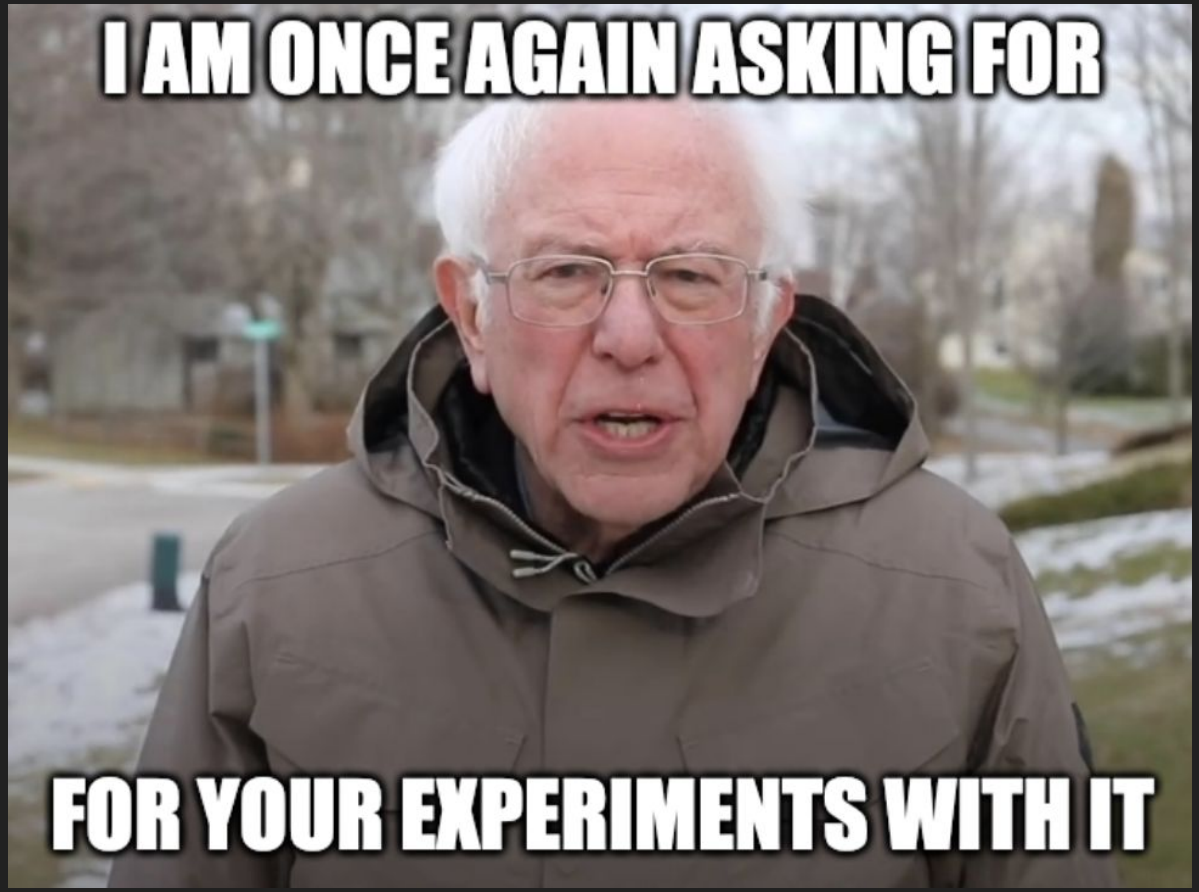
- In case a type implements **`Deref/DerefMut`** and blessed with **`#[derive(SmartPointer)]`**, for **`Pin`** to be safe for use with it ..

- It needs to behave well and not to break the **`Pin`** contract ..

Current Status

- It is on nightly!

- We are going to stabilize it and before that:


I AM ONCE AGAIN ASKING FOR
FOR YOUR EXPERIMENTS WITH IT

Glove test

SmartPointer and PinCoerceUnsized

Kangrejos 2024

Xiangfei Ding <dingxiangfei2009@protonmail.ch> @dingxiangfei2009
08.09.2024