jQuery Plugin Development: Getting Started

Zach Watkins

Who?

- > 6yrs xp, mostly JavaScript
- Implemented / troubleshot 3rd party plugins
- Read lots of API docs
- Developed many JS plugins (incl. jQuery)
 - modal window
 - content sliders
 - form data autofill
 - countdown

- google maps
- data templating
- search boxes
- JSON API handling

Overview

- 1. Basic jQuery plugin structure
- 2. User Extensions
- 3. Documentation
- 4. Github Repository
- 5. Publish to NPM

Basic jQuery Plugin Structure

- 1. Isolate your functionality
- 2. Wrap in plugin structure that allows chaining

```
;( function( $ ){
    $.fn.yourname = function() {
      return this.each( function() {} );
    };
} )( jQuery );
```

User Extensions

aka "Advanced Functionality"

- Settings
 - o \$("body").gumboot({separator: "-"});
- Default Settings
 - o jQuery.fn.gumboot.defaults = {};
- Methods
 - o jQuery.fn.gumboot.stylize = function(){};
- Callback

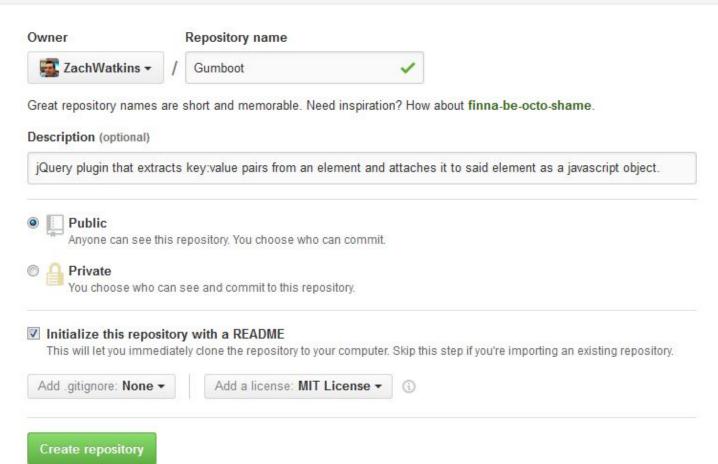
Documentation

- 1. Describe the plugin's purpose
- 2. Each setting name, object type(s), description(s)
 - a. settingname (default: ")Type: String or Array of StringsA string or array of strings that the plugin uses to ...
- 3. Example user-defined defaults or methods
- 4. Use cases with plugin declarations

Github Repository

(if desired)

- 1. Create account / log in
- 2. New repository
- 3. README.md
- 4. Use repository link for package.json



Publishing to NPM

- 1. \$ npm adduser
 - a. username, password, and email address
- 2. \$ npm init
 - a. Use keywords "jquery-plugin" and "ecosystem:jquery"
- 3. \$ npm publish

Questions?

codepen.io/zw/pen/YXLoWj

- @zach_wat
- Github.com/ZachWatkins
- remarklet.com