Summary of Setting Up Iroh for Dataset Replication

1. **Install with:**

```
curl -fsSL https://sh.iroh.computer/install.sh | sh
```

1. **Confirm Installation:**
    - Run `iroh help` to verify Iroh is installed correctly.
2. **Start Iroh Node:**
    - Execute `iroh start` to initiate the Iroh node.
3. **Access Iroh Console:**
    - Use `iroh console` to open the Iroh console and interact with the node.
4. **Create an Author:**
    - Create an author identity using `author new --switch` to represent actions in the console.
5. **Create a Document:**
    - Create a document with `doc new --switch` and set it as the default for the session.
    - Set a value in the document using `doc set foo bar`.
6. **Sync Document Between Nodes:**
    - Share write access to the document from Node A using `doc share write`.
    - Create a second node (Node B) and join the document using the shared ticket.
    - Verify that changes in one node are reflected in the other in real time.

PROFI

Detailed Steps in Markdown

Installation

There are two replication methods, we go with the easiest one (we don't need more than that). Install Iroh with:

```
curl -fsSL https://sh.iroh.computer/install.sh | sh
```

Confirm Iroh Installation

Confirm Iroh is installed correctly by running `iroh help`. You should see help text that starts like this:

```
$ iroh help
Iroh is a tool for syncing bytes.
https://iroh.computer/docs

Usage: iroh [OPTIONS] [COMMAND]
...
```

Start the Iroh Console

```
$ iroh start
Iroh is running
Node ID: l5kkl6u6fqcnjxvdkoihq3nbsqczxeulf5qvatb2qh3bwheoyaha
```

Now that we have a node running, let's create a document. We'll do that from the console. The console is a REPL that lets you talk to the node. We open the console with `iroh console` and connect to the node we started using the `iroh start` command.

```
$ iroh console
Welcome to the Iroh console!
Type `help` for a list of commands.
>
```

Create an Author

Before we can work with documents, we need to create an author, which is an identity that will represent us while we work.

PROFI

```
> author new --switch
fhu3uk4wc2hl2e45npepyyic5u2tv37gnmmeove4ka2bgjeqhzh7a
Active author is now fhu3uk4wc2hl2e45
```

We use the `--switch` flag to make this author the default author for the console. This means we don't have to pass the author ID to every command.

Create a Document

Next, let's create a document. Again, we'll use the `--switch` flag to use this document as the default document for the console session:

```
> doc new --switch
x5qffedimrovdpsr3andm3hdj06g2nlaqstdm6oe3utblxsdyh3eq
Active doc is now x5qffedimrovdpsr
```

```
author:fhu3uk4wc2hl2e45 doc:x5qffedimrovdpsr
```

Set a Value in the Document

Let's set a value in the document, and read it back.

```
author:fhu3uk4wc2hl2e45 doc:x5qffedimrovdpsr
> doc set foo bar
6lu1jp3w2idm3bk51qkzr6ssdk6hlkumsu7jpqbqzfqsymgbf6e6q

author:fhu3uk4wc2hl2e45 doc:x5qffedimrovdpsr
> doc get foo
@fhu3uk4wc2hl2e45: foo = "bar" (3 B)
```

If the content can be displayed as utf-8, the console will display the beginning of the text, otherwise the `doc get` command will display the content's hash. You can use the `--mode` flag to force the content to be printed in different ways. For example, using the option `--mode hash`, will always show the hash of the content, and `--mode content` will write the entire contents to the terminal, regardless of format.

Sync the Document

Now let's actually sync this document with a second node. To disambiguate, we'll call the node we've been working with so far "Node A", and the new one we'll create "Node B". We'll start by sharing write access to the document from `Node A`:

Node A

```
author:fhu3uk4wc2hl2e45 doc:x5qffedimrovdpsr
> doc share write
doclzkcgw25utyfj3o4q5dbeo5tyevcjutr25nagmzi5yfsgtqoeohqcig35slag7ibfh3dd
cls4szbl3k4q6m6d7gnrpgnv66tlnk3gktxaabqa3bjffy7riydabwcsklryrlqbqfikyrmi
vybae
```

That long `doc1zkgw...` string is a ticket to join the document. Anyone who has that ticket can join & write to this document. Let's use it to join the document from `Node B`, but first, we need to create Node B. In another terminal, run:

Node B

```
$ export IROH_DATA_DIR=./
$ iroh console --start
Iroh is running
Node ID: 06llqfj4wjtzar777baet27crxvqu4bkms6uciawwer2qo64awheq
```

```
Welcome to the Iroh console!
Type `help` for a list of commands.
>
```

Iroh has a default data directory, and a default rpc port (`1337`), for internal communication between the different Iroh processes. We pass `IROH_DATA_DIR=./` as an environment variable to override this directory because we're already using it for our first Iroh node. Iroh will also detect that another process is using the default rpc port, and will assign a different one for internal communication, using the `IROH_DATA_DIR` as the place to coordinate which rpc port that Iroh process is listening on.

Notice that here we have combined the previous steps to get the console running from two commands (each in a separate terminal), to just one command. The `--start` flag tells the Iroh node that we want to both start an Iroh node and also connect the console to that node.

To run these as separate commands, you would need to set the `IROH_DATA_DIR` flag to point to the same directory for each terminal window:

In One Terminal:

```
$ export IROH_DATA_DIR=./
$ iroh start
```

In Another Terminal:

```
$ export IROH_DATA_DIR=./
$ iroh console
```

PROFI

Join Document from Node B

Now we can join the document from `Node B`, pasting in the value of the ticket we created with `doc share write` on `Node A`:

Node B

```
> author new --switch
ksdx5fs4kwjmoaca2bod3z62vsm7ny42vubfxpczyr34trjvxp6q
Active author is now ksdx5fs4kwjmoaca

> doc join --switch
doclzkcgw25utyfj3o4q5dbeo5tyevcjutr25nagmzi5yfsgtqoeohqcig35slag7ibfh3dd
cls4szbl3k4q6m6d7gnrpgnv66tlnk3gktxaabqa3bjffy7riydabwcsklryrlqbqfikyrmi
vybae
Active doc is now x5qffedimrovdpsr
```

```
author:ksdx5fs4kwjmoaca doc:x5qffedimrovdpsr
> doc get foo -c
@fhu3uk4wc2hl2e45: foo = "bar" (3 B)
```

The two consoles are now synchronized! We can see the value we set from Node A on Node B. Any edit you make in one console will be reflected in the other in real time.