Creating iPhone/iPod/iPad web applications with Delphi & TMS controls

Introduction

Apple's proposition to create applications for its iOS devices to learn Objective C + xCode + InterfaceBuilder + the iOS/Cocoa APIs can be quite daunting for an experienced Delphi developer. Imagine building iPhone, iPod, iPad applications with Delphi though: reusing Delphi skills, benefiting from Delphi's rich offerings for database connectivity, doing a server side deployment only and on top of that, circumvent Apple's strict rules for publishing applications on its AppStore and avoid paying the yearly 99USD Apple tax.
For most Delphi developers this sounds like music to their ears and with Delphi and the IntraWeb framework, you can already go a very long way to create web applications accessible from the mobile device's browser. The missing pieces are offering fast and responsive applications with a user interface that matches the iOS look & feel. This is where the TMS IntraWeb iPhone Controls pack comes in. This is a set of components that is fine-tuned to offer a look and feel very close and in many cases nearly pixel-level identical to native iOS controls.
In addition, the components are equipped with asynchronous & client-side Javascript event handlers for UI interaction to perform page updates without needing a full page refresh. The magic is performed by a combination of Webkit optimized CSS3/HTML5 and the fine-grained asynchronous capabilities of the VCL for the Web (IntraWeb) framework. As the Android based mobile devices also use the Webkit browser, this means the web applications can also be used on the wide range of Android mobile phones and tablets.

Getting started

To create this type of mobile device web applications, a Delphi version with IntraWeb 10 or IntraWeb 11 framework installed is required. Start a new VCL for the Web application from the Delphi repository and on the main form, add the control TIWIphoneStyle. Thanks to this non visual control, the necessary CSS is generated to allow to create a web application for which a desktop icon can be created and a web app that runs without browser URL bar when started from the desktop icon.

Component overview

Labels:
TTIWiPhoneEmailLabel, TTIWiPhoneLocationLabel, TTIWiPhonePhoneLabel & TTIWiPhoneSMSLabel

These label components integrate with the mobile device. When clicking the label, this can start the phone, SMS, email or Google maps application with the information preset in the control.

Buttons:
TIWiPhoneButton, TTIWiPhoneOnOffButton, TTIWiPhoneTrackbar

This standard button control has the appearance of the iOS button. It offers sync, async and client Javascript events. The TTIWiPhoneOnOffButton is the toggle button with smooth transition animation. The TTIWiPhoneTrackbar is a trackbar with async events and a look and feel that matches the native iOS trackbar control.

Page design:

TTIWiPhoneHeader,TTIWiPhoneFooter,TIWiPhoneMenu,TTIWiPhoneRegion,
TTIWiPhoneScrollRegion, TTIWiPhonePageFlip

The components offer a footer or header with possibility to add text and buttons/images with click actions or the typical iPhone/iPad menu at the bottom of the screen that can host menu items with optional red bubble status indicators. The TTIWiPhonePageFlip allows to provide smooth animated transitions between two regions like slide-in, rotate,... The TTIWiPhoneRegion & TTIWiPhoneScrollRegion are designed to allow to layout the controls at design-time on the exact screen size of the iPhone or iPad and optionally make the region scrollable at runtime using the typical iOS scroll effect.

List:
TTIWiPhoneList

The TTIWiPhoneList is a sophisticated list control that can be configured in a list mode similar to the iPhone contacts list or iPad email list as well as a mode similar to the iPhone settings list with grouped sections of items. The TTIWiPhoneList has built-in scrolling behavior, asynchronous events and loading of items, capability to show a detail form and features to show images, caption, notes, chevron, status per item.

System:
TIWiPhoneGEOLocation

This component enables to fetch the longitude & latitude of the device and send it asynchronously back to the server. TTIWiPhoneStyle holds several application settings. It renders the necessary CSS to ensure the application fonts match the device font and can be configured to let your web application run full screen and specify an icon to use when the application is put on the iPhone or iPad desktop.

Deeper look at the TIWIPhoneMenu

The TIWiPhoneMenu represents the buttonbar typically found in many native iPhone/iPad applications. It consists of a collection of menu items. Each menu item has a caption, an image, an optional indicator (red bulb indicator) and also optionally an image for the selected state. Using TIWiPhoneMenu is simple. Drop it on the form. By default it is bottom aligned (as in most native iPhone apps) and insert menu items via TIWiPhoneMenu.Items.

Code:

```
TIWiPhoneMenu.Items.Clear;
TIWiPhoneMenu.Items.Add;
TIWiPhoneMenu.Items[0].Image := 'Files/MenuOption1.png';
TIWiPhoneMenu.Items.Add;
TIWiPhoneMenu.Items[1].Image := 'Files/MenuOption2.png';
```

The TIWiPhoneMenu has built-in support for asynchronous updates. For example, from an async event triggered by another control on the form, we can set the indicator. Assuming there is a "Refresh" button in the TIWiPhoneHeader left button, we could update the indicator to reflect the latest nr. of new data available with code:

```
procedure TIWForm1.TIWIPhoneHeader1AsyncLeftButtonClick(Sender: TObject;
  EventParams: TStringList);
begin
  // set the red bulb indicator that 2 new unread information is available
  TIWIPhoneMenu1.Items[0].IndicatorCaption := '2';
  TIWIPhoneMenu1.Items[0].IndicatorVisible := true;
end;
```

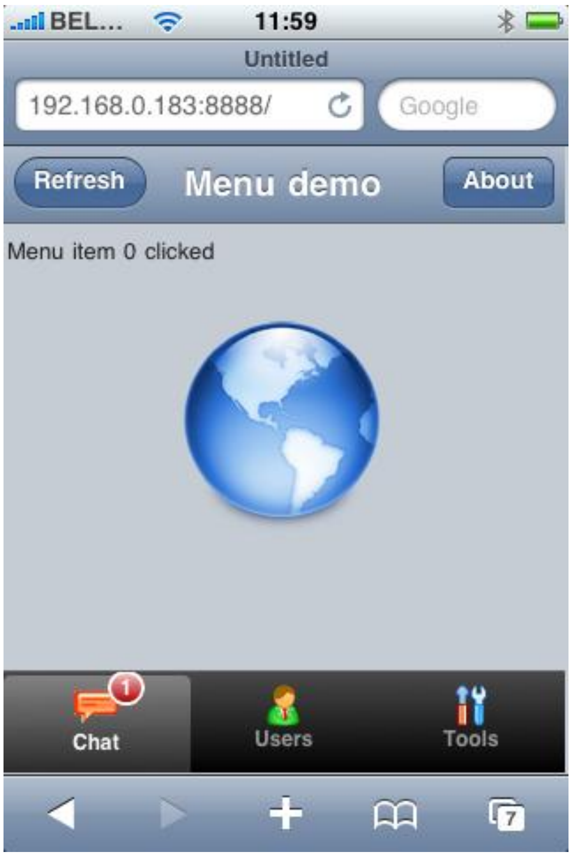
A click on the TIWiPhoneMenu menuitem can also be asynchronously handled, in this case to update an IWLabel:

```
procedure TIWForm1.TIWIPhoneMenu1AsyncItemClick(Sender: TObject;
  EventParams: TStringList; ItemIndex: Integer);
begin
  iwlabel1.Caption := 'Menu item '+ inttostr(ItemIndex)+' clicked';
end;
```

To further improve the efficiency & performance, Javascript client-side event handling code can be added via the property TIWiPhoneMenu.ScriptEvents. The Javascript code added in ItemClick is executed first and can be used to perform client-side validation whether to post back to the server or not. This sample code snippet demonstrates this:

```
if (index == 0)
{
  var answer = confirm("Are you sure to update?");
  if (!answer) return;
}
```

When the user clicks "Cancel" on the prompt shown after clicking on the first menu item, the postback is cancelled.



Deeper look at the TIWIPhoneList

TIWIPhoneList is a versatile list control with many built-in features. It can be configured as a regular list as well as a settings lists with sections. To show the powerful capabilities of TIWIPhoneList, we'll create here a list with asynchronous loading of items as well as asynchronous slide in/out. The TIWIPhoneList consists of a collection of items accessible via TIWIPhoneList.Items. Each item in the collection has following properties:

- Caption: Main text of the item.
- Data: Extra none visual data associated with the item.
- Detail: When true, a chevron is displayed on the right side.
- Image: Sets the filename of an image displayed in the list on the left side.
- Name: Name identifier of the item.
- Notes: Additional description text that is optionally displayed under the caption.
- Section: When true, the item is a section header. This applies when the TIWIPhoneList.ListType is ltSettings.
- Tag: Extra non visual integer property for the item.
- Value: Text that can be optionally displayed on the right side of the item.

Items are added to the list via:

```
TIWIPhoneList.Items.Add;
TIWIPhoneList.Items[TIWIPhoneList.Items.Count - 1].Caption := 'ListItem';
TIWIPhoneList.Items[TIWIPhoneList.Items.Count - 1].Detail := true;
```

and by default, all items in the collection are automatically sent to the browser and displayed in the list. In case a large list of data is available, it can be beneficial to send only the visible items to the browser and let the user load the remaining items asynchronously. This is technique also often used in applications that present Twitter or Facebook streams and only load additional information when needed. To configure the TIWIPhoneList for this behavior, set TIWIPhoneList.InitialItemCount to a value different from zero. When this is the case, the bottom item can be clicked to load additional items. The text that is displayed in this bottom item is set via TIWIPhoneList.AsyncLoadCaption. Set TIWIPhoneList.AsyncLoadCount to the number of items that should be loaded when this bottom item is clicked and the TIWIPhoneList will automatically handle this.

The initialization of the TIWIPhoneList becomes:

```
procedure TIWForm4.IWAppFormCreate(Sender: TObject);

  procedure AddItem(city, people, country: string);
  begin
    TIWIPhoneList1.Items.Add;
    TIWIPhoneList1.Items[TIWIPhoneList1.Items.Count - 1].Caption := City;
    TIWIPhoneList1.Items[TIWIPhoneList1.Items.Count - 1].Detail := true;
    TIWIPhoneList1.Items[TIWIPhoneList1.Items.Count - 1].Data.Add(people);
    TIWIPhoneList1.Items[TIWIPhoneList1.Items.Count - 1].Data.Add(country);
    TIWIPhoneList1.Items[TIWIPhoneList1.Items.Count - 1].Data.Add('');
  end;


begin
  AddItem('Brussels','1.000.000','Belgium');
  AddItem('Paris','5.000.000','France');
  AddItem('Amsterdam','2.000.000','Netherlands');
  AddItem('London','6.000.000','United Kingdom');
  AddItem('Berlin','2.000.000','Germany');
  AddItem('Cologne','1.500.000','Germany');
  AddItem('Los Angeles','4.000.000','USA');
  AddItem('New York','10.000.000','USA');
  AddItem('Las Vegas','500.000','USA');
  AddItem('Madrid','1.200.000','Spain');
  AddItem('Oslo','750.000','Norway');
  AddItem('Dublin','900.000','Ireland');
  AddItem('Rio De Janeiro','4.000.000','Brazil');

  TIWIPhoneList1.InitialItemCount := 5;
  TIWIPhoneList1.AsyncLoadCount := 3;
end;
```

When an item is clicked on the TIWIPhoneList, the async click event is used to instruct the TIWIPhonePageFlip to slide in a detail page and asynchronously initialize the information on the detail page with the data from the item clicked. This is done with the code:

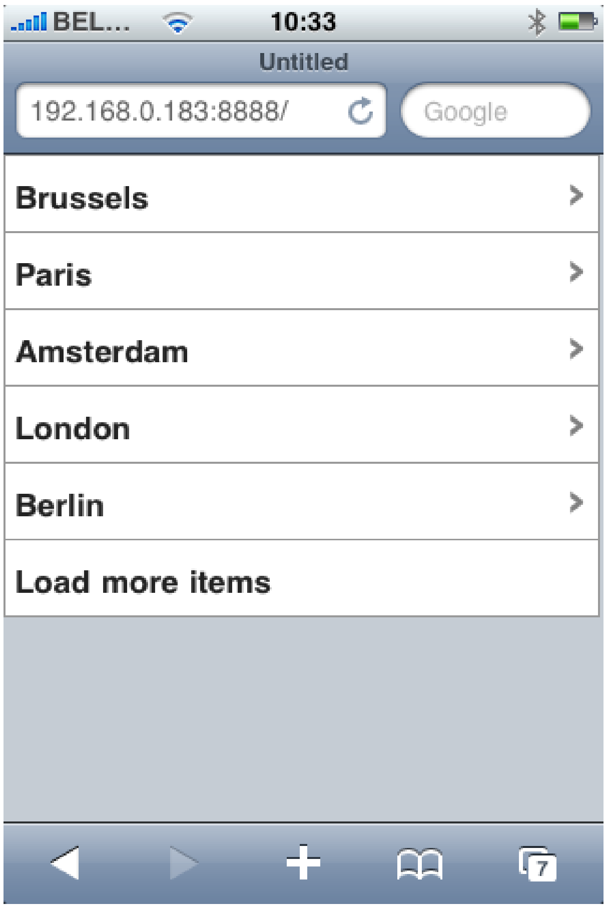
```
procedure TIWForm4.TIWIPhoneList1AsyncItemClick(Sender: TObject;
  EventParams: TStringList; ItemIndex: Integer);
begin
  TIWIPhonePageFlip1.SlideToBack;
  IWLabel1.Caption := TIWIPhoneList1.Items[ItemIndex].Caption;
  IWEdit1.Text := TIWIPhoneList1.Items[ItemIndex].Data[0];
  IWEdit2.Text := TIWIPhoneList1.Items[ItemIndex].Data[1];
  IWEdit3.Text := TIWIPhoneList1.Items[ItemIndex].Data[2];
  SelectedItem := ItemIndex;
end;
```

Note that in this code, the label and edit controls on the detail page are asynchronously initialized with data from the items in the TIWIPhoneList.

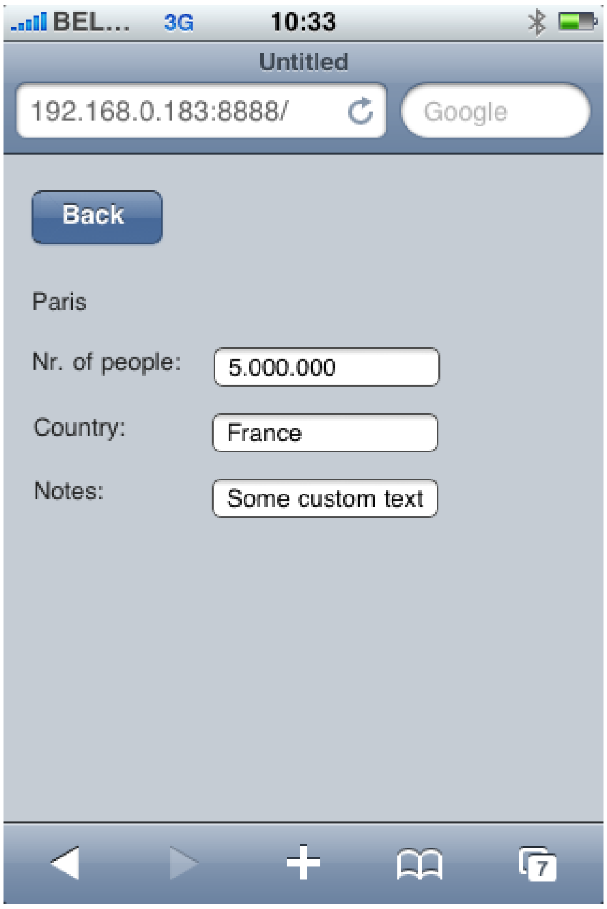
The same technique is applied when asynchronously sliding out the detail screen again to the list where the edited values are asynchronously stored in the list:

```
procedure TIWForm4.TIWIPhoneButton1AsyncButtonClick(Sender: TObject;
  EventParams: TStringList);
begin
  TIWIPhoneList1.Items[SelectedItem].Data[0] := IWEdit1.Text;
  TIWIPhoneList1.Items[SelectedItem].Data[1] := IWEdit2.Text;
  TIWIPhoneList1.Items[SelectedItem].Data[2] := IWEdit3.Text;

  TIWIPhonePageFlip1.SlideToFront;
end;
```

List control with async. load of extra items



Detail screen when list item is clicked



Paris

Nr. of people: 5.000.000

Country: France

Notes: Some custom text

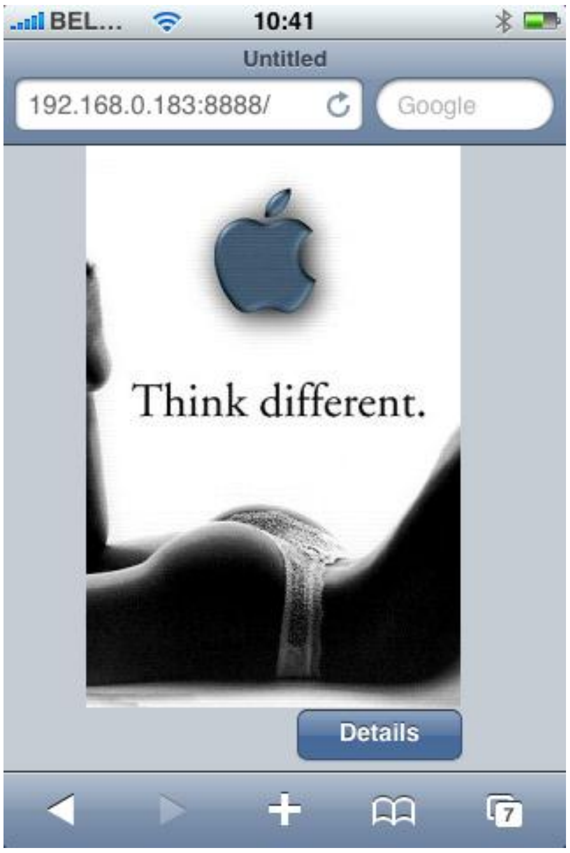
Deeper look at the TIWIPhonePageFlip

With the TIWIPhonePageFlip component, it is possible to integrate smooth animated transitions between pages in an iPhone web application. This uses HTML5/CSS3 that is supported by webkit browsers in iPhone, iPod, iPad. To add such animated transitions, drop a TIWIPhonePageFlip component on the form as well as two TIWiPhoneRegion components.

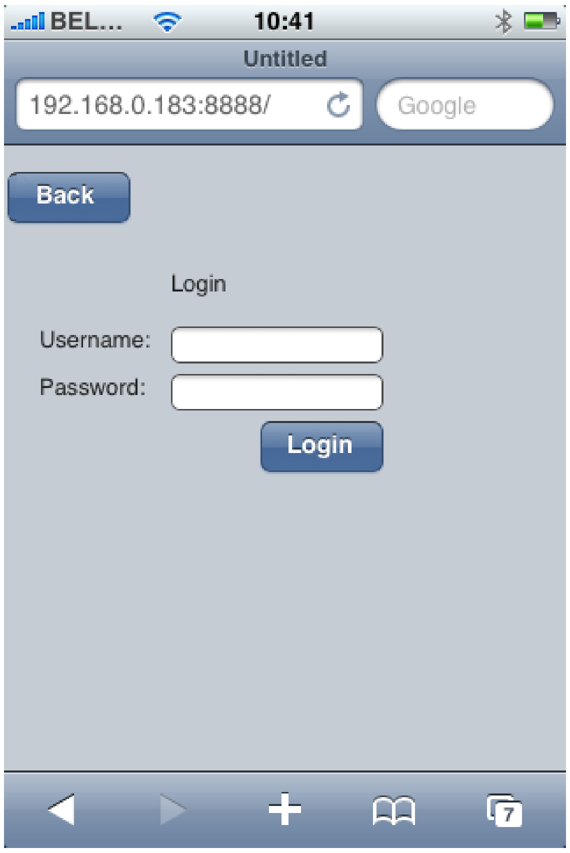
Assign one of the regions to TIWIPhonePageFlip.FrontRegion and the other region to TIWIPhonePageFlip.BackRegion. Drop any component on the two regions. To start the animated transition from front to back region or vice versa, it is sufficient to asynchronously call TIWIPhonePageFlip.SlideToBack or TIWIPhonePageFlip.SlideToFront. The animation type and animation speed is configured with: TIWIPhonePageFlip.AnimationType and TIWIPhonePageFlip.AnimationSpeed. The current supported animation types are:

- atFlip: horizontal flip rotation between front & back region
- atSlide: slide from left to right
- atSwing: vertical swing from front to back
- atTurn: animation like opening/closing door
- atDrop: vertical slide in from top

Front region



Asynchronously flipping to back region



Deeper look at the TIWIPhoneSpinner

The TIWIPhoneSpinner is a web implementation of the native iOS date/time selector wheel control. TIWIPhoneSpinner offers a configurable number of slots, set with the TIWIPhoneSpinner.Slots collection. Each slot can have a configurable number of items, set via a stringlist TIWIPhoneSpinner.Slots[index].Items. Initializing the TIWIPhoneSpinner control is as such simple. This code snippet initializes the spinner control with 2 slots and the first slot contains a number of cities, the second slot contains an hour:

```
procedure TIWForm4.IWAppFormCreate(Sender: TObject);
var
  i: integer;
begin
```

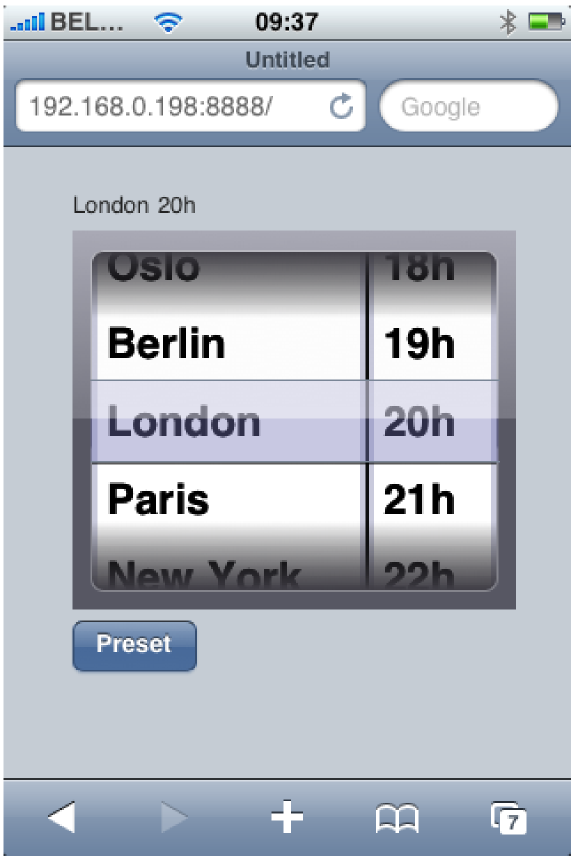
```
  TIWIPhoneSpinner1.Slots.Clear;
  TIWIPhoneSpinner1.Slots.Add;
  TIWIPhoneSpinner1.Slots[0].Items.Add('Brussels');
  TIWIPhoneSpinner1.Slots[0].Items.Add('Madrid');
  TIWIPhoneSpinner1.Slots[0].Items.Add('Oslo');
  TIWIPhoneSpinner1.Slots[0].Items.Add('Berlin');
  TIWIPhoneSpinner1.Slots[0].Items.Add('London');
  TIWIPhoneSpinner1.Slots[0].Items.Add('Paris');
  TIWIPhoneSpinner1.Slots[0].Items.Add('New York');
  TIWIPhoneSpinner1.Slots[0].Items.Add('Amsterdam');

  TIWIPhoneSpinner1.Slots.Add;
  for i := 0 to 23 do
  TIWIPhoneSpinner1.Slots[1].Items.Add(inttostr(i)+'h');
end;
```

When the user spins one of the slot wheels to change the value, the event OnAsyncScrolled is triggered. It returns the index of the slot that was spinned, together with the new SelectedIndex in the slot. It's easy to asynchronously update for example a TIWLabel with the new selected value:

```
procedure TIWForm4.TIWIPhoneSpinner1AsyncScrolled(Sender: TObject;
  EventParams: TStringList; SlotIndex, SelectedIndex: Integer);
var
  selidx: integer;
  city,hr: string;
begin
  city := '';
  hr := '';

  selidx := TIWIPhoneSpinner1.Slots[0].SelectedIndex;

  if selidx <> -1 then
    city := TIWIPhoneSpinner1.Slots[0].Items[selidx];

  selidx := TIWIPhoneSpinner1.Slots[1].SelectedIndex;

  if selidx <> -1 then
    hr := TIWIPhoneSpinner1.Slots[1].Items[selidx];

  IWLabel1.Caption := city + ' ' + hr;
end;
```

It's equally easy to asynchronously set the index of a slot. In the OnAsyncButtonClick event of a TIWiPhoneButton, following code will asynchronously update the selection in the two slots configured in the TIWIPhoneSpinner:

```
procedure TIWForm4.TIWIPhoneButton1AsyncButtonClick(Sender: TObject;
  EventParams: TStringList);
begin
  TIWIPhoneSpinner1.Slots[0].SelectedIndex:=
TIWIPhoneSpinner1.Slots[0].Items.IndexOf('Paris');
  TIWIPhoneSpinner1.Slots[1].SelectedIndex := 8;
end;
```



Other than a general purpose single or multi slot spin control, the TIWIPhoneSpinner can also be configured to be used as a date or time selector. This can be done with the TIWIPhoneSpinner.Mode property that offers following presets: smDateDMY, smDateMDY, smDateYMD, smTimeHM, smTimeHMS. Without using any code, the spinner is this way immediately set up as a day,month, year or month day year or hour,minute or hour,minute,seconds spinner.

Conclusion

We hope this brief overview of the TMS IntraWeb iPhone Controls Pack showed how powerful and easy to use the set of controls is to create immersive iPhone or iPad web applications that give users a near native look and feel and that are throughout optimized for performant asynchronous screen updates. With TMS IntraWeb iPhone Controls Pack you can start building iPhone,iPad applications leveraging your Delphi skills, database access technologies and avoiding any restrictions imposed by Apple's AppStore regulations.