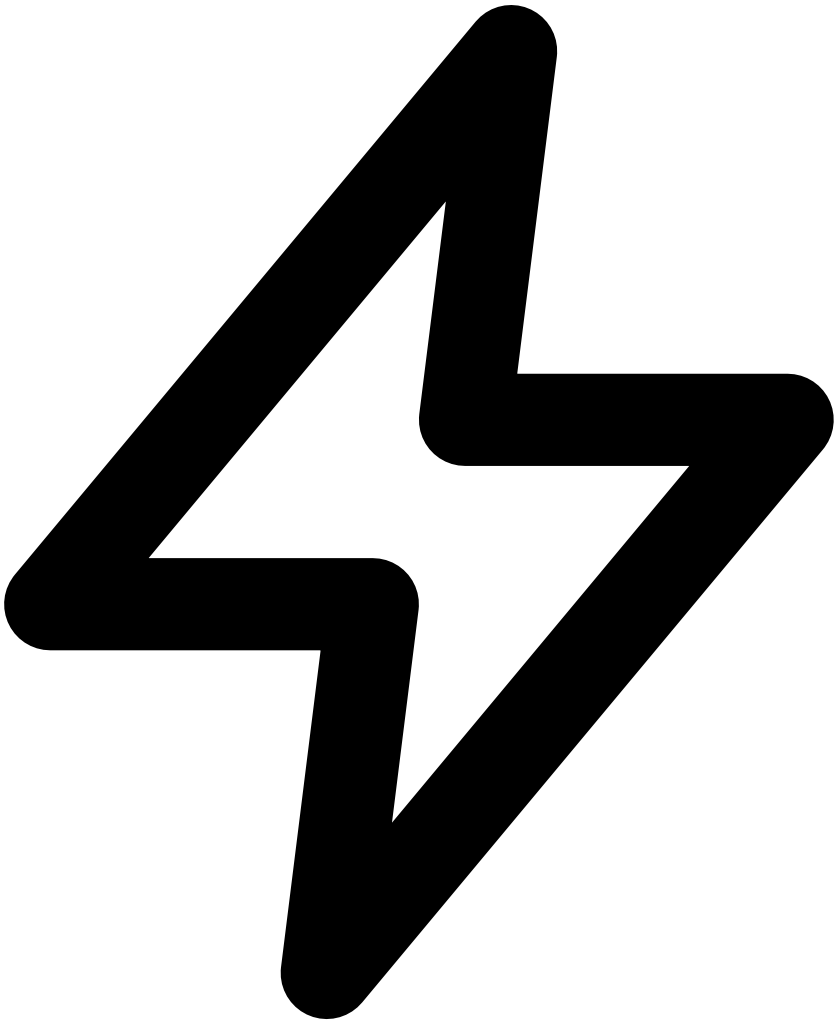

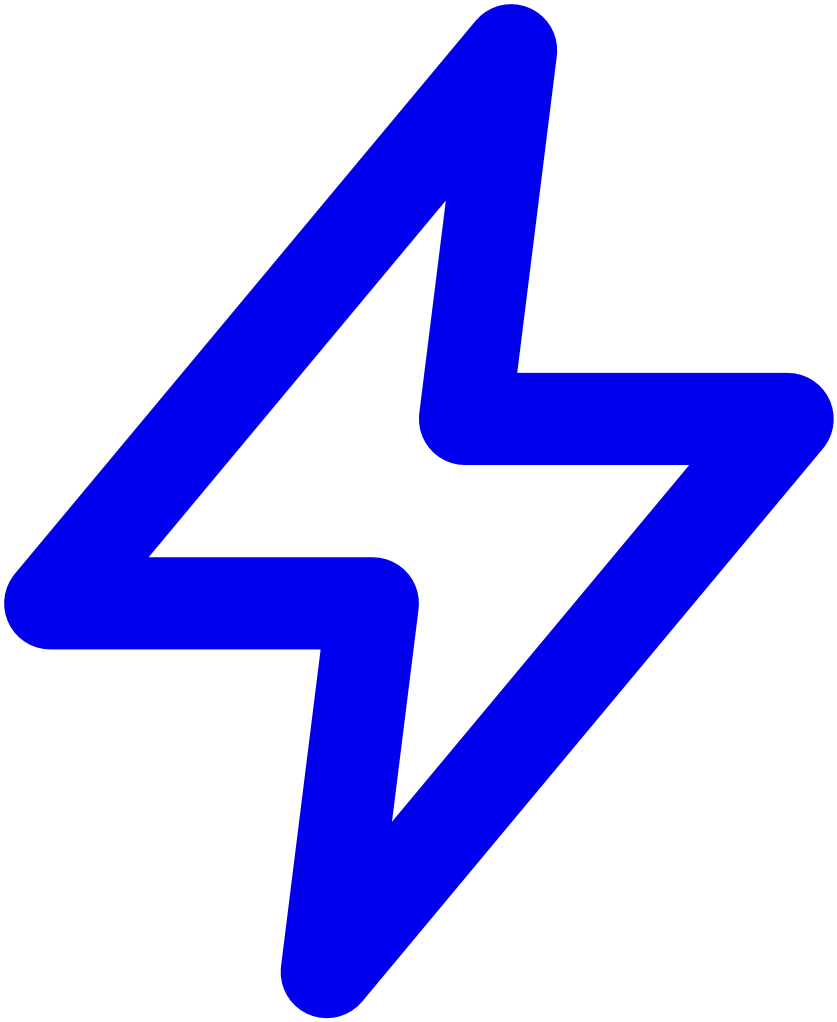

Hexadecimal Explained: Why Developers Count in Base 16

By the **AutonomiqTech Editorial Team** · Reviewed against our editorial standards · 7 min read · Last reviewed 2026

Hexadecimal Explained: Why Developers Count in Base 16

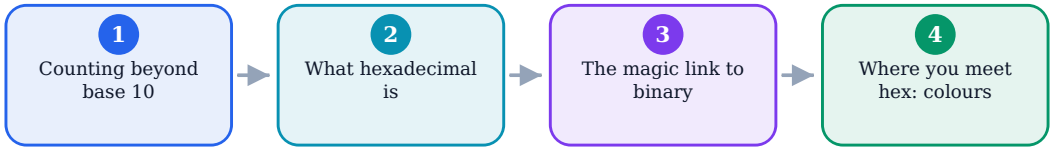

Figure: Hexadecimal Explained: Why Developers Count in Base 16

Anyone who's picked a colour on the web has seen something like `#3498db`, and anyone who's peeked at low-level data has met codes like `0x1F`. These are **hexadecimal** — base 16 — and while base 16 sounds exotic, it exists for a very practical reason: it's a compact, human-friendly way to represent the binary that computers actually use.

This guide explains hexadecimal, how it relates to binary, and where you'll encounter it.

On this page

Counting beyond base 10

We normally count in **base 10**: ten digits (0–9), and when we run out we add a new column. Computers, though, work in **base 2** (binary): just 0 and 1. Binary is natural for machines but painful for humans — even small numbers become long strings of 1s and 0s. Hexadecimal is the compromise.

What hexadecimal is

Hexadecimal is base 16. It uses the familiar digits 0–9 and then borrows letters A–F to represent the values 10 through 15. So counting goes 8, 9, A, B, C, D, E, F, and then 10 (which means sixteen). It looks odd at first, but it's just counting with sixteen symbols instead of ten.

The magic link to binary

Here's why developers love hex: each hexadecimal digit maps to **exactly four binary bits**. Four bits can represent sixteen values (0–15), which is precisely one hex digit. This means you can convert between binary and hex by handling four bits at a time, and two hex digits perfectly represent one **byte** (eight bits).

That clean mapping is what makes hex a compact, readable shorthand for binary data.

Where you meet hex: colours

The most familiar place is **web colours**. A colour like `#3498db` is three pairs of hex digits: red, green and blue, each from 00 to FF (0 to 255). Because two hex digits cover a full byte, this neatly encodes each colour channel's intensity in a compact form.

Where you meet hex: memory and bytes

Beyond colours, hex appears throughout low-level computing. **Memory addresses**, raw **byte values**, and data in debuggers are usually shown in hex because it's far more readable than binary and maps to bytes so cleanly. The `0x` prefix (as in `0x1F`) is a common convention signalling ‘this is hexadecimal’.

A simple way to think about it

You don't need to do hex arithmetic in your head to benefit from understanding it. Just remember: hex is a compact, human-friendly way to write binary, each digit is four bits, and two digits make a byte. With that, colours, addresses and byte dumps stop looking like magic and start making sense.

Decimal, binary and hex side by side

Seeing the same values in different bases makes hexadecimal click, especially its neat relationship with binary:

Decimal	Binary	Hex
10	1010	A
15	1111	F
16	10000	10
255	11111111	FF

Notice how 255 — a mouthful in binary — is just ‘FF’ in hex. That compactness is exactly why hex is so widely used to represent binary data.

Where you'll actually see hex

Hexadecimal shows up in several everyday technical contexts:

- **Colours:** web colours like #FF5733 are hex values for red, green and blue.
- **Memory addresses:** often shown in hex for compactness.
- **Bytes and data:** each byte fits neatly into two hex digits.

- **Encodings and identifiers:** hashes and IDs are frequently in hex.

Why hex pairs so neatly with binary

The reason hexadecimal is so beloved in computing comes down to an elegant mathematical convenience: because 16 is a power of two, every single hex digit corresponds exactly to four binary digits, with no awkward remainder. This means you can translate between binary and hex almost by inspection, grouping bits into fours and swapping each group for its hex digit, which is far easier than converting to and from decimal. Computers ultimately work in binary, but long strings of ones and zeros are painful for humans to read, write and check — a single byte is eight bits, and larger values quickly become unwieldy. Hex compresses that same information into a quarter of the characters while preserving the exact bit pattern, so two hex digits perfectly represent one byte and nothing is lost or approximated in the way it would be if you used decimal. This tidy grouping is why hex is the natural choice for displaying memory addresses, byte values, colour codes and raw data: it stays faithful to the underlying binary while being compact and readable. Once you internalise that each hex digit is simply a shorthand for four bits, a lot of low-level computing notation stops looking mysterious and starts looking sensible. You don't need to perform conversions constantly to benefit from this understanding; recognising that hex is essentially human-friendly binary is enough to make sense of colour codes, addresses and byte dumps whenever you encounter them, which is precisely the practical payoff of understanding the base at all.

Reading a hex colour code

One place almost everyone encounters hexadecimal, even without realising it, is web colours, and understanding the notation turns a cryptic string into something readable. A hex colour such as #FF5733 is really three pairs of hex digits stuck together, each pair describing how much of one primary colour of light to mix: the first pair is red, the second is green, the third is blue. Because a pair of hex digits can range from 00 up to FF, each colour channel has 256 possible levels, from none at all to full intensity, and combining the three channels produces the millions of colours screens can display. Reading it this way, #FF0000 is pure full-strength red with no green or blue, #000000 is all channels off which is black, and #FFFFFF is all channels at maximum which is white. The example #FF5733 therefore means full red, a moderate amount of green, and a little blue, which the eye perceives as a warm orange. You do not need to compute exact values to benefit from this — simply knowing that the code is red, green and blue amounts written in hex lets you predict roughly what a colour will look like and adjust it sensibly, nudging a pair up to add more of that channel or down to reduce it. This is a perfect small illustration of why hex is useful in the first place: it packs the underlying numeric values into a compact, consistent form that is easy to read once you know the pattern, which is exactly the kind of everyday payoff that makes learning the base worthwhile.

Printable checklist

Print this page or save the PDF to keep these steps handy.

- [ ] Counting beyond base 10
- [ ] What hexadecimal is
- [ ] The magic link to binary
- [ ] Where you meet hex: colours
- [ ] Where you meet hex: memory and bytes
- [ ] A simple way to think about it
- [ ] Decimal, binary and hex side by side
- [ ] Where you'll actually see hex

Summary

Hexadecimal is a base-16 number system using digits 0–9 and letters A–F. It exists because each hex digit maps cleanly to exactly four binary bits, making long strings of 1s and 0s far more readable. You'll meet hex in web colours, memory addresses, byte values and debugging — anywhere binary needs a compact, human-friendly form.

Key Takeaways

- Hexadecimal is base 16, using 0–9 then A–F for values 10–15.
- Each hex digit represents exactly four binary bits, so two hex digits equal one byte.
- Hex is a compact, readable stand-in for long binary strings.
- You'll see it in web colours (#RRGGBB), memory addresses and byte values.
- The 0x prefix commonly signals that a number is hexadecimal.

Frequently Asked Questions

Why not just use binary or decimal?

Binary is too long for humans to read comfortably, and decimal doesn't map cleanly to binary. Hexadecimal is the sweet spot: compact like decimal, but each digit maps to exactly four bits.

What does the 0x prefix mean?

It's a convention indicating the number that follows is hexadecimal. For example, 0x1F is a hex value (equal to 31 in decimal), distinguishing it from the decimal number 1F would otherwise be ambiguous.

How do hex colours work?

A hex colour like #3498db encodes red, green and blue channels as three pairs of hex digits, each from 00 to FF. Higher values mean more of that colour.

Related Guides

- UTF-8 vs ASCII

- What Is Base64 Used For?
- Regular Expressions Basics
- What Is an API?

Related articles

- HTTP Status Codes Explained: What 200, 301, 404 and 500 Really Mean
- What Is Base64 Actually Used For? Real-World Examples
- What Is a URL Slug? Best Practices for Clean, Readable URLs

---