


```
void RealTimeAsyncUpdater::RealTimeAsyncUpdateDispatcher::add (RealTimeAsyncUpdaterMessage& m)
{
    const ScopedLock sl (lock);
    jassert (! updaters.contains (&m));
    updaters.add (&m);
}
```

```
void RealTimeAsyncUpdater::RealTimeAsyncUpdateDispatcher::remove (RealTimeAsyncUpdaterMessage& m)
{
    const ScopedLock sl (lock);
    updaters.removeFirstMatchingValue (&m);
}
```

```
void RealTimeAsyncUpdater::RealTimeAsyncUpdateDispatcher::serviceUpdaters()
{
    const ScopedLock sl (lock);

    for (auto updater : updaters)
        updater->serviceMessage();
}
```







```
void serviceMessage()  
{  
    if (shouldDeliver.compareAndSetBool (0, 1))  
        owner.handleAsyncUpdate();  
}
```




```
void RealTimeAsyncUpdater::RealTimeAsyncUpdateDispatcher::add (RealTimeAsyncUpdaterMessage& m)
{
    const ScopedLock sl (lock);
    jassert (! updaters.contains (&m));
    updaters.add (&m);
}
```

```
void RealTimeAsyncUpdater::RealTimeAsyncUpdateDispatcher::remove (RealTimeAsyncUpdaterMessage& m)
{
    const ScopedLock sl (lock);
    updaters.removeFirstMatchingValue (&m);
}
```

```
void RealTimeAsyncUpdater::RealTimeAsyncUpdateDispatcher::serviceUpdaters()
{
    const ScopedLock sl (lock);

    for (auto updater : updaters)
        updater->serviceMessage();
}
```



```
void serviceMessage()
{
    if (shouldDeliver.compareAndSetBool (0, 1))
        owner.handleAsyncUpdate();
}
```



```
juce::AsyncUpdater  
Average = 20 microsecs, minimum = 5 microsecs, maximum = 102 microsecs
```