Individual Submission
Internet-Draft
Intended status: Standards Track
Expires: August 27, 2020

J. Snell, Ed.
NearForm Research
N. Greco
J. Benet
D. Dalrymple
D. Dias
L. Gierth
Protocol Labs
February 24, 2020

Multiformats
draft-snell-multihash-00

Abstract

Defines Multiformats, a collection of data formats that aim to future-proof systems by adding self-description to format values; and defines the Multihash and Multibase Multiformat data formats.

Status of This Memo

This Internet-Draft is submitted to IETF 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 https://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 August 27, 2020.

Copyright Notice

Table of Contents

1. Introduction

Multiformats are a collection of data formats that aim to future-proof systems by adding self-description to format values; allowing interoperability and protocol agility while avoiding lock in.

The self-describing aspects of the Multiformat values have a few stipulations:

- They MUST be in-band (encoded within and carried with the value).
- They MUST avoid lock-in and promote extensibility.
- They MUST be compact and have a binary-packed representation.
- They MUST have a human-readable representation.

This specification defines the Multihash and Multibase formats.

This specification also defines extensible registries for codec and base-encoding identifiers.

New Multiformat data formats SHOULD be registered within the Codec Registry using the "multiformat" type identifier.

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in [RFC2119].

This specification uses the Augmented Backus-Naur Form (ABNF) notation of [RFC5234]

2. Unsigned Variable Length Integer

The various Multiformats defined in this specification make use of the Unsigned Variable Length Integer encoding defined in Appendix C of the [LEB128] standard.

As suggested by the name, this variable length encoding is only capable of representing unsigned integers. Further, while there is no theoretical maximum integer value that can be represented by the format, implementations MUST NOT encode more than nine (9) bytes giving a practical limit of integers in a range between 0 and 2^63 - 1.

Specific Multiformats using the Unsigned Variable Length Integer encoding MAY explicitly declare a more restricted range of encoded values but MUST NOT require values greater than 2^63 - 1.

All integer values between 0 and 127 are encoded as a single byte that is identical to the input. That is, the value 1 is encoded as 00000001, and the value 127 is encoded as 01111111.

Values greater than 127 require two or more bytes as illustrated in the examples below.

Numbers MUST be encoded using the least number of bytes possible.

Integer	Encoding
1	00000001
127	01111111
128	10000000 00000001
255	11111111 00000001
300	10101100 00000010
16384	10000000 10000000 00000001

2.1. ABNF Definition

Specifications signal use of the Unsigned Variable Length Encoded Integer format by referencing the unsigned-varint ABNF grammar rule defined below.

```
varint-terminator = %x00-7f
varint-continuation = %x80-ff
unsigned-varint = *8varint-continuation varint-terminator
```

3. Multibase

Multibase is a format for differentiating outputs from various well-established text base-encoding algorithms.

A Multibase consists simply of a single prefix character that identifies the base-encoding of the remaining characters. All valid Multibase prefix characters are registered in The Multibase Registry.

The prefix character for each base are selected such that they are included in the alphabets of the base they represent. For example, "f" is the base code for base16 (hex) because "f" is in hex's 16 character alphabet.

3.1. Examples

Given the sample text, "Multibase is awesome! \o/", the following example Multibase encodings may be generated.

Base	Multibase
16 (upper)	F4D756C7469626173652069732061765736F6D6521205C6F2F
16 (lower)	f4d756c7469626173652069732061765736f6d6521205c6f2f
32	BJV2WY5DJMJQXGZJANFZSAYLXMVZW63LFEEQFY3ZP
58	zYAjKoNbau5KiqmHPmSxYCvn66dA1vLmwbt
64	MTXVsdGliYXNlIGlzIGF3ZXNvbWUhIFxvLw==

3.2. Multibase ABNF

ABNF

```
base-identifier = <ascii character>
multibase = base-identifier {base-encoded-data}
```

4. Multihash

Multihash is a format for differentiating outputs from various well-established cryptographic hash functions while adressing output length and encoding considerations.

Multihash values encode the hash function output prefixed by an id identifying the hash function encoded as an unsigned variable length integer, and the length of the hash function output encoded as an unsigned variable length integer. All valid Multihash function identifiers are registered in the Codec Registry.

Multihash values are byte sequences that MAY be encoded in multiple base-encodings including hex, base64, base32, or others. When representing a Multihash using a base-encoding, the Multibase format SHOULD be used.

4.1. Example Multihashes

Given the sample text "multihash", the following example Multihash values may be generated:

Function	ID	Multihash (Multibase/Base16)
sha1	0x11	f111488c2f11fb2ce392acb5b2986e640211c4690073e
sha2-256	0x12	f12209cbc07c3f991725836a3aa2a581ca2029198aa420b9d99bc0e131d9f3e2cbe47

4.2. Multihash ABNF

Multihash

```
hash-function-code = unsigned-varint
hash-digest-size = unsigned-varint
hash = {hash function output}

multihash = hash-function-code hash-digest-size hash
```

5. Security Considerations

There are no additional security considerations introduced by the use of Multiformat data formats. Note, however, that various hash functions that may be used with multiformats may be vulnerable to

various known security issues and limitations (e.g. sha1) and thus SHOULD be avoided.

6. IANA Considerations

6.1. The Codec Registry

6.1.1. Codec Registration

Codec identifiers can be registered using the procedure described herein.

Codecs are registered using the Expert Review policy (see Section 4.5 of [RFC8126]). The goal of the registry is to reflect common use of codecs on the Internet. Therefore, the expert(s) should be strongly biased towards approving registrations, unless they are abusive, frivolous, not likely to be used on the Internet, or actively harmful to the Internet and/or the Web (not merely aesthetically displeasing or architecturally dubious). Expert(s) can withhold registration of codecs that are too general for the proposal application.

Expert(s) will clearly identify any issues that cause a registration to be refused. Advice about the semantics of a proposed codec can be given, but if it does not block registration, this should be explicitly stated.

When a request if approved, the expert(s) shall inform IANA, and the regstration will be processed. The IESG is the final arbiter of any objection.

The Codec Registry is located at https://www.iana.org/assignments/multiformat-codecs. Registration requests can be made by following instructions located there or by sending an email to the multiformat-codecs@ietf.org mailing list.

Registration requests consist of at least the following information:

- *Codec Name*: The name of codec
- *Type*: The type of applicable multiformat
- *Code*: The requested identity code
- *Description*: An optional short text description of the codec.
- *Reference*: A reference to the document that specifies the codec, preferably including a URI that can be used to retrieve a copy of

the document. An indication of the relevant section(s) can also be included but is not required.

Multiformat data formats themselves MAY be registered within The Codec Registry using a Type field value of "multiformat". For all other registrations, the Type field MUST refer to an entry in The Codec Registry whose Type is "multiformat".

The expert(s) can define additional fields to be collected in the registry.

Registrations SHOULD reference a freely available, stable specification.

Note that codecs can be registere by third parties (including the expert(s)), if the expert(s) determine that an unregistered codec is widely deployed and not likely to be registered in a timely manner otherwise. Such registrations still are subject to the requirements defined.

6.1.2. Initial Contents

The initial contents of the Codec Registry include:

Name	Type	Code	Description
identity	multihash	0x00	raw binary
sha1	multihash	0x11	
sha2-256	multihash	0x12	
sha2-512	multihash	0x13	
sha3-512	multihash	0x14	
sha3-384	multihash	0x15	
sha3-256	multihash	0x16	
sha3-224	multihash	0x17	
shake-128	multihash	0x18	
shake-256	multihash	0x19	
keccak-224	multihash	0x1a	keccak has variable output length. The number specifies the core length
keccak-256	multihash	0x1b	
keccak-384	multihash	0x1c	
keccak-512	multihash	0x1d	
murmur3-128	multihash	0x22	
murmur3-32	multihash	0x23	
multicodec	multiformat	0x30	
multihash	multiformat	0x31	
multibase	multiformat	0x33	

dbl-sha2-256	multihash	0x56	
md4	multihash	0xd4	
md5	multihash	0xd5	
bmt	multihash	0xd6	Binary Merkle Tree Hash
x11	multihash	0x1100	
blake2b-8	multihash	0xb201	Blake2b consists of 64 output lengths that give different hashes
blake2b-16	multihash	0xb202	
blake2b-24	multihash	0xb203	
blake2b-32	multihash	0xb204	
blake2b-40	multihash	0xb205	
blake2b-48	multihash	0xb206	
blake2b-56	multihash	0xb207	
blake2b-64	multihash	0xb208	
blake2b-72	multihash	0xb209	
blake2b-80	multihash	0xb20a	
blake2b-88	multihash	0xb20b	
blake2b-96	multihash	0xb20c	
blake2b-104	multihash	0xb20d	
blake2b-112	multihash	0xb20e	
blake2b-120	multihash	0xb20f	
blake2b-128	multihash	0xb210	
blake2b-136	multihash	0xb211	
blake2b-144	multihash	0xb212	
blake2b-152	multihash	0xb213	
blake2b-160	multihash	0xb214	
blake2b-168	multihash	0xb215	
blake2b-176	multihash	0xb216	
blake2b-184	multihash	0xb217	
blake2b-192	multihash	0xb218	
blake2b-200	multihash	0xb219	
blake2b-208	multihash	0xb21a	
blake2b-216	multihash	0xb21b	
blake2b-224	multihash	0xb21c	
blake2b-232	multihash	0xb21d	
blake2b-240	multihash	0xb21e	
blake2b-248	multihash	0xb21f	
blake2b-256	multihash	0xb220	
blake2b-264	multihash	0xb221	
blake2b-272	multihash	0xb222	
blake2b-280	multihash	0xb223	
blake2b-288	multihash	0xb224	
blake2b-296	multihash	0xb225	
blake2b-304	multihash	0xb226	
blake2b-312	multihash	0xb227	
blake2b-320	multihash	0xb228	
blake2b-328	multihash	0xb229	

blake2b-336	multihash	0xb22a		
blake2b-344	multihash	0xb22b		
blake2b-352	multihash	0xb22c		
blake2b-360	multihash	0xb22d		
blake2b-368	multihash	0xb22e		
blake2b-376	multihash	0xb22f		
blake2b-384	multihash	0xb230		
blake2b-392	multihash	0xb231		
blake2b-400	multihash	0xb232		
blake2b-408	multihash	0xb233		
blake2b-416	multihash	0xb234		
blake2b-424	multihash	0xb235		
blake2b-432	multihash	0xb236		
blake2b-440	multihash	0xb237		
blake2b-448	multihash	0xb238		
blake2b-456	multihash	0xb239		
blake2b-464	multihash	0xb23a		
blake2b-472	multihash	0xb23b		
blake2b-480	multihash	0xb23c		
blake2b-488	multihash	0xb23d		
blake2b-496	multihash	0xb23e		
blake2b-504	multihash	0xb23f		
blake2b-512	multihash	0xb240		
blake2s-8	multihash	0xb241	Blake2s consists of 32 output lengths that give different hashes	
blake2s-16	multihash	0xb242		
blake2s-24	multihash	0xb243		
blake2s-32	multihash	0xb244		
blake2s-40	multihash	0xb245		
blake2s-48	multihash	0xb246		
blake2s-56	multihash	0xb247		
blake2s-64	multihash	0xb248		
blake2s-72	multihash	0xb249		
blake2s-80	multihash	0xb24a		
blake2s-88	multihash	0xb24b		
blake2s-96	multihash	0xb24c		
blake2s-104	multihash	0xb24d		
blake2s-112	multihash	0xb24e		
blake2s-120	multihash	0xb24f		
blake2s-128	multihash	0xb250		
blake2s-136	multihash	0xb251		
blake2s-144	multihash	0xb252		
blake2s-152	multihash	0xb253		
blake2s-160	multihash	0xb254		
blake2s-168	multihash	0xb255		
blake2s-176	multihash	0xb256		
blake2s-184	multihash	0xb257		

blake2s-192	multihash	0xb258		
blake2s-200	multihash	0xb259		
blake2s-208	multihash	0xb25a		
blake2s-216	multihash	0xb25b		
blake2s-224	multihash	0xb25c		
blake2s-232	multihash	0xb25d		
blake2s-240	multihash	0xb25e		
blake2s-248	multihash	0xb25f		
blake2s-256	multihash	0xb260		
skein256-8	multihash	0xb301	Skein256 consists of 32 output lengths that give different hashes	
skein256-16	multihash	0xb302		
skein256-24	multihash	0xb303		
skein256-32	multihash	0xb304		
skein256-40	multihash	0xb305		
skein256-48	multihash	0xb306		
skein256-56	multihash	0xb307		
skein256-64	multihash	0xb308		
skein256-72	multihash	0xb309		
skein256-80	multihash	0xb30a		
skein256-88	multihash	0xb30b		
skein256-96	multihash	0xb30c		
skein256-104	multihash	0xb30d		
skein256-112	multihash	0xb30e		
skein256-120	multihash	0xb30f		
skein256-128	multihash	0xb310		
skein256-136	multihash	0xb311		
skein256-144	multihash	0xb312		
skein256-152	multihash	0xb313		
skein256-160	multihash	0xb314		
skein256-168	multihash	0xb315		
skein256-176	multihash	0xb316		
skein256-184	multihash	0xb317		
skein256-192	multihash	0xb318		
skein256-200	multihash	0xb319		
skein256-208	multihash	0xb31a		
skein256-216	multihash	0xb31b		
skein256-224	multihash	0xb31c		
skein256-232	multihash	0xb31d		
skein256-240	multihash	0xb31e		
skein256-248	multihash	0xb31f		
skein256-256	multihash	0xb320		
skein512-8	multihash	0xb321	Skein512 consists of 64 output lengths that give different hashes	
skein512-16	multihash	0xb322		
skein512-24	multihash	0xb323		

skein512-32	multihash	0xb324	
skein512-40	multihash	0xb325	
skein512-48	multihash	0xb326	
skein512-56	multihash	0xb327	
skein512-64	multihash	0xb328	
skein512-72	multihash	0xb329	
skein512-80	multihash	0xb32a	
skein512-88	multihash	0xb32b	
skein512-96	multihash	0xb32c	
skein512-104	multihash	0xb32d	
skein512-112	multihash	0xb32e	
skein512-120	multihash	0xb32f	
skein512-128	multihash	0xb330	
skein512-136	multihash	0xb331	
skein512-144	multihash	0xb332	
skein512-152	multihash	0xb333	
skein512-160	multihash	0xb334	
skein512-168	multihash	0xb335	
skein512-176	multihash	0xb336	
skein512-184	multihash	0xb337	
skein512-192	multihash	0xb338	
skein512-200	multihash	0xb339	
skein512-208	multihash	0xb33a	
skein512-216	multihash	0xb33b	
skein512-224	multihash	0xb33c	
skein512-232	multihash	0xb33d	
skein512-240	multihash	0xb33e	
skein512-248	multihash	0xb33f	
skein512-256	multihash	0xb340	
skein512-264	multihash	0xb341	
skein512-272	multihash	0xb342	
skein512-280	multihash	0xb343	
skein512-288	multihash	0xb344	
skein512-296	multihash	0xb345	
skein512-304	multihash	0xb346	
skein512-312	multihash	0xb347	
skein512-320	multihash	0xb348	
skein512-328	multihash	0xb349	
skein512-336	multihash	0xb34a	
skein512-344	multihash	0xb34b	
skein512-352	multihash	0xb34c	
skein512-360	multihash	0xb34d	
skein512-368	multihash	0xb34e	
skein512-376	multihash	0xb34f	
skein512-384	multihash	0xb350	
skein512-392	multihash	0xb351	
skein512-400	multihash	0xb352	
skein512-408	multihash	0xb353	

skein512-416	multihash	0xb354	
skein512-424	multihash	0xb355	
skein512-432	multihash	0xb356	
skein512-440	multihash	0xb357	
skein512-448	multihash	0xb358	
skein512-456	multihash	0xb359	
skein512-464	multihash	0xb35a	
skein512-472	multihash	0xb35b	
skein512-480	multihash	0xb35c	
skein512-488	multihash	0xb35d	
skein512-496	multihash	0xb35e	
skein512-504	multihash	0xb35f	
skein512-512	multihash	0xb360	
skein1024-8	multihash	0xb361	Skein1024 consists of 128 output lengths that give different hashes
skein1024-16	multihash	0xb362	
skein1024-24	multihash	0xb363	
skein1024-32	multihash	0xb364	
skein1024-40	multihash	0xb365	
skein1024-48	multihash	0xb366	
skein1024-56	multihash	0xb367	
skein1024-64	multihash	0xb368	
skein1024-72	multihash	0xb369	
skein1024-80	multihash	0xb36a	
skein1024-88	multihash	0xb36b	
skein1024-96	multihash	0xb36c	
skein1024-104	multihash	0xb36d	
skein1024-112	multihash	0xb36e	
skein1024-120	multihash	0xb36f	
skein1024-128	multihash	0xb370	
skein1024-136	multihash	0xb371	
skein1024-144	multihash	0xb372	
skein1024-152	multihash	0xb373	
skein1024-160	multihash	0xb374	
skein1024-168	multihash	0xb375	
skein1024-176	multihash	0xb376	
skein1024-184	multihash	0xb377	
skein1024-192	multihash	0xb378	
skein1024-200	multihash	0xb379	
skein1024-208	multihash	0xb37a	
skein1024-216	multihash	0xb37b	
skein1024-224	multihash	0xb37c	
skein1024-232	multihash	0xb37d	
skein1024-240	multihash	0xb37e	
skein1024-248	multihash	0xb37f	
skein1024-256	multihash	0xb380	
skein1024-264	multihash	0xb381	

skein1024-272	multihash	0xb382	
skein1024-280	multihash	0xb383	
skein1024-288	multihash	0xb384	
skein1024-296	multihash	0xb385	
skein1024-304	multihash	0xb386	
skein1024-312	multihash	0xb387	
skein1024-320	multihash	0xb388	
skein1024-328	multihash	0xb389	
skein1024-336	multihash	0xb38a	
skein1024-344	multihash	0xb38b	
skein1024-352	multihash	0xb38c	
skein1024-360	multihash	0xb38d	
skein1024-368	multihash	0xb38e	
skein1024-376	multihash	0xb38f	
skein1024-384	multihash	0xb390	
skein1024-392	multihash	0xb391	
skein1024-400	multihash	0xb392	
skein1024-408	multihash	0xb393	
skein1024-416	multihash	0xb394	
skein1024-424	multihash	0xb395	
skein1024-432	multihash	0xb396	
skein1024-440	multihash	0xb397	
skein1024-448	multihash	0xb398	
skein1024-456	multihash	0xb399	
skein1024-464	multihash	0xb39a	
skein1024-472	multihash	0xb39b	
skein1024-480	multihash	0xb39c	
skein1024-488	multihash	0xb39d	
skein1024-496	multihash	0xb39e	
skein1024-504	multihash	0xb39f	
skein1024-512	multihash	0xb3a0	
skein1024-520	multihash	0xb3a1	
skein1024-528	multihash	0xb3a2	
skein1024-536	multihash	0xb3a3	
skein1024-544	multihash	0xb3a4	
skein1024-552	multihash	0xb3a5	
skein1024-560	multihash	0xb3a6	
skein1024-568	multihash	0xb3a7	
skein1024-576	multihash	0xb3a8	
skein1024-584	multihash	0xb3a9	
skein1024-592	multihash	0xb3aa	
skein1024-600	multihash	0xb3ab	
skein1024-608	multihash	0xb3ac	
skein1024-616	multihash	0xb3ad	
skein1024-624	multihash	0xb3ae	
skein1024-632	multihash	0xb3af	
skein1024-640	multihash	0xb3b0	
skein1024-648	multihash	0xb3b1	

skein1024-656	multihash	0xb3b2	
skein1024-664	multihash	0xb3b3	
skein1024-672	multihash	0xb3b4	
skein1024-680	multihash	0xb3b5	
skein1024-688	multihash	0xb3b6	
skein1024-696	multihash	0xb3b7	
skein1024-704	multihash	0xb3b8	
skein1024-712	multihash	0xb3b9	
skein1024-720	multihash	0xb3ba	
skein1024-728	multihash	0xb3bb	
skein1024-736	multihash	0xb3bc	
skein1024-744	multihash	0xb3bd	
skein1024-752	multihash	0xb3be	
skein1024-760	multihash	0xb3bf	
skein1024-768	multihash	0xb3c0	
skein1024-776	multihash	0xb3c1	
skein1024-784	multihash	0xb3c2	
skein1024-792	multihash	0xb3c3	
skein1024-800	multihash	0xb3c4	
skein1024-808	multihash	0xb3c5	
skein1024-816	multihash	0xb3c6	
skein1024-824	multihash	0xb3c7	
skein1024-832	multihash	0xb3c8	
skein1024-840	multihash	0xb3c9	
skein1024-848	multihash	0xb3ca	
skein1024-856	multihash	0xb3cb	
skein1024-864	multihash	0xb3cc	
skein1024-872	multihash	0xb3cd	
skein1024-880	multihash	0xb3ce	
skein1024-888	multihash	0xb3cf	
skein1024-896	multihash	0xb3d0	
skein1024-904	multihash	0xb3d1	
skein1024-912	multihash	0xb3d2	
skein1024-920	multihash	0xb3d3	
skein1024-928	multihash	0xb3d4	
skein1024-936	multihash	0xb3d5	
skein1024-944	multihash	0xb3d6	
skein1024-952	multihash	0xb3d7	
skein1024-960	multihash	0xb3d8	
skein1024-968	multihash	0xb3d9	
skein1024-976	multihash	0xb3da	
skein1024-984	multihash	0xb3db	
skein1024-992	multihash	0xb3dc	
skein1024-1000	multihash	0xb3dd	
skein1024-1008	multihash	0xb3de	
skein1024-1016	multihash	0xb3df	
skein1024-1024	multihash	0xb3e0	

For each of the above entries, this document serves as the reference.

6.2. The Multibase Registry

6.2.1. Base-Encoding Registration

Base-encoding identifiers can be registered using the procedure described herein.

Base-encodings are registered using the Expert Review policy (see Section 4.5 of [RFC8126]). The goal of the registry is to reflect common use of codecs on the Internet. Therefore, the expert(s) should be strongly biased towards approving registrations, unless they are abusive, frivolous, not likely to be used on the Internet, or actively harmful to the Internet and/or the Web (not merely aesthetically displeasing or architecturally dubious). Expert(s) can withhold registration of codecs that are too general for the proposal application.

Expert(s) will clearly identify any issues that cause a registration to be refused. Advice about the semantics of a proposed codec can be given, but if it does not block registration, this should be explicitly stated.

When a request if approved, the expert(s) shall inform IANA, and the regstration will be processed. The IESG is the final arbiter of any objection.

The Multibase Registry is located at https://www.iana.org/assignments/multiformat-base. Registration requests can be made by following instructions located there or by sending an email to the multiformat-codecs@ietf.org mailing list.

Registration requests consist of at least the following information:

- *Name*: The name of base-encoding
- *Prefix*: The requested prefix identifier
- *Description*: An optional short text description of the base encoding.
- *Status*: An indicator of the current implementation status of the encoding and prefix. MUST be one of "draft", "candidate", or "default".
- *Reference*: A reference to the document that specifies the base encoding, preferably including a URI that can be used to retrieve

a copy of the document. An indication of the relevant section(s) can also be included but is not required.

Each of the Status field values indicates the following meanings:

- *draft*: The encoding has been proposed but is not widely implemented and may be removed in the future.
- *candidate*: The encoding is mature and widely implemented by many but not all implementations.
- *default*: The encoding SHOULD be implemented by all implementations and are widely used.

The expert(s) can change the Status of a registration at any time.

The expert(s) can define additional fields to be collected in the registry.

Registrations SHOULD reference a freely available, stable specification.

Note that base encoding can be registered by third parties (including the expert(s)), if the expert(s) determine that an unregistered base encoding is widely deployed and not likely to be registered in a timely manner otherwise. Such registrations still are subject to the requirements defined.

6.2.2. Initial Contents

The initial contents of the Multibase Registry are as follows:

Encoding	Code	Description	Status
identity	0x00	8-bit binary (encoder and decoder keeps data unmodified)	default
base2	0	binary (01010101)	candidate
base8	7	octal	draft
base10	9	decimal	draft
base16	f	hexadecimal	default
base16upper	F	hexadecimal	default
base32hex	v	rfc4648 no padding - highest char	candidate
base32hexupper	V	rfc4648 no padding - highest char	candidate
base32hexpad	t	rfc4648 with padding	candidate
base32hexpadupper	T	rfc4648 with padding	candidate
base32	b	rfc4648 no padding	default
base32upper	B	rfc4648 no padding	default
base32pad	c	rfc4648 with padding	candidate
base32padupper	C	rfc4648 with padding	candidate
base32z	h	z-base-32 (used by Tahoe-LAFS)	draft
base58flickr	Z	base58 flicker	candidate
base58btc	z	base58 bitcoin	default
base64	m	rfc4648 no padding	default
base64pad	M	rfc4648 with padding - MIME encoding	candidate
base64url	u	rfc4648 no padding	default
base64urlpad	U	rfc4648 with padding	default

For each of the above entries, this document serves as the reference.

7. Normative References

[LEB128] DWARF Debugging Information Format Workgroup, "DWARF Debugging Information Format", December 2005, <http://dwarfstd.org/doc/Dwarf3.pdf>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

Authors' Addresses

James M Snell (editor)
NearForm Research

Email: jasnell@gmail.com

Nicola Greco
Protocol Labs

Email: me@nicola.io

Juan Benet
Protocol Labs

Email: juan@benet.ai

David A. Dalrymple
Protocol Labs

Email: david.dalrymple@protocol.ai

David Dias
Protocol Labs

Email: mail@daviddias.me

Lars Gierth
Protocol Labs

Email: lars.gierth@gmail.com