Documentation

To place the player on your site, you need to know just a little about the general principles of video playback on the Internet and be able to use that site.

Player version

The current SPRUTO player version is 3.0.3

Version 3.0.3

Functions:

- Added support for subtitles
- Added support for social buttons
- Management from the keyboard is improved
- Minor bug fixes

Version 3.0.2

Functions:

- Minor bug fixes

Version 3.0.1

Functions:

- Minor bug fixes

Version 3.0

Functions:

- Added support for player skins
- Minor bug fixes

Version 2.1.3

Functions:

- Integrated with Google Analytics
- Minor bug fixes

Version 2.1.2

Functions:

- Minor bug fixes

Version 2.1.1

Functions:

- Minor bug fixes

Version 2.1.0

Functions:

- Playlist operation is changed
- Video recordings connection to Advarkads.com is added from Youtube.com
- Minor bugs are fixed

Version 2.0.3

Functions:

- Minor bug fixes

Version 2.0.2

Functions:

- Minor bug fixes

Version 2.0.1

Functions:

- Minor bug fixes

Version 2.0

Added:

- Support for switching video quality (SD through HD)
- Pre-integration with the Advark advertising system (Advarkads.com)
- Minor bug fixes

Version 1.2.1

Functions:

- Error correction of logo display in the player

Version 1.2

Functions:

- Added the ability to store player settings on your server
- Minor bug fixes

Version 1.1

Functions:

- HTML5-player version
- The logic to determine which platform tries to play the video and getting the right player
- Simplified JS-code form player (default settings concerning the behavior of the player is no longer displayed if they have changed)

Version 1.0

Functions:

- Flash player:
    - change player skins
    - change element colors
    - add your logo
    - create playlists
    - customizable video previews
    - customizable interface
    - customizable player options
    - supports video playback directly from YouTube.com, Myvi.ru, and Vimple.ru video links
    - player multi-language support
    - pseudo-streaming support (lighttpd)

General information

Video sharing

Video by nature uses a lot of space and bandwidth. Streams can range from 1 MB per min for low-quality videos to 20 MB per minute for HD quality videos. Moreover, before you start streaming video, you need to choose a convenient place to store your video files.

Among the myriad of options are:

1. **Using a free video hosting platform** (for example YouTube.com, Vimeo.com, Vimple.ru and others). Video hosting services all have their own rules and conditions and generally, they have a number of shortcomings:
    - They display their logo on the screen when the video is playing
    - They limit the size of the file that can be uploaded
    - In order to conserve space and bandwidth, they lower the quality of your video during conversion and playback (size, bitrate, compression)
    - They don't allow you to stream in your own player and/or their player design is very limited or not possible at all
    - They run their own advertisements before, after, or while watching video with no benefit to you
2. **Streaming your videos via an ordinary website hosting platform.** This option would enable you stream a video with its initial quality (i.e, there are no restrictions) assuming you have a video player such as Spruto. Logos are not displayed over the video and there are no ads. In this case, everything needed to play the video is there, except the necessary bandwidth as an ordinary hosting platform transmits small volumes of data (web pages) and so video streaming may be

quite slow and traffic limits would be reached quickly making it very uneconomical

3. **Streaming your videos via a CDN (Content Delivery Network).** This is a professional video hosting site – a network of servers used to serve large volumes of content. A CDN calculates the shortest path from the viewer to the closest server with the video and transmits the information from this server. You pay for these servers usually based on the number of video views and total amount of data transmitted. The more views your video receives, the more expensive a CDN becomes. High-traffic sites usually have to use a CDN due to their storage and bandwidth requirements (for example, video portals).

An important feature of a video storage server is that it supports pseudo streaming, that is, you can stream a video from any location. With pseudo streaming, if the connection fails while the video is playing, you can re-start playback from where you stopped, without having to go back to the beginning. Pseudo streaming also enables you to seek to not-yet-downloaded parts of a video. Depending on the video format, pseudo streaming can be implemented by passing different offset parameters – time offset or byte offset.

You can check whether pseudo streaming is supported only by actually playing a video (in the Flash version) and checking. This is due to security restrictions in the Flash technology. In the HTML5 version of the player, pseudo streaming is supported automatically.

Please note that a stand-alone video player is not a program for pre-processing or converting video files and cannot receive frames from a video to be used as a preview (like the video thumbnails you see on most video sharing hosting platforms). Therefore, we have installed a feature for your playlists that enables you to create a preview of frames to use as thumbnails.

Placing the player

There are two ways to place the video on your website:

1. **Inserting the code on your website.** With this method, the player files are stored with us, while only the player call code is stored on your site. This is suitable if your site offers short videos with small file sizes and experiences a limited number of daily views. If you just paste the player code into your website, you do not have to burden yourself with the finer intracacies of how the technology actually works. Moreover, the difference in the speed of the player's response to your users' requests is very small. Using this method, your website will always display the most current version of the player, thereby saving you from having to worry about updating it.
2. **Integrating the player files into your website.** This method is for advanced users who are creating sites with at least 10,000 video views per day. The player files are stored on your site and the response speed of the player will be slightly higher than in the first method. However, this requires a not insignificant level of technical prowess to implement. Using this method, you will need to update the player files anytime we release a new version. Download player files

Pseudo streaming

Pseudo streaming is the ability of the server to render a file not only from the beginning but also from a specified position. To do this, the server receives a request for the file with an additional parameter (usually "start"). If the server does not support pseudo streaming, it will just render the file from the beginning.
Advantages of pseudo streaming:

- you can seek to not-yet-downloaded parts of a video.
- you can start playback at any point of the video timeline.

- you can restart playback from where you stopped after a connection failure.

For popular servers, such as Nginx, Lighttpd, Apache, and IIS, there are ready-to-use solutions that support pseudo streaming for FLV and MP4 files. The player can disable pseudo streaming. If pseudo streaming is disabled, it cannot be used even if the server supports it. Disabling pseudo streaming is only recommended when your server does not support it.

Query parameter "start" is usually used to request a file from a certain position. The units of measure depend on the file type (bytes are used for flv files and seconds for mp4 files). These parameters are used in the player by default.

If your server requires other settings, you can specify them when configuring the player.You can separately specify a query parameter and units of measure for FLV and MP4 files. The player can request for offset in bytes or seconds.

To use pseudo streaming, the files must be prepared appropriately:

1. FLV file
    - Offset in bytes (used by default) – the file metadata must contain data on the key frames (they can be added to the file using metadata injectors such as flvmeta or yamdi)
    - Offset in seconds – the player does not require additional data.
2. MP4 file
    - Offset in seconds (used by default) – the MOOV atom (containing metadata) should be in the beginning of the file. (this atom is usually located at the end of the file after conversion. You can fix this using such tools as MP4Box)
    - Offset in bytes – it is recommended (but not a must) to place the MOOV atom at the beginning of the file, otherwise, the player will execute a separate request (with the required offset) for the file download.

Integrating player files in a site

Limitations

- For a user to launch the player, they will need to have Flash installed (no older than version 10.3)
- The player will not send the request for a video file until it receives a playback command **- load the player** if auto play is enabled or **press the play button** if auto play is disabled.*
- Any errors occurring as a result of an unsupported format or unavailability of the video will be appear only after an attempt to start playing the video has been made.*
- The player embed code passes all the settings of the player to the user's browser and the first video to be played is the video used when the code was created, not the first video in the playlist.

* — This may not be applicable for chrome-less players, such as Myvi.ru, Vimple.ru, and Youtube.com.

Manual configuration (embed code)

To manually create an embed code, you can use the following settings under the following conditions:

1. The player accepts parameters through one of the following ways:
    - In flashvars parameters
    - In flashvars `settingsUrl` parameter, a link to a text file containing the settings of the player in JSON format is passed. In this case, the other settings will be ignored.
2. You must specify the location of the video you want to play, all other parameters are optional.
3. When inadmissible parameters are passed (if the player cannot be correctly initialized), the player will issue an error message.
4. Playlist controls will become available in the player if there are at least two videos in the playlist.

5. A playlist containing one video is equivalent to transferring that video by the VideoURL parameter (with additional data in the relevant parameters).
6. To prevent showing the embed code via the player, you need to disable the GetCodeButton button.
7. Enabling the buttons PlayButton, PauseButton, StopButton, PrevButton, NextButton, ShowPlaylistButton, FollowPlaylistButton, and Volume, will create the respective items in the context menu.
8. Supported link types:
    - Direct link to the video. Http and https protocols.
    - Link to the video on Youtube.com. A link to the video page (http://www.youtube.com/watch?v=id) or a link to the player (http://www.youtube.com/v/id)
    - Link to the video page on Vimple.ru (http://vimple.ru/id))
    - Link to the video page on Myvi.ru. (http://www.myvi.ru/watch/video or http://myvi.ru/ru/videodetail.aspx?video=video)
9. **Video playback in multiple resolutions.** To add multiple links to the video in different qualities you will add multiple 'URLs' with differing playback resolution. The 'label' parameter specifies the quality of the video being uploaded. Label links for each video must be unique. The 'default' parameter specifies that this link will be used by default. If there is not a version marked as the default, the first uploaded version will be marked as default. In case individual labels aren't created, the labels '1', '2', and '3' will be automatically generated. For Myvi.ru, Vimple.ru, and YouTube.com videos, only the reference video is used; the other links are ignored.

Examples of an embed code:

Player without a playlist

```
<embed src='http://s3.spruto.org/embed/player.swf'
type='application/x-shockwave-flash' allowfullscreen='true'
```

allowScriptAccess='always' width='640' height='360' flashvars='set_video1_url=http%3A%2F%2Fvideo.domain.com%2Fvideo.mp4&set_title_text=Example%20video%20title&set_posterUrl=http%3A%2F%2Fposter.domain.com%2Fposter.jpg'/>

In this example, the following parameters were passed to the player:

- Location of the video: 'http://video.domain.com/video.mp4' (параметр set_video1_url)
- Title of the video 'Example video title' (параметр set_title_text)
- Location of the poster (a poster shown before the video starts playing): 'http://poster.domain.com/poster.jpg' (parameter set_posterUrl)

Player with a playlist

<embed src='http://s3.spruto.org/embed/player.swf' type='application/x-shockwave-flash' allowfullscreen='true' allowScriptAccess='always' width='640' height='360' flashvars='set_pl1_video1_url=http%3A%2F%2Fvideo.domain.com%2Fvideo1.mp4&set_pl1_title=Example%20video%20title%201&set_pl1_posterUrl=http%3A%2F%2Fposter.domain.com%2Fposter1.jpg&set_pl2_video1_url=http%3A%2F%2Fvideo.domain.com%2Fvideo2.mp4&set_pl2_title=Example%20video%20title%202&set_pl2_posterUrl=http%3A%2F%2Fposter.domain.com%2Fposter2.jpg&set_pl3_video1_url=http%3A%2F%2Fvideo.domain.com%2Fvideo3.mp4&set_pl3_title=Example%20video%20title%203&set_pl3_posterUrl=http%3A%2F%2Fposter.domain.com%2Fposter3.jpg&set_title_show=beforePlay&set_behavior_afterPlay=nextvideo'/>

The following playlist will be passed to the player:

- Video from file 'http://video.domain.com/video1.mp4', заголовок 'Example video title 1', poster (and a preview in the playlist) 'http://poster.domain.com/poster1.jpg'
- Video from file 'http://video.domain.com/video2.mp4', заголовок 'Example video title 2', poster (and a preview in the playlist) 'http://poster.domain.com/poster2.jpg'

- Video from file 'http://video.domain.com/video3.mp4', заголовок 'Example video title 3', poster (and a preview in the playlist) 'http://poster.domain.com/poster3.jpg'

The video title will be shown before the video plays (the set_title_show parameter is set to "beforePlay"). The next video will play automatically (the set_behavior_afterPlay parameter is set to nextvideo)

Example of logo settings

```
<embed src='http://s3.spruto.org/embed/transparent/player.swf' type='application/x-shockwave-flash' allowfullscreen='true' allowScriptAccess='always' width='640' height='360' flashvars='set_video1_url=http%3A%2F%2Fvideo.domain.com%2Fvideo.mp4&set_logo_imageUrl=http%3A%2F%2Fplayer.domain.com%2Flogo.png&set_logo_position=bl&set_logo_clickUrl=http%3A%2F%2Fhome.com'/>
```

The player loaded with these parameters, will display a logo (image taken from the location “http://player.domain.com/logo.png”) in the lower left corner. Clicking on this logo will open the page http://home.com in a new browser tab.

Example of settings with hidden player controls

```
<embed src='http://s3.spruto.org/embed/transparent/player.swf' type='application/x-shockwave-flash' allowfullscreen='true' allowScriptAccess='always' width='640' height='360' flashvars='set_video1_url=http%3A%2F%2Fvideo.domain.com%2Fvideo.mp4&set_hide=volumeButton%2Ctime%2CstopButton%2CshareCodeButton%2CfullscreenButton'/>
```

The player loaded with these parameters will not display some controls (volume control, current time and video duration, "Stop" button, get code button and full screen button). The "Stop Video" and "Turn on/off sound" items will appear on the context menu of the player. Besides, since a playlist was not passed to the

player, the next/previous, playlist display, on/off, next video auto play buttons will not be displayed.

Loading settings from a file

```
<embed src='http://s3.spruto.org/embed/player.swf' type='application/x-shockwave-flash' allowfullscreen='true' allowScriptAccess='always' width='640' height='360' flashvars='settingsUrl=http%3A%2F%2Fsettings.domain.com%2Fsettings.json'/>
```

The player will load JSON with settings from the file located at http://settings.domain.com/settings.json.

This file contains the following JSON:

```
  {
    'AfterPlay': 'start',
    'BeforePlay': 'playlist',
    'ColorScheme': 'light',
    'SkinName': 'islands',
    'LogoURL':'http://player.domain.com/logo.png',
    'LogoPosition':'br',
    'Playlist':[
     {
      'VideoURL':'http://video.domain.com/video1.mp4',
      'Title':'Video Title 1'
     },
     {
      'VideoURL':'http://video.domain.com/video2.mp4',
      'Title':'Video Title 2'
     },
     {
```

```
      'VideoURL':'http://video.domain.com/video3.mp4',
      'Title':'Video Title 3'
    }
  ]
}
```

After the player has loaded, a playlist will be shown immediately for you to select the video you want to play. A logo will be displayed in the lower right corner, but nothing happens when you click on it. The player design chosen differs from the default design ("Business" skin, light color scheme).

Manually configuring the player (JS code)

To manually create a JS code, do the following:

1. Connect the script http://s3.spruto.org/embed/player.js on the page where you intend to place the player.
2. 2. Place the script element with the "splayer" class on the page in the place where you intend to place the player. This element must contain a call to the method player.embed(params). Instead of params, you should place the configuring object of the player
    - b. Width (default value: 640) and Height (default value: 360) are additionally available. These options describe the size of the player.

An example of a JS code:

To insert the player on the page, you have to connect the script http://s3.spruto.org/embed/player.js by adding the following tag

```
<script type='text/javascript'
src='http://s3.spruto.org/embed/player.js'></script>
```

The player itself is placed using the following code:

```
<script class='splayer'> var params =
{'Language':'en','VideoURL':'http://video.domain.com/vide
o.mp4','Width':640,'Height':360,'ColorScheme':'light','Auto
play':true}; player.embed(params); </script>
```

The player placed using this code will display text messages and signatures on buttons in English language. The parameter "ColorScheme":"light" sets a light color scheme instead of a dark one (used by default). The video will start playing as soon as the page loads ("Autoplay":true)

Policy files

The policy file, which allows access from the domain and on which the player lies:

- is required if your video is located on a domain different from the domain where the player is located and you have not disabled pseudo streaming.
- is recommended for download of images (logo, previews and posters), located on other domains. If the policy file is absent, image will appear aliased.

Examples of policy files

A policy file must be located in the root of the site on which video files, configuration files and/or image files are located.

A policy file that contains the text given below allows access from any domain.`<cross-domain-policy> <allow-access-from domain='*'/> </cross-domain-policy>`

The following policy file allows access only from domain player.domain.com:`<cross-domain-policy> <allow-access-from domain='player.domain.com'/> </cross-domain-policy>`

More information about policy files can be found here

Supported formats

1. FLV container
    - Files not containing a video track are not supported
    - Video formats
        - Sorenson Spark
        - ON2 VP6
        - AVC/H.264 (The most modern format. Recommended)
        - Screen Video / Screen Video 2 (Lossless compression – very large file sizes)
    - Audio formats
        - Linear PCM (Little Endian) (lossless audio)
        - ADPCM
        - MP3 (The most common format)
        - Nellymoser
        - AAC (The most modern format. Better quality than MP3)
        - Speex
2. container MP4 (M4V/MOV/F4V)
    - Files not containing a video track are not supported
    - Video formats
        - AVC/H.264. Profiles: Baseline, Main, High, High 4:2:2
    - Audio formats
        - AAC. Profiles: Main, LC, HE/SBR

It is recommended to use videos with MP4 container using H.264 video codec and AAC audio codec because this format has hardware acceleration in most modern devices. Besides, H.264 and AAC codecs provide the best quality (compared with others supported by flash technology). Free codecs to convert video into H.264 standard – x264 and ffdshow.

Errors in the player

1. Player error. Check whether the video is available and whether the player is configured correctly.

    - If you are using a chromeless player, check whether the video is available, whether it can be used in embedded players, and check whether service is available.
    - If you are using your own server, check whether the player settings are correct
2. Video cannot be played. The file format is not supported.
    - Check whether the file meets the requirements listed under the Supported Formats section
3. Video cannot be played. The specified address is not available.
    - Check whether the file meets the requirements listed under the Supported Formats section
    - Check whether the video address is correct
    - Check whether the file is available
4. Security error when loading the video. Security policy is denying access to the video file.
    - Check whether there is a permitting policy file on the server with the video file
    - File is requested for on the local disk
5. Video cannot be played. The owner of the video has forbidden embedding the video.
    - Embedding the video is forbidden by the user. If this is your video, change its settings to allow embedding
6. Video cannot be played. The URL address is not specified.
    - The player has not received the video address. The address should be transferred for the player to work.
7. Video cannot be played. File contains invalid metadata.
    - TThe file cannot be played because the metadata in it are damaged. Use a metadata injector (for example, flvmeta or yamdi) to restore them.
8. Player settings error. Check the parameters passed.
    - Failed to load the player settings. Check the settings.
9. Failed to load the player settings. Invalid player settings format.
    - The player settings are incorrectly formatted and cannot be loaded.

10. Failed to load the player settings. The player settings file is not available.
    - Check whether the path to the settings file of the player is correct
    - Check whether the settings file of the player is available

Technical support

We understand that we cannot answer all your questions in advance. Therefore, if you cannot find the information you want in our documentation, please check our knowledge base in the technical support section or send us your question.