Storing HealthVault User Image in Windows Azure Blob Storage

Author: Mark Arteaga
Date: July 31, 2012

In this article, we'll look at how to save a HealthVault user's avatar image into Windows Azure blob storage for use in an ASP.NET MVC web application.

We will continue of the project created in the previous article Deploying HealthVault Application to Windows Azure and update our MVC web app to show the avatar image.

Before you get started, make sure you create a new Azure Storage. If you have not created one before see How to Create a Storage Account for Windows Azure Subscription.

Creating the Helper Class

First thing we need to do is create a helper singleton class to retrieve and save the image to Azure blob storage.

1. Create a new **Class** called **HVUserImageHelper.cs**
2. Add the following using statements

```
using Microsoft.Health;
using Microsoft.Health.ItemTypes;
using Microsoft.WindowsAzure;
using Microsoft.WindowsAzure.StorageClient;
using System.IO;
```

3. Add the following static properties

```
private static HVUserImageHelper _default;
public static HVUserImageHelper Default
{
    get
    {
        if (_default == null)
            _default = new HVUserImageHelper();
        return _default;
    }
}
```

4. Add the following internal variables

```
// Azure blob storage keys
private string _accountName = "[CHANGE THIS]";
private string _accountKey = "[CHANGE THIS";
private string _containerName = "userimages";
```

5. Add the following constructor

```
private HVUserImageHelper() { }
```

6. Add the following Property

```
private string ConnectionString
        {
            get
            {
                return
string.Format("DefaultEndpointsProtocol=https;AccountName={0};AccountKey={1}",
_accountName, _accountKey);
            }
        }
```

7. Add the following method

```
/// <summary>
/// Saves a HealthVault record image to blob storage for future use
/// </summary>
/// <param name="record"></param>
public void SaveImageToBlobStorage(HealthRecordInfo record)
{
    // get the items for the health record and set the type to personImage
    var collection = record.GetItemsByType(PersonalImage.TypeId,
HealthRecordItemSections.All);

    PersonalImage image = null;
    if (collection.Count != 0)
    {
        // get the first item which is the image
        image = collection[0] as PersonalImage;

        // Create a stream to read the image into
        using (Stream currentImageStream = image.ReadImage())
        {
            // Read the image
            byte[] imageBytes = new byte[currentImageStream.Length];
            currentImageStream.Read(imageBytes, 0, (int)currentImageStream.Length);

            // create vars to access blob storage account
            var account = CloudStorageAccount.Parse(ConnectionString);
            var client = account.CreateCloudBlobClient();
            var container = client.GetContainerReference(_containerName);
            // if it does not exist create it
            if (container.CreateIfNotExist())
            {
                // since it's new we want to change the persmissions of the container to
be public
                var p = new BlobContainerPermissions();
                p.PublicAccess = BlobContainerPublicAccessType.Container;
                container.SetPermissions(p);
            }

            // get a block blob reference
            var item = container.GetBlockBlobReference(record.Id.ToString() + ".jpg");

            // set the type to image
```

```
            item.Properties.ContentType = "image\\jpeg";

            // upload to blob storage
            item.UploadByteArray(imageBytes);
        }
    }
}
```

8. Compile the project.

Now that we have our helper class we can use it to save the images.

Selected Record and Ids

HealthVault has a lot of Ids in the system to identify records. If you delve into the PersonInfo object, you will notice that there is a SelectedRecord property which signifies which record you are currently working with as a HealthVault user may have multiple records within their account. To account for this and to be able to show the avatar image, we need to store the SelectedRecord.Id in our IPrincipal object.

Open **HVPrincipal.cs** and add the following property

```
/// <summary>
/// This is the id from the Selected record as it is different from the Person Info
/// </summary>
public Guid RecordId { get; set; }
```

In the constructor add the following line

```
RecordId = pi.SelectedRecord.Id;
```

Now we are ready to call our helper class and display to the user.

Calling Helper Class

Through the development of Asthma Journal, I determined it was best to save the image when the user signs in. This way, if they decide to change the image on the HealthVault system, we can update our local system.

1. Open **AccountController.cs**
2. In the **Login** method locate the following comment
   `// redirect to the actionqs`
3. Above this line add the following

```
// save the user avatar image to blob
HVUserImageHelper.Default.SaveImageToBlobStorage(personInfo.SelectedRecord);
```

4. Compile and run the project.

If all goes well, you will be able to log in and the avatar image will be saved to Azure Blob storage.

Updating UI To Display Image

Now that the image is saved, we want to display it to the user. A nice place to do this is in the _LoginPartial.cshtml file and only display it when the user is logged in.

First open **HVUserImageHelper.cs** and add the following methods to the class

```
/// <summary>
/// Gets a health records image url in Azure blob storage
/// </summary>
/// <param name="record"></param>
/// <returns></returns>
public string GetImageUrl(HealthRecordInfo record)
{
    return GetImageUrl(record.Id);
}

/// <summary>
/// Gets a health records image url in Azure blob storage
/// </summary>
/// <param name="record"></param>
/// <returns></returns>
public string GetImageUrl(Guid id)
{
    return GetImageUrl(id.ToString());
}

/// <summary>
/// Gets the image Url of a user by id in Azure blob storage
/// </summary>
/// <param name="id"></param>
/// <returns></returns>
public string GetImageUrl(string id)
{
    return string.Format("http://{0}.blob.core.windows.net/{1}/{2}.jpg", _accountName,
_containerName, id);
}
```

These are just extra helper methods to help get the url of the image for the user that is currently logged in.

Open **_LoginPartial.cshtml** and add the following `img` tag to the first `<li>` in the true section of the `@if` statement

```
<img src='@Samples.HvMvc.HVUserImageHelper.Default.GetImageUrl((User as
Samples.HvMvc.HVPrincipal).RecordId)' style="width:30px; height:30px; vertical-align:
top; margin-right:10px">
```

Compile and run the project and when you login, you should have the following output.

Conclusion

In this article we covered how to save a user's HealthVault avatar image to Windows Azure Storage. We also looked at updating the sample code to display the image to the user and retrieve the image from Azure Storage for display.

In the next article, we'll look at integrating our sample HealthVault app with a Windows 8 WinJS application.