Class

JSONDecoder

An object that decodes instances of a data type from JSON objects.

iOS 8.0+ | iPadOS 8.0+ | Mac Catalyst 8.0+ | macOS 10.10+ | tvOS 9.0+ | visionOS 1.0+ | watchOS 2.0+

```
class JSONDecoder
```

Overview

The example below shows how to decode an instance of a simple `GroceryProduct` type from a JSON object. The type adopts `Codable` so that it's decodable using a `JSONDecoder` instance.

```swift
struct GroceryProduct: Codable {
    var name: String
    var points: Int
    var description: String?
}

let json = """
{
    "name": "Durian",
    "points": 600,
    "description": "A fruit with a distinctive scent."
}
""".data(using: .utf8)!

let decoder = JSONDecoder()
let product = try decoder.decode(GroceryProduct.self, from: json)

print(product.name) // Prints "Durian"
```

Topics

Creating a Decoder

`init()`

Creates a new, reusable JSON decoder with the default formatting settings and decoding strategies.

Decoding

`func decode<T>(T.Type, from: Data) throws -> T`

Returns a value of the type you specify, decoded from a JSON object.

Customizing Decoding

`var keyDecodingStrategy: JSONDecoder.KeyDecodingStrategy`

A value that determines how to decode a type’s coding keys from JSON keys.

`enum KeyDecodingStrategy`

The values that determine how to decode a type’s coding keys from JSON keys.

`var userInfo: [CodingUserInfoKey : any Sendable]`

A dictionary you use to customize the decoding process by providing contextual information.

`var allowsJSON5: Bool`

Specifies that decoding supports the JSON5 syntax.

`var assumesTopLevelDictionary: Bool`

Specifies that decoding assumes the top level of the JSON data is a dictionary, even if it doesn’t begin and end with braces.

Decoding Dates

`var dateDecodingStrategy: JSONDecoder.DateDecodingStrategy`

The strategy used when decoding dates from part of a JSON object.

`enum DateDecodingStrategy`

The strategies available for formatting dates when decoding them from JSON.

Decoding Raw Data

`var dataDecodingStrategy: JSONDecoder.DataDecodingStrategy`

The strategy that a decoder uses to decode raw data.

`enum DataDecodingStrategy`

The strategies for decoding raw data.

Decoding Exceptional Numbers

`var nonConformingFloatDecodingStrategy: JSONDecoder.NonConformingFloatDecodingStrategy`

The strategy used by a decoder when it encounters exceptional floating-point values.

`enum NonConformingFloatDecodingStrategy`

The strategies for encoding nonconforming floating-point numbers, also known as IEEE 754 exceptional values.

Instance Methods

`func decode<T, C>(T.Type, from: Data, configuration: C.Type) throws -> T`

```
func decode<T>(T.Type, from: Data, configuration: T.Decoding
Configuration) throws -> T
```

Relationships

Conforms To

`Copyable`
`NetworkDecoder`
`Sendable`
`SendableMetatype`
`TopLevelDecoder`

See Also

JSON

`class JSONEncoder`

An object that encodes instances of a data type as JSON objects.

`class JSONSerialization`

An object that converts between JSON and the equivalent Foundation objects.