Network Working Group
Request for Comments: 3875
Category: Informational

D. Robinson
The Apache Software Foundation
K. Coar
The Apache Software Foundation
October 2004

The Common Gateway Interface (CGI) Version 1.1

Status of this Memo

This memo provides information for the Internet community. It does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Copyright Notice

IESG Note

This document is not a candidate for any level of Internet Standard. The IETF disclaims any knowledge of the fitness of this document for any purpose, and in particular notes that it has not had IETF review for such things as security, congestion control or inappropriate interaction with deployed protocols. The RFC Editor has chosen to publish this document at its discretion. Readers of this document should exercise caution in evaluating its value for implementation and deployment.

Abstract

The Common Gateway Interface (CGI) is a simple interface for running external programs, software or gateways under an information server in a platform-independent manner. Currently, the supported information servers are HTTP servers.

The interface has been in use by the World-Wide Web (WWW) since 1993. This specification defines the 'current practice' parameters of the 'CGI/1.1' interface developed and documented at the U.S. National Centre for Supercomputing Applications. This document also defines the use of the CGI/1.1 interface on UNIX(R) and other, similar systems.

Table of Contents

1 Introduction

1.1 Purpose

The Common Gateway Interface (CGI) [22] allows an HTTP [1], [4] server and a CGI script to share responsibility for responding to client requests. The client request comprises a Uniform Resource Identifier (URI) [11], a request method and various ancillary information about the request provided by the transport protocol.

The CGI defines the abstract parameters, known as meta-variables, which describe a client's request. Together with a concrete programmer interface this specifies a platform-independent interface between the script and the HTTP server.

The server is responsible for managing connection, data transfer, transport and network issues related to the client request, whereas the CGI script handles the application issues, such as data access and document processing.

1.2 Requirements

The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'MAY' and 'OPTIONAL' in this document are to be interpreted as described in BCP 14, RFC 2119 [3].

An implementation is not compliant if it fails to satisfy one or more of the 'must' requirements for the protocols it implements. An implementation that satisfies all of the 'must' and all of the 'should' requirements for its features is said to be 'unconditionally compliant'; one that satisfies all of the 'must' requirements but not all of the 'should' requirements for its features is said to be 'conditionally compliant'.

1.3 Specifications

Not all of the functions and features of the CGI are defined in the main part of this specification. The following phrases are used to describe the features that are not specified:

'system-defined'
: The feature may differ between systems, but must be the same for different implementations using the same system. A system will usually identify a class of operating systems. Some systems are defined in section 7 of this document. New systems may be defined by new specifications without revision of this document.

'implementation-defined'
: The behaviour of the feature may vary from implementation to implementation; a particular implementation must document its behaviour.

1.4 Terminology

This specification uses many terms defined in the HTTP/1.1 specification [4]; however, the following terms are used here in a sense which may not accord with their definitions in that document, or with their common meaning.

'meta-variable'
: A named parameter which carries information from the server to the script. It is not necessarily a variable in the operating system's environment, although that is the most common implementation.

'script'
: The software that is invoked by the server according to this interface. It need not be a standalone program, but could be a dynamically-loaded or shared library, or even a subroutine in the server. It might be a set of statements interpreted at run-time, as the term 'script' is frequently understood, but that is not a requirement and within the context of this specification the term has the broader definition stated.

'server'
: The application program that invokes the script in order to service requests from the client.

2 Notational Conventions and Generic Grammar

2.1 Augmented BNF

All of the mechanisms specified in this document are described in both prose and an augmented Backus-Naur Form (BNF) similar to that used by RFC 822 [13]. Unless stated otherwise, the elements are case-sensitive. This augmented BNF contains the following constructs:

name = definition
: The name of a rule and its definition are separated by the equals character ('='). Whitespace is only significant in that continuation lines of a definition are indented.

"literal"
: Double quotation marks (") surround literal text, except for a literal quotation mark, which is surrounded by angle-brackets ('<' and '>').

rule1 | rule2
: Alternative rules are separated by a vertical bar ('|').

(rule1 rule2 rule3)
: Elements enclosed in parentheses are treated as a single element.

*rule
: A rule preceded by an asterisk ('*') may have zero or more occurrences. The full form is '*n***m* rule' indicating at least *n* and at most *m* occurrences of the rule. *n* and *m* are optional decimal values with default values of 0 and infinity respectively.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.