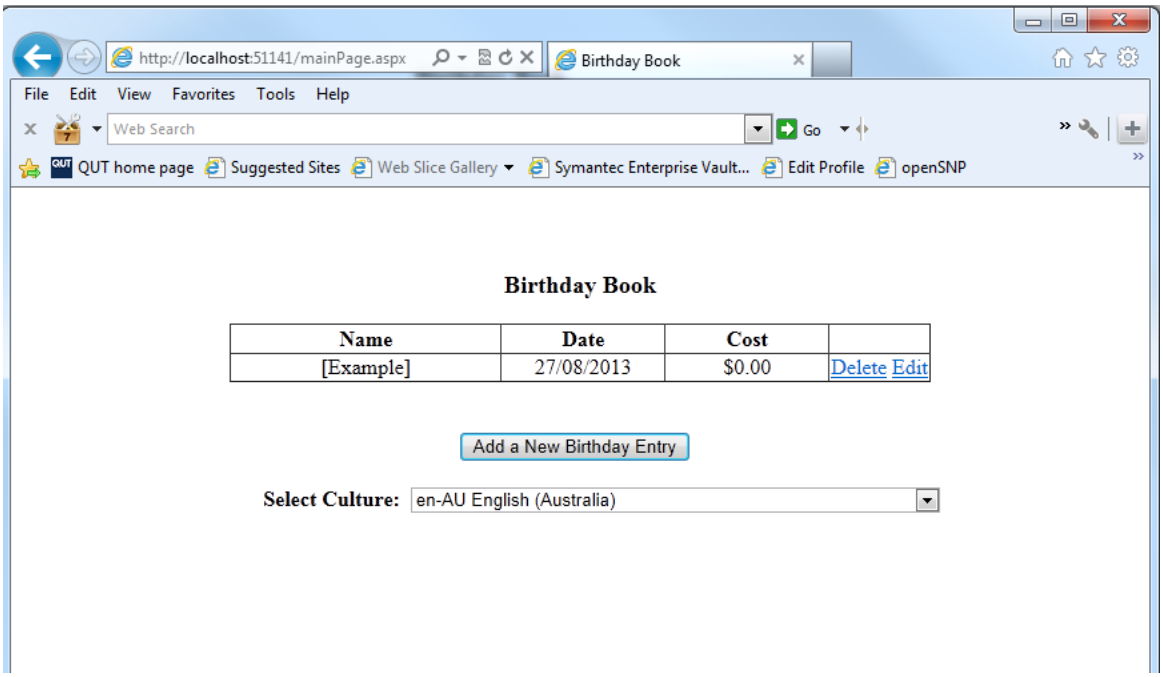
INN570 2013 Week 10 Prac Exercises

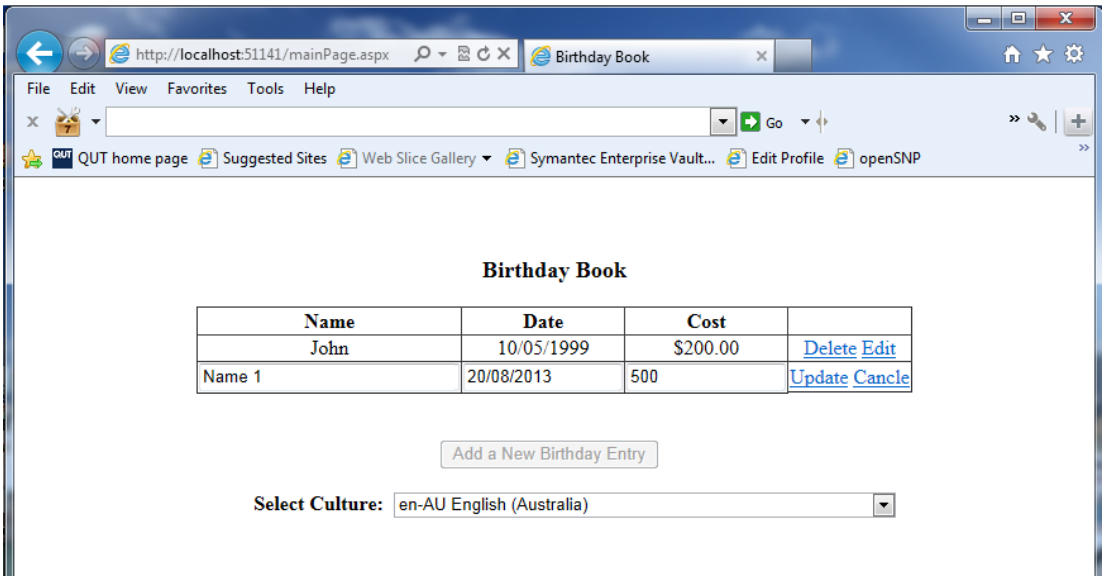
Web localization in .NET

This week we will still work on the Birthday Book application, but here it is a Web based application rather than a Windows form application. The provided code can create a Web page which contains a *GridView* table representing the birthday book, a *DropDownList* displaying a culture list, two *Labels* displaying the labels "Birthday Book" and "Select Culture", and a *Button* allowing users to choose a culture from the culture list. When you run the provided code, you will get a page like the following screenshot:



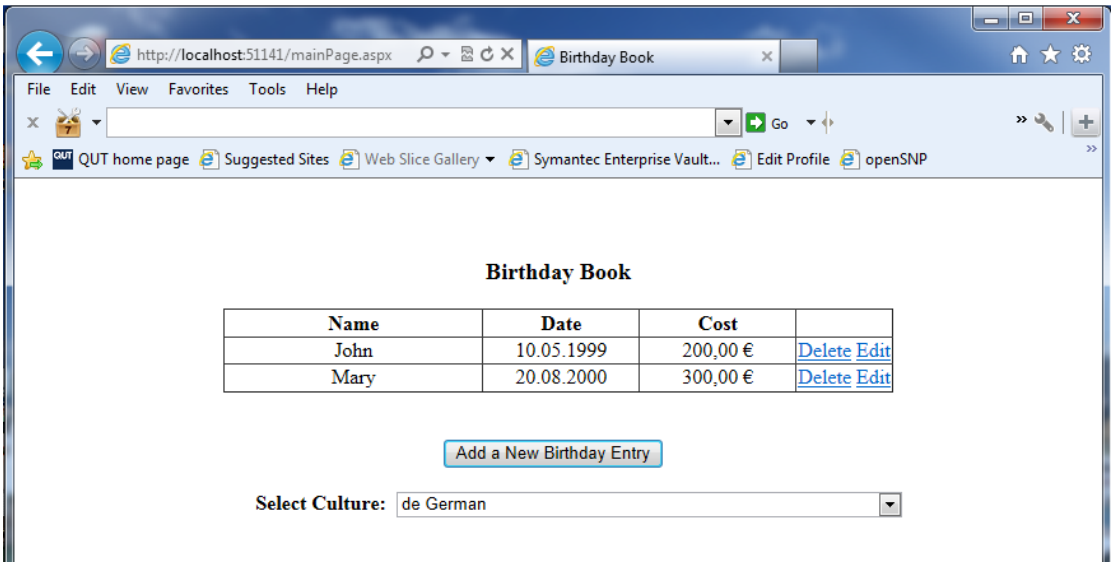
This is a simplified Birthday Book table which doesn't allow us to upload files. Instead, we can add data using the "Add a New Birthday Entry" component.

The birthday book consists of four columns, i.e., a person name `Name`, birthday date `Date`, budget `Cost`, and a column consisting of two buttons, `Delete` and `Edit` which allow the user to delete or modify a row. In the initial table, there is an example record with default values. You can use the button "Add a New Birthday Entry" to add more records. When you click on the button "Add a New Birthday Entry", a new row will be added into the table with the current date and default values something like the following screenshot:



You can change the values and then click on "Update" to add the new row, or click on "Cancel" to add the new row with the default values.

You can choose a locale from the dropdown list to localize the page. For example, in the following screenshot, the locale is `de`.

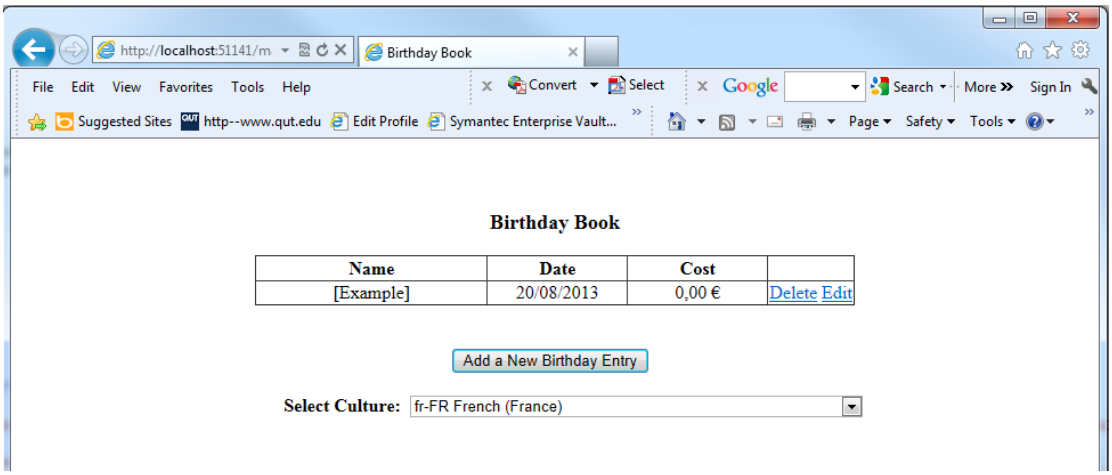


You may have noticed that in the screenshot above, only the current culture (i.e., date format and the currency symbol) has been changed to locale `de`. However, the UI strings are still in English, e.g., the two labels "Birthday Book" and "Select Culture". In this practical, we will first have a look at how to set cultures in a web application, then we will learn how to localize UI strings by using resource files.

1. Culture setting

In the provided code, the culture was not specified. When you run the application, the initial culture will be the current culture in your machine set using Control Panel.

Assume the current culture in your machine is `fr-FR`, when you run the application, the page will be in culture `fr-FR`, as shown below:



Now we want to localize the application to other cultures.

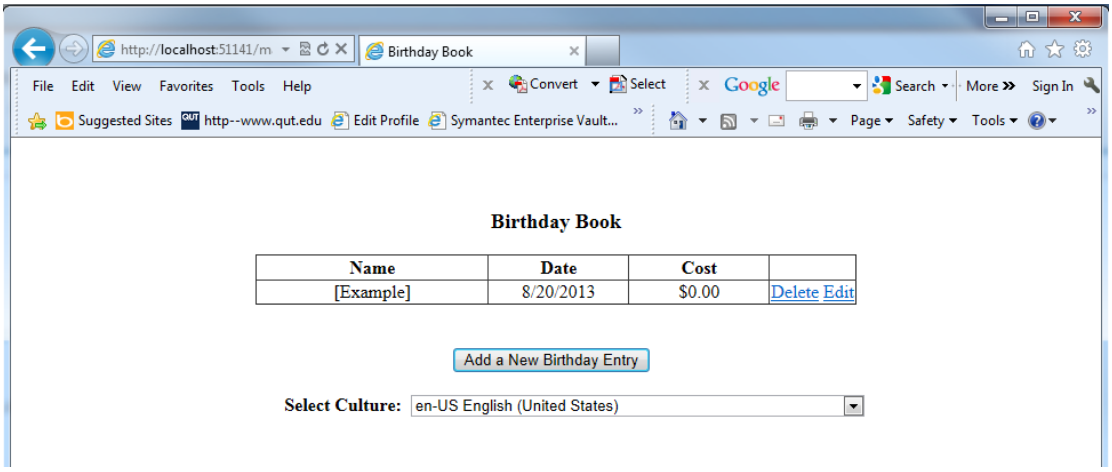
- **Set culture and UI culture in `Web.config`**

We can specify the initial culture in the configuration file `Web.config` for all the pages in the web site (however, for this application, at the moment, there is only one page).

Open the file `Web.config` and add the culture specification, `culture="en-US" uiCulture="en-US",` into the <globalization> element as shown below:

```
<globalization requestEncoding="utf-8"
               responseEncoding="utf-8"
                culture="en-US"
               uiCulture="en-US"/>
```

Then run the application, we will have the following page which is in Culture `en-US`.



- **Set culture and UI culture in `@Page` directive**

We can specify the initial culture of the page in the `@Page` directive, which can be located at the top of the file `mainPage.aspx`.

- If we want to allow the initial culture of the page to be the culture specified in clients' browser, we can set the culture and UI culture to be `"auto"` in the `@Page` directive:

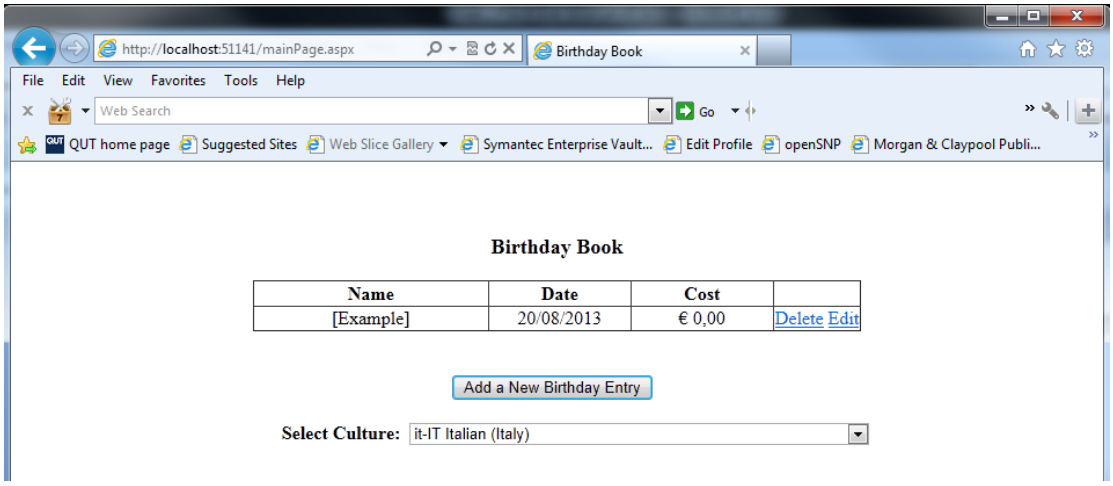
```
<%@Page Language="C#" AutoEventWireup="true" culture="auto"
uiculture="auto" CodeBehind="mainPage.aspx.cs"   Inherits
="BirthdayBookWeb.mainPage"%>
```

Now let's set the browser's language preference. Open Internet Explorer, go to

Tools->Internet Options->General ->Languages->Add

Choose `Italian[it]`, then Ok, make sure you move `Italian[it]` to be on the top of the Language list.

Note that at the moment the culture set in `Web.conf` is `en-US` and the culture in the operating system (set via control panel) is `fr-FR`. When you run the application, the culture will be `it-IT` as illustrated below:

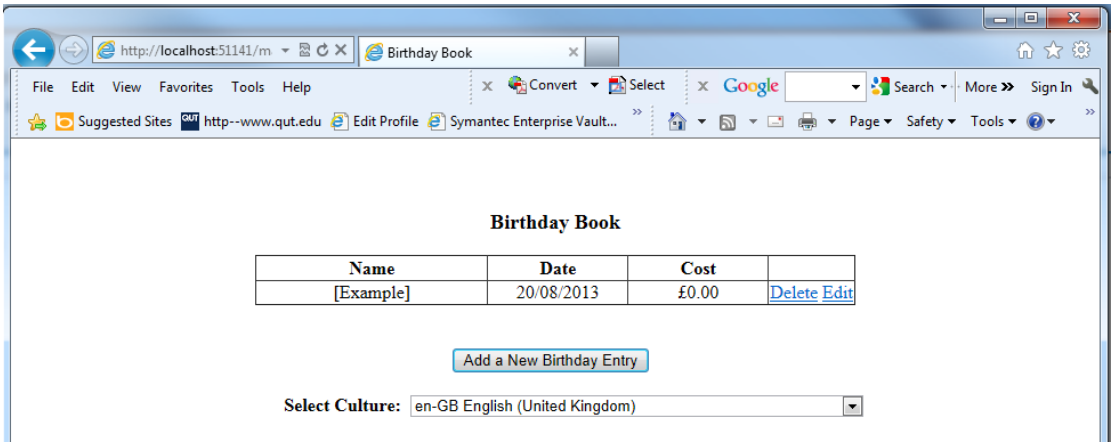


This indicates that the client browser's language preference has a higher priority than the culture set in Control Panel and also the culture set in `Web.conf`.

- Now, we want to set specific cultures in `@Page` directive, e.g., `en-GB`, as shown below:

```
<%@Page Language="C#" AutoEventWireup="true" culture="en-GB"
uiculture="en-GB" CodeBehind="mainPage.aspx.cs"   Inherits
="BirthdayBookWeb.mainPage"%>
```

Run the application, you will find that the culture used in the page is `en-GB`, which means, the specific culture set in `@Page` directive has the highest priority.

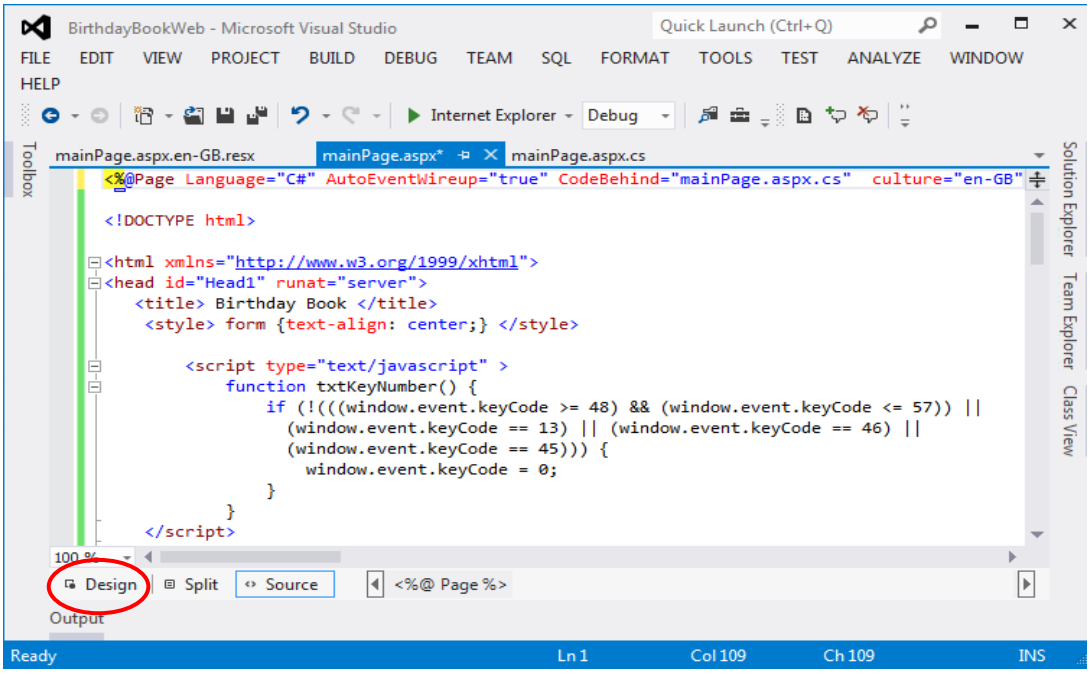


2. UI Localization using Resource files

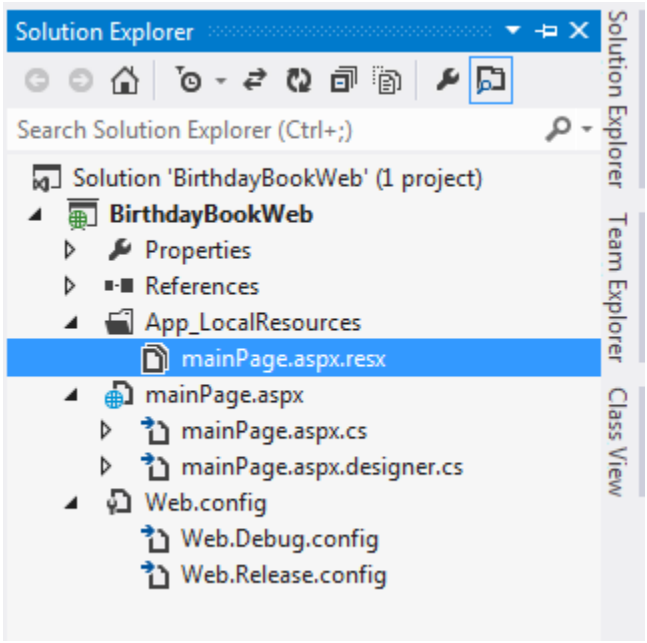
The same as Windows applications, UI content localization in Web applications can be achieved by using resources files. A resource file is an XML document that contains the strings that you want to translate into different languages. The resource file contains key/value pairs. Each pair is an individual resource, representing the source string and its translation for a particular locale.

Firstly we need to create a default resource file using the Visual Web Developer. The local resource file will contain the resource strings for the controls on the page. The Visual Web Developer generates a resource string for each property that is designated internally in the control as localizable.

- Open `mainPage.aspx`, at the bottom of the panel, switch to `Design` view as indicated below:



- On the Tools menu, click **Generate Local Resource,** Visual Web Developer creates a new folder named **App_LocalResource folder**, and within the folder, a new file named `maiPage.aspx.resx` .



- Switch to `Source` view to see the changes to your control declarations. Visual Web Developer has added an attribute to your controls to retrieve their values from your newly created resource. For example, in the @page directive, an attribute "resourcekey" has been added:

```
<%@Page Language="C#" AutoEventWireup="true"
CodeBehind="mainPage.aspx.cs"  culture="en-GB" uiCulture="en-GB"
Inherits ="BirthdayBookWeb.mainPage" meta:resourcekey="PageResource1"%>
```

  Actually all the controls have been given the attribute.

This resource file is used as the default resource file for all requests. (It is the resource file for the fallback culture.) If no culture is specified by the browser, or if the browser request includes a language or culture that you do not support, resource values are pulled from this default file.

Now that the resource file is created, you can place localized text within it by using the Resource Editor. In Solution Explorer, open the resource file `maiPage.aspx.resx` . In the Resource Editor, under Value, are the Text properties for each of the controls that you placed onto your page. You can change the values if you wish. Also you can delete the control attributes which do not have values by right-clicking the attribute (left side) and choosing 'Delete'. After deleting these attributes, the resource table may look like:

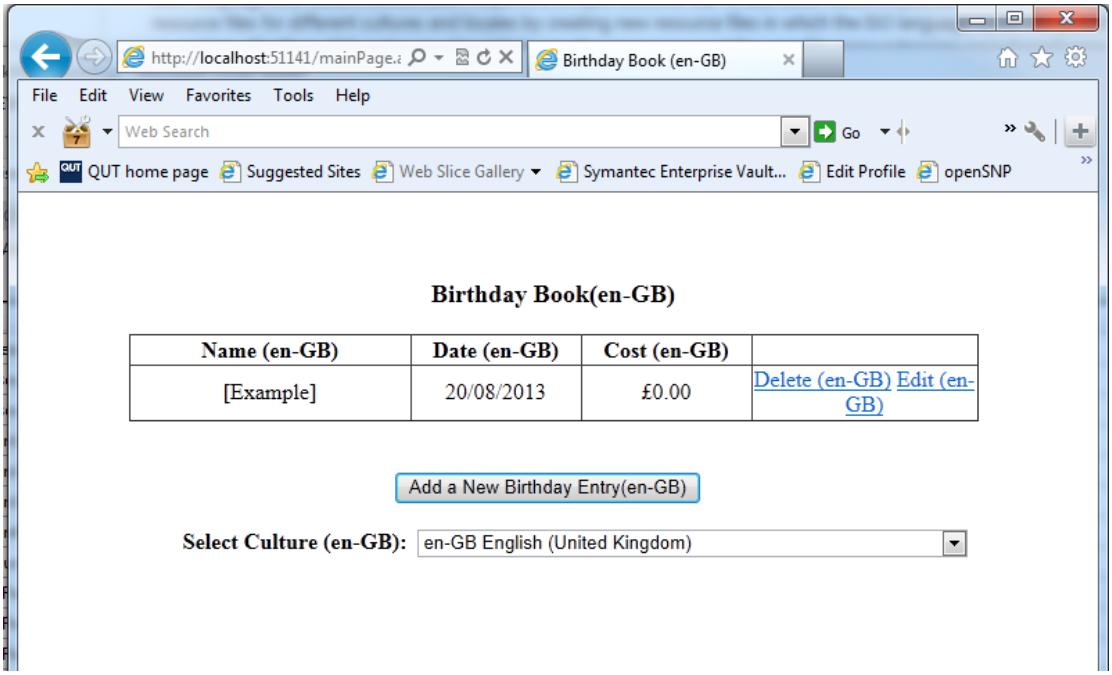
Name	Value	Comment
Button1Resource2.Text	Add a New Birthday Entry	
Label1Resource1.Text	Birthday Book	
Label2Resource1.Text	Select Culture:	
LinkButton1Resource1.Text	Delete	
LinkButton2Resource1.Text	Edit	
LinkButton3Resource1.Text	Update	
LinkButton4Resource1.Text	Cancle	
PageResource1.Title	Birthday Book	
TemplateFieldResource1.HeaderText	Name	
TemplateFieldResource2.HeaderText	Date	
TemplateFieldResource3.HeaderText	Cost	

Now, we can use the default resource file as a starting point to add more resource files for other cultures.

- In Solution Explorer, right-click the `mainPage.aspx.resx` file, and then click Copy.
- Right-click the **App_LocalResources** folder, and then click Paste, a copy of the default resource file will be created.
- Change the name of the newly created file to `mainPage.aspx.en-GB.resx`.
- Open `mainPage.aspx.en-GB.resx`, and change the values of the controls to some values of your preference, something like:

Name	Value
Button1Resource2.Text	Add a New Birthday Entry(en-GB)
Label1Resource1.Text	Birthday Book(en-GB)
Label2Resource1.Text	Select Culture (en-GB):
LinkButton1Resource1.Text	Delete (en-GB)
LinkButton2Resource1.Text	Edit (en-GB)
LinkButton3Resource1.Text	Update (en-GB)
LinkButton4Resource1.Text	Cancle (en-GB)
PageResource1.Title	Birthday Book (en-GB)
TemplateFieldResource1.HeaderText	Name (en-GB)
TemplateFieldResource2.HeaderText	Date (en-GB)
TemplateFieldResource3.HeaderText	Cost (en-GB)

Now, run the application, your page may look like:



You can use the same way to add more resource files.

In the practice above, we used ASP.NET **implicit localization** to have controls display localized text. We generated a resource file with property values in it. Implicit localization works automatically, in that you do not need to specify how to read information from a resource file.

There is another way to create resource files which is called **Explicit Localization.** Explicit localization allows you to have more direct control over how properties are set. If you are interested, you can find more information about explicit localization with ASP.NET at http://msdn.microsoft.com/en-us/library/fw69ke6f(v=vs.90).aspx.