ContextView architecture

leo franchi

Overview of the Context View framework for the 2009 Amarok DevSprint.

1 Classes

1.1 ContextView

- Subclasses Plasma::View (and hence QGraphicsView). Is passed the containment that it should be a view on.
- Kept in sync by plasma with the containment size and location, updates the containment size when it is itself resized.
- Loads and saves applets (indirectly, by loading and saving containment)

1.2 ContextScene

- Plasma::Corona (stub implementation, everythin done by Qt)

1.3 ContextObserver

- Notifies engines and applets about state changes to the contextview. Switches between sending Home and Current messages.

1.4 LyricsManager

- In the middle between the lyrics scripts (and AmarokLyricsScript.h/cpp) and the lyrics applets. Lyrics engine is an observer to this, and this is what handles the lyrics from the scripts themselves.

1.5 ToolbarView

- The applet toolbar is not where it appears to be. In order for scrolling to work (which is now disabled), the toolbar exists at around (0, 2000) in screen coordinates. The ToolbarView is fixed over this location on the scene. Done like this so the popup applet menu can be visualized in the main ContextView above.
- Also, the whole overlay (config mode) item thing is QWidget-based, not QGraphicsItem-based. This is so we can lock the horizontal position of the widgets when they are dragged around. Hence, all the config mode logic is here, creating and destroying the overlay, connecting the overlay items with the main CV layout, etc.

1.6 containments/verticallayout/VerticalToolbarContainment

- The main containment that managed the applets in the ContextView.
- Mostly exists to contain the VerticalAppletLayout, which actually manages the applets. Recieves the add requests, and passes them along accordingly.

1.7 containments/verticallayout/VerticalAppletLayout

- The actual applet layout manager. Loads/saves applets to config file. Manages heights and locations of each applet, inserts applets in appropriate places.

1.8 widgets/* (miscellaneous)

- ToolBoxMenu is the popup menu that is where you select applets from. It has ToolBoxIcons (not really icons, more the menu items).
- TrackWidgets are displayed in the Current applet when a track is not playing—each represents 1 recently played track (for example).

2 Initialization notes

The startup process is a bit convoluted. Here is what happens:

1. MainWindow creates the ContextWidget (containter for CV). MainWindow creates the ContextScene. MainWindow connects to the Plasma::Corona containmentAdded() signal.
2. When created, ContextScene loads the VerticalToolbarContainment (via KTrader).
3. MainWindow::createContextView() is called with the loaded containment.
4. MainWindow creates ContextView, with the loaded containment and ContextScene, and Home "mode" is shown.
5. MainWindow also creates ToolbarView with the loaded containment and ContextScene.

3 Applets

- Applets are normal KDE plugins, they live in our SVN but are built completely separately and installed into system directories. KDE loads them up on-demand with the KTrader system.
- Applets derive from Context::Applet, which is basically Plasma::Applet with some convenience functions (like background painting) to make all applets look similar and fit in. Use shrinkTextSizeToFit(), drawRoundedRectAroundText(), addGradientToAppletBackground(), standardPadding().
- Although the Plasma Way (tm) is to use SVG to theme widgets, after extensive trials using SVG the standard practice for the Amarok CV is to avoid SVG if possible. This is because it is slow to resize, which is much more of an issue in an app than on the desktop. A SVG-heavy applet can easily make the whole of amarok difficult to resize.
- Do not forget the K_EXPORT_AMAROK_APPLET in the header file.
- Important methods to implement are init() (do all initializations here), paintInterface() (do all painting here), constraintsEvent() (do all layouting here), and dataUpdate() (respond to new data from engine here). Also, contextualActions() can return a list of additional QActions to be inserted in the default right-click menu.
- DO NOT DO ANYTHING EXCEPT FOR PAINT IN paintInterface. If you trigger an update you will end up in an infinite loop.
- See plasma tutorials and docs for more info.

4 DataEngines

- Also normal KDE plugins, quite straightforward really. Read the KDE api docs for more info.