## Building MiPages – Basic Concepts

A MiPage is MiServer web page. It's a Dyalog APL class that is based on the MiPage class.
A MiPage has to follow a few simple rules:

1) It has to ultimately be based on the MiPage class
2) It has to have a public method called Compose
3) If the page uses callbacks, the callback function must be named either APLJax or the name specified in the callback definition. A page can have multiple callback functions, one of which could be APLJax.

The following example is a valid MiPage.

```
:Class mymipage : MiPageTemplate     ⍝ template based on the MiPage class

    ∇ Compose;h
      :Access public
      h←Add _.h3'Welcome to MiServer 3.0!'
      h.On'mouseover' 'Callback'
    ∇

    ∇ r←Callback
      :Access public
      r←Execute _JSS.Alert'Hi'
    ∇

:EndClass
```

## Widgets, Controls, and Elements

MiServer implements a number of classes to provide the APL user an easy way to add content to a MiPage.  Currently there are APIs defined for all HTML5 elements, many Syncfusion and jQuery widgets, and a number of Dyalog-developed controls that were developed with the APLer in mind.

All of the widgets, controls, and elements can be accessed from the _ (underscore) namespace.  We found that using _, coupled with the convenience of Dyalog APL's autocomplete to be a great way to enter control names.  There are a few naming conventions to help you know which type of control you're looking at:

| If the control name... | Then it's a... |
|---|---|
| is all lowercase letter (e.g. div, pre, table) | Base HTML5 element<br>Why? HTML5 recommends all element names be lower case |
| begins with an uppercase letter (e.g. RadioButtonGroup, Table) | Dyalog developed control |
| begins with ej | Syncfusion widget<br>Why ej?  Because that's what Syncfusion uses – for enterprise javascript |
| begins with jq | jQuery widget<br>Why jq?  We chose the prefix  to avoid name conflicts with other widgets. |

## Writing MiPage Content

The basic way to add content to a MiPage is to use the Add method.

```
      Add 'Hello'
```

Adds the string "Hello" to your page.

```
      Add _.pre 'Some text'
```

Adds a <pre> (HTML5 preformatted) element with the content "Some text" to your page.

You can add just about any content to your page and MiServer will render it correctly.

The left argument to Add allows you to specify the id, class, and other attributes easily in a single statement using the following rules:

- If the left argument is a simple string, treat it as the id for the element, unless it begins with a . (period), in which case treat it as a class. If the first character is a #, it is forced to be treated as an id.
- If the left argument consists of two simple strings (neither of which contain the equals sign =)
- Elements that are obviously paired, either by an equals sign or nesting are treated as attribute name/value pairs.
- Everything else is treated as an attribute.

```
      ('abc' z.Add _.div 'content').Render
<div id="abc">content</div>

      ('abc' 'def' z.Add _.div 'content').Render
<div abc="def">content</div>

      ('.abc' 'def' z.Add _.div 'content').Render
<div class="abc" def="def">content</div>

      ('#abc' 'def' z.Add _.div 'content').Render
<div id="abc" def="def">content</div>

      ('id1' ('abc' 'def') z.Add _.div 'content').Render
<div id="id1" abc="def">content</div>

      ('id1' ('abc=def ghi=jkl') z.Add _.div 'content').Render
<div id="id1" abc="def" ghi="jkl">content</div>
```