Network Working Group
Internet-Draft
Intended status: Informational
Expires: March 29, 2011

B. Hoehrmann

September 25, 2010

The 'javascript' resource identifier scheme
draft-hoehrmann-javascript-scheme-03

Abstract

This memo defines the 'javascript' resource identifier scheme. Using this scheme, executable script code can be specified in contexts that support resource identifiers.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 29, 2011.

Copyright Notice

1. Introduction

The 'javascript' resource identifier scheme allows to encode script code in a resource identifier in a way similar to the 'data' scheme, but with extended semantics. This document defines the scheme and two operations that describe how existing implementations handle it.

The first operation, source text retrieval, defines which script code a given 'javascript' resource identifier represents. This operation is fully defined in this document and some applications might take advantage of only this operation.

The second operation, in-context evaluation, is often implemented by web browser applications, and provides a means to run custom script code when the resource identifier is dereferenced. As an example, consider a HTML document containing a hyperlink like:

```
<a href='javascript:doSomething()'>...</a>
```

In typical implementations, when the user activates the hyperlink, the web browser will pass control to the doSomething() function, and render its result, if any, in place of the current document.

Some semantics of this operation are out of scope of this document. For instance, in the example above, if the doSomething() function returns a string object, the implementation would lack clues, like an Internet media type, how to process it; it could treat it as a script, style sheet, HTML document, resource identifier, or other type of resource, as appropriate for the context.

In order not to limit the applicability of this scheme for certain applications, this document just describes this operation in terms of an abstract model; it is expected that, where needed, other specifications define the semantics in more detail using this model.

2. Terminology and Conformance

Resource identifiers, including percent-encoding and requirements for IRIs, are defined in STD 66, [RFC3986], and [RFC3987]. Source text and the media type application/javascript are defined in [RFC4329], the 'data' scheme in [RFC2397], and UTF-8, including the term byte order mark, in STD 63, [RFC3629].

An application that generates resource identifiers conforms to this specification if and only if, given a valid application/javascript entity, it generates only 'javascript' resource identifiers that conform to this specification.

An application that dereferences 'javascript' resource identifiers conforms to this specification if and only if it implements the source text retrieval operation as defined in this specification.

A resource identifier conforms to this specification if and only if it is a valid IRI and application of the source text retrieval operation yields a valid application/javascript entity without generating any error. Use of a byte order mark is discouraged; percent-encoding of "/" (U+002F SOLIDUS) characters is encouraged.

A resource identifier is said to have encoding errors when applying the source text retrieval operation results in one or more errors. Resource identifiers with encoding errors do not conform to this specification. The considerations for handling encoding errors in application/javascript entities apply.

3. Operations

This section defines two operations that can be applied to resource identifiers that conform to this specification. Other operations may be defined in other specifications.

3.1. Source text retrieval

This operation retrieves the source text that is included in the scheme-specific part of a given 'javascript' resource identifier.

1. Represent the scheme-specific part as sequence of octets in the UTF-8 character encoding.

2. Replace any percent-encoded octet by its corresponding octet.

3. If the sequence starts with the sequence 0xEF 0xBB 0xBF, the UTF-8 signature, then discard this signature.

4. Decode the octet sequence using the UTF-8 character encoding and transform the result into source text.

3.2. In-context evaluation

This operation defines a model under which applications may evaluate the source text included in a given 'javascript' resource identifier.

1. Retrieve the source text using the source text retrieval operation.

2. Determine the dereference context for further processing.

3. Evaluate the source text in this context and memorize the result as dereference by-product.

4. Process the dereference by-product as appropriate for the dereference context.

4. Interoperability Considerations

The character "#" is used to separate a fragment identifier from the scheme-specific part of a resource identifier and consequently needs to be percent-encoded when used as data in the scheme-specific part. In certain protocol elements some existing implementations treat the character as data regardless of whether it is percent-encoded.

Protocol element designers who wish to sanction this behavior should specify a pre-processing step that applies percent-encoding to this character for the relevant protocol elements. Such a step precludes use of fragment identifiers for 'javascript' resource identifiers.

The in-context evaluation operation is not fully defined in this memo and inherently context-dependant; it follows that implementations can differ in how they support this operation in a given context and some resource identifiers may only function in specific contexts.

For instance, a 'javascript' resource identifier might be embedded in a HTML document and depend on properties of the document. A typical consequence is that hyperlinks using this scheme can be activated in a specific document, but trying to open them in a new browser window or a different document fails.

Specifications for protocol elements that permit resource identifiers usually do not include special provisions for the 'javascript' scheme and implementations consequently vary in where and how they support them. In the interest of interoperability it is therefore advisable to use the scheme only where no viable alternatives exist.

The definition of the scheme does not permit specification of out of band information like which particular incarnation of the underlying scripting language is used by a resource identifier. In consequence version-specific language features may perform unreliably.

5. Security Considerations

A 'javascript' resource identifier contains a application/javascript

entity and the security considerations for such entities apply. The source text retrieval operation has no considerations beyond that; other specifications may define operations in addition to the ones defined in this document; security considerations for them are out of scope.

The in-context evaluation operation necessitates extreme caution in deciding where resource identifiers using this scheme are recognized and permitted and what facilities are made available to script code, like access to private information and operations with side effects.

6. Internationalization Considerations

None beyond those inherent to resource identifiers and entities of type application/javascript. The scheme-specific part of javascript resource identifiers represents JavaScript source text encoded using the UTF-8 character encoding.

7. IANA Considerations

This document registers the 'javascript' scheme as permanent scheme in the Uniform Resource Identifier scheme registry as per [BCP0035].

8. References

8.1. Normative References

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, January 2005.

[RFC3987] Duerst, M. and M. Suignard, "Internationalized Resource Identifiers (IRIs)", RFC 3987, January 2005.

[RFC4329] Hoehrmann, B., "Scripting Media Types", RFC 4329, April 2006.

8.2. Informative References

[BCP0035] Hansen, T., Hardie, T., and L. Masinter, "Guidelines and Registration Procedures for New URI Schemes", BCP 35, RFC 4395, February 2006.

[RFC2397] Masinter, L., "The "data" URL scheme", RFC 2397, August 1998.

Author's Address

Bjoern Hoehrmann
Mittelstrasse 50
39114 Magdeburg
Germany

Email: mailto:bjoern@hoehrmann.de
URI: http://bjoern.hoehrmann.de

Note: Please write "Bjoern Hoehrmann" with o-umlaut (U+00F6) wherever possible, e.g., as "Björn Höhrmann" in HTML and XML.