USING THE DISTRIBUTION SERVICE

Adobe®

Updated 18 September 2009

Adobe Systems Incorporated, 345 Park Avenue, San Jose, California 95110, USA.

Contents

Chapter 1: Distribution Manager for advertisers and publishers

Distribution Manager basics

The Flash Platform Services Distribution Manager lets application developers promote applications written for the Flash Platform by using a unique ad unit that is associated with viral sharing. Developers can also track the distribution and use of their applications across the Internet. The Distribution Manager allows web publishers a simple way to monetize applications written for the Flash Platform by hosting ads for other applications.

The Distribution Manager is a software-as-a-service application that communicates directly with the Flash Platform Services. The Distribution Manager offers features specific to different types of users:

Advertisers The Distribution Manager is a self-service desktop application through which advertisers can create, manage, and track application advertising campaigns. Application campaigns get advertising in front of users of social network sites, the types of places most Internet users visit multiple times each day.

Advertisers set a specific daily budget in order to receive a specific number of application installs, targeted at users of social media web sites and applications. The daily budget, a maximum "spend" per day, allows them to carefully manage their advertising budgets. Advertisers are charged for application distribution on a per-install basis. All ad impressions are free. Ads distributed through the ad network reach more than 250 million unique users each month.

Advertisers use the Campaigns and Track views of Distribution Manager for most of their tasks.

Publishers The Distribution Manager is a self-service desktop application through which publishers can elect to host advertisements whenever their viral applications are copied. In return, they receive a share of the advertising revenue that is generated.

Publishers participate in the ad network by allowing an application's share menu to also promote a branded application. This does not limit existing advertising real estate on a publisher's site or on the application itself. Publishers earn revenue each time a user installs a branded application, either from the site or from the share menu of one of the applications.

Publishers use the Manage and Track view of Distribution Manager for most of their tasks.

Terminology

To learn how to use the Flash Platform Distribution Service, you need to understand some basic terminology.

Partner ID An ID issued by Gigya that identifies a specific account on the Gigya ad network where applications, campaigns and analytics are stored. A Partner ID is automatically linked with your main Adobe account upon first use of the Distribution Manager. It is possible to have more than one Partner ID, and it is possible for each user of the Distribution Manager to manage more than one Partner ID. For more information, see "Group management of accounts" on page 2. You can go to http://gigya.com and click Account Settings to display your Partner ID, or view it in the Distribution Manager.

Code ID The Code ID is compiled into the application to identify that particular binary asset or creative. It can optionally be paired with a Config ID. If you want to reuse the same basic application to display different content, you use the same Code ID but add a different Config ID.

Config ID The Config ID identifies any custom content programming injected into the application using FlashVars. The Config ID is passed into the application through the HTML embed code. The Config ID is used to identify the embed code itself (which could contain FlashVars that determine the specific content displayed by the application). The Config ID is used in conjunction with the Code ID to uniquely identify a particular variant of an application. For example, if your Code ID is "Sample App" and your Config ID is "US English version", your application is identified as "Sample App / US English version". Different variants of the Sample App can be programmed using FlashVars, such as "Sample App / ES Spanish version".

(the) creative Some advertisers and publishers use the terms "creative" or "the creative" to describe applications. These terms mean the same thing: the application that is created using the Distribution Service Flash and Flex Builder extensions, shared using the Distribution Service Dreamweaver extension, and tracked, monetized and managed using the Distribution Manager.

SWF Flash or Flex developers may use "SWF" or "the SWF" to mean the same thing that advertisers and publishers call "the creative." These terms all refer to the application that is created using the Distribution Service Flash and Flex Builder extensions, shared using the Distribution Service Dreamweaver extension, and tracked, monetized and managed using the Distribution Manager.

widget The term widget refers to small applications that are limited in scope and are used primarily in social media web sites. Second-generation applications are richer, offer more services, and allow you to send them to your mobile phone or anywhere on the Internet that you choose.

Existing Gigya Customers

If you are an existing Gigya customer who wants to begin using Adobe's Distribution Manager to manage your applications and campaigns, you need to enable your existing Gigya account to be accessible from the Distribution Manager. You must have an Adobe ID that is linked to your pre-existing Partner ID on the Gigya network.

To convert your existing Gigya account to an Adobe account, do the following.

1 If you don't have an Adobe ID, create a new Adobe account by pointing a Web browser to https://www.adobe.com/cfusion/membership/index.cfm and clicking Create an Adobe Account. Follow the steps to create an Adobe account using your e-mail address and a password of your choice.

2 Send an e-mail to fps-distribution@adobe.com requesting that your Gigya account be converted to an Adobe account. Include the following in the body of the e-mail:

- Your Adobe ID. Typically, this is the e-mail address you used to register your Adobe account.
- Your Partner ID. To locate your Partner ID log in to http://gigya.com and click Account Settings.

Your account will be converted and you'll be able to log in to the Distribution Manager within 24 hours. When you log in, you'll be able to manage all of the applications previously associated with your Gigya account.

Group management of accounts

Advertisers and publishers often have a need for multiple people to access the same Adobe account so that applications and campaigns can be viewed and managed by several people, including team members and vendors. A single Adobe account can be shared by multiple Adobe account IDs that are all linked together with a common Partner ID.

If you are an account holder it is possible to allow other account holders to access your applications, campaigns, and analytics data. All accounts must be linked together by Adobe Customer Support to share a common Partner ID.

Enabling others to access your team's shared account To enable others to access a shared account, send an e-mail to Adobe Customer Support at fps-distribution@adobe.com and indicate your Adobe account ID and the Partner ID to be shared as well as the additional Adobe account IDs that you'd like to have access to your account.

Using a single Adobe ID to access multiple accounts It is also possible for a single Adobe ID to be associated with multiple accounts. This can occur when an advertising agency employee needs access to the accounts of multiple customers, for example. Logging into the Distribution Manager will display the current Partner ID in a drop down box on the bottom right of the application. If the current Adobe ID has been linked to multiple accounts, clicking the drop down box will list all Partner IDs associated with the account. Selecting a different Partner ID will update Distribution Manager to manage the selected Partner ID. To request this, send an e-mail to Adobe Customer Support at fps-distribution@adobe.com and indicate your Adobe account ID as well as the additional partner IDs that you'd like to grant access to your account.

For both types of linkages described above, as long as you send the e-mail from the e-mail address registered with your account, the additional accounts will be linked within 24 hours and an Adobe Customer Care agent sends a response e-mail indicating the change has been made.

If you send an email from a different email address, the Customer Care agent will contact you to verify your identity and ensure that you are in fact the account holder.

Getting started with the Distribution Manager

Installing the Distribution Manager

To install the Distribution Manager on your local system, log into the Adobe web site at http://www.adobe.com/go/fps and follow the instructions to download and install the Distribution Manager. If your IT organization manages the distribution and installation of applications, you will need to ask for their assistance with installing the Distribution Manager and must be applied when it becomes available to continue use of Distribution Manager.

Once installed, the Distribution Manager checks for updates automatically and prompts you to upgrade when new versions become available. There two types of updates: optional and mandatory. Optional updates add new features and fix bugs, and can be applied at any time. Mandatory updates address API changes that cause the Distribution Manager to stop working until you apply the update.

Signing in

Although the Distribution Manager prompts you to sign when it starts, you can click the link Continue without signing in to skip sign-in. This takes you to the Create tab. Otherwise, Distribution Manager prompts you to sign in when you try to use a feature that requires it.

To sign in to Distribution Manager:

1 Click Sign In to the upper right of the Distribution Manager.

2 Type your Adobe ID e-mail address and password into the fields provided.

When you sign into the Distribution Manager for the first time, a Partner ID is created for you automatically and is displayed in the lower-right corner of the application window. For most users, this is the Partner ID you would provide to your developers so they can encode it into the applications. Until applications are activated on the network using your Partner ID, nothing is displayed in the Distribution Manager's main window, the Manage view.

Get an Adobe ID

The Distribution Manager requires you to create an Adobe ID and authenticate when logging on to keep your information private. If you already have an Adobe ID, you can use it instead of creating a new one for the Distribution Manager. However, you may want to create a separate Adobe ID for use with the Distribution Manager, especially if your original Adobe ID was created using a personal e-mail address and you intend to use the Distribution Manager for work.

For information about getting an Adobe ID, see Adobe ID and Membership. To open an Adobe account, click here.

The Distribution Manager workspace

Five links appear across the horizontal top-level navigation bar of Distribution Manager: Manage, Campaigns, Track, Account, and Create. Each of these links opens a view of the Distribution Manager. The workspace, including selections in the vertical folder pane to the left, the tabs that sometimes appear under the top-level navigation bar, and the buttons near the bottom of the workspace, changes according to the view selected.

The buttons near the bottom of the workspace add functionality to an item selected in the pane to the left. For example, you might click the Manage tab to open the Manage view, select an application to register, and click the Register button near the bottom of the workspace to register the application.

Manage You use this view to manage your applications. You can designate applications as active or inactive, view their registration and monetization status, and view cross-promotion information. An active application means that the Distribution Manager retrieves information for it when the Distribution Manager launches. An inactive application's information is retrieved when the user views the data. The more active applications you have, the longer it can take for the Distribution Manager to launch.

Campaigns You use this view to create, schedule and manage your campaigns.

Track You use this view to track and view reports on unique users, revenue, individual applications, and application campaigns.

Account You use this view to set up and manage your account information, including how you pay for advertising and how you receive revenue payments.

Create You use the tabs in this view to make applications shareable and add Share menus to applications and websites.uttons and Bookmark buttons to a web site. You can design a Share menu's theme, configure its behavior, and generate embed code in this view. This functionality is also available in the Distribution Service extensions for Flash and Flex. This view is used primarily by application developers and web site publishers. See "The Create View" on page 46 for more information.

Establishing an account

To make full use of the Distribution Manager, you must have an Adobe ID from an Adobe account, and a linked Partner ID that the Distribution Manager can generate for you. A Partner ID is automatically created when you first launch Distribution Manager. You can elect to contact Adobe to use an existing Partner ID instead, or to have multiple Partner IDs associated with your Adobe ID.

About Partner IDs

A Partner ID is a permanent, unique identifier that customers receive when they begin using the Distribution Manager. Advertisers must have a Partner ID to establish a line of credit to pay for their advertising. Publishers must have a Partner ID to designate PayPal accounts, and register their PayPal accounts to get paid.

Advertisers who already hold a Partner ID and want to use the Distribution Manager to manage Flash applications and campaigns must link their Partner ID with an Adobe ID. See "Existing Gigya Customers" on page 2 for information.

Note: *Publishers and advertisers set up pay accounts and credit accounts with Gigya, not with Adobe. Even though Adobe is providing these services, your financial account is with Gigya. For more information, see the Services Agreement.*

Managing your accounts

You can view account information in the Accounts view of Distribution Manager. The Accounts view provides advertisers and publishers with summary and detailed financial information regarding their account status. It also provides convenient links to the appropriate information regarding the establishment of an account.

The Campaigns and Account tabs have a Balance Summary in the lower left corner. The three fields are Account Balance, Available Credit, and Buying Power. Buying Power is the sum of the other two. Available Credit will not appear unless you have already set up a credit account with Gigya. When the Buying Power becomes less than the cost of active campaigns, it turns red to indicate an insufficient balance. If you have an insufficient balance, a low balance alert window will pop up. This alert can be dismissed or turned off by clicking a check box on the alert that says Hide This Until I Restart The Application.

Registering applications

Before an application can participate in the Flash Platform Services advertising network in any way, it must first be registered. When you register an application, you submit its embed code, and the version of ActionScript that was used to create it, to the ad network along with tagging information. Tagging information tells which kinds of content your application contains. For example, an application containing information about travel destinations would be tagged Travel, and if it allows users to upload photos which are displayed through the application it would also be tagged User Generated Content. Before an application can participate in the ad network, it must be registered, it must be properly tagged, and the tags you assign to the application must be verified. Tagging allows Adobe to ensure that inappropriate content does not find its way into the ad network.

What you need to get from your application developer

Advertisers and publishers need to receive the following information from the developer who created your applications:

- The application SWF file
- HTML embed code
- Version of ActionScript that the application uses
- Image to be used as a thumbnail to identify the application

If you have more than one application, you need separate information (both embed code and ActionScript version) for each one. See “Creating and publishing shareable applications” on page 18 for information about how HTML embed code is generated and how ActionScript versions are specified.

The publisher, who puts the application somewhere on a web site, needs to provide the developer a couple things as well:

- the URL where the application will reside on the Internet
- the URL where a pixel-perfect screenshot of the application, in its default state, resides. The screenshot must be a JPG file.

If the publisher wants to create specific versions of the application for any social network, he or she must also provide the URLs for the locations where these special versions will reside.

Tagging applications

During the application registration process, you are prompted to categorize (or *tag*) your application, by checking boxes next to a wide variety of content types. Select the tags that most accurately describe the content of your application. The selections you make are reviewed prior to your application being placed as an ad, and prior to it being made available to host advertising. See "Application approval process" on page 12 for more information. Some content types (for example tobacco, alcohol, gambling and dating) may have restrictions with respect to how they can be used. Consult the Services Agreement to learn more about these restrictions.

Using folders to manage applications

In the Manage view and Campaigns view, the left-side vertical pane shows a list of folders. You can use the folders in the Manage view to organize applications, and you can use the folders in the Campaigns view to organize campaigns.

As your library of applications grows, you may find it convenient to organize your applications into folders. Some folders are user-created and static; others are dynamic Smart Folders, in which the applications and campaigns are organized according to pre-determined criteria.

In the Manage view, the following smart folders appear in the Active Applications list in the left-side vertical pane:

Registered Applications When this smart folder is selected, Distribution Manager shows only those applications that you have registered.

Unregistered Applications When this smart folder is selected, Distribution Manager shows only those applications that you have not registered.

Hosting Cross Promotions When this smart folder is selected, Distribution Manager shows only those applications that are hosting cross promotions.

In the Manage view, the following folder appears in the left-side vertical pane:

Archived Applications Drag applications here that you may not wish to view on a daily basis.

In the Campaigns view, the following smart folders appear in the My Campaigns list in the left-side vertical pane:

Running Now When this smart folder is selected, Distribution Manager shows only those campaigns with Start Dates earlier, and End Dates later, than the current date.

Future When this smart folder is selected, Distribution Manager shows only those campaigns with Start Dates later than the current date.

Past When this smart folder is selected, Distribution Manager shows only those campaigns with End Dates earlier than the current date.

Create a new folder

In the Manage view or the Campaigns view, you can create one or more new folders for organizing applications or campaigns.

1 In the top-level navigation bar, select Manage or Campaigns.

2 Do one of the following:

- Select the My Applications (My Campaigns) folder or the Archived Applications (Archived Campaigns) folder.
- To create a subfolder within another folder, select the folder that should contain the subfolder.

3 In the left-side vertical pane, click Add Folder.

4 Click once to select the new folder. Click again to edit the name. Type a new name, then click out of the text box.

Delete a folder

You can only delete user-created folders. To delete a folder click on the X that appears next to the folder when you move the mouse over it. This moves the contents of the folders to the root folder (My Applications/Campaigns or Archived Applications/Campaigns).

Previewing applications

After you register an application, you can view it in the Distribution Manager's Manage View in one of the following ways:

- Select an application and click View.
- Double-click an application.

Editing registered applications

Once an application has been registered, the registration information can be edited. As a precautionary device for advertisers and publishers, any changes to the tagging information will cause the application to be re-reviewed by Gigya. The application is removed from participation in the ad network until the tagging information is verified.

Changing the tags will place all campaigns that use the application into a Pending state, which will temporarily stop the running of the campaign. Changing tags will also prevent the application from hosting any ads from the ad network until the new tagging information has been verified.

Edit the registration of an application

1. Select Manage.
2. In the navigation pane, browse to an application folder.
3. In the pane on the right, select an application that has been registered.
4. Click Edit. The Distribution Manager opens the Edit Application Registration dialog box.
5. Select Identity, and select the desired version of ActionScript.
6. Click Browse, browse to the desired image, and click OK.
7. Do any of the following, in any order. None of these steps is required.

- Select Tags, and select the tags that describe the application.
- Select Source. Type in a SWF file URL and set any FlashVars.
- Select Advanced. To add a FlashVar to the list of FlashVars, click + next to FlashVars and select a parameter from the list.
- To remove a Flash Var from the list of FlashVars, select the parameter from the list of FlashVars, and click -. Click Save.
- Select Advanced. To add a URL parameter to the list of URL parameters, click + next to Advanced, and enter a new parameter name and value.
- To remove a URL parameter from the list of URL parameters, select the parameter from the list of URL parameters, and click -.

Campaigns

Using the Campaigns view, advertisers can create, modify, pause or resume application campaigns, target and promote campaigns.

Only registered applications can be used in campaigns. Typically, publishers register applications so that advertisers can use them in campaigns. For more information about registering applications, see "Registering applications" on page 5.

The ad network does not place ads in applications that contain certain types of content.

Advertisers can choose whether to have their applications placed on social networking websites, or those that host user-generated content.

Any registered application can be used in multiple advertising campaigns. For example, advertisers may wish to create one campaign with one set of targeting information and compare that to a second campaign for the same application but with different targeting information. Since advertisers pay only for copies of applications made from the advertising campaign (and not for impressions), there is no financial impact to this kind of A/B testing.

More Help topics

Creating Campaigns

"Establishing an account" on page 4

Establish a credit account to pay for campaigns

Advertisers must establish a line of credit to pay for their advertising. Publishers must have PayPal accounts, and register their PayPal accounts to get paid.

Only advertisers must apply for credit terms. Advertisers are charged only for applications installed by members of the ad network, not for viral installations. As part of their account setup, advertisers set a daily budget for each campaign. When that budget is spent on a given day, the ad network stops offering the advertiser's applications for that day.

Only publishers must submit a W-9, and they must do so before they can be paid. Publishers get paid through PayPal and thus must supply a PayPal account.

Publishers and advertisers set up pay accounts and credit accounts with Gigya, not with Adobe. Even though Adobe is providing these services, your financial account is with Gigya. For more information, see the Services Agreement.

To set up a credit or PayPal account, open the Distribution Manager and click the Accounts tab. Click the links under How I Pay for Advertising (for advertisers) or How I Receive Payments (for Publishers) for information about how to set up accounts.

Creating campaigns

There are two ways to create a campaign. You can select a registered Flash application from the Manage view and select the New Campaign button; or you can click the New button in the Campaigns view and then select an application to advertise. You can create a campaign for any registered application. If your application is not yet registered, see "Registering applications" on page 5.

Create a campaign from the Manage view

1 In the Manage view, select a registered application.

2 Click New Campaign. The New Campaign dialog opens with the Campaign Details pane showing.

3 Enter the detailed information for your campaign.

Create a campaign from the New Campaign wizard

1 In the Campaigns view, click New. The Distribution Manager opens a New Campaign Wizard.

2 Follow the steps given in the wizard.

Previewing an example advertisement

You can preview an ad by clicking the Preview button on the Campaign dialog. The exact size and shape of the ad vary based on the dimensions of the host application.

Campaign approval

To ensure that content is appropriately targeted and displayed within the advertising network, Adobe has implemented a campaign approval process to ensure that the Ad Title, Ad Text and Ad Image appropriately reflects the application or content that you would like to promote within the ad network.

A campaign will not run until the application with which it is associated has had its tagging information verified. Once that has happened, every campaign created for that application automatically enters a verification process. The status of a campaign reflects the categorization approval status of its associated application. If a proposed categorization of an application is denied, campaigns including that application also will be denied. If the application has the pending status, all campaigns associated with it will have the Pending Review status.

After creating a campaign or editing the values for an existing campaign associated with an approved application, the Registration Status for your campaign will be set to Pending Review while your campaign is reviewed by the ad network. Once reviewed your campaign Registration Status will automatically transition to another status.

The registration status of a campaign can be viewed in the Registration Status column of the Campaigns view. The registration status can be one of the following:

Scheduled Your campaign has been approved and is scheduled to run beginning on the Start Date.

Active Your campaign has been approved and has begun.

Paused Your campaign has been approved, but it currently paused at your request.

Insufficient Funds Your campaign cannot run because your account does not have enough Buying Power to pay for the campaign.

Completed Your campaign end date has been passed.

Error There is an unknown problem and you should contact Adobe Customer Support.

Declined Either the tags given the campaign’s application are not appropriate, or the campaign’s associated application is tagged with a category inappropriate for the ad network, or the campaign's ad title, ad test, or ad image was deemed inappropriate.

***Note:** You can re-tag a campaign or application and resubmit it for review.*

Pending Review The campaign has this status when the application with which it is associated has the pending status or the associated application is approved and the campaign is in the process of being approved.

Managing campaigns

When you select Campaigns, the navigation pane to the left of the Distribution Manager contains an Add Folder button, and two top-level folders, by default: My Campaigns and Archived Campaigns. You can create custom campaigns and add them to either of these top-level folders.

Newly created campaigns are listed in the My Campaigns folder. Old or obsolete campaigns may be moved to the Archived Campaigns folder.

My Campaigns contains three smart folders by default: Running Now, Future, and Past. You cannot create smart folders, and you cannot add folders to any of these smart folders. The smart folders show sets of your campaigns, sorted according to whether they are current, future, or past.

Distribution Manager alerts advertisers any time the total cost of their active and scheduled campaigns exceeds the available credit in their account.

In the Campaigns view, applications are listed in the main viewing area. The following columns show details for each application in the list:

Registration Status The status given the campaign through the campaign approval process. For more information about Registration Status, see "Campaign approval" on page 9.

Ad Image Thumbnail of the image used in the ad.

Campaign Name The name given the campaign, typically by the advertiser.

Daily Budget The maximum amount the advertiser will spend per day for installations of the application.

Total Spend The total spent by the advertiser for installations of this application to date.

Installs The current number of installations for the application.

Cost/Install The price the advertiser pays for each installation of the application.

Start The start date for the campaign.

End The end date for the campaign.

Organizing campaigns with folders

You can use folders to group related campaigns, and to retire campaigns which are no longer active.

Organize active and inactive campaigns with folders Create new folders according to campaign type, target audience, or any categories desired. For information about creating new folders, see "Using folders to manage applications" on page 6. After folders are created, you can drag campaigns into the appropriate folders.

Retire inactive campaigns Although the End Date for a campaign may have passed, Distribution Manager continues to automatically track data from the campaign as long as the campaign remains in the My Campaigns list.

Browse to the campaign you want to retire in the My Campaigns list, or in one of its subfolders.Drag the campaign to the Inactive Campaigns folder.

Modifying campaigns

Once you have created a campaign, you can modify it whenever you want to. You can change how it's targeted, pause or resume it, or change other options as needed. The general steps to modify a campaign are as follows.

1. In the Campaigns view, select an existing campaign.
2. Click Edit. The Distribution Manager opens the Edit Campaign dialog box.
3. Modify campaign options as desired.
4. Click Save.

Pause a campaign

Pausing a campaign will remove the application from all placements.

In the Campaigns view, select an active campaign and click Pause. The Distribution Manager pauses the campaign, and the label on the Pause button changes to Resume.

Resume a campaign

In the Campaigns view, select a paused campaign and click Resume. The Distribution Manager resumes the campaign, and the label on the Resume button changes to Pause.

Target a campaign

You can control the targeting of your campaign across various geographies.

1 Click Campaigns to open the Campaigns view.
2 Select a campaign, and click Edit. The Distribution Manager opens the Edit Campaign wizard.
3 Click Next twice to open the Define Targeting step of the wizard.
4 Select desired countries to target.
5 (Optional) If you selected the United States, you can also select specific states.

***Note:** In some cases certain age and gender demographics can also be targeted. When the Distribution Manager can determine those attributes, the targeting information will be used. However, not all destinations support this type of targeting.*

Tracking campaigns

Advertisers and publishers can track the effectiveness and reach of application campaigns using the Track view of the Distribution Manager. To see the reports that are available for campaigns, click Track at the top of the Distribution Manager and then click Campaigns in the left panel. For information about the individual reports available, see “Tracking campaigns with reports” on page 13.

Monetizing Applications

About hosting cross-promotions to generate revenue

Publishers can earn money with, or *monetize*, their registered applications by allowing cross-promotion advertising to appear when users copy their applications.

Cross-promotion works by giving the consumer the option to install your application when they are installing another application. You receive a share of the revenue each time a consumer successfully installs an advertised application and the advertiser is charged. This applies not only to installs initiated from your web site, but every time an application is shared.

Opting in to host cross-promotions for an application

Follow these steps to host a cross-promotion.

1 Click Manage in the top-level navigation bar.
2 Under My Applications, select Registered Applications, and select an application to monetize.
3 Click Monetize. Distribution Manager opens the Monetize dialog box.
4 Check Enable Cross Promotions.
5 Check every category of advertising that would be appropriately paired with the selected application. The more categories you select, the more opportunities you will have for generating revenue from hosting cross-promotions.

***Note:** There are restrictions and limitations regarding the appropriateness of ad content around the world. Certain categories may be refused in certain locales.*

6 Click Submit.

In the Manage view, Distribution Manager changes the Monetization Status of the application to Pending Review. After the application has been reviewed, the status will change either to Hosting Cross Promotions, or to Declined. An application is declined when it violates the Services Agreement for participating in the ad network.

Select categories of advertisements to host

During the application registration process, application developers and publisher are prompted to categorize their applications, by checking boxes next to a wide variety of content types.

When publishers opt in to host advertisements in their applications, they select the categories that seem most closely matched to the interests of visitors to their sites. They can change these categories later if desired to achieve a better fit.

Application approval process

To participate in the ad network, developers and publishers must declare what type of content their application contains. See "Tagging applications" on page 6 for more information about tags and how they are applied.

You can participate in the Distribution Service network in two ways:

- As an advertiser by buying distribution through campaigns
- As a publisher by hosting cross promotions

When an application is submitted to either host cross promotions (by pressing the Monetize button) or a campaign is built around an application, the application enters an approval queue. A Content Manager then reviews the application and the tags to make sure they match. The content manager reviews applications within one business day. Any application tagged with one of the restricted tags (Alcohol, Adult Content, Tobacco, etc.) is forbidden from participating in the ad network (either hosting cross- promotions or purchasing campaigns).

Tracking and analytics

The Distribution Manager provides four types of reports for tracking and analysis: Unique User Reports, Application Reports, Revenue Reports, and Campaign Reports. The available reports are as follows.

- "Application reports" on page 14 (per application)
  - Views (by destination and country)
  - Menu Views (by destination and country)
  - Interactions
  - Installs (by destination and country)
  - Conversions (by destination and country)
  - Stickiness (by destination and country)
  - Emails sent (by destination)
  - Viral grabs (by destination)
  - Custom events

- "Campaign reports" on page 15 (per campaign)
  - Campaign Summary
  - Paid Views (by destination and country)
  - Paid Installs (by destination and country)
  - Conversion Rate (by destination and country)
- "Unique users" on page 15 (per Partner ID)
- "Revenue reports" on page 15 (per Partner ID)

Tracking campaigns with reports

You can read reports on the various metrics of your application campaigns in the Track view of the Distribution Manager. Both advertisers and publishers use these reports to assess the effectiveness of their application campaigns, and to retarget and adjust them over time.

More Help topics

Tracking Applications

Picking date ranges

You can choose a range of dates for which to generate reports in the Distribution manager.

1. Click Track in the top-level horizontal navigation bar.
2. In the left-side vertical navigation pane, click Campaigns.
3. To set the time frame for the report, click one of the following next to the word Last:
   - 1D to set the time frame for the last day
   - 1W to set the time frame for the last week
   - 1M to set the time frame for the last month
   - 3M to set the time frame for the last three months
   - 6M to set the time frame for the last six months

   ***Note:** The longer the time range, the longer it takes to generate the report.*
4. To specify your own time frame, click Custom and then click the calendar icons next to the From and To fields to browse to the start and end dates of the time frame desired. Click the Go button to retrieve the report.

Refreshing reports

You can refresh the data for any report you are viewing by clicking the Refresh button while the report's data is displayed.

Including and excluding social network destinations and countries

You can filter the Paid Views, Paid Installs, and Conversion Rate reports by application destinations: the social networking sites on which applications are installed. You can also filter these reports by country. To filter by social network destination, select the Destination tab, and scroll down to the list of social networks. Check the box next to each social network you want to include in the report.

To filter by the countries in which applications are installed, select the Country tab, and scroll down to the list of countries. Check the box next to each country you want to include in the report.

Comparing one measure to another

In most reports, it is possible to see two different measures on one screen for comparison purposes. For example, on the Views report, you can select Installs as a comparison measure to see how the data for Views and Installs relates to one another over time.

Switching Report Types

You can switch the report type to view the data as a bar chart or a line chart. To change the view of a report's data, click the Type drop-down box and select a report type.

Exporting to CSV (Excel)

On each report, there is a button at the bottom that says "Export To Excel". When you click it, you are prompted to save a copy of a CSV (comma separated values) file somewhere on your computer. You can open this CSV file in any spreadsheet or reporting program to further manipulate or slice the data.

Exporting to PDF

In any report, click the button labelled "Export To PDF" to save a copy of the report as a PDF file on your computer.

Exporting charts as images

You can export any of the charts as an image file, which allows you to include the chart in another document or presentation. Click Export As Image at the upper right of the desired chart to export its data as an image.

Switching between applications and campaigns

You can use the left side navigation to switch to viewing reports for any application or campaign. Click the report name on the left to display its data.

Application reports

Advertisers and publishers can track the performance of individual applications with the application reports in the Distribution Manager. They can also track campaigns. See "Campaign reports" on page 15 for information about tracking campaigns.

To view application reports in the Distribution Manager, open the Manage view and select an application. Then click Track, and the Distribution Manager opens the Track view for that application.

You can also open the Track view and select an application, and then select a type of report to view.

Types of application reports

Each time an application in the network is shared, the analytics of the application are updated. Most of the analytics are available by default and do not require any special programming. The analytics categories available are:

Views the number of times an application has been seen.

Menu Views the number of Share menu loads. This corresponds to the number of times users clicked on "grab this" or a comparable link to launch the Share menu. The summary box shows the percentage of views where a user opened the Share menu.

Installs the number of applications installed per day.

Conversions the percentage of Share menu loads that result in a new install. This number reflects the conversion from clicking "Grab Application" to actually completing the installation process

Stickiness the average number of views an application receives before being removed by the user. This is calculated as cumulative views divided by cumulative installs.

Emails sent the number of emails sent to recipients. This is calculated by counting each e-mail that is sent by an application user to a friend.

Viral grabs the total number of installs originating from a friend's page. These are second generation installs and beyond and do not include seeded installs.

Custom events the specific interactions that end users perform while using your application. A custom event implies an action defined by the developer that is specific to that application. Examples of custom events tracking include users watching a video, playing a game or simply mousing over certain regions of the application. Custom events are added to the application by a developer using ActionScript within Flash Professional or Flex Builder.

Campaign reports

Advertisers and publishers can track the performance of campaigns with the campaign reports in the Distribution Manager. You can also track the performance of individual campaigns. See "Application reports" on page 14 for information.

Campaign Summary The Campaign Summary report allows you to compare ad network activity, viral installs and views, and total activities for a selected campaign. In the Track view of the Distribution Manager, after specifying the time frame and selecting Campaign Summary, check the boxes next to the desired types of Installs and Views.

Paid Views The Paid Views report shows the number of times per day an application has had a paid view, or has been installed, through the ad network. You can filter this data by destination or country.

Paid Installs The Paid Installs report shows the number of times per day an application has been installed through the ad network. You can filter this data by destination or country.

Conversion Rate The Conversion Rate report calculates the percentage of views from the ad network which result in installations. You can filter this data by destination or country.

Navigate to the Campaign reports

1 Click Track in the top-level horizontal navigation bar.
2 In the left-side vertical navigation pane, click Campaigns.
3 Set the timeframe for the report (1 day, 1 week, 1 month, 3 months, 6 months, or custom).
4 From the list of campaigns in the left-side vertical navigation pane, select the name of a campaign for which you want reports.
5 From the list in the left-side vertical navigation pane, select the name of the type of report desired.

Distribution Manager displays the reports for the selected campaign.

Unique users

This report lists the number of unique users who have viewed all of your applications in the date range you specified. It breaks results down by destination.

Revenue reports

The Revenue Report shows a date ordered table displaying ads served, installs, revenue, and ECPM (effective cost per thousand impressions).

Chapter 2: Using Flash Professional to create shareable applications

Distribution Service extension for Adobe Flash CS4 Professional

About making applications shareable

The Distribution Service extension for Adobe® Flash® CS4 Professional provides Flash developers with tools to help site visitors share applications, to track the sharing activity of the visitors, and to earn income through the ad network. Distribution Service extension for Flash CS4 enables you to create an application from scratch that incorporates a Share menu, or to add a Share menu to an existing FLA or SWF file.

Sharable applications are rich media applications, typically designed in small formats for embedding in web pages, and for sharing among users on social networks.

When visitors click a button on a shareable application, the Share menu is displayed over the application. The Share menu lets visitors do any of the following:

- copy the embed code for posting the shareable application onto another page,
- post the shareable application directly on their blog or social networking site,
- bookmark the shareable application with a social bookmarking service such as Delicious, Digg, or Google Bookmarks,
- e-mail the shareable application to others.
- Download and install a desktop or mobile version of the application.

To use all the functions and tools of Distribution Service extension for Flash CS4, you must have a Partner ID. You can obtain a Partner ID with the Distribution Manager.

You can earn money by joining the ad network. If you choose to participate in the ad network, you could be paid to promote shareable applications from top-tier brand advertisers.

Install the Distribution Service extension for Flash CS4 Professional

1 Log into the Adobe web site at http://www.adobe.com/go/fps and follow the instructions to download and install the Distribution Service extension for Flash CS4 (MXP file) to your local hard disk.

2 Double-click the MXP file to install the extension.

Note: *For general information about installing Flash extensions, see Flash Help.*

Using the Distribution Service extension for Flash CS4 Professional

Once you have installed the Distribution Service extension for Flash Professional, you can get access to the extension by following these steps:

1. Open Flash Professional.
2. Open the FLA file that contains your shareable application's content, or create a new file using the Share template (File->New->Templates tab).
3. Click Window->Other Panels->Share to display the Share panel.

   Note: *Although the Distribution Service extension is an Adobe extension, and is installed using the Flash extensions menu and processes, the Share panel is not displayed under Window->Extensions. Rather, as described above, you access it through the Other Panels menu.*

Sharing applications with mobile devices

In addition to making applications shareable across social networks, bookmarking services, and desktop runtime environments, applications are shareable with mobile devices. You can share applications with mobile devices either as Flash Lite® 3.1 content or as native application content that is specific to a phone platform.

Adobe offers the Distribution service to mobile devices as a free, Beta service and reserves the right to change the pricing, availability and other aspects during and after the Beta period. Initially, the configuration to enable mobile sharing is available only by using the Distribution Service extension for Adobe Flash CS4 Professional and it is not available for applications created in Flex Builder or Dreamweaver CS4.

When sharing an application with a mobile device, the advertiser or publisher will almost always have one or more mobile-optimized versions of the application that is separate from the web version. All variants of the application binary, however, will be shareable by way of the Share menu and should offer a similar experience to the end user. Depending on the destination that the end user chooses, whether it is social network, bookmark service, desktop environment, or mobile device) it's possible to have a different variant of the same application that is optimized for each destination. In the case of mobile devices, it is the rule rather than the exception.

To enable your application to be shared with mobile devices, you must complete these steps:

1. Configure the Share menu in the web version of your application to support sharing to mobile devices.
2. (Optional) Create a variant of your application for Flash Lite® 3.1, and then package it into a native installer as a .CAB file for the Windows Mobile platform.
3. (Optional) Create native variants of your application for the Apple iPhone, Nokia S60 and other device platforms.
4. Deploy your application to the web and optionally to the various mobile application stores so users can discover and share them.

The Mobile sharing user experience

The user's experience of sharing an application with a mobile device is similar to that of sharing an application with a social network. The process consists of the following steps:

1. A user discovers your application on the web and clicks a button to open the Share menu.
2. The Share menu presents a tab named Mobile for sharing the application to a mobile device.
3. After clicking the Mobile tab, the user is asked to provide his or her telephone number and pick the country where his phone is registered.
4. Almost immediately, the user's phone receives a text message containing a URL to click to install the application.

5 When the user clicks the link, the installation process begins and installs the Flash Lite 3.1 Distributable Player, if necessary. If the publisher has not created a version of the application that's suitable for the user's phone, a web page that has been chosen by the publisher displays.

The contents of the text message is one of two formats, depending on which version can fit in a single text message. The messages are:

- Your application name is available to download: URL to download. Please ignore if this was not requested.
- Your application is available to download: URL to download. Please ignore if this was not requested.

Where application name is the name of your application set in the Share panel configuration and URL to download is a short URL generated by the Distribution service. The generated URl is available for six hours after the initial text message is sent.

Creating and publishing shareable applications

Distribution Service extension for Flash CS4 enables users of Flash CS3 or Flash CS4 to create and publish shareable applications. The following steps outline the general workflow for creating shareable applications using Flash Professional.

Workflow for creating and publishing shareable applications

To create and publish shareable applications with Adobe Flash, follow the following general steps:

1 Create a shareable application from a template. See "Create a shareable application from a template" on page 19.

2 Select the Share menu component on the Stage, then open the Share panel to configure it.

3 Customize the Share menu theme using the options in the Color, Layout, Destinations, and Advanced tabs of the Share menu panel.

4 Set up the application configuration by filling in the forms for Basic Config. You can further customize the Share menu by entering data for Optional Config using the Sharing, Mobile, FlashVars, URL Parameters and Advanced tabs.

   The following fields are mandatory:

   - Partner ID - The Partner ID you want to associate with this application.
   - Code ID - The identifier or name you'd like to give this application for tracking analytics.
   - Config ID (optional) - The name you'd like to identify a particular configuration or variant of this application when programming content with FlashVars.
   - Script Version - The ActionScript Version used in your project.
   - Title - The name that end users will see when sharing your application.
   - SWF URL - The URL that points to the location where your SWF will be hosted on the Internet.
   - Preview Image URL - A URL to an image file that shows a preview of your application. This is used to display a reference to your application in certain social networks.

5 On the Embed Code tab, the Share panel automatically generates an HTML embed code that you can paste into a web page where you want to place your application. You can edit this generated embed code directly; any edits are saved within the selected instance of the Share menu component.

6 Upload your HTML document and SWF file to the server.

Create a shareable application from a template

You can use a template to build an application from scratch that incorporates sharing features. When visitors click a button on the shareable application, a Share menu is displayed over the shareable application.

1 Select File > New.

2 In the New From Template dialog box, select the Templates tab.

3 Select an application type from the Category menu: Apps-Custom, Apps-Standard, Apps-Mobile.

4 Select a template from the Templates menu.

5 Click OK.

6 Select the Share menu component on the stage.

7 To open the Share panel, select Window > Other Panels > Share.

8 Set up the application configuration by filling in the forms for Basic Config. You can further customize the Share menu by entering data for Optional Config using the Sharing, Mobile, FlashVars, URL Parameters and Advanced tabs.

   The following fields are mandatory:

   - Partner ID - The Partner ID you want to associate with this application. For information about finding your Partner ID, see “About Partner IDs” on page 4.
   - Code ID - The identifier or name you'd like to give this application for tracking analytics (for example, myWeatherApp).
   - Config ID (optional) - The name you'd like to identify a particular configuration or variant of this application when programming content with FlashVars.
   - Script Version - The ActionScript Version used in your project.
   - Title - The name that end users will see when sharing your application.
   - Copyright - Information about your application’s copyright holder.
   - Description - A brief description of your application.
   - SWF URL - The URL that points to the location where your SWF will be hosted on the Internet.
   - Preview Image URL - A URL to an image file that shows a preview of your application. This is used to display a reference to your application in certain social networks.

9 Select the button and specify its position and size with the Properties panel (Window > Properties).

10 Finish building the shareable application.

   Note: *When building the shareable application, make sure tags for any visual components are placed before the Share menu code.*

More Help topics

Creating a widget using the Flash Professional authoring extension

Creating a shareable application using Flash Professional

Maximum widths for shareable applications

Add a Share Menu to an existing application

You can add a Share menu component to an existing FLA or SWF file.

1 Open the Components panel, if it is not already open. Select Window > Components.

2 If the Share section of the Components panel is not expanded, click the "+" next to Share to expand it.

3 Drag Share Menu from the Components panel, and drop it onto the stage.

4 Open the Share panel, if it is not already open. Select Window > Other Panels > Share.

5 Set up the application configuration by filling in the fields for Basic Config. You can further customize the Share menu by entering data for Optional Config using the Sharing, Mobile, FlashVars, URL Parameters, and Advanced tabs.

The following fields are mandatory:

- Partner ID - The Partner ID you want to associate with this application. For information about finding your Partner ID, see "About Partner IDs" on page 4.
- Code ID - The identifier or name you'd like to give this application for tracking analytics (for example, myWeatherApp).
- Config ID (optional) - The name you'd like to identify a particular configuration or variant of this application when programming content with Flash Vars.
- Script Version - The ActionScript Version used in your project.
- Title - The name that end users will see when sharing your application.
- SWF URL - The URL that points to the location where your SWF will be hosted on the Internet.
- Preview Image URL - A URL to an image file that shows a preview of your application. This is used to display a reference to your application in certain social networks.

More Help topics

"Configuration parameters reference tables" on page 67

http://www.adobe.com/go/distribution_service_max_widths

Edit an existing shareable application

You can edit an existing shareable application's FLA or SWF file.

1 In Flash Professional, open an existing application.

2 Open the Share panel, if it is not already open. Select Window > Other Panels > Share.

3 Customize the Share menu theme using the options in the Color, Layout, Destinations, and Advanced tabs of the Share menu panel.

4 Set up the application configuration by filling in the forms for Basic Config. You can further customize the Share menu by entering data for Optional Config using the Sharing, Mobile, Flash Vars, URL Parameters and Advanced tabs.

The following fields are mandatory:

- Partner ID - The Partner ID you want to associate with this application.
- Code ID - The identifier or name you'd like to give this application for tracking analytics.
- Config ID (optional) - The name you'd like to identify a particular configuration or variant of this application when programming content with Flash Vars.
- Script Version - The ActionScript Version used in your project.
- Title - The name that end users will see when sharing your application.
- SWF URL - The URL that points to the location where your SWF will be hosted on the Internet.

- Preview Image URL - A URL to an image file that shows a preview of your application. This is used to display a reference to your application in certain social networks.

Program a button to open the Share Menu

Add a button to the stage. Then, put an event on the button that will open the Share Menu.

***Note:** If you create a Share menu from a template this procedure is unnecessary.*

This is an example for ActionScript 3:

```
my_ShareMenu.hide();
share_btn.addEventListener(MouseEvent.CLICK, myShareHandler);
function myShareHandler(evt){
    my_ShareMenu.show();
}
```

This is an example for ActionScript 2:

```
my_ShareMenu.hide();
share_btn.onPress = function()
{
my_ShareMenu.show();
}
```

Customizing a Share Menu theme

What is a theme?

Included in the Share menu panel are pre-populated themes. Each theme contains saved configurations of Share menu components. You can modify, import, and export saved themes to or from XML files rather than re-using the Share menu config panel every time you build an application. Each theme contains pre-defined settings for a style of shareable application. Not included in a theme are the application settings, such as dimensions, and IDs.

Import and export a theme

You can create new themes by changing settings in the panel, and saving your changes. New themes will be added to the list of themes for future use. Themes can also be exported for use in Distribution Service extension for Dreamweaver CS4, Distribution Service extension for Flex Builder 3 or Distribution Manager. To import a theme:

1 Select Window > Other Panels > Share.

2 In the share panel, click the Layout tab.

3 Select a theme.

If you customize a theme and want to store the customizations for use in future shareable applications:

1 click Export to save the theme for future use.

2 In the Save In field, browse to a location to save your theme.

3 Type a filename in the File Name field.

4 Click Save.

Share menu colors

You can further customize the color scheme of the Share menu by clicking the Color panel of the Customize Theme view, and then selecting colors for many of the Share menu’s design characteristics.

Share menu layout

To change the layout of the Share menu, click the Layout panel, and choose among a variety of layout presets, or specify your own. You can also import existing layouts in this panel.

Destinations and advanced tracking

In the Destinations panel, you specify the social networks to which your application will be shared. You also specify bookmarking sites. If you select the Enable Advanced Tracking option, your shareable application can take advantage of sophisticated tracking services. See "Using the Distribution Service API for Flash and Flex shareable applications" on page 48 for information about reporting on advanced metrics.

Advanced theme options

The advanced theme options fall into the basic categories of: Display, Background, Social Network Buttons, Textboxes, Buttons, Listboxes, Checkboxes, Service Marker, Tooltips, Privacy Text, Headers, Messages and Links. For each of these, fonts, colors, sizes, borders, gradients, etc., can be modified.

Customize a Share Menu theme

1. Place a Share menu on the stage.
2. If it is not already open, open the Share panel.
3. In the Share panel, do any of the following:

- To change the color of the Share Menu, click the Color tab, and select a color preset.
- To change the layout of the Share Menu, click the Layout tab. Select a layout, and check the box next to each of the options in the Layout Options menu you want to enable.
- To change the available social networks and bookmarking networks, click the Destinations tab. Check the box next to each of the options in the Social Networking menu and Bookmarking menu you want to enable
- To change additional options, click the Advanced tab. Check the box next to each option desired from the menu.

4. Fill out the fields in each of the four tabs (Edit Theme, Basic Config, Optional Config and Embed Code). Red asterisks indicate required fields.

More Help topics

"About Partner IDs" on page 4

"Share menu options" on page 24

Shareable application configuration

You use the Share panel to set your shareable application's basic and optional configurations. See "Share menu options" on page 24 for information about each field.

Email, Bookmarks and Desktop sharing

Within the Share panel, you can select Optional Config and then Sharing to set specific values to control how your application will be shared using Email, Bookmarking services and Desktop environments.

The specific config options are: Email Subject, Email Body, Bookmark URL, Desktop Icon URL, and Copyright. For more information, see "Share menu options" on page 24.

Enabling an application to be shared with mobile devices

To share your application with mobile devices, you must configure the Share menu for the web version of your application. This involves the following two steps, both of which are required:

1. Enable the Mobile tab on the Share menu.

   Within the Share panel, select Edit Theme and then Layout. On the right side of the Share panel, check the box next to Show Mobile (Beta) to configure the Share menu to display an option to end users that allows them to share your application to their mobile device. Notice that after checking this box, the preview is updated to display a new option, Mobile, on top of the Share menu.

2. Configure the locations where your application can be found from the Share menu.

   Within the Share panel, select Optional Config and then Mobile to set the URLs that point to the various mobile versions of your application. For platforms where users locate and download applications from an applications store (for example, Apple IPhone), set the URL field to the location that refers to your application in the store. For Windows Mobile and Symbian platforms, set a URL that points to the specific .CAB file (for Windows Mobile) or .SIS file (for Symbian) where your mobile applications will be hosted on the internet.

***Note:** If you enable the Mobile tab on the Share menu but do not configure the locations where the Share menu can find your application, the Distribution service displays the message: No Mobile content.*

Set the Other field to a URL that the mobile device will be redirected to if the Distribution service determines that user's mobile phone is not one of the types listed above. Use this URL to display a web version of your application, perform further handset detection on your own, or simply display a web page that says there is no appropriate version of the application for the mobile device.

Using FlashVars to program content

You can use FlashVars to alter the content and behavior of your shareable application at runtime. See Using Dynamic Variables with Applications for information.

Using URL Parameters

URL parameters provide another method of displaying different content using the same basic shareable application. See "Share menu options" on page 24 for information.

Advanced Share menu parameters

From the Optional Config tab, click the button labeled Advanced. The menu here allows you to add advanced parameters to the Share menu. Use this drop down menu to add any additional Share Menu parameters such as entering a custom SWF URL for a specific social network. See http://www.adobe.com/go/fps_distribution_flash_adv_params_help for more information including a description of each of the advanced parameters.

HTML Embed code

The last step of the Share panel is the creation of the Embed code for the shareable application. The embed code that is generated can be placed in any HTML page. The HTML code displays the shareable application and makes it trackable by the ad network.

1. Select an application.
2. Select the Share component of the application on the stage.

3 Click on the Embed Code tab at the top of the application. It becomes enabled as soon as you fill out the mandatory fields (Partner ID, Code ID, Title, SWF URL, Preview Image URL) on the Basic Config Tab. The panel walks you through a wizard.

4 Specify the options in each step and click Next to proceed to the next step, or Previous to go back.

5 Copy the HTML embed code and paste it into a web page. You can also edit the HTML embed code directly. Any edits you make are saved in the instance of the Share menu component.

Using FlashVars to modify content programming

You can use FlashVars to alter the content that your shareable application displays at runtime. See Using Dynamic Variables with Applications for more information.

Share menu options

Basic Config

Partner ID identifies the account used for tracking all of your shareable applications. You can obtain a Partner ID with Distribution Manager. Your Partner ID is visible in the lower right corner of Distribution Manager. Alternatively, you can find your Partner ID in your account information on the Gigya website.

Code ID Used to identify a particular shareable application along with an optional Config ID. In certain cases where you use FlashVars to program one shareable application to show different content you can use the Config ID to identify the variants of a particular shareable application. Code ID and Config ID can have up to 49 alphanumeric characters, including whitespace. Trailing spaces will not be included. No special characters are allowed. See Using Dynamic Variables with Applications for information.

Config ID (Optional) The name you use to identify a particular configuration or variant of this of this application when programming content with FlashVars.

Script Version The ActionScript version that you have used for your application.

Title The name given the shareable application, typically by the developer of the application.

SWF URL The URL of the location where your SWF file will be hosted on the Internet.

Preview Image URL The URL of the location of an image file that shows a preview of your application. This is used to display a reference to your application in certain social networks.

Description A brief description of the shareable application.

Optional Config: Sharing tab

Email Subject You can customize the e-mail that is sent when your application is shared via an e-mail. If you include the replacement text "$sender$" in the subject it will be replaced with the name entered by the sender.

Email Body You can customize the e-mail that is sent when your application is shared via an e-mail. If you include the replacement text $sender$ in the body it will be replaced with the name entered by the sender. Also, f you include the replacement text URL in the body it will be replaced by the actual URL of the shareable application.

Bookmark URL By default, the URL to the page containing your shareable application will be used for bookmark services. It can be overridden to use a different URL by setting a value in the Bookmark URL field.

Desktop Icon URL used to retrieve and display an icon in desktop application environments.

Copyright The date the shareable application was copyrighted.

Optional Config: Mobile tab

Windows Mobile allows you to specify a destination for a mobile device that uses the Windows Mobile platform.

Symbian allows you to specify a destination for a mobile device that uses the Symbian S60 mobile platform.

Other allows you to specify a destination for a mobile device that uses a mobile platform other than Windows Mobile or Symbian.

Optional Config: Flash Vars tab

Flash Vars can be entered into your embed code here which allow you to control the programming of your shareable application by modifying the values. Your SWF must have been developed to take advantage of specific Flash vars by name or they will have no effect. A typical use of Flash vars is to program a templated shareable application into a number of different applications simply by varying the Flash vars to adjust the content. See Using Dynamic Variables with Applications for a tutorial that shows how to use FlashVars in a shareable application.

`FlashVars` are variables that can be passed from a web page to a SWF file (in this case a shareable application). You can define any variables you like in the HTML code of a page and these variables can be used by the shareable application.

You pass values to a shareable application from the HTML code as the shareable application loads using the `FlashVars` parameter/attribute of the `<object>` and `<embed>` tags. There are many reasons for doing this. For example, you can reuse the same basic application on different pages, but alter certain aspects using variables.

`FlashVars` can be entered into your embed code using the Distribution Service extension, or you can add them to your SWF file's ActionScript manually. Every shared application has a Code ID that is used to uniquely identify it, along with an optional Config ID. In cases where you use `FlashVars` to program one shareable application to show different content, you can use the Config ID to identify the variants of a particular shareable application.

Using FlashVars to change a shareable application's content at runtime means that you can use a single application as a template to format other similar content. For example, you can design a shareable application that contains categories of content (title, description, video, contact information, for example), but that does not contain the actual content. Up to 16 different versions of the content can be passed in at runtime using FlashVars. You don't have to redesign and recompile separate SWF files to achieve this result.

***Note:** this use of the word "template" to describe using a single shareable application as a template for multiple application versions (using FlashVars) is different from the use of the word "template" in the Flash Professional authoring tool. In the authoring tool, a template is a predesigned structure that you can use when you are initially creating a shareable application's SWF file.*

Optional Config: URL Parameters tab

URL Parameters can be entered here to control the URL of your shareable application. When you add a URL parameter, the URL to appended with a name/value pair in the following format: "http://some-url.com?parameter-name=parameter-value". Typically this is used for retrieving language specific versions of your shareable application, for example: "http://some-url.com?language=en-US".

Optional Config: Advanced Parameters

Advanced Parameters The menu here allows you to add advanced parameters to the Share menu. Use this drop down menu to add any additional Share Menu parameters such as entering a custom SWF URL for a specific social network. Click Help for more information including a description of each of the advanced parameters.

See http://www.adobe.com/go/fps_distribution_flash_adv_params_help for more information including a description of each of the advanced parameters.

Embed Code

The embed code is a generated HTML snippet that, when placed in an HTML page, displays your application and a Share menu that is configured with your application's details, which enables users to share it. Paste the generated HTML code snippet between the <body> tags in the web page in which you want to place the shareable application.

Generate revenue with a Share Menu

In addition to making applications shareable, you can generate revenue from your shareable applications by participating in the Distribution service as a publisher. Applications from top-tier brand advertisers could be cross-promoted with your shareable application. Advertisers would pay you based on how many times their application is installed. A Share menu does not displace any current advertising real estate on your page so all revenues earned are incremental.

For more information about joining the ad network, click Account in Distribution Manager.

Tracking sharing activity

You can track the sharing activity of your shareable applications. For more information, see "Tracking and analytics" on page 12.

Building and Packaging Flash Lite 3.1 applications for Windows Mobile and Symbian S60

You can build new Flash Lite 3.1 applications, or use existing Flash Lite content to create mobile applications to share virally, or distribute over the internet. The distributable player solution enables developers to create rich applications for the latest version of Adobe® Flash Lite® and distribute their content directly to millions of open OS smartphones, providing a better on-device user experience. For more information on the Flash Lite Distributable player see: http://labs.adobe.com/technologies/distributableplayer/

To take advantage of the distributable player solution, simply create and test your Flash Lite application and then package it.

Create and test your Flash Lite application

Create and test your Flash Lite application using Adobe Flash CS4 Professional and Adobe Device Central CS4 (with corresponding updates below), or Adobe Flash CS3 Professional and Adobe Device Central CS3.

Adobe AIR 1.5 update for Flash CS4 Professional

Flash Lite 3.1 update for Adobe Device Central CS4

The Distribution service extension for Flash CS4 Professional also offers templates for some standard sized applications. To create a new application using one of these templates, launch Flash CS4 Professional and then select File > New… and selects the Templates tab. Under the Categories list select "Apps-Mobile" and choose one of the standard sized templates. These templates are pre-configured with their publish settings set to the Flash Lite 3.1 player. For more information on creating Flash Lite applications see http://www.adobe.com/devnet/devices/.

Packaging for Windows Mobile devices

The Distribution Service extension for Flash CS4 Professional contains a packaging tool that enables developers to package Flash Lite applications into native install packages (.CAB files) for Windows Mobile devices. The Packager panel also requires that the Distribution Manager application is installed on the same computer.

Once you have created an application and published it to a SWF file, you can package it in a Windows Mobile .CAB file by following these steps:

1 Select Window > Other Panels > Packager to open the Packager panel.

2 Set the following attributes for your application:

- Application name

  Set this to the name of your application. This name displays in the text message that is sent to users and also identifies your application on the users mobile device.

- Application version

  This is embedded into the application's metadata to identify the particular version of your application. The version displays during the installation process and can be displayed using the mobile device's application manager.

- Author

  Set this to the name of the application's author. This name displays during the installation process and can be displayed using the mobile device's application manager.

- Description

  This should be set to a short description of the application. This value displays during the installation process and can be displayed using the device's application manager.

- Mobile Platform

  For the Beta release of the mobile sharing feature, this can only be set to Windows Mobile.

- Main SWF

  Select the location of the SWF file on your local file system that you want to package.

- Icon

  Select the location of an .ICO image file on your local file system that will be used to identify your application on the user's mobile device. The standard size for mobile icon application images is 48 pixels wide by 48 pixels tall.

- Output file

  Select the location on your local file system where you want the generated .CAB file to be written. Make sure that you have access to write a new file to this location.

3 Click the Create Package button to produce a .CAB installer file. This process will involve using functionality in the Distribution Manager application, so you will receive an error if it is not installed on the local machine.

4 Upload the .CAB installer file to your internet hosting site, where it can then be accessed by the Share menu.

***Note:** The Packager panel can not produce digitally signed Windows Mobile installer files. If you need to produce a signed .CAB file you must use a signing tool provided by Microsoft.*

Packaging for the Symbian S60 devices

For the Beta version of the mobile sharing feature packaging is not yet available for Symbian S60 in the Distribution Service extension for Flash CS4 Professional. Use the Adobe Mobile Packager to wrap your SWF file with an icon, metadata, and a version checker that downloads the latest Flash Lite player over the air. To download the Adobe Mobile Packager see: http://labs.adobe.com/downloads/distributableplayer.html.

For product documentation on using the Adobe Mobile Packager to wrap SWF files for the Symbian S60 platform, see http://labs.adobe.com/technologies/distributableplayer/.

Chapter 3: Using Flex Builder 3 to create shareable applications

Distribution Service extension for Adobe Flex Builder 3

The Distribution Service extension for Adobe Flex Builder 3 is a plug-in for Flex Builder 3 Standalone and the Eclipse plug-in. The Distribution Service extension for Flex Builder 3 allows developers to create applications that can be shared and, through the ad network, monetized.

About making applications shareable

The Distribution Service Extension for Flex Builder 3 provides developers with tools to help site visitors share applications, to track the sharing activity of the visitors, and to earn income through the ad network. The Distribution Service Extension for Flex Builder 3 enables you to create an application from scratch that incorporates a Share menu, or to add a Share menu to an existing MXML file.

Shareable applications are rich media applications typically designed in small formats for embedding in web pages, and for sharing between users on social networking sites.

When visitors click a button on a shareable application, the Share menu is displayed over the application. The Share menu lets visitors do any of the following:

- copy the embed code for posting the application onto another page,
- post the application directly on their blog or social networking site,
- install a copy of the application on a mobile device or desktop environment
- bookmark the application with a social bookmarking service such as Delicious, Digg, or Google Bookmarks,
- E-mail the application to others.

To use the all the functions and tools of Distribution Service extension for Flex Builder 3, you must have a Partner ID. You can obtain a Partner ID through the Distribution Manager.

You can earn money by joining the ad network. If you choose to participate in the advertising network, you can be paid to promote applications from top-tier brand advertisers.

More Help topics

Creating a shareable application using Flex Builder 3

Sharing Flash applications to the desktop

Uninstall Distribution Service extension for Flex Builder 3

Follow these steps to uninstall a previous version of the Distribution Service extension:

1 Launch Flex Builder 3 (or Eclipse).

2 Click Help > Software Updates...

3 Click the Installed Software tab. In the list below, select Distribution Service extension for Flex Builder 3.

4 Click Uninstall... and wait for the Uninstall dialog box to appear.

5 Make sure Project Radiate Extension for Flex Builder 3 is checked and click Finish.

6 Wait for it to finish and restart Flex Builder 3 (or Eclipse) when prompted.

7 Click Help > Software Updates...

8 Click the "Manage Configuration" tab and right-click the component you want to uninstall.

9 Click Manage Sites...

10 Click the entry in the list that says "jar:file [path to the zip file of the extension]. Make sure this is the only thing selected and click "Remove"

Install Distribution Service extension for Flex Builder 3

Follow these steps to install the Distribution Service extension for Flex Builder 3:

1 Launch Flex Builder 3 (or Eclipse).

2 Click Help > Software Updates...

3 Click the "Find and Install" tab.

4 Click "Add Site..." and select "New Archived site..."

5 Browse and select the downloaded zip file and click "Open", then "OK".

6 In the "Search for new features to install" tab, find the entry that says "jar:file [path to the zip file of the extension]. Expand this to see Flex Distribution Service Plugin. If necessary, expand that to see Distribution Service extension for Flex Builder 3. Click the check box next to the extension.

7 After a moment, the Install dialog will appear with "Distribution Service extension for Flex Builder 3" checked. Click "Finish".

8 Restart Eclipse when prompted.

Make applications shareable

The Distribution Manager extension for Flex Builder 3 enables users of Flex Builder 3 to create and publish shareable applications, and to monetize those applications by attaching advertising-based Share menu es to them.

Workflow for creating and publishing applications

To create and publish applications with Adobe Flex Builder, follow these general steps:

1 Create a new application from a template. See “Create a shareable application from a template” on page 31.

2 Implement your application.

3 Customize the Share menu. To do this:

   a Switch to Design view.

   b In the Outline, select the Share menu component.

   c Using the Flex Properties panel, you can edit:

      - Theme (select from a pre-defined set of themes or import a theme from a file)
      - Show Bookmarks (true or false)
      - URL for Bookmarks

- Show e-mail (true or false)
- E-mail subject
- E-mail body
- Show Desktop (true or false)
- Show code box (true or false)
- Show icons only (true or false)
- Show close button (true or false)

4 Export a release build of the project (Project > Export Release Build . . .)

Flex Builder will update the Share menu component on the stage with all of the actual code from the panel.

5 Upload the HTML and SWF file to the server where it will be hosted on the Internet.

Using an application that can upload files to a web server, such as Dreamweaver, put the SWF file and the HTML wrapper for the application onto a web server.

Create a shareable application from a template

You can create a shareable application with the New Shareable Application wizard. For definitions of the fields you fill in the wizard, see "Shareable application project settings" on page 33.

1 Select File > New > Other > Flex Builder > Shareable Application Project.

2 Click Next.

3 Type a project name.

4 Do one of the following:

- Click Browse, and browse to a project location.
- Check Use Default Location.

5 (Optional) To set an output folder different from the default, click Browse, and browse to a different folder.

6 Click Next.

7 Enter the following configurable values:

- Partner ID - The Partner ID you want to associate with this application. For information about finding your Partner ID, see "About Partner IDs" on page 4.
- Code ID - The identifier or name you'd like to give this application for tracking analytics (for example, myWeatherApp).
- Config ID (optional) - The name you'd like to identify a particular configuration or variant of this application when programming content with FlashVars.
- Script version - The ActionScript Version used in your project.
- Title - The name that end users will see when sharing your application.
- Copyright - Information about your application's copyright holder.
- Description - A brief description of your application.
- SWF URL - The URL that points to the location where your SWF file will be hosted on the Internet.
- Preview Image URL - A URL to an image file that shows a preview of your application. This is used to display a reference to your application in certain social networks.

- Advanced Tracking (checkbox: on or off) - A checkbox that enables advanced tracking for the application.

8 Click Next.

9 (Optional) Type any Flash variables or URL parameters.

10 Click Next.

11 From the Shareable Application Template menu, choose the size of the application (not the size of the Share menu but size of the application itself). Choose Custom to enter a custom size.

12 (Optional) Check Create A Default Share Menu if you want Flex Builder automatically to set up the application project for you with a button that opens the Share menu on the stage.

13 Click Next.

14 (Optional) Select the Source Path tab, and click Add Folder to add one or more folders to the source path.

15 (Optional) To change the source folder or main application file, click their respective Browse buttons.

16 (Optional) Select the Library Path tab.

17 (Optional) To change the source folder or main application file, click their respective Browse buttons.

18 Click Finish.

19 Finish building the Flex shareable application.

***Note:** When building the Flex shareable application, make sure tags for any visual components are placed before the Share menu code.*

More Help topics

Creating a shareable application using Flex Builder 3

Add a Share menu component to a Flex shareable application

You can add a Share menu to an existing Flex shareable application. This is only necessary if you are working on an existing Flex project. If you create a new Share Project, everything is set up automatically.

***Note:** This is only necessary if you are working on an existing Flex project. If you create a new Share Project, everything is set up automatically.*

1 Right click the Flex project and select properties.

2 Click Flex Build Path, then the Library Path tab.

3 Click Add SWC…

4 Navigate to the path where the SWC was installed by the plug-in and select it. The path can be found by performing steps 1 and 2 on a Flex Shareable Application Project and looking at the path that is added automatically.

5 Add the shareable application namespace to the Flex project by adding the following code to the <mx:Application> tag: xmlns:widget=http://flex.adobe.com/widget

You can now add a ShareMenu component to the project with the tag <widget:ShareMenu>

6 In the Flex Navigator panel, click the + next to the name of the desired shareable application to expand the list of folders.

7 Click the + next to the src folder for the desired shareable application.

8 Double-click the mxml file in the src folder.

The selected shareable application appears on the stage.

9 In the Components panel, click the + next to the Shareable Application folder.

10 Drag ShareMenu from the Components panel, and drop it onto the shareable application that is on the stage.

11 In the Flex Navigator panel, select the desired mxml file.

12 Select Project > Properties > Flex Shareable Application.

13 Enter the Partner ID, Code ID, Application Title and SWF URL. For the definitions of these fields, see “Shareable application project settings” on page 33.

14 Set other shareable application properties as desired, and click OK.

15 Select File > Save to save the shareable application with the associated Share menu component.

Shareable application project settings

Partner ID Text field. An ID issued by Gigya that identifies a specific account on the Gigya ad network where applications, campaigns and analytics are stored. You can obtain a Partner ID from your advertiser or publisher, or find your Partner ID in the lower right corner of Distribution Manager. The Partner ID must be passed as a property to the ShareMenu component and the EventService component with a parameter named pid.

Restrictions: Up to 32 alphanumeric characters.

Code ID Text field. The Code ID is compiled into the application to identify that particular binary asset or creative. It can optionally be paired with a Config ID. If you want to reuse the same basic application to display different content, you use the same Code ID but add a different Config ID.

Restrictions: max 50 characters; letters, numbers, and whitespaces.

Config ID Text field. The Config ID identifies any custom content programming injected into the application using FlashVars. The Config ID is passed into the application through the HTML embed code. The Config ID is used to identify the embed code itself (which could contain FlashVars that determine the specific content displayed by the application). The Config ID is used in conjunction with the Code ID to uniquely identify a particular variant of an application. For example, if your Code ID is "Sample App" and your Config ID is "US English version", your application is identified as "Sample App / US English version". Different variants of the Sample App can be programmed using FlashVars, such as "Sample App / ES Spanish version". This value should be used in the HTML embed code, inserted into the FlashVar named 'gig_cfg'. The FlashVar should be included in both the "<embed ..>" and "<object ..>" tag (see the template provided by Adobe).

Restrictions: max 49 characters; letters, numbers, and whitespaces.

Default value: empty.

In the Source view, this is *cid*. It must be passed as a property to the ShareMenu component and the EventService component, if you are using it.

Title Text field. The name that end users will see when sharing your application. In the Source view this is widgetTitle. Max 50 chars.

Copyright Text field. This string will be used as the copyright notice of the shareable application.

Restrictions: max. 512 characters.

Description Text field. A brief description of the contents and functions of your shareable application. Fee-form text up to 1024 chars. In Source view this is widgetDescription.

SWF URL Text field max 512 chars. SWF URL is the location where the application’s SWF file will be hosted on the Internet. For example: http://some.url.com/MyShareableApp.swf. This value is inserted into the HTML embed code.

Advanced Tracking This is a checkbox that enables advanced tracking for the application.

Preview Image URL Text field. max 512 characters. A URL to an image file that shows a preview of your application. This is used to display a reference to your application in certain social networks.

Shareable Application Template Menu. For pre-defined or custom template to control the size of the application. Templates available:

- 120 x 600 - Skyscraper
- 160 x 600 - Wide Skyscraper
- 180 x 150 - Rectangle
- 240 x 400 - Vertical Rectangle
- 250 x 250 - Square Pop-up
- 300 x 100 - 3 – 1 Rectangle
- 300 x 250 - Medium Rectangle
- 300 x 600 - Half Page Ad
- 336 x 280 - Large Rectangle
- 720 x 300 - Pop Under
- Custom

Width, Height Number field. The width and height in pixels of the application (=<mx:Application). The Width and Height values of the application also needs to be inserted into the HTML embed code in both the <OBJECT …/> and <EMBED …/> tags. These two fields will be editable only when the Custom template option is selected. Other templates set these two values automatically. No negative values. Max values are 3200 x 3200.

URL Parameters These are arbitrary values that can be entered to control the URL of your shareable application. When you add a URL parameter, the URL to appended with a name-value pair in the following format: "http://some-url.com?parameter-name=parameter-value". Typically this is used for retrieving language specific versions of your shareable application, for example: "http://some-url.com?language=en-US".

Flash Vars These are arbitrary values that can be entered into your embed code which allow you to control the programming of your shareable application by modifying the values. Your SWF file must have been developed to take advantage of specific FlashVars by name or they will have no effect. A typical use of FlashVars is to program a templated shareable application into a number of different applications simply by varying the FlashVars to adjust the content. See “Share menu options” on page 24 for information on using FlashVars.

Modify Flex shareable application project properties

You can edit the project choices that you made initially for Partner ID, Code ID, Config ID, Title, Copyright, Description, SWF URL, Preview Image URL, Advanced Tracking (checkbox: on or off), Flash Vars and URL Parameters.

1 Select Project > Properties > Flex Shareable Application.

2 Modify the values as desired.

3 Click OK.

Modify the Share menu component

1 Open the Flex shareable application MXML file in Design view.

2 Using the outline window, select the Share menu component.

3 In the Flex Properties panel, edit values for:

- Theme (select from a pre-defined set of themes or import a theme from a file).
- Show Bookmarks (true or false)
- URL for Bookmarks
- Show e-mail (true or false)
- E-mail subject
- E-mail body
- Show Desktop (true or false)
- Show code box (true or false)
- Show icons only (true or false)
- Show close button (true or false)

Including custom events in a Share menu

Make sure that advanced tracking is checked in the Application Properties.

If it is not already in the MXML, add the Event services component. For example, you could add:

```
<widget:EventService id="customEvents" pid="[PID]" cid="[your CID]"/>
```

Then events can be sent back to the analytics service by calling customEvents.send('some Event name'). An example of how to use this is to put it on a button. For example, you could add:

```
<mx:Button x="225" y="228" label="event" id="eventButton"
click="{customEvents.send('customEvent')}" />
```

Program a button to open the Flex shareable application menu

By default, Visibility is set to False and the button will not appear in design view. You must change visibility to true to see it in design view. Remember to set it to False and include a button that can set it to True during runtime before publishing.

Add a button component using either design or source view. Hook up the click event so that it opens the Share menu:

```
<mx:Button id="shareButton" label="Share" click="{shareMenu.visible = true}" x="0" y="228"/>
```

Import a Share menu theme

You can import themes that were created from the Distribution Manager or the Distribution Service extension for Flash Professional CS4.

❖ Do one of the following:

- In Design view, on the properties view, next to Import custom theme, click the folder icon to browse for a theme xml file.
- Choose a theme from the Theme menu.

Hide social networks for a Share menu

1 In Design View, open Category View in the Flex Properties panel, and scroll down to the Social Networks category.

2 To hide any network, set it to False.

View the External Embed code to paste into an HTML page

1 Right-click the project and select Show Embed Code.

2 Share component on the stage.

Note: *The HTML template that is generated (under the project folder > html-template > index.template.html) will also have the external embed code. The Internal Embed code, which propagates with the Flex shareable application, can be found in the Source view of the mxml. It is a child component of the ShareMenu component and it is called <widget:embedCode>.*

Chapter 4: Using Dreamweaver to share applications and web pages

Getting started with the Distribution Service extension for Dreamweaver

About making web pages and applications shareable

The Distribution Service extension for Adobe® Dreamweaver CS4 provides web page designers with tools to help visitors share web pages and applications (SWF files). Applications can be small, shareable pieces of content or branded applications that blend interesting content with an advertising message.

The extension lets you insert an application in a web page and attach a Share menu to the application. The Share menu lets page visitors grab the application and post it on their blog or on a social networking site.

The extension also lets you insert elements such as a Share menu or a Bookmarks button in a page so visitors can share the page itself with others. The sharing elements let visitors do any of the following:

- post a link to the page in a blog or social networking site
- bookmark the page with a social bookmarking service
- E-mail a link to the page to friends
- Download and install a desktop or mobile version of the application.

You can configure sharing elements to track sharing activity. You can also participate in an advertising network. To use this functionality, you must have a Partner ID, which you can obtain with the Distribution Manager. For more information, see "Distribution Manager for advertisers and publishers" on page 1.

Install the Distribution Service extension for Dreamweaver

You can install the extension in Adobe Dreamweaver CS3 or Adobe Dreamweaver CS4.

1 Log into the Adobe website at http://www.adobe.com/go/fps and follow the instructions to download and install the Distribution Service extension for Dreamweaver (MXP file) to your local hard disk.

2 Double-click the downloaded MXP file to install the extension.

The Adobe Extension Manager opens and guides you through the installation process. For more information, see Downloading and installing extensions in Extension Manager Help.

Preview the sharing elements locally

The instance of Flash Player that Dreamweaver uses for previewing content in a web page lets you preview the sharing elements such as the Share menu when the page is staged on a web server. Flash Player may not let you preview the sharing element in a local version of the page or in Live View on a computer not configured as web server without changing the following Flash Player setting.

1 Point your browser to Flash Player Help.

2 Click Global Security Settings Panel.

3 Change the setting to Always Allow and add the location of your website’s project files on your local hard drive.

4 Close the browser and restart it before trying to preview the file again.

Make applications shareable

The Distribution Service extension for Dreamweaver lets you insert an application (a SWF file) with a Share menu button in a page. The button opens a Share menu that lets visitors do the following:

- copy the embed code to post the application on another page
- post the application directly on their blog or social networking site
- bookmark the application with a social bookmarking service
- E-mail the application to friends

Applications can be small, shareable pieces of content or branded applications that blend interesting content with an advertising message. They are created separately in Adobe Flash Professional or Adobe Flex Builder.

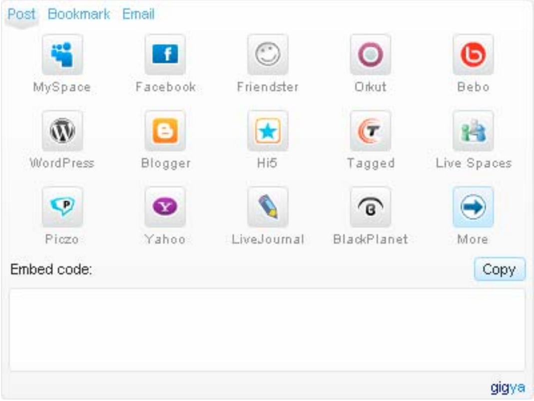
***Note:** You can also attach a Share menu directly to a SWF file in Flash Professional or Flex Builder. For more information, see “Using Flash Professional to create shareable applications” on page 16 or “Using Flex Builder 3 to create shareable applications” on page 29.*

More Help topics

Using Dreamweaver to share existing applications

Add a Share menu in page

A Share menu in page enables your users to easily grab and place your application (or any embed code) on their preferred destination.



A Share menu

1 With a web page open in Dreamweaver, drag the Share Menu in Page element from the Insert panel to the page. The Share Menu in Page element is located in the Share Menu category of the Insert panel.

***Note:** If you get a Flash Player error when trying to preview the page, see "Preview the sharing elements locally" on page 38.*

2 Change the size and shape of the Share menu to fit the design of your page by using the width and height (W & H) properties in the Property inspector. The layout of the Share menu adjusts automatically to the new size.

3 Set any of the following properties in the Property inspector. All the properties are optional.

Partner ID Enter a Partner ID. The Partner ID identifies who owns the shareable content the Share menu is attached to, and allows sharing activity to be tracked for reporting and analytics purposes.

Dreamweaver saves the Partner ID value on a per-site basis for use the next time any sharing element is added to a page in the site. Adding or updating the Partner ID within the site stores that ID for the current and future sharing elements within the site. Dreamweaver does not update prior instances within the site.

Title Enter a title for the application that will be presented to users.

Theme Select a theme for the Share menu.

Edit embed code Click to specify the HTML embed code for your application, or any portion of the web page that you wish to share.

Import Theme Click to import an existing theme. For more information, see "Generating a Share Menu theme" on page 47.

Advanced tracking Check this box to enable reporting services to track viral distribution and other advanced metrics in social networks. For more information, see "Application reports" on page 14.

Code ID Enter the Code ID for the selected application instance. The Code ID is used to identify this particular application along with an optional Config ID. The Code ID is compiled into the application SWF file by the Flash developer. Dreamweaver combines this value with the Config ID with a "/" between values. Dreamweaver then makes this the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Config ID Enter the Config ID for the selected application instance, if needed. In certain cases, you can use FlashVars to program one application to show different content. You can use the Config ID to identify the variants of a particular application. Dreamweaver combines the Config ID value with the Code ID with a "/" between values. Dreamweaver then uses as the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Networks Click to change the social networks that appear in the Share menu, the order in which they appear, or whether they appear at all. Use the Plus (+) or Minus (-) buttons to add and remove items from the list. Use the arrow buttons to reorder the items in the list. Click OK to accept your changes. Click Update Networks to pull the latest copy of the network list.

Bookmarks URL Enter a URL for the social bookmarking services to use instead of the current page's URL.

Show Bookmark Check this option to display the Bookmark link in the Share menu.

Show Email Check this option to display the Email link in the Share menu.

Show Desktop Check this option to display the Desktop link in the Share menu.

Show Code box Check this option to display the Embed Code box in the Share menu.

Show Icons only Check this box to display the social networks' icons without labels in the Share menu.

4 Save and upload the page to your site.

***Note:** If you delete a Share menu in Design view, Dreamweaver deletes the menu's code from the document except for the script tag inserted into the head tag. Delete the script tag manually.*

Add a Share menu button

Adding a Share menu button achieves the same goal as a Share Menu in Page, but the button uses far less space than a full Share menu does. When users click the button a Share menu for your application opens in a small overlay window. The window lets visitors do any of the following:

- post a link to the page in a blog or social networking site
- bookmark the page with a social bookmarking service such a Del.icio.us, Digg, and Google Bookmarks
- E-mail a link to the page to their friends

A Share button

1 With a web page open in Dreamweaver, drag the Share Menu button from the Insert panel to the page. The Share Menu button is located in the Share Menu category of the Insert panel.

***Note:** If you get a Flash Player error when trying to preview the page, see "Preview the sharing elements locally" on page 38.*

2 Set any of the following properties in the Property inspector. All the properties are optional.

Partner ID Enter a Partner ID. The Partner ID identifies who owns the shareable content the Share menu is attached to, and allows sharing activity to be tracked for reporting and analytics purposes. The Partner ID can be obtained by creating an account with Adobe using the Distribution Manager. For more information, see "Establishing an account" on page 4.

Dreamweaver saves the Partner ID value on a per-site basis for use the next time any sharing element is added to a page in the site. Adding or updating the Partner ID within the site stores that ID for the current and future sharing elements within the site. Dreamweaver does not update prior instances within the site.

Title Enter a title that will be presented to users for the web page you are sharing.

Theme Select a theme for the Share menu.

Edit embed code Click to specify the HTML embed code for your application, or any portion of the web page that you want to share.

Code ID Enter the Code ID for the selected application instance. The Code ID is used to identify this particular application along with an optional Config ID. The Code ID is compiled into the application SWF file by the Flash developer. Dreamweaver combines this value with the Config ID with a "/" between values. Dreamweaver then makes this the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Config ID Enter the Config ID for the selected application instance, if needed. In certain cases, you can use FlashVars to program one application to show different content. You can use the Config ID to identify the variants of a particular application. Dreamweaver combines the Config ID value with the Code ID with a "/" between values. Dreamweaver then uses as the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Image Custom URL Select this option and enter the URL of an image to use to customize the button.

3 Save and upload the page to your site.

More Help topics

Using Dreamweaver to share existing content

Wrap an existing application with a Share menu

You can wrap your application with a transparent Flash wrapper using the Share Menu Wrapper. This presents a “Get This” button that can be customized and placed wherever you choose. When this button is clicked, it will present the Share menu on your application. This is the best choice for making existing rich media content shareable when you no longer have access to the original FLA file.

To wrap an existing application with a Share menu and place it on a web page, do the following:

1 In Dreamweaver, open the web page in which you want to insert the application.

2 Drag the Share Menu Wrapper element from the Insert panel to the page. The Share Menu Wrapper element is located in the Share Menu category of the Insert panel.

3 Enter the URL of the application in the SWF URL text box in the Property inspector. Example: http://www.mysite.com/mycoolstuff/myapplication.swf.

Note: If the application does not appear when you preview the page, try selecting ActionScript 3 in the Property inspector.

4 Match the wrapper's dimensions to the application's dimension using the width and height (W & H) properties in the Property inspector.

5 Specify the position of the Get This button on your application by using the coordinate button properties (X & Y).

6 Set any of the following properties in the Property inspector. All the properties except application URL are optional.

Partner ID Enter a Partner ID. The Partner ID identifies who owns the application and allows sharing activity to be tracked for reporting and analytics purposes.

Dreamweaver saves the Partner ID value on a per-site basis for use the next time any sharing element is added to a page in the site. Adding or updating the Partner ID within the site stores that ID for the current and future sharing elements within the site. Dreamweaver does not update prior instances within the site.

Theme Select a theme for the Share menu that appears when the user clicks the Get This button.

ActionScript 2.0 Select this option if ActionScript 2.0 was used to create the application.

ActionScript 3.0 Select this option if ActionScript 3.0 was used to create the application.

Advanced tracking Check this box to enable reporting services to track viral distribution and other advanced metrics in social networks. For more information, see “Application reports” on page 14.

Code ID Enter the Code ID for the selected application instance. The Code ID is used to identify this particular application along with an optional Config ID. The Code ID is compiled into the application SWF file by the Flash developer. Dreamweaver combines this value with the Config ID with a "/" between values. Dreamweaver then makes this the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Config ID Enter the Config ID for the selected application instance, if needed. In certain cases, you can use FlashVars to program one application to show different content. You can use the Config ID to identify the variants of a particular application. Dreamweaver combines the Config ID value with the Code ID with a "/" between values. Dreamweaver then uses as the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Button Type Select the type of button to display within the application. Select Sticky to place the button on one of the edges of the application. Select Classic to place the button within the application. Select Custom to place the button elsewhere.

Button URL Enter a URL to the image or SWF file to use for a custom button. This option is not available if the button type is Sticky or Classic.

7 Save and upload the page to your site.

Make web pages shareable

The Distribution Service extension for Dreamweaver lets you insert sharing elements such as a Share menu or Bookmarks button in a page so visitors can share the page with others. The sharing elements let visitors do any of the following:

- post a link to the page in a blog or social networking site
- bookmark the page with a social bookmarking service
- E-mail a link to the page to friends

Add a Bookmarks button to a web page

A Bookmarks button opens a pop-up menu that lets visitors bookmark the page with a social bookmarking service such a Del.icio.us, Digg, or Google Bookmarks. When this button is used to bookmark a web page, it uses the URL of the page on which it is located as the URL bookmark.

A Bookmarks Button

1 With a web page open in Dreamweaver, drag the Bookmarks button from the Insert panel to the page. The Bookmarks button is located in the Share Menu category of the Insert panel.

   ***Note:** If you get a Flash Player error when trying to preview the page, see "Preview the sharing elements locally" on page 38.*

2 Set any of the following properties in the Property inspector. All the properties are optional.

   Partner ID Enter a Partner ID. The Partner ID identifies who owns the shareable content the Bookmark button is attached to, and allows sharing activity to be tracked for reporting and analytics purposes. The Partner ID can be obtained by creating an account with Adobe using the Distribution Manager. For more information, see "Establishing an account" on page 4.

   Dreamweaver saves the Partner ID value on a per-site basis for use the next time any sharing element is added to a page in the site. Adding or updating the Partner ID within the site stores that ID for the current and future sharing elements within the site. Dreamweaver does not update prior instances within the site.

Title Enter a title that will be presented to users for the web page you are sharing.

Theme Select a theme for the Add Bookmark menu that appears when a visitor clicks the Bookmark button.

Code ID Enter the Code ID for the selected application instance. The Code ID is used to identify this particular application along with an optional Config ID. The Code ID is compiled into the application SWF file by the Flash developer. Dreamweaver combines this value with the Config ID with a "/" between values. Dreamweaver then makes this the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Config ID Enter the Config ID for the selected application instance, if needed. In certain cases, you can use FlashVars to program one application to show different content. You can use the Config ID to identify the variants of a particular application. Dreamweaver combines the Config ID value with the Code ID with a "/" between values. Dreamweaver then uses as the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Image Custom URL Select this option and enter the URL of an image to use to customize the button.

3 Save and upload the page to your site.

Add a Share with Friends button to a web page

A Share with Friends button opens a pop-up window that lets visitors e-mail the page's link to their friends and also include a comment. When this button is used to e-mail the location of a web page to a friend, it places the URL of the page on which it is located in the e-mail.

A Share With Friends Button

1 With a web page open in Dreamweaver, drag the Share with Friends button from the Insert panel to the page. The Share with Friends button is located in the Share Menu category of the Insert panel.

Note: *If you get a Flash Player error when trying to preview the page, see "Preview the sharing elements locally" on page 38.*

2 Set any of the following properties in the Property inspector. All the properties are optional.

Partner ID Enter a Partner ID. The Partner ID identifies who owns the shareable content the Share with Friends button is attached to, and allows sharing activity to be tracked for reporting and analytics purposes. The Partner ID can be obtained by creating an account with Adobe using the Distribution Manager. For more information, see "Establishing an account" on page 4.

Dreamweaver saves the Partner ID value on a per-site basis for use the next time any sharing element is added to a page in the site. Adding or updating the Partner ID within the site stores that ID for the current and future sharing elements within the site. Dreamweaver does not update prior instances within the site.

Subject Specify a custom Subject heading for the e-mail. A custom subject can also help prevent the e-mail from being blocked by spam filters.

Theme Select a theme for the e-mail dialog box that appears when a visitor clicks the button.

Code ID Enter the Code ID for the selected application instance. The Code ID is used to identify this particular application along with an optional Config ID. The Code ID is compiled into the application SWF file by the Flash developer. Dreamweaver combines this value with the Config ID with a "/" between values. Dreamweaver then makes this the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Config ID Enter the Config ID for the selected application instance, if needed. In certain cases, you can use FlashVars to program one application to show different content. You can use the Config ID to identify the variants of a particular application. Dreamweaver combines the Config ID value with the Code ID with a "/" between values. Dreamweaver then uses as the Content ID (CID) value for the instance in the following format: {Code ID}/{Config ID}.

Image Custom URL Select this option and enter the URL of an image to use to customize the button.

3 Save and upload the page to your site.

Pasting sharing elements

If you paste a sharing element such as a Share menu or Bookmarks button in the same page from which you copied it, Dreamweaver may automatically update the values of the ID and name attributes of the text area associated with the element to ensure the values are unique.

If Dreamweaver updates the ID and name attributes, manually update the code to ensure that the associated text area is correctly referenced within the code. For example, if you copy and paste a Share Menu button and Dreamweaver updates the ID or name attributes, then update the CID query string value in the script src attribute to reflect the new attribute values. For the Share Menu In Page, update the defaultContent value with the new value.

If Dreamweaver does not update the ID and name attributes, then update the attribute values manually to ensure they are unique, as well as the CID query string value. You may need to update the defaultContent value to reflect the new ID or name attributes.

When you paste a sharing element from one document to another, update the ID and name attributes of the text area and the associated CID or defaultContent values to ensure uniqueness. If pasting a Share Menu In Page into another page, make sure to paste the script tag in the head tag.

Editing embed code

Modify shareable application code

1 Select a shareable application. Dreamweaver displays the share properties in the Property Inspector.

2 In the Property Inspector, click Edit Embed Code. The Edit Embed Code dialog opens.

3 Edit the code, using the options described below.

Edit embed code options

Embed code Enter or edit the HTML embed code for the shareable application. The code populates the text area associated with the selected shareable application. You can copy this code from Flash Professional when it has the Distribution Service extension installed.

OK Click this button to close the Edit Embed Code dialog box. The current application code value updates the text area associated with the selected shareable application.

Cancel Click this button to close the Edit Embed Code dialog box without causing any changes to the shareable application's code.

Help Click this button to open the online Help pages in a browser.

Specify networks

The Networks command allows you to specify the social networks that appear in the Share menu, the order in which they appear, or whether they appear at all. You can add and remove items from the list, as well as reorder the items in the list to achieve the mix of networks desired. The order of the networks may change depending on the context or the user's history.

1 Select a shareable application. Dreamweaver displays share properties in the Property Inspector.

2 In the Property Inspector, click Networks. The Networks dialog appears.

Sharable application networks options

Add Click this button to display a pop-up menu listing all the network options that are not already listed in the Networks list. When OK is clicked, any items that are listed in the Add button pop-up menu are used as the networks' To Hide list for the selected instance.

Remove Click this button to remove the selected network from the Networks list. This button is disabled if there is only one network remaining in the Networks list.

Up Click this button to move the selected item in the Networks list up one row. This button is disabled if the top item in the list is selected.

Down Click this button to move the selected item in the Networks list down one row. This button is disabled if the bottom item in the list is selected.

Networks This list contains all the networks that you have selected to offer to viewers of the application. By default, this list contains all the networks supported for the current version of the extension. Update this list the first time you run the extension and frequently thereafter.

Note: *Select Other to have this list updated, in the shareable application when it is live, with additional networks as they become available.*

OK Click this button to close the Networks command. The specified networks are updated in the code chunk for the instance. Clicking OK also saves the set of selected networks to facilitate reuse of this customized set of networks.

Cancel Click this button to close the Networks command without causing any changes to the application code of the text area for the inspected Share Menu In Page.

Update Networks Click this button to pull the latest copy of the network list.

Help Click this button to open a browser to the online Help pages.

Chapter 5: Programming Shareable Applications in ActionScript and HTML

Distribution Manager for Developers

About the Distribution Manager

The Distribution Manager is a standalone software-as-a-service application that is a component of Adobe Flash Platform Services. Flash and Flex developers install a local version of the Distribution Manager and use it (optionally in conjunction with extensions for Flash Professional, Flex Builder and Dreamweaver) to create shareable applications.

Sharable applications are rich media applications, typically designed in small formats for embedding in web pages, and for sharing among users on social networks. Adobe Flash Platform Services is powered by technology from Gigya, which gives developers tools to create, track and monetize shareable applications. You can earn money by joining the ad network and hosting ads on your shareable application for top-tier brand advertisers.

***Note:** Because the Share menu is powered by Gigya's Wildfire technology, in some places in the APIs and their documentation the word "Wildfire" and the name "Gigya" appear.*

To use the all the functions and tools of Flash Platform Services, you must have a Partner ID. You can obtain a Partner ID with the Distribution Manager.You can earn money by joining the ad network and hosting ads on your shareable application for top-tier brand advertisers. See "Getting started with the Distribution Manager" on page 3 for information on how to download and install the Distribution Manager, and how to get started using it.

The Create View

The Create view of the Distribution Manager (like the Distribution Service extension for Flash CS4) enables you to create an application from scratch that incorporates a Share menu, or to add a Share menu to an existing FLA or SWF file. When visitors click a button on a shareable application, the Share menu is displayed over the application. The Share menu lets visitors do any of the following:

- copy the embed code for posting the shareable application onto another page,
- post the shareable application directly on their blog or social networking site,
- bookmark the shareable application with a social bookmarking service such as Delicious, Digg, or Google Bookmarks,
- e-mail the shareable application to others.

To use the all the functions and tools of Flash Platform Services, you must have a Partner ID. You can obtain a Partner ID with the Distribution Manager.You can earn money by joining the ad network and hosting ads on your shareable application for top-tier brand advertisers.

The Create view has four tabs across the top that can be used to edit a shareable application's theme, configure the application's behavior, and generate the HTML embed code, which is a small snippet of HTML code that is used to embed your shareable application within a web page.

The left panel of the Create view has controls you can use to:

- insert a Share menu into an application.
- insert a Share menu in a web page.

Generating a Share Menu theme

Edit theme You use the options on the Edit Theme panel of the Create view to specify a wide variety of aspects of the Share menu's appearance and behavior. You can choose among a number of color presets that are already loaded, or you can import a theme you've created for another project. You can also export themes for use with the Distribution Service extensions for Flash, Flex and Dreamweaver by clicking Export once you've finished designing your Share menu in the Create view. At any point during this process you can click the Preview button to see what your completed Share menu will look like.

In addition to tweaking the colors and layout of your Share menu, you can specify the social network and bookmarking destinations for your shareable application in the Edit panel of the Create view. You can choose whether to enable Advanced Tracking, which lets you monitor various aspects of your shareble application's dis st rib uti on and monetization. See "Tracking and analytics" on page 12 for more information. You can also specify the default language in which your application's Share menu is displayed.

Basic Config You use this panel to enter basic information about your shareable application. See "Share menu options" on page 24 for details on what to enter in each field.

Optional Config You use this panel to enter two types of parameters: URL parameters and `FlashVars`. Both of these parameter types are used to change how your shared application works at runtime. URL parameters allow you to vary the URL of your shareable application to display different versions of the application, and `FlashVars` allow you to reuse the same basic shareable application on different pages, but alter certain aspects of it using variables. See "Using FlashVars to program content" on page 23 and "Using URL Parameters" on page 23 for more information.

Embed Code This panel displays the HTML embed code for your shareable application. The embed code can be placed in any HTML page. The HTML code displays the application and makes it trackable by the ad network.

Sharing applications

You can use the Create view of the Distribution Manager to create an application that incorporates a Share menu, or to insert a shareable application and Share menu into a web page. You can also use the Flash Professional or Flex Builder 3 extension to do the same thing.

Inserting a Share menu into an application

You can use the Create view of the Distribution Manager to design and create code for a Share menu, and then insert it in an application. You can also do this with the Flash or Flex extensions.

To insert a Share menu into an application using the Distribution Manager, perform the following steps.

1. Open the Distribution Manager and select the Create view.
2. In the Create view's left panel, click Share Menu in Application.
3. Select and fine-tune the Share menu's appearance using the options in the Edit Theme panel.
4. Enter your application's identifying information in the Basic Config panel. See "Share menu options" on page 24 for information about each field.
5. If desired, enter FlashVars and URL parameters in the Optional Config panel. See "Share menu options" on page 24 for information about each field.
6. An HTML Embed code is generated for you on the Embed Code tab. Use this Embed code to place your application on a web page.
7. On the Grab Code tab, ActionScript code is generated for you. You must insert this ActionScript code into your Flash application in the main timeline. Then create a button in your application and associate the code with the button.

Inserting a shareable application and share menu in a web page

You can use the Create view of the Distribution Manager to design and create code for a Share menu, and then insert it in an application. You can also do this with the Flash or Flex extensions.

To insert a Share menu into an application using the Distribution Manager, perform the following steps.

1 Open the Distribution Manager and select the Create view.

2 In the Create view's left panel, click Share Menu in Page.

3 Select and fine-tune the Share menu's appearance using the options in the Edit Theme

4 Select and fine-tune the Share menu's appearance using the options in the Edit Theme panel.

5 Enter your application's identifying information in the Basic Config panel. See "Share menu options" on page 24 for information about each field.

6 In the Embed Code field on the Basic Config tab, enter the HTML embed code for your application.

7 If desired, enter FlashVars and URL parameters in the Optional Config panel. See "Share menu options" on page 24 for information about each field.

8 The Embed Code panel contains the HTML embed code that is used by an HTML page to actually load the application. Paste this code between the <body> tags of the HTML in the location in which you want to place the shareable application.

Using the Distribution Service API for Flash and Flex shareable applications

Custom reporting is implemented as a type of service. There are several steps you need to take to use a service. See "Loading the Distribution Service" on page 48 and "Using Distribution Service API methods" on page 51 for more information.

Reporting services include services that enable shareable application developers to report custom events and activate advanced reporting options. Examples include:

- Reporting custom events
- Measure time spent
- Report interaction by users

Shareable applications have the ability to track custom user-defined events. For example, you can set up a shareable application so that it tracks every time a user clicks the shareable application.

Loading the Distribution Service

The Distribution Service ActionScript API is a set of namespaces which correlate to different services, all using a common programming methodology and aimed at shareable application developers using Flash or Flex and ActionScript 2 and 3.

Using any Distribution Service API call is a three-step process:

- Include the distribution.as file in your project
- Load the required service
- Make the actual API calls

Include distribution.as in your project

Download the distribution.as file appropriate for your development environment (ActionScript 2 or ActionScript 3) from the Adobe website and include it in your project by placing the file in your working directory or any other root of your classPath. These files are available at the following locations:

Distribution service ActionScript library (contains both ActionScript 2 and ActionScript 3 versions):

http://www.adobe.com/go/distribution_download

These files provide bootstrapping code for the Distribution Service API and should never be changed. You are encouraged to get the latest version of this file when you make changes to your project so that it includes the latest improvements and fixes.

The distribution.as will implicitly give the Distribution Service API code permissions to access your SWF file's data members. If the Distribution Service API is used in a SWF file that is loaded in another SWF file, or in a hierarchy of other SWF files, then all the parent SWF files must give Gigya permissions to access them as well, as follows:

In ActionScript 2:

```
System.security.allowDomain("cdn.gigya.com");
```

In ActionScript 3 and Flex:

```
Security.allowDomain("cdn.gigya.com");
```

Load service(s)

Before you can use any services, they must be loaded first. You load services by making a call to `distribution.load()` and waiting for your callback method to be called with an indication if the load was successful or not.

Note: The API methods are not ready for use before your callback method is called.

Syntax:

```
distribution.load(conf,params)
```

See "Making API calls" on page 51 for details on how to create the `conf` and `params` objects.

The result object passed to the callback function has the following members:

hadError - A Boolean indicating if an error has occurred.

requestParams - This is the conf obj that was passed on the original call.

errorCode - An error code or 0 if no error has occurred.

errorMessage - A string containing a description of the error.

errorData - This object provides additional data about the error.

failedServices - A comma separated list of services that failed to load.

detailedErrors- A string with additional details about the errors causing each failure respectively.

Example

```
// In AS2:
// Create the load configuration object
var loadConf:Object = {
    mcRoot: _root,
    pid: REPLACE_WITH_YOUR_PARTNER_ID
}
// Create the load params object
var loadParams:Object = {
    services:'social,reports',
    callback:onDistributionServicesInit
}
// Load the services
distribution.load(loadConf,loadParams);
// Wait for the load to complete and handle failures
function onDistributionServicesInit(res)
{
    if (res.hadError) {
        // handle failure
    } else {
        //continue knowing that the required services are now available.
        //please note that some services may require initialization after they are loaded.
    }
}
---------------------------------------
//  In AS3:
// Create the load configuration object
var loadConf:Object = {
    mcRoot: root,
    pid: REPLACE_WITH_YOUR_PARTNER_ID
}
// Create the load params object
var loadParams:Object = {
    services:'social,reports',
    callback:onDistributionServicesInit
}
// Load the services
distribution.load(loadConf,loadParams);
// Wait for the load to complete and handle failures
function onDistributionServicesInit(res):void
{
    if (res.hadError) {
        // handle failure
    } else {
        //continue knowing that the required services are now available.
        //please note that some services may require initialization after they are loaded.
    }
}
---------------------------------------
In Flex:
// Create the load configuration object
var loadConf:Object = {
```

```
    mcRoot: Application.application.root
    pid: REPLACE_WITH_YOUR_PARTNER_ID
}
// Create the load params object
var loadParams:Object = {
    services:'social,reports',
    callback:onDistributionServicesInit
}
// Load the services
distribution.load(loadConf,loadParams);
// Wait for the load to complete and handle failures
function onDistributionServicesInit(res):void
{
    if (res.hadError) {
        // handle failure
    } else {
        //continue knowing that the required services are now available.
        //please note that some services may require initialization after they are loaded.
    }
}
```

Using Distribution Service API methods

The API Namespace

The Distribution Service API uses a three-level, hierarchical namespace to specify all method names with the scheme distribution.services.{service name}.{method name}. The only exception to this rule is the `distribution.load()` method described below.

Making API calls

All the methods in the Distribution Service API (except the `distribution.load` method) have a common signature: `distribution.services.service_name.method_name(conf,params);` Both `conf` and `params` are actually objects that contain multiple members used for configuring the method. The API uses an asynchronous programming model where operations are started by calling an API method, run in the background, and call a user-supplied callback function when they are complete. The callback function is called regardless if the operation succeeded of failed, so the application can handle both cases gracefully. A single callback function may be used to handle multiple events. In this case the application can set a context object when it calls the API and the callback function can later use this object to determine the context in which it was called and react accordingly.

The Conf object

This object is used for storing common configuration parameters that are expected to remain static for the lifetime of the application. The conf object can be created once and passed to all the method calls in the application lifetime. Currently the object's members are:

•`pid` - The partner ID of the developer.

•`mcRoot` - The `_root` of the flash container.Example:

```
var conf = {
    mcRoot:_root,
    pid:_root.pid
}
```

The Params object

This object is used for passing parameters to the methods in the Distribution Service API. Each method accepts different parameters, as appropriate for that method. Additionally, all the methods in the Distribution Service API accept the following common parameters:

•callback - A function that Distribution Service will call, with the response of the API method when it is completed.

•context - A user (=developer) created object that will be passed back to the application in the result object passed to the callback function.

Getting return values

When an API call is made it returns immediately and starts processing the request in the background. When the processing is complete the user provided callback function is called and result object is passed to it. The result object contains the values returned from the specific API method. Additionally every result object also contains the following common members:

- hadError - A Boolean indicating if an error has occurred.
- requestParams - This is the params object that was passed on the original method call.
- errorCode - An error code or 0 if no error has occurred.
- errorMessage - A string containing a description of the error.
- errorData - This object provides additional data about the error.

Using Reports services to track interactions and custom events

The Reports service provides extended reporting capabilities to shareable applications using Distribution Manager, allowing developers to easily track shareable application usage patterns. The service supports tracking of interaction and activity times as well as sending custom event reports, appropriate for the specific shareable application.

For example, a recipe shareable application could send a "recipe viewed" event report and later get reports based on that event. Both Interaction reports and Custom Event reports can be seen on a per-application basis under the Track view in the Distribution Manager. The API calls are tied to a specific shareable application by setting your Partner ID and your shareable application's CID when using the services API.

The reports service is only applicable to shareable applications and is supported only in ActionScript.

Tracking custom events

Event reporting is used to count the occurrence of different application generated events during the runtime of the shareable application. The shareable application developer can identify any number of interesting events that occur during the lifetime of the shareable application and call the `reportEvent()` method to report them to the analytics service. Those event reports will be aggregated and made available in the Track view of the Distribution Manager. For example, a recipe shareable application could send a "recipe viewed" event report and later get reports on how many recipes have been viewed, and those reports could also be correlated with other tracking elements such as shareable application installs so it will be possible to calculate the ratio between installs and recipe views to estimate the effectiveness of the shareable application.

reports.reportEvent

This method is used for tracking of any kind of events relevant to the shareable application. For example, a video shareable application may want to count how many times the video was played and how many users actually watched it to the end, for that the shareable application could report a "video played" event when the video starts playing and a "video finished" when the video reaches the end. Event names may be any string and they do not require any kind of registration prior to sending. Adobe recommends that you keep the number of events to a reasonably small number so it will be possible to analyze them later in the reports.

Syntax:

```
distribution.services.reports.reportEvent(conf,params)
```

`Params` object members

Name	Type	Description
eventName (required)	string	A callback function that will be called with the results of the method when it completes.
cid	string	A Content ID associated with this shareable application.
callback	function	A callback function that will be called with the results of the method when it completes.
context	object	A context object that will be passed back unmodified to the application in the result object passed to the callback function.

See “Making API calls” on page 51 for details on how to create the `conf` and `params` objects.

Response data:

This method doesn't return any data.

Tracking end user interactions

End user interactions can be tracked in the following ways using the reporting service.

Interacted views counting Interacted views are defined as loads of the shareable application that had any kind of user interaction. User interaction is defined as a click or mouse hover for over 5 seconds anywhere on the shareable application. Interacted views are automatically counted after calling the `startTracking()` method.

Exposure time tracking Exposure time is the total time the shareable application was loaded on a web page (whether visible or not). Exposure time is measured automatically after calling the `startTracking()` method.

Interaction time Interaction time refers to the time users actually spend interacting with the shareable application. Interaction time can't be measured automatically because only the application knows what should be counted as interaction time. For example, a music player shareable application may want to count the time a song is being played as interacted time even if the shareable application wasn't visible the entire time. Use `interactionStarted()` and `interactionStopped()` to indicate when interaction time measurements should start and stop.

reports.interactionStarted

Indicates that user interaction with the shareable application has started and notifies Distribution Service to start measuring the interaction time. Multiple consecutive calls to this method are allowed and ignored. When the interaction ends the shareable application must call `interactionStopped()` to stop the interaction time measurements.

Syntax:

```
distribution.services.reports.interactionStarted(conf,params)
```

`Params` object members

Name	Type	Description
callback (optional)	function	A callback function that will be called with the results of the method when it completes.
context (optional)	object	A context object that will be passed back unmodified to the application in the result object passed to the callback function.
cid	string	a Content ID associated with this shareable application.

See "Making API calls" on page 51 for details on how to create the `conf` and `params` objects.

Response data:

This method doesn't return any data.

reports.interactionStopped

Indicates that user interaction has stopped and stops the measurement timer. Multiple consecutive stops are allowed and ignored.

Syntax:

```
distribution.services.reports.interactionStopped(conf,params)
```

`Params` object members

Name	Type	Description
callback (optional)	function	A callback function that will be called with the results of the method when it completes.
context	object	A context object that will be passed back unmodified to the application in the result object passed to the callback function.
cid	string	A Content ID associated with this shareable application.

See "Making API calls" on page 51 for details on how to create the `conf` and `params` objects.

Response data:

This method doesn't return any data.

reports.startTracking

Sets up mouse event handlers to detect user interaction with the shareable application to identify if this is an interacted view and measure exposure time for the shareable application. (unless disable via the reportExposureTime member of the params parameter).

Syntax:

```
distribution.services.reports.startTracking(conf,params)
```

`Params` object members

Name	Type	Description
callback (optional)	function	A callback function that will be called with the results of the method when it completes.
context	object	A context object that will be passed back unmodified to the application in the result object passed to the callback function.
cid	string	A Content ID associated with this shareable application.
reportExposureTime	Boolean	Indicates if exposure time should be measured for the shareable application. Default is True.

See "Making API calls" on page 51 for details on how to create the `conf` and `params` objects.

Response data:

This method doesn't return any data.

Using the Environment service to customize your content

Environment services enable shareable application developers to get information about the environment the shareable application resides in and the user that interacts with the shareable application, such as:

- Retrieving Geographic information about the user
- Getting data about the domain and URL the shareable application is in
- Indicating the network or platform the shareable application is in (Facebook, Blogger, etc.)

There are several preliminary steps you need to take to use a service. See "Loading the Distribution Service" on page 48 and "Using Distribution Service API methods" on page 51 for more information.

The Environment API consists of the following methods:

- env.Init
- env.getGeo
- env.getLoc
- env.isValidURL

env.Init

Initializes the Distribution Service environmental service. Must be called before any other environmental function.

Syntax:

```
distribution.services.env.init(conf,params)
```

`Params` object members

Name	Type	Description
callback (optional)	function	A callback function that will be called with the results of the method when it completes.
context	object	A context object that will be passed back unmodified to the application in the result object passed to the callback function.

See "Making API calls" on page 51 for details on how to create the `conf` and `params` objects.

Response data:

This method doesn't return any data.

Errors:

Code Error

OK The operation completed successfully.

Getting geographic information

env.getGeo

This method is used to retrieve geographic information on the location the user is browsing from.

Notes:

1 You must first call `distribution.services.env.init` and wait for its callback to be called before you may call getGeo.

2 Unlike most of the calls in the Distribution Service API, in addition to asynchronously calling the callback with the response object, this method also returns the response object synchronously when called.

Syntax:

```
distribution.services.env.getGeo(conf,params)
```

`Params` object members

Name	Type	Description
callback (optional)	function	A callback function that will be called with the results of the method when it completes.
context	object	A context object that will be passed back unmodified to the application in the result object passed to the callback function.
cid	string	A Content ID associated with this shareable application

See "Making API calls" on page 51 for details on how to create the `conf` and `params` objects.

Response data:

countryCode ISO 3166-1_alpha-2 country code for the country the current user is browsing from.

countryName The name of the country the current user is browsing from.

regionCode ISO 3166-2 country code for the region the current user is browsing from if it is part of the US or Canada. If the user is outside US or Canada then a FIBS 10-4 region code is returned instead.

regionName The name of the region the current user is browsing from.

city The name of the city the current user is browsing from.

For all the above fields, unknown values are returned as empty strings.

Getting the page URL

env.getLoc

This method is used to retrieve information on the location the shareable application is currently running from.

Notes:

1 You must first call distribution.services.env.init and wait for its callback to be called before you may call getLoc.

2 Unlike most of the calls in the Distribution Service API, in addition to asynchronously calling the callback with the response object, this method also returns the response object synchronously when called.

3 For proper detection of the network you should set advancedTracking to "True" when using the Share menu to distribute your application.

Syntax:

```
distribution.services.env.getLoc(conf,params)
```

`Params` object members

Name	Type	Description
callback (optional)	function	A callback function that will be called with the results of the method when it completes.
context	object	A context object that will be passed back unmodified to the application in the result object passed to the callback function.
cid	string	A Content ID associated with this shareable application

See "Making API calls" on page 51 for details on how to create the `conf` and `params` objects.

Response data:

network The name of the network in which the shareable application is currently running. This can be any network name of the networks supported by the Share menu. It will usually be available only for shareable applications posted using the Share menu.

URL The URL of the page in which the shareable application is currently running up to the first "?".

domain The domain name of the page in which the shareable application is currently running. For domains that end with two letters (e.g. .au, .jp, etc.) it will return the last 3 segments (X.Y.Z.au => y.z.au) otherwise it will return only the last 2 segments (X.Y.com => Y.com).

For all the above fields, unknown values are returned as empty strings.

Validating URLs against a blacklist

env.isValidURL

This method checks whether or not the URL of the page which is displaying the shareable application, is on a blacklist. If the URL of the page is on the blacklist (exact match), the result will be 'false'.

Notes:

1 If the params object does NOT contain a blacklistURL parameter, the result will 'true'.

2 If the blacklistURL parameter in the params object is an empty string, the result will 'true'.

3 If the blacklistURL parameter in the params object contains a URL that cannot be accessed, the result will 'true'.

4 If the blacklistURL parameter in the params object contains a non-URL string, the result will 'true'.

5 You must first call distribution.services.env.init and wait for its callback to be called before you may call isValidURL.

Syntax:

```
distribution.services.env.isValidURL(conf,params)
```

`Params` object members

Name	Type	Description
blacklistURL (required)	string	A URL of the black list file. e.g: http://www.myDomain.com/myBlacklist.txt. The blacklist file should contain one URL per line.
callback (optional)	function	A callback function that will be called with the results of the method when it completes.
context	object	A context object that will be passed back unmodified to the application in the result object passed to the callback function.
cid	string	A Content ID associated with this shareable application

See "Making API calls" on page 51 for details on how to create the `conf` and `params` objects.

Response data:

isValidURL True if URL of the page which is displaying the shareable application is a valid one.

blacklistURL The URL of the black list file as passed to this method.

yourURL The URL of the page which is displaying the shareable application.

matchedURL The matched URL in the black list file. - i.e: The URL that caused the isValidURL to be false.

matchedLine The line number in the black list file that contains the matched URL.

accessError In case of an error, this value will contain the error details if available. In any other scenario, this value will be an empty string.

requestParams The params object as passed to this method.

For all the above fields, unknown values are returned as empty strings.

Include the Share menu in applications

You can configure and embed the Share menu in applications in two ways: using Adobe's extensions, or programmatically using APIs.

For information about using the Adobe extensions, see "Using Dreamweaver to share applications and web pages" on page 37, "Using Flex Builder 3 to create shareable applications" on page 29, and "Using Flash Professional to create shareable applications" on page 16. The following topics contain information about configuring and embedding shareable applications programmatically.

Include the Share menu without using Distribution Service extensions

If you choose not to install or use the Distribution Service Flash, Flex or Dreamweaver extensions, you can still include the Share menu in your applications, customize your applications, and embed your applications in a web page. The following code and explanations describe how to do this in Flash for ActionScript 2 and ActionScript 3, and in Flex for ActionScript 3.

Flash ActionScript 2

The following code example shows how to add the Share menu to your application in Flash ActionScript 2 without using the Distribution Service extension.

```
// Step 1 - Import required libraries:
import flash.Security;
class Main
{
    public static function main():Void
    {

        //--- The following code configures and loads the Share menu---
        // Step 2 - Set up security to allow your shared app to interact with the Share menu
        System.security.allowDomain("cdn.gigya.com");
        // prevent creation of multiple instances of the Share menu
        if (_root.Wildfire !=undefined ) {
            _root.Wildfire._visible=true;
            return;
        }
        // Step 3 - Create an empty movie clip to host the Share menu user interface
        var mcWF:MovieClip=_root.createEmptyMovieClip
        ('Wildfire',_root.getNextHighestDepth());
        mcWF._lockroot=true; //lock the root of the newly created movieclip

        // Step 4 - Position the Share menu in your Flash application
        mcWF._x=15;
        mcWF._y=42;
        // Step 5 - Create a configuration object through which the Share menu will
        communicate with the host swf:
        mcWF['ModuleID']='PostModule1';        // passing the module id to the Share menu
        var cfg=_root[mcWF['ModuleID']]={};     // initializing the configuration object
        // Step 6 - Setting the configuration parameters (based on your selections
        in the wizard):
        cfg['width']='200';
        cfg['height']='250';
        cfg['partner']='000'; //Set your partner ID (will automatically be assigned
        by the wizard)
        cfg['UIConfig']='<config><display showEmail="true"
        showBookmark="true"></display></config>';
        // Step 7 - Set up the content to be posted
        cfg['defaultContent']= 'Hello world! (Post content)'; // <-- YOUR EMBED CODE
        GOES HERE
        // Step 8 - Set up an event handler for the onClose event. This is called
        when the Share menu is closed.
        cfg['onClose']=function(eventObj){
            mcWF._visible = false;
            mcWF.INIT();
            //you can do additional cleanup here
        }
        // Step 9 - Load the Share menu
        mcWF.loadMovie('http://cdn.gigya.com/WildFire/swf/wildfire.swf','get');

    }
}
```

Explanation of code

In the example above, the wrapping ActionScript 2 code basically defines a "Main" class and a `main()` method. The code from step 2 on initializes and displays the Share menu. The best way to use it is to wrap it inside a method and call it based on user interaction that should trigger the display of the Wildfire user interface. The following is an explanation of the nine steps taken in the code above. This is useful for advanced integration scenarios, or if you want to insert manual configuration changes.

1 Import required libraries. The Share menu requires that you import some ActionScript libraries. This line should be placed inside your main timeline or before your class declaration:

```
 import flash.Security;
```

2 Set up security to allow your application to interact with the Share menu. The following lines must be executed before you load the Share menu:

```
System.security.allowDomain("cdn.gigya.com");
```

3 Because your application loads from a different domain than that of the Share menu, you have to explicitly allow the two to interact.

4 Create a container to host the Wildfire user interface--called the Share menu in Project Radiate. The Share menu requires the hosting application to provide a uniquely identified container where the Share menu will be drawn. For this purpose, create an empty movie clip to hold the Share menu user interface:

```
 var mcWF:MovieClip=_root.createEmptyMovieClip('Wildfire',_root.getNextHighestDepth());
```

5 You may need to adjust the creation method depending on the environment you use. For example, if you are using V2 components you use:

```
 var mcWF:MovieClip = _root.createChildAtDepth("EmptyMovieClip", DepthManager.kTop);
mcWF._name='Wildfire';
```

6 Position the new movie clip. To make sure the user interface appears exactly where you want it to appear within your application, position the host container:

```
mcWF._x=15;
mcWF._y=42;
```

7 Create a configuration object. Create an empty configuration object from which the Share menu will read its configuration.

```
 mcWF['ModuleID']='PostModule1';        // passing the module ID to the Share menu
var cfg=_root[mcWF['ModuleID']]={};     // initializing the configuration object
```

8 When the Share menu loads, it expects to find a "ModuleID" parameter, the value of this parameter is used to get to a configuration object in the hosting SWF file. Actual configuration of the Share menu is done by assigning values to different attributes of this configuration object.

9 Set the configuration parameters and user interface design. You may manually change the values of these parameters and many more parameters which the Share menu supports. See “Configuration parameters reference tables” on page 67 for information about all supported parameters.

10 Set the content of the post. Place your content to post here. Depending on your application you can choose to set this as a literal string (example:

```
cfg['defaultContent']= 'Hello world! (Post content)'; // <-- YOUR EMBED CODE GOES HERE
```

11 Or, if your application can generate dynamic code based on user interaction (e.g. selecting a color) you can use a function instead of a string. The Share menu calls the function when it needs the content:

```
cfg['defaultContent']= function() { return 'Hello world!'; }
```

12 Setting up an event handler. This is an optional step. See “Share Menu” on page 79 to learn more about the events that the Share menu generates.

13 Load the Share menu. Include the Share menu in your application by calling `loadMovie` on the container movie clip you have created in the previous steps:

```
mcWF.loadMovie('http://cdn.gigya.com/WildFire/swf/wildfire.swf','get');
```

Flash ActionScript 3

The following code example shows how to add the Share menu to your application in Flash ActionScript 3 without using the Distribution Service extension.

```
package
{
   // Step 1 - Import required libraries:
   import flash.system.Security;
   import flash.external.*;
   import flash.display.Loader;
   import flash.net.URLRequest;
   import flash.display.MovieClip;
   public class Main extends flash.display.Sprite
   {
      public function Main():void
      {
         //--- The following code configures and loads the Share menu---

         // Step 2 - Set up security to allow your app to interact with the Share menu
         Security.allowDomain("cdn.gigya.com");
         // Step 3 - This code creates an empty movie clip to host the Share menu
        interface
         var mcWF:MovieClip = new MovieClip();
         addChild(mcWF).name='mcWF';
         // Step 4 - Position the Share menu in your Flash application
         mcWF.x=15;
         mcWF.y=42;
         // Step 5 - This code creates a configuration object through which the
        Share menu will communicate with the host swf
         var ModuleID:String='PostModule1'; // pass the module id to the Share menu
         var cfg:Object = { }; // initialize the configuration object
         // Step 6 - This code assigns the configurations you set in our site to
        the Share menu configuration object
         cfg['width']=200;
```

```
            cfg['height']=250;
            cfg['UIConfig']= '<config><display showEmail="true"
            showBookmark="true" /></body></config>';
            cfg['partner']='0000'; //Your partner ID will automatically be assigned by
            the wizard
            // Step 7 - Set up the content to be posted
            cfg['defaultContent']= ''; // <-- YOUR EMBED CODE GOES HERE

            // Step 8 - This code calls up the Share menu
            var ldr:Loader = new Loader();
            var url:String =
           'http://cdn.gigya.com/Wildfire/swf/WildfireInAS3.swf?ModuleID=' + ModuleID;
            var urlReq:URLRequest = new URLRequest(url);
            mcWF[ModuleID] = cfg;
            ldr.load(urlReq);
            mcWF.addChild(ldr);

        }
    }
}
```

Explanation of code

In the example, the wrapping ActionScript 3 code defines a "Main" class and a main() method. The code to initialize and display the Share menu is then pasted into the main() method. This example illustrates the best way to use the code that initializes and displays the Share menu: wrap it inside a method and call it based on user interaction that should trigger the display of the Share menu. The following is an explanation of the eight steps taken in the code above.

1 Import required libraries. The Share menu requires that you import some ActionScript libraries. These lines should be placed inside your main timeline or before your class declaration.

```
import flash.system.Security;
import flash.external.*;
import flash.display.Loader;
import flash.net.URLRequest;
import flash.display.MovieClip;
```

2 Set up security to allow your application to interact with the Share menu. These lines must be executed before you load the Share menu. Because your application loads from a different domain than that of the Share menu, you have to explicitly allow the two to interact.

```
Security.allowDomain("cdn.gigya.com");
```

3 Create a container to host the Share menu. The Share menu requires the hosting application to provide a uniquely identified container where the Share menu’s user interface will be drawn. For this purpose, create an empty movie clip to hold the Share menu:

```
var mcWF:MovieClip = new MovieClip();
addChild(mcWF).name='mcWF';
```

4 Position the new movie clip. To make sure the Share menu appears exactly where you want it to appear within your application, position the host container:

```
mcWF._x=15;
mcWF._y=42;
```

5 Create a configuration object. Create an empty configuration object from which the Share menu will read its configuration. When the Share menu loads, it expects to find a "ModuleID" parameter. The value of this parameter is used to get to a configuration object in the hosting SWF file. Actual configuration of the Share menu is done by assigning values to different attributes of this configuration object:

```
var ModuleID:String='PostModule1';// pass the module id to the Share menu
var cfg:Object = { };// initialize the configuration object
```

6 Set the configuration parameters and user interface design. The values of these parameters are based on your selections in the wizard. You may manually change the values of these parameters and many more parameters that the Share menu supports. The example sets the size of the Share menu:

```
cfg['width']='200';
cfg['height']='250';
```

Set your partner ID. You can find your partner ID using the Distribution Manager. See "About Partner IDs" on page 4 for information.

```
cfg['partner']='000'; //Set your partner ID
```

Set the user interface configuration parameters: The UIConfig parameter defines the user interface design of the Share menu component. It is assigned using an XML structure that is generated by the Project Radiate extensions, according to your selected user interface design.

```
cfg['UIConfig']='<config><display showEmail="true" showBookmark="true"
></display></config>';
```

7 Set the content of the post. Depending on your application you can choose to set this as a literal string as shown in the example:

```
cfg['defaultContent']= 'Hello world! (Post content)'; // <-- YOUR EMBED CODE GOES HERE
```

Or, if your application can generate dynamic code based on user interaction (e.g. selecting a color) you can use a function instead of a string. The Share menu will call the function when it needs the content.

```
cfg['defaultContent']= function() { return 'Hello world!';}
```

8 Load the Share menu. The following code loads the Share menu using the movie clip ("mcWF") that was created in step 3:

```
var ldr:Loader = new Loader();
var url:String = 'http://cdn.gigya.com/Wildfire/swf/WildfireInAS3.swf?ModuleID=
' + ModuleID;
var urlReq:URLRequest = new URLRequest(url);
mcWF[ModuleID] = cfg;
ldr.load(urlReq);
mcWF.addChild(ldr);
```

Note that the parameter "ModuleID" is pointing to the configuration object "cfg" which was prepared in previous steps.

Flex ActionScript 3

The following code example shows how to add the Share menu to your application in Flex ActionScript 3 without using the Distribution Manager extension.

```
<?xml version="1.0"?>
<mx:Application xmlns:mx="http://www.adobe.com/2006/mxml"
applicationComplete="WFInit()" layout="absolute" verticalScrollPolicy="off"
horizontalScrollPolicy="off" >

    <!-- Step 1 - Define a component to be used to load the Share menu:  -->
    <mx:Box id="wfLoaderBox" x="30" y="70" visible="false" />

    <mx:Script>
        <![CDATA[
            import mx.events.DynamicEvent;

            // Step 2 - Create a configuration object through which the Share menu will
            communicate with the host swf:
            public var cfg:Object = { };
            public function WFInit():void {

                // Step 3 - Set up security to allow your application to interact
                with the Share menu:
                Security.allowDomain("cdn.gigya.com");

                if(wfLoaderBox.numChildren > 0 ){
                    wfLoaderBox.visible = true;
                    return;
                }

                // Step 4 - Setting the configuration parameters:
                cfg['width']='200';
                cfg['height']='250';
                cfg['partner'] = '0000';
                cfg['UIConfig']='<config><display showEmail="true"
                showBookmark="true" ></display></config>';

                // Step 5 - Set up the content to be posted
                cfg['defaultContent']= 'Hello world! (Post content)';
                // <-- YOUR EMBED CODE GOES HERE

                // Step 6 - Set up an event handler for the onClose event. This is
                called when the Share menu is closed.
                cfg['onClose']=function(eventObj:Object):void{
                    //add here code to hide
                    wfLoaderBox.visible = false;
                    MovieClip(Loader(mx.core.UIComponent(wfLoaderBox.getChildAt(0))
                    .getChildAt(0)).content).INIT();
                }

                var wfLoader:flash.display.Loader;
                wfLoader=new flash.display.Loader();
                // Step 7 - Load the Share menu or re-initialize if already loaded:
                if (wfLoaderBox.numChildren == 0 ) {
                    wfLoader.contentLoaderInfo.sharedEvents.addEventListener("sendConfig",
                        function(e:Event):void {
                            var evtObj:mx.events.DynamicEvent = new
                        mx.events.DynamicEvent('onStoreConfig');
                            evtObj.cfg = cfg;
                            wfLoader.contentLoaderInfo.sharedEvents.dispatchEvent(evtObj);
                        }
```

```
                )

                wfLoader.contentLoaderInfo.addEventListener(Event.COMPLETE,
                    function(event:Event):void{
                        var uic:mx.core.UIComponent = new mx.core.UIComponent();
                        wfLoaderBox.addChild(uic);
                        uic.addChild(wfLoader);
                    }
                );

                // The following method Loads the Share menu
                wfLoader.load(new
    URLRequest('http://cdn.gigya.com/wildfire/swf/wildfireInAS3.swf?ModuleID=cfg'));
            }
            else {
                MovieClip(Loader(
    mx.core.UIComponent(wfLoaderBox.getChildAt(0)).getChildAt(0)).content).INIT();
            }
        }

    ]]>
    </mx:Script>
     <!-- Step 8 - call WFInit():  -->
    <mx:Button label="Load Wildfire" x="30" y="30" click="WFInit()"/>

</mx:Application>
```

Explanation of code

Note: *The following is an explanation of the eight steps taken in the code above, giving a deeper understanding of the content of the code. This is only required for advanced integration scenarios, or if you want to insert manual configuration changes.*

1 Create a container to host the Share menu’s user interface. The Share menu requires the hosting application to provide a uniquely identified container where the Share menu’s user interface will be drawn.

```
<mx:Box id="wfLoaderBox" x="30" y="70" visible="false" />
```

The "wfLoaderBox" component is used to load the Share menu. Modify the x and y attributes to set the component's position using absolute layout. Alternatively you may use layout elements such as Flex's VBox and HBox to control the position of the host container.

2 Create a configuration object (line 12): Create an empty configuration object from which the Share menu will read its configuration. The name of this variable ("cfg") is significant in that the variable must be declared to be public:

```
public var cfg:Object = { };
```

When the Share menu loads, it expects to find a "ModuleID" parameter. The value of this parameter is used to get to a configuration object in the hosting SWF file. Actual configuration of the Share menu is done by assigning values to different attributes of this configuration object. The value of the ModuleID parameter is the actual name of the configuration variable:

```
wfLoader.load(new URLRequest
('http://cdn.gigya.com/wildfire/swf/wildfireInAS3.swf?ModuleID=cfg'));
```

3 Set up security to allow your application to interact with the Share menu. These lines must be executed before you load the Share menu:

```
Security.allowDomain("cdn.gigya.com");
```

Because your application loads from a different domain than that of the Share menu, you have to explicitly allow the two to interact.

4 Set the configuration parameters and user interface design. The values of these parameters are based on your selections in the Project Radiate authoring extension. You can manually change the values of these parameters and many more parameters that the Share menu supports. This code sets the size of the Share menu:

```
cfg['width']='200';
cfg['height']='250';
```

Set your partner ID. You can find your partner ID using the Distribution Manager. See "About Partner IDs" on page 4 for information.

```
cfg['partner']='000'; //Set your partner ID
```

Set the user interface configuration parameters. The UIConfig parameter defines the user interface design of the Share component. It is assigned using an XML structure generated by the Project Radiate extensions, according to your selected user interface design.

```
cfg['UIConfig']='<config><display showEmail="true"
showBookmark="true" ></display></config>';
```

5 Set the content of the post. Depending on your application you can choose to either set this as a literal string like in the example:

```
cfg['defaultContent']='Hello world! (Post content';//<--YOUR EMBED CODE GOES HERE.
```

If your application can generate dynamic code based on user interaction (e.g. selecting a color) you can use a function instead of a string. The Share menu calls the function when it needs the content.

```
cfg['defaultContent']= function() { return 'Hello world!'; }
```

6 Set up an event handler. This is an optional step. See "Share Menu" on page 79 for more information.

learn more about the events that the Share menu generates.

7 Load the Share menu or re-initialize if already loaded. The following method loads the Share menu:

```
wfLoader.load(new
URLRequest('http://cdn.gigya.com/wildfire/swf/wildfireInAS3.swf?ModuleID=cfg'));
```

Note that the parameter "ModuleID" is pointing to the configuration object "cfg" that was prepared in previous steps.

8 Call WFInit(). The "WFInit()" method should be called when you want the Share menu to load. The example defines a button that calls the "WFInit()" method.

Customize applications and share menus

You can customize the share menu in shareable applications in two ways: using Adobe's extensions, or programmatically using APIs developed by Gigya or by Adobe.

Customize applications and Share menus with Adobe extensions

See "Creating and publishing shareable applications" on page 18 (Flash) and "Workflow for creating and publishing applications" on page 30 (Flex) for information about using the Adobe extensions to customize applications and Share menus.

Customize applications and Share menus without Adobe extensions

To configure applications and their Share menus without using Adobe's Flash, Flex, or Dreamweaver extensions, you can use Distribution Service APIs.

Configure the Share menu programmatically

Every parameter has a default value which can be overridden by setting its value in the configuration object. Below is an example of creating a configuration object named "conf" and setting one parameter (the defaultContent parameter):

ActionScript 2

```
mcWF['ModuleID']='PostModule1';        // passing the module id to the Share menu
var conf=_root[mcWF['ModuleID']]={};     // initializing the configuration object

conf['defaultContent']= 'POST CONTENT';
```

ActionScript 3

```
var ModuleID:String='PostModule1';// pass the module id to the Share menu
var conf:Object = { };// initialize the configuration object

conf['defaultContent']= 'POST CONTENT';
```

Flex

```
public var conf:Object = { };

conf['defaultContent']= 'POST CONTENT';
```

This section specifies all of the parameters supported by the Wildfire technology that powers the Share menu. The tables that follow provide brief reference information about each parameter, with links to more complete information about each.

Configuration parameters reference tables

The following tables list all of the available configuration parameters for the Share menu.

Basic parameters

Parameter Name	Type	Description
partner	string	The partner ID
width	number	The width in pixels of the Share menu
height	number	The height in pixels of the Share menu
CID	string	The content ID of the shareable application.

Partner The partner ID. This ID can be found in the "Account Settings" page on the Gigya website.

Width The width in pixels of the Share menu.

Height The height in pixels of the Share menu.

CID The content ID of the shareable application. The CID is used to uniquely identify the shareable application and can be made up of two parts: "Code ID" + "/" + "Config ID". If no "Config ID" is used, then the CID should simply be set to the "Code ID".

User interface Parameters

Parameter Name	Type	Description
UIConfig	string	An XML string that provide detailed control over the appearance of the Share menu.
mixNetworks	Boolean	When set to True, the Share menu displays all the icons of "Post", "Bookmark" and "Desktop" networks on a single tab.
lang	string	Determine the language that the Share menu will render.
localConfig	string	An XML string for setting user interface configuration, which overrides the configuration set by the UIConfig parameter.
networksToShow, networksToHide	string	This pair of parameters allow you to set which social network buttons are displayed on Share menu's "Post" tab, and also determine the order.
bookmarksToShow, bookmarksToHide	string	This pair of parameters allow you to set which social-bookmarking-sites buttons are displayed on "Bookmark" tab, and also determine the order.
customNetworks	string	This parameter allows you to add new custom network buttons to the Share menu.
emailImportProvider s	string	This parameter lets you select what e-mail providers will appear in the import dropdown of the Share menu's e-mail import section.
nowmode	Boolean	Assigning "true" to this parameter, forces the Share menu not to use the wmode parameter.
outsideColor	string	The color of the area outside rounded corners.

UiConfig The Share menu uses a hierarchical XML string to allow a flexible and highly configurable design. You can control fonts, colors, sizes, gradients, frames, roundness of corners for almost any graphic element in the Share menu's user interface by overriding the default settings. Below is a list of all of the UIConfig categories and their possible parameters and settings.

Note: *You can customize the Share menu by editing specific parameters of UIConfig, or you can generate and customize the contents of UIConfig using the Distribution Service Flash and Flex extensions. However, you cannot do both. Using the extensions resets all parameters to their defaults, and saves changes only for the parameters you edit using the extension's panels. For example, if you hand-edited the color of a textbox using the Component Inspector, and then later used the extension to set some other parameters, the change you made by hand to the textbox's color is overwritten by the extension, unless you explicitly make the same change via the extension.*

Category	**Parameter (format)**
display	codeBoxHeight (Boolean) networksWithCodeBox (Boolean) showBookmarks (Boolean) showCloseButton (Boolean) showCodeBox (Boolean) showEmail (Boolean) useTransitions (Boolean)
background	background-color (string) corner-roundness (pt;pt;pt;pt) frame-color (string) gradient-color-begin (string) gradient-color-end (string)
controls	bold (Boolean) color (string) corner-roundness (string) gradient-color-begin (string) gradient-color-end (string)
snbuttons	bold (Boolean) color (string) down-color (string) down-frame-color (string) down-gradient-color-begin (string) down-gradient-color-end (string) frame-color (string) gradient-color-begin (string) gradient-color-end (string) over-bold (Boolean) over-color (string) over-frame-color (string) size (pt) type (textunder)
more	frame-color (string) gradient-color-begin (string) gradient-color-end (string) over-frame-color (string) over-gradient-color-begin (string) over-gradient-color-end (string)

Category	Parameter (format)
previous	frame-color (string) gradient-color-begin (string) gradient-color-end (string) over-frame-color (string) over-gradient-color-begin (string) over-gradient-color-end (string)
textboxes	color (string) frame-color (string) gradient-color-begin (string) gradient-color-end (string)
codeboxes	background-color (string) color (string) frame-color (string) gradient-color-begin (string) gradient-color-end (string) size (pts-numerals)
inputs	color (string) frame-color (string) gradient-color-begin (string) gradient-color-end (string)
dropdowns	frame-color (string) list-item-over-color (string)
buttons	bold (Boolean) color (string) down-gradient-color-end (string) frame-color (string) gradient-color-begin (string) gradient-color-end (string) over-frame-color (string) over-gradient-color-begin (string) over-gradient-cover-end (string)
post-buttons	frame-color (string) gradient-color-begin (string) gradient-color-end (string)
list-boxes	frame-color (string) corner-roundness (pt;pt;pt;pt) gradient-color-begin (string) gradient-color-end (string)

Category	Parameter (format)
checkboxes	checkmark-color (string) corner-roundness (pt;pt;pt;pt) frame-color (string) gradient-color-begin (string) gradient-color-end (string)
servicemarker	gradient-color-begin (string) gradient-color-end (string)
tooltips	color (string) frame-color (string) gradient-color-begin (string) gradient-color-end (string) size (pts-numeral)
texts	color (string)
headers	color (string)
messages	color (string)
links	color (string) underline (Boolean) over-color (string) down-color (string) down-bold (Boolean)

mixNetworks Boolean attribute, part of the "display" element of the UIConfig. When mixNetworks is set to "true" the Share menu will display all the icons of both "Post", "Bookmark" and "Desktop" networks on a single tab. The single tab would have a "Post" title. By default, the icons are presented in three different tabs: "Post", "Bookmark" and "Desktop". Setting mixNetworks:

```
UIConfig: '<config><display mixNetworks="true" ></display></config>'
```

lang By assigning a two-character language code to this parameter the Share menu can render in that language. English ("en") is the default if no other language is specified or the specified code is not supported.

Currently Supported Languages

Language	Language Code
English	en
Chinese (Hong Kong)	zh-hk
Chinese (Taiwan)	zh-tw
Czech	cs
Danish	da
Dutch	nl
Finnish	fi

Language	Language Code
French	fr
German	de
Greek	el
Hungarian	hu
Italian	it
Japanese	ja
Korean	ko
Norwegian	no
Polish	pl
Portuguese	pt
Portuguese (Brazil)	pt-br
Russian	ru
Spanish	es
Spanish (Mexican)	es-mx
Swedish	sv

Note: If you are using ActionScript 2 you have to manually load the language-specific version of wildfire.swf file. To do that, append "_" and the language code to the Wildfire SWF file name. For example, to load the Mexican Spanish version of the Share menu in ActionScript 2 change the line:

```
mcWF.loadMovie('http://cdn.gigya.com/WildFire/swf/wildfire.swf','get');
```

to:

```
mcWF.loadMovie('http://cdn.gigya.com/WildFire/swf/wildfire_es-mx.swf','get');
```

localConfig An XML string for setting user interface configuration, which overrides the configuration set by the UIConfig parameter. This can be helpful when building a complex user interface configuration that has to be slightly different in different situations. The localConfig parameter allows setting only the differences while the main configuration remains unchanged.

networksToShow, networksToHide This pair of parameters allow you to set which social network buttons are displayed on the Share menu's "Post" tab, and also to determine the order. Each parameter is set with a comma-separated string of network names. Valid network names include: myspace, friendster, facebook, orkut, bebo, tagged, wordpress, blogger, hi5, livespaces, piczo, freewebs, livejournal, blackplanet, myyearbook, typepad, xanga, multiply, igoogle, netvibes, pageflakes, migente, vox, yahoo, ameba, livedoor and eons. In addition networksToShow can include a * to indicate any other network supported by the Share menu. For example: if you specify "myspace,facebook,*" as the value of networksToShow and "hi5,vox" as the value of networksToHide you will end up with myspace and facebook first, followed by all the other supported networks with the exception of hi5 and vox.

Note - If the current user has already posted to a certain social network, this social network will be shown first (prior to the list defined in networksToShow).

bookmarksToShow, bookmarksToHide This pair of parameters allow you to set which social-bookmarking-sites buttons are displayed on the Share menu's "Bookmark" tab, and also determine the order. Each parameter is set with a comma separated string of network names. These pairs of parameters have similar logic to the networksToShow and networksToHide pair of parameters. Valid bookmark networks include: delicious, digg, twitter, friendfeed, googlebookmarks, myaol, aolbookmarks, stumbleupon, facebookshare, linkedin, yahoobookmarks, yahoomyweb, Iivebookmarks, technorati, slashdot, reddit, spurl, furl, faves, newsvine, fark, mixx, misterwong, ask, wikio, propeller, currenttv, blogmarks, simpy, sardbarker, slinklist, diigo, sphinn, backflip, dropjack, segnalo, linkagogo, kaboodle, feedmelinks, buzzup, netvouz, rawsugar, magnolia, tailrank.

If the current user have already bookmarked using certain social-bookmarking-sites, this social-bookmarking-sites will be shown first (prior to the list defined in bookmarksToShow).

customNetworks This parameter allows you to add new custom network buttons to the Share menu. When the user presses your custom button the regular NetworkButtonClicked Event will be fired. Defining Custom Network Button: The custom network buttons are defined using an XML string, in the following format:

```
customNetworks: '<nets><net id="customnet1" name="Network 1 Display Name"
iconURL="http://www.quarktet.com/Icon-small.jpg" service="POST"
/><net id="customnet2" name="Network 2 Display Name"
iconURL="http://www.quarktet.com/Icon-small.jpg" service="BOOKMARK"
/></nets>'
```

Where:
`<net />` defines a single network button. Each button is defined using the following properties:

ID - Identifies the network button. The new button ID, may be used in: networksToShow, networksToHide, bookmarksToShow, bookmarksToHide parameters.

Name - The caption of the button

Icon URL - URL to the icon to be displayed on the button.

Service - Will determine on which tab to locate the button. Possible values: "POST", "BOOKMARK".

emailImportProviders This parameter lets you select what e-mail providers will appear in the import dropdown of the Share menu's e-mail import section. Use any of the names: manual, gmail, hotmail, aol, yahoomail or *.

nowmode A Boolean parameter. Assigning "true" to this parameter forces the Share menu not to use the wmode parameter. A bug in FireFox leads to a situation where when using wmode (which the Share menu uses by default for transparency) input from non-English US keyboards is interpreted as if it came from an English US keyboard. This means that for example a German keyboard that has the @ sign on the Alt-Gr+Q key combination will produce the @ sign only when shift+2 is pressed.The downside of using this setting is that you can not use transparent background. If you don’t need a transparent background you should set nowmode to "true". When setting nowmode to "true" you can and probably should also use outsideColor.

outsideColor When using rounded corners, you may use this parameter to set the color of the area outside the corners so that it matches the background of your page. The default is #ffffff (white).

Post to Social Network parameters

The following parameters are associated with the Post operation.

Parameter Name	Type	Description
defaultContent	string	The content to be posted.
<network-name>Content	string	The content to be posted to a specific social network.
defaultUsername	string	The default username on all the social networks.
<network-name>Username	string	The username on a specific social network.
<network-name>Sections	string	A comma separated list of sections to be displayed in the section selection drop-down for the specific network.
widgetTitle	string	The default title suggested to the user on networks that support such title.
defaultPreviewURL	string	URL of an image that will be used as the preview for your shareable application in all the networks that support such a preview.
<network-name>PreviewURL	string	URL of an image that will be used as the preview for your shareable application on a specific social network.
contentIsLayout	Boolean	Indicating if the content is a layout.
includeShareButton	Boolean	Assigning "true" to this parameter appends a "Share" button to your posted shareable application.
facebookURL	string	URL of your own Facebook application that the Share menu should use to post content.
useFacebookNewsfeed	Boolean	If this parameter is set to 'true', your shareable application will be posted to the "newsfeed" section in Facebook.

defaultContent The content to be posted. The content is a piece of HTML code or it may also be a text-area ID (in case of JavaScript). This content will be posted to the social network that was selected by the user.

<network-name>Content It is possible to set different content to post for each social network supported by the Share menu. For this purpose, there are various parameters with the same name format: <network-name> (Myspace, Facebook, Friendster, etc.) followed by "Content".

For example: myspaceContent - The content to be used specifically for myspace.

The content of <network-name>Content overrides, for the specific network, the content set in the defaultContent. In other words, defaultContent is the content to be used as the default for all social networks, while <network-name>Content is the content to be used for the specific network. If the content includes the string "$network$", then on automatic posts it will be replaced with the network's name or with the word "unknown" if the code has been copied from a codebox.

defaultUsername The default username on all the social networks.

<network-name>Username It is possible to specify a different username for each social network supported by the Share menu. For this purpose, there are various parameters with the same name format: <network-name> (Myspace, facebook, friendster, etc.) followed by "Username".

For example, myspaceUsername is the user-name to be used specifically for Myspace.

<network-name>Sections A comma separated list of sections to be displayed in the section selection drop-down for the specific network. Only valid section names for the specific network will be displayed. If the result of this is that only a single section is to be displayed then it will be used but the dropdown will not be displayed at all. It is possible to set different sections for each social network supported by the Share menu. For this purpose, there are various parameters with the same name format: <network-name>(Myspace,Facebook, Friendster, etc.) followed by "Sections".

For example: myspaceSections specifies a comma-separated list of sections to be displayed in the section selection drop-down specifically for myspace.

widgetTitle A string with maximum 50 characters, which will be used as the default title suggested to the user on networks that support titles.

defaultPreviewURL By default, the Share menu automatically creates a preview for your shareable application to be used in networks that require such preview (like Facebook). Using this parameter you can supply a URL of an image that will be used as the preview for your shareable application in all the networks that support such a preview.

<network-name>PreviewURL It is possible to set different preview images for each social network that supports previews. For this purpose, there are various parameters with the same name format: <network-name> (Facebook, Igoogle, etc.) followed by "PreviewURL" For example, facebookPreviewURL specifies a preview to be used specifically for facebook.

contentIsLayout A Boolean value indicating if the content is a layout. If set to 'true', there will be no sections' selection dropdown and the default section will be automatically decided. By default contentIsLayout value is "false". In this case, only sections which are not layout-related are displayed in the section's selection dropdown.

includeShareButton A Boolean that when set to 'true' causes a "Share" button to be appended to your posted application. Pressing a "Share" button sends the user to a page where he can share your application with his e-mail friends.

facebookURL By default the Share menu will automatically post your content to Facebook using a specially created application on the Facebook platform. This application will automatically handle everything for you, including the creation of images from your SWF file's content. However, if you want to override this behavior and open your own Facebook application you may use this parameter to provide the URL that the Share menu should open.

useFacebookNewsfeed A Boolean parameter. If this parameter is set to 'true', your shareable application will be posted to the "newsfeed" section in Facebook.

Note that Facebook resizes Flash content to fit the newsfeed and resizes preview images shown to users. The limits are: Feed story (used when posting shareable applications using the MyStuff application) - has max limit of 260X320 pixels for embed content and 130X100 pixels for preview image.Share - Facebook will scale every shareable application to be exactly 320 pixels, so if shareable application is smaller it actually scales up which may be bad (unless scaling is disabled). Preview image limit is 160 pixels wide. The solution is to allow scaling on shareable applications bigger than 320 width. Small shareable applications (under 320 pixel width) are recommended not to have scaling enabled so they will not be scaled up in share.

Social Bookmarking parameters

Parameter Name	Type	Description
defaultBookmarkURL	string	The URL that will be sent to all the social bookmarking sites.
<social-bookmarking-site-name>BookmarkURL	string	The URL that will be sent to the specific social bookmarking site.
widgetTitle	string	The default title suggested to the user on social bookmarking sites that support such titles.

defaultBookmarkURL When using the bookmarks feature, by default the URL of the current page is automatically bookmarked. This parameter allows you to specify an alternative URL that will be sent to the social bookmarking site. Note - If the page that hosts the Share menu is dynamic, Adobe recommends setting defaultBookmarkURL with link to your home page.

<social-bookmarking-site-name>BookmarkURL It is possible to set a different URL for each social bookmarking site supported by the Share menu. For this purpose, there are various parameters with the same name format: <social-bookmarking-site-name> (all lowercase letters only: digg, twitter, etc.) followed by "BookmarkURL".For example, to set a different URL for digg you can use the parameter diggBookmarkURL.

widgetTitle A string with maximum 50 chars, which will be used as the default title suggested to the user on social bookmarking sites that support such title.

Note - this is the same parameter which is used to specify the shareable application title when posting to social networks.

Send to Email parameters

Parameter Name	Type	Description
emailSubject	string	The subject line of the e-mail message to be sent.
emailBody	string	The body of the e-mail message to be sent in HTML format.
showEmailAfterPost	Boolean	Determines if the Share menu offers the user the opportunity to send an e-mail message to his friends after successful posting.
dontSendEmail	Boolean	Assigning "true" to this parameter, causes the Share menu not to actually send the e-mail message, so you can send it from your own servers.

emailSubject The subject line of the e-mail message to be sent. If the subject includes the string $sender$ it will be replaced with the name given by the sender.

emailBody The body of the e-mail message to be sent in HTML format. The following place holders may be used in the e-mail body and will be replaced when the e-mail message is sent: $sender$ will be replaced with the name given by the sender.URL will be replaced with a Share-menu-generated URL, where the recipient can go to view the shareable application.

showEmailAfterPost A Boolean parameter, which determines if the Share menu offers the user the option to send an e-mail message to his friends after successful posting. The default value is "true".

dontSendEmail A Boolean parameter, which allows you to tell the Share menu not to actually send the e-mail message. You can use this parameter along with the onEmail event to send the actual e-mail message from your own servers.

Statistics on the e-mail messages will still be accumulated as if the e-mail message was sent by the Share menu. The default value is "false" (the Share menu DOES send e-mail).

Send to Desktop parameters

Parameter Name	Type	Description
widgetTitle		This string is used as the title of the shareable application.
widgetDescription		This string is used as the description of the shareable application.
widgetCopyright		This string is used as the copyright notice of the shareable application.
iconURL		The URL from which to download an icon.
defaultPreviewURL		URL of an image used as the preview of your shareable application.

widgetTitle This string will be used as the title of the shareable application where appropriate. The default value is "widget".

widgetDescription This string will be used as the description of the shareable application where appropriate. The default value is "widget".

widgetCopyright This string will be used as the copyright notice of the shareable application where appropriate. The default value is "".

iconURL This string will be used as the URL from which to download an icon. This icon will be placed inside the shareable application's zip package.

defaultPreviewURL Using this parameter you can supply a URL of an image that will be used as the preview of your shareable application for all the shareable application containers which support such a preview.

Reporting parameters

Parameter Name	Type	Description
CID	string	A string of max 100 chars length, that will be associated with this transaction and will later appear on reports.
advancedTracking	Boolean	Assigning "true" to this parameter will enable viral distribution tracking feature.

CID A string of max 100 chars length. This string will be associated with this transaction and will later appear on reports generated by the Distribution Manager. This data will allow you to associate the report information with your own internal data. For example, to identify a specific shareable application or page on your website.

Usage Examples

ActionScript:

```
cfg['CID']='shareableApp A';
```

JavaScript:

```
var pconf = {
    CID: 'shareableApp A',
    ...
}
```

advancedTracking A Boolean parameter. Assigning "true" to this parameter enables the tracking of viral distribution of your shareable application, and collect information that will be available in the Distribution Service reporting service. The posted content (FlashVars) will be changed for this purpose.

Share API Methods

The following methods are used to create the Share menu.

Wildfire.initPost() method

Initializes the Share menu's post module for the first time and draws its SWF file in the specified container on the page.

```
Wildfire.initPost(partnerID, containerID, width, height, pconf);
```

Parameters:

`partnerID` – The partner ID as assigned to your account in the Distribution Manager

`containerID` – A string with the ID of the container element where the Share menu will be drawn

`Width` – The width, in pixels, of the Share menu

`Height` – The height, in pixels, of the Share menu

`pconf` – A configuration object to be used for initializing the shareable application

Return value:

None

Wildfire.renderPostButton() method

Creates a Share button. When clicked, the Share post module is displayed in a generated floating div near the button.

```
Wildfire.renderPostButton (partnerID, width, height, pconf);
```

Parameters:

`partnerID` – The partner ID as assigned by Gigya

`width` – The width, in pixels, of the Share menu

`height` – The height, in pixels, of the Share menu

`pconf` – A configuration object to be used for initializing the shareable application

Return Value:

None

Wildfire.<containerID>.applyConfig() method

This method is intended for use when the content that you want your user to post is either not available at the time when initPost was executed or has changed after the initialization. The <containerID> should be replaced with the containerID that was used in the initPost() call for this specific instance of the Share menu. For example, if the <DIV> that contains the Share menu has the ID "divWildfirePost", the method should be called as follows:

```
Wildfire.divWildfirePost.applyConfig(pconf);
```

Parameters:

`pconf` – A configuration object to be used for re-initializing the shareable application.

Return value:

None

Share Menu

The Share menu generates many different events for various situations which are driven by user interactions. A program can listen for events which are relevant to it. A program interested in certain events can register event handlers for those events and execute code when those events are received.

Defining an event handler

An event handler is a function that receives a single "event object" parameter, having the following signature:

ActionScript2

```
function eventHandlerName(eventObj:Object): Void
```

ActionScript3

```
function eventHandlerName(eventObj:Object): void
```

The "eventObj" object includes members, which correspond to the specific event that was fired.

For example: for the Email Event, the eventObj will include the following members:

Field	Type	Description
type	string	The name of the event: “email”.
ModuleID	string	The ID of the module that triggered the event.
recipients	array	An array of objects that each has a name property and an email property corresponding to the recipients of the e-mail message that was sent.
senderName	string	The sender’s name, as entered in the Share menu’s user interface.

The following members of the event object are available for all events:

- type property - the name of the event fired.
- ModuleID property - with the Share menu instance ModuleID and additional properties depending on the specific event.

Event handler example

ActionScript 2

```
function displayEventMessage(eventObj:Object): Void {
    trace(eventObj.type + ' event fired! module ID=' + eventObj.ModuleID);
}
```

ActionScript 3

```
function displayEventMessage(eventObj:Object): void {
    trace(eventObj.type + ' event fired! module ID=' + eventObj.ModuleID);
}
```

Registering a handler for an event

Registering a handler for an event is performed as part of the Share menu configuration. The configuration object includes several parameters for registering event handlers:

- onLoad
- onPostProfile
- onClose
- onEmail
- onRenderDone
- onNetworkButtonClicked
- onCopy

Each of the above parameters may be assigned a value, which is the Event Handler function name, that will be called whenever the corresponding event occurs.

ShareMenuEvent.LOAD Event

Fired: When the Share menu is loaded

Event Handler: onLoad

Field	Type	Description
type	string	The name of the event: “load”.
ModuleID	string	The ID of the module that triggered the event.

ShareMenuEvent.POST_PROFILE Event

Fired: After a user posts content to a profile, after a user sends an e-mail message, and after a user posts a comment or bookmark.

Event Handler: onPostProfile

Field	Type	Description
type	string	The name of the event: “postProfile”.
ModuleID	string	The ID of the module that triggered the event.
network	string	The social network to which the user posted the comment.
partnerData	string	Deprecated. Use CID instead.
CID	string	The data set by the partner when the Share menu was initialized. For more information, see the entry for CID in “Reporting parameters” on page 77.
username	string	The name the user entered for the social network.
content	string	The content that was posted.

ShareMenuEvent.CLOSE Event

Fired: When the user clicks the Close button on the Share menu. Note that the Close button is only available if you explicitly requested that it be displayed by setting showCloseButton=true in XMLConfig.

Event Handler: onClose

Field	Type	Description
type	string	The name of the event:"close".
ModuleID	string	The ID of the module that triggered the event.

ShareMenuEvent.EMAIL Event

Fired: After an e-mail message has been sent by the user.

Event Handler: onClose

Field	Type	Description
type	string	The name of the event: "email".
ModuleID	string	The ID of the module that triggered the event.
recipients	array	An array of objects that each has a name property and an email property corresponding to the recipients of the email that was sent.
senderName	string	The sender's name as entered in the Share menu user interface.

ShareMenuEvent.RENDER_DONE Event

Fired: Whenever the Share menu finishes rendering one of the screens.

Event Handler: onRenderDone

Field	Type	Description
type	string	The name of the event: "renderDone".
ModuleID	string	The ID of the module that triggered the event.
page	string	The name of the screen that has been rendered. Possible values for this field are: NetworkSelection, AfterOpenURL, SelectEmailContacts, PostToSocialNetwork, ImportEmailContacts, and AskAboutRemember.

ShareMenuEvent.NETWORK_BUTTON_CLICK Event

Fired: Whenever a social network button (icon) has been clicked.

Event Handler: onNetworkButtonClicked

Field	Type	Description
type	string	the name of the event: "networkButtonClicked".
ModuleID	string	The ID of the module that triggered the event.
network	string	The name of the social network whose associated button (icon) has been clicked.

ShareMenuEvent.COPY Event

Fired: Whenever the user clicks the copy button or clicks within the codebox.

Event Handler: onCopy

Event Object Fields:

Field	Type	Description
type	string	The name of the event: "copy".
ModuleID	string	The ID of the module that triggered the event.
content	string	The content that was copied.
CID	string	The data set by the partner when Share menu was initialized. For more information, see the entry for the CID parameter in the "Reporting parameters" on page 77 table.
network	string	The ID of the module that triggered the event.

Share menu properties

In The Distribution Service's shareable application implementation, the Share Menu component manages the loading of the shareable application as well as the population of the configuration XML (of which UIConfig is a single parameter). When you make changes to a Share component instance using the options on the Share panels, component parameters on the component instance are populated by the panels. You can also manipulate the Share component's parameters using the methods and properties of the UIComponent class, described below.

share property

A reference to the loaded SWF content.

percentLoaded property

A normalized value between 0 and 100 representing the load progress of the SWF file.

Share menu methods

You can manipulate the Share component's parameters using the methods of the UIComponent class, described below.

addParam() method

```
addParam(paramName:String, paramValue:Object):void
```

Add a parameter to the Share menu config object at runtime.

removeParam() method

```
removeParam(paramName:String):void
```

Remove a parameter from the Share menu config object. Only parameters added using the addParam method can be removed this way.

show() method

```
show():void
```

Set the visibility of the component to true.

hide() method

```
hide():void
```

Set the visibility of the component to false.

refresh() method

refresh():void

Reload the Share SWC.

Insert shareable applications and share menus into web pages

You can insert the Share menu into web pages in two ways: using Adobe's Project Radiate extensions, or programmatically.

Insert the Share menu into web pages with the Dreamweaver extension

For information about inserting the Share menu into web pages using the Dreamweaver extension, see "Make applications shareable" on page 38.

Insert the Share menu into web pages without the Dreamweaver extension

You can insert the Share menu into web pages without using the Dreamweaver extension. The following example code shows how to do this.

Example code

```
<html>
  <head>
    <!-- Step 1 - Including the Share menu API: -->
    <script src="http://cdn.gigya.com/wildfire/js/wfapiv2.js"></script>
  </head>
  <body>
    <br />Share Menu Post example page<br /><br />

    <!-- Step 2 - Setting the content of the post (place your content HTML to post here): -->
    <textarea rows="1" cols="1" id="TEXTAREA_ID" style="display: none">
        Hello World! (Post content)
    </textarea>

    <!-- Step 3 - Positioning Share menu Post on the screen: -->
    <div id="divWildfirePost"></div>

    <script>
        var pconf = {
            defaultContent: 'TEXTAREA_ID'
        };

        Wildfire.initPost('PARTNER-ID', 'divWildfirePost', 400, 300, pconf);
    </script>

  </body>
</html>
```