链滴

SpringBoot 系列 -- 整合 Redis 做缓存

作者：Qiyue0726
原文链接：https://ld246.com/article/1564748999066
来源网站：链滴

<blockquote>
<h2 id="一--缓存的作用">一. 缓存的作用</h2>
</blockquote>
当应用体积大了，用户量上去了，数据规模也越来越大之后，数据库查询操作将成为用户体验的颈，这时使用缓存会是一个非常好的解决办法。Spring 开始从 3.1 开始就为我们提供了基于注解的存支持，通过注解方式低侵入地为我们的应用提供缓存支持。在 SpringBoot 中，更是以一系列自动置的方式使我们能更加方便的使用缓存功能。
<blockquote>
<h2 id="二--几个重要的注解">二. 几个重要的注解</h2>
</blockquote>
<table>
<thead>
<tr>
<th>名称</th>
<th>解释</th>
</tr>
</thead>
<tbody>
<tr>
<td>@EnableCaching</td>
<td>开启缓存注解</td>
</tr>
<tr>
<td>@Cacheable</td>
<td>主要针对方法配置，能够根据方法的请求参数对其结果进行缓存</td>
</tr>
<tr>
<td>@CachePut</td>
<td>保证方法被调用，又希望结果被缓存。
 与 @Cacheable 区别在于是否每次都调用方法，用于新增、更新</td>
</tr>
<tr>
<td>@CacheEvict</td>
<td>清空缓存</td>
</tr>
</tbody>
</table>
<blockquote>
<h2 id="三---cacheeable--cacheput--cacheevict-的几个常用参数">三. @Cacheeable/@CacheEvict 的几个常用参数</h2>
</blockquote>
<table>
<thead>
<tr>
<th>名称</th>
<th>解释</th>
<th>示例</th>
</tr>
</thead>
<tbody>
<tr>
<td>value</td>

```
<td>缓存块的名称，必须指定一个</td>
<td><code>@Cacheable(value = " default ")</code></td>
</tr>
<tr>
<td>key</td>
<td>缓存的 key，可以为空，如果指定要按照 SpEL 表达式编写，<br>如果不指定，则缺省按照方
的所有参数进行组合</td>
<td><code>@Cacheable(value = " default ",key = " 'info' ")</code> <br><code>@Cacheab
e(value = " default ",key = " #id ")</code></td>
</tr>
<tr>
<td>sync <br>(@Cacheable)</td>
<td>指示底层将缓存锁住，使只有一个线程可以进入计算，而其他线程堵塞，直到返回结果更新到
存中。<br> 可以避免缓存击穿</td>
<td><code>@Cacheable(value = " default ",sync = true)</code></td>
</tr>
<tr>
<td>allEntries  <br>(@CacheEvict )</td>
<td>是否清空所有缓存内容，缺省为 false，如果指定为 true，则方法调用后将立即清空所有缓存</
d>
<td><code>@CachEvict(value = ”  default ”  ,allEntries = true)</code></td>
</tr>
</tbody>
</table>
<blockquote>
<h2 id="四--开始使用-----整合-Redis">四. 开始使用 --  整合 Redis</h2>
</blockquote>
<p><strong>注：</strong> 需先启动 Redis 服务器</p>
<h3 id="1--导入-Maven-包">1. 导入 Maven 包</h3>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">        &lt;dependency&gt;
</span></span><span class="highlight-line"><span class="highlight-cl">            &lt;groupI
&gt;org.springframework.boot&lt;/groupId&gt;
</span></span><span class="highlight-line"><span class="highlight-cl">            &lt;artifact
d&gt;spring-boot-starter-data-redis&lt;/artifactId&gt;
</span></span><span class="highlight-line"><span class="highlight-cl">        &lt;/depend
ncy&gt;
</span></span></code></pre>
<h3 id="2--配置-application-yml">2. 配置 application.yml</h3>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">spring:
</span></span><span class="highlight-line"><span class="highlight-cl">   redis:
</span></span><span class="highlight-line"><span class="highlight-cl">     #指定Redis数
库索引，在redis-cli中使用select index切换数据库
</span></span><span class="highlight-line"><span class="highlight-cl">     database: 0
</span></span><span class="highlight-line"><span class="highlight-cl">     #Redis服务器
址
</span></span><span class="highlight-line"><span class="highlight-cl">     host: 127.0.0.1
</span></span><span class="highlight-line"><span class="highlight-cl">     #Redis连接端
，默认为 6379
</span></span><span class="highlight-line"><span class="highlight-cl">     port: 6379
</span></span><span class="highlight-line"><span class="highlight-cl">     #Redis连接密
，默认为空
</span></span><span class="highlight-line"><span class="highlight-cl">     password:
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">    #连接超时时间
</span></span><span class="highlight-line"><span class="highlight-cl">    timeout: 0
</span></span><span class="highlight-line"><span class="highlight-cl">    #连接池
</span></span><span class="highlight-line"><span class="highlight-cl">    pool:
</span></span><span class="highlight-line"><span class="highlight-cl">      #最大连接数
使用负值表示没有限制）
</span></span><span class="highlight-line"><span class="highlight-cl">      max-acive: 1
00
</span></span><span class="highlight-line"><span class="highlight-cl">      #最大阻塞等
时间
</span></span><span class="highlight-line"><span class="highlight-cl">      max-wait: -1
</span></span><span class="highlight-line"><span class="highlight-cl">      #最大空闲连

</span></span><span class="highlight-line"><span class="highlight-cl">      max-idle: 10
</span></span><span class="highlight-line"><span class="highlight-cl">      #最小空闲连

</span></span><span class="highlight-line"><span class="highlight-cl">      min-idle: 2
</span></span></code></pre>
<h3 id="3--在启动类上开启缓存注解">3. 在启动类上开启缓存注解</h3>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">@SpringBootApplication
</span></span><span class="highlight-line"><span class="highlight-cl">@EnableCaching
/ 开启缓存注解
</span></span><span class="highlight-line"><span class="highlight-cl">public class Anim
Application {
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    public static vo
d main(String[] args) {
</span></span><span class="highlight-line"><span class="highlight-cl">        SpringApplic
tion.run(AnimaApplication.class, args);
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">}
</span></span></code></pre>
<h3 id="4---缓存--Cacheable">4. 缓存 @Cacheable</h3>
<p><code>@Cacheable</code> 注解会先查询是否已经有缓存，有会使用缓存，没有则会执行
法并缓存。该注解还可预防缓存穿透。</p>
<pre><code class="highlight-chroma"><span class="highlight-line"><span class="highlight
cl">    /**
</span></span><span class="highlight-line"><span class="highlight-cl">     * key:可使用参
时，即 key#参数名，亦可直接使用字符串 ‘key’
</span></span><span class="highlight-line"><span class="highlight-cl">     * sync：避免
存击穿
</span></span><span class="highlight-line"><span class="highlight-cl">     */
</span></span><span class="highlight-line"><span class="highlight-cl">    @Cacheable(va
ue = "default",key = " 'anima' + #page",sync = true)
</span></span><span class="highlight-line"><span class="highlight-cl">    public ArrayLis
&amp;lt;animaInfo&amp;gt; getAnimaInfo(int page,int limit) {
</span></span><span class="highlight-line"><span class="highlight-cl">        return anima
nfoMapper.selectAnima(page,limit);
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span></code></pre>
<p><strong>注：</strong><code>@Cacheable</code> 不支持设置缓存过期时间和自动更新<
p>
```

<h3 id="5--更新--CachePut">5. 更新 @CachePut</h3>
<code>@CachePut</code> 标注的方法在执行前不会去检查缓存中是否存在之前执行过的结，而是每次都会执行该方法，并将执行结果以键值对的形式存入指定的缓存中。
<pre><code class="highlight-chroma">   @CachePut(value = "default",key = " 'anima' + #anima.id")
    public int save(Anima anima) {
        return animaInfoMapper.saveAnima(anima);
    }
</code></pre>
<h3 id="6--清除--CacheEvict">6. 清除 @CacheEvict</h3>
<code>allEntries</code> 参数表示是否需要清除缓存中的所有元素。默认为 false，表示不需。当指定了 allEntries 为 true 时，Spring Cache 将忽略指定的 key。有的时候我们需要 Cache 一清除所有的元素。
<pre><code class="highlight-chroma">   @CacheEvict(value = "default",allEntries = true)
    public int del(Aima anima) {
        return animaInfoMapper.delAnima(anima.id);
    }
</code></pre>