Smart Pointers

Learn to Code with Rust

Smart Pointers I

- A **smart pointer** is a type that *behaves like* a pointer.
- A **smart pointer** can store additional information and perform more actions compared to a plain pointer/reference.
- Most **smart pointers** are built with structs. Structs grant the capacity to store more data.

Smart Pointers II

- A pointer/reference in Rust is like an address to a house.
- A smart pointer is like an address to a house with additional information -- such as property tax records or nearby restaurants.

Smart Pointers III

- References like **&T** borrow data.
- References are not responsible for deallocating the data; the original owner is responsible for deallocation.
- Smart pointers often own and manage their own data, typically on the heap.
- The advantage is that smart pointers *behave like* pointers but can be treated like owned types.

A String is a Smart Pointer!

- A **String** stores a pointer to the heap memory where the text data is located.
- A **String** also stores extra metadata like the length and capacity of the text.
- The **String** smart pointer handles the complexity of the pointer/reference behind the scenes.
- We treat the **String** like a regular owned type.