Besides the configuration in the *Styling* panel at design time, the automatic loading indicator can be also turned on and off via scripting:

```
// Enable/disable automatic loading indicator
Application.setAutomaticBusyIndicatorEnabled(true);
```

5.20.2 Script API

Besides displaying an automatic loading indicator, you can show or hide the loading indicator via a script API as well.

The script API is available for the following objects: `Application`, `Popup`, and container widgets, such as `TabStrip` and `Panel`. The text shown along with the indicator can be configured via an optional parameter.

```
// Show loading indicator, add text to loading indicator if text is specified
Application.showBusyIndicator("Loading the application"); // cover the whole
application page
Popup_1.showBusyIndicator("Loading the popup"); // cover the popup
TabStrip_1.showBusyIndicator("Loading the tab strip"); // cover the tab strip
Panel_1.showBusyIndicator("Loading the panel"); // cover the panel

// Hide loading indicator
Application.hideBusyIndicator();
Popup_1.hideBusyIndicator();
TabStrip_1.hideBusyIndicator();
Panel_1.hideBusyIndicator();
```

5.21 Bookmark API

You, as an analytic application developer, can use the Bookmark API to control bookmarks in analytic designer scripts.

5.21.1 Saving Bookmarks

You can save a bookmark with the `save()` script API method.

Example:

In the following example, the state of the analytic application is captured in a bookmark with the bookmark name “application-bookmark”. The bookmark is available globally. The save operation overwrites any previous bookmark of this name:

```
BookmarkSet_1.save("application bookmark", true, true);
```

5.21.2 Saving Additional Properties with Bookmarks

When saving a bookmark, you can save additional information (properties) along with the bookmark.

You can accomplish this with the following APIs:

The `saveBookmark()` script API method of the BookmarkSet component. It has the following signature: