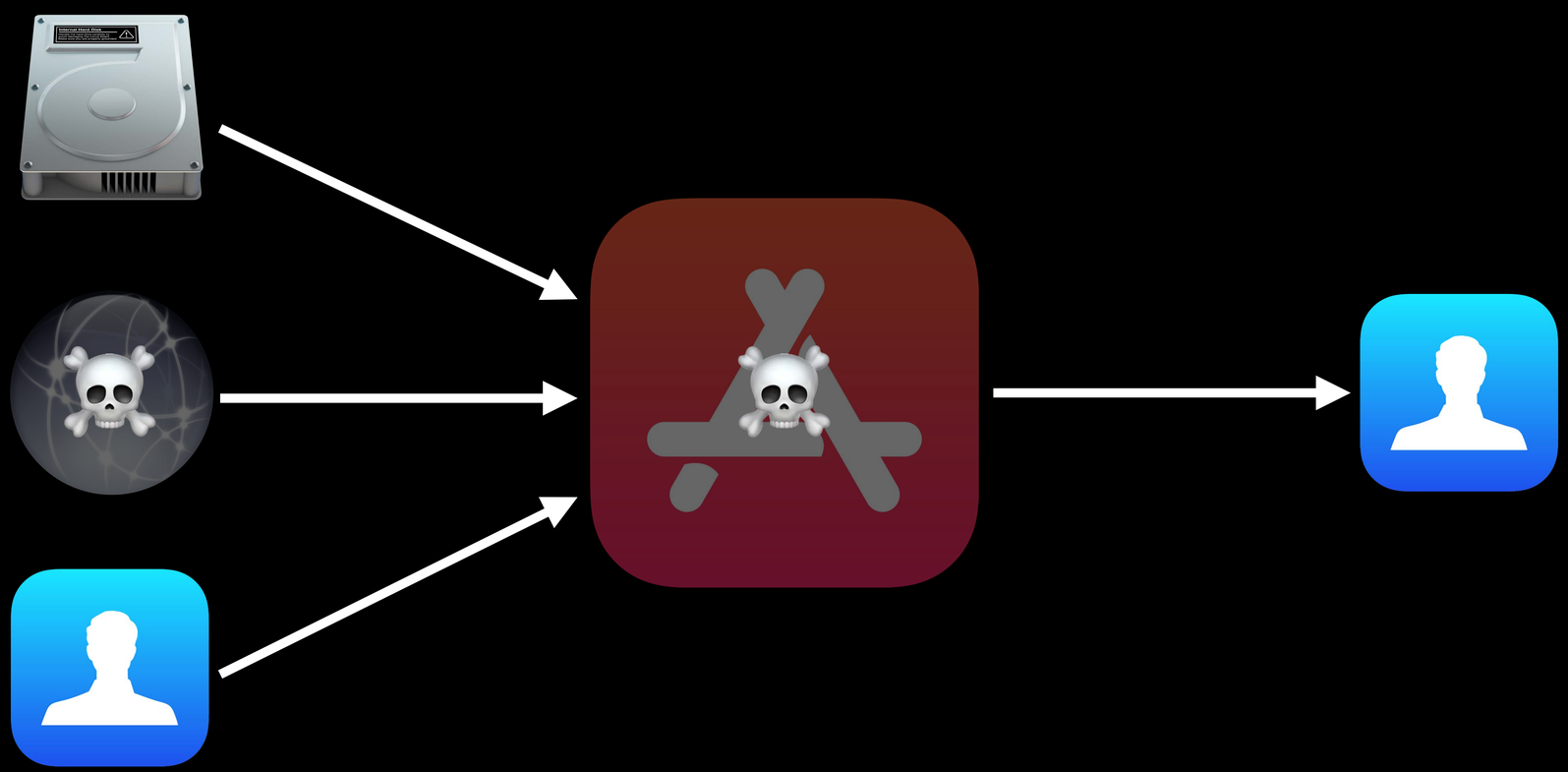
#WWDC18

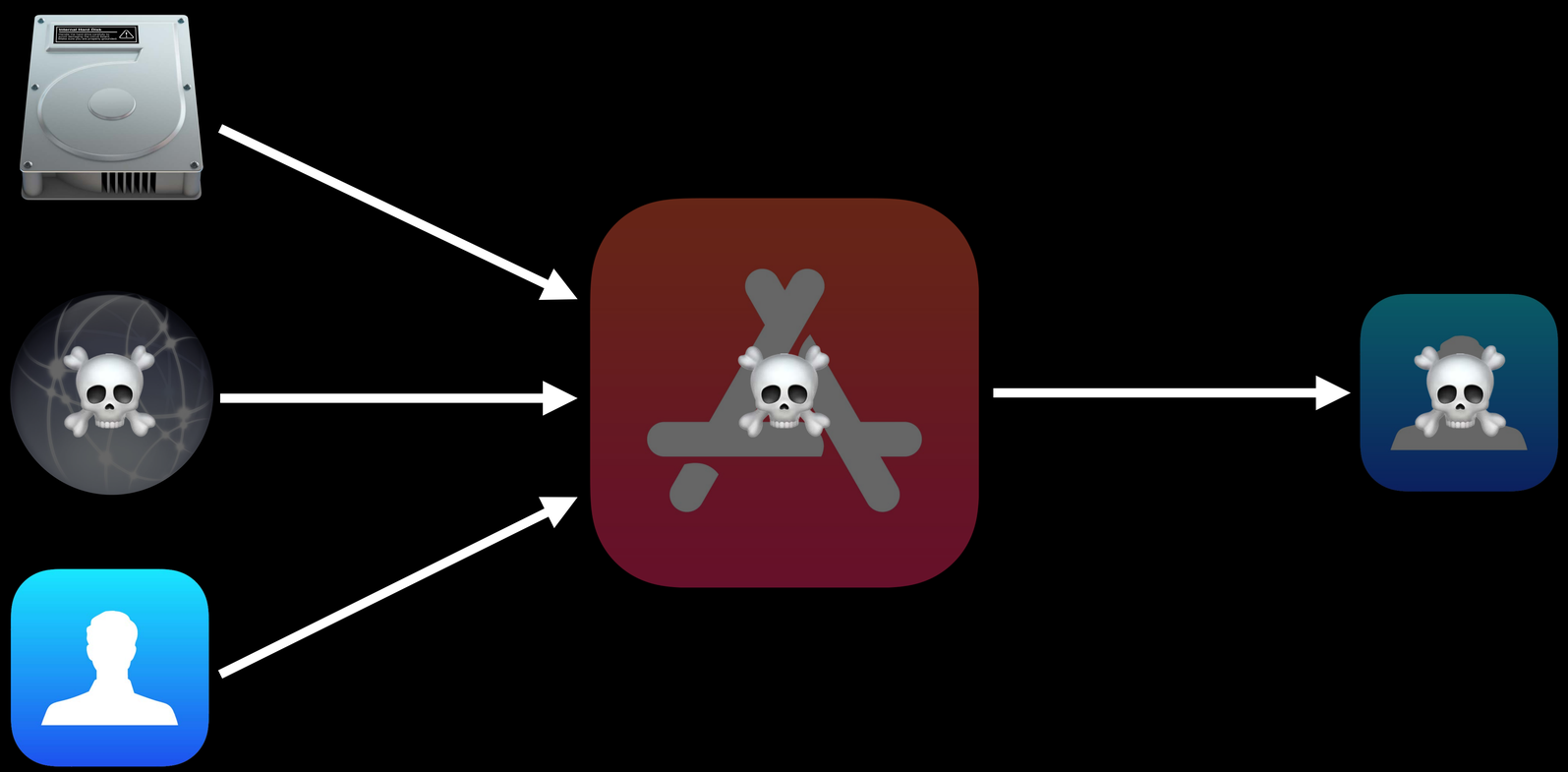
Data You Can Trust

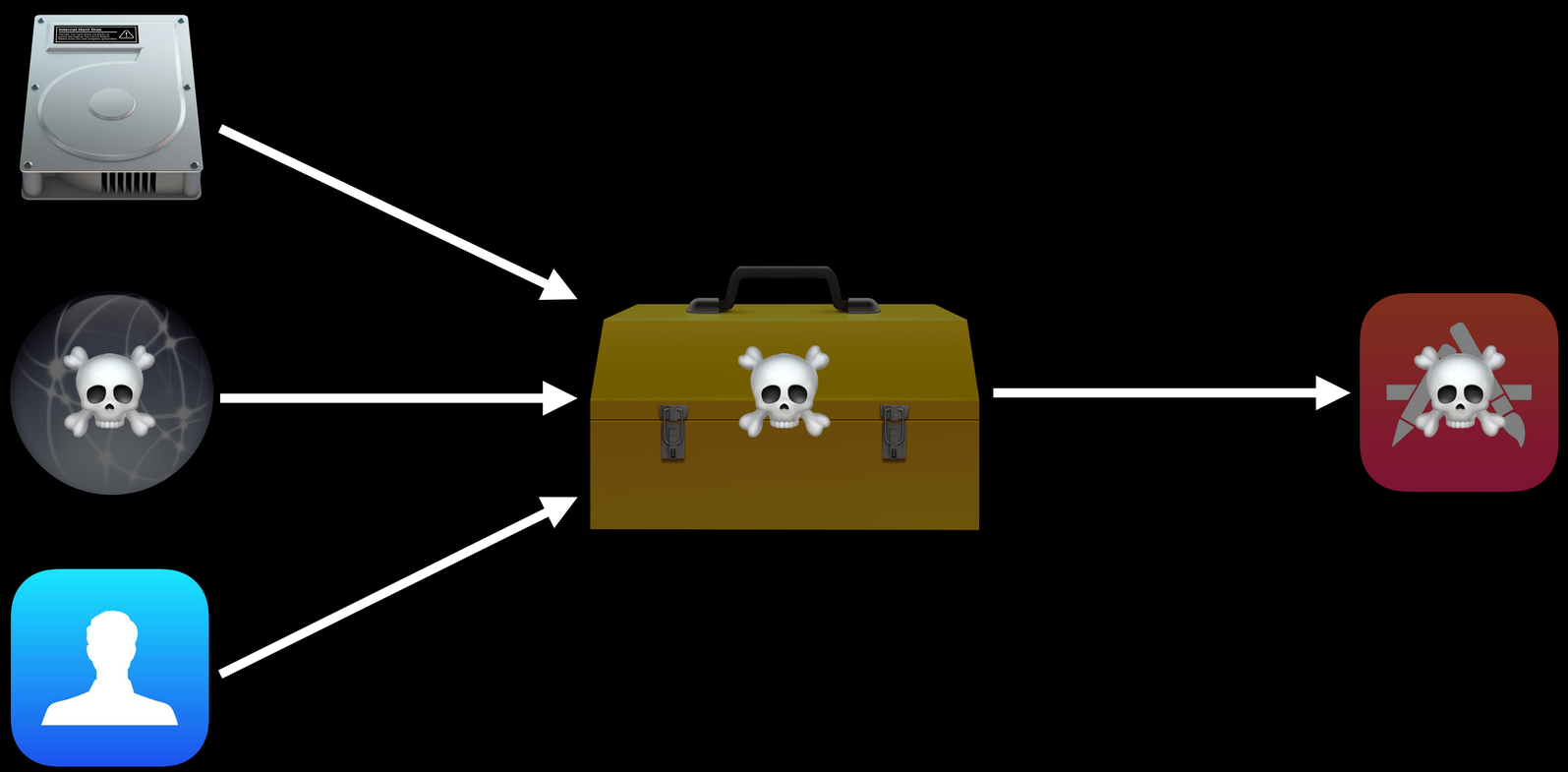
Safer Cocoa Archival and Serialization

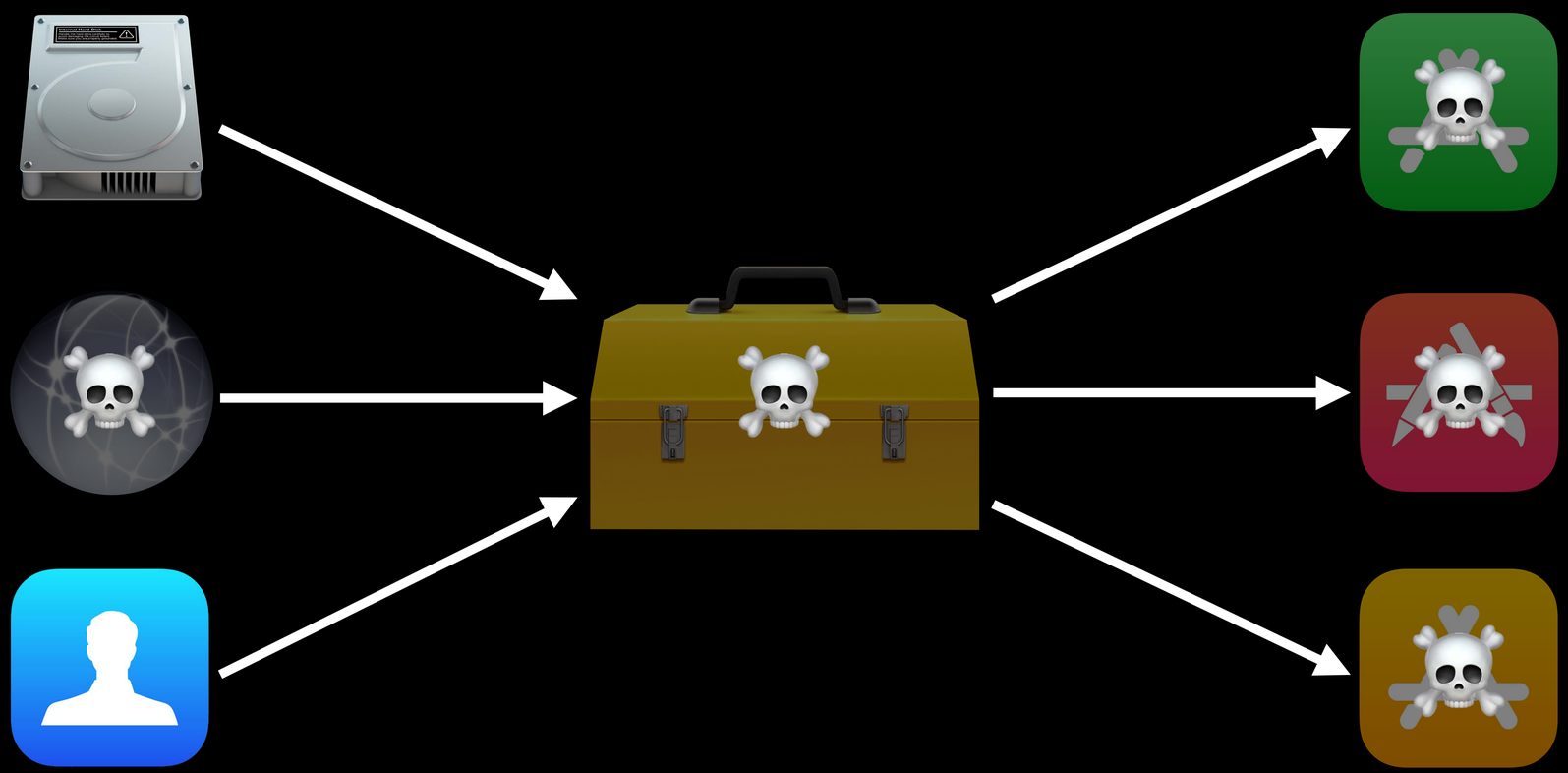
Session 222

Itai Ferber, Foundation









Trusting Data

Goal—ensure it

- Hasn't been modified
- Doesn't contain anything unexpected
- Follows the format/model/structure you need

Validating Raw Data

Validating Primitive Data

Validating Structured Data

`NSSecureCoding`

`Codable`

Modeling Data

Modeling Data

Raw

```
0x7B 0x22 0x67 0x72 0x65 0x65 0x74 0x69 0x6E 0x67 0x22 0x3A 0x22
0x68 0x65 0x6C 0x6C 0x6F 0x20 0x77 0x6F 0x72 0x6C 0x64 0x22 0x7D
```

Modeling Data

```
0x7B 0x22 0x67 0x72 0x65 0x65 0x74 0x69 0x6E 0x67 0x22 0x3A 0x22
0x68 0x65 0x6C 0x6C 0x6F 0x20 0x77 0x6F 0x72 0x6C 0x64 0x22 0x7D
```

Raw

```
0x7B 0x22
```

Modeling Data

Formatted

```
{    "    g    r    e    e    t    i    n    g    "    :    "
h    e    l    l    o         w    o    r    l    d    "    }
```

Raw

```
0x7B 0x22
```

Modeling Data

Formatted

```
{ "greeting": "hello world" }
```

Raw

```
0x7B 0x22
```

Modeling Data

```
{ "greeting": "hello world" }
```

Raw

```
0x7B 0x22
```

Formatted

```
{ ... }
```

Modeling Data

Primitive

```
[ "greeting": "hello world" ]
```

Raw

```
0x7B 0x22
```

Formatted

```
{ ... }
```

Modeling Data

```
[ "greeting": "hello world" ]
```

Raw	Formatted	Primitive
`0x7B 0x22`	`{ ... }`	`[ ... ]`

Modeling Data

Structured

```
Greeting("hello world")
```

Raw

```
0x7B 0x22
```

Formatted

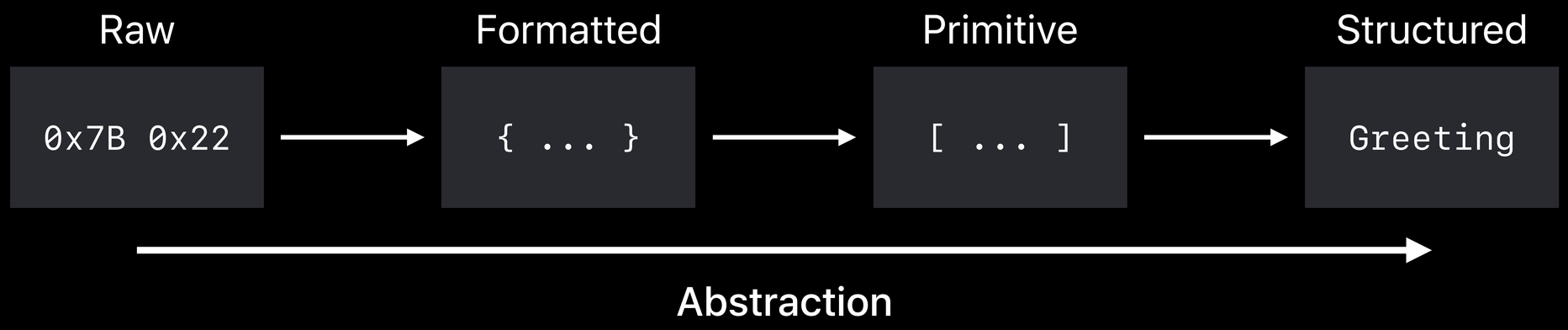
```
{ ... }
```

Primitive

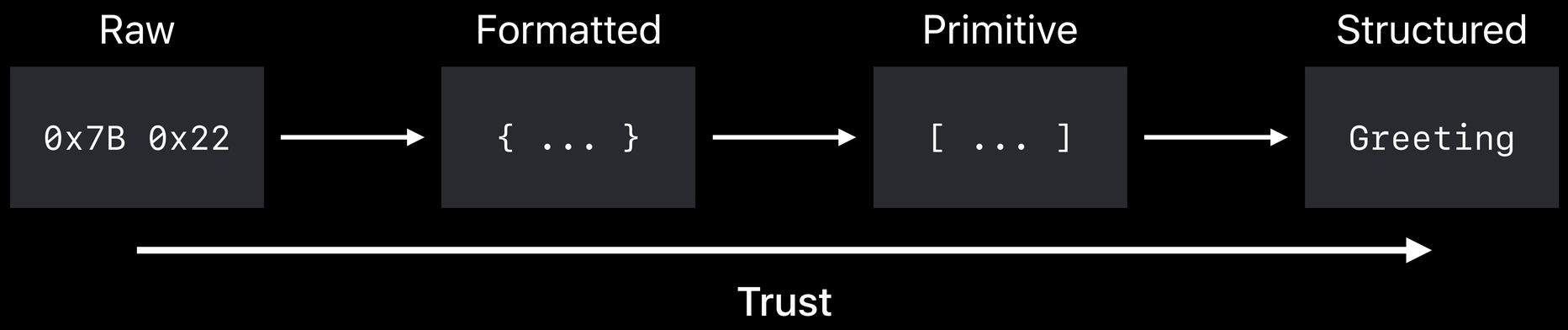
```
[ ... ]
```

Modeling Data

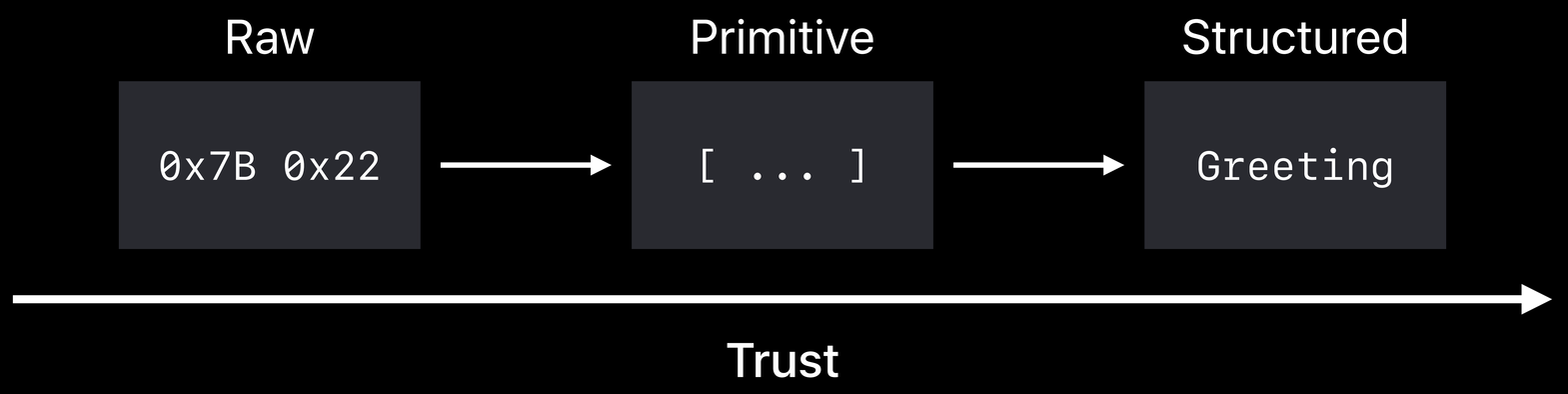
Modeling Data



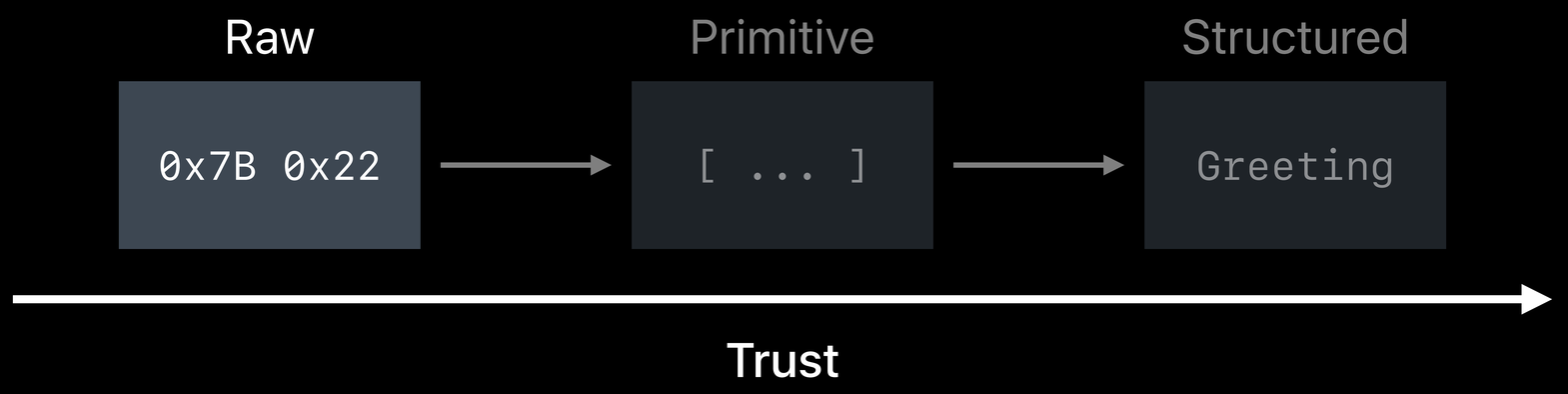
Modeling Data



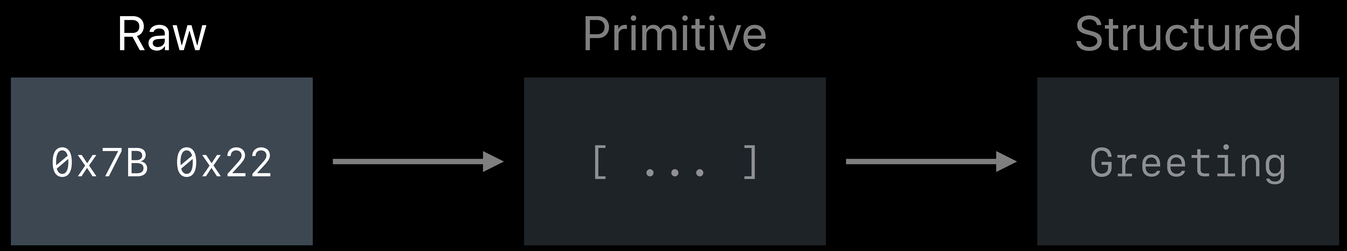
Modeling Data



Raw Data

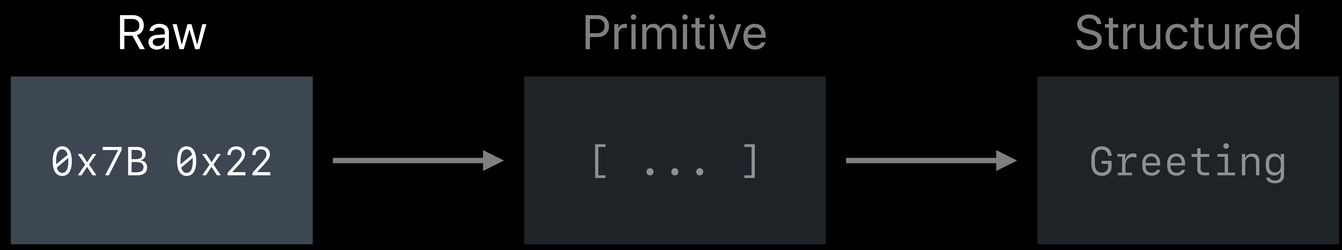


Raw Data



Raw Data

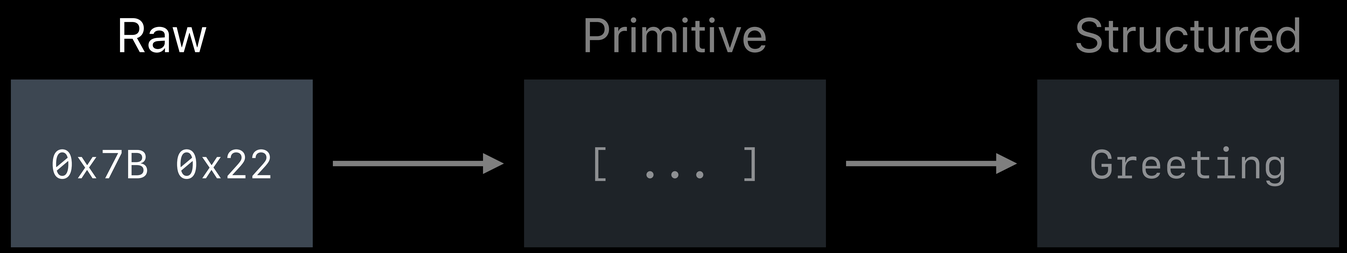
```
.... 0x77 0x72 0x61 0x61 0x20 0x6A 0x76 0x79 0x79 0x20 0x6C 0x62
0x68 0x20 0x7A 0x6E 0x65 0x65 0x6C 0x20 0x7A 0x72 0x3F 0x3F ....
```



Raw Data

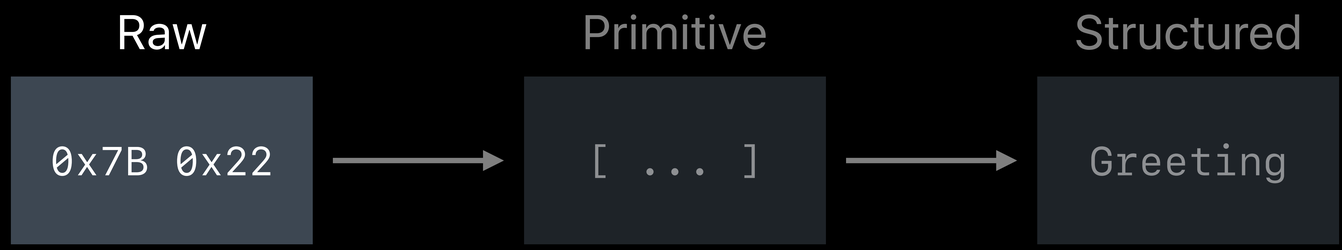
```
.... 0x77 0x72 0x61 0x61 0x20 0x6A 0x76 0x79 0x79 0x20 0x6C 0x62
0x68 0x20 0x7A 0x6E 0x65 0x65 0x6C 0x20 0x7A 0x72 0x3F 0x3F ....
```

Length



Raw Data

```
.... 0x77 0x72 0x61 0x61 0x20 0x6A 0x76 0x79 0x79 0x20 0x6C 0x62
0x68 0x20 0x7A 0x6E 0x65 0x65 0x6C 0x20 0x7A 0x72 0x3F 0x3F ....
```

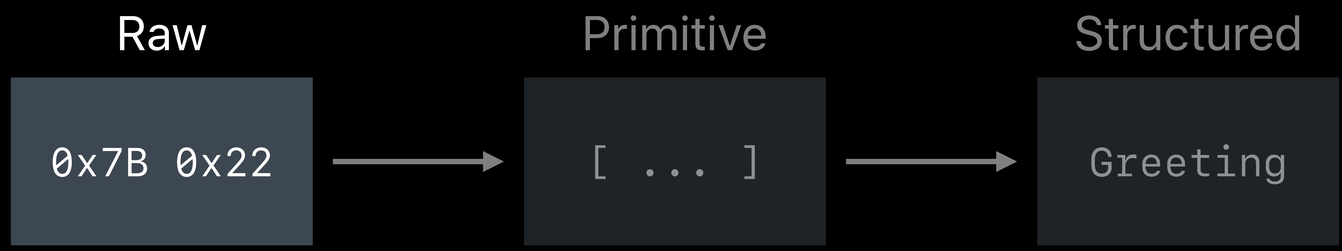


Raw Data

```
93e04478af8814e74900b13d533f5ecd3394adcf7fbdb631c2731095913e8ed3
```

```
.... 0x77 0x72 0x61 0x61 0x20 0x6A 0x76 0x79 0x79 0x20 0x6C 0x62
0x68 0x20 0x7A 0x6E 0x65 0x65 0x6C 0x20 0x7A 0x72 0x3F 0x3F ....
```

Checksum

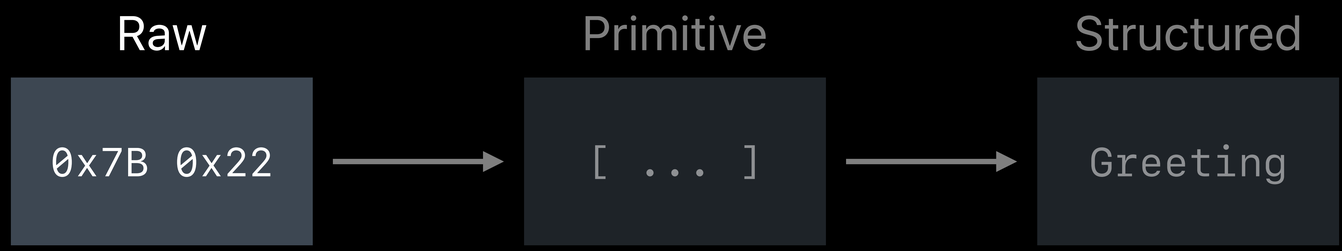


Raw Data

```
93e04478af8814e74900b13d533f5ecd3394adcf7fbdb631c2731095913e8ed3
```

```
.... 0x77 0x72 0x61 0x61 0x20 0x6A 0x76 0x79 0x79 0x20 0x6C 0x62
0x68 0x20 0x7A 0x6E 0x64 0x65 0x6C 0x20 0x7A 0x72 0x3F 0x3F ....
```

Checksum

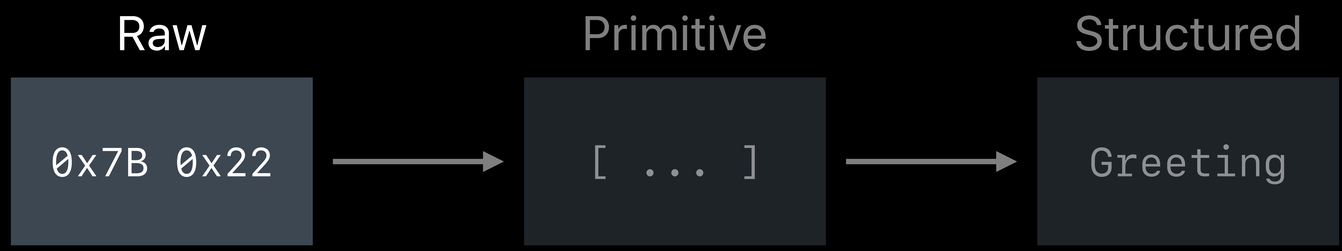


Raw Data

```
1cc62b270da266fdc00387b8b23ae47346010351ce6277e86c7d871c7684c43f
```

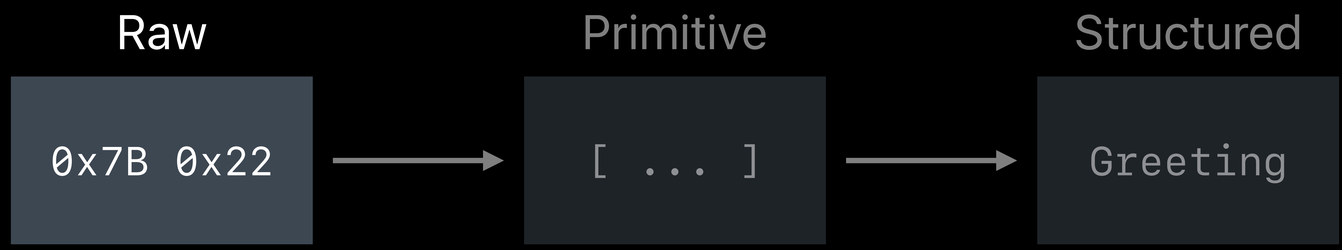
```
.... 0x77 0x72 0x61 0x61 0x20 0x6A 0x76 0x79 0x79 0x20 0x6C 0x62
0x68 0x20 0x7A 0x6E 0x64 0x65 0x6C 0x20 0x7A 0x72 0x3F 0x3F ....
```

Checksum



Raw Data

```
.... 0x77 0x72 0x61 0x61 0x20 0x6A 0x76 0x79 0x79 0x20 0x6C 0x62
0x68 0x20 0x7A 0x6E 0x64 0x65 0x6C 0x20 0x7A 0x72 0x3F 0x3F ....
```



Primitive Data

Primitive Data

```
[ 1 ]
```

```
["a":"b"]
```

```
"hello"
```

```
2.71828
```

```
null
```

Primitive Data

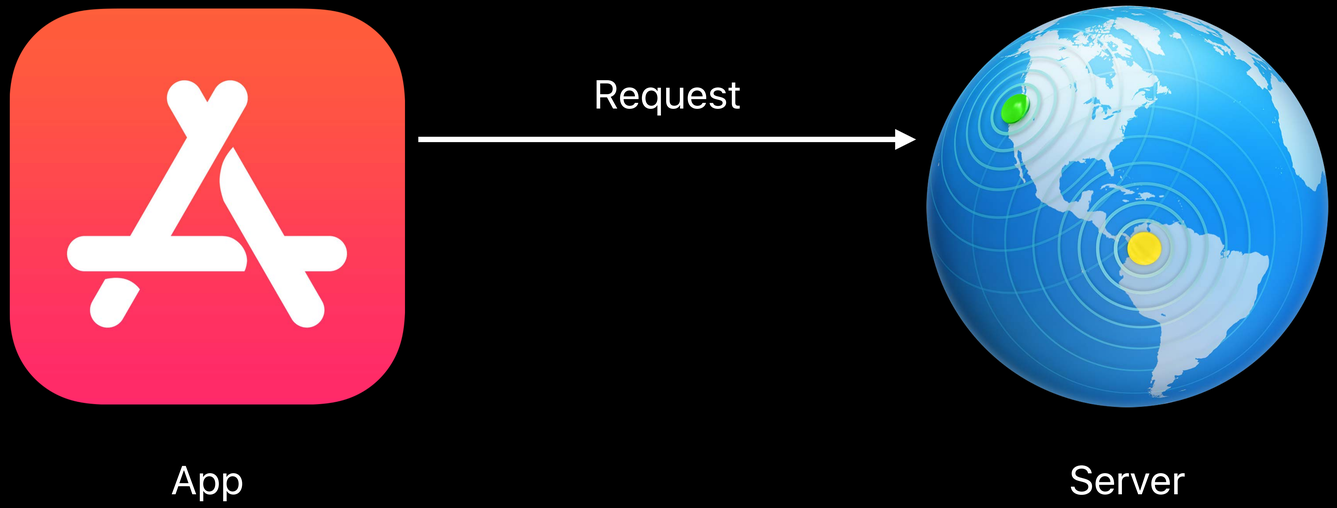
Primitive Data

My App

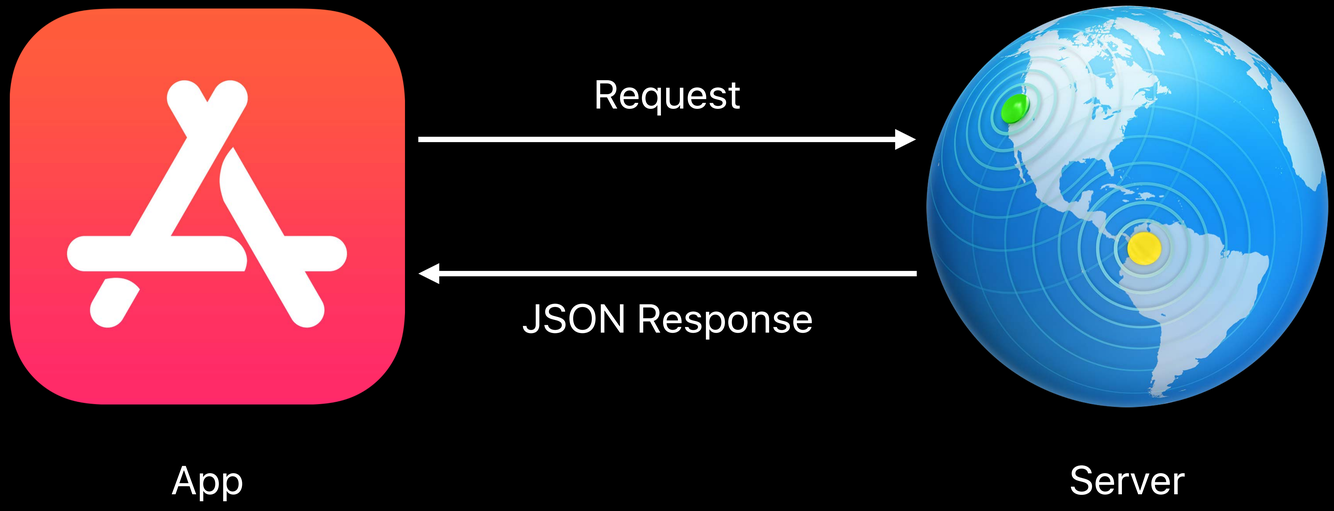
App

Server

My App



My App



Sample API Response


```
[
  {
    "id": 1,
    "name": "Guitar",
    "description": "Still has most of its strings!",
    "price": "41.99",
    "sold": false,
    "url": "https://example.com/products/1",
    "tags": [ "music", "electric" ],
    "created": "2018-05-01T10:39:04 -0700",
    "last_updated": "2018-05-04T16:47:39 -0700",
  },
  ...
]
```

Sample API Response


```
[
  {
    "id": 1,
    "name": "Guitar",
    "description": "Still has most of its strings!",
    "price": "41.99",
    "sold": false,
    "url": "https://example.com/products/1",
    "tags": [ "music", "electric" ],
    "created": "2018-05-01T10:39:04 -0700",
    "last_updated": "2018-05-04T16:47:39 -0700",
  },
  ...
]
```

Sample API Response


```
[
  {
    "id": 1,
    "name": "Guitar",
    "description": "Still has most of its strings!",
    "price": "41.99",
    "sold": false,
    "url": "https://example.com/products/1",
    "tags": [ "music", "electric" ],
    "created": "2018-05-01T10:39:04 -0700",
    "last_updated": "2018-05-04T16:47:39 -0700",
  },
  ...
]
```

Sample API Response


```
[
  {
    "id": 1,
    "name": "Guitar",
    "description": "Still has most of its strings!",
    "price": "41.99",
    "sold": false,
    "url": "https://example.com/products/1",
    "tags": [ "music", "electric" ],
    "created": "2018-05-01T10:39:04 -0700",
    "last_updated": "2018-05-04T16:47:39 -0700",
  },
  ...
]
```

Sample API Response


```
[
  {
    "id": 1,
    "name": "Guitar",
    "description": "Still has most of its strings!",
    "price": "41.99",
    "sold": false,
    "url": "https://example.com/products/1",
    "tags": [ "music", "electric" ],
    "created": "2018-05-01T10:39:04 -0700",
    "last_updated": "2018-05-04T16:47:39 -0700",
  },
  ...
]
```

Sample API Response

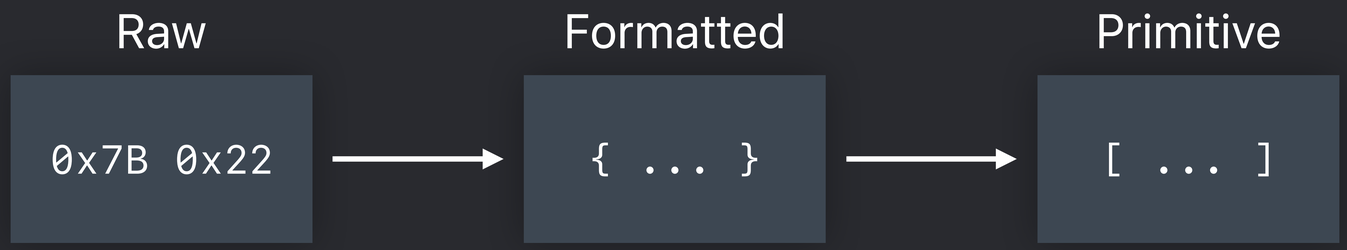

```
[
  {
    "id": 1,
    "name": "Guitar",
    "description": "Still has most of its strings!",
    "price": "41.99",
    "sold": false,
    "url": "https://example.com/products/1",
    "tags": [ "music", "electric" ],
    "created": "2018-05-01T10:39:04 -0700",
    "last_updated": "2018-05-04T16:47:39 -0700",
  },
  ...
]
```

```
// Deserializing API Response

let data: Data = ... /* load & validate */
let json: Any
do {
     json = try JSONSerialization.jsonObject(with: data, options: 0)
} catch { /* handle the error */ }

// `json` contains valid primitive objects.
// Now what?
```

```
// Deserializing API Response

let data: Data = ... /* load & validate */
let json: Any
do {
     json = try JSONSerialization.jsonObject(with: data, options: 0)
} catch { /* handle the error */ }

// `json` contains valid primitive objects.
// Now what?
```

```
// Deserializing API Response

let data: Data = ... /* load & validate */
let json: Any
do {
     json = try JSONSerialization.jsonObject(with: data, options: 0)
} catch { /* handle the error */ }

// `json` contains valid primitive objects.
// Now what?
```

```swift
// Deserializing API Response

let data: Data = ... /* load & validate */
let json: Any
do {
     json = try JSONSerialization.jsonObject(with: data, options: 0)
} catch { /* handle the error */ }

// `json` contains valid primitive objects.
// Now what?
```



```
// Consuming API Response

// `json` is an Any; cast to consume, right?
let listings = json as! [[String : Any]]

let musicListings = listings.filter { listing in
    (listing["tags"] as! [String]).contains("music")
}
```

```
// Consuming API Response

// `json` is an Any; cast to consume, right?
let listings = json as! [[String : Any]]

let musicListings = listings.filter { listing in
    (listing["tags"] as! [String]).contains("music")
}
```

```
// Consuming API Response

// `json` is an Any; cast to consume, right?
let listings = json as! [[String : Any]]

let musicListings = listings.filter { listing in
    (listing["tags"] as! [String]).contains("music")
}
```

```
// Consuming API Response

// `json` is an Any; cast to consume, right?
let listings = json as! [[String : Any]]

let musicListings = listings.filter { listing in
    (listing["tags"] as! [String]).contains("music")
}
```

Type Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Type Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Type Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Type Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music",     42      ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Type Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music",     42      ], as! [String]
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Validate First, Execute Later

```
// Validate First, Execute Later

let listings = json as! [[String : Any]]
```

```
// Validate First, Execute Later

let listings = json as! [[String : Any]]
```

```
// Validate First, Execute Later


guard let listings = json as? [[String : Any]] else {
    throw ...
}
```

```
// Validate First, Execute Later
```

```
guard let listings = json as? [[String : Any]] else {
    throw ...
}
```

```
let musicListings = listings.filter { listing in
    let tags = listing["tags"] as! [String]
    return tags.contains("music")
}
```

```
// Validate First, Execute Later
```

```
guard let listings = json as? [[String : Any]] else {
    throw ...
}
```

```
let musicListings = listings.filter { listing in
    let tags = listing["tags"] as! [String]
    return tags.contains("music")
}
```

```
// Validate First, Execute Later


guard let listings = json as? [[String : Any]] else {
    throw ...
}


let musicListings = listings.filter { listing in
    let tags = listing["tags"] as? [String] ?? []
    return tags.contains("music")
}
```

Nullability Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Nullability Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music",    null    ], as! [String]
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Range Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Range Validation

```
{
  "id": -1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Range Validation

```
{
  "id": 18446744073709551615,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Length Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Length Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Length Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Two houses, both alike in d...",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Domain-Specific Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Domain-Specific Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "Still has most of its strings!",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Domain-Specific Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "http://evilphishingdomain.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Inter-Value Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

Inter-Value Validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "1969-12-31T16:00:00 -0800",
}
```

```
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
```

```
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
    // Type, nullability, range checking
    guard let id = listing["id"] as? Int,
        id > 0, id <= Int32.max else { throw ... }
```

```
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
    // Type, nullability, range checking
    guard let id = listing["id"] as? Int,
        id > 0, id <= Int32.max else { throw ... }
```

```
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
    // Type, nullability, range checking
    guard let id = listing["id"] as? Int,
        id > 0, id <= Int32.max else { throw ... }
```

```swift
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
    // Type, nullability, range checking
    guard let id = listing["id"] as? Int,
        id > 0, id <= Int32.max else { throw ... }

    // Type, nullability, length checking, and domain validation (via URL)
    guard let urlString = listing["url"] as? String,
        urlString.count <= 50,
        let url = URL(string: urlString) else { throw ... }
```

```swift
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
    // Type, nullability, range checking
    guard let id = listing["id"] as? Int,
        id > 0, id <= Int32.max else { throw ... }

    // Type, nullability, length checking, and domain validation (via URL)
    guard let urlString = listing["url"] as? String,
        urlString.count <= 50,
        let url = URL(string: urlString) else { throw ... }
```

```swift
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
    // Type, nullability, range checking
    guard let id = listing["id"] as? Int,
        id > 0, id <= Int32.max else { throw ... }

    // Type, nullability, length checking, and domain validation (via URL)
    guard let urlString = listing["url"] as? String,
        urlString.count <= 50,
        let url = URL(string: urlString) else { throw ... }
```

```swift
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
    // Type, nullability, range checking
    guard let id = listing["id"] as? Int,
        id > 0, id <= Int32.max else { throw ... }

    // Type, nullability, length checking, and domain validation (via URL)
    guard let urlString = listing["url"] as? String,
        urlString.count <= 50,
        let url = URL(string: urlString) else { throw ... }
```

```swift
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
    // Type, nullability, range checking
    guard let id = listing["id"] as? Int,
        id > 0, id <= Int32.max else { throw ... }

    // Type, nullability, length checking, and domain validation (via URL)
    guard let urlString = listing["url"] as? String,
        urlString.count <= 50,
        let url = URL(string: urlString) else { throw ... }

    // Further domain validation
    guard url.host == "example.com" else { throw ... }
    // ...
}
```

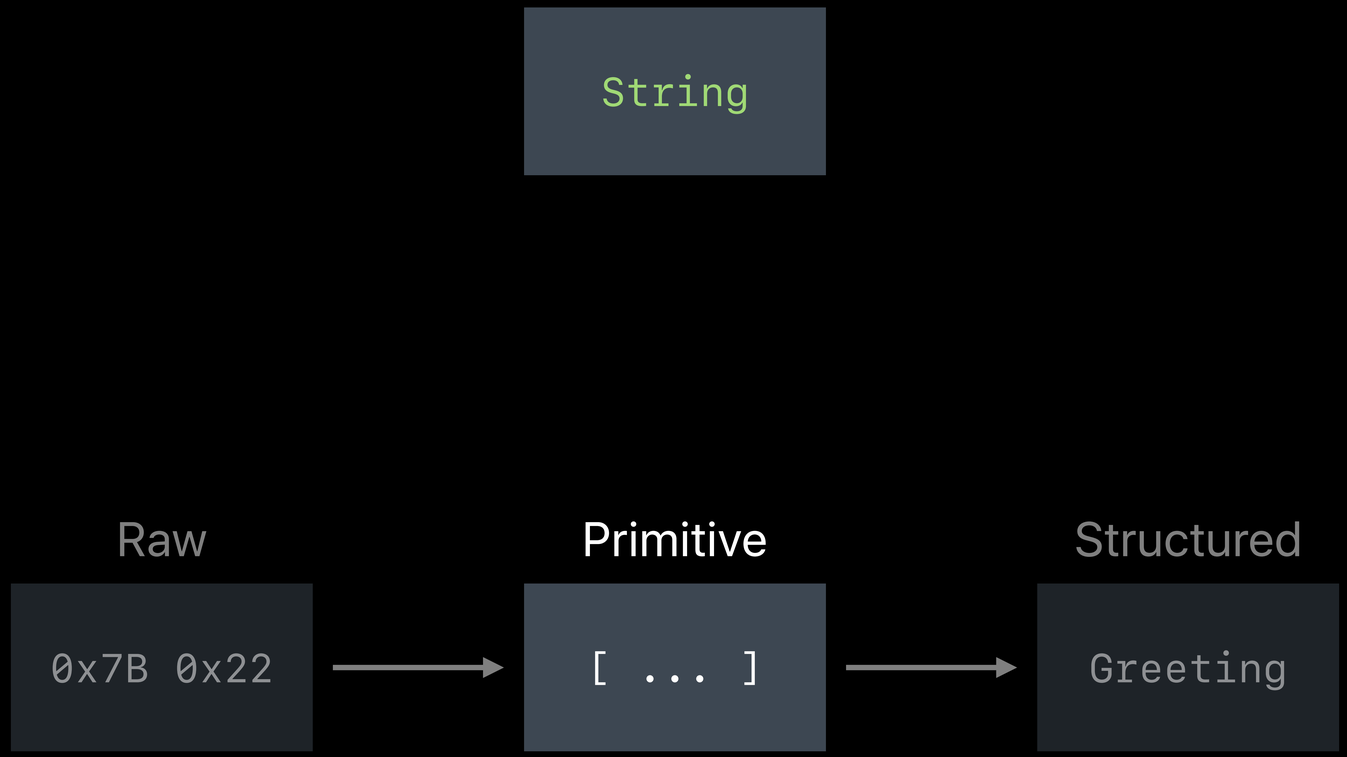
```swift
// Validating a Listing

func validate(_ listing: [String : Any]) throws {
    // Type, nullability, range checking
    guard let id = listing["id"] as? Int,
        id > 0, id <= Int32.max else { throw ... }

    // Type, nullability, length checking, and domain validation (via URL)
    guard let urlString = listing["url"] as? String,
        urlString.count <= 50,
        let url = URL(string: urlString) else { throw ... }

    // Further domain validation
    guard url.host == "example.com" else { throw ... }
    // ...
}
try listings.forEach(validate)
```

Structured Data

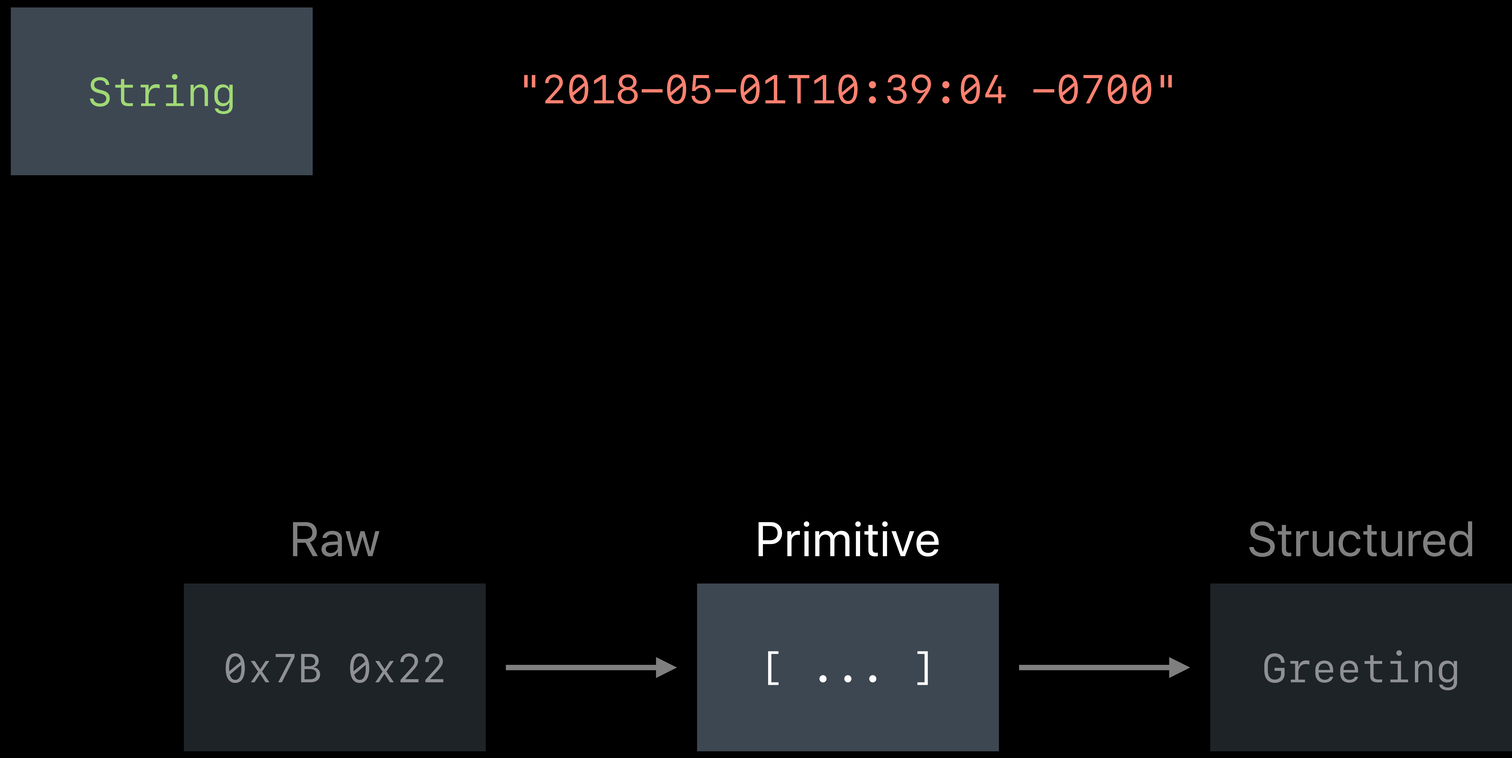
Structured Data

```
"Guitar"
```

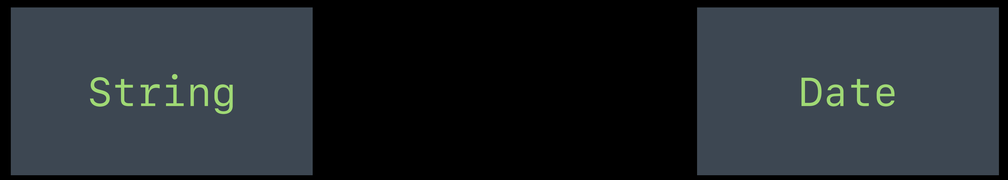
Structured Data



Structured Data



Structured Data



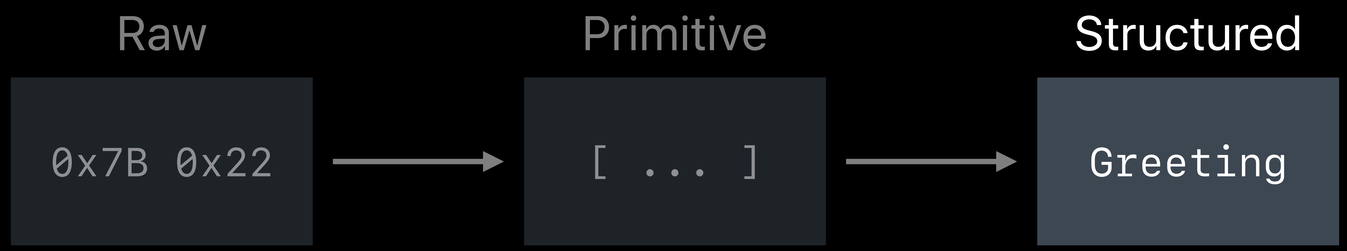
Structured Data

Structured Data

Structured Data

Structured Data

Structured Data



```
// Structured Model Types

class Purchase : NSCoding {
    let listing: Listing
    let purchaseDate: Date
    let receipt: Receipt

    func encode(with coder: NSCoder) { ... }
    required init?(coder: NSCoder) { ... }
}
```

```
// Structured Model Types

class Purchase : NSCoding {
    required init?(coder: NSCoder) {
```

```
// Structured Model Types

class Purchase : NSCoding {
    required init?(coder: NSCoder) {
        guard let listing = coder.decodeObject(forKey: "listing") as? Listing else {
            return nil
        }
        self.listing = listing
```

```
// Structured Model Types

class Purchase : NSCoding {
    required init?(coder: NSCoder) {
        guard let listing = coder.decodeObject(forKey: "listing") as? Listing else {
            return nil
        }
        self.listing = listing
```

```
// Structured Model Types

class Purchase : NSCoding {
    required init?(coder: NSCoder) {
        guard let listing = coder.decodeObject(forKey: "listing") as? Listing else {
            return nil
        }
        self.listing = listing
```

```
// Structured Model Types

class Purchase : NSCoding {
    required init?(coder: NSCoder) {
        guard let listing = coder.decodeObject(forKey: "listing") as? Listing else {
            return nil
        }
        self.listing = listing

        guard let date = coder.decodeObject(forKey: "purchaseDate") as? Date else {
            return nil
        }
        self.purchaseDate = date
```

```
// Structured Model Types

class Purchase : NSCoding {
    required init?(coder: NSCoder) {
        guard let listing = coder.decodeObject(forKey: "listing") as? Listing else {
            return nil
        }
        self.listing = listing

        guard let date = coder.decodeObject(forKey: "purchaseDate") as? Date else {
            return nil
        }
        self.purchaseDate = date
```

```
// Structured Model Types

class Purchase : NSCoding {
    required init?(coder: NSCoder) {
        guard let listing = coder.decodeObject(forKey: "listing") as? Listing else {
            return nil
        }
        self.listing = listing

        guard let date = coder.decodeObject(forKey: "purchaseDate") as? Date else {
            return nil
        }
        self.purchaseDate = date

        // ...
    }
}
```

```
// Model Storage

func savePurchase(_ purchase: Purchase) throws {
    let archive = NSKeyedArchiver.archivedData(withRootObject: purchase)
    try archive.write(to: /* storage location */)
}
```

```
// Model Storage

func savePurchase(_ purchase: Purchase) throws {
    let archive = NSKeyedArchiver.archivedData(withRootObject: purchase)
    try archive.write(to: /* storage location */)
}
```

```
// Model Storage

func savePurchase(_ purchase: Purchase) throws {
    let archive = NSKeyedArchiver.archivedData(withRootObject: purchase)
    try archive.write(to: /* storage location */)
}
```

```
// Model Storage

func savePurchase(_ purchase: Purchase) throws {
    let archive = NSKeyedArchiver.archivedData(withRootObject: purchase)
    try archive.write(to: /* storage location */)
}

func loadPurchases() throws -> [Purchase] {
    let storage: [Data] = ... /* load from storage location */
    return try storage.map { data in
        guard let object = try NSKeyedUnarchiver.unarchiveTopLevelObject(with: data),
            let purchase = object as? Purchase else { throw ... }
        return purchase
    }
}
```

```
// Model Storage

func savePurchase(_ purchase: Purchase) throws {
    let archive = NSKeyedArchiver.archivedData(withRootObject: purchase)
    try archive.write(to: /* storage location */)
}

func loadPurchases() throws -> [Purchase] {
    let storage: [Data] = ... /* load from storage location */
    return try storage.map { data in
        guard let object = try NSKeyedUnarchiver.unarchiveTopLevelObject(with: data),
            let purchase = object as? Purchase else { throw ... }
        return purchase
    }
}
```

```
// Model Storage

func savePurchase(_ purchase: Purchase) throws {
    let archive = NSKeyedArchiver.archivedData(withRootObject: purchase)
    try archive.write(to: /* storage location */)
}

func loadPurchases() throws -> [Purchase] {
    let storage: [Data] = ... /* load from storage location */
    return try storage.map { data in
        guard let object = try NSKeyedUnarchiver.unarchiveTopLevelObject(with: data),
            let purchase = object as? Purchase else { throw ... }
        return purchase
    }
}
```

```
// Model Storage

func savePurchase(_ purchase: Purchase) throws {
    let archive = NSKeyedArchiver.archivedData(withRootObject: purchase)
    try archive.write(to: /* storage location */)
}

func loadPurchases() throws -> [Purchase] {
    let storage: [Data] = ... /* load from storage location */
    return try storage.map { data in
        guard let object = try NSKeyedUnarchiver.unarchiveTopLevelObject(with: data),
            let purchase = object as? Purchase else { throw ... }
        return purchase
    }
}
```

Model Representations in Keyed Archives

```
{
    "$class": "Purchase",
    "listing": { ... },
    "receipt": { ... },
    "purchaseDate": "...",
}
```

Model Representations in Keyed Archives

```
{
    "$class": "Purchase",
    "listing": { ... },
    "receipt": { ... },
    "purchaseDate": "...",
}
```

Model Representations in Keyed Archives

```
{
    "$class": "Purchase",
    "listing": { ... },
    "receipt": { ... },
    "purchaseDate": "...",
}
```

Object Decoding

```
{
    "$class": "Purchase",
    "listing": { ... },
    "receipt": { ... },
    "purchaseDate": "...",
}
```

Object Decoding

```
try NSKeyedUnarchiver.unarchiveTopLevelObject(with: data)
```

```
{
    "$class": "Purchase",
    "listing": { ... },
    "receipt": { ... },
    "purchaseDate": "...",
}
```

Object Decoding

```
unarchiver.decodeObject(forKey: NSKeyedArchiveRootObjectKey)
```

```
{
    "$class": "Purchase",
    "listing": { ... },
    "receipt": { ... },
    "purchaseDate": "...",
}
```

Object Decoding

```
unarchiver.decodeObject(forKey: NSKeyedArchiveRootObjectKey)
```

```
{
    "$class": "Purchase",
    "listing": { ... },
    "receipt": { ... },
    "purchaseDate": "...",
}
```

Object Decoding

```
unarchiver.decodeObject(forKey: NSKeyedArchiveRootObjectKey)
```

```
"Purchase"
```

Object Decoding

```
unarchiver.decodeObject(forKey: NSKeyedArchiveRootObjectKey)
```

```
Purchase
```

Object Decoding

```
unarchiver.decodeObject(forKey: NSKeyedArchiveRootObjectKey)
```

```
Purchase.alloc()
```

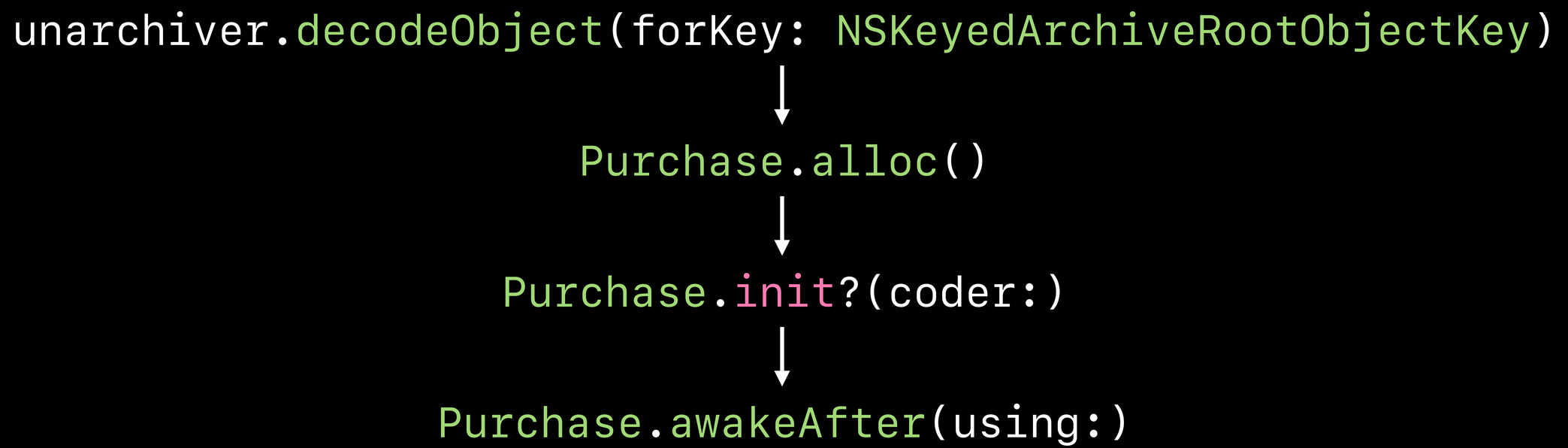
Object Decoding

```
unarchiver.decodeObject(forKey: NSKeyedArchiveRootObjectKey)
```

```
Purchase.alloc()
```

```
Purchase.init?(coder:)
```

Object Decoding



Object Decoding

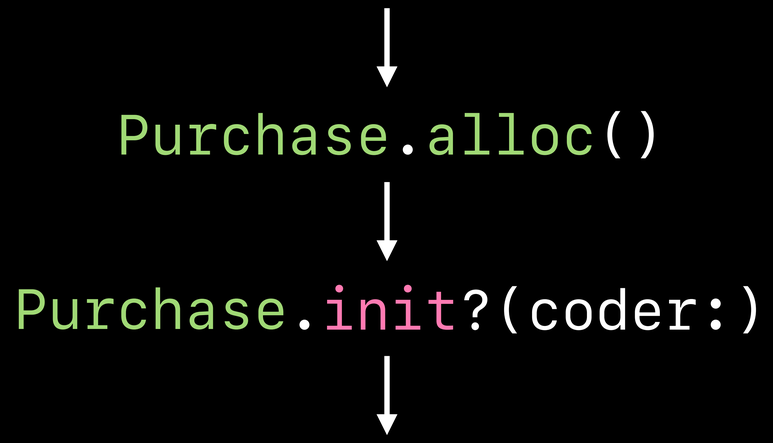
```
{
    "$class": "Purchase",
    "listing": { ... },
    "receipt": { ... },
    "purchaseDate": "...",
}
```

Object Decoding

```
{
    "$class": "UnexpectedClass",
    "listing": { ... },
    "receipt": { ... },
    "purchaseDate": "...",
}
```

Object Decoding

```
unarchiver.decodeObject(forKey: NSKeyedArchiveRootObjectKey)
```



```
Purchase.alloc()
```

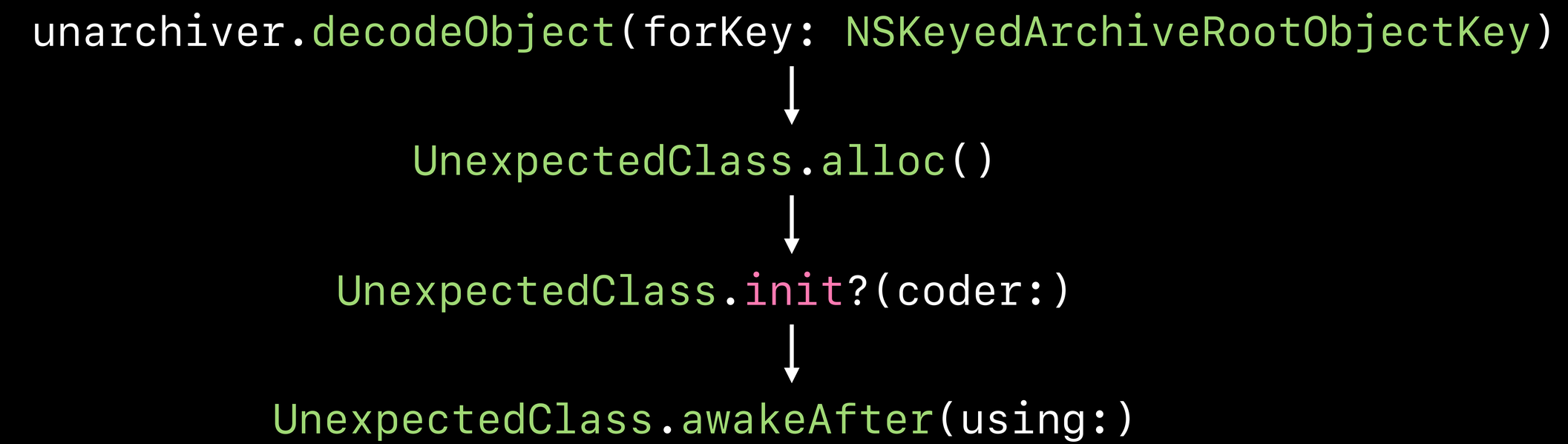


```
Purchase.init?(coder:)
```



```
Purchase.awakeAfter(using:)
```

Object Decoding



```
// Model Storage

func savePurchase(_ purchase: Purchase) throws {
    let archive = NSKeyedArchiver.archivedData(withRootObject: purchase)
    try archive.write(to: /* storage location */)
}

func loadPurchases() throws -> [Purchase] {
    let storage: [Data] = ... /* load from storage location */
    return try storage.map { data in
        guard let object = try NSKeyedUnarchiver.unarchiveTopLevelObject(with: data),
            let purchase = object as? Purchase else { throw ... }
        return purchase
    }
}
```

NSSecureCoding

NSSecureCoding Goals

Prevent arbitrary code execution

Establish expectations in code

Provide explicit way to validate types

Object Decoding with NSSecureCoding

```
NSCoder.decodeObject(forKey:)
```

Object Decoding with NSSecureCoding

```
NSCoder.decodeObject(forKey:)
```

```
NSCoder.decodeObject(of:forKey:)
NSCoder.decodeObject(ofClasses:forKey:)
```

Object Decoding with NSSecureCoding

```
NSCoder.decodeObject(forKey:)
```

Object Decoding with NSSecureCoding

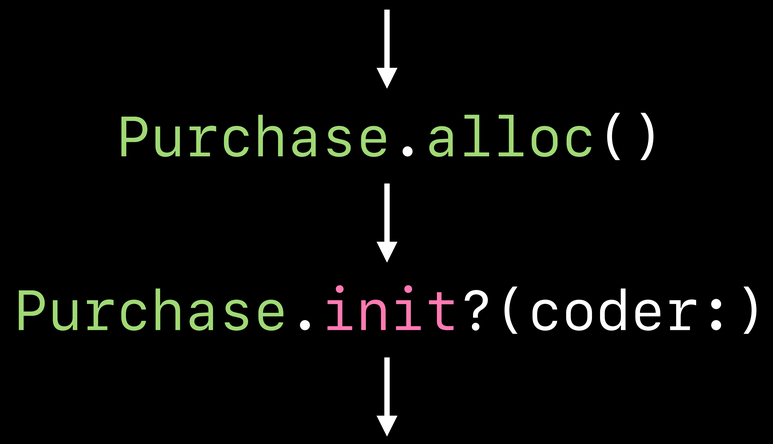
```
unarchiver.decodeObject(forKey: NSKeyedArchiveRootObjectKey)
```

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```

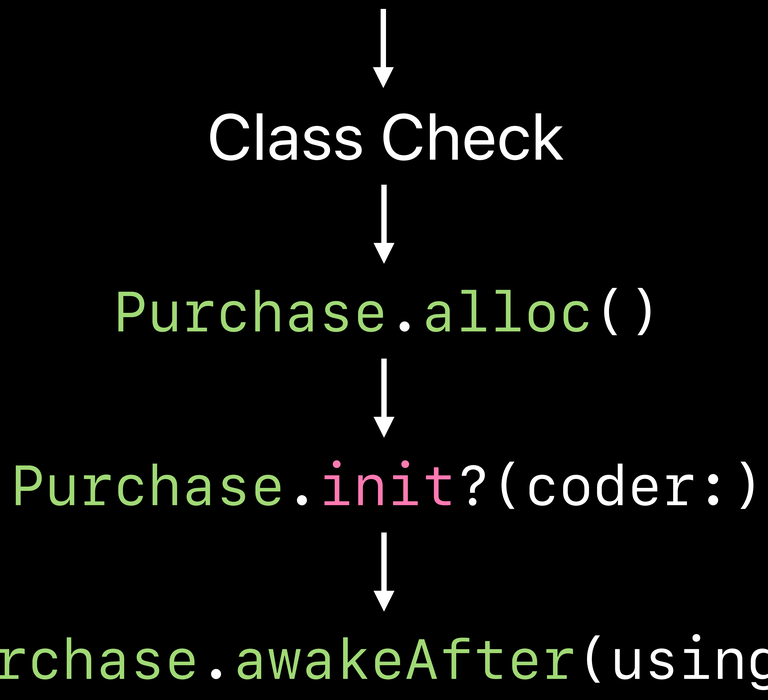
Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```



Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```



Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```

Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```

Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```

Purchase

Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```

Purchase

Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```

UnexpectedClass

Purchase

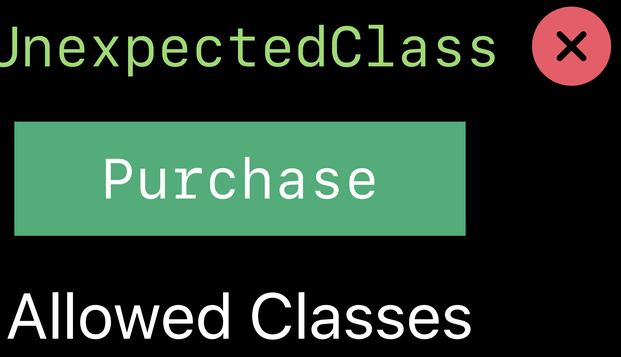
Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```



Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```

Purchase

Allowed Classes

Object Decoding with NSSecureCoding

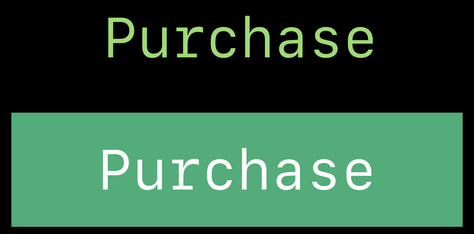
```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```

```
Purchase
```

Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Purchase.self,
                        forKey: NSKeyedArchiveRootObjectKey)
```



Object Decoding with NSSecureCoding

Conforms to `NSSecureCoding`

`supportsSecureCoding` returns true

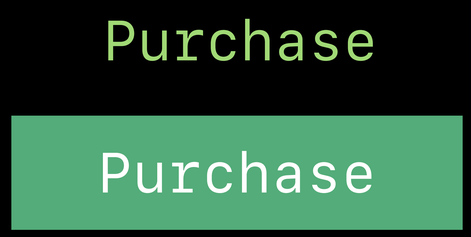
Overrides or inherits both `supportsSecureCoding` and `init?(coder:)`

Object Decoding with NSSecureCoding

Conforms to `NSSecureCoding`

`supportsSecureCoding` returns true

Overrides or inherits both `supportsSecureCoding` and `init?(coder:)`

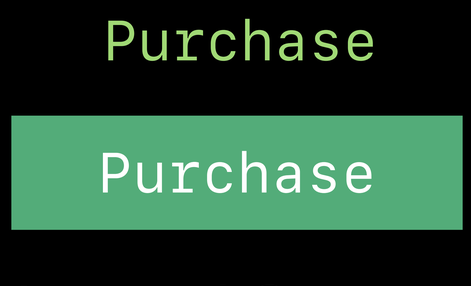


Object Decoding with NSSecureCoding

Conforms to `NSSecureCoding`

`supportsSecureCoding` returns true

Overrides or inherits both `supportsSecureCoding` and `init?(coder:)`



Object Decoding with NSSecureCoding

Conforms to `NSSecureCoding`

`supportsSecureCoding` returns true

Overrides or inherits both `supportsSecureCoding` and `init?(coder:)`

Allowed Classes

Object Decoding with NSSecureCoding

Conforms to `NSSecureCoding`

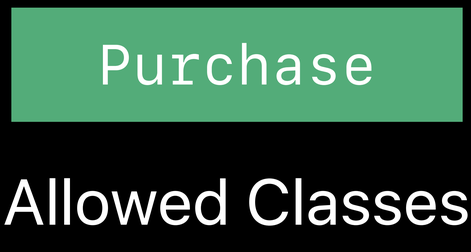
`supportsSecureCoding` returns true

Overrides or inherits both `supportsSecureCoding` and `init?(coder:)`

`Purchase`

Allowed Classes

Object Decoding with NSSecureCoding



Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

Purchase

Allowed Classes

Object Decoding with NSSecureCoding

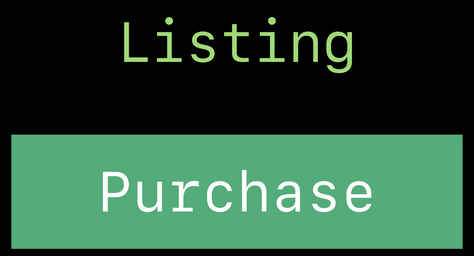
```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

Purchase

Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```



Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

```
Listing
```

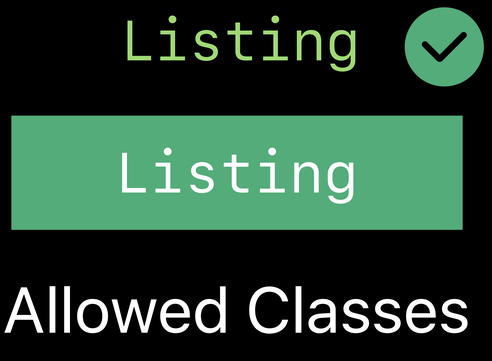
Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```



Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

UnexpectedClass

Allowed Classes

Object Decoding with NSSecureCoding

```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

Decoding Failures

Secure coding violation

Type mismatch

Archive corruption

Decoding Failures

`.decodingFailurePolicy` determines what happens on error

```
enum NSCoder.NSDecodingFailurePolicy {
    case raiseException
    case setErrorAndReturn
}
```

`.raiseException` is the current default

Decoding Failures

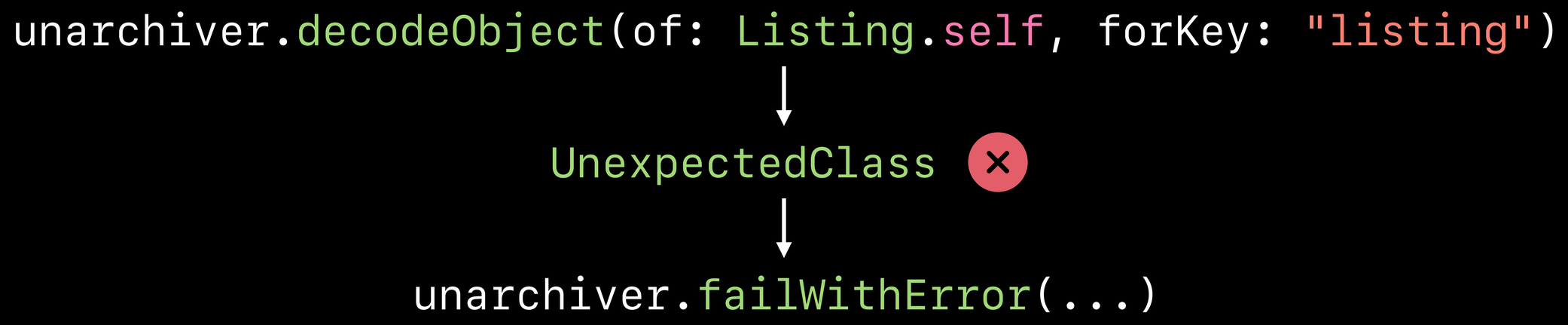
```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

Decoding Failures

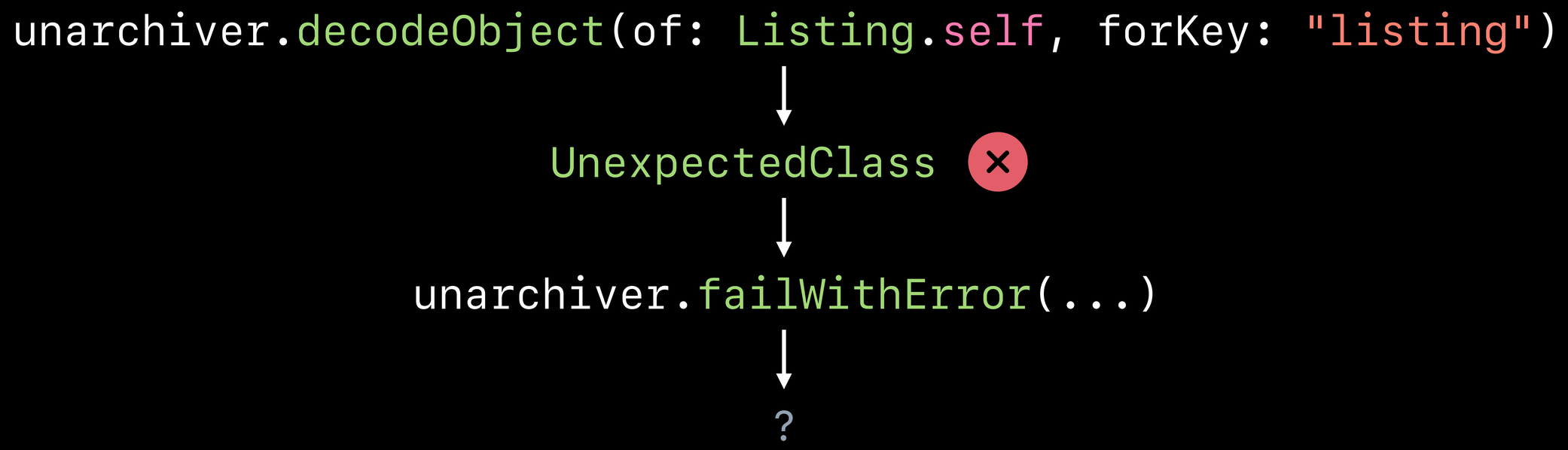
```
unarchiver.decodeObject(of: Listing.self, forKey: "listing")
```

UnexpectedClass

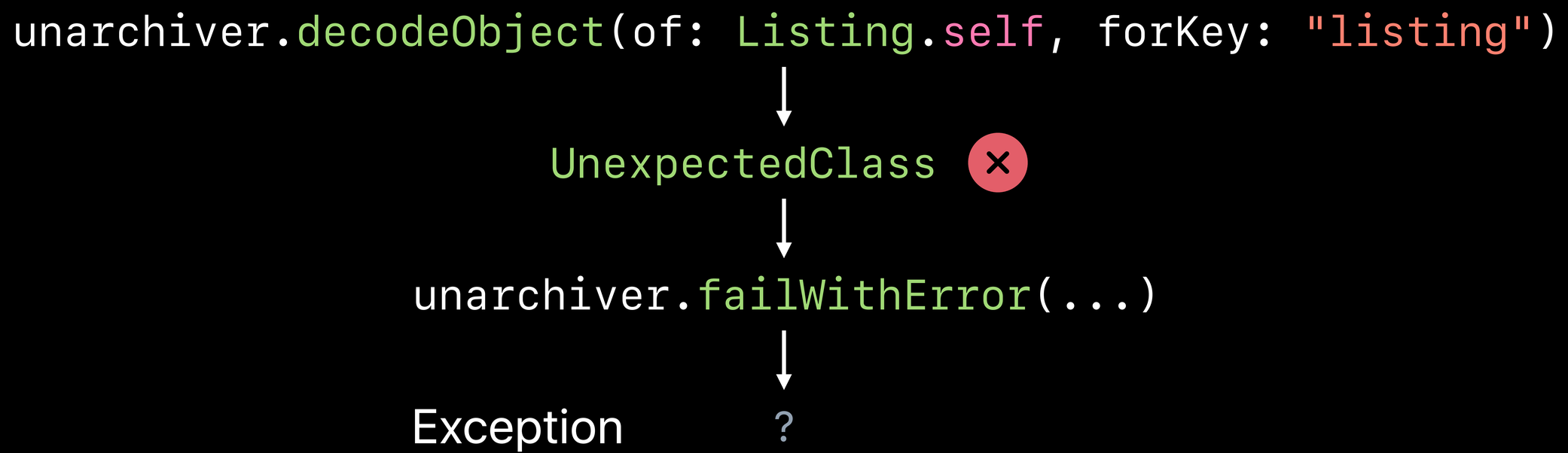
Decoding Failures



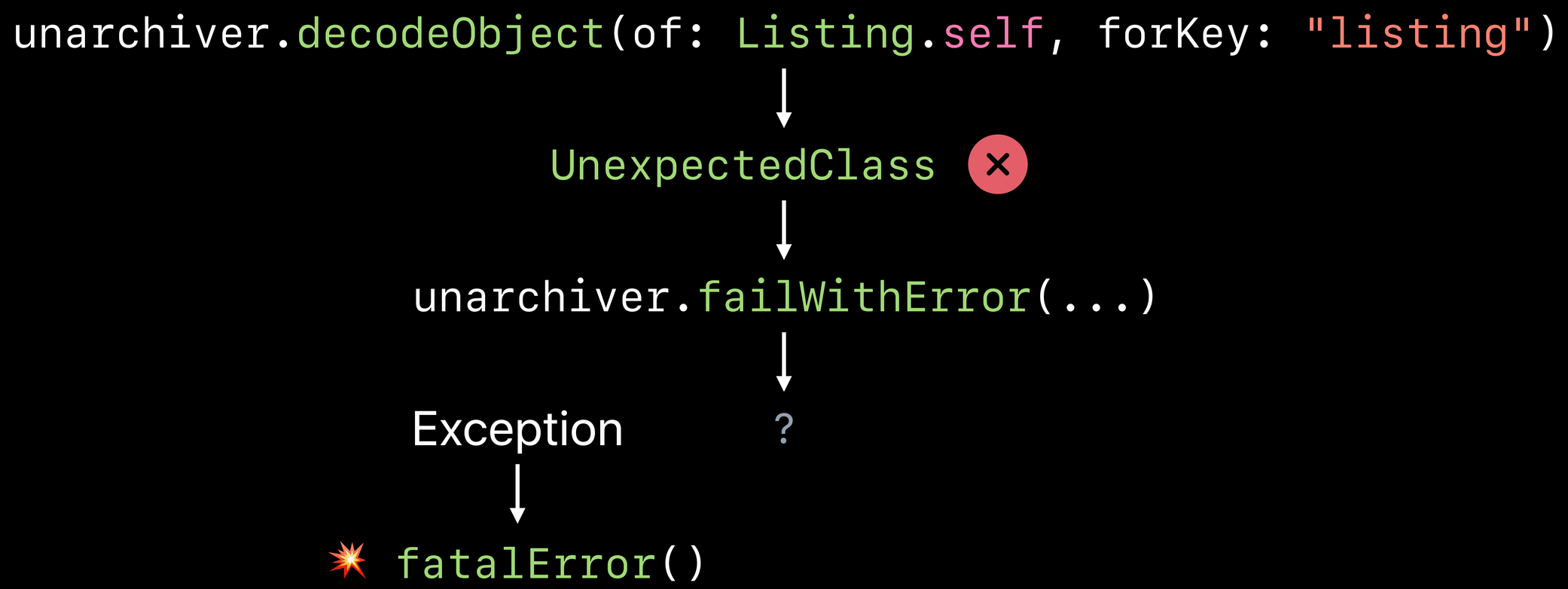
Decoding Failures



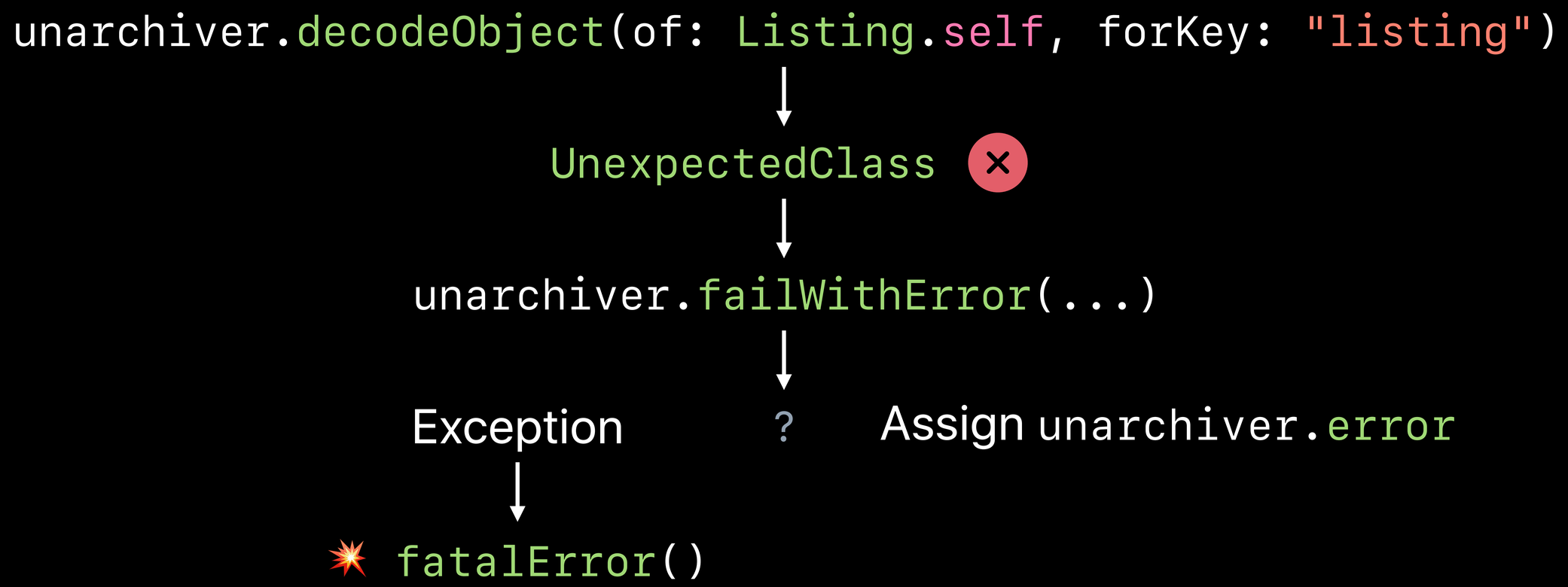
Decoding Failures



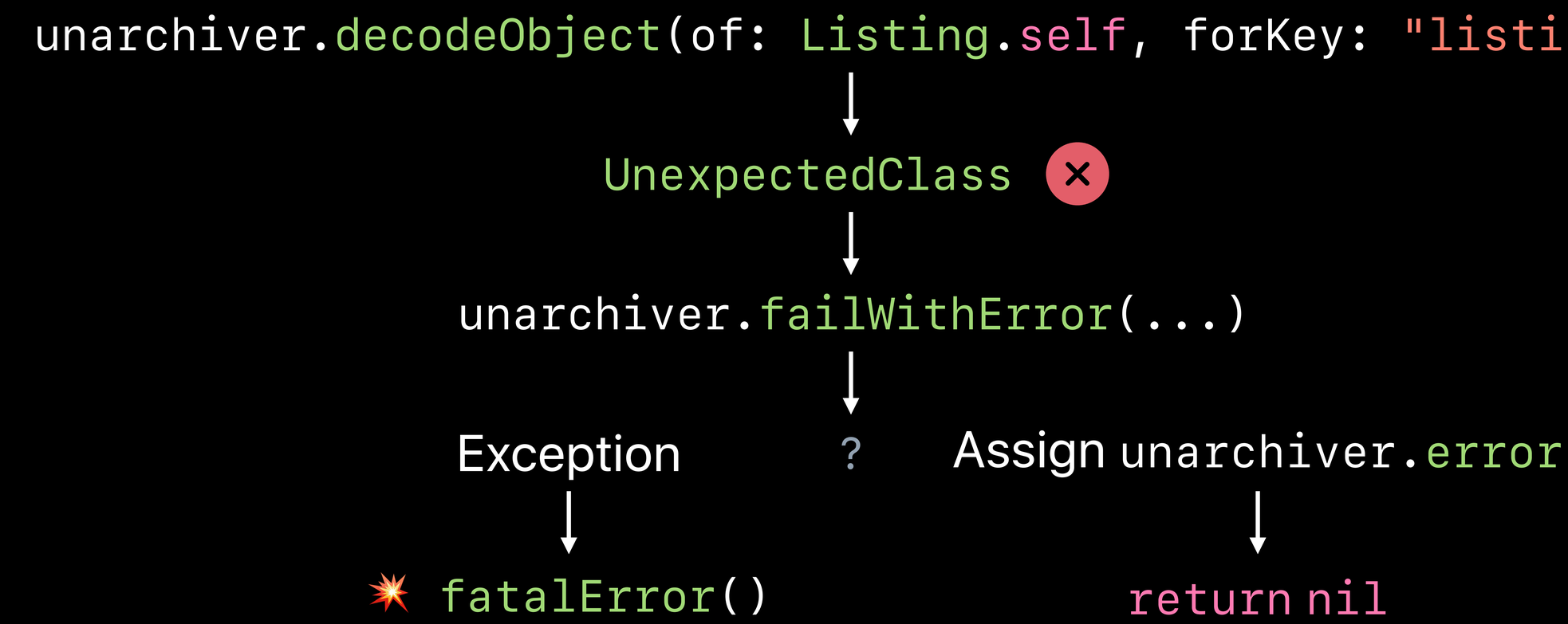
Decoding Failures



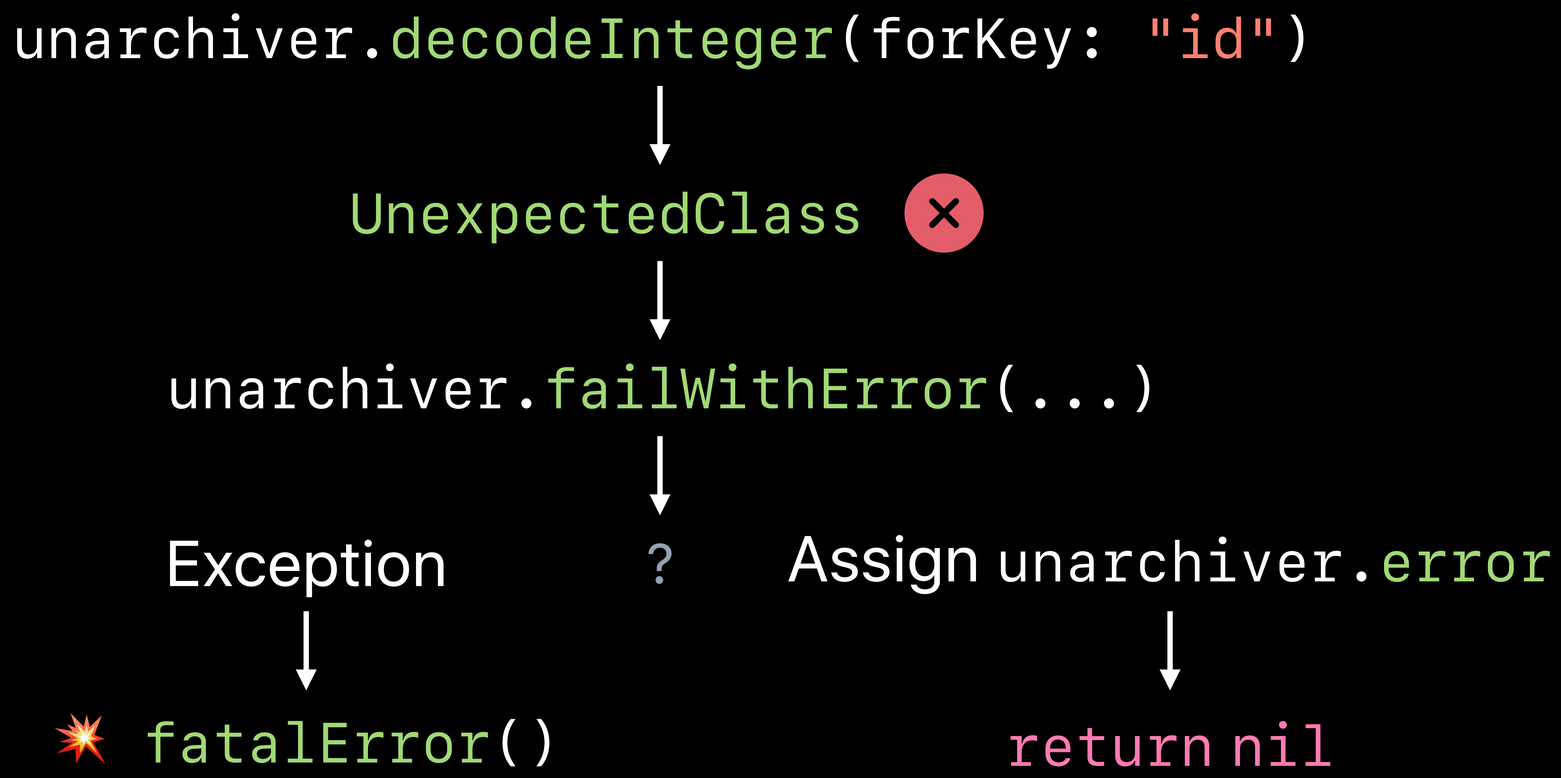
Decoding Failures



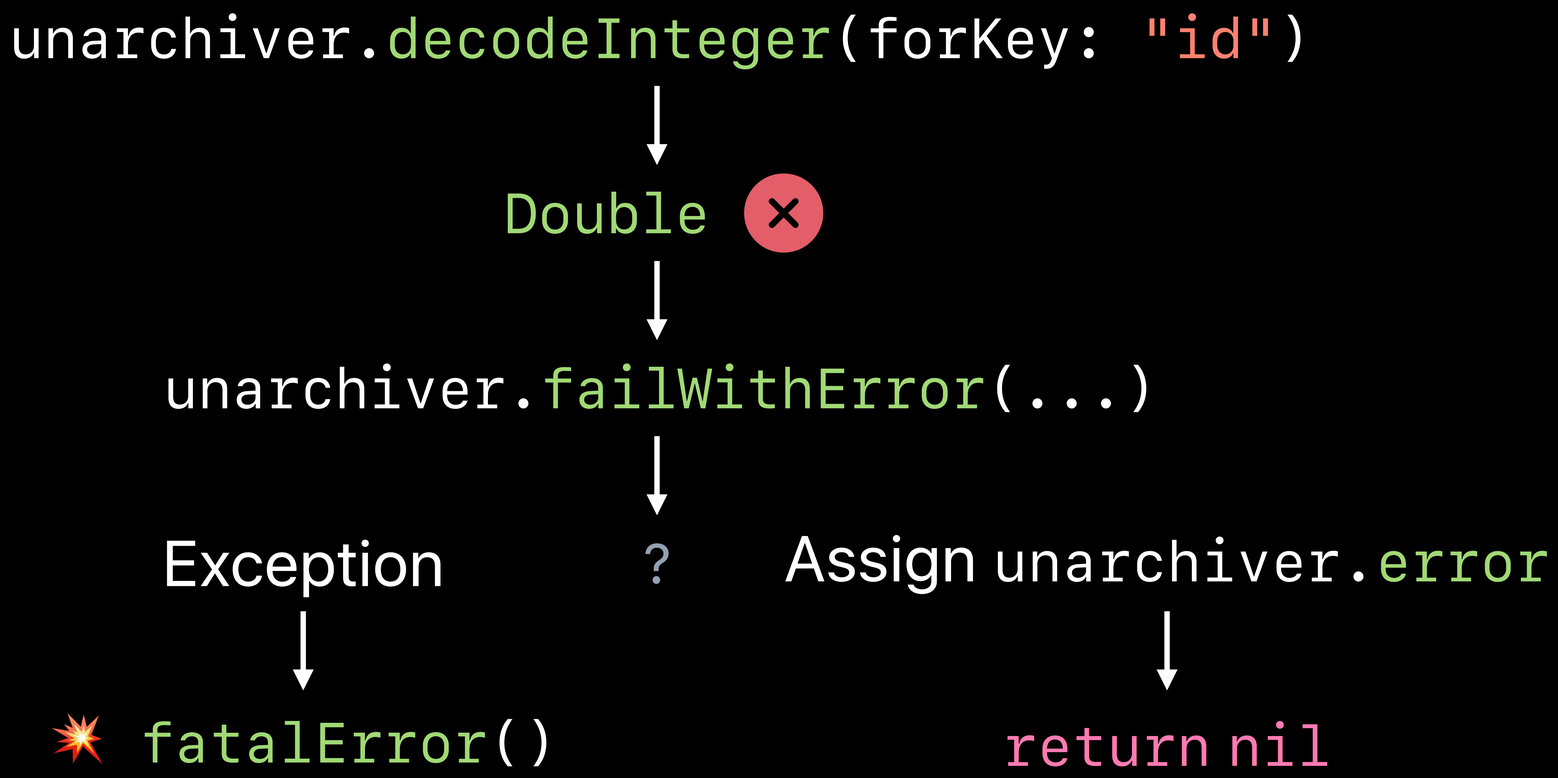
Decoding Failures



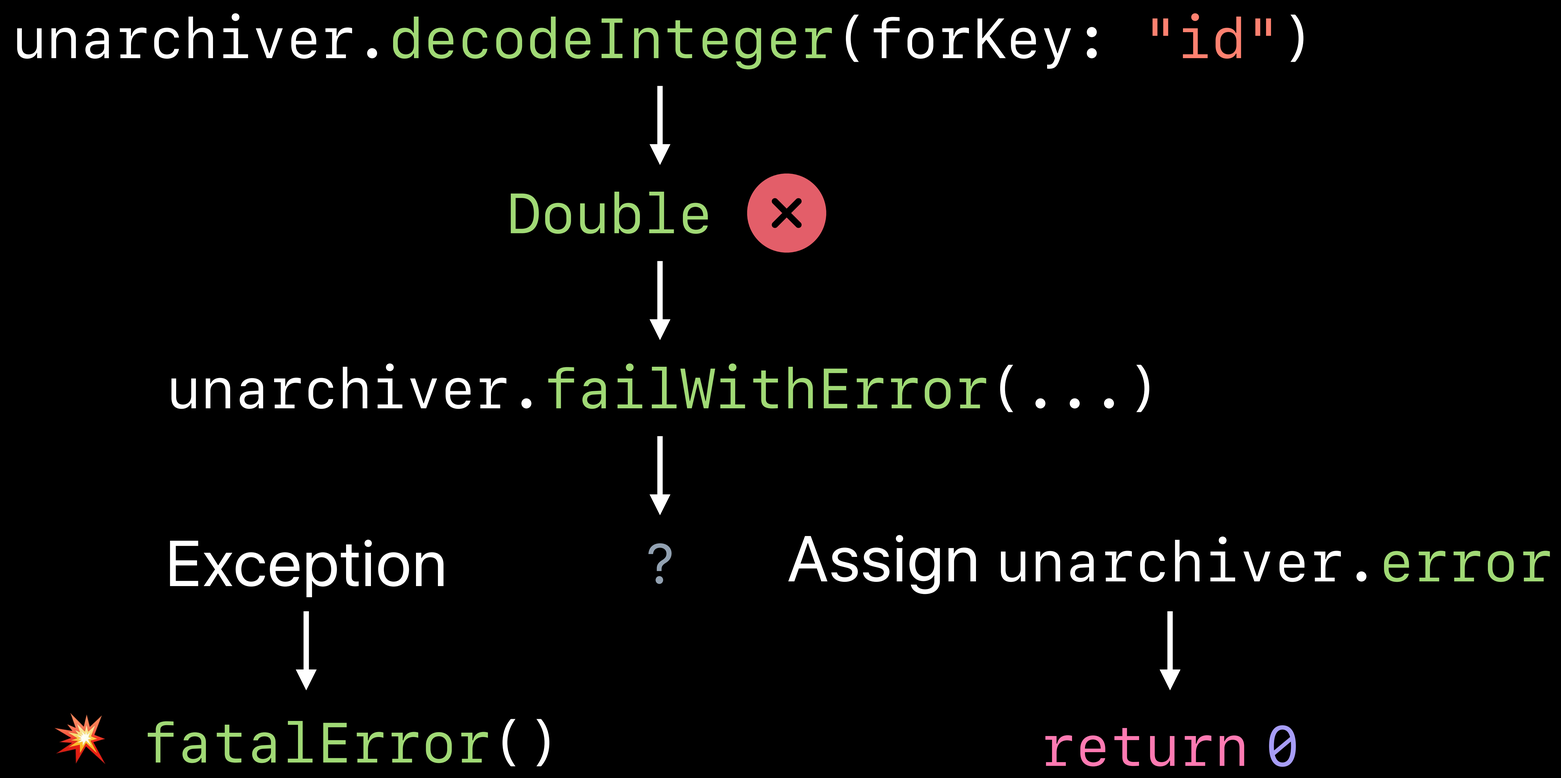
Decoding Failures



Decoding Failures



Decoding Failures



Failing Meaningfully

Use `failWithError(_:)` in your own code

Keep in mind

- Once an error is set, it is not reset
- Expect either an exception or continued execution
- `nil/0` could be because value is missing—always check `.error`

NSSecureCoding Adoption Checklist

- ☐ Translate all `decodeObject(forKey:)` calls to `decodeObject(of:forKey:)`
- ☐ Fail meaningfully
- ☐ Audit for further validation points

```
// Translate All decodeObject(forKey:) Calls to decodeObject(of:forKey:)
```

```
guard let listing = coder.decodeObject(forKey: "listing")
                                                as? Listing else {
    return nil
}
```

```
// Translate All decodeObject(forKey:) Calls to decodeObject(of:forKey:)
```

```
guard let listing = coder.decodeObject(of: Listing.self,
                                       forKey: "listing") else {
    return nil
}
```

```
// Fail Meaningfully
```

```
guard let listing = coder.decodeObject(of: Listing.self,
                                       forKey: "listing") else {
    return nil
}
```

```
// Fail Meaningfully
```

```
guard let listing = coder.decodeObject(of: Listing.self,
                                       forKey: "listing") else {
    let error = CocoaError.error(.coderValueNotFound, ...)
    coder.failWithError(error)
    return nil
}
```

```
// Fail Meaningfully
```

```
guard let listing = coder.decodeObject(of: Listing.self,
                                       forKey: "listing") else {
    let error = CocoaError.error(.coderValueNotFound, ...)
    coder.failWithError(error)
    return nil
}
```

```
// Audit for Further Validation Points



        guard let date = coder.decodeObject(of: NSDate.self,
                                            forKey: "purchaseDate") else {
            // ...
        }

        self.purchaseDate = date
```

```
// Audit for Further Validation Points


        guard let date = coder.decodeObject(of: NSDate.self,
                                            forKey: "purchaseDate") else {
            // ...
        }

        guard Purchase.isValidPurchaseDate(date) else {




        self.purchaseDate = date
```

```
// Audit for Further Validation Points



        guard let date = coder.decodeObject(of: NSDate.self,
                                            forKey: "purchaseDate") else {
            // ...
        }

        guard Purchase.isValidPurchaseDate(date) else {
            coder.failWithError(CocoaError.error(.coderReadCorrupt, ...))
            return nil
        }

        self.purchaseDate = date
```

NSSecureCoding Adoption Checklist

- ☑ Translate all `decodeObject(forKey:)` calls to `decodeObject(of:forKey:)`
- ☑ Fail meaningfully
- ☑ Audit for further validation points

NSSecureCoding Adoption Checklist

- ☑ Translate all `decodeObject(forKey:)` calls to `decodeObject(of:forKey:)`
- ☑ Fail meaningfully
- ☑ Audit for further validation points

```
class Purchase : NSCoding {
    // ...
    func encode(with coder: NSCoder) { ... }
    required init?(coder: NSCoder) { ... }
}
```

NSSecureCoding Adoption Checklist

- ☑ Translate all `decodeObject(forKey:)` calls to `decodeObject(of:forKey:)`
- ☑ Fail meaningfully
- ☑ Audit for further validation points

```
class Purchase : NSCoding {
    // ...
    func encode(with coder: NSCoder) { ... }
    required init?(coder: NSCoder) { ... }
    class var supportsSecureCoding: Bool { return true }
}
```

NSSecureCoding Adoption Checklist

- ☑ Translate all `decodeObject(forKey:)` calls to `decodeObject(of:forKey:)`
- ☑ Fail meaningfully
- ☑ Audit for further validation points

```
class Purchase : NSSecureCoding {
    // ...
    func encode(with coder: NSCoder) { ... }
    required init?(coder: NSCoder) { ... }
    class var supportsSecureCoding: Bool { return true }
}
```

NSSecureCoding Adoption Checklist

- ☑ Translate all `decodeObject(forKey:)` calls to `decodeObject(of:forKey:)`
- ☑ Fail meaningfully
- ☑ Audit for further validation points

Secure-by-Default APIs

NEW

New `NSKeyedUnarchiver` APIs enable secure coding by default

```
init(forReadingFrom: Data) throws
static func unarchivedObject<Class>(ofClass: Class.Type, from: Data) throws -> Class?
static func unarchivedObject(ofClasses: [AnyClass], from: Data) throws -> Any?
```

Default failure policy is `.setErrorAndReturn`

Old initializer and convenience methods are deprecated

Secure-by-Default APIs

NEW

New `NSKeyedArchiver` APIs to make it easier to enable secure coding

```
init(requiringSecureCoding: Bool)
class func archivedData(withRootObject: Any, requiringSecureCoding: Bool) throws -> Data
```

Old initializer and convenience methods are deprecated

// Secure-by-Default APIs

```
let data = NSKeyedArchiver.archivedData(withRootObject: myObject)
let decoded = NSKeyedUnarchiver.unarchiveObject(from: data)
```

// Secure-by-Default APIs

```
let data = NSKeyedArchiver.archivedData(withRootObject: myObject,
                                        requiringSecureCoding: true)

let decoded = NSKeyedUnarchiver.unarchiveObject(from: data)
```

// Secure-by-Default APIs

```
let data = NSKeyedArchiver.archivedData(withRootObject: myObject,
                                        requiringSecureCoding: true)

let decoded = NSKeyedUnarchiver.unarchivedObject(of: MyClass.self,
                                                 from: data)
```

// Secure-by-Default APIs

```
let data = NSKeyedArchiver.archivedData(withRootObject: myObject,
                                        requiringSecureCoding: true)

let decoded = NSKeyedUnarchiver.unarchivedObject(of: MyClass.self,
                                                 from: data)
```

```
// Secure-by-Default APIs

let data = NSKeyedArchiver.archivedData(withRootObject: myObject,
                                        requiringSecureCoding: false)

let decoded = NSKeyedUnarchiver.unarchivedObject(of: MyClass.self,
                                                 from: data)
```

```
// Secure-by-Default APIs


        let data = NSKeyedArchiver.archivedData(withRootObject: myObject,
                                                requiringSecureCoding: false)

        let unarchiver = try NSKeyedUnarchiver(forReadingFrom: data)
        unarchiver.requiresSecureCoding = false
        // unarchiver.decodingFailurePolicy = .raiseException
```

```
// Secure-by-Default APIs


        let data = NSKeyedArchiver.archivedData(withRootObject: myObject,
                                                requiringSecureCoding: false)

        let unarchiver = try NSKeyedUnarchiver(forReadingFrom: data)
        unarchiver.requiresSecureCoding = false
        // unarchiver.decodingFailurePolicy = .raiseException

        let decoded = try unarchiver.decodeObject(of: MyClass.self,
                                                  forKey: NSKeyedArchiveRootObjectKey)
```

Codable

Codable

Leaves types in code, not in archives

Synthesized code gives type and nullability checking

Non-trivial types need further validation

```
{
  "id": 1,
  "name": "Guitar",
  "description": "Still has most of its strings!",
  "price": "41.99",
  "sold": false,
  "url": "https://example.com/products/1",
  "tags": [ "music", "electric" ],
  "created": "2018-05-01T10:39:04 -0700",
  "last_updated": "2018-05-04T16:47:39 -0700",
}
```

```
struct Listing : Codable {
    let id: Int
    let name: String
    let description: String
    let price: Decimal
    let sold: Bool
    let url: URL
    let tags: [Tag]
    let created: Date
    let lastUpdated: Date
}
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
```

```swift
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)



        // guard let id = listing["id"] as? Int else { throw ... }
        // guard id > 0 && id <= Int32.max else { throw ... }
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)




        // guard let id = listing["id"] as? Int else { throw ... }
        // guard id > 0 && id <= Int32.max else { throw ... }
```

```swift
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)



        // guard id > 0 && id <= Int32.max else { throw ... }
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)



        // guard id > 0 && id <= Int32.max else { throw ... }
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else {
            throw DecodingError.dataCorruptedError(forKey: .id, in: container,
                                                   debugDescription: "id out of bounds")
        }
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else {
            throw DecodingError.dataCorruptedError(forKey: .id, in: container,
                                                   debugDescription: "id out of bounds")
        }
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        // guard let createdString = listing["created"] as? String,
        //     let created = formatter.date(from: createdString) else { throw ... }
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        // guard let createdString = listing["created"] as? String,
        //     let created = formatter.date(from: createdString) else { throw ... }
```

```swift
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        // guard let createdString = listing["created"] as? String,
        //     let created = formatter.date(from: createdString) else { throw ... }
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        self.created = try container.decode(Date.self, forKey: .created)
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        self.created = try container.decode(Date.self, forKey: .created)
        // decoder.dateDecodingStrategy = .iso8601
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        self.created = try container.decode(Date.self, forKey: .created)
        self.lastUpdated = try container.decode(Date.self, forKey: .lastUpdated)
```

```swift
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        self.created = try container.decode(Date.self, forKey: .created)
        self.lastUpdated = try container.decode(Date.self, forKey: .lastUpdated)
        guard self.lastUpdated >= self.created else { throw ... }
```

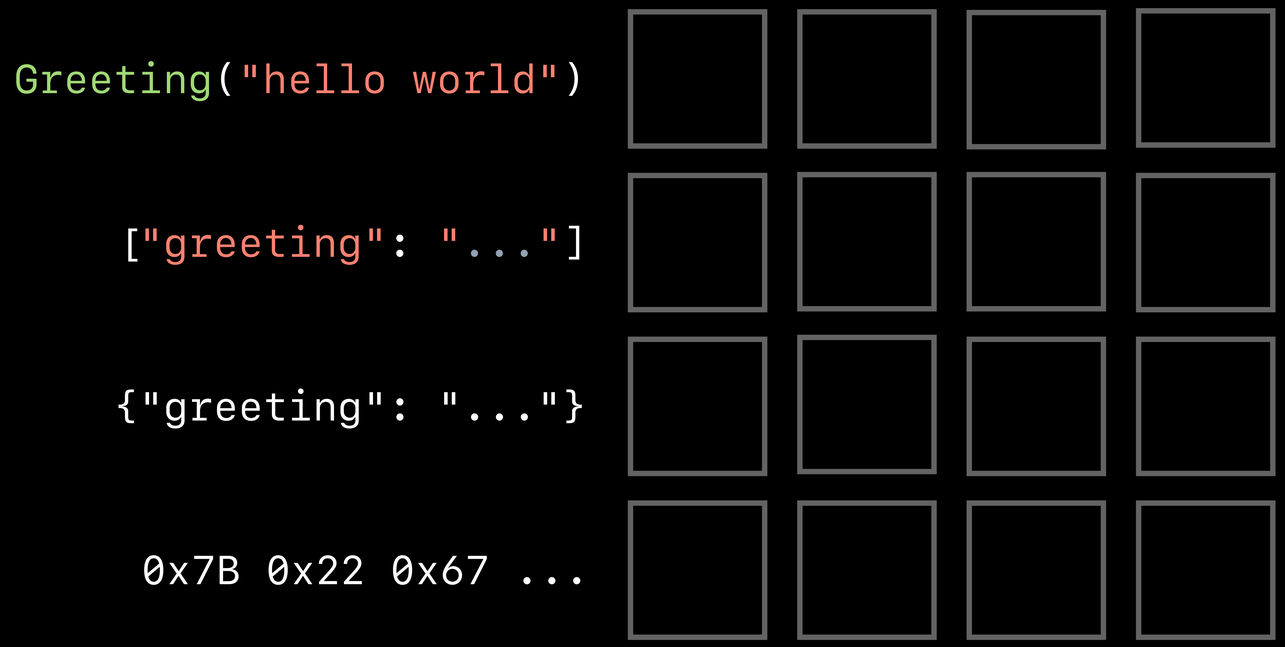
```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        self.created = try container.decode(Date.self, forKey: .created)
        self.lastUpdated = try container.decode(Date.self, forKey: .lastUpdated)
        guard self.lastUpdated >= self.created else { throw ... }

        // guard let tagStrings = listing["tags"] as? [String],
        //     let tags = tagStrings.map { Tag(rawValue: $0) } else { throw ... }
        // ...
    }
}
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        self.created = try container.decode(Date.self, forKey: .created)
        self.lastUpdated = try container.decode(Date.self, forKey: .lastUpdated)
        guard self.lastUpdated >= self.created else { throw ... }

        // guard let tagStrings = listing["tags"] as? [String],
        //     let tags = tagStrings.map { Tag(rawValue: $0) } else { throw ... }
        // ...
    }
}
```

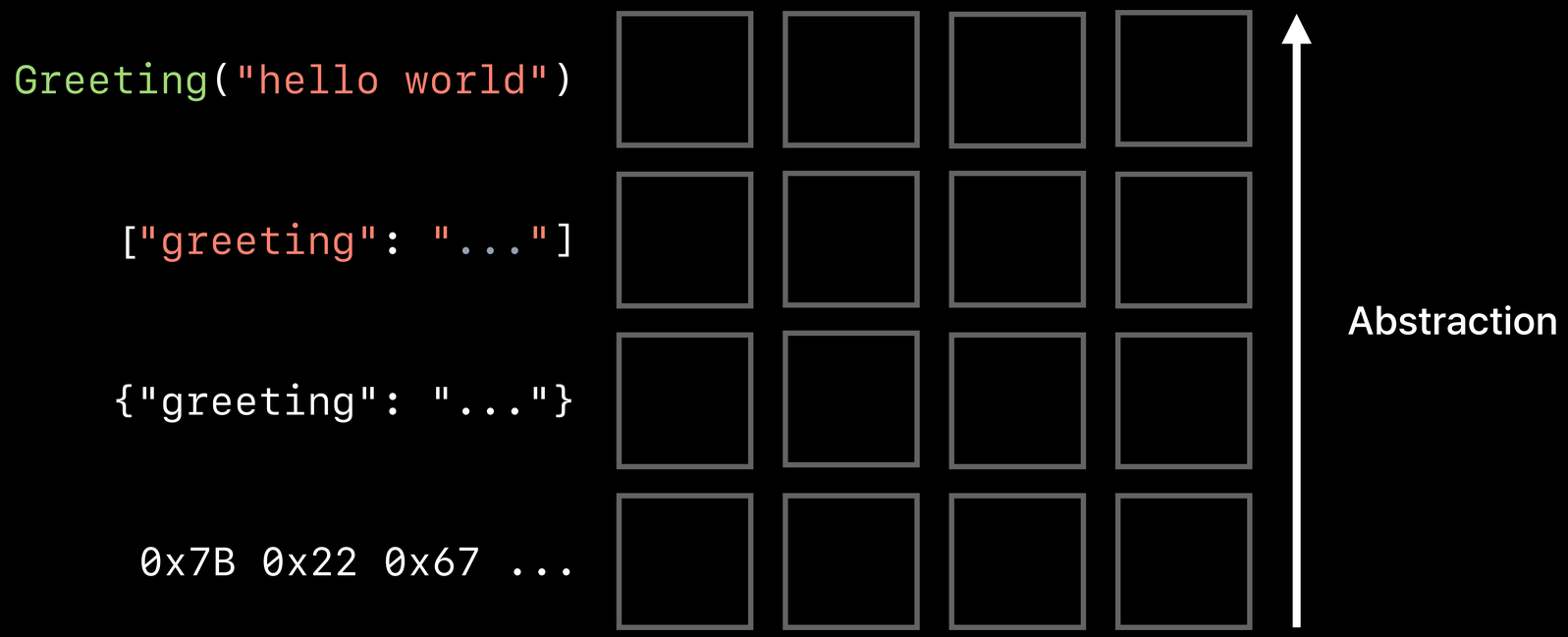
```swift
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        self.created = try container.decode(Date.self, forKey: .created)
        self.lastUpdated = try container.decode(Date.self, forKey: .lastUpdated)
        guard self.lastUpdated >= self.created else { throw ... }

        // guard let tagStrings = listing["tags"] as? [String],
        //     let tags = tagStrings.map { Tag(rawValue: $0) } else { throw ... }
        // ...
    }
}
```

```
// Validation of Codable Types

struct Listing : Codable {
    init(from decoder: Decoder) throws {
        let container = try decoder.container(keyedBy: CodingKeys.self)
        self.id = try container.decode(Int.self, forKey: .id)
        guard self.id > 0 && self.id <= Int32.max else { throw ... }

        self.created = try container.decode(Date.self, forKey: .created)
        self.lastUpdated = try container.decode(Date.self, forKey: .lastUpdated)
        guard self.lastUpdated >= self.created else { throw ... }

        self.tags = try container.decode([Tag].self, forKey: .tags)

        // ...
    }
}
```

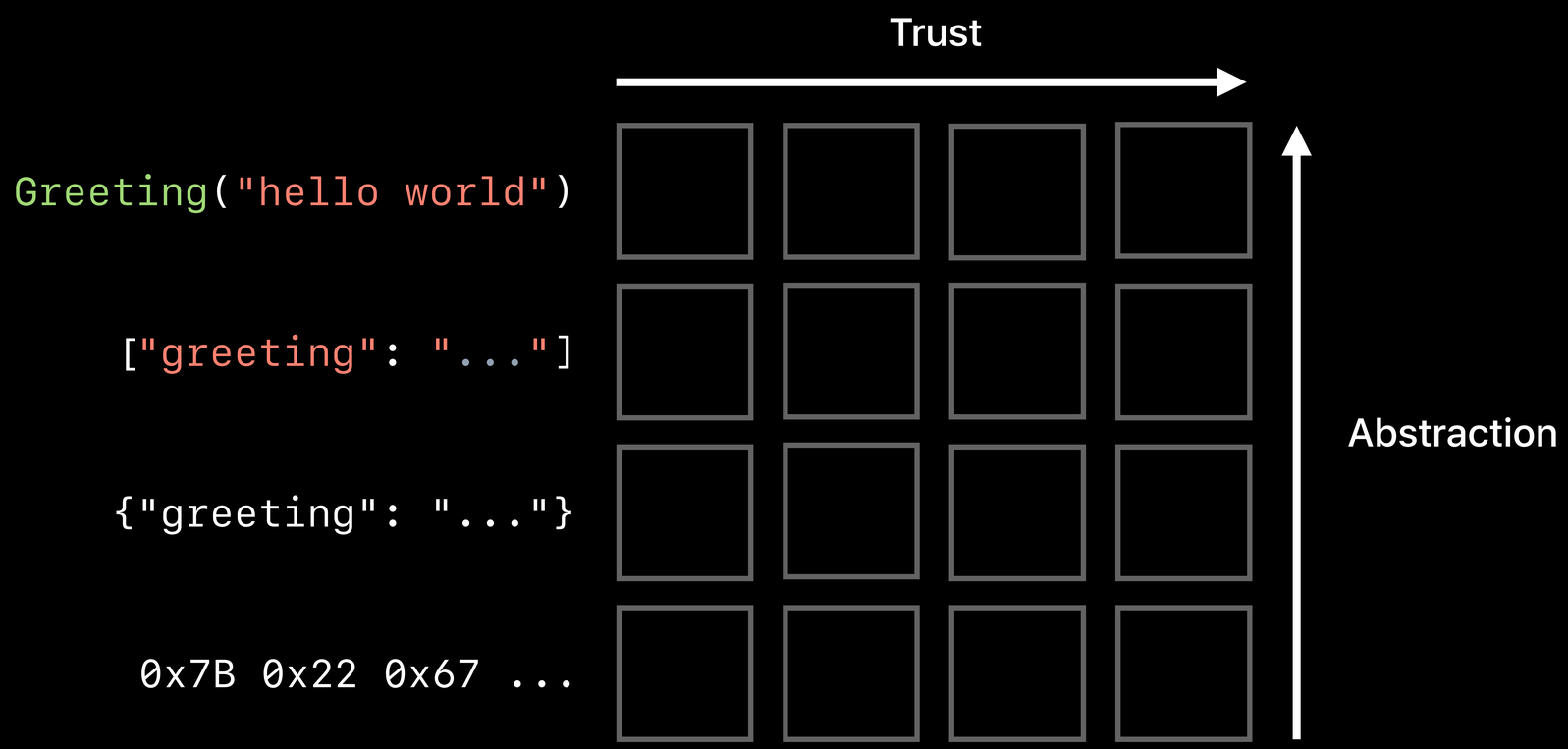
Summary



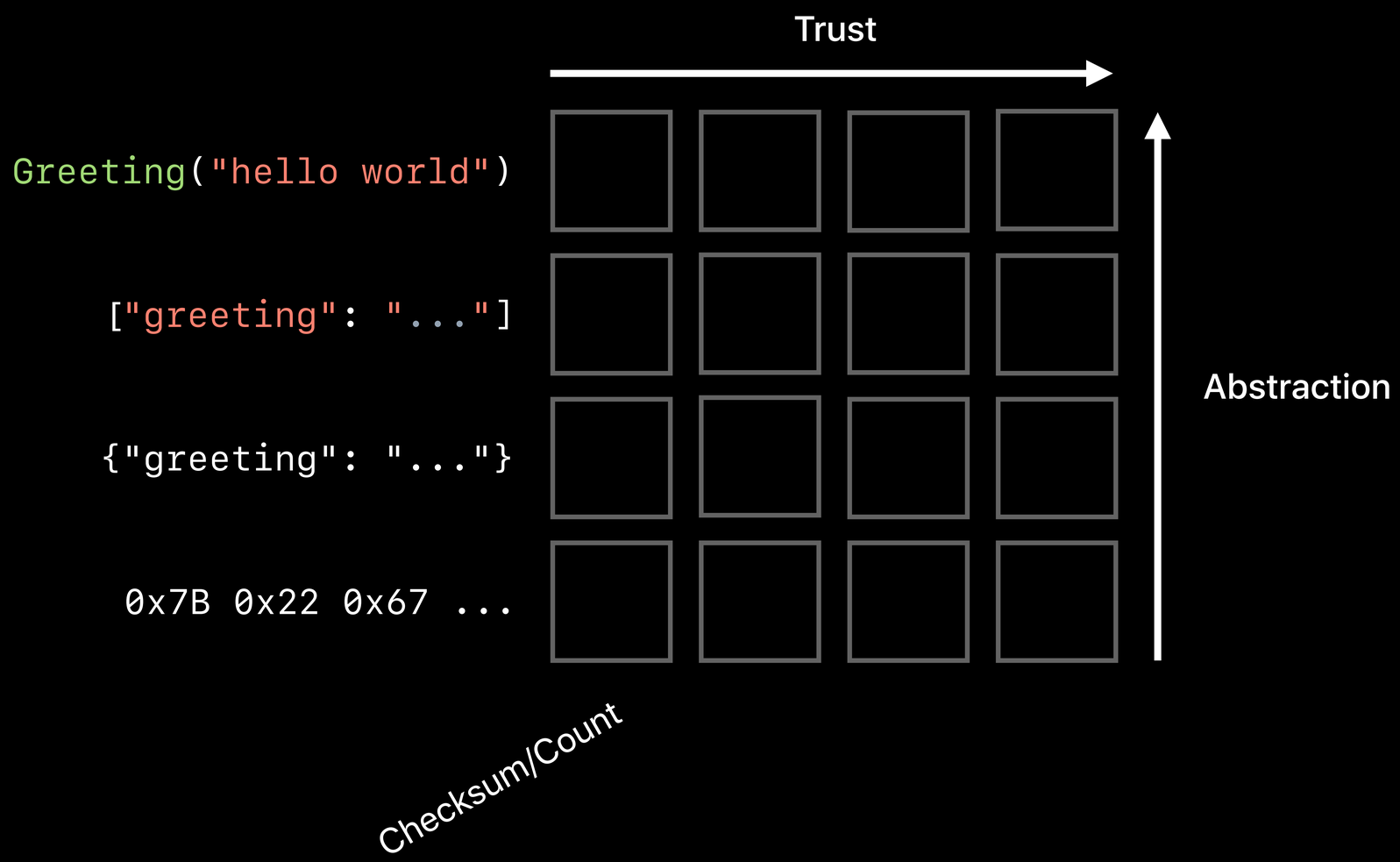
Summary


Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
Abstraction

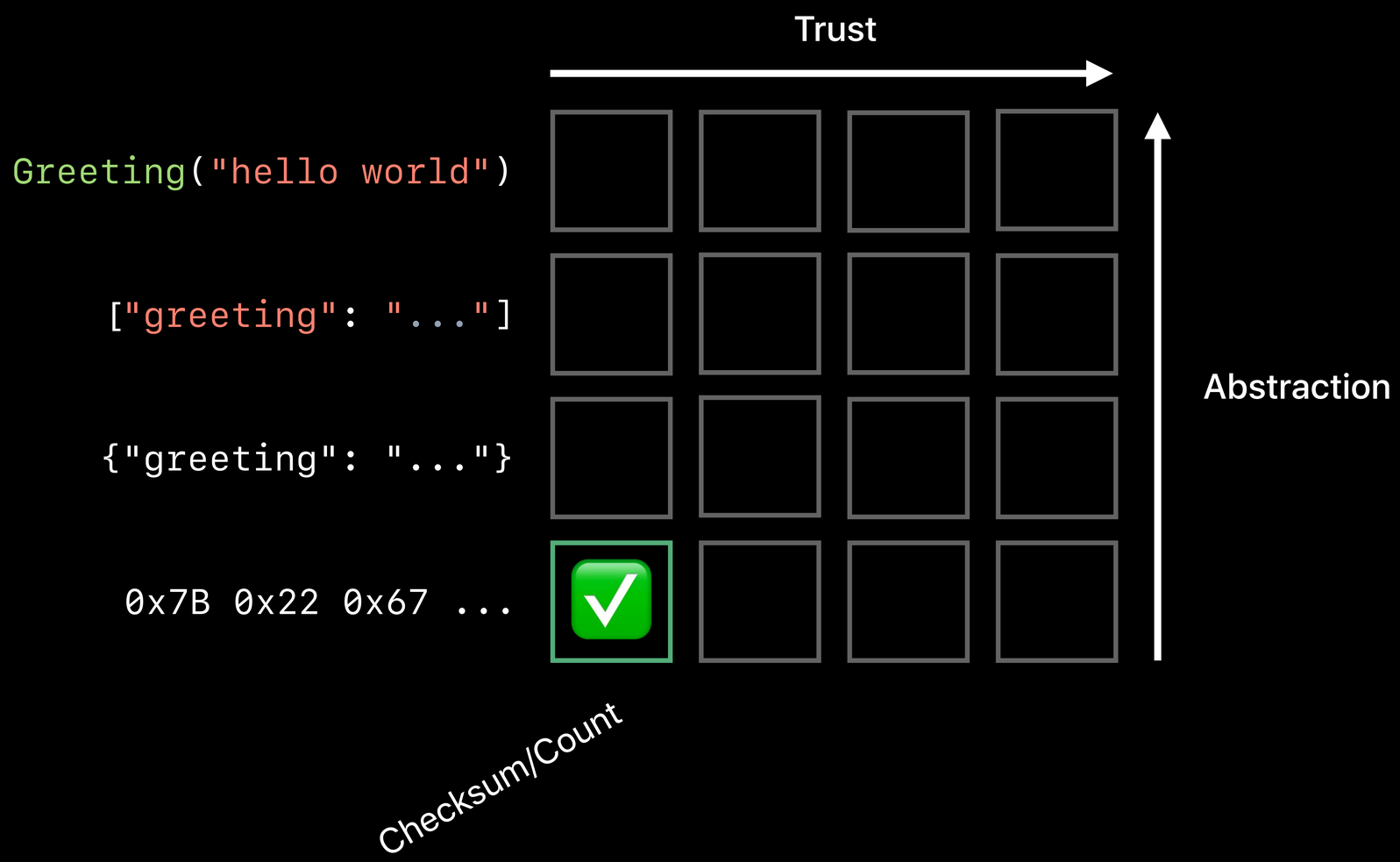
Summary


Trust
Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
Abstraction

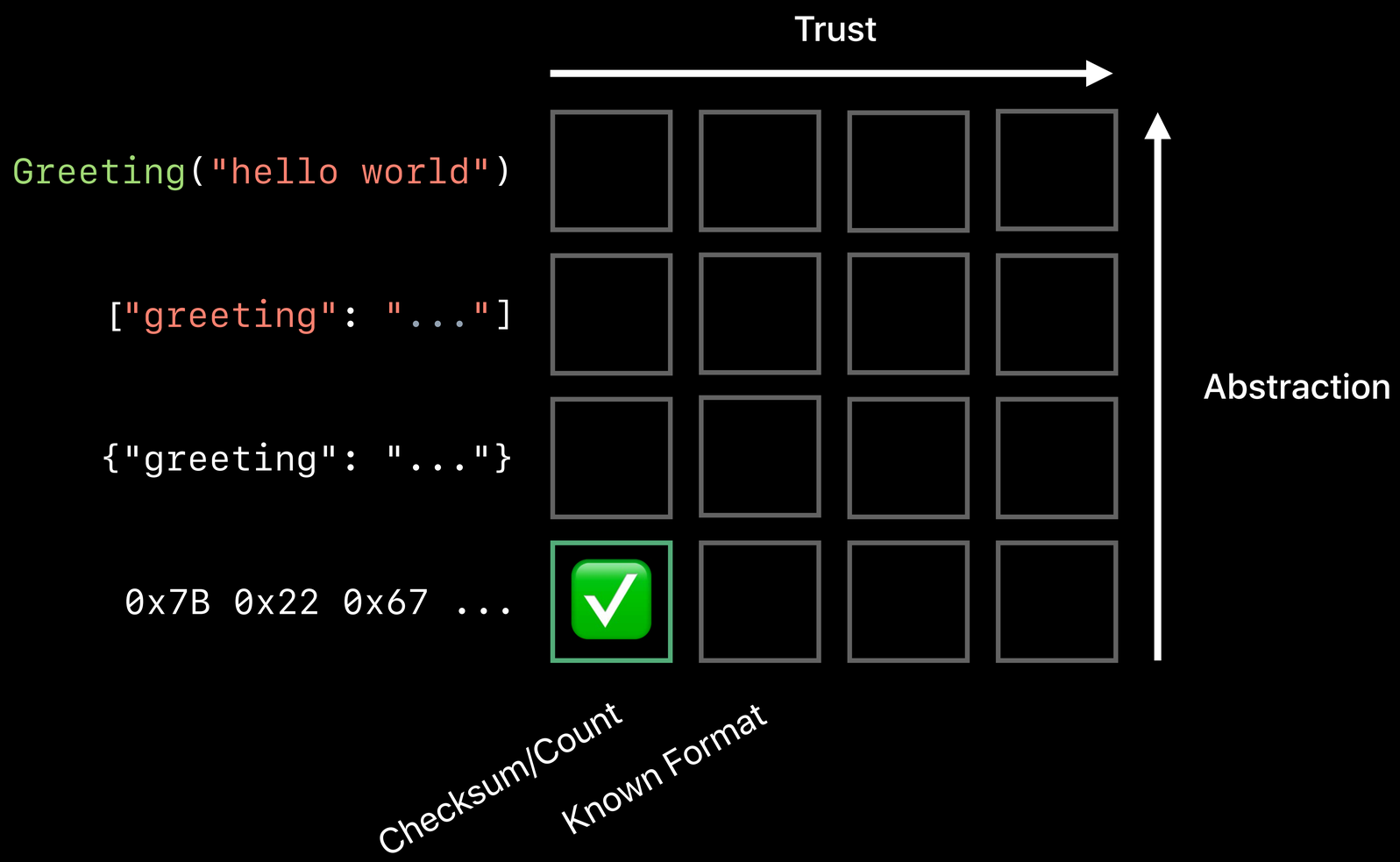
Summary


Trust
Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
Abstraction
Checksum/Count

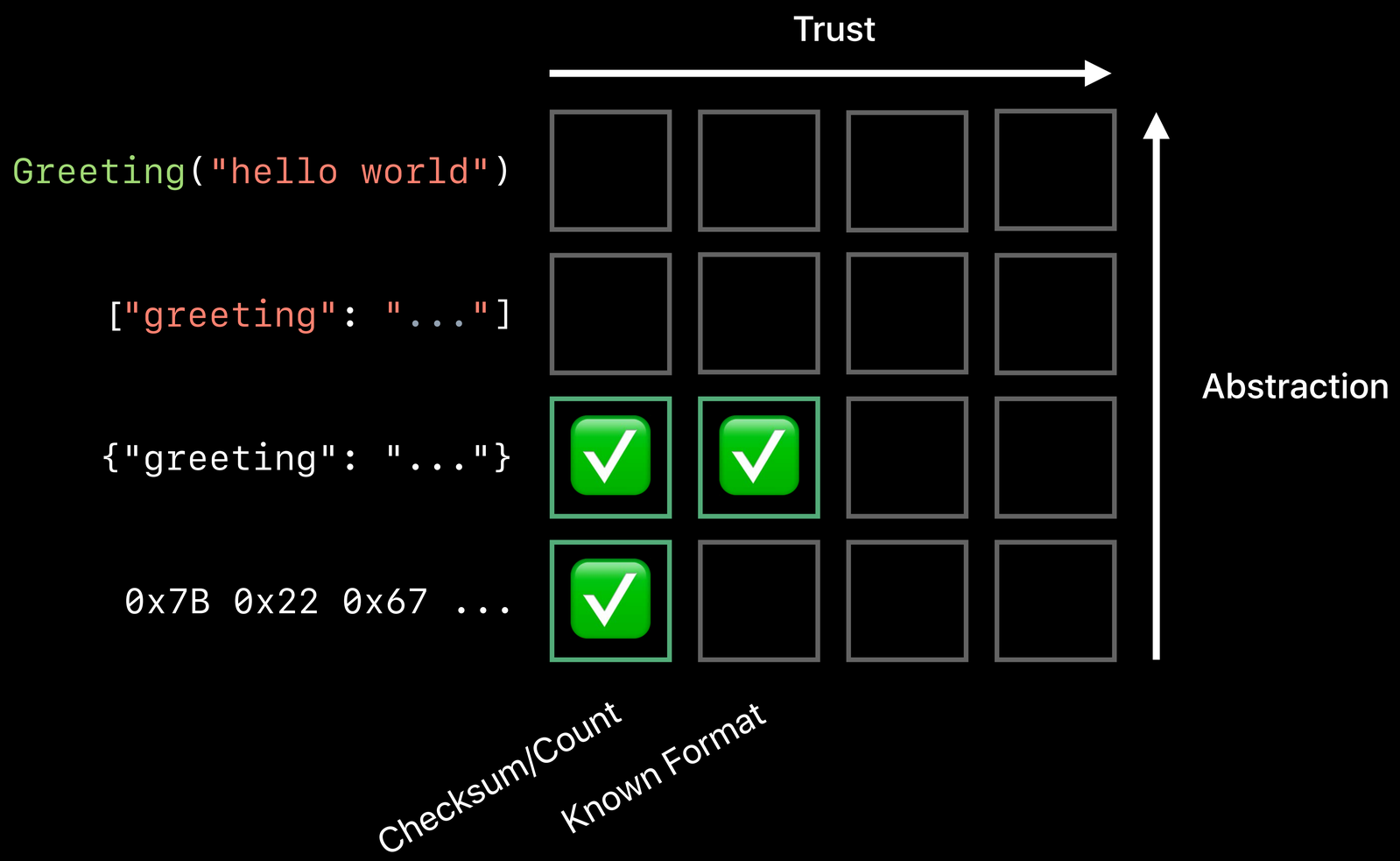
Summary


Trust
Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
✅
Abstraction
Checksum/Count

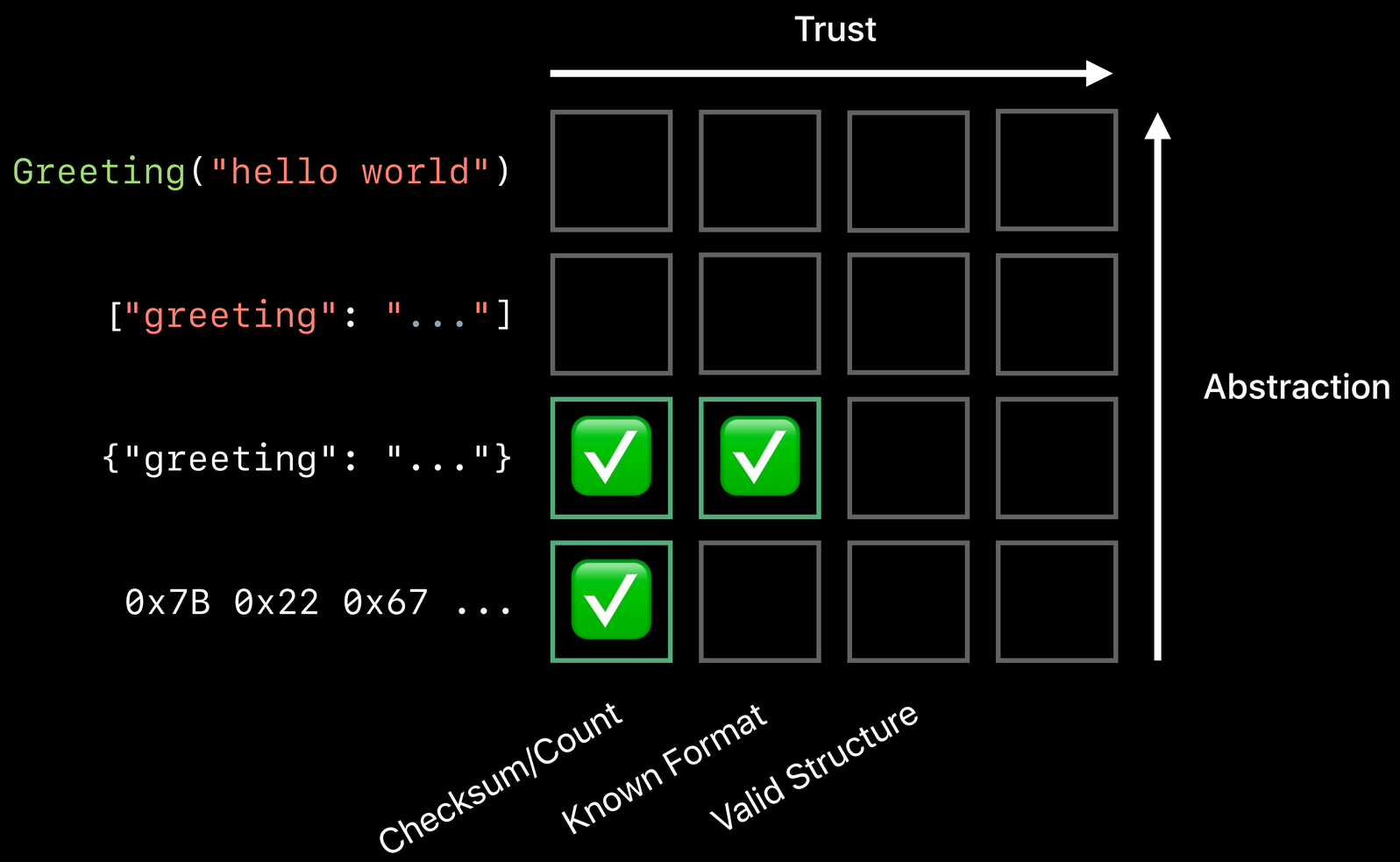
Summary


Trust
Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
✅
Abstraction
Checksum/Count
Known Format

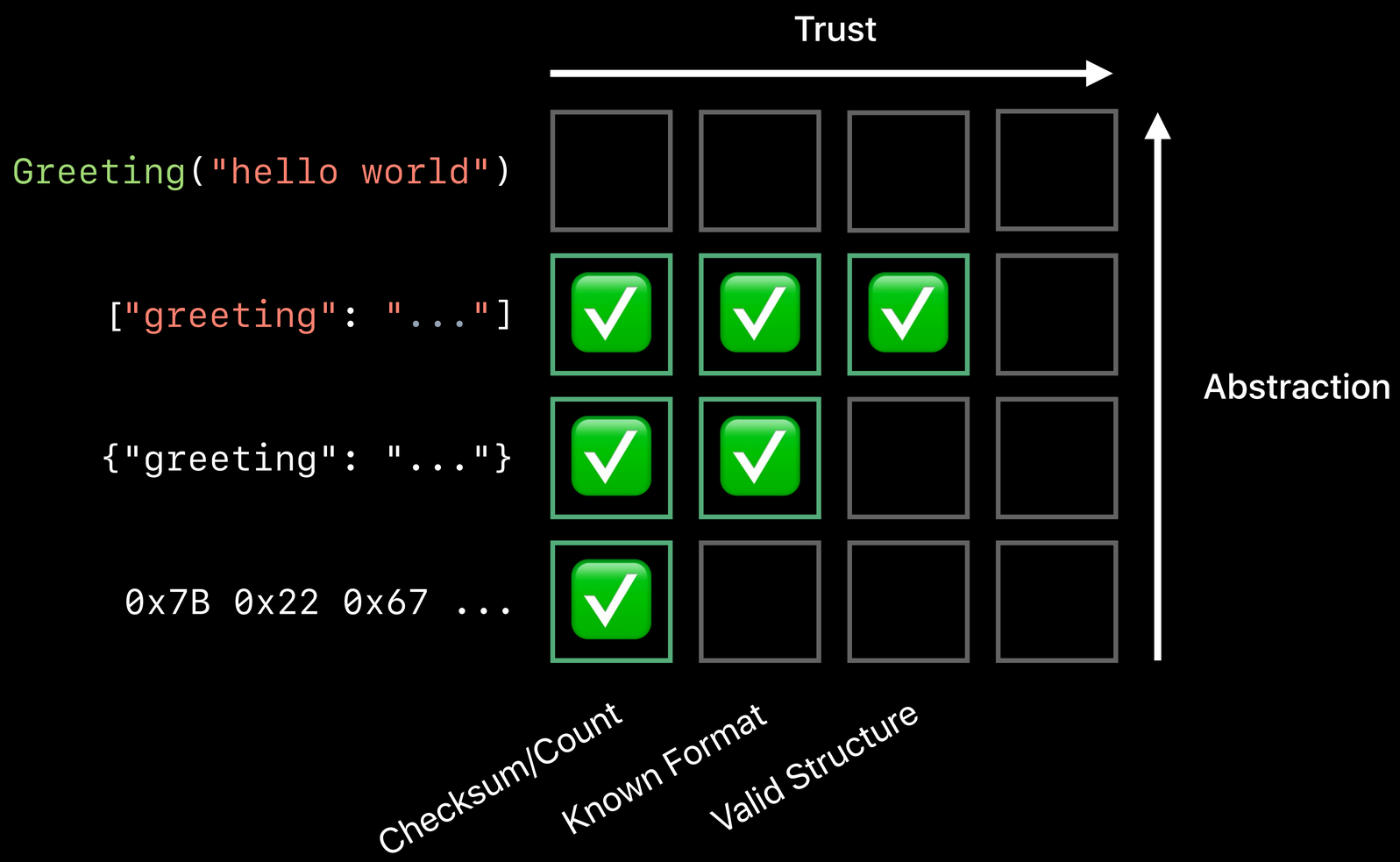
Summary


Trust
Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
Abstraction
Checksum/Count
Known Format

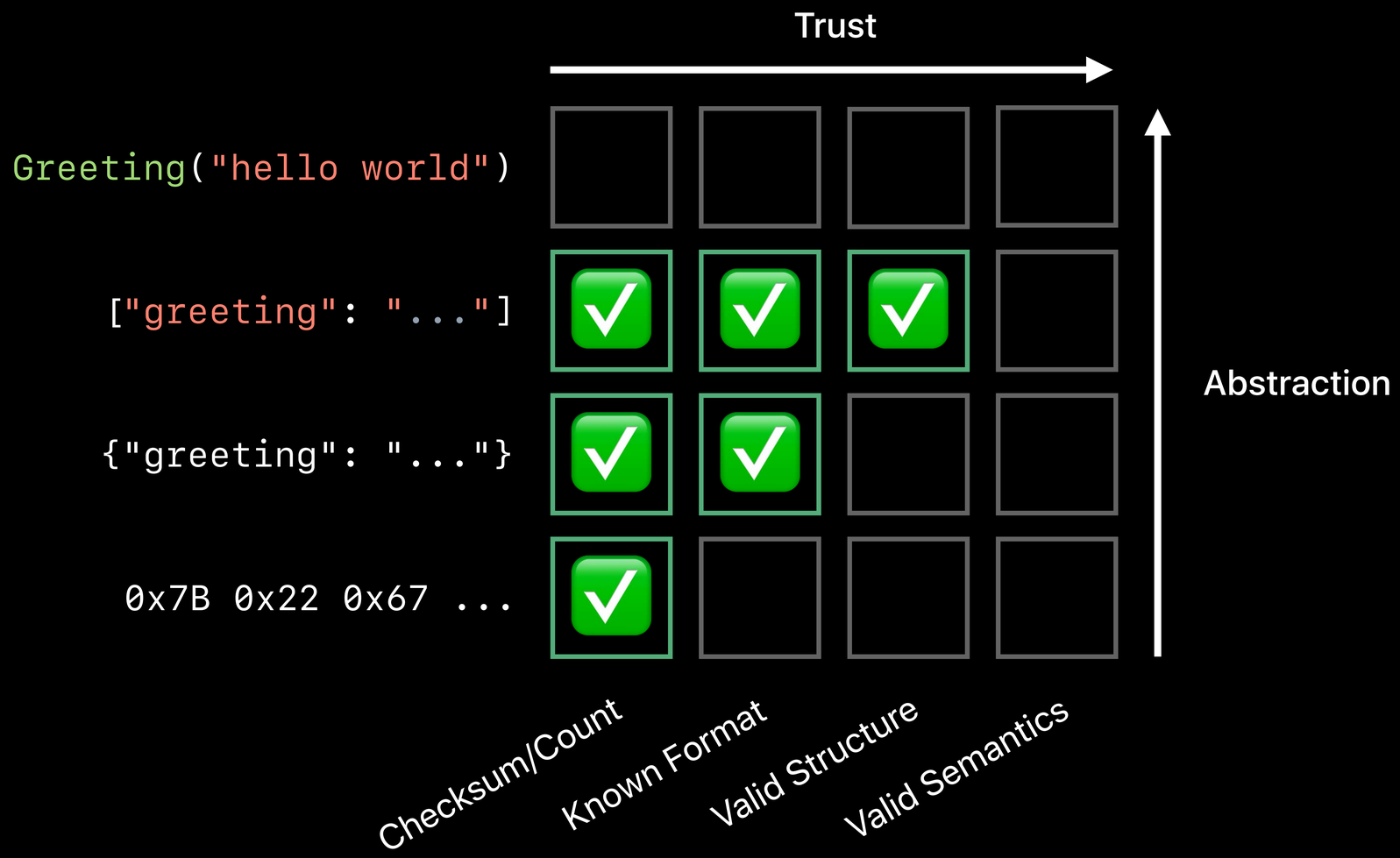
Summary


Trust
Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
Abstraction
Checksum/Count
Known Format
Valid Structure

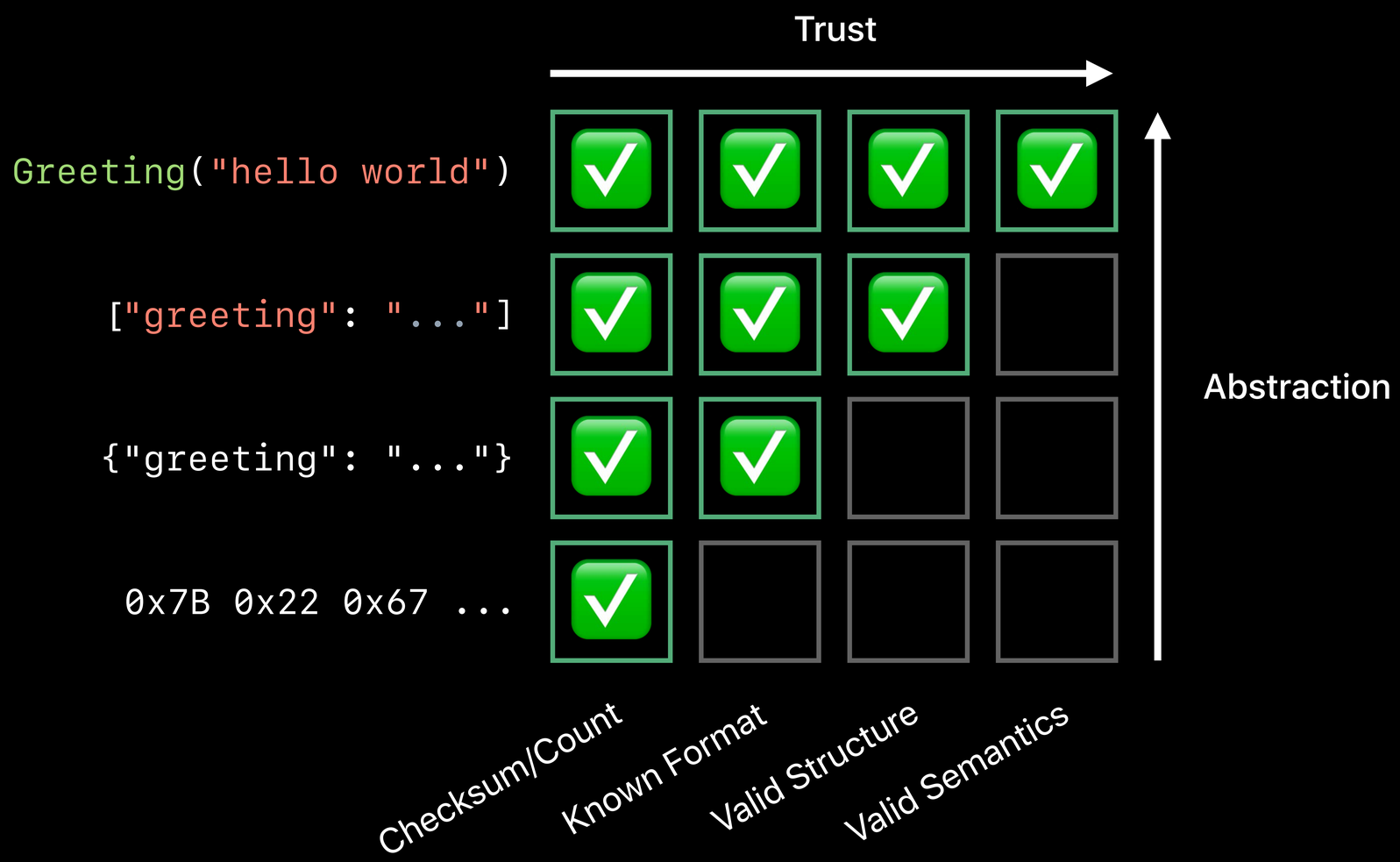
Summary


Trust
Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
Abstraction
Checksum/Count
Known Format
Valid Structure

Summary


Trust
Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
Abstraction
Checksum/Count
Known Format
Valid Structure
Valid Semantics

Summary


Trust
Greeting("hello world")
["greeting": "..."]
{"greeting": "..."}
0x7B 0x22 0x67 ...
Abstraction
Checksum/Count
Known Format
Valid Structure
Valid Semantics

Next Steps

Validate data at every stage

Check types, ranges, lengths, and domain

Audit `NSCoding` types and earn your `NSSecureCoding` badge

Adopt `Codable` for new data

More Information

https://developer.apple.com/wwdc2018/222


WWDC18