Creating a Plugin

Pinegrow plugins are Javascript functions that utilize the Pinegrow API to extend the capabilities of the editor. All Pinegrow plugins are instantiated by calling a function, either anonymous or named, once the Pinegrow App signals it is ready. This function gets passed the event and the `pinegrow` window variable as arguments.

```
$(function() {
    $('body').one('pinegrow-ready', function(e, pinegrow) {
        //plugin body
    });
});
```

or

```
$(function() {
    $('body').one('pinegrow-ready', function(e, pinegrow) {
        myFunctionName(pinegrow);
    });
});
function myFunctionName(pinegrow){
    //plugin body
};
```

Within the body of the plugin, a new framework object is created using the `PgFramework` constructor and passing in a unique plugin id that will be prefixed internally, along with a name for the framework.

```
var framework = new PgFramework(plugin_id, name);
```

This framework variable can then be populated with a variety of key:value pairs. Some of these pairs are informational, like `framework.author`, which will be displayed to the end user, or give parameters to the Pinegrow App about how the plugin is supposed to be managed or displayed. The most important of these are listed below in the section **framework descriptive elements**. Other pairs add the individual library components, items to the property panel, or actions panel. The most important of these are listed on the **Components** and **Sections and Fields**.

Typically, the descriptive key:value pairs for the framework are defined prior to returning the new object to the Pinegrow App using the `addFramework()` function.

```
pinegrow.addFramework(framework);
```

Framework Descriptive Elements

type

This key takes a value identifying the framework - usually the same as the plugin id passed into the framework, but it can also be used to delineate different versions or "types" of the same framework. For example, all of the included versions of the Bootstrap framework have a `type` of 'bootstrap', but a plugin id unique to their version - 'bs3.4.1' or 'bs4'. Defaults to the passed in key.

```
framework.type = 'key';
```

allow_single_type

This key takes a boolean value, usually "true", that prevents activation of multiple frameworks of the same `type`. Defaults to "false".

```
framework.allow_single_type = true;
```

description

This key takes an HTML or plain text string describing the plugin and can contain a link that is displayed when creating a new page using a template from the plugin.

```
framework.description = '<a href="http://my.website.com/">Custom plugin</a> that
```

author

This key takes an HTML or plain text string with the author's name and is displayed when creating a new page using a template from the plugin.

```
framework.author = '<em>Pinegrow</em>';
```

author_link

This key takes a URL link that will be opened if the author name is clicked in the new page pop-up.

```
framework.author_description = 'https://pinegrow.com';
```

info_badge

This key takes a short string that will be displayed when creating a new page using a template from the plugin. This is useful for displaying version numbers, for example.

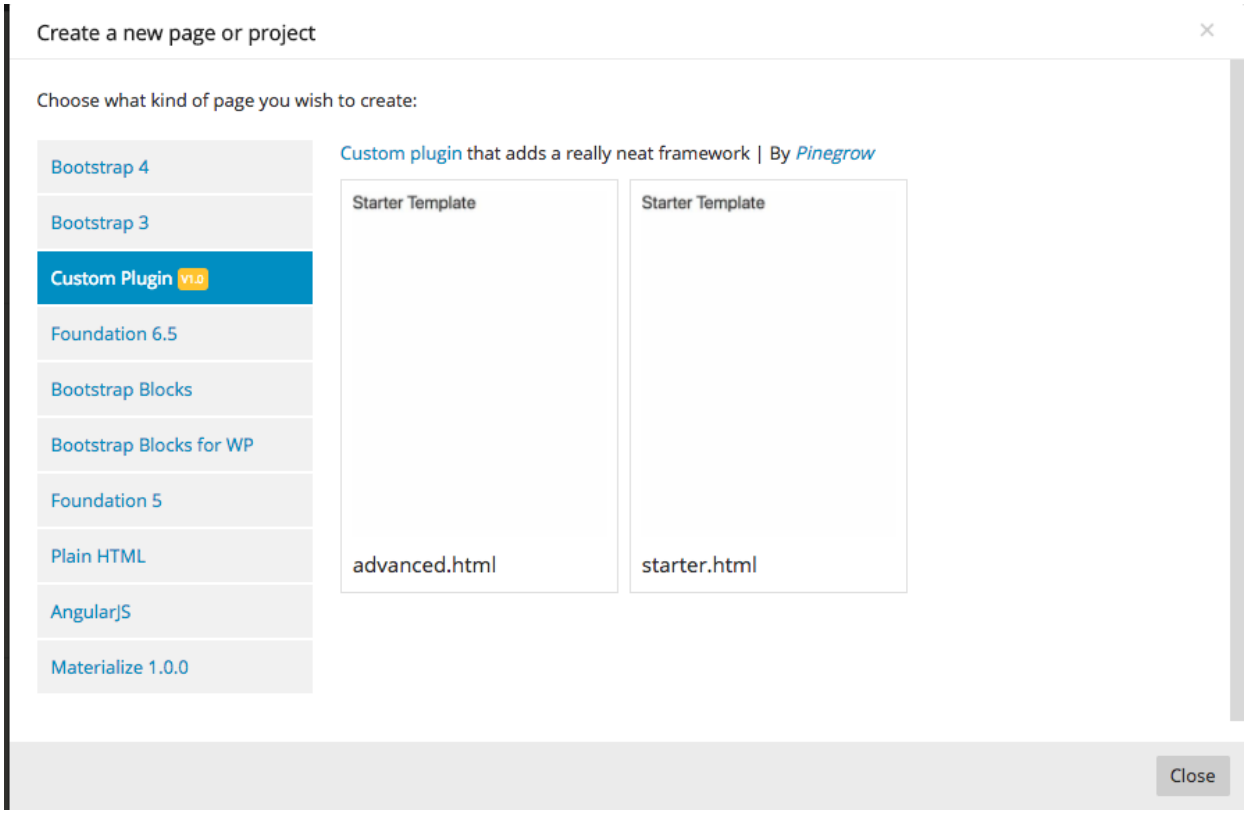
```
framework.info_badge = 'v1.0.0';
```

Typical example of basic framework instantiation:

```
$(function() {
    $('body').one('pinegrow-ready', function(e, pinegrow) {

        var plugin_id = 'pge';

        var framework = new PgFramework(plugin_id, 'Pinegrow Example');

                framework.type = plugin_id;
                framework.allow_single_type = true;

                framework.description = '<a href="http://my.website.com/">Custom
                framework.author = '<em>Pinegrow</em>';
                framework.author_description = 'https://pinegrow.com';
                framework.info_badge = 'v1.0';
```

```
        pinegrow.addFramework(framework);

    });
});
```

As shown below, this descriptor information will only be displayed when creating a plugin that adds a template for the user to select when starting a new page or project. It will not be displayed if the plugin only adds HTML snippets, actions, modifies workflow, or adds to the CMS.



Once your framework is created you can add optional templates and resources that are made available to the user. This is documented on the Templates and Resources page. Additionally, you can now begin adding components and sections and fields that control page elements.

Next: Templates and Resources