Article
Dmitry Maslennikov · Mar 3, 2021 4m read

Open Exchange

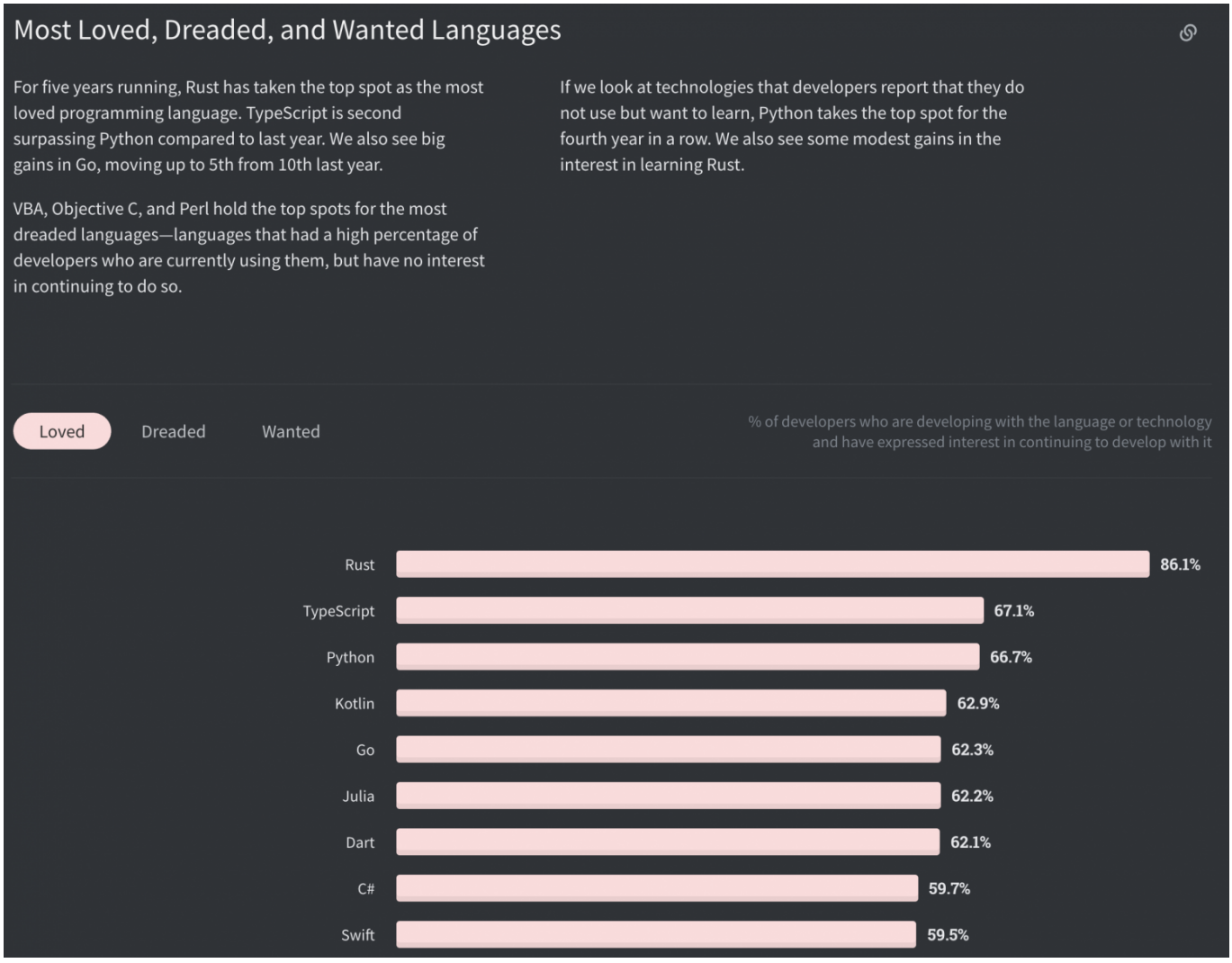
Access to IRIS from Rust

What do you think If I will say you, that very soon you will be able to connect to IRIS from the application written in Rust.

What is Rust

Rust is a multi-paradigm programming language designed for performance and safety, especially safe concurrency. Rust is syntactically similar to C++, but can guarantee memory safety by using a borrow checker to validate references. Rust achieves memory safety without garbage collection, and reference counting is optional. (c) Wikipedia

Most loved language for the last five years by the time of StackOverflow survey 2020.



What is possible right now.

It can already work with globals and do simple SQL queries. Look at the working example.

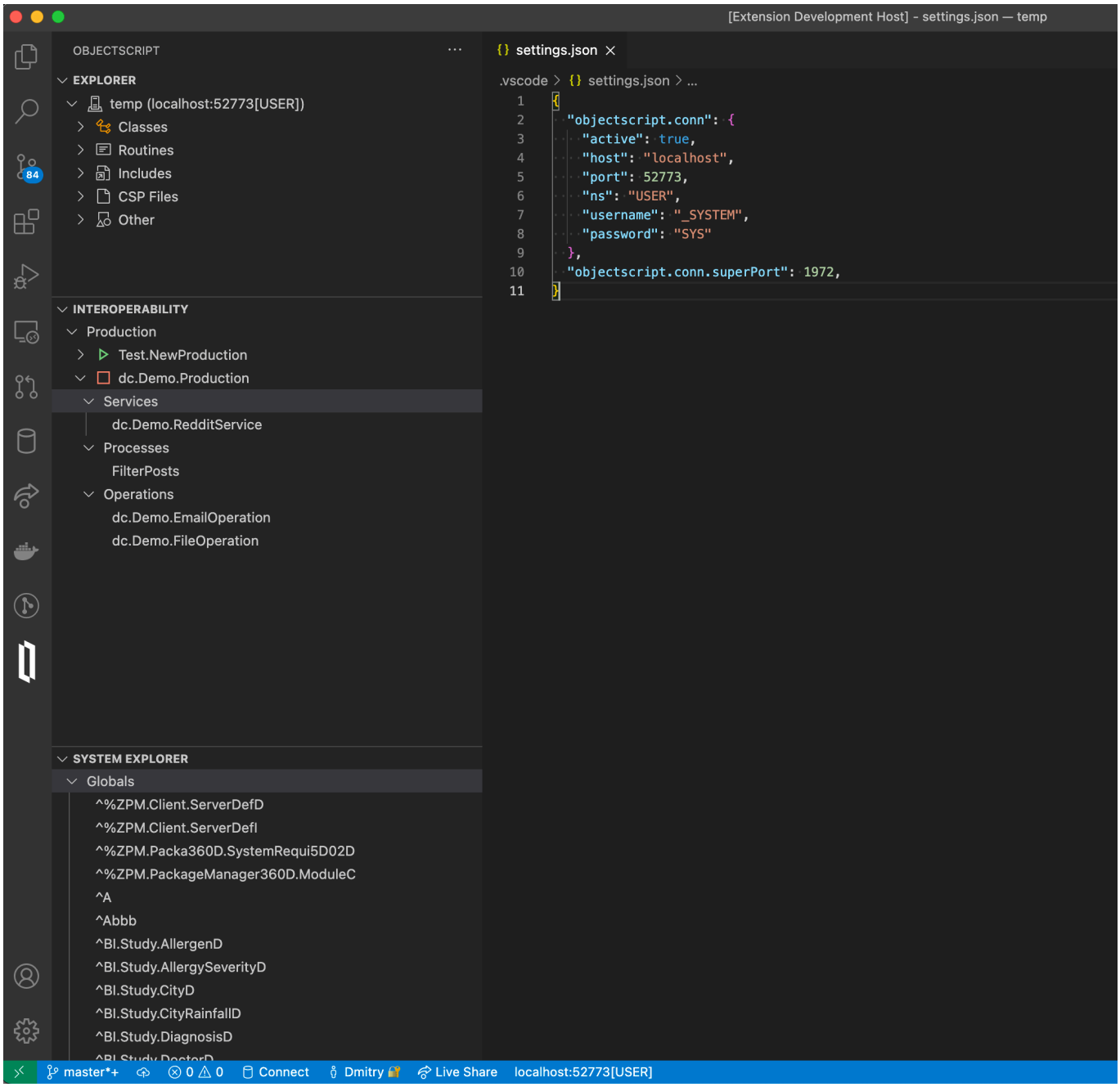
```
use irisnative;
use irisnative::{connection::*, global, global::Sub, Global};

fn main() {
  let host = "127.0.0.1";
  let port = 1972;
  let namespace = "USER";
  let username = "_SYSTEM";
  let password = "SYS";
  match irisnative::connect(host, port, namespace, username, password) {
    Ok(mut connection) => {
      println!("Connection established");

      println!("Server: {}", connection.server_version());

      connection.kill(&global!(A));
      connection.set(&global!(A(1)), "1");
      connection.set(&global!(A(1, 2)), "test");
      connection.set(&global!(A(1, "2", 3)), "123");
      connection.set(&global!(A(2, 1)), "21test");
      connection.set(&global!(A(3, 1)), "test31");

      let mut global = global!(A(""));
      while let Some(key) = connection.next(&mut global) {
        println!("^A({:?}) = {:?}", key, {
          if connection.is_defined(&global).0 {
            let value: String = connection.get(&global).unwrap();
            value
          } else {
            String::from("<UNDEFINED>")
          }
        });
        let mut global1 = global!(A(key, ""));
        while let Some(key1) = connection.next(&mut global1) {
          let value: String;
          if connection.is_defined(&global1).0 {
            value = connection.get(&global1).unwrap();
          } else {
            value = String::from("<UNDEFINED>");
          }
          println!("^A({:?}, {:?}) = {:?}", key, key1, value);
        }
      }

      let mut rs = connection.query(String::from(
        "SELECT Name from %Dictionary.ClassDefinition WHERE Super = 'Ens.Production'
and Abstract<>1"));
      while rs.next() {
        let name: String = rs.get(0).unwrap();
        println!("{}", name);
      }
    }
    Err(err) => {
      println!("Error: {}", err.message);
    }
  }
```

```
}
```

So, it will be possible to connect to IRIS over the network, as soon as it has access to the super server port (1972).

Real case usage

To use it in production, it has to be compiled into an executable file or library. And I have a project where I use it now. It is the VSCode extension for advanced control of InterSystems IRIS. The project is participating in InterSystems Grand Prix Contest, please vote there. Rust helps to get direct access to IRIS, and it will be possible to check the status or stop/start Production, observe globals, and many other things in the future.



Rust has to be compiled to a binary format for the desired platform, and at the moment this extension is built for macOS x64 and Windows x64. But Rust can be compiled for a very wide range of platforms, including Arm64.

Let's see what else can be done

Let's try to run the Rust application (from the example above) with IRIS connector in the Docker. Simple Dockerfile

to build an image with the application.

```
FROM ekidd/rust-musl-builder

ADD --chown=rust:rust . ./

RUN cargo build --release --example main

FROM scratch

COPY --from=0 /home/rust/src/target/x86_64-unknown-linux-musl/release/examples/main /

CMD [ "/main" ]
```

Let's build it

```
$ docker build -t rust-irisnative .
```

And run it

```
$ docker run -it rust-irisnative
Connection established
Server: IRIS for UNIX (Ubuntu Server LTS for x86-64 Containers) 2020.4 (Build 524U) T
hu Oct 22 2020 13:04:25 EDT
^A("1") = "1"
^A("1", "2") = "test"
^A("2") = "<UNDEFINED>"
^A("2", "1") = "21test"
^A("3") = "<UNDEFINED>"
^A("3", "1") = "test31"
Test.NewProduction
dc.Demo.Production
```

Wonder, how big is that image?

```
$ docker images rust-irisnative
REPOSITORY        TAG       IMAGE ID       CREATED         SIZE
rust-irisnative   latest    0b1e54e7aa6f   2 minutes ago   3.92MB
```

Just a few megabytes of the image can connect to IRIS, running somewhere else. Looks like it is very good for microservices and serverless applications. And as a bonus for IoT, Rust applications can run on tiny pc, for instance, RaspberyPi Pico.

What do you think about it and how would you use Rust?

#CaretDev #Deployment #Languages #VSCode #InterSystems IRIS #InterSystems IRIS for Health
Check the related application on InterSystems Open Exchange