Data Types in Rust

Spoken Tutorial Project
https://spoken-tutorial.org
National Mission on Education through ICT
https://sakshat.ac.in

Jayesh Katta Ramalingaiah
Full Stack Developer

23 November 2020

Learning Objectives

In this tutorial, we will learn about:

Learning Objectives

In this tutorial, we will learn about:

- Supported Data Types

Learning Objectives

In this tutorial, we will learn about:

- **Supported Data Types**
- **Scalar Data Types**

Learning Objectives

In this tutorial, we will learn about:

- **Supported Data Types**
- **Scalar Data Types**
- **Compound Data Types**

Learning Objectives

In this tutorial, we will learn about:

- **Supported Data Types**
- **Scalar Data Types**
- **Compound Data Types**
- **It's types in** Rust

System Specifications

This tutorial is recorded using:

System Specifications

This tutorial is recorded using:

- Ubuntu Linux OS version 18.04

System Specifications

This tutorial is recorded using:

- **Ubuntu Linux OS version 18.04**
- **Rust version 1.47.0**

System Specifications

This tutorial is recorded using:

- Ubuntu Linux OS version 18.04
- Rust version 1.47.0
- Visual Studio Code version 1.45.0 (code editor)

Prerequisites

Prerequisites

- **You should be familiar with** `compiling` **and** `running Rust` **files**

Prerequisites

- **You should be familiar with** `compiling` **and** `running Rust` **files**
- **If not, please go through the prerequisite** `Rust` **tutorials on https://spoken-tutorial.org**

Code Files

- The file used in this tutorial is available in the `Code files` link on this tutorial page

Code Files

- **The file used in this tutorial is available in the `Code files` link on this tutorial page**
- **Pls download and extract the file**

Code Files

- **The file used in this tutorial is available in the `Code files` link on this tutorial page**
- **Pls download and extract the file**
- **Make a copy and then use it for practising**

Data Types in Rust

- **Every value in `Rust` is of a certain `Type`**

Data Types in Rust

- **Every value in** `Rust` **is of a certain** `Type`
- **It tells the** `compiler` **what kind of** `data` **is being specified**

Data Types in Rust

- **This helps the** `compiler` **know how to work with that** `data`

Data Types in Rust

- **This helps the** `compiler` **know how to work with that** `data`
- **Helps optimize for efficient & faster runtime execution**

Data Types in Rust

- **This helps the** `compiler` **know how to work with that** `data`
- **Helps optimize for efficient & faster runtime execution**
- **There are a number of in-built** `Data Types` **in** `Rust`

Data Types in Rust

Data Types in Rust

- **Scalar**

Data Types in Rust

- **Scalar**
- **Compound**

Scalar Data Types

- **A** `Scalar Data Type` **represents a single value**

Scalar Data Types

Rust **has four primary** scalar data types

Scalar Data Types

`Rust` **has four primary** `scalar data types`

- **Integers**

Scalar Data Types

`Rust` **has four primary** `scalar data types`

- **Integers**
- **Float**

Scalar Data Types

`Rust` **has four primary** `scalar data types`

- **Integers**
- **Float**
- **Boolean**

Scalar Data Types

`Rust` **has four primary** `scalar data types`

- **Integers**
- **Float**
- **Boolean**
- **Character**

Float

- Rust has 2 primitive types for float similar to integers
  - f32
  - f64

Float

- **Here the numbers are with decimal points**

Float

- **Here the numbers are with decimal points**
- **For `float`, by default `Rust` allocates `64-bits`**

Float

- **Here the numbers are with decimal points**
- **For** `float`**, by default** `Rust` **allocates** `64-bits`
- `let a = 1.0`

  `rust` **allocates** `64-bits` **of memory**

Float

- **If you want** `Rust` **to allocate** `32-bits` **of memory, we need to explicitly** `typecast` **to** `32-bits` **using** `f32`

```
let a:f32 = 1.0;
```

Boolean

- In `Rust` also we have two possible values for `boolean`

Boolean

- **In `Rust` also we have two possible values for `boolean`**
  - **True**

Boolean

- In `Rust` also we have two possible values for `boolean`
  - True
  - False

Boolean

```
let a:bool = true;
```

- **To** typecast **a** boolean variable **we use** bool

Boolean

```
let a:bool = true;
```

- **To** `typecast` **a** `boolean variable` **we use** `bool`
- **Typically these** `variables` **would be used whenever we set** `flags`

Boolean

```
let a:bool = true;
```

- **Depending upon the state of** `flag` **we allow further** `execution` **of the** `program`

Character

Character

- **The** `Character` **type is the most primitive alphabetic** `type`

Character

- **The** `Character` **type is the most primitive alphabetic** `type`
- **The value should be declared between single quotes**

Character

- **The** `Character` **type is the most primitive alphabetic** `type`
- **The value should be declared between single quotes**
- `let a:char = ‘z’;`

Compound Data Types

- `Compound Types` **group multiple values of other** `types` **into one** `type`

Compound Data Types

- `Rust` **has two primitive** `Compound types` **namely**

Compound Data Types

- `Rust` **has two primitive** `Compound types` **namely**
  - `Tuples`

Compound Data Types

- `Rust` **has two primitive** `Compound types` **namely**
  - `Tuples`
  - `Arrays`

Tuples

- **A `Tuple` is a general way of grouping together some number of other values**

Tuples

- **A `Tuple` is a general way of grouping together some number of other values**
- **A `Tuple` can contain a variety of `types`**

Tuples

- **We create `Tuples` by writing a comma separated list of values within parentheses**

Tuples

- `Destructing` **is a process where a** `Tuple` **is broken into a number of parts using** `pattern matching`

Tuples

- `Destructing` **is a process where a** `Tuple` **is broken into a number of parts using** `pattern matching`
- **Also we can directly access a** `Tuple element` **using a** `period .`

Arrays

Arrays

- **Unlike a** `tuple`**, every element in an** `array` **should have the same** `data type`

Arrays

- **Unlike a** `tuple`**, every element in an** `array` **should have the same** `data type`
- `Arrays` **in** `Rust` **are different as compared to other** `programming languages`

Arrays

- **Unlike a** `tuple`**, every element in an** `array` **should have the same** `data type`
- `Arrays` **in** `Rust` **are different as compared to other** `programming languages`
- **As they have a fixed length**

Arrays

- Once declared they cannot grow or shrink in size

Arrays

- **Once declared they cannot grow or shrink in size**
- **In `Arrays`, the `data` is allocated in a stack rather than a heap and has a fixed number of `elements`**

Summary

In this tutorial, we have learnt:

- **Supported Data Types**
- **Scalar Data Types**
- **Compound Data Types**
- **Its types in** `Rust`

Assignment

- **Go to the project folder** `rust-assignment`
- **In the** `main.rs` **file**
  - **Initialize a** `variable` **named** `m` **and assign an** `array` **to it**
  - **The** `array` **should contain** `5 numbers` **as** `elements`
  - **Print the** `elements` **of the** `array`

Assignment

- `Compile` **and** `execute` **the project**
- **Observe the output in the** `Terminal`

About Spoken Tutorial project

- Watch the video available at https://spoken-tutorial.org /What_is_a_Spoken_Tutorial
- It summarises the Spoken Tutorial project
- If you do not have good bandwidth, you can download and watch it

Spoken Tutorial Workshops

The Spoken Tutorial Project Team

- **Conducts workshops using spoken tutorials**
- **Gives certificates to those who pass an online test**
- **For more details, please write to contact@spoken-tutorial.org**

Answers for THIS Spoken Tutorial

- **Questions in THIS Spoken Tutorial?**
- **Visit https://forums.spoken-tutorial.org**
- **Choose the minute and second where you have the question**
- **Explain your question briefly**
- **The Spoken Tutorial project will ensure an answer**
- **You will have to register to ask questions**

Acknowledgements

- **Spoken Tutorial project is funded by Ministry of Education (MoE), Govt. of India**

Thank you

Jayesh Katta Ramalingaiah
Full Stack Developer
https://www.linkedin.com/in/jayeshkattar