Creating External Components

Ent SP

You can further extend StresStimulus run-time by creating an external component. An external component traps StresStimulus run-time events to extend its functionality.

To create an external component implement the **`StresStimulus.Extensibility.IExternalComponent`** interface. It has the following interface definition:

IExternalComponent Members

```
/// <summary>
/// Interface to handle test events. Must have a no argument - constructor.
/// </summary>
public interface IExternalComponent
{
    /// <summary>
    /// Fired when test started.
    /// </summary>
    void OnTestStart();
    /// <summary>
    /// Fired when test ends.
    /// </summary>
    void OnTestEnd();
    /// <summary>
    /// Fired before a request is sent.
    /// </summary>
    /// <param name="session">The session object</param>
    void OnBeforeRequest(RuntimeSession session);
    /// <summary>
    /// Fired after a response is received.
    /// </summary>
    /// <param name="session">The session object</param>
    void OnAfterResponse(RuntimeSession session);
    /// <summary>
    /// Fired after iteration is complete
    /// </summary>
    /// <param name="vu">The vu object</param>
    void OnAfterIteration(RuntimeVU vu);
}
```

- **OnTestStart()**: Use this method initialize any test specific objects that will be used during the test, or to run any pre-test script (such as cleaning previously created database records).
- **OnTestEnd()**: Use this method to clean up any objects used during the test or run any post-test script.
- **OnBeforeRequest()**: This is called before a request is sent. It receives an instance of **RuntimeSession** object that can be used to customize the request headers and body. This method is called after all parameters have been applied to the request.
- **OnAfterResponse()**: This is called after a response is received. It receives an instance of **RuntimeSession** object to read the response headers and body for further automation.
- **OnAfterIteration()**: This is called after an iteration is complete. It receives an instance of **RuntimeVU** object to read the VU properties.

The **RuntimeSession** class has access to the run-time session object that has access to request and response data. It contains the following properties and methods:

RuntimeSession Members

```
 /// <summary>
/// A replayed runtime session object.
/// </summary>
public class RuntimeSession
```

```
{
    /// <summary>
    /// The recorded session id.
    /// </summary>
    public int RecordedSession { get; }
    /// <summary>
    /// The recorded test case name.
    /// </summary>
    public string RecordedTestCase { get; }
    /// <summary>
    /// The current iteration number.
    /// </summary>
    public int IterationNumber { get; }
    /// <summary>
    /// The current request number in iteration.
    /// </summary>
    public int RequestNumber { get; }
    /// <summary>
    /// The VU number.
    /// </summary>
    public int VUNumber { get; }
    /// <summary>
    /// The extractor runtime object for the VU.
    /// </summary>
    public ExtractorRuntime ExtractorRuntime { get; }
        /// <summary>
    /// Returns the data source with the given name.
    /// </summary>
    /// <param name="name">The name of the data set.</param>
    /// <returns>The DataTable object that represents the data set. Returns null if data source does not exist.<
/returns>
    public DataTable GetDatasource(string name);
    /// <summary>
    /// The url.
    /// </summary>
    public Uri Url { get; }
    /// <summary>
    /// Get request headers.
    /// </summary>
    /// <returns>An enumerable list of request headers.</returns>
    public IEnumerable<NameValuePair> GetRequestHeaders();
    /// <summary>
    /// Gets a string representation of the request body.
    /// </summary>
    /// <returns>The string representation of the request body.</returns>
    public string GetRequestBody();
    /// <summary>
    /// Gets the raw request body.
    /// </summary>
    /// <returns>The byte[] representation of the request body.</returns>
    public byte[] GetRawRequestBody();
    /// <summary>
    /// Change or add a request header. Set the value to null to remove the header.
    /// </summary>
    /// <param name="name">The name of the request header.</param>
    /// <param name="value">The value of the request header.</param>
    public void SetRequestHeader(string name, string value);
    /// <summary>
    /// Change the request body.
    /// </summary>
    /// <param name="body">The byte[] of the new response body.</param>
    public void SetRequestBody(byte[] body);
    /// <summary>
    /// Returns the response code.
    /// </summary>
    public int ResponseCode;
    /// <summary>
    /// Get response headers.
    /// </summary>
    /// <returns>An enumerable list of response headers.</returns>
    public IEnumerable<NameValuePair> GetResponseHeaders();
```

```
    /// <summary>
    /// Gets a string representation of the response body.
    /// </summary>
    /// <returns>The string representation of the response body.</returns>
    public string GetResponseBody();
    /// <summary>
    /// Gets the raw request body.
    /// </summary>
    /// <returns>Returns the byte[] representation of the response body.</returns>
    public byte[] GetRawResponseBody();
}
/// <summary>
/// Name/value pair representation.
/// </summary>
public class NameValuePair
{
    /// <summary>
    /// The name.
    /// </summary>
    public string Name { get; }
    /// <summary>
    /// The value.
    /// </summary>
    public string Value { get; }
}
```

The **RuntimeVU** class has access to the run-time VU object. It contains the following properties and methods:

RuntimeVU Members

```
/// <summary>
///  A runtime VU object.
/// </summary>
public class RuntimeVU
{
    /// <summary>
    /// The extractor runtime object for the VU.
    /// </summary>
    public ExtractorRuntime ExtractorRuntime { get; }
    /// <summary>
    /// The VU number.
    /// </summary>
    public int VUNumber { get; }
    /// <summary>
    /// The VU iteration number.
    /// </summary>
    public int Iteration { get; }
    /// <summary>
    /// If iteration was aborted
    /// </summary>
    public bool IsAbortedIteration { get; }
    /// <summary>
    /// If virtual user is warming up.
    /// </summary>
    public bool IsWarmup { get; }
    /// <summary>
    /// If virtual user is running a verify.
    /// </summary>
    public bool IsVerify { get; }
    /// <summary>
    /// Which test case is currently being executed.
    /// </summary>
    public string TestCase{ get; }
        /// <summary>
    /// Returns the data source with the given name.
    /// </summary>
    /// <param name="name">The name of the data set.</param>
    /// <returns>The DataTable object that represents the data set. Returns null if data source does not exist.<
/returns>
    public DataTable GetDatasource(string name);
}
```

See the subsequent section for examples of external components.