1

IOS Development Interview Questions

What is the URLSession API in iOS?

The URLSession API is a framework provided by Apple in iOS for making network requests. It provides a way to download or upload data from a remote server or web service.

```swift
let session = URLSession.shared
let url = URL(string: "https://example.com/data.json")!
let request = URLRequest(url: url)
let task = session.dataTask(with: request) { data, response, error in
    // handle response
}
task.resume()
```

2

IOS Development Interview Questions

What is the difference between synchronous and asynchronous networking?

Synchronous networking blocks the application until a response is received, while asynchronous networking allows the application to continue executing while waiting for a response.

```swift
let url = URL(string: "https://example.com/data.json")!

// Synchronous networking
let data = try? Data(contentsOf: url)

// Asynchronous networking
let task = URLSession.shared.dataTask(with: url) { data, response, error in
    // handle response
}
task.resume()
```

3

IOS Development Interview Questions

How can you implement background downloads in iOS?

Background downloads can be implemented using URLSession API with background configuration and background tasks.

```swift
// Create a URL session configuration that allows background downloads
let config = URLSessionConfiguration
    .background(withIdentifier: "com.example.myapp.backgroundDownload")
let session = URLSession(configuration: config, delegate: self, delegateQueue: nil)

// Create a download task for the desired URL
let url = URL(string: "https://example.com/myfile.mp3")!
let task = session.downloadTask(with: url)

// Start the download task
task.resume()
```

4

IOS Development Interview Questions

What is a completion handler in iOS networking, and how is it used?

A completion handler is a block of code that can be passed as an argument to a function or method, and is called by that function or method when it completes its task.

```swift
func fetchUser(completion: @escaping (User) -> Void) {
    // Perform network request to fetch user data
    // ...
    let user = User(name: "Ferdous Mahmud", email: "admin@ferdous.tech")
    completion(user)
}

// Call the function with a completion handler
fetchUser { user in
    print("User name: \(user.name)")
    print("User email: \(user.email)")
}
```

5

IOS Development

Interview Questions

What is the purpose of HTTP headers in networking, and how are they used in iOS?

HTTP headers contain additional information about the request or response, such as content type, encoding, or authentication credentials.

```swift
let urlString = "https://example.com/api/users"
guard let url = URL(string: urlString) else { return }

var request = URLRequest(url: url)
request.setValue("application/json", forHTTPHeaderField: "Content-Type")
request.setValue("Bearer \(accessToken)", forHTTPHeaderField: "Authorization")

let session = URLSession.shared
let task = session.dataTask(with: request) { data, response, error in
    // Handle the response data or error here
}
task.resume()
```

6

IOS Development Interview Questions

What is a RESTful API, and how is it used in iOS networking?

A RESTful API is a web service that follows the principles of Representational State Transfer (**REST**) architecture. A RESTful API is **stateless**, meaning that each request to the API contains all the information necessary for the server to understand and process the request.

```swift
guard let url = URL(string: "https://example.com/api/users") else { return }

URLSession.shared.dataTask(with: url) { data, response, error in
    guard let data = data, error == nil else {
        // Handle the error here
        return
    }
    if let json = try? JSONSerialization.jsonObject(with: data, options: []) as? [String: Any] {
        // Process the JSON response here
    }
}.resume()
```

7

IOS Development

Interview Questions

How can you implement error handling in iOS networking?

Error handling in iOS networking can be implemented using try-catch blocks, error codes, or HTTP status codes.

```swift
func fetchData() {
    URLSession.shared.dataTask(with: url) { (data, response, error) in
        do {
            guard let data = data, let httpResponse = response as? HTTPURLResponse else {
                throw NetworkingError.invalidResponse
            }
            switch httpResponse.statusCode {
            case 200...299:
                // Parse the response data
            case 401:
                throw NetworkingError.authenticationError
            ....
            }
        } catch {
            print("Error: \(error.localizedDescription)")
        }
    }.resume()
}
```

8

IOS Development

Interview Questions

What is the purpose of HTTP methods like GET, POST, PUT, and DELETE, and how are they used in iOS networking?

HTTP methods like GET, POST, PUT, and DELETE are used to specify the type of operation being performed on the resource.

- GET: used to retrieve data from a resource
- POST: used to submit new data to a resource
- PUT: used to update existing data on a resource
- DELETE: used to remove data from a resource

```swift
// DELETE request
guard let url = URL(string: "https://example.com/api/data/1") else { return }
var request = URLRequest(url: url)
request.httpMethod = "DELETE"
```

9

IOS Development

Interview Questions

What is the difference between JSON and XML, and which one is preferred for iOS networking?

JSON and XML are both data interchange formats used in iOS networking, but JSON is generally preferred over XML.

- **JSON** is lightweight and easier to parse, making it faster and more efficient for mobile devices. It is also the more commonly used format in modern web development.

- **XML** is a more verbose format and requires more processing power to parse. However, it offers more advanced features like data validation and transformation.

10

IOS Development

Interview Questions

How can you implement network caching in iOS, and what are the benefits of doing so?

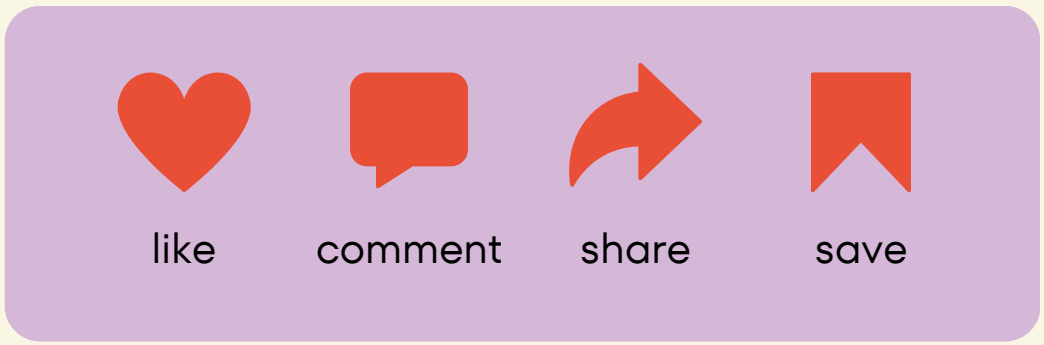
Network caching can improve performance and reduce data usage of iOS applications by storing frequently used data locally and avoiding unnecessary network requests.

```swift
// Set the caching policy to use the cache if available, otherwise make a network request
request.cachePolicy = .returnCacheDataElseLoad

let task = URLSession.shared.dataTask(with: request) { data, response, error in
    guard let data = data, let response = response else {
        // Handle the error
        return
    }

    // Store the response in the cache
    let cachedResponse = CachedURLResponse(response: response, data: data)
    cache.storeCachedResponse(cachedResponse, for: request)

    // Use the data
}
task.resume()
```

@ferdous19

Ferdous Mahmud Akash

WAS THIS POST HELPFUL?



Share the information with your friends if it was useful. Every like or comment helps promote the post. **Thank you!**

www.ferdous.tech