Base64 Cheat Sheet

rjl.io/base64/

How it works

Read 3 bytes (24 bits), cut them into four 6-bit groups, map each 0-63 value to one alphabet character. Example `Man`:

Bytes	`77 97 110`
Bits	`01001101 01100001 01101110`
6-bit groups	`010011 010110 000101 101110`
Indexes	`19 22 5 46`
Characters	`T W F u` = `TWFu`

Standard alphabet (RFC 4648 section 4)

0 - 25	`A - Z`
26 - 51	`a - z`
52 - 61	`0 - 9`
62, 63	`+` and `/`
Padding	`=`

Padding

Input	Example	Output	Padding
3 bytes	`Man`	`TWFu`	none
2 bytes	`Ma`	`TWE=`	one `=`
1 byte	`M`	`TQ==`	two `==`

Standard output length is always a multiple of 4. A decoder can restore stripped padding from the length.

Variants

Variant	62 / 63	Padding	Lines
Standard (RFC 4648 s.4)	`+ /`	`=`	none
URL-safe (RFC 4648 s.5)	`- _`	often omitted	none
MIME (RFC 2045)	`+ /`	`=`	76 chars

URL-safe is used for URLs, filenames and JWT parts. MIME is used for email attachments.

Size overhead

Output = `4 x ceil(n / 3)` characters, about **+33%**.

Input	Bytes	Base64 chars
1 KB	1,024	1,368
10 KB	10,240	13,656
100 KB	102,400	136,536
1 MB	1,048,576	1,398,104

MIME adds a line break every 76 characters. Already-compressed data (PNG, JPEG, ZIP) does not shrink again.

Data URI format

```
data:[mime-type]; base64,[data]
...
```

- Good for small assets (about 10 KB or less): saves a request.
- Larger files are better as separate cached files.
- To decode, strip everything up to and including `base64,`.

Other encodings (same tool)

Format	Alphabet	Size	"Hello"
Base64	64 chars, `=`	+33%	`SGVsbG8=`
Hex	`0-9 a-f`	+100%	`48656c6c6f`
Base32 (RFC 4648)	`A-Z 2-7`, `=`	+60%	`JBSWY3DP`
Base58 (Bitcoin)	no `0 O I l`	about +37%	`9Ajdvzr`

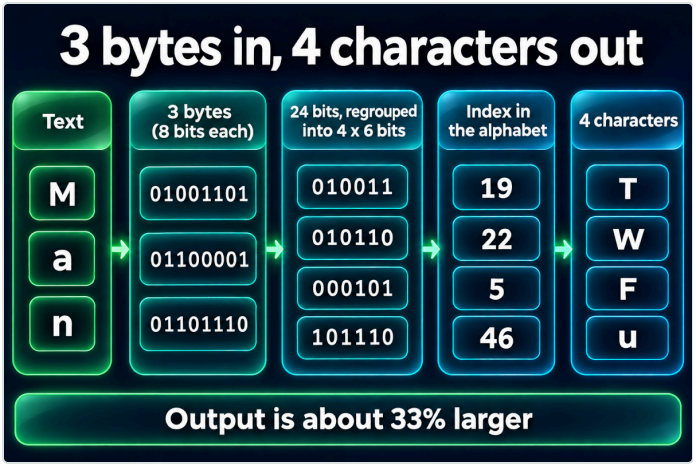
Base32 test vector: `foobar` = `MZXW6YTBOI======`. Base58: `Hello World!` = `2NEpo7TZRRrLZSi2U`; each leading zero byte becomes a leading `1`.

Common mistakes

Symptom	Fix
Base64 is not encryption	Encrypt first, then encode
Garbled accents after decoding	Use the same character set both ways (UTF-8)
`-` or `_` rejected	Map to `+` and `/`, restore padding
Length not a multiple of 4	Append `=` until it is
Fails after copying from email	Remove line breaks and spaces
Output starts with `data:`	Strip the prefix through `base64,`
Decoded text is Base64 again	It was encoded twice; decode once more

JWT and the tool

A JWT is three URL-safe Base64 parts: `header.payload.signature`. Header and payload decode to JSON; the signature is binary. Decoding does not verify it. Deep links: `/base64/?encode=Hello`, `/base64/?decode=SGVsbG8=`, add `&format=hex|base32|base58`.



Worked example: three bytes become four characters.