The Basic Program

This lesson gets you acquainted with the Hello World program in Rust.

We'll cover the following

Rust code is always put in a file with `.rs` extension.

Hello World Program

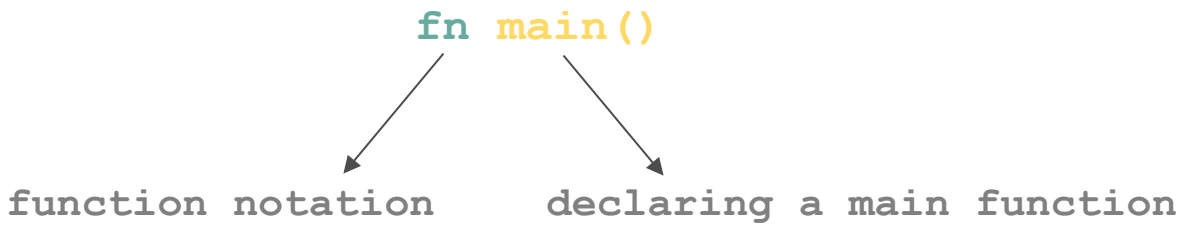
Below is the source code for a traditional Hello World program.

```
fn main() {
    println!("Hello World!");
}
```

Anatomy of a Hello World Program

Let's look at the anatomy of a Hello World program. We'll start from the very first line and go step by step.

Line 1: The main function is the beginning of every Rust program.

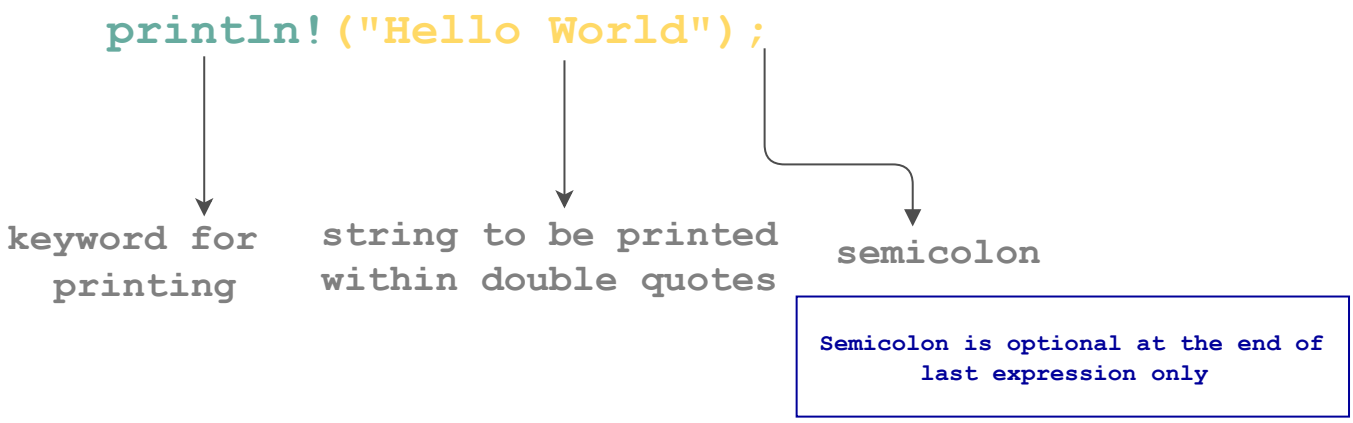


This line declares a function named **main** that takes no arguments and returns nothing. If arguments were present they would have been passed within the round

nothing. If arguments were present they would have been passed within the round brackets `( )`.

The function body starts with the opening curly brace `{`.

Line 2: The second line prints the Hello World to the screen.



Here `println!()` macro takes the string **"Hello World"** and displays it on the screen. This line ends with a semicolon(`;`) which indicates that the expression is over and the next one is ready to begin. The function body ends with a closing curly brace `}`.

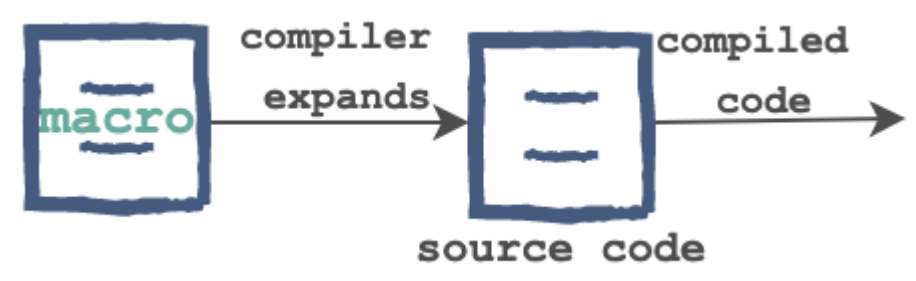
> 📝 Ending an expression with a semicolon is just a convention. If a semicolon is added at the end of an expression, it becomes a statement.

> 📝 Anything declared within pair of braces `{}` denotes a **block of code**.

> **What is a macro**?
>
> A **macro** is an expression that has
> an exclamation mark (`!`) before the parenthesis `()` , i.e.,
>
> ```
> macro_name ! ( );
> ```
>
> **What are macros used for?**
>
> They are used in *metaprogramming*, i.e., code that writes code. They look like functions in other system programming languages like C and C++, but instead of generating a function call like functions, they are expanded into source code that gets compiled with the rest of the program. In this way, they provide

code that gets compiled with the rest of the program. In this way, they provide more run-time features.

Metaprogramming —→



Types of Macros

Rust provides us with some built-in macros, like the `println!()` above, and users can define their own macros as well.

For now, the information above will suffice, but more details on macros will be covered in the advanced course on Rust!

Quiz

Test your understanding of the basics of a program!

Quick Quiz on Basics !

1 What is the key word for declaring a function?

2 What is the output of the following code?

```
fn main() {
    println!("Hello World!")
    println!("Hello");
}
```

Retake Quiz

Now that you have learned the basic syntax of a Rust program, let's learn to format the output in the next lesson.

Or, you can go back to the Learn Rust from Scratch course homepage.