robbiblubber.org

Diabolic Device Protocol Specification

Version 3.0

Introduction

The **Diabolic Device Protocol** (ddp) is an application layer protocol introduced by **robbiblubber.org** in February 2000 as a simple, HTTP-based means of remote procedure call. In the context of ddp these calls are called (ddp) Web Methods.

ddp typically makes use of the HTTP GET parameters and **Diabolic Data Notation** (ddn) for data interchange, but it does not explicitly restrict usage to these means. Although ddp has developed independently, it may be considered an application of the REST architectural pattern.

This document describes the specification version 3.0 as of April 2022. The specification changes are tracked in the version history at the end of this document.

The robbiblubber.org Spellbook Libraries offer support for ddp in a number of common programming languages.

Base Protocol

The **Diabolic Device Protocol** (ddp) is defined as an application over the HTTP protocol. It uses HTTP requests and responses to implement remote procedure calls.

A HTTP request is used to address a ddp Web Method and submit necessary input data. The method results are returned in a HTTP response message.

ddp Request

The ddp request selects a ddp Web Method by its address (URI) and submits all necessary data.

Simple data is submitted as HTTP GET parameters, complex or large data items are passed in the request payload. The use of HTTP headers for data transfer is allowed if needed.

Method Name

Each ddp Web Method has a unique name and address (URI).

Request parameters should not be part of the Web Method URI.

```
https://robbiblubber.org/ddp/testMethod
```

Sample: a Web Method URI

HTTP Method

When calling a method with simple input data passed as HTTP GET parameters, the HTTP GET method is recommended. For complex input data passed as data payload, the HTTP POST method should be used.

The use of HTTP PUT, DELETE, and PATCH is allowed. In **Diabolic Device Protocol** (ddp) the operation should be defined by the method name instead of HTTP method, though.

HTTP GET Parameters

HTTP GET parameters are widely used to pass simple parameters to web methods.

HTTP Data

Complex input data is stored in the HTTP content section. The preferred data interchange format is the **Diabolic Data Notation** (ddn).

ddp allows all types of data to be passed to a web method in any format.

HTTP Headers

ddp does not require any specific HTTP headers to be set. If data is transmitted, the use of the "content-type" header is recommended.

The transfer of input data via HTTP headers is dissuaded from.

ddp Response

The ddp response replies to the ddp request and passes all result information. The data should be transferred in the content of the response.

HTTP Status Code

The response status code may be used to indicate success or failure of the ddp method call.

ddp allows all HTTP status codes to be returned. The preferable implementation is to return 200 OK for successful operations and 4xx, 5xx for failure.

HTTP Status Text

ddp allows the usage of HTTP status message text to give detailed information about the call status. It is recommended to use the text specified for the HTTP status and pass details in response ddn.

HTTP Data

ddp allows a web method to return all types of data in any format, although preferably in the **Diabolic Data Notation** (ddn). If no data is returned on a successful call, the http Status 204 is recommended.

HTTP Headers

ddp does not require any specific HTTP headers to be set. If data is transmitted, the use of the "content-type" header is recommended.

The transfer of result data via HTTP headers is dissuaded from.

Examples

```
GET /ddp/foo?a=1&b=Hello HTTP/1.1
Host: robbiblubber.org
```

Sample: a simple get request

```
POST /ddp/bar?a=2&b=Text HTTP/1.1
Host: robbiblubber.org
Content-Type: application/ddn
Content-Length: 58

n = 1;
data
{
   values = 2, 8, 10;
   f = true;
}
```

Sample: GET parameters and ddp content

```
HTTP/1.1 204 No Content
```

Sample: simple success response

```
HTTP/1.1 200 OK
Content-Type: application/ddn
Content-Length: 24

success = true;
result = 10082;
```

Sample: success response

```
HTTP/1.1 400 Bad Request
Content-Type: application/ddn
Content-Length: 68

success = false;
reason = Input parameter 'c' missing or out of range;
```

Sample: failure response

Version History

Version 1.0	2000-02-16	released specification
Version 1.1	2000-03-04	support for ddn version 1.1
Version 1.2	2001-04-23	support for ddn version 1.2
Version 2.0	2009-09-01	support for ddn version 2.0, allows HTTP PUT, DELETE, PATCH
Version 2.1	2010-02-16	support for ddn version 2.1
Version 3.0	2022-04-21	support for ddn version 2.0, allows usage of HTTP headers