

how use postgres::config::Config in postgres::client::Client

To use `postgres::config::Config` with `postgres::client::Client` in Rust, you need to create a `Config` instance and then use it to establish a connection with the PostgreSQL database. Here's a step-by-step guide:

Step 1: Add Dependencies

First, ensure you have the `postgres` crate in your `Cargo.toml`:

```
[dependencies]
postgres = "0.7.5"
```

Step 2: Create a Config Instance

You can create a `Config` instance by parsing a connection string or by setting each parameter individually. Here's how to do it both ways:

Using a Connection String

```
use postgres::config::Config;

fn main() -> Result<(), postgres::Error> {
    let mut config = Config::new();
    config.host(Some("localhost"));
    config.port(Some(5432));
    config.user(Some("myuser"));
    config.password(Some("mypassword"));
    config.dbname(Some("mydatabase"));

    // Alternatively, you can use a connection string
    // let config = Config::from_str("host=localhost port=5432 user=myuser password=mypas

    // Now you can use this config to connect
    let client = postgres::Client::connect(config)?;

    // Perform queries
    let rows = client.query("SELECT 1", &[])?;
    for row in rows {
        println!("{:?}", row);
    }
}
```

```
    Ok(())  
}
```

Using a Connection String Directly

If you prefer to use a connection string directly:

```
use postgres::config::Config;  
  
fn main() -> Result<(), postgres::Error> {  
    let config = Config::from_str("host=localhost port=5432 user=myuser password=mypassword");  
  
    let client = postgres::Client::connect(config)?;  
  
    let rows = client.query("SELECT 1", &[])?;  
    for row in rows {  
        println!("{}", row);  
    }  
  
    Ok(())  
}
```

Explanation

- **Config Creation:** The `Config` instance is created and populated with connection details such as host, port, user, password, and database name.
- **Connection Establishment:** The `Client::connect` method is used with the `Config` instance to establish a connection to the PostgreSQL database.
- **Query Execution:** Once connected, you can execute SQL queries using the `client.query` method.

This approach allows you to configure and connect to a PostgreSQL database using the `postgres` crate in Rust.

