JavaScript 文档标准

目录
需要记录的内容
文档编写技巧
语言规范
语法规范
格式规范
行换行
对齐注释
函数
Backbone classes
Local functions
本地祖先类
类成员
命名空间
行内注释
单行注释
多行注释
File Headers
支持的 JSDoc 标签
Unsupported JSDoc Tags

WordPress 遵循 JSDoc 3 标准 进行内联 JavaScript 文档编写。

需要记录的内容
WordPress 中的 JavaScript 文档采用格式化文档块或内联注释的形式。

以下是 WordPress JavaScript 文件中应记录的内容列表：

函数和类方法
对象
闭包
对象属性
依赖项
事件
文件头部信息

文档编写技巧

语言规范

简短描述应清晰、简洁、扼要。主要记录 "是什么 "和 "何时 "——通常无需包含 "为什么 "。如有必要， "为什么 " 可放在详细描述中。例如：

函数和闭包属于 第三人称单数 元素，这意味着应使用 第三人称单数动词 来描述它们的功能。

[tip] 需要帮助记忆第三人称单数动词的变位规则吗？可以想象在函数、钩子、类或方法摘要前加上 "它"：

正确示例 ： "(它) 执行某个操作。 "
错误示例 ： "(它) 执行某个操作。 "

[/tip]

函数说明 ：函数的功能是什么？

正确示例 ：处理 X 元素的点击事件。
错误示例 ：为 X 元素提供向后兼容性。

@since 标签 ：在 WordPress 中查找功能添加版本时，推荐使用 svn blame 工具。

如果使用这些工具后仍无法确定版本号，请使用 @since Unknown。

代码重构 ：修改文档时请勿重构文件中的代码。

语法规范

描述性内容应使用完整句子撰写。此标准唯一的例外是文件头摘要，其定位更接近于文件 "标题 "而非完整句子。

在摘要、描述、参数或返回值说明中列举多个元素时，应使用序列（牛津）逗号。

格式规范
为确保文档块内容能被正确解析并纳入代码参考，请遵循以下规范。

简短描述 ：

简短描述应为单句，且不包含任何标记。若描述涉及 HTML 元素或标签，应写作 “链接标签 ”而非 “<a>”
。例如： “在页眉中打印链接标签时触发 ”。

详细描述 ：

如有需要，可在详细描述中使用 Markdown 格式。

@param 与 @return 标签 ：

这些标签的描述中不允许使用 HTML 或 Markdown。 HTML 元素和标签应写作 “音频元素 ”或“链接标签 ”。

行换行

文档块文本应在超过 80 个字符后换行。如果文档块本身在左侧缩进 20 个字符位置，换行可能发生在第 100
个字符位置，但总宽度不应超过 120 个字符。

对齐注释

相关注释应通过间距调整实现对齐，以提升可读性。

例如：

/**
* @param {very_long_type} name 描述。
* @param {type} very_long_name 描述。
*/

函数
函数应按以下格式编写：

摘要 ：对函数用途的简要单行说明。结尾使用句号。
描述 ：对摘要的补充，提供更详细的描述。结尾使用句号。

@deprecated x.x.x：仅用于已弃用的函数，并提供该函数被弃用的版本号（应为三位数，例如 @deprecated 3.6.0），以及应使用的替代函数。
@since x.x.x：首次引入时应为三位数（例如 @since 3.6.0
）。如果进行了重大更改，应添加额外的 @since 标签、版本号和描述，以作为变更日志。 @access
：仅当函数为私有函数时使用。如果函数是私有的，则仅供内部使用，代码参考中将不会提供其文档
。
@class：用于类构造函数。
@augments：对于类构造函数，列出其直接父类。
@mixes：列出混入该对象的混入类。
@alias：如果此函数首先被赋值给一个临时变量，这允许您更改其文档记录的名称。 @memberof：如果 JSDoc 无法自动解析，则指定此函数所属的命名空间。 @static：对于类，用于标记函数是类构造函数上的静态方法。
@see：所依赖的函数或类。
@link：提供更多信息的 URL。
@fires：函数触发的事件。与特定类关联的事件应列出类名。
@listens：此函数监听的事件。事件必须带有事件命名空间前缀。与特定类关联的事件应列出类名。 @global：将此函数标记为全局函数，以包含在全局命名空间中。

@param：对变量的简要描述；使用 JSDoc @param 语法
注明细节（例如，变量是否为可选变量及其默认值）。结尾使用句号。

@yield：对于 生成器函数 ，描述预期从此函数生成的值。与其他描述一样，结尾包含句号。
@return：注意描述后的句号。

/**
* Summary. (use period)

*

* Description. (use period)

*

* @since x.x.x
* @deprecated x.x.x Use new_function_name() instead.
* @access private

*

* @class
* @augments parent
* @mixes mixin

*

* @alias realName
* @memberof namespace

*

* @see Function/class relied on
* @link URL

```
 * @global
 *
 * @fires   eventName
 * @fires   className#eventName
 * @listens event:eventName
 * @listens className~event:eventName
 *
 * @param {type}   var           Description.
 * @param {type}   [var]         Description of optional variable.
 * @param {type}   [var=default] Description of optional variable with default variable.
 * @param {Object} objectVar     Description.
 * @param {type}   objectVar.key Description of a key in the objectVar parameter.
 *
 * @yield {type} Yielded value description.
 *
 * @return {type} Return value description.
 */
```

Backbone classes

Backbone's `extend` calls should be formatted as follows:

- **`@lends`** This tag will allow JSDoc to recognize the `extend` function from Backbone as a class definition. This should be placed right before the Object that contains the class definition.

Backbone's `initialize` functions should be formatted as follows:

- **Summary**: A brief, one line explanation of the purpose of the function. Use a period at the end.
- **Description**: A supplement to the summary, providing a more detailed description. Use a period at the end.
- **`@deprecated x.x.x`**: Only use for deprecated classes, and provide the version the class was deprecated which should always be 3-digit (e.g. `@deprecated 3.6.0`), and the class to use instead.
- **`@since x.x.x`**: Should be 3-digit for initial introduction (e.g. `@since 3.6.0`). If significant changes are made, additional `@since` tags, versions, and descriptions should be added to serve as a changelog.
- **`@constructs`**: Marks this function as the constructor of this class.
- **`@augments`**: List direct parents.
- **`@mixes`**: List mixins that are mixed into the class.

- `@requires`: Lists modules that this class requires. Multiple `@requires` tags can be used.
- `@alias`: If this class is first assigned to a temporary variable this allows you to change the name it's documented under.
- `@memberof`: Namespace that this class is contained within if JSDoc is unable to resolve this automatically.
- `@static`: For classes, used to mark that a function is a static method on the class constructor.
- `@see`: A function or class relied on.
- `@link`: URL that provides more information.
- `@fires`: Event fired by the constructor. Should list the class name.
- `@param`: Document the arguments passed to the constructor even if not explicitly listed in `initialize`. Use a period at the end.
  - Backbone Models are passed `attributes` and `options` parameters.
  - Backbone Views are passed an `options` parameter.

```
Class = Parent.extend( /** @lends namespace.Class.prototype */{
/**
 * Summary. (use period)
 *
 * Description. (use period)
 *
 * @since      x.x.x
 * @deprecated x.x.x Use new_function_name() instead.
 * @access     private
 *
 * @constructs namespace.Class
 * @augments   Parent
 * @mixes      mixin
 *
 * @alias    realName
 * @memberof namespace
 *
 * @see   Function/class relied on
 * @link  URL
 * @fires Class#eventName
 *
 * @param {Object} attributes     The model's attributes.
 * @param {type}   attributes.key One of the model's attributes.
 * @param {Object} [options]      The model's options.
 * @param {type}   options.key One of the model's options.
 */
initialize: function() {
```

```
    //Do stuff.
}
} );
```

If a Backbone class does not have an initialize function it should be documented by using `@inheritDoc` as follows:

```
/**
 * Summary. (use period)
 *
 * Description. (use period)
 *
 * @since      x.x.x
 * @deprecated x.x.x Use new_function_name() instead.
 * @access     private
 *
 * @constructs namespace.Class
 * @memberOf   namespace
 * @augments   Parent
 * @mixes      mixin
 * @inheritDoc
 *
 * @alias    realName
 * @memberof namespace
 *
 * @see   Function/class relied on
 * @link  URL
 */
Class = Parent.extend( /** @lends namespace.Class.prototype */{
// Functions and properties.
} );
```

> Note: This currently doesn't provide the expected functionality due to a bug with JSDoc's inheritDoc tag. See JSDocs3 issue 1012.

Local functions

At times functions will be assigned to a local variable before being assigned as a class member. Such functions should be marked as inner functions of the namespace that uses them using `~`. The functions should be formatted as follows:

```
/**
 * Function description, you can use any JSDoc here as long as the function remains private.
 *
 * @alias namespace~doStuff
 */
var doStuff = function () {
// Do stuff.
};

Class = Parent.extend( /** @lends namespace.Class.prototype */{
/**
 * Class description
 *
 * @constructs namespace.Class
 *
 * @borrows namespace~doStuff as prototype.doStuff
 */
initialize: function() {
//Do stuff.
},

/*
 * This function will automatically have it's documentation copied from above.
 * You should make a comment ( not a DocBlock using /**, instead use /* or // )
 * noting that you're describing this function using @borrows.
 */
doStuff: doStuff,
} );
```

本地祖先类

有时类会拥有仅被赋值给局部变量的祖先类。这类祖先类应被分配到其子类所在的命名空间，并使用 `~` 符号设为内部类。

类成员

类成员应按以下格式编写：

- 简短描述 ：结尾使用句号。
- `@since x.x.x`：首次引入时应为 3 位数字（例如 `@since 3.6.0` ）。如果进行了重大更改，应添加额外的 `@since` 标签、版本号和描述，以作为变更日志。
- `@access`：如果成员是私有、受保护或公共的。私有成员仅供内部使用。
- `@type`：列出类成员的类型。
- `@property`：如果它是一个对象，则列出此对象的所有属性。结尾使用句号。
- `@member`：可选，用于覆盖 JSDoc 的成员检测，替代 `@type` 以强制指定一个类成员。
- `@memberof`：可选，用于覆盖此成员所属的类。

```
/**
 * 简短描述。（使用句号）
 *
 * @since  x.x.x
 * @access (private, protected, or public)
 *
 * @type     {type}
 * @property {type} key 描述。
 *
 * @member   {type} realName
 * @memberof className
 */
```

命名空间

命名空间应按以下格式编写：

- 简短描述 ：以句号结尾。
- `@namespace`：将此符号标记为命名空间，可选择提供名称作为覆盖。
- `@since x.x.x`：首次引入时应为 3 位数字（例如 `@since 3.6.0` ）。如果进行了重大更改，应添加额外的 `@since` 标签、版本和描述，以作为变更日志。
- `@memberof`：此命名空间所属的父命名空间。
- `@property`：此命名空间公开的属性。以句号结尾。

```
/**
 * 简短描述。（使用句号）
```

```
 *
 * @namespace realName
 * @memberof  parentNamespace
 *
 * @since x.x.x
 *
 * @property {type} key 描述。
 */
```

行内注释

方法或函数内部的行内注释应遵循以下格式：

单行注释

```
// 提取数组的值。
```

多行注释

```
/*
 * 这是一个足够长的注释，需要跨越多行。
 * 您会注意到这基本上遵循与 JSDoc 包装和注释块样式相同的格式。
 */
```

重要提示：多行注释不得以 `/**`（双星号）开头。请改用 `/*`（单星号）。

File Headers

The JSDoc file header block is used to give an overview of what is contained in the file.

Whenever possible, all WordPress JavaScript files should contain a header block.

WordPress uses JSHint for general code quality testing. Any inline configuration options should be placed at the end of the header block.

```
/**
 * Summary. (use period)
 *
 * Description. (use period)
 *
 * @link   URL
 * @file   This files defines the MyClass class.
 * @author AuthorName.
 * @since  x.x.x
 */

/** jshint {inline configuration here} */
```

□□ JSDoc □□

Tag	Description
`@abstract`	This method can be implemented (or overridden) by the inheritor.
`@access`	Specify the access level of this member (private, public, or protected).
`@alias`	Treat a member as if it had a different name.
`@augments`	This class inherits from a parent class.
`@author`	Identify the author of an item.
`@borrows`	This object uses something from another object.
`@callback`	Document a callback function.
`@class`	This function is a class constructor.
`@classdesc`	Use the following text to describe the entire class.
`@constant`	Document an object as a constant.
`@constructs`	This function member will be the constructor for the previous class.
`@copyright`	Document some copyright information.
`@default`	Document the default value.
`@deprecated`	Document that this is no longer the preferred way.
`@description`	Describe a symbol.

Tag	Description
`@enum`	Document a collection of related properties.
`@event`	Document an event.
`@example`	Provide an example of how to use a documented item.
`@exports`	Identify the member that is exported by a JavaScript module.
`@external`	Document an external class/namespace/module.
`@file`	Describe a file.
`@fires`	Describe the events this method may fire.
`@function`	Describe a function or method.
`@global`	Document a global object.
`@ignore`	[todo] Remove this from the final output.
`@inner`	Document an inner object.
`@instance`	Document an instance member.
`@kind`	What kind of symbol is this?
`@lends`	Document properties on an object literal as if they belonged to a symbol with a given name.
`@license`	[todo] Document the software license that applies to this code.
`@link`	Inline tag - create a link.
`@member`	Document a member.
`@memberof`	This symbol belongs to a parent symbol.
`@mixes`	This object mixes in all the members from another object.
`@mixin`	Document a mixin object.
`@module`	Document a JavaScript module.
`@name`	Document the name of an object.
`@namespace`	Document a namespace object.
`@param`	Document the parameter to a function.
`@private`	This symbol is meant to be private.
`@property`	Document a property of an object.
`@protected`	This member is meant to be protected.
`@public`	This symbol is meant to be public.
`@readonly`	This symbol is meant to be read-only.

Tag	Description
`@requires`	This file requires a JavaScript module.
`@return`	Document the return value of a function.
`@see`	Refer to some other documentation for more information.
`@since`	Documents the version at which the function was added, or when significant changes are made.
`@static`	Document a static member.
`@this`	What does the 'this' keyword refer to here?
`@throws`	Describe what errors could be thrown.
`@todo`	Document tasks to be completed.
`@tutorial`	Insert a link to an included tutorial file.
`@type`	Document the type of an object.
`@typedef`	Document a custom type.
`@variation`	Distinguish different objects with the same name.
`@version`	Documents the version number of an item.
`@yield`	Document the yielded values of a generator function.

Unsupported JSDoc Tags

Tag	Why it's not supported
`@summary`	Should not be used. See the example of how to separate a summary from the full description.
`@virtual`	An unsupported synonym. Use `@abstract` instead.
`@extends`	An unsupported synonym. Use `@augments` instead.
`@constructor`	An unsupported synonym. Use `@class` instead.
`@const`	An unsupported synonym. Use `@constant` instead.
`@defaultvalue`	An unsupported synonym. Use `@default` instead.
`@desc`	An unsupported synonym. Use `@description` instead.
`@host`	An unsupported synonym. Use `@external` instead.
`@fileoverview`	An unsupported synonym. Use `@file` instead.
`@overview`	An unsupported synonym. Use `@file` instead.
`@emits`	An unsupported synonym. Use `@fires` instead.
`@func`	An unsupported synonym. Use `@function` instead.

Tag	Why it's not supported
`@method`	An unsupported synonym. Use `@function` instead.
`@var`	An unsupported synonym. Use `@member` instead.
`@emits`	An unsupported synonym. Use `@fires` instead.
`@arg`	An unsupported synonym. Use `@param` instead.
`@argument`	An unsupported synonym. Use `@param` instead.
`@prop`	An unsupported synonym. Use `@property` instead.
`@returns`	An unsupported synonym. Use `@return` instead.
`@exception`	An unsupported synonym. Use `@throws` instead.

□□□ #2
□ □□□ □□□ 2026-03-27 01:08:34 UTC
□ □□□ □□□ 2026-03-27 01:08:34 UTC