# fitbit-subclass-widget

### Structure

#### Member Hiding

widget may need to expose a (public) API but keep other members private. It therefore needs a closure. construct() should return the widget closure (API).

#### Object Structure

widget

* private members
* widgetAPI object
* public members

#### Prototype Chain

* widgetAPI (object within widget object)
* el (fitbit SVG element object)

### Issues

If we want useEl objects returned by getElementById to function as widgets, the widget can’t be a parent of useEl.

* useEl (calls to Fitbit API .x can only be made via this object, but .x isn’t implemented directly in this object)
* widgetAPI (we want to provide an alternative .x here, that can call the superclass (prototype) .x)
* useEl.[[prototype]] (.x implemented here or in some subclass/prototype)

Could only work if useEl doesn’t contain any useful members.

May interfere with inheritance because widgetAPI might want to use properties defined on useEl, but if they’re not defined on useEl itself, they’ll be called on widgetAPI: recursion? Fitbit API seems to require that calls only be made against useEl rather than a prototype.

Can our version of .x detect when its specific functionality is wanted vs. whether the superclass version? Inspect `this`?

Can bind(), etc, be used to specify useEl? Can bind(), etc, be used with setters?

Could implement document.getWidgetById(). Fitbit element objects may need a pointer to the widget object (once created) to prevent repeated creation if getWidget... is called again on the same object.

Think of a realistic example. Must be a member of useEl; eg, x, style.

Properties available on useEl:

* .x, .y, .width, .height (from Bounded interface?)
* style (from Styled interface?)
* .text, .textContent, .image, .value, .mode, .state. getBBox, .parent, .nextSibling, .firstChild, .children, .sendEvent, .animate, .layer, .type (from interface GraphicsElement?)
* .getElementById/ClassName/TypeName/Tagname (from ElementSearch interface)
* .highlight, .textContent, .innerText, .appendChild (from unknown interface)
* .id, .class (from Element interface)

#### Restrictive vs Permissive

Restrictive: all superclass (prototype) members are unavailable, except for those passed through from the widget.

Permissive: all superclass (prototype) members are available, except for those blocked by the widget.

#### CSS

If we over-ride useEl members at widget (subclass) level, what are the implications for property assignments in SVG and CSS? Presumably the useEl implementations would be used, rather than those of the widget. Could kludge it using config (like HTML style).

### <use> Structure

| **Level 0:** |  |
| --- | --- |
| **Level 1:** |  |
| strokeWidth (undefined) | lines.\_closure |
| setPoint (function) | lines.\_closure |
| **Level 2:** |  |
| constructor (function) | — |
| text (string) | Element |
| image (string) | Element |
| value (undefined) | Element |
| mode (number) | Element |
| state (undefined) | Element |
| highlight (INACCESSIBLE) | ??? |
| style (object) | Styled |
| x (number) | Bounded |
| y (number) | Bounded |
| width (undefined) | Bounded |
| height (undefined) | Bounded |
| **Level 3:** |  |
| constructor (function) | — |
| getBBox (function) | GraphicsElement |
| **Level 4:** |  |
| constructor (function) | — |
| parent (object) | Element (Node has parentNode) |
| nextSibling (object) | Element (or Node) |
| firstChild (object) | Element (or Node) |
| children (object) | Element (Node has childNodes) |
| text (string) | Element |
| textContent (INACCESSIBLE) | TextElement(!!!) or Node(!!!) |
| innerText (INACCESSIBLE) | HTMLElement(!!!) |
| getElementById (function) | ElementSearch |
| getElementsByClassName (function) | ElementSearch |
| getElementsByTypeName (function) | ElementSearch |
| getElementsByTagName (function) | ElementSearch |
| sendEvent (function) | Element |
| appendChild (function) | Node(!!!) |
| animate (function) | Element |
| id (string) | Element |
| class (string) | Element |
| type (string) | Element |
| layer (undefined) | Element |
| **Level 5:** |  |
| constructor (function) | — |
| addEventListener (function) | Element? EventTarget? |
| removeEventListener (function) | EventTarget |
| dispatchEvent (function) | EventTarget |
| **Level 6:** |  |

#### Interfaces

TextElement (probably not used) extends GraphicsElement

GraphicsElement extends Element, Styled, Bounded

Element extends ElementSearch, GlobalEvents

ElementSearch

GlobalEvents extends EventTarget

EventTarget ([JS standard](https://developer.mozilla.org/en-US/docs/Web/API/EventTarget))

Styled

Bounded

Node ([Web standard](https://developer.mozilla.org/en-US/docs/Web/API/Node))

### Structure Options

useEl: the JS object returned by getElementById(‘idOfUseElement’).

These options may apply at multiple levels; eg:

* widget itself
* whether widget returns an object that corresponds to one of its constituent elements (eg, a TextElement object) or a different object that can control access to the Element object
* whether widget returns a Fitbit Style object or a different object named style that can control access to the Fitbit Style object

|  | **Widget IS THE useEl** | **Widget IS A KIND OF useEl** | **Widget HAS A useEl** |
| --- | --- | --- | --- |
| Object structure | widget is not a distinct object, but adds its own properties to useEl | widget is a distinct object that has useEl as its prototype | widget is a distint object that has useEl as a member |
| Example | curvedText | lines | FitFont |
| Can call Fitbit API on object returned by getElementById? | Yes (unless explicitly overridden in widget) | No | No (unless explicitly implemented in widget) |
| Modifies Fitbit element object? | Yes | No | No |
| widget needs to implement pass-through code for every Fitbit API member it supports | No | No | Yes |
| Support for private properties and methods |  |  |  |
| Can access useEl members if overridden in widget? | No (Fitbit API calls can only be made on useEl, not a prototype) | Yes (in theory; untested) | Yes |

### To Do