Widget Punch List

1. Must support publicly documented Widget Factory Methods.
    - destroy
    - disable
    - enable
    - option
    - widget
2. For dom based widgets: make sure you have a public refresh method.
3. Widgets should be stateful and support dynamic setting of all options.
    - keep options to a minimum you will have to support these being dynamically set
4. Make sure to fully utilize widget factory private methods
    - _create
    - _delay
    - _destroy (MUST)
    - _focusable
    - _getCreateEventData
    - _getCreateOptions
    - _hide
    - _hoverable
    - _init
    - _off (MUST)
    - _on (MUST)
    - _setOption
    - _setOptions
    - _show
    - _super
    - _superApply
    - _trigger (MUST)

Widget Best Practices

5. consider making private methods rather than nested functions
    - better for unit testing
    - can be extended
    - better organization
6. For things that should not be set per widget consider moving to a defaults object ie initSelector
7. All bindings should be done with _on and _off
8. trigger all events with _trigger
9. Consider extensions rather than adding functionality
    - allows exclusion in download builder
    - keeps code focused and extensible
10. Make sure destroy leaves dom as it was before init

11. rather than large bloated methods try to keep methods discreet and testable
    - also aids extensibility
12. Consider accessibility support all possible input methods (touch, mouse, keyboard, etc)
    - test on screen reader
13. Favor css over JS for display
    - if something is possible with just a css class this is better than a js option
    - only things that should be set with .css() are positions and dimensions
14. Performance should always be a top priority
15. does this widget make sense outside of a page?
    - if so it should work standalone
16. Progressive enhancement what happens if JS is disabled?
    - if its a form element it must still be functional
    - should not impede page function
    - Make every effort to still look as good as possible
17. use widget pseudo selector to find other instances (:widgetname)
    - this is only safe inside a widget looking for the same widgets
    - if outside the same widget use :data(widgetname)
18. use private methods to set options
    - these can then be called from _create, setoption, refresh etc
    - easy to test
19. avoid anonymous functions they are hard to test and debug
20. Make variable and function names that are descriptive of the value / functionality
21. Follow jQuery Core style guide http://contribute.jquery.org/style-guide/js/
22. Check to ensure the API docs and demos are in sync with the final widget code
23. allow a non enhance option for initializing widgets with pre-enhanced markup
24. handle all markup enhancement in refresh makes for easy option setting
25. if a widget may be used in a global context (on every page) make it able to live outside the page
    - allows it to not have to be sent and enhanced on every page load.
26. data-attributes should only be referenced on widget creation do not support dynamic data-attribute options.

24.for slider like widgets it should be possible to base on div instead of input
    - best approach for input attributes https://github.com/jquery/jquery-ui/blob/master/ui/jquery.ui.spinner.js#L75

25. Plugins must work regardless of what options are passed to init. All options are truly optional and may be changed at any time.
26. Don't save any data on the element other than the plugin instance.
    - An exception to this rule is if you need to store data on subordinate elements.
27. **Use jQuery.ui.keyCode for all keyboard events.**
28. Do not abbreviate variable names; the bytes will be saved when compressed.
29. Do not modify options that a user passes in, except when they are used to reflect the state of the plugin.

Possible additions:

- Flag to disable ARIA attributes for use in hybrid apps where ARIA is potentially confusing. Be good be be able to set this as a global config in addition to being at the widget level

Performance Best practices

- references are always faster than selectors consider extending the widget with references to elements that will be used throughout
- the jQuery() function has a lot of call overhead store references to objects that will get reused in variables even if its only twice.
- remember $.mobile.window and $.mobile.document
- it is often preferable to use native dom methods rather than jQuery but browser testing is a MUST!! ex. get / set attribute
- use $.data vs $.fn.data
- always use tag names in selectors where possible tag name is the second fastest jquery selector div.foo is always faster than .foo
- limit dom manipulation as much as possible. create complete objects and only append once
- consider using clone its faster than creating new elements from scratch.
- use dom documentFragments where appropriate this is much faster http://ejohn.org/blog/dom-documentfragments/
- make sure to properly delegate events
- chain jquery function calls
- pass object literals to functions like .css() to prevent multiple calls
- add and remove classes rather than .css where possible
- never append content in a loop
- pseudo and attribute selectors are the slowest selectors
- consider detaching an element to manipulate it then reinsert this is much faster than direct manipulation
- remember sizzle checks selectors right to left. make rightmost part of selector most specific leftmost more broad
- avoid convenience methods like getJSON() which calls get() which calls ajax(). Just use ajax()