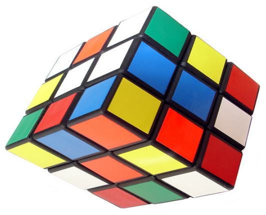
Cut to the Chase Series
More Walk – Less Talk

zenyan webapp framework



Toycode Examples

zenyan webapp framework – Toycode Examples

Table of Contents

zenyan Toycode Examples

Please Readme First

The philosophy behind toycode is simple. Toycode is nothing more than a working piece of code that demonstrates a useful technique, offers syntax, and/or demonstrates something the consumer might deem as useful or desirable.

Toycode can really be represented by any of the following:

- ✓ General language syntax that demonstrates something as simple as how to write a "for loop."
- ✓ Generalized algorithms
- ✓ Langauge or api specific. Like a JQuery technique, or an example on how to use various functions within a given api, module, or plugin.
- ✓ A hardcoded code stub that demonstrates use of a particular UI widget.

The essence of toycode is to offer up a static (**already functioning and tested**) piece of code that reflects a technique or something useful one may need to do within their application. In terms of zenyan toycode will typically pertain to things one needs to do and/or use to develop applications within the zenyan framework.

Well offered and well document toycode examples can actually serve as building blocks for development. They can also be used for testing.

Further Scoop Regarding this Document

It is doubtful that this document will be able to keep up with documenting the toycode examples offered up. But, given that there is a web based master list that allows you to view the toycode examples. I will add to this as new examples are introduced.

The List (thus far)

At the moment the list is small. I hope it to grow measurably over time. I certainly have a ton of stuff to document and offer up. Hopefully others will too.

Where the Toycode Resides

In the “php_code_examples” directory. This directory can include toycode for examples other than php (JQuery, Javascript, etc...). Also, since zenyan 0.96 there are also numerous toycode examples attached to the menus themselves once you login.

Accessing and Viewing Toycode

http://<dns or ip>/<app_root>/php_code_examples/View_Toycode.php

for example

http://localhost/dataminer_proto/php_code_examples/View_Toycode.php

UI Toycode

Name	Tech	Description
toycode_ui_datePicker	JQuery	Control to allow users to select data from visual calendar and populate text fields.
toycode_ui_datePicker_persistence	JQuery	Same as above, but keeps the context in the form fields after form submission.
toycode_ui_treeview_cookie_persistence	JQuery	Simple implementation of a nested treeview with browser-side persistence via a cookie.

Date Selector Control (toycode_ui_datePicker.php)

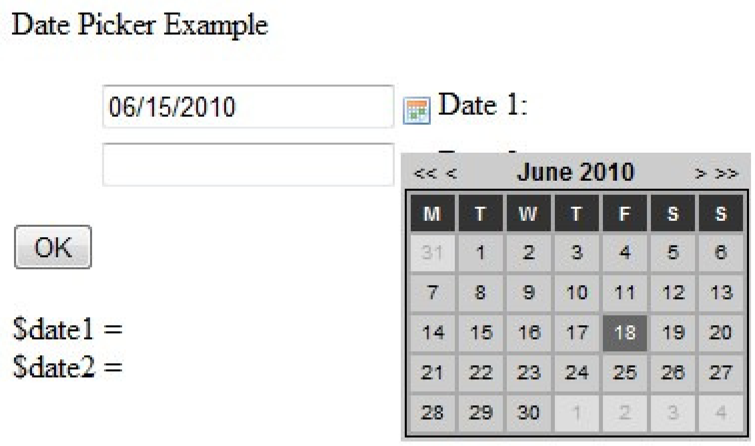
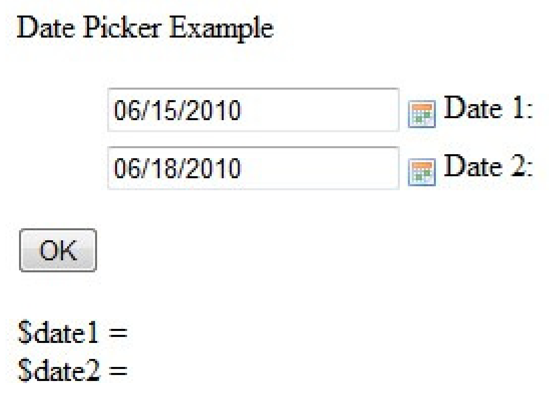


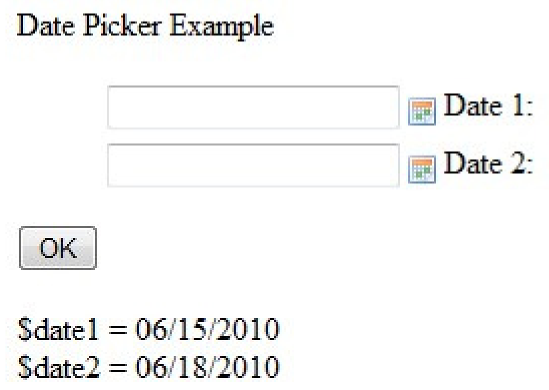
Illustration 1: Selecting a Date



06/15/2010
06/18/2010

$date1 =
$date2 =

Illustration 2: Populated Textboxes with Date from Date Picker



$date1 = 06/15/2010
$date2 = 06/18/2010

Illustration 3: After submit - "look ma, no context"

What is does

This control allows the user to pick a date from a visual calendar. It populate the results into a text field. It is intended to be used for runtime data access where date ranges and row restriction are required from the data access layer. It also is designed to be part of a form to allow end-users to enter date data.

The fields that are populated with the date selected from the picker are not retained upon submission of the form. If you desire to maintain this context you need to implement the “toycode_ui_datePicker_persistence.php” version of this control.

How to Implement it (the code)

This is a JQuery control. All code references exist within the html section of a Php script.

The following are the reference to the general date picker api.

```
        <!-- required plugins -->
         <script type="text/javascript" src="../jquery/date.js"></script>
         <!--[if IE]><script type="text/javascript"
src="../jquery/jquery.bgiframe.min.js"></script><![endif]-->


        <!-- jquery.datePicker.js -->
         <script type="text/javascript"
src="../jquery/jquery.datePicker.js"></script>


        <!-- datePicker required styles -->
         <link rel="stylesheet" type="text/css" media="screen"
href="../jquery/stylesheet/datepicker.css">
```

The code that follows is the external css reference. It includes comment codes you should read that describe the decision to externalize this css. In terms of this css, you may very well determine that you want to create and set a single standard for this css. If that is the case, you may want to consider moving this css file into the “stylesheet” directly that is immediately below the root directory.

```
This stylesheet is purposely externalized from the api in order to give you
max control over formatting of the control. As-is its only use right now is
in referencing the date icons. But, there is much more functionality than this
that can be applied. Refer to the documentation for this control for the
details.
-->
<link rel="stylesheet" type="text/css" media="screen"
href="datepickerexample.css">
```

And, this in-line JQuery script determines how far back to allow the calender to go. You can modify this and the endDate in your specific implementation of this control.

```
        <!-- page specific scripts -->
         <script type="text/javascript" charset="utf-8">
            Date.format = 'mm/dd/yyyy';
            $(function()
            {
                          $('.date-pick').datePicker({startDate:'01/01/2000'});
            });
         </script>
```

And, following is the html markup that implements the example you are looking at in the UI.

```
<form name="chooseDateForm" id="chooseDateForm" method="get"
action="toycode_ui_datePicker.php">
<legend>Date Picker Example</legend>
  <ol>
    <li>
      <label for="date1">Date 1:</label>
      <input name="date1" id="date1" class="date-pick" />
    </li>
    <li>
      <label for="date2">Date 2:</label>
   <input name="date2" id="date2" class="date-pick" />
    </li>
   </ol>
    <input type="submit" name="submit" value="OK" />
</form>
```

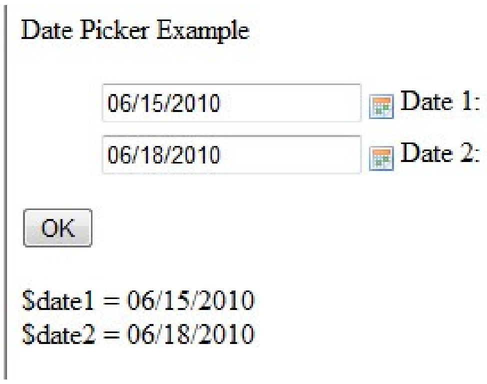
Date Selector Control with Persistence (toycode_ui_datePicker_persistance.php)

Toycode Ancestry

This example is derived from toycode_ui_datePicker.php.

What is does

It does the same thing as toycode_ui_datePicker.php with one notable exception. After form submission context is maintained in the text field(s).



```
$date1 = 06/15/2010
$date2 = 06/18/2010
```

Illustration 4: "look ma, persistence"

How to Implement it (the code)

Only the code unique to this version is presented here. See toycode_ui_datePicker.php for the rest (majority) of the details involving implementing this control within your application.

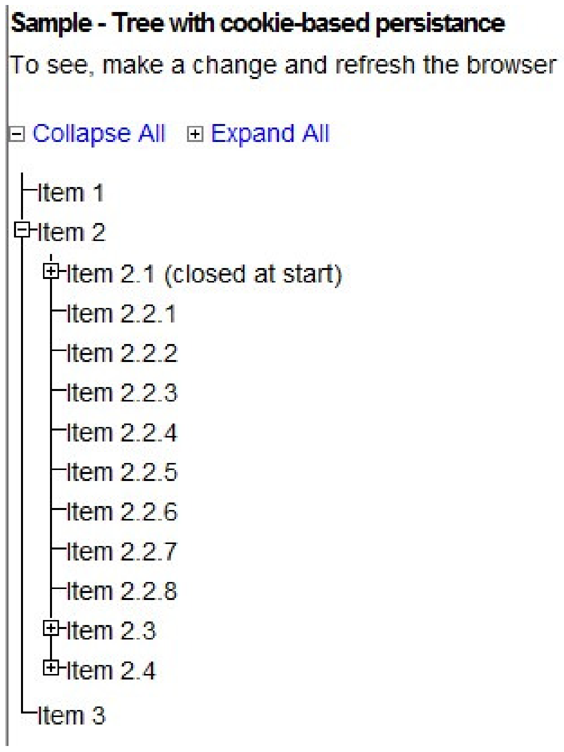
As you can see in the code fragment below, context is maintained by writing the submitted value back into the form.

```
<input name="date1" id="date1" class="date-pick"
<?php echo 'value="' . $_REQUEST['date1'] . '"'; ?> />


<input name="date2" id="date2" class="date-pick"
<?php echo 'value="' . $_REQUEST['date2'] . '"'; ?> />
```

Treeview Selector (toycode_ui_treeview_cookie_persistence.php)

Please note that for more options pertaining to this control look at “treeview.html.”



Sample - Tree with cookie-based persistance

To see, make a change and refresh the browser

Illus tration 5: Treeview example

What is does

This is the JQuery Treeviewer control. It is a very flexible control (both literally and figuratively). In zenyan we have taken this control and created a document management subsystem. You can also use it to...

- ◆ Add hyperlinks (anchor tags) into the nodes.
- ◆ As a means of selecting items for forms.
- ◆ As a XML viewer.
- ◆ As a UI viewer for hierarchical data.
- ◆ As a control for application navigation.

It is really an awesome control that is functional and visually appealing and has a myriad of possibilities pertaining to morphing it into higher order application controls.

I am demonstrating this control using browser-side cookie persistence as I feel this is the desired default behavior for a user interface. I will also demonstrate how to remove persistence.

How to Implement it (the code)

References in <head>

```
<link rel="stylesheet" href="../jquery/stylesheet/jquery.treeview.css" />
<script src="../jquery/jquery.cookie.js" type="text/javascript"></script>
<script src="../jquery/jquery.treeview.js" type="text/javascript"></script>
```

Partial sample code from <body>

Code to implement global “collapse” and “expand” functionality

```
<div id="treecontrol">
 <a title="Collapse the entire tree below" href="">
  <img src="../jquery/stylesheet/images/minus.gif" />&nbspCollapse
All&nbsp&nbsp</a>
 <a title="Expand the entire tree below" href="">
  <img src="../jquery/stylesheet/images/plus.gif" />&nbspExpand All</a>
</div>
```

Implementing a treeview

```
<ul id="black" class="treeview-black">
<li>Item 1</li>
<li>
 <span>Item 2</span>
 <ul>
   <li class="closed">
         <span>Item 2.1 (closed at start)</span>
         <ul>
               <li>Item 2.1.1</li>
               <li>Item 2.1.2</li>
         </ul>
   </li>
   <li>Item 2.2.1</li>
   <li>Item 2.2.2</li>
   <li>Item 2.2.3</li>
   <li>
         <span>Item 2.3</span>
         <ul>
```

```
               <li>Item 2.3.1</li>
               <li>Item 2.3.2</li>
               <li>Item 2.3.3</li>
         </ul>
   </li>
 </ul>
</li>
</ul>
```

The main UL creates the treeview

```
<ul id="black" class="treeview-black">
  ...
</ul>
```

Within the treeviewer the spans create the individual nodes.

```
 <span>Item 2</span>
```

Within a node you just use standard unordered list markup to create the leafs to meet your specific hierarchical needs.

The following is a code fragment demonstrating how to set the initial default to closed or open for a node.

```
<li class="closed">
```

or

```
<li class="open">
```

If you have notice in perusing the code that there is no specific reference to the cookie. We now need to take a look at the reference specific to our example (and the “treeview.html” example). This is the external js file “treeview.js.”

```
   $("#black, #gray").treeview({
         control: "#treecontrol",
         persist: "cookie",
         cookieId: "treeview-black"
   });
```

If you want persistence for your treeviewer it is a matter of adding the tags.

```
         persist: "cookie",
         cookieId: "treeview-black"
```

You can think of this js file as the master template for the various treeviewer control designs you want to implement. You can analyze the code here in conjunction with the example to learn more about the settings and how they affect behavior of the control. You can also get more information on this control from the JQuery website.

Code Examples Under The Main Menu

Once you login you will find a number of code examples under “UI Candy”, and “Robust Prototypes”. Many of these extend beyond simple toycode, but will give you an example of what you can do in the UI.

zenyan comes with a number of 3rd party plugins that have been integrated into the framework. Plugins like:

- CKEditor
- PhpExcel
- pChart

You can find documentation for these plugins under the eDMSroot directory.

Creating Your Toycode the zenyan Way

What is the zenyan way?

Guidelines

- ✓ The toycode needs to be working code that someone can execute and view. Why would anyone want to use your stuff if it does not work to begin with?
- ✓ Toycode should be well documented within the code to the degree you think people will need. I must confess I am not always real good at doing this.
- ✓ Think quality instead of quantity when adding to the toycode library.
- ✓ Try not to demonstrate too many different concepts within your code. This is a common mistake one makes. The goal is for someone to find a technique and go, "aha, so that is how you do that... or... I have use for that in my app."
- ✓ Toycode can be for any of the technologies used within the zenyan framework.
- ✓ A template should be added to the "View_Toycode.php" page in an applicable category so that the user can see what is available. How do you expect someone to reuse stuff and follow your design patterns if you make it difficult to determine what the heck they are?
- ✓ Give some thought to naming your file and label before adding it to the "View_Toycode.php" page.
- ✓ Put your toycode under the zenyan directory "code_examples" so as to provide a one-stop shopping location for these zenyan artifacts.
- ✓ If the toycode library gets too big to manage the files for a single directory start creating subdirectories under the code_examples root directory.