Post Formats

- Intro
- Supported Formats
- Function Reference
- Adding Theme Support
- Adding Post Type Support
- Using Formats
  - Suggested Styling
  - Formats in a Child Theme
- Backwards Compatibility
- Source File
- External Resources

Intro

Post Formats is a theme feature introduced with Version 3.1. A Post Format is a piece of meta information that can be used by a theme to customize its presentation of a post. The Post Formats feature provides a standardized list of formats that are available to all themes that support the feature. Themes are not required to support every format on the list. New formats cannot be introduced by themes or even plugins. The standardization of this list provides both compatibility between numerous themes and an avenue for external blogging tools to access this feature in a consistent fashion.

In short, with a theme that supports Post Formats, a blogger can change how each post looks by choosing a Post Format from a radio-button list.

Using **Asides** as an example, in the past, a category called Asides was created, and posts were assigned that category, and then displayed differently based on styling rules from post_class() or from in_category('asides'). With **Post Formats**, the new approach allows a theme to add support for a Post Format (e.g. add_theme_support('post-formats', array('aside')), and then the post

format can be selected in the Publish meta box when saving the post. A function call of get_post_format($post->ID) can be used to determine the format, and post_class() will also create the “format-asides” class, for pure-css styling.

Supported Formats

The following Post Formats are available for users to choose from, if the theme enables support for them.

Note that while the actual post content entry won’t change, the theme can use this user choice to display the post differently based on the format chosen. For example, a theme could leave off the display of the title for a “Status” post. How things are displayed is entirely up to the theme, but here are some general guidelines.

- **aside**: Typically styled without a title. Similar to a Facebook note update.
- **gallery**: A gallery of images. Post will likely contain a gallery shortcode and will have image attachments.
- **link**: A link to another site. Themes may wish to use the first `<a href="">` tag in the post content as the external link for that post. An alternative approach could be if the post consists only of a URL, then that will be the URL and the title (`post_title`) will be the name attached to the anchor for it.
- **image**: A single image. The first `<img>` tag in the post could be considered the image. Alternatively, if the post consists only of a URL, that will be the image URL and the title of the post (`post_title`) will be the title attribute for the image.
- **quote**: A quotation. Probably will contain a blockquote holding the quote content. Alternatively, the quote may be just the content, with the source/author being the title.
- **status**: A short status update, similar to a Twitter status update.
- **video**: A single video or video playlist. The first `<video>` tag or object/embed in the post content could be considered the video. Alternatively, if the post consists only of a URL, that will be the video URL. May also contain the video as an attachment to the post, if video support is enabled on the blog (like via a plugin).
- **audio**: An audio file or playlist. Could be used for Podcasting.
- **chat**: A chat transcript, like so:

```
John: foo
Mary: bar
John: foo 2
```

Note: When writing or editing a Post, Standard is used to designate that no Post Format is specified. Also if a format is specified that is invalid then standard (no format) will be used.

Function Reference

Main Functions: set_post_format(), get_post_format(), has_post_format().

Other Functions: get_post_format_link(), get_post_format_string().

Adding Theme Support

Themes need to use add_theme_support() in the *functions.php* file to tell WordPress which post formats to support by passing an array of formats like so:

```
add_theme_support( 'post-formats', array( 'aside', 'gallery' ) );
```

Note that you must call this before the init hook gets called! A good hook to use is the after_setup_theme hook.

Adding Post Type Support

Post Types need to use add_post_type_support() in the *functions.php* file to tell WordPress which post formats to support:

```
// add post-formats to post\_type 'page'
add_post_type_support( 'page', 'post-formats' );
```

Next example registers custom post type `my_custom_post_type`, and add Post Formats.

```
// register custom post type 'my_custom_post_type'
add_action( 'init', 'create_my_post_type' );
function create_my_post_type() {
  register_post_type( 'my_custom_post_type',
    array(
      'labels' => array( 'name' => __( 'Products' ) ),
      'public' => true
    )
  );
```

```
}

//add post-formats to post_type 'my_custom_post_type'
add_post_type_support( 'my_custom_post_type', 'post-formats' );
```

Or in the function register_post_type(), add `post-formats`, in `supports` parameter array. Next example is equivalent to above one.

```
// register custom post type 'my_custom_post_type' with 'supports' parameter
add_action( 'init', 'create_my_post_type' );
function create_my_post_type() {
  register_post_type( 'my_custom_post_type',
    array(
      'labels' => array( 'name' => __( 'Products' ) ),
      'public' => true,
      'supports' => array('title', 'editor', 'post-formats')
    )
  );
}
```

Using Formats

In the theme, make use of get_post_format() to check the format for a post, and change its presentation accordingly. Note that posts with the default format will return a value of FALSE. Or make use of the has_post_format() conditional tag:

```
if ( has_post_format( 'video' )) {
  echo 'this is the video format';
}
```

An alternate way to use formats is through styling rules. Themes should use the post_class() function in the wrapper code that surrounds the post to add dynamic styling classes. Post formats will cause extra classes to be added in this manner, using the “format-foo” name.

For example, one could hide post titles from status format posts by putting this in your theme’s stylesheet:

```
.format-status .post-title {
  display:none;
}
```

Suggested Styling

Although you can style and design your formats to be displayed any way you see fit, each of the formats lends itself to a certain type of “style”, as dictated by modern usage. It is well to keep in mind the intended usage for each format, as this will lend them towards being easily recognized as a specific type of thing visually by readers.

For example, the aside, link, and status formats will typically be displayed without title or author information. They are simple, short, and minor. The aside could contain perhaps a paragraph or two, while the link would probably be only a sentence with a link to some URL in it. Both the link and aside might have a link to the single post page (using the_permalink()) and would thus allow comments, but the status format very likely would not have such a link.

An image post, on the other hand, would typically just contain a single image, with or without a caption/text to go along with it. An audio/video post would be the same but with audio/video added in. Any of these three could use either plugins or standard Embeds to display their content. Titles and authorship might not be displayed for them either, as the content could be self-explanatory.

The quote format is especially well suited to posting a simple quote from a person with no extra information. If you were to put the quote into the post content alone, and put the quoted person’s name into the title of the post, then you could style the post so as to display the_content() by itself but restyled into a blockquote format, and use the_title() to display the quoted person’s name as the byline.

A chat in particular will probably tend towards a monospaced type display, in many cases. With some styling on the `.format-chat`, you can make it display the content of the post using a monospaced font, perhaps inside a gray background div or similar, thus distinguishing it visually as a chat session.

Formats in a Child Theme

Child Themes inherit the post formats defined by the parent theme. Calling add_theme_support() for post formats in a child theme must be done at a later priority than that of the parent theme and will **override** the existing list, not add to it.

```
add_action( 'after_setup_theme', 'childtheme_formats', 11 );
function childtheme_formats() {
```

```
  add_theme_support( 'post-formats', array( 'aside', 'gallery', 'link' ) );
}
```

Calling remove_theme_support('post-formats') will remove it all together.

Backwards Compatibility

If your plugin or theme needs to be compatible with earlier versions of WordPress, you need to add terms named `post-format-$format` to the `post_format` taxonomy. For example,

```
wp_insert_term( 'post-format-aside', 'post_format' );
```

You must also register the `post_format` taxonomy with register_taxonomy().

Source File

- wp-includes/post-formats.php

External Resources

- Styling Chat Transcript with WordPress Custom Post Format
- Post Types and Formats and Taxonomies, oh my!
- On standardized Post Formats
- Post Formats vs. Post Types
- Smarter Post Formats?
- WordPress Theme Support Generator

□□□ #2
□ □□□ □□□ 2026-10-31 18:04:32 UTC
□ □□□ □□□ 2026-10-31 18:04:34 UTC