Adobe® Flex® SDK "Hero" Release Notes

Welcome to the Adobe Flex SDK "Hero" preview release.

About Flex

Flex SDK includes the Flex framework (also known as the Flex class library), Flex command-line compilers, the Flex debugger, the ASDoc utility, and the debugger version of Adobe® Flash® Player and Adobe® AIR® runtimes. Use the Flex SDK to develop, compile, and deploy Flex applications that connect to XML and SOAP web services or connect to various server technologies such as PHP, Adobe® ColdFusion®, Java, and .NET using a server technology such as BlazeDS. Flex applications can be built and configured to target browsers via the Flash Player runtime and desktops or mobile devices via the AIR runtime.

What's New

Hero Desktop

- Spark DataGrid
- Spark Image
- Enhancements to Spark BitmapImage primitive
- Spark Form
- Spark Formatters (Currency, Number, and DateTime formatters)
- Native support for OSMF 1.0 and TLF 2.0
- RSL Enhancements
- Size Report

Hero Mobile

- MobileApplication component
- View & ViewNavigator components
- ActionBar component
- Mobile- and touch-optimized skins for Button, CheckBox, RadioButton, and TextInput
- Touch-based Scroller and List scrolling
- Mobile-optimized item renderer

Contains modified AIR 2.5 SDK which fixes SDK-28348.

Install your software

Installation instructions

The Flex SDK installation is delivered as a ZIP file and contains the Flex framework, Adobe AIR framework, and command-line tools, such as the mxmlc command-line utility, Adobe AIR command-line utility, the ASDoc utility, the Flex command-line debugger, and the debugger version of Flash Player.

Uninstall Current Flash Player

You should use Flex SDK with the latest version of the debug Flash Player 10. Prior to installing the Flex SDK, you should uninstall your current Flash Player.

Windows–Plugin-based browsers
Run the appropriate uninstaller available from this Tech Note.

Macintosh
Run the appropriate uninstaller available from this Tech Note.

Linux
Manual removal (for users who installed the plug-in via Install script):

- Delete the libflashplayer.so binary and the flashplayer.xpt file in directory /home/<user>/.mozilla/plugins/

RPM removal:

1. As root, enter the following command:

   ```
   # rpm -e flash-plugin
   ```

2. Click Enter and follow the prompts.

Install the Flex SDK

1. Download Flex SDK ZIP file from the Adobe website or the Adobe Open Source site.
2. Create a directory to contain Flex SDK.
3. Extract the Flex SDK ZIP file to this directory. The Flex SDK contains the following directories:
   - /ant — Contains Flex Ant Tasks.
   - /asdoc — Contains helper files for the ASDoc tool that creates HTML documentation from your MXML and ActionScript source code.
   - /bin — Contains the mxmlc, compc, asdoc, and fdb utilities. The bin directory also contains the jvm.config file, which specifies Java settings that you can modify, if necessary.
   - /frameworks — Contains compiled framework classes, configuration files, and framework source code.
   - /lib — Contains JAR files used by the utilities.
   - /runtimes — Contains installers for the Adobe AIR runtime inside the air directory and installers for debug versions of Flash Player 9 inside the player directory.
   - /samples — Contains sample applications.
   - /templates — Contains HTML templates for Flash Player detection and browser integration and inside the air folder, a sample Adobe AIR application.xml file.
4. Ensure that the Java Runtime Environment (JRE) is installed on the computer and that the *java_home*/bin directory is defined in the system path. JRE 1.5,or 1.6 is required.
5. Install the appropriate debug Flash Player from the

   install_root/runtimes/player/*platform* directory.

6. (Optional) When the Flash Player installation finishes, restart your computer to ensure that the updated Flash Player browser plug-in is enabled.

Compatibility Issues

There have been some changes made in Flex SDK "Hero" that alter certain behaviors compared to Flex 4. To get the full list of issues, please refer to the Flex Hero Backwards Compatibility document. You cannot yet preserve Flex 4 behavior through the --compatibility-version flag in the Flex SDK "Hero" compiler. That work will be enabled later in the "Hero" development cycle.

Using the Data Visualization Components with Flex SDK

As of Flex 4, the data visualization components are part of the SDK distribution.

Using Automated Testing with Flex SDK

As of Flex 4, automated testing support is part of the SDK distribution.

Known issues

This section contains selected known issues. For a complete list of Flex issues and their status, see the public bugbase. The public bugbase lets you search for known issues, comment on them and add new bugs.

Tip: Use Filters to customize your search.

Desktop Features

- For information on the new Spark DataGrid, Form, Image, Formatters, RSL Enhancement or Size Report features, please reference their specifications on the Hero open source site.
- Spark DataGrid is under development and included with basic functionality. In its current form, Spark DataGrid can display data from an IList dataProvider, supports single and multiple selection of both row and cells, skinning, custom itemRenderers, programmatic sizing, and text styles. Please note that the following functionality is not currently supported: column sorting, column resizing via the mouse, and editability. Future releases of the Flex SDK will contain additional functionality. In addition, Spark DataGrid has not yet been fully optimized for performance. Future releases of the Flex SDK will contain additional functionality and optimizations. For a full list of Spark DataGrid bugs, go here.
- SDK-27729 [FTETextField] text not centered relative to baseline.
- SDK-23567 - [TLFTextField] mx:Text not correctly rendering HTML content when using

UITLFTextField.

- SDK-27846 - Bad type coercion error on incremental compiles.
- SDK-27917 - Modules and sub-application are loading their RSLs into the top-level application domain, but they should be loading their RSLs into their own application domain.

Mobile Features

- For information on the new MobileApplication and associated UI components (e.g. ViewNavigator, View, and ActionBar), please refer to the documentation posted with the build.
- Only Spark components are supported by mobile projects. MX components are not supported, except for charts, which should be usable (but have not been performance-tuned). To use charts in a mobile project, you will need to manually add datavisualization.swc and mx.swc to your library path.
- We recommend using the following components in mobile applications in this prerelease:
    - Only Button, List, CheckBox, RadioButton, TextInput, and TextArea have mobile skins in this preview release. Other skinnable components do not yet have touch- and performance-optimized skins.
    - Spark List and Scroller now support touch interaction when the inputMode style is set to "touch". This is set by default for all lists and scrollers in mobile projects.
    - For images, both Spark Image and BitmapImage should be usable.
    - For text, use Spark Label, Text Area, or TextInput. Other text components are not currently supported in mobile projects.
- Scrolling performance has not yet been fully optimized.
- MXML skins and graphics are generally not recommended in mobile projects for performance reasons. If you want to create a custom skin for a component (e.g. a component that we haven't yet provided a custom skin for), we recommend creating an ActionScript skin that uses FXG for graphics. However, if you have a limited number of custom-skinned components in a given view, MXML skinning performance may be acceptable.
- MXML item renderers are not recommended in mobile projects. We recommend using the MobileIconItemRenderer provided with the SDK. This item renderer provides two separate text labels, an optional icon on the left, and an optional decorator on the right. If this doesn't meet your needs, you can subclass the base class, MobileItemRenderer, and add and lay out your components in ActionScript. See the TwitterTrends sample for an example of how to use the MobileIconItemRenderer in a list.
- When displaying text in an ActionScript-based item renderer, use the MobileTextField

class for best performance. See MobileItemRenderer or MobileIconItemRenderer for examples of how this is used.

- MXML graphics are not recommended. For best performance, use FXG graphics.
- SDK-27008: Mobile optimized skins and item renderers don't support mirroring
- The TextArea control does not support scrolling yet. See SDK-27485 for a temporary workaround.
- Working with on-screen keyboards on devices:
    - To open the on-screen keyboard, the user must tap on an editable text component or long press the Menu key. There is currently no API in AIR for programmatically opening the on-screen keyboard.
    - The user can use the Back button to dismiss the on-screen keyboard. You can set stage.focus = null to programmatically close the keyboard.
    - When the on-screen keyboard is open, any content underneath the keyboard or scrolled off the top of the screen is currently not accessible. Keep this in mind when authoring content, and be sure to test with on-screen keyboards.
    - These issues will be addressed before the final release of "Hero".