**Кеширующий прокси**

Некоторые методы могут выполняться очень долго, хочется иметь возможность кешировать результаты возврата. Иногда хочется чтобы результаты расчета могли сохраняться при перезапуске JVM.

Например, у нас есть интерфейс Servicecметодом doHardWork(). Повторный вызов этого метода с теми же параметрами должен возвращать рассчитанный результат из кэша.

**void**run(Serviceservice) {  
**double**r1 = service.doHardWork(**"work1"**, 10); //считаетрезультат  
**double**r2 = service.doHardWork(**"work2"**, 5);//считаетрезультат  
**double**r3 = service.doHardWork(**"work1"**, 10); //результатизкеша  
}

Должна быть возможность тонкой настройки кеша:

1. Указывать с помощью аннотаций, какие методы кешировать и как: Просчитанный результат хранить в памяти JVM или сериализовывать в файле на диск.
2. Возможность указывать, какие аргументы метода учитывать при определении уникальности результата, а какие игнорировать(по умолчанию все аргументы учитываются). Например, должна быть возможность указать, что doHardWork()должен игнорировать значение второго аргумента, уникальность определяется только по Stringаргументу.

**double**r1 = service.doHardWork(**"work1"**, 10); //считает результат  
**double**r2 = service.doHardWork(**"work1"**, 5); // результат из кеша, несмотря на то что второй аргумент различается

1. Если возвращаемый тип это List–возможность указывать максимальное количество элементов в нем. То есть, если нам возвращается Listс size = 1млн, мыможем сказать что в кеше достаточно хранить 100т элементов.
2. Возможность указывать название файла/ключа по которому будем храниться значение. Если не задано - использовать имя метода.
3. Если мы сохраняем результат на диск, должна быть возможность указать, что данный файл надо дополнительно сжимать в zip архив.
4. Любые полезные настройки на ваш вкус.
5. Все настройки кеша должны быть optionalи иметь дефолтные настройки.
6. Все возможные исключения должны быть обработаны спонятнымописание, что делать, чтобы избежать ошибок. (Например, если вы пытаетесь сохранить на диск результат метода, но данный результат не сериализуем, надо кинуть исключение с понятным описанием как это исправить)
7. Логика по кешированию должна навешиваться с помощью DynamicProxy. Должен быть класс CacheProxy с методомcache(), который принимает ссылку на сервис и возвращает кешированную версию этого сервиса.CacheProxyдолжен тоже принимать в конструкторе некоторые настройки, например рутовую папку в которой хранить файлы, дефолтные настройки кеша и тд.

Дизайн аннотаций, атрибутов аннотаций, классов реализаций остается на ваш вкус. Код должен быть читаем, классы не перегружены логикой, классы должны лежать в нужных пакетах.

Примервключениякеширования(можно менять названия классов, методов, аннотаций и атрибутов):

CacheProxycacheProxy = **new** CacheProxy(...);  
Service service = cacheProxy.cache(**new** ServiceImpl());  
Loader loader = cacheProxy.cache(**new** LoaderImpl());

**interface** Service {  
@Cache(cacheType = FILE, fileNamePrefix = **"data"**, zip = **true**, identityBy = {String.**class**, **double**.**class**})  
 List<String> run(String item, **double** value, Date date);  
  
@Cache(cacheType = ***IN\_MEMORY***, listList = 100\_000)  
 List<String> work(String item);  
}