Simple Web Runtime Design Patterns Using YUI 3

Introduction

Design patterns are “known solutions to common design problems”. The following design patterns make use of the YUI version 3 JavaScript™ library and are intended for designers and developers working with Web Runtime (WRT) and the S60 mobile browser.

“YUI 3 is Yahoo!'s next-generation JavaScript and CSS library; it powers the new Yahoo! homepage and incorporates what we've learned in five years of dedicated library development. Today, the YUI 3 Core and a full utility set are ready for use. All YUI projects are BSD-licensed and are available for forking and contribution on GitHub.” developer.yahoo.com

The YUI Library is fully documented on its website; detailed API documentation accompanies the library download. It has six types of components: YUI core, utilities, UI controls, CSS components, developer tools, and build tools. To get started visit the YUI developer page at developer.yahoo.com/yui/3/

Note: The following examples have been tested on the 5th Edition, N97 device. Support for YUI will vary amongst 3rd Edition devices.

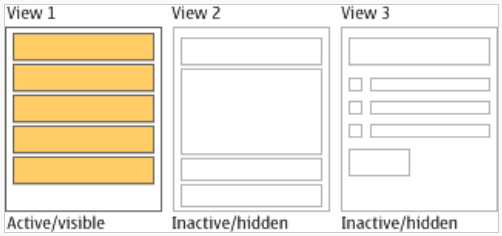
1. Show/Hide design pattern

Problem summary A user must interact with multiple views of an application yet only one view can be visible on-screen at a time.

Use

- To switch views within the widget.
- To show and hide elements within a view.
- Within a tabbed interface; to show and hide tabbed views.

Solution Provide a user or system-initiated trigger to instantly show or hide views or elements within the widget.



This widget has three views yet only one can be active at a time

You can use the normal display block/none for the toggling Effect.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title></title>
 <script type="text/javascript" src="http://yui.yahooapis.com/3.0.0/build/yui/yui-
min.js"></script>
   <script type="text/javascript">

  var myY = null;

  YUI().use('node', doInit);

  function showList(){
    myY.one('#list').setStyle('display','block');
    myY.one('#detail').setStyle('display','none');
  }
```


(C) Copyright Nokia 2014. All rights reserved.

```
  function showDetails(){
    myY.one('#detail').setStyle('display','block');
    myY.one('#list').setStyle('display','none');
  }

  function showToggle(){
   if(myY.one('#detail').getStyle('display') == 'block'){
    myY.one('#list').setStyle('display','block');
    myY.one('#detail').setStyle('display','none');
   }else{
    myY.one('#detail').setStyle('display','block');
    myY.one('#list').setStyle('display','none');
   }
  }

  function doInit(Y){

   myY = Y;
   myY.get('#btn_list').on('click', showList);
   myY.get('#btn_detail').on('click', showDetails);
   myY.get('#btn_toggle').on('click', showToggle);

   myY.one('#list').setStyle('display','block');
  }

   </script>

   <style type="text/css">
    div {
   height: 5em;
   display: none; }
  div#list {
   background:#f00; }
  div#detail {
   background:#0f0; }
   </style>
</head>
<body>
  <button id="btn_list">List</button>
  <button id="btn_detail">Detail</button>
  <button id="btn_toggle">Toggle</button>
  <div id="list">
   <h1>List View</h1>
  </div>

  <div id="detail">
   <h1>Detail View</h1>
  </div>
</body>
</html>
```

WRT Widget for the example code can be found from File: File:Toggle YUI.zip

2. Slide transition

Problem summary

(C) Copyright Nokia 2014. All rights reserved.

Abrupt changes in view or state are uncomfortable and often disorienting to the user.

Use

- To gradually hide an item within a view.

Solution Create a connection between the original and final view (or state) through use of an animated transition.

The slide transition is used to gradually close the content area

You can use the Y.Anim API for changing the height with timer and easing algorithm to get thi Effect.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title></title>
 <script type="text/javascript" src="http://yui.yahooapis.com/3.0.0/build/yui/yui-
min.js"></script>
   <script type="text/javascript">

  YUI().use('anim', function(Y) {

   var anim1 = new Y.Anim({
           node: '#slide',
           to: { height: 0 },
           duration: 2.0,
       });

   var anim2 = new Y.Anim({
          node: '#slide',
           to: { height: 100 },
           duration: 2.0,
       });

   Y.get('#btn_slide').on('click', anim1.run, anim1);
   Y.get('#btn_reset').on('click', anim2.run, anim2);
  });
   </script>
  <style type="text/css">
   div {
  height: 100px;
  background: #f00; }
  </style>
</head>
<body>
   <button id="btn_slide">Slide</button>
   <button id="btn_reset">Reset</button>
  <div id="slide"></div>
  <h1>Slide</h1>
</body>
</html>
```


(C) Copyright Nokia 2014. All rights reserved.

WRT Widget for the example code can be found from File: File:Slide YUI.zip

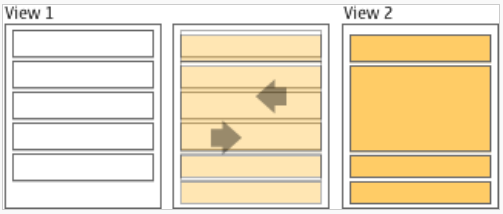
3. Fade in/out

Problem summary Abrupt changes in view are uncomfortable and often disorienting to the user.

Use

- When transitioning between widget views.
- To attract attention to, yet smooth out the loading of new data, images or other media.

Solution Incorporate a transition to gradually fade the visibility of an element into or out of view.



The transition is used to gradually fade out View 1 and fade in View 2

The Y.Anim can again be used for this effect, and instead of working with height we would be working with the opacity.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
                    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title></title>
 <script type="text/javascript" src="http://yui.yahooapis.com/3.0.0/build/yui/yui-
min.js"></script>
   <script type="text/javascript">

  var myY = null;

  YUI().use('anim', doInit);

  function doFade(){

   var listFrom = myY.one('#list').getStyle('opacity');
   var detaFrom = myY.one('#detail').getStyle('opacity');

   myY.animList.set('from',{opacity: listFrom});
   myY.animDetail.set('from',{opacity: detaFrom});

   if(detaFrom < 1){
    myY.animList.set('to',{opacity: 0});
    myY.animDetail.set('to',{opacity: 1});
   }else{
    myY.animList.set('to',{opacity: 1});
    myY.animDetail.set('to',{opacity: 0});
   }

   myY.animList.run();
   myY.animDetail.run();
  }

  function doInit(Y){
```


(C) Copyright Nokia 2014. All rights reserved.

```
    myY = Y;

    myY.animList = new Y.Anim({
            node: '#list',
      from: { opacity: 0 },
      to: { opacity: 1 },
            duration: 2.0,
        });

    myY.animDetail = new myY.Anim({
           node: '#detail',
      from: { opacity: 0 },
      to: { opacity: 1 },
            duration: 2.0,
        });

    myY.get('#btn_fade').on('click', doFade);
   }

    </script>
   <style type="text/css">
   div {
   display: block; }
  div#list {
   height: 100px;
   width: 150px;
   position:absolute;
   top: 50px;
   background:#f00; }
  div#detail {
   height: 100px;
   width: 150px;
   position:absolute;
   top: 50px;
   background:#0f0;
   opacity: 0;}
   </style>
 </head>
 <body>
   <button id="btn_fade">Fade Transition</button>
   <div id="list">
    <h1>List View</h1>
   </div>
   <div id="detail">
    <h1>Detail View</h1>
   </div>
 </body>
 </html>
```

WRT Widget for the example code can be found from File: File:FadeOut YUI.zip

Notes: As WRT currently does not support opacity through CSS, the example code does not really work, you could get it working better by adding WRT's "fade" transition codes into it.

4. Expand/Collapse

(C) Copyright Nokia 2014. All rights reserved.

Problem summary Users require access to content or functionality but they only need it under certain circumstances and/or there is insufficient space to provide all of it at all times.

Use

- To reveal or obscure additional content associated with an element.
- Within a messaging application, to temporarily collapse non-critical information such as sender address, date, persons Ccd. etc.
- When needing to view content (i.e. text, images) that is currently represented in a thumbnail view.
- When making content editable. Ex. Expand a text field to allow for more visible content while editing.

Solution Hide a portion of this information then provide the user with a means to trigger the reveal of this information when required. This action should be easily discoverable, easy to trigger and the reveal should (ideally) occur in context of the original data.

The area is expanded to reveal additional content

This is basically the same effect as the sliding, so the Y.Anim is used again to change the height of the item.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
                    "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title></title>
 <script type="text/javascript" src="http://yui.yahooapis.com/3.0.0/build/yui/yui-
min.js"></script>
   <script type="text/javascript">

  YUI().use('anim', function(Y) {

   var anim1 = new Y.Anim({
           node: '#item',
           to: { height: 40 },
           duration: 2.0,
       });

   var anim2 = new Y.Anim({
          node: '#item',
           to: { height: 200 },
           duration: 2.0,
       });

   Y.get('#btn_collapse').on('click', anim1.run, anim1);
   Y.get('#btn_expand').on('click', anim2.run, anim2);
  });
   </script>
  <style type="text/css">
   p, h1 {
    padding: 0; margin: 0; }
 div#item {
  height: 40px;
  background:#f00;
  overflow: hidden; }
```


(C) Copyright Nokia 2014. All rights reserved.

```
 div#more {
  background:#0f0; }
  </style>
</head>
<body>
  <button id="btn_expand">Expand</button>
  <button id="btn_collapse">Collapse</button>
  <div id="item">
   <h1>Header</h1>
   <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Maecenas lorem. Duis pellentesque. Cras nec tortor vel lacus fermentum facilisis. Nam id
pede eu eros posuere cursus. Mauris dolor ligula, eleifend vitae, aliquam sit amet,
bibendum vel, est. Cras sit amet augue sit amet ligula cursus scelerisque.</p>
  </div>
  <div id="more">
   <h1>More...</h1>
  </div>
</body>
</html>
```

WRT Widget for the example code can be found from File: File:ExpandCollapse YUI.zip

Notes: Another way to create expanding and collapsing elements is to hide and show sub-elements causing the display block of the container block to expand and contract as necessary. This method does not typically incorporate animation as a means of transition.

(C) Copyright Nokia 2014. All rights reserved.