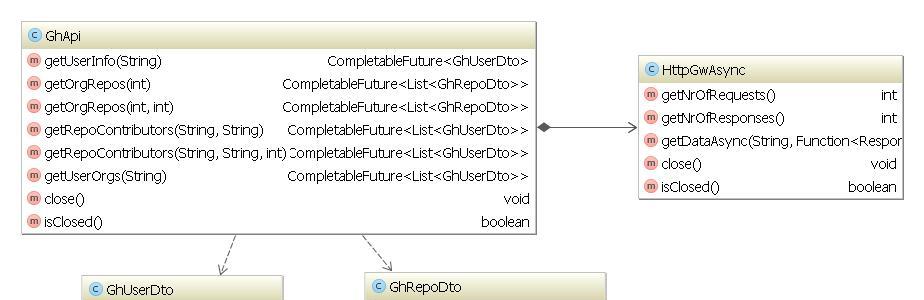
A biblioteca githubGw oferece um modelo de domínio assíncrono sobre um subconjunto de informação obtida da Web API do Github.

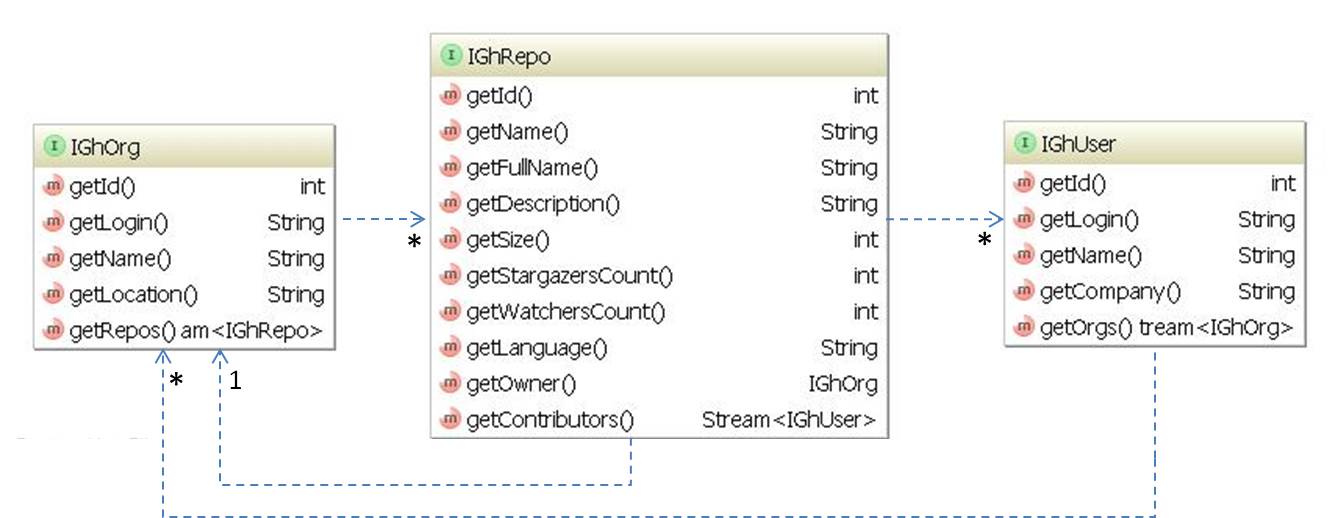
Para realizar os pedidos à Web API do Github é disponibilizada a classe GhApi cujos métodos retornam informação na forma de objectos imutáveis do tipo GhUserDto e GhRepoDto.

A classe pt.isel.mpd.githubgw.webapi.GhApi depende da classe pt.isel.mpd.util.HttpGwAsyncNio para realizar pedidos HTTP assíncronos.

**NOTA**: a classe GhApi necessita de um token de acesso à API Web do Github.

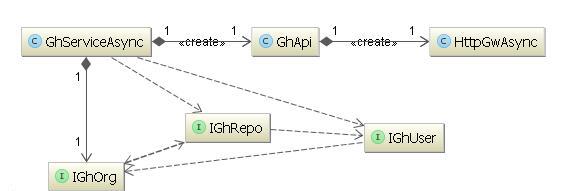


O modelo de domínio assíncrono obedece ao desenho das interfaces IGhOrg, IGhRepo e IGhUser.



Uma instância de IGhOrg representa uma organização sobre a qual podem ser obtidos uma sequência dos repositórios dessa organização. Por sua vez, de cada instância de IGhRepo podem ser obtidos os seus colaboradores (*contributors*).

Para obter a raíz deste agregado implemente a classe GhServiceAsync:



**Nota**:

* Todas as associações no modelo de domínio são iniciadas **assincronamente** no momento da instanciação de um objecto de domínio. Por exemplo, quando é obtida uma organização deve ser imediatamente pedidos os repositórios dessa organização. O exemplo seguinte demonstra esta abordagem:

|  |
| --- |
| **try**(HttpGwAsyncNio httpGw = **new** HttpGwAsyncNio ()) {  **try** (GhServiceAsync gh = **new** GhServiceAsync(**new** GhApi(httpGw))) {  Future<IGhOrg> org = gh.getOrg(**"zendframework"**);  *assertEquals*(1, httpGw.getNrOfRequests()); *// 1.a One request through GhApi.getUserInfo  assertEquals*(0, httpGw.getNrOfResponses());   org.get(); *// Wait for Response  assertEquals*(1, httpGw.getNrOfResponses()); *// 1.b One response received for GhApi.getUserInfo  assertEquals*(2, httpGw.getNrOfRequests()); *// 2.a One request through GhApi.getOrgRepos* org.get().getRepos(); *// -> Wait for Stream<IGhRepo>  assertEquals*(2, httpGw.getNrOfResponses()); *// 2.b One response received for GhApi.getOrgRepos*  }  } |

* Os pedidos à Web API do Github que retornem múltiplos elementos são paginados em 30 elementos, por omissão. As restantes páginas de uma associação múltipla são obtidas de forma *lazy*.

Por exemplo, na instanciação de um IGhRepo são obtidos os 30 primeiros repositórios, sendo obtidos os restantes apenas se iteramos além do 30º elemento e assim sucessivamente.

Poderá estender as funcionalidades de GhApi de modo a parametrizar o número de elementos por página ou até obter informação dos cabeçalhos de resposta http.

* Numa segunda consulta dos repositórios da mesma organização, não deverão existir pedidos HTTP à Web API estando já os resultados *cached*.
* Existe uma referência circular de IGhUser para IGhOrg, pelo que é ser mantido um identity map em GhServiceAsync que evita que sejam duplicados objectos que representam a mesma identidade, neste caso a mesma organização.