macromedia®

Flash Player Detection Kit

by Michael Williams

August 2005

Macromedia, Inc.
601 Townsend Street
San Francisco, CA 94103
415–832–2000

Contents

Introduction

Web technologies evolve rapidly. Content developers need to know that end users have the latest tools they need to interact with the developer's creations. In the case of Macromedia Flash, developers who use the latest Flash features must ensure that their audience always has the right version of Flash Player installed.

This document provides the following sections to help developers detect the installed version of Flash Player and initiate upgrades:

- **Detecting Flash Player**—Describes what you need in order to detect the user's Flash Player version and to determine if that version can display your Flash content.
- Overview of Detection and Installation Techniques—Outlines the different Flash Player detection and installation methods, discusses under what circumstances you would want to each method, and describes the contents of the Detection Kit ZIP file.
- **Using Client-Side Scripting to Detect the Flash Player Version**—Discusses the use of JavaScript and VBScript to detect the Flash Player, and provides scripts to insert in web pages.
- Using ActionScript to Detect the Flash Player Version—Discusses the use of ActionScript in a SWF file to detect the Flash Player version, and provides details on the ActionScript included with the Detection Kit.
- Using Server-Side Code to Detect the Player Version—Discusses sever-side Flash Player detection and provides scripts for PHP and ColdFusion.
- Installing or Updating Flash Player—If Flash Player is not installed on the end user's computer, or if the installed version won't allow proper viewing of your Flash content, you can initiate the installation of the latest version of Flash Player.

Detecting Flash Player

Flash content – whether interactive games, advertising, flexible messaging, video programming, or commerce-based applications – runs on Flash Player. The version of Flash Player required to view your content needs to be installed on the end user's computer. Even with the wide variety of computing platforms and browsers your audience uses to access the web and your Flash content, you can detect the presence of Flash Player and its version in most instances and initiate an installation or upgrade of Flash Player on the end user's computer.

It would be quite a task, however, to tests all possible operating system and browser combinations when deploying new Flash content. Macromedia has put together the following best practices and templates to help you target a wide variety of deployment environments.

Most expectations you may have for your detection process can be met if the process you use fulfills the following requirements:

- The detection process should run on all operating systems and browser combinations supported by Flash Player. For a list of supported operating systems and browsers, see the Flash System Requirements at **http://www.macromedia.com/software/flash/productinfo/systemreqs/**.
- The detection process should detect both major and minor versions of Flash Player. For example, Flash Player 6 was revised several times during its lifecycle. To target those specific versions, the detection method must be able to determine the exact player version installed on the user's computer, and the exact version needed to correctly display your Flash content.
- The detection process should work with all future versions of Flash Player, without requiring any changes to the detection mechanism you've chosen to implement.
- The detection process should direct users to install the latest version of Flash Player in as unobtrusive a way as possible.

Many detection methods fail to meet one or more of these requirements, and often confuse visitors to the web site who cannot access content, or figure out which version of Flash Player is required. The Flash Player Detection Kit has been designed to help you identify the version of Flash Player installed on end users' computers, and upgrade or install the latest version, with minimum distraction for the end-user.

In addition to the Flash Player detection techniques described in this document, or as an alternative, you can always provide an introduction page that displays version information and contains a link to the Flash Player download page:

This site requires Flash Player 7 or later.

Get macromedia
FLASH
PLAYER

The Flash Player download page is located here:

http://www.macromedia.com/go/getflashplayer/

Overview of Detection and Installation Techniques

Flash content is incorporated into websites in many different ways. The various architectures and configurations of websites necessitate different approaches to detecting the Player version an end-user is using to view your Flash content. One detection method won't work for all websites, but most websites can implement at least one of the methods described in this document. Once a player and its version have been detected, you can present the user with an experience that best fits your site design, whether that is to prompt the user to proceed with a player product installation or displaying alternate content. This section points to the resources the Detection Kit provides to assist you.

Selecting a Flash Player detection method

To detect the Flash Player version, you select and customize a detection method. Make sure to review the discussion of each method, below, to see the advantages and disadvantages of each method:

- Client-side scripting detection–This method uses both JavaScript and VBScript to check for the existence of a Netscape plug-in or an ActiveX control, and provides the logic for either displaying your Flash content or initiating an installation of the correct player version.

  Use script-based detection when:

  - Script-based detection can be used when deploying Flash content to modern browsers and operating systems. Based on a census of computer users, Macromedia has determined that 98% of Internet-enabled computers are equipped with modern browsers and operating systems. Furthermore, 98% of Internet-enabled computers have Flash Player 5 or later installed.
  - If you want to display alternate content based on the configuration of end users' computers and browsers, script-based detection can determine version and configuration information that is not detectable by other methods.

  Avoid script-based detection when:

  - While generally reliable, script-based detection will fail if the browser loading the content has had scripting disabled. For the rare occurrences where scripting is disabled, Macromedia recommends using the `<noscript>` HTML tag to display alternate content.
  - While increasingly uncommon, some older browsers do not support scripting (for example, the Macintosh versions of Internet Explorer 4.0 and earlier). If older browsers are important to distributing your content, you will need to detect the browser version and display alternate content suitable for the browser's display capabilities.

- ActionScript-based detection–This method uses a Flash SWF file to detect which version of the player is installed and redirects the end-user to the Macromedia installation if their player needs to be updated.

  Use ActionScript detection when:

  - You only need to target end users who have Flash Player 4.0r11 (revision 11) or greater installed.
  - Browser redirects are not a problem on your web site.

  Avoid ActionScript detection when:

  - Flash Player is not installed on the end users' computers.
  - If browser redirects are a problem for your web site (for example, to redirect the user to the Flash Player installation page).
  - When you would prefer to display alternate content inline rather than directing end users to a different page to install Flash Player.

- Server-side detection–This method creates a server-side application to determine which version of Flash Player is installed.

  Use server-side code to detect Flash Player when:

  - You expect users to have Flash Player 6.0r65 or later.
  - You want a server-side implementation as part of a dynamic web site.

  Avoid server-side code to detect Flash Player when:

  - You do not have access to server-side scripting resources.
  - You want to target users with versions of Flash Player older than Flash Player 6.

- A combination of the above techniques–You can develop your own custom solution by combining two or more of the above techniques to give your users the experience that's most appropriate for your site and content.

Remember, the success of the detection method you use depends upon the configuration of your website, your target audience, and the end-user environment. Review the recommendations discussed in each section before implementing a detection and installation method.

Initiating an installation or upgrade of Flash Player

Once you have determined whether an end-user has a Flash Player installed and what version it is, you have the choice of displaying alternative content, directing the user to the Macromedia Flash Player download page, or initiating an installation or update of Flash Player through a player product installation method, as described in the Installing or Updating Flash Player section, below.

To initiate a player product installation on the end-user's computer, you can use Macromedia's Flash Player Express Install. This method, and how to customize it, is described in the Customizing Express Install section. The location of these files is described in the Express Install files section.

Contents of the Detection Kit

After reviewing the different detection methods discussed in this document, familiarize yourself with the files contained in the Detection Kit ZIP file. When you unzip the Detection Kit into a directory of your choosing, you can view the following sets of files:

- *A copy of this document* – Found in the root directory, which describes the requirements, methods, and procedures for detecting and updating the end-user's Flash Player version. Read this document to understand, choose, and customize a detection method.
- *ActionScript-based detection method sample files* – Found in the ActionScript Based Detection directory. Use these files – along with the instructions in the Using ActionScript to Detect the Flash Player Version section, below – to detect the end-user's Flash Player version, and to display Flash content if the correct version was detected, or to prompt the user to install the correct version of Flash Player.
- *Client-Side detection method sample files* – Found in the Client-Side Detection directory. Use these files – along with the instructions in the Using Client-Side Scripting to Detect Flash Player Version section, below – to detect the end-user's Flash Player as both ActiveX control and a plug-in, and to display Flash content if the correct version was detected, or to prompt the user to install the correct version of Flash Player.
- *Express Install sample files* – found in the root directory. Use these files – along with the instructions in the Installing or Updating Flash Player section, below – to enable the detection of the end-user's Flash Player version, initiate an end-user installation, and after successful installation, return the end-user to your site to view your content with the correct version of the player.

ActionScript-based detection method sample files

Under the root directory, the ActionScript Detection directory contains the following files:

Directory/File Name	**Description**
actionscript_example.html	Sample HTML page; containing the SWF for completing the ActionScript-based detection method
flash_AS_detection.as	Sample ActionScript file; contains the logic for completing the ActionScript-based detection method
flash_AS_detection.fla	Sample source file; calls on the ActionScript files of the same name to complete the ActionScript-based detection method
flash_AS_detection.swf	Sample SWF file; contains the ActionScript logic used to initiate the ActionScript-based detection method
flash_content/ directory	Sample Flash content; contains the sample content you want to provide the user if they have the correct version of Flash Player installed
upgrade_flash/ directory	Sample upgrade files; contains the sample files used to prompt the end-user to upgrade to the latest version of Flash Player

Client-side detection method sample files

Under the root directory, the Client-Side Detection directory contains the following files:

Directory/File Name	Description
example.swf	File used to test your implementation of the Client-side detection
ClientSideDetection.html	Sample HTML page; contains the JavaScript and VBScript logic for client-side Flash Player detection

Flash Player Express Install sample files

Under the root directory, the Express Installation directory contains the following files:

File Name	Description
example.swf	File used to test your implementation of the Express Install process.
playerProductInstall.as	Sample ActionScript file; contains the logic for implementing the Express Install process.
playerProductInstall.fla	Sample source file; calls on the ActionScript files of the same name to complete the Express Install process.
playerProductInstall.html	Sample HTML page; contains the JavaScript logic for completing the Express Install process.
playerProductInstall.swf	Sample SWF file; contains the ActionScript logic used to initiate the Express Install process.
playerProductInstallCallback.as	Sample ActionScript file; contains the callback function dictating what happens when end user does not install the new Flash Player through the Express Installprocess.

Using Client-Side Scripting to Detect the Flash Player Version

The most common way to detect Flash Player is by using JavaScript and VBScript to check for the existence of either a Netscape plug-in or an ActiveX control. The type of script you use depends on the browser your end-user uses to view web content. Because Flash Player exists as both a browser plug-in and an ActiveX control, a combination of both JavaScript and VBScript is needed to properly check for the player. VBScript is used to detect the ActiveX installation in Internet Explorer, while JavaScript is used to detect browser plug-ins for Netscape, Firefox, Opera and other browsers that support plug-ins.

For details on when to use this detection method (and on when to avoid it), see "Selecting a Flash Player detection method" on page 3.

The client-side scripting method of Flash Player detection is illustrated in the sample file ClientSideDetection.html.

In this method, you first create functions for script-based detection of the ActiveX control and the plug-in. These two functions will be called by another function that returns the version number of the user's installed Flash Player. That function calls the ActiveX detection function if the user is running Internet Explorer under Windows. Otherwise it calls the JavaScript detection function.

Depending on the version number retrieved, the client-side script proceeds in one of the following ways:

- If the retrieved version number indicates the right Flash Player is installed, the `<object>` and `<embed>` tags for your content will be written to the page and your content will be shown to the user.
- If the user has an older version of Flash Player installed or if no Flash Player is detected at all the user will be shown the alternate content, including a link to the Get Flash web page on the Macromedia web site.

The next sections discuss portions of these client-side scripts in detail.

Using VBScript to detect the ActiveX version of Flash Player

The following example uses VBScript to detect the ActiveX control version of Flash Player in Internet Explorer for Windows. As you review the script, note that it does not search for a specific version of Flash Player, but instead returns the major version number of the installed Player. This lets the script function properly as new versions (and thus numerically larger version numbers) of Flash Player are released.

```
<SCRIPT language=VBScript>
<!-- // Detect Flash Player ActiveX control version information
Function VBGetSwfVer(i)
  on error resume next
  Dim swControl, swVersion
  swVersion = 0

  set swControl = CreateObject("ShockwaveFlash.ShockwaveFlash." + CStr(i))
  if (IsObject(swControl)) then
    swVersion = swControl.GetVariable("$version")
  end if
  VBGetSwfVer = swVersion
End Function
// -->
</SCRIPT>
```

Using the `VBGetSwfVer(i)` function, the script searches for the “ShockwaveFlash.ShockwaveFlash.x” ActiveX object in the Windows registry. If the script determines that Flash Player is installed using `IsObject()`, the Player itself is queried for its version number using the `GetVariable("$version")` function.

Flash Player versions 4.0r11 and later report a full version and platform string when `$version` is requested. This lets you query for the full version, which contains the operating system, version number, and the minor revision information. For example, you might retrieve a string similar to the following: WIN 6,0,65,0. In this instance, the user’s computer has Flash Player 6, revision 65 running on the Windows operating system.

Using JavaScript to detect the Netscape Plug-in version of Flash Player

Detecting the existence of Flash Player and its version number requires some additional scripting work for browsers that use the Netscape plug-in API. For these browsers you not only have to write a script to find out if the Player exists and what version number it is, but also what browser is installed on the user’s computer.

The following script determines the existence and version information of the Flash Player plug-in. This applies to Netscape, Firefox, Mozilla, Safari, Opera, and all other browsers that support the Netscape Plug-in API and a way to interrogate it using browser scripting.

Note: In creating this script, we identify Netscape plug-in based browsers as any web browser other then Internet Explorer. A special case was made for the Opera browser, which, while it identifies itself as Internet Explorer, did not provide support for ActiveX controls until version 8.01 (the latest release at the time this document was written).

```
<SCRIPT language=JavaScript1.1 type=text/javascript>
<!-- // Detect Client Browser type
var isIE  = (navigator.appVersion.indexOf("MSIE") != -1) ? true : false;
var isWin = (navigator.appVersion.toLowerCase().indexOf("win") != -1) ? true : false;
var isOpera = (navigator.userAgent.indexOf("Opera") != -1) ? true : false;

jsVersion = 1.1;
// JavaScript helper required to detect Flash Player PlugIn version information
function JSGetSwfVer(i){
    // NS/Opera version >= 3 check for Flash plugin in plugin array
    if (navigator.plugins != null && navigator.plugins.length > 0) {
                if (navigator.plugins["Shockwave Flash 2.0"] || navigator.plugins["Shockwave
    Flash"]) {
                     var swVer2 = navigator.plugins["Shockwave Flash 2.0"] ? " 2.0" : "";
                     var flashDescription = navigator.plugins["Shockwave Flash" +
    swVer2].description;
                     descArray = flashDescription.split(" ");
                tempArrayMajor = descArray[2].split(".");
                versionMajor = tempArrayMajor[0];
                versionMinor = tempArrayMajor[1];
                     if ( descArray[3] != "" ) {
                          tempArrayMinor = descArray[3].split("r");
                     } else {
                          tempArrayMinor = descArray[4].split("r");
                     }
                     versionRevision = tempArrayMinor[1] > 0 ? tempArrayMinor[1] : 0;
                     flashVer = versionMajor + "." + versionMinor + "." + versionRevision;
                } else {
                     flashVer = -1;
                }
    }
    // MSN/WebTV 2.6 supports Flash 4
    else if (navigator.userAgent.toLowerCase().indexOf("webtv/2.6") != -1) flashVer = 4;
    // WebTV 2.5 supports Flash 3
    else if (navigator.userAgent.toLowerCase().indexOf("webtv/2.5") != -1) flashVer = 3;
    // older WebTV supports Flash 2
    else if (navigator.userAgent.toLowerCase().indexOf("webtv") != -1) flashVer = 2;
    // Can't detect in all other cases
    else {
        flashVer = -1;
    }
    return flashVer;
}
```

Like the `VBGetSWFVer()` function used in the VBScript detection script, the JavaScript `JSGetSwfVer()` function searches for the Shockwave Flash plug-in by parsing the `navigator.plugins` array, which is used by all browsers that support the Netscape plug-in API. If the Flash Player plug-in is located in the array, its version number is stored in the `flashVer` variable, and returned to any script that calls it (see the Putting It Together section, below). Browsers that do not support the `navigator.plugins` array, or that do not have Flash Player installed, return a value -1, indicating that you must install the latest version of Flash Player.

Note: Internet Explorer 4 and earlier running under Mac OS classic does not support the `navigator.plugins` array.

Putting It Together

Now that you have methods to detect Flash Player and return the version for both the ActiveX control and plug-in across platforms, you need to combine the information in a function that can be used to present the desired experience to your end users. The following script retrieves the version number of the installed Flash Player, and compares the installed version to the minimum version required to view your content.

```
// When called with reqMajorVer, reqMinorVer, reqRevisionreturns true
// if that version or greater is available
function DetectFlashVer(reqMajorVer, reqMinorVer, reqRevision)
{
    reqVer = parseFloat(reqMajorVer + "." + reqRevision);
    // loop backwards through the versions until we find the newest version
    for (i=25;i>0;i--) {
         if (isIE && isWin && !isOpera) {
               versionStr = VBGetSwfVer(i);
         } else {
               versionStr = JSGetSwfVer(i);
         }
         if (versionStr == -1 ) {
               return false;
         } else if (versionStr != 0) {
             if(isIE && isWin && !isOpera) {
                    tempArray         = versionStr.split(" ");
                    tempString        = tempArray[1];
                    versionArray      = tempString .split(",");
             } else {
                    versionArray      = versionStr.split(".");
             }
             versionMajor      = versionArray[0];
             versionMinor      = versionArray[1];
             versionRevision   = versionArray[2];

             // 7.0r24 == 7.24
             versionString     = versionMajor + "." + versionRevision;
             versionNum        = parseFloat(versionString);

             // is the major.revision >= requested major.revision AND the
             // minor version >= requested minor
             if ( (versionMajor > reqMajorVer) && (versionNum >= reqVer) ) {
                  return true;
             } else {
                  return ((versionNum >= reqVer && versionMinor >= reqMinorVer) ? true : false );
             }
        }
    }
}
```

The `DetectFlashVer()` function accepts the following three parameters: `reqMajorVer, reqMinorVer,` and `reqRevision`.

These values represent the minimum version of Flash Player your content requires for playback. For example, to check for Flash Player 6.0r65 you would use the following call:

```
var hasReqestedVersion= DetectFlashVer(6, 0, 65);
```

The `DetectFlashVer()` function calls `VBGetSwfVer()` for Windows Internet Explorer, and `JSGetSwfVer()` for all other browsers and operating systems. The called function loops backward through the version numbers, searching from highest to lowest version number, until the latest installed version is located. The returned Flash Player version is compared against the values that were passed to `DetectFlashVer()`, which checks to see whether the required version of Flash Player is installed, and returns a true or false value.

If the detected version of the Player is equal to or greater than the version your content requires, the `DetectFlashVer()` function returns `true`, and the SWF file is embedded in the page and plays using Flash Player. If the function returns `false`, the minimum version of Flash Player needed to view the content is not installed, and you should either display alternate content or direct the user to install the latest version of Flash Player.

The script below shows how to structure an `if` statement to choose between embedding the SWF file if the correct version of Flash Player is installed, or directing the user to try a different method of installing Flash Player:

```
if (hasReqestedVersion) {
    // EMBED the SWF
} else {
    // Instruct the user to install the player, start Player Product Install,
    // or show alternate content
}
```

The file ClientSideDetection.html uses JavaScript within the body of the page to write the `<object>` and `<embed>` tags for the Flash content directly into the page at that spot. If you prefer to redirect the user to a different page containing the Flash content, installation content, or alternate content you can customize that file accordingly.

By default the ClientSideDetection.html example requires that you enter a few global values for the detection to work properly. These values are:

1. Required Major, Minor, and Revision versions of the player necessary to view your content. This is specified at the beginning the script. The following script requires Flash Player 7.0r19.

```
// -----------------------------------------------------------------------------
// Globals
// Major version of Flash required
var requiredMajorVersion = 7;
// Minor version of Flash required
var requiredMinorVersion = 0;
// Minor version of Flash required
var requiredRevision = 19;
```

2. Alternate content that will be displayed in the event that the player is not installed, the player is installed but not the version you require, or browser scripting is disabled. This is defined in the `alternateContent` variable inside the script block contained in the example `<body>` tag.

```
var alternateContent = 'Alternate HTML content should be placed here.'
+ 'This content requires the Macromedia Flash Player.'
+ '<a href=http://www.macromedia.com/go/getflash/>Get Flash</a>';
document.write(alternateContent);  // insert non-flash content
```

Using ActionScript to Detect the Flash Player Version

If your targeted user audience has Flash Player 4.0r11 (version 4, revision 11) or later installed on their computers, you can use a Flash SWF file to detect which version of Flash Player is installed. The Detection Kit includes a Flash FLA file and the ActionScript code needed to create a SWF file that determines which version of Flash Player is installed on an end user's computer.

For details on when to use this detection method (and on when to avoid it), see "Selecting a Flash Player detection method" on page 3.

The Flash files needed to detect the Flash Player version are:

- flash_AS_detection.fla – The FLA file from which you generate the Flash Player Detection SWF file.
- flash_AS_detection.as – The ActionScript include file whose parameters you must modify to detect the version of Flash Player required to view your content.

To detect Flash Player using ActionScript:

1. Edit the flash_AS_detection.as file to detect the version of Flash Player needed to view your content. For a description of the variables you can modify see, "Flash Player detection ActionScript variables" later in this section.
2. Publish the flash_AS_detection.fla file as a SWF to embed on your web site.
3. Upload the Flash ActionScript detection files to your web server. For information on the files, see "Uploading the Flash Player Detection Kit files to your web server."
4. Test the Flash Player Detection SWF file by viewing it in a browser that has the Flash Player version 4.0r11 or higher installed.

Flash Player detection ActionScript variables

You can change the values of the following variables, which are declared in the flash_AS_detection.as file, to customize the behavior of the Flash Player Detection SWF file:

`altContentURL` The URL that the visitor should be redirected to if they do not have the required version of Flash installed. The URL can be alternate content for use without Flash, or an upgrade page as in this example. The following example uses a relative path to redirect the user's browser to a web page from which they can upgrade to the latest version of Flash:

```
altContentURL = "upgrade_flash/upgrade_flash.html";
```

`flashContentURL` The URL that the visitor should be sent to if they have the required version of Flash. The following example uses a relative path to redirect the user's browser to a web page containing your Flash content:

```
flashContentURL = "flash_content/flash_content.html";
```

`contentVersion` The minimum required Player version needed to view your Flash content. The following example specifies that Flash Player 6.0r65 be installed to view the content:

```
contentVersion = 6;
```

`contentMajorRevision` The major revision number (or dot release) of the Player needed to view your Flash content. The following example specifies 0 (zero), which, in conjunction with the above example, specifies that Flash Player 8.0 is the required version:

```
contentMajorRevision = 0;
```

`contentMinorRevision` The revision number of the player necessary to view the content. The following example specifies 3 (three), which, in conjunction with the above example, specifies that Flash Player 8.0.3 is the required version:

```
contentMinorRevision = 65;
```

`requireLatestVersion` Specifies that the latest available version of Flash Player must be installed in order to view your content. The following example is set to `false`, which checks for the minimum required version of the Player.

```
requireLatestVersion = false;
```

Note: If this setting is set to `true`, it is recommended that you check for the latest version of the Player you require, and ensure that settings inside the Flash movie reflect the version as well. These settings begin at line 35 of the flash_AS_detection.as include file. For example:

```
MACLatestVersion = 7;
MACLatestMajorRevision = 0;
MACLatestMinorRevision = 19;
```

Additionally, there is a variable setting in the `<object>` tag of the HTML page:

`allowFlashAutoInstall` Specifies that, when running under Internet Explorer on Windows, the latest version of Flash Player is to be automatically installed. Specifying `true` causes the Flash Player Detection SWF file to skip the detection test, and lets Internet Explorer automatically upgrade Flash Player to the latest available version.

Note: Macromedia recommends that you specify this in the HTML `<object>` tag, and *not* in the `<embed>` tag.

The following example specifies that the Flash Player Detection SWF not automatically update the Player.

```
allowFlashAutoInstall = false;
```

Upgrading Flash Player 3 and earlier versions

At the bottom of the flash_AS_detection.as include file is a `getURL()` command that contains a hard coded address. This address is used when the Flash Player Detection SWF file detects Flash Player 3 or earlier versions. Prior to Flash Player 4, the Player did not support the use of variable. For this reason, we use the `getURL()` command to redirect the browser to a page from which the user can upgrade to the latest version of Flash Player.

Macromedia recommends that you change the Meta Tag setting in the actionscript_example.html file. The Meta Tag redirects the browser when the user either does not have Flash Player installed, or has a faulty installation. The URL setting in this tag is usually set to the same HTML file as the `altContentURL` variable, as in the following example:

```
getURL("upgrade_flash/upgrade_flash.html", "_self");
```

Uploading the Flash Player Detection Kit files to your web server

Once you've modified the flash_AS_detection.swf and flash_AS_detection.as files using the ActionScript variables described earlier in this section, you must upload these files and the HTML files specified by the altContentURL and flashContentURL variables to your web server.

The files you need to upload to the web server are:

- flash_AS_detection.swf – Do not change the name of this file.
- actionscript_example.html – You can rename this file, as needed.
- altContentURL HTML file – The HTML file specified by the `altContentURL` variable.
- flashContentURL HTML file – The HTML file specified by the `flashContentURL` variable, which contains your Flash content.

Using Server-Side Code to Detect the Flash Player Version

If you are familiar with server-side development languages such as ColdFusion, PHP, or others, you can create a server side application to determine which version of Flash Player is installed using the MIME type information included in the HTTP_ACCEPT header information. To do this, users must have Flash Player 6.0r65 (revision 65) or later installed.

For details on when to use this detection method (and on when to avoid it), see "Selecting a Flash Player detection method" on page 3.

With the release of Flash Player 6, the Flash MIME type "application/x-shockwave-flash" was added to the client HTTP_ACCEPT header. The HTTP_ACCEPT header is sent to the server each time a web page is requested, and includes information on all MIME types the browser can accept. If the user's computer has Flash Player 6.0r65 (revision 65) or later installed, the "application/x-shockwave-flash" entry is added to the request header.

Important Notes about server side detection

The following issues with server side detection should be closely noted. Internet Explorer on Mac OS is hard coded to accept `'*/*'` and cannot be modified dynamically. Secondly, users can manually uninstall Flash Player by deleting the executable itself. This orphans the custom accept header such that server detection logic thinks Flash has been installed, when in fact it is no longer available. It is also possible to disable ActiveX controls through the Windows XP Service Pack 2 security settings, which will also orphan the accept header.

Using ColdFusion to detect Flash Player

The following ColdFusion code uses the `cfif` tag to parse the return value of the HTTP Accept header, and checks if the application/x-shockwave-flash MIME type is returned:

```
<cfif #cgi.http_accept# contains "application/x-shockwave-flash">
   // Flash 6r65 or above is installed
<cfelse>
   // Flash was not detected
</cfif>
```

For more information on ColdFusion and the `cfif` tag, see the following article on the Macromedia Developer Center: **http://www.macromedia.com/devnet/mx/flash/articles/browser_hawk.html**.

Using PHP to detect Flash Player

The following PHP code uses the `$hasFlash` variable to parse the return value of the HTTP Accept header, and checks if the application/x-shockwave-flash MIME type is returned:

```
<?php
// Search through the HTTP_ACCEPT header for the Flash Player MIME type.
if (strstr($_SERVER['HTTP_ACCEPT'], 'application/x-shockwave-flash'))
{
   $hasFlash=true;
}
?>
```

Installing or Updating Flash Player

Macromedia provides a number of installation options for end users and developers. Standalone installers are provided for download and installation, as are CAB and XPI files that handle integrated installation within Internet Explorer and Netscape-based browsers.

In cases where the detection kit determines that the end user does not have the required Flash Player version, there are a number of options, for example:

- **Display alternate, non-Flash content** – In the case where Flash is not installed and the correct version is not available, or browser scripting is disabled, you can offer alternative, non-Flash content.
- **Direct Users to the Macromedia Flash Download Center** – You can direct users to the Macromedia Flash Download Center at **http://www.macromedia.com/go/getflashplayer**. The Flash Download Center automatically determines the browser and platform the user is using, and provides the appropriate Flash Player installer.
- **Invoke the Flash Player Express Install process** – Provide a streamlined, in-context experience that installs the latest version of Flash Player, and then sends the user back to the site and content that initiated the installation process.
- **Rely on the default browser behavior** – If you don't implement a Flash Player detection process, the browser will try to automatically install the player in many cases.

Users with Internet Explorer who do not have Flash Player your content requires will go through an "Autoinstall" process using the CAB file specified in the `codebase` attribute of the `<object>` tag. The CAB install will start if the version specified in your `<object>` tag is less than the version the user has.

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/flash/swflash.cab#v
   ersion=6,0,65,0
width="400"
height="200">
  <param name="src" value="example.swf">
</object>
```

The `#version` parameter tells the browser to look for that version to be installed. If the user does not have the appropriate version the browser will take over and install the latest version from Macromedia.

In Firefox, Netscape 8, and Mozilla based browsers a Cross-Platform Install (XPI) process will be invoked for users that do not have Flash Player. This process usually starts by showing a plug-in icon and the text, "Click here to get the plug-in." When the user clicks on the icon they are directed to an appropriate location, depending on the browser in use, where they can download and install Flash Player.

Flash Player is also available from the Firefox plug-in finder service, which can be used for new installs.

So users who do not have Flash Player installed at all will be prompted to install it before they can view your content. If the user already has an older version of Flash Player, or if they have a browser that does not support XPI, your content will be displayed, but it will not display properly. To handle these cases you should implement one of the detection processes to give users the best possible experience.

Using Flash Player Express Install

Express Install uses the existing Flash Player to upgrade users to the latest version of the Player when it is required for viewing content. As a developer, this means that you have the ability to not only detect when users do not have the latest version of Flash Player, but you can initiate an update that securely installs the latest version of the Player from the Macromedia web site. When the installation is complete, the user is directed back your web site, where they can view your Flash content. Express install also relies on the Client-side detection discussed earlier to make sure that the player required to start the process exists. In the case where the requirements are not met alternate content will be presented.

Note: Express Install requires that the user has Flash Player 6r65 or later installed on Mac OS or Windows, and that their browser have JavaScript enabled.

Express Install files

To use Express Install, integrate Express Install files into your HTML and SWF content. You can either use the HTML and SWF templates that come with the detection kit, or modify them to integrate with your web site's look and feel. The required files are:

- **playerProductInstall.html –** Web page that users will first encounter when attempting to access your Flash content. If they encounter this page with Flash Player 6.0r65, and do not have the minimum required version of Flash Player installed, an installation SWF (the installkit.swf file) that displays the update dialog prompt is downloaded to their browser from the Macromedia website.
- **playerProductInstall.swf –** SWF file that contains the ActionScript logic used to initiate Flash Player installation.
- **SWF content –** When the installation is complete, or if it the Player version your content requires is installed, this is the SWF file that the playerProductInstall.html file will load and play. The Flash Player Detection Kit includes the file example.swf for testing purposes.

The playerProductInstall.html file contains the JavaScript used to start the detection process. You must edit the JavaScript contained in this file to include the following information:

- The major version of Flash Player required to view your content
- The minor version and revision of Flash Player required to view your content
- The location of the content to view after installation has completed
- Alternate content to display in cases where Flash Player does not exist, or where a Player version earlier than 6.0r65 is installed

For example, if your content requires Flash Player 8.0, the JavaScript variables need to be modified as shown below:

```
// Globals
// Major version of Flash required
var requiredMajorVersion = 8;
// Minor version of Flash required
var requiredMinorVersion = 0;
// Minor version of Flash required
var requiredRevision = 0;
```

Next you can set the page that end users will be sent to after the install is complete. By default, the page will be the same as the location you have placed the playerProductInstall.html file:

```
// Location visited after installation is complete if installation is required
var MMredirectURL = window.location;
```

The script saves the current location in a variable named MMredirectURL and directs the browser back to this location when the install process is complete.

If you wish to direct users to a different web page, you must modify that line of JavaScript to specify the URL of your page:

```
// Location visited after installation is complete if installation is required
var MMredirectURL = "http://www.yoursite.com/yourcontent.html";
```

When you have configured the playerProductInstall.html file, you can immediately start using the Express Install process. Upload the required files (playerProductInstall.html, playerProductInstall.swf, and your content SWF) to the same directory on your web server, and navigate to the playerProductInstall.html page. If you already have Flash Player version that meets your requirements, you will see your Flash content correctly displayed. If you view the page using an older Flash Player, you will directed to the playerProductInstall.swf file, and the Flash Player installation process will begin.

Customizing Express Install

The Express Install experience can be customized beyond what Macromedia provides in this Flash Player Detection Kit. Macromedia provides an ActionScript callback function that you can change to create your own version of the playerProductInstall.swf file. This function is defined in the playerProductInstallCallback.as file.

The full set of source files for the Flash Player Express Install process includes:

- playerProductInstall.fla – FLA source file for Macromedia Flash MX 2004 and later.
- playerProductInstall.as – ActionScript source for playerProductInstall.fla.
- playerProductInstallCallback.as – ActionScript source containing the callback function dictating what happens when end users cancel, complete, or fail to install the new Flash Player.

```
function installStatus(statusValue) {
   if (statusValue == "Download.Complete") {
      // Installation is complete. In most cases the browser window
      // that this SWF is hosted in will be closed by the installer
      // or manually by the end user
   }
   else if (statusValue == "Download.Cancelled") {
      // The end user chose "NO" when prompted to install the new
      // player by default no User Interface is presented in this
      // case. It is left up to the developer to provide an
      // alternate experience in this case
   }
   else if (statusValue == "Download.Failed") {
      // The end user failed to download the installer due to a
      // network failure by default no User Interface is presented
      // in this case. It is left up to the developer to provide
      // an alternate experience in this case
   }
}
```

These functions are provided to give developers better control over the user experience. Over time as the process becomes more widely used this capability may be expanded to give developers even greater flexibility.

Conclusion

Flash Player detection has gone through different iterations over the years with no true definitive source. This kit serves not as a single solution to every detection need, but a source for some of the best solutions that have been created by the Flash community and Macromedia engineering. This kit provides a variety of solutions to common challenges that developers run into when deploying Flash content. Over time this document will grow and change, adding even more solutions to the ever growing Flash community and the platform that serves it.