**AfterSaveListener**

AfterSaveListener interface is used to collect these events and handle them according to needs. Custom listener should collect events of specific type and perform custom business logic.

A listener can be notified about newly created events in two modes:

**Synchronous:** In this mode event is notified to a listener immediately after it is created.

**Asynchronous:** In this mode information about new event is put in a queue first. Then depending on the configuration it is also notified to a listener.

The asynchronous mode is enabled by default with some default configuration. Modes are set globally for all listeners. The current configuration of after save events is:

Property

* core.aftersave.async=true
* core.aftersave.queuesize=1024

This design does not scale:

* limitation of database connections
* fixed size of the aftersave queue

It has also a known risk of deadlocks if maximum size of the queue is reached. Furthermore, when hybris server is stopped, events published in the queue and that have not been handle by AfterSaveEventPublisher threads are lost.  Also if after save listeners would process in synchronous way (core.aftersave.async=false) this also would not help. It has also a flaw, which may have significant influence on the performance (with each create/read/update operation list of listeners are called in the same thread).

It and has also a known risk of deadlocks if maximum size of the queue is reached. Furthermore, when hybris server is stopped, events published in the queue and that have not been handle by AfterSaveEventPublisher threads are lost.

Also if after save listeners would process in synchronous way (core.aftersave.async=false) this also would not help. It has also a flaw, which may have significant influence on the performance (with each create/read/update operation list of listeners are called in the same thread).