ano-prise-caches

ano-prise Caches.

ano-prise offers a set of built-in caches for object caching in services. Currently we offer two types of caches which have several implementations:

Cache - roundrobin caches for hit-reduction.

Definition

RoundRobin caches have a limited size. If the cache is full a newly placed element will overwrite the least recently placed element. The access time of the element doesn't matter, only the put order is considered. This is pretty effective and simple at same time.

Implementations

Following implementations of the RoundRobin Cache are available:

Cache	Effect
RoundRobinHardwiredCache	Cache of defined size and capacity. Useful if the size of the en-cached objects is known (for example user objects
RoundRobinSoftReferenceCache	Cache of defined capacity. However it uses SoftReferences instead of direct References, which prevents the cache from going OutOfMemory. Useful for en-caching objects of different sizes, for example messages, texts etc
ExpiringCache	A wrapper around another cache implementation which adds an expiration check to each en-cached element and removes it on access if the element has expired

Interface

SVN: **ano-prise/trunk/java/net/anotheria/anoprise/cache/Cache.java**

groovy: Notify your Confluence administrator that "Scripting for Confluence" requires a valid license. Reason: EXPIRED

```
def data = new URL('http://svn.anotheria.net/opensource/ano-prise/trunk/java/net/anotheria/anoprise/cache/Cache.java').getText()
println '
```

```
' + data + '
```

```
'
```

BoundedCache

Definition

BoundedCache differes from RoundRobin cache in only one aspect, if its full, it refuses to accept more elements, until some elements are removed. To enable this behavior it offers no *put* method, but an *offer* method instead. If *offer* returns true the element is accepted, otherwise it returns false.

Implementations

Following implementations of the RoundRobin Cache are available:

Cache	Effect
BoundedHardwiredCache	Cache of defined size and capacity. Useful if the size of the en-cached objects is known (for example user objects
BoundedSoftReferenceCache	*yet missing*
BoundedExpiringCache	*yet missing*

Interface

SVN: **ano-prise/trunk/java/net/anotheria/anoprise/cache/BoundedCache.java**

groovy: Notify your Confluence administrator that "Scripting for Confluence" requires a valid license. Reason: EXPIRED

```
def data = new URL('http://svn.anotheria.net/opensource/ano-prise/trunk/java/net/anotheria/anoprise/cache/BoundedCache.java').getText()
println '
```

```
' + data + '
```

```
'
```

Caches Utility.

Similar to Collections or Arrays in Core Java, *Caches* offers some useful methods to create preconfigured caches.

Methods
public static final <K,V> Cache<K,V> createSoftReferenceCache(String name)
public static final <K,V> Cache<K,V> createSoftReferenceCache(String name, int startSize, int maxSize)
public static final <K,V> Cache<K,V> createHardwiredCache(String name)
public static final <K,V> Cache<K,V> createHardwiredCache(String name, int startSize, int maxSize)
public static final <K,V> Cache<K,V> createHardwiredExpiringCache(String name, int startSize, int maxSize, int expirationTime)
public static final <K,V> Cache<K,V> createSoftReferenceExpiringCache(String name, int startSize, int maxSize, int expirationTime)
public static final <K,V> Cache<K,V> createConfigurableSoftReferenceCache(String name)
public static final <K,V> Cache<K,V> createConfigurableHardwiredCache(String name)

For details on the methods please check the javadoc documentation.

Configuration

Configurable caches support dynamic configuration and re-configuration via ConfigureMe.
The configuration supports three configuration options:

Option	Type	Meaning
cacheOn	Boolean	Switches the cache on and off.
startSize	Int	Initial size of the cache. If the size is reached the cache will enlarge itself.
maxSize	Int	Max size of the cache. The cache won't enlarge itself above this.

Example of a configuration:

SVN: **ano-prise/trunk/etc/appdata/cachetest.json**

groovy: Notify your Confluence administrator that "Scripting for Confluence" requires a valid license. Reason: EXPIRED

```
def data = new URL('http://svn.anotheria.net/opensource/ano-prise/trunk/etc/appdata/cachetest.json').getText()
println '
```

```
' + data + '
```

'