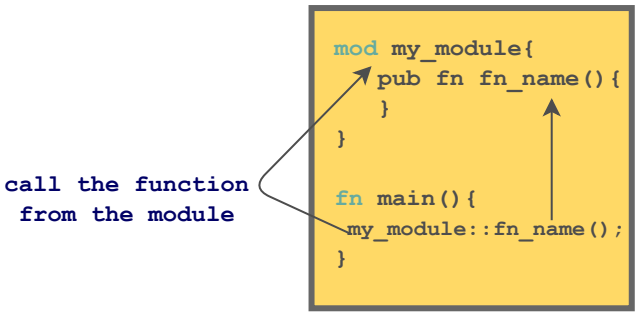
# Controlling Visibility Within the Same File Using 'pub'

This lesson will teach you how to access the module in the same file and within the same directory.

**We'll cover the following**



The `pub` keyword makes the item public and visible outside its scope.




## Privacy Rules

The following are two privacy rules when declaring modules:

### Rule No: 1

If an item is public it can be accessed from anywhere, i.e., within `main` function or any other module.

#### Example : Invoke a Public Function Directly

The following example declares a function public `print_statement()` within the

`mod` `r`:

```
// declare a module
mod r {
  pub fn print_statement(){
    println!("Hi, this a function of module r");
  }
}
// main function
fn main() {
  println!("Let's go inside the module");
  // invoke a module 'r'
   r::print_statement();
}
```

## Rule No: 2

- If an item is private it can be accessed using its parent module meaning it can be accessed within the module but not outside it.

### Example: Invoke a Private Function Indirectly through a Public Function

The example declares a **module** `mod` `r` which has two functions:

- A public function `my_public_function()`
- A private function `my_private_function()`.

> `self` can refer to a function or any item within the same module.

```
// declare a module
mod r{
  fn my_private_function(){
    println!("Hi, I'm a private function within the module");
  }
  pub fn my_public_function(){
    //! also works without writing self i.e.
    //! my_private_function();
    println!("Hi,I'm a public function within the module");
    println!("I'll invoke private function within the module");
    self::my_private_function();

  }
}
// main function
fn main() {
  println!("Let's go inside the module");
  // invoke a module 'r'
   r::my_public_function();
}
```

- If an item is private, it can be called from within the child module.

### Example: Access a Private Function through a Child Module

> If there is a module within the module, then the outer module is called the *parent module* and the module inside the parent module is called the *child module*. This is known as a nested module. This will be covered in more detail in the upcoming lesson.

The example declares a **module** `mod` `outer_module` which has:

- A private function `my_private_function()`.
- `inner_module`
  - Inner module has one public function `my_public_function()`

The following example shows how the private function is accessed in the child module using the keyword `super` followed by `::` and the function name in the parent module.

> `super` keyword refers to the parent module.

```
// main function
fn main() {
  println!("Let's go inside the module");
  outer_module::inner_module::my_public_function();
}
// declare a module
mod outer_module {
  // function within outer module
  fn my_private_function() {
    println!("Hi, I got into the private function of outer module");
  }
  // declare a nested module
  pub mod inner_module {
    // function within nested module
    pub fn my_public_function() {
      println!("Hi, I got into the public function of inner module");
      println!("I'll invoke private function of outer module");
      super::my_private_function();
```

```
    }
  }
}
```

Even though the function `my_private_function()` is declared private, the `main()` function is able to invoke it indirectly because the function it calls is public.

### Example: Access a Root Function

The example below shows how the root function (a function that exists outside the module) can be accessed within the function of a module. Write `super::` followed by the root function name.

> `super` can allow accessing a root function from within the module.

```
// main function
fn main() {
  println!("Let's go inside the module");
  my_module ::my_public_function();
}
fn my_function(){
  println!("Hi, you came inside the root function using super");
  }

// declare a module
mod my_module {
  // function within outer module
  pub fn my_public_function() {
    println!("Invoke root function");
    super::my_function();
  }
}
```

## Quiz

Test your understanding of the `pub` keyword.

1. How can you make a call to the function inside module r?

```
mod r {
```

```
  pub fn print_statement() {
    println!("Hi, this a function of module r");

  }
}
```

2

You can invoke a private function directly.

3

You can invoke a private function through a public function.

4

How can you access a function within the module containing it?

5

How can you access a parent module's private function from a child module's function?

6

Suppose a function is defined outside of a module. How would you invoke it through a module?

Retake Quiz

Now that you have learned how to control visibility within the same file in a directory, let’s learn how it to do it with different files.