

Links

RPC API implementation status:

<https://docs.google.com/spreadsheets/d/1Xv0Gw5PzIRaVou2xrl6r7qySrcmOKjQWLBjJA73YnJM/edit#gid=0>

Pokemongo Dev subreddit:

<https://www.reddit.com/r/pokemongodev/>

Contributing/Extending the PokemonGo Java API.

Expanding the API will usually involve 4 steps:

1. Create a RequestMessage of the request you want to send in java (see <https://github.com/AeonLucid/POGOProtos/tree/master/pogo/Networking/Requests>)
2. Create a ResponseMessage in relation to the request (see <https://github.com/AeonLucid/POGOProtos/tree/master/pogo/Networking/Responses>)
3. Create a ServerRequest object
4. Send the response with ResponseHandler

These steps are then wrapped up nicely to have an easily to use API.

Create Request & Response Messages

JavaAPI is now using AeonLucids POGOProtos, so if you want to implement a response that does not have message/reply protos yet, you will need to submit a pull request with the relevant details to

<https://github.com/AeonLucid/POGOProtos/>

This is fairly straight forward, you should have two objects, one for the request and another for the response:

```
GetPlayerMessageOuterClass.GetPlayerMessage reqMsg =
GetPlayerMessageOuterClass.GetPlayerMessage.newBuilder().build();

GetPlayerResponseOuterClass.GetPlayerResponse response =
GetPlayerResponseOuterClass.GetPlayerResponse.newBuilder().build();
```

Creating Server Request object

The server request takes 3 fields:

1. The Request Type (see: <https://github.com/AeonLucid/POGOProtos/blob/master/pogo/Networking/Requests/RequestType.proto>)
2. The REQUEST message made above (in this GetPlayerMessage)
3. The RESPONSE message made above (getPlayerResponse)

So:

```
ServerRequest serverRequest = new ServerRequest(RequestTypeOuterClass.RequestType.GET_PLAYER,
reqMsg, response);
```

Sending Requests

To send the newly created server request, you pass it to the RequestHandler

(com.pokegoapi.api.pokemonGo.getRequestHandler()) and then send the request:

```
getRequestHandler().request(serverRequest);

getRequestHandler().sendServerRequests();
```

Please note that the request handler can send multiple requests at once incase it needs to simulate the game more closely, but one request seems to work fine.

After this, the payload of the response will be passed into the response object made earlier.

Wrapping The API nicely

One of the main focuses of the API is to make it easy/user friendly for developers, so we want to try and keep them away from the Proto objects and have wrappers etc for them. Usually this is just a bunch of taking data from the proto and setting it in an object.

Please see the `PokemonGo.java` api for examples

Contacting/Slack/Discussion

Java API has a channel on the pokemongo RE group:

<https://pkre.slack.com/>

Invites:

<https://shielded-earth-81203.herokuapp.com/>

#javaapi channel for github alerts and discussion.