

[Foundation](#) / Archives and Serialization

API Collection

# Archives and Serialization

Convert objects and values to and from property list, JSON, and other flat binary representations.

## Overview

Use these APIs to convert your app's in-memory types to representations suitable for serialization over I/O and network interfaces or to long-term storage.

In Swift, the standard library defines the [Encodable](#), [Decodable](#), and [Codable](#) types, along with [Encoder](#) and [Decoder](#) APIs to perform encoding and decoding, as described in [Encoding, Decoding, and Serialization](#). Foundation extends this with the [EncodableWithConfiguration](#) and [DecodableWithConfiguration](#) protocols, used for types that require additional static information to encode and decode, such as [AttributedString](#).

In Objective-C, [NSCoding](#) defines a protocol for encoding and decoding objects. When adding serialization to your own types, you should also adopt [NSSecureCoding](#). This protocol adds protection against security vulnerabilities introduced by instantiating arbitrary objects as part of the decoding process.

Many system frameworks use these types. When working with external systems, such as URL endpoints, use the JSON and XML APIs to serialize your app's types to standard formats.

## Topics

### Adopting Codability

 [Encoding and Decoding Custom Types](#)

Make your data types encodable and decodable for compatibility with external representations such as JSON.

```
typealias Codable = Decodable & Encodable
```

A type that can convert itself into and out of an external representation.

```
protocol NSCoding
```

A protocol that enables an object to be encoded and decoded for archiving and distribution.

```
protocol NSSecureCoding
```

A protocol that enables encoding and decoding in a manner that is robust against object substitution attacks.

## Serializing Arbitrary Payloads

```
typealias CodableWithConfiguration
```

A type that can convert itself into and out of an external representation with the help of a configuration that handles encoding contained types.

```
struct CodableConfiguration
```

A property wrapper that makes a type codable, by supplying a configuration that provides additional information for serialization.

```
protocol DecodableWithConfiguration
```

A protocol for types that support decoding when supplied with an additional configuration type.

```
protocol DecodingConfigurationProviding
```

A protocol whose conformers provide a configuration instance to help decode types that don't support encoding by themselves.

```
protocol EncodableWithConfiguration
```

A protocol for types that support encoding when supplied with an additional configuration type.

```
protocol EncodingConfigurationProviding
```

A protocol whose conformers provide a configuration instance to help encode types that don't support encoding by themselves.

## JSON

`class JSONEncoder`

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

`class JSONDecoder`

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

`class JSONSerialization`

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

## Property Lists

`class PropertyListEncoder`

An object that encodes instances of data types to a property list.

`class PropertyListDecoder`

An object that decodes instances of data types from a property list.

`class PropertyListSerialization`

An object that converts between a property list and one of several serialized representations.

## XML

☰ XML Processing and Modeling

Parse XML documents.

## Keyed Archivers

`class NSKeyedArchiver`

An encoder that stores an object's data to an archive referenced by keys.

`protocol NSKeyedArchiverDelegate`

The optional methods implemented by the delegate of a keyed archiver.

`class NSKeyedUnarchiver`

A decoder that restores data from an archive referenced by keys.

`protocol NSKeyedUnarchiverDelegate`

The optional methods implemented by the delegate of a keyed unarchiver.

`class NSCoder`

An abstract class that serves as the basis for objects that enable archiving and distribution of other objects.

`class NSDataUnarchiveFromDataTransformer`

A value transformer that converts data to and from classes that support secure coding.

## Deprecated

~~`class NSArchiver`~~

A coder that stores an object's data to an archive.

Deprecated

~~`class NSUnarchiver`~~






A decoder that restores data from an archive.

Deprecated

---

## See Also

### Files and Data Persistence

-  **File System**  
Create, read, write, and examine files and folders in the file system.
-  **Preferences**  
Persistently store domain-scoped pieces of information for configuring your app.
-  **Spotlight**  
Search for files and other items on the local device, and index your app's content for searching.
-  **iCloud**  
Manage files and key-value data that automatically synchronize among a user's iCloud devices.
-  **Optimizing Your App's Data for iCloud Backup**  
Minimize the space and time that backups take to create by excluding purgeable and nonpurgeable data from backups.