G

GUID and UUID: Format, Versions, Examples, and Security

Learn what GUIDs and UUIDs are, how their 128-bit format works, what UUID versions mean, where Windows uses GUIDs, and what security limits apply.

PUBLISHER
Gridinsoft Support

UPDATED
Oct 8, 2026

READ TIME
4 min

CONTENTS

In this guide

01 What does GUID stand for?
02 GUID and UUID format
03 Common UUID versions
04 Where GUIDs appear in Windows
05 Are GUIDs guaranteed to be unique?
06 GUIDs are identifiers, not secrets
07 GUID, hash, and token are different
08 Generate and validate GUIDs safely

A **GUID** (Globally Unique Identifier) is a 128-bit value used to label an object without coordinating with one central numbering authority. **UUID** (Universally Unique Identifier) is the standards-based term. In everyday Windows and .NET use, GUID and UUID usually

describe compatible identifiers such as `550e8400-e29b-41d4-a716-446655440000`.

01 What does GUID stand for?

GUID stands for **Globally Unique Identifier**. Microsoft platforms commonly use the name GUID, while standards and many other ecosystems use UUID. They are generally the same kind of 128-bit identifier; the surrounding API, byte order, textual wrapper, or generation version can still matter when exchanging data.

02 GUID and UUID format

The familiar textual form contains 32 hexadecimal characters divided into an 8-4-4-4-12 pattern:

```
xxxxxxxx-xxxx-Mxxx-Nxxx-xxxxxxxxxxxx
```

The `M` bits identify the UUID version, while bits in `N` identify the variant. Uppercase and lowercase hexadecimal characters represent the same value. Braces, a `urn:uuid:` prefix, or a compact form without hyphens may appear in a particular API, database, or configuration format.

03 Common UUID versions

Version	How it is produced	Typical consideration
UUIDv1	Timestamp and node-related data	Sortable by time, but older generation methods can reveal timing or hardware-related information.
UUIDv3	Namespace and name hashed with MD5	Deterministic; the same namespace and name produce the same UUID.
UUIDv4	Random or pseudorandom bits	Widely used when a non-ordered identifier is sufficient.

Version	How it is produced	Typical consideration
UUIDv5	Namespace and name hashed with SHA-1	Deterministic replacement for many v3 use cases, though SHA-1 is not suitable for modern cryptographic signatures.
UUIDv6	Reordered time-based layout	Designed to sort more efficiently than v1 in many databases.
UUIDv7	Unix timestamp plus random data	Time-ordered and useful for modern distributed databases.
UUIDv8	Application-defined layout	Custom semantics require an agreement between producers and consumers.

04 Where GUIDs appear in Windows

- COM classes, interfaces, and type libraries.
- Windows Registry keys and component identifiers.
- MSI product, package, and upgrade codes.
- Device, volume, partition, and driver identifiers.
- .NET objects, database primary keys, logs, and diagnostic reports.

Some Microsoft binary structures use a different byte order for part of the identifier even though the displayed string looks conventional. Compare identifiers through the relevant UUID/GUID library rather than rearranging bytes manually.

05 Are GUIDs guaranteed to be unique?

No finite identifier can provide a mathematical guarantee across every possible generator and implementation. Correctly generated UUIDs make accidental collisions extraordinarily unlikely for normal workloads. Poor random-number generators, cloned virtual-machine state, broken libraries, manual constants, or truncated values can still create duplicates.

A database should continue to enforce a uniqueness constraint when duplicates would be harmful. Do not assume that a GUID validates the object, proves who created it, or confirms that a record is trustworthy.

06 GUIDs are identifiers, not secrets

A GUID is normally safe to display as an identifier, but it is not an access-control mechanism. If knowledge of an identifier grants access to an invoice, password-reset action, private file, or API object, the application has an authorization vulnerability. Always verify the authenticated user's permission independently.

Context can also make an otherwise random identifier personal or sensitive. A service may associate a device or installation GUID with an account, IP history, or telemetry. Remove unrelated identifiers and access tokens before publishing diagnostic logs.

07 GUID, hash, and token are different

- **GUID/UUID:** labels an object or record.
- **Hash:** is calculated from input data and can be used for comparison or integrity workflows.
- **Token or API key:** may grant access and must be protected as a secret.

A GUID-shaped string is not proof that a file is safe, and a hash-shaped string is not automatically a password.

08 Generate and validate GUIDs safely

Use the operating system, database, or language's standard UUID library. For example, .NET provides `Guid.NewGuid()`, PowerShell provides `New-Guid`, and many databases provide UUID functions or extensions. Choose a version based on ordering, determinism, and privacy requirements.

Validate the complete expected format and version at a trust boundary, but do not confuse format validation with authorization. Never delete an unfamiliar registry key merely because its name is a GUID; identify its owning component and create a backup first.

NEED MORE HELP?

We are here when the guide is not enough.

Visit the Gridinsoft Support Center for current product guidance or submit a support request with the details of your case.

Open this guide online at support.gridinsoft.com