VNCKey

RUST PERÚ

Diviértete con Rust

Viaja al futuro con el poder de Rust

- Programación de Sistemas
- Zero-Cost Abstractions
- Memory Safety

Alexander Villanueva

Contenido

Primeros pasos

¿Qué es Rust?

Rust es un lenguaje de sistemas moderno, diseñado para garantizar seguridad de memoria sin recolector de basura y rendimiento comparable a C y C++.

Su desarrollo comenzó en 2006 por Graydon Hoare y fue patrocinado por Mozilla, culminando con su primera versión estable en mayo de 2015.

Rust resuelve problemas clásicos de lenguajes de bajo nivel como punteros nulos, doble liberación y condiciones de carrera **data races**, mediante un sistema de propiedad **Ownership** y un verificador de préstamos **Borrowing** que el compilador valida en tiempo de compilación.

Rust no solo permite crear aplicaciones completas desde cero, puesto que su diseño de abstractions y memory safety lo convierte en una opción atractiva para reescribir componentes críticos en ecosistemas ya existentes.

Ejemplos concretos:

1. Firefox sustituyó partes de su motor CSS con Servo, reduciendo errores de memoria en un 70 %.
2. Discord migró su servicio de voz a Rust y dividió por 10 el uso de CPU frente a la versión en Go.
3. Dropbox reescribió su sistema de sincronización con Rust + Tokio, logrando menos latencia y menor consumo de memoria sin aumentar el coste de hardware.

Estas migraciones demuestran que Rust puede aumentar el rendimiento y reducir costes operativos sin sacrificar seguridad ni productividad.

La seguridad garantizada por su diseño ha generado una confianza excepcional entre los desarrolladores. De manera ininterrumpida desde 2016, Rust ha sido votado como el “lenguaje de programación más querido” en la encuesta anual de Stack Overflow.

Esta popularidad y la alta satisfacción de los desarrolladores son un fuerte indicador de su productividad y baja curva de frustración una vez dominado.

El compilador de Rust puede parecer exigente al principio, pero está diseñado para detectar errores *antes* de que tu código se ejecute. Esto significa menos bugs en producción y noches más tranquilas.

Además, Rust:

- **Compila directamente a código máquina**: Sin overhead de runtime ni máquinas virtuales
- **Soporta múltiples hilos sin data races**: El compilador garantiza que tu código concurrente es seguro
- **Elimina categorías enteras de bugs**: Muchos errores de concurrencia simplemente no pueden ocurrir

Por qué las empresas eligen Rust

Las razones principales que impulsan esta adopción son:

1. **Menos bugs en producción**: El compilador atrapa errores antes del despliegue
2. **Mayor rendimiento**: Comparable a C/C++ sin sacrificar seguridad
3. **Menor uso de recursos**: Traduce en ahorro de costos de infraestructura
4. **Código más mantenible**: Las garantías del lenguaje facilitan refactorings grandes

Instalación de Rust

Rust se instala mediante rustup, el instalador oficial que gestiona versiones del compilador, toolchains y herramientas asociadas. Este proceso es prácticamente idéntico en todos los sistemas operativos.

Instalación en Linux y macOS

1. Abre una terminal.
2. Ejecuta el siguiente comando para descargar e iniciar el instalador de Rust:

   ```
   curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh
   ```

   Este comando descarga el instalador de rustup, que configurará el compilador Rust, Cargo y las herramientas necesarias.
3. Durante el proceso, se te preguntará si deseas continuar con la instalación por defecto o personalizarla.

   La opción recomendada es presionar Enter para aceptar la configuración estándar. Verás algo como esto:

   ```
   --snip--

   Rust is installed now. Great!

   To get started you may need to restart your current shell.
   This would reload your PATH environment variable to include
   Cargo's bin directory ($HOME/.cargo/bin).

   # For fish
   ```
4. Reinicia tu terminal o cierra sesión y vuelve a iniciar para que los cambios en el PATH surtan efecto (rustup lo configura automáticamente, pero a veces necesita esto).

Instalación de Rust en Windows

1. Instalar Visual Studio Build Tools

   Descarga Visual Studio desde:

   https://visualstudio.microsoft.com/es/downloads/

   Rust en Windows necesita el compilador MSVC para poder generar archivos .exe.

   Para eso es necesario instalar el componente **“Desktop Development with C++”**, que incluye todas las herramientas esenciales para compilar programas nativos.
2. Instalación minimalista

   Si no planeas desarrollar en C++ y quieres instalar solo lo indispensable, puedes ir a la pestaña “Componentes individuales” y seleccionar únicamente:

   - MSVC v143 – VS 2022 C++ x64/x86 build tools
   - Windows 11 SDK (10.0.22621.0)

   La versión exacta del SDK no afecta al código puro de Rust, pero si también trabajas con C++, puede que necesites una versión específica según tu proyecto.

   Después de seleccionar los componentes, haz clic en Instalar y espera a que finalice.

3. Descargar Rust desde la página oficial

   Accede a: https://www.rust-lang.org/tools/install

   Aquí encontrarás tres instaladores:

   - rustup-init.exe (32 bits)
   - rustup-init.exe (x64 / 64 bits)
   - rustup-init.exe (ARM64)

   Selecciona el que corresponda a la arquitectura de tu sistema.

4. Ejecutar el instalador

   Ejecuta el archivo rustup-init.exe y sigue las instrucciones de la consola.

   Cuando aparezca el menú, simplemente presiona Enter para la instalación estándar:

   ```
   --snip--
   1) Proceed with standard installation (default - just press enter)
   2) Customize installation
   3) Cancel installation
   >1
   ```

   En la mayoría de casos es necesario cerrar y volver a abrir la terminal para que Windows reconozca los nuevos comandos `rustc` `cargo` `rustup` y sus todas sus herramientas.

Verificación de la Instalación

Ejecuta los siguientes comandos en tu terminal “Windows, Linux o macOS” para confirmar que Rust se instaló correctamente:

```
rustc --version
cargo --version
```

Si la instalación fue exitosa, verás algo similar a lo siguiente:

```
cargo --version
cargo 1.89.0 (c24e10642 2025-06-23)

rustc --version
rustc 1.89.0 (29483883e 2025-08-04)

rustdoc --version
rustdoc 1.89.0 (29483883e 2025-08-04)
```

En la siguiente sección exploraremos las herramientas principales del ecosistema Rust: cargo, rustc y rustdoc.

Estas utilidades forman el núcleo de tu flujo de trabajo y te acompañarán en cada proyecto.

Cargo es el administrador de compilación y dependencias, además de la herramienta de propósito general de Rust. Se utiliza para crear nuevos proyectos, compilar, ejecutar y administrar bibliotecas externas “dependencias”.

```
cargo new mi_proyecto
```

Rustc es el compilador de Rust convierte tu código fuente en Rust a código máquina ejecutable. Aunque normalmente lo usamos a través de Cargo, rustc resulta útil para compilar archivos individuales o hacer pruebas rápidas sin crear un proyecto completo.

```
rustc mi_archivo.rs
```

Rustdoc es la herramienta de documentación de Rust. Si escribes comentarios de documentación con `///`, rustdoc puede generar automáticamente páginas HTML bien formateadas.

```
cargo doc --open
```

Rustup es el gestor del entorno Rust que permite actualizar, cambiar entre diferentes versiones del compilador "stable, beta o nightly" y administrar toolchains.

Rust se actualiza con facilidad gracias a **rustup**.

bash

```
rustup update
```