Rendering

As RFB protocol assumes receiving of remote desktop updates as a bunch of encoded rectangles, there are 2 options for rendering them:

1. Sequential rendering
2. Collecting of updates

Sequential Rendering

Every single received update rectangle is rendered on the user screen as it is received and decoded. And when the previous rectangle is rendered, then processing of the next received update rectangle starts. And so on.

This is the classic option used in the TightVNC Viewer and the only option in the Remote Core SDKs version prior v2020.3

Updates Collecting

Working this way the SDK receiving a bunch of updates rectangles is trying firstly to decode all of them and save on the internal framebuffer and only then render it on the user screen.

In most cases, this approach allows to improve the rendering quality and avoid the tearing effect. However, this way also has an obvious flaw. If you have low bandwidth, it may cause rendering lags.

To avoid such lags there is an option to set updates flushing timeout. The main idea of this timeout is to render all currently decoded rectangles from the received bunch on the user screen after the specified timeout is expired since the moment of receiving the bunch. It allows to escape the visual lags and make the rendering smoother.

Remote Core SDK settings

There are 2 settings to control rendering behavior in this way:

- **UseUpdatesCollecting** - gets or sets the value, which indicates whether collecting of the remote framebuffer updates
- **UpdatesFlushingTimeout** - gets or sets the timeout in milliseconds after which all collected changes in UpdateCollecting mode will be flushed to the UI. Setting it to 0 will disable this timeout.

Both of the settings are acceptable for low-level ViewerCore component as well as for both UI-controls: WPF and WinForms