**Mobile Adapter GB - Reon-Team, Serverside and Clientside Setup Process:**

If you see me use too much sudo (yes I know :( ).

*Credits and Thanks obviously to the REON-Devs/Community, Discord and especially @kabi.chan for having the patience to guide me through my painfully broken first time setup.  
  
This was tested on an ubuntu-server 24.04 and a raspberry pi 4b with debian on it.*

**PREREQUISITES**

**Download the *LATEST* (dont use the 5 year old ones 💀) files from Github:**

**1. Server files for the actual server setup:**

- <https://github.com/REONTeam/reon>  
- <https://github.com/REONTeam/mobile-relay>  
- <https://getcomposer.org/download/>  
- <https://nodejs.org/en/download/package-manager>  
- <https://github.com/gb-mobile/pokecrystal-news-en>

**2. Mobile Adapter GB files for the client:**

- <https://github.com/REONTeam/libmobile-bgb> | Mobile Adapter GB for your emulator  
- <https://github.com/REONTeam/dummy-servers> | DNS and POP-Mail Server for the Adapter  
- Python for your machine to be able to use the DNS Server script.

**3. One of the pokecrystal-mobile builds (choose preferred language):**

- <https://github.com/gb-mobile/pokecrystal-mobile-eng>  
- <https://github.com/gb-mobile/pokecrystal-mobile-ger>  
- <https://github.com/gb-mobile/pokecrystal-mobile-fra>  
- <https://github.com/gb-mobile/pokecrystal-mobile-ita>  
- <https://github.com/gb-mobile/pokecrystal-mobile-spa>

**4. Download an emulator with mobile adapter features:**

- <https://bgb.bircd.org/>

**5. Get ahold of a copy of Mobile Trainer GB (for configuring the Mobile Adapter)**

- I bought a copy of the game and dumped the rom for my own use

***INFO:*** if you do buy a Mobile Trainer Cartridge, or any of the other Mobile Adapter compatible games, it would be advised to dump the savefile without starting it up anywhere or plugging it somewhere before. You may be able to recover some old data still on the cartridges (**please correct me if I am wrong about this**).  
  
**6. (*optional*) Dump your original Pokemon Crystal savefile from your cartridge:**

- Please download the (corresponding to your region) tool from the discord to convert your savefile and make it usable with Pokecrystal  
(Pokemon Crystal Channel on REON Discord; dated 15.04.2024)

**SERVERSIDE SETUP:**

**INFO: This documentation (so far) was only tested on a Raspberry Pi Debian (ARM), I will also test this on an Ubuntu-Server (x64) and the following instructions are commands to use on Linux.**  
  
To begin our Serverside Setup, we need to install some of our required software:  
- apache2 (Web-Server)  
- php and composer (Web-Server and Website Stuff)  
- mariadb-server / mariadb   
- node.js and npm (Mail-Server and Trading-Corner),   
 might need to reboot while installing node.js!

**Managing your files:**

Download the files from the reon repository:  
> git clone <https://github.com/REONTeam/reon.git>

Move all files to /var/www/web/   
All files in [ /reon/web/ ] should be also be moved into [ /var/www/web/ ] and then you can go ahead and delete [ /var/www/web/web/ ].

> cd /var/www/  
> sudo mkdir web  
> sudo mv /path/to/reon/\* /var/www/web/  
> cd web/web/  
> sudo mv /var/www/web/web/\* ..  
> cd ..  
> sudo rmdir web  
  
This should have moved everything to where it should be. Additionally, we can also go into /var/www/web/app/ and delete the crystal-trade-corner folder in there since it is not needed.

It should look something like this: (*Don't mind the additional files in there from my testing*)



**Web-Server Setup:**

After this we need to configure our apache2 Server. For this we can use the provided "vhost.example.conf" in [ /var/www/web/ ]  
  
> cd /etc/apache2/sites-enabled/  
> sudo mv /var/www/web/vhost.example.conf /etc/apache2/sites-enabled/000-default.conf  
> sudo nano 000-default.conf  
  
And then edit the "DocumentRoot" and "<Directory... " lines to point to: [ /var/www/web/htdocs/ ] **instead of** [/var/www/cgb/html/ ]

It should then look like this:  


Now we can install php, php-mysqli and composer:

> sudo apt install php  
> sudo apt install php-mysqli  
  
For installing composer, follow the explanation on the composer website to install it.  
  
<https://getcomposer.org/download/>  
  
This installs the PHP part of the Webserver and will make the website available to the users.  
Now we go to the directory containing the composer files, in this case [ /var/www/web/ ]:  
  
> cd /var/www/web/  
> php composer.phar install  
> php composer.phar update

After that do:  
  
> sudo a2enmod rewrite  
> sudo systemctl restart apache2

**Database Setup:**

In our [ /var/www/web/ ] directory, there is a tables.sql, which we will use to create our database for our services.  
First, if you haven't done so yet, install needed software:  
  
> sudo apt install mariadb-server   
**OR IN CASE OF RASPBERRY PI DEBIAN**> sudo apt install mariadb  
  
Both mariadb-server and mariadb should work the same.  
We now need to choose a name for our database and edit the **tables.sql** and also the SQL command accordingly:  
  
> sudo nano /var/www/web/tables.sql  
  
Edit the first two lines to your desired database name and save it.  
Should look like this  
  
  
Now run the mysql command and run tables.sql:  
  
> sudo mariadb  
[**in mariadb**]>   
 CREATE DATABASE **my\_db\_name** CHARACTER SET utf8mb4 COLLATE utf8mb4\_unicode\_ci;

CREATE USER '**my\_db\_user**'@'%' IDENTIFIED BY '**my\_db\_password**';

GRANT ALL ON **my\_db\_user**.\* TO '**my\_db\_user**'@'%';

FLUSH PRIVILEGES;

[Replace all **my\_db\_name** with the database name you chose. In my case it was just reon. Put in a password for **my\_db\_password**, for example '**totallySecretPassword**' and replace **my\_db\_user** with a username you chose.]  
  
[**in mariadb**]> source /var/www/web/tables.sql  
  
[**in mariadb**]> ALTER USER 'root'@'localhost' IDENTIFIED VIA mysql\_native\_password USING PASSWORD('root');

[**in mariadb**]> GRANT ALL PRIVILEGES ON \*.\* TO 'my\_db\_user'@'%' IDENTIFIED BY 'my\_db\_password';  
  
[**in mariadb**]> exit  
  
In the future to get into mariadb, you can use:  
  
> sudo mariadb -p  
[which will ask for a password}  
root [enter]  
  
This should have set up your database correctly.   
And yes the password for root is now root in mariadb. Feel free to change that.  
Also since I have no real experience with the database permissions feel free to correct this, this is just what worked for me.

**Mail-Server Setup:**

We now want to install our Mail-Server so the services do actually work.  
For this, install Node.js and npm to be able to start the Mail-Service.  
See the Node.js website for instructions: <https://nodejs.org/en/download/package-manager>  
And then use following command:  
  
> sudo ln -s "$(which node)" /usr/bin/node  
  
We can now go into [ /var/www/web/ ] and move our "config.example.json" one directory level higher into [ /var/www/ ], rename it to "config.json", change needed settings and create a Symlink so it is also available in [ /var/www/web/ ] for other services.  
  
> sudo mv /var/www/web/config.example.json /var/www/config.json  
> cd /var/www/  
> sudo nano config.json  
  
[where we change the settings to suit our needs]  
  
This is what my config looks like and we need to change the following lines:  
  
hostname > localhost  
email\_domain\_dion > name.dion.ne.jp [choose a 4 letter subdomain for your mail adresses]  
mysql\_host > 127.0.0.1 [ Do not forget to do this, might break otherwise... ]  
mysql\_user > my\_db\_user we created  
mysql\_password > my\_db\_password we created  
mysql\_database > my\_db\_name (from running tables.sql)  
  
To create the needed Symlink, go back up to [ /var/www/web/ ] and use:  
  
> cd /var/www/web/  
> sudo ln -s ../config.json .  
  
Then we go into the mail directory:   
  
> cd /var/www/web/mail/  
  
and install all needed dependencies for our Mail-Server with:  
  
> npm install  
> npm update  
  
then we can do:  
> which npm [this gives you the directory of the npm install to use with sudo]  
  
> sudo /copy/which/npm/result/here/path/to/npm run start (&)  
[add the & to start as background process]

And your Mail-Server should now be up and running.

**Creating User(s):**

We now want to be able to create our users which should then be able to log-in on the website and use their dion user credentials to configure their Mobile Trainer and Adapter Config and use our services.  
To create an encrypted password, use:

> htpasswd -bnBC 10 "" **YOUR\_PASSWORD** | tr -d ':\n'

  
  
Replace **YOUR\_PASSWORD** with the requested password in plaintext and send the command. This should output you an encrypted password which you can copy into your next step.  
We now start back up MYSQL and want to create our new user:

> sudo mariadb use **my\_db\_name**  
[**in mariadb**]> INSERT INTO sys\_users

(email, password, dion\_ppp\_id, dion\_email\_local, log\_in\_password, money\_spent) VALUES (

'test@example.net', -- Email address to log into the website with

'$2y$10$/VhwAWF6p3jZDtSL2y0vaO7RcHa.Jh5hk6Kt.wLV/FDRSnO0hYF/2', -- Password for website, generated with command above

'g000000000', -- Must be formatted as 'g' followed by 9 random digits.

'kaanowo1', -- dion email address. 8 characters

'kaan1122', -- dion password. 8 characters. Must have mix of letters and numbers

0);

Here we will want to replace all values in '.' and put in the credentials for the user:  
- email adress for the website can/should be a real email address  
- password is the encrypted password we just generated.   
- g-ID can have any **9 random digits** (choose any 9 digits)  
- username part of the mobile trainer address (**8 characters**) before the @ of your reon/ or any other 4-letter name you chose while setting up the Mail-Server - email address  
[In my case this would be kaanowo1 for kaanowo1@reon.dion.ne.jp, for you it could be username@name.dion.ne.jp and username is what you need.]  
- your desired **8 character** password of letters **and** numbers

**The character/letter limits are to be complied with, since otherwise we will not be able to configure the Mobile Trainer and Adapter with these credentials!**

**Trading-Corner Setup:**

First time setup:  
npm install  
npm update

To run the trading corner automatically, all we have to do is create a script to run (I did that to make it easier for myself and then directed the output to a logfile) and add an entry in cronjob.   
  
Creating the script:   
  
I created the script so it would change the directory to the correct one and then run the exchange script:

#!/usr/bin/bash  
cd /var/www/web/app/pokemon-exchange/  
/home/kaan/.nvm/versions/node/v23.0.0/bin/npm run start  
  
To find your npm directory just run:  
  
>which npm   
  
and copy it into your script.  
Then we want to add the cronjob:  
  
> crontab -e  
  
Scroll down to an empty new line and type in (please format the line to your needs):  
  
\*/10 \* \* \* \* /var/www/web/app/pokemon-exchange/run.sh >> /var/www/web/pokemon-exchange/exchange.log   
  
This creates a new cronjob to refresh the pokemon-exchange every 10 minutes and writes a log to [ /var/www/web/app/pokemon-exchange/exchange.log ].

**Battle tower updates:**  
  
Follow the same instructions as in the trading-corner setup and replace all [../pokemon-exchange/ ] with [../pokemon-battle/ ] and rename the log to something like [ battle.log ].

**News-Machine Setup:**

We can now get started with setting up the news machine.  
Download the first news issue and put the downloaded file into /tmp/ and unzip and put it into our database:  
  
See <https://github.com/gb-mobile/pokecrystal-news-en> for the data or check the discord.  
  
> cd /tmp/  
[copy the file over, download in discord or build it yourself]  
> unzip reon\_news\_fixed.zip  
> sudo mkdir reon\_news  
> sudo mv reon\_news\_fixed/\* reon\_news/  
> cd reon\_news/  
> nano add\_news.sql  
[first 3 lines, edit the database name "reon" > "my\_db\_name" (name you chose)]   
  
> sudo mariadb -p  
[**in mariadb**]> use my\_db\_name;  
[**in mariadb**]> source /tmp/reon\_news/add\_news.sql  
[**in mariadb**]> exit

This should make the first issue of the news downloadable.

**Mobile-Relay Server Setup (calling functionality):**

*This part is only needed if you intend to run the server publicly and want your users to be able to "call" each other for link-trading and battling.*

To setup the mobile-relay server, we clone the git repo and just for it to be a bit easier to navigate all server files in the end, I copied it to [ /var/www/ ] next to the other files.  
  
> cd /var/www/  
> sudo git clone <https://github.com/REONTeam/mobile-relay.git>  
> cd mobile-relay  
  
Go into the config.ini  
  
> sudo nano config.ini  
  
and since my choice for the setup is mariadb, I uncommented the mysql part of the ini and set the sqlite part into a comment   
(delete the # from and including the [mysql]-lines and put # to the [sqlite] part of the .ini).  
Then we can go into mariadb and create our database and tables.  
  
> sudo mariadb -p   
[**in mariadb**]> source /var/www/mobile-relay/create\_db.sql  
[**in mariadb**]> exit  
> ./users.py  
> ./server.py  
  
This should create the table for creating your users (which happens automatically everytime a user connects to the relay server from their end).  
See clientside setup to connect to the relay-server as a client (Setup the Mobile Adapter).  
  
After a user connects the next time, they are automatically issued a new number and get a new entry in your relay\_users table in the mobile-relay database.  
If we want to change a number, or someone asks for a specific number, we have to edit the entry for that specific user. I set the the mobile-relay server up with mariadb and created a database user with the username "mobile" and password "mobile" which is the default:  
  
>mariadb -u mobile -pmobile mobile   
[logs you in as user "mobile" with password "mobile" and into the mobile database]  
  
Identify the number you want to change:  
  
[**in mariadb**]> SELECT \* FROM relay\_users;  
  
Then to change the number:  
  
[**in mariadb**]> UPDATE relay\_users SET number=newnumber WHERE number=oldnumber;  
  
Now the user should have a new number (should also probably restart their mobile adapter).

**-------------------------------------------------------------------------------------------------------------------**

This should complete our serverside setup and when we restart our machine (where the server is on), we should not forget to start up the needed services:  
- Mail-Server (needed)  
- Relay-Server (if used)

Everything else should already be running automatically.

**-------------------------------------------------------------------------------------------------------------------**

**CLIENTSIDE SETUP:**

**Building your pokecrystal:**

Follow the instructions on the respective github page. (See 3.)  
  
**INFO:** you do not have to build rgbds to make your pokecrystal rom! Just download the 0.6.1 release and extract it somewhere. Then when you start your make command just add the flag to tell make where your rgbds directory is with RGBDS=/path/to/rgbs/  
  
You can also use your own savefile from your Pokemon Crystal cartridge if you have a way to dump your savefile to your computer (GB Operator, GBxCart RW etc.).   
Otherwise just generate a new savefile.  
To use the dumped savefile: Downlaod the savefile-converter tool and convert your savefile to make it usable with the pokecrystal rom. (See 6.)

**Setup the Mobile Adapter:**

To connect to the server (***if installed on an external machine***) we need to configure the Mobile Adapter startup and connection to the server.  
For that we need the DNS Server to resolve the dion.ne.jp and gameboy.datacenter.ne.jp domains to your servers IP.

Before starting up the DNS-Server (downloaded from the dummy-servers github page), open up the dummydns.py file with your preferred text-editor.  
We need to **make changes to lines 19-23 (IPv4) and/or lines 24-30 (IPv6) in the script** to redirect the Mobile Adapter to the server (put in the IP in IPv4 and/or IPv6) instead of your local machine (or the long dead official servers). I also changed the port for the DNS-Server to 40 but feel free to leave it as is.

After this we can start up the DNS-Server and start up the emulator with a copy of Mobile Trainer GB.   
To connect the emulator to your Mobile Adapter, right-click on the emulator window and choose **Link > Listen** and keep the **port as 8765**. The title bar should now show a **"(listening)"** while it is waiting for us to start up the mobile-windows.exe or mobile-linux.

To direct the Mobile Adapter to the correct DNS and DNS port, open a terminal/cmd in the directory of the Mobile Adapter files and use the flags to specify the DNS, DNS Port etc.  
If you also set up the   
(See github-page for documentation):

**Windows:**.\nobile-windows.exe --dns1 127.0.0.1 (*--dns\_port 40*) --unmetered (--relay server.ip)

**Linux:** (NEED TO CHECK IF CORRECT USAGE, HAVENT USED ON LINUX YET)  
If it is not executable use "*chmod +x /path/to/file*" to make it executable.  
./mobile-linux --dns1 127.0.0.1 *(--dns\_port 40*) --unmetered

Now we should see a "(linked)" in the title bar.   
**INFO:** If it says (linked) but does not work, right-click the emulator window and reset the gameboy, this should fix it.  
  
We can now go ahead and click through the Mobile Trainer First-Time-Setup and input the user credentials we created serverside.

MAIL: user1234@reon.dion.ne.jp  
ID: g123456789   
PASS: pass4321

Let it connect to the server to finish up the setup and generate the Config.bin with your user credentials.

**-------------------------------------------------------------------------------------------------------------------**

This should conclude the clientside setup. In the future to use our setup,  
 we can just start up the services in this specific order:  
DNS-Server > Emulator and let it listen > Mobile Adapter

**Using the Mobile-Service:**

When you arrive in Goldenrod City go into the Pokemon Communication Center and choose your desired service (Online Trading/Battling with friends, the Trading Corner or the News Machine) and have fun!  
If it all was setup correctly you should be able to connect to the services, download news and be able to trade or battle.  
In Olivine city the battle tower should also have its doors opened and you should be able to start climbing the ranks!

**-------------------------------------------------------------------------------------------------------------------**