# How can I return custom content for specific URLs requested by a Windows Runtime WebView?

devblogs.microsoft.com/oldnewthing/20210217-00

February 17, 2021

Raymond Chen

If you're using a Windows Runtime WebView control, you may want to return custom content when the page requests content from specific URIs.

One scenario for this is that you want to block specific sites.

Another scenario is that you have copied the contents of a site locally, and you want all intra-site navigations to be served from your local copy. This is a handy way of converting a Web-based site into an app that can run offline.

Yet another scenario is that the site and your app are in cahoots, and the site makes queries to specific URIs, knowing that your app will intercept the request and inject custom results.

Whatever the scenario, you can do this by using the `WebResourceRequested` event. Start with the WebView sample and make these changes:

```
using Windows.Web.Http;

    public Scenario2_NavToString()
    {
        this.InitializeComponent();
        WebViewControl.WebResourceRequested += OnResourceRequested;
    }

    Uri fakeUri = new Uri("http://example.com/fakeme");

    void OnResourceRequested(WebView sender,
             WebViewWebResourceRequestedEventArgs e)
    {
        if (e.Request.RequestUri == fakeUri)
        {
            var response = new HttpResponseMessage(HttpStatusCode.Ok);
            response.Content = new HttpStringContent("Here is some fake content");
            e.Response = response;
        }
    }
```

And edit the `html/html_example.html` file to contain this:

```
<!DOCTYPE html>
<html>
    <button id=tryme>Try me</button><div id=result></div>
    <script>
tryme.addEventListener('click', function(e) {
    var xhr = new XMLHttpRequest();
    xhr.open('GET', 'http://example.com/fakeme', true);
    xhr.onreadystatechange = function() {
        if (xhr.readyState == XMLHttpRequest.DONE) {
            result.innerText = `status: ${xhr.status}, content =
${xhr.responseText}`;
        }
    };
    xhr.send();
});
</script>
</html>
```

We change the web page content so it issues an `XMLHttpRequest` to the fake site `http://example.com/fakeme` and prints the result.

On the app side, we respond to the `WebResourceRequested` event by seeing if the request is for our custom fake site. If so, then we construct a custom response that is consists of a string of fake content. (If not, then the event handler does nothing, and the request goes out over the wire.)

Raymond Chen

**Follow**