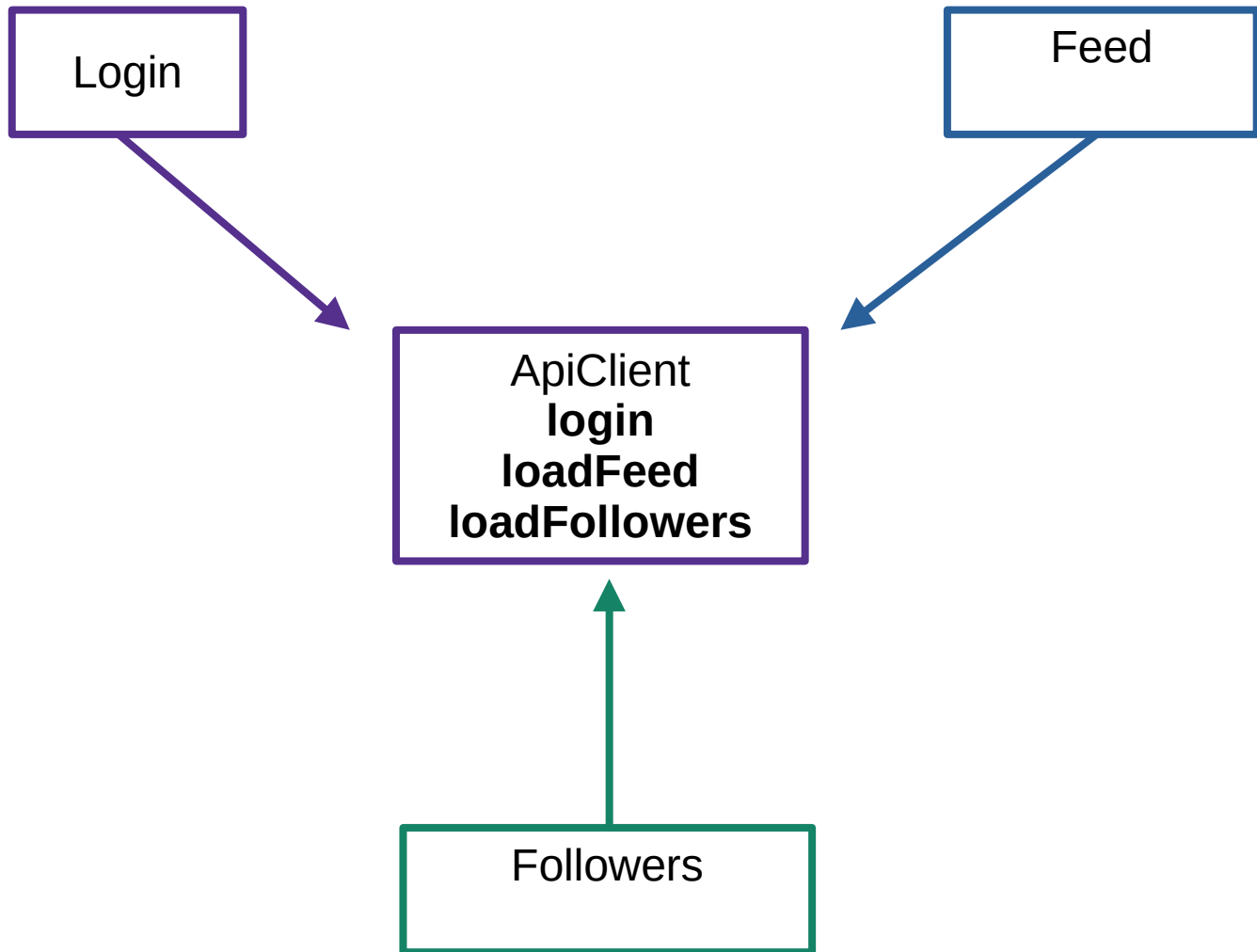Version I : Dependencies, simplified
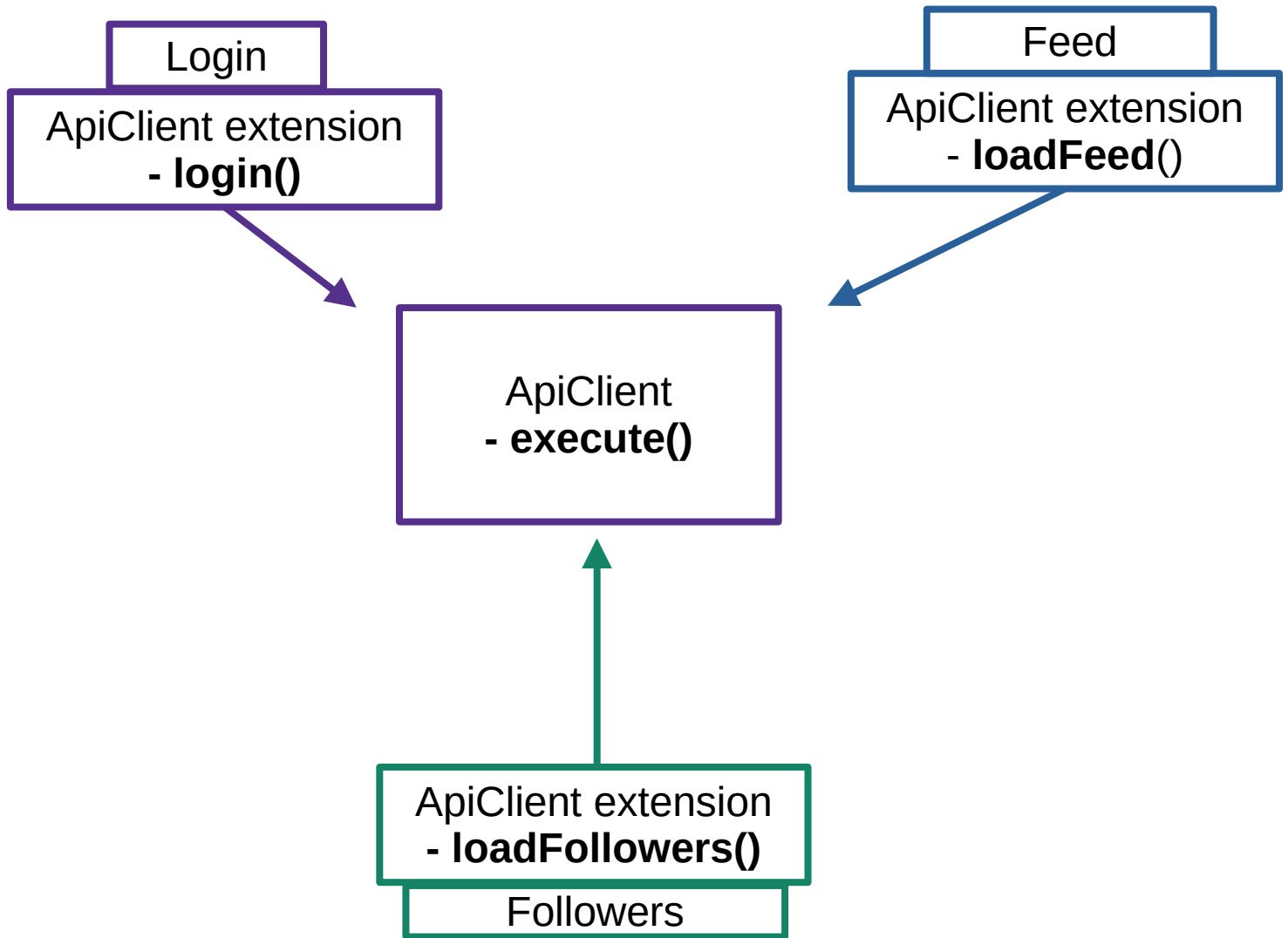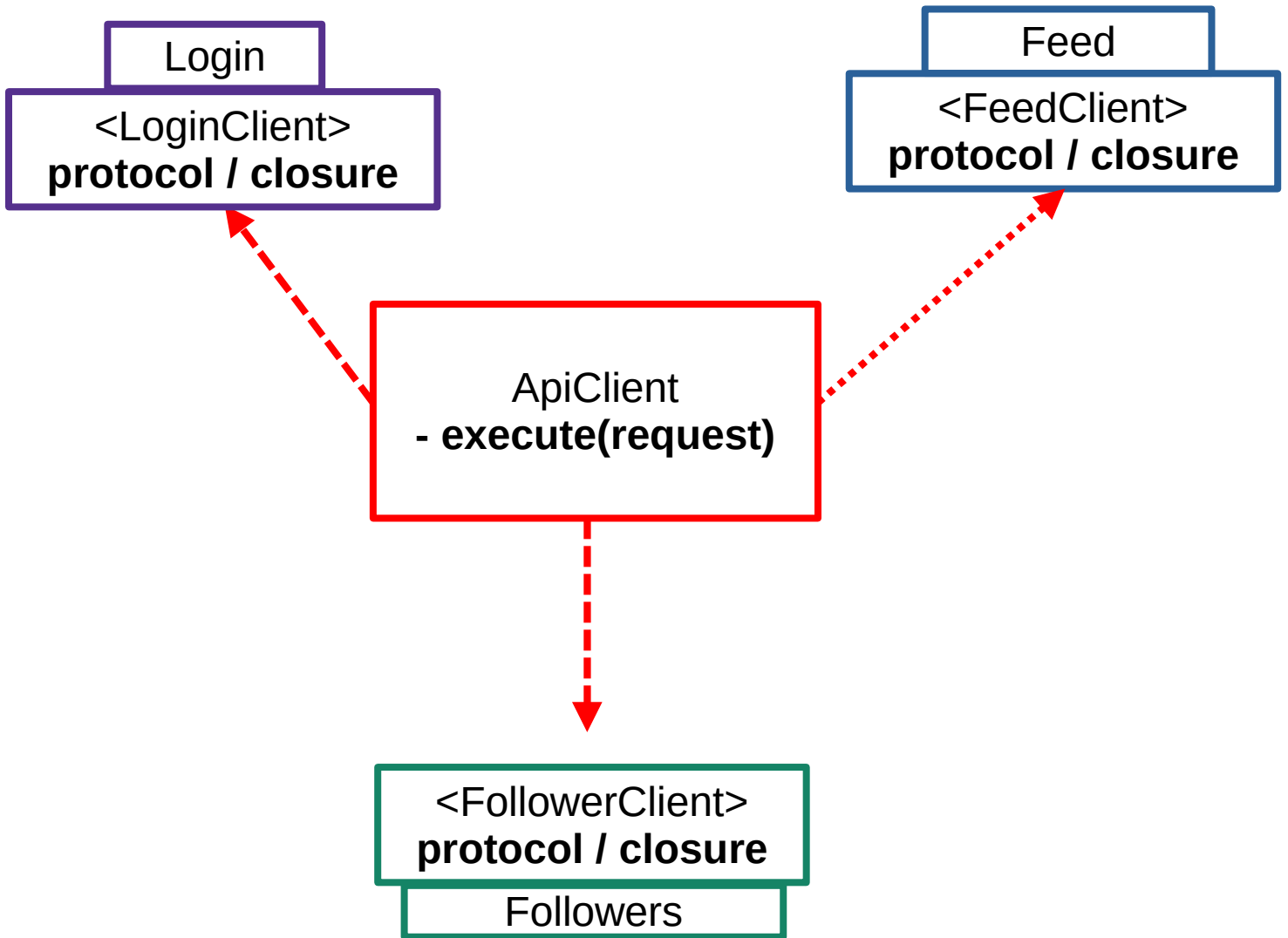

PROS and CONS

CONS:
- Now a concrete type.
- If I want to add another method to the ApiClient, I have to
  recompile all the other modules including the ApiClient.
- I can't move the Login without the ApiClient, easily

Version II : Dependencies via Swift extensions

Login

ApiClient extension
- **login()**

Feed

ApiClient extension
- **loadFeed**()

ApiClient
- **execute()**

ApiClient extension
- **loadFollowers()**

Followers

Most developers stop here

Version III : Dependency inversion via protocols or closures

Login

<LoginClient>
protocol / closure

Feed

<FeedClient>
protocol / closure

ApiClient
- execute(request)

<FollowerClient>
protocol / closure

Followers

Version IV : Completely modular architecture using Adapters

Login

<LoginClient>
protocol / closure

LoginClient
Adapter

ApiClient

FeedClient
Adapter

Feed

<FeedClient>
protocol / closure

FollowerClient
Adapter

*Main
module*

<FollowerClient>
protocol / closure

Followers